SIEMENS

SINUMERIK

SINUMERIK 840D sl / 828D 基本功能

功能手册

适用于

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

数控软件　版本 4.8 SP1

05/2017
A5E40870716

前言

基本安全说明 1

A2: 不同的 NC/PLC 接口信号与功能 2

A3:轴监控 3

A5：保护区 4

B1: 连续路径运行、准停、预读 5

B2：加速度 6

F1:运行到固定挡块 7

G2：速度、设定值/实际值系统、闭环控制 8

H2：输出到 PLC 的辅助功能 9

K1：BAG、通道、程序运行、复位特性 10

K2：轴、坐标系、框架 11

N2：急停 12

P1：端面轴 13

P3：SINUMERIK 840D sl 的 PLC 基本程序 14

P4：SINUMERIK 828D 的 PLC 15

R1：回参考点 16

S1：主轴 17

V1：进给率 18

W1：刀具补偿 19

Z1:NC/PLC 接口信号 20

附录 A

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 ® 的都是西门子股份有限公司的注册商标。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

A5E40870716
℗ 06/2017 本公司保留更改的权利

Copyright © Siemens AG 1994 - 2017.
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)。

目标使用人群

本手册供以下人员使用：

- 设计人员
- 机床制造商的技术人员
- 控制系统/机床的调试人员
- 编程人员

用途

本功能手册旨在介绍各项系统功能，使相关人员可以了解、选择功能，并掌握功能的调试步骤。

标准功能范畴

在本文档中描述了标准功能范畴。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档中未描述的某些功能。但这并不表示在交付系统时必须提供这些功能以及相关的维修服务。

同样，因为只是概要，所以该文档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所有情况。

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

结构

本手册由以下部分组成：

- 扉页（第 3 页），包含了本功能手册的标题、SINUMERIK 控制系统名称、系统软件、本手册版本针对的软件版本以及各功能说明一览。
- 功能说明，按字母和数字排序（例如 A2、A3、B1 等）。
- 附录，包含：
  - 缩写一览表
  - 手册一览
- 术语一览

说明

详细的数据和报警说明请见：

- 机床数据和设定数据：
  详细机床数据说明
- NC/PLC 接口信号：
  参数手册，NC 变量和接口信号
- 报警：
  诊断手册

系统数据的表示方法

本文档中，系统数据采用以下表示方法：

信号/数据	表示方法	示例
NC/PLC 接口信号	...NC/PLC 接口信号： <信号地址> (<信号名称>)	若切换了齿轮档，则会由 PLC 程序置位以下 NC/PLC 接口信号： DB31, ... DBX16.0-2（实际齿轮档 A 至 C） DB31, ... DBX16.3（齿轮档已切换）
机床数据	...机床数据： <类型><编号> <完整名称> (<含义>)	主主轴为机床数据中指定的主轴： MD20090 $MC_SPIND_DEF_MASTER_SPIND（通道中主主轴的缺省设置）
设定数据	...设定数据： <类型><编号> <完整名称> (<含义>)	逻辑主主轴包含在设定数据中： SD42800 $SC_SPIND_ASSIGN_TAB[0]（主轴编号转换器）

说明

信号地址

功能说明中指出的 NC/PLC 接口信号的地址仅仅是适用于 SINUMERIK 840D sl 的地址。适用于 SINUMERIK 828D 的信号地址请见相应功能说明末尾处的数据列表“信号发送至/来自...”。

组态范围

NC/PLC 接口的相关说明以下列特性的最大绝对数量为基础：

- 运行方式组（DB11）
- 通道（DB21, ...）
- 加工轴/主轴（DB31, ...）

数据类型

在控制系统中，以下数据类型用于在零件程序中进行编程：

类型	含义	取值范围
INT	整数，带符号	-2.147.483.648 ... +2.147.483.647
REAL	带小数点的实数	≈ $\pm 5.0*10^{-324}$ … ≈ $\pm 1.7*10^{+308}$

类型	含义	取值范围
BOOL	逻辑值	TRUE(≠0) , FALSE (0)
CHAR	ASCII 字符或字节	0 ... 255 或 -128 ... 127
STRING	字符串，以零结束	最多 400 个字符 + /0 （无特殊字符）
AXIS	轴名称	所有在控制系统中存在的轴名称
FRAME	用于平移、旋转、比例缩放和镜像的几何参数	---

数组

数组只可由相同的基本数据类型构成。最多可以为 3 维数组。

示例：`DEF INT FELD[2, 3, 4]`

数制

提供了下列数制：

- 十进制：`DEF INT NUMBER = 1234` 或 `DEF REAL NUMBER = 1234.56`
- 十六进制： `DEF INT NUMBER = 'H123ABC'`
- 二进制： `DEF INT NUMBER = 'B10001010010'`

询问 REAL 变量

建议在 NC 程序和同步动作中编程 REAL 或 DOUBLE 变量作为极限值监控。

示例：询问轴的实际值是否达到某个特定值

程序代码	注释
`DEF REAL AXPOS = 123.456`	
`IF ($VA_IM[<轴>] - 1ex-6) <= AXPOS <= ($VA_IM[<轴>] + 1ex-6)`	；实际位置
`...`	== AXPOS
`ELSE`	
`...`	<> AXPOS
`ENDIF`	

目录

前言 3

1 基本安全说明 41

1.1 一般安全说明 41

1.2 工业安全 41

2 A2: 不同的 NC/PLC 接口信号与功能 43

2.1 简要说明 43

2.2 NC/PLC 接口信号 43
2.2.1 简介 43
2.2.2 发送至 PLC 的就绪信号 45
2.2.3 发送至 PLC 的状态信号 46
2.2.4 发送至/来自操作面板的信号 47
2.2.5 发送至通道的信号 49
2.2.6 发送至进给轴/主轴的信号 49
2.2.7 来自进给轴/主轴的信号 58
2.2.8 发送至进给轴/主轴的信号（数字量驱动） 59
2.2.9 来自进给轴/主轴的信号（数字量驱动） 60

2.3 功能 63
2.3.1 过载时的图像更新特性 - 仅适用于 840D sl 63
2.3.2 渐开线插补的设置 - 只针对 840D sl 64
2.3.3 激活 DEFAULT 存储器 - 只针对 840D sl 67
2.3.4 读取和写入 PLC 变量 - 只针对 840D sl 67
2.3.5 通过口令和钥匙开关实现访问保护 71
2.3.5.1 口令 72
2.3.5.2 钥匙开关位置（DB10, DBX56.4 到 7） 74
2.3.5.3 可设置的保护等级 75
2.3.6 切换电机数据组和驱动数据组 76
2.3.6.1 简介 76
2.3.6.2 格式接口 76
2.3.6.3 请求接口 77
2.3.6.4 显示接口 78
2.3.6.5 示例 78
2.3.6.6 接口一览 79
2.3.6.7 前提条件 81

2.4 示例 82
2.4.1 参数组切换 82

2.5 数据表 84

2.5.1 机床数据.....84
2.5.1.1 显示机床数据.....84
2.5.1.2 NC 专用机床数据.....84
2.5.1.3 通道专用机床数据.....85
2.5.1.4 进给轴/主轴专用机床数据.....85
2.5.2 系统变量.....86
2.5.3 信号.....86
2.5.3.1 发送至 NC 的信号.....86
2.5.3.2 从 NC 发出的信号.....86
2.5.3.3 发送至操作面板的信号.....87
2.5.3.4 从操作面板发出的信号.....88
2.5.3.5 发送至通道的信号.....88
2.5.3.6 来自通道的信号.....88
2.5.3.7 发送至进给轴/主轴的信号.....89
2.5.3.8 来自进给轴/主轴的信号.....89

3 A3:轴监控.....91

3.1 轮廓监控.....91
3.1.1 轮廓误差.....91
3.1.2 跟随误差监控.....92

3.2 定位、静态和夹紧监控.....94
3.2.1 定位、静态和夹紧监控的相互关系.....94
3.2.2 定位监控.....94
3.2.3 静态监控.....96
3.2.4 和参数组相关的准停公差和静态公差.....97
3.2.5 夹紧监控.....97
3.2.5.1 功能.....97
3.2.5.2 机床数据.....98
3.2.5.3 NC/PLC 接口信号.....98
3.2.5.4 故障响应.....99
3.2.5.5 夹紧功能“自动停止以松开夹具”.....99
3.2.5.6 夹紧功能“在最佳时间松开夹具”.....101
3.2.5.7 夹紧功能“自动停止以设置夹具”.....102
3.2.5.8 前提条件.....104

3.3 转速设定值监控.....107

3.4 实际速度监控.....109

3.5 测量系统监控.....110
3.5.1 编码器极限频率监控.....112
3.5.2 绝对值编码器的合理性检查.....113
3.5.3 用户自定义的故障响应.....115

3.6 限位开关监控.....117
3.6.1 硬件限位开关.....117
3.6.2 软件限位开关.....118

3.7 工作区域限制...120
3.7.1 简介...120
3.7.2 BCS 中的工作区域限制...122
3.7.3 WCS/AZS 中的工作区域限制...125
3.7.4 示例：WCS/AZS 中的工作区域限制...128
3.8 “驻留”机床轴...131
3.9 被动位置测量系统驻留...133
3.9.1 功能...133
3.9.2 前提条件...136
3.9.3 示例： 直接位置测量系统上的附加测头切换...137
3.9.4 示例： 两个直接位置测量系统上的附加测头切换...142
3.9.5 示例： 编码器无法适用运行范围时的测量系统切换...147
3.10 边界条件...150
3.11 数据表...151
3.11.1 机床数据...151
3.11.1.1 NC 专用机床数据...151
3.11.1.2 通道专用机床数据...151
3.11.1.3 进给轴/主轴专用机床数据...152
3.11.2 设定数据...153
3.11.2.1 进给轴/主轴专用设定数据...153
3.11.3 信号...154
3.11.3.1 发送至进给轴/主轴的信号...154
3.11.3.2 来自进给轴/主轴的信号...154
4 A5：保护区...155
4.1 功能...155
4.2 调试...162
4.2.1 机床数据...162
4.3 编程...163
4.3.1 定义保护区 (CPROTDEF、NPROTDEF)...163
4.3.2 激活/取消激活保护区 (CPROT, NPROT)...167
4.3.3 检查保护区、工作区域限制和软件限位开关 (CALCPOSI)...171
4.4 特殊情形...182
4.4.1 暂时允许轴进入保护区...182
4.4.2 AUTO 和 MDA 运行模式下的特性...182
4.4.3 JOG 运行模式下的特性...183
4.5 边界条件...187
4.6 示例...188
4.6.1 车床上的保护区...188
4.6.2 零件程序中的保护区定义...189
4.6.3 通过系统变量定义保护区...189

4.6.4 激活保护区...199

4.7 数据表...200
4.7.1 机床数据...200
4.7.1.1 NC 专用机床数据...200
4.7.1.2 通道专用机床数据...200
4.7.2 信号...200
4.7.2.1 发送至通道的信号...200
4.7.2.2 来自通道的信号...201

5 B1: 连续路径运行、准停、预读...203

5.1 简要说明...203

5.2 准停运行...205

5.3 连续路径运行...210
5.3.1 一般功能...210
5.3.2 按过载系数降低速度...212
5.3.3 平滑...214
5.3.3.1 按照位移条件开展平滑(G641)...216
5.3.3.2 按照定义的公差开展平滑(G642/G643)...218
5.3.3.3 按照允许的最大动态响应进行平滑(G644)...222
5.3.3.4 相切程序段过渡的平滑(G645)...225
5.3.3.5 平滑和重新定位 (REPOS)...226
5.3.4 预读...227
5.3.4.1 标准功能...227
5.3.4.2 任意形状表面模式：扩展功能...233

5.4 动态响应自适应功能...237
5.4.1 轨迹速度平滑...237
5.4.2 轨迹动态响应自适应...240
5.4.3 确定动态响应极限值...244
5.4.4 “轨迹速度平滑”和“轨迹动态响应自适应”的共同作用...245
5.4.5 轨迹插补的动态响应模式...249
5.4.6 任意形状表面模式：基本功能...252

5.5 压缩功能...255
5.5.1 压缩线性程序段、圆弧程序段和快速运行程序段...255
5.5.1.1 功能...255
5.5.1.2 调试...257
5.5.1.3 编程...267
5.5.1.4 前提条件...268
5.5.2 压缩较短的样条程序段...269

5.6 轮廓/定向公差...271
5.6.1 调试...271
5.6.1.1 参数设置...271
5.6.2 编程...272

5.6.2.1 轮廓公差/定向公差的编程 (CTOL、OTOL、ATOL)....272
5.6.2.2 轮廓公差/定向公差的编程 (CTOL、OTOL、ATOL)：更多信息....275
5.7 G0 运行的公差系数....276
5.7.1 功能....276
5.7.2 调试....277
5.7.2.1 参数设置....277
5.7.3 编程....277
5.7.3.1 编程 G0 运行的公差系数 (STOLF)....277
5.8 复位特性....279
5.9 前提条件....280
5.9.1 程序段切换和定位轴....280
5.9.2 程序段切换延时....280
5.10 数据表....281
5.10.1 机床数据....281
5.10.1.1 通用机床数据....281
5.10.1.2 通道专用机床数据....281
5.10.1.3 进给轴/主轴专用机床数据....283
5.10.2 设定数据....283
5.10.2.1 通道专用设定数据....283
5.10.3 信号....284
5.10.3.1 来自通道的信号....284
5.10.3.2 来自进给轴/主轴的信号....284
6 B2：加速度....285
6.1 简要说明....285
6.1.1 简介....285
6.1.2 特性....285
6.2 功能....287
6.2.1 刚性加速模式 (BRISK/BRISKA)（通道/轴专用）....287
6.2.1.1 简介....287
6.2.1.2 参数设置....289
6.2.1.3 编程....290
6.2.2 恒速运行时间（通道专用）....291
6.2.2.1 简介....291
6.2.2.2 参数设置....292
6.2.3 加速度自适应 (ACC)（轴专用）....293
6.2.3.1 简介....293
6.2.3.2 编程....293
6.2.4 加速裕量（通道专用）....294
6.2.4.1 简介....294
6.2.4.2 参数设置....294
6.2.5 最大轨迹加速度（通道专用）....294
6.2.5.1 简介....294

6.2.5.2 参数设置....295
6.2.5.3 编程....295
6.2.6 针对实时事件的轨迹加速度（通道专用）....296
6.2.6.1 简介....296
6.2.6.2 编程....297
6.2.7 快速移动条件下的单独加速度 (G00)（轴专用）....298
6.2.7.1 简介....298
6.2.7.2 参数设置....298
6.2.8 软加速模式 (SOFT/SOFTA)（轴专用）....299
6.2.8.1 简介....299
6.2.8.2 参数设置....299
6.2.9 在非相切的程序段过渡条件下上调加速度（轴专用）....299
6.2.9.1 简介....299
6.2.9.2 参数设置....300
6.2.10 为径向加速度保留的加速裕量（通道专用）....300
6.2.10.1 简介....300
6.2.10.2 参数设置....301
6.2.11 轨迹插补中的最大急动度 (SOFT)（通道专用）....302
6.2.11.1 简介....302
6.2.11.2 参数设置....304
6.2.11.3 编程....304
6.2.12 单轴插补中的最大急动度 (SOFTA)（轴专用）....305
6.2.12.1 参数设置....305
6.2.12.2 编程....306
6.2.13 最大轨迹急动度（通道专用）....307
6.2.13.1 简介....307
6.2.13.2 参数设置....307
6.2.13.3 编程....307
6.2.14 针对实时事件的轨迹急动度（通道专用）....308
6.2.14.1 简介....308
6.2.14.2 编程....309
6.2.15 快速移动条件下的急动度 (G00)（轴专用）....310
6.2.15.1 简介....310
6.2.15.2 参数设置....310
6.2.16 在非恒定曲率的程序段过渡处自动上调急动度（轴专用）....311
6.2.16.1 简介....311
6.2.16.2 参数设置....311
6.2.17 随速度提高急动度（轴专用）....311
6.2.18 急动度滤波器（轴专用）....314
6.2.18.1 简介....314
6.2.18.2 参数设置....316
6.2.19 折线式加速度特性曲线....317
6.2.19.1 适应电机特性曲线....317
6.2.19.2 对轨迹加速度的影响....319
6.2.19.3 备用特性曲线....320

6.2.19.4 参数设置...322
6.2.19.5 编程...323
6.2.19.6 前提条件...324
6.2.20 JOG 运动中的加速度和急动度...325
6.2.20.1 参数设置...325
6.2.20.2 前提条件...327
6.3 示例...328
6.3.1 加速度...328
6.3.1.1 轨迹速度曲线...328
6.3.2 急动度...329
6.3.2.1 轨迹速度曲线...329
6.3.3 加速度和急动度...331
6.3.4 折线式加速度特性曲线...333
6.3.4.1 激活...333
6.4 数据表...334
6.4.1 机床数据...334
6.4.1.1 NC 专用机床数据...334
6.4.1.2 通道专用机床数据...334
6.4.1.3 进给轴/主轴专用机床数据...335
6.4.2 设定数据...336
6.4.2.1 通道专用设定数据...336
6.4.3 系统变量...336
7 F1:运行到固定挡块...337
7.1 简要说明...337
7.2 详细说明...338
7.2.1 编程...338
7.2.2 工作流程...340
7.2.2.1 选择...340
7.2.2.2 到达固定挡块...341
7.2.2.3 未到达固定挡块...343
7.2.2.4 取消...344
7.2.3 程序段搜索时的特性...346
7.2.4 复位和功能中断时的特性...350
7.2.5 与其他功能综合使用时的特性...351
7.2.6 设定数据...351
7.2.7 系统变量...353
7.2.8 报警...354
7.2.9 以受限转矩/力 FOC 运行...356
7.3 示例...359
7.4 数据表...360
7.4.1 机床数据...360
7.4.1.1 进给轴/主轴专用机床数据...360

7.4.2 设定数据....361
7.4.2.1 进给轴/主轴专用设定数据....361
7.4.3 信号....361
7.4.3.1 发送至进给轴/主轴的信号....361
7.4.3.2 来自进给轴/主轴的信号....362
8 G2：速度、设定值/实际值系统、闭环控制....363
8.1 简要说明....363
8.2 速度、运行范围、精度....363
8.2.1 速度....363
8.2.2 运行范围....365
8.2.3 控制系统的定位精度....366
8.2.4 输入/显示精度，计算精度....367
8.2.5 机床数据和设定数据物理量的定标....369
8.3 英制/公制单位制....372
8.3.1 功能....372
8.3.1.1 参数设置和编写的单位制....372
8.3.1.2 扩展单位制功能....373
8.3.1.3 通过操作界面切换单位制....374
8.3.2 调试....376
8.3.3 编程....380
8.3.3.1 切换单位制 (G70/G71/G700/G710)....380
8.4 设定值/实际值系统....384
8.4.1 简介....384
8.4.2 设定值和编码器分配....387
8.4.3 调整电机/负载齿轮比....391
8.4.4 转速设定值输出....394
8.4.5 实际值系统的机床数据....396
8.4.6 实际值精度....397
8.4.6.1 实际值分辨率的机床数据....397
8.4.6.2 示例：带光栅尺的线性轴....401
8.4.6.3 示例：电机上带旋转编码器的线性轴....401
8.4.6.4 示例： 机床上带旋转编码器的线性轴....403
8.4.6.5 示例：电机上带旋转编码器的回转轴....404
8.4.6.6 示例： 机床上带旋转编码器的回转轴....405
8.4.6.7 示例：刀具上带编码器的中间齿轮比....406
8.5 闭环控制....407
8.5.1 简介....407
8.5.2 位置控制器的参数组....410
8.6 控制优化....412
8.6.1 位置控制器，位置设定值滤波器：对称滤波器....412
8.6.2 位置控制器，位置设定值滤波器：急动度滤波器....415
8.6.3 位置控制器，位置设定值滤波器：相位滤波器....418

8.6.4 位置控制器：位置差接通 ..420
8.6.5 带比例积分控制器的位置闭环控制 ..421

8.7 数据表 ..423
8.7.1 机床数据 ..423
8.7.1.1 显示机床数据 ..423
8.7.1.2 NC 专用机床数据 ..423
8.7.1.3 通道专用机床数据 ..424
8.7.1.4 进给轴/主轴专用机床数据 ..424
8.7.2 信号 ..426
8.7.2.1 从 NC 发出的信号 ..426
8.7.2.2 发送至进给轴/主轴的信号 ..427

9 H2：输出到 PLC 的辅助功能 ..429

9.1 简要说明 ..429
9.1.1 功能 ..429
9.1.2 辅助功能的定义 ..430
9.1.3 辅助功能一览 ..431

9.2 预定义的辅助功能 ..437
9.2.1 一览：预定义的辅助功能 ..437
9.2.2 一览：输出特性 ..451
9.2.3 参数设置 ..456
9.2.3.1 分组指定 ..456
9.2.3.2 类型、地址扩展和数值 ..456
9.2.3.3 输出特性 ..458

9.3 用户自定义辅助功能 ..462
9.3.1 参数设置 ..464
9.3.1.1 用户自定义辅助功能的最大数量 ..464
9.3.1.2 分组指定 ..464
9.3.1.3 类型、地址扩展和数值 ..465
9.3.1.4 输出特性 ..466

9.4 关联辅助功能 ..467

9.5 类型专用输出特性 ..468

9.6 设置的输出特性的优先级 ..470

9.7 辅助功能编程 ..472

9.8 可编程的输出时长 ..473

9.9 输出至 PLC 的辅助功能 ..475

9.10 无程序段切换延迟的辅助功能 ..476

9.11 带隐性预处理停止的 M 功能 ..476

9.12 溢出转存时的特性 ..477

9.13 程序段搜索时的特性....478
9.13.1 程序段搜索类型 1、2 和 4 时的辅助功能输出....478
9.13.2 将一个辅助功能指定给多个组....480
9.13.3 生效的 M 辅助功能的时间戳....482
9.13.4 确定输出顺序....482
9.13.5 抑制主轴专用辅助功能的输出....484
9.13.6 使用程序段搜索类型 5（SERUPRO）时的辅助功能输出....487
9.13.7 SERUPRO 末尾 ASUB....492
9.14 隐性输出的辅助功能....499
9.15 获取信息....500
9.15.1 针对组的模态 M 辅助功能显示....500
9.15.2 查询系统变量....501
9.16 前提条件....504
9.16.1 一般前提条件....504
9.16.2 输出特性....505
9.17 示例....506
9.17.1 预定义辅助功能的扩展....506
9.17.2 定义辅助功能....508
9.18 数据表....512
9.18.1 机床数据....512
9.18.1.1 NC 专用机床数据....512
9.18.1.2 通道专用机床数据....513
9.18.2 信号....514
9.18.2.1 发送至通道的信号....514
9.18.2.2 来自通道的信号....514
9.18.2.3 发送至进给轴/主轴的信号....517
9.18.2.4 来自进给轴/主轴的信号....517
10 K1：BAG、通道、程序运行、复位特性....519
10.1 简要说明....519
10.2 运行方式组（BAG）....522
10.2.1 BAG 停止....524
10.2.2 BAG 复位....525
10.3 运行方式和运行方式切换....525
10.3.1 对各运行方式的监控和锁定....531
10.3.2 运行方式切换....531
10.4 通道....532
10.4.1 启动禁止，全局和通道专用....535
10.5 程序测试....537
10.5.1 不输出设定值的程序执行....537
10.5.2 程序单段运行....539

10.5.3 采用空运行进给执行程序 ..541
10.5.4 跳过零件程序段 ..543
10.6 工件仿真 ..544
10.7 程序段搜索类型 1、2 和 4 ..545
10.7.1 功能说明 ..546
10.7.2 程序段搜索与其它 NCK 功能的组合使用 ..549
10.7.2.1 程序段搜索后/中的 ASUB ..549
10.7.2.2 程序段搜索后的 PLC 动作 ..550
10.7.2.3 程序段搜索后的主轴功能 ..550
10.7.2.4 程序段搜索时读取系统变量 ..552
10.7.3 程序段搜索后自动启动 ASUB ..552
10.7.4 级联程序段搜索 ..554
10.7.5 带计算的程序段搜索的示例 ..555
10.7.6 前提条件 ..559
10.7.6.1 压缩器功能（COMPON、COMPCURV、COMPCAD） ..559
10.8 程序段搜索类型 5（SERUPRO） ..559
10.8.1 功能说明 ..559
10.8.2 重新定位至轮廓（REPOS） ..563
10.8.2.1 通过受控 REPOS 重定位至轮廓 ..572
10.8.3 搜索加速 ..573
10.8.4 SERUPRO ASUB ..575
10.8.5 自动执行的 SERUPRO ..578
10.8.6 禁用程序部分用于恢复执行 ..580
10.8.7 上电、运行方式切换和复位时的特性 ..583
10.8.8 前提条件 ..583
10.8.8.1 目标程序段中的 STOPRE ..583
10.8.8.2 目标程序段中的 SPOS ..584
10.8.8.3 运行到固定点（FXS） ..585
10.8.8.4 以限制的力矩/力运行（FOC） ..585
10.8.8.5 同步主轴 ..586
10.8.8.6 耦合和主从运行 ..586
10.8.8.7 轴功能 ..589
10.8.8.8 齿轮档切换 ..591
10.8.8.9 叠加运动 ..591
10.8.8.10 NC/PLC 接口信号 ..592
10.8.8.11 缺省设置的灵活设定 ..592
10.8.8.12 压缩器功能（COMPON、COMPCURV、COMPCAD） ..593
10.8.9 系统变量 ..593
10.9 程序运行 ..593
10.9.1 缺省设置 ..594
10.9.1.1 机床数据 ..595
10.9.1.2 编程 ..595
10.9.2 选择并启动 NC 程序 ..600

10.9.3 程序中断 601
10.9.4 通道复位 603
10.9.5 程序状态 604
10.9.6 通道状态 605
10.9.7 对操作和程序动作的响应 607
10.9.8 程序运行的时间图示例 608
10.9.9 程序跳转 609
10.9.9.1 跳回至程序开始处（GOTOS） 609
10.9.10 程序部分重复 611
10.9.10.1 编程 611
10.9.11 事件控制的程序调用(PROG_EVENT) 617
10.9.11.1 功能 617
10.9.11.2 参数设置 621
10.9.11.3 编程 625
10.9.11.4 前提条件 626
10.9.11.5 示例 626
10.9.12 通过停止延迟区控制停止事件 628
10.9.12.1 功能 628
10.9.12.2 参数设置 631
10.9.12.3 编程 632
10.9.12.4 边界条件 634
10.10 异步子程序（ASUB） 635
10.10.1 功能 635
10.10.1.1 程序运行中 ASUB 的执行过程 638
10.10.1.2 带 REPOSA 的 ASUB 638
10.10.1.3 NC 特性 640
10.10.2 调试：机床数据 641
10.10.2.1 NC 专用：BAG 专用 NC/PLC 接口信号和运行方式切换 641
10.10.2.2 NC 专用：ASUB 启动使能 642
10.10.2.3 NC 专用：参数设置的启动使能的生效范围 643
10.10.2.4 通道专用：在轴未回参考点的情况下仍使能启动 643
10.10.2.5 通道专用：在读取禁止的情况下仍使能启动 644
10.10.2.6 通道专用：在采用单程序段的情况下仍连续执行 644
10.10.2.7 通道专用：显示更新 645
10.10.3 编程：系统变量 645
10.10.3.1 REPOS 可能性（$P_REPINF） 645
10.10.3.2 激活事件（$AC_ASUP） 646
10.10.4 编程（SETINT、PRIO） 646
10.10.5 前提条件 648
10.10.6 示例 649
10.11 用于 RET 和 REPOS 的用户专用 ASUB 650
10.11.1 功能 650
10.11.2 参数设置 650
10.11.3 编程 652

10.12 在存在用户报警的情形下启动 ASUB....653
10.12.1 功能....653
10.12.2 激活....654
10.12.3 示例....655
10.12.3.1 通过复位触发的用户 ASUB - 示例 1....655
10.12.3.2 通过复位触发的用户 ASUB - 示例 2....656
10.12.3.3 写入 M0 的用户 ASUB....657
10.12.3.4 带停止的用户 ASUB....658
10.12.3.5 从停止状态触发的用户 ASUB....659

10.13 单程序段....661
10.13.1 参数设置....661
10.13.2 编程....663
10.13.2.1 关闭/激活单程序段执行（SBLOF，SBLON）....663
10.13.2.2 边界条件....664
10.13.2.3 示例....664
10.13.3 BAG 专用单程序段类型 A / B....667
10.13.4 前提条件....669
10.13.4.1 单程序段类型 SBL2 和逐段生效的同步动作....669
10.13.4.2 编程的停止（M0），单程序段和单程序段类型切换....669

10.14 程序控制....669
10.14.1 通过操作界面或 PLC 用户程序选择功能....669
10.14.2 激活跳转级....670
10.14.3 插补缓存的大小调整....672
10.14.4 启用附加基本程序段显示时的程序显示模式....673
10.14.5 ShopMill/ShopTurn 中的基本程序段显示....674
10.14.6 DIN 程序段的结构....677
10.14.7 外部执行....680
10.14.8 执行外部子程序（EXTCALL）....682

10.15 从外部存储器执行（选件）....685
10.15.1 功能....685
10.15.2 调试....689
10.15.2.1 驱动器的配置....689
10.15.2.2 全局零件程序存储器（GDIR）....690
10.15.2.3 EES 功能下零件程序中的文件处理设置....692
10.15.2.4 存储器配置....694
10.15.3 边界条件....694

10.16 Process DataShare - 数据输出到外部设备/文件上....694
10.16.1 功能....694
10.16.2 调试....696
10.16.3 编程....699
10.16.4 边界条件....703

10.17 启动、复位/零件程序结束和零件程序开始的系统设置....704

10.17.1 上电后通过定向转换回退刀具...710
10.18 由子程序替换功能...712
10.18.1 概述...712
10.18.2 替换 M、T/TCA 和 D/DL 功能...713
10.18.2.1 替换 M 功能...713
10.18.2.2 替换 T/TCA 和 D/DL 功能...716
10.18.2.3 系统变量...718
10.18.2.4 示例：M 功能的替换...720
10.18.2.5 示例：T 功能和 D 功能的替换...723
10.18.2.6 冲突情形下的特性...724
10.18.3 替换主轴功能...725
10.18.3.1 简介...725
10.18.3.2 M40 - M45（齿轮档切换）的替换...727
10.18.3.3 SPOS、SPOSA、M19（主轴定位）的替换...728
10.18.3.4 系统变量...729
10.18.3.5 示例：齿轮档切换...730
10.18.3.6 示例：主轴定位...732
10.18.4 子程序的属性...735
10.18.5 前提条件...737
10.19 重命名/禁用 NC 指令...737
10.20 程序运行时间/工件计数器...740
10.20.1 程序运行时间...740
10.20.1.1 功能...740
10.20.1.2 调试...744
10.20.1.3 前提条件...746
10.20.1.4 示例...746
10.20.2 工件计数器...748
10.20.2.1 功能...748
10.20.2.2 调试...749
10.20.2.3 前提条件...750
10.20.2.4 示例...751
10.21 数据表...752
10.21.1 功能...752
10.21.2 机床数据...754
10.21.2.1 通用机床数据...754
10.21.2.2 通道专用机床数据...755
10.21.2.3 进给轴/主轴专用机床数据...759
10.21.3 设定数据...759
10.21.3.1 通道专用设定数据...759
10.21.4 信号...760
10.21.4.1 发送至 NC 的信号...760
10.21.4.2 发送至 BAG 的信号...760
10.21.4.3 从 BAG 发出的信号...761

10.21.4.4 发送至通道的信号 ..761
10.21.4.5 从通道发出的信号 ..762
10.21.4.6 发送至进给轴/主轴的信号 ..764
10.21.4.7 来自进给轴/主轴的信号 ..764

11 K2：轴、坐标系、框架 ..765

11.1 简要说明 ..765
11.1.1 轴 ..765
11.1.2 坐标系 ..767
11.1.3 框架 ..769

11.2 轴 ..772
11.2.1 概述 ..772
11.2.2 机床轴 ..773
11.2.3 通道轴 ..774
11.2.4 几何轴 ..774
11.2.5 辅助轴 ..775
11.2.6 轨迹轴 ..775
11.2.7 定位轴 ..776
11.2.8 主处理轴 ..777
11.2.9 同步轴 ..778
11.2.10 轴配置 ..780
11.2.11 Link 轴 ..782

11.3 零点和参考点 ..783
11.3.1 工作区域中的参考点 ..783
11.3.2 坐标系和参考点的位置 ..785

11.4 坐标系 ..786
11.4.1 概述 ..786
11.4.2 机床坐标系（MCS） ..788
11.4.2.1 参考点状态的实际值设置和损失（PRESETON） ..790
11.4.2.2 参考点状态的实际值设置，无损失（PRESETONS） ..794
11.4.3 基本坐标系（BCS） ..800
11.4.4 附加补偿 ..802
11.4.4.1 外部零点偏移 ..802
11.4.4.2 DRF 偏移 ..803
11.4.4.3 叠加运动 ..804
11.4.4.4 复位特性 ..804
11.4.5 基本零点坐标系（BZS） ..806
11.4.6 可设定的零点坐标系（SZS） ..808
11.4.7 工件坐标系（WCS） ..810

11.5 框架 ..810
11.5.1 框架类型 ..810
11.5.2 框架分量 ..812
11.5.2.1 偏移 ..812

11.5.2.2 精偏...813
11.5.2.3 旋转：概述（只适用于几何轴）...814
11.5.2.4 通过欧拉角旋转：ZY'X" 转换（RPY 角）...815
11.5.2.5 通过欧拉角旋转：ZX'Z" 转换...821
11.5.2.6 在任意平面中旋转...822
11.5.2.7 比例缩放...824
11.5.2.8 镜像...824
11.5.2.9 级联运算符...825
11.5.2.10 可编程的轴名称...825
11.5.2.11 坐标转换...826
11.5.3 数据管理框架和生效框架...827
11.5.3.1 概述...827
11.5.3.2 激活数据管理框架...829
11.5.3.3 NCU 全局和通道专用变量...831
11.5.4 框架链和坐标系...831
11.5.4.1 概述...831
11.5.4.2 相对坐标系...833
11.5.4.3 可选 SZS...834
11.5.4.4 选择在 WCS 或 SZS 中手动运行几何轴 ($AC_JOG_COORD)...835
11.5.4.5 抑制框架...836
11.5.5 框架链的框架...837
11.5.5.1 概述...837
11.5.5.2 可设定框架 $P_UIFR[<n>]...838
11.5.5.3 磨削框架 $P_GFR[<n>]...841
11.5.5.4 通道专用基本框架[<n>]...845
11.5.5.5 NCU 全局基本框架 $P_NCBFR[<n>]...846
11.5.5.6 生效的总基本框架 $P_ACTBFRAME...847
11.5.5.7 可编程框架 $P_PFRAME...849
11.5.5.8 通道专用系统框架...851
11.5.6 隐性框架修改...854
11.5.6.1 切换几何轴...854
11.5.6.2 选择/取消坐标转换：常规...857
11.5.6.3 选择/取消坐标转换：TRANSMIT...858
11.5.6.4 选择/取消坐标转换：TRACYL...865
11.5.6.5 选择/取消坐标转换：TRAANG...870
11.5.6.6 调整生效框架...876
11.5.6.7 映射框架...877
11.5.7 预定义框架功能...881
11.5.7.1 反转框架...881
11.5.7.2 框架链中的叠加框架...883
11.5.8 功能...885
11.5.8.1 设置零点、工件测量和刀具测量...885
11.5.8.2 轴外部零点偏移...885
11.5.8.3 刀架...886
11.5.9 带 SAVE 属性的子程序...896

11.5.10 数据备份 ..897
11.5.11 坐标系中的位置 ..898
11.5.12 控制系统特性 ..898
11.5.12.1 上电 ..898
11.5.12.2 运行方式切换 ..899
11.5.12.3 通道复位/零件程序结束 ..900
11.5.12.4 零件程序开始 ..904
11.5.12.5 程序段搜索 ..904
11.5.12.6 REPOS ..905
11.6 工件相关的实际值系统 ..905
11.6.1 概述 ..905
11.6.2 使用工件相关实际值系统 ..906
11.6.3 特殊响应 ..908
11.7 前提条件 ..910
11.8 示例 ..910
11.8.1 轴 ..910
11.8.2 坐标系 ..914
11.8.3 框架 ..916
11.9 数据表 ..918
11.9.1 机床数据 ..918
11.9.1.1 显示机床数据 ..918
11.9.1.2 NC 专用机床数据 ..919
11.9.1.3 通道专用机床数据 ..919
11.9.1.4 进给轴/主轴专用机床数据 ..921
11.9.2 设定数据 ..921
11.9.2.1 通道专用设定数据 ..921
11.9.3 系统变量 ..921
11.9.4 信号 ..923
11.9.4.1 来自通道的信号 ..923
11.9.4.2 发送至进给轴/主轴的信号 ..923
11.9.4.3 来自进给轴/主轴的信号 ..923
12 N2：急停 ..925
12.1 简要说明 ..925
12.2 标准 ..925
12.3 急停控制元件 ..926
12.4 急停过程 ..927
12.5 急停应答 ..928
12.6 数据表 ..930
12.6.1 机床数据 ..930
12.6.1.1 进给轴/主轴专用机床数据 ..930

12.6.2 信号....931
12.6.2.1 发送至 NC 的信号....931
12.6.2.2 从 NC 发出的信号....931
12.6.2.3 发送至 BAG 的信号....931
13 P1：端面轴....933
13.1 功能....933
13.2 参数设置....936
13.3 编程....940
13.4 前提条件....941
13.5 示例....942
13.6 数据表....943
13.6.1 机床数据....943
13.6.1.1 通道专用机床数据....943
13.6.1.2 进给轴/主轴专用机床数据....943
14 P3：SINUMERIK 840D sl 的 PLC 基本程序....945
14.1 简要说明....945
14.2 PLC-CPU 的基础数据....947
14.3 PLC 操作系统版本....948
14.4 PLC 运行方式开关....949
14.5 预留资源（计时器、计数器、FC、FB、DB、I/O）....949
14.6 调试 PLC-CPU 的硬件配置....950
14.7 调试 PLC 程序....950
14.7.1 安装基本程序....950
14.7.2 基本程序的使用....951
14.7.3 版本标识....952
14.7.4 机床程序....953
14.7.5 数据备份....953
14.7.6 PLC 批量调试，PLC 存档....954
14.7.7 软件升级....956
14.7.8 I/O 模块（FM 模块、CP 模块）....957
14.7.9 排故....958
14.8 PLC-CPU 的连接....959
14.8.1 简介....959
14.8.2 PLC-CPU 的属性....959
14.8.3 集成 PLC 的接口....959
14.8.4 PLC 的诊断缓存....961
14.9 接口结构....962

14.9.1 PLC/NCK 接口 ..962
14.9.2 PLC/HMI 接口 ..969
14.9.3 PLC/MCP/HHU 接口 ..973

14.10 基本程序的结构和功能 ..975
14.10.1 NCK-PLC 启动和同步 ..977
14.10.2 循环运行（OB1） ..977
14.10.3 时间报警的处理（OB35） ..980
14.10.4 过程报警处理 (OB40) ..980
14.10.5 诊断报警，模块故障处理(OB82, OB86) ..980
14.10.6 NCK 故障时的特性 ..981
14.10.7 由用户程序调用的基本程序功能 ..982
14.10.8 含接口 DB 的用户程序的符号编程 ..985
14.10.9 基于列表的 M 解码 ..987
14.10.10 PLC 机床数据 ..991
14.10.11 机床控制面板、手持操作设备、直接控制按键的配置 ..995
14.10.12 机床控制面板、手持操作设备的切换 ..1006

14.11 用于 Safety Integrated 的 SPL ..1007

14.12 分配一览 ..1008
14.12.1 分配：NCK/PLC 接口 ..1008
14.12.2 分配：FB/FC ..1008
14.12.3 分配：DB ..1008
14.12.4 分配：计时器 ..1010

14.13 用于 HMI 的 PLC 功能 (DB19) ..1010
14.13.1 通道选择 ..1010
14.13.2 程序选择 ..1012
14.13.3 激活“按键禁用” ..1015
14.13.4 操作区域编号 ..1015
14.13.5 屏幕窗口号 ..1016
14.13.5.1 屏幕窗口号：JOG、手动加工 ..1017
14.13.5.2 屏幕窗口号：回参考点 ..1023
14.13.5.3 屏幕窗口号：MDI ..1024
14.13.5.4 屏幕窗口号：AUTO ..1024
14.13.5.5 屏幕窗口号：“参数”操作区 ..1025
14.13.5.6 屏幕窗口号：“程序”操作区 ..1027
14.13.5.7 屏幕窗口号：“程序管理器”操作区 ..1027
14.13.5.8 屏幕窗口号：“诊断”操作区 ..1028
14.13.6 HMI 监示器 ..1028

14.14 用于集成 PROFIBUS 上的驱动组件的 PLC 功能 ..1029
14.14.1 一览 ..1029
14.14.2 调试 ..1030
14.14.3 示例 ..1031

14.15 PLC 基本程序的存储空间需求 ..1032

14.16 前提条件和 NC-VAR-Selector ... 1036
14.16.1 前提条件 ... 1036
14.16.1.1 编程工具和参数设置工具 ... 1036
14.16.1.2 必需的 SIMATIC 文档 ... 1038
14.16.1.3 相关的 SINUMERIK 文档 ... 1038
14.16.2 NC-VAR-Selector ... 1039
14.16.2.1 概述 ... 1039
14.16.2.2 功能说明 ... 1042
14.16.2.3 调试、安装 ... 1051

14.17 模块说明 ... 1051
14.17.1 FB1：RUN_UP - 基本程序、启动部分 ... 1051
14.17.2 FB2：GET - 读取 NC 变量 ... 1061
14.17.3 FB3：PUT - 写入 NC 变量 ... 1072
14.17.4 FB4:PI_SERV - 请求 PI 服务 ... 1080
14.17.4.1 可供使用的 PI 服务列表 ... 1085
14.17.4.2 PI 服务：ASUB ... 1086
14.17.4.3 PI 服务：CANCEL ... 1087
14.17.4.4 PI 服务：CONFIG ... 1088
14.17.4.5 PI 服务：DIGION ... 1088
14.17.4.6 PI 服务：DIGIOF ... 1088
14.17.4.7 PI 服务：FINDBL ... 1089
14.17.4.8 PI 服务：LOGIN ... 1089
14.17.4.9 PI 服务：LOGOUT ... 1090
14.17.4.10 PI 服务：NCRES ... 1090
14.17.4.11 PI 服务：SELECT ... 1090
14.17.4.12 PI 服务：SETUDT ... 1091
14.17.4.13 PI 服务：SETUFR ... 1092
14.17.4.14 PI 服务：RETRAC ... 1092
14.17.4.15 PI 服务：CRCEDN ... 1093
14.17.4.16 PI 服务：CREACE ... 1094
14.17.4.17 PI 服务：CREATO ... 1094
14.17.4.18 PI 服务：DELECE ... 1094
14.17.4.19 PI 服务：DELETO ... 1095
14.17.4.20 PI 服务：MMCSEM ... 1095
14.17.4.21 PI 服务：TMCRTO ... 1097
14.17.4.22 PI 服务：TMFDPL ... 1098
14.17.4.23 PI 服务：TMFPBP ... 1099
14.17.4.24 PI 服务：TMGETT ... 1100
14.17.4.25 PI 服务：TMMVTL ... 1101
14.17.4.26 PI 服务：TMPOSM ... 1103
14.17.4.27 PI 服务：TMPCIT ... 1104
14.17.4.28 PI 服务：TMRASS ... 1104
14.17.4.29 PI 服务：TRESMO ... 1105
14.17.4.30 PI 服务：TSEARC ... 1105

14.17.4.31 PI 服务：TMCRMT 1109
14.17.4.32 PI 服务：TMDLMT 1110
14.17.4.33 PI 服务：POSMT 1110
14.17.4.34 PI 服务：FDPLMT 1111
14.17.5 FB5：GETGUD - 读取 GUD 变量 1113
14.17.6 FB7：PI_SERV2（请求 PI 服务） 1122
14.17.7 FB9: MzuN - 操作单元切换 1123
14.17.8 FB10：安全继电器（SI-Relais） 1128
14.17.9 FB11:制动测试 1131
14.17.10 FB29:诊断信号记录器和数据触发器 1136
14.17.11 FC2：GP_HP - 基本程序，循环部分 1141
14.17.12 FC3：GP_PRAL - 基本程序，报警控制部分 1142
14.17.13 FC5：GP_DIAG 基本程序，诊断报警和模块故障 1145
14.17.14 FC6：TM_TRANS2 刀具管理和多刀的传输模块 1147
14.17.15 FC7：TM_REV - 刀塔换刀的传输模块 1148
14.17.16 FC8：TM_TRANS - 刀具管理的传输模块 1152
14.17.17 FC9：ASUB - 启动异步子程序 1160
14.17.18 FC10：AL_MSG - 故障消息和运行消息 1163
14.17.19 FC12：AUXFU - 辅助功能的用户调用接口 1165
14.17.20 FC13：BHGDisp - 手持操作设备的显示控制 1166
14.17.21 FC17：YDelta - 星形/三角形切换 1170
14.17.22 FC18: SpinCtrl - 主轴控制 1175
14.17.23 FC19:MCP_IFM - 将 MCP 信号传送到接口上 1186
14.17.24 FC21:传输 - NC/PLC 数据交换 1194
14.17.24.1 功能 1194
14.17.24.2 功能说明 1194
14.17.24.3 对形式参数的说明 1195
14.17.24.4 功能 1、2：发送至/来自通道的同步动作信号 1195
14.17.24.5 功能 3、4：PLC-NC 快速数据交换 1197
14.17.24.6 功能 5：更新发送至通道的控制信号 1201
14.17.24.7 功能 6：更新发送至轴的控制信号 1202
14.17.24.8 功能 7：更新发送至轴的控制信号 1203
14.17.25 FC22: TM_DIR - 刀具管理的方向选择 1203
14.17.26 FC24: MCP_IFM2 - 将 MCP 信号传送到接口上 1206
14.17.27 FC25: MCP_IFT - 将 MCP/BT 信号传送到接口上 1209
14.17.28 FC26: HPU_MCP - 将 HT 8 信号传送到接口上 1212
14.17.28.1 HT 8 发出的 NC/PLC 接口信号一览 1218
14.17.28.2 发送至 HT 8 的 NC/PLC 接口信号一览 1219
14.17.29 FC19, FC24, FC25, FC26 - 源代码说明 1220
14.17.30 FC1005：AG_SEND - 将数据传输至以太网 CP 1222
14.17.31 FC1006：AG_RECV - 从以太网 CP 接收数据 1224
14.18 信号/数据说明 1225
14.18.1 接口信号 NCK/PLC、HMI/PLC、MCP/PLC 1225
14.18.2 解码 M 信号 1225

14.18.3 G 指令 1227
14.18.4 DB2 中的消息信号 1228

14.19 STEP 7 中编程的说明 1228
14.19.1 复制数据 1229
14.19.2 ANY 和 POINTER 1229
14.19.2.1 在 FC 中使用 POINTER 和 ANY 1229
14.19.2.2 在 FB 中使用 POINTER 和 ANY 1231
14.19.2.3 用于传输至 FC 或 FB 的 POINTER 或 ANY 变量 1232
14.19.3 多实例 DB 1235
14.19.4 字符串 1236
14.19.5 确定数据块结构中的偏移地址 1237
14.19.6 FB 调用 1237

14.20 数据表 1239
14.20.1 机床数据 1239
14.20.1.1 显示机床数据 1239
14.20.1.2 NC 专用机床数据 1239
14.20.1.3 通道专用机床数据 1240
14.20.2 信号 1240
14.20.2.1 来自操作面板的信号 1240

15 P4：SINUMERIK 828D 的 PLC 1241

15.1 概述 1241
15.1.1 PLC 固件 1241
15.1.2 PLC 用户接口 1241
15.1.2.1 循环交换的数据 1243
15.1.2.2 报警和消息 1243
15.1.2.3 掉电保持数据 1244
15.1.2.4 非掉电保持数据 1244
15.1.2.5 PLC 机床数据 1244
15.1.3 PLC 基础数据 1244
15.1.4 PLC I/O、快速板载输入/输出 1245
15.1.5 PLC Toolbox 1245
15.1.5.1 星形/三角形切换 1245

15.2 快速板载输入/输出 1246

15.3 梯形图浏览器和梯形图插件 1248
15.3.1 一览 1248
15.3.2 参数设置 1248

15.4 PLC 编程工具 1249

15.5 数据接口 1250
15.5.1 PLC-NC 接口 1250
15.5.1.1 运行方式信号 1250
15.5.1.2 NC 通道信号 1252

15.5.1.3 轴信号和主轴信号 1253
15.5.1.4 通用 NC 信号 1254
15.5.1.5 PLC-NC 快速数据交换 1254
15.5.2 PLC-HMI 接口 1255
15.5.2.1 操作区域编号 1255
15.5.2.2 屏幕窗口号 1256
15.6 功能接口 1268
15.6.1 读写 NC 变量 1268
15.6.1.1 用户接口 1268
15.6.1.2 任务设定 1268
15.6.1.3 任务管理器：启动任务 1270
15.6.1.4 任务管理器：等待任务结束 1270
15.6.1.5 任务管理：任务结束 1271
15.6.1.6 任务管理：流程图 1272
15.6.1.7 任务分析 1273
15.6.1.8 可操作变量 1275
15.6.1.9 设定所选 NC 变量 1285
15.6.2 程序实例服务（PI 服务） 1286
15.6.2.1 任务设定 1286
15.6.2.2 任务反馈信息 1287
15.6.2.3 PI 服务 ASUB 1288
15.6.2.4 PI 服务 LOGOUT 1290
15.6.2.5 PI 服务 DATA_SAVE 1291
15.6.2.6 PI 服务 TMMVTL 1291
15.6.2.7 PI 服务：周期图 1292
15.6.3 PLC 用户报警 1293
15.6.3.1 用户接口 1293
15.6.3.2 用户报警的激活接口 1293
15.6.3.3 用户报警的变量接口 1295
15.6.3.4 定义用户报警 1296
15.6.3.5 读取生效的报警响应和删除标准 1299
15.6.3.6 用户报警的应答接口 1299
15.6.3.7 HMI 接口 1299
15.6.4 PLC 轴控制 1300
15.6.4.1 简介 1300
15.6.4.2 用户接口：将 NC 轴作为 PLC 轴 1301
15.6.4.3 用户接口：功能 1302
15.6.4.4 主轴定位 1304
15.6.4.5 主轴旋转 1306
15.6.4.6 主轴往复 1309
15.6.4.7 分度轴 1312
15.6.4.8 定位轴（公制） 1314
15.6.4.9 定位轴（英制） 1316
15.6.4.10 带手轮叠加的公制定位轴 1318

15.6.4.11 带手轮叠加的英制定位轴 1320
15.6.4.12 主轴旋转与自动齿轮档选择 1321
15.6.4.13 通过恒定切削速度 [m/min] 旋转主轴 1323
15.6.4.14 通过恒定切削速度 [feet/min] 旋转主轴 1325
15.6.4.15 故障消息 1327
15.6.5 启动 ASUB 1329
15.6.5.1 任务启动 1329
15.6.5.2 任务结果 1330
15.6.5.3 信号流 1331
15.6.6 在 HMI 上执行通道选择 1331

15.7 数控禁用功能（选件） 1333
15.7.1 功能 1333
15.7.2 前提条件 1334
15.7.3 限制 1334
15.7.4 防止篡改 1335
15.7.5 数控禁用功能的初次调试 1335
15.7.6 延长数控禁用功能 1338
15.7.7 取消数控禁用功能 1340
15.7.8 更换损坏的控制系统硬件 (PPU) 1342
15.7.9 更换损坏的 CF 卡 1344
15.7.10 忘记 OEM-PIN 1346
15.7.11 其它信息 1347

16 R1：回参考点 1349

16.1 简要说明 1349

16.2 轴专用回参考点 1350

16.3 通道专用回参考点 1352

16.4 在零件程序中写入回参考点（G74） 1354

16.5 使用增量测量系统回参考点 1355
16.5.1 硬件信号 1355
16.5.2 零脉冲选择 1356
16.5.3 时序 1358
16.5.4 阶段 1：运行到参考凸轮 1359
16.5.5 阶段 2：与零脉冲同步 1362
16.5.6 阶段 3：运行至参考点 1367

16.6 通过距离编码的参考脉冲回参考点 1369
16.6.1 概述 1369
16.6.2 基础参数设置 1370
16.6.3 时序 1372
16.6.4 阶段 1：同步越过参考脉冲 1372
16.6.5 阶段 2：运行到目标点 1374

16.7 通过实际值补偿回参考点 1376

16.7.1 对回参考点测量系统进行实际值补偿 1376
16.7.2 使用距离编码参考脉冲的测量系统中的实际值补偿 1377
16.8 跟踪模式下回参考点 1378
16.9 使用绝对值编码器回参考点 1381
16.9.1 校准信息 1381
16.9.2 通过输入参考点偏移校准 1382
16.9.3 通过输入参考点值校准 1383
16.9.4 通过测头自动校准 1385
16.9.5 通过 BERO 校准 1386
16.9.6 使用绝对值编码器回参考点 1387
16.9.7 通过旋转绝对值编码器启用替代零脉冲回参考点 1388
16.9.8 激活测量系统 1390
16.9.9 不支持的回参考点方式 1391
16.10 自动恢复机床参考 1392
16.10.1 自动回参考点 1393
16.10.2 恢复实际位置 1394
16.11 前提条件 1396
16.11.1 较大运行范围 1396
16.12 数据表 1397
16.12.1 机床数据 1397
16.12.1.1 NC 专用机床数据 1397
16.12.1.2 通道专用机床数据 1397
16.12.1.3 进给轴/主轴专用机床数据 1398
16.12.2 信号 1399
16.12.2.1 发送至 BAG 的信号 1399
16.12.2.2 从 BAG 发出的信号 1399
16.12.2.3 发送至通道的信号 1400
16.12.2.4 来自通道的信号 1400
16.12.2.5 发送至进给轴/主轴的信号 1400
16.12.2.6 来自进给轴/主轴的信号 1400
17 S1：主轴 1401
17.1 简要说明 1401
17.2 运行方式 1402
17.2.1 概述 1402
17.2.2 运行方式切换 1403
17.2.3 控制模式 1404
17.2.4 往复模式 1407
17.2.5 定位模式 1408
17.2.5.1 一般功能 1408
17.2.5.2 从旋转状态定位主轴 1415
17.2.5.3 从静止状态定位主轴 1420

17.2.5.4 换刀信号“主轴到达位置” 1423
17.2.6 进给轴模式 1425
17.2.6.1 一般功能 1425
17.2.6.2 隐性过渡至进给轴模式 1428
17.2.7 主轴初始设置 1431
17.2.8 刚性攻丝 1432
17.2.8.1 功能 1432
17.2.8.2 编程 1432
17.2.8.3 示例：使用 G331 / G332 攻丝 1434
17.2.8.4 示例：给出当前齿轮级已编程的钻削转速 1434
17.2.8.5 示例：使用第二齿轮级数据组 1435
17.2.8.6 示例：未进行转速编程，齿轮级监控 1435
17.2.8.7 示例：无法进行齿轮换档，齿轮级监控 1435
17.2.8.8 示例：不带 SPOS 的编程 1436
17.2.8.9 特殊情形：在 NC 程序中通过 NC/PLC 接口信号将旋转方向反转 1436
17.2.9 带弹性卡头的攻丝 1438
17.2.9.1 功能 1438
17.2.9.2 编程 1438
17.3 回参考点/同步 1439
17.4 可配置的齿轮档调整 1444
17.4.1 主轴的齿轮档和齿轮档切换 1444
17.4.2 主轴齿轮档 0 1455
17.4.3 确定主轴齿轮档 1457
17.4.4 齿轮档切换时的参数组选择 1458
17.4.5 中间齿轮比 1461
17.4.6 未应答的齿轮档切换 1462
17.4.7 通过往复模式进行齿轮档切换 1463
17.4.8 在固定位置切换齿轮档 1470
17.4.9 执行 M70 时启用的齿轮档 1477
17.4.10 在 DryRun、程序测试和 SERUPRO 下抑制齿轮档切换 1479
17.5 其他主轴功能调整设置 1481
17.6 可切换主轴 1483
17.7 编程 1486
17.7.1 在零件程序中编程 1486
17.7.2 通过同步动作编程 1491
17.7.3 通过 PLC 使用 FC18 编写主轴控制 - 仅针对 840D sl 1491
17.7.4 通过 NC/PLC 接口信号编程 1492
17.7.4.1 功能 1492
17.7.4.2 调试：机床数据 1493
17.7.4.3 调试：NC/PLC 接口信号 1494
17.7.4.4 转速设定（SD43200） 1495
17.7.4.5 恒定切削速度的设定（SD43202） 1495
17.7.4.6 针对主主轴的主轴转速类型设定（SD43206） 1496

17.7.5 外部编程（PLC、HMI） 1497

17.8 主轴监控 1498
17.8.1 允许的转速范围 1498
17.8.2 进给轴/主轴停止 1499
17.8.3 主轴位于设定区域内 1499
17.8.4 齿轮档的最小/最大转速 1501
17.8.5 主轴转速限制诊断 1502
17.8.6 主轴转速上限 1504
17.8.7 最大编码器极限频率 1505
17.8.8 目标点监控 1507
17.8.9 M40：转速超出配置的开关阈值时的自动齿轮档选择 1509

17.9 配备 SMI 24 的主轴（Weiss 主轴） 1510
17.9.1 简介 1510
17.9.2 传感器数据 1511
17.9.3 夹紧状态 1513
17.9.4 其他驱动参数 1515

17.10 前提条件 1516
17.10.1 闭环控制参数的修改 1516

17.11 示例 1516
17.11.1 自动齿轮档选择（M40） 1516

17.12 数据表 1517
17.12.1 机床数据 1517
17.12.1.1 NC 专用机床数据 1517
17.12.1.2 通道专用机床数据 1518
17.12.1.3 进给轴/主轴专用机床数据 1518
17.12.2 设定数据 1520
17.12.2.1 通道专用设定数据 1520
17.12.2.2 进给轴/主轴专用设定数据 1521
17.12.3 信号 1521
17.12.3.1 发送至进给轴/主轴的信号 1521
17.12.3.2 来自进给轴/主轴的信号 1522

18 V1：进给率 1525

18.1 简要说明 1525

18.2 轨迹进给率 F 1526
18.2.1 进给类型 G93、G94、G95 1528
18.2.2 进给类型 G96、G961、G962、G97、G971 1532
18.2.3 螺纹切削（G33, G34, G35, G335, G336）中的进给率 1536
18.2.3.1 G33 中的进给率 1536
18.2.3.2 G34 和 G35 中线性递增式/递减式的螺距变化 1537
18.2.3.3 G33、G34 和 G35 中轴的加速度特性 1540
18.2.3.4 G33、G34、G35 中可编程的导入和导出行程（DITS、DITE） 1541

18.2.3.5 螺纹切削期间快速返回 1543
18.2.3.6 球螺纹（G335，G336） 1548
18.2.4 刚性攻丝（G331、G332）中的进给率 1553
18.2.5 带补偿夹具攻丝（G63）中的进给率 1555
18.2.6 FGROUP 和 FGREF 1556
18.3 定位轴的进给率（FA） 1558
18.4 进给控制 1560
18.4.1 进给禁止和进给/主轴停止 1560
18.4.2 通过机床控制面板设置进给倍率 1561
18.4.3 可编程的进给倍率 1566
18.4.4 空运行进给率 1567
18.4.5 一个程序段中的多个进给值 1569
18.4.6 固定进给值 1576
18.4.7 可编程的进给特性曲线 1577
18.4.8 倒角/倒圆进给率 FRC、FRCM 1578
18.4.9 逐段进给率 FB 1581
18.4.10 单轴动态响应的控制 1582
18.5 前提条件 1587
18.6 数据表 1588
18.6.1 机床数据 1588
18.6.1.1 NC 专用机床数据 1588
18.6.1.2 通道专用机床数据 1589
18.6.1.3 进给轴/主轴专用机床数据 1589
18.6.2 设定数据 1590
18.6.2.1 通道专用设定数据 1590
18.6.2.2 进给轴/主轴专用设定数据 1590
18.6.3 信号 1591
18.6.3.1 发送至通道的信号 1591
18.6.3.2 来自通道的信号 1591
18.6.3.3 发送至进给轴/主轴的信号 1591
18.6.3.4 来自进给轴/主轴的信号 1592
19 W1：刀具补偿 1593
19.1 简要说明 1593
19.2 刀具 1596
19.2.1 简介 1596
19.2.2 补偿存储器结构 1598
19.2.3 刀具补偿计算 1600
19.2.4 用于 NC 地址 T 和 M 的地址扩展 1600
19.2.5 任意 D 号设定 1602
19.2.6 换刀中出现故障时的补偿程序段 1609
19.2.7 定义刀具参数的作用 1611

19.3 平面 D 号结构 ... 1612
19.3.1 简介 ... 1612
19.3.2 新建一个 D 号（补偿程序段） ... 1613
19.3.3 D 号编程 ... 1614
19.3.4 T 号编程 ... 1616
19.3.5 M6 编程 ... 1616
19.3.6 程序测试 ... 1617
19.3.7 刀具管理或者“平面 D 号结构” ... 1618

19.4 刀具刀沿 ... 1619
19.4.1 简介 ... 1619
19.4.2 刀具参数 1：刀具类型 ... 1621
19.4.3 刀具参数 2：刀沿位置 ... 1624
19.4.4 刀具参数 3 - 5： 几何数据 - 刀具长度 ... 1626
19.4.5 刀具参数 6 - 11： 几何尺寸 - 刀具形状 ... 1628
19.4.6 刀具参数 12 - 14： 磨损尺寸 - 刀具长度 ... 1629
19.4.7 刀具参数 15 - 20： 磨损尺寸 - 刀具形状 ... 1630
19.4.8 刀具参数 21 - 23： 基本尺寸 / 适配器尺寸 ... 1630
19.4.9 刀具参数 24：后角 ... 1632
19.4.10 带相关刀沿位置的刀具 ... 1634

19.5 2D 刀具半径补偿（2D-WRK） ... 1634
19.5.1 简介 ... 1634
19.5.2 选择刀具半径补偿 （G41/G42） ... 1636
19.5.3 逼近和回退特性 （NORM/KONT/KONTC/KONTT） ... 1637
19.5.4 平滑逼近和退回 ... 1642
19.5.4.1 功能 ... 1642
19.5.4.2 参数 ... 1642
19.5.4.3 速度 ... 1649
19.5.4.4 系统变量 ... 1651
19.5.4.5 前提条件 ... 1652
19.5.4.6 示例 ... 1653
19.5.5 取消刀具半径补偿 (G40) ... 1656
19.5.6 外角的补偿 ... 1656
19.5.7 内角的补偿 ... 1661
19.5.8 防撞监控和瓶颈识别 ... 1663
19.5.9 槽形识别（选件）- 仅 840D sl ... 1665
19.5.10 使用可变补偿值的程序段 ... 1667
19.5.11 报警特性 ... 1669
19.5.12 用于多项式的交点程序 ... 1670
19.5.13 G461/G462: 逼近/回退方案扩展 ... 1670

19.6 刀具半径补偿保持恒定 ... 1674

19.7 可定向刀架 ... 1678
19.7.1 简介 ... 1678
19.7.2 运动关系与机床结构 ... 1685

19.7.3 使用 3 + 2 轴进行斜面加工 1694
19.7.4 使用可旋转工作台的车床 1695
19.7.5 使用可定向刀架的步骤 1698
19.7.6 编程 1702
19.7.7 用于定向的前提条件和控制器特性 1703
19.7.8 示例 1706
19.7.8.1 示例：可定向刀架 1706
19.7.8.2 使用可旋转平台的可定向刀架示例 1706
19.7.8.3 计算使用位置和工件专用的补偿 1709

19.8 可旋转刀具上的刀沿数据修正 1712
19.8.1 功能 1712
19.8.2 计算旋转角度 1712
19.8.3 使用车刀时的刀沿位置、切削方向和角度 1713
19.8.4 车刀旋转时的修正 1715
19.8.5 使用铣刀和钻头时的刀沿位置 1717
19.8.6 铣刀和钻头旋转时的修正 1718
19.8.7 参数设置 1719
19.8.8 编程 1720
19.8.9 示例 1724

19.9 增量编程补偿值 1725
19.9.1 G91 扩展 1725
19.9.2 朝刀具定向运行（MOVT） 1727

19.10 刀具基本定向 1728
19.10.1 功能 1728
19.10.2 参数设置 1731
19.10.3 示例 1733

19.11 刀具补偿的特殊处理 1733
19.11.1 相关的设定数据 1733
19.11.2 刀具长度镜像 （SD42900 $SC_MIRROR_TOOL_LENGTH） 1734
19.11.3 磨损长度镜像 （SD42920 $SC_WEAR_SIGN_CUTPOS） 1735
19.11.4 刀具长度和平面转换 （SD42940 $SC_TOOL_LENGTH_CONST） 1737
19.11.5 刀具类型 (SD42950 $SC_TOOL_LENGTH_TYPE) 1738
19.11.6 考虑方向时工件坐标系中的刀具长度 1739
19.11.7 刀具方向上的刀具长度偏移 1740

19.12 总补偿与设置补偿 1744
19.12.1 简介 1744
19.12.2 功能性说明 1745
19.12.3 激活 1749
19.12.4 示例 1755
19.12.5 刀具长度测定的扩展 1756
19.12.5.1 计算使用位置和工件专用的补偿 1756
19.12.5.2 单个磨损值的功能性 1761

19.13 使用刀具环境工作 1765
19.13.1 保存刀具环境 (TOOLENV) 1765
19.13.2 删除刀具环境 (DELTOOLENV) 1768
19.13.3 读取 T 号、D 号和 DL 号 (GETTENV) 1769
19.13.4 读取保存的刀具环境的信息 ($P_TOOLENVN, ($P_TOOLENV) 1770
19.13.5 读取刀具长度或刀具长度分量 (GETTCOR) 1771
19.13.6 修改刀具分量 (SETTCOR) 1778
19.14 读取刀具长度 L1、 L2、 L3 对坐标轴的指定关系 (LENTOAX) 1792
19.15 前提条件 1795
19.15.1 平面 D 号结构 1795
19.15.2 SD42935 扩展 1795
19.15.3 对刀 1796
19.16 数据表 1796
19.16.1 机床数据 1796
19.16.1.1 NC 专用机床数据 1796
19.16.1.2 通道专用机床数据 1797
19.16.1.3 进给轴/主轴专用机床数据 1798
19.16.2 设定数据 1798
19.16.2.1 通道专用设定数据 1798
19.16.3 信号 1799
19.16.3.1 来自通道的信号 1799
20 Z1:NC/PLC 接口信号 1801
A 附录 1803
A.1 缩略语列表 1803
A.2 手册一览 1814
词汇表 1815
索引 1837

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可引发生命危险

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 工业安全

说明

工业安全

西门子提供了含工业安全功能的产品和解决方案，以支持设备、系统、机器和电网的安全运行。

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)。

⚠ 警告

篡改软件会引起不安全的驱动状态从而导致生命危险

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。

A2: 不同的 NC/PLC 接口信号与功能 2

2.1 简要说明

内容

PLC/NC 接口由数据接口和功能接口构成。数据接口包含有状态信号、控制信号、辅助指令和 G 指令等，PLC 至 NC 的任务传输则通过功能接口实现。

本说明介绍了一些常用的接口信号的功能，在特殊功能的说明中不会再介绍这些常用信号：

- 异步事件
- 状态信号
- PLC 变量（读取和写入）

2.2 NC/PLC 接口信号

2.2.1 简介

NC/PLC 接口

NC/PLC 接口由以下部分组成：

- 数据接口
- 功能接口

数据接口

数据接口用于协调各个组件：

- PLC 用户程序
- NC
- HMI（操作组件）
- MCP（机床控制面板）

数据交换是由 PLC 基本程序组织的。

循环信号交换

以下接口信号是由 PLC 基本程序循环传输的，即以 OB1 为周期进行传输：

- NC 和操作面板前端专用信号
- 运行方式组专用信号
- 通道专用信号
- 进给轴/主轴专用信号

NC 和操作面板前端专用信号 (DB10)

PLC 至 NC：

- CNC 输入和输出的控制信号
- 钥匙开关信号（及口令）

NC 至 PLC：

- CNC 输入端的实际值
- CNC 输出端的设定值
- 来自 NC 和 HMI 的“Ready”就绪信号
- NC 状态信号（报警信号）

通道专用信号 (DB21, ...)

PLC 至 NC：

- 控制信号 “删除剩余行程”

NC 至 PLC：

- NC 状态信号（存在 NC 报警）

进给轴/主轴专用信号 (DB31, ...)

PLC 至 NC：

- 发送至进给轴/主轴的控制信号（例如：跟踪运行、伺服使能 ...）
- 发送至驱动的控制信号（字节 20、21）

NC 至 PLC：

- 来自进给轴/主轴的状态信号（例如：位置控制器有效、电流控制器有效 ...）
- 来自驱动的状态信号（字节 93、94）

功能接口

功能接口是由功能块 (FB) 和功能调用 (FC) 构成的。功能请求（比如：运行轴）是通过功能接口从 PLC 传输到 NC 的。

文档

- 对于 PLC 基本程序的说明：
  → 功能手册之基本功能分册；章节“PLC 基本程序（P3）”
- 对于事件控制的信号交换的说明（辅助功能和 G 指令）：
  → 功能手册之基本功能分册：“输出给 PLC 的辅助功能” (H2)
- 接口信号、功能块和数据块一览
  → 参数手册 2

2.2.2 发送至 PLC 的就绪信号

DB10 DBX104.7（NC-CPU 就绪）

NC-CPU 运行就绪并循环向 PLC 报告。

DB10 DBX108.3（操作面板上连接的 HMI - CPU 就绪）

SINUMERIK Operate 已运行就绪并循环向 NC 报告。

DB10 DBX108.5（驱动处于循环运行中）

前提条件：在 NC **所有的**机床轴上，相应的驱动都处于循环运行中，即驱动和 NC 循环交换 PROFIdrive 报文。

DB10 DBX108.6（驱动就绪）

前提条件：在 NC **所有的**机床轴上，相应的驱动以及外部驱动已运行就绪，通过 PROFIBUS 发出就绪信号。

DB31, ... DBX93.5 == 1（驱动就绪）

DB10 DBX108.7（NC 就绪）

NC 已运行就绪。

2.2.3 发送至 PLC 的状态信号

DB10 DBX103.0（远程诊断生效）

HMI 组件报告给 PLC 远程诊断（选件!）已生效，即通过一个外部 PC 来操作控制系统。

DB10 DBX109.6（气温报警）

环境温度监控或风扇监控响应。

DB10 DBX109.7（NC 电池报警）

电池电压已降到限值以下。控制系统可以继续运行。关闭控制系统或断开电源电压会导致数据丢失。

DB10 DBX109.0（存在 NC 报警）

NC 报告至少出现一个 NC 报警。可通过通道专用的接口查询该报警涉及哪些通道以及是否触发了程序处理停止。

DB21, ... DBX36.6（存在通道专用 NC 报警）

NC 报告给 PLC 相关通道中至少出现一个 NC 报警（另见 DB21, ... DBX36.7）

DB21, ... DBX36.7（存在导致加工停止的 NC 报警）

NC 报告给 PLC 相关通道中至少出现一个 NC 报警，此时当前的零件程序处理已中断或终止。

2.2.4 发送至/来自操作面板的信号

DB19 DBX0.0（屏幕待机未生效）

屏幕待机模式失效，屏幕恒亮。

DB19 DBX0.1（屏幕待机生效）

屏幕待机功能生效，屏幕变暗。

通过接口信号激活屏幕待机时：

- 不能再通过键盘（见下）重启屏幕。
- 操作面板前端的第一次按键操作立即触发相应动作。

说明

为了避免通过接口信号激活屏幕待机后的误操作，我们推荐**同时**锁定键盘：
DB19 DBX0.1 = 1 且 DB19 DBX0.2 = 1（按键禁用）

通过键盘或自动的屏幕保护程序来激活屏幕待机

如果在设定的时间内没有在操作面板上进行任何操作（默认时间为 3 分钟）：
MD9006 $MM_DISPLAY_SWITCH_OFF_INTERVAL（屏幕待机定时）
那么屏幕会自动便暗。

屏幕进入待机模式后，按下一个按键就可以重启屏幕。该按键操作不会触发其他动作。

参数设置：

- DB19 DBX0.1 = 0
- MD9006 $MM_DISPLAY_SWITCH_OFF_INTERVAL > 0

DB19 DBX0.2（按键禁用）

通过相连键盘进行的所有输入都被禁用。

DB19 DBX 0.3 / 0.4（删除 Cancel 型报警/删除 Recall 型报警）

请求删除当前所有带有删除条件 Cancel 或 Recall 的报警。报警的删除操作是由以下接口信号应答的：

- DB19 DBX20.3（Cancel 型报警已删除）
- DB19 DBX20.4（Recall 型报警已删除）

DB19 DBX0.7（WCS 中的实际值，0=MCS）

在机床坐标系和工件坐标系之间切换实际值显示：

- DB19 DBX0.7 = 0:机床坐标系（MCS）
- DB19 DBX0.7 = 1:工件坐标系（WCS）

DB19 DBB13（控制文件传输）

用于控制文件传输的任务字节。任务处理的是接口信号中的用户控制文件：

- DB19 DBB16（零件程序管理：用户文件名控制文件编号）
- DB19 DBB17（零件程序管理：用户列表中待传输文件的索引）

DB19 DBB16（控制文件传输）

控制字节，指定在传输文件时控制文件的索引（工作列表）。该文件根据 DB19 DBB13 中的任务处理：

DB19 DBB17（零件程序处理：用户列表中待传输文件的索引）

控制字节，指定在传输文件时待传输的控制文件位于用户控制文件的哪一行。

DB19 DBB26（零件程序处理：状态）

状态字节，指出在“选择”、“装载”或“卸载”时数据传输的当前状态或数据传输是否有误。

DB19 DBB27（程序管理故障）

输出字节，指出文件传输的故障值

2.2.5 发送至通道的信号

DB21, ... DBX6.2（删除剩余行程）

一旦该接口信号发出上升沿，在相应的 NC 通道中，轴便通过当前生效的轨迹减速度在编程的轨迹上停止。 接着，剩余行程被删除，程序切换到下一条程序段。

2.2.6 发送至进给轴/主轴的信号

DB31, ... DBX1.0（驱动测试：运行使能）

注意
驱动测试说明 驱动测试期间人员和机器的安全完全由机床制造商/调试人员负责，机床制造商/调试人员必须通过适当的措施/检查确保安全。

如果由特殊的测试功能（如函数发生器）来运行机床轴，则需要明确请求获得驱动测试使能：

DB31, ... DBX61.0 = 1（驱动测试：运行请求）

一旦运行使能生效，便会执行轴运行。

DB31, ... DBX1.0 = 1（驱动测试：运行使能）

DB31, ... DBX1.3（进给轴/主轴禁止）

静止轴

对静止的轴置位该接口信号时，系统会忽略所有运行请求。

运行请求仍然保留。如果此时再次取消了进给轴禁止（即 DB31, ... DBX1.3 = 0），便会执行运行请求。

运行轴

对运行的轴置位该接口信号时，该轴会按照其当前生效的制动特性曲线停止。如果在复位了接口信号后运行请求仍然存在，轴会继续运行。

主轴

- 控制模式：立即输出转速设定值零
- 定位模式：见上“静止轴”/“运行轴”

DB31, ... DBX1.4（跟踪运行）

该接口信号只能与接口信号 DB31, ... DBX2.1（控制器使能）共同生效

DB31, ... DBX2.1	DB31, ... DBX1.4	运行方式
1	未生效	闭环控制模式
0	1	跟踪运行
0	0	停止

跟踪

在跟踪运行时，机床轴的设定位置会持续跟踪实际位置（设定位置等于实际位置）。

反馈信息：DB31, ... DBX61.3 == 1（跟踪生效）

在“跟踪运行”期间，夹紧监控或静态监控**无效**。

说明

如果在跟踪运行期间置位接口信号 DB31, ... DBX2.1（控制器使能），在 NC 程序激活时，系统会通过 REPOSA 重新定位至最后编程的位置（所有轴沿直线运动）。在其他情况中，所有轴运行都是在当前实际位置处开始的。

停止

在“停止”模式下，机床轴的设定位置不会跟踪实际位置。如果机床轴从设定位置处运行，设定位置与实际位置之间的偏差（跟随误差）会越来越大。在设置控制器使能时跟随误差会瞬间（不按照轴加速特性曲线）减小（转速骤降）。

反馈信息：DB31, ... DBX61.3 == 0（跟踪生效）

在“停止”期间，夹紧监控或静态监控**生效**。

说明

在设置控制器使能时跟随误差会瞬间（不按照轴加速特性曲线）减小（转速骤降）。

应用示例

设置“伺服使能”、夹紧操作结束后 Y 轴的定位方式。此时机床轴因夹紧操作从实际位置 Y_1 被挤到夹紧位置 Y_k。

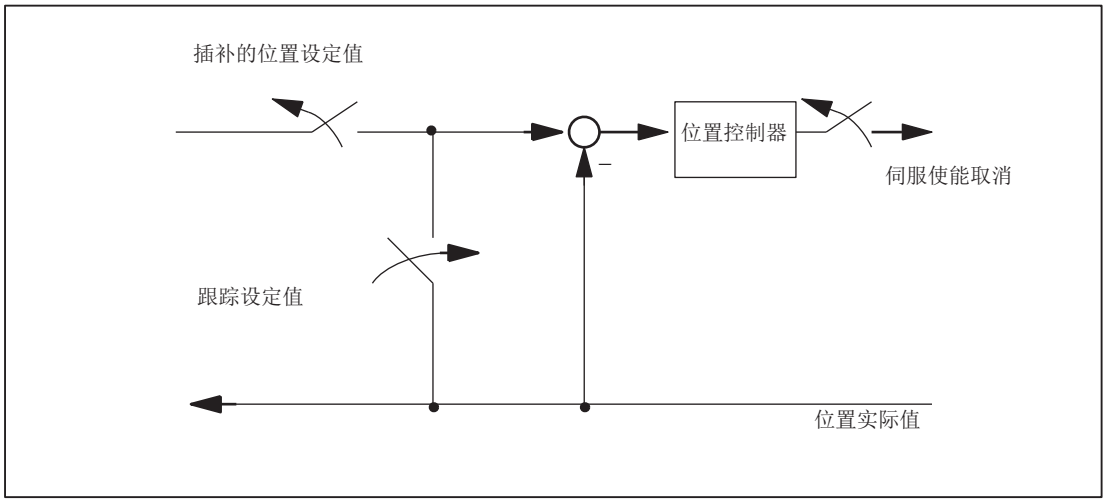


图 2-1　　伺服使能和跟踪运行的作用

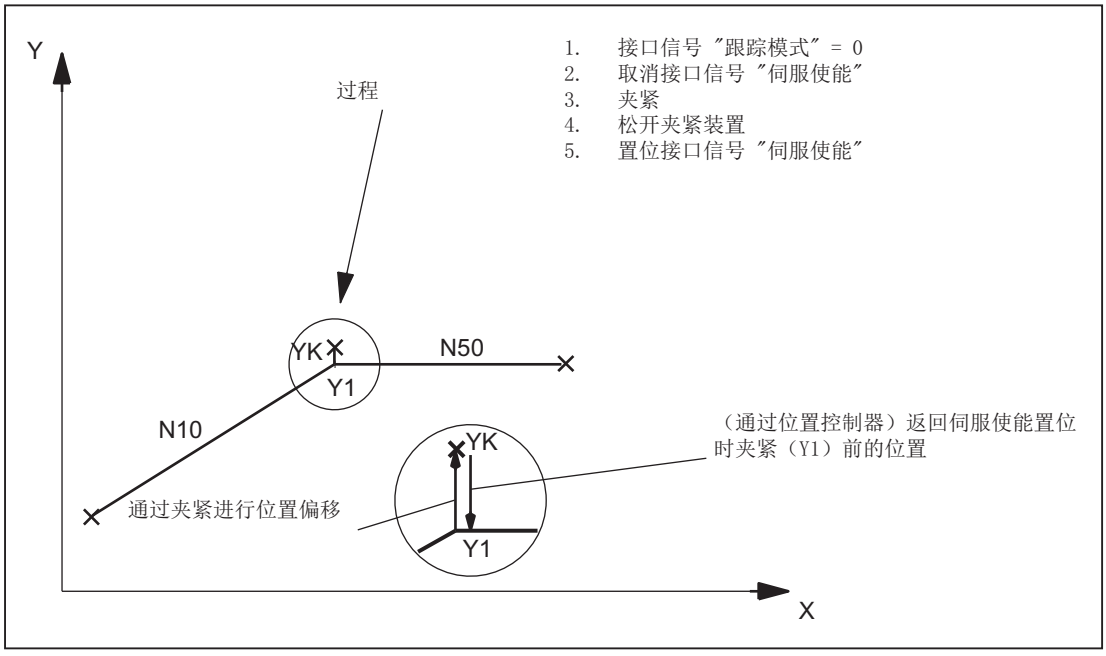


图 2-2　　夹紧操作和“停止”时的轨迹图

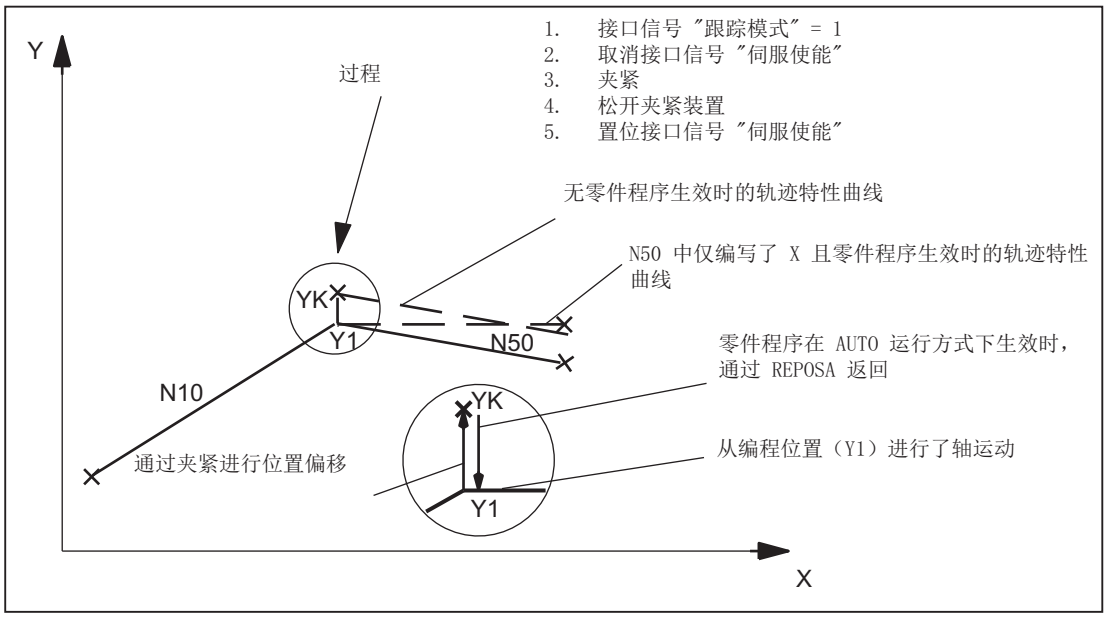


图 2-3　　夹紧操作和“跟踪运行”时的轨迹图

带模拟设定值接口的驱动

带模拟设定值接口的驱动可通过一个外部设定值来运行机床轴。如果已经为机床轴设置了“跟踪运行”，就可以继续采集实际位置。跟踪运行取消后无需回参考点。

对此建议进行以下操作：

1. 激活跟踪运行：
   DB31, ... DBX2.1 = 0（伺服使能）
   DB31, ... DBX1.4 = 1（跟踪运行）（在同一个或之前的 OB1 循环中）
   → 进给轴/主轴位于跟踪运行中
2. 设置外部伺服使能，给出外部转速设定值
   → 进给轴/主轴以外部设定值运转
   → NC 继续采集实际位置并使设定位置跟踪实际位置
3. 撤销外部伺服使能和外部转速设定值
   → 进给轴/主轴停止
4. 取消跟踪运行
   DB31, ... DBX2.1 = 1（伺服使能）
   DB31, ... DBX1.4 = 0（跟踪运行）
   → NC 与当前实际位置同步。后续轴运行从该位置开始。

说明

由于“跟踪运行”只有和“伺服使能”组合使用时才有效，因此可继续保持置位状态。

取消跟踪运行

如果在跟踪运行期间没有超过激活编码器允许的最大编码器极限频率，则取消跟踪运行后，机床轴无需重新回参考点。如果超过了编码器极限频率，控制系统会识别出这一错误：

- DB31, ... DBX60.4 / 60.5 = 0（已回参考点/已同步 1/2）
- 报警：“21610 超过编码器频率”

说明

如果参与有效坐标转换（如：TRANSMIT）的机床轴被取消了“跟踪运行”，则参与该坐标转换的其他机床轴可能会在再定位 (REPOS) 期间运转。

监控

若机床轴位于跟踪运行中，以下监控不会生效：

- 静态监控
- 夹紧监控
- 定位监控

对其他接口信号的影响：

- DB31, ... DBX60.7 = 0（达到精准停位置窗口）
- DB31, ... DBX60.6 = 0（达到粗准停位置窗口）

DB31, ... DBX1.5 / 1.6（位置测量系统 1/2）

一个机床轴上可连接 2 个测量系统，如：

- 间接电机测量系统
- 负载上的直接测量系统

一个时间点上只能有一个测量系统生效。机床轴的所有控制运行、定位运行等总是以生效的测量系统为基准。

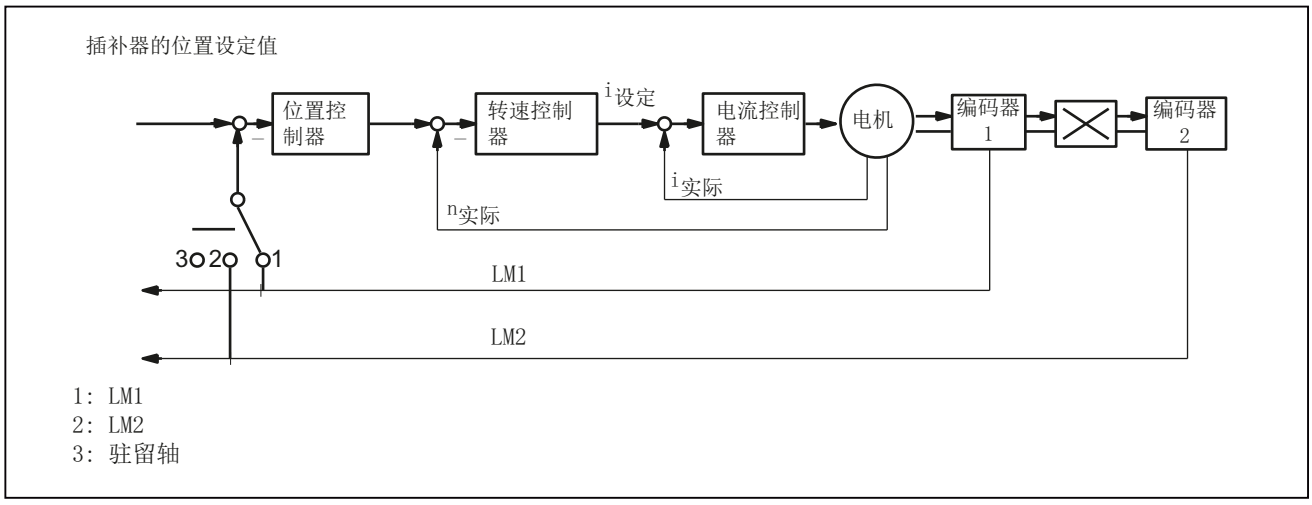


图 2-4　　位置测量系统 1 和 2

与“控制器使能”相关的接口信号“位置测量系统 1 / 2”的功能：

DB31, ... DBX1.5	DB31, ... DBX1.6	DB31, ... DBX2.1	功能
1	任意	1	位置测量系统 1 生效
0	1	1	位置测量系统 2 生效
0	0	0	“驻留”生效
0	0	1	不带位置测量系统的主轴（转速控制）
1 → 0	0 → 1	1	切换：位置测量系统 1 → 2
0 → 1	1 → 0	1	切换：位置测量系统 2 → 1

DB31, ... DBX2.1（伺服使能）

设置伺服使能后，机床轴的位置控制回路闭合。机床轴位于位置闭环控制中。

DB31, ... DBX2.1 == 1

撤销伺服使能后，机床轴的位置控制回路打开，机床轴的转速控制回路延迟一段时间后打开。

DB31, ... DBX2.1 == 0

激活方式

机床轴的伺服使能由以下信号控制：

- NC/PLC 接口信号：
  - DB31, ... DBX2.1（伺服使能）
  - DB31, ... DBX21.7（脉冲使能）
  - DB31, ... DBX93.5（驱动就绪）
  - DB10, DBX56.1（急停）
- NC 内部
  作为报警响应的报警可撤销机床轴的伺服使能。关于报警响应的说明请参见：
  文档：
  诊断手册

在机床轴停止时撤销伺服使能：

- 打开机床轴的位置控制回路
- DB31, ... DBX61.5 == 0（位置控制器生效）

在机床轴运转时撤销伺服使能：

如果机床轴参与了插补轨迹运行或耦合，而撤销了该轴的伺服使能，那么所有相关的轴都会快速停止（转速设定值 = 0）并显示报警：

报警：“21612 运行期间复位了伺服使能”

- 在故障状态下，机床轴会以快速停止（转速设定值 = 0）按设置的制动斜坡时长制动：
  MD36610 $MA_AX_EMERGENCY_STOP_TIME（故障状态下制动斜坡的最大时长）
  会发出报警：
  报警：“21612 运行期间复位了伺服使能”

  说明
  伺服使能会在封锁延时后被撤销：
  MD36610 $MA_AX_EMERGENCY_STOP_TIME

- 打开机床轴的位置控制回路。这一状态通过以下接口信号反馈：
  DB31, ... DBX61.5 == 0（位置控制器生效）。
  伺服使能的封锁延时开始时间可由以下机床数据设置：
  MD36620 $MA_SERVO_DISABLE_DELAY_TIME（伺服使能封锁延时）
- 一旦实际速度达到静态速度区时，即撤销驱动的伺服使能。这一状态通过以下接口信号反馈：
  DB31, ... DBX61.6 == 0（转速控制器生效）

- 机床轴的位置实际值继续由系统采集。
- 制动过程结束时，机床轴会不受相应的 NC/PLC 接口信号的影响自动进入跟踪运行。静态监控和夹紧监控此时未生效。有关接口信号的描述见上文：
  DB31, ... DBX1.4（跟踪运行）。

实际值同步（回参考点运行）

如果在机床轴不位于位置闭环控制的这段时间内，没有超过允许的测量系统的最大极限频率，那么在设置伺服使能后就无需重新同步机床轴的实际位置（回参考点运行）。

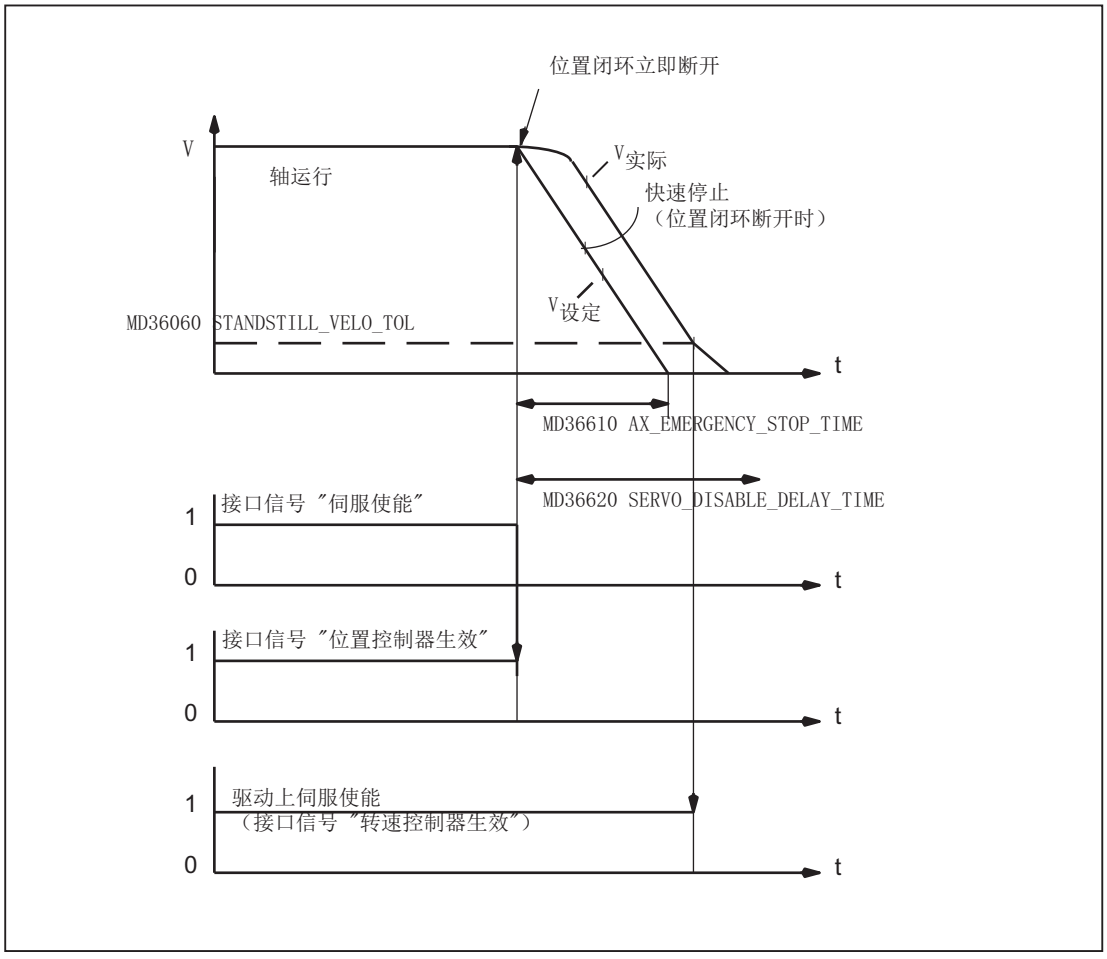


图 2-5　　在机床轴运转时撤销伺服使能

DB31, ... DBX2.2（剩余行程/主轴复位（进给轴/主轴专用））

只有和定位轴组合使用时，“删除剩余行程”才能在 AUTOMATIC 或 MDI 运行模式下生效。定位轴此时会通过当前的制动特性曲线减速至静止状态。还未运行的轴剩余行程会被删除。

主轴复位

对主轴复位的详细说明请参见章节“S1：主轴 (页 1401)”。

DB31, ... DBX9.0 / 9.1 / 9.2（控制器参数组）

请求激活给定的控制器参数组。

控制器参数组	DBX9.2	DBX9.1	DBX9.0
1	0	0	0
2	0	0	1
3	0	1	0
4	0	1	1
5	1	0	0
6	1	0	1
7	1	1	0
8	1	1	1

必须通过以下机床数据使能参数组切换（主轴上不需要）：

MD35590 $MA_PARAMSET_CHANGE_ENABLE = 1 或 2

对参数组切换的详细说明请参见章节“齿轮档切换时的参数组选择 (页 1458)”。

在机床轴运行时进行参数组切换

参数组切换的特性取决于切换时控制回路增益系数 K_V 是否变化：

MD32200 $MA_POSCTRL_GAIN（KV 系数）

- “K_V 系数相同”或“位置闭环控制未生效”：
  NC 立即对参数组切换作出响应。在运动期间也会对参数组进行切换。
- “K_V 系数不相同”和“位置闭环控制生效”：
  为保证平缓的参数组切换，NC 会等到轴“静止”，即达到或低出指定的静态速度后才切换参数组。
  DB31, ... DBX61.4 = 1（进给轴/主轴停止）
  MD36060 $MA_STANDSTILL_VELO_TOL（“进给轴/主轴停止”阈值速度/转速）

在零件程序中进行参数组切换

为了在零件程序中进行参数组切换，用户（即机床制造商）必须自定义辅助功能并在 PLC 用户程序中编写这些功能。PLC 用户程序接着发出请求，请求切换到对应的参数组。

对辅助功能输出的详细说明请参见章节“H2：输出到 PLC 的辅助功能 (页 429)”。

DB31, ... DBX9.3（由 NC 进行的参数组设定被禁止）

接口信号置位时，系统会忽略控制器-参数组切换 (DB31, ... DBX9.0 / 9.1 / 9.2) 请求。

2.2.7 来自进给轴/主轴的信号

DB31, ... DBX61.0（驱动测试：运行请求）

如果由特殊的测试功能（如函数发生器）来运行机床轴，则需要明确请求获得驱动测试使能：

DB31, ... DBX61.0 = 1（驱动测试：运行请求）

一旦运行使能生效，便会执行轴运行。

DB31, ... DBX1.0 = 1（驱动测试：运行使能）

DB31, ... DBX61.3（跟踪生效）

机床轴位于跟踪运行中。

DB31, ... DBX61.4（进给轴/主轴停止（$n < n_{min}$）

“进给轴/主轴停止”是由 NC 设置的，当：

- 不再输出任何新的设定值且
- 机床轴的实际速度小于指定的静态速度：
  MD36060 $MA_STANDSTILL_VELO_TOL（“进给轴停止”阈值速度）

DB31, ... DBX61.5（位置控制器生效）

机床轴的位置控制回路闭合，位置闭环控制生效。

DB31, ... DBX61.6（转速控制器生效）

机床轴的转速控制回路闭合，转速控制生效。

DB31, ... DBX61.7（电流控制器生效）

机床轴的电流控制回路闭合，电流控制生效。

DB31, ... DBX69.0 / 69.1 / 69.2（伺服参数组）

生效的参数组 编码符合：

DB31, ... DBX9.0 / 9.1 / 9.2（控制器参数组选择）

DB31, ... DBX76.0（润滑脉冲）

控制系统上电/复位后，信号状态为 0(FALSE)。

一旦机床轴退回了指定的润滑行程，润滑脉冲便会**取反**（脉冲沿切换）。

MD33050 $MA_LUBRICATION_DIST（PLC 的润滑行程）

2.2.8 发送至进给轴/主轴的信号（数字量驱动）

DB31, ... DBX21.0 - 4 (电机数据组和/或驱动数据组的切换请求)

由 PLC 生成对驱动的请求，要求切换到新的电机和/或驱动数据组。

接口可灵活设置，通过：DB31, ... DBX130.0 - 4（参见章节“切换电机数据组和驱动数据组 (页 76)”）

激活的电机数据组和/或驱动数据组的显示通过以下接口进行：DB31, ... DBX93.0 - 4

说明

主主轴驱动

在进行主主轴驱动时，只有电机数据组 1 和 2 可用于星形和三角形接线方式：

- 电机数据组 1： 星形接线方式
- 电机数据组 2： 三角形接线方式

DB31, ... DBX21.5（选择了电机）

PLC 用户程序通过该信号向驱动报告电机选择结束。 接着由驱动使能脉冲。

DB31, ... DBX21.6（转速控制器的积分器禁止）

PLC 用户程序取消了驱动上转速控制器的积分器使能。 此时转速控制器从 PI 控制器转换成单纯的比例控制器。

在激活转速控制器的积分器禁用时，取决于应用，可能会出现补偿过程（例如：当积分器之前始终保持一个负载时）。

反馈通过： DB31, ... DBX93.6 = 1（转速控制器的积分器被禁用）。

DB31, ... DBX21.7（脉冲使能请求）

只有当所有的使能信号（硬件和软件）都生效时，才会请求驱动模块的脉冲使能。

- 触发装置使能
- 伺服使能和脉冲使能
- 脉冲使能 (Safe Operation Stop)
- 存储的硬件输入
- 设定值使能
- “状态：运行就绪”
  - 无驱动报警（ZK1 故障）
  - 直流母线已接通
  - 加速已完成

反馈通过：DB31, ... DBX93.7（脉冲已使能）

2.2.9 来自进给轴/主轴的信号（数字量驱动）

DB31, ... DBX92.1（斜坡函数发生器禁用生效）

驱动反馈给 PLC，斜坡函数发生器快速停止已生效。 随后驱动不使用斜坡功能（转速设定值为 0）停止。

DB31, ... DBX93.0 - 4（生效的电机和驱动数据组的显示）

驱动会将生效的电机和驱动数据组反馈给 PLC。

接口可灵活设置，通过：DB31, ... DBX130.0 - 4（参见章节“切换电机数据组和驱动数据组 (页 76)”）

电机数据组和/或驱动数据组的切换请求通过以下接口进行： DB31, ... DBX21.0 - 4

DB31, ... DBX93.5（驱动就绪）

反馈，驱动已准备就绪。 这表明进给轴/主轴运动的前提条件已满足。

DB31, ... DBX93.6（转速控制器的积分器被禁用）

转速控制器的积分器被禁用。 因此转速控制器从 PI 控制器转换成单纯的比例控制器。

DB31, ... DBX93.7（脉冲已使能）

驱动模块的脉冲使能已给出。 进给轴/主轴可以运行。

DB31, ... DBX94.0（电机温度预报警）

电机温度超出了设置的电机温度报警阈值（驱动参数 p0604）。

另见下文对 “DB31, ... DBX94.1（散热器温度预警）” 的说明。

DB31, ... DBX94.1（散热器温度预警）

功率部件散热器温度超出了允许的范围。 如果持续保持高温，则驱动在约 20 秒后关闭。

说明

温度预报警 DB31, ... DBX94.0 和 DBX94.1

接口信号是由以下循环驱动报文的信号导出的：

- 情形 1： 信息字中的温度报警
  - DB31, ... DBX94.0 = MELDW，位 6（无电机过热报警）
  - DB31, ... DBX94.1 = MELDW，位 7（无功率部件热过载报警）
- 情形 2： B 级报警（只在 “SIMODRIVE 611u” 接口模式中，p2038=1）
  DB31, ... DBX94.0 == 1 且 DBX94.1 == 1；
  当循环驱动报文，状态字 1（ZSW1），位 11 / 12 == 2（警告级 B）时

当没有来自信息字的特殊信息时，便从 B 级报警中引导出接口信号。

显示报警： 报警号 = 200.000 + 报警值 (r2124)

对设置电机温度监控的详细说明请参见：

文档：

- S120 调试手册，章节 “调试” > “SINAMICS 组件上的温度传感器”
- S120 功能手册，章节 “监控功能和保护功能”
- S120 参数手册
  - MELDW，位 6 ≙ BO：r2135.14 → 功能图：2548、8016
  - MELDW，位 7 ≙ BO：r2135.15 → 功能图：2548、2452、2456、8016

DB31, ... DBX94.2（加速过程结束）

转速实际值在转速设定值变化后再次位于指定的公差带内。 到此加速过程结束。

后续因负载变化而导致的转速变化（例如：超出公差带）对该接口信号无影响。

DB31, ... DBX94.3 ($|M_d| < M_{dx}$)

当前转矩的绝对值 $|M_d|$ 小于指定的转矩阈值 M_{dx}（转矩阈值 2，p2194）。

转矩阈值是当前和转速相关的转矩限值的百分比 [%]。

DB31, ... DBX94.4 ($|n_{act}| < n_{min}$)

转速实际值 n_{act} 小于 n_{min}（转速阈值 3，p2161）。

DB31, ... DBX94.5 ($|n_{act}| < n_x$)

转速实际值 n_{act} 小于 n_x（转速阈值 2，p2155）。

DB31, ... DBX94.6 ($n_{act} = n_{set}$|)

转速实际值位于转速设定值的公差带 (p2163) 内。

DB31, ... DBX95.7（出现 C 级报警）

驱动报告出现 C 级报警。

2.3 功能

2.3.1 过载时的图像更新特性 - 仅适用于 840D sl

在某些零件程序中，主处理须等待，直至预处理提供新程序段。

预处理和显示更新会争夺 NC 计算时间。

可通过以下机床数据设置预处理过慢时的 NC 特性：

MD10131 $MN_SUPPRESS_SCREEN_REFRESH

值	含义
0	当通道的预处理过慢时，抑制所有通道中的显示更新。
1	当通道的预处理过慢时，仅在时间关键通道中抑制抑制显示更新，以获取计算时间用于预处理。
2	基本不抑制显示更新。

2.3.2 渐开线插补的设置 - 只针对 840D sl

引言

将一个圆轴固定在一个平面上，轴上缠线，拉紧一个线头，让该线绕圆轴运动且始终与圆轴相切，那么线上一个定点在该平面上的轨迹就是渐开线。 渐开线插补使得轨迹曲线沿渐开线运动。

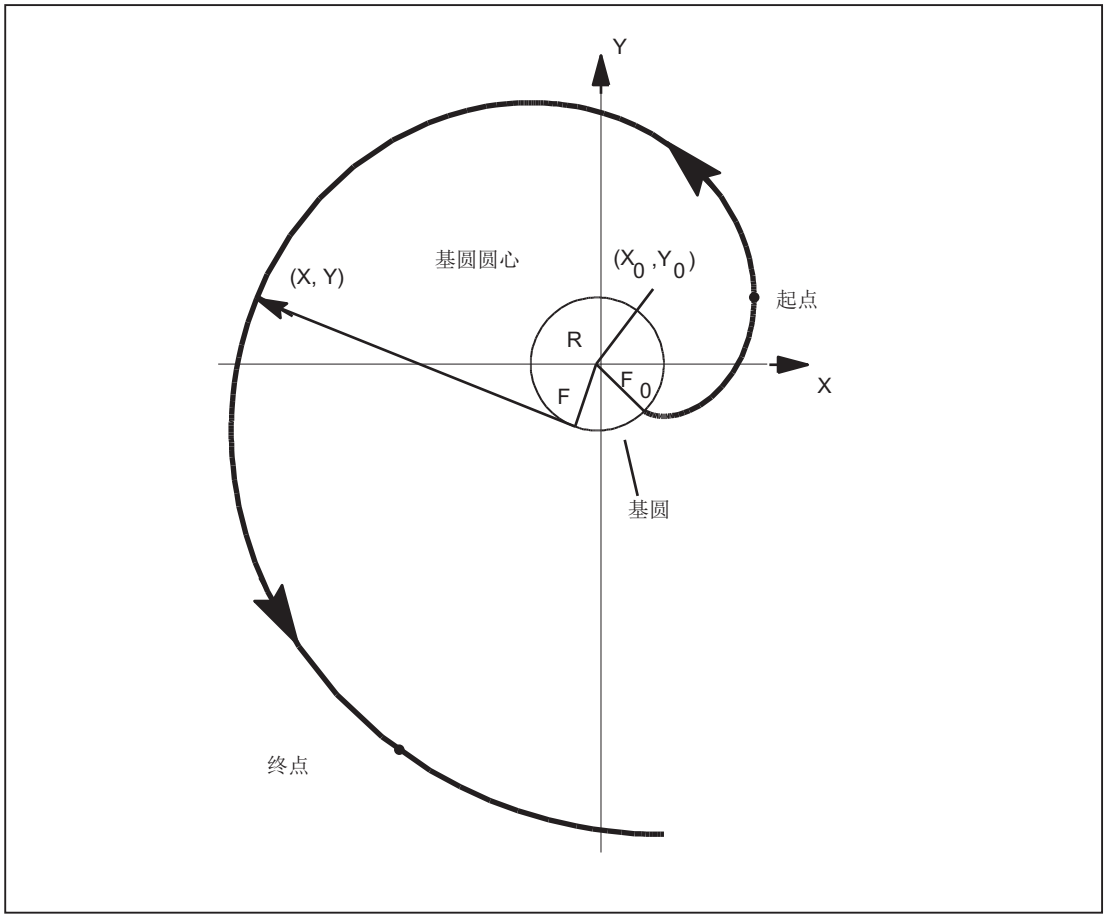


图 2-6 渐开线（起点在基圆上）

编程

有关如何编程渐开线插补的大致描述请参见：

文档：
编程手册之基本原理分册

对于渐开线插补的两种情况，除编程的参数外，还有些机床数据须由机床制造商/最终用户设置。

精度

如果编程的终点不能准确的落在由起点定义的渐开线上，那么在这两条由起点或终点定义的渐开线之间进行插补（参见下图）。

终点的最大偏差由机床数据来确定：

MD21015 $MC_INVOLUTE_RADIUS_DELTA（渐开线上的终点监控）

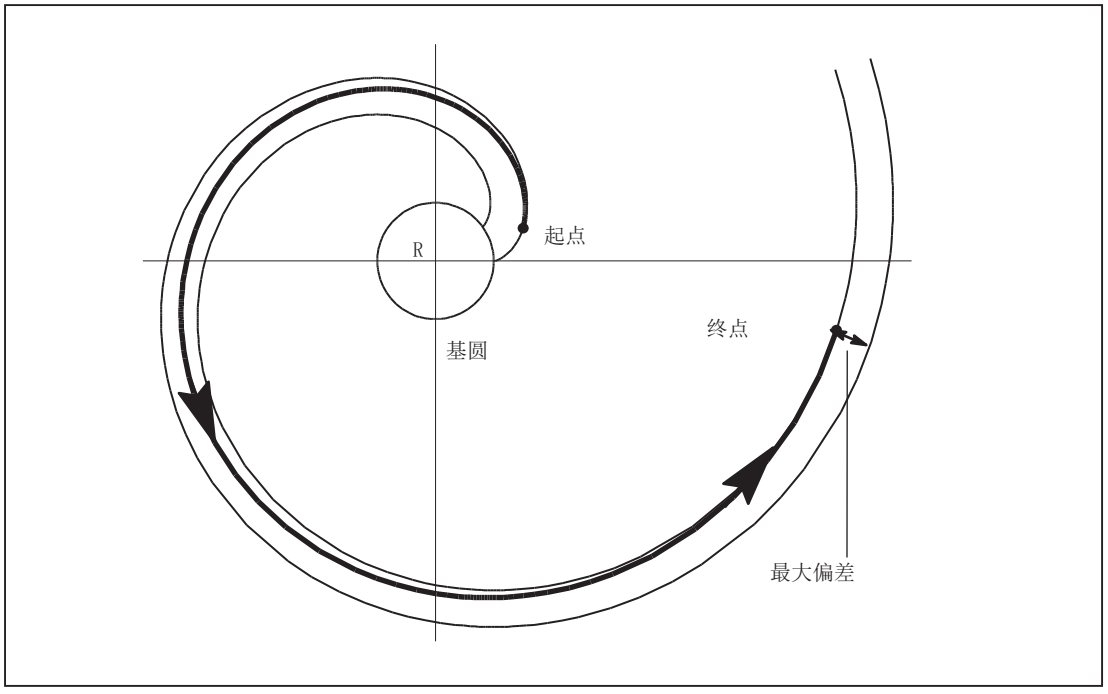


图 2-7　　MD21015 确定了允许的最大偏差

极限角度

如果使用 AR 为朝基圆运动的渐开线编程一个大于最大允许值的旋转角，则会输出报警并停止程序处理。

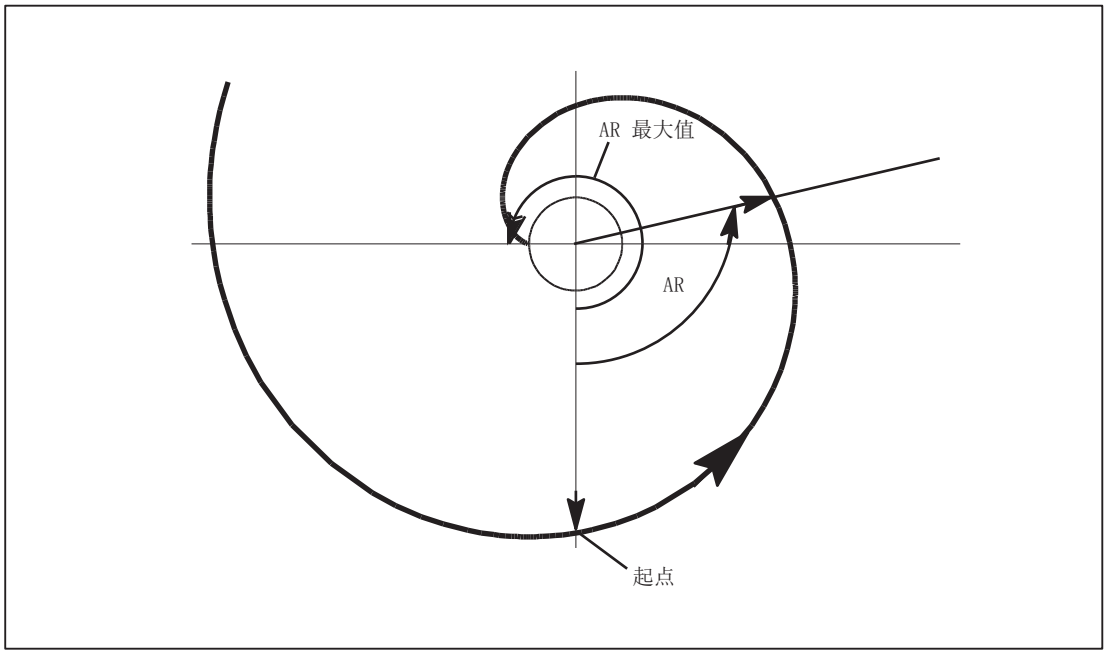


图 2-8　　终点在基圆上渐开线最大旋转角

可通过以下设置来抑制报警显示：

MD21016 $MC_INVOLUTE_AUTO_ANGLE_LIMIT = TRUE（渐开线插补时的自动角度限制）

之后会自动限制编程的旋转角，且插补轨迹会结束在渐开线和基圆相交的点上。 借助该设置可简单地编程一条起点不在基圆上、终点在基圆上的渐开线。

刀具半径补偿

渐开线只支持 2-1/2D 刀具半径补偿。 如果激活了 3D 刀具半径补偿（圆周铣削和端面铣削），会在编程渐开线时终止程序处理并输出报警 10782。

在 2-1/2-D 刀具半径补偿中，渐开线的平面必须在补偿平面内。 若不在范围内，则输出报警 10781。 但补偿平面内除了有渐开线外，还可以有螺旋线。

动态响应

在基圆上开始或结束的渐开线在该位置上有一个无限曲率。 为了在刀具半径补偿激活时限制该点上的速度（但不会过分限制其他位置上的速度），必须激活“速度限制曲线”功能：

MD28530 $MC_MM_PATH_VELO_SEGMENTS > 1（用于限制轨迹速度的存储器单元数量）

推荐使用值 5。如果使用的渐开线线段的弯曲半径小幅变化，则无需该设置。

2.3.3 激活 DEFAULT 存储器 - 只针对 840D sl

GUD 起始值

通过语言指令 DEF... / REDEF... 可以分配全局用户变量（GUD）缺省值。如果要使该缺省值在设置的初始化时间点可供使用，例如上电后通过 INIPO 属性，就必须将该缺省值永久保存在系统中。该缺省值的存储空间必须由以下机床数据进行使能：

MD18150 $MM_GUD_VALUES_MEM （用于 GUD 值的可记忆存储空间）

文档：

- 功能手册之扩展功能分册；S7：“存储器配置”
- 编程手册之工作准备

2.3.4 读取和写入 PLC 变量 - 只针对 840D sl

高速数据通道

在该模块 (DPR) 的通讯缓冲器中预留一个存储区以实现 PLC 和 NC 之间的高速信息交换。在该存储区中可以交换任意多个 PLC 变量（I/O、DB、DW、标记）。

PLC 通过‘FunctionCalls’ (FC) 访问该存储器而 NC 通过系统变量访问该存储器。

划分存储区

用户程序的编程人员（NC 和 PLC）自行负责存储区的划分。

此时可任意划分存储区，但是必须根据数据格式选择限制（“DWORD” 限制为 4 个字节，“WORD” 限制为 2 个字节，等等）。

存储区是通过指定数据类型和存储区内的位置偏移来访问的。

从 NC 访问

NC 系统中提供变量以便从零件程序或同步动作快速访问 PLC 变量。可通过 NC 直接读写数据。数据类型由系统变量的名称得出。存储区内的位置作为索引指定，单位为字节。

系统变量	数据类型	取值范围
$A_DBB（<索引>）	字节（8 位）	0 <= x <= 255
$A_DBW（<索引>）	字（16 位）	-32768 <= x <= 32767
$A_DBD（<索引>）	双字（32 位）	-2147483648 <= x <= 2147483647
$A_DBR（<索引>）	浮点（32 位）	$\pm(1.5 \cdot 10^{-45} <= x <= 3.4 \cdot 10^{38})$

从 PLC 访问

PLC 借助“FunctionCall”(FC) 来访问存储器。在 FC 中，立即（而非在 PLC 循环开始时才）在 DPR 中读写数据。数据类型和存储区中的位置被作为参数传输至 FC。

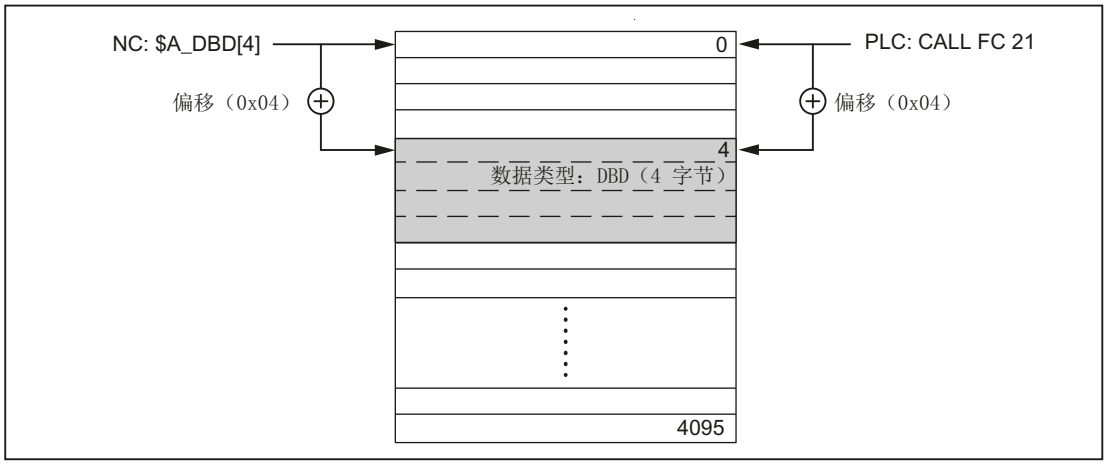


图 2-9　　用于 NC/PLC 通讯的通信缓冲器 DPR

前提条件

- DPR 存储区的划分完全由用户负责。系统不会对配置的一致性进行检查。
- 在输入和输出方向上共有 4096 个字节可供使用。
- 系统不支持单个位指令，用户必须使用字节指令。
- 由于变量的内容是直接在通讯缓冲器中被修改的，用户程序的编程人员应注意：在多次分析一个变量或多个变量互联时，值可能被中途修改（即有必要将值暂时保存在局部变量或 R 参数中或安装信号装置）。

- 用户编程人员应负责协调不同通道对通讯缓冲器的访问。
- 在数据访问中，系统可保证 16 位以内（字节型和字型）数据的一致性。32 位数据（双字型和实数型）的一致性由用户自行负责。为此在 PLC 侧提供了一个简单的信号装置。
- PLC 将数据以 ‘Little Endian’ 格式保存在 DPR 中。
- 通过 $A_DBR 传输的值需要经过转换，因此会损失一定的精度。浮点型数据在 NC 上为 DOUBLE（64 位），而在 PLC 上却只是 FLOAT（32 位）。DPR 中的保存格式为 FLOAT。转换在保存前后各进行一次。
  例如：如果是从 NC 写入然后再次读取 DPR 中的变量，则会进行两次转换。由于数据是以两种格式保存的，也就无法避免读写的值有偏差。
  示例
  通过比较“EPSILON” 来绕开该问题（细微偏差）

程序代码

```
N10          DEF REAL DBR
N12          DEF REAL EPSILON = 0.00001
N20          $A_DBR[0]=145.145
N30          G4 F2
N40          STOPRE
N50          DBR=$A_DBR[0]
N60          IF ( ABS(DBR/145.145-1.0) < EPSILON ) GOTOF ENDE
N70          MSG（“故障”）
N80          M0
N90          ENDE:
N99          M30
```

激活

可通过 MD28150 $MC_MM_NUM_VDIVAR_ELEMENTS（用于写入 PLC 变量的单元数）来设置
可同时写入的最大输出变量数

示例

一个单字型变量需从 PLC 传送至 NC。

NC 输入端内的位置偏移应为第 4 个字节（PLC 输出区）。位置偏移必须为数据宽度的整数倍。

由 PLC 写入：

程序代码	注释
. . .	
CALL FC21 (	
Enable :=M10.0,	; 若为 TRUE，则 FC21 生效
Funct :=B#16#4,	
S7Var :=P#M 104.0 WORD1,	
IVAR1 :=04,	
IVAR2 :=-1,	
Error :=M10.1,	
ErrCode :=MW12);	
. . .	
)	

在零件程序中读取

程序代码	注释
. . .	
PLCDATA = $A_DBW[4];	; 读取一个单字
. . .	

重新上电后的特性，程序段搜索

在上电时通讯缓冲器 DPR 初始化。

在进行“程序段搜索”时系统会收集输出的 PLC 变量，并通过移动程序段将其传送给通讯缓冲器 DPR（这一过程类似于写入模拟量和数字量输入）。

其他状态过渡在这儿无效。

文档

有关 PLC 通过 FC 21 进行数据交换的详细信息请参见：

SINUMERIK 840D sl:章节“功能 (页 1194)”

2.3.5 通过口令和钥匙开关实现访问保护

访问权限

针对不同的使用者对功能、程序和数据的访问，系统一共设置 8 个保护级。它们分为：

- 多个口令级别，分别用于西门子公司内部、机床制造商和最终用户
- 多个钥匙开关位置，用于最终用户

多级安全方案

系统提供一种由多个口令级别以及钥匙开关位置组成的多级安全方案来控制访问权限。

保护等级	方式	用户	访问对象（示例）
0	口令	西门子	所有功能、程序和数据
1	口令	机床制造商：开发人员	指定的的功能、程序和数据； 例如： 输入选件
2	口令	机床制造商：调试人员	指定的的功能、程序和数据； 例如： 大部分机床数据
3	口令	最终用户：服务	指定的功能、程序和数据
4	钥匙开关位置 3	最终用户： 编程员、调试员	保护等级 0 到 3 以下的数据；由机床制造商或最终用户确定
5	钥匙开关位置 2	最终用户： 有资质无需编程的操作员	保护等级 0 到 3 以下的数据；由最终用户确定
6	钥匙开关位置 1	最终用户： 经过培训、无需编程的操作员	示例： 只能选择程序、输入刀具磨损数据以及零点偏移
7	钥匙开关位置 0	最终用户：刚刚入门的操作员	示例： 不可以输入和选择程序， 只可以操作机床控制面板

访问特性

- 保护等级 0 具有最高访问权限，
  保护等级 7 具有最低访问权限。
- 如果给某个保护等级设定了某个访问权限，则比它更高的保护等级会自动纳入该访问权限。
- 反之，如果需要修改某保护等级的访问权限，只能在比它更高的保护等级中进行修改。
- 保护等级 0 到 3 的访问权限由西门子自动设置（缺省设置）。
- 通过询问当前钥匙开关位置和比较输入的口令来设置访问权限。此时，输入的口令会覆盖钥匙开关位置的访问权限。
- 在每个保护等级中都可以保存选项数据。但是只有在保护等级 0 和 1 中才可以输入选项数据。
- 保护等级 4 到 7 的访问权限是推荐值，可以由机床制造商或者最终用户来进行更改。

2.3.5.1 口令

设置口令

通过 HMI 操作界面输入保护等级（0 - 3）的口令：

操作区“调试” > “口令” > “设置口令”

删除口令

通过输入设置的口令获得的访问权限一直保持生效，在明确删除口令后失效：

操作区“调试” > “口令” > “删除口令”

修改口令

通过 HMI 操作界面修改保护等级（0 - 3）的口令：

操作区“调试” > “口令” > “修改口令”

说明

热启动

热启动不会影响访问权限或口令状态（已设置/已删除）！

新口令

一个口令最多可由八个字符组成。选择口令时建议仅使用操作面板上的字符。

输入少于 8 个字符的口令时，系统会将缺少的字符视为空格。

在设定新口令时，必须遵循用于设定安全密码的规则。

说明

设定安全密码

在设定新密码时请遵循以下规则：

- 在设定新密码时，请勿设定易于猜测的密码，例如简单的词、键盘上的易于猜测的按键序列等。
- 密码必须包含大小写字母的组合，以及数字和特殊字符。密码最少须包含 8 个字符。PINS 必须是无规律的数字序列。
- 在可行且 IT 系统和软件支持的情况下，在选择密码时必须采用尽可能复杂的字符。

有关安全密码设定的其他规则参见德国联邦信息安全办公室（BSI）。 (https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/International/GSK_15_EL_EN_Draft.pdf?__blob=publicationFile&v=2)

在采用密码的情况下，可以使用用于密码管理的程序来加以协助。借助该程序能够对密码和口令进行加密保存、管理以及生成安全密码。

缺省设置

缺省状态下，下列口令对保护等级 1 - 3 生效：

- 保护等级 1：SUNRISE
- 保护等级 2：EVENING
- 保护等级 3：CUSTOMER

注意
冷启动会使口令恢复为缺省设置 在冷启动（通过载入缺省机床数据启动 NC）后，保护等级 1 - 3 的口令会重新恢复为缺省设置。因此，出于安全原因，在冷启动后强烈建议修改口令的缺省设置。

文档

- 调试手册“SINUMERIK 840D sl，基本软件和操作软件”；章节“SINUMERIK Operate (IM9)” > “通用设置” > “访问等级”
- 调试手册 “CNC 调试：NC、PLC、驱动”；章节 “调试的前提条件” > “接通和启动”以及“访问等级”

2.3.5.2 钥匙开关位置（DB10, DBX56.4 到 7）

钥匙开关

钥匙开关四个位置，分别对应保护等级 4 到 7。钥匙有不同颜色，它们可以在不同的开关位置插拔。

开关位置	减去位置	DB10, DBB56	保护级
位置 0 	-	位 4	7
位置 1 	0 或 1 黑色钥匙	位 5	6
位置 2 	0 或 1 或 2 绿色钥匙	位 6	5
位置 3 	0 或 1 或 2 或 3 红色钥匙	位 7	4

图 2-10　　开关位置 0 到 3

开关位置

开关位置 0 具有最低访问权限。开关位置 3 具有最高访问权限：

DB10, DBX56.4 / .5 / .6 / .7（开关位置 0 / 1 / 2 / 3）

可根据具体机床为开关位置定义访问程序、数据和功能的权限。详细信息请参见：

文档

- 调试手册，CNC:NC、PLC、驱动；基本原理，
  章节：保护等级原理
- 调试手册 SINUMERIK Operate (IM9)；通用设置，
  章节：访问等级

由 PLC 用户程序设定

钥匙开关位置是通过 PLC 基本程序传送至 NC/PLC 接口的。可通过 PLC 用户程序来修改相应的接口信号。此时，从 NC 角度出发应始终只有一个开关位置是激活的，也就是说只有一个接口信号置 1。如果从 NC 的角度出发同时有多个开关位置激活，那么开关位置 3，即具有最高访问权限的钥匙开关位置将会在 NC 内部激活。

2.3.5.3 可设置的保护等级

可设置的保护等级

可自由设置不同的功能和数据区的保护等级。 可通过名为 $MM_USER_CLASS_<*功能_数据区*>的操作面板机床数据来设置保护等级。

示例：

$MM_USER_CLASS_READ_TOA	读取刀具补偿
$MM_USER_CLASS_WRITE_TOA	写入刀具补偿
$MM_USER_CLASS_READ_PROGRAM	读取零件程序
$MM_USER_CLASS_WRITE_PROGRAM	写入/编辑零件程序

缺省值

除了少数个别情况，在出厂时或进行标准调试后，缺省的保护等级通常为 7，即最低保护等级。

2.3.6 切换电机数据组和驱动数据组

2.3.6.1 简介

电机数据组和驱动数据组

为了最佳地符合不同加工条件或不同机床配置的需求，一台驱动中有时需要具备多个不同的数据组用于电机、驱动参数和编码器。调试期间可借助“驱动向导”来设置驱动对象的基本数据组。

说明

文档

调试手册：CNC:NC，PLC - 驱动，章节“调试 NC 控制的驱动”

下列对数据组的修改和管理是通过操作界面实现的：

SINUMERIK Operate:操作区“调试” > “驱动系统” > “驱动” > “数据组”

具体加工条件中机床轴所需的电机数据组 (MDS) 或驱动数据组 (DDS)必须由 PLC 用户程序通过下文说明的接口来激活。

轴 NC/PLC 接口

用于切换电机数据组和驱动数据组的轴 NC/PLC 接口可分为三个区域：

- 格式接口 (页 76)
- 请求接口 (页 77)
- 显示接口 (页 78)

2.3.6.2 格式接口

格式

通过格式接口可以设置，请求接口和显示接口的哪些位用于电机数据组(MDS)的定址以及哪些位用于驱动数据组(DDS)的定址：

DB31, ... DBX130.0 - 4，其中位 x = <值>

<值>	含义
0	电机数据组(MDS)的位的位置或无效位的位置
1	驱动数据组(DDS)的位的位置

驱动中的电机数据组和驱动数据组

格式取决于驱动中当前电机数据组(MDS)和驱动数据组(DDS)的数量。 各个数量可通过以下驱动参数测定：

- p0130（电机数据组数量）
- p0180（驱动数据组数量）

接口有效性

一旦在控制系统启动时驱动传送了所有需要的信息，控制系统分析完这些信息，请求 (页 77)和显示接口 (页 78)就显示为有效：

DB31, ... DBX130.7 == 1（请求和显示接口有效）

如果驱动没有传送任何的信息或是传送了不兼容的信息，那么请求和显示接口便显示为无效。

说明

当请求和显示接口无效时，用户/机床制造商若通过无效接口进行数据组切换，则所有责任由其自行承担。

参见

示例 (页 78)

接口一览 (页 79)

2.3.6.3 请求接口

切换到新的电机数据组(MDS)和/或驱动数据组(DDS)的请求通过以下接口进行：

DB31, ... DBX21.0 - .4 = <MDS / DDS 下标>

取值范围

电机数据组或驱动数据组的定址 n（n = 1, 2, 3, ...）根据其下标 i 进行，其中 i = n - 1 = 0, 1, 2, ...。

- 电机数据组：MDS[0, 1, 2, ... 15]
- 驱动数据组：DDS[0, 1, 2, ... 31]

接口格式

请求接口的格式，即哪些位用于电机数据组(MDS)的定址以及哪些位用于驱动数据组(DDS)的定址，通过格式接口 (页 76)进行设置。

驱动中的电机数据组和驱动数据组

驱动中现有的电机数据组(MDS)和驱动数据组(DDS)的数量可通过以下驱动参数测定：

- p0130（电机数据组数量）
- p0180（驱动数据组数量）

主主轴驱动的电机数据组(MDS)

主主轴驱动适用以下对应关系：

- MDS[0] → 星形接线方式
- MDS[1] → 三角形接线方式

2.3.6.4 显示接口

生效电机数据组(MDS)和驱动数据组(DDS)通过以下接口进行显示：

DB31, ... DBX93.0 - .4 == <MDS / DDS 下标>

取值范围和格式与请求接口 (页 77)一致。

2.3.6.5 示例

驱动中有两个电机数据组(MDS)，每个电机数据组有两个驱动数据组(DDS)。 这与图 2-11 电机数据组/驱动数据组切换的原理 (页 81)中所列出的编号为 9 的数据组组合一致。

格式

驱动数据组切换 (DDS) 的位位置：

- DB31, ... DBX130.0 == 1

电机数据组切换 (MDS) 的位位置：

- DB31, ... DBX130.1 == 0

无效的位位置：

- DB31, ... DBX130.2 == 0
- DB31, ... DBX130.3 == 0
- DB31, ... DBX130.4 == 0

驱动数据组接口 (DDS)

请求和显示接口相关的位位置：

- DB31, ... DBX21.0 / DBX93.0
  - DB31, ... DBX21.0 / DBX93.0 == 0 ⇒ 第 1 个驱动数据组 DDS[0]
  - DB31, ... DBX21.0 / DBX93.0 == 1 ⇒ 第 2 个驱动数据组 DDS[1])

电机数据组接口 (MDS)

请求和显示接口相关的位位置：

- DB31, ... DBX21.1 / DBX93.1
  - DB31, ... DBX21.1 / DBX93.1 == 0 ⇒ 第 1 个电机数据组 MDS[0]
  - DB31, ... DBX21.1 / DBX93.1 == 1 ⇒ 第 2 个电机数据组 MDS[1]

无效的位位置 (MDS / DDS)

请求和显示接口无效的位位置：

- DB31, ... DBX21.1 / DBX93.2 == 0
- DB31, ... DBX21.1 / DBX93.3 == 0
- DB31, ... DBX21.1 / DBX93.4 == 0

参见

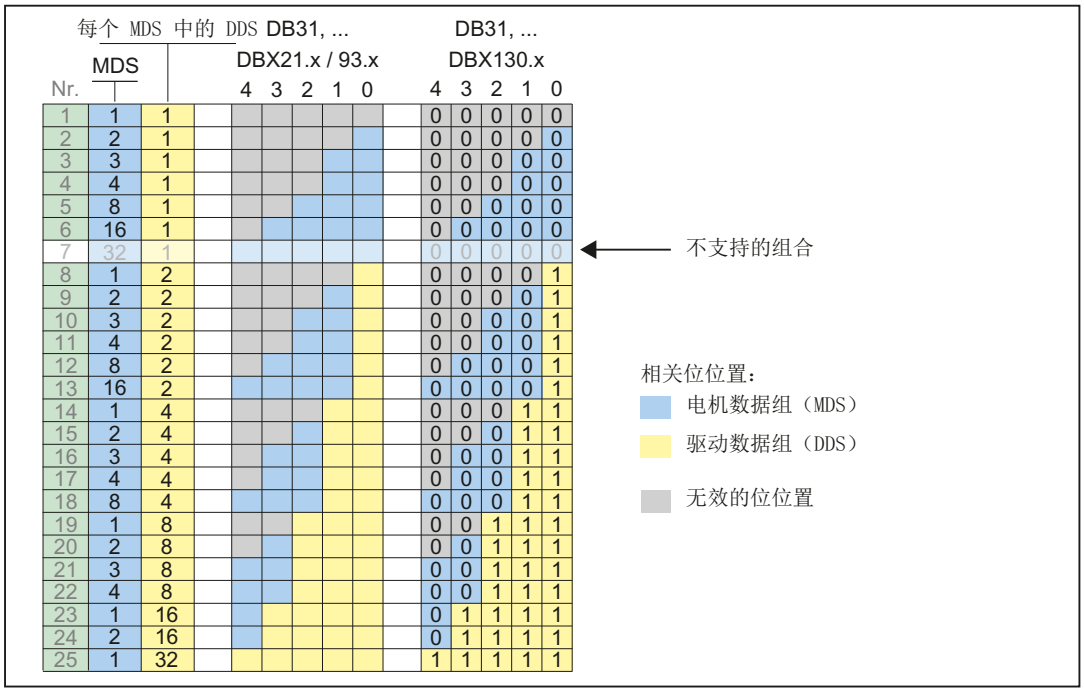
接口一览 (页 79)

2.3.6.6 接口一览

表格 2-1 可配置的 MDS / DDS 组合

MDS（电机）数量	每个 MDS 对应的 DDS（驱动）数量
1	1 ... 32
2	1, 2, 4, 8, 16
3	1, 2, 4, 8
4	1, 2, 4, 8

MDS（电机）数量	每个 MDS 对应的 DDS（驱动）数量
5	1, 2, 4
6	1, 2, 4
7	1, 2, 4
8	1, 2, 4
9	1, 2
10	1, 2
11	1, 2
12	1, 2
13	1, 2
14	1, 2
15	1, 2
16	1, 2



Nr.	MDS	每个 MDS 中的 DDS	DBX21.x / 93.x 4	3	2	1	0	DBX130.x 4	3	2	1	0
1	1	1						0	0	0	0	0
2	2	1						0	0	0	0	0
3	3	1						0	0	0	0	0
4	4	1						0	0	0	0	0
5	8	1						0	0	0	0	0
6	16	1						0	0	0	0	0
7	32	1						0	0	0	0	0
8	1	2						0	0	0	0	1
9	2	2						0	0	0	0	1
10	3	2						0	0	0	0	1
11	4	2						0	0	0	0	1
12	8	2						0	0	0	0	1
13	16	2						0	0	0	0	1
14	1	4						0	0	0	1	1
15	2	4						0	0	0	1	1
16	3	4						0	0	0	1	1
17	4	4						0	0	0	1	1
18	8	4						0	0	0	1	1
19	1	8						0	0	1	1	1
20	2	8						0	0	1	1	1
21	3	8						0	0	1	1	1
22	4	8						0	0	1	1	1
23	1	16						0	1	1	1	1
24	2	16						0	1	1	1	1
25	1	32						1	1	1	1	1

MDS	电机数据组的数量
每个 MDS 中的 DDS	每个电机数据组中的驱动数据组的数量
DB31, ... DBX21.x	请求接口
DB31, ... DBX93.x	显示接口
DB31, ... DBX130.x	格式接口

图 2-11　　电机数据组/驱动数据组切换的原理

2.3.6.7　　前提条件

“最后”一个电机数据组的驱动数据组的数量

“最后”一个电机数据组带有最高编号或下标。

通常，在驱动中会为每个电机数据组创建相同数量的驱动数据组（“每个 MDS 中的 DDS”数）。 但允许“最后”一个电机数据组有所不同，可以设置任意数量的驱动数据组：

1 ≤ a ≤（“每个 MDS 中的 DDS”数）

示例

现在要设置 4 个电机数据组、每个电机数据组有 8 个驱动数据组（每个 MDS 中的 DDS）。这与图 2-11 电机数据组/驱动数据组切换的原理 (页 81)中所列出的编号为 22 的数据组组合一致：

- 电机数据组：MDS[0], MDS[1], ... MDS[3]（“最后”一个电机数据组）
- 每个电机数据组中的驱动数据组：DDS[0] ... DDS[7]

因此，各个电机数据组中的驱动数据组的数量为：

电机数据组（MDS）	每个电机数据组(MDS)对应的驱动数据组(DDS)的数量
MDS[0] ... MDS[2]	8
MDS[3]	1 - 8

切换点：驱动参数组

原则上驱动参数组的切换可在任意时间执行。 在轴运行期间，特别是从转速控制器参数切换到电机转速定标时，可能会出现转矩跃变。 因此建议只在静止状态下切换驱动参数组，特别是轴处于静止时。

参见

接口一览 (页 79)

2.4 示例

2.4.1 参数组切换

参数组切换

通过参数组切换针对机床轴 X1 将位置闭环控制增益系数（K_V 系数）从 $K_V = 4.0$ 切换至 $K_V = 0.5$。

前提条件

必须通过以下机床数据来使能参数组切换：

MD35590 $MA_PARAMSET_CHANGE_ENABLE [AX1] = 1 或 2（可进行参数组切换）

根据下标 “0” 的机床数据，选择的是机床轴 X1 的第 1 个参数组。NC/PLC 接口：

DB31, ... DBX9.0 - DBX9.2 = 0（控制器参数组）

取决于参数组的机床数据

取决于参数组的机床数据是按照以下方式设置的：

机床数据	注释
MD32200 $MA_POSCTRL_GAIN [0, AX1] = 4.0	参数组 1 的 K_V 设置
MD32200 $MA_POSCTRL_GAIN [1, AX1] = 2.0	参数组 2 的 K_V 设置
MD32200 $MA_POSCTRL_GAIN [2, AX1] = 1.0	参数组 3 的 K_V 设置
MD32200 $MA_POSCTRL_GAIN [3, AX1] = 0.5	参数组 4 的 K_V 设置
MD32200 $MA_POSCTRL_GAIN [4, AX1] = 0.25	参数组 5 的 K_V 设置
MD32200 $MA_POSCTRL_GAIN [5, AX1] = 0.125	参数组 6 的 K_V 设置
MD31050 $MA_DRIVE_AX_RATIO_DENOM [0, AX1] = 3	参数组 1 的负载传动级分母
MD31050 $MA_DRIVE_AX_RATIO_DENOM [1, AX1] = 3	参数组 2 的负载传动级分母
MD31050 $MA_DRIVE_AX_RATIO_DENOM [2, AX1] = 3	参数组 3 的负载传动级分母
MD31050 $MA_DRIVE_AX_RATIO_DENOM [3, AX1] = 3	参数组 4 的负载传动级分母
MD31050 $MA_DRIVE_AX_RATIO_DENOM [4, AX1] = 3	参数组 5 的负载传动级分母
MD31050 $MA_DRIVE_AX_RATIO_DENOM [5, AX1] = 3	参数组 6 的负载传动级分母
MD31060 $MA_DRIVE_AX_RATIO_NUMERA [0, AX1] = 5	参数组 1 的负载传动级分子
MD31060 $MA_DRIVE_AX_RATIO_NUMERA [1, AX1] = 5	参数组 2 的负载传动级分子
MD31060 $MA_DRIVE_AX_RATIO_NUMERA [2, AX1] = 5	参数组 3 的负载传动级分子
MD31060 $MA_DRIVE_AX_RATIO_NUMERA [3, AX1] = 5	参数组 4 的负载传动级分子
MD31060 $MA_DRIVE_AX_RATIO_NUMERA [4, AX1] = 5	参数组 5 的负载传动级分子
MD31060 $MA_DRIVE_AX_RATIO_NUMERA [5, AX1] = 5	参数组 6 的负载传动级分子
MD35130 $MA_AX_VELO_LIMIT [0...5, AX1]	每个参数组的设置*)
MD32800 $MA_EQUIV_CURRCTRL_TIME [0..5, AX1]	每个参数组的设置*)
MD32810 $MA_EQUIV_SPEEDCTRL_TIME [0..5, AX1]	每个参数组的设置*)
MD32910 $MA_DYN_MATCH_TIME [0...5, AX1]	每个参数组的设置*)
*) 每个参数组所对应的行都需要根据句法规则单独指定。	

切换

PLC 用户程序选择机床轴 X1 的第 4 个参数组来切换位置闭环的增益系数。

- 通过 PLC 用户程序请求：
  DB31, … DBX9.0 – DBX9.2 = 3（伺服参数组）
  - 向机床轴 AX1 发送请求，请求切换至第 4 个参数组。
  - 延迟时间之后会切换参数组。
  - 参数组 4 现已激活，与带有下标 “3” 的机床数据相一致。
- NC 发出反馈：
  DB31, … DBX69.0 – DBX69.2 = 3（伺服参数组）
  - 参数组的切换由 NC 来确认/应答。

2.5 数据表

2.5.1 机床数据

2.5.1.1 显示机床数据

编号	名称：$MM_	说明
SINUMERIK Operate		
9000	LCD_CONTRAST	对比度
9001	DISPLAY_TYPE	监视器类型
9004	DISPLAY_RESOLUTION	显示精度
9006	DISPLAY_SWITCH_OFF_INTERVAL	屏幕变暗时间

2.5.1.2 NC 专用机床数据

编号	名称：$MN_	说明
10350	FASTIO_DIG_NUM_INPUTS	生效的 NC 数字量输入字节的数量
10360	FASTIO_DIG_NUM_OUTPUTS	生效的 NC 数字量输出字节的数量
10361	FASTIO_DIG_SHORT_CIRCUIT	数字量输入/输出短路
11120	LUD_EXTENDED_SCOPE	激活程序全局变量 (PUD)

编号	名称：$MN_	说明
11270	DEFAULT_VALUES_MEM_MSK	激活 功能：保存 GUD 的缺省值。
18150	MM_GUD_VALUES_MEM	为 GUD 预留存储空间

2.5.1.3 通道专用机床数据

编号	名称：$MC_	说明
21015	INVOLUTE_RADIUS_DELTA	不带参考点的 NC 启动禁用
21016	INVOLUTE_AUTO_ANGLE_LIMIT	渐开线插补时的自动角度限制
27800	TECHNOLOGY_MODE	通道工艺方式
28150	MM_NUM_VDIVAR_ELEMENTS	PLC 变量的写单元数目
28530	MM_PATH_VELO_SEGMENTS	程序段中用于限制轨迹加速度的存储空间单元数量

2.5.1.4 进给轴/主轴专用机床数据

编号	名称：$MA_	说明
30350	SIMU_AX_VDI_OUTPUT	模拟轴信号输出
33050	LUBRICATION_DIST	润滑脉冲距离
35590	PARAMSET_CHANGE_ENABLE	通过 PLC 可进行参数组设置
36060	STANDSTILL_VELO_TOL	进给轴/主轴停止时的最大速度/转速
36610	AX_EMERGENCY_STOP_TIME	故障状态下制动斜坡的持续时间
36620	SERVO_DISABLE_DELAY_TIME	伺服使能封锁延时

2.5.2 系统变量

名称	说明
$P_FUMB	空余的零件程序存储器 (Free User Memory Buffer)
$A_DBB[n]	PLC 上的数据（字节型数据）
$A_DBW[n]	PLC 上的数据（字型数据）
$A_DBD[n]	PLC 上的数据（双字型数据）
$A_DBR[n]	PLC 上的数据（实数型数据）

2.5.3 信号

2.5.3.1 发送至 NC 的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
钥匙开关位置 0 到 3	DB10.DBX56.4 - 7	DB2600.DBX0.4 - 7

2.5.3.2 从 NC 发出的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
远程诊断生效（出现 HMI 报警）	DB10.DBX103.0	-
AT-Box 就绪	DB10.DBX103.5	-
HMI 温度限制	DB10.DBX103.6	-
HMI 电池报警	DB10.DBX103.7	-
NC 就绪	DB10.DBX104.7	-
操作面板 2：“就绪”	DB10.DBX108.1	-
MPI 上的操作面板：“就绪”	DB10.DBX108.2	-
BTSS 上的操作面板：“就绪”	DB10.DBX108.3	DB2700.DBX2.3
驱动处于循环运行中	DB10.DBX108.5	DB2700.DBX2.5
驱动就绪	DB10.DBX108.6	DB2700.DBX2.6
NC 就绪	DB10.DBX108.7	DB2700.DBX2.7
出现 NC 报警	DB10.DBX109.0	DB2700.DBX3.0

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
散热器温度报警 NCU	DB10.DBX109.5	-
气温报警	DB10.DBX109.6	DB2700.DBX3.6
NC 电池报警	DB10.DBX109.7	-

2.5.3.3 发送至操作面板的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
屏幕待机未生效	DB19.DBX0.0	-
屏幕待机生效	DB19.DBX0.1	DB1900.DBX5000.1 [1]
按键禁用	DB19.DBX0.2	DB1900.DBX5000.2
删除 Cancel 型报警（仅针对 HMI 高级版）	DB19.DBX0.3	-
删除 Recall 型报警（仅针对 HMI 高级版）	DB19.DBX0.4	-
WCS(1)/MCS(0)中的实际值	DB19.DBX0.7	DB1900.DBX5000.7
卸载零件程序	DB19.DBX13.5	-
装载零件程序	DB19.DBX13.6	-
选择零件程序	DB19.DBX13.7	DB1700.DBX1000.7
主动(0)/被动(1)文件系统	DB19.DBX14.7	-
零件程序处理：用户文件名的控制数据编号	DB19.DBB16 [2]	DB1700.DBB1001 [2]
零件程序处理： 用户列表中待传输文件的下标值	DB19.DBB17	DB1700.DBB1002
运行方式切换禁止	DB19.DBX44.0	-

[1] 对于 SINUMERIK 828D，屏幕待机仅通过 DB1900.DBX5000.1 控制：

DB1900.DBX5000.1=**0**： 屏幕待机未生效

DB1900.DBX5000.1=**1**： 屏幕待机生效

DB1900.DBX5000.0 对待机控制无影响。

[2] 位 7：总是为 1

2.5.3.4 从操作面板发出的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
屏幕待机生效	DB19.DBX20.1	-
操作界面：“模拟”功能生效	DB19.DBX20.6	DB1900.DBX0.6
MCS/WCS 切换	DB19.DBX20.7	DB1900.DBX0.7
零件程序管理状态：Error	DB19.DBX26.0	DB1700.DBX2000.2
零件程序管理状态：正常	DB19.DBX26.1	DB1700.DBX2000.1
零件程序管理状态：激活	DB19.DBX26.3	DB1700.DBX2000.3
零件程序管理状态：卸载	DB19.DBX26.5	DB1700.DBX2000.5
零件程序管理状态：装载	DB19.DBX26.6	DB1700.DBX2000.6
零件程序管理状态：选择	DB19.DBX26.7	DB1700.DBX2000.7
FC9:启动“JOG 方式下的测量”	DB19.DBX42.0	-
FC9 输出：调节型	DB19.DBX45.0	-
FC9 输出：Done	DB19.DBX45.1	-
FC9 输出：Error	DB19.DBX45.2	-
FC9 输出：StartErr	DB19.DBX45.3	-

2.5.3.5 发送至通道的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
删除剩余行程（通道专用）	DB21, ...DBX6.2	DB320x.DBX6.2

2.5.3.6 来自通道的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
出现通道专用 NC 报警	DB21, ...DBX36.6	DB330x.DBX4.6
出现会停止加工的 NC 报警	DB21, … .DBX36.7	DB330x.DBX4.7
溢出转存生效	DB21, ...DBX318.7	-

2.5.3.7 发送至进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
进给轴/主轴禁止	DB31, ...DBX1.3	DB380x.DBX1.3
跟踪运行	DB31, ...DBX1.4	DB380x.DBX1.4
位置测量系统 1	DB31, ...DBX1.5	DB380x.DBX1.5
位置测量系统 2	DB31, ...DBX1.6	DB380x.DBX1.6
伺服使能	DB31, ...DBX2.1	DB380x.DBX2.1
删除剩余行程（轴专用）/主轴复位	DB31, ...DBX2.2	DB380x.DBX2.2
电机数据组/驱动数据组： 请求接口	DB31, ...DBX21.0 - 4	DB380x.DBX4001.0 - 4
电机选择成功	DB31, ...DBX21.5	DB380x.DBX4001.5
转速控制器积分器禁止	DB31, ...DBX21.6	DB380x.DBX4001.6
脉冲使能	DB31, ...DBX21.7	DB380x.DBX4001.7

2.5.3.8 来自进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
已回参考点/已同步，编码器 1/2	DB31, ...DBX60.4 / 5	DB390x.DBX0.4 / 5
运行指令 -/+	DB31, ...DBX64.6 / 7	DB390x.DBX4.6 / 7
跟踪生效	DB31, ...DBX61.3	DB390x.DBX1.3
进给轴/主轴停止（$n < n_{min}$）	DB31, ...DBX61.4	DB390x.DBX1.4
位置控制器生效	DB31, ...DBX61.5	DB390x.DBX1.5
转速控制器生效	DB31, ...DBX61.6	DB390x.DBX1.6
电流控制器生效	DB31, ...DBX61.7	DB390x.DBX1.7
润滑脉冲	DB31, ...DBX76.0	DB390x.DBX1002.0
电机数据组/驱动数据组： 显示接口	DB31, ...DBX93.0 - 4	DB390x.DBX4001.0 - 4
驱动就绪	DB31, ...DBX93.5	DB390x.DBX4001.5
转速控制器的积分器被禁用	DB31, ...DBX93.6	DB390x.DBX4001.6
脉冲使能	DB31, ...DBX93.7	DB390x.DBX4001.7
电机温度预警	DB31, ...DBX94.0	DB390x.DBX4002.0
散热器温度预警	DB31, ...DBX94.1	DB390x.DBX4002.1
加速过程结束	DB31, ...DBX94.2	DB390x.DBX4002.2

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
$\|M_d\| < M_{dx}$	DB31, ...DBX94.3	DB390x.DBX4002.3
$\|n_{实际}\| < n_{最小}$	DB31, ...DBX94.4	DB390x.DBX4002.4
$\|n_{实际}\| < n_x$	DB31, ...DBX94.5	DB390x.DBX4002.5
$n_{实际}=n_{设定}$	DB31, ...DBX94.6	DB390x.DBX4002.6
电机数据组/驱动数据组： 格式接口	DB31, ...DBX130.0 - 4	DB390x.DBX4008.0 - 4
电机数据组/驱动数据组： 格式有效	DB31, ... DBX130.7	DB390x.DBX4008.7

A3:轴监控

3

3.1 轮廓监控

3.1.1 轮廓误差

轮廓误差是由位置环中的信号失真引起的。

信号失真可分为线性失真和非线性失真。

线性信号失真

线性信号失真的产生原因有：

- 转速控制器或位置控制器没有达到最佳设置
- 参与轨迹运行的轴的 K_V 系数不同
  当两个线性插补轴的 K_V 系数相同时，实际位置会沿着相同的轨迹跟随设定位置，只是时间上有一个延迟。 当 K_V 系数不同时，设定轨迹与实际轨迹之间会产生一个平行偏移。
- 进给驱动的动态响应不同
  不同的驱动动态响应会导致轨迹偏差，尤其是轮廓变化处。 两个进给驱动动态响应的不同会使圆弧变为椭圆形。

非线性信号失真

非线性信号失真的产生原因有：

- 电流限幅在加工区域内生效
- 转速设定值限幅生效
- 在位置环内部和/或外部存在反向死区
  在沿圆弧运行时，轮廓误差主要是由于反向死区和摩擦产生。
  在沿直线运行时，轮廓误差由位置环外部的反向死区导致，例如：由倾斜的铣削主轴导致。 这会导致实际轨迹与设定轨迹之间出现平行偏移。 直线的斜率越低，该偏移就越大。
- 滑块导轨的非线性摩擦性能

3.1.2 跟随误差监控

功能

在控制技术领域，机床轴运行时始终会产生某个跟随误差，即设定位置与实际位置之间的偏差。

产生的跟随误差取决于：

- 位置环增益
  MD32200 $MA_POSCTRL_GAIN（KV 系数）
- 最大加速度
  MD32300 $MA_MAX_AX_ACCEL（最大轴加速度）
- 最大速度
  MD32000 $MA_MAX_AX_VELO（最大轴速度）
- 前馈控制激活时：
  被控对象和参数的精度：
  MD32610 $MA_VELO_FFW_WEIGHT（速度前馈控制的系数）
  MD32800 $MA_EQUIV_CURRCTRL_TIME（用于前馈控制的电流环等效时间常数）
  MD32810 $MA_EQUIV_SPEEDCTRL_TIME（用于前馈控制的转速环等效时间常数）

在机床轴运行的加速阶段中，跟随误差起初会逐渐增大。 在经过一段时间后（时间长短取决于位置环设置），理想情况下该跟随误差应保持不变。 由于外部的影响，跟随误差在加工过程中始终会出现或大或小的波动。 为了避免因跟随误差的波动而触发报警，须为跟随误差监控设置一个公差带，跟随误差可在该公差带内发生变化：

MD36400 $MA_CONTOUR_TOL（轮廓监控公差带）

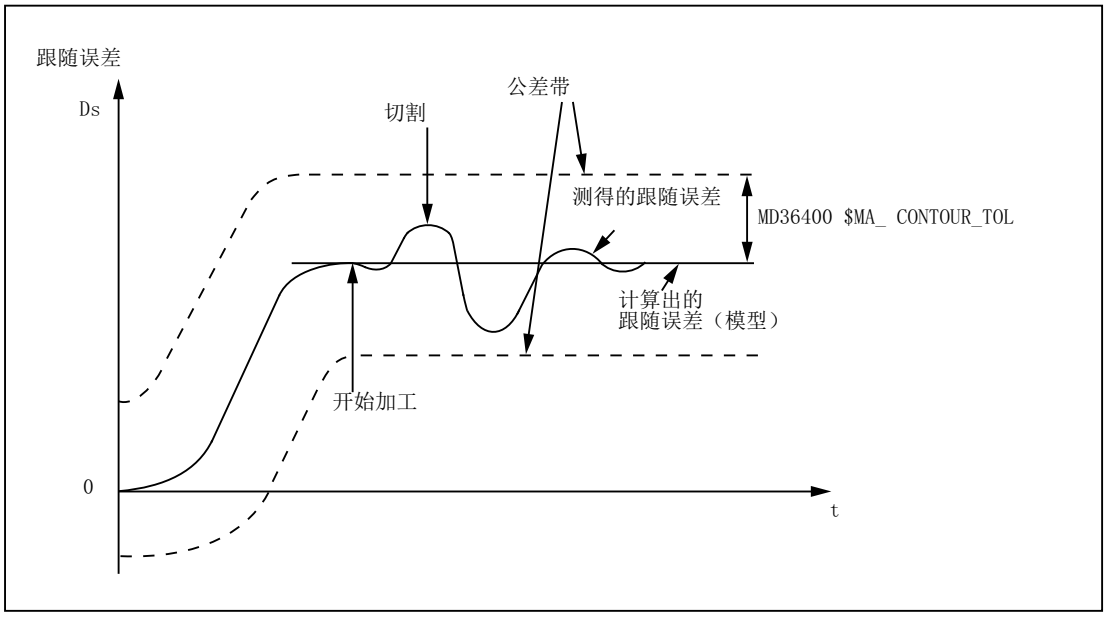


图 3-1　　跟随误差监控

有效性

跟随误差监控只有在位置闭环控制激活时及以下轴类型中才有效：

- 带/不带前馈控制的线性轴
- 带/不带前馈控制的回转轴
- 采用位置闭环控制的主轴

故障情况

一旦超出了设置的公差极限，便显示以下报警：

25050 “轴 <轴名称> 轮廓监控”

在跟踪运行中，相关的进给轴/主轴通过设置的制动斜坡减速至静止：

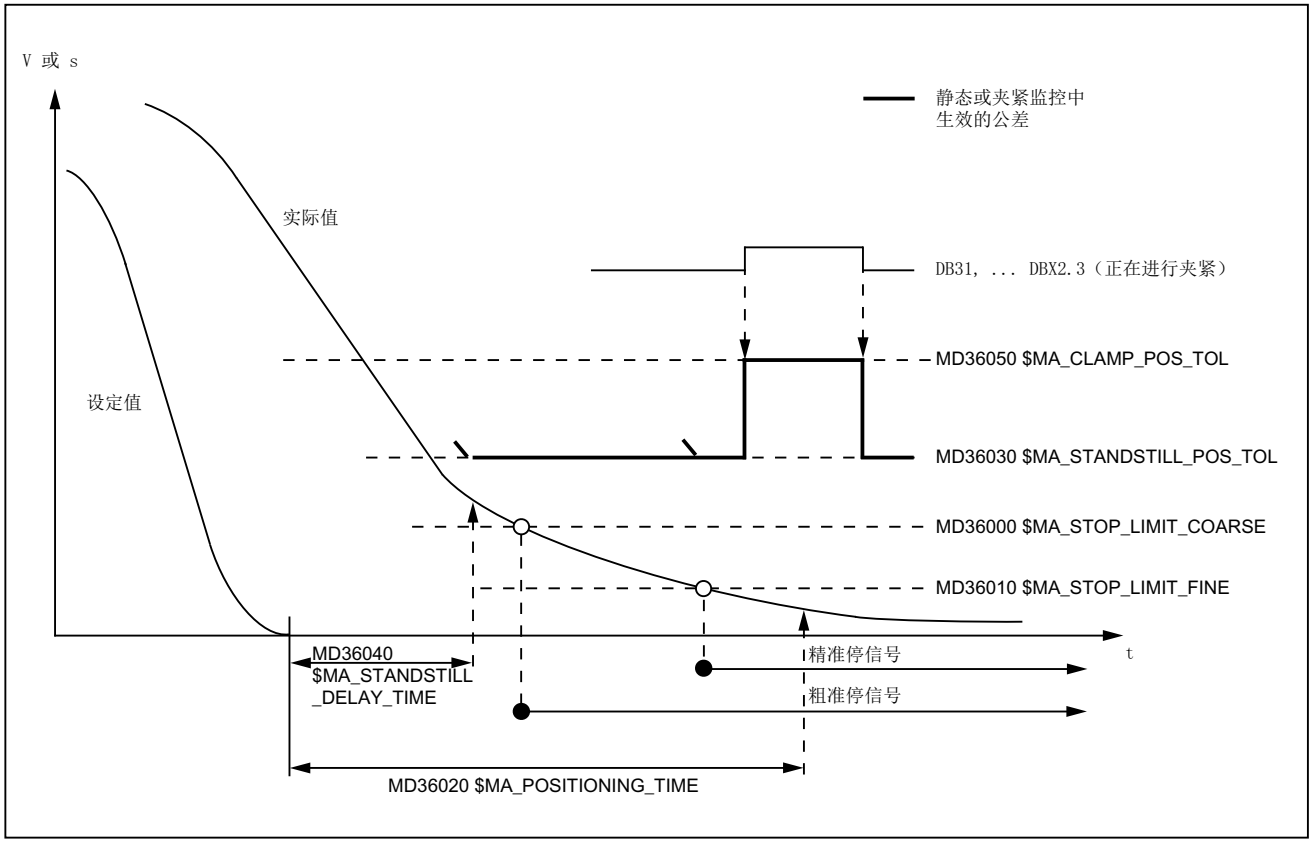
MD36610 $MA_AX_EMERGENCY_STOP_TIME
（故障状态下制动斜坡的最大持续时间）

3.2 定位、静态和夹紧监控

3.2.1 定位、静态和夹紧监控的相互关系

概述

下图展示了定位、静态和夹紧监控的相互关系：



3.2.2 定位监控

功能

在定位过程结束时：

- 设定速度 = 0 且
- DB31, ... DBX64.6/7（运行指令 -/+）= 0

检查定位监控以保证每个参与的机床轴的跟随误差在延迟时间内都小于精准停公差：

MD36010 $MA_STOP_LIMIT_FINE（精准停）

MD36020 $MA_POSITIONING_TIME（精准停延迟时间）

到达“精准停”后，系统便会关闭定位监控。

说明

精准停公差越小，定位过程持续的时间也就越长，程序段切换也就越慢。

机床数据设置的规则

MD36010 $MA_STOP_LIMIT_FINE	MD36020 $MA_POSITIONING_TIME
大	可选择相对短的时间
小	必须选择相对长的时间

MD32200 $MA_POSCTRL_GAIN（KV 系数）	MD36020 $MA_POSITIONING_TIME
小	必须选择相对长的时间
大	可选择相对短的时间

有效性

定位监控只有在位置闭环控制激活时及以下轴类型中才有效：

- 线性轴
- 回转轴
- 采用位置闭环控制的主轴

故障情况

一旦超出了设置的定位监控时间，便显示以下报警：

25080 “轴 <轴名称> 定位监控”

在跟踪运行中，相关轴通过设置的制动斜坡减速至静止：

MD36610 $MA_AX_EMERGENCY_STOP_TIME
（故障状态下制动斜坡的最大持续时间）

3.2.3 静态监控

功能

在定位过程结束时：

- 设定速度 = 0 且
- DB31, ... DBX64.6/7（运行指令 -/+）= 0

检查静态监控，以保证每个参与的机床轴的跟随误差在延迟时间内都小于静态公差：

MD36040 $MA_STANDSTILL_DELAY_TIME（静态监控延迟时间）

MD36030 $MA_STANDSTILL_POS_TOL（静态公差）

到达所需的准停状态后，定位过程结束：

DB31, ... DBX60.6/7（达到粗/精准停位置窗口）= 1

定位监控关闭并被静态监控取代。

静态监控检查是否遵循了静态公差。 只要没有发出新的运行请求，就不允许机床轴超出静态公差。

有效性

静态监控只有在位置闭环控制激活时及以下轴类型中才有效：

- 线性轴
- 回转轴
- 采用位置闭环控制的主轴

故障情况

一旦超出了延迟时间和/或静态公差，便显示以下报警：

25040 “轴 <轴名称> 静态监控”

在跟踪运行中，相关轴通过设置的制动斜坡减速至静止：

MD36610 $MA_AX_EMERGENCY_STOP_TIME
（故障状态下制动斜坡的最大持续时间）

3.2.4 和参数组相关的准停公差和静态公差

为了适应不同的工况和/或轴动态响应，例如：

- 运行状态 A：高精度、长加工时间
- 运行状态 B：低精度、短加工时间
- 传动级换档后的质量改变

位置公差：

- MD36000 $MA_STOP_LIMIT_COARSE（粗准停）
- MD36010 $MA_STOP_LIMIT_FINE（精准停）
- MD36030 $MA_STANDSTILL_POS_TOL（静态公差）

可通过同一个和参数组相关的的系数计算得出：

MD36012 $MA_STOP_LIMIT_FACTOR（粗准停/精准停和停机系数）

由于该系数共用于所有三个位置公差，因此各个值的比值保持不变。

3.2.5 夹紧监控

3.2.5.1 功能

定位过程结束后被机械夹紧的机床轴会因夹紧过程从设定位置上偏移。设置 NC/PLC 接口信号 DB31, ... DBX2.3（正在夹紧）后，系统会对夹紧公差 (MD36050 $MA_CLAMP_POS_TOL)，而不是静态公差 (MD36030 $MA_STANDSTILL_POS_TOL) 进行监控。如果超出了夹紧公差，系统会显示报警 26000 “夹紧监控”。

报警延时

如果允许在一定时间范围内超出夹紧公差，则须通过机床数据 MD36051 $MA_CLAMP_POS_TOL_TIME 设置一个报警延时。这样在超出夹紧公差时，只有在该设定时间届满后才会输出报警。如果在该时间届满前再次低于夹紧公差，则不会输出报警。下一次超出夹紧公差时，系统会重启该时间。

为了能在报警延时届满前对超出夹紧公差作出响应，系统会设置通道专用的 NC/PLC 接口信号 DB31, ... DBX102.3（超出了夹紧公差）。一旦低于夹紧公差，系统会再次复位该信号。

3.2.5.2 机床数据

夹紧公差

该机床数据中记录了比静态公差更大的夹紧公差。

MD36050 $MA_CLAMP_POS_TOL[<轴>] = <夹紧公差>

报警延时

如果允许在一定时间范围内超出夹紧公差，则须在该机床数据中输入允许的最大报警延时。

MD36051 $MA_CLAMP_POS_TOL_TIME[<轴>] = <报警延时>

这样在超出夹紧公差时，只有在该报警延时届满后才会显示报警 26000“夹紧监控”。

如果在该报警延时届满前再次低于夹紧公差，则不会输出报警。

下一次超出夹紧公差时，系统会重启该报警延时。

特殊夹紧功能

通过该机床数据逐位激活特殊夹紧功能，其用于在零件程序运行中自动松开和设置夹具：

MD36052 $MA_STOP_ON_CLAMPING[<轴>], <位> = <值>

<位>	<值>	含义
0	自动停止以松开夹具	
	0	不生效
	1	生效
1	以最优方式松开夹具	
	0	不生效
	1	生效，前提条件：位 0 == 1
2	自动停止以设置夹具	
	0	不生效
	1	生效

3.2.5.3 NC/PLC 接口信号

激活夹紧监控

通过设置 NC/PLC 接口信号激活夹紧监控：

DB31, ... DBX2.3 = 1（正在进行夹紧）

超出夹紧公差

超出夹紧公差通过 NC/PLC 接口信号显示：

DB31, ... DBX102.3 == 1（超出了夹紧公差）

如果在报警延时内超出了夹紧公差，该信号会**置位**。

如果在报警延时内低于夹紧公差或激活了跟踪运行，该信号会**复位**。

3.2.5.4 故障响应

超出夹紧公差时的故障响应：

- 显示报警 26000“夹紧监控”
- 轴通过编程的最大加速度停止：
  MD32300 $MA_MAX_AX_ACCEL
  此时会监控故障状态下制动斜坡的最大持续时间：
  MD36610 $MA_AX_EMERGENCY_STOP_TIME
- 激活跟踪运行：
  DB31, ... DBX61.3 == 1
- 信号“超出了夹紧公差”复位：
  DB31, ... DBX102.3 == 0

3.2.5.5 夹紧功能“自动停止以松开夹具”

通过夹紧功能“自动停止以松开夹具”可在连续轨迹控制运行模式下在夹紧轴的运行程序段前插入一个 NC 内部停止指令。

如果在切换程序段前置位了夹紧轴的控制器使能 (DB31, ... DBX2.1)，该停止指令则不会生效或不会中断连续轨迹控制运行。

如果在切换程序段前**未**置位夹紧轴的控制器使能，该停止指令生效。

参数设置

MD36052 $MA_STOP_ON_CLAMPING[<夹紧轴>] = 'H01'

前提条件/假设

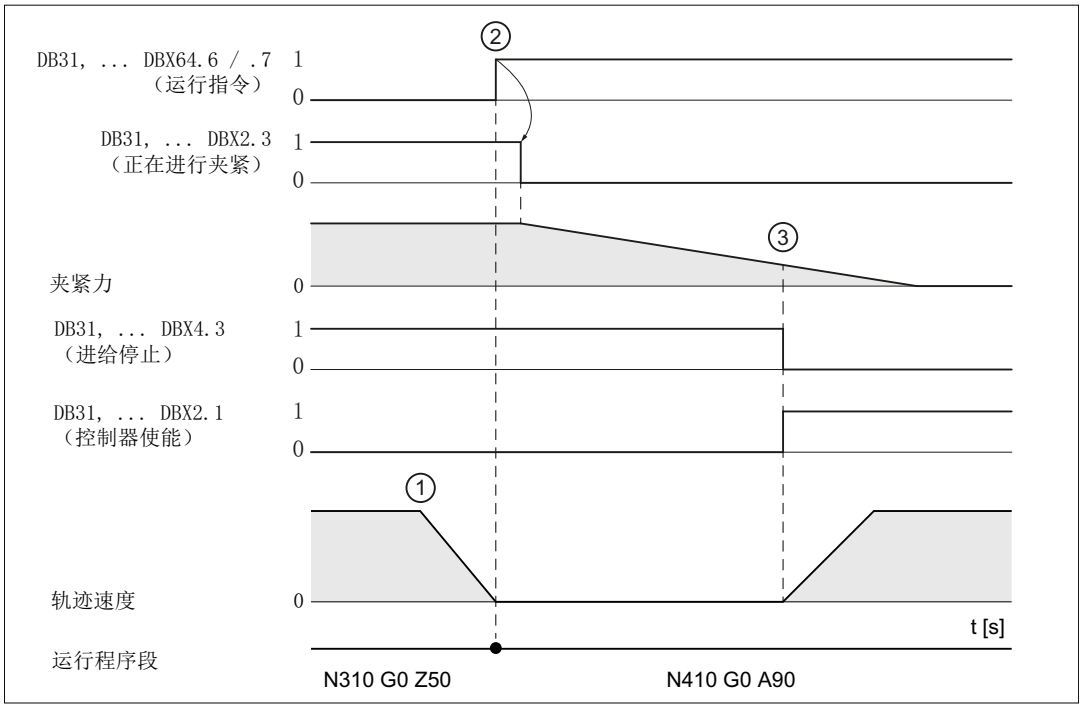
- 存在针对夹紧轴的**运行指令**（DB31, ... DBX64.6 / .7）时，夹具通过 PLC 用户程序**松开**。
- 控制器使能（DB31, ... DBX2.1）与夹紧轴的夹紧之间必须是以下关系：

控制器使能	⇒	夹紧轴
未设置	⇒	夹紧
已设置	⇒	未夹紧

示例

程序代码	注释
N100 G0 X0 Y0 Z0 A0 G90 G54 F500	；逼近初始位置
N101 G641 ADIS=.1 ADISPOS=5	；激活连续轨迹控制运行
N210 G1 X10	；运行程序段
N220 G1 X5 Y20	; "
N310 G0 Z50	；定位程序段
N410 G0 A90	; "（**夹紧轴**）
N510 G0 X100	; "
N520 G0 Z2	; "
N610 G1 Z-4	；运行程序段
N620 G1 X0 Y-20	; "

NC/PLC 接口信号的曲线图与程序段 N310 和 N410 的状态：



① NC：因自动插入的停止指令在 N310 的程序段末尾停止。

② NC → PLC：在切换程序段后设置夹紧轴的运行指令

PLC：因运行指令松开夹具。

③ PLC → NC：夹紧力完全消除。使能夹紧轴运行。

3.2.5.6 夹紧功能“在最佳时间松开夹具”

通过夹紧功能“在最佳时间松开夹具”和“自动停止以松开夹具”可在连续轨迹控制运行模式下通过预先设置夹紧轴的运行指令请求松开 NC 内部夹具。只有在运行夹紧轴前只进行了定位（G0 程序段）且未执行（G1 程序段），才可设置运行指令。

为获取夹紧轴的运行程序段参照点，运行指令在运行程序段前最多可以设置两个快速程序段 (G0)，包括内部生成的中间程序段。

激活

MD36052 $MA_STOP_ON_CLAMPING[<夹紧轴>] = 'H03'

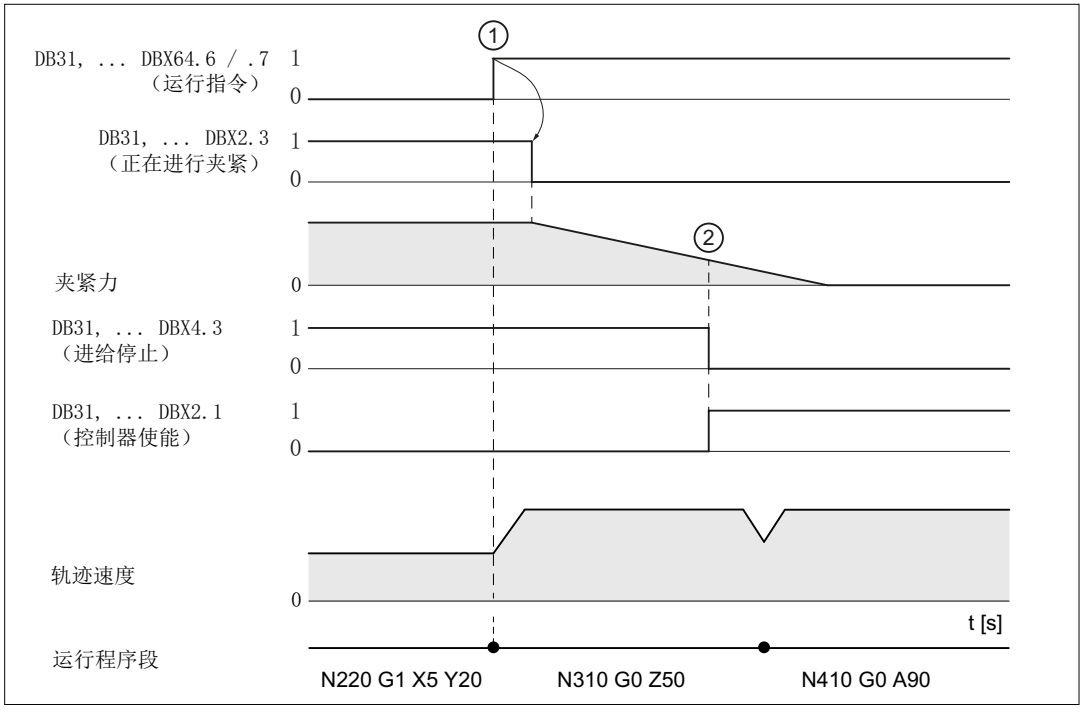
前提条件/假设

- 存在针对夹紧轴的**运行指令**（DB31, ... DBX64.6 / .7）时，夹具通过 PLC 用户程序**松开**。
- 在以快速运行（G0）运行其他轴期间，夹紧轴**不**可夹紧。

示例

程序代码	注释
N100 G0 X0 Y0 Z0 A0 G90 G54 F500	; 逼近初始位置
N101 G641 ADIS=.1 ADISPOS=5	; 激活连续轨迹控制运行
N210 G1 X10	; 加工程序段
N220 G1 X5 Y20	; "
N310 G0 Z50	; 定位程序段
N410 G0 A90	; " (**夹紧轴**)
N510 G0 X100	; "
N520 G0 Z2	; "
N610 G1 Z-4	; 加工程序段
N620 G1 X0 Y-20	; "

NC/PLC 接口信号的曲线图与程序段 N220 ~ N410 的状态：



① NC → PLC：因切换程序段设置夹紧轴的运行指令。
PLC：因运行指令松开夹具。

② PLC → NC：夹紧力完全消除。使能夹紧轴运行。

3.2.5.7 夹紧功能“自动停止以设置夹具”

某一时间要求夹紧过程。在连续路径运行模式下，需要通过编写例如 G09、G60 或辅助功能输出来设定明确的运行停止，从而在加工开始前确保可靠的夹紧。

通过夹紧功能“自动停止以设置夹具”可在连续轨迹控制运行模式下自动执行运行停止指令。若夹紧轴届时尚未夹紧，则在下一个加工程序段（无 G0 的运行程序段）**前**或 **中**进行运行停止进行夹紧以及使能其他运行的标准是：通过 PLC 用户程序设置的不等于 0% 的通道专用进给倍率（DB21, ... DBB4 ≠ 0%）

激活

MD36052 $MA_STOP_ON_CLAMPING[<夹紧轴>] = 'H04'

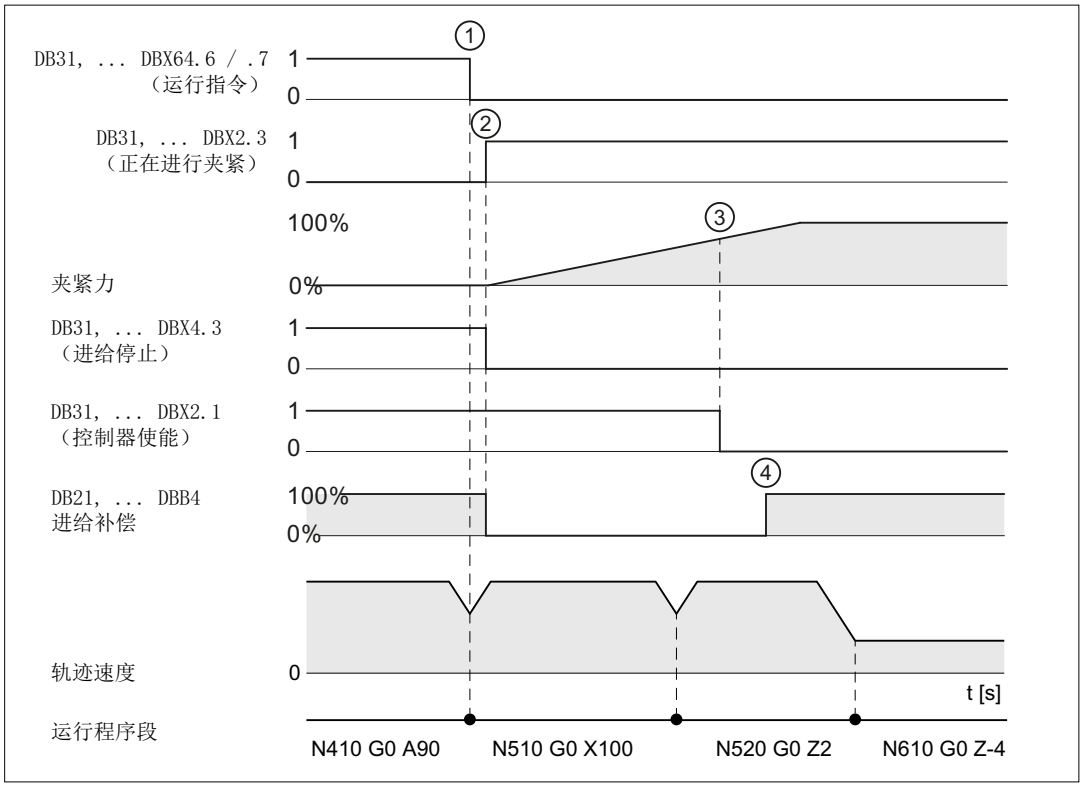
前提条件/假设

- 在**无**针对夹紧轴的运行指令（DB31, ... DBX64.6 / .7）时，夹具通过 PLC 用户程序**闭合**。
- 在以快速运行（G0）运行其他轴期间，夹紧轴**不**可夹紧。
- 在通道专用进给倍率**不等于 0%**（DB21, ... DBB4 ≠ 0%）的情况下，夹紧轴**夹紧**。

示例

程序代码	注释
N100 G0 X0 Y0 Z0 A0 G90 G54 F500	；逼近初始位置
N101 G641 ADIS=.1 ADISPOS=5	；激活连续轨迹控制运行
N210 G1 X10	；加工程序段
N220 G1 X5 Y20	; "
N310 G0 Z50	；定位程序段
N410 G0 A90	; " （**夹紧轴**）
N510 G0 X100	; "
N520 G0 Z2	; "
N610 G1 Z-4	；加工程序段
N620 G1 X0 Y-20	; "

NC/PLC 接口信号的曲线图与程序段 N410 ~ N610 的状态：



① NC → PLC：因切换程序段复位夹紧轴的运行指令

② PLC：装入夹具

③ PLC → NC：夹紧力足以复位控制器使能

④ PLC → NC：使能加工 N610，通过设置不等于 0% 的通道专用进给使能

3.2.5.8 前提条件

中断的连续轨迹控制运行

如果通过未编程运行动作的程序段（例如：输出一个 M 功能 M82 / M83）中断了连续轨迹控制运行，上述夹紧功能和“预读”功能的表现方式如下：

- **夹紧功能：“以时间最优方式松开轴夹具”**
  （MD36052 $MA_STOP_ON_CLAMPING[<轴>] = 'B011'）
  由于只能对激活了连续轨迹控制运行的程序段提前设置运行指令，因此，该功能不再有效。输出以下示例程序的程序段 N320 中的 M 功能 M82 会生成一个运行停止指令，进而中断连续轨迹控制运行。
  由于通过 N320 停止无论如何都会发生，因此无需“预先”停止在 N410 上。
- **夹紧功能：“自动停止以设置夹具”：**
  （MD36052 $MA_STOP_ON_CLAMPING[<轴>] = 'B100'）
  此功能独立于 M83 生成一个根据“进给倍率 0%”执行的停止。在第一个加工程序段之前轴就已经停止。

说明

使用不带夹具的夹紧功能

可使用以下夹紧功能（不用考虑轴的夹具）：

- “自动停止以松开夹具”：
  MD36052 $MA_STOP_ON_CLAMPING[<轴>] = 'B001'
  特性：如果**未**对<轴>设置控制器使能 (DB31, ... DBX2.1) 且是在以下其中一个程序段中运行的，轴则会停止在当前程序中的轨迹上。
- “自动停止以设置夹具”：
  MD36052 $MA_STOP_ON_CLAMPING[<轴>] = 'B100'
  特性：如果在从快速运行程序段 (G0) 过渡到运行程序段 (G1) 时通道专用的进给倍率 (DB21, ... DBB4) = 0%，轴则会停止在当前程序段中的轨迹上。

这两个功能都可以确保在连续轨迹控制运行模式下，轨迹运行在相关程序段开始前就已经停止，而不是在程序段运行过程中停止。

表格 3-1　示例程序：中断的连续轨迹控制运行

程序代码	注释
N100 G0 X0 Y0 Z0 A0 G90 G54 F500	; 逼近初始位置
N101 G641 ADIS=.1 ADISPOS=5	; 激活连续轨迹控制运行
N210 G1 X10	; 运行程序段
N220 G1 X5 Y20	; "
N310 G0 Z50	; 快速程序段
N320 M82	; **中断**连续轨迹控制运行
N410 G0 A90	; 快速程序段
N420 M83	; **中断**连续轨迹控制运行
N510 G0 X100	; 快速程序段
N520 G0 Z2	; "

程序代码	注释
N610 G1 Z-4	；运行程序段
N620 G1 X0 Y-20	; "

程序段切换标准：夹紧公差

激活夹紧监控后 (DB31, ... DBX2.3)，运行程序段中的程序段切换标准（轴在程序段结束时停止）就不再作为准停条件而是设置的夹紧公差：

MD36050 $MA_CLAMP_POS_TOL（接口信号“夹紧有效”的夹紧公差）

松开夹具时的特性

如果夹紧轴在夹紧过程中被移动，在松开夹具和设置控制器使能 (DB31, ... DBX2.1) 后，NC 会使其再次回到设定位置上。重新定位取决于夹紧过程期间是否激活了轴的“跟踪运行”：

- DB31, ... DBX1.4 == 0（跟踪运行未激活）⇒ 通过位置控制器运行
- DB31, ... DBX1.4 == 1（跟踪运行激活）⇒ 插补式运行

说明

以下数据可由 PLC 用户程序来分析从而作为激活跟踪运行 (DB31, ... DBX1.4) 的标准：

- DB31, ... DBX60.6/.7（采用粗/精准停到达位置）
- 夹紧轴的实际位置

跟踪运行

夹紧监控在“跟踪运行”下是未激活的

DB31, ... DBX1.4 == 1（跟踪运行）。

3.3 转速设定值监控

功能

转速设定值由以下部分组成：

- 位置控制器的转速设定值
- 前馈控制的转速设定值部分（只在激活的前馈控制中）
- 偏差补偿（只在带模拟设定值接口的驱动中）

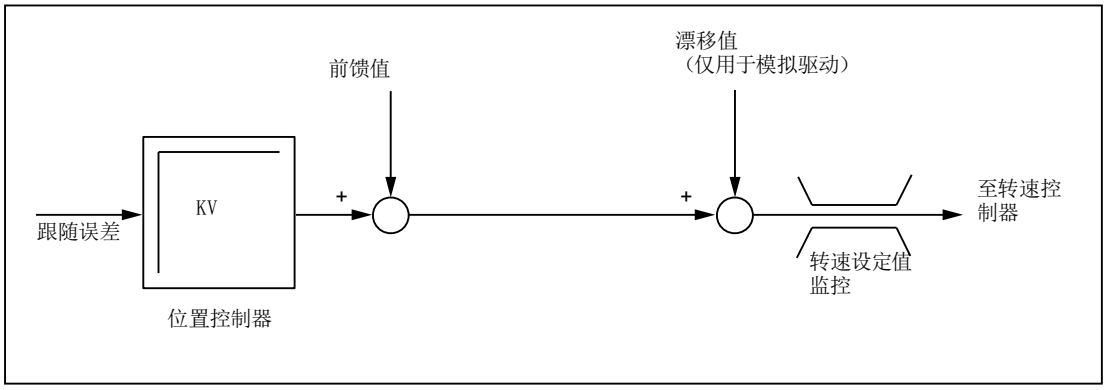


图 3-2 转速设定值计算

转速设定值监控通过限制调节值或输出值（模拟设定值接口上的 10 V 电压或数字量驱动上的额定转速）来确保不超过驱动的物理限制。

MD36210 $MA_CTRLOUT_LIMIT（最大转速设定值）

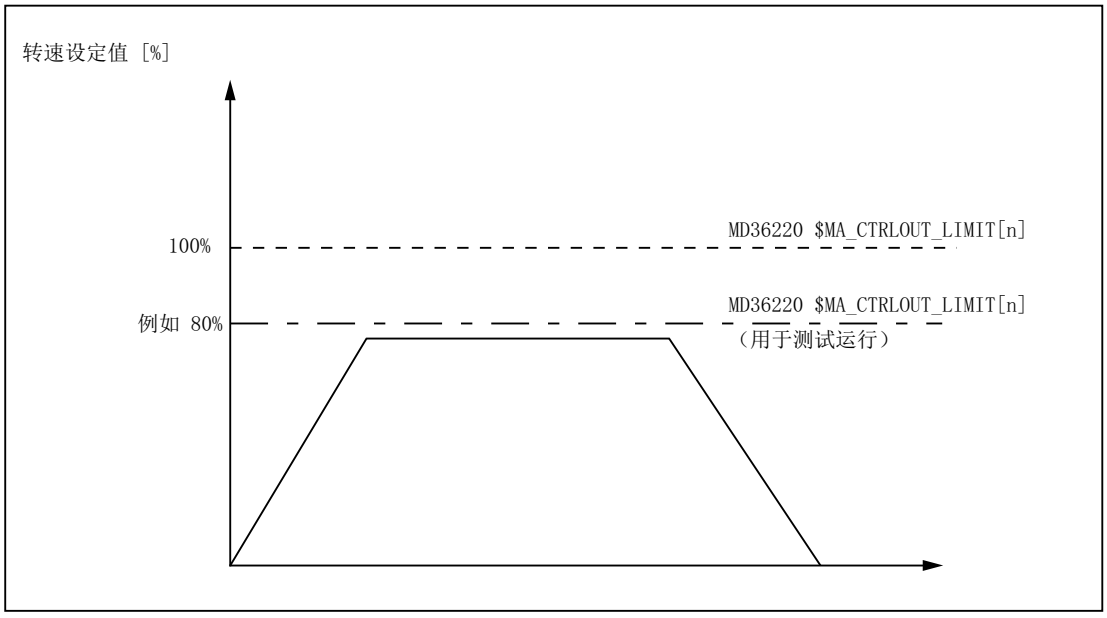


图 3-3　　转速设定值限制

转速设定值监控的延迟时间

可通过设置一个延迟时间来确保转速限制在任何情况下都不会导致故障响应：

MD36220 $MA_CTRLOUT_LIMIT_TIME（转速设定值监控的延迟时间）

只有在要求转速限制大于设置的时间时才会触发相应的故障响应。

有效性

转速设定值监控只可用于处于位置闭环控制状态的进给轴且无法将其关闭。

故障情况

一旦超出了设置的延迟时间，便显示以下报警：

25060 “进给轴 <轴名称> 转速设定值限制”

在跟踪运行中，相关轴通过设置的制动斜坡减速至静止：

MD36610 $MA_AX_EMERGENCY_STOP_TIME
（故障状态下制动斜坡的最大持续时间）

说明

达到转速设定值监控时，轴的位置环会因限制而变为非线性。 如果轴参与到了轮廓创建中，就会产生轮廓误差。

3.4 实际速度监控

功能

实际速度监控检查的是进给轴/主轴的当前实际速度未超过设置的阈值。

MD36200 $MA_AX_VELO_LIMIT（速度监控阈值）

该阈值应位于设置的最大速度的 10-15% 以上：

- 进给轴：
  MD32000 $MA_ MAX_AX_VELO（最大进给轴速度）
- 主轴：
  MD35110 $MA_GEAR_STEP_MAX_VELO_LIMIT[n]（齿轮级的最大转速）

采用该设置后，通常情况下转速就不会超过速度监控的阈值（例外情况：驱动故障）。

激活

一旦主动测量系统提供了有效的实际值，实际速度监控就会生效（未超过编码器极限频率）。

有效性

实际速度监控只有在位置闭环控制激活时及以下轴类型中才有效：

- 线性轴
- 回转轴
- 开环控制和采用位置闭环控制的主轴

故障情况

一旦超出阈值，便显示以下报警：

25030 “进给轴 <轴名称> 实际速度报警极限”

在跟踪运行中，相关轴通过设置的制动斜坡减速至静止：

MD36610 $MA_AX_EMERGENCY_STOP_TIME
（故障状态下制动斜坡的最大持续时间）

3.5 测量系统监控

NC 无法直接访问测量系统硬件，所以测量系统监控主要是由驱动软件执行的。

驱动中的监控功能

- 硬件故障监控（例如：测量系统故障、断相）
- 零脉冲监控

文档：
SINAMICS S120 驱动功能

在驱动中执行的测量系统监控的结果以 NC 报警（报警 25000 及以后）或 NC 响应（例如：回参考点或快速测量中断）的形式输出。NC 的准确特性取决于以下机床数据中的设置：

MD36310 $MA_ENC_ZERO_MONITORING

值	含义	
= 0	硬件故障监控：	打开 在主动测量系统中检测出硬件故障时，系统会显示上电报警 25000： “进给轴 <轴名称> 主动编码器的硬件故障” 在跟踪运行中，相关轴通过设置的制动斜坡减速至静止： MD36610 $MA_AX_EMERGENCY_STOP_TIME（故障状态下制动斜坡的最大持续时间） 在被动测量系统中检测出硬件故障时，系统会显示报警 25001： “进给轴 <轴名称> 被动编码器的硬件故障” 不再有其他的报警响应。
	零脉冲监控：	关闭 报警 25020 和 25021（见下）被抑制。

值	含义	
= 100	即使无零脉冲监控以及跳过全部的编码器监控（即：除了报警 25020 (25021)），报警 25000 (25001) 和 25010 (25011) 也会被抑制。	
> 0 但 < 100	硬件故障监控：	打开（见上）
	零脉冲监控：	打开 在主动测量系统中对零脉冲监控进行响应时，系统会显示报警 25020： “进给轴 <轴名称> 主动编码器的零脉冲监控” 在跟踪运行中，相关轴通过设置的制动斜坡减速至静止： MD36610 $MA_AX_EMERGENCY_STOP_TIME（故障状态下制动斜坡的最大持续时间） 在被动测量系统中对零脉冲监控进行响应时，系统会显示报警 25021： “进给轴 <轴名称> 被动编码器的零脉冲监控” 不再有其他的报警响应。
> 100	硬件故障监控：	ON，带有简化的故障信息： 上电报警 25000 由复位报警 25010 代替，而复位报警 25001 由 Cancel 型报警 25011 代替。
	零脉冲监控：	打开（见上）

有关报警的详细信息请参见：

文档：
诊断手册

说明
出现硬件故障时，机床轴的回参考点状态会复位：
DB31, ... DBX60.4/5（已回参考点/已同步 1/2）= 0

NC 中的监控功能

- 编码器极限频率监控
- 绝对值编码器的合理性检查

3.5.1 编码器极限频率监控

功能

NC 侧的编码器极限频率监控是以驱动的选型信息和报文信息为基础的。 它负责检查编码器频率未超过设置的编码器极限频率：

MD36300 $MA_ENC_FREQ_LIMIT（编码器极限频率）

编码器极限频率监控始终针对的是在 NC/PLC 接口中所选择的主动测量系统：

DB31, ... DBX1.5 / 1.6（位置测量系统 1/2）

有效性

编码器极限频率监控对以下轴有效：

- 线性轴
- 回转轴
- 开环控制和采用位置闭环控制的主轴

故障情况

一旦超出了编码器极限频率：

- 向 PLC 反馈信息：
  DB31, ... DBX60.2 或 60.3 = 1（超出了编码器极限频率 1 或 2）
- 主轴
  主轴不会停止，而是在转速控制方式下继续旋转。
  如果主轴转速继续降低，直至编码器频率再次低于编码器频率极限时，系统会自动重新同步主轴的实际值系统。
- 轴
  显示以下报警：
  21610 “超出了通道 <通道号> 进给轴 <轴名称> 编码器 <编码器编号> -频率”
  在跟踪运行中，相关轴通过设置的制动斜坡减速至静止：
  MD36610 $MA_AX_EMERGENCY_STOP_TIME
  （故障状态下制动斜坡的最大持续时间）

说明

超出编码器极限频率后，位置闭环控制的机床轴必须重新回参考点（参见章节“R1：回参考点 (页 1349)”）。

3.5.2 绝对值编码器的合理性检查

功能

在绝对值编码器 (MD30240 $MA_ENC_TYPE = 4) 上，NC 会使用该绝对值编码器提供的绝对值来检查实际值的合理性。

为此，NC 会采用软件技术，在位置控制循环中循环对比两个位置值，一个是随编码器增量信息引入的位置值，一个是直接从编码器绝对信息和增量信息中重新生成的位置值，NC 会检查两者之间的位置差值是否超过了允许的偏差：

MD36310 $MA_ENC_ZERO_MONITORING（绝对和增量编码器信号之间允许的偏差是以 1/2 个原始编码器刻线为单位的）

说明

绝对值编码器上的合理性检查主要是用于算出因绝对信号受到污染或绝对值传输故障而产生的偏差。但它无法识别出增量信号上细微的失真（干扰脉冲、脉冲故障），只有在偏差达到毫米级时，合理性检查才会发出响应。因此它只可用于辅助监控，旨在优先诊断出绝对位置的故障。

说明

旋转绝对值编码器

由于合理性检查是在旋转的绝对值编码器上设置的，在设置模数范围 (MD34220 $MA_ENC_ABS_TURNS_MODULO) 时必须考虑 SINAMICS 参数 p0979。

说明

升级 NC 软件

如果绝对值编码器上的合理性检查是激活的 (MD36310 > 0)，那么在升级 NC 软件时就必须检查并提高之前的 MD36310 设置值。

零脉冲诊断

在调试绝对值编码器时必须计算出合理性检查允许的差值。可通过以下机床数据进行设置：

MD36312 $MA_ENC_ABS_ZEROMON_WARNING（零脉冲监控报警阈值）

值	含义
0	无零脉冲诊断
> 0	绝对和增量编码器信号之间允许的偏差是以 1/2 个原始编码器刻线为单位的

开机调试时的步骤：

1. 取消激活零脉冲监控：
   MD36310 $MA_ENC_ZERO_MONITORING = 0
2. 激活零脉冲诊断：
   MD36312 $MA_ENC_ABS_ZEROMON_WARNING = 1
3. 移动轴并观察系统变量 $VA_ENC_ZERO_MON_ERR_CNT（识别出的极限值超出的数量）。
4. 如果 $VA_ENC_ZERO_MON_ERR_CNT ≠ 0:
   提高 MD36312 值并重复步骤 3。
5. 如果 $VA_ENC_ZERO_MON_ERR_CNT = 0（经过一个较长的时间段！）：
   找到了 MD36310 的正确值！将 MD36312 中的值接收到 MD36310 中，然后将 MD36312 设置为“0”。

说明

机床的刚度（最理想的情况是尽可能小的负载质量/惯性矩）与控制器的设置决定了调整间隙不同程度的“摆动”。为此必须考虑到 MD36310 中机床专用的极限值输入。

故障情况

报警 25020

在 **主动** 测量系统中对合理性检查进行响应时，系统会显示报警 25020：

“进给轴 <轴名称> 主动编码器的零脉冲监控”

在跟踪运行中，相关轴通过设置的制动斜坡减速至静止：

MD36610 $MA_AX_EMERGENCY_STOP_TIME
（故障状态下制动斜坡的最大持续时间）

报警 25021

在 **被动** 测量系统中对合理性检查进行响应时，系统会显示报警 25021：

“进给轴 <轴名称> 被动编码器的零脉冲监控”

不再有其他的报警响应。

说明

在故障情况下，对绝对值编码器的调节丢失且轴也不再回参考点。必须重新调节绝对编码器（参见章节“使用绝对值编码器回参考点 (页 1381)”）。

说明

增量信号中无法通过振幅监控来采集的故障会导致毫米范围内的位置偏差。该偏差取决于光栅刻线/刻线数以及出现故障时轴的运行速度。

完整的位置监控只有通过冗余度，即通过与一个不相干的第二测量系统进行比较才能实现。

3.5.3 用户自定义的故障响应

用户自定义的零脉冲监控

可在绝对测量系统 (MD30240 $MA_ENC_TYPE = 4) 中借助用户自定义的系统变量来调整零脉冲监控的标准报警和响应。 用户可以针对各同步动作或 OEM 应用来设计自己的监控方式，并利用所有该应用下相关的响应方式，比如：

- 取消报警
- 使用循环（例如：运行至换刀位置）
- ...

示例：

用户可以调整报警和响应，比如：轴停止时可能会导致昂贵的工件被损坏，因此会一直保持加工，直到使用相应的同步指令来评估工件的加工质量。

有效性

用户自定义的监控可以和标准的零脉冲监控同时或单独生效，这取决于以下机床数据中的设置：

MD36310 $MA_ENC_ZERO_MONITORING

值	含义
0	如果只是要实现用户自定义的监控，就必须取消标准的零脉冲监控： MD36310 = 0 且 MD36312 = 0
> 0	用户自定义的监控和标准零脉冲监控同时生效。
100	所有的编码器监控都已关闭。

当两个监控都生效时 (MD36310 > 0)，就可以实现**级联监控**。

示例：

当值位于 MD36310 中给定的阈值以下时，用户自定义监控会发出一个预警，只有在超出阈值时，标准零脉冲监控才会发现故障并关闭系统。

系统变量

可使用以下系统变量来实现用户自定义的故障响应：

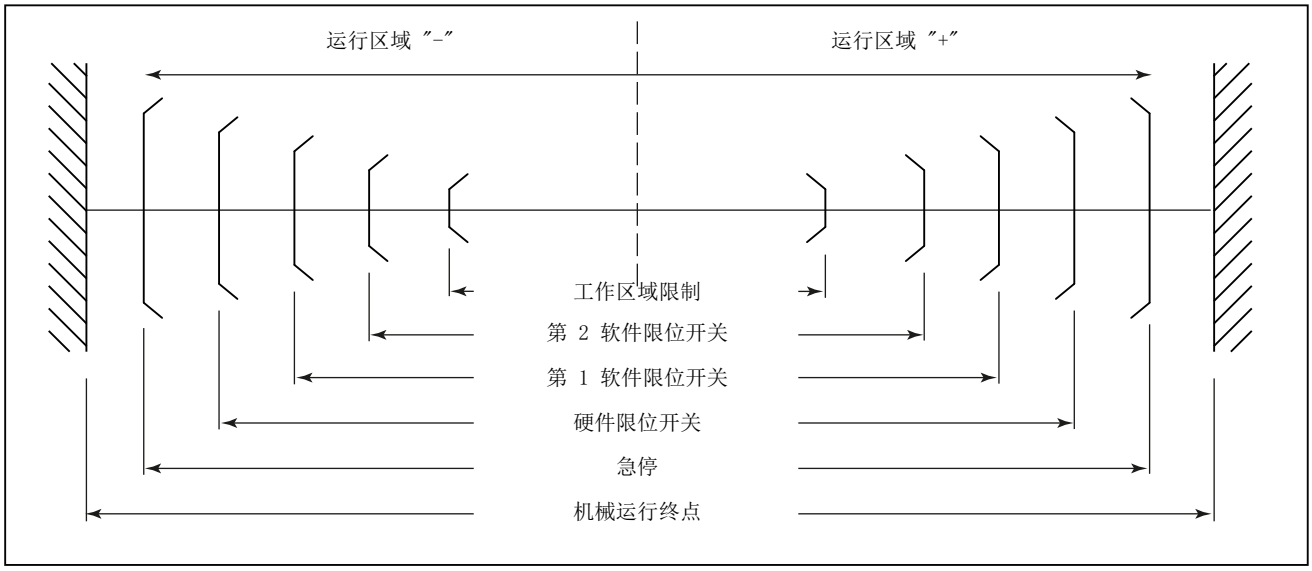
系统变量	含义
$VA_ENC_ZERO_MON_ERR_CNT[<n>,<轴>]	识别出的超限次数 包含在比较绝对编码器信号与增量编码器信号时识别出的超限次数。 在以下情况时值复位为 0： ● 上电 ● 选择或撤销驻留 复位操作不会导致该值复位。
$VA_ABSOLUTE_ENC_DELTA_INIT[<n>,<轴>]	绝对编码器上的起始差值 包含在静态 NC 存储器最后一个缓冲的绝对位置与当前的绝对位置之间的起始差值。 差值的格式：内部增量数（参见 MD10200 $MN_INT_INCR_PER_MM 或 MD10210 $MN_INT_INCR_PER_DEG） 在以下情况时值会更新： ● 上电 ● 热启动 ● 撤销驻留 ● 回到编码器极限频率以下 复位操作不会导致该值复位。

<n>： 编码器编号

<轴>：轴名称

3.6 限位开关监控

限位及限位开关监控一览：



3.6.1 硬件限位开关

功能

硬件限位开关通常被安装在机床轴行程的末端。它用于防止在机床轴还未回参考点时就意外驶出了最大行程。

如果触发了硬件限位开关，由机床制造商创建的 PLC 用户程序便会设置相应的接口信号：

DB31, ... DBX12.0/1 = 1（硬件限位开关 -/+）

参数设置

可通过以下机床数据来设置达到硬件限位开关时机床轴的制动方式：

MD36600 $MA_BRAKE_MODE_CHOICE（硬件限位开关的制动方式）

值	含义
0	使用设置的轴加速度制动
1	快速停止（设定速度 = 0）

有效性

控制器起动后，硬件限位开关监控在所有的运行模式中都有效。

影响

达到硬件限位开关时：

- 报警 21614“通道<通道编号>轴<轴名称>硬件限位开关<方向>”
- 机床轴按照设置的制动方式减速至停止。
- 如果进给轴/主轴和其它的进给轴/主轴插补，这些轴会根据设置的制动方式减速至停止。
- 相关机床轴特定方向上的移动键被禁用。

3.6.2 软件限位开关

功能

软件限位开关用于限制机床轴的运行范围。每个机床轴在每个运行方向上可以使用两个（第 1 个和第 2 个）软件限位开关：

MD36100 POS_LIMIT_MINUS（第 1 个负向软件限位开关）

MD36110 POS_LIMIT_PLUS（第 1 个正向软件限位开关）

MD36120 POS_LIMIT_MINUS2（第 2 个负向软件限位开关）

MD36130 POS_LIMIT_PLUS2（第 2 个正向软件限位开关）

标准情况下是第 1 个软件限位开关生效。通过 PLC 用户程序可激活某方向上的第 2 个软件限位开关：

DB31, ... DBX12.2 / 12.3（第 2 个负向/正向软件限位开关）

有效性

软件限位开关：

- 在机床轴回参考点后立即生效。
- 在所有运行模式中有效。

前提条件

- 软件限位开关以机床坐标系为参考。
- 软件限位开关必须位于硬件限位开关范围内。
- 机床轴可一直运行到生效的软件限位开关的位置。
- PRESET（预设）
  使用预设功能后，软件限位开关就不再生效。机床轴必须首先重新回参考点。
- 循环旋转的回转轴
  循环旋转的回转轴上没有生效的软件限位开关监控：
  MD30310 $MA_ROT_IS_MODULO == 1（回转轴和主轴的模数转换）
  例外：回转轴

影响

自动运行模式 (AUTOMATIK, MDA)

- 无坐标转换，无叠加运动，未改变的软件限位开关：
  如果一个零件程序段中所包含的运动指令有可能导致驶过软件限位开关，则不启动该零件程序段。
- 带坐标转换：
  不同的坐标转换方式产生不同的响应：
  - 方式如上。
    或者
  - 如果一个零件程序段中所包含的运动指令有可能导致驶过软件限位开关，则仍启动该零件程序段。相关的机床轴停止在生效的软件限位开关处。其他参与该运动的机床轴减速至停止。轴在此期间离开编程的轮廓。
- 带叠加运动
  如果一个零件程序段中所包含的运动指令有可能导致驶过软件限位开关，则仍启动该零件程序段。正在执行叠加运动或已经执行叠加运动的机床轴会停止在各个生效的软件限位开关处。其他参与该运动的机床轴减速至停止。轴在此期间离开编程的轮廓。

手动运行模式

- JOG，无坐标转换
  机床轴停止在软件限位开关位置处。
- JOG，带坐标转换
  机床轴停止在软件限位开关位置处。其他参与该运动的机床轴减速至停止。轴在此期间离开预设的轨迹。

简介

- 切换软件限位开关（第 1 个 ↔ 第 2 个软件限位开关）
  如果切换后机床轴的实际位置位于软件限位开关后，那么机床轴会以允许的最大加速度停止。
- 在 JOG 运行模式下驶过软件限位开关
  如果已经到达了软件限位开关的位置并重新按下移动键继续在该方向上运行，系统会显示报警且轴不再继续运行：
  报警 10621“通道<通道编号>轴<轴名称>停止在软件限位开关<方向>”

3.7 工作区域限制

3.7.1 简介

功能

通过“工作区域限制”功能可将通道中的几何轴和辅助轴的运行范围限制在允许的工作区域内。 该功能在 AUTO 模式和 JOG 模式下都可以监控工作区域。

有以下方式可供选择：

- 基准坐标系中的工作区域限制 (BCS)
  运行范围限制的指定是以基准坐标系为参考的。
- 工件坐标系 (WCS) 或可设置的零点坐标系 (AZS) 中的工作区域限制
  运行范围限制的指定是以工件坐标系或可设置的零点坐标系为参考的。

两种监控方式相互独立存在。 如果两种方式同时生效，能够在某方向上最大程度地限制轴运行范围的监控方式生效。

刀具上的参考点

对刀具数据（刀具长度和刀具半径）的参考以及在监控工作区域限制时刀具上的参考点取决于通道中的坐标转换状态：

- **坐标转换未生效**
  在无坐标转换的情况下，会监控刀尖 P 的位置，即：在监控时自动参考刀具长度。
  刀具半径参考必须单独激活：
  MD21020 $MC_WORKAREA_WITH_TOOL_RADIUS（在工作区域限制中参考刀具半径）
- **坐标转换生效**
  在特定的坐标转换中，工作区域限制的监控方式和无坐标转换的情况有所不同：
  - 刀具长度是坐标转换的一个组成部分
    ($MC_TRAFO_INCLUDES_TOOL_X = TRUE):
    在这种情况下并不考虑刀具长度，即：监控是以刀架参考点为参照的。
  - 带定向变化的坐标转换：
    在带定向变化的坐标转换中，监控始终是以刀具中心为参照的。MD21020 无任何影响。

  说明

  机床数据 $MC_TRAFO_INCLUDES_TOOL_... 只在特定的坐标转换中计算。 可进行计算的条件是：与基准坐标系相关的刀具定向不会因坐标转换而发生改变。 在标准坐标转换中，只有“倾斜轴”满足该条件。

特性

自动运行模式

- 带/无坐标转换
  如果一个零件程序段中所包含的运动指令有可能导致超过工作区域限制，则不执行该零件程序段。
- 带叠加运动
  如果轴的叠加运行有可能导致超过工作区域限制，则该轴会以最大的减速度（无急动度限制 (BRISK)）减速，并停止在工作区域限制处。 其他参与该运动的轴会按照当前的速度特性（例如：SOFT）制动。 设置不同的减速度可能会导致轨迹关联丢失（即轮廓偏差）。

手动运行模式

- JOG，带/无坐标转换
  轴停止在工作区域限制处。

启用方式

如果在启用工作区域限制时，轴在允许的工作区外运动，它会立即以最大减速度停止。

在 JOG 运行模式下超出工作区域限制

在 JOG 运行模式下，轴最多可以运行到工作区域限制处。 重新按下移动键后，系统会显示报警，轴也不再继续运行。

跨通道取几何轴

通过以下机床数据可以设置是否在跨通道取几何轴时保留或取消工作区域限制。

MD10604 $MN_WALIM_GEOAX_CHANGE_MODE = <值>

<值>	含义
0	在跨通道取几何轴时取消工作区域限制
1	在跨通道取几何轴激活时保留激活的工作区域限制。

3.7.2 BCS 中的工作区域限制

应用

通过“BCS 中的工作区域限制”可以限制车床的工作范围，防止周围设备（如：刀塔、测量站）损坏。

工作区域限制

各轴的工作区域下限和上限是由设定数据来设置或零件程序指令来编程的：

通过设定数据设置工作区域限制

该设置是通过立即生效的轴专用设定数据来进行的：

SD43420 $SA_WORKAREA_LIMIT_PLUS（正向工作区域限制）

SD43430 $SA_WORKAREA_LIMIT_MINUS（负向工作区域限制）

编程的工作区域限制

该编程是通过 G 指令来进行的：

G25 X…Y…Z… 工作区域下限

G26 X…Y…Z… 工作区域上限

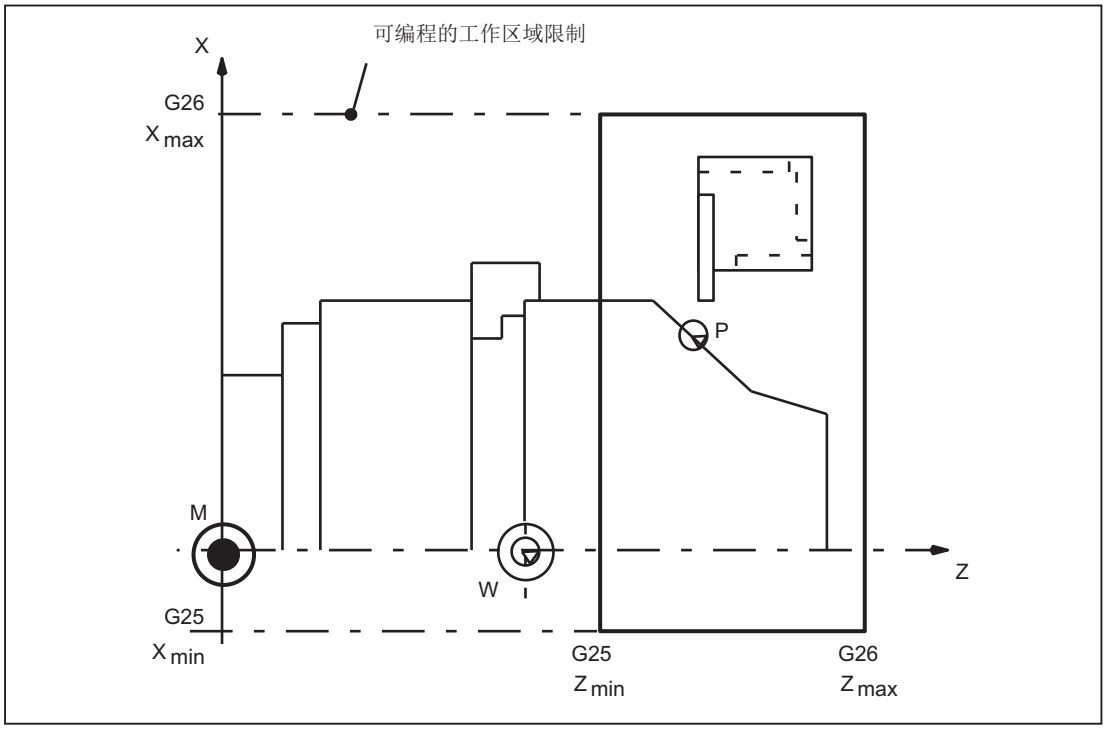


图 3-4 编程的工作区域限制

编程的工作区域限制具有优先权并会覆盖 SD43420 和 SD43430 中已输入的值。

激活/取消

通过设定数据设置工作区域限制

激活或取消各轴的工作区域限制是通过立即生效的设定数据针对各方向进行的：

SD43400 $SA_WORKAREA_PLUS_ENABLE（正向的工作区域限制激活）

SD43410 $SA_WORKAREA_MINUS_ENABLE（负向的工作区域限制激活）

值	含义
0	正向或负向的工作区域限制是**关闭的**。
1	正向或负向的工作区域限制是**激活的**。

编程的工作区域限制

激活或取消整个“BCS 中的工作区域限制”是通过零件程序指令来进行的：

WALIMON　　工作区域限制 开

或者

WALIMOF　　工作区域限制 关

修改工作区域限制

通过设定数据设置工作区域限制

HMI 操作界面：操作区“参数”

- 自动运行模式：
  - 变更：只可在复位状态下进行
  - 有效性：立即生效
- 手动运行模式：
  - 变更：始终可用
  - 有效性：在下一个运行程序段开始时生效

编程的工作区域限制

可在零件程序中通过 G25 或 G26 <轴名称> <值> 来修改工作区域限制。该修改立即生效。

如果为 SD43420 和 SD43430 激活了备份（备份到 NC 的永久数据区），那么在 NC 复位和上电后也会继续保留工作区域限制的新值：

MD10710 $MN_PROG_SD_RESET_SAVE_TAB[0] = 43420

MD10710 $MN_PROG_SD_RESET_SAVE_TAB[1] = 43430

初始设置

可通过以下数据来设置工作区域限制的初始设置 (WALIMON 或 WALIMOF)：

MD20150 $MC_GCODE_RESET_VALUES（G 功能组的初始设置）

3.7.3 WCS/AZS 中的工作区域限制

应用

“WKS/ENS 工作区域限制”可根据通道的不同形成灵活的工件坐标系统（WKS）或可调零点系统（ENS）中通道轴活动区域的工件限制。它主要设计用于传统的车床。

前提条件

必须参考通道轴。

工作区域限制组

为了在切换轴分配，比如在开/关坐标转换或开/关生效的框架时无须每次都为所有的通道轴重新写入工作区域限制，系统提供了工作区域限制组。

一个工作区域限制组包含以下数据：

- 所有通道轴的工作区域限制
- 工作区域限制的参照系

工作区域限制组的数量是在以下机床数据中设置的：

MD28600 $MC_MM_NUM_WORKAREA_CS_GROUPS

每个通道最多可以有 10 个工作区域限制组。

设置工作区域限制

每个通道轴的工作区域限制是通过以下系统变量设置的：

- $P_WORKAREA_CS_LIMIT_PLUS[<WALimNo>, <Ax>]
- $P_WORKAREA_CS_LIMIT_MINUS[<WALimNo>, <Ax>]

其中：
<WALimNo> = 工作区域限制组
值域： 0（组 1）... 9（组 10）
<Ax> = 通道轴名称

使能工作区域限制

每个通道轴的工作区域限制是通过以下系统变量使能的：

- $P_WORKAREA_CS_PLUS_ENABLE[<WALimNo>, <Ax>]
- $P_WORKAREA_CS_MINUS_ENABLE[<WALimNo>, <Ax>]

其中：
<WALimNo> = 工作区域限制组

值域： 0（组 1）... 9（组 10）

<Ax> = 通道轴名称

通过定向使能，可将轴的工作区域限制在一个方向上。

使能并不会激活工作区域限制。

选择参照系

工作区域限制组的参照系是通过以下系统变量设置的：

$P_WORKAREA_CS_COORD_SYSTEM[<WALimNo>] = *<值>*

其中：
<WALimNo> = 工作区域限制组

值域： 0（组 1）... 9（组 10）

<值>	含义
1	参照系为 WCS。
3	参照系为 AZS。

激活工作区域限制

工作区域限制组是在零件程序中通过 G 指令 WALCS<n> 来激活的：

其中：
<n> = 工作区域限制组的编号

值域： 1 ... 10

取消工作区域限制组

工作区域限制组是在零件程序中通过 G 指令 WALCS0 来取消的：

修改工作区域限制

通过上述系统变量可以随时修改工作区域限制。所做的修改会在下一次激活工作区域限制组 (WALCSn) 时生效。

数据存储

工作区域限制的系统变量会永久保存在静态 NC 存储器中。

说明

在线性轴上存储限制值时系统会考虑到单位制 (MD10240 $MN_SCALING_SYSTEM_IS_METRIC) 的缺省设置。

数据备份

工作区域限制的系统变量可保存在单独的文件中。

- _N_CHx_WAL
  用于确保通道 x 的系统变量值。
- _N_COMPLETE_WAL
  用于确保所有通道的系统变量值。

说明

工作区域限制的系统变量是文件 “_N_INITIAL_INI” 的一个组成部分。

JOG 操作模式下的特性

初始情况：

- 在 JOG 操作模式下可同时运行**多个**几何轴（如：通过多个手轮）
- 在工作区域限制的基准坐标系和参考坐标系（WCS 或 AZS）之间有一个激活的**旋转**框架。

工作区域限制响应时的特性：

- 不相关的几何轴继续运行
- 相关的几何轴停止在工作区域限制处

设置初始设置

应在启动、复位或零件程序结束及零件程序开始时生效的工作区域限制组是通过以下机床数据确定的：

MD20150 $MC_GCODE_RESET_VALUE[59] = <n>

其 <n> = 工作区域限制组的编号
中：

值域： 1 ... 10

设置的工作区域限制组在启动、复位或零点程序结束时是否真的使用取决于以下设置：

MD20152 $MC_GCODE_RESET_MODE[59] = *<值>*

<值>	含义
0	符合 MD20150 的工作区域限制生效（初始设置）。
1	最后激活的工作区域限制组保持生效。

3.7.4 示例：WCS/AZS 中的工作区域限制

假设

通道轴

在通道中定义了四个轴：

- 直线轴：X, Y, Z
- 回转轴：A（非模数）

要求

通道轴

在通道中定义了四个轴：

- 直线轴：X, Y, Z
- 回转轴：A（非模数）

工作区域限制组

在通道中需要提供三个工作区域限制组：

MD28600 $MC_MM_NUM_WORKAREA_CS_GROUP = 3

在这三个工作区域限制组中，下面定义两个组。

坐标系

- 工作区域限制组 1：可设定的零点系统（**SZS**）中的工作区域限制
- 工作区域限制组 2：工件坐标系（**WCS**）中的工作区域限制

工作区域限制组 1

- X 轴正方向上：10 mm
- X 轴负方向上：没有限制
- Y 轴正方向上：没有限制
- Y 轴负方向上：25 mm
- Z 轴正方向上：没有限制
- Z 轴负方向上：没有限制
- A 轴正方向上：10 度
- A 轴负方向上：-40 度

通过 NC 程序中的系统变量定义

程序代码

```
; 工作区域限制组 1
$P_WORKAREA_CS_COORD_SYSTEM[1] = 3          ; SZS 中的工作区域限制
$P_WORKAREA_CS_PLUS_ENABLE[1,X] = TRUE
$P_WORKAREA_CS_LIMIT_PLUS[1,X] = 10
$P_WORKAREA_CS_MINUS_ENABLE[1,X] = FALSE
$P_WORKAREA_CS_PLUS_ENABLE[1,Y] = FALSE
$P_WORKAREA_CS_MINUS_ENABLE[1,Y] = TRUE
```

程序代码

```
$P_WORKAREA_CS_LIMIT_MINUS[1,Y] = 25
$P_WORKAREA_CS_PLUS_ENABLE[1,Z] = FALSE
$P_WORKAREA_CS_MINUS_ENABLE[1,Z] = FALSE
$P_WORKAREA_CS_PLUS_ENABLE[1,A] = TRUE
$P_WORKAREA_CS_LIMIT_PLUS[1,A] = 10
$P_WORKAREA_CS_MINUS_ENABLE[1,A] = TRUE
$P_WORKAREA_CS_LIMIT_MINUS[1,A] = -40
```

工作区域限制组 2

- X 轴正方向上：10 mm
- X 轴负方向上：没有限制
- Y 轴正方向上：34 mm
- Y 轴负方向上： -25 mm
- Z 轴正方向上：没有限制
- Z 轴负方向上： -600 mm
- A 轴正方向上：没有限制
- A 轴负方向上：没有限制

通过 NC 程序中的系统变量定义

程序代码

```
; 工作区域限制组 2
$P_WORKAREA_CS_COORD_SYSTEM[2] = 1          ; WCS 中的工作区域限制
$P_WORKAREA_CS_PLUS_ENABLE[2,X] = TRUE
$P_WORKAREA_CS_LIMIT_PLUS[2,X] = 10
$P_WORKAREA_CS_MINUS_ENABLE[2,X] = FALSE
$P_WORKAREA_CS_PLUS_ENABLE[2,Y] = TRUE
$P_WORKAREA_CS_LIMIT_PLUS[2,Y] = 34
$P_WORKAREA_CS_MINUS_ENABLE[2,Y] = TRUE
$P_WORKAREA_CS_LIMIT_MINUS[2,Y] = -25
$P_WORKAREA_CS_PLUS_ENABLE[2,Z] = FALSE
$P_WORKAREA_CS_MINUS_ENABLE[2,Z] = TRUE
$P_WORKAREA_CS_LIMIT_PLUS[2,Z] = -600
$P_WORKAREA_CS_PLUS_ENABLE[2,A] = FALSE
$P_WORKAREA_CS_MINUS_ENABLE[2,A] = FALSE
```

激活

在 NC 程序中通过 WALCS<x> 指令激活工作区域限制组，其中 x：工作区域限制组的编号

3.8 “驻留”机床轴

如果一个机床轴被设为“驻留”状态，就无法再采集与该轴相关的编码器实际值，而且之前章节中描述的所有监控（测量系统、静态和夹紧监控）都被取消。

激活/取消激活

激活驻留

通过**复位**位置测量系统的专用 NC/PLC 接口信号，以及复位机床轴的控制器使能来**激活**“驻留”功能：

- DB31, ... DBX1.5 = 0（位置测量系统 1）
- DB31, ... DBX1.6 = 0（位置测量系统 2）
- DB31, ... DBX2.1 = 0（控制器使能）

轴的位置测量系统的编码器状态随后设置为“未回参考点”：

- DB31, ... DBX60.4 = 0（已回参考点/已同步，位置测量系统 1）
- DB31, ... DBX60.5 = 0（已回参考点/已同步，位置测量系统 2）

下列 NC/PLC 接口信号同样会复位：

- DB31, ... DBX61.5 = 0（位置控制器生效）
- DB31, ... DBX61.6 = 0（转速控制器生效）
- DB31, ... DBX61.7 = 0（电流控制器生效）
- DB31, ... DBX93.7 = 0（脉冲已使能）
- DB31, ... DBX102.5 = 0（位置测量系统 1 接通）
- DB31, ... DBX102.6 = 0（位置测量系统 2 接通）

取消激活驻留

通过为待激活的位置测量系统**置位**轴专用 NC/PLC 接口信号，以及置位机床轴的控制器使能来**取消激活**“驻留”功能：

- DB31, ... DBX1.5 = 1（位置测量系统 1）
  或
  DB31, ... DBX1.6 = 1 （位置测量系统 2）= 1
- DB31, ... DBX2.1 = 1（控制器使能）= 1

随后机床轴的位置闭环控制会在当前位置重新生效。

位置测量系统的编码器状态取决于测量系统类型：

- 增量位置测量系统 ⇒ “未回参考点”状态
  - DB31, ... DBX60.4 = 0（已回参考点/已同步，位置测量系统 1）
  - DB31, ... DBX60.5 = 0（已回参考点/已同步，位置测量系统 2）
- 绝对位置测量系统 ⇒ “已回参考点/已同步”状态
  - DB31, ... DBX60.4 = 1（已回参考点/已同步，位置测量系统 1）
  - DB31, ... DBX60.5 = 1（已回参考点/已同步，位置测量系统 2）

下列 NC/PLC 接口信号同样会重新置位：

- DB31, ... DBX61.5 = 1（位置控制器生效）
- DB31, ... DBX61.6 = 1（转速控制器生效）
- DB31, ... DBX61.7 = 1（电流控制器生效）
- DB31, ... DBX93.7 = 1（脉冲已使能）
- DB31, ... DBX102.5 = 1（位置测量系统 1 接通）
- DB31, ... DBX102.6 = 1（位置测量系统 2 接通）

增量位置测量系统

取消“驻留”状态后，必须为增量位置测量系统重新执行回参考点，以达到“已回参考点”状态。

⚠ 警告

因机床轴实际位置偏差造成的位置测量系统同步故障

如果“驻留”期间位置测量系统有所变化（例如：安装了另一个编码器），而该变化导致需要修改已设置的机床数据，则必须重新完整地校准测量系统并使其回参考点。参见“R1：回参考点 (页 1349)”章节。

无位置测量系统的机床轴

在无位置测量系统的机床轴（转速控制的主轴）上，和“驻留”对应的状态通过撤销伺服使能来激活：

- DB31, ... DBX2.1 = 0（控制器使能）

3.9 被动位置测量系统驻留

3.9.1 功能

与机床轴的所有位置测量系统都被关闭的““驻留”机床轴 (页 131)”功能不同，借助“被动位置测量系统驻留”功能，用户能够实现仅针对机床轴被动位置测量系统的“驻留”（即取消驱动和控制系统中的编码器检测及监控），而生效的位置测量系统仍保持运行。

说明

关于主动/被动测量系统的解释请见“设定值/实际值系统 (页 384)”。

应用

“被动位置测量系统驻留”功能可应用在以下示例情形中：

- 带/不带内置编码器的附加测头的切换
  通过“被动位置测量系统驻留”功能可以实现针对不同的加工任务在主主轴上交替安装带或不带内置编码器的附加测头，而不会因为缺少编码器信号而导致驱动故障和控制系统故障。
  另见：
  - 示例： 直接位置测量系统上的附加测头切换 (页 137)
  - 示例： 两个直接位置测量系统上的附加测头切换 (页 142)
- 线性位置测量系统的使用，该系统无法在机床轴的整个运行区域上使用
  通过“被动位置测量系统驻留”功能可以实现在线性位置测量系统范围以外的运行，而不会因为缺少编码器信号而导致驱动故障和控制系统故障。
  另见：
  - "示例： 编码器无法适用运行范围时的测量系统切换 (页 147)".

激活/取消激活

激活

机床轴的被动位置测量系统会在以下条件下驻留：

- 为测量系统激活了“被动测量系统驻留”功能：
  MD31046 $MA_ENC_PASSIVE_PARKING[<n>] = 1
  其中 <n> = 0（位置测量系统 1）或 1（位置测量系统 2）

说明

以下情形下 MD31046 不生效：

- 轴配备的编码器少于两个：
  MD30200 $MA_NUM_ENCS < 2
- 仿真编码器：
  MD30240 $MA_ENC_TYPE = 0

说明

对于用作电机测量系统的位置测量系统，应关闭“被动位置测量系统驻留”功能 (MD31046 = 0)!

与

- 以下 NC/PLC 接口信号由用户置“0”：
  DB31, ... DBX1.5（位置测量系统 1）= 0
  或者
  DB31, ... DBX1.6（位置测量系统 2）= 0

在这些情形下控制系统会将位置测量系统的接通状态信号设置为“0”：

DB31, ... DBX102.5（位置测量系统 1 已激活）== 0

或者

DB31, ... DBX102.6（位置测量系统 2 已激活）== 0

此时将不再对位置测量系统进行监控和更新。

禁用

当用户激活位置测量系统时，“驻留”被关闭：

DB31, ... DBX1.5（位置测量系统 1）= 1

或者

DB31, ... DBX1.6（位置测量系统 2）= 1

在这些情形下控制系统会将位置测量系统的接通状态信号重新设为“1”：

DB31, ... DBX102.5（位置测量系统 1 已激活）== 1

或者

DB31, ... DBX102.6（位置测量系统 2 已激活）== 1

说明

切换到已驻留位置测量系统的用时会比切换到未驻留的位置测量系统长。 鉴于这段时间的存在，应在轴静止状态下进行切换。

位置测量系统的位置

绝对位置测量系统

对于绝对位置测量系统，取消“驻留”后的测量系统位置就是编码器的当前绝对位置。

位置测量系统已回参考点：

DB31, ... DBX60.4 (已回参考点/已同步，位置测量系统 1) == 1

或者

DB31, ... DBX60.5 (已回参考点/已同步，位置测量系统 2) == 1

增量位置测量系统

对于增量位置测量系统，取消“驻留”后的测量系统位置原则上是位置测量系统上一次关闭时的位置。

只有在经参数设定的两个位置测量系统实际值之间的允许偏差(参见 MD36500 $MA_ENC_CHANGE_TOL)未被超出时，才会切换到已驻留的位置测量系统。 否则，用户应使用不进行此类检查的“机床轴驻留功能”。

位置测量系统**未**回参考点：

DB31, ... DBX60.4 (已回参考点/已同步，位置测量系统 1) == 0

或者

DB31, ... DBX60.5 (已回参考点/已同步，位置测量系统 2) == 0

带位置接收的增量位置测量系统

使用带生效“被动位置测量系统驻停”功能(MD31046 $MA_ENC_PASSIVE_PARKING[<n>] = 1)的增量位置测量系统时，还可以在关闭“驻停”后接收之前生效的位置测量系统的位置以及可能存在的“已回参考点”状态。

该功能可通过以下机床数据为机床轴的每个位置测量系统启用：

MD34210 $MA_ENC_REFP_STATE[<n>]

其中 <n> = 0（位置测量系统 1）或 1（位置测量系统 2）

值	含义
1	只接收之前生效的位置测量系统的位置。 位置测量系统**未**回参考点： DB31, ... DBX60.4 (已回参考点/已同步，位置测量系统 1) == 0 或者 DB31, ... DBX60.5 (已回参考点/已同步，位置测量系统 2) == 0
2	接收之前生效的位置测量系统的位置**和**“已回参考点”状态。 位置测量系统已回参考点： DB31, ... DBX60.4 (已回参考点/已同步，位置测量系统 1) == 1 或者 DB31, ... DBX60.5 (已回参考点/已同步，位置测量系统 2) == 1

说明

只在使用增量位置测量系统时依据 MD34210 $MA_ENC_REFP_STATE[<n>]的设置，并且只在“被动位置测量系统驻留”(MD31046 $MA_ENC_PASSIVE_PARKING[<n>] = 1)功能生效时才接收之前生效的位置测量系统的位置和“已回参考点”状态。

所接收位置的精度与之前生效的位置测量系统的精度一致。 如果该精度过低，应重新使位置测量系统回参考点。

⚠ 警告

因机床轴实际位置偏差造成的位置测量系统同步故障

如果“驻留”期间位置测量系统有所变化（例如：安装了另一个编码器），而该变化导致需要修改已设置的机床数据，则必须重新完整地校准测量系统并使其回参考点。参见“R1：回参考点 (页 1349)”章节。

3.9.2 前提条件

与“双位置反馈”的相互影响

“被动位置测量系统驻留”功能**不**能与“双位置反馈”(MD32960 $MA_POSCTRL_DUAL_FEEDBACK_TIME > 0)功能配合使用。

与“位置差接通”的相互影响

“被动位置测量系统驻留”功能不能与“位置差接通”(MD32950 $MA_POSCTRL_DAMPING > 0)功能配合使用。

与 APC（SINUMERIK 840D sl 选件）的相互影响

“被动位置测量系统驻留”功能不能与驱动功能“Advanced Positioning Control（APC，高级定位控制）”配合使用。

与编码器安全保护方案的相互影响

启用“被动位置测量系统驻留”功能的同时只能使用单编码器安全保护方案。

与 DRIVE-CLiQ 上插拔操作的相互影响

若插拔的不是编码器电缆，而是 SMC 和电机模块间的 DRIVE-CLiQ 电缆，则只有采用““驻留”机床轴 (页 131)”功能才能确保此类编码器的无故障驻留。

3.9.3 示例： 直接位置测量系统上的附加测头切换

初始状态

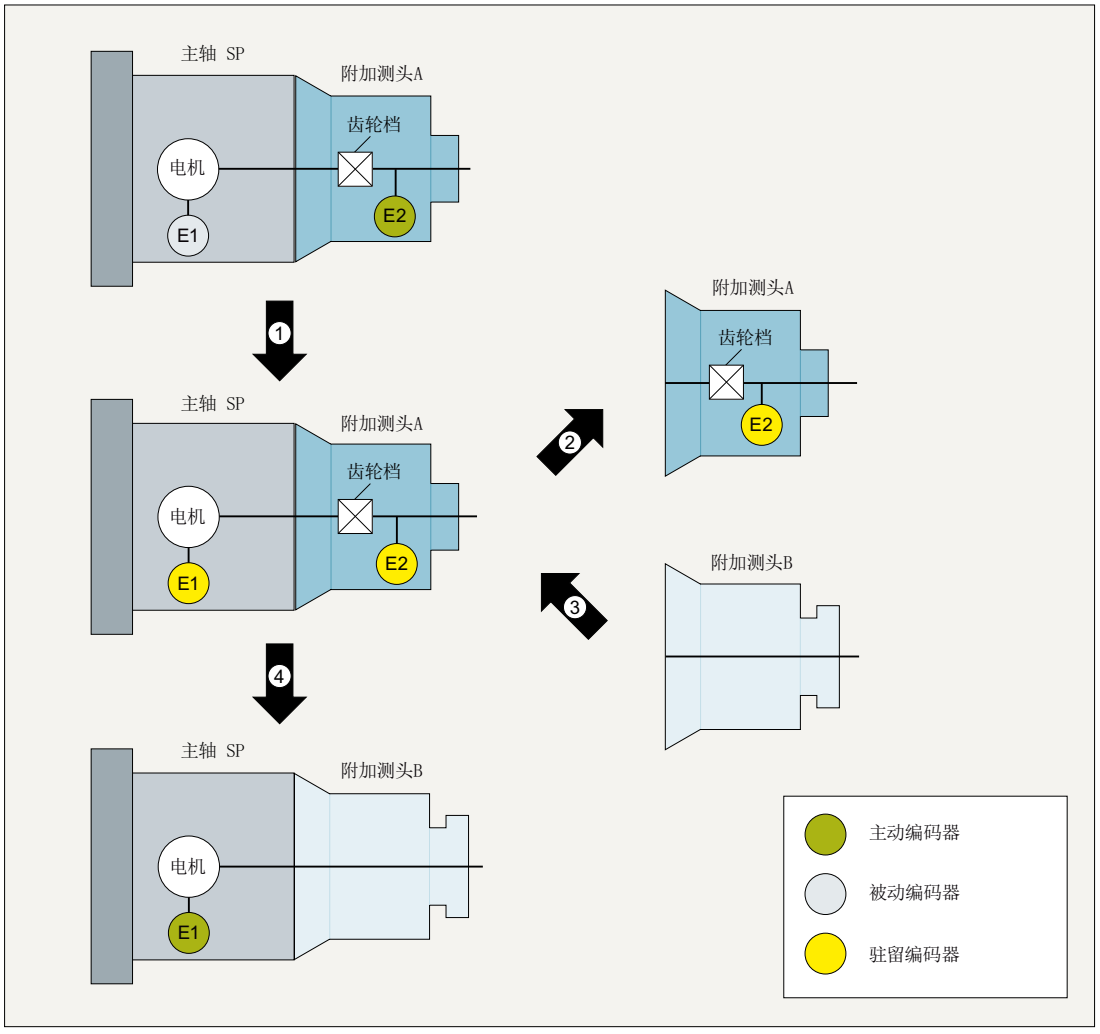
- 附加测头“A”配备了一个编码器 E2。
- 附加测头“B”无编码器。
- 主轴“SP”配备了一个编码器 E1。
- 在 MD13060 $MN_DRIVE_TELEGRAM_TYPE（PROFIdrive 的标准报文类型）中设置了以下两种报文类型其中之一：
  - 报文 116（电机编码器 + 外部编码器）
    或者
  - 报文 136（电机编码器 + 外部编码器，含转矩前馈）
- 主轴“SP”中配置了以下位置测量系统：
  - 电机编码器 E1 作为位置测量系统 1
  - 直接编码器 E2 作为位置测量系统 2
- 当前主轴上安装的是带编码器 E2 的附加测头“A”。

- 位置测量系统 2 是生效的测量系统：
  DB31, ... DBX1.6 = 1
  位置测量系统 1 无效。
- “被动位置测量系统驻留”功能：
  - 未对位置测量系统 1 生效：
    MD31046 $MA_ENC_PASSIVE_PARKING [0] = 0
  - 对位置测量系统 2 生效：
    MD31046 $MA_ENC_PASSIVE_PARKING [1] = 1

目标

用户希望从附加测头“A”切换到附加测头“B”。

执行



① 在附加测头切换前，用户应通过功能“ “驻留”机床轴 (页 131)”关闭机床轴的所有位置测量系统：

DB31, ... DBX1.5（位置测量系统 1）= 0

DB31, ... DBX1.6（位置测量系统 2）= 0

接着控制系统会复位位置测量系统的状态信号：

DB31, ... DBX102.5（位置测量系统 1 已激活）== 0

DB31, ... DBX102.6（位置测量系统 2 已激活）== 0

② 用户等待状态信号，之后再将附加测头“A”从主轴上移除。此时附加测头“A”和联轴节间的编码器电缆也会电气断开。编码器 E2 的缺失不会触发 NC 或 驱动故障。

③ 现在主轴上安装的是附加测头“B”。

④ 用户只接通位置测量系统 1：

DB31, ... DBX1.5（位置测量系统 1）= 1

接着控制系统对状态信号置位：

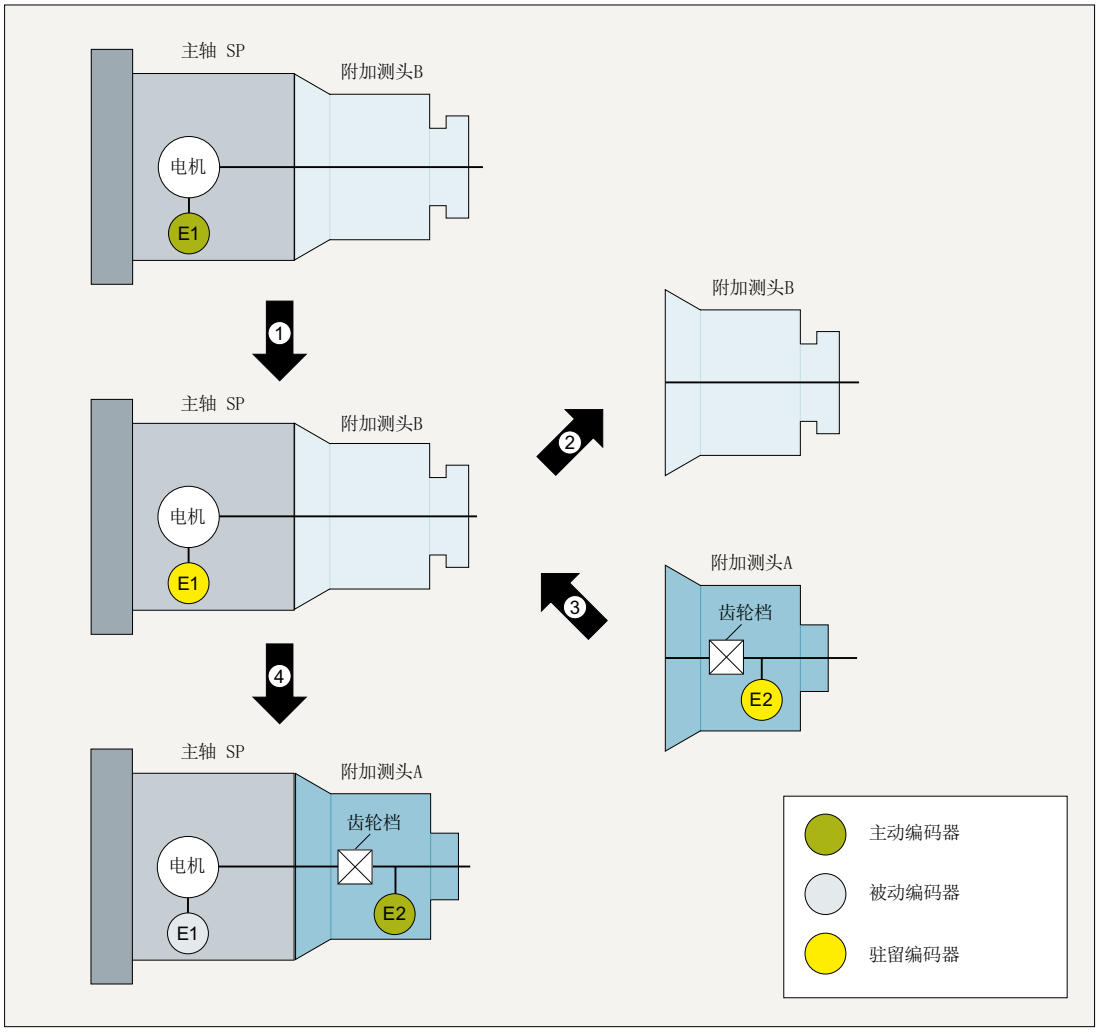
DB31, ... DBX102.5（位置测量系统 1 已激活）== 1

由于已为位置测量系统 2 激活了“被动位置测量系统驻留”功能，因此位置测量系统 2 无效，而是处于“驻留”状态。

目标

接下来用户希望重新换入附加测头“A”。

执行



① 用户通过“机床轴驻留”功能关闭位置测量系统 1：

DB31, ... DBX1.5（位置测量系统 1）= 0

接着控制系统会复位位置测量系统的状态信号：

DB31, ... DBX102.5（位置测量系统 1 已激活）== 0

② 用户等待状态信号，之后再将附加测头“B”从主轴上移除。

③ 现在主轴上安装的是附加测头“A”。

④ 用户激活位置测量系统 2：

DB31, ... DBX1.6（位置测量系统 2）= 1

这样可同时接通位置测量系统 1，因为位置测量系统 1（电机测量系统！）的“被动位置测量系统驻留”功能未生效。位置测量系统 1 成为被动位置测量系统。

控制系统对位置测量系统的状态信号进行置位：

DB31, ... DBX102.5（位置测量系统 1 已激活）== 1

DB31, ... DBX102.6（位置测量系统 2 已激活）== 1

3.9.4 示例： 两个直接位置测量系统上的附加测头切换

初始状态

- 附加测头“A”配备了一个编码器 E3。
- 附加测头“B”无编码器。
- 主轴“SP”配备了两个编码器 E1 和 E2。
- 在 MD13060 $MN_DRIVE_TELEGRAM_TYPE（PROFIdrive 的标准报文类型）中设置了以下两种报文类型其中之一：
  - 报文 118（两个外部编码器）
    或者
  - 报文 138（两个外部编码器，含转矩前馈）

  文档：

  编码器分配相关信息参见：

  调试手册“CNC 调试：NC、PLC、驱动”；

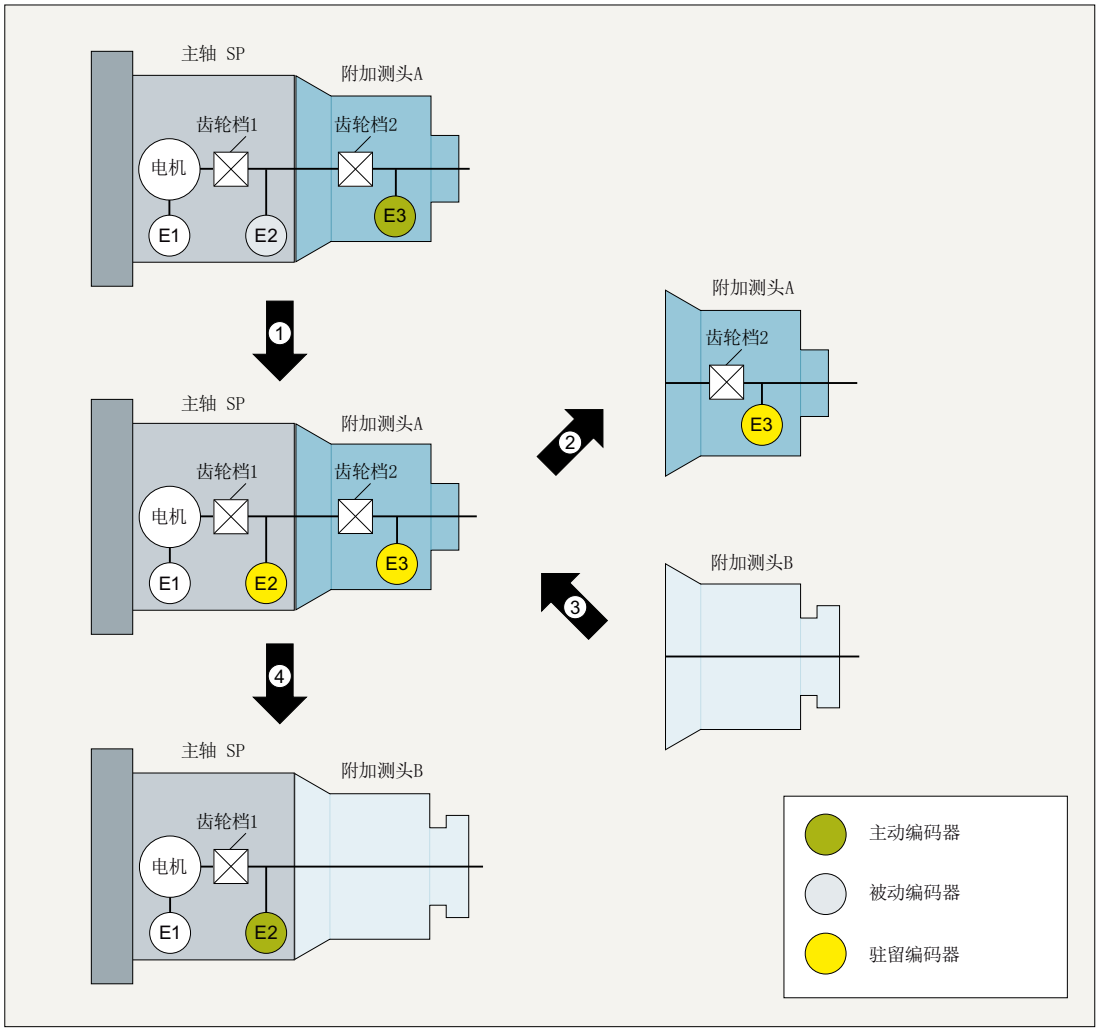
  章节：“NC 和驱动之间的通讯” > “驱动：轴分配”
- 主轴“SP”中配置了以下位置测量系统：
  - 直接编码器 E2 作为位置测量系统 1
  - 直接编码器 E3 作为位置测量系统 2
- 当前主轴上安装的是带编码器 E3 的附加测头“A”。

- 位置测量系统 2 是生效的测量系统：
  DB31, ... DBX1.6 = 1
  位置测量系统 1 无效。
- “被动位置测量系统驻留”功能：
  – 未对位置测量系统 1 生效：
    MD31046 $MA_ENC_PASSIVE_PARKING [0] = 0
  – 对位置测量系统 2 生效：
    MD31046 $MA_ENC_PASSIVE_PARKING [1] = 1

目标

用户希望从附加测头“A”切换到附加测头“B”。

执行



① 在附加测头切换前，用户应通过功能“ “驻留”机床轴 (页 131)”关闭机床轴的所有位置测量系统：

DB31, ... DBX1.5（位置测量系统 1）= 0

DB31, ... DBX1.6（位置测量系统 2）= 0

接着控制系统会复位位置测量系统的状态信号：

DB31, ... DBX102.5（位置测量系统 1 已激活）== 0

DB31, ... DBX102.6（位置测量系统 2 已激活）== 0

② 用户等待状态信号，之后再将附加测头“A”从主轴上移除。此时附加测头“A”和联轴节间的编码器电缆也会电气断开。编码器 E3 的缺失不会触发 NC 或 驱动故障。

③ 现在主轴上安装的是附加测头“B”。

④ 用户只接通位置测量系统 1：

DB31, ... DBX1.5（位置测量系统 1）= 1

控制系统对状态信号置位：

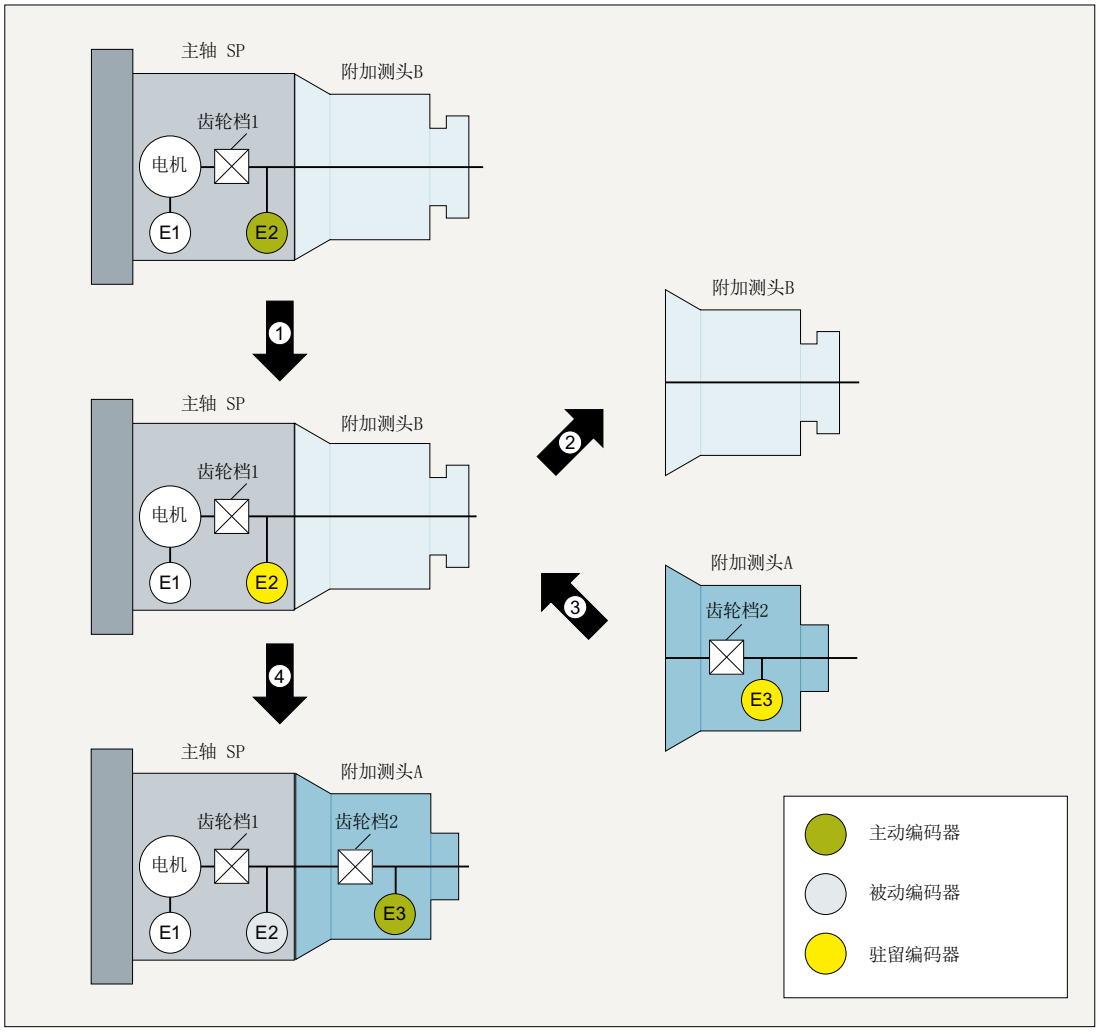
DB31, ... DBX102.5（位置测量系统 1 已激活）== 1

由于已为位置测量系统 2 激活了“被动位置测量系统驻留”功能，因此位置测量系统 2 无效，而是处于“驻留”状态。

目标

接下来用户希望重新换入附加测头“A”。

执行



① 用户通过“机床轴驻留”功能关闭位置测量系统 1：

DB31, ... DBX1.5（位置测量系统 1）= 0

接着控制系统会复位位置测量系统的状态信号：

DB31, ... DBX102.5（位置测量系统 1 已激活）== 0

② 用户等待状态信号，之后再将附加测头“B”从主轴上移除。

③ 现在主轴上安装的是附加测头“A”。

④ 用户激活位置测量系统 2：

DB31, ... DBX1.6（位置测量系统 2）= 1

这样可同时接通位置测量系统 1，因为位置测量系统 1 的“被动位置测量系统驻留”功能未生效。位置测量系统 1 成为被动位置测量系统。

控制系统对位置测量系统的状态信号进行置位：

DB31, ... DBX102.5（位置测量系统 1 已激活）== 1

DB31, ... DBX102.6（位置测量系统 2 已激活）== 1

3.9.5 示例： 编码器无法适用运行范围时的测量系统切换

在下面的示例中直接线性位置测量系统只能在加工区域内使用，加工区域之外的运行范围中只能使用电机测量系统。

初始状态

- 线性轴“X”配备了两个增量编码器：
  - 电机编码器 E1
  - 直接线性编码器 E2
- 直接线性编码器 E2 仅存在于加工区域中。
- 在 MD13060 $MN_DRIVE_TELEGRAM_TYPE（PROFIdrive 的标准报文类型）中设置了以下两种报文类型其中之一：
  - 报文 116（电机编码器 + 外部编码器）
    或者
  - 报文 136（电机编码器 + 外部编码器，含转矩前馈）
- 为机床轴配置了以下位置测量系统：
  - 电机编码器 E1 作为位置测量系统 1
  - 直接线性编码器 E2 作为位置测量系统 2
- 启动机床时用户接通两个位置测量系统：
  - DB31, ... DBX1.5（位置测量系统 1）= 1
  - DB31, ... DBX1.6（位置测量系统 2）= 1

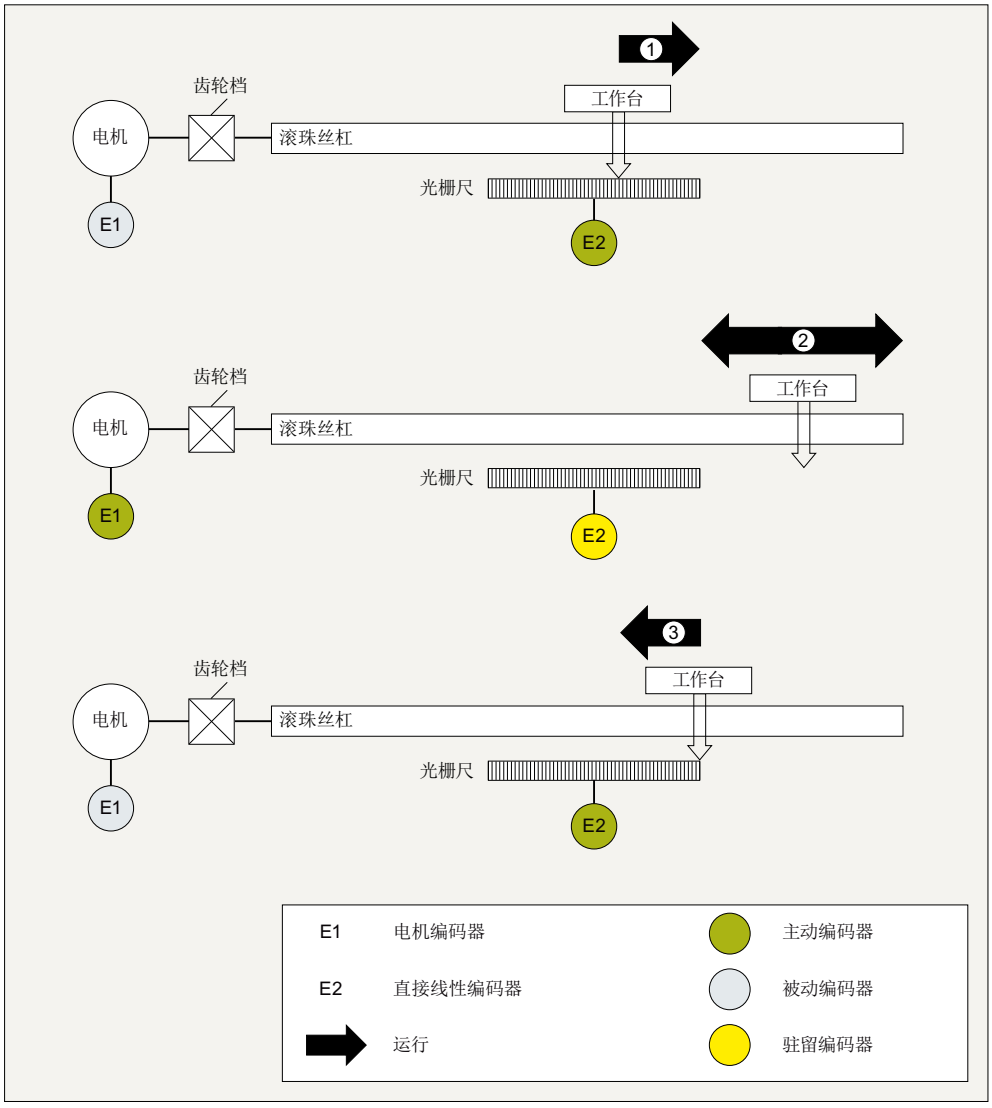
  同时激活两个位置测量系统时，控制系统会选择位置测量系统 1 作为生效的位置测量系统。

- 加工时会切换到位置测量系统 2：
  - DB31, ... DBX1.5（位置测量系统 1）= 0
  - DB31, ... DBX1.6（位置测量系统 2）= 1
- “被动位置测量系统驻留”功能：
  - 未对位置测量系统 1 生效：
    MD31046 $MA_ENC_PASSIVE_PARKING [0] = 0
  - 对位置测量系统 2 生效：
    MD31046 $MA_ENC_PASSIVE_PARKING [1] = 1
- 启用对位置测量系统 2 的位置和“已回参考点”状态的接收：
  MD34210 $MA_ENC_REFP_STATE[1] = 2

目标

在线性位置测量系统 E2 的适用范围以外运行时，缺失的编码器信号不会导致驱动和控制系统出现故障。

执行



① 在工作台到达线性位置测量系统的终点前，应切换到电机测量系统。用户可通过激活两个位置测量系统来达到此目的：

DB31, ... DBX1.5（位置测量系统 1）= 1

DB31, ... DBX1.6（位置测量系统 2）= 1

通过“被动位置测量系统驻留”功能，控制系统在测量系统切换后会使被动线性位置测量系统驻留。

控制系统复位状态信号：

DB31, ... DBX102.6（位置测量系统 2 已激活）== 0

用户等待状态信号，之后再继续线性位置测量系统适用范围之外区域的运行。

② 使用电机测量系统进行线性位置测量系统适用范围之外区域的运行。

③ 当工作台返回线性位置测量系统的适用范围时，用户可在静止时从电机测量系统切换到线性位置测量系统：

DB31, ... DBX1.5（位置测量系统 1）= 0

DB31, ... DBX1.6（位置测量系统 2）= 1

控制系统对状态信号置位：

DB31, ... DBX102.6（位置测量系统 2 已激活）== 1

电机测量系统成为被动位置测量系统。

结果

两个位置测量系统均已回参考点。线性位置测量系统的位置就是切换时电机测量系统的位置。如果位置精度过低，应重新使线性位置测量系统回参考点。

3.10 边界条件

其他机床数据

除轴监控机床数据以外，还需要设置或检查以下机床数据：

所有机床轴

- MD31030 $MA_LEADSCREW_PITCH（丝杠螺距）
- MD31050 $MA_DRIVE_AX_RATIO_DENOM（负载齿轮比分母）
- MD31060 $MA_DRIVE_AX_RATIO_NUMERA（负载齿轮比分子）
- MD31070 $MA_DRIVE_ENC_RATIO_DENOM（测量齿轮比分母）
- MD31080 $MA_DRIVE_ENC_RATIO_NUMERA（测量齿轮比分子）
- MD32810 $MA_EQUIV_SPEEDCTRL_TIME（用于前馈控制的转速环等效时间常数）
- 编码器分辨率：参见章节“G2：速度、设定值/实际值系统、闭环控制 (页 363)”。

带模拟转速设定值接口的机床轴

- MD32260 $MA_RATED_VELO（额定电机转速）
- MD32250 $MA_RATED_OUTVAL（额定输出电压）

3.11 数据表

3.11.1 机床数据

3.11.1.1 NC 专用机床数据

编号	名称：$MN_	说明
10604	WALIM_GEOAX_CHANGE_MODE	切换几何轴时的工作区域限制
10710	PROG_SD_RESET_SAVE_TAB	待更新的设定数据

3.11.1.2 通道专用机床数据

编号	名称：$MC_	说明
20150	GCODE_RESET_VALUES	G 功能组的初始设置
21020	WORKAREA_WITH_TOOL_RADIUS	工作区域限制中考虑刀具半径
24130	TRAFO_INCLUDES_TOOL_1	第 1 转换生效时的刀具处理
24230	TRAFO_INCLUDES_TOOL_2	第 2 转换生效时的刀具处理
24330	TRAFO_INCLUDES_TOOL_3	第 3 转换生效时的刀具处理
24426	TRAFO_INCLUDES_TOOL_4	第 4 转换生效时的刀具处理
24436	TRAFO_INCLUDES_TOOL_5	第 5 转换生效时的刀具处理
24446	TRAFO_INCLUDES_TOOL_6	第 6 转换生效时的刀具处理
24456	TRAFO_INCLUDES_TOOL_7	第 7 转换生效时的刀具处理
24466	TRAFO_INCLUDES_TOOL_8	第 8 转换生效时的刀具处理
24476	TRAFO_INCLUDES_TOOL_9	第 9 转换生效时的刀具处理
24486	TRAFO_INCLUDES_TOOL_10	第 10 转换生效时的刀具处理
25106	TRAFO_INCLUDES_TOOL_11	第 11 转换生效时的刀具处理
25116	TRAFO_INCLUDES_TOOL_12	第 12 转换生效时的刀具处理
25126	TRAFO_INCLUDES_TOOL_13	第 13 转换生效时的刀具处理
25136	TRAFO_INCLUDES_TOOL_14	第 14 转换生效时的刀具处理
25146	TRAFO_INCLUDES_TOOL_15	第 15 转换生效时的刀具处理
25156	TRAFO_INCLUDES_TOOL_16	第 16 转换生效时的刀具处理

编号	名称：$MC_	说明
25166	TRAFO_INCLUDES_TOOL_17	第 17 转换生效时的刀具处理
25176	TRAFO_INCLUDES_TOOL_18	第 18 转换生效时的刀具处理
25186	TRAFO_INCLUDES_TOOL_19	第 19 转换生效时的刀具处理
25196	TRAFO_INCLUDES_TOOL_20	第 20 转换生效时的刀具处理
28600	MM_NUM_WORKAREA_CS_GROUPS	坐标系专用的工作区域限制的数量

3.11.1.3 进给轴/主轴专用机床数据

编号	名称：$MA_	说明
30200	NUM_ENCS	编码器数量
30240	ENC_TYPE	检测实际值的编码器类型（位置实际值）
30310	ROT_IS_MODULO	回转轴/主轴的模数转换
30800	WORK_AREA_CHECK_TYPE	工作区域限制的检查方式
31046	ENC_PASSIVE_PARKING	被动位置测量系统驻留
32200	POSCTRL_GAIN[n]	K_V 系数
32250	RATED_OUTVAL	额定输出电压
32260	RATED_VELO	电机额定转速
32300	MAX_AX_ACCEL	最大轴加速度
32800	EQUIV_CURRCTRL_TIME	电流环前馈控制的等效时间常数
32810	EQUIV_SPEEDCTRL_TIME	转速环前馈控制的等效时间常数
32910	DYN_MATCH_TIME[n]	动态响应自适应的时间常数
34210	ENC_REFP_STATE	编码器状态
35160	SPIND_EXTERN_VELO_LIMIT	PLC 主轴转速限制
36000	STOP_LIMIT_COARSE	粗准停
36010	STOP_LIMIT_FINE	精准停
36020	POSITIONING_TIME	精准停延迟时间
36030	STANDSTILL_POS_TOL	静态公差
36040	STANDSTILL_DELAY_TIME	静态监控延迟时间
36050	CLAMP_POS_TOL	接口信号“夹紧生效”的夹紧公差
36052	STOP_ON_CLAMPING	夹紧轴上的特殊功能

编号	名称：$MA_	说明
36060	STANDSTILL_VELO_TOL	最大进给轴/主轴停止速度
36100	POS_LIMIT_MINUS	第 1 个负向软件限位开关
36110	POS_LIMIT_PLUS	第 1 个正向软件限位开关
36120	POS_LIMIT_MINUS2	第 2 个负向软件限位开关
36130	POS_LIMIT_PLUS2	第 2 个正向软件限位开关
36200	AX_VELO_LIMIT	速度监控阈值
36210	CTRLOUT_LIMIT	最大转速设定值
36220	CTRLOUT_LIMIT_TIME	转速设定值监控的延迟时间
36300	ENC_FREQ_LIMIT	编码器极限频率
36302	ENC_FREQ_LIMIT_LOW	编码器重新同步的极限频率
36310	ENC_ZERO_MONITORING	零脉冲监控
36312	ENC_ABS_ZEROMON_WARNING	零脉冲监控报警阈值
36400	CONTOUR_TOL	轮廓监控公差带
36500	ENC_CHANGE_TOL	位置实际值切换时的最大公差
36510	ENC_DIFF_TOL	测量系统同步公差
36600	BRAKE_MODE_CHOICE	硬件限位开关的制动性能
36610	AX_EMERGENCY_STOP_TIME	故障状态下制动斜坡的最大持续时间
36620	SERVO_DISABLE_DELAY_TIME	伺服使能封锁延时

3.11.2 设定数据

3.11.2.1 进给轴/主轴专用设定数据

编号	名称：$SA_	说明
43400	WORKAREA_PLUS_ENABLE	正向工作区域限制生效
43410	WORKAREA_MINUS_ENABLE	负向工作区域限制生效
43420	WORKAREA_LIMIT_PLUS	正工作区域限制
43430	WORKAREA_LIMIT_MINUS	负工作区域限制

3.11.3 信号

3.11.3.1 发送至进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
跟踪运行	DB31, ...DBX1.4	DB380x.DBX1.4
位置测量系统 1/2	DB31, ...DBX1.5/6	DB380x.DBX1.5/6
伺服使能	DB31, ...DBX2.1	DB380x.DBX2.1
正在进行夹紧	DB31, ...DBX2.3	DB380x.DBX2.3
速度/主轴转速限制	DB31, ...DBX3.6	DB380x.DBX3.6
进给停止/主轴停止	DB31, ...DBX4.3	DB380x.DBX4.3
负硬件限位开关/正硬件限位开关	DB31, ...DBX12.0/1	DB380x.DBX1000.0/1
负向软件限位开关/第 2 软件限位开关 正	DB31, ...DBX12.2/3	DB380x.DBX1000.2/3

3.11.3.2 来自进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
超过编码器极限频率 1/2	DB31, ...DBX60.2/3	DB390x.DBX0.2
已回参考点/已同步 1/2	DB31, ...DBX60.4/5	DB390x.DBX0.4/5
运行指令 -/+	DB31, ...DBX64.6/7	DB390x.DBX4.6/7
位置测量系统 1/2 已激活	DB31, ... DBX102.5/6	DB390x.DBX5006.5/6

A5：保护区

4

4.1 功能

保护区是需要由用户定义的，机床内部用于防止机床组件碰撞的 2 维或 3 维静态或动态区域。

可保护以下组件：

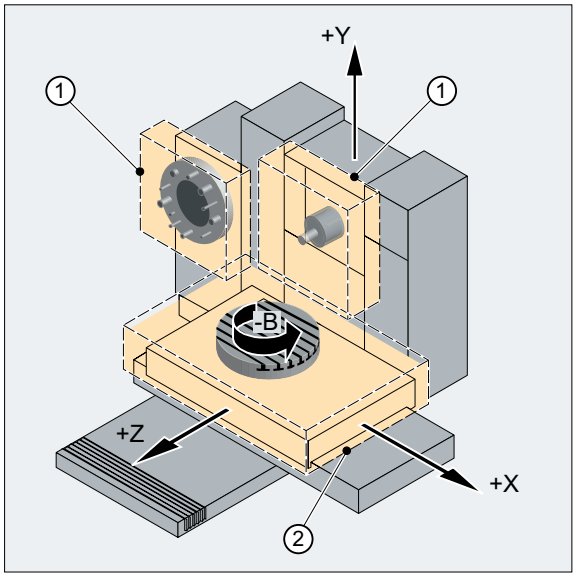
- 静止的机床组件（例如刀库，可摆动的测头）
- 属于刀具的可动机床组件（例如刀具，刀架）
- 属于工件的可动机床组件（例如工件的零件，装夹台，夹爪，主轴卡盘，尾架）

为保证对机床组件的保护，必须如下定义保护区：使得待保护的元件完全被包围。

针对特定通道对保护区进行监控，即监控一条通道中所激活的所有保护区是否会相互碰撞。

在 AUTO 或 MDA 运行模式下自动执行零件程序时，NC 会在程序开头中检查按照编程的轨迹运行时，是否可能在保护区内出现碰撞。

手动取消一个激活的保护区后便可进入该保护区。离开保护区后，保护区会再次自动激活。



① **刀具**相关的保护区

② **工件**相关的保护区

图 4-1 铣床上的保护区示例

定义保护区

可以 2 维或 3 维方式，用最多包含 10 个角点的曲线和圆弧将保护区作为轮廓单元定义。可通过零件程序中的指令（参见“定义保护区 (CPROTDEF、NPROTDEF) (页 163)”）或通过系统变量进行定义。其中轮廓单元均位于可通过 G17、G18 或 G19 选择的平面中。

第 3 维度

第 3 维度的保护区范围可限定在 - ∞ 到 + ∞ 之间。

- -∞ 至 +∞
- -∞ 至上限值
- 下限值至 +∞
- 下限值至上限值

坐标系

保护区的定义基于通道的几何轴，故在基准坐标系（BCS）中进行。

基准

- 刀具相关的保护区
  刀具相关保护区的坐标必须参照**刀架参考点 F** 以绝对方式给定。
- 工件相关的保护区：
  工件相关保护区的坐标必须参照**基本坐标系（BCS）**的零点以绝对方式给定。

说明

若无刀具相关保护区生效，则系统按照工件相关保护区对刀具轨迹进行检查。
若无工件相关保护区生效，则不进行保护区监控。

定向

保护区是通过确定轮廓所在的平面（横坐标/纵坐标）以及垂直于该保护区轮廓的轴（垂直坐标）来定向的。

刀具相关保护区与工件相关保护区的定向必须是相同的。

机床/通道专用保护区

- 机床专用保护区
  机床专用保护区的数据是在控制系统中一次性定义的。可由所有通道激活这些保护区。
- 通道专用保护区
  通道专用保护区的数据是在通道中定义的。这些保护区只能由该通道来激活。

系统变量

通过零件程序中的指令定义保护区时，系统会将保护区数据保存在系统变量中。也可直接写入系统变量，即直接在系统变量中定义保护区。在描述保护区的轮廓时，通过零件程序中的指令定义保护区时适用的边界条件同样适用（参见“定义保护区 (CPROTDEF、NPROTDEF) (页 163)”）。

保护区定义包含以下系统变量：

系统变量	类型	含义	
$SN_PA_ACTIV_IMMED[<n>] $SC_PA_ACTIV_IMMED[<n>]	BOOL	激活类型 保护区在控制系统启动以及轴回参考点后立即生效/失效。	
		FALSE	不立即生效
		TRUE	立即生效
$SN_PA_T_W[<n>] $SC_PA_T_W[<n>]	INT	保护区类型	
		0	工件相关的保护区
		1	预留
		2	预留
		3	刀具相关的保护区
$SN_PA_ORI[<n>] $SC_PA_ORI[<n>]	INT	保护区的定向，即平面中的曲线由以下轴组成：	
		0	第 1 和第 2 几何轴
		1	第 3 和第 1 几何轴
		2	第 2 和第 3 几何轴
$SN_PA_LIM_3DIM[<n>] $SC_PA_LIM_3DIM[<n>]	INT	第 3 个维度的限制类型	
		0	无限制
		1	正向限制
		2	负向限制
		3	正负方向的限制
$SN_PA_PLUS_LIM[<n>] $SC_PA_PLUS_LIM[<n>]	REAL	Wert der Begrenzung in Plus-Richtung in der 3.维度的限制类型	
$SN_PA_MINUS_LIM[<n>] $SC_PA_MINUS_LIM[<n>]	REAL	Wert der Begrenzung in Minus-Richtung in der 3.维度的限制类型	
$SN_PA_CONT_NUM[<n>] $SC_PA_CONT_NUM[<n>]	INT	有效的轮廓单元的数目	

系统变量	类型	含义
$SN_PA_CONT_TYP[<n>, <i>] $SC_PA_CONT_TYP[<n>, <i>]	INT	轮廓类型 [<i>]，第 <i> 个轮廓单元的轮廓类型（G1，G2，G3）
$SN_PA_CONT_ABS[<n>, <i>] $SC_PA_CONT_ABS[<n>, <i>]	REAL	轮廓 [<i>] 的终点，横坐标值
$SN_PA_CONT_ORD[<n>, <i>] $SC_PA_CONT_ORD[<n>, <i>]	REAL	轮廓 [<i>] 的终点，纵坐标值
$SN_PA_CENT_ABS[<n>, <i>] $SC_PA_CENT_ABS[<n>, <i>]	REAL	圆弧轮廓 [<i>] 的中心 ，绝对横坐标值
$SN_PA_CENT_ORD[<n>, <i>] $SC_PA_CENT_ORD[<n>, <i>]	REAL	圆弧轮廓 [<i>] 的中心 ，绝对纵坐标值
$SN_... 是用于 NC 或机床专用保护区的系统变量。 $SC_... 是用于通道专用保护区的系统变量。 索引 “<n>” 对应保护区的编号：0 = 第 1 保护区 索引“<i>”对应轮廓单元的编号：0 = 第 1 轮廓单元 必须按升序的方式来定义轮廓单元。		

说明

执行 REORG 指令时不会恢复保护区定义的系统变量！

保护区定义的数据

数据存储

保护区定义保存在以下文件中：

文件	模块
_N_NCK_PRO	针对机床专用保护区的数据块
_N_CHAN1_PRO	针对通道 1 中的通道专用保护区的数据块
_N_CHAN2_PRO	针对通道 2 中的通道专用保护区的数据块

数据备份

保护区定义备份在以下文件中：

文件	模块
_N_INITIAL_INI	保护区的所有数据块
_N_COMPLETE_PRO	保护区的所有数据块
_N_CHAN_PRO	通道专用保护区的所有数据块

激活、预激活和取消激活保护区

激活状态

保护区的激活状态可以是：

- 激活
- 预激活
- 带有条件停止的预激活
- 取消激活

即便就机床专用保护区而言，激活状态总是针对特定通道！

在零件程序中激活、预激活和取消激活

可通过零件程序中的指令随时修改保护区的激活状态（参见“激活/取消激活保护区 (CPROT, NPROT) (页 167)”）。

说明

在激活保护区的通道中，仅在所有通道均回参考点后，系统才会将该保护区考虑在内。

若需要在随后的时间点上通过 PLC 用户程序激活保护区（参见下文），则必须在零件程序中进行预激活。

预激活的保护区是通过以下 NC/PLC 接口信号显示的：

- DB21, ... DBX272.0 - 273.1（机床专用保护区 1 - 10 预激活）== 1
- DB21, ... DBX274.0 - 275.1（通道专用保护区 1 - 10 预激活）== 1

带有条件停止的预激活

注意
可能超出保护区边界 若未及时激活带有条件停止的预激活保护区，NC 可能会因至激活时间点为止未考虑制动路径而无法在保护区前及时停止。

对于带有条件停止的预激活保护区，在运行将进入保护区的情况下，运行**不**会在这个保护区前停止。仅在将保护区激活时才会停止。针对仅暂时需要保护区的情形，上述特性能够实现可由用户控制的不间断加工。

通过 NC/PLC 接口信号激活

只有通过零件程序**预激活**的保护区才能在 PLC 用户程序中通过 NC/PLC 接口信号激活：

- DB21, ... DBX8.0 - 9.1（激活机床专用保护区 1 - 10）= 1
- DB21, ... DBX10.0 - 11.1（激活通道专用保护区 1 - 10）= 1

必须在几何轴运行前激活已预激活的保护区！如果在运行期间激活，则系统不会再在当前运行中考虑该保护区。响应：

- 报警 10704 “未确保保护区监控”
- DB21, ... DBX39.0（未确保保护区监控）= 1

说明

必须在几何轴运行前激活已预激活的保护区！

通过 NC/PLC 接口信号取消激活

仅通过零件程序**预激活**并且通过 NC/PLC 接口信号激活的保护区才能借助 NC/PLC 接口信号重新取消激活：

- DB21, ... DBX8.0 至 DBX9.1（激活机床专用保护区 1 - 10）= 0
- DB21, ... DBX10.0 至 DBX11.1（激活通道专用保护区 1 - 10）= 0

通过零件程序直接**激活**的保护区**无法**通过 PLC 用户程序来取消。

在坐标转换切换/跨通道取几何轴时自动取消激活

在缺省设置下，系统会在转换切换以及跨通道取几何轴时自动取消生效的保护区。若需要保持保护区生效，则需相应调整位编码机床数据 MD10618 $MN_PROTAREA_GEOAX_CHANGE_MODE（参见“机床数据 (页 162)”）。

特殊系统状态下的激活状态

带计算的程序段搜索

在带计算的程序段搜索中会始终考虑到最后编程的保护区激活状态。

程序测试

在 AUTO 和 MDA 运行模式中，系统也会在“程序测试”状态下监控激活和预激活的保护区。

NC 复位及程序结束

NC 复位和程序结束后，保护区的激活状态保持不变。

显示超出保护区边界的情况

超出激活的保护区边界，或者超出预激活的保护区（若其被激活）边界的情况通过以下 NC/PLC 接口信号显示：

- DB21, ... DBX276.0 - 277.1（超出机床专用保护区 1 - 10 的边界）== 1
- DB21, ... DBX278.0 - 279.1（超出通道专用保护区 1 - 10 的边界）== 1

检查是否超出保护区边界

可通过 CALCPOSI 功能进行检查，以确定是否能在不超出保护区边界的情况下使几何轴运行经过设定的路径（参见“检查保护区、工作区域限制和软件限位开关 (CALCPOSI) (页 171)”）。

4.2 调试

4.2.1 机床数据

所需存储容量

保护区所需的存储容量是通过以下机床数据设置的：

- 永久存储器
  - MD18190 $MN_MM_NUM_PROTECT_AREA_NCK（可用机床专用保护区数量）
  - MD28200 $MC_MM_NUM_PROTECT_AREA_CHAN（可用通道专用保护区数量）
- 动态存储器
  - MD28210 $MC_MM_NUM_PROTECT_AREA_ACTIVE（通道中可同时激活的保护区的最大数量）
  - MD28212 $MC_MM_NUM_PROTECT_AREA_CONTUR（通道中可定义轮廓单元的最大数量）

在转换切换/跨通道取几何轴时的特性：

通过以下机床数据定义，在执行坐标转换切换或者跨通道取几何轴时，保留激活的保护区还是将其取消：

MD10618 $MN_PROTAREA_GEOAX_CHANGE_MODE

位	值	含义
0	0	执行坐标转换切换时，激活的保护区被取消。
	1	执行坐标转换切换时，激活的保护区继续生效。
1	0	执行跨通道取几何轴时，激活的保护区被取消。
	1	执行跨通道取几何轴时，激活的保护区继续生效。

4.3 编程

4.3.1 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_Name>
G17/G18/G19
CPROTDEF/NPROTDEF(<n>,<t>,<AppLim>,<AppPlus>,<AppMinus>)
G0/G1/...X/Y/Z...
...
EXECUTE(<Var_Name>)
```

含义

DEF INT <Var_Name>：	定义数据类型为 INTEGER 的本地辅助变量	
<Var_Name>：	辅助变量的名称	
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 个维度的限制类型		
	数据类型：	实数	
<AppMinus>:	第 3 个维度的限制类型		
	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_Name>):	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 167)”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

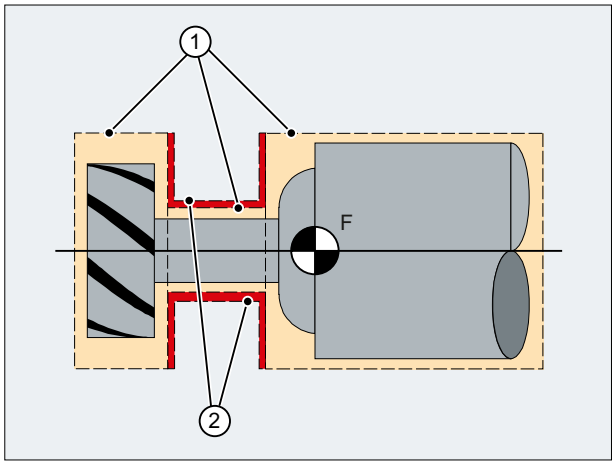
- 与刀具有关的保护区：
  与**刀具**相关的保护区的坐标必须**以参照刀架参考点 F 的绝对值的形式**给出。
- 与工件相关的保护区
  与**工件**相关的保护区的坐标必须**以参照基准坐标系零点的绝对值的形式**给出。

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

4.3.2 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

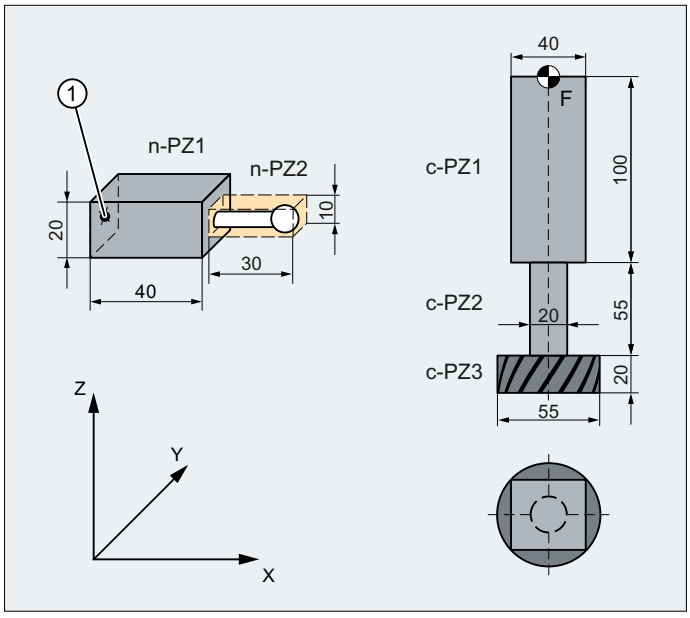
对于铣床而言，应对铣刀与探头可能会有的碰撞进行监控。探头的位置应在激活时通过位移来设定。

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	

程序代码	注释
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。

4.3.3 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 <System>, INT <TestLim>)
```

句法

```
<State> = CALCPOSI(VAR <Start>, VAR <Dist>, VAR <Limit>, VAR
<MaxDist>, <System>, <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

<State>: （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。

<State>: （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。
<State>: （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。
<State>: （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。

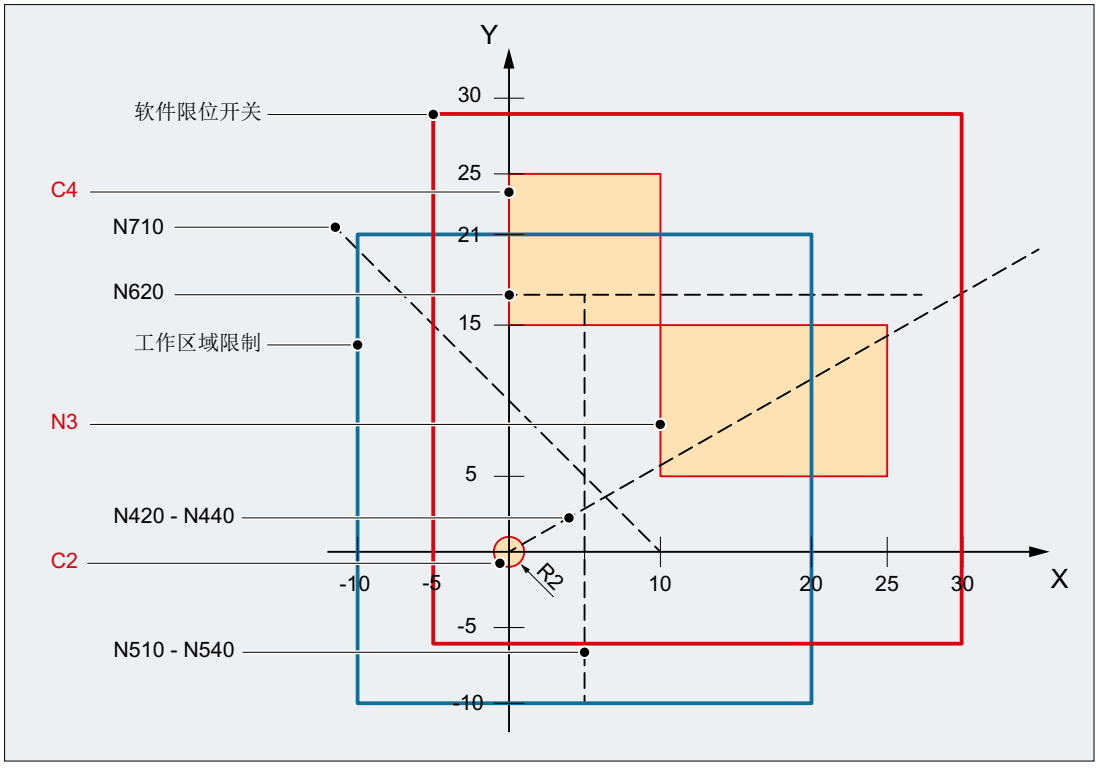
<State>: （第 5 部分）	十万位		
	值：	0xxxxx	十万位 == 0：<Dist> 保持不变
		1xxxxx	**<Dist>**中返回一个方向矢量，该矢量定义了限位面上的后续运动方向。 只允许在以下边界条件中出现： ● 超出软件限位开关或工作区域限制（不在起点） ● 坐标转换**未**激活 ● **<TestID>**，位 4 或 5 == 1
<Start>:	含起始位置的矢量： ● **<Start> [0]**:第 1 几何轴 ● **<Start> [1]**:第 2 几何轴 ● **<Start> [2]**:第 3 几何轴		
	参数类型：	输入	
	数据类型：	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**<State>**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

<System>:	位置和长度的单位（英制/公制）（**可选**）		
	数据类型：		BOOL
	值：	FALSE（默认）	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。
		TRUE	符合所设基本单位制的单位制： MD52806 $MN_ISO_SCALING_SYSTEM
<TestLim>:	选择需要监控的限位，位编码（**可选**）		
	数据类型：		INT
	默认值：		位 0、1、2、3、6、7 == 1 (207)
	位	**十进制**	**含义**
	0	1	软件限位开关
	1	2	工作区域限制
	2	4	激活的传统保护区
	3	8	预激活的传统保护区
	4	16	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**。
	5	32	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**。
	6	64	激活的防撞保护区
	7	128	预激活的防撞保护区
	8	256	激活的防撞保护区和预激活的防撞保护区
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。			

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 _STATE
```

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 _STATE = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST)
N430 _STATE = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,,3)
N440 _STATE = 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 _STATE = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,,14)
N520 _STATE = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,, 6)
N530 _LIMIT[1] = 2.
N540 _STATE = 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 _STATE = 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 _STATE = 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 _STATE = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,,17)
N830 M30
```

CALCPOSI() 的结果

N...	<State>	<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（<State>）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。

4.4 特殊情形

4.4.1 暂时允许轴进入保护区

如果在启动或运行期间超出保护区边界，在特定前提条件下可暂时释放保护区，即暂时允许轴穿过。通过操作既能在 AUTO、MDA 也能在 JOG 运行模式下暂时允许轴进入保护区。

另见：

- AUTO 和 MDA 运行模式下的特性 (页 182)
- JOG 运行模式下的特性 (页 183)

说明

暂时释放**仅**适用于**工件**相关的保护区！**刀具**相关的保护区必须在零件程序中取消激活，或是通过 NC/PLC 接口设置为“预激活”状态。

暂时允许轴进入保护区是在以下事件后结束的：

- 控制系统启动
- AUTO 或 MDA 运行模式：程序段末尾处于保护区外
- JOG 运行模式：运行终点处于保护区外
- 保护区激活

4.4.2 AUTO 和 MDA 运行模式下的特性

在 AUTO 和 MDA 运行模式下，不允许有轴进入或穿过保护区：

- 可能从外部进入保护区的轴会停止在位于保护区外部的最后一个程序段终点处。
- 从某保护区内出发的运动不会开始。

暂时允许轴进入保护区

如果在 AUTO 或 MDA 运行模式下，轴运行因超出保护区边界而停止，系统便会向操作员显示一个报警。若操作员决定继续轴运行，则可通过使能允许穿越保护区。

使能只是暂时的，并且通过触发 NC 启动实现：

DB21, ... DBX7.1（NC 启动）= 1

每当轴超出保护区边界时，系统都会显示一条报警。操作员必须为每个待释放的保护区触发一个 NC 启动信号。

在操作员触发重启，允许轴暂时进入之前导致报警的所有保护区后，轴才继续运行。

继续轴运行，不允许轴暂时进入保护区

轴超出保护区边界时，系统会停止轴运行并发出报警。若相关保护区通过 NC/PLC 接口设为“预激活”状态，便可通过 NC 启动继续运行，而无需允许轴进入保护区。

加强对轴进入保护区的控制

如果不希望通过单纯的 NC 启动即允许轴暂时进入保护区，机床制造商或用户可通过 PLC 用户程序加强对该允许的控制。

4.4.3 JOG 运行模式下的特性

同时运行多个几何轴

在 JOG 运行模式下可同时运行多个几何轴。每个参与的几何轴的运行范围会在轴运行开始时受到运行范围界限（软件限位开关、工作区域限制等）以及生效的保护区的限制。系统无法确保对所有生效保护区的可靠监控。此时会向用户发送反馈信息：

- 报警 10704 ：“未确保保护区监控”
- DB31, ... DBX39.0（未确保保护区监控）= 1

运行结束后报警自动删除。

如果当前位置位于激活或预激活的保护区内，则触发以下动作：

- 报警 10702“手动模式下超出 NC 保护区的边界”或 10703“手动模式下超出通道专用保护区的边界”，并给出被超出的保护区和运行的轴。
- 禁止轴继续运行。
- 针对相关的保护区，置位以下 NC/PLC 接口信号：
  DB21, … DBX276.0 – 277.1（超出机床专用保护区 1 - 10 的边界）== 1
  或者
  DB21, … DBX278.0 – 279.1（超出通道专用保护区 1 - 10 的边界）== 1

有关继续运行的信息请参见“暂时允许轴进入保护区”段落。

示例：

三个激活的保护区以及同时运行两个几何轴：

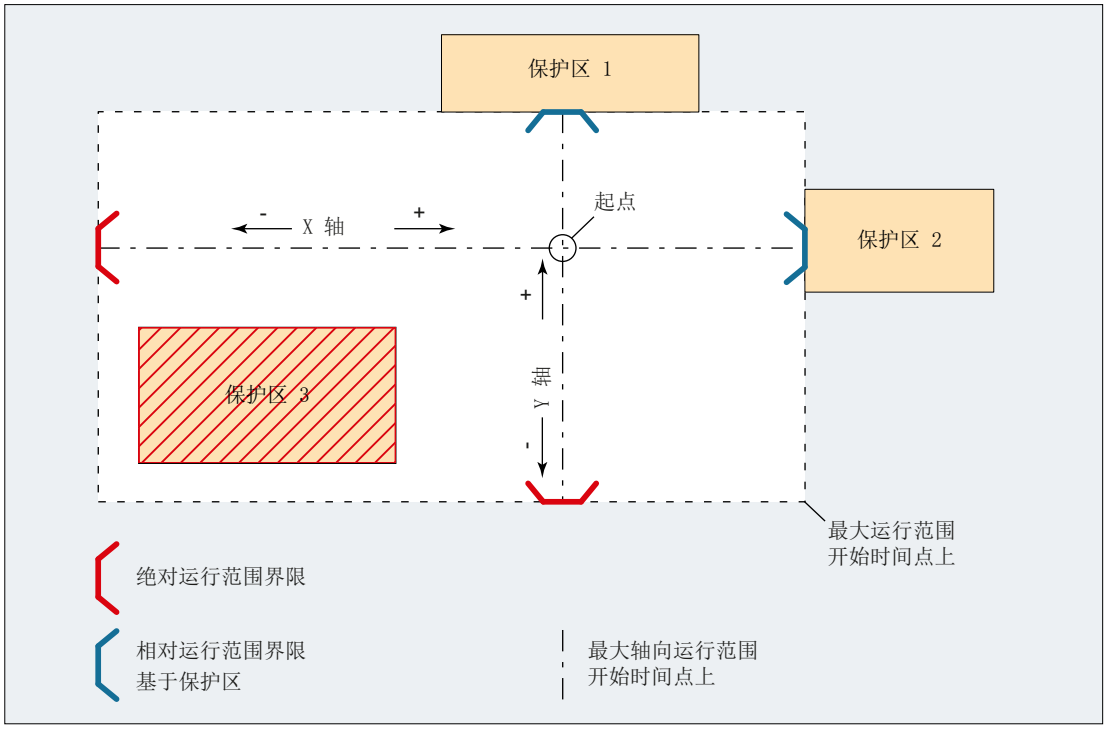


图 4-2　　起始时间点上几何轴的运行范围

在轴 X 和 Y 的运行的起始时间点上，自起点起确定轴专用运行范围界限：

- X 轴
  - 正运行方向：保护区 2
  - 负运行方向：绝对运行范围界限（例如软件限位开关）
- Y 轴
  - 正运行方向：保护区 1
  - 负运行方向：绝对运行范围界限（例如工作区域限制）

由此得出的起始时间点上的最大运行范围并未考虑到保护区 3。因此，有可能在保护区 3 中超出保护区边界。

说明

系统也会在手动运行模式（JOG、INC 及 DRF）下监控激活和预激活的保护区。

轴运行限制

若轴的运行因到达保护区而受到限制，则系统会发出报警 10706“手动模式下到达 NC 保护区”或 10707“手动模式下到达通道专用保护区”，并给出到达的保护区和运行的轴。在通过 JOG 运行轴时确保不到达保护区的边界。（此特性与运行至软件限位开关或工作区域限制类似。）

系统会删除报警：

- 在运行不进入保护区的轴时。
- 在允许轴进入保护区时。
- 在控制系统启动时。

若在保护区边界上启动朝向保护区的运行，则系统同样会发出报警 10706 或 10707，并且不会启动运行。

暂时允许轴进入保护区

若在激活的保护区内或其边界上启动运行，则系统会发出报警 10702“手动模式下超出 NC 保护区的边界”或 10703“手动模式下超出通道专用保护区的边界”，并给出被超出的保护区和运行的轴，并且不会启动运行。在为此暂时释放相关保护区的情况下，才可执行运行。为此需要执行以下操作：

- 在以下 NC/PLC 接口信号上生成一个上升沿：
  DB21, ... DBX1.1（允许轴进入保护区）
- 重新启动原本的运行。

说明

对于暂时释放的保护区而言，在其被穿过时，NC/PLC 信号“超出保护区边界”会置位。

DB21, … DBX276.0 – 277.1（超出机床专用保护区 1 - 10 的边界）== 1

或者

DB21, … DBX278.0 – 279.1（超出通道专用保护区 1 - 10 的边界）== 1

如果启动的轴运行不会进入保护区，系统会复位该使能。

如果轴运行涉及到其他的保护区，则系统会为每个保护区显示报警 10702 或 10703。可通过在 NC/PLC 接口信号 DB21, ... DBX1.1 上生成上升沿来允许轴进入显示在报警中的保护区。

切换运行方式时的特性

在 JOG 运行模式中对保护区作出的临时使能在切换至 AUTO 或 MDA 运行模式后继续有效。同样，在 AUTO 或 MDA 运行模式中作出的临时使能也在切换至 JOG 运行模式后继续有效。

复位使能

在几何轴下一次静止并完全离开暂时允许轴进入的保护区时，系统会在内部和以下 NC/PLC 接口上复位该使能：

DB21, … DBX276.0 – 277.1（超出机床专用保护区 1 - 10 的边界）== 0

或者

DB21, … DBX278.0 – 279.1（超出通道专用保护区 1 - 10 的边界）== 0

4.5 边界条件

保护区监控的局限性

在以下情形下**无法**进行保护区监控：

- 定向轴的运行
- **静止机床**，端面转换（TRANSMIT）或柱面转换（TRACYL）中的特殊保护区
  特例：相对主轴旋转对称的保护区。其中不允许有 DRF 偏移生效。
- **刀具**相关保护区的互相监控

定位轴

在定位轴上，系统只会监控编程的程序段终点。在定位轴运行期间显示报警 10704“不确保保护区监控”。

跨通道取轴

就保护区而言，在跨通道取轴后，在输出通道中以该通道中最后运行到的位置为起始点。系统不会考虑接收通道中的轴运行。因此须注意，不从超出保护区边界的位置开始进行跨通道取轴。

若为跨通道取轴设定的轴未在通道中激活，则将该轴在该通道中最后运行到的位置作为当前位置。若尚未在通道中运行过这个轴，则采用位置 0.0。

叠加运动时的特性

有关生效的保护区的程序段预处理可不考虑已算在主处理中的叠加运动。

会发生以下响应：

- 报警 10704 ：“未确保保护区监控”
- DB31, ... DBX39.0 = 1（未确保保护区监控）

4.6 示例

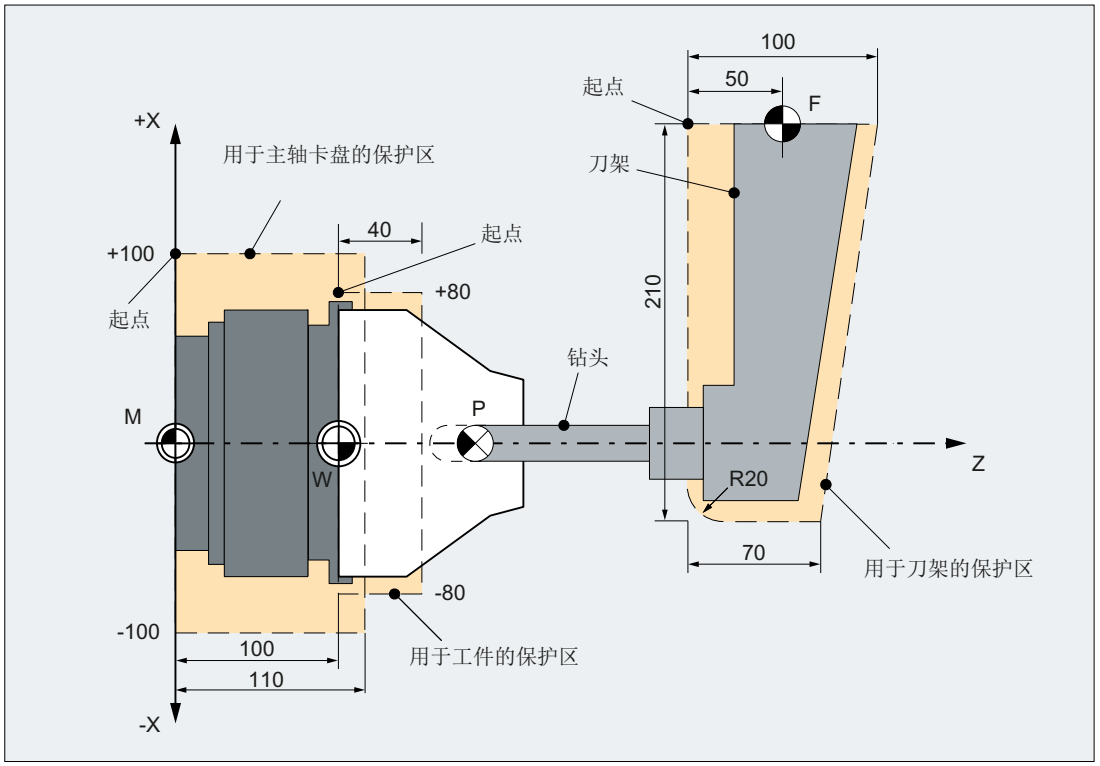
4.6.1 车床上的保护区

需要为车床定义以下内部保护区：

- ein maschinenspezifischer und werkstückbezogener Schutzbereich für das Spindelfutter, ohne Begrenzung in der 3.维度的限制类型
- ein kanalspezifischer Schutzbereich für das Werkstück, ohne Begrenzung in der 3.维度的限制类型
- ein kanalspezifischer, werkzeugbezogener Schutzbereich für den Werkzeugträger, ohne Begrenzung in der 3.维度的限制类型

为定义用于工件的保护区，将工件零点与机床零点重合。

激活保护区时，保护区会在 Z 轴正向移动 100 mm。



4.6.2 零件程序中的保护区定义

用于保护区定义的零件程序节选：

程序代码	注释
DEF INT AB	
G18	; 工作平面 ZX
NPROTDEF(1,FALSE,0,0,0)	; 定义开始：用于主轴卡盘的保护区
G01 X100 Z0	; 轮廓描述：运行至轮廓
G01 X-100 Z0	; 轮廓描述：第 1 轮廓单元
G01 X-100 Z110	; 轮廓描述：第 2 轮廓单元
G01 X100 Z110	; 轮廓描述：第 3 轮廓单元
G01 X100 Z0	; 轮廓描述：第 4 轮廓单元
EXECUTE(AB)	; 定义结束：用于主轴卡盘的保护区
CPROTDEF(1,FALSE,0,0,0)	; 定义开始：用于工件的保护区
G01 X80 Z0	; 轮廓描述：运行至轮廓
G01 X-80 Z0	; 轮廓描述：第 1 轮廓单元
G01 X-80 Z40	; 轮廓描述：第 2 轮廓单元
G01 X80 Z40	; 轮廓描述：第 3 轮廓单元
G01 X80 Z0	; 轮廓描述：第 4 轮廓单元
EXECUTE(AB)	; 定义结束：用于工件的保护区
CPROTDEF(2,TRUE,0,0,0)	; 定义开始：用于刀架的保护区
G01 X0 Z-50	; 轮廓描述：运行至轮廓
G01 X-190 Z-50	; 轮廓描述：第 1 轮廓单元
G03 X-210 Z-30 I-20	; 轮廓描述：第 2 轮廓单元
G01 X-210 Z20	; 轮廓描述：第 3 轮廓单元
G01 X0 Z50	; 轮廓描述：第 4 轮廓单元
G01 X0 Z-50	; 轮廓描述：第 5 轮廓单元
EXECUTE(AB)	; 定义结束：用于刀架的保护区

4.6.3 通过系统变量定义保护区

用于主轴卡盘的机床专用保护区

系统变量	值	说明
$SN_PA_ACTIV_IMMED[0]	0	用于主轴卡盘的保护区不立即生效
$SN_PA_T_W[0]	0	用于主轴卡盘的保护区与工件相关
$SN_PA_ORI[0]	1	保护区的定向：1 = 第 3 和第 1 几何轴
$SN_PA_LIM_3DIM[0]	0	第 3 个 Dimension:0 = 没有限制

系统变量	值	说明
$SN_PA_PLUS_LIM[0]	0	Wert der Begrenzung in Plus-Richtung in der 3.维度的限制类型
$SN_PA_MINUS_LIM[0]	0	Wert der Begrenzung in Minus-Richtung in der 3.维度的限制类型
$SN_PA_CONT_NUM[0]	4	有效的轮廓单元的数目
$SN_PA_CONT_TYP[0.0]	1	轮廓类型[<i>] :1 = G1，直线， 用于主轴卡盘的保护区，轮廓单元 1
$SN_PA_CONT_TYP[0.1]	1	轮廓类型[<i>] :1 = G1，直线， 用于主轴卡盘的保护区，轮廓单元 2
$SN_PA_CONT_TYP[0.2]	1	轮廓类型[<i>] :1 = G1，直线， 用于主轴卡盘的保护区，轮廓单元 3
$SN_PA_CONT_TYP[0.3]	1	轮廓类型[<i>] :1 = G1，直线， 用于主轴卡盘的保护区，轮廓单元 4
$SN_PA_CONT_TYP[0.4]	1	轮廓类型[<i>] :1 = G1，直线， 用于主轴卡盘的保护区，轮廓单元 5
$SN_PA_CONT_TYP[0.5]	0	轮廓类型[<i>] :0 = 未定义， 用于主轴卡盘的保护区，轮廓单元 6
$SN_PA_CONT_TYP[0.6]	0	轮廓类型[<i>] :0 = 未定义， 用于主轴卡盘的保护区，轮廓单元 7
$SN_PA_CONT_TYP[0.7]	0	轮廓类型[<i>] :0 = 未定义， 用于主轴卡盘的保护区，轮廓单元 8
$SN_PA_CONT_TYP[0.8]	0	轮廓类型[<i>] :0 = 未定义， 用于主轴卡盘的保护区，轮廓单元 9
$SN_PA_CONT_TYP[0.9]	0	轮廓类型[<i>] :0 = 未定义， 用于主轴卡盘的保护区，轮廓单元 10
$SN_PA_CONT_ORD[0.0]	-100	轮廓[<i>]的终点，纵坐标值 用于主轴卡盘的保护区，轮廓单元 1
$SN_PA_CONT_ORD[0.1]	-100	轮廓[<i>]的终点，纵坐标值 用于主轴卡盘的保护区，轮廓单元 2
$SN_PA_CONT_ORD[0.2]	100	轮廓[<i>]的终点，纵坐标值 用于主轴卡盘的保护区，轮廓单元 3
$SN_PA_CONT_ORD[0.3]	100	轮廓[<i>]的终点，纵坐标值 用于主轴卡盘的保护区，轮廓单元 4

系统变量	值	说明
$SN_PA_CONT_ORD[0.4]	0	轮廓[<i>]的终点，纵坐标值 用于主轴卡盘的保护区，轮廓单元 5
$SN_PA_CONT_ORD[0.5]	0	轮廓[<i>]的终点，纵坐标值 用于主轴卡盘的保护区，轮廓单元 6
$SN_PA_CONT_ORD[0.6]	0	轮廓[<i>]的终点，纵坐标值 用于主轴卡盘的保护区，轮廓单元 7
$SN_PA_CONT_ORD[0.7]	0	轮廓[<i>]的终点，纵坐标值 用于主轴卡盘的保护区，轮廓单元 8
$SN_PA_CONT_ORD[0.8]	0	轮廓[<i>]的终点，纵坐标值 用于主轴卡盘的保护区，轮廓单元 9
$SN_PA_CONT_ORD[0.9]	0	轮廓[<i>]的终点，纵坐标值 用于主轴卡盘的保护区，轮廓单元 10
$SN_PA_CONT_ABS[0.0]	0	轮廓[<i>]的终点，横坐标值 用于主轴卡盘的保护区，轮廓单元 1
$SN_PA_CONT_ABS[0.1]	110	轮廓[<i>]的终点，横坐标值 用于主轴卡盘的保护区，轮廓单元 2
$SN_PA_CONT_ABS[0.2]	110	轮廓[<i>]的终点，横坐标值 用于主轴卡盘的保护区，轮廓单元 3
$SN_PA_CONT_ABS[0.3]	0	轮廓[<i>]的终点，横坐标值 用于主轴卡盘的保护区，轮廓单元 4
$SN_PA_CONT_ABS[0.4]	0	轮廓[<i>]的终点，横坐标值 用于主轴卡盘的保护区，轮廓单元 5
$SN_PA_CONT_ABS[0.5]	0	轮廓[<i>]的终点，横坐标值 用于主轴卡盘的保护区，轮廓单元 6
$SN_PA_CONT_ABS[0.6]	0	轮廓[<i>]的终点，横坐标值 用于主轴卡盘的保护区，轮廓单元 7
$SN_PA_CONT_ABS[0.7]	0	轮廓[<i>]的终点，横坐标值 用于主轴卡盘的保护区，轮廓单元 8
$SN_PA_CONT_ABS[0.8]	0	轮廓[<i>]的终点，横坐标值 用于主轴卡盘的保护区，轮廓单元 9
$SN_PA_CONT_ABS[0.9]	0	轮廓[<i>]的终点，横坐标值 用于主轴卡盘的保护区，轮廓单元 10
$SN_PA_CENT_ORD[0.0]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于主轴卡盘的保护区，轮廓单元 1

系统变量	值	说明
$SN_PA_CENT_ORD[0.1]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于主轴卡盘的保护区，轮廓单元 2
$SN_PA_CENT_ORD[0.2]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于主轴卡盘的保护区，轮廓单元 3
$SN_PA_CENT_ORD[0.3]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于主轴卡盘的保护区，轮廓单元 4
$SN_PA_CENT_ORD[0.4]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于主轴卡盘的保护区，轮廓单元 5
$SN_PA_CENT_ORD[0.5]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于主轴卡盘的保护区，轮廓单元 6
$SN_PA_CENT_ORD[0.6]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于主轴卡盘的保护区，轮廓单元 7
$SN_PA_CENT_ORD[0.7]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于主轴卡盘的保护区，轮廓单元 8
$SN_PA_CENT_ORD[0.8]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于主轴卡盘的保护区，轮廓单元 9
$SN_PA_CENT_ORD[0.9]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于主轴卡盘的保护区，轮廓单元 10
$SN_PA_CENT_ABS[0.0]	0	圆弧轮廓[<i>]的中心，横坐标值 用于主轴卡盘的保护区，轮廓单元 1
$SN_PA_CENT_ABS[0.1]	0	圆弧轮廓[<i>]的中心，横坐标值 用于主轴卡盘的保护区，轮廓单元 2
$SN_PA_CENT_ABS[0.2]	0	圆弧轮廓[<i>]的中心，横坐标值 用于主轴卡盘的保护区，轮廓单元 3
$SN_PA_CENT_ABS[0.3]	0	圆弧轮廓[<i>]的中心，横坐标值 用于主轴卡盘的保护区，轮廓单元 4
$SN_PA_CENT_ABS[0.4]	0	圆弧轮廓[<i>]的中心，横坐标值 用于主轴卡盘的保护区，轮廓单元 5
$SN_PA_CENT_ABS[0.5]	0	圆弧轮廓[<i>]的中心，横坐标值 用于主轴卡盘的保护区，轮廓单元 6
$SN_PA_CENT_ABS[0.6]	0	圆弧轮廓[<i>]的中心，横坐标值 用于主轴卡盘的保护区，轮廓单元 7
$SN_PA_CENT_ABS[0.7]	0	圆弧轮廓[<i>]的中心，横坐标值 用于主轴卡盘的保护区，轮廓单元 8

系统变量	值	说明
$SN_PA_CENT_ABS[0.8]	0	圆弧轮廓[<i>]的中心，横坐标值 用于主轴卡盘的保护区，轮廓单元 9
$SN_PA_CENT_ABS[0.9]	0	圆弧轮廓[<i>]的中心，横坐标值 用于主轴卡盘的保护区，轮廓单元 10

用于工件的通道专用保护区

系统变量	值	注释
$SC_PA_ACTIV_IMMED[0]	0	用于工件的保护区不立即生效
$SC_PA_TW[0]	0	用于工件的保护区与工件相关
$SC_PA_ORI[0]	1	保护区的定向：1 = 第 3 和第 1 几何轴
$SC_PA_LIM_3DIM[0]	0	第 3 个 Dimension:0 = 没有限制
$SC_PA_PLUS_LIM[0]	0	Wert der Begrenzung in Plus-Richtung in der 3.维度的限制类型
$SC_PA_MINUS_LIM[0]	0	Wert der Begrenzung in Minus-Richtung in der 3.维度的限制类型
$SC_PA_CONT_NUM[0]	4	有效的轮廓单元的数目
$SC_PA_CONT_TYP[0,0]	1	轮廓类型[<i>] :1 = G1，直线， 用于工件的保护区，轮廓单元 1
$SC_PA_CONT_TYP[0,1]	1	轮廓类型[<i>] :1 = G1，直线， 用于工件的保护区，轮廓单元 2
$SC_PA_CONT_TYP[0,2]	1	轮廓类型[<i>] :1 = G1，直线， 用于工件的保护区，轮廓单元 3
$SC_PA_CONT_TYP[0,3]	1	轮廓类型[<i>] :1 = G1，直线， 用于工件的保护区，轮廓单元 4
$SC_PA_CONT_TYP[0,4]	1	轮廓类型[<i>] :1 = G1，直线， 用于工件的保护区，轮廓单元 5
$SC_PA_CONT_TYP[0,5]	0	轮廓类型[<i>] :0 = 未定义， 用于工件的保护区，轮廓单元 6
$SC_PA_CONT_TYP[0,6]	0	轮廓类型[<i>] :0 = 未定义， 用于工件的保护区，轮廓单元 7
$SC_PA_CONT_TYP[0,7]	0	轮廓类型[<i>] :0 = 未定义， 用于工件的保护区，轮廓单元 8

系统变量	值	注释
$SC_PA_CONT_TYP[0,8]	0	轮廓类型[<i>] :0 = 未定义， 用于工件的保护区，轮廓单元 9
$SC_PA_CONT_TYP[0,9]	0	轮廓类型[<i>] :0 = 未定义， 用于工件的保护区，轮廓单元 10
$SC_PA_CONT_ORD[0,0]	-80	轮廓[<i>]的终点，纵坐标值 用于工件的保护区，轮廓单元 1
$SC_PA_CONT_ORD[0,1]	-80	轮廓[<i>]的终点，纵坐标值 用于工件的保护区，轮廓单元 2
$SC_PA_CONT_ORD[0,2]	80	轮廓[<i>]的终点，纵坐标值 用于工件的保护区，轮廓单元 3
$SC_PA_CONT_ORD[0,3]	80	轮廓[<i>]的终点，纵坐标值 用于工件的保护区，轮廓单元 4
$SC_PA_CONT_ORD[0,4]	0	轮廓[<i>]的终点，纵坐标值 用于工件的保护区，轮廓单元 5
$SC_PA_CONT_ORD[0,5]	0	轮廓[<i>]的终点，纵坐标值 用于工件的保护区，轮廓单元 6
$SC_PA_CONT_ORD[0,6]	0	轮廓[<i>]的终点，纵坐标值 用于工件的保护区，轮廓单元 7
$SC_PA_CONT_ORD[0,7]	0	轮廓[<i>]的终点，纵坐标值 用于工件的保护区，轮廓单元 8
$SC_PA_CONT_ORD[0,8]	0	轮廓[<i>]的终点，纵坐标值 用于工件的保护区，轮廓单元 9
$SC_PA_CONT_ORD[0,9]	0	轮廓[<i>]的终点，纵坐标值 用于工件的保护区，轮廓单元 10
$SC_PA_CONT_ABS[0,0]	0	轮廓[<i>]的终点，横坐标值 用于工件的保护区，轮廓单元 1
$SC_PA_CONT_ABS[0,1]	40	轮廓[<i>]的终点，横坐标值 用于工件的保护区，轮廓单元 2
$SC_PA_CONT_ABS[0,2]	40	轮廓[<i>]的终点，横坐标值 用于工件的保护区，轮廓单元 3
$SC_PA_CONT_ABS[0,3]	0	轮廓[<i>]的终点，横坐标值 用于工件的保护区，轮廓单元 4
$SC_PA_CONT_ABS[0,4]	0	轮廓[<i>]的终点，横坐标值 用于工件的保护区，轮廓单元 5

系统变量	值	注释
$SC_PA_CONT_ABS[0,5]	0	轮廓[<i>]的终点，横坐标值 用于工件的保护区，轮廓单元 6
$SC_PA_CONT_ABS[0,6]	0	轮廓[<i>]的终点，横坐标值 用于工件的保护区，轮廓单元 7
$SC_PA_CONT_ABS[0,7]	0	轮廓[<i>]的终点，横坐标值 用于工件的保护区，轮廓单元 8
$SC_PA_CONT_ABS[0,8]	0	轮廓[<i>]的终点，横坐标值 用于工件的保护区，轮廓单元 9
$SC_PA_CONT_ABS[0,9]	0	轮廓[<i>]的终点，横坐标值 用于工件的保护区，轮廓单元 10
$SC_PA_CENT_ORD[0,0]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于工件的保护区，轮廓单元 1
$SC_PA_CENT_ORD[0,1]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于工件的保护区，轮廓单元 2
$SC_PA_CENT_ORD[0,2]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于工件的保护区，轮廓单元 3
$SC_PA_CENT_ORD[0,3]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于工件的保护区，轮廓单元 4
$SC_PA_CENT_ORD[0,4]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于工件的保护区，轮廓单元 5
$SC_PA_CENT_ORD[0,5]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于工件的保护区，轮廓单元 6
$SC_PA_CENT_ORD[0,6]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于工件的保护区，轮廓单元 7
$SC_PA_CENT_ORD[0,7]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于工件的保护区，轮廓单元 8
$SC_PA_CENT_ORD[0,8]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于工件的保护区，轮廓单元 9
$SC_PA_CENT_ORD[0,9]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于刀架的保护区，轮廓单元 10
$SC_PA_CENT_ABS[0,0]	0	圆弧轮廓[<i>]的中心，横坐标值 用于工件的保护区，轮廓单元 1
$SC_PA_CENT_ABS[0,1]	0	圆弧轮廓[<i>]的中心，横坐标值 用于工件的保护区，轮廓单元 2

系统变量	值	注释
$SC_PA_CENT_ABS[0,2]	0	圆弧轮廓[<i>]的中心，横坐标值 用于工件的保护区，轮廓单元 3
$SC_PA_CENT_ABS[0,3]	0	圆弧轮廓[<i>]的中心，横坐标值 用于工件的保护区，轮廓单元 4
$SC_PA_CENT_ABS[0,4]	0	圆弧轮廓[<i>]的中心，横坐标值 用于工件的保护区，轮廓单元 5
$SC_PA_CENT_ABS[0,5]	0	圆弧轮廓[<i>]的中心，横坐标值 用于工件的保护区，轮廓单元 6
$SC_PA_CENT_ABS[0,6]	0	圆弧轮廓[<i>]的中心，横坐标值 用于工件的保护区，轮廓单元 7
$SC_PA_CENT_ABS[0,7]	0	圆弧轮廓[<i>]的中心，横坐标值 用于工件的保护区，轮廓单元 8
$SC_PA_CENT_ABS[0,8]	0	圆弧轮廓[<i>]的中心，横坐标值 用于工件的保护区，轮廓单元 9
$SC_PA_CENT_ABS[0,9]	0	圆弧轮廓[<i>]的中心，横坐标值 用于工件的保护区，轮廓单元 10

用于刀架的通道专用保护区

系统变量	值	注释
$SC_PA_ACTIV_IMMED[1]	0	用于刀架的保护区不立即生效
$SC_PA_TW[1]	3	用于刀架的保护区与刀具相关
$SC_PA_ORI[1]	1	保护区的定向：1 = 第 3 和第 1 几何轴
$SC_PA_LIM_3DIM[1]	0	第 3 个 Dimension:0 = 没有限制
$SC_PA_PLUS_LIM[1]	0	Wert der Begrenzung in Plus-Richtung in der 3.维度的限制类型
$SC_PA_MINUS_LIM[1]	0	Wert der Begrenzung in Minus-Richtung in der 3.维度的限制类型
$SC_PA_CONT_NUM[1]	5	有效的轮廓单元的数目
$SC_PA_CONT_TYP[1,0]	1	轮廓类型[<i>] :1 = G1，直线， 用于刀架的保护区，轮廓单元 1
$SC_PA_CONT_TYP[1,1]	3	轮廓类型[<i>] :3 = G3，逆时针方向的圆弧单元， 用于刀架的保护区，轮廓单元 2

系统变量	值	注释
$SC_PA_CONT_TYP[1,2]	1	轮廓类型[<i>] :1 = G1，直线， 用于刀架的保护区，轮廓单元 3
$SC_PA_CONT_TYP[1,3]	1	轮廓类型[<i>] :1 = G1，直线， 用于刀架的保护区，轮廓单元 4
$SC_PA_CONT_TYP[1,4]	1	轮廓类型[<i>] :1 = G1，直线， 用于刀架的保护区，轮廓单元 5
$SC_PA_CONT_TYP[1,5]	0	轮廓类型[<i>] :0 = 未定义， 用于刀架的保护区，轮廓单元 6
$SC_PA_CONT_TYP[1,6]	0	轮廓类型[<i>] :0 = 未定义， 用于刀架的保护区，轮廓单元 7
$SC_PA_CONT_TYP[1,7]	0	轮廓类型[<i>] :0 = 未定义， 用于刀架的保护区，轮廓单元 8
$SC_PA_CONT_TYP[1,8]	0	轮廓类型[<i>] :0 = 未定义， 用于刀架的保护区，轮廓单元 9
$SC_PA_CONT_TYP[1,9]	0	轮廓类型[<i>] :0 = 未定义， 用于刀架的保护区，轮廓单元 10
$SC_PA_CONT_ORD[1,0]	-190	轮廓[<i>]的终点，纵坐标值 ；用于刀架的保护区，轮廓单元 1
$SC_PA_CONT_ORD[1,1]	-210	轮廓[<i>]的终点，纵坐标值 ；用于刀架的保护区，轮廓单元 2
$SC_PA_CONT_ORD[1,2]	-210	轮廓[<i>]的终点，纵坐标值 ；用于刀架的保护区，轮廓单元 3
$SC_PA_CONT_ORD[1,3]	0	轮廓[<i>]的终点，纵坐标值 ；用于刀架的保护区，轮廓单元 4
$SC_PA_CONT_ORD[1,4]	0	轮廓[<i>]的终点，纵坐标值 ；用于刀架的保护区，轮廓单元 5
$SC_PA_CONT_ORD[1,5]	0	轮廓[<i>]的终点，纵坐标值 ；用于刀架的保护区，轮廓单元 6
$SC_PA_CONT_ORD[1,6]	0	轮廓[<i>]的终点，纵坐标值 ；用于刀架的保护区，轮廓单元 7
$SC_PA_CONT_ORD[1,7]	0	轮廓[<i>]的终点，纵坐标值 ；用于刀架的保护区，轮廓单元 8
$SC_PA_CONT_ORD[1,8]	0	轮廓[<i>]的终点，纵坐标值 ；用于刀架的保护区，轮廓单元 9

系统变量	值	注释
$SC_PA_CONT_ORD[1,9]	0	轮廓[<i>]的终点，纵坐标值；用于刀架的保护区，轮廓单元 10
$SC_PA_CONT_ABS[1,0]	-50	轮廓[<i>]的终点，横坐标值；用于刀架的保护区，轮廓单元 1
$SC_PA_CONT_ABS[1,1]	-30	轮廓[<i>]的终点，横坐标值；用于刀架的保护区，轮廓单元 2
$SC_PA_CONT_ABS[1,2]	20	轮廓[<i>]的终点，横坐标值；用于刀架的保护区，轮廓单元 3
$SC_PA_CONT_ABS[1,3]	50	轮廓[<i>]的终点，横坐标值；用于刀架的保护区，轮廓单元 4
$SC_PA_CONT_ABS[1,4]	-50	轮廓[<i>]的终点，横坐标值；用于刀架的保护区，轮廓单元 5
$SC_PA_CONT_ABS[1,5]	0	轮廓[<i>]的终点，横坐标值；用于刀架的保护区，轮廓单元 6
$SC_PA_CONT_ABS[1,6]	0	轮廓[<i>]的终点，横坐标值；用于刀架的保护区，轮廓单元 7
$SC_PA_CONT_ABS[1,7]	0	轮廓[<i>]的终点，横坐标值；用于刀架的保护区，轮廓单元 8
$SC_PA_CONT_ABS[1,8]	0	轮廓[<i>]的终点，横坐标值；用于刀架的保护区，轮廓单元 9
$SC_PA_CONT_ABS[1,9]	0	轮廓[<i>]的终点，横坐标值；用于刀架的保护区，轮廓单元 10
$SC_PA_CENT_ORD[1,0]	0	圆弧轮廓[i]的中心，纵坐标值 用于刀架的保护区，轮廓单元 1
$SC_PA_CENT_ORD[1,1]	-190	圆弧轮廓[<i>]的中心，纵坐标值 用于刀架的保护区，轮廓单元 2
$SC_PA_CENT_ORD[1,2]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于刀架的保护区，轮廓单元 3
$SC_PA_CENT_ORD[1,3]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于刀架的保护区，轮廓单元 4
$SC_PA_CENT_ORD[1,4]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于刀架的保护区，轮廓单元 5
$SC_PA_CENT_ORD[1,5]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于刀架的保护区，轮廓单元 6

系统变量	值	注释
$SC_PA_CENT_ORD[1,6]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于刀架的保护区，轮廓单元 7
$SC_PA_CENT_ORD[1,7]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于刀架的保护区，轮廓单元 8
$SC_PA_CENT_ORD[1,8]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于刀架的保护区，轮廓单元 9
$SC_PA_CENT_ORD[1,9]	0	圆弧轮廓[<i>]的中心，纵坐标值 用于刀架的保护区，轮廓单元 10
$SC_PA_CENT_ABS[1,0]	0	圆弧轮廓[<i>]的中心，横坐标值 用于刀架的保护区，轮廓单元 1
$SC_PA_CENT_ABS[1,1]	-30	圆弧轮廓[<i>]的中心，横坐标值 用于刀架的保护区，轮廓单元 2
$SC_PA_CENT_ABS[1,2]	0	圆弧轮廓[<i>]的中心，横坐标值 用于刀架的保护区，轮廓单元 3
$SC_PA_CENT_ABS[1,3]	0	圆弧轮廓[<i>]的中心，横坐标值 用于刀架的保护区，轮廓单元 4
$SC_PA_CENT_ABS[1,4]	0	圆弧轮廓[<i>]的中心，横坐标值 用于刀架的保护区，轮廓单元 5
$SC_PA_CENT_ABS[1,5]	0	圆弧轮廓[<i>]的中心，横坐标值 用于刀架的保护区，轮廓单元 6
$SC_PA_CENT_ABS[1,6]	0	圆弧轮廓[<i>]的中心，横坐标值 用于刀架的保护区，轮廓单元 7
$SC_PA_CENT_ABS[1,7]	0	圆弧轮廓[<i>]的中心，横坐标值 用于刀架的保护区，轮廓单元 8
$SC_PA_CENT_ABS[1,8]	0	圆弧轮廓[<i>]的中心，横坐标值 用于刀架的保护区，轮廓单元 9
$SC_PA_CENT_ABS[1,9]	0	圆弧轮廓[<i>]的中心，横坐标值 用于刀架的保护区，轮廓单元 10

4.6.4 激活保护区

用于激活针对主轴卡盘、工件和刀架的保护区的零件程序节选：

程序代码	注释
`NPROT(1,2,0,0,0)`	; 保护区：主轴卡盘

程序代码	注释
CPROT(1,2,0,0,100)	; 保护区：工件在 z 轴上移动 100 mm。
CPROT(2,2,0,0,0)	; 保护区：刀架

4.7 数据表

4.7.1 机床数据

4.7.1.1 NC 专用机床数据

编号	名称：$MN_	说明
10618	PROTAREA_GEOAX_CHANGE_MODE	在坐标转换切换/跨通道取几何轴时的特性
18190	MM_NUM_PROTECT_AREA_NCK	可用的机床专用保护区的数量

4.7.1.2 通道专用机床数据

编号	名称：$MC_	说明
28200	MM_NUM_PROTECT_AREA_CHAN (SRAM)	可用的通道专用保护区的数量
28210	MM_NUM_PROTECT_AREA_ACTIVE	通道中可同时激活的保护区的最大数量
28212	MM_NUM_PROTECT_AREA_CONTUR	通道中的可定义轮廓单元的最大数量

4.7.2 信号

4.7.2.1 发送至通道的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
使能保护区	DB21, ...DBX1.1	DB320x.DBX1.1
进给禁止	DB21, ...DBX6.0	DB320x.DBX6.0
激活机床专用保护区 1 ... 8	DB21, ...DBX8.0 ... 7	DB320x.DBX8.0 ... 7
激活机床专用保护区 9	DB21, ...DBX9.0	DB320x.DBX9.0

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
激活机床专用保护区 10	DB21, ...DBX9.1	DB320x.DBX9.1
激活通道专用保护区 1 ... 8	DB21, ...DBX10.0 ... 7	DB320x.DBX10.0 ... 7
激活通道专用保护区 9	DB21, ...DBX11.0	DB320x.DBX11.0
激活通道专用保护区 10	DB21, ...DBX11.1	DB320x.DBX11.1

4.7.2.2 来自通道的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
机床专用保护区 1 ... 8 已预激活	DB21, ...DBX272.0 ... 7	DB330x.DBX8.0 ... 7
机床专用保护区 9 已预激活	DB21, ...DBX273.0	DB330x.DBX9.0
机床专用保护区 10 已预激活	DB21, ...DBX273.1	DB330x.DBX9.1
通道专用保护区 1 ... 8 已预激活	DB21, ...DBX274.0 ... 7	DB330x.DBX10.0 ... 7
通道专用保护区 9 已预激活	DB21, ...DBX275.0	DB330x.DBX11.0
通道专用保护区 10 已预激活	DB21, ...DBX275.1	DB330x.DBX11.1
机床专用保护区 1 ... 8 已超出	DB21, ...DBX276.0 ... 7	DB330x.DBX12.0 ... 7
机床专用保护区 9 已超出	DB21, ...DBX277.0	DB330x.DBX13.0
机床专用保护区 10 已超出	DB21, ...DBX277.1	DB330x.DBX13.1
通道专用保护区 1 ... 8 已超出	DB21, ...DBX278.0 ... 7	DB330x.DBX14.0 ... 7
通道专用保护区 9 已超出	DB21, ...DBX279.0	DB330x.DBX15.0
通道专用保护区 10 已超出	DB21, ...DBX279.1	DB330x.DBX15.1

B1: 连续路径运行、准停、预读

5

5.1 简要说明

准停或准停运行

准停是一种运行模式，在该模式下每个运行程序段结束时，所有参与运动的轴（除了跨程序段运行的轴）将制动至静止状态。只有当所有参与运动的轴都根据所选的准停标准到达编程的目标位置后，NC 才会切换至下一个程序段。

连续路径运行

连续路径运行是一种运行模式，在该模式下，NC 会尽可能保持编写的轨迹速度恒定不变。尤其是在需要避免轨迹轴在零件程序程序段交界处制动的情况下。

预读

预读是一个用于优化连续路径运行的功能。

高质量的工件表面要求进行均匀加工，因此在加工期间应尽量避免轨迹速度的波动。没有预读功能时，NC 只读取紧跟着当前程序段的后一条程序段，以确定可能的轨迹速度。如果该后续程序段只包含一段很短的行程，则 NC 必须降低轨迹速度，即在当前程序段内就进行制动，以便及时地在后续程序段结束时停止。

使用预读功能，NC 可以预先读取当前程序段后面一定数量的后续程序段，在一些条件下可以显著提高轨迹速度，因为现在 NC 可以提供更多的程序段或更长的行程供计算。

使用该功能的优点有：

- 平均轨迹速度更高
- 减少制动和加速过程，工件表面质量更佳

轨迹速度平滑

“轨迹速度平滑”是一项针对对均匀轨迹速度有高要求的应用而开发的功能，比如：模具制造上的高速铣削。为此，在平滑轨迹速度时会舍弃一些易引起机床高频共振的制动和减速过程。

使用该功能的优点有：

- 避免了机床共振，工件表面质量更佳，加工时间更短
- 避免了多余的加速过程，即对程序处理时间没有很大益处的加速过程，轨迹速度或切削速度更均匀稳定。

轨迹动态响应自适应

除了“轨迹速度平滑”外，“轨迹动态响应自适应”是另一项用于避免机床高频共振并可同时优化轨迹动态响应的功能。为此，相比于机床数据设置的原始动态响应极限值，轨迹速度的高频变化会自动采用更低的急动值或加速度值。

因此在轨迹速度低频变化时原始动态响应极限值生效，而在轨迹速度高频变化时 NC 会自动调整动态响应，低动态响应极限值生效。

轨迹插补的动态响应模式

在优化轨迹动态响应方面，工艺专用的动态响应设置也可产生很大影响，它针对不同的工况预设（如攻丝、粗加工、精加工等），可以通过在零件程序中调用对应的动态响应模式来激活。

任意形状表面模式

曲率或挠度的每次波动都会引起轨迹速度的变化，因此，通常在加工表面为任意形状的工件时，会产生一些多余的制动和加速过程，这些过程常常会降低工件的表面质量。

所以，针对该表面加工系统提供以下功能：

- 功能“任意形状表面模式：基本功能”
  该功能可以使轨迹速度的变化相对于曲率和挠度的变化变得“迟钝”。
- 功能“任意形状表面模式：扩展功能”
  该功能是预读标准功能的扩展，可以在加工任意形状表面时计算轨迹速度曲线。

任意形状表面模式的优点在于，首先工件表面更均匀，其次机床承受的负载更轻。

线性程序段的压缩

在用 CAD/CAM 系统设计完工件后，还需要使用该系统创建用于生成工件表面的零件程序。大多数 CAD/CAM 系统也采用线性程序段来描述弯曲的工件表面。为达到所需的轮廓精度，通常该过程需要大量的插补点，从而产生了大量短行程线性程序段。

通过使用“压缩器功能”，可以事后借助多项式程序段来近似由线性程序段设定的轮廓。因此，多条线性程序段被一条多项式程序段取代。

优点：

- 减少运行程序段的数量
- 提升轨迹速度
- 提升表面质量
- 程序段过渡稳定

压缩较短的样条程序段

样条定义一个由 2 阶或 3 阶多项式合并的曲线。通过样条插补，控制系统只需少数几个目标轮廓插补点便可生成一条光滑的曲线。

相比于线性插补而言，样条插补的优点有：

- 减少了描述曲面轮廓所需零件程序段的数目
- 曲线更加“软”，保护机械装置，程序段过渡处也是如此。

相比于线性插补而言，样条插补的缺点有：

- 对于样条曲线无法设置精确曲线形状，而只能设置样条曲线所在的公差带。

和线性插补一样，在处理样条时也会产生短程序段，插补这些短样条程序段时必须降低轨迹速度。就算样条是一条平滑的长曲线时也是如此。通过功能“压缩较短的样条程序段”可以合并样条程序段，从而产生足够长的程序段并避免降低轨迹速度。

5.2 准停运行

准停或准停运行

准停或准停运行是一种运行模式，在该模式下每个运行程序段结束时，所有参与运动的轨迹轴和辅助轴（除了跨程序段运行的轴）将制动至静止状态。只有当所有参与运动的轴都根据所选的准停条件到达编程的目标位置后，NC 才会切换至下一个程序段。

从而产生以下特性：

- 由于所有参与运动的机床轴都需要先制动，然后等待进入“准停”状态，所以相对于连续路径运行，程序运行时间会大大延长。
- 如果在加工期间进行准停运行，会在工件表面上留下加工灼痕。

准停条件

可设置下列准停条件：

- “粗准停”
- “精准停”
- “插补终点”

应用

如果需要精确地加工轮廓，则需要使用准停运行。

激活

在程序中，可通过下列指令模态或逐段设定准停运行：

指令	含义
G60	准停运行自当前程序段起**模态**生效。
G9	准停运行在当前程序段中生效。

准停条件：“粗准停”和“精准停”

在机床轴的当前跟随误差小于等于以下机床数据中设置的围绕设定位置的公差范围的情况下，便满足准停条件“粗”或“精”。

- MD36000 $MA_STOP_LIMIT_COARSE（粗准停）
- MD36010 $MA_STOP_LIMIT_FINE（精准停）

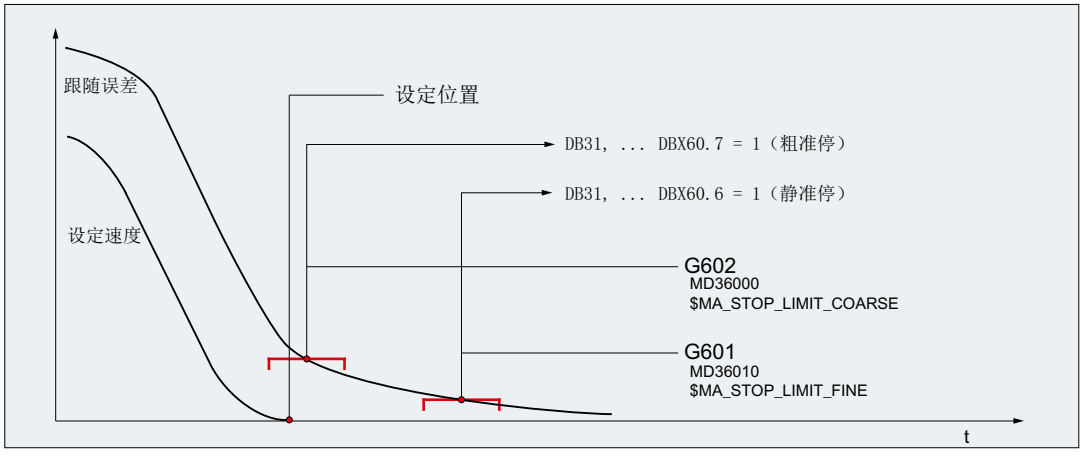


图 5-1　　准停条件的公差带

说明

在设置准停条件“粗准停”和“精准停”的公差带时，需满足以下条件：

粗准停 > 精准停

准停条件：“插补结束”

选择准停条件“插补结束”时，一旦所有参与运动的轨迹轴和辅助轴（除了跨程序段运行的轴外）都达到了在运行程序段中编写的位置，便立即切换到下一程序段。即插补器已经处理完该程序段。

该准停条件“插补结束”不考虑机床轴的实际位置或跟随误差。因此，与准停条件“粗准停”/“精准停”相比，程序段过渡处的轮廓更加平滑，平滑幅度取决于机床轴的动态响应。

激活可编程的准停条件

通过下列指令激活可编程的准停条件：

指令	准停条件
G601	精准停
G602	粗准停
G603	插补结束

根据生效的准停条件进行程序段切换

下图示出取决于所选择的准停条件的程序段切换时间点。

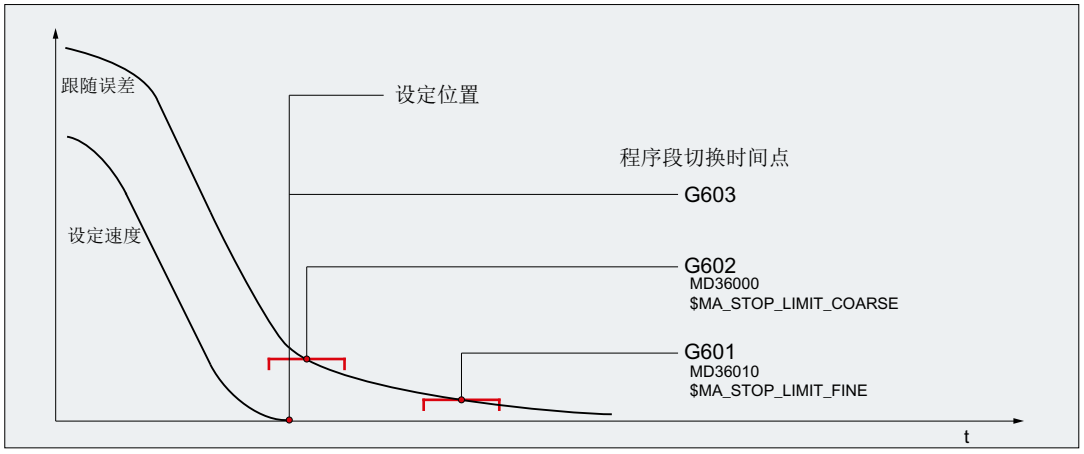


图 5-2　　根据生效的准停条件进行程序段切换

准停条件的权重系数

可通过以下轴专用机床数据设定准停条件的取决于参数组的权重：

MD36012 $MA_STOP_LIMIT_FACTOR[<参数组>] = <值>

权重系数会影响下列机床数据的值：

- MD36000 $MA_STOP_LIMIT_COARSE
- MD36010 $MA_STOP_LIMIT_FINE
- MD36030 $MA_STANDSTILL_POS_TOL

应用示例

- 根据新的质量比来调整定位方式，比如：在切换齿轮档后。
- 根据不同的运行状态（粗加工或精加工）来缩短定位时间。

设定生效的准停条件

针对 G 功能组 1 的准停条件可固定设定。这使得编写的准停条件无效。

可**相互独立**地为下列指令设置准停条件：

- 快速移动 G0
- G 功能组 1 的所有其他指令

通过以下十进制编码机床数据针对特定通道设置准停条件：

MD20550 $MC_EXACT_POS_MODE = <Z><E>

Z	E	生效的准停条件
0	0	编写的准停条件
1	1	G601（精准停窗口）
2	2	G602（粗准停窗口）
3	3	G603（插补结束）
E（个位）：针对快速移动的准停条件设置。 Z（十位）：针对 G 功能组 1 的所有其他指令的准停条件设置。		

示例

MD20550 $MC_EXACT_POS_MODE = 02

- **<E> = 2**：执行快速移动时，不论如何编程，始终是准停条件“G602”（粗准停窗口）生效。
- **<Z> = 0**：在以 G 功能组 1 的所有其他指令运行时，编写的准停条件生效。

连续路径运行中 G0 ↔ 非 G0 程序段切换时的程序段切换特性

在连续路径运行中，可通过以下机床数据设置在快速移动与非快速移动（G0 ↔ 非 G0）程序段之间切换时的特性：

MD20552 $MC_EXACT_POS_MODE_G0_TO_G1 = <值>

<值>	含义
0	在程序段过渡处没有额外的停止。
1	在程序段过渡处停止。 特性对应 G601（精准停窗口）
2	在程序段过渡处停止。 特性对应 G602（粗准停窗口）
3	在程序段过渡处停止。 特性对应 G603（插补结束）

<值>	含义
4	在程序段过渡处**不**停止。 在连续路径运行中，在 **G0 ↔ 非 G0 程序段切换**时，在 G0 程序段中预先将随后之非 G0 程序段的**进给补偿**的当前值考虑在内。根据轴动态特性以及当前程序段的轨迹长度，以后续程序段的经精确或及尽可能最佳调整的速度执行程序段切换。
5	在程序段过渡处**不**停止。 在连续路径运行中，在 **G0 → 非 G0** 和 **非 G0 → G0 程序段切换**时，预先将后续程序段的**进给补偿**（G0 → 非 G0）或**快速移动补偿**（非 G0 → G0）的当前值考虑在内。根据轴动态特性以及当前程序段的轨迹长度，以后续程序段的经精确或及尽可能最佳调整的速度执行程序段切换。

5.3 连续路径运行

5.3.1 一般功能

连续路径运行

在连续路径运行中，结束一条程序段而切换到另一条程序段时，轨迹速度**不会**为了达到精准停标准而降低到很小。目标是，在程序段切换点处避免轴停止加工，尽可能以相同的轨迹速度转到下一个程序段。 为了实现此目标，激活连续路径运行时还应激活“预读”功能。

连续路径运行通过局部修改编程曲线，使原本突兀曲折的程序段过渡更加平滑、圆顺。 对编程曲线的修改幅度可以由过载系数标准或平滑加以限制。

通过连续路径运行可以实现：

- 轮廓圆顺。
- 省去了达到准停标准所需的制动和加速过程， 从而缩短了加工时间。
- 平缓的速度变化，获得良好的切削质量

在下列情形下建议使用连续路径运行：

- 需要尽可能快速地加工轮廓（比如通过快速移动）。
- 实际运行与编程的运行允许有所偏差（在设定的公差内），以使运行保持连续、稳定。

在下列情形下，不建议使用连续路径运行：

- 要求精确加工轮廓。
- 要求绝对恒定速度。

隐含的准停功能

在某些情形下，在连续路径加工方式中需要执行准停功能，以便进行后续操作。 这时轨迹速度被降到零。

- 辅助功能指令如果在轴运行前给出，则前一个程序段只有在到达所选择的准停标准后才结束。
- 辅助功能指令如果在轴运行后给出，则在程序段插补结束后输出辅助功能。
- 如果可执行程序段（比如：启动一根定位轴）中不包含任何轨迹轴的运行信息，当符合所选择的准停标准时，前一个程序段结束。
- 如果定位轴被申明为“几何轴”并在程序中写入了该几何轴，则当插补结束时，前一个程序段结束。
- 如果程序中写入了一根同步轴，该轴最后又曾作为定位轴或主轴写入程序（辅助轴默认定义为定位轴），则当插补结束时，前一个程序段结束。
- 如果切换了坐标转换，则达到当前生效的准停标准后，切换前的程序段结束。
- 如果在下一个程序段中包含了加速特性 BRISK/SOFT 的转换，则当前程序段在插补结束时结束，参见章节“B2：加速度 (页 285)”。
- 如果在零件程序中写入了“清空缓冲存储器”功能，则前一个程序段在到达所选择的准停标准后才结束。

在连续路径加工方式下速度=0

在下列情形下，在程序段结束时加工速度被制动为零，不管是否隐含准停功能：

- 通过指令 POS 写入了定位轴，该轴的运行时间长于轨迹轴的运行时间。 在此情况下，只有在定位轴达到“准停”后才进行程序段的切换。
- 通过指令 SPOS 写入的主轴的定位时间长于轨迹轴的运行时间。在此情况下，只有在定位主轴“精准停”后才进行程序段的切换。
- 在当前程序段中运行的是几何轴，但在下一程序段中运行的是同步轴；或在当前程序段中运行的是同步轴，而在下一个程序段中运行的是几何轴。
- 需要进行同步。

5.3.2 按过载系数降低速度

功能

此功能可以在连续路径方式下降低轨迹速度，以便非相切程序段过渡可以在一个插补周期中完成，并保持加速度极限和考虑过载系数。

如果程序段过渡处的轮廓不是相切轮廓，减速会导致轴速度突然改变。 该现象也可以从一同运行的同步轴上观察得出。 速度跃变可避免将进给速度降到零。如果当轴速度降低到某一速度值并从该速度值进行加速，从而可以到达新的速度设定值时，则发生速度跃变。 可以使用过载系数来控制速度跃变值。速度跃变值是与轴相关的，因此在程序段切换时考虑的是当前有效轴的速度跃变值。

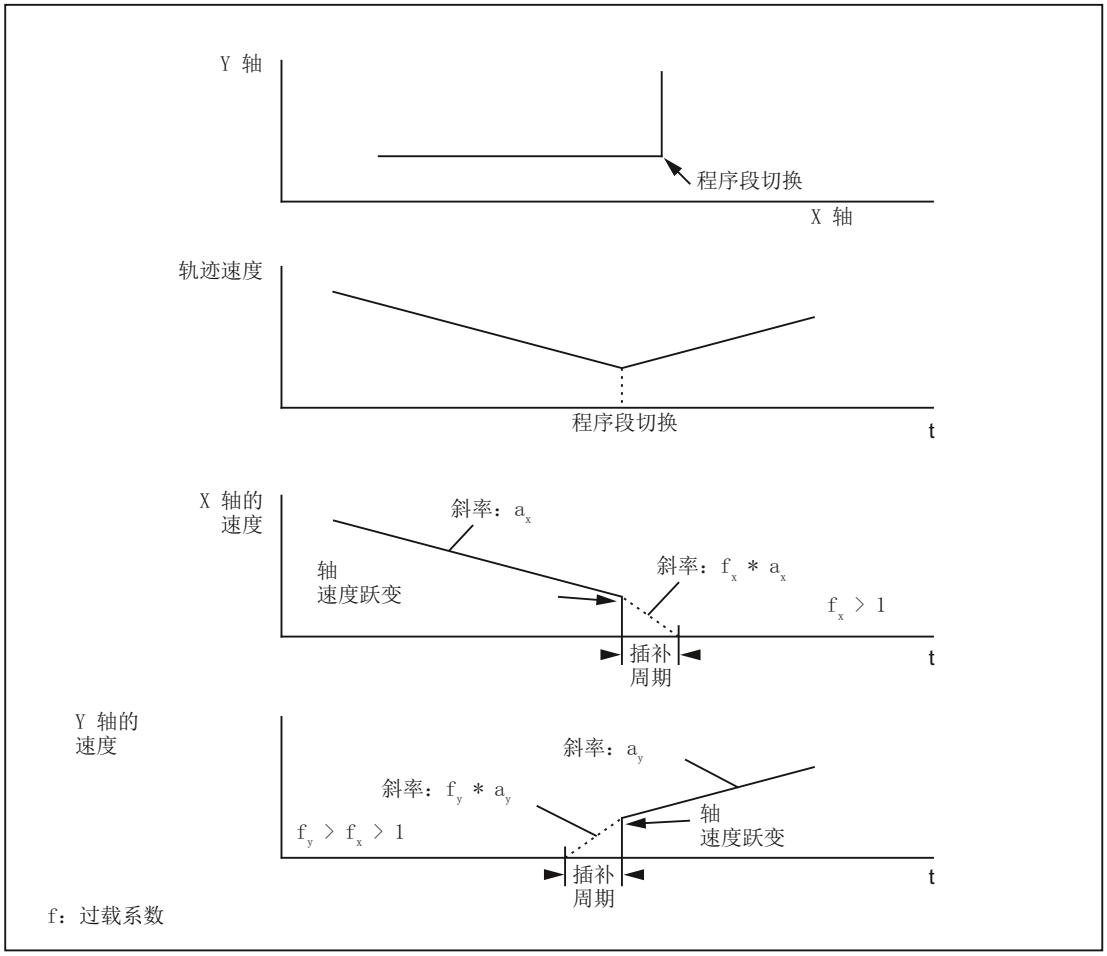


图 5-3 程序段切换处的轴速度变化

对于几乎是相切的程序段切换，如果未超过允许的轴加速度，则不降低轨迹速度。 因此可以对轮廓进行很小角度（比如：0.5°）的超程加工。

过载系数

过载系数限制机床轴在程序段切换时的速度跃变。为了确保速度跃变不超出轴负载，跃变值通过轴的加速度计算得到。

过载系数定义了机床轴的加速度在一个插补周期内可以超出极限的幅度，此加速度值在 MD32300 $MA_MAX_AX_ACCEL 中定义。

速度跃变值的计算公式为：

速度跃变值 = 轴加速度 * (过载系数 -1) * 插补周期。

在以下机床数据中可以对过载系数进行调整：

MD32310 $MA_MAX_ACCEL_OVL_FACTOR（轴速度跃变的过载系数）

系数 1.0 表示只能按限定系数进行相切程序段切换。 对于任何其它的切换，速度将根据设定值降为 0。 其特性相当于“准停标准：插补结束”。 由于该方式对于连续路径运行而言不理想，因此应设置大于 1.0 的系数。

说明

在调试时注意，当在突兀的程序段过渡处机床振动而不希望使用“平滑”功能时，应降低该系数。

设置以下机床数据，可以对程序段过渡进行平滑，不管当前设置的过载系数 G641 / G642：

MD20490 $MC_IGNORE_OVL_FACTOR_FOR_ADIS

激活/取消激活

在每个数控程序段中都可以通过模态指令 G64 激活“连续路径运行 + 按过载系数降低速度”。

可以选择非模态准停指令 G9 来中断连续路径运行。

或者选择以下某个指令来撤销连续路径运行 G64：

- 模态准停 G60
- 平滑 G641, G642, G643, G644 或 G645

隐含的连续路径运行

如果由于程序段行程过短（比如：零周期程序段）而导致无法在 G641 中插入平滑程序段，则切换到 G64。

5.3.3 平滑

功能

“平滑”功能会沿着编写的轮廓（轨迹轴）在非圆顺（弯折）程序段过渡处插入中间程序段（平滑程序段），由此便可产生圆顺（切向）的新程序段过渡。

同步轴

除几何轴外，平滑还会考虑所有的同步轴。但在同时运行轨迹轴和同步轴时，两种类型的轴可能无法同时生成一个圆顺的程序段过渡。由于轨迹轴的运行精度始终高于同步轴，因此，只能生成一个接近圆顺的程序段过渡。

G64 平滑

如果为了遵循程序段过渡时的动态限值，要求的速度低于 G64 允许的程序段过渡速度，则也可以使用平滑功能（参见 “按过载系数降低速度 (页 212)” 一章中的过载系数）。

对同步条件的影响

在进行平滑时，系统会缩短编写的程序段，这些程序段之间插入了一个或多个平滑轮廓。编写的程序段分界被取消，不再可用作同步条件的标准，比如：和运动、停止一同输出的辅助功能。

说明

在使用“平滑”功能时，同步条件针对的都是平滑位置前的程序段的末尾，而不是插入的平滑程序段的末尾。因此，下一条程序段没有开始，在停止在程序段末尾时，还可以手动修改下一条程序段的轮廓。

例外情况

在以下程序段过渡中，此处以从 N10 过渡带到 N20 为例，无平滑，即不会插入任何平滑程序段：

隐性停止运行

可能的原因：

- N20 运行前辅助功能输出生效
- N20 不包含轨迹轴运行动作
- 一个轴之前是定位轴，但在 N20 中首次作为轨迹轴运行
- 一个轴之前是轨迹轴，但在 N20 中首次作为定位轴运行

- N10 中运行几何轴，而 N20 中不运行
- N20 中运行几何轴，而 N10 中不运行
- 激活 N20 中的螺纹切削 G33
- 在 BRISK 和 SOFT 之间进行切换
- 对坐标转换非常重要的轴没有完全分配到轨迹运动 （比如摆动轴、定位轴）。

插入平滑程序段会使零件程序运行速度过渡减慢

可能的原因：

- 一个程序或程序段是由大量很短的运行程序段组成的（≈ 1 个插补周期 / 运行程序段；由于每个运行程序段至少需要一个插补周期，所以插入的中间程序段使运行时间加倍）
- 进行程序段切换时，不减速的 G64（不带平滑的连续路径运行）生效
- 编写的过载系数 (MD32310 $MA_MAX_ACCEL_OVL_FACTOR) 可实现在不减小轨迹速度的情况下运行出编写的轮廓。另见：
  MD20490 $MC_IGNORE_OVL_FACTOR_FOR_ADIS

轨迹参数忽略平滑

可能的原因：

- G641（符合行程标准的带平滑连续路径运行）生效，快速移动也生效 (G0) UND ADISPOS == 0（G0 时的平滑间距）
- G641（符合行程标准的带平滑连续路径运行）生效，但快速移动未生效 UND ADIS == 0（轨迹功能 G1、G2、G3 ... 的平滑间距）
- G642 或 G643（遵循指定公差的带平滑连续路径运行）生效，但所有公差 == 零

N10 或 N20 不包含运行动作（零程序段）

通常不会生成零程序段。例外情况：

- 同步动作激活时
- 程序跳转

对同步条件的影响

在进行平滑时，系统会缩短编写的程序段，这些程序段之间插入了平滑轮廓。最初编写的程序段分界被取消，不再可用于同步条件，比如：和运动、停止一同输出的辅助功能。

说明

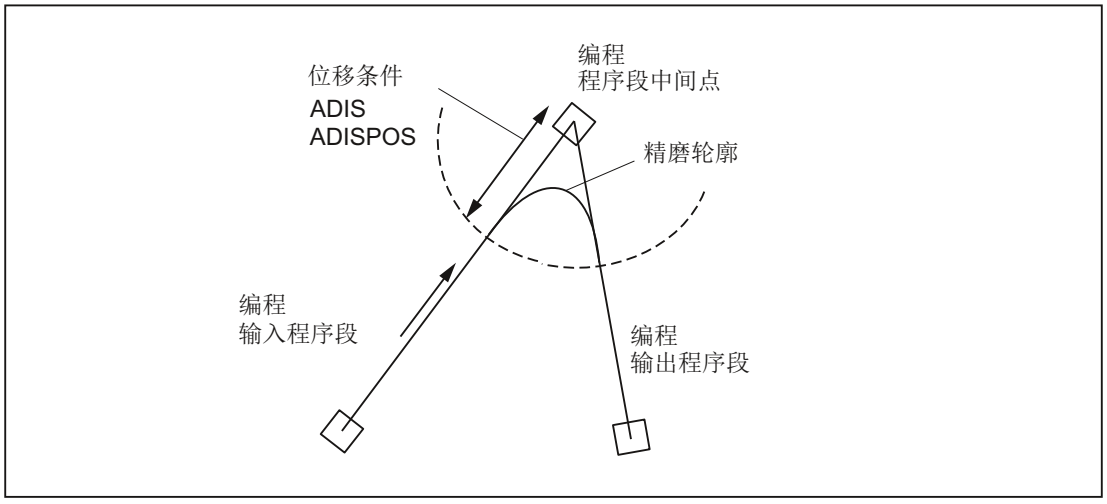
在使用“平滑”功能时，同步条件针对的都是平滑位置前的程序段的末尾，而不是插入的平滑程序段的末尾。因此，下一条程序段没有开始，在停止在程序段末尾时，还可以修改下一条程序段的轮廓。

5.3.3.1 按照位移条件开展平滑(G641)

功能

在按照位移条件开展平滑的连续路径运行中，平滑区域的大小是由位移条件 ADIS 和 ADISPOS 决定的。

位移条件 ADIS 或 ADISPOS 描述了程序段结束前平滑程序段最早可以开始的距离，或者程序段结束后必须结束的距离。



说明

尖角会产生特别弯曲的平滑曲线，从而导致减速。

说明

ADISPOS 和 ADIS 处理方式相同，但只能用于快速移动 G0。

位移条件的有效性

- 必须编程 ADIS 或 ADISPOS。 保持缺省值“零”时，G641 和 G64 的特性一样。
- 如果两个相连程序段不是快速移动 G0，则较短的平滑距离生效。
- 如果 ADIS 的值过小，要注意，控制系统会确保每个被插补的程序段（即使是一个平滑中间程序段）也至少具有一个插补点。 因此，最大的轨迹速度被限制在“ADIS/插补周期”以下。
- 不管是 ADIS 还是 ADISPOS，平滑区域都是通过程序段位移长度来限制的。
  在短行程程序段中（行程小于 4 倍的 ADIS 或 4 倍的 ADISPOS），平滑距离会被缩短，使原始程序段仍能保持一段可移动行程。 剩余长度由轴特性决定，大概是轴待移动行程的 60%。 ADIS 或 ADISPOS 因此被缩短为待移动行程的 40%! 该算法还可避免在非常细微的轮廓变化中插入平滑程序段。 此时系统会一直切换到连续路径运行 G64，直到可以再次插入平滑程序段。

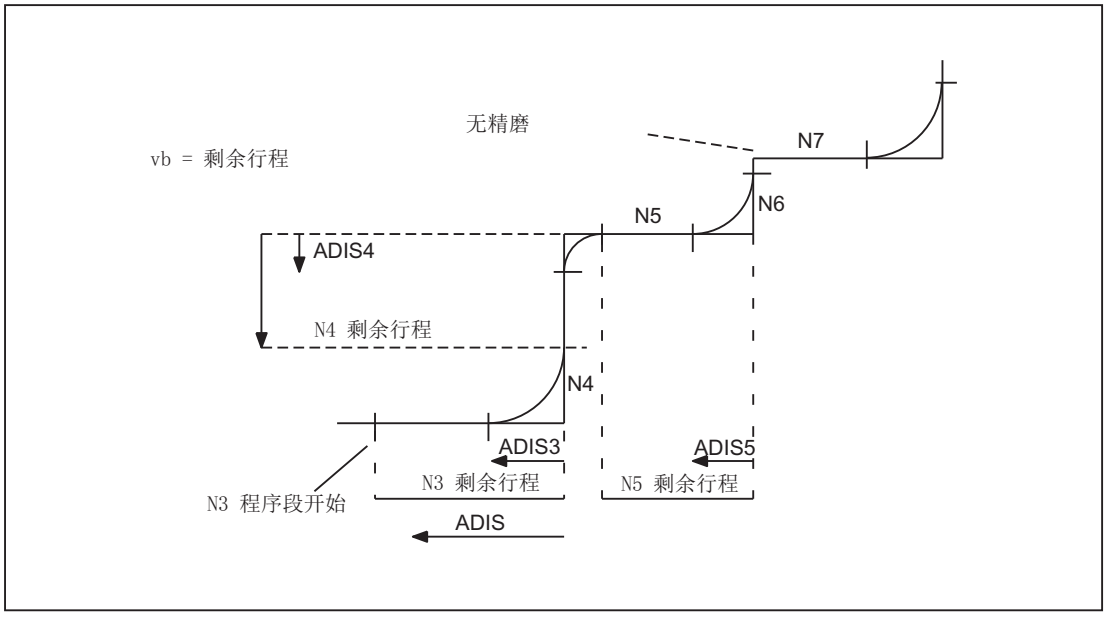


图 5-4　　含 ADIS 限制的轨迹曲线

激活/取消激活

在每个数控程序段中都可以通过模态指令 G641 激活按照位移条件开展平滑的连续路径运行。 在激活前或激活时应设定位移条件 ADIS/ADISPOS。

可以选择非模态准停指令 G9 来中断连续路径运行。

或者选择以下某个指令来撤销 G641：

- 模态准停 (G60)
- 连续路径运行 G64, G642, G643, G644 或 G645

程序示例

程序代码	注释
N1 G641 Y50 F10 ADIS=0.5	; 连续路径运行，根据位移条件进行平滑（平滑距离： 0.5 mm）。
N2 X50	
N3 X50.7	
N4 Y50.7	
N5 Y51.4	
N6 Y51.0	
N7 X52.1	

5.3.3.2 按照定义的公差开展平滑(G642/G643)

功能

在按照定义的公差开展平滑的连续轨迹运行中，通常平滑按照最大允许的轨迹公差进行。该轴专用公差也可通过最大轮廓偏差（轮廓公差）或者刀具定向的最大角度偏差（定向公差）来取代。

激活

在每个数控程序段中都可以通过模态指令 G642 或 G643 激活按照定义的公差开展平滑的连续轨迹运行。

可以选择非模态准停指令 G9 来中断连续路径运行。

或者选择以下某个指令来撤销 G642/G643：

- 模态准停 (G60)
- 连续路径运行 G64, G641, G644 或 G645

G642 和 G643 之间的区别

G642 和 G643 在平滑特性上有以下区别：

G642	G643
G642 中，平滑距离由所有轴最短的平滑距离决定。在生成平滑程序段时会考虑该值。	G643 中，各轴的平滑距离可以有所不同。程序段内部会考虑各轴的平滑距离（⇒ 不是各自的平滑距离）。
在 G642 中，平滑区域由最小的公差设定得出。	如果轮廓公差和刀具定向公差的设定数据区别很大，则只在 G643 中生效。

参数设置

最大轨迹公差

G642/G643 平滑中最大允许的轨迹公差在以下机床数据中为各个轴单独设定：

MD33100 $MA_COMPRESS_POS_TOL

轮廓公差和定向公差

轮廓公差和定向公差在通道专用设定数据中设置：

SD42465 $SC_SMOOTH_CONTUR_TOL（最大轮廓偏差）

SD42466 $SC_SMOOTH_ORI_TOL（刀具定向最大角度偏差）

设定数据可在程序中编程，因此为各程序段过渡分别设定。

说明

设定数据 SD42466 $SC_SMOOTH_ORI_TOL 只有在定向转换激活时生效。

平滑特性

G642/G643 的平滑特性由以下机床数据设置：

MD20480 $MC_SMOOTHING_MODE（G64x 平滑特性）

个位(**E**) 定义 G643 特性，十位(**Z**) 定义 G642 特性：

E 或 Z 的值	含义
0	**所有轴：** 按照允许的最大轨迹公差开展平滑： MD33100 $MA_COMPRESS_POS_TOL
1	**几何轴：** 按照轮廓公差开展平滑： SD42465 $SC_SMOOTH_CONTUR_TOL **其他轴：** 按照允许的最大轨迹公差开展平滑： MD33100 $MA_COMPRESS_POS_TOL
2	**几何轴：** 按照定向公差开展平滑： SD42466 $SC_SMOOTH_ORI_TOL **其他轴：** 按照允许的最大轨迹公差开展平滑： MD33100 $MA_COMPRESS_POS_TOL
3	**几何轴：** 按照定向公差和轮廓公差开展平滑： SD42465 $SC_SMOOTH_CONTUR_TOL SD42466 $SC_SMOOTH_ORI_TOL **其他轴：** 按照允许的最大轨迹公差开展平滑： MD33100 $MA_COMPRESS_POS_TOL
4	**所有轴：** 使用 ADIS 或 ADISPOS 编程的平滑长度，和 G641 一样。 轴专用公差、轮廓公差或定向公差的设定会被忽略。

极限速度的特性

按照定义公差进行平滑时的速度特性是由 MD20480 的百位确定的：

值	含义	
< 100:	在平滑区域内，会计算极限速度的特性，它是由设定的各轴/轨迹最大加速度和最大急动度得出的。 这种计算有可能导致轨迹速度在平滑区域内提升，并由此导致参与轴加速。	
≥100:	在 G641/G642 的平滑程序段中，不计算极限速度的特性。 只确定恒定的极限速度。 它可以避免在 G641/G642 平滑中参与轴在平滑区域内加速。 但是在一些条件下，尤其是平滑区域很大时，该设置会导致平滑程序段的处理速度过慢。	
	1xx:	G641 无速度特性
	2xx:	G642 无速度特性

说明

参见 MD28530 $MC_MM_PATH_VELO_SEGMENTS（存储单元的数量，用于限制轨迹速度）

前提条件

在半径补偿和刀具定向激活时保护区的局限性：

对于不垂直于基本坐标系中某个基准面的刀具定向而言，它虽然考虑半径补偿，但是保护区不会转入对应的平面。

G643 中必须作如下设置：

MD28530 $MC_MM_PATH_VELO_SEGMENTS > 0（存储单元的数量，用于限制轨迹速度）

如果不满足该条件，则必须为所有轴设置：

MD35240 $MC_ACCEL_TYPE_DRIVE = FALSE（加速特性曲线 DRIVE 开/关）

5.3.3.3 按照允许的最大动态响应进行平滑(G644)

功能

在该带平滑的连续路径运行中，允许的最大动态响应是重要因素。

说明

只有在以下条件下，才允许 G644 平滑：

- 在两个被观察的程序段中所有参与轴只包含线性运动。
- **没有**坐标转换功能生效。

如果一根参与轴包含多项式（编程了多项式、样条或压缩器）或生效的坐标转换，则使用 G642 进行平滑。

激活

在每个数控程序段中都可以通过模态指令 G644 激活按照允许的最大动态响应开展平滑的连续路径运行。

可以选择非模态准停指令 G9 来中断连续路径运行。

或者选择以下某个指令来撤销 G644：

- 模态准停 (G60)
- 连续路径运行 G64, G641, G642, G643 或 G645

参数设置

G644 的平滑特性由以下机床数据的千位和万位设置：

MD20480 $MC_SMOOTHING_MODE（G64x 平滑特性）

值	含义
千位：	
0xxx:	G644 平滑按照以下数据设定的每轴最大公差进行： MD33100 $MA_COMPRESS_POS_TOL 如果轴的动态响应允许，有时不会完全利用设置的公差。
1xxx:	通过编程 ADIS=…或 ADISPOS=…设定最大的平滑距离，同 G641。

值	含义
2xxx:	通过以下机床数据设置平滑区域内每轴的最大平滑频率： MD32440 $MA_LOOKAH_FREQUENCY （预读时的平滑频率） 确定合适的平滑范围，使平滑频率不会超出设置的最大频率。
3xxx:	在拐角处会出现速度跃变的每轴会以最大允许的动态响应（最大加速度和最大急动度）绕行拐角。 SOFT: 在使用 SOFT 时，会限制每轴的最大加速度**以及**最大急动度。 BRISK: 在使用 BRISK 时，只限制每轴的最大加速度，**不**限制最大急动度。 在该设置中，即不监控最大可能出现的公差也不监控最大平滑距离。 公差或平滑距离仅仅由各轴的动态极限值和当前轨迹速度得出。
4xxx:	同 0xxx ，按照以下数据设定的每轴最大公差进行平滑： MD33100 $MA_COMPRESS_POS_TOL 和 0xxx 不同之处在于，在可能条件下会充分利用设置的公差。 因此，轴不会达到最大可能的动态响应。
5xxx:	同 1xxx，通过编程 ADIS=…或 ADISPOS=…设定最大的平滑距离。 和 1xxx 不同之处在于，在可能条件下会充分利用设定的平滑距离。 因此，参与轴不会达到最大可能的动态响应。
万位	
0xxxx	采用 BRISK 时，平滑区域内轴的速度特性不含急动度限制，采用 SOFT 时，含急动度限制，
1xxxx	不管采用 BRISK 或是 SOFT，平滑区域内轴的速度特性都会含急动度限制。

在设定了最大轴公差 (MD33100 $MA_COMPRESS_POS_TOL) 或最大平滑距离(ADIS/ADISPOS) 时，如果参与轴的动态响应允许，通常不会充分利用平滑距离。 因此，平滑距离的长度由当前轨迹进给率决定。 在轨迹速度较低时，和编程轮廓的偏差较小。 但也可以作其他设定，即此时尽可能充分利用设定的最大公差或最大平滑距离。 这样和编程轮廓的公差便与编程的轨迹进给率毫无关联。

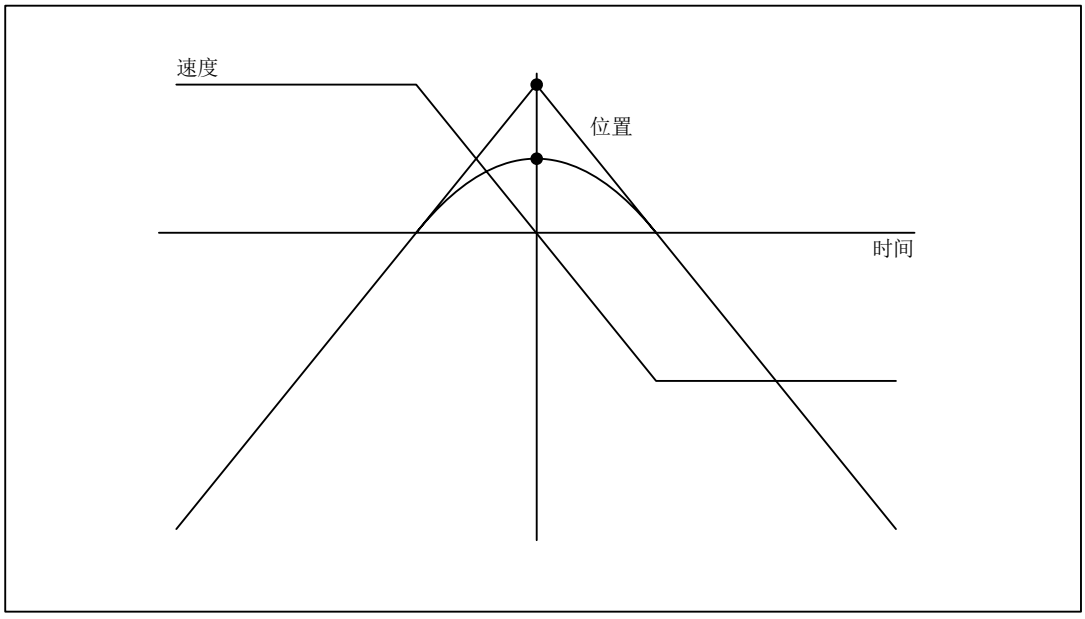
说明

另外还可以激活以下附加的限制：
平滑距离最大是原始程序段行程的一半。

急动度限制

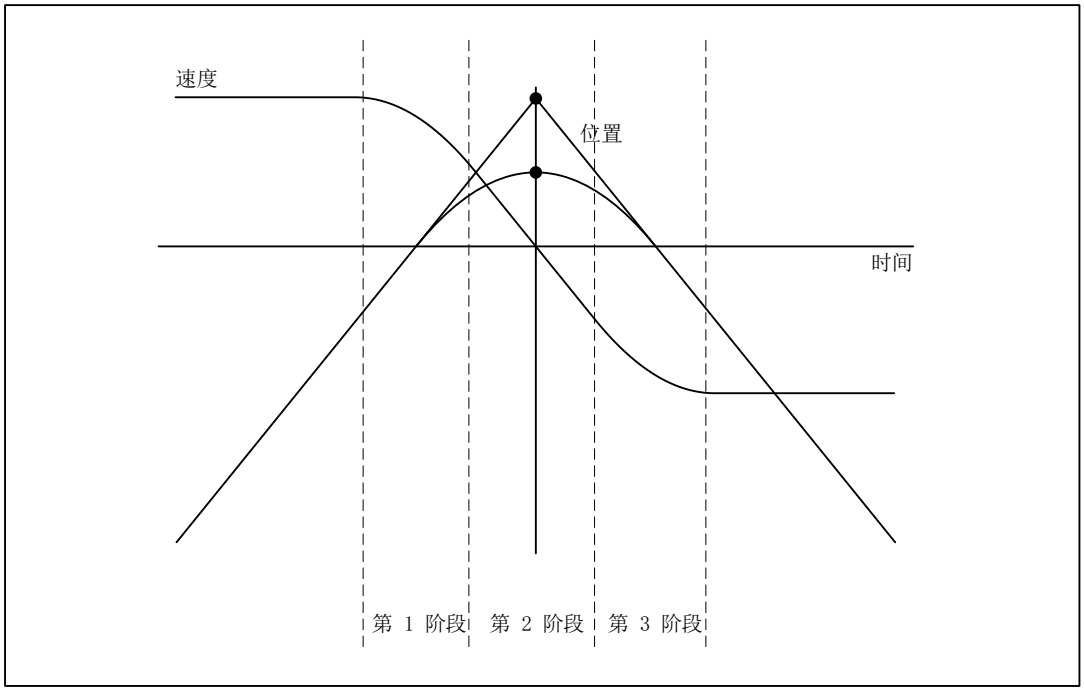
每轴速度跃变的平滑以及平滑距离的类型取决于插补是带急动度限制还是不带该限制。

不带该限制时，在整个平滑区域内每轴加速度都可以达到其最大值：



带该限制时，在整个平滑区域内每轴急动度都被限制在最大值以内。 因此，平滑运动通常分为 3 个阶段：

- **第 1 阶段**
  在第 1 阶段中，逐渐提升到每轴最大加速度。 此时，急动度保持恒定，等于每轴允许的最大急动度。
- **第 2 阶段**
  在第 2 阶段中，一直以最大加速度运行。
- **第 3 阶段**
  在第 3 阶段中，每轴加速度以最大急动度降到零。



5.3.3.4 相切程序段过渡的平滑(G645)

功能

在该带平滑的连续路径运行模式中，如果至少一根轴上原始轮廓的曲率出现跃变，即使是在相切程序段过渡处也会生成平滑程序段。

在平滑运动过程中，系统会确保所有参与轴不发生加速度跃变且不超出参数设置的、与原始轮廓的最大偏差（MD33120 $MA_PATH_TRANS_POS_TOL）。

对于折线式的、不相切的程序段过渡，平滑特性 G642，参见章节“按照定义的公差开展平滑(G642/G643) (页 218)”。

激活/取消激活

在每个数控程序段中都可以通过模态指令 G645 激活带相切程序段过渡平滑的连续路径运行。

可以选择非模态准停指令 G9 来中断连续路径运行。

或者选择以下某个指令来撤销 G645：

- 模态准停 (G60)
- 连续路径运行 G64, G641, G642, G643 或 G644

G642 - G645 的区别

G642 只平滑有尖角的程序段过渡，也就是说：至少一根轴的速度会出现跃变。 然而如果程序段为相切过渡，但是曲率呈跃变，则 G642 不会插入平滑程序段。 如果该程序段过渡以限定速度进行，则轴会进行稍稍的加速度跃变，但该跃变在急动度限制激活时不会超出设置的极限值（MD32432 $MA_PATH_TRANS_JERK_LIM）。 取决于该极限值的大小，程序段过渡处的轨迹速度有可能大幅降低。 可使用 G645 避免该情况，因为在该模式中平滑运动不会导致加速度跃变。

参数设置

通过以下机床数据可以为每轴设置 G645 时最大允许的轨迹公差：

MD33120 $MA_PATH_TRANS_POS_TOL

该轴只针对加速度不恒定的相切程序段过渡。 在对弯折的、非相切程序段过渡进行平滑时，同 G642，在该模式中 MD33100 $MA_COMPRESS_POS_TOL 的公差值生效。

参见

任意形状表面模式：基本功能 (页 252)

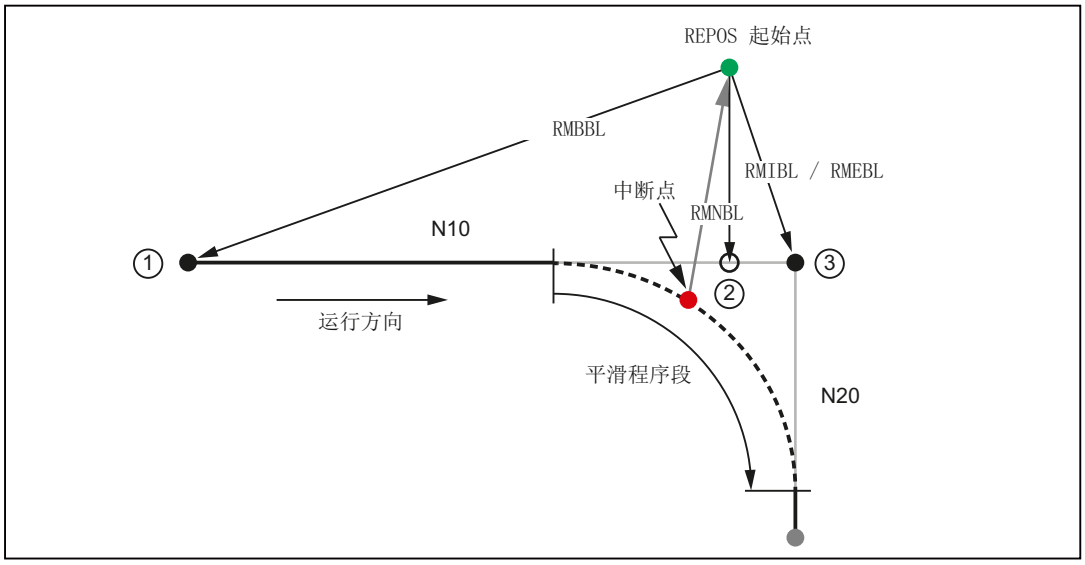
5.3.3.5 平滑和重新定位 (REPOS)

如果加工在平滑轮廓范围内中断，通过一个 REPOS 进程则**无法**再次直接定位到平滑轮廓上，而是只能定位到**编写的**轮廓上。

示例

编程：带编程平滑 G641 的两个运行程序段 N10 和 N20。

轴在平滑范围内中断运行。然后轴被手动运行到 REPOS 起始点上。根据所选 REPOS 模式重新定位至轮廓上的点 ①、② 或 ③。



RMBBL 重新定位至已中断运行程序段的开头

RMIBL 重新定位至中断点

RMEBL 重新定位至已中断运行程序段的末尾

RMNBL 重新定位至下一个轮廓点

① 程序段开头 N10

② 从 REPOS 起始点至下一个轮廓点

③ 程序段末尾 N10 / 程序段开头 N20

5.3.4 预读

5.3.4.1 标准功能

功能

预读是连续路径运行(G64、G64x)下的一个功能，它可以预先决定当前程序段以后的几个零件程序段的速度控制。

说明

预读功能仅限用于轨迹轴，不用于主轴和定位轴。

不使用预读功能时，如果相连程序段的行程很短，则处理每条程序段时只能达到某个速度，该速度值可以使轴在程序段终点制动并遵守加速度极限值。这说明，根本没有达到编程速

度。使用预读功能后，如果程序段过渡接近相切，则可以跨多条程序段进行加速和减速过程，对于短行程也可以获得高进给率。

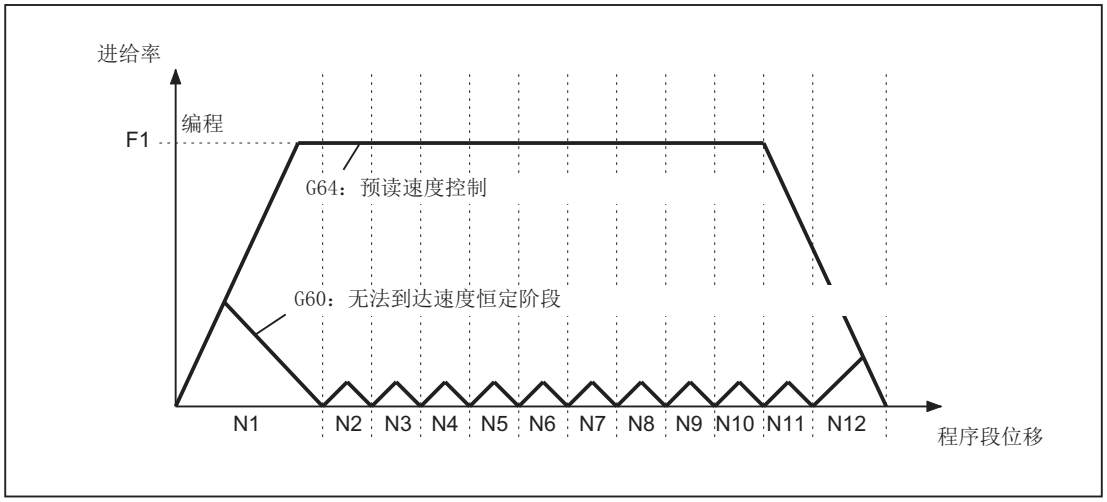


图 5-5　　短行程程序段中 G60 准停和 G64 （带预读连续路径运行）之间的速度特性对比

此时，系统会根据速度极限提前对轴进行制动，避免超出加速度极限和速度极限。

预读考虑了可以预见的速度极限：

- 程序段结束时的准停
- 程序段中的速度极限
- 程序段中的加速度极限
- 程序段切换时的速度极限
- 程序段切换时同步

工作原理

预读功能预先分析各程序段中可预见的速度极限，然后确定对应的制动斜坡。预读功能会自动根据程序段长度、制动能力和允许的轨迹速度加以调整。

出于安全原因，最后一条预处理程序段终点处的速度被视为 0，因为下一条程序段可能很短或者可能是准停程序段，而轴必须在达到程序段终点处前已经停止，

在一些具有高设定速度和极短行程的多条程序段中，可以根据当前预读出的速度值提高速度，以达到目标设定速度，然后再次降低该速度，在最后一条预读程序段终点处速度降为 0。由此会获得锯齿形的速度特性曲线（见下图），锯齿可以通过降低设定速度或提高预读程序段数量来加以避免。

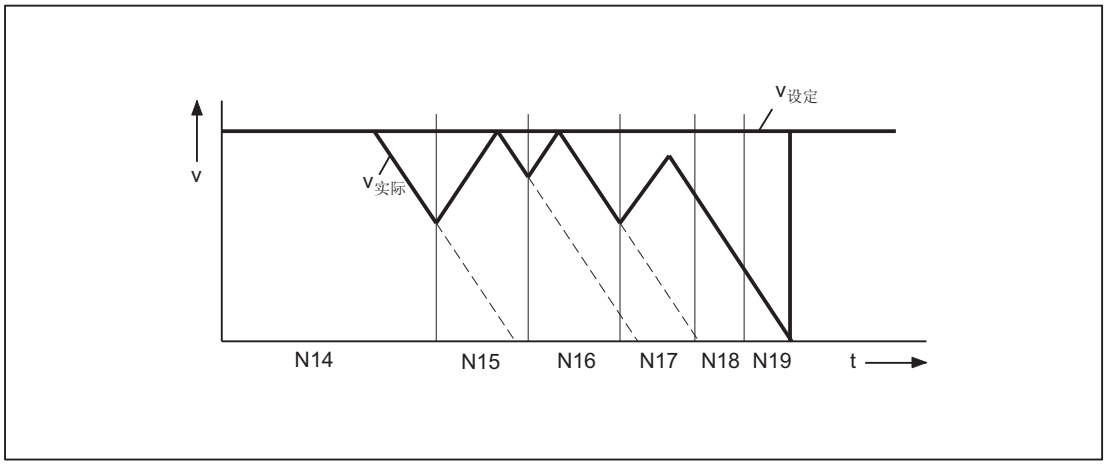


图 5-6　　示例：跨程序段的速度控制（预读程序段数量为 2）

激活/取消激活

预读通过选择连续路径运行 G64, G641, G642, G643, G644 或 G645 激活。

可以选择非模态指令准停 G09 来中断连续路径运行。

选择模态指令 G60 来取消预读。

参数设置

程序段数量

为确保连续路径运行中的安全运行，必须修改多条程序段的进给率。预读程序段的数量是系统内部自动确定的，但是也可以通过机床数据手动限制。该数据缺省值为 1，也就是说只预读后一条程序段，以进行速度控制。

预读主要用于短程序段（相对于制动行程而言），因此对于预读制动而言，程序段数量非常重要。将行程长度视为制动行程（即从最大速度到静止所需的行程），就已经足够。

如果一台机床的轴加速度比较低，为 $a = 1\ m/s^2$；而进给率比较高，为 $v_{轨迹} = 10\ m/min$，则在系统程序段处理周期 TB = 10 ms 时，得出的预读程序段数量 $n_{预读}$为：

$n_{预读}$= 制动行程/ 程序段长度= $(v_{轨迹}{}^2 / (2a)) / (v_{轨迹} * TB) = 9$

在该条件下，不建议预读超过 10 个程序段的进给率。指定的预读程序段数量会改变预读算法以及存储器占用。

在一个程序中加工速度通常是低于最大速度的，因此预读程序段数量比实际需要的多，从而不必要的占用计算性能。因此，实际所需的程序度数量是由速度和以下系数相乘后的结果得出的：

- 编程速度 * MD12100 $MN_OVR_FACTOR_LIMIT_BIN
  （使用**二进制**码的进给倍率开关时）
- 编程速度 * MD12030 $MN_OVR_FACTOR_FEEDRATE[**30**]
  （使用**格雷**码的进给倍率开关时）

MD12100 的值或者 MD12030 的第 31 个倍率值确定了动态响应裕量，它预留用于速度控制以提高轨迹进给率。

说明

MD12030 的第 31 个倍率值建议设为实际使用的倍率系数。

说明

预读程序段数量由受到插补缓冲器内允许的程序段数量的限制。

速度特性

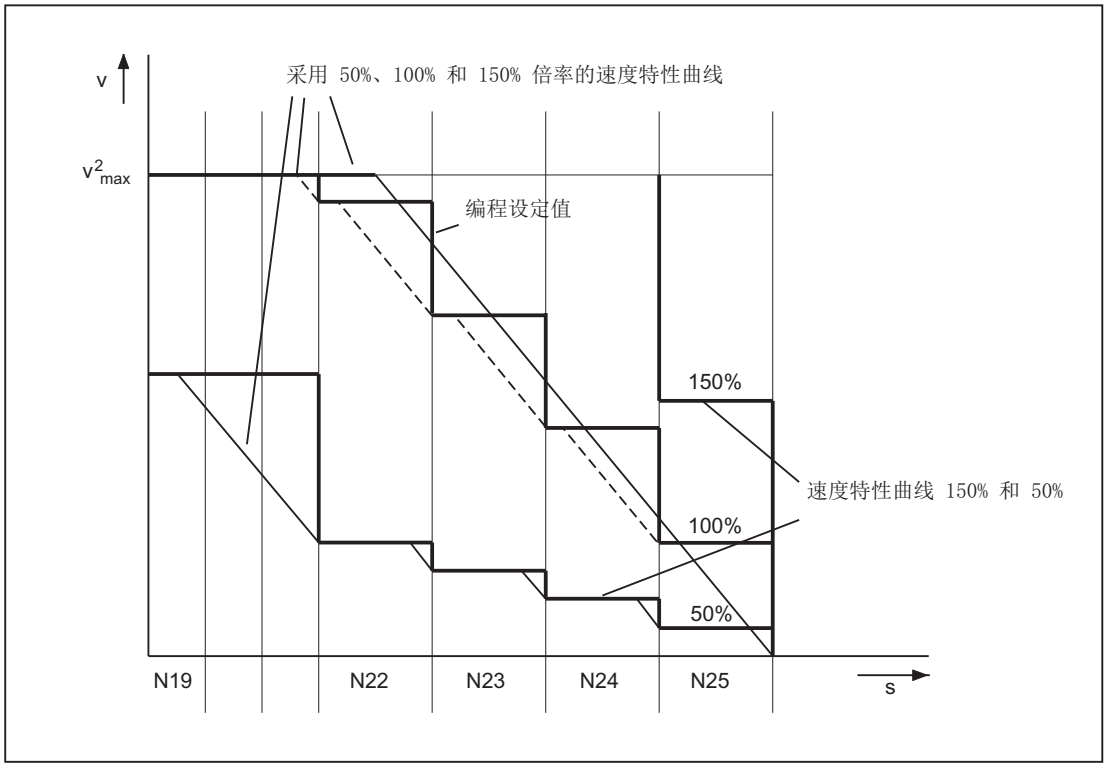
除了可以预见的固定速度极限，预读功能还可以处理编程的速度，这可以使当前程序段以外的程序段获得较小的速度。

- **确定后续程序段的速度**
  速度特性包括对后续程序段速度的检测。
  根据当前程序段和后续程序段的信息，预读功能计算出速度特性，并为当前超出值减去所需的速度修调量。
  速度特性的最大值由最大轨迹速度限制。
  此功能通过考虑修调值，可以在当前的程序段中降低速度，这样可以在下一个程序段开始时就到达较小的速度值。如果降低速度所需的时间比当前程序段中的进给时间，则在下一个程序段中继续降低速度。速度控制功能只用于后续程序段。
  该功能可由以下机床数据激活:
  MD20400 $MC_LOOKAH_USE_VELO_NEXT_BLOCK

值	含义
TRUE	功能生效
FALSE	功能**无效**

- **确定倍率基准值**
  如果后续程序段的速度特性不足够，比如：因为使用的倍率值过高（200%）或者因为激活了恒定切削速度 G96/G961，从而必须再次降低后续程序段中的速度，那么预读功能便会预读多个程序段，降低编程速度：
  通过定义的倍率基准值预读功能便可以为每个基准值计算出一个基准速度特性。从该特性中推导出当前倍率所需的速度降低幅度。
  速度特性的最大值由最大轨迹速度限制。
  最高基准值应该覆盖以下机床数据中的最大值能达到的速度范围：
  MD12030 $MN_OVR_FACTOR_FEEDRATE[n]（轨迹进给倍率开关的权重系数）
  以下机床数据值也可以达到该速度范围：
  MD12100 $MN_OVR_FACTOR_LIMIT_BIN（二进制编码的倍率开关的限制）
  因此可以避免速度指令所在的程序段出现减速情况。
  如果在 100%倍率条件下就需要达到明显的跨程序段减速，则建议在低倍率区内也设置一个基准值。
  使用的每通道倍率基准值的数量由以下机床数据确定：
  MD20430 $MC_LOOKAH_NUM_OVR_POINTS（预读中倍率开关基准值的数量）
  对应的基准值在以下机床数据中确定：
  MD20440 $MC_LOOKAH_OVR_POINTS（预读中倍率开关基准值）
  示例：
  基准速度特性，其中：
  – 倍率 = 50 %、100 % 或 150 %
  – 预读程序段数量 = 4
  – MD20430 $MC_LOOKAH_NUM_OVR_POINTS = 2
  – MD20440 $MC_LOOKAH_OVR_POINTS = 1.5, 0.5

– MD20400 $MC_LOOKAH_USE_VELO_NEXT_BLOCK = 1



可以组合这两种方法（即确定后续程序段速度和确定倍率基准值）来确定速度特性，通常我们也建议如此做，因为在该功能机床数据的缺省值中，已经覆盖了倍率相关速度限制的最大范围。

说明

如果不使用任何一种方法，则只有在当前程序段中才按设定速度移动。

说明

可预见的速度限制会限制倍率相关的速度限制。

出现程序段周期问题时的稳定系数

如果待处理程序段的行程过短，迫使预读功能降低机床速度，为程序段处理提供充足的时间，那么便会出现程序段处理周期问题。此时轨迹运动可能会出现持续的加速和减速制动过程。

这种速度波动可以通过稳定系数加以抑制：

MD20450 $MC_LOOKAH_RELIEVE_BLOCK_CYCLE（程序段处理周期的稳定系数）

前提条件

轴专用的进给停止/轴禁用

轴专用的进给停止/轴禁用不在预读功能的考虑范围内。

如果需要插补一根轴，而该轴又需要依据轴专用的进给停止/轴禁用指令停止，那么预读功能不会在该程序段前停止轨迹运动，而是**在该程序段中**停止运动。

如果不希望出现该特性，则可以将**轴专用**的进给停止改为**通道专用**的进给停止，这样轨迹运动就会立即停止了，参见章节“功能 (页 97)”。

5.3.4.2 任意形状表面模式：扩展功能

功能

功能“任意形状表面模式：扩展功能”是预读标准功能的一项扩展，用于计算任意形状表面加工中的轨迹速度特性，参见章节“任意形状表面模式：基本功能 (页 252)”。

该功能优化了连续路径运行，主要有：

- 对称的加速和制动特性
- 均匀的加速过程，即使是急动度限制或加速度限制变化时也是如此
- 均匀的设定速度加速过程，不管是否带设定的动态响应限制。
- 按更低的设定速度“预读式”制动

均匀的加速过程以及对动态响应限制的符合使设定速度特性更加平滑，从而可以将跟随误差对表面质量的影响降到最低程度。

优点

- 工件表面质量更加均匀
- 机床负载更轻

应用

功能“任意形状表面模式：扩展功能”用于加工主要由任意形状表面构成的工件。

说明

在标准加工中，该功能不体现任何优势，因此在标准加工中建议使用预读标准功能。

有效性

该功能只在以下条件下生效：

- 在运行方式：AUTOMATIC
- 在“软加速模式 (SOFT)”下

参数设置

工作存储器

通过以下机床数据配置“任意形状表面模式：扩展功能”功能存储器：

MD28533 $MC_MM_LOOKAH_FFORM_UNITS = <值>

所需存储器取决于零件程序、程序段长度、轴动态响应以及激活的坐标转换。

针对任意形状表面加工的参考值为：MD28533 = 18

说明

由于该功能会占用内存，因此建议只为参与任意形状表面加工的通道设置 MD28533。

插补缓冲器中的 NC 程序段数量

通常建议在使用“任意形状表面模式：扩展功能”时大幅提高插补缓冲器中的程序段数量：

MD28060 $MC_MM_IPO_BUFFER_SIZE > 100

程序段内存太小可能会降低轨迹速度特性的均匀度。

激活/取消激活

可以为每个动态响应模式单独地激活或取消该功能，详见章节“轨迹插补的动态响应模式 (页 249)”。

MD20443 $MC_LOOKAH_FFORM[<n>]= <值>

下标 <n>	动态响应模式	<值>	任意形状表面模式：扩展功能
0	标准动态响应设置 (DYNNORM)	0	关闭
		1	打开
1	定位运行，攻丝 (DYNPOS)	0	关闭
		1	打开
2	粗加工 (DYNROUGH)	0	关闭
		1	打开

下标 <n>	动态响应模式	<值>	任意形状表面模式：扩展功能
3	精加工 (DYNSEMIFIN)	0	关闭
		1	打开
4	精修整 (DYNFINISH)	0	关闭
		1	打开

只有在一同激活了“任意形状表面模式：基本功能”时，“任意形状表面模式：扩展功能”才生效。MD20443 $MC_LOOKAH_FFORM[<n>] 中的设置因此应该和 MD20606 $MC_PREPDYN_SMOOTHING_ON[<n>] 一致。

在“任意形状表面模式：扩展功能”关闭的动态响应模式中，预读标准功能生效。

编程

通常在程序中切换动态响应模式时，“任意形状表面模式：扩展功能”生效。

示例

已进行了以下参数设置：

MD20443 $MC_LOOKAH_FFORM[0] = 0

MD20443 $MC_LOOKAH_FFORM[1] = 0

MD20443 $MC_LOOKAH_FFORM[2] = 1

MD20443 $MC_LOOKAH_FFORM[3] = 1

MD20443 $MC_LOOKAH_FFORM[4] = 1

在零件程序中切换动态响应模式：

程序代码	注释
N10 DYNPOS	；激活动态响应模式 DYNPOS。 在该模式中，预读标准功能生效。
...	
N100 G17 G54 F10000	
N101 DYNFINISH	；激活动态响应模式 DYNFINISH。 在该模式中，“任意形状表面模式：扩展功能”生效。
N102 SOFT G642	
N103 X-0.274 Y149.679 Z100.000 G0	
N104 COMPCAD	
...	
N1009 Z4.994 G01	
N10010 X.520 Y149.679 Z5.000	

程序代码	注释
N10011 X10.841 Y149.679 Z5.000	
N10012 X11.635 Y149.679 Z5.010	
N10013 X12.032 Y149.679 Z5.031	
M30	

说明

在预读标准功能和“任意形状表面模式：扩展功能”之间切换时，连续轨迹运行会因插补器停止而中断。

前提条件

自动功能切换

使用以下功能会自动切换到预读标准功能：

- 螺纹切削/攻丝(G33, G34, G35, G331, G332, G63)
- 轨迹主轴耦合
- 步冲/冲压
- 坐标 PTP 运动

随后“任意形状表面模式：扩展功能”再次自动激活。

使用 G 功能组 15 的指令（进给率类型）

使用“任意形状表面模式：扩展功能”时不建议使用以下进给率类型：

- 旋转进给率(G95, G96, G97, ...)
- 反比时间进给率（G93）

使用 FLIN

使用“任意形状表面模式：扩展功能”时不可以使用进给属性 FLIN。

进给补偿的影响

和预读标准功能相比，通过机床控制面板或 $AC_OVR 设定的进给倍率可能会大大延长运行时间。

通过快速运行 (G0) 切换

在预读标准功能和“任意形状表面模式：扩展功能”之间切换时，预读功能不会切换到“任意形状表面模式中零散的 G0 程序段。也就是说：虽然 G0 和标准动态响应模式生效 (DYNNORM)，但是为 DYNNORM 预设的预读标准功能 (→ MD20443 $MC_LOOKAH_FFORM)不会一同自动生效。由于保留了当前激活的预读功

能，因此通常 G0 程序段和多项式程序段经过“平滑”后会变得平整，从而可以达到更加均匀的速度特性。

5.4 动态响应自适应功能

5.4.1 轨迹速度平滑

引言

速度控制会充分利用所给定的轴动态响应。如果无法达到编程进给率，则系统会使轨迹速度逼近设置的轴极限值和轨迹极限值，这会导致在轨迹上频繁制动和加速。

在高轨迹速度条件下先进行短暂的加速，再进行制动时，无法明显地缩短加工时间。加速过程反而会导致异常，比如：引发机床共振。

在一些应用，比如：模具制造和高速铣削中，需要保持均匀的轨迹速度。 这种应用中就应该舍弃短暂的加速过程，以保持轨迹速度的平稳。

功能

在“轨迹速度平滑”生效时，一个平滑系数会生效，以保持平稳的轨迹速度。该系数会确定允许的最高生产率下降幅度。 如果一个加速过程缩短的程序运行时间比该系数短，则不执行该加速过程。 此处只考虑那些频率高于设定的参与轴极限频率的加速过程。

优点：

- 避免因为频繁的加速和制动过程（在几个插补周期内）导致机床共振。
- 避免对程序运行时间影响不大的加速过程导致切削速度持续变化。

说明

轨迹速度平滑功能对轮廓误差没有影响。

因此，以恒定轨迹速度加工时，轮廓曲率导致的轴速度波动还是无法通过该功能加以避免。

同样，由于设定了新的进给率而导致的轨迹速度波动也无法抑制。避免该波动属于编程人员的责任。

前提条件

- 只有在 SOFT 和 BRISK 中激活了带预读的连续路径运行时，轨迹速度平滑才生效。 该平滑在 G0 时不生效。
- 控制系统的周期时间应合理设置，使预处理程序可以处理足够多的程序段，以分析加速过程。

激活/取消激活

轨迹速度平滑功能通过以下机床数据激活/取消：

MD20460 $MC_LOOKAH_SMOOTH_FACTOR（预读中的平滑系数）

值	含义
0.0	**不激活**轨迹速度平滑（缺省值）
> 0	激活轨迹速度平滑

只有在通过重新配置后该机床数据的修改才生效。

参数设置

平滑系数

平滑系数通过以下通道专用的机床数据设置：

MD20460 $MC_LOOKAH_SMOOTH_FACTOR（预读中的平滑系数）

该百分比值确定了和有加速/制动过程相比时，无加速/制动过程的工步可以最多延长多少时间。

该值应设为“最差”值，即假设一个程序内除了首个运动外的所有加速过程都被平滑。 实际上延长的时间肯定更短，不满足任何加速条件时，延长时间甚至为 0。 当然，也可以输入 50% 到 100% 的值，以避免明显延长加工时间。

考虑编程进给率

轨迹速度的平滑可以考虑和不考虑编程进给率。 通过以下机床数据进行选择：

MD20462 $MC_LOOKAH_SMOOTH_WITH_FEED（轨迹平滑考虑编程进给率）

值	含义
0	不考虑编程进给率。
1	考虑编程进给率（缺省值）。

在考虑编程进给率时，如果倍率为 100 %，会更好地保持 MD20460 确定的平滑系数。

轴专用的极限频率

轴专用的极限频率通过以下机床数据确定：

MD32440 $MA_LOOKAH_FREQUENCY （预读时的平滑频率）

如果加速和制动过程的频率过高，则系统会根据以下机床数据的设置平滑加速和制动过程，或降低动态响应：

MD20460 $MC_LOOKAH_SMOOTH_FACTOR（预读中的平滑系数）

MD20465 $MC_ADAPT_PATH_DYNAMIC（轨迹动态响应自适应）

有关 MD20465 的更多信息参见章节“轨迹动态响应自适应 (页 240)”。

说明

如果某轴机械部件出现振动，而且其振动频率已知，则应该将 MD32440 设为低于该频率的值。

比如：共振频率可以通过内置的测量功能测出。

工作原理

从轨迹参与轴中自动计算出 MD32440 的最小值（= $f_{轨迹}$）。 平滑功能只考虑在以下时间内再次回到运动起始速度或结束速度的加速过程：

$t = t_2 - t_1 = 2 / f_{轨迹}$

如果加速过程导致的加工时间延长超出平滑系数（MD20460）设置的极限值，则舍弃该加速过程。

示例

已进行了以下参数设置：

MD20460 $MC_LOOKAH_SMOOTH_FACTOR = 10 %

MD32440 $MA_LOOKAH_FREQUENCY[AX1] = 20 Hz

MD32440 $MA_LOOKAH_FREQUENCY[AX2] = 20 Hz

MD32440 $MA_LOOKAH_FREQUENCY[AX3] = 10 Hz

在该轨迹上有 3 根轴参与：X = AX1, Y = AX2, Z = AX3。

3 根轴上 MD32440 的最小值为 10 Hz， 因此只检查在时间段“$t_2 - t_1 = 2 / 10\ Hz = 200\ ms$”内执行的加速过程。 t_2 是从速度 v_1 开始进行加速后再次回到 v_1 的时间点。 只有该时间段才会一同考虑进延长的加工时间。

如果时间段“$t_2 - t_1$”超过 200 毫秒或者额外的程序运行时间“$t_3 - t_2$”超过“$t_2 - t_1$”的 10 % (= MD20460)，则进行以下动作：

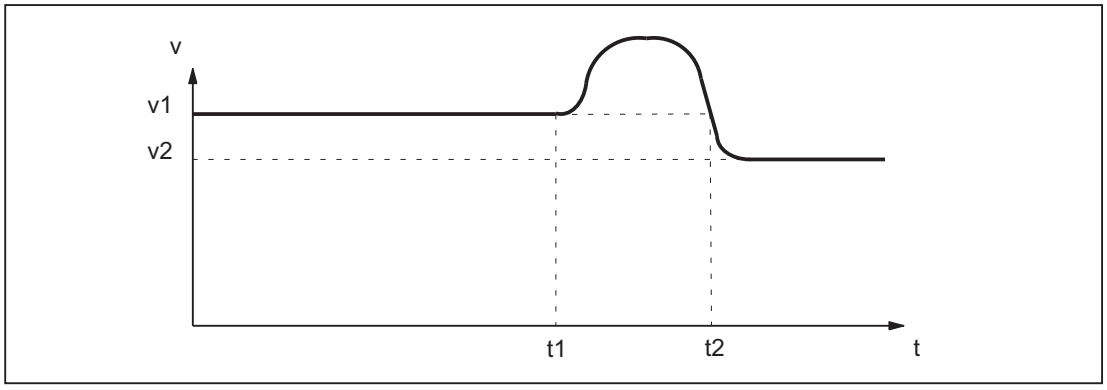


图 5-7　　轨迹速度在时间上最优，无平滑

如果与此相反，时间段“$t_2 - t_1$”短于 200 毫秒或者额外的程序运行时间“$t_3 - t_2$”没有超过“$t_2 - t_1$”的 10 %)，则进行以下动作：

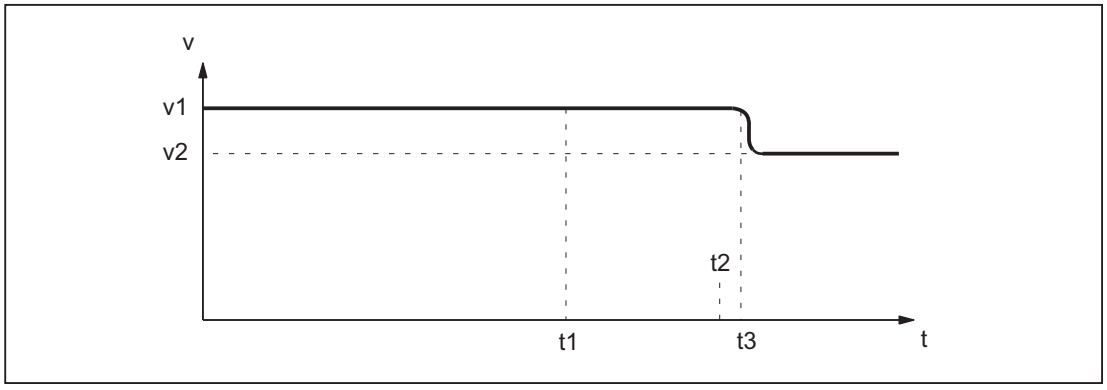


图 5-8　　平滑轨迹速度

5.4.2　轨迹动态响应自适应

功能

加工期间高动态的加速和制动过程可能会引起机床机械振动，从而降低工件的表面质量。

利用功能“轨迹动态响应自适应”可以根据机床具体条件来调整加速和制动过程的动态响应。

说明

该功能不考虑参与轨迹的单轴的加速和制动过程，只考虑整个轨迹的加速和制动过程。 因此，即使在保持恒定轨迹速度条件下，也可能由于不均匀的轮廓变化或者坐标转换，导致一些引发机械振动的加速和制动过程。

有效性

“轨迹动态响应自适应”只在轨迹运动期间有效：

- 连续路径运行（G64, G64x）
  在连续路径运行中，当倍率为 100 % 时，可以达到最佳的动态响应自适应效果。 当倍率明显低于该值时或者有功能导致轨迹轴制动时（比如：输出给 PLC 的辅助功能），自适应效果会明显变差。
- 准停（G60）

在以下条件下，即使在轨迹运动中，“轨迹动态响应自适应”也**不生效**：

- 编程了快速运行（G0）
- 倍率值变化
- 在运动期间出现停止请求（如 NC 停止、NC 复位）
- “速度相关的轨迹加速度”（DRIVE）功能激活

激活/取消激活

该功能通过以下机床数据激活/取消：

MD20465 $MC_ADAPT_PATH_DYNAMIC（轨迹动态响应自适应）

值	含义
= 1.0	**不激活**轨迹动态响应自适应（缺省值）
> 1.0	激活轨迹动态响应自适应

在激活该功能后，连续路径运行内部会自动一**同激活“轨迹速度平滑”**，参见章节“轨迹速度平滑 (页 237)”。

如果 MD20460 $MC_LOOKAH_SMOOTH_FACTOR 平滑系数被设为 0（即取消功能，缺省值！），则使用 100 % 的平滑系数。 如果平滑系数不是 0 %，则按其原本设置值生效。

参数设置

轨迹动态响应调整系数

通过轨迹动态响应调整系数可以按下调后的动态响应限值执行一些短暂的轨迹速度变化。

调整系数可以针对各个通道单独设置：

- 如果是不带急动度限制的刚性加速 (**BRISK**)，则该系数通过以下机床数据设置：
  MD20465 $MC_ADAPT_PATH_DYNAMIC[**0**]
  → 调整系数作用于加速度。
- 如果是带急动度限制的软加速 (**SOFT**)，则该系数通过以下机床数据设置：
  MD20465 $MC_ADAPT_PATH_DYNAMIC[**1**]
  → 调整系数作用于急动度。

轴专用的极限频率

动态响应限制功能应只用于那些机械振动频率大于定义的极限频率，从而导致机床共振的加速和制动过程。

启用动态响应限制功能的极限频率可以通过以下机床数据针对各轴单独设置：

MD32440 $MA_LOOKAH_FREQUENCY （预读时的平滑频率）

更多信息参见章节“轨迹速度平滑 (页 237)”。

工作原理

在加工期间，控制系统会周期性地从所有轨迹参与轴中计算出所有极限频率中的最小值，将该值用作启用动态响应自适应的极限频率(f) ，由此计算出对应的时间窗口（$t_{自适应}$）

$t_{自适应} = 1 / f$

$t_{自适应}$的大小确定了后续特性：

1. 速度变化所需时间短于 $t_{自适应}$：
   加速度按照大于 1 的系数降低，小于等于以下机床数据：
   MD20465 ADAPT_PATH_DYNAMIC （轨迹动态响应自适应）
   降低加速度后，速度变化时间会延长。
   有以下几种情况：
   – 加速度降到比 MD20465 小的值，以持续 $t_{自适应}$[s]。 不一定要充分利用允许的降低幅度。
   – 加速时间按照 MD20465 值缩短。 加速度虽然更低，但时间比 $t_{自适应}$ 短。 充分利用了允许的降低幅度。
2. 速度变化所需时间长于 $t_{自适应}$：
   无需进行动态响应自适应。

示例

下面的示例生动地地展示了“轨迹动态响应自适应”对刚性加速 BRISK 的影响：

已进行了以下参数设置：

MD20465 $MC_ADAPT_PATH_DYNAMIC[0] = 1.5

MD20460 $MC_LOOKAH_SMOOTH_FACTOR = 1.0

MD32440 $MA_LOOKAH_FREQUENCY[AX1] = 20 Hz T_{AX1} = 1/20 Hz = 50 ms

MD32440 $MA_LOOKAH_FREQUENCY[AX2] = 10 Hz T_{AX2} = 1/10 Hz = 100 ms

MD32440 $MA_LOOKAH_FREQUENCY[AX3] = 20 Hz T_{AX3} = 1/20 Hz = 50 ms

说明

为了清晰地展示动态响应自适应的效果，平滑系数 MD20460 被设为 1，因此实际上一同激活的“轨迹速度平滑”功能并未起效。

在该轨迹上有 3 根轴参与：X = AX1, Y = AX2, Z = AX3。

在 AX2 参与的轨迹运动中，所有短于 T_{AX2} 的加速和制动过程都被调整。

在只有 AX1 和/或 AX3 参与的轨迹运动中，所有短于 $T_{AX1} = T_{AX3}$ 的加速和制动过程都被调整。

相关的时间窗口在下图中标为 $t_{自适应...}$。

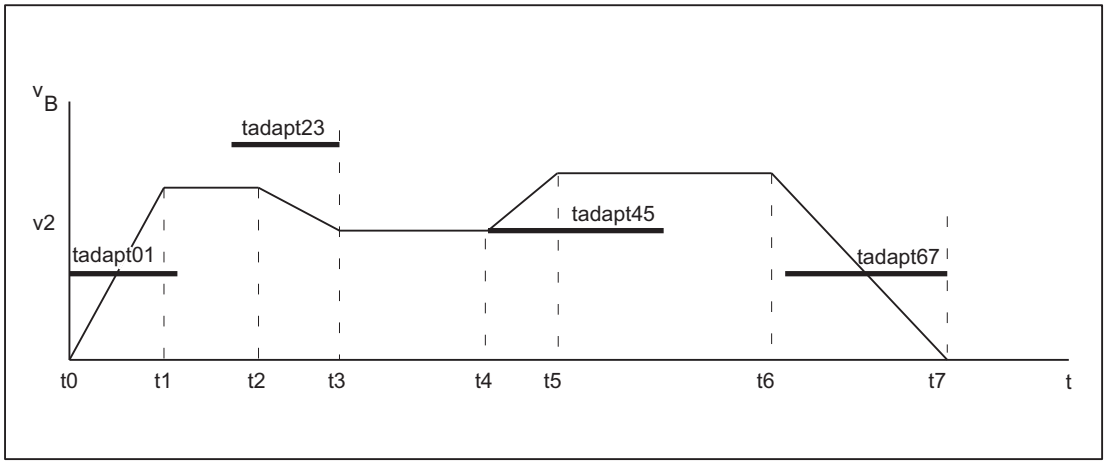


图 5-9 时间最优的轨迹速度曲线，无平滑，无动态响应自适应

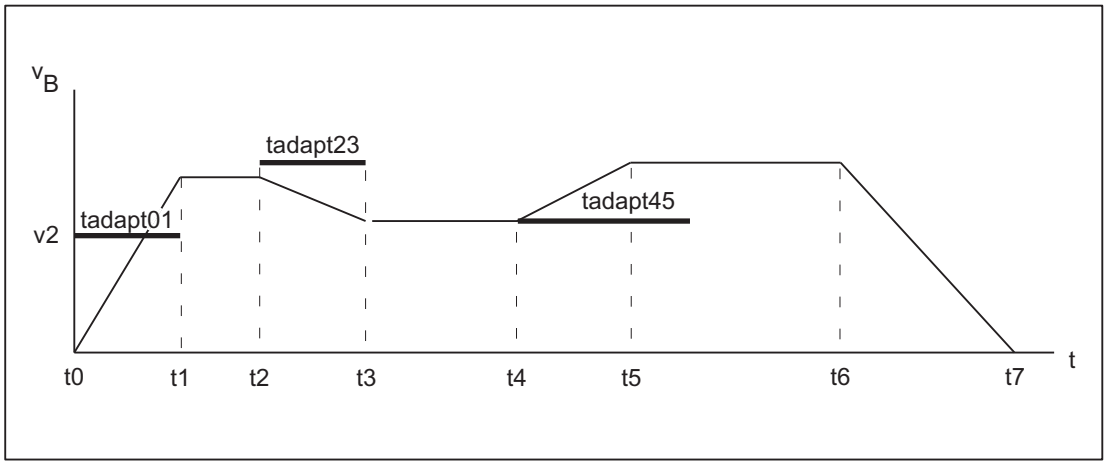


图 5-10　　轨迹速度曲线，有动态响应自适应

间隔“t_0 - t_1”和“t_2 - t_3”:　　由于对加速度的调整，“t_0 - t_1”之间的加速过程和“t_2 - t_3”之间的制动过程延长到 $t_{自适应\ 01}$ 或 $t_{自适应\ 23}$。

间隔“t_4 - t_5”:　　“t_4 - t_5”之间的加速过程以按最大调整系数 1.5 下调后的加速度进行。 但是，加速过程在 $t_{自适应\ 45}$ 前结束。

间隔“t_6 - t_7”:　　“t_6 - t_7”之间的制动过程保持不变，因为它持续的时间比 $t_{自适应\ 67}$ 长。

5.4.3　确定动态响应极限值

调试功能“轨迹动态响应自适应”时除了要确定轨迹轴固有频率，以确定轴专用的极限频率（MD32440 $MA_LOOKAH_FREQUENCY）外，还需要确定速度、加速度和急动度的动态响应极限值。

步骤

下文将介绍如何通过软加速 SOFT 来确定轨迹轴的动态响应极限值。 这些步骤同样适用于刚性加速 BRISK。

1. 取消“轨迹动态响应自适应”：
   MD20465 $MC_ADAPT_PATH_DYNAMIC [1] = 1
2. 检查在不同运行速度下各轨迹轴的定位特性。 设置合适的急动度，使运行保持理想的定位公差。

说明

定位开始时的运行速度越高，通常急动度就可以设置得越高。

3. 采用安全运行速度允许的最大急动度：
   MD32431 $MA_MAX_AX_JERK（最大急动度）
4. 确定所有轨迹轴的系数 F_{APD}，其中：
   F_{APD} = 得出的最大急动度/得出的最小急动度

   说明

   得出的最小急动度是在临界运行速度条件下的急动度。

5. 输入所有轨迹轴上得出的最大系数 F_{APD}，它将用作动态响应调整系数：
   MD20465 $MC_ADAPT_PATH_DYNAMIC [1] = F_{APD}

5.4.4 “轨迹速度平滑”和“轨迹动态响应自适应”的共同作用

下面的示例生动地说明了在连续路径运行中，“轨迹速度平滑”和“轨迹动态响应自适应”的共同作用效果。

示例 1

加速模式：BRISK

在该轨迹上有 3 根轴参与：X = AX1, Y = AX2, Z = AX3。

已进行了以下参数设置：

MD20465 $MC_ADAPT_PATH_DYNAMIC [0] = 3

MD20460 $MC_LOOKAH_SMOOTH_FACTOR = 80.0

MD32440 $MA_LOOKAH_FREQUENCY[AX1] = 20　　T_{AX1} = 1/20 Hz = 50 ms

MD32440 $MA_LOOKAH_FREQUENCY[AX2] = 20　　T_{AX2} = 1/20 Hz = 50 ms

MD32440 $MA_LOOKAH_FREQUENCY[AX3] = 20　　T_{AX3} = 1/20 Hz = 50 ms

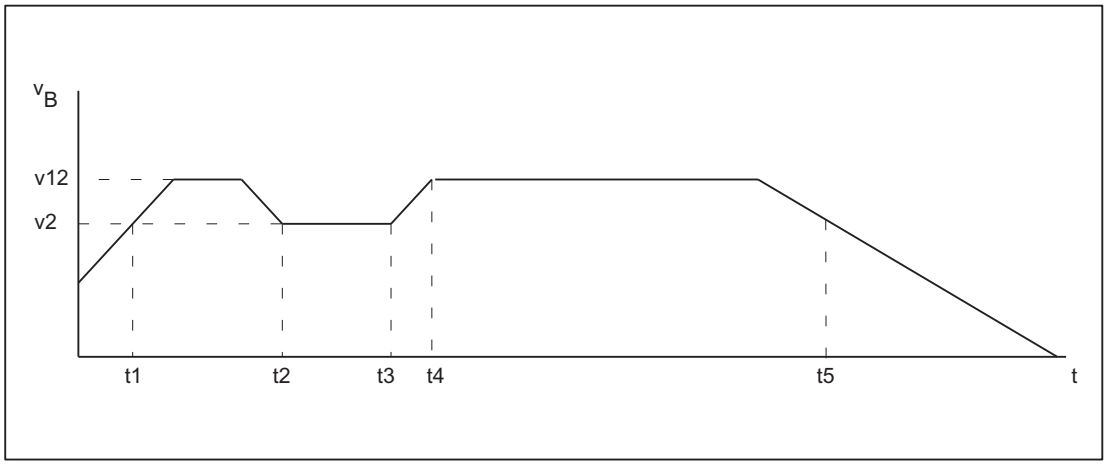


图 5-11　　时间最优的轨迹速度曲线，无平滑，无动态响应自适应

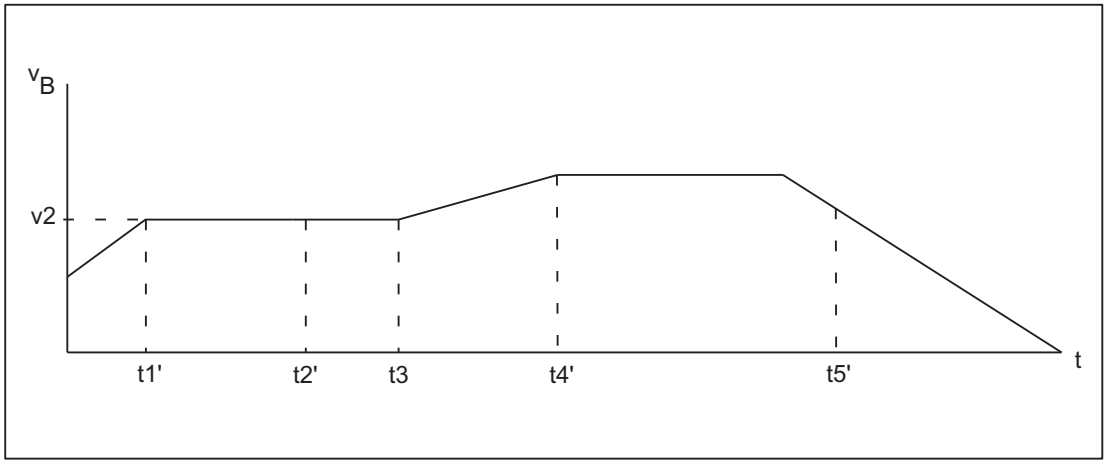


图 5-12　　轨迹速度曲线，轨迹速度平滑和轨迹动态响应自适应共同作用

轨迹速度平滑的作用：

间隔“t_1 - t_2”:　　由于不带加速到 v_{12} 这一过程时，延长的加工时间比平滑系数 80 % 得出的时间短，因此跳过了“ t_1 - t_2”之间的加速和制动过程。

间隔“t_3 - t_5”:　　“ t_3 - t_5”之间的加速和制动过程不满足这一条件，或者长于设置的平滑时间 T_{Axn} = 2/20 Hz = 100 ms。

动态响应自适应的作用：

间隔“t_3 - t_4”: “t_3 - t_4”的加速过程短于 MIN(T_{AXn}) = 1/20 Hz = 50 ms，因此采用按调整系数 3 下调后的加速度。

间隔“开始到 t_1”：由于动态响应自适应，t_1 前进行轨迹平滑后剩下的加速过程被延长到 t_1'。

说明

该示例表明，即使轨迹速度平滑功能没有消除一些加速或制动过程，这些过程也会接着由动态响应自适应功能加以优化。 因此，应尽量同时激活这两个功能。

示例 2

加速模式：SOFT

在该轨迹上有 3 根轴参与：X = AX1, Y = AX2, Z = AX3。

已进行了以下参数设置：

MD20465 $MC_ADAPT_PATH_DYNAMIC[1] = 1

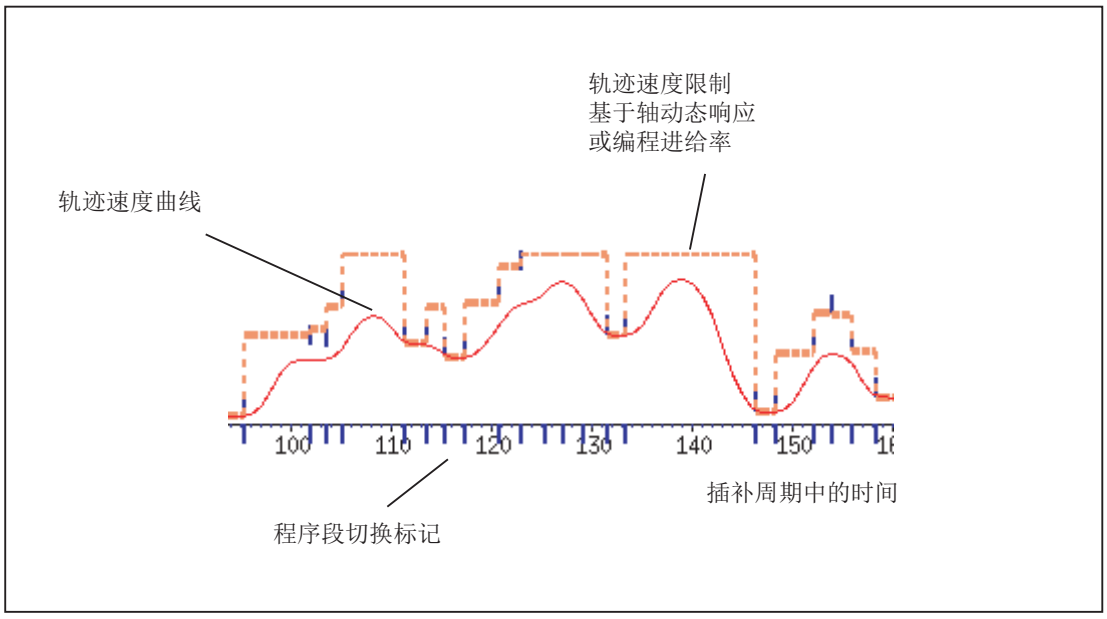
MD20460 $MC_LOOKAH_SMOOTH_FACTOR = 0.0

MD32440 $MA_LOOKAH_FREQUENCY[AX1] = 10 T_{AX1} = 1/20 Hz = 100 ms

MD32440 $MA_LOOKAH_FREQUENCY[AX2] = 10 T_{AX2} = 1/20 Hz = 100 ms

MD32440 $MA_LOOKAH_FREQUENCY[AX3] = 20 T_{AX3} = 1/20 Hz = 50 ms

这些设置会产生时间最优的轨迹速度曲线，没有轨迹速度平滑和动态响应自适应：

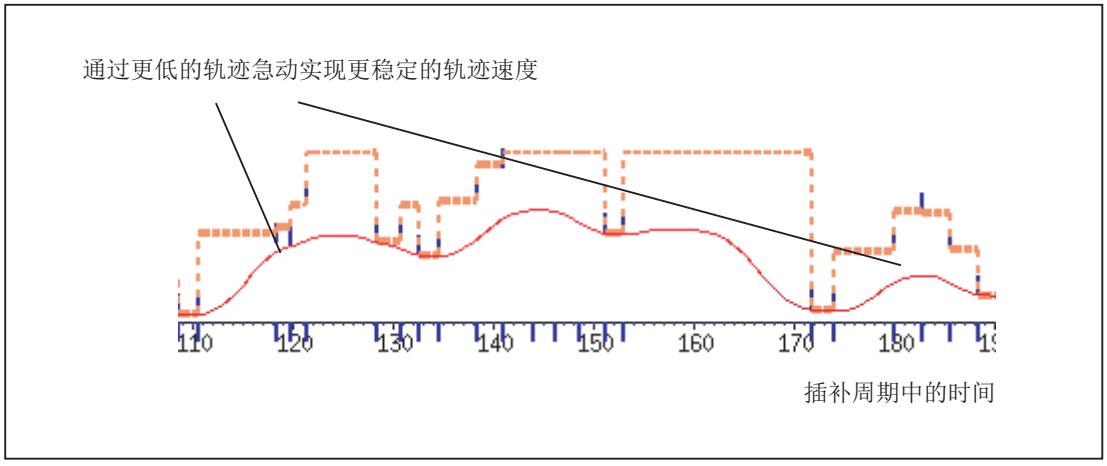


进行以下参数设置：

MD20465 $MC_ADAPT_PATH_DYNAMIC[1] = 4

MD20460 $MC_LOOKAH_SMOOTH_FACTOR = 1.0

这些设置会产生带动态响应自适应、最低程度（几乎关闭）轨迹速度平滑的轨迹速度曲线：

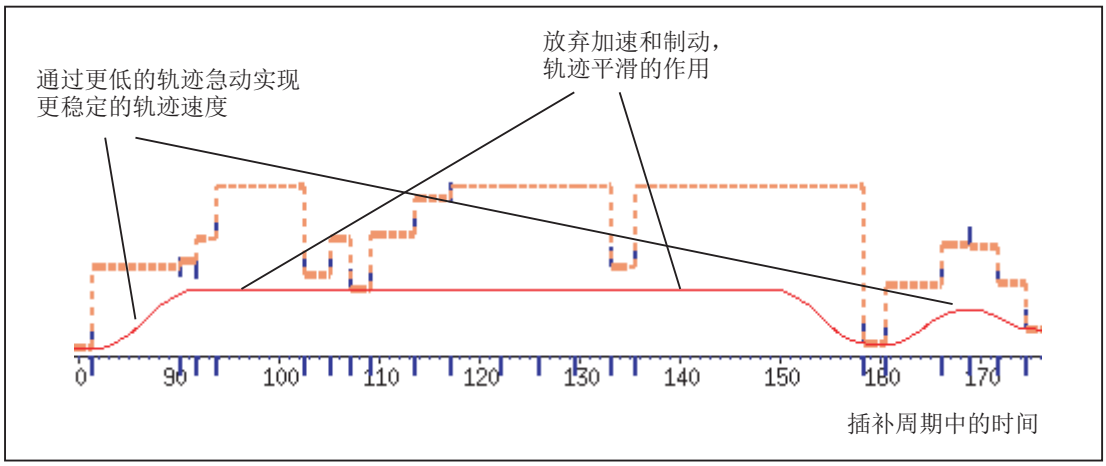


平滑系数从 1 % 改为 0 %（即缺省值！）：

MD20460 $MC_LOOKAH_SMOOTH_FACTOR = 0.0

该设置使 100 % 的平滑系数生效。

由此得出了一条轨迹速度平滑和轨迹动态响应自适应共同作用的轨迹速度曲线：



5.4.5 轨迹插补的动态响应模式

功能

针对不同工艺的动态响应设置可以通过机床数据确定，然后在程序中通过第 59 G 功能组指令（轨迹插补的动态响应模式）激活。

指令	激活动态响应设置
DYNNORM	标准动态响应设置
DYNPOS	定位模式，攻丝
DYNROUGH	粗加工
DYNSEMIFIN	精加工
DYNFINISH	精修整

说明

第 59 G 功能组指令（轨迹插补的动态响应模式）只能激活**轨迹轴**的动态响应。它对以下轴没有影响：

- 定位轴
- PLC 轴
- 指令轴
- 基于轴耦合的运动
- 手轮叠加运动
- JOG 运动
- 参考点运行 (G74)
- 运行至固定点 (G75)
- 快速运行 （G0）

对于这些运动而言，始终是标准动态响应设置 (DYNNORM) 生效。

应用

比如：通过不同的动态响应设置，工件的粗加工可以达到时间最优，而精加工可以获得最佳工件表面。

参数设置

轴专用的动态响应设置：

- MD32300 $MA_MAX_AX_ACCEL[<n>]（轴加速度）
- MD32310 $MA_MAX_ACCEL_OVL_FACTOR[<n>]（用于限制速度跳跃的过载系数）
- MD32431 $MA_MAX_AX_JERK[<n>]（轨迹运动时的最大轴急动度）
- MD32432 $MA_PATH_TRANS_JERK_LIM[<n>]（连续路径运行中程序段过渡处的最大轴急动度）
- MD32433 $MA_SOFT_ACCEL_FACTOR[<n>]（SOFT 指令时的最大加速度比例）

通道专用的动态响应设置：

- MD20600 $MC_MAX_PATH_JERK[<n>]（最大轨迹急动度）
- MD20602 $MC_CURV_EFFECT_ON_PATH_ACCEL[<n>]（轨迹曲率对轨迹加速度的影响）
- MD20603 $MC_CURV_EFFECT_ON_PATH_JERK[<n>]（轨迹曲率对轨迹急动度的影响）

其中下标 <n> = 0 用于 DYNNORM
1 用于 DYNPOS
2 用于 DYNROUGH
3 用于 DYNSEMIFIN
4 用于 DYNFINISH

说明

在**写入无下标机床数据**时，系统将向该机床数据的所有数组单元内写入相同值。
在**读无下标机床数据**时，系统始终下标 0 的数组单元的值。

封锁 G 指令

我们建议通过以下机床数据封锁不使用的第 59 G 功能组指令（轨迹插补的动态响应模式）：

MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[<n>]（重新配置后的指令列表）

使用一个封锁的 G 指令时，系统会输出报警。以避免未经设置的机床数据生效。

示例

以下设置可以封锁 G 指令 DYNPOS 和 DYNSEMIFIN：

- MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[0]="DYNPOS"
- MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[1]=" "
- MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[2]="DYNSEMIFIN"
- MD10712 $MN_ NC_USER_CODE_CONF_NAME_TAB[3]=" "

文档

关于第 59 G 功能组指令的编程说明请参见
文档：
编程手册之工作准备分册；章节：轨迹特性

5.4.6 任意形状表面模式：基本功能

引言

在刀具和模具制造中，工件表面的均匀度非常重要，其重要性甚至高于工件表面的精度。

不均匀的工件表面可能是由以下原因引起的：

- 工件加工程序内包含的尺寸不均匀，这主要是一些曲率和挠率的变化。

  说明

  轮廓的曲率 k 是针对轮廓上某个点的切线方向角对弧长的转动率，是半径的倒数 (k = 1/r)。挠率是曲率的变化率（1 阶导数）。

  执行程序时，如果几何尺寸不均匀，会达到机床的动态响应极限，导致多余的加速和制动过程。由此会导致轮廓误差，误差水平取决于轴的有效滞后量。

- 多余的加速和制动过程还可导致机床共振，在工件上留下加工痕迹。

为排除这些异常，可以采取以下方法：

- 使用 CAD/CAM
  用 CAD/CAM 系统创建程序，这种程序会包含非常均匀的曲率和挠率曲线，因此不会导致轨迹速度的异常减速。
- 确定合理的最大轨迹速度，避免曲率和挠率的波动产生影响。

功能

使用“任意形状表面模式：基本功能”，可以使定义的轨迹速度极限对细微的曲率或挠率变化更加“迟钝”，同时还不会超出机床关于加速度和急动度方面的动态响应限制。

使用该功能的优点有：

- 均匀的轨迹速度
- 工件表面质量更加均匀
- 缩短加工时间，条件是机床的动态响应允许

应用

该功能用于加工主要由任意形状表面构成的工件。

前提条件

只有在预留了所需内存后，才能激活该功能：

MD28610 $MC_MM_PREPDYN_BLOCKS = 10

输入值确定了在确定轨迹速度时（速度处理）需要计算多少条程序段。
推荐值为 10。

如果 MD28610 的值为 0，则在确定一条程序段的最大轨迹速度时，只考虑该程序段中的运动。如果该机床数据值大于 0，则在确定轨迹速度时会一同考虑相邻程序段中的尺寸，从而使轨迹速度更加均匀。

激活/取消激活

可以为每个动态响应模式单独地激活或取消该功能，详见章节“轨迹插补的动态响应模式 (页 249)”。

MD20606 $MC_PREPDYN_SMOOTHING_ON[<n>] = <值>

下标 <n>	动态响应模式	<值>	任意形状表面模式：基本功能
0	标准动态响应设置 (DYNNORM)	0	关闭
		1	打开
1	定位运行，攻丝 (DYNPOS)	0	关闭
		1	打开
2	粗加工 (DYNROUGH)	0	关闭
		1	打开
3	精加工 (DYNSEMIFIN)	0	关闭
		1	打开
4	精修整 (DYNFINISH)	0	关闭
		1	打开

说明

由于该功能会占用内存，因此建议只在对应加工通道中激活该功能。

参数设置

轮廓采样系数的变化

在插补曲面轮廓时产生的割线误差取决于以下因素：

- 曲率
- 插补周期（在 MD10071 $MN_IPO_CYCLE_TIME 中显示）
- 加工对应轮廓的速度

允许的最大割线误差可通过以下机床数据针对各轴设置：

MD33100 $MA_COMPRESS_POS_TOL（压缩时的最大偏差）

如果设置的插补周期不够短，在特别弯曲的轮廓上最大轨迹速度有可能会下降。减速是为了保证工件表面的加工精度。

通过修改轮廓采样系数，可以将插补器对曲面轮廓进行采样的周期（即轮廓采样周期）设为和插补周期不一样的值。如果轮廓采样周期短于插补周期，在特别弯曲的轮廓上最大轨迹速度不会降低。

轮廓采样系数通过以下机床数据设置：

MD10682 $MN_CONTOUR_SAMPLING_FACTOR

有效的轮廓采样周期的计算公式为：

$T_s = f * T_1$

其中： T_s = 有效的轮廓采样时间
T_1 = 插补周期
f = 轮廓采样系数（MD10682 的值）

轮廓采样系数的缺省值为 1，即等于插补周期。

该系数可大于也可小于 1。

如果小于 1，则对轮廓采样精度的检查失效。

设置的采样时间不能短于设置的最短轮廓采样时间：

MD10680 $MN_MIN_CONTOUR_SAMPLING_TIME

说明

MD10680 是根据具体控制系统型号固定设置的，无法修改。

编程

取决于机床数据 MD20606 $MC_PREPDYN_SMOOTHING_ON 的设置，可以通过切换生效的动态响应模式在程序中激活或取消“任意形状表面模式：基本功能”。

示例：

设置 MD20606 $MC_PREPDYN_SMOOTHING_ON[2-4] = 1 且 MD20606 $MC_PREPDYN_SMOOTHING_ON[0-1] = 0 后，便可以通过指令 DYNROUGH、DYNSEMIFIN 和 DYNFINISH 激活该功能，通过指令 DYNNORM 和 DYNPOS 关闭该功能。

参见

相切程序段过渡的平滑(G645) (页 225)

随速度提高急动度（轴专用） (页 311)

任意形状表面模式：扩展功能 (页 233)

5.5 压缩功能

5.5.1 压缩线性程序段、圆弧程序段和快速运行程序段

5.5.1.1 功能

CAD/CAM 为描述复杂轮廓而生成了大量的线性和圆弧程序段，其中一些的轨迹长度很短。允许的最大轨迹速度常常会受到程序段切换时间的限制。自某个轨迹速度开始，预处理不再快速处理新的运行程序段并切换至主处理。

线性程序段的压缩

压缩器功能能够将连续的线性程序段替换为轨迹长度尽可能长的多项式程序段，同时维持可参数设置的轮廓精度。使用该功能的优点有：

- 减少运行程序段的数量
- 提升轨迹速度
- 提升表面质量
- 程序段过渡稳定

下表列出了可用的压缩器功能及其重要的特性：

压缩器	功能	程序段过渡上的稳定性	使用说明
COMPON	COMPON erzeugt aus **bis zu 10** aufeinander folgenden Linearsätzen der Form "G01 X...Y...Z...F..." einen Polynom-Satz.	速度稳定	
COMPCURV	与 COMPON 一样	速度稳定和加速度稳定	
COMPCAD	COMPCAD 可以将**任意数量的**连续的线性程序段压缩成一条多项式程序段。	速度稳定和加速度稳定	COMPCAD 非常占用内存和计算能。因此我们建议只有在 CAD/CAM 程序中的措施无法改进工件表面质量时，才使用 COMPCAD。
COMPSURF	与 COMPCAD 一样	速度稳定和加速度稳定	使用 COMPSURF 比使用 COMPCAD 达到的效果更好。与 COMPCAD 一样，COMPSURF 非常占用内存和计算能。 使用 COMPSURF 可以在 CAD/CAM 程序中明显提高工件表面质量，例如在倾斜划分的简单程序上、不好的数据质量和/或不规则的点分布时。 此外，COMPSURF 还能进行相同的、可切断的铣削轨迹平滑（与方向无关），从而明显提高了两个方向上的铣削表面质量。

在待压缩运行程序段中，以及在这些程序段之间编写了非运行指令（例如辅助功能输出）时，压缩进程会中断。

可通过机床数据为所有压缩器功能设定计算出的轨迹和编写的位置之间所允许的最大偏差（参见“参数设置 (页 257)”）。和 COMPON、COMPCURV 不同的是， COMPCAD 和 COMPSURF 不会在相邻轨迹各个方向充分利用设置的偏差。

压缩圆弧程序段和快速运行程序段

除压缩线性程序段以外，所有压缩器功能也可用于压缩快速运行程序段（G0 程序段）。而圆弧程序段只能通过压缩器功能 COMPCAD 压缩。

圆弧和/或快速运行程序段的压缩通过以下机床数据的**百位**针对具体通道设置：

MD20482 $MC_COMPRESSOR_MODE = <值>

值	含义
0xx	**不**压缩圆弧程序段和 G0 程序段。这与之前的软件版本兼容。
1xx	通过 COMPCAD 将圆弧程序段线性化并压缩。 优点： 压缩器功能更加精确，并且通常由此产生更佳的表面。 缺点： 压缩器功能对 NC 程序中的缺陷更加敏感。出于兼容性原因，可能需要保留设置 0xx。
2xx	压缩 G0 程序段，其中另一公差视具体情况生效（参见“G0 运行的公差系数 (页 276)”）。 优点： 由于可设置更大的公差和压缩 G0 进给运动，能够更加快速流畅地运行。
3xx	上述两种方案的组合：既压缩圆弧程序段，也压缩 G0 程序段。

激活/取消

压缩器功能 COMPON、COMPCURV、COMPCAD 和 COMPSURF 通过 G 功能组 30 的 G 指令激活/关闭（参见“编程 (页 267)”）。

5.5.1.2 调试

参数设置

轴专用机床数据

编号	名称 $MA_	含义
MD33100	COMPRESS_POS_TOL	压缩时允许的特定轴的最大轨迹偏差

通道专用机床数据

编号	名称 $MC_	含义
MD20170	COMPRESS_BLOCK_PATH_LIMIT	可压缩 NC 程序段的最大运行长度
MD20171	SURF_BLOCK_PATH_LIMIT	使用 COMPSURF 压缩时可压缩 NC 程序段的最大运行长度
MD20172	COMPRESS_VELO_TOL	压缩时允许的最大轨迹进给率偏差
MD20173	SURF_VELO_TOL	使用 COMPSURF 压缩时允许的最大轨迹进给率偏差
MD20482	COMPRESSOR_MODE	压缩器的工作方式
MD20484	COMPRESSOR_PERFORMANCE	压缩器效率
MD20485	COMPRESS_SMOOTH_FACTOR	使用 COMPCAD 压缩用于各个动态响应模式时的平滑系数
MD20486	COMPRESS_SPLINE_DEGREE	使用 COMPCAD 压缩用于各个动态响应模式时的样条度数
MD20487	COMPRESS_SMOOTH_FACTOR_2	使用 COMPCAD 压缩用于各个动态响应模式时的回转轴平滑系数
MD28071	MM_NUM_SURF_LEVELS	COMPSURF 功能定尺寸（DRAM）
MD28072	MM_MAXNUM_SURF_GROUPS	有关轴组的 COMPSURF 功能定尺寸（DRAM）

通道专用设定数据

编号	名称 $SC_	含义
SD42470	CRIT_SPLINE_ANGLE	使用 COMPCAD 压缩时的角点极限角度
SD42471	MIN_CURV_RADIUS	使用 COMPCAD 压缩时的最小曲率半径
SD42472	MIN_SURF_RADIUS	使用 COMPSYRF 压缩时的最小曲率半径

编号	名称 $SC_	含义
SD42473	ACTNUM_SURF_GROUPS	COMPSURF 上的轴组数量
SD42475	COMPRESS_CONTUR_TOL	压缩时允许的最大轮廓偏差
SD42476	COMPRESS_ORI_TOL	压缩时刀具定向的最大偏差 **注意：** 只在定向转换有效时！
SD42477	COMPRESS_ORI_ROT_TOL	压缩时刀具旋转的最大偏差 **注意：** 只在带可旋转刀具的 6 轴机床上！

说明

角点极限角度和压缩器 COMPCAD

通过设定数据 SD42470 $SC_CRIT_SPLINE_ANGLE 设置的压缩器功能 COMPCAD 角点极限角度只是一个用于识别角点的大致角度。压缩器也会根据合理性检查将平坦的程序段过渡判定为“角点”，将“大角度”判定为“异常值”。

设置建议

压缩器功能在刀具制造和模具制造中具有重要意义。相关机床数据和设定数据的设置建议参见下表。

表格 5-1　精优曲面（Advanced Surface）/臻优曲面（Top Surface）刀具制造和模具制造的设置建议

MD	名称	说明	推荐值		注释
			精优曲面	臻优曲面	
10131	$MN_SUPPRESS_SCREEN_REFRESH	预处理过载时的显示更新特性	0		
10200	$MN_INT_INCR_PER_MM	线性轴的内部计算精度	100000		输出型为 10000
10210	$MN_INT_INCR_PER_DEG	回转轴的内部计算精度	= MD10200		输出型为 10000
10682	$MN_CONTOUR_SAMPLING_FACTOR	轮廓采样系数	1		预设置
10708	$MN_SERUPRO_MASK	程序段搜索模式	位 5 = 1		至少 20hex

MD	名称	说明	推荐值		注释
			精优曲面	臻优曲面	
11450	$MN_SEARCH_RUN_MODE	程序段搜索参数设置	位 6 = 1		至少 40hex
18360	$MN_MM_EXT_PROG_BUFFER_SIZE	外部执行时的最大缓存容量	500 ... 2000	2000	
18362	$MN_MM_EXT_PROG_NUM	可同时执行的外部程序的数量	2 ... 4	4	COMPCAD: 至少使用值 2。如果也从外部执行主程序时，使用更大的值。 说明： 设置时需要注意，MD18362 是 MD18360 的乘数。
20150	$MC_GCODE_RESET_VALUES [3]	G 功能组 4 的初始设置	3		FIFOCTRL
20150	$MC_GCODE_RESET_VALUES [19]	G 功能组 20 的初始设置	2		SOFT
20150	$MC_GCODE_RESET_VALUES [44]	G 功能组 45 的初始设置	2		UPATH（5 轴加工时）
20150	$MC_GCODE_RESET_VALUES [48]	G 功能组 49 的初始设置	1		CP
20150	$MC_GCODE_RESET_VALUES [50]	G 功能组 50 的初始设置	2		ORIAXES（5 轴加工）
20170	$MC_COMPRESS_BLOCK_PATH_LIMIT	可压缩 NC 程序段的最大运行长度	20		
20171	$MC_SURF_BLOCK_PATH_LIMIT	使用 COMPSURF 压缩时可压缩 NC 程序段的最大运行长度	-	0	预设置

MD	名称	说明	推荐值		注释
			精优曲面	臻优曲面	
20172	$MC_COMPRESS_VELO_TOL	压缩时允许的最大轨迹进给率偏差	1000		预设置
20173	$MC_SURF_VELO_TOL	使用 COMPSURF 压缩时允许的最大轨迹进给率偏差	-	1000	预设置
20443	$MC_LOOKAH_FFORM [0-1]	动态响应模式 DYNNORM / DYNPOS 中激活扩展预读功能	0		预设置
20443	$MC_LOOKAH_FFORM [2-4]	动态响应模式 DYNROUGH / DYNSEMIFIN / DYNFINISH 中激活扩展预读功能	1		
20455	$MC_LOOKAH_FUNCTION_MASK	预读功能的特殊功能	3		
20470	$MC_CPREC_WITH_FFW	可编程的轮廓精度	3		可编程的轮廓精度也在激活预控制时生效。可编程的轮廓公差 CTOL 确定了允许的轮廓差。 **说明：** 必要时可将值设为“2”，这样就能通过 SD42450 $SC_CONTPREC 获得“更大的公差（0.1 mm）”。

MD	名称	说明	推荐值		注释
			精优曲面	臻优曲面	
20476	$MC_ORISON_STEP_LENGH	ORISON 上程序段划分的轨迹长度	0.5		预设置
20478	$MC_ORISON_MODE	定向平滑模式	100		划分可编程程序段，使得整体定向平滑能均匀地运行。只划分其中压缩器激活的程序段。
20480	$MC_SMOOTHING_MODE	G64x 平滑特性	x**0**x		预设置
20482	$MC_COMPRESSOR_MODE	压缩器的工作方式	300		如果采用该设置时出现速度设定值问题，则需要使用“100”代替。
20485	$MC_COMPRESS_SMOOTH_FACTOR [0-1]	使用 COMPCAD 压缩时动态响应模式 DYNNORM / DYNPOS 中的平滑系数	0		预设置
20485	$MC_COMPRESS_SMOOTH_FACTOR [2-4]	使用 COMPCAD 压缩时动态响应模式 DYNROUGH / DYNSEMIFIN / DYNFINISH 中的平滑系数	0.0001		不应生效。

MD	名称	说明	推荐值		注释
			精优曲面	臻优曲面	
20486	$MC_COMPRESS_SPLINE_DEGREE [0-1]	使用 COMPCAD 压缩时动态响应模式 DYNNORM / DYNPOS 中的样条度数	3		预设置
20486	$MC_COMPRESS_SPLINE_DEGREE [2-4]	使用 COMPCAD 压缩时动态响应模式 DYNROUGH / DYNSEMIFIN / DYNFINISH 中的样条度数	5		
20487	$MC_COMPRESS_SMOOTH_FACTOR_2 [0-1]	使用 COMPCAD 压缩时动态响应模式 DYNNORM / DYNPOS 中的回转轴平滑系数	0		
20487	$MC_COMPRESS_SMOOTH_FACTOR_2 [2-4]	使用 COMPCAD 压缩时动态响应模式 DYNROUGH / DYNSEMIFIN / DYNFINISH 中的回转轴平滑系数	0.5		
20490	$MC_IGNORE_OVL_FACTOR_FOR_ADIS	G642、G645 的影响	1		

MD	名称	说明	推荐值		注释
			精优曲面	臻优曲面	
20560	$MC_G0_TOLERANCE_FACTOR	压缩器的 G0 公差系数，G645、OST、ORISON	3		
20600	$MC_MAX_PATH_JERK [0-4]	各个动态响应模式中的轨迹加加速度	10000		不应生效。
20602	$MC_CURV_EFFECT_ON_PATH_ACCEL [0-1]	动态响应模式 DYNNORM / DYNPOS 中的离心加速度和向心加速度的比值	0		不应生效。
20602	$MC_CURV_EFFECT_ON_PATH_ACCEL [2]	动态响应模式 DYNROUGH 中的离心加速度和向心加速度的比值	0（必要时通过圆弧测试计算）	0.65	主要是对“大型”机床上的圆弧加速度进行限制！
20602	$MC_CURV_EFFECT_ON_PATH_ACCEL [3]	动态响应模式 DYNSEMIFIN 中的离心加速度和向心加速度的比值		0.6	
20602	$MC_CURV_EFFECT_ON_PATH_ACCEL [4]	动态响应模式 DYNFINISH 中的离心加速度和向心加速度的比值		0.5	
20606	$MC_PREPDYN_SMOOTHING_ON [0-1]	激活动态响应模式 DYNNORM / DYNPOS 中的曲线平滑	0		预设置

MD	名称	说明	推荐值		注释
			精优曲面	臻优曲面	
20606	$MC_PREPDYN_SMOOTHING_ON [2-4]	激活动态响应模式 DYNROUGH / DYNSEMIFIN / DYNFINISH 中的曲线平滑	1		
22430	$MC_FGROUP_PATH_MODE	在特别情况下的轨迹速度特性	3		5 轴加工时强制要求！
22440	$MC_FGROUP_PATH_RATIO	使用 FGROUP 特别方案的方法	1		预设置
28060	$MC_MM_IPO_BUFFER_SIZE	插补缓冲器中的 NC 程序段数量	150	300	
28070	$MC_MM_NUM_BLOCKS_IN_PREP	预处理的程序段数量	80	500	
28071	$MC_MM_NUM_SURF_LEVELS	COMPSURF 功能定尺寸	-	6	
28072	$MC_MM_MAXNUM_SURF_GROUPS	有关轴组的 COMPSURF 功能定尺寸	-	2	
28520	$MC_MM_MAX_AXISPOLY_PER_BLOCK	每个程序段轴多项式的最大数量	5	15	
28530	$MC_MM_PATH_VELO_SEGMENTS	用于限制轨迹加速度的存储空间单元数量	5	8	
28533	$MC_MM_LOOKAH_FFORM_UNITS	扩展 LookAhead 的存储空间	18		
28540	$MC_MM_ARCLENGTH_SEGMENTS	用于显示曲线长度功能的存储空间单元数量	10	20	

MD	名称	说明	推荐值		注释
			精优曲面	臻优曲面	
28590	$MC_MM_ORISON_BLOCKS	最大程序段数量，通过该程序段进行定向平滑	100		5 轴加工时！ 根据 SD42678 $SC_ORISON_TOL 情况，必要时必须将该值增加多倍。
28610	$MC_MM_PREPDYN_BLOCKS	用于速度准备的程序段数量	10		
29000	$OC_LOOKAH_NUM_CHECKED_BLOCKS	LookAhead 程序段数量	150	300	必须与 MD28060 中的值相同！
42450	$SC_CONTPREC	MD20470 $MC_CPREC_WITH_FFW = 1 或 2 时的 CPRECON 轮廓精度	0.1		MD20470 = 3 时，SD42450 中的值无效！
42470	$SC_CRIT_SPLINE_ANGLE	使用 COMPCAD 压缩时的角点极限角度	36		预设置 该值必须大于 30°。
42471	$SC_MIN_CURV_RADIUS	使用 COMPCAD 压缩时的最小曲率半径	1		预设置 推荐设置范围：0.3 ... 3
42472	$SC_MIN_SURF_RADIUS [0]	轴组<n>中的平滑（几何轴）轮廓	-		预设置 = 1 由 CYCLE832 写入。 1 = 平滑 on 0 = 平滑 off

MD	名称	说明	推荐值		注释
			精优曲面	臻优曲面	
42472	$SC_MIN_SURF_RADIUS [1]	轴组<n>中的平滑（定向轴） 定向	-		预设置 = 1 由 CYCLE832 写入。 1 = 平滑 on 0 = 平滑 off
42472	$SC_MIN_SURF_RADIUS [2]	轴组<n>中的平滑（辅助轴）	-	1	
42473	$SC_ACTNUM_SURF_GROUPS	COMPSURF 上的轴组数量	-	2	
42500	$SC_IS_MAX_PATH_ACCEL	通过 SD 限制轨迹加速度	10000		
42502	$SC_IS_SD_MAX_PATH_ACCEL	通过 SD 激活轨迹加速度	0 (FALSE)		
42510	$SC_SD_MAX_PATH_JERK	通过 SD 限制轨迹加加速度	10000		
42512	$SC_IS_SD_MAX_PATH_JERK	通过 SD 激活轨迹加加速度	0 (FALSE)		
42678	$SC_ORISON_TOL	ORISON 定向平滑公差	0.1		5 轴加工时！ 该值取决于加工/循环，由 OTOL 决定！ 建议： 0：圆周铣削 0、1：端面铣削

5.5.1.3 编程

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。
...	

5.5.1.4 前提条件

定向转换（TRAORI）

定向转换激活时，压缩器可以压缩用于刀具定向和刀具旋转的运动程序段。

文档：
功能手册之特殊功能分册；章节“F2：多轴转换” > “定向” > “定向压缩”

带计算的程序段搜索

在程序段搜索**类型 2** 或**类型 4**（在 ... 处**计算**的程序段搜索）中，若目标程序段所处于的程序部分中激活了压缩器功能，那么在重定位至轮廓时会逼近通过压缩器计算出的轨迹上的位置。这些位置不必与零件程序中编写的轨迹上的位置精确一致。

在程序段搜索时不能将通过压缩替换的零件程序程序段作为目标程序段。会输出报警 15370 “未找到搜索目标”。

5.5.2 压缩较短的样条程序段

功能

在通过 CAD/CAM 系统生成样条程序段来描述复杂轮廓时，轨迹长度较长的程序段间总是会存在轨迹长度非常短的程序段。这会迫使控制系统显著降低轨迹速度。压缩较短的样条程序段功能会生成轨迹长度尽可能长的新样条程序段。

可用性

系统	可用性
SINUMERIK 840D sl	标准（基本范围）
SINUMERIK 828D	选件

调试

激活

“压缩较短的样条程序段”功能可以用于以下样条类型：

- BSPLINE
- BSPLINE / ORICURVE
- CSPLINE

通过以下通道专用机床数据激活：

MD20488 $MC_SPLINE_MODE，位 <n> = <值>

位	<值>	样条类型	压缩较短的样条程序段
0	0	BSPLINE	未生效
	1	BSPLINE	生效
1	0	BSPLINE / ORICURVE	未生效
	1	BSPLINE / ORICURVE	生效
2	0	CSPLINE	未生效
	1	CSPLINE	生效

前提条件

- 在待压缩运行程序段中，以及在这些程序段之间编写了非运行指令（例如辅助功能输出）时，可能会无法合并样条程序段。
- 可以合并成一个程序块的连续样条程序段的数量取决于程序段预处理内存的大小：MD28070 $MC_MM_NUM_BLOCKS_IN_PREP（预处理的程序段数量）

示例

为了能在执行运行程序段时达到较高的轨迹速度，针对 BSPLINE 插补激活“压缩较短的样条程序段”功能：

MD20488 $MC_SPLINE_MODE, 位 0 = 1

程序代码	注释
`N10 G1 G64 X0 Y0 Z0 F1000`	; 默认设置
`N20 G91 F10000 BSPLINE`	; 激活：B 样条
`N30 X0.001 Y0.001 Z0.001`	; 从此处开始：合并较短样条程序段
`N40 X0.001 Y0.001 Z0.001`	
`...`	

5.6 轮廓/定向公差

5.6.1 调试

5.6.1.1 参数设置

机床数据

轮廓公差/定向公差

MD33100 $MA_COMPRESS_POS_TOL[<轴>] = <值>（压缩时的最大偏差）

此轴专用机床数据用于为特定轴设置允许的最大轮廓偏差（轮廓公差）或刀具定向的角度偏差（定向公差）。此机床数据在以下功能中生效：

- 平滑：G642，G643，G644，G645
- 压缩器：COMPON，COMPCURV，COMPCAD，COMPSURF
- 值越大，可压缩成一条长程序段的短程序段的数量也就越大。

此机床数据在平滑功能 G641 中不生效。在 G641 中，可通过 ADIS 或 ADISPOS 编写的到程序段过渡处的距离生效。

平滑模式

MD20480 $MC_SMOOTHING_MODE（G64x 平滑特性）

压缩器模式

MD20482 $MC_COMPRESSOR_MODE（压缩模式）

平滑 G645

MD33120 $MA_PATH_TRANS_POS_TOL（G645 平滑中的最大轮廓偏差）

在平滑相切但曲率非恒定的程序段过渡（例如圆弧 - 直线）时生效

设定数据

通道专用轮廓公差

SD42465 $SC_SMOOTH_CONTUR_TOL（最大轮廓偏差）

通道专用定向公差

SD42466 $SC_SMOOTH_ORI_TOL（刀具定向最大角度偏差）

通过 OST 平滑时的通道专用定向公差

SD42676 $SC_ORI_SMOOTH_TOL（平滑时的定向平滑公差）

通过 ORISON 进行定向平滑时的通道专用定向公差

SD42678 $SC_ORISON_TOL（定向平滑公差）

5.6.2 编程

5.6.2.1 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...	

5.6.2.2 轮廓公差/定向公差的编程 (CTOL、OTOL、ATOL)：更多信息

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。

5.7 G0 运行的公差系数

5.7.1 功能

借助 G0 公差系数能够为 G0 运动（快速移动，进给运动）设置不同于工件加工公差的公差。

优点

为 G0 运动采用更大公差 （G0 公差系数 > 1），故能更快地运行 G0 程序段。

前提条件

仅在满足下列条件时，G0 公差系数才生效：

- 下列功能中有一个生效：
  - 压缩器功能 COMPON、COMPCURV、COMPCAD 或 COMPSURF
  - 平滑功能 G642 或 G645
  - 定向精磨 OST
  - 定向平滑 ORISON
  - 路径相关的定向平滑 ORIPATH
- 零件程序中有多个（≥ 2）相连的 G0 程序段。
  在只有一个 G0 程序段时 G0 公差系数不会生效，因为在从非 G0 运动过渡至 G0 运动（以及反之）时，原则上是“**较小的公差**”（工件加工公差）生效！

5.7.2 调试

5.7.2.1 参数设置

机床数据

G0 公差系数

G0 公差系数通过以下机床数据针对各个通道设置：

MD20560 $MC_G0_TOLERANCE_FACTOR（G0 公差系数）

G0 公差系数可大于也可小于 1.0。系数等于 1.0（缺省值）时，G0 运动启用与非 G0 运动相同的公差。通常情况下 G0 公差系数设为大于 1.0 的值。

5.7.3 编程

5.7.3.1 编程 G0 运行的公差系数 (STOLF)

借助机床数据 (MD20560 $MC_G0_TOLERANCE_FACTOR) 设置的 G0 运行的公差系数可通过在零件程序中编程 STOLF 来进行调整。此时不会修改机床数据中的值。复位或零件程序结束后，机床数据中设置的 G0 公差系数会再次生效。

句法

```
STOLF=<值>
```

含义

STOLF:	用于编程 G0 运行公差系数的地址		
	应用范围：	• 压缩器功能 COMPON、COMPCURV、COMPCAD 和 COMPSURF • 精磨功能 G642 和 G645 • 定向精磨 OST • 定向平滑 ORISON • 路径相关的定向平滑 ORIPATH	
	预处理停止：	不选择	
	生效方式：	模态	
	<Value>:	G0 公差系数 G0 公差系数可大于也可小于 1.0。系数等于 1.0（缺省值）时，G0 运行启用和非 G0 运行相同的公差。通常情况下 G0 公差系数设为大于 1.0 的值。	
		类型：	实数
		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 ⇒ 机床数据中设置的公差值重新生效。

示例

程序代码	注释
COMPCAD G645 G1 F10000	; 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...	
G0 X...Y...Z...	
G0 X...Y...Z...	; 此处机床数据 $MC_G0_TOLERANCE_FACTOR（例如 =3）生效，即 $MC_G0_TOLERANCE_FACTOR * $MA_COMPRESS_POS_TOL 的平滑公差生效。
CTOL=0.02	
STOLF=4	
G1 X...Y...Z...	; 从此处开始，0.02 毫米的轮廓公差生效。
X...Y...Z...	
X...Y...Z...	
G0 X...Y...Z...	

程序代码	注释
X...Y...Z...	；从此处开始 G0 公差系数 4 生效，即 0.08 mm 的轮廓公差生效。

系统变量

零件程序中或当前插补程序中生效的 G0 公差系数可通过系统变量读取。

- 在同步动作或在带预处理停止的零件程序中，通过系统变量：

  $AC_STOLF　　生效的 G0 公差系数
  当前主程序段预处理时生效的 G0 公差系数。

- 在不带预处理停止的零件程序中，通过系统变量：

  $P_STOLF　　编程的 G0 公差系数

如果在生效的零件程序中未使用 STOLF 赋值，则两个系统变量会输出通过 MD20560 $MC_G0_TOLERANCE_FACTOR 设置的值。

如果在程序段中无快速移动（G0），则这些系统变量总是输出值 1。

5.8 复位特性

MD20150

复位（通道复位或 BAG 复位）时，所有 G 功能组启用通道专用初始设置：

MD20150 $MC_GCODE_RESET_VALUES（G 功能组的初始设置）

和“连续路径运行、准停、预读”相关的 G 功能组有：

- 组 10：准停－连续路径模式
- 组 12：准停时的程序段切换条件
- 组 21：加速模式
- 组 30：数控程序段压缩器
- 组 59：轨迹插补的动态响应模式

有关初始设置的详细说明请参见章节“K1：BAG、通道、程序运行、复位特性 (页 519)”。

5.9 前提条件

5.9.1 程序段切换和定位轴

在一个零件程序中轨迹轴处于连续路径运行时，同时运行的定位轴可能会影响轨迹轴的特性和程序段切换。

有关定位轴的详细说明请见：

文档：

功能手册之扩展功能分册；定位轴（P2）

5.9.2 程序段切换延时

即使所有在零件程序段中运行的轨迹轴和辅助轴都满足各自特殊的程序段切换标准，其他条件不满足和/或生效的功能也可能导致程序段切换延时。

示例：

- 没有通过 PLC 应答辅助功能
- 后续程序段不存在
- 功能“清空缓冲存储器”生效

影响

如果在连续路径运行期间无法执行程序段切换，那么所有在该零件程序段中编程的轴（除了跨程序段运行的辅助轴）都会停止运行。此时不会出现轮廓误差。

在加工期间停止轨迹轴运行可导致工件表面上出现切削痕迹。

5.10 数据表

5.10.1 机床数据

5.10.1.1 通用机床数据

编号	名称：$MN_	说明
10110	PLC_CYCLE_TIME_AVERAGE	平均 PLC 应答时间
10680	MIN_CONTOUR_SAMPLING_TIME	最小轮廓采样时间
10682	CONTOUR_SAMPLING_FACTOR	轮廓采样系数
10712	NC_USER_CODE_CONF_NAME_TAB	重新配置后的 NC 指令列表
12030	OVR_FACTOR_FEEDRATE	轨迹进给倍率开关的权重系数
12100	OVR_FACTOR_LIMIT_BIN	二进制编码的倍率开关的限制
18360	MM_EXT_PROG_BUFFER_SIZE	用于执行外部程序的 FIFO 缓冲器大小 (DRAM)

5.10.1.2 通道专用机床数据

编号	名称：$MC_	说明
20150	GCODE_RESET_VALUES	组的初始设置
20170	COMPRESS_BLOCK_PATH_LIMIT	可压缩 NC 程序段的最大运行长度
20171	SURF_BLOCK_PATH_LIMIT	使用 COMPSURF 压缩时可压缩 NC 程序段的最大运行长度
20172	COMPRESS_VELO_TOL	压缩时允许的最大轨迹进给率偏差
20173	SURF_VELO_TOL	使用 COMPSURF 压缩时允许的最大轨迹进给率偏差
20400	LOOKAH_USE_VELO_NEXT_BLOCK	预读后续程序段的速度
20430	LOOKAH_NUM_OVR_POINTS	预读中倍率开关基准值的数量
20440	LOOKAH_OVR_POINTS	预读中倍率开关基准值
20443	LOOKAH_FFORM	激活扩展预读功能
20450	LOOKAH_RELIEVE_BLOCK_CYCLE	程序段处理周期的稳定系数
20455	LOOKAH_FUNCTION_MASK	预读功能的特殊功能

编号	名称：$MC_	说明
20460	LOOKAH_SMOOTH_FACTOR	预读时的平滑系数
20462	LOOKAH_SMOOTH_WITH_FEED	平滑考虑编程进给率
20465	ADAPT_PATH_DYNAMIC	轨迹动态响应自适应
20480	SMOOTHING_MODE	G64x 平滑特性
20482	COMPRESSOR_MODE	压缩器的工作方式
20484	COMPRESSOR_PERFORMANCE	压缩器效率
20485	COMPRESS_SMOOTH_FACTOR	使用 COMPCAD 压缩用于各个动态响应模式时的平滑系数
20486	COMPRESS_SPLINE_DEGREE	使用 COMPCAD 压缩用于各个动态响应模式时的样条度数
20487	COMPRESS_SMOOTH_FACTOR_2	使用 COMPCAD 压缩用于各个动态响应模式时的回转轴平滑系数
20488	SPLINE_MODE	样条插补的设置
20490	IGNORE_OVL_FACTOR_FOR_ADIS	不受过载系数影响的 G641/G642
20550	EXACT_POS_MODE	G0/G1 指令时的准停条件
20552	EXACT_POS_MODE_G0_TO_G1	在连续路径运行中快速移动程序段过渡处的准停标准
20560	G0_TOLERANCE_FACTOR	G0 公差系数
20600	MAX_PATH_JERK	最大轨迹急动度
20602	CURV_EFFECT_ON_PATH_ACCEL	轨迹曲率对轨迹动态响应的影响
20603	CURV_EFFECT_ON_PATH_JERK	轨迹曲率对轨迹急动率的影响
20606	PREPDYN_SMOOTHING_ON	激活曲率平滑
28060	MM_IPO_BUFFER_SIZE	插补缓冲器中的数控程序段数量 (DRAM)
28070	MM_NUM_BLOCKS_IN_PREP	预处理的数控程序段数量 (DRAM)
28071	MM_NUM_SURF_LEVELS	COMPSURF 功能定尺寸 (DRAM)
28072	MM_MAXNUM_SURF_GROUPS	有关轴组的 COMPSURF 功能定尺寸 (DRAM)
28520	MM_MAX_AXISPOLY_PER_BLOCK	每个程序段轴多项式的最大数量
28530	MM_PATH_VELO_SEGMENTS	程序段中用于限制轨迹加速度的存储器单元数
28533	MM_LOOKAH_FFORM_UNITS	扩展 LookAhead 的存储空间
28540	MM_ARCLENGTH_SEGMENTS	用于显示每程序段弧长函数的存储器单元数
28610	MM_PREPDYN_BLOCKS	用于速度准备的程序段数量

5.10.1.3 进给轴/主轴专用机床数据

编号	名称：$MA_	说明
32310	MAX_ACCEL_OVL_FACTOR	用于限制速度跳跃的过载系数
32431	MAX_AX_JERK	轨迹运动时的最大轴急动度
32432	PATH_TRANS_JERK_LIM	连续路径运行中程序段过渡处的最大轴急动度
32433	SOFT_ACCEL_FACTOR	SOFT 最大加速度比例
32434	G00_ACCEL_FACTOR	G00 最大加速度比例
32435	G00_JERK_FACTOR	G00 最大轴急动度比例
32440	LOOKAH_FREQUENCY	预读时的平滑极限频率
33100	COMPRESS_POS_TOL	压缩时的最大偏差
33120	PATH_TRANS_POS_TOL	G645 平滑中的最大偏差
35240	ACCEL_TYPE_DRIVE	加速特性曲线 DRIVE 开/关
36000	STOP_LIMIT_COARSE	粗准停
36010	STOP_LIMIT_FINE	精准停
36012	STOP_LIMIT_FACTOR	粗准停/精准停和静态监控系数
36020	POSITIONING_TIME	精准停延迟时间

5.10.2 设定数据

5.10.2.1 通道专用设定数据

编号	名称：$SC_	说明
42465	SMOOTH_CONTUR_TOL	平滑时的最大轮廓偏差
42466	SMOOTH_ORI_TOL	平滑时刀具定向的最大偏差
42470	CRIT_SPLINE_ANGLE	使用 COMPCAD 压缩时的角点极限角度
42471	MIN_CURV_RADIUS	使用 COMPCAD 压缩时的最小曲率半径
42472	MIN_SURF_RADIUS	使用 COMPSURF 压缩时的最小曲率半径
42473	ACTNUM_SURF_GROUPS	使用 COMPSURF 时的轴组数量
42475	COMPRESS_CONTUR_TOL	压缩时允许的最大轮廓偏差

编号	名称：$SC_	说明
42476	COMPRESS_ORI_TOL	压缩时刀具定向的最大偏差 **注意：** 只在定向转换有效时！
42477	COMPRESS_ORI_ROT_TOL	压缩时刀具旋转的最大偏差 **注意：** 只在带可旋转刀具的 6 轴机床上！
42676	ORI_SMOOTH_TOL	平滑时的定向平滑公差
42678	ORISON_TOL	定向平滑公差

5.10.3 信号

5.10.3.1 来自通道的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
所有轴停止	DB21, ...DBX36.3	DB330x.DBX4.3

5.10.3.2 来自进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
采用粗准停到达位置	DB31, ...DBX60.6	DB390x.DBX0.6
采用精准停到达位置	DB31, … .DBX60.7	DB390x.DBX0.7

B2：加速度

6

6.1 简要说明

6.1.1 简介

功能范围

该功能描述中包含以下子功能：

- 加速度
- 急动度
- 折线式加速度特性曲线

加速度和急动度

通过以下途经生效的加速度和急动度能够更好地适应机床和各种工况：通过各设定数据设置轴和通道的最大值；通过零件程序和同步动作设置加速模式、动态自适应功能和限制功能。

折线式加速度特性曲线

折线式加速度特性曲线表示：在转矩紧跟转速变速的电机（尤其是步进电机）所在的机床轴上，系统可自动下调加速度，使电机在达到最佳负载率的同时，不会出现过载。

6.1.2 特性

加速度

轴专用功能：

- 最大加速度可设置
- 加速模式可通过零件程序指令选择：
  刚性加速模式 (BRISKA)
- 通过零件程序指令设置一个最大加速度 (ACC)
- 快速移动条件下的单独最大加速度 (G00)。

- 软加速模式下的单独最大加速度
- 在非相切的程序段过渡条件下上调加速度

通道专用功能：

- 加速模式可通过零件程序指令选择：
  刚性加速模式 (BRISK)
- 恒速运行时间可设置，以避免极端加速跃变
- 为叠加运动保留的加速裕量可设置
- 最大加速度可设置
- 特定实时事件下的加速度可设置
- 为径向加速度保留的加速裕量可设置

急动度

轴专用功能：

- 加速模式可通过零件程序指令选择：
  软加速模式 (SOFTA)
- 单轴插补中的最大急动度可设置
- 轨迹插补中的最大急动度可设置

通道专用功能：

- 加速模式可通过零件程序指令选择：
  软加速模式 (SOFT)
- 最大急动度可设置
- 特定实时事件下的轨迹急动度可设置
- 快速移动条件下的单独最大急动度 (G00)
- 在非恒定曲率的程序段过渡处自动上调急动度

折线式加速度特性曲线

折线式加速度特性曲线可由以下参数设置：

- 最大速度 $v_{最大}$
- 最大加速度 $a_{最大}$
- 转矩开始回落时的速度 $v_{回落}$

- 转矩开始回落时的加速度 $a_{回落}$
- 加速度回落的曲线（恒定、抛物线、线性）

6.2 功能

6.2.1 刚性加速模式 (BRISK/BRISKA)（通道/轴专用）

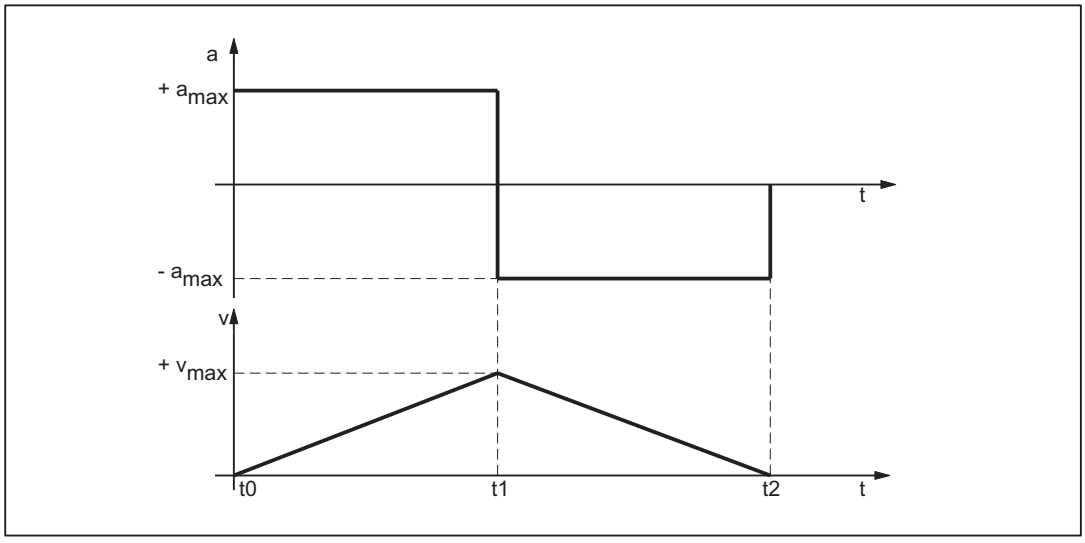
6.2.1.1 简介

简介

在刚性加速模式中，轴会立即以最大加速度加速。和软加速模式相比，有以下不同：

- **优点**
  在同等的最大速度和加速度条件下，工时更短。
- **缺点**
  机械装置承受的负载更大，难以控制高频机械振动。

加速模式



$a_{最大}$： 最大加速度

$v_{最大}$： 最大速度

t: 时间

图 6-1 刚性加速模式下的速度和加速度原理图

从上图中可发现加速模式的以下特性：

- 时间点：t_0
  加速度从 0 跳到 $+a_{最大}$
- 间隔：t_0 - t_1
  加速度保持在 $+a_{最大}$；此时速度线性增加
- 时间点：t_1
  直接从加速切换至制动时回落 2 倍的 $a_{最大}$

说明

通常可通过指定恒速运行时间（参见章节“恒速运行时间（通道专用） (页 291)”）来避免加速度跃变。

- 间隔：t_1 - t_2
  加速度保持在 $-a_{最大}$；此时速度线性降低

6.2.1.2 参数设置

轨迹运动中的最大轴加速度

轨迹运动中的每根机床轴的最大加速度可根据具体的工艺通过以下机床数据进行设置：

MD32300 $MA_MAX_AX_ACCEL[<参数组下标>]

其中，参数组下标 = 0, 1, 2 ...（最大参数组编号 - 1）

有关工艺专用参数组的详细信息请参见章节“轨迹插补的动态响应模式 (页 249)”。

在设计轨迹时，系统会依据上述设置的最大值（该值不要超过参与轨迹运行的机床轴的最大值）来计算轨迹参数。

说明

在特定条件下允许超过最大值（参见章节“加速度自适应 (ACC)（轴专用） (页 293)”和“针对实时事件的轨迹加速度（通道专用） (页 296)”）。

定位轴运动中的最大轴加速度

在定位轴运动中，根据设定的定位轴动态模式，下列最大值中的一个生效：

- MD32300 $MA_MAX_AX_ACCEL [0]（动态模式 DYNNORM 中轨迹运动中的最大单轴加速度）
- MD32300 $MA_MAX_AX_ACCEL [1]（动态模式 DYNPOS 中轨迹运动中的最大单轴加速度）

定位轴动态模式在以下 NC 专用机床数据中设置：

MD18960 $MN_POS_DYN_MODE = <模式>

<模式>	含义
0	生效的最大单轴加速度：MD32300 $MA_MAX_AX_ACCEL[0]
1	生效的最大单轴加速度：MD32300 $MA_MAX_AX_ACCEL[1]

JOG 最大轴加速度

在 JOG 模式下，可为每根机床轴配置一个 JOG 专用的最大加速度（参见章节“JOG 运动中的加速度和急动度 (页 325)”）。

6.2.1.3 编程

轨迹刚性加速模式 (BRISK)

句法

```
BRISK
```

功能

通过零件程序指令 BRISK 可选择“刚性”加速模式。

G 功能组：21

有效性：模态有效

复位特性

进行复位操作后，以下通道专用的初始设置生效：

MD20150 $MC_GCODE_RESET_VALUES[20]

前提条件

如果中途在零件程序中切换了加速模式 (BRISK / SOFT)，系统会在程序段结束时执行一个准停。

单轴刚性加速模式 (BRISKA)

句法

BRISKA (*轴{,轴}*)

功能

通过零件程序指令 BRISKA 可为单轴运动（JOG、JOG/INC、定位轴、摆动轴等）选择“刚性”加速模式。

G 功能组：-

有效性： 模态有效

轴：

- 取值范围：通道轴名称

轴专用的初始设置

刚性加速模式可设为单轴运动中的初始设置：

MD32420 $MA_JOG_AND_POS_JERK_ENABLE = FALSE

复位特性

进行复位操作后，以下轴专用的初始设置生效：

MD32420 $MA_JOG_AND_POS_ENABLE

6.2.2 恒速运行时间（通道专用）

6.2.2.1 简介

概述

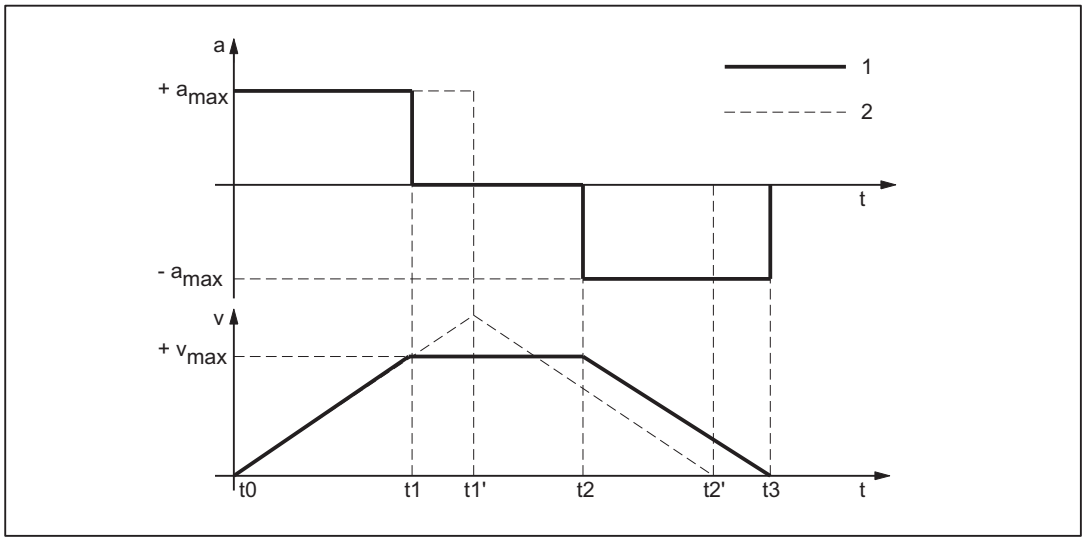
在刚性加速模式中，从加速切换至制动时，加速度会回落 2 倍的 $a_{最大}$。可设置一个通道专用的恒速运行时间以避免该加速度跃变。该时间就是指加速阶段和制动阶段之间的恒速运行时间：

MD20500 $MC_CONST_VELO_MIN_TIME（恒速最小时间）

说明

恒速运行时间在以下情况中不起作用：

- 生效的功能：预读
- 运行程序段的运行时间小于或等于差补周期。



1： 带恒速运行时间的运行

2： 不带恒速运行时间的运行

$a_{最大}$： 最大加速度

$v_{最大}$： 最大速度

t: 时间

图 6-2 刚性加速模式下的原理图

从上图中可发现恒速运行时间的作用：

- 时间点：t_1
  加速阶段结束，加速度跳跃 1 倍的 $a_{最大}$
- 间隔：t_1 - t_2
  加速度 0；通过设置恒速运行时间达到恒定速度
- 时间点：t_2
  制动阶段开始，加速度跳跃 1 倍的 $a_{最大}$

时间点 t_0、t_1' 和 t_2' 表示的是没有恒速运行时间时的运行。

6.2.2.2 参数设置

恒速运行时间是通过以下机床数据设置的：

MD20500 $MC_CONST_VELO_MIN_TIME
（恒速最小时间）

6.2.3 加速度自适应 (ACC)（轴专用）

6.2.3.1 简介

功能

通过指令 ACC 可针对具体的轴将在加速度专用机床数据中设置的当前生效的轴最大加速度降低。 降低通过设置百分比系数实现，该系数在进行指令编程时一同给定。

例如，可在加工程序段中对最大可能的轴加速度进行降低，以避免因巨大的加工力度而造成的机械振动。

有效性

使用 ACC 的加速度降低在 AUTO、MDA 以及空运行进给模式下的所有插补方式都**起效**。

在下面情况中加速度降低**无效**：

- JOG 运行方式下
- 执行 REF（回参考点）机床功能期间。
- 由于检测到故障，轴通过快速停止（设定值 = 0）而静止时。

6.2.3.2 编程

句法

```
ACC[<轴>]=<降低系数>
ACC[SPI(<主轴编号>)]=<降低系数>
ACC(S<主轴编号>)=<降低系数>
```

含义

<ACC>:	针对具体轴的从机床数据中导出的可降低当前最大可能加速度的指令	
	有效性：	模态
<轴>：	轨迹轴的通道轴名称	
	数据类型：	AXIS
	值范围：	通道轴名称
SPI(<主轴编号>):	函数 SPI(...) 可将主轴编号转换为相应的通道轴名称。	
S<主轴编号>:	通道中的主轴名称	

其他信息

系统变量

通道中当前生效的，使用 ACC 设置的，加速度降低可针对具体轴通过以下指令读取：

```
$AA_ACC[<轴>]
```

复位特性

使用 ACC 设置的加速度降低不会因通道复位或程序结束而变化。这通过以下机床数据设置：

MD32320 $MA_DYN_LIMIT_RESET_MASK

6.2.4 加速裕量（通道专用）

6.2.4.1 简介

简介

通常，轨迹加速度的预处理会 100% 的利用设置的机床轴的最大值。但也可以预留一定的轨迹加速度以便在“快速离开工件轮廓”功能范畴内为叠加运动保留一定的加速裕量。例如：当系数设为 0.2 时，预处理最多只能利用 80% 的轨迹加速度。剩余 20% 的加速裕量会保留给叠加运动。

6.2.4.2 参数设置

加速裕量的设置是通过以下机床数据进行的：
MD20610 $MC_ADD_MOVE_ACCEL_RESERVE
（为叠加运动保留的加速裕量）

6.2.5 最大轨迹加速度（通道专用）

6.2.5.1 简介

简介

为了灵活应对各种工况，可通过设定数据来限制由预处理计算出的加速度：

SD42500 $SC_SD_MAX_PATH_ACCEL（最大轨迹加速度）

只有当在设定数据中设置的值小于由预处理计算出的轨迹加速度时，系统才会考虑该值。

该限制必须由以下设定数据来激活：

SD42502 $SC_IS_SD_MAX_PATH_ACCEL = TRUE

6.2.5.2 参数设置

设置是通过以下设定数据进行的：

SD42500 $SC_SD_MAX_PATH_ACCEL（最大轨迹加速度）

SD42502 $SC_IS_SD_MAX_PATH_ACCEL（激活最大轨迹加速度）

6.2.5.3 编程

最大值

句法

```
$SC_SD_MAX_PATH_ACCEL = 最大值
```

功能

最大轨迹加速度可通过编程设定数据来调整。

最大值：

- 值范围：≥ 0
- 单位：m/s^2

可用于：

- 零件程序
- 静态同步动作

激活/关闭

句法

```
$SC_IS_SD_MAX_PATH_ACCEL = 值
```

功能

最大轨迹加速度可通过编程设定数据来启用/关闭。

参数：*值*

- 值范围：TRUE、FALSE

可用于：

- 零件程序
- 静态同步动作

6.2.6 针对实时事件的轨迹加速度（通道专用）

6.2.6.1 简介

简介

由于无需兼顾加速度在加工质量方面和时间方面的最优性，在以下实时事件中：

- NC 启停
- 进给倍率变化
- “Safety Integrated” 功能范畴内的速度设定值发生变化

可通过通道专用的系统变量来设定针对上述事件的轨迹加速度：

`$AC_PATHACC = `*轨迹加速度*

针对实时事件的轨迹加速度只会在因上述事件导致速度变化的这段时间内生效。

限制

一旦指定的轨迹加速度超出了参与到轨迹运行的机床轴的加速能力，控制系统就会对轨迹加速度进行限制，使得生成的轴加速度 ($a_{回落}$) 最高只能是设置的最大值 ($a_{最大}$) 的 2 倍。

$a_{回落}$ = 2 倍的 $a_{最大}$, $a_{最大}$ = MD32300 $MA_MAX_AX_ACCEL

说明

针对实时事件的轨迹加速度无需考虑径向加速度。

有效性

起作用	针对实时事件的轨迹加速度只有涉及到以下实时事件时才能在 AUTO 和 MDA 模式下起作用： • NC 启停 • 倍率变化 • “Safety Integrated” 功能范畴内的速度设定值发生变化
不起作用	针对实时事件的轨迹加速度对由于通道中的预处理功能重新设计轨迹（如轮廓弯曲、拐角、运动转化限制等）导致的轨迹速度变化不起作用。 当编程的值小于由预处理为相关的轨迹段计算出的轨迹加速度时，针对实时事件的轨迹加速度不起作用。

编程

有关在零件程序或同步动作中设置系统变量的详细信息请参见章节“编程 (页 297)”。

6.2.6.2 编程

句法

```
$AC_PATHACC = 轨迹加速度
```

功能

针对实时事件的轨迹加速度是通过通道专用的系统变量来设定的。

参数： *轨迹加速度*

- 值范围：轨迹加速度 ≥ 0
- 单位：m/s^2

关闭 `$AC_PATHACC = 0`

可用于：

- 零件程序
- 静态同步动作

复位特性

复位时系统会关闭针对实时事件的轨迹加速度。

前提条件

在零件程序中设置 $AC_PATHACC 时，系统会通过 Reorg 指令自动触发一个预处理停止 (STOPRE)。

6.2.7 快速移动条件下的单独加速度 (G00)（轴专用）

6.2.7.1 简介

通常来说，因加工条件的不同，参与加工的机床轴的加速度应设得比实际的加速能力低。

为了使快速移动条件下的加速时间最优化（零件程序指令 G00），可为机床轴设置一个单独最大加速度。

JOG 手动运行模式

在 JOG 手动运行模式中进行快速叠加时的加速度不受功能的影响。

6.2.7.2 参数设置

快速移动条件下的单独加速度 (G00) 是由以下轴专用机床数据设置的：

MD32434 $MA_G00_ACCEL_FACTOR
（G00 G00 最大加速度比例）

预处理中重新设计轨迹时所考虑的快速移动 (G00) 条件下的最大单独加速度是按如下方式计算的：

加速度[轴] =

MD32300 $MA_MAX_AX_ACCEL * MD32434 $MA_G00_ACCEL_FACTOR

6.2.8 软加速模式 (SOFT/SOFTA)（轴专用）

6.2.8.1 简介

功能

在最大加速度相同的条件下，相比刚性加速模式，软加速模式中的加工时间更长。可为软加速模式 (SOFT/SOFTA) 设置一个单独的轴加速度以缩短该时间。

该软加速模式中的最大加速度是通过相对于轴最大加速度的系数来设置的。因此，预处理中重新设计轨迹时所考虑的最大加速度是按如下方式计算的：

加速度[轴] =

MD32300 $MA_MAX_AX_ACCEL * MD32433 $MA_SOFT_ACCEL_FACTOR

6.2.8.2 参数设置

软加速模式 (SOFT/SOFTA) 中的最大加速度是由以下轴专用机床数据设置的：

MD32434 $MA_SOFT_ACCEL_FACTOR
（SOFT 指令时的最大加速度比例）

6.2.9 在非相切的程序段过渡条件下上调加速度（轴专用）

6.2.9.1 简介

功能

在非相切的程序段过渡（拐角）条件下，控制系统有时必须对几何轴进行突然制动，以保持设置的轴动态响应。为避免这种制动，可以上调加速度。

加速度上调是通过相对于轴最大加速度的系数来设置的。因此，预处理中重新设计轨迹时所考虑的非相切的程序段过渡条件下的最大加速度是按如下方式计算的：

加速度[轴] =

MD32300 $MA_MAX_AX_ACCEL * MD32310 $MA_MAX_ACCEL_OVL_FACTOR

6.2.9.2 参数设置

在非相切的程序段过渡条件中上调加速度是由以下轴专用机床数据设置的：
MD32310 $MA_MAX_ACCEL_OVL_FACTOR
（速度跃变的过载系数）

6.2.10 为径向加速度保留的加速裕量（通道专用）

6.2.10.1 简介

概述

弯曲的轮廓上除了产生轨迹加速度（切向加速度）以外，也会产生径向加速度。 如果在设置轨迹参数时未考虑到这一点，那么在加速或制动期间，弯曲轮廓上的轴加速度短时间内会达到最大加速度的 2 倍。

生效的轴加速度 =
轨迹加速度分量 + 径向加速度分量 =
2 倍的 (MD32300 $MA_MAX_AX_ACCEL)

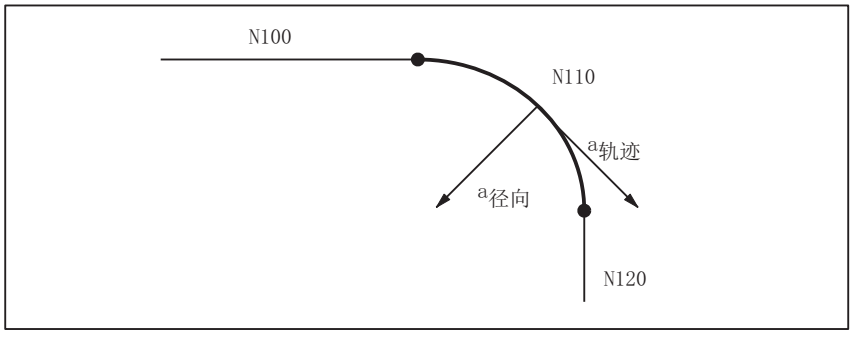


图 6-3 弯曲轮廓上的径向加速度和轨迹加速度

通过以下通道专用机床数据：
MD20602 $MC_CURV_EFFECT_ON_PATH_ACCEL
（轨迹曲率对轨迹动态响应的影响）
可以设置轴加速度的分量，该分量应作为保留给径向加速度的加速裕量。

以值为 0.75 为例，75% 的轴专用加速度分配给了径向加速度，剩余 25% 分配给了轨迹加速度。

通常，最大加速度是按以下方式计算的：

径向加速度 =
MD20602 $MC_CURV_EFFECT_ON_PATH_ACCEL * MD32300 $MA_MAX_AX_ACCEL

轨迹加速度 =
(1 - MD20602 $MC_CURV_EFFECT_ON_PATH_ACCEL) * MD32300 $MA_MAX_AX_ACCEL

示例

设定了以下下机床参数：

- MD32300 $MA_MAX_AX_ACCEL 用于所有几何轴： 3 m/s
- 基于机床的机械条件，轨迹半径为 10 mm 时的最大轨迹速度为： 5 m/min

径向加速度是按以下方式计算的：

$$a_{径向} = \frac{v^2_{轨迹}\ [m/min]}{r\ [mm] * 3.6\ [m/s^2]} = \frac{5^2}{10 * 3.6} = 0.694\ m/s^2$$

加速裕量是按如下方式设置的：

$$MD20602\ \$MC\_CURV\_EFFECT\_ON\_PATH\_ACCEL = \frac{a_{径向}\ [m/s^2]}{MD32300\ \$MA\_MAX\_AX\_ACCEL\ \ [m/s^2]} = \frac{0.694}{3} \approx 0.23$$

线性程序段

在不带运动转换的线性程序段（直线插补）上，线性加速裕量不起作用。

6.2.10.2 参数设置

最大轴加速度的分量（该分量应作为弯曲轮廓上为径向加速度保留的加速裕量）是通过以下通道专用机床数据来设置的：

MD20602 $MC_CURV_EFFECT_ON_PATH_ACCEL
（轨迹曲率对轨迹动态响应的影响）

6.2.11 轨迹插补中的最大急动度 (SOFT)（通道专用）

6.2.11.1 简介

概述

该功能手册中描述的其他功能始终是针对加速度恒定和软加速模式（急动度 = 无穷值）。在软加速模式中，加速度会在 0 到最大值之间进行线性插补。

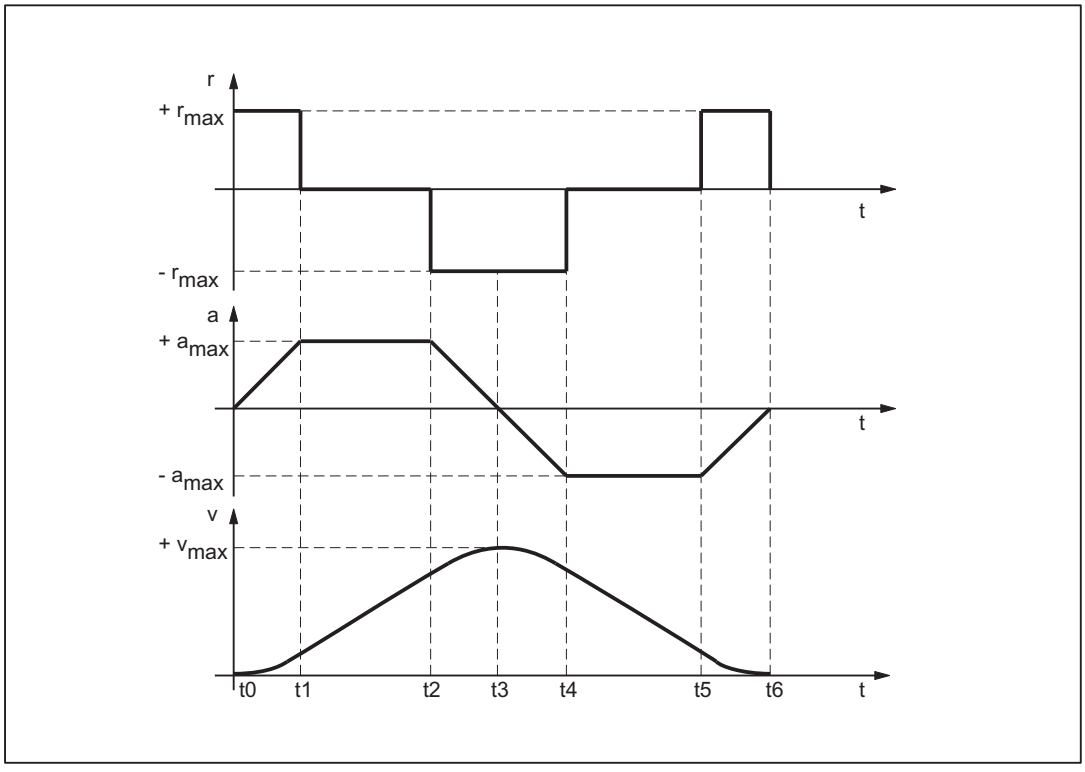
优点

机械装置承受的负载更小，不存在难以控制的高频机械振动。

缺点

在同等的最大速度和加速度条件下，工时更长。

加速模式



$r_{最大}$： 最大急动度

$a_{最大}$： 最大加速度

$v_{最大}$： 最大速度

t: 时间

图 6-4 软加速模式下的急动度、加速度和速度原理图

从上图中可发现加速模式的以下特性：

- 间隔：t_0 - t_1
  急动度保持在 $+r_{最大}$；此时加速度线性增加；此时速度平方增加
- 间隔：t_1 - t_2
  加速度保持在 $+a_{最大}$；此时速度线性增加
- 间隔：t_2 - t_3
  急动度保持在 $-r_{最大}$；此时加速度线性降低；此时速度提升至最大值平方降低 $+v_{最大}$
- 间隔：t_3 - t_4
  急动度保持在 $-r_{最大}$；此时制动加速度线性增加；此时速度平方降低

- 间隔：t_4 - t_5
  制动加速度保持在 $-a_{最大}$；此时速度线性降低
- 间隔：t_5 - t_6
  急动度保持在 $+r_{最大}$；此时制动加速度线性降低；此时速度平方降低至静止状态 v = 0

6.2.11.2 参数设置

轨迹运动的最大急动度（轴专用）

轨迹运动中的每根机床轴的最大急动度可根据具体的工艺通过以下机床数据进行设置：

MD32431 $MA_MAX_AX_JERK[<参数组下标>]

其中，参数组下标 = 0, 1, 2 ...（最大参数组编号 - 1）

有关工艺专用参数组的详细信息请参见章节“轨迹插补的动态响应模式 (页 249)”。

在设计轨迹时，系统会依据上述设置的最大值（该值不要超过参与轨迹运行的机床轴的最大值）来计算轨迹参数。

说明

在特定条件下允许超过最大值（参见章节“针对实时事件的轨迹急动度（通道专用） (页 308)”）。

轨迹运动的最大急动度（通道专用）

除了能够设置轴专用的最大急动度，也可以通过以下机床数据将最大急动度预设为通道专用的轨迹参数：

MD20600 $MC_MAX_PATH_JERK（与轨迹相关的最大急动度）

为了避免轴专用和通道专用的最大急动度之间相互影响，通道专用的最大急动度必须大于轴专用的最大急动度。

6.2.11.3 编程

句法

```
SOFT
```

功能

通过零件程序指令 SOFT 可为通道中的几何轴运行选择软加速模式。

G 功能组：21

有效性：模态有效

复位特性

进行复位操作后，以下通道专用的初始设置生效：

MD20150 $MC_GCODE_RESET_VALUES[20]

前提条件

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。

6.2.12 单轴插补中的最大急动度 (SOFTA)（轴专用）

6.2.12.1 参数设置

最大轴急动度的初始设置

软加速模式可设为轴的初始设置：

MD32420 $MA_JOG_AND_POS_JERK_ENABLE== TRUE

定位轴运动中的最大轴急动度

在软加速模式中运行定位轴时，以下机床数据中的其中一个值将作为最大轴急动度生效：

- MD32430 $MA_JOG_AND_POS_MAX_JERK（定位轴运动中的最大轴急动度）
- MD32431 $MA_MAX_AX_JERK [0]（动态模式 DYNNORM 下的轨迹运动中的最大轴急动度）
- MD32431 $MA_MAX_AX_JERK [1]（动态模式 DYNPOS 下的轨迹运动中的最大轴急动度）

所设的定位轴动态模式决定了使用哪个机床数据：

MD18960 $MN_POS_DYN_MODE = <模式>

<模式>	含义
	作为定位轴运动中的最大轴急动度生效：
0	MD32430 $MA_JOG_AND_POS_MAX_JERK G75 生效时（运行至固定点）：MD32431 $MA_MAX_AX_JERK[**0**]
1	MD32431 $MA_MAX_AX_JERK[**1**]

JOG 运动中的最大轴急动度

在 JOG 模式下，可为每根机床轴配置一个 JOG 专用的最大急动度（参见章节“JOG 运动中的加速度和急动度 (页 325)”）。

6.2.12.2 编程

句法

SOFTA (*轴* {*轴*})

功能

通过零件程序指令 SOFTA 可为单轴运动（定位轴、摆动轴等）选择软加速模式。

G 功能组-

有效性： 模态有效

轴:

- 取值范围：通道轴名称

轴专用的初始设置

软加速模式可设为单轴运动中的初始设置：

MD32420 $MA_JOG_AND_POS_JERK_ENABLE = TRUE

复位特性

进行复位操作后，以下轴专用的初始设置生效：

MD32420 $MA_JOG_AND_POS_ENABLE

6.2.13 最大轨迹急动度（通道专用）

6.2.13.1 简介

概述

为了灵活应对各种工况，可通过设定数据来限制由预处理计算出的急动度：

SD42510 $SC_SD_MAX_PATH_JERK（最大轨迹急动度）

只有当在设定数据中设置的值小于由预处理计算出的轨迹急动度时，系统才会考虑该值。

该限制必须由以下设定数据来激活：

SD42512 $SC_IS_SD_MAX_PATH_JERK = TRUE

6.2.13.2 参数设置

设置是通过以下设定数据进行的：

SD42510 $SC_SD_MAX_PATH_JERK（最大轨迹急动度）

SD42512 $SC_IS_SD_MAX_PATH_JERK
（激活最大轨迹急动度）

6.2.13.3 编程

最大轨迹急动度

句法

$SC_SD_MAX_PATH_JERK = *急动度值*

功能

最大轨迹急动度可通过编程设定数据来调整。

急动度值：

- 值范围：≥ 0
- 单位：m/s^3

可用于：

- 零件程序
- 静态同步动作

激活/关闭

句法

$SC_IS_SD_MAX_PATH_JERK = *值*

功能

最大轨迹急动度可通过编程设定数据来启用/关闭。

参数：*值*

- 值范围：TRUE、FALSE

可用于：

- 零件程序
- 静态同步动作

6.2.14 针对实时事件的轨迹急动度（通道专用）

6.2.14.1 简介

概述

由于无需兼顾加速度在加工质量方面和时间方面的最优性，在以下实时事件中：

- NC 启停
- 进给倍率变化
- “Safety Integrated” 功能范畴内的速度设定值发生变化

可通过通道专用的系统变量来设定针对上述事件的轨迹急动度：

$AC_PATHJERK = *轨迹急动度*

针对实时事件的轨迹急动度只会在因上述事件导致速度变化的这段时间内生效。

限制

由于急动度不是相关驱动的物理量，**因此不**限制预设的急动度。

有效性

起作用	针对实时事件的轨迹急动度只有涉及到以下实时事件时才能在 AUTO 和 MDA 模式下起作用： • NC 启停 • 倍率变化 • “Safety Integrated” 功能范畴内的速度设定值发生变化
不起作用	针对实时事件的轨迹急动度对由于通道中的预处理功能重新设计轨迹（如轮廓弯曲、拐角、运动转化限制等）导致的轨迹速度变化不起作用。 当编程的值小于由预处理为相关的轨迹段计算出的轨迹急动度时，针对实时事件的轨迹急动度不起作用。

编程

为了根据加速度来设置实时事件的急动度，可按以下步骤来设置系统变量：

$AC_PATHJERK = $AC_PATHACC / 平滑时间

- $AC_PATHACC：轨迹加速度 [m/s²]
  平滑时间：可随意选择，如 0.02 s

有关在零件程序或同步动作中设置系统变量的详细信息请参见章节“编程 (页 309)”。

6.2.14.2 编程

句法

$AC_PATHJERK = *急动度值*

功能

针对实时事件的轨迹急动度是通过通道专用的系统变量来设定的。

急动度值：

- 取值范围：轨迹急动度 ≥ 0
- 单位： m/s^3

可用于：

- 零件程序
- 静态同步动作

复位特性

复位时系统会关闭该功能。

前提条件

在零件程序中设置 $AC_PATHJERK 时，系统会通过 Reorg 指令自动触发一个预处理停止 (STOPRE)。

6.2.15 快速移动条件下的急动度 (G00)（轴专用）

6.2.15.1 简介

概述

通常来说，因加工条件的不同，参与加工的机床轴的最大急动度应设得比实际的能力低。

为了使快速移动条件下加速的时间最优化（零件程序指令 G00），可为机床轴设置一个最大轴急动度。

JOG 手动运行模式

在 JOG 运行模式中进行快速叠加时的急动度不受功能的影响。

6.2.15.2 参数设置

快速移动条件下的急动度 (G00) 是由以下轴专用机床数据设置的：

MD32434 $MA_G00_ACCEL_FACTOR
（G00 指令时的最大加速度比例）

预处理中重新设计轨迹时所考虑的快速移动 (G00) 条件下的最大急动度是按如下方式计算的：

急动度[轴] =
MD32431 $MA_MAX_AX_JERK * MD32435 $MA_G00_JERK_FACTOR

6.2.16 在非恒定曲率的程序段过渡处自动上调急动度（轴专用）

6.2.16.1 简介

概述

在非相切的程序段过渡（拐角）条件下，控制系统有时必须对几何轴进行突然制动，以保持设置的轴动态响应。为避免这种制动，可以上调加速度。

上调急动度是通过一个单独的最大急动度来设置的。

6.2.16.2 参数设置

在非恒定曲率的程序段过渡处自动上调急动度是通过以下轴专用机床数据设置的：

MD32432 $MA_PATH_TRANS_JERK_LIM
（在非恒定曲率的程序段过渡处自动上调急动度）

6.2.17 随速度提高急动度（轴专用）

功能

轨迹动态响应是由参与到轨迹运动中恒定的轴的最大速度、最大加速度和最大急动度计算得出的：

- MD32000 $MA_MAX_AX_VELO（最大轴速度）
- MD32300 $MA_MAX_AX_ACCEL（最大轴加速度）
- MD32431 $MA_MAX_AX_JERK（轨迹运动中的最大轴急动度）

在**加工非恒定曲率的轮廓**时，如加工任意形状的平面时，轴急动度是导致轨迹速度波动的主要因素，尤其是在高速区。该波动会导致工件的表面质量下降。

随速度相应地提高轴急动度（ 随速度调整急动度功能） 可避免该波动。在有些条件下设置正确时可完全避免该波动。

该功能对允许的最大轨迹加速度和轨迹急动度没有任何的影响。这两个值即使是该功能激活时也可以从机床数据中设置的最大值算出。

由于直线运动中的曲率和挠度都为零，因此该功能对直线运动没有任何影响。

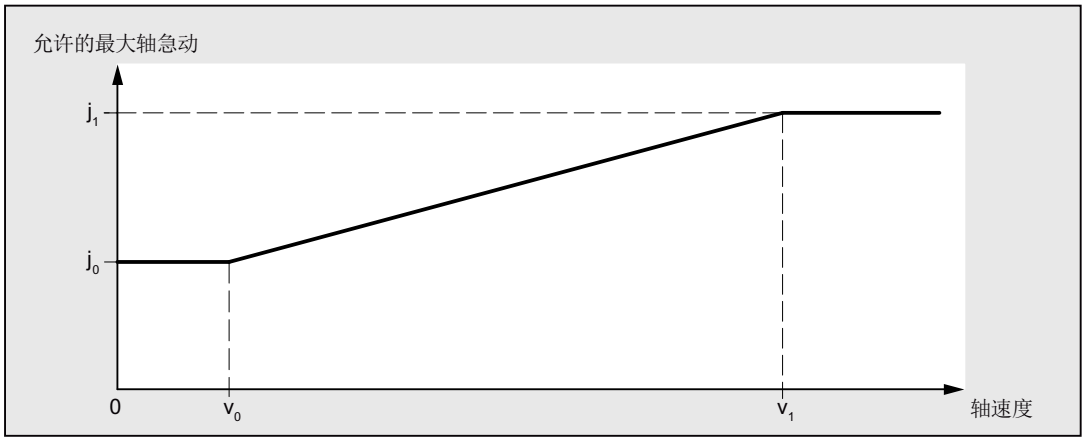
可用性

“随速度提高急动度”功能可独立于“任意形状表面模式：基本功能 (页 252)”功能存在。

参数设置

“随速度提高急动度”功能是通过以下机床数据设置的：

- MD32437 $MA_AX_JERK_VEL0[<n>] = <阈值$_{下限}$>
  急动度的速度阈值下限。从该速度起，“随速度提高急动度”功能生效。
  可通过下标 n 为每个动态模式（参见章节“轨迹插补的动态响应模式 (页 249)”）单独设置速度阈值下限：
- MD32438 $MA_AX_JERK_VEL1[<n>] = <阈值$_{上限}$>
  急动度的速度阈值上限。随速度提高的急动度在该速度时达到了它通过 MD32439 $MA_MAX_AX_JERK_FACTOR 所设的最大值 $j_{最大}$。
  可通过下标 n 为每个动态模式（参见章节“轨迹插补的动态响应模式 (页 249)”）单独设置速度阈值上限：
- MD32439 $MA_MAX_AX_JERK_FACTOR = <系数>
  用于在达到速度阈值上限 MD32438 $MA_AX_JERK_VEL1[<n>] 时设置随速度提高的最大急动度 $j_{最大}$ 的系数：
  $j_{最大}$ = (MD32431 $MA_MAX_AX_JERK) * (MD32439 $MA_MAX_AX_JERK_FACTOR)
  值 > 1.0 时，随速度提高急动度有效。
  值 = 1.0 时，随速度提高急动度无效。



v_0 MD32437 $MA_AX_JERK_VEL0

:

v_1 MD32438 $MA_AX_JERK_VEL1

:

j_0: MD32431 $MA_MAX_AX_JERK

j_1: MD32439 $MA_MAX_AX_JERK_FACTOR * MD32431 $MA_MAX_AX_JERK

图 6-5 “轴速度-急动度”函数图

说明

随速度提高急动度只在以下条件时有效：

MD32439 $MA_MAX_AX_JERK_FACTOR > 1.0

示例

设置示例：

- MD32437 $MA_AX_JERK_VEL0 = 3000 mm/min
- MD32438 $MA_AX_JERK_VEL1 = 6000 mm/min
- MD32439 $MA_MAX_AX_JERK_FACTOR[AX**1**] = 2.0
- MD32439 $MA_MAX_AX_JERK_FACTOR[AX**2**] = 3.0
- MD32439 $MA_MAX_AX_JERK_FACTOR[AX**3**] = 1.0

影响

- 随速度提高急动度对第 1 和第 2 根轴有效，而该函数对第 3 根轴无效。
- 当轴速度位于 0 到 3000 mm/min 范围内时，所设的急动度生效。

- 当轴速度位于 3000 mm/min 到 6000 mm/min 范围内时，最大急动度会直线提高。
- 当速度超过 6000 mm/min 时，第 1 根轴上允许的最大急动度是原先的 2 倍，第 2 根轴上允许的最大急动度是原先的 3 倍。
- 所设的值在每个动态模式中都有效。

6.2.18 急动度滤波器（轴专用）

6.2.18.1 简介

概述

在某些应用情况中，如对任意形状的平面进行铣削时，对机床轴的位置设定值变化进行平滑可能会非常有用。平滑可减轻机床上的机械振动，从而达到更高的表面质量。

为进行该平滑，可激活位于位置控制级的急动度滤波器，该滤波器不受无需通道专用的或轴专用的最大急动度影响，该值在插补级中考虑。

必须在不影响轮廓精度的条件下尽可能的加强急动度滤波器的效果，另外应尽可能的达到“对称”的平滑效果。也就是说，如果向前或向后运行同一个轮廓，经过滤波器平滑后的轮廓在两个方向上应尽可能的相似。

为了使急动度滤波器能够更好的适应不同的机床条件，有以下几种滤波器模式可供选择：

- 二阶滤波器 (PT2)
- 滑动求平均值
- 带阻滤波器

模式：二阶滤波器

二阶滤波器是一个简单的低通滤波器，只有在相对较小的滤波器时间常数（大约 10 ms）的条件下，才能满足上述要求。当时间常数较大时，系统会迅速产生过大的轮廓偏差。滤波效果相对较差。

当要求非常高的滤波时间常数而轮廓精度处于次要地位（如定位轴）时，该模式具有优势。

基于历史原因，这种滤波器模式仍是缺省设置。

模式：滑动求平均值

在轮廓偏差较小时，通过滑动求平均值这种滤波器模式可将滤波器恒定时间设置在 20 - 40 ms 范围内。平滑效果也是非常对称的。

操作界面中显示的计算出的控制回路增益系数（K_V 系数）低于根据滤波效果测出的值。 轮廓精度高于显示的 K_V 系数所对应的水准。

从“二阶滤波器”模式切换至“滑动求平均值”模式时，在滤波器时间常量相同的条件下，尽管轮廓精度提高了，但所显示的 K_V 系数会有所降低。

模式：带阻滤波器

带阻滤波器是一个二阶滤波器，以分子和分母的形式表示：

$$H(s) = \frac{\frac{s^2}{(2 * \pi * f_Z)^2} + \frac{2 * s * D_Z}{(2 * \pi * f_Z)}}{\frac{s^2}{(2 * \pi * f_N)^2} + \frac{2 * s * D_N}{2 * \pi * {}_N f}}$$

其中：

f_Z： 分子固有频率

f_N： 分母固有频率

D_Z： 分子阻尼系数

D_N： 分母阻尼系数

为避免易振动的滤波器设置也能提供可用的结果，和低通“二阶滤波器” (PT2) 模式一样，该模式也不提供 D_N 的设置选项，即 D_N 固定为 1。

有 2 种设置带阻滤波器的方式：

- 真正的带阻滤波器
- 高频条件下幅值响应相应放大或衰减的带阻滤波器

真正的带阻滤波器

当分子和分母固有频率相同时，就可以获得真正的带阻滤波器：

$f_Z = f_N = f_{截止}$（截止频率）

当分子阻尼系数为 0 时，就可以获得最大程度的衰减。3dB 阻带宽度是由以下方式得出的：

$f_{3dB\ 阻带宽度} = 2 * f_{截止}$

如果不需要进行一个完整的衰减，只是希望按系数 k 衰减，则必须选择与分子阻尼系数相应的 k。因此，上述 3dB 的阻带宽度就不再有效。

高频条件下幅值响应相应放大或衰减的带阻滤波器

在这种情况下，分子和分母固有频率的设置是不同的。此时，分子固有频率确定了截止频率。

选择一个比分子固有频率更高或更低的分母固有频率，就可以在高频条件下相应地衰减或放大幅值响应。在大多数情况下，高频条件下幅值响放大会非常有用，因为被控对象本身具有低通特性，也就是说：高频下幅值会衰减。

前提条件

如果分子固有频率过大，滤波器会自动关闭。此时，极限频率 $f_{Z\,最大}$ 取决于位置控制器周期。

$$f_{Zmax} = \frac{1}{2 * \pi * {}_{Zm}T_n} = \frac{1}{2 * \pi * T_{位置控制器周期}}$$ （香农定理）

6.2.18.2 参数设置

激活

急动度滤波器是通过以下机床数据激活的：

MD32400 $MA_AX_JERK_ENABLE（最大轴急动度）

急动度滤波器在每个运行方式和插补方式中都有效。

滤波器模式

滤波器模式是通过以下机床数据选择的：

MD32402 $MA_AX_JERK_MODE

值	滤波器模式
1	二阶滤波器
2	滑动求平均值
3	带阻滤波器

时间常数

急动度滤波器的时间常数是通过以下机床数据设置的：

MD32410 $MA_AX_JERK_TIME

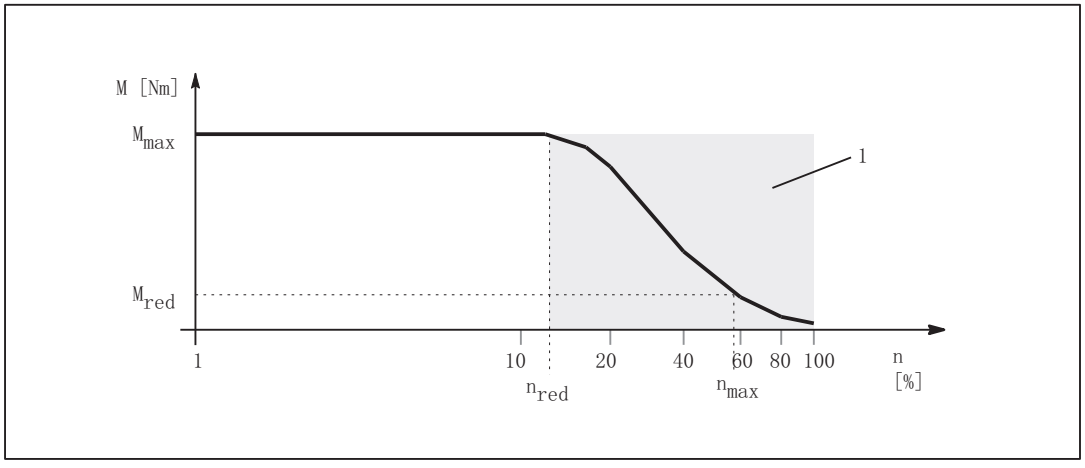
只有当时间常数大于位置控制周期时，急动度滤波器才会生效。

6.2.19 折线式加速度特性曲线

6.2.19.1 适应电机特性曲线

功能

各种电机类型，尤其是步进电机，转矩会紧跟着转速变化，在高速区内转矩会急剧下降。为充分利用这种电机特性曲线，需要从某个转速点起降低加速度。



1： 转矩下降区域

$n_{回落}$： 从降低的转矩起开始计算的转速

$n_{最大}$： 最大转速

$M_{最大}$： 最大转矩

$M_{回落}$： $n_{最大}$ 时的转矩（与转矩开始回落时的加速度一致）

图 6-6 转矩紧跟着转速变化的电机的转矩特性曲线

模拟转矩曲线

为了模拟电机转矩曲线，可通过以下机床数据：
MD35242 $MA_ACCEL_REDUCTION_TYPE = *曲线*
选择不同的特性曲线类型：

0	= 恒定曲线
1	= 抛物线
2	= 直线

下图分别展示了各个特性曲线类型的典型速度和加速度特性曲线：

恒定曲线

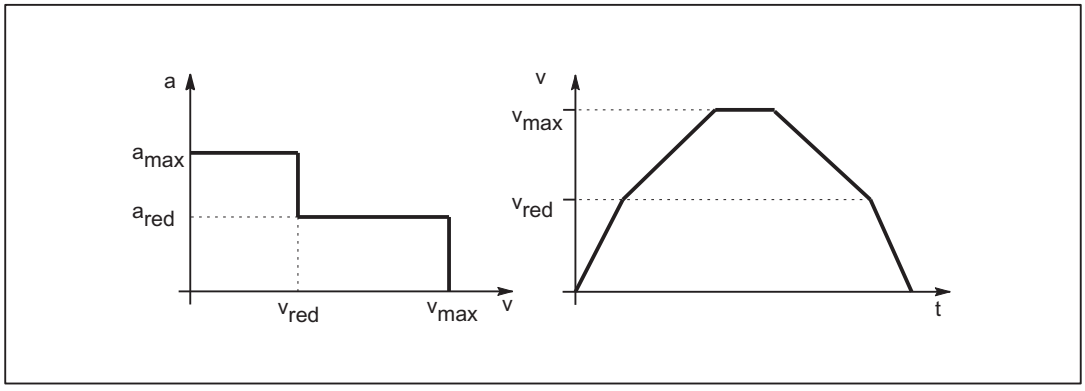


图 6-7　　加速度回落时的加速度和速度曲线：0 = 恒定曲线

抛物线

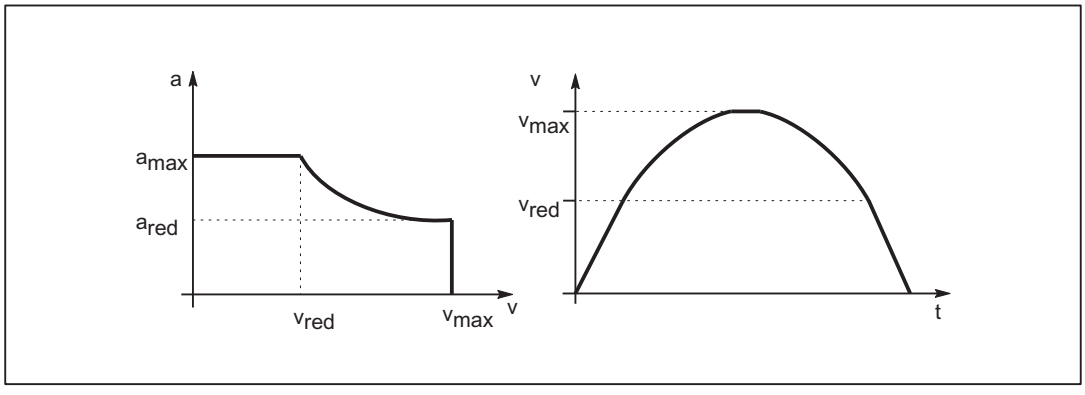


图 6-8　　加速度回落时的加速度和速度曲线：1 = 抛物线

直线

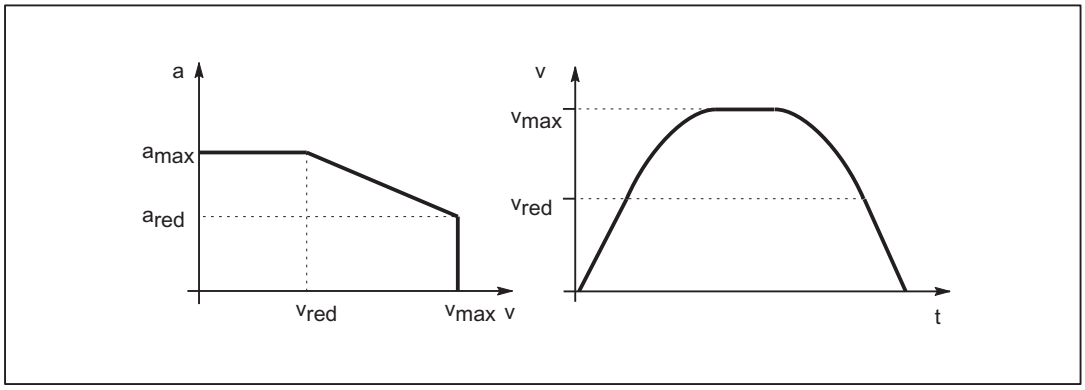


图 6-9　　加速度回落时的加速度和速度曲线：2 = 直线

特性曲线的参数是通过以下方式得出的：

$v_{最大}$　= $MA_MAX_AX_VELO

$v_{回落}$　= $MA_ACCEL_REDUCTION_SPEED_POINT * $MA_MAX_AX_VELO

$a_{最大}$　= $MA_MAX_AX_ACCEL

$a_{回落}$　= (1 - $MA_ACCEL_REDUCTION_FACTOR) * $MA_MAX_AX_ACCEL

6.2.19.2　对轨迹加速度的影响

功能

轨迹加速度的特性曲线是由参与到轨迹运动中的轴的特性曲线得出的。如果带有不同特性曲线的轴进行了共同插补，那么轨迹加速模式会由最具限制性的回落类型决定。

优先级的顺序如下，其中 1 表示最高优先级：

1. 加速度回落：0 = 恒定曲线
2. 加速度回落：1 = 抛物线
3. 加速度回落：2 = 直线
4. 无加速度回落生效
   无生效的加速度回落可由以下方式得出，如：
   MD35220 $MA_ACCEL_REDUCTION_SPEED_POINT = 1
   和/或
   MD35230 $MA_ACCEL_REDUCTION_FACTOR = 0

说明

带有步进电机和直流电机的机床轴可共同插补。

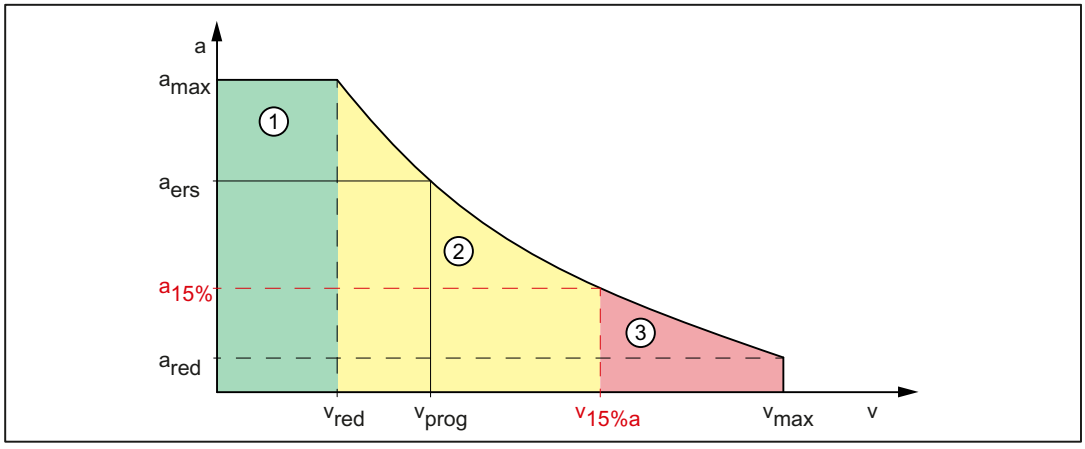
6.2.19.3 备用特性曲线

功能

若通过相关轴的机床数据中所设置的加速度特性曲线无法运行编写的轮廓，例如在运动转换激活时以及在加速度方式 BRISK 或 SOFT 下，系统会降低动态限值来生成备用特性曲线。 在确定降低的动态限值时，系统会使备用特性曲线在最大速度和恒定加速度之间达到最佳平衡。

直线轨迹段上的备用特性曲线

编程的轨迹速度会被限制在加速能力仍剩余 15 % 的某个速度下 ($v_{15\%a}$)。这样，在任何一个加工条件下都至少有 15% 的最大电机加速能力或电机转矩可用。



① 普通区域

② 降低区域

③ 禁用区域

$a_{备用}$ 备用特性曲线的恒定加速度

$a_{15\%}$ 最小恒定加速度

$a_{15\%} = 0.15 * (a_{最大} - a_{回落}) + a_{回落}$

$v_{备用}$ 备用特性曲线速度

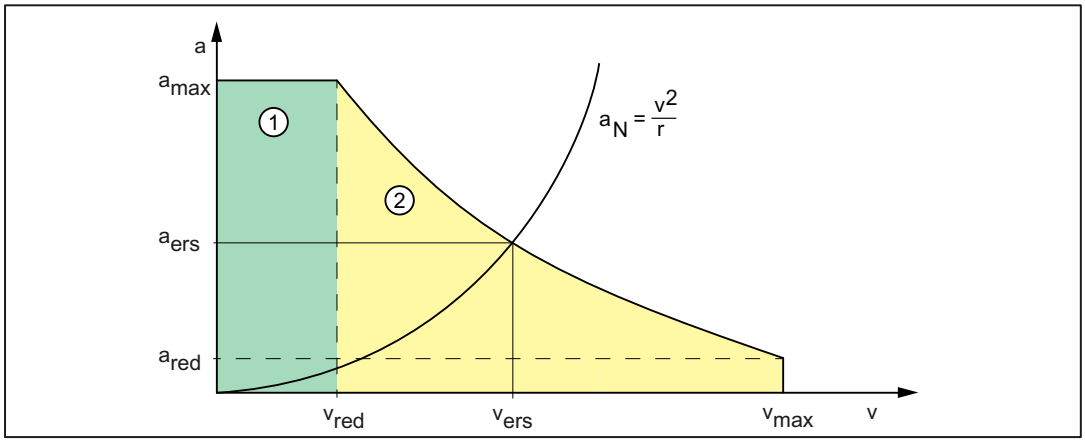
$v_{编程}$ 编程的速度

$v_{15\%a}$ $a_{15\%}$ 时的速度

图 6-10 轨迹备用特性曲线：直线轨迹

弯曲轨迹段上的备用特性曲线

在弯曲的轨迹段上，系统会同时考虑法线加速度和切向加速度。轨迹速度此时会降低至最多只为常规加速度留有 25 % 的加速能力。剩余 75 % 的加速能力会留给切向加速度，即预留给在轨迹上制动或加速。



① 普通区域

② 降低区域

a_N 常规加速度

$a_{备用}$ 备用特性曲线的恒定加速度

$v_{备用}$ 备用特性曲线速度

r 轨迹半径

图 6-11 轨迹备用特性曲线：弯曲轨迹

连续路径运行中的程序段过渡

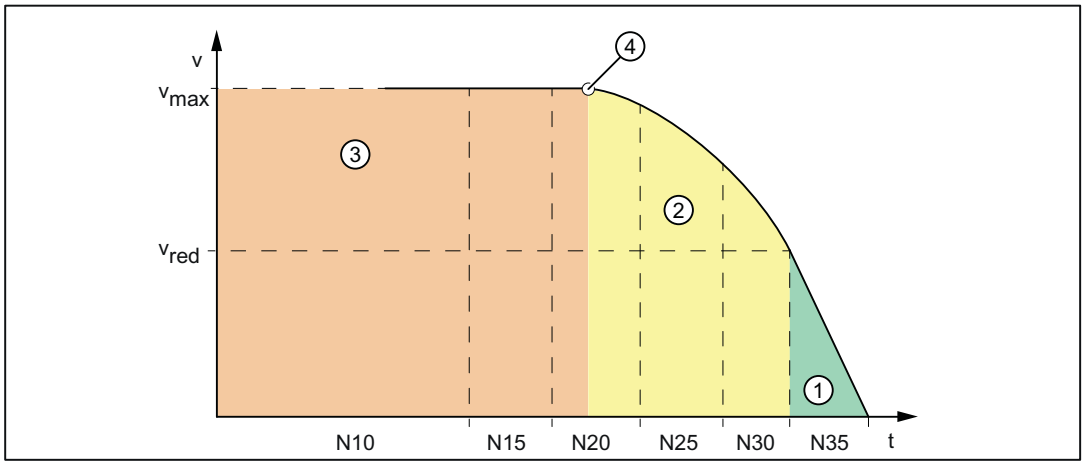
如果连续路径运行是激活的，在以设置的轨迹速度运行时在非相切的程序段过渡处可能会出现轴速度跃变。

之后，系统会对轨迹速度加以控制，以避免在程序段过渡处出现大于转矩开始回落时的速度 $v_{回落}$ 的分量。

连续路径运行和预读功能下的制动斜坡

在多个连续短行程程序段中，加速过程或制动过程可能要持续多个程序段。

此时“预读”功能也会考虑所设的随速度变化的加速度曲线。



① 普通区域 ⇒ $a = a_{max}$

② 降低区域 ⇒ $a < a_{max}$

③ 恒速运行区域 ⇒ $a = 0\ m/s^2$

④ 制动起点

$v_{回落}$ 转矩开始回落时的速度

$v_{最大}$ 最大速度

Nx 程序段编号为 Nx 的运行程序段

图 6-12 带预读功能的制动过程

6.2.19.4 参数设置

折线式加速度特性曲线是通过以下机床数据激活的：

MD35240 $MA_ACCEL_TYPE_DRIVE = TRUE

折线式加速度特性曲线是通过以下机床数据设置的：

MD32000 $MA_MAX_AX_VELO（最大轴速度）

MD35220 $MA_ACCEL_REDUCTION_SPEED_POINT
（加速度开始回落的起始转速）

MD35230 $MA_ACCEL_REDUCTION_FACTOR（回落的加速度）

MD32300 $MA_MAX_AX_ACCEL（最大轴加速度）

MD35242 $MA_ACCEL_REDUCTION_TYPE
（加速度回落方式：0=恒定曲线、1=抛物线、2=直线）

6.2.19.5 编程

通道专用的激活 (DRIVE)

句法

```
DRIVE
```

功能

通过零件程序指令 DRIVE 可为轨迹加速度激活折线式加速度特性曲线

G 功能组 21

有效性：模态有效

复位特性

进行复位操作后，以下通道专用的初始设置生效：

MD20150 $MC_GCODE_RESET_VALUES[20]

关联性

如果为机床轴设置了折线式加速度特性曲线，那么缺省条件下该轴会以该加速模式运行。

如果通过零件程序指令 SOFT 或 BRISK 切换了某个轨迹段的加速模式，那么系统会采用带有回落的动态限值的备用特性曲线来代替折线式加速度特性曲线。

重新编程 DRIVE 后可再次激活折线式加速度特性曲线。

单轴激活 (DRIVEA)

句法

DRIVEA (*轴* {*轴*})

功能

通过零件程序指令可为所有的单轴插补（定位轴、摆动轴等）激活折线式加速度特性曲线。

G 功能组-

有效性：模态

轴:

- 取值范围：通道轴名称

复位特性

进行复位操作后，以下通道专用的初始设置生效：

MD20150 $MC_GCODE_RESET_VALUES[20]

关联性

如果为机床轴设置了折线式加速度特性曲线，那么缺省条件下该轴会以该加速模式运行。

如果通过零件程序指令 SOFTA 或 BRISKA 切换了某个轨迹段的加速模式，那么系统会采用带有回落的动态限值的备用特性曲线来代替折线式加速度特性曲线。

通过编程 DRIVEA 可再次切换回折线式加速度特性曲线。

6.2.19.6 前提条件

单轴插补

在单轴插补（定位轴、摆动轴、手动运行等）中激活折线式加速度特性曲线后，就只能在 DRIVEA 模式下运行机床轴。

无法通过以下零件程序指令切换加速模式：

- 刚性加速模式 (BRISKA)
- 软加速模式 (SOFTA)

轨迹插补

如果在零件程序指令 DRIVE 未生效的情况下为参与到编程的轨迹的机床轴设置了折线式加速度特性曲线，系统将会为该轨迹确定一个带有回落的动态限值的备用特性曲线。

运动转换

运动转换生效时，系统不会考虑折线式加速度特性曲线。控制系统内部会切换至刚性加速模式 (BRISK)，同时备用特性曲线对轨迹加速度生效。

6.2.20 JOG 运动中的加速度和急动度

可为 JOG 运动预设单独的最大轴加速度和最大轴急动度以避免 JOG 模式中出现机床急动度运动故障。

另外，可以在运行几何轴和定向轴时对加速度和急动度进行限制，以提升坐标转换功能的易操作性，并可完全由回转轴产生坐标运动（机器人）。

6.2.20.1 参数设置

最大轴加速度和轴急动度

JOG 最大轴加速度

通过以下机床数据可为每根机床轴预设 JOG 运动中的最大轴加速度：

MD32301 $MA_JOG_MAX_ACCEL

当 MD32301 = 0 时，MD32300 $MA_MAX_AX_ACCEL 中的最大轴加速度代替 JOG 专用的最大轴加速度生效。

JOG 运动中的最大轴专用急动度

通过以下机床数据可为每根机床轴预设 JOG 运动中的最大轴急动度：

MD32436 $MA_JOG_MAX_JERK

当 MD32436 = 0 时，MD32430 $MA_JOG_AND_POS_MAX_JERK 中的最大轴急动度代替 JOG 专用的最大轴急动度生效。

说明

只有在为 JOG 模式中的待运行机床轴激活了最大轴急动度时，MD32436 $MA_JOG_MAX_JERK 才会生效：

MD32420 $MA_JOG_AND_POS_JERK_ENABLE [<轴>] == TRUE

这也可以通过在零件程序中编程 `SOFTA(<Achse1>,<Achse2>,...)` 实现。

通道专用的最大加速度和急动度

几何轴手动运行时的加速度上限

手动运行几何轴时，可通过以下机床数据来预设最大加速度：

MD21166 $MC_JOG_ACCEL_GEO [<几何轴>]

其中 <几何轴> = 0, 1, 2

当 MD21166 = 0 时，MD32301 $MA_JOG_MAX_ACCEL 中的轴专用限值会代替通道专用的最大加速度生效。

说明

MD21166 $MC_JOG_ACCEL_GEO [<几何轴>] 对 MD32300 $MA_MAX_AX_ACCEL 没有直接的限制。

说明

当转换激活时，MD32300 $MA_MAX_AX_ACCEL 决定了最大轴加速度。

几何轴手动运行时的急动度上限

在加速模式 SOFT（软加速模式）中手动运行几何轴时，可通过以下机床数据设置最大加速度：

MD21168 $MC_JOG_JERK_GEO [<几何轴>]

其中 <几何轴> = 0, 1, 2

当 MD21168 = 0 时，MD32436 $MA_JOG_MAX_JERK 中的轴专用限值会代替通道专用的最大急动度生效。

说明

只有在为 JOG 模式中的机床轴激活了最大轴急动度时，MD21168 $MC_JOG_JERK_GEO 才会生效：
MD32420 $MA_JOG_AND_POS_JERK_ENABLE [<轴>] == TRUE

定向轴手动运行时的急动度上限

手动运行定向轴时，可通过以下机床数据设置最大急动度：

MD21158 $MC_JOG_JERK_ORI [<定向轴>]

为了使 MD21158 生效，必须通过以下机床数据为手动运行定向轴激活通道专用的最大急动度：

MD21159 $MC_JOG_JERK_ORI_ENABLE == TRUE

说明

定向轴与机床数据 MD32301 $MA_JOG_MAX_ACCEL 和 MD32436 $MA_JOG_MAX_JERK 无关。

6.2.20.2 前提条件

行程叠加/叠加运动

在行程叠加/叠加运动中（如 DRF），JOG 专用的最大加速度和急动度 (MD32301 $MA_JOG_MAX_ACCEL 和 MD32436 $MA_JOG_MAX_JERK)不生效，取而代之的是用于定位轴运动的最大加速度和急动度：

- 加速度：
  - MD32300 $MA_MAX_AX_ACCEL [0]（动态模式 DYNNORM 中轨迹运动中的最大单轴加速度）
    另见章节“刚性加速模式 (BRISK/BRISKA)（通道/轴专用） (页 287)”。
- 急动度
  - MD32430 $MA_JOG_AND_POS_MAX_JERK（定位轴运动中的最大轴急动度）或者 (G75)：
  - MD32431 $MA_MAX_AX_JERK [0]（动态模式 DYNNORM 下的轨迹运动中的最大轴急动度）

  另见章节“单轴插补中的最大急动度 (SOFTA)（轴专用） (页 305)”。

说明

对于 JOG 模式，始终只有动态模式 DYNNORM 有效。

旋转指令生效时手动运行几何轴的特性

在加速模式 SOFT（软加速模式）中手动运行几何轴时，激活的旋转或激活的可定向刀架中也会使用 MD32436 $MA_JOG_MAX_JERK 或 MD32430 $MA_JOG_AND_POS_MAX_JERK 中的值。

零件程序指令 SOFTA / BRISKA / DRIVEA

零件程序指令 SOFTA(<轴 1>,<轴 2>,...) 也可在 JOG 模式中生效，即在 JOG 模式下运行时，MD32436 $MA_JOG_MAX_JERK 中的最大轴急动度对指定轴有效（和设置 MD32420 $MA_JOG_AND_POS_JERK_ENABLE [<轴>] == TRUE 时一样）。

说明

零件程序指令 SOFT 对 JOG 模式无效。

和 SOFTA 一样，零件程序指令 BRISKA 和 DRIVEA 在 JOG 模式中也有效，也就是说，即使将 MD32420 $MA_JOG_AND_POS_JERK_ENABLE 设为了 “TRUE”，相关机床轴的刚性加速模式依然有效。

说明

手动运行定向轴不受 BRISKA/SOFTA/DRIVEA 的影响。

6.3 示例

6.3.1 加速度

6.3.1.1 轨迹速度曲线

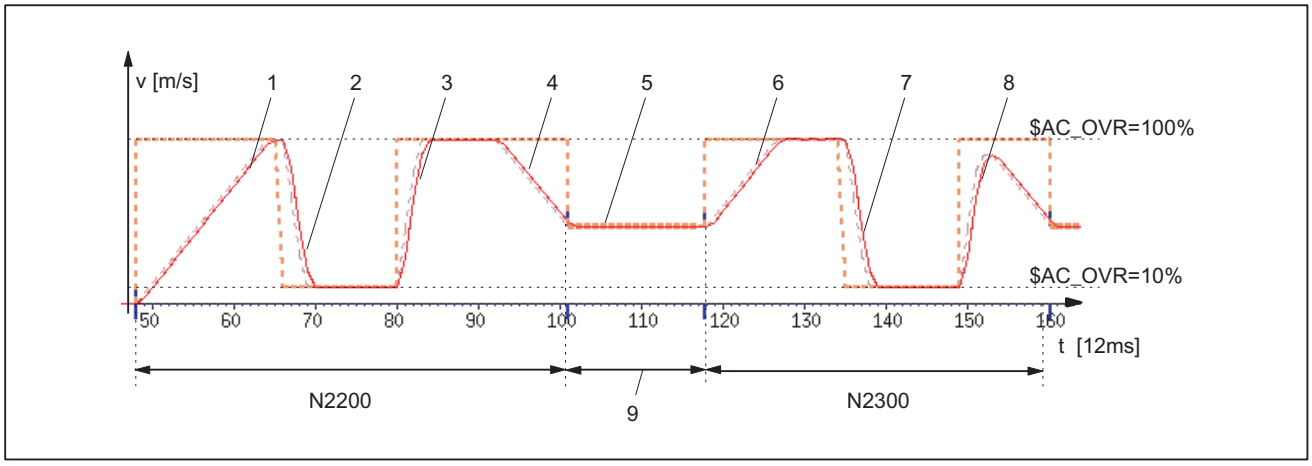
在下面的示例中将借助编程设置的零件程序段的运动和动作来展示轨迹速度的变化曲线。

零件程序段

程序代码

```
; 同步动作： 取决于快速运行 1 ($A_IN[1]) 的加速度切换：
N53 ID=1 WHEN $A_IN[1] == 1 DO $AC_PATHACC = 2.*$MA_MAX_AX_ACCEL[X]
; 同步动作： 测试倍率曲线（模拟外部作用）：
N54 ID=2 WHENEVER ($AC_TIMEC > 16) DO $AC_OVR=10
N55 ID=3 WHENEVER ($AC_TIMEC > 30) DO $AC_OVR=100
; 逼近
N1000 G0 X0 Y0 BRISK
N1100 TRANS Y=-50
N1200 AROT Z=30 G642
; 轮廓
N2100 X0 Y0
N2200 X = 70 G1 F10000 RNDM=10 ACC[X]=30 ACC[Y]=30
N2300 Y = 70
N2400 X0
N2500 Y0
```

轨迹速度的变化曲线



加速曲线： BRISK

1: 按照加速度设定加速到轨迹速度 (F10000) 的 100%：ACC (N2200...)

2: 在倍率变化 ($AC_OVR) 的基础上按照实时加速度 $AC_PATHACC (N53/N54...) 制动至轨迹速度的 10%。

3: 在倍率变化 ($AC_OVR) 的基础上按照实时加速度 $AC_PATHACC (N53/N55...) 加速至轨迹速度的 100%。

4: 按照加速度设定：ACC (N2200...) 制动至倒圆中间程序段结束时的速度

5: 基于倒圆的最大速度（见 9）

6: 按照加速度设定加速到轨迹速度 ($AC_OVR) 的 100%：ACC (N2300...)

7: 在倍率变化的基础上按照实时加速度 $AC_PATHACC (N53/N54...) 制动

8: 在倍率变化 ($AC_OVR) 的基础上按照实时加速度 $AC_PATHACC (N53/N55...) 加速至轨迹速度的 100%。

9: 在所编程基圆 (RNDM) (N2200...) 的基础上插入到控制系统内部的中间程序段

图 6-13 在在预处理中确定的轨迹加速度和实时加速度之间进行切换

6.3.2 急动度

6.3.2.1 轨迹速度曲线

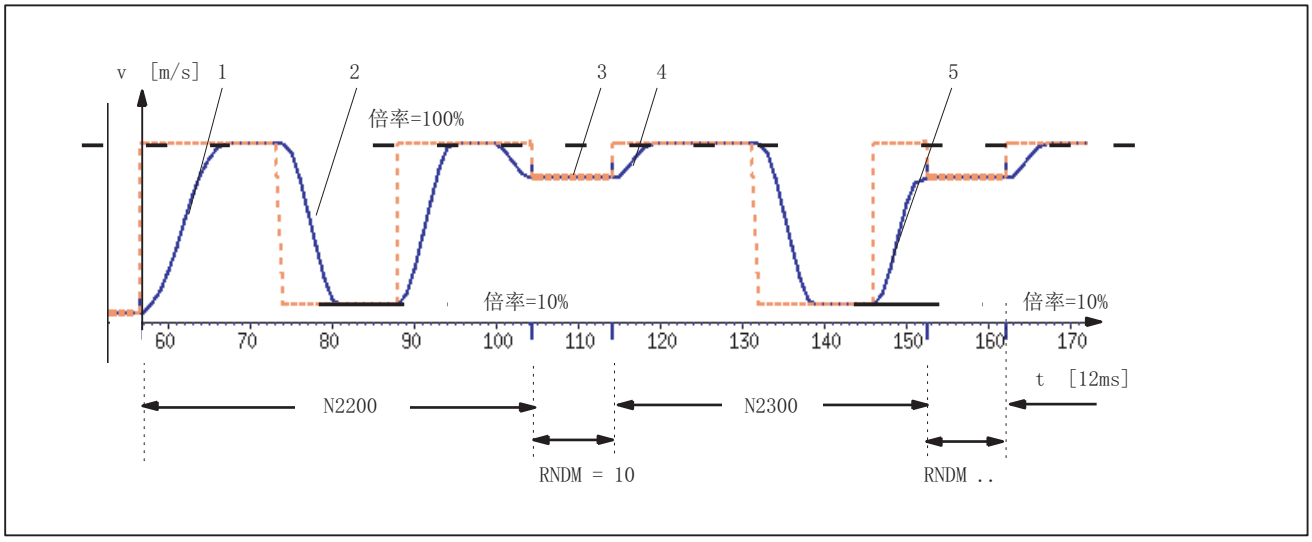
在下面的示例中将借助编程设置的零件程序段的运动和动作来展示轨迹速度和急动度的变化曲线。

零件程序段

程序代码

```
；在有外部作用时设置轨迹加速度和轨迹急动度：
N0100 $AC_PATHACC = 0.0
N0200 $AC_PATHJERK = 4.0 * ($MA_MAX_AX_JERK[X] + $MA_MAX_AX_JERK[Y]) / 2.0
；同步动作：  倍率变化（模拟外部作用）
N0530 ID=1 WHENEVER ($AC_TIMEC > 16) DO $AC_OVR=10
N0540 ID=2 WHENEVER ($AC_TIMEC > 30) DO $AC_OVR=100
；逼近
N1000 G0 X0 Y0 SOFT
N1100 TRANS Y=-50
N1200 AROT Z=30 G642
；轮廓
N2100 X0 Y0
N2200 X = 70 G1 F10000 RNDM=10
N2300 Y = 70
N2400 X0
N2500 Y0
```

轨迹速度和急动度的变化曲线



加速曲线： SOFT

1: 急动度，符合 $MA_MAX_AX_JERK[..]

2: 急动度，符合 $AC_PATHJERK

3: 急动度，符合 $MA_MAX_AX_JERK[..]（逼近程序段结束时的速度）

4: 因圆弧产生的速度限制

5: 急动度，符合 $AC_PATHJERK

图 6-14 在在预处理中确定的轨迹急动度和 $AC_PATHJERK 之间进行切换

6.3.3 加速度和急动度

以下示例中借助一个零件程序段展示了 X 轴的速度和加速度曲线以及哪些与速度和加速度相关的机床数据决定了哪些轮廓段。

零件程序段

程序代码	注释
N90 F5000 SOFT G64	；连续路径运行，软加速模式
N100 G0 X0 Y0 Z0	；快速移动
N110 G1 X10	；直线
N120 G3 CR=5 X15 Y5	；圆弧，半径 5 mm，程序段过渡：切向
N130 G3 CR=10 X5 Y15	；圆弧，半径 10 mm，程序段过渡：切向
N140 G1 X-5 Y17.679	；直线，15°弯曲

轮廓

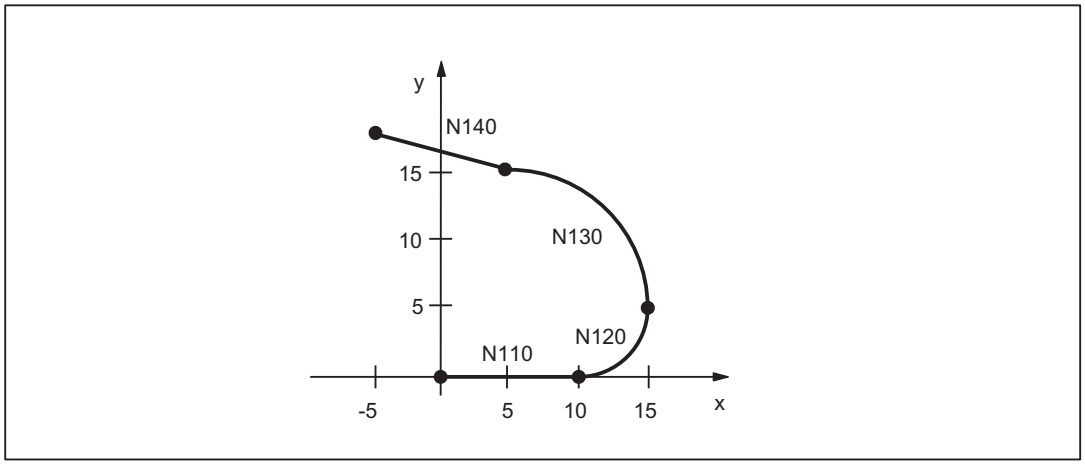


图 6-15　　零件程序段的轮廓

速度和加速度变化曲线

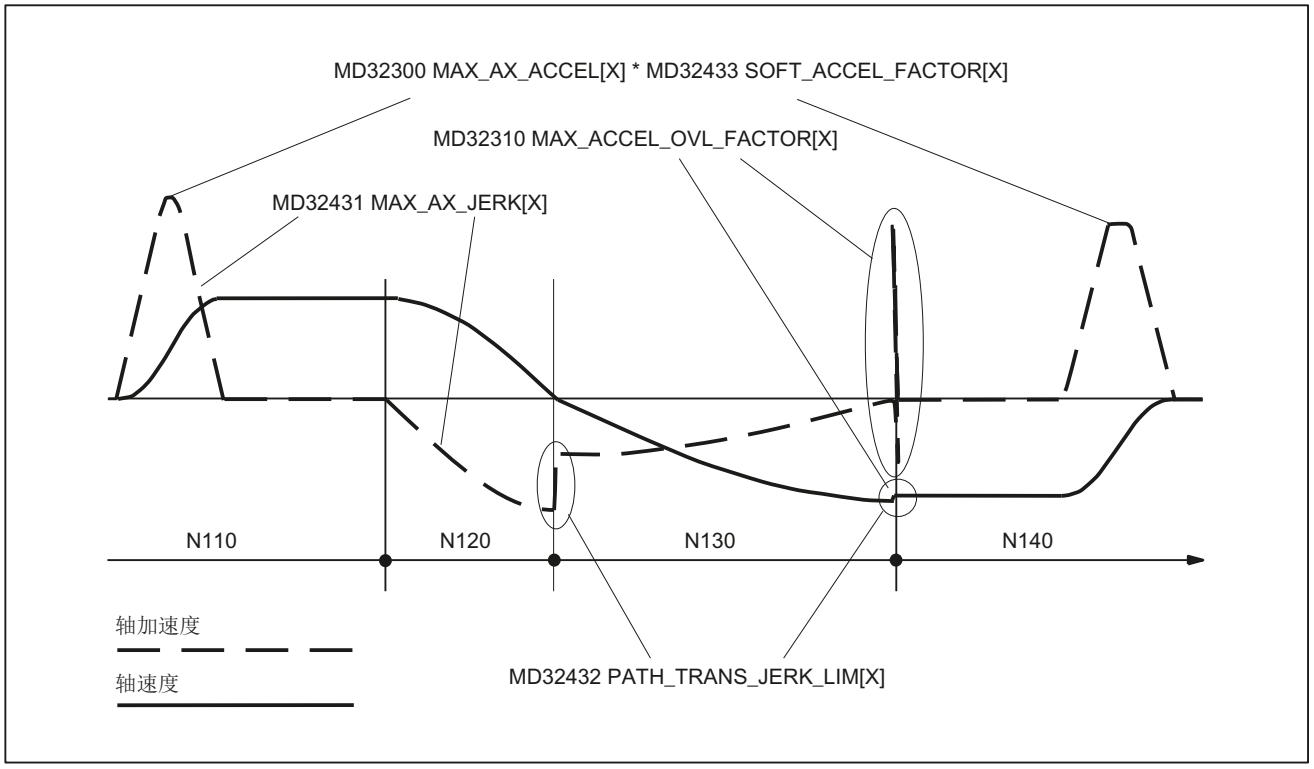


图 6-16　　速度和加速度变化曲线： X 轴

6.3.4 折线式加速度特性曲线

6.3.4.1 激活

该示例展示了如何借助机床数据和零件程序段激活折线式加速度特性曲线。

机床数据

机床数据	值
MD35220 $MA_ACCEL_REDUCTION_SPEED_POINT[**X**]	= 0.4
MD35230 $MA_ACCEL_REDUCTION_FACTOR[**X**]	= 0.85
MD35242 $MA_ACCEL_REDUCTION_TYPE[**X**]	= 2
MD35240 $MA_ACCEL_TYPE_DRIVE[**X**]	= TRUE
MD35220 $MA_ACCEL_REDUCTION_SPEED_POINT[**Y**]	= 0.0
MD35230 $MA_ACCEL_REDUCTION_FACTOR[**Y**]	= 0.6
MD35242 $MA_ACCEL_REDUCTION_TYPE[**Y**]	= 1
MD35240 $MA_ACCEL_TYPE_DRIVE[**Y**]	= TRUE
MD35220 $MA_ACCEL_REDUCTION_SPEED_POINT[**Z**]	= 0.6
MD35230 $MA_ACCEL_REDUCTION_FACTOR[**Z**]	= 0.4
MD35242 $MA_ACCEL_REDUCTION_TYPE[**Z**]	= 0
MD35240 $MA_ACCEL_TYPE_DRIVE[**Z**]	= FALSE

通过设定为通道专用的初始设置激活：

MC_GCODE_RESET_VALUE[20] = 3 (DRIVE)

零件程序段

程序代码	注释
N10 G1 X100 Y50 Z50 F700	; 带 DRIVE 的轨迹运动 (X,Y,Z)
N15 Z20	; 带 DRIVE 的轨迹运动 (Z)
N20 BRISK	; 切换至 BRISK
N25 G1 X120 Y70	; 带备用特性曲线的轨迹运动 (Y,Z)
N30 Z100	; 带 BRISK 的轨迹运动 (Z)
N35 POS[X] = 200 FA[X] = 500	; 带 DRIVEA 的定位运动 (X)
N40 BRISKA(Z)	; 为 Z 激活 BRISKA
N40 POS[Z] = 50 FA[Z] = 200	; 带 BRISKA 的定位运动 (Z)
N45 DRIVEA(Z)	; 为 Z 激活 DRIVEA
N50 POS[Z] = 100	; 带 DRIVE 的定位运动 (Z)

程序代码	注释
N55 BRISKA(X)	；导致故障报警

6.4 数据表

6.4.1 机床数据

6.4.1.1 NC 专用机床数据

编号	名称：$MN_	说明
18960	POS_DYN_MODE	定位轴动态响应方式

6.4.1.2 通道专用机床数据

编号	名称：$MC_	说明
20150	GCODE_RESET_VALUES	G 功能组的初始设置
20500	CONST_VELO_MIN_TIME	恒定速度的最短持续时间
20600	MAX_PATH_JERK	最大轨迹急动度
20602	CURV_EFFECT_ON_PATH_ACCEL	轨迹曲率对轨迹动态响应的影响
20610	ADD_MOVE_ACCEL_RESERVE	为叠加运动保留的加速度余量
21158	JOG_JERK_ORI	定位轴在 JOG 模式下运行时的最大急动度
21159	JOG_JERK_ORI_ENABLE	定位轴在 JOG 模式下运行时的通道专用最大急动度的初始设置
21166	JOG_ACCEL_GEO	几何轴在 JOG 模式下运行时的最大加速度
21168	JOG_JERK_GEO	几何轴在 JOG 模式下运行时的最大急动度

6.4.1.3 进给轴/主轴专用机床数据

编号	名称：$MA_	说明
32000	MAX_AX_VELO	最大轴速度
32300	MAX_AX_ACCEL	最大轴加速度
32301	JOG_MAX_ACCEL	JOG 模式中的最大加速度
32310	MAX_ACCEL_OVL_FACTOR	速度跃变的过载系数
32320	DYN_LIMIT_RESET_MASK	动态响应限制的复位特性
32400	AX_JERK_ENABLE	最大轴急动度
32402	AX_JERK_MODE	用于最大轴急动度的滤波器类型
32410	AX_JERK_TIME	用于轴急动度滤波器的时间常数
32420	JOG_AND_POS_JERK_ENABLE	最大轴急动度的初始设置
32430	JOG_AND_POS_MAX_JERK	单轴运动中的轴急动度
32431	MAX_AX_JERK	连续路径运行中程序段过渡处的最大轴急动度
32432	PATH_TRANS_JERK_LIM	位于程序段边界处的几何轴的最大急动度
32433	SOFT_ACCEL_FACTOR	SOFT 最大加速度比例
32434	G00_ACCEL_FACTOR	G00 最大加速度比例
32435	G00_JERK_FACTOR	G00 最大轴急动度比例
32436	JOG_MAX_JERK	JOG 运动中的最大轴急动度
32437	AX_JERK_VEL0	用于随速度提高急动度的第一个速度阈值
32438	AX_JERK_VEL1	用于随速度提高急动度的第二个速度阈值
32439	MAX_AX_JERK_FACTOR	用于高速下设置最大急动度的系数（随速度提高急动度）
35220	ACCEL_REDUCTION_SPEED_POINT	加速度开始回落的起始转速
35230	ACCEL_REDUCTION_FACTOR	回落的加速度
35240	ACCEL_TYPE_DRIVE	加速特性曲线 DRIVE 开/关
35242	ACCEL_REDUCTION_TYPE	加速度回落方式

6.4.2 设定数据

6.4.2.1 通道专用设定数据

编号	名称：$SC_	说明
42500	SD_MAX_PATH_ACCEL	最大轨迹加速度
42502	IS_SD_MAX_PATH_ACCEL	SD 42500 的分析：打开/关闭
42510	SD_MAX_PATH_JERK	与轨迹相关的最大急动度
42512	IS_SD_MAX_PATH_JERK	SD 42510 的分析：打开/关闭

6.4.3 系统变量

名称	说明
$AC_PATHACC	针对实时事件的轨迹加速度
$AC_PATHJERK	针对实时事件的轨迹急动度

F1:运行到固定挡块

7

7.1 简要说明

功能

借助“运行到固定挡块”功能，可运行某些运动机床部件（如尾架或套筒），从而通过这些部件在任意时间范围内对其他机床部件施加可定义的转矩或力。

特性

- 可通过零件程序或同步动作中的指令编程：
  - 激活和取消“运行到固定挡块”功能
  - 设置夹紧转矩
  - 设置监控窗口
- 通过设定数据手动调整：
  - 激活和取消“运行到固定挡块”功能
  - 设置夹紧转矩
  - 设置监控窗口
- 通过机床数据预设
  - 夹紧转矩
  - 监控窗口
- 通过 NC/PLC 接口信号识别功能状态
- 通过 NC/PLC 接口信号使能或应答
- 通过系统变量读取功能的设定状态和实际状态
- 可通过进给轴和主轴执行“运行到固定挡块”。
- 可同时为多个轴执行“运行到固定挡块”功能，并且可以同时运行其他进给轴。
- 可在多通道中对所需的所有附加数据进行带计算的程序段搜索（SERUPRO）。
- 可配合“运行到固定挡块”和“转矩降低”执行轴的仿真运行。
- “垂直”轴可在 FXS 报警时运行到固定挡块。

7.2 详细说明

7.2.1 编程

功能

运行到固定挡块

通过 FXS、FXST 和 FXSW 指令可对“运行到固定挡块”功能进行控制。

在不运行相关轴的情况下同样可以激活该功能。 此时系统会立即对转矩进行限制。 一旦轴运行，系统便会对该功能进行监控。

说明

同步动作

“运行到固定挡块”功能也可以由同步动作控制。

文档：
功能手册之同步动作分册

以受限转矩/力运行

通过 FOCON、FOCOF 和 FOC 指令可对以受限转矩/力运行进行控制（参见章节“以受限转矩/力 FOC 运行 (页 356)”）。

句法

```
FXS[<轴>]=<请求>
FXST[<轴>]=<夹紧转矩>
FXSW[<轴>]=<窗口宽度>
```

含义

参数	含义
FXS：	“运行至固定挡块”功能，生效方式： 模态有效
<请求>：	0 = 取消 1 = 激活
FXST：	设置夹紧转矩
<夹紧转矩>：	夹紧转矩以驱动最大转矩的 % 形式设定。 SINAMICS S120：p2003

参数	含义
FXSW:	设置监控窗口
<窗口宽度>:	以固定挡块为中心的监控窗口宽度 单位： 毫米，英寸或度
<轴>:	通道轴的名称，类型： AXIS

修改转矩限制（FXST）

可在每个程序段中修改转矩限值。 修改会在执行程序段中编写的运动前生效。 转矩限制用作对加速度限制（ACC）的补充。

斜坡式变化

通过以下机床数据可设置一个时间，在此时间内线性修改转矩限值：

MD37012 $MA_FIXED_STOP_TORQUE_RAMP_TIME（达到新转矩限值所经过的时间）

修改监控窗口（FXSW）

可在每个程序段中修改监控窗口。 修改会在执行程序段中编写的运动前生效。

修改监控窗口时，不仅窗口宽度会改变，窗口相对轴当前实际位置的参考点也会改变。

连续路径运行（G64）激活时“运行到固定挡块”

通过以下机床数据可以设置：在连续路径运行（G64）生效期间取消功能（FXS）时，不会触发引起程序段切换（G60）的准停。

MD37060 $MA_FIXED_STOP_ACKN_MASK PLC（是否等待 PLC 响应来执行“运行到固定挡块”功能）

位	值	含义
0	0	无需 PLC 响应即可启动运动
	1	需要 PLC 响应即可启动运动

前提条件

- 该运行可作为单个轨迹、单个程序段或跨程序段的定位轴运行进行编程。
- 多个机床轴可同时运行到固定挡块。
- 通过机床轴运行至固定挡块时，该轴上不得激活转换、耦合及框架功能。

- 在零件程序中，运行距离和该功能的激活指令必须在一个程序段中编程。
- 如果通过同步动作激活了“运行到固定挡块”功能，则也可以在单独的程序段中编程运行距离和该功能的激活指令。

7.2.2 工作流程

7.2.2.1 选择

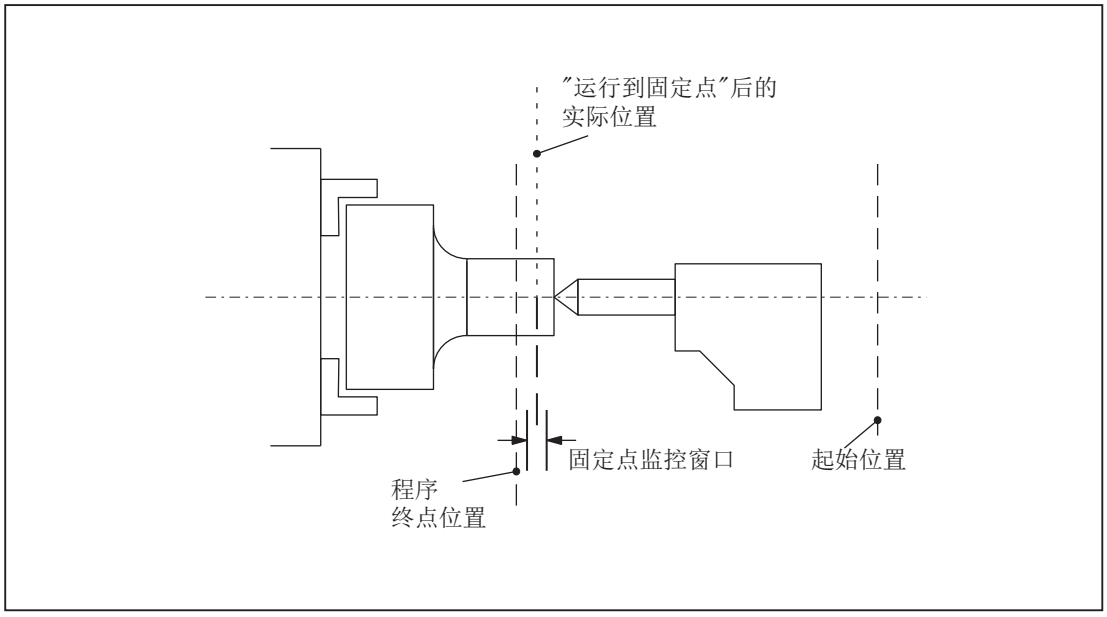


图 7-1 运行到固定挡块示例

过程

NC 检测到已通过指令 FXS[x]=1 选择了“运行到固定挡块”功能并通过接口信号 DB31, ... DBX62.4 “激活运行到固定挡块”通知 PLC，表示此功能已选择。

如果设定了相应的机床数据：

MD37060 $MA_FIXED_STOP_ACKN_MASK PLC（是否等待 PLC 响应来执行“运行到固定挡块”功能）

系统会等待 PLC 通过接口信号 NST DB31, ... DBX3.1 发出应答，来执行“运行到固定挡块”功能。

获得响应后，轴会以编程的速度从起始位置运行至编程的目标位置。固定挡块必须位于进给轴/主轴的起始位置和目标位置之间。 编程的最大转矩（通过 FXST[<轴>] 设置的夹紧转

矩）自程序段开始时便生效，即：运动到固定挡块时也是采用下调后的转矩。转矩下调是通过 NC 中的自动加速度回落来实现。

如果在程序段中或是从程序开始时没有编程任何转矩，系统将会采用在轴专用机床数据：

MD37010 $MA_FIXED_STOP_TORQUE_DEF（固定挡块夹紧转矩的缺省设置）

中输入的值。

7.2.2.2 到达固定挡块

识别固定挡块

可通过以下机床数据设置对固定挡块的识别，或对“机床轴到达固定挡块”状态的识别：

MD37040 $MA_FIXED_STOP_BY_SENSOR = <值>（通过传感器识别是否到达固定挡块）

<值>	含义
0	机床轴的轮廓偏差超出机床数据中设置的值时，系统会认定到达固定挡块： MD37030 $MA_FIXED_STOP_THRESHOLD（用于识别是否到达固定挡块的阈值）
1	到达固定挡块是由一个外部传感器来识别的并通过以下轴 NC/PLC 接口信号进行通知： DB31, ... DBX1.2 == 1（固定挡块传感器）
2	当满足 <值> == 0 或 <值> == 1 下的条件时，系统会认定到达固定挡块。

失效的 NC/PLC 接口信号

如果轴处于“到达固定挡块”状态，那么以下 NC/PLC 接口信号失效：

- DB31, ... DBX1.3（进给轴/主轴禁止）
- DB31, ... DBX2.1（伺服使能）

到达固定挡块时的动作

到达固定挡块时，系统会执行以下动作：

- 将驱动中的转矩提升至编写的夹紧转矩（FXST）
- 删除剩余行程
- 跟踪位置设定值

- 置位 NC/PLC 接口信号：DB31, ... DBX62.5 = 1（“到达固定挡块”）
- 执行程序段切换：
  - 到达固定挡块时执行程序段切换：MD37060 $MA_FIXED_STOP_ACKN_MASK，位 1 = 0
    到达固定挡块后立即执行程序段切换。
  - 通过 PLC 用户程序应答后再执行程序段切换：MD37060 $MA_FIXED_STOP_ACKN_MASK，位 1 = 1
    到达固定挡块后进行等待，通过 PLC 用户程序应答后再执行程序段切换：DB31, ... DBX1.1 == 1（对到达固定挡块进行应答）

  为了在程序段切换后继续保持夹紧转矩，NC 会继续为机床轴输出设定值。
- 激活固定挡块监控或监控窗口

监控窗口

若在运行至固定挡块程序段中或自程序通过 FXSW 开始起未专门为监控窗口编写数值，那么将启用以下机床数据中设置的值：

MD37020 $MA_FIXED_STOP_WINDOW_DEF（固定挡块监控窗口的默认设置）

如果进给轴离开了识别出固定挡块时的位置并超出了设定窗口，那么会触发报警 20093 “固定挡块监控已响应”并取消“运行到固定挡块”功能。

选择监控窗口时，用户须确保只有在挡块取消时才会触发报警。

注意
"到达固定挡块"和挡块取消 一旦识别出“到达固定挡块”状态，系统便会在驱动上固定设定通过轴专用机床数据 K_V 系数（MD32200）和固定挡块识别阈值（MD37030）推导出的转速设定值。 在此状态下取消挡块时，轴会加速直至达到监控窗口限值。 此时达到的速度与上述机床数据中设置的值成比例。 在值较大时，轴可能会加速至驱动的最大电机转速。

概述

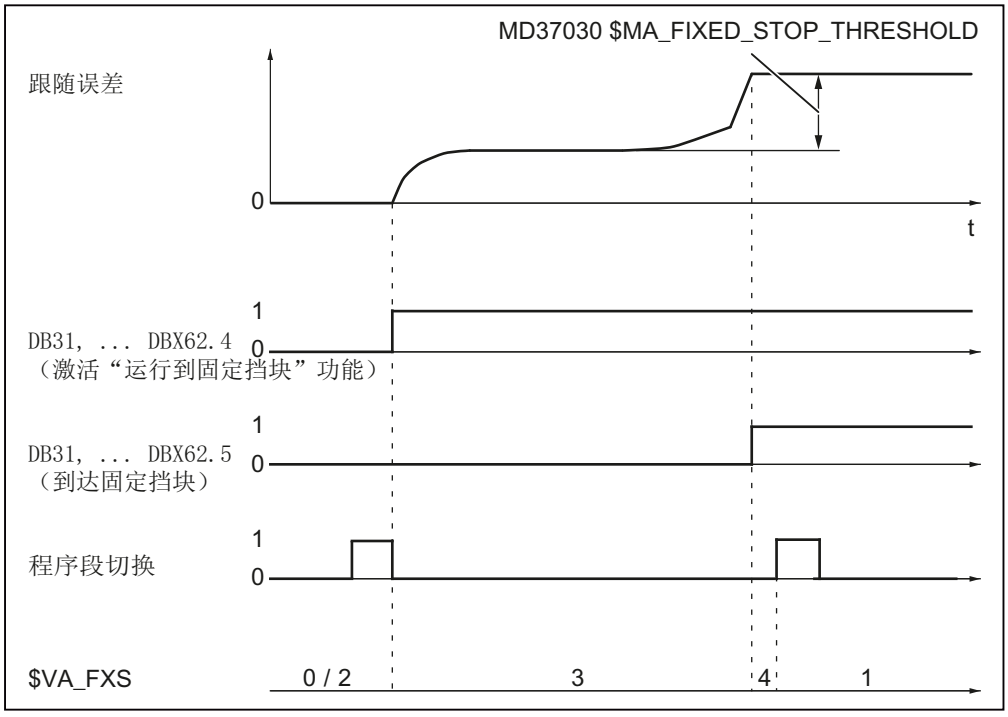


图 7-2　　到达固定挡块

7.2.2.3　未到达固定挡块

报警抑制

可通过以下机床数据为各种终止原因抑制报警：

MD37050 $MA_FIXED_STOP_ALARM_MASK = <值>

值	说明：被抑制的报警
0	报警 20091“未到达固定挡块”
2	报警 20091“未到达固定挡块” 报警 20094“固定挡块终止”
3	报警 20094“固定挡块终止”

故障或终止时的动作

出现故障或终止时，系统会执行以下动作：

- 复位 NC/PLC 接口信号： DB31, ... DBX62.4 = 0（激活“运行到固定挡块”功能）
- 根据以下机床数据中的设置等待应答：
  - MD37060 $MA_FIXED_STOP_ACKN_MASK
  - DB31, ... DBX3.1 == 0（激活“运行到固定挡块”功能）
- 取消转矩限制
- 执行程序段切换

概述

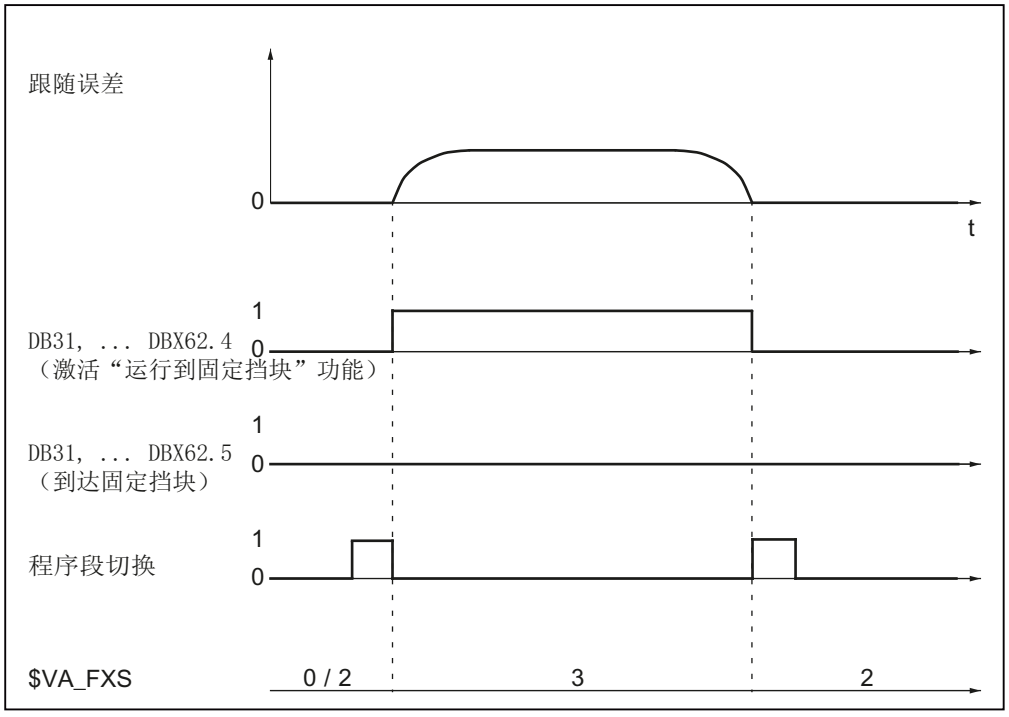


图 7-3　　未到达固定挡块

7.2.2.4　取消

“运行到固定挡块”功能通过在 NC 程序的程序段中写入 FXS[<轴>] = 0 指令来取消。

取消功能时的动作

取消此功能时，系统会执行以下动作：

- 触发预处理停止（STOPRE）
- 复位 NC/PLC 接口信号
  - DB31, ... DBX62.4 = 0（激活“运行到固定挡块”功能）
  - DB31, ... DBX62.5 = 0（到达固定挡块）
- 根据以下机床数据等待 PLC 用户程序应答：
  - MD37060 $MA_FIXED_STOP_ACKN_MASK
  - DB31, ... DBX3.1 == 0（激活“运行到固定挡块”功能）
  - DB31, ... DBX1.1 == 0（对到达固定挡块发出响应）
- 结束跟踪运行
- 将轴重新接收至位置闭环控制

脉冲使能

可通过以下方式撤销脉冲使能或脉冲禁用：

- 驱动：通过端子 EP（使能脉冲）
- NC/PLC 接口信号：DB31, ... DBX21.7（“脉冲使能”）

通过以下机床数据可以设置固定挡块处的特性：

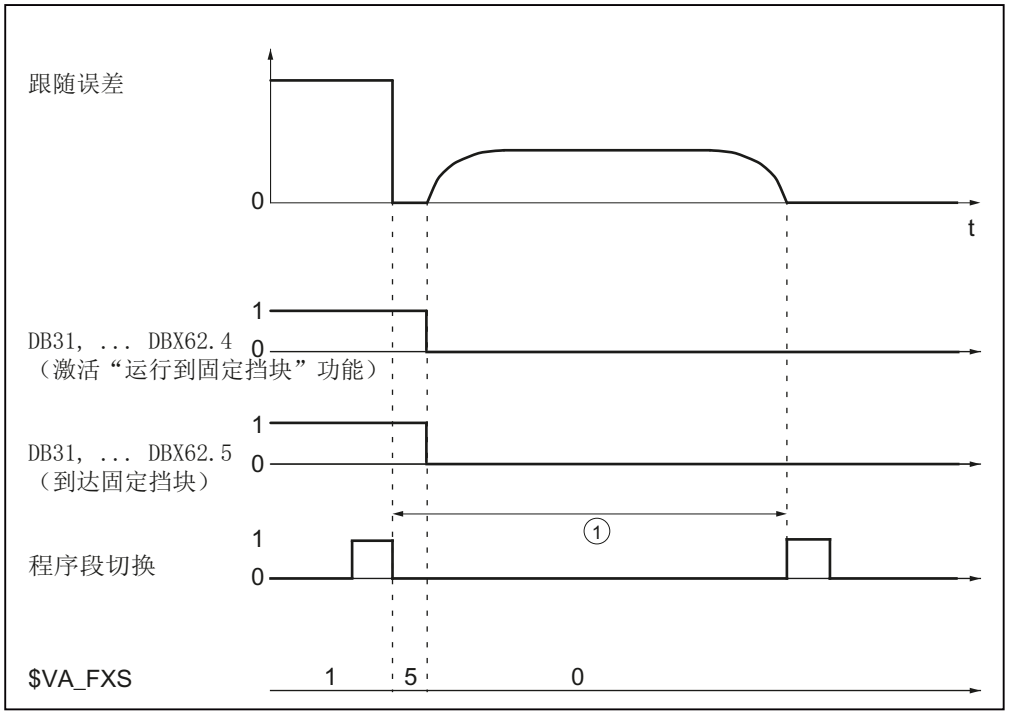
MD37002 $MA_FIXED_STOP_CONTROL，位 0 和位 1（对“运行到固定挡块”功能的顺序控制）

位	值	含义
0	挡块上脉冲**禁用**时的特性	
	0	“运行到固定挡块”功能终止
	1	“运行到固定挡块”功能中断，即驱动不再运转。
1	挡块上脉冲**使能**时的特性	
	0	转矩跳到最大转矩
	1	通过以下机床数据中设定的时间线性启用转矩： MD37012 $MA_FIXED_STOP_TORQUE_RAMP_TIME

特殊情形：取消功能期间删除脉冲使能

若在取消功能期间在“等待 PLC 应答”状态下删除脉冲使能，那么转矩限制会被设置为 0。在此阶段中重新设置脉冲使能时，驱动中将不再生成转矩。 一旦取消完成，便可重新正常运行轴。

概述



① 写入了取消指令 FXS[<轴>]=0 的运行程序段

图 7-4 取消“运行到固定挡块”功能

7.2.3 程序段搜索时的特性

功能

带计算的程序段搜索

- 如果目标程序段位于一个含 FXS 指令的程序段中且轴尚未到达固定挡块，NC 会使轴逼近固定挡块。
- 如果目标程序段位于一个不含 FXS 指令的程序段中且轴还处于那个位置，NC 会使轴离开固定挡块。

- 若轴处于“到达固定挡块”状态，则会显示信息 10208“通过 NC 启动继续执行程序”。通过 NC 启动可继续执行程序。
- 启动目标程序段时，夹紧力矩 FXST 和监控窗口 FXSW 都为正常程序加工时的值。

不带计算的程序段搜索

程序段搜索期间，系统会忽略 FXS、FXST 和 FXSW 指令。

FOCON/FOCOF 的有效性

程序段搜索期间，系统会一同降低模态生效的转矩/力 FOCON/FOCOF，且该状态在逼近程序段的过程中也有效。

带 FXS 或 FOC 的程序段搜索

用户在目标程序段的程序区域内标出 FXS 或 FOC 功能，以获知该加工过程中所有最后有效的状态和功能。NC 在程序测试模式下自行启动所选择的程序。查找到目标程序段后，NC 会停在目标程序段开头处，在内部再次取消程序测试，接着在程序段显示窗口中显示停止条件“搜索目标已找到”。

⚠ 小心
SERUPRO 实际上并没有考虑 FXS 指令 NC 只会在**没有转矩限制**的条件下仿真逼近编程的 FXS 程序段的终点位置。

如果 FXS 指令位于程序开头和搜索目标中间，NC 不会真正地执行该指令，它只会仿真到编程终点的运动。

用户可以在零件程序中同时记录 FXS 的激活和取消。如果有需要，用户可以启动异步子程序，从而在该 SERUPRO 异步子程序中激活或取消 FXS。

系统变量

可通过以下系统变量读取“运行到固定挡块”功能的设定和实际状态：

- $AA_FXS（设定状态）
- $VA_FXS（实际状态）

SERUPRO：$AA_FXS（设定状态）

在执行 SERUPRO 期间，$AA_FXS 会根据“运行到固定挡块”功能的激活状态提供以下值：

“运行到固定挡块”功能的激活状态	系统变量 $AA_FXS ==
“取消”	0（轴不在固定挡块处）
“激活”	3（“运行到固定挡块”选择生效）

说明

在执行 SERUPRO 期间，变量 $AA_FXS 只提供值 0 和 3。因此，如果有基于 $AA_FXS 的程序分支，SERUPRO 指令下的程序运行会和正常的程序加工有所区别。

SERUPRO: $VA_FXS（实际状态）

在执行 SERUPRO 期间，变量 $VA_FXS 始终提供轴的真正状态。

示例

在 SERUPRO 异步子程序中，可通过系统变量 $AA_FXS 和 $VA_FXS 得出“运行到固定挡块”功能的当前状态并对此作出响应：

程序代码：FXS_SERUPRO_ASUP.MPF	注释
N100 WHEN ($AA_FXS[X]==3) AND ($VA_FXS[X]==0) DO FXS[X]=1	; 设定状态==“选择未生效”，且 ; 实际状态==“轴不在固定挡块处” ; => “激活”
N200 WHEN ($AA_FXS[X]==0) AND ($VA_FXS[X]==1) DO FXS[X]=0	; 设定状态==“轴不在固定挡块处”，且 ; 实际状态==“成功逼近固定挡块” ; => “激活”
N1020 REPOSA	; 所有轴再次逼近轮廓

显示 REPOS 偏移

找到搜索目标后，以下 NC/PLC 接口信号会显示每根轴当前的“运行到固定挡块”相关状态：

- DB31, ... DBX62.4（激活“运行到固定挡块”功能）
- DB31, ... DBX62.5（到达固定挡块）

示例：

如果轴停止在固定挡块处且目标程序段在取消 FXS 后依然存在，系统会通过 DB31, ... DBX62.5（到达固定挡块）显示作为 REPOS 偏移的新的目标位置。

REPOS 和 FXS

通过 REPOS 指令，FXS 的功能可以自行重复，在下文以 FXS-REPOS 代指该功能。FXS-REPOS 指令的顺序与 FXS_SERUPRO_ASUP.MPF 程序类似。执行该程序时，NC 会考虑每根轴的情况，并使用搜索目标前最后编程的转矩。

用户可在 SERUPRO 异步子程序中对 FXS 作特殊处理。

这适用于：

在 SERUPRO 异步子程序中执行的每个 FXS 动作都会自动负责

$AA_FXS[<轴>] = $VA_FXS[<轴>]。

这样，FXS-REPOS 对轴 X 无效。

取消 FXS-REPOS

FXS-REPOS 是通过以下方式取消的：

- FXS 同步动作与 REPOSA 指令有关
- SERUPRO 异步子程序中的 $AA_FXS[X] = $VA_FXS[X]

说明

不带 FXS 指令的 SERUPRO 异步子程序或没有 SERUPRO 异步子程序都会自动导致 FXS-REPOS。

⚠ 小心

FXS-REPOS 时轴高速运行

FXS-REPOS 会使所有的轨迹轴同时向目标点运行。这样，带或不带 FXS 指令的轴都会同时以在目标程序段中有效的 G 指令和进给率运行。这会使轴快速 (G0) 或高速逼近固定挡块。

FOC-REPOS

FOC-REPOS 的作用与 FXS-REPOS 类似。

通过 FOC-REPOS 指令无法实现在程序预处理期间出现不断变化的转矩曲线。

示例

轴 X 从位置 0 运行到位置 100。此时 NC 每隔 20 毫米就激活一次 FOC，一次长达 10 毫米。转矩曲线是由非模态 FOC 生成的，因此无法通过 FOC-REPOS 进行跟踪。FOC-REPOS 可使带或不带 FOC 的轴 X 按照在目标程序段前最后编程的位置运行。

FXS“运行到固定挡块”功能的编程示例，参见章节“程序测试 (页 537)”。

7.2.4 复位和功能中断时的特性

NC 复位

只要“运行到固定挡块”功能还未处于“轴成功到达固定挡块”状态，就可以通过 NC 复位中断该功能。

即使已经到达了固定挡块，但还未完全到达指定的固定挡块转矩，就仍可以通过 NC 复位中断该功能。此时，轴的设定位置会与当前实际位置同步。

一旦该功能处于“轴成功到达固定挡块”状态，同样是通过 NC 复位就可以继续保持有效。

功能中断

功能中断是通过以下事件触发的：

- 急停

  ⚠ 小心

  运行到固定挡块时可导致机床故障

  必须要确保在“运行到固定挡块”功能激活期间触发和复位“急停”不会导致机床故障。

  例如：设置和撤销脉冲禁用时的特性：
  MD37002 $MA_FIXED_STOP_CONTROL，位 0（固定挡块处脉冲禁用时的特性）
  - 位 0 = 0：“运行到固定挡块”功能被中断
  - 位 0 = 1：“运行到固定挡块”功能被中断，即驱动不再运转。
    一旦再次取消脉冲禁用，驱动会再次以固定挡块转矩运转。

  说明

  NC 和驱动在“急停”状态下是断开的，也就是说，PLC 必须作出响应。

- 功能状态“未到达固定挡块”
- 功能状态“固定挡块已中断”
- 通过 PLC 用户程序中断：
  DB31, ... DBX62.4 = 0（激活“运行到固定挡块”功能）
- 撤销脉冲使能和机床参数设置：
  MD37002 $MA_FIXED_STOP_CONTROL，位 0 = 0（见上）

7.2.5 与其他功能综合使用时的特性

带剩余行程删除的测量

无法在一个程序段中同时编程“运行到固定挡块”功能 (FXS) 和“带剩余行程删除的测量”功能 (MEAS)。一个功能对一个轨迹轴起作用，另一个功能对定位轴起作用或两者同时对定位轴起作用。

轮廓监控

在“运行到固定挡块”功能激活期间无法进行轮廓监控。

定位轴

在定位轴 POSA 上使用“运行到固定挡块”功能时，即使定位轴在该时间点上仍未到达固定挡块，NC 依然会执行程序段切换。

垂直轴

即使 NC 发出报警信息，仍可以在垂直轴上使用“运行到固定挡块”功能。

如果在运行到固定挡块期间，垂直轴上出现了功能专用的报警，NC 就不会复位 NC/PLC 接口信号 DB11, DBX6.3（运行方式组）。这样，相应的驱动也不会断电。

该功能与电子配重对垂直轴造成的影响一样，可通过以下机床数据进行配置：

MD37052 $MA_FIXED_STOP_ALARM_REACTION

文档

有关垂直轴的其他信息请参考：

- 功能手册 SINAMICS S120
- 功能手册之扩展功能分册；补偿 (K3)，
  章节：电子配重

7.2.6 设定数据

通过功能专用指令 FXS、FXST 和 FXSW 编程的值会和程序段同步写入以下立即生效的轴专用设定数据中：

激活/取消该功能

SD43500 $SA_FIXED_STOP_SWITCH（激活/取消“运行到固定挡块”功能）

夹紧转矩

SD43510 $SA_FIXED_STOP_TORQUE（夹紧转矩）

说明

夹紧转矩超过最大电机转矩的 100%

SD43510 中的夹紧转矩只允许短时间内超过最大电机转矩 100%。除此以外，驱动会对最大电机转矩进行限制。比如：可通过以下驱动参数对最大电机转矩进行限制：

- p1520/p1521 转矩/力上限值/转矩/力下限值
- p1522/p1523 转矩/力上限值/转矩/力下限值
- p1530/p1531 电机功率限值/再生功率限值
- p0640 电流限值
- p0326 电机翻转力修调系数

有关驱动参数和功能的详细信息请参考：

文档

- SINAMICS S120/S150 参数手册
- SINAMICS S120 功能手册

监控窗口

SD43520 $SA_FIXED_STOP_WINDOW（监控窗口）

缺省设置

可通过以下机床数据设置设定数据的缺省设置：

- 夹紧转矩：
  MD37010 $MA_FIXED_STOP_TORQUE_DEF（夹紧转矩的缺省设置）
- 监控窗口：
  MD37020 $MA_FIXED_STOP_WINDOW_DEF（监控窗口的缺省设置）

有效性

夹紧转矩和监控窗口的设定数据是立即生效的。因此，用户或 PLC 用户程序可随时调整夹紧状态以适应不同的加工条件。

文档

有关机床数据和设定数据的详细信息请参考：

参数手册之详细的机床数据描述

7.2.7 系统变量

设定状态/实际状态

可通过以下系统变量读取“运行到固定挡块”功能的设定和实际状态：

- $AA_FXS = <值>（“运行到固定挡块”功能的设定状态）
- $VA_FXS = <值>（“运行到固定挡块”功能的实际状态）

<值>	说明
0	轴不在固定挡块处
1	轴成功到达固定挡块
2	轴未到达固定挡块
3	“运行到固定挡块”选择生效
4	已识别出固定挡块
5	“运行到固定挡块”功能被取消

附加信息

若在运行至固定挡块时出现错误（$VA_FXS == 2），则会在以下系统变量中显示附加信息：

- $VA_FXS_INFO = <值>（“运行到固定挡块”时的附加信息）

<值>	说明
0	无附加信息
1	未编写逼近运动
2	到达编写的终点位置，运动结束
3	通过复位终止，DB21, ... DBX7.7 == 1
4	已离开监控窗口
5	转矩降低已被驱动拒绝
6	PLC 已撤销使能，DB31，... DBX3.1 == 0（“运行到固定挡块”功能已使能）

变量 $AA_FXS 的应用示例

为了实现程序段切换，故障情形下不可触发报警。 故障原因可于随后通过系统变量 $VA_FXS_INFO 获取并相应应对。

前提条件：MD37050 $MA_FIXED_STOP_ALARM_MASK = 0（运行到固定挡块时不触发任何报警）

程序代码	注释
X300 Y500 F200 FXS[X1]=1 FXST[X1]=25 FXSW[X1]=5	; 运行到固定挡块
IF $AA_FXS[X1]==2 GOTOF FXS_ERROR	; IF 固定挡块 == 到达
R100=$AA_IM[X1]	; THEN（正常情形）
IF R100 ...	; 分析当前
...	; 实际位置
GOTOF PROG_END	
FXS_ERROR:	; ELSE（故障情形）
CASE($VA_FXS_INFO[X1]) OF 0 GOTOF LABEL_0 OF 1 GOTOF LABEL_1 ...	; 故障处理
LABEL_0: ...	
GOTOF CASE_END	
LABEL_1: ...	
GOTOF CASE_END	
...	
CASE_END:	
...	
PROG_END: M30	; ENDIF 程序结束

7.2.8 报警

报警 20091 “未到达固定挡块”

如果在执行“运行到固定挡块”功能时未到达目标位置，NC 会显示报警 20091 “未到达固定挡块”并执行程序段切换。

报警 20092 “运行到固定挡块功能仍有效”

如果轴到达固定挡块后，NC 向轴发出了新的运行请求或是功能选择，NC 会显示报警 20092 “运行到固定挡块功能仍有效”。

报警 20093 “固定挡块处的静态监控已触发”

如果轴到达固定挡块后，轴运动的行程超过了设定数据

SD43520 FIXED_STOP_WINDOW（固定挡块监控窗口）

中给定的值，NC 会显示报警 20093 “固定挡块处的静态监控已触发”，取消“运行到固定挡块”功能并设置以下系统变量：
$AA_FXS[x] = 2

报警 20094 “功能已被中断”

如果因撤销脉冲使能而导致无法再输出夹紧转矩或 NC/PLC 接口上的响应信号已经复位，NC 会中断“运行到固定挡块”功能：

- 所需的响应信号：MD37060 $MA_FIXED _STOP_ACKN_MASK，位 0 = 1
- 响应信号：DB31, ... DBX3.1 == 0（激活“运行到固定挡块”功能）

激活固定挡块报警

通过以下机床数据

- 报警 20091 “未到达固定挡块”
- 报警 20094 “固定挡块被中断”

可以设置是否显示固定挡块报警：

MD37050 $MA_FIXED_STOP_ALARM_MASK（激活固定挡块报警）

固定挡块报警时的功能特性可设置

通过以下机床数据可以设置：在出现功能报警时，该功能不会被中断：

- 报警 20090 无法运行到固定挡块
- 报警 20091 未到达固定挡块
- 报警 20092 运行到固定挡块功能仍有效
- 报警 20093 固定挡块处的静态监控已触发
- 报警 20094 运行到固定挡块功能被中断

MD37052 $MA_FIXED_STOP_ALARM_REACTION（固定挡块报警时的响应）

重新编程后的报警抑制

“运行到固定挡块”功能可用于简单的测量过程。

例如：可以使刀具顶住一个指定的障碍物来测量刀具长度，从而判断刀具是否折断。在该情况下需抑制固定挡块报警。在之后“正常”使用该功能来夹紧工件时，需要通过零件程序指令重新激活该报警。

7.2.9 以受限转矩/力 FOC 运行

功能

“Force Control”功能用于在运行时限制转矩/力，借助此功能可将所允许的最大转矩/力限制为轴能达到的最大转矩的百分比值。 此限值可随时修改，轴运行期间亦可。 可在插补周期中依据位移、时间或通过其他任意值推导来进行修改。 “Force Control”功能也可编写在同步动作中。 此功能可模态激活，也可逐段激活。

编程

句法

```
FOCON[<轴>]
FOCOF[<轴>]
FOC[<轴>]
```

含义

参数	含义
FOCON：	激活转矩/力限制
FOCOF：	取消转矩/力限制
FOC：	激活逐段生效的转矩/力限制
<轴>：	通道轴，类型：AXIS

示例

程序代码	注释
N10 FOCON[X]	; 模态激活转矩限制
N20 X100 Y200 FXST[X]=15	; X 以降低的转矩（15%）运行
N30 FXST[X]=75 X20	; 将转矩修改为 75%，
N40 FOCOF[X]	; 取消转矩限制

参数设置

机床数据

- MD37010 $MA_FIXED_STOP_TORQUE_DEF（固定挡块夹紧转矩的缺省设置）
  激活功能后，只要未通过 FXST 显性编写数值，此机床数据中设定的值便一直生效。
- MD36042 $MA_FOC_STANDSTILL_DELAY_TIME（转矩/力限制生效时的静态监控延时）

模态激活（FOCON/FOCOF）

可通过以下机床数据设置：上电或复位后模态激活“Force Control”功能：

MD37080 $MA_FOC_ACTIVATION_MODE，位 0 和位 1（力/转矩模态限制的初始设置）

位	值	含义
0	0	上电后“Force Control”功能不生效。
	1	上电后“Force Control”功能生效。
1	0	复位后“Force Control”功能不生效。
	1	复位后“Force Control”功能生效。

若通过 FXST = <值> 激活前已编写了限制转矩或限制力，那么该限值从激活点开始生效。

若通过 FXST = <值> 激活前未编写限制转矩或限制力，则将启用机床数据中设定的值：

MD37010 $MA_FIXED_STOP_TORQUE_DEF（固定挡块夹紧转矩的缺省设置）

逐段激活（FOC）

使用 FOC 指令为当前程序段激活功能。

通过同步动作激活时，功能一直生效直至当前零件程序段结束。

FXS 的优先级高于 FOC

“运行到固定挡块”FXS 功能的激活较“Force Control”功能具有更高优先级。 若同时激活了“运行到固定挡块”和“Force Control”功能，那么系统会执行前者。

取消“运行到挡块”FXS 功能会引起夹紧取消。 同时模态生效的“Force Control”功能则继续保持生效。

系统变量

“Force Control”功能的状态

可通过系统变量 $AA_FOC 读取“Force Control”功能的状态。

“运行到固定挡块”FXS 功能的状态变化不会改变“Force Control”功能的状态。

$AA_FOC	
值	含义
0	FOC 未生效
1	FOC 模态有效
2	FOC 非模态有效

转矩限制的状态

可通过系统变量 $VA_TORQUE_AT_LIMIT 读取转矩限制的当前状态。

$VA_TORQUE_AT_LIMIT	
值	含义
0	当前生效的转矩 < 转矩限值
1	当前生效的转矩 == 转矩限值

限制

“Force Control”功能有以下局限：

- 转矩/力限制显示为加速度限制，其修改只会在**程序段交界处**应用于运行（参见 ACC 指令）。
- 只针对 FOC： 通过 NC/PLC 接口**无法监控**是否达到生效的转矩限值。
- 由于没有对加速度限制进行调整，运行期间有可能导致跟随误差变大。
- 由于没有对加速度限制进行调整，到达程序段终点所经过的时间可能会比以下机床数据中设定的要长：
  MD36040 $MA_STANDSTILL_DELAY_TIME
  对此，NC 会采用机床数据
  MD36042 $MA_FOC_STANDSTILL_DELAY_TIME
  并在该状态下对其进行监控。

Link 轴和容器轴上的应用

可在一个通道中运行的所有轴（既有 Link 轴，也有容器轴）都能运行至固定挡块。

文档：
功能手册之扩展功能分册；多操作面板配备多 NCU，分布式系统（B3）

容器旋转时依然含有机床轴的状态，即夹紧的机床轴依然位于固定挡块处。

如果使用 FOCON 指令激活了一个模态转矩限制，容器旋转后，该限制依然对机床轴有效。

7.3 示例

示例 1：通过静态同步动作运行到固定挡块

在一个静态动态动作中通过 R 参数（$R1）请求触发“运行到固定挡块”（FXS）。

程序代码	注释
N10 IDS=1 WHENEVER	; 静态同步动作 1：
(($R1==1) AND	; R1==1（请求 Y 的 FXS）以及
($AA_FXS[Y]==0)) DO	; 避免重复选择
	; $AA_FXS[Y]==0（轴不在固定挡块处）： =>
$R1=0 FXS[Y]=1	; 复位 $R1，激活 Y 的 FXS
FXST[Y]=10	; 固定挡块转矩：10%
FA[Y]=200	; 轴向进给率 Y：200
POS[Y]=150	; 定位运动 Y
N11 IDS=2 WHENEVER	; 静态同步动作 2：
($AA_FXS[Y]==4) DO	; $AA_FXS[Y]==4（识别出固定挡块）：=>
FXST[Y]=30	; 固定挡块转矩：30%
N12 IDS=3 WHENEVER	; 静态同步动作 3：
($AA_FXS[Y]==1) DO	; $AA_FXS[Y]==1（轴成功到达固定挡块）：=>
FXST[Y]=$R0	; 固定挡块转矩：R 参数 $R0 中的值
N13 IDS=4 WHENEVER	; 静态同步动作 4：
(($R3==1) AND	; R3==1：请求取消 Y 的 FXS
($AA_FXS[Y]==1)) DO	; $AA_FXS[Y]==1（轴成功到达固定挡块）：=>
FXS[Y]=0	; 取消 FXS
FA[Y]=1000 POS[Y]=0	; 定位运动 Y
N20 FXS[Y]=0 G0 G90 X0 Y0	; 设置初始设置：FXS 已取消，
	; X 和 Y 位于起始位置
N30 RELEASE(Y)	; 使能同步动作中的运动 Y
...	
N60 GET(Y)	将轴 Y 重新纳入轨迹组合中
...	

说明

避免重复选择 FXS

为避免重复选择，建议在激活 FXS 前询问系统变量 $AA_FXS==0 或一个用户专用标志位。见上：程序示例 N10

示例 2：通过逐段同步动作运行到固定挡块

从后续程序段运行的某个位置开始激活“运行到固定挡块”功能

程序代码	注释
`N10 G0 G90 X0`	`; 起始位置`
`N20 WHEN $AA_IW[X] > 17 DO FXS[X]=1`	`; 同步动作：轴的实际位置 X > 17`
	`; => 为 X 轴激活 FXS`
`N30 G1 F200 X100`	`; X 轴的运行`

说明

逐段同步动作会在后续主处理程序段中执行。

7.4 数据表

7.4.1 机床数据

7.4.1.1 进给轴/主轴专用机床数据

编号	名称：$MA_	说明
36042	FOC_STANDSTILL_DELAY_TIME	转矩/力限制激活时的静态监控延迟时间
37000	FIXED_STOP_MODE	“运行到固定挡块”功能模式
37002	FIXED_STOP_CONTROL	“运行到固定挡块”功能的流程控制
37010	FIXED_STOP_TORQUE_DEF	固定挡块夹紧转矩的缺省设置
37012	FIXED_STOP_TORQUE_RAMP_TIME	到达修改后的转矩限值所需的时间。
37020	FIXED_STOP_WINDOW_DEF	固定挡块监控窗口的缺省设置
37030	FIXED_STOP_THRESHOLD	固定挡块识别的阈值

编号	名称：$MA_	说明
37040	FIXED_STOP_BY_SENSOR	通过传感器识别固定挡块
37050	FIXED_STOP_ALARM_MASK	激活固定挡块报警
37052	FIXED_STOP_ALARM_REACTION	固定挡块报警时的响应
37060	FIXED_STOP_ACKN_MASK	是否等待 PLC 响应来执行“运行到固定挡块”功能
37070	FIXED_STOP_ANA_TORQUE	为模拟的驱动执行“运行到固定挡块”功能时的转矩限值
37080	FOC_ACTIVATION_MODE	模态转矩/力限制的初始设置

7.4.2 设定数据

7.4.2.1 进给轴/主轴专用设定数据

编号	名称：$SA_	说明
43500	FIXED_STOP_SWITCH	选择“运行到固定挡块”功能
43510	FIXED_STOP_WINDOW	固定挡块夹紧转矩
43520	FIXED_STOP_TORQUE	固定挡块监控窗口

7.4.3 信号

7.4.3.1 发送至进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
对到达固定挡块发出响应	DB31, ...DBX1.1	DB380x.DBX1.1
固定挡块传感器	DB31, ...DBX1.2	DB380x.DBX1.2
进给轴/主轴禁止	DB31, ...DBX1.3	DB380x.DBX1.3
伺服使能	DB31, … .DBX2.1	DB380x.DBX2.1
运行到固定挡块使能	DB31, … .DBX3.1	DB380x.DBX3.1

7.4.3.2 来自进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
激活“运行到固定挡块”功能	DB31, ...DBX62.4	DB390x.DBX2.4
到达固定挡块	DB31, ...DBX62.5	DB390x.DBX2.5

G2：速度、设定值/实际值系统、闭环控制 8

8.1 简要说明

该功能手册描述的是如何设置机床轴参数，其中包括：

- 实际值系统或测量系统
- 设定值系统
- 操作精度
- 运行范围
- 进给轴速度
- 控制参数

8.2 速度、运行范围、精度

8.2.1 速度

最大轨迹速度、进给轴速度和主轴转速

最大轨迹速度、进给轴速度和主轴转速受机床结构、驱动动态响应配置和实际值采集装置的极限频率（编码器极限频率）影响。

最大进给轴速度是在以下机床数据：

MD32000 $MA_MAX_AX_VELO（最大轴速度）

中定义的。

允许的最大主轴转速是通过以下机床数据：

MD35100 $MA_SPIND_VELO_LIMIT（最大主轴转速）

设定的。

说明参见章节“S1：主轴 (页 1401)”。

进给率较大时（编程的进给率较大或受进给倍率的影响），速度会被限制为 $V_{最大}$。

由 CAD 系统生成的程序，其程序段很短，因而自动限制最大进给率可能导致速度降低。

示例：

插补周期 = 12 毫秒

```
N10 G0 X0 Y0; [mm]
N20 G0 X100 Y100; [mm]
```

⇒ 程序段中编程的路径长度 = 141.42 毫米

⇒ $V_{最大}$ = (141.42 毫米/12 毫秒) * 0.9 = 10606.6 毫米/秒 = 636.39 米/分

最小轨迹速度、进给轴速度

最小轨迹速度或进给轴速度有以下限制：

$$V_{min} \geq \frac{10^{-3}}{\text{计算精度} \quad [\frac{\text{增量}}{\text{毫米或度}}] \quad * \quad \text{插补周期} \quad [s]}$$

计算精度是借助机床数据：
MD10200 $MN_INT_INCR_PER_MM（线性位置计算精度）

或
MD10210 $MN_INT_INCR_PER_DEG（角位置计算精度）

定义的：

值低于 $V_{最小}$ 时，无法运行。

示例：

MD10200 $MN_INT_INCR_PER_MM = 1000 [增量/毫米]；

插补周期 = 12 毫秒；

⇒ $V_{最小}$ = 10^{-3} / (1000 $^{增量}/_{毫米}$ x 12 毫秒) = 0.005 $^{毫米}/_{分}$；

进给率的取值范围取决于所选的计算精度。

机床数据为缺省设置时：

MD10200 $MN_INT_INCR_PER_MM
（线性位置计算精度）（1000 增量/毫米）

或者

MD10210 $MN_INT_INCR_PER_DEG

（角位置计算精度）（1000 增量/度）

可通过指定的精度编程以下取值范围：

几何轴上的轨迹进给率 F 的取值范围:	
公制系统：	英制系统：
0.001 ≤ F ≤ 999999.999 [毫米/分、毫米/转、度/分、度/转]	0.001 ≤ F ≤ 399999.999 [英寸/分、英寸/转]

定位轴上的进给率的取值范围：	
公制系统：	英制系统：
0.001 ≤ FA ≤ 999999.999 [毫米/分、毫米/转、度/分、度/转]	0.001 ≤ FA ≤ 399999.999 [英寸/分、英寸/转]

主轴转速 S 的取值范围：	0.001 ≤ S ≤ 999999.999 [转/分]

如果提高/降低了计算精度，取值范围也会发生相应的变化。

8.2.2 运行范围

运行范围的取值

运行范围的取值取决于所选的计算精度。

机床数据为缺省设置时：

MD10200 $MN_INT_INCR_PER_MM

（线性位置计算精度）（1000 增量/毫米）

或者

MD10210 $MN_INT_INCR_PER_DEG

（角位置计算精度）（1000 增量/度）

可通过指定的精度编程以下取值范围：

表格 8-1　轴运行范围

	G71 [毫米、度]	**G70 [英寸、度]**
	区域	**区域**
线性轴 X、Y、Z、...	∓ 999999.999	∓ 399999.999
回转轴 A、B、C、...	∓ 999999.999	∓ 999999.999
插补参数 I、J、K	∓ 999999.999	∓ 399999.999

回转轴的尺寸单位始终为度。

如果计算精度提高/降低了 10 倍，取值范围也会发生相应的变化。

运行范围可以由软件限位开关和工作区加以限制（参见章节“A3:轴监控 (页 91)”）。

线性轴和回转轴的运行范围较大时的特殊性参见章节“R1：回参考点 (页 1349)”。

回转的运行范围可由机床数据加以限制。

文档：

功能手册之扩展功能分册；回转轴 (R2)

8.2.3　控制系统的定位精度

实际值精度和计算精度

控制系统的定位精度取决于实际值精度（即编码器增量/毫米或度）和计算精度（即内部增量/毫米或度）。

两个精度中较低的那个决定了控制系统的定位精度。

输入精度、插补周期和位置控制周期的选择对该精度没有任何影响。

除了可以用 MD32000 限制最大轨迹速度外，某些情况下控制系统也可以根据下列公式来限制最大轨迹速度：

$$\frac{\text{内部增量/mm}}{\text{编码器增量/mm}} = \frac{1}{\text{ENC\_RESOL [n] * ENC\_PULSE\_MULT[n]}} * \frac{\text{DRIVE\_ENC\_RATIO\_NUMERA [n]}}{\text{DRIVE\_ENC\_RATIO\_DENOM [n]}} * \frac{\text{DRIVE\_AX\_RATIO\_DENOM [n]}}{\text{DRIVE\_AX\_RATIO\_NUMERA [n]}} * \text{LEADSCREW\_PITCH} * \text{INT\_INCR\_PER\_MM}$$

8.2.4 输入/显示精度，计算精度

精度：区别

线性位置和角位置、速度、加速度和急动度的精度可分为以下三种：

- 输入精度
  通过操作面板或零件程序输入数据。
- 显示精度
  通过操作面板显示数据。
- 计算精度
  内部换算通过操作面板或零件程序输入的数据。

输入和显示精度是在操作面板上设定的，可通过以下机床数据：

MD9004 $MM_DISPLAY_RESOLUTION（显示精度）

修改位置值的显示精度。

通过机床数据：

MD9011 $MM_DISPLAY_RESOLUTION_INCH（英制单位制显示精度）

可将位置值的显示精度设为英制。

设为英制时，最多可以显示小数点后的六位。

零件程序中的编程应采用在编程说明中介绍的输入精度。

所需的计算精度是由机床数据：

MD10200 $MN_INT_INCR_PER_MM（线性位置计算精度）

与

MD10210 $MN_INT_INCR_PER_ DEG（角位置计算精度）

定义的。

这两个值与输入/显示精度无关，但它们至少应具有相同的精度。

利用计算精度可以确定零件程序中的位置值、速度等的小数点后的最大有效位，也可以确定刀具补偿值、零点偏移值等的小数点后的有效位（这样可以达到最高的精度）。

编程值的乘积通过计算精度取整，从而使输入的角度位置和线性位置的精度被限制在计算精度内。

为了确保所执行的取整易于跟踪，建议使用 10 的幂来表示计算精度。

取整示例：

计算精度：1000 增量/毫米

编程的行程：97.3786 毫米

有效值 = 97.379 毫米

$^1/_{10}$-µm 范围内的编程示例：

所有线性轴都应在 0.1 - 1000 µm 范围中编程及运行。

⇒ 为精确定位到 0.1µm，必须将计算精度设为 ≥ 10^4 增量/毫米。

⇒ MD10200 $MN_INT_INCR_PER_MM = 10000 [增量/毫米]：

⇒ 相关的零件程序示例：

程序代码	注释
N20 G0 X 1.0000 Y 1.0000	; 轴运行到位置上 X=1.0000 毫米、Y=1.0000 毫米
N25 G0 X 5.0002 Y 2.0003	; 轴运行到位置上 =5.0002 毫米、Y=2.0003 毫米

8.2.5 机床数据和设定数据物理量的定标

输入/输出单位

根据所使用的基本单位制（公制/英制），有物理单位的机床数据和设定数据是以下列输入/输出单位来说明的：

物理单位	标准基本单位制的输入/输出单位	
	公制	英制
线性位置	1 毫米	1 英寸
角度位置	1 度	1 度
线性速度	1 毫米/分钟	1 英寸/分钟
角速度	1 转/分	1 转/分
线性加速度	1 米/秒 ^{2}	1 英寸/秒 ^{2}
角加速度	1 转/秒 ^{2}	1 转/秒 ^{2}
线性急动度	1 米/秒 ^{3}	1 英寸/秒 ^{3}
角急动度	1 转/秒 ^{3}	1 转/秒 ^{3}
时间	1 秒	1 秒
位置控制回路增益	1/秒	1/秒
旋转进给率	1 毫米/转	英寸/转
线性位置补偿值	1 毫米	1 英寸
角位置补偿值	1 度	1 度

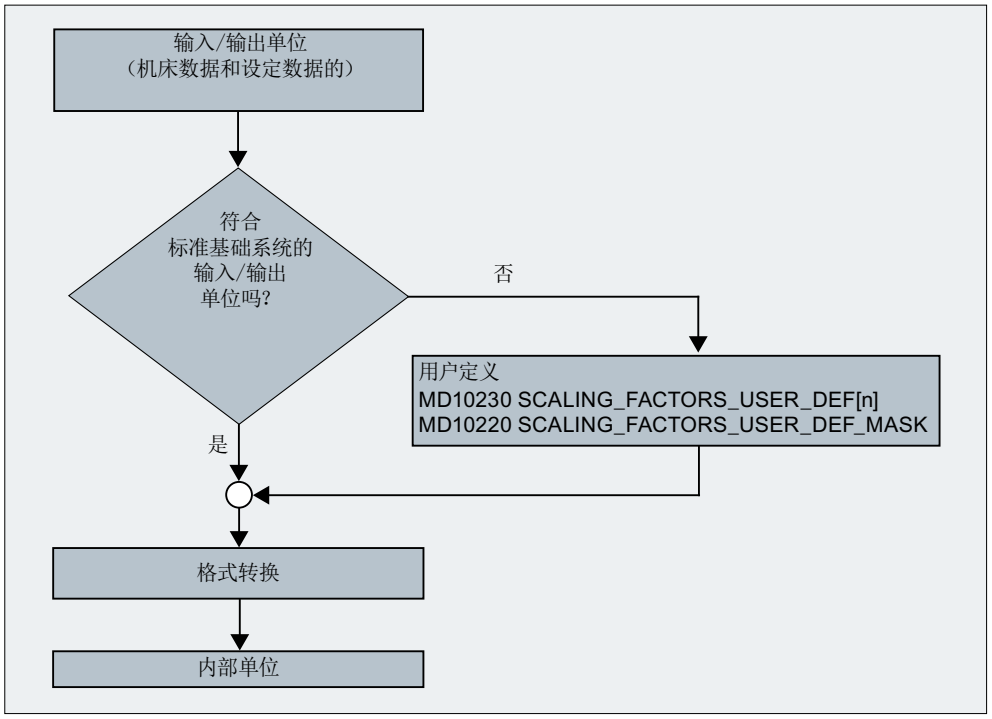
控制系统内部使用以下单位，与所选择的基本单位制无关：

物理单位	单位
线性位置	1 毫米
角度位置	1 度
线性速度	1 毫米/秒
角速度	1 度/秒
线性加速度	1 毫米/秒 ^{2}
角加速度	1 度/秒 ^{2}
线性急动度	1 毫米/秒 ^{3}
角急动度	1 度/秒 ^{3}
时间	1 秒

物理单位	单位
位置控制回路增益	1/秒
旋转进给率	1 毫米/度
线性位置补偿值	1 毫米
角位置补偿值	1 度

用户可以为机床数据和设定数据定义其他的输入/输出单位。为此必须通过以下机床数据实现新选择的输入/输出单位和内部单位之间的匹配：

- MD10220 $MN_SCALING_USER_DEF_MASK
- MD10230 $MN_SCALING_FACTORS_USER_DEF[n]



其中：

所选的输入/输出单位 = （MD10230 $MN_SCALING_FACTORS_USER_DEF[n]）* 内部单位

因此在机床数据 MD10230 $MN_SCALING_FACTORS_USER_DEF[n] 中须相应输入所选择的输入/输出单位，以内部单位 1 毫米、1 度和 1 秒表示。

示例 1：

线性速度的机床数据输入/输出应该以米/分替代毫米/分（初始设置）。

（内部单位为毫米/秒）

⇒ 线性速度的定标系数应不同于初始设置，为此，必须在机床数据：

MD10220 $MN_SCALING_USER_DEF_MASK

中将位号设为 2。

⇒ MD10220 $MN_SCALING_USER_DEF_MASK = 'H4'；（位号 2 作为十六进制值）

⇒ 线性速度的定标系数应不同于初始设置，为此，必须在机床数据：

MD10220 $MN_SCALING_USER_DEF_MASK

中将位号设为 2。

⇒ MD10220 $MN_SCALING_USER_DEF_MASK = 'H4'；（位号 2 作为十六进制值）

⇒ 定标系数按下列公式计算：

$$\text{MD10230 SCALING\_FACTORS\_USER\_DEF[n]} = \frac{\text{所选的输入/输出单位}}{\text{内部单位}}$$

$$\text{MD10230 SCALING\_FACTORS\_USER\_DEF[n]} = \frac{1\ \frac{\text{m}}{\text{min}}}{1\ \frac{\text{mm}}{\text{s}}} = \frac{\frac{1000\ \text{mm}}{60\ \text{s}}}{1\ \frac{\text{mm}}{\text{s}}} = \frac{1000}{60} = 16.667;$$

⇒ MD10230 SCALING_FACTORS_USER_DEF[n] = 16.667

下标 n 定义了“物理量定标系数”列表中的“线性速度”。

示例 2：

另外针对示例 1 的改变，机床数据输入/输出以（基本设置）英尺/秒 ² 代替米/秒 ²。（内部单位为毫米/秒 ²）

⇒ MD10220 SCALING_USER_DEF_MASK = H14 ; （示例 1 的位编号 4 和 2 作为十六进制值）

⇒ MD10220 SCALING_FACTORS_USER_DEF[4] = $\frac{1\ \frac{ft}{s^2}}{1\ \frac{mm}{s^2}} = \frac{12*25{,}4\ \frac{mm}{s^2}}{1\ \frac{mm}{s^2}} = 304.8$;

⇒ MD10220 SCALING_FACTORS_USER_DEF[4] = 304.8

下标 4 定义了“物理量定标系数”列表中的“线性加速度”。

8.3 英制/公制单位制

8.3.1 功能

8.3.1.1 参数设置和编写的单位制

SINUMERIK 控制系统既可采用公制单位制，也可采用英制单位制工作。

参数设置的单位制（基本单位制）

单位制的基本设置（基本单位制）在机床数据 MD10240 $MN_SCALING_SYSTEM_IS_METRIC 中设置（参见“调试 (页 376)”）。

根据基本单位制，所有与长度相关的数据均以公制或英制单位制表示：

公制单位制：	mm，mm/min，m/s2，m/s3，mm/rev
英制单位制：	inch，inch/min，inch/s2，nch/s3，inch/rev

单位制也决定了为线性轴编写的 F 值的含义：

进给类型	公制单位制	英制单位制
G94	mm/min	mm/min
G95	mm/rev	inch/rev

编写的单位制

通过 G 功能组 13 的指令（英制/公制）可在零件程序内在公制与英制之间进行切换（参见“编程 (页 380)”）。

在任何时候，编写的单位制与基本单位制可相同或不同。借助在零件程序段中切换单位制，例如可以在采用公制基本单位制的情况下在工件上加工英制螺纹。

G 功能组的初始设置在机床数据 MD20150 $MC_GCODE_RESET_VALUES 中定义（参见“调试 (页 376)”）。

HMI 显示

在操作界面上如下显示长度相关数据：

	显示	
	以基本单位制	以编写的单位制
机床数据	x	
机床坐标系中的数据	x	
刀具数据	x	
零点偏移	x	
工件坐标系中的数据		x

NC/PLC 接口

对于包含尺寸相关信息的 NC/PLC 接口信号，如轨迹轴和定位轴的进给率，以基本单位制与 PLC 进行数据交换。

读入外部零件程序

在从外部读入零件程序时，若其包含的数据组（零点偏移、刀具补偿等）的编程所采用的单位制不同于基本单位制，则须预先修改基本单位制。

8.3.1.2 扩展单位制功能

自 SW 5 起提供扩展单位制功能，其使得单位制切换大幅简化。

新功能为：

- 通过操作界面切换单位制 (页 374)
- 新的 G 指令 G700/G710（参见“编程 (页 380)”）

- 借助单位制标识 INCH/METRIC 进行数据备份（参见“调试 (页 376)”）
- 在单位制切换时自动换算数据，例如针对：
  - 零点偏移
  - 补偿数据（EEC，QEC）
  - 刀具补偿
  - ...

出于兼容性原因，必须通过机床数据激活扩展单位制功能（参见“调试 (页 376)”）。

8.3.1.3 通过操作界面切换单位制

前提条件

扩展单位制功能生效（MD10260 $MN_CONVERT_SCALING_SYSTEM = 1；参见“调试 (页 376)”）

功能

通过 HMI 操作界面在“加工”操作区中的相应软键对控制系统进行单位制切换。

仅在下列条件下执行单位制切换：

- 各通道中 MD20110 $MC_RESET_MODE_MASK 位 0 置位。
- 所有通道都处于复位状态。
- 轴当前未通过 JOG、DRF 或 PLC 运行。
- 恒定砂轮圆周速度 （SUG）未处于激活状态。

在切换单位制期间，某些动作（如零件程序启动或运行模式切换）被禁用。

原本通过写入所需所有机床数据，然后使其生效来进行单位制切换。

此时系统会为配置的所有通道自动并统一地切换以下机床数据：

- MD10240 $MN_SCALING_SYSTEM_IS_METRIC
- MD20150 $MN_GCODE_RESET_VALUES

初始设置

在通过操作界面切换单位制时，系统会自动将 G 功能组 13 的初始设置调整至当前单位制。

MD20150 $MC_GCODE_RESET_VALUES[12] = <当前单位制>

NCU-Link

说明

若通过 NCU-Link 连接了多个 NCU，则会在连接的所有 NCU 上统一切换。若连接的 NCU 中的一个不满足切换前提条件，则不在任何 NCU 上进行切换。因此应在现有的 NCU-Link 上进行跨 NCU 的插补，该插补仅在统一的单位制中才能给出正确的结果。

文档：

功能手册，扩展功能；章节“B3：分布式系统”

系统数据

切换单位制时，从操作员的角度而言，所有与长度相关的数据都自动切换成新的单位制。

其中包括：

- 位置
- 进给率
- 加速度
- 加加速度
- 刀具补偿
- 可编程，可设置和外部的零点偏移，DRF 偏移
- 补偿值
- 保护区
- 机床数据
- JOG 和手轮权重

切换后，所有物理量数据均可用。

未明确定义物理单位的数据**无法**自动换算。

其中包括：

- R 参数
- GUD (**G**lobal **U**ser **D**ata)
- LUD (**L**ocal **U**ser **D**ata)
- PUD (**P**rogram global **U**ser **D**ata)

- 模拟量输入/输出
- 通过 FC21 进行的数据交换

在此用户需要考虑当前有效的单位制（MD10240 $MN_SCALING_SYSTEM_IS_METRIC）。

NC/PLC 接口

可在 NC/PLC 接口上通过以下信号读取当前单位制设置：

DB10 DBX107.7（英制单位制）

可通过以下信号字节读取自上一次控制系统启动以来的单位制切换次数。

DB10 DBB71（修改计数器，英制/公制单位制）

控制系统启动后的起始值 1

8.3.2 调试

NC 专用机床数据

基本单位制

通过以下机床数据定义在数据输入/输出时，控制系统为缩放长度相关物理量而使用的基本单位制：

MD10240 $MN_SCALING_SYSTEM_IS_METRIC

值	含义
0 (= FALSE)	长度相关数据作为**英制**尺寸数据。
1 (= TRUE)	长度相关数据作为**公制**尺寸数据。

修改此机床数据后需要启动，否则相应具有物理单位的机床数据会被错误地标准化。

遵守以下步骤：

- 通过手动输入修改 MD：
  - 1. 执行启动。
  - 2. 具有物理单位的机床数据：输入值。
- 通过机床数据文件修改 MD：
  - 1. 执行启动。
  - 2. 再次载入机床数据文件，从而将新的物理范围考虑在内。

在修改 MD10240 时，系统会发出报警 4070“标准化机床数据被修改”。

说明

在控制系统上进行单位制切换后，必须将所有长度相关数据一致且完整地换算至另一单位制。

换算系数（NC 专用）

从公制到英制的换算系数在以下机床数据中设置：

MD10250 $MN_SCALING_VALUE_INCH（针对英制的换算系数）

缺省值：25.4

在选择非公制基本单位制（MD10240 $MN_SCALING_SYSTEM_IS_METRIC = 0）时，此换算系数生效。

系统会将下列数据与换算系数相乘：

- 输入/输出时的长度相关数据（例如在上载机床数据、零点偏移时）
- 为线性轴编写的 F 值
- 在用 G70/G71 编写的单位制不同于基本单位制（MD10240 $MN_SCALING_SYSTEM_IS_METRIC）的情况下，编写的轴几何数据（位置、多项式系数、圆弧编程中的半径等）

说明

通过修改换算系统，能够将控制系统调整成用户专用的单位制。

扩展单位制功能

扩展单位制功能可通过以下兼容性机床数据激活：

MD10260 $MN_CONVERT_SCALING_SYSTEM

值	含义
0 (= FALSE)	扩展单位制功能不生效（缺省设置；与先前的软件版本兼容）
1 (= TRUE)	扩展单位制功能生效

在修改 MD10240 时，系统会发出报警 4070“标准化机床数据被修改”。

数据备份

在读取过程中，从控制系统读取并具有单位制相关数据的各数据组会得到一个标识，其对应当前的单位制设置：

- MD10260 $MN_CONVERT_SCALING_SYSTEM
- MD10240 $MN_SCALING_SYSTEM_IS_METRIC

通过此标识可确定以哪种单位制读取数据。这样可避免将具有与当前设置不同的单位制的数据组读入控制系统。

在零件程序中也会分析该标识，故能以上述方式防止误操作。这样可以防止例如只包含公制数据的零件程序在英制系统中运行。

采用以下设置时，存档和机床数据组向下兼容：

MD11220 $MN_INI_FILE_MODE = 2

说明

仅在以下兼容性机床数据置位时，才会生成 INCH/METRIC 标识：

MD10260 $MN_CONVERT_SCALING_SYSTEM = TRUE

说明

机床数据取整

为了避免取整问题，所有与长度相关的机床数据在以英制单位制 (MD10240 $MN_SCALING_SYSTEM_IS_METRIC=0 和 MD10260 $MN_CONVERT_SCALING_SYSTEM=1) 写入时都被取整为 1 pm。

这样一来，以英制单位制读取备份数据时因转换为 ASCII 而产生的精度损失会在数据读入系统中时重新得到修正。

输入和计算精度

输入精度和计算精度以每毫米的内部增量数定义。

MD10200 $MN_INT_INCR_PER_MM

线性位置的输入精度被限制为计算精度。此时编写的位置值同计算精度的乘积会取整为最接近的整数。为了使取整易于理解，建议使用 10 的幂来表示计算精度。

示例：

1 Inch = 25.4 mm ⇒ 0.0001 Inch = 0.00254 mm = 2.54 µm = 2540 nm

为了编写和显示最后的 40 nm，必须将输入精度和计算精度的值设为 100000。

两种单位制的给定值必须相同，这样才能保证单位制的切换不会损失精度。如果已做过此设定，不需要每次转换系统时，对 MD10200 进行修改。

位置表的单位制

分度轴表格中的位置数据及软件凸轮切换点的单位制是通过以下机床数据进行配置的：

MD10270 $MN_POS_TAB_SCALING_SYSTEM

文档：
功能手册，扩展功能；章节“N3：软件凸轮，行程开关信号”和“T1：分度轴”

用户刀具数据

对于用户自定义的刀具数据和刀沿数据，可通过以下机床数据为其设置物理单位：

- MD10290 $MN_CC_TDA_PARAM_UNIT
- MD10292 $MN_CC_TOA_PARAM_UNIT

说明
切换单位制时，所有长度相关刀具数据均会换算至新的单位制。

通道专用机床数据

初始设置

在以下机床数据中为每个通道设置 G 功能组 13（英制/公制单位制）的初始设置：

MD20150 $MC_GCODE_RESET_VALUES[12]（G 功能组 13 的初始设置）

轴/主轴专用机床数据

换算系数（轴专用）

为使定位轴独立于 G70/G71，可为每个轴单独设置从公制到英制单位制的换算系数：

MD31200 $MA_SCALING_FACTOR_G70_G71（G70/G71 指令激活时的换算系数）

缺省值：25.4

说明
所有三个几何轴的换算系数应相同。

JOG 和手轮权重

以下机床数据包含两个值，分别包含针对两个单位制中的一个的轴专用增量加权：

MD31090 $MA_JOG_INCR_WEIGHT

根据 MD10240 SCALING_SYSTEM_IS_METRIC 的当前设置，控制系统自动选择匹配值。

用户在调试阶段就已经确定了两个增量加权。

示例：增量加权，用于第 1 轴

- 公制：
  MD31090 $MA_JOG_INCR_WEIGHT[0 ; AX1] = 0.001 mm
- 英制：
  MD31090 $MA_JOG_INCR_WEIGHT[1 ; AX1] = 0.00254 mm ≙ 0.0001 inch

这样就无需在每次切换英制/公制时写入 MD31090。

由于所有内部位置始终以 mm 为基准，切换单位制时 JOG 增量运行不会累加任何余程。

垂度补偿的单位制

垂度补偿的单位制是通过以下机床数据进行配置的：

MD32711 $MA_CEC_SCALING_SYSTEM_METRIC

文档：
功能手册，扩展功能；章节“K3：补偿”

8.3.3 编程

8.3.3.1 切换单位制 (G70/G71/G700/G710)

通过 G 指令组 13 的指令（英制/公制）可在零件程序范围内在公制与英制之间进行切换。

句法

```
G70/G700
...
G71/G710
```

含义

G70：	激活英制单位制 在英制单位制中读取和写入和长度相关的几何数据。 在设置的基本单位制中读取和写入和长度相关的的工艺数据（如进给率、刀具补偿、可设定零点偏移、机床数据和系统变量）。
G71：	激活公制单位制 在公制单位制中读取和写入和长度相关的的几何数据。 在设置的基本单位制中读取和写入和长度相关的的工艺数据。
G700：	激活英制单位制 在英制单位制中读取和写入所有和长度相关的几何数据和工艺数据。
G710：	激活公制单位制 在公制单位制中读取和写入所有和长度相关的几何数据和工艺数据。

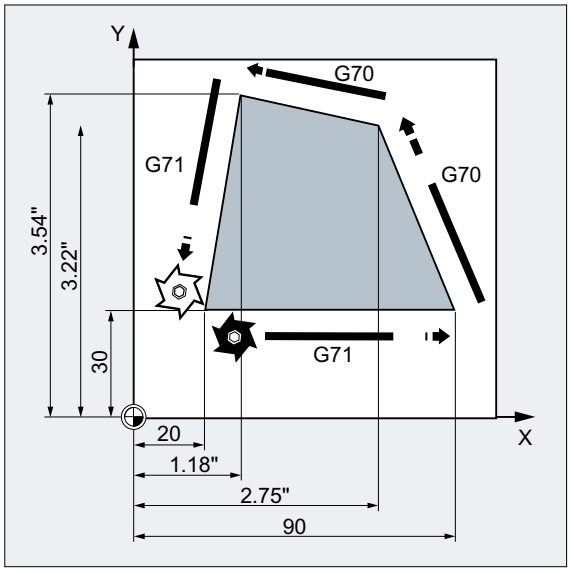
说明

G700/G710

指令 G700 和 G710 仅在扩展单位制功能激活时可用 (MD10260 $MN_CONVERT_SCALING_SYSTEM = 1)。

示例

基本单位制为公制 (MD10240 $MN_SCALING_SYSTEM_IS_METRIC = 1)。但工件图纸中的尺寸数据为英制。因此会在零件程序中切换为英制。根据英制尺寸数据，会再次切换为公制。



程序代码	注释
N10 G0 G90 X20 Y30 Z2 S2000 M3 T1	；X=20 毫米，Y=30 毫米，Z=2 毫米，F= 快速运行 毫米/分钟
N20 G1 Z-5 F500	；Z=-5 毫米，F=500 毫米/分钟
N30 X90	；X=90 毫米
N40 **G70** X2.75 Y3.22	；编写的单位制：英制
	X=2.75 英寸，Y=3.22 英寸，F=500 毫米/分钟
N50 X1.18 Y3.54	；X=1.18 英寸，Y=3.54 英寸，F=500 毫米/分钟
N60 **G71** X20 Y30	；编写的单位制：公制
	；X=20 毫米，Y=30 毫米，F= 500 毫米/分钟
N70 G0 Z2	；Z=2 毫米，F=快速运行 毫米/分钟
N80 M30	；程序结束

更多信息

在 G70/G71 和 G700/G710 指令下读写数据

数据区	G70 / G71		G700 / G710	
	读取	写入	读取	写入
显示逗号后的位数 (WCS)	P	P	P	P
显示逗号后的位数 (MCS)	G	G	G	G
进给率	G	G	P	P
位置数据 X、Y、Z	P	P	P	P
插补参数 I、J、K	P	P	P	P
圆半径 (CR)	P	P	P	P
极半径 (RP)	P	P	P	P
螺距	P	P	P	P
可编程的框架	P	P	P	P
可设定框架	G	G	P	P
基本框架	G	G	P	P
外部零点偏移	G	G	P	P
轴预设偏移	G	G	P	P
工作区域限制 (G25/G26)	G	G	P	P
保护区	P	P	P	P
刀具补偿	G	G	P	P
与长度相关的机床数据	G	G	P	P
与长度相关的设定数据	G	G	P	P
与长度相关的系统变量	G	G	P	P
GUD	G	G	G	G
LUD	G	G	G	G
PUD	G	G	G	G
R 参数	G	G	G	G
西门子循环	P	P	P	P
增量加权 点动/手轮	G	G	G	G
P：以编写的单位制读写 G：以参数设置的基本单位制读写				

同步动作

说明

读取同步动作中的位置数据

如果在同步动作（条件部分和/或动作部分或工艺功能）中未显式编程程序单位制，系统将始终以**基本单位制**读取同步动作中的**与长度相关的位置数据**。

文档：功能手册之同步动作

8.4 设定值/实际值系统

8.4.1 简介

控制环

可为闭环控制的进给轴/主轴配置具有以下结构的控制环：

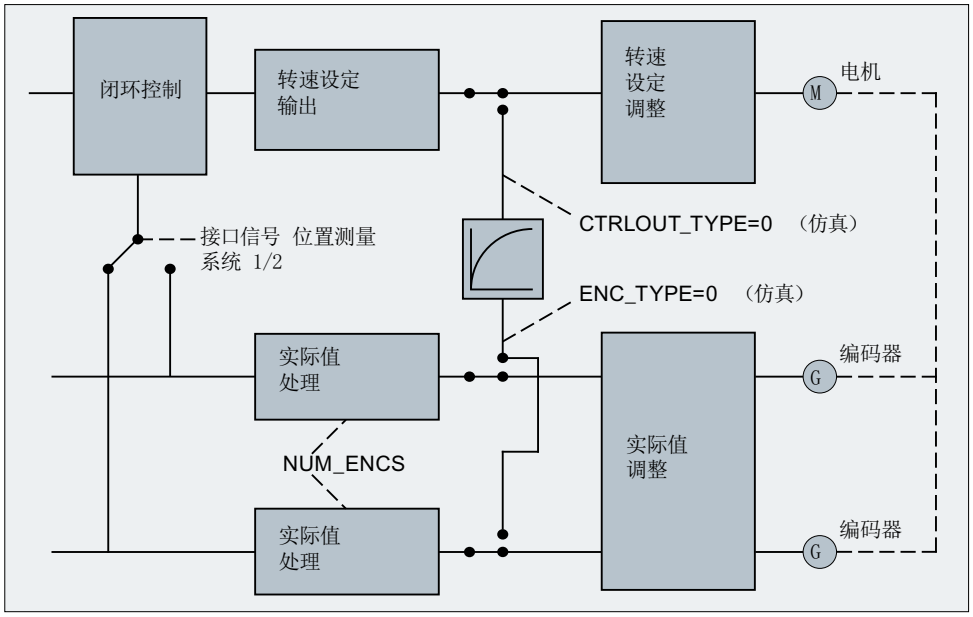


图 8-1 控制环原理图

设定值输出

可为每根进给轴/主轴输出一个设定值报文。在 SINUMERIK 840D sl 上将设定值输出至调节器。

实际值采集

每根进给轴/主轴最多可连接两个测量系统，例如：一个直接测量系统用于对精度有较高要求的加工过程，一个间接测量系统用于快速定位任务。

所使用的编码器数量输入在以下机床数据中：

MD30200 $MA_NUM_ENCS（编码器数量）

可在现有的两个实际值分支上进行实际值采集。

位置闭环控制、绝对值计算和显示始终会采用激活的测量系统。如果通过 PLC 接口同时激活了这两个测量系统，控制系统内部会选择位置测量系统 1。

回参考点运行由所选的测量系统执行。

每个位置测量系统必须分开回参考点。

有关编码器监控的详细说明请参见章节“A3:轴监控 (页 91)”。

有关实际值采集补偿功能的详细说明请参见：
文档：
功能手册之扩展功能分册；补偿 (K3)

切换测量系统

通过以下 NC/PLC 接口信号可在两个测量系统之间进行切换：

DB31, ... DBX1.5（位置测量系统 1）

DB31, ... DBX1.6（位置测量系统 2）

更多信息参见章节“A2: 不同的 NC/PLC 接口信号与功能 (页 43)”。

可随时在两个测量系统之间进行切换，与轴的静止状态无关，但只有在未超过允许的两个测量系统实际值之间的偏差时才能实现切换。

相应的公差输入在以下机床数据中：

MD36500 $MA_ENC_CHANGE_TOL（位置实际值切换时的最大公差）

切换时，位置测量系统 1 和 2 之间的当前偏差会立即运行。

监控

允许的两个测量系统实际值之间的偏差必须输入在以下机床数据中：

MD36510 $MA_ENC_DIFF_TOL

在循环比较所使用的两个测量系统时，不能超过这个差值，否则会生成报警 25105 “测量系统差值过大”。

如果轴中的 2 个测量系统无一生效或是轴未回参考点（至少是在当前的闭环测量系统中），在 MD36510 = 0 时，相应的监控也不会生效。

实际值采集方式

所使用的编码器类型必须通过以下机床数据确定：

MD30240 $MA_ENC_TYPE（实际值采集方式（位置实际值））

模拟轴

出于测试的目的，可对轴的转速控制回路进行模拟。

与真实轴相同，模拟轴运行时同样会出现跟随误差。

模拟轴是通过将以下两个机床数据置 “0” 来定义的：

MD30130 $MA_CTRLOUT_TYPE[n]（设定值输出方式）

MD30240 $MA_ENC_TYPE[n]（实际值采集方式）

载入缺省机床数据后，这些轴就成了模拟轴。

通过回参考点运行可将设定值/实际值设为回参考点值。

通过机床数据：

MD30350 $MA_SIMU_AX_VDI_OUTPUT（模拟轴上的轴信号输出）

可以确定模拟期间轴专用的接口信号是否输出到 PLC 上。

实际值补偿

如果希望 NC 对所选位置控制编码器进行的实际值补偿不影响该轴上定义的其他编码器的实际值，则需要通过以下机床数据将该编码器定义为“独立”编码器：

MD30242 $MA_ENC_IS_INDEPENDENT

实际值补偿包括：

- 模数处理
- 回参考点运行
- 测量系统调整
- PRESET（预设）

8.4.2 设定值和编码器分配

设定值分配

机床轴的设定值分配与以下机床数据有关。

MD30100		$MA_CTRLOUT_SEGMENT_NR
设定值分配：总线段		
系统	值	含义
840D sl	5	PROFIBUS-DP / PROFINET（初始设置）

MD30110		$MA_CTRLOUT_MODULE_NR
设定值分配：驱动编号/模块编号		
系统	值	含义
840D sl	x	通过驱动号从 MD13050 $MN_DRIVE_LOGIC_ADDRESS[n] 中分配驱动的逻辑 I/O 地址。 驱动号 (x) 是由 MD13050 的索引 (n) 得出的： x = n + 1 **提示：** 如果是模拟驱动 (MD30130 $MA_CTRLOUT_TYPE = 0)，该机床数据没有任何意义。

MD30120		$MA_CTRLOUT_NR
设定值分配：驱动模块/模块上的设定值输出		
系统	值	含义
840D sl	1	带有 PROFIdrive 协议的 PROFIBUS / PROFINET 上的模块化驱动（初始设置）

MD30130		$MA_CTRLOUT_TYPE
设定值输出方式		
系统	值	含义
840D sl	0	模拟（无驱动运行）
	1	设定值输出有效

编码器分配

在 PROFIdrive 报文中传送的编码器信息和对应的机床轴编码器输入通过以下机床数据设定：

MD30210		$MA_ENC_SEGMENT_NR[n]
实际值分配总线段		
系统	值	含义
840D sl	5	PROFIBUS-DP / PROFINET

MD30220		$MA_ENC_MODULE_NR[n]
实际值分配：驱动号/测量回路号		
系统	值	含义
840D sl	x	通过驱动号从 MD13050 $MN_DRIVE_LOGIC_ADDRESS[n] 中分配驱动的逻辑 I/O 地址。 驱动号 (x) 是由 MD13050 的索引 (n) 得出的： x = n + 1

MD30230		$MA_ENC_INPUT_NR[n]
实际值分配：驱动模块上的输入端/测量回路		
系统	值	含义
840D sl	x	PROFIdrive 报文内部的编码器接口编号 示例 PROFIdrive 报文 103 x = 1 → 第 1 编码器接口（G1_ZSW、G1_XIST1、G1_XIST2） x = 2 → 第 2 编码器接口（G2_ZSW、G2_XIST1、G2_XIST2） PROFIdrive 报文 118 x = 1 → 第 1 编码器接口（G2_ZSW、G2_XIST1、G2_XIST2） x = 2 → 第 2 编码器接口（G3_ZSW、G3_XIST1、G3_XIST2） **提示：** 就 SINAMICS S120 而言： - 编码器 1（G1_...）：电机编码器 - 编码器 2（G2_...）：直接测量系统 - 编码器 3（G3_...）：附加增量系统

MD30240		$MA_ENC_TYPE[n]
检测实际值的编码器类型（位置实际值）		
系统	值	含义
840D sl	0	模拟（无编码器运行）
	1	增量编码器
	4	绝对值编码器
提示： 与 PROFIdrive 参数 p979 相符		

MD30242		$MA_ENC_IS_INDEPENDENT[n, 轴]
编码器是独立的		
系统	值	含义
840D sl	0	编码器不是独立的。
	1	编码器是独立的。 如果对所选位置控制编码器进行的实际值补偿不影响该轴上定义的第二编码器的实际值，则需要通过以下机床数据将该编码器定义为“独立”编码器： 实际值补偿包括： ● - 模数处理 ● - 回参考点运行 ● - 测量系统调整 ● - PRESET（预设） 示例：一根轴、2 个编码器、第 2 个编码器是独立的 MD30200 $MA_NUM_ENCS[AX1] = 2 MD30242 $MA_ENC_IS_INDEPENDENT[0, AX1] = 0 MD30242 $MA_ENC_IS_INDEPENDENT[1, AX1] = 1 选择位置测量系统 1 / 2：DB31.DBX1.5 / 1.6 如果为位置闭环控制选择了编码器 1，由于编码器 2 是独立的，那么只在该编码器上执行实际值补偿。 如果为位置闭环控制选择了编码器 2，由于编码器 1 不是独立的，那么两个编码器上都会执行实际值补偿。 即：机床数据只对机床轴上的被动编码器起作用。
	2	被动编码器不是独立的。 编码器实际值是通过主动编码器修改的。与 MD35102 $MA_REFP_SYNC_ENCS = 1 组合后，在进行回参考点运行时，被动编码器会自动根据主动编码器调整，但不回参考点。 在回参考点模式 MD34200 $MA_ENC_REFP_MODE = 3（距离编码的参考标记）中，在驶过零脉冲距离后，被动编码器会自动在下一个运行动作时回参考点。这与当前的运行方式设置无关。
	3	编码器是独立的。 在模数回转轴上，模数实际值补偿也是在被动编码器中执行的。

说明

机床数据下标 [n]

用于编码器分配的机床数据下标 [n] 有以下含义：

- n = 0:分配给机床轴的第一个编码器
- n = 1:分配给机床轴的第二个编码器

该分配是通过以下机床数据进行的：

- MD30220 $MA_ENC_MODULE_NR[n]
- MD30230 $MA_ENC_INPUT_NR[n]

文档

CNC 调试手册：NC、PLC、驱动”；
章节：“NC 和驱动之间的通讯” > “驱动：轴分配”

8.4.3 调整电机/负载齿轮比

齿轮箱类型

以下齿轮箱类型可用于调整机械齿轮比：

齿轮箱类型	激活	调整	安装位置
“电机-负载”齿轮箱	参数组	固定配置	齿轮箱盒
“编码器-测量”齿轮箱	上电	随编码器变化	编码器侧的
“负载-中间”齿轮箱	热启动	随负载变化	刀具侧的

齿轮箱/编码器的局部位置

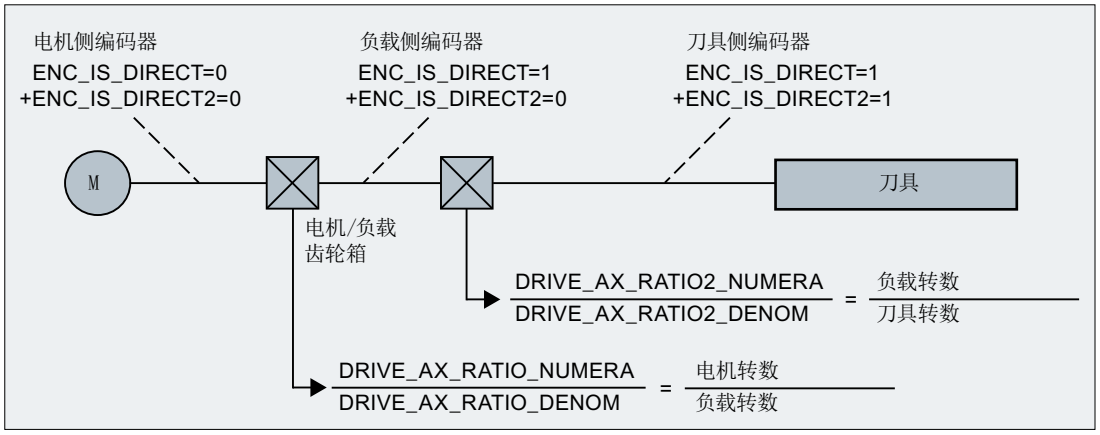


图 8-2　　齿轮箱类型和编码器安装位置

“电机-负载”齿轮箱

SINUMERIK 支持的“电机-负载”齿轮箱是通过以下机床数据配置的：

- MD31060 $MA_DRIVE_AX_RATIO_NUMERA（负载齿轮比分子）
- MD31050 $MA_DRIVE_AX_RATIO_DENOM（负载齿轮比分母）

齿轮比是由两个机床数据的分子与分母之比得出的。控制系统会通过相应的参数组自动将位置控制器与各个齿轮比同步。

由于齿轮箱不必始终自动进行换档且有多种方式进行换档，因此，位置控制并不总是通过参数组整合在一起。

说明

有关齿轮箱换档时参数组的更多信息请参考章节“S1：主轴 (页 1401)”。

中间齿轮比

控制系统还支持可设定的负载中间齿轮比：

- MD31066 $MA_DRIVE_AX_RATIO2_NUMERA（中间齿轮比分子）
- MD31064 $MA_DRIVE_AX_RATIO2_DENOM（中间齿轮比分母）

动力刀具通常都有它们“自己的”中间齿轮比。可通过增加一个和“电机-负载”齿轮箱相乘的中间齿轮比来设置此类变速装置。

⚠ 小心

切换时的不同传动比

与“电机-负载”齿轮箱不同的是，中间齿轮比中没有任何参数组，因此，也就无法控制到零件程序或 PLC（NC/PLC 接口）的同步切换。它可以避免换档时的零件加工。如何协调和激活修改后的机床数据与相应的换档之间的同步，是用户的职责。在轴运行期间进行换档时，由于定标系数会发生跃变，因此，**无法**避免执行补偿过程。NC 不会监控这些补偿过程是否符合最大加速度的要求。

直接连接在刀具上的编码器

中间齿轮比还可以采用另一种用于“刀具侧编码器”的连接方式，这可通过配置以下机床数据实现：

MD31044 $MA_ENC_IS_DIRECT2

非直接连接在刀具上的编码器

在对处于位置闭环控制模式中的中间齿轮比进行换档时，须注意以下前提条件：此种情况下，NC 会将待切换的齿轮比一同考虑到编码器信息的重新定标中。

此时，处于定位模式中的进给轴/主轴可以：

- **只有在静止状态下**齿轮箱换档才不会引起跃变。
  由于齿轮箱换档期间机械位置几乎或完全不会发生变化，为此，刀具侧的位置在换档前和换档后的变化与齿轮比发生变化时的情况是相同的。
  建议：
  为避免报警 21612“运行期间复位了伺服使能”，应只在“静止状态”下进行换档，且仍允许并建议在换档前或换档期间将进给轴或主轴切换至转速控制模式或跟踪运行模式。

前提条件

如果用于位置控制的**编码器直接连接在了刀具**上，那么电机-负载齿轮箱或中间齿轮比的换档只会影响 NC 和驱动器之间的转速接口上的物理量。此时，系统内部的参数组不会自动切换。

参考点和机床基准

⚠ 小心
机床基准丢失 控制系统无法识别出所有可能导致机床位置参考基准的状态，因此，出现此类情况时，调试人员或用户有责任主动地进行回参考点或执行零脉冲同步。

齿轮箱换档会对编码器定标产生影响，因此失去了参考点基准或机床位置基准。在此类情况下，控制系统有时会取消状态“已回参考点/已同步”。

如果机床基准丢失，必须通过校准或回参考点过程恢复已经丢失的机床基准。

参见

R1：回参考点 (页 1349)

8.4.4 转速设定值输出

进给轴的调节方向和运行方向

开始加工前须确定进给轴的运行方向。

调节方向

调试位置控制前，须对驱动的转速控制器和电流控制器进行调试和优化。

运行方向

通过机床数据：

MD32100 $MA_AX_MOTION_DIR（运行方向）

可以切换轴的运行方向，

且不会影响位置控制的调节方向。

转速设定值调整

SINUMERIK 840D sl

在调整转速设定值时，NC 需要了解哪些转速设定值对应驱动中的哪些电机转速，以便设置轴控制和监控功能。

PROFIBUS-DP 驱动上还可以选择手动调整转速设定值。

- **手动调整**
  在机床数据：
  MD32250 $MA_RATED_OUTVAL
  输入一个不为零的值。

说明

速度调整和最大转速设定值

由于本身已经具备了自动转速设定值调整功能，SINUMERIK 840D sl 上无需再进行速度调整！

最大转速设定值

SINUMERIK 840D sl 上的最大转速设定值是按百分比的形式表示的。100% 表示最大转速设定值或 PROFIdrive 驱动器上的最大转速（驱动中制造商专用的设置参数，如 SINAMICS 中的 p1082）。

主轴转速是在 SINUMERIK 840D sl 的 NC 中输出的。

在控制系统中包含了 5 个齿轮档的数据。

齿轮档通过该档位的最小转速和最大转速以及用于自动换档的最小转速和最大转速定义。只有当新编程的转速设定值超出当前齿轮档时才会输出新的目标齿轮档。

通过机床数据：

MD36210 $MA_CTRLOUT_LIMIT[n]（最大转速设定值）

可按百分比形式限制转速设定值。

最高可达 200%。

超出限值时，系统会输出报警。

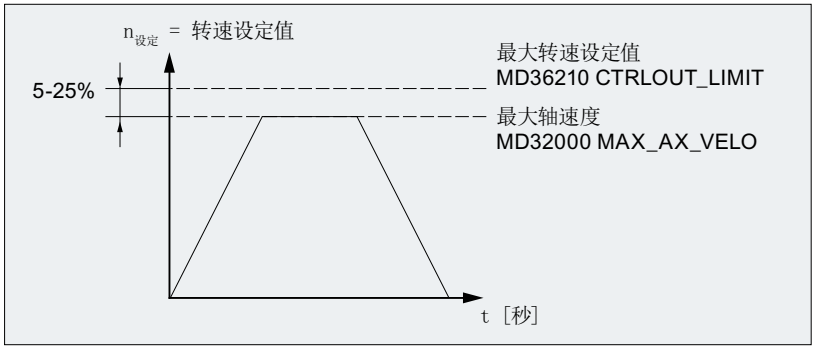


图 8-3　　最大转速设定值

但是为改善控制，轴在 80% 到 95% 的转速设定值时就应达到最大速度 (MD32000 $MA_MAX_AX_VELO)，而不是在 100% 的转速设定值时才达到。

如果轴在大约 80% 的转速设定值时达到了最大转速，则机床数据（缺省值 80%）：

MD32000 $MA_MAX_AX_VELO（最大轴速度）

生效。

8.4.5 实际值系统的机床数据

轴专用机床数据

为了参数设置实际值系统，须设置以下轴专用机床数据：

与编码器和参数组无关的机床数据：$MA_	含义
MD30200 NUM_ENCS	编码器数量
MD30300 IS_ROT_AX	回转轴/主轴
MD30310 ROT_IS_MODULO	回转轴/主轴的模数转换
MD30320 DISPLAY_IS_MODULO	回转轴和主轴的 360 度模数显示
MD30330 MODULO_RANGE	模数区域的大小
MD30340 MODULO_RANGE_START	模数区域的起始位置
MD31030 $MA_LEADSCREW_PITCH	丝杠螺距
MD31064 $MA_DRIVE_AX_RATIO2_DENOM	中间齿轮比分母
MD31066 $MA_DRIVE_AX_RATIO2_NUMERA	中间齿轮比分子
MD32000 $MA_MAX_AX_VELO	最大轴速度

编码器相关的机床数据：$MA_	含义
MD30210 ENC_SEGMENT_NR[n]	实际值分配：总线段的编号
MD30220 ENC_MODULE_NR[n]	实际值分配：驱动编号/测量回路编号
MD30230 ENC_INPUT_NR[n]	实际值分配：驱动模块/测量回路上的输入
MD30240 ENC_TYPE[n]	检测实际值的编码器类型（位置实际值）
MD30242 ENC_IS_INDEPENDENT[n]	编码器是独立的

编码器相关的机床数据：$MA_	含义
MD30244 ENC_MEAS_TYPE[n]	编码器测量类型
MD30250 ACT_POS_ABS[n]	内部编码器位置
MD30260 ABS_INC_RATIO[n]	绝对值编码器：绝对分辨率与增量分辨率的比例
MD30270 ENC_ABS_BUFFERING[n]	绝对值编码器：运行范围扩展
MD34090 $MA_REFP_MOVE_DIST_CORR[n]	参考点偏移
MD34320 $MA_ENC_INVERS[n]	长度测量系统方向相反
n:编码器下标，其中 n = 0、1、...（第 1 编码器、第 2 编码器、...）	

说明

可以在零件程序中通过 NEWCONF 指令，或者借助操作界面通过软键激活“激活机床数据”。

8.4.6 实际值精度

8.4.6.1 实际值分辨率的机床数据

根据测量系统（编码器）所安装于的位置，须考虑以下测量系统类型：

- 负载侧编码器：直接测量系统（DM）
- 电机侧编码器：间接测量系统（IM）

根据轴类型（线性轴/回转轴）参数设置实际值分辨率

基于以下机床数据计算控制系统的实际值分辨率。

用于计算实际值分辨率的机床数据	线性轴	线性轴		回转轴	
	光栅尺/直接测量系统	间接测量系统	直接测量系统：机床/刀具	间接测量系统	直接测量系统：机床/刀具
MD30300 $MA_IS_ROT_AX	0	0	0	1	1
MD31000 $MA_ENC_IS_LINEAR[n]	1	0	0	0	0
MD31010 $MA_ENC_GRID_POINT_DIST[n]	分度	-	-	-	-
MD34320 $MA_ENC_INVERS[n]	1)	-	-	-	-
MD31020 $MA_ENC_RESOL[n]	-	增量/转	增量/转	增量/转	增量/转
MD31025 $MA_ENC_PULSE_MULT[n]	编码器倍频				
MD31030 $MA_LEADSCREW_PITCH	-	毫米/转	毫米/转	-	-
MD31040 $MA_ENC_IS_DIRECT[n]	- / 1	0	1	0	1
MD31044 $MA_ENC_IS_DIRECT2[n]	- / 1	0	1	0	1
MD31050 $MA_DRIVE_AX_RATIO_DENOM[n]	-	负载转数	- / 1	负载转数	2)
MD31060 $MA_DRIVE_AX_RATIO_NUMERA[n]	-	电机转数，如果进给齿轮箱存在	- / 1	电机转数	2)
MD31070 $MA_DRIVE_ENC_RATIO_DENOM[n]	-	编码器转数	编码器转数	编码器转数	编码器转数
MD31080 $MA_DRIVE_ENC_RATIO_NUMERA[n]	-	电机侧编码器 3)	电机转数	电机转数	负载转数

- 不相关

1) 在有间距编码的测量系统中

2) 尽管在匹配编码器时并不需要这些机床数据。 但在计算设定值时要求这些机床数据必须正确输入！ 否则不会得到正确的 K_V 系数。 在机床数据 MD31050 $MA_DRIVE_AX_RATIO_DENOM 中输入负载转数，在机床数据 MD31060 $MA_DRIVE_AX_RATIO_NUMERA 中输入电机转数。
3) 电机侧编码器是一个内置编码器，因此，**没有**测量齿轮箱。
齿轮比始终为 1:1。

实际值分辨率的机床数据

实际值精度是由机床的结构、是否有齿轮箱及其齿轮比、线性轴上的丝杠螺距和所用编码器的分辨率得出的。为此，须在控制系统中设置以下机床数据：

与编码器和参数组无关的机床数据：$MA_	含义
MD30300 IS_ROT_AX	轴是回转轴/主轴
MD31010 ENC_GRID_POINT_DIST	光栅尺的栅距
MD31030 LEADSCREW_PITCH	丝杠螺距
MD31064 DRIVE_AX_RATIO2_DENOM	中间齿轮比分母
MD31066 DRIVE_AX_RATIO2_NUMERA	中间齿轮比分子

取决于编码器的机床数据：$MA_	含义
MD31000 ENC_IS_LINEAR[n]	测量系统是一个光栅尺
MD31020 ENC_RESOL[n]	每转的编码器线数
MD31025 ENC_PULSE_MULT[n]	编码器倍频（高分辨率）
MD31040 ENC_IS_DIRECT[n]	直接或间接测量系统
MD31044 ENC_IS_DIRECT2[n]	中间齿轮比上的编码器
MD31070 DRIVE_ENC_RATIO_DENOM[n]	测量齿轮比分母
MD31080 DRIVE_ENC_RATIO_NUMERA[n]	测量齿轮比分子
n：编码器下标，n = 0、1、...（第 1 个编码器、第 2 个编码器、...）	

取决于参数组的机床数据：$MA_	含义
MD31050 $MA_DRIVE_AX_RATIO_DENOM[m]	负载齿轮比分母
MD31060 $MA_DRIVE_AX_RATIO_NUMERA[m]	负载齿轮比分子
m：参数组下标，m = 0、1、...（第 1 个参数组、第 2 个参数组、...）	

参数设置计算精度

控制系统内部的计算精度与实际值精度之比是衡量由控制系统计算出的值转化精度的标准。

计算精度： 线性轴

$$\frac{\text{计算精度}}{\text{实际值分辨率}} = \frac{\text{内部增量/（mm）}}{\text{编码器增量/（mm）}}$$

计算精度： 回转轴

$$\frac{\text{计算精度}}{\text{实际值分辨率}} = \frac{\text{内部增量/（度）}}{\text{编码器增量/（度）}}$$

机床数据

通用机床数据：$MN_	含义
MD10200 INT_INCR_PER_MM	线性位置计算精度
MD10210 INT_INCR_PER_DEG	角位置计算精度

设置建议

在选择上述影响实际值精度的分量和设置时，应确保实际值精度大于设置的计算精度。

$$\frac{\text{计算精度}}{\text{实际值分辨率}} \leq 1$$

8.4.6.2 示例：带光栅尺的线性轴

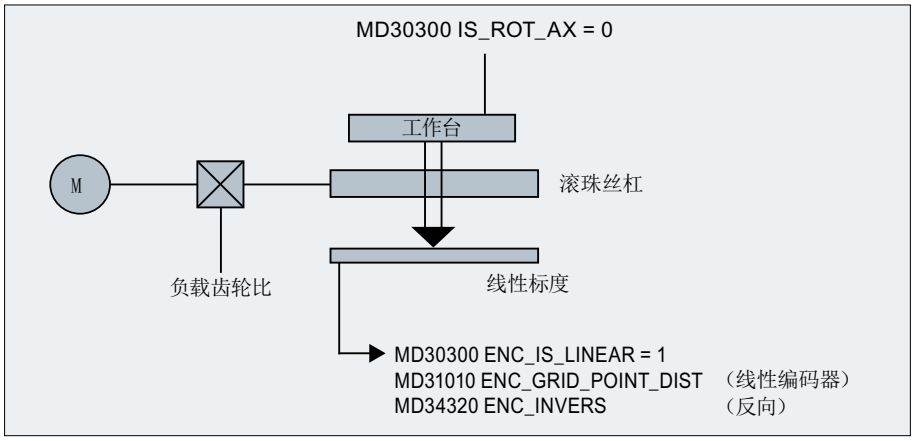


图 8-4 带光栅尺的线性轴

内部增量与编码器增量（每毫米）之比是按如下公式得出的：

$$\frac{\text{内部增量/mm}}{\text{编码器增量/mm}} = \frac{\text{ENC\_GRID\_POINT\_DIST [n]} * \text{INT\_INCR\_PER\_MM}}{\text{ENC\_PULSE\_MULT[n]}}$$

8.4.6.3 示例：电机上带旋转编码器的线性轴

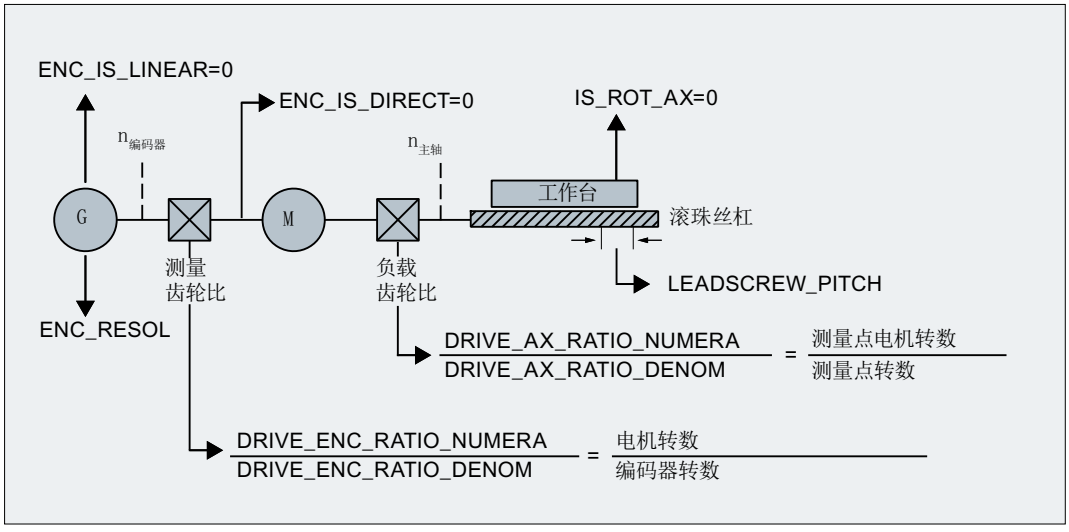


图 8-5 电机上带旋转编码器的线性轴

内部增量与编码器增量（每毫米）之比是按如下公式得出的：

$$\frac{\text{内部增量/mm}}{\text{编码器增量/mm}} = \frac{1}{\text{ENC\_RESOL [n] * ENC\_PULSE\_MULT[n]}} * \frac{\text{DRIVE\_ENC\_RATIO\_NUMERA [n]}}{\text{DRIVE\_ENC\_RATIO\_DENOM [n]}} * \frac{\text{DRIVE\_AX\_RATIO\_DENOM [n]}}{\text{DRIVE\_AX\_RATIO\_NUMERA [n]}} * \text{LEADSCREW\_PITCH} * \text{INT\_INCR\_PER\_MM}$$

示例

假设：

- 电机上的旋转编码器：2048 脉冲/转
- 内部脉冲倍频：2048
- 电机/丝杠齿轮比：5:1
- 丝杠螺距：10 毫米/转
- 计算精度：10000 增量/毫米

机床数据	值
MD30300 $MA_IS_ROT_AX	0
MD31000 $MA_ENC_IS_LINEAR[0]	0
MD31040 $MA_ENC_IS_DIRECT[0]	0
MD31020 $MA_ENC_RESOL[0]	2048
MD31025 $MA_ENC_PULSE_MULT	2048
MD31030 $MA_LEADSCREW_PITCH	10
MD31080 $MA_DRIVE_ENC_RATIO_NUMERA[0]	1
MD31070 $MA_DRIVE_ENC_RATIO_DENOM[0]	1
MD31060 $MA_DRIVE_AX_RATIO_NUMERA[0]	5
MD31050 $MA_DRIVE_AX_RATIO_DENOM[0]	1
MD10210 $MN_INT_INCR_PER_DEG	10000

$$\rightarrow \frac{\text{内部增量/mm}}{\text{编码器增量/mm}} = \frac{1}{2048 * 2048} * \frac{1}{1} * \frac{1}{5} * 10\ \text{mm}$$

$$*10000\ \text{增量/mm} = 0{,}004768$$

一个编码器增量等于 0.004768 个内部增量；换句话说：209.731543 个编码器增量等于一个内部增量。

8.4.6.4 示例： 机床上带旋转编码器的线性轴

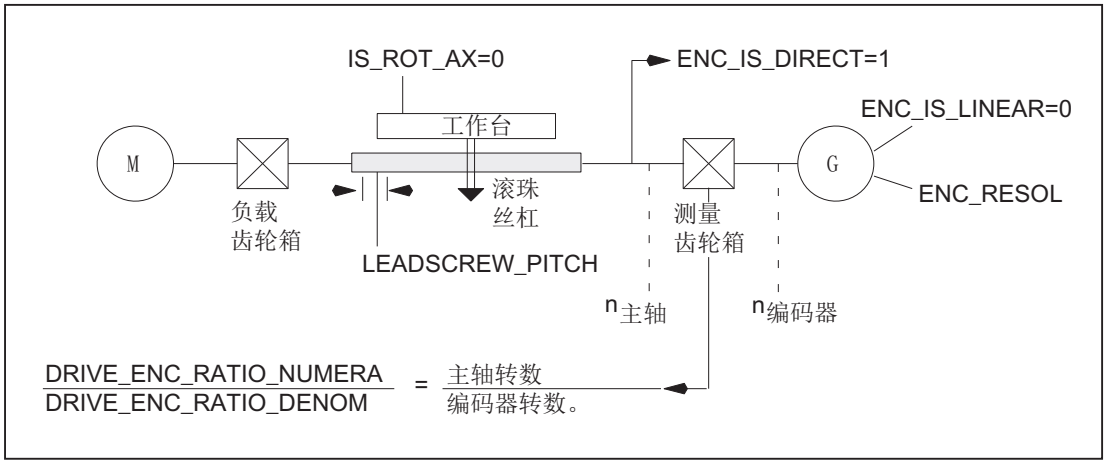


图 8-6 机床上带旋转编码器的线性轴

内部增量与编码器增量（每毫米）之比是按如下公式得出的：

$$\frac{\text{内部增量/mm}}{\text{编码器增量/mm}} = \frac{1}{\text{ENC\_RESOL [n]} * \text{ENC\_PULSE\_MULT[n]}} * \frac{\text{DRIVE\_ENC\_RATIO\_NUMERA [n]}}{\text{DRIVE\_ENC\_RATIO\_DENOM [n]}} * \text{LEADSCREW\_PITCH} * \text{INT\_INCR\_PER\_MM}$$

8.4.6.5 示例：电机上带旋转编码器的回转轴

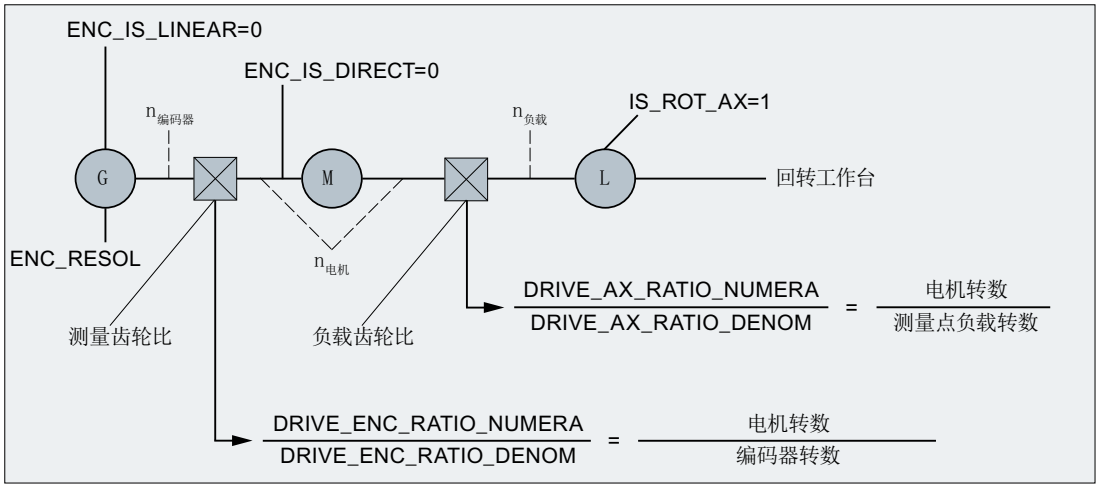


图 8-7 电机上带旋转编码器的回转轴

内部增量与编码器增量（每度）之比是按如下公式得出的：

$$\frac{\text{内部增量/度}}{\text{编码器增量/度}} = \frac{360\ \text{度}}{\text{ENC\_RESOL [n]} * \text{ENC\_PULSE\_MULT[n]}} * \frac{\text{DRIVE\_ENC\_RATIO\_NUMERA [n]}}{\text{DRIVE\_ENC\_RATIO\_DENOM [n]}} * \frac{\text{DRIVE\_AX\_RATIO\_DENOM [n]}}{\text{DRIVE\_AX\_RATIO\_NUMERA [n]}} * \text{INT\_INCR\_PER\_DEG}$$

示例

假设：

- 电机上的旋转编码器：2048 脉冲/转
- 内部脉冲倍频：2048
- 电机/回转轴齿轮比：5:1
- 计算精度： 1000 增量/度

机床数据	值
MD30300 $MA_IS_ROT_AX	1
MD31000 $MA_ENC_IS_LINEAR[0]	0
MD31040 $MA_ENC_IS_DIRECT[0]	0
MD31020 $MA_ENC_RESOL[0]	2048
MD31025 $MA_ENC_PULSE_MULT	2048
MD31080 $MA_DRIVE_ENC_RATIO_NUMERA[0]	1
MD31070 $MA_DRIVE_ENC_RATIO_DENOM[0]	1
MD31060 $MA_DRIVE_AX_RATIO_NUMERA[0]	5
MD31050 $MA_DRIVE_AX_RATIO_DENOM[0]	1
MD10210 $MN_INT_INCR_PER_DEG	1000

$$\rightarrow \frac{\text{内部增量/度}}{\text{编码器增量/度}} = \frac{360\text{ 度}}{2048 * 2048} * \frac{1}{1} * \frac{1}{5}$$

$$*1000\text{ 增量/度} = 0{,}017166$$

一个编码器增量等于 0.017166 个内部增量；换句话说：58.254689 个编码器增量等于一个内部增量。

8.4.6.6 示例： 机床上带旋转编码器的回转轴

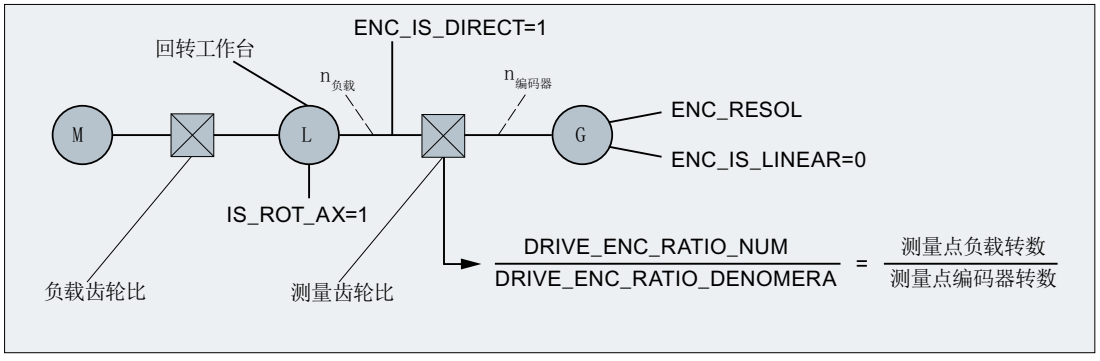


图 8-8 机床上带旋转编码器的回转轴

内部增量与编码器增量（每度）之比是按如下公式得出的：

$$\frac{\text{内部增量/度}}{\text{编码器增量/度}} = \frac{360\ \text{度}}{\text{ENC\_RESOL [n] * ENC\_PULSE\_MULT[n]}} * \frac{\text{DRIVE\_ENC\_RATIO\_NUMERA [n]}}{\text{DRIVE\_ENC\_RATIO\_DENOM [n]}} * \text{INT\_INCR\_PER\_DEG}$$

8.4.6.7 示例：刀具上带编码器的中间齿轮比

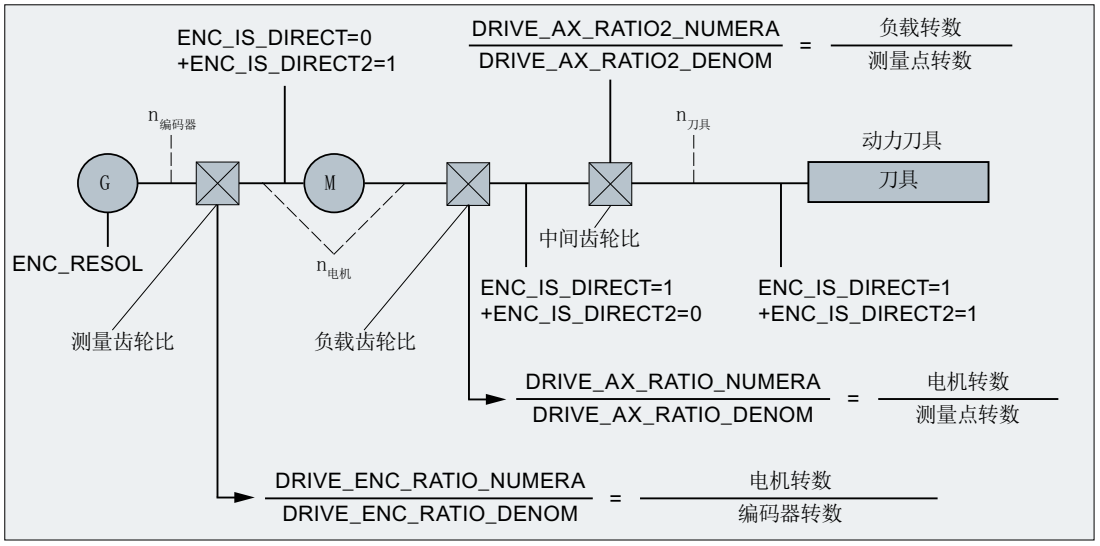


图 8-9 直接连接在动力刀具上的带编码器的中间齿轮比

内部增量与编码器增量（每度）之比是按如下公式得出的：

$$\frac{\text{内部增量/度}}{\text{编码器增量/度}} = \frac{360\ \text{度}}{\text{ENC\_RESOL [n] * ENC\_PULSE\_MULT[n]}} * \frac{\text{DRIVE\_ENC\_RATIO\_NUMERA [n]}}{\text{DRIVE\_ENC\_RATIO\_DENOM [n]}} * \text{INT\_INCR\_PER\_DEG}$$

8.5 闭环控制

8.5.1 简介

进给轴/主轴的位置闭环控制

轴的闭环控制包括驱动的电流和转速控制回路以及 NC 中的更高一级位置控制回路。

进给轴/主轴的位置闭环控制原则上采用如下结构：

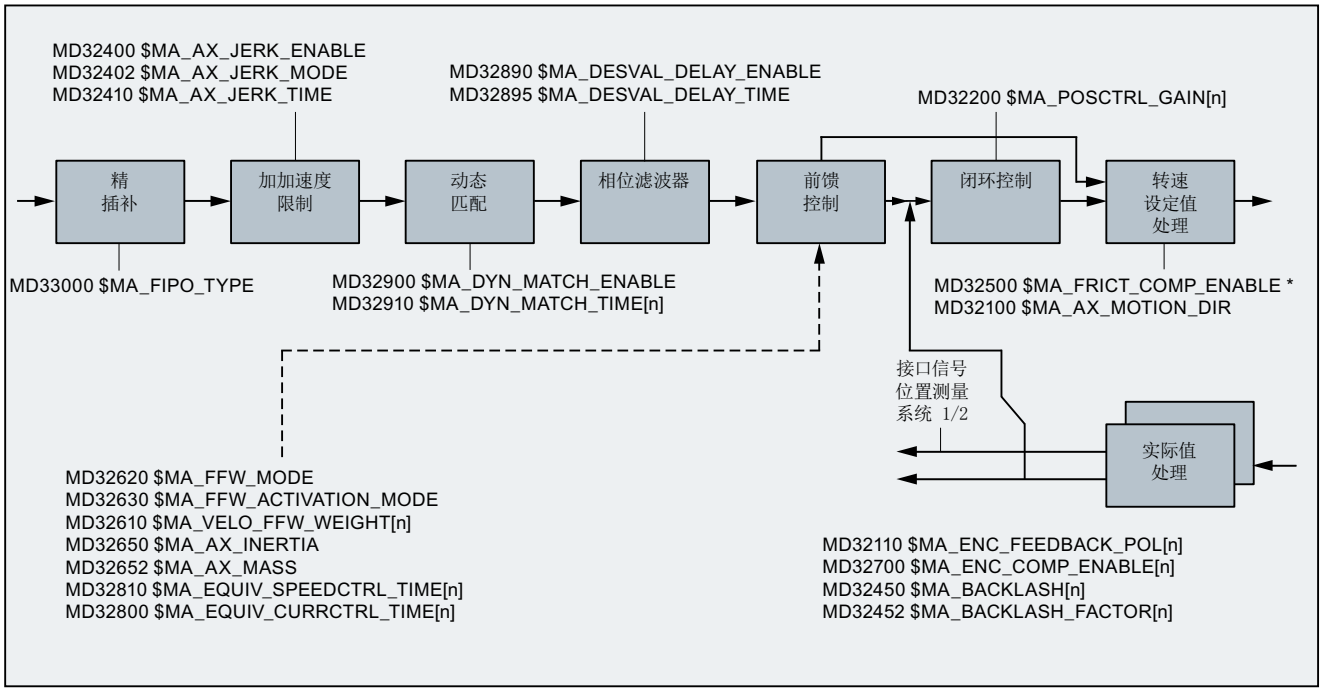


图 8-10　设定值处理和闭环控制的原理图

有关最大轴急动度的信息请参见章节“B2：加速度 (页 285)”。

关于前馈控制，背隙补偿、摩擦补偿和丝杠螺距误差补偿的说明，参见：

文档：
功能手册之扩展功能分册；补偿 (K3)

精插补

使用精插补 (FIPO) 时，可通过转速设定值的降低阶梯效应来进一步提高轮廓精度。可设置 3 种类型的精插补：

MD33000 $MA_FIPO_TYPE = <FIPO 模式>

<FIPO 模式>	含义
1	微分精插补，包含持续一个插补周期的平均值计算（平滑）
2	三次精插补
3	三次精插补，含前馈，可获得最高轮廓精度

K_V 系数

为确保连续路径运行中不会出现较大的轮廓偏差，需要一个高 K_V 系数：

MD32200 $MA_POSCTRL_GAIN[n]

K_V 系数太大会导致不稳、超调并可能使机床超出许可范围的负载。

允许的最大 K_V 系数取决于:

- 驱动的配置和动态性能
  （上升时间、加速和制动能力）
- 机床质量
  （弹性、振动阻尼）
- DSC 激活时的位置控制周期或转速控制周期

K_V 系数定义为：

$$K_v = \frac{\text{速度}}{\text{跟随误差}} \quad ; \quad \frac{[m/min]}{[mm]}$$

动态响应自适应功能

通过动态响应自适应功能能为可相互插补，但具有不同 K_V 系数的轴设置相同的跟随误差。将 K_V 系数降低至动态响应最弱的那根轴，就可以在不损失控制质量的同时进一步优化轮廓精度。

该功能是通过以下机床数据激活的：

MD32900 $MA_DYN_MATCH_ENABLE = 1（动态响应自适应功能）

动态响应自适应功能是通过输入一个新的等效时间常数来设定的，是由动态性最弱轴与待匹配轴的等效时间常数之差计算得出的：

MD32910 $MA_DYN_MATCH_TIME[n] = <等效时间常数的差值>

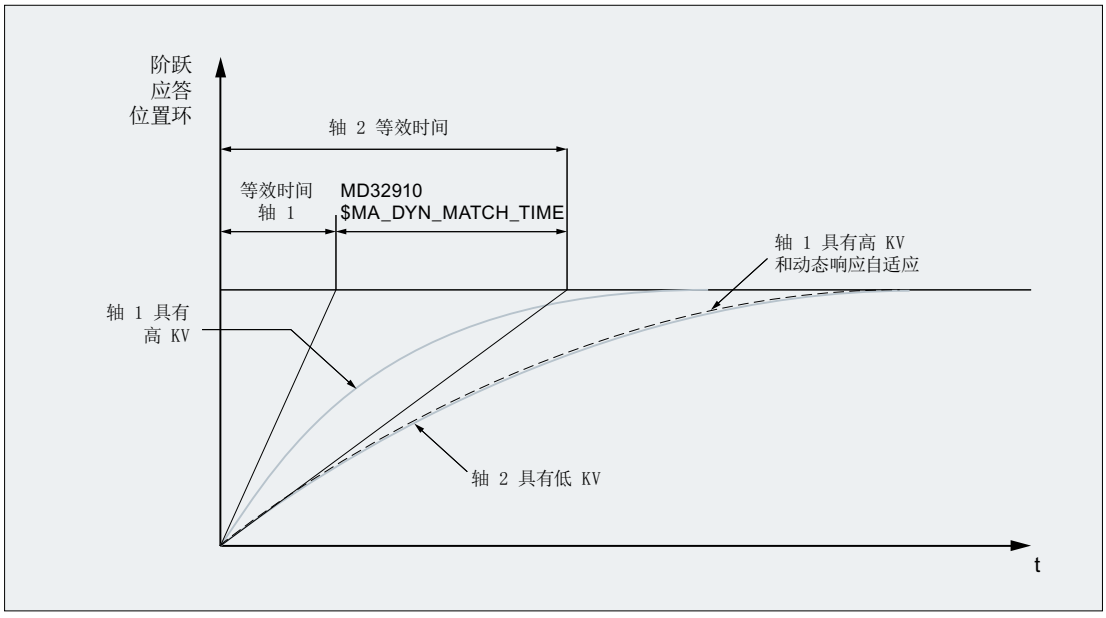


图 8-11　　动态响应自适应功能

无转速前馈控制的 3 轴动态响应自适应功能示例

位置控制回路的等效时间常数为：	
轴 1：	30 ms
轴 2：	20 ms
轴 3：	24 ms

等效时间常数为 30 ms 的轴 1 是动态响应最弱的那根轴。

从而得出下列新的等效时间常数：

轴 1：　　MD32910 $MA_DYN_MATCH_TIME = 0 毫秒
轴 2：　　MD32910 $MA_DYN_MATCH_TIME = 30 毫秒 - 20 毫秒 = 10 毫秒
轴 3：　　MD32910 $MA_DYN_MATCH_TIME = 30 毫秒 - 24 毫秒 = 6 毫秒

轴位置控制回路的等效时间常数的近似公式

根据前馈控制类型，轴位置控制回路的等效时间常数 T_{equiv} 的计算方式分为以下三种：

- 不带前馈控制：

$$T_{等效} \approx \frac{1}{\text{MD32200 POSCTRL\_GAIN [1/s]}}$$

- 带转速前馈控制：

$$T_{等效} \approx \text{MD32810 EQUIV\_SPEEDCTRL\_TIME}$$

- 转矩/转速前馈控制组合时：

$$T_{等效} \approx \text{MD32800 EQUIV\_CURRCTRL\_TIME}$$

说明

如果已经激活了一根几何轴的动态响应自适应功能，那么其他所有几何轴的动态响应都必须相同。

文档：

开机调试手册 CNC：NC、PLC、驱动

8.5.2 位置控制器的参数组

每根机床轴有六个参数组可用，以便根据运行期间机床条件的变化（比如：主轴换档）迅速调整位置控制，或以便根据攻丝时另一根轴的条件调整动态响应。

机床数据

一个参数组包括下列机床数据：

编号	名称 $MA_	含义
31050	DRIVE_AX_RATIO_DENOM	负载齿轮比分母
31060	DRIVE_AX_RATIO_NUMERA	负载齿轮比分子
32200	POSCTRL_GAIN	K_V 系数
32452	BACKLASH_FACTOR	间隙补偿
32610	VELO_FFW_WEIGHT	前馈控制系数
36012	STOP_LIMIT_FACTOR	粗准停/精准停和停机系数

编号	名称 $MA_	含义
32800	EQUIV_CURRCTRL_TIME	等效时间常数 前馈控制的电流控制回路
32810	EQUIV_SPEEDCTRL_TIME	等效时间常数 前馈控制的转速控制回路
32910	DYN_MATCH_TIME	动态响应自适应的时间常数
36200	AX_VELO_LIMIT	速度监控阈值

攻丝/螺纹切削

下列轴参数组适用于攻丝/螺纹切削：

- 对于**未**参与攻丝或螺纹切削的机床轴，始终是参数组 1（下标为 0）有效。 其他参数组不必考虑。
- 对于参与攻丝或螺纹切削的机床轴，参数组号等于当前主轴齿轮档的参数组有效。 所有等于主轴齿轮档的参数组必须进行参数设置。

操作界面上会显示当前生效的参数组：

“操作区切换” > “诊断” > “轴信息”

切换齿轮档时的参数组

给主轴的每个齿轮档分配一个合适的参数组。 齿轮档是通过以下 NC/PLC 接口信号选择的：

DB31, ... DBX16.0 - 16.2 = <实际齿轮档>

实际齿轮档	DB31, ... DBX16.0 - 16.2	参数组	
1. 齿轮档	000	2	（下标 1）
1. 齿轮档	001	2	（下标 1）
2. 齿轮档	010	3	（下标 2）
3. 齿轮档	011	4	（下标 3）
4. 齿轮档	100	5	（下标 4）
5. 齿轮档	101 110 111	6	（下标 5）

有关主轴齿轮档的更多信息请参见章节 “S1：主轴 (页 1401)”。

8.6 控制优化

8.6.1 位置控制器，位置设定值滤波器：对称滤波器

功能

前馈控制激活时，系统会在位置设定值到达真正的控制器前通过一个所谓的对称滤波器将其传送出去。这样就可以将转速设定值预先控制在 100 %，且在定位时不会出现振荡。

激活

通过修改以下轴专用机床数据来激活滤波器和选择前馈控制方式：

MD32620 $MA_FFW_MODE（前馈控制方式）

值	含义
3	转速前馈控制
4	组合式转矩/转速前馈控制

通过零件程序激活/取消

可通过零件程序中的指令 FFWON 和 FFWOF 来激活或取消所有轴的前馈控制。

如果无法通过 FFWON/FFWOF 指令激活或取消单个轴的前馈控制，则必须对以下机床数据中的设置进行调整：

MD32630 $MA_FFW_ACTIVATION_MODE（从程序激活前馈控制）

参数设置

首次调试时的推荐设置

在首次调试时或已事先载入了缺省值（调试开关上的开关位置 1 和 POWER ON 指令），则采用以下机床数据设置：

- MD32620 $MA_FFW_MODE = 3
- MD32610 $MA_VELO_FFW_WEIGHT（转速前馈控制系数）= 1

然后还须在以下机床数据中调整转速前馈控制的对称时间：

MD32810 $MA_EQUIV_SPEEDCTRL_TIME（用于前馈控制的转速环等效时间常数）

转速控制回路的等效时间常数设置 (MD32810)

建议在 AUTO 模式下来回移动轴，使轴到达目标位置，即通过伺服跟踪功能观察当前生效的测量系统的位置实际值。

设置的初始值是转速控制回路的时间常数，可从转速控制回路的参考频率响应中读取。在常见的带转速设定值平滑的 PI 控制器中，也可从驱动机床数据 p1414、p1415、p1416 和 p1421 中读取近似等效时间。

现在输入该初始值（如 1.5 ms）：

MD32810 $MA_EQUIV_SPEEDCTRL_TIME = 0.0015

然后来回移动轴并观察放大后朝目标位置运行时的位置实际值曲线。

手动精调时须遵循以下规定：

监控	措施	影响
超调	提高 MD32810	位置控制器响应缓慢。 ⇒ 超调的倾斜度降低。 问题：曲率上的轮廓误差（与所编程的轮廓偏差）增大。
逼近过慢	降低 MD32810	位置控制器响应过快。 ⇒ 曲率上的轮廓误差减小。 问题：超调的倾斜度增加。

说明

影响与改变位置控制器增益（MD32200 $MA_POSCTRL_GAIN）的效果类似。可在操作界面上的操作区域“诊断”>“轴信息”中基于计算出的 K_V 值进行观察。

精调 MD32810

因此，MD32810 的值应尽可能小，这样超调在定位时就可以设置限值。为此，在精调时通常只需对初始值进行微调，典型做法是向上或向下调整 0.25 毫秒。

例如：如果初始值为 1.5 毫秒，手动调整的最佳范围通常在 1.25 毫秒到 1.75 毫秒之间。

在装配了直接测量系统（负载编码器）和具备充足弹性的轴上，微米级的超调是可接受的。可借助用于动态响应自适应功能 (MD32910 $MA_DYN_MATCH_TIME) 和急动度 (MD32410 $MA_AX_JERK_TIME) 的位置设定值滤波器来降低超调，这会再次减缓轴的速度。

插补组中的相同轴数据

插补组中的所有轴在以下数据中的**设置应是相同的**：

- MD32200 $MA_POSCTRL_GAIN（通过 MD32910 调整）
- MD32620 $MA_FFW_MODE
- MD32610 $MA_VELO_FFW_WEIGHT
- MD32810 $MA_EQUIV_SPEEDCTRL_TIME 或 MD32800 $MA_EQUIV_CURRCTRL_TIME（取决于机械系统和驱动器）
- MD32400 $MA_AX_JERK_ENABLE
- MD32402 $MA_AX_JERK_MODE
- MD32410 $MA_AX_JERK_TIME

可通过操作界面上操作区域“诊断” > “轴信息”所显示的 K_V 值进行检查。

插补组中的不同轴数据

如果上述轴数据中包含了不同的值，则可以借助以下机床数据进行调整：

MD32910 $MA_DYN_MATCH_TIME（动态响应自适应的时间常数）

这样，界面上就会显示相同的 K_V 值。

通常，不同的 K_V 显示值通常表示：

- 一根或多根轴中的齿轮级系数不匹配。
- 前馈控制的设定数据不一致。

电流控制回路的等效时间常数设置 (MD32800)

用于转矩前馈控制的滤波器是通过以下机床数据激活的：

MD32620 $MA_FFW_MODE = 4

和转速前馈控制中的设置一样，电流控制回路的时间常数的设置 MD32800 $MA_EQUIV_CURRCTRL_TIME 也须遵循此类规定和建议。

刚性机床上的限制

经验表明，只要在刚性极高的机床上才需要进行此设置，而且要求相应的设置经验。通常来说，接入转矩会提高机床弹性，虽然保持了轮廓精度，但也产生了振动。

在该情况下，建议使用动态刚性控制 Dynamic Stiffness Control (DSC) 进行验证：

MD32640 $MA_STIFFNESS_CONTROL_ENABLE = 1

说明

使用 DSC 时需要满足以下条件：

- 不可以在 NC 中设置实际值取反 (MD32110 $MA_ENC_FEEDBACK_POL = -1) 。允许在驱动中设置实际值取反（SINAMICS 参数 p0410）。
- 必须激活所有插补相关的（轨迹运动或耦合）轴的该项功能（MD32640 = 1）。
- 如果轴编程为模拟轴（MD30130 $MA_CTRLOUT_TYPE = 0, MD30240 $MA_ENC_TYPE = 0），则无法使用 DSC。在此情况下，必须取消激活所有插补相关的轴的该项功能（MD32640 = 0）。

⚠ 小心

特殊故障状态

当为不存在的测量系统设置 p1192 并且 DSC 激活时，就会出现特殊故障状态！

维修时如要取消附加测量系统并且 DSC 激活，则应确保对 p1192 也进行相应的调整。

POWER ON、RESET、REPOS 等指令时的控制特性

在执行 POWER ON 和 RESET 以及“设置机床数据生效”指令时，系统会重新读入前馈控制的设定数据（参见各个机床数据的相关说明）。

运行方式切换、程序段搜索和 REPOS 对前馈控制无效。

8.6.2 位置控制器，位置设定值滤波器：急动度滤波器

功能

在某些应用中，如对任意形状的表面进行铣削时，借助急动度滤波器对机床轴的位置设定值变化进行平滑可能会非常有效，以减轻机床上的机械振动，从而获得更高的工件表面质量。必须在不影响轮廓精度的条件下尽可能的加强位置设定值滤波的效果。此外，应尽可能的达到“对称”的平滑效果。也就是说，如果向前或向后运行同一个轮廓，经过滤波器平滑后的轮廓曲线在两个方向上应尽可能的相似。

可通过“轴信息”窗口中显示的有效 K_V 系数来观察滤波器的效果。由于滤波效果会对位置设定值造成轻微的影响，一定程度上会降低轨迹精度。因此，随着滤波时间的增加，系统上显示的有效 K_V 系数也会逐渐变小。

说明

急动度滤波器在任何振幅设置下都会生成一个相应的相移。只有与相位滤波器组合使用（参见“位置控制器，位置设定值滤波器：相位滤波器 (页 418)”）才可实现对轴动态特性的精确设置。

激活

轴设定值急动度滤波器功能必须通过以下机床数据激活：

MD32400 $MA_AX_JERK_ENABLE[<轴>] = "TRUE"

设置

最大轴急动度的滤波器类型

最大轴急动度通常会使用“均值滤波器”：

MD32402 $MA_AX_JERK_MODE[<轴>] = **2**

说明

如果之前没有激活过滤波器类型 MD32402 $MA_AX_JERK_MODE = 2，则必须触发一次“上电”指令。其他情况下，按下机床控制面板上的“设置机床数据生效”或“复位”按键就可以了。

轴急动度滤波器的时间常数

轴急动度滤波器的时间常数（单位：秒）是在以下机床数据中设置的：

MD32410 $MA_AX_JERK_TIME[<轴>]

精调

急动度滤波器的精调方式如下：

1. 评估轴的运行特性（如借助伺服跟踪功能在定位过程中评估）
2. 在机床数据 MD32410 $MA_AX_JERK_TIME 中修改滤波时间。
3. 可借助机床控制面板上的“设置机床数据生效”或“复位”按键来激活所修改的时间。

禁用

取消急动度滤波器：

1. 禁用滤波器计算：
   MD32400 $MA_AX_JERK_ENABLE = 0
2. 可借助机床控制面板上的“设置机床数据生效”或“复位”按键来激活禁用指令。

前提条件

滤波时间

只有当时间常数 (MD32410) 大于位置控制周期时，急动度滤波器才会生效。

滤波效果

- “轴信息”窗口中显示的计算出的 K_V 系数有可能低于根据滤波效果测出的值。
- 轨迹精度实际上高于显示的 K_V 系数值。
  因此，同等滤波时间下，在从 MD32400 = 1 改到 MD32400 = 2 时，尽管轨迹精度更高了，但显示的 K_V 系数会变小。

相互插补的轴

- 插补轴的设置必须一致。
- 如果这些轴有各自的最佳方案，在上述设置条件下必须为所有插补轴输入最长滤波时间。

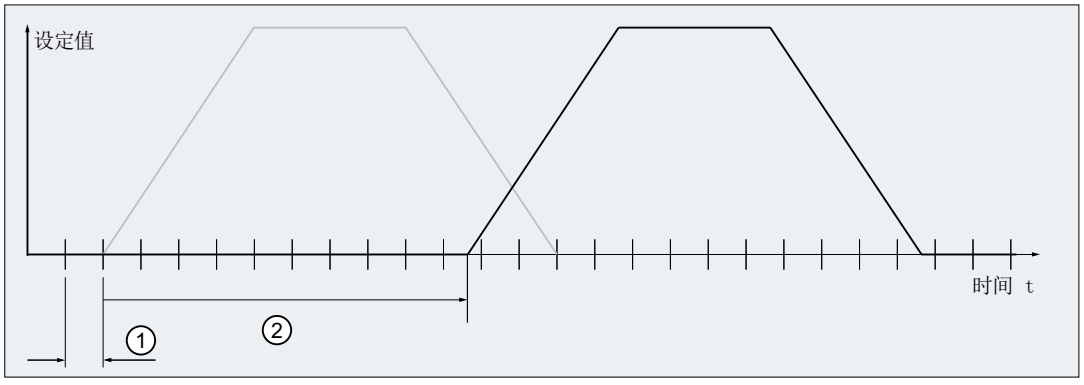
文档

有关插补级上的最大急动度的更多信息请参考章节 “单轴插补中的最大急动度 (SOFTA)（轴专用） (页 305)” 和 “进给轴/主轴专用机床数据 (页 335)”。

8.6.3 位置控制器，位置设定值滤波器：相位滤波器

功能

轴设定值相位滤波器（时滞/延时）可实现一个移相器的功能，借助其可调整设定值相位特性曲线。配合轴设定值急动滤波器（MD32402_$MA_AX_JERK_MODE[<轴>] = 2；参见“位置控制器，位置设定值滤波器：急动度滤波器 (页 415)”章节），可相互独立地调整振幅特性曲线和相位特性曲线，以适应运行编写的轮廓所涉及的多根轴中动态特性最弱的那根。



① 位置控制器周期：2 毫秒

② 轴设定值相位滤波器的时间常数：19.2 毫秒

图 8-12 轴设定值相位滤波器的作用

参数设置：时间常数

轴设定值相位滤波器的时间常数的设置范围如下：

(0 至 64) * 位置控制器周期（MD10061 $MN_POSCTRL_CYCLE_TIME）

必须以秒 [s] 为单位将时间常数值输入以下机床数据：

MD32895 $MA_DESVAL_DELAY_TIME[<轴>]

示例：位置控制器周期：2 毫秒 ⇒ 可设置的时间常数：0.0 至 0.128 秒

说明

设定值相位滤波器的时间常数会延缓轴的响应特性，例如在攻丝、回退或准停/程序段切换时。因此建议将时间常数设置地尽可能小。

限制为最大值

当设置的时间常数大于所允许的最大值（64 * 位置控制器周期）时，系统内部会将其限制为最大值。此时不会触发信息/报警。

参数设置：激活

轴设定值相位滤波器功能必须通过以下机床数据激活：

MD32890 $MA_DESVAL_DELAY_ENABLE[<轴>] = "TRUE"

示例

假设：位置控制器周期 = 2 毫秒

1. MD32890 $MA_DESVAL_DELAY_ENABLE[<轴>] = "FALSE"
   MD32895 $MA_DESVAL_DELAY_TIME[<轴>] = <时间常数>
   - 设定值相位滤波器：未生效
   - 时间常数：无含义
2. MD32890 $MA_DESVAL_DELAY_ENABLE[<轴>] = TRUE
   MD32895 $MA_DESVAL_DELAY_TIME[<轴>] = 0.002
   - 设定值相位滤波器：生效
   - 时间常数：2 毫秒 ⇒ 设定值输出延后一个位置控制器周期。
3. MD32890 $MA_DESVAL_DELAY_ENABLE[<轴>] = TRUE
   MD32895 $MA_DESVAL_DELAY_TIME[<轴>] = 0.256
   - 设定值相位滤波器：生效
   - 设置的时间常数：256 毫秒；
     可能的最大时间常数：64 * 2 毫秒 = 128 毫秒
     ⇒ 内部生效的时间常数：128 毫秒

前提条件

SINUMERIK Safety Integrated

设定值相位滤波器会延后轴设定值的输出，例如在回退运行中（Stop E 时 ESR）。但设定值相位滤波器不会影响取消进程，例如 SBH 的激活时间点。

8.6.4 位置控制器：位置差接通

前提条件

- 该功能只在带有两个编码器的轴上可用：
  MD30200 $MA_NUM_ENCS = 2
  其中一个编码器必须 配置为间接测量系统，另一个则须配置为直接测量系统：
  - 直接测量系统：
    MD31040 $MA_ENC_IS_DIRECT[1]=1
    用于实际值测量的编码器直接安装在机床上（负载编码器）。
  - 间接测量系统：
    MD31040 $MA_ENC_IS_DIRECT[0]=0
    用于实际值测量的编码器安装在电机上（电机编码器）。
- 作为 PROFIdrive 的标准报文类型，必须在驱动或 NC (MD13060 $MN_DRIVE_TELEGRAM_TYPE) 中配置为报文类型 136 或 138。

功能

位置差接通功能激活时，系统会测定一根轴的直接测量系统与间接测量系统之间的差值位置并根据加权系数的设置接通该差值位置，将其作为位置控制器周期中前馈控制的附加电流设定值。由此产生的振动阻尼有助于提升轴的稳定性和定位精度。

应用

该功能用于带有较高振动倾向的轴。

有效性

该功能仅在带有低固有频率（最大约 20 Hz）的轴上生效。

激活/设置

通过给定加权系数激活该功能：

MD32950 $MA_POSCTRL_DAMPING（转速控制回路衰减）= <值>

取值范围：-100% ... +100%

输入值 “100%” 表示：如果测出的两个测量系统之间的位置差达到以下值，系统则会根据 SINAMICS 参数 p2003 接通附加转矩：

- 直线电机：1 毫米
- 带旋转电机的线性轴：MD31030 $MA_LEADSCREW_PITCH（丝杠螺距）
- 回转轴/主轴：360 度

缺省设置为 0。此时，位置差接通功能无效。

说明

比如：可借助阶跃函数响应设置加权系数 MD32950 $MA_POSCTRL_DAMPING。
如果闭环控制达到稳定限值（振动倾向增强），则表明参数值设置的过大或该值符号错误。

8.6.5 带比例积分控制器的位置闭环控制

功能

标准情况下，位置控制器的核心是比例控制器。 针对某些特殊应用（如电子齿轮箱），可以激活一个积分分量。在设置适宜时，比例积分控制器会根据相应的机床数据在有限、可设置的时间范围内将设定位置和实际位置之间的误差修正为零。

⚠ 小心

比例积分控制器激活时，实际位置会出现超调

此时，用户必须确定此类影响是否是允许或可接受的。只有具备了控制方面的专业知识以及了解如何使用伺服跟踪功能进行测量才可以使用该功能。如果相关机床数据设置错误，机床可能会因不稳定而损坏。

步骤

1. 按照之前章节中描述的方法先将位置控制回路作为比例控制器进行优化。
2. 在测量期间提高以下机床数据的公差，以便确定比例积分控制器的效果：
   - MD36020 $MA_POSITIONING_TIME
   - MD36030 $MA_STANDSTILL_POS_TOL
   - MD36040 $MA_STANDSTILL_DELAY_TIME
   - MD36400 $MA_CONTOUR_TOL

3. 通过设置以下机床数据将位置控制回路作为比例积分控制器激活：
   MD32220 $MA_POSCTRL_INTEGR_ENABLE；值设为 1
   MD32210 $MA_POSCTRL_INTEGR_TIME；积分时间 [s]
   积分时间的有效性：
   - $T_n \rightarrow 0$：
     控制误差会被快速修正，但控制回路可能会变得不稳定。
   - $T_n \rightarrow \infty$：
     积分部件的有效性不为 0。控制器的特性等同于纯比例控制器。
4. 应在这两个极端情况之间寻求一个可达到平衡效果的 T_n。

   说明

   不可以选择一个非常接近稳定限值的 T_n，否则，机床会因不稳定而损坏。

   因此建议将 T_n 设置为不小于 1 秒的值。

   通过一个自动执行的程序来回移动轴，使轴到达目标位置，用伺服跟踪功能记录该过程。
5. 伺服跟踪功能中会显示：
   - 跟随误差
   - 实际速度
   - 实际位置
   - 设定位置
6. 一旦查找出 T_n 的最佳值，则应将以下机床数据中的公差值再次复位为所需值。
   - MD36020 $MA_POSITIONING_TIME
   - MD36030 $MA_STANDSTILL_POS_TOL
   - MD36040 $MA_STANDSTILL_DELAY_TIME
   - MD36400 $MA_CONTOUR_TOL

前提条件

DSC

使用积分器功能时，必须取消 DSC（Dynamic Stiffness Control，动态刚度控制）。

示例

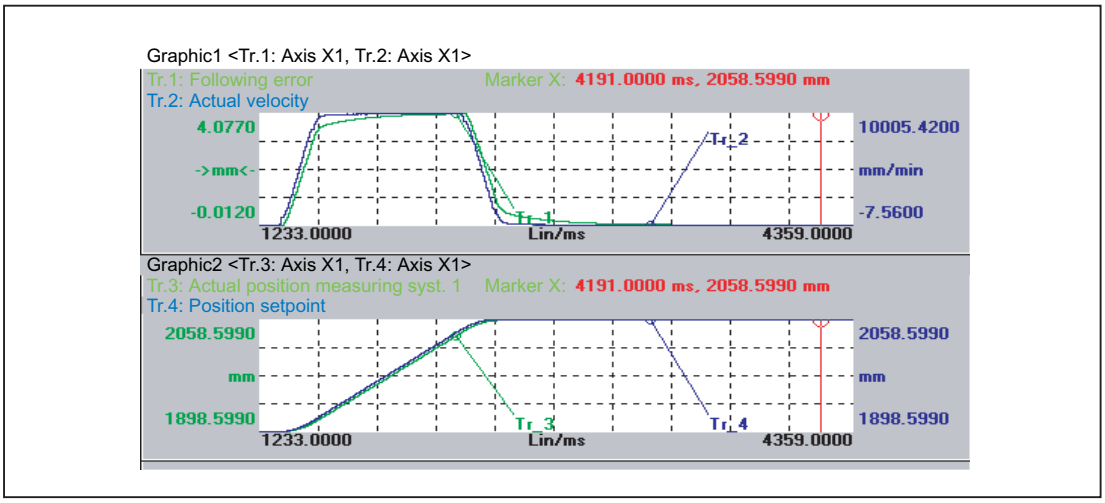
多次重复过程后的 K_R 和 T_n 的设置结果

机床数据设置：

- MD32220 $MA_POSCTRL_INTEGR_ENABLE = 1
- MD32210 $MA_POSCTRL_INTEGR_TIME = 0.003
- MD32200 $MA_POSCTRL_GAIN[1] = 5.0

选择参数组 0

伺服功能会分别记录跟随误差、实际速度、位置实际值和位置设定值。在 JOG 模式中运行时，还会记录下图中显示的各个数据的曲线。



8.7 数据表

8.7.1 机床数据

8.7.1.1 显示机床数据

编号	名称：$MM_	说明
9004	DISPLAY_RESOLUTION	显示精度
9010	SPIND_DISPLAY_RESOLUTION	主轴显示精度
9011	DISPLAY_RESOLUTION_INCH	英制单位制显示精度

8.7.1.2 NC 专用机床数据

编号	名称：$MN_	说明
10000	AXCONF_MACHAX_NAME_TAB	机床轴名称
10050	SYSCLOCK_CYCLE_TIME	系统基本周期

编号	名称：$MN_	说明
10070	IPO_SYSCLOCK_TIME_RATIO	插补周期系数
10060	POSCTRL_SYSCLOCK_TIME_RATIO	位置控制周期系数
10200	INT_INCR_PER_MM	线性位置计算精度
10210	INT_INCR_PER_DEG	角位置计算精度
10220	SCALING_USER_DEF_MASK	激活标准化系数
10230	SCALING_FACTORS_USER_DEF	物理量定标系数
10240	SCALING_SYSTEM_IS_METRIC	公制尺寸系统
10250	SCALING_VALUE_INCH	到英制单位制 的换算系数
10260	CONVERT_SCALING_SYSTEM	基本单位制切换有效
10270	POS_TAB_SCALING_SYSTEM	位置表的单位制
10290	CC_TDA_PARAM_UNIT	用于 CC 的刀具数据物理单位
10292	CC_TOA_PARAM_UNIT	用于 CC 的刀沿数据物理单位
13050	DRIVE_LOGIC_ADDRESS	逻辑驱动地址
13060	DRIVE_TELEGRAM_TYPE	用于 PROFIBUS-DP 的标准报文类型
13070	DRIVE_FUNCTION_MASK	使用的 DP 功能
13080	DRIVE_TYPE_DP	驱动方式 PROFIBUS-DP

8.7.1.3 通道专用机床数据

编号	名称：$MC_	说明
20150	GCODE_RESET_VALUES	G 功能组的初始设置

8.7.1.4 进给轴/主轴专用机床数据

编号	名称：$MA_	说明
30110	CTRLOUT_MODULE_NR	设定值分配：驱动号
30120	CTRLOUT_NR	设定值分配：驱动模块上的设定值输出
30130	CTRLOUT_TYPE	设定值输出方式
30200	NUM_ENCS	编码器数量
30220	ENC_MODULE_NR	实际值分配：驱动模块号

编号	名称：$MA_	说明
30230	ENC_INPUT_NR	实际值分配：驱动模块上的输入端
30240	ENC_TYPE	实际值采集类型（位置实际值）
30242	ENC_IS_INDEPENDENT	编码器是独立的
30300	IS_ROT_AX	回转轴
31000	ENC_IS_LINEAR	直接测量系统（线性标度）
31010	ENC_GRID_POINT_DIST	光栅尺的栅距
31020	ENC_RESOL	每转的编码器线数
31030	LEADSCREW_PITCH	丝杠螺距
31040	ENC_IS_DIRECT	编码器直接安装在机床上
31044	ENC_IS_DIRECT2	中间齿轮比上的编码器
31050	DRIVE_AX_RATIO_DENOM	负载齿轮比分母
31060	DRIVE_AX_RATIO_NUMERA	负载齿轮比分子
31064	DRIVE_AX_RATIO2_DENOM	中间齿轮比分母
31066	DRIVE_AX_RATIO2_NUMERA	中间齿轮比分子
31070	DRIVE_ENC_RATIO_DENOM	测量齿轮比分母
31080	DRIVE_ENC_RATIO_NUMERA	测量齿轮比分子
31090	JOG_INCR_WEIGHT	手动运行/手轮运行中的步长
31200	SCALING_FACTOR_G70_G71	G70/G71 指令激活时的换算系数
32000	MAX_AX_VELO	最大轴速度
32100	AX_MOTION_DIR	运行方向
32110	ENC_FEEDBACK_POL	实际值符号（调节方向）
32200	POSCTRL_GAIN	K_V 系数
32210	POSCTRL_INTEGR_TIME	位置控制器的积分时间
32220	POSCTRL_INTEGR_ENABLE	激活位置控制器的积分部件
32250	RATED_OUTVAL	额定输出电压
32260	RATED_VELO	额定电机速度
32450	BACKLASH	反向间隙
32500	FRICT_COMP_ENABLE	摩擦补偿生效
32610	VELO_FFW_WEIGHT	转速前馈控制系数
32620	FFW_MODE	前馈控制类型

编号	名称：$MA_	说明
32630	FFW_ACTIVATION_MODE	从程序激活前馈控制
32650	AX_INERTIA	转矩前馈控制的转动惯量
32652	AX_MASS	用于转矩前馈控制的轴质量
32711	CEC_SCALING_SYSTEM_METRIC	垂度补偿的单位制
32800	EQUIV_CURRCTRL_TIME	电流环前馈控制的等效时间常数
32810	EQUIV_SPEEDCTRL_TIME	转速环前馈控制的等效时间常数
32890	DESVAL_DELAY_ENABLE	轴设定值相位滤波器
32895	DESVAL_DELAY_TIME	轴设定值相位滤波器的时间常数
32900	DYN_MATCH_ENABLE	动力匹配
32910	DYN_MATCH_TIME[n]	动态响应自适应的时间常数
32930	POSCTRL_OUT_FILTER_ENABLE	激活位置控制器输出端上的二阶滤波器
32950	POSCTRL_DAMPING	转速控制回路衰减
33000	FIPO_TYPE	精插补类型
34320	ENC_INVERS[n]	长度测量系统方向相反
35100	SPIND_VELO_LIMIT	最大主轴转速
36200	AX_VELO_LIMIT[n]	速度监控阈值
36210	CTRLOUT_LIMIT[n]	最大转速设定值
36400	AX_JERK_ENABLE	最大轴急动度
36410	AX_JERK_TIME	轴急动度滤波器的时间常数
36500	ENC_CHANGE_TOL	位置实际值切换时的最大公差
36510	ENC_DIFF_TOL	测量系统同步公差
36700	ENC_COMP_ENABLE[n]	插补补偿

8.7.2 信号

8.7.2.1 从 NC 发出的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
修改计数器，英制/公制单位制	DB10.DBB71	DB2700.DBB0015
英制单位制	DB10.DBX107.7	DB2700.DBX1.7

8.7.2.2 发送至进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
位置测量系统 1	DB31, ...DBX1.5	DB38xx.DBX1.5
位置测量系统 2	DB31, ...DBX1.6	DB38xx.DBX1.6

H2：输出到 PLC 的辅助功能

9

9.1 简要说明

9.1.1 功能

辅助功能可用于激活 NC 的系统功能和 PLC 用户功能。可在以下程序或动作中编程辅助功能：

- 零件程序
- 同步动作
- 用户循环

有关在同步动作中使用辅助功能输出的详细信息请参考：

文档：
功能手册之同步动作分册

预定义的辅助功能

预定义的辅助功能可激活系统功能。此外， 辅助功能还会输出到 NC/PLC 接口上。

预定义的辅助功能有：

类型	功能	示例	含义
M	附加功能	M30	程序结束
S	主轴功能	S100	主轴转速 100（如 rpm）
T	刀具号	T3	刀具号 3
D, DL	刀具补偿	D1	刀沿号 1
F	进给率	F1000	进给率 1000（如毫米/分）

用户自定义辅助功能

用户定义的辅助功能要么是经过扩展的预定义辅助功能，要么是用户专用的辅助功能。

预定义辅助功能的扩展

预定义辅助功能的扩展基于“地址扩展”参数。通过地址扩展定义辅助功能涉及的主轴的编号。例如，针对主主轴预定义了主轴功能 M3（主轴正转）。若为通道分配了第 2 根主轴，则需要定义相应的用户自定义辅助功能，以作为对预定义辅助功能的扩展。

类型	功能	示例	含义
M	附加功能	M2=3	第 2 根主轴：主轴正转
S	主轴功能	S2=100	第 2 根主轴：主轴转速 = 100（如 rpm）
T	刀具号	T2=3	

用户专用辅助功能

用户专用辅助功能不会激活系统功能。用户专用辅助功能只会被输出至 NC/PLC 接口。辅助功能的功能性须由机床制造商/用户在 PLC 用户程序中实现。

类型	功能	示例	含义
H[1)]	辅助功能	H2=5	用户专用的功能

1) 推荐

9.1.2 辅助功能的定义

辅助功能是通过以下参数来定义的：

- **类型、地址扩展和值**
  这 3 个参数会输出到 NC/PLC 接口上。
- **输出特性**
  通过辅助功能特定的输出特性可以确定：辅助功能输出到 NC/PLC 接口上所需的时间以及相对于零件程序中编程的运动何时进行输出。
- **分组指定**
  一个辅助功能可分配给某个辅助功能组。每个辅助功能组的输出特性可单独确定。当没有为某辅助功能确定输出特性时，辅助功能组的输出特性有效。此外，在进行程序段搜索后，分组也会影响辅助功能的输出。

有关辅助功能输出到 NC/PLC 接口上的详细信息请参考章节 “P3：SINUMERIK 840D sl 的 PLC 基本程序 (页 945)”。

9.1.3 辅助功能一览

M 功能

M（附加功能）					
地址扩展		**值**			
取值范围	**含义**	**取值范围** [10]	**类型**	**含义**	**数量** [8]
0（固有的）	---	0 ... 99	INT	功能	5
取值范围	**含义**	**取值范围** [11]	**类型**	**含义**	**数量** [8]
1 ... 20	主轴号	1 ... 99	INT	功能	5
取值范围	**含义**	**取值范围** [12]	**类型**	**含义**	**数量** [8]
0 ... 99	任意	100 ... 2147483647	INT	功能	5

[8] 参见概览末尾处的“脚注含义”

[10] 对于 0 至 99 的取值范围地址扩展为 0。强制无地址扩展：M0，M1，M2，M17，M30

[11] M3，M4，M5，M19，M70：地址扩展为主轴编号，例如 M2=5 ⇒ 主轴 2 的主轴停止（M5）
在无地址扩展的情况下，该 M 功能作用于主主轴。

[12] 用户专用的 M 功能。

使用

同步控制机床功能和零件程序。

更多信息

- 下列 M 功能具有预定义的含义：M0，M1，M2，M17，M30，M3，M4，M5，M6，M19，M70，M40，M41，M42，M43，M44，M45。
- 针对 M 功能（M0 - M99）设有对应的用于显示有效性的动态 NC/PLC 接口信号。还可为自有 M 功能分配 64 个附加信号（参见章节“P3：SINUMERIK 840D sl 的 PLC 基本程序 (页 945)”）。
- 对于子程序，可通过以下机床数据来设置是否应将用于零件程序结束的 M 功能 M17、M2 或 M30 输出到 PLC 上：
  MD20800 $MC_SPF_END_TO_VDI（子程序结束输出给 PLC）
- 对于预定的 M 功能 M40 ... M45，对输出特性的重新定义受到限制。

- 预定义的辅助功能 M0、M1、M17、M30、M6、M4、M5 无法进行重新定义。
- M 功能专用的机床数据：
  - MD10800 $MN_EXTERN_CHAN_SYNC_M_NO_MIN
  - MD10802 $MN_EXTERN_CHAN_SYNC_M_NO_MAX
  - MD10804 $MN_EXTERN_M_NO_SET_INT
  - MD10806 $MN_EXTERN_M_NO_DISABLE_INT
  - MD10814 $MN_EXTERN_M_NO_MAC_CYCLE
  - MD10815 $MN_EXTERN_M_NO_MAC_CYCLE_NAME
  - MD20094 $MC_SPIND_RIGID_TAPPING_M_NR
  - MD20095 $MC_EXTERN_RIGID_TAPPING_M_NR
  - MD20096 $MC_T_M_ADDRESS_EXT_IS_SPINO
  - MD22200 $MC_AUXFU_M_SYNC_TYPE
  - MD22530 $MC_TOCARR_CHANGE_M_CODE
  - MD22532 $MC_GEOAX_CHANGE_M_CODE
  - MD22534 $MC_TRAFO_CHANGE_M_CODE
  - MD22560 $MC_TOOL_CHANGE_M_CODE

S 功能

S（主轴功能）					
地址扩展 [10]		**值**			
取值范围	**含义**	**取值范围**	**类型**	**含义**	**数量** [8]
0 ... 20	主轴编号 [5]	0 ... ± 3.4028 exp38 [3]	REAL	主轴转速	3

[3] [5] [8] 参见概览末尾处的“脚注含义”。

[10] 如果没有指定地址扩展，系统会对通道的主主轴进行编址。

使用

主轴转速。

更多信息

- Die S-Funktionen sind standardmäßig der 3.Hilfsfunktionsgruppe zugeordnet.
- 如果没有地址扩展，S 功能用于通道的主主轴。
- S 功能专用的机床数据：
  MD22210 $MC_AUXFU_S_SYNC_TYPE（S 功能的输出时间点）

H 功能

H 功能需要在 PLC 用户程序中实现。

H（辅助功能）[10]					
地址扩展		值			
取值范围	含义	取值范围	类型	含义	数量 [8]
0 ... 99	任意	- 2147483648 ... + 2147483647	INT	任意	3
		0 ... ± 3.4028 exp38 [2] [3] [4]	REAL		

[2] [3] [4] [8] 参见概览末尾处的“脚注含义”。

使用

用户专用的辅助功能。

更多信息

H 功能专用的机床数据：

- MD22110 $MC_AUXFU_H_TYPE_INT（H 辅助功能的类型为整数）
- MD22230 $MC_AUXFU_H_SYNC_TYPE（H 功能的输出时间点）

T 功能

刀具名称不输出至 NC/PLC 接口。

T（刀具编号）[1] [5] [6]					
地址扩展		值			
取值范围	含义	取值范围	类型	含义	数量 [8]
1 ... 12	主轴号（在刀具管理生效时）	0 ... 32000（在刀具管理生效时也可以为符号刀具名称）	INT	选择刀具	1

[1] [5] [6] [8] 参见概览末尾处的“脚注含义”。

使用

刀具选择。

更多信息

- 选择通过刀具编号或刀位编号来识别刀具（参见章节“W1：刀具补偿 (页 1593)”）。
  文档
  功能手册，刀具管理
- 用 T0 从刀架中取下刀具，但不换上新的刀具（初始设置）。
- T 功能专用的机床数据：
  MD22220 $MC_AUXFU_T_SYNC_TYPE（T 功能的输出时间点）

D 功能

用 D0 取消刀具补偿。预赋值为 D1。

D（刀具补偿）					
地址扩展		**值**			
取值范围	**含义**	**取值范围**	**类型**	**含义**	**数量** [8]
---	---	0 ... 9	INT	选择刀具补偿	1

[8] 参见概览末尾处的“脚注含义”。

使用

选择刀具补偿。

更多信息

- 初始设置：D1
- 可通过以下机床数据对换刀后的缺省刀沿进行设置：
  MD20270 $MC_CUTTING_EDGE_DEFAULT （未编程的刀沿初始设置）
- 取消刀具补偿：D0
- D 功能专用的机床数据：
  MD22250 $MC_AUXFU_D_SYNC_TYPE（D 功能的输出时间点）

DL 功能

通过 DL 选中的刀具总补偿针对的是生效的 D 号。

DL（刀具总补偿）					
地址扩展		值			
取值范围	含义	取值范围	类型	含义	数量 [8]
---	---	0 ... 6	INT	选择刀具总补偿	1

[8] 参见概览末尾处的“脚注含义”。

使用

选择与生效的刀具补偿相关的刀具总补偿。

更多信息

- 初始设置：DL = 0
- 无法通过同步动作将 DL 值输出到 PLC 上。
- DL 功能未生效时的刀具总补偿的初始设置：
  MD20272 $MC_SUMCORR_DEFAULT （未编程总补偿初始设置）
- 取消刀具总补偿：DL = 0
- DL 功能专用的机床数据：
  MD22252 $MC_AUXFU_DL_SYNC_TYPE（DL 功能的输出时间）

F 功能

F（轨迹进给率）					
地址扩展		值			
取值范围	含义	取值范围	类型	含义	数量 [8]
---	---	0.001 ... 999 999.999	REAL	轨迹进给率	6

[8] 参见概览末尾处的“脚注含义”。

使用

轨迹速度。

更多信息

F 功能专用的机床数据：

- MD22240 $MC_AUXFU_F_SYNC_TYP（F 功能的输出时间）

FA 功能

FA（轴专用进给率）					
地址扩展		**值**			
取值范围	**含义**	**取值范围**	**类型**	**含义**	**数量** [8]
1 - 31	轴编号	0.001 ... 999 999.999	REAL	轴进给	6

[8] 参见概览末尾处的“脚注含义”。

使用

轴专用速度

更多信息

F 功能专用的机床数据：

- MD22240 $MC_AUXFU_F_SYNC_TYP（F 功能的输出时间）

脚注的含义

[1] 当刀具管理激活时， 系统既不会将 T 修改信号，也不会将 T 字发送至通道专用 NC/PLC 接口。

[2] 用户可通过机床数据 MD22110 $MC_AUXFU_H_TYPE_INT 选择值的类型。

[3] 由于操作面板屏幕上的显示方式受限，REAL 型的值被限制在以下范围内：

–999 999 999.9999 至 999 999 999.9999

NC 会在内部使用完整的精度进行计算。

[4] 在设置以下机床数据时，NC 会对 REAL 值进行取整，然后将其输出到 PLC 上：

MD22110 $MC_AUXFU_H_TYPE_INT = 1（H 辅助功能的类型为整数）

PLC 用户程序须根据机床数据的设置来编译传递值。

[5] 当刀具管理激活时，可对地址扩展的含义进行参数设置。地址扩展 = 0 表示该值须由主主轴号替换，即与没有对地址扩展进行编程时的含义相同。

在进行程序段搜索时，收集的辅助功能 M19 “定位主轴”不会输出到 PLC 上。

[6] M6：地址扩展的取值范围：

- 无刀具管理：0 ... 99

- 带刀具管理：0 ... 最大主轴编号

0：由主主轴号或主刀架的值替换

7) 当刀具管理激活时，无论是否编程了地址扩展，辅助功能 M6 “换刀”在零件程序段中只能编程一次。

8) 每个零件程序段的最大辅助功能数。

9.2 预定义的辅助功能

功能

每个预定义的辅助功能都会分配一个系统功能且不可更改。如果已经在一个零件程序/循环中编程了一个预定义的辅助功能，该功能会通过 NC/PLC 接口输出到 PLC 上并在 NC 中执行相应的系统功能。

预定义辅助功能的定义

预定于辅助功能的参数在机床数据中定义，且可对其进行部分修改。所有属于一个辅助功能的机床数据都具有相同的下标 <n>。

- MD22040 $MC_AUXFU_PREDEF_GROUP[]（预定义辅助功能的分组指定）
- MD22050 $MC_AUXFU_PREDEF_TYPE[<n>]（预定义辅助功能的类型）
- MD22060 $MC_AUXFU_PREDEF_EXTENSION[<n>]（预定义辅助功能的地址扩展）
- MD22070 $MC_AUXFU_PREDEF_VALUE[<n>]（预定义辅助功能的数值）
- MD22080 $MC_AUXFU_PREDEF_SPEC[<n>]（预定义辅助功能的输出特性）

9.2.1 一览：预定义的辅助功能

下表中所列参数的含义：

参数	含义
下标 <n>	辅助功能参数的机床数据下标
类型	MD22050 $MC_AUXFU_PREDEF_TYPE[<n>]
地址扩展	MD22060 $MC_AUXFU_PREDEF_EXTENSION[<n>]
值	MD22070 $MC_AUXFU_PREDEF_VALUE[<n>]
组	MD22040 $MC_AUXFU_PREDEF_GROUP[<n>]

预定义的辅助功能

通用辅助功能：第 1 部分					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
停止	0	M	0	0	1
有条件停止	1	M	0	1	1
子程序结束	2	M	0	2	1
	3	M	0	17	1
	4	M	0	30	1
换刀	5	M	(0)	6 [1)]	(1)

主轴专用的辅助功能，主轴 1					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
主轴正转	6	M	1	3	(2)
主轴反转	7	M	1	4	(2)
主轴停止	8	M	1	5	(2)
主轴定位	9	M	1	19	(2)
进给轴模式	10	M	1	70 [2)]	(2)
自动换档	11	M	1	40	(4)
齿轮档 1	12	M	1	41	(4)
齿轮档 2	13	M	1	42	(4)
齿轮档 3	14	M	1	43	(4)
齿轮档 4	15	M	1	44	(4)
齿轮档 5	16	M	1	45	(4)
主轴转速	17	S	1	-1	(3)

通用辅助功能：第 2 部分					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
进给率	18	F	0	-1	(1)
刀沿选择	19	D	0	-1	(1)
DL	20	L	0	-1	(1)

通用辅助功能：第 2 部分					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
刀具选择	21	T	(0)	-1	(1)
停止（组合式）	22	M	0	-1 3)	1
有条件停止（组合式）	23	M	0	-1 4)	1
子程序结束	24	M	0	-1 5)	1
步冲	25	M	0	20 6)	(10)
步冲	26	M	0	23 6)	(10)
步冲	27	M	0	22 6)	(11)
步冲	28	M	0	25 6)	(11)
步冲	29	M	0	26 6)	(12)
步冲	30	M	0	122 6)	(11)
步冲	31	M	0	125 6)	(11)
步冲	32	M	0	27 6)	(12)

主轴专用的辅助功能，主轴 2					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
主轴正转	33	M	2	3	(72)
主轴反转	34	M	2	4	(72)
主轴停止	35	M	2	5	(72)
主轴定位	36	M	2	19	(72)
进给轴模式	37	M	2	70 2)	(72)
自动换档	38	M	2	40	(74)
齿轮档 1	39	M	2	41	(74)
齿轮档 2	40	M	2	42	(74)
齿轮档 3	41	M	2	43	(74)
齿轮档 4	42	M	2	44	(74)
齿轮档 5	43	M	2	45	(74)
主轴转速	44	S	2	-1	(73)

主轴专用的辅助功能，主轴 3					
系统功能	下标 <n>	类型	地址扩展	值	组
主轴正转	45	M	3	3	(75)
主轴反转	46	M	3	4	(75)
主轴停止	47	M	3	5	(75)
主轴定位	48	M	3	19	(75)
进给轴模式	49	M	3	70 2)	(75)
自动换档	50	M	3	40	(77)
齿轮档 1	51	M	3	41	(77)
齿轮档 2	52	M	3	42	(77)
齿轮档 3	53	M	3	43	(77)
齿轮档 4	54	M	3	44	(77)
齿轮档 5	55	M	3	45	(77)
主轴转速	56	S	3	-1	(76)

主轴专用的辅助功能，主轴 4					
系统功能	下标 <n>	类型	地址扩展	值	组
主轴正转	57	M	4	3	(78)
主轴反转	58	M	4	4	(78)
主轴停止	59	M	4	5	(78)
主轴定位	60	M	4	19	(78)
进给轴模式	61	M	4	70 2)	(78)
自动换档	62	M	4	40	(80)
齿轮档 1	63	M	4	41	(80)
齿轮档 2	64	M	4	42	(80)
齿轮档 3	65	M	4	43	(80)
齿轮档 4	66	M	4	44	(80)
齿轮档 5	67	M	4	45	(80)
主轴转速	68	S	4	-1	(79)

主轴专用的辅助功能，主轴 5					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
主轴正转	69	M	5	3	(81)
主轴反转	70	M	5	4	(81)
主轴停止	71	M	5	5	(81)
主轴定位	72	M	5	19	(81)
进给轴模式	73	M	5	70 [2)]	(81)
自动换档	74	M	5	40	(83)
齿轮档 1	75	M	5	41	(83)
齿轮档 2	76	M	5	42	(83)
齿轮档 3	77	M	5	43	(83)
齿轮档 4	78	M	5	44	(83)
齿轮档 5	79	M	5	45	(83)
主轴转速	80	S	5	-1	(82)

主轴专用的辅助功能，主轴 6					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
主轴正转	81	M	6	3	(84)
主轴反转	82	M	6	4	(84)
主轴停止	83	M	6	5	(84)
主轴定位	84	M	6	19	(84)
进给轴模式	85	M	6	70 [2)]	(84)
自动换档	86	M	6	40	(86)
齿轮档 1	87	M	6	41	(86)
齿轮档 2	88	M	6	42	(86)
齿轮档 3	89	M	6	43	(86)
齿轮档 4	90	M	6	44	(86)
齿轮档 5	91	M	6	45	(86)
主轴转速	92	S	6	-1	(85)

主轴专用的辅助功能，主轴 7					
系统功能	下标 <n>	类型	地址扩展	值	组
主轴正转	93	M	7	3	(87)
主轴反转	94	M	7	4	(87)
主轴停止	95	M	7	5	(87)
主轴定位	96	M	7	19	(87)
进给轴模式	97	M	7	70 2)	(87)
自动换档	98	M	7	40	(89)
齿轮档 1	99	M	7	41	(89)
齿轮档 2	100	M	7	42	(89)
齿轮档 3	101	M	7	43	(89)
齿轮档 4	102	M	7	44	(89)
齿轮档 5	103	M	7	45	(89)
主轴转速	104	S	7	-1	(88)

主轴专用的辅助功能，主轴 8					
系统功能	下标 <n>	类型	地址扩展	值	组
主轴正转	105	M	8	3	(90)
主轴反转	106	M	8	4	(90)
主轴停止	107	M	8	5	(90)
主轴定位	108	M	8	19	(90)
进给轴模式	109	M	8	70 2)	(90)
自动换档	110	M	8	40	(92)
齿轮档 1	111	M	8	41	(92)
齿轮档 2	112	M	8	42	(92)
齿轮档 3	113	M	8	43	(92)
齿轮档 4	114	M	8	44	(92)
齿轮档 5	115	M	8	45	(92)
主轴转速	116	S	8	-1	(91)

主轴专用的辅助功能，主轴 9					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
主轴正转	117	M	9	3	(93)
主轴反转	118	M	9	4	(93)
主轴停止	119	M	9	5	(93)
主轴定位	120	M	9	19	(93)
进给轴模式	121	M	9	70 [2)]	(93)
自动换档	122	M	9	40	(95)
齿轮档 1	123	M	9	41	(95)
齿轮档 2	124	M	9	42	(95)
齿轮档 3	125	M	9	43	(95)
齿轮档 4	126	M	9	44	(95)
齿轮档 5	127	M	9	45	(95)
主轴转速	128	S	9	-1	(94)

主轴专用的辅助功能，主轴 10					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
主轴正转	129	M	10	3	(96)
主轴反转	130	M	10	4	(96)
主轴停止	131	M	10	5	(96)
主轴定位	132	M	10	19	(96)
进给轴模式	133	M	10	70 [2)]	(96)
自动换档	134	M	10	40	(98)
齿轮档 1	135	M	10	41	(98)
齿轮档 2	136	M	10	42	(98)
齿轮档 3	137	M	10	43	(98)
齿轮档 4	138	M	10	44	(98)
齿轮档 5	139	M	10	45	(98)
主轴转速	140	S	10	-1	(97)

主轴专用的辅助功能，主轴 11					
系统功能	下标 <n>	类型	地址扩展	值	组
主轴正转	141	M	11	3	(99)
主轴反转	142	M	11	4	(99)
主轴停止	143	M	11	5	(99)
主轴定位	144	M	11	19	(99)
进给轴模式	145	M	11	70 [2)]	(99)
自动换档	146	M	11	40	(101)
齿轮档 1	147	M	11	41	(101)
齿轮档 2	148	M	11	42	(101)
齿轮档 3	149	M	11	43	(101)
齿轮档 4	150	M	11	44	(101)
齿轮档 5	151	M	11	45	(101)
主轴转速	152	S	11	-1	(100)

主轴专用的辅助功能，主轴 12					
系统功能	下标 <n>	类型	地址扩展	值	组
主轴正转	153	M	11	3	(102)
主轴反转	154	M	12	4	(102)
主轴停止	155	M	12	5	(102)
主轴定位	156	M	12	19	(102)
进给轴模式	157	M	12	70 [2)]	(102)
自动换档	158	M	12	40	(104)
齿轮档 1	159	M	12	41	(104)
齿轮档 2	160	M	12	42	(104)
齿轮档 3	161	M	12	43	(104)
齿轮档 4	162	M	12	44	(104)
齿轮档 5	163	M	12	45	(104)
主轴转速	164	S	12	-1	(103)

主轴专用的辅助功能，主轴 13					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
主轴正转	165	M	13	3	(105)
主轴反转	166	M	13	4	(105)
主轴停止	167	M	13	5	(105)
主轴定位	168	M	13	19	(105)
进给轴模式	169	M	13	70 [2)]	(105)
自动换档	170	M	13	40	(107)
齿轮档 1	171	M	13	41	(107)
齿轮档 2	172	M	13	42	(107)
齿轮档 3	173	M	13	43	(107)
齿轮档 4	174	M	13	44	(107)
齿轮档 5	175	M	13	45	(107)
主轴转速	176	S	13	-1	(106)

主轴专用的辅助功能，主轴 14					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
主轴正转	177	M	14	3	(108)
主轴反转	178	M	14	4	(108)
主轴停止	179	M	14	5	(108)
主轴定位	180	M	14	19	(108)
进给轴模式	181	M	14	70 [2)]	(108)
自动换档	182	M	14	40	(110)
齿轮档 1	183	M	14	41	(110)
齿轮档 2	184	M	14	42	(110)
齿轮档 3	185	M	14	43	(110)
齿轮档 4	186	M	14	44	(110)
齿轮档 5	187	M	14	45	(110)
主轴转速	188	S	14	-1	(109)

主轴专用的辅助功能，主轴 15					
系统功能	下标 <n>	类型	地址扩展	值	组
主轴正转	189	M	15	3	(111)
主轴反转	190	M	15	4	(111)
主轴停止	191	M	15	5	(111)
主轴定位	192	M	15	19	(111)
进给轴模式	193	M	15	70 [2)]	(111)
自动换档	194	M	15	40	(113)
齿轮档 1	195	M	15	41	(113)
齿轮档 2	196	M	15	42	(113)
齿轮档 3	197	M	15	43	(113)
齿轮档 4	198	M	15	44	(113)
齿轮档 5	199	M	15	45	(113)
主轴转速	200	S	15	-1	(112)

主轴专用的辅助功能，主轴 16					
系统功能	下标 <n>	类型	地址扩展	值	组
主轴正转	201	M	16	3	(114)
主轴反转	202	M	16	4	(114)
主轴停止	203	M	16	5	(114)
主轴定位	204	M	16	19	(114)
进给轴模式	205	M	16	70 [2)]	(114)
自动换档	206	M	16	40	(116)
齿轮档 1	207	M	16	41	(116)
齿轮档 2	208	M	16	42	(116)
齿轮档 3	209	M	16	43	(116)
齿轮档 4	210	M	16	44	(116)
齿轮档 5	211	M	16	45	(116)
主轴转速	212	S	16	-1	(115)

主轴专用的辅助功能，主轴 17					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
主轴正转	213	M	17	3	(117)
主轴反转	214	M	17	4	(117)
主轴停止	215	M	17	5	(117)
主轴定位	216	M	17	19	(117)
进给轴模式	217	M	17	70 [2)]	(117)
自动换档	218	M	17	40	(119)
齿轮档 1	219	M	17	41	(119)
齿轮档 2	220	M	17	42	(119)
齿轮档 3	221	M	17	43	(119)
齿轮档 4	222	M	17	44	(119)
齿轮档 5	223	M	17	45	(119)
主轴转速	224	S	17	-1	(118)

主轴专用的辅助功能，主轴 18					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
主轴正转	225	M	18	3	(120)
主轴反转	226	M	18	4	(120)
主轴停止	227	M	18	5	(120)
主轴定位	228	M	18	19	(120)
进给轴模式	229	M	18	70 [2)]	(120)
自动换档	230	M	18	40	(122)
齿轮档 1	231	M	18	41	(122)
齿轮档 2	232	M	18	42	(122)
齿轮档 3	233	M	18	43	(122)
齿轮档 4	234	M	18	44	(122)
齿轮档 5	235	M	18	45	(122)
主轴转速	236	S	18	-1	(121)

主轴专用的辅助功能，主轴 19					
系统功能	下标 <n>	类型	地址扩展	值	组
主轴正转	237	M	19	3	(123)
主轴反转	238	M	19	4	(123)
主轴停止	239	M	19	5	(123)
主轴定位	240	M	19	19	(123)
进给轴模式	241	M	19	70 [2)]	(123)
自动换档	242	M	19	40	(125)
齿轮档 1	243	M	19	41	(125)
齿轮档 2	244	M	19	42	(125)
齿轮档 3	245	M	19	43	(125)
齿轮档 4	246	M	19	44	(125)
齿轮档 5	247	M	19	45	(125)
主轴转速	248	S	19	-1	(124)

主轴专用的辅助功能，主轴 20					
系统功能	下标 <n>	类型	地址扩展	值	组
主轴正转	249	M	20	3	(126)
主轴反转	250	M	20	4	(126)
主轴停止	251	M	20	5	(126)
主轴定位	252	M	20	19	(126)
进给轴模式	253	M	20	70 [2)]	(126)
自动换档	254	M	20	40	(128)
齿轮档 1	255	M	20	41	(128)
齿轮档 2	256	M	20	42	(128)
齿轮档 3	257	M	20	43	(128)
齿轮档 4	258	M	20	44	(128)
齿轮档 5	259	M	20	45	(128)
主轴转速	260	S	20	-1	(127)

刀架专用的辅助功能、T 辅助功能					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
刀具选择	261	T	1	-1	129
刀具选择	262	T	2	-1	130
刀具选择	263	T	3	-1	131
刀具选择	264	T	4	-1	132
刀具选择	265	T	5	-1	133
刀具选择	266	T	6	-1	134
刀具选择	267	T	7	-1	135
刀具选择	268	T	8	-1	136
刀具选择	269	T	9	-1	137
刀具选择	270	T	10	-1	138
刀具选择	271	T	11	-1	139
刀具选择	272	T	12	-1	140
刀具选择	273	T	13	-1	141
刀具选择	274	T	14	-1	142
刀具选择	275	T	15	-1	143
刀具选择	276	T	16	-1	144
刀具选择	277	T	17	-1	145
刀具选择	278	T	18	-1	146
刀具选择	279	T	19	-1	147
刀具选择	280	T	20	-1	148

刀架专用的辅助功能、M6 辅助功能					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
换刀	281	M	1	6 [1)]	149
换刀	282	M	2	6 [1)]	150
换刀	283	M	3	6 [1)]	151
换刀	284	M	4	6 [1)]	152
换刀	285	M	5	6 [1)]	153

刀架专用的辅助功能、M6 辅助功能					
系统功能	**下标 <n>**	**类型**	**地址扩展**	**值**	**组**
换刀	286	M	6	6 1)	154
换刀	287	M	7	6 1)	155
换刀	288	M	8	6 1)	156
换刀	289	M	9	6 1)	157
换刀	290	M	10	6 1)	158
换刀	291	M	11	6 1)	159
换刀	292	M	12	6 1)	160
换刀	293	M	13	6 1)	161
换刀	294	M	14	6 1)	162
换刀	295	M	15	6 1)	163
换刀	296	M	16	6 1)	164
换刀	297	M	17	6 1)	165
换刀	298	M	18	6 1)	166
换刀	299	M	19	6 1)	167
换刀	300	M	20	6 1)	168

图例：

( ) 可对该值进行修改。

1) 该值取决于机床数据：

MD22560 $MC_TOOL_CHANGE_M_MODE（用于换刀的 M 功能）

2) 可通过以下机床数据为该值预设另一个值：

MD20095 $MC_EXTERN_RIGID_TAPPING_M_NR（用于切换到开环控制轴运行的 M 功能（外部模式））

MD20094 $MC_SPIND_RIGID_TAPPING_M_NR（用于切换到开环控制轴运行的 M 功能）

提示

输出到 PLC 上的值始终为值 70。

3) 该值是通过以下机床数据设置的：

MD22254 $MC_AUXFU_ASSOC_M0_VALUE（用于程序停止的附加 M 功能）

4) 该值是通过以下机床数据设置的：
MD22256 $MC_AUXFU_ASSOC_M1_VALUE（用于有条件停止的附加 M 功能）

5) 该值是通过以下机床数据设置的：
MD10714 $MN_M_NO_FCT_EOP（复位后，主轴的 M 功能生效）

6) 该值是通过以下机床数据设置的：
MD26008 $MC_NIBBLE_PUNCH_CODE（确定 M 功能）

9.2.2 一览：输出特性

下表中列出的参数的含义：

参数	含义
下标 <n>	辅助功能参数的机床数据下标
输出特性	MD22080 $MC_AUXFU_PREDEF_SPEC[<n>]，位 0 ... 18 位 19 ... 31：预留

预定义辅助功能的输出特性

系统功能	下标 <n>																			
		输出特性，位																		
18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
停止	0	0	0	0	(0)	0	0	0	0	(0)	0	0	1	0	0	0	0	0	(0)	(1)
有条件停止	1	0	0	0	(0)	0	0	0	0	(0)	0	0	1	0	0	0	0	0	(0)	(1)
子程序结束	2	0	0	0	(0)	0	0	0	0	(0)	0	0	1	0	0	0	0	0	(0)	(1)
	3	0	0	0	(0)	0	0	0	0	(0)	0	0	1	0	0	0	0	0	(0)	(1)
	4	0	0	0	(0)	0	0	0	0	(0)	0	0	1	0	0	0	0	0	(0)	(1)
换刀	5	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	(0)	(0)	(1)	(0)	(0)	(0)	(0)	(1)

系统功能	下标 <n>																			
		输出特性，位																		
主轴正转	6	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	(0)	(0)	(1)	(0)	(0)	(0)	(0)	(1)
主轴反转	7	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	(0)	(0)	(1)	(0)	(0)	(0)	(0)	(1)
主轴停止	8	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	(0)	(0)	(1)	(0)	(0)	(0)	(0)	(1)
主轴定位	9	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	(0)	(0)	(1)	(0)	(0)	(0)	(0)	(1)
进给轴模式	10	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	(0)	(0)	(1)	(0)	(0)	(0)	(0)	(1)
自动换档	11	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	0	0	1	(0)	(0)	(0)	(0)	(1)
齿轮档 1	12	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	0	0	1	(0)	(0)	(0)	(0)	(1)
齿轮档 2	13	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	0	0	1	(0)	(0)	(0)	(0)	(1)
齿轮档 3	14	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	0	0	1	(0)	(0)	(0)	(0)	(1)
齿轮档 4	15	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	0	0	1	(0)	(0)	(0)	(0)	(1)
齿轮档 5	16	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	0	0	1	(0)	(0)	(0)	(0)	(1)
主轴转速	17	0	0	0	(0)	0	0	0	(0)	(0)	(0)	0	(0)	(1)	(0)	(0)	(0)	0	(0)	(1)
进给率	18	0	0	0	(0)	0	0	0	(0)	(0)	(0)	0	(0)	(1)	(0)	0	(1)	0	(0)	(1)
刀沿选择	19	0	0	0	(0)	0	0	0	(0)	(0)	(0)	0	(0)	(0)	(1)	0	(0)	0	(0)	(1)
DL	20	0	0	0	(0)	0	0	0	(0)	(0)	(0)	0	(0)	(0)	(1)	0	(0)	0	(0)	(1)
刀具选择	21	0	0	0	(0)	0	0	0	(0)	(0)	(0)	0	(0)	(0)	(1)	0	(0)	0	(0)	(1)

系统功能	下标 <n>																			
		输出特性，位																		
停止（组合式）	22	0	0	0	(0)	0	0	0	0	(0)	0	0	1	0	0	0	0	0	(0)	(1)
有条件停止（组合式）	23	0	0	0	(0)	0	0	0	0	(0)	0	0	1	0	0	0	0	0	(0)	(1)
子程序结束	24	0	0	0	(0)	0	0	0	0	(0)	0	0	1	0	0	0	0	0	(0)	(1)
步冲	25	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	(0)	(1)	(0)	0	(0)	(0)	(0)	(1)
步冲	26	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	(0)	(1)	(0)	0	(0)	(0)	(0)	(1)
步冲	27	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	(0)	(1)	(0)	0	(0)	(0)	(0)	(1)
步冲	28	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	(0)	(1)	(0)	0	(0)	(0)	(0)	(1)
步冲	29	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	(0)	(1)	(0)	0	(0)	(0)	(0)	(1)
步冲	30	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	(0)	(1)	(0)	0	(0)	(0)	(0)	(1)
步冲	31	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	(0)	(1)	(0)	0	(0)	(0)	(0)	(1)
步冲	32	0	0	0	(0)	0	0	0	(0)	(0)	(0)	(0)	(0)	(1)	(0)	0	(0)	(0)	(0)	(1)

( ) 可对该值进行修改。

位的含义

位	含义
0	一个 OB1 周期后“normal（普通）”应答
	采用普通应答的辅助功能会在 OB1 周期开始时输出至 NC/PLC 接口。 此时系统通过辅助功能专用变更信号向 PLC 用户程序指示辅助功能有效。 组织块 OB1 完全运行一次后便会立即对辅助功能进行应答。 这相当于一个完整的 PLC 用户循环。 采用普通应答的辅助功能与写入了该功能的零件程序段同步输出。 如果零件程序段的执行（例如轨迹轴和/或定位轴运动）在应答辅助功能前结束，那么程序段切换将延迟，直至通过 PLC 应答。 在连续路径运行中，若与采用普通应答的辅助功能组合使用，那么保持恒定的轨迹速度需满足以下条件：在运行中输出辅助功能，并在到达程序段末尾前通过 PLC 进行应答。
1	通过 OB40“quick（快速）”应答
	采用快速应答的辅助功能会在下一个 OB1 周期前输出至 NC/PLC 接口。 此时系统通过辅助功能专用变更信号向 PLC 用户程序指示辅助功能有效。 PLC 基本程序会在下一个 OB40 周期中立即对辅助功能进行应答。 因此应答辅助功能并不能保证执行了对应的 PLC 用户功能。 辅助功能仍然在 OB1 周期中执行。 因此只有在该 OB1 周期结束后辅助功能才会接着输出给 PLC。 连续路径运行中在多个连续零件程序段中输出采用快速应答的辅助功能时，这一点尤为明显（轨迹速度降低）。 对于采用快速应答的辅助功能，无法确保 PLC 用户程序中的响应与程序段同步。 提示 只有在与用户自定义辅助功能一同使用时，才可将辅助功能的输出特性设置为“快速辅助功能”。
2	非预定义辅助功能
	采用此设置时，预定义辅助功能会被作为用户自定义辅助功能处理。 此时辅助功能不再会触发对应的系统功能，而只是输出至 PLC。 示例： 将辅助功能“主轴定位”（下标 9）重设为采用普通应答的用户自定义辅助功能，并在运行前输出。 MD22080 $MC_AUXFU_PREDEF_SPEC [9] = 'H25' (100101B)
3	不输出至 PLC
	辅助功能不输出至 PLC。
4	通过 PLC 应答后执行主轴响应
	在通过 PLC 应答后再执行对应的系统功能。
5	运行前输出
	在零件程序段中编写的运行（轨迹轴和/或程序段相关定位轴运动）前将辅助功能输出至 PLC。

位	含义
6	运行中输出
	在零件程序段中编写的运行（轨迹轴和/或程序段相关定位轴运动）期间将辅助功能输出至 PLC。
7	在程序段末尾输出
	在零件程序段中编写的运行（轨迹轴和/或程序段相关定位轴运动）完成后将辅助功能输出至 PLC。
8	程序段搜索类型 1、2、4 后不输出
	程序段搜索类型 1、2、4：程序段搜索中收集的辅助功能不输出。
9	在带程序测试的程序段搜索（类型 5，SERUPRO）中收集
	在带程序测试的程序段搜索中，辅助功能按照各组别收集在以下系统变量中： • $AC_AUXFU_M_VALUE[<n>] • $AC_AUXFU_M_EXT[<n>] • $AC_AUXFU_M_STATE[<n>]
10	在带程序测试的程序段搜索（类型 5，SERUPRO）中不输出
	在带程序测试的程序段搜索中，辅助功能不输出至 PLC。
11	跨通道辅助功能（SERUPRO）
	在带程序测试的程序段搜索（SERUPRO）中，辅助功能跨通道收集到全局辅助功能列表中。 提示 每个辅助功能组只会收集该组的最后一个辅助功能。
12	已通过同步动作输出（只读）
	若辅助功能已通过同步动作输出至 PLC，则该位置位。
13	隐性辅助功能（只读）
	若辅助功能已隐性输出至 PLC，则该位置位。
14	M01 生效（只读）
	若 M01 生效时辅助功能已输出至 PLC，则该位置位。
15	定位测试运行期间不输出
	在定位测试运行期间，辅助功能不输出至 PLC。
16	关闭步冲
17	激活步冲
18	步冲

说明

对于未定义输出特性的辅助功能，以下缺省输出特性生效：

- 位 0 = 1：输出持续一个 OB1 周期
- 位 7 = 1：在程序段末尾输出

9.2.3 参数设置

9.2.3.1 分组指定

程序段搜索时，辅助功能的处理方式通过其分组确定。 168 个可供使用的辅助功能组又划分为预定义组和用户自定义组：

预定义组：	1 ... 4	10 ... 12	72 ... 168
用户自定义组：	5 ... 9	13 ... 71	

默认状况下，每个预定义辅助功能属于一个辅助功能组。 通过以下机床数据可修改多数预定义辅助功能的分组。

MD22040 $MC_AUXFU_PREDEF_GROUP[<n>]（预定义辅助功能的分组指定）

若不将辅助功能指定给任何组，则须在机床数据中输入“0”值。

下标 <n> 为以下数字的预定义辅助功能不可修改分组：0、1、2、3、4、22、23、24

说明

第 1 辅助功能组和程序段搜索

程序段搜索中，第 1 辅助功能组的辅助功能只会被收集，而不会被输出。

9.2.3.2 类型、地址扩展和数值

辅助功能通过类型、地址扩展和数值这几个参数进行编程（参见“辅助功能编程 (页 472)”章节）。

类型

“类型”参数用于定义辅助功能的标识符，例如：

"M"	代表附加功能
"S"	代表主轴功能
"F"	代表进给率

此时通过以下机床数据设置：

MD22050 $MC_AUXFU_PREDEF_TYPE[<n>]（预定义辅助功能的类型）

说明

预定义辅助功能的“类型”无法修改。

地址扩展

辅助功能的“地址扩展”参数用于确定同类型不同分量的地址。 对于预定义辅助功能，“地址扩展”的值为辅助功能所关联的主轴编号。

此时通过以下机床数据设置：

MD22060 $MC_AUXFU_PREDEF_EXTENSION[<n>]（预定义辅助功能的地址扩展）

辅助功能汇总

若要将一个通道中所有主轴的某个辅助功能指定给同一个辅助功能组，那么须将“地址扩展”参数的值设为“-1”。

示例：

针对通道的所有主轴，辅助功能 M3（机床数据下标 = 6）指定给第 2 辅助功能组。

```
MD22040 $MC_AUXFU_PREDEF_GROUP[ 6 ]      = 2
MD22050 $MC_AUXFU_PREDEF_TYPE[ 6 ]       = "M"
MD22060                                   = -1
$MC_AUXFU_PREDEF_EXTENSION[ 6 ]
MD22070 $MC_AUXFU_PREDEF_VALUE[ 6 ]      = 3
```

值

“值”和“类型”参数用于定义辅助功能的含义，即基于此辅助功能激活的系统功能。

辅助功能的“值”在以下机床数据中定义：

MD22070 $MC_AUXFU_PREDEF_VALUE[<n>]（预定义辅助功能的数值）

说明

预定义辅助功能的“值”无法修改。 个别预定义辅助功能的“值”可通过附加的机床数据重新配置（参见“关联辅助功能 (页 467)”章节）。

9.2.3.3 输出特性

“输出特性”参数用于定义预定义辅助功能输出至 NC/PLC 接口的时间，以及由 PLC 对其进行应答的时间。

此时通过以下机床数据设置：

MD22080 $MC_AUXFU_PREDEF_SPEC[<n>]（预定义辅助功能的输出特性）

以运行为基准的输出特性

运行前输出

- 前一零件程序段的运行（轨迹轴和/或程序段相关定位轴运行）通过准停结束。
- 辅助功能在当前零件程序段起始处输出。
- PLC 对辅助功能进行过应答后，当前零件程序段才开始运行（轨迹轴和/或定位轴运行）。
  - 输出持续一个 OB1 周期（普通应答）：一个 OB1 周期后输出
  - 输出持续一个 OB40 周期（快速应答）：一个 OB40 周期后输出

运行中输出

- 辅助功能在运行（轨迹轴和/或定位轴运行）开始时输出。
- 此时系统会降低当前零件程序段的轨迹速度，使到达程序段末尾所花费的时间大于 PLC 应答辅助功能所用的时间：
  - 输出持续一个 OB1 周期（普通应答）：一个 OB1 周期
  - 输出持续一个 OB40 周期（快速应答）：一个 OB40 周期

运行后输出

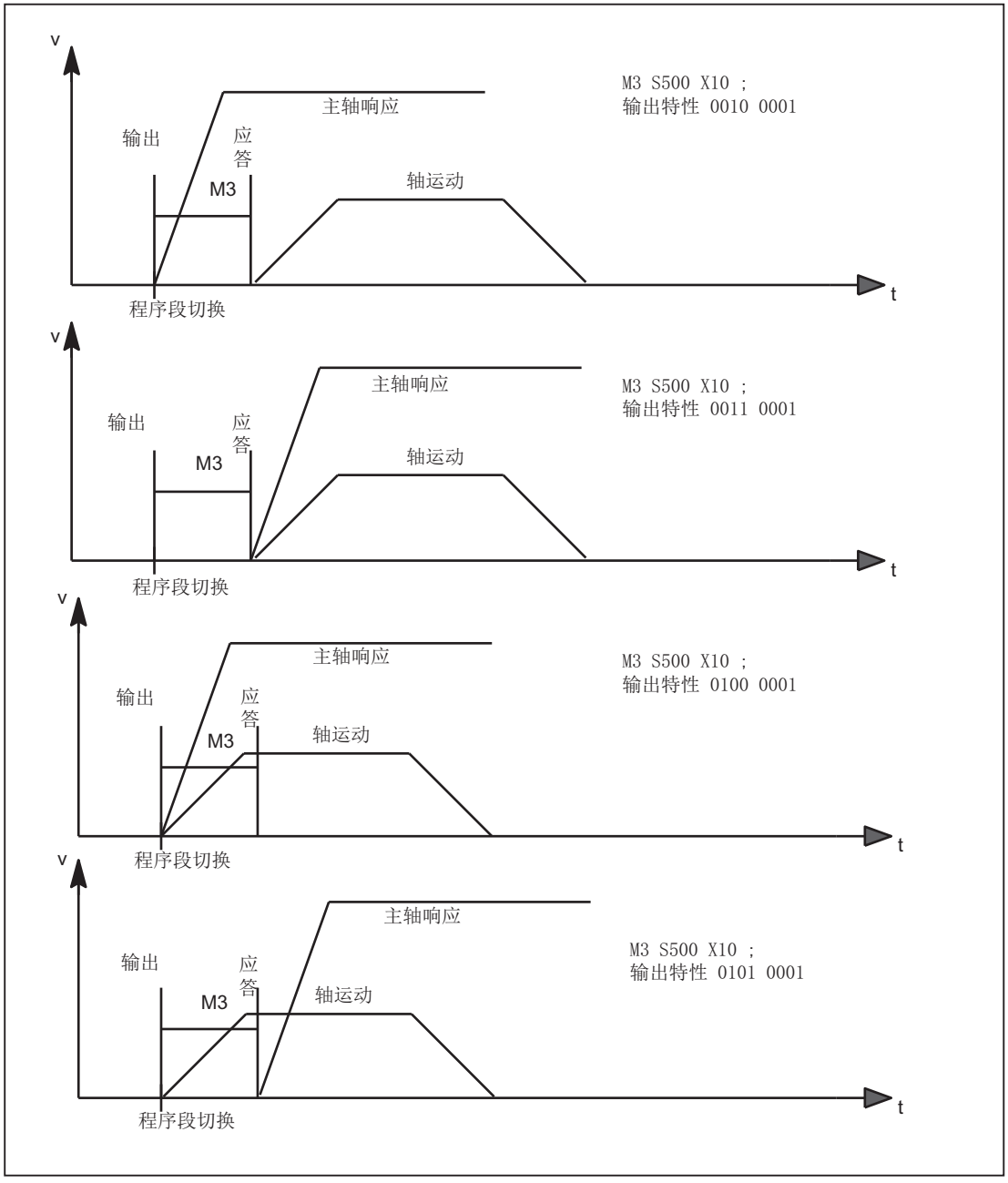
- 当前零件程序段的运行（轨迹轴和/或程序段相关定位轴运行）通过准停结束。
- 辅助功能在运行结束后输出。
- PLC 对辅助功能进行应答后，系统执行程序段切换。
  - 输出持续一个 OB1 周期（普通应答）：一个 OB1 周期后输出
  - 输出持续一个 OB40 周期（快速应答）：一个 OB40 周期后输出

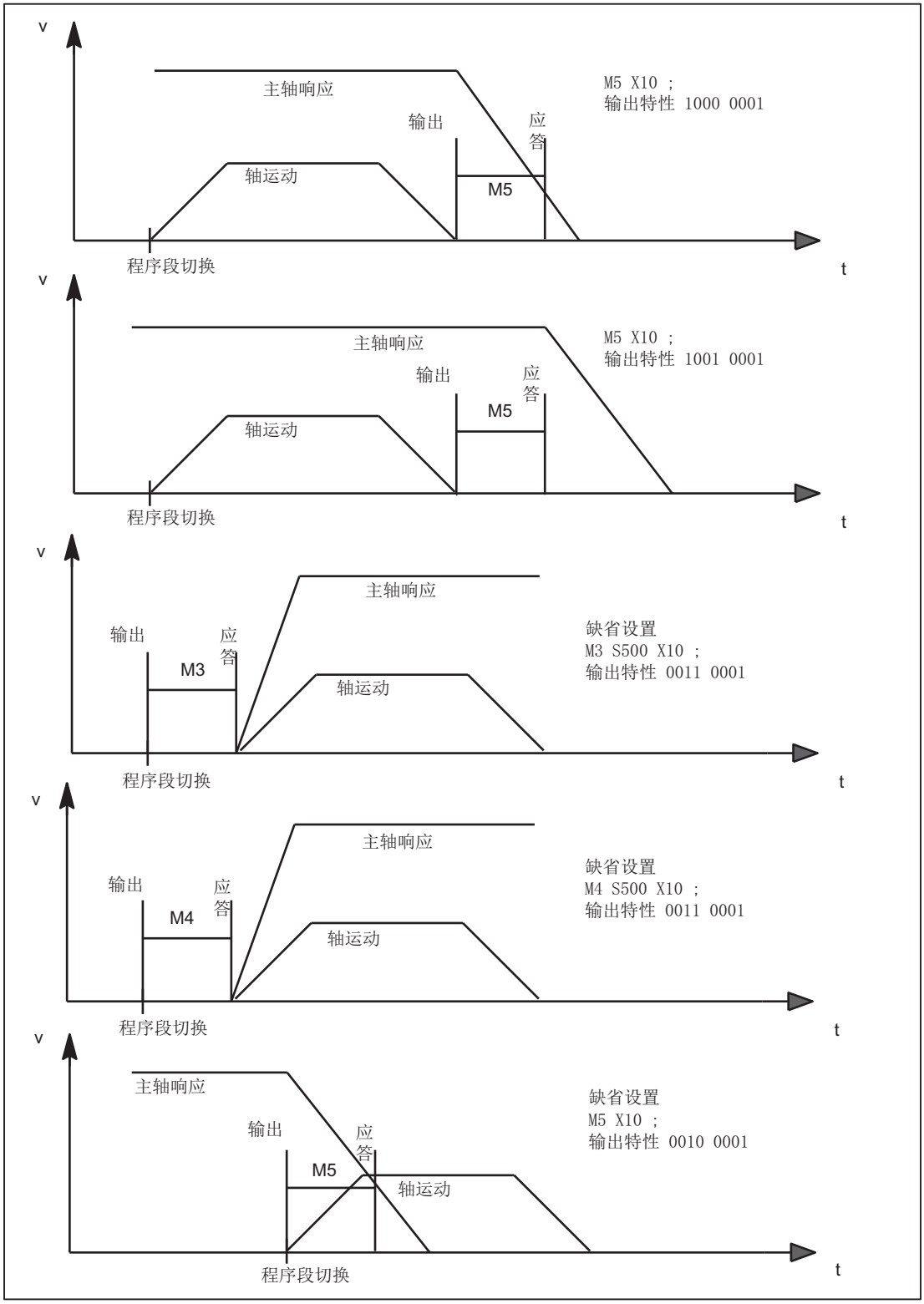
不同输出特性的示例

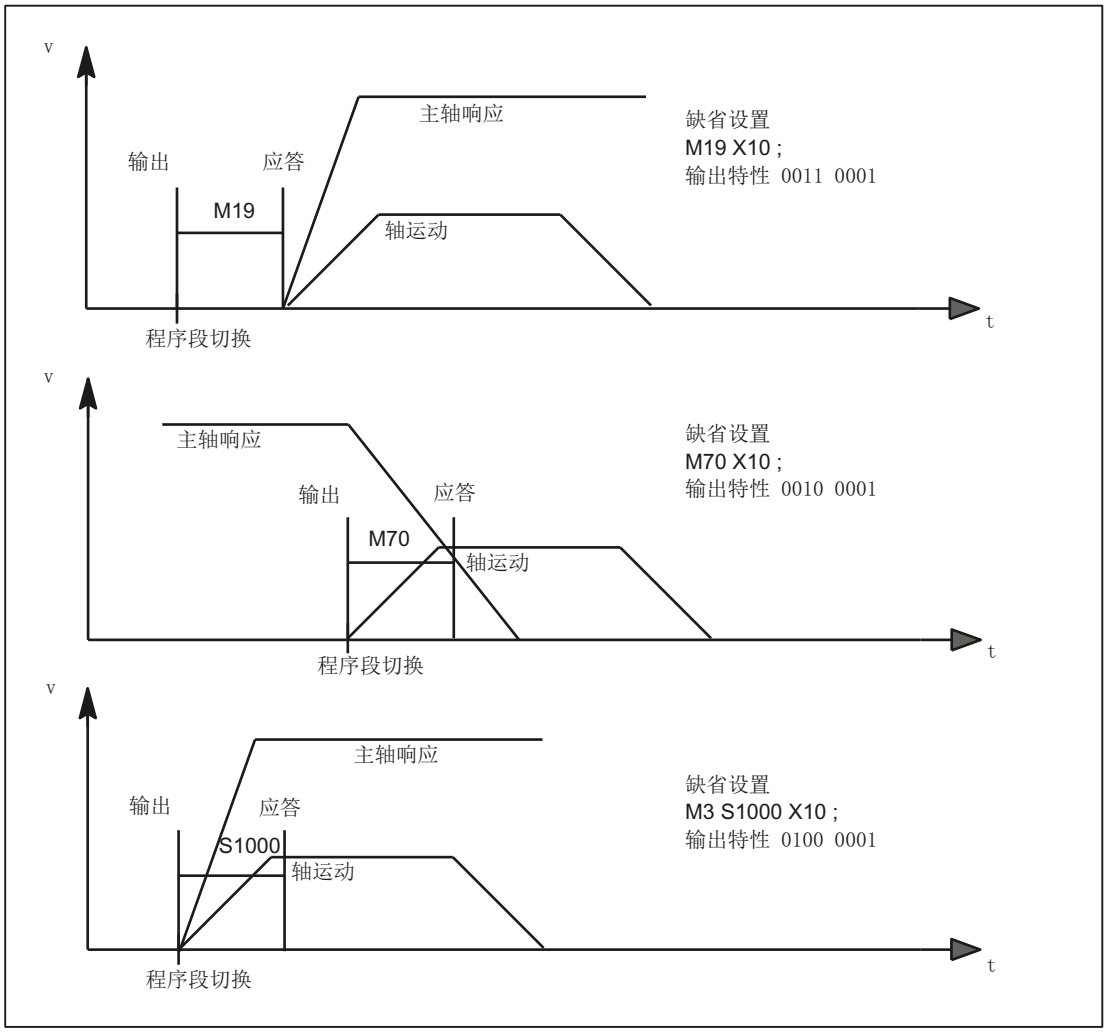
下图直观地展示了基于不同因素时的输出特性：

- 辅助功能的输出和应答
- 主轴响应（转速变化）
- 运行（速度变化）

图中“输出特性”下标注的二进制值以设置的输出特性（MD22080）为基准。


v
主轴响应
M3 S500 X10 ;
输出特性 0010 0001
输出
应答
M3
轴运动
t
程序段切换
v
主轴响应
M3 S500 X10 ;
输出特性 0011 0001
输出
应答
M3
轴运动
t
程序段切换
v
主轴响应
M3 S500 X10 ;
输出特性 0100 0001
输出
应答
M3
轴运动
t
程序段切换
v
主轴响应
M3 S500 X10 ;
输出特性 0101 0001
输出
应答
M3
轴运动
t
程序段切换


v
主轴响应
输出
应答
M5 X10 ;
输出特性 1000 0001
轴运动
M5
程序段切换
t
v
主轴响应
输出
应答
M5 X10 ;
输出特性 1001 0001
M5
轴运动
程序段切换
t
v
主轴响应
输出
应答
M3
缺省设置
M3 S500 X10 ;
输出特性 0011 0001
轴运动
程序段切换
t
v
主轴响应
输出
应答
M4
缺省设置
M4 S500 X10 ;
输出特性 0011 0001
轴运动
程序段切换
t
v
主轴响应
输出
应答
M5
缺省设置
M5 X10 ;
输出特性 0010 0001
轴运动
程序段切换
t



9.3 用户自定义辅助功能

用户自定义辅助功能的使用分为两个部分：

- 预定义辅助功能的扩展
- 用户专用辅助功能

预定义辅助功能的扩展

预定义辅助功能的机床数据只能存在一次，因此只能通过其确定通道中一根主轴的地址。为了确定其他主轴的地址，必须参设置用户自定义辅助功能，作为对预定义辅助功能的扩展。

预定义辅助功能的扩展只基于“地址扩展”参数。 在“地址扩展”参数中输入辅助功能对应的主轴的编号。

可对以下系统功能所对应的预定义辅助功能进行扩展：

系统功能	**类型**		
		地址扩展 [1]	
			值
换刀	M	1	6
主轴正转	M	1	3
主轴反转	M	1	4
主轴停止	M	1	5
主轴定位	M	1	19
进给轴模式	M	1	70
自动换档	M	1	40
齿轮档 1	M	1	41
齿轮档 2	M	1	42
齿轮档 3	M	1	43
齿轮档 4	M	1	44
齿轮档 5	M	1	45
主轴转速	S	1	-1
刀具选择	T	1	-1

[1] 地址扩展 = 1 为预定义辅助功能的机床数据中使用的缺省值

示例：

系统功能“主轴正转”的预定义辅助功能扩展，针对通道中第 2 主轴和第 3 主轴。

辅助功能“主轴正转”，针对通道第 2 主轴：

MD22010 $MC_AUXFU_ASSIGN_TYPE [n] = "M"

MD22020 $MC_AUXFU_ASSIGN_EXTENSION [n] = 2

MD22030 $MC_AUXFU_ ASSIGN_VALUE [n] = 3

辅助功能“主轴正转”，针对通道第 3 主轴：

MD22010 $MC_AUXFU_ ASSIGN_TYPE [m] = "M"

MD22020 $MC_AUXFU_ASSIGN_EXTENSION [m]　= 3
MD22030 $MC_AUXFU_ ASSIGN_VALUE [m]　= 3

用户专用辅助功能

用户专用辅助功能有以下属性：

- 用户专用辅助功能只能用于激活用户功能。
- 用户专用辅助功能无法用于激活系统功能。
- 用户专用辅助功能会根据设置的输出特性输出至 PLC。
- 用户专用辅助功能的功能性由机床制造商/用户在 PLC 用户程序中实现。

9.3.1 参数设置

9.3.1.1 用户自定义辅助功能的最大数量

每个通道中用户自定义辅助功能的最大数量可通过以下机床数据设置：

MD11100 $MN_AUXFU_MAXNUM_GROUP_ASSIGN（用户自定义辅助功能的最大数量）

9.3.1.2 分组指定

程序段搜索时，辅助功能的处理方式通过其分组确定。 168 个可供使用的辅助功能组又划分为预定义组和用户自定义组：

预定义组：	1 ... 4	10 ... 12	72 ... 168
用户自定义组：	5 ... 9	13 ... 71	

默认状况下，每个用户自定义辅助功能被指定给第 1 个辅助功能组。 分组指定可通过以下机床数据修改：

MD22000 $MC_AUXFU_ASSIGN_GROUP[<n>]（用户自定义辅助功能的分组指定）

若不将辅助功能指定给任何组，则须在机床数据中输入“0”值。

说明

第 1 辅助功能组和程序段搜索

程序段搜索中，第 1 辅助功能组的辅助功能只会被收集，而不会被输出。

9.3.1.3 类型、地址扩展和数值

辅助功能通过类型、地址扩展和数值这几个参数进行编程（参见“辅助功能编程 (页 472)”章节）。

类型

“类型”参数用于定义辅助功能的标识符。

用户自定义辅助功能的标识符有：

类型	名称	含义
"H"	辅助功能	用户专用辅助功能
"M"	附加功能	预定义辅助功能的扩展
"S"	主轴功能	
"T"	刀具号	

此时通过以下机床数据设置：

MD22010 $MC_AUXFU_ASSIGN_TYPE[<n>]（用户自定义辅助功能的类型）

地址扩展

MD22020 $MC_AUXFU_ASSIGN_EXTENSION[<n>]（用户自定义辅助功能的地址扩展）

对于用户专用辅助功能，地址扩展参数的功能不确定。 其通常用于区分“值”相同的辅助功能。

辅助功能汇总

若需将所有类型和数值相同的辅助功能指定给同一个辅助功能组，那么须将“地址扩展”参数的值设为“-1”。

示例：

将所有数值 = 8 的用户专用辅助功能指定给第 10 辅助功能组。

```
MD22000 $MC_AUXFU_ASSIGN_GROUP [ 1 ]          = 10
MD22010 $MC_AUXFU_ ASSIGN_TYPE [ 1 ]          = "H"
MD22020 $MC_AUXFU_ ASSIGN_EXTENSION [ 1 ]     = -1
MD22030 $MC_AUXFU_ ASSIGN_VALUE [ 1 ]         = 8
```

值

MD22030 $MC_AUXFU_ASSIGN_VALUE[<n>]（用户自定义辅助功能的数值）

对于用户专用辅助功能，“值”参数的功能不确定。 通常其用于激活对应的 PLC 用户功能。

辅助功能汇总

若需将所有类型和地址扩展相同的辅助功能指定给同一个辅助功能组，那么须将“值”参数的值设为“-1”。

示例：

将所有地址扩展 = 2 的用户专用辅助功能指定给第 11 辅助功能组。

```
MD22000 $MC_AUXFU_ASSIGN_GROUP [ 2 ]          = 11
MD22010 $MC_AUXFU_ ASSIGN_TYPE [ 2 ]          = "H"
MD22020 $MC_AUXFU_ ASSIGN_EXTENSION [ 2 ]     = 2
MD22030 $MC_AUXFU_ ASSIGN_VALUE [ 2 ]         = -1
```

9.3.1.4 输出特性

用户自定义辅助功能的“输出特性”可通过以下机床数据设置：

MD22035 $MC_AUXFU_ASSIGN_SPEC[<n>]（用户自定义辅助功能的输出特性）

对各输出参数的说明请见预定义辅助功能的“输出特性 (页 458)”章节。 该章节的内容大体上也适用于用户自定义辅助功能的输出特性。

9.4 关联辅助功能

功能

关联辅助功能为用户自定义辅助功能，其与对应的预定义辅助功能有相同的作用。可为以下预定义辅助功能关联用户自定义辅助功能：

- M0（停止）
- M1（有条件停止）

前提条件

将一个用户自定义辅助功能关联至上述预定义辅助功能的前提条件是：对用户自定义辅助功能的设置。此时用户自定义辅助功能的“类型”参数只允许设置为“M”。

参数设置

通过以下机床数据将一个用户自定义辅助功能关联至上述预定义辅助功能。

MD22254 $MC_AUXFU_ASSOC_M0_VALUE（程序停止的附加 M 功能）

MD22256 $MC_AUXFU_ASSOC_M1_VALUE（有条件停止的附加 M 功能）

分组指定

关联的用户自定义辅助功能的分组固定采用对应的预定义辅助功能的分组。

应用

关联辅助功能可用于：

- 主程序
- 子程序
- 循环

说明

关联辅助功能不可在同步动作中使用。

NC/PLC 接口信号

使用关联用户自定义辅助功能时，输出至 NC/PLC 接口的信号与使用对应预定义辅助功能时相同；不过输出的数值还是用户自定义辅助功能的数值（“值”参数），用于区分实际编写了哪个辅助功能，这样一来便可在 PLC 用户程序中区分预定义和用户自定义辅助功能。

说明

此外，修改机床数据 MD22254 和/或 MD22256 还需对 PLC 用户程序进行相应调整。

专用 NC/PLC 接口信号

有以下专用 NC/PLC 接口信号可供使用：

- DB21, ... DBX24.4（激活 NC 关联 M01）
- DB21, ... DBX30.5（激活关联 M01）
- DB21, ... DBX318.5（关联 M01 生效）

前提条件

注意下列前提条件：

- 一个用户自定义辅助功能不允许被多次关联。
- 预定义辅助功能（例如 M3、M4、M5 等）不可被关联。

示例

将用户自定义辅助功能 M123 关联至 M0：

MD22254 $MC_AUXFU_ASSOC_M0_VALUE = 123

现在用户自定义辅助功能 M123 便具有和 M0 相同的功能。

9.5 类型专用输出特性

功能

基于零件程序段中编写的运行的辅助功能输出特性可针对类型定义。

参数设置

类型专用输出特性通过以下机床数据设置：

MD22200 $MC_AUXFU_**M**_SYNC_TYPE（M 功能输出时间点）

MD22210 $MC_AUXFU_**S**_SYNC_TYPE（S 功能输出时间点）

MD22220 $MC_AUXFU_**T**_SYNC_TYPE（T 功能输出时间点）

MD22230 $MC_AUXFU_**H**_SYNC_TYPE（H 功能输出时间点）

MD22240 $MC_AUXFU_**F**_SYNC_TYPE（F 功能输出时间点）

MD22250 $MC_AUXFU_**D**_SYNC_TYPE（D 功能输出时间点）

MD22252 $MC_AUXFU_**DL**_SYNC_TYPE（DL 功能输出时间点）

可设置以下输出特性：

MD $MC_AUXFU_**xx**_SYNC_TYPE = <数值>

值	输出特性
0	运行前输出
1	运行中输出
2	在程序段末尾输出
3	不输出至 PLC
4	根据通过 MD22080 定义的输出特性进行输出

对各种输出特性的说明请见“输出特性 (页 458)”章节。

说明

各辅助功能类型可设置的输出特性请见参数手册中的“详细机床数据说明”。

示例

在一个包含运行的零件程序段中输出具有不同输出特性的辅助功能。

设置的输出特性：

MD22200 $MC_AUXFU_M_SYNC_TYPE = 1 ⇒ M 功能：运行**中**输出

MD22220 $MC_AUXFU_T_SYNC_TYPE = 0 ⇒ T 功能：运行**前**输出

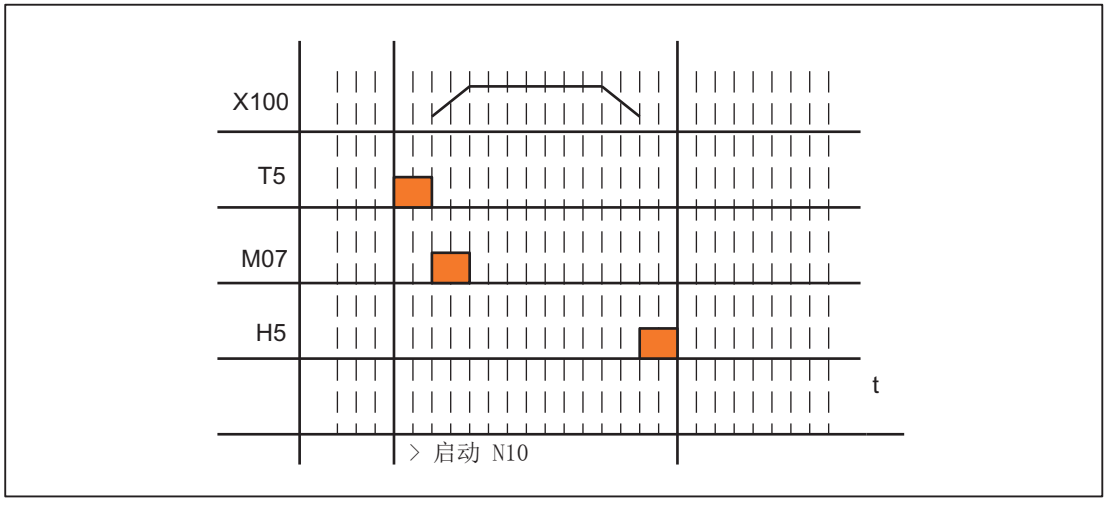
MD22230 $MC_AUXFU_H_SYNC_TYPE = 2 ⇒ H 功能：**在程序段末尾**输出

零件程序段：

程序代码

```
...
N10 G01 X100 M07 H5 T5
...
```

辅助功能输出的时间顺序：



9.6 设置的输出特性的优先级

设置的输出特性的优先级须参考以下标准分别加以考量：

- 输出时长（普通应答/快速应答）
- 以运行为基准的输出（运行前/中/后输出）

通常情况下，未设置高优先级的输出特性时，优先级低一级的输出特性生效。

输出时长

以下优先级适用于输出时长：

优先级	输出特性	定义方式：
最高	辅助功能专用	零件程序指令：QU(…) （参见“可编程的输出时长 (页 473)”章节）
↓	辅助功能专用	MD22035 $MC_AUXFU_ASSIGN_SYNC[<n>] MD22080 $MC_AUXFU_PREDEF_SYNC[<n>]
↓	组专用	MD11110 $MC_AUXFU_GROUP_SPEC[<n>]
最低	未定义	缺省输出特性：输出持续一个 OB1 周期

以运行为基准的输出

以下优先级适用于以运行为基准的输出：

优先级	输出特性	定义方式：
最高	辅助功能专用	MD22035 $MC_AUXFU_ASSIGN_SYNC[<n>] MD22080 $MC_AUXFU_PREDEF_SYNC[<n>]
↓	组专用	MD11110 $MC_AUXFU_GROUP_SPEC[<n>]
↓	类型专用	MD22200 $MC_AUXFU_M_SYNC_TYPE MD22210 $MC_AUXFU_S_SYNC_TYPE MD22220 $MC_AUXFU_T_SYNC_TYPE MD22230 $MC_AUXFU_H_SYNC_TYPE MD22240 $MC_AUXFU_F_SYNC_TYPE MD22250 $MC_AUXFU_D_SYNC_TYPE MD22252 $MC_AUXFU_DL_SYNC_TYPE
最低	未定义	缺省输出特性：在程序段末尾输出

说明

不含轨迹运行的零件程序段

在不含轨迹运行的零件程序段中（同样针对定位轴和主轴），辅助功能立即在一个块中输出。

9.7 辅助功能编程

句法

辅助功能在零件程序段中通过以下句法编写：
<类型>[<地址扩展>=]<数值>

说明

若未编写地址扩展，则会隐性设置地址扩展 = 0。
地址扩展 = 0 的预定义辅助功能总是基于通道的主主轴。

符号定址

“地址扩展”和“值”参数的值也可通过符号设定。地址扩展的符号名称必须在方括号中设定。

示例：

针对第 1 主轴的辅助功能 M3（主轴正转）的符号编程：

程序代码	注释
DEF SPINDEL_NR=1	; 通道中的第 1 主轴
DEF 旋转方向=3	; 旋转方向向右
N100 M[主轴号]=旋转方向	; 相应地：M1=3

说明

使用符号名称编写辅助功能的情形下，该辅助功能输出至 PLC 时不传输符号名称，而是对应的数值。

示例

示例 1：预定义辅助功能的编程

程序代码	注释
N10 M3	; “主轴正转”，针对通道主主轴。
N20 M0=3	; “主轴正转”，针对通道主主轴。
N30 M1=3	; “主轴正转”，针对通道第 1 主轴。
N40 M2=3	; “主轴正转”，针对通道第 2 主轴。

示例 2：辅助功能编程示例，对应值输出至 PLC

程序代码	注释
DEF 冷却剂=12	; 输出至 PLC：- - -
DEF 润滑剂=130	; 输出至 PLC：- - -
H[冷却剂]=润滑剂	; 输出至 PLC：H12=130
H=冷却剂	; 输出至 PLC：H0=12
H5	; 输出至 PLC：H0=5
H=5.379	; 输出至 PLC：H0=5.379
H17=3.5	; 输出至 PLC：H17=3.5
H[冷却剂]=13.8	; 输出至 PLC：H12=13.8
H='HFF13'	; 输出至 PLC：H0=65299
H='B1110'	; 输出至 PLC：H0=14
H5.3=21	; 错误

9.8 可编程的输出时长

功能

对于设置了“输出持续一个 OB1 周期（慢速应答）”输出特性的用户专用辅助功能，可通过零件程序指令 QU（Quick）将其定义为采用快速应答的辅助功能，用于单次输出。

句法

可在零件程序段中通过以下句法定义采用快速应答的辅助功能：
<类型>[<地址扩展>]=QU(<数值>)

示例

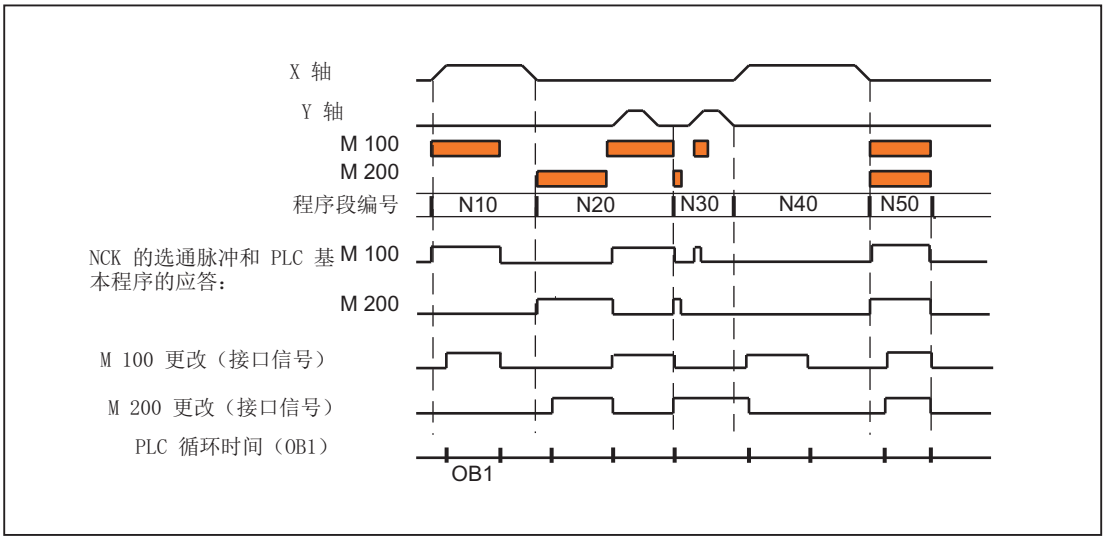
在一个零件程序段中辅助功能 M100 和 M200 有不同输出特性。辅助功能的输出特性如下设置：

- M100
  - 输出持续一个 OB1 周期（慢速应答）
  - 运行中输出
- M200
  - 输出持续一个 OB1 周期（慢速应答）
  - 运行前输出

程序代码	注释
N10 G94 G01 X50 M100	; 输出 M100：运动期间
	; 应答：慢
N20 Y5 M100 M200	; 输出 M200：运动前
	; 输出 M100：运动期间
	; 应答：慢
N30 Y0 M=QU(100) M=QU(200)	; 输出 M200：运动前
	; 输出 M100：运动期间
	; 应答：快
N40 X0	
N50 M100 M200	; 输出 M200：立即 1)
	; 输出 M100：立即 1)
	; 应答：慢
M17	

1) 不包含运行时，辅助功能总是即时输出至 PLC。

下图显示了零件程序的时间顺序。特别需要注意的是执行零件程序段 N20 和 N30 时的时间差。



9.9 输出至 PLC 的辅助功能

功能

在将辅助功能输出至 PLC 时，下列信号和数值会传输至 NC/PLC 接口：

- 变更信号
- “地址扩展”参数
- “值”参数

NC/PLC 接口中的数据区域

变更信号和辅助功能的数值位于 NC/PLC 接口的以下数据区域中：

- 从 NC 通道传输辅助功能时的变更信号：
  DB21, ... DBB58 - DBB67
- 传输的 M 功能和 S 功能：
  DB21, ... DBB68 - DBB112
- 传输的 T 功能、D 功能和 DL 功能：
  DB21, ... DBB116 - DBB136
- 传输的 H 功能和 F 功能：
  DB21, ... DBB140 - DBB190
- 解译的 M 信号（M0 - M99）：
  DB21, ... DBB194 - DBB206（动态 M 功能）

访问 NC/PLC 接口的相关信息请见“P3：SINUMERIK 840D sl 的 PLC 基本程序 (页 945)”章节。

对上述 NC/PLC 接口中数据区域的详细说明请见：

文档：
参数手册，第 2 册；PLC 用户接口，
章节：通道专用信号（DB 21 – DB 30）

9.10 无程序段切换延迟的辅助功能

功能

即便是对于设置和/或编写了以下输出特性的辅助功能：

- “输出持续一个 OB40 周期（快速应答）”
- “运行前输出”或“运行中输出”

在连续路径运行（行程短，速度高）中仍可能出现速度跃变，因为在到达程序段末尾前须等待 PLC 应答辅助功能。 为了避免此类速度跃变，可独立于辅助功能应答进行程序段切换。

参数设置

使用快速辅助功能时对程序段切换延迟的抑制通过以下机床数据设置：

MD22100 $MC_AUXFU_QUICK_BLOCKCHANGE（使用快速辅助功能时的程序段切换延迟）

值	含义
0	在辅助功能快速输出至 PLC 时，程序段切换延迟直至 PLC 应答（OB40）。
1	在辅助功能快速输出至 PLC 时，程序段切换不延迟。

前提条件

对于输出时无程序段延迟的辅助功能，无法保证该功能与其程序段之间的同步性。 在“最差”情形下，切换至下一零件程序段后应答还需一个 OB40 周期，辅助功能执行还需一个 OB1 周期。

9.11 带隐性预处理停止的 M 功能

功能

若需配合辅助功能触发预处理停止，可通过零件程序指令 STOPRE 显性编程。若需在 M 功能的编程中默认触发预处理停止，可通过以下机床数据设置此 M 功能专用特性：

MD10713 $MN_M_NO_FCT_STOPRE[<n>]（带预处理停止的 M 功能）

示例

用户自定义 M 功能 M88 需要触发预处理停止。

参数设置：

MD10713 $MN_M_NO_FCT_STOPRE [0] = 88

应用：

零件程序（节选）

程序代码	注释
...	
N100 G0 X10 M88	; 运行，并通过 M88 隐性触发预处理停止。
N110 Y=R1	; 在运行完成和 M 功能被应答后才会解译 N110。
...	

前提条件

若在零件程序中借由以下两种方式之一通过 M 功能调用了一个子程序，则不会触发预处理停止：

- MD10715 $MN_M_NO_FCT_CYCLE（须由子程序替换的 M 功能）
- M98（ISO 车铣编程语言）

9.12 溢出转存时的特性

溢出转存

通过 SINUMERIK 操作界面，可在以下功能开始前：

- 零件程序启动
- 恢复执行中断的零件程序

通过“溢出转存”功能修改启动时输出的辅助功能。

此功能的应用场合例如有：

- 在程序段搜索后添加辅助功能
- 恢复初始状态用于零件程序试运行

可溢出转存的辅助功能类型

下列辅助功能类型可被溢出转存：

- M（附加功能）
- S（主轴转速）
- T（刀具号）
- H（辅助功能）
- D（刀具补偿号）
- DL（总补偿）
- F（进给率）

生效时间

经过溢出转存的辅助功能（例如 M3，主轴正转）保持生效，直至其被同一辅助功能组的另一个辅助功能覆盖，或由于重新溢出转存或零件程序段中的编程被覆盖。

9.13 程序段搜索时的特性

9.13.1 程序段搜索类型 1、2 和 4 时的辅助功能输出

输出特性

在程序段搜索类型 1、2 和 4 中，系统会根据组收集辅助功能。 系统启动后，每个辅助功能组的最后一个辅助功能会在真正的重入程序段前在一个单独的零件程序段中输出，特性如下：

- 输出持续一个 OB1 周期（普通应答）
- 运行前输出

输出控制

通过以下机床数据的位 8，可设置在程序段搜索后是否向 PLC 输出辅助功能：

- MD22080 $MC_AUXFU_PREDEF_SPEC[<n>]
  （预定义辅助功能的输出特性）
  其中 <n> = 系统功能下标（0 ... 32）
- MD22035 $MC_AUXFU_ASSIGN_SPEC[<n>]
  （用户自定义辅助功能的输出特性）
  其中 <n> = 辅助功能下标（0 ... 254）
- MD11110 $MN_AUXFU_GROUP_SPEC[<n>]
  （组中辅助功能的输出特性）
  其中 <n> = 组下标（0 ... 63）

位	值	含义
10	0	在程序段搜索类型 1、2、4 中输出
	1	在程序段搜索类型 1、2、4 中不输出

此特性对显示或变量 $AC_AUXFU_STATE[<n>]、$AC_AUXFU_VALUE[<n>]、$AC_AUXFU_EXT[<n>] 无影响。
程序段搜索后辅助功能仍被作为“已收集”，尽管其未被输出至 PLC。

收集时，程序段搜索后未输出的辅助功能仍会覆盖位 8 未置位的辅助功能。

用户可在程序段搜索后查询收集的辅助功能，某些情形下还可通过零件程序或同步动作将其再次输出。

说明

下列辅助功能不会被收集：
- 未指定分组的辅助功能
- 被指定给第 1 辅助功能组的辅助功能

辅助功能的溢出转存

程序段搜索完成后，下一次系统启动时会输出收集的辅助功能。 若还需输出额外的辅助功能，可通过“溢出转存”功能添加（参见“溢出转存时的特性 (页 477)”章节）。

M19（主轴定位）相关特性

程序段预处理后，系统总是会执行最后使用 M19 编写的主轴定位，即便是在 M19 零件程序段和目标程序段间仍编写了其他主轴专用辅助功能。 因此必须在 PLC 用户程序中通过运行指令的接口信号导出所需主轴使能：

DB31, ... DBX64.6 / 64.7（运行指令 -/+）

主轴专用辅助功能 M3、M4、M5 不适用于此情形，因为有时在主轴定位后其才被输出至 PLC。

程序段搜索的详细信息请见“K1：BAG、通道、程序运行、复位特性 (页 519)”章节。

9.13.2 将一个辅助功能指定给多个组

功能

通过分组指定（MD22000 $MC_AUXFU_ASSIGN_GROUP）也可将用户自定义辅助功能指定给多个组。 程序段搜索时，系统会基于所涉及的所有功能组收集这些辅助功能。

说明

预定义辅助功能只能指定给一个功能组。

示例

在 DIN 标准中，下列 M 指令设计用于冷却剂输出：

- M7：冷却剂 2 ON
- M8：冷却剂 1 ON
- M9：冷却剂 1 和 2 OFF

若需使两组冷却剂能够同时生效：

- 必须在两个独立的功能组中收集 M7 和 M8（例如组 5 和组 6）
- 必须将 M9 同时指定给这两个组，例如：
  - 组 5：M7、M9
  - 组 6：M8、M9

参数设置：

MD11100 $MN_AUXFU_MAXNUM_GROUP_ASSIGN = 4

MD22000 $MC_AUXFU_ASSIGN_GROUP [0] = 5

MD22000 $MC_AUXFU_ASSIGN_GROUP [1] = 5

MD22000 $MC_AUXFU_ASSIGN_GROUP [2] = 6

MD22000 $MC_AUXFU_ASSIGN_GROUP [3] = 6

MD22010 $MC_AUXFU_ASSIGN_TYPE [0] = M

MD22010 $MC_AUXFU_ASSIGN_TYPE [1] = M

MD22010 $MC_AUXFU_ASSIGN_TYPE [2] = M

MD22010 $MC_AUXFU_ASSIGN_TYPE [3] = M

MD22020 $MC_AUXFU_ASSIGN_EXTENSION [0] = 0

MD22020 $MC_AUXFU_ASSIGN_EXTENSION [1] = 0

MD22020 $MC_AUXFU_ASSIGN_EXTENSION [2] = 0

MD22020 $MC_AUXFU_ASSIGN_EXTENSION [3] = 0

MD22030 $MC_AUXFU_ASSIGN_VALUE [0] = 7

MD22030 $MC_AUXFU_ASSIGN_VALUE [1] = 9

MD22030 $MC_AUXFU_ASSIGN_VALUE [2] = 8

MD22030 $MC_AUXFU_ASSIGN_VALUE [3] = 9

MD22035 $MC_AUXFU_ASSIGN_SPEC [0] = 'H121'

MD22035 $MC_AUXFU_ASSIGN_SPEC [1] = 'H121'

MD22035 $MC_AUXFU_ASSIGN_SPEC [2] = 'H121'

MD22035 $MC_AUXFU_ASSIGN_SPEC [3] = 'H121'

零件程序（节选）：

程序代码
... N10 ... M8 N20 ... M9 N30 ... M7 ...

程序段搜索时，系统会基于组 5 和组 6 收集辅助功能 M9。

查询收集到的 M 辅助功能：

第 5 组的 M 功能：$AC_AUXFU_M_VALUE [4] = 7

第 6 组的 M 功能：$AC_AUXFU_M_VALUE [5] = 9

9.13.3 生效的 M 辅助功能的时间戳

在程序段搜索后输出收集的辅助功能时，必须遵循收集时的顺序。 因此每个组都配有一个时间戳，其可通过以下系统变量针对组查询：

$AC_AUXFU_M_TICK[<n>]（生效的 M 辅助功能的时间戳）

9.13.4 确定输出顺序

功能

为了使编程人员方便地确定 M 功能输出顺序，特提供以下预定义程序：

```
AUXFUMSEQ(VAR INT _NUM_IN, VAR INT _M_IN[], VAR INT _EXT_IN[],
VAR INT _NUM_OUT, VAR INT _M_OUT[], VAR INT _EXT_OUT[])
```

输入参数：

`VAR INT _NUM_IN:`	相关 M 指令的数量
`VAR INT _M_IN[]:`	相关 M 代码的字段
`VAR INT _EXT_IN[]:`	相关 M 地址扩展的字段

输出参数：

`VAR INT _NUM_OUT:`	确定的 M 代码的数量
`VAR INT _M_OUT[]:`	确定的 M 代码的字段
`VAR INT _EXT_OUT[]:`	确定的 M 地址扩展的字段

此功能用于为给定的 M 代码确定针对组收集的 M 辅助功能的输出顺序。 顺序取决于收集时间点 $AC_AUXFU_M_TICK[<n>]（参见“生效的 M 辅助功能的时间戳 (页 482)”章节）。

一个特定的 M 代码只会被考虑一次，即便其归属于多个组。 若相关 M 指令的数量小于或等于 0，则会输出收集的所有 M 代码。 相关 M 指令的数量上限为 64。

示例

用于冷却剂输出的 M 指令：

- M7：冷却剂 2 ON
- M8：冷却剂 1 ON
- M9：冷却剂 1 和 2 OFF

分组指定：

- 组 5：M7、M9
- 组 6：M8、M9

零件程序（节选）：

程序代码

```
...
N10 ... M8
N20 ... M9
N30 ... M7
...
```

程序段搜索时，系统会基于组收集辅助功能。 每个辅助功能组的最后一个辅助功能会在程序段搜索结束后输出至 PLC：

- 组 5：M7
- 组 6：M9

若以 M7 → M9 的顺序输出，那么之后将无冷却剂生效。 但是冷却剂 2 需要在程序运行中生效。 因此 M 辅助功能正确的输出顺序需通过一个含预定义程序 AUXFUMSEQ(…) 的 ASUB 确定：

程序代码

```
DEF INT _I, _M_IN[3], _EXT_IN[3], _NUM_OUT, _M_OUT[2], _EXT_OUT[2]
_M_IN[0]=7 _EXT_IN[0]=0
_M_IN[1]=8 _EXT_IN[1]=0
_M_IN[2]=9 _EXT_IN[2]=0
AUXFUMSEQ(3,_M_IN,_EXT_IN,_NUM_OUT,_M_OUT,_EXT_OUT)
FOR _I = 0 TO _NUM_OUT-1
   M[_EXT_OUT[_I]]=_M_OUT[_I]
ENDFOR
```

9.13.5 抑制主轴专用辅助功能的输出

功能

在特定情形下（例如换刀时），程序段搜索时收集的主轴专用辅助功能可能不需要在动作程序段中输出，而是在随后的某个时间点上（例如换刀后）再输出。为此可抑制程序段搜索后主轴专用辅助功能的自动输出。此时可于稍后的时间点通过溢出转存功能或 ASUB 手动输出。

参数设置

通过以下机床数据抑制程序段搜索后主轴专用辅助功能的自动输出：

MD11450 $MN_SEARCH_RUN_MODE（程序段搜索后的特性）

位	值	含义
2	0	主轴专用辅助功能在动作程序段中输出
	1	抑制辅助功能在动作程序段中的输出。

系统变量

程序段搜索时，主轴专用辅助功能总是存储在下列系统变量中，与上文所述设置无关：

系统变量	说明	
$P_SEARCH_S [<n>]	收集的主轴转速	
	取值范围：	0 ... Smax
$P_SEARCH_SDIR [<n>]	收集的主轴旋转方向	
	取值范围：	3, 4, 5, -5, -19, 70
$P_SEARCH_SGEAR [<n>]	收集的主轴齿轮档 M 功能	
	取值范围：	40 ... 45
$P_SEARCH_SPOS [<n>]	收集的主轴位置	
	取值范围：	0 ... MD30330 $MA_MODULO_RANGE（模数区域的大小）
	或者	
	收集的行程	
	取值范围：	-100000000 ... 100000000

系统变量	说明	
$P_SEARCH_SPOSMODE [<n>]	收集的位置逼近模式	
	取值范围：	0 ... 5

随后需要输出主轴专用辅助功能时，可在一个 ASUB 中读取系统变量，并于动作程序段输出后进行输出：

DB21, ... DBX32.6 = 1（最后的动作程序段生效）

说明

系统变量 $P_S、$P_DIR 和 $P_SGEAR 的内容可能会在程序段搜索后由于同步而丢失。

ASUB、程序段搜索以及动作程序段的更多详细信息请见“K1：BAG、通道、程序运行、复位特性 (页 519)”章节。

示例

进行程序段搜索，抑制主轴专用辅助功能的输出，并在动作程序段输出后启动一个 ASUB。

参数设置：MD11450 $MN_SEARCH_RUN_MODE，位 2 = 1

在 N55 程序段搜索后启动 ASUB。

零件程序：

程序代码	注释
N05 M3 S200	; 主轴 1
N10 G4 F3	
N15 SPOS=111	; 在 ASUB 中将主轴 1 定位至 111 度
N20 M2=4 S2=300	; 主轴 2
N25 G4 F3	
N30 SPOS[2]=IC(77)	; 主轴 2 增量运行 77 度
N55 X10 G0	; 目标程序段
N60 G4 F10	
N99 M30	

ASUB：

程序代码	注释
PROC ASUP_SAVE	
MSG (“输出主轴功能”)	
DEF INT SNR=1	
AUSG_SPI:	
M[SNR]=$P_SEARCH_SGEAR[SNR]	; 输出齿轮档。

程序代码	注释
S[SNR]=$P_SEARCH_S[SNR]	; 输出转速（M40 时确定匹配的齿轮档）。
M[SNR]=$P_SEARCH_SDIR[SNR]	; 输出旋转方向、定位或进给轴模式。
SNR=SNR+1	; 下一主轴。
REPEAT AUSG_SPI P=$P_NUM_SPINDLES-1	; 针对所有主轴。
MSG("")	
REPOSA	
RET	

示例说明

若主轴数已知，可将相同类型的输出写入至一个零件程序段，从而减少程序处理时间。

$P_SEARCH_SDIR 的输出须在单独的零件程序段中进行，因为主轴定位或切换至进给轴模式与齿轮档切换一起时可能会触发报警。

若通过 REPOSA 使启动的 ASUB 结束，那么主轴 1 将留在 111 度的位置，而主轴 2 则将重新定位至 77 度的位置。

如需实现另一种特性，则须特别调整用于程序段搜索的程序序列，例如“N05 M3 S...”和“N30 SPOS[2]=IC(...)”。

可在 ASUB 中通过系统变量 $P_SEARCH 确定程序段搜索是否生效：
$P_SEARCH==1；程序段搜索生效

在转速控制运行后的增量定位中，虽然目标行程已定义，但是某些情形下要在定位过程中才能确定达到的最终位置。例如，在越过零标记时进行位置校准，或启用位置闭环控制时均属于此类情形。因此，从零位置开始的行程被用作 REPOS 位置（ASUB 中为 REPOSA）。

前提条件

收集的 S 值

零件程序中 S 值的含义取决于当前生效的进给率类型：

G93、G94、G95、G97、G971：	S 值为转速
G96、G961：	S 值为恒定切削速度

输出系统变量 $P_SEARCH_S 前，若进给率类型发生变化（例如用于换刀），必须将进给率类型恢复为零件程序目标程序段中的原始设置，以避免采用错误的进给率类型。

收集的旋转方向

输出旋转方向时，程序段搜索开始的时间点系统变量 $P_SEARCH_SDIR 的预赋值为“-5”。该数值在输出时无效。

因此，在对没有主轴编写主轴旋转方向、主轴定位或进给轴模式的的程序段进行搜索时，能保持最后生效的主轴运行模式。

M19、SPOS 和 SPOSA 编程在系统变量 $P_SEARCH_SDIR 中作为“M-19”（内部 M19）收集，以替代 M3、M4、M5 和 M70。

输出“M-19”时，系统会从系统变量 $P_SEARCH_SPOS 和 $P_SEACH_SPOSMODE 读取定位数据。这两个系统变量均可写入，例如用于进行补偿。

说明

对于上文介绍的指令（例如 M[<n>] = $P_SEARCH_SDIR[<n>]），值“–5”和“19”通常为用户不可见，仅在 ASUB 中对系统变量进行特殊分析时才需查看。

9.13.6 使用程序段搜索类型 5（SERUPRO）时的辅助功能输出

输出特性

使用程序段搜索类型 5（SERUPRO）时，辅助功能可在程序段搜索期间输出至 PLC，并/或按分组被收集到以下系统变量中：

- $AC_AUXFU_PREDEF_INDEX[<n>]（预定义辅助功能的下标）
- $AC_AUXFU_TYPE[<n>]（辅助功能的类型）
- $AC_AUXFU_STATE[<n>]（辅助功能的输出状态）
- $AC_AUXFU_EXT[<n>]（辅助功能的地址扩展）
- $AC_AUXFU_VALUE[<n>]（辅助功能的数值）

有关系统变量的描述请见 “查询系统变量 (页 501)” 一章。

输出控制

通过配置以下机床数据的位 9 和位 10，可定义是否在程序段搜索类型 5（SERUPRO）期间将辅助功能输出至 PLC，按分组被收集到上述系统变量中：

- MD22080 $MC_AUXFU_PREDEF_SPEC[<n>]
  （预定义辅助功能的输出特性）
  其中 <n> = 系统功能下标（0 ... 32）
- MD22035 $MC_AUXFU_ASSIGN_SPEC[<n>]
  （用户自定义辅助功能的输出特性）
  其中 <n> = 辅助功能下标（0 ... 254）
- MD11110 $MN_AUXFU_GROUP_SPEC[<n>]
  （组中辅助功能的输出特性）
  其中 <n> = 组下标（0 ... 63）

位	值	含义
9	0	在程序段搜索类型 5（SERUPRO）期间不收集
	1	在程序段搜索类型 5（SERUPRO）期间收集
10	0	在程序段搜索类型 5（SERUPRO）期间输出
	1	在程序段搜索类型 5（SERUPRO）期间不输出

输出计数器

用户可逐个通道地在程序段搜索 ASUB 中将收集的辅助功能输出至 PLC。为了实现多个通道的连续输出，每次输出辅助功能时三个输出计数器均会跨通道变化：

系统变量	含义		
$AC_AUXFU_TICK[<n>,<m>]	生效辅助功能的输出计数器		
	下标	含义	
	<n>	组下标（0 … 63）	
	<m>	输出计数器（0 ... 2）	
		值	含义
		0	输出序列计数器（所有输出在一个插补周期内进行）
		1	插补周期中一个输出序列内的包计数器
		2	一个包内的辅助功能计数器

说明

- 一个辅助功能包可包含最多 10 个辅助功能。
- SERUPRO 期间每个通道在每个插补周期中可执行两个包，因为同步动作按该周期处理。
- 在一个插补周期内可跨所有通道执行一个最多含 20 个包的输出序列（每个通道 2 个包 * 10 通道）。

通过此编码可显示在同一个插补周期内执行了多少个辅助功能包和多少个辅助功能：

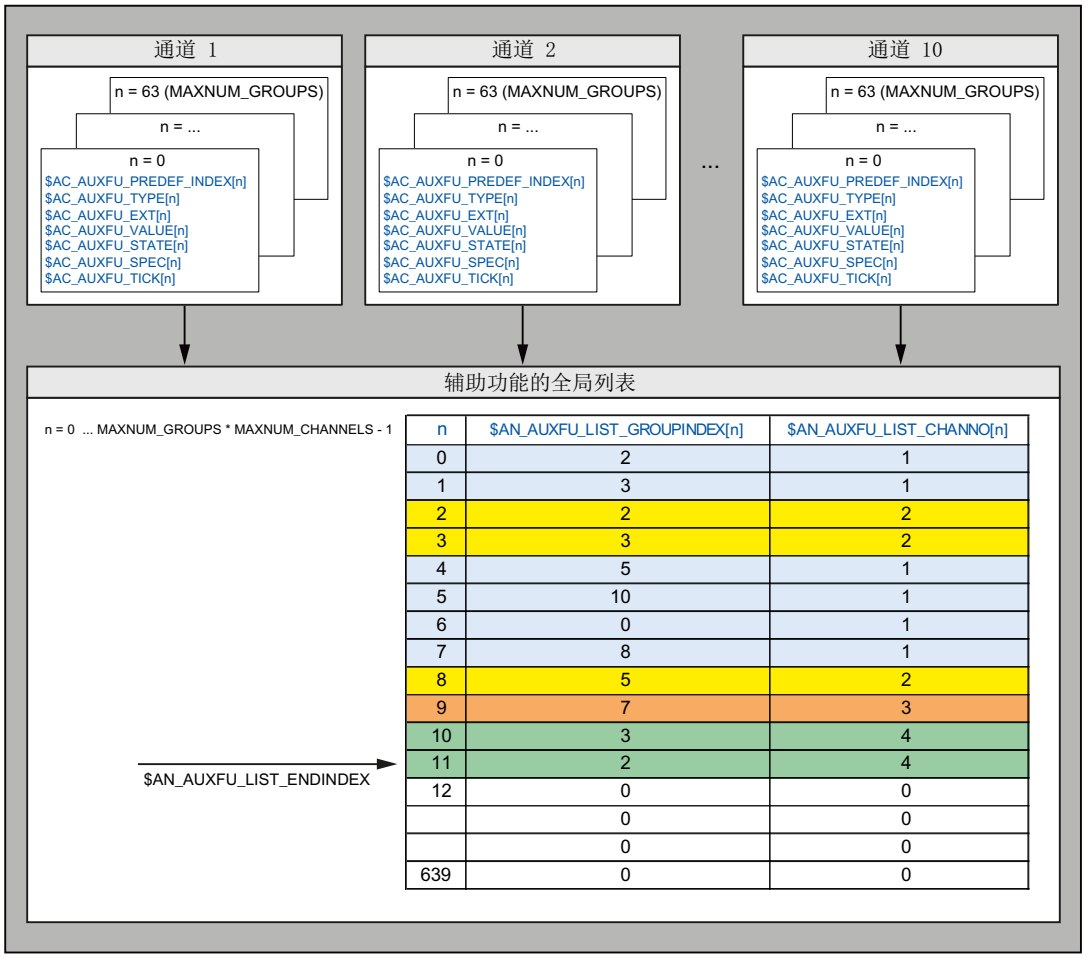
- 在一个插补周期内收集的辅助功能均有相同的序列计数器。
- 在一个包（程序段或同步动作）内收集的辅助功能均有相同的包计数器。

每收集到一个辅助功能，辅助功能计数器依次递增。

辅助功能的全局列表

SERUPRO 结束时，各通道中针对组收集的辅助功能会依据其计数器状态记录至一个跨通道列表中，并附注通道编号和组下标。

系统变量 1)	含义
$AC_AUXFU_TICK[<n>,<m>]	计数器状态
$AN_AUXFU_LIST_CHANNO[<n>] 2)	通道号
$AN_AUXFU_LIST_GROUPINDEX[<n>] 2)	组下标
1) 下标 <n> 的取值范围：0 ... MAXNUM_GROUPS * MAXNUM_CHANNELS - 1 2) 系统变量均可读写。	



n	$AN_AUXFU_LIST_GROUPINDEX[n]	$AN_AUXFU_LIST_CHANNO[n]
0	2	1
1	3	1
2	2	2
3	3	2
4	5	1
5	10	1
6	0	1
7	8	1
8	5	2
9	7	3
10	3	4
11	2	4
12	0	0
	0	0
	0	0
639	0	0

找到搜索目标后，系统便会建立全局列表。此列表作为对下一个 SERUPRO 末尾 ASUB 中待输出辅助功能的系统建议。若辅助功能不需要输出，那么必须将对应的组下标设置为“0”。

主轴辅助功能相关特性

程序段搜索启动后，所有通道都会在通道变量中针对组收集辅助功能。为了在 SERUPRO 目标程序段中通过收集的辅助功能尽可能恢复主轴状态，任意主轴辅助功能组中最后生效的辅助功能必须能代表目标程序段中的主轴状态。在主轴状态过渡时，失效的辅助功能会从列表中清除，或者会在必要时输入隐性辅助功能。

全局辅助功能列表的所有主轴辅助功能必须在目标程序段中对应所达到的主轴状态，确保列表输出时这些辅助功能可执行，且不会引起报警或不期望的主轴状态使零件程序的继续执行受到影响。

相应地，对于系统中配置的每根主轴的辅助功能组，其主轴编号须对应辅助功能的地址扩展。

组 a： M3、M4、M5、M19、M70

组 b： M40、M41、M42、M43、M44、M45

组 c： S

清除失效的辅助功能

使用以下功能时，系统会为相关主轴清除组 a 的辅助功能：

- 对跟随主轴激活通用耦合时，如 COUPON、TRAILON、EGON 等

生成隐性组 a 辅助功能

使用以下功能时，系统会为相关主轴隐性生成组 a 的辅助功能：

- 对跟随主轴取消同步主轴耦合时
  - 根据耦合情形，COUPOF 会在主处理中生成 M3、M4、S 或 M5。
  - COUPOF(S<n>, S<m>, POS) 和 COUPOFS(S<n>, S<m>, POS, POS) 会生成 M3、M4 和 S。
  - COUPOFS 会在主处理中生成 M5。
  - COUPOFS(S<n>, S<m>, POS) 会在主处理中生成 M19。
    隐性 M19（ASUB 中“SPOS[<地址扩展>] = IC(0)”）会激活无运行的定位模式。
- 在主轴作为进给轴运行，或者通过选择转换（其中主轴作为进给轴）过渡至进给轴模式时，系统会生成 M70。
- 使用 SPCOF 时会生成 M5。

说明

在将用于主轴编程的辅助功能与“跨通道取轴”或“轴容器旋转”功能一起使用时，必须确保其与交换/旋转时的实际状态（电机）相匹配。此时须区别跨通道取轴和轴容器的机制。

轴容器旋转示例：

一个轴容器包含 4 根主轴。每根主轴分别被指定给一个通道（1 - 4）。通道 1 中总是编写 M3 S1000，并执行轴容器旋转。其他通道不进行主轴编程。在第 3 次轴容器旋转和第 4 次主轴编程 M3 后，4 根主轴均以 1000 rpm 的速度顺时针旋转。若 SERUPRO 末尾于此区域，则视为每个通道的 ASUB 都包含用于该通道中主轴的 M3 S1000。

使用跨通道取轴时则须将收集的辅助功能指定给主轴实际处于的通道。

跨通道辅助功能

在程序段搜索类型 5（SERUPRO）中，辅助功能也可跨通道在全局辅助功能列表中收集。此时只会将该组最后收集的辅助功能（计数最高）记录至全局列表。

通过下列机床数据进行相应的配置：

- MD22080 $MC_AUXFU_PREDEF_SPEC[<n>]，位 11
  （预定义辅助功能的输出特性）
  其中 <n> = 系统功能下标
- MD22035 $MC_AUXFU_ASSIGN_SPEC[<n>]，位 11
  （用户自定义辅助功能的输出特性）
  其中 <n> = 辅助功能下标
- MD11110 $MN_AUXFU_GROUP_SPEC[<n>]，位 11
  （组中辅助功能的输出特性）
  其中 <n> = 组下标

位	值	含义
11	0	针对特定通道收集
	1	跨通道收集

根据主轴状态，主轴辅助功能会在程序段搜索结束时提前滤除。通道数据会相应更新。全局辅助功能列表可在 SERUPRO 末尾 ASUB 中按顺序执行，经过排序的辅助功能可同步于通道输出。

查询最后收集的辅助功能

通过系统变量 $AN_AUXFU_LIST_ENDINDEX 可查询全局列表中最后收集的辅助功能的下标。

9.13.7 SERUPRO 末尾 ASUB

功能

带程序测试的程序段搜索（SERUPRO）完成后，在下一步处理开始前必须将程序段搜索中收集的辅助功能输出。为此，程序段搜索期间系统会在一个全局列表中收集辅助功能。SERUPRO 末尾的 ASUB 从此列表针对通道生成对应的零件程序段。这样可确保收集的辅助功能针对特定通道或跨通道输出时的时间顺序正确。完整功能的 SERUPRO 末尾 ASUB 是 NC 软件的组成部分。

SERUPRO 末尾 ASUB 可由用户/机床制造商修改。下面介绍的功能用于支持对辅助功能全局列表的处理，以及生成同步辅助功能输出所需的零件程序段。

AUXFUSYNC(...) 功能

功能：

AUXFUSYNC 功能能够在每次调用时从辅助功能全局列表以字符串形式生成一个完整的零件程序段。该零件程序段包含辅助功能，或者用于对辅助功能输出进行同步的指令（WAITM、G4 等）。

此功能会触发预处理停止。

句法：

```
PROC AUXFUSYNC(VAR INT <NUM>, VAR INT <GROUPINDEX>[10], VAR
STRING[400] <ASSEMBLED>)
```

参数：

<NUM>： **<**ASSEMBLED**>** 参数中提供的零件程序段的信息，或程序段中包含的辅助功能的信息。

取值范围：-1, 0, 1 ... 10

值	含义
≥1	零件程序段中包含的辅助功能的数量
0	无辅助功能的零件程序段，例如 WAITM、G4
-1	结束标识。针对当前通道，辅助功能全局列表已完全处理。

<GROUPINDEX>： 零件程序段中包含的辅助功能组的下标。其中下标 = 辅助功能组编号 - 1

<ASSEMBLED>： 字符串形式的完整零件程序段，用于针对特定通道的 SERUPRO 结束处 ASUB。

更多详细信息：

若辅助功能通过一个同步动作收集，则会生成两个 NC 程序段。一个 NC 程序段用于输出辅助功能。另一个可执行 NC 程序段，用于将输出辅助功能的 NC 程序段传输至主处理：

1. 通过同步动作输出辅助功能，例如：WHEN TRUE DO M100 M102
2. 可执行 NC 程序段，例如：G4 F0.001

AUXFUDEL(...) 功能

功能：

AUXFUDEL 功能可为调用通道将指定的辅助功能从全局列表删除。删除通过将对应的组索引 ...GROUPINDEX[n] 置 0 来实现。

此功能必须在调用 AUXFUSYNC 前调用。

此功能会触发预处理停止。

句法：

```
PROC AUXFUDEL(CHAR <TYPE>, INT <EXTENSION>, REAL <VALUE>, INT
<GROUP>)
```

参数：

<TYPE>：	待删除的辅助功能的类型
<EXTENSION>：	待删除的辅助功能的地址扩展
<VALUE>：	待删除的辅助功能的数值
<GROUP>：	辅助功能组的编号

AUXFUDELG(...) 功能

功能：

AUXFUDELG 功能可为调用通道将指定辅助功能组中的所有辅助功能从全局列表删除。删除通过将对应的组索引 ...GROUPINDEX[n] 置 0 来实现。

此功能必须在调用 AUXFUSYNC 前调用。

此功能会触发预处理停止。

句法：

```
PROC AUXFUDELG(INT <GROUP>)
```

参数：

<GROUP>：	辅助功能组的编号

多通道程序段搜索

注意
多通道程序段搜索和 AUXFUDEL / AUXFUDELG 在进行多通道程序段搜索时，若在 SERUPRO 末尾 ASUB 中通过 AUXFUDEL / AUXFUDELG 将辅助功能从全局列表中删除，那么在调用 AUXFUSYNC 功能前必须对所涉及的通道进行同步。通过同步可确保调用 AUXFUSYNC 功能前所有删除任务均被处理，并确保列表的一致性。

示例

以下为两个建立用户专用 SERUPRO 末尾 ASUB 的示例。

示例 1：删除辅助功能，并通过 AUXFUSYNC(...) 生成辅助功能输出

```
N10 DEF STRING[400] ASSEMBLED=""
N20 DEF STRING[31] FILENAME="/_N_CST_DIR/_N_AUXFU_SPF"
N30 DEF INT GROUPINDEX[10]
N40 DEF INT NUM
N60 DEF INT ERROR
N90
N140 AUXFUDEL("M",2,3,5)                          ; 删除 M2=3（第 5 辅助功能组）
N150
N170 AUXFUDELG(6)                                 ; 删除收集的
                                                  ; 删除第 6 组。
N180
N190 IF ISFILE(FILENAME)
N210    DELETE(ERROR,FILENAME)                    ; 删除 FILENAME 文件
N220    IF (ERROR<>0)                             ; 错误分析
N230       SETAL(61000+ERROR)
N240    ENDIF
N250 ENDIF
; 注意!
; 在多通道程序段搜索时，若通过 AUXFUDEL/AUXFUDELG
; 从全局列表删除辅助功能，那么必须在
; 通过 AUXFUSYNC 生成子程序 FILENAME 的 LOOP 前进行通道同步，
; 通过同步可确保所有通道中的所有删除任务
; 均被处理，并确保列表的一致性。
; 示例：WAITM(99,1,2,3)
N270 LOOP
N300    AUXFUSYNC(NUM,GROUPINDEX,ASSEMBLED)       ; 生成零件程序段
N310
```

程序代码	注释
N320 IF (NUM==-1)	; 通道的所有辅助功能
	; 均已执行。
N340 GOTOF LABEL1	
N350 ENDIF	
N380 WRITE(ERROR,FILENAME,ASSEMBLED)	; 将零件程序段写入 FILENAME 文件。
N390 IF (ERROR<>0)	; 错误分析
N400 SETAL(61000+ERROR)	
N410 ENDIF	
N430 ENDLOOP	
N440	
N450 LABEL1:	
N460	
N480 CALL FILENAME	; 执行生成的子程序。
N490	
N510 DELETE(ERROR,FILENAME)	; 执行后再次删除文件。
N520 IF (ERROR<>0)	
N530 SETAL(61000+ERROR)	
N540 ENDIF	
N550	
N560 M17	

示例 2：删除辅助功能，不通过 AUXFUSYNC(...) 生成辅助功能输出

程序代码	注释
N0610 DEF STRING[400] ASSEMBLED=""	
N0620 DEF STRING[31] FILENAME="/_N_CST_DIR/_N_AUXFU_SPF"	
N0630 DEF INT GROUPINDEX[10]	
N0640 DEF INT NUM	
N0650 DEF INT LAUF	
N0660 DEF INT ERROR	
N0670 DEF BOOL ISQUICK	
N0680 DEF BOOL ISSYNACT	
N0690 DEF BOOL ISIMPL	
...	
N0760 AUXFUDEL("M",2,3,5)	; 删除 M2=3（第 5 辅助功能组）
N0770	
N0790 AUXFUDELG(6)	; 删除收集的
	; 删除第 6 组。
N0800	
N0810 IF ISFILE(FILENAME)	
N0830 DELETE(ERROR,FILENAME)	; 文件已存在，必须
	; 删除。
N0840 IF (ERROR<>0)	

程序代码	注释

```
N0850       SETAL(61000+ERROR)
N0860    ENDIF
N0870 ENDIF
N0880
; 注意!
; 在多通道程序段搜索时，若通过 AUXFUDEL/AUXFUDELG
; 从全局列表删除辅助功能，那么必须在
; 通过 AUXFUSYNC 生成子程序 FILENAME 的 LOOP 前进行通道同步，
; 通过同步可确保所有通道中的所有删除任务
; 均被处理，并确保列表的一致性。
; 示例：WAITM(99,1,2,3)
N0890 LOOP
N0920   AUXFUSYNC(NUM,GROUPINDEX,ASSEMBLED)          ; 从全局辅助功能列表
                                                     ; 生成辅助功能程序段
                                                     ; 的步骤。
N0930
N0940   IF (NUM==-1)                                 ; 通道的所有辅助功能
                                                     ; 均已执行。
N0960     GOTOF LABEL1
N0970   ENDIF
N0980
N1000   IF (NUM>0)                                   ; 输出辅助功能时
                                                     ; 生成程序段。
N1010
N1020     ASSEMBLED=""
N1030
N1050     FOR LAUF=0 TO NUM-1                        ; 对一个程序段收集的
                                                     ; 辅助功能。
N1060
N1080       IF GROUPINDEX[LAUF]<>0                   ; 从全局列表删除的
                                                     ; 辅助功能的组下标为 0。
N1090
N1100         ISQUICK=$AC_AUXFU_SPEC[GROUPINDEX[LAUF]] BAND'H2'
N1110
N1120         ISSYNACT=$AC_AUXFU_SPEC[GROUPINDEX[LAUF]] BAND'H1000'
N1130
N1140         ISIMPL=$AC_AUXFU_SPEC[GROUPINDEX[LAUF]] BAND'H2000'
N1150
N1180         IF ISSYNACT                            ; 编写用于输出 M 辅助功能
                                                     ; 的程序段
N1190           ASSEMBLED= ASSEMBLED << "WHEN TRUE DO "
N1200         ENDIF
N1210 ; 隐性生成的 M19 映射至 SPOS[SPI(<主轴编号>)] = IC(0) 。
N1230         IF (ISIMPL AND ($AC_AUXFU_VALUE[GROUPINDEX[LAUF]==19))
```

程序代码	注释

```
N1240           ASSEMBLED= ASSEMBLED << "SPOS[SPI(" <<
                  $AC_AUXFU_EXT[GROUPINDEX[LAUF]] << ")=IC(0)"
N1260         ELSE
N1270           ASSEMBLED= ASSEMBLED << "M[" << $AC_AUXFU_EXT[GROUPINDEX[LAUF]] << "]="
N1280
N1290           IF ISQUICK
N1300             ASSEMBLED= ASSEMBLED << "QU("
N1310           ENDIF
N1320
N1330           ASSEMBLED= ASSEMBLED << $AC_AUXFU_VALUE[GROUPINDEX[LAUF]]
N1340
N1350           IF ISQUICK
N1360             ASSEMBLED= ASSEMBLED << ")"
N1370           ENDIF
N1380         ENDIF
N1400       ENDIF
N1420     ENDFOR
N1430
N1450     WRITE(ERROR,FILENAME,ASSEMBLED)            ; 将辅助功能程序段写入文件。
N1460
N1470     IF ISSYNACT
N1480       ASSEMBLED="G4 F0.001"
N1490       WRITE(ERROR,FILENAME,ASSEMBLED)
N1500     ENDIF
N1510
N1520   ELSE
N1540     WRITE(ERROR,FILENAME,ASSEMBLED)            ; 将辅助功能程序段写入文件。
N1550   ENDIF
N1560
N1570 ENDLOOP
N1580
N1590 LABEL1:
N1600
N1620 CALL FILENAME                                  ; 执行生成的子程序。
N1630
N1650 DELETE(ERROR,FILENAME)                         ; 执行后再次删除文件。
N1660 IF (ERROR<>0)
N1670   SETAL(61000+ERROR)
N1680 ENDIF
N1690
N1700 M17
```

9.14 隐性输出的辅助功能

功能

隐性输出的辅助功能是指未显性编写，而通过其他系统功能（例如转换选择、刀具选择等）输出的辅助功能。 此类隐性辅助功能不会引起系统功能；根据为其参数设置的输出特性，系统会收集 M 代码并/或输出至 PLC。

参数设置

隐性输出的辅助功能的 M 代码通过以下机床数据确定：

- MD22530 $MC_TOCARR_CHANGE_M_CODE（刀架切换时的 M 代码）
  此机床数据的值指示的是激活刀架时 NC/PLC 接口上输出的 M 代码的编号。
  若该值为正值，那么输出的是未经变化的 M 代码。
  若改值为负值，则会将刀架编号累加至机床数据的绝对值，并输出此编号。
- MD22532 $MC_GEOAX_CHANGE_M_CODE（几何轴切换时的 M 代码）
  几何轴切换时，NC/PLC 接口上输出的 M 代码的编号。
- MD22534 $MC_TRAFO_CHANGE_M_CODE（转换切换时的 M 代码）
  几何轴的转换切换时，NC/PLC 接口上输出的 M 代码的编号。

说明

若待输出的 M 代码的编号或 MD22530 / MD22532 / MD22534 的值本身为 0 至 6、17 或 30，则不输出 M 代码。对于所生成的 M 代码是否会与其它功能冲突，系统不进行监控。

输出特性

对于隐性输出的辅助功能，机床数据 MD22080 或 MD22035（预定义或用户自定义辅助功能的输出特性）的位 13 置位。

可通过系统变量 $AC_AUXFU_SPEC[<n>] 查询此位。

隐性输出的辅助功能 M19

为了使 M19 和 SPOS/SPOSA 在 NC/PLC 接口上的特性方面保持一致，可在使用 SPOS 和 SPOSA 时将辅助功能 M19 输出至 NC/PLC 接口（参见“一般功能 (页 1408)”章节）。

程序段搜索时会收集隐性输出的辅助功能 M19。

9.15 获取信息

辅助功能的相关信息（例如输出状态）可通过下列方式获取：

- 操作界面上针对组的模态 M 辅助功能显示。
- 在零件程序和同步动作中查询系统变量。

9.15.1 针对组的模态 M 辅助功能显示

功能

在操作界面上可针对组显示 M 辅助功能的输出状态和应答状态。

前提条件

对 M 辅助功能实现以功能为取向的应答和显示的前提条件是：对辅助功能的管理在 PLC 中进行，也就是在用户程序中进行。因此 PLC 编程人员需负责对这些辅助功能进行应答。其必须了解需要应答的辅助功能在哪些组中。

标准

不通过 PLC 管理的 M 辅助功能由 NC 输出至 PLC，并标记为“已传输”。对此类辅助功能无功能应答。此外程序段搜索后收集的所有 M 辅助功能均会被显示，从而告知操作人员启动后会输出哪些辅助功能。

PLC 相关事项

对于由 PLC 自身管理的辅助功能组，**接收**和**功能结束**时 PLC 用户程序必须对这些组的所有辅助功能进行应答。PLC 编程人员必须知晓这些组的所有辅助功能。

其它

M 辅助功能只**针对组**显示。此外也保留逐段显示。最多可显示 15 个组，其中每个组总是显示该组**最后收集或输出至 PLC 的 M 功能**。根据其状态，M 功能会以多种方式显示：

状态	显示方式
辅助功能已被收集	反色，黄色字体
辅助功能已由 NC 输出至 PLC	反色

状态	显示方式
辅助功能已由 NC 传输至 PLC，并进行传输应答	黑色字体，灰色背景
辅助功能由 PLC 管理，且已由 PLC 直接接收。	黑色字体，灰色背景
辅助功能由 PLC 管理，且已进行功能应答。	黑色字体，灰色背景

显示更新

显示总是先输出收集到的辅助功能，其次是由 PLC 管理和由 NC 管理的辅助功能。收集到的辅助功能始终标记为“已收集”，直至 NC 将其输出至 PLC。PLC 管理的辅助功能一直保留，直至其被其他辅助功能取代。复位时只会删除收集的和 NC 管理的辅助功能。

9.15.2 查询系统变量

功能

在零件程序和同步动作中，可通过系统变量针对组查询辅助功能：

$AC_AUXFU_... [<n>] = <数值>

系统变量	含义		
$AC_AUXFU_PREDEF_INDEX[<n>]	<数值>：	一个辅助功能组中收集（程序段搜索）或输出的最后一个预定义辅助功能的下标	
		类型：	INT
		若指定组中尚未输出辅助功能，或者辅助功能为用户自定义辅助功能，那么该变量会输出“-1”值。	
	<n>：	组下标（0 … 63）	
	提示： 通过此系统变量可确定唯一的预定义辅助功能。		
$AC_AUXFU_TYPE[<n>]	<数值>：	一个辅助功能组中收集（程序段搜索）或输出的最后一个辅助功能的类型	
		类型：	CHAR
	<n>：	组下标（0 … 63）	

系统变量	含义		
$AC_AUXFU_EXT[<n>] 或者 M 功能专用： $AC_AUXFU_M_EXT[<n>]	<数值>：	一个辅助功能组中收集（程序段搜索）或输出的最后一个辅助功能的地址扩展	
		类型：	INT
	<n>：	组下标（0 … 63）	
$AC_AUXFU_VALUE[<n>] 或者 M 功能专用： $AC_AUXFU_M_VALUE[<n>]	<数值>：	一个辅助功能组中收集（程序段搜索）或输出的最后一个辅助功能的数值	
		类型：	REAL
	<n>：	组下标（0 … 63）	
$AC_AUXFU_SPEC[<n>]	值：	一个辅助功能组中收集（程序段搜索）或输出的最后一个辅助功能的位编码输出特性，对应 MD22080/MD22035（或 QU 编程）	
		类型：	INT
	<n>：	组下标（0 … 63）	
	提示： 通过此变量可确定是否进行采用快速应答的辅助功能输出。		

系统变量	含义		
$AC_AUXFU_STATE[<n>] 或者 M 功能专用： $AC_AUXFU_M_STATE[<n>]	<数值>：	一个辅助功能组中收集（程序段搜索）或输出的最后一个辅助功能的输出状态	
		类型：	INT
		取值范围：	0 ... 5
		0： 辅助功能不存在	
		1： M 辅助功能已通过程序段搜索收集	
		2： M 辅助功能已输出至 PLC	
		3： M 辅助功能已输出至 PLC，且已进行传输应答	
		4： M 辅助功能由 PLC 管理，且已由 PLC 接收	
		5： M 辅助功能由 PLC 管理，且已进行功能应答	
	<n>：	组下标（0 … 63）	

示例

需要按照输出顺序对第 1 组的所有 M 辅助功能进行存储：

```
id=1 every $AC_AUXFU_M_STATE[0]==2 do $AC_FIFO[0,0]=
$AC_AUXFU_M_VALUE[0]
```

文档

系统变量的更多相关信息请见：

参数手册 系统变量

9.16 前提条件

9.16.1 一般前提条件

跨通道取主轴

对辅助功能的参数设置针对通道进行，因此在使用“跨通道取主轴”功能时，所有使用主轴的通道中主轴专用辅助功能的设置必须相同。

刀具管理

刀具管理生效时，以下前提条件适用：

- T 功能和 M<k> 功能不输出至 PLC。
  提示
  k 是参数设置的换刀辅助功能的值（缺省设置：6）：
  MD22560 $MC_TOOL_CHANGE_M_CODE（换刀辅助功能）
- 若未编写地址扩展，那么辅助功能基于通道的主主轴或主刀架。
  定义主主轴：
  - MD20090 $MC_SPIND_DEF_MASTER_SPIND
  - 零件程序指令：SETMS

  定义主刀架：
  - MD20124 $MC_TOOL_MANAGEMENT_TOOLHOLDER
  - 零件程序指令：SETMTH

每个零件程序段中辅助功能的最大数量

一个零件程序段中最多可以编写 10 个辅助功能。

DL（总补偿）

对 DL 功能有以下限制：

- 每个零件程序段中只可编写一个 DL 功能。
- 在同步动作中使用 DL 功能时，“值”参数不输出至 PLC。

9.16.2 输出特性

螺纹切削

螺纹切削 G33、G34 和 G35 生效时，主轴专用辅助功能

- M3（主轴顺时针旋转）
- M4（主轴逆时针旋转）

总是采用以下输出特性：

- 输出持续一个 OB40 周期（快速应答）
- 运行中输出

主轴专用辅助功能 M5（主轴停止）总是在程序段末尾输出。 也就是说，包含 M5 的零件程序段总是以准停结束，即便是在连续路径运行生效的情况下。

同步动作

在从同步动作输出辅助功能时，除以下参数外，设置的输出特性均忽略：

- 位 0：输出持续一个 OB1 周期（普通应答）
- 位 1：输出持续一个 OB40 周期（快速应答）

辅助功能：M17 或 M2 / M30（子程序结束）

编写在单独的零件程序段中

若辅助功能 M17、M2 或 M30 被编写在单独的程序段中，且尚有轴在运动，那么在轴停止后辅助功能才会输出至 PLC。

参数设置的输出特性的叠加

为辅助功能 M17 或 M2/M30 设置的输出特性会与以下机床数据中定义的输出特性叠加：

MD20800 $MC_SPF_END_TO_VDI，位 0（子程序结束/停止输出至 PLC）

位	值	含义
0	0	辅助功能 M17 或 M2/M30（子程序结束）不输出至 PLC。 子程序结束处连续路径运行不中断。
	1	辅助功能 M17 或 M2/M30（子程序结束）输出至 PLC。

辅助功能：M1（有条件停止）

参数设置的输出特性的叠加

为辅助功能 M1 设置的输出特性会与以下机床数据中定义的输出特性叠加：

MD20800 $MC_SPF_END_TO_VDI，位 1（子程序结束/停止输出至 PLC）

位	值	含义
1	0	辅助功能 M01（有条件停止）总是输出至 PLC。 快速应答无效，因为 M01 固定归属于第 1 辅助功能组并始终在程序段末尾输出。
	1	仅在通过 HMI 操作界面激活了“编程的停止”时，辅助功能 M01（有条件停止）才输出至 PLC。 采用快速应答时，M1 在运行中输出至 PLC。 只要此功能不生效，连续路径运行便不会中断。

不含运行的零件程序段

在不含运行的零件程序段中，所有辅助功能均立即在一个块中输出，不考虑参数设置的输出特性。

主轴专用辅助功能输出只作为提供给 PLC 用户程序的信息

在特定控制情形下，例如为了完成程序段搜索，系统会将收集到的主轴专用辅助功能（例如 M3、M4、M5、M19、M40...M45、M70）输出至 NC/PLC 接口，而只是作为提供给 PLC 用户程序的信息。 为此控制系统会生成一个零件程序段（动作程序段），其中记录收集的辅助功能，其地址扩展**取负值**。 这样一来便**不会**执行对应的系统功能。

示例：M(-2) = 41，为第 2 主轴请求齿轮档切换

9.17 示例

9.17.1 预定义辅助功能的扩展

任务

为通道的第 2 主轴设置辅助功能 M3、M4 和 M5。

参数设置：M3

要求：

- 机床数据下标：0（第一个用户自定义辅助功能）
- 辅助功能组：5
- 类型和数值：M3（主轴顺时针旋转）
- 地址扩展：2，对应通道第 2 主轴
- 输出特性：
  - 输出持续一个 OB1 周期（普通应答）
  - 运行前输出

参数设置：

```
MD22000 $MC_AUXFU_ASSIGN_GROUP [ 0 ]      = 5
MD22010 $MC_AUXFU_ASSIGN_TYPE [ 0 ]       = "M"
MD22020 $MC_AUXFU_ASSIGN_EXTENSION [ 0 ]= 2
MD22030 $MC_AUXFU_ASSIGN_VALUE [ 0 ]      = 3
MD22035 $MC_AUXFU_ASSIGN_SPEC [ 0 ]       = 'H21'
```

参数设置：M4

要求：

- 机床数据下标：1（第二个用户自定义辅助功能）
- 辅助功能组：5
- 类型和数值：M4（主轴逆时针旋转）
- 地址扩展：2，对应通道第 2 主轴
- 输出特性：
  - 输出持续一个 OB1 周期（普通应答）
  - 应答后进行主轴响应
  - 运行中输出

参数设置：

```
MD22000 $MC_AUXFU_ASSIGN_GROUP [ 1 ]      = 5
MD22010 $MC_AUXFU_ASSIGN_TYPE [ 1 ]       = "M"
MD22020 $MC_AUXFU_ASSIGN_EXTENSION [ 1 ]= 2
```

MD22030 $MC_AUXFU_ASSIGN_VALUE [1] = 4
MD22035 $MC_AUXFU_ASSIGN_SPEC [1] = 'H51'

参数设置：M5

要求：

- 机床数据下标：2（第三个用户自定义辅助功能）
- 辅助功能组：5
- 类型和数值：M5（主轴停止）
- 地址扩展：2，对应通道第 2 主轴
- 输出特性：
  - 输出持续一个 OB1 周期（普通应答）
  - 应答后进行主轴响应
  - 在程序段末尾输出

参数设置：

MD22000 $MC_AUXFU_ASSIGN_GROUP [2] = 5
MD22010 $MC_AUXFU_ASSIGN_TYPE [2] = "M"
MD22020 $MC_AUXFU_ASSIGN_EXTENSION [2]= 2
MD22030 $MC_AUXFU_ASSIGN_VALUE [2] = 5
MD22035 $MC_AUXFU_ASSIGN_SPEC [2] = 'H91'

9.17.2 定义辅助功能

任务

针对采用以下配置的机床，对辅助功能专用机床数据进行参数设置：

主轴

- 主轴 1：主主轴
- 主轴 2：第二主轴

齿轮档

- 主轴 1：5 个齿轮档
- 主轴 2：无齿轮档

用于冷却水 ON/OFF 的开关功能

- 主轴 1
  - “ON”= M50
  - “OFF”= M51
- 主轴 2
  - “ON”= M52
  - “OFF”= M53

要求

主轴 1（主主轴）

说明

默认指定关系

- 主轴 1（主主轴）的辅助功能 M3、M4、M5、M70 和 M1=3、M1=4、M1=5、M1=70 默认指定给第 2 辅助功能组。
- 主轴 1（主主轴）的所有 S 值和 S1 值默认指定给第 3 辅助功能组。

- 程序段搜索后输出最后编写的齿轮档。为此将下列辅助功能指定给第 9 辅助功能组：
  - M40、M41、M42、M43、M44、M45
  - M1=40、M1=41、M1=42、M1=43、M1=44、M1=45
- 辅助功能 M3、M4、M5、M70 和 M1=3、M1=4、M1=5、M1=70 （第 2 辅助功能组）及 S 值和 S1 值（第 3 辅助功能组）采用以下输出特性：
  - 输出持续一个 OB40 周期（快速应答）
  - 运行前输出
- 用于齿轮档切换的辅助功能 M40 至 M45，M1=40 至 M1=45（第 9 辅助功能组）采用以下输出特性：
  - 输出持续一个 OB1 周期（普通应答）
  - 运行前输出

主轴 2

- 一个程序段中只允许编写一个用于切换旋转方向的 M 功能。程序段搜索后输出最后编写的旋转方向。为此将下列辅助功能指定给第 10 辅助功能组：
  - M2=3、M2=4、M2=5、M2=70
- 将所有 S2 值指定给第 11 辅助功能组。
- 辅助功能 M2=3、M2=4、M2=5、M2=70（第 10 辅助功能组）及 S2 值（第 11 辅助功能组）采用以下输出特性：
  - 输出持续一个 OB40 周期（快速应答）
  - 运行前输出

冷却水

- 不允许在一个零件程序段中接通和关闭。程序段搜索后将冷却水接通或关闭。为此将以下辅助功能指定给第 12 和第 13 辅助功能组：
  - 12.辅助功能组：M50, M51
  - 13.辅助功能组：M52, M53
- 辅助功能 M50、M51（第 12 辅助功能组）和 M52、M53（第 13 辅助功能组）采用以下输出特性：
  - 输出持续一个 OB1 周期（普通应答）
  - 运行前输出

机床数据的参数设置

对机床数据的参数设置通过零件程序段中的相应编程进行：

程序代码	注释
$MN_AUXFU_MAXNUM_GROUP_ASSIGN=21	; 每个通道用户自定义辅助功能的数量
$MN_AUXFU_GROUP_SPEC[1]='H22'	; 第 2 辅助功能组的输出特性
$MN_AUXFU_GROUP_SPEC[2]='H22'	; 第 3 辅助功能组的输出特性
$MN_AUXFU_GROUP_SPEC[8]='H21'	; 第 9 辅助功能组的输出特性
$MC_AUXFU_ASSIGN_TYPE[0]="M"	; 对辅助功能 1 的描述：M40
$MC_AUXFU_ASSIGN_EXTENSION[0]=0	
$MC_AUXFU_ASSIGN_VALUE[0]=40	
$MC_AUXFU_ASSIGN_GROUP[0]=9	
	; ...（辅助功能 2 至 5 以此类推）
$MC_AUXFU_ASSIGN_TYPE[5]="M"	; 对辅助功能 6 的描述：M45
$MC_AUXFU_ASSIGN_EXTENSION[5]=0	
$MC_AUXFU_ASSIGN_VALUE[5]=45	
$MC_AUXFU_ASSIGN_GROUP[5]=9	

程序代码	注释
$MC_AUXFU_ASSIGN_TYPE[6]="M"	; 对辅助功能 7 的描述：M1=40
$MC_AUXFU_ASSIGN_EXTENSION[6]=1	
$MC_AUXFU_ASSIGN_VALUE[6]=40	
$MC_AUXFU_ASSIGN_GROUP[6]=9	
	; . . .（辅助功能 8 至 11 以此类推）
$MC_AUXFU_ASSIGN_TYPE[11]="M"	; 对辅助功能 12 的描述：M1=45
$MC_AUXFU_ASSIGN_EXTENSION[11]=1	
$MC_AUXFU_ASSIGN_VALUE[11]=45	
$MC_AUXFU_ASSIGN_GROUP[11]=9	
$MN_AUXFU_GROUP_SPEC[9] = 'H22'	; 第 10 辅助功能组的输出特性
$MC_AUXFU_ASSIGN_TYPE[12]="M"	; 对辅助功能 13 的描述：M2=3
$MC_AUXFU_ASSIGN_EXTENSION[12]=2	
$MC_AUXFU_ASSIGN_VALUE[12]=3	
$MC_AUXFU_ASSIGN_GROUP[12]=10	
$MC_AUXFU_ASSIGN_TYPE[13]="M"	; 对辅助功能 14 的描述：M2=4
$MC_AUXFU_ASSIGN_EXTENSION[13]=2	
$MC_AUXFU_ASSIGN_VALUE[13]=4	
$MC_AUXFU_ASSIGN_GROUP[13]=10	
$MC_AUXFU_ASSIGN_TYPE[14]="M"	; 对辅助功能 15 的描述：M2=5
$MC_AUXFU_ASSIGN_EXTENSION[14]=2	
$MC_AUXFU_ASSIGN_VALUE[14]=5	
$MC_AUXFU_ASSIGN_GROUP[14]=10	
$MC_AUXFU_ASSIGN_TYPE[15]="M"	; 对辅助功能 16 的描述：M2=70
$MC_AUXFU_ASSIGN_EXTENSION[15]=2	
$MC_AUXFU_ASSIGN_VALUE[15]=70	
$MC_AUXFU_ASSIGN_GROUP[15]=10	
$MN_AUXFU_GROUP_SPEC[10]='H22'	; 第 11 辅助功能组的设定
$MC_AUXFU_ASSIGN_TYPE[16]="S"	; 对辅助功能 17 的描述：S2=<所有值>
$MC_AUXFU_ASSIGN_EXTENSION[16]=2	
$MC_AUXFU_ASSIGN_VALUE[16]=-1	
$MC_AUXFU_ASSIGN_GROUP[16]=11	
$MN_AUXFU_GROUP_SPEC[11]='H21'	; 第 12 辅助功能组的设定
$MC_AUXFU_ASSIGN_TYPE[17]="M"	; 对辅助功能 18 的描述：M50
$MC_AUXFU_ASSIGN_EXTENSION[17]=0	

程序代码	注释
$MC_AUXFU_ASSIGN_VALUE[17]=50	
$MC_AUXFU_ASSIGN_GROUP[17]=12	
$MC_AUXFU_ASSIGN_TYPE[18]="M"	; 对辅助功能 19 的描述：M51
$MC_AUXFU_ASSIGN_EXTENSION[18]=0	
$MC_AUXFU_ASSIGN_VALUE[18]=51	
$MC_AUXFU_ASSIGN_GROUP[18]=12	
$MN_AUXFU_GROUP_SPEC[12]='H21'	; 第 13 辅助功能组的设定
$MC_AUXFU_ASSIGN_TYPE[19]="M"	; 对辅助功能 20 的描述：M52
$MC_AUXFU_ASSIGN_EXTENSION[19]=0	
$MC_AUXFU_ASSIGN_VALUE[19]=52	
$MC_AUXFU_ASSIGN_GROUP[19]=13	
$MC_AUXFU_ASSIGN_TYPE[20]="M"	; 对辅助功能 21 的描述：M53
$MC_AUXFU_ASSIGN_EXTENSION[20]=0	
$MC_AUXFU_ASSIGN_VALUE[20]=53	
$MC_AUXFU_ASSIGN_GROUP[20]=13	

9.18 数据表

9.18.1 机床数据

9.18.1.1 NC 专用机床数据

编号	名称：$MN_	说明
10713	M_NO_FCT_STOPRE	带预处理停止的 M 功能
10714	M_NO_FCT_EOP	NC-RESET 后生效的主轴用 M 功能
10715	M_NO_FCT_CYCLE	须由子程序替换的 M 功能
11100	AUXFU_MAXNUM_GROUP_ASSIGN	每个通道用户自定义辅助功能的最大数量
11110	AUXFU_GROUP_SPEC	组专用输出特性
11450	SEARCH_RUN_MODE	程序段搜索后的特性

9.18.1.2 通道专用机床数据

编号	名称：$MC_	说明
20110	RESET_MODE_MASK	零件程序启动后控制系统初始设置的定义
20112	START_MODE_MASK	启动后以及复位或零件程序结束时控制系统初始设置的定义
20270	CUTTING_EDGE_DEFAULT	未作设定时刀沿的初始设置
20800	SPF_END_TO_VDI	子程序结束/停止发送至 PLC
22000	AUXFU_ASSIGN_GROUP	用户自定义辅助功能的分组指定
22010	AUXFU_ASSIGN_TYPE	用户自定义辅助功能的类型
22020	AUXFU_ASSIGN_EXTENSION	用户自定义辅助功能的地址扩展
22030	AUXFU_ASSIGN_VALUE	用户自定义辅助功能的数值
22035	AUXFU_ASSIGN_SPEC	用户自定义辅助功能的输出特性
22040	AUXFU_PREDEF_GROUP	预定义辅助功能的分组指定
22050	AUXFU_PREDEF_TYPE	预定义辅助功能的类型
22060	AUXFU_PREDEF_EXTENSION	预定义辅助功能的地址扩展
22070	AUXFU_PREDEF_VALUE	预定义辅助功能的数值
22080	AUXFU_PREDEF_SPEC	预定义辅助功能的输出特性
22100	AUXFU_QUICK_BLOCKCHANGE	使用快速辅助功能时的程序段切换延迟
22110	AUXFU_H_TYPE_INT	H 辅助功能的类型
22200	AUXFU_M_SYNC_TYPE	M 功能的输出时间点
22210	AUXFU_S_SYNC_TYPE	S 功能的输出时间点
22220	AUXFU_T_SYNC_TYPE	T 功能的输出时间点
22230	AUXFU_H_SYNC_TYPE	H 功能的输出时间点
22240	AUXFU_F_SYNC_TYPE	F 功能的输出时间点
22250	AUXFU_D_SYNC_TYPE	D 功能的输出时间点
22252	AUXFU_DL_SYNC_TYPE	DL 功能的输出时间点
22254	AUXFU_ASSOC_M0_VALUE	用于程序停止的附加 M 功能
22256	AUXFU_ASSOC_M1_VALUE	用于有条件停止的附加 M 功能
22530	TOCARR_CHANGE_M_CODE	刀架切换时的 M 代码
22532	GEOAX_CHANGE_M_CODE	几何轴切换时的 M 代码

编号	名称：$MC_	说明
22534	TRAFO_CHANGE_M_CODE	转换切换时的 M 代码
22560	TOOL_CHANGE_M_CODE	换刀辅助功能

9.18.2 信号

9.18.2.1 发送至通道的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
激活关联的 M01	DB21, ...DBX30.5	DB320x.DBX14.5

9.18.2.2 来自通道的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
M 功能 1 - 5 更改	DB21, ...DBX58.0-4	DB250x.DBX4.0-4
M 功能 1 - 5 未解译	DB21, ...DBX59.0-4	-
S 功能 1 - 3 更改	DB21, ...DBX60.0-2	DB250x.DBX6.0
S 功能 1 - 3 快速	DB21, ...DBX60.4-6	-
T 功能 1 - 3 更改	DB21, ...DBX61.0-2	-
T 功能 1 - 3 快速	DB21, ...DBX61.4-6	-
D 功能 1 - 3 更改	DB21, ...DBX62.0-2	DB250x.DBX10.0
D 功能 1 - 3 快速	DB21, ...DBX62.4-6	-
DL 功能更改	DB21, ...DBX63.0	-
DL 功能快速	DB21, ...DBX63.4	-
H 功能 1 - 3 更改	DB21, ...DBX64.0-2	DB250x.DBX12.0-2
H 功能 1 - 3 快速	DB21, ...DBX64.4-6	-
F 功能 1 - 6 更改	DB21, ...DBX65.0-5	-
M 功能 1 - 5 快速	DB21, ...DBX66.0-4	-
F 功能 1 - 6 快速	DB21, ...DBX67.0-5	-
M 功能 1 的扩展地址（16 位 Int）	DB21, ...DBB68-69	DB250x.DBB3004
M 功能 1（DInt）	DB21, ...DBB70-73	DB250x.DBD3000

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
M 功能 2 的扩展地址（16 位 Int）	DB21, ...DBB74-75	DB250x.DBB3012
M 功能 2（DInt）	DB21, ...DBB76-79	DB250x.DBD3008
M 功能 3 的扩展地址（16 位 Int）	DB21, ...DBB80-81	DB250x.DBB3020
M 功能 3（DInt）	DB21, ...DBB82-85	DB250x.DBD3016
M 功能 4 的扩展地址（16 位 Int）	DB21, ...DBB86-87	DB250x.DBB3028
M 功能 4（DInt）	DB21, ...DBB88-91	DB250x.DBD3024
M 功能 5 的扩展地址（16 位 Int）	DB21, ...DBB92-93	DB250x.DBB3036
M 功能 5（DInt）	DB21, ...DBB94-97	DB250x.DBD3032
S 功能 1 的扩展地址（16 位 Int）	DB21, ...DBB98-99	DB250x.DBB4004
S 功能 1（Real）	DB21, ...DBB100-103	DB250x.DBD4000
S 功能 2 的扩展地址（16 位 Int）	DB21, ...DBB104-105	DB250x.DBB4012
S 功能 2（Real）	DB21, ...DBB106-109	DB250x.DBD4008
S 功能 3 的扩展地址（16 位 Int）	DB21, ...DBB110-111	DB250x.DBB4020
S 功能 3（Real）	DB21, ...DBB112-115	DB250x.DBD4016
T 功能 1 的扩展地址（16 位 Int）	DB21, ...DBB116-117	-
T 功能 1（Int）	DB21, ...DBB118-119	DB250x.DBD2000
T 功能 2 的扩展地址（16 位 Int）	DB21, ...DBB120-121	-
T 功能 2（Int）	DB21, ...DBB122-123	-
T 功能 3 的扩展地址（16 位 Int）	DB21, ...DBB124-125	-
T 功能 3（Int）	DB21, ...DBB126-127	-
D 功能 1 的扩展地址（8 位 Int）	DB21, ...DBB128	-
D 功能 1（8 位 Int）	DB21, ...DBB129	DB250x.DBD5000
D 功能 2 的扩展地址（8 位 Int）	DB21, ...DBB130	-
D 功能 2（8 位 Int）	DB21, ...DBB131	-
D 功能 3 的扩展地址（8 位 Int）	DB21, ...DBB132	-
D 功能 3（8 位 Int）	DB21, ...DBB133	-
DL 功能的扩展地址（8 位 Int）	DB21, ...DBB134	-
DL 功能（Real）	DB21, ...DBB136	-
H 功能 1 的扩展地址（16 位 Int）	DB21, ...DBB140-141	DB250x.DBB6004
H 功能 1（Real 或 DInt）	DB21, ...DBB142-145	DB250x.DBD6000

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
H 功能 2 的扩展地址（16 位 Int）	DB21, ...DBB146-147	DB250x.DBB6012
H 功能 2（Real 或 DInt）	DB21, ...DBB148-151	DB250x.DBD6008
H 功能 3 的扩展地址（16 位 Int）	DB21, ...DBB152-153	DB250x.DBB6020
H 功能 3（Real 或 DInt）	DB21, ...DBB154-157	DB250x.DBD6016
F 功能 1 的扩展地址（16 位 Int）	DB21, ...DBB158-159	-
F 功能 1（Real）	DB21, ...DBB160-163	-
F 功能 2 的扩展地址（16 位 Int）	DB21, ...DBB164-165	-
F 功能 2（Real）	DB21, ...DBB166-169	-
F 功能 3 的扩展地址（16 位 Int）	DB21, ...DBB170-171	-
F 功能 3（Real）	DB21, ...DBB172-175	-
F 功能 4 的扩展地址（16 位 Int）	DB21, ...DBB176-177	-
F 功能 4（Real）	DB21, ...DBB178-181	-
F 功能 5 的扩展地址（16 位 Int）	DB21, ...DBB182-183	-
F 功能 5（Real）	DB21, ...DBB184-187	-
F 功能 6 的扩展地址（16 位 Int）	DB21, ...DBB188-189	-
F 功能 6（Real）	DB21, ...DBB190-193	-
动态 M 功能：M00-M07	DB21, ...DBB194	DB250x.DBB1000
动态 M 功能：M08-M15	DB21, ...DBB195	DB250x.DBB1001
动态 M 功能： M16 - M23	DB21, ...DBB196	DB250x.DBB1002
动态 M 功能： M24 - M31	DB21, ...DBB197	DB250x.DBB1003
动态 M 功能： M32 - M39	DB21, ...DBB198	DB250x.DBB1004
动态 M 功能： M40 - M47	DB21, ...DBB199	DB250x.DBB1005
动态 M 功能： M48 - M55	DB21, ...DBB200	DB250x.DBB1006
动态 M 功能： M56 - M63	DB21, ...DBB201	DB250x.DBB1007
动态 M 功能： M64 - M71	DB21, ...DBB202	DB250x.DBB1008
动态 M 功能： M72 - M79	DB21, ...DBB203	DB250x.DBB1009
动态 M 功能： M80 - M87	DB21, ...DBB204	DB250x.DBB1010
动态 M 功能： M88 - M95	DB21, ...DBB205	DB250x.DBB1011
动态 M 功能：M96-M99	DB21, ...DBX206.0-3	DB250x.DBB1012.0-3
关联 M00/M01 生效	DB21, ...DBX318.5	DB330x.DBX4002.5

9.18.2.3 发送至进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
用于定位轴的 F 功能（REAL）	DB31, ...DBB78-81	-

9.18.2.4 来自进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
用于主轴的 M 功能（Int）	DB21, ...DBB86-87	DB370x.DBD0000
用于主轴的 S 功能（Real）	DB21, ...DBB88-91	DB370x.DBD0004

K1：BAG、通道、程序运行、复位特性 10

10.1 简要说明

通道

手动运行轴和自动执行零件程序时，一个 NC 通道为最小单位。一个通道在某个时间点总是处于特定的运行方式下，例如 AUTOMATIC、MDI 或 JOG。可将一个通道视为一个独立的 NC。

运行方式组（BAG）

一个通道总是属于一个运行方式组。一个运行方式组也可能由多个通道组成。

运行方式组的标志是：同一运行方式组中的所有通道在一个时间点总是处于同一运行方式下，例如 AUTOMATIC、MDI 或 JOG。这通过 NC 内部运行方式逻辑确保。

可将一个运行方式组视为独立的多通道 NC。

通道空位

配置通道时可设置占位通道，从而使对同系列机床的配置尽可能一致，并只激活实际使用的通道。

程序测试

可通过以下方式对新的零件程序进行测试或试运行：

- 不输出设定值的程序执行
- 程序单段运行
- 采用空运行进给执行程序
- 跳过零件程序段
- 进行/不进行计算的程序段搜索

程序段搜索

通过程序段搜索可实现下列程序仿真，用于搜索特定的程序位置：

- 类型 1，不在轮廓处计算
- 类型 2，在轮廓处计算
- 类型 4，在程序段终点计算
- 类型 5，通过从以往的信息计算所需的所有数据来实现所选程序位置的自动启动
- 程序段搜索后自动启动 ASUB
- 级联程序段搜索
- 在“程序测试”模式下执行跨通道程序段搜索

程序运行

程序运行是指在 AUTOMATIC 或 MDI 运行方式下执行零件程序或零件程序段。执行期间可通过 PLC 接口信号和指令对程序运行进行调整。

通过通道专用机床数据可为每个通道设定初始设置。这些初始设置例如会影响 G 功能组合辅助功能输出。

只有在所涉及通道处于复位状态下时才能选择零件程序。

此外，所有其他程序运行都通过 PLC 接口信号和对应的指令处理。

- 启动零件程序或零件程序段
- 零件程序计算和程序控制
- RESET 指令，程序状态和通道状态
- 对操作和程序动作的响应
- 事件控制的程序调用

异步子程序（ASUB），中断程序

借助中断输入功能，NC 可中断当前的零件程序执行，并在中断程序/ ASUB 中对高优先级事件进行响应。

单程序段

单程序段功能可协助用户逐段执行零件程序。

单程序段功能有 3 种设置方式：

- SLB1: = 插补单程序段
- SLB2: = 解码单程序段
- SLB3: = 循环中停止

基本程序段显示

除现有的程序段显示外，还可通过另一个基本程序段显示功能对可引起机床上动作的所有程序段进行显示。

实际逼近的终点位置作为绝对位置显示。位置值可选择以工件坐标系（WCS）为基准，或以可设定零点坐标系（SZS）为基准。

外部执行程序

在加工复杂的工件时， NC 的存储空间可能不足以存放程序。使用“外部执行”或“EES（Execution from External Storage，从外部存储器执行）”功能可从一个外部存储器（例如硬盘）调用和执行零件程序。

上电/复位等情形后的特性

下列情形后的控制系统特性：

- 启动（上电）
- 复位/零件程序结束
- 零件程序开始

可通过机床数据修改以针对特定系统设置，例如用于 G 指令、刀具长度补偿、转换、耦合组、切向跟踪、可编程同步主轴等功能。

调用含 M 功能、T 功能或 D 功能的子程序

在某些应用中，通过子程序替代 M 功能、T 功能、D 功能以及个别 NC 语言指令（SPOS、SPOSA）更为适宜。这例如可用于调用换刀程序。

通过相应的机床数据可对含 M 功能、T 功能或 D 功能的子程序（例如用于齿轮档切换）进行定义和控制。

程序运行时间/工件计数器

控制系统可提供程序运行时间和工件计数的相关信息，以协助机床操作人员。

此处定义的功能**与刀具管理功能不同**，特别针对无刀具管理功能的 NC 系统。

10.2 运行方式组（BAG）

运行方式组

在运行方式组（BAG）中，NC 的多个通道组合成一个加工单元。原理上可将其视作 NC 内部的一个独立“NC”。

BAG 的主要特征：归属于同一 BAG 的所有通道在同一时间点始终处于相同的运行方式（AUTOMATIC、JOG、MDI）下。

说明

下面的功能说明以一个 BAG 和一条通道为例。

需要多条通道的功能（例如“跨通道取轴”功能）请见：

文档：

功能手册之扩展功能分册；“K5：跨通道程序协调”和“K10：跨通道取轴”

参数设置

通过以下数据将通道指定给 BAG：

MD10010 $MN_ASSIGN_CHAN_TO_MODE_GROUP[<通道下标>] = BAG 编号

所有具有相同 BAG 编号的通道共同构成一个 BAG。

说明

默认情形下，NC 始终包含一个有一条通道的 BAG。无法参数设置无通道的 NC。

BAG 专用 NC/PLC 接口信号

BAG 专用 NC/PLC 接口信号位于数据块 DB11 中。

BAG 专用 NC/PLC 接口主要涵盖以下接口信号：

- 请求信号 PLC → NC
  - BAG 复位
  - BAG 停止，进给轴和主轴
  - BAG 停止
  - 运行方式切换禁止
  - 运行方式：JOG、MDI、AUTO
  - 单程序段：类型 A、类型 B
  - 机床功能：REF、REPOS、TEACH IN、INC x
- 状态信号 NC → PLC
  - 运行方式选通脉冲：JOG、MDI、AUTOMATIC
  - 机床功能选通脉冲：REF、REPOS、TEACH IN
  - 所有通道处于复位状态
  - BAG 复位
  - BAG 就绪
  - 运行方式选通脉冲：JOG、MDI、AUTOMATIC
  - 机床功能生效：REF、REPOS、TEACH、INC x

激活的和未激活的通道，通道空位

激活的通道

BAG 编号 ≠ 0 的通道为激活的通道。

未激活的通道

BAG 编号 为 0 的通道是未激活的通道。其不会占用控制系统内部的存储空间。

通道空位

BAG 编号 为 0 的通道不仅仅是未激活的通道。在通道序列中其即所谓的通道空位。

通道空位的优势在于：能够在一系列结构相似的机床中尽可能保持配置数据相同。特定调试中只会激活当前机床所需的通道。未占用的存储空间则可作为附加的用户存储器自由使用。

表格 10-1　示例

机床数据	含义
MD10010 $MN_ASSIGN_CHAN_TO_MODE_GROUP[0] = 1	通道 1，BAG 1
MD10010 $MN_ASSIGN_CHAN_TO_MODE_GROUP[1] = 2	通道 2，BAG 2
MD10010 $MN_ASSIGN_CHAN_TO_MODE_GROUP[2] = 0	通道 3，不存在
MD10010 $MN_ASSIGN_CHAN_TO_MODE_GROUP[3] = 1	通道 4，BAG 1
MD10010 $MN_ASSIGN_CHAN_TO_MODE_GROUP[4] = 2	通道 5，BAG 2

参见

K2：轴、坐标系、框架 (页 765)

S1：主轴 (页 1401)

10.2.1 BAG 停止

功能

通过以下 NC/PLC 接口信号在 BAG 的所有通道中停止进给轴的运行（或进给轴和主轴的运行），以及中断零件程序执行：

DB11 DBX0.5（BAG 停止）

DB11 DBX0.6（BAG 停止，进给轴和主轴）

10.2.2 BAG 复位

功能

BAG 复位请求通过以下 BAG 专用 NC/PLC 接口信号发出：

DB11 DBX0.7 = 1（BAG 复位）

作用

对 BAG 中的通道生效：

- 零件程序准备（预处理）停止。
- 所有进给轴和主轴均根据加速度特性曲线，在不损坏轮廓的情况下制动至静止状态。
- 尚未输出至 PLC 的辅助功能将不再输出。
- 预处理指针会被设置至中断位置，程序段指针会被设置至各零件程序的起始处。
- 所有原始设置（例如 G 指令）均设置为参数设置的值。
- 所有删除标准为“通道复位”的报警均会被删除。

当 BAG 的所有通道都处于复位状态时：

- 所有删除标准为“BAG 复位”的报警均被删除。
- 会在 NC/PLC 接口上显示“BAG 复位完成”和“BAG 就绪”：
  DB11 DBX6.7 = 1（所有通道处于复位状态）
  DB11 DBX6.3 = 1（BAG 就绪）

10.3 运行方式和运行方式切换

运行方式的唯一性

一个运行方式组（BAG）的所有通道总是处于同一运行方式下：

- AUTOMATIC
- JOG
- MDI

若各通道被指定给不同的运行方式组，那么通道切换时其同样会切换至相应的 BAG。借此可通过通道切换实现运行方式切换。

运行方式

可供使用的运行方式如下：

- **AUTOMATIC**
  自动执行零件程序：
  - 零件程序测试
  - BAG 的所有通道可同时生效。
- **JOG in AUTOMATIC**
  JOG in AUTOMATIC 是对 AUTOMATIC 运行方式的扩展，目的在于简化操作。在前提条件允许的情形下，可不退出 AUTOMATIC 模式执行 JOG。
- **JOG**
  通过机床控制面板上的运行键或机床控制面板上连接的手轮手动运行轴：
  - 在通过 ASUB 或静态同步动作执行的运行中，须注意通道专用信号和闭锁。
  - 同时须留意耦合关系。
  - BAG 中每条通道均可生效。
- **MDI**
  Manual Data Automatic（通过操作界面输入程序段）：
  - 在限制下执行零件程序和部分零件程序。
  - 零件程序测试
  - 每个 BAG 最多 1 条通道生效（仅在 TEACH IN 中）。
  - 只能在 JOG、REPOS 或 TEACHIN 等下级机床功能中手动运行轴。

适用于所有运行方式

跨运行方式同步动作

对于所有运行方式，可通过 IDS 并行于通道为以下功能执行模态同步动作：

- 指令轴功能
- 主轴功能
- 工艺循环

选择

用户可通过操作界面上的软键选择所需的运行方式。

此选择（AUTOMATIC、MDI 或 JOG）会被传输至 PLC 的 NC/PLC 接口，但尚不会被激活：
DB11 DBX4.0, 0.1, 0.2（运行方式选通脉冲）

激活和优先级

BAG 的运行方式通过以下 NC/PLC 接口激活：

DB11 DBX0.0, 0.1, 0.2（运行方式）

若在同一时间选择了多个运行方式，那么其优先级如下：

优先级	运行方式	BAG 信号（NC → PLC）
第 1 优先级，高	JOG	DB11 DBX0.2
第 2 优先级，中	MDI	DB11 DBX0.1
第 3 优先级，低	AUTOMATIC	DB11 DBX0.0

显示

BAG 的当前运行方式通过以下 NC/PLC 接口显示：

DB11 DBX6.0, 0.1, 0.2（生效的运行方式）

BAG 信号（NC → PLC）	生效的运行方式
DB11 DBX6.2	JOG
DB11 DBX6.1	MDI
DB11 DBX6.0	AUTOMATIC

机床功能

可在运行方式内选择机床功能，其在 BAG 内同样适用：

- JOG 运行方式内的机床功能
  - REF（回参考点）
  - REPOS（重新定位）
  - JOG 回退（沿刀具方向回退）
- MDI 运行方式内的机床功能
  - REF（回参考点）
  - REPOS（重新定位）
  - TEACHIN（轴位置示教）

NC/PLC 接口信号

- DB11 DBX5.0, 0.1, 0.2（机床功能选通脉冲）：请求
- DB11 DBX1.0, 0.1, 0.2（机床功能）：激活
- DB11 DBX7.0-2（生效的机床功能）：反馈信息

通道状态

- **通道复位**
  机床处于初始状态。这由机床制造商通过 PLC 程序定义，例如在接通或程序结束后。
- **通道生效**
  一个程序已启动，正在执行程序或回参考点。
- **通道中断**
  运行的程序或回参考点运行已被中断。

运行方式内的功能

可为运行方式补充用户自定义功能。这些功能与工艺和机床无关，可从“通道复位”、“通道生效”或“通道中断”这三种状态启动和/或执行。

适用于机床功能 TEACH IN 的前提条件

TEACH IN 不可用于生效轴组中的引导轴，例如：

- 龙门轴组或龙门轴对
- 引导轴和跟随轴组成的耦合轴组

JOG in AUTOMATIC

当 BAG 处于“RESET”状态下且轴可执行 JOG 时，允许在 AUTOMATIC 运行方式下使用 JOG。BAG 的“RESET”状态表示：

- 所有通道处于“RESET”状态
- 所有程序已终止
- 各通道中均无 DRF 生效

当轴不处于以下状态时，则表示**可执行 JOG**：

- PLC 轴作为并行定位轴（由 PLC 请求轴）
- 指令轴（轴已通过同步动作编程，且运动尚未结束）

- 旋转主轴（RESET 后仍旋转的主轴）
- 异步往复轴

提示：“可执行 JOG”属性与“JOG in AUTOMATIC”功能无关。

激活

可通过以下机床数据激活“JOG in AUTOMATIC”功能：

MD10735 $MN_JOG_MODE_MASK

- 上电前必须设置以下机床数据：
  MD10735 $MN_JOG_MODE_MASK，位 0 = 1
- 用户切换至 AUTO（PLC 用户接口 DB11 DBX0.0 = 0→1 脉冲沿）。若此前 BAG 所有通道中 NC 的通道状态均为“RESET”且程序状态均为“终止”，那么“JOG in AUTOMATIC”生效。此外所涉及的轴必须“可执行 JOG”。DRF 必须取消。
- 在所有通道状态不为“RESET”和程序状态不为“终止”的 BAG 通道中，系统会触发 RESET，或者通过 M30/M2 使运行中的程序终止。
- 所涉及的轴会自动变为“可执行 JOG”（例如跨通道取轴：PLC → NC）。

提示：在多数应用场合下，待运行轴均“可执行 JOG”，因此切换至 AUTOMATIC 运行方式后“JOG in AUTOMATIC”便生效。

特性

- 通过 +/–键可产生 JOG 运动，BAG 将**内部**切换至 JOG（即“内部 JOG”）。
- DRF 未生效时，通过运行手轮可产生 JOG 运动，BAG 将**内部**切换至 JOG（即“内部 JOG”）。
- JOG 运动开始后，达到增量的终点位置（若事先进行了设置）或通过“删除剩余行程”终止运行时运动才告结束。
  这样一来，可使用停止键暂停增量运行，并通过开始键重新运行至终点。在此期间 NC 均处于“内部 JOG”模式下。可运行部分增量，但不可通过停止键中断。另有一种模式，在该模式下松开运行键会使增量运行中断。
- 不执行 JOG 运动时，“JOG in AUTOMATIC”的特性与“AUTOMATIC”相同，特别是通过开始键启动所选择的零件程序，以及通过相应的 HMI 软键触发程序段搜索这些方面。
- JOG 运动生效时，NC 进入内部 JOG 模式，此时程序段搜索请求会被拒绝，也无法通过开始键启动零件程序。此时开始键用于启动可能剩余的增量运行，或者无作用。
- 只要 BAG 中的一根轴在 JOG 下运行，该 BAG 便处于内部 JOG 模式下。
  注释：此阶段可能随一根轴的 JOG 运动开始，在*另一根*轴的 JOG 运动完成时才告结束。

- 不允许对 JOG 运动生效的轴执行跨通道取轴。（该轴可能会切换 BAG）。此时 NC 会禁止跨通道取轴。
- PLC 用户接口显示“Automatic”运行方式：
  - DB11 DBX6.0, 6.1, 6.2 = 1
  - DB11 DBX7.0, 7.1, 7.2 = 0
- 此 NC/PLC 接口显示，“JOG in AUTOMATIC”模式下 BAG 处于“BAG 复位”状态。
  - DB11 DBX6.4（BAG 已复位，BAG 1）
  - DB11 DBX26.4（BAG 已复位，BAG 2）
  - DB11 DBX46.4（BAG 已复位，BAG 3）
- 此 NC/PLC 接口显示，“JOG in AUTOMATIC”模式下 NC 已于内部切换至“内部 JOG”模式。
  - DB11 DBX6.5（NC 内部 JOG 生效，BAG 1）
  - DB11 DBX26.5（NC 内部 JOG 生效，BAG 2）
  - DB11 DBX46.5（NC 内部 JOG 生效，BAG 3）

前提条件

仅当 BAG 处于“BAG 复位”状态时，“JOG in AUTOMATIC”才能于内部切换至 JOG。也就是说在一个暂停的程序中无法直接执行点动。在这种情形下，用户可在 BAG 的*所有通道中*按下 JOG 键或复位键来执行点动。

选择 AUTOMATIC 运行方式时，INC 键会被取消激活，用户可以/必须重新按下 INC 键来选择所需的增量。NC 切换至“内部 JOG”时，所选择的增量将得以保留。

若用户尝试对几何轴或定向轴执行点动，NC 同样会切换至“内部 JOG”，并会执行运动。此时可物理运行多根轴，这些轴必须均为“可执行 JOG”。

JOG 运动后 NC 会取消“内部 JOG”并重新选择 AUTO。内部模式切换会被延迟至运动结束。这样一来，很多开关步骤（例如在使用手轮时就可能出现）会受到不必要的避免。此时 PLC 只允许以 PLC 信号“内部 JOG 生效”为准。

在轴无使能的情形下，NC 仍会切换至“内部 JOG”。

参见

R1：回参考点 (页 1349)

10.3.1 对各运行方式的监控和锁定

通道状态决定监控功能

运行方式下的监控

各种运行方式下生效的监控不尽相同。 这些监控与工艺和机床无关。

在每种运行方式下，根据运行状态会有部分监控功能生效。 哪些监控在哪些运行方式下和哪些运行状态中生效，这由通道状态决定。

运行方式下的锁定

各种运行方式下可能会有不同的锁定功能生效。 这些锁定与工艺和机床无关。

在每种运行方式下，根据运行状态可激活几近所有锁定功能。

10.3.2 运行方式切换

引言

运行方式切换通过 BAG 接口（DB11）请求和激活。一个运行方式组只能处于 AUTOMATIC、JOG 或 MDI 这其中一种运行方式下，即一个运行方式组中的多个通道不能同时处于不同的运行方式下。

是否能打到所请求的运行方式，以及如何执行该运行方式，这些均通过 PLC 程序针对机床配置。

说明

仅当不处于“通道状态生效”情形时，才能于控制系统内部实现运行方式切换。但是，为了实现无故障的运行方式切换，所有通道都须处于允许采用的运行状态下。

可进行的运行方式切换

下表显示了可对一个通道进行的运行方式切换。

	AUTOMATIC		JOG				MDI		
				AUTO	MDI		手动运行		AUTO
	复位	中断	复位	中断	中断	复位	中断	生效	中断
AUTOMATIC			X	X		X			
JOG	X	X				X	X		X
MDI	X	X	X		X				

“x”标记表示可进行的运行方式切换。

特殊情况

- **运行方式切换时报错**
  若运行方式切换请求被系统拒绝，则会发出故障信息“NC 停止后才可进行运行方式切换”。无需修改通道状态便可清除此故障信息。
- **运行方式切换禁止**
  通过以下接口信号：
  DB11 DBX0.4（运行方式切换禁止）
  可禁止运行方式切换。
  此时运行方式请求即已被抑制。
  用户须自行配置一条消息，以提示操作人员禁止生效。系统不会为此提供任何信号。
- **运行方式从 MDI 切换至 JOG**
  在运行方式从 MDI 切换至 JOG 后，若 BAG 的所有通道均处于复位状态，那么 NC 将从 JOG 切换至 AUTO。在此状态下可执行零件程序指令 `START` 或 `INIT`。
  若运行方式切换后 BAG 的一个通道不再处于复位状态，那么此时零件程序指令 `START` 会被系统拒绝，并触发报警 16952。

10.4 通道

NC 通道用于执行由用户设定的零件程序。

通道具有以下属性：

- 一条通道始终被指定给一个运行方式组 (页 522)（BAG）。
- 一条通道在一个时间点始终只能执行一个零件程序。

- 可将通道指定给机床轴、几何轴、副主轴以及主轴。只有这些轴可通过通道中执行的零件程序来运行。
- 一条通道包含以下内部单元
  - 预处理（程序解码和程序段预处理）
  - 主处理（轨迹插补和轴插补）
- 通道有一个 PLC 接口。借助此 NC/PLC 接口可通过 PLC 用户程序读取各种通道专用状态数据，以及写入对通道的请求。
- 通道中会启用通道专用的刀具补偿 (页 1593)。
- 通道中会启用通道专用的坐标系 (页 786)。
- 默认情形下，每条通道均具有一个独有的通道名称。设定的通道名称可通过以下机床数据修改：
  MD20000 $MC_CHAN_NAME = "<通道名称>"

多条通道可划归一个运行方式组（BAG）。一个 BAG 中的通道始终处于相同的运行方式（AUTOMATIC、JOG、MDI）下。

通道配置

通过以下机床数据可为通道设定名称：

MD20000 $MC_CHAN_NAME（通道名称）

各轴通过机床数据指定给现有通道。对于进给轴/主轴，同时只能有一个输出设定值的通道。进给轴/主轴的实际值可由多个通道同时读取。进给轴/主轴的信息必须已提供给相应通道。

此外还可通过机床数据定义以下通道专用设置：

- G 功能组的缺省设置或编程初始设置，通过以下机床数据设定：
  MD20150 $MC_GCODE_RESET_VALUES（G 功能组的初始设置）
- 辅助功能组，构成方式和输出时间点
- 机床轴和几何轴之间的转换条件
- 其他用于零件程序执行的设置

修改通道配置

通道配置无法以编程的方式在零件程序中或通过 PLC 用户程序在线修改。配置修改必须通过机床数据进行。重新上电后修改才生效。

容器轴和链接轴

轴容器将一组轴组合在一个容器中。这些轴称为容器轴。此时系统会将一个容器槽（相应容器内的环形缓存位）上的指针指定给一个通道轴。该容器中的轴会交替位于该槽。

轴容器中的机床轴在一个时间点只能归属于**一条**通道轴。

链接轴可固定指定给一个通道，或动态（轴容器切换）指定给本地乃至其他 NCU 的多个通道。链接轴未实际连接至通道的 NCU 的情形下，对该通道而言此轴非本地轴。

链接轴如下指定给一条通道：

- 采用固定指定时通过机床数据实现：
  显示对链接轴的直接逻辑机床轴映射。
- 采用动态指定时：
  显示链接轴的轴容器槽机床数据。

链接轴和容器轴的更多信息请见：

文档：

功能手册之扩展功能分册；多操作面板配备多 NCU，分布式系统（B3）

接口信号

通道 1 的信号位于 DB21 的 NC/PLC 接口中，通道 2 的信号位于 DB22 中，可通过 PLC 或 NC 监控或控制通道。

通道专用工艺设定

可为每个通道设定使用的工艺：

MD27800 $MC_TECHNOLOGY_MODE

供货状态下，机床数据默认用于铣削工艺。

通过 PLC 使用主轴功能

除功能块 FC18 外，也可通过轴专用 NC/PLC 接口信号并行于零件程序启动和停止主轴功能。

前提条件：

- 通道状态：“中断”或“RESET”
- 程序状态：“中断”或“终止”

以下功能可由 PLC 通过接口信号控制：

- 停止（相当于 M5）
- 开始顺时针旋转（相当于 M3）
- 开始逆时针旋转（相当于 M4）
- 选择齿轮档
- 定位（相当于 M19）

存在多个通道时，通过 PLC 启动的主轴在启动时所归属于的通道中生效。

特殊主轴接口的更多信息请见“S1：主轴 (页 1401)”章节。

PLC 控制的单轴进程

通过 PLC 不仅可以控制一个通道，也可实现对单轴的控制。为此，PLC 会通过 NC/PLC 接口从 NC 请求轴：

DB31, ... DBX28.7 = 1（PLC 控制轴）

以下功能可由 PLC 控制和调整：

- 终止进给轴/主轴的运行（相当于删除剩余行程）
- 停止或中断进给轴/主轴
- 继续运行进给轴/主轴（恢复运行）
- 将进给轴/主轴恢复至初始状态

通道专用信号交换（PLC → NC）的更多信息请见“P3：SINUMERIK 840D sl 的 PLC 基本程序 (页 945)”章节。

对自控单轴进程的详细说明请参见：

文档：
功能手册之扩展功能分册；定位轴（P2）

10.4.1 启动禁止，全局和通道专用

功能

在 ShopMill / ShopTurn 中，缺省情况下只能在机床区域中执行程序启动。其他区域（例如刀具）中的启动被启动禁止（PI 服务）锁定。

可通过 NC/PLC 接口信号取消自动化顺序运行的启动禁止。

NC/PLC 接口信号

取消启动禁止

借助此接口信号取消现有的启动禁止：

DB21, ... DBX7.5（取消启动禁止）

机床数据

通过此机床数据进行以下设置：若在启动禁止置位的情况下请求启动，则显示报警。

MD11411 $MN_ENABLE_ALARM_MASK，位 6

BTSS 变量

变量	说明
startLockState	全局启动禁止的状态
chanStartLockState	通道专用启动禁止的状态
startRejectCounter	采用全局启动禁止时的启动计数器
startLockCounter	采用通道专用启动禁止时的启动计数器

PI 服务

启动禁止的置位（_N_STRTLK）

可通过 PI 服务 _N_STRTLK 以全局或针对特定通道的方式将启动禁止置位。

- 全局启动禁止
  若全局启动禁止置位，则所有通道中的 NC 程序启动均被禁止。已运行的 NC 程序不受影响。下一次 NC 程序启动才会被禁止。
- 通道专用启动禁止
  通道专用启动禁止仅在 AUTO 运行模式以及通道状态“复位”下禁止 NC 程序的启动。

全局或通道专用启动禁止的复位（_N_STRTUL）

可通过 PI 服务 _N_STRTUL 将全局或通道专用启动禁止复位。

10.5 程序测试

控制系统具有多种用于测试或试运行新零件程序的功能。这些功能可显著降低测试阶段机床受损的风险及时间消耗。 可同时激活多个程序测试功能，以实现更佳的测试效果。

测试方法

本手册将对下列测试方法进行说明：

- 不输出设定值的程序执行
- 程序单段运行
- 采用空运行进给执行程序
- 跳过零件程序段
- 进行/不进行计算的程序段搜索

10.5.1 不输出设定值的程序执行

功能

在“程序测试”状态下系统会执行零件程序，但不会输出进给轴或主轴设定值。

借此用户可对编写的轴位置以及零件程序的辅助功能输出进行检查。此外还可将此程序仿真作为扩展的句法校验。

选择

通过操作界面在“程序控制”菜单中选择此功能。

选择时下列接口信号将置位：

DB21, ... DBX25.7（程序测试已选择）

但此时该功能还没有激活。

激活

通过以下接口信号激活功能：

DB21, ... DBX1.7（激活程序测试）

显示

程序测试生效后，操作界面上的对应区域会切换至反色，此外 PLC 中会置位以下接口信号：

DB21, ... DBX33.7（程序测试生效）

程序测试和程序运行

在程序测试功能生效的情况下，可通过以下接口信号启动和执行零件程序（包含辅助功能输出、等待时间、G 指令输出等）：

DB21, ... DBX7.1（NC 启动）

软件限位开关、工作区域限制等安全功能继续生效。

程序测试与一般程序运行的唯一区别在于：程序测试中会对所有轴（含主轴）启用内部**轴禁用**。因此机床轴不会发生运动，实际值于系统内部通过未输出的设定值生成。编写的速度保持不变。这表示，操作界面上的位置和速度设定与通常执行零件程序时完全相同。此时位置控制不会中断，因此在测试功能关闭后无需使进给轴回参考点。

说明

准停信号 DB31, ... DBX60.6/60.7（粗/精准停）能够反映机床上的实际状态。

在程序测试中，仅当轴被推离设定位置时才需取消这些信号（程序测试期间设定位置保持恒定）。

对于信号 DB21, ... DBX33.7（程序测试生效），PLC 程序和零件程序均可通过变量 $P_ISTEST 确定如何在测试期间响应或跳转至此信号。

说明

空运行进给率

“不触发轴运行的程序执行”也可与“空运行进给率”功能一同激活。这样一来能够缩短编写了较小进给值的零件程序的执行时间。

说明

刀具管理

程序测试时轴被禁用，因此刀库配置保持不变。必须通过一个 PLC 应用程序确保刀具管理数据和刀库之间的一致性。为此您可从刀盘获取 PLC 基本程序的示例。

10.5.2 程序单段运行

功能

采用“程序单段运行”时，零件程序执行会在每个程序段后暂停。 若选择了铣刀或刀沿半径补偿，那么会在每个由控制系统插入的中间程序段前暂停程序执行。

程序状态切换至“程序状态：停止”。

通道状态保持为激活。

通过 NC 启动来执行下一个零件程序段。

应用

用户可以使用此功能逐段执行零件程序并检查各个加工步骤。 如果已执行的程序段正确，可请求执行下一程序段。

单段方式

单段方式分为以下两种类型：

- 译码单程序段

  采用这种单段方式时，零件程序的所有程序段（包括没有运行功能的纯计算程序段）依次通过“NC 启动”执行。

- 动作单程序段（缺省设置）

  采用这种单段方式时，只单独执行触发动作（运行、辅助功能输出等）的程序段。
  译码时额外生成的程序段（例如尖角处的铣刀半径补偿）也会被单独运行。
  在计算程序段处则不会暂停执行，因为该程序段不触发动作。

单段方式通过操作界面在“程序控制”菜单中定义。

⚠ 小心
单程序段模式下的功能特点 对于 G33/G34/G35 程序段，仅当选择了“空运行进给率”时单程序段模式才生效。 计算程序段不会被单段执行（仅针对译码单程序段）。 SBL2 在 G33/G34/G35 中同样无效。

选择

单程序段模式可通过以下方式选择：

- 通过机床控制面板（“Single Block”键）
- 通过操作界面
  详细步骤请见：
  文档：
  安装的 HMI 应用的操作手册

激活

此功能由 PLC 基本程序通过以下接口信号激活：

DB21, ... DBX0.4（激活单程序段）

显示

单程序段模式生效时，操作界面上对应区域的状态栏会切换至反色。

启用了单程序段模式时，一个零件程序段执行完毕时会立即置位以下信号：

DB21, ... DBX35.3（程序状态：中断）

无单段停止的程序执行

在特定的程序运行中，选择了单程序段模式的情况下仍可设置无单段停止的程序执行，例如用于：

- 内部 ASUB
- 用户 ASUB
- 中间程序段
- 程序段搜索汇总程序段（动作程序段）
- 初始化程序段
- 含 DISPLOF 的子程序
- 无法重新组织的程序段
- 无法重新定位的程序段
- 无运行信息的重定位程序段
- 刀具定位程序段

此时通过以下机床数据设置：

MD10702 $MN_IGNORE_SINGLEBLOCK_MASK（忽略单程序段停止）

文档：
参数手册，机床数据的详细说明

10.5.3 采用空运行进给执行程序

功能

“以空运行进给执行程序”功能用于对通道中选择的 NC 程序的协作进行测试，其中例如包含 PLC 用户程序、外部信号和/或 NC 的其他通道。其中系统按照编写的指令、功能和运行速度精确地执行 NC 程序。为在测试期间缩短程序执行时间，可通过激活空运行进给更快地执行运动。例如在采用 G01、G02、G03、G33、G34、G35、G95 的情况下，空运行进给会取代编写的进给率生效。

在 AUTO 或 MDA 运行模式中在操作界面上选择空运行进给，随后可在自动中断时或在程序段末尾将其激活。可借助 NC/PLC 接口通过 PLC 用户程序进行激活。

有关进给率控制的更多信息请见 “V1：进给率 (页 1525)” 一章。

注意
过高的切削速度可致刀具或工件损坏 若在“以空运行进给执行程序”功能生效的情况下执行工件加工，产生的切削速度可能会超出允许的范围并导致刀具和/或工件损坏。

参数设置

空运行进给

通道中生效的空运行进给通过以下方式设置：

SD42100 $SC_DRY_RUN_FEED = <空运行进给>

说明

旋转进给率

空运行进给值也可替代通过 G95 编写的旋转进给率。

空运行进给设置

在“以空运行进给执行程序”时，通道中生效的进给率的选择标准通过以下方式设置：

SD42101 $SC_DRY_RUN_FEED_MODE = <值>

<值>	含义
0	SD42100 和编写的进给率中的最大值作为空运行进给率生效。
1	SD42100 和编写的进给率中的最小值作为空运行进给率生效。
2	SD42100 作为空运行进给率生效。
10	除螺纹切削（G33、G34、G35）和攻丝（G331、G332、G63）外，与设置为“0”时相同。在这些功能中，编写的进给率生效。
11	除螺纹切削（G33、G34、G35）和攻丝（G331、G332、G63）外，与设置为“1”时相同。在这些功能中，编写的进给率生效。
12	除螺纹切削（G33、G34、G35）和攻丝（G331、G332、G63）外，与设置为“2”时相同。在这些功能中，编写的进给率生效。

激活

选择

借助操作界面通过操作区“加工”选择此功能> “AUTO”或“MDA”运行模式 > “程序控制” > “空运行进给”。

作为发送至 PLC 用户程序的激活此功能的请求，将以下 NC/PLC 接口信号置位：

DB21, ... DBX24.6 = 1（空运行进给已选择）

激活

作为发送至 NC 的激活此功能的请求，PLC 用户程序须将以下 NC/PLC 接口信号置位：

DB21, ... DBX0.6 = 1（激活空运行进给）

反馈信息

此功能在 NC 中生效后，作为对机床操作者的反馈信息，操作界面上“加工”操作区的状态显示中会显示 **DRY**。

文档

有关“以空运行进给执行程序”功能的操作的详细信息参见：

- 操作手册，车削版
- 操作手册，铣削版

关键词：程序控制，空运行进给，DRY

10.5.4 跳过零件程序段

功能

在测试或试运行新程序时，可针对程序执行禁用或跳过特定程序段，以提升测试效率。 为此须使用斜线标记相关程序段。

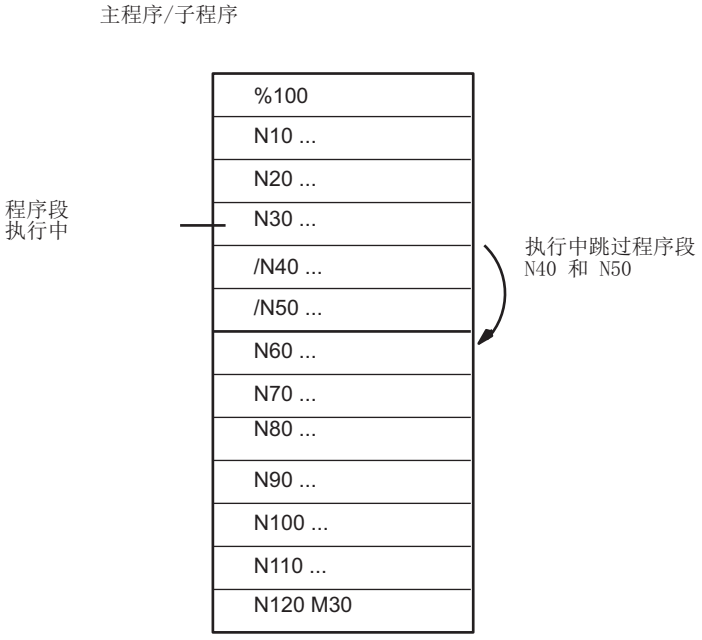


图 10-1 跳过零件程序段

选择

通过操作界面在“程序控制”菜单中选择此功能。

选择时下列接口信号将置位：

DB21, ... DBX26.0（程序段跳跃已选择）

但此时该功能还没有激活。

激活

通过以下接口信号激活功能：

DB21, ... DBX2.0（激活程序段跳跃）

说明

“跳过零件程序段”功能在程序段搜索中同样生效。

显示

“跳过零件程序段”功能激活时，操作界面上对应区域的状态栏会反显。

10.6 工件仿真

功能

在工件仿真中，系统会对当前零件程序进行完整的计算，并在操作界面中以图形显示结果。这样一来无需运行机床轴便可检查程序的加工结果，并能提前发现编写错误的加工步骤，从而避免工件上的错误加工。

仿真 NC

仿真功能使用一个独立的 NC 实例（仿真 NC）。因此开始仿真前必须将真实 NC 调整为仿真 NC。调整时会从 NC 读取所有生效的机床数据，并导入仿真 NC。NC 机床数据和循环机床数据均包含在生效的机床数据中。

仿真中的编译循环（仅适用于 840D sl）

4.4 之前的软件版本不支持工件仿真中的编译循环（CC），4.4 及更高的版本也仅支持个别编译循环。控制系统启动后会对支持的编译循环的机床数据进行一次性的调整。“仿真启动”时则不会进行调整！

说明

在零件程序中**不能**使用系统不支持的 CC 的专用语言指令及机床数据（参见“零件程序中的 CC 指令”部分）。

某些情形下，系统支持的 CC 的特殊运动（OEM 转换）会被错误显示。

零件程序中的 CC 指令

将系统不支持的编译循环（OMA1 ... OMA5，OEMIPO1/2，G810 ... G829，独立程序和功能）的语言指令写入零件程序时，若不特别处理，则会触发报警并使仿真终止。

解决方案：在零件程序中单独处理不支持的 CC 专用语言指令（$P_SIM 查询）。示例：

程序代码	注释
N1 G01 X200 F500	
IF (1==$P_SIM)	
N5 X300	; 仿真中 CC 不生效。
ELSE	
N5 X300 OMA1=10	
ENDIF	

10.7 程序段搜索类型 1、2 和 4

功能

通过程序段搜索功能可从近乎任意零件程序段开始执行零件程序。

使用该功能时，系统会在不触发运行的情况下快速运行零件程序，直至到达所选择的目标程序段。在此过程中会尽可能使目标程序段处的控制系统状态与正常执行零件程序时相同（例如轴位置、主轴转速、换入的刀具、NC/PLC 接口信号、变量值等方面），从而在尽可能减少手动调整的情况下从目标程序段起继续自动执行零件程序。

程序段搜索类型

- **类型 1：不带计算的程序段搜索**
  不带计算的程序段搜索能够最快地搜索到零件程序段。搜索中不执行任何计算。目标程序段处的控制系统状态与启动搜索前相同。
- **类型 2：在轮廓处计算的程序段搜索**
  通过使用在轮廓处计算的程序段搜索，可在任意情形下定位至编写的轮廓。使用 NC 启动移动到目标程序段的起始位置或者目标程序段之前的程序段的终点位置，再从该位置运行至终点位置。使用此搜索时的程序执行遵循轮廓。

- **类型 4：在程序段终点计算的程序段搜索**
  通过使用在程序段终点计算的程序段搜索，可在任意情形下定位至一个目标位置（例如换刀位置）。此时会使用目标程序段中生效的插补类型定位至目标程序段的终点或编写的下一个位置。此过程将不遵循轮廓。
  系统只会运行目标程序段中编写的轴。在启动其他零件程序的自动执行前，必要时须在机床上通过“JOG-REPOS”运行方式手动确保无碰撞的初始状态。
- **类型 5：在“程序测试”模式计算的程序段搜索（SERUPRO）**
  SERUPRO（**se**arch **ru**n by **pro**gramtest，程序测试时搜索）是一个带计算的跨通道程序段搜索功能。NC 在“程序测试”模式下启动所选择的零件程序。到达目标程序段后再取消程序测试。采用此程序段搜索类型时，还可实现执行程序段搜索的通道、同步动作以及 NC 的其它通道之间的交互。
  参见“程序段搜索类型 5（SERUPRO） (页 559)”章节。

说明

对程序段搜索的更多说明请见“程序段搜索时的特性 (页 478)”章节。

跟随动作

在程序段搜索完成后，可进行以下跟随动作：

- 类型 1 - 类型 5：自动启动一个 ASUB
  切换至最后一个动作程序段后，可将一个用户程序作为 ASUB 启动。
- 类型 1 - 类型 4：级联程序段搜索
  可从“找到搜索目标”状态启动另一个程序段搜索来查找另一个目标。

10.7.1 功能说明

类型 2 或类型 4 的基本流程

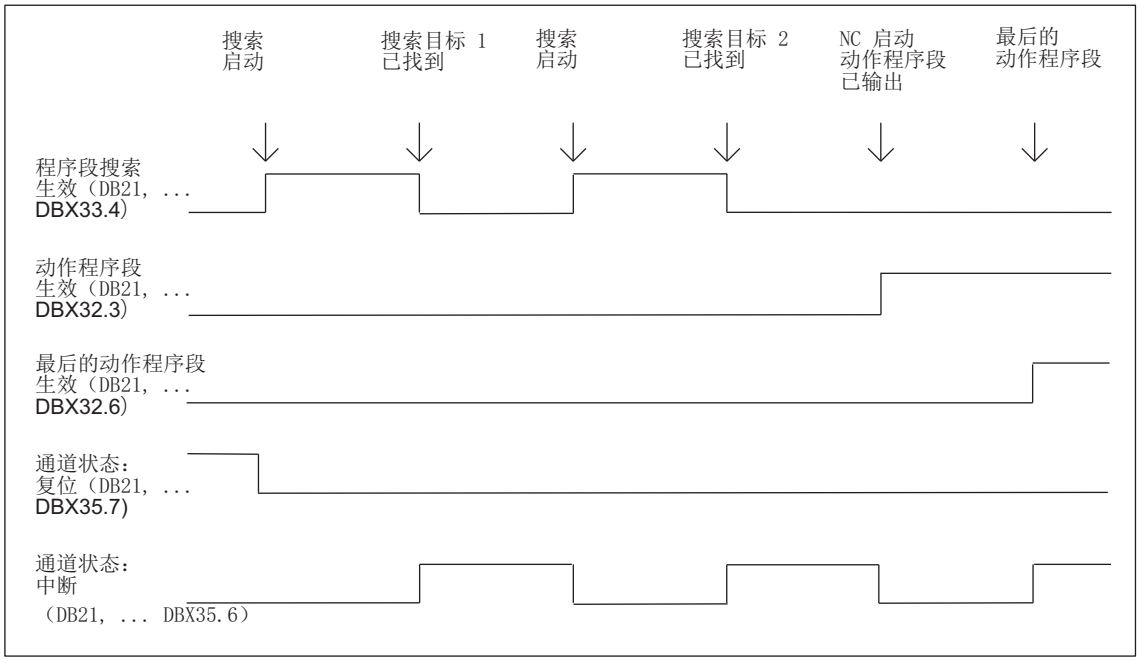
1. 用户：通过操作界面激活程序段搜索**类型 2** 或 **类型 4**（在 ... 处**计算**的程序段搜索）
2. 通过收集辅助功能搜索目标程序段
3. 在找到搜索目标 ⇒ 显示报警 10208”通过 NC 启动继续执行程序“后停止
4. 用户：通过 NC 启动执行动作程序段 ⇒ DB21, ... DBX7.1 = 1（NC 启动）
5. 执行动作程序段
6. 切换至最后一个动作程序段 ⇒ 自动启动 /_N_CMA_DIR/_N_PROG_EVENT_SPF（缺省设置）作为 ASUB

7. 切换至最后一个 ASUB 程序段（REPOSA）⇒ DB21, ... DBX32.6 = 1（最后的动作程序段生效）
8. **可选**：通过 PLC 用户程序执行用户专用请求
9. 显示报警 10208“通过 NC 启动继续程序程序”
10. 用户：通过 NC 启动继续执行程序 ⇒ DB21, ... DBX7.1 = 1（NC 启动）

未找到搜索目标

若未找到搜索目标，则会显示报警 15370“程序段搜索中未找到搜索目标”，并终止程序段搜索。

时序



动作程序段

在**类型 2** 或**类型 4**（在 ... 处**计算**的程序段搜索）程序段搜索期间，系统会收集动作，例如刀具编程（T、D）、主轴编程（S）、进给率编程或 M 功能输出。 通过 NC 启动执行动作程序段时，收集的动作会输出至 PLC。

说明

为动作程序段进行 NC 启动时，在**类型 2** 或**类型 4**（在 ... 处**计算**的程序段搜索）程序段搜索期间收集的主轴编程（S 值、M3 / M4 / M5 / M19、SPOS）将生效。
用户须通过 PLC 用户程序确保刀具可运行，或确保主轴编程复位及不输出：
DB31, ... DBX2.2 = 1（主轴复位）

前提条件

程序段搜索类型 4 后的累加模式

程序段搜索类型 4（在程序段终点计算的程序段搜索）后，若首个轴编程采用增量方式，可将编写的增量值叠加至到搜索目标为止所收集的位置值，或叠加至轴的当前实际值。这通过以下机床数据设置：

SD42444 $SC_TARGET_BLOCK_INCR_PROG

NC 启动后系统会分析此设定数值，以输出动作程序段。

单程序段

在类型 2 或类型 4（在 ... 处**计算**的程序段搜索）程序段搜索中找到搜索目标，且“单程序段”功能激活（DB21, ... DBX0.4 == 1（激活单程序段））后，若不希望在每个动作程序段后停止，可通过以下机床数据取消此特性：

MD10702 $MN_IGNORE_SINGLEBLOCK_MASK，位 3 = 1（在动作程序段中抑制单段模式）

接口信号“定位程序段生效”

此接口信号**仅**在程序段搜索**类型 2**（**在轮廓处计算**的程序段搜索）时置位：

- DB21, ... DBX32.4 = 1（定位程序段生效）

在程序段搜索**类型 4**（**在程序段终点计算**的程序段搜索）中，此接口信号不会置位，因为此时不会生成定位程序段（定位程序段等同于目标程序段）。

目标程序段的插补方式

在程序段搜索**类型 4**（**在程序段终点计算**的程序段搜索）中，定位运动以目标程序段中生效的插补方式执行。 若插补方式不是线性插补（G0 或 G1），则可能引起定位终止并触发报警（例如圆弧插补 G2 或 G3 时出现圆弧终点错误）。

10.7.2 程序段搜索与其它 NCK 功能的组合使用

10.7.2.1 程序段搜索后/中的 ASUB

程序段搜索类型 2 和类型 4：通道轴的同步

若在类型 2 或类型 4（在 ... 处计算的程序段搜索）的程序段搜索后启动 ASUB，系统会将所有通道轴的实际位置同步。

影响

在 ASUB 中读取下列系统变量时，得到的值如下：

- $P_EP：通道轴的当前实际位置（WCS）
- $AC_RETPOINT：收集的通道轴的程序段搜索位置（WCS）

程序段搜索类型 2：结束 ASUB

在程序段搜索类型 2（在轮廓处计算的程序段搜索）中，必须编写后续指令 REPOSA（重新定位至轮廓；线性；所有通道轴）来结束 ASUB。

影响

- 所有通道轴均被运行至程序段搜索中收集的搜索位置。
- $P_EP ==“收集的通道轴的程序段搜索位置（WCS）”

程序段搜索类型 4：REPOS 特性

在程序段搜索类型 4（在程序段终点计算的程序段搜索）后，在通过**起点**和**终点**描述的时间范围内，REPOS 指令不会触发自动重定位：

- **起点**：NC/PLC 接口信号 DB21,...DBX32.6 == 1（最后的动作程序段生效）
- **终点**：通过 NC 启动继续执行程序。

定位运行的起点是发出 NC 启动指令时通道轴的当前位置。终点通过零件程序中编写的其他运行得出。

在程序段搜索类型 4 中，NC 不会生成定位运行。

影响：

- 在退出 ASUB 后，系统变量 $P_EP（编写的终点位置）提供通道轴通过 ASUB 或手动（JOG 运行方式）运行定位至的实际位置。
  $P_EP == 通道轴的当前实际位置（WCS）

10.7.2.2 程序段搜索后的 PLC 动作

若所有动作程序段均通过 NC 执行，且动作可通过 PLC 实现，例如启动 ASUB 来执行换刀（或通过操作人员，例如溢出转存），那么以下通道专用 NC/PLC 接口信号会置位：

- DB21, ... DB32.6 = 1（最后的动作程序段生效）

参数设置输出报警的时间点

为了向操作人员提示“需要在通道中进行 NC 启动方能继续执行程序”，系统会显示报警 10208“通过 NC 启动继续执行程序”。

该报警的显示时间点可通过以下机床数据设置：
MD11450 $MN_SEARCH_RUN_MODE，位 0 = <值>

<值>	含义
0	程序段搜索后切换至最后一个动作程序段时： • 零件程序执行停止 • DB21, ... DBB32.6 = 1（最后的动作程序段生效） • 显示报警 10208
1	程序段搜索后切换至最后一个动作程序段时： • 零件程序执行停止 • DB21, ... DBB32.6 = 1（最后的动作程序段生效） • DB21, ... DBX1.6 == 1（PLC 动作结束）时，才显示报警 10208

伴随报警，系统还会置位以下接口信号：

- DB21, ... DBX36.7 = 1（导致程序处理停止的 NCK 报警，尚未应答）
- DB21, ... DBX36.6 = 1（NCK 报警，尚未针对通道应答）

10.7.2.3 程序段搜索后的主轴功能

控制系统特性和输出

程序段搜索中收集的主轴专用辅助功能是在动作程序段中自动输出至 PLC，还是依据用户定义于随后输出，可通过以下机床数据定义：

MD11450 $MN_SEARCH_RUN_MODE，位 2 = <值>

<值>	含义
0	程序段搜索中收集的主轴专用辅助功能（M3、M4、M5、M19、M70）在动作程序段中输出。
1	程序段搜索中收集的主轴专用辅助功能在动作程序段中不输出。 可依据用户定义于随后输出，例如在一个 ASUB 中输出。 为此，系统会将收集的主轴专用辅助功能保存在系统变量中：

系统变量

系统会将程序段搜索中收集的主轴专用辅助功能保存在以下系统变量中：

系统变量	说明
$P_SEARCH_S [<n>]	最近一次编写的主轴转速或切削速度
$P_SEARCH_SDIR [<n>]	最近一次编写的主轴旋转方向
$P_SEARCH_SGEAR[<n>]	最近一次编写的齿轮档 M 功能
$P_SEARCH_SMODE [<n>]	最近一次编写的主轴运行方式
$P_SEARCH_SPOS [<n>]	最近一次通过 M19、SPOS 或 SPOSA 编写的主轴位置或运行行程
$P_SEARCH_SPOSMODE [<n>]	最近一次通过 M19、SPOS 或 SPOSA 编写的位置定位模式
<n>：主轴号	

随后需要输出主轴专用辅助功能时，例如可在一个 ASUB 中读取系统变量，并于动作程序段输出后进行输出：

DB21, ... DBX32.6 == 1（最后的动作程序段生效）

说明

系统变量 $P_S、$P_DIR 和 $P_SGEAR 的内容可能会在程序段搜索后由于同步而丢失。

ASUB、程序段搜索以及动作程序段的更多详细信息请见“抑制主轴专用辅助功能的输出 (页 484)”和“程序测试 (页 537)”章节。

10.7.2.4 程序段搜索时读取系统变量

在零件程序中，可通过系统变量从预处理、主处理或伺服/驱动区域读取值：

$P_...	预处理相关系统变量，提供编写的值
$A_...	主处理相关系统变量，提供当前值
$V_...	伺服/驱动相关系统变量，提供当前值

在类型 2 和类型 4 的程序段搜索（在 ... 处**计算**的程序段搜索）中，程序段不会进入主处理，因此程序段搜索期间主处理相关和伺服/驱动相关系统变量不会改变。 针对这些变量，必要时须在 NC 程序中通过 $P_SEARCH（程序段搜索生效）查询程序段搜索是否生效，从而实现对程序段搜索的特殊处理。

预处理相关系统变量在所有搜索类型中均能提供正确的值。

10.7.3 程序段搜索后自动启动 ASUB

参数设置

激活功能

程序段搜索后的自动 ASUB 启动通过以下机床数据设置激活：

MD11450 $MN_SEARCH_RUN_MODE，位 1 = 1

待激活的程序

采用缺省设置时，程序段搜索后切换至最后一个动作程序段时，系统会从 _N_CMA_DIR 目录激活 **_N_PROG_EVENT_SPF** 程序作为 ASUB。 若需激活另一个程序，则须在以下机床数据中输入该用户程序的名称：

MD11620 $MN_PROG_EVENT_NAME

启用单程序段执行时的特性

启用了单程序段模式时，可通过以下通道专用机床数据设置是无中断地执行激活的 ASUB，还是使单程序段执行生效：

MD20106 $MC_PROG_EVENT_IGN_SINGLEBLOCK，位 4 = <值>

<值>	含义
0	单程序段执行生效。
1	抑制单程序段执行。

启用读取禁止时的特性

设置了读取禁止时（DB21, ... DBX6.1 = 1），可通过以下通道专用机床数据设置是完整执行 ASUB，还是使读取禁止生效：

MD20107 $MC_PROG_EVENT_IGN_INHIBIT，位 4 = <值>

<值>	含义
0	读取禁止生效。
1	抑制读取禁止。

说明

MD11620、MD20108 和 MD20107 的更多参数设置信息请见“参数设置 (页 621)”章节。

编程

启动 ASUB 的事件保存在系统变量 $P_PROG_EVENT 中。 在程序段搜索后自动激活的情形下，$P_PROG_EVENT 输出值“5”。

过程

程序段搜索后自动启动 ASUB 的过程

1. 用户：通过操作界面激活程序段搜索**类型 2** 或 **类型 4**（在 ... 处**计算**的程序段搜索）
2. 通过收集辅助功能搜索目标程序段
3. 在找到搜索目标 ⇒ 显示报警 10208”通过 NC 启动继续执行程序“后停止
4. 用户：通过 NC 启动执行动作程序段 ⇒ DB21, ... DBX7.1 = 1（NC 启动）
5. 执行动作程序段
6. 切换至最后一个动作程序段 ⇒ 自动启动 /_N_CMA_DIR/_N_PROG_EVENT_SPF（缺省设置）作为 ASUB
7. 切换至最后一个 ASUB 程序段（REPOSA）⇒ DB21, ... DBX32.6 = 1（最后的动作程序段生效）
8. **可选**：通过 PLC 用户程序执行用户专用请求
9. 显示报警 10208“通过 NC 启动继续程序程序”

   说明

   MD11450 $MN_SEARCH_RUN_MODE，位 0 == 1 时，通过 PLC 用户程序使能（DB21, ... DBX1.6 = 1（PLC 动作结束））后才输出报警 10208。

10. 用户：通过 NC 启动继续执行程序 ⇒ DB21, ... DBX7.1 = 1（NC 启动）

10.7.4 级联程序段搜索

功能

通过“级联程序段搜索”功能可从“找到搜索目标”状态启动另一个程序段搜索。每次找到搜索目标后可任意级联搜索，其可用于以下程序段搜索功能：

- 程序段搜索类型 1（不带计算）
- 程序段搜索类型 2（在轮廓处计算）
- 程序段搜索类型 3（在程序段终点计算）

说明

只有当查找到搜索目标后，才可以从停止的程序执行开始启动另一个“级联程序段搜索”。

激活

“级联程序段搜索”在现有机床数据中配置：
MD11450 $MN_SEARCH_RUN_MODE

- 通过位 3 = 0（FALSE）使能级联程序段搜索
（即可设定多个搜索目标）。
- 考虑到兼容性，也可通过位 3 = 1（TRUE）禁用级联程序段搜索。预设为：级联程序段搜索位 3 = 0。

过程特性

找到搜索目标，重新启动程序段搜索

到达搜索目标后，程序执行停止，并将搜索目标作为当前程序段显示。每次找到搜索目标后，可任意次数地重复新的程序段搜索。

修改搜索目标设定

在每次开始程序段搜索前，可对搜索目标设定和程序段搜索功能进行修改。

示例：含级联程序段搜索的执行序列

- 复位
- 程序段搜索，找到搜索目标 1

- 程序段搜索，找到搜索目标 2 →“级联程序段搜索”
- 通过 NC 启动输出动作程序段 → 报警 10208
- NC 启动 → 继续执行程序

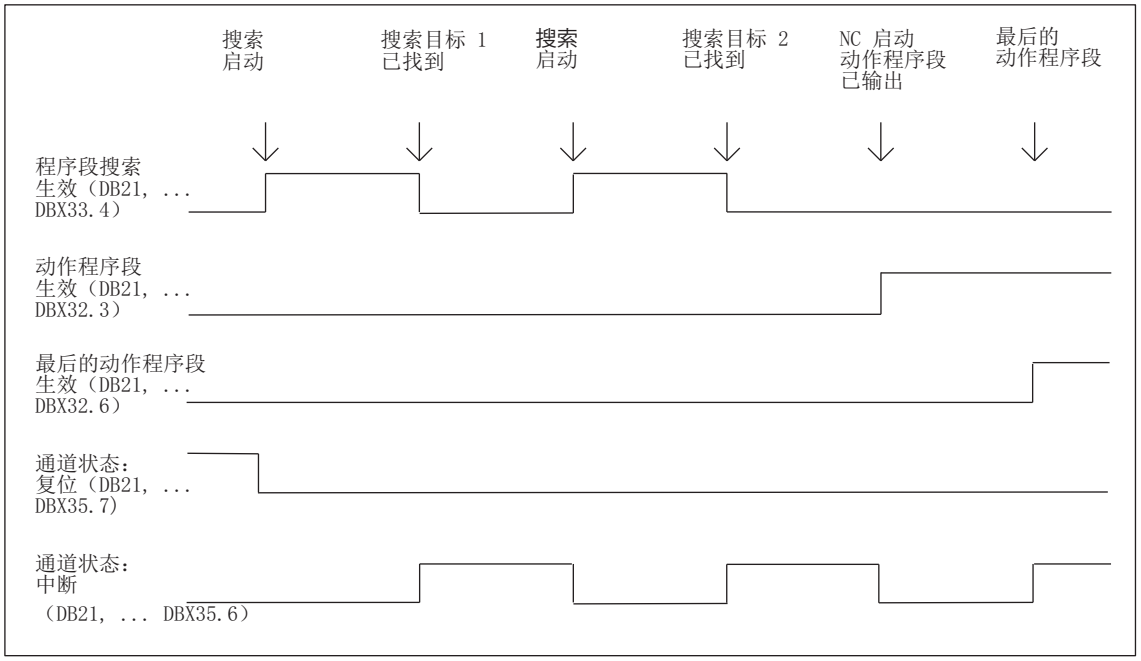


图 10-2　　接口信号的时间顺序

10.7.5　带计算的程序段搜索的示例

选择

参考以下示例，选择最能应对您的问题的程序段搜索类型。

类型 4：在程序段终点计算的程序段搜索

示例包含刀具管理生效时程序段搜索后的自动换刀：

1. 设置机床数据：
   MD11450 $MN_ SEARCH_RUN_MODE 设为 1
   MD11602 $MN_ASUP_START_MASK 位 0 = 1（从停止状态启动 ASUB）
2. 从 PLC 通过 FB4 选择 ASUB“SUCHLAUF_ENDE”（另见“P3：SINUMERIK 840D sl 的 PLC 基本程序 (页 945)”章节）。
3. 载入和选择零件程序“WERKSTUECK_1”。
4. 程序段搜索编号为 N220 的程序段的终点。

5. HMI 发送“找到搜索目标”信息。
6. 通过 NC 启动输出动作程序段。
7. 通过 PLC 信号：
   DB21...DB32.6（最后一个动作程序段生效）
   由 PLC 通过 FC9 启动 ASUB“SUCHLAUF_ENDE”（另见“P3：SINUMERIK 840D sl 的 PLC 基本程序 (页 945)”章节）。
8. ASUB 结束（例如可通过待定义 M 功能 M90 分析，参见示例程序段 N1110）后，PLC 置位信号：
   DB21, ... DBX1.6（PLC 动作结束）。
   或者可查询 NC/PLC 接口信号：
   DB21-DB30 DBB318 位 0（ASUB 已停止）。
   这样一来会显示报警 10208，也就是说现在可由操作人员执行其它动作。
9. 操作人员进行手动调整（JOG、JOG-REPOS、溢出转存）。
10. 通过 NC 启动继续执行零件程序。

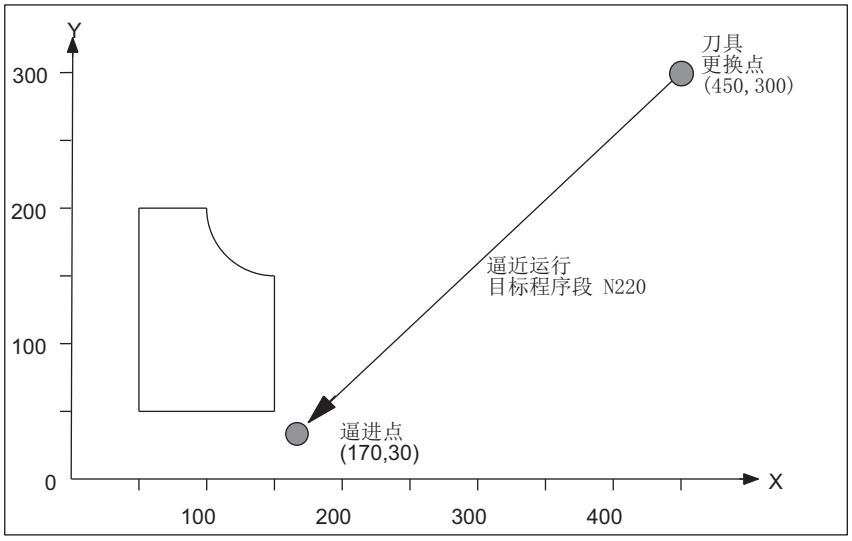


图 10-3　　程序段搜索程序段终点时的定位运行（目标程序段 N220）

说明

采用目标程序段为 N220 的“在轮廓处计算的程序段搜索”时，会生成至换刀点（目标程序段的起点）的定位运行。

类型 2：在轮廓处计算的程序段搜索

示例包含刀具管理生效时程序段搜索后的自动换刀：

第 1 步至第 3 步 如程序段搜索类型 4 的示例

第 4 步 沿轮廓搜索程序段编号 N260

第 5 步至第 10 步 如程序段搜索类型 4 的示例

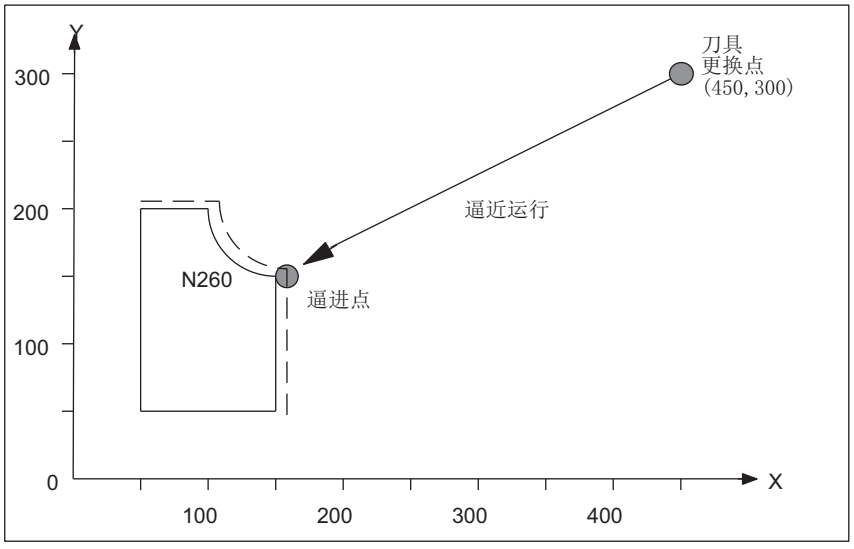


图 10-4 沿轮廓进行程序段搜索时的定位运行（目标程序段 N260）

说明

目标程序段为 N260 的“搜索程序段终点”会触发报警 14040（圆弧终点错误）。

类型 4 和类型 2 的零件程序

PROC WERKSTUECK_1

程序代码	注释
; 主程序	
...	
; 使用刀具“FRAESER_1”加工轮廓段 1	
...	

程序代码	注释
N100 G0 G40 X200 Y200	; 取消半径补偿
N110 Z100 D0	; 取消长度补偿
; 轮廓段 1 结束	
;	
; 使用刀具**"FRAESER_2"**加工轮廓段 2	
N200 T="FRAESER_2"	; 预选择刀具
N210 WZW	; 调用换刀程序
N220 G0 X170 Y30 Z10 S3000 M3 D1	; 轮廓段 2 的定位程序段
N230 Z-5	; 进刀
N240 G1 G64 G42 F500 X150 Y50	; 轮廓起始点
N250 Y150	
N260 G2 J50 X100 Y200	
N270 G1 X50	
N280 Y50	
N290 X150	
N300 G0 G40 G60 X170 Y30	; 取消半径补偿
N310 Z100 D0	; 取消长度补偿
轮廓段 2 结束	
...	
M30	
PROC WZW	
; **换刀程序**	
N500 DEF INT TNR_AKTIV	; 生效 T 号的变量
N510 DEF INT TNR_VORWAHL	; 预选 T 号的变量
N520 TNR_AKTIV = $TC_MPP6[9998,1]	; 读取生效刀具的 T 号
N530 GETSELT(TNR_VORWAHL)	; 读取预选刀具的 T 号
;	
; 仅当刀具尚未生效时才执行换刀	
N540 IF TNR_AKTIV == TNR_VORWAHL GOTOF ENDE	
N550 G0 G40 G60 G90 SUPA X450 Y300 Z300 D0	; 定位至换刀点
N560 M6	; 执行换刀
;	
ENDE:M17	
PROC SUCHLAUF_ENDE SAVE	
; **ASUB**，用于在程序段搜索后调用换刀程序	
N1000 DEF INT TNR_AKTIV	; 生效 T 号的变量
N1010 DEF INT TNR_VORWAHL	; 预选 T 号的变量
N1020 DEF INT TNR_SUCHLAUF	; 程序段搜索中获取的 T 号 ; 的变量
N1030 TNR_AKTIV = $TC_MPP6[9998,1]	; 读取生效刀具的 T 号
N1040 TNR_SUCHLAUF = $P_TOOLNO	; 读取通过程序段搜索获取的 T 号
N1050 GETSELT(TNR_VORWAHL)	; 读取预选刀具的 T 号
N1060 IF TNR_AKTIV ==TNR_SUCHLAUF GOTOF ASUP_ENDE	
N1070 T = $TC_TP2[TNR_SUCHLAUF]	; 通过刀具名称选择 T

程序代码	注释
N1080 WZW	; 调用换刀程序
N1090 IF TNR_VORWAHL == TNR_SUCHLAUF GOTOF ASUP_ENDE	
N1100 T = $TC_TP2[TNR_VORWAHL]	; 通过刀具名称恢复 T 预选
ASUP_ENDE:	
N1110 M90	; 发送给 PLC 的反馈信息
N1120 REPOSA	; ASUB 结束

10.7.6 前提条件

10.7.6.1 压缩器功能（COMPON、COMPCURV、COMPCAD）

- 在程序段搜索**类型 2** 或**类型 4**（在 ... 处**计算**的程序段搜索）中，若目标程序段所处于的程序部分中激活了压缩器功能（COMPON、COMPCURV、COMPCAD），那么在重定位至轮廓时会逼近通过**压缩器**计算出的轨迹上的位置。 这些位置不必与零件程序中编写的轨迹上的位置精确一致。
- 若压缩时省去了零件程序中编写的程序段，那么在程序段搜索中将无法将这些程序段作为目标程序段找到 ⇒ 报警 15370“未找到搜索目标”。

10.8 程序段搜索类型 5（SERUPRO）

10.8.1 功能说明

通过程序段搜索类型 5，即“程序测试”模式下的带计算的程序段搜索（SERUPRO，**“Se**arch-**Run** by **Pro**gramtest”），可在选择的中断点实现带计算的跨通道程序段搜索。为此，在 SERUPRO 期间，根据当前的程序协调指令，系统会测定在中断通道中继续执行程序所需的所有状态数据，并于随后将 NC 和 PLC 调整为继续执行程序所需的状态。

在随后继续执行程序重新定位至轮廓前，可通过一个自动启动的用户专用 ASUB 生成可能需要的所有初始状态。

通道

配合 HMI 使用时，SERUPRO 可应用于以下通道范围：

- 仅用于当前的 SERUPRO 通道 (1)
- 用于所有与 SERUPRO 通道具有相同工件名称的通道 (2)

- 用于所有与 SERUPRO 通道归属于相同 BAG 的通道 (3)
- 用于 NCU 的所有通道 (4)

对 SERUPRO 的通道范围选择在 **maschine.ini** 文件中的 [BlockSearch] 部分进行：

[BlockSearch] 部分	**为 HMI 使能程序段搜索功能和选择程序段搜索配置**
SeruproEnabled=1	; SERUPRO 软键在 HMI 中可用。缺省值为 (1)
SeruproConfig=1	; 上文所述通道分组的编号 (1) 至 (4) 。缺省值为 (1)

通过 SERUPRO 启动的所有其他通道均在“自执行 Serupro”模式下运行。只有选择了目标程序段的通道才可通过 SERUPRO 模式下的程序段搜索启动。

激活

SERUPRO 通过 HMI 激活。通过“程序测试轮廓”软键操作 SERUPRO。

SERUPRO 使用 REPOS 定位至目标程序段。

SERUPRO 的时序过程

1. 通过 HMI 操作“程序测试轮廓”软键和搜索目标。
2. NC 在“程序测试”模式下自行启动所选择的程序。
   - 轴在此过程中不运行。
   - 输出辅助功能 $A_OUT 和直接 PLC-IO。
   - 目标程序段的辅助功能则不输出。
3. NC 在目标程序段开始处停止，于系统内部取消程序测试，并显示停止条件“等待：找到搜索目标”。
4. 若存在用户专用 ASUB“PROG_EVENT.SPF”，则其会自动启动。
5. 下一次 NC 启动时会重新定位至轮廓（REPOS）。
   REPOS 进程通过一个系统 ASUB 实现，并可通过“可编辑的 ASUB”功能扩展。

程序段搜索 SERUPRO 的前提条件

SERUPRO 功能只允许在“AUTOMATIC”运行方式下激活，以及在程序状态（通道状态 RESET）下终止。

若在普通模式下仅 PLC 同时启动多个通道，则可在每个通道中通过 SERUPO 对其进行仿真。

采用以下机床数据设置时：
MD10708 $MN_SERUPRO_MASK，位 1 = 0

零件程序指令 START 会使仿真终止，并触发报警 16942
“通道 %1 启动程序指令动作 %2<ALNX> 不可用”。

机床数据：
MD10707 $MN_PROG_TEST_MASK
支持在停止状态下取消程序测试，并且不会影响 SERUPRO 进程。采用缺省设置时，只允许在 RESET 状态下取消功能。

说明

取消程序测试后 REPOS 进程开始，此时 SERUPRO 定位时的限制条件同样适用。可通过一个 ASUB 来消除负面效应。

调整 SERUPRO 特性

对于下列功能示例，可针对 NC 设定 SERUPRO 特性：

- 编程的停止（M0）
- 程序协调指令 START
- 组 SERUPRO
- 跨通道结束 SERUPRO
- 倍率

MD10708 $MN_SERUPRO_MASK = <SERUPRO 中的特性>

SERUPRO 的通道专用初始设置

在一般情形下，可通过以下机床数据定义零件程序启动后的通道专用初始设置：

MD20112 $MC_START_MODE_MASK= <初始设置>

可为 SERUPRO 设定独立的初始设置，来代替 MD20112 中的初始设置：

MD22620 $MC_START_MODE_MASK_PRT = <SERUPRO 初始设置>

SERUPRO 初始设置必须通过以下机床数据显性使能：

MD22621 $MC_ENABLE_START_MODE_MASK_PRT = 1

NC/PLC 接口信号：“程序测试模式下的程序段搜索生效”

程序测试模式下的程序段搜索通过 NC/PLC 接口信号显示：
DB21, ... DBX318.1 == 1

该接口信号在程序段搜索启动时置位，直至切换至目标程序段进入主处理。

对于 SERUPRO 进程后的用户自定义 ASUB

提示

若机床制造商决定在 SERUPRO 后按照上文介绍的第 7 点启动一个 ASUB，则须注意以下事项：

第 6 点后的停止状态：

机床数据：
MD11602 $MN_ASUP_START_MASK
和
MD11604 $MN_ASUP_START_PRIO_LEVEL
可支持 NC 从停止状态自行通过 FC9 模块启动 ASUB。

REPOS 程序段结束后再应答 FC9：

REPOS 程序段也已结束时，才可由 FC9 模块通过“ASUB Done”提示 ASUB 结束。

设定的 REPOS 进程在第 8 点后取消：

ASUB 启动时，设定的 REPOS 进程会被取消！

因此，若需保持 REPOS 进程，则应通过 REPOSA 完成 ASUB。

删除不需要的 REPOS 进程：

通过 `M17` 或 `RET` 结束 ASUB，从而删除不需要的 REPOS 进程。

对 ASUB 的特别处理：

对通过 REPOS 结束和从停止状态启动的 ASUB，原则上需要特别处理。

ASUB 会在 REPOS 程序段前自行停止，并通过以下信号予以显示：

DB21, ... DBX318.0（ASUB 已停止）

自动 ASUB 启动

采用以下机床数据设置时：
MD11450 $MN_SEARCH_RUN_MODE，位 1 = 1
系统会按照以下流程在 SERUPRO 定位运行中自动启动
/_N_CMA_DIR/_N_PROG_EVENT_SPF
路径下的 ASUB。

1. SERUPRO 进程已完全执行。
2. 用户触发“NC 启动”。

3. 启动 ASUB。
4. NC 在零件程序指令 REPOS **前**自动停止，并显示信息“需通过 NC 启动继续执行程序”。
5. 用户再次按下“NC 启动”。
6. NC 执行重定位运行，并从目标程序段起继续执行零件程序。

说明

通过 MD11450 进行的自动 ASUB 启动需要按下 **Starts** 用于程序的继续执行。
这样一来，其流程特性便与其它程序段搜索类型近似。

10.8.2 重新定位至轮廓（REPOS）

借助“重新定位至轮廓”（REPOS）功能可在中断位置上继续进行已中断的加工。与 REPOS 不同的是，SERUPRO 可“还原”或“重复”程序段。为此，在 SERUPRO 找到目标程序段后，系统会定位至可通过 REPOS 模式选择的位置处的轮廓并继续进行加工。

SERUPRO：设置 REPOS 特性

REPOS 特性（即重定位程序段中的特性）是通过以下机床数据设置的：
MD11470 $MN_REPOS_MODE_MASK = <REPOS 模式>

<REPOS 模式>		
位	**值**	**含义**
0	0	重复已中断的暂停时间
	1	继续已中断的暂停时间
1	-	预留
2	0	重定位程序段中**不会**考虑 DB31, ... DBX10.0（REPOS 延时）。
	1	重定位程序段中会考虑 DB31, ... DBX10.0（REPOS 延时）。
3	0	**SERUPRO**：在重定位程序段中只运行轨迹轴
	1	**SERUPRO**：在重定位程序段中同时运行轨迹轴和定位轴
4	0	**REPOS**：在重定位程序段中只运行轨迹轴
	1	**REPOS**：在重定位程序段中同时运行轨迹轴和定位轴
5	0	中断期间，经过修改的进给率和主轴转速只会从中断位置后的第一个零件程序段起生效
	1	中断期间，经过修改的进给率和主轴转速从中断位置起生效，故其已在剩余程序段中生效并因此被优先使用。此特性针对每个 REPOS 进程。

<REPOS 模式>		
位	值	含义
6	0	SERUPRO：中性轴和定位主轴在重定位程序段中作为轨迹轴运行。
	1	SERUPRO：中性轴和定位主轴在重定位程序段中作为指令轴运行。 中性轴和定位主轴在 SERUPRO 后重定位。 对于不允许重定位的中性轴而言，必须将 REPOS 延时置位。DB31, ... DBX10.0 = 1（REPOS 延时）
7	0	REPOS 延时未使能。
	1	REPOS 延时已使能。 激活了 REPOS 延时（DB31, ... DBX10.0 == 1）且并非几何轴或定向轴的轴在重定位时不运行。

⚠ 小心

碰撞危险

MD11470 $MN_REPOS_MODE_MASK，位 3 或位 4 = 1
用户有责任确保在重定位程序段中同时进行轴重定位时，轴不会在机床上产生碰撞。

通过受控的 REPOS 进行重定位

可通过 NC/PLC 接口设定轨迹轴的 REPOS 模式：

DB21, ... DBX31.0 - 2（REPOS 模式）

REPOS 模式在 NC 程序中编写，并决定定位特性（参见章节“通过受控 REPOS 重定位至轮廓 (页 572)”）。

各轴的 REPOS 特性也可通过 NC/PLC 接口信号控制，并且必须通过以下机床数据使能：
MD11470 $MN_REPOS_MODE_MASK.位 2 = 1

轨迹轴无法单独控制。对于其他所有非几何轴的轴，可针对单个轴暂时抑制或推延重定位。对于 REPOS 准备运行的通道轴，可通过 NC/PLC 接口信号在稍后的时间点重新使能或继续禁用。

⚠ 危险

碰撞危险

在选择了“抑制单个轴的重定位”的情况下，通过信号 DB31, ... DBX2.2（删除剩余行程）会引发以下危险特性：

MD11470 $MN_REPOS_MODE_MASK.位 2 == 1

若在中断后采用增量方式编写了轴，那么 NC 定位至的位置将与未发生中断时不同。

参见下例：以增量方式编写轴

示例：以增量方式编写回转轴 A

回转轴 A 为第四机床轴。

- 在 REPOS 进程前，回转轴 A 位于 11°
  在中断程序段中，即在 SERUPRO 的目标程序段中，需要将回转轴 A 运行至 27°。
  在任意数量的程序段后，将回转轴 A 增量运行 5°：
  ```
  N1010 POS[A]=IC(5) FA[A]=1000
  ```
  在接口信号 DB34 DBX10.0 = 1（REPOS 延时）置位的情况下，回转轴 A 在 REPOS 进程中不运行，并通过 N1010 向 32° 运行。必要时用户须对从 11° 向 27° 的行程进行应答。

  ⚠ 危险

  碰撞危险

  由于在中断后以增量方式编写轴，其向 16° 运行，而非 32°。

- 单独启动轴
  包含多轴的 SERUPRO 定位的 REPOS 特性通过以下机床数据选择：
  MD11470 $MN_REPOS_MODE_MASK.位 3 = 1
  NC 通过一个程序段启动 SERUPRO 定位，其会将**所有**定位轴运行至编写的终点，并将轨迹轴运行至目标程序段。
  用户通过选择相应进给使能来启动单个轴。接下来便是目标程序段的运行。

- 在重定位程序段中对定位轴进行重定位
  定位轴不在剩余程序段中，而是在重定位程序段中进行重定位，并且不仅仅适用于程序测试下的程序段搜索中的 SERUPRO 定位：
  MD11470 $MN_REPOS_MODE_MASK.位 3 = 1：用于程序测试模式下的程序段搜索（SERUPRO）
  MD11470 $MN_REPOS_MODE_MASK.位 4 = 1:用于每个 REPOS

  说明

  若位 3 和位 4 均未置位，那么此阶段会在剩余程序段中对非轨迹轴的轴进行重定位。

通过 REPOS 偏移延迟轴定位

若轴专用接口信号 DB31, ... DBX10.0（REPOS 延时）置位，则在下一次编写时才会通过 DB21, ... DBX31.4（REPOS 模式变更）的上升沿运行该轴的 REPOS 偏移。

可使用同步动作通过 $AA_REPOS_DELAY 确定该轴当前是否处于 REPOS 偏移下。

⚠ 小心
碰撞危险 DB31, ... DBX10.0（REPOS 延时）对形成轨迹的机床轴无效。 可通过 DB31, ... DBX76.4（轨迹轴）确定轴是否为轨迹轴。

REPOS 信号的接收时间点

通过 DB21, ... DBX31.4（REPOS 模式变更）的上升沿将以下 REPOS 信号接收至 NC：

- 通道专用：DB21, ... DBX31.0 - 2（REPOS 模式）
- 轴专用：DB31, ... DBX10.0（REPOS 延时）

REPOS 信号的电平基于当前的主处理程序段。此时可分为以下两种情形：

1. 主处理中存在**一个**当前生效 REPOS 进程的重定位程序段。
   运行中的 REPOS 进程会终止并重新启动，且 REPOS 偏移通过上述 REPOS 信号调整。
2. 主处理中**无**当前生效 REPOS 进程的重定位程序段。
   未来每一个需要重定位至当前主处理程序段的 REPOS 进程均通过上述 REPOS 信号调整。

说明

在运行的 ASUB 中，DB21, ... DBX31.4（REPOS 模式变更）不对最后的 REPOS 起作用，在执行 REPOS 程序段的时间点上，该信号随机置位。

在第 1 种情形中，信号只适用于停止状态。

RESET 时的特性：

- NC 已应答 PLC 信号：
  DB21, ... DBX31.4 == 1（REPOS 模式变更）**和**
  DB21, ... DBX319.0 == 1（REPOS 模式变更应答）
  若在此情形下发生通道复位，则系统将生效的 REPOS 模式删除：
  DB21, ... DBX319.1 - 3 = 0（生效的 REPOS 模式）
- NC 尚未应答 PLC 信号：
  DB21, ... DBX31.4 == 1（REPOS 模式变更）**和**
  DB21, ... DBX319.0 == 0（REPOS 模式变更应答）
  若在此情形下发生通道复位，则系统将 REPOS 模式变更应答和生效的 REPOS 模式删除：
  DB21, ... DBX319.0 = 0（REPOS 模式变更应答）
  DB21, ... DBX319.1 - 3 = 0（生效的 REPOS 模式）

通过 NC/PLC 接口信号控制 SERUPRO 定位

可通过 DB21, ... DBX31.4（REPOS 模式变更）和对应信号在以下阶段使用 SERUPRO 定位：

- 从“找到搜索目标”到“SERUPRO ASUB 启动”
- 从“SERUPO ASUB 在 REPOS 前自动停止”到“执行目标程序段”

执行 SERUPRO ASUB（例如在 REPOS 前的程序部分）时，接口信号对 SERUPRO 定位不生效。

包含 NC/PLC 接口信号的 REPOS 进程

通过 NC/PLC 接口信号控制 REPOS

可通过以下 NC/PLC 接口信号控制 REPOS 偏移：

- DB21, ... DBX31.0 - 2（REPOS 模式）
- DB21, ... DBX31.4（REPOS 模式变更）
- DB31, ... DBX10.0（REPOS 延时）
- DB31, ... DBX72.0（REPOS 延时）

REPOS 应答信号

使用以下 NC/PLC 接口信号**通过 NC** 对借助 PLC 调整 REPOS 特性的功能进行应答：

- DB21, ... DBX319.0（REPOS 模式变更应答）
- DB21, ... DBX319.1 - 3（生效的 REPOS 模式）
- DB21, ... DBX319.5（REPOS 延时应答）
- DB31, ... DBX70.0（REPOS 偏移）
- DB31, ... DBX70.1（REPOS 偏移生效）
- DB31, ... DBX70.2（REPOS 延时应答）
- DB31, ... DBX76.4（轨迹轴）

更多相关信息请见“接口中的 REPOS 偏移”部分。

REPOS 应答进程

若 NC 识别出“REPOS 模式变更”（DB21, ... DBX31.4 == 1），则借助 PLC 通过 DB21, ... DBX319.0 = 1 对其进行应答。

说明

若 NC 尚未通过接口信号 DB21, ... DBX319.0（REPOS 模式变更应答）对接口信号 DB21, ... DBX31.4（REPOS 模式变更）进行应答，
那么在此情形下的通道复位会使程序终止，并不会执行 REPOS 来控制 REPOS 模式。

NC 通过以下接口信号应答由 PLC 设定的 REPOS 模式：

- DB21, ... DBX319.1 - 3（生效的 REPOS 模式）
- DB31, ... DBX10.0（REPOS 延时）
- DB31, ... DBX70.2（REPOS 延时应答）

示例

- 时间点 ②：在程序段 N20 中通过 NC 停止将 NC 程序停止。所有轴通过为其设置的制动斜坡制动至静止状态。
- 时间点 ③：在 PLC 用户程序将“REPOS 模式”置位后，NC 通过“REPOS 模式变更”的 0/1 脉冲沿接收 REPOS 模式。
- 时间点 ④：“REPOS 模式变更应答”始终保持置位，直至触发 ASUB。
- 时间点 ⑤：ASUB 中的 REPOS 进程开始。
- 时间点 ⑥：重新转向 ASUB 的剩余程序段。

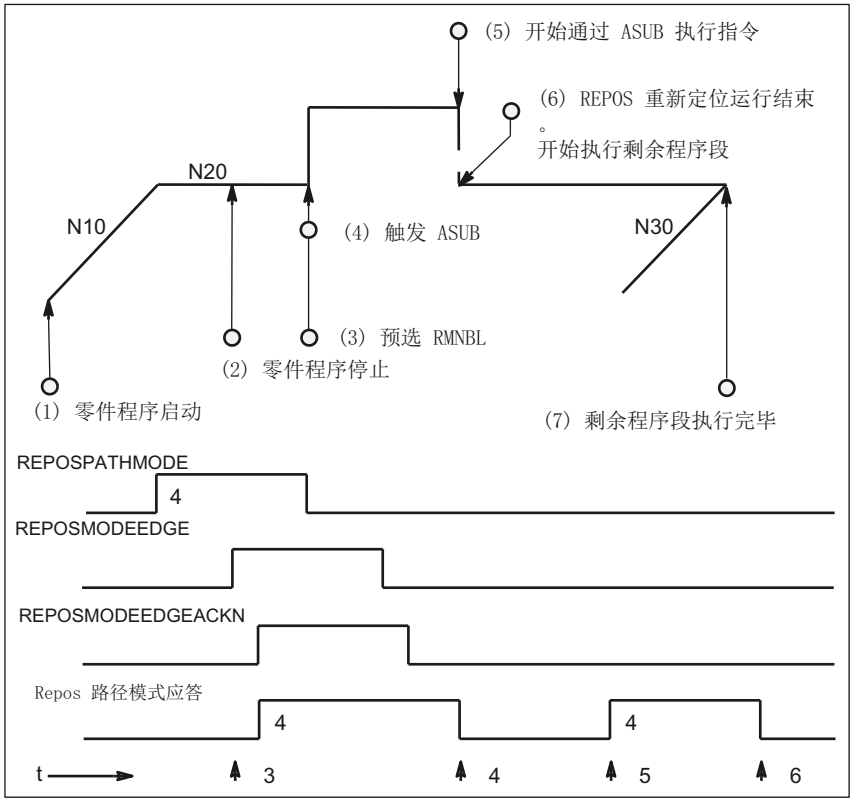


图 10-5　　零件程序中的 REPOS 流程，以及按时序排列的 NC 应答信号

NC 重新设置应答

此阶段中 REPOSPATHMODE 继续生效（→ 时间点 (2) 停止的程序的剩余程序段尚未执行完毕）。

一旦执行 ASUB 的 REPOS 重定位运行，NC 便会立即重新设置 “Repos Path Mode Quitt”（→ 时间点 (5)）。若未预先通过 NC/PLC 接口信号选择 REPOSPATHMODE，则会显示编写的 REPOS 模式。

“Repos Path Mode Quitt” 会在切换至剩余程序段时撤销（→ 时间点 (6)）。
→ 时间点 (2) 处后续的零件程序段 N30 将继续执行。

接口信号：
DB31, ... DBX70.2（REPOS 延时应答）以类似的方式定义。

DB31, ... DBX70.1（REPOS 偏移生效）= 1，若：

DB21, ... DBX319.1-319.3（生效的 REPOS 模式）= 4 (RMNBL).

有效的 REPOS 偏移

SERUPRO 进程结束时，用户可通过轴/主轴专用 NC/PLC 接口信号（NC→PLC）：
DB31, ... DBX70.0（REPOS 偏移）读取 REPOS 偏移。

此信号对轴的作用如下：

值 0： 不启用 REPOS 偏移。

值 1： 启用 REPOS 偏移。

有效范围

接口信号：
DB31, ... DBX70.0（REPOS 偏移）
在 SERUPRO 进程结束时提供。

启动 SERUPRO ASUB 或自动启动 ASUB 时 REPOS 偏移失效。

更新有效范围内的 REPOS 偏移

在 SERUPRO 结束和启动之间，可通过模式切换在 JOG 下运行轴。

用户需要在 JOG 模式下手动运行 REPOS 偏移，从而将 NST
DB31, ... DBX70.0（REPOS 偏移）设为 0 值。

在有效范围内也可通过 FC18 运行轴，此时 NST
DB31, ... DBX70.0（REPOS 偏移）会持续更新。

显示有效范围

REPOS 偏移的有效范围通过以下接口信号显示：

DB31, ... DBX70.1（REPOS 偏移生效）

该信号可指示计算的有效性：

值 0： 该轴的 REPOS 偏移计算正确。

值 1： 该轴的 REPOS 无法计算，因为尚未进行 REPOS（例如处于 ASUB 结尾）或无 REPOS 生效。

跨通道取轴后的 REPOS 偏移

通过汇总信号 DB21, ... DBX319.5（REPOS 延时）可确定是否发生了有效的 REPOS 偏移：

值 0： 该通道当前控制的所有轴无 REPOS 偏移，或其 REPOS 偏移无效。

值 1： 其它。

经过同步的同步主轴耦合中的 REPOS 偏移

在通过 SERUPRO 重定位时，会先预处理至中断位置。若已对同步主轴耦合进行同步，那么将不存在跟随轴的 REPOS 偏移，也不会有同步行程。同步信号保持置位。

程序段切换时找到搜索目标

当轴归属于轨迹组时，轴专用 NC/PLC 接口信号 DB31, ... DBX76.4（轨迹轴）为 1。

此信号用于显示程序段切换时当前待执行程序段的状态。之后的状态变化则不考虑。

SERUPRO 进程通过“找到搜索目标”结束时，DB31, ... DBX76.4（轨迹轴）基于目标程序段。

10.8.2.1 通过受控 REPOS 重定位至轮廓

在通过 SERUPRO 找到目标程序段后，系统会在继续执行中断的程序前先执行一个重新定位至轮廓的 REPOS 进程。缺省设置下，REPOS 模式“重新定位至目标程序段的程序段起点” (RMBBL) 生效。可通过以下 NC/PLC 接口针对特定用户设定 REPOS 模式：

DB21, ... DBX31.0 - .2（REPOS 模式）

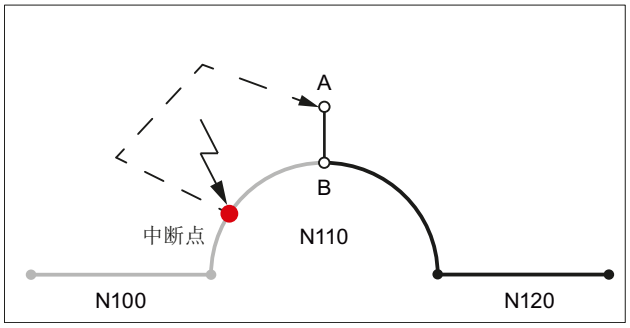
文档：
对接口信号的详细说明请参见 NC 变量参数手册和接口信号。

REPOS 模式：重新定位至下一个轨迹点 (RMNBL)

在 REPOS 模式 RMNBL 下，系统会从 REPOS 起始位置开始定位至下一个轮廓点。

示例

程序在程序段 N110 中的任意位置中断。随后例如以手动方式将轴运行至位置（A）。在 SERUPRO 找到目标程序段 N110 后，以 REPOS 模式 RMNBL 执行 REPOS 进程。从 REPOS 起始位置（A）出发，下一轮廓点为点（B）。到达点（B）时，REPOS 进程完成。从点（B）起重新运行中断程序的编写的轮廓。



通过 NC/PLC 接口设定 REPOS 模式

REPOS 模式可通过以下 NC/PLC 接口信号设定：
DB21, ... DBX31.0 - .2（REPOS 模式）

说明

RMNBL 是一个通用的 REPOS 扩展功能，不仅限用于 SERUPRO。

RMIBL 和 RMBBL 的作用与 SERUPRO 的一致。

DB21, ... DBX31.0 - .2（REPOS 模式）只会影响轨迹轴的运行。

其它轴的特性可通过接口信号 DB31, ... DBX10.0（REPOSDELAY）单独修改。此 REPOS 偏移不会立即运行，而是在下一次编写时才运行。

重定位点编程的更多信息请见：

文档：
编程手册，工作准备；轨迹特性，章节：重新定位至轮廓

10.8.3 搜索加速

机床数据设定

SERUPRO 进程的执行速度可通过以下机床数据调整

MD22600 $MC_SERUPRO_SPEED_MODE = <值>

<值>	含义
0	程序测试采用程序段搜索速度/空运行进给速度： ● 进给轴 MD22601 $MC_SERUPRO_SPEED_FACTOR * 空运行进给率 ● 主轴：MD22601 $MC_SERUPRO_SPEED_FACTOR * 编写的转速 不考虑轴/主轴的动态限制。
1	程序测试采用编写的速度： ● 进给轴空运行进给 ● 主轴：编写的转速 遵循轴/主轴的动态限制。
2	程序测试采用空运行进给率 程序测试下以编写的速度/转速运行。 遵循轴/主轴的动态限制。
3	程序测试采用程序段搜索速度 程序测试下采用以下速度运行： ● 进给轴 MD22601 $MC_SERUPRO_SPEED_FACTOR * 编写的进给率 ● 主轴：MD22601 $MC_SERUPRO_SPEED_FACTOR * 编写的转速 不考虑轴/主轴的动态限制。 **提示** 旋转进给率（例如 G95）生效时，编写的进给率不与 MD22601 $MC_SERUPRO_SPEED_FACTOR 相乘，只采用编写的主轴转速。这样一来有效轨迹速度会提升 MD22601 $MC_SERUPRO_SPEED_FACTOR。

前提条件

主处理轴

SERUPRO 中 MD22600 $MC_SERUPRO_SPEED_MODE 对下列主处理轴生效：

- PLC 轴
- 指令轴
- 定位轴
- 往复轴

同步动作

注意
SERUPRO 中有时不会执行同步动作的动作 SERUPRO 期间系统内部生成的实际值（例如轴位置）与常规程序运行中不同，因此系统可能会认定验证实际值（例如轴位置）的同步动作条件不再为真（TRUE），并因而不再执行同步动作的动作部分。

旋转进给率

DryRun 下 MD22601 $MC_SERUPRO_SPEED_FACTOR 的作用：

- 从 G95/G96/G961/G97/G971 切换至 G94
- 攻丝和螺纹切削：正常的 DryRun 速度

刚性攻丝

- “刚性攻丝”（G331/G332）时，主轴在位置闭环控制下以轨迹组插补。此时会设定钻孔深度（直线轴）、螺纹螺距和转速（主轴）。
  DryRun 期间会设定直线轴的速度，转速保持恒定，螺纹螺距会被相应调整。
  SERUPRO 后得到的主轴位置会与常规运行有偏差，因为 SERUPRO 中主轴旋转相对较少。

10.8.4 SERUPRO ASUB

SERUPRO ASUB 的特殊性

使用 SERUPRO ASUB 时须注意以下方面的特殊性：

- 回参考点运行：通过零件程序 G74
- 刀具管理：换刀和刀库数据
- 主轴加速：启动 SERUPRO ASUB 时

G74 回参考点运行

若程序起点和搜索目标之间有 G74（回参考点运行）指令，则会被 NC 忽略。

SERUPRO 定位不会考虑 G74 指令！

刀具管理

刀具管理生效时，建议采用以下设置：

MD20310 $MC_TOOL_MANAGEMENT_MASK 位 20 = 0

这样一来，SERUPRO 进程中产生的刀具管理指令便**不会**输出至 PLC！

刀具管理指令如下生效：

- NC 自动应答指令。
- 刀库数据不会被修改。
- 刀具数据不会被修改。
  特例：
  测试模式下激活的刀具可能会变为“生效”状态。因此 SERUPRO 进程后主轴上可能会有错误的刀具。
  解决办法：
  用户启动一个实际运行的 SERUPRO ASUB。启动前用户可启动一个 ASUB 来换入正确的刀具。

SERUPRO 进程：流程中第 2 点至第 6 点
SERUPRO ASUB：流程中第 7 点。

此外必须设置机床数据 MD20310 $MC_TOOL_MANAGEMENT_MASK 位 11 = 1，因为 ASUB 必要时须重复 T 选择。

配备刀具管理和副主轴的设备不支持 SERUPRO！

示例

换刀子程序

程序代码	注释
PROC L6	; 换刀程序
N500 DEF INT TNR_AKTUELL	; 生效 T 号的变量
N510 DEF INT TNR_VORWAHL	; 预选 T 号的变量
	; 确定当前刀具
N520 STOPRE	; 程序测试模式下
N530 IF $P_ISTEST	; 从程序上下文
N540 TNR_AKTUELL = $P_TOOLNO	; 读取“当前”刀具。
N550 ELSE	; 否则会读取主轴的刀具。
N560 TNR_AKTUELL = $TC_MPP6[9998,1]	; 读取主轴上刀具的 T 号
N570 ENDIF	

程序代码	注释
N580 GETSELT(TNR_VORWAHL)	；读取预选的主主轴刀具的 T 号
	；仅当刀具当前未生效时才执行换刀
N590 IF TNR_AKTUELL <> TNR_VORWAHL	；返回刀具更换点
N600 G0 G40 G60 G90 SUPA X450 Y300 Z300 D0	
N610 M206	；执行刀具更换
N620 ENDIF	
N630 M17	

ASUB，用于在程序段搜索类型 5 后调用换刀程序

程序代码	注释
PROC ASUPWZV2	
N1000 DEF INT TNR_SPINDEL	；生效 T 号的变量
N1010 DEF INT TNR_VORWAHL	；预选 T 号的变量
N1020 DEF INT TNR_SUCHLAUF	；程序段搜索中获取的 T 号的变量
N1030 TNR_SPINDEL = $TC_MPP6[9998,1]	；读取主轴上刀具的 T 号
N1040 TNR_SUCHLAUF = $P_TOOLNO	；读取通过程序段搜索获取的 T 号，
	；即该刀具决定
	；当前刀具补偿
N1050 GETSELT(TNR_VORWAHL)	；读取预选刀具的 T 号
N1060 IF TNR_SPINDEL ==TNR_SUCHLAUF GOTOF ASUP_ENDE1	
N1070 T = $TC_TP2[TNR_SUCHLAUF]	；通过刀具名称选择 T
N1080 L6	；调用换刀程序
N1085 ASUP_ENDE1:	
N1090 IF TNR_VORWAHL == TNR_SUCHLAUF GOTOF ASUP_ENDE	
N1100 T = $TC_TP2[TNR_VORWAHL]	；通过刀具名称恢复 T 预选
N1110 ASUP_ENDE:	
N1110 M90	；发送给 PLC 的反馈信息
N1120 REPOSA	；ASUB 结束

在 PROC L6 和 PROC ASUPWZV2 这两个程序中，换刀均采用 M206 取代 M6 编写。ASUB 程序“ASUPWZV2”使用了各种系统变量，其一方面能够识别程序进度（$P_TOOLNO），另一反面又能显示机床的当前状态（$TC_MPP6[9998,1] ）。

主轴启动

启动 SERUPRO ASUB 时，主轴不会加速至程序中设定的转速，因为 SERUPRO ASUB 用于在换刀后将新道具校准至正确的工件位置。

通过 SERUPRO ASUB 如下执行主轴加速：

- SERUPRO 进程已完全结束。
- 用户通过 FC-9 模块启动 SERUPRO ASUB，
  用于在必要时加速主轴。
- ASUB 中 M0 后的启动不会修改主轴状态。
- SERUPRO ASUB 在 REPOS 零件程序段前自动停止。
- 用户按下 START。
- 只要 ASUB 中未对主轴进行其它编程，主轴便会加速至目标程序段状态。

说明

调整主轴 REPOS：
在调整 SERUPRO 定位和主轴功能时，须将转速控制模式和定位模式的过渡纳入考量。
主轴运行方式切换的更多信息请见“运行方式 (页 1402)”章节。

10.8.5 自动执行的 SERUPRO

自动执行的 SERUPRO

通过使用通道专用功能“自动执行的 SERUPRO”，**无需**在相关 SERUPRO 通道的程序中事先定义搜索目标便可实现 SERUPRO。

此外可为**每个**“自动执行的 SERUPRO”定义一个“serurpoMasterChan”特殊通道。可在该通道中定义一个搜索目标。

“自动执行的 SERUPRO”功能支持跨通道的 SERUPRO 程序段搜索。

功能

“自动执行的 SERUPRO”进程无法找到搜索目标。若未到达搜索目标，那么通道也不会停止。不过特定情形下通道会暂时停止。此时该通道通常会等待另一个通道。此类情形例如包括：等待标记、耦合或跨通道取轴。

等待阶段发生：

在此等段阶段中，NC 会检查“seruproMasterChan”通道是否已到达一个搜索目标。未到达搜索目标时会再次退出等待阶段。

若到达了搜索目标，
那么 SERUPRO 进程在该通道中亦结束。“serupro-MasterChan”通道须已在通常 SERUPRO 模式下启动。

无等待阶段发生：

“自动执行的 SERUPRO”通过零件程序的 M30 结束。

之后通道重新恢复复位状态。

不会发生 SERUPRO 运行。

启动一组通道

若只通过“自动执行的 SERUPRO”来启动一组通道，那么所有通道均通过“RESET”结束。

特例情形：
通道等待一个根本未启动的协作通道。

可如下执行跨通道程序段搜索：

- 用户通过 HMI 选择须共同运作的通道（通道组）。
- 用户从通道组中选取一个想为其选定搜索目标的重要通道（目标通道）。
- 这样一来 HMI 会在目标通道上启动 SERUPRO，并在通道组的其它通道上启动“自动执行的 SERUPRO”。

当涉及的**每条**通道均清除了“seruproActive”时，进程结束。

“自动执行的 SERUPRO”不接受另一个 NCU 上的主主轴。

激活

通过 HMI 激活“自动执行的 SERUPRO”，作为针对目标通道“seruproMasterChan”的程序段搜索类型 5 的程序段搜索启动。

对于通过目标通道启动的相关通道，无需为其设定搜索目标。

10.8.6 禁用程序部分用于恢复执行

编写的中断指示

由于生产或过程工艺的原因，若特定程序部分内发生程序中断时预计无法恢复执行，可为搜索的目标程序段禁用该程序部分。

在程序中断后，若在为恢复执行而禁用的程序部分中进行了对中断位置的程序段搜索，那么控制系统会将禁用区域前最后的可执行程序段（主处理程序段）用作目标程序段（停止程序段）。

编程

句法

IPTRLOCK()

功能

标记需要禁用恢复执行的程序部分的起点。对于“在中断位置恢复执行”的程序段搜索，下一个 IPTRLOCK 生效的可执行程序段（主处理程序段）将被用作目标程序段，直至通过 IPTRUNLOCK 使能。该程序段下文称作**停止程序段**。

有效性：模态有效

句法

IPTRUNLOCK()

标记为恢复执行禁用的程序部分的终点。对于“在中断位置恢复执行”的程序段搜索，从下一个 IPTRLOCK 生效的可执行程序段（主处理程序段）起，当前程序段重新用作目标程序段。该程序段下文称作**使能程序段**。

有效性：模态有效

示例

程序代码	注释
...	
N010 IPTRLOCK()	; 禁用区域：起始
N020 R1=R1+1	
N030 G4 F1	**; 停止程序段**
...	; 禁用区域
N200 IPTRUNLOCK()	; 禁用区域：结束
N220 R1=R1+1	
N230 G4 F1	**; 使能程序段**
...	

前提条件

- 程序（*.MPF，*.SPF）中的 IPTRLOCK 最多生效至程序结束（M30、M17、RET）。程序结束时 IPTRUNLOCK 会隐性生效。
- 在一个程序内多次编写 IPTRLOCK 时，效果不会累加。在程序内首次编写 IPTRUNLOCK 或达到程序末尾时，之前的所有 IPTRLOCK 调用将结束。
- 若在禁用区域内进行了子程序调用，那么恢复执行禁用同样对该子程序级及其后可能存在的所有程序级生效。即使在调用的子程序中显性编写 IPTRUNLOCK，禁用仍不会被取消。

示例：两个程序级中禁用程序部分的套用

通过在 PROG_1 中激活恢复执行禁用，同时对 PROG_2 及其后可能存在的所有程序级禁用恢复执行。

程序代码	注释
PROC PROG_1	; 程序 1
...	
N010 IPTRLOCK()	
N020 R1=R1+1	
N030 G4 F1	**; 停止程序段**
...	; 禁用区域：起始
N040 PROG_2	; 禁用区域
...	; 禁用区域：结束
N050 IPTRUNLOCK()	
N060 R2=R2+2	
N070 G4 F1	**; 使能程序段**
...	

程序代码	注释
PROC PROG_2	; 程序 2
N210 IPTRLOCK()	; 由于程序 1 而不生效
...	
N250 IPTRUNLOCK()	; 由于程序 1 而不生效
...	
N280 RET	; 由于程序 1 而不生效

示例 3：多重编写 IPTRLOCK

程序代码	注释
PROC PROG_1	; 程序 1
...	
N010 IPTRLOCK()	

程序代码	注释
N020 R1=R1+1	
N030 G4 F1	**；停止程序段**
...	；禁用区域：起始
N150 IPTRLOCK()	；禁用区域
...	；禁用区域
N250 IPTRLOCK()	；禁用区域
...	；禁用区域：结束
N360 IPTRUNLOCK()	
N370 R2=R2+2	
N380 G4 F1	**；使能程序段**
...	

系统变量

系统变量 $P_IPTRLOCK 可用于获取当前程序段的状态：

$P_IPTRLOCK	含义
FALSE	当前程序段**不处于**为恢复执行禁用的程序部分中。
TRUE	当前程序段**处于**为恢复执行禁用的程序部分中。

自动功能专用恢复执行禁用

对于各种耦合，恢复执行禁用可自动随耦合的启用/撤销而激活或取消：

MD22680 $MC_AUTO_IPTR_LOCK，位 x

位	值	含义
0	电子齿轮箱（EGON / EGOF）	
	1	自动恢复执行禁用生效
	0	自动恢复执行禁用不生效
1	轴向引导值耦合 (LEADON / LEADOF)	
	1	自动恢复执行禁用生效
	0	自动恢复执行禁用不生效

该程序区域从激活**前**的最后一个可执行程序段开始，并在取消处结束。

自动中断指示对通过同步动作激活或取消的耦合不生效。

示例:轴向引导值耦合自动定义为不可搜索：

程序代码	注释
N100 G0 X100	
N200 EGON(Y,"NOC",X,1,1)	；不可搜索的程序部分开始。
N300 LEADON(A,B,1)	
...	
N400 EGOFS(Y)	
...	
N500 LEADOF(A,B)	；不可搜索的程序部分结束。
N600 G0 X200	

不可搜索的程序区域（N200 - N500）中的程序中断总是通过 N100 提供中断指示。

注意
功能同时启用可导致异常 通过机床数据同时启用了“可编程中断指示”和“自动中断指示”时，NC 会选择其中最大的不可搜索区域。 一个程序可能在整个运行时间内都需要启用耦合。此时自动中断指示将总是指向程序开始处，而 SERUPRO 功能事实上将无作用。

10.8.7 上电、运行方式切换和复位时的特性

SERUPRO 在上电时不生效。SERUPRO 期间允许进行运行方式切换。复位会使 SERUPRO 终止，内部选择的程序测试会被重新取消。SERUPRO 不可与其它程序段搜索类型组合使用。

10.8.8 前提条件

10.8.8.1 目标程序段中的 STOPRE

STOPRE 程序段从前一程序段获取所有跨程序段设置，因此可针对以下情形将原程序段前的条件纳入考量：

- 将当前执行的程序行与主处理同步。
- 为 SERUPRO 退刀跨程序段设置，例如用于在 SERUPRO 定位时调整 REPOS 运行。

示例：通过 X 轴的设定值给定定位 Z 轴。

系统解释程序段“G1 F100 Z=$AA_IM[X]”时，前一 STOPRE 程序段用于实现与主处理的同步。这样便可通过 $AA_IM 读取正确的 X 轴设定值，从而将 Z 轴运行至相同位置。

示例：读取并正确计算外部零点偏移。

```
N10 G1 X1000 F100
N20 G1 X1000 F500
N30 G1 X1000 F1000
N40 G1 X1000 F5000
N50 SUPA G1 F100 X200              ；将外部零点偏移向 200 运行
N60 G0 X1000
N70 ...
```

通过 N50 前的隐性 STOPRE，NC 可读取并正确计算当前零点偏移。
在以 N50 为目标的 SERUPRO 搜索中，系统会在 SERUPRO 定位中重新定位至隐性 STOPRE，并从 N40 通过 F5000 确定速度。

隐性预处理停止

取消隐性预处理停止的情形：

1. 在所有会发生以下变量访问的程序段中：
   - 编写以 $A... 开头的系统变量
   - 编程一个属性为 SYNR / SYNRW 的重定义变量
2. 对于以下指令：
   - 零件程序指令 MEACALC、MEASURE
   - 编程 SUPA（抑制框架和在线补偿）
   - 编程 CTABDEF（开始曲线图表定义）
   - 零件程序指令 WRITE/DELETE（写入/删除文件）
   - 在此类指令序列的第一个 WRITE/DELETE 指令前
   - 零件程序指令 EXTCALL
   - 零件程序指令 GETSELT、GETEXET
   - 在换刀和刀具精补偿 FTOCON 激活时
3. 对于以下指令编辑：
   - 结束程序段搜索，类型 1（“无计算搜索程序段”）
   - 结束程序段搜索，类型 2，带计算（“在轮廓终点上搜索程序段”）

10.8.8.2 目标程序段中的 SPOS

若通过 M3 / M4 编程了主轴，并在目标程序段中切换至 SPOS，那么主轴会在 SERUPRO 进程结束（找到搜索目标）时切换至 SPOS。

DB31, ... DBX84.5 = TRUE（有效主轴运行方式：定位运行）

10.8.8.3 运行到固定点（FXS）

在重新运行至轮廓（REPOS）时，系统会自动重复“运行到固定点”（FXS）功能。此时每根轴都会被包含在内。力矩将采用搜索目标前最后编写的力矩。

系统变量

“运行到固定点”的系统变量在 SERUPRO 中的含义如下：

- $AA_FXS：程序仿真的进度
- $VA_FXS：实际机床状态

在 SERUPRO 功能**外**，这两个系统变量总是位相同的值。

ASUB

可为 SERUPRO 激活一个用户专用 ASUB。

文档

程序段搜索 SERUPRO 的详细信息请见“详细说明 (页 338)”章节。

10.8.8.4 以限制的力矩/力运行（FOC）

在重新运行至轮廓（REPOS）时，系统会自动重复“以限制的力矩/力运行”（FOC）功能。 此时每根轴都会被包含在内。 力矩将采用搜索目标前最后编写的力矩。

系统变量

“以限制的力矩/力运行”的系统变量在 SERUPRO 中的含义如下：

- $AA_FOC：程序仿真的进度
- $VA_FOC：实际机床状态

前提条件

在重新运行至轮廓时，**无法**实现持续变化的力矩特性曲线。

示例

程序将 X 轴从位置 0 运行至 100，并对每 20 个增量中的 10 个增量启用“以限制的力矩/力运行”（FOC）。 此力矩特性曲线通常通过逐段生效的 FOC 生成，并且无法在重新运行至轮廓（REPOS）时实现。 此时会在限制力矩/力或无限制的情况下根据最后的编程将 X 轴从 0 运行至 100。

文档

程序段搜索 SERUPRO 的详细信息请见“详细说明 (页 338)”章节。

10.8.8.5 同步主轴

可对同步主轴进行仿真。

在现有的所有通道中均可通过 SERUPRO 对包含一根引导主轴和任意根跟随主轴的同步主轴运行进行仿真。

同步主轴的更多相关信息请见：

文档：

功能手册之扩展功能分册；同步主轴（S3）

10.8.8.6 耦合和主从运行

设定值耦合和实际值耦合

SERUPRO 是程序测试模式下的程序仿真，可用于对设定值耦合和实际值耦合进行仿真。

用于 EG 仿真的定义

对于 EG 仿真，以下定义适用：

1. 总是通过设定值耦合进行仿真。
2. 若仅个别引导轴，而不是所有引导轴均处于 SERUPRO 下，那么仿真会被终止，并触发报警 16952“ResetClear/NoStart”。在采用跨通道耦合时可能会出现此情形。
3. 对于从 NC 方面看来只有一个编码器且由外部运行的轴无法实现正确的仿真。特别需要注意的是，这些轴不能参与耦合。

⚠ 小心
仿真错误 为了能实现对耦合的正确仿真，必须事先取消耦合。 这可通过机床数据 MD10708 $MA_SERUPRO_MASK 进行。

用于轴耦合的定义

通过 SERUPRO 对轴耦合进行仿真时，系统总是将其认定为设定值耦合。此时会为**所有**轴计算终点，这些点用作 SERUPRO 定位的目标点。“找到搜索目标”的同时耦合即已生效。从当前点到终点的行程在耦合生效的情形下在 SERUPRO 定位中执行。

LEADON

相应的，对于轴向引导值耦合的仿真，以下定义适用：

1. 总是通过设定值耦合进行仿真。
2. SERUPRO 定位在耦合生效且跟随轴叠加运行的情形下进行，用于达到仿真的目标点。

只通过耦合运行的跟随轴不是总能到达目标点。在 SERUPRO 定位中会为跟随轴计算出一个叠加线性运动，以定位至仿真点！

通过 JOG 到达 LEAD 的仿真目标点

在“找到搜索目标”的时间点耦合已生效，特别是对于 JOG 运行。在未到达目标点时，SERUPRO 定位中可通过生效的耦合及叠加运动将跟随轴运行至目标点。

说明

轴耦合重新定位的更多信息参见“重新定位至轮廓（REPOS） (页 563)”章节。

主从耦合

程序段搜索结束后，可自动启动一个系统 ASUB。在该 ASUB 中，用户可对耦合状态及对应的轴位置进行后续调整。为此所需的信息通过以下系统变量获取：

系统变量	说明
$P_SEARCH_MASLD[<从轴>]	耦合关闭时间点从轴和主轴间的位置偏移。
$AA_MASL_STAT[<从轴>]	主从耦合的当前状态
$P_SEARCH_MASLC[<从轴>]	状态：耦合状态已在程序段搜索中变化
在通过 MASLON 激活耦合时，这些系统变量会被清除。	

说明

在程序段搜索的时间点，参与耦合的轴必须处于同一通道中。

文档

主从耦合的更多相关信息请见：

功能手册之特殊功能；章节“TE3：主从转速/转矩耦合，主/从连接”

示例

- **系统 ASUB**
  - 路径和名称：/_N_CMA_DIR/PROGEVENT.SPF
  - 主动轴：X
  - 从动轴：Y

程序代码	
PROG PROGEVENT	
N10 IF(($S_SEARCH_MASLC[Y] <> 0) AND ($AA_MASL_STAT[Y] <> 0))	
N20 MASLOF(Y)	
N30 SUPA Y = $AA_IM[X] - $P_SEARCH_MASLD[Y]	
N40 MASLON(Y)	
N50 ENDIF	
N60 REPOSA	
...	
RET	

- **机床数据**

  为了能自动启动 ASUB，必须设置以下机床数据：
  - NC 专用机床数据：
    - MD11604 $MN_ASUP_START_PRIO_LEVEL = 100
    - MD11450 $MN_SEARCH_RUN_MODE = 'H02'
  - 启动 ASUB 的通道专用，或适用于所有通道：
    - MD20105 $MC_PROG_EVENT_IGN_REFP_LOCK，位 <n> = TRUE
      n:用于全部所需事件控制的程序调用（程序事件）
    - MD20115 $MC_IGNORE_REFP_LOCK_ASUP，位 <n> = TRUE
      n:用于全部**所需用户**中断

注意
系统中断 通过 MD20115 $MC_IGNORE_REFP_LOCK_ASUP，位 8 ~ 31 使能系统中断。 通过位 8 / 中断 9 启动一个包含运行的 ASUB。

轴耦合

- 加快执行速度以及不同通道中的引导轴和跟随轴
  对于跟随轴处于另一个通道的引导轴，用于加快执行速度 (MD22601 (页 573)$MC_SERUPRO_SPEED_FACTOR) 的设置不生效：
- 跟随运行
  SERUPRO 支持跟随运行（TRAILON）功能。
  跟随运行（TRAILON，TRAILOF）的更多详细信息请见：
  文档：
  – 功能手册之特殊功能分册；轴耦合（M3）
  – 编程手册之工作准备分册，章节“轴耦合”
- 龙门轴
  SERUPRO 支持龙门轴功能。
  龙门轴功能的更多相关信息请见：
  文档
  ：功能手册之特殊功能分册，章节“G1：龙门轴”
- 切向控制
  SERUPRO 支持单轴的切向跟踪功能。
  切向控制的更多相关信息请见：
  文档
  ：功能手册之特殊功能分册，章节“T3：切向控制”

10.8.8.7 轴功能

SERUPRO 条件

在轴使能、自控轴运动和跨通道取轴中，必须注意针对 SERUPRO 的特殊条件。

轴使能

若不宜或无法将闭环控制使能发送至机床，且程序测试或 SERUPRO 期间无使能生效，那么轴接口信号 DB31, ... DBX3.7（“程序测试轴/主轴使能”）可用于控制轴使能。

可通过接口信号 PLC → NC
DB31, ... DBX3.7（“程序测试轴/主轴使能”）进行使能。若程序测试或 SERUPRO 中缺少实际的闭环控制使能，则会对轴/主轴产生以下影响：

- 仿真的程序运行需要运行轴/主轴时，会立即显示信息“等待轴使能”或“等待主轴使能”，并停止仿真。
- 若在仿真运行期间重新取消 NC/PLC 接口信号 DB31, ... DBX3.7（程序测试轴/主轴使能），则会触发报警 21612：“通道 %1 轴 %2 NC/PLC 接口信号‘闭环控制使能’在运行期间被取消”。

自控轴运动

自控单轴运动是由 PLC 控制的轴，其也可在 SERUPRO 中进行仿真。为此，SERUPRO 期间 PLC 会像一般运行时一样接收或交还轴的控制权。必要时也可通过 FC18 运行该轴。PLC 在逼近程序段**前**接收轴的控制权，并负责该轴的定位。这适用于所有程序段搜索类型。

自控单轴运动的更多相关信息请见：

文档：
功能手册之扩展功能分册；定位轴（P2）

跨通道取轴

问题：轴通过程序运行，而改程序在写入了 WAITP(X) 的目标程序段前交还控制权。这样一来 X 不归属于 REPOS，SERUPRO 定位中将不包含该轴。

通过机床数据 MD11470 $MN_REPOS_MODE_MASK 可为 SERUPRO-REPOS 设定以下特性：

中性轴作为“指令轴”在 SERUPRO-REPOS 中运行。该轴的插补无轨迹关联，即便其最近一次是被编程作为轨迹轴。在此情形下速度取决于 MD32060 $MA_POS_AX_VELO。SERUPRO 定位后该轴重新恢复为中性轴。

不允许重新定位的中性轴必须设定轴专用 NC/PLC 接口信号“REPOSDELAY”。这样便可删除 REPOS 运行。

示例：

SERUPRO 后通过同步动作中的工艺循环运行一根轴。指令轴总是在逼近程序段中运行，而从不在目标程序段中运行。所有指令轴都切换至终点时，才可切换至目标程序段。

⚠ 小心

PLC 控制的轴不进行重新定位

目标程序段前通过 RELEASE(X) 使能的轴不进行重新定位。

10.8.8.8 齿轮档切换

步骤

齿轮档切换（GSW）从 NC 请求物理运动，用于切换至新的齿轮档。
SERUPRO 进程中不需要齿轮档切换，并如下执行：

某些齿轮箱只能在 NC 的控制下进行切换，因为轴会出现摆动，或者需要在切换前定位至特定位置。

可通过 MD35035 $MA_SPIND_FUNCTION_MASK 的位 0 至位 2 选择为 DryRun、程序测试 和 SERUPRO 抑制齿轮档切换。

之后须在 REPOS 中追加执行 GSW；即使对应轴需要在目标程序段处于“转速控制运行”，追加的 GWS 同样生效。否则，若零件程序中 GWS 和目标程序段之间轴参与了转换或耦合，那么自动 GWS 将会被系统拒绝并触发报警。

说明

有关 DryRun、程序测试和 SERUPRO 时的齿轮档切换的更多信息请见“S1：主轴 (页 1401)”章节。

10.8.8.9 叠加运动

仅可使用 SERUPRO

若使用“叠加运动”，则只可采用程序测试下的程序段搜索（SERUPRO），因为此时叠加运动会在主处理中进行相应插补。 这尤其适用于 $AA_OFF。

速度特性图取代最大轴速度

在程序测试中必须使用速度特性图，其支持在主处理中插补“叠加运动”。 这样一来便无法通过最大轴速度进行插补。

轴速度在“空运行进给”模式下通过
SD42100 $SC_DRY_RUN_FEED 设置。

SERUPRO 进程的速度通过
MD22600 $MC_SERUPRO_SPEED_MODE 选择。

10.8.8.10 NC/PLC 接口信号

REPOS 偏移存在

若 SERUPRO 期间轴存在 REPOS 偏移，则会在 SERUPRO 进程结束时通过轴专用 NC/PLC 接口信号显示：

DB31, ...DBX70.0 == 1（REPOS 偏移存在）

REPOS 偏移的生效范围

SERUPRO ASUB 启动，或通过 NC 启动 继续执行时，REPOS 偏移失效：

DB31, ... DBX70.1 == 1（REPOS 偏移失效）

在 SERUPRO 进程结束和通过 NC 启动 继续执行之间，可在 JOG 运行方式下手动运行轴，或借助 PLC 用户程序通过 FC 18 运行轴。 REPOS 偏移完全走完时，接口信号复位。

10.8.8.11 缺省设置的灵活设定

缺省设置 / SERUPRO 缺省设置

通过机床数据 MD20112 $MC_START_MODE_MASK 定义零件程序开始时控制系统在 G 代码（特别是当前层级和可设定的零点偏移）、刀具长度补偿、转换和轴耦合方面的缺省设置。特别是对于 SERUPRO 进程而言，还可通过 MD22620 $MC_ENABLE_START_MODE_MASK_PRT 选择与一般零件程序启动有别的缺省设置。其中新的设置必须保存在以下机床数据中：

MD22620 $MC_START_MODE_MASK_PRT

MD22620 各位的含义与 MD20112 $MC_START_MODE_MASK 相同。

示例：

零件程序启动时保留 SERUPRO 开始时的同步主轴耦合。

```
                                                    ; 未配置的同步主轴耦合
$MC_START_MODE_MASK = 'H400'                        ; 被撤销
$MC_START_MODE_MASK_PRT = 'H00'                     ; 保持生效
```

```
$MC_ENABLE_START_MODE_MASK_PRT = 'H01'                    ; SERUPRO 中会分析
                                                          $MC_START_MODE_MASK_PRT，而不
                                                          是 $MC_START_MODE_MASK
```

10.8.8.12 压缩器功能（COMPON、COMPCURV、COMPCAD）

- 在程序段搜索**类型 2** 或**类型 4**（在 ... 处**计算**的程序段搜索）中，若目标程序段所处于的程序部分中激活了压缩器功能（COMPON、COMPCURV、COMPCAD），那么在重定位至轮廓时会逼近通过**压缩器**计算出的轨迹上的位置。 这些位置不必与零件程序中编写的轨迹上的位置精确一致。
- 若压缩时省去了零件程序中编写的程序段，那么在程序段搜索中将无法将这些程序段作为目标程序段找到 ⇒ 报警 15370“未找到搜索目标”。

10.8.9 系统变量

SERUPRO 的相关系统变量概览：

系统变量	含义
$AC_ASUP，位 20	ASUB 激活原因： $AC_ASUB, 位 20 == 1 ⇒ 系统 ASUB 生效，原因：到达 SERUPRO 搜索目标
$AC_SERUPRO	SERUPRO 状态： $AC_SERUPRO == 1 ⇒ SERUPRO 生效
$P_ISTEST	程序测试状态： SERUPRO 生效 ⇒ $P_ISTEST == 1
$P_SEARCHL	最后生效的程序段搜索类型： $P_SEARCHL == 5 从 SERUPRO 开始至复位或程序结束
$AC_REPOS_PATH_MODE	REPOS 模式按照 SERUPRO 用于轮廓重新定位

10.9 程序运行

定义

程序运行是指在 AUTOMATIK 或 MDA 模式下执行 NC 程序或 NC 程序段。

NC/PLC 接口信号

程序运行可由 PLC 用户程序通过 BAG 和通道专用的 NC/PLC 接口信号进行控制或将相应的反馈信息反馈至 PLC 用户程序。

NC/PLC 接口信号功能一览请见：

文档

- 功能手册之基本功能分册，章节“Z1：NC/PLC 接口信号”
- 功能手册之扩展功能分册，章节“Z2：NC/PLC 接口信号”
- 功能手册之特殊功能分册，章节“Z3：NC/PLC 接口信号”

NC/PLC 接口信号一览请见：

文档

参数手册“NC 变量和接口信号”；章节“接口信号一览”

10.9.1 缺省设置

通过通道专用机床数据可为每个通道设定初始设置。这些初始设置例如会影响 G 功能组合辅助功能输出。

辅助功能输出

可通过机床数据 AUXFU_x_SYNC_TYPE（MD22200、22210、22220、22230、22240、22250、22260；M、S、T、H、F、D、E 功能的输出时间点）预定义辅助功能的时序输出。更详细的说明请参见“H2：输出到 PLC 的辅助功能 (页 429)”章节。

G 功能组

对存在的每个 G 功能组均可通过 MD20150 $MC_GCODE_RESET_VALUES（G 组的缺省设置）设定一个编程缺省设置。在程序启动或复位状态中该编程缺省设置自动生效，直至其由于同 G 组的 G 指令被取消。

通过 MD22510 $MC_GCODE_GROUPS_TO_PLC（程序段切换/RESET 时向 NC-PLC 接口输出的 G 指令）可激活 G 指令向 PLC 接口的输出。

文档

G 指令组和对应 G 指令的列表请见：

编程手册之基本原理

SINUMERIK solution line 的 NC 语言范围缺省配置

对于 SINUMERIK 840D sl，可通过机床数据生成特定 NC 语言范围缺省配置，并对其进行设置。这样一来，用户可根据自身所需的选件和功能对 NC 语言范围进行定制。

10.9.1.1 机床数据

NC 语言范围

通过语言指令运行未生效选件和功能的方式可通过以下机床数据设置：

MD10711 $MN_NC_LANGUAGE_CONFIGURATION = <值>

值	含义
0	所有语言指令都可识别。执行时才识别所需功能是否激活。
1	所有语言指令都可识别。 未使能选件的语言指令在程序解释时会被识别 ⇒ 触发报警 12553 “选件/功能未生效”。
2	仅可识别已使能选件的语言指令。 未使能选件的语言指令在程序解释时会被识别 ⇒ 触发报警 12550 “名称未定义，或选件/功能不存在”。
3	所有语言指令都可识别。 未激活功能的语言指令在程序解释时会被识别 ⇒ 触发报警 12553 “选件/功能未生效”。 **示例** 1. 柱面转换选件已设置 2. 柱面转换在机床数据 MD24100 $MC_TRAOF_TYPE_1 中**未**激活 3. 编程指令 TRACYL 会触发报警 12553。
4	仅可识别已激活功能的语言指令。 未激活功能的语言指令不会被识别 ⇒ 触发报警 12550 “名称未定义，或选件/功能不存在”。 **提示** 在此情形下无法分辨相关语言指令是不属于西门子语言范围，还是仅在相应设备上不存在。

10.9.1.2 编程

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。

可检查以下 NC 编程语言元素：

- 现有所有 G 功能组的 G 指令
- DIN 或 NC 地址
- 功能
- 步骤
- 关键字
- 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

定义

```
INT STRINGIS(STRING <名称>)
```

句法

```
<返回值> = STRINGIS(<名称>)
```

含义

STRINGIS()：	带返回值的校验函数
<名称>：	需要检查的字符串

<返回值>：	返回值在前三个小数位中进行编码 yxx	
	000	给定的字符串不是当前语言集的元素 1)
	100	给定的字符串是当前语言集的元素，但当前不可编程（选件/功能无效）
	2xx	给定的字符串是当前语言集的一个可编程元素（选项/功能生效）。 详细信息请见第 1 和第 2 个小数位 xx：
		xx **含义**
		01 DIN 地址或 NC 地址 2)
		02 G 指令（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 中装载，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 指令（ISO 语言模式必须生效）
	400	给定的字符串是一个既未识别为 DIN 地址或 NC 地址 (xx==01)，又未识别为宏名称 (xx==10) 且不是 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，也被识别为宏

示例

在下面的示例中假设给定的 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”定义为命名的 G 指令：
   ```
   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”定义为宏：
```
210 == STRINGIS("LONGMACRO")
```

13.字符串“MYVAR”定义为 LUD 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")
```

边界条件

刀具刀库管理

如果刀具刀库管理功能未生效，则用于刀具刀库管理系统参数的 STRINGIS() 总是独立于以下机床数据

- MD10711 $MN_NC_LANGUAGE_CONFIGURATION

输出值 000。

ISO 模式

“ISO 模式”功能生效：

如果“ISO 模式”功能生效，STRINGIS() 会先将给定的字符串作为 SINUMERIK G 指令检查：

- 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）

“ISO 模式”生效时，STRINGIS() 会先检查给定的字符串与 SINUMERIK G 指令的归属关系。

如果该字符串不是 SINUMERIK G 指令，则随后检查其与 ISO G 指令的归属关系。

编程的切换 ISO 模式

编程的通过指令 G290 （SINUMERIK 模式）和 G291 （ISO 模式）进行的切换对 STRINGIS() 没有影响。

10.9.2 选择并启动 NC 程序

NC/PLC 接口信号

选择

仅当相关通道处于复位状态时，才可选择 NC 程序。

- DB21, ... DBX35.7 == 1（复位）

启动

NC 程序是通过两个不同的事件启动的：

1. DB21, ... DBX7.1 = 1（NC 启动），该信号通常是通过按下机床控制面板上的“NC 启动”键触发的。
2. 另一个生效通道的 NC 程序中的指令 START。该通道必须处于 AUTOMATIK 或 MDA 模式下以及“复位”或“中断”状态下。
   - DB21, ... DBX35.7 == 1（复位）
   - DB21, ... DBX35.6 == 1（中断）

输出条件

仅当满足以下输出条件时，才能启动 NC 程序。

- DB11 DBX4.4 == 1（BAG 就绪）
- DB11 DBX0.7 == 0（BAG 复位）
- DB21, ... DBX1.7 == 0（激活程序测试）
- DB21, ... DBX7.0 == 0（NC 启动禁止）
- DB21, ... DBX7.2 == 0（程序段交界处 NC 停止）
- DB21, ... DBX7.3 == 0（NC 停止）
- DB21, ... DBX7.4 == 0（进给轴和主轴 NC 停止）
- DB21, ...:DBX7.7 == 0（复位）
- DB10 DBX56.1 == 0（急停）
- 不可以存在轴或 NC 专用报警

执行指令

零件程序或零件程序段会自动执行，以下接口信号会被置位：

- DB21, ... DBX35.5（通道状态：生效）
- DB21, ... DBX35.0（程序状态：运行）

程序一直运行下去，直至到达程序结束，或者通道由于 STOP/RESET 指令中断或终止。

文档

对接口信号的详细说明请参见参数手册，NC 变量和接口信号。

报警

在特定情形下，START 指令不生效，并会触发下列报警：

- 10200“报警生效时不允许进行 NC 启动”
- 10202“指令生效时不允许进行 NC 启动”
- 10203“对未回参考点的轴不允许进行 NC 启动”

文档：

诊断手册 报警

10.9.3 程序中断

NC/PLC 接口信号

前提条件

仅当通道和 NC 程序激活时，才能执行程序中断：

- DB21, ... D35.5 == 1（通道生效）
- DB21, ... D35.0 == 1（程序运行中）

程序中断

通过以下事件可以中断程序执行：

- DB21, ... DBX7.2 == 1（程序段交界处 NC 停止）
- DB21, ... DBX7.3 == 1（NC 停止）
- DB21, ... DBX7.4 == 1（进给轴和主轴 NC 停止）
- DB21, ... DBX2.0 == 1（单程序段）
- 已执行 NC 程序中的编程指令 `M00` 或 `M01`

之后通道和 NC 程序处于“中断”状态：

- DB21, ... D35.6 == 1（通道中断）
- DB21, ... D35.3 == 1（程序中断）

文档

对接口信号的详细说明请参见参数手册，NC 变量和接口信号。

过程

识别出程序中断后会执行以下操作：

- 中断程序执行：
  - 在以下事件中会停止在下一个程序段边界上：“程序段交界处NC停止”、M00、M01或单程序段
  - 立即：所有其他时间
- 运行的通道轴通过制动斜坡停止。轴制动至静止状态可能需要持续多个程序段。
- 程序段指针显示中断位置的当前程序段。
- 到中断位置为止尚未输出的辅助功能将不再输出。

中断状态下可进行的操作

零件程序中断期间，可在通道中执行多种功能，例如：

- **覆盖**
  文档
  SINUMERIK Operate 操作手册，章节“加工工件” > “覆盖”
- **程序段搜索**
  文档
  功能手册之基本功能分册，“BAG、通道、程序运行、复位特性（K1）”章节 > “程序段搜索”或“程序段搜索类型 5 SERUPRO”
- **重新定位至轮廓（REPOS）**
  文档
  功能手册之基本功能分册，“BAG、通道、程序运行、复位特性（K1）”章节 > “程序段搜索类型 5 SERUPRO” > “REPOS” > “通过受控 REPOS 重新定位至轮廓”
- **定向退刀**
  文档
  - 编程手册之工作准备分册，“刀具补偿”章节
  - 功能手册之基本功能分册，“刀具补偿（W1）”章节 >“可定向刀架”
- **ASUB**（参见章节“异步子程序（ASUB） (页 635)”）

- **DRF 功能，工件零点平移**
  文档
  功能手册之扩展功能分册；手动运行和通过手轮进行手动运行（H1）
- **继续执行中断的 NC 程序**
  - 通过另一个通道中的 START 指令
    文档
    编程手册之工作准备分册，“灵活 NC 编程”章节 > “程序协调（INIT、START、WAITM、WAITMC、WAITE、SETM、CLEARM）”
  - NC/PLC 接口信号：
    DB21, ... DBX7.1 = 1（NC 启动）

10.9.4 通道复位

功能

在 AUTO 模式下，通过通道复位可以中断一个正在执行的 NC 程序；在 MDA 模式下，通过通道复位可以中断一个正在执行的程序段。

NC 程序或程序段无法再从中断位置继续执行。通道复位结束后，通道的所有进给轴和主轴都处于“准停”状态（例外：跟踪运行）。

过程

通过 NC 复位可在通道中执行以下操作：

- 立即停止程序准备。
- 进给轴和主轴通过其设置的制动斜坡减速至静止状态。
- 当前程序段尚未输出的辅助功能将不再输出到 PLC。
- 程序段指针返回到所选 NC 程序的起始处。
- 显示器中所有通道复位报警清除。

规则

- 通道复位可在各通道状态下执行。
- 通道复位不会由其他指令终止。

NC/PLC 接口信号

请求：通道复位

通过以下 NC/PLC 接口信号请求通道复位：

- DB21, ... DBX7.7 = 1（复位）

请求：BAG 复位

BAG 复位在所有 BAG 通道中都会触发通道复位。

通过以下 NC/PLC 接口信号请求 BAG 复位：

- DB11, ... DBX0.7 = 1（BAG 复位）

反馈信息：通道复位结束

- DB21, ... DBX35.7 == 1（通道状态复位）

反馈信息：BAG 复位结束

- DB11, ... DBX6.5 == 1（BAG 已复位）

文档

对接口信号的详细说明请参见参数手册，NC 变量和接口信号。

10.9.5 程序状态

对于每个通道，所选 NC 程序的状态在接口中显示。

在 AUTOMATIC 和 MDA 运行方式下可以显示所有程序状态。在所有其他运行方式下或机床功能中，程序状态为终止或中断。

NC/PLC 接口信号

在 NC/PLC 接口（DB21, ...）上会显示以下程序状态：

- DB21, ... DBX35.4（“终止”）
- DB21, ... DBX35.3（“中断”）
- DB21, ... DBX35.2（“停止”）
- DB21, ... DBX35.1（“等待”）
- DB21, ... DBX35.0（“运行”）

对接口信号的详细说明请参见参数手册，NC 变量和接口信号。

状态变化

程序状态受指令、NC/PLC 接口信号和报警影响。从程序状态“正在运行”出发，表格中显示了各个事件下的后续状态。

程序的输出状态：“正在运行”					
事件	程序的后续状态				
	终止	中断	停止	等待	运行
DB21, ... DBX7.7（复位）	x				
DB21, ... DBX7.3（NC 停止）			x		
DB21, ... DBX7.2（程序段交界处 NC 停止）			x		
DB21, ... DBX7.4（进给轴和主轴 NC 停止）			x		
DB21, ... DBX6.1（读取禁止）					x
DB21, ... DBX6.0（进给停止）					x
DB21, ... DBX12.3 / 16.3 / 20.3（进给停止，几何轴 1 / 2 / 3）					x
DB21, ... DBB4；进给倍率 = 0%					x
DB31, ... DBX4.3（进给停止/主轴停止）					x
DB21, ... DBX194.2 / DBX197.6（程序段中的 M02 / M30）	x				
DB21, ... DBX194.0 / DBX194.1（程序段中的 M00 / M01）			x		
DB21, ... DBX0.4（单程序段）			x		
DB21, ... DBX6.2（删除剩余行程）					x
向 PLC 输出辅助功能，但尚未应答			x		
程序中的 WAIT 指令				x	
报警及系统响应“NOREADY”		x			

10.9.6 通道状态

在所有运行方式下，每个通道的通道状态均在 NC/PLC 接口（DB21, ...）上显示。

NC/PLC 接口信号

在 NC/PLC 接口上通过以下信号显示通道状态：

- DB21, ... DBX35.7（“复位”）
- DB21, ... DBX35.6（“中断”）
- DB21, ... DBX35.5（“生效”）

对接口信号的详细说明请参见参数手册，NC 变量和接口信号。

状态变化

通道状态受指令和 NC/PLC 接口信号影响。从通道状态“生效”出发，表格中显示了各个事件下的后续状态。

通道的输出状态：“生效”			
事件	通道的后续状态		
	“复位”	“中断”	“生效”
DB21, ... DBX7.7（复位）	x		
DB21, ... DBX7.3（NC 停止）		x	
DB21, ... DBX7.2（程序段交界处 NC 停止）		x	
DB21, ... DBX7.4（进给轴和主轴 NC 停止）		x	
DB21, ... DBX6.1（读取禁止）			x
DB21, ... DBX6.0（进给停止）			x
DB21, ... DBX12/16/20.3（进给停止/几何轴 1/2/3）			x
DB21, ... DBB4；进给倍率 = 0%			x
DB31, ... DBX4.3（进给停止/主轴停止）			x
DB21, ... DBX194.2/DBX197.6（程序段中的 M02/M30）	x		
DB21, ... DBX194.0/DBX194.1（程序段中的 M00/M01）		x	
DB21, ... DBX0.4（单程序段）		x	
DB21, ... DBX6.2（删除剩余行程）			x
向 PLC 输出辅助功能，但尚未应答			x
程序中的 WAIT 指令			x

执行 NC 程序或 NC 程序段，或者在 JOG 运行方式下运行轴时为“通道状态：生效”。

10.9.7 对操作和程序动作的响应

状态过渡

下表列出了在某些操作或程序动作后所产生的通道和程序的状态。

表格左侧列出了通道和在通道中选择的程序的不同状态以及生效的运行方式。

表格右侧列出了操作/程序动作和后续状态。

状态	通道状态			程序状态					运行方式			操作或程序动作 => 后续状态
	R	U	A	R	U	S	W	A	A	M	J	
1		x						x	x			复位 ⇒ 4
2		x						x		x		复位 ⇒ 5
3		x						x			x	复位 ⇒ 6
4	x			x					x			NC 启动 ⇒ 13；运行方式切换 ⇒ 5 或 6
5	x			x						x		NC 启动 ⇒ 14；运行方式切换 ⇒ 4 或 6
6	x			x							x	NC 启动 ⇒ 15；运行方式切换 ⇒ 4 或 5
7		x		x						x		NC 启动 ⇒ 14
8		x		x							x	NC 启动 ⇒ 15
9		x			x				x			NC 启动 ⇒ 13；运行方式切换 ⇒ 10 或 11
10		x			x					x		NC 启动 ⇒ 16；运行方式切换 ⇒ 9 或 11
11		x			x						x	方向键 ⇒ 17；运行方式切换 ⇒ 9 或 10
12		x				x			x			NC 启动 ⇒ 13；运行方式切换 ⇒ 10 或 11
13			x					x	x			NC 停止 ⇒ 12
14			x	x						x		NC 停止 ⇒ 7；在程序段末尾 ⇒ 5
15			x	x							x	NC 停止 ⇒ 8；在 JOG 末尾 ⇒ 6
16			x		x					x		NC 停止 ⇒ 10；在程序段末尾 ⇒ 10
17			x		x						x	NC 停止 ⇒ 11；在 JOG 末尾 ⇒ 11
18			x				x		x			复位 ⇒ 4；等待其他通道 ⇒ 18

通道状态

R → 终止

U → 中断

A → 正在运行

程序状态

R → 终止

U → 中断

S → 停止

运行方式

A → AUTO

M → MDA

J → JOG

W → 等待

A → 正在运行

10.9.8 程序运行的时间图示例

程序代码

```
N10 G01 G90 X100 M3 S1000 F1000 M170
N20 M0
```

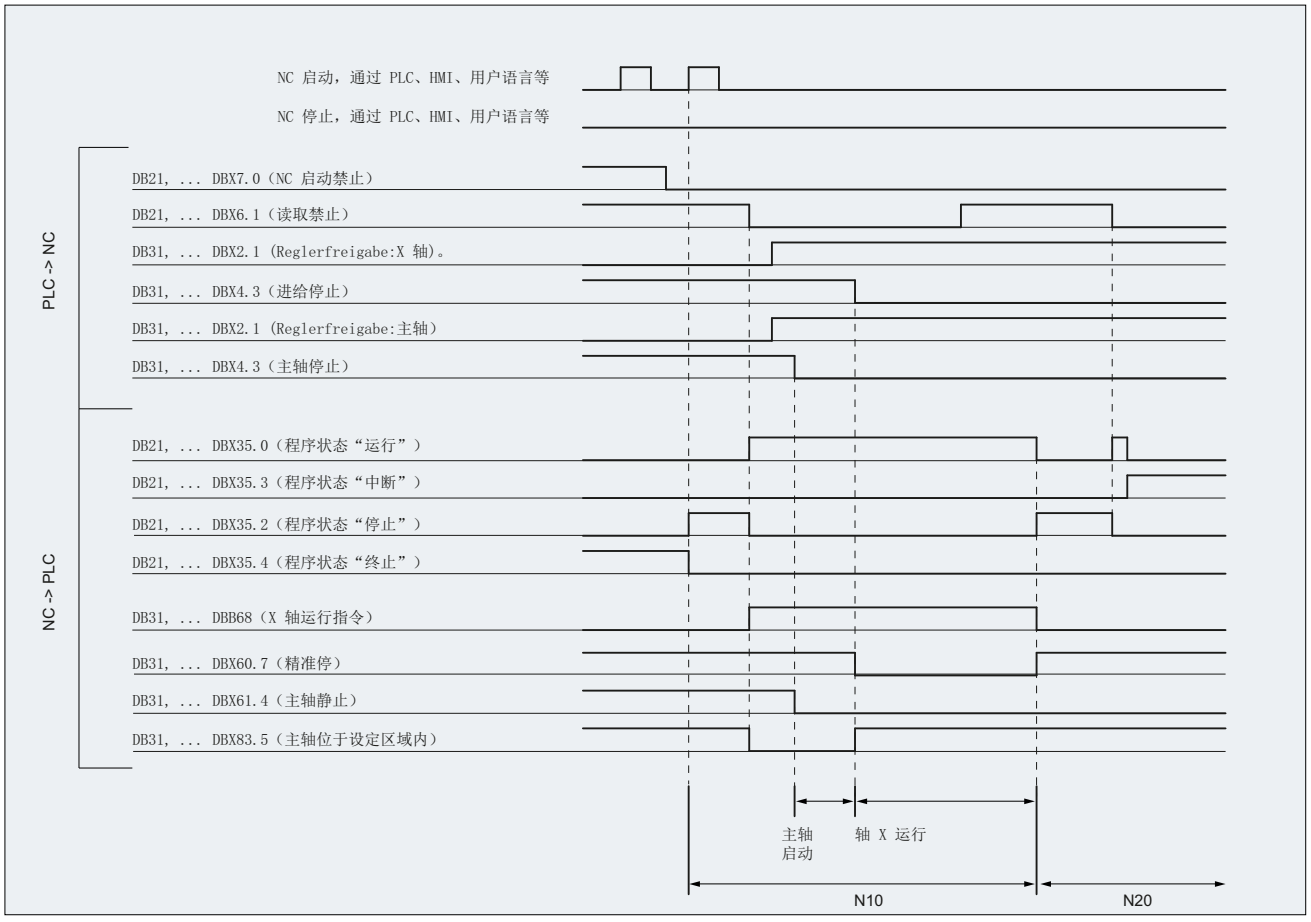


图 10-6 程序期间的信号特性曲线

10.9.9 程序跳转

10.9.9.1 跳回至程序开始处（GOTOS）

功能

通过“跳回至程序开始处”功能可从零件程序跳转回程序开始处。之后该程序会被重新执行。

与同样可用于实现零件程序的重新执行的“程序跳转至跳转标记”功能相比，“跳回至程序开始处”功能具有以下优势：

- 不需要在程序开始处编写跳转标记。
- 程序重启可通过 NC/PLC 接口信号控制：
  DB21, ... DBX384.0（控制程序跳转）
- 可在程序重启时将程序运行时间的定时器复位为“0”。
- 可在程序重启时将工件计数的定时器增加“1”。

应用示例

此功能适用于以下情形：需要通过自动程序重启加工后面的工件，例如对于配备棒式进料器/换料器的车床。

NC/PLC 接口信号

仅当以下 NC/PLC 接口信号置位时，才进行跳回：

DB21, ... DBX384.0（控制程序跳转）= 1

参数设置

程序运行时间

所选 NC 程序的运行时间保存在系统变量 $AC_CYCLE_TIME 中。启动新的程序时，该系统变量会自动复位为“0”（参见“程序运行时间 (页 740)”章节）

通过设置以下机床数据，可在通过“跳回至程序开始处”功能重启程序时也将系统变量 $AC_CYCLE_TIME 复位为“0”：

MD27860 $MC_PROCESSTIMER_MODE，位 8 = <值>（激活程序运行时间测量）

位	值	含义
8	0	“跳回至程序开始处”功能**不会**使 $AC_CYCLE_TIME 复位为“0”。
	1	“跳回至程序开始处”功能会使 $AC_CYCLE_TIME 复位为“0”。

说明

为了使位 8 的设置生效，当前程序运行时间的测量必须生效（MD27860 位 1 = 1）。

工件计数

在到达零件程序结束（M02 / M30）后，激活的工件计数器（$AC_TOTAL_PARTS / $AC_ACTUAL_PARTS / $AC_SPECIAL_PARTS）的数值加“1”（参见“工件计数器 (页 752)”章节）。

通过设置以下机床数据，可在通过“跳回至程序开始处”功能重启程序时也将激活的工件计数器向上计数：

MD27880 $MC_PART_COUNTER，位 <n> = <值>（激活工件计数器）

位	值	含义：通过“跳回至程序开始处”功能重启程序时，工件计数器：
7	0	$AC_**TOTAL**_PARTS **未**向上计数。
	1	$AC_**TOTAL**_PARTS 向上计数。
11	0	$AC_**ACTUAL**_PARTS **未**向上计数。
	1	$AC_**ACTUAL**_PARTS 向上计数。
15	0	$AC_**SPECIAL**_PARTS **未**向上计数。
	1	$AC_**SPECIAL**_PARTS 向上计数。

编程

句法

GOTOS

含义

GOTOS：	跳回到当前程序段的开头	
	预处理停止：	是
	生效方式：	逐段式

示例

编程	注释
N10 ...	；程序开始
...	
IF ...	
N100 GOTOS	跳回到程序开头 (N10)
ENDIF	
...	
RET	

10.9.10 程序部分重复

10.9.10.1 编程

程序部分重复是指在一个 NC 程序中，可以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. REPEATB：重复单个程序行

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

2. REPEAT + 跳转标记：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. REPEAT + 跳转标记_1 + 跳转标记_2：重复两个跳转标记间的段落

```
<开始-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. REPEAT + 跳转标记 + ENDLABEL：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： • REPEATB：需要重复的程序行 • REPEAT：需要重复的程序段落的开始
	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 **提示：** 如果没有指定 P=<n>，则程序部分仅重复一次。
<n>:	重复次数
	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 指令之后的程序行。

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 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	；程序段开头
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	；程序段开头
N20 Z=10-R10	
N30 G1 X=R10 F200	
N40 Y=R10	
N50 X=-R10	
N60 Y=-R10	
N70 END:Z=10	；程序段结尾
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:	；程序段 1 的开头
N30 X10	
N40 Y10	
N50 BEGIN2:	；程序段 2 的开头
N60 X20	
N70 Y30	
N80 ENDLABEL:Z10	
N90 X0 Y0 Z0	
N100 Z-10	
N110 BEGIN3:X20	；程序段 3 的开头

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:	；程序段 1 的开头；钻孔位置 1
N30 X1 Y1	
N40 X2	
N50 Y2	
N60 X3 Y3	
N70 ENDLABEL:	
N80 POS_2:	；程序段 2 的开头；钻孔位置 2
N90 X10 Y5	
N100 X9 Y-5	
N110 X3 Y3	
N120 ENDLABEL:	；程序段 1 和 2 的结尾
N130 BOHRER()	；钻削循环
N140 GEWINDE(6)	；螺纹循环
N150 REPEAT POS_1	；重复程序部分一次，自 POS_1 到 ENDLABEL。
N160 BOHRER()	；钻削循环
N170 GEWINDE(8)	；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。

10.9.11 事件控制的程序调用(PROG_EVENT)

10.9.11.1 功能

功能

在出现一个所选事件时，“事件控制的程序调用” (PROG_EVENT) 功能会在 NC 中启动一个 NC 程序（PROG_EVENT 程序）。

应用示例

系统或用户变量的功能或初始化设置

事件

触发事件通过机床数据 MD20108 $MC_PROG_EVENT_MASK 选择（参见章节“参数设置 (页 621)”）。

PROG_EVENT 程序

通过机床数据 MD11620 $MN_PROG_EVENT_NAME 设置 PROG_EVENT 程序的名称（参见章节“参数设置 (页 621)”）。

PROG_EVENT 程序在发生对应事件的通道中执行。

PROG_EVENT 程序以最低的优先级执行，因此可由用户 ASUB 中断。

执行顺序

通过事件激活时的执行顺序：程序开始

初始状态：

通道：	复位状态
运行方式：	AUTOMATIK（可选：覆盖激活）
	MDI
	TEACH IN

1. 在通道中执行程序开始
2. 初始化步骤，包含对以下机床数据的检测：
   – MD20112 $MC_START_MODE_MASK
3. 作为子程序隐含调用 PROG_EVENT 程序
4. 处理主程序的数据部分
5. 处理主程序的程序部分

通过事件激活时的执行顺序：程序结束

初始状态

通道： 激活

运行方式： AUTOMATIK（可选：覆盖激活）

MDI

TEACH IN

1. 通道中已执行的程序中将换为程序结束程序段。
2. 执行程序结束，检测以下机床数据：
   - MD20110 $MC_RESET_MODE_MASK
   - MD20150 $MC_GCODE_RESET_VALUES
   - MD20152 $MC_GCODE_RESET_MODE
3. 作为 ASUB 隐含调用 PROG_EVENT 程序
4. 在通道中执行复位，检测以下机床数据：
   - MD20110 $MC_RESET_MODE_MASK
   - MD20150 $MC_GCODE_RESET_VALUES
   - MD20152 $MC_GCODE_RESET_MODE

通过事件激活时的顺序：通道复位

初始状态：

通道： 任意

运行方式： 任意

1. 控制系统激活复位步骤，包含对以下机床数据的检测：
   - MD20110 $MC_RESET_MODE_MASK
   - MD20150 $MC_GCODE_RESET_VALUES
   - MD20152 $MC_GCODE_RESET_MODE
2. 作为 ASUB 隐含调用 PROG_EVENT 程序
3. 控制系统激活复位步骤，包含对以下机床数据的检测：
   - MD20110 $MC_RESET_MODE_MASK
   - MD20150 $MC_GCODE_RESET_VALUES
   - MD20152 $MC_GCODE_RESET_MODE

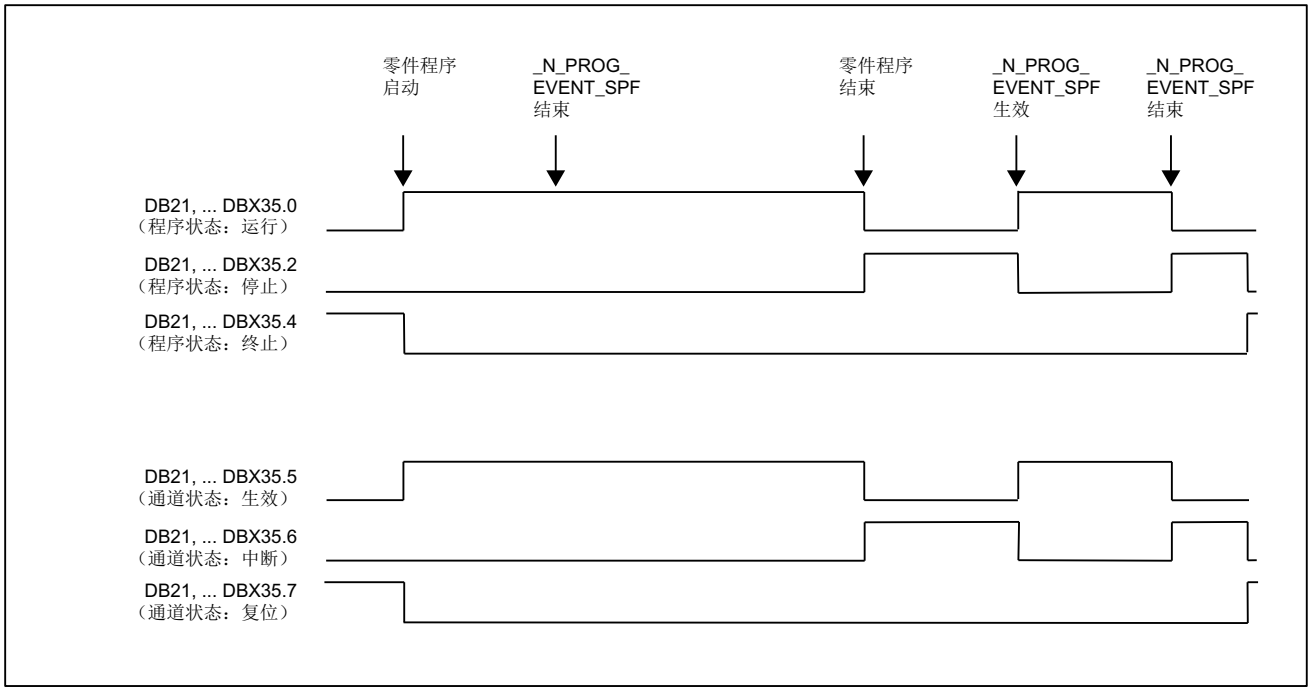
通过事件激活时的顺序：NC 启动

1. 控制系统在启动后激活复位步骤，包含对以下机床数据的检测：
   - MD20110 $MC_RESET_MODE_MASK
   - MD20150 $MC_GCODE_RESET_VALUES
   - MD20152 $MC_GCODE_RESET_MODE
2. 作为 ASUB 隐含调用 PROG_EVENT 程序
3. 控制系统激活复位步骤，包含对以下机床数据的检测：
   - MD20110 $MC_RESET_MODE_MASK
   - MD20150 $MC_GCODE_RESET_VALUES
   - MD20152 $MC_GCODE_RESET_MODE

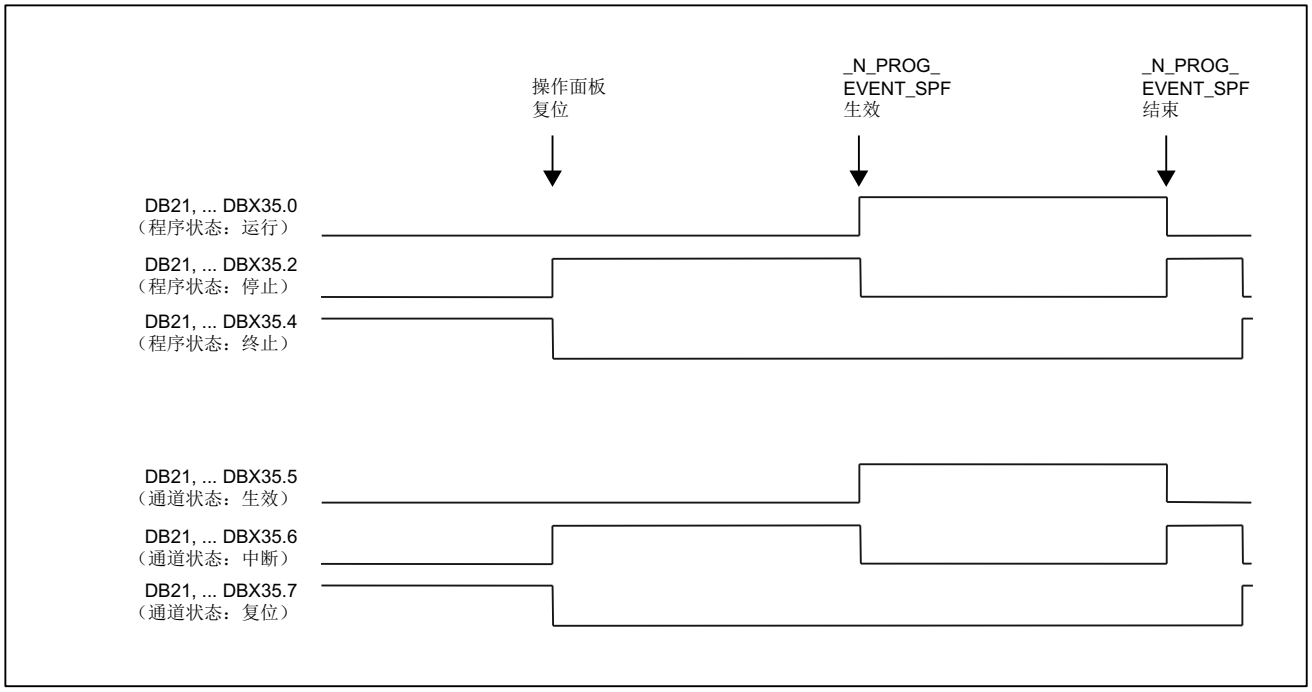
NC/PLC 接口信号：“程序状态”和“通道状态”的变化过程

下图展示了在事件控制的程序调用期间 NC/PLC 接口信号“程序状态”和“通道状态”的变化过程：

由程序开始和程序结束激活的信号变化图



由通道复位激活的信号变化图



NC/PLC 接口信号：DB21, ... DBX35.4（程序状态终止）和 DB21, ... DBX35.7（通道状态复位）

- 当激活的由 PROG_EVENT 程序再次结束时，接口信号才被置位。
- 在以下事件之间时接口信号不会被置位：
  - 程序结束和 ROG_EVENT 程序开始之间
  - 通道复位和 ROG_EVENT 程序开始之间

NC/PLC 接口信号：DB21, ... DBX376.0 ... 4（触发事件）

触发事件通过接口信号 DB21, ... DBX376.0 ... 4 显示：

位	值	事件
0	1	程序从“复位”通道状态开始
1	1	程序结束
2	1	通道复位
3	1	NC 启动
4	1	程序段搜索后的第 1 次程序启动（参见“程序段搜索后 ASUB 的自动启动 (页 552)”）

如果 PROG_EVENT 程序结束或被通道复位终止，接口信号会再次被清除。

接口信号至少应在 PLC 循环期间存在。

10.9.11.2 参数设置

事件选择

通过设置针对具体通道的以下机床数据，可定义用于激活 PROG_EVENT 程序的事件：

MD20108 $MC_PROG_EVENT_MASK，位 <n> = 1

位	<值>	含义：事件
0	1	零件程序开始
1	1	零件程序结束
2	1	通道复位
3	1	启动
5	1	启动中的安全程序

说明

MD20108 $MC_PROG_EVENT_MASK 不在仿真中进行检测。

PROG_EVENT 程序

PROG_EVENT 程序（预设：_N_PROG_EVENT_SPF）必须已载入并使能。

缺省设置

事件发生时默认会执行用户程序_N_CMA_DIR/_N_PROG_EVENT_SPF。

PROG_EVENT 程序必须已载入并使能。

用户专用设置

如要在事件发生时执行另一个 PROG_EVENT 程序，则应在以下机床数据中输入 NC 程序的名称：

MD11620 $MN_PROG_EVENT_NAME = <程序名称>

查找路径

PROG_EVENT 程序必须位于以下其中一个循环目录中。设置的事件出现时，系统会按以下路径搜索：

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)

最先找到的带有在 MD11620 $MN_PROG_EVENT_NAME 中指定的名称的程序将被执行。

说明

- 系统会像对子程序名称一样对设定的名称进行句法检查，即其前两个字符必须为字母或下划线（非数字）。若未为程序名称设定前缀（_N_）和后缀（_SPF），则会自动添加。
- 和适用于循环的情况类似，保护机制对该程序生效（读写的保护级等）。

启动用户 ASUB 时的特性

通过以下机床数据，可针对通道对从通道复位状态启动用户 ASUB 时“事件控制的程序调用”的特性进行设置：

MD20109 $MC_PROG_EVENT_MASK_PROPERTIES，位 0 = <值>

位	值	含义
0	0	出现通过 MD20108 设置的事件时，系统会启动 PROG_EVENT 程序。
	1	出现通过 MD20108 设置的事件时，系统不会启动 PROG_EVENT 程序。

启用单程序段执行时的特性

通过以下机床数据，可针对通道为每个触发事件设置启用单程序段执行时“事件控制的程序调用”功能的特性：

MD20106 $MC_PROG_EVENT_IGN_SINGLEBLOCK，位 <n> = <值>

位	值	含义：在 PROG_EVENT 程序中，以下事件下系统会执行单程序段：
0	0	零件程序开始：**生效**
	1	零件程序开始：**抑制**
1	0	零件程序结束：**生效**
	1	零件程序结束：**抑制**

位	值	含义：在 PROG_EVENT 程序中，以下事件下系统会执行单程序段：
2	0	通道复位：**生效**
	1	通道复位：**抑制**
3	0	启动：**生效**
	1	启动：**抑制**

单程序段模式被抑制时，系统会继续执行 PROG_EVENT 程序。

说明

- MD20106 $MC_PROG_EVENT_IGN_SINGLEBLOCK 对所有单程序段执行类型生效。
- 一般情况下可通过以下配置取消 PROG_EVENT 程序中的单程序段执行：
  MD10702 $MN_IGNORE_SINGLEBLOCK_MASK，位 0 = 1（忽略单程序段停止）
  此时 MD20106 $MC_PROG_EVENT_IGN_SINGLEBLOCK 中的设置将无效。

读取禁止生效时的特性

通过以下机床数据，可针对通道为每个触发事件设置读取禁止生效（DB21, ... DBX6.1 = 1）时“事件控制的程序调用”功能的特性：

MD20107 $MC_PROG_EVENT_IGN_INHIBIT，位 <n> = <值>

位	值	含义：在 PROG_EVENT 程序中，以下事件下系统会执行读取禁止：
0	0	零件程序开始：**生效** 如果 PROG_EVENT 程序被指令 RET 结束，则**不**会再生成可执行的程序段
	1	零件程序开始：**抑制** 如果 PROG_EVENT 程序被指令 RET 结束，则会再生成**一个**与 M17 类似的可执行程序段。
1	0	零件程序结束：**生效**
	1	零件程序结束：**抑制**
2	0	通道复位：**生效**
	1	通道复位：**抑制**
3	0	启动：**生效**
	1	启动：**抑制**

抑制程序状态和通道状态的显示更新

为了避免 HMI 操作界面中程序状态和通道状态的显示闪动，对于通常很短的 PROG_EVENT 程序执行可抑制显示更新。这样一来界面将显示激活 PROG_EVENT 程序前的程序状态和通道状态。

MD20192 $MC_PROG_EVENT_IGN_PROG_STATE，位 <n> = <值>

位	值	含义：在执行 PROG_EVENT 程序时，以下事件下系统会进行程序状态和通道状态的显示更新：
1	0	零件程序结束：**不抑制**
	1	零件程序结束：**抑制**
2	0	通道复位：**不抑制**
	1	通道复位：**抑制**
3	0	启动：**不抑制**
	1	启动：**抑制**

说明

系统变量 $AC_STAT 和 $AC_PROG 不受此功能影响，即事件控制用户程序运行时 $AC_STAT 将设置为“生效”， $AC_PROG 将设置为“运行”。

NC/PLC 接口信号 DB21, ... DBX35.0-7（“程序状态 ...”和“通道状态 ...”）同样不受此功能影响。

DB21, ... DBX7.2 / 3 / 4（NC 停止 ...）时的特性

通过以下机床数据，可针对通道为零件程序结束、通道复位和控制系统启动这些触发事件设置 NC 停止（即“NC 停止”、“程序段交界处 NC 停止”和“进给轴和主轴 NC 停止”）时“事件控制的程序调用”功能的特性：

MD20193 $MC_PROG_EVENT_IGN_STOP，位 <n> = <值>

位	值	含义：PROG_EVENT 程序在 NC 停止和事件中
1	0	零件程序结束：**停止/避免**
	1	零件程序结束：**全部执行**
2	0	通道复位：**停止/避免**
	1	通道复位：**全部执行**
3	0	启动：**停止/避免**
	1	启动：**全部执行**

应用示例

借此，操作人员可在通道复位或系统启动时按下 NC 停止键忽略执行 PROG_EVENT 程序时产生的接口信号 DB21, ... DBX7.3（NC 停止）的脉冲沿切换，并避免机床上出现意外停止特性。

说明

通过 MD20193 设置的特性不支持在 PROG_EVENT 程序中编写 DELAYFSTON/DELAYFSTOF，因为执行 DELAYFSTON 指令前的 NC 停止仍会引起中断。

10.9.11.3 编程

PROG_EVENT 程序

程序结束

若需通过零件程序开始激活用户程序，则须注意以下几点：

- 该用户程序必须通过 M17 或 RET 结束。
- 不允许通过 REPOS 指令进行跳回。

程序段显示

可以通过 PROC 指令中 DISPLOF 属性抑制当前程序段中的显示。

与 PLC 用户程序的通讯

通过写入 PROG_EVENT 程序的用户 M 功能可知道 PLC 用户程序的信息，例如：PROG_EVENT 程序的执行状态。

系统变量

查询触发事件

可在 PROG_EVENT 程序中通过以下系统变量查询触发事件：

<值> = $P_PROG_EVENT（事件控制的程序调用生效）

值	含义：激活，通过
1	零件程序开始
2	零件程序结束
3	通道复位

值	含义：激活，通过
4	启动
5	程序段搜索后在输出最后一个动作程序段后激活（参见“程序段搜索后自动启动 ASUB (页 552)”）

查询当前通道

可通过以下机床数据确定执行 PROG_EVENT 程序的通道：

<值> = $P_CHANNO（查询当前通道编号）

说明

PROG_EVENT 程序在发生对应触发事件的通道中执行。
控制系统启动事件在所有通道中同步发生。

10.9.11.4 前提条件

急停/报警

通道复位时或系统启动后若存在急停或 BAG/NC 专用报警，那么仅当在所有相关通道中进行急停应答或故障应答后才会执行 PROG_EVENT 程序。

说明

控制系统启动事件在所有通道中同步发生。

10.9.11.5 示例

示例 1：在所有事件中调用 PROG_EVENT 程序

参数设置

MD20108 $MC_PROG_EVENT_MASK = 'H0F'

在以下事件时调用 _N_PROG_EVENT_SPF：

- 零件程序开始
- 零件程序结束
- 通道复位
- 控制系统启动

编程

程序代码	注释
PROC PROG_EVENT DISPLOF	
; 零件程序开始时执行	
IF ($P_PROG_EVENT==1)	
MY_GUD_VAR=0	; 初始化 GUD 变量。
RET	
ENDIF	
; 零件程序结束和操作面板复位时执行	
IF ($P_PROG_EVENT==2) OR ($P_PROG_EVENT==3)	
DRFOF	; 取消 DRF 偏移。
IF $MC_CHAN_NAME=="CHAN1"	
CANCEL(2)	; 删除模态同步动作 2。
ENDIF	
RET	
ENDIF	
; 控制系统启动时执行	
IF ($P_PROG_EVENT==4)	
IF $MC_CHAN_NAME=="CHAN1"	
IDS=1 EVERY $A_INA[1]>5.0 DO $A_OUT[1]=1	
ENDIF	
RET	
ENDIF	
RET	

示例 2：在通道复位时调用 PROG_EVENT 程序

参数设置

MD20108 $MC_PROG_EVENT_MASK = 'H04'　在以下事件时调用 _N_PROG_EVENT_SPF：

- 操作面板复位

编程

程序代码	注释
PROC PROG_EVENT DISPLOF	
N10 DRFOF	; 取消 DRF 偏移
N20 M17	

示例 3：功能初始化

参数设置

机床数据布局，调试文件（_N_INITIAL_INI）的节选

程序代码	注释
...	
CHANDATA(3)	；通道 3 初始化
$MC_PROG_EVENT_IGN_INHIBIT='H01F'	
$MC_PROG_EVENT_MASK='H04'	；事件：通道复位
...	

含义

执行通道复位时，在第 3 个通道中启动 PROG_EVENT 程序 _N_CMA_DIR / _N_PROG_EVENT_SPF 并一直执行到其结束，不考虑是否启用读取禁止。

10.9.12 通过停止延迟区控制停止事件

10.9.12.1 功能

停止延迟区

通过 NC 程序中可有条件中断的区域，可调整对停止事件的响应。这样的程序区域被称为停止延迟区。

在停止延迟区内不会发生停止，进给率也不会改变。该程序区域执行完毕后，可能存在的停止才会生效。

因此具有以下优点：

- 在没有速度干扰的情况下，处理程序段。
- 在停止之后，如果用户使用 RESET 中断程序，则被中断的程序段就在受到保护的程序块后面。这样的程序段适用于作为下一个程序段搜索的搜索目标。
- 只要一个停止延迟区被加工，则以下的主运行轴不会被停止。
  - 指令轴
  - 使用 POSA 运动的定位轴

应用

示例：加工螺纹。

定义

停止延迟区是在零件程序中通过预定义程序 DELAYFSTON 和 DELAYFSTOF（参见“编程 (页 632)”）定义的。

停止事件

会引发停止的事件一览：

事件	响应
DB21, ... DBX7.7（复位）或 DB11 DBX20.7（BAG 复位）	立即
NC 程序：M30	报警 16954
用于启动 ASUB 的用户中断： FC 9 或 DB10 DBB1（由 PLC 置位 NC 数字量输入）	延迟
DB21, ... DBX6.2 / DB31, ... DBX2.2（删除剩余行程）通道/轴专用	立即
DB21, ... DBX6.3（删除子程序执行次数）	延迟
DB21, ... DBX6.4（程序级终止）	延迟
因单程序段停止 ● 在停止延迟区： NC 在停止延迟区外的第 1 个程序段处停止。 ● 单程序段在停止延迟区前已生效： NC 在每个程序段交界处停止，在停止延迟区中也是如此： DB21, ... DBX7.2（程序段交界处 NC 停止） **停止延迟区已取消！**	延迟
DB11 DBX21.7（激活 A 型单程序段）	延迟
DB11 DBX21.6（激活 B 型单程序段）	延迟
DB21, ... DBX7.4（NC 停止）和 DB11 DBX0.6（进给轴和主轴 BAG 停止）	立即
DB21, ... DBX7.3（程序段交界处 NC 停止）和 DB11 DBX0.5（BAG 停止）	延迟
NC 程序：因覆盖缓存为空而停止	报警 16954
NC 程序：编程或隐性 STOPRE	报警 16954
NC 程序：M0 或 M1	报警 16954
DB21, ... DBX7.2（程序段交界处 NC 停止）	延迟
子程序结束应总是撤销停止延迟区。	系统故障
NC 程序：WAITM	报警 16954

事件	响应
NC 程序：WAITE	报警 16954
NC 程序：INIT，通过参数 “S”	报警 16954
NC 程序：MMC (STRING, CHAR)	报警 16954
DB21, ... DBB26（激活/切换程序段跳跃）	延迟
DB21, ... DBB26（取消程序段跳跃）	延迟
DB21, ... DBX0.6（激活 DryRun）	延迟
DB21, ... DBX0.6（取消 DryRun）	延迟
DB21, ... DBX6.1（读取禁止）	延迟
报警：报警配置 STOPATENDBYALARM	立即
报警：报警配置 STOPBYALARM	立即
内部：插补缓存为空时在报警后停止	立即
内部：插补缓存为空时在报警后停止	立即
NC 程序：LFON 时报警 16954	报警 16954
NC 程序：WAITMC	报警 16954
NC 程序：NEWCONF	报警 16954
NC 程序：NEWCONF	报警 16954
BTSS:PI "_N_SETUDT"	延迟
扩展的停止和退回	延迟
DB31, ... DBX3.0（接收外部零点偏移）	延迟
BTSS:PI "_N_FINDST" STOPRUN	报警 16955
响应为 NOREADY 的报警	立即
DB21, ... DBX7.2 == 1（程序段交界处 NC 停止） DB21, ... DBX7.3 == 1（NC 停止） DB21, ... DBX7.4 == 1（进给轴和主轴 NC 停止）	延迟
DB21, ... DBX6.1 == 1（读取禁止）	延迟

事件	响应
DB21, ... DBX2.0 == 1（单程序段）	延迟

响应

- **立即**
  立即停止，即便在停止延迟区中也是如此。此类停止事件称为“硬停止事件”。
- **延迟**
  停止（也包括短时间的）只在停止延迟区后产生。此类停止事件称为“软停止事件”。
- **报警 16954**
  程序将被停止，因为在停止延迟区中使用了不被允许的程序命令。
- **报警 16955**
  程序继续执行，停止延迟区中进行一个不允许的动作。
- **报警 16957**
  通过 DELAYFSTON 和 DELAYFSTOF 括起来的程序区域（停机延迟区）未能激活。因此停止均会立即生效，无延迟！在停止延迟区进行制动时总是会出现此情形，即制动过程在停止延迟区前开始，但是在停止延迟区内才结束。若以倍率 0 进入停止延迟区，则同样无法激活停止延迟区（示例：借助停止延迟区前的 G4，用户可将倍率降至 0，此时停止延迟区中的下一个程序段将以倍率 0 启动，这样便出现了报警情形）。
  提示
  MD11411 $MN_ENABLE_ALARM_MASK（激活警告）位 7 会激活此报警。

说明

某些 NC 事件只会为执行切换而短暂停止，并于随后立即重新启动。例如：可短暂停止 NC 程序以启动对应 ASUB 的中断。此类事件同样可用于停止延迟区，但是其会被推移至区域的末尾并作为“软”停止事件。

说明

如果一个“硬”的停止事件碰到“停止延迟区”，那“停止延迟区”完全不再被选择。这就是说，如果在该程序段中出现另一个任意的停止，就会立即停住。只有重新编程 (更新的 DELAYFSTON) 才可开始一个新的停止延迟区。

10.9.12.2 参数设置

机床数据

G331/G332 时的停止特性

对于刚性攻丝（G331，G332），停止特性可如下设置：

MD11550 $MN_STOP_MODE_MASK

位	值	含义
0	0（缺省）	若 G331/G332 生效并额外编写了轨迹运动或暂停时间（G4），则会生成一个隐性停止延迟区。
	1	在以下情形下，可在 G331/G332 生效时停止： ● 在连续路径运行（G64）中断时。 ● 在编写位于 G33x 程序段之间的停留时间（G4）时。 必须使用 NC 指令 DELAYFSTON 和 DELAYFSTOF 来定义停止延迟区。

10.9.12.3 编程

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

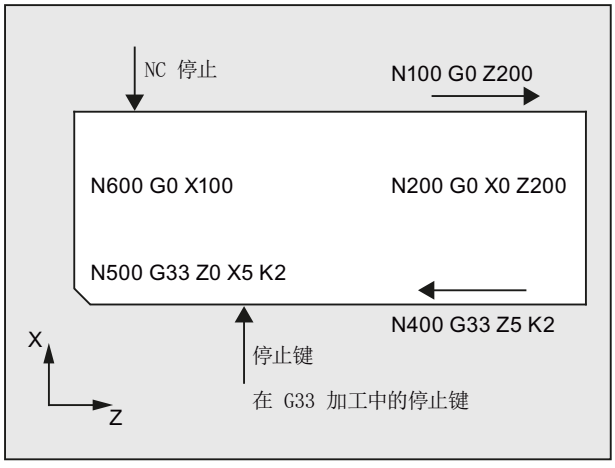
编程示例

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

10.9.12.4 边界条件

重叠

如果两个停止延迟区相互交迭，一个来自 DELAYFSTON/DELAYFSTOF，另一个来自 MD11550 $MN_STOP_MODE_MASK，则会生成尽可能大的停止延迟区。

暂停时间（G4）

G4 可在停止延迟区中使用。其它会引起临时停止的零件程序指令（例如 WAITM）则不允许使用，并会触发报警 16954。

倍率

如果在停止延迟区**前**调整倍率，倍率会在停止延迟区**中**生效。

如果在停止延迟区**中**调整倍率，倍率会在停止延迟区**后**生效。

说明

倍率 = 0

如果在停止延迟区**前**将倍率置 0，则无法激活停止延迟区！

进给禁止

DB21, ... DBX6.0（进给禁止）进给禁止在停止延迟区中不起作用，只有在离开停止延迟区后才会停住。

单程序段

如果在停止延迟区中激活单程序段，NC 会在停止延迟区外的第一个程序段处停止。

如果在停止延迟区前已经选择了单程序段，NC 会在每个程序段交界处停止，在停止延迟区中也是如此！**停止延迟区已取消。**

10.10 异步子程序（ASUB）

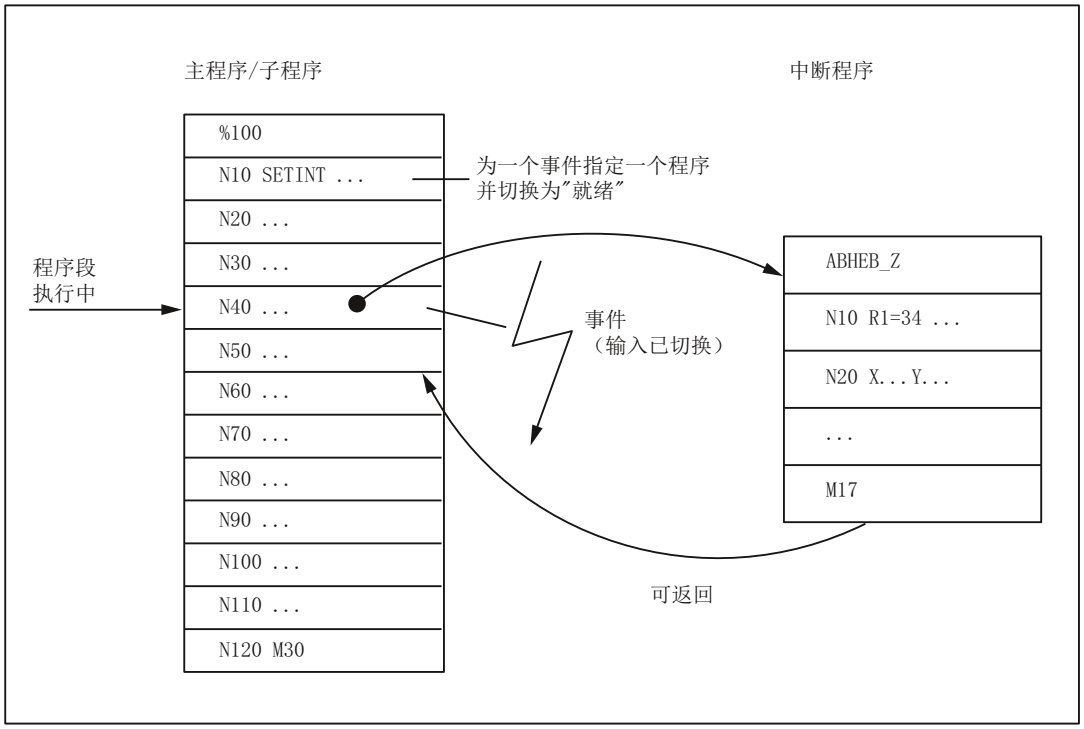
10.10.1 功能

说明

下文所用的术语“异步子程序”、“ASUB”和“中断程序”表示同一种功能。

常规

异步子程序（ASUB）是在 NC 通道中作为对异步事件（中断输入信号、过程或机床状态）的响应启动的 NC 程序。通过激活 ASUB 可以中断一个正在执行的 NC 程序。通过结束 ASUB 可在中断位置上继续执行 NC 程序。



正在通道中执行的 NC 程序在中断前可由 ASUB 进行整体或部分保护。参见章节“编程（SETINT、PRIO） (页 646)”，“灵活编程”。

定义

为了从 NC 程序中生成一个 ASUB（中断程序），必须通过指令 SETINT（参见章节“编程（SETINT、PRIO） (页 646)”）或 PI 服务“ASUB”（参见章节“PI 服务：ASUB (页 1086)”）给 NC 程序分配一个中断信号。

中断信号

- 共有 8 个外设输入可用作中断信号。
- 可通过 PLC 用户程序控制外设输入信号。
- 前四个外设输入为 NCU 模块的 4 个快速输入。可通过数据块 DB10 中的 NC/PLC 接口读取信号状态。此外，通过数据块 DB10 中的 NC/PLC 接口还可禁用输入信号。

通过 PLC 控制快速 NC 输入（中断信号）的更多相关信息请见“P3：SINUMERIK 840D sl 的 PLC 基本程序 (页 945)”一章。

文档：
功能手册之扩展功能分册；A4：NC 数字量和模拟量 I/O

调用

在程序运行中

在程序运行中，即在 AUTO 或 MDI 模式下，原则上始终可以调用 ASUB。

在程序运行外

在程序运行外，在以下运行方式、机床功能和状态下可以调用 ASUB。

- JOG，JOG-REF
- MDI-Teach In，MDI-Teach In-REF，MDI-Teach In-JOG，MDI-REF，MDI-JOG
- AUTOMATIK，程序状态“已停止”、“就绪”
- 进给轴状态“未回参考点”

如果在 JOG 或 JOG-REF 期间启动 ASUB，系统会终止当前运行。

激活

可通过以下方式激活 ASUB：

- 中断信号的 0/1 脉冲沿，通过快速 NC 输入上的 0/1 脉冲沿触发
- 调用“功能调用 ASUB”（参见章节“P3：SINUMERIK 840D sl 的 PLC 基本程序 (页 945)”）
- 通过同步动作设置输入，该输入通过短路在中断输入上进行参数设置（参见“示例 (页 649)”）
  文档：
  功能手册之同步动作分册

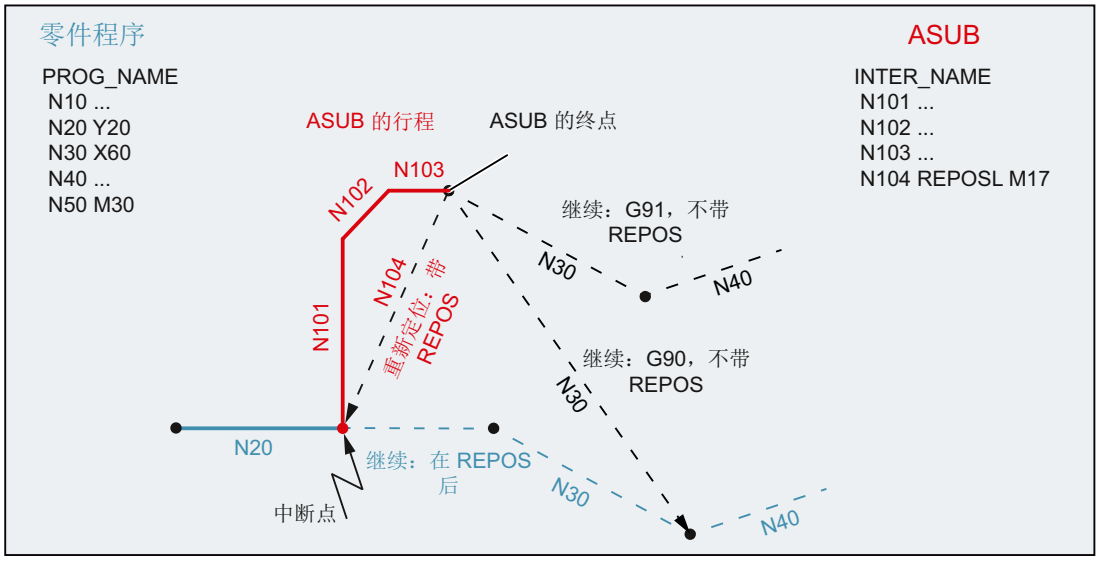
显示

激活 ASUB 通过以下 NC/PLC 接口信号显示：

DB21, … DBX378.0 == 1（ASUB 生效）

10.10.1.1 程序运行中 ASUB 的执行过程

1. 轴的制动
   激活 ASUB 后，所有机床轴都会沿制动斜坡（MD32300 $MA_MAX_AX_ACCEL）被制动到静止状态，轴位置会被保存。
2. 重新整理
   除了轴的制动外，预解码计算程序段乃至中断程序段都会被反推计算，即确定中断位置无零件程序预解码情形下的值并指定给所有变量、框架和 G 指令。这些值也会保存在缓存中，以便在 ASUB 结束后对其进行访问。
   不可进行重整的特例情形：
   – 螺纹切割程序段内
   – 几何值复杂的情形下（例如样条或半径补偿）
3. 执行 ASUB
   重整结束后，ASUB 自动启动。
   像普通子程序一样执行 ASUB（嵌套深度等）。
4. 结束 ASUB
   在执行完 ASUB 的结束标识（M02、M30、M17）后，默认情形下会向中断程序段后的零件程序段的终点位置运行。
   若需定位回中断点，则必须在 ASUB 结束处编写 REPOS 指令，例如：N104 REPOSL M17



10.10.1.2 带 REPOSA 的 ASUB

在 NC 程序因 NC 停止或报警而停止，并且随后通过 PLC 用户程序借助 FC9 触发包含 REPOSA 的 ASUB 的情况下，通常会出现以下过程：

- ASUB 或其中编写的运行被执行：
  – 程序状态：“停止”
  – DB21, ... DBX318.0（ASUB 已停止）= 1
- 在重新定位至轮廓（REPOS）前重新停止。

- 为重新定位至轮廓（REPOS），操作员触发 NC 启动：
  - DB21, ... DBX318.0（ASUB 已停止）= 0
  - 系统执行重新定位运行。
- 重新定位运行结束后，系统会置位 FC9 应答信号“ASUB 完成”，并继续执行中断的 NC 程序。

说明

NC/PLC 接口信号 DB21, ... DBX318.0（ASUB 已停止）只在以下情形下置位：
程序运行中“中断”通道状态下的中断

说明

对于无 REPOS 的 ASUB，FC9 应答信号“ASUB 完成”和 NC/PLC 接口信号 DB21, ... DBX318.0 = 0（ASUB 已停止）的复位同时出现。

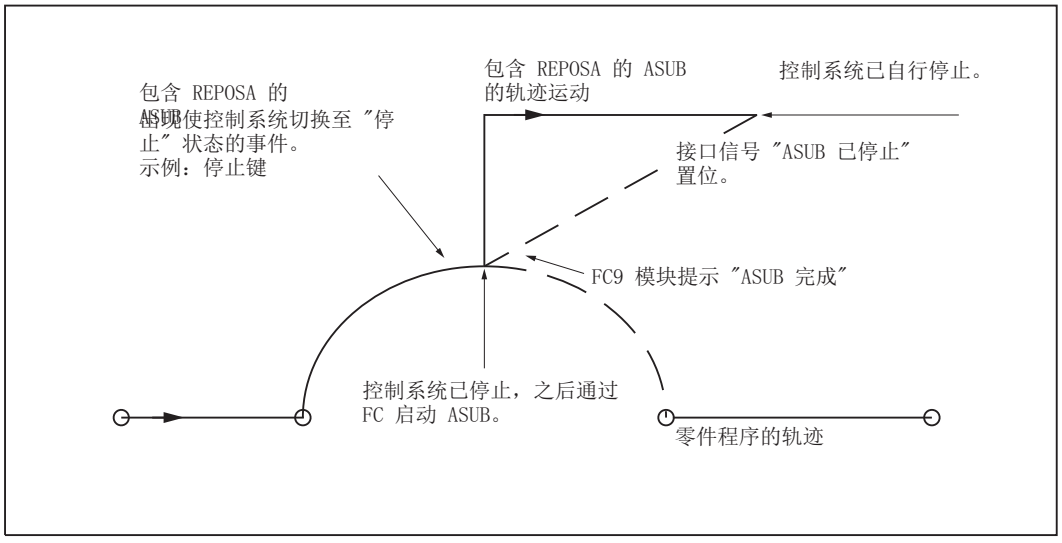


图 10-7　　示意性流程图带 REPOSA 的 ASUB

10.10.1.3 NC 特性

不同状态下的通道、BAG 或 NC 对一个激活的 ASUB 的响应参见下表：

状态	ASUB 启动	控制系统的响应
程序生效	中断，（PLC）	1. 快速退刀或停止轴 2. 在 ASUB 执行期间中断程序 3. ASUB 中编写了 REPOS 时，定位至中断位置 4. 继续执行零件程序
复位	中断，（PLC）	ASUB 如主程序运行。在 ASUB 结束处重新执行复位（无 M30）。控制系统随后的状态取决于以下机床数据： MD20110 $MC_RESET_MODE_MASK MD20112 $MC_START_MODE_MASK **文档：** 功能手册之基本功能分册；轴、坐标系、框架（K2），章节：“工件相关的实际值系统”
程序运行（AUTOMATC 或 MDI） + 通道停止	中断，（PLC）	ASUB 运行。其运行完毕时，系统重新进入停止状态。 ASUB 中编写了 REPOS 时： • ASUB 执行在定位程序段前停止。 • 定位可通过启动键触发。
	启动键	ASUB 运行后，停止的程序继续运行。
手动模式 + 通道停止	中断，（PLC）	控制系统针对响应的通道启用“内部程序执行运行方式”（对外不显示），并激活 ASUB。所选择的运行方式将保留。ASUB 结束（M17）后控制系统重新恢复原来的状态。
JOG 自动示教 自动示教回参考点	中断，（PLC）	执行停止，分析以下机床数据： MD11602 $MN_ASUP_START_MASK MD11604 $MN_ASUP_START_PRIO_LEVEL 必要时于系统内部切换至“内部程序执行运行方式”，激活 ASUB，恢复为 ASUB 启动前的状态。 JOG 模式下，配合 SETINT 定义的 LIFTFAST 不会被激活。
MDI-JOG, MDI-Teach-In, MDI-Teach 参考点	中断，（PLC）	
手动模式 + 通道运行	中断，（PLC）	停止当前生效的运动。删除剩余行程。之后的过程与“手动模式，通道停止”相同。
用户报警 65500 - 65999， 通道处于复位状态	中断，（PLC）	设置了以下机床数据时，存在报警响应“NC 启动禁用”的情形下系统仍会执行用户 ASUB： MD20194 $MC_IGNORE_NONCSTART_ASUP

状态	ASUB 启动	控制系统的响应
执行 INITIAL.INI	无法使用	系统会生成“无法请求中断”信号。
程序段搜索		
无法通过 NC 启动消除的报警		
启用数字化		
通道处于故障状态		
其他用户报警 65500 - 65999， 通道处于复位状态， MD20194 未生效	无法使用	生成信号“请求已因报警而终止”。

10.10.2 调试：机床数据

10.10.2.1 NC 专用：BAG 专用 NC/PLC 接口信号和运行方式切换

通过机床数据定义 DB11 的 BAG 专用 NC/PLC 接口信号的生效方式，以及执行运行方式切换的通道：

MD11600 $MN_BAG_MASK = <值>

值	含义
0	DB11 的 BAG 专用 NC/PLC 接口信号**生效**。 在运行方式组的**所有**通道中进行内部运行方式切换。
1	DB11 的 BAG 专用 NC/PLC 接口信号**不**生效。 **仅**在 **ASUB** 生效的通道中进行内部运行方式切换。
2	DB11 的 BAG 专用 NC/PLC 接口信号**生效**。 **仅**在 **ASUB** 生效的通道中进行内部运行方式切换。
3	DB11 的 BAG 专用 NC/PLC 接口信号**不**生效。 **仅**在 **ASUB** 生效的通道中进行内部运行方式切换。

说明

多通道系统

若在多通道系统中可使用“JOG 运行方式下 ASUB 中断期间手动运行”功能（见下文），则必须将 MD11600 $MN_BAG_MASK 设置成值“2”或“3”。

参见

编程（SETINT、PRIO） (页 646)

10.10.2.2 NC 专用：ASUB 启动使能

通过机床数据定义在 ASUB 启动时忽略哪些停止原因：

MD11602 $MN_ASUP_START_MASK，<位> = <值>

位	值	含义
0	0	在因停止键、M0 或 M01 停止的情况下，不启动 ASUB。
	1	尽管因停止键、M0 或 M01 停止，仍**启动** ASUB。
2	0	在启动 ASUB 时，在通道中存在读取禁止的情况下，以下通道专用机床数据中的设置生效： • MD20107 $MC_PROG_EVENT_IGN_INHIBIT • MD20116 $MC_IGNORE_INHIBIT_ASUP
	1	即便在相应通道中启用读取禁止的情况下，仍在 NC 的所有通道中启动用户或系统 ASUB。 以下通道专用机床数据中的设置不生效： • MD20107 $MC_PROG_EVENT_IGN_INHIBIT • MD20116 $MC_IGNORE_INHIBIT_ASUP **注意** 尽管存在读取禁止，启动使能仍在 NC 的所有通道中生效。这既适用于用户 ASUB，也适用于系统 ASUB。因此**强烈建议**使用**通道专用**使能来替代 NC 专用使能。
3	0	若在 JOG 运行模式中启动 ASUB 并通过 NC 停止中断，在此状态下不可手动运行轴。
	1	若在 JOG 运行模式中启动 ASUB 并通过 NC 停止中断，在此状态下可以手动运行轴。 可通过 NC 启动继续执行 ASUB。其中自动进行 REPOS。 **提示** 若在控制系统中参数设置了超过一个通道，必须额外设置以下机床数据： MD11600 $MN_BAG_MASK，位 1 = 1

手动启动使能

若在参数设置了启动使能的情况下 ASUB 未**自动**启动，可通过 PLC 用户程序借助 NC/PLC 接口信号（DB21, ... DBX7.1），或者通过手动操作 NC 启动来启动 ASUB。

说明

用于“从轮廓快速退刀”（LIFTFAST）的 ASUB 在任何情形下都会启动。

10.10.2.3 NC 专用：参数设置的启动使能的生效范围

通过以下机床数据设置：从最高优先级起至哪个 ASUB 优先级 (页 646)为止，MD11602 $MN_ASUP_START_MASK 中的设置生效。

MD11604 $MN_ASUP_START_PRIO_LEVEL = <ASUB 优先级>

示例

MD11604 $MN_ASUP_START_PRIO_LEVEL = 5

MD11602 $MN_ASUP_START_MASK 中的设置对优先级为 1 → 5 的 ASUB 生效。

10.10.2.4 通道专用：在轴未回参考点的情况下仍使能启动

通过以下机床数据设置：在哪些中断中，在参数设置了“不带参考点的 NC 启动禁用”（MD20700 $MC_REFP_NC_START_LOCK）的情况下仍启动对应的 ASUB：

MD20115 $MC_IGNORE_REFP_LOCK_ASUP，位 (1 - <中断>) = TRUE

注意

系统中断

通过 MD20115 $MC_IGNORE_REFP_LOCK_ASUP，位 8 - 31 使能与系统中断对应的 ASUB。

通过位 8 / 中断 9 启动一个包含运行的 ASUB。

NC 专用 ASUB 启动使能

若 MD11602 $MN_ASUP_START_MASK 位 2 == TRUE，那么尽管参数设置了通道专用功能“不带参考点的 NC 启动禁用”（MD20700 $MC_REFP_NC_START_LOCK），仍对 NC 的所有通道进行 ASUB 使能。因此**强烈建议**使用**通道专用**使能来替代 NC 专用使能。

10.10.2.5 通道专用：在读取禁止的情况下仍使能启动

通过以下机床数据设置：在哪些中断中，在通道中启用读取禁止（DB21, ... DBX6.1）的情况下仍启动对应的 ASUB：

MD20116 $MC_IGNORE_INHIBIT_ASUP，位 (1 - <中断>) = TRUE

注意
系统中断 通过 MD20116 $MC_IGNORE_INHIBIT_ASUP，位 8 - 31 使能与系统中断对应的 ASUB。 通过位 8 / 中断 9 启动一个包含运行的 ASUB。 **NC 专用 ASUB 启动使能** 若 MD11602 $MN_ASUP_START_MASK 位 2 == TRUE，则在 NC 的所有通道中忽略 MD20116 $MC_IGNORE_INHIBIT_ASUP 中的通道专用设置。因此**强烈建议**使用**通道专用**使能来替代 NC 专用使能。

10.10.2.6 通道专用：在采用单程序段的情况下仍连续执行

通过以下机床数据设置：在哪些中断中，在通道中启用单程序段处理（DB21, ... DBX0.4）的情况下仍**连续**地，即以无逐段中断的方式执行对应的 ASUB：

MD20117 $MC_IGNORE_SINGLEBLOCK_ASUP，位 (1 - <中断>) = TRUE

边界条件

MD20117 $MC_IGNORE_SINGLEBLOCK_ASUP 中的设置仅在单程序段 SBL1（主处理单程序段）中生效。

注意
系统中断 通过 MD20117 $MC_IGNORE_SINGLEBLOCK_ASUP，位 8 - 31 使能与系统中断对应的 ASUB。 通过位 8 / 中断 9 启动一个包含运行的 ASUB。 **NC 专用 ASUB 启动使能** 若 MD10702 $MN_IGNORE_SINGLEBLOCK_MASK 位 1 == TRUE，则在 NC 的所有通道中忽略 MD20117 $MC_IGNORE_SINGLEBLOCK_ASUP 中的通道专用设置。因此**强烈建议**使用**通道专用**使能来替代 NC 专用使能。

10.10.2.7 通道专用：显示更新

通过以下机床数据设置：在执行 ASUB 期间不更新显示，从而防止在执行非常短的 ASUB 时，操作界面上的程序状态和通道状态的显示闪动：

MD20191 $MC_IGN_PROG_STATE_ASUP，位 (1 - <中断>) = TRUE

说明

NC/PLC 接口信号

在抑制显示的情形下执行 ASUB 时，以下 NC/PLC 接口信号置位：

DB21, … DBX378.1 = 1（“静止”ASUB 生效）

系统变量和 NC/PLC 接口信号

在执行 ASUB 期间抑制显示时，用于程序状态和通道状态的系统变量和 NC/PLC 接口信号不会受影响：

- $AC_STAT（通道状态）
- $AC_PROG（程序状态）
- DB21, ... DBX35.5 - 7（通道状态）
- DB21, ... DBX35.0 - 4（程序状态）

10.10.3 编程：系统变量

10.10.3.1 REPOS 可能性（$P_REPINF）

对于 ASUB，可能存在无法退回重新逼近轮廓（REPOS）的情形。

通过以下系统变量，能够在 ASUB 中读取是否能够进行 REPOS：

<值> = $P_REPINF

值	含义
0	不可通过 REPOS 重新定位，因为： • 未在 ASUB 中调用 • ASUB 从复位状态开始运行 • ASUB 从 JOG 模式开始运行
1	可在 ASUB 中通过 REPOS 重新定位

10.10.3.2 激活事件（$AC_ASUP）

通过系统变量 $AC_ASUP 能够读取有关导致 ASUB 激活的事件的下列信息：

- 为何激活 ASUB，例如位 0：用户中断“采用 Blsync 的 ASUB”
- 如何激活 ASUB，例如位 0：NC/PLC 接口信号，数字量模拟量接口
- 可采用哪些继续执行的方式，例如位 0：可自由选择 REORG 或 RET

10.10.4 编程（SETINT、PRIO）

指定对应关系：中断信号与 NC 程序

通过指令 SETINT 指定 NC 程序与中断信号的对应关系。进而确定 NC 程序与 ASUB 的对应关系。

句法

```
SETINT(<n>) <NC 程序>
```

含义

SETINT：	NC 程序与中断信号的对应关系	
<n>：	中断信号编号	
	值域：	0, 1, 2, ... 8
<NC 程序：	程序名	

示例

程序代码	注释
N20 SETINT(3) ABHEBEN_Z	; IF 输入 3 == 1 ; 则启动 ASUB“ABHEBEN_Z”

此外还可配合 SETINT 编写以下指令：

- LIFTFAST
  出现中断信号时，在 ASUB 启动前执行“从轮廓快速退刀”。快速退刀的运动方向通过程序指令 ALF 定义。
- BLSYNC
  出现中断信号时仍继续执行运行中的程序段，之后再启动 ASUB。

说明

在下列条件下，指定的“中断信号 ↔ 零件程序”对应关系会被清除：
- 通道处于复位状态
- 零件程序中编写 CLRINT 指令

优先级

如果在一个 NC 程序中通过 SETINT 激活了多个中断，则须为对应的 NC 程序或 ASUB 指定不同的优先级。

句法
PRIO=<值>

含义

PRIO：	用于确定中断优先级的关键字
<值>：	优先级：1, 2, 3 ... 128。1 对应最高优先级。

示例

程序代码	注释
N20 SETINT(3) PRIO=2 ABHEBEN_Z	; IF 输入 3 == 1
	; 则启动 ASUB“ABHEBEN_Z”
N30 SETINT(2) PRIO=3 ABHEBEN_X	; IF 输入 2 == 1
	; 则启动 ASUB“ABHEBEN_X”

如果输入端 2 和 3 同时保留，则 ASUB 会根据级别数的顺序进行处理：

1. “ABHEBEN_Z”
2. “ABHEBEN_X”。

其他中断专用指令

指令	含义
SAVE	如果在 ASUB 中使用指令 SAVE，那么 ASUB 结束时，中断之前在中断的 NC 程序中生效的 G 指令、框架和转换重新生效。
DISABLE ENABLE	通过指令对 DISABLE ENALBE 可保护程序段，防止其被 ASUB 中断。 通过 SETINT 指定的中断信号与 NC 程序之间的对应关系仍然保留。 在 ENALBE 之后通过中断信号的下一个 0/1 沿启动 ASUB。
CLRINT(<n>)	删除中断信号与通过 SETINT 指定的 NC 程序的对应关系。

文档

编程手册之工作准备分册；章节：“灵活 NC 编程”>“中断程序（ASUB）”

参见

编程 (页 652)

10.10.5 前提条件

跨运行方式的 ASUB 启动

需要检查的设置

- MD11600 $MN_BAG_MASK
- MD11604 $MN_ASUP_START_PRIO_LEVEL
- 中断指定的优先级

推荐设置

NC 专用机床数据：

- MD11600 $MN_BAG_MASK = 'H3'

说明

在采用此设置时需要注意，DB11 的 BAG 专用 NC/PLC 接口信号不再对执行 ASUB 的通道生效。若不期望此特性，则可使用设置 MD11600 $MN_BAG_MASK = 'H2' 作为替代（参见“调试：机床数据 (页 641)”）。

MD11602 $MN_ASUP_START_MASK = 'H5'

- MD11604 $MN_ASUP_START_PRIO_LEVEL = 7

通道专用机床数据，用于启动 ASUB 的通道，或适用于所有通道：

- MD20105 $MC_PROG_EVENT_IGN_REFP_LOCK，位 <n> = TRUE
  <n>：用于全部所需事件控制的程序调用（程序事件）
- MD20115 $MC_IGNORE_REFP_LOCK_ASUP，位 <n> = TRUE
  <n>：用于全部**所需用户**中断

注意
系统中断 通过 MD20115 $MC_IGNORE_REFP_LOCK_ASUP，位 8 - 31 使能系统中断。 通过位 8 / 中断 9 启动一个包含运行的 ASUB。

10.10.6 示例

通过中断从同步动作中激活 ASUB

1. 设置两个生效的数字量输入/输出字节：
   – MD10350 $MN_FASTIO_DIG_NUM_INPUTS = 2
   – MD10360 $MN_FASTIO_DIG_NUM_OUTPUTS = 2
2. 通过逻辑或运算将短路设为从输出 9 到输入 9：
   – 输入 1，输入字节 2 =（输出 1，输出字节 2）**或** 硬件输入信号 1，输入字节 2）：
   MD10361 $MN_FASTIO_DIG_SHORT_CIRCUIT[0] = 'H0102B102'
3. 将硬件输入字节指定给中断编程 SETINT：
   – 输入字节 2：
   MD21210 $MC_SETINT_ASSIGN_FASTIN = 2

4. 将输入定义为 ASUB 触发器：
   – 第二个输入字节中的输入 1，即绝对输入 9，启动程序 SYNCASUP
   ```
   SETINT(1) PRIO=1 SYNCASUP
   ```
5. 定义用于设置输出的同步动作：
   – 如果标准轨迹参数 >= 0.5，带 ID 1 的同步动作会将输出 9 始终设为 1。
   ```
   IDS=1 EVERY $$AC_PATHN >= 0.5 DO $A_OUT[9]=1
   ```
   从输出 9 到输入 9 的短路会触发中断 1 且 NC 程序 “SYNCASUP” 会作为 ASUB 启动。

10.11 用于 RET 和 REPOS 的用户专用 ASUB

10.11.1 功能

功能

控制系统软件包含一个用于实现 NC 程序结束 (RET) 和重新逼近轮廓 (REPOS) 功能的西门子专用 ASUB。机床制造商可用一个用户专用 ASUB 替换系统 ASUB。

⚠ 危险
编程错误 如果该 ASUB 导致编程错误，责任由机床制造商自行承担。

10.11.2 参数设置

安装

ASUB 名称

必须为用户专用 ASUB 给定以下名称：

- **_N_ASUP_SPF**

ASUB 目录

用户专用 ASUB "_N_ASUP_SPF" 必须保存在以下其中一个目录中：

- **_N_CMA_DIR**（制造商目录）
- **_N_CUS_DIR**（用户目录）

激活和查找路径

在以下机床数据中通过位 0 和 1 设置激活哪个事件中的用户专用 ASUB "_N_ASUP_SPF"。

通过位 2 设置在激活用户专用 ASUB "_N_ASUP_SPF" 时从哪个位置开始查找。

MD11610 $MN_ASUP_EDITABLE，位 0、1、2 = <值>

位	值	含义
0 和 1	0	在程序结束 (**RET**) 或重新逼近轮廓 (**REPOS**) 时不会激活用户专用 ASUB。
	1	执行 **RET** 时会激活**用户专用** ASUB。 执行 **REPOS** 时会激活**系统专用** ASUB。
	2	执行 **RET** 时会激活**系统专用** ASUB。 执行 **REPOS** 时会激活**用户专用** ASUB。
	3	执行 **RET** 和 **REPOS** 时会激活**用户专用** ASUB。
2	0	首先在用户目录 **_N_CUS_DIR** 中搜索用户专用 ASUB。
	1	首先在制造商目录 **_N_CMA_DIR** 中搜索用户专用 ASUB。

定义保护级

需要为 RET 和/或 REPOS 启用用户专用 ASUB 时（MD11610 $MN_ASUP_EDITABLE ≠ 0），可为用户专用程序 “_N_ASUP_SPF” 定义一个保护级。保护级的取值范围为 0 - 7。此时通过以下机床数据设置：

MD11612 $MN_ASUP_EDIT_PROTECTION_LEVEL = <用户专用 ASUB 的保护级>

保护级的详细信息请见：

文档：

调试手册之保护级方案

启用单程序段执行时的特性

通过设置以下机床数据，可在启用单程序段执行的情形下实现系统专用 ASUB 和用户专用 ASUB “_N_ASUP_SPF” 的无中断执行：

MD10702 $MN_IGNORE_SINGLEBLOCK_MASK，位 0 = <值>

位	值	含义
0	0	单程序段运行激活时，系统会在**每个 ASUB 程序段中停止**。
	1	单程序段运行激活时，系统会**无中断地执行 ASUB**。

10.11.3 编程

确定 ASUB 激活的原因

引起 ASUB 激活的原因可通过系统变量 $AC_ASUP 以位编码读取。

继续执行

使用系统 ASUB 时，在 ASUB 内执行动作后的继续执行特性为固定设置：

- 系统 ASUB1 → 通过 RET（子程序返回）继续执行
- 系统 ASUB2 → 通过 REPOS（重新定位）继续执行

在系统变量的说明中，在“通过...继续执行”下注明了针对每种原因的系统 ASUB 相关特性。

说明

使用用户专用 ASUB 时的继续执行

使用用户专用 ASUB 时，建议保留对应的系统 ASUB 继续执行特性。

原因：运行方式切换（$AC_ASUP，位 9 == 1）

运行方式组切换时，继续执行特性取决于以下机床数据：

MD20114 $MC_MODESWITCH_MASK（通过运行方式切换中断 MDI）

- 位 0 == 0：系统 ASUB1 → 通过 RET 继续执行
- 位 0 == 1：系统 ASUB2 → 通过 REPOS 继续执行

文档

对系统变量的详细说明请见：

参数手册“系统变量”

10.12 在存在用户报警的情形下启动 ASUB

10.12.1 功能

说明

ASUB 可在各种情形下由用户、系统或通过事件控制来触发。不过在某些情形下，报警响应为“NC 启动禁用”的用户报警会阻止 ASUB 启动。存在报警响应时，系统会通过报警 16906 应答复位触发的用户 ASUB。

通过设置特定的通道专用机床数据，在报警响应为“NC 启动禁用”的用户报警生效的情形下同样可启动和执行 ASUB。此时系统会为 ASUB 启动抑制该报警响应，从而可执行程序。

说明

报警响应为“NC 启动禁用”的 NC 报警则不会受此抑制影响。复位触发的 ASUB 仍无法启动，系统会通过报警 16906 将其拒绝。

可抑制的用户报警

可抑制的用户报警的序号带划分如下：

序号范围	影响	删除
65000 - 65499	显示，NC 启动禁用	复位
65500 - 65999	显示，NC 启动禁用（设置 MD20194 时不适用于 ASUB）	复位
66000 - 66999	显示，NC 启动禁用，执行预编码程序段后运动停止	复位
67000 - 67999	显示	Cancel
68000 - 68999	显示，NC 启动禁用，即刻插补停止	复位
69000 - 69999	显示，NC 启动禁用，在下一个程序段结束处停止	复位

说明

对于以下报警序号带，尽管可抑制报警响应“NC 启动禁用”，但相应报警的其他响应会构成停止条件。这些停止条件无法通过本章节中描述的功能抑制：

- 66000 – 66999
- 68000 – 68999
- 69000 – 69999

10.12.2 激活

设置

可通过以下通道专用机床数据独立设置每个 ASUB 通道：

MD20194 $MC_IGNORE_NONCSTART_ASUP（针对特定用户报警在报警应答“NC 启动禁用”存在的情形下支持 ASUB 启动）

对此机床数据设置的修改需通过零件程序指令 NEWCONF 或操作界面上的软键激活。

过程

ASUB 启动的一般过程如下：

- 相应设置机床数据 MD20194 $MC_IGNORE_NONCSTART_ASUP，并通过 NEWCONF 激活。
- 启动零件程序。
  出现可抑制序号带内的用户报警，例如报警 65500。同步动作或零件程序指令均可能触发该报警。
- 尽管存在报警，系统仍执行零件程序，直至到达程序结束 M02、M30 或 M17。
  通道状态“复位”生效。
- 系统执行复位触发的用户 ASUB。

说明

存在报警响应“NC 启动禁用”时，也可从运行中的或停止的零件程序启动 ASUB。这对响应为“NC 启动禁用”的用户报警和 NC 报警均适用。

说明

若执行用户 ASUB 期间零件程序停止，则无法再通过 NC 启动键继续执行 ASUB，例如 ASUB 中写入了 M0 或启用用户停止时。系统会通过报警 16906 拒绝报警响应“NC 启动禁用”。之前生成的用户报警仍可通过复位应答。

10.12.3 示例

10.12.3.1 通过复位触发的用户 ASUB - 示例 1

在此应用中 MD20194 不置位。

主程序

程序代码

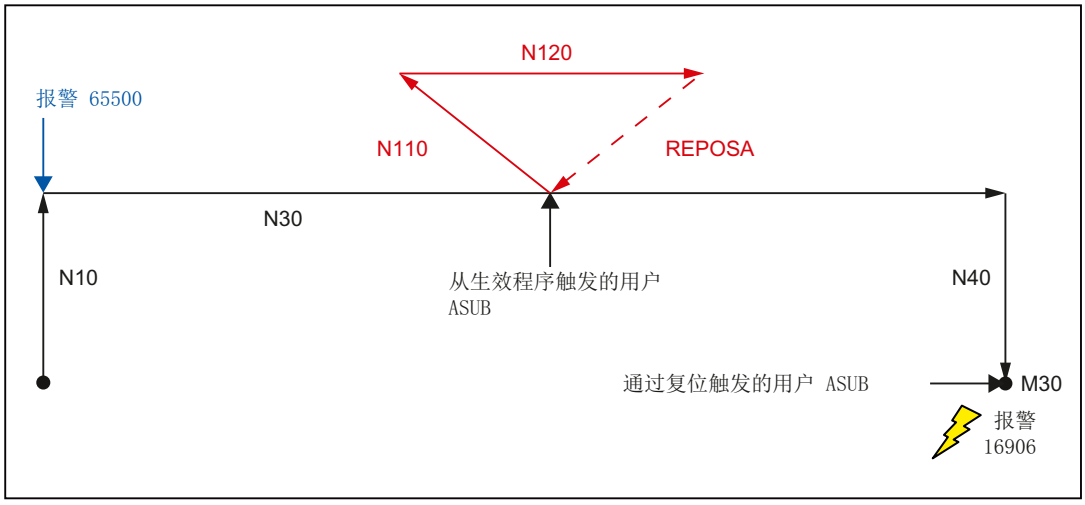
```
N10 G90 G0 Z10
N20 SETAL(65500)
N30 X100
N40 Z0
N50 M30
```

用户 ASUB

程序代码

```
N110 G91 G0 X-10 Z5
N120 X20
N130 REPOSA
```

过程



执行程序段 N10。 出现报警 65500，其响应为“显示”和“NC 启动禁用”。 接下来零件程序不停止。 切换至程序段 N30 并运行。 若在程序段中央启用用户 ASUB，尽管存在报警响应“NC 启动禁用”，系统仍会予以执行。 REPOSA 在程序中断处重新开始，并运行零件程序直至程序结束 M30。 若此时启用用户 ASUB，则会被系统拒绝，并触发报警 16906。 NC 处于复位状态。

10.12.3.2 通过复位触发的用户 ASUB - 示例 2

此应用为一般情形，MD20194 置位。

主程序

程序代码

```
N4 $MC_IGNORE_NONCSTART_ASUP=1
N6 NEWCONF
N10 G90 G0 Z10
N20 SETAL(65500)
N30 X100
N40 Z0
N50 M30
```

用户 ASUB

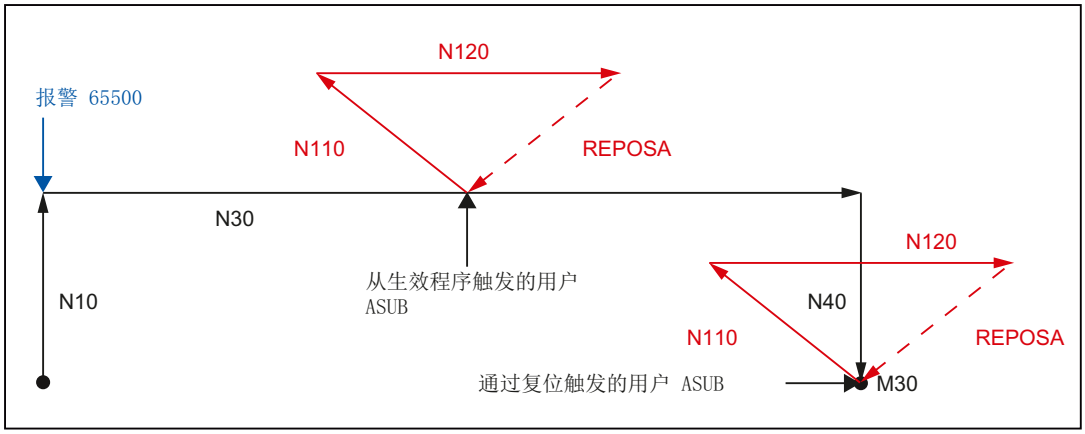
程序代码

```
N110 G91 G0 X-10 Z5
```

程序代码

```
N120 X20
N130 REPOSA
```

过程



为 ASUB 通道 1 设置机床数据 MD20194 $MC_IGNORE_NONCSTART_ASUP，并通过 NEWCONF 激活。 执行程序段 N10。 出现报警 65500，其响应为“显示”和“NC 启动禁用”。 接下来零件程序不停止。 切换至程序段 N30 并运行。 若在程序段中央启用用户 ASUB，尽管存在报警响应“NC 启动禁用”，系统仍会予以执行。 REPOSA 在程序中断处重新开始，并运行零件程序直至程序结束 M30。 若此时启用用户 ASUB，系统会基于设置的机床数据 MD20194 重新执行。

10.12.3.3 写入 M0 的用户 ASUB

在此应用中 MD20194 置位。

主程序

程序代码

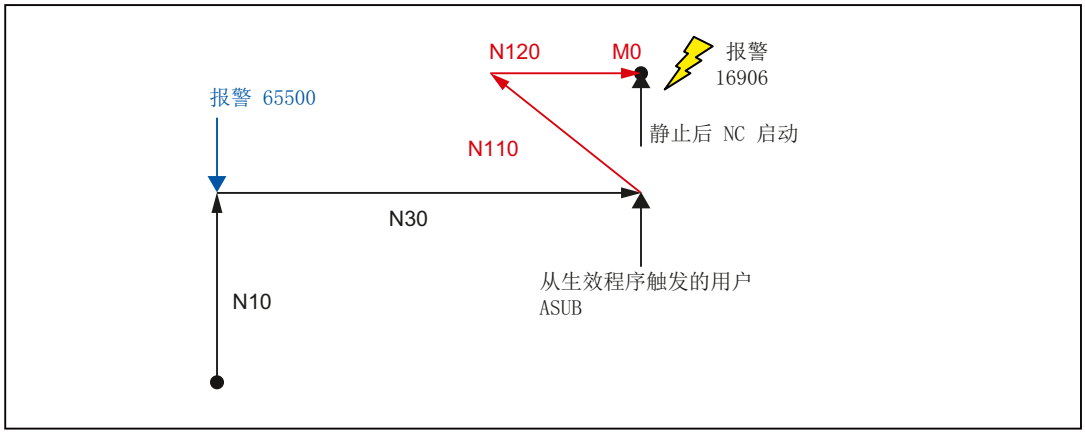
```
N4 $MC_IGNORE_NONCSTART_ASUP=1
N6 NEWCONF
N10 G90 G0 Z10
N20 SETAL(65500)
N30 X100
N40 Z0
N50 M30
```

用户 ASUB

程序代码

```
N110 G91 G0 X-10 Z5
N120 X10
N122 M0
N124 X10
N130 REPOSA
```

过程



为 ASUB 通道 1 设置机床数据 MD20194 $MC_IGNORE_NONCSTART_ASUP，并通过 NEWCONF 激活。 执行程序段 N10。 出现报警 65500，其响应为“显示”和“NC 启动禁用”。 接下来零件程序不停止。 切换至程序段 N30 并运行。 若在程序段中央启用用户 ASUB，尽管存在报警响应“NC 启动禁用”，系统仍会予以执行。 在程序段 N122 处，M0 使零件程序停止。 无法再通过按下 NC 启动键继续执行 ASUB。 系统会通过报警 16906 拒绝报警响应“NC 启动禁用”。 之前生成的报警 65500 仍可通过复位应答。

10.12.3.4 带停止的用户 ASUB

在此应用中 MD20194 置位。

主程序

程序代码

```
N4 $MC_IGNORE_NONCSTART_ASUP=1
N6 NEWCONF
N10 G90 G0 Z10
```

程序代码

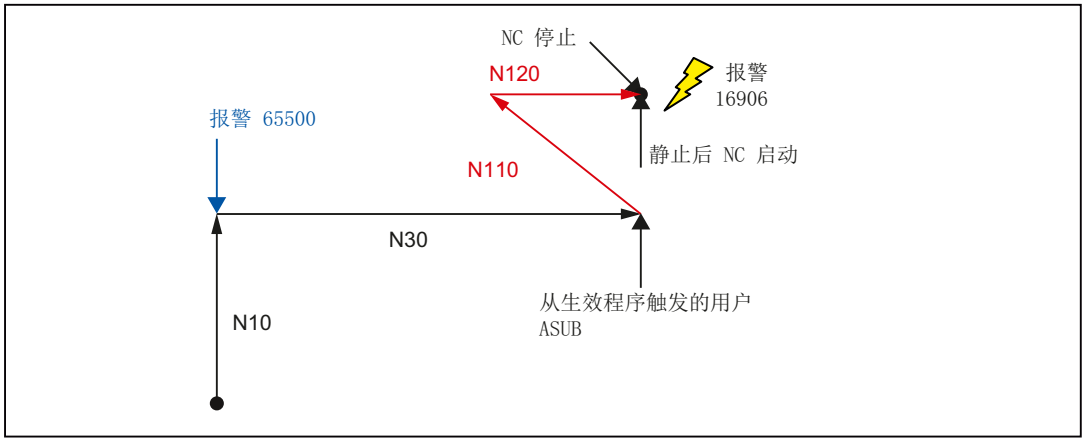
```
N20 SETAL(65500)
N30 X100
N40 Z0
N50 M30
```

用户 ASUB

程序代码

```
N110 G91 G0 X-10 Z5
N120 X20
N130 REPOSA
```

过程



为 ASUB 通道 1 设置机床数据 MD20194 $MC_IGNORE_NONCSTART_ASUP，并通过 NEWCONF 激活。 执行程序段 N10。 出现报警 65500，其响应为“显示”和“NC 启动禁用”。 接下来零件程序不停止。 切换至程序段 N30 并运行。 若在程序段中央启用用户 ASUB，尽管存在报警响应“NC 启动禁用”，系统仍会予以执行。 若在 N120 处在程序段中央按下 NC 停止键，ASUB 将停止。 无法再通过按下 NC 启动键继续执行 ASUB。 系统会通过报警 16906 拒绝报警响应“NC 启动禁用”。 之前生成的报警 65500 仍可通过复位应答。

10.12.3.5 从停止状态触发的用户 ASUB

在此应用中 MD20194 可置位或不置位。

主程序

程序代码

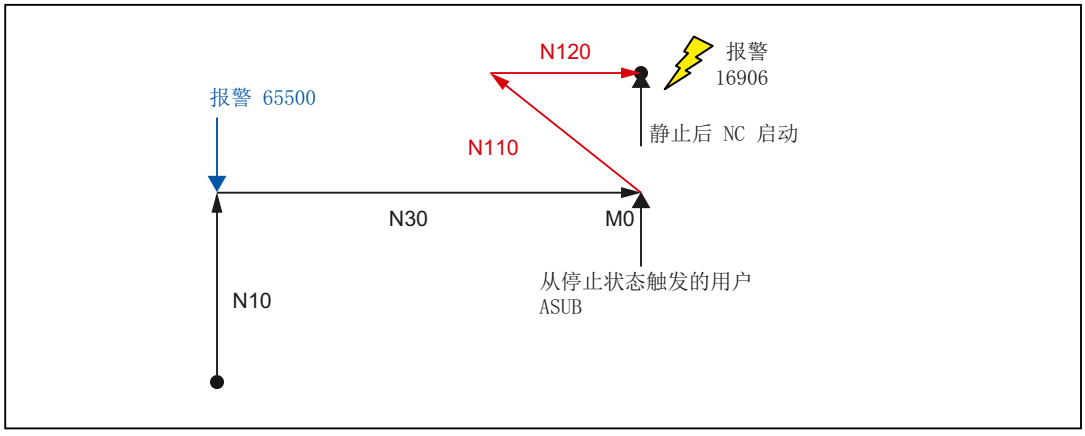
```
N10 G90 G0 Z10
N20 SETAL(65500)
N30 X50
N35 M0
N38 X100
N40 Z0
N50 M30
```

用户 ASUB

程序代码

```
N110 G91 G0 X-10 Z5
N120 X20
N130 REPOSA
```

过程



执行程序段 N10。 出现报警 65500，其响应为“显示”和“NC 启动禁用”。 接下来零件程序不停止。 切换至程序段 N30 并运行。 在程序段 N35 处，M0 使零件程序停止。 NC 处于停止状态。 若此时启用用户 ASUB，尽管 NC 处于停止状态且存在报警响应“NC 启动禁用”，系统仍会予以执行。 机床数据 MD20194 $MC_IGNORE_NONCSTART_ASUP 是否置位对此过程无影响。 程序段 N120 后，ASUB 在 REPOSA 前停止。 下一次 NC 启动时才可进行重新定位。 但是报警响应“NC 启动禁用”会使 NC 启动被拒绝，并触发报警 16906。 之前生成的报警 65500 仍可通过复位应答。

10.13 单程序段

"单程序段"功能以三种不同方式实现 NC 程序的逐段执行。

可供使用的单程序段类型如下：

- **"SBL1:Einzelsatz grob"** (Stopp nach jedem Hauptlauf-Satz)
  在每个完整执行的主处理程序段后停止 NC 程序或加工。
- **"SBL2:Rechensatz"** (Stopp nach jedem decodierten Satz)
  在每个解码程序段后停止 NC 程序或加工。
  如果由一个解码程序段生成了多个主处理程序段，则在每个主处理程序段后停止。

  说明

  螺纹切削程序段

  对于螺纹切削程序段而言，NC 程序和加工不会停止。
- **"SBL3:Einzelsatz fein"** (Stopp nach jedem decodierten Satz auch in einem Zyklus)
  如 SBL2，但即使在循环内也会在每个程序段后停止 NC 程序或加工。

操作界面：选择单程序段类型

通过 HMI 操作界面选择单程序段类型 SBL1、SBL2 或 SBL3：

“加工”操作区 > “程序控制” > 菜单：“程序控制”

10.13.1 参数设置

接口信号

激活

“单程序段”通过以下 NC/PLC 接口信号针对特定通道激活：

DB21, ... DBX0.4（激活单程序段）

机床数据

关闭单程序段执行（MD10702、MD20106、MD20117）

- 针对特定加工情形和程序类型，可通过以下机床数据进行设置，使得在单程序段功能生效的情况下不发生停止：
  MD10702 $MN_IGNORE_SINGLEBLOCK_MASK

说明

- 通过在 ASUB 或子程序内编写 (页 663) SBLON/SBLOF，能够显性激活/关闭单程序段执行。
- Bei Einzelsatztyp "SBL2:Rechensatz" wirkt das Maschinendatum nur bei System-ASUPs, Anwender-ASUPs und Unterprogrammen mit dem Attribut DISPLOF.

- 事件控制的程序调用（Prog-Events）在单程序段方面的特性通过以下机床数据设置：
  MD20106 $MC_PROG_EVENT_IGN_SINGLEBLOCK
- 中断程序（ASUB）在单程序段方面的特性通过以下机床数据设置：
  MD20117 $MC_IGNORE_SINGLEBLOCK_ASUP
  在单程序段模式下激活了一个 ASUB 时，该 ASUB 将完整执行。直到该 ASUB 结束或在首个未激活单程序段抑制的插补程序段中，单程序段才会重新生效。如果在从 ASUB 过渡到 NC 程序时轨迹速度过大，以至于无法在后续程序段中制动至静止状态，那么例如在连续路径运行 G64 生效时，视情况可通过再后面的多个程序段进行制动。

说明

在 ASUP 内编写 (页 663) SBLON 后，在此情形下**无法**重新激活单程序段执行。

设定数据

Debugmode für Einzelsatz "SBL2:Rechensatz" (SD42200)

Bedingt durch die vorauseilende Dekodierung der Sätze, kann auf der Bedienoberfläche der Bezug zwischen der hauptlaufbezogenen aktuellen Satzanzeige und den angezeigten Variablenwerten verloren gehen.Es werden dann u. U. nicht plausible Variablenwerte angezeigt.

Mit dem folgenden kanalspezifischen Settingdatum kann eingestellt werden, dass bei aktivem Einzelsatz "SBL2:Rechensatz" bei jedem Satz ein Vorlaufstopp ausgeführt wird. 这样一来，零件程序段的预先执行会被抑制，当前程序段显示与变量值显示之间的关联得以保留。

SD42200 $SC_SINGLEBLOCK2_STOPRE（激活 SBL2 的排故模式）

说明

轮廓偏差

Beim Abarbeiten von Verfahrsätzen mit Einzelsatztyp "**SBL2:Rechensatz**" im **Debugmode** kann es zu Konturabweichungen kommen.

10.13.2 编程

10.13.2.1 关闭/激活单程序段执行（SBLOF，SBLON）

为整个 NC 程序关闭单程序段执行

若在**主程序**的第一行（PROC）中通过编程关闭单程序段执行（SBLOF），则其一直生效，直至该 NC 程序结束或终止为止。借此在采用单程序段的情况下以不停止的方式执行 NC 程序。

若在**子程序**的第一行（PROC）中通过编程关闭单程序段执行（SBLOF），则其一直生效，直至该 NC 程序结束或终止为止。编写的返回指令将决定是否在该子程序的末尾停止：

- 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

在 NC 程序内关闭单程序段执行

若在 NC 程序内的一个程序段中通过编程关闭单程序段执行（SBLOF），则自此程序段起，直至下一次编写激活单程序段执行（SBLON）或直至生效的子程序级结束为止，关闭单程序段执行。

句法

```
SBLOF
SBLON
```

含义

SBLOF:	用于关闭单程序段执行的预定义步骤	
	在程序段中单独编程：	是，可在 PROC 程序段中
	生效方式：	模态

SBLON:	用于激活单程序段执行的预定义步骤	
	在程序段中单独编程：	是
	生效方式：	模态

10.13.2.2 边界条件

单程序段抑制和程序段显示

可以在子程序中使用 DISPLOF 抑制当前的程序段显示。若一同编写 DISPLOF 和 SBLOF，则会在单程序段停止时在子程序内显示子程序调用。

各单程序段类型的特殊情形

- "SBL2:Rechensatz" UND MD10702 $MN_IGNORE_SINGLEBLOCK_MASK, Bit 12 == 1:
  在 SBLON 程序段中不停止。
- "SBL3:Einzelsatz fein": Der Befehl SBLOF wird unterdrückt

针对异步子程序（ASUB）的单程序段抑制

为在单程序段执行生效的情况下以不停止的方式执行 ASUB，必须在 ASUB 的第一程序行中一同编写 PROC 和 SBLOF。

示例

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

10.13.2.3 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	

程序代码	注释
N40 M100	
N50 R10=90	
N60 **SBLON**	; 激活单程序段
N70 M110	
N80 ...	

在单程序段模式生效时，系统将 N20 与 N60 之间的区域作为一个程序段处理。

示例 2：不停止的子程序

主程序

程序代码
...
N100 G1 X10 G90 F200
N120 X-4 Y6
N130 **CYCLE1**
N140 G1 X0
N150 M30

子程序

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	

即便在单程序段执行生效的情况下，仍完整执行循环。

示例 3：ASUB 包含单程序段抑制且不可见

需要在单程序段执行生效的情况下不中断地执行通过 PLC 用户程序启动的 ASUB。此外该 ASUB 还需要不可见。

程序代码	
N100 PROC NV **SBLOF DISPLOF**	; 单程序段抑制和显示抑制
N110 CASE $P_UIFRNUM OF	
0 GOTOF _G500	
1 GOTOF _G54	
2 GOTOF _G55	

程序代码
3 GOTOF _G56 4 GOTOF _G57 DEFAULT GOTOF END N120 _G54:G54 D=$P_TOOL T=$P_TOOLNO N130 RET N140 _G54:G55 D=$P_TOOL T=$P_TOOLNO N150 RET N160 _G56:G56 D=$P_TOOL T=$P_TOOLNO N170 RET N180 _G57:G57 D=$P_TOOL T=$P_TOOLNO N190 RET N200 END:D=$P_TOOL T=$P_TOOLNO N210 RET

示例 4：子程序中的针对性停止

假设：

- 单程序段执行激活。
- MD10702 $MN_IGNORE_SINGLEBLOCK_MASK, 位 12 = 1

主程序

程序代码	注释
N10 G0 X0	; 单程序段停止
N20 X10	; 单程序段停止
N30 **CYCLE**	; 由循环产生的运行程序段。
N50 G90 X20	; 单程序段停止
M30	

子程序

程序代码	注释
PROC CYCLE **SBLOF**	; 单程序段抑制
N100 R0 = 1	
N110 **SBLON**	; 由于 MD10702 位 12 = 1，不单程序段停止
N120 X1	; 单程序段停止
N140 **SBLOF**	
N150 R0 = 2	
RET	

示例 5：程序嵌套时单段抑制

假设：单程序段执行激活。

```
程序代码                                    注释
N10 X0 F1000                                ; 单程序段停止
N20 UP1(0)
      PROC UP1(INT _NR) SBLOF               ; 为 UP1 关闭单程序段
      N100 X10
      N110 UP2(0)
            PROC UP2(INT _NR)
            N200 X20
            N210 SBLON                      ; 激活单程序段
            N220 X22                        ; 单程序段停止
            N230 UP3(0)
                  PROC UP3(INT _NR)
                  N300 SBLOF                ; 关闭单程序段
                  N305 X30
                  N310 SBLON                ; 激活单程序段
                  N320 X32                  ; 单程序段停止
                  N330 SBLOF                ; 关闭单程序段
                  N340 X34
                  N350 M17                  ; 单程序段停止（M17）
            N240 X24                        ; 单程序段停止（N210）
            N250 M17                        ; 单程序段停止（M17）
      N120 X12
      N130 M17                              ; 单程序段停止（M17）
N30 X0                                      ; 单程序段停止
N40 M30                                     ; 单程序段停止
```

10.13.3 BAG 专用单程序段类型 A / B

在采用 BAG 专用单程序段时，在一个通道（控制通道）中通过单程序段逐段执行 NC 程序。在该控制通道中，单程序段必须通过 NC/PLC 接口信号（DB21 ... DBX0.4）激活。

在 BAG 的其余通道（关联通道）中，根据针对特定 BAG 通过 NC/PLC 接口信号选择的单程序段类型 A 或 B（DB11 DBX1.6 / 7）逐段执行相应的 NC 程序。在该等关联通道中 NC/PLC 接口信号（DB21 ... DBX0.4）不允许置位。

单程序段类型 A / B

- 单程序段类型 A：若控制通道停止，关联通道也立即停止，与 NC 停止相似。
- 单程序段类型 B：若控制通道停止，则关联通道在相应的程序段末尾停止，与程序段交界处 NC 停止相似。

接口信号

控制通道

- DB21 ... DBX0.4（激活单程序段）

BAG 的所有通道

- DB21 ... DBX7.1（NC 启动）

BAG

- DB11 DBX1.6（单程序段类型 B）
- DB11 DBX1.7（单程序段类型 A）

单程序段类型 A 的流程图

前提条件：BAG 的所有通道处于“复位”或“中断”状态下。

1. PLC 用户程序：在控制通道中选择单程序段，DB21 ... DBX0.4 = 1
2. PLC 用户程序：为 BAG 选择单程序段类型 A，DB11 DBX1.7 = 1
3. PLC 用户程序：启动 BAG 的所有通道，DB21 ... DBX0.4 = 1
4. 控制通道在程序段末尾处停止。
5. 所有关联通道获得立即停止执行的内部信号。
6. 在所有关联通道到达制动阶段末尾的情况下，BAG 的所有通道处于“中断”状态。

单程序段类型 B 的流程图

前提条件：BAG 的所有通道处于“复位”或“中断”状态下。

1. PLC 用户程序：在控制通道中选择单程序段，DB21 ... DBX0.4 = 1
2. PLC 用户程序：为 BAG 选择单程序段类型 B，DB11 DBX1.6 = 1
3. PLC 用户程序：启动 BAG 的所有通道，DB21 ... DBX0.4 = 1
4. 控制通道在程序段末尾处停止。
5. 所有关联通道获得在程序段末尾处停止执行的内部信号。
6. 在所有关联通道到达相应的程序段末尾的情况下，BAG 的所有通道处于“中断”状态。

10.13.4 前提条件

10.13.4.1 单程序段类型 SBL2 和逐段生效的同步动作

Bei Einzelsatztyp **"SBL2:Rechensatz"** wird bei einer satzbezogenen Synchronaktion der nächste Stopp erst nach dem nächsten Hauptlaufsatz ausgeführt.该同步动作与下一主处理程序段之间的预处理程序段不会停止。

10.13.4.2 编程的停止（M0），单程序段和单程序段类型切换

初始情况：在通道中，NC 程序因其中编写的 **M0** 停止，且该通道中**激活了单程序段**（DB21, ... DBX0.4 == 1）

在此情形下，若在操作界面上**多次**在**单程序段类型** SBL1 或 SBL3 与 SBL2 之间切换，则会显示报警 16922“超出最大嵌套深度”。

10.14 程序控制

方式

1. 通过操作界面或 PLC 选择功能
2. 激活跳转级
3. 插补缓存的大小调整
4. 启用附加基本程序段显示时的程序显示模式
5. 外部执行（缓存大小和数量）
6. 执行外部子程序

10.14.1 通过操作界面或 PLC 用户程序选择功能

过程

选择

在操作区 “AUTOMATIC” > 水平软键“**程序控制**”下，可从操作界面中选择功能。通过 HMI 设置 HMI/PLC 接口中的请求信号。

激活

必须通过设置相应的请求信号由 HMI 将请求传送到 NC。

用户也可通过该方法直接激活功能。

反馈信息

激活功能后，NC 会向 PLC 发送反馈信息（见表格）。

NC/PLC 接口信号

表格 10-2　程序控制：接口信号

功能	选择信号	激活信号	反馈信号
SKP “跳转程序段 0 至 7” SKP “跳转程序段 8 至 9”	DB21, ... DBX26.0 - 7 DB21, ... DBX27.0 - 1	DB21, ... DBX2.0 - 7 DB21, ... DBX31.6 - 7	---
DRY “空运行进给”	DB21, ... DBX24.6	DB21, ... DBX0.6	DB21, ... DBX318.6
RG0 “快速移动补偿”	DB21, ... DBX25.3	DB21, ... DBX6.6	---
“单程序段”	通过 HMI 的程序控制显示预选 SBL1、SBL2 或 SBL3		
SBL1 “主处理单程序段”	操作面板 HMI	DB21, ... DBX0.4	---
SBL2 “解码单程序段”			
SBL3 “在循环中”			
M01 “编程的停止”	DB21, ... DBX24.5	DB21, ... DBX0.5	DB21, ... DBX32.5
关联 M01	DB21, ... DBX24.4	DB21, ... DBX30.5	DB21, ... DBX318.5
DRF “手轮偏移”	DB21, ... DBX24.3	DB21, ... DBX0.3	DB21, ... DBX33.3
PRT “无轴运动，程序测试”	DB21, ... DBX25.7	DB21, ... DBX1.7	DB21, ... DBX33.7

文档

- 参数手册，NC 变量和接口信号
- HMI Advanced 操作手册之“加工操作区”

10.14.2　激活跳转级

功能

对于不需要在每次程序运行中执行的程序段，可对其设置跳转。在程序段编号前用“/”（斜线）标记要跳转的程序段。

在零件程序中以“/0”至“/9”指定跳转级。

每个零件程序段只能指定一个跳转级。

参数设置

跳转级的数量通过以下机床数据定义：

MD51029 $MM_MAX_SKP_LEVEL（NC 程序中跳转级的最大数量）

编程

对于不需要在每次程序运行时执行的程序段（例如程序试车程序段），可按下图为其设置跳转：

程序代码	注释
/N005	; 程序段跳转，（DB21,...DBX2.0），第 1 跳转级
/0 N005	; 程序段跳转，（DB21,...DBX2.0），第 1 跳转级
/1 N010	; 程序段跳转，（DB21,...DBX2.1），第 2 跳转级
/2 N020	; 程序段跳转，（DB21,...DBX2.2），第 3 跳转级
/3 N030	; 程序段跳转，（DB21,...DBX2.3），第 4 跳转级
/4 N040	; 程序段跳转，（DB21,...DBX2.4），第 5 跳转级
/5 N050	; 程序段跳转，（DB21,...DBX2.5），第 6 跳转级
/6 N060	; 程序段跳转，（DB21,...DBX2.6），第 7 跳转级
/7 N070	; 程序段跳转，（DB21,...DBX2.7），第 8 跳转级
/8 N080	; 程序段跳转，（DB21,...DBX31.6），第 9 跳转级
/9 N090	; 程序段跳转，（DB21,...DBX31.7），第 10 跳转级

激活

10 个跳转级“/0”至“/9”由 PLC 通过置位 PLC → NC 接口信号激活。

在 HMI 中通过“程序控制”菜单在“加工”操作区中激活功能：

- 跳转级“/0”至“/7”：
  通过接口 HMI → PLC DB21, ... DBB26（程序段跳转已选择）。
- 跳转级“/8”至“/9”：
  通过接口 HMI → PLC DB21, ... DBX27.0 至 DBX27.1。

文档:
操作手册

说明

跳转级修改只能在控制系统停止/复位状态下进行。

10.14.3 插补缓存的大小调整

MD28060

在零件程序运行中，通道专用插补器会从插补缓存执行准备的程序段。插补缓存中可同时保存的程序段的最大数量由存储器配置机床数据 MD28060 $MM_IPO_BUFFER_SIZE（插补缓存 DRAM 中 NC 程序段的数量）定义。在某些应用中可选择不占满该缓存，从而保证准备和插补间的较小“间隔”。

SD42990

通过设定数据 SD42990 $SC_MAX_BLOCKS_IN_IPOBUFFER（插补缓存中程序段的最大数量），可将插补缓存中的程序段数量动态限制为小于 MD28060 $MC_MM_IPO_BUFFER_SIZE（插补缓存 DRAM 中 NC 程序段的数量）的值，最低为 2 个程序段。

设定数据 SD42990 $SC_MAX_BLOCKS_IN_IPOBUFFER 的值：

值	作用
< 0	对插补缓存无限制。 对应 MD 28060:MM_IPO_BUFFER_SIZE 激活最大的插补缓存。
或 1	允许的最小插补缓存生效，大小为 2 个程序段。
< MM_IPO_BUFFER_SIZE	插补缓存的激活上限为指定的程序段数量。
>= MM_IPO_BUFFER_SIZE	通过 MD 28060:MM_IPO_BUFFER_SIZE 中指定的程序段数量激活插补缓存。

说明

零件程序中设置了 SD42990 $SC_MAX_BLOCKS_IN_IPOBUFFER 时，解释器在准备中执行包含该 SD 的程序段时插补缓存限制将立即生效。

即插补缓存限制可能会较期望提前数个程序段生效（另见 MD28070 $MC_MM_NUM_BLOCKS_IN_PREP）。

若需避免此情形，使插补缓存限制同步于程序段生效，则须在零件程序中设置 SD 前编写 STOPRE（预处理停止）指令。

有效性

SD42990 $SC_MAX_BLOCK_IN_IPOBUFFER 针对通道全局生效，并可在零件程序中修改。修改过的值会在程序结束时保留。若需在定义的事件下复位该设定数据，则须为此创建一个所谓的事件控制程序。例如可在 RESET 时将该设定数据设置为预定义值。

应用

若需尽可能减少程序段准备和插补间的程序段数量，例如在零件程序中读取和进一步处理实际位置时，可启用插补缓存限制。

示例

```
N10 ...
N20 ...
..........
N100 $SC_MAX_BLOCKS_IN_IPOBUFFER = 5              ; 将插补缓存限制为 5 个 NC 程序段
N110 ...
N120 ...
............
N200 $SC_MAX_BLOCKS_IN_IPOBUFFER = -1             ; 取消插补缓存限制
N210 ...
............
```

10.14.4 启用附加基本程序段显示时的程序显示模式

基本程序段显示（仅适用于 ShopMill/ShopTurn）

除现有的程序段显示外，还可通过另一个基本程序段显示功能来显示可引起**机床上动作**的所有程序段。

基本程序段预显示

实际逼近的终点位置作为绝对位置显示。位置值可选择以工件坐标系（WCS）为基准，或以可设定零点坐标系（SZS）为基准。

显示缓存中预见性保存的显示程序段数量取决于 NC 预处理中对应加工状态下准备的程序段的数量。处理预处理停止时，显示程序段的数量会降至零，并在应答预处理停止后重新提升。在 REORG 事件（例如运行方式切换，ASUB 启动）下，预见性保存的显示程序段会被删除并于之后重新预处理。

预处理的值

在基本程序段显示中预处理的值与以下因素相匹配：

- 选择的刀具
- 进给率和主轴转速
- 实际逼近的位置值
  特例：
  刀具半径补偿生效时可能会存在偏差。
  对于模数轴，基本程序段显示中显示的编程值可能位于模数区域以外。

说明

通常位置以工件坐标系或可设定零点坐标系显示。

基本程序段显示功能可通过以下设定数据激活或取消：

SD42750 $SC_ABSBLOCK_ENABLE（使能基本程序段显示）

10.14.5 ShopMill/ShopTurn 中的基本程序段显示

配置基本程序段显示

基本程序段显示可通过以下机床数据配置：

基本程序段显示相关的 NC 机床数据	含义：
MD28400 $MC_MM_ABSBLOCK	激活基本程序段显示
MD28402 $MC_MM_ABSBLOCK_BUFFER_CONF[2]	显示缓存的大小
显示机床数据	待设置的位置值：
MD9004 $MM_DISPLAY_RESOLUTION	用于公制尺寸设定
MD9011 $MM_DISPLAY_RESOLUTION_INCH	用于英制尺寸设定
MD9010 $MM_SPIND_DISPLAY_RESOLUTION	待设置坐标系的主轴显示精度
MD9424 $MM_MA_COORDINATE_SYSTEM	WCS 或 SZS 的实际值显示

这些显示机床数据会复制到 NC 机床数据
MD17200 $MN_GMMC_INFO_UNIT[0]（全局 HMI 信息）至 MD17200 $MN_GMMC_INFO_UNIT[3] 中。这样便可通过 NC 访问这些显示机床数据。

激活

基本程序段显示通过 MD28400 $MC_MM_ABSBLOCK（激活采用绝对值的程序段显示）及上电激活。若 MD28400 $MC_MM_ABSBLOCK 赋值为 1，则会在主处理中创建一个通道专用显示缓存（FIFO）。

显示缓存（FIFO）的大小 =（MD28060 $MC_MM_IPO_BUFFER_SIZE（插补缓存中 NC 程序段的数量） + MD28070 $MC_MM_NUM_BLOCKS_IN_PREP（用于程序段预处理的程序段数量））乘以 128 字节。采用缺省机床数据设置时，该缓存的大小为 6 KB。

优化显示缓存的大小：
可设定 128 和 512 之间的值来优化存储空间需求。显示缓存中预处理的显示程序段将通过一个可配置的上载缓存传输至 HMI。

上载缓存的大小上限由（MD28402 $MC_MM_ABSBLOCK_BUFFER_CONF[0] + MD28402 $MC_MM_ABSBLOCK_BUFFER_CONF[1] + 1）
与通过 MD28400 $MC_MM_ABSBLOCK 配置的程序段长度相乘得出。

当前程序段**之前**的程序段数量在 MD28402 $MC_MM_ABSBLOCK_BUFFER_CONF[0] 中配置，当前程序段**之后**的程序段数量在 MD28402 $MC_MM_ABSBLOCK_BUFFER_CONF[1] 中配置。

前提条件

在超出 MD28400 $MC_MM_ABSBLOCK 中配置的显示程序段长度时，系统会对该显示程序段进行相应裁切。此时会在程序段末尾显示“...”字符串作为提示。

对于预编译循环
（MD10700 $MN_PREPROCESSING_LEVEL > 1，程序预处理级），显示程序段**只**包含轴位置。

适用于基本程序段显示的更多前提条件：

- 包含绝对值的模态同步动作程序段不在考量范围内。
- 在进行/不进行计算的程序段搜索期间，基本程序段显示取消。
- 极坐标编程不以笛卡尔坐标显示。

半径/直径值

基本程序段显示和位置显示所显示的直径值在内部计算时可能需要作为半径。可通过以下方式按照 G 代码组 29 对用于设定半径/直径的值进行调整：

- G 指令 DIAMCYCOF（通道专用直径编程扩展）
  该 G 指令用于在循环执行期间取消通道专用直径编程。这样在循环中可始终采用半径进行计算。位置显示和基本程序段显示继续对应 DIAMCYCOF 前的直径编程状态。
  基本程序段显示中保留最后显示的值。
- G 指令 DIACYCOFA[AX]（轴专用直径编程）
  该 G 指令用于在循环执行期间针对轴取消直径编程。这样在循环中可始终采用半径进行计算。位置显示和基本程序段显示继续对应 DIACYCOFA[AX] 前的直径编程状态。
  基本程序段显示中保留最后显示的值。
- MD27100 $MC_ABSBLOCK_FUNCTION_MASK（参数设置采用绝对值的程序段显示）

位 0 = 1　　平面轴的设定轴在基本程序段显示中通常作为直径值显示。

压缩器生效时的特性

在压缩器生效且 G 代码组 30 不为 COMPOF 时，系统会生成两个显示程序段。

- 第一个包含生效压缩器的 G 指令。
- 第二个包含“...”字符串，提示缺少显示程序段。

示例：

```
G0 X10 Y10 Z10          ；尚需为基本程序段显示预处理的程序段
COMPCAD                 ；启用压缩器优化表面质量（CAD 编程）
...                     ；字符串提示缺少显示程序段
```

为了避免 NC 性能瓶颈，基本程序段显示被自动取消。此时生成含“...”字符串的显示程序段提示缺少显示程序段。
所有显示程序段均以单程序段生成。

10.14.6 DIN 程序段的结构

DIN 程序段的显示程序段的结构

DIN 程序段的显示程序段的基本结构

- 程序段编号/标签
- 第一个 G 组的 G 指令
  （仅在相对上一个机床功能数据组有变化时）。
- 轴位置
  （顺序对应 MD20070 $MC_AXCONF_MACHAX_USED，通道中生效的机床轴编号）。
- 其他模态生效的 G 指令
  （仅在相对上一个机床功能数据组有变化时）。
- 编写的其他地址。

用于基本程序段显示的显示程序段按照以下规则直接从编写的零件程序段导出：

- 对宏进行扩展。
- 省去跳转标识和注释。
- 程序段编号和标签取自原程序段，DISPLOF 生效时则省去。
- 小数点后位数由显示机床数据 MD9004、MD9010 和 MD9011 通过 HMI 定义。

HMI 显示机床数据	在 NC 机床数据中访问
MD9004 $MM_DISPLAY_RESOLUTION	MD17200 $MN_GMMC_INFO_NO_UNIT[0]
MD9011 $MM_DISPLAY_RESOLUTION_INCH	MD17200 $MN_GMMC_INFO_NO_UNIT[1]
MD9010 $MM_SPIND_DISPLAY_RESOLUTION	MD17200 $MN_GMMC_INFO_NO_UNIT[2]
MD9424 $MM_MA_COORDINATE_SYSTEM	MD17200 $MN_GMMC_INFO_NO_UNIT[3]

- 编写的轴位置作为绝对位置在通过 MD9424 $MM_MA_COORDINATE_SYSTEM（用于实际值显示的坐标系）设定的坐标系（WCS/SZS）中显示。

说明

对于模数轴，模数补偿会被省去。这样一来可能会显示模数区域外的位置，且必定与基本为模数转换的位置显示存在偏差。

示例

显示程序段（原程序段）和基本程序段显示的对比：

- **编写的位置**以绝对值显示。
  地址 AP/RP 以编写的值显示。

原程序段：	显示程序段：
N10 G90 X10.123	N10 X10.123
N20 G91 X1	N20 X11.123

- **地址指定**（非 DIN 地址）以 **<**地址**> = <**常量**>** 的格式显示。

原程序段：	显示程序段：
N110 R1 = -67.5 R2 = 7.5	
N130 Z = R1 RND = R2	N130 Z-67.5 RND = 7.5

- **地址索引**（地址扩展）作为常量显示，**<**地址**> [<**常量**>] = <**常量**>** 。

原程序段：	显示程序段：
N220 DEF AXIS AXIS_VAR = X	
N240 FA[AXIS_VAR] = R2	N240 FA[X] = 1000

- **无地址扩展的 DIN 地址**以 **<**din 地址**> <**常量**>** 的格式显示。

原程序段：	显示程序段：
N410 DEF REAL FEED = 1.5	
N420 F = FEED	N420 F1.5

对于 **H 功能**：与向 PLC 的输出类型
（MD22110 $MC_AUXFU_H_TYPE_INT，H 辅助功能的类型为整数）无关，显示每次编写的值。

- 对于**通过 T 指令进行的刀具选择**
  会生成 T<值> 或 T=<字符串> 格式的显示信息。若先前编写了地址扩展，则其也会被显示。
  若配置了多个主轴，或者“通过刀架换刀”（MD20124 $MC_TOOL_MANAGEMENT_TOOLHOLDER (刀架编号)）功能生效，那么 T 号输出时总是带有地址扩展。
  若之前未编写地址扩展，则会使用主主轴或主刀架的编号（T<主轴编号/刀架> = ）。
- 对于通过 S、M3、M4、M5、M19、M40 - M45 和 M70（或 MD 20094 $MC_SPIND_RIGID_TAPPING_M_NR，用于切换至受控轴模式的 M 功能）进行的**主轴编程**，以下规则适用：
  若先前编写了地址扩展，则其也会被显示。
  若之前配置了多个主轴，那么地址扩展也将一并输出。
  若之前未编写地址扩展，则使用主主轴编号（S<主轴编号>=）。
- G[<组>] = <表述> 格式的**间接 G 指令编程**由对应的 G 指令替代。

原程序段：	显示程序段：
N510 R1=2	
N520 G[8]= R1	N520 G54

- 不生成可执行程序段的**模态生效的 G 指令**会被收集，并在句法允许的前提下通过下一个可执行程序段的显示程序段显示（DIN 程序段）。若非此情形（例如预定义子程序调用 TRANSMIT），系统会将包含变更过 G 指令的独立显示程序段前置。

原程序段：	显示程序段：
N610 G64	G64
N620 TRANSMIT	N620 TRANSMIT

- 对于出现 **F 和 FA** 地址的**零件程序行**，系统总是会生成一个显示程序段（即便在 MD22240 $MC_ AUXFU_F_SYNC_TYPE = 3（F 功能的输出时间）的情形下）。

原程序段：	显示程序段：
N630 F1000	N630 F1000
N640 X100	N640 X100

- **为程序段显示生成的显示程序段直接**从编写的零件程序段推导。若通过轮廓预处理生成了中间程序段（例如刀具半径补偿 G41/G42，半径/倒角 RNDM，RND，CHF，CHR），那么这些程序段从运动所基于的零件程序段获取显示信息。

原程序段：	显示程序段：
N710 Y157.5 G42	N710 Y157.5 G42
N720 Z-67.5 RND=7.5	N720 Z-67.5 RND=7.5

- 使用 **EXECTAB 指令**（执行轮廓元素表）时，显示程序段中会显示通过 EXECTAB 生成的程序段。

原程序段：	显示程序段：
N810 EXECTAB (KTAB[5])	N810 G01 X46.147 Z-25.38

- 使用 **EXECSTRING 指令**时，显示程序段中会显示通过 EXECSTRING 生成的程序段。

原程序段：

```
N910 DEF STRING[40] PROGSTRING = "N905 M3 S1000 G94 Z100 F1000 G55"
N920 EXECSTRING(PROGSTRING)
```

原程序段：

```
N905 Z100 G55 G94 M3 S1000 F1000
```

10.14.7 外部执行

功能

借助“外部执行”功能，可通过一个外部存储器执行由于其大小无法保存在 NC 存储器中的程序。

说明

受保护的循环（_CPF 文件） **无法**通过外部程序存储器执行。

外部程序存储器

外部程序储存器可以位于下列数据载体上：

- 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。

应用

- **直接执行外部程序**
  原则上可通过操作界面的目录结构在 HMI 模式“外部执行”下访问的所有程序均可选择和执行。
- **从零件程序执行外部子程序**
  “外部”子程序通过零件程序指令 EXTCALL 及指定的调用路径（可选）和子程序名称来调用（→ 参见“执行外部子程序（EXTCALL） (页 682)”）。

参数设置

为了在“外部执行”模式下执行程序，动态 NC 存储器中须预留一块缓存（FIFO 缓存）。

FIFO 缓存的大小

FIFO 缓存的大小通过以下机床数据设置：

MD18360 $MN_MM_EXT_PROG_BUFFER_SIZE（用于外部执行的 FIFO 缓存大小）

说明

带跳转语句的程序

对于含有跳转指令(GOTOF，GOTOB，CASE，FOR，LOOP，WHILE，REPEAT，IF，ELSE，ENDIF 等)的外部程序，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。

FIFO 缓存的数量

“外部执行”模式下同时执行的所有程序须分别配备一个 FIFO 缓存。

FIFO 缓存的数量通过以下机床数据设置：

MD18362 $MN_MM_EXT_PROG_NUM（可同时外部执行的程序级的数量）

复位、上电时的特性

复位和上电会使外部程序调用终止，并清除对应的 FIFO 缓存。

通过外部程序存储器选择主程序的情形下，若上电时该程序存储器继续可用，且 MD9106 $MM_SERVER_EXTCALL_PROGRAMS 中激活了通过 EXTCALL 调用加工，则系统会自动重新选择该程序。

10.14.8 执行外部子程序（EXTCALL）

功能

在加工复杂的工件时，某些加工步骤的程序串可能会由于存储空间需求的问题无法直接保存在 NC 存储器中。

在此情形下，用户可通过零件程序指令 EXTCALL 在“外部执行”模式下从外部存储器将程序串作为子程序调用并执行。

前提条件

执行外部子程序须满足以下前提条件：

- 子程序必须可通过操作界面的目录结构访问。
- 针对每个子程序，动态 NC 存储器中均须预留一块缓存（FIFO 缓存）。

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。

可以通过如下指令设置加载存储器的大小：

MD18360 MM_EXT_PROG_BUFFER_SIZE

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。

参数设置

可通过以下设定数据预设到外部子程序目录的路径：

SD42700 $SC_EXT_PROG_PATH（外部子程序调用 EXTCALL 的程序路径）

此路径和编程时指定的子程序路径或名称共同构成待调用程序的整体路径。

编程

通过零件程序指令 EXTCALL 调用一个外部子程序。

句法： EXTCALL("<路径/><程序名称>")

参数：

<路径/>： 绝对或相对路径设定（**可选**）

类型： STRING

<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**）。

类型： STRING

说明

路径设定： 缩写

在指定路径时可使用以下缩写：

- **LOCAL_DRIVE:** 本地驱动
- **CF_CARD:** CF 卡
- **USB:** 前端 USB 接口

CF_CARD: 和 **LOCAL_DRIVE:** 可以互换使用。

采用绝对路径设定的 EXTCALL 调用

如果在给定的路径下存在子程序，则在 EXTCALL 调用后执行子程序。 如果不存在该子程序，则中断程序执行。

采用相对路径设定/无路径设定的 EXTCALL 调用

在进行采用相对路径设定/无路径设定的 EXTCALL 调用时，系统根据下列模式查找存在的程序存储器：

- 如果在 SD42700 $SC_EXT_PROG_PATH 中预设了路径设定，则首先从此路径出发查找 EXTCALL 调用中的设定（程序名或者相对路径设定）。 而绝对路径由字符串组成：
  - SD42700 中预设的路径说明
  - "/" 为分隔符
  - EXTCALL 中指定的子程序路径或者子程序名称
- 如果没有在预设的路径下找到调用的子程序，接下来系统会从用户存储器的目录查找 EXTCALL 调用中的设定。
- 一旦找到子程序，查找结束。 如果没有查找到子程序，则程序中断。

示例

从本地驱动执行

主程序：

程序代码

```
N010 PROC MAIN
N020 ...
N030 EXTCALL ("SCHRUPPEN")
N040 ...
N050 M30
```

外部子程序：

程序代码

```
N010 PROC SCHRUPPEN
N020 G1 F1000
N030 X= ... Y= ... Z= ...
N040 ...
...
...
N999999 M17
```

主程序“Main.mpf”位于 NC 存储器中，并已被选中用于执行。

需要下载的子程序“SCHRUPPEN.SPF”或“SCHRUPPEN.MPF”位于本地驱动器的目录“/user/sinumerik/data/prog/WKS.DIR/WST1.WPD”下。

在 SD42700 中，子程序路径的预设为：

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

说明

未在 SD42700 中设定路径时，必须为此示例编程以下 EXTCALL 指令：

```
EXTCALL("LOCAL_DRIVE:WKS.DIR/WST1.WPD/SCHRUPPEN")
```

10.15 从外部存储器执行（选件）

10.15.1 功能

说明

使用该功能必须具有需要购买授权的选件“CNC 用户存储器扩展”或“从外部存储器执行 (EES)”！

功能

EES（Execution from External Storage，从外部存储器执行）功能可协助用户通过 NC **直接从一个外部存储器**执行程序。

以下驱动器可用作外部存储器：

驱动器	符号名称 1)	SINUMERIK 840D sl	SINUMERIK 828D
NC Extend（从前的“本地驱动器”）	CF_CARD LOCAL_DRIVE SYS_DRIVE	NCU 的 CF 卡 或者 PCU 的本地硬盘	PPU 的系统 CF 卡
用户 CF 卡	CF_CARD LOCAL_DRIVE	-	PPU 的用户 CF 卡
网络驱动器			
静态引导 USB 驱动器			

1) 在 SINUMERIK 840D sl 上，驱动器 NC Extend 被固定分配有符号名称 LOCAL_DRIVE、CF_CARD 和 SYS_DRIVE（⇒ NC Extend 可通过 LOCAL_DRIVE、CF_CARD 和 SYS_DRIVE 进行寻址）。
在 SINUMERIK 828D 上，可配置 NC Extend 的符号名称 LOCAL_DRIVE、CF_CARD 和 SYS_DRIVE 的分配。如此一来，在没有用户 CF 卡时，PPU 的系统 CF 卡也能通过符号名称 LOCAL_DRIVE 和 CF_CARD 进行寻址。

注意
USB 闪存可损坏刀具/工件 不建议使用 USB 闪存来执行外部程序。如在执行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，会导致加工意外停止。这可能会损坏刀具或/和工件。

前提条件

使用 EES 需要满足下列前提条件：

- 必须具有需要购买授权的选件“CNC 用户存储器扩展”或“从外部存储器执行（EES）”。
- 需要在控制系统上用作外部存储器的驱动器必须配置成逻辑驱动器（参见“调试 (页 689)”）。

工作模式

根据选件和驱动器配置，可采用不同的 EES 工作模式。控制系统的生效的工作模式通过机床数据 MD18045 $MN_EES_MODE_INFO 显示：

MD18045	工作模式	选件	外部存储器	
			SINUMERIK 840D sl	SINUMERIK 828D
= 0	EES 未生效	-	-	
= 1	本地 EES 生效	6FC5800-0AP77-0YB0 CNC 用户存储器扩展	NCU 上 EES 的使用被限制于 CF 卡的扩展用户存储器（100 MB）。 使用 PPU 上的 EES 时，可以使用 NC Extend 中总的空白存储器。	EES 的使用被限制于系统 CF 卡的扩展用户存储器（100 MB）。
			EES 不适用于网络或 USB。	
		6FC5800-0AP12-0YB0 NCU 的 CF 卡上附加的 HMI 用户存储器	使用 NCU 上的 EES 时，可通过附加选件将本地用户存储器扩展到最大 6 GB（取决于 MD9111）。 使用 PCU 上的 EES 时，不需要附加选件。	-
= 2	全局 EES 生效	6FC5800-0AP75-0YB0 从外部存储器执行（EES）	EES 适用于所有可用外部存储器。	EES 适用于所有可用外部存储器。 **提示：** 通过网络的 EES 额外需要使能 HMI 功能“网络驱动器管理”（MD19730 位 2 = 1）。
		6FC5800-0AP12-0YB0 NCU 的 CF 卡上附加的 HMI 用户存储器	使用 NCU 上的 EES 时，可通过附加选件将本地用户存储器扩展到最大 6 GB（取决于 MD9111）。 使用 PCU 上的 EES 时，不需要附加选件。	-

MD18045	工作模式	选件	外部存储器	
			SINUMERIK 840D sl	SINUMERIK 828D
= 5	本地 EES 生效 + 全局零件程序存储器	如 MD18045 = 1，区别仅在于在扩展用户存储器上增设了全局零件程序存储器 (页 690)。		
= 6	全局 EES 生效 + 全局零件程序存储器	如 MD18045 = 2，区别仅在于在外部存储器上增设了全局零件程序存储器 (页 690)。		

属性

EES 功能能够取代“外部执行 (页 680)”和“执行外部子程序（EXTCALL） (页 682)”功能。

EES 功能具有以下优点：

- 程序处理在系统内一致
- 可使用指令方面无限制
  使用“外部执行”和“执行外部子程序（EXTCALL）”时的限制，例如无反向跳转、跳转指令的跳转范围受缓存制约等，在 EES 功能中被取消。
- 可在各种程序存储器（NC、GDIR、外部驱动器）之间移动程序。
- 实际应用中对零件程序大小和程序数量无限制（其只受外部数据存储器容量制约）。
- 子程序调用的句法一致，与子程序的存储路径无关（不需要 EXTCALL 调用）。
- 网络驱动器可由多个工作站（PCU/NCU）共用。前提条件是这些工作站采用一致的驱动器配置。以获取统一程序视图。
- 所有工作站通过一致方式查看外部程序存储器，因此对外部存储器中保存的程序的修改持续对所有工作站生效。

10.15.2 调试

10.15.2.1 驱动器的配置

在使用 EES 功能前，必须确定控制系统上使用的驱动器。

文档：

- 调试手册之基本软件和操作软件
- 操作手册，通用版/车削版/铣削版/磨削版

激活新的驱动器配置后，可将程序自由分配给可用驱动器。

说明

在新建的驱动器配置中，先前的驱动器可能并非均可用。因此，无法再访问驱动器中的程序。

解决方法：Programme dieser Ablagen vorher auf ein weiterhin zugreifbares Laufwerk kopieren.

说明

受系统限制，受保护循环（CPF 文件）只能从 NC 零件程序存储器执行，因此**无法**保存用于外部存储器上的执行。

目前为止使用的、建有 MPF.DIR、SPF.DIR 和 WKS.DIR 目录的 NC 零件程序存储器对 EES 功能非强制必需。也可配置不采用 NC 零件程序存储器的系统。

注意

执行不可见程序

即便将 NC 零件程序存储器从驱动器配置中删除，其仍始终存在于系统内部。这特别是意味着，在执行程序时，该存储器的 SPF.DIR 目录下尚存在的程序也可能会被意外执行。

解决方法：Wird das System bei der Laufwerksprojektierung ohne NC-Teileprogrammspeicher projektiert, sollten alle dort eventuell noch abgelegten Programme manuell gelöscht werden.

如需继续使用 NC 零件程序存储器，建议不要将其从系统完全删除，必要时可为其启用较高的保护级。

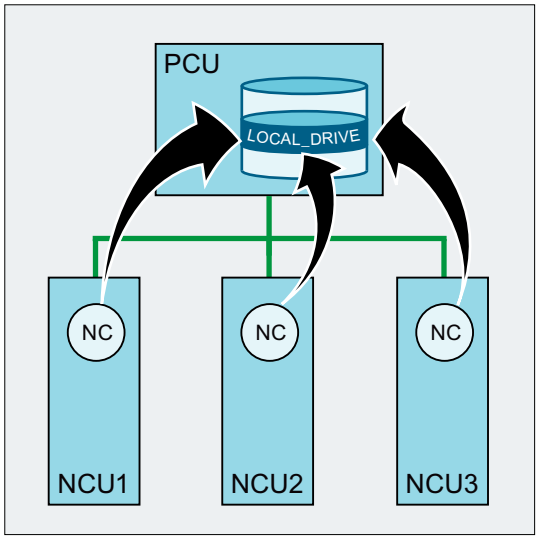
驱动器可由多个工作站（PCU/NCU）共用。为这些工作站启用一致的驱动器配置时，便可获取不受工作站制约的统一程序视图。

说明

NCU/PPU 的 CF 卡不可由多个工作站共享。

示例：

多个 NCU 共用一个位于 PCU 的本地硬盘（LOCAL_DRIVE）上的程序存储器。



说明

在 EES 运行中，若多个工作站共用外部程序存储器，则须遵循以下规则：

- 不可同时通过多个工作站编辑一个程序。
- 不可修改执行中的程序。

仅针对 SINUMERIK 840D sl

通过外部 HMI 运行时，必须启用外部 HMI 上的驱动器配置！必须从外部 HMI 将驱动器配置（logdrive.ini）载入至对应 NCU。驱动器配置对话框中有传输软键。

对于多 NC 协作的设备，所有 NC 的驱动器配置须相同。这通过将 logdrive.ini 文件分配给 mmc.ini 中列出的所有 NCU 来实现。这样便将该处的现有配置覆盖。

10.15.2.2 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）。

文档：
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（GDIR 的标配目录）。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。

对于不通过给定绝对路径调用的子程序，GDIR 将其的搜索路径扩展。

GDIR 取代 NC 零件程序存储器

当 NC 零件程序存储器的 MPF.DIR、SPF.DIR 和 WKS.DIR 目录完全为空时，GDIR 会取代 NC 零件程序存储器。目前的 NC 搜索路径通过 GDIR 1:1 映射。

在外部存储器上选择主程序：

子程序的查找顺序：

1. 外部存储器上的当前目录
2. GDIR 存储器中的 SPF.DIR
3. 通过 CALLPATH 指向的驱动器
4. 循环

GDIR 作为对 NC 零件程序存储器的扩展

若 NC 零件程序存储器的 MPF.DIR、SPF.DIR 和 WKS.DIR 目录中不为空，那么子程序的搜索序列取决于主程序的存储路径（生效的目录）。

在 NC 零件程序存储器中选择主程序（MPF.DIR 或 WKS.DIR 中的 xxx.WPD）

子程序的查找顺序：

1. NC 零件程序存储器中的当前目录
2. NC 零件程序存储器中的 SPF.DIR
3. 通过 CALLPATH 指向的驱动器
4. 循环

在 EES 共享外部存储器上选择主程序

子程序的查找顺序：

1. 外部存储器上的当前目录
2. NC 零件程序存储器中的 SPF.DIR
3. GDIR 中的 SPF.DIR
4. 通过 CALLPATH 指向的驱动器
5. 循环

说明

查找顺序的确定另见 MD11625 $MN_FILE_ONLY_WITH_EXTENSION 和 MD11626 $MN_CYCLES_ONLY_IN_CYCDIR！

说明

通过 CALLPATH 指令也可指向外部驱动器。

10.15.2.3 EES 功能下零件程序中的文件处理设置

设备内唯一的程序名称

在 EES 运行中，若多个工作站共用外部程序存储器，那么在各工作站上同时执行文件操作（WRITE、DELETE、…）可能会导致访问冲突。为了避免此类访问冲突，建议在每个工作站上为文件名称设置一个设备内唯一的命名空间。

设备内唯一的命名空间

文件名称的设备内唯一命名空间例如可如下实现：将文件名称与机床数据中可参数设置的 EES 专用 NC 名称，以及执行程序的通道的编号相连。例如，在执行程序时，以下程序会将 EES 专用的 NC 名称（NC1）和通道编号（通道 1）附加至程序名称，从而生成一个设备内唯一的文件名（MYFILE _NC1_1.SPF）。

```
$MN_EES_NC_NAME="NC1"
N10 DEF STRING[31] FILENAME
N20 FILENAME="MYFILE_" << $MN_EES_NC_NAME << "_" << $P_CHANNO <<
".SPF"
```

参数设置

NC 的 EES 专用名称通过以下 NC 专用机床数据设置：

MD10125 $MN_EES_NC_NAME = <NC 名称>

说明

设备内唯一的 NC 名称

为了避免访问冲突，NC 的 EES 专用名称须为设备内唯一。相关责任由机床制造商/用户自行担负。

调用程序时仅搜索有文件标识的文件

为了在调用子程序时加快程序搜索，建议只搜索**带有**文件标识（例如：SPF、MPF 等）的文件：

MD11625 $MN_FILE_ONLY_WITH_EXTENSION = 1

说明

在通过 EXTCALL 执行外部子程序时，MD11625 对程序搜索没有影响。

文档：

对子程序调用搜索路径的说明参见：编程手册之工作准备分册。

仅在循环目录下搜索含接口的程序

为了在调用子程序时加快程序搜索，建议在搜索创建了接口描述（借助 PROC 指令）、且从循环目录（CUS、CMA、CST）生成接口描述的子程序时，可将搜索范围限制为循环目录：

MD11626 $MN_CYCLES_ONLY_IN_CYCDIR = 1

说明

MD11626 对通过 EXTERN 说明创建接口的子程序无影响。对此类子程序的搜索会在所有程序目录下执行。

注意
搜索不到循环目录外的循环 通过设置 MD11626 = 1 无法再找到当前目录和本地子程序目录中的循环！ 解决方法：将循环始终保存在循环目录中。

10.15.2.4 存储器配置

缩小被动文件系统中的最终用户程序存储器容量

EES 激活时，可缩小被动文件系统中的最终用户程序存储器容量：

MD18352 $MN_MM_U_FILE_MEM_SIZE（用于零件程序/循环/文件的最终用户存储器容量）

未占用的存储器容量可用于刀具数据或制造商循环 (MD18353 $MN_MM_M_FILE_MEM_SIZE)。

文档：
有关存储器配置的详细信息参见功能手册之扩展功能

共享缓存

EES 功能可代替“外部执行”和“执行外部子程序 (EXTCALL)”功能。

如通过 EXTCALL 调用（而非“执行外部子程序 (EXTCALL)”功能）执行零件程序中的子程序，必须将 EXTCALL 调用转换为 CALL 调用并修改路径信息。

转换完成后可共享“外部执行”和“执行外部子程序 (EXTCALL)”所需的缓存（FIFO 缓存）。

MD18362 $MN_MM_EXT_PROG_NUM（可同时外部执行的程序级的数量）= 0

10.15.3 边界条件

示教

在 EES 运行中，不可在 AUTO 运行模式下使用“示教”功能。

10.16 Process DataShare - 数据输出到外部设备/文件上

10.16.1 功能

通过“Process DataShare”功能，可以把数据从零件程序输出到外部设备/外部文件中，以便如记录生产数据，调控操作系统上的附加装置等。

可用性

该功能可用在：

- 只在真实的 NC 中（不是在模拟软件 SNC 和 VNC 中）。
- 仅限于零件程序（不用于同步动作）。
- 并行用于 NC 的所有处理通道，用于所有可用的输出设备。

外部设备/文件

外部设备/文件可以是：

- 本地 CF 卡上的文件
  “本地 CF 卡”是 HMI 上的标识符“LOCAL_DRIVE”对应的存储器。在 SINUMERIK 840D sl 上，该存储器就是本地驱动；在 SINUMERIK 828D 就是用户 CF 卡。

  说明

  在 SINUMERIK 840D sl 上向“LOCAL_DRIVE”输出数据时，需要使用选件“ NCU CF 卡上的附加 xxx MB HMI 用户储存器”。在 SINUMERIK 828D 上只需要使用用户 CF 卡，无需其他选件。

- 网络驱动上的文件
- V.24 接口

  说明

  在 SINUMERIK 840D sl 上向 V.24 接口输出数据时，需要使用 NCU 选件“RS232 接口”。而在 SINUMERIK 828D 上，可直接将数据输出到集成端口 V.24 （前提条件：MD51233 $MNS_ENABLE_GSM_MODEM = 0）。

打开的外部设备的最大数量

在零件程序/通道中，也可以有多个外部设备/文件。除了所有 NC 通道外，可以同时最多打开 10 个输出设备。另外，还有两个备用的条目，用于西门子循环。

这些输出设备最多可以同时执行 5 个任务。

使用模式

在打开每个输出设备时都可以指定，该设备是仅被一个通道占用，还是被多个通道共同使用（即“分享”模式）。

零件程序结束和通道复位时的特性

程序段结束和通道的复位会关闭所有在该通道中打开的外部设备/文件。

使用数据传输（至控制系统）功能

注意
数据安全 在使用 Process DataShare 功能将数据从一个外部设备通过以太网 - 接口 X130 传输到控制系统时，可能会混入第三方数据，导致数据不一致。因此，在使用该功能时须确保网络能够阻止第三方访问。

10.16.2 调试

在文件“/oem/sinumerik/nck/extdev.in”或“/user/sinumerik/nck/extdev.ini.”中可以配置需要使用的外部设备。如果这两个文件同时存在，后一个文件中的条目优先。可以在“调试”操作区的“系统数据/CF 卡”中编辑该文件。

说明

使用“LOCAL_DRIVE”和“CYC_DRIVE”时，无需在文件“extdev.ini”中开展配置。只要有用户 CF 卡，并且设置了对应的选件，就一直可以使用这两个设备。

在文件“extdev.ini”的段落[ExternalDevices]中，可以定义/计数需要使用的外部设备。一个串行设备(/dev/v24)和最多九个文件或目录(/dev/ext/1…9)可以指定为“设备”。输出的记法采用的是 Linux 记数法。以“；”开始的行都是注释，可以忽略。

以“/”符号结尾的设备是目录路径，以文件名称结尾的（无“/”）设备是文件路径，也就是全验证路径，但 /dev/v24 是特例。对于“目录路径”型设备，在零件程序中还必须一同输入文件名称（路径）。

设备由三个用逗号隔开的信息定义：“服务器”、“路径”和可选的“写入模式”，但 /dev/v24 是特例。

可以对文件或目录进行设置：在打开文件后，输出数据是覆盖文件（"O" = 覆盖）还是添加到文件中 ("A" = 添加）。对目录进行的设置会作用于目录下的所有文件。缺省值是“A”。在打开文件/目录时，请新建缺少的文件/目录。

设备“V.24 接口”请按照传输率、数据位、停止位、奇偶位、协议和结束位的顺序依次设置。

在需要将数据输出/保存到“LOCAL_DRIVE”上时，请通过“LOCAL_DRIVE_MAX_FILESIZE ”设置一个最大文件大小，单位是字节，该设置对所有文件统一生效。在“添加模式”中执行 EXTOPEN 指令时，会检测文件的大小。也可以用“LOCAL_DRIVE_FILE_MODE”来确定写入模式（O= 覆盖，A= 添加）。缺省值是“A”。

说明

在目录“/siemens/sinumerik/nck”中有一份配置文件“extdev.ini”的备份。

说明

只有在重新启动 NC 后，文件“extdev.ini”上的修改才生效。

说明

USB 设备

在 SINUMERIK 828D 上，也可以将前端 USB 设备（没有分区信息!）定义为数据输出目标设备。USB 设备也只能从零件程序中通过设备标识符 “/dev/ext/x” 间接调用。

在 SINUMERIK 840D sl 上，只能把 TCU 上稳态连接的 USB 接口定义为 USB 设备。它通过“SERVER:/PATH ”配置，其中“SERVER”是 TCU 的名称，“/PATH”是 USB 接口。TCU 上的 USB 接口名称为“dev0-0”、“dev0-1”、“dev1-0”。路径由“/Partition”引导，其中，分区由两位的分区号或分区名称指定，有时还补充有文件路径，例如：

/dev/ext/8 = "TCU4:/dev0-0, /01/, A"

/dev/ext/8 = "TCU4:/dev0-0, /01/mydir.dir/"

/dev/ext/8 = "TCU4:/dev0-0, /myfirstpartition/Mydir.dir/myfile.txt, O"

示例

```
[ExternalDevices]
；注释行
; V24 示例
; /dev/v24 = "9600, 8, 1, none, rts [, etx]"
； 网络驱动示例
; /dev/ext/1 = "//[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE/, /, A"
; /dev/ext/2 = "//[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE, /myfile.txt, O"
; /dev/ext/3 = "//[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE, /mydir/, A"
```

; /dev/ext/4 = "SERVER:/dev0-0, /01/, A"

; …

；SINUMERIK 828 only （USB）

; /dev/ext/9 = "usb, / [, O]"

；缺省：分区号 = 1

；仅限西门子

; /dev/cyc/1= "//[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE, /mydir/, A"

; /dev/cyc/2= "//[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE/mydir, /, A"

LOCAL_DRIVE_MAX_FILESIZE = 50000

LOCAL_DRIVE_FILE_MODE = “O”

EXTOPEN 参数的有效性 <写入模式>

由于写入模式既可以在文件 “extdev.ini” 中定义，又可以在 EXTOPEN 指令中定义，因此可能会出现设置冲突，导致 EXTOPEN 指令报错。

“extdev.ini” 文件中的值	EXTOPEN 参数值		
	“OVR”	“APP”	-
“O”	O	错误	O
“A”	错误	A	A
-	O	A	A
	说明： O：“覆盖”模式有效。 A:“添加”模式有效。 错误：EXTOPEN 指令报错。		

LOCAL_DRIVE：文件属性

通过 EXTOPEN 在 “LOCAL_DRIVE” 上打开的文件具有以下属性：

- 所有人： "user" 设置了读写权限
- 组群： "operator" 设置了读写权限

10.16.3 编程

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
…
```

10.16.4 边界条件

对连续路径运行的影响

EXTOPEN、WRITE 和 EXTCLOSE 每次都会触发预处理停止，从而中断连续路径运行。

程序段搜索动作

在“带计算的程序段搜索期间”，WRITE 指令不会输入任何结果。但是，在找到搜索目标，并按下“NC 启动”后，EXTOPEN 和 EXTCLOSE 会变为有效。后面的 WRITE 指令因此处于正常的程序环境下。

在“程序测试”模式下进行“带计算的程序段搜索”时，EXTOPEN、WRITE 和 EXTCLOSE 按照正常的程序处理过程执行。

10.17 启动、复位/零件程序结束和零件程序开始的系统设置

方案

可通过机床数据针对以下事件设置控制系统特性：

- 启动（上电）
- 复位/零件程序结束
- 零件程序开始

下列情形后的控制系统特性：	可通过哪些机床数据设置：
启动（上电）*)	MD20110 $MC_RESET_MODE_MASK MD20144 $MC_TRAFO_MODE_MASK MD20150 $MC_GCODE_RESET_VALUES
复位/零件程序结束	MD20110 $MC_RESET_MODE_MASK MD20150 $MC_GCODE_RESET_VALUES MD20152 $MC_GCODE_RESET_MODE
零件程序开始	MD20112 $MC_START_MODE_MASK MD20110 $MC_RESET_MODE_MASK
*) 另见 上电 (页 898)	

启动后的系统设置

MD20110 $MC_RESET_MODE_MASK，位 0 = 0 或 1

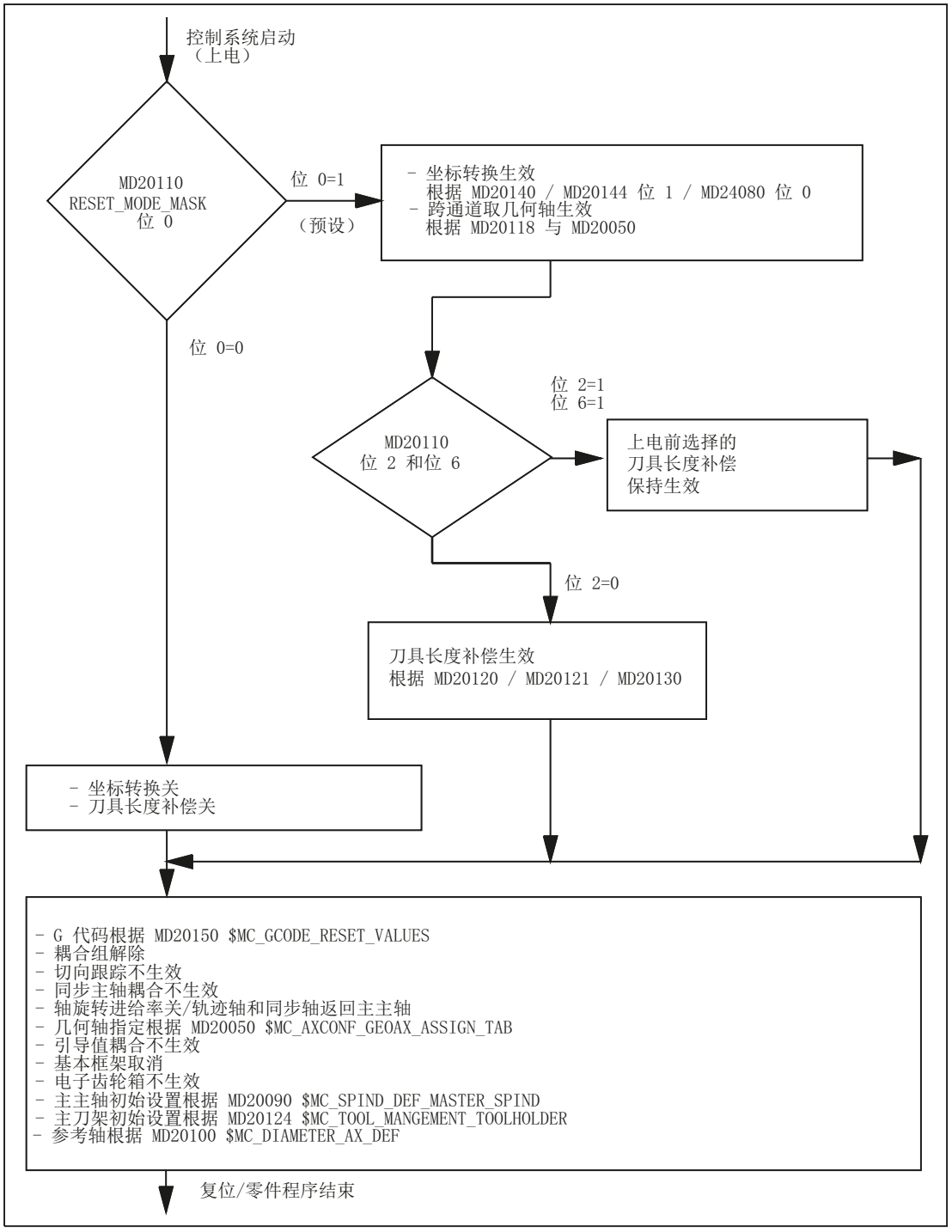


图 10-8　　启动（上电）后的系统设置

复位/零件程序结束和零件程序开始后的系统设置

MD20110 $MC_RESET_MODE_MASK，位 0 = 0 或 1

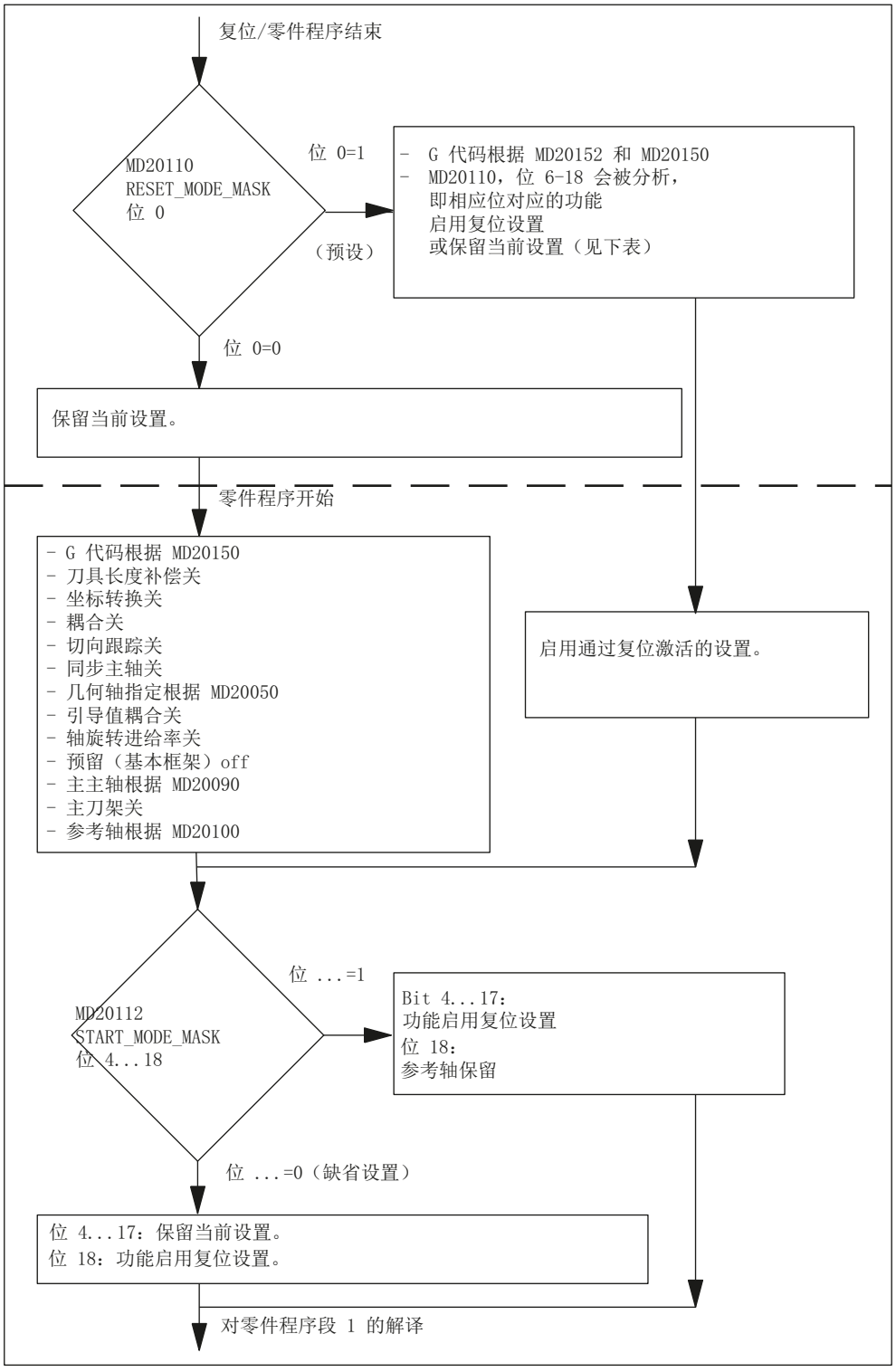


图 10-9　　复位/零件程序结束和零件程序开始后的系统设置

启动和复位/零件程序结束后生效的 G 指令

启动（上电）和复位/零件程序结束后每个 G 功能组中生效的 G 代码通过以下机床数据设置：

MD20150 $MC_GCODE_RESET_VALUES[**<G 组>**] = **<**缺省 G 代码**>**

MD20152 $MC_GCODE_RESET_MODE[**<G 组>**] = **<**值**>**

值	说明：针对每个 G 功能组
0	MD20150 $MC_GCODE_RESET_VALUES 中的缺省 G 指令生效。
1	最后生效的/当前 G 指令生效

启动、复位/零件程序结束和零件程序开始后的系统初始设置

通过以下机床数据定义启动（上电）、复位/零件程序结束和零件程序开始后的系统初始设置

- MD20110 $MC_RESET_MODE_MASK（定义**启动**和**复位/零件程序结束**后的控制系统初始设置）
- MD20112 $MC_START_MODE_MASK（定义**零件程序开始**后的控制系统初始设置）

文档

详细机床数据说明

机床数据关联

机床数据	含义
MD20120 $MC_TOOL_RESET_VALUE	启动、复位/零件程序结束时的刀具长度补偿
MD20121 $MC_TOOL_PRESEL_RESET_VALUE	复位时的预选刀具
MD20130 $MC_CUTTING_EDGE_RESET_VALUE	启动时的刀沿长度补偿
MD20140 $MC_TRAFO_RESET_VALUE	启动转换数据组
MD20144 $MC_TRAFO_MODE_MASK	坐标转换功能的选择
MD20150 $MC_GCODE_RESET_VALUES	G 功能组的初始设置
MD20152 $MC_GCODE_RESET_MODE	G 功能组的复位特性
MD21330 $MC_COUPLE_RESET_MODE_1	耦合撤销特性

机床数据	含义
MD20050 $MC_AXCONF_GEOAX_ASSIGN_TAB	几何轴指定为通道轴
MD20118 $MC_GEOAX_CHANGE_RESET	允许自动几何轴切换

示例

复位时激活复位设置：

- MD20110，位 0 = 1
- MD20112 = 0

复位/零件程序启动时保留转换：

- MD20110，位 0 = 1
- MD20110，位 7 = 1
- MD20112 = 0

复位/零件程序启动后保留刀具长度补偿：

- MD20110，位 4 = 1
- MD20110，位 6 = 1
- MD20112 = 0

复位后保留生效的级（位 4）和可设置的框架（位 5），零件程序开始时复位：

- MD20110，位 4 = 1
- MD20110，位 5 = 1
- MD20112，位 4 = 1
- MD20112，位 5 = 1

说明

MD20110 / MD20112，位 5 和位 6

在 AUTO 或 MDI 运行方式下启动零件程序时，若 MD20110 / MD20112 的参数设置使得刀具长度补偿或框架生效，第一个轴编程则须采用绝对尺寸，用于运行偏移。

例外：启用 MD42442 / MD42440 时 G91 下的偏移运行被抑制。

10.17.1 上电后通过定向转换回退刀具

功能

若包含刀具定向加工的零件程序由于掉电或复位而终止，那么在控制系统启动（上电）后可重新选择之前生效的转换并沿刀具轴的方向生成一个框架。 之后可在 JOG 运行方式下沿刀具轴的方向回退刀具。

前提条件

对于参与转换的所有机床轴，生效的测量系统必须采用一个机床参考。参见“自动恢复机床参考 (页 1392)”章节。

参数设置

为了使最后生效的转换在上电后保留，必须设置以下机床数据：

- MD20144 $MC_TRAFO_MODE_MASK，位 1 = 1
- MD20110 $MC_RESET_MODE_MASK，位 0 = 1
- MD20110 $MC_RESET_MODE_MASK，位 7 = 1

另见章节“启动、复位/零件程序结束和零件程序开始的系统设置 (页 704)”。

编程

等待机床参考 WAITENC

通过 WAITENC 指令在程序中针对通道进行等待，直至参数设置的轴的所有测量系统均为同一有效的机床参考。 参见上文的“前提条件”。 轴的参数设置如下：

MD34800 $MA_WAIT_ENC_VALID = 1

应用

在启动（前提条件：MD20108 位 3 = 1）时待调用的事件控制用户程序（.../_N_CMA_DIR/_N_PROG_EVENT_SPF）中，必须通过 WAITENC 指令等待，直至有效的轴位置可用。

之后可通过 NC 语言指令 TOROTX/TOROTY/TOROTZ 生成一个框架，将刀具轴置于 X 轴、Y 轴或 Z 轴方向。

示例

定向转换和采用增量编码器的定向轴。

配置：	含义：
MD10720 $MN_OPERATING_MODE_DEFAULT [0] = 6	在 JOG 运行方式下启动
MD30240 $MA_ENC_TYPE [0, <轴>] = 1	增量测量系统
MD34210 $MA_ENC_REFP_STATE [0, <轴>] = 3	对增量编码器使能轴位置恢复。
MD20108 $MC_PROG_EVENT_MASK = 'H9'	在启动和零件程序开始时激活事件控制的用户程序（PROG_EVENT）。
MD20152 $MC_GCODE_RESET_MODE [52] = 1	复位后保留 TOFRAME。
MD20110 $MC_RESET_MODE_MASK = 'HC1'	复位后保留转换和刀具补偿。
MD20144 $MC_TRAFO_MODE_MASK = 'H02'	下电后保留转换。

事件控制的用户程序（.../_N_CMA_DIR/_N_PROG_EVENT_SPF）

程序代码	注释
; 示例：启动和通过零件程序开始回退时，激活框架沿刀具方向对准 WCS。	
IF $P_PROG_EVENT == 4	; 启动
IF $P_TRAFO <> 0	; 已选择坐标转换。
WAITENC	; 等待有效的定向轴位置。
TOROTZ	; 将 WCS 的 Z 轴转到刀具轴方向。
ENDIF	
M17	
ENDIF	
IF $P_PROG_EVENT == 1	; 零件程序开始。
TOROTOF	; 复位刀具框架。
RET	
ENDIF	

WAITENC 指令基本对应以下程序串（采用 AB 运动系统的 5 轴机床）：

程序代码	注释
WHILE TRUE	; 等待测量系统。
IF (($AA_ENC_ACTIVE[X]==TRUE) AND ($AA_ENC_ACTIVE[Y]==TRUE) AND ($AA_ENC_ACTIVE[Z]==TRUE) AND ($AA_ENC_ACTIVE[A]==TRUE) AND ($AA_ENC_ACTIVE[B]==TRUE)) GOTOF GET_LABEL	
ENDIF	
G4 F0.5	; 等待时间 0.5s
ENDWHILE	
; 同步位置	
GET_LABEL: GET(X,Y,Z,A,B,)	

继续程序执行

AUTO 运行方式

为了在 AUTO 运行方式下自动执行程序，必须对所有生效测量系统实际位置恢复的机床轴进行回参考点。

MDI 运行方式和溢出转存

在 MDI 运行方式和溢出转存中，也可采用恢复后的位置进行加工，而无需执行轴回参考点。为此，必须针对特定通道明确使能系统恢复后的位置启动：

MD20700 $MC_REFP_NC_START_LOCK = 2

前提条件

采用增量编码器和无实际值缓存的轴

假设采用增量编码器和**无**实际值缓存的轴在掉电时能够足够快地夹紧，以防止最后设定位置偏移。

10.18 由子程序替换功能

10.18.1 概述

功能

用户专用辅助功能（例如 M101）不会触发系统功能。其只会被输出至 NC/PLC 接口。辅助功能的功能性须由用户/机床制造商在 PLC 用户程序中实现。 下面将介绍配置用户专用子程序（替换子程序）调用的方法，来取代缺省设置下的“输出至 NC/PLC 接口”特性。

零件程序中仍编写 M101 功能。 但是在执行零件程序时则会调用替换子程序。 这样一来，该功能便由 NC 替换为子程序调用。 这会带来以下优势：

- 在调整生产流程时，现有的、经过验证的可靠零件程序可不经修改继续使用。 需要进行的调整转移到用户专用子程序内部进行。
- 在完整的 NC 语言功能的支持下，替换子程序能内能实现优越的功能性。
- NC 和 PLC 间的通讯开销得以省去。

可替换的功能

下列功能可通过子程序替换：

辅助功能	
M	开关功能
T	刀具选择
TCA	和刀具状态无关的刀具选择
D	刀具补偿
DL	附加刀具补偿

同步主轴耦合生效时的主轴相关功能	
M40	自动齿轮档切换
M41 - M45	齿轮档选择 1 ... 5
SPOS	主轴定位
SPOSA	主轴定位
M19	主轴定位

10.18.2 替换 M、T/TCA 和 D/DL 功能

10.18.2.1 替换 M 功能

简介

下列条件适用于 M 功能替换：

- 每个程序段只替换一个 M 功能。
- 需要替换 M 功能的程序段**不允许**包含以下元素：
  - M98
  - 模态子程序调用
  - 子程序返回
  - 零件程序结束
- 会触发系统功能的 M 功能不允许通过子程序替换（参见“不可替换的 M 功能”部分）。

参数设置

M 功能和子程序

M 功能和替换子程序通过以下机床数据进行参数设置：

- MD10715 $MC_M_NO_FCT_CYCLE[**<索引>**] = **<**M 功能编号**>**
- MD10716 $MC_M_NO_FCT_CYCLE_NAME[**<索引>**] = "**<**子程序名称**>**"

M 功能和对应的替换子程序通过相同的索引相关联。

示例： M 功能 M101 通过子程序 SUB_M101 替代，M 功能 M102 通过 SUB_M102 替代：

MD10715 $MC_M_NO_FCT_CYCLE[**0**] = 101
MD10716 $MC_M_NO_FCT_CYCLE_NAME[**0**] = "SUB_M101"

MD10715 $MC_M_NO_FCT_CYCLE[**1**] = 102
MD10716 $MC_M_NO_FCT_CYCLE_NAME[**1**] = "SUB_M102"

用于信息传输的系统变量

对于可自由选择的 M 功能，可通过系统变量（参见“系统变量 (页 718)”章节）获取被替换的 M 功能及其他功能（T、TCA、D、DL）的相关信息，用于替换子程序中的分析。 系统变量中包含的数据基于编写了待替换 M 功能的程序段。

M 功能通过机床数据 MD10715 $MC_M_NO_FCT_CYCLE[**<索引>**] 的索引选择，待替换的 M 功能在该机床数据中进行参数设置。

MD10718 $MC_ M_NO_FCT_CYCLE_PAR = **<索引>**

说明

在通过系统变量传输信息的 M 功能替换中，M 功能的地址扩展和功能值必须编写为常量值。

允许的编程：

- `M<功能值>`
- `M=<功能值>`
- `M[<地址扩展>]=<功能值>`

不允许的编程：

- `M=<变量 1>`
- `M[<变量 2>]=<变量 1>`

编程

M 功能的替换规则：

- 替换子程序在程序段末尾调用
- 在替换子程序内 M 功能不被替换
- 在 ASUB 中，仅当 ASUB 是在替换子程序内启动时，M 功能才会被替换。

不可替换的 M 功能

下列 M 功能作为预定义辅助功能，会触发系统功能，不允许由子程序替换：

- M0 ... M5
- M17, M30
- M19
- M40 ... M45
- M98，M99（仅在 MD18800 $MN_MM_EXTERN_LANGUAGE ≠ 0 时）

通过机床数据参数设置的用户专用 M 功能也会触发系统功能，因此同样不允许由子程序替换。

机床数据	含义
MD10714 $MN_M_NO_FCT_EOP	复位后生效的主轴用 M 功能
MD10804 $MN_EXTERN_CHAN_M_NO_SET_INT	用于激活 ASUB 的 M 功能（外部模式）
MD10806 $MN_EXTERN_CHAN_M_NO_DISABLE_INT	用于取消 ASUB 的 M 功能（外部模式）
MD10814 $MN_EXTERN_M_NO_MAC_CYCLE	M 功能宏调用
MD20094 $MC_SPIND_RIGID_TAPPING_M_NR	用于切换至开环控制轴运行的 M 功能
MD20095 $MC_EXTERN_RIGID_TAPPING_M_NR	用于切换至开环控制轴运行的 M 功能（外部模式）
MD22254 $MC_AUXFU_ASSOC_M0_VALUE	用于程序停止的附加 M 功能
MD22256 $MC_AUXFU_ASSOC_M1_VALUE	用于有条件停止的附加 M 功能
MD26008 $MC_NIBBLE_PUNCH_CODE	M 功能定义（步冲专用）
MD26012 $MC_PUNCHNIB_ACTIVATION	激活冲裁和步冲功能

说明

特例

MD22560 $MC_TOOL_CHANGE_M_CODE（使用 M 功能换刀）参数设置的 M 功能允许通过子程序替换。

10.18.2.2 替换 T/TCA 和 D/DL 功能

前提条件

下列前提条件适用于 T、TCA、D 和 DL 功能的替换：

- 每个程序段中最多只能有一个功能替换生效。
- 编写了功能替换的程序段**不允许**包含以下元素：
  - M98
  - 模态子程序调用
  - 子程序返回
  - 零件程序结束
- 如果在多刀选择时用 T = 刀位编号通过 MTL 地址进行多刀刀位编号的编程，则由 T 功能替换代替 MTL 地址。在替换子程序中可查询带有系统变量 $C_MTL_PROG 和 $C_MTL 的编程的值。

参数设置：替换子程序

替换子程序在下列机床数据中针对功能设定：

功能	机床数据
T	MD10717 $MN_T_NO_FCT_CYCLE_NAME
TCA	MD15710 $MN_TCA_CYCLE_NAME
D/DL	MD11717 $MN_D_NO_FCT_CYCLE_NAME

说明

建议为 T、TCA 和 D/DL 功能的替换使用相同的子程序。

参数设置：T 功能同时存在时 D 或 DL 功能的相关特性

一个程序段中同时编写了 D/DL 和 T 功能时，或者将 D/DL 编号作为参数传输至替换子程序，或者在调用替换子程序前执行 D/DL 功能。此特性可通过以下机床数据设置：

MD10719 $MN_T_NO_FCT_CYCLE_MODE（T 功能替换的参数设置）

位	值	含义
0	0	D 或 DL 编号在子程序中以系统变量的形式提供（缺省设置）。
	1	D 或 DL 编号直接在程序段中计算。 **提示：** 仅当通过 M 功能配置了换刀时，此功能才生效： MD22550 $MC_TOOL_CHANGE_MODE = 1 否则就总是传输 D 或 DL 值。

用于信息传输的系统变量

程序段中编写的功能的所有相关信息均通过系统变量（参见“系统变量 (页 718)”章节）提供给替换子程序。

系统变量中包含的数据基于编写了待替换功能的程序段。

参数设置：替换子程序的调用时间点

替换子程序的调用时间点通过以下机床数据设置：

MD10719 $MN_T_NO_FCT_CYCLE_MODE，位 1 和位 2

位 2	位 1	替换子程序的调用时间点
0	0	程序段末尾处 执行替换子程序后，通过触发替换的程序段之后的程序行继续插补。
0	1	程序段开始处 执行替换子程序后，对引起替换子程序调用的程序行进行解译。此时将不再处理地址 T、D/DL 和用于换刀的 M 功能。
1	-	程序段开始处和程序段末尾处 调用两次替换子程序。

用于调用时间点的系统变量

通过系统变量 $P_SUB_STAT 可确定替换是否生效；若为是，则还可确定替换子程序的调用时间（以程序段为基准）：

值	含义
0	替换未生效
1	替换生效，在程序段开始处调用子程序
2	替换生效，在程序段末尾处调用子程序

示例：T 功能的替换

参数设置	含义
MD22550 $MC_TOOL_CHANGE_MODE = 0	使用 T 功能换刀
MD10717 $MN_T_NO_FCT_CYCLE_NAME = "MY_T_CYCLE"	用于替换 T 功能的子程序的名称
MD10719 $MN_T_NO_FCT_CYCLE_MODE = 0	调用时间点：程序段末尾

编程	注释
N110 D1	; D1
N120 G90 G0 X100 Y100 Z50	; D1 生效。
N130 D2 X110 Z0 T5	; D1 保持生效。T 功能在程序段末尾处被子程序调用 MY_T_CYCLE 替换。D2 通过一个系统变量提供给 MY_T_CYCLE。

T 功能替换的详细示例请见章节：“换刀时 M/T 功能替换的示例 (页 720)”。

10.18.2.3 系统变量

简介

程序段中编写的功能（T/TCA、D/DL、M）的所有相关信息均通过系统变量提供给替换子程序。

特例

下列情形下不传输 D/DL 编号：

- MD10719 $MN_T_NO_FCT_CYCLE_MODE，位 0 = 1
- MD22550 $MC_TOOL_CHANGE_MODE = 1

与

- D/DL 与 T 功能或 M 功能编写在同一程序段中。

⚠ 小心
值未生效 系统变量中提供给替换子程序的值尚未生效。 用户/机床制造商须自行在替换子程序中进行适宜的编程作为补充。

系统变量

系统变量	含义
$C_M_PROG	编写了 M 功能时为 TRUE
$C_M	$C_M_PROG == TRUE 时，提供地址 M 的值 此时须区分两种情形： • 若针对使用 M 功能的换刀通过参数传输配置了子程序， 那么 $C_M 提供以下机床数据的值：MD10715 MN_M_NO_FCT_CYCLE • 若只为地址 T 和/或 D/DL 配置了一个子程序，且在该程序中同时编写了用于换刀的 M 功能和待替换地址中的一个，那么 $C_M 提供以下机床数据的值：MD22560 $MN_TOOL_CHANGE_M_MODE
$C_AUX_VALUE[0]	替换的 M 功能的值
$C_ME	$C_M_PROG == TRUE 时，提供 M 功能地址扩展的值
$C_AUX_EXT[0]	M 功能的地址扩展（与 $C_ME 相同）
$C_AUX_IS_QUICK[0]	编写了快速输出至 PLC 的 M 功能时为 TRUE
$C_T_PROG	编写了 T 功能时为 TRUE
$C_T	$C_T_PROG == TRUE 时，提供 T 功能的值
$C_TE	在以下情形下： • $C_T_PROG == TRUE • $C_TS_PROG == TRUE 提供 T 功能地址扩展的值
$C_TS_PROG	在 T 或 TCA 替换中编写了刀具名称时为 TRUE
$C_TS	$C_TS_PROG == TRUE 时，提供 T/TCA 替换中编写的刀具名称
$C_TCA	TCA 替换生效时为 TRUE

系统变量	含义
$C_DUPLO_PROG	在 TCA 替换中编写了副编号时为 TRUE
$C_DUPLO	$C_DUPLO_PROG == TRUE 时，提供编写的副编号的值
$C_THNO_PROG	在 TCA 替换中编写了刀架/主轴编号时为 TRUE
$C_THNO	$C_THNO_PROG == TRUE 时，提供编写的刀架/主轴编号的值
$C_D_PROG	编写了 D 功能时为 TRUE
$C_D	$C_D_PROG == TRUE 时，提供 D 功能的值
$C_DL_PROG	编写了 DL 功能时为 TRUE
$C_DL	$C_DL_PROG == TRUE 时，提供 DL 功能的值
$P_SUB_STAT	替换子程序的调用时间点，以程序段为基准
$C_MTL_PROG	TRUE，如果写入了地址 MTL
$C_MTL	$C_MTL_PROG == TRUE 时，提供地址 MTL 的值

10.18.2.4 示例：M 功能的替换

示例 1

通过调用子程序“SUB_M6”替换 M6 功能。

换刀相关信息通过系统变量传输。

参数设置

机床数据	含义
MD10715 $MN_M_NO_FCT_CYCLE[2] = 6	使用 M6 换刀
MD10716 $MN_M_NO_FCT_CYCLE_NAME[2] = "SUB_M6"	M6 的替换子程序
MD10718 $MN_M_NO_FCT_CYCLE_PAR = 2	通过系统变量传输信息

主程序

编程	注释
PROC MAIN	
...	;
N10 T1 D1 M6	; M6 由子程序“SUB_M6”
	; 替换
...	;

编程	注释
N90 M30	

子程序“SUB_M6”

编程	注释
PROC SUB_M6	
N110 IF $C_T_PROG==TRUE	; IF 编写了 T 地址
N120 T[$C_TE]=$C_T	; 执行 T 选择
N130 ENDIF	; ENDIF
N140 M[$C_ME]=6	; 执行换刀。
N150 IF $C_D_PROG==TRUE	; IF 编写了 D 地址
N160 D=$C_D	; 执行 D 选择
N170 ENDIF	; ENDIF
N190 M17	

示例 2

使用 T 功能为换刀准备新刀具。换刀则通过 M6 功能实现。T 功能由子程序“MY_T_CYCLE”替换。D/DL 编号传输至子程序。

参数设置

参数设置	含义
MD22550 $MC_TOOL_CHANGE_MODE = 1	使用 T 功能准备换刀
MD10717 $MN_T_NO_FCT_CYCLE_NAME = "MY_T_CYCLE"	替换子程序
MD10719 $MN_T_NO_FCT_CYCLE_MODE = 0	传输 D/DL 编号

主程序

程序代码	注释
N210 D1	;
N220 G90 G0 X100 Y100 Z50	; D1 生效。
N230 D2 X110 Z0 T5	; D1 保持生效，编写的 D2
	; 作为变量传输给子程序
N240 M6	; 执行换刀

示例 3

使用 T 功能为换刀准备新刀具。换刀则通过 M6 功能实现。T 功能由子程序“MY_T_CYCLE”替换。D/DL 编号不传输至子程序。

参数设置

参数设置	含义
MD22550 $MC_TOOL_CHANGE_MODE = 1	使用 T 功能准备换刀
MD10717 $MN_T_NO_FCT_CYCLE_NAME = "MY_T_CYCLE"	替换子程序
MD10719 $MN_T_NO_FCT_CYCLE_MODE = 1	不传输 D/DL 编号

主程序

程序代码	注释
N310 D1	
N320 G90 G0 X100 Y100 Z50	; D1 生效。
N330 D2 X110 Z0 T5	; D2 生效，不作为变量
	; 传输替换子程序。
N340 M6	; 执行换刀。

示例 4

通过子程序“MY_T_CYCLE”替换 T 功能和 M6 功能。

参数在 M6 替换时传输至子程序。

若程序段中同时编写了 M6 和 D/DL，那么即使未设置 D/DL 编号的传输，D 编号或 DL 编号也会作为参数传输至子程序。

MD10719 $MN_T_NO_FCT_CYCLE_MODE = 1

参数设置

配置	含义
MD22550 $MC_TOOL_CHANGE_MODE = 1	使用 M 功能换刀
MD22560 $MC_TOOL_CHANGE_M_CODE = 6	用于换刀的 M 代码
MD10715 $MC_M_NO_FCT_CYCLE[3] = 6	待替换 M 功能
MD10716 $MC_M_NO_FCT_CYCLE_NAME[3] = "MY_T_CYCLE"	M 功能的替换子程序
MD10717 $MN_T_NO_FCT_CYCLE_NAME = "MY_T_CYCLE"	T 功能的替换子程序
MD10718 $MN_M_NO_FCT_CYCLE_PAR = 3	参数传输至 M6 的替换子程序
MD10719 $MN_T_NO_FCT_CYCLE_MODE = 1	不传输 D/DL 编号

主程序

程序代码	注释
N410 D1	
N420 G90 G0 X100 Y100 Z50	; D1 生效。
N330 D2 X110 Z0 T5 M6	; D1 保持生效，D2 和 T5 作为变量传输至 M6 替换子程序。

10.18.2.5 示例：T 功能和 D 功能的替换

通过调用子程序“D_T_SUB_PROG”替换 T 功能和 D 功能。此示例还遵循以下条件：

- 换刀通过 T 地址进行。
- 子程序调用在程序段开始处进行。
- 刀具管理不生效。
- 轴 B 为配备端齿盘的分度轴。

参数设置

机床数据	含义
MD11717 $MN_D_NO_FCT_CYCLE_NAME = "D_T_SUB_PROG"	D 功能的替换子程序
MD10717 $MN_T_NO_FCT_CYCLE_NAME = "D_T_SUB_PROG"	M 功能的替换子程序
MD10719 $MN_T_NO_FCT_CYCLE_MODE = 'H2'	在程序段开始处调用
MD22550 $MC_TOOL_CHANGE_MODE = 0	使用 T 功能换刀

主程序

编程	注释
PROC MAIN	
...	;
N10 G01 F1000 X10 T1=5 D1	; 替换 T 功能和 D 功能，通过
	; 在程序段开始处调用“D_T_SUB_PROG”实现
...	;
N90 M30	

子程序“D_T_SUB_PROG”

编程	注释
N1000 PROC D_T_SUB_PROG DISPLOF SBLOF	

编程	注释
N4100 IF $C_T_PROG==TRUE	; IF 编写了 T 地址
N4120 POS[B]=CAC($C_T)	; 逼近分度位置
N4130 T[$C_TE]=$C_T	; 选择刀具（T 选择）
N4140 ENDIF	; ENDIF
N4300 IF $C_D_PROG==TRUE	; IF 编写了 D 地址
N4320 D=$C_D	; 选择补偿（D 选择）
N4330 ENDIF	; ENDIF
N4400 IF $C_DL_PROG==TRUE	; IF 编写了 DL 地址
N4420 D=$C_DL	; 选择附加补偿
N4430 ENDIF	; ENDIF
N9999 RET	

10.18.2.6 冲突情形下的特性

冲突情形

在一个程序段中编写了多个功能，而这些功能需要使用不同的子程序进行替换时，会出现冲突。

- D 和 DL 地址通过以下子程序替换：
  MD11717 $MN_FCT_CYCLE_NAME = "D_SUB_PROG"
- T 地址通过以下子程序替换：
  MD10717 $MN_FCT_CYCLE_NAME = "T_SUB_PROG"
- M6 功能通过以下子程序替换：
  MD10715 $MN_M_NO_FCT_CYCLE[0] = 6
  MD10716 $MN_M_NO_FCT_CYCLE_NAME[0] = "M6_SUB_PROG"
  MD10718 $MN_M_NO_FCT_CYCLE_PAR = 0
  MD22550 $MC_TOOL_CHANGE_MODE = 1
  MD22560 $MC_TOOL_CHANGE_M_CODE = 6

解决方案

冲突情形依据下表处理：

在一个程序行中编写了：			调用的子程序：
D 和/或 DL	T 或 TCA	M6	
–	–	x	M6_SUB_PROG
–	x	–	T_SUB_PROG
–	x	x	M6_SUB_PROG
x	–	–	D_SUB_PROG
x	–	x	M6_SUB_PROG
x	x	–	T_SUB_PROG
x	x	x	M6_SUB_PROG

10.18.3 替换主轴功能

10.18.3.1 简介

功能

耦合生效时，可为引导主轴替换以下主轴功能：

- M40：自动齿轮档切换
- M41 ... M45：编程的齿轮档切换
- SPOS, SPOSA 和 M19：主轴定位

前提条件

- 主轴功能的替换须满足以下条件：
  - 编写的主轴必须为生效耦合中的引导主轴。
  - 引导主轴和跟随主轴必须处于同一通道中。 这一点仅在引导主轴处于耦合关闭的通道中时才能识别出。 若引导主轴被交换至另一通道，那么该主轴的齿轮档切换或定位不会触发替换子程序的调用。
  - 编写的齿轮档切换必须能够产生实际的齿轮档切换。 为此须区分编写的齿轮档和生效的齿轮档。
- 在一个程序段中只能替换一个主轴功能。 多次替换会导致程序执行终止。 因此须将需要替换的主轴功能分别写入多个程序段。

参数设置

主轴功能

需要通过子程序替换的主轴功能在以下机床数据中选择：

MD30465 $MA_AXIS_LANG_SUB_MASK

位	含义	
0	自动齿轮档切换（M40）和直接齿轮档切换（M41-M45）	
	值	含义
	0	不替换
	1	由 MD15700 和 MD15702 中设置的子程序替换
1	通过 SPOS/SPOSA/M19 定位主轴	
	值	含义
	0	不替换
	1	由 MD15700 和 MD15702 中设置的子程序替换

子程序：名称

替换子程序的名称在以下机床数据中输入：

MD15700 $MN_LANG_SUB_NAME = "<子程序名称>"

子程序：路径

替换子程序的路径在以下机床数据中设置：

MD15702 $MN_LANG_SUB_PATH = <值>

值	含义
0	**制造商**循环目录：/_N_**CMA**_DIR
1	**用户**循环目录：/_N_**CUS**_DIR
2	**西门子**循环目录：/_N_**CST**_DIR

系统变量：替换子程序的调用时间点

替换子程序的调用时间点可通过系统变量 $P_SUB_STAT 读取：

值	含义
0	替换未生效
1	替换生效，在程序段开始处调用子程序
2	替换生效，在程序段末尾处调用子程序

程序段执行

若替换子程序在程序段开始处调用，那么系统会在执行完替换子程序后执行触发调用的程序段。 此时将不再执行被替换的指令。

若替换子程序在程序段末尾调用，那么系统会先执行引起替换子程序调用的程序段，而不执行待替换的指令。 之后系统会调用替换子程序。

10.18.3.2 M40 - M45（齿轮档切换）的替换

功能

在耦合生效时，通过调用用户专用子程序替换引导主轴的齿轮档切换指令（M40，M41 ... M45）。

参数设置

激活

- MD30465 $MA_AXIS_LANG_SUB_MASK，位 0 = 1

子程序的调用时间点

- M40
  无法设置调用时间点。 替换子程序总是在程序段开始处调用。
- M41 ... M45
  调用时间点取决于所配置的辅助功能向 PLC 的输出特性（参见下面的 MD22080）：
  – 运行**前**或运行**期间**输出：在**程序段开始处**调用子程序。
  – 运行**后**输出：在**程序段末尾**调用子程序
  MD22080 $MC_AUXFU_PREDEF_SPEC[12 ... 16]（M41 ... M45 的输出特性）

位	值	含义
5	1	辅助功能在运行**前**输出
6	1	辅助功能在运行**期间**输出
7	1	辅助功能在运行**后**输出

用于信息传输的系统变量

程序段中编写的功能的所有相关信息均通过系统变量（参见“系统变量 (页 729)”章节）提供给替换子程序。 这些数据只基于编写了待替换功能的程序段。

10.18.3.3 SPOS、SPOSA、M19（主轴定位）的替换

功能

在耦合生效时，通过调用用户专用子程序（替换子程序）替换引导主轴的定位指令（SPOS、SPOSA 或 M19）。

应用示例

在双主轴机床上对工件进行并行加工时，两根主轴以不为 1 的系数耦合。 为了执行换刀，必须将其定位至相同的位置。 为此，替换子程序将关闭耦合并将主轴分别定位至换刀位置，之后再重新激活耦合。

参数设置

激活

- MD30465 $MA_AXIS_LANG_SUB_MASK，位 1 = 1

替换子程序的调用时间点

- SPOS，SPOSA
  无法设置调用时间点。 替换子程序总是在程序段开始处调用。
- M19
  调用时间点取决于所配置的辅助功能向 PLC 的输出特性（参见下面的 MD22080）：
  - 运行**前**或运行**期间**输出：在**程序段开始处**调用子程序。
  - 运行**后**输出：在**程序段末尾**调用子程序

MD22080 $MC_AUXFU_PREDEF_SPEC[9]		
位	值	含义
5	1	辅助功能在运行**前**输出
6	1	辅助功能在运行**期间**输出
7	1	辅助功能在运行**后**输出

用于信息传输的系统变量

程序段中编写的功能的所有相关信息均通过系统变量（参见“系统变量 (页 729)”章节）提供给替换子程序。 这些数据只基于编写了待替换功能的程序段。

10.18.3.4 系统变量

系统变量	含义
$P_SUB_AXFCT	M40、M41 ... M45 的替换生效时为 TRUE
$P_SUB_GEAR	编写的或计算出的齿轮档 在替换子程序以外使用：主主轴的齿轮档
$P_SUB_AUTOGEAR	触发替换的程序段中 M40 生效时为 TRUE。 在替换子程序以外使用：解释器中的当前设置
$P_SUB_LA	提供触发了替换的生效耦合中引导主轴的轴名称。 **提示** 若在替换子程序以外使用此变量，程序执行将终止并发出报警。
$P_SUB_CA	提供触发了替换的生效耦合中跟随主轴的轴名称。 **提示** 若在替换子程序以外调用此变量，程序执行将终止并发出报警。
$P_SUB_AXFCT	提供生效的替换方式，对应 MD30465 $MA_AXIS_LANG_SUB_MASK

系统变量	含义	
$P_SUB_SPOS	SPOS 替换生效时为 TRUE	
$P_SUB_SPOSA	SPOSA 替换生效时为 TRUE	
$P_SUB_M19	M19 替换生效时为 TRUE	
$P_SUB_SPOSIT	提供编写的主轴位置 **提示** 若在替换子程序以外调用此变量，程序执行将终止并发出报警。	
$P_SUB_SPOSMODE	提供针对编写的主轴位置的逼近模式：	
	值	**含义**
	0	位置逼近模式无变化
	1	AC
	2	IC
	3	DC
	4	ACP
	5	ACN
	6	OC
	7	PC
	提示 若在替换子程序以外调用此变量，程序执行将终止并发出报警。	
$P_SUB_STAT	替换子程序的调用时间点，以程序段为基准	

10.18.3.5 示例：齿轮档切换

在子程序中替换所有齿轮档切换指令 M40、M41 ... M45。

参数设置

机床数据	含义
MD15700 $MN_LANG_SUB_NAME = "LANG_SUB"	子程序
MD15702 $MN_LANG_SUB_PATH = 0	制造商目录
MD22080 $MC_AUXFU_PREDEF_SPEC[12] = 'H21'	M41：运行前输出
MD22080 $MC_AUXFU_PREDEF_SPEC[13] = 'H21'	M42：运行前输出
MD22080 $MC_AUXFU_PREDEF_SPEC[13] = 'H21'	M43：运行前输出
MD22080 $MC_AUXFU_PREDEF_SPEC[15] = 'H21'	M44：运行前输出

MD22080 $MC_AUXFU_PREDEF_SPEC[16] = 'H21' M45：运行前输出
MD30465 $MA_AXIS_LANG_SUB_MASK[AX5] = 'H0001' 替换齿轮档切换指令

主程序

编程	注释
PROC MAIN	
N110 COUPON(S2,S1)	; 激活同步主轴耦合
N120 G01 F100 X100 S5000 M3 M43	; 基于 M43 调用子程序
N130 M40	; 激活自动齿轮档切换
N140 M3 S1000	; 基于 S1000 以及
	; 因此触发的自动齿轮档切换
	; 调用子程序
N9999 M30	

替换子程序“LANG_SUB”，方案 1

通过直接定址简化编程和优化速度（S1：引导主轴，S2：跟随主轴）。

编程	注释
N1000 PROC LANG_SUB DISPLOF SBLOF	
N1100 IF($P_SUB_AXFCT ==1)	; 齿轮档切换触发替换
N1140 DELAYFSTON	; 开始停止延时段
N1150 COUPOF(S2,S1)	; 取消同步主轴耦合
N1160 ; 引导主轴和跟随主轴分别切换齿轮档	
N1170 M1=$P_SUB_GEAR M2=$P_SUB_GEAR	
N1180 DELAYFSTON	; 结束停止延时段
N1190 COUPON(S2,S1)	; 激活同步主轴耦合
N1200 ENDIF	
...	
N9999 RET	

替换子程序“LANG_SUB”，方案 2

通过系统变量间接定址，提升灵活度（引导主轴：$P_SUB_LA，跟随主轴：$P_SUB_CA）。

编程	注释
N1000 PROC LANG_SUB DISPLOF SBLOF	
N1010 DEF AXIS _LA	; 引导轴/主轴的标志位
N1020 DEF AXIS _CA	; 跟随轴/主轴的标志位

编程	注释
N1030 DEF INT _GEAR	; 齿轮档的标志位
N1100 IF($P_SUB_AXFCT==1)	; 齿轮档切换触发替换
N1110 _GEAR=$P_SUB_GEAR	; 待激活的齿轮档
N1120 _LA=$P_SUB_LA	; 引导主轴的轴名称
N1130 _CA=$P_SUB_CA	; 跟随主轴的轴名称
N1140 DELAYFSTON	; 开始停止延时段
N1150 COUPOF(_CA,_LA)	; 取消同步主轴耦合
N1160 ；引导主轴和跟随主轴切换齿轮档	
N1170 M[AXTOSPI(_LA)]=_GEAR M[AXTOSPI(_CA)]=_GEAR	
N1180 DELAYFSTOF	; 结束停止延时段
N1190 COUPON(_CA,_LA)	; 激活同步主轴耦合
N1200 ENDIF	
...	
N9999 RET	

10.18.3.6 示例：主轴定位

在子程序中只显性执行 SPOS 和 SPOSA 指令的替换。其他替换须相应添加。

参数设置

机床数据	含义
MD30465 $MA_AXIS_LANG_SUB_MASK[AX5] = 'H0002'	替换定位指令
MD22080 $MC_AUXFU_PREDEF_SPEC[9] = 'H0021'	M19 在运行前输出至 PLC

设定数据	含义
SD43240 $SA_M19_SPOS[AX5] = 260	M19 下的主轴位置 = 260
SD43250 $SA_M19_SPOSMODE[AX5] = 4	M19 下的位置逼近模式：“正向趋近（ACP）”

主程序

编程	注释
PROC MAIN	
...	
N210 COUPON(S2,S1)	; 激活同步主轴耦合

编程	注释
N220 SPOS[1]=100	; 用 SPOS 定位引导主轴
...	
N310 G01 F1000 X100 M19	; 用 M19 定位引导主轴

替换子程序“LANG_SUB”，方案 1

通过直接定址简化编程和优化速度（S1：引导主轴，S2：跟随主轴）。

编程	注释
N1000 PROC LANG_SUB DISPLOF SBLOF	
N2100 IF($P_SUB_AXFCT==2)	
N2110 ; 在同步主轴耦合生效时替换 SPOS/SPOSA/M19	
N2185 DELAYFSTON	; 开始停止延时段
N2190 COUPOF(S2,S1)	; 取消同步主轴耦合
N2200	; 定位引导主轴和跟随主轴
N2210 IF($P_SUB_SPOS==TRUE) OR ($P_SUB_SPOSA==TRUE)	
N2220 ; SPOS 和 SPOSA 映射至 SPOS	
N2230 CASE $P_SUB_SPOSMODE OF \	
0 GOTOF LABEL1_DC \	
1 GOTOF LABEL1_IC \	
2 GOTOF LABEL1_AC \	
3 GOTOF LABEL1_DC \	
4 GOTOF LABEL1_ACP \	
5 GOTOF LABEL1_ACN \	
DEFAULT GOTOF LABEL_ERR	
LABEL1_DC:SPOS[1]=DC($P_SUB_SPOSIT) SPOS[2]=DC($P_SUB_SPOSIT)	
GOTOF LABEL1_CONT	
LABEL1_IC:DELAYFSTOF	
SPOS[1]=IC($P_SUB_SPOSIT) SPOS[2]=IC($P_SUB_SPOSIT)	
DELAYFSTON	
GOTOF LABEL1_CONT	
LABEL1_AC:SPOS[1]=AC($P_SUB_SPOSIT) SPOS[2]=AC($P_SUB_SPOSIT)	
GOTOF LABEL1_CONT	
LABEL1_ACP:SPOS[1]=ACP($P_SUB_SPOSIT) SPOS[2]=ACP($P_SUB_SPOSIT)	
GOTOF LABEL1_CONT	
LABEL1_ACN:SPOS[1]=ACN($P_SUB_SPOSIT) SPOS[2]=ACN($P_SUB_SPOSIT)	
LABEL1_CONT:	
N2250 ELSE	; 用 M19 定位主轴
N2270 M1=19 M2=19	; 引导主轴和跟随主轴
N2280 ENDIF	; SPOS、SPOSA 替换结束
N2285 DELAYFSTOF	; 结束停止延时段
N2290 COUPON(S2,S1)	; 激活同步主轴耦合
N2410 ELSE	

编程	注释
N2420 ；自此开始进行其他替换	
...	
N3300 ENDIF	; 替换结束
...	
N9999 RET	; 一般程序结束
LABEL_ERR:SETAL(61000)	; 出错

替换子程序“LANG_SUB”，方案 2

通过系统变量间接定址，提升灵活度（引导主轴：$P_SUB_LA，跟随主轴：$P_SUB_CA）。

编程	注释
N1000 PROC LANG_SUB DISPLOF SBLOF	
N1010 DEF AXIS _LA	; 引导轴/主轴
N1020 DEF AXIS _CA	; 跟随轴/主轴
N1030 DEF INT _LSPI	; 引导主轴编号（编写的
	; 主轴）
N1040 DEF INT _CSPI	; 跟随主轴编号
...	
N2100 IF($P_SUB_AXFCT==2)	
N2110 ; 在同步主轴耦合生效时替换 SPOS/SPOSA/M19	
N2120 _LA=$P_SUB_LA	; 引导主轴的轴名称
N2130 _CA=$P_SUB_CA	; 跟随主轴的轴名称
N2140 _LSPI=AXTOSPI(_LA)	; 引导主轴的编号
N2180 _CSPI=AXTOSPI(_LA)	; 跟随主轴的编号
N2185 DELAYFSTON	; 开始停止延时段
N2190 COUPOF(_CA,_LA)	; 取消同步主轴耦合
N2200	; 定位引导主轴和跟随主轴：
N2210 IF($P_SUB_SPOS==TRUE) OR ($P_SUB_SPOSA==TRUE)	
N2220 ; SPOS 和 SPOSA 映射至 SPOS	
N2230 CASE $P_SUB_SPOSMODE OF	
0 GOTOF LABEL1_DC \	
1 GOTOF LABEL1_IC \	
2 GOTOF LABEL1_AC \	
3 GOTOF LABEL1_DC \	
4 GOTOF LABEL1_ACP \	
5 GOTOF LABEL1_ACN \	
DEFAULT GOTOF LABEL_ERR	
LABEL1_DC:SPOS[_LSPI]=DC($P_SUB_SPOSIT) SPOS[_CSPI]=DC($P_SUB_SPOSIT)	
GOTOF LABEL1_CONT	

编程	注释
LABEL1_IC:DELAYFSTOF	
SPOS[_LSPI]=IC($P_SUB_SPOSIT) SPOS[_CSPI]=IC($P_SUB_SPOSIT)	
DELAYFSTON	
GOTOF LABEL1_CONT	
LABEL1_AC:SPOS[_LSPI]=AC($P_SUB_SPOSIT) SPOS[_CSPI]=AC($P_SUB_SPOSIT)	
GOTOF LABEL1_CONT	
LABEL1_ACP:SPOS[_LSPI]=ACP($P_SUB_SPOSIT) POS[_CSPI]=ACP($P_SUB_SPOSIT)	
GOTOF LABEL1_CONT	
LABEL1_ACN:SPOS[_LSPI]=ACN($P_SUB_SPOSIT) POS[_CSPI]=ACN($P_SUB_SPOSIT)	
LABEL1_CONT:	
N2250 ELSE	; 用 M19 定位主轴
N2270 M[_LSPI]=19 M[_CSPI]=19	
N2280 ENDIF	
N2285 DELAYFSTOF	; 结束停止延时段
N2290 COUPON(_CA,_LA)	; 激活同步主轴耦合
N2410 ELSE	
N2420 ; 自此开始进行其他替换	
...	
N3300 ENDIF	
...	
N9999 RET	; 一般程序结束
LABEL_ERR:SETAL(61000)	; 出错

10.18.4 子程序的属性

一般规则

- 替换中调用的子程序可包含 PROC 指令，以及 SBLOF 和 DISPLOF 属性。
- 替换也可在 ISO 语言模式下进行。但是替换子程序只在标准语言模式（西门子）下执行。此时系统会隐性切换至标准语言模式。从替换子程序返回时，系统将重新切换回原先的语言模式。
- 对替换子程序的信息传输只通过系统变量进行。无法使用传输参数。
- 单程序段和 SBLOF 属性下的特性取决于以下机床数据中的设置：
  MD10702 IGNORE_SINGLEBLOCK_MASK，位 14（忽略单程序段停止）

值	含义
0	替换子程序的特性与“一般”子程序相同： ● 通过 M17 返回：在子程序末尾停止 **提示** M 功能向 PLC 的输出取决于以下机床数据： MD20800 $MC_SPF_END_TO_VDI，位 0（子程序结束输出至 PLC） - 位 0 = 0：不输出 - 位 0 = 1：M17 输出至 PLC。 ● 通过 RET 跳回：在替换子程序末尾处不停止
1	在调用替换子程序的程序段中**只停止一次**。并且这与以下因素无关： ● 子程序是否是在程序段开始处和/或程序段末尾调用 ● 子程序中是否调用了其他子程序 ● 子程序是否通过 M17 或 RET 退出 对于 M 功能的替换，单程序段停止在替换子程序末尾处进行。 对于 T 和 D/DL 功能的替换，单程序段停止的时间点取决于子程序的调用时间点： ● 在程序段开始处调用：在程序段末尾进行单程序段停止 ● 在程序段末尾处调用：在替换子程序末尾进行单程序段停止

- 对于带 DISPLOF 属性的替换子程序，程序段显示中会显示引起子程序调用的程序行作为当前程序段。
- 在替换子程序中可使用 DELAYFSTON 和 DELAYFSTOF 指令指定区域或整个替换子程序，防止其受到中断（例如 NC 停止、读取禁止等）。
- 替换不会递归进行。也就是说，对于引起替换子程序调用的功能，若在替换子程序中再次编写，则其不再会被执行。

辅助功能向 PLC 的输出

替换辅助功能时，调用替换子程序不会引起辅助功能向 PLC 的输出。仅当在替换子程序中重新编写了辅助功能时，其才会被输出。

程序段搜索时的特性

替换子程序也可在“进行计算的程序段搜索”和“程序测试模式下进行计算的程序段搜索（SERUPRO）”这些程序段搜索模式下调用。可能存在的特殊性必须在替换子程序中使用系统变量 $P_SEARCH 和 $AC_SERUPRO 实现。

在“进行计算的程序段搜索”下的动作收集方面，替换子程序的特性与“一般”子程序相似。

10.18.5 前提条件

- 以下情形下不允许进行功能替换：
  - 同步动作
  - 工艺循环
- 若程序段开始处包含待替换的功能，那么该程序段之前不允许为逐段生效的同步动作。 参见下面的“示例：逐段生效的同步动作”部分。
- 在替换子程序中只允许执行相应替换所需的动作。
- 对于在程序段末尾调用替换子程序的程序段，须注意以下事项：
  - 不允许有模态生效的子程序调用生效。
  - 不允许编写子程序返回
  - 不允许编写程序结束

注意
控制系统不会监控替换子程序中是否实现了待替换功能。

示例：逐段生效的同步动作

MD30465 $MA_AXIS_LANG_SUB_MASK，位 0 = 1（齿轮级切换）

程序代码

```
...
N1000 WHENEVER $AA_IM[X2] <= $AA_IM[X1] + 0.5 DO $AA_OVR[X1]=0
N1010 G1 X100 M43
...
```

若程序段 N1010 中 M43 功能引起替换子程序的调用，那么加工将随之终止并显示报警。

10.19 重命名/禁用 NC 指令

功能

借助“重命名/禁用 NC 指令”功能，机床制造商或最终用户可以修改现有 NC 指令的名称。

应用

该功能可用于：

- 改善零件程序的可读性
- 禁用 NC 指令
- 扩展 NC 功能

参数设置

通过以下机床数据重命名/禁用 NC 指令：

MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[<n>]（重新配置后的 NC 代码列表）

偶数索引 [<n>] 包含指令的原始名称。奇数索引包含指令的新名称。

说明

在 SINUMERIK 828D 上，偶数索引（原始指令名称）是预先给定的且不可修改。只能写入奇数索引（新指令名称）。

空字符串 ("") 表示没有给定指令名称。指令因此被禁用且无法再进行编程。

对 MD10712 的修改在重新上电后生效。

示例

示例 1 (840D sl)：重命名指令 G00 并禁用指令 G01

MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[0] = "G00"

MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[1] = "EILGANG"

MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[2] = "G01"

MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[3] = ""

通过重新上电生效后，无法再通过指令 G00，而是通过指令 EILGANG 编程快速运行。G01 无法再进行编程。

示例 2 (828D)：修改 SINUMERIK 828D 的预设

为了适应 ISO 语言模式，SINUMERIK 828D 上预设了索引 [0] 到 [7]，将指令 G505 和 G506 更名为 G58 和 G59。G58 和 G59 为可编程框架，G505 和 G506 为可设定框架。更名后，G58 和 G59 变为可设定框架。

按如下步骤撤销重命名可使 G58 和 G59 变回可编程框架：

MD10712 的索引	指令名称	
	预定义的	新建
[0]	"G58"	"G58"
[1]	**""**	**"G58"**
[2]	"G59"	"G59"
[3]	**""**	**"G59"**
[4]	"G505"	"G505"
[5]	**"G58"**	**"G505"**
[6]	"G506"	"G506"
[7]	**"G59"**	**"G506"**

示例 3 (828D)：将指令 _TCA 更名为 TCA

缺省设置下，指令 TCA 在 SINUMERIK 828D 中自动更名为 _TCA。TCA 功能有一个循环可用。如不使用该循环，必须再次将指令 _TCA 更名为 TCA：

MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[**8**] = "_TCA"

MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[**9**] = "TCA"

示例 4 (828D)：将现有指令转为用户专用循环

用一个可实现用户专用功能的循环替代现有指令 WAITM。

1. 通过 MD10712 将现有指令 WAITM 更名为 _WAITM：
   MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[**20**] = "WAITM"
   MD10712 $MN_NC_USER_CODE_CONF_NAME_TAB[**21**] = "_WAITM"
2. 创建一个名为 WAITM 的零件程序。
   PROC WAITM 的接口与 NC 功能 WAITM 的接口必须一致。
   该零件程序现在可以替代之前的 WAITM NC 功能。零件程序、西门子循环和 Shopmill/Shopturn 循环中的每个 WAITM 指令都可调用用户专用子程序 WAITM。用户专用功能在该子程序 WAITM 中运行，必要时可以调用更名的 NC 功能 _WAITM。

程序代码	注释
N5 PROC WAITM（<标记号>、<通道号>）	
...	；用户专用功能
N100 _WAITM（<标记号>、<通道号>）	
...	；用户专用功能
M30	

10.20 程序运行时间/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。

10.20.1 程序运行时间

10.20.1.1 功能

功能“程序运行时间”提供了用于监控工艺过程，它可以通过系统变量在零件程序和同步动作中读取。有两种不同类型的计时器：

1. 标准计时器
   标准计时器始终生效。
2. 可激活的计时器
   可激活的计时器必须通过机床数据激活。

标准计时器

从上一次控制系统启动到现在的时间

系统变量	含义
$AN_SETUP_TIME	从上一次使用缺省值启动控制系统到现在的时间，单位分 在每次使用缺省值启动控制系统时，该变量都将自动复位为“0”。
$AN_POWERON_TIME	从上一次控制系统普通启动（热启动）到现在的时间，单位 min 在每次控制系统普通启动时，该变量都将自动复位为“0”。

程序运行时间

用于测量程序运行时间的计时器仅在 AUTO 模式下可用。

通道专用系统变量	说明
$AC_ACT_PROG_NET_TIME	当前程序的当前净运行时间，单位 s 净运行时间表示：程序停止所花费的时间已被减去。 若在 AUTO 运行方式下从 RESET 通道状态重新启动一个零件程序，那么 $AC_ACT_PROG_NET_TIME 会自动复位为“0”。 其他属性： ● 复位键不会使 $AC_ACT_PROG_NET_TIME 复位为“0”，而只是使计时器停止。 ● 启动 ASUB 时，系统会将 $AC_ACT_PROG_NET_TIME 设置为“0”，并对 ASUB 的运行时间进行计数。Am Ende eines ASUPs verhält es sich wie bei der RESET-Taste:der Timer wird nur angehalten, aber nicht auf "0" gesetzt. ● 在启动事件控制的程序（PROG_EVENT）时，$AC_ACT_PROG_NET_TIME 不会被复位。 仅当该程序为启动/M30/搜索 PROG_EVENT 时，程序运行时间才继续计数。 ● GOTOS 和倍率 = 0% 条件下 $AC_ACT_PROG_NET_TIME 的特性可通过 MD27850 设置（参见“参数设置”部分） **提示：** 可通过 $AC_PROG_NET_TIME_TRIGGER 继续操控 $AC_ACT_PROG_NET_TIME。
$AC_OLD_PROG_NET_TIME	刚正确结束程序的净运行时间，单位 s “正确结束”表示程序不是通过 RESET 终止，而是通过常规的 M30 结束。 启动新程序时，$AC_OLD_PROG_NET_TIME 保持不变，直至重新到达 M30。 其他属性： ● 当前选择的程序被编辑时，$AC_OLD_PROG_NET_TIME 会被设置为“0”。 ● 在 ASUB 或事件控制程序（PROG_EVENT）的末尾 $AC_OLD_PROG_NET_TIME 不会改变。 **提示：** 仅当 $AC_PROG_NET_TIME_TRIGGER 未赋值时，才会发生从 $AC_ACT_PROG_NET_TIME 向 $AC_OLD_PROG_NET_TIME 的隐性复制。

<table>
<tr><th>通道专用系统变量</th><th colspan="2">说明</th></tr>
<tr><td>$AC_OLD_PROG_NET_TIME_COUNT</td><td colspan="2">$AC_OLD_PROG_NET_TIME 的变化
上电后 $AC_OLD_PROG_NET_TIME_COUNT 置“0”。
当控制系统重新写入 $AC_OLD_PROG_NET_TIME 时，
$AC_OLD_PROG_NET_TIME_COUNT 总是随之提升。
用户通过 RESET 终止运行中的程序时，
$AC_OLD_PROG_NET_TIME 和
$AC_OLD_PROG_NET_TIME_COUNT 保持不变。
因此，通过 $AC_OLD_PROG_NET_TIME_COUNT 可确定
$AC_OLD_PROG_NET_TIME 是否已被写入。
示例：
若两个连续运行的程序运行时间相同并已正确结束，那么用户可通过 $AC_OLD_PROG_NET_TIME_COUNT 中改变的值加以识别。</td></tr>
<tr><td rowspan="6">$AC_PROG_NET_TIME_TRIGGER</td><td colspan="2">用于选择性测量程序段的触发器
通过将值写入变量触发功能：</td></tr>
<tr><td>值</td><td>功能</td></tr>
<tr><td>0</td><td>无效</td></tr>
<tr><td>1</td><td>结束测量
• $AC_OLD_PROG_NET_TIME = $AC_ACT_PROG_NET_TIME
• $AC_ACT_PROG_NET_TIME = 0，随后继续运行</td></tr>
<tr><td>2</td><td>启动测量
• $AC_ACT_PROG_NET_TIME = 0
• $AC_OLD_PROG_NET_TIME 未改变</td></tr>
<tr><td>3</td><td>停止测量
• $AC_OLD_PROG_NET_TIME 未改变
• $AC_OLD_PROG_NET_TIME 未改变</td></tr>
<tr><td>4</td><td>继续停止的测量
• $AC_ACT_PROG_NET_TIME 继续运行
• $AC_OLD_PROG_NET_TIME 未改变</td></tr>
<tr><td colspan="3">上电时所有系统变量将复位为“0”！</td></tr>
</table>

说明

工件的剩余时间

如果需要依次加工相同的工件，可通过下列计时器值计算该工件的剩余时间：

- 上一个工件的加工时间（参见 $AC_OLD_PROG_NET_TIME）
- 当前的加工时间（参见 $AC_ACT_PROG_NET_TIME）

除当前加工时间外，剩余时间也会在操作界面上显示。

说明

使用 STOPRE

系统变量 $AC_OLD_PROG_NET_TIME 和 $AC_OLD_PROG_NET_TIME_COUNT 不会产生隐性的预处理停止。在零件程序中使用时，若系统变量的值来自于之前的程序运行，那么这一点影响不大。但若高频率地写入运行时间测量触发器（$AC_PROG_NET_TIME_TRIGGER），并由此导致 $AC_OLD_PROG_NET_TIME 频繁变更，则需在零件程序中写入一个显性的 STOPRE。

可激活的计时器

程序运行时间

用于测量程序运行时间的计时器仅在 AUTO 模式下可用。

通道专用系统变量	说明
$AC_OPERATING_TIME	AUTO 运行方式下 NC 程序的总运行时间，单位 s AUTO 运行方式下，NC 启动和程序结束/NC 复位间所有程序的运行时间的累加值。 缺省条件下，NC 停止和倍率 = 0% 时此变量不进行计数。可通过 MD27860 激活 0% 倍率下的继续计数。 在每次控制系统启动时，该变量都将自动复位为“0”。
$AC_CYCLE_TIME	所选择的 NC 程序的运行时间，单位 s 测量所选 NC 程序中 NC 启动和程序结束/NC 复位间的运行时间。 缺省条件下，NC 停止和倍率 = 0% 时此变量不进行计数。可通过 MD27860 激活 0% 倍率下的继续计数。 在每次启动新的 NC 程序时，该值都将自动复位为“0”。可通过机床数据 MD27860 设置，是否在使用 GOTOS 跳转至程序头或启动 ASUB 和 PROG_EVENT 时进行复位。
$AC_CUTTING_TIME	加工时间，单位 s 测量所有 NC 程序中 NC 启动和程序结束/NC 复位间无快进生效的轨迹轴（其中至少一根轴生效）的运行时间。可通过 MD27860 设置，是只通过生效的刀具进行测量，还是进行与刀具无关的测量。 此外测量会在暂停时间生效时中断。 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”。

10.20.1.2 调试

激活/取消激活

可激活的计时器可通过以下机床数据激活/取消：

MD27860 $MC_PROCESSTIMER_MODE，位 0 - 2 = <值>

位	值	含义
0	0	$AC_OPERATING_TIME 的计时器不生效。
	1	$AC_OPERATING_TIME 的计时器生效。

位	值	含义
1	0	$AC_CYCLE_TIME 的计时器不生效。
	1	$AC_CYCLE_TIME 的计时器生效。
2	0	$AC_CUTTING_TIME 的计时器不生效。
	1	$AC_CUTTING_TIME 的计时器生效。

参数设置

恒定生效的计时器的特性

恒定生效的计时器在 GOTOS 和倍率 = 0% 情形下的特性可通过以下机床数据设置：

MD27850 $MC_PROG_NET_TIMER_MODE

位	值	含义
0	0	通过 GOTOS 跳转至程序头时，$AC_ACT_PROG_NET_TIME 不复位为“0”（缺省设置）。
	1	通过 GOTOS 跳转至程序头时，$AC_ACT_PROG_NET_TIME 复位为“0”，$AC_OLD_PROG_NET_TIME 中之前的值被保存，程序计数器 $AC_OLD_PROG_NET_TIME_COUNT 提升。
1	0	在倍率 = 0% 时 $AC_ACT_PROG_NET_TIME 不提升。亦即，在倍率设置为“0”的时间内不计算程序运行时间（缺省设置）。
	1	倍率 = 0% 时 $AC_ACT_PROG_NET_TIME 同样提升。亦即，在倍率设置为“0”的时间内仍计算程序运行时间。

可激活计时器的特性

特定功能（例如空运行进给率、程序测试）下可激活计时器的特性通过以下机床数据设置：

MD27860 $MC_PROCESSTIMER_MODE

位	值	含义
4	0	空运行进给率生效时不进行测量。
	1	空运行进给率生效时仍进行测量。
5	0	程序测试模式下不进行测量。
	1	程序测试模式下仍进行测量。

位	值	含义
6	仅针对位 1 = 1（$AC_CYCLE_TIME 的计时器生效）	
	0	通过 ASUB 和 PROG_EVENT 启动时，$AC_CYCLE_TIME 也复位为“0”。
	1	通过 ASUB 和 PROG_EVENT 启动时，$AC_CYCLE_TIME 不复位为“0”。
7	仅针对位 2 = 1（$AC_CUTTING_TIME 的计时器生效）	
	0	$AC_CUTTING_TIME 的计时器仅在刀具生效时计数。
	1	$AC_CUTTING_TIME 的计时器计数与刀具无关。
8	仅针对位 1 = 1（$AC_CYCLE_TIME 的计时器生效）	
	0	通过 GOTOS 跳转至程序头时，$AC_CYCLE_TIME 不复位为“0”（缺省设置）。
	1	通过 GOTOS 跳转至程序头时，$AC_CYCLE_TIME 复位为“0”。
9	仅针对位 0，1 = 1（$AC_OPERATING_TIME 和 $AC_CYCLE_TIME 的计时器生效）	
	0	倍率 = 0% 时，程序运行时间不继续计数
	1	倍率 = 0% 时，程序运行时间继续计数

10.20.1.3 前提条件

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）。

10.20.1.4 示例

示例 1：Parametrierung der Laufzeitmessung über MD27860

为生效的 NC 程序激活运行时间测量，同时在试运行进给和程序测试生效时不进行测量：

MD27860 $MC_PROCESSTIMER_MODE = 'H2'

激活对刀具动作时间的测量，在试运行进给和程序测试生效时同样进行测量：

MD27860 $MC_PROCESSTIMER_MODE = 'H34'

激活刀具生效时对总运行时间和加工时间的测量，在程序测试模式下同样进行测量：

MD27860 $MC_PROCESSTIMER_MODE = 'H25'

激活对总运行时间和加工时间的测量（与刀具无关），在程序测试模式下同样进行测量：

MD27860 $MC_PROCESSTIMER_MODE = 'Ha5'

激活刀具生效时对加工时间的测量，在倍率 = 0% 时同样进行测量，但在空运行进给率生效时不进行测量：

MD27860 $MC_PROCESSTIMER_MODE = 'H22'

示例 2：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
```

程序处理 N80 行后，$AC_OLD_PROG_NET_TIME 中为“mySubProgrammA”的净运行时间。

$AC_OLD_PROG_NET_TIME 的值：

- 在 M30 后保持不变
- 在每次完整运行循环后更新

示例 3：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
```

程序代码

```
N70 DO $AC_PROG_NET_TIME_TRIGGER=1
N80 mySubProgrammD
N90 M30
```

10.20.2 工件计数器

10.20.2.1 功能

“工件计数器”功能以通道专用系统变量的形式提供多个值域为 0 至 999.999.999 的计数器。这些系统变量均可读写。

通过下列通道专用机床数据可以对计数器激活、归零时刻和计数算法进行调整。

用于工件计数的系统变量

系统变量	含义
$AC_REQUIRED_PARTS	待加工工件的数量（设定工件数量） 在此计数器中可以定义工件的个数，在到达这个数值之后，实际工件数（$AC_ACTUAL_PARTS）复位为“0”。 通过 MD27880 可生成显示报警： “通道 %1 工件设定数量 = %2 已达到” 和通道专用 NC/PLC 接口信号： DB21, ... DBX317.1（达到工件设定数量）。
$AC_TOTAL_PARTS	已加工工件的总数（实际工件总数） 该计数器给出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”。
$AC_ACTUAL_PARTS	已加工工件的数量（实际工件数） 此计数器会记录自开始时刻起加工的所有工件的数量。当达到设定工件数量（$AC_REQUIRED_PARTS）时，该计数器会自动归“0”（前提条件是 $AC_REQUIRED_PARTS > 0）。
$AC_SPECIAL_PARTS	用户计数的工件数量 该计数器可协助用户进行自定义的工件计数。可定义在到达设定工件数（$AC_REQUIRED_PARTS）时输出报警。用户必须自行将该计数器归零。

说明

在以缺省值启动控制系统时，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。

10.20.2.2 调试

激活

工件计数器通过以下机床数据激活：

MD27880 $MC_PART_COUNTER（激活工件计数器）

位	值	含义
0	1	$AC_REQUIRED_PARTS 生效。
1	0	以下情形时输出报警/信号：$AC_ACTUAL_PARTS = $AC_REQUIRED_PARTS
	1	以下情形时输出报警/信号：$AC_SPECIAL_PARTS = $AC_REQUIRED_PARTS
4	1	$AC_TOTAL_PARTS 生效。
5	0	M02 / M30 使 $AC_TOTAL_PARTS 的计数提升“1”。
	1	通过 MD27882[0] 定义的 M 指令使 $AC_TOTAL_PARTS 的计数提升“1”。
6	0	$AC_TOTAL_PARTS 在程序测试/程序段搜索时同样生效。
7	1	通过 GOTOS 返回时，$AC_TOTAL_PARTS 的计数提升“1”。
8	1	$AC_ACTUAL_PARTS 生效。
9	0	M02 / M30 使 $AC_ACTUAL_PARTS 的计数提升“1”。
	1	通过 MD27882[1] 定义的 M 指令使 $AC_ACTUAL_PARTS 的计数提升“1”。
10	0	$AC_ACTUAL_PARTS 在程序测试/程序段搜索时同样生效。
11	1	通过 GOTOS 返回时，$AC_ACTUAL_PARTS 的计数提升“1”。
12	1	$AC_SPECIAL_PARTS 生效。
13	0	M02 / M30 使 $AC_SPECIAL_PARTS 的计数提升“1”。
	1	通过 MD27882[2] 定义的 M 指令使 $AC_SPECIAL_PARTS 的计数提升“1”。

位	值	含义
14	0	$AC_SPECIAL_PARTS 在程序测试/程序段搜索时同样生效。
15	1	通过 GOTOS 返回时，$AC_SPECIAL_PARTS 的计数提升“1”。

参数设置

通过用户定义的 M 指令触发工件计数

若 MD27880 中设置了相应位，可通过以下机床数据设置的 M 指令代替程序结束 M2/M30 来触发计数脉冲。

MD27882 $MC_PART_COUNTER_MCODE[<n>]（过用户定义的 M 指令触发工件计数）

<n>	含义
0	MD27882[0] 用于定义使 $AC_TOTAL_PARTS 计数提升的 M 指令。
1	MD27882[1] 用于定义使 $AC_ACTUAL_PARTS 计数提升的 M 指令。
2	MD27882[2] 用于定义使 $AC_SPECIAL_PARTS 计数提升的 M 指令。

在调用用户定义的 M 指令时，相关工件计数器的数值会提升“1”。

工件计数的保护等级

借助操作界面，通过以下机床数据为工件计数的激活/关闭设置保护等级：

MD51074 $MN_ACCESS_WRITE_WPC_COUNTER = <保护等级>

10.20.2.3 前提条件

运行方式切换/NC-RESET

运行方式切换和 NC-RESET 不会影响工件计数器。

$AC_REQUIRED_PARTS ≤ 0

$AC_REQUIRED_PARTS ≤ 0 且 MD27880 $MC_PART_COUNTER 位 0 == 1 时，所有生效的计数器不执行计数及通过 MD27880 设置的识别校验。

10.20.2.4 示例

激活工件计数器 $AC_REQUIRED_PARTS：

- MD27880 $MC_PART_COUNTER = 'H3'

$AC_REQUIRED_PARTS 生效。

以下情形时显示报警：$AC_REQUIRED_PARTS == $AC_SPECIAL_PARTS

激活工件计数器 $AC_TOTAL_PARTS:

- MD27880 $MC_PART_COUNTER = 'H10'
- MD27882 $MC_PART_COUNTER_MCODE[0] = 80

$AC_TOTAL_PARTS 生效，每个 M02 会使计数提升“1”。

$MC_PART_COUNTER_MCODE[0] 没有意义。

激活工件计数器 $AC_ACTUAL_PARTS：

- MD27880 $MC_PART_COUNTER = 'H300'
- MD27882 $MC_PART_COUNTER_MCODE[1] = 17

$AC_TOTAL_PARTS 生效，每个 M17 会使计数提升“1”。

激活工件计数器 $AC_SPECIAL_PARTS:

- MD27880 $MC_PART_COUNTER = 'H3000'
- MD27882 $MC_PART_COUNTER_MCODE[2] = 77

$AC_SPECIAL_PARTS 生效。

每个 M77 使 $AC_SPECIAL_PARTS 的计数提升 1

关闭工件计数器 $AC_ACTUAL_PARTS：

- MD27880 $MC_PART_COUNTER = 'H200'
- MD27882 $MC_PART_COUNTER_MCODE[1] = 50

$AC_ACTUAL_PARTS 不生效。

激活所有计数器：

- MD27880 $MC_PART_COUNTER = 'H3313'
- MD27882 $MC_PART_COUNTER_MCODE[0] = 80
- MD27882 $MC_PART_COUNTER_MCODE[1] = 17
- MD27882 $MC_PART_COUNTER_MCODE[2] = 77

$AC_REQUIRED_PARTS 生效。

以下情形时显示报警：$AC_REQUIRED_PARTS == $AC_SPECIAL_PARTS

$AC_TOTAL_PARTS 生效，每个 M02 会使计数提升“1”。

$MC_PART_COUNTER_MCODE[0] 没有意义。

$AC_ACTUAL_PARTS 生效，每个 M17 会使计数提升“1”。

$AC_SPECIAL_PARTS 生效，每个 M77 会使计数提升 1。

程序测试/程序段搜索时，工件计数器 $AC_ACTUAL_PARTS 无效：

- MD27880 $MC_PART_COUNTER = 'H700'
- MD27882 $MC_PART_COUNTER_MCODE[1] = 75

$AC_ACTUAL_PARTS 生效，每个 M75 会使计数提升“1”，程序测试和程序段搜索除外。

位 0 = 1 时取消 MD27880 $MC_PART_COUNTER 中涉及的计数模式：

- MD27882 $MC_PART_COUNTER_MCODE[0] = 41
- MD27882 $MC_PART_COUNTER_MCODE[1] = 42
- MD27882 $MC_PART_COUNTER_MCODE[2] = 43

程序代码	注释
...	
N100 $AC_REQUIRED_PARTS=-10	; 值 < 0：设置计数。
N200 M41 M43	; 不计数。
N300 M42	
...	
N500 $AC_REQUIRED_PARTS=52	; 值 > 0：计数依据 MD27880 生效。
N501 M43	; 计数。
N502 M42 M41	; 计数。
...	

10.21 数据表

10.21.1 功能

“工件计数器”功能以通道专用系统变量的形式提供多个值域为 0 至 999.999.999 的计数器。这些系统变量均可读写。

通过下列通道专用机床数据可以对计数器激活、归零时刻和计数算法进行调整。

用于工件计数的系统变量

系统变量	含义
$AC_REQUIRED_PARTS	待加工工件的数量（设定工件数量） 在此计数器中可以定义工件的个数，在到达这个数值之后，实际工件数（$AC_ACTUAL_PARTS）复位为“0”。 通过 MD27880 可生成显示报警： “通道 %1 工件设定数量 = %2 已达到” 和通道专用 NC/PLC 接口信号： DB21, ... DBX317.1（达到工件设定数量）。
$AC_TOTAL_PARTS	已加工工件的总数（实际工件总数） 该计数器给出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”。
$AC_ACTUAL_PARTS	已加工工件的数量（实际工件数） 此计数器会记录自开始时刻起加工的所有工件的数量。当达到设定工件数量（$AC_REQUIRED_PARTS）时，该计数器会自动归“0”（前提条件是 $AC_REQUIRED_PARTS > 0）。
$AC_SPECIAL_PARTS	用户计数的工件数量 该计数器可协助用户进行自定义的工件计数。可定义在到达设定工件数（$AC_REQUIRED_PARTS）时输出报警。用户必须自行将该计数器归零。

说明

在以缺省值启动控制系统时，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。

10.21.2 机床数据

10.21.2.1 通用机床数据

显示机床数据

编号	名称：$MM_	说明
SINUMERIK Operate		
9421	MA_AXES_SHOW_GEO_FIRST	最先显示通道的几何轴
9422	MA_PRESET_MODE	JOG 方式下 PRESET/基本偏移
9423	MA_MAX_SKP_LEVEL	跳转级的最大数量

NC 专用机床数据

编号	名称： $MN_	说明
10010	ASSIGN_CHAN_TO_MODE_GROUP	运行方式组中有效的通道
10125	EES_NC_NAME	NCU 名称，用于在 EES 运行中生成独有 NC 程序名称
10280	PROG_FUNCTION_MASK	比较指令“>”和“<”
10700	PREPROCESSING_LEVEL	程序预处理级
10702	IGNORE_SINGLEBLOCK_MASK	忽略单程序段停止
10707	PROG_TEST_MASK	程序测试模式
10708	SERUPRO_MASK	程序段搜索模式
10710	PROG_SD_RESET_SAVE_TAB	待更新的设定数据
10711	NC_LANGUAGE_CONFIGURATION	使用选件或功能未激活的语言指令运行的方式。
10713	M_NO_FCT_STOPRE	带预处理停止的 M 功能
10715	M_NO_FCT_CYCLE	需由子程序替换的 M 功能
10716	M_NO_FCT_CYCLE_NAME	用于 M 功能替换的子程序名称
10717	T_NO_FCT_CYCLE_NAME	用于 T 功能替换的换刀循环的名称
10718	M_NO_FCT_CYCLE_PAR	带参数的 M 功能替换
10719	T_NO_FCT_CYCLE_MODE	T 功能替换的参数设定
11450	SEARCH_RUN_MODE	搜索的参数设置

编号	名称： $MN_	说明
11470	REPOS_MODE_MASK	重新定位属性
11600	BAG_MASK	ASUB 相关的运行方式组特性
11602	ASUP_START_MASK	忽略 ASUB 的停止原因
11604	ASUP_START_PRIO_LEVEL	$MN_ASUP_START_MASK 生效的起始优先级
11610	ASUP_EDITABLE	激活用户专用 ASUB 程序
11612	ASUP_EDIT_PROTECTION_LEVEL	用户专用 ASUB 程序的保护级
11620	PROG_EVENT_NAME	PROG_EVENT 的程序名称
11625	FILE_ONLY_WITH_EXTENSION	调用程序时仅搜索有文件标识的文件
11626	CYCLES_ONLY_IN_CYCDIR	仅在循环目录下搜索含接口的程序
11717	D_NO_FCT_CYCLE_NAME	用于 D 功能替换的子程序名称
15700	LANG_SUB_NAME	替换子程序的名称
15702	LANG_SUB_PATH	替换子程序的调用路径
17200	GMMC_INFO_NO_UNIT	全局 HMI 信息（无物理单位）
17201	GMMC_INFO_NO_UNIT_STATUS	全局 HMI 状态信息（无物理单位）
18045	EES_MODE_INFO	执行 EES 功能时所处于的模式
18360	MM_EXT_PROG_BUFFER_SIZE	用于执行外部程序的 FIFO 缓冲器大小 (DRAM)
18362	MM_EXT_PROG_NUM	可同时于外部执行的程序级的数量

10.21.2.2 通道专用机床数据

基本机床数据

编号	名称：$MC_	说明
20000	CHAN_NAME	通道名称
20050	AXCONF_GEOAX_ASSIGN_TAB	几何轴指定为通道轴
20060	AXCONF_GEOAX_NAME_TAB	通道中的几何轴名称
20070	AXCONF_MACHAX_USED	通道内有效的机床轴编号
20080	AXCONF_CHANAX_NAME_TAB	通道中的通道轴名称 [通道轴编号]：0...7
20090	SPIND_DEF_MASTER_SPIND	通道内主主轴的复位位置
20100	DIAMETER_AX_DEF	带端面轴功能的几何轴
20106	PROG_EVENT_IGN_SINGLEBLOCK	程序事件忽略单程序段

编号	名称：$MC_	说明
20107	PROG_EVENT_IGN_INHIBIT	程序事件忽略读取禁止
20108	PROG_EVENT_MASK	事件控制的程序调用
20109	PROG_EVENT_MASK_PROPERTIES	程序事件的属性
20114	MODESWITCH_MASK	重新定位的设置
20116	IGNORE_INHIBIT_ASUP	忽略读取禁止完整执行用户 ASUB
20117	IGNORE_SINGLEBLOCK_ASUP	忽略单程序段完整执行用户 ASUB
20160	CUBIC_SPLINE_BLOCKS	C 样条下程序段的数量
20170	COMPRESS_BLOCK_PATH_LIMIT	压缩时一条程序段的最大运行长度
20191	IGN_PROG_STATE_ASUP	中断程序的执行在操作界面上不显示
20192	PROG_EVENT_IGN_PROG_STATE	程序事件的执行在操作界面上不显示
20193	PROG_EVENT_IGN_STOP	程序事件忽略停止键
20194	IGNORE_NONCSTART_ASUP	针对特定用户报警在报警应答“NC 启动禁用”存在的情形下支持 ASUB 启动
20210	CUTCOM_CORNER_LIMIT	TRC 中平衡程序段的最大角度
20220	CUTCOM_MAX_DISC	DISC 的最大值
20230	CUTCOM_CURVE_INSERT_LIMIT	TRC 中交点计算的最大角度
20240	CUTCOM_MAXNUM_CHECK_BLOCKS	TRC 中预读轮廓计算的程序段
20250	CUTCOM_MAXNUM_DUMMY_BLOCKS	TRC 中不含运行指令程序段的最大数量
20270	CUTTING_EDGE_DEFAULT	未作设定时刀沿的初始设置
20400	LOOKAH_USE_VELO_NEXT_BLOCK	预读（Lookahead）后续程序段的数量
20430	LOOKAH_NUM_OVR_POINTS	预读中倍率开关基准值数量
20440	LOOKAH_OVR_POINTS	预读中倍率开关基准值
20500	CONST_VELO_MIN_TIME	恒定速度的最短持续时间
20600	MAX_PATH_JERK	最大轨迹急动度
20610	ADD_MOVE_ACCEL_RESERVE	为叠加运动保留的加速度余量
20700	REFP_NC_START_LOCK	不带参考点的 NC 启动禁用
20750	ALLOW_GO_IN_G96	G96 、G961 时的 G0 逻辑
20800	SPF_END_TO_VDI	子程序结束发送至 PLC
21000	CIRCLE_ERROR_CONST	圆弧终点监控常量
21010	CIRCLE_ERROR_FACTOR	圆弧终点监控系数
21100	ORIENTATION_IS_EULER	定向编程时的角度定义

编号	名称：$MC_	说明
21110	X_AXIS_IN_OLD_X_Z_PLANE	自动框架定义中的坐标系
21200	LIFTFAST_DIST	从轮廓快速退回时的运行行程
21210	SETINT_ASSIGN_FASTIN	为 NC 程序中断的外部 NC 输入字节指定硬件
21202	LIFTFAST_WITH_MIRROR	镜像下的快速退刀

程序段搜索

编号	名称：$MC_	说明
20128	COLLECT_TOOL_CHANGE	程序段搜索后向 PLC 发出换刀指令
22600	SERUPRO_SPEED_MODE	程序段搜索类型 5 下的速度
22601	SERUPRO_SPEED_FACTOR	程序段搜索类型 5 下的速度系数
22621	ENABLE_START_MODE_MASK_PRT	SERUPRO 程序段搜索时切换 MD22620:START_MODE_MASK_ PRT
22622	DISABLE_PLC_START	允许通过 PLC 启动零件程序
22680	AUTO_IPTR_LOCK	禁用中断指针

复位特性

编号	名称：$MC_	说明
20110	RESET_MODE_MASK	复位/零件程序结束后的初始设置
20112	START_MODE_MASK	零件程序启动后 NC-Start 时的初始设置
20118	GEOAX_CHANGE_RESET	允许自动几何轴切换
20120	TOOL_RESET_VALUE	启动（复位/零件程序结束）中的刀具长度补偿
20121	TOOL_PRESEL_RESET_VALUE	复位时的预选刀具
20130	CUTTING_EDGE_RESET_VALUE	启动（复位/零件程序结束）中的刀具刀沿长度补偿
20140	TRAFO_RESET_VALUE	启动（复位/零件程序结束）中的转换数据组
20150	GCODE_RESET_VALUES	G 功能组的初始设置
20152	GCODE_RESET_MODE	G 功能组的复位特性
20156	MAXNUM_GCODES_EXT	外部 G 功能组的复位特性
22620	START_MODE_MASK_PRT	启动后 NC-Start 和复位时的初始设置

辅助功能设置

编号	名称：$MC_	说明
22000	AUXFU_ASSIGN_GROUP	辅助功能组
22010	AUXFU_ASSIGN_TYPE	辅助功能类型
22020	AUXFU_ASSIGN_EXTENSION	辅助功能扩展
22030	AUXFU_ASSIGN_VALUE	辅助功能数值
22200	AUXFU_M_SYNC_TYPE	M 功能的输出时间
22210	AUXFU_S_SYNC_TYPE	S 功能的输出时间
22220	AUXFU_T_SYNC_TYPE	T 功能的输出时间
22230	AUXFU_H_SYNC_TYPE	H 功能的输出时间
22240	AUXFU_F_SYNC_TYPE	F 功能的输出时间
22250	AUXFU_D_SYNC_TYPE	D 功能的输出时间
22400	S_VALUES_ACTIVE_AFTER_RESET	S 功能复位后保持生效
22410	F_VALUES_ACTIVE_AFTER_RESET	F 功能复位后保持生效
22510	GCODE_GROUPS_TO_PLC	程序段切换/复位时输出至 NC/PLC 接口的 G 指令
22550	TOOL_CHANGE_MODE	使用 M 功能时的新刀具补偿
22560	TOOL_CHANGE_M_CODE	用于换刀的 M 功能

存储器设置

编号	名称：$MC_	说明
27900	REORG_LOG_LIMIT	用于使能日志文件的插补缓存大小，%
28000	MM_REORG_LOG_FILE_MEM	用于重组的存储空间（DRAM）
28010	MM_NUM_REORG_LUD_MODULES	重组时用于本地用户变量的模块数量
28020	MM_NUM_LUD_NAMES_TOTAL	本地用户变量的数量（DRAM）
28040	MM_LUD_VALUES_MEM	本地用户变量的存储空间（DRAM）
28050	MM_NUM_R_PARAM	通道专用 R 参数的数量（SRAM）
28060	MM_IPO_BUFFER_SIZE	插补缓存中 NC 程序段的数量（DRAM）
28070	MM_NUM_BLOCKS_IN_PREP	预处理的程序段数量（DRAM）
28080	MM_NUM_USER_FRAMES	可设置框架的数量（SRAM）

编号	名称：$MC_	说明
28090	MM_NUM_CC_BLOCK_ELEMENTS	用于编译循环的程序段单元的数量（DRAM）
28100	MM_NUM_CC_BLOCK_USER_MEM	用于编译循环的程序段存储空间（DRAM）
28400	MM_ABSBLOCK	激活含绝对值的基本程序段
28402	MM_ABSBLOCK_BUFFER_CONF	确定上载缓存的大小
28500	MM_PREP_TASK_STACK_SIZE	准备任务的栈大小（DRAM）

程序运行时间和工件计数器

编号	名称：$MC_	说明
27860	PROCESSTIMER_MODE	激活运行时间测量
27880	PART_COUNTER	激活工件计数器
27882	PART_COUNTER_MCODE[ ]	通过 M 指令触发工件计数

10.21.2.3 进给轴/主轴专用机床数据

编号	名称：$MA_	说明
30465	AXIS_LANG_SUB_MASK	NC 语言指令的替换
30550	AXCONF_ASSIGN_MASTER_CHAN	用于跨通道取轴的通道缺省设置
30600	FIX_POINT_POS	G75 中轴的固定值位置
33100	COMPRESS_POS_TOL	压缩时的最大偏差

10.21.3 设定数据

10.21.3.1 通道专用设定数据

编号	名称：$SC_	说明
42000	THREAD_START_ANGLE	螺纹加工起始角
42010	THREAD_RAMP_DISP	攻丝中轴的加速性能

编号	名称：$SC_	说明
42100	DRY_RUN_FEED	空运行进给率
42200	SINGLEBLOCK2_STOPRE	激活 SBL2 的排故模式
42444	TARGET_BLOCK_INCR_PROG	进行计算的程序段搜索后的累加模式
42700	EXT_PROG_PATH	外部子程序调用 EXTCALL 的程序路径
42750	ABSBLOCK_ENABLE	使能基本程序段显示
42990	MAX_BLOCKS_IN_IPOBUFFER	插补缓存中程序段的最大数量

10.21.4 信号

10.21.4.1 发送至 NC 的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
急停	DB10.DBX56.1	DB2600.DBX0.1

10.21.4.2 发送至 BAG 的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
AUTO 运行方式	DB11.DBX0.0	DB3000.DBX0.0
MDI 运行方式	DB11.DBX0.1	DB3000.DBX0.1
JOG 运行方式	DB11.DBX0.2	DB3000.DBX0.2
禁止运行方式切换	DB11.DBX0.4	DB3000.DBX0.4
BAG 停止	DB11.DBX0.5	DB3000.DBX0.5
BAG 停止，进给轴和主轴	DB11.DBX0.6	DB3000.DBX0.6
BAG 复位	DB11.DBX0.7	DB3000.DBX0.7
机床功能 TEACH IN	DB11.DBX1.0	DB3000.DBX1.0
机床功能 REPOS	DB11.DBX1.1	-
机床功能 REF	DB11.DBX1.2	DB3000.DBX1.2

10.21.4.3 从 BAG 发出的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
选择的 AUTO 运行方式	DB11.DBX4.0	-
选择的 MDI 运行方式	DB11.DBX4.1	-
选择的 JOG 运行方式	DB11.DBX4.2	-
选择的机床功能 Teach In	DB11.DBX5.0	-
选择的机床功能 REPOS	DB11.DBX5.1	-
选择的机床功能 REF	DB11.DBX5.2	-
AUTO 运行方式生效	DB11.DBX6.0	DB3100.DBX0.0
MDI 运行方式生效	DB11.DBX6.1	DB3100.DBX0.1
JOG 运行方式生效	DB11.DBX6.2	DB3100.DBX0.2
BAG 就绪	DB11.DBX6.3	DB3100.DBX0.3
BAG 复位已执行	DB11.DBX6.4	DB3100.DBX0.4
NC 内部 JOG 生效	DB11.DBX6.5	-
所有通道处于复位状态	DB11.DBX6.7	DB3100.DBX0.7
机床功能 TEACH IN 生效	DB11.DBX7.0	DB3100.DBX1.0
机床功能 REPOS 生效	DB11.DBX7.1	-
机床功能 REF 生效	DB11.DBX7.2	DB3100.DBX1.2

10.21.4.4 发送至通道的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
激活 DRF	DB21, ...DBX0.3	DB320x.DBX0.3
激活单程序段	DB21, ...DBX0.4	DB320x.DBX0.4
激活 M01	DB21, ...DBX0.5	DB320x.DBX0.5
激活空运行进给率	DB21, ...DBX0.6	DB320x.DBX0.6
PLC 动作结束	DB21, ...DBX1.6	-
激活程序测试	DB21, ...DBX1.7	DB320x.DBX1.7
程序段跳转级：/0 至 /7	DB21, ...DBX2.0-7	DB320x.DBX2.0-7
读取禁止	DB21, ...DBX6.1	DB320x.DBX6.1
程序级中断	DB21, ...DBX6.4	DB320x.DBX6.4

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
NC 启动禁止	DB21, ...DBX7.0	DB320x.DBX7.0
NC 启动	DB21, ...DBX7.1	DB320x.DBX7.1
程序段交界处 NC 停止	DB21, ...DBX7.2	DB320x.DBX7.2
NC 停止	DB21, ...DBX7.3	DB320x.DBX7.3
NC 停止进给轴和主轴	DB21, ...DBX7.4	DB320x.DBX7.4
复位	DB21, ...DBX7.7	DB330x.DBX3.7
REPOSPATHMODE	DB21, ...DBX31.0-2	-
REPOSMODEEDGE	DB21, ...DBX31.4	-

10.21.4.5 从通道发出的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
DRF 已选择	DB21, ...DBX24.3	DB170x.DBX0.3
选择 NC 关联 M01	DB21, ...DBX24.4	-
M01 已选择	DB21, ...DBX24.5	DB170x.DBX0.5
空运行进给率已选择	DB21, ...DBX24.6	DB170x.DBX0.6
REPOSPATHMODE 0 - 2	DB21, ...DBX25.0-2	-
快速移动进给倍率已选择	DB21, ...DBX25.3	DB170x.DBX1.3
REPOS MODE EDGE	DB21, ...DBX25.4	-
程序测试已选择	DB21, ...DBX25.7	DB170x.DBX1.7
程序段跳转已选择：/0 - /7	DB21, ...DBX26.0-7	DB170x.DBX2.0-7
程序段跳转已选择：/8	DB21, ...DBX27.0	DB170x.DBX3.0
程序段跳转已选择：/9	DB21, ...DBX27.1	DB170x.DBX3.1
REPOSPATHMODE:0 - 2	DB21, ...DBX31.0-2	-
REPOS MODE EDGE	DB21, ...DBX31.4	-
程序段跳转生效：/8	DB21, ...DBX31.6	DB320x.DBX15.6
程序段跳转生效：/9	DB21, ...DBX31.7	DB320x.DBX15.7
外部执行生效	DB21, ...DBX32.0	DB330x.DBX0.0
动作程序段生效	DB21, ...DBX32.3	DB330x.DBX0.3
逼近程序段生效	DB21, ...DBX32.4	DB330x.DBX0.4

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
M0/M1 生效	DB21, ...DBX32.5	DB330x.DBX0.5
最后的动作程序段生效	DB21, ...DBX32.6	DB330x.DBX0.6
程序段搜索生效	DB21, ...DBX33.4	DB330x.DBX1.4
M02/M30 生效	DB21, ...DBX33.5	DB330x.DBX1.5
转换生效	DB21, ...DBX33.6	DB330x.DBX1.6
程序测试生效	DB21, ...DBX33.7	DB330x.DBX1.7
程序状态：运行	DB21, ...DBX35.0	DB330x.DBX3.0
程序状态：等待	DB21, ...DBX35.1	DB330x.DBX3.1
程序状态：停止	DB21, ...DBX35.2	DB330x.DBX3.2
程序状态：中断	DB21, ...DBX35.3	DB330x.DBX3.3
程序状态：终止	DB21, ...DBX35.4	DB330x.DBX3.4
通道状态：生效	DB21, ...DBX35.5	DB330x.DBX3.5
通道状态：中断	DB21, ...DBX35.6	DB330x.DBX3.6
通道状态：复位	DB21, ...DBX35.7	DB330x.DBX3.7
中断处理生效	DB21, ...DBX36.4	-
通道就绪	DB21, ...DBX36.5	DB330x.DBX4.5
读取使能被忽略	DB21, ...DBX37.6	-
SBL 下程序段末尾的停止被抑制	DB21, ...DBX37.7	-
G 指令组 1 – n 的生效 G 功能的编号（8 位整数）	DB21, ...DBB208-271	DB350x.DBB0-63
达到设定工件数	DB21, ...DBX317.1	DB330x.DBX4001.1
ASUB 已停止	DB21, ...DBX318.0	DB330x.DBX4002.0
通过程序测试进行的程序段搜索生效	DB21, ...DBX318.1	-
REPOS MODE EDGEACKN	DB21, ...DBX319.0	-
Repos 路径模式应答：0 - 2	DB21, ...DBX319.1-3	-
Repos DEFERAL Chan	DB21, ...DBX319.5	-
在事件控制的程序调用中显示触发事件	DB21, ...DBX376.0-7	DB330x.DBB4004
ASUB 生效	DB21, ...DBX378.0	DB330x.DBB4006.0
抑制了显示更新的 ASUB 生效	DB21, ...DBX378.1	DB330x.DBB4006.1

10.21.4.6 发送至进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
REPOSDELAY	DB31, ...DBX10.0	-

10.21.4.7 来自进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
REPOS 偏移	DB31, ...DBX70.0	-
REPOS 偏移有效	DB31, ...DBX70.1	-
REPOS 延迟应答	DB31, ...DBX70.2	-
REPOSDELAY	DB31, ...DBX72.0	-
轨迹轴	DB31, ...DBX76.4	DB390x.DBX1002.4

K2：轴、坐标系、框架 11

11.1 简要说明

11.1.1 轴

机床轴

机床轴指的是在机床上实际存在的轴。

通道轴

每根几何轴和辅助轴总是被指定给一个通道，从而被指定为通道轴。 几何轴和辅助轴总是在所归属于的通道中运行。

几何轴

三根几何轴总是构成一个虚拟的直角坐标系，即基本坐标系（BCS）。

通过使用框架（偏移、旋转、比例缩放、镜像）可将工件坐标系（WCS）的几何轴映射至 BCS。

辅助轴

与几何轴不同，辅助轴之间未定义几何关联。

轨迹轴

轨迹轴的特点是一同进行插补（一个通道的所有轨迹轴共用一个轨迹插补器）。

一个通道的所有轨迹轴具有共同的加速、恒速运行和减速阶段。

定位轴

定位轴的特点是独立进行插补（每根定位轴均有一个专属的轴插补器）。 每根定位轴均具有独立的进给率和加速特性曲线。

同步轴

同步轴和轨迹轴一同进行插补（一个通道的所有轨迹轴和同步轴共用一个轨迹插补器）。

一个通道的所有轨迹轴和同步轴具有共同的加速、恒速运行和减速阶段。

轴配置

有关几何轴、辅助轴、通道轴和机床轴之间的分配以及各个轴类型的名称定义通过下列机床数据进行：

MD20050 $MC_AXCONF_GEOAX_ASIGN_TAB（几何轴指定为通道轴）

MD20060 $MC_AXCONF_GEOAX_NAME_TAB（通道中的几何轴名称）

MD20070 $MC_AXCONF_MACHAX_USED（通道中生效的机床轴编号）

MD20080 $MC_AXCONF_CHANAX_NAME_TAB（通道中的通道轴名称）

MD10000 $MN_AXCONF_MACHAX_NAME_TAB（机床轴名称）

MD35000 $MA_SPIND_ASSIGN_TO_MACHAX（主轴指定为机床轴）

可切换的几何轴

借助“可切换的几何轴”功能，可通过零件程序将其他通道轴组成几何轴组。

通道中先前配置为同步辅助轴的轴也可通过程序指令替换任意几何轴。

Link 轴

Link 轴指的是实际连接至另一个 NCU，且受该 NCU 位置闭环控制的轴。Link 轴可动态指定给**另一个** NCU 的通道。对于特定的 NCU 而言，Link 轴不是本地轴。

轴容器方案用于动态变更对 NCU 的指定关系。 通过零件程序中的 GET 和 RELEASE 进行的跨通道取轴不适用于超出 NCU 界限的 Link 轴。

对 Link 轴的说明请见：
文档：
功能手册之扩展功能分册；多操作面板配备多 NCU，分布式系统（B3）

轴容器

轴容器是一种环形缓存数据结构，用于将本地轴和/或 Link 轴指定给通道。环形缓存中的记录为**循环浮动**。。

在配置链接轴时，在机床轴逻辑图中除了可以直接参照本地轴或者 Link 轴外，也允许参照轴容器。

此类参照由以下要素构成：

- 轴容器编号
- 槽（相应轴容器中的环形缓存位）

环形缓存位中的记录可为：

- 一根本地轴
  或者
- 一根 Link 轴

对“轴容器”功能的说明请见：
文档：
功能手册之扩展功能分册；多操作面板配备多 NCU，分布式系统（B3）

11.1.2 坐标系

MCS

机床坐标系（MCS）具有以下属性：

- 由机床轴构成。
- 机床轴可相互垂直构成笛卡尔坐标系，也可采用任意布局。
- 机床轴的名称可设置。
- 机床轴可为线性轴或回转轴。

BCS

基本坐标系（BCS）具有以下属性：

- 由几何轴构成的笛卡尔直角坐标系。
- BCS 通过从 MCS 运动转换得到。

BZS

基本零点坐标系（BZS）是带基准偏移的基本坐标系。

SZS

从工件坐标系（WCS）角度看来，可设定的零点坐标系（SZS）是带可编程框架的 WCS。 工件零点通过可设定框架 G54 ... G599 定义。

WCS

工件坐标系（WCS）具有以下属性：

- 在 WCS 中，所有轴坐标均为编写（零件程序）。
- 由几何轴和辅助轴构成。
- 几何轴总是构成一个笛卡尔直角坐标系。
- 辅助轴构成一个坐标系，辅助轴之间无几何关联。
- 几何轴和辅助轴的名称可设置。
- 可通过框架实现对 WCS 的偏移、旋转、比例缩放和镜像（TRANS、ROT、SCALE、MIRROR）。
  也可实现多重偏移、多重旋转等。

外部零点偏移

外部零点偏移具有以下属性：

- 预先定义的基本坐标系和工件坐标系间的附加零点偏移，在一个由 PLC 决定的时间点激活。
- 对于涉及的每根轴，偏移量均可通过以下方式设置：
  - PLC
  - 操作面板
  - 零件程序
- 激活后，该偏移从执行轴的第一个运动程序段起开始生效。 这些偏移会被叠加至编写的路径（非插补）。
  运行外部零点偏移的速度为：
  编写的 F 值 + 1/2 JOG 速度
  在 G0 程序段中，零点偏移会在程序段末尾运行。
- RESET 和程序结束时，激活的偏移将被保留。
- 上电后，上一次生效的偏移继续保存在控制系统中，但是必须通过 PLC 重新激活。

11.1.3 框架

框架是一种既定的，用于实现笛卡尔坐标系转换的计算规定。

框架分量

一个框架由以下分量组成：

框架分量		编程时可使用：
偏移	粗偏	TRANS ATRANS（叠加偏移分量） CTRANS（对多根轴的零点偏移） G58（轴零点偏移）
	精偏	CFINE G59（轴零点偏移）
旋转		ROT / ROTS AROT / AROTS CROTS
比例缩放		SCALE ASCALE
镜像		MIRROR AMIRROR

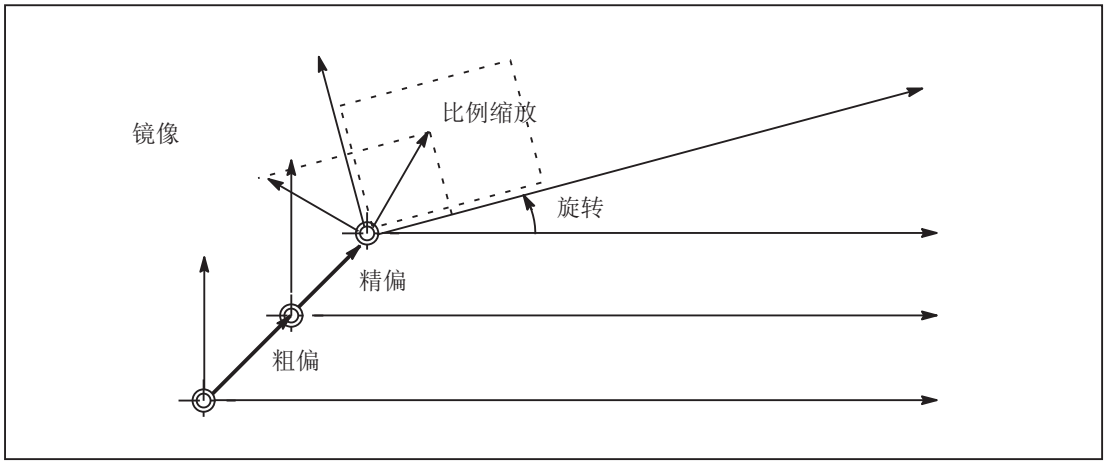


图 11-1 框架分量

粗偏和精偏

在所有通道中，机床轴对通道轴的指定关系，特别是机床轴对几何轴的指定关系可能不同，因此通道轴之间不存在几何关联。因此，NCU 全局框架中只可进行偏移、缩放和镜像。不可旋转。

G58：绝对轴专用可编程零点偏移（粗偏移）

通过 G58 编程平移（粗偏移）的绝对分量。保留平移（精偏移）的叠加分量。

G59：叠加轴专用可编程零点偏移（精偏移）

通过 G59 编程平移（精偏移）的叠加分量。保留平移（粗偏移）的绝对分量。

仅当精偏移使能时才可使用 G59：

- MD18600 $MN_MM_FRAME_FINE_TRANS = TRUE
- MD24000 $MC_FRAME_ADD_COMPONENTS = TRUE

旋转

空间中的定向通过框架旋转如下定义：

- 通过 ROT 定义针对所有几何轴的单次旋转。
- 通过 ROTS、AROTS、CROTS 定义空间角，确定平面在空间内的定向。
- 通过 TOFRAME 定义框架旋转，确定一个 Z 轴指向刀具方向的框架。

比例缩放

使用 SCALE 为所有几何轴和辅助轴编写可编程的比例缩放（标度系数）。

若需在其他比例缩放、旋转、偏移或镜像的基础上建立新的比例缩放，则须编写 ASCALE。

镜像

通过以下机床数据设置如何执行镜像：

MD10610 $MN_MIRROR_REF_AX

级联

框架和框架分量可通过级联运算符 ":" 进行级联。通道专用框架 $P_ACTFRAME 由所有通道生效框架级联得出：

$P_ACTFRAME = $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

边界条件

增量尺寸设定（G91）

对于 G91 增量编程，选择零点偏移时，补偿值会被累加至增量编写的值一并运行。

此特性取决于以下设定数据中的设置：

SD42440 $SC_FRAME_OFFSET_INCR_PROG （框架中的零点偏移）

值	含义
1	在 框架和轴采用增量编程的情形下运行零点偏移（= 缺省设置）。
0	只运行编写的行程。

一致性

在写入、读取和激活框架（例如通过通道协调）时，用户须自行确保通道内的一致性。无法跨通道激活框架。

11.2 轴

11.2.1 概述

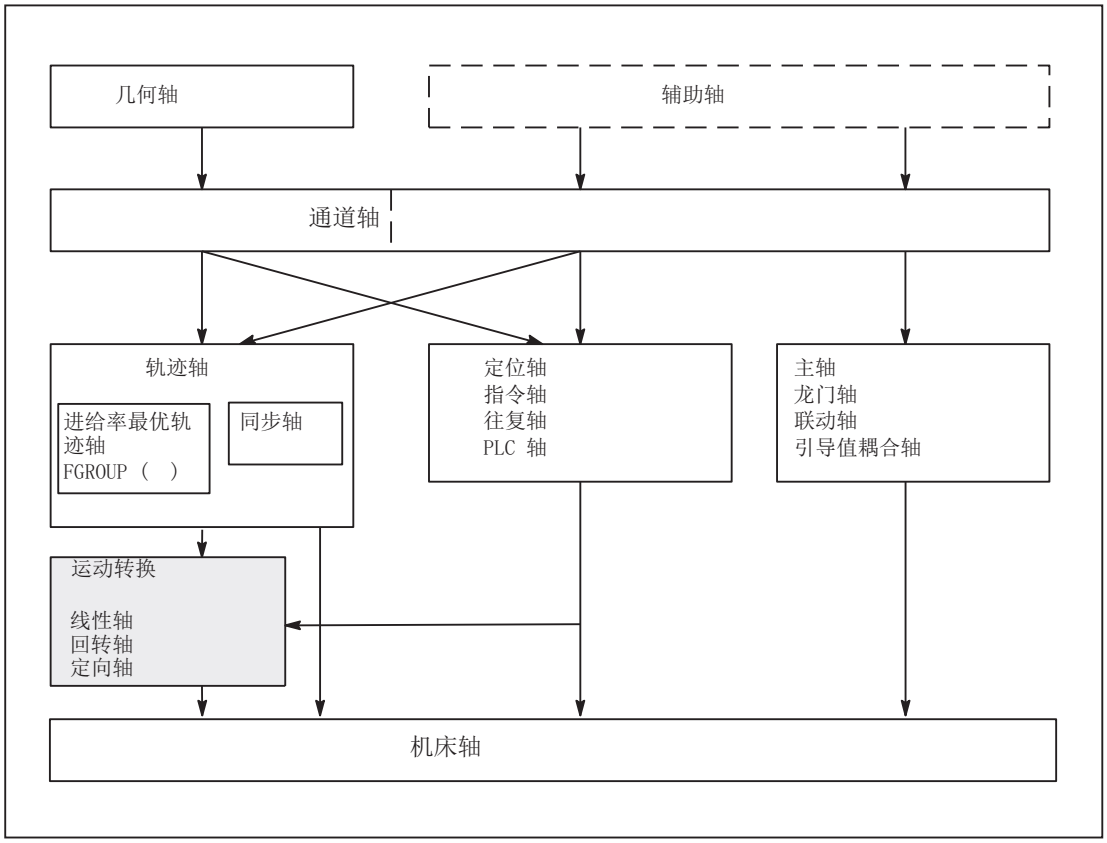


图 11-2 几何轴、辅助轴和机床轴之间的关联

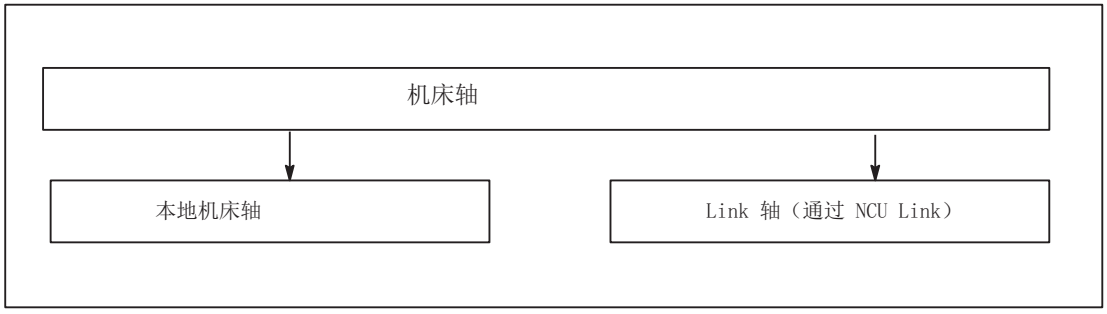


图 11-3 本地机床轴和外部机床轴（Link 轴）之间的关联

11.2.2 机床轴

含义

机床轴指的是在机床上实际存在的轴。

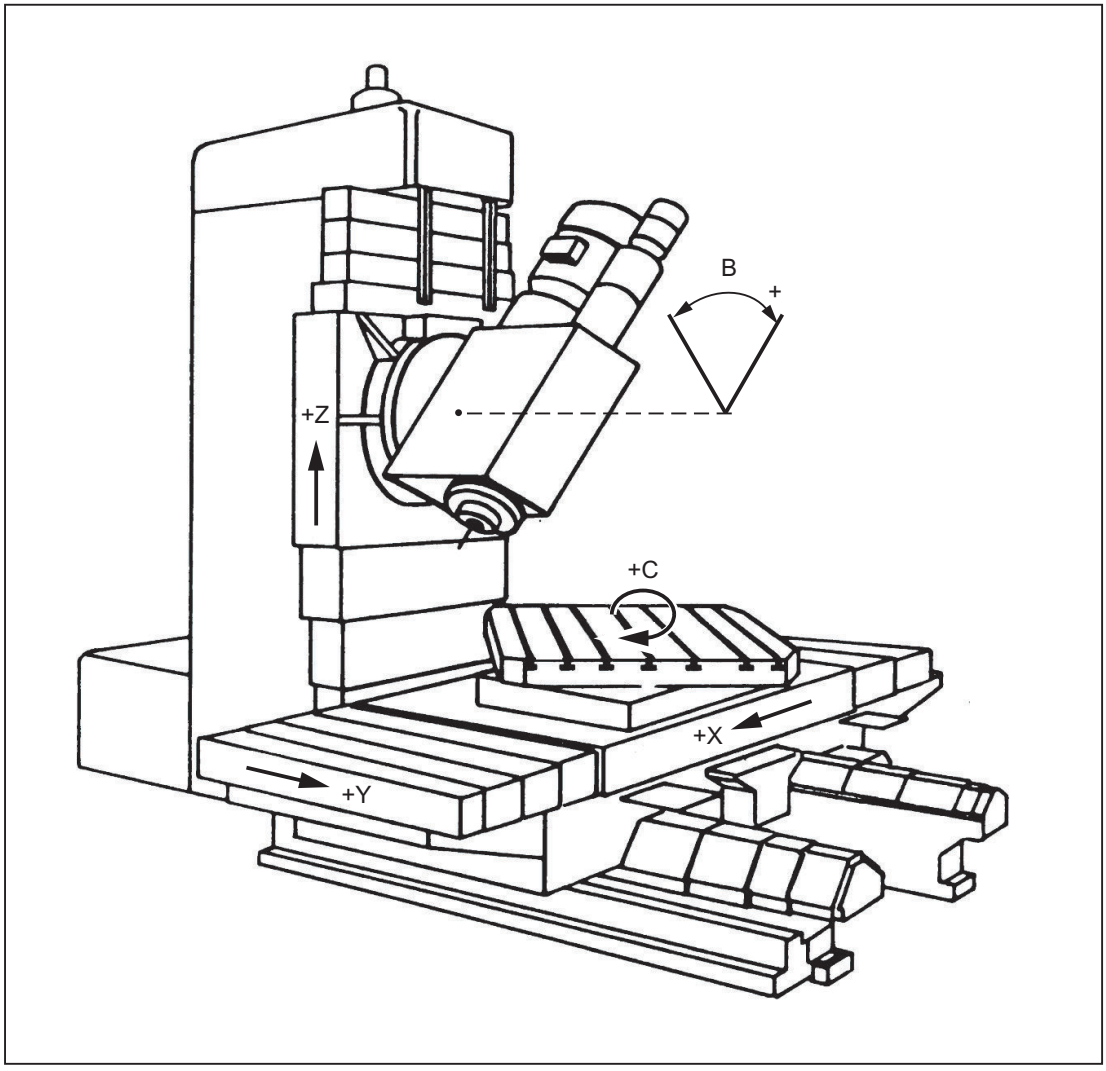


图 11-4 一台直角坐标机床的机床轴 X、Y、Z、B、S

应用

机床轴可以是：

- 几何轴 X、Y、Z
- 定向轴 A、B、C

- 进料轴
- 刀具转塔
- 用于刀库的轴
- 用于移动机械手的轴
- 顶尖套筒
- 用于盘式机械手的轴
- 等

11.2.3 通道轴

含义

每根几何轴和每根辅助轴均被指定给一个通道。 几何轴和辅助轴总是在所归属于的通道中运行。

11.2.4 几何轴

含义

三根几何轴总是构成一个虚拟的直角坐标系。

通过使用框架（偏移、旋转、比例缩放、镜像）可将工件坐标系（WCS）的几何轴映射至 BCS。

应用

几何轴用于编写工件几何数据（轮廓）。

平面选择 G17、G18 和 G19（DIN 66217）总是以三根几何轴为基准。 因此将三根几何轴命名为 X、Y 和 Z 会很便利。

11.2.5 辅助轴

含义

与几何轴不同，辅助轴之间未定义几何关联。

说明

几何轴之间的关联则为精确定义，以直角坐标系为形式。

辅助轴是基本坐标系（BCS）的组成部分。 通过使用框架（偏移、比例缩放、镜像）可将工件坐标系（WCS）的辅助轴映射至 BCS。

应用

典型的辅助轴有：

- 回转轴
- 刀具刀库轴
- 刀具转塔轴
- 进料轴

11.2.6 轨迹轴

含义

轨迹轴的特点是一同进行插补（一个通道的所有轨迹轴共用一个轨迹插补器）。

一个通道的所有轨迹轴具有共同的加速、恒速运行和减速阶段。

在地址 F 下编写的进给率（轨迹进给率）适用于所有在程序段中编程的轨迹轴，但有以下特例：

- 编写的轴通过 FGROUP 指令定义为不决定轨迹速度的轴。
- 通过 POS 或 POSA 指令编写的轴，具有独立的进给率（轴插补器）。

应用

轨迹轴用于以编写的轮廓加工工件。

11.2.7 定位轴

含义

定位轴的特点是独立进行插补（每根定位轴均有一个专属的轴插补器）。 每根定位轴均具有独立的进给率和加速特性曲线。 可编写定位轴作为对轨迹轴的补充（也可编写在同一程序段中）。 轨迹轴的插补（轨迹插补器）不会受定位轴影响。 轨迹轴和单独的定位轴不必同时到达程序段终点。

通过 POS 和 POSA 指令编写定位轴和定义程序段切换标准：

- POS
  轨迹轴和定位轴到达程序段终点时，进行程序段切换。
- POSA
  轨迹轴到达程序段终点时，进行程序段切换。 定位轴超越程序段界限运行至程序段终点。

并行定位轴相比定位轴有以下区别：

- 只从 PLC 获取程序段终点。
- 可在任意时间点（非程序段交界处）启动。
- 对运行中的零件程序无影响。

应用

典型的定位轴有：

- 用于工件传输的进料轴
- 刀库/刀具转塔

参考

更多相关信息请见“P3：SINUMERIK 840D sl 的 PLC 基本程序 (页 945)”和“S1：主轴 (页 1401)”章节。

文档：

- 功能手册之扩展功能分册；定位轴（P2）
- 功能手册之特殊功能分册；龙门轴（G1）
- 功能手册之特殊功能分册；轴耦合和 ESR（M3）

- 功能手册之扩展功能分册；往复（P5）
- 功能手册之同步动作分册

11.2.8 主处理轴

含义

主处理轴是指通过主处理进行插补的轴。

该插补可通过以下方式启动：

- 从同步动作启动
  （作为指令轴，基于一个逐段、模态或静态同步动作相关的事件）
- 由 PLC 通过特殊功能块在 PLC 基本程序中启动
  （作为并行定位轴，或称为 PLC 轴）
- 通过设定数据或从零件程序启动
  （作为异步或程序段同步往复轴）

控制

通过主处理插补的轴会对以下情形进行响应：

- NC 停止
- 报警处理
- 程序控制
- 程序结束
- 复位

说明

程序末尾的特性不同。 轴运动不必总是在程序末尾完成，因此可超出程序末尾进行。

应用

可在主处理中解除特定轴与由 NC 程序运行触发的通道特性的耦合，并通过 PLC 控制该轴。 这些轴同样在主处理中插补，且其特性与通道运行及程序运行无关。

这样一来便可通过 NC 控制由 PLC 控制的轴。 这涉及到下列动作：

- 终止轴运行（等同于删除剩余行程）
- 停止或中断轴
- 继续运行轴（恢复运行）
- 将轴恢复至初始状态

11.2.9 同步轴

含义

同步轴为轨迹轴的一部分，在计算轨迹速度时不用作参考。其和轨迹轴一同进行插补（一个通道的所有轨迹轴和同步轴共用一个轨迹插补器）。

一个通道的所有轨迹轴和同步轴具有共同的加速、恒速运行和减速阶段。

在地址 F 下编写的进给率（轨迹进给率）适用于所有在程序段中编程的轨迹轴，但不适用于同步轴。

对于编写的行程，同步轴所需的运行时间与轨迹轴相同。

FGROUP 指令

通过 FGROUP 指令可定义将轴作为决定进给率的**轨迹轴**（计算轨迹速度时作为参考）或**同步轴**（计算轨迹速度时不作为参考）。

示例

程序代码	注释
N05 G00 G94 G90 M3 S1000 X0 Y0 Z0	;
N10 FGROUP(X,Y)	; 轴 X 和 Y 为轨迹轴 轴 Z 为同步轴
N20 G01 X100 Y100 F1000	; 编写的进给率 1000 mm/min 轴 X 的进给率 = 707 mm/min 轴 Y 的进给率 = 707 mm/min
N30 FGROUP (X)	; 轴 X 为轨迹轴 轴 Y 为同步轴
N20 X200 Y150	; 编写的进给率 1000 mm/min 轴 X 的进给率 = 1000 mm/min 轴 Y 的进给率设为 500 mm/min，因为只须运行一半行程。

说明

在 FGROUP 指令中必须使用通道名称。

这通过以下机床数据定义：

MD20080 $MC_AXCONF_CHANAX_NAME_TAB（通道中的通道轴名称）

应用

在螺旋线插补中，可选择通过 FGROUP 设置：

- 编写的进给率是否在轨迹上生效
  （编写的 3 根轴均为轨迹轴）
- 编写的进给率是否在圆弧上生效
  （2 根轴为轨迹轴，进给轴为同步轴）

11.2.10 轴配置

几何轴、辅助轴、通道轴、机床轴和驱动之间的对应关系

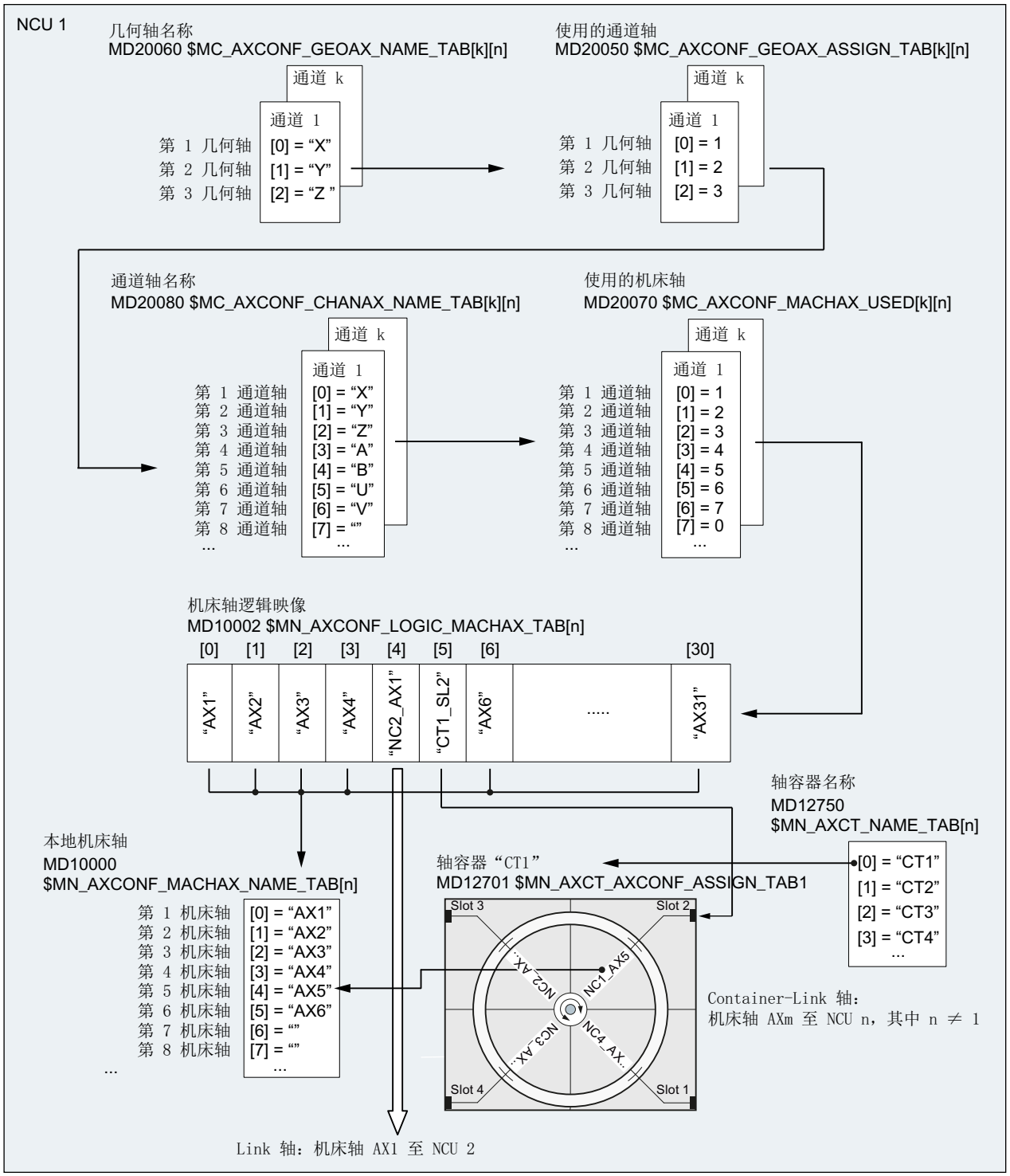


图 11-5　　轴分配

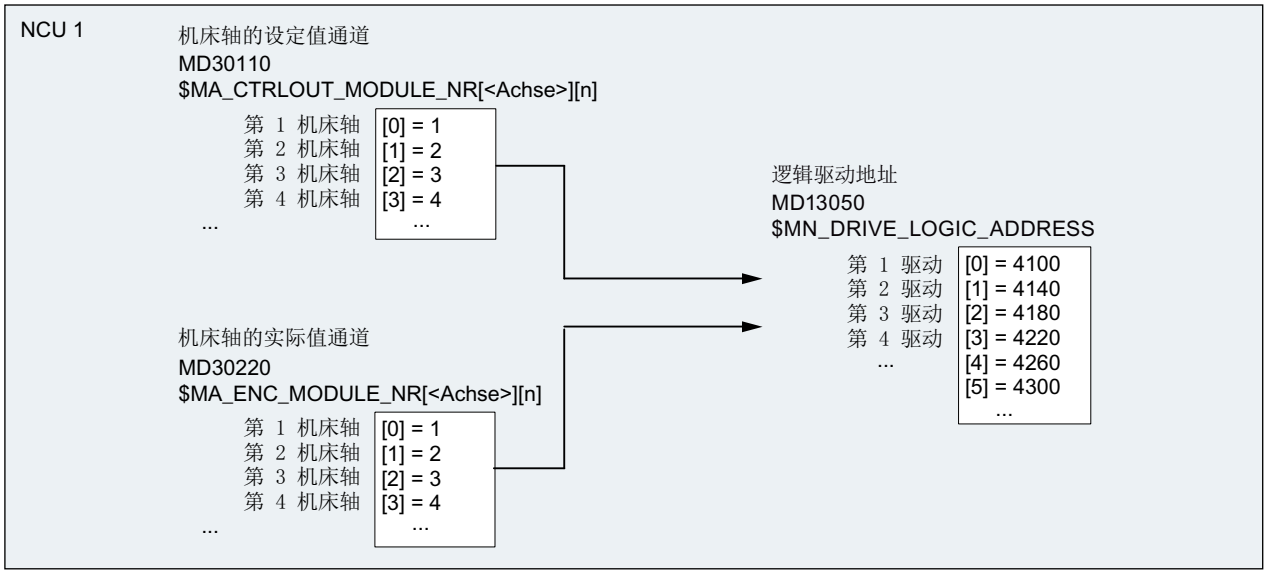


图 11-6　　驱动分配

边界条件

- 用户定义轴名称中，开头的零会被忽略：
  MD10000 `$MN_AXCONF_MACHAX_NAME_TAB[0] = X01，等同于 X1
- 通道轴必须按升序无空隙地指定为几何轴。
- 所有非几何轴的通道轴均为辅助轴。

通道轴空隙

通常情形下，必须通过 MD20070 $MC_AXCONF_MACHAX_USED 为机床数据 MD20080 $MC_AXCONF_CHANAX_NAME_TAB 中定义的每根通道轴指定一根机床轴。

为了简化对机床轴数量不同的机床系列的调试，也可定义未指定机床轴的通道轴。这样一来，通道轴升序排列时可能会存在空隙。

对通道轴空隙的许可必须显性使能：

MD11640 $MN_ENABLE_CHAN_AX_GAP = 1

若未使能，那么机床数据 MD20070 $MC_AXCONF_MACHAX_USED 中的 0 值结束将其他通道轴指定给机床轴。

示例

未向通道轴 B 指定机床轴。

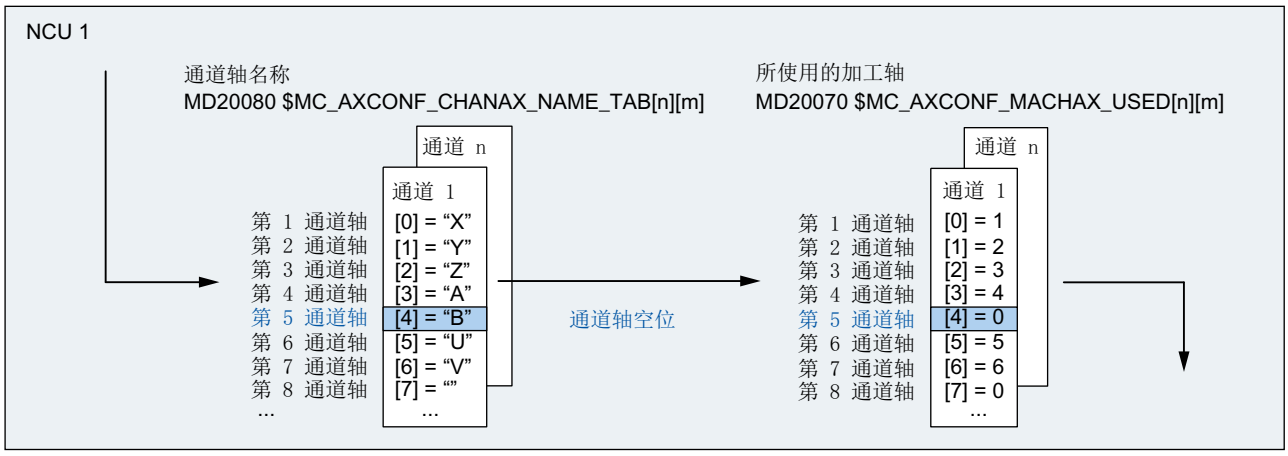


图 11-7　　含通道轴空隙的轴配置（节选）

边界条件

- 系统会视数量和通道轴索引将未指定机床轴的通道轴（通道轴空隙）像指定了机床轴的普通通道轴一样处理。
- 若将未指定机床轴的通道轴（通道轴空隙）定义为几何轴，那么此操作会被拒绝，并发出报警。

11.2.11　Link 轴

含义

Link 轴是不由实际连接的 NCU，而由其他 NCU 运行的机床轴。 为此，在运行 NCU 的逻辑机床轴映射的机床数据中记录的不是本地机床轴的名称，而是该轴实际连接的 NCU 的 NCU 名称和机床轴名称。

例如需通过 NCU1 运行 NCU2 的机床轴 AX1：

- NCU1： MD10002 $MN_AXCONF_LOGIC_MACHAX_TAB[n] = NC2_AX1

前提条件

使用 Link 轴的前提条件是：所涉及的 NCU 必须通过 Link 通讯连接。 对 Link 轴功能和 Link 通讯的详细说明请见：

文档：
功能手册之扩展功能分册；多操作面板配备多 NCU，分布式系统（B3）

11.3 零点和参考点

11.3.1 工作区域中的参考点

零点和参考点

机床的中性位置通过坐标轴和机床的构造特性得出。 坐标系零点通过在机床上的中性位置定义一个适宜的参考点来确定。

坐标系（MCS、BCS、BZS、SZS、WCS）的位置通过零点定义。

零点		参考点	
	**M =** 机床零点		**R =** 参考点
	**W =** 工件零点		**T =** 刀架参考点

机床零点 M

使用机床零点 M 可以确定机床坐标系 MCS。所有其他参考点都以机床零点为基准。

工件零点 W

以机床零点 W 为基准的工件零点可以用来确定工件坐标系。 编写的零件程序段在工件坐标系 WCS 中执行。

参考点 R

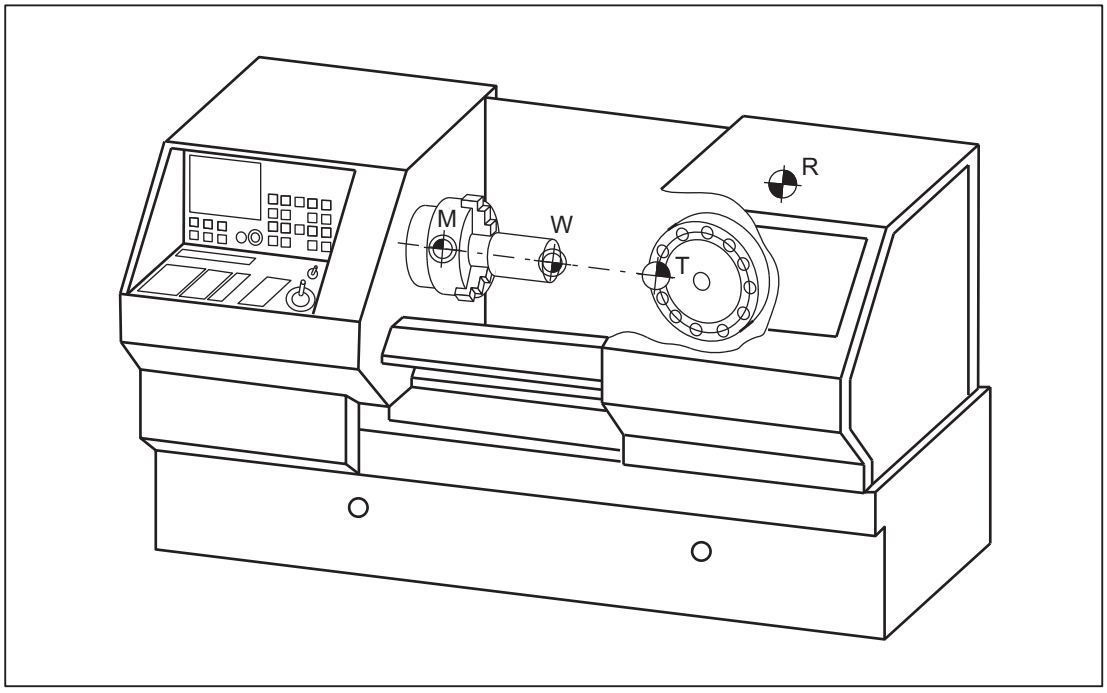
参考点 R 的位置通过凸轮开关设定。 其可用于校准路径测量系统。

使用增量测量编码器时，必须在每次接通控制系统后执行回参考点。 之后该控制系统才可使用测量系统运行，并将所有位置值传输至坐标系。

刀架参考点 T

刀架参考点 T 位于刀架安装处。 通过输入刀具长度，控制系统可以计算出刀尖（TCP，Tool Center Position，刀具中心位置）至刀架参考点的距离。

示例：车床上的零点和参考点



11.3.2 坐标系和参考点的位置

接通控制系统

使用增量测量编码器时，每次接通控制系统后必须执行回参考点，确保控制系统能将所有位置值传输至坐标系。

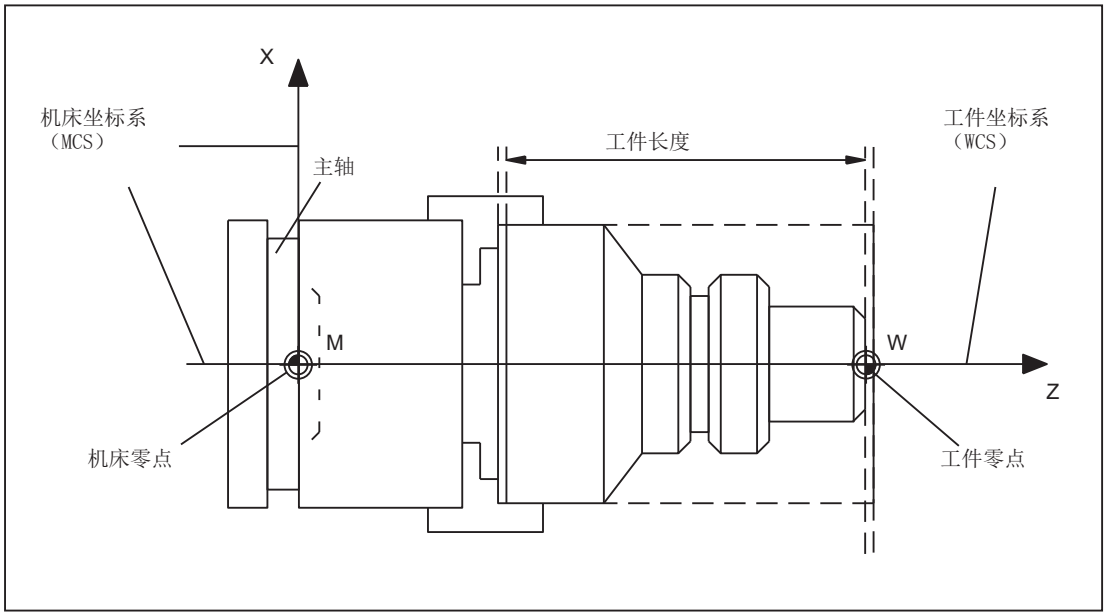


图 11-8　以机床零点 M 和工件零点 W 为基准的坐标系位置

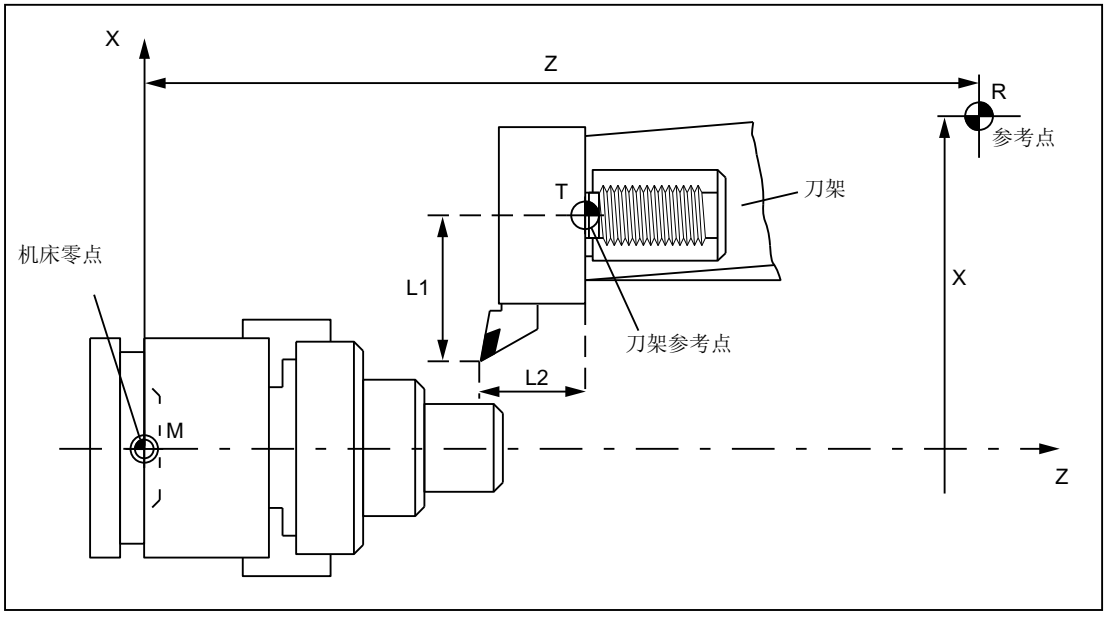


图 11-9　参考点相对机床零点的位置

11.4 坐标系

11.4.1 概述

定义

根据 DIN 66217，在机床的编程中使用直角（笛卡尔）坐标系。可通过“右手定则”确定坐标轴的正方向。

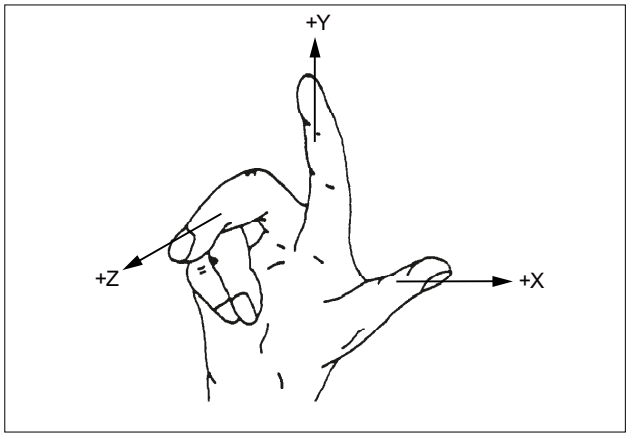


图 11-10　右手定则

编程采用的坐标系以工件为基准。编程不受刀具或者工件移动的影响。编程时始终假定工件静止，而刀具相对于工件坐标系发生位移。

通过“右手定则”也可确定回转轴的正（右旋）旋转方向。若右手拇指指向坐标轴（线性轴）的正方向，则手指指向相应回转轴的正旋转方向。

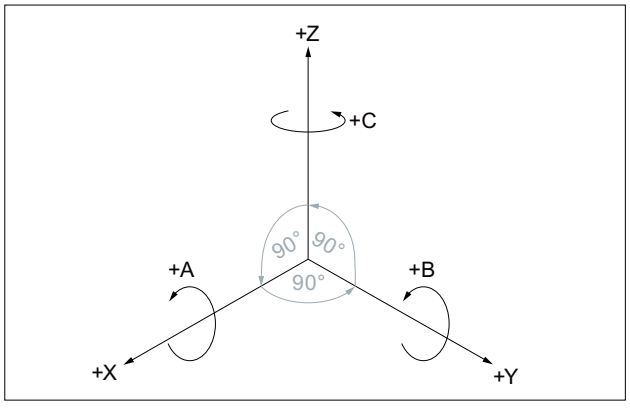


图 11-11　右旋的笛卡尔直角坐标系

坐标系

为机床定义的坐标系包括：

坐标系	缩写	注释
Workpiece **C**oordinate **S**ystem，工件坐标系	WCS	在 WCS 中编写几何轴的运行，用于加工工件。
Settable **Z**ero **S**ystem，可设定的零点坐标系	SZS	Koordinatentransformation über Frames:WKS ⇒ ENS
Basic **Z**ero **S**ystem，基本零点坐标系	BZS	通过框架进行坐标转换：SZS ⇒ BZS
Basic **C**oordinate **S**ystem，基本坐标系	BCS	通过框架进行坐标转换：BZS ⇒ BCS
Maschine **C**oordinate **S**ystem，机床坐标系	MCS	通过框架和运动转换进行坐标转换：BCS ⇒ MCS 在 WCS 中编写的几何轴运行通过框架和运动转换映射至 MCS 的机床轴。

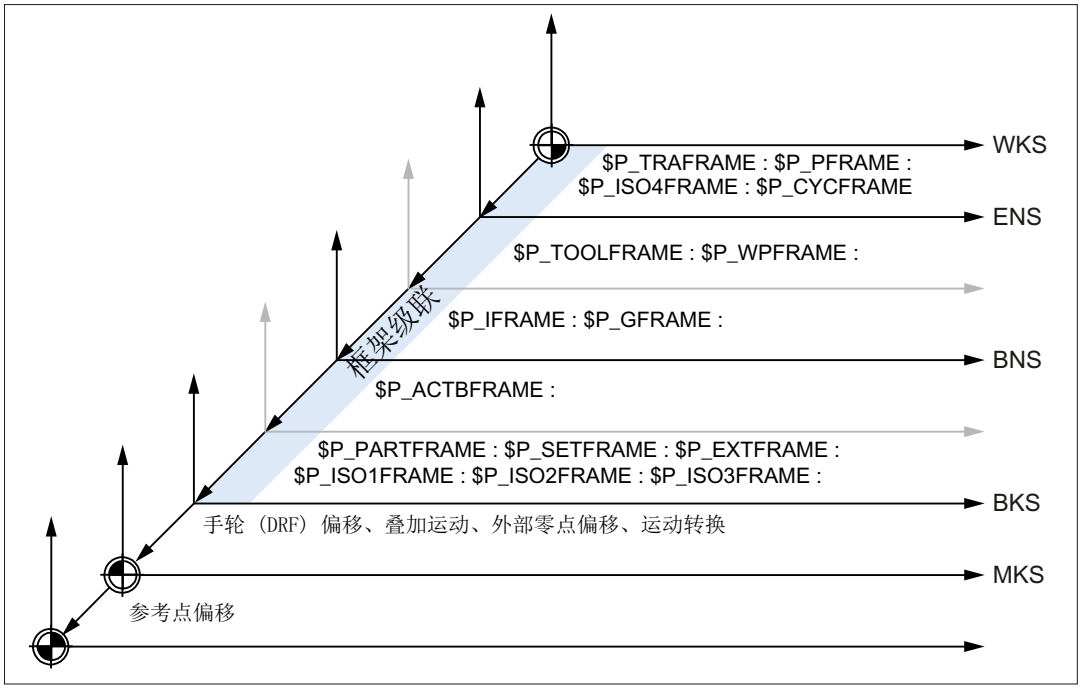


图 11-12　坐标系、框架和运动转换

11.4.2 机床坐标系（MCS）

机床坐标系（MCS）

机床坐标系（MCS）由所有实际存在的机床轴构成。

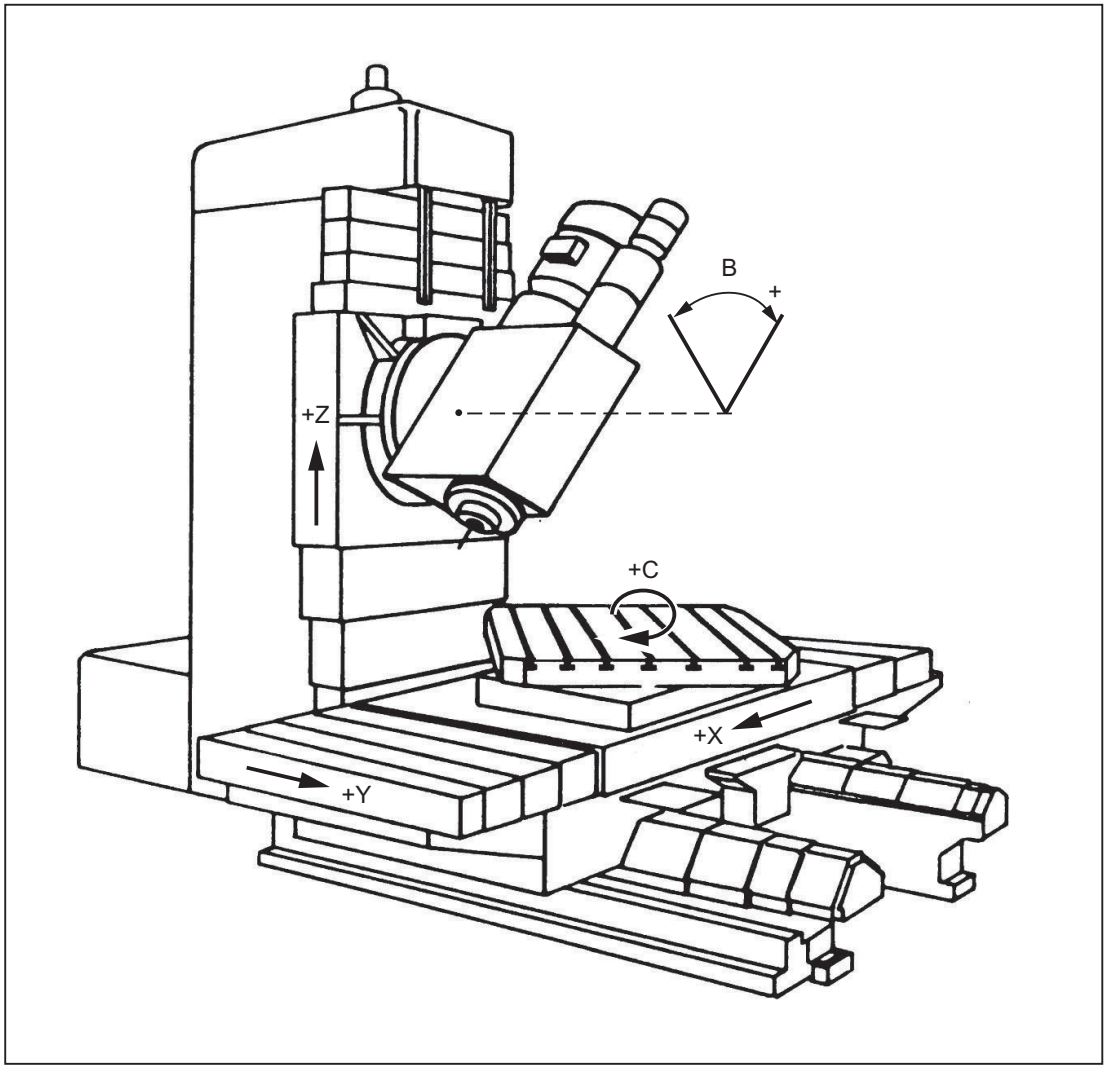


图 11-13　机床轴 X、Y、Z、B、C 构成的 MCS（5 轴铣床）

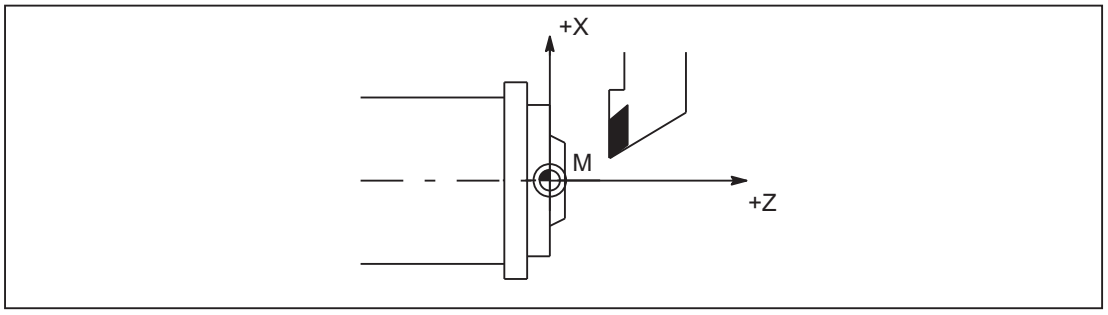


图 11-14　机床轴 X、Z 构成的 MCS（车床）

轴预设偏移

通过“预设定偏移（PRESETON）”功能，可以机床坐标系（机床零点）重新设置控制系统参考点。

⚠ 小心

编码器校准丢失

在预设定偏移后，对应的机床轴切换至“未回参考点”状态！ 在使用绝对值编码器时，这意味着编码器校准将丢失并须重新执行（例如通过激光干涉仪测量）。 因此不建议将 PRESETON 功能与绝对值编码器配合使用。

说明

建议只对不需回参考点的机床轴使用此功能。

若需恢复原始的机床坐标系，必须使机床轴重新回参考点，例如使用 G74（回参考点运行）。

启用预设定偏移时，机床轴不发生运动。

文档

- 编程手册 基本原理
  章节：“补充指令”>“回参考点（G74）”
- 编程手册 工作准备
  章节：“坐标转换（FRAMES）”>“预设定偏移（PRESETON）”

11.4.2.1 参考点状态的实际值设置和损失（PRESETON）

功能

程序 PRESETON() 在机床坐标系 (MCS) 为一个或多个轴设置新的实际值。该值等于轴的 MCS 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，随后的过程取决于以下机床数据中参数设置的跨通道取轴特性：

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
值域： 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（MCS）
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

系统变量

$AC_PRESET

轴专用系统变量 $AC_PRESET 提供了从当前移动的 MCS 的零点到机床轴回参考点后初始 MCS_0 的零点的矢量。

$AC_PRESET<轴> = $AC_PRESET<轴> + “轴在 MCS 中的当前实际位置” - “PRESETON 实际位置”

通过系统变量可再次取消零点偏移：

PRESETON(<轴>, $VA_IM + $AC_PRESET[<轴>]); “轴在 MCS 中的当前实际值” + “偏移”

示例

程序代码

```
N10 G1 X10 F5000
N20 PRESETON(X, $AA_IM[X]+70) ; 实际值 = 10 + 70 = 80 =>
                              ; $AC_PRESET = $AC_PRESET - 70
```

边界条件

不可以使用 PRESETON 的轴

- 正在运行的轨迹轴
- 正在运行的定位轴
- 主轴模式下正在运行的指令轴
- 正在运行的并行定位轴 (FC18)
- 参与转换的轴

- 有以下一个或多个 Safety Integrated 功能生效的轴：
  - 安全软件限位开关（SE）：
    MD36901 $MA_SAFE_FUNCTION_ENABLE[<SafeAxis>]，位 1 = 1
  - 安全软件凸轮，安全凸轮信号（SN）：
    MD36901 $MA_SAFE_FUNCTION_ENABLE[<SafeAxis>]，位 8 ... 15 = 1
- 摆动轴
- 端面齿轴
- 龙门轴组中的同步轴
- 用于从零件程序接近参考点 (G74) 的轴生效
- 转速/转矩耦合中的从动轴

几何轴

- PRESETON 可在静态几何轴上使用，只要此时通道中没有另一根几何轴在同步运行。
- PRESETON 可在静态几何轴上使用，即使此时通道中有另一根几何轴在“中性轴”状态下同步运行或作为指令轴同步运行。

程序代码	注释
N10 G0 X0 Y0	; X, Y:几何轴
N15 RELEASE(Y)	; 中性轴
N20 PRESETON(Y,20)	; Y 在 MCS 中的实际位置 = 20
N30 G0 X40	; 几何轴 X 运行
N40 M30	

示例：另一几何轴（X）在“**中性轴**”状态下同步运行

程序代码	注释
N10 G0 X0 Y0	; X, Y:几何轴
N20 POS[X]=40 FA[X]=1000	; 指令轴 X
N30 DO PRESETON(Y,20)	; Y 在 MCS 中的实际位置 = 20
N40 M30	

示例：几何轴（X）同时作为**指令轴**运行

PLC 控制的轴

PRESETON 可在 PLC 控制的轴上使用（根据其当前类型）。

主轴状态

下表列出了 PRESETON 在同步动作中对主轴的响应：

NC 程序中的 PRESETON			
主轴运行方式	**运行状态**	**分配给 NC 程序**	**主运行轴**
转速控制模式	运动中	报警 22324	报警 22324
	静止	+	+
定位模式 SPOS	运动	-	+
	静止	+	+
通过 SPOSA 定位	运动中	报警 10610	-
进给轴模式	运动中	-	+
	静止	+	+
+：允许 -：不允许			

轴耦合

- 引导轴：由 PRESETON 引起的引导轴位置的骤变不会出现在跟随轴中。耦合保持不变。
- 跟随轴：通过 PRESETON 只能控制跟随轴的叠加位置分量。

龙门轴组

如果在龙门轴组中的引导轴上使用 PRESETON，系统则会在龙门轴组的所有同步轴中执行零点偏移。

分度轴

PRESETON 可在分度轴上使用。

软件限位开关、工作区域限制、保护区

如果在零点偏移后 PRESETON 使轴位置超出了规定的范围，那么只有在尝试运行轴时，系统才会显示报警。

带计算的程序段搜索

程序段搜索期间，系统会收集 PRESETON 指令并通过 NC 启动继续执行 NC 程序。

NC/PLC 接口信号

因实际位置发生变化，需重新确定 NC/PLC 接口信号的状态。

示例：固定点位置

- 设置的固定点位置：MD30600 $MA_FIX_POINT_POS[0...3] = <固定点位置 1...4>
- NC/PLC 接口信号：DB31, ... DBX75.3 - 5（已手动运行至固定点）

如果轴位于固定点位置上（准停公差范围内），系统会置位对应的 NC/PLC 接口信号。如果通过 PRESETON 将实际值设为另一个准停公差范围外的值，系统会复位接口信号。

DRF 偏移

通过 PRESETON 删除轴的 DRF 偏移。

叠加运动 $AA_OFF

同步动作中的叠加运动 $AA_OFF 不受 PRESETON 影响。

在线刀具补偿 FTOC

同步动作中的生效在线刀具补偿 FTOC 在 PRESETON 后继续生效。

轴专用补偿

轴专用补偿在 PRESETON 后继续生效。

运行方式 JOG

PRESETON 只可在静态轴上使用。

运行方式 JOG，机床功能 REF

不可以使用 PRESETON。

11.4.2.2 参考点状态的实际值设置，无损失（PRESETONS）

功能

程序 PRESETONS() 在机床坐标系 (MCS) 为一个或多个轴设置新的实际值。该值等于轴的 MCS 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，随后的过程取决于跨通道取轴特性的轴专用配置：

MD30552 $MA_AUTO_GET_TYPE

回参考点状态

通过 PRESETONS 在机床坐标系 (MCS) 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（原始信号编码器）

- **回参考点模式**

  PRESETONS 仅适用有效测量系统的以下回参考点模式：

  – MD34200 $MA_ENC_REFP_MODE[<测量系统>] = 0（无法回参考点）
  – MD34200 $MA_ENC_REFP_MODE[<测量系统>] = 1（增量、旋转或线性测量系统的回参考点：编码器信号上的零脉冲）

调试

轴专用机床数据

必须针对具体轴激活无回参考点状态损失的实际值设定（PRESETONS）：

MD30455 $MA_MISC_FUNCTION_MASK，位 9 = 1

说明

取消 PRESETON

激活“**无**回参考点状态损失的实际值设定 PRESETONS”功能会将“**有**回参考点状态损失的实际值设定 PRESETON”功能取消。两个功能相互排斥。

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
	值域：	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 (MCS)
该值以生效的单位制（英制/公制）为基准
考虑到有效的直径编程 (DIAMON)
类型： REAL

系统变量

$AC_PRESET

轴专用系统变量 $AC_PRESET 提供了从当前移动的 MCS 的零点到机床轴回参考点后初始 MCS_0 的零点的矢量。

$AC_PRESET<轴> = $AC_PRESET<轴> + “轴在 MCS 中的当前实际位置” - “PRESETONS 实际位置”

通过系统变量可再次取消零点偏移：

PRESETONS(<轴>, $VA_IM + $AC_PRESET[<轴>]); “轴在 MCS 中的当前实际值” + “偏移”

示例

轴 X 的 MCS 的零点偏移有 70 个单位。

通过 PRESETONS 将编程的轴 X（指令轴）的最终位置转换到 MCS 中。

程序代码

```
N10 G1 X10 F5000
N20 PRESETONS(X, $AA_IM[X]+70) ; 实际值 = 10 + 70 = 80 =>
                               ; $AC_PRESET = $AC_PRESET - 70
```

边界条件

不可以使用 PRESETONS 的轴

- 正在运行的定位轴
- 主轴模式下正在运行的指令轴
- 正在运行的并行定位轴 (FC18)
- 参与转换的轴
- 正在运行的轨迹轴
- 摆动轴

- 有以下一个或多个 Safety Integrated 功能生效的轴：
  - 安全软件限位开关（SE）：
    MD36901 $MA_SAFE_FUNCTION_ENABLE[<SafeAxis>]，位 1 = 1
  - 安全软件凸轮，安全凸轮信号（SN）：
    MD36901 $MA_SAFE_FUNCTION_ENABLE[<SafeAxis>]，位 8 ... 15 = 1
- 端面齿轴
- 龙门轴组中的同步轴
- 用于从零件程序接近参考点 (G74) 的轴生效
- 转速/转矩耦合中的从动轴

几何轴

- PRESETONS 可在静态几何轴上使用，只要此时通道中没有另一根几何轴在同步运行。
- PRESETONS 可在静态几何轴上使用，即使此时通道中有另一根几何轴在“中性轴”状态下同步运行或作为指令轴同步运行。
  示例：另一根几何轴 (X) 在“**中性轴**”状态下同步运行

程序代码	注释
N10 G0 X0 Y0	; X, Y:几何轴
N15 RELEASE(Y) 1)	; 中性轴
N20 PRESETONS(Y,20)	; Y 在 MCS 中的实际位置 = 20
N30 G0 X40	; 几何轴 X 运行
N40 M30	
1) 提示 在使能同步动作动作分量中的轴时不能确保及时使能。 N20 ID=1 WHEN 20.0 < $AA_IM[X] DO RELEASE(Y) PRESETONS(Y,20) ; 不建议！	

示例：另一几何轴（X）同时作为**指令轴**运行

程序代码	注释
N10 G0 X0 Y0	; X, Y:几何轴
N20 POS[X]=40 FA[X]=1000	; 指令轴 X
N30 PRESETONS(Y,20)	; Y 在 MCS 中的实际位置 = 20
N40 M30	

PLC 控制的轴

PRESETONS 可在 PLC 控制的轴上使用（根据其当前类型）。

主轴状态

下表列出了 PRESETONS 在同步动作中对主轴的响应：

NC 程序中的 PRESETONS			
主轴运行方式	**运行状态**	**分配给 NC 程序**	**主运行轴**
转速控制模式	运动中	报警 22324	报警 22324
	静止	+	+
定位模式 SPOS	运动	-	+
	静止	+	+
通过 SPOSA 定位	运动中	报警 10610	-
进给轴模式	运动中	-	+
	静止	+	+
+：允许 -：不允许			

轴耦合

- 引导轴：由 PRESETONS 引起的引导轴位置的骤变不会出现在跟随轴中。耦合保持不变。
- 跟随轴：通过 PRESETONS 只能控制跟随轴的叠加位置分量。

龙门轴组

如果在龙门轴组中的引导轴上使用 PRESETONS，系统则会在龙门轴组的所有同步轴中执行零点偏移。

分度轴

PRESETONS 可在分度轴上使用。

软件限位开关、工作区域限制、保护区

如果在零点偏移后 PRESETONS 使轴位置超出了规定的范围，那么只有在尝试运行轴时，系统才会显示报警。

带计算的程序段搜索

程序段搜索期间，系统会收集 PRESETONS 指令并通过 NC 启动继续执行 NC 程序。

NC/PLC 接口信号

因实际位置发生变化，需重新确定 NC/PLC 接口信号的状态。

示例：固定点位置

- 设置的固定点位置：MD30600 $MA_FIX_POINT_POS[0...3] = <固定点位置 1...4>
- NC/PLC 接口信号：DB31, ... DBX75.3 - 5（已手动运行至固定点）

如果轴位于固定点位置上（准停公差范围内），系统会置位对应的 NC/PLC 接口信号。如果通过 PRESETONS 将实际值设为另一个准停公差范围外的值，系统会复位接口信号。

DRF 偏移

通过 PRESETONS 删除轴的 DRF 偏移。

叠加运动 $AA_OFF

同步动作中的叠加运动 $AA_OFF 不受 PRESETONS 影响。

在线刀具补偿 FTOC

同步动作中的生效在线刀具补偿 FTOC 在 PRESETONS 后继续生效。

轴专用补偿

轴专用补偿在 PRESETONS 后继续生效。

运行方式 JOG

PRESETONS 只可在静态轴上使用。

运行方式 JOG，机床功能 REF

不可以使用 PRESETONS。

同步动作

若在一个针对轴的同步动作中包含多个 PRESETONS 指令，则系统只执行从左向右的序列中的最后一个。

示例：

程序代码

```
N10 ID=1 WHEN TRUE DO PRESETONS(X,40) PRESETONS(X,39) PRESETONS(X,38)
；含义等同于：
N10 ID=1 WHEN TRUE DO PRESETONS(X,38)
```

11.4.3 基本坐标系（BCS）

基本坐标系（BCS）

基本坐标系（BCS）由三条相互垂直的轴（几何轴）、以及其他没有几何关联的轴（辅助轴）构成。

无运动转换的机床

将无运动转换（例如 TRANSMIT/端面转换，5 轴转换和最多三根机床轴）的 BCS 映射至 MCS 时，BCS 和 MCS 总是重合。

在此类机床上，机床轴与几何轴可以使用相同的名称。

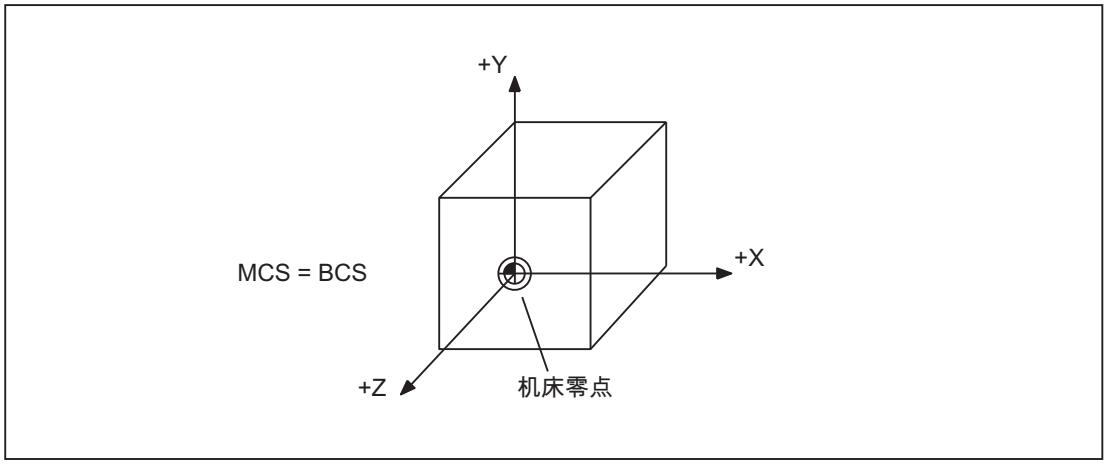


图 11-15 MCS=不进行运动转换的 BCS

带运动转换的机床

将带运动转换（例如 TRANSMIT/端面转换，5 轴转换或多于三根轴）的 BCS 映射至 MCS 时，BCS 和 MCS 不重合。

在此类机床上，机床轴与几何轴必须使用不同的名称。

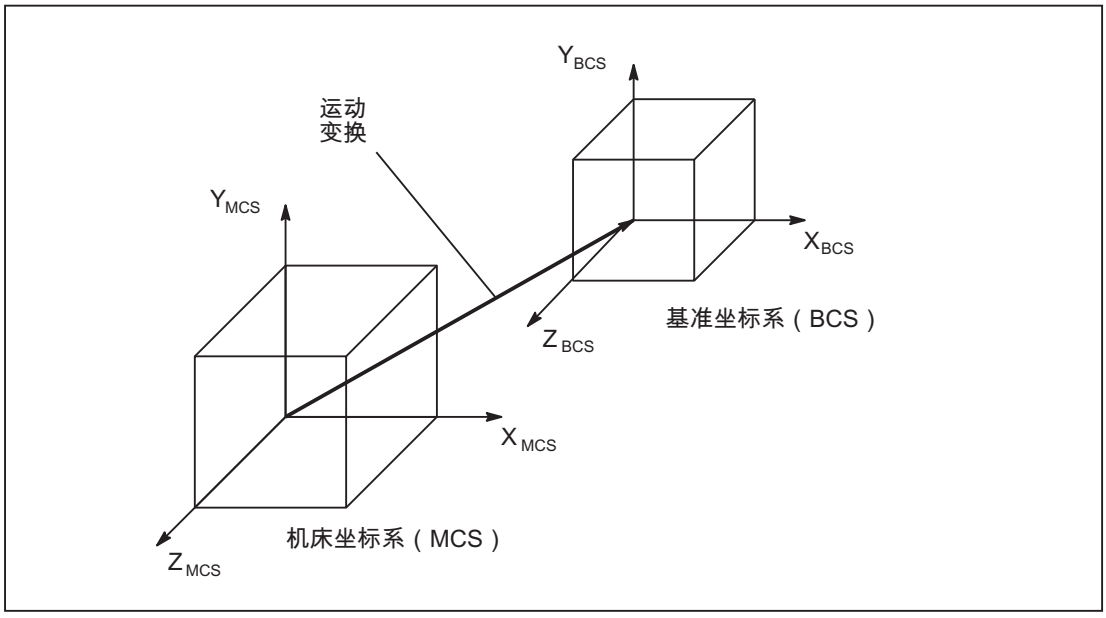


图 11-16　MCS 和 BCS 间的运动转换

机床运动系统

工件编程总是在一个二维或者三维的直角坐标系（WCS）中进行。但加工工件时经常需要使用带回转轴或非垂直布局的直线轴的机床。为了将 WCS 中编程的（直角）坐标投影到实际的加工轴运动中，需要用到运动转换。

文档：

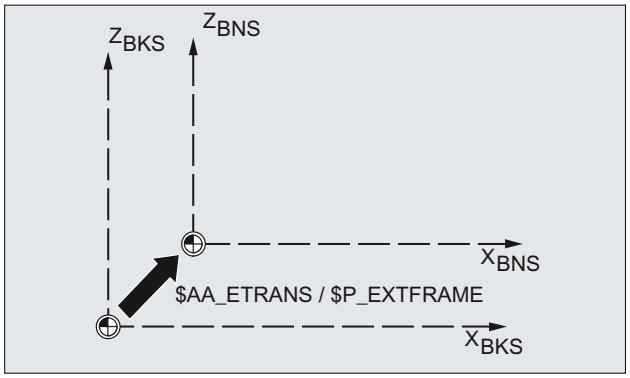
功能手册之特殊功能分册； 3 至 5 轴转换 （F2）

功能手册之扩展功能分册；运动转换（M1）

11.4.4 附加补偿

11.4.4.1 外部零点偏移

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 程序直接写入 $AA_ETRANS[<轴>] 。 解锁，从 $AA_ETRANS[<轴>] 移出零点偏移至下一个可能的运行程序段：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) ● 生效框架的粗偏移 $P_EXTFRAME[<轴>] = $AA_ETRANS[<轴>] ● 数据管理框架的粗偏移 $P_EXTFR[<轴>] = $AA_ETRANS[<轴>] ● 重新计算有效的总框架 $P_ACTFRAME ● 偏移移出至编程轴 ● 继续执行中断的运行或 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>：	偏移值

NC/PLC 接口信号

激活外部零点偏移：

DB31, ... DBX3.0 = 0 → 1 ⇒ $P_EXTFRAME[<轴>] = $P_EXTFR[<轴>] = $AA_ETRANS[<轴>]

抑制：外部零点偏移

- 通过指令 SUPA，可在程序段执行期间抑制“外部零点偏移”。
- 可通过 G74（回参考点运行）指令及“回参考点”运行方式下的相应操作在回参考点期间抑制“外部零点偏移”。
- 使用 G74 时，即 “AUTO” 或 “MDI” 操作方式下，之前生效的“外部零点偏移”会在程序段中的下一次运行时自动重新生效。
- 在从“回参考点”模式进行运行方式切换后，必须为回参考点的轴置位用于重新激活的 NC/PLC 接口信号。

11.4.4.2 DRF 偏移

借助 DRF 偏移功能，可**通过手轮**在基本坐标系中为几何轴和辅助轴设置附加的增量零点偏移。

系统变量

DRF 偏移可通过轴专用系统变量读取：

$AC_DRF[<轴>]

文档

功能手册之扩展功能分册；手动运行和通过手轮进行手动运行（H1）；章节：DRF 偏移

11.4.4.3 叠加运动

机床数据

- 通过轴专用机床数据设置轴的叠加运动：
  36750 $MA_AA_OFF_MODE，位 <n> = <值>

位	值	含义
0	0	编程带**绝对位置**的 $AA_OFF
	1	编程带**增量行程**的 $AA_OFF
1	0	复位时撤销叠加运动
	1	复位后保留叠加运动
2	0	在 JOG 运行方式下，**不**移出基于 $AA_OFF 的叠加运动。
	1	在 JOG 运行方式下，移出基于 $AA_OFF 的叠加运动。
3	0	NC 停止时中断叠加运动。
	1	NC 停止时**不**中断叠加运动。

系统变量

- 通过轴专用系统变量设定轴的绝对位置或增量行程。绝对位置或增量行程之后会与生效的运动同时启动或运行。
  $AA_OFF[<轴>] = <值>
- 通过轴专用系统变量可以读取集成的轴叠加运动值。
  $AA_OFF_VAL[<轴>]

11.4.4.4 复位特性

可通过机床数据设置生效框架 $P_EXTFRAME 和数据管理框架 $P_EXTFR 的复位和上电特性：

机床数据

- 通过以下机床数据设置通道中生效的外部零点偏移的系统框架 $P_EXTFRAME 的复位特性：
- MD24006 $MC_CHSFRAME_RESET_MASK，位 1 = <值>

值	含义 **生效的外部零点偏移的系统框架在通道/程序结束复位后：**
0	未生效
1	生效

- 通过以下机床数据设置数据管理外部零点偏移的系统框架 $P_EXTFR 的上电特性：MD24008 $MC_CHSFRAME_POWERON_MASK，位 1 = <值>

值	含义 **外部零点偏移的系统框架 $P_EXTFR 在上电时：**
0	未删除
1	删除

11.4.5 基本零点坐标系（BZS）

基本零点坐标系（BZS）

基本零点坐标系（BZS）由基本坐标系通过基本偏移后得到。

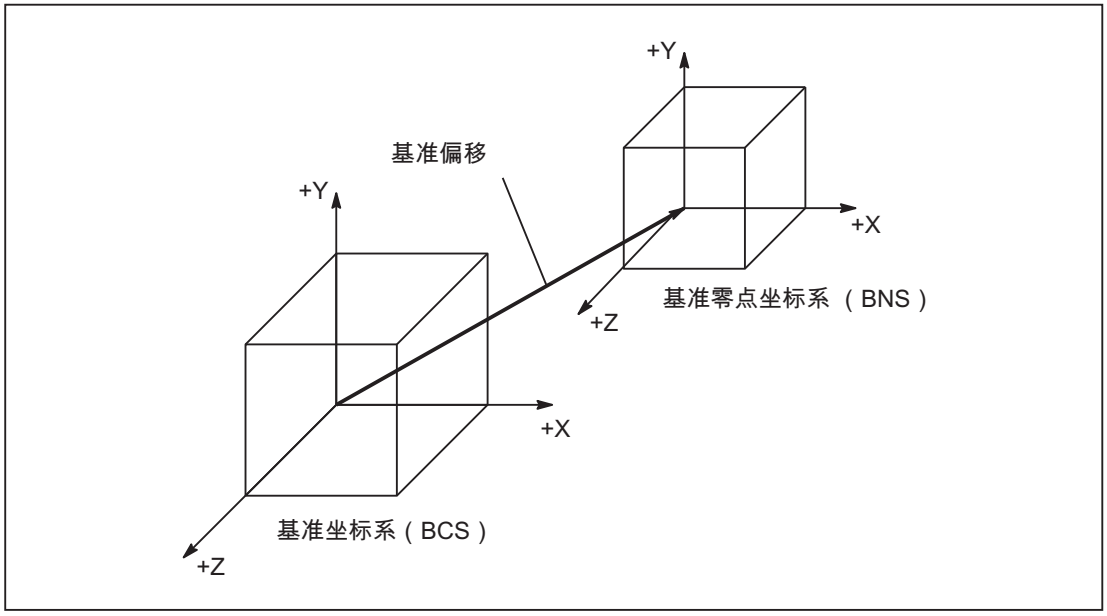


图 11-17 BCS 和 BZS 间的基本偏移

基本偏移

基本偏移表示 BCS 和 BZS 之间的坐标转换。其例如可用于确定托盘零点。

基本偏移由以下部分组成：

- 外部零点偏移
- DRF 偏移
- 叠加运动

- 级联的系统框架
- 级联的基本框架

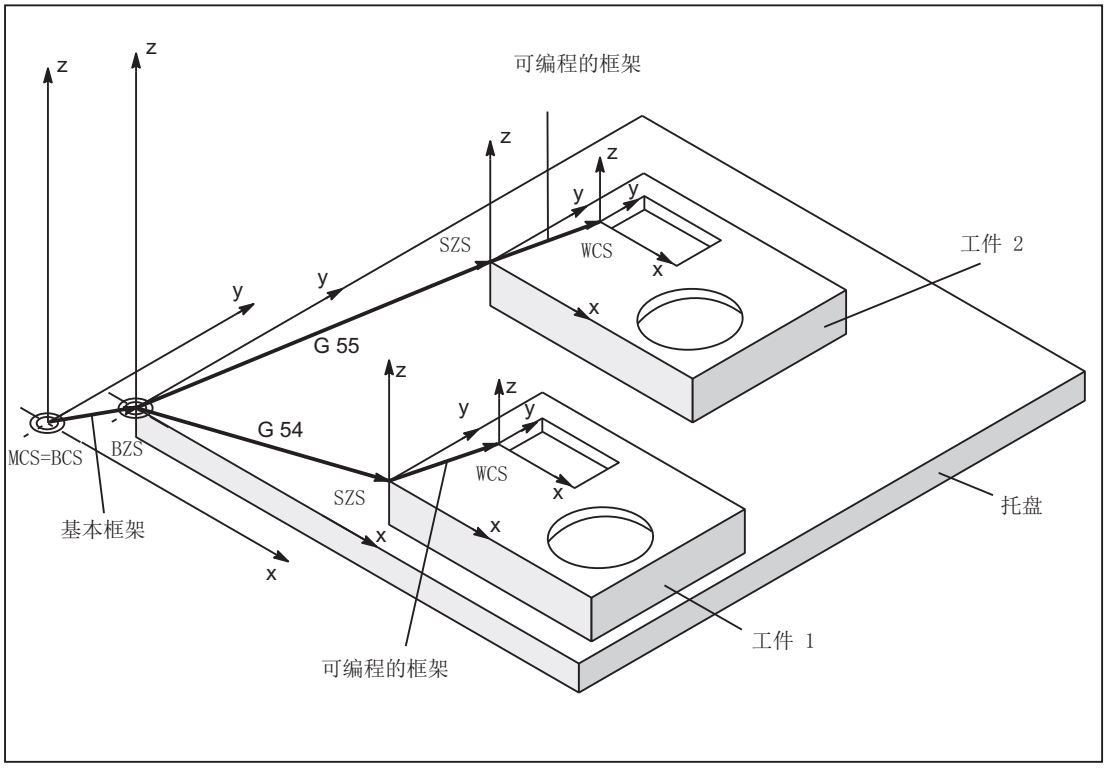


图 11-18　基本偏移的应用示例

可实现的调整：

- 用户可通过零件程序、操作和 PLC 修改基本偏移。
- 若需使基本偏移立即生效，可通过 PLC 使用 FC9 启动一个 ASUB，由该 ASUB 执行相应的 G 指令。

说明

对机床制造商的建议

针对自有应用，请使用自基本偏移 3 起的偏移。

基本偏移 1 和 2 设计用于“实际值设置”和“外部零点偏移”。

11.4.6 可设定的零点坐标系（SZS）

可设定的零点坐标系（SZS）

从工件坐标系（WCS）角度看来，“可设定的零点坐标系”（SZS）是带可编程框架的 WCS。 工件零点通过可设定框架 G54 ... G599 定义。

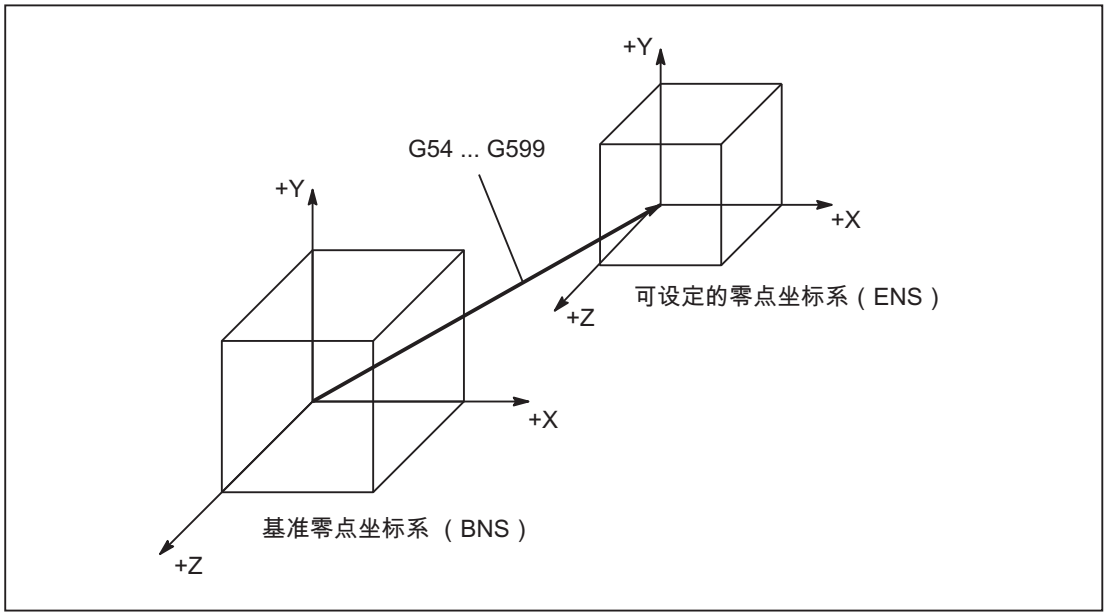


图 11-19 BZS 和 SZS 之间的可设定框架 G54 ... G599

可编程的偏移通过“可设定的零点坐标系”生效。 所有可编程偏移均以“可设定的零点坐标系”为基准。

以 WCS 或 SZS 显示 WCS 实际值

在 HMI 操作界面上，轴的实际值可以机床坐标系（MCS）或工件坐标系（WCS）显示。 以 WCS 显示时，也可基于 SZS 显示实际值。 相应设置通过以下机床数据进行：

MD9424 $MM_MA_COORDINATE_SYSTEM（用于实际值显示的坐标系）

值	含义
0	基于 WCS 显示实际值
1	基于 SZS 显示实际值

说明

当前坐标系的显示

“基于 SZS 显示实际值”生效时，HMI 操作界面上仍会将 WCS 作为实际值显示的基准坐标系加以显示。

示例

基于 WCS 或 SZS 显示实际值

代码（节选）：	实际值显示： 轴 X（WCS）	实际值显示： 轴 X（SZS）
`N10 X100`	100	100
`N20 X0`	0	0
`N30 $P_PFRAME = CTRANS(X,10)`	0	10
`N40 X100`	100	110

11.4.7 工件坐标系（WCS）

工件坐标系 WCS

工件坐标系（WCS）是编程的基础。

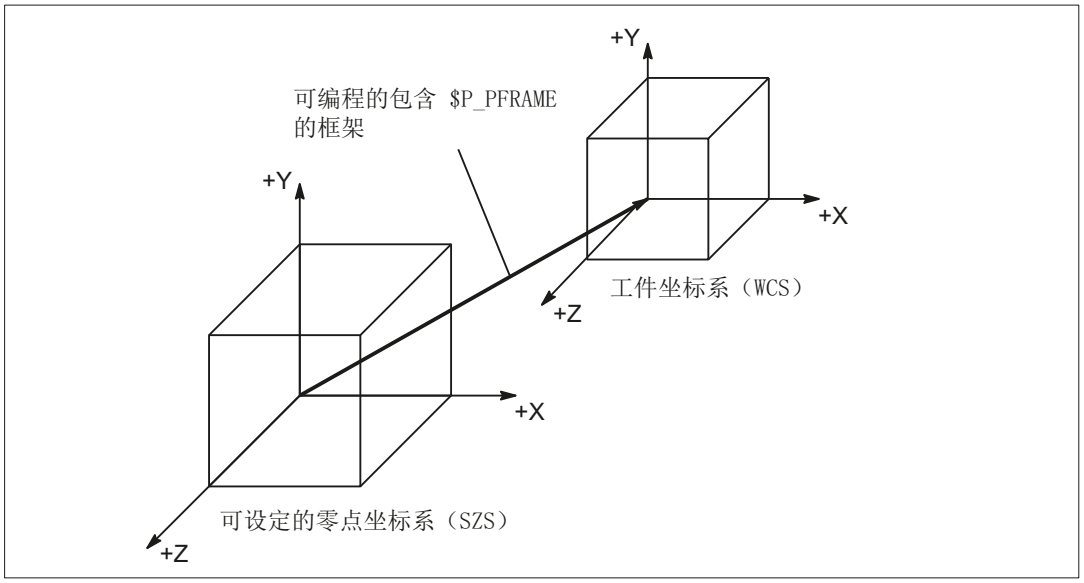


图 11-20 SZS 和 WCS 间的可编程框架

11.5 框架

11.5.1 框架类型

框架是包含轴的偏移（TRANS）、精偏（FINE）、旋转（ROT）、镜像（MIRROR）和比例缩放（SCALE）数值的数据结构。

框架激活时，系统会依据框架值通过既定的计算规则对所涉及的轴执行静态坐标转换。

轴框架

轴框架中包含了一根轴的框架值。

几何轴 X 的轴框架数据结构示例：

轴	TRANS	FINE	ROT	MIRROR	SCALE
X	10.0	0.1	**0.0**	0	1

通道专用框架

通道专用框架包含了针对所有通道轴（几何轴、辅助轴和机床轴）的框架值。

旋转（ROT）只应用于几何轴。

通道专用框架仅在定义了框架的通道中生效。

通道专用框架的数据结构示例：

- 几何轴：X, Y, Z
- 辅助轴 A
- 机床轴：AX1

轴	TRANS	FINE	ROT	MIRROR	SCALE
X	10.0	0.1	**0.0**	0	1
Y	0.0	0.0	**0.0**	1	1
Z	0.0	0.0	**45.0**	0	1
A	2.0	0.1	0.0	0	2
AX1	0.0	0.0	0.0	0	0

全局框架

全局框架包含针对所有机床轴的框架值。

全局框架在 NC 的所有通道中生效。

全局框架的数据结构示例：

- 机床轴：AX1, ... AX5

轴	TRANS	FINE	ROT	MIRROR	SCALE
AX1	10.0	0.1	-	0	1
AX2	0.0	0.0	-	1	1
AX3	0.0	0.0	-	0	1

轴	TRANS	FINE	ROT	MIRROR	SCALE
AX4	2.0	0.1	-	0	2
AX5	0.0	0.0	-	1	1

11.5.2 框架分量

11.5.2.1 偏移

编程

偏移或粗偏可通过以下指令编写：

- 数据管理框架 $P_UIFR 的示例
  - 总框架：$P_UIFR[<n>] = CTRANS(<K1>,<V1>[,<K2>,<V2>][,<K3>,<V3>])
    其中 Km = x、y 或 z 坐标，Vm = 偏移 m
  - 框架分量：$P_UIFR[<n>,<K>,TR] = <V>
    其中 K = x、y 或 z 坐标，V = 偏移
- 可编程框架的示例
  - TRANS <K1> <V1> [<K2> <V2>][<K3> <V3>]
    其中 Km = x、y 或 z 坐标，Vm = 偏移 m

程序示例：

程序代码	注释
$P_UIFR[1] = CTRANS(X,10,Y,10)	总框架
$P_UIFR[1,X,TR] = 10	框架分量
TRANS X=10 Y=10	可编程的框架

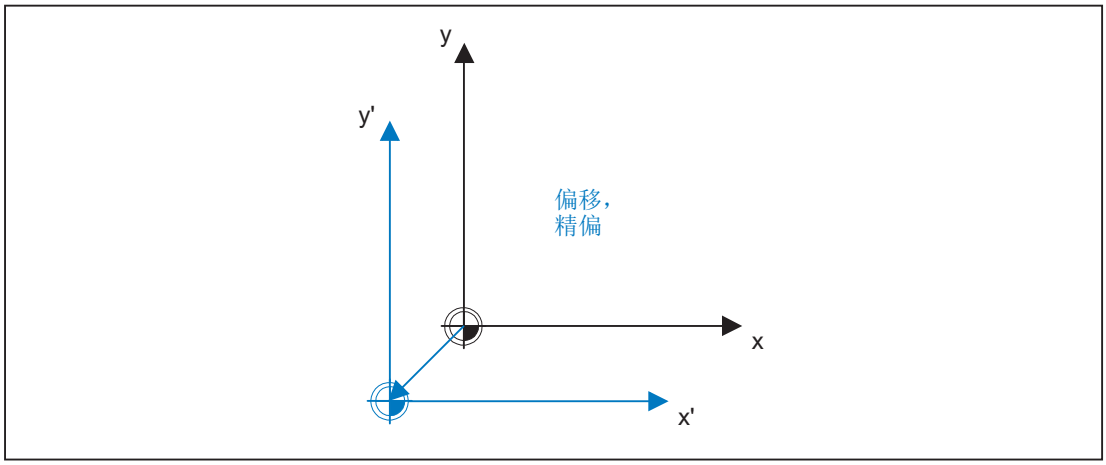


图 11-21　Z 方向上的偏移

11.5.2.2　精偏

参数设置

精偏通过以下机床数据使能：

MD18600 $MN_MM_FRAME_FINE_TRANS = <值>

值	含义
0	无法输入或编写精偏。
1	可通过操作或程序为可设定框架、基本框架和可编程框架启用精偏。

编程

偏移或粗偏可通过以下指令编写：

- 数据管理框架 $P_UIFR 的示例
  - 总框架：$P_UIFR[<n>] = CFINE(<K1>,<V1>[,<K2>,<V2>][,<K3>,<V3>])
    其中 Km = x、y 或 z 坐标，Vm = 偏移 m
  - 框架分量：$P_UIFR[<n>,<K>,FI] = <V>
    其中 K = x、y 或 z 坐标，V = 偏移
- 可编程框架的示例
  - TRANS <K1> <V1> [<K2> <V2>][<K3> <V3>]
    其中 Km = x、y 或 z 坐标，Vm = 偏移 m

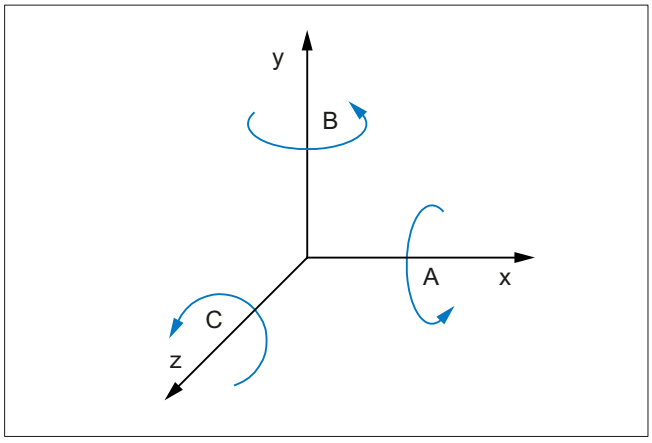
编程示例：

程序代码	注释
`$P_UIFR[1] = CTRANS(X,10,Y,10)`	总框架
`$P_UIFR[1,X,TR] = 10`	框架分量
`TRANS X=10 Y=10`	可编程的框架

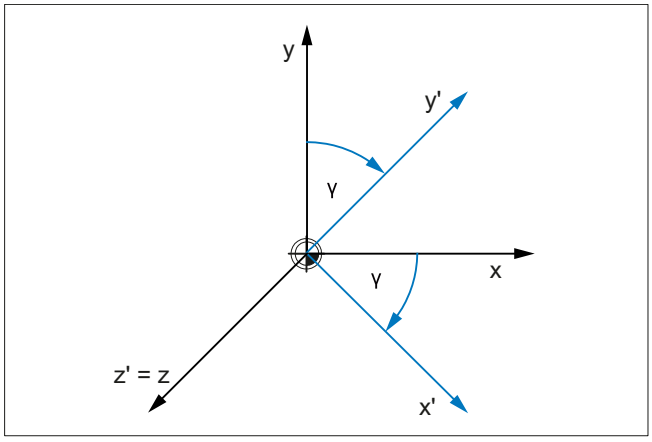
11.5.2.3 旋转：概述（只适用于几何轴）

功能

围绕坐标轴的旋转方向由通过 x、y 和 z 轴构成的右手直角坐标系决定。 从坐标轴正方向观察，顺时针旋转时旋转方向为正。 A、B 和 C 代表轴平行于坐标轴的旋转。



下图显示了围绕 z 旋转 γ = -45°后坐标系 x'、y' 和 z' 的新位置。



旋转顺序的参数设置

通过以下机床数据，可在编写了超过一个旋转角时设置围绕哪根坐标轴，以及以何种顺序执行旋转：

MD10600 $MN_FRAME_ANGLE_INPUT_MODE = <值>

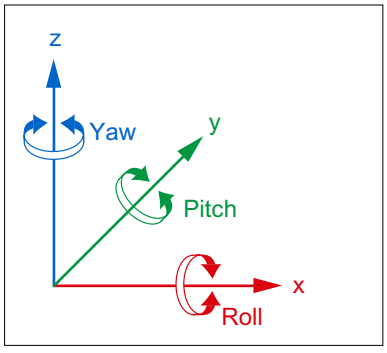
值	含义
1	zy'x" 转换中的欧拉角（RPY 角）
2	zx'z" 转换中的欧拉角

说明

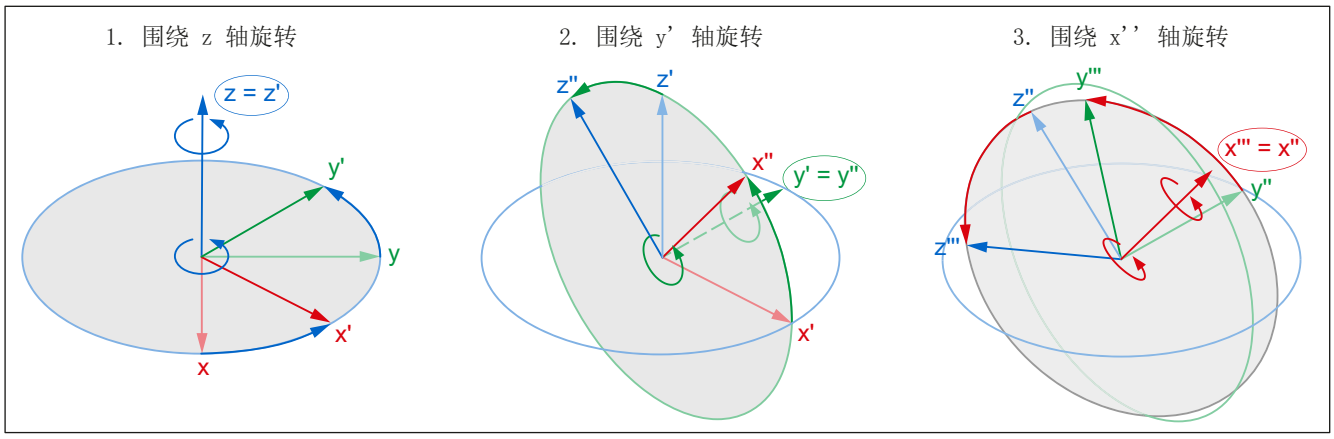
由于历史原因，系统支持 zx'z" 转换中的欧拉角。 但是强烈建议只使用 zy'x" 转换中的欧拉角（RPY 角，参见“通过欧拉角旋转：ZY'X" 转换（RPY 角） (页 815)”章节）。

11.5.2.4 通过欧拉角旋转：ZY'X" 转换（RPY 角）

ZY'X" 转换中的欧拉角也称作 RPY 角。RPY 来源于英语：Roll（横滚）、Pitch（俯仰）和 Yaw（航向）



在 ZY'X" 转换中以如下次序旋转：



取值范围

使用 PRY 角时，仅当编写的值位于以下取值范围内时，才能确保倒推值的唯一性：

-180	≤	x	≤	180
-90	<	y	<	90
-180	≤	z	≤	180

编程：Schreiben aller Drehkkomponenten

在通过 CROT、ROT 或 AROT 编写框架的旋转分量时，总是将所有旋转分量写入。未明确编写的旋转分量会被赋值为 0°。

句法

```
<Frame> = CROT([<1.GAx>,<Winkel>,][<2.GAx>,<Winkel>,]
[<3.GAx>,<Winkel>])
ROT [<1.GAx> <Winkel>] [<2.GAx> <Winkel>] [<3.GAx> <Winkel>]
AROT [<1.GAx> <Winkel>] [<2.GAx> <Winkel>] [<3.GAx> <Winkel>]
```

含义

CROT：	绝对旋转
<框架>：	任意生效框架或数据管理框架
ROT：	绝对旋转 基准框架：可编程框架 $P_PFRAME， 参考点：使用 G54 ... G57，G505 ... G599 设置的当前工件坐标系的零点

AROT：	附加旋转 基准框架：可编程框架 $P_PFRAME， 参考点：使用 G54 ... G57，G505 ... G599 设置的当前工件坐标系的零点
<几何轴 n>：	几何轴 n 的名称，围绕该轴以给定的角度旋转。对于未编程的几何轴，旋转角度隐性设为 0°。 几何轴和旋转轴的指定关系：

几何轴	旋转轴
第 1 几何轴	x''
第 2 几何轴	y'
第 3 几何轴	z

<角度>：	角度设定，以度为单位
[...]：	Die Angaben in eckigen Klammern sind optional.

编程：Schreiben einer Drehkomponente

在明确编写框架的旋转分量时，仅编写的旋转分量会被写入。未编写的分量保持不变。

句法
<框架>[<索引>,<几何轴>,RT] = <角度>

含义

<框架>：	任意生效框架或数据管理框架
<索引>：	框架的数组索引，例如 $P_UIFR[0 ... n]
<几何轴>：	几何轴的名称，围绕该轴以给定的角度旋转。
RT：	旋转“RoTation”的关键字
<角度>	角度设定，以度为单位

回读旋转分量

通常情形下，回读的框架旋转分量与编写的值相同：

编程	读回时的值		
	x, RT	y, RT	z, RT
<框架> = CROT(X,45,Y,30,Z,-20)	45	30	-20

值超出取值范围

若编写的值超出取值范围，则会映射至区域限值：

编程	读回时的值		
	x, RT	y, RT	z, RT
<框架> = CROT(X,190,Y,0,Z,-200)	-170	0	160

说明

建议在写入框架旋转分量时遵循给定的取值范围，从而能在回读旋转分量时重新获取相同的值。

万向节死锁（Gimbal-Lock）

万向节死锁（Gimbal-Lock）指的是无法从位置矢量倒推出旋转分量的唯一值这一几何问题。对于 RPY 角，旋转分量 y 的角度设置 = 90°时便会出现万向节死锁。在此情形下，控制系统会在写入后对旋转分量进行换算，从而使：

- 旋转分量 z = 旋转分量 z - 旋转分量 x
- 旋转分量 x = 0°
- 旋转分量 y = 90°

编程	读回时的值		
	x, RT	y, RT	z, RT
<框架> = CROT(X,30,Y,90,Z,40)	0	90	40 - 30 = 10

⚠ 小心

回读旋转分量 z 的不同数值

由于换算时间点不同，在写入**总框架**，或**数据管理框架**的**单个旋转分量**，或**生效框架**的**单个旋转分量**后，为旋转分量 z 回读的值有可能不同。

写入总框架和框架分量时的区别

在写入框架的旋转分量时，须区分两种情形：

1. 写入总框架：<框架> = CROT(X,a,Y,b,Z,c)
   写入总框架时，系统会在写入时间点立即进行换算。
2. 写入各旋转分量，例如围绕 X 旋转：<框架>[0,X,RT]= <a>
   写入单个旋转分量时，换算取决于框架的存储位置：
   - 数据管理框架
     对于数据管理框架，换算基于目前为止写入的旋转分量在框架激活的时间点进行。就旋转分量换算方面而言，写入单个旋转分量后数据管理框架的特性与写入总框架时相似。
   - 生效框架
     对于生效的框架，换算在写入旋转分量的时间点立即进行。

示例

- 写入总框架
  写入**总框架**后，在**每个程序段中**进行**换算**。

编程	读回时的值		
	x, RT	y, RT	z, RT
N10 <框架> = CROT(X,0,Y,90,Z,90)	0	90	90
N20 <框架> = CROT(X,90,Y,90)	0	90	**-90** 1)
N30 <框架> = CROT(X,90,Y,90,Z,90)	0	90	**0** 1)
1) 与写入**生效框架**的**单个旋转分量**相比，数值不同			

- 写入**数据管理框架**的各旋转分量
  在数据管理架激活时进行**换算**。例如在 N30 后的任意一个时间点。

编程	读回时的值		
	x, RT	y, RT	z, RT
N10 <数据管理框架>[0,X,RT] = 0 N20 <数据管理框架>[0,Y,RT] = 90 N30 <数据管理框架>[0,Z,RT] = 90	0	90	90
N10 <数据管理框架>[0,X,RT] = 90 N20 <数据管理框架>[0,Y,RT] = 90 N30 <数据管理框架>[0,Z,RT] = 0	0	90	**-90** 1)
N10 <数据管理框架>[0,X,RT] = 90 N20 <数据管理框架>[0,Y,RT] = 90 N30 <数据管理框架>[0,Z,RT] = 90	0	90	**0** 1)
1) 与写入**生效框架**的**各旋转分量**相比，数值不同			

- 写入**生效框架**的各旋转分量
  写入相应旋转分量时立刻进行可能需要的**换算**。
  保存的生效框架的输出值为：x = 0, y = 0, z = 0。

编程	读回时的值		
	x, RT	y, RT	z, RT
N10 <生效框架>[0,X,RT] = 0 N20 <生效框架>[0,Y,RT] = 90 N30 <生效框架>[0,Z,RT] = 90	0	90	90
N10 <生效框架>[0,X,RT] = 90 N20 <生效框架>[0,Y,RT] = 90 N30 <生效框架>[0,Z,RT] = 0	0	90	**0** 1)
N10 <生效框架>[0,X,RT] = 90 N20 <生效框架>[0,Y,RT] = 90 N30 <生效框架>[0,Z,RT] = 90	0	90	**90** 1)
1) 与写入**总框架**或写入**数据管理框架**的**单个旋转分量**相比，数值不同			

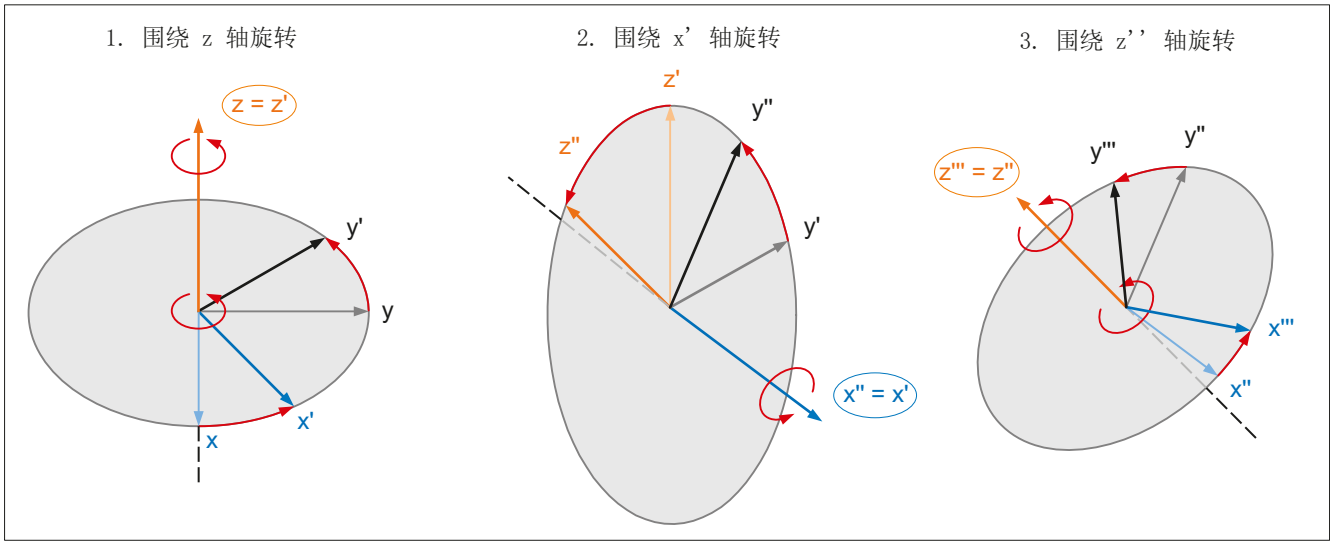
11.5.2.5 通过欧拉角旋转：ZX'Z" 转换

使用欧拉角以 z、x'、z" 的顺序进行旋转。

说明

使用建议

由于历史原因，系统支持 zx'z" 转换中的欧拉角。但是强烈建议只使用 zy'x" 转换中的欧拉角（RPY 角，参见“通过欧拉角旋转：ZY'X" 转换（RPY 角） (页 815)”章节）。



旋转轴和几何轴的指定关系

旋转轴	通道中的几何轴
z	第 3 几何轴
x'	第 1 几何轴
z"	第 3 几何轴

取值范围

仅当欧拉角的设定处于以下取值范围内时，才能确保倒推出唯一值：

```
  0    <=   x    <    180
-180   <=   y    <=   180
-180   <=   z    <=   180
```

设定值位于取值范围以外时，系统会基于限值进行模数换算。

说明

建议在写入框架旋转分量时遵循给定的取值范围，从而能在回读旋转分量时重新获取相同的值。

11.5.2.6 在任意平面中旋转

CRPL - Constant Rotation Plane（恒定旋转平面）

通过预定义功能“Constant Rotation Plane（恒定旋转平面）”，可为框架编写任意平面（G17、G18、G19）中的旋转，且无需指定几何轴的名称。 这样一来，在通道基于特殊机床配置只有两根几何轴的情形下，也可编写第三个平面中的旋转。

句法

```
CRPL(<旋转轴>,<旋转角度>)
```

含义

CRPL: 在任意平面中旋转

<旋转轴>: 围绕旋转的轴

类型： INT

值	含义
0	生效平面中的旋转

1	围绕 Z 轴旋转
2	围绕 Y 轴旋转
3	围绕 X 轴旋转

<旋转角度>： 旋转角度，以度为单位

类型： REAL

强烈建议遵循给出的取值范围。 若未遵循限值，则无法倒推出唯一值。处于限值外的角度设定不会被系统拒绝。

RPY 角：	X	-180 <= <旋转角度> <= 180
	Y	-90 <= <旋转角度> <= 90
	Z	-180 <= <旋转角度> <= 180
ZX'Z" 转换：	X	-180 <= <旋转角度> <= 180
	Y	0 <= <旋转角度> <= 180
	Z	-180 <= <旋转角度> <= 180

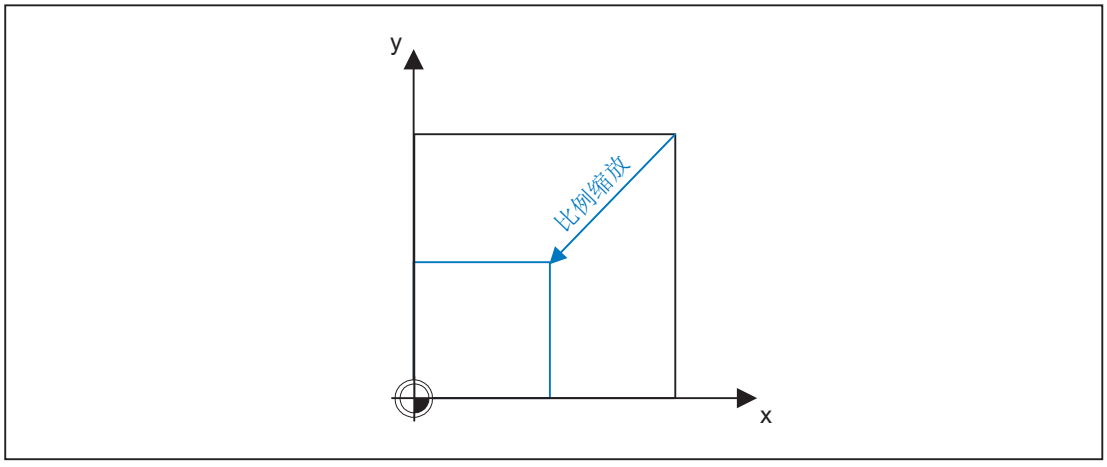
与框架级联

CRPL() 可与框架及 CTRANS()、CROT()、CMIRROR()、CSCALE()、CFINE() 等框架功能级联。

示例：

```
$P_PFRAME = $P_PFRAME : CRPL(0,30.0)
$P_PFRAME = CTRANS(X,10): CRPL(1.30.0)
$P_PFRAME = CROT(X,10): CRPL(2.30.0)
$P_PFRAME = CRPL(3,30.0) : CMIRROR(Y)
```

11.5.2.7 比例缩放

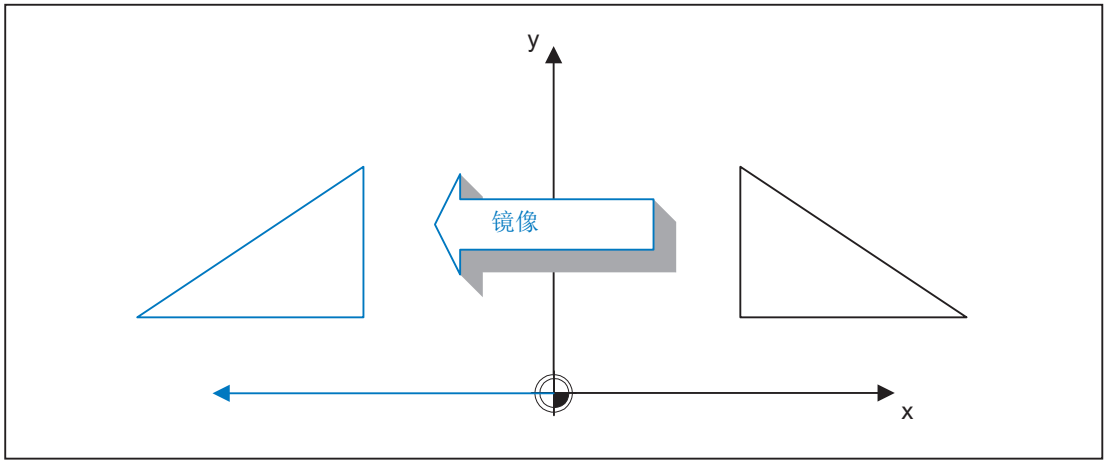


编程

通过以下程序指令编写比例缩放：

```
$P_UIFR[1] = CSCALE(x,1,y,1)
SCALE x = 1y = 1
$P_UIFR[1,x,sc] = 1
```

11.5.2.8 镜像



编程

通过以下程序指令编写镜像：

```
$P_UIFR[1] = CMIRROR(x,1,y,1)

MIRROR x = 1y = 1

$P_UIFR[1,x,mi] = 1
```

11.5.2.9 级联运算符

框架分量或总框架可通过级联运算符 (:) 组合成一个整体框架。

11.5.2.10 可编程的轴名称

在框架指令中可使用几何轴、通道轴和机床轴名称。 在通道专用框架中，编写的轴必须为系统已知。

SPI

编写框架指令时，轴名称的位置也可使用轴功能 SPI(<主轴编号>)。

此时 SPI(<主轴编号>) 为主轴相对通道轴的参考。
→ 参见 MD35000 $MA_SPIND_ASSIGN_TO_MACHAX[ ]（主轴指定为机床轴）

下列框架指令可通过 SPI(主轴编号) 编写：

```
CTRANS()

CFINE()

CMIRROR ()

CSCALE ()
```

一根主轴只能指定给一根回转轴。 因此 CROT(..) 功能不可通过 SPI() 编写，因为 CROT() 只支持几何轴。

在框架回译时，即便在零件程序中通过 SPI(..) 编写了轴名称，对于归属于主轴的轴，系统也总是输出其通道轴名称或机床轴名称。

例如主轴被指定给通道轴“A”，那么编程：

```
N10 $P_UIFR[1] = CTRANS(SPI(1),33.33,X,1):CSCALE(SPI(1),
33.33):CMIRROR(SPI(1))
```

回译时：

```
$P_UIFR[1]=CTRANS(X,1,A,33.33):CSCALE(A,33.33):CMIRROR(A)
```

若在一个框架指令中编写了主轴和指定的轴，则会触发报警 16420“轴 % 多次编写”。

示例：

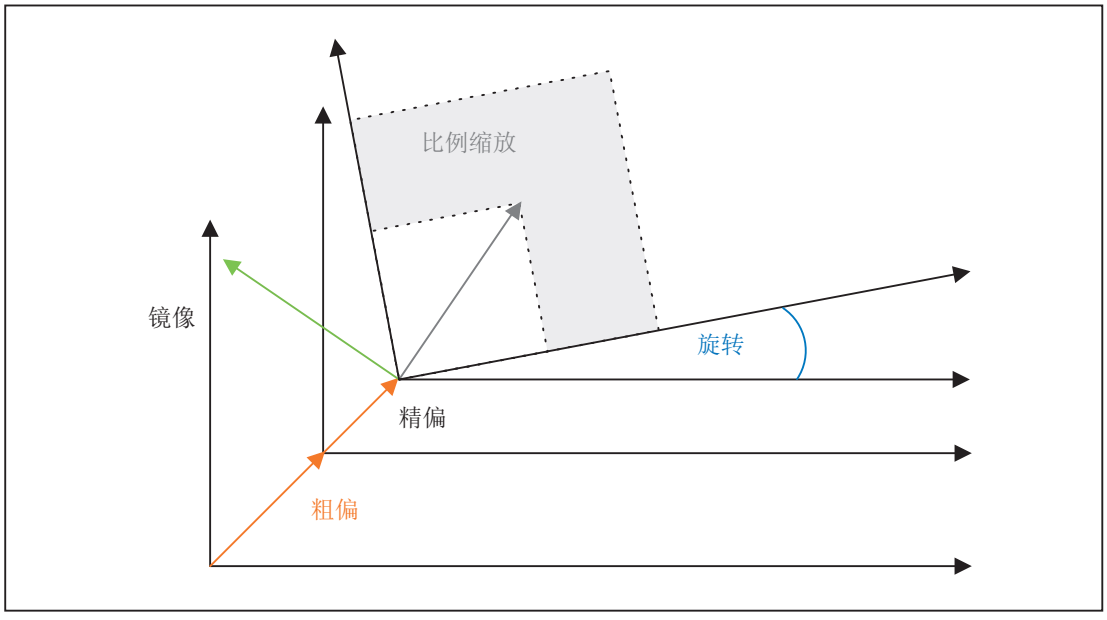
```
$P_UIFR[1] = CTRANS(SPI(1),33.33,X,1,A,44)
```

（主轴被指定给轴 A。）

编程示例

```
$P_PFRAME[SPI(1),TR]=22.22
$P_PFRAME=CTRANS(X,轴值,Y,轴值,SPI(1),轴值)
$P_PFRAME=CSCALE(X,标度,Y,标度,SPI(2),标度)
$P_PFRAME=CMIRROR(S1,Y,Z)
$P_UBFR=CTRANS(A,10):CFINE(SPI(1),0.1)
```

11.5.2.11 坐标转换



几何轴的坐标转换依据以下公式得出：

WCS → BCS	$\vec{v} = R * \underline{S} * \underline{M} * \vec{v}' + \vec{t}$
BCS → WCS	$\vec{v}' = inv(\underline{M}) * inv(\underline{S}) * inv(\underline{R}) + (\vec{v} - \vec{t})$

V： BCS 中的位置矢量

V' WCS 中的位置矢量

：

11.5.3 数据管理框架和生效框架

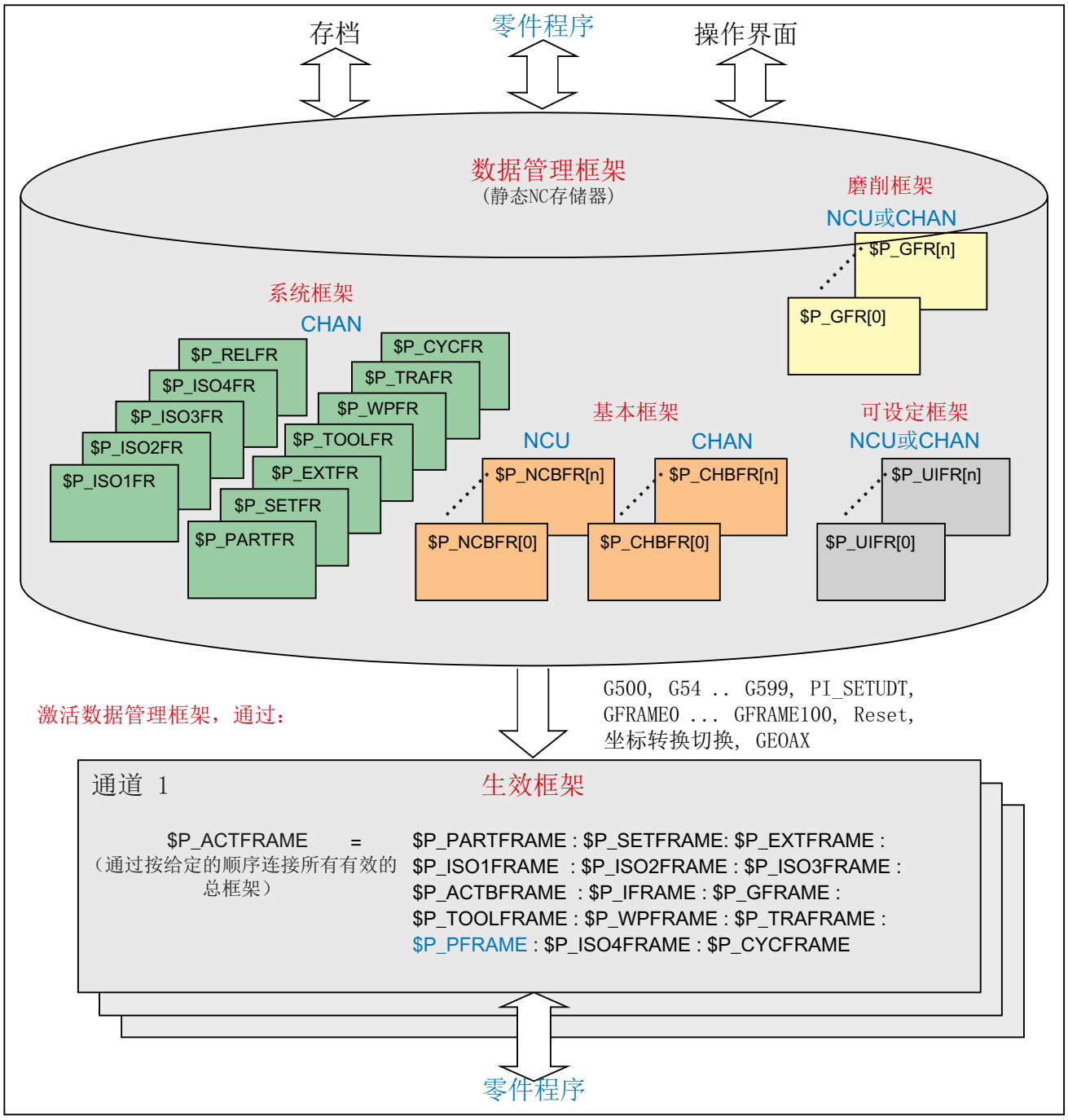
11.5.3.1 概述

框架类型

框架类型有：

- 系统框架 （$P_PARTFR, ... 见图）
- 基本框架 ($P_NCBFR[<n>], $P_CHBFR[<n>])
- 磨削框架 ($P_GFR[<n>])
- 可设定框架 ($P_UIFR[<n>])
- 可编程框架 ($P_PFRAME)

除去可编程框架，对所有框架，除通道中生效的框架外，均还在一个或多个数据管理中的框架（数据管理框架）。对于可编程框架，只存在一个在通道中生效的框架。



写入框架

可通过零件程序写入数据管理框架和生效框架。通过操作界面只可写入数据管理框架。

框架存档

只可存档数据管理框架。

11.5.3.2 激活数据管理框架

数据管理框架通过以下动作转换为生效框架：

- G 指令组“可设定框架”：G54 ... G57, G500, G505 ... G599
- G 指令组“磨削框架”：GFRAME0 ... GFRAME100
- 复位和 MD20110 $MC_RESET_MODE_MASK，位 14 == 1（保留基本框架的当前设置）
- 坐标转换切换
- 几何轴对应关系修改 GEOAX
- 由 HMI 通过 PI 服务 “_N_SETUDT”

由 HMI 激活

仅在当前 NC 程序恢复运行后，由 HMI 通过 PI 服务 “_N_SETUDT” 激活的数据管理框架才在通道中生效。

设置了以下机床数据时，激活在复位状态下立即生效：

MD9440 $MM_ACTIVATE_SEL_USER_DATA（使激活偏移立即生效）

激活系统框架

系统框架可通过以下方式激活：

- 在零件程序中编写对应的系统功能
- SINUMERIK Operate 上的操作

说明

修改数据管理的系统框架

原则上，数据管理的系统框架可由循环编程人员修改，并通过 G500、G54...G599 指令激活。但建议对此操作加以限制，在特定条件下使用。

激活数据管理框架

通过以下机床数据设置数据管理框架激活时的特性：

MD24050 $MC_FRAME_SAA_MODE（数据管理框架的保存和激活）

位	值	含义
0	0	@@@
	1	• 数据管理框架**只**通过编写系统变量 $P_CHBFRMASK、$P_NCBFRMASK 和 $P_CHSFRMASK 生效。 • G500...G599 只激活对应的可设定框架。
1	0	数据管理框架会因 TOROT、PAROT、外部零点偏移、坐标转换这些功能而被隐性赋值。
	1	数据管理框架**不**会因 TOROT、PAROT、外部零点偏移、坐标转换这些功能而被隐性赋值。

通过系统变量 $P_CHSFRMASK 激活系统框架

数据管理的系统框架可通过系统变量 **$P_CHSFRMASK** 激活。变量值以位编码设定，对应机床数据：
MD28082 $MC_MM_SYSTEM_FRAME_MASK（数据管理的系统框架）

通过将系统变量 $P_CHSFRMASK 的一个位设置为 1，使对应的数据管理系统框架在通道中生效。赋值为 0 时，通道中当前生效的系统框架继续生效。

复位后激活系统框架

依据下列机床数据中置位的位，复位后在通道中激活对应系统框架：

MD24006 $MC_CHSFRAME_RESET_MASK（复位后生效的系统框架）

激活 TCARR、PAROT 和 TOROT、TOFRAME 的系统框架

TCARR、PAROT 和 TOROT、TOFRAME 的系统框架根据以下机床数据中的设置激活：

MD20150 $MC_GCODE_RESET_VALUES（G 功能组的初始设置）

在选择/取消坐标转换或 GEOAX 指令使几何轴切换时，系统会删除当前总框架 $P_ACTFRAME，或依据新的几何轴配置重新计算和激活。系统框架和所有其他框架一样根据几何轴重新准备。

11.5.3.3 NCU 全局和通道专用变量

- 可设定框架和磨削框架只可配置为 NCU 全局型**或**通道专用型。
- 基本框架可配置为 NCU 全局型**或**通道专用型。
- NCU 全局框架在 NCU 的所有通道中生效。
- NCU 的所有通道都可读写 NCU 全局框架。
- 在所有通道中，机床轴对通道轴的指定关系，特别是机床轴对几何轴的指定关系可能不同，因此通道轴之间不存在几何关联。因此，NCU 全局框架中只可进行偏移、缩放和镜像。不可旋转。

说明

程序协调

用户自行负责协调 NCU 全局框架的访问。为此，建议使用程序协调指令。

文档

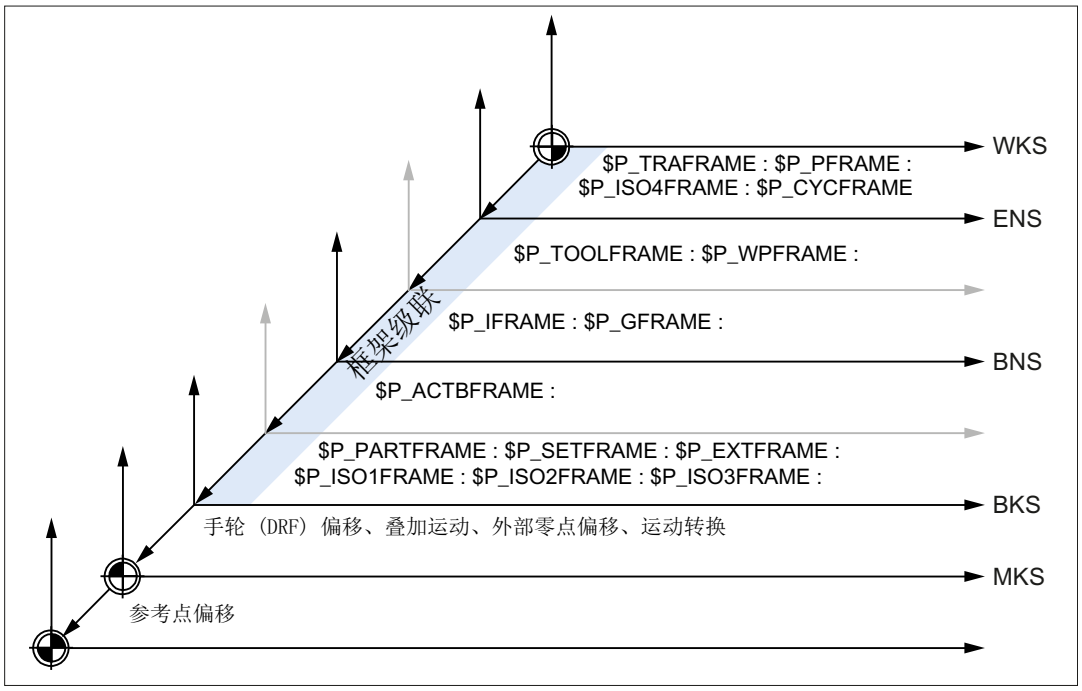
编程手册之工作准备分册；章节“灵活 NC 编程” > “程序协调（INIT、START、WAITM、WAITMC、WAITE、SETM、CLEARM）”

11.5.4 框架链和坐标系

11.5.4.1 概述

下图中显示了当前总框架的框架链。框架链位于基本坐标系（BCS）和工件坐标系（WCS）之间。

可设定的零点坐标系（SZS）等同于通过可编程框架转换的 WCS。基本零点坐标系（BZS）还包含当前的可设定框架。仅当进行了配置时，外部零点偏移的系统框架才存在，否则外部零点偏移会被作为轴的叠加运动运行。



WCS　**W**orkpiece **C**oordinate **S**ystem，工件坐标系

:

SZS　**S**ettable **Z**ero **S**ystem，可设定的零点坐标系

:

BZS　**B**asic **Z**ero **S**ystem，基本零点坐标系

:

BCS　**B**asic **C**oordinate **S**ystem，基本坐标系

:

MCS　**M**aschine **C**oordinate **S**ystem，机床坐标系

:

总框架

当前的总框架 $P_ACTFRAME 由框架链的所有生效框架级联得出：

```
$P_ACTFRAME =   $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
```

11.5.4.2 相对坐标系

相对坐标系的用处是：相对当前显示的坐标系中设定的参考点，显示当前的轴设定位置。相对坐标系方面无相关编程。只能在该系统中通过系统变量读取轴位置。

新的显示坐标系是相对于 WCS 和 SZS 的坐标系，其通过以下方式得到：使用生效的系统框架 $P_RELFRAME 对 WCS 或 SZS 轴位置进行坐标转换。相对坐标系不仅可以线性平移，还可旋转、镜像、压缩或扩展。

轴设定值的位置显示在 WCS 或 SZS 中进行。配置通过 HMI 机床数据进行。通道中始终只能有一个显示坐标系生效。因此只有一个相对框架可供使用，用于以相同的比例生成两个相对坐标系。HMI 依据配置显示相对坐标。

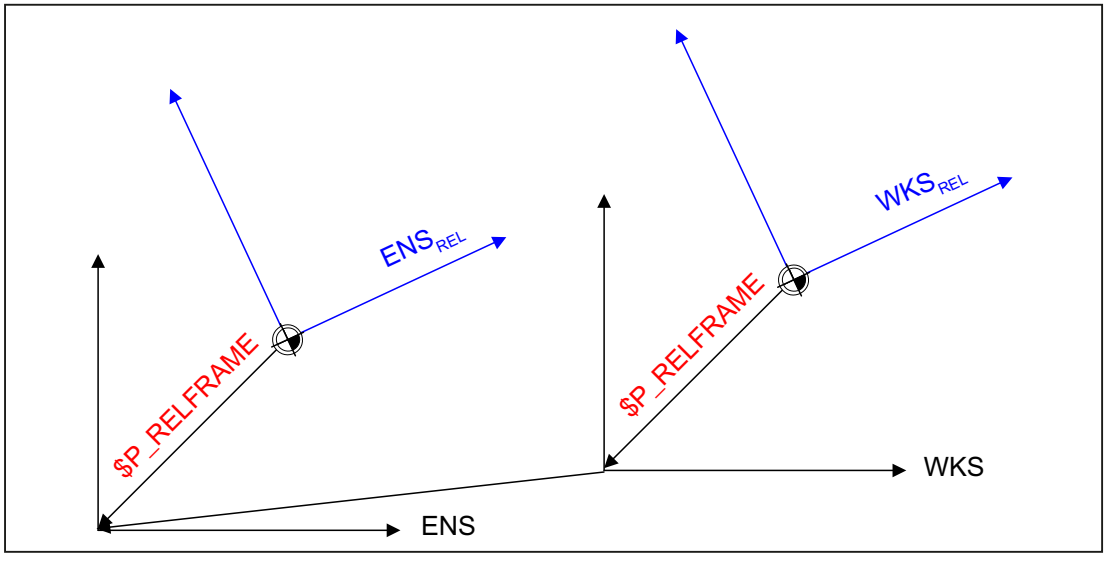


图 11-22 相对坐标系

通过 MD51036 $MNS_ENABLE_COORDINATE_REL=1 激活“相对坐标系”功能。

数据管理框架 $P_RELFR 可在零件程序中或通过 OPI 赋值。所有框架分量均可修改。

生效的系统框架 $P_RELFRAME 可在零件程序中赋值，并通过 OPI 读取。

系统框架 $P_RELFR 通过以下机床数据进行配置：

机床数据	位	含义
MD28082 $MC_MM_SYSTEM_FRAME_MASK	11	创建 $P_RELFR；相对坐标系将因此存在
MD28083 $MC_MM_SYSTEM_DATAFRAME_MASK	11	数据管理框架 $P_RELFR
MD24006 $MC_CHSFRAME_RESET_MASK.	11	$P_RELFR 在复位时生效

机床数据	位	含义
MD24007 $MC_CHSFRAME_RESET_CLEAR_MASK	11	$P_RELFR 在复位时删除
MD24008 $MC_CHSFRAME_POWERON_MASK	11	$P_RELFR 在上电时删除

相对坐标系 $WCS_{相对}$ 中的轴位置可通过变量 $AA_PCS_REL[轴] 读取。该变量可在零件程序中、OPI 上、以及通过同步动作读取。

相对坐标系 $SZS_{相对}$ 中的轴位置可通过变量 $AA_ACS_REL[轴] 读取。该变量可在零件程序中、OPI 上、以及通过同步动作读取。

通过操作界面设置相对参考点时，使用用于工件和刀具测量的通用命令接口进行设置。相对坐标系的系统变量 $P_RELFR 如下计算和激活：

- $AC_MEAS_TYPE = 14
- PI 通讯 _N_SETUDT(6, 7)

相对轴位置的设置示例请见：

文档：

功能手册之扩展功能分册；测量（M5），
“测量几何轴和辅助轴（测量类型 14、15）”章节

11.5.4.3 可选 SZS

在循环专用工件坐标系 (WCS) 中进行加工。此时，循环专用 WCS 是通过为循环编程的可编程框架 $P_PFRAME 和/或循环框架 $P_CYCFRAME 从 SZS 转换而来的。

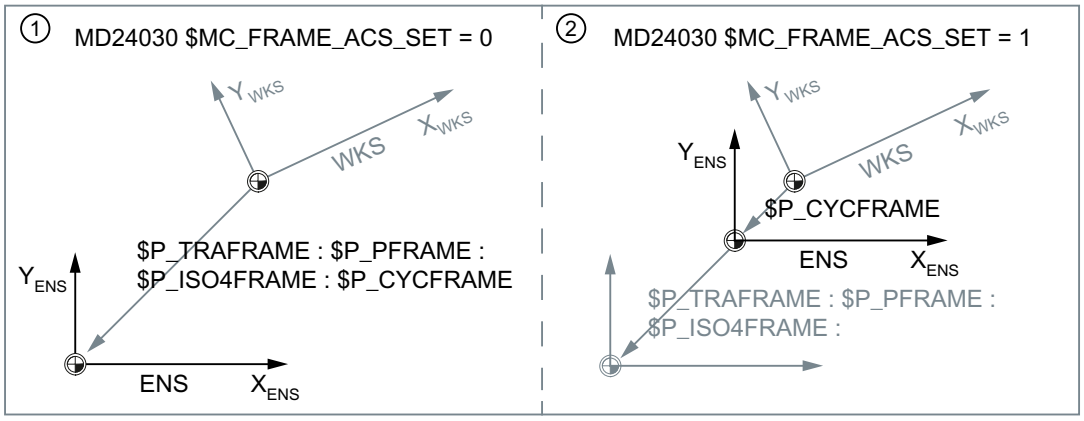
如果机床操作人员通过 NC 停止中断了循环，在激活循环前应运行有效的坐标系 SZS。

机床数据

可通过以下机床数据设置如何从循环专用 WCS 计算 SZS。

MD24030 $MC_FRAME_ACS_SET = <值>

<值>	含义
0	SZS = 通过 $P_TRAFRAME、$P_PFRAME、$P_ISO4FRAME 和 $P_CYCFRAME 转换的 WCS
1	SZS = 只通过 $P_CYCFRAME 转换的 WCS



① SZS = 通过 $P_TRAFRAME、$P_PFRAME、$P_ISO4FRAME 和 $P_CYCFRAME 转换的 WCS

② SZS = 只通过 $P_CYCFRAME 转换的 WCS

影响

SZS 的重新配置会影响：

- SZS 相关的实际值实际值显示，系统变量（例如： $AA_IEN 等）
- 在 SZS 中手动运行 (JOG) 几何轴

11.5.4.4 选择在 WCS 或 SZS 中手动运行几何轴 ($AC_JOG_COORD)

目前在 JOG 运行方式下执行手动运行时，几何轴是在 WCS 中运行。此外也可在 SZS 坐标系中执行手动运行。为此，可使用变量 $AC_JOG_COORD 在 WCS 中的手动运行和 SZS 中的手动运行之间进行切换。用户可选择在 SZS 或 WCS 中运行。

在 JOG 运行方式下，可选择在工件坐标系 (WCS) 或可设定零点坐标系 (SZS) 中手动运行几何轴。

系统变量

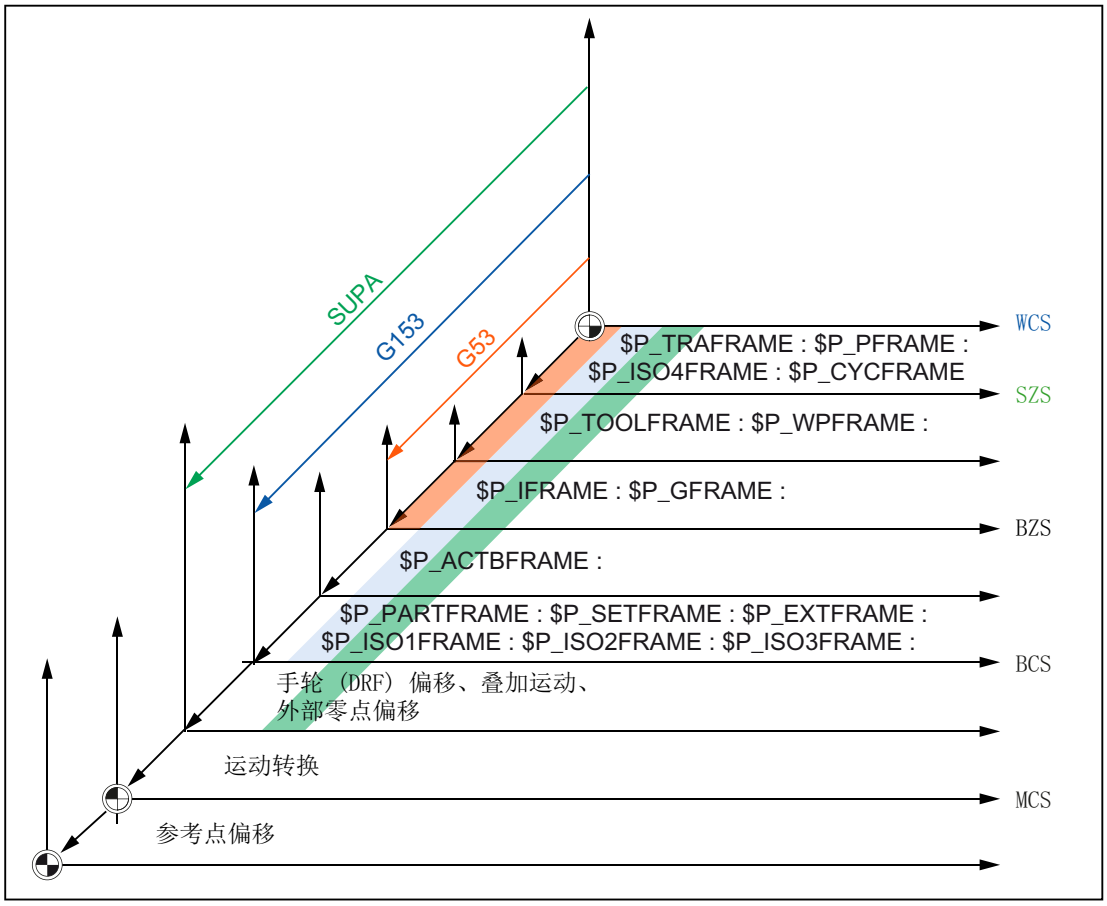
在 JOG 运行方式下，可选择在工件坐标系 (WCS) 或可设定零点坐标系 (SZS) 中手动运行几何轴。通过系统变量 $AC_JOG_COORD 选择：

$AC_JOG_COORD = <值>

<值>	含义
0	工件坐标系 (WCS)
1	可设定零点坐标系 (SZS)

11.5.4.5 抑制框架

通过下述指令 G53、G135 和 SUPA 抑制框架。激活框架抑制会导致与 WCS、SZS 或 BZS 相关的系统变量中的位置显示 (HMI) 及位置数据跳动。该特性可通过机床数据设置。



机床数据

通过以下机床数据确定系统变量中的位置显示 (HMI) 和位置数据的特性：

MD24020 $MC_FRAME_SUPPRESS_MODE，位 <n> = <值>（框架抑制时的位置）

位	值	含义
0	0	位置数据 (OPI)，**带**框架抑制 [1]
	1	位置数据 (OPI)，**不带**框架抑制 [2]

位	值	含义
1	0	系统变量中的位置数据，**带**框架抑制 1)
	1	系统变量中的位置数据，**不带**框架抑制 2)
1) 位置值跳动 2) 位置值**不**跳动		

编程

指令	含义
G53：	逐段抑制下列框架： $P_TRAFRAME : $P_PFRAME : $P_ISO4FRAME : $P_CYCFRAME $P_IFRAME : $P_GFRAME : $P_TOOLFRAME : $P_WPFRAME :
G153：	逐段抑制 G53 时的框架及下列框架： $P_PARTFRAME : $P_SETFRAME : $P_EXTFRAME : $P_ACTBFRAME $P_ISO1FRAME : $P_ISO2FRAME : $P_ISO3FRAME :
SUPA：	隐性预处理停止并逐段抑制 G53 和 G135 时的框架及下列框架： ● 手轮偏移（DRF） ● 叠加运动 ● 外部零点偏移
G500：	模态激活 G500 框架。G500 框架通常为零框架。
DRFOF：	关闭（取消）手轮偏移 (DRF)

11.5.5 框架链的框架

11.5.5.1 概述

可使用的框架有：

- 可设定框架 (G500, G54 ... G57, G505 ... G599)
- 磨削框架 (GFRAME0 ... GFRAME100)
- 基本框架

- 可编程的框架
- 系统框架

11.5.5.2 可设定框架 $P_UIFR[<n>]

机床数据

通道专用可设定框架

通道专用可设定框架的数量通过以下机床数据设置：

MD28080 $MC_MM_NUM_USER_FRAMES = <数量>

系统变量索引 n = 0, 1, 2, ... <数量> - 1

NCU 全局可设定框架

NCU 全局可设定框架的数量通过以下机床数据设置：

MD18601 $MN_MM_NUM_GLOBAL_USER_FRAMES = <数量>

系统变量索引 n = 0, 1, 2, ... <数量> - 1

如果机床数据的值 > 0，则**不**存在**通道专用**可设定框架。系统不会检测用于设置通道专用可设定框架的机床数据。

可设定框架的 G 功能组的初始设置

通过以下机床数据设置初始位置或通道复位或上电后可设定框架专用的**第 8 个 G 功能组**中的哪些 G 指令生效：

MD20150 $MC_GCODE_RESET_VALUES[**7**] = <值>

值	G 指令
1	G500
2	G54
3	G55
4	G56
5	G57
6	G505
...	...
100	G599

磨削专用的 G 功能组的复位特性

通过以下机床数据设置可设定框架专用的**第 8 个 G 功能组**的复位特性：

MD20152 $MC_GCODE_RESET_MODE[**7**] = <值>

值	含义 **通道复位或零件程序结束后：**
0	可设定框架专用的 G 指令生效，依据 **MD20150**。
1	当前生效的可设定框架专用的 G 指令继续生效。

说明

MD20110 $MC_RESET_MODE_MASK

机床数据 MD20152 $MC_GCODE_RESET_MODE 在以下条件时才被检测：

MD20110 $MC_RESET_MODE_MASK，位 0 == 1

系统变量

$P_UIFR[<n>]（数据管理的可设定框架）

通过系统变量 $P_UIFR[<n>] 可以读取和写入数据管理的可设定框架。在写入数据管理的可设定框架时，新值不会立即在通道中生效。只有在编程了零点偏移 G500,G54...G599 后，才可在通道中激活。

对于 NCU 全局框架，修改过的数据管理的可设定框架在 NCU 的每个执行 G500、G54..G599 指令的通道中生效。

在进行数据备份时会一并备份数据管理中的可设定框架。

$P_IFRAME（生效的可设定框架）

通过系统变量 $P_IFRAME 可以读取和写入通道中**生效的**可设定框架。在写入可设定框架时，新值不会立即在通道中生效。

在 NCU 全局可设定框架中，修改过的可设定框架仅在编程了新值的通道中生效。如果修改过的 NCU 全局可设定框架适用于 NCU 的所有通道，必须同时写入在通道中生效的可设定框架**和**数据管理中相应的可设定框架。

$P_UIFR[<n>] = $P_IFRAME = <新值>

- $P_UIFR[<n>]（数据管理中的可设定框架）
- $P_IFRAME（通道中生效的可设定框架）

要使修改过的可设定框架在另一个通道中生效，还须在该通道中通过相应的指令（例如：G54）进行激活。

$P_UIFRNUM（生效的可设定框架的编号）

通过系统变量 $P_UIFRNUM 可以读取通道中生效的可设定框架的索引 <n>：

通道中生效的可设定框架 $P_IFRAME == $P_UIFR[$P_UIFRNUM]

$P_UIFRNUM	$P_IFRAME == $P_UIFR[<n>]，其中 n =
0	0
1	1
2	2
...	...
99	99

编程

用于激活通道中的可设定框架的指令

通过编程指令 G500,G54...G599 可使通道中的数据管理可设定框架 $P_UIFR[<n>] 生效或者按照与数据管理可设定框架 $P_UIFR[<n>] 相同的方式设置生效的可设定框架 $P_IFRAME：

G<x> ⇒ $P_IFRAME = $P_UIFR[<n>]

指令	生效的可设定框架 $P_IFRAME =
G500	$P_UIFR[0]
G54	$P_UIFR[1]
G55	$P_UIFR[2]
G56	$P_UIFR[3]
G57	$P_UIFR[4]
G505	$P_UIFR[5]
...	...
G599	$P_UIFR[99]

边界条件

通过 HMI / PLC 写入可设定框架

通过 HMI 或 PLC 用户程序只能写入**数据管理**的可设定框架。

11.5.5.3 磨削框架 $P_GFR[<n>]

磨削框架专门用于磨削工艺，提供了附加零点偏移和补偿。还可作用于可设定框架 (页 838) 的零点偏移。

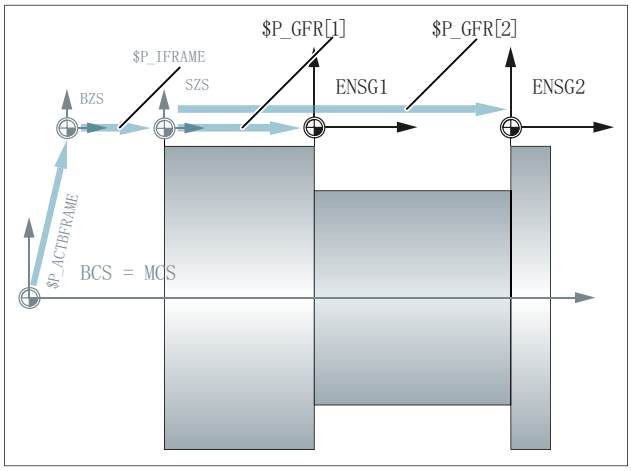


图 11-23 磨削框架

根据基本零点坐标系 (BZS)，由通道中生效的框架的级联得出可设定零点坐标系 (SZS)。

$P_IFRAME : **$P_GFRAME** : $P_TOOLFRAME :$P_WPFRAME

机床数据

通道专用磨削框架的数量

通过以下机床数据设置通道专用磨削框架的数量：

MD28079 $MN_MM_NUM_G_FRAMES = <数量>

其中 <数量> = 0, 1, 2, ... 最大数量

NCU 全局磨削框架的数量

通过以下机床数据设置 NCU 全局磨削框架的数量：

MD18603 $MN_MM_NUM_GLOBAL_G_FRAMES = <数量>

其中 <数量> = 0, 1, 2, ... 最大数量

如果机床数据的值 > 0，则**不**存在**通道专用**磨削框架。系统不会检测用于设置通道专用磨削框架的机床数据。

磨削专用 G 功能组的初始位置 (64)

通过以下机床数据设置初始位置或通道复位或上电后磨削专用的**第 64 个 G 功能组**中的哪些 G 指令生效：

MD20150 $MC_GCODE_RESET_VALUES[**63**] = <值>

值	G 指令
1	GFRAME0（零框架）
2	GFRAME1
3	GFRAME2
...	...
101	GFRAME100

磨削专用的 G 功能组的复位特性

通过以下机床数据设置磨削专用的**第 64 个 G 功能组**的复位特性：

MD20152 $MC_GCODE_RESET_MODE[**63**] = <值>

值	含义 **通道复位或零件程序结束后：**
0	可设定框架专用的 G 指令生效，依据 **MD20150**。
1	当前生效的磨削专用 G 指令继续生效。

说明

MD20110 $MC_RESET_MODE_MASK

机床数据 MD20152 $MC_GCODE_RESET_MODE 在以下条件时才被检测：

MD20110 $MC_RESET_MODE_MASK，位 0 == 1

系统变量

$P_GFR[<n>]（数据管理的磨削框架）

通过系统变量 $P_GFR[<n>] 可以读取和写入数据管理的磨削框架。在写入磨削框架时，新值不会立即在通道中生效。只有在编程了相应的指令 GFRAME0 ... GFRAME100 后，才可在通道中激活。对于 NCU 全局框架，修改过的框架在 NCU 的每个执行 GFRAME0 ... GFRAME100 指令的通道中生效。

在进行数据备份时会一并备份数据管理的磨削框架。

说明

显示（SINUMERIK Operate）

数据管理的磨削框架会显示在 SINUMERIK Operate 操作界面的一个单独的窗口中。

- 参考文档
  磨削版操作手册；章节“设置机床” > “零点偏移” > “显示并编辑基于位置的精偏移”

删除（SINUMERIK Operate）

数据管理的磨削框架可以单独或全部一起，例如杂更换工件后，在 SINUMERIK Operate 操作界面上进行删除。

- 参考文档
  磨削版操作手册；章节“设置机床” > “零点偏移” > “删除基于位置的精偏移”

$P_GFRAME（生效的磨削框架）

通过系统变量 $P_GFRAME 可以读取和写入通道中**生效的**磨削框架。在写入磨削框架时，新值不会立即在通道中生效。

对于 NCU 全局磨削框架，修改过的框架仅在编写了新框架值的通道中生效。

如果修改过的 NCU 全局磨削框架适用于 NCU 的所有通道，必须同时写入在通道中生效的磨削框架 $P_GFRAME 和数据管理中的磨削框架 $P_GFR[<n>]：

$P_GFRAME = $P_GFR[<n>] = <新值>

要使修改过的磨削框架 $P_GFR[<n>] 在另一个通道中生效，还须在该通道中通过相应的指令（例如：GFRAME<n>）进行激活。

$P_GFRNUM（生效的磨削框架的编号）

通过系统变量 $P_GFRNUM 可以读取通道中生效的磨削框架的索引 <n>：

通道中生效的磨削框架 $P_GFRAME == $P_GFR[$P_GFRNUM]

通过 G 指令 GFRAME<n> 激活的磨削框架 $P_GFRAME：	$P_GFRNUM
GFRAME0	0
GFRAME1	1
GFRAME2	2
...	...
GFRAME100	100

编程

用于激活通道中的磨削框架的指令

通过编程指令 GFRAME<n> 使通道中的相应磨削框架 $P_GFR[<n>] 生效。为此，必须按照与磨削框架 $P_GFR[<n>] 相同的方式设置生效的磨削框架 $P_GFRAME：

GFRAME<n> ⇒ $P_GFRAME = $P_GFR[<n>]

指令	通道中生效的磨削框架
GFRAME0	$P_GFR[0]（**零框架**）
GFRAME1	$P_GFR[1]
...	...
GFRAME100	$P_GFR[100]

句法

GFRAME<n>

含义

GFRAME<n>：	激活数据管理的磨削框架 **<n>**	
	G 功能组：	64
	初始设置：	MD20150 $MC_GCODE_RESET_VALUES[**63**]
	生效方式：	模态
<n>：	磨削框架的编号	
	值域：	0, 1, 2, ... 100

边界条件

通过 HMI / PLC 写入磨削框架

通过 HMI 或 PLC 用户程序只能写入**数据管理**的磨削框架。

11.5.5.4 通道专用基本框架[<n>]

机床数据

通道专用基本框架的数量

通道专用基本框架的数量通过以下机床数据设置：

MD28081 $MC_MM_NUM_BASE_FRAMES = <数量>

系统变量索引 n = 0, 1, 2, ... <数量> - 1

系统变量

$P_CHBFR[<n>]（数据管理的通道专用基本框架）

通过系统变量 $P_CHBFR[<n] 可以读取和写入数据管理的通道专用基本框架。在写入通道专用基本框架时，新值不会立即在通道中生效。只有在编程了相应的指令 G500,G54..G599 后，才可在通道中激活。

在进行数据备份时会一并备份数据管理的通道专用基本框架。

$P_CHBFRAME[<n>]（生效的通道专用基本框架）

通过系统变量 $P_CHBFRAME[<n] 可以读取和写入通道中**生效的**通道专用基本框架。在写入生效的通道专用基本框架时，通过重新计算生效的总基本框架 $P_ACTBFRAME 使新值在通道中立即生效。

出于兼容性原因的系统变量

$P_UBFR（第一个通道专用基本框架）

此变量是 $P_CHBFR[0] 的冗余变量，由于兼容性的原因得以保留。

向预定义变量 $P_UBFR 写入时，数组索引为 0 的基本框架不会被立即激活，而是在执行 G500、G54..G599 这些指令中的一个时才激活。对于 NCU 全局框架，修改过的框架在 NCU 的每个执行 G500、G54..G599 指令的通道中生效。该变量主要用作从 HMI 或 PLC 写入到基本框架时的存储器。此变量也可在程序中读写。

$P_UBFR 与 $P_CHBFR[0] 一致。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，那么在读写时会产生报警“框架：不允许使用的指令”。

$P_BFRAME（第一个生效的通道专用基本框架）

此变量是 $P_CHBFRAME[0] 的冗余变量，由于兼容性的原因得以保留。

通过预定义框架变量 $P_BFRAME，可在零件程序中读写通道中生效的、数组索引为 0 的基本框架。写入的基本框架会被立即计算在内。对于 NCU 全局可设定框架，修改的框架仅在编写了该框架的通道中生效。若需修改某个 NCU 的所有通道的框架，则须同时写入 $P_UBFR 和 $P_BFRAME。然后其它通道必须激活相应框架，例如通过 G54。

$P_BFRAME 与 $P_CHBFRAME[0] 一致。缺省情形下，系统变量始终有一个有效值。如果没有通道专用基准框架，那么在读写时会产生报警“框架：不允许使用的指令”。

边界条件

通过 HMI / PLC 写入基本框架

通过 HMI 或 PLC 用户程序只能写入**数据管理**的基本框架。

11.5.5.5 NCU 全局基本框架 $P_NCBFR[<n>]

机床数据

NCU 全局基本框架的数量

NCU 全局基本框架的数量通过以下机床数据设置：

MD18602 $MN_MM_NUM_GLOBAL_BASE_FRAMES = <数量>

系统变量索引 n = 0, 1, 2, ... <数量> - 1

系统变量

$P_NCBFR[<n>]（数据管理的 NCU 全局基本框架）

通过系统变量 $P_NCBFR[<n] 可以读取和写入数据管理的 NCU 全局基本框架。在写入 NCU 全局基本框架时，新值不会立即在通道中生效。只有在编程了相应的指令 G500,G54..G599 后，才可在通道中激活。

在进行数据备份时会一并备份数据管理的 NCU 全局基本框架。

$P_NCBFRAME[<n>]（当前 NCU 全局基本框架）

通过系统变量 $P_NCBFRAME[<n] 可以读取和写入通道中**生效的** NCU 全局基本框架。在写入生效的 NCU 全局基本框架时，通过重新计算生效的总基本框架 $P_ACTBFRAME 使新值在通道中立即生效。

如果修改过的 NCU 全局基本框架适用于 NCU 的所有通道，必须同时写入在通道中生效的 NCU 全局基本框架和数据管理的 NCU 全局基本框架：

$P_NCBFR[<n>] = $P_NCBFRAME = <新值>

- $P_NCBFR[<n>]（数据管理的 NCU 全局基本框架）
- $P_NCBFRAME（通道中生效的 NCU 全局基本框架）

要使修改过的 NCU 全局基本框架在另一个通道中生效，还须在该通道中通过相应的指令（例如：G500,G54..G599）进行激活。

编程

通过相应的指令 (G54 ... G57, G505 ... G599 und G500) 将数据管理的通道专用可设定框架 $P_UIFR[<n>] 转换为在通道中生效的可设定框架 $P_IFRAME。

指令	激活数据管理的 NCU 全局和通道专用基本框架
G500	$P_CHBFR[0] : $P_NCBFR[0]
G54	$P_CHBFR[1] : $P_NCBFR[1]
G55	$P_CHBFR[2] : $P_NCBFR[2]
G56	$P_CHBFR[3] : $P_NCBFR[3]
G57	$P_CHBFR[4] : $P_NCBFR[4]
G505	$P_CHBFR[5] : $P_NCBFR[5]
...	...
G599	$P_CHBFR[99] : $P_NCBFR[99]

11.5.5.6 生效的总基本框架 $P_ACTBFRAME

功能

总基本框架 $P_ACTBFRAME 中汇总了所有生效的 NCU 全局和通道专用基本框架：

```
$P_ACTBFRAME = $P_NCBFRAME[0] : ... : $P_NCBFRAME[<n>] :
               $P_CHBFRAME[0] : ... : $P_CHBFRAME[<n>]
```

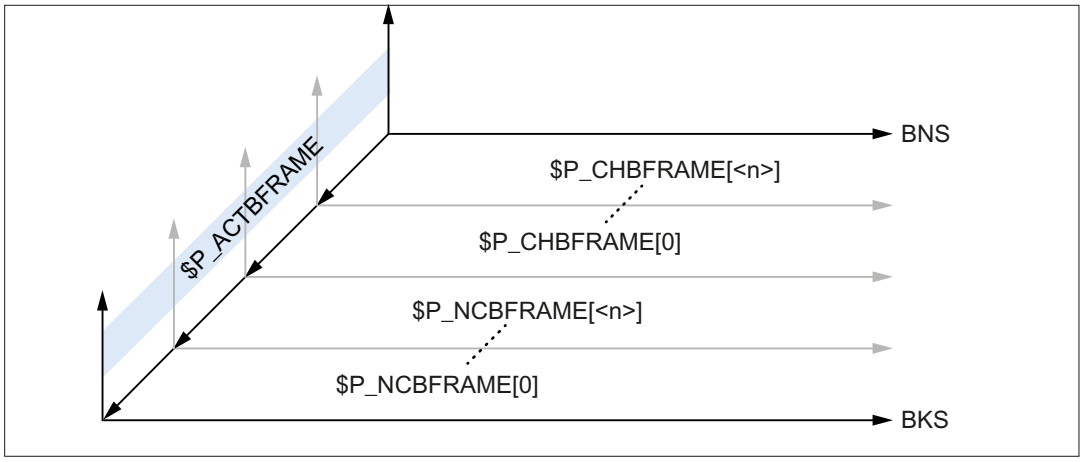


图 11-24　总基准框架

机床数据

复位特性

通过以下机床数据设置复位（通道复位、程序结束复位或上电）后生效的基本框架：

MD20110 $MC_RESET_MODE_MASK，位 0 = 1 和位 14 = 1

- 位 0 = 1：缺省值 ⇒ 设置其他位时的复位特性
- 位 14 = 0：通过复位彻底取消基本框架。
- 位 14 = 1：通过复位将机床数据设置接收到系统变量中且所选基本框架生效：
  – $P_NCBFRMASK = MD10613 $MN_NCBFRAME_RESET_MASK
  – $P_CHBFRMASK = MD24002 $MC_CHBFRAME_RESET_MASK

示例

机床数据设置	生效的基本框架
$P_NCBFRMASK = MD10613 $MN_NCBFRAME_RESET_MASK = 'H81'	$P_NCBFRAME[0] : $P_NCBFRAME[7]

编程

基本框架掩码

通过基本框架掩码 $P_NCBFRMASK 和 $P_CHBFRMASK 选择与总基本框架级联的基本框架。

通过设置基本框架**掩码**中的位选择相应的基本框架：

- $P_NCBFRMASK，位 **0, 1, 2, ... n** ⇒ $P_NCBFRAME[**0, 1, 2, ... n**]
- $P_CHBFRMASK，位 **0, 1, 2, ... n** ⇒ $P_NCHFRAME[**0, 1, 2, ... n**]

基本框架掩码 $P_NCBFRMASK 和 $P_CHBFRMASK 只能在 NC 程序中读取和写入。通过 OPI 可以读取基本框架掩码。

写入基本框架掩码后，系统会重新计算生效的总基本框架 $P_ACTBFRAME 和总框架 $P_ACTFRAME。

示例

程序代码	注释
$P_NCBFRMASK = 'H81'	; 生效的 NCU 全局基本框架：$P_NCBFRAME[0] : $P_NCBFRAME[7]

11.5.5.7 可编程框架 $P_PFRAME

可编程框架只存在生效框架。
此框架是为编程人员预留。

采用以下设置时，可编程框架在复位时保持生效：

MD24010 $MC_PFRAME_RESET_MODE（可编程框架的复位模式）= 1

在复位后需要从斜孔运行出来的情形下，此功能尤为重要。

MIRROR

到目前为止（至软件 P4），机床轴的镜像均以以下机床数据：
MD10610 $MN_MIRROR_REF_AX（镜像的基准轴）
定义的参考轴为基准。

从用户角度看来，此定义难以理解。在进行 z 轴的镜像时，显示中会提示 x 轴已执行镜像，y 轴已旋转 180°。若对两根轴进行镜像，情况将更为复杂，且无法轻易理解哪些轴已执行镜像、哪些轴没有。

从 SW-P5 开始，可通过一种新的方式明确显示轴的镜像。其不再投影为参考轴的镜像和其他轴的旋转。

这可通过以下机床数据设置：

MD10610 $MN_MIRROR_REF_AX = 0

通过 MIRROR 和 AMIRROR 进行的可编程框架编程得到扩展。在此之前，设定的坐标轴的值不会被控制系统分析，例如 MIRROR X0 中的值 0；而 AMIRROR 却是有切换功能，

MIRROR X0 能够激活镜像，再次编写 AMIRROR X0 则又将其取消。MIRROR 总是绝对生效，而 AMIRROR 为附加生效。

通过以下机床数据设置：
MD10612 $MN_MIRROR_TOGGLE = 0（“镜像切换”）
可定义对编写的值进行分析。
在值为 0，例如 AMIRROR X0 时，取消该轴的镜像。值不为 0 时，若轴尚未镜像，则对其执行镜像。

对镜像的逐分量读写与以下机床设置无关：

MD10612 $MN_MIRROR_TOGGLE

值 = 0 表示之后轴未经过镜像；值 = 1 表示之后总是进行镜像，与该轴是否已经过镜像无关。

```
$P_NCBFR[0,x,mi]=1          ; x 轴总是执行镜像。
$P_NCBFR[0,x,mi]=0          ; x 轴镜像关闭。
```

；轴替换 G58，G59（仅适用于 840D sl）

可编程框架的偏移分量划分为绝对分量，以及用于将所有叠加编写的偏移求和的分量。绝对分量可通过 TRANS、CTRANS 或写入偏移分量修改，此时叠加分量会被设置为零。G58 只会修改指定轴的绝对偏移分量，叠加编写的偏移的总和保持不变。

```
G58 X...Y...Z...A... ...
```

G59 用于针对指定轴改写通过 ATRANS 编写的叠加偏移。

```
G59 X...Y...Z...A... ...
```

示例

```
TRANS X10 Y10 Z10
ATRANS X5 Y5                ; 总偏移 X15 Y15 Z10
G58 X20                     ; 总偏移 X25 Y15 Z10
G59 X10 Y10                 ; 总偏移 X30 Y20 Z10
```

仅当满足以下条件时，才可使用 G58 和 G59：

MD24000 $MC_FRAME_ADD_COMPONENTS（G58 / G59 的框架分量）== TRUE

下表中介绍了各种编程指令对绝对偏移和附加偏移的影响。

	粗偏或绝对偏移	精偏或叠加偏移
TRANS X10	10	0
ATRANS X10	不变	alt_fine + 10
CTRANS(X,10)	10	0
CTRANS()	0	0
CFINE(X,10)	0	10
$P_PFRAME[X,TR] = 10	10	不变
$P_PFRAME[X,FI] = 10	不变	10
G58 X10	10	不变
G59 X10	不变	10

11.5.5.8 通道专用系统框架

通道专用系统框架只由系统功能描述，例如实际值设置、对刀、外部零点偏移和倾斜加工。

机床数据

通道专用系统框架的参数设置

由于存储空间原因，建议只配置系统功能所必需的通道专用系统框架。

每个系统框架会在每个通道中占用约 1 kB SRAM 和约 6 kB DRAM。

通过以下机床数据设置通道专用系统框架的参数：

MD28082 $MC_MM_SYSTEM_FRAME_MASK，位 <n>

位	值	现有系统框架：
0	1	$P_SETFR：实际值设置和对刀
1	1	$P_EXTFR：通过系统框架实现外部零点偏移
2	1	$P_PARTFR：使用可定向刀架时的 TCARR 和 PAROT
3	1	$P_TOOLFR：TOROT 和 TOFRAME
4	1	$P_WPFR：工件参考点的框架
5	1	$P_CYCFR：循环的框架
6	1	$P_TRAFR：用于坐标转换选择/取消的框架
7	1	$P_ISO1FRAME：G51.1 镜像（ISO）的框架

位	值	现有系统框架：
8	1	$P_ISO2FRAME：G68 2DROT（ISO）的框架
9	1	$P_ISO3FRAME：G68 3DROT（ISO）的框架
10	1	$P_ISO4FRAME：G51 比例缩放（ISO）的框架
11	1	$P_RELFR：相对坐标系的框架

ACS 坐标系的参数设置

通过以下机床数据确定 ACS 坐标系是由哪些系统框架构成的

MD24030 $MC_FRAME_ACS_SET = <值>

<值>	含义：ACS 坐标系由以下系统框架构成：
0	$P_PARTFRAME : $P_SETFRAME : $P_EXTFRAME :$P_ISO1FRAME : $P_ISO2FRAME : $P_ISO3FRAME :$P_ACTBFRAME : $P_IFRAME : $P_GFRAME : $P_TOOLFRAME : **$P_WPFRAME**
1	$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**

系统变量

数据管理的通道专用系统框架

数据管理的通道专用系统框架可通过以下框架变量读写：

系统变量	含义：数据管理的系统框架，用于
$P_SETFR	实际值设置和对刀（**Set-Fr**ame，设置框架）
$P_EXTFR	外部零点偏移（**Ext-Fr**ame，外部框架）
$P_PARTFR	使用可定向刀架时的 TCARR 和 PAROT（**Part-Fr**ame，零件框架）
$P_TOOLFR	TOROT 和 TOFRAME（**Tool-Fr**ame，刀具框架）
$P_WPFR	工件参考点（**W**ork-**P**iece-**Fr**ame，工件框架）
$P_CYCFR	循环（**Cyc**le-**Fr**ame，循环框架）
$P_TRAFR	坐标转换（(**Tra**nsformation-**Fr**ame，转换框架）
$P_ISO1FR	G51.1 镜像（ISO）
$P_ISO2FR	G68 2DROT (ISO)

系统变量	含义：数据管理的系统框架，用于
$P_ISO3FR	G68 3DROT (ISO)
$P_ISO4FR	G51 比例缩放 (ISO)
$P_RELFR	相对坐标系

说明

循环编程

系统框架的系统变量仅可用于循环编程。因此，在 NC 程序中，系统变量不是由用户直接，而是通过系统功能（例如：TOROT、PAROT 等）写入的。

生效的通道专用系统框架

生效的通道专用系统框架的系统变量：

系统变量	含义：生效的系统框架，用于
$P_SETFRAME	实际值设置和对刀（**Set**-**Fr**ame，设置框架）
$P_EXTFRAME	外部零点偏移（**Ext**-**Fr**ame，外部框架）
$P_PARTFRAME	使用可定向刀架时的 TCARR 和 PAROT（**Part**-**Fr**ame，零件框架）
$P_TOOLFRAME	TOROT 和 TOFRAME（**Tool**-**Fr**ame，刀具框架）
$P_WPFRAME	工件参考点（**W**ork-**P**iece-**Fr**ame，工件框架）
$P_CYCFRAME	循环（**Cyc**le-**Fr**ame，循环框架）
$P_TRAFRAME	坐标转换（(**Tra**nsformation-**Fr**ame，转换框架）
$P_ISO1FRAME	G51.1 镜像（ISO）
$P_ISO2FRAME	G68 2DROT (ISO)
$P_ISO3FRAME	G68 3DROT (ISO)
$P_ISO4FRAME	G51 比例缩放 (ISO)
$P_RELFRAME	相对坐标系

如果数据管理的通道专用系统框架未设置参数，系统框架“**$P_<**系统框架**> ==** 零框架”自动生效。

通道专用的 ACS 总框架

通过系统变量 $P_ACSFRAME 可以读取构成 ACS 坐标系的系统框架。通过上述机床数据 MD24030 $MC_FRAME_ACS_SET 确定。参见章节“机床数据” > “ACS 坐标系的参数设置”

系统变量	含义：生效的系统框架，用于
$P_ACSFRAME	根据机床数据 MD24030 $MC_FRAME_ACS_SET 中的参数设置构成 ACS 坐标系的系统框架。

11.5.6 隐性框架修改

11.5.6.1 切换几何轴

可在激活/取消坐标转换或使用指令 GEOAX() 时切换几何轴。

通过以下机床数据可针对当前总框架 $P_ACTFRAME 的操作进行四种不同的设置：

MD10602 $MN_FRAME_GEOAX_CHANGE_MODE = <值>

<值>	含义
0	删除 几何轴切换时，例如激活和取消坐标转换以及使用 GEOAX() 指令时，删除当前总框架。 激活新的框架后，变更过的几何轴配置才会被启用。
1	通过接收旋转重新计算 几何轴切换时重新计算当前总框架，同时新几何轴的框架分量生效。切换前编写的几何轴旋转同样对新几何轴保持生效。 TRANSMIT、TRACYL 和 TRAANG 的相关信息请见“选择/取消坐标转换：常规 (页 857)”一章。

<值>	含义
2	只能在无生效的旋转时重新计算 几何轴切换时重新计算当前总框架，同时新几何轴的框架分量生效。若切换前当前基本框架、当前可设定框架或可编程框架中有生效的旋转，那么切换将被终止，并触发报警“框架：不允许的几何轴切换”。 TRANSMIT、TRACYL 和 TRAANG 的相关信息请见“选择/取消坐标转换：常规 (页 857)”一章。
3	转换：删除 / GEOAX()：通过接收旋转重新计算 ● 转换：选择和取消坐标转换时，删除当前总框架。 ● GEOAX()：GEOAX() 指令下重新计算总框架，同时新几何轴的偏移、比例缩放和镜像生效。切换前编写的几何轴旋转同样对新几何轴保持生效。

工件几何数据通过几何轴构成的坐标系描述。每根几何轴均被指定一根通道轴，每根通道轴均被指定一根机床轴。针对每个框架（系统框架、基本框架、可设定框架、可编程框架），每根机床轴都有一个轴框架。如果几何轴被指定了另一根机床轴，机床轴会算入其自有轴框架分量。这样一来，通道中的新几何配置将由新的轮廓框架构成，其包含最多三根新几何轴。

几何轴切换时，当前生效的框架会被重新计算，从而得到一个总框架。数据管理框架要在激活后才会被启用。

示例

需通过 GeoAx 切换将通道轴 A 设定为几何轴 X。切换后，可编程框架在 X 轴方向上有 10 的偏移分量。保留当前可设定框架。

MD10602 $MN_FRAME_GEOAX_CHANGE_MODE = 1

程序代码	注释
`$P_UIFR[1] = CROT(X,10,Y,20,Z,30)`	`; GeoAx 切换后保留框架。`
`G54`	`; 可设定框架生效。`
`TRANS A10`	`; 切换时附带轴 A 的轴偏移。`
`GEOAX(1, A)`	`; 将轴 A 设定为 X 轴`
	`; $P_ACTFRAME = CROT(X,10,Y,20,Z, 30) :CTRANS(X10)`

坐标转换切换时，可同时将多个通道轴设定为几何轴。

示例

通过 5 轴定向转换，通道轴 4、5、6 成为转换的几何轴。即在转换前替换所有几何轴。激活转换时，所有当前框架相应变化。为了计算新的 WCS 坐标系，将成为几何轴的通道轴的轴框架分量纳入计算。转换前编写的旋转保留。取消转换后，重新恢复为原先的 WCS。最常见应用就是使几何轴在坐标转换前后不发生变化，并保留转换前生效的框架。

机床数据

程序代码

```
$MC_AXCONF_CHANAX_NAME_TAB[0] = "CAX"
$MC_AXCONF_CHANAX_NAME_TAB[1] = "CAY"
$MC_AXCONF_CHANAX_NAME_TAB[2] = "CAZ"
$MC_AXCONF_CHANAX_NAME_TAB[3] = "A"
$MC_AXCONF_CHANAX_NAME_TAB[4] = "B"
$MC_AXCONF_CHANAX_NAME_TAB[5] = "C"

$MC_AXCONF_GEOAX_ASSIGN_TAB[0] = 1
$MC_AXCONF_GEOAX_ASSIGN_TAB[1] = 2
$MC_AXCONF_GEOAX_ASSIGN_TAB[2] = 3

$MC_AXCONF_GEOAX_NAME_TAB[0] = "X"
$MC_AXCONF_GEOAX_NAME_TAB[1] = "Y"
$MC_AXCONF_GEOAX_NAME_TAB[2] = "Z"

$MC_TRAFO_GEOAX_ASSIGN_TAB_1[0]=4
$MC_TRAFO_GEOAX_ASSIGN_TAB_1[1]=5
$MC_TRAFO_GEOAX_ASSIGN_TAB_1[2]=6

$MC_TRAFO_AXES_IN_1[0]=4
$MC_TRAFO_AXES_IN_1[1]=5
$MC_TRAFO_AXES_IN_1[2]=6
$MC_TRAFO_AXES_IN_1[3]=1
$MC_TRAFO_AXES_IN_1[4]=2
```

程序：

程序代码

```
$P_NCBFRAME[0] = CTRANS(X,1,Y,2,Z,3,A,4,B,5,C,6)
$P_CHBFRAME[0] = CTRANS(X,1,Y,2,Z,3,A,4,B,5,C,6)
$P_IFRAME = CTRANS(X,1,Y,2,Z,3,A,4,B,5,C,6):CROT(Z,45)
$P_PFRAME = CTRANS(X,1,Y,2,Z,3,A,4,B,5,C,6):CROT(X,10,Y,20,Z,30)
```

程序代码	注释
TRAORI	; 坐标转换，设置 GEOAX(4,5,6) ; $P_NCBFRAME[0] = CTRANS(X,4,Y,5,Z,6,CAX,1,CAY,2,CAZ,3) ; $P_ACTBFRAME =CTRANS(X,8,Y,10,Z,12,CAX,2,CAY,4,CAZ,6) ; $P_PFRAME = CTRANS(X,4,Y,5,Z,6,CAX,1,CAY,2,CAZ,3):CROT(X,10,Y,20,Z,30)

程序代码	注释
	; $P_IFRAME = CTRANS(X,4,Y,5,Z,6,CAX,1,CAY,2,CAZ,3):CROT(Z,45)
TRAFOOF	；取消转换，设置 GEOAX(1,2,3)
	; $P_NCBFRAME[0] = CTRANS(X,1,Y,2,Z,3,A,4,B,5,C,6)
	; $P_CHBFRAME[0] = CTRANS(X,1,Y,2,Z,3,A,4,B,5,C,6)
	; $P_IFRAME = CTRANS(X,1,Y,2,Z,3,A,4,B,5,C,6):CROT(Z,45)
	; $P_PFRAME = CTRANS(X,1,Y,2,Z,3,A,4,B,5,C,6):CROT(X,10,Y,20,Z,30)

11.5.6.2 选择/取消坐标转换：常规

选择和取消坐标转换时，几何轴对通道轴的指定关系通常会发生变化。在将回转轴切换为直线轴，以及将直线轴切换为回转轴的转换中，无法将轴框架分量明确指定为几何轮廓框架分量。在这些非线性转换中，必须采用特殊措施来处理轮廓框架。

通过 MD10602 $MN_FRAME_GEOAX_CHANGE_MODE = 1 和 2 设置的模式需要进行扩展，从而将上述转换考虑在内。

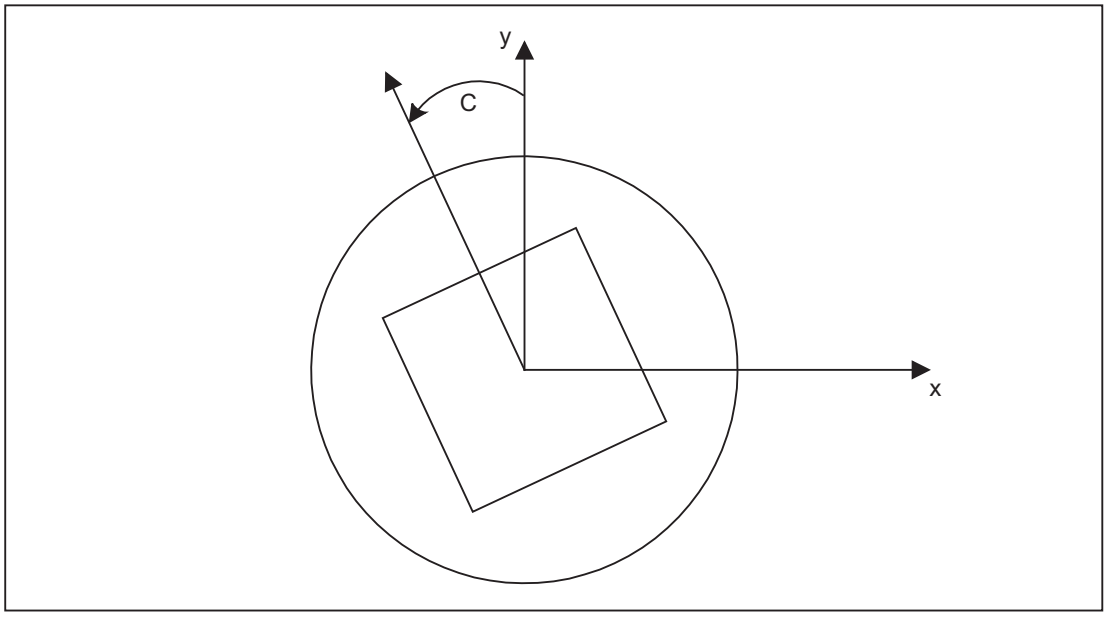
选择转换时，系统基于轴框架建立轮廓框架。此时 TRANSMIT、TRACYL 和 TRAANG 转换中的几何轴会受到特殊处理。

说明

带虚拟轴的转换

在选择 TRANSMIT 或 TRACYL 时，实际 Y 轴的偏移、比例缩放和镜像不接收到虚拟 Y 轴。TRAFOOF 时，虚拟 Y 轴的偏移、比例缩放和镜像会被删除。

11.5.6.3 选择/取消坐标转换：TRANSMIT



Transmit 扩展

可通过以下机床数据将 TRANSMIT 回转轴的轴总框架（即偏移、镜像和比例缩放）应用于转换：

- MD24905 $MC_TRANSMIT_ROT_AX_FRAME_1 = 1
- MD24955 $MC_TRANSMIT_ROT_AX_FRAME_2 = 1

例如可对框架链内一个框架中的工件倾斜角度进行补偿，从而输入回转轴的偏移。此偏移通常也用作坐标转换中的回转轴偏移。如上图所示的 C 轴偏移会产生对应的 X 值和 Y 值。

- MD24905 $MC_TRANSMIT_ROT_AX_FRAME_1 = 2
- MD24955 $MC_TRANSMIT_ROT_AX_FRAME_2 = 2

采用此设置时，回转轴的轴偏移乃至 SZS 都会在转换中被启用。SZS 框架中包含的回转轴轴偏移会作为旋转输入转换框架。仅当配置了转换框架时，此设置才生效。

框架扩展

下文介绍的扩展只适用于以下机床数据设置：

- MD10602 $MN_FRAME_GEOAX_CHANGE_MODE = 1
- MD10602 $MN_FRAME_GEOAX_CHANGE_MODE = 2

选择 TRANSMIT 转换时，生成一根通过回转轴耦合的虚拟几何轴，其不参考轴框架，而只在轮廓框架中被启用。几何值通过回转轴旋转得出。所有其他几何轴在选择转换时接收其轴分量。

分量：

- 偏移
  选择 TRANSMIT 时，虚拟轴的偏移会被删除。回转轴的偏移可在转换中启用。
- 旋转
  转换前编写的旋转会被接收。
- 镜像
  虚拟轴的镜像会被删除。回转轴的镜像可在转换中启用。
- 比例缩放
  虚拟轴的比例缩放会被删除。回转轴的比例缩放可在转换中启用。

示例：机床数据

```
; 框架配置

$MC_MM_SYSTEM_FRAME_MASK='H41'                ; TRAFRAME, SETFRAME
$MC_CHSFRAME_RESET_MASK='H41'                 ; 框架在复位后生效。
$MC_CHSFRAME_POWERON_MASK='H41'               ; 框架在上电时删除。

$MN_FRAME_GEOAX_CHANGE_MODE=1                 ; 框架在几何轴切换后进行换算。
$MC_RESET_MODE_MASK='H4041'                   ; 基本框架在复位后不取消。
;$MC_RESET_MODE_MASK='H41'                    ; 基本框架在复位后取消。

;$MC_GCODE_RESET_VALUES[7]=2                  ; G54 为缺省设置。
$MC_GCODE_RESET_VALUES[7]=1                   ; G500 为缺省设置。

$MN_MM_NUM_GLOBAL_USER_FRAMES=0
$MN_MM_NUM_GLOBAL_BASE_FRAMES=3

$MC_MM_NUM_USER_FRAMES=10                     ; 从 5 到 100
$MC_MM_NUM_BASE_FRAMES=3                      ; 从 0 到 8

$MN_NCBFRAME_RESET_MASK='HFF'
$MC_CHBFRAME_RESET_MASK='HFF'

$MN_MIRROR_REF_AX=0                           ; 镜像时无定标。
$MN_MIRROR_TOGGLE=0
$MN_MM_FRAME_FINE_TRANS=1                     ; 精偏
$MC_FRAME_ADD_COMPONENTS=TRUE                 ; 可使用 G58、G59。
```

```
; TRANSMIT 为第 1 转换
$MC_TRAFO_TYPE_1=256
```

```
$MC_TRAFO_AXES_IN_1[0]=1
$MC_TRAFO_AXES_IN_1[1]=6
$MC_TRAFO_AXES_IN_1[2]=3
$MC_TRAFO_AXES_IN_1[3]=0
$MC_TRAFO_AXES_IN_1[4]=0

$MA_ROT_IS_MODULO[AX6]=TRUE;

$MC_TRAFO_GEOAX_ASSIGN_TAB_1[0]=1
$MC_TRAFO_GEOAX_ASSIGN_TAB_1[1]=6
$MC_TRAFO_GEOAX_ASSIGN_TAB_1[2]=3

$MC_TRANSMIT_BASE_TOOL_1[0]=0.0
$MC_TRANSMIT_BASE_TOOL_1[1]=0.0
$MC_TRANSMIT_BASE_TOOL_1[2]=0.0

$MC_TRANSMIT_ROT_AX_OFFSET_1=0.0
$MC_TRANSMIT_ROT_SIGN_IS_PLUS_1=TRUE

$MC_TRANSMIT_ROT_AX_FRAME_1=1


; TRANSMIT 为第 2 转换
$MC_TRAFO_TYPE_2=256

$MC_TRAFO_AXES_IN_2[0]=1
$MC_TRAFO_AXES_IN_2[1]=6
$MC_TRAFO_AXES_IN_2[2]=2
$MC_TRAFO_AXES_IN_2[3]=0
$MC_TRAFO_AXES_IN_2[4]=0

$MC_TRAFO_GEOAX_ASSIGN_TAB_2[0]=1
$MC_TRAFO_GEOAX_ASSIGN_TAB_2[1]=6
$MC_TRAFO_GEOAX_ASSIGN_TAB_2[2]=2

$MC_TRANSMIT_BASE_TOOL_2[0]=4.0
$MC_TRANSMIT_BASE_TOOL_2[1]=0.0
$MC_TRANSMIT_BASE_TOOL_2[2]=0.0

$MC_TRANSMIT_ROT_AX_OFFSET_2=19.0
$MC_TRANSMIT_ROT_SIGN_IS_PLUS_2=TRUE
```

```
$MC_TRANSMIT_ROT_AX_FRAME_2=1
```

示例：零件程序

```
; 框架设置
N820 $P_UIFR[1] = ctrans(x,1,y,2,z,3,c,4)
N830 $P_UIFR[1] = $P_UIFR[1] : crot(x,10,y,20,z,30)
N840 $P_UIFR[1] = $P_UIFR[1] : cmirror(x,c)
N850
N860 $P_CHBFR[0] = ctrans(x,10,y,20,z,30,c,15)
N870


; 刀具选择、装夹补偿、平面选择
N890 T2 D1 G54 G17 G90 F5000 G64 SOFT
N900


; 逼近起始位置
N920 G0 X20 Z10
N930
N940 if $P_BFRAME <> CTRANS(X,10,Y,20,Z,30,C,15)
N950 setal(61000)
N960 endif
N970 if $P_BFRAME <> $P_CHBFR[0]
N980 setal(61000)
N990 endif
N1000 if $P_IFRAME <>
CTRANS(X,1,Y,2,Z,3,C,4):CROT(X,10,Y,20,Z,30):CMIRROR(X,C)
N1010 setal(61000)
N1020 endif
N1030 if $P_IFRAME <> $P_UIFR[1]
N1040 setal(61000)
N1050 endif
N1060 if $P_ACTFRAME <>
CTRANS(X,11,Y,22,Z,33,C,19):CROT(X,10,Y,20,Z,30):CMIRROR(X,C)
N1070 setal(61000)
N1080 endif
N1090
N1100 TRANSMIT(2)
N1110
N1120 if $P_BFRAME <> CTRANS(X,10,Y,0,Z,20,CAZ,30,C,15)
```

```
N1130 setal(61000)
N1140 endif
N1180 if $P_IFRAME <>
CTRANS(X,1,Y,0,Z,2,CAZ,3,C,4):CROT(X,10,Y,20,Z,30):CMIRROR(X,C)
N1190 setal(61000)
N1200 endif
N1240 if $P_ACTFRAME <>
CTRANS(X,11,Y,0,Z,22,CAZ,33,C,19):CROT(X,10,Y,20,Z,30):CMIRROR(X,C)
N1250 setal(61001)
N1260 endif
N1270
N1280
N1290 $P_UIFR[1,x,tr] = 11
N1300 $P_UIFR[1,y,tr] = 14
N1310
N1320 g54
N1330


; 设置框架
N1350 ROT RPL=-45
N1360 ATRANS X-2 Y10
N1370


; 四边粗加工
N1390 G1 X10 Y-10 G41 OFFN=1; 余量 1mm
N1400 X-10
N1410 Y10
N1420 X10
N1430 Y-10
N1440


; 换刀
N1460 G0 Z20 G40 OFFN=0
N1470 T3 D1 X15 Y-15
N1480 Z10 G41
N1490

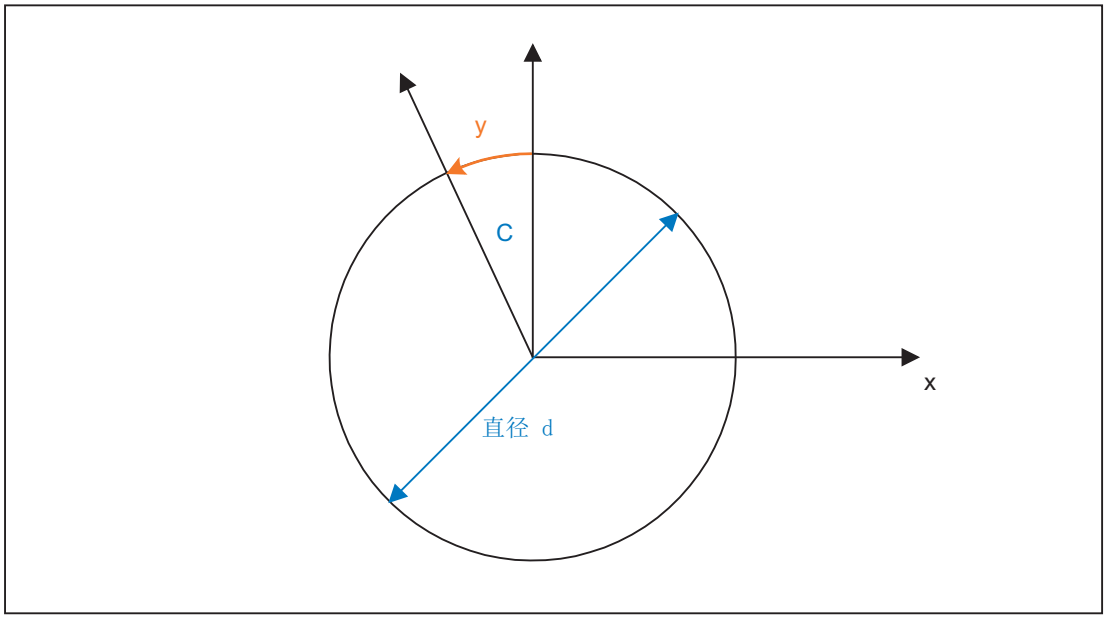

; 四边精加工
N1510 G1 X10 Y-10
```

```
N1520 X-10
N1530 Y10
N1540 X10
N1550 Y-10
N1560


; 取消框架
 N2950 m30 N1580 Z20 G40
N1590 TRANS
N1600
N1610 if $P_BFRAME <> CTRANS(X,10,Y,0,Z,20,CAZ,30,C,15)
N1620 setal(61000)
N1630 endif
N1640 if $P_BFRAME <> $P_CHBFR[0]
N1650 setal(61000)
N1660 endif
N1670 if $P_IFRAME <>
TRANS(X,11,Y,0,Z,2,CAZ,3,C,4):CROT(X,10,Y,20,Z,30):CMIRROR(X,C)
N1680 setal(61000)
N1690 endif
N1730 if $P_ACTFRAME <>
TRANS(X,21,Y,0,Z,22,CAZ,33,C,19):CROT(X,10,Y,20,Z,30):CMIRROR(X,C)
N1740 setal(61001)
N1750 endif
N1760
N1770 TRAFOOF
N1780
N1790 if $P_BFRAME <> CTRANS(X,10,Y,20,Z,30,C,15)
N1800 setal(61000)
N1810 endif
N1820 if $P_BFRAME <> $P_CHBFR[0]
N1830 setal(61000)
N1840 endif
N1850 if $P_IFRAME <>
TRANS(X,11,Y,2,Z,3,C,4):CROT(X,10,Y,20,Z,30):CMIRROR(X,C)
N1860 setal(61000)
N1870 endif
N1880 if $P_IFRAME <> $P_UIFR[1]
N1890 setal(61000)
N1900 endif
N1910 if $P_ACTFRAME <>
```

```
TRANS(X,21,Y,22,Z,33,C,19):CROT(X,10,Y,20,Z,30):CMIRROR(X,C)
N1920 setal(61002)
N1930 endif
N1940
N2010 $P_UIFR[1] = ctrans()
N2011 $P_CHBFR[0] = ctrans()
N2020 $P_UIFR[1] = ctrans(x,1,y,2,z,3,c,0)
N2021 G54
N2021 G0 X20 Y0 Z10 C0
N2030 TRANSMIT(1)
N2040 TRANS x10 y20 z30
N2041 ATRANS y200
N2050 G0 X20 Y0 Z10
N2051 if $P_IFRAME <> CTRANS(X,1,Y,0,Z,3,CAY,2)
N2052 setal(61000)
N2053 endif
N2054 if $P_ACTFRAME <> CTRANS(X,11,Y,20,Z,33,CAY,2):CFINE(Y,200)
N2055 setal(61002)
N2056 endif
N2060 TRAFOOF
N2061 if $P_IFRAME <> $P_UIFR[1]
N2062 setal(61000)
N2063 endif
N2064 if $P_ACTFRAME <> CTRANS(X,11,Y,2,Z,33):CFINE(Y,0)
N2065 setal(61002)
N2066 endif
```

11.5.6.4 选择/取消坐标转换：TRACYL



TRACYL 扩展

可通过以下机床数据将 TRACYL 回转轴的轴总框架（即偏移、精偏、镜像和比例缩放）应用于转换：

- MD24805 $MC_TRACYL_ROT_AX_FRAME_1 = 1
- MD24855 $MC_TRACYL_ROT_AX_FRAME_2 = 1

例如可对框架链内一个框架中的工件倾斜角度进行补偿，从而输入回转轴的偏移。此偏移通常也用作坐标转换中的回转轴偏移或 y 偏移。如上图所示的 C 轴偏移会产生对应的 X 值和 Y 值。

- MD24805 $MC_TRACYL_ROT_AX_FRAME_1 = 2
- MD24855 $MC_TRACYL_ROT_AX_FRAME_2 = 2

采用此设置时，回转轴的轴偏移乃至 SZS 都会在转换中被启用。SZS 框架中包含的回转轴轴偏移会作为外表面上的偏移输入转换框架。仅当配置了转换框架时，此设置才生效。

框架扩展

下文介绍的扩展只适用于以下机床数据设置：

- MD10602 $MN_FRAME_GEOAX_CHANGE_MODE = 1
- MD10602 $MN_FRAME_GEOAX_CHANGE_MODE = 2

选择 TRACYL 转换时，在外表面上生成一根通过回转轴耦合的虚拟几何轴，其不参考轴框架，而只在轮廓框架中被启用。虚拟几何轴的所有分量均会被删除。所有其他几何轴在选择转换时接收其轴分量。

分量：

- 偏移
  选择 TRACYL 时，虚拟轴的偏移会被删除。回转轴的偏移可在转换中启用。
- 旋转
  转换前编写的旋转会被接收。
- 镜像
  虚拟轴的镜像会被删除。回转轴的镜像可在转换中启用。
- 比例缩放
  虚拟轴的比例缩放会被删除。回转轴的比例缩放可在转换中启用。

示例：机床数据

```
; 框架配置

$MC_MM_SYSTEM_FRAME_MASK = 'H41'          ; TRAFRAME, SETFRAME
$MC_CHSFRAME_RESET_MASK = 'H41'           ; 框架在复位后生效。
$MC_CHSFRAME_POWERON_MASK='H41'           ; 框架在上电时删除。

$MN_FRAME_GEOAX_CHANGE_MODE = 1           ; 框架在几何轴切换后进行换算。

$MC_RESET_MODE_MASK='H4041'               ; 基本框架在复位后不取消。
;$MC_RESET_MODE_MASK = 'H41'              ; 基本框架在复位后取消。

;$MC_GCODE_RESET_VALUES[7]=2              ; G54 为缺省设置。
$MC_GCODE_RESET_VALUES[7] = 1             ; G500 为缺省设置。

$MN_MM_NUM_GLOBAL_USER_FRAMES = 0
$MN_MM_NUM_GLOBAL_BASE_FRAMES = 3

$MC_MM_NUM_USER_FRAMES = 10               ; 从 5 到 100
$MC_MM_NUM_BASE_FRAMES = 3                ; 从 0 到 8

$MN_NCBFRAME_RESET_MASK = 'HFF'
$MC_CHBFRAME_RESET_MASK = 'HFF'

$MN_MIRROR_REF_AX = 0                     ; 镜像时无定标。
$MN_MIRROR_TOGGLE = 0
$MN_MM_FRAME_FINE_TRANS = 1               ; 精偏
```

```
$MC_FRAME_ADD_COMPONENTS = TRUE            ; 可使用 G58、G59
```

```
; 带槽壁补偿的 TRACYL 为第 3 转换

$MC_TRAFO_TYPE_3 = 513; TRACYL

$MC_TRAFO_AXES_IN_3[0] = 1
$MC_TRAFO_AXES_IN_3[1] = 5
$MC_TRAFO_AXES_IN_3[2] = 3
$MC_TRAFO_AXES_IN_3[3] = 2

$MC_TRAFO_GEOAX_ASSIGN_TAB_3[0] = 1
$MC_TRAFO_GEOAX_ASSIGN_TAB_3[1] = 5
$MC_TRAFO_GEOAX_ASSIGN_TAB_3[2] = 3

$MC_TRACYL_BASE_TOOL_1[0] = 0.0
$MC_TRACYL_BASE_TOOL_1[1] = 0.0
$MC_TRACYL_BASE_TOOL_1[2] = 0.0

$MC_TRACYL_ROT_AX_OFFSET_1 = 0.0
$MC_TRACYL_ROT_SIGN_IS_PLUS_1 = TRUE

$MC_TRACYL_ROT_AX_FRAME_1 = 1
```

示例：零件程序

```
; 带槽壁补偿的简单运行测试
N450 G603
N460


; 框架设置
N500 $P_UIFR[1] = CTRANS(x,1,y,2,z,3,b,4)
N510 $P_UIFR[1] = $P_UIFR[1] : CROT(x,10,y,20,z,30)
N520 $P_UIFR[1] = $P_UIFR[1] : CMIRROR(x,b)
N530
N540 $P_CHBFR[0] = CTRANS(x,10,y,20,z,30,b,15)
N550
N560 G54
N570


; 连续路径运行，使用选择的精磨
```

```
N590 G0 x0 y0 z-10 b0 G90 F50000 T1 D1 G19 G641 ADIS=1 ADISPOS=5
N600
N610 if $P_BFRAME <> CTRANS(X,10,Y,20,Z,30,B,15)
N620 setal(61000)
N630 endif
N640 if $P_BFRAME <> $P_CHBFR[0]
N650 setal(61000)
N660 endif
N670 if $P_IFRAME <>
TRANS(X,1,Y,2,Z,3,B,4):CROT(X,10,Y,20,Z,30):CMIRROR(X,B)
N680 setal(61000)
N690 endif
N700 if $P_IFRAME <> $P_UIFR[1]
N710 setal(61000)
N720 endif
N730 if $P_ACTFRAME <>
TRANS(X,11,Y,22,Z,33,B,19):CROT(X,10,Y,20,Z,30):CMIRROR(X,B)
N740 setal(61000)
N750 endif
N760

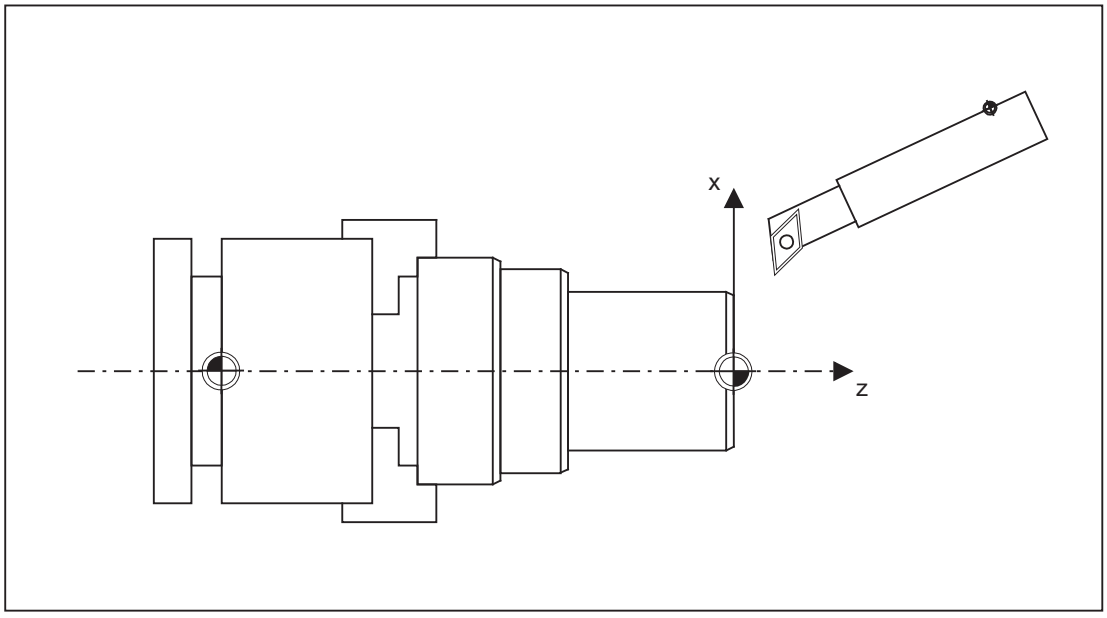

; 激活转换
N780 TRACYL(40.)
N790
N800 if $P_BFRAME <> CTRANS(X,10,Y,0,Z,30,CAY,20,B,15)
N810 setal(61000)
N820 endif
N830 if $P_CHBFR[0] <> CTRANS(X,10,Y,0,Z,30,CAY,20,B,15)
N840 setal(61000)
N850 endif
N860 if $P_IFRAME <>
TRANS(X,1,Y,0,Z,3,CAY,2,B,4):CROT(X,10,Y,20,Z,30):CMIRROR(X,B)
N870 setal(61000)
N880 endif
N890 if $P_UIFR[1] <>
TRANS(X,1,Y,0,Z,3,CAY,2,B,4):CROT(X,10,Y,20,Z,30):CMIRROR(X,B)
N900 setal(61000)
N910 endif
N920 if $P_ACTFRAME <>
TRANS(X,11,Y,0,Z,33,CAY,22,B,19):CROT(X,10,Y,20,Z,30):CMIRROR(X,B)
N930 setal(61001)
```

```
N940 endif
N950
N960 $P_UIFR[1,x,tr] = 11
N970 $P_UIFR[1,y,tr] = 14
N980
N990 g54
N1000
N1010 if $P_BFRAME <> CTRANS(X,10,Y,0,Z,30,CAY,20,B,15)
N1020 setal(61000)
N1030 endif
N1040 if $P_BFRAME <> $P_CHBFR[0]
N1050 setal(61000)
N1060 endif
N1070 if $P_IFRAME <>
TRANS(X,11,Y,0,Z,3,CAY,2,B,4):CROT(X,10,Y,20,Z,30):CMIRROR(X,B)
N1080 setal(61000)
N1090 endif
N1100 if $P_IFRAME <> $P_UIFR[1]
N1110 setal(61000)
N1120 endif
N1130 if $P_ACTFRAME <>
TRANS(X,21,Y,0,Z,33,CAY,22,B,19):CROT(X,10,Y,20,Z,30):CMIRROR(X,B)
N1140 setal(61001)
N1150 endif
N1160


; 取消转换
N1180 TRAFOOF
N1190
N1200 if $P_BFRAME <> CTRANS(X,10,Y,20,Z,30,B,15)
N1210 setal(61000)
N1220 endif
N1230 if $P_BFRAME <> $P_CHBFR[0]
N1240 setal(61000)
N1250 endif
N1260 if $P_IFRAME <>
TRANS(X,11,Y,2,Z,3,B,4):CROT(X,10,Y,20,Z,30):CMIRROR(X,B)
N1270 setal(61000)
N1280 endif
N1290 if $P_IFRAME <> $P_UIFR[1]
N1300 setal(61000)
```

```
N1310 endif
N1320 if $P_ACTFRAME <>
TRANS(X,21,Y,22,Z,33,B,19):CROT(X,10,Y,20,Z,30):CMIRROR(X,B)
N1330 setal(61002)
N1340 endif
N1350
N1360 G00 x0 y0 z0 G90
N1370
N1380 m30
```

11.5.6.5 选择/取消坐标转换：TRAANG



框架扩展：

下文介绍的扩展只适用于以下机床数据设置：

- MD10602 $MN_FRAME_GEOAX_CHANGE_MODE = 1
- MD10602 $MN_FRAME_GEOAX_CHANGE_MODE = 2

分量：

- 偏移
  选择 TRAANG 时，虚拟轴的偏移会被保留。
- 旋转
  转换前编写的旋转会被接收。

- 镜像
  虚拟轴的镜像会被接收。
- 比例缩放
  虚拟轴的比例缩放会被接收。

示例：机床数据

```
; 框架配置

$MC_MM_SYSTEM_FRAME_MASK = 'H1'              ; SETFRAME
$MC_CHSFRAME_RESET_MASK = 'H41'              ; 框架在复位后生效。
$MC_CHSFRAME_POWERON_MASK='H41'              ; 框架在上电时删除。

$MN_FRAME_GEOAX_CHANGE_MODE = 1              ; 框架在几何轴切换后进行换算。

$MC_RESET_MODE_MASK='H4041'                  ; 基本框架在复位后不取消。
;$MC_RESET_MODE_MASK = 'H41'                 ; 基本框架在复位后取消。

;$MC_GCODE_RESET_VALUES[7]=2                 ; G54 为缺省设置。
$MC_GCODE_RESET_VALUES[7] = 1                ; G500 为缺省设置。

$MN_MM_NUM_GLOBAL_USER_FRAMES = 0
$MN_MM_NUM_GLOBAL_BASE_FRAMES = 3

$MC_MM_NUM_USER_FRAMES = 10                  ; 从 5 到 100
$MC_MM_NUM_BASE_FRAMES = 3                   ; 从 0 到 8

$MN_NCBFRAME_RESET_MASK = 'HFF'
$MC_CHBFRAME_RESET_MASK = 'HFF'

$MN_MIRROR_REF_AX = 0                        ; 镜像时无定标。
$MN_MIRROR_TOGGLE = 0
$MN_MM_FRAME_FINE_TRANS = 1                  ; 精偏
$MC_FRAME_ADD_COMPONENTS = TRUE              ; 可使用 G58、G59。
```

```
; TRAANG 为第 1 转换

$MC_TRAFO_TYPE_1 = 1024

$MC_TRAFO_AXES_IN_1[0] = 4                   ; 斜轴
$MC_TRAFO_AXES_IN_1[1] = 3                   ; 轴平行于 z 轴
$MC_TRAFO_AXES_IN_1[2] = 2
$MC_TRAFO_AXES_IN_1[3] = 0
```

```
$MC_TRAFO_AXES_IN_1[4] = 0

$MC_TRAFO_GEOAX_ASSIGN_TAB_1[0] = 4
$MC_TRAFO_GEOAX_ASSIGN_TAB_1[1] = 2
$MC_TRAFO_GEOAX_ASSIGN_TAB_1[2] = 3

$MC_TRAANG_ANGLE_1 = 85.
$MC_TRAANG_PARALLEL_VELO_RES_1 = 0.
$MC_TRAANG_PARALLEL_ACCEL_RES_1 = 0.

$MC_TRAANG_BASE_TOOL_1[0] = 0.0
$MC_TRAANG_BASE_TOOL_1[1] = 0.0
$MC_TRAANG_BASE_TOOL_1[2] = 0.0
```

；TRAANG 为第 2 转换

```
$MC_TRAFO_TYPE_2 = 1024

$MC_TRAFO_AXES_IN_2[0] = 4
$MC_TRAFO_AXES_IN_2[1] = 3
$MC_TRAFO_AXES_IN_2[2] = 0
$MC_TRAFO_AXES_IN_2[3] = 0
$MC_TRAFO_AXES_IN_2[4] = 0

$MC_TRAFO_GEOAX_ASSIGN_TAB_2[0] = 4
$MC_TRAFO_GEOAX_ASSIGN_TAB_2[1] = 0
$MC_TRAFO_GEOAX_ASSIGN_TAB_2[2] = 3

$MC_TRAANG_ANGLE_2 = -85.
$MC_TRAANG_PARALLEL_VELO_RES_2 = 0.2
$MC_TRAANG_PARALLEL_ACCEL_RES_2 = 0.2

$MC_TRAANG_BASE_TOOL_2[0] = 0.0
$MC_TRAANG_BASE_TOOL_2[1] = 0.0
$MC_TRAANG_BASE_TOOL_2[2] = 0.0
```

示例：零件程序

```
; 框架设置
N820 $P_UIFR[1] = ctrans(x,1,y,2,z,3,b,4,c,5)
N830 $P_UIFR[1] = $P_UIFR[1] : crot(x,10,y,20,z,30)
N840 $P_UIFR[1] = $P_UIFR[1] : cmirror(x,c)
N850
```

```
N860 $P_CHBFR[0] = ctrans(x,10,y,20,z,30,b,40,c,15)
N870


; 刀具选择、装夹补偿、平面选择
N890 T2 D1 G54 G17 G90 F5000 G64 SOFT
N900


; 逼近起始位置
N920 G0 X20 Z10
N930
N940 if $P_BFRAME <> CTRANS(X,10,Y,20,Z,30,B,40,C,15)
N950 setal(61000)
N960 endif
N970 if $P_BFRAME <> $P_CHBFR[0]
N980 setal(61000)
N990 endif
N1000 if $P_IFRAME <>
TRANS(X,1,Y,2,Z,3,B,4,C,5):CROT(X,10,Y,20,Z,30):CMIRROR(X,C)
N1010 setal(61000)
N1020 endif
N1030 if $P_IFRAME <> $P_UIFR[1]
N1040 setal(61000)
N1050 endif
N1060 if $P_ACTFRAME <>
TRANS(X,11,Y,22,Z,33,B,44,C,20):CROT(X,10,Y,20,Z,30):CMIRROR(X,C)
N1070 setal(61000)
N1080 endif
N1090
N1100 TRAANG(,1)
N1110
N1120 if $P_BFRAME <> CTRANS(X,10,Y,20,Z,30,CAX,10,B,40,C,15)
N1130 setal(61000)
N1140 endif
N1150 if $P_BFRAME <> $P_CHBFR[0]
N1160 setal(61000)
N1170 endif
N1180 if $P_IFRAME <>
CTRANS(X,1,Y,2,Z,3,CAX,1,B,4,C,5):CROT(X,10,Y,20,Z,30):CMIRROR(X,CAX,C)
N1190 setal(61000)
N1200 endif
```

```
N1210 if $P_IFRAME <> $P_UIFR[1]
N1220 setal(61000)
N1230 endif
N1240 if $P_ACTFRAME <>
TRANS(X,11,Y,22,Z,33,CAX,11,B,44,C,20):CROT(X,10,Y,20,Z,
30):CMIRROR(X,CAX,C)
N1250 setal(61001)
N1260 endif
N1270
N1280
N1290 $P_UIFR[1,x,tr] = 11
N1300 $P_UIFR[1,y,tr] = 14
N1310
N1320 g54
N1330


; 设置框架
N1350 ROT RPL=-45
N1360 ATRANS X-2 Y10
N1370


; 四边粗加工
N1390 G1 X10 Y-10 G41 OFFN=1; 余量 1mm
N1400 X-10
N1410 Y10
N1420 X10
N1430 Y-10
N1440


; 换刀
N1460 G0 Z20 G40 OFFN=0
N1470 T3 D1 X15 Y-15
N1480 Z10 G41
N1490


; 四边精加工
N1510 G1 X10 Y-10
N1520 X-10
N1530 Y10
```

```
N1540 X10
N1550 Y-10
N1560


; 取消框架
N1580 Z20 G40
N1590 TRANS
N1600
N1610 if $P_BFRAME <> CTRANS(X,10,Y,20,Z,30,CAX,10,B,40,C,15)
N1620 setal(61000)
N1630 endif
N1640 if $P_BFRAME <> $P_CHBFR[0]
N1650 setal(61000)
N1660 endif
N1670 if $P_IFRAME <>
TRANS(X,11,Y,14,Z,3,CAX,1,B,4,C,5):CROT(X,10,Y,20,Z,30):CMIRROR(X,CAX,C)
N1680 setal(61000)
N1690 endif
N1700 if $P_IFRAME <> $P_UIFR[1]
N1710 setal(61000)
N1720 endif
N1730 if $P_ACTFRAME <>
TRANS(X,21,Y,34,Z,33,CAX,11,B,44,C,20):CROT(X,10,Y,20,Z,
30):CMIRROR(X,CAX,C)
N1740 setal(61001)
N1750 endif
N1760
N1770 TRAFOOF
N1780
N1790 if $P_BFRAME <> CTRANS(X,10,Y,20,Z,30,B,40,C,15)
N1800 setal(61000)
N1810 endif
N1820 if $P_BFRAME <> $P_CHBFR[0]
N1830 setal(61000)
N1840 endif
N1850 if $P_IFRAME <>
TRANS(X,1,Y,14,Z,3,B,4,C,5):CROT(X,10,Y,20,Z,30):CMIRROR(X,C)
N1860 setal(61000)
N1870 endif
N1880 if $P_IFRAME <> $P_UIFR[1]
N1890 setal(61000)
```

```
N1900 endif
N1910 if $P_ACTFRAME <>
TRANS(X,11,Y,34,Z,33,B,44,C,20):CROT(X,10,Y,20,Z,30):CMIRROR(X,C)
N1920 setal(61002)
N1930 endif
N1940
N1950 m30
```

11.5.6.6 调整生效框架

程序执行期间或复位时，几何轴配置有可能发生变化。届时几何轴的数量可能为零到三不等。在无几何轴存在的情况下，生效框架中的分量（例如旋转）可能导致生效框架对该轴配置无效。通过报警“通道 %1 程序段 %2 为不存在的几何轴编写了旋转”显示。此报警将一直存在，直至框架被相应修改。

通过以下机床数据可启用对生效框架的自动调整：

MD24040 $MC_FRAME_ADAPT_MODE，位 <n> = <值>

位	<值>	含义
0	1	生效框架中的坐标轴旋转没有可用的几何轴时，将该旋转从生效框架删除。
1	1	生效框架中的剪切角变为垂直角。
2	1	将生效框架中所有几何轴的比例缩放设为 1。

通过以下机床数据可以删除生效框架中所有引起不存在几何轴的轴运动的旋转：

MD24040 $MC_FRAME_ADAPT_MODE = 1

数据管理框架此时不会改变。激活数据管理框架时，只接收可实现的旋转。

示例

不存在 Y 轴：

- MD20050 $MC_AXCONF_GEOAX_ASSIGN_TAB[0] = 1
- MD20050 $MC_AXCONF_GEOAX_ASSIGN_TAB[1] = 0
- MD20050 $MC_AXCONF_GEOAX_ASSIGN_TAB[2] = 3
- $P_UIFR[1] = CROT(X,45,Y,45,Z,45)

程序代码	；注释
N390 G54 G0 X10 Z10 F10000	
IF $P_IFRAME <> CROT(Y,45)	；只接收围绕 Y 轴的旋转

程序代码	；注释
SETAL(61000)	
ENDIF	

11.5.6.7 映射框架

概述

“映射框架”功能用于在通道专用或全局数据管理框架内实现对轴框架的跨通道一致性修改。在哪些轴之间执行映射通过轴专用机床数据定义。

例如，为机床轴 AX1 和 AX4 启用了框架映射时，若在一个通道专用数据管理框架（例如基本框架 $P_CHBFR[x]）中修改了 AX1 的轴框架（偏移、精偏、比例缩放、镜像），那么 AX1 和 AX4 的这些框架数据将传输至其作为通道轴的所有通道的所有通道专用数据管理框架（例如基本框架 $P_CHBFR[x]）。

修改旋转的轴框架数据时不会进行框架映射。

前提条件

框架映射须满足下列前提条件：

- 用于映射的数据管理框架必须经过配置：
  MD28083 $MC_MM_SYSTEM_DATAFRAME_MASK（系统框架）
- **通道专用**数据管理框架须显性使能映射：
  MD10616 $MN_MAPPED_FRAME_MASK（框架映射使能）

  提示
  全局数据管理框架上总是会执行映射，无需使能。

参数设置

映射关系通过以下轴专用机床数据设置：

MD32075 $MA_MAPPED_FRAME[<AXn>] = "AXm"

AXn，AXm：机床轴名称，其中 n，m = 1, 2, ... 机床轴最大数量

映射规则

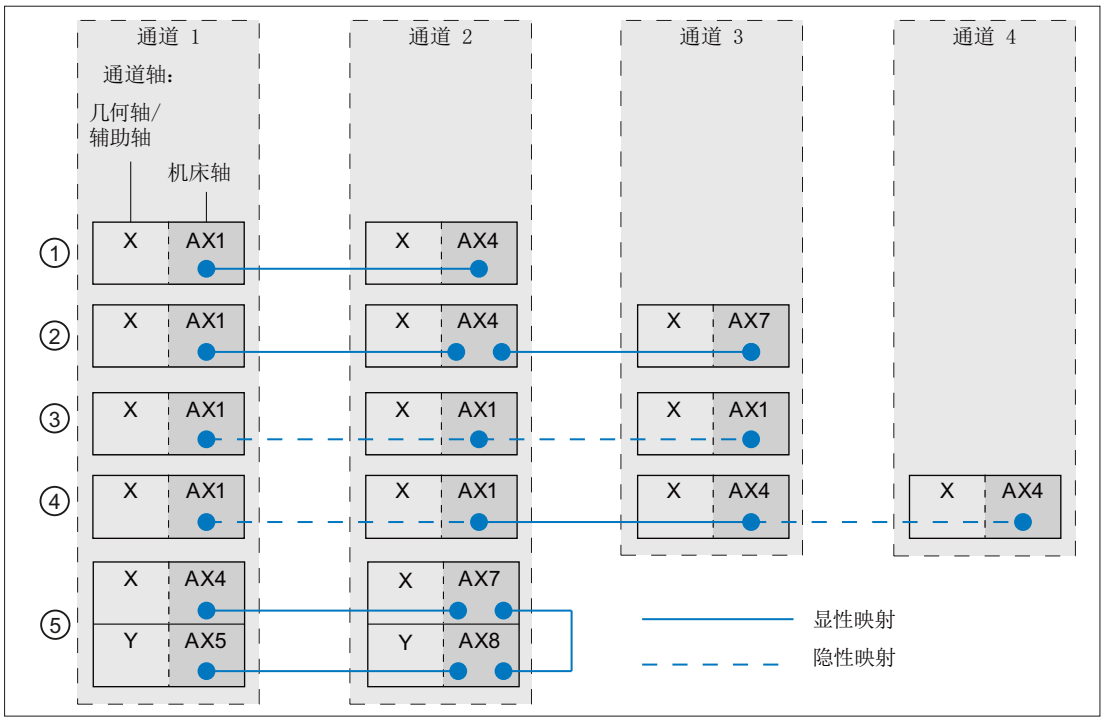
对于框架映射有以下规则：

- 映射为双向。
  可为轴 AXn 或 AXm 写入轴框架。框架数据总是为另一根轴接收。
- 系统总是会对设置的所有映射关系进行分析。
  在写入轴 AXn 的轴框架时，系统会分析其所有映射关系，并为所有直接或间接相关的轴接收框架数据。
- 映射为通道全局进行。
  在为一个通道专用框架写入轴 AXn 或 AXm 的轴框架时，系统会为 AXn 或 AXm 作为通道轴的所有通道接收框架数据。
- 在通过几何轴名称或辅助轴名称写入轴框架时，映射关系分析针对当前指定给该几何轴或辅助轴的机床轴进行。
- 映射针对特定框架。
  在写入轴框架时，框架数据映射只在相同的通道专用或全局数据管理框架内进行。

说明

数据一致性

用户/机床制造商须自行确保写入框架后所有通道中的框架数据一致，例如通过通道同步。



说明	参数设置：$MA_
① 简单映射关系： AX1(K1) ↔ AX4(K2)	MAPPED_FRAME[<AX1>] = "AX4"
② 级联映射关系： AX1(K1) ↔ AX4(K2) ↔ AX7(K3)	MAPPED_FRAME[<AX1>] = "AX4" MAPPED_FRAME[<AX4>] = "AX7"
③ 映射至自身，其中 AX1 作为通道 1、2、3 的通道轴： AX1(K1+K2+K3)	MAPPED_FRAME[<AX1>] = "AX1"
④ 两根轴之间的映射关系，分别为两个通道中的通道轴： AX1(K1+K2) ↔ AX4(K3+K4)	MAPPED_FRAME[<AX1>] = "AX4"
⑤ 级联映射关系，同个通道中写入多个通道轴： AX4(K1) ↔ AX7(K2) ↔ AX8(K2) ↔ AX5(K1)	MAPPED_FRAME[<AX4>] = "AX7" MAPPED_FRAME[<AX7>] = "AX8" MAPPED_FRAME[<AX8>] = "AX5"

图 11-25　映射示例

激活数据管理框架

数据管理框架可在零件程序中写入，或通过 SINUMERIK Operate 的操作界面写入。在通道中激活直接写入或通过框架映射写入的数据管理框架时，须注意以下几点：

- 在零件程序中写入
  数据管理框架必须在每个通道中显性激活（G500、G54 ... G599）。
- 通过操作界面写入
  数据管理框架通过操作界面写入，例如通过输入新的零点偏移。若该框架涉及的通道均不处于“通道生效”状态，那么修改过的数据管理框架将在这些通道中立即生效。若其中一个通道处于“通道生效”状态，那么数据管理框架不会在任何通道中生效。此时必须在每个通道中通过零件程序显性激活（G500、G54 ... G599）。或者其将在通道下一次切换至“通道复位”状态时生效。

示例

控制系统上设置了以下通道和通道轴：

- 通道 1
  - Z:几何轴
  - AX1:几何轴 Z 的机床轴
- 通道 2
  - Z:几何轴
  - AX4:几何轴 Z 的机床轴

两个通道中的 Z 轴零点总是相同：

- 映射关系：$MA_MAPPED_FRAME[AX1] = "AX4"

通道 1 和 2 中的零件程序

通道 1	通道 2
...	...
N100 WAIT (10,1,2)	N200 WAIT (10,1,2)
N110 $P_UIFR[1] = CTRANS(Z, 10)	
N120 WAIT (20,1,2)	N220 WAIT (20,1,2)
N130 G54	N230 G54
N140 IF ($P_IFRAME[0, Z, TR] <> 10)	N230 IF ($P_IFRAME[0, Z, TR] <> 10)
N150 SETAL(61000)	N250 SETAL(61000)
N160 ENDIF	N260 ENDIF
...	...

说明：

N100 / N200	通道同步，用于确保框架数据写入和映射的一致性
N110	写入可设定数据管理框架 $P_UIFR[1]： Z 轴零点偏移至 10 mm 轴框架数据映射： 通道 1：Z ≙ AX1 ⇔ 通道 2：Z ≙ AX4
N120 / N220	通道同步，用于确保新框架数据的一致激活
N130 / N230	激活新框架数据
N140 / N240	检查 Z 轴零点：10 mm

11.5.7 预定义框架功能

11.5.7.1 反转框架

INVFRAME() 功能从一个框架计算出相应的反框架。

功能说明

一个框架与其反转框架级联时总是得到零框架
`FRAME : INVFRAME( FRAME )` □ 零框架

框架反转是用于坐标转换的辅助功能。测量框架的计算通常在 WCS 中进行。若需将计算出的框架转换至另一坐标系，即将计算出的框架输入框架链内的一个任意框架，则可通过以下计算实现：

新的总框架由旧的总框架与计算出的框架级联得出：

```
$P_ACTFRAME = $P_ACTFRAME : $AC_MEAS_FRAME
```

框架链中的新框架如下得出：

- 目标框架为 $P_SETFRAME：

```
$P_SETFRAME = $P_ACTFRAME : $AC_MEAS_FRAME :
INVFRAME($P_ACTFRAME) : $P_SETFRAME
```

- 目标框架为第 n 个通道基本框架 $P_CHBFRAME[<n>]：

```
k = $MN_MM_NUM_GLOBAL_BASE_FRAMES
```

  - n = 0 时，TMP 按如下方式得出：

```
TMP = $P_PARTFRAME : $P_SETFRAME : $P_EXTFRAME :
$P_NCBFRAME[<0...k>]
```

  - n ≠ 0 时，TMP 按如下方式得出：

```
TMP = $P_PARTFRAME : $P_SETFRAME : $P_EXTFRAME :
$P_NCBFRAME[<0..k>] : $P_CHBFRAME[<0...n> - 1]
```

```
$P_CHBFRAME[<n>] = INVFRAME(TMP) : $P_ACTFRAME :
$AC_MEAS_FRAME : INVFRAME($P_ACTFRAME) : TMP : $P_CHBFRAME[<n>]
```

- 目标框架为 $P_IFRAME：

```
TMP =  $P_PARTFRAME : $P_SETFRAME : $P_EXTFRAME : $P_BFRAME
$P_IFRAME = INVFRAME(TMP) : $P_ACTFRAME : $AC_MEAS_FRAME :
INVFRAME($P_ACTFRAME) : TMP : $P_IFRAME
```

示例

需要在当前 SETFRAME 中输入一个框架（例如通过测量功能测得），使新的总框架为旧的总框架和测量框架的级联。SETFRAME 通过框架反转进行相应换算。

程序代码	注释
DEF INT RETVAL	
DEF FRAME TMP	
$TC_DP1[1,1]=120	; 类型
$TC_DP2[1,1]=20.	; 0
$TC_DP3[1,1]=10.	; (z) 长度补偿矢量
$TC_DP4[1,1]= 0.	; (y)
$TC_DP5[1,1]= 0.	; (x)
$TC_DP6[1,1]= 2.	; 半径
T1 D1	
g0 x0 y0 z0 f10000	
G54	
$P_CHBFRAME[0] = CROT(Z,45)	
$P_IFRAME[X,TR] = -SIN(45)	
$P_IFRAME[Y,TR] = -SIN(45)	
$P_PFRAME[Z,TR] = -45	
$AC_MEAS_VALID = 0	; 通过 4 个测量点测量拐角

程序代码	注释
G1 X-1 Y-3	; 逼近第 1 测量点
$AC_MEAS_LATCH[0] = 1	; 保存第 1 测量点
G1 X5 Y-3	; 逼近第 2 测量点
$AC_MEAS_LATCH[1] = 1	; 保存第 2 测量点
G1 X-4 Y4	; 逼近第 3 测量点
$AC_MEAS_LATCH[2] = 1	; 保存第 3 测量点
G1 X-4 Y1	; 逼近第 4 测量点
$AC_MEAS_LATCH[3] = 1	; 保存第 4 测量点
$AA_MEAS_SETPOINT[X] = 0	; 设置拐角的设定位置
$AA_MEAS_SETPOINT[Y] = 0	
$AA_MEAS_SETPOINT[Z] = 0	
$AC_MEAS_CORNER_SETANGLE = 90	; 设置交角设定值
$AC_MEAS_WP_SETANGLE = 30	
$AC_MEAS_ACT_PLANE = 0	; 测量平面为 G17
$AC_MEAS_T_NUMBER = 1	; 选择刀具
$AC_MEAS_D_NUMBER = 1	
$AC_MEAS_TYPE = 4	; 将测量类型设置为“拐角 1”
RETVAL = MEASURE()	; 开始测量
IF RETVAL <> 0	
SETAL(61000 + RETVAL)	
ENDIF	
IF $AC_MEAS_WP_ANGLE <> 30	
SETAL(61000 + $AC_MEAS_WP_ANGLE)	
ENDIF	
IF $AC_MEAS_CORNER_ANGLE <> 90	
SETAL(61000 + $AC_MEAS_CORNER_ANGLE)	
ENDIF	
; 对测得的框架进行转换并写入 $P_SETFRAME，使新的总框架	
; 为旧的总框架和测量框架的级联。	
$P_SETFRAME = $P_ACTFRAME : $AC_MEAS_FRAME : INVFRAME($P_ACTFRAME) : $P_SETFRAME	
$P_SETFR = $P_SETFRAME	; 写入数据管理中的系统框架
G1 X0 Y0	; 逼近拐角
G1 X10	; 沿旋转 30 度的矩形退回
Y10	
X0	
Y0	
M30	

11.5.7.2 框架链中的叠加框架

通过工件上的测量或零件程序或循环中的计算得到的框架通常是叠加于生效的总框架生效。因此，WCS 和编程的零点需要进行偏移和/或旋转。此时，测量的框架位于框架变量中且不算入框架链内。

功能说明

ADDFRAME() 功能从一个临时框架计算出通过参数 <STRING> 指定的目标框架，这样，新生效的总框架 $P_ACTFRAME 便可由旧生效总框架与临时框架的级联得出：

ERG = ADDFRAME(TMPFRAME,"$P_SETFRAME") ⇒ $P_SETFRAME$_{新}$ = $P_SETFRAME$_{旧}$ ADD TMPFRAME 和 $P_ACTFRAME$_{新}$ = $P_ACTFRAME$_{旧}$: TMPFRAME

如果指定了生效的框架作为目标框架，那么新的总框架**在预处理中**生效。

若目标框架为数据管理框架，那么必须在通道中（例如：零件程序、循环）显性激活才能使其生效。

该功能会反馈可在用户程序中对其作出响应的返回值。

编程

句法

INT ADDFRAME(<FRAME>,<STRING>)

含义

<FRAME>:	带需要额外算入的值的框架变量	
	类型	FRAME
<STRING>:	生效框架或数据管理框架的名称： • 生效框架 "$P_CYCFRAME", "$P_ISO4FRAME", "$P_PFRAME", "$P_WPFRAME", "$P_TOOLFRAME", "$P_IFRAME", "$P_GFRAME", "$P_CHBFRAME[<n>]", "$P_NCBFRAME[<n>]", "$P_ISO1FRAME", "$P_ISO2FRAME", "$P_ISO3FRAME", "$P_EXTFRAME", "$P_SETFRAME", "$P_PARTFRAME" • 数据管理框架 "$P_CYCFR", "$P_ISO4FR, "$P_TRAFR", "$P_WPFR", "$P_TOOLFR", "$P_UIFR[<n>]", "$P_GFR", "$P_CHBFR[<n>]", "$P_NCBFR[<n>]", "$P_ISO1FR, "$P_ISO2FR, "$P_ISO3FR, "$P_EXTFR", "$P_SETFR", "$P_PARTFR"	
	类型	STRING

返回值：	允许的返回值： • 0：正常 • 1：目标设定（字符串）错误 • 2：目标框架未配置 • 3：框架中的旋转不被允许	
	类型	INT

11.5.8 功能

11.5.8.1 设置零点、工件测量和刀具测量

实际值设定通过 HMI 或测量循环进行。计算出的框架会被写入系统框架 SETFRAME。设定实际值时，可对 WCS 中的轴设定位置进行修改。

“对刀”这一概念表示的是工件测量和刀具测量。进行工件测量时，可以测量工件一条边沿、一个拐角或一个钻孔的位置。为确定工件或钻孔的零点，可将实测位置和 WCS 中的设定位置相比。由此得出的偏移可输入到选中的框架中。进行刀具测量时，可根据一个经过测量的标准件测量刀具的长度或半径。

测量可通过操作或测量循环进行。与 NC 的通讯通过预定义系统变量进行。在 NC 中，通过 HMI 操作激活 PI 通讯，或通过测量循环中的零件程序指令来执行计算。可选择一把刀具或一个平面作为计算的基础。计算出的框架会被输入结果框架。

11.5.8.2 轴外部零点偏移

机床数据

通过以下机床数据激活外部零点偏移或系统框架 $P_EXTFRAME：

MD28082 $MC_MM_SYSTEM_FRAME_MASK，位 1 = TRUE

外部零点偏移的值可使用 OPI 通过 HMI 操作界面和 PLC 用户程序手动设定，或在零件程序中通过轴专用系统变量 $AA_ETRANS [<轴>] 编写。

激活

外部零点偏移通过以下接口信号激活：
DB31, ... DBX3.0（接收外部零点偏移）

特性

外部零点偏移激活时，系统会立即停止所有轴（除去指令轴和 PLC 轴）的运行并重组预处

理。当前系统框架和数据管理中系统框架的粗偏会被设置为轴专用系统变量 $AA_ETRANS[<轴>] 的值。之后系统会先运行偏移，然后再恢复中断的运动。

采用增量尺寸设定时的特性

采用增量尺寸设定 G91 和以下机床数据设置时：

MD42440 $MC_FRAME_OFFSET_INCR_PROG（框架中的零点偏移）= 0

系统会与机床数据配置相反通过逼近程序段在“通过系统框架实现外部零点偏移”范围内运行偏移，尽管偏移通过框架设定。

说明

外部零点偏移总是作为绝对值生效。

11.5.8.3 刀架

偏移

在“P”和“M”类型的运动系统中，选择刀架时会激活一个叠加框架（可定向刀架的工作台偏移），其将零点偏移作为工作台旋转的后果。零点偏移会被输入系统框架 $P_PARTFR。此时该框架的偏移分量被改写。其他框架分量则保持不变。

系统框架 $P_PARTFR 必须通过以下机床数据使能：

MD28082 $MC_MM_SYSTEM_FRAME_MASK，位 2 = 1（用于 TCARR 和 PAROT 的系统框架）

说明

也可通过以下机床数据来设置偏移，用于接收工作台偏移：

MD20184 $MC_TOCARR_BASE_FRAME_NUMBER = <基本框架的编号>

由于兼容性原因，只能在较老的软件版本中使用此方案。建议不采用此方案。

刀架切换引起的框架偏移会在通过 TCARR=... 进行选择时立即生效。而对刀具长度的修改仅在刀具生效时才会立即生效。

激活不会触发框架旋转，已生效的旋转也不会改变。和 T 情形（仅刀具可旋转）相似，用于计算的回转轴位置根据 TCOFR / TCOABS 指令从生效框架的旋转分量或 $TC_CARR 的记录得出。激活框架会使工件坐标系中的位置相应改变，而不会引起机床的补偿运动。

情形如下图所示：

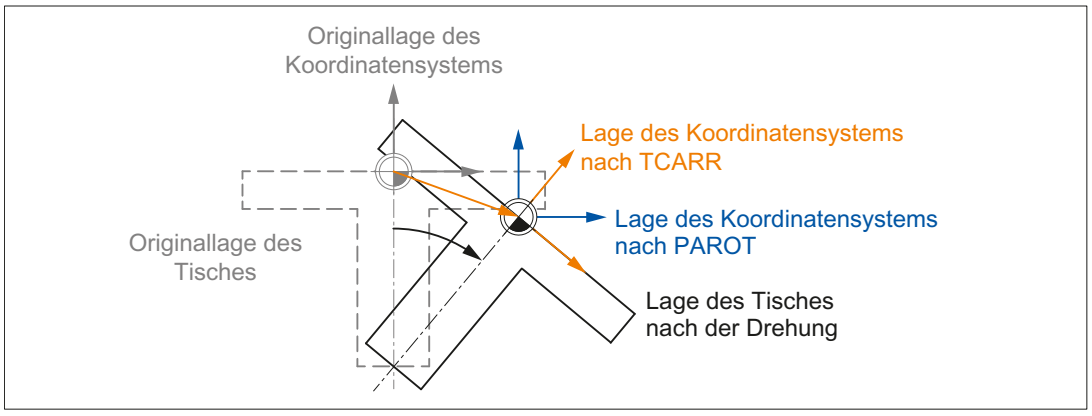


图 11-26　通过 TCARR 激活可旋转刀具工作台时的框架

在运动类型为 M 时（刀具和平台可以分别绕轴旋转），使用 TCARR 激活刀架会导致有效的刀具长度（如果刀具有效的话）和零点偏移同时发生变化。

旋转

根据加工任务，在使用可旋转刀架或工作台时，不仅需要考虑零点偏移（作为框架或作为刀具长度），还需考虑旋转。然而激活一个可定向刀架在任何情况下都不会直接导致坐标系旋转。

若只有刀具可旋转，可通过 TOFRAME 或 TOROT 为其定义一个框架。

对于可旋转工作台（运动系统类型 P 和 M），通过 TCARR 激活最先同样不会引起旋转，也就是说，尽管在以机床为基准时坐标系发生了偏移，但以工件零点为基准时其保持固定，定向也保持不变。

若需相对工件固定的坐标系，即相对原始位置不仅发生偏移，还根据工作台旋转进行旋转，则可类似采用可旋转刀具时的情形通过 PAROT 激活相应旋转。

使用 PAROT 指令时，生效框架中的偏移、比例缩放和镜像保留，旋转分量则由可定向刀架的旋转分量替代，该刀架依据工作台旋转。此时总体可编程框架保持不变，包括其旋转分量。

描述刀具工作台旋转的旋转分量可输入系统框架 $PARTFR，或输入通过 MD20184 $MC_TOCARR_BASE_FRAME_NUMBER 设置的基本框架。

$MC_MM_SYSTEM_FRAME_MASK，位 2 = <值>

值	含义
1	旋转分量 → $PARTFR
0	旋转分量 → MD20184 $MC_TOCARR_BASE_FRAME_NUMBER

依据描述工作台平移时的提示，建议不要为新设备采用第二种方案。

Partframe 的旋转分量可通过 PAROTOF 删除，与该框架在基本框架还是系统框架中无关。偏移分量通过激活不触发偏移的刀架来删除，或通过 TCARR=0 取消可能生效的可定向刀架来删除。

在回转台或刀具通过两根旋转轴定位的情形下，PAROT 或 TOROT 会考虑总体定向变化。在混合运动中只考虑由旋转轴引起的、相应部分的情况。这样一来，例如可在使用 TOROT 时旋转工件，使斜向平面平行于空间固定的 X-Y 平面；加工时则须考虑刀具旋转，例如须加工不垂直于该平面的钻孔时。

示例

机床中回转台的旋转轴指向 Y 轴正方向。工作台已旋转 +45 度。之后通过 PAROT 定义一个框架，同样为围绕 Y 轴的 45 度旋转。相对于外界未旋转的坐标系（在图中用“ TCARR 之后的坐标系位置”进行标识）相对于一起运动的坐标系（PAROT 之后的位置）却旋转了 -45 度。若通过 ROT Y-45 定义此坐标系，之后在 TCOFR 生效时选择刀架，则会为刀架的旋转轴得到 +45 度的角度。

无可定向刀架生效时，语言指令 PAROT 不会被系统拒绝。但是此类调用不会引起框架变化。

在刀具方向上加工

首先，在配备可定向刀具的机床上，在未激活框架（例如通过 TOFRAME 或 TOROT）的情况下有时需要沿刀具方向运行（钻削时常见），并且其中一根轴指向刀具方向。当斜向加工中定义斜面的框架生效，但是索引刀架（端面齿）使刀具定向无法任意设置，因此无法完全垂直设置刀具时，也会出现问题。在这些情形下必须沿刀具方向钻孔，与原本要求的垂直于平面的运动有偏差，否则钻头无法向其纵轴方向引导（刀具断裂）。

增量运行

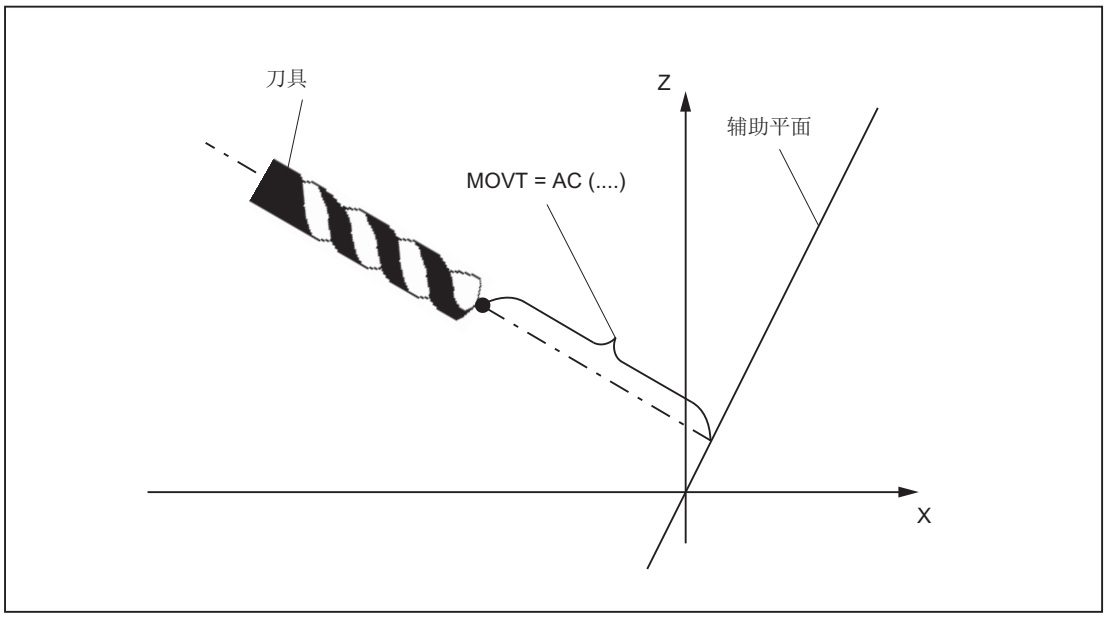
沿刀具方向的增量运行的终点通过 MOVT = <值> 或 MOVT=IC(<值>) 编写。

此时正运行方向定义为刀尖指向刀架的的方向。这对应平行于轴的加工，例如通过 G91 Z...。

绝对运行

沿刀具方向的绝对运行的终点通过 MOVT=AC(<值>) 编写。

为此定义一个穿过当前零点的平面，其表面法线矢量平行于刀具定向。MOVT 指定的位置将基于该平面（见图）。参考轴用于计算最终位置。生效框架不受这些内部计算影响。



MOVT 编程与是够存在可设定刀架无关。运动方向取决于生效的平面。
其沿垂直坐标的方向运行，即 G17 中沿 Z 轴方向，G18 中沿 Y 轴方向，G19 中沿 X 轴方向。这既适用于无可定向刀架生效的情形，也适用于可定向刀架无/有可旋转刀具的情形。

定向转换（3-4-5 轴转换）生效时，MOVT 以相同方式生效。

若在编写了 MOVT 的程序段中同时修改了刀具定向（例如在 5 轴转换生效时通过同时插补回转轴），那么程序段开始处的定向决定 MOVT 的运行方向。
5 轴转换生效时，刀具中心点不会因为定向改变受到影响，即轨迹保持为直线，方向由程序段开始处的刀具定向确定。

若编写了 MOVT，线性插补和样条插补（G0、G1、ASPLINE、BSPLINE、CSPLINE）必须生效。否则会输出报警。
若有样条插补方式生效，那么得到的轨迹通常非直线，因为由 MOVT 得到的终点作为通过 X、Y、Z 显性编写的终点处理。

在包含 MOVT 的程序段中不允许编写几何轴。

通过空间角定义框架旋转

若需定义围绕多于一根轴进行旋转的框架，可通过将单次旋转级联实现。其中在经旋转的新坐标系中进行后续旋转。

这既适用于一个程序段中的编程，也适用于在多个连续程序段中建立框架：

- 一个程序段：N10 ROT X...Y...Z...
- 多个连续的程序段：

```
N10 ROT Y...
N20 AROT X...
N30 AROT Z...
```

空间角

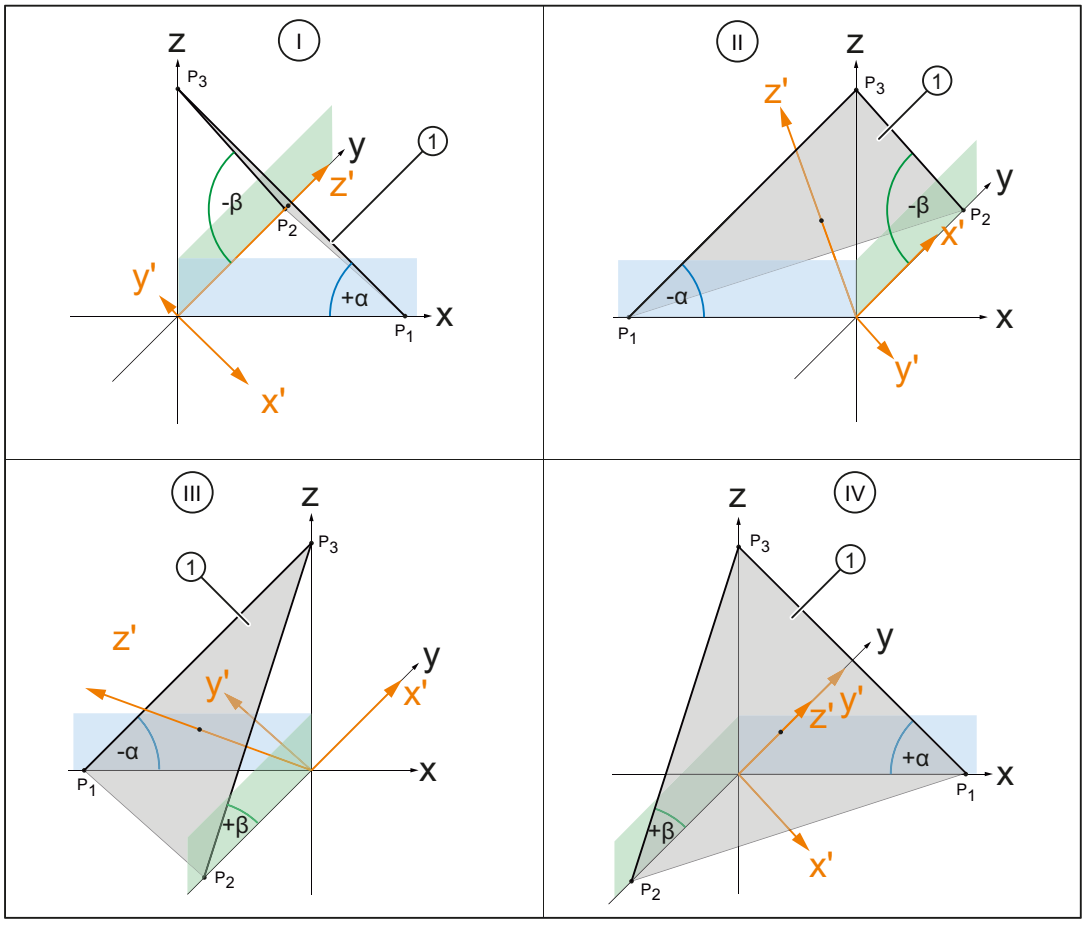
在工件图纸中，为描述斜面常会给出空间角。空间角是斜面与工件坐标系的主平面（X-Y、Y-Z、Z-X 平面）的交线形成的角度（见下图）。通过设定两个空间角在空间中对平面进行定位。第三个空间角由前两个得出。

借助 ROTS、AROTS 和 CROTS 指令，可将旋转直接描述为空间角。

允许设定单个空间角。在此情形下通过 ROTS 或 AROTS 执行的旋转与 ROT 或 AROT 时相同。

指令中编写的两个轴定义一个平面。未编写的轴则定义对应的笛卡尔坐标系第三轴。这样一来，对于编写的两个轴亦明确定义第一轴和第二轴。其中定义对应 G17/G18/G19 中的平面定义。

通过平面轴的轴字母编写的角度可指定一根轴，平面的另一根轴必须围绕该轴旋转从而转换为交线，交线与通过另一根轴和第三根轴组成的平面一同构成旋转过的平面。在编写两个角度中有一个趋近于零的情况下，通过此定义可确保以此方式定义的平面能够转换为只编写一根轴（例如通过 ROT 或 AROT）的平面。



I, ..., IV　象限 1 至 4

① 斜面作为针对新 G17 平面的设定

α, β 斜面的空间角

图 11-27　围绕空间角旋转

图中在象限 I 至 IV 中示出示例性平面的空间角。该斜面定义工件坐标系 WCS 旋转后 G17 平面的定向。空间角的符号给出了坐标系围绕相应轴旋转时所围绕的方向：

1. 围绕 y 轴旋转：
   工件坐标系 WCS 围绕 y 轴旋转符号相关的角度 α ⇒
   x' 轴平行于 xz 平面与该斜面的交线（共线）
2. 围绕 x' 轴旋转：
   工件坐标系 WCS' 围绕 x' 轴旋转符号相关的角度 β ⇒
   - y' 轴平行于 zy 平面与该斜面的交线（共线）
   - z' 轴垂直于该斜面
   - G17' 平行于该斜面

针对每个象限，用于编写工件坐标系 WCS 的 G17 平面朝斜面的定向的相应编程为：

- 象限 I：ROTS X<+α> Y<-β>
- 象限 II：ROTS X<-α> Y<-β>
- 象限 III：ROTS X<-α> Y<+β>
- 象限 IV：ROTS X<+α> Y<+β>

定向

设定空间角时，平面内的二维坐标系定向（即围绕表面法线矢量的旋转角）未定义。因此在定义坐标系的位置时，须使旋转过的第一轴位于未旋转的坐标系的第一轴和第三轴所构成的平面内。

这表示：

- 编写 X 和 Y 时，新的 X 轴位于原本的 Z-X 平面中。
- 编写 Z 和 X 时，新的 Z 轴位于原本的 Y-Z 平面中。
- 编写 Y 和 Z 时，新的 Y 轴位于原本的 X-Y 平面中。

如果需要与预设偏离的坐标系位置，必须通过 AROT... 执行叠加旋转。

参数设置：ZY'X"（RPY 角）或 ZX'Z" 转换

在输入时，系统根据以下机床数据将编写的空间角换算成 zy'x" 转换（RPY 角）或 zx'z"转换后的等效欧拉角：

MD10600 $MN_FRAME_ANGLE_INPUT_MODE

沿刀具方向进行框架旋转

通过旧有软件版本中即已提供的语言指令 TOFRAME，可定义一个 Z 轴指向刀具方向的框架。此时已编写的框架会被改写为一个纯粹描述旋转的框架。之前生效的框架中可能存在的零点偏移、镜像和比例缩放均会被删除。
此特性有时会造成干扰。特别是在经常需要保留通过工件中的参考点定义的零点偏移时。

因此系统又引入了语言指令 TOROT，在编写的框架中该指令只改写旋转分量，其他分量则保持不变。通过 TOROT 定义的旋转与使用 TOFRAME 时相同。
同 TOFRAME 一样，TOROT 也与是否存在可定向刀架无关。此语言指令还可在 5 轴转换中使用。

此外，在可定向刀架生效时，通过新的语言指令 TOROT 可为每种运动系统类型实现一致性的编程。

通过 TOFRAME 或 TOROT 定义 Z 方向指向刀具方向的框架。此定义适用于通常采用 G17 平面的铣削加工。对于车削加工或在 G18 或 G19 平面进行的加工，定义的框架能在 X 轴或 Y 轴对准将更有意义。为此 G 组 53 中存在以下指令：

- TOFRAMEX、TOFRAMEY、TOFRAMEZ
- TOROTX、TOROTY、TOROTZ

通过这些指令可定义相应的框架。此时 TOFRAME 和 TOFRAMEZ，以及 TOROT 和 TOROTZ 的功能相同。

通过 TOROT 或 TOFRAME 生成的框架可写入到独立的系统框架（$P_TOOLFR）中。可编程的框架保持不变。

- 前提条件：MD28082 $MC_MM_SYSTEM_FRAME_MASK，位 3 = 1

在编写 TOROT 或 TOFRAME 等指令时，有系统框架或无系统框架时的特性相同。继续处理可编程框架时才会出现差别。

说明

建议在新设备上对通过 G 功能组 53 的指令生成的框架只使用为其设计的系统框架。

示例

TOROT 后编写 TRANS。通过不设定参数的 TRANS 删除可编程框架。在无系统框架的方案中，通过 TOROT 产生的可编程框架的框架分量也会因此被删除；若 TOROT 分量处于系统框架中则会被保留。

TOROT 或 TOFRAME 等指令通过语言指令 TOROTOF 取消。TOROTOF 会删除总系统框架 $P_TOOLFR。若 TOFRAME 等指令描述的不是系统框架，而是可编程框架（旧方案），TOROT 只会删除旋转分量，其他框架分量保持不变。

若激活语言指令 TOFRAME 或 TOROT 前已有旋转框架生效，通常要求新定义的框架与旧框架的偏差尽可能小。例如，由于端面齿回转轴使刀具定向无法任意设置，必须微调框架定义时，便是此类情形。通过上述语言指令可将新框架的 Z 方向定义为唯一值。

参数设置：TOFRAME、TOROT 和 PAROT 中的框架定义（SD42980）

在借助 TOFRAME、TOROT（TOROTY、TOROTX）或 PAROT 定义框架时，通过以下设定数据定义当前加工平面的几何轴（G17：X 轴和 Y 轴）的方向。

在进行框架计算时，如此定义刀具方向，使得刀具方向平行于框架的垂直坐标（G17：Z 轴）并且垂直于加工平面。

围绕刀具轴的旋转最初是任意的。可通过此设定数据决定该自由旋转，使得新定义的框架与先前生效框架的偏差尽可能小。

在此设定数据不等于零的所有情形下，若新框架与旧框架的刀具方向一致，则生效的框架保持不变。

SD42980 $SC_TOFRAME_MODE

TCARR（请求刀架）和 PAROT（在工件上对准工件坐标系）

TCARR 使用通过以下机床数据标识的基本框架：
MD20184 $MC_TOCARR_BASE_FRAME_NUMBER

为了避免与已使用所有基本框架的系统冲突，可为 TCARR 和 PAROT 创建一个独立的系统框架。

PAROT、TOROT 和 TOFRAME 目前用于改写可编程框架的旋转分量。在此情形下无法单独取消 PAROT 或 TOROT。复位后可编程框架会被删除，这意味着运行方式切换至 JOG 后 PAROT 和 TOROT 的旋转分量将不再存在。此外用户必须能无限制地访问可编程框架。通过 PAROT 和 TOROT 生成的框架必须通过数据备份存档和重新载入。

TCARR 和 PAROT 的系统框架如下配置：
MD28082 $MC_MM_SYSTEM_FRAME_MASK，位 2 = 1

此时以下机床数据将不再被分析：
MD20184 $MC_TOCARR_BASE_FRAME_NUMBER

若为 TCARR 配置了系统框架，那么 TCARR 和 PAROT 将描述对应的系统框架，否则描述的是通过机床数据 MD20184 标识的基本框架。

使用 P 类型和 M 类型的运动系统时，TCARR 将可定向刀架的工作台偏移（工作台旋转引起的零点偏移）作为偏移输入系统框架。PAROT 对系统框架进行换算，从而得到基于工件的工件坐标系。

系统框架采用剩余存储，并在复位后保留。即便在运行方式切换时系统框架也保持生效。

PAROT 和 TOROT、TOFRAME 分别被指定给一个独立 G 组用于显示。

PAROTOF

PAROTOF 是 PAROT 的取消指令。该指令会删除 PAROT 的系统框架中的旋转。此时当前 $P_PARTFRAME 和数据管理框架 $P_PARTFR 中的旋转均会被删除。之后坐标系的位置会根据 TCARR 重新恢复。PAROTOF 和 PAROT 位于同一 G 功能组中，因此在 G 指令显示中出现。

TOROT（通过框架旋转平行于刀具定向对准 WCS 的 Z 轴）和 TOFRAME（dto.）

TOROT 和 TOFRAME 的系统框架通过以下机床数据激活：
MD28082 $MC_MM_SYSTEM_FRAME_MASK，位 3 = 1

此系统框架位于可编程框架前的框架链中。
SZS 坐标系相应地位于可编程框架前。

TOROTOF

TOROTOF 是 TOROT 和 TOFRAME 的取消指令。此指令会删除对应的系统框架。此时当前 $P_TOOLFRAME 和数据管理框架 $P_TOOLFR 均会被删除。TOROTOF 和 TOROT 及 TOFRAME 位于同一 G 功能组中，因此在 G 指令显示中出现。

示例

使用取消了运动系统的可定向刀架

```
N10 $TC_DP1[1,1]= 120
N20 $TC_DP3[1,1]= 13                    ; 刀具长度 13 mm
; 定义刀架 1 :
N30 $TC_CARR1[1] = 0                    ; 偏移矢量 1 的 X 分量偏移矢量
N40 $TC_CARR2[1] = 0                    ; 偏移矢量 1 的 Y 分量偏移矢量
N50 $TC_CARR3[1] = 0                    ; 偏移矢量 1 的 Z 分量偏移矢量
N60 $TC_CARR4[1] = 0                    ; 偏移矢量 2 的 X 分量偏移矢量
N70 $TC_CARR5[1] = 0                    ; 偏移矢量 2 的 Y 分量偏移矢量
N80 $TC_CARR6[1] = -15                  ; 偏移矢量 2 的 Z 分量偏移矢量
N90 $TC_CARR7[1] = 1                    ; 第 1 根轴的 X 轴
N100 $TC_CARR8[1] = 0                   ; 第 1 根轴的 Y 轴
N110 $TC_CARR9[1] = 0                   ; 第 1 根轴的 Z 轴
N120 $TC_CARR10[1] = 0                  ; 第 2 根轴的 X 轴
N130 $TC_CARR11[1] = 1                  ; 第 2 根轴的 Y 轴
N140 $TC_CARR12[1] = 0                  ; 第 2 根轴的 Z 轴
N150 $TC_CARR13[1] = 30                 ; 第 1 根轴的轴
N160 $TC_CARR14[1] = -30                ; 第 2 根轴的轴
N170 $TC_CARR15[1] = 0                  ; 偏移矢量 3 的 X 分量偏移矢量
N180 $TC_CARR16[1] = 0                  ; 偏移矢量 3 的 Y 分量偏移矢量
N190 $TC_CARR17[1] = 0                  ; 偏移矢量 3 的 Z 分量偏移矢量
N200 $TC_CARR18[1] = 0                  ; 偏移矢量 4 的 X 分量偏移矢量
N210 $TC_CARR19[1] = 0                  ; 偏移矢量 4 的 Y 分量偏移矢量
N220 $TC_CARR20[1] = 15                 ; 偏移矢量 4 的 Z 分量偏移矢量
N230 $TC_CARR21[1] = A                  ; 用于第 1 根轴的轴
N240 $TC_CARR22[1] = B                  ; 用于第 2 根轴的轴
N250 $TC_CARR23[1] = "M"                ; 刀架的类型
```

```
N260 X0 Y0 Z0 A0 B45 F2000
N270 TCARR=1 X0 Y10 Z0 T1 TCOABS       ; 选择定向刀架
N280 PAROT                             ; 工作台旋转
N290 TOROT                             ; 在刀具定向中旋转 z 轴
N290 X0 Y0 Z0
N300 G18 MOVT=AC(20)                   ; 在 G18 平面中加工
N310 G17 X10 Y0 Z0                     ; 在 G17 平面中加工
N320 MOVT=-10
N330 PAROTOF                           ; 取消工作台旋转
N340 TOROTOF                           ; 不再在刀具上对准 WCS
N400 M30
```

11.5.9 带 SAVE 属性的子程序

对于不同的框架，其子程序相关特性可通过 SAVE 属性设置。

可设定框架 G54 - G599

通过 MD10617 $MN_FRAME_SAVE_MASK.**位 0** 可对可设定框架的特性进行设置：

- 位 0 = 0
  若通过子程序使用系统变量 $P_IFRAME 只修改了生效的可设定框架的值，而 G 指令保留，那么修改在子程序结束后将保留。
- 位 0 = 1
  子程序结束时，调用子程序前生效的可设定框架、G 指令和数值重新激活。

基本框架 $P_CHBFR[ ] 和 $P_NCBFR[ ]

通过 MD10617 $MN_FRAME_SAVE_MASK.**位 1** 可设置基本框架的特性：

- 位 1 = 0
  若通过子程序修改生效的基本框架，那么修改在子程序结束后将保留。
- 位 1 = 1
  子程序结束时，调用子程序前生效的基本框架重新激活。

可编程的框架

子程序结束时。调用子程序前生效的可编程框架重新激活。

系统框架

若通过子程序修改系统框架，那么修改在子程序结束后将保留。

11.5.10 数据备份

数据块

在以下数据块中进行框架的数据备份：

- 通道专用框架 ⇒ 数据块 _N_CHAN<x>_UFR
- 全局框架 ⇒ 数据块 _N_NC_UFR

说明

- 强烈建议在备份与重置备份的系统框架期间不要修改以下机床数据。否则可能会导致备份的系统框架无法再次载入。
  MD28082 $MC_MM_SYSTEM_FRAME_MASK（在通道中计算的通道专用的系统框架设计）
- 数据备份总是按照当前生效的几何轴指定关系进行，而不是按照机床数据中设置的轴配置进行。

系统框架的数据备份

系统框架的数据备份仅针对创建在数据管理中的系统框架。这些系统框架在调试控制系统时通过以下机床数据选择：

MD28083 $MC_MM_SYSTEM_DATAFRAME_MASK

不备份未创建在数据管理中的框架。

NC 全局框架的数据备份

仅当在以下其中一个机床数据中设置了至少一个 NC 全局框架时，才能对 NC 全局框架进行数据备份。

- MD18601 $MN_MM_NUM_GLOBAL_USER_FRAMES
- MD18602 $MN_MM_NUM_GLOBAL_BASE_FRAMES
- MD18603 $MN_MM_NUM_GLOBAL_G_FRAMES

11.5.11 坐标系中的位置

坐标系中的当前设定值位置可通过以下系统变量读取。通过 PLC 可选择以 WCS、SZS、BZS 或 MCS 显示实际值。为此可使用软键“以 MCS/WCS 显示实际值”。机床制造商可通过 PLC 设置，在其机床中哪个坐标系等同于工件坐标系。HMI 会从 NC 请求对应的实际值。

$AA_IM[轴]

使用变量 $AA_IM[轴] 可为每根轴读取机床坐标系中的设定值。

$AA_IEN[轴]

使用变量 $AA_IEN[轴] 可为每根轴读取可设定的零点坐标系 SZS 中的设定值。

$AA_IBN[轴]

使用 $AA_IBN[轴] 可为每根轴读取基本零点坐标系 BZS 中的设定值。

$AA_IW[轴]

使用 $AA_IW[轴] 可为每根轴读取工件坐标系 WCS 中的设定值。

11.5.12 控制系统特性

11.5.12.1 上电

框架	上电后的状态
可编程框架 $P_PFRAME	删除
可设定框架 $P_IFRAME	保留，取决于： • MD24080 $MC_USER_FRAME_POWERON_MASK，位 0 • MD20152 $MC_GCODE_RESET_MODE[7]
磨削框架 $P_GFRAME	保留，取决于： • MD24080 $MC_USER_FRAME_POWERON_MASK，位 0 • MD20152 $MC_GCODE_RESET_MODE[63]

框架	上电后的状态
总基本框架 $P_ACTBFRAME	保留，取决于： • MD20110 $MC_RESET_MODE_MASK 位 0 和位 14 通过以下机床数据可以删除基本框架： • MD10615 $MN_NCBFRAME_POWERON_MASK • MD24004 $MC_CHBFRAME_POWERON_MASK
系统框架： $P_PARTFRAME, $P_SETFRAME, $P_ISO1FRAME, $P_ISO2FRAME, $P_ISO3FRAME, $P_TOOLFRAME, $P_WPFRAME, $P_TRAFRAME, $P_ISO4FRAME, $P_RELFRAME, $P_CACFRAME	保留 通过以下机床数据可以删除单个系统框架： • MD24008 $MC_CHSFRAME_POWERON_MASK 系统框架的删除优先在数据管理中进行。
外部零点偏移 $P_EXTFRAME	保留，但是必须重新激活。
DRF 偏移	删除

11.5.12.2 运行方式切换

系统框架

系统框架继续生效。

JOG 运行方式

在 JOG 运行方式下，旋转生效时只为几何轴将当前框架纳入计算。所有其他轴框架均不被考虑。

PLC 轴和指令轴

对于 PLC 轴和指令轴，其特性可通过以下机床数据设置：

MD32074 $MA_FRAME_OR_CORRPOS_NOTALLOWED（不允许框架或 HL 补偿）

11.5.12.3 通道复位/零件程序结束

基本框架的复位特性

基本框架的复位特性通过以下机床数据设置：

MD20110 $MC_RESET_MODE_MASK（定义通道复位/零件程序结束后的控制系统初始设置）

基本框架的复位特性

通道复位/零件程序结束后，系统框架仍保留在数据管理中。

可通过以下机床数据配置单个系统框架的激活：

MD24006 $MC_CHSFRAME_RESET_MASK，位 <n> = <值>（通道复位/零件程序结束后生效的系统框架）

位	值	含义
0	1	通道复位/零件程序结束后实际值设置和对刀的系统框架生效。
1	1	通道复位/零件程序结束后外部零点偏移的系统框架生效。
2	---	不分析。
3	---	不分析。
4	1	通道复位/零件程序结束后工件参考点的系统框架生效。
5	1	通道复位/零件程序结束后循环的系统框架生效。
6	---	预留，复位特性取决于 MD20110 $MC_RESET_MODE_MASK。
7	1	通道复位/零件程序结束后系统框架 $P_ISO1FR 生效。
8	1	通道复位/零件程序结束后系统框架 $P_ISO2FR 生效。
9	1	通道复位/零件程序结束后系统框架 $P_ISO3FR 生效。
10	1	通道复位/零件程序结束后系统框架 $P_ISO4FR 生效。
11	1	通道复位/零件程序结束后系统框架 $P_RELFR 生效。
位 <n> = 0 时，相应的系统框架不生效。		

TCARR、PAROT、TOROT 和 TOFRAME 的系统框架的复位特性

通道复位/零件程序结束后 TCARR、PAROT、TOROT 和 TOFRAME 的系统框架的复位特性取决于 G 指令复位设置。

可通过以下机床数据进行设置：

- MD20110 $MC_RESET_MODE_MASK，位 <n> = <值>

位	值	含义	
0	0	保留 TCARR 和 PAROT 的当前系统框架。	
0	1	**其他相关机床数据设置**	**作用**
		MD20152 $MC_GCODE_RESET_MODE[51] = **0** 和 MD20150 $MC_GCODE_RESET_VALUES[51] = **1**	PAROTOF
		MD20152 $MC_GCODE_RESET_MODE[51] = **0** 和 MD20150 $MC_GCODE_RESET_VALUES[51] = **2**	PAROT
		MD20152 $MC_GCODE_RESET_MODE[51] = **1**	保留 TCARR 和 PAROT
		MD20152 $MC_GCODE_RESET_MODE[52] = **0** 和 MD20150 $MC_GCODE_RESET_VALUES[52] = **1**	TOROTOF
		MD20152 $MC_GCODE_RESET_MODE[52] = **0** 和 MD20150 $MC_GCODE_RESET_VALUES[52] = **2**	TOROT
		MD20152 $MC_GCODE_RESET_MODE[52] = **0** 和 MD20150 $MC_GCODE_RESET_VALUES[52] = **3**	TOFRAME
		MD20152 $MC_GCODE_RESET_MODE[52] = **1**	保留 TOROT 和 TOFRAME
位 0 == 1 时的其他位的说明			

位	值	含义
14	0	级联总基本框架会被删除。
	1	总基本框架是由以下机床数据得出的：
		MD24002 $MC_CHBFRAME_RESET_MASK，位 <n> = 1 n：第 n 个通道专用基本框架被纳入级联的总基本框架内。
		MD10613 $MN_NCBFRAME_RESET_MASK，位 <n> = 1 n：第 n 个 NCU 全局基本框架被纳入级联的总基本框架内。

TCARR 为 PAROT 两个独立的功能，其定义的框架相同。使用 PAROTOF 时，TCARR 的分量同样不会在通道复位/零件程序结束时激活。

- MD20152 $MC_GCODE_RESET_MODE[ ]（G 功能组的复位特性）
- MD20150 $MC_GCODE_RESET_VALUES[ ]（G 功能组的初始设置）

通道复位/零件程序结束后的框架状态

框架	通道复位/零件程序结束后的状态
可编程框架：$P_PFRAME	删除
可设定框架 $P_IFRAME	保留，取决于： • MD20110 $MC_RESET_MODE_MASK • MD20152 $MC_GCODE_RESET_MODE[7]
磨削框架 $P_GFRAME	保留，取决于： • MD20110 $MC_RESET_MODE_MASK • MD20152 $MC_GCODE_RESET_MODE[63]
总基本框架 $P_ACTBFRAME	保留，取决于： • MD20110 $MC_RESET_MODE_MASK 位 0 和位 14 • MD10613 $MN_NCBFRAME_RESET_MASK • MD24002 $MC_CHBFRAME_RESET_MASK

框架	通道复位/零件程序结束后的状态
系统框架： $P_PARTFRAME, $P_SETFRAME, $P_ISO1FRAME, $P_ISO2FRAME, $P_ISO3FRAME, $P_TOOLFRAME, $P_WPFRAME, $P_TRAFRAME, $P_ISO4FRAME, $P_RELFRAME, $P_CACFRAME	保留，取决于： • MD24006 $MC_CHSFRAME_RESET_MASK • MD20150 $MC_GCODE_RESET_VALUES[<n>]
外部零点偏移 $P_EXTFRAME	保留
DRF 偏移	保留

删除系统框架

在通道复位/零件程序结束时可通过以下机床数据删除数据管理中的系统框架：

MD24007 $MC_CHSFRAME_RESET_CLEAR_MASK，位 <n> = <值>

位	值	含义
0	1	通道复位/零件程序结束时删除实际值设置和对刀的系统框架。
1	1	通道复位/零件程序结束时删除外部零点偏移的系统框架。
2	---	预留，TCARR 和 PAROT 参见 MD20150 $MC_GCODE_RESET_VALUES[ ]。
3	---	预留，TOROT 和 TOFRAME 参见 MD20150 $MC_GCODE_RESET_VALUES[ ]。
4	1	通道复位/零件程序结束时删除工件参考点的系统框架。
5	1	通道复位/零件程序结束时删除循环的系统框架。
6	---	预留，复位特性取决于 MD20110 $MC_RESET_MODE_MASK。
7	1	通道复位/零件程序结束时删除系统框架 $P_ISO1FR。
8	1	通道复位/零件程序结束时删除系统框架 $P_ISO2FR。
9	1	通道复位/零件程序结束时删除系统框架 $P_ISO3FR。
10	1	通道复位/零件程序结束时删除系统框架 $P_ISO4FR。

位	值	含义
11	1	通道复位/零件程序结束时删除系统框架 $P_RELFR。
位 <n> = 0 时，不删除相应的系统框架。		

11.5.12.4 零件程序开始

零件程序开始后的框架状态

框架	零件程序开始后的状态
可编程框架 $P_PFRAME	删除
可设定框架 $P_IFRAME	保留，取决于： MD20112 $MC_START_MODE_MASK
磨削框架 $P_GFRAME	保留，取决于： MD20112 $MC_START_MODE_MASK
总基本框架 $P_ACTBFRAME	保留
系统框架： $P_PARTFRAME, $P_SETFRAME, $P_ISO1FRAME, $P_ISO2FRAME, $P_ISO3FRAME, $P_TOOLFRAME, $P_WPFRAME, $P_TRAFRAME, $P_ISO4FRAME, $P_RELFRAME, $P_CACFRAME	保留
外部零点偏移 $P_EXTFRAME	保留
DRF 偏移	保留

11.5.12.5 程序段搜索

带计算的程序段搜索

在“进行计算的程序段搜索”中，数据管理框架也会被修改。

程序段搜索终止

若程序段搜索由于通道复位终止，可通过以下机床数据将所有数据管理框架复位为程序段搜索前的值：

MD28560 $MC_MM_SEARCH_RUN_RESTORE_MODE，位 <n>

位	值	含义
0	1	恢复数据管理中的所有框架。

在级联程序段搜索中，框架会被设置为前一程序段搜索中的状态。

SERUPRO

不支持“SERUPRO”功能。

11.5.12.6 REPOS

对框架无特殊处理。 若在 ASUB 中修改了框架，则其保留在程序中。 通过 REPOS 重新定位时，只要在 ASUB 中激活了修改，经过修改的框架便会被启用。

11.6 工件相关的实际值系统

11.6.1 概述

定义

“工件相关实际值系统”涵盖了一系列功能，用于协助用户实现以下操作：

- 启动后，启用通过机床数据定义的工件坐标系。
  特性：
  - 无需额外操作
  - 在 JOG 和 AUTO 运行方式下生效
- 零件程序结束时生效的设置在下一个零件程序中保留，所涉及的设置包括：
  - 生效的平面
  - 可设定框架（G54-G57）
  - 运动转换
  - 生效的刀具补偿
- 通过操作界面在工件坐标系和机床坐标系间进行切换。
- 通过操作修改工件坐标系（例如修改可设定框架或刀具补偿）。

11.6.2 使用工件相关实际值系统

前提条件

已针对该系统执行上一章节中描述的设置。
HMI 软件启动后的初始设置为 MCS。

切换至 WCS

通过操作界面切换至 WCS 时，会触发以 WCS 原点为基准的轴位置显示。

切换至 MCS

通过操作界面切换至 MCS 时，会触发以 MCS 原点为基准的轴位置显示。

坐标系之间的关联

下图显示了从机床坐标系 MCS 到工作坐标系 WCS 之间的关联。

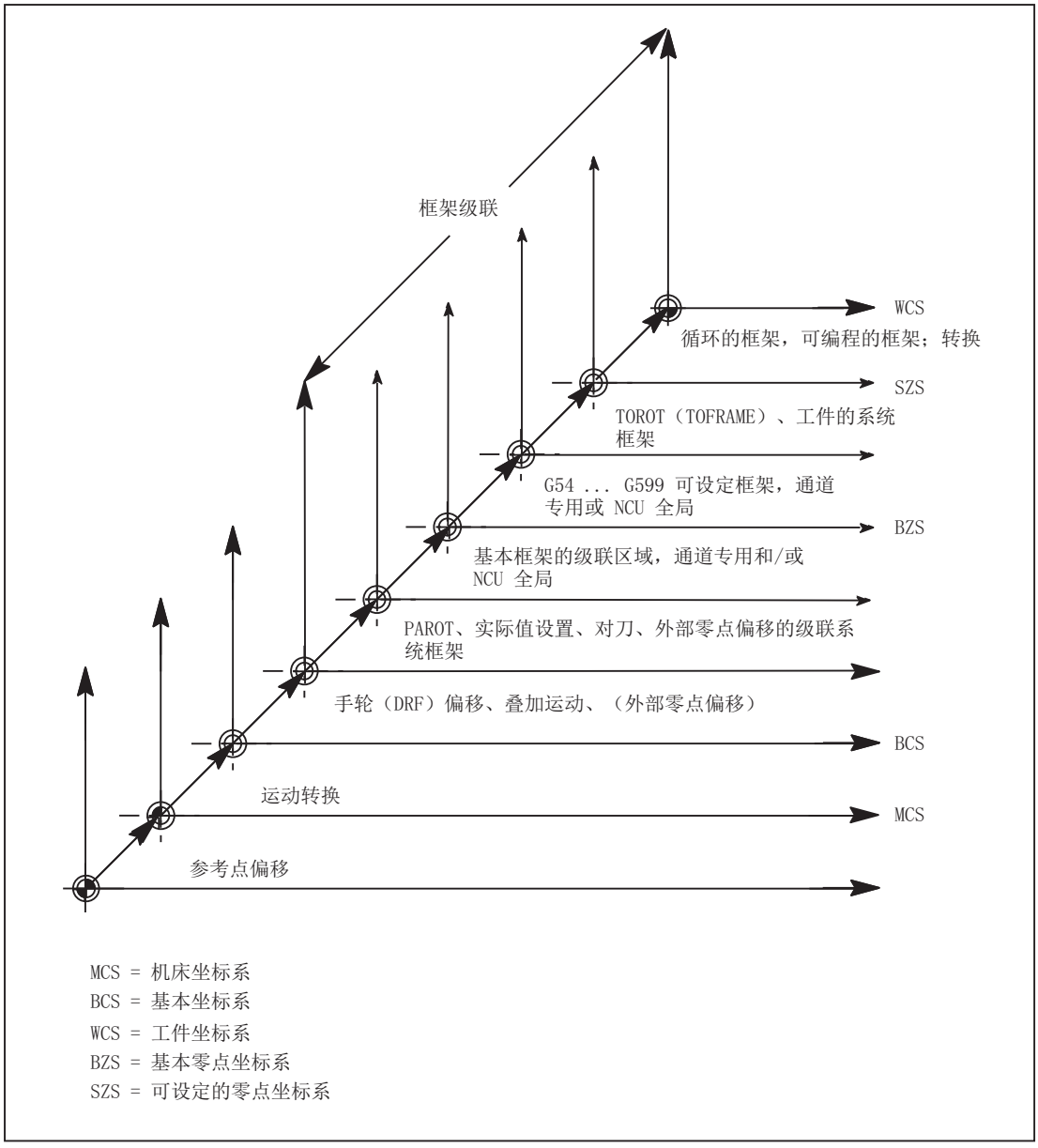


图 11-28　坐标系关联

更多相关信息请见“H2：输出到 PLC 的辅助功能 (页 429)”和“W1：刀具补偿 (页 1593)”章节。

文档：

- 编程手册之基本原理分册
- 功能手册之扩展功能分册；运动转换（M1）
- 功能手册之特殊功能分册；轴耦合和 ESR（M3）；“跟随运行”和“引导值耦合”章节
- 功能手册之特殊功能分册；切向控制（T3）

11.6.3 特殊响应

溢出转存

复位状态下对以下数据的溢出转存：

- 框架（零点偏移）
- 生效的平面
- 激活的转换
- 刀具补偿

立即作用于通道中所有轴的实际值显示。

通过操作面板前端输入

通过操作面板前端上的操作修改以下数值时：
“生效的框架”（零点偏移，“参数”操作区域）
和
“生效的刀具长度补偿”（“参数”操作区域）
则会通过下列措施之一在显示中激活修改：

- 按下 RESET 键
- 重新选择：
  - 零点偏移，通过零件程序
  - 刀具补偿，通过零件程序
- 重新设置：
  - 零点偏移，通过溢出转存
  - 刀具补偿，通过溢出转存
- 零件程序开始

MD9440

若设置了用于操作面板前端的 HMI 机床数据：
MD9440 ACTIVATE_SEL_USER_DATA
那么输入的值会在复位状态下立即生效。

在零件程序执行暂停的情形下进行输入时，数值会在继续运行开始时生效。

实际值读取

在激活框架（零点偏移）或刀具补偿后，若从 $AA_IW 读取 WCS 中的实际值，那么读取结果中将包含激活的修改，即便尚未通过激活的修改运行轴。

使用变量 $AA_IEN[轴] 可为每根轴从零件程序读取可设定零点坐标系 SZS 中的实际值。

使用 $AA_IBN[轴] 可为每根轴从零件程序读取基本零点坐标系 BZS 中的实际值。

实际值显示

WCS 中总是显示编写的轮廓。

以下偏移会被叠加至 MCS：

- 运动转换
- DRF 偏移/外部零点偏移
- 生效的框架
- 当前刀具的生效刀具补偿

通过 PLC 切换

通过 PLC 可选择以 WCS、SZS、BZS 或 MCS 显示实际值。 可通过 PLC 设定，机床上的哪个坐标系等同于工件坐标系。

启动后的缺省坐标系为 MCS。
也可使用 PLC 通过信号 DB19 DBB0.7“MCS/WCS 切换”切换至 WCS。

传输至 PLC

通过设置以下机床数据：
MD20110 / MD20112，位 1
可设定在选择刀具长度补偿时是否将辅助功能（D、T、M）输出至 PLC。

说明

若通过 PLC 选择了 WCS，则可通过操作针对各运行方式在 WCS 和 WCS 间进行切换。不过在运行方式或/和运行区域切换时，系统还是会分析和激活通过 PLC 选择的 WCS（参见“K1：BAG、通道、程序运行、复位特性 (页 519)”章节）。

11.7 前提条件

没有需要遵循的前提条件。

11.8 示例

11.8.1 轴

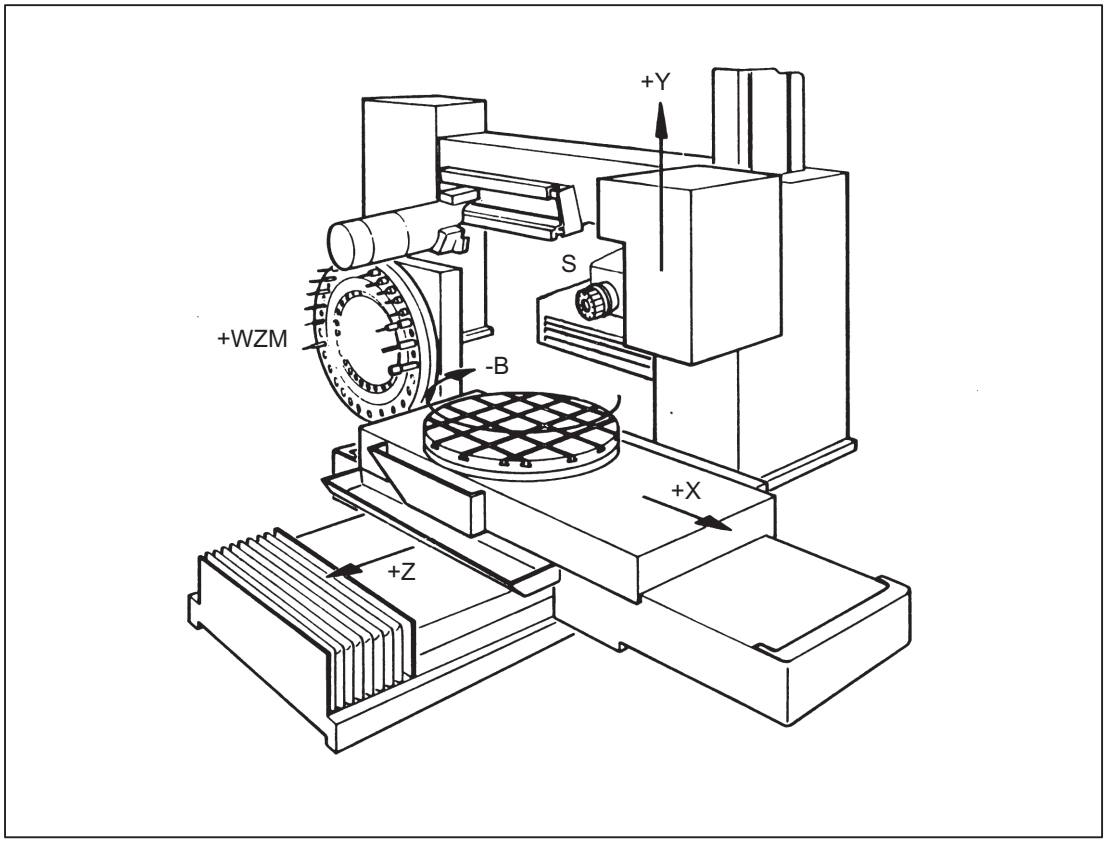
带回转台的 3 轴铣床的轴配置

机床轴 1：X1	线性轴
机床轴 2：Y1	线性轴
机床轴 3：Z1	线性轴
机床轴 4： B1	回转台（用于多面加工中的车削）
机床轴 5：W1	刀库的回转轴（刀具托盘）
机床轴 6： C1	（主轴）

几何轴 1：X	（通道 1）
几何轴 2：Y	（通道 1）
几何轴 3：Z	（通道 1）
辅助轴 1：B	（通道 1）

辅助轴 2： WZM （通道 1）

主轴 1： S1/C （通道 1）



机床数据的参数设置

机床数据	值
MD10000 AXCONF_MACHAX_NAME_TAB[0]	= X1
MD10000 AXCONF_MACHAX_NAME_TAB[1]	= Y1
MD10000 AXCONF_MACHAX_NAME_TAB[2]	= Z1
MD10000 AXCONF_MACHAX_NAME_TAB[3]	= B1
MD10000 AXCONF_MACHAX_NAME_TAB[4]	= W1

机床数据	值
MD10000 AXCONF_MACHAX_NAME_TAB[5]	= C1
MD20050 AXCONF_GEOAX_ASSIGN_TAB[0]	= 1
MD20050 AXCONF_GEOAX_ASSIGN_TAB[1]	= 2
MD20050 AXCONF_GEOAX_ASSIGN_TAB[2]	= 3
MD20060 AXCONF_GEOAX_NAME_TAB[0]	= X
MD20060 AXCONF_GEOAX_NAME_TAB[1]	= Y
MD20060 AXCONF_GEOAX_NAME_TAB[2]	= Z
MD20070 AXCONF_MACHAX_USED[0]	= 1
MD20070 AXCONF_MACHAX_USED[1]	= 2
MD20070 AXCONF_MACHAX_USED[2]	= 3
MD20070 AXCONF_MACHAX_USED[3]	= 4
MD20070 AXCONF_MACHAX_USED[4]	= 5
MD20070 AXCONF_MACHAX_USED[5]	= 6
MD20080 AXCONF_CHANAX_NAME_TAB[0]	= X
MD20080 AXCONF_CHANAX_NAME_TAB[1]	= Y
MD20080 AXCONF_CHANAX_NAME_TAB[2]	= Z
MD20080 AXCONF_CHANAX_NAME_TAB[3]	= B

机床数据	值
MD20080 AXCONF_CHANAX_NAME_TAB[4]	= WZM
MD20080 AXCONF_CHANAX_NAME_TAB[5]	= S1
MD30300 IS_ROT_AX[3]	= 1
MD30300 IS_ROT_AX[4]	= 1
MD30300 IS_ROT_AX[5]	= 1
MD30310 ROT_IS_MODULO[3]	= 1
MD30310 ROT_IS_MODULO[4]	= 1
MD30310 ROT_IS_MODULO[5]	= 1
MD30320 DISPLAY_IS_MODULO[3]	= 1
MD30320 DISPLAY_IS_MODULO[4]	= 1
MD20090 SPIND_DEF_MASTER_SPIND	= 1
MD35000 SPIND_ASSIGN_TO_MACHAX[AX1]	= 0
MD35000 SPIND_ASSIGN_TO_MACHAX[AX2]	= 0
MD35000 SPIND_ASSIGN_TO_MACHAX[AX3]	= 0
MD35000 SPIND_ASSIGN_TO_MACHAX[AX4]	= 0
MD35000 SPIND_ASSIGN_TO_MACHAX[AX5]	= 0
MD35000 SPIND_ASSIGN_TO_MACHAX[AX6]	= 1

11.8.2 坐标系

配置全局基本框架

需要一个有 2 个通道的 NC。其中：

- 两个通道均可写入全局基本框架。
- 重新激活全局基本框架后，另一通道能够识别出变化。
- 两个通道均可读取全局基本框架。
- 两个通道均可为自身激活全局基本框架。

机床数据

机床数据	值
MD10000 $MN_AXCONF_MACHAX_NAME_TAB[0] MD10000 $MN_AXCONF_MACHAX_NAME_TAB[1] MD10000 $MN_AXCONF_MACHAX_NAME_TAB[2] MD10000 $MN_AXCONF_MACHAX_NAME_TAB[3] MD10000 $MN_AXCONF_MACHAX_NAME_TAB[4] MD10000 $MN_AXCONF_MACHAX_NAME_TAB[5]	= X1 = X2 = X3 = X4 = X5 = X6
MD18602 $MN_MM_NUM_GLOBAL_BASE_FRAMES	= 1
MD28081 $MC_MM_NUM_BASE_FRAMES	= 1

通道 1 的机床数据	值		通道 1 的机床数据	值
$MC_AXCONF_CHANAX_NAME_TAB[0] $MC_AXCONF_CHANAX_NAME_TAB[1] $MC_AXCONF_CHANAX_NAME_TAB[2]	= X = Y = Z		$MC_AXCONF_CHANAX_NAME_TAB[0] $MC_AXCONF_CHANAX_NAME_TAB[1] $MC_AXCONF_CHANAX_NAME_TAB[2]	= X = Y = Z
$MC_AXCONF_MACHAX_USED[0] $MC_AXCONF_MACHAX_USED[1] $MC_AXCONF_MACHAX_USED[2]	= 1 = 2 = 3		$MC_AXCONF_MACHAX_USED[0] $MC_AXCONF_MACHAX_USED[1] $MC_AXCONF_MACHAX_USED[2]	= 4 = 5 = 6
$MC_AXCONF_GEOAX_NAME_TAB[0] $MC_AXCONF_GEOAX_NAME_TAB[1] $MC_AXCONF_GEOAX_NAME_TAB[2]	= X = Y = Z		$MC_AXCONF_GEOAX_NAME_TAB[0] $MC_AXCONF_GEOAX_NAME_TAB[1] $MC_AXCONF_GEOAX_NAME_TAB[2]	= X = Y = Z
$MC_AXCONF_GEOAX_ASSIGN_TAB[0] $MC_AXCONF_GEOAX_ASSIGN_TAB[1] $MC_AXCONF_GEOAX_ASSIGN_TAB[2]	= 1 = 2 = 3		$MC_AXCONF_GEOAX_ASSIGN_TAB[0] $MC_AXCONF_GEOAX_ASSIGN_TAB[1] $MC_AXCONF_GEOAX_ASSIGN_TAB[2]	= 4 = 5 = 6

第 1 通道中的通道

代码（节选）：	; 注释
. . .	
N100 $P_NCBFR[0] = CTRANS(x, 10)	; 激活 NC 全局基本框架
. . .	
N130 $P_NCBFRAME[0] = CROT(X, 45)	; 激活包含旋转的 NC 全局基本框架 => ; 报警 18310，因为 NC 全局框架中 ; 不允许有旋转
. . .	

第 2 通道中的通道

代码（节选）：	; 注释
. . .	
N100 $P_NCBFR[0] = CTRANS(x, 10)	; NC 全局基本框架也在通道 2 中生效
. . .	
N510 G500 X10	; 激活基本框架
N520 $P_CHBFRAME[0] = CTRANS(x, 10)	通道 2 的当前框架通过偏移激活
. . .	

11.8.3 框架

示例 1

需通过几何轴切换将通道轴设定为几何轴。

切换后，可编程框架包含 x 轴方向上 10 的偏移分量。

保留当前可设定框架。

FRAME_GEOX_CHANGE_MODE = 1

```
$P_UIFR[1] = CROT(x,10,y,20,z,30)  ; 几何轴切换后保留框架
G54                                ; 可设定框架生效
TRANS a10                          ; 切换时附带轴 a 的轴偏移
GEOAX(1, a)                        ; 将轴 a 设定为 x 轴
                                   ; $P_ACTFRAME= CROT(x,10,y,20,z,30):CTRANS(x10)
```

坐标转换切换时，可同时将多个通道轴设定为几何轴。

示例 2

通过 5 轴定向转换，通道轴 4、5、6 成为转换的几何轴。 即在转换前替换所有几何轴。

激活转换时，所有当前框架相应变化。

为了计算新的 WCS 坐标系，将成为几何轴的通道轴的轴框架分量纳入计算。 转换前编写的旋转保留。 取消转换后，重新恢复为原先的 WCS。

最常见应用就是使几何轴在坐标转换前后不发生变化，并保留转换前生效的框架。

机床数据：

$MN_FRAME_GEOAX_CHANGE_MODE = 1

$MC_AXCONF_CHANAX_NAME_TAB[0] = "CAX"
$MC_AXCONF_CHANAX_NAME_TAB[1] = "CAY"
$MC_AXCONF_CHANAX_NAME_TAB[2] = "CAZ"
$MC_AXCONF_CHANAX_NAME_TAB[3] = "A"
$MC_AXCONF_CHANAX_NAME_TAB[4] = "B"
$MC_AXCONF_CHANAX_NAME_TAB[5] = "C"

$MC_AXCONF_GEOAX_ASSIGN_TAB[0] = 1
$MC_AXCONF_GEOAX_ASSIGN_TAB[1] = 2
$MC_AXCONF_GEOAX_ASSIGN_TAB[2] = 3

$MC_AXCONF_GEOAX_NAME_TAB[0] = "X"
$MC_AXCONF_GEOAX_NAME_TAB[1] = "Y"
$MC_AXCONF_GEOAX_NAME_TAB[2] = "Z"

$MC_TRAFO_GEOAX_ASSIGN_TAB_1[0] = 4
$MC_TRAFO_GEOAX_ASSIGN_TAB_1[1] = 5
$MC_TRAFO_GEOAX_ASSIGN_TAB_1[2] = 6

$MC_TRAFO_AXES_IN_1[0] = 4
$MC_TRAFO_AXES_IN_1[1] = 5
$MC_TRAFO_AXES_IN_1[2] = 6
$MC_TRAFO_AXES_IN_1[3] = 1
$MC_TRAFO_AXES_IN_1[4] = 2

程序：

```
$P_NCBFRAME[0] = ctrans(x,1,y,2,z,3,a,4,b,5,c,6)
$P_CHBFRAME[0] = ctrans(x,1,y,2,z,3,a,4,b,5,c,6)
$P_IFRAME = ctrans(x,1,y,2,z,3,a,4,b,5,c,6):crot(z,45)
$P_PFRAME = ctrans(x,1,y,2,z,3,a,4,b,5,c,6):crot(x,10,y,20,z,30)
```

```
TRAORI                          ; 坐标转换，设置 GeoAx(4,5,6)
                                ; $P_NCBFRAME[0] = ctrans(x,4,y,5,z,6,cax,1,cay,2,caz,
                                  3)
                                ; $P_ACTBFRAME = ctrans(x,8,y,10,z,12,cax,2,cay,4,caz,
                                  6)
                                ; $P_PFRAME = ctrans(x,4,y,5,z,6,cax,1,cay,2,caz,3):
                                ; crot(x,10,y,20,z,30)
                                ; $P_IFRAME = ctrans(x,4,y,5,z,6,cax,1,cay,2,caz,
                                  3):crot(z,45)

TRAFOOF;                        ; 取消转换，设置 GeoAx(1,2,3)
                                ; $P_NCBFRAME[0] = ctrans(x,1,y,2,z,3,a,4,b,5,c,6)
                                ; $P_CHBFRAME[0] = ctrans(x,1,y,2,z,3,a,4,b,5,c,6)
                                ; $P_IFRAME = ctrans(x,1,y,2,z,3,a,4,b,5,c,6):crot(z,
                                  45)
                                ; $P_PFRAME = ctrans(x,1,y,2,z,3,a,4,b,5,c,6):crot(x,
                                  10,y,20,z,30)
```

11.9 数据表

11.9.1 机床数据

11.9.1.1 显示机床数据

编号	名称：$MM_	说明
SINUMERIK Operate		
9242	MA_STAT_DISPLAY_BASE	用于显示关节位置 STAT 的基数
9243	MA_TU_DISPLAY_BASE	用于显示回转轴位置 TU 的基数
9244	MA_ORIAXES_EULER_ANGLE_NAME	以欧拉角显示定向轴
9245	MA_PRESET_FRAMEIDX	对刀和实际值设置功能的值存储
9247	USER_CLASS_BASE_ZERO_OFF_PA	“参数”操作区域中基本偏移的可用性
9248	USER_CLASS_BASE_ZERO_OFF_MA	“加工”操作区域中基本偏移的可用性
9424	MA_COORDINATE_SYSTEM	用于实际值显示的坐标系

编号	名称：$MM_	说明
9440	ACTIVE_SEL_USER_DATA	生效的数据（框架）在修改后立即生效
9449	WRITE_TOA_LIMIT_MASK	MD9203 在刀沿数据和位置相关补偿方面的有效性
9450	MM_WRITE_TOA_FINE_LIMIT	精损的极限值
9451	MM_WRITE_ZOA_FINE_LIMIT	精偏的极限值

11.9.1.2 NC 专用机床数据

编号	名称：$MN_	说明
10000	AXCONF_MACHAX_NAME_TAB	机床轴名称
10600	FRAME_ANGLE_INPUT_MODE	框架中的旋转顺序
10602	FRAME_GEOAX_CHANGE_MODE	几何轴切换时的框架
10610	MIRROR_REF_AX	用于镜像的参考轴
10612	MIRROR_TOGGLE	切换镜像
10613	NCBFRAME_RESET_MASK	复位后生效的 NCU 全局基本框架
10615	NCBFRAME_POWERON_MASK	上电后复位全局基本框架
10617	FRAME_SAVE_MASK	SAVE 子程序中框架的特性
10650	IPO_PARAM_NAME_TAB	插补参数的名称
10660	INTERMEDIATE_POINT_NAME_TAB	G2/G3 下中间点坐标的名称
11640	ENABLE_CHAN_AX_GAP	允许通道轴空隙
18600	MM_FRAME_FINE_TRANS	FRAME 时的精偏（SRAM）
18601	MM_NUM_GLOBAL_USER_FRAMES	全局预定义用户框架的数量（SRAM）
18602	MM_NUM_GLOBAL_BASE_FRAMES	全局基本框架的数量（SRAM）

11.9.1.3 通道专用机床数据

编号	名称：$MC_	说明
20050	AXCONF_GEOAX_ASSIGN_TAB	几何轴指定为通道轴
20060	AXCONF_GEOAX_NAME_TAB	通道中的几何轴名称
20070	AXCONF_MACHAX_USED	通道内有效的机床轴编号

编号	名称：$MC_	说明
20080	AXCONF_CHANAX_NAME_TAB	通道内的通道轴名称
20110	RESET_MODE_MASK	确定复位/零件程序结束后控制系统的初始设置
20118	GEOAX_CHANGE_RESET	允许自动几何轴切换
20126	TOOL_CARRIER_RESET_VALUE	复位时生效的刀架
20140	TRAFO_RESET_VALUE	启动（复位/零件程序结束）中的转换数据组
20150	GCODE_RESET_VALUES	G 功能组的初始设置
20152	GCODE_RESET_MODE	G 功能组的复位特性
20184	TOCARR_BASE_FRAME_NUMBER	用于接收工作台偏移的基本框架的编号
21015	INVOLUTE_RADIUS_DELTA	渐开线功能中的终点监控
22532	GEOAX_CHANGE_M_CODE	几何轴切换时的 M 代码
22534	TRAFO_CHANGE_M_CODE	转换切换时的 M 代码
24000	FRAME_ADD_COMPONENTS	G58 和 G59 的框架分量
24002	CHBFRAME_RESET_MASK	通道专用基本框架的复位特性
24004	CHBFRAME_POWERON_MASK	上电后复位通道专用基本框架
24006	CHSFRAME_RESET_MASK	复位后生效的系统框架
24007	CHSFRAME_RESET_CLEAR_MASK	复位时删除系统框架
24008	CHSFRAME_POWERON_MASK	上电后复位系统框架
24010	PFRAME_RESET_MODE	可编程框架的复位模式
24020	FRAME_SUPPRESS_MODE	框架抑制时的位置
24030	FRAME_ACT_SET	SZS 坐标的设置
24040	FRAME_ADAPT_MODE	调整生效框架
24050	FRAME_SAA_MODE	保存和激活数据管理框架
24805	TRACYL_ROT_AX_FRAME_1	回转轴偏移 `TRACYL 1`
24855	TRACYL_ROT_AX_FRAME_2	回转轴偏移 `TRACYL 2`
24905	TRANSMIT_ROT_AX_FRAME_1	回转轴偏移 `TRANSMIT 1`
24955	TRANSMIT_ROT_AX_FRAME_2	回转轴偏移 `TRANSMIT 2`
28080	MM_NUM_USER_FRAMES	可设置框架的数量（SRAM）
28081	MM_NUM_BASE_FRAMES	基本框架的数量
28082	MM_SYSTEM_FRAME_FRAMES	系统框架（SRAM）
28560	MM_SEARCH_RUN_RESTORE_MODE	仿真后恢复数据

11.9.1.4 进给轴/主轴专用机床数据

编号	名称：$MA_	说明
32074	FRAME_OR_CORRPOS_NOTALLOWED	不允许使用框架或 HL 补偿
35000	SPIND_ASSIGN_TO_MACHAX	主轴指定为机床轴

11.9.2 设定数据

11.9.2.1 通道专用设定数据

编号	名称：$SC_	说明
42440	FRAME_OFFSET_INCR_PROG	框架中的零点偏移
42980	TOFRAME_MODE	TOFRAME、TOROT 和 PAROT 中的框架定义

11.9.3 系统变量

名称	说明
$AA_ETRANS[轴]	外部零点偏移
$AA_IBN[轴]	基本零点坐标系（BZS）中的实际值
$AA_IEN[轴]	可设定零点坐标系（EZS）中的当前实际值
$AA_OFF[轴]	编程轴的叠加运动
$AC_DRF[轴]	轴的手轮叠加
$AC_JOG_COORD	用于手动运行的坐标系
$P_ACSFRAME	BCS 和 SZS 之间生效的框架
$P_ACTBFRAME	生效的总基本框架
$P_ACTFRAME	生效的总框架
$P_BFRAME	第 1 个生效的基本框架对应 $P_CHBFRAME[0]
$P_CHBFR[<n>]	数据管理框架：基本框架可通过 G500、G54...G599 激活
$P_CHBFRAME[<n>]	生效的基本框架

名称	说明
$P_CHBFRMASK	通道中的基本框架掩码
$P_CHSFRMASK	系统框架掩码
$P_CYCFR	数据管理框架：循环系统框架
$P_CYCFRAME	生效的用于循环的系统框架
$P_EXTFR	数据管理框架：外部零点偏移的系统框架
$P_EXTFRAME	生效的用于外部零点偏移的系统框架
$P_IFRAME	生效的可设定框架
$P_ISO1FR	数据管理框架：ISO G51.1 镜像系统框架
$P_ISO2FR	数据管理框架：ISO G68 2DROT 系统框架
$P_ISO3FR	数据管理框架：ISO G68 3DROT 系统框架
$P_ISO4FR	数据管理框架：ISO G51 比例缩放系统框架
$P_ISO1FRAME	生效的 ISO G51.1 镜像系统框架
$P_ISO2FRAME	生效的 ISO G68 2DROT 系统框架
$P_ISO3FRAME	生效的 ISO G68 3DROT 系统框架
$P_ISO4FRAME	生效的 ISO G51 比例缩放系统框架
$P_NCBFR[n]	数据管理框架：NCU 全局数据管理基本框架，可通过 G500、G54...G599 激活
$P_NCBFRAME[n]	生效的 NCU 全局基本框架
$P_NCBFRMASK	全局基本框架掩码
$P_PARTFR	数据管理框架：TCARR 和 PAROT 的系统框架
$P_PARTFRAME	使用可定向刀架时 TCARR 和 PAROT 的生效系统框架
$P_PFRAME	生效的可编程框架
$P_SETFR	数据管理框架：实际值设置的系统框架
$P_SETFRAME	生效的用于实际值设置的系统框架
$P_TOOLFR	数据管理框架：TOROT 和 TOFRAME 的系统框架
$P_TOOLFRAME	TOROT 和 TOFRAME 的生效系统框架
$P_TRAFR	数据管理框架：转换系统框架
$P_TRAFRAME	生效的用于转换的系统框架
$P_UBFR	数据管理中通道中的第 1 个基本框架，在 G500、G54...G599 后激活。对应 $P_CHBFR[0]。
$P_UIFR[n]	数据管理框架：可设定框架，可通过 G500、G54...G599 激活

名称	说明
$P_UIFRNUM	生效的可设定框架的编号
$P_WPFR	数据管理框架：工件系统框架
$P_WPFRAME	生效的用于工件的系统框架

11.9.4 信号

11.9.4.1 来自通道的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
T 功能修改	DB21, ...DBX61.0-.2	-
D 功能修改	DB21, ...DBX62.0-.2	-
T 功能 1	DB21, ...DBB118-119	DB250x.DBD2000
D 功能 2	DB21, ...DBB129	DB250x.DBD5000
G 功能组 1 生效功能的编号，8（Bit -Int）	DB21, ...DBB208	DB350x.DBB0
G 功能组 2 生效功能的编号，8（Bit -Int）	DB21, ...DBB209	DB350x.DBB1
...	...	…
G 功能组 29 生效功能的编号，8（Bit -Int）	DB21, ...DBB236	DB350x.DBB28

11.9.4.2 发送至进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
接收外部零点偏移	DB31, ...DBX3.0	-

11.9.4.3 来自进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
主轴/回转轴	DB31, ...DBX60.0	DB390x.DBX0.0

N2：急停

12

12.1 简要说明

功能

控制系统通过以下功能协助机床制造商实现急停：

- 所有 SINUMERIK 机床控制面板上均配备了一个急停键，操作人员可方便地触及。 急停键的功能包括强制打开电子开关触点，以及机械自动闭锁。
- 急停请求由 PLC 通过 NC/PLC 接口发送至 NC。
- 急停功能必须实现 0 类停机或 1 类停机（EN 60204）。
- 所有通过 PLC 控制的机床功能可在急停时进入安全状态，此状态可由机床制造商设置。
- 急停键的解锁不会使急停状态取消，也不会触发重启。
- 在取消急停状态后，不需要对机床轴执行回参考点或执行主轴同步。 系统会在急停期间持续跟踪机床轴的实际位置。

12.2 标准

相关标准

对于急停功能，须遵循下列标准：

- EN ISO 12000-1
- EN ISO 12000-2
- EN 418
- EN 60204

急停

根据 EN 418，急停功能的特点是：

- 用于防止人员遭受潜在或现存的危险，避免机械或生产材料受损。
- 当普通的停止功能不适用时，可由人员通过单次操作触发。

危险

根据 EN 418，危险可能来源于：

- 功能不规则（机械失灵、待加工材料的属性不当、人员操作失误等）。
- 一般运行。

EN ISO 12000-2 标准

根据欧盟机械指令中对急停的基本安全要求，机械必须配备急停装置。

特例

以下机械不需要急停装置：

- 急停装置无法降低风险，因为无法减少停止时间，或因所需采取的风险控制措施不适宜。
- 手持和手动控制的机械。

注意
机床制造商会接收到书面提示，敦促其遵循国家和国际标准。 SINUMERIK 控制系统能够协助机床制造商按照下文的功能说明实现急停功能。 但机床制造商须自行对急停功能（触发、运行以及应答）负责。

12.3 急停控制元件

急停控制元件

根据 EN 418，急停控制元件必须能够实现机械自动闭锁，且在紧急状况下能由操作人员和其他人员方便地操作。

例如可使用以下类型的控制元件：

- 蘑菇头开关（按键开关）
- 导线/线缆、索线、棒杆
- 手柄
- 特殊情形下：无保护帽的足控开关

急停键和控制系统

急停键执行或直接由此产生的信号必须作为 PLC 输入信号传送到控制系统（PLC）。 在 PLC 用户程序中，此 PLC 输入信号转送至 NC，到达接口信号：

DB10 DBX56.1（急停）

急停键复位或直接由此产生的信号必须作为 PLC 输入信号传送到控制系统（PLC）。 在 PLC 用户程序中，此 PLC 输入信号转送至 NC，到达接口信号：

DB10 DBX56.2（应答急停）

连接条件

急停键的连接请参见：
文档：
设备手册之操作组件分册

12.4 急停过程

在执行急停控制元件后，急停装置以特定方式运作，从而以最优方式自动规避或削减风险。

“最优方式”表示：依据风险评估，选择最适宜的减速速率和定义正确的停止类别（EN 60204 中的定义）。

NC 中的急停过程

依据 EN 418，控制系统中用于实现急停状态的内部功能的预定义顺序为：

1. 零件程序的执行中断。
   在针对各轴设置的时间内，对所有机床轴进行制动：
   MD36610 $MA_AX_EMERGENCY_STOP_TIME（故障下制动斜坡的持续时间）
   此时可实现的最大制动斜坡由相应驱动的最大制动电流定义。 最大制动电流通过设定“设定值 = 0”达到（快速制动）。
2. 接口信号复位：
   DB11 DBX6.3（BAG 就绪）
3. 接口信号置位：
   DB10 DBX106.1（急停生效）
4. 显示报警 3000“急停”。

5. 在可设置的延时结束后，机床轴的控制器使能复位。
   延时在以下机床数据中进行设置：
   MD36620 $MA_SERVO_DISABLE_DELAY_TIME（伺服使能封锁延时）
   请遵循以下设置规则：MD36620 ≥ MD36610
6. 所有机床轴均于控制系统内部切换至跟踪模式。
   此时机床轴不再处于位置闭环控制中。

机床上的急停过程

机床上的急停过程只由机床制造商决定。

参考 NC 中的过程，此时须注意：

- NC 中的急停过程通过以下接口信号启动：
  DB10 DBX56.1（急停）
  机床轴处于静止状态后，必须按照 EN 418 中断供电。

  说明

  断电由机床制造商负责执行。

- PLC I/O 的数字量和模拟量输出不受 NC 中急停过程的影响。
  若需使单个输出在急停时启用特定状态或电压电平，则须由机床制造商在 PLC 用户程序中进行设置。
- NCK I/O 的快速数字量输出不受 NC 中急停过程的影响。
  若需使单个输出在急停时启用特定状态，那么机床制造商必须在 PLC 用户程序中通过接口信号将所需的状态传输至 NC。
  DB10 DBB4-7

  说明

  急停时，若需使 NC 中的过程与上文描述不同，那么接口信号 DB10 DBX56.1（急停）不可置位，直至达到由机床制造商在 PLC 用户程序中定义的急停状态。

  只要该接口信号未置位，且不存在其他报警，那么所有接口信号在 NC 中生效。 这样便可实现各种制造商专用（以及轴专用、主轴专用和通道专用）急停状态。

12.5 急停应答

根据 EN 418，急停元件的复位只允许通过手动操作该元件来实现。

急停元件的复位不可独立触发重启指令。

直至对执行的所有急停元件分别进行手动复位后，才可重启机床。

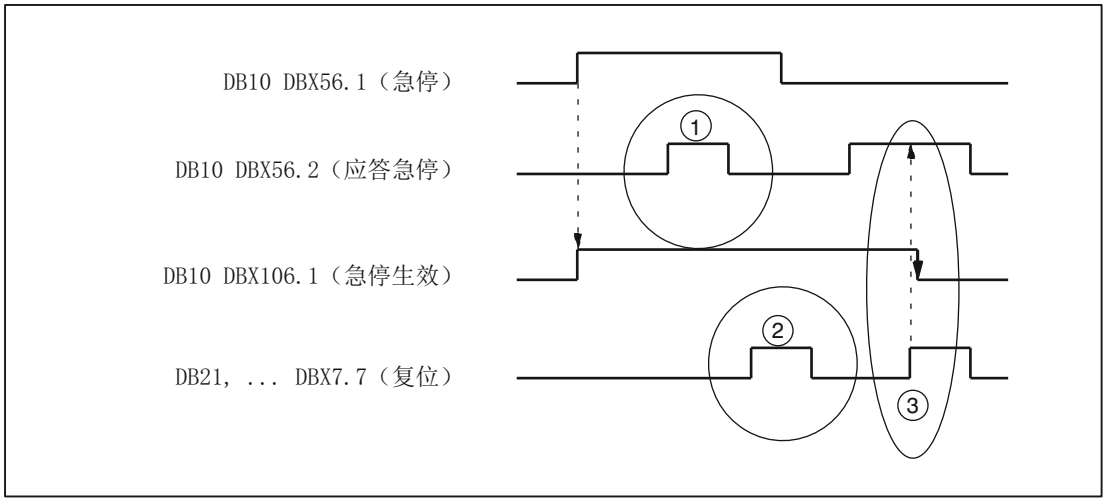
急停应答

仅在接口信号 DB10 DBX56.2（应答急停）和接口信号 DB11 DBX0.7（BAG 复位）先后置位时，急停状态才重新复位。

此时须注意，接口信号 DB10 DBX56.2（应答急停）和 DB21, ... DBX7.7（复位）必须一同置位一段时间，最少要到接口信号 DB10 DBX106.1（急停生效）复位。

说明

只通过接口信号 DB21, ... DBX7.7（复位）无法复位急停状态。



(1) DB10 DBX56.2（应答急停）无效

(2) DB21, ... DBX7.7（复位）无效

(3) DB10 DBX56.2 和 DB21, ... DBX7.7 将 DB10 DBX106.1（急停生效）复位

图 12-1 复位急停状态

影响

通过复位急停状态：

- 于控制系统内部为所有机床轴：
  - 设置控制器使能。
  - 取消跟踪运行。
  - 激活位置闭环控制。
- 置位以下接口信号：
  DB31, ... DBX60.5（位置闭环控制生效）
  DB11 DBX6.3（BAG 就绪）

- 复位以下接口信号：
  DB10 DBX106.1（急停生效）
- 删除报警 3000“急停”。
- 终止 NC 所有通道中的零件程序执行。

PLC I/O 和 NC I/O

PLC I/O 和 NC I/O 必须通过 PLC 用户程序重新设置到机床运行时的状态。

断电/上电（电源 off/on）

急停状态也可通过关闭和接通控制系统（断电/上电）复位。

前提条件：

控制系统启动时，接口信号 DB10 DBX56.1（急停）不允许置位。

12.6 数据表

12.6.1 机床数据

12.6.1.1 进给轴/主轴专用机床数据

编号	名称：$MA_	说明
36610	AX_EMERGENCY_STOP_TIME	故障状态下制动斜坡的持续时间
36620	SERVO_DISABLE_DELAY_TIME	伺服使能封锁延时

12.6.2 信号

12.6.2.1 发送至 NC 的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
急停	DB10.DBX56.1	DB2600.DBX0.1
应答急停	DB10.DBX56.2	DB2600.DBX0.2

12.6.2.2 从 NC 发出的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
急停生效	DB10.DBX106.1	DB2700.DBX0.1

12.6.2.3 发送至 BAG 的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
BAG 复位	DB11.DBX0.7	DB3000.DBX0.7

P1：端面轴

13

13.1 功能

端面轴

在“车削”工艺中，端面轴指的是垂直于主轴的对称轴或纵轴 Z 运行的机床轴。

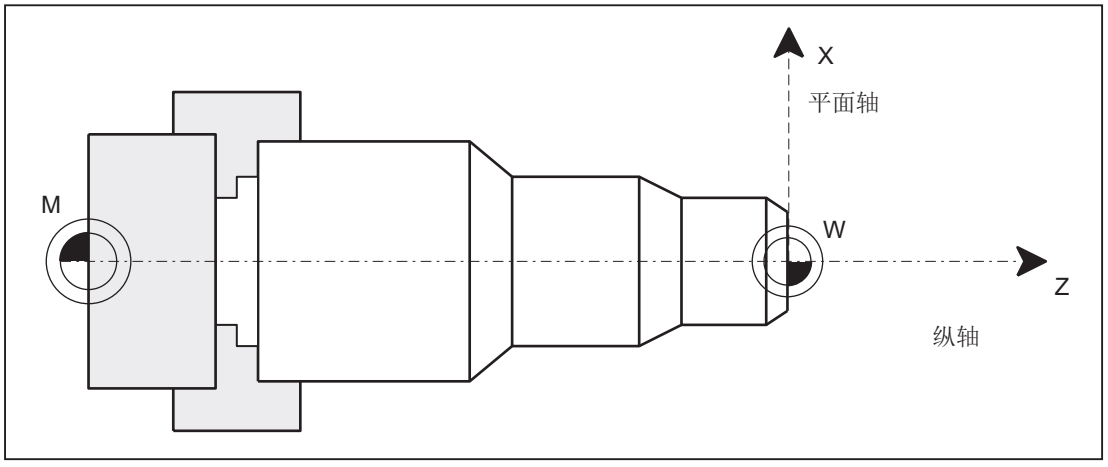


图 13-1 机床坐标系中端面轴的位置

几何轴作为端面轴

通道的每根几何轴均可定义为端面轴。

为了将几何轴作为端面轴，可同时或单独使能及激活下列功能：

- 以直径编程和显示
- 恒定切削速度 G96/G961/G962 的参考轴

通道中的多个端面轴

在通道中引入多个端面轴时，直径编程和 G96/G961/G962 的参考轴功能会解耦。直径编程和 G96/G961/G962 的参考轴功能可对不同的端面轴生效（见下表）。

	以直径 编程和显示			**G96 / G961 / G962 的参考轴**	
允许的轴类型：	几何轴		直线通道轴	几何轴	
在通道中选择：	一	m 中选 3	m 中选 n	一	3 中选一
特殊作用： **机床数据：**	通道 MD20100	轴 MD30460		通道 MD20100	
编程：	DIAM* 针对通道模态生效 G 功能组 29			SCC[<轴>] 针对通道模态生效 G96 / G961 / G962 的参考轴	
在跨通道取轴时接收：	DIAM*A[<轴>] 针对轴模态生效				
针对轴非模态生效的直径/半径编程	DAC、DIC；RAC、RIC 针对轴逐段生效，仅编程				
DIAM*：DIAMOF, DIAMON, DIAM90, DIAMCYCOF DIAM*A[<轴>]:DIAMOFA[<轴>]、DIAMONA[<轴>]、DIAM90A[<轴>]、DIACYCOFA[<轴>]、DIAMCHANA[<轴>] <轴>：几何轴/通道轴或机床轴的轴名称					

说明

回转轴不允许作为端面轴。

基于直径的数据

DIAMON/DIAMONA[<轴>]

通过 DIAMON/DIAMONA[<轴>] 激活直径编程后（参见“编程 (页 940)”），以下数据基于直径设定：

- 工件坐标系中端面轴的显示数据：
  - 设定位置和实际位置
  - 剩余行程
  - REPOS 偏移
- “JOG”运行方式：
  - 通过手轮进行增量手动运行时的步进尺寸（INC）（取决于生效的 MD）
- 零件程序编程：
  - 终点位置，与参考模式（G90 / G91）无关
  - 圆弧编程（G2 / G3）的插补参数，若其采用零件程序指令 AC 绝对编写。
- 基于工件坐标系（WCS）读取的实际值：
  - $AA_MW[<端面轴>]
    测量功能 MEAS（删除剩余行程的测量）和 MEAW（不删除剩余行程的测量）的系统变量
  - $P_EP[<端面轴>]
  - $AA_IW[<端面轴>]

DIAM90/DIAM90A[<轴>]

通过 DIAM90/DIAM90A[<轴>] 激活取决于参考模式的直径编程后，以下数据始终基于直径显示，与参考模式（G90 / G91）无关：

- 实际值
- 基于工件坐标系（WCS）读取的实际值：
  - $AA_MW[<端面轴>]
    测量功能 MEAS（删除剩余行程的测量）和 MEAW（不删除剩余行程的测量）的系统变量
  - $P_EP[<端面轴>]
  - $AA_IW[<端面轴>]

DIAMCYCOF/DIACYCOFA[<轴>]

编写 DIAMCYCOF 或 DIACYCOFA[<轴>] 时控制系统内部会切换至半径编程。DIAMCYCOF 或 DIACYCOFA[AX] 前生效的状态会基于直径编程继续对 HMI 显示。

恒定基于半径的数据

对于端面轴，下列数据**始终**基于半径输入、编写和显示：

- 零件程序编程
  - CIP 圆弧编程的插补参数
- 偏移
  - 刀具补偿
  - 可编程框架和可设定框架
  - 外部零点偏移
  - DRF 偏移和预设定偏移
  - 等
- 工作区域限制
- 软件限位开关
- 进给率
- 基于机床坐标系的显示数据
- 轴、VSA 和 HSA 的维护图的显示数据

恒定基于半径数据的功能扩展：

对于 PLC 轴、通过 FC18 或只通过 PLC 控制的轴：

- 对 PLC 轴的半径尺寸设定同样适用于带直径功能的多端面轴，且独立于通道专用或轴专用直径编程。
- 在 JOG（Inc）运行方式下，PLC 轴归属于通道状态。若直径编程生效且 MD20624 $MC_HANDWH_CHAN_STOP_COND 位 15 = 0，那么将只运行设定增量的一半行程。

13.2 参数设置

将几何轴定义为端面轴

通过以下机床数据将**一根**几何轴定义为通道中的端面轴：

MD20100 $MC_DIAMETER_AX_DEF（带端面轴功能的几何轴）

示例：

MD20100 $MC_DIAMETER_AX_DEF="X" ; 几何轴 X 是通道中的端面轴。

对于此轴，即可采用直径编程，也可通过 G96/G961/G962 指定恒定切削速度。

通道中的多个端面轴

通过以下机床数据定义更多端面轴，这些轴可启用轴专用直径编程功能：

MD30460 BASE_FUNCTION_MASK（轴功能）

位	值	含义
2	0	不允许轴专用的直径编程。
	1	允许轴专用的直径编程。

说明

MD30460 位 2 = 1 这一设置只适用于直线轴。

一根轴可同时在 MD20100 和 MD30460（位 2）中定义。此时通道专用的 MD20100 相比轴专用的 MD30460 具有更高优先级。

通过 MD20100 在启动中为端面轴：

- 指定 G96/G961/G962 功能。
- 指定通道专用直径编程 DIAMON、DIAMOF、DIAM90、DIAMCYCOF。

启动后，这些轴启用轴专用初始设置 DIAMCHANA[<轴>]。

通过 MD30460 位 2 额外使能轴专用指令 DIAMONA[<轴>]、DIAMOFA[<轴>]、DIAM90A[<轴>]、DIACYCOFA[<轴>]、DIMCHANA[<轴>]。

刀具参数的尺寸设定

通过 MD20360 $MC_TOOL_PARAMETER_DEF_MASK 可为通道中定义的所有端面轴将以下刀具参数激活作为直径值：

MD20360 $MC_TOOL_PARAMETER_DEF_MASK		
位	值	含义
1	1	端面轴刀具长度作为直径
2	1	磨损或刀具长度作为直径时报警，以及平面切换
3	1	端面轴中的框架中零点偏移作为直径 框架内部将零点偏移作为半径值保存。在直径编程和半径编程间切换时不会进行换算。
4	1	预设值作为直径

MD20360 $MC_TOOL_PARAMETER_DEF_MASK		
位	值	含义
5	1	端面轴中的外部零点偏移作为直径 在直径编程和半径编程间切换时不会进行换算。
6	1	端面轴实际值作为直径
7	1	端面轴实际值作为实际值显示
10	1	无刀具生效时，生效的可定向刀架的刀具分量
11	1	$TC_DP6 的分析作为直径
12	1	$TC_DP15 的分析作为刀具直径磨损
15	1	循环掩码中端面轴的增量值作为直径

MD20100 $MC_DIAMETER_AX_DEF 和 MD30460 $MA_BASE_FUNCTION_MASK 位 2 中的半径编程根据 MD20360 $MC_TOOL_PARAMETER_DEF_MASK 如下启用：

MD20360 $MC_TOOL_PARAMETER_DEF_MASK		
位	值	含义
3	0	零点偏移 $P_EXTFRAME 和框架 对于端面轴，框架中的零点偏移总是作为半径值纳入计算。
5	0	外部零点偏移（轴叠加） 对于端面轴，外部零点偏移总是作为半径值纳入计算。
8	1	WCS 中总是以半径显示剩余行程
9		对于所有端面轴，采用以下设置时：MD11346 $MN_HANDWH_TRUE_DISTANCE==1
	0	• 若通道专用或轴专用直径编程对该轴生效，那么运行的行程将为设定手轮增量的一半。
	1	• 运行行程始终为设定手轮增量的一半。
13	1	在沿圆弧点动时，圆心坐标始终为半径值，参见 SD42690 $SC_JOG_CIRCLE_CENTRE
14	1	对于循环掩码，端面轴的绝对值基于半径。

通过手轮进行手动运行时的特性

直径编程生效时，端面轴在通过手轮进行手动运行（MD11346 $MN_HANDWH_TRUE_DISTANCE == 1 或 3）时的特性通过以下机床数据调整：

MD20624 $MC_HANDWH_CHAN_STOP_COND

位	值	含义
15	0	只运行设定增量的一半。
	1	完全运行设定的增量。

以直径显示位置值

设置了 MD27100 $MC_ABSBLOCK_FUNCTION_MASK 位 0 = 1 时，
端面轴的位置值总是作为直径值显示。

启动、复位后的通道专用初始设置

启动、复位或零件程序结束后，G 功能组 29（DIAMON、DIAM90、DIAMOF、DIAMCYCOF）的通道专用初始设置通过以下机床数据定义：

- MD20150 $MC_GCODE_RESET_VALUE

或（MD20110 $MC_RESET_MODE_MASK 位 0 = 1 时）：

- MD20152 $MC_GCODE_RESET_MODE

用户可通过事件控制的程序调用（Prog-Event）设置所需的状态。

若 G96/G961/G962 为启动后的初始设置，则必须通过 MD20100 $MC_DIAMETER_AX_DEF 定义一根端面轴，否则会触发报警 10870。

保留 G96 / G961 / G962 的参考轴：

MD20110 $MC_RESET_MODE_MASK，位 18=1，复位或零件程序结束时

MD20112 $MC_START_MODE_MASK，位 18=1，零件程序开始时

也可不启用端面轴，在 MD20100 中通过 SCC[AX] 指定 G96/G961/G962 的参考轴。在此情形下不允许通过 G96 激活恒定切削速度。更多相关信息请见：

文档：

编程手册之基本原理分册，进给控制和主轴运动
恒定切削速度（G96/G961/G962、G97/G971/G972、G973、LIMS、SCC）

13.3 编程

端面轴既可基于直径，也可基于半径编写。通常情况下采用直径编程，即使用双倍的行程值编写，从而可在零件程序中直接取用技术图纸中的对应尺寸值。

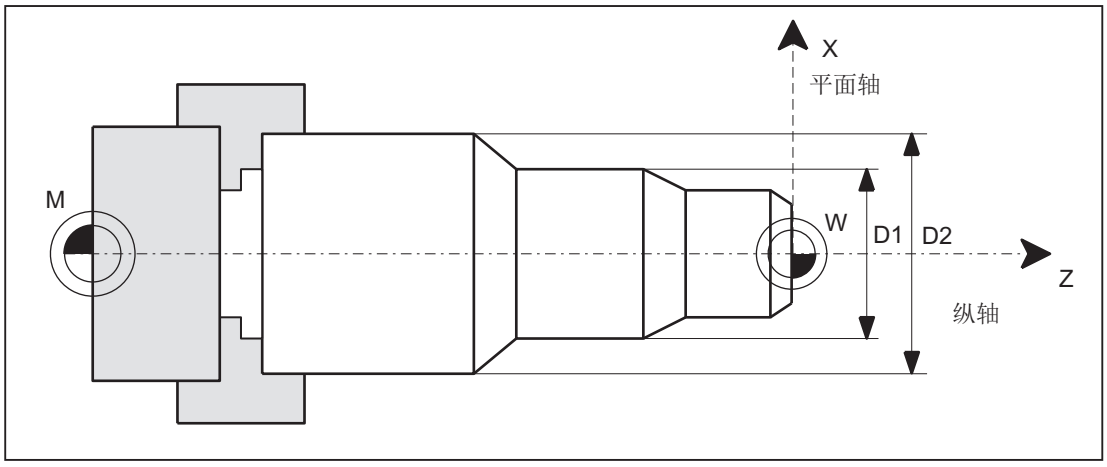


图 13-2　　端面轴采用直径设定（D1、D2）

激活/取消直径编程

通道专用的直径编程

直径编程通过 G 功能组 29 中的模态生效的零件程序指令激活或取消：

- DIAMON：直径编程 ON
- DIAMOF：直径编程 OFF 或半径编程 ON
- DIAM90：直径编程或半径编程取决于参考模式：
  - 直径编程 ON，配合绝对尺寸设定 G90
  - 半径编程 ON，配合增量尺寸设定 G91
- DIAMCYCOF：对 G90 和 G91 启用半径编程，对于 HMI，该 G 指令组中最近一次生效的 G 指令继续生效。

只以通道的端面轴为基准。

针对通道内多个端面轴的轴专用直径编程

说明

额外指定的轴必须通过 MD30460 $MA_BASE_FUNCTION_MASK 位 2 = 1 激活。
指定的轴必须是通道内已知的轴。允许采用几何轴、通道轴或机床轴。
不允许在同步动作中编程。

下列轴专用模态生效指令可在一个零件程序段中多次编写：

- DIAMONA[<轴>]：为 G90、G91、AC 和 IC 启用直径编程
- DIAMOFA[<轴>]：直径编程 OFF 或半径编程 ON
- DIAM90A[<轴>]：直径编程或半径编程取决于参考模式：
  - 直径编程 ON，配合绝对尺寸设定 G90 和 AC
  - 半径编程 ON，配合增量尺寸设定 G91 和 IC
- DIACYCOFA[轴]：对 G90 和 G91 启用半径编程，对于 HMI，该 G 指令组中最近一次生效的 G 指令继续生效。
- DIAMCHANA[轴]：接收直径编程通道状态
- DIAMCHAN：所有轴通过 MD30460 位 2 = 1 接收直径编程通道状态

相比通道设置，轴专用模态生效指令具有更高优先级。

13.4 前提条件

跨通道取轴

基于来自零件程序的 GET 请求，在通过 RELEASE[轴] 进行跨通道取轴时，系统会在新通道中为附加端面轴接收直径编程状态。

在同步动作中取轴

在同步动作中跨通道取轴时，若为下列情形，那么端面轴的轴专用直径编程状态将带入新通道：

- 通过 MD30460 位 2 = 1 为端面轴使能了轴专用直径编程。
- 在让渡轴的通道中，未将端面轴划归至通道专用直径编程下。

通过系统变量 $AA_DIAM_STAT[<轴>] 可查询生效的尺寸设定。

通过轴容器旋转取轴

轴容器旋转可能会使通道轴对机床轴的指定关系发生变化。但对于通道轴，直径编程的当前状态在旋转后仍保留。 这也适用于通道状态和轴状态，因为在“将机床数据设置生效”的时间点，对于轴容器中的所有轴，MD30460 $MA_BASE_FUNCTION_MASK 中的状态相同。

13.5 示例

示例 1

X 是通过 MD20100 $MC_DIAMETER_AX_DEF 定义的端面轴。

Y 为几何轴，U 为辅助轴。 这两根轴同时又在 MD30460 $MA_BASE_FUNCTION_MASK 中通过位 2 = 1 定义为基于直径设定的端面轴。

DIAMON 在启动后不生效。

程序代码	注释
N10 G0 G90 X100 Y50	; 无直径编程生效
N20 DIAMON	; 通道专用直径编程，对 X 生效
N30 Y200 X200	; 尺寸设定：X 以直径，Y 以半径
N40 DIAMONA[Y]	; 轴专用模态生效直径编程， ; 对 Y 生效
N50 Y250 X300	; 尺寸设定：X 和 Y 以直径
N60 DIAM90	; 尺寸设定：X G90/AC 以直径，G91/IC 以半径
N70 Y200	; Y：继续轴专用模态生效直径编程
N75 G91 Y20 U=DIC(40)	; 尺寸设定：Y 以直径，U 以直径逐段 IC
N80 X50 Y100	; 尺寸设定：X 以半径（G91），Y 以直径
N85 G90 X100 U200	; 尺寸设定：X 以直径，U 以半径
N90 DIAMCHANA[Y]	; Y 接收通道状态 DIAM90
N95 G91 X100 Y100	; 尺寸设定：X 和 Y 以半径（G91）
N100 G90 X200 Y200	; 尺寸设定：X 和 Y 以直径

示例 2

和上面的示例一样，端面轴设定基于直径。

X 和 Y 处于通道 1 中，且在通道 2 中已知，即支持跨通道取轴。

程序代码	注释
通道 1	
N10 G0 G90 X100 Y50	; 无直径编程生效
N20 DIAMON	; 通道专用直径编程，对 X 生效
N30 Y200 X200	; 尺寸设定：X 以直径，Y 以半径
N40 DIAMONA[Y]	; Y 启用轴专用模态生效直径编程
N50 Y250 X300	; 尺寸设定：X 和 Y 以直径
N60 SETM(1)	; 同步标记 1
N70 WAIT(1,2)	; 等待通道 2 中的同步标记 1
通道 2	
...	
N50 DIAMOF	; 通道 2 无直径编程生效

程序代码	注释
...	
N100 WAIT(1,1)	; 等待通道 1 中的同步标记 1
N110 GETD(Y)	; 直接取 Y 轴
N120 Y100	; Y 划归于通道 2 中的通道专用直径编程， ; 即尺寸设定基于半径

13.6 数据表

13.6.1 机床数据

13.6.1.1 通道专用机床数据

编号	名称：$MC_	说明
20050	AXCONF_GEOAX_ASSIGN_TAB[n]	几何轴指定为通道轴
20060	AXCONF_GEOAX_NAME_TAB[n]	通道中的几何轴名称
20100	DIAMETER_AX_DEF	带端面轴功能的几何轴
20110	RESET_MODE_MASK	启动和复位/零件程序结束后控制系统的初始设置
20112	START_MODE_MASK	确定 NC 启动时控制系统的初始设置
20150	GCODE_RESET_VALUES[n]	G 功能组的初始设置
20152	GCODE_RESET_MODE[n]	复位/零件程序结束时的 G 指令初始设置
20360	TOOL_PARAMETER_DEF_MASK	刀具参数的定义
20624	HANDWH_CHAN_STOP_COND	通过手轮进行手动运行时的特性的定义
27100	ABSBLOCK_FUNCTION_MASK	设置采用绝对值的程序段显示

13.6.1.2 进给轴/主轴专用机床数据

编号	名称：$MA_	说明
30460	BASE_FUNCTION_MASK	轴功能

P3：SINUMERIK 840D sl 的 PLC 基本程序 14

14.1 简要说明

简介

PLC 基本程序用于组织 PLC 用户程序和 NC、HMI 及 MCP 之间的信号和数据交换。依据信号和数据可划分出下列组别：

- 循环信号交换
- 事件控制的信号交换
- 信息

循环信号交换

循环交换的数据主要由位域组成。

- 其包含了从 PLC 传输至 NC 的**指令**（例如启动、停止等）以及 NC 的**状态信息**（例如程序运行、程序中断等）。
- 位域划分为用于以下功能的信号：
  - BAG
  - 通道
  - 轴/主轴
  - 通用 NC 信号

循环数据交换由基本程序在 PLC 循环开始处（OB1）执行。这样例如可确保 NC 发出的信号在一整个循环内保持恒定。

事件控制的 NC → PLC 信号交换

必须依据工件程序执行的 PLC 功能通过工件程序中的辅助功能触发。执行包含辅助功能的程序段时，辅助功能的类型决定了 NC 是否必须等待此功能的执行（例如换刀），或者是否随工件加工一并执行该功能（例如采用链式刀库的铣床上的刀具准备）。

为了尽可能减小对 NC 处理的影响，数据传输必须尽可能快速且可靠地实现。为此，系统通过报警和应答对数据传输进行控制。基本程序能够检测信号和数据，将其应答至 NC，并在循环开始处将数据传输至用户接口。若数据不需要用户应答，那么 NC 处理不会受此影响。

事件控制的 PLC → NC 信号交换

PLC 将一个任务传输至 NC（例如运行辅助轴）时，总是会发生“事件控制的 PLC → NC 信号交换”。此时数据传输同样通过应答进行控制。从用户程序执行时，此类信号交换通过一个 FB 或 FC 触发。

对应的 FB（功能块）和 FC（功能调用）会和基本程序一同提供。

信息

用户消息通过基本程序采集和预处理。消息信号会通过一个议定的位域传输至基本程序。基本程序会检测这些信号，并在出现消息结果时通过 ALARM S/SQ 功能将其记录至 PLC 的报警缓存。若存在 HMI（例如 SINUMERIK Operate），那么消息会被传输至 HMI 并加以显示。

PLC / HMI 数据交换

在此类数据交换中，HMI 接收主动权，作为总线系统上的客户端。HMI 请求数据或描述数据。PLC 在循环控制点通过操作系统处理这些任务。PLC 基本程序不参与此类传输。

说明

机床的功能很大程度上由 PLC 程序决定。工作存储器中的每个 PLC 用户程序均可通过编程设备修改。

对用户数据块的专有知识保护

为了保护用户数据块（OB、FB 和 FC）中包含的专有知识，可通过 SIMATIC STEP 7 中的 SBP-Tool（SIMATIC Block Protection，SIMATIC 程序块保护）对其进行加密。若未给出加密时使用的密码，则无法对该数据块进行打开、排故和修改。

加密时须指定自动化系统，数据块需要能够在该系统的 PLC-CPU 上执行：SIMATIC 和/或 SINUMERIK PLC-CPU。

对数据块的操作（例如载入至 CPU）不受加密影响。

前提条件

SIMATIC STEP 7 版本 5.5 SP3 或更高

14.2 PLC-CPU 的基础数据

PLC-CPU 的基础数据

文档：

SINUMERIK NCU 中集成的 PLC-CPU 的基本数据概览请见：
设备手册 NCU 7x0.3 PN，“技术数据”一章

说明

集成驱动的 I/O 地址

4096 以上的 I/O 地址为 NCU 集成的驱动预留，不允许占用。

PLC 基本程序的功能

范围		
	轴/主轴	31
	通道	10
	BAG	10
功能		

	状态信号/控制信号	+
	M 解码器（M00-99）	+
	G 功能组解码器	+
	辅助功能分配器	+
	报警控制的辅助功能传输	+
	基于列表的 M 解码	+
	由 PLC 运行轴/主轴	+
	ASUB 接口	+
	故障消息/运行消息	+
	传输 MCP（机床控制面板）信号和 HHU（手持操作设备）信号	+
	手持操作设备显示控制	+
	NC 变量和 GUD 的读写	+
	PI 服务	+
	刀具管理	+
	星形/三角形切换	+
M : N		+
Safety Integrated		+
程序诊断		+

14.3 PLC 操作系统版本

PLC 操作系统的版本信息如下获取：

- SINUMERIK Operate 操作界面：“操作区域切换”>“诊断”>“版本”⇒ 版本数据；NCU 系统软件：选择“PLC”>“详细信息”⇒ 版本数据；NCU/PLC 系统软件：PLC 操作系统的版本在第一行中“PLC 3xx…”下显示。

  提示

  显示的版本针对 SINUMERIK。 其与基础 SIMATIC CPU 不兼容。

- SIMATIC STEP 7，HW Config：SINUMERIK 模块载体中 PLC-CPU 的属性：“属性 – CPU 3xx…” > “订货号/固件”：xxxx / **Vx.y.z**

  提示

  显示的是基础 SIMATIC CPU 的版本。

14.4 PLC 运行方式开关

PLC 运行方式开关位于 NCU 模块的正面。通过 PLC 运行方式开关可设置下列 PLC 运行方式：

S [1)]	含义	注释
0	RUN-P	无需激活口令便可在 PLC 程序上进行修改
1	RUN	通过编程设备（PG）只可进行读取访问。设置口令后，才可在 PLC 程序上执行修改。
2	STOP	PLC 用户程序的执行停止，所有 PLC 输出设置为替换值。
3	MRES	PLC 进入 STOP 状态，之后执行 PLC 总复位（缺省数据）。
1) PLC 运行方式开关的开关位置		

文档

对位于 NCU 模块正面的 PLC 运行方式开关的位置的详细说明，以及其在 NC 和 PLC 总复位方面的应用请见：

调试手册, CNC:NC、PLC、驱动”

- “接通启动”>“用于启动的操作元件和显示元件”章节
- “接通启动”>“NC 和 PLC 总复位”章节
- “一般提示”>“独立的 NC 和 PLC 总复位”章节

14.5 预留资源（计时器、计数器、FC、FB、DB、I/O）

预留资源（计时器、计数器、FC、FB、DB、I/O）

以下组件为基本程序预留：

- **计时器**
  不预留。
- **计数器**
  不预留。

- **FC、FB、DB**
  为基本程序预留了 FC0 至 FC29 以及 FB0 至 FB29。此外，FC 和 FB 从 1000 到 1023 的编号带也预留。为数据块预留了 DB1 至 DB61，DB71 至 DB80。此外，DB 从 1000 到 1099 的编号带也预留 。未激活通道、轴/主轴、刀具管理的数据块可由用户自由使用。
- **I/O 区域**
  PLC 用于输入和输出的 I/O 地址量分别为 16384 字节。其中 4096 起的地址区域预留给集成驱动。但是正如 STEP 7 所建议的，模块的诊断地址可置于最上方的地址区域。此外在 SIMATIC 300 工作站的 Rack 0 中，地址区域 4080 至 4097 被指定给 NC、CP 和 HMI (NCU 7x0.3)。

14.6 调试 PLC-CPU 的硬件配置

对 PLC-CPU 调试的详细说明请见：

文档

调试手册, CNC:NC、PLC、驱动”

- 章节：“将 PG/PC 与 PLC 连接”
- 章节：“PLC 调试”
- 章节：“基本原理”> “PLC 程序”
- 章节：“一般提示”>“独立的 NC 和 PLC 总复位”
- 章节：“一般提示”>“将 PG/PC 接入网络（NetPro）”

14.7 调试 PLC 程序

14.7.1 安装基本程序

对基本程序安装的详细说明请见：

文档

调试手册, CNC:NC、PLC、驱动；章节：“PLC 调试”>“创建 PLC 程序”

说明

安装/升级

安装 SINUMERIK 840D 的 Toolbox 前，必须已安装好 SIMATIC STEP 7。

在执行完 STEP 7 的升级后，建议从 Toolbox 重新安装 STEP 7 的硬件补充。

内容

基本程序中包含了 OB 源程序（包括标准参数设置）、手持操作设备的接口符号和 DB 模版、以及 M 解码功能。

14.7.2 基本程序的使用

对于每台设备（机床），必须通过 STEP 7 软件在项目中创建一个新的 CPU 程序（例如“Drehma1”）。

注释

项目的目录结构，以及项目和用户程序的创建步骤请参见相应的 SIMATIC 文档。

步骤

使用 SIMATIC 管理器通过“文件”>“打开”>“库”菜单复制基本程序模块。

必须从库中复制以下要素：

- 从模块容器：FC、FB、DB、OB、SFC、SFB、UDT
- Source_files（从资源容器）：GPOB840D
- 必要时 MDECLIST、BHG_DB 和其他
- 符号表（从符号容器）

与 STEP 7 的兼容性

基本程序与当前的 STEP 7 版本无关联。

14.7.3 版本标识

基本程序

基本程序的版本（包括控制系统类型）在操作界面的版本画面中显示。

控制系统类型的代码如下：

DB17.DBD0 的左对齐十进制位（字节 0）	控制系统类型
03	SINUMERIK 840D sl (NCU 7x0)

用户程序的版本标识

用户也可在版本画面中显示自定义的 PLC 版本标识。为此须在一个任意数据块中定义一个 STRING 类型的数据，最多 **54** 个字符。不执行版本编译，只接收输入的字符串。该字符串的参数设置通过 FB1 上的指针进行。因此应对数据块进行符号式定义。参见模块说明 FB1 (页 1051)。

版本标识可按如下方式在字符串中进行格式化：

- xx.yy
- xx.yy.zz
- ww.xx.yy.zz
- vv.ww.xx.yy.zzz
- x.y
- x.y.z
- w.x.y.z
- v.w.x.y.z

除版本标识外，还可在字符串中输入在进行相应的格式化时 HMI 版本画面中显示的日期。此时不执行日期编译，只接收输入的字符串。可与版本标识组合。目前支持的格式有：

- 00/00/0000
- 0000/00/00
- 00/00/00

示例：

- “测试程序版本 01.02.03 01/01/2015”
- “1.2 2015/01/01 测试程序”
- “01/01/15 版本 01.02 测试程序”

14.7.4 机床程序

机床程序由机床制造商借助基本程序的库路径创建。机床程序中包含了逻辑运算和机床流程。此外其还可用于操控发送至 NC 的接口信号。更复杂的对 NC 通讯功能则通过基本程序的 FC、FB 模块触发和执行，例如读取和写入 NC 数据、刀具管理应答等。

机床数据可采用不同的 STEP 7 创建语言创建，例如 AWL、KOP、FUP、S7-HIGRAPH、S7GRAPH、SCL。完整的机床程序必须以正确的顺序创建和编译。

也就是说，若模块由其他模块调用，则其须在执行调用的模块**前**进行编译。

若之后的程序开发中这些被调用的模块又在接口（VAR_INPUT、VAR_OUTPUT、VAR_IN_OUT、VAR）中进行了修改，那么之后必须对执行调用的模块及其关联的所有模块进行编译。此步骤大体上也适用于 FB 的实例数据块。若未遵循此顺序，那么在 STEP 7 中进行回译时会出现时间戳冲突。这样一来无法确保模块能够回译，且在使用“模块状态”功能时会出现不必要的冲突。此外，对于在梯形图或单个指令（增量模式）中创建的模块，建议以 ASCII-AWL 通过 STEP 7 编辑器生成。

14.7.5 数据备份

PLC-CPU 不保存符号名称，而只保存模块参数 VAR_INPUT、VAR_OUTPUT、VAR_IN_OUT、VAR 的数据类型说明，以及全局数据块的数据类型。

说明

没有机床对应的项目时，便无法实现合理的回译。 特别是在使用“模块状态”功能，或对 PLC-CPU 程序进行必要的后续修改时会出现此情形。 因此，PLC-CPU 中的 STEP 7 项目一定要在机床上保留。 这会在维修时提供很大帮助，因为不必重新恢复对应的原始项目，从而能节省不必要的时间成本。

若存在 STEP 7 项目，并按照上文介绍的规则进行了创建，那么在该机床上便可在 PLC-CPU 中处理符号。 必要时也可将机床的源程序作为“.awl”文件保存，用于可能需要的升级。

所有组织模块的源程序和所有实例数据块须始终存在。

14.7.6 PLC 批量调试，PLC 存档

在将模块载入 PLC-CPU 后，可通过 HMI 操作界面创建一个批量存档，用于机床上的数据备份。 数据备份应当在模块载入后直接在 PLC 停止状态下进行，以确保数据的一致性。 此数据备份不能替代 SIMATIC 项目的备份，因为批量存档中只能保存二进制数据。 这样例如无法提供符号信息。 此外 CPU-DB（SFC22-DB）和 CPU 中生成的 SDB 也不会被备份。

选择 SINUMERIK 存档程序

另一种方案是直接从 SIMATIC 项目生成 PLC 批量存档：

- 在 SIMATIC 管理器中打开“设置”对话框：菜单栏“其他”>“设置”
- 打开“存档”选项卡
- 在下拉列表栏“优先的存档程序”中选择 SINUMERK 存档程序“SINUMERIK (*.arc)”。

启动 SINUMERIK 存档程序

SINUMERIK 存档程序在 SIMATIC 管理器中通过菜单项 “文件”>“存档”启动。

设定完存档名称后便会生成 PLC 存档。 若项目包含多个程序路径，可在对话框中选择 S7 程序，为其创建 PLC 存档。 所选程序路径中所包含的所有模块都会被存档（除了通过 SFC22 在 CPU 中（在线）创建的数据块）。

可为存档程序激活或取消“SDB 存档”功能。 若“SDB 存档”被激活，那么创建的 PLC 存档中将只包含所选程序路径下的系统数据块（SDB）。

自动化

批量存档的生成也可自动进行（可与 STEP 7 的命令接口相比较）。 生成批量存档是对命令接口的扩展。

此扩展中提供以下功能：

这些功能（此处通过 VB-Script 显示）在调用服务器实例化和 Magic 调用后才可用：

```
Const S7BlockContainer = 1138689, S7PlanContainer = 17829889

Const S7SourceContainer = 1122308

set S7 = CreateObject("Simatic.Simatic.1")

实例化 STEP 7 的 rem 命令接口

Set S7Ext = CreateObject("SimaticExt.S7ContainerExt")

Call S7Ext.Magic("")
```

功能：

- `Function Magic(bstrVal As String) As Long`
- `Function MakeSerienIB (FileName As String, Option As Long, Container As S7Container) As Long`

说明

Function **Magic**(bstrVal As String) As Long

通过调用实现对特定功能的访问。 此功能必须在服务器实例化后调用一次。 bstrVal 的值可为空。 此时系统会检查 STEP 7 版本及 Autoexec 中的路径设定是否正确。 返回值为 0 时，这些功能使能。

返回值 (-1) = 错误的 STEP 7 版本

返回值 (-2) = Autoexec.bat 中无记录

Function **MakeSerienIB**(FileName As String, Option As Long, Container As S7Container) As Long

“选项”参数：

0:	一般批量调试文件，带总复位。
位 0 = 1：	批量调试文件，无总复位。 SDB 在项目中时，此选项不生效。 之后总是执行总复位。
位 1 = 1：	批量调试文件，带 PLC 重启

返回值：

0	= 正常
-1	= 功能不可用，“Magic”功能提前调用
-2	= 文件名不可生成
-4	= 参数容器无效，或模块容器为空
-5	= 内部故障（存储空间请求被 Windows 拒绝）
-6	= 内部故障（STEP 7 项目中存在问题）
-7	= 生成批量调试文件时的写入故障（例如磁盘已满）

脚本中的使用

程序代码

```
If S7Ext.Magic("") < 0 Then
    Wscript.Quit(1)
End If
    Set Proj1 = s7.Projects("new")
    set S7Prog = Nothing
    Set s7prog = Proj1.Programs.Item(1) '当只有一个程序时
For Each cont In s7prog.Next
    If (Cont.ConcreteType = S7BlockContainer) Then
    '检查模块容器
    Exit For
    End if
    Cont = Nothing
    Next
Error = S7Ext.MakeSerienIB("f:\dh\arc.dir\PLC.arc", 0, Cont)
' 现在检测故障
```

上面编写的 For Each ... Next 可采用 Delphi 编程语言如下编写（相似的编程也适用于 C、C++ 编程语言）：

程序代码

```
Var
    EnumVar: IEnumVariant;
    rgvar: OleVariant;
    fetched: Cardinal;

//For Each Next
EnumVar := (S7Prog.Next._NewEnum) as IEnumVariant;
While (EnumVar.Next(1,rgvar,fetched) = S_OK) Do Begin
    Cont := IS7Container(IDispatch(rgvar)); //模块容器，
    检查源
    If (Cont.ConcreteType = S7BlockContainer) Then Break;
    Cont := NIL;
End;
```

14.7.7 软件升级

为了实现定义的初始状态，必须在 PLC 软件升级前执行 PLC 总复位。此时所有用户程序（程序块和数据块）均会被删除。PLC 总复位的相关信息请见：

文档：
调试手册 IBN CNC：NC、PLC、驱动；一般提示，
章节：PLC 清零

生成新的 SIMATIC S7 项目

对于新的 NCU 软件版本，通常须为其建立新的 PLC 基本程序。为此需要将基本程序模块传输至用户项目。如果以下程序块和数据块已存在于用户项目中，则不应再将它们与 PLC 基本程序的程序块进行传输：OB1、OB40、OB82、OB86、OB100、FC12 和 DB4。这些块可能已由用户修改，且不应被改写。新的基本程序必须与用户程序相关联。此时请遵循以下步骤：

1. 在复制 PLC 基本程序前，从所有用户模块生成文本文件或源文件。
2. 将新的基本程序模块复制到 SIMATIC S7 项目（相关说明参见“基本程序的使用 (页 951)”章节）
3. 以正确的顺序重新编译所有用户程序“*.awl”（另见“机床程序 (页 953)”）
4. 之后须通过 STEP 7 将重新编译的 SIMATIC S7 项目载入至 PLC。

不过一般情形下只需重新编译 S7 项目的组织块（OB）和实例数据块即可。也就是说，升级前只需生成组织块和实例数据块的源。

NC 变量

对每个 NC 软件版本（包括较旧的版本）均可使用最新的 NC-VAR-Selector。对于较旧的 NC 软件版本，同样可以从最新的总列表选择变量。DB120（变量的标准 DB）的信息内容与软件版本无关。也就是说，在较旧的软件版本中选择的变量不必在软件升级时重新选择。

14.7.8 I/O 模块（FM 模块、CP 模块）

对于较复杂的 I/O 模块（FM 模块、CP 模块），通常需要 STEP 7 的附加包。 附加包中随附了支持模块（FC / FB）。 这些模块包含了用于运行特定模块的专用功能。 这些功能可在用户程序中参数设置和调用。

编号统一

在对操作模块和基本程序模块进行编号统一化时，基本程序的模块编号必须保持不变。 操作模块的模块编号必须通过 STEP 7 改为自由编号。

14.7.9 排故

更换硬件前，请参考本章节中的故障提示、故障原因及解决办法。

故障、原因/说明、解决办法			
序号 故障提示	**故障**	**原因/说明**	**解决办法**
1	未通过 MPI 建议与 PLC 的连接。	MPI 电缆未插入或损坏。 也可能是由于 STEP 7 软件的 MPI 卡相关配置不正确。	尝试使用 PG 在 STEP 7 编辑器中通过“Direct_PLC”建立连接。 此时会显示一些节点地址。 若未显示，则表示 MPI 电缆损坏/未插入。
2	进行 PLC 总复位后仍无法访问 PLC。	系统数据块 SDB 0 以经过修改的 MPI 地址载入。 此时地址被重复设定，从而会引起 MPI 总线冲突。	将所有连接至其他组件的 MPI 电缆拔下。 使用 PG 建立“Direct_PLC”链接。 通过重置更正 MPI 地址。
3	PLC 的 4 个 LED 均闪烁（DI 崩溃）	PLC 中出现系统故障。 **措施：** 必须读取 PLC 的诊断缓存，以对系统故障进行更精确的分析。 为此先要使 PLC 进入停止状态（例如将“PLC”开关置于位置 2）。之后需进行硬件复位。 之后可通过 STEP 7 读取诊断缓存。 诊断缓存中的信息必须提供给热线/研发部分。 若在硬件复位后又收到总复位请求，则须先执行总复位。 之后才可在停止状态下读取诊断缓存。	复位或重新载入 PLC 程序后，系统有时能正常运行。 诊断缓存的内容则始终必须提供给研发部门。

14.8 PLC-CPU 的连接

14.8.1 简介

在 SINUMERIK 840D sl 中，自动化系统 S7-300 的 CPU 被用作 PLC。 PLC-CPU 作为子模块集成至 NCU 组件。 PLC-CPU 的性能数据请参见“PLC-CPU 的基础数据 (页 947)”章节。

14.8.2 PLC-CPU 的属性

SINUMERIK 840D sl 中集成的 PLC 的功能大体上与对应的 SIMATIC S7-300 PLC 相同。

两者的差别请参见“PLC-CPU 的基础数据 (页 947)”章节。

其存储方案与 SIMATIC S7-300 PLC 有部分区别，因此不支持特定功能（例如将模块保存至存储卡、将项目保存至存储卡）。

说明

对于 SIMATIC S7-300 PLC 和 SINUMERIK 中集成的 PLC，在编程设备上的操作触发“PLC 停止时”，掉电并重新上电后 PLC 不会自动启动。 由于安全原因，PLC 保持在停止状态，并带有相应的诊断记录。 只能通过编程设备上的“执行重启”操作或运行方式开关“Stop”>“Run”（热启动）来启动 PLC。

14.8.3 集成 PLC 的接口

物理接口

使用 SINUMERIK 840D sl 时，NCU 中集成的 PLC 支持直接通过一个 Dual-Port-RAM（双端口 RAM）在 NC 和 PLC 间进行信号交换。

与操作面板的数据交换

与操作面板（例如 TCU/OP）的数据交换可通过以太网或 PROFIBUS 进行。通过以太网连接时，通讯通过集成的通讯处理器（CP 840D sl）进行。

与机床控制面板（MCP）和手持操作设备（HHU）的数据交换可通过 MPI、PROFIBUS 或以太网进行。

编程设备优先通过以太网或 MPI（Multi-Point-Interface，多点接口）直接连接至 PLC。

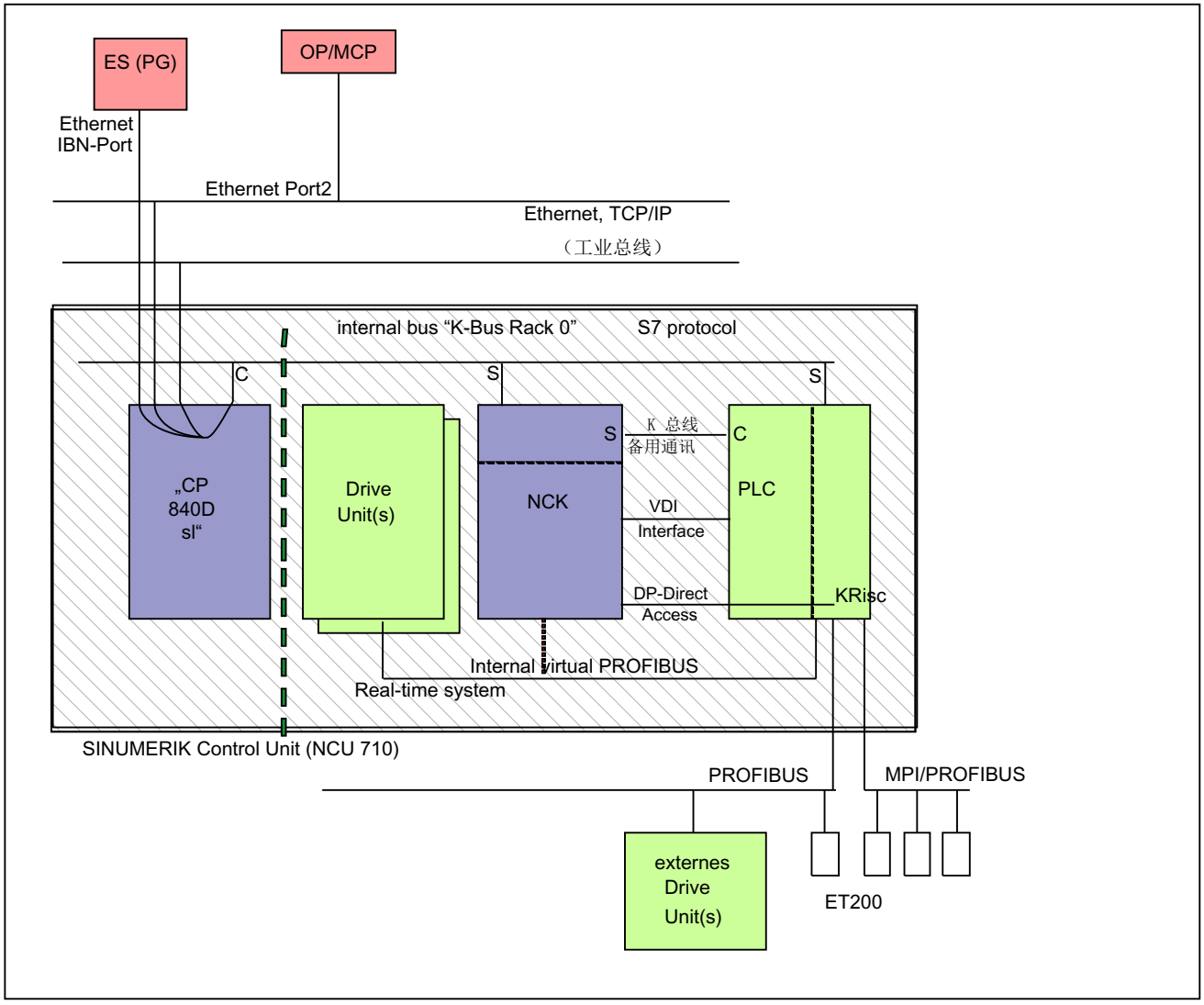


图 14-1　　SINUMERIK 840D sl 上的 NC-PLC 连接（集成 PLC）

接口：NC / PLC

NC 和 PLC 之间的数据交换由 PLC 通过基本程序组织。由 NC 保存在 NC/PLC 接口中的**状态信息**（例如“程序运行中”）通过基本程序在循环开始处（OB1）复制到数据块，之后便可在用户程序中访问该信息（用户接口）。由用户输入到接口 DB 的、发送至 NC 的**控制信号**（例如“NC 启动”）同样在循环开始处传输至针对 NC 的内部 DPR。

基于工件程序传输至 PLC 的**辅助功能**首先会由基本程序在报警控制下进行分析，之后在 OB1 的开始处传输至用户接口。若相关程序段中包含了需要中断 NC 处理的辅助功能（例如用于换刀的 M06），那么基本程序的 NC 程序段解码会先停止一个 PLC 循环的时间。之后用户可通过接口信号“读取禁止”一直停止解码，直至换刀完成。与之相反，若相关 NC 程序段中只包含不需要中断解码的辅助功能（例如 M08“冷却剂 ON”），那么这些“快

速”辅助功能的传输会直接在 OB40 中应答，这样一来向 PLC 的传输只会对解码造成较小影响。

由 NC 传输的 **G 指令**同样会在报警控制下接受检测和应答，不过之后会直接传输至用户接口。若在 PLC 程序中的多个位置检测一个 G 指令，那么一个 PLC 循环内 G 指令的信息可能会出现差异。

对于由 PLC 触发和参数设置的 **NC 动作**（例如运行并行轴），触发和设置不通过接口 DB 进行，而是通过 FC 和 FB 进行。从属于 NC 动作的 FC 会和基本程序一起提供。所需的 FC 和 FB 必须由用户载入，并在机床制造商的 PLC 程序（机床程序）中调用。基于基本功能和扩展功能分类的 FC、FB 和 DB 模块的概览请参见“调试 PLC 程序”章节。

接口：HMI / PLC

HMI / PLC 数据交换通过集成 CP 进行，其中 HMI 总是作为主动方（客户端），PLC 则总是作为被动方（服务器）。由 HMI 请求或传输的数据会通过 PLC 操作系统从 HMI / PLC 接口区域读取，或记录至该区域（时间点：循环控制点）。从 PLC 用户的角度看来，数据的特性与 I/O 信号相似。

接口：MCP / PLC 或 HHU / PLC（连接：以太网）

MCP / PLC、HHU (HT 2) / PLC 数据交换通过集成 CP 进行。MCP 信号/ HHU 信号由 CP 保存至 PLC 的内部 DPR（Dual Port RAM），或从 DPR 读取。PLC 侧通过基本程序实现与用户接口的数据交换。操作数区域（例如 I/O 区域）和起始地址通过基本程序的参数（FB1、DB7）确定。

接口：MCP / PLC（连接：PROFIBUS）

MCP / PLC 数据交换通过 PLC 的 PROFIBUS 进行。MCP 的 I/O 地址必须通过 STEP 7 中的硬件配置设置到 PLC 的过程映射区域。Pointer 变量 MCP*In、MCP*Out 必须设置到相同地址。MCP*BusAdr 中必须输入设置的 DP 从站编号。

接口：HHU / PLC（连接：MPI）

HHU / PLC 数据通讯通过 PLC 的 MPI 接口进行。为此会使用“通过全局数据（GD）通讯”（参见 STEP 7 用户手册）。与用户接口的信号交换由 PLC 操作系统实现。通过 STEP 7 配置工具“通讯配置”除了可定义 GD 参数，还可定义操作数区域（例如 I/O 区域）和其起始地址。

14.8.4 PLC 的诊断缓存

PLC 的诊断缓存（可通过 STEP 7 读取）中记录的是 PLC 操作系统的诊断信息。

14.9 接口结构

接口数据块

NC 和 PLC 间的信号量很大，因此需使用接口数据块中的映射。从 PLC 程序的角度看来，其为全局数据块。基本程序会在系统启动时基于当前的 NC 机床数据（通道数量、轴等）生成这些数据块。此方案的优点是：只占用当前机床配置所需的 PLC-RAM。

14.9.1 PLC/NCK 接口

简介

PLC/NC 接口包含数据接口和功能接口。数据接口包含有状态信号、控制信号、辅助功能和 G 指令等，PLC 至 NC 的任务传输则通过功能接口实现。

数据接口

数据接口划分为下列组别：

- NC 专用信号
- 运行方式组专用信号
- 通道专用信号
- 轴/主轴/驱动专用信号

功能接口

功能接口通过 FB 和 FC 构成。下图显示了 PLC 和 NC 间接口的一般结构。

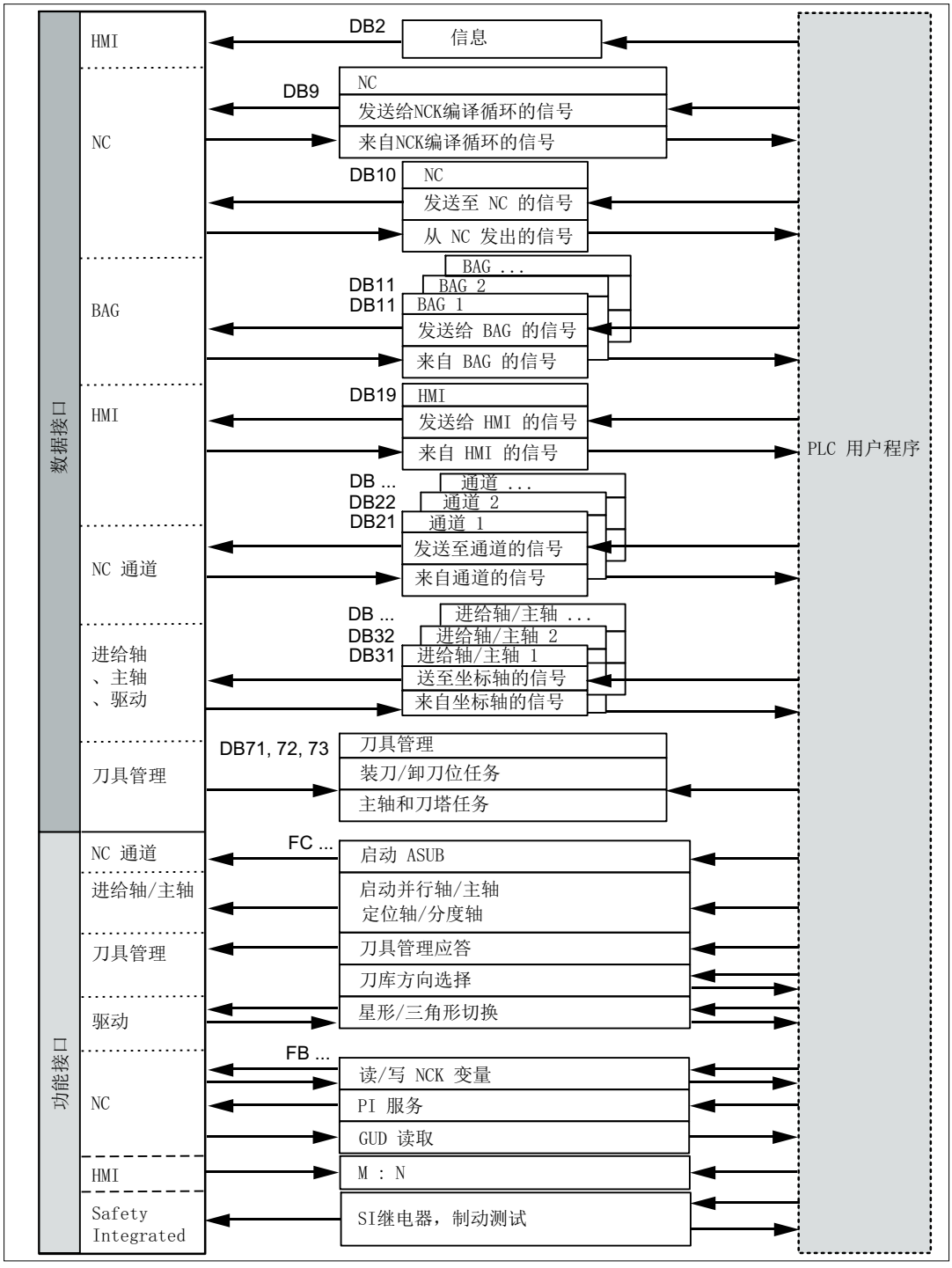


图 14-2　　用户接口 PLC / NC

编译循环信号

除了 PLC 和 NC 间的默认信号外，根据需要系统还会生成一个用于编译循环的接口数据块（DB9）。基于编译循环的相应信号在 OB1 的开始处循环传输。传输通过基本程序以从低到高的地址顺序进行。首先会从 PLC 向 NC 传输信号，之后再从 NC 向 PLC 传输信号。用户必须在 NC 和 PLC 间执行必要的同步（例如通过信号量）。信号传输在 NC 和 PLC 间异步进行。此时，通过 PLC 进行的已开始的 NC 数据传输可能会中断。这样一来数据有时会不统一。

PLC / NC 信号

从 PLC 发送至 NC 的信号包括：

- 用于控制 NC 数字量和模拟量 I/O 信号的信号
- 钥匙开关信号和急停信号

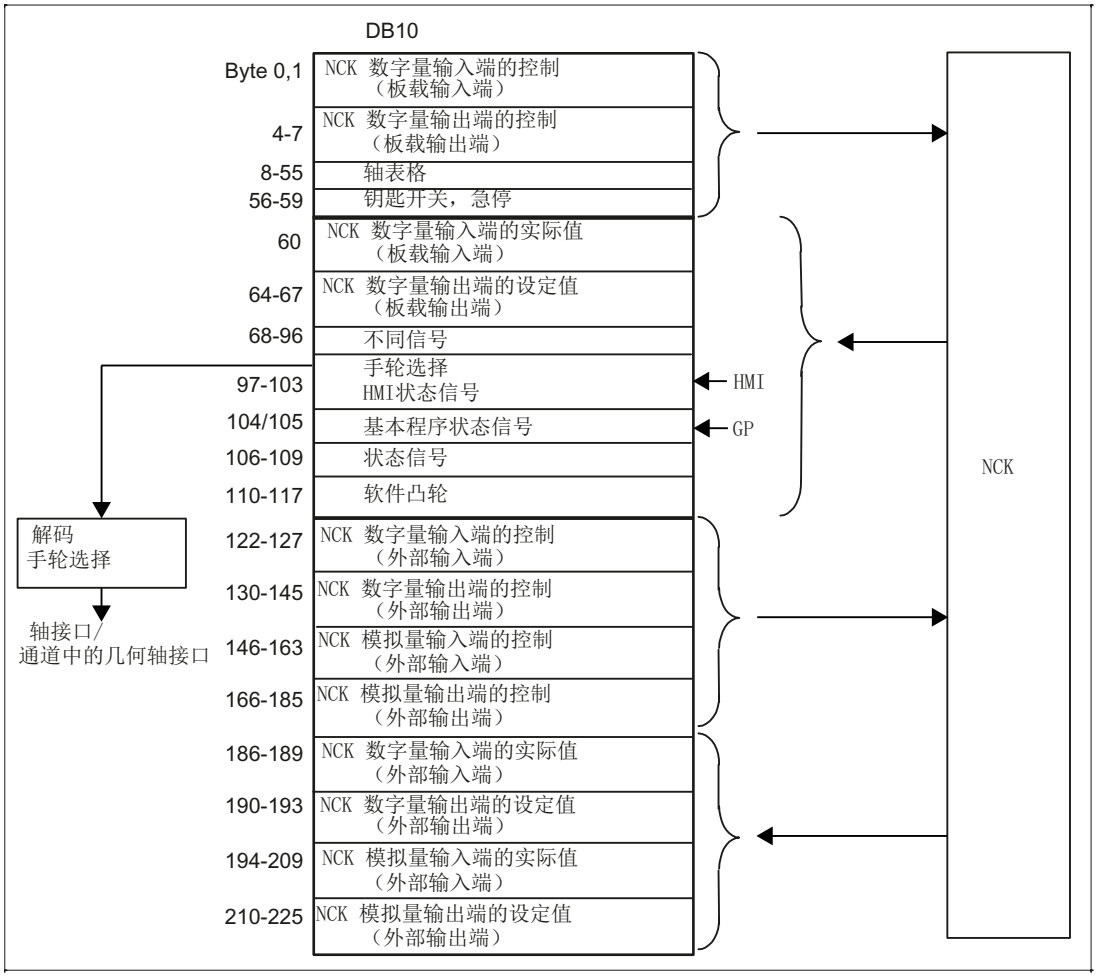


图 14-3　　PLC / NC 接口

NC / PLC 信号

从 NC 发送至 PLC 的信号包括：

- NC 的数字量和模拟量 I/O 信号的实际值
- NC 的就绪信号和状态信号

此外此处也可涵盖 HMI 的手轮选择信号和通道状态信号。

用于手轮选择的信号通过基本程序解码，并记录至机床专用或通道专用接口。

NC 的数字量/模拟量 I/O

对于 NC 的数字量和模拟量输入输出，须注意以下几点：

输入：

- NC 的所有输入信号或输入值也会传输至 PLC。
- 可通过 PLC 抑制向 NC 零件程序的信号传输。可通过 PLC 设定一个信号或一个值来作为替代。
- 若 NC 侧无对应该通道的硬件，那么还可将信号或值从 PLC 传输至 NC。

输出：

- 所有待输出的信号或值也会传输至 PLC。
- 若 NC 侧无对应该通道的硬件，那么也可将信号或值从 NC 传输至 PLC。
- 由 NC 设定的值可通过 PLC 改写。
- 也可由 PLC 通过 NC I/O 将信号或值直接输出。

说明

配置数字量和模拟量 NC I/O 时，必须参考以下文档中的信息：
文档：
功能手册之扩展功能分册；数字量和模拟量 NC I/O（A4）

PLC / BAG 信号

通过机床控制面板或 HMI 设定的运行方式信号会传输至 NC 的运行方式组（BAG）。这适用于所有 NC 通道。在 NC 中可选择定义多个 BAG。

BAG 会将其当前状态发送至 PLC。

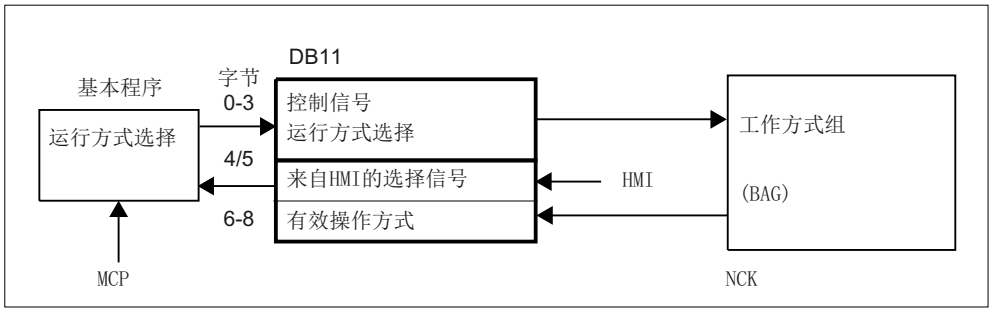


图 14-4　　PLC / BAG 接口

“PLC / NC 通道”信号

此接口对应下列信号组：

- 控制信号/状态信号
- 辅助/G 指令
- 刀具管理信号
- NC 功能

控制信号/状态信号在 OB1 的开始处循环传输。对于通过 HMI 记录至通道专用接口的信号（HMI 信号通过 PLC 的操作系统记录），若其不是通过 MCP，而是通过 HMI 操作面板设定，那么这些信号也会在该时间点传输。

辅助指令和 G 指令通过两种方式记录至接口数据块。这些信号首先会和修改信号一起被记录。

- **M 信号** M00 - M99（由 NC 以扩展地址 0 传输）会被解码，且对应循环时间的接口位会置位。
- 对于 **G 指令**，只有通过机床数据选择的功能组会被记录至接口数据块。
- **S 值**还会与对应的 M 信号（M03、M04、M05）一同记录至主轴专用接口。同样地，轴专用进给率也会记录至相应的轴专用接口。

NC 中激活了**刀具管理（刀库管理）**时，主轴或刀塔的占用以及装刀/卸刀位置均会记录至专用的接口数据块（DB71 - 73）。

NC 功能通过调用 PLC 功能触发和设置。
可供调用的功能例如包括：

- 直线轴或回转轴的定位
- 分度轴的定位
- 启动准备的异步子程序（ASUB）

- 读/写 NC 变量
- 更新刀库运动和刀具运动

上述功能中有一部分会在专属的功能手册中介绍。

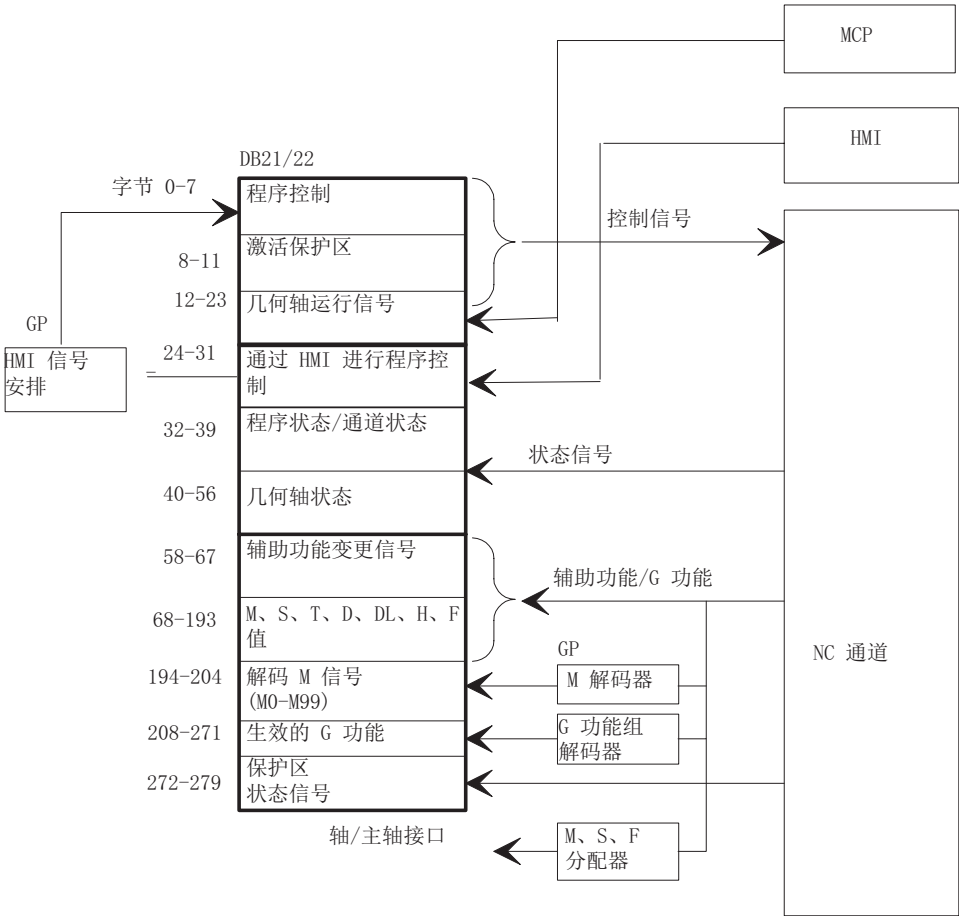


图 14-5 “PLC / NC 通道”接口

“PLC / 轴、主轴、驱动”信号

轴/主轴专用信号分为下列组别：

- 轴/主轴共用信号
- 轴信号
- 主轴信号
- 驱动信号

这些信号会在 OB1 的开始处循环传输，以下特例情形除外。

特例包括：

- 轴向 F 值
- M 值
- S 值

若**轴向 F 值**在 NC 程序执行过程中传输至 PLC，则会通过基本程序的 M、S、F 分配器记录。

对于 **M 值和 S 值**，若其中一个值或两个值被执行，则同样通过基本程序的 M、S、F 分配器记录。

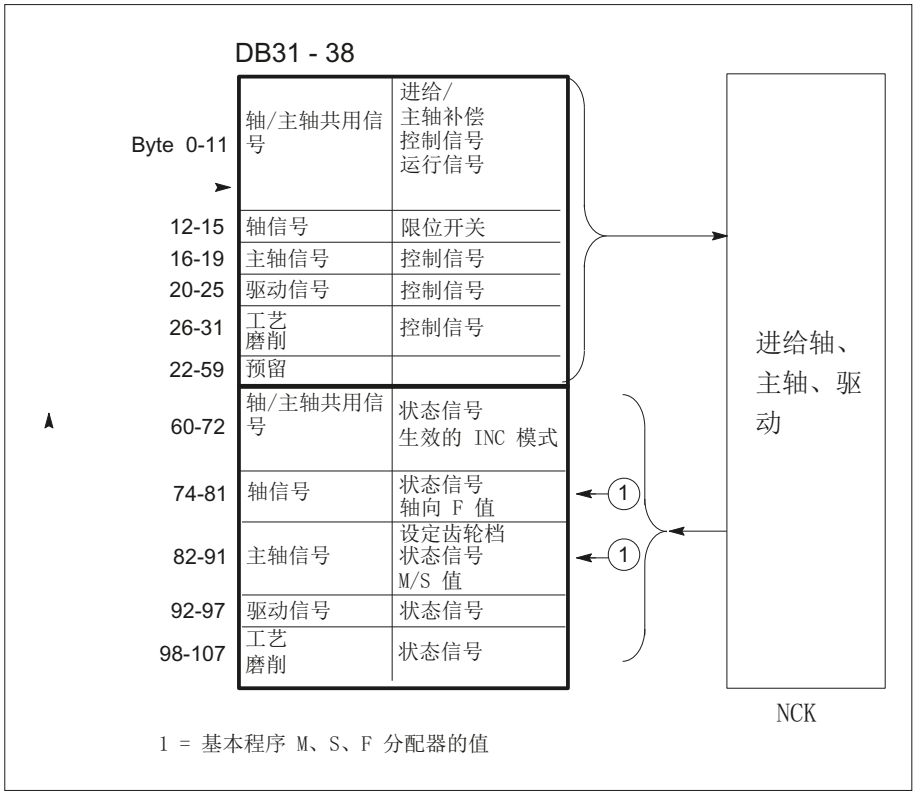


图 14-6　“PLC / 轴、主轴、驱动”接口

14.9.2 PLC/HMI 接口

简介

PLC/HMI 接口涵盖的功能范畴如下：

- 控制信号
- 机床操作
- PLC 消息
- PLC 状态显示

控制信号

控制信号指的是通过机床控制面板设定，且必须被 HMI 考虑在内的信号。 此类信号例如包括：在 MCS 或 WCS 中显示实际值、按键禁用等。这些信号通过一个专属的接口数据块（DB19）与 HMI 进行交换。

机床操作

所有会引起机床上的动作的操作均由 PLC 控制。这些操作通常通过机床控制面板（MCP）进行。不过也可选择通过 HMI 来执行部分操作，例如运行方式切换。

来自 HMI 的操作信号会通过 PLC 操作系统直接记录至接口数据块。默认情形下，基本程序会对这些操作信号进行安排，使得能够选择通过 HMI **或** MCP 执行操作（前提条件是 MCP 也能实现同样的操作）。必要时用户可通过 FB1 的“MMCToIF”参数取消 HMI 的操作权。

PLC 消息

消息功能的基础是 AS 300 的操作系统中集成的系统诊断功能，其具有以下属性：

- PLC 操作系统将所有重要的系统状态及状态过渡记录在一个**诊断状态列表**中。通讯事件和（I/O）模块诊断数据（针对带诊断功能的模块）也会被记录。
- 此外，引起系统停止的诊断事件会按照发生的时间顺序配以时间戳记录在**诊断缓存**（环形缓存）中。
- 在操作系统/监控系统（OP 或 HMI）通报“接收就绪”状态（消息通讯）的情形下，诊断缓存中记录的事件会通过总线系统自动发送至这些系统。对通报节点进行传输是 PLC 操作系统的一项功能。消息的接收和解释由 HMI 软件确保。

- 也可由 PLC 用户程序通过 SFC（System-Function-Call，系统功能调用）将消息记录至诊断缓存或 ALARM_S/ ALARM_SQ 缓存。
- 事件记录在报警缓存中。
  对应的文本必须保留在 OP 或 HMI 中。

系统还提供一个用于消息采集的 FC （FC10）以配合基本程序。此 FC 能够采集需要报告的事件（划分为信号组）并通过报警缓存将其报告给 HMI。

消息采集的结构在“PLC 事件的采集和报告”一图中显示。其具有以下特性：

- 涉及 NC/PLC 接口的事件的位域与用户消息的位域在 DB2 中汇总。
- 位域会由 FC10 多次分析。
  - **分析 1；** 采集**汇总信号**
    每个信号组中至少有一个位信号置“1”时，便会生成一个汇总信号。此信号通常被链接至对应的 NC/PLC 接口禁用信号（针对带诊断功能的模块）。汇总信号会被周期性完整采集。
  - **分析 2；** 采集**报警消息**
    组中的哪些信号从“0”切换至“1”时生成报警消息为固定设定。
  - **分析 3；** 采集**运行消息**
    组中的哪些信号产生运行消息为固定设定。
- 用户位域（user area）范围的缺省设置为 10 个区域，每个区域 8 字节；不过可通过 FB1 上的基本程序参数基于区域的数量根据机床制造商的需求进行调整。

应答方案

对故障消息和运行消息将采用以下应答方案：

运行消息用于将机床的一般运行状态作为信息向操作人员显示。因此无需为其配置应答信号。事件的出现或消失都会被采集，并记录至诊断状态列表。HMI 会根据“运行消息出现”和“运行消息消失”标识显示当前存在的运行消息。

报警消息显示的是机床上的故障状态，这些状态通常会导致机床停止。若“同时”存在多个故障，那么确定故障出现的顺序对找出故障尤为重要。这一方面通过诊断缓存中的记录顺序显示，另一方面通过每条记录包含的时间戳显示。

若故障原因消失，则须进行用户应答（例如通过 MCP 上的用户键），在此之后对应的报警消息才会被删除。作为对此信号的响应，“消息采集”FC 会检查报告的故障中哪一个已消失，并通过“报警消失”标识将其记录至诊断缓存。这样一来 HMI 便可生成一个包含当前存在的报警消息的画面。对于尚未处理的消息，其发生时间将保留（与新查询相反）。

STEP 7

在 SIMATIC 管理器中可通过菜单项“目标系统”>“CPU 消息”启动一个工具。借助此工具能够以编号显示报警和消息。为此必须激活“报警”制表器并在画面上方的“A”下打钩。

用户程序

在循环程序部分中，用户 PLC 程序只须调用采用对应设置的基本程序块 FC10，以及设置或复位 DB2 中的位域。所有其他操作由基本程序和 HMI 进行。

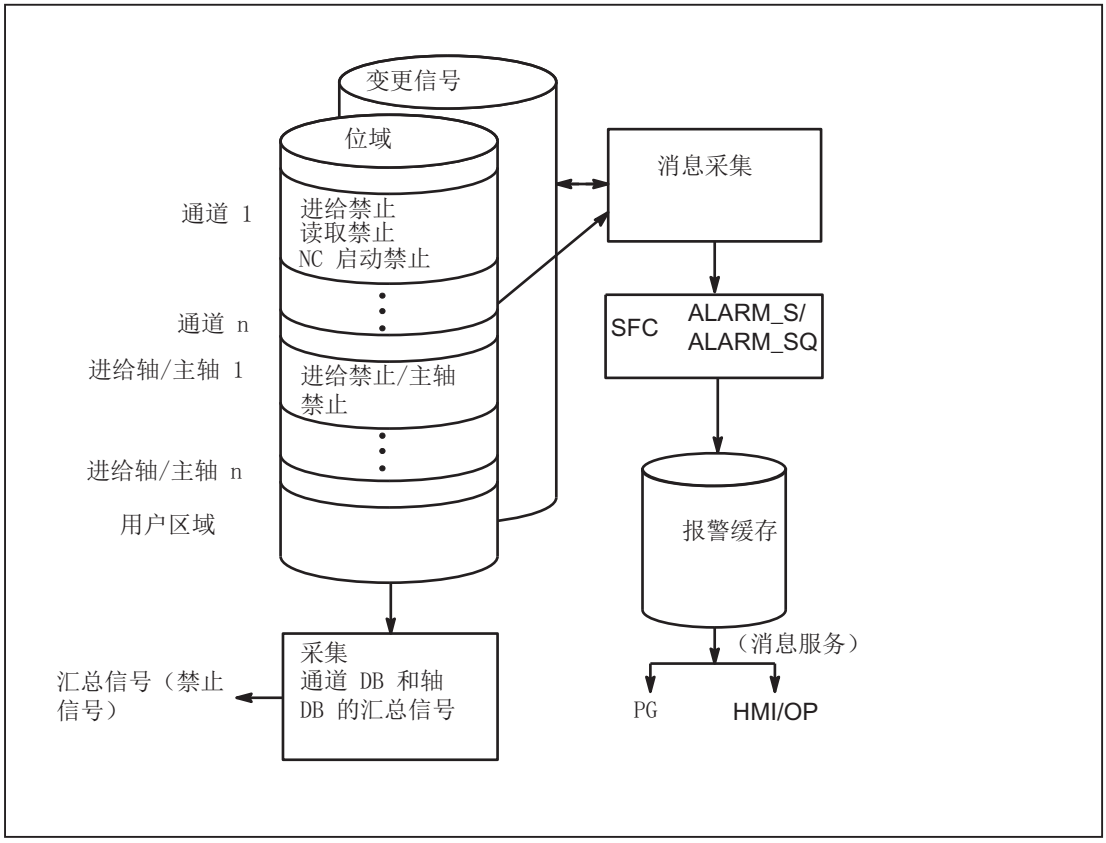


图 14-7　　PLC 事件的采集和报告

通过 FC10 扩展 PLC 报警

可通过 FB1 参数“ExtendAlMsg”选择 PLC 报警机制。

“ExtendAlMsg:= FALSE”时，之前的 FC10 设定和 DB2 作为位域数据块的设定生效。已知的对通道数量和轴数量的限制适用。

“ExtendAlMsg:= TRUE”时则是 FC10 的扩展生效。DB2 和 DB3 保持当前的创建状态。用户必须在 DB2 中对位进行设置或复位。对消息和报警的参数设置，以及对用户报警的第 2 个十进制位的数值设置包含在 DB5 中。

扩展为：

- 支持 10 通道、31 轴、64 用户区域（用户区域的数量须在 FB1 参数 “MsgUser” 上输入）。
- 提供用于进给停止、读取禁止等功能的区域，这些区域中不会触发消息。此区域中的信息会根据 FC10 参数“ToUserIF”与对应的消息位一起作为汇总信号设置至 DB21、DB31 中的接口。这样便可免除一直以来对这些信号的繁冗操作。
- 对于用户区域 0，报警/消息除编号外还将得到 16 位整数附加值（报警文本中的 %Z 参数）。用户必须在设置报警位的同时将这 16 位整数值在 DB2 中写入数组变量 ZInfo0。用户区域 0 中的每个位均对应一个整数值，参见基本程序中的 UDT1002。
- 可在消息编号的第 2 个十进制位中设置用户消息，范围为 0 至 9。第 2 个十进制位的显示值必须由用户写入 DB5 中的数组变量 UserDek2No。可为每个用户区域定义一个数字，参见基本程序中的 DB5。
  为第 2 个十进制位预设的缺省值为 0。

DB2 的结构可在 UDT1002（基本程序）中识别。对于新的报警功能，应在符号表中将 UDT1002 以符号指定给 DB2。

DB2 的开头为不会生成消息的信号的位域。之后是一个大小为 64 整数的数组，用于用户区域 0 的附加信息。

再往后的区域可触发报警/消息（参见参数手册）。此区域被扩展为 10 通道、31 轴。

从用户程序到新报警的简单转换

基本程序的资源容器中有“udt2_for_Convert.awl”这一文件，其包含了来自 UDT1002 的以下结构单元：

- ChanA 作为 1 ... 8 数组
- AxisA 作为 1 ... 18 数组
- UserA 作为 0 ... 63 数组

此 UDT2 必须通过 LAD/FBD/STL 编辑器编译。此 UDT2 必须在符号表中指定给 DB2。

对于和 DB2 有指定关系的模块，应从其创建源。或者可从所有模块创建源。UT1002 仍须在符号表中指定给 DB2。之后须重新编译这些源。

这样一来所有报警都被指定给 DB2 中新的数据区域，只须将 FB1 参数“ExtendAlMsg”设置为 TRUE。

上电复位后，报警特性与之前相同。

14.9.3 PLC/MCP/HHU 接口

简介

机床控制面板（MCP）和手持操作设备（HHU）可采用多种连接方式。这部分源于 MCP 和 HHU 的历史。本章节将介绍通过以太网组件进行的连接。

在 SINUMERIK 840D sl 上，机床控制面板（MCP）和手持操作设备（HHU）通过以太网总线连接，同时该总线还将 TCU 与 NCU 相连。此方案的优点是只需敷设连接操作单元的总线电缆。

SINUMERIK 840D sl 的拓扑结构

在 840D sl 中，机床控制面板和手持操作设备连接至 CP 840D sl 的以太网总线。若需为用户操作面板连接其他按键和显示设备，可使用一个额外的键盘接口（无操作单元的 MCP）。每个键盘接口可通过扁平带状电缆连接 64 个按键、开关以及 64 个显示单元。

MCP 发出的信号通过 CP-840D sl 集成的以太网复制到 PLC 的 DPR（Dual-Port-RAM）。PLC 基本程序会将接收到的信号记录至 FB1 上配置的输入映像。NC 相关信号通常由基本程序分配至 NC/PLC 接口。用户可根据需要修改信号。

从 PLC 发送至 MCP 的信号（显示）则以相反的方向传输。

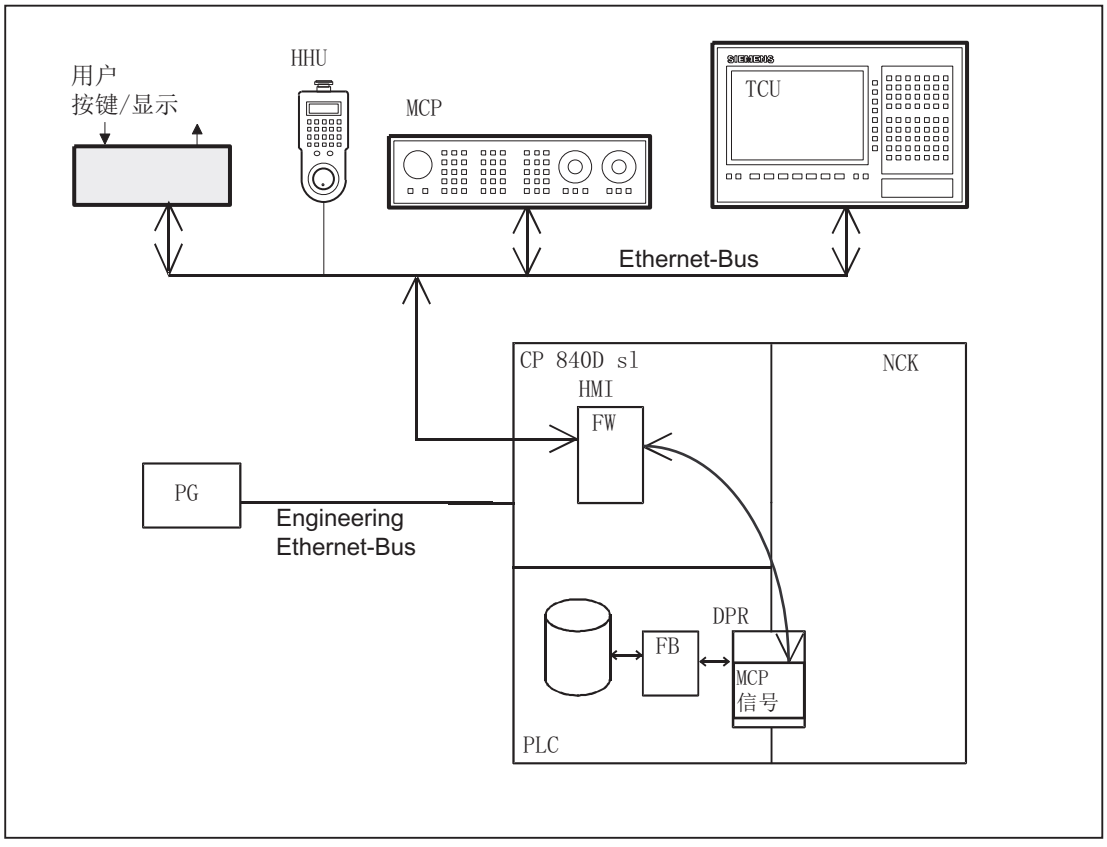


图 14-8　　840D sl 中机床控制面板的连接

总线地址

对于以太网组件，MAC 地址、IP 地址或逻辑名称对通讯至关重要。逻辑名称向 MAC 地址或 IP 地址的转换通过控制系统的系统程序进行。PLC 中会使用逻辑名称的数字部分来进行通讯。该数字部分由用户通过“MCPxBusAdr”参数传输至 FB1。

MCP 或 HHU 的逻辑名称总是以“DIP”开头。之后的编号对应 MCP 组件的开关位置（例如 DIP192、DIP17）。

PLC 中的 MCP 接口

缺省设置下，机床控制面板的信号通过 I/O 接口传输至 PLC 区域。此时须区分 NC 专用和机床专用信号。NC 专用按键信号默认从 FC19（或 FC24、FC25、FC26，依据 MCP 规格）分配至相应的 BAG 专用、NC 专用、轴专用和主轴专用接口。将对应的状态信号传输至 MCP 接口时，情况则相反。为此必须在**用户程序**中调用 FC19 或上面提到的其他模块。

用于触发各种机床功能的用户按键必须直接由用户程序分析。对应的状态信号同样由用户程序传输至 LED 的输出区域。

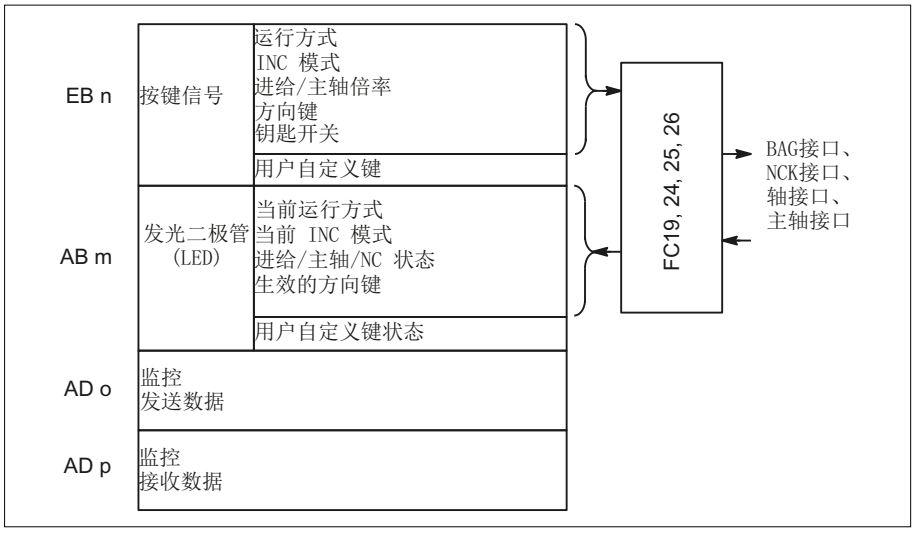


图 14-9　　机床控制面板的进出接口

14.10　基本程序的结构和功能

简介

PLC 程序采用模块化结构。 组织块（OB）构成操作系统与基本程序及用户程序之间的结构。

- 重启（热启动）和同步（OB100）
- 循环运行（OB1）
- 过程报警 (OB40)
- 异步故障：诊断报警（OB82）、模块故障（OB86）

用户必须在组织块（OB）中编写对基本程序和用户程序的功能块的调用。

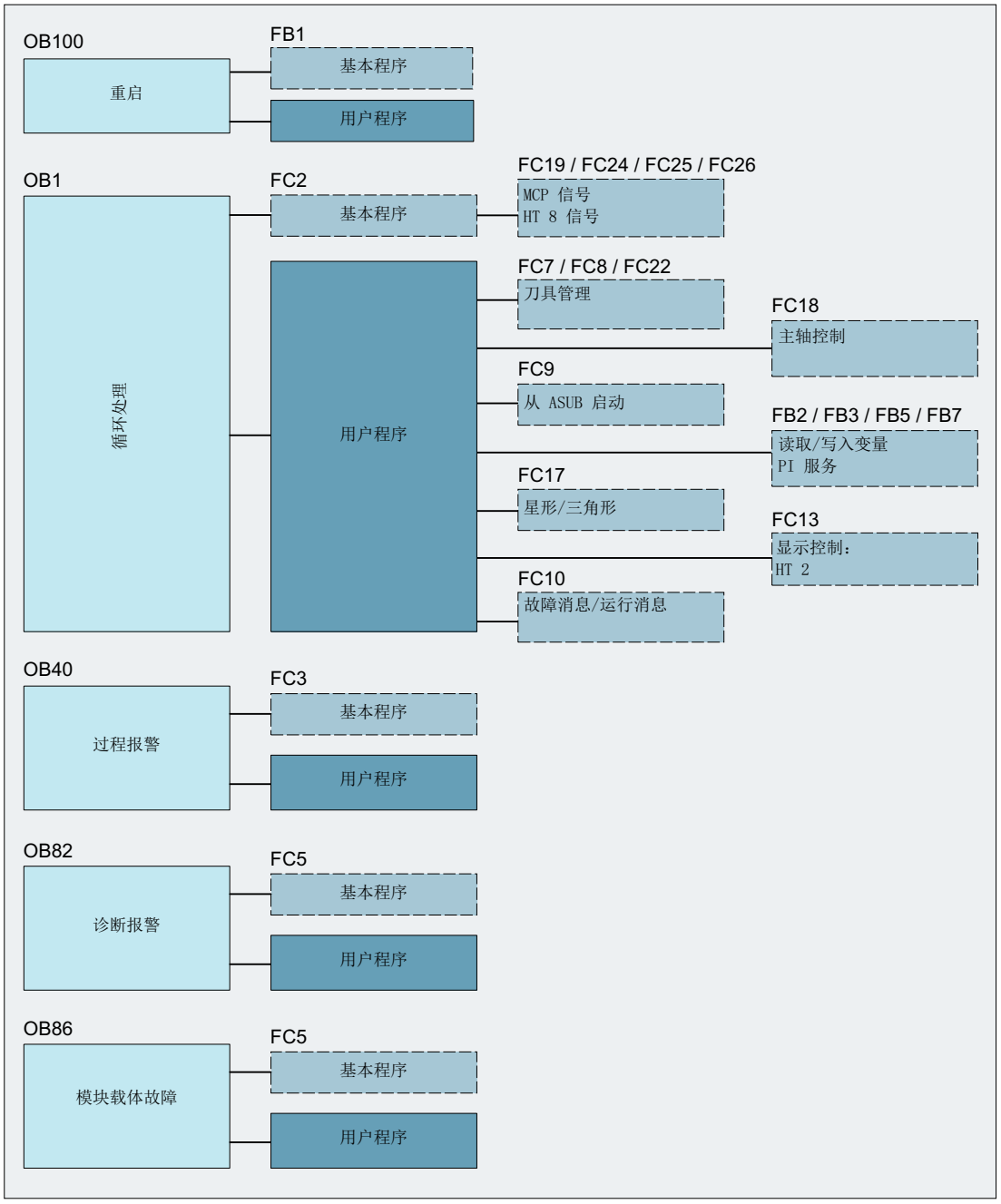


图 14-10　基本程序结构（原理图）

14.10.1 NCK-PLC 启动和同步

载入基本程序

基本程序必须通过 S7 工具在 PLC 停止状态下载入。这样可确保下一次启动时基本程序的模块能够正确地初始化。若不遵循此操作，则可能导致未定义的 PLC 状态（例如所有 PLC-LED 均闪烁）。

启动

启动期间会进行 NC 和 PLC 之间的同步。此时会检查系统数据块和用户数据块的完整性，以及检查最重要的基本程序参数是否合理。若存在错误，基本程序会发出报警（HMI 上可见），并将 PLC 切换至停止状态。

系统初始化完成后，操作系统会完整运行组织块 OB100，并总是在 OB1 的开始处启动循环运行。

同步

启动时 PLC 会与 HMI、NC 及 CP 进行同步。

生命符号

在按照规定启动和首个完整的 OB1 循环（初始设置循环）后，PLC 和 NC 会持续交换生命符号。若 NC 生命符号缺失，那么 PLC/NC 接口将初始化，且 DB10 中的“NC-CPU 就绪”信号会被设置为 FALSE。

14.10.2 循环运行（OB1）

简介

NC/PLC 接口只在循环运行中完整处理。从时间角度而言，基本程序在执行用户程序之前运行。为了保持较低的基本程序运行时间，只有控制信号/状态信号会被循环传输，辅助功能和 G 指令传输只会应请求处理。

下列功能会在基本程序的循环部分执行：

- 控制信号/状态信号的传输
- 辅助功能的分配
- M 解码（M00 - M99）

- M、S、F 分配
- 通过 NC 传输 MCP 信号
- 用户故障消息和运行消息的采集和预处理

控制信号/状态信号

控制信号和状态信号的共同特性是：其均为位域。基本程序会在 OB1 的开始处对其进行更新。

这些信号划分为下列组别：

- 通用信号
- BAG 专用信号（例如运行方式）
- 通道专用信号（例如程序控制和进给控制）
- 轴和主轴专用信号（例如进给禁止）

辅助指令和 G 指令

辅助指令和 G 指令具有以下特性：

- 其均程序段同步传输至 PLC（以零件程序段为基准）。
- 在应答控制下进行传输。
- 应答时间直接影响包含须应答辅助功能的 NC 程序段的执行时间。

取值范围请见下表：

功能	结构		取值范围		数据类型	
	第 1 个值	第 2 个值	第 1 个值	第 2 个值	第 1 个值	第 2 个值
G 指令		G 指令		255[1)]		字节
M 字	M 组	M 字	99	99.999.999	Word	DWord
S 字	主轴编号	S 字	6	浮点 [2)]	Word	DWord
T 字	刀库号	T 字	99	65535	Word	Word
D 字	-	D 字	99	255	字节	字节

功能	结构		取值范围		数据类型	
H 字	H 组	H 字	99	浮点	Word	DWord
F 字	轴编号	F 字	18	浮点	Word	DWord

1) 相对编号，针对每个 G 功能组传输

2) 对应 STEP 7 格式（24 位尾数，8 位指数）

由 NC 传输的 M 值、S 值、T 值、H 值、D 值、F 值和对应的修改信号一同通过辅助功能/G 指令分配器输出至 **通道 DB** 接口（参见参数手册）。此时辅助功能的两个值会分别传输至对应的数据字。其对应的修改信号会在一整个 PLC 循环内置 1。复位修改信号时，系统会将应答发送至 NC。快速辅助功能的应答在基本程序识别出辅助功能后由基本程序立即执行。

除了分配辅助功能和 G 指令外，对于所选的信号还会按照下面的介绍进行处理 。

M 解码器

M 功能可用于传输切换指令和固定点值。对于标准 M 功能（范围 M00 - M99），解码的动态信号会被输出至 **通道 DB** 接口（信号持续时间 = 1 循环周期）。

G 功能组解码器

对于由 NC 传输的 G 指令，其对应的 G 指令组会被解码，生效的 G 编号会被记录至通道 DB 的接口字节，即所有生效的 G 指令均记录至通道 DB。记录的 G 指令在 NC 程序结束或程序终止后仍保留。

说明

系统启动时会向所有 G 功能组字节输入“0”值。

M、S、F 分配器

主轴专用的 M 字 M(1...6)=[3,4,5]，以及用于轴向进给的 S 字和 F 字通过 M 分配器、S 分配器、F 分配器记录至对应的**主轴和轴 DB**。分配的标准是针对 M 字、S 字和轴向 F 字传输至 PLC 的扩展地址。

MCP 信号传输

根据总线地址，MCP 信号直接传输至 PLC，或借助基本程序通过内部方式间接传输至设置的 I/O 区域。

用户消息

用户故障消息和运行消息通过基本程序的一个 FC 采集和预处理。

14.10.3 时间报警的处理（OB35）

用户须编写 **OB35** 用于处理时间报警。 OB 35 的时间区间的缺省设置为 100 ms。 也可通过 STEP 7 工具“HW Config”选择另一个时间区间。 不过 OB35 的时间设置不得小于 3 ms，否则会导致 PLC-CPU 停止。 在 HMI 启动时读取其系统状态列表（SZL）会造成停止。 该读取操作会使运行级控制（AES）阻塞约 2 ms。 因此在时间设置较小时无法正确处理 OB35。

14.10.4 过程报警处理 (OB40)

过程报警 **OB40**（中断）例如可由相应配置的 I/O 或特定 NC 功能触发。 中断可能由各种原因造成，因此 PLC 用户程序必须先在 OB40 中解释中断原因。 中断原因包含在 OB40 的本地数据中。

文档：
SIMATIC STEP 7 说明，或 STEP 7 的在线帮助

14.10.5 诊断报警，模块故障处理(OB82, OB86)

简介

I/O 模块的诊断报警或模块故障会导致基本程序模块 OB82 或 OB86 的调用。由此调用基本程序模块 FC5 “诊断报警和模块故障” (页 1145)。

总线诊断

PROFIBUS 接口 MPI/DP、DP1 或 PROFINET 接口 PN 上的 DP 从站模块的状态通过基本程序作为相应总线系统的汇总信号“从站正常”发送至用户程序：

- DB10.DBX92.0 == 1 (MPI/DP Bus:Slaves OK)
- DB10.DBX92.1 == 1 (DP1-Bus:Slaves OK)
- DB10.DBX92.2 == 1 (PN-Bus:Slaves OK)

汇总信号从相应总线系统的 LED 状态推导得出（系统状态列表 SZL 0x174）。

输出报警

若在相应总线系统上识别出从站故障或失灵，则会显示报警：

- 报警 400551（MPI/DP 总线）
- 报警 400552（DP1 总线）
- 报警 400553（PN 总线）

故障消除时该报警会被自动重新删除。

抑制报警输出 400551、400552、400553

通过置位下列信号，为相应总线系统抑制报警显示：

- DB10.DBX92.4 = 1（抑制报警 400551）
- DB10.DBX92.5 = 1（抑制报警 400552）
- DB10.DBX92.6 = 1（抑制报警 400553）

在信号置位后，报警输出立即被抑制。在出现故障前或故障期间已存在。

通过将信号复位，在故障情形下显示相应的报警。

在控制系统的启动中，系统会将这些信号复位。

说明

通过置位信号，整个总线支路的故障监控都会被取消！

14.10.6 NCK 故障时的特性

简介

循环运行期间，PLC 基本程序会通过查询生命符号对 NC 就绪状态进行持续监控。若 NC 不再响应，那么 NC/PLC 接口会被初始化，且 **NC 发出信号区域**中的 **NC-CPU 就绪（DB10.DBX104.7）**信号会被复位。此外，从 NC 传输至 PLC 和从 PLC 传输至 NC 的信号也会被设置为初始状态。

PLC 自身保持生效，从而可继续控制机床功能。但此时用户程序将获得将机床切换至安全状态的权限。

信号 NC → PLC

从 NC 传输至 PLC 的信号划分为下列组别：

- NC、通道、轴和主轴的状态信号
- 辅助功能的修改信号
- 辅助功能的值
- G 指令的值

状态信号：

NC、通道、轴和主轴的状态信号会被复位。

辅助功能的修改信号：

辅助功能的修改信号同样会被复位。

辅助功能的值：

辅助功能的值保留，从而可恢复最近一次由 NC 触发的功能。

G 指令的值：

G 指令的值会被复位（即分别输入 0 值）。

信号 PLC → NC

从 PLC 传输至 NC 的信号分为控制信号及通过 FC 传输至 NC 的任务。

控制信号：

从 PLC 发送至 NC 的控制信号会被冻结；通过 PLC 基本程序进行的循环更新中断。

从 PLC 发送至 NC 的任务：

用于向 NC 传输任务的 FB 和 FC 不允许通过 PLC 用户程序处理，因为有时会出现错误的反馈信息。在控制系统启动时必须在用户程序中等待，直至信号 **NC-CPU 就绪**置位后才能激活任务（例如：读取 NC 数据）。

14.10.7 由用户程序调用的基本程序功能

简介

除了在 OB1、OB40 和 OB100 的开始处调用的基本程序模块外，也有一些功能必须在用户程序中的适宜位置调用和参数设置。

通过这些功能例如可将下列任务从 PLC 传输至 NC：

- 运行并行轴（FC18）
- 启动异步子程序 ASUB（FC9）
- 选择 NC 程序（FB4）
- 控制主轴（FC18）
- 读/写变量(FB2, FB3)

说明

检查和诊断 PLC 基本程序的功能调用

对于通过触发信号控制（例如通过参数 Req、Start）和提供执行应答作为输出参数（例如通过参数 Done、NDR、Error）的 PLC 基本程序的功能调用（FB 或 FC），为了简化对其的检查和诊断，可采取以下措施。

设置一个由其他信号整合得到的、能够触发功能调用的变量。初始条件的复位只能由参数 Done、NDR、Error 引起。

对应的控制机制既可置于功能调用前，也可置于功能调用后。若控制机制置于调用后，则可将输出变量定义为本地变量（优点：减少全局变量标志位，数据变量，且相对数据变量有时间优势）。

触发参数必须为全局变量（例如标志位、数据变量）。

在 OB100 中必须通过用户程序将尚在生效的任务复位（参数 Req、Start 等 TRUE ⇒ FALSE）。通过关闭和重启便可进入尚有激活任务的状态。

并行轴

并行轴具有以下属性：

- 其必须通过 NC 机床数据定义。
- 其既可由 PLC 运行，也可由 NC 通过 JOG 键运行。
- 可在 NC 运行方式 MDI 和 JOG 下借助 FC 通过 PLC 启动。
- 启动独立于 NC 程序段界限进行。

针对定位轴、分度轴和主轴可使用 FC（FC18）。

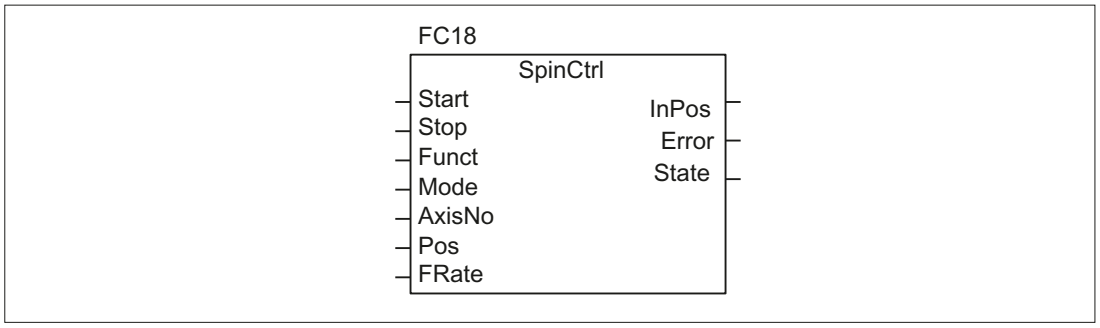
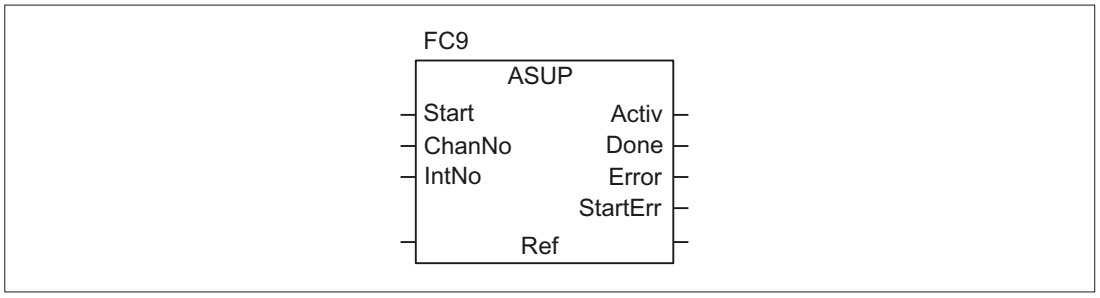


图 14-11　　FC18 输入/输出参数

异步子程序（ASUB）

通过 ASUB 可触发 NC 中的任意功能。通过 PLC 启动 ASUB 的前提条件是：该 ASUB 存在，且通过 NC 程序或 FB4 PI 服务（ASUB）进行了预处理。

以此类方式预处理的 ASUB 可由 PLC 在任意时刻启动。参数设置的 FC9 通道中运行的 NC 程序会被 ASUB 中断。ASUB 启动通过从用户程序以参数 Start = 1 调用 FC9 实现。



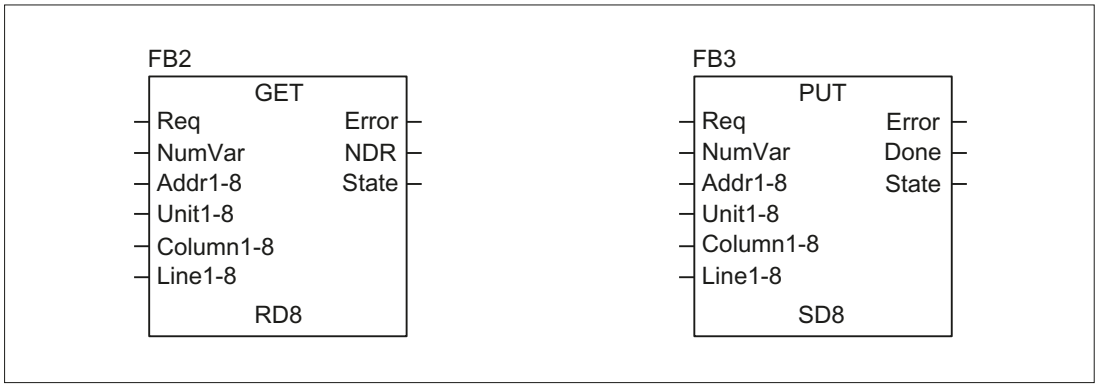
说明

若 ASUB 尚未通过零件程序或 FB4（ASUB）进行预处理（例如未设定中断编号），则会报告启动故障（StartErr = TRUE）。

读写 NC 变量

可通过 FB GET 读取 NC 的变量，以及通过 FB PUT 将数值输入至 NC 变量。NC 变量通过输入 Addr1 至 Addr8 上的名称定址。名称（符号）指向地址设定，该设定必须保存在一个全局 DB 中。基本程序会随附一个 PC 软件（NC-Var-Selector）用于生成该 DB，通过此软件可从随附的变量表中选择所需变量。选择的变量首先会被汇总至另一个项目相关列表。通过**生成 DB** 命令可生成一个“.AWL”文件，该文件必须集成至相关机床的程序文件中，并与机床程序一同编译。

通过一个读取或写入任务可读写 1 到 8 个数值。必要时系统会对这些值进行转换[例如 NC 64 位浮点值转换为 PLC 格式或反之；PLC 格式为 32 位，其中 24 位尾数和 8 位指数]。从 64 位转换为 32 位 REAL 后数值精度降低。32 位 REAL 数值的最大精度约为 10 的 7 次方。



AG_SEND / AG_RECV 功能

AG_SEND / AG_RECV 的功能对应 STEP 7 中 S7-300 CPU 的库“SIMATIC_NET_CP”的功能。在线帮助总体适用于这两个功能。
AG_SEND / AG_RECV 可通过集成的“CP 840D sl”用于和另一个工作站的数据交换。对这两个功能的说明请参见“模块说明 (页 1051)”章节。

说明

SINUMERIK 840D sl 不支持其他通过 CP343-1 生效的通讯模块（例如 BSEND、USEND）。

14.10.8 含接口 DB 的用户程序的符号编程

简介

说明

Toolbox 840D sl 的供货 CD 上在基本程序库中随附了 NST_UDTB.AWL 和 TM_UDTB.AWL 文件。

基本程序的 CPU 程序中保存了从这两个文件编译得到的 UDT 模块。

UDT 是由用户定义的数据类型，其例如可指定给一个 CPU 中生成的数据块。

这些 UDT 模块中定义了近乎所有接口信号的符号名称。

使用的 UDT 编号为 2、10、11、19、21、31、71、72、73、77、1002、1071、1072、1073。

已指定的对应关系如下：

UDT 指定		
UDT 编号	**接口 DB 指定**	**含义**
UDT2	DB2	报警/消息
UDT10	DB10	NCK 信号
UDT11	DB11	BAG 信号
UDT19	DB19	HMI 信号
UDT21	DB21 至 DB30	通道信号
UDT31	DB31 至 61	轴/主轴信号
UDT71	DB71	刀具管理：装刀/卸刀位
UDT72	DB72	刀具管理：主轴切换
UDT73	DB73	刀具管理：刀塔切换
UDT77	DB77	MCP 信号和 HHU 信号，含标准 SDB210
UDT1002	DB2	扩展报警/消息（FB1 参数“ExtendAlMsg:=TRUE”）
UDT 1071	DB 1071	刀具管理：装刀/卸刀位置（多刀）
UDT 1072	DB 1072	刀具管理：主轴切换（多刀）
UDT 1073	DB 1073	刀具管理：刀塔切换（多刀）

为了对接口信号进行符号编程，首先须通过符号编辑器对接口数据块进行符号设定。

为此，例如在符号文件中将符号“AchseX”指定给数据类型为 UDT31 的操作数 DB31。

完成输入后，对该接口的 STEP 7 编程可采用符号进行。

说明

此前创建的、使用这些接口 DB 的程序也可转换为符号形式。为此，先前创建的程序中需要有一个完整且正确的指令用于数据访问，例如“U DB31.DBX60.0”用于“主轴/回转轴”。启用符号编程时，该指令会在编辑器中转换为“AchseX.E_SpKA”。

说明

NST_UDTB.AWL 和 TM_UDTB.AWL 这两个 AWL 文件中定义了接口信号的缩写符号名称。

为了建立对这些接口信号名称的参照，必须在每个信号后的注释中注明名称。

这些名称基于英语。注释也同样为英语。

通过 STEP 7 编辑器，可在打开 UDT 模块时查看其符号名称、注释和绝对地址。

说明

未使用的位和字节例如会以“f56_3”的形式注明。

- “56”：对应数据块的字节地址
- “3”：该字节中的位编号

14.10.9 基于列表的 M 解码

功能说明

借助“基于列表的 M 解码”功能，可通过基本程序对最多 256 个带扩展地址的 M 功能进行解码。 此功能通过 FB1 参数“ListMDecGrp”（需解码的 M 功能组的数量）激活。 带扩展地址的 M 功能通过解码列表指定给信号列表中的信号。 此时还会进行分组。

解码列表 (DB75)

解码列表的源文件（MDECLIST.AWL）随基本程序提供。 对 AWL 源进行编译后生成数据块 DB 75。 激活功能前，必须将解码列表(DB75)传输至 AG 并执行重启。

当一个 M 功能被包含在解码列表中时，系统会对其进行解码。 M 功能解码时，系统会针对特定组置位信号列表中的信号。 信号列表中信号置位的同时，基本程序会在对应 NC 通道中将接口信号“读取禁止”置位。 一旦用户将该通道输出的所有信号复位（即应答），系统会立即为此通道复位该接口信号。

信号列表 (DB76)

功能激活时，基本程序会在数据块 DB76 中建立信号列表。 之后针对每个依据列表解码的 M 信号，信号列表（DB76）的对应组中都会置位一个信号。 同时系统还会在输出 M 功能的通道中置位“读取禁止”信号。 一旦用户将该通道输出的所有信号复位（即应答），系统会立即为此通道复位该接口信号。

快速辅助功能

在将解码列表中包含的 M 功能作为“快速辅助功能”输出时，系统不会为对应的 NC 通道置位读取禁止。

下图显示了**基于列表的 M 解码**的结构：

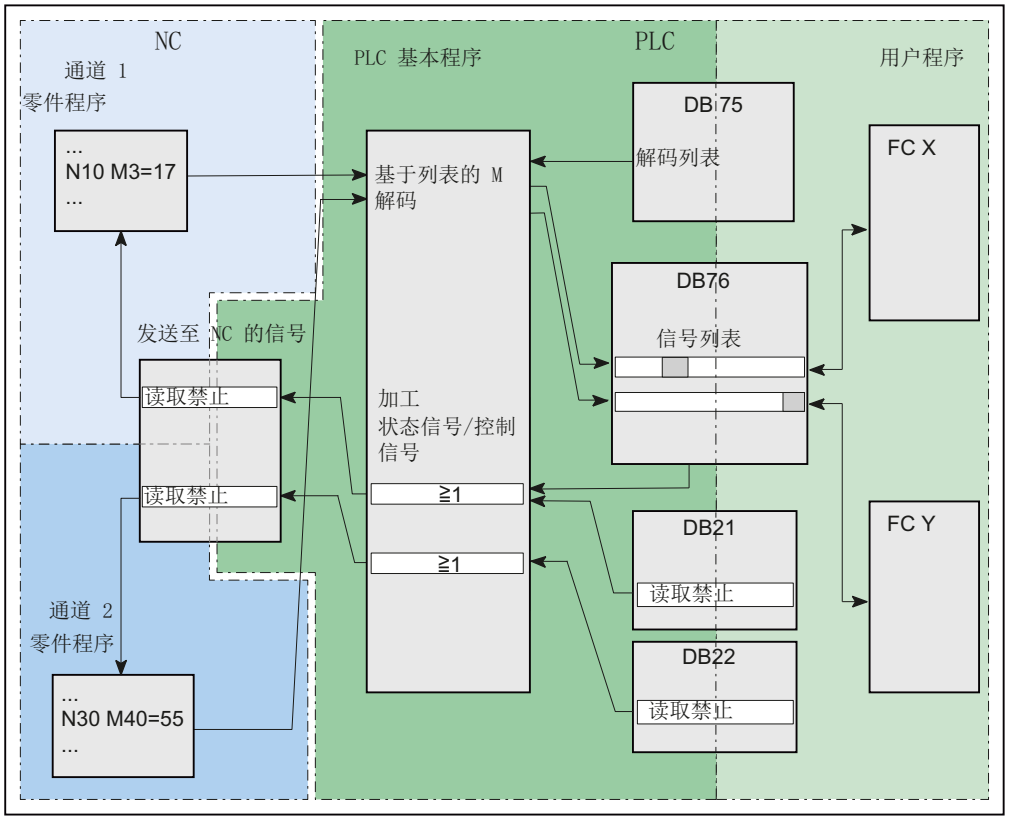


图 14-12　　基于列表的 M 解码

激活

M 解码通过 FB1 参数“**ListMDecGrp**”激活

通过此参数设定待分析或待解码的 M 功能组的数量。 设定的参数值 = 1 ... 16 时，功能激活。

- 基本程序，OB100，FB1 参数 **ListMDecGrp = <**M 功能组数量**>**（另见“FB1：RUN_UP - 基本程序、启动部分 (页 1051)”）。

解码列表（DB75）的属性和结构

解码列表（DB75）的**属性**：

- 解码列表只有一个，为跨通道功能。
- 解码列表最多可包含 16 个组。
- 一个组最多包含 16 个信号
- 对于每个待解码的 M 功能组，解码列表中都包含一条对应的记录
- 带扩展地址的 M 功能在解码列表中通过对应组的第一个和最后一个 M 功能指定给信号列表中的待置位信号。
  - 第一个 M 功能：参数：“MFirstAdr”≙信号或位 0
  - 最后一个 M 功能：参数：“MLastAdr”≙ 取决于与第一个 M 功能偏差的最大信号或位 15

解码列表（DB75）的**结构**：

解码列表中的一条记录由 3 个参数组成，其分别被指定给一个组：

组	扩展 M 地址	组中的第一个 M 地址	组中的最后一个 M 地址
1	MSigGrp[1].MExtAdr	MSigGrp[1].MFirstAdr	MSigGrp[1].MLastAdr
2	MSigGrp[2].MExtAdr	MSigGrp[2].MFirstAdr	MSigGrp[2].MLastAdr
...	...	...	...
16	MSigGrp[16].MExtAdr	MSigGrp[16].MFirstAdr	MSigGrp[16].MLastAdr

信号的类型和取值范围：

信号	类型	取值范围	含义
MExtAdr	INT	0 至 99	扩展 M 地址
MFirstAdr	DINT	0 至 99,999,999	组中的第一个 M 地址
MLastAdr	DINT	0 至 99,999,999	组中的最后一个 M 地址

信号列表（DB76）的属性

信号列表（DB76）有以下属性：

- 信号列表只有一个，为跨通道功能。
- 信号列表中每个 M 功能组最多包含 16 个信号。

示例

需要对 3 个 M 功能组进行解码：

- 组 1： M2 = 1 至 M2 = 5
- 组 2： M3 = 12 至 M3 = 23
- 组 3： M40 = 55

解码列表和信号列表的结构

组	解码列表 (DB75)			信号列表 (DB76)
	扩展 M 地址	组中的第一个 M 地址	组中的最后一个 M 地址	
1	2	1	5	DB76.DBX0.0 ... DBX0.4
2	3	12	23	DB76.DBX2.0 ... DBX3.3
3	40	55	55	DB76.DBX4.0

程序代码

```
DATA_BLOCK DB 75
TITLE =
VERSION: 0.0
   STRUCT
      MSigGrp : ARRAY [1 .. 16 ] OF STRUCT
         MExtAdr : INT;
         MFirstAdr : DINT;
         MLastAdr : DINT;
      END_STRUCT;
BEGIN
   MSigGrp[1].MExtAdr   :=    2; 组 1 的扩展 M 地址
   MSigGrp[1].MFirstAdr :=  L#1; 组中的第一个 M 地址
   MSigGrp[1].MLastAdr  :=  L#5; 组中的最后一个 M 地址
   MSigGrp[2].MExtAdr   :=    3; 组 2 的扩展 M 地址
   MSigGrp[2].MFirstAdr := L#12; 组中的第一个 M 地址
   MSigGrp[2].MLastAdr  := L#23; 组中的最后一个 M 地址
   MSigGrp[3].MExtAdr   :=    40; 组 3 的扩展 M 地址
   MSigGrp[3].MFirstAdr := L#55; 组中的第一个 M 地址
   MSigGrp[3].MLastAdr  := L#55; 组中的最后一个 M 地址
END_DATA_BLOCK
```

OB100 中 FB1 的结构

插入用于设定待解码 M 功能数量的参数“ListMDecGrp”以激活功能。

```
Call FB 1, DB 7(
   ...
   ListMDecGrp := 3;      // 对 3 个组进行 M 解码
   ...
);
```

说明

在 OB100 中完成输入并将解码列表（DB75）传输至 AG 后，必须执行重启。 基本程序会在重启过程中创建信号列表（DB76）。

接下来系统例如会在通道 1 中启动 NC 程序。 该程序中包含一个扩展 M 功能（M3=17）。M 功能解码（M3 ≙ 组 2）时，系统会置位信号列表（DB76）中的对应信号（DBW1.5），并在通道 1 中置位接口信号“读取禁止”。 NC 程序的执行停止。 此外还会在通道 1 的通道 DB 中显示“M 功能扩展地址”和“M 功能编号”。

一旦用户将通道 1 输出的信号列表（DB76）中的所有信号复位，从而完成应答，该通道中的“读取禁止”信号会立即复位。

14.10.10 PLC 机床数据

简介

用户可在 NC 中保存 PLC 专用机床数据。这些机床数据可在 PLC 启动后（OB100）由用户处理。这样一来例如可执行用户选件、机床扩展级、机床配置等功能。

用于读取这些数据的接口位于 DB20 中。而 DB20 仅在使用用户机床数据时（即基本程序参数“UDInt”、“UDHex”和“UDReal”的总和大于零）才会由基本程序在启动中创建。

DB20 各区域的大小，以及由此得到的总长度通过以下 PLC 机床数据设置：

MD14504 $MN_MAXNUM_USER_DATA_INT

MD14506 $MN_MAXNUM_USER_DATA_HEX

MD14508 $MN_MAXNUM_USER_DATA_FLOAT

这些设置会注明在基本程序参数“UDInt”、“UDHex”和“UDReal”中，以作为对用户的提示。

DB20 中的数据存储由基本程序以以下顺序进行：

1. 整数 MD
2. 十六进制数组 MD
3. 实数 MD

整数值和实数值以 S7 格式保存在 DB20 中。

十六进制数据以输入顺序（作为位域使用）保存在 DB20 中。

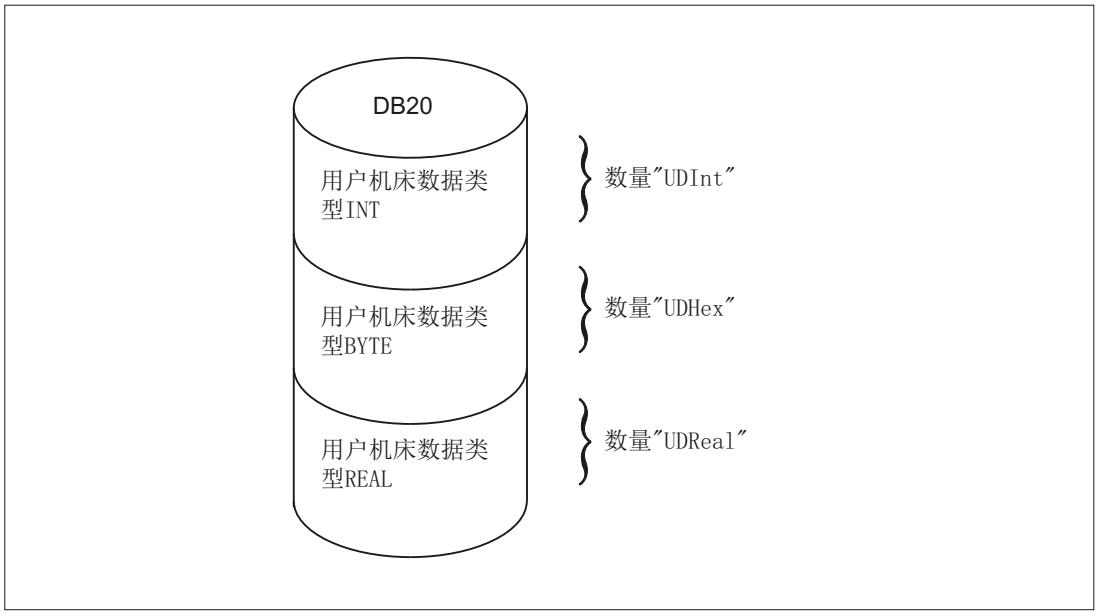


图 14-13　DB20

说明

若之后需提升使用的 PLC 机床数据的数量，则需提前删除 DB20。为了避免此类扩展对当前用户程序产生影响，DB20 中的数据访问应尽量以符号形式进行，例如通过 UDT 结构说明。

报警	
400120	**在 PLC 中删除 DB20 并重启**
说明	DB 长度与所需长度不同
响应	报警响应和 PLC 停止
解决办法	删除 DB20 并随后执行复位
继续执行	重启后

示例

示例中的项目需要 4 个 INTEGER 值、2 个含位信息 Hexa 数组、1 个 Real 值。

机床数据：

MD14510 $MN_USER_DATA_INT[0]	123
MD14510 $MN_USER_DATA_INT[1]	456
MD14510 $MN_USER_DATA_INT[2]	789
MD14510 $MN_USER_DATA_INT[3]	1011
...	
MD14512 $MN_USER_DATA_HEX[0]	12
MD14512 $MN_USER_DATA_HEX[1]	AC
...	
MD14514 $MN_USER_DATA_FLOAT[0]	123.456

基本程序参数（OB100）：

```
CALL FB1, DB7(
          MCPNum :=                 1,
          MCP1In :=                 P#E0.0,
          MCP1Out :=                P#A0.0,
          MCP1StatSend :=           P#A8.0,
          MCP1StatRec :=            P#A12.0,
          MCP1BusAdr :=             6,
          MCP1Timeout :=            S5T#700MS,
          MCP1Cycl :=               S5T#200MS,
          NCCyclTimeout :=          S5T#200MS,
          NCRunupTimeout :=         S5T#50S;
```

基本程序参数（查询运行时间）：

```
          l gp_par.UDInt;           //=4,
          l gp_par.UDHex;           //=2,
          l gp_par.UDReal;          //=1 )
```

PLC 启动期间已创建长度为 28 字节的 DB20：

DB20	
地址	**数据**
0.0	123
2.0	456
4.0	789
6.0	1011
8.0	b#16#12
9.0	b#16#AC
10.0	1.234560e+02

所使用的机床数据的结构以 UDT 设定：

```
TYPE UDT20
   STRUCT
      UDInt :    ARRAY [0 ..3] OF INT;
      UDHex0 :   ARRAY [0 ..15] OF BOOL;
      UDReal :   ARRAY [0 ..0] OF REAL;      //作为数组描述，
                                             //用于之后的扩展
   END_STRUCT;
END_TYPE
```

说明

ARRAY OF BOOL 总是对应偶数地址。因此 UDT 定义中通常须选择 0 至 15 的数组区域，或单独设定所有布尔变量。

尽管示例中最初只使用了一个 REAL 值，但仍为变量创建了一个数组（包含一个单元）。这样一来，之后无需修改符号地址便可方便地进行扩展。

符号访问

符号访问会被记录在符号列表中：

符号	操作数	数据类型
UData	DB20	UDT20

在用户程序中访问（只显示符号读取访问）：

```
...
          L       "UData".UDInt[0];
          L       "UData".UDInt[1];
          L       "UData".UDInt[2];
          L       "UData".UDInt[3];

          U       "UData".UDHex0[0];
          U       "UData".UDHex0[1];
          U       "UData".UDHex0[2];
          U       "UData".UDHex0[3];
          U       "UData".UDHex0[4];
          U       "UData".UDHex0[5];
          U       "UData".UDHex0[6];
          U       "UData".UDHex0[7];
          ...     ...
          U       "UData".UDHex0[15];

          L       "UData".UDReal[0];
...
```

14.10.11 机床控制面板、手持操作设备、直接控制按键的配置

简介

最多可同时运行 2 个机床控制面板和一个手持操作设备。 机床控制面板（MCP）和手持操作设备（HHU）可采用多种连接方式（以太网/PROFINET、PROFIBUS）。 可将 2 个 MCP 连接至不同的总线系统（仅在采用以太网和 PROFIBUS 时才支持混合运行）。 这可通过 FB1 参数“MCPBusType”实现。 此参数中右边的十进制位（个位）对应第 1 个 MCP，左边的十进制位（十位）对应第 2 个 MCP。

操作组件一般通过在 OB100 中调用基本程序功能块 FB1 来进行参数设置。FB1 将其参数保存在对应的实例 DB（DB7，符号“gp_par”）中。为此，每个机床控制面板和手持操作设备都会有一个专属的参数组。 用户必须在这些参数组中定义输入地址和输出地址。 这些输入和输出地址也在 FC19、FC24、FC25、FC26 和 FC13 中使用。 此外还须定义状态信息、PROFIBUS 或以太网/PROFINET 的地址。 超时的时间设置和循环强制追加触发则应保留预设值。 MCP、HHU 组件的更多相关信息请参见操作单元手册。

激活

每个组件可通过机床控制面板的数量（“MCPNum” 参数）激活，手持操作设备还可通过 “HHU” 参数激活。 MCP、HHU 的连接通过 FB1 参数“MCPMPI”、“MCPBusType” 或“BHG”、“BHGMPI”定义。

手动操作装置 (HT 2)

手持操作设备的定址通过一个基本程序参数组中的参数进行。 采用此方案是出于参数名称兼容性方面的原因。

配置

MCP/HHU 和 PLC 之间的数据传输可能会基于不同的通讯机制，这取决于 MCP 或 HHU 的总线连接。 其中一种情形是通过“CP 840D sl”传输数据（以太网）。

此时参数设置完全通过 FB1 中的 MCP/HHU 参数进行。

另一种情形是基于 PROFIBUS 配置通过 PLC 操作系统传输数据。

此时参数设置通过 STEP 7 在 HW Config 中进行。为了通过基本程序访问这些数据，以及对 MCP/HHU 进行故障监控，必须将 FB1 参数中设置的地址提供给基本程序。

下文会对各种对各种连接方式进行简要介绍。 此外也可配置混合运行。

若通过时间监控识别出一个故障，则会将其记录在 PLC-CPU 的报警缓存中（报警 400260 至 400262）。 此时 MCP 或手持操作设备（MCP1In/MCP2In 或 BHGIn）发出的输入信号会被复位为 0。 若 PLC 和 MCP/HHU 之间重新同步，那么通讯将自动恢复，报警消息会由基本程序删除。

说明

下表中的“(n.r.)”缩写表示不相关（not relevant）。

以太网连接（MCPBusType = 5）

通讯由 PLC 基本程序通过 CP 840D sl 直接建立，无需进一步配置。 参数设置通过下面列出的 FB1 参数进行。

在“MCP1 BusAdr”、“MCP2 BusAdr”或“BHGRecGDNo”（对应节点的总线地址）中必须输入组件逻辑名称的数字部分。 逻辑名称通过 MCP 上的开关或接线盒设置。

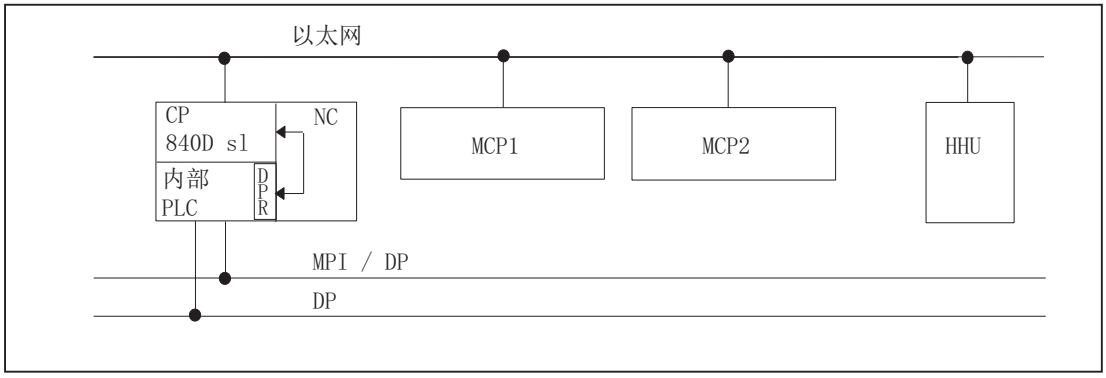


图 14-14　以太网连接

相关参数（FB1）		
MCP		**HHU**
MCPNum=1 或 2（MCP 数量）		BHG = 5（通过 CP 840D sl）
MCP1In	MCP2In	BHGIn
MCP1Out	MCP2Out	BHGOut
MCP1StatSend	MCP2StatSend	BHGStatSend
MCP1StatRec (n.r.)	MCP2StatRec (n.r.)	BHGStatRec
MCP1BusAdr	MCP2BusAdr	BHGInLen (n.r.)
MCP1Timeout (n.r.)	MCP2Timeout (n.r.)	BHGOutLen (n.r.)
MCP1Cycl (n.r.)	MCP2Cycl (n.r.)	BHGTimeout (n.r.)
MCPMPI = FALSE		BHGCycl (n.r.)
MCP1Stop	MCP2Stop	BHGRecGDNo
MCP1NotSend	MCP2NotSend	BHGRecGBZNo (n.r.)
		BHGRecObjNo (n.r.)
MCPBusType = b#16#55（通过 CP 840D sl）		BHGSendGDNo (n.r.)
		BHGSendGBZNo (n.r.)
MCPSDB210 = FALSE		BHGSendObjNo (n.r.)
MCPCopyDB77 = FALSE		**BHGMPI = FALSE**
		BHGStop
		BHG NotSend

针对时间监控，PLC 的报警缓存中会生成故障记录。 借此在 HMI 上显示以下故障消息：

- 400260: OPI 1 故障
  或者
- 400261: OPI 2 故障
- 400262: HHU 故障

重启后，在 MCP/HHU 和 PLC 之间尚未交换数据时系统便可识别出 MCP 或 BHG 的故障。一旦所有组件在启动后通报“就绪”，监控功能便会立即激活。

示例： 带直接控制按键的 OP

需要将以太网总线上的 OP 的直接控制按键传输至 PLC。 在此之前，直接控制按键通过 PROFIBUS 或 OP 和 MCP 间的特殊电缆连接传输至 PLC。

为了通过以太网连接直接控制按键（此示例中为“OP 08T”），需要在基本程序中进行参数设置用于激活数据传输。 对应的参数位于 FB 1 的实例 DB 中（OpKeyNum 至 OpKeyBusType，参见数据表）。 这些参数由用户在 OB100 启动期间 FB1 调用中通过参数互联提供。 总线地址和 Op1/2KeyStop 也可在循环运行中通过“FB1 实例 DB”DB7 进行修改。

直接控制按键的用户数据的传输过程与以太网 MCP 相似。 也可通过写入 DB7 参数“Op1/2KeyStop”停止和重新启动数据传输。 停止期间还可对直接控制按键模块的地址（TCU 索引或 MCP 地址）进行修改。

停止信号复位后将建立与新地址的连接。

直接控制按键接口的状态可在接口信号中读取：

DB10.DBX104.3（OP1Key 就绪）

或者

DB10.DBX104.4（OP2Key 就绪）

直接控制按键的地址

对于 Op1/2KeyBusAdr 参数，通常须使用 TCU 索引。 这适用于 OP 08T、OP 12T 这一类**未配备**特殊电缆连接（用于将直接控制按键连接至以太网 MCP）的 OP。

若带直接控制按键的 OP 配备了特殊电缆连接，且其与一个以太网 MCP 相连，那么必须为 Op1/2KeyBusAdr 参数使用 MCP 地址（MCP 的 DIP 开关位置）。 直接控制按键接口只用于传输该按键的数据流（2 字节）。

直接控制按键报警

针对时间监控，PLC 的报警缓存中会生成故障记录。 借此在 HMI 上显示以下故障消息：

- 400274: 直接控制按键 1 故障
  或者
- 400275: 直接控制按键 2 故障

使用直接控制按键时的操作单元切换

用户在启动模块 OB100 中将 Op1/2KeyBusAdr 与 0xFF 及 Stopp = TRUE 互联。通过 M:N 模块 FB9，M:N 接口的直接控制按键地址置于“Op1KeyBusAdr”参数。

相关参数（FB1）		
直接控制按键		**例如直接控制按键 OP 08T**
OpKeyNum = 1 或 2（带直接控制按键的 OP 的数量）		
Op1KeyIn	Op2KeyIn	
Op1KeyOut	Op2KeyOut	
OpKey1BusAdr	Op2KeyBusAdr	地址： TCU 索引
Op1KeyStop	Op2KeyStop	
Op1KeyNotSend	Op2KeyNotSend	
OpKeyBusType = b#16#55（通过 CP 840D sl）		

MCP 识别

通过 DB7 中的识别接口，可在循环运行中通过输入/输出上的相关参数查询以太网组件的类型（MCP、HT 2、HT 8 或直接控制按键）。

- 输入上的相关参数：
  “IdentMcpBusAdr”、“IdentMcpProfilNo”、“IdentMcpBusType”、“IdentMcpStrobe”
- 输出上的相关参数：
  “IdentMcpStrobe”、“IdentMcpLengthIn”、“IdentMcpLengthOut”

此时参数“IdentMcpBusAdr”上的 DIP 设备地址或 TCU 索引会由用户程序配合选通脉冲信号的置位激活。

输入参数“IdentMcpProfilNo”通常须设为“0”值。 仅在识别直接控制按键式才须将此参数设为“1”。 参数“IdentMcpBusType”目前对用户程序无意义，可采用缺省值。

选通脉冲信号复位后，用户可通过基本程序获取有效的输出信息。 由基本程序进行的选通脉冲信号复位可能会持续多个 PLC 周期（最多 2 s）。

输出参数用于向用户显示相应设备的数据区域的大小。 此外例如还可确定接线盒上是否连接了 HT 2 或 HT 8，或未连接设备。 此信息可用于激活 MCP 通道或 HHU 通道。 在循环运行中可通过 DB7 的符号名称（gp_par）以符号写入和读取用户程序的参数。

相关参数（FB1）		
MCP 设备识别		**输入参数，以 OP 08T 为例**
输入	**输出**	**直接控制按键对应的值**
IdentMcpBusAdr	IdentMcpType	IdentMcpBusAdr = TCU 索引
IdentMcpBusProfilNo	IdentMcpLengthIn	IdentMcpBusProfilNo = 值 1
IdentMcpBusType	IdentMcpLengthOut	IdentMcpBusType = 缺省值
IdentMcpStrobe		
IdentMcpBusProfilNo		**值**
MCP, BHG, HT 8, HT 2		B#16#0
直接控制按键，例如 OP 08T、OP 12T		B#16#1
IdentMcpType (Mcp-Type)		
未连接设备		0
MCP 483C IE (Compact)		B#16#80
MCP 483C IE		B#16#81
MCP 310		B#16#82
MCP OEM		B#16#83
MCP DMG		B#16#84
HT 8		B#16#85
TCU_DT（直接控制按键）		B#16#86
MCP_MPP		B#16#87
HT 2		B#16#88
OP 08T（直接控制按键）		B#16#89

DP 端口上的 PROFIBUS 连接（MCPBusType = 3）

MCP 采用 PROFIBUS 连接时，必须在 STEP 7 的硬件配置中对相应组件进行参数设置。 MCP 连接至 PLC 的标准 DP 总线（**不连接至 MPI/DP**）。 地址必须保存至输入和输出映射区域。 初始地址还须保存至 FB1 的 Pointer 参数中。 其他参数设置通过下面列出的 FB1 参数进行。

HHU 尚无支持 PROFIBUS 的规格。 因此下图中将为 HHU 显示以太网连接。 PROFIBUS 从站地址必须保存在参数“MCP1BusAdr”和“MCP2BusAdr”中。 “MCPxStatRec”中必须输入指向配置的诊断地址的 Pointer 参数（例如 P#A8190.0）。

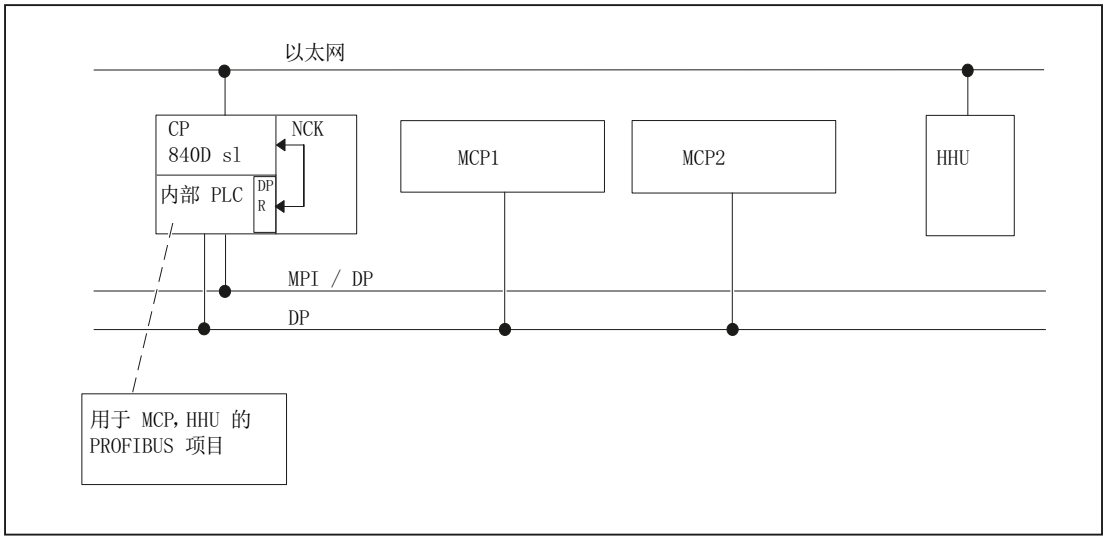


图 14-15　PROFIBUS 连接

相关参数（FB1）		
MCP		**HHU**
MCPNum = 1 或 2（MCP 数量）		BHG = 5（通过 CP 840D sl）
MCP1In	MCP2In	BHGIn
MCP1Out	MCP2Out	BHGOut
MCP1StatSend (n.r.)	MCP2StatSend (n.r.)	BHGStatSend
MCP1StatRec	MCP2StatRec	BHGStatRec
MCP1BusAdr	MCP2BusAdr	BHGInLen
MCP1Timeout	MCP2Timeout	BHGOutLen
MCP1Cycl (n.r.)	MCP2Cycl (n.r.)	BHGTimeout (n.r.)
MCPMPI = FALSE		BHGCycl (n.r.)

相关参数（FB1）		
MCP		**HHU**
MCP1Stop	MCP2Stop	BHGRecGDNo
MCPBusType = b#16#33		BHGRecGBZNo (n.r.)
		BHGRecObjNo (n.r.)
MCPSDB210 = FALSE		BHGSendGDNo (n.r.)
MCPCopyDB77 = FALSE		BHGSendGBZNo (n.r.)
		BHGSendObjNo (n.r.)
		BHGMPI = FALSE
		BHGStop

MCP 故障通常会使 PLC 切换至停止状态。 若需改变此特性，可通过 OB82、OB86 避免停止。 基本程序在缺省设置下会调用 OB82 和 OB86。 在这些 OB 中会调用 FC5。 FC5 会检查发生故障的从站是否为 MCP。 若为是，则不触发 PLC 停止。 通过设置“MCPxStop”= TRUE，基本程序通过 SFC12 将 MCP 作为从站取消。 若 MCP 发生故障时 PLC 未进入停止状态，那么基本程序会生成报警消息。 工作站恢复时报警会被删除。

MPI/DP 端口上的 PROFIBUS 连接（MCPBusType = 4）

MCP 采用 PROFIBUS 连接时，必须在 STEP 7 的硬件配置中对相应组件进行参数设置。 MCP 连接至 PLC 的 MPI/DP 总线。

地址必须保存至输入和输出映射区域。 初始地址还须保存至 FB1 的 Pointer 参数中。 其他参数设置通过下面列出的 FB1 参数进行。HHU 尚无支持 PROFIBUS 的规格。 因此下图中将为 HHU 显示以太网连接。 PROFIBUS 从站地址必须保存在 MCP1BusAdr 和 MCP2BusAdr 中。 MCPxStatRec 中必须输入指向配置的诊断地址的 Pointer 参数（例如 P#A8190.0）。

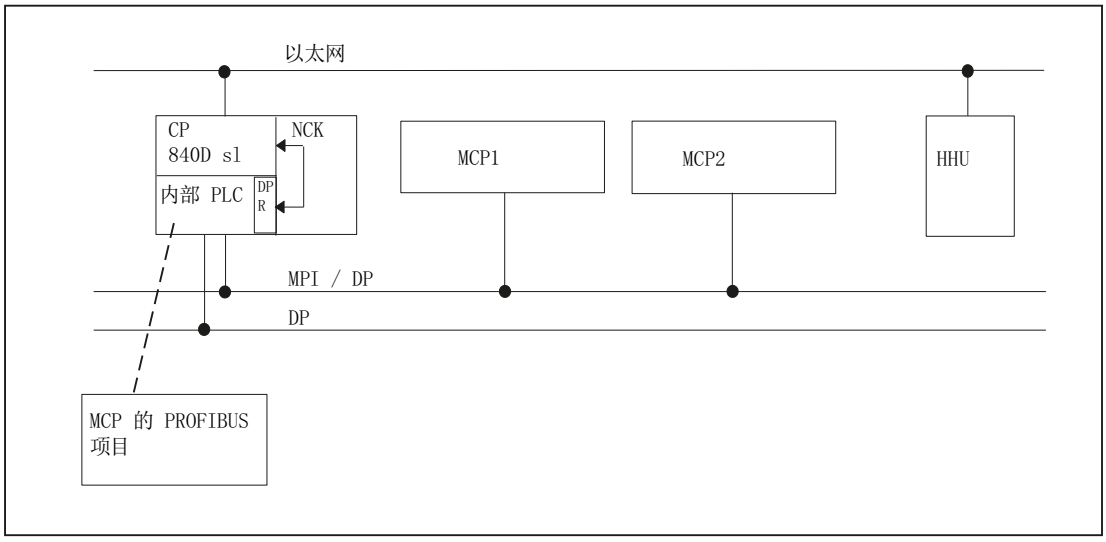


图 14-16　MPI/DP 端口上的 PROFIBUS 连接

相关参数（FB1）		
MCP		**HHU**
MCPNum = 1 或 2（MCP 数量）		BHG = 5（通过 CP 840D sl）
MCP1In	MCP2In	BHGIn
MCP1Out	MCP2Out	BHGOut
MCP1StatSend (n.r.)	MCP2StatSend (n.r.)	BHGStatSend
MCP1StatRec	MCP2StatRec	BHGStatRec
MCP1BusAdr	MCP2BusAdr	BHGInLen
MCP1Timeout	MCP2Timeout	BHGOutLen
MCP1Cycl (n.r.)	MCP2Cycl (n.r.)	BHGTimeout (n.r.)
MCPMPI = FALSE		BHGCycl (n.r.)
MCP1Stop	MCP2Stop	BHGRecGDNo
MCPBusType = b#16#44		BHGRecGBZNo (n.r.)
		BHGRecObjNo (n.r.)
MCPSDB210 = FALSE		BHGSendGDNo (n.r.)
MCPCopyDB77 = FALSE		BHGSendGBZNo (n.r.)
		BHGSendObjNo (n.r.)
		BHGMPI = FALSE
		BHGStop

MCP 故障通常会使 PLC 切换至停止状态。 若需改变此特性，可通过 OB82、OB86 避免 PLC 停止。 基本程序在缺省设置下会调用 OB82 和 OB86。 在这些 OB 中会调用 FC5。 FC5 会检查发生故障的从站是否为 MCP。 若为是，则不触发 PLC 停止。 通过设置 MCPxStop:= TRUE，基本程序通过 SFC12 将 MCP 作为从站取消。 若 MCP 发生故障时 PLC 未进入停止状态，那么基本程序会生成报警消息。 工作站恢复时报警会被删除。

PROFINET 连接（MCPBusType = 6）

MCP 采用 PROFINET 连接时，必须在 STEP 7 的硬件配置中对相应组件进行参数设置。 此时 MCP 与 CPU 的 PROFINET 模块相连。

在 HW Config 中对 MCP 进行参数设置时，地址必须保存至输入和输出映射区域。 初始地址还须保存至 FB1 的 Pointer 参数（MCPxIn 和 MCPxOut）中。 这是因为 MCP 和基本程序之间通过这些参数进行信号传输。 参数 MCPxIn 还用于对 MCP 的监控。 因此对于此 MCP 规格，MCPxBusAdr 为不相关参数。

MCPxStatRec 中必须输入指向配置的诊断地址的 Pointer 参数（例如 P#A8190.0）。

PROFINET-MCP 的总线类型为独有类型，必须在 MCPBusType 参数中启用。

其他参数设置通过下面列出的 FB1 参数进行。 HHU 尚无支持 PROFIBUS 的规格。 下图中将为 HHU 显示以太网连接。

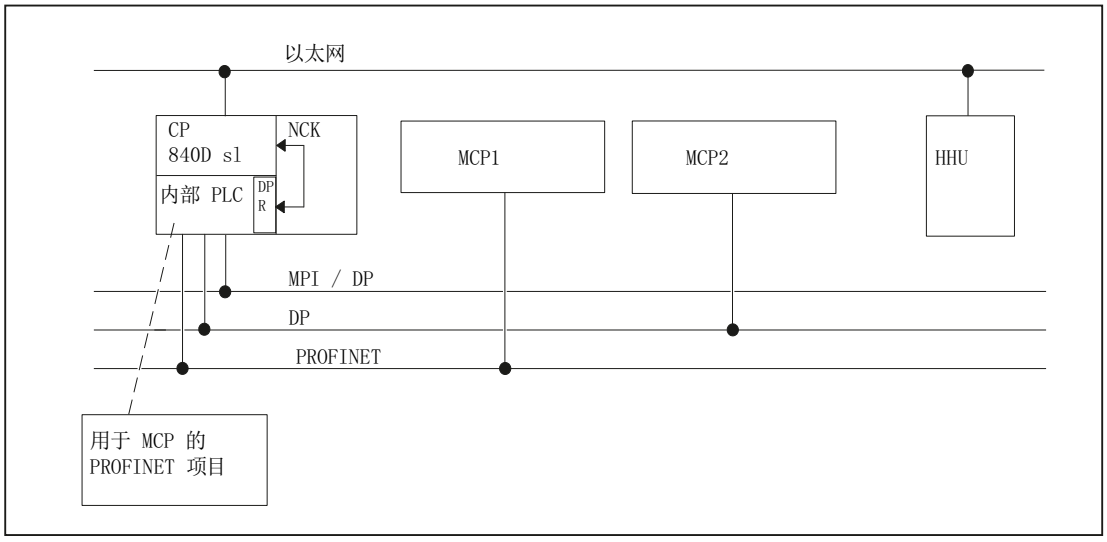


图 14-17　　PROFINET 连接

相关参数（FB1）		
MCP		**HHU**
MCPNum = 1 或 2（MCP 数量）		BHG = 5（通过 CP 840D sl）
MCP1In	MCP2In	BHGIn
MCP1Out	MCP2Out	BHGOut
MCP1StatSend (n.r.)	MCP2StatSend (n.r.)	BHGStatSend
MCP1StatRec	MCP2StatRec	BHGStatRec
MCP1BusAdr (n.r.)	MCP2BusAdr (n.r.)	BHGInLen
MCP1Timeout	MCP2Timeout	BHGOutLen
MCP1Cycl	MCP2Cycl	BHGTimeout (n.r.)
MCPMPI = FALSE		BHGCycl (n.r.)
MCP1Stop	MCP2Stop	BHGRecGDNo
MCPBusType = b#16#36 （如图中示例） (6 = PROFINET 用于 MCP1) (3 = PROFIBUS 用于 MCP2)		BHGRecGBZNo (n.r.)
		BHGRecObjNo (n.r.)
MCPSDB210 = FALSE		BHGSendGDNo (n.r.)
MCPCopyDB77 = FALSE		BHGSendGBZNo (n.r.)
		BHGSendObjNo (n.r.)
		BHGMPI = FALSE
		BHGStop

MCP 故障通常会使 PLC 切换至停止状态。 若需改变此特性，可通过 OB82、OB86 避免 PLC 停止。 基本程序在缺省设置下会调用 OB82 和 OB86。 在这些 OB 中会调用 FC5。 FC5 会检查发生故障的从站是否为 MCP。 若为是，则不触发 PLC 停止。 监控中出现 MCP 故障时，参数 MCPxIn 上的输入地址有重要意义。

通过设置 MCPxStop:= TRUE，基本程序通过 SFC12 将 MCP 作为从站取消。 若 MCP 发生故障时 PLC 未进入停止状态，那么基本程序会生成报警消息。 工作站恢复时报警会被删除。

14.10.12 机床控制面板、手持操作设备的切换

从标准意义上而言，只有以太网通讯方案支持操作组件（MCP 或 HHU）的切换或断开。

PROFIBUS 通讯方案

采用 PROFIBUS 通讯方案时，此功能被限制在特定范围，且须由用户进行额外的操作。

例如，对于 PROFIBUS 规格的 MCP，可将为 MCP1、MCP2 或 HHU 指定的 DB77 数据区域用于 FB1 上的 MCP 的 Pointer。MCP 的从站总线地址必须在 MCPxBusAdr 下正确输入，因为监控通过该参数实现。生效 MCP 的信号必须通过用户程序的复制程序从 HW Config 中配置的 I/O 区域复制到 DB77。这样便可通过信号对 PROFIBUS 上的多个 MCP 进行切换。在从一个 MCP 切换至另一个的过程中，必须将参数 MCPxStop 设置为 TRUE。

控制信号

可通过 MCP1Stop、MCP2Stop 和 BHGStop 参数停止与各组件的通讯（赋值 = 1）。通讯的停止及激活可在运行的循环中进行。不过数值修改不允许通过重新调用 FB1 进行，而是通过以符号方式写入参数来实现。

停止 MCP 1 传输的示例：

```
SET;
S gp_par.MCP1Stop;
```

设置参数 MCP1Stop、MCP2Stop、BHGStop 时，报警 400260 至 400262 也会被抑制或删除。

切换总线地址

请按如下步骤中断与现有操作组件（MCP 或 HHU）的通讯连接并通过另一个通讯地址建立与另一个操作组件（MCP 或 HHU）的新通讯连接。

1. 停止需要解耦的操作组件的通讯：参数 MCP1Stop、MCP2Stop 或 BHGStop = 1
2. 通讯停止，当：DB10, DBX104.0, .1 或 .2 == 0
3. 修改总线地址：
   - MCP：FB1 参数 MCP1BusAdr 或 MCP2BusAdr = <新操作组件的总线地址>
   - HHU（以太网通讯方案）：FB1 参数 BHGRecGDNo = <新操作组件的总线地址>
4. 使能通讯（在同一个 PLC 循环中，如同点 3）：参数 MCP1Stop、MCP2Stop 或 BHGStop = 0
5. 与新组件的通讯激活，当 DB10, DBX104.0, .1 或 .2 == 1

取消以太网 MCP 的 LED 闪烁

重新上电后，MCP 会通过 LED 闪烁显示启动结束以及等待建立连接。可按如下方式取消 LED 闪烁。目前尚**无法**实现该特性在 MCP 上的掉电存储！

前提条件

MCP 的固件版本（自 V02.02.04 起）

用于取消闪烁的设置

开始与 MCP 通讯前，必须在 MCPxStop 中设置发送状态。开始通讯前表示在设置 DB7 参数 MCPxStop = FALSE 前处于启动状态 (OB100) 或循环运行状态 (OB1)

设置发送状态：FB1 参数 MCPxStatSend，位 30 = 0 和位 31 = 1

此时不会有当前状态的反馈信息。

示例

OB100 中的节选：（基于 MCP1 的示例）

```
CALL "RUN_UP" , "gp_par"
   ...
   MCP1StatSend := P#A 8.0
   ...
   //取消 MCP 闪烁
   SET
   R   A 11.6
   S   A 11.7
   ...
```

14.11 用于 Safety Integrated 的 SPL

PLC 不是基本程序功能，而是用户功能。基本程序提供一个数据块 DB18 用于 Safety PLC 信号，并通过数据校验确保信息相对 NC 中 PLC 程序的一致性。

文档：
Safety Integrated 功能手册

14.12 分配一览

14.12.1 分配：NCK/PLC 接口

对 SINUMERK 840D sl 的 NC/PLC 接口分配的详细说明请见：

文档：

参数手册 sl（第 2 册）

14.12.2 分配：FB/FC

编号	含义
FB15	基本程序
FB1, FC2, FC3, FC5	基本程序
FC0 ... 29	预留用于西门子
FB0 ... 29	预留用于西门子
FC30 ... 999 [1)]	由用户自由分配
FB30 ... 999 [1)]	由用户自由分配
FC1000 ... 1023	预留用于西门子
FB1000 ... 1023	预留用于西门子
FC1024 ... 上限	由用户自由分配
FB1024 ... 上限	由用户自由分配

[1)] 模块（FB/FC）编号的实际上限取决于所选 NCU 中包含的 PLC-CPU。

说明

FC、FB 的分配请参见“PLC 基本程序的存储空间需求 (页 1032)”。

14.12.3 分配：DB

说明

系统会基于 NC 机床数据中的配置创建所需数量的数据块。

数据块一览			
DB 编号	**名称**	**名称**	**包**
1		预留用于西门子	GP
2 ... 5	PLC-MELD	PLC 消息	GP
6 ... 8		基本程序	
9	NC-COMPILE	NC 编译循环的接口	GP
10	NC-NAHTSTELLE	中央 NC 接口	GP
11	BAG 1	运行方式组接口	GP
12		计算机连接和传输系统接口	
13 ... 14		为基本程序预留	
15		基本程序	
16		PI 服务定义	
17		版本标识	
18		为基本程序预留	
19		HMI 接口	
20		PLC 机床数据	
21 ... 30	CHANNEL 1 ... n	NC 通道接口	GP
31 ... 61	AXIS 1 ... m	轴/主轴接口 或由用户自由分配	GP
62 ... 70		由用户自由分配	
71 ... 74		刀具管理	GP
75 ... 76		M 功能组解码	
77		MCP 信号的 DB	
78 ... 80		预留用于西门子	
81 ... 999 [1)]		见下：ShopMill、ManualTurn	
1000 ... 1099		预留用于西门子	
1100 ... 上限		由用户自由分配	

1) 模块（DB）编号的实际上限取决于所选 NCU 中包含的 PLC-CPU。未激活通道、轴/主轴、刀具管理的数据块可由用户自由使用。

说明

未激活通道、轴/主轴、刀具管理的数据块可由用户自由使用。

14.12.4 分配：计时器

计时器编号	含义
T 0 ... T 512 [1)]	用户范围

[1)] 计时器（DB）编号的实际上限取决于所选 NCU 中包含的 PLC-CPU。

14.13 用于 HMI 的 PLC 功能 (DB19)

14.13.1 通道选择

功能

HMI 上显示的通道可由 PLC 用户程序通过 HMI/PLC 接口在机床基本画面中选择。

前提条件

在 NC 中设置了多个通道。

任务和应答接口

DB19	含义	
DBX32.0 - .5	PLC → HMI	功能编号：1 = 通道选择
DBX32.6	PLC ↔ HMI	功能请求
DBX32.7	HMI → PLC	Status：1 = “该功能执行”。

DB19	含义	
DBB33	PLC → HMI	通道编号：1, 2, 3, ... 最大通道数 下一个通道：FF_H
DBB36	HMI → PLC	故障标识： ● 0：无故障 ● 1：故障号无效 (DBX32.0 - .5) ● 2：参数无效 (DBB33 - DBB35) ● 3：HMI 内部变量的写入异常中断。 ● 10：无通道 (DBB33)

功能流程

PLC → HMI

PLC 用户程序必须遵循以下流程：

1. 检查是否有空余接口用于新任务：
   - DB19.DBX32.6 == 0（功能请求）
   - DB19.DBX32.7 == 0（状态）
2. 如果接口未占用，则须输入任务数据并置位功能请求：
   - DB19.DBB33 = <通道编号>
   - DB19.DBX32.0 - .5 = 1（功能编号）
   - DB19.DBX32.6 = 1（功能请求）

HMI → PLC

如果参数设置**正确**，HMI 会作出以下响应：

1. HMI 识别出通道选择的功能请求后，会将状态设为“执行功能”并复位功能请求：
   - DB19.DBX32.7 = 1（状态）
   - DB19.DBX32.6 = 0（功能请求）
2. 执行完通道选择后，HMI 会再次复位状态并设置值 0 作为故障标识：
   - DB19.DBX32.7 = 0（状态）
   - DB19.DBX36 = 0（故障标识）

如果参数设置**错误**，HMI 会作出以下响应：

- 复位功能请求并置位相应的故障标识：
  - DB19.DBX32.6 = 0（功能请求）
  - DB19.DBX36 = <故障标识>

14.13.2 程序选择

功能

借助 PLC/HMI 接口可选择预设程序/工件，以通过 NC 执行。

预设通过文件中的程序/工件记录实现，即所谓的 PLC 程序列表（*.ppl）。

前提条件

为了通过 HMI 使能任务处理，必须设置以下机床数据：

MD9106 $MM_SERVE_EXTCALL_PROGRAMS

为了激活区域专用的 PLC 程序列表，必须设置相应的机床数据，且至少须设置保护级的口令：

- 区域**用户（user）**
  - MD51041 $MN_ENABLE_PROGLIST_USER = 1
  - 保护级的口令：3（用户）
  - 程序列表：/user/sinumerik/hmi/plc/programlist/plc_proglist_user.ppl
- 区域 **制造商（oem）**
  - MD51043 $MN_ENABLE_PROGLIST_MANUFACT = 1
  - 保护级的口令：1（制造商）
  - 程序列表：/oem/sinumerik/hmi/plc/programlist/plc_proglist_manufacturer.ppl

程序列表的结构

程序列表是一个文本文件。其每行包含的信息如下：

<程序号> <程序路径><程序名称> [CH=<通道号>]

- 程序号
  程序列表中可使用的程序号取决于区域：
  - 用户（user）：1 - 100
  - 个人（oem_i）：101 - 200（仅适用于 SINUMERIK 828D）
  - 制造商（oem）：201 - 255
- 通道号
  通道号“CH=<通道号>”为可选设定。仅在 NC 有多条通道时才需要设定。

- 程序路径
  程序路径须以绝对方式完整设定。
  程序路径的设定参见：
  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”
- 通道号
  当 NC 中配置了超过一条通道时，才须设定通道号（1、2、3、...）。

用户程序列表的结构如下所示（仅节选作为示例）：

程序列表：plc_proglist_user.ppl

```
1 //DEV2:/MPFDir/PROG_01.MPF CH=1
2 //DEV2:/MPFDir/PROG_01.MPF CH=2
```

在程序列表中创建记录

程序列表（*.ppl）中的记录可直接在文件中编辑，或通过操作界面屏幕编辑。

- 通过**用户**区域的操作界面
  操作区域“程序管理器”>“ETC 键（“>”）”>“程序列表”
- 通过**制造商**区域的操作界面
  操作区域“调试”>“系统数据”>“ETC 键（“>”）”>“程序列表”

程序选择：任务接口

说明

通过 HMI 应答了前一个任务（DB19.DBB26 == 0）的情形下，才能通过 PLC 请求新任务。

程序列表

DB19.DBB16 = <程序列表的编号>

编号	程序列表
129	/user/sinumerik/hmi/plc/programlist/**plc_proglist_user.ppl**
130	/oem_i/sinumerik/hmi/plc/programlist/**plc_proglist_individual.ppl**（仅适用于 SINUMERIK 828D）
131	/oem/sinumerik/hmi/plc/programlist/**plc_proglist_manufacturer.ppl**

程序号

程序号基于所选程序列表中所包含的程序。

DB19.DBB17 = <程序号>

- “用户（user）”区域：1 - 100
- “个人（oem_i）”区域：101 - 200（仅适用于 SINUMERIK 828D）
- “制造商（oem）”区域：201 - 255

请求程序选择

DB19.DBX13.7 = 1

程序选择：应答接口

任务应答

- DB19.DBX26.7 == 1（选择已识别）
- DB19.DBX26.3 == 1（正在执行程序选择）
- DB19.DBX26.2 == 1（程序选择中出错，参见错误标识 DB19.DBB27）
- DB19.DBX26.1 == 1（任务已完成）

错误标识

DB19.DBB27 == <错误标识>

错误标识	
值	**含义**
0	无故障
1	程序列表编号（DB19.DBB16）无效
3	未找到用户专用程序列表 plc_proglist_main.ppl（仅在 DB19.DBB16 ≠ 1、2、3 时）
4	程序号（DB19.DBB17）无效
5	无法打开所选工件中的工作列表。
6	工作列表中有错误（工作列表编译器报告有错）。
7	工作列表编译器报告任务列表为空。

程序选择：任务流程

按照以下步骤执行程序选择任务：

1. 检查应答字节：DB19.DBB26 == 0
   若该应答字节不为 0，则表示上一个任务尚未完成。
2. 设定程序列表：DB19.DBB16
3. 设定程序编号：DB19.DBB17
4. 置位以请求程序选择：DB19.DBX13.7 = 1
5. 分析应答接口和故障接口：DB19.DBB26 和 DBB27
   只要 DB19.DBX26.3 == 1（生效），便表示任务在 HMI 侧尚未完成
   下面两个信号均置位时，任务在 HMI 侧完成：
   - DB19.DBX26.1 == 1（OK）
   - DB19.DBX26.2 == 1（Error）
6. 为了完成任务，必须复位程序选择请求：DB19.DBX13.7 = 0
7. HMI 通过复位应答字节提示可接受新任务：（DB19.DBB26 == 0）的情形下，才能通过 PLC 请求新任务。

参见

A2: 不同的 NC/PLC 接口信号与功能 (页 43)

14.13.3 激活“按键禁用”

可通过以下接口信号禁用操作面板的键盘和 HMI 上直接连接的键盘

- HMI 1：DB19.DBX0.2 = <值>
- HMI 2：DB19.DBX50.2 = <值>

值	含义
0	按键禁用不生效
1	按键禁用生效

14.13.4 操作区域编号

生效操作区域的编号显示在：DB19.DBB21

HMI 监视器激活时，生效操作区的编号不再显示在 DB19.DBB21 中，而是显示在用户配置的 HMI 监示器 (页 1028)区域。

操作区	编号
加工	201
参数	205
编程	203
程序管理器	202
诊断	204
调试	206

14.13.5 屏幕窗口号

当前屏幕窗口号显示在：DB19.DBW24

HMI 监视器激活时，当前屏幕窗口号不再显示在 DB19.DBW24 中，而是显示在用户配置的 HMI 监示器 (页 1028)区域。

屏幕窗口号区域

屏幕窗口号区域有：

- JOG、手动加工 (页 1017)
- 回参考点 (页 1023)
- MDA (页 1024)
- AUTO (页 1024)
- “参数”操作区 (页 1025)
- “程序”操作区 (页 1027)
- “程序管理器”操作区 (页 1027)
- “诊断”操作区 (页 1028)

14.13.5.1 屏幕窗口号：JOG、手动加工

运行方式 JOG

画面	编号
车削工艺	
用于所有可接收窗口的循环启动窗口	81
铣削工艺	
用于所有可接收窗口的循环启动窗口	3
车削/铣削工艺	
基本画面	19
T,S,M	2
设置零偏	21
定位	4
平面铣削	18
切削	80
用于所有用户屏幕窗口的循环启动窗口	91
一般设置	1
多通道功能设置	106
碰撞监测设置	107
测量记录设置	108
回转	60
所有 G 指令	100
实际值缩放 (MCS/WCS)	101
螺纹同步	102
退回	103
手轮	104
同步动作	105
车削工艺：工件零点	
工件零点（基本菜单）	30
用户屏幕窗口	31
用户屏幕窗口	34
用户屏幕窗口	35

画面	编号
用户屏幕窗口	36
用户屏幕窗口	37
用户屏幕窗口	38
用户屏幕窗口	40
测量边沿 Z	5
车削工艺：工件测量	
测量刀具（基本菜单）	50
手动 X 或用户屏幕窗口	51
手动 Y	71
手动 Z 或用户屏幕窗口	52
放大镜或用户屏幕窗口	53
用户屏幕窗口	54
用户屏幕窗口	55
校准测头 X 或用户屏幕窗口	56
校准测头 Z 或用户屏幕窗口	57
Z 中的长度自动设定	58
Y 中的长度自动设定	73
X 中的长度自动设定	59
铣削工艺：工件零点	
工件零点（基本菜单）	30
测量边沿 X	5
测量边沿 Y	22
测量边沿 Z	23
用户屏幕窗口	7
边对齐或用户屏幕窗口	31
2 个边沿的间距或用户屏幕窗口	32
直角	33
任意角或用户屏幕窗口	8
矩形腔	34
1 个钻孔或用户屏幕窗口	9
2 个孔	35

画面	编号
3 个孔	36
4 个孔	37
矩形凸台	38
1 个圆弧凸台或用户屏幕窗口	10
2 个圆弧凸台	39
3 个圆弧凸台	40
4 个圆弧凸台	41
校准平面	42
校准探头长度或用户屏幕窗口	11
校准探头半径	12
铣削工艺：测量工件	
测量刀具（基本菜单）	50
- 手动测量长度（用铣刀）或 - 手动测量 X 轴长度（用车刀）或 - 用户屏幕窗口	16
手动测量 Y 轴长度（用车刀）	74
手动测量 Z 轴长度（用车刀）	24
手动测量直径或用户屏幕窗口	17
- 自动测量长度（用铣刀）或 - 自动测量 X 轴长度（用车刀）或 - 用户屏幕窗口	13
自动测量 Y 轴长度（用车刀）	75
自动测量 Z 轴长度（用车刀）	25
自动测量直径或用户屏幕窗口	14
用户屏幕窗口	51
校准测头或用户屏幕窗口	15
校准固定点或用户屏幕窗口	52
RunMyScreens（仅当 JobShopIntegration 置位时）	
用于第 1 个水平软键的用户屏幕窗口	96
用于第 2 个水平软键的用户屏幕窗口	98
用于第 3 个水平软键的用户屏幕窗口	99
用于第 4 个水平软键的用户屏幕窗口	94

画面	编号
用于第 5 个水平软键的用户屏幕窗口	95
用于第 6 个水平软键的用户屏幕窗口	92
用于第 7 个水平软键的用户屏幕窗口	97
用于第 8 个水平软键的用户屏幕窗口	90
用于第 9 个水平软键的用户屏幕窗口	83
用于第 10 个水平软键的用户屏幕窗口	82
用于第 11 个水平软键的用户屏幕窗口	93
用于第 12 个水平软键的用户屏幕窗口	84
用于第 13 个水平软键的用户屏幕窗口	85
用于第 14 个水平软键的用户屏幕窗口	86
用于第 15 个水平软键的用户屏幕窗口	87
用于第 16 个水平软键的用户屏幕窗口	88

JOG、手动加工

DB19.DBB24	
画面	**画面编号**
车削/铣削工艺	
车削-锥	61
铣削-角	62
挡块	63
直线	1300
所有轴直线	1330
X alpha 直线	1340
Z alpha 直线	1350
圆弧	1360
钻削	1400
钻削-中心钻孔	1410
钻削-螺纹中心	1420
钻削-钻中心孔	1433
钻削-钻削	1434
钻削-铰孔	1435

DB19.DBB24	
画面	**画面编号**
钻削-镗孔	1436
钻削-深孔钻削	1440
钻削-深孔钻削 2	1441
钻削 – 螺纹钻削	1453
钻孔螺纹铣削	1455
位置	1473
序列位置	1474
栅格位置	1477
框架位置	1478
圆弧位置	1475
节距圆位置	1479
障碍物	1476
车削	1500
车削-切削 1	1513
车削-切削 2	1514
车削-切削 3	1515
车削-凹槽 1	1523
车削-凹槽 2	1524
车削-凹槽 3	1525
车削-E 形退刀槽	1533
车削-F 形退刀槽	1534
车削-螺纹 DIN 退刀槽	1535
车削-螺纹退刀槽	1536
车削-螺纹长度	1543
车削-螺纹锥	1544
车削-螺纹端面	1545
车削-螺纹链	1546
车削-断削	1550
铣削	1600
铣削-平面铣削	1610

DB19.DBB24	
画面	**画面编号**
铣削-矩形腔	1613
铣削-圆形腔	1614
铣削-矩形轴颈	1623
铣削-圆形轴颈	1624
铣削-长形腔	1633
铣削-圆形槽	1634
铣削-开口槽	1635
铣削-多边形	1640
铣削-螺纹铣削	1454
铣削-雕刻	1670
轮廓车削	1200
轮廓车削-新轮廓/上一个轮廓	1210
轮廓车削-轮廓切削	1220
轮廓车削-轮廓槽式车削	1230
轮廓车削-轮廓槽式车削	1240
轮廓铣削	1100
轮廓铣削-新轮廓/上一个轮廓	1110
轮廓铣削-轨迹铣削	1120
轮廓铣削-钻中心孔	1130
轮廓铣削-预钻削	1140
轮廓铣削-轮廓腔	1150
车削工艺：模拟	
侧视图	1740
正视图	1750
3D 视图	1760
2 窗口视图	1770
半剖视图	1780
车削工艺：同步记录	
侧视图	1741
正视图	1751

DB19.DBB24	
画面	**画面编号**
3D 视图	1761
2 窗口视图	1771
机床内部	1791
半剖视图	1781
铣削工艺：模拟	
顶视图	1742
3D 视图	1760
前视图	1744
后视图	1746
左视图	1748
右视图	1752
半剖视图	1780
车削视图	1782
铣削工艺：同步记录	
顶视图	1743
3D 视图	1761
前视图	1745
后视图	1747
左视图	1749
右视图	1753
机床内部	1791
半剖视图	1781
车削视图	1783

14.13.5.2 屏幕窗口号：回参考点

画面	**编号**
实际值缩放 MCS/WCS	101

14.13.5.3 屏幕窗口号：MDI

画面	编号
MDI	20
所有 G 指令	100
实际值缩放 MCS/WCS	101
手轮	104
同步动作	105
程序控制	210
设置	250

14.13.5.4 屏幕窗口号：AUTO

画面	编号
自动方式	200
溢出转存	202
程序控制	210
程序段搜索	220
一般设置	250
多通道功能设置	106
碰撞监测设置	107
所有 G 指令	100
实际值缩放 MCS/WCS	101
手轮	104
同步动作	105
车削工艺：同步记录	
侧视图	243
正视图	244
3D 视图	245
2 窗口视图	246
机床内部	247
半剖视图	253

画面	编号
铣削工艺：同步记录	
顶视图	242
3D 视图	244
前视图	248
后视图	249
左视图	251
右视图	252
机床内部	247
半剖视图	253
车削视图	254

14.13.5.5 屏幕窗口号：“参数”操作区

画面	编号
刀具表	600
刀具磨损	610
用户刀具列表	620
刀库	630
零点偏移	
零点偏移生效	642
零点偏移概述	643
基本零偏	644
零点偏移 G54 - G509	645
零点偏移生效、零点偏移概述、基本零偏或 G54 - G509 的详细信息	647
用户变量	
R 参数	650
全局 GUD 1 (SGUD)	660
全局 GUD 2 (MGUD)	661
全局 GUD 3 (UGUD)	662
全局 GUD 4	663
全局 GUD 5	664

画面	编号
全局 GUD 6	665
全局 GUD 7	666
全局 GUD 8	667
全局 GUD 9	668
通道 GUD 1 (SGUD)	690
通道 GUD 2 (MGUD)	691
通道 GUD 3 (UGUD)	692
通道 GUD 4	693
通道 GUD 5	694
通道 GUD 6	695
通道 GUD 7	696
通道 GUD 8	697
通道 GUD 9	698
局部 LUD	681
局部 LUD / PUD	684
设定数据	
工作区域限制	671
主轴数据	670
主轴卡盘数据	672
Ctrl-Energy	
Ctrl-Energy 基本菜单	6170
Ctrl-Energy 分析	6171
Ctrl-Energy 协议	6172
Ctrl-Energy 分析图形	6176
Ctrl-Energy 分析长期测量	6177
Ctrl-Energy 分析详细信息	6179
Ctrl-Energy 测量比较	6178

14.13.5.6 屏幕窗口号：“程序”操作区

画面	编号
车削工艺：模拟	
侧视图	413
正视图	414
3D 视图	415
2 窗口视图	416
半剖视图	423
铣削工艺：模拟	
顶视图	412
3D 视图	414
前视图	418
后视图	419
左视图	421
右视图	422
半剖视图	423
车削视图	424

14.13.5.7 屏幕窗口号：“程序管理器”操作区

画面	编号
NC 目录	300
本地驱动器	325
USB / 已设计的驱动器 1	330
已设计的驱动器 2	340
已设计的驱动器 3	350
已设计的驱动器 4	360
已设计的驱动器 5	383
已设计的驱动器 6	384
已设计的驱动器 7	385
已设计的驱动器 8	386

14.13.5.8 屏幕窗口号：“诊断”操作区

画面	编号
报警列表	500
信息	501
报警日志	502
NC/PLC 变量	503

14.13.6 HMI 监示器

功能

HMI 监示器是位于可自由选择数据块中的、长度为 8 字节的数据区域。在该数据区域中可由 HMI 向 PLC 用户程序提供以下数据：

- 操作区域编号 (页 1015)
- 屏幕窗口号 (页 1016)

参数设置

此数据区域通过以下显示机床数据设置：

MD9032 $MM_HMI_MONITOR = "String"

其中 "String" = "DB<DB 编号>.DBB<字节地址>"

说明

偶数字节地址

此数据区域必须从一个偶数字节地址开始。

数据区域的结构

字节	含义
EB n + 0	生效的 SINUMERIK 操作区
EB n + 1	预留
EB n + 2	当前屏幕窗口号
EB n + 3	
EB n + 4	预留

字节	含义
...	...
EB n + 7	预留

前提条件

HMI 监视器生效时，系统将不再处理以下 PLC/HMI 接口信号：

- DB19.DBB10（PLC 硬键）
- DB19.DBB21（生效的 SINUMERIK 操作区）
- DB19.DBW24（当前屏幕窗口号）

示例

假设

- 当前操作区：“加工”，编号：201
- 当前屏幕窗口：“自动方式基本画面”，编号：200
- PLC 数据区：DB60.DBB10

参数设置

- MD9032 $MM_HMI_MONITOR = "DB60.DBB10"

数据区域中的值

- DB60.DBB10：201
- DB60.DBW12：200

14.14 用于集成 PROFIBUS 上的驱动组件的 PLC 功能

14.14.1 一览

借助下面介绍的功能，可从**硬件 PLC** 的 PLC 用户程序持续循环读写集成 PROFIBUS 上的驱动组件的输入/输出数据。 此时必须遵循下列前提条件：

- 读写输入/输出数据须使用系统功能 **SFC14/SFC15**。
- 输入/输出数据的读写始终通过**整个**槽长度进行。

- 此功能只支持循环的、**非等距**的数据传输。
- 输出槽不得在 NC 侧被占用（例如驱动的输出槽）。

14.14.2 调试

前提条件

在 NC 侧调试此功能前，必须满足以下前提条件：

- NCU 的集成 PROFIBUS 上的驱动组件必须已通过 SIMATIC STEP 7，HW Config 完整配置。
- 配置已载入至 PLC。

NC 机床数据

必须将待循环传输的槽的起始地址输入以下机床数据：

- 输入槽：
  MD10520 $MN_PLCINTERN_LOGIC_ADDRESS_IN[<下标>] = <槽地址>
- 输出槽：
  MD10525 $MN_PLCINTERN_LOGIC_ADDRESS_OUT[<下标>] = <槽地址>

<下标>： 0, 1, 2, ... (最多槽数) - 1

<槽地址>：HW Config 中参数设置的槽地址

说明

数据的最大数量

所有需要循环输入和输出的数据的总和当前不得超出 **2048** 字节。

14.14.3 示例

确定槽地址

在 HW-Config 的工作站窗口中选择集成 PROFIBUS“PROFIBUS Integrated: DP master system (3)”上的 DP 从站“SINAMICS_Integrated”后，其 PROFIdrive 报文及对应的槽地址会显示在详细视图中。

- 报文 136：驱动
- 报文 391：控制单元
- 报文 **370**：电源

在 NC 机床中进行参数时所需的电源槽地址为：

- 输入槽：槽 31，地址 **6514**
- 输出槽：槽 32，地址 **6514**

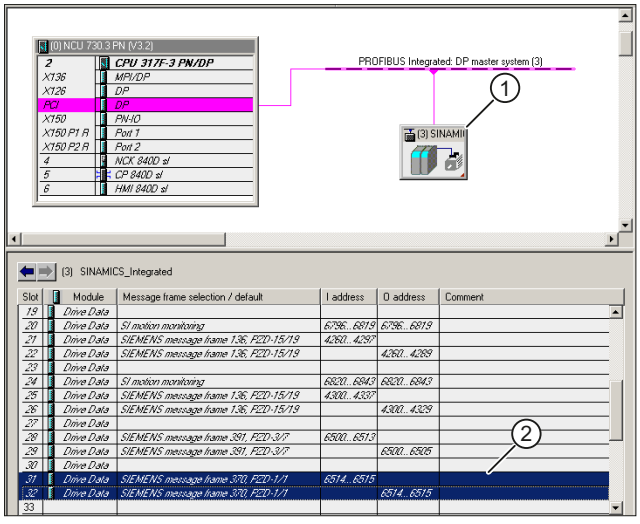


图 14-18 电源的 PROFIBUS 报文

设置 NC 机床数据

- 输入槽：
  MD10520 $MN_PLCINTERN_LOGIC_ADDRESS_IN[0] = 6514
- 输出槽：
  MD10525 $MN_PLCINTERN_LOGIC_ADDRESS_OUT[0] = 6514

通过 FB390 “ALM_Control” 控制 ALM

SINUMERIK 硬件 PLC 通过内部 PROFIBUS 与 SINAMICS S120 的 CU320 相连。 ALM 通过 Drive-CLIQ 与 CU 相连。

在 SINAMICS S120 中可为每个模块（CU、ALM、电机模块等）定义一个控制报文和一个状态报文。 若为 ALM 定义了这些报文，便可通过 PLC 用户程序接通或关闭该模块。

SIMATIC S7 模块“ALM_Control”会检查 ALM 的状态，并协助用户控制其通断。

可通过下面的 Industry Online Support（工业在线支持）链接获取对模块的说明，以及下载示例项目：

http://support.automation.siemens.com/WW/view/de/49515414

14.15 PLC 基本程序的存储空间需求

基本程序由基本功能和可选功能组成。**基本功能**包括 NC ↔ PLC 循环信号交换。**选件**例如包括 FC，其可根据需要使用。

下表中列出了基本功能和选件的存储空间需求。列出的值为近似值，实际值取决于当前的软件版本。

SINUMERIK 840D sl 中模块的存储空间需求			
模块 类型、编号	**功能**	**注释**	**模块大小（字节）**
	工作存储器		

基本程序基本功能		
FB1,FB15	必须载入至 / CF 卡	52182
FC2, 3, 5, 12	必须载入	470
DB4, 5, 7, 8	必须载入	1006
DB2, 3, 17	由基本程序生成	632
OB1, 40, 100, 82, 86	必须载入	398
	总和	55698

PLC / NC、PLC / HMI 接口			
DB10	PLC / NC 信号	必须载入	262
DB11	PLC / BAG 信号	由基本程序生成	56
DB19	PLC / HMI 信号	由基本程序生成	434
DB21 至 30	“PLC / 通道”信号	由基本程序根据 NC-MD 生成：每个 DB	416
DB31 至 61	“PLC / 轴、主轴”信号	由基本程序根据 NC-MD 生成：每个 DB	148

基本程序选件			
	机床控制面板		
FC19	传输 MCP 信号，M 规格	采用 M 规格的 MCP 时必须载入	92
FC25	传输 MCP 信号，T 规格	采用 T 规格的 MCP 时必须载入	92
FC24	传输 MCP 信号，窄型规格	采用窄型规格的 MCP 时必须载入	100
FC26	传输 MCP 信号，HT8 规格	采用 HT8 时必须载入	68
	手持操作设备		
FC13	HHU 显示控制	使用手持操作设备时可载入	144
	故障消息/运行消息		
FC10	FM/BM 采集	使用 FM / BM 时载入	66
	ASUB		
FC9	ASUB 启动	通过 PLC 使用 ASUB 时载入	128

基本程序选件			
	星形/三角形切换		
FC17	MSD 的星形/三角形切换	使用星形/三角形切换时载入	114
	主轴控制		
FC18	主轴控制	通过 PLC 控制主轴时载入	132
	PLC 通讯/NC 通讯		
FB2	读取 NC 变量	读取 NC 变量时载入	76

基本程序选件			
DBn	读取 NC 变量	一个实例 DB，每次 FB2 调用	270
FB3	写入 NC 变量	写入 NC 变量时载入	76
DBm	写入 NC 变量	一个实例 DB，每次 FB3 调用	270
FB4	PI 服务	PI 服务时载入	76
DBo	PI 服务	一个实例 DB，每次 FB4 调用	130
DB16	PI 服务描述	PI 服务时载入	618
FB5	读取 GUD 变量	PI 服务时载入	76
DBp	读取 GUD 变量	一个实例 DB，每次 FB5 调用	166
DB15	一般通讯	用于通讯的全局 DB	146
FB7	PI 服务 2	PI 服务时载入	76
DBo	PI 服务 2	一个实例 DB，每次 FB4 调用：每次	144
FC21	传输	DualPortRAM 时调用	164
	M : N		
FB9	M : N 切换	M : N 时载入	58
	Safety Integrated		
FB10	安全继电器	使用 Safety 选件时载入	74
FB11	制动测试	使用 Safety 选件时载入	76
DB18	Safety 数据	用于 Safety 的 DB	308
	刀具管理		
FC7	传输功能 刀塔	使用刀具管理选件时载入	84
FC8	传输功能	使用刀具管理选件时载入	132
FC22	方向选择	需要方向选择时载入	138
DB71	装刀位置	由基本程序根据 NC-MD 生成	40+30*B
DB72	主轴	由基本程序根据 NC-MD 生成	40+48*Sp
DB73	刀塔	由基本程序根据 NC-MD 生成	40+44*R
DB74	基本功能	由基本程序根据 NC-MD 生成	100+(B+Sp+R)*22

基本程序选件			
	编译循环		
DB9	NC 编译循环的接口	由基本程序根据 NC 选件生成	436

示例：

根据上表中列出的存储空间需求，为两个样本配置计算存储空间需求（见下表）。

模块类型、编号	**功能**	**注释**	**模块大小（字节）**
	工作存储器		
最小配置（1 根主轴、2 根轴和 T-MCP）			
见上	基本程序		54688
	接口数据块		1612
	MCP		92
		总和	56392

模块类型、编号	**功能**	**注释**	**模块大小（字节）**
	工作存储器		
最大配置（2 个通道、4 根主轴、4 根轴、T-MCP）			
见上	基本程序		54688
见上	接口数据块		2768
见上	MCP		92
见上	故障消息/运行消息		66
见上	ASUB	触发 1 个 ASUB	128
见上	并行轴	用于 2 个刀塔	132
见上	PLC 通讯/NC 通讯	读取 1 x 变量和写入 1 x 变量	838
见上	刀具管理	2 个刀塔，每个 1 装刀位置	674

最大配置（2 个通道、4 根主轴、4 根轴、T-MCP）		
见上	编译循环	436
	总和	59822

14.16 前提条件和 NC-VAR-Selector

14.16.1 前提条件

14.16.1.1 编程工具和参数设置工具

硬件

对于 SINUMERIK 840D sl 中使用的 PLC，编程设备或 PC 的配置须满足以下要求：

	最低配置	推荐配置
处理器	奔腾（Pentium）	奔腾（Pentium）
RAM（MB）	256	512 或更高
硬盘， 自由存储空间（MB）	500	> 500
接口	MPI、以太网（含电缆） 存储卡	
图像	SVGA（1024*768）	
鼠标	配备	
操作系统	Windows 2000 /XP Professional，STEP 7 版本 5.3 SP2 或更高	

在符合上述前提条件的设备上可安装所需的 **STEP 7** 软件（前提是该软件未包含在 PG 的供货范围中）。

通过此软件包可实现下列功能：

- 编程
  - STL（完整的语言范围，包括 SFB-/SFC 调用）、LAD、FBD 的编辑器和编译器
  - 创建和处理指定关系列表（符号编辑器）
  - DB 编辑器
  - 在线/离线输入和输出模块
  - 在线/离线插入修改和补充
  - 从 PG 向 PLC 传输模块，以及反向传输
- 参数设置
  - 参数设置工具 **HW Config**，用于设置 CPU 和 I/O
  - 参数设置工具 **NetPro**，用于设置 CPU 通讯参数
  - 输出系统数据，如硬件版本、软件版本、存储器扩展、I/O 扩展、I/O 分配
- 测试和诊断（在线）
  - 状态/控制变量（输入/输出、标志位、DB 内容等）
  - 各模块的状态
  - 显示系统状态（USTACK、BSTACK、SZL）
  - 显示系统消息
  - 通过 PG 触发 PLC 停止/重启/总复位
  - 压缩 PLC
- 文档
  - 对单个或所有模块的描述
  - 分配符号名称（也适用于 DB 中的变量）
  - 输入和输出每个模块内的注释
  - 对测试的描述和诊断显示
  - 硬拷贝（Hardcopy）功能
  - 交叉参考列表
  - 程序概览
  - I/O/M/T/Z/D 分配图

- 实用程序的存档
  - 分配各模块的输出状态
  - 模块对比
  - 重新布线
  - STEP 5 → STEP 7 转换器
- 选件包
  - 以 S7–HIGRAPH、S7–GRAPH、SCL 编程
    这些选件包可通过 SIMATIC 经销商订购。
  - 用于模块配置的附加包（例如 CP3425 → NCM 包）

说明

可实现功能的更多详细信息请参见 SIMATIC 产品样本和 STEP 7 文档。

14.16.1.2 必需的 SIMATIC 文档

文档：

- SIMATIC S7 系统说明
- S7-300 操作列表
- 使用 STEP 7 编程
- STEP 7 用户手册
- STEP 7 编程手册；用户程序的设计
- STEP 7 参考手册；STL 指令列表
- STEP 7 参考手册；梯形图 LAD
- STEP 7 参考手册；标准功能和系统功能
- STEP 7 手册：STEP 5 程序的转换
- STEP 7 总索引
- CPU 317-2DP 手册

14.16.1.3 相关的 SINUMERIK 文档

文档：

- 调试手册 IBN CNC：NC, PLC, 驱动
- 设备手册，操作组件与互联

- 功能手册之基本功能分册
- 功能手册之扩展功能分册
- 功能手册之特殊功能分册
- 参数手册 sl（第 1 册）
- 参数手册 sl（第 2 册）

14.16.2 NC-VAR-Selector

14.16.2.1 概述

简介

PC 应用程序“NC-VAR-Selector”用于获取所需 NC 变量的地址，以及处理该地址用于 PLC 程序中（FB2 / FB3）的访问。借助此程序，PLC 编程人员可从 NC 和驱动变量总列表中选择 NC 和驱动变量、保存变量选择并通过 STEP 7 编译器的代码生成器预处理，从而将其作为 ASCII（*.AWL）文件保存在机床 CPU 程序中。下面的“NC-VAR-Selector”一图显示了该处理的过程。

为了保存由 NC-VAR-Selector 创建的文件，必须通过 Windows Explorer 以任意名称创建一个目录。选择的 NC-VAR-Selector 数据会被保存至该目录（Daten.VAR 和 Daten.AWL 文件）。之后必须通过菜单项 “代码” → “至 STEP 7 项目”传输和编译 AWL 文件。或者须通过 STEP 7 管理器将“Daten.AWL”文件的“外部源”“插入”至 STEP 7 机床项目。为此必须在管理器中选择源容器。这样一来，该文件被保存至项目结构。文件传输完成后，必须通过 STEP 7 编译这些 AWL 文件。

说明

对每个 NC 软件版本（包括较旧的版本）均可使用最新的 NC-VAR-Selector。对于较旧的 NC 软件版本，同样可以从最新的总列表选择变量。DB120（变量的标准 DB）的信息内容与软件版本无关。也就是说，在较旧的软件版本中选择的变量不必在软件升级时重新选择。

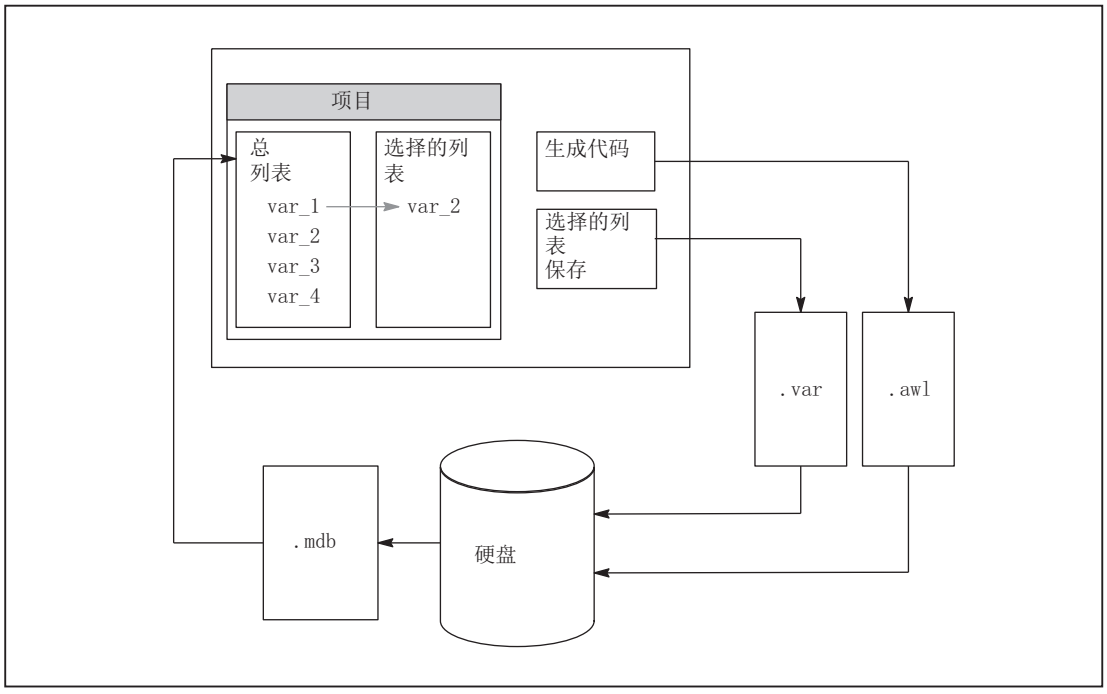


图 14-19　NC-VAR-Selector

应用程序“NC-VAR-Selector”启动后，选择对应 NC 类型的变量列表（硬盘 → Ncv.mdb 文件），该列表中提供的所有变量均显示在一个窗口中。

可供选择的变量列表 ncv*.mdb 如下：

变量	列表
NC 变量，包括机床数据和设定数据：	ncv_NcData.mdb
驱动的参数：	ncv_SinamicsServo.mdb

操作人员可将变量接收至另一个列表（其他窗口）。选择的变量之后可保存在 ASCII 文件中，以及作为 STEP 7 源文件（.awl）预处理和保存。

借助生成的 STEP 7 文件，PLC 编程人员能够在 PLC 数据块生成后使用 STEP 7 编译器通过基本程序功能块“PUT”和“GET”读取或写入 NC 变量。

选择的变量的列表同样作为 ASCII 文件保存（文件扩展名为 .var）。

随“NC-VAR-Selector”提供的变量表对应当前的 NC 版本。该列表中不含用户定义的变量（GUD 变量）。这些变量由基本程序通过功能块 FB5 处理。

说明

最新版本的“NC-VAR-Selector”适用于之前的所有 NC 软件版本。因此无需基于 NC 软件版本安装特定版本的“NC-VAR-Selector”。

系统特性、前提条件

使用 PC 应用程序“NC-VAR-Selector”的前提条件是 Windows 2000 或更高版本的操作系统。

对变量的名称设定的说明请参见：

文档：

参数列表 sl（第 1 册）；章节：变量

或参见变量的帮助文件（集成在 NC-VAR-Selector 中）

14.16.2.2 功能说明

概述

下图显示了 NC-VAR-Selector 在 STEP 7 环境中的应用范围。

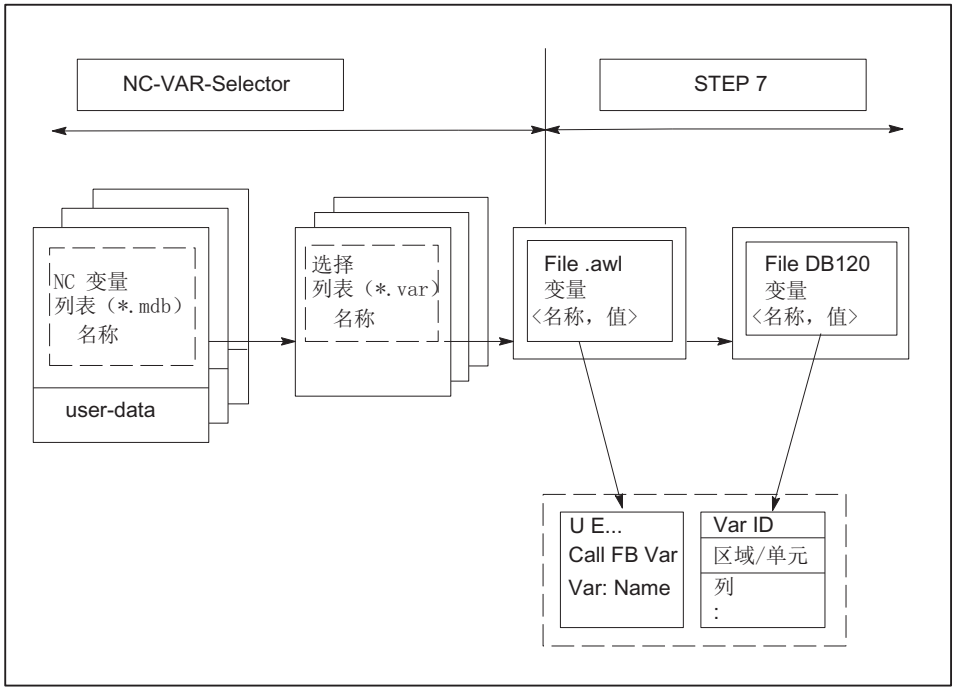


图 14-20　在 STEP 7 环境中使用 NC-VAR-Selector

NC-VAR-Selector 用于从一个变量列表选择变量并创建列表，并于随后生成一个可由 STEP 7 编译器编译的 **.awl** 文件。

- *.awl 文件既包含名称或 ALIAS 名称，也包含 NC 变量地址参数的信息。 由其生成的数据块则只包含地址参数（每个参数 10 字节）。
- 生成的数据块始终须按照 STEP 7 定义保存在机床专用的文件存储区。
- 为了确保功能块 GET/PUT（FB2/3）的参数设置能够基于 NC 地址以符号方式进行，生成的数据块的可自由分配符号名称必须接收至 STEP 7 符号列表。

基本画面/主菜单

选择（启动）NC-VAR-Selector 后将显示含所有操作选项（上方菜单栏）的基本画面。 显示的其他窗口均会置于主窗口内。

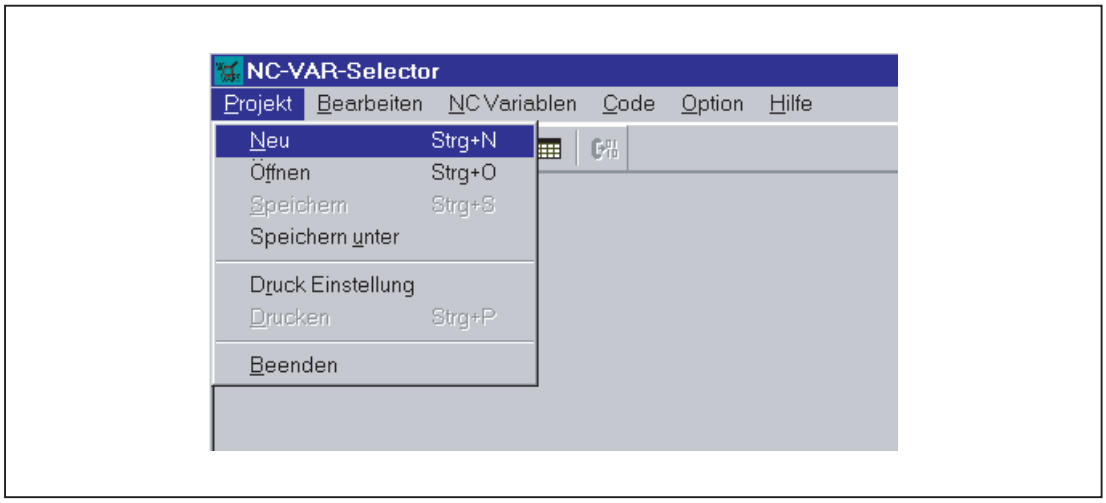


图 14-21　基本画面及主菜单

“项目”菜单

所有与项目文件（所选变量的文件）相关的操作均通过此菜单项执行。

退出应用程序

在“项目”菜单下选择“退出”关闭应用程序。

创建新项目

在“项目”菜单下创建一个新项目（所选变量的新文件）。

选择“新建”后，将弹出一个显示已选择变量的窗口。 确认弹出提示后，可选择 NC 变量列表文件（仅在 NC 变量列表尚未打开时）。

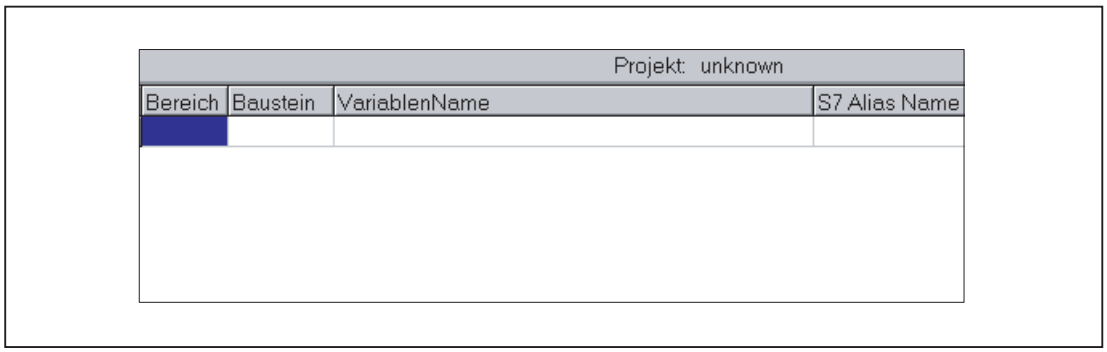


图 14-22　新建项目时显示已选择变量的窗口

所选择的变量在一个窗口中显示。

打开已存在的项目

在“项目”菜单下，可选择“打开”来打开一个已存在的项目（已选择的变量）。 此时会显示一个文件选择窗口，可从中选择扩展名为“.var”的对应项目。

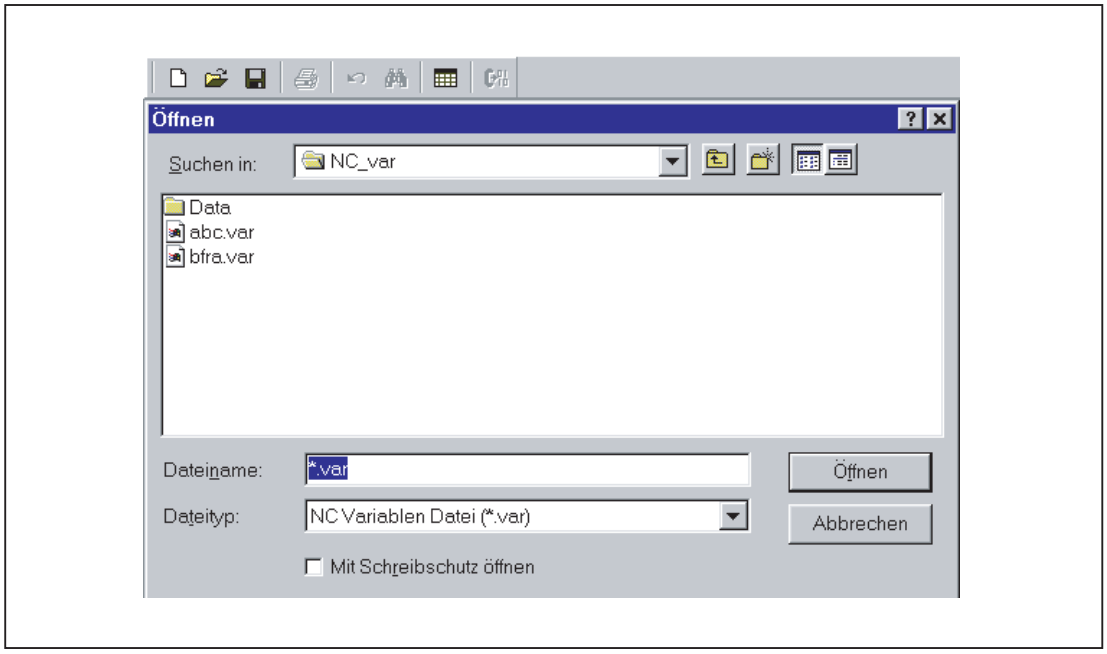


图 14-23　已存在项目的选择窗口

选择项目后，若需添加新变量，必须重新选择 NC 变量的总列表。 若只需删除变量，则无需选择总列表。

保存项目

通过菜单“项目”>“保存”或“另存为...”保存变量列表。

使用“保存”时，变量列表会被保存至已知的路径下。 若未设定项目路径，那么此功能的特性将与“另存为...”相同。

选择“另存为...”时，软件将弹出一个窗口用于设定项目的存储路径。

打印项目

在“项目”菜单下可选择“打印”来打印项目文件。 每页的行数通过菜单项“打印设施”定义。 缺省设置为 77 行。

“编辑”菜单

通过此菜单项例如可直接执行下列操作：

- 接收变量
- 删除变量
- 修改 Alias 名称
- 搜索变量

此外还可撤销这些操作。

撤销

与项目文件创建相关的操作（接收变量、删除变量、修改 Alias 名称）可通过此选项撤销。

“NC 变量”菜单

所有变量的基本列表保存在 NC-Var-Selector 路径 Data\Swxy 下（xy 对应软件版本编号，例如 SW 5.3:=xy=53）。 此列表可作为 NC 变量列表选择。 对于 SINUMERIK 840D sl，基本列表保存在 Data\Swxy_sl 路径下。

选择 NC 变量列表

通过“NC 变量列表”、“选择”可选择和显示对应 NC 版本的 NC 变量列表。

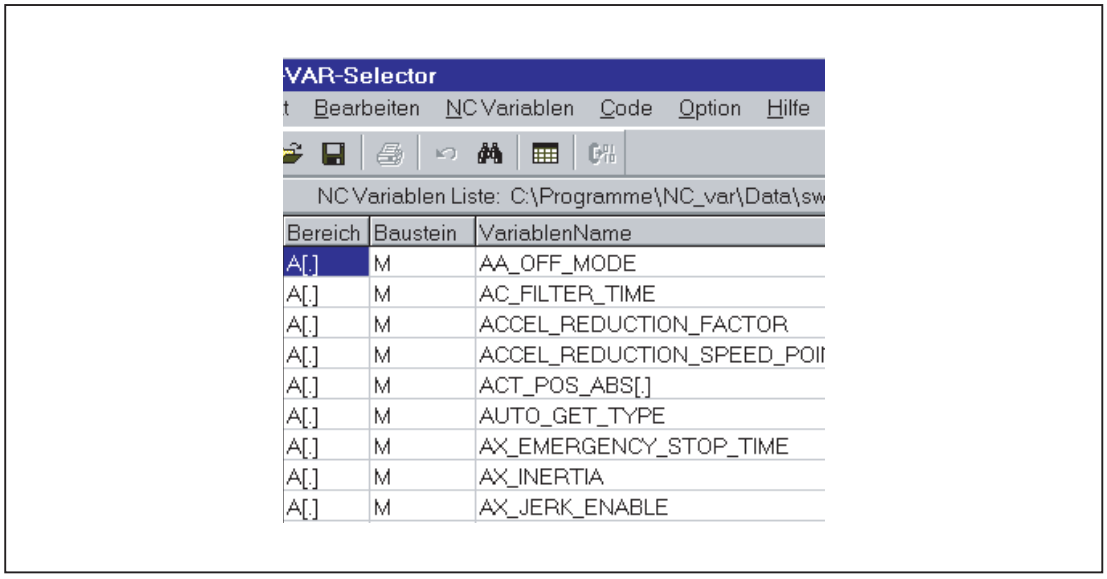


图 14-24　选择了总列表时的窗口

数组变量（例如轴区域、T 区域数据等）以中括号 [.] 标注。 此处必须设定附加信息。 在将这些变量添加至项目列表时，软件会询问所需的附加信息。

显示子集

双击任意表格栏（特例：变量栏！）后会弹出一个窗口，可在其中设定筛选标准。

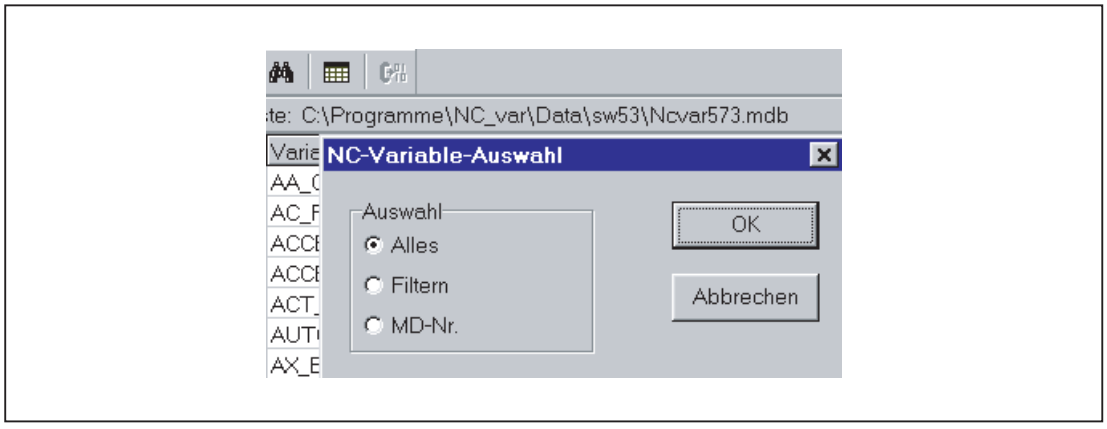


图 14-25　用于定义变量列表显示筛选标准的窗口

有三个选项可供选择：

- 显示所有
- 设定区域、模块和名称（也可组合）
- 显示 MD/SE 数据编号

也可使用以下通配符：

*	用于扩展搜索标准，对应任意长度

搜索标准的示例

搜索标准名称：CHAN*　　搜索：　　CHAN_NAME
chanAlarm
chanStatus
channelName
chanAssignment

- 选择变量
  变量通过单击鼠标选取，并通过双击接收至已选变量的窗口。 在“编辑”菜单下还可撤消此操作。

Alias 名称

设定的变量名称长度最多可为 32 字符。 为了保证待生成数据块中变量的唯一性，所选择的名称还将扩展数个 ASCII 字符。 但是 STEP 7 编译器只会将最多 24 个 ASCII 字符作为 S7 变量识别。 不能排除变量名称只在最后 8 位有区别的情形，因此对过长的名称需使用 **Alias** 名称。 选择变量时，软件会检查待使用的 S7 名称的长度。 若其超出 24 个字符，则须由用户额外输入一个名称（Alias 名称）。

此时用户须确保该名称的唯一性。

Alias 输入可由用户通过“选项”菜单设定为长期激活。
之后在每次接收变量时都可采用 Alias 输入。

此外还可双击 S7 变量名称栏对该名称进行编辑。 在“编辑”菜单下还可撤消此操作。

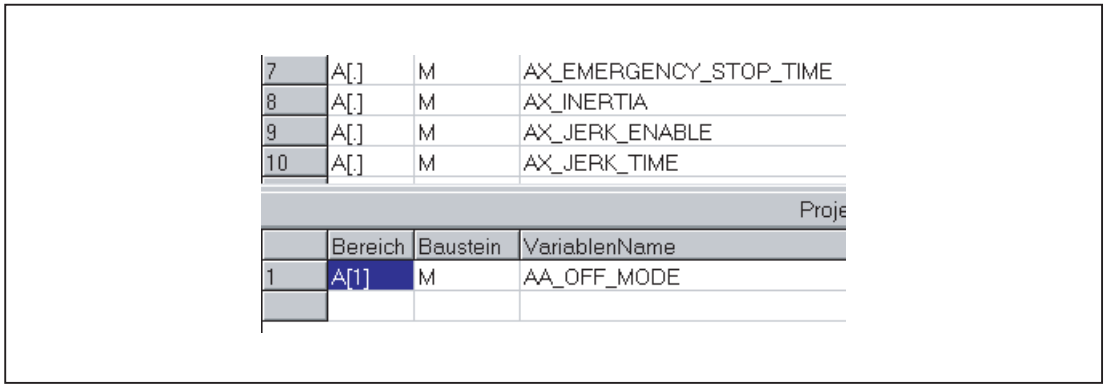


图 14-26　包含总列表和已选择变量的画面

滚动

若窗口中无法显示所有变量，可滚动显示。 借助滚动（Page-Up/Down）便可查看余下的变量。

多维结构中的变量

若从多维结构选择变量，则须输入列编号和/或行编号以及区域编号来实现该变量的定址。 所需编号可从 NC 变量文档获取。

文档：
参数手册 sl（第 1 册）；变量

为区域编号、行索引或列索引输入零（0）时，可使用 S7 PLC 中的变量作为对这些数据的指针。 在通过“PUT”和“GET”功能读写这些变量时，必须为可选参数“UnitX”、“ColumnX”和“LineX”填写必要信息。

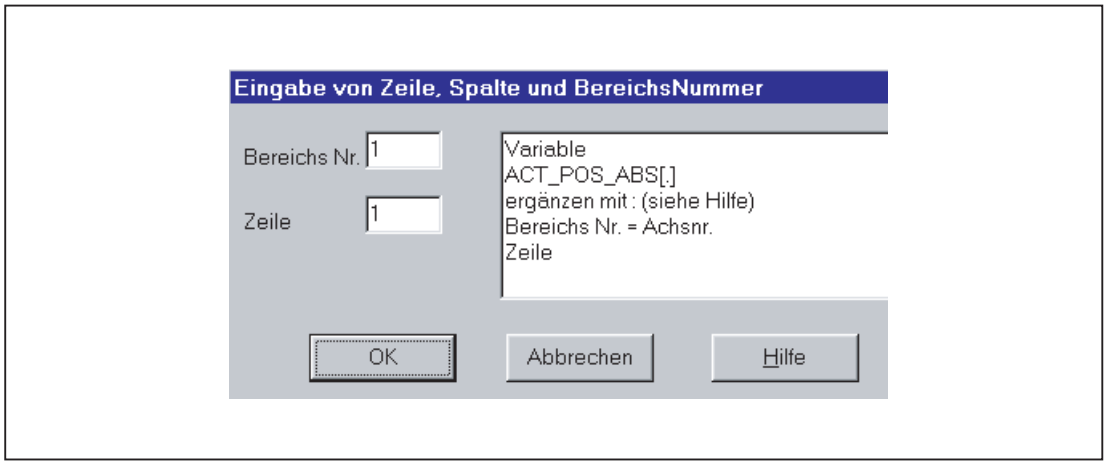


图 14-27　行、列和模块编号的输入栏

删除变量

在已选择变量窗口中如下删除变量：单击鼠标选中变量，之后按下“删除”键执行删除。双击不会触发动作。 可选择多个变量进行删除（参见“搜索标准的示例 > 选择变量”部分）。

在“编辑”菜单下还可撤消此操作。

说明

删除变量会触发对指向变量的指针结构的绝对地址修改。 因此在修改变量选择时，必须在**修改前生成**一个或多个**含所有用户模块的文本文件**。 只有这样才能确保重新编译后 FB“GET”或 FB“PUT”中的指定关系仍一致。

保存选择的列表

在完成变量选择后，可将变量保存在一个项目名称下。 文件的保存针对项目进行。

对于需要保存的文件，软件会弹出一个窗口用于设定项目路径和文件名称。

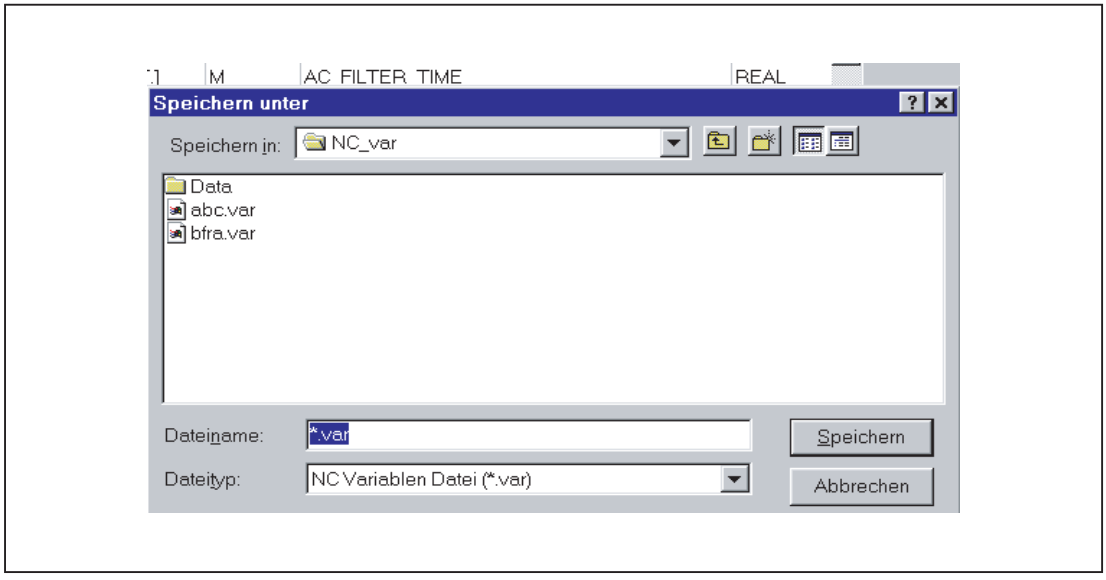


图 14-28　项目路径和待存储文件名称的设定窗口

代码生成

此菜单下有三个子菜单项可供选择：

1. 设置（设定待生成的数据块编号）和其他设置
2. 生成（生成数据块）
3. 接收至 STEP 7 项目（将数据块接收至 STEP 7 项目）

设置

通过此菜单输入用于生成代码的 DB 编号和这些 DB 编号的符号。

在“尺寸系统”标签下，可设定尺寸系统相关的变量在 PLC 中如何计算。

在“生成”标签下可为相应的目标系统定义生成。

生成

通过此菜单从已选择变量列表对扩展名为“.awl”的 STEP 7 文件进行设置。

按下“选择”时会生成一个文件：

一个 **.awl** 文件，可用作 STEP 7 编译器的输入。

对于需要保存的文件，软件会弹出一个窗口用于设定待生成 .awl 文件的路径和名称。

接收至 STEP 7 项目

生成的 AWL 文件会被传输至一个可选择的 SIMATIC 项目（程序路径）并进行编译。 此外也可接收符号。 STEP 7 自版本 5.1 起支持此功能。 由于需要调用 STEP 7，因此接收会持续较长时间。在传输新的 AWL 文件前必须在 LAD/FBD/STL 编辑器中关闭 AWL 文件的窗口。

“选项”菜单

在“选项”菜单下可选择：

- 当前语言
- Alias 输入的模式（始终启用 / > 24 字符）

“帮助”菜单

可通过选择相应菜单项读取以下信息：

- 操作说明
- 变量说明

此外此处还显示版权和版本编号。

14.16.2.3 调试、安装

Windows 应用程序“NC-VAR-Selector”通过随附的 SETUP 程序安装。

14.17 模块说明

14.17.1 FB1：RUN_UP - 基本程序、启动部分

功能

启动期间会进行 NC 和 PLC 之间的同步。此时系统会基于通过机床数据定义的 NC 配置生成用于用户接口 NC/PLC 的数据块，并检查最重要的基本程序参数是否合理。存在故障时，FB1 会将故障识别传输至诊断缓存，并使 PLC 切换至停止状态。

启动方式“重新启动”

集成 PLC 只支持“重新启动”。此时，在系统初始化之后，必须先执行组织块 OB100 “重新启动”，然后再执行组织块 OB1 “循环运行”。

输入参数

设置基本程序的参数时，只能用用户专用值对功能块 FB1 的各个相关参数进行说明。FB1 的实例数据块 DB7 中的预设值不需要指定。功能块 FB1 只允许在组织块 OB100 中调用。

输出参数

功能块 FB1 的输出参数也可由基本程序的循环程序部分读取。有两种方法可以选择：

1. 以符号形式直接访问 FB1 的实例数据块 DB7。
   示例：“L gp_par.MaxChan”，其中 “gp_par” 用作 DB7 的符号名称
2. 在对 FB1 进行参数设置时，系统会为输出参数指定一个标志位。该标志位之后会在基本程序中读取，以确定输出参数的值。
   示例：“MaxChan”：= MW 20

说明

MCP 和 HHU 的 FB1 参数的布局参见“机床控制面板、手持操作设备、直接控制按键的配置 (页 995)”。

功能说明

```
FUNCTION_BLOCK FB1
VAR_INPUT
   MCPNum :          INT:=1;              // 0：无 MCP
                                          //1：1 个 MCP（缺省设置）
                                          //2：2 MCP
   MCP1In :          POINTER;             //起始地址输入信号 MCP1
   MCP1Out :         POINTER;             //起始地址输出信号 MCP1
   MCP1StatSend :    POINTER;             //MCP1 发送的状态 DW
   MCP1StatRec :     POINTER;             //MCP1 接收的状态 DW
   MCP1BusAdr :      INT:=6;              //缺省
   MCP1Timeout :     S5TIME:=S5T#700MS;
   MCP1Cycl :        S5TIME:=S5T#200MS;
   MCP2In :          POINTER;             //起始地址输入信号 MCP2
   MCP2Out :         POINTER;             //起始地址输出信号 MCP2
   MCP2StatSend :    POINTER;             //MCP2 发送的状态 DW
   MCP2StatRec :     POINTER;             //MCP2 接收的状态 DW
   MCP2BusAdr :      INT;
   MCP2Timeout :     S5TIME:=S5T#700MS;
   MCP2Cycl :        S5TIME:=S5T#200MS;
   MCPMPI :          BOOL:=FALSE;
   MCP1Stop :        BOOL:=FALSE;
   MCP2Stop :        BOOL:=FALSE;
   MCP1NotSend :     BOOL:=FALSE;
   MCP2NotSend :     BOOL:=FALSE;
   MCPSDB210 :       BOOL:=FALSE;
   MCPCopyDB77 :     BOOL:=FALSE;
   MCPBusType :      BYTE=B#16#0;
   BHG :             INT:=0;              //手持操作设备接口
                                          // 0：无 HHU
                                          // 1：HHU 连接至 MPI
                                          // 2：HHU 连接至 OPI
   BHGIn :           POINTER;             //手持操作设备的发送数据
   BHGOut :          POINTER;             //手持操作设备的接收数据
   BHGStatSend :     POINTER;             //HHU 发送的状态 DW
```

```
BHGStatRec :       POINTER;              //HHU 接收的状态 DW
BHGInLen :         BYTE:=B#16#6;         //输入 6 字节
BHGOutLen :        BYTE:=B#16#14;        //输出 20 字节
BHGTimeout :       S5TIME:=S5T#700MS;
BHGCycl :          S5TIME:=S5T#100MS;
BHGRecGDNo :       INT:=2;
BHGRecGBZNo :      INT:=2;
BHGRecObjNo :      INT:=1;
BHGSendGDNo :      INT:=2;
BHGSendGBZNo :     INT:=1;
BHGSendObjNo :     INT:=1;
BHGMPI :           BOOL:=FALSE;
BHGStop :          BOOL:=FALSE;
BHGNotSend :       BOOL:=FALSE;
NCCyclTimeout :    S5TIME:=S5T#200MS;
NCRunupTimeout :   S5TIME:=S5T#50S;
ListMDecGrp :      INT:=0;
NCKomm :           BOOL:=FALSE;
MMCToIF :          BOOL:=TRUE;
HWheelMMC :        BOOL:=TRUE;           //通过 HMI 选择手轮
ExtendAlMsg :      BOOL;
MCP_IF_TCS:        BOOL;
ExtendChanAxMsg:   BOOL;
MsgUser :          INT:=10;              //DB2 中用户区域数量
UserIR :           BOOL:=FALSE;          //OB40 中用户程序，
                                         //注意本地数据扩展！
IRAuxfuT :         BOOL:=FALSE;          //检测 OB40 中 T 功能
IRAuxfuH :         BOOL:=FALSE;          //检测 OB40 中 H 功能
IRAuxfuE :         BOOL:=FALSE;          //检测 OB40 中 DL 功能
UserVersion :      POINTER;              //指针，指向在操作界面的版本画面中显示的
                                         字符串变量
OpKeyNum :         INT;
Op1KeyIn :         POINTER;
Op1KeyOut :        POINTER;
Op1KeyBusAdr :     INT;
Op2KeyIn :         POINTER;
Op2KeyOut :        POINTER;
Op2KeyBusAdr :     INT;
Op1KeyStop :       BOOL;
Op2KeyStop :       BOOL;
Op1KeyNotSend :    BOOL;
Op2KeyNotSend :    BOOL;
```

```
   OpKeyBusType :     BYTE;
   IdentMcpBusAdr :   INT;
   IdentMcpProfilNo :BYTE;
   IdentMcpBusType :  BYTE;
   IdentMcpStrobe :   BOOL;
END_VAR

VAR_OUTPUT
   MaxBAG :           INT;
   MaxChan :          INT;
   MaxAxis :          INT;
   ActivChan :        ARRAY[1..10] OF
                      BOOL;
   ActivAxis :        ARRAY[1..31] OF
                      BOOL;
   UDInt :            INT;
   UDHex :            INT;
   UDReal :           INT;
   IdentMcpType :     BYTE;
   IdentMcpLengthIn :BYTE;
   IdentMcpLengthOut  BYTE;
   :
END_VAR
```

对形式参数的说明

信号	方式	类型	取值范围	含义
MCPNum：	E	INT	0, 1, 2	生效 MCP 数量 0：无 MCP 存在
MCP1In： MCP2In：	E	POINTER	E0.0 ... E120.0 或 M0.0 ... M248.0 或 DBn DBX0.0 ... DBXm.0	相关机床控制面板的输入信号的机床控制面板
MCP1Out： MCP2Out：	E	POINTER	A0.0 ... A120.0 或 M0.0 ... M248.0 或 DBn DBX0.0 ... DBXm.0	相关机床控制面板的输出信号的机床控制面板

信号	方式	类型	取值范围	含义
MCP1StatSend： MCP2StatSend：	E	POINTER	A0.0 ... A124.0 或 M0.0 ... M252.0 或 DBn DBX0.0 ... DBXm.0	只适用于以太网 MCP： 取消闪烁（参见章节"机床控制面板、手持操作设备的切换 (页 1006)"）
MCP1StatRec： MCP2StatRec：	E	POINTER	A0.0 ... A124.0 或 M0.0 ... M252.0 或 DBn DBX0.0 ... DBXm.0	目前无含义
MCP1BusAdr： MCP2BusAdr：	E	INT	1, 2, 3 ... 126	DP 从站：PROFIBUS 地址
			192, 193, 194 ..223	以太网 MCP：DIP 设置
MCP1Timeout： MCP2Timeout：	E	S5time	建议：700 ms	对机床控制面板的循环生命符号监控
MCP1Cycl： MCP2Cycl：	E	S5time	建议：200 ms	仅针对 PROFIBUS
MCPMPI：	E	BOOL	FALSE	由于兼容性原因存在
MCP1Stop： MCP2Stop：	E	BOOL	0 (FALSE), 1 (TRUE)	0：启动机床控制面板信号的传输 1：停止机床控制面板信号的传输
				DP 从站：从站取消
MCP1NotSend： MCP2NotSend：	E	BOOL	0 (FALSE), 1 (TRUE)	0：发送及接收运行激活 1：只接收机床控制面板信号
MCPSDB210：	E	BOOL	false	由于兼容性原因存在
MCPCopyDB77：	E	BOOL	false	由于兼容性原因存在

信号	方式	类型	取值范围	含义
MCPBusType：	E	BYTE	3, 4, 5, 6	b#16#yx： • MCP1 的总线类型：下部半位元组 (x) • MCP2 的总线类型：上部半位元组 (y) 3：PROFIBUS 4：MPI/DP 连接的 PROFIBUS 5：以太网 6：PROFINET 可采用混合运行（参见章节“机床控制面板、手持操作设备、直接控制按键的配置 (页 995)”）
BHG：	E	INT	0, 5	手持操作设备接口： 0：无 HHU 5：HHU 连接至以太网
BHGIn：	E	POINTER	E0.0 ... E124.0 或 M0.0 ... M252.0 或 DBn DBX0.0 ... DBXm.0	PLC 从手持操作设备接收数据的起始地址
BHGOut：	E	POINTER	A0.0 ... A124.0 或 M0.0 ... M252.0 或 DBn DBX0.0 ... DBXm.0	PLC 向手持操作设备发送数据的起始地址
BHGStatSend：	E	POINTER	A0.0 ... A124.0 或 M0.0 ... M252.0 或 DBn DBX0.0 ... DBXm.0	由于兼容性原因存在
BHGStatRec：	E	POINTER	A0.0 ... A124.0 或 M0.0 ... M252.0 或 DBn DBX0.0 ... DBXm.0	由于兼容性原因存在

信号	方式	类型	取值范围	含义
BHGInLen：	E	BYTE	HHU 缺省： B#16#6（6 字节）	由于兼容性原因存在
BHGOutLen：	E	BYTE	HHU 缺省： B#16#14（20 字节）	由于兼容性原因存在
BHGTimeout：	E	S5time	建议：700 ms	由于兼容性原因存在
BHGCycl：	E	S5time	建议：100 ms	由于兼容性原因存在
BHGRecGDNo：	E	INT	HHU 缺省：2	以太网 DIP 开关
BHGRecGBZNo：	E	INT	HHU 缺省：2	由于兼容性原因存在
BHGRecObjNo：	E	INT	HHU 缺省：1	由于兼容性原因存在
BHGSendGDNo：	E	INT	HHU 缺省：2	由于兼容性原因存在
BHGSendGBZNo：	E	Int	HHU 缺省：1	由于兼容性原因存在
BHGSendObjNo：	E	INT	HHU 缺省：1	由于兼容性原因存在
BHGMPI：	E	BOOL	0 (FALSE)	由于兼容性原因存在
BHGStop：	E	BOOL	0 (FALSE), 1 (TRUE)	0：启动手持操作设备信号的传输 1：停止手持操作设备信号的传输
BHGNotSend：	E	BOOL	0 (FALSE), 1 (TRUE)	0：发送及接收运行激活 1：只接收手持操作设备信号
NCCyclTimeout：	E	S5time	建议：200 ms	NC 循环生命符号监控
NCRunupTimeout：	E	S5time	建议：50 s	NC 启动监控
ListMDecGrp：	E	INT	0, 1, 2 ... 16	激活扩展 M 功能组解码 0：未生效 1...16：M 功能组数量
NCKomm：	E	BOOL	0 (FALSE), 1 (TRUE)	PLC-NC 通讯服务 FB2、3、4、5、7 1：生效
MMCToIF：	E	BOOL	0 (FALSE), 1 (TRUE)	HMI 信号向接口的传输（运行方式、程序控制等） 1：生效
HWheelMMC：	E	BOOL	0 (FALSE), 1 (TRUE)	0：通过 HMI 选择手轮 1：通过用户程序选择手轮

信号	方式	类型	取值范围	含义
ExtendAlMsg：	E	BOOL	0 (FALSE), 1 (TRUE)	激活 FC10 的故障消息和运行消息扩展（参见“PLC/HMI 接口 (页 969)”一章中的“通过 FC10 扩展 PLC 报警”部分）
MCP_IF_TCS	E	BOOL	0 (FALSE), 1 (TRUE)	FC19：检测按键 R11 0：不检测 1：按键 R11 作为“在刀具定向中手动运行”生效（参见“FC19:MCP_IFM - 将 MCP 信号传送到接口上 (页 1186)”一章中的“直角手动运行”部分）
ExtendChanAxMsg	E	BOOL	0 (FALSE), 1 (TRUE)	激活 FC10 的故障消息和运行消息的所有区域（参见章节“FC10：AL_MSG - 故障消息和运行消息 (页 1163)”）
MsgUser：	E	INT	0, 1, 2 ... 64	用户区域数量 (DB2) – 取决于参数 “ExtendAlMsg”
UserIR：	E	BOOL	0 (FALSE), 1 (TRUE)	需要 OB40 本地数据扩展，从而由用户处理信号
IRAuxfuT：	E	BOOL	0 (FALSE), 1 (TRUE)	检测 OB40 中 T 功能
IRAuxfuH：	E	BOOL	0 (FALSE), 1 (TRUE)	检测 OB40 中 H 功能
IRAuxfuE：	E	BOOL	0 (FALSE), 1 (TRUE)	检测 OB40 中 DL 功能
UserVersion：	E	POINTER	P#DBn.DBXx.0	指针，指向在操作界面的版本画面中显示的字符串变量 此字符串变量（最多 41 个字符）的书写方式如下： “**名称> <**版本 **xx.xx.xx> <**日期 **yy/mm/dd>**” 示例：“Test-Version 07.06.02 13/06/04”
OpKeyNum：	E	INT	0, 1, 2	生效的直接控制按键的数量 0：无以太网直接控制按键

信号	方式	类型	取值范围	含义
Op1KeyIn Op2KeyIn	E	POINTER	P#Ex.0 或 P#Mx.0 或 P#DBn.DBXx.0.	相关直接控制按键模块输入信号的起始地址
Op1KeyOut： Op2KeyOut：	E	POINTER	P#Ax.0 或 P#Mx.0 或 P#DBn.DBXx.0.	相关直接控制按键模块输出信号的起始地址
Op1KeyBusAdr： Op2KeyBusAdr：	E	INT	1, 2, 3 ... 191	直接控制按键通过以太网：TCU 索引
Op1KeyStop： Op2KeyStop：	E	BOOL	0 (FALSE), 1 (TRUE)	0：启动直接控制按键信号的传输 1：停止直接控制按键信号的传输
Op1KeyNotSend： Op2KeyNotSend：	E	BOOL	0 (FALSE), 1 (TRUE)	0：发送及接收运行激活 1：只接收直接控制按键信号
OpKeyBusType：	E	BYTE	b#16#55	以太网
IdentMcpBusAdr：	E	INT	1, 2, 3 ... 254	仅 IE 设备
IdentMcpProfilNo：	E	BYTE	0, 1	设备范畴 0：完整设备 1：仅直接控制按键
IdentMcpBusType：	E	BYTE	b#16#5	仅 IE 设备
IdentMcpStrobe：	E	BOOL	0 (FALSE), 1 (TRUE)	1：激活查询
MaxBAG：	A	INT	1, 2, 3 ... 10	运行方式组数量
MaxChan：	A	INT	1, 2, 3 ... 10	通道数量
MaxAxis：	A	INT	1, 2, 3 ... 31	轴数量
ActivChan：	A	ARRAY [1..10] OF BOOL	0 (FALSE), 1 (TRUE)	生效通道的位域
ActivAxis：	A	ARRAY [1..31] OF BOOL	0 (FALSE), 1 (TRUE)	生效轴的位域

信号	方式	类型	取值范围	含义
UDInt：	A	INT	0,	DB20 中 INTEGER 机床数据数量
UDHex：	A	INT	---	DB20 中十六进制机床数据数量
UDReal：	A	INT	---	DB20 中 REAL 机床数据数量
IdentMcpType：	A	BYTE	0, B#16#80, B#16#81, B#16#82 ... B#16#89	MCP 类型
IdentMcpLengthIn：	A	BYTE	---	MCP 输入数据的长度 (MCP → PLC)
IdentMcpLengthOut：	A	BYTE	---	MCP 输出数据的长度 (PLC → MCP)

如果在与机床控制面板 (MCP) 或手持操作设备 (HHU) 通讯时出错，HMI 上会显示以下报警且系统会将输入信号（MCP1In、MCP2In 或 BHGIn）设为值零：

故障情况：MCP / BHG

- 报警 400260：“OPI 1 故障”
- 报警 400261：“OPI 2 故障”
- 报警 400262：“HHU 故障”

如果 PLC 与 MCP/HHU 之间实现了重新同步，通讯会再次恢复，HMI 上的故障信息会通过基本程序删除，过程值被再次传送到输入信号（MCP1In、MCP2In 或 BHGIn）中。

示例：OB100 中的 FB1 调用

```
ORGANIZATION_BLOCK OB100
VAR_TEMP
    OB100_EV_CLASS :              BYTE;
    OB100_STRTUP :                BYTE;
    OB100_PRIORITY :              BYTE;
    OB100_OB_NUMBR :              BYTE;
    OB100_RESERVED_1 :            BYTE;
    OB100_RESERVED_2 :            BYTE;
    OB100_STOP :                  WORD;
    OB100_RESERVED_3 :            WORD;
    OB100_RESERVED_4 :            WORD;
    OB100_DATE_TIME :             DATE_AND_TIME;
END_VAR
```

```
BEGIN
    CALL FB1, DB7(                  // FB1 调用，实例 DB：DB7
          MCPNum :=                 1,
          MCP1In :=                 P#E0.0,
          MCP1Out :=                P#A0.0,
          MCP1StatSend :=           P#A8.0,
          MCP1StatRec :=            P#A12.0,
          MCP1BusAdr :=             6,
          MCP1Timeout :=            S5T#700MS,
          MCP1Cycl :=               S5T#200MS,
          NC-CyclTimeout :=         S5T#200MS,
          NC-RunupTimeout :=        S5T#50S);
//此处插入用户程序
END_ORGANIZATION_BLOCK
```

14.17.2 FB2：GET - 读取 NC 变量

功能

功能块 FB2 “GET” 用于读取 NC 区域中的变量。

为了建立对 NC 变量的参照，必须先通过“NC-VAR-Selector”工具选择这些变量，并将其作为 STL 源生成到一个数据块。之后必须在 S7 符号列表中为该数据块设定名称。这样一来，在调用 FB2 时便会以以下格式传输变量地址：参数 “Addr1” 至 “Addr8” = “<数据块名称>.<S7 名称>”。

请求读取 NC 变量

通过上升沿切换调用 FB2，参数 “Req” = 0 → 1

NC 变量的 S7 名称：参数 “Addr1” 至 “Addr8” = “NCVAR.<S7 名称>”

用于写入变量值的指针：参数 “RD1” 至 “RD8” = “P#<地址>”

完成读取请求

读取请求完成：参数 “Done” == 1。

读取请求出错：参数 “Error” == 1，故障原因在参数 “State” 中显示

前提条件

- 通过 OB100 使能 NC/PLC 通讯，FB1 参数 “NCKomm” = 1
- 必须通过 S7 项目的 S7 符号列表中的 S7 符号编辑器为数据块 DB120（数据接口）指定一个符号（标准：NCVAR）。通过该符号之后可在 FB2 参数 “Addr<x>” 中指定 NC 变量，例如：“ADDR1: = “NCVAR”.<NC 变量>”

边界条件

- FB2 支持多实例
- 每次调用 FB2 都必须指定一个独立的、用户区域中的实例 DB。
- 读取**通道专用**变量时，FB2 调用中通过 “Addr1” 至 “Addr8” 只能对**一个**通道的变量进行定址。
- 读取**驱动专用**变量时，FB2 调用中通过 “Addr1” 至 “Addr8” 只能对**一个**驱动对象 SERVO 的变量进行定址。驱动对象 SERVO 必须已指定给 NC 的一根机床轴。行索引对应逻辑驱动编号。
- 在此读取任务中只允许读取相同区域、通道或驱动对象的变量。

说明

故障

故障情形下，若从不同的通道或驱动对象读取变量，或同时从一个通道和一个驱动对象读取变量，则会产生以下故障信息：

- "Error" == TRUE
- "State" == W#16#02

变量定址

有一部分 NC 变量需要为其在 NC-VAR-Selector 中选择“区域编号”和/或“行”/“列”。此时可以选择一个基本类型，即将“区域编号”、“列”和“行”设为“0”。在 FB2 中会检查 NC-VAR-Selector 设定的“区域编号”、“列”和“行”是否为“0”。如果 NC-VAR-Selector 的值 == “0”，系统会接收相应的 FB2 参数的值。为此，在调用 FB2 前必须写入 FB2 参数 “Unit<x>”、“Column<x>” 和 “Line<x>”，其中 <x> = 1 - 8。

表格 14-1　对应参数

FB2 参数	NC-VAR-Selector
Unit	区域编号
Column	列
Line	行

在一个任务中可将同一组内的变量组合：

组别	区域				
1	C[1]	N	B	A	T
2	C[2]	N	B	A	T
3	V[.]	H[.]	---	---	---
示例中显示的组 1 和组 2 的规则同样适用于通道 3 到 10。					

说明

同时读取多个“String”类型的变量时，可使用变量的数量可能会小于 8。

功能说明

```
FUNCTION_BLOCK FB2
VAR_INPUT
```

```
    Req :                BOOL;
    NumVar :             INT;
    Addr1 :              ANY;
    Unit1 :              BYTE;
    Column1 :            WORD;
    Line1 :              WORD;
    Addr2 :              ANY;
    Unit2 :              BYTE;
    Column2 :            WORD;
    Line2 :              WORD;
    Addr3 :              ANY;
    Unit3 :              BYTE;
    Column3 :            WORD;
    Line3 :              WORD;
    Addr4 :              ANY;
    Unit4 :              BYTE;
    Column4 :            WORD;
    Line4 :              WORD;
    Addr5 :              ANY;
    Unit5 :              BYTE;
    Column5 :            WORD;
    Line5 :              WORD;
    Addr6 :              ANY;
    Unit6 :              BYTE;
    Column6 :            WORD;
    Line6 :              WORD;
    Addr7 :              ANY;
    Unit7 :              BYTE;
    Column7 :            WORD;
    Line7 :              WORD;
    Addr8 :              ANY;
    Unit8 :              BYTE;
    Column8 :            WORD;
    Line8 :              WORD;
END_VAR
VAR_OUTPUT
    Error :              BOOL;
    NDR :                BOOL;
    State :              WORD;
END_VAR
```

```
VAR_IN_OUT
    RD1 :                ANY;
    RD2 :                ANY;
    RD3 :                ANY;
    RD4 :                ANY;
    RD5 :                ANY;
    RD6 :                ANY;
    RD7 :                ANY;
    RD8 :                ANY;
END_VAR
```

对形式参数的说明

参数	方式	类型	取值范围	含义
Req：	E	BOOL	---	通过上升沿启动任务
NumVar：	E	INT	1 ... 8	待读取变量的数量："Addr1" - "Addr8"
Addr1 - Addr8：	E	ANY	“数据块名称”.<变量名称>	NC-VAR-Selector 中的变量名称
Unit1 - Unit8：	E	BYTE	---	区域地址，可选择用于变量定址
Column1 - Column8：	E	WORD	---	列地址，可选择用于变量定址
Line1 - Line8：	E	WORD	---	行地址，可选择用于变量定址
Error：	A	BOOL	0 (FALSE), 1 (TRUE)	1：任务受到消极应答或无法执行
NDR：	A	BOOL	0 (FALSE), 1 (TRUE)	1：任务已成功执行。数据可用
State：	A	WORD	---	参见“故障标识”
RD1 - RD8：	I/O	ANY	P#Mm.n BYTE x ... P#DBnr.dbxm.n BYTE x	读取数据的目标区域

故障标识

NC 变量

State		含义	提示
高位字节 1)	低位字节		
1 - 8	1	访问故障	---
0	2	任务中出错	任务中变量的组合错误
0	3	消极应答，任务无法执行	内部错误，解决办法： • 检查任务数据 • NC 复位
1 - 8	4	本地用户存储空间不足	读取的变量的数据类型比 “RD1” - “RD8” 中设定的要长；
0	5	格式转换错误	转换变量类型 double 时出错：变量不处于 S7-REAL 区域内
0	6	FIFO 已满	必须重复任务，因为等待序列已满
0	7	选件未设置	FB1 参数 “NCKomm” 未设置
1 - 8	8	目标区域错误（RD）	“RD1” - “RD8” 不允许为本地数据
0	9	传输被占用	必须重复任务
1 - 8	10	变量定址时出错	“Unit” 或 “Column” / “Line” 含 0 值
0	11	变量地址无效	检查“Addr”（或变量名称）、“Area”、“Unit”
0	12	NumVar == 0	检查参数 “NumVar”
1 - 8	13	ANY 数据的参照错误	所需的 “NcVar” 数据未设置
1) 高位字节 > 0 ⇒ 出现故障的变量的编号			

驱动专用变量

如果在读取/写入驱动专用变量时出错 (DB1200.DBX3000.1 == 1)，访问结果中会显示一个故障编号（该编号基于在 PROFIdrive 协议中定义的故障编号）。

State	含义
x	<PROFIdrive 协议的故障编号> + 20_H 或 36_D

确定访问结果的含义：

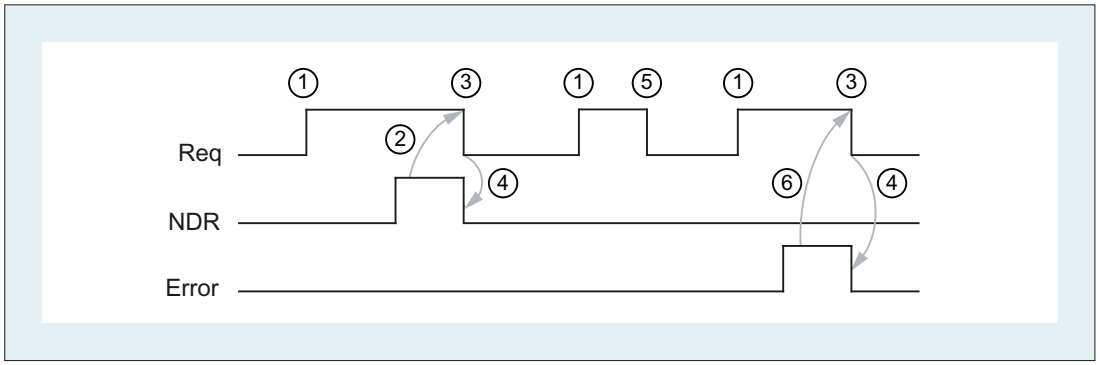
1. 计算 PROFIdrive 协议的故障编号
   <PROFIdrive 协议的故障编号**> +** 访问结果 - 20_H 或 36_D。
2. 确定 PROFIdrive 协议的故障编号的含义
   有关 PROFIdrive 协议的故障编号的说明请见：
   文档
   功能手册之驱动功能 SINAMICS S120；章节“通讯” **>** “PROFIdrive 通讯” **>** “非循环通讯” **>** “任务和应答结构” **>** “参数应答中的故障值"

配置步骤

读取 NC 变量需执行下列配置步骤：

- 通过 NC-VAR-Selector 选择变量
- 将所选变量保存为 *.VAR 文件
- 生成一个 STEP 7 源文件 *.AWL
- 通过相应的地址设定生成一个 DB
- 将生成的 DB 的符号输入符号表，从而可在用户程序中以符号访问地址参数
- FB2 的参数设置

脉冲示意图



① 用户：置位请求，Req = 0 → 1

② FB2 成功执行，NDR = 1
用户：复位请求，如果 NDR == 1 则 Req = 0

③ 用户：如果 NDR == 1 则复位请求：1 → 0

④ FB2 复位任务确认，NDR = 0

⑤ 用户：如果 NDR == 0 且 Error == 0 则**不允许**复位请求 Req = 1 → 0

⑥ FB2 执行时出错，Error = 1
用户：复位请求，如果 NDR == 1 或 Error == 1 则 Req = 0，可能需要执行故障处理

调用示例

从通道 1 读取三个通道专用机床数据，其地址设定保存在 DB120 中。

指定数据

数据通过 NC-VAR-Selector 选择并保存在文件 DB120.**VAR** 中。之后会生成文件 DB120.**AWL**。

选择 S7-(ALIAS-) 名称，

为了将通道名称接收到变量名称中并删除 STEP 7 符号中不允许使用的 "[" und "]" 字符，系统会选择新的 S7 名称：

区域	模块	名称	类型	编号	字节	S7 名称
C[1]	M	MD20070 $MC_AXCONF_MACHAX_USED[1]	CHAR	20070	1	C1AxConfMachAxUsed1
C[1]	M	MD20070 $MC_AXCONF_MACHAX_USED[2]	CHAR	20070	1	C1AxConfMachAxUsed2
C[1]	M	MD20090 $MC_SPIND_DEF_MASTER_SPIND	INT	20090	1	C1SpindDefMasterSpind

S7 符号列表

作为数据块 DB120 的符号名称，“NCVAR” 会被写入 S7 符号列表中。

符号	操作数	数据类型
NCVAR	DB120	DB120

文件 DB120.AWL 必须进行编译并传输至 PLC。

带有实例 DB110 的 FB2 的参数设置：

```
DATA_BLOCK DB110          //自由用户 DB，作为 FB2 的实例
FB2
BEGIN
END_DATA_BLOCK
功能 FC "VariablenCall" :VOID
```

```
U      E 7.7;                        //机床控制面板自由按键
S      M 100.0;                      //激活 Req
U      M 100.1;                      //NDR 结束消息
R      M 100.0;                      //结束任务
U      E 7.6;                        //手动应答故障
U      M 102.0;                      //故障存在
R      M 100.0;                      //结束任务
CALL FB2, DB110(
   Req :            M 100.0,
   NumVar :         3,              //读取 3 个变量
   Addr1:=          "NCVAR".C1AxConfMachAxUsed1,
   Addr2:=          "NCVAR".C1AxConfMachAxUsed2,
   Addr3 :=         "NCVAR".C1SpindDefMasterSpind,
   Error :=         M102.0,
   NDR :=           M100.1,
   State :=         MW104,
   RD1 :=           P#DB99.DBX0.0 BYTE 1,
   RD2 :=           P#DB99.DBX1.0 BYTE 1,
   RD3 :=           P#M110.0 INT 1);
```

示例：变量定址

从通道 1 读取两个 R 参数，其地址设定作为基本类型保存在 DB120 中。R 参数编号通过参数 “Line<x>” 设置。

```
DATA_BLOCK DB120
VERSION: 0.0
STRUCT
   C1_RP_rpa0_0 :
   STRUCT
   SYNTAX_ID :                BYTE := B#16#82;
   区域和单位:                BYTE := B#16#41;
   列 :                       WORD := W#16#1;
   行 :                       WORD := W#16#0;
   模块类型 :                 BYTE := B#16#15;
   行数 :                     BYTE := B#16#1;
   类型 :                     BYTE := B#16#F;
   长度:                      BYTE := B#16#8;
   END_STRUCT;
END_STRUCT;
BEGIN
END_DATA_BLOCK
```

```
CALL FB2, DB110(
   Req :                                  M 0.0,
   NumVar :                               2,
   Addr1:=                                "NCVAR".C1_RP_rpa0_0,
   Line1 :=                               W#16#1,
   Addr2:=                                "NCVAR".C1_RP_rpa0_0,
   Line2 :=                               W#16#2,
   Error :=                               M 1.0,
   NDR :=                                 M 1.1,
   State :=                               MW 2,
   RD1 :=                                 P#M 4.0 REAL 1,
   RD2 :=                                 P#M 24.0 REAL 1);
```

数据类型对应关系

表格 14-2　数据类型对应关系

NC 数据类型	S7 数据类型
DOUBLE	REAL
DOUBLE	REAL2
FLOAT	REAL
LONG	DINT
INTEGER	DINT
UINT_32	DWORD
INT_16	INT
UINT_16	WORD
UNSIGNED	WORD
CHAR	CHAR 或 BYTE
STRING	STRING
BOOL	BOOL
DATETIME	DATE_AND_TIME

示例

为了在不调整格式的前提下从 NC 读取 “DOUBLE” 类型的变量，必须在目标区域 “RDx” 中设定一个指向 REAL 2 类型的 ANY 指针。（例如：P#M100.0 REAL2）。读取 “DOUBLE” 类型的变量时，若 PLC 基本程序将其作为目标类型 REAL2 识别，那么该数据将作为 64 位浮点值接收至 PLC 数据区域。

14.17.3 FB3：PUT - 写入 NC 变量

功能

功能块 FB3 “PUT” 用于写入 NC 区域中的变量。

为了建立对 NC 变量的参照，必须先通过 “NC-VAR-Selector” 工具选择这些变量，并将其作为 STL 源生成到一个数据块中。之后必须在 S7 符号列表中为该数据块设定名称。这样一来，在调用 FB3 时便会以以下格式传输变量地址：参数 “Addr1” 至 “Addr8” = “<数据块名称>”.“<S7 名称>”

请求写入 NC 变量

通过上升沿切换调用 FB3，参数 “Req” = 0 → 1

NC 变量的 S7 名称：参数 “Addr1” 至 “Addr8” = “NCVAR.<S7 名称>”

用于写入变量值的指针：参数 “RD1” 至 “RD8” = “P#<地址>”

完成写入请求

写入请求完成：参数 “Done” == 1。

写入请求出错：参数 “Error” == 1，故障原因在参数 “State” 中显示

前提条件

- 通过 OB100 使能 NC/PLC 通讯，FB1 参数 “NCKomm” = 1
- 必须通过 S7 项目的 S7 符号列表中的 S7 符号编辑器为数据块 DB120（数据接口）指定一个符号（标准：NCVAR）。通过该符号之后可在 FB3 参数 “Addr<x>” 中指定 NC 变量，例如：“ADDR1:= “NCVAR”.<NC 变量>”

边界条件

- FB3 支持多实例
- 每次调用 FB3 都必须指定一个独立的、用户区域中的实例 DB。
- 为了能在不输入密码的情形下写入机床数据或 GUD，必须将相应数据的保护级改为最低。

  文档：
  - 调试手册；章节：“保护等级方案”
  - 编程手册之工作准备分册；章节：“定义用户数据保护等级”
- 写入**通道专用**变量时，FB2 调用中通过 “Addr1” 至 “Addr8” 只能对**一个**通道的变量进行定址。

- 读取**驱动专用**变量时，FB2 调用中通过 “Addr1” 至 “Addr8” 只能对**一个**驱动对象 SERVO 的变量进行定址。驱动对象 SERVO 必须已指定给 NC 的一根机床轴。行索引对应逻辑驱动编号。
- 在此写入任务中只允许写入相同区域、通道或驱动对象的变量。

说明

故障

故障情形下，若从不同的通道或驱动对象写入变量，或同时从一个通道和一个驱动对象写入变量，则会产生以下故障信息：

- "Error" == TRUE
- "State" == W#16#02

变量定址

有一部分 NC 变量需要为其在 NC-VAR-Selector 中选择“区域编号”和/或“行”/“列”。此时可以选择一个基本类型，即将“区域编号”、“列”和“行”设为“0”。在 FB2 中会检查 NC-VAR-Selector 设定的“区域编号”、“列”和“行”是否为“0”。如果 NC-VAR-Selector 的值 == “0”，系统会接收相应的 FB3 参数的值。为此，在调用 FB3 前必须写入 FB3 参数 “Unit<x>”、“Column<x>” 和 “Line<x>”，其中 <x> = 1 - 8。

FB3 参数	NC-VAR-Selector
Unit<x>	区域编号
Column<x>	列
Line<x>	行

在一个任务中可将同一组内的 NC 变量组合：

组别	区域				
1	C[1]	N	B	A	T
2	C[2]	N	B	A	T
3	V[.]	H[.]	---	---	---
示例中显示的组 1 和组 2 的规则同样适用于通道 3 到 10。					

说明

同时写入多个“String”类型的变量时，可使用变量的数量可能会小于 8。

功能说明

```
FUNCTION_BLOCK FB3
VAR_INPUT
    Req :               BOOL;
    NumVar :            INT;
    Addr1 :             ANY;
    Unit1 :             BYTE;
    Column1 :           WORD;
    Line1 :             WORD;
    Addr2 :             ANY;
    Unit2 :             BYTE;
    Column2 :           WORD;
    Line2 :             WORD;
    Addr3 :             ANY;
    Unit3 :             BYTE;
    Column3 :           WORD;
    Line3 :             WORD;
    Addr4 :             ANY;
    Unit4 :             BYTE;
    Column4 :           WORD;
    Line4 :             WORD;
    Addr5 :             ANY;
    Unit5 :             BYTE;
    Column5 :           WORD;
    Line5 :             WORD;
    Addr6 :             ANY;
    Unit6 :             BYTE;
    Column6 :           WORD;
    Line6 :             WORD;
    Addr7 :             ANY;
    Unit7 :             BYTE;
    Column7 :           WORD;
    Line7 :             WORD;
    Addr8 :             ANY;
    Unit8 :             BYTE;
    Column8 :           WORD;
    Line8 :             WORD;
END_VAR
VAR_OUTPUT
```

```
    Error :             BOOL;
    Done :              BOOL;
    State :             WORD;
END_VAR
VAR_IN_OUT
    SD1 :               ANY;
    SD2 :               ANY;
    SD3 :               ANY;
    SD4 :               ANY;
    SD5 :               ANY;
    SD6 :               ANY;
    SD7 :               ANY;
    SD8 :               ANY;
END_VAR
```

对形式参数的说明

信号	方式	类型	取值范围	含义
Req：	E	BOOL	-	通过上升沿启动任务
NumVar：	E	INT	1 ... 8	待写入变量的数量："Addr1" - "Addr8"
Addr1 - Addr8：	E	ANY	“数据块名称”.<变量名称>	NC-VAR-Selector 中的变量名称
Unit1 - Unit8：	E	BYTE	-	区域地址，可选择用于变量定址
Column1 - Column8：	E	WORD	-	列地址，可选择用于变量定址
Line1 - Line8：	E	WORD	-	行地址，可选择用于变量定址
Error：	A	BOOL	0 (FALSE), 1 (TRUE)	任务受到消极应答或无法执行
Done：	A	BOOL	0 (FALSE), 1 (TRUE)	任务已成功执行。
State：	A	WORD	-	参见“故障标识”
SD1 - SD8：	I/O	ANY	P#Mm.n BYTE x... P#DBnr.dbxm.n BYTE x	待写入数据

故障标识

NC 变量

State		含义	提示
高位字节 1)	低位字节		
1 - 8	1	访问故障	---
0	2	任务中出错	任务中变量的组合错误
0	3	消极应答，任务无法执行	内部故障，可能的解决办法： • 检查任务 • NC 复位
1 - 8	4	数据区域或数据类型不一致，或字符串为空	检查 “SD1” - “SD8” 中的数据
0	6	FIFO 已满	必须重复任务，因为等待序列已满
0	7	选件未设置	FB1 参数 “NCKomm” 未设置
1 - 8	8	目标区域错误（SD）	“SD1” - “SD8” 不允许为本地数据
0	9	传输被占用	必须重复任务
1 - 8	10	变量定址时出错	“Unit” 或 “Column” / “Line” 含 0 值
0	11	变量地址无效，或变量为只读	检查“Addr”（或变量名称）、“Area”、“Unit”
0	12	NumVar == 0	检查参数 “NumVar”
1 - 8	13	ANY 数据的参照错误	所需的 “NcVar” 数据未设置
1 - 8	15	有效数据过长	解决方法：减少每个任务写入的变量数量，或使用较短的字符串变量
1) 高位字节 > 0 ⇒ 出现故障的变量的编号			

驱动专用变量

如果在读取/写入驱动专用变量时出错 (DB1200.DBX3000.1 == 1)，访问结果中会显示一个故障编号（该编号基于在 PROFIdrive 协议中定义的故障编号）。

State	含义
x	<PROFIdrive 协议的故障编号> + 20_H 或 36_D

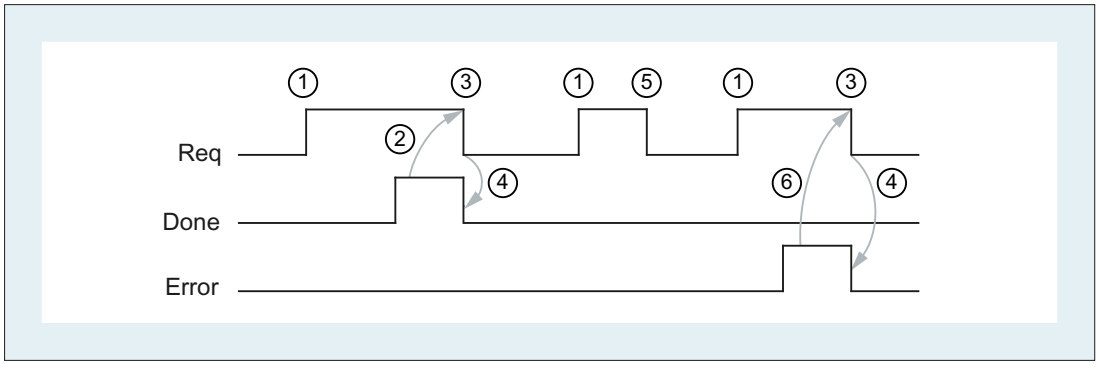
确定访问结果的含义：

1. 计算 PROFIdrive 协议的故障编号
   <PROFIdrive 协议的故障编号**> +** 访问结果 - 20_H 或 36_D。
2. 确定 PROFIdrive 协议的故障编号的含义
   有关 PROFIdrive 协议的故障编号的说明请见：
   文档
   功能手册之驱动功能 SINAMICS S120；章节“通讯” **>** “PROFIdrive 通讯” **>** “非循环通讯” **>** “任务和应答结构” **>** “参数应答中的故障值"

配置步骤

写入 NC 变量所需的配置步骤与读取 NC 变量时相同。建议将需要读取或写入的 NC 变量的地址设定保存在一个 DB 中。

脉冲示意图



① 用户：置位请求，Req = 0 → 1

② FB3 成功执行，Done = 1
用户：复位请求，如果 Done == 1 则 Req = 0

③ 用户：如果 Done == 1 则复位请求：1 → 0

④ FB3 复位任务确认，Done = 0

⑤ 用户：如果 Done == 0 且 Error == 0 则**不允许**复位请求 Req = 1 → 0

⑥ FB3 执行时出错，Error = 1
用户：复位请求，如果 Done == 1 或 Error == 1 则 Req = 0，可能需要执行故障处理

调用示例

从通道 1 写入三个通道专用机床数据：

通过 NC-VAR-Selector 选择这三个数据并保存至文件 DB120.VAR：

选择 S7-(ALIAS-) 名称，从而将数据块名称接收到变量名称中并删除 STEP 7 符号中不允许使用的 “[ ]” 字符

区域	模块	名称	类型	字节	S7 名称
C[1]	RP	rpa[5]	DOUBLE	4	rpa_5C1RP
C[1]	RP	rpa[11]	DOUBLE	4	rpa_11C1RP
C[1]	RP	rpa[14]	DOUBLE	4	rpa_14C1RP

通过 S7 符号编辑器为 DB120 输入 NCVAR

符号	操作数	数据类型
NCVAR	DB120	DB120

文件 DB120.AWL 必须进行编译并传输至 PLC。

通过实例 DB111 调用和参数设置 FB3

```
DATA_BLOCK DB111        //自由用户 DB，作为 FB3 的实例
FB3
BEGIN
功能 FC "VariablenCall" :VOID
END_DATA_BLOCK
```

```
U      E 7.7;          //机床控制面板自由按键
S      M 100.0;        //激活 Req
U      M 100.1;        //Done 结束消息
R      M 100.0;        //结束任务
U      E 7.6;          //手动应答故障
U      M 102.0;        //故障存在
R      M 100.0;        //结束任务
CALL FB3, DB111(
   Req :=           M 100.0,
   NumVar :=        3,                      //写入 3 个变量
   Addr1:=          "NCVAR".rpa_5C1RP,
   Addr2:=          "NCVAR".rpa_11C1RP,
   Addr3 :=         "NCVAR".rpa_14C1RP,
   Error :=         M102.0,
   Done :=          M100.1,
   State :=         MW104,
   SD1 :=           P#DB99.DBX0.0 REAL 1,
   SD2 :=           P#DB99.DBX4.0 REAL 1,
   SD3 :=           P#M110.0 REAL 1);
```

示例：变量定址

写入两个通道 1 的 R 参数，其地址设定作为基本类型保存在 DB120 中。R 参数编号通过参数 LineX 设置。

```
DATA_BLOCK DB120
VERSION: 0.0
STRUCT
   C1_RP_rpa0_0 :
   STRUCT
   SYNTAX_ID :              BYTE := B#16#82;
   区域和单位:              BYTE := B#16#41;
   列 :                     WORD := W#16#1;
   行 :                     WORD := W#16#0;
   模块类型 :               BYTE := B#16#15;
   行数 :                   BYTE := B#16#1;
   类型 :                   BYTE := B#16#F;
   长度:                    BYTE := B#16#8;
   END_STRUCT;
END_STRUCT;
BEGIN
END_DATA_BLOCK
```

```
CALL FB3, DB122(
   Req :=                  M 10.0,
   NumVar :=               2,
   Addr1:=                 "NCVAR".C1_RP_rpa0_0,
   Line1 :=                W#16#1,
   Addr2:=                 "NCVAR".C1_RP_rpa0_0,
   Line3 :=                W#16#2
   Error :=                M 11.0,
   Done :=                 M 11.1,
   State :=                MW 12,
   SD1 :=                  P#M 4.0 REAL 1,
   SD2 :=                  P#M 24.0 REAL 1);
```

数据类型对应关系

参见“FB2：GET - 读取 NC 变量 (页 1061)”一章中的表格“数据类型对应关系”。

14.17.4 FB4:PI_SERV - 请求 PI 服务

功能

功能块 FB4 “PI_SERV”用于启动 PI 服务。

可用的 PI 服务连同其专用参数在下一章节中作详细说明。现有 PI 服务一览请见：可供使用的 PI 服务列表 (页 1085)

说明

由于参数 “WVar” 上的数量较大，建议使用扩展功能块 FB7 来替代 FB4。参见“FB7：PI_SERV2（请求 PI 服务） (页 1122)”章节。

启动 PI 服务

请求启动 PI 服务：通过上升沿切换调用 FB4，参数 “Req” = 0 → 1

完成 PI 服务

任务或 PI 服务完成：参数 “Done” == 1。

任务或 PI 服务出错：参数 “Error” == 1，故障原因在参数 “State” 中显示

前提条件

- 通过 OB100 使能 NC/PLC 通讯，FB1 参数 “NCKomm” = 1
- Für den Datenbaustein DB16 (Datenschnittstelle der PI-Dienste) muss mit dem S7-Symbol Editor in der S7-Symbolliste des S7-Projekts ein Symbol (Standard:PI) vergeben werden.通过该符号之后可在 FB4 参数 “PIService” 中指定请求的 PI 服务，例如：“PIService:= “PI”.<PI 服务>”

前提条件

- 每次调用 FB4 都必须指定一个独立的、用户区域中的实例 DB。
- 只允许在 PLC 基本程序 (OB1) 的循环分量中启动 PI 服务（通过 Req = 1 调用 FB4）。
  如果未启动 PI 服务（通过 Req = 0 调用 FB4），则也可在 PLC 基本程序 (OB100) 的启动分量中写入参数。之后可通过已写入的参数在 PLC 基本程序 (OB1) 的循环分量中启动 PI 服务（通过 Req = 1 调用 FB4）。
- PI 服务的执行通常会持续多个 PLC 循环。

功能说明

```
FUNCTION_BLOCK FB4
VAR_INPUT
w   Req :               BOOL;
    PIService :         ANY;
    Unit :              INT;
    Addr1 :             ANY;
    Addr2 :             ANY;
    Addr3 :             ANY;
    Addr4 :             ANY;
    WVar1 :             WORD;
    WVar2 :             WORD;
    WVar3 :             WORD;
    WVar4 :             WORD;
    WVar5 :             WORD;
    WVar6 :             WORD;
    WVar7 :             WORD;
    WVar8 :             WORD;
    WVar9 :             WORD;
    WVar10 :            WORD;
END_VAR
VAR_OUTPUT
```

```
    Error :             BOOL;
    Done :              BOOL;
    State :             WORD;
END_VAR
```

对形式参数的说明

信号	方式	类型	取值范围	含义
Req：	E	BOOL	0 (FALSE), 1 (TRUE)	任务请求
PIService：	E	ANY	“<数据块名称>”.<PI 服务>	请求的 PI 服务 • <数据块名称>：Symbolname für DB16, Standard:"PI" • <PI 服务>：可供使用的 PI 服务列表 (页 1085)
Unit：	E	INT	1, 2, 3 ...	区域编号
Addr1 至 Addr4：	E	ANY	“<数据块名称>”.<变量名称>	参照字符串 含义取决于所选 PI 服务
WVar1 至 WVar10：	E	WORD	-32768 ... 32767 8000_H - $7FFF_H$	INTEGER 或 WORD 变量 含义取决于所选 PI 服务
Error：	A	BOOL	0 (FALSE), 1 (TRUE)	故障状态 1：任务受到消极应答或无法执行
Done：	A	BOOL	0 (FALSE), 1 (TRUE)	任务状态 1：任务已成功执行
State：	A	WORD	[1)]	故障标识 只在 “Error” == 1 时相关：

信号		方式	类型	取值范围	含义
1)	State	含义			提示
	3	消极应答，任务无法执行			内部故障，可尝试通过 NC 复位解决
	6	FIFO 已满			重复任务
	7	选件未设置			FB1 参数 “NCKomm” 未设置
	9	传输被占用			重复任务
	13	Addr1..Adddr4:Referenz ungültig			字符串数据未设定
	14	PI 服务未知			参数 “PIService” 中给定的 PI 服务未知。→ 进行检查。
	15	Addr1..Adddr4:String zu lang			检查字符串长度

调用示例

功能：通道 1 中的程序选择（主程序和工件程序）

通过 S7-SYMBOL-Editor 为 DB16 输入 PI，为 DB124 输入 STR：

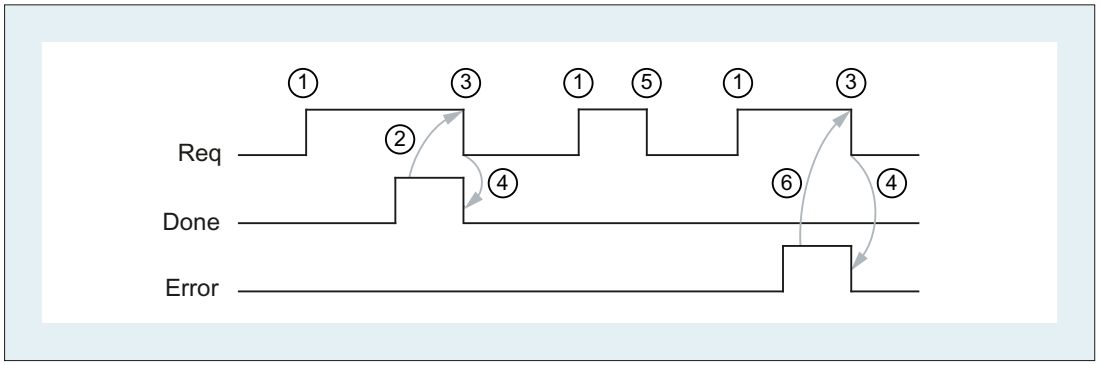
参数设置		
符号	操作数	数据类型
PI	DB16	DB16
STR	DB124	DB124

```
DATA_BLOCK DB126          //FB4 的实例，自由用户 DB
FB4
BEGIN
END_DATA_BLOCK
// ---------------------------------------------------------------------------
DATA_BLOCK DB124
   STRUCT
      PName:        string[32]:= '_N_TEST_MPF'; //主程序
      Path:         string[32]:= '/             //主程序路径
                    _N_MPF_DIR/';
      PName_WST:    string[32]:= '_N_ABC_MPF';  //工件程序
      Path_WST:     string[32]:= '/_N_WKS_DIR/  //工件程序路径
                    _N_ZYL_WPD';
   END_STRUCT
```

```
BEGIN
END_DATA_BLOCK
// ------------------------------------------------------------------------------
Function FC "PICall" : VOID
   U   E 7.7;          //机床控制面板自由按键
   S   M 0.0;          //激活 Req
   U   M 1.1;          //Done 结束消息
   R   M 0.0;          //结束任务
   U   E 7.6;          //手动应答故障
   U   M 1.0;          //故障存在
   R   M 0.0;          //结束任务

   CALL FB4, DB126 (
      Req:=          M0.0,
      PIService:=    "PI".SELECT,                 // PI 服务：SELECT
      Unit:=         1,                           //通道 1
      Addr1:=        "STR".Path,                  //主程序：路径
      Addr2:=        "STR".PName,                 //主程序：程序
   //  Addr1:=        "STR".Path_WST,              //工件：路径
   //  Addr2:=        "STR".PName_WST,             //工件：程序
      Error:=        M1.0,
      Done:=         M1.1,
      State:=        MW2
   );
```

流程图



① 用户：置位请求，Req = 0 → 1

② FB4：PI 服务成功执行，Done = 1
用户：复位请求，如果 Done == 1 则 Req = 0

③ 用户：如果 Done == 1 则复位请求：1 → 0

④ FB4：复位任务确认，Done = 0

⑤ 用户：如果 Done == 0 且 Error == 0 则**不允许**复位请求 Req = 1 → 0

⑥ FB4：PI 服务执行时出错，Error = 1
用户：复位请求，如果 Done == 1 或 Error == 1 则 Req = 0，可能需要执行故障处理

14.17.4.1 可供使用的 PI 服务列表

通用 PI 服务

PI 服务	功能
ASUB (页 1086)	指定中断
CANCEL (页 1087)	取消执行
CONFIG (页 1088)	对标记的机床数据的重新配置
DIGION (页 1088)	激活数字化
DIGIOF (页 1088)	取消数字化
FINDBL (页 1089)	激活程序段搜索
LOGIN (页 1089)	激活口令
LOGOUT (页 1090)	复位口令
NCRES (页 1090)	触发 NC 复位
SELECT (页 1090)	为一个通道选择程序用于执行

PI 服务	功能
SETUDT (页 1091)	将当前的用户数据激活
SETUFR (页 1092)	激活用户框架
RETRAC (页 1092)	沿刀具方向退刀

刀具管理 PI 服务

PI 服务	功能
CRCEDN (页 1093)	在输入的 T 号下创建刀沿
CREACE (页 1094)	用下一个未占用的 D 号创建刀沿
CREATO (页 1094)	通过设定 T 号创建刀具
DELECE (页 1094)	删除一个刀沿
DELETO (页 1095)	删除刀具
MMCSEM (页 1095)	各种 PI 服务的信号量
TMCRTO (页 1097)	通过给定一个名称，一个副编号来创建刀具
TMFDPL (页 1098)	搜索空刀位用于装刀
TMFPBP (页 1099)	搜索空刀位
TMGETT (页 1100)	设定的含副编号的刀具名称的 T 号
TMMVTL (页 1101)	准备刀库用于装刀，卸载刀具
TMPOSM (页 1103)	定位刀库或刀具
TMPCIT (页 1104)	设置工件计数器的增量值
TMRASS (页 1104)	复位生效状态
TRESMO (页 1105)	复位监控值
TSEARC (页 1105)	通过搜索掩码进行复杂搜索
TMCRMT (页 1109)	创建多刀
TMDLMT (页 1110)	删除多刀
POSMT (页 1110)	定位多刀
FDPLMT (页 1111)	搜索/检查多刀内的空刀位

14.17.4.2 PI 服务：ASUB

功能：指定中断

将一个保存在 NC 上的程序指定给通道的一个中断信号。程序必须可执行。路径名称和程序名称必须完全指定。

程序管理、路径和文件名称的详细信息参见：

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”

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	"PI".ASUP	指定中断 1)
Unit：	INT	1, 2, 3 ... 10	通道
Addr1：	STRING		路径名
Addr2	STRING		程序名
WVar1：	WORD	1, 2, 3 ... 8	中断编号
WVar2：	WORD	1, 2, 3 ... 8	优先级
WVar3：	WORD	0, 1	LIFTFAST 2)
WVar4：	WORD	0, 1	BLSYNC 3)

1) 也可使用 SETINT 指令指定中断。见 2)

2)**文档：**
编程手册之工作准备分册；章节：“灵活 NC 编程” > “中断程序（ASUB）” > “从轮廓快速退刀（SETINT、**LIFTFAST**、ALF）”

3)**文档：**
编程手册之工作准备分册；章节：“灵活 NC 编程” > “中断程序（ASUB）” > “指定和启动中断程序（SETINT、PRIO、**BLSYNC**）”

说明

PI 服务 ASUB 只能在指定通道的复位状态下执行。

文档:
编程手册之工作准备分册，章节：“灵活 NC 编程” > “中断程序（ASUB）”

14.17.4.3 PI 服务：CANCEL

功能：取消执行

触发“Cancel”（取消）功能，与操作界面（操作面板前端）的“Alarm Cancel”（报警取消）键具有同等效果。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”CANCEL	Cancel

14.17.4.4 PI 服务：CONFIG

功能：重新配置

此 PI 服务能够近乎同步地激活由操作人员或 PLC 按顺序输入的机床数据。

此命令只能在控制系统复位或程序中断（程序段交界处 NC 停止）状态下激活。若不遵循此要求，则会引起 FB4 的故障反馈（State = 3）。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	"PI".CONFIG	重新配置
Unit：	INT	1	
WVar1：	INT	1	分类

14.17.4.5 PI 服务：DIGION

功能：激活数字化

在设置的通道中选择数字化。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”DIGION	激活数字化
Unit：	INT	1, 2, 3 ... 10	通道

14.17.4.6 PI 服务：DIGIOF

功能：取消数字化

在设置的通道中取消数字化。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”DIGIOF	取消数字化
Unit：	INT	1, 2, 3 ... 10	通道

14.17.4.7 PI 服务：FINDBL

功能：激活程序段搜索

将通道切换至程序段搜索模式并向其发送应答。之后 NC 会立即执行程序段搜索。此时程序段搜索指针必须已位于 NC 中。可随时通过 NC 复位终止程序段搜索。程序段搜索完成后将自动重新激活正常的执行模式。以找到的搜索目标为起点进行 NC 启动。

操作人员须自行确保逼近过程中不会发生碰撞。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	"PI".FINDBL	程序段搜索
Unit：	INT	1 至 10	通道
WVar1：	WORD	1, 2, 3	预处理模式 1：不进行计算 2：进行计算 3：考虑主程序段

14.17.4.8 PI 服务：LOGIN

功能：创建口令

将参数设置的口令传输至 NC。口令一般由 8 个字符组成。必要时须在口令的字符串中添加空字符。

示例

口令：STRING[8] := 'SUNRISE';

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”LOGIN	创建口令
Unit：	INT	1	NC
Addr1：	STRING	8 个字符	口令

14.17.4.9 PI 服务：LOGOUT

功能：复位口令

将最近一次传输至 NC 的口令复位。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”LOGOUT	复位口令
Unit：	INT	1	NC

14.17.4.10 PI 服务：NCRES

功能：触发 NC 复位

触发 NC 复位。参数“Unit”和“WVar1”始终须设为 0。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”NCRES	触发 NC 复位
Unit：	INT	0	-
WVar1：	WORD	0	-

14.17.4.11 PI 服务：SELECT

功能：为通道选择用于执行的程序

为一个通道选择一个保存在 NC 上的程序用于执行。前提条件是该文件可执行。必须以正确的格式输入路径名和程序名。详细信息请参见：

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”

可使用的模块类型

模块类型	
工件目录	WPD
主程序	MPF
子程序	SPF
循环	CYC
异步子程序	ASP
二进制文件	BIN

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”SELECT	程序选择
Unit：	INT	1, 2, 3 ... 10	通道
Addr1：	STRING	-	路径名
Addr2：	STRING	-	程序名称

14.17.4.12 PI 服务：SETUDT

功能：激活当前的用户数据

将当前用户数据（例如刀具补偿、基本框架和可设定框架）设置为在下一个 NC 程序段中生效（设置只在停止状态下进行）。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”SETUDT	激活用户数据
Unit：	INT	1, 2, 3 ... 10	通道
WVar1：	WORD	1, 2, 3, 4, 5	用户数据类型 1：生效的刀具补偿 2：生效的基本框架 3：生效的可设定框架 4：生效的全局基本框架 5：生效的全局可设定框架

信号	类型	取值范围	含义
WVar2：	WORD	0	预留
Wvar3：	WORD	0	预留

14.17.4.13 PI 服务：SETUFR

功能：激活用户框架

将用户框架载入 NC。所有必需的框架值必须已事先通过 FB3“写入变量”传输至 NC。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”SETUFR	激活用户框架
Unit：	INT	1, 2, 3 ... 10	通道

14.17.4.14 PI 服务：RETRAC

功能：选择 JOG-Retract

选择 JOG-Retract 运行方式。退刀轴（即用于执行回退的几何轴）可由 NC 自动选定或显性设定。

运行方式直到以 RESET 结束时保持生效。

说明

PI 服务 RETRAC 只能在 JOG 运行方式中“复位”状态下激活。

自动确定

采用自动选定时，系统会将垂直于当前选择的工作平面的几何轴选作退刀轴。

- G17：退刀轴 ⇒ 第 3 几何轴 (Z)
- G18：退刀轴 ⇒ 第 2 几何轴 (Y)
- G19：退刀轴 ⇒ 第 1 几何轴 (X)

说明

OPI 变量 retractState

可通过 OPI 变量 retractState 位 2/3 读取生效的退刀轴。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”RETRAC	选择 JOG-Retract 运行方式
Unit：	INT	1, 2, 3 ... 10	通道
WVar1：	WORD	0, 1, 2, 3	退刀轴 0：由 NC 自动选定退刀轴。 1：退刀轴是 WCS 的第 1 几何轴 2：退刀轴是 WCS 的第 2 几何轴 3：退刀轴是 WCS 的第 3 几何轴
WVar2：	WORD	0	预留。该值必须预设为 0。

14.17.4.15 PI 服务：CRCEDN

功能：创建新刀沿

在此 PI 服务中通过“T 编号”参数设定现有刀具的 T 编号时，将会创建该刀具的刀沿。在此情形下，参数“D 编号”（待创建刀沿的编号）的取值范围为 1, 2, 3 ... 9。

若设定了一个正的 T 编号，但该 T 编号对应的刀具不存在，那么 PI 服务失败。

若将 T 编号设定为 0（绝对 D 编号模型），那么 D 编号的取值范围将扩展至 1, 2, 3 ... 31999。此时新的刀沿通过设定的 D 编号生成。

若存在已设定的刀沿，那么上述两种情形下 PI 服务均会失败。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”CRCEDN	创建新刀沿
Unit：	INT	1, 2, 3 ... 10	TOA
WVar1：	INT		需要创建刀沿的刀具的 T 编号。赋值为 00000 时，表示不启用对刀具的参考（绝对 D 编号）
WVar2：	INT	1 ... 9	刀沿编号
		1 - 31999	

14.17.4.16 PI 服务：CREACE

功能：创建刀沿

针对指定的 T 编号所对应的刀具，在 TO、TS 中通过下一个/自由 D 编号创建刀沿，并在 TUE（若存在）模块中为 OEM 刀沿数据创建刀沿。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”CREACE	创建刀沿
Unit：	INT	1, 2, 3 ... 10	TOA
WVar1：	INT		T 编号

14.17.4.17 PI 服务：CREATO

功能：创建刀具

通过设定 T 编号创建刀具。该刀具会在 TV 区域（刀具目录）中记录为存在，同时会在 TO 模块中为补偿创建第一个刀沿“D1”（内容为零），以及在 TUE 模块（若存在）中为 OEM 刀沿数据创建第一个刀沿“D1”（内容为零）。若存在 TU 模块，那么其将提供刀具的数据组。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”CREATO	创建刀具
Unit：	INT	1, 2, 3 ... 10	TOA
WVar1：	INT		T 编号

14.17.4.18 PI 服务：DELECE

功能：删除一个刀沿

在此 PI 服务中通过“T 编号”参数设定现有刀具的 T 编号时，该刀具的刀沿会被删除。在此情形下，参数“D 编号”（待创建刀沿的编号）的取值范围为 00001-00009。若设定了一个正的 T 编号，但该 T 编号对应的刀具不存在，那么 PI 服务失败。若将 T 编号设定为 00000（绝对 D 编号模型），那么 D 编号的取值范围将扩展至 00001-31999。若指定的刀沿不存在，那么上述两种情形下 PI 服务均会失败。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”DELECE	删除刀沿
Unit：	INT	1, 2, 3 ... 10	TOA
WVar1：	INT		需要创建刀沿的刀具的 T 编号。赋值为 00000 时，表示不启用对刀具的参考（绝对 D 编号）
WVar2：	INT	1 ... 9	需要删除的刀沿的编号
		1 - 31999	

14.17.4.19 PI 服务：DELETO

功能：删除刀具

删除指定的 T 编号对应的刀具及所有刀沿（TO 中，有时还会在 TU、TUE、TG （类型 4xx）、TD 和 TS 模块中）

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”DELETO	删除刀具
Unit：	INT	1, 2, 3 ... 10	TOA
WVar1：	INT		T 编号

14.17.4.20 PI 服务：MMCSEM

功能

PI 服务用于测试/设置和复位 HMI 和 PLC 发出的通道专用信号量。针对每个通道提供 10 个信号量，用于保护关键数据区。

将功能（PI 服务）指定给信号量 1 至 6。信号量 7 至 10 可自由使用。

FB4 返回值

- “Done” == TRUE
  信号量已设置，可调用关键功能
- “Error” == TRUE 且 “State” == 3
  信号量已设置，关键功能目前尚无法调用。

未占用信号量的流程图
测试和设置信号量

```
IF 信号量 == 未占用
THEN
    写入/读取关键数据
    复位信号量
ELSE//信号量被禁用
...
ENDIF
```

注意
复位信号量 在通过设置信号量及之后读写数据禁用关键数据区后，必须通过复位信号量重新使能该关键数据区。

禁用信号量的流程图

```
测试和设置信号量
IF 信号量 == 未占用
THEN
    ...
ELSE//信号量被禁用
    设置“功能无法执行，需重新操作”的标志位
ENDIF
```

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	"PI".MMCSEM	处理信号量
Unit：	INT	1, 2, 3 ... 10	通道
WVar1：	INT	1)	信号量的 PI 服务专用编号
WVar2：	WORD	0, 1	任务类型 0：复位信号量 1：测试和设置信号量
1)	**编号**	**PI 服务**	
	1	TMCRTO（创建刀具）	
	2	TMFDPL（搜索空刀位用于装刀）	
	3	TMMVTL（准备刀库用于装刀，卸载刀具）	
	4	TMFPBP（搜索刀位）	
	5	TMGETT（搜索刀具编号）	
	6	TSEARC（搜索刀具）	
	7 ... 10	可自由使用	

14.17.4.21 PI 服务：TMCRTO

功能：创建刀具

通过设定名称、副编号创建刀具，例如：通过 $TC_TP1[y] = “姊妹刀具号”或 $TC_TP2[y] = “刀具名称”。或者可选择通过 T 编号创建，例如通过 y = T 编号

该刀具会在 TV 区域（刀具目录）中记录为存在，

同时会在 TO 模块中为补偿创建第一个刀沿“D1”（内容为零），

在 TS 模块中为监控数据创建第一个刀沿“D1”（内容为零），

在 TUE 模块（若存在）中为 OEM 刀沿数据创建第一个刀沿“D1”，

TD 模块提供选择设定或由 NC 分配的 T 编号所对应的名称、副编号和刀沿数量（=1）。

若存在 TU 模块，那么其将提供刀具的数据组。

执行完此 PI 任务后，创建的刀具所对应的 T 编号可在 TV 模块中的 **TnumWZV** 下获取。

说明

执行此 PI 服务的前后都必须通过带功能编号 1 (TMCRTO) 的参数 “WVar1” 调用 PI 服务 MMCSEM。参见“PI 服务：MMCSEM (页 1095)”章节。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”TMCRTO	创建刀具
Unit：	INT	1, 2, 3 ... 10	TOA
WVar1：	INT	1)	T 编号
WVar2：	INT		副编号
Addr1：	STRING	最多 32 个字符	刀具名称
1) T 编号 > 0 ⇒ 给定指定的 T 编号 T 编号 = -1 ⇒ NC 给定 T 编号			

14.17.4.22 PI 服务：TMFDPL

功能

搜索空刀位用于装刀，基于参数设置

- 刀位编号_目标 (Location_number_to) = -1，刀库编号_目标 (Magazine_number_to) = -1
  为通过 T 编号指定的刀具在设定区域（= 通道）的所有刀库下搜索空刀位。执行此 PI 服务后，找到的刀库和刀位编号保存在通道的配置块中（**magCMCmdPar1**（刀库编号）和 **magCMCmdPar2**（刀位编号））。刀位编号_名称（Location_number_ID）和刀库编号_名称（Magazine_number_ID）可用作搜索标准，也可不启用（= -1）。根据搜索结果，PI 服务会被积极或消极应答。
- 刀位编号_目标 (Location_number_to) = -1，刀库编号_目标 (Magazine_number_to) = 刀库编号
  为通过 T 编号设定的刀具在指定刀库下搜索空刀位。刀位编号_名称 (Location_number_ID) 和刀库编号_名称 (Magazine_number_ID) 可用作搜索标准，也可设为 -1。根据搜索结果，PI 服务会被积极或消极应答。
- 刀位编号_目标 (Location_number_to) = 刀位编号，刀库编号_目标 (Magazine_number_to) = 刀库编号
  检查设定的刀位是否为空刀位，能否用于装载指定的刀具。刀位编号_名称 (Location_number_ID) 和刀库编号_名称 (Magazine_number_ID) 可用作搜索标准，也可设为 -1。根据检查结果，PI 服务会被积极或消极应答。

参数 “WVar1” 和 “WVar2” 位于源中。

装刀： 源为内部装载刀库时，参数位于目标处（一个实际刀库）。

卸刀： 源始终为实际刀库。

说明

执行此 PI 服务的前后都必须通过带功能编号 2 (TMFDPL) 的参数 “WVar1” 调用 PI 服务 MMCSEM。参见 “PI 服务：MMCSEM (页 1095)” 章节。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”TMFDPL	用于装刀的空刀位
Unit：	INT	1, 2, 3 ... 10	TOA
WVar1：	INT		T 编号

信号	类型	取值范围	含义
WVar2：	INT		刀位编号_目标
WVar3：	INT		刀库编号_目标
WVar4：	INT		刀位编号_名称
WVar5：	INT		刀库编号_名称

14.17.4.23 PI 服务：TMFPBP

功能：搜索空刀位

PI 服务用于在指定刀库内搜索满足指定标准（如刀具尺寸和刀位类型）的空刀位。

搜索完成时，结果可通过以下 OPI 变量读取：

- magCMCmdPar1（刀库编号）
- magCMCmdPar2（刀位编号）

说明

只能通过 FB7 请求此 PI 服务。参见“FB7：PI_SERV2（请求 PI 服务） (页 1122)”章节。

说明

执行此 PI 服务的前后都必须通过带功能编号 4 (TMFPBP) 的参数 “WVar1” 调用 PI 服务 MMCSEM。参见“PI 服务：MMCSEM (页 1095)”章节。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	"PI".TMFPBP	搜索空刀位
Unit：	INT	1, 2 , 3 ... 10	TOA
WVar1：	INT		**开始**执行搜索的刀库的**刀库编号**
WVar2：	INT		“WVar1” 中刀库中刀位的**刀位编号**
WVar3：	INT		结束搜索的刀库的**刀库编号**
WVar4：	INT		“WVar3” 中刀库中刀位的**刀位编号**
WVar5：	INT		刀库编号参照
WVar6：	INT		刀位编号参照
WVar7：	INT	0, 1, 2 ... 7	所需的向左半刀位的数量
WVar8：	INT	0, 1, 2 ... 7	所需的向右半刀位的数量

信号	类型	取值范围	含义
WVar9：	INT	0, 1, 2 ... 7	所需的向上半刀位的数量
WVar10：	INT	0, 1, 2 ... 7	所需的向下半刀位的数量
WVar11：	INT		所需刀位类型的编号
WVar12：	INT	0, 1, 2, 3, 4	指定所需的搜索方向 0：搜索方案像 $TC_MAMP2 中一样设置 1：向前 2：向后 3：对称

示例：设置搜索区域

从	刀位	到	刀位	说明
WVar1	WVar2	WVar3	WVar4	
#M1	#P1	#M1	#P1	只检查刀库 #M1 中的刀位 #P1
#M1	#P1	#M2	#P2	在从刀库 #M1、刀位 #P1 到刀库 #M2、刀位 #P2 区域中搜索刀位
#M1	-1	#M1	-1	搜索刀库 #M1 的所有刀位。
#M1	-1	-1	-1	在从刀库 #M1 开始的区域内搜索所有刀位
#M1	#P1	-1	-1	在从刀库 #M1、刀位 #P1 开始的区域内搜索所有刀位
#M1	#P1	#M1	-1	在从刀库 #M1、刀位 #P1 开始的区域内搜索刀库 #M1 中的刀位
#M1	#P1	#M2	-1	在从刀库 #M1、刀位 #P1 到刀库 #M2 的区域内搜索刀位
#M1	-1	#M2	#P2	在从刀库 #M1 到刀库 #M2、刀位 #P2 的区域内搜索刀位
#M1	-1	#M2	-1	在从刀库 #M1 到刀库 #M2 的区域内搜索刀位
-1	-1	-1	-1	搜索所有刀库刀位

14.17.4.24 PI 服务：TMGETT

功能：确定含副编号的刀具名称的 T 编号

PI 服务用于通过刀具名称和副编号确定刀具的 T 编号。

结果被写入 TF 模块（参数设置、 TMGETT、TSEARC 的返回参数）中的 OPI 变量中：

- resultNrOfTools
  - resultNrOfTools == 0：未找到指定刀具
  - resultNrOfTools == 1：已找到指定刀具
- resultToolNr：“resultNrOfTools” == 1 时指定刀具的 T 编号

说明

执行此 PI 服务的前后都必须通过带功能编号 5 (TMGETT) 的参数 “WVar1” 调用 PI 服务 MMCSEM。参见“PI 服务：MMCSEM (页 1095)”章节。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”TMGETT	确定 T 编号
Unit：	INT	1, 2, 3 ... 10	TOA
Addr1：	STRING	最多 32 个字符	刀具名称
WVar1：	INT		副编号

14.17.4.25 PI 服务：TMMVTL

功能：准备刀库用于装刀，卸载刀具

PI 服务用于装刀、卸刀及换刀

1. 装刀和卸刀：装刀位置 ↔ 刀库
2. 装刀和卸刀：装刀位置 ↔ 中间存储器，例如：主轴
3. 刀库内的换位
4. 不同刀库间的换位
5. 刀库和中间存储器之间的换位
6. 中间存储器内的换位

情形 1、3、4 和 5：写入 TM 模块（刀库数据：通用数据）的以下 OPI 变量：

- magCmd（区域编号 = TO 单元，行 = 刀库编号）
- magCmdState ← “应答”

情形 2 和 6：写入 TMC 模块（刀库数据：配置数据）的以下 OPI 变量：

- magCBCmd（区域编号 = TO 单元）
- magCBCmdState ← “应答”

装刀

- WVar2 “刀位编号_来源”，WVar3 “刀库编号_来源”
  刀库的刀位会运行至装刀位置以进行装刀并换入刀具。
- WVar4 “刀位编号_目标” == -1
  先在刀库中搜索一个空刀位。之后，刀库的空刀位会运行至装刀位置以进行装刀并换入刀具。
  执行 PI 服务后，找到的空刀位编号位于对应通道**实际**刀库的参数 “magCMCmdPar2”（OPI 模块 TM）中。
- WVar4 “刀位编号_目标” == -2
  刀具被装入刀库的当前刀位。执行 PI 服务后，找到的刀位编号位于对应通道**实际**刀库的参数 “magCMCmdPar2”（OPI 模块 TM）中。

卸载

- WVar4 “刀位编号_目标”，WVar5 “刀库编号_目标”
  刀库的刀位会运行至装刀位置并换下刀具。

在 OPI 模块 TP（刀库数据：刀位数据）中，换出刀具的刀库位置会被标记为未占用。

刀具定址

可选择通过 T 编号或通过刀位及刀库编号进行刀具定址。未使用的参数必须输入值 -1。

说明

执行此 PI 服务的前后都必须通过带功能编号 3 (TMMVTL) 的参数 “WVar1” 调用 PI 服务 MMCSEM。参见“PI 服务：MMCSEM (页 1095)”章节。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”TMMVTL	准备刀库用于装刀，卸载刀具
Unit：	INT	1, 2, 3 ... 10	TOA
WVar1：	INT		T 编号
WVar2：	INT		刀位编号_来源
WVar3：	INT		刀库编号_来源
WVar4：	INT		刀位编号_目标
WVar5：	INT		刀库编号_目标

14.17.4.26 PI 服务：TMPOSM

功能：基于参数设置定位刀库或刀具

通过此 PI 服务将一个刀库刀位运行至设定的位置（例如装刀刀位前）。该刀位可直接指定，也可通过刀位上的刀具授权。

以下参数中给定目标（例如装刀刀位）数据：

- WVar5 “刀位编号_参照”
- WVar6 “刀库编号_参照”

根据规格在以下参数中给定待定位的刀库位置：

- WVar1 “刀具的 T 编号”
  此时不涉及以下参数：
  - Addr1 “刀具名称” = ""
  - WVar2 “副编号” = -1
  - WVar3 “刀位编号_来源” = -1
  - WVar4 “刀库编号_来源” = -1
- Addr1 “刀具名称”，WVar2 “副编号”
  此时不涉及以下参数：
  - WVar1 “刀具的 T 编号” = -1
  - WVar3 “刀位编号_来源” = -1
  - WVar4 “刀库编号_来源” = -1
- WVar3 “刀位编号_来源”，WVar4 “刀库编号_来源”
  此时不涉及以下参数：
  - Addr1 “刀具名称” = ""
  - WVar1 “刀具的 T 编号” = -1
  - WVar2 “副编号” = -1

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”TMPOSM	定位刀库或刀具
Unit：	INT	1, 2, 3 ... 10	TOA
Addr1：	STRING	最多 32 个字符	刀具名称
WVar1：	INT		刀具 T 编号
WVar2：	INT		副编号
WVar3：	INT		刀位编号_来源

信号	类型	取值范围	含义
WVar4：	INT		刀库编号_来源
WVar5：	INT		刀位编号_参照
WVar6：	INT		刀库编号_参照

14.17.4.27 PI 服务：TMPCIT

功能：设置工件计数器的增量值

设置主轴刀具工件计数器的增量提升。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”TMPCIT	设置工件计数器的增量值
Unit：	INT	1, 2, 3 ... 10	TOA
WVar1：	WORD	0, 1, 2 ... 最大	主轴编号；对应刀位数据中的类型索引，在通道中中间存储刀库的刀位类型为主轴时。 000 = 主主轴
WVar2：	WORD	0 ... 最大	增量值；设定工件计数器在多少次主轴旋转后增量提升计数。

14.17.4.28 PI 服务：TMRASS

功能：复位刀具状态“生效”

PI 服务将所有刀具状态为“生效”或“禁用”的刀具的状态设为“未生效”。

复位刀具状态的正确时间点是在发生以下事件时：

- NC/PLC 接口信号“刀具禁用未生效”的下降沿
- 程序结束
- 通道复位

此 PI 服务主要适用于 PLC，该系统中可确定禁用的刀具何时报废。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”TMRASS	复位生效状态
Unit：	INT	1, 2, 3 ... 10	TO 区域

14.17.4.29 PI 服务：TRESMO

功能：复位监控值

通过此 PI 服务可将对指定刀具的指定刀沿的监控值复位为设定值/初始值。
其只适用于激活了监控的刀具。

相关信息也请参见 RESETMON 指令。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”TRESMO	复位监控值
Unit：	INT	1, 2, 3 ... 10	TO 区域
WVar1：	WORD	- max ... max	刀具编号 0：加工所有刀具 > 0：加工指定刀具 < 0：加工所有姊妹刀具
WVar2：	WORD	0, 1, 2 ... 最大	D 编号 < 0：复位指定刀沿的监控 0：复位所有刀沿的监控
Wvar3：	WORD		需要复位的监控方式（包括所有组合）：
			位 0 = 1：刀具寿命监控
			位 1 = 1：工件数监控
			位 2 = 1：磨损监控
			位 3 = 1：补偿和监控
			0：所有生效的刀具监控 ($TC_TP9)。

14.17.4.30 PI 服务：TSEARC

功能：基于参数设置通过搜索掩码进行复杂搜索

PI 服务用于在一个搜索区域（一个或多个刀库，从一个特定刀位开始，到一个特定刀位结束）内搜索具有指定属性的刀具

说明

生效的刀具管理

仅当刀具管理生效时，PI 服务才可用。

设定方式

- 搜索方向
- 搜索下一个具有指定属性的刀具
- 搜索所有具有指定属性的刀具

结果

用户将得到一个结果列表，其中包含搜索到的刀具的内部 T 编号。

逻辑运算方法

只提供了逻辑与运算用于筛选属性。逻辑或运算必须由用户通过多次调用 PI 服务并逐一分析结果才可实现。

刀具属性的参数设置

在 OPI 模块 TF（参数设置，TMGETT、TSEARC 的返回参数）中通过以下变量设置找到的刀具的属性：

- “parMask<X>”（参数设置窗口）
- “parData<X>”（比较值）

其中 <X> = TAD、TAO、TAS、TD

结果列表

PI 服务结束后，搜索结果位于 OPI 模块 TF（参数设置，TMGETT、TSEARC 的返回参数）的以下变量中：

- “resultCuttingEdgeNrUsed”（从上一次工件计数起所用刀沿的 D 编号）
- “resultNrOfCutEdgesUsed”（从上一次工件计数起的刀沿数量）
- “resultNrOfTools”（已找到的刀具的数量）
- “resultToolNr”（已找到的刀具的 T 编号）
- “resultToolNrUsed”（从上一次工件计数起所用刀沿的 T 编号）

如果未找到刀具，已找到的刀具数为零 (“resultNrOfTools” == 0)

搜索区域设定

从刀库编号 WVar1	从刀位编号 WVar2	到刀库编号 WVar3	到刀位编号 WVar4	说明 搜索以下刀库位置：
#M<a>	#P<b>	#M<x>	#P<y>	从：刀库 #M<a>，刀位 #P<b> 到：刀库 #M<x>，刀位 #P<y>
#M<a>	-1	#M<a>	-1	从：刀库 #M<a>，第一个刀位 到：刀库 #M<a>，最后一个刀位
#M<a>	-1	-1	-1	从：刀库 #M<a>，第一个刀位 到：最后一个刀库，最后一个刀位
#M<a>	#P<b>	-1	-1	从：刀库 #M<a>，刀位 #P<b> 到：最后一个刀库，最后一个刀位
#M<a>	#P<b>	#M<a>	-1	从：刀库 #M<a>，刀位 #P<b> 到：刀库 #M<a>，最后一个刀位
#M<a>	#P<b>	#M<x>	-1	从：刀库 #M<a>，刀位 #P<b> 到：刀库 #M<b>，最后一个刀位
#M<a>	-1	#M<x>	#P<y>	从：刀库 #M<a>，第一个刀位 到：刀库 #M<b>，刀位 #P<y>
#M<a>	-1	#M<x>	-1	从：刀库 #M<a>，第一个刀位 到：刀库 #M<x>，最后一个刀位
-1	-1	-1	-1	从：第一个刀库，第一个刀位 到：最后一个刀库，最后一个刀位

对称搜索

根据刀位执行对称搜索须满足下列前提条件：

- 搜索区域中只包含一个刀库：“WVar1”（从：刀库编号）== “WVar3”（到：刀库编号）
- 给定围绕其进行对称搜索的参考刀位，即刀位：“WVar5”（参考刀库的编号）和“WVar6”（参考刀位的编号）
- 对于参考刀位，必须在 TPM 模块中为其配置对待搜索刀库的重复指定。
- “WVar7”（搜索方向）= 3

参考刀位可以是一个中间存储器刀位（刀库中间存储器中的刀位，即切换位置、机械手等）或一个装刀刀位/位置（内部装刀刀库中的刀位）。对称搜索会以刀位为基准运行至参考刀位前。

若参考刀位前的刀位超出了搜索区域，那么此服务的特性将与未找到匹配刀位时相同。

说明

执行此 PI 服务的前后都必须通过参数“WVar1”调用 PI 服务 MMCSEM。参见“PI 服务：MMCSEM (页 1095)”章节。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”TSEARC	通过搜索掩码进行复杂搜索
Unit：	INT	1, 2, 3 ... 10	TOA
WVar1：	INT		从：刀库编号
WVar2：	INT		从：刀位编号
WVar3：	INT		到：刀库编号
WVar4：	INT		到：刀位编号
WVar5：	INT		参考刀库的编号 （仅涉及对称搜索： 搜索方向 == 3）
WVar6：	INT		参考刀位的编号 （仅涉及对称搜索： 搜索方向 == 3）

信号	类型	取值范围	含义
WVar7：	INT	1, 2, 3	搜索方向： 1：从搜索区域的第一个刀位向前 2：从搜索区域的最后一个刀位向后 3：与位于在 “WVar5”（参考刀库的编号）和 “WVar6”（参考刀位的编号）中指定的位置**前**的实际刀库位置对称
WVar8：	INT	0, 1, 2, 3	搜索方式 在搜索区域中搜索属性： 0：所有刀具，刀沿专用 1：第一个刀具，刀沿专用 2：所有刀具，通过所有刀沿 3：第一个刀具，通过所有刀沿

14.17.4.31 PI 服务：TMCRMT

功能：创建多刀

PI 服务用于创建一个含指定名称、可设定多刀编号/刀位数及间距编码方式的新多刀。

说明

执行此 PI 服务的前后都必须通过参数 “WVar1” =1 (TMCRTO) 调用 PI 服务 MMCSEM。参见 “PI 服务：MMCSEM (页 1095)” 章节。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”TMCRMT	创建多刀
Unit：	INT	1, 2, 3 ... 10	TOA
Addr1：	STRING		多刀名称（最多 32 个字符）
WVar1：	INT	0	预留
WVar2：	INT	-1, 1, 2 ... 32000	多刀编号 -1：由 NC 自动给定多刀编号 1, 2, 3 ... 32000：多刀编号， **提示**：多刀编号在整个系统中必须是唯一的

信号	类型	取值范围	含义
WVar3：	INT	2, 3, 4 ... 最大	刀位数 MAX = 在 MD17504 $MN_MAX_TOOLS_PER_MULTITOOL 中设置的数量
WVar4：	INT	1, 2, 3	距离编码方式

14.17.4.32 PI 服务：TMDLMT

功能：删除多刀

此 PI 服务用于在所有存储多刀的数据块中删除多刀。之后多刀中配备的刀具将不再包含在多刀中且不会载入，但定义保留。

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”TMDLMT	删除多刀
Unit：	INT	1, 2, 3 ... 10	TOA
Addr1：	STRING		多刀名称（最多 32 个字符）
WVar1：	INT	0	预留
WVar2：	INT	-1, 1, 2 ... 32000	多刀编号 -1：删除 “Addr1” 中设定名称所对应的多刀 1, 2, 3 ... 32000：删除带指定多刀编号的多刀； **提示：**不分析参数 “Addr1”
WVar3：	INT	0, 1	多刀中包含的刀具： 0：不删除 1：删除

14.17.4.33 PI 服务：POSMT

功能：定位多刀

此 PI 服务用于将多刀定位至编写的刀位，或定位至编写的、位于多刀其中一个刀位上的刀具。刀具可通过 T 编号指定，也可通过名称和副编号指定。仅当多刀位于一个刀架刀位上，且无基于该刀架刀位的刀具补偿生效时，才能对其进行定位。

位置给定

可通过以下数据给定位置：

编号	Addr1	WVar1	WVar2	WVar3	WVar4
1	空字符串	刀架编号	刀具编号	-1	
2	刀具名称	刀架编号	-1	副编号	
3	空字符串	刀架编号	-1	-1	多刀刀位号

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”POSMT	定位多刀
Unit：	INT	1, 2, 3 ... 10	TOA
Addr1：	STRING		多刀中需定位的刀具的名称（最多 32 个字符）
WVar1：	INT	1, 2, 3 ... 999	刀架编号
WVar2：	INT	-1, 1, 2, 3 ... 32000	多刀中需定位的刀具的编号（T 编号） 1, 2, 3 ... 32000：刀具编号 -1：刀具编号不相关，必须编写 “Addr1” 和 “WVar3”
WVar3：	INT	-1, 1, 2, 3 ... 32000	多刀中需定位的刀具的副编号 1, 2, 3 ... 32000：副编号 -1：副编号不相关，必须编写 “WVar2”
WVar4：	INT	1, 2, 3 ... 999	定位的目标刀位的多刀刀位编号

14.17.4.34 PI 服务：FDPLMT

功能：搜索/检查多刀内的空刀位

PI 服务用于在多刀内搜索一个用于接收指定刀具的未占用的刀位或检查指定的用于接收指定刀具的刀位是否被占用。刀具可通过 T 编号设定，也可通过名称和副编号设定。

说明

执行此 PI 服务的前后都必须通过参数 “WVar1” =x (FDPLMT) 调用 PI 服务 MMCSEM。参见“PI 服务：MMCSEM (页 1095)”章节。

位置给定

可通过以下数据给定需要在多刀中定位的刀具：

- “Addr1” = <空字符串>，“WVar1” = <刀具编号>，“WVar2” = <副编号>
- “Addr1” = <刀具名称>，“WVar1” = -1 和 “WVar2” = <副编号>
- “Addr1” = <空字符串>，“WVar1” = <刀具编号>，“WVar2” = -1

可通过以下数据给定位置：

形式参数说明

信号	类型	取值范围	含义
PIService：	ANY	“PI”FDPLMT	搜索/检查多刀内的空刀位
Unit：	INT	1, 2, 3 ... 10	TOA
Addr1：	STRING		多刀中需定位的刀具的名称（最多 32 个字符）
WVar1：	INT	-1, 1, 2, 3 ... 32000	多刀中需定位的刀具的编号（T 编号） -1：刀具编号不相关，必须编写 “Addr1” 和 “WVar2” 1, 2, 3 ... 32000：刀具编号
WVar2：	INT	-1, 1, 2, 3 ... 32000	多刀中需定位的刀具的副编号 1, 2, 3 ... 32000：副编号 -1：副编号不相关，必须编写 “WVar1”
WVar3：	INT	-1, 1, 2, 3 ... 32000	多刀编号 1, 2, 3 ... 32000：多刀编号 -1：针对所有多刀搜索空刀位，或针对所有多刀检查用于接收刀具的指定刀位是否被占用
WVar4：	INT	-1, 1, 2, 3 ... 999	定位的目标刀位的多刀刀位编号 1, 2, 3, ... 999：多刀刀位号 -1：在多刀内搜索任意一个空刀位

14.17.5 FB5：GETGUD - 读取 GUD 变量

功能

功能块 FB5 “GETGUD” 用于读取 NC 或通道中的全局用户数据 (GUD)。

请求读取 NC 变量

通过上升沿切换调用 FB5，参数 “Req” = 0 → 1

参数 “Addr”：指针指向采用“<数据块>”.<变量名称>符号形式的 GUD 变量的名称。

参数 “Area”、“Unit”、“Index1” 和 “Index2”：用于变量定址的附加信息

激活参数 “CnvtToken” 可为待读取的 GUD 变量获得一个 Token（变量指针）。之后可通过 FB2 和 FB3（参数 “Addr1” ... “Addr8” = “<Token>”）读取或写入 GUD 变量。写入 GUD 变量要求必须借助 Token 进行定址。使用 Token 进行 GUD 变量区定址时，还需额外考虑到 FB2/FB3 的参数 “Line1” ...“Line8” = “<Feldindex>”。

为了在不调整格式的前提下读取 “DOUBLE” 类型的 GUD 变量，必须在目标区域中编写一个指向 REAL 2 类型的 ANY 指针，例如：P#M100.0 REAL 2。读取 “DOUBLE” 类型的 GUD 变量的值时，该数据将作为 64 位浮点值接收至 PLC 数据区。

完成读取请求

读取请求完成：参数 “Done” == 1。

读取请求出错：参数 “Error” == 1，故障原因在参数 “State” 中显示

前提条件

- 通过 OB100 使能 NC/PLC 通讯，FB1 参数 “NCKomm” = 1
- 对于接收包含 GUD 变量名称的字符串的数据块，必须通过 S7 项目的 S7 符号列表中的 S7 符号编辑器为其指定一个符号（比较以下调用示例 1：DB_GUDVAR）。数据块中必须创建长度合适的字符串，接收 GUD 变量名称（比较以下调用示例 1："DB_GUDVAR".GUDVar1）。然后，该符号传输至 FB5 的参数“Addr”中，例如：Addr := "DB_GUDVAR".GUDVar1。如果 GUD 变量的 Token 需要用于 FB2/FB3 的后续调用，则必须在 DB（例如：相同的 DB）中创建 10 字节的 Token 结构（比较以下调用示例 1："DB_GUDVAR".GUDVar1Token）。该结构在 FB5 的参数 VarToken 中详细说明，比较以下调用示例 1：FB5, VarToken := "DB_GUDVAR".GUDVar1Token。调用 FB2/FB3 时，该 Token 在参数 "Addr1" … "Addr8" 中详细说明，比较以下调用示例 1：FB3, Addr1 := "GUDVar1Token"。

边界条件

- FB5 支持多实例
- 每次调用 FB5 都必须指定一个独立的、用户区域中的实例 DB。

- 只允许在 PLC 基本程序 (OB1) 的循环分量重读取 GUD 变量（通过 Req = 1 调用 FB5）。如果未启动任务（通过 Req = 0 调用 FB5），则也可在 PLC 基本程序 (OB100) 的启动分量中写入参数。之后可通过已写入的参数在 PLC 基本程序 (OB1) 的循环分量中执行该任务（通过 Req = 1 调用 FB5）。
- GUD 变量的名称只允许采用大写字母。
- 读取 GUD 变量通常会持续多个 PLC 循环。

说明

故障

从不同的通道中读取变量时会产生以下反馈信息：

- “Error” == TRUE
- “State” == W#16#02

说明

为了在不调整格式的前提下从 NC 读取 “DOUBLE” 类型的变量，必须在目标区域中设定一个指向 REAL 2 类型的 ANY 指针（例如：P#M100.0 REAL 2）。读取 “DOUBLE” 类型的变量时，若基本程序将其作为目标类型 REAL 2 识别，那么该数据将作为 64 位浮点值接收至 PLC 数据区域。

功能说明

```
FUNCTION_BLOCK FB5          //服务器名称
   KNOW_HOW_PROTECT
   VERSION: 3.0
VAR_INPUT
   Req :          BOOL;
   Addr :         ANY;       //变量名称
   Area           BYTE;      // 区域：NCK = 0，通道 = 2
   Unit :         BYTE;
   Index1 :       INT;       //数组索引 1
   Index2 :       INT;       //数组索引 2
   CnvtToken :    BOOL;      //转换为 10 字节 Token
   VarToken       ANY;       //10 字节结构，用于变量 Token
END_VAR

VAR_OUTPUT
   Error :        BOOL;
```

```
   Done :         BBOL;
   State :        WORD;
END_VAR

VAR_IN_OUT
   RD :           ANY;
END_VAR

BEGIN
END_FUNCTION_BLOCK
```

对形式参数的说明

信号	方式	类型	取值范围	含义
Req：	E	BOOL		通过上升沿启动任务
Addr：	E	ANY	“<数据块名称>”.<变量名称>	STRING 型变量中的变量名称
Area：	E	BYTE	0, 2	区域： 0：NC 2：通道
Unit：	E	BYTE	1, 2, 3, ... 10	区域 == NC：Unit：= 1 区域 == 通道：通道编号
索引 1：	E	INT	0, 1, 2, ... <最大数组索引>	数组索引 1 无数组索引的变量：索引 1 = 0
索引 2：	E	INT	0, 1, 2, ... <最大数组索引>	数组索引 2 无第 2 个数组索引的变量： 索引 2 = 0
CnvtToken：	E	BOOL	0 (FALSE), 1 (TRUE)	激活生成一个 10 字节的变量 Token
VarToken：	E	ANY	“<数据块名称>”.<变量名称>	10 字节 Token 的地址（见示例）
Error：	A	BOOL	0 (FALSE), 1 (TRUE)	任务受到消极应答或无法执行
Done：	A	BOOL	0 (FALSE), 1 (TRUE)	任务已成功执行。

信号	方式	类型	取值范围	含义
State：	A	WORD	---	参见“故障标识”
RD：	I/O	ANY	P#Mm.n BYTE x ... P#DBnr.dbxm.n BYTE x	读取数据的目标区域

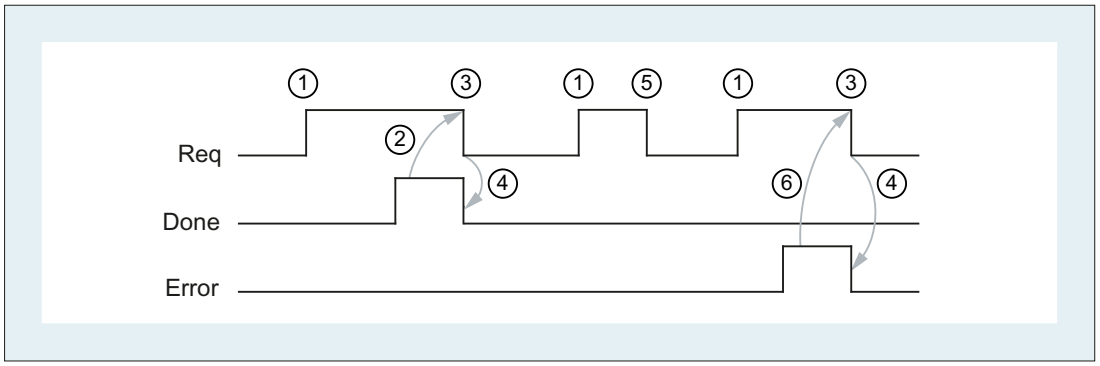
故障标识

State		含义	提示
WORT-H [1]	WORT-L		
0	1	访问故障	
0	2	任务中出错	任务中变量的组合错误
0	3	消极应答，任务无法执行	内部故障，可能的解决办法： NC 复位
0	4	数据区域或数据类型不一致	在 RD 中检查 待读取数据；
1	4	本地用户存储空间不足	读取的变量比 RD 中设定的要长
0	6	FIFO 已满	必须重复任务，因为等待序列已满
0	7	选件未设置	基本程序参数“NCKomm”未设置
0	8	目标区域错误（SD）	RD 不允许为本地数据
0	9	传输被占用	必须重复任务
0	10	定址时出错	Unit 包含 0 值
0	11	变量地址无效	检查 Addr（或变量名称）、Area、Unit
1 - 8	13	ANY 数据的参照错误	所需的 String 或 NcVar 数据未设置
0	15	字符串长度超过 32 个字符	GUD 变量名称过长
1) 高位字节 > 0 ⇒ 出现故障的变量的编号			

配置步骤

为了读取 GUD 变量，必须将该 GUD 变量的名称保存在一个 String 类型的变量中。包含该 String 变量的数据块必须在符号列表中进行定义，从而可以针对 FB GETGUD 以符号形式指定“Addr”。可选择在 PLC 的任意数据区域定义一个结构变量，用于接收变量指针，参见以下示例中的设定。

脉冲示意图



① 用户：置位请求，Req = 0 → 1

② FB5 成功执行，Done = 1
用户：复位请求，如果 Done == 1 则 Req = 0

③ 用户：如果 Done == 1 则复位请求：1 → 0

④ FB5 复位任务确认，Done = 0

⑤ 用户：如果 Done == 0 且 Error == 0 则**不允许**复位请求 Req = 1 → 0

⑥ FB5 执行时出错，Error = 1
用户：复位请求，如果 Done == 1 或 Error == 1 则 Req = 0，可能需要执行故障处理

调用示例 1

从通道 1 中读取 GUD 变量：

- 名称“GUDVAR1”
- 类型：INTEGER

转换成一个 10 字节变量指针。参见“FB2：GET - 读取 NC 变量 (页 1061)”一章中的表格“数据类型对应关系”。

读取 GUD 变量：带实例 DB111 的 FB5

```
//GUD 变量的数据块
DATA_BLOCK DB_GUDVAR    //在符号列表中进行指定

STRUCT
   GUDVar1 :STRING[32] := 'GUDVAR1';                    //名称由用户定义
   GUDVar1Token :
   STRUCT
       SYNTAX_ID :BYTE;
       区域和单位:BYTE;
       列 :WORD;
       行 :WORD;
       模块类型 :BYTE;
       行数 :BYTE;
       类型 :BYTE;
       长度:BYTE;
   END_STRUCT;
END_STRUCT;

BEGIN
END_DATA_BLOCK

//自由用户 DB，作为 FB5 的实例
DATA_BLOCK DB111

        FB5
BEGIN
END_DATA_BLOCK

//自由用户 DB，作为 FB3 的实例
DATE_BLOCK DB112

        FB3
BEGIN
END_DATA_BLOCK

//读取通道 1 中的通道专用 GUD 变量并转换成一个变量指针
功能 FC "VariablenCall" :VOID
U      E 7.7;            //机床控制面板自由按键
S      M 100.0;          //激活 Req
U      M 100.1;          //Done 结束消息
```

```
R      M 100.0;          //结束任务
U      E 7.6;            //手动应答故障
U      M 102.0;          //故障存在
R      M 100.0;          //结束任务
CALL FB5, DB111(
            Req          := M 100.0,              //触发读取的脉冲沿
            Addr         := "DB_GUDVAR".GUDVar1, // GUD 名称
            Area         := B#16#2,               //通道变量
            Unit         := B#16#1,               //通道 1
            Index1       := 0,                    //无数组索引 1
            Index2       := 0,                    //无数组索引 2
            CnvtToken    := TRUE,                 // 请求：转换为 10 字节 Token
            VarToken     := "DB_GUDVAR".GUDVar1Token, // Tokens 的地址
            Error        := M 102.0,
            Done         := M 100.1,
            State        := MW 104,
            RD           := P#DB99.DBX0.0 DINT 1 //自由存储空间
);
```

写入 GUD 变量：FB3，带实例 DB112

FB5 中的 GUD 变量 Token，用于通过 FB3 写入的参数：“VarToken”，参数 “Addr1”

```
CALL FB3, DB112(
            Req          := M 200.0,
            NumVar       := 1,                      //一个 GUD 变量
            Addr1        :=                         // Token
                         "DB_GUDVAR".GUDVar1Token,
            Error        := M 102.0,
            Done         := M 100.1,
            State        := MW 104,
            SD1          := P#DB99.DBX0.0 DINT 1
);
```

调用示例 2

从通道 1 中读取 GUD 变量

- 名称“GUD_STRING”
- 类型：STRING[32]

转换成一个 10 字节变量指针。

读取 GUD 变量：带实例 DB111 的 FB5

```
//GUD 变量的数据块
DATA_BLOCK DB_GUDVAR    //在符号列表中进行指定

STRUCT
    GUDVarS :STRING[32] := 'GUD_STRING';     //名称由用户定义
    GUDVarSToken :
    STRUCT
       SYNTAX_ID :BYTE;
       区域和单位:BYTE;
       列 :WORD;
       行 :WORD;
       模块类型 :BYTE;
       行数 :BYTE;
       类型 :BYTE;
       长度:BYTE;
    END_STRUCT;

    string_of_GUD :STRING[30];      //长度至少须为
                                    //"GUD_STRING" 的定义!
    new_name :STRING[30] := 'GUD_123';
END_STRUCT;

BEGIN
END_DATA_BLOCK

//自由用户 DB，作为 FB5 的实例
DATA_BLOCK DB111

        FB5
BEGIN
END_DATA_BLOCK

//自由用户 DB，作为 FB3 的实例
DATE_BLOCK DB112

        FB3
BEGIN
END_DATA_BLOCK

//读取通道 1 中的通道专用 GUD 变量并转换成一个变量指针
```

```
功能 FC "VariablenCall" :VOID
U      E 7.7;            //机床控制面板自由按键
S      M 100.0;          //激活 Req
U      M 100.1;          //Done 结束消息
R      M 100.0;          //结束任务
U      E 7.6;            //手动应答故障
U      M 102.0;          //故障存在
R      M 100.0;          //结束任务
CALL FB5, DB111(
            Req          := M 100.0,     //触发读取的脉冲沿
            Addr         := "DB_GUDVAR".GUDVarS,
            Area         := B#16#2,      //通道变量
            Unit         := B#16#1,      //通道 1
            Index1       := 0,           //无数组索引
            Index2       := 0,           //无数组索引
            CnvtToken    := TRUE,        // 请求：转换为 10 字节 Token
            VarToken     := "DB_GUDVAR".GUDVarSToken, // Tokens 的地址
            Error        := M 102.0,
            Done         := M 100.1,
            State        := MW 104,
            RD           := "DB_GUDVAR".string_of_GUD
);
```

写入 GUD 变量：FB3，带实例 DB112

FB5 中的 GUD 变量 Token，用于通过 FB3 写入的参数：“VarToken”，参数 “Addr1”

```
CALL FB3, DB112(
            Req          := M 200.0,
            NumVar       := 1,                  //一个 GUD 变量
            Addr1        := "DB_GUDVAR".GUDVarSToken, // Token
            Error        := M 102.0,
            Done         := M 100.1
            State        := MW 104,
            SD1          := "DB_GUDVAR".new_name
);
```

数据类型对应关系

参见“FB2：GET - 读取 NC 变量 (页 1061)”一章中的表格“数据类型对应关系”。

14.17.6 FB7：PI_SERV2（请求 PI 服务）

功能

功能块 FB7 的功能与 FB4 大致相同，但是 “WVar” 参数 (“WVar11” - “WVar16”) 的数量更多。因此建议使用功能块 FB7 替代 FB4。

详细说明参见章节 “FB4:PI_SERV - 请求 PI 服务 (页 1080)”。

功能说明

```
FUNCTION_BLOCK FB7
Var_INPUT
    Req :          BOOL;
    PIService :    ANY;
    Unit :         INT;
    Addr1 :        ANY;
    Addr2 :        ANY;
    Addr3 :        ANY;
    Addr4 :        ANY;
    WVar1 :        WORD;
    WVar2 :        WORD;
    WVar3 :        WORD;
    WVar4 :        WORD;
    WVar5 :        WORD;
    WVar6 :        WORD;
    WVar7 :        WORD;
    WVar8 :        WORD;
    WVar9 :        WORD;
    WVar10 :       WORD;
    WVar11:        WORD;
    WVar12:        WORD;
    WVar13:        WORD;
    WVar14:        WORD;
    WVar15:        WORD;
    WVar16:        WORD;
END_VAR
VAR_OUTPUT
    Error :        BOOL;
    Done :         BOOL;
    State :        WORD;
END_VAR
```

14.17.7 FB9: MzuN - 操作单元切换

功能

功能块 FB9 “MtoN” 用于对通过总线系统连接至一个/多个 NCU 控制系统模块的多个操作单元（MCP / OP）进行切换。

Die Schnittstelle zwischen den einzelnen Bedieneinheiten und der NCU (PLC) ist die M :N-Schnittstelle im Datenbaustein DB19.FB9 使用这些接口的信号进行工作。

除了初始化、生命符号监控和模块的纠错程序，还可针对操作单元切换执行下列基本功能：

功能列表：	
基本功能	**含义**
HMI 呼叫等待	HMI 期望连接 NCU 进入在线模式
HMI 连接	HMI 建立与 NCU 的连接
HMI 断开	HMI 从 NCU 断开
排挤	HMI 必须解除与 NCU 的连接
服务器模式下的操作焦点切换	操作焦点从一个 NCU 切换至另一个
主动/被动操作模式	操作和监控/只进行监控
MCP 切换	可选择通过 HMI 切换 MCP

说明

此功能块必须通过用户程序调用。为此用户须提供一个任意编号的实例 DB。此调用不支持多实例。

对部分重要功能的简要说明

主动/被动操作模式：

在线 HMI 可启用两种不同的操作模式：

主动模式： 操作人员可操作和监控

被动模式： 操作人员可进行监控（仅 HMI 的开头行）

切换至 NCU 后，先在在线 NCU 的 PLC 中请求主动操作模式。若一个 NCU 上同时连接了两个在线 HMI，那么这两个 HMI 总是一个处于主动操作模式，另一个处于被动操作模式。操作人间可在被动 HMI 上通过按键请求主动操作模式。

MCP 切换

指定给 HMI 的 MCP 可与该 HMI 一同进行切换。前提条件是将 MCP 地址输入到了 HMI 配置文件 NETNAMES.INI 的参数“mstt_adress”中，且设置了 "MCPEnable" = TRUE。被动 HMI 的 MCP 会被取消激活。因此一个 NCU 上始终只有一个生效的 MCP。

启动条件

Um zu verhindern, dass bei einem Neustart der NCU, die davor zuletzt angewählte MCP aktiviert wird, muss beim Aufruf von FB1 in OB100 der Eingangsparameter "MCP1BusAdr" = **255** (Adresse 1.MCP) und "MCP1Stop" **= TRUE** (1. MCP ausschalten) gesetzt werden.

使能

从一个 MCP 切换至另一个时，可能已启动的进给使能或轴使能会被保留。

说明

切换的同时按下的键会保持生效，直至激活新的 MCP（通过 HMI，之后激活）。此外进给率、主轴的倍率位置也会保留。为了取消按下的键，必须在信号 DB10.DBX104.0（OPI 1 就绪）的下降沿将机床控制信号的输入映射切换至未执行的信号电平。倍率位置可保持不变。必须在 PLC 用户程序中执行取消按键的措施（参见“倍率切换”示例）。

调用只允许在循环程序 OB1 中执行。

功能说明

```
FUNKTION_BLOCK FB9
VAR_INPUT
    Quit:           BOOL;           //应答报警
    OPMixedMode:    BOOL:=FALSE;    //通过不支持 MtoN 的 OP 取消//混合运行
    ActivEnable:    BOOL:=TRUE;     //不支持
    MCPEnable:      BOOL:=TRUE;     //激活 MCP 切换
END_VAR
VAR_OUTPUT
    Alarm1:         BOOL;           // Alarm:Fehler HMI-Busadresse, Bustyp!
    Alarm2:         BOOL;           // Alarm:Keine Bestätigung HMI 1 offline!
    Alarm3:         BOOL;           // Alarm:HMI 1 geht nicht offline!
    Alarm4:         BOOL;           // Alarm:Keine Bestätigung HMI 2 offline!
    Alarm5:         BOOL;           // Alarm:HMI 2 geht nicht offline!
    Alarm6:         BOOL;           // Alarm:Anklopf- HMI geht nicht online!
```

```
    Report:         BOOL;           // Meldung:生命符号监控
    ErrorMMC:       BOOL;           //HMI 故障标识
END_VAR
```

对形式参数的说明

信号	方式	类型	值域	含义
Quit：	E	BOOL	0 (FALSE), 1 (TRUE)	应答报警
OPMixedMode：	E	BOOL	0 (FALSE), 1 (TRUE)	通过不支持 MtoN 的 OP 取消混合运行
ActivEnable：	E	BOOL	0 (FALSE), 1 (TRUE)	不支持该功能。在 DB19 中通过 MMCx_SHIFT_LOCK 禁止操作区域切换
MCPEnable：	E	BOOL	0 (FALSE), 1 (TRUE)	激活 MCP 切换 1：MCP 随操作区域切换。 0：MCP 不随操作区域切换。可借此绑定 MCP。另见 DB19 中的 MMCx_MSTT_SHIFT_LOCK。
Alarm1：	A	BOOL	0 (FALSE), 1 (TRUE)	报警：Fehler HMI -Busadresse, Bustyp!
Alarm2：	A	BOOL	0 (FALSE), 1 (TRUE)	报警：Keine Bestätigung HMI 1 offline!
Alarm3：	A	BOOL	0 (FALSE), 1 (TRUE)	报警：HMI 1 geht nicht offline!
Alarm4：	A	BOOL	0 (FALSE), 1 (TRUE)	报警：Keine Bestätigung HMI 2 offline!
Alarm5：	A	BOOL	0 (FALSE), 1 (TRUE)	报警：HMI 2 geht nicht offline!
Alarm6：	A	BOOL	0 (FALSE), 1 (TRUE)	报警：Anklopf- HMI geht nicht online!
Report：	A	BOOL	0 (FALSE), 1 (TRUE)	信息：HMI 生命符号监控
ErrorMMC：	A	BOOL	0 (FALSE), 1 (TRUE)	HMI 故障标识

FB9 调用示例

```
CALL FB9, DB109(
Quit            := Fehler_Quitt,    //例如：MCP 复位
```

```
    OPMixedMode     := FALSE,
    ActivEnable     := TRUE,
    MCPEnable       := TRUE);          //使能 MCP 切换
```

说明

输入参数“MCPEnable”必须为 TRUE，以支持 MCP 切换。此参数的缺省值即为此值，调用功能时无需特别赋值。

报警、故障

输出参数“Alarm1”至“Alarm6”及“Report”在 PLC 中对 MtoN 功能中出现的故障进行提示，且会在 HMI 上以报警 410900 - 410906 的形式输出。

若一项 HMI 功能无法执行（且无法显示对应的故障消息），则会在状态参数“ErrorMMC”上以逻辑 1 显示（例如尚未建立连接时的启动故障）。

FB 的调用示例（在 OB100 中调用）

```
CALL "RUN_UP" , "gp_par" (
    MCPNum                := 1,
    MCP1In                := P#E 0.0,
    MCP1Out               := P#A 0.0,
    MCP1StatSend          := P#A 8.0,
    MCP1StatRec           := P#A 12.0,
    MCP1BusAdr            := 255,              // 地址 1MCP
    MCP1Timeout           := S5T#700MS,
    MCP1Cycl              := S5T#200MS,
    MCP1Stop              := TRUE,             //MCP 关闭
    NCCyclTimeout         := S5T#200MS,
    NCRunupTimeout        := S5T#50S);
```

示例：Override-Umschaltung

```
//使用辅助标志位 M100.0、M100.1、M100.2、M100.3
//MCP1Ready 为上升沿时必须检查倍率
//并激活措施
//启动 MCP 模块
//此示例适用于进给倍率；
//对于主轴倍率，必须更换接口字节和输入字节
U     DB10.DBX104.0;      // MCP1Ready
```

```
FN    M   100.0;          //脉冲沿标志位 1
SPBN wei1;
S     M   100.2;          //设置辅助标志位 1
R     M   100.3;          //复位辅助标志位 2


//保存倍率
      L DB21.DBB4;        //进给倍率接口
      T EB28;             //中间存储器（自由输入字节
                          //或标志位字节）

wei1:
U     M   100.2;          //进行切换
O     DB10.DBX104.0;      // MCP1Ready
SPBN wei2;
U     DB10.DBX104.0;      // MCP1Ready
FP    M   100.1;          //脉冲沿标志位 2
SPB wei2;
U     M   100.2;          //进行切换
R     M   100.2;          //复位辅助标志位 1
SPB wei2;
U     M   100.3;          //校验完成
SPB MCP;                  //调用 MCP 程序
//在经过切换的 MCP 的接口上控制保存的倍率
//直至倍率值一致
      L EB28;             //打开中间存储器
      T DB21.DBB4;        //接口倍率控制
      L EB3;              //进给率的倍率输入字节
      <>i;                //是否一致?
SPB wei2;                 //否，跳转
S     M   100.3;          //是，设置辅助标志位 2
//倍率值一致后重新调用 MCP 程序
MCP: CALL "MCP_IFM" (     // FC19
      BAGNo         := B#16#1,
      ChanNo        := B#16#1,
      SpindleIFNo   := B#16#0,
      FeedHold      := M 101.0,
      SpindleHold   := M 101.1);
wei2:NOP            0;
```

14.17.8 FB10：安全继电器（SI-Relais）

功能

PLC 中用于 “Safety Integrated” 的 SPL 功能块 FB10 “安全继电器”与同名的 NC 功能具有同等效力。标准 SPL 模块“安全继电器”用于通过安全可编程逻辑请求急停。不过该模块也可用于其他类似请求，例如防护门控制。

此功能包含 3 个输入参数（In1、In2、In3）。其中一个输入参数切换为 0 值时，输出 Out0 会立即断开，输出 Out1、Out2 和 Out3 则根据设置的时间值延迟断开（参数 TimeValue1、TimeValue2、TimeValue3）。输入 In1 至 In3 取值 1 且在应答输入 Quit1、Quit2 的其中一个识别出上升沿时，输出会立即重新接通。

调用标志位

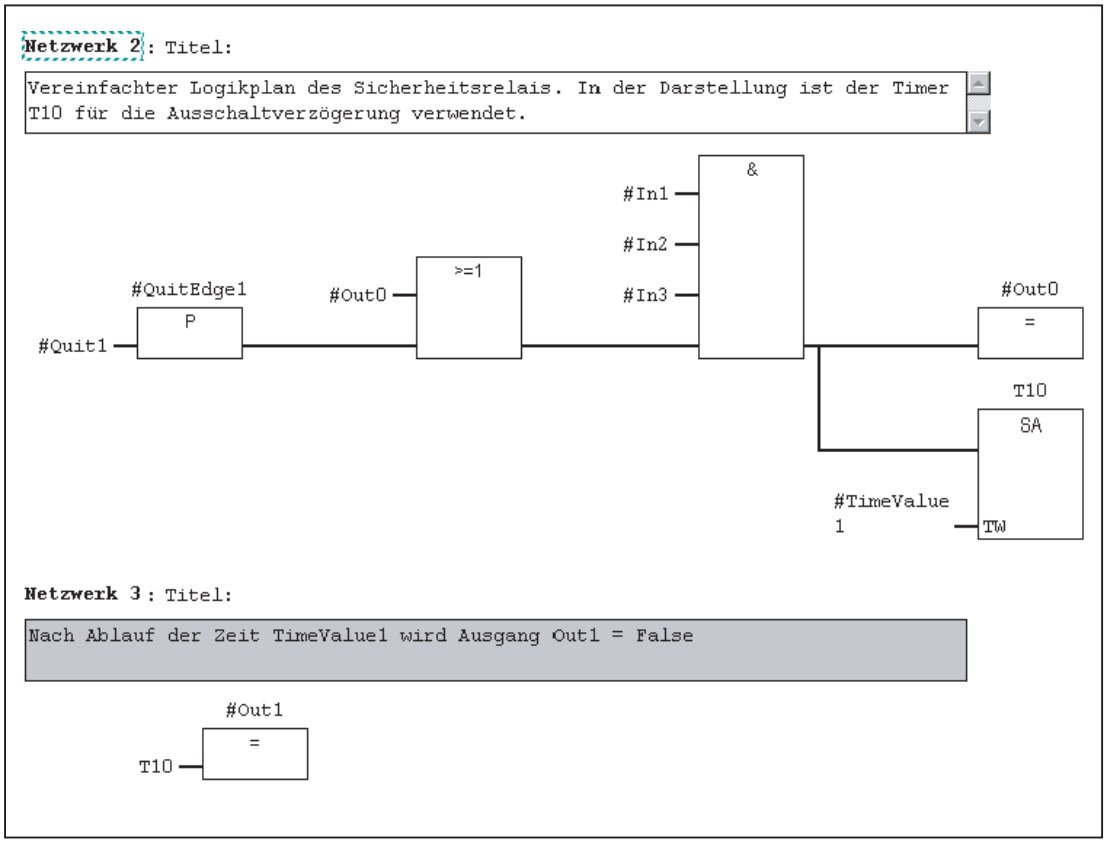
为了使输出在重新接通后回到初始设置（值 = 0），必须如下配置参数“FirstRun”。在控制系统启动后的第 1 次运行中，必须通过一个保持数据（标志位，数据块中的位）将参数 “FirstRun” 切换至 TRUE 值。该数据例如可在 OB100 中预设。FB10 的首次运行完成后，参数 “FirstRun” 会被复位为 FALSE 值。对于每个采用独立实例的调用，必须为“FirstRun”参数使用一个独立的数据。

边界条件

- FB10 支持多实例
- 从启动 SPL 程序开始，必须在 PLC 基本程序 (OB1) 的循环分量中调用 FB10。
- 每次调用 FB10 都必须指定一个独立的、用户区域中的实例 DB。

功能图中的简化接线示意图

下图中只显示一个应答输入 “Quit1” 和一个关断延迟输出 “Out1”。“Quit2” 和其他关断延迟输出的连接以相同的方式实现。功能图中也没有 “FirstRun” 参数。其生效方式请见上文的说明。



功能说明

```
FUNCTION_BLOCK FB10
VAR_INPUT
    In1 :           BOOL;
    In2 :           BOOL;
    In3 :           BOOL;
    Quit1:          BOOL;
    Quit2:          BOOL;
    TimeValue1:     TIME;
    TimeValue2:     TIME;
```

```
    TimeValue3:     TIME;
END_VAR
VAR_OUTPUT
    Out0:           BOOL;
    Out1:           BOOL;
    Out2:           BOOL;
    Out3:           BOOL;
END_VAR
VAR_INOUT
    FirstRun:       BOOL;
END_VAR
```

对形式参数的说明

参数	方式	类型	值域	含义
In1：	E	BOOL	0 (FALSE), 1 (TRUE)	输入 1
In2：	E	BOOL	0 (FALSE), 1 (TRUE)	输入 2
In3：	E	BOOL	0 (FALSE), 1 (TRUE)	输入 3
Quit1：	E	BOOL	0 (FALSE), 1 (TRUE)	应答输入 1
Quit2：	E	BOOL	0 (FALSE), 1 (TRUE)	应答输入 2
TimeValue1：	E	S5TIME		关断延迟的时间值 1
TimeValue2：	E	S5TIME		关断延迟的时间值 2
TimeValue3：	E	S5TIME		关断延迟的时间值 3
Out0：	A	BOOL	0 (FALSE), 1 (TRUE)	不延迟输出
Out1：	A	BOOL	0 (FALSE), 1 (TRUE)	输出延迟 TimeValue1
Out2：	A	BOOL	0 (FALSE), 1 (TRUE)	输出延迟 TimeValue2
Out3：	A	BOOL	0 (FALSE), 1 (TRUE)	输出延迟 TimeValue3
FirstRun：	I/O	BOOL	0 (FALSE), 1 (TRUE)	调用标志位

14.17.9 FB11:制动测试

功能

对于所有通过抱闸防止不受控运行的轴，都应对其制动装置的功能进行测试。主要应用是“悬挂轴”。

机床制造商可在其 PLC 用户程序中选择适宜的时间点定期（标准值为每 8 小时）闭合制动，并由驱动施加等同于轴重力的额外力矩/力。在无故障状态下，制动能够施加必要的制动力矩/制动力，轴几乎不会运动。

在故障状态下，系统将识别出位置实际值偏离参数设置的监控窗口。系统会通过位置控制器避免轴的下沉，制动装置的功能测试会收到消极应答。

文档

有关在 “Safety Integrated” 功能框架内设置 NC 和驱动的参数的详细说明请见：

功能手册 “Safety Integrated”

启动制动测试

制动测试必须在轴的静止状态下开始执行。在制动测试的整个过程中，参数设置的轴的使能信号必须设置为使能（例如：控制器禁用、进给使能）。此外发送至轴/主轴的信号 DB31, ...DBX28.7（PLC 控制轴）在整个测试期间必须通过用户程序设为状态 1。

激活 NC/PLC 接口 DB31, ...DBX28.7（PLC 控制轴）前，必须将轴切换为“中性轴”，例如须将 DB31, ...DBX8.0 - 8.3（NC 轴通道指定）设置为通道 0，以及设置 DB31, ...DBX8.4（修改此字节时的激活信号）。

反馈信息：

- 可在 DB31, ...DBB68 中查询当前状态。
- 模块启动前必须通过信号 DB31, ...DBX63.1（PLC 控制轴）等待 NC。驱动施加力矩/力的方向由 PLC 通过一次“运行”设定，例如通过 FC18。

在制动无法施加必要的力或力矩时，此运行的终点必须能在无危险的情况下到达（无碰撞）。

说明

FC18 的提示

若在用户程序中其他部分为相同的轴调用 FC18，那么这些调用必须相互闭锁。这例如可通过整体调用，并启用一个针对 FC18 参数的共用加密数据接口来实现。第 2 种方案是多次调用 FC18，其中无效 FC18 不由程序运行。必须对多次使用进行加密。

制动测试的步骤

步骤		期望的反馈信息	监控时间值
1	启动制动测试	DBX71.0 = 1	TV_BTactiv
2	闭合制动	Bclosed = 1	TV_Bclose
3	输出运行指令	DBX64.6 或 DBX64.7	TV_FeedCommand
4	测试运行指令输出	DBX62.5 = 1	TV_FXSreached
5	等待抱闸时间	DBX62.5 = 1	TV_FXShold
6	取消制动测试/打开制动	DBX71.0 = 0	TV_BTactiv
7	输出“测试完成”	---	---

边界条件

- FB2 支持多实例
- 每次调用 FB11 都必须指定一个独立的、用户区域中的实例 DB。

功能说明

```
Function_BLOCK FB11
VAR_INPUT
    Start              BOOL;
    Quit:              BOOL;
    Bclosed:           BOOL;
    Axis:              INT;
    TimerNo:           TIMER;
    TV_BTactiv:        S5TIME;
    TV_Bclose:         S5TIM;
    TV_FeedCommand:    S5TIME;
    TV_FXSreached:     S5TIME;
    TV_FXShold:        S5TIME;
END_VAR
VAR_OUTPUT
    CloseBrake:   BOOL;
    MoveAxis:     BOOL;
    Done:         BOOL;
    Error:        BOOL;
```

```
    State:        BYTE;
END_VAR
```

对形式参数的说明

信号	方式	类型	含义
Start	E	BOOL	启动制动测试
Quit	E	BOOL	故障应答
Bclosed	E	BOOL	反馈输入，制动闭合是否激活（单通道 - PLC）
Axis	E	INT	**待测试轴的轴编号**
TimerNo	E	TIMER	用户程序中的计时器
TV_BTactiv	E	S5TIME	监控时间值 → 制动测试生效，检查轴信号 DBX71.0
TV_Bclose	E	S5TIME	监控时间值 → 闭合制动。设置输出 CloseBrake 后检查输入信号 Bclosed
TV_FeedCommand	E	S5TIME	监控时间值 → 输出运行指令。设置 MoveAxis 后检查运行指令
TV_FXSreched	E	S5TIME	监控时间值 → 到达固定停止点
TV_FXShold	E	S5TIME	监控时间值 → 测试制动
CloseBrake	A	BOOL	请求闭合制动
MoveAxis	A	BOOL	请求触发运行
Done	A	BOOL	测试完成
Error	A	BOOL	出错
State	A	BYTE	参见“故障标识”

故障标识

State	含义
0	无故障
1	未满足启动条件，例如轴不处于闭环控制下/制动关闭/轴禁用生效
2	选择制动测试时，无 NC 反馈信息“制动测试生效”
3	输入信号 Bclosed 未得到反馈信息“制动闭合”
4	无运行指令输出（例如轴运动未开始）

State	含义
5	未到达固定停止点 → 轴复位已触发。
6	运行禁止/逼近过慢 → 无法到达固定停止点。监控时间 TV FXSreached 结束。
7	制动完全无法停止（到达终点位置）/ 逼近速度过快
8	制动在抱闸时间内打开
9	选择制动测试时出错
10	内部错误
11	未通过用户程序激活“PLC 控制轴”信号

FB11 调用示例：

```
      UN    M       111.1;   //通过 FB 请求对 Z 轴的制动闭合
      =     A       85.0;    //对 Z 轴的制动控制
      AUF           Axis3";  //Z 轴制动测试
      O     E       73.0;    //触发对 Z 轴的制动测试
      O     M       110.7;   //制动测试进行中
      FP    M       110.0;
      UN    M       111.4;   //出错
      S     M       110.7;   //制动测试进行中
      S     M       110.6;   //下一步骤
      SPBN  m001
      L     DBB     68;
      UW    W#16#F;
      T     MB      115;     //标记通道状态
      L     B#16#10
      T     DBB     8;       //请求中性轴

m001: U     DBX     68.6;    //反馈信息，轴为中性
      U     M       110.6;
      FP    M       110.1;
      R     M       110.6;
      S     M       110.5;   //下一步骤
      S     DBX     28.7;    //请求 PLC 控制的轴

      U     DBX     63.1;    //反馈信息，轴由 PLC 控制
      U     M       110.5;
      FP    M       110.2;
      R     M       110.5;
      S     M       111.0;   //为 FB 启动制动测试
```

```
CALL FB11, DB211 (//制动测试模块
      Start             := M  111.0,   //启动制动测试
      Quit              := E  3.7,     //通过复位键应答故障
      Bclosed           := E  54.0,    //反馈信息，制动闭合
                                       //已激活
      Axis              := 3,          //待测试轴的轴编号
                                       //Z 轴
      TimerNo           := T  110,     //计时器编号
      TV_BTactiv        := S5T#200MS,  //监控时间值：
                                       //制动测试生效 DBX71.0
      TV_Bclose         := S5T#1S,     //监控时间值：
                                       //制动闭合
      TV_FeedCommand    := S5T#1S,     //监控时间值：
                                       //运行指令已输出
      TV_FXSreache      := S5T#1S,     //监控时间值：
                                       //到达固定停止点
      TV_FXShold        := S5T#2S,     //监控时间值：
                                       //制动测试时间
      CloseBrake        := M  111.1,   //请求闭合制动
      MoveAxis          := M  111.2,   //请求触发运行
      Done              := M  111.3,   //测试完成
      Error             := M  111.4,   //出错
      State             := MB 112);    //故障状态

AUF           "Axis3"; //Z 轴制动测试

U     M       111.2;   //运行轴
FP    M       111.5;   //FC18 启动
S     M       111.7;   //FC18 启动

O     M       111.3;   //测试完成
O     M       111.4;   //出错
FP    M       110.3;
R     DBX     28.7;    //请求 PLC 控制的轴

UN    DBX     63.1;    //反馈信息，轴由 PLC 控制
U     M       111.0;   //为 FB 启动制动测试
U     M       110.7;   //制动测试进行中
FP    M       110.4;
R     M       111.0;   //为 FB 启动制动测试
```

```
      R     M       110.7;   //制动测试进行中

//可选开始
      SPBN  m002;
      L     MB      115;     //旧通道状态
      OW    W#16#10;
      T     DBB     8;       //请求通道轴
m002: NOP   0;
//可选结束

      CALL "SpinCtrl" (//运行 Z 轴
           Start   := M     111.2,     //启动运行
           Stop    := FALSE,
           Funct   := B#16#5,          //进给轴模式
           Mode    := B#16#1,          //增量运行
           AxisNo  := 3,               //待运行轴 Z 轴的轴编号
           Pos     := -5.000000e+000, //行程：负 5 毫米
           FRate   := 1.000000e+003,  //进给率：1000 毫米/分钟
           InPos   := M     113.0,     //到达位置
           Error   := M     113.1,     //出错
           State   := MB    114);      //故障状态

      AUF           "Axis3"; //Z 轴制动测试
      U     M       113.0;   //到达位置
      O     M       113.1;   //出错
      FP    M       113.2;
      R     M       111.7;   //FC18 启动
```

14.17.10 FB29:诊断信号记录器和数据触发器

功能

信号记录器

FB29 “Diagnose”用于在 PLC 用户程序上执行各种诊断功能。其中一项诊断是对信号状态和信号变化的记录。在此诊断中，功能编号 1 被指定给参数“Func”。参数“Signal_1”到“Signal_8”的最多 8 个信号会在其中一个发生变化时记录至一个环形缓存。参数“Var1

（作为字节值）”以及“Var2”、“Var3”（作为 INTEGER 值）的当前信息也会保存在环形缓存中。

此外经过的 OB1 循环的数量会作为状态信息保存在环形缓存中。借此可对 OB1 循环框架中的信号和值进行图形分析。

调用规则

OB1 循环中的 FB29 的初次调用：参数 “NewCycle” = 1

同一个 OB1 循环中的 FB29 的所有其他调用：参数 “NewCycle” = 0

环形缓存

由用户自定义的环形缓存必须和源代码中设定的一样采用 ARRAY 结构。ARRAY 单元的数量可为任意值。建议的单元数为 250。通过参数 “ClearBuf” 可清空缓存并将指针 “BufAddr” 置于开始处。FB29 的对应实例 DB 是一个用户区域中的 DB，且须通过参数 “BufDB” 传输至 FB Diagnose。

数据触发器

“数据触发器”功能用于在一个任意的可用存储单元上触发特定值（和位）。在诊断模块中校验参数“TestVal”前，待触发单元会通过位掩码（“AndMask”参数）“取整”。

说明

此功能在基本程序库的源容器中作为源提供，名称为“Diagnose.awl”。该源模块中也定义了实例 DB 和环形缓存 DB。此外还会显示功能调用。此处须调整 DB 编号和调用。

功能说明

```
FUNCTION_BLOCK FB29
VAR_INPUT
Func                :INT;
    Signal_1        :BOOL;
    Signal_2        :BOOL;
    Signal_3        :BOOL;
    Signal_4        :BOOL;
    Signal_5        :BOOL;
    Signal_6        :BOOL;
    Signal_7        :BOOL;
    Signal_8        :BOOL;
```

```
    NewCycle       :BOOL;
    Var1           :BYTE;
    Var2           :INT;
    Var3           :INT;
    BufDB          :INT;
    ClearBuf       :BOOL;
    DataAdr        :POINTER;
    TestVal        :WORD;
    AndMask        :WORD;
END_VAR
VAR_OUTPUT
    TestIsTrue     :BOOL;
END_VAR
VAR_IN_OUT
    BufAddr        :INT;
END_VAR
```

环形缓存结构

```
TITLE =
                                        //FB29 的环形缓存 DB
VERSION: 1.0

STRUCT
   Feld : ARRAY [0 .. 249 ] OF          //此结构可为任意尺寸
   STRUCT

   Cycle: INT;                          //缓存中最近一次存储的 Delta 循环
   Signal_1: BOOL;                      //信号名称与 FB29 相同
   Signal_2: BOOL;
   Signal_3: BOOL;
   Signal_4: BOOL;
   Signal_5: BOOL;
   Signal_6: BOOL;
   Signal_7: BOOL;
   Signal_8: BOOL;
   Var1: BYTE;
   Var2: WORD;
   Var3: WORD;
```

```
   END_STRUCT;
END_STRUCT;
BEGIN
END_DATA_BLOCK
```

对形式参数的说明

信号	方式	类型	取值范围	含义
Func：	E	INT	0, 1, 2	功能 0：关闭 1：信号记录器 2：数据触发器
功能 1 的参数				
Signal_1 ... Signal_8：	E	BOOL	0 (FALSE), 1 (TRUE)	位信号，检查其是否变化
NewCycle：	E	BOOL	0 (FALSE), 1 (TRUE)	参见“功能”
Var1：	E	BYTE		附加值
Var2：	E	INT		附加值
Var3：	E	INT		附加值
BufDB：	E	INT		环形缓存 DB 编号
ClearBuf：	E	BOOL	0 (FALSE), 1 (TRUE)	清空环形缓存 DB，复位指针 BufAddr
BufAddr：	I/O	INT		读取数据的目标区域
功能 2 的参数				
DataAdr：	E	POINTER		对待测试字的指针
TestVal：	E	WORD		校验值
AndMask：	E	WORD		参见“功能”
TestIsTrue：	A	BOOL	0 (FALSE), 1 (TRUE)	校验结果

配置步骤

- 选择诊断模块的功能。
- 为信号记录或数据触发定义适宜的数据。
- 在用户程序中搜索适宜的位置，用于调用 FB Diagnose。
- 为环形缓存创建一个数据块，参见示例。
- 在用户程序中通过参数调用 FB Diagnose。

使用功能 1 时，通过参数“ClearBuf”清空环形缓存是明智的操作。通过功能 1 进行的记录结束后，通过 STEP 7 使用“在数据视图中打开数据块”功能读取环形缓存 DB。这样便可分析环形缓存 DB 的内容。

调用示例

```
FUNCTION FC99: VOID
TITLE =
VERSION: 0.0

BEGIN
NETWORK
TITLE = NETWORK

CALL FB29, DB80(
Func                := 1,
        Signal_1    := M        100.0,
        Signal_2    := M        100.1,
        Signal_3    := M        100.2,
        Signal_4    := M        100.3,
        Signal_5    := M        10.4,
        Signal_6    := M        100.5,
        Signal_7    := M        100.6,
        Signal_8    := M        100.7,
        NewCycle    := TRUE,
        Var1        := MB       100,
        BufDB       := 81,
        ClearBuf    := M        50.0);
END_FUNCTION
```

14.17.11 FC2：GP_HP - 基本程序，循环部分

功能

NC/PLC 接口通过基本程序在循环运行 (OB1) 中执行。为了尽可能保持低的运行时间，只有控制信号/状态信号会被循环传输。辅助指令和 G 指令传输仅在 NC 发出请求时执行。

模块中还会将操作面板 (HMI) 的手轮选择、运行方式及其他操作信号传输至 NC/PLC 接口，这样一来在相应运行方式下可通过 MCP 或 HMI 执行选择。

可通过将 FB1 (DB7) 中的参数 “MMCToIF” 设为 “FALSE” 值来取消 HMI 信号向 NC/PLC 接口的传输。

手轮选择信号

前提条件：FB1，参数 “HWheelMMC” == “TRUE”

HMI 的手轮选择信号会被解码，并在该手轮对应的机床轴接口或几何轴接口中激活。

说明

```
FUNCTION FC2:VOID
//无参数
```

调用示例

从时间角度而言，基本程序必须在执行用户程序**前**运行。因此其必须在 OB1 中最先调用。

下面的示例包含了对 OB1 和基本程序调用（FC2）、MCP 信号传输（FC19）以及故障消息和运行消息采集（FC10）的标准说明。

```
ORGANIZATION_BLOCK OB1
VAR_TEMP
   OB1_EV_CLASS :    BYTE;
   OB1_SCAN_1 :      BYTE;
   OB1_PRIORITY :    BYTE;
   OB1_OB_NUMBR :    BYTE;
   OB1_RESERVED_1 :  BYTE;
   OB1_RESERVED_2 :  BYTE;
   OB1_PREV_CYCLE :  INT;
   OB1_MIN_CYCLE :   INT;
   OB1_MAX_CYCLE :   INT;
   OB1_DATE_TIME :   DATE_AND_TIME;
```

```
END_VAR
BEGIN
CALL FC2：                    // Aufruf Grundprogramm als 1.FC
//此处插入用户程序
CALL FC19(                     //发送至接口的 MCP 信号
BAGNo :=                       B#16#1,    //BAG 编号 1
ChanNo :=                      B#16#1,    //通道编号 1
SpindleIFNo :=                 B#16#4,    //主轴接口编号 = 4
                                          //（对应机床轴的编号）
FeedHold :=                    m22.0,     //进给停止信号
                                          //模态生效
SpindleHold :=  db2.dbx151.0);            //主轴停止，模态生效
                                          //在消息 DB 中
CALL FC10(                                //故障消息和运行消息
                ToUserIF := TRUE,         //从 DB2 发送至接口的信号
                                          //传输
                Quit :=     E6.1);        //应答故障消息
                                          //通过 E6.1
END_ORGANIZATION_BLOCK
```

14.17.12 FC3：GP_PRAL - 基本程序，报警控制部分

功能

在基本程序的报警控制部分中，系统会通过辅助功能和 G 指令处理从 NC 到 PLC 的程序段同步传输。其中**辅助功能**分为一般辅助功能和快速辅助功能。

NC 程序段的快速辅助功能保存在缓存中，并会在接收时向 NC 应答。下一个 OB1 循环开始时，这些辅助功能会传输至用户接口。

对于用户程序，连贯编写的快速辅助功能不会丢失。这通过基本程序中的机制确保。

一般辅助功能仅在经过一个循环周期后才会向 NC 应答。借此用户可根据需要将读取禁止发送至 NC。

G 指令会被系统立即分析，并传输至用户接口。

NC 过程报警

通过 NC 触发中断时（在每个插补周期中都可进行），仅当 FB1 参数 “UserIR”= TRUE 时，基本程序才会设置 OB40 的本地数据中的位（“GP_IRFromNCK”）。其他事件（I/O 引起的过程报警）下此参数不会被置位。借助此信息可在用户程序内跳转至对应的中断程序，用于引入必要的动作。

为了针对机床实现快速的、任务控制的用户程序处理，下列 NC 功能可在中断处理（OB40 程序部分）中用于 PLC 用户程序：

- 选择的**辅助功能**
- 刀具管理（选件）下的**换刀功能**
- **到达位置**，针对定位轴、分度轴和主轴，通过 PLC 激活

上述功能必须通过用户程序在 OB40 中分析，以实现机床上的快速响应。例如，在车床上编写 T 指令时可激活刀塔的切换装置。

过程报警编程、延时、可中断性等内容的更多详细信息请参见相应的 SIMATIC 文档。

辅助功能

一般而言，快速或进行应答的辅助功能不需要指定便可在中断控制下或无中断控制的情形下进行处理。

通过 FB1 中的基本程序参数，可选择仅在中断控制下通过用户程序处理的辅助功能（T、H、DL）。

和之前一样，未通过中断指定的功能通过循环基本程序才可用。此时这些功能的修改信号会保持一个 PLC 循环。

在中断控制下选择了辅助功能组（T、H、DL）时，可通过用户程序为所选的功能进行中断处理。

对于用户程序，系统会基于通道在本地数据 “GP_AuxFunction” 中设置一个位（若设置了 “GP_AuxFunction[1]”，则向通道 1 提供一个辅助功能供其使用）。

在相应的通道 DB 中，修改信号和功能值可供用户使用。该中断控制的功能的修改信号在循环基本程序部分经过至少一个完整的 OB1 循环（最多为近两个 OB1 循环）后才会被复位为零。

换刀

启用刀具管理选件时，针对刀塔的换刀指令以及对主轴进行的换刀能够由中断支持。此时 OB40 中的本地数据位“GP_TM”会置位。这样一来，PLC 用户程序可测试刀具管理 DB（DB72 或 DB73）的换刀功能，并触发换刀。

到达位置

OB40 的本地数据的位结构 “GP_InPosition” 针对机床轴（每个位对应一根轴/主轴，例如 GP_InPosition[5] 对应轴 5）。

若为轴或主轴激活了一个 FC18 功能（主轴控制、定位轴、分度轴），对应的 “GP_InPosition” 位可立即分析出上述 FC 的“InPos” 信号。借此例如可实现对分度轴的立即夹紧。

说明

```
FUNCTION FC3:VOID
//无参数
```

调用示例

从时间角度而言，基本程序必须在其他报警控制用户程序**前**运行。因此其必须在 OB40 中最先调用。

下面的示例包含了对 OB40 和基本程序调用的标准说明。

```
ORGANIZATION_BLOCK OB40
VAR_TEMP
   OB40_EV_CLASS :      BYTE;
   OB40_STRT_INF :      BYTE;
   OB40_PRIORITY :      BYTE;
   OB40_OB_NUMBR :      BYTE;
   OB40_RESERVED_1 :    BYTE;
   OB40_MDL_ID :        BYTE;
   OB40_MDL_ADDR :      INT;
   OB40_POINT_ADDR :    DWORD;
   OB40_DATE_TIME :     DATE_AND_TIME;

//由基本程序占用
GP_IRFromNCK :BOOL;                        //通过 NC 为用户中断
GP_TM :BOOL;                               //刀具管理
GP_InPosition :ARRAY[1..3] OF BOOL;        //基于轴用于定位轴、
                                           //分度轴、主轴
GP_AuxFunction :ARRAY[1..10] OF BOOL;      //基于通道用于辅助功能
GP_FMBlock :ARRAY[1..10] OF BOOL;          //目前未使用
//此处开始可定义其他用户本地数据
END_VAR
BEGIN
```

```
    CALL FC3;
    //此处插入用户程序
END_ORGANIZATION_BLOCK
```

14.17.13 FC5：GP_DIAG 基本程序，诊断报警和模块故障

功能

模块 FC5 “GP_DIAG” 用于采集模块故障和失灵。

通过参数 “PlcStop” 可以触发 PLC 停止。PLC 停止仅在发生事件时触发。连接在 FB1 设置的 PROFIBUS (DP1) 上的 MCP 除外。

说明

```
FUNCTION FC5:VOID
   VAR_INPUT
      PlcStop:BOOL:=TRUE;
   END_VAR
```

调用示例

基本程序应在执行完用户程序后执行。原因是通过 FC5 可以触发 PLC 停止。

示例中包含了对 OB82 和 OB86 以及 FC5 模块调用的标准说明。

```
ORGANIZATION_BLOCK OB82
VAR_TEMP
     OB82_EV_CLASS :BYTE;
     OB82_FLT_ID :BYTE;
     OB82_PRIORITY :BYTE;
     OB82_OB_NUMBR :BYTE;
     OB82_RESERVED_1 :BYTE;
     OB82_IO_FLAG :BYTE;
     OB82_MDL_ADDR :INT;
     OB82_MDL_DEFECT :BOOL;
     OB82_INT_FAULT :BOOL;
     OB82_EXT_FAULT :BOOL;
     OB82_PNT_INFO :BOOL;
     OB82_EXT_VOLTAGE :BOOL;
```

```
    OB82_FLD_CONNCTR :BOOL;
    OB82_NO_CONFIG :BOOL;
    OB82_CONFIG_ERR :BOOL;
    OB82_MDL_TYPE :BYTE;
    OB82_SUB_NDL_ERR :BOOL;
    OB82_COMM_FAULT :BOOL;
    OB82_MDL_STOP :BOOL;
    OB82_WTCH_DOG_FLT :BOOL;
    OB82_INT_PS_FLT :BOOL;
    OB82_PRIM_BATT_FLT :BOOL;
    OB82_BCKUP_BATT_FLT :BOOL;
    OB82_RESERVED_2 :BOOL;
    OB82_RACK_FLT :BOOL;
    OB82_PROC_FLT :BOOL;
    OB82_EPROM_FLT :BOOL;
    OB82_RAM_FLT :BOOL;
    OB82_ADU_FLT :BOOL;
    OB82_FUSE_FLT :BOOL;
    OB82_HW_INTR_FLT :BOOL;
    OB82_RESERVED_3 :BOOL;
    OB82_DATE_TIME :DATE_AND_TIME;
END_VAR
    BEGIN
       CALL FC5
          (PlcStop := FALSE) ;
END_ORGANIZATION_BLOCK

ORGANIZATION_BLOCK OB86
VAR_TEMP
    OB86_EV_CLASS :BYTE;
    OB86_FLT_ID :BYTE;
    OB86_PRIORITY :BYTE;
    OB86_OB_NUMBR :BYTE;
    OB86_RESERVED_1 :BYTE;
    OB86_RESERVED_2 :BYTE;
    OB86_MDL_ADDR :WORD;
    OB86_RACKS_FLTD :ARRAY [0 ..31 ] OF BOOL ;
    OB86_DATE_TIME :DATE_AND_TIME;
END_VAR
    BEGIN
       CALL FC5
```

```
            (PlcStop := TRUE) ;
END_ORGANIZATION_BLOCK
```

14.17.14 FC6：TM_TRANS2 刀具管理和多刀的传输模块

功能

模块 FC6 “TM_TRANS2” 模块在刀具位置修改、状态修改和多刀这些情形下使用。

模块 FC6 与模块 FC8 的功能相同，只是额外集成了**多刀**功能。

模块 FC6 的功能说明中只包含**多刀**功能。

模块 FC8 的功能说明请见 “FC8：TM_TRANS - 刀具管理的传输模块 (页 1152)”。

功能说明

```
FUNCTION FC6:VOID
VAR_INPUT
   Start:           BOOL;
   TaskIdent:       BYTE;
   TaskIdentNo:     BYTE;
   NewToolMag:      INT;
   NewToolLoc:      INT;
   OldToolMag:      INT;
   OldToolLoc:      INT;
   状态:            INT;
   MtoolPlaceNum:   INT;
END_VAR
VAR_OUTPUT
   Ready:           BOOL;
   Error:           INT;
END_VAR
BEGIN
END_FUNCTION
```

对形式参数的说明

信号	方式	类型	取值范围	含义
Start：	E	BOOL	0 (FALSE), 1 (TRUE)	参见模块说明 FC8
TaskIdent：	E	BYTE		参见模块说明 FC8
TaskIdentNo：	E	BYTE		参见模块说明 FC8
NewToolMag：	E	INT		参见模块说明 FC8
NewToolLoc：	E	INT		参见模块说明 FC8
OldToolMag：	E	INT		参见模块说明 FC8
OldToolLoc：	E	INT		参见模块说明 FC8
Status：	E	INT		参见模块说明 FC8
MtoolPlaceNum：	**E**	**INT**		**多刀刀位编号**
Ready：	A	BOOL	0 (FALSE), 1 (TRUE)	参见模块说明 FC8
Error：	A	INT		参见模块说明 FC8

14.17.15 FC7：TM_REV - 刀塔换刀的传输模块

功能

用户在刀塔切换完成后调用 FC7 “TM_REV” 模块。此时须在参数 “ChgdRevNo” 中注明刀塔编号（对应 DB73 中的接口编号）。调用此模块时，系统通过参数 “Ready”= TRUE 反馈后，数据块 DB73.DBW0 中的对应“接口生效”位会由 FC7 复位。

任务执行成功

如果任务执行成功，“Ready” == 1。之后用户必须设置参数 “Start” = 0 或不再调用 FC7。

执行任务时出错

如果执行任务时出错，参数 “Ready” == 0，参数 “Error” == 1。必须在下一个 PLC 循环中重新执行任务。由于执行后续任务时参数 “Start” 不需要上升沿，且任务尚未完成，因此，保持 “Start” = 1。参见以下 “调用示例” 和 “脉冲图” 。

边界条件

- 当通过刀具管理为此传输激活了对应接口 (DB73.DBW0) 时，FC7 模块只允许通过参数 “Start”= 1 启动。
- 不允许终止传输（例如通过通道复位）。
- 参数 “Start” = 1，到参数 “Ready” == 1 或 “Error” == 1

文档

- 有关刀具管理的详细信息请见功能手册之刀具管理分册。
- 刀具管理的 PI 服务请见：
  - FB4:PI_SERV - 请求 PI 服务 (页 1080)
  - FC8：TM_TRANS - 刀具管理的传输模块 (页 1152)
  - FC22: TM_DIR - 刀具管理的方向选择 (页 1203)

手动切换刀塔

通过手动操作旋转刀塔时，不涉及换刀或补偿选择。第一步是将刀架上的刀具换下，并返回其刀塔刀位。为此将通过 FC8（或 FC6）执行异步传输。对应的参数设置有：

```
TaskIdent = 4
TaskIdentNo = 通道编号
NewToolMag = 刀塔的刀库编号
NewToolLoc = 刀具的原始刀位
OldToolMag = Magazinnr.Zwischenspeicher (Spindel) = 9998
OldToolLoc = 主轴的中间存储器编号
Status = 1
```

若现在将刀塔旋转至任意有刀具的位置，那么该刀具必须激活。最简单的方式是在零件程序中进行新的 T 编程。如需在刀塔周期结束时通过 PLC 用户程序实现，那么 PLC 必须为此启动一个 ASUB。必须将当前刀塔位置传输至该 ASUB。这样便可在 ASUB 中确定该刀位上的刀具并选择（参见 Toolbox 上的 Jobshop 示例）。

功能说明

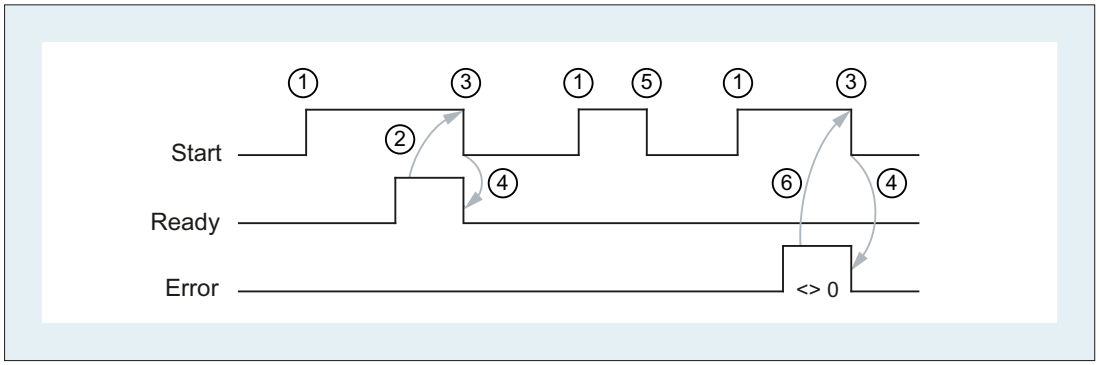
```
FUNCTION FC7:       VOID
//NAME :TM_REV
VAR_INPUT
    Start:          BOOL;
    ChgdRevNo:      BYTE;
END_VAR
VAR_OUTPUT
```

```
    Ready:          BOOL;
    Error:          INT;
END_VAR
BEGIN
END_FUNCTION
```

对形式参数的说明

信号	方式	类型	取值范围	含义
Start：	E	BOOL	0 (FALSE), 1 (TRUE)	1 = 传输开始
ChgdRevNo：	E	BYTE	1, 2, 3, ...	刀塔接口的编号
Ready：	A	BOOL	0 (FALSE), 1 (TRUE)	1 = 传输完成
Error：	A	INT	0, 1 , 2, 3	故障反馈信息 0：无故障 1：无刀塔存在 2：参数“ChgdRevNo”中设定的刀塔编号不允许使用 3：不允许的任务（所选刀塔的“接口生效”信号 =“FALSE”）

脉冲示意图



① 用户：置位请求，Start = 0 → 1

② FB4：PI 服务执行成功，Ready = 1
用户：复位请求，如果 Ready == 1 则 Start = 0

③ 用户：如果 Ready == 1 则复位请求：1 → 0

④ FB4：复位任务确认，Ready = 0

⑤ 用户：如果 Ready == 0 且 Error == 0 则**不允许**复位请求 Start = 1 → 0

⑥ FB4：PI 服务执行时出错，Error = 1
用户：复位请求，如果 Ready == 1 或 Error == 1 则 Start = 0，可能需要执行故障处理

调用示例

```
CALL     // Werkzeugverwaltung:Transfer-Baustein für Revolver
FC7(
    Start :=          m 20.5,         //Start := "1 " => 触发传输
    ChgdRevNo :=      DB61.DBB1,
    Ready :=          m 20.6,
    Error :=          DB61.DBW12
};
u        m 20.6;                      //Ready 询问
r        m 20.5;                      //开始复位
spb      m001;                        //一切正常时则跳过
l        db61.dbw 12;                 //故障信息
ow       w#16#0;                      //分析故障
spn      fehl;                        // <> 0 时，跳转至故障处理
```

```
m001:                                    //其他程序开始
fehl :
r       m 20.5;                          //存在故障时，启动复位
```

14.17.16 FC8：TM_TRANS - 刀具管理的传输模块

功能

用户在刀具位置修改或传输过程状态修改时调用此 FC TM_TRANS 模块。通过参数“TaskIdent”在刀具管理接口上为 FC8 指定传输任务：

- 用于装刀/卸刀
- 用于主轴切换位置
- 作为传输标识用于刀塔切换位置
- 异步传输
- 预留刀位的异步传输

接口编号在参数“TaskIdentNo”中注明。

装刀位置 5 的示例：

参数“TaskIdent”= 1，且“TaskIdentNo”= 5。

此外还为此传输提供**当前**刀具位置和状态信息，参见下文中“Status”参数列出的状态。

说明

FC8 将旧刀具的当前位置提供给 NC。
NC 从而获知位置修改前旧刀具和新刀具所处于的位置。

在无“旧刀具”的传输（例如装刀时）中，将参数“OldToolMag”、“OldToolLoc”设为 0 值。

当通过刀具管理为此传输激活了对应接口（字 0 中的 DB71、DB72、DB73）时，其只允许通过参数“Start”= TRUE 启动。

此任务正确执行完毕后，输出参数“Ready”将显示 TRUE 值。

此时用于须设置参数“Start”= FALSE，并不再调用模块。

若参数“Ready”= FALSE，则须解释参数“Error”中的故障代码，参见 FC8 调用示例和脉冲示意图。

故障代码 = 0 时，必须在下一个 PLC 循环中重复执行此任务（“Start”保持为 TRUE）。这表示传输任务尚未完成。

若用户对“Status”参数的赋值小于 100，那么数据块 DB71/DB72/DB73，字 0 中的对应接口会被取消（进程结束）。接口对应的位会由 FC8 设置为 0。

执行后续任务时，参数“Start”不需要脉冲沿。这表示，在得到“Ready”= TRUE 时便可通过“Start”= TRUE 指定新参数。

异步传输

为了实现刀具位置修改信息从 PLC 到刀具管理的独立传输（例如指令生效时断电或 PLC 独立进行的位置修改），通过 “TaskIdent”= 4 或 5 调用 FC8。该调用不必通过刀具管理激活接口。

参数 “TaskIdent”= 5 时，除了位置修改外还会通过刀具管理进行刀位预留。不过仅当刀具已从实际刀库传输至中间存储器时，才能实现刀位预留。

参数“TaskIdentNo”中必须设置对应的 NC 通道。

在参数 “OldToolMag”、“OldToolLoc” 中设定刀具之前的位置。并在参数 “NewToolMag”、“NewToolLoc” 中设定刀具当前的位置。此时必须设定 “Status” = 1。

设定 “Status” 5 时，指定刀具会停留在 “OldToolMag”、“OldToolLoc” 刀位。此刀位必须为中间存储器（例如主轴）。在参数“NewToolMag”、“NewToolLoc”中须设定实际刀库及刀位，其中刀位位于中间存储器的位置上。需要向刀具管理提供特定刀库刀位的位置信息时，始终须采用此步骤。此步骤在搜索方案中用于校准。

边界条件

- 不允许终止传输（例如通过通道复位）。
- 参数 “Start” = 1，到参数 “Ready” == 1 或 “Error” == 1

文档

- 有关刀具管理的详细信息请见
  功能手册之刀具管理分册
- 刀具管理的 PI 服务
  - FB4:PI_SERV - 请求 PI 服务 (页 1080)
  - FC7：TM_REV - 刀塔换刀的传输模块 (页 1148)
  - FC22: TM_DIR - 刀具管理的方向选择 (页 1203)

功能说明

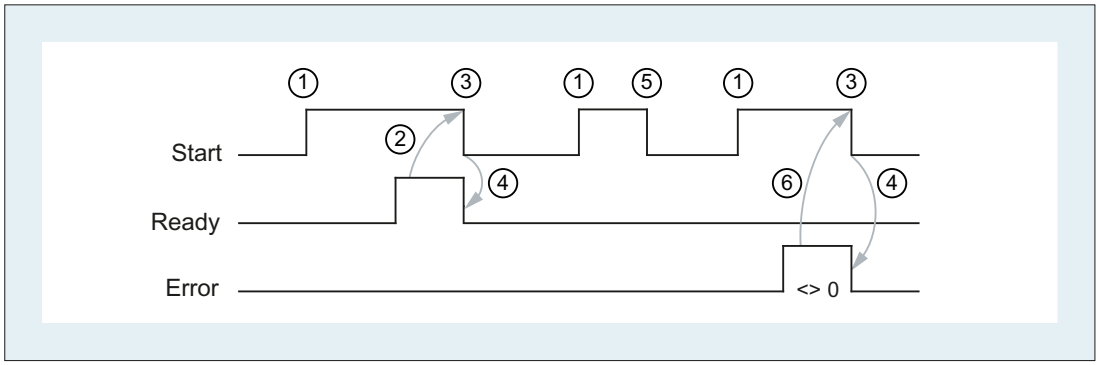
```
FUNCTION FC8:VOID
// NAME :TM_TRANS
VAR_INPUT
    Start :              BOOL;
    TaskIdent:           BYTE;
    TaskIdentNo:         BYTE;
    NewToolMag:          INT;
    NewToolLoc:          INT;
    OldToolMag:          INT;
    OldToolLoc:          INT;
    Status :             INT;
END_VAR
VAR_OUTPUT
    Ready :              BOOL;
    Error :              INT;
END_VAR
BEGIN
END_FUNCTION
```

对形式参数的说明

信号	方式	类型	取值范围	含义
Start：	E	BOOL	0 (FALSE), 1 (TRUE)	1：启动传输
TaskIdent：	E	BYTE	1, 2, 3, 4, 5	接口或任务的标识 1：装刀/卸刀位置 2：主轴切换位置 3：刀塔切换位置 4：异步传输 5：预留刀位的异步传输
TaskIdentNo：	E	BYTE	1, 2, 3 ... 10	对应接口的编号或通道编号

信号	方式	类型	取值范围	含义
NewToolMag:	E	INT	-1, 0, 1, 2 ...	新刀具的当前刀库编号 -1：刀具停留在之前的刀位。 NewToolLoc = 任意值 仅在 TaskIdent = 2 时才可这样设定。
NewToolLoc:	E	INT	0, 1, 2 … 最大刀位号	新刀具的当前刀位编号
OldToolMag:	E	INT	-1, 0 ...	待换刀具的当前刀库编号 -1：刀具停留在之前的刀位。 “OLDToolLoc” = <任意值>。仅在“TaskIdent” = 2 时才可这样设定。
OldToolLoc:	E	INT	最大刀位编号	待换刀具的当前刀位编号
Status:	E	INT	1, 2, 3 ... 7, 103, 104, 105	传输过程的状态信息
Ready:	A	BOOL	0 (FALSE), 1 (TRUE)	1：传输完成
Error:	A	INT	0 ... 65535	故障反馈信息 0：无故障 1：未知“TaskIdent” 2：未知“TaskIdentNo” 3：不允许的任务（所选刀塔的“接口生效”信号 = 0） 其他值：该值相当于此次传输造成的、NC 中刀具管理的故障信息。

脉冲示意图



① 通过上升沿触发功能

② 积极应答：刀具管理传输已执行

③ 收到应答后复位功能启动

④ 通过 FC 切换新号

⑤ 不允许此信号特性。一般须结束任务，因为必须将新的刀具位置提供给 NC 中的刀具管理。

⑥ 消极应答：出现故障，故障代码在输出参数 Error 中

状态

状态	说明
1	**刀具管理任务已完成** 必须将 FC8 模块的参数“NewToolMag”、“NewToolLoc”、“OldToolMag”、“OldToolLoc”设置为相关刀具的实际位置。除了“换刀准备”情形外，设定的刀具目标位置通常为对应的刀具管理接口，另见“对形式参数的说明”。 1. 在**装刀/卸刀/刀具换位**中，刀具会到达请求的目标地址。DB71.DBX(n+0).3 中接口中的位“定位至装刀位置”置位时，为了确保功能完成，不允许使用 Status 1。此时须使用 Status 5 保证功能的正确完成。 2. 在“**换刀准备**”中会进行新刀具的准备。该刀具例如可保存在一个中间存储器（机械手）上。在新刀具保存在中间存储器中后，必要时旧刀具的目标（刀库、刀位）也会被定位至切换位置。而旧刀具仍停留在主轴中。这样便完成了换刀准备。在换刀准备完成应答后可接收“换刀”指令。参数“NewToolMag”、“NewToolLoc”、“OldToolMag”、“OldToolLoc”中的位置对应刀具的当前位置。 3. 进行“**换刀**”（在主轴中或刀塔中）时，接口中对应的刀具会到达请求的目标位置。 这样一来换刀步骤便告完成。
2	**无法提供“新”刀具** 仅在使用“换刀”指令时才允许启用此状态。启用此状态时，PLC 会阻止推荐刀具的换入。NC 中的刀具管理会禁用该推荐（新）刀具。之后通过副刀具执行新的刀具管理命令。参数“NewToolMag”、“NewToolLoc”、“OldToolMag”、“OldToolLoc”中的位置对应刀具的原始位置。
3	**出错** 不允许有刀具位置修改。若在此期间刀具的刀库位置发生变化，则必须借助 FC8 传输模块通过 Status = 105 提前通知。只有这样才能将修改过的位置信息提供给刀具管理。
4	**最好将“旧”刀具置于参数 “OldToolMag”、“OldToolLoc” 中设定的刀库位置。** 此状态只能在换刀准备（切换至主轴）中启用。不可以占用指定给“旧”刀具的刀位。在该状态发送至 NC 中的刀具管理后，系统会生成一个新准备指令（Status_4 = 结束应答）并将其输出至 DB72，该指令会参考要求的旧刀具的刀库位置。 不执行新的刀具搜索，从初始准备指令中接收 “NewToolMag” 和 “NewToolLoc” 的位置。仅在该位置未被占用时才能实现。参数“NewToolMag”和“NewToolLoc”不会被考虑。

状态	说明
5	**进程已结束** “新”刀具位于参数 “NewToolMag”、“NewToolLoc” 中指定的位置上。此时指定的刀具并不实际位于该位置，而是继续停留在相同的刀库刀位中。不过刀库位置已被定位至该位置（例如切换位置）。此状态只适用于刀塔、链式刀库和盘式刀库。借助此状态可通过刀具管理校准刀库的当前位置，从而为之后的指令改善搜索方案。此状态只能在装刀、卸刀、刀具换位和换刀准备中启用。参数 “OldToolMag” 和 “OldToolLoc” 必须设置为中间存储器的数据。 ● **装刀、刀具换位：** 在进行装刀或刀具换位时，NC 中已为刀具预留了刀位。之后机床操作人员必须在目标刀位启用该刀具。注意：重新接通控制系统后，刀位预留会被取消。 ● **换刀准备：** 尚未执行的刀具运动直到换刀后才会继续进行。 ● **定位至装刀位置：** DB71.DBX(n+0).3 中接口中的位“定位至装刀位置”置位时，为了确保功能完成，只允许使用 Status 5（而不是 Status 1）。
6	**刀具管理任务已完成** 此状态的功能与 Status 1 相同，但还会额外进行源刀位的预留。此状态只适用于刀具换位。目标刀位处于一个中间存储刀库中时，命令结束并对刀具的源刀位进行预留。
7	**触发“准备换刀”指令的重复执行** 仅在使用“换刀准备”指令时才允许启用此状态。“新”刀具的位置变化（例如通过“新”刀具的异步指令）时，应启用此状态。FC8 反馈“Ready = 1”后，系统会使用相同的刀具自动重复执行“准备换刀”。为了自动重复执行，系统会重新执行刀具搜索。参数“NewToolMag”、“NewToolLoc”、“OldToolMag”、“OldToolLoc”中的位置必须对应刀具的原始位置。
103	**可启用“新”刀具。** 只有在换刀准备中 PLC 可能会拒绝新刀具（例如 MD20310 $MC_TOOL_MANAGEMENT_MASK，位 4 = 1，从而可由 PLC 重新请求经过修改的参数）的情形下才允许启用此状态。刀具的位置保持不变。需要在 NC 中继续预处理，而不进行不必要停止的情形下，便需要启用此状态。

状态	说明
104	**“新”刀具位于参数 “NewToolMag”、“NewToolLoc” 中指定的位置上。** 仅当刀具仍处于刀库中的同一刀位上时，才允许启用此状态。“旧”刀具位于参数“OldToolMag”、“OldToolLoc”中指定的位置上（中间存储器）。此时新刀具并不实际位于该位置，而是继续停留在相同的刀库刀位中。不过刀库位置已被定位至该位置（例如切换位置）。此状态只可在“准备换刀”中用于刀塔、链式刀库和盘式刀库。借助此状态可通过刀具管理校准刀库的当前位置，从而为之后的指令改善搜索方案。
105	**所有相关刀具均已到达指定的中间存储刀位** 进程尚未结束时的标准情形 刀具处于指定的刀具位置（参数“NewToolMag”、“NewToolLoc”、“OldToolMag”、“OldToolLoc”）。

状态定义

对于应答状态，状态信息 1 至 7 通常会引起命令结束。若将这些状态信息中的一个传输至 FC8，那么 FC8 中设定的接口的“接口生效位”会被复位为“0”（另见 DB71 至 DB73 的接口列表）。这样一来进程将结束。状态信息 103 至 105 下的特性则不同。将这些状态信息中的一个传输至 FC8 时，接口的“接口生效位”仍保持在 1。之后需通过用户程序继续在 PLC 中执行操作（例如继续刀具定位）。这些状态信息通常用于在不引起进程终止的情况下传输一把或两把刀具的位置变更信息。

调用示例

```
CALL FC8(           //刀具管理传输模块
   Start :          m 20.5,         //Start := “1” => 触发传输
   TaskIdent :      DB61.DBB0,
   TaskIdentNo :    DB61.DBB1,
   NewToolMag :     DB61.DBW2,      //新刀具的当前位置
   NewToolLoc :     DB61.DBW4,
   OldToolMag :     DB61.DBW6,      //旧刀具的当前位置
   OldToolLoc :     DB61.DBW8,
   Status :         DB61.DBW10,     //状态
   Ready :          m 20.6,
   Error :          DB61.DBW12);
u m 20.6;                           //Ready 询问
r m 20.5;                           //Start 复位
spb m001;                           //一切正常时则跳过
l DB61.dbw12;                       //故障信息
ow w#16#0;                          //分析故障
```

```
spn fehl;                          //跳转至故障处理

m001:                              //一般分支

fehl :                             //故障处理
r m 20.5 :                         //Start 复位
```

14.17.17 FC9：ASUB - 启动异步子程序

功能

通过模块 FC9 “ASUB” 可触发 NC 中的任意功能。通过 PLC 启动 ASUB 的前提条件是：通过 NC 程序或 FB4（PI 服务 ASUB）进行了选择和设置。此时通道编号和中断编号必须与 FC9 中的参数一致。

以此类方式预处理的 ASUB 可由 PLC 在任意时刻启动。ASUB 会使相关通道中运行的 NC 程序中断。

在一个通道中，同一时间只能启动一个 ASUB。如果在一个 PLC 循环中启动了多个 ASUB，这些 ASUB 在 NC 中按此顺序依次启动。

当 ASUB 运行完成 (“Done” == 1) 或出错 (“Error” == 1) 时，用户必须设置参数 “Start” = 0。

每个 FC9 都需要独占一个全局用户存储区域中的参数 “Ref” ，用于执行任务。该参数为内部使用，且不得由用户修改。参数 “Ref” 在第一个 OB1 循环中启用值 0 初始化，因此每个 FC9 都须采用**绝对**调动。或者可由用户在启动中使用值 0 初始化 “Ref” 参数。这样便可进行有条件调用。如果在激活 FC9 时要求调用，那么在参数 “Done” 出现下降沿切换 (1 → 0) 前，参数 “Start” 始终为 1。

边界条件

- FB4 功能块完成后才可启动 FC9。
- DB10, DBX56.1 == 1（急停）时，无法启动模块 FC9。
- 如果在通道中启动了 ASUB 且通道复位激活，则不可以启动模块 FC9。

功能说明

```
FUNCTION FC9:VOID
//NAME :ASUP
VAR_INPUT
    Start :      BOOL;
```

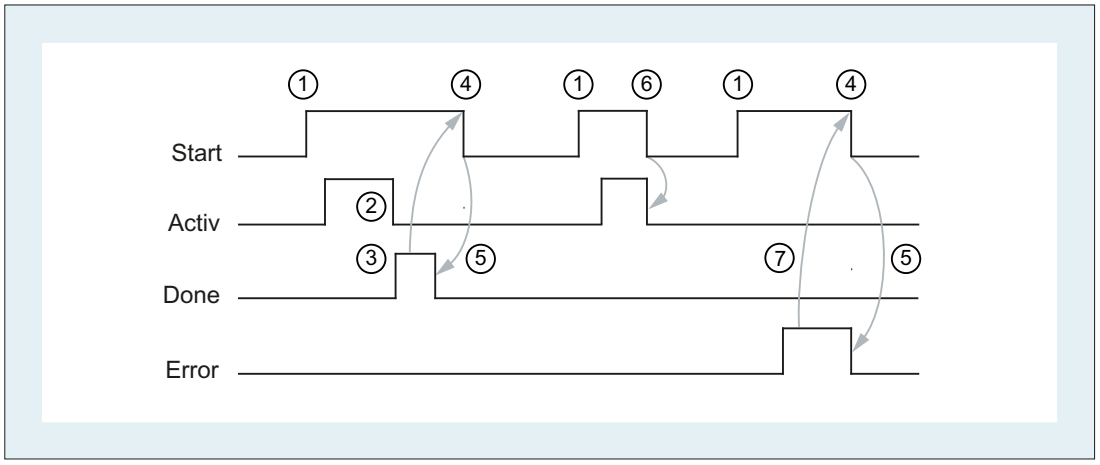
```
    ChanNo :      INT;
    IntNo :       INT;
END_VAR
VAR_OUTPUT
    Activ :       BOOL;
    Done :        BOOL;
    Error :       BOOL;
    StartErr :    BOOL;
END_VAR
VAR_IN_OUT
    Ref :         WORD;
END_VAR
```

对形式参数的说明

下表显示了 ASUB 功能的所有形式参数。

信号	方式	类型	取值范围	含义
Start：	E	BOOL	0 (FALSE), 1 (TRUE)	通过上升沿启动任务
ChanNo：	E	INT	1, 2, 3 ... 10	通道编号
IntNo：	E	INT	1, 2, 3 ... 8	中断编号
Activ：	A	BOOL	0 (FALSE), 1 (TRUE)	1：生效
Done：	A	BOOL	0 (FALSE), 1 (TRUE)	1：ASUB 结束
Error：	A	BOOL	0 (FALSE), 1 (TRUE)	1：中断被取消
StartErr：	A	BOOL	0 (FALSE), 1 (TRUE)	1：中断编号未设定或被删除
Ref：	I/O	WORD	全局变量（MW、DBW ...）	1 Wort，每个 FC9（用于内部应用）

脉冲示意图



(1) 功能触发

(2) ASUB 生效

(3) 积极应答：ASUB 结束

(4) 收到应答后复位功能启动

(5) 通过 FC 切换新号

(6) 不允许。若接收到应答前功能触发被复位，那么输出信号不会更新，对触发功能的运行不会产生影响。

(7) 消极应答：出错

调用示例

```
CALL FC9(                          //启动一个异步子程序
                                   //通道 1 中，中断编号 1
     Start :=      E 45.7,
     ChanNo :=     1,
     IntNo :=      1,
     Activ :=      M 204.0,
     Done :=       M204.1,
     Error :=      M 204.4,
     StartErr :=   M 204.5,
     Ref :=        MW 200);
```

14.17.18 FC10：AL_MSG - 故障消息和运行消息

功能

模块 FC10 “AL_MSG” 用于分析 DB2 中记录的信号，并将这些信号作为“出现”或“消失”的故障消息和运行消息在操作界面上显示。

出现信号（上升沿）在故障消息和运行消息中均会立即被显示。

而消失信号（下降沿）只会在运行消息中立即删除。在故障消息中则需通过参数 “Quit” 删除不再存在的消息。也就是说，即便故障信号已不存在，故障消息仍会在操作界面上继续显示，直至由用户应答。

通过参数“ToUserIF”可触发进给禁止、读取禁止、 NC 启动禁止及进给停止的汇总信号向轴接口、主轴接口和通道接口的传输。这些报警信号独立于报警应答直接从 DB2 的状态信息传输至用户接口。

1. 参数 “ToUserIF” = 0 时，这些信号不会传输至用户接口。在此情形下，用户必须通过 PLC 程序在接口中实现对这些信号的控制。分析 FB1 参数 “ExtendChanAxMsg”，通过配置 NCK 机床数据取消对可用消息区域的限制。
2. 参数 “ToUserIF” = 1 时，上述信号将分别汇总成一个总信号传输至用户接口。这样一来，用户 PLC 程序可只通过 DB2 配合消息或报警输出控制上述信号。用户接口中的相应信息会被改写。

除了第 2 点中说明的方法外，用户也可在调用 FC10 后将禁用或停止状态应用于接口信号，从而在无消息输出的情况下实现对禁用信号和停止信号的控制。

下面的程序示例便是对此方法的说明：

```
CALL FC10(
   ToUserIF :=     TRUE,
   Quit :=         e 6.1);

u m 50.0;                        //通道 1 进给禁止
auf db 21;
s dbx 6.0;                       //设置禁止条件，
                                 //通过 FC AL_MSG 复位，
                                 //当 M 50.0 输出信号 "0" 时
```

FB1 参数“ExtendAlMsg”

激活 FB1 参数 “ExtendAlMsg” 时，一个新的 DB2 结构将生效（参见 “PLC/HMI 接口 (页 969)”）。激活时，提供用于禁止和停止信号的位域，支持 10 通道、31 轴、64 用户区域（用户区域的数量须在 FB1 参数 “MsgUser” 上输入）。在 DB2 中通过对信号的简单设置和复位便可实现各个功能。

数据块 DB2 中的故障和运行消息须由用户设定。

FB1 参数 “ExtendChanAxMsg”

激活该参数时，通道号或轴号生效（与采集报警和消息无关）。用户可以使用所有 DB2 区域。无法将汇总信号传输至用户接口。仅当 FC10 参数 “ToUserIF” 取消时才可分析参数。

在 HMI 上显示

在 DB2 中，一个“1 信号”必须存在多个 OB1 循环，从而确保消息能在 HMI 上显示。

对同时存在的报警和消息有上限限制上限取决于 PLC-CPU。在 PLC317-2DP，可同时存在的消息上限为 60 条。

文档：
参数手册之 NC 变量和接口信号，章节“PLC 用户接口“ > “PLC 报警/消息”

功能说明

```
FUNCTION FC10:       VOID
    // NAME:         AL_MSG
VAR_INPUT
    ToUserIF :       BOOL;
    Quit:            BOOL;
END_VAR
END_FUNCTION
```

对形式参数的说明

信号	方式	类型	取值范围	含义
ToUserIF：	E	BOOL	0 (FALSE), 1 (TRUE)	1：每个循环中信号传输至用户接口
Quit：	E	BOOL	0 (FALSE), 1 (TRUE)	1：应答故障消息

调用示例

```
CALL FC10(                  //故障消息和运行消息
   ToUserIF :=   TRUE,      //DB2 中的信号传输至接口
```

```
   Quit :=       E6.1       //通过输入 E6.1 应答故障消息。
);
```

14.17.19 FC12：AUXFU - 辅助功能的用户调用接口

功能

对于输入参数中传输的通道有新的辅助功能可用时，一般在基本程序中以事件控制的方式调用 FC12 “AUXFU”。借助此 FC，PLC 用户可通过程序指令扩展辅助功能处理，从而在使用辅助功能时避免通道 DB 的循环轮询。通过此机制可实现任务控制的辅助功能处理。该 FC 以经过编译的形式作为空模块在基本程序中提供。此时由基本程序提供“Chan”参数及通道编号。这样一来 PLC 用户便可识别出有新辅助功能可用的通道。通过该通道中辅助功能的修改信号可确定新辅助功能。

功能说明

```
FUNCTION FC12:VOID            //辅助功能的事件控制
VAR_INPUT
    Chan :    BYTE;
END_VAR
BEGIN
    BE;
END_FUNCTION
```

对形式参数的说明

信号	方式	类型	取值范围	含义
Chan：	E	BYTE	0, 1, 2 ... 9	通道索引 = 通道号 -1

示例

```
FUNCTION FC12:VOID          //辅助功能的事件控制
VAR_INPUT
    Chan :      BYTE;       //参数由基本程序提供
END_VAR
VAR_TEMP
    ChanDB:     INT;
```

```
END_VAR

BEGIN
L Chan;                          //通道索引
+ 21;                            //通道 DB 偏移
T ChanDB;                        //保存通道 DB 编号
AUF DB[ChanDB];                  //通道 DB 间接打开
//现在进行查询辅助功能的修改信号等操作
    BE;
END_FUNCTION
```

14.17.20 FC13：BHGDisp - 手持操作设备的显示控制

功能

模块 FC13 “BHGDisp” 用于对手持操作设备（HHU 或 HT 2）的显示特性进行控制。需要在屏幕上显示的信息保存在字符串变量中。在参数 “ChrArray” 中指定指向字符串的指针。因此在创建此数据块时，需要为该字符串指定 32 (HHU) 或 64 (HT 2) 个字符的固定文本。

每个任务向手持操作设备发送 16 个字符。“ChrArray” 中的字符与显示屏中的行的对应关系是固定的。字符串数据“ChrArray”的字符 1 到 16 会传输至行 1，字符 17 到 32 会传输至行 2。对于 HT 2 还会在行 3 显示字符 33 到 48，以及在行 4 显示字符 49 到 64。一个任务会持续多个 OB1 循环。

显示

模块 FC13 会检查 “ChrArray” 字符串长度是够符合运行手持操作设备所需的最小长度要求。若字符串变量中的字符数比需要显示的字符数要少，则会在显示行中填充空字符。通过参数 “Row” = 0 可抑制显示输出（例如需要在一个或多个 PLC 循环中在字符串中记录多个变量，同时不进行显示输出）。字符向行的传输会持续多个 OB1 循环。若“同时”更新多个行的显示（参数 “Row” > 1），那么这些行会以每次一行 16 个字符的方式依次更新。

变量分量

在字符串内可通过带参数 “Convert” = 1 的可选数字转换功能插入变量分量。需要显示的变量通过参数 “Addr” 回参考点。变量格式在参数 “DataType” 中说明。变量的字节数与格式描述相关联。字符串内的右对齐地址通过参数“StringAddr”设定。写入字符的数量可查看参数表。

提高显示分辨率

例如需要以更高的分辨率显示轴实际值时，请注意以下事项：

- 变量和之前一样通过 FB2 或 FB5 读取。Statt Anypointer BYTE 8 als Kriterium zur Ausgabe als 64-Bit Gleitkommazahl wird REAL 2 verwendet (z. B.:P#M100.0 REAL 2).
- 在 HHU/HT 2 上设定 64 位浮点数时，可使用最多 14 位、可自由分配小数点前后位数的输出格式代替固定设定格式。

HHU 的输出信号

HHU 的输出信号中，字节 1 和字符设定基于模块 FC13。这些设定不能通过 PLC 用户程序写入。

相关的 FB1 参数

手持操作设备 HHU

在 OB100 中，必须为手持操作设备的输入和输出数据设置 FB1 参数：

- 参数 “BHGIn” 对应手持操作设备对 PLC 的输入数据（PLC 的接收数据）。
- 参数 “BHGOut” 对应 PLC 向手持操作设备的输出数据（PLC 的发送数据）。

这两个指针都必须设置到数据区域开始处（数据区域开始处在 MPI 连接中通过 SDB210 设置）。

为了运行 HHU，必须将 FB1 参数 “HHU” 设为 2。

手持终端 HT 2

使用 HT 2 时，必须将 FB1 参数 “HH” 设为 5。输入和输出数据的参数须按 “手持操作设备 HHU” 中描述的方式设定。

在参数 “BHGRecGDNo” 和 “BHGRecGBZNo” 中必须输入 HT 2 的接线模块的 DIP-Fix 开关（解码开关）的 S2 上配置的值。

功能说明

```
DATA_BLOCK "strdat"
   STRUCT
    disp :     STRING [32]:= 'character_line1 character_line2';
   END_STRUCT;
BEGIN
END_DATA_BLOCK
```

```
FUNCTION FC13:VOID
  VAR INPUT
    Row :        BYTE;
    ChrArray :   STRING;
    Convert :    BOOL;
    Addr :       POINTER;
    DataType :   BYTE;
    StringAddr : INT;
    Digits :     BYTE;
END VAR
VAR OUTPUT
    Error :      BOOL;
END VAR
```

对形式参数的说明

信号	方式	类型	取值范围	含义
Row：	E	BYTE	0, 1, 2 ... 8, B#16#F	显示行“二进制”分析 0：无显示输出 1：行 1 2：行 2 3：行 1 和行 2 切换 4：行 3 5：行 1 和行 3 切换 8：行 4 B#16#F:所有 4 行自动切换
ChrArray：	E	STRING	“数据块名称”.<变量名称>	显示作为指向字符串的指针的内容 字符串[32]：非 HT 2 字符串[64]：HT 2
Convert：	E	BOOL	0 (FALSE), 1 (TRUE)	激活数字转换
Addr：	E	Pointer		指向待转换变量的指针

信号	方式	类型	取值范围	含义
DataType：	E	BYTE	1, 2, 3 ... 8, B#16#13, B#16#30	变量数据类型 1：BOOL，1 字符 2：BYTE，3 字符 3：CHAR，1 字符 4：WORD，5 字符 5：INT，6 字符 6：DWORD，7 字符 7：DINT，8 字符 8：REAL，9 字符 （7 个数字加上符号和小数点；小数点后位数参见 Digits 参数） B#16#13:字符串，最多 32/64 字符，“Addr” 必须指向 STRING。 B#16#30:REAL64, (12 Zeichen:10 Ziffern plus Vorzeichen und Dezimalpunkt; Nachkommastellen siehe Parameter Digits)
StringAddr：	E	INT	1 ... 32 / 64	变量 “ChrArray” 内的右对齐地址
Digits：	E	BYTE	1, 2, 3 ... 9	小数点后位数 1 ... 4：DataType REAL 1 ... 9：DataType REAL64
Error：	A	BOOL	0 (FALSE), 1 (TRUE)	故障 1：出错

数值范围

数据类型的取值范围	
数据类型	可显示的数字范围
BOOL	0, 1
BYTE	0 ... 255
WORD	0 ... 65535

数据类型的取值范围	
数据类型	可显示的数字范围
INT	- 32768 ... 32767
DWORD	0 至 9999999
DINT	-9999999 ... 9999999
REAL (Digits := 1)	-999999.9 ... 999999.9
REAL (Digits := 2)	-99999.99 ... 99999.99
REAL (Digits := 3)	-9999.999 ... 9999.999
...	...
REAL (Digits := 9)	-0.9999999 ... 0.9999999

调用示例

//以符号表中的名称 strdat 调用 DB，数据单元 disp 作为 String[32]（对于 HT 2：//String[64]），完全通过符号指定

```
CALL FC13(
  Row :=         MB 26,
  ChrArray :=    "strdat".disp,
  Convert :=     M 90.1,
  Addr :=        P#M 20.0,      //待转换数字
  DataType :=    MB 28,         //变量数据类型
  StringAddr :=  MW 30,
  Digits :=      B#16#3,        //小数点后位数 3
  Error :=       M 90.2);
```

14.17.21 FC17：YDelta - 星形/三角形切换

功能

模块 FC17 用于数字主主轴驱动上的星形/三角形切换。切换可双向进行（星形 > 三角形或三角形 > 星形）。

前提条件

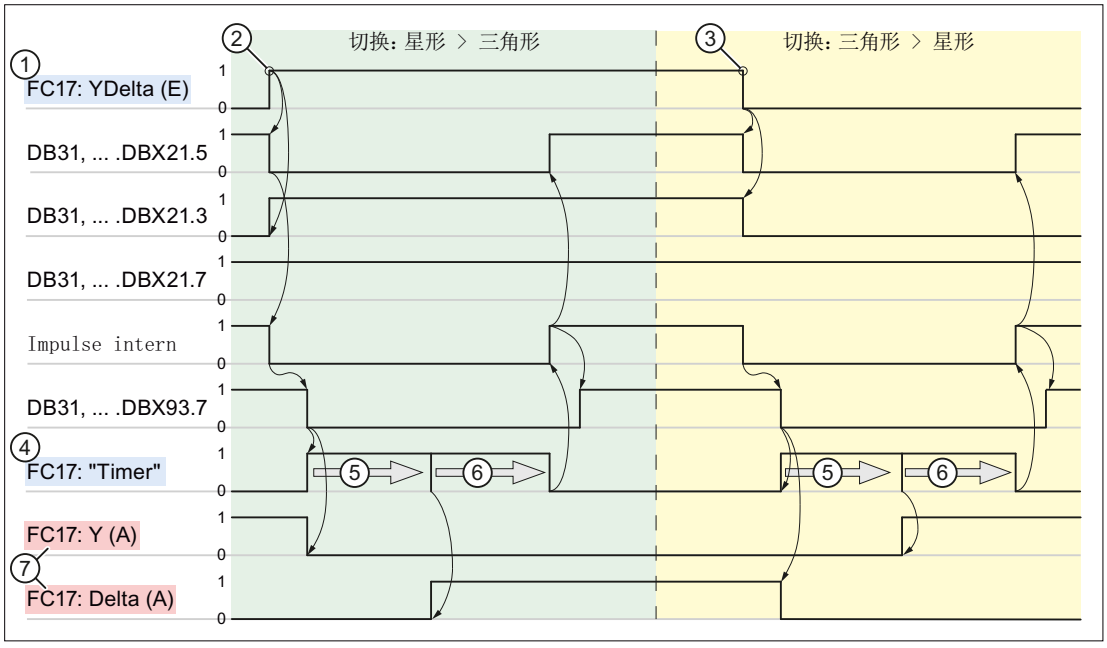
前提条件是两个接触器分开。Die Schütze werden über die an den FC17 Ausgängen:"Y" bzw. "Delta" projektierten Peripherie-Ausgangssignale angesteuert.

内部过程

Nachfolgend ist der interne Ablauf einer Stern-Dreieck-Umschaltung nach dem Umschalten des Ansteuersignals FC17:"YDelta" aufgezeigt.

1. DB31, ...DBX21.5 = 0（复位反馈“电机选择已进行”）
   DB31, ...DBX21.x = 1 (根据接口参数设置 DB31, ...DBX130.0 - 4 (页 76)置位请求“第 2 电机数据组”)
2. DB31, ... DBX93.7 == 0（反馈“脉冲已使能”已复位）⇒
   – 启动 FC17 计时器
   – FC17:“Y” = 0 (复位星形接触器输出)
3. Nach Ablauf des FC17-Timers (FC17:"TimeVal") ⇒
   – FC17:“Y” = 1 (置位三角形接触器输出)
4. Nach erneutem Ablauf des FC17-Timers (FC17:"TimeVal") ⇒
   – 内部再次对脉冲置位
   – DB31, ... DBX93.7 == 1（反馈“脉冲已使能”已置位）
   – DB31, ...DBX21.5 = 0（已置位反馈“电机选择已进行”）

信号图



① FC17 输入：用于星形-三角形切换的信号

② - 切换：星形 > 三角形

③ - 切换：三角形 > 星形

④ FC17 输入：可设置的切换时间 “TimeVal”

⑤ - 至控制输出信号的等待时间：“Y” 或 “Delta”

⑥ - 至脉冲使能的等待时间

⑦ FC17 输出：用于接触器控制的信号

文档

有关电机转速调整的详细说明请见：

- 功能手册之基本功能；主轴 (S1)；章节：“可定义的齿轮级匹配”
- 功能手册之基本功能；速度、设定值/实际值系统、闭环控制 (G2)

说明

驱动参数

星形-三角形切换须参考以下驱动参数：

- p833（数据组切换的配置）
  - 位 0 = **1**（通过 **应用程序**切换接触器）
  - 位 1 = **0**（由**驱动**封锁脉冲）
- p826（电机切换中的电机编号）
- p827（电机切换中的状态字位编号）

前提条件

- 模块 FC17 必须为每根主轴单独调用。
- 在星形/三角形切换期间会关闭驱动脉冲。通过以下设置进行向 PLC 的反馈：
  - DB31, ...DBX93.7 == 0（脉冲已使能）
  - DB31, ...DBX61.7 == 0（电流控制器生效）
  - DB31, ...DBX61.6 == 0（转速控制器生效）
- 处于 M70 或 SPOS 轴运行方式下的主轴不执行星形-三角形切换。
- 使用位置受控的主轴时(DB31, ...DBX61.5 == 1 (位置控制器生效))，在主轴运行期间不能进行星形/三角形切换。
  发生故障时会显示报警 25050“轮廓监控”并且不执行星形/三角形切换。
- 通过 FC17 触发的星形/三角形切换无法由用户延迟（例如通过等待星形/三角形接触器成功切换）。此类延迟必须由用户在 PLC 用户程序中实现。

功能说明

```
VAR_INPUT
    YDelta :        BOOL;
    SpindleIFNo :   INT;
    TimeVal :       S5TIME;
    TimerNo :       INT;
END_VAR
VAR_OUTPUT
    Y :             BOOL;
    Delta :         BOOL;
END_VAR
```

```
VAR_IN_OUT
    Ref :           WORD;
END_VAR
```

对形式参数的说明

信号	方式	类型	取值范围	含义	
YDelta：	E	BOOL	0 (FALSE), 1 (TRUE)	用于星形-三角形切换的输入信号 信号变化会触发切换： • 0 (FALSE)：星形 • 1 (TRUE)：三角形	
SpindleIFNo：	E	INT	1 ...	主轴接口编号（对应机床轴的编号）	
TimeVal：	E	S5time	0, 50 ms ...	切换时间	
				配置	**内部生效**
				0	100
				< 50	50
TimerNo：	E	INT	10 ...	所使用计时器的编号	
Y：	A	BOOL	0 (FALSE), 1 (TRUE)	用于控制星形接触器的外设输出	
Delta：	A	BOOL	0 (FALSE), 1 (TRUE)	用于控制三角形接触器的外设输出	
Ref：	I/O	WORD		附加信息的实例。内部应用	

调用示例

```
CALL FC17(
    YDelta :          E 45.7,          // 星形/三角形切换由输入 45.7 接收
    SpindleIFNo :     4,               //主轴接口编号：4
                                       //（对应机床轴的编号）
    TimeVal :         S5T#150ms,       // 切换时间：150 ms
    TimerNo :         10,              // 计时器：10
    Y :               A 52.3,          // 对星形接触器的控制：输出端 52.3
    Delta :           A 52.4,          // 对三角形接触器的控制：输出端 52.4
    Ref :             MW 50            // 标志位字 50
);
```

14.17.22 FC18: SpinCtrl - 主轴控制

功能

借助模块 FC18“SpinCtrl”可由 PLC 控制主轴和其他轴。此模块支持下列功能：

- 定位主轴
- 主轴旋转
- 主轴往复
- 运行分度轴
- 运行定位轴

每个功能均通过对应触发信号（Start、Stop）的上升沿启动。触发信号必须始终保持在逻辑“1”，直至功能通过 InPos =“1”或 Error =“1”得到积极或消极应答。输出参数会在对应触发信号复位和功能结束时被删除。

为了通过 PLC 控制主轴/轴，必须为 PLC 将其激活。这例如可通过调用由参数“Start”或“Stop”激活的模块来实现。此时模块会请求由 NC 控制主轴/轴。

NC 通过相应的轴专用接口 DB31, ... DBX68.4 - 7 来报告主轴/轴的状态。PLC 获得轴/主轴的控制权后，可通过对应的轴专用接口对用于生效状态的运行指令进行分析。

运行结束后（“InPos”为 TRUE，“Start”切换至零），对轴/主轴的控制通过模块 FC18 切换至中性状态。

或者也可在调用 FC18 前通过 PLC 用户程序为 PLC 请求控制权。

在多次连续调用该功能时，此方案能够实现更优的主轴/轴响应特性，因为省去了 FC 中的切换过程。

通过 PLC 用户程序进行的激活在对应主轴接口的字节 8 中执行。

控制权交还后，可重新通过 NC 程序进行主轴编程。

文档

- 功能手册之基本功能分册；主轴（S1）
- 功能手册之扩展功能分册；定位轴（P2）
- 功能手册 扩展功能部分；分度轴（T1）

⚠ 警告

轴/主轴变更的响应特性

若在 PLC 用户程序中为同根轴/主轴编写了多个模块调用（FC18），则须通过有条件调用对这些功能进行闭锁。对已启动功能（参数 Start 或 Stop = TRUE）的有条件调用必须循环执行，直至输出参数“Activ”或“InPos”的状态从 1 向 0 切换。

说明

调用提示

FC18 必须循环调用，直至信号“InPos”或“Error”（故障信号）输出从 1 向 0 的脉冲沿切换。仅当信号 “InPos”/“Error” 输出 0 值时，才可为该主轴/轴继续执行 “Start” 或 “Stop”。（下一次 “Start” 或 “Stop” 至少须等待一个 PLC 循环。）上述要求同样适用于在轴接口的数据字节 8 中修改指定关系时的情形。

取消

无法通过参数“Start”或“Stop”来终止功能，只能通过轴接口信号进行（例如删除剩余行程）。同样地，轴接口还会反馈轴的状态信号，必要时须对其进行分析（例如准停、运行指令）。

同时性

通过 FC18 模块可同时或分时运行多根轴。上限只能通过 NC 的最大轴数量进行限制。

轴禁止

设置了轴禁止(DB31, ...DBX1.4 == 1)时，通过 FC18 控制的轴无法运行。此时只会生成仿真的实际值。由 NC 控制的轴在轴禁止时的运行也无法进行。

功能

功能 1：定位主轴

参数	含义
Start :	0 → 1：功能启动
Funct :	1：“定位主轴”的功能编号
Mode :	定位模式 1、2、3、4（参见下面的“形式参数说明”）
AxisNo :	机床轴的编号
Pos :	位置
FRate :	FRate ≠ 0:定位速度 FRate = 0:速度符合 MD35300 $MA_SPIND_POSCTRL_VELO
InPos :	1：通过“精准停”到达位置

参数	含义
Error :	1：定位故障
State :	故障代码

功能 2：主轴旋转

参数	含义
Start :	0 → 1：功能启动
Stop :	0 → 1：功能停止
Funct :	2：“主轴旋转”的功能编号
Mode :	Mode = 5:旋转方向 M4 Mode ≠ 5:旋转方向 M3
AxisNo :	机床轴的编号
FRate :	主轴转速
InPos :	1：输出设定转速，另见 DB31, ... DBX83.5（主轴位于设定区域内）
Error :	1：定位故障
State :	故障代码

功能 3：主轴往复

参数	含义
Start :	0 → 1：功能启动
Stop :	0 → 1：功能停止
Funct :	3：“主轴往复”的功能编号
AxisNo :	机床轴的编号
Pos :	设定齿轮档
InPos :	1：输出设定转速，另见 DB31, ... DBX83.5（主轴位于设定区域内）
Error :	1：定位故障
State :	故障代码

设置的往复转速：MD35400 $MA_SPIND_OSCILL_DES_VELO

参数 Pos 的功能取决于 MD35010 $MA_GEAR_STEP_CHANGE_ENABLE = <值>的设置

<值>	Pos	功能
0	0, 1, 2, ... 5	往复
1	0	往复运行和齿轮档切换 M40
	1	往复运行和齿轮档切换 M41
	2	往复运行和齿轮档切换 M42
	3	往复运行和齿轮档切换 M43
	4	往复运行和齿轮档切换 M44
	5	往复运行和齿轮档切换 M45

功能 4：运行分度轴

说明

可使用零件程序中通过 POS[AX]=CIC（值）运行的分度位置对模数转换进行校验。

参数	含义
Start :	0 → 1：功能启动
Funct :	4：“分度轴”的功能编号
Mode :	定位模式 0、1、2、3、4
AxisNo :	机床轴的编号
Pos :	分度位置
FRate :	FRate ≠ 0:定位速度 FRate = 0:速度符合 MD32060 $MA_POS_AX_VEL
InPos :	1：通过“精准停”到达位置
Error :	1：定位故障
State :	故障代码

功能 5, 6, 7, 8:轴定位

参数	含义
Start :	0 → 1：功能启动
Funct :	5, 6, 7, 8：“定位轴”的功能编号
Mode :	定位模式 0、1、2、3、4

参数	含义
AxisNo :	机床轴的编号
Pos :	位置
FRate :	FRate ≠ 0:定位速度 FRate = 0:速度符合 MD32060 $MA_POS_AX_VELO
InPos :	1：通过“精准停”到达位置
Error :	1：定位故障
State :	故障代码

功能 9：主轴旋转与自动齿轮档选择：

参数	含义
Start :	0 → 1：功能启动
Stop :	0 → 1：功能停止
Funct :	9：“主轴旋转与齿轮档选择”的功能编号
Mode :	Mode = 5:旋转方向 M4 Mode ≠ 5:旋转方向 M3
AxisNo :	机床轴的编号
FRate :	主轴转速
InPos :	1：设定转速已输出
Error :	1：定位故障
State :	故障代码

功能 10、11：以恒定切削速度旋转主轴

功能“恒定切削速度” (G96) 应在 NC 中生效。

参数	含义
Start :	0 → 1：功能启动
Stop :	0 → 1：功能停止
Funct :	10：“恒定切削速度（m/min）”的功能编号 11：“恒定切削速度（feet/min）”的功能编号
Mode :	Mode = 5:旋转方向 M4 Mode ≠ 5:旋转方向 M3
AxisNo :	机床轴的编号

参数	含义
FRate :	切削速度
InPos :	1：设定转速已输出
Error :	1：定位故障
State :	故障代码

功能说明

```
FUNCTION FC18:VOID          //SpinCtrl
VAR_INPUT
    Start:                  BOOL;
    Stop :                  BOOL;
    Funct :                 BYTE;
    Mode :                  BYTE;
    AxisNo :                INT;
    Pos :                   REAL;
    FRate :                 REAL;
END_VAR
VAR_OUTPUT
    InPos :                 BOOL;
    Error:                  BOOL;
    State :                 BYTE;
END_VAR
```

对形式参数的说明

信号	方式	类型	取值范围	含义
Start:	E	BOOL	0 (FALSE), 1 (TRUE)	0 → 1：功能启动
Stop:	E	BOOL	0 (FALSE), 1 (TRUE)	0 → 1：功能停止

信号	方式	类型	取值范围	含义
Funct：	E	BYTE	1, 2, 3, ... 11	1：定位主轴 2：主轴旋转 3：主轴往复 4：分度轴 5：定位轴（公制） 6：定位轴（英制） 7：定位轴（公制，带手轮叠加） 8：定位轴（英制，带手轮叠加） 9：主轴旋转与自动齿轮档选择 10：通过恒定切削速度（m/min）旋转主轴 11：通过恒定切削速度（feet/min）旋转主轴
Mode：	E	BYTE	0, 1, 2, ... 5	0：定位到绝对位置 1：增量定位 2：定位到最短行程 3：沿正方向绝对定位 4：沿负方向绝对定位 5：旋转方向同 M4
AxisNo：	E	INT	1, 2, 3, ... 31	待运行的轴/主轴的编号
Pos：	E	REAL	∓ 0.1469368 E -38 至 ∓ 0.1701412 E +39	回转轴：度 分度轴：分度位置 线性轴：毫米或英寸
FRate：	E	REAL	∓ 0.1469368 E -38 至 ∓ 0.1701412 E +39	回转轴和主轴：[转/分钟] 线性轴：[米/分钟]或[英尺/分钟]
InPos：	A	BOOL	0 (FALSE), 1 (TRUE)	1：位置已到达或功能已执行
Error：	A	BOOL	0 (FALSE), 1 (TRUE)	1：故障
State：	A	BYTE	0, 1, 2, ... 255	错误标识

故障标识

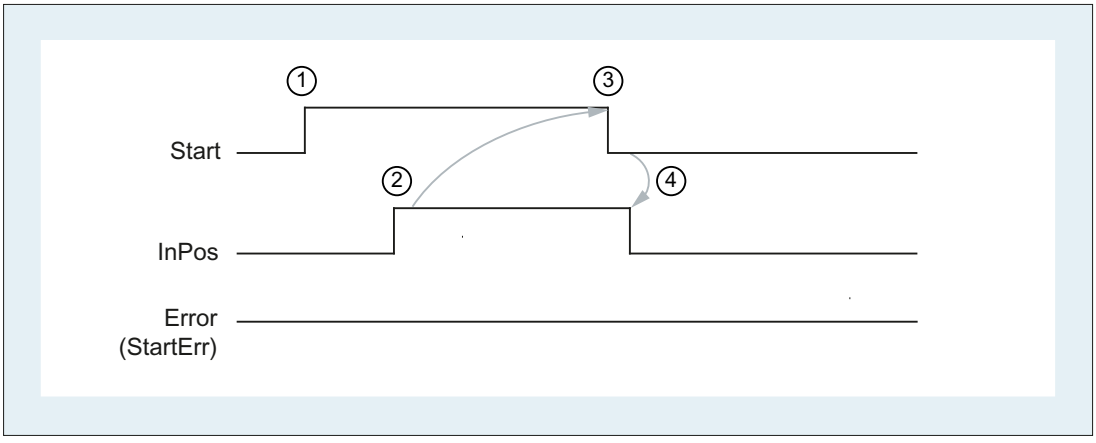
当出现以下设置时，表示存在故障：参数 “Error” == 1 (TRUE)

故障原因在参数 “State” 中显示

State	含义
PLC 侧的故障原因	
1	已同时激活多个轴/主轴功能
20	启动了一个功能，但未到达位置
30	轴/主轴的控制权在运动结束前即已移交给 NC
40	通过 NC 程序对轴进行编程，NC 内部故障
50	固定分配的 PLC 轴：运行（JOG）或回参考点
60	固定分配的 PLC 轴：通道状态不允许开始
NC 侧的故障原因	
100	对轴/主轴编程时的位置错误（对应报警 16830）
101	编写的转速过高
102	恒定切削速度的取值范围错误（对应报警 14840）
104	副主轴:不允许的编程（对应报警 22030）
105	无测量系统（对应报警 16770）
106	轴定位尚在生效（对应报警 22052）
107	未找到参考标记（对应报警 22051）
108	从转速闭环控制到位置闭环控制没有过渡（对应报警 22050）
109	未找到参考标记（对应报警 22051）
110	速度/转速为负
111	设定转速 == 零
112	无效的齿轮档
115	未到达编程的位置
117	在 NC 中 G96/G961 **未生效**
118	在 NC 中 G96/G961 **仍有效**
120	该轴不是分度轴（对应报警 20072）
121	分度位置错误（对应报警 17510）
125	DC（最短行程）无法启用（对应报警 16800）
126	无法启用负绝对值（对应报警 16820）

State	含义
127	无法启用正绝对值（对应报警 16810）
128	无用于直径编程的平面轴（对应报警 16510）
130	软件限位开关正（对应报警 20070）
131	软件限位开关负（对应报警 20070）
132	工作区域限制正（对应报警 20071）
133	工作区域限制负（对应报警 20071）
134	框架不可用于分度轴
135	配备“端面齿”的分度轴生效（对应报警 17501）
136	配备“端面齿”的分度轴生效，且轴未回参考点（对应报警 17503）
137	不可为经过转换的主轴/轴启动主轴模式（对应报警 22290）
138	轴：超出特定坐标系中的正向工作区域限制（对应报警 20082）
139	轴：超出特定坐标系中的负向工作区域限制（对应报警 20082）
系统故障	
200	对应报警 450007
报警编号：**文档** 诊断手册	

信号图：通常情况



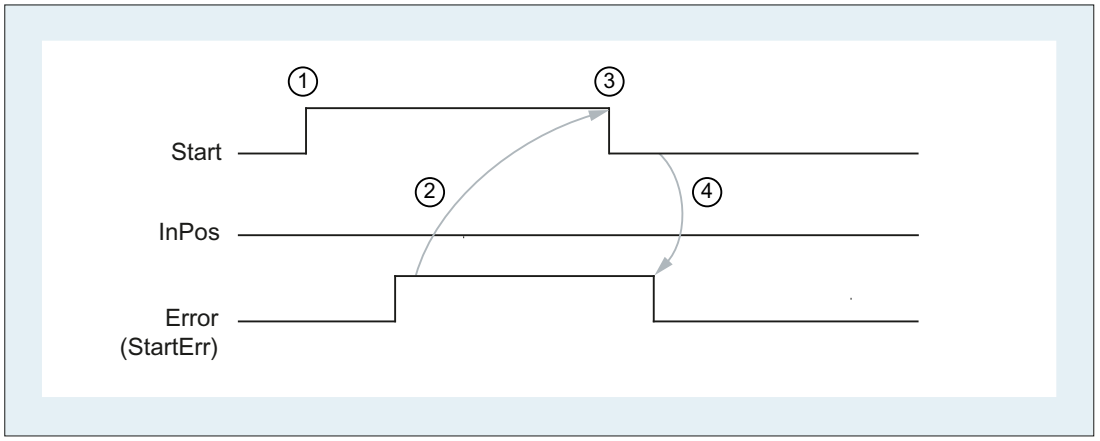
① PLC 用户程序：通过上升沿启动功能：0 → 1

② NC：积极应答，功能已执行/位置已到达

③ PLC 用户程序：识别到积极应答后复位

④ FC18:积极应答的复位

信号图：故障情况



① PLC 用户程序：通过上升沿启动功能：0 → 1

② NC：消极应答，出现故障

③ PLC 用户程序：识别到消极应答后复位

④ FC18:消极应答的复位

调用示例

示例 1：主轴定位：

```
//积极应答使 Start 复位：
U M112.0;                    //InPos
R M 100.0;                   //Start
//消极应答，故障分析（State：MB114）后通过 T12 复位 Start
U M113.0;                    //Error
U E 6.4;                     //按键 T12
R M 100.0;                   //Start
//通过 T13 启动
U E 6.3;                     //按键 T13
UN M 112.0;                  //仅 InPos 或 Error = 0 时才进行新的 Start
UN M 113.0;
S M 100.0;

CALL FC18(
    Start: = M100.0,
    Stop: = FALSE,
    Funct: = B#16#1,         //主轴定位
    Mode: = B#16#2,          //最短行程
    AxisNo: = 5,
    Pos: = MD104,
```

```
    FRate: = MD108,
    InPos: = M112.0,
    Error: = M113.0,
    State: = MB114);
```

示例 2：启动主轴旋转：

```
CALL FC18(
    Start: = M100.0,
    Stop: = FALSE,
    Funct :=  B#16#2,          //主轴旋转
    Mode: = B#16#5,            //旋转方向同 M4
    AxisNo: = 5,
    Pos: = 0.0,
    FRate: = MD108,
    InPos: = M112.0,
    Error: = M113.0,
    State: = MB114);
```

示例 3：启动主轴往复

```
CALL FC18(
    Start: = M100.0,
    Stop: = FALSE,
    Funct: = B#16#3,           //主轴往复
    Mode: = B#16#0,
    AxisNo: = 5,
    Pos: = 0.0,
    FRate: = MD108,
    InPos: = M112.0,
    Error: = M113.0,
    State: = MB114);
```

示例 4：运行分度轴

```
CALL FC18(
    Start: = M100.0,
    Stop: = FALSE,             //未使用
    Funct: = B#16#4,           //运行分度轴
    Mode: = B#16#0,            //绝对定位
    AxisNo: = 4,
    Pos: = MD104,              //以 REAL 设定：1.0;2.0;..
    FRate: = MD108,
    InPos: = M112.0,
    Error: = M113.0,
    State: = MB114);
```

示例 5：轴定位

```
CALL FC18(
    Start: = M100.0,
    Stop: = FALSE,          //未使用
    Funct: = B#16#5,        //轴定位
    Mode: = B#16#1,         //增量定位
    AxisNo: = 6,
    Pos: = MD104,
    FRate: = MD108,
    InPos: = M112.0,
    Error: = M113.0,
    State: = MB114);
```

14.17.23 FC19:MCP_IFM - 将 MCP 信号传送到接口上

功能

模块 FC19“MCP_IFM”（M 型，例如 MCP 483）用于将数据从机床控制面板传输至 NC/PLC 接口：

- 运行方式
- 轴选择
- WCS/MCS 切换
- 运行键
- 倍率
- 钥匙开关

根据生效的运行方式或选择的坐标系，以下定义适用于**进给倍率、轴运行键**和 **INC 键**。

- **进给倍率**
  - 进给倍率会传输至所选通道的接口和轴的接口。
  - 当 HMI 信号“快进的进给倍率生效”置位时，除了接口字节“快进补偿”（DBB5）外，进给倍率信号还会传输至 NC 通道（特例：开关位置“零”）。此外“快进补偿生效”也会随此 HMI 信号置位。
- **机床功能 INC 键和轴运行键**
  - 选择了 MCS 时，这些信号会被传输至所选机床轴的接口。
  - 选择了 WCS 时，这些信号会被传输至参数设置的通道的几何轴接口。
  - 在 MCS 和 WCS 间切换时，之前选择的轴一般会被取消。

对机床控制面板的 LED 的控制基于相应选择的反馈信息。

进给和主轴启动/停止不会传输至接口，而是作为信号“FeedHold”或“SpindleHold”模态输出。用户可将这些信号与其他用于引起进给停止或主轴停止的信号互联（这例如可通过 FC10:AL_MSG 的相应输入信号实现）。此外其对应的 LED 也会一同被控制。

机床控制面板失灵时，其发出的信号会被设为零，输出信号“FeedHold”和“SpindleHold”同样如此。

在一个 PLC 循环中允许多次调用 FC19 以及 FC24、FC25、FC26。此时 LED 显示基于循环中的首次调用。此外，在首次调用中会执行参数设置的模块的所有动作。之后的调用中只会处理通道和 BAG 接口。对于几何轴，只需在循环中首次调用模块时设定方向。

单程序段模式的选择/取消只通过循环中首个调用进行控制。

当参数“BAGNo”提升 B#16#10 时，可处理第二块机床控制面板。进行参数设置时，BAG 编号包含在下部半位元组中。

“BAGNo” = 0 或 B#16#10 ⇒ 不处理 BAG 信号。

“ChanNo” = 0 ⇒ 表示不处理通道信号。

INC 选择会被传输至 BAG 接口。此设定由此模块通过 DB10.DBX57.0（BAG 区域内的 INC 输入生效）在启动后单次激活。

此外还可通过模块 FC19 并行处理两个机床控制面板。此时必须将 OB1 循环中对 MCP 2 的模块调用设置到对 MCP 1 的调用后。机床控制面板模块能够在特定界限内支持 2 个 MCP。在为 2 个 MCP 指定了相同的轴时，不支持轴选择功能中的相互闭锁 。

直角手动运行

通过机床控制面板的方向键 R11（WCS/MCS 的左侧）可以实现“在刀具定向中手动运行”功能。为此，必须通过 DB7 中的 FB1 输入参数 “MCP_IF_TCS” 激活。

“MCP_IF_TCS” = TRUE 时，通过按键 R11 可以切换至“在刀具定向中手动运行”。此时，固定通过 FC19 选择 Z 键 (R3)。方向键作用于各个通道的第 3 几何轴。

灵活的轴配置

选择轴时的指定方式以及机床轴编号的方向键都能体现出此模块的灵活性。

MCP 模块能够更好地支持 2 块机床控制面板的同时运行，特别针对 2 通道、2 BAG 的应用情形。在相应 MCP 的轴表格中，轴编号也可以 MCP 模块的 BAG 编号设定。

借助 DB10 中的轴编号表格便可实现此灵活性。

对于**第一**块机床控制面板，表格从字节 8 开始（符号名称：MCP1AxisTbl[1..22]）；对于**第二**块机床控制面板则从字节 32 开始（符号名称：MCP2AxisTbl[1..22]）。此处必须以字节方式输入机床轴编号。

可在轴表格中输入 0。系统不会检查是否有不允许的轴编号，输入错误时可能会引起 PLC 停止。

对于 **FC19**，也可对**可选择的最大轴数量**进行限制。对于 MCP 1，此上限在 DB10.DBW30（符号名称：MCP1MaxAxis）中设置，对于 MCP 2 则在 DB10.DBW54（符号名称：MCP2MaxAxis）中设置。

其预设值为 0，此时配置的最大轴数量将作为上限生效。轴编号和限制也可启用动态调整。之后必须在 FC19 中重新执行轴选择。轴通过相应的方向键运行期间不允许切换轴编号。缺省设置为兼容模式，其中两个 MCP 的轴编号为 **1 至 9**，可选择的轴数上限为配置的轴数。

示例

需要使用 FC19 通过附加应用控制超过 9 根轴。此时须执行下列步骤：

- 在 MCP 上预留未占用按键
- 将这些按键作为触发器进行分析
- 分析触发器输出的上升沿和下降沿
- 上升沿时在轴表格（DB10）中写入一组轴编号，并通过该按键接通 LED
- 下降沿时在轴表格（DB10）中写入另一组轴编号，并通过该按键关闭 LED

功能说明

```
FUNCTION FC19:VOID                  //符号名称：MCP_IFM

   VAR_INPUT
      BAGNo :           BYTE;
      ChanNo :          BYTE;
      SpindleIFNo :     BYTE;
   END_VAR

   VAR_OUTPUT
      FeedHold :        BOOL;
      SpindleHold :     BOOL;
   END_VAR
```

```
BEGIN
END_FUNCTION
```

对形式参数的说明

信号	方式	类型	取值范围	含义
BAGNo：	E	BYTE	B#16#00 - B#16#0A	**第 1 个 MCP**:运行方式信号需要传输至的 BAG 的编号。
			B#16#10 - B#16#1A	**第 2 个 MCP**:运行方式信号需要传输至的 BAG 的编号。
ChanNo：	E	BYTE	B#16#00 - B#16#0A	通道信号需要传输至的通道的编号
SpindleIFNo：	E	BYTE	0 - 31 (B#16#1F)	主轴数据需要传输至的进给轴/主轴的编号（对应机床轴的编号）
FeedHold：	A	BOOL	0 (FALSE), 1 (TRUE)	MCP 进给停止，模态生效
SpindleHold：	A	BOOL	0 (FALSE), 1 (TRUE)	MCP 主轴停止，模态生效

传输至用户接口的 MCP 选择信号

表格 14-3　钥匙开关

来源： **MCP 开关**	**目标：** **接口 DB**
位置 0	DB10.DBX56.4
位置 1	DB10.DBX56.5
位置 2	DB10.DBX56.6
位置 3	DB10.DBX56.7

表格 14-4　运行方式和机床功能

来源： MCP 按键	目标： 接口 DB（参数 BAGNo） BAG 1 的显示
AUTOMATIC	DB11.DBX0.0
MDI	DB11.DBX0.1
JOG	DB11.DBX0.2
REPOS	DB11.DBX1.1
REF	DB11.DBX1.2
TEACH IN	DB11.DBX1.0
INC 1 ... 10 000, INC Var.	DB11.DBX2.0 - 2.5

表格 14-5　方向键快进叠加

来源： MCP 按键	目标： 接口 DB（参数 ChanNo）
方向键 +	DB21, ...DBX12.7
方向键 -	DB21, ...DBX12.6
快进叠加	DB21, ...DBX12.5
方向键 +	DB21, ...DBX16.7
方向键 -	DB21, ...DBX16.6
快进叠加	DB21, ...DBX16.5
方向键 +	DB21, ...DBX20.7
方向键 -	DB21, ...DBX20.6
快进叠加	DB21, ...DBX20.5

来源： MCP 按键	目标： 接口 DB（所有轴 DB）
方向键 +	DB31, ...DBX4.7
方向键 -	DB31, ...DBX4.6
快进叠加	DB31, ...DBX4.5

传输取决于选择的轴。未选择的轴所对应的接口位会被删除。

表格 14-6　倍率

来源： MCP 开关	目标： 接口 DB（参数 ChanNo）
进给倍率	DB21, ...DBB4

来源： MCP 开关	目标： 接口 DB（所有轴 DB）
进给倍率	DB31, ...DBB0（选择的刀具编号）MCP 1 的进给倍率应用于所有轴。
主轴倍率	DB31, ...DBB19（参数：SpindleIFNo）

表格 14-7　通道信号

来源： MCP 按键	目标： 接口 DB（参数 ChanNo）
NC 启动	DB21, ...DBX7.1
NC 停止	DB21, ...DBX7.3
复位	DB21, ...DBX7.7
单程序段	DB21, ...DBX0.4

表格 14-8　进给率，主轴信号

来源： MCP 按键	目标： FC 的输出参数
进给停止 进给使能	参数：“FeedHold”存储互联，对 LED 进行控制
主轴停止 主轴使能	参数：“SpindleHold”存储互联，对 LED 进行控制

表格 14-9　直角手动运行

来源： **MCP 按键**	**目标：** **接口 DB（参数 ChanNo）**
方向键 R11	DB21, ...DBB392
方向键 +	DB21, ...DBX20.7
方向键 -	DB21, ...DBX20.6

用户接口信号的反馈信息，用于控制 LED 显示

表格 14-10　运行方式和机床功能

目标： **MCP - LED**	**来源：** **接口 DB（参数 BAGNo）** **BAG 1 的显示**
AUTOMATIC	DB11.DBX6.0
MDI	DB11.DBX6.1
JOG	DB11.DBX6.2
REPOS	DB11.DBX7.1
REF	DB11.DBX7.2
TEACH IN	DB11.DBX7.0

目标： **MCP - LED**	**来源：** **接口 DB（参数 BAGNo）** **BAG 1 的显示**
INC 1 ... 10 000, INC Var.	DB11.DBX8.0 - 8.5

表格 14-11 通道信号

目标： MCP - LED	来源： 接口 DB（参数 ChanNo）
NC 启动	DB21, ...DBX35.0
NC 停止	DB21, ...DBX35.2 或 DB21, ...DBX35.3
单程序段	DB21, ...DBX0.4

说明

方向键的 LED 通过方向键操作进行控制。

轴选择和 WCS/MCS LED 通过对应按键的操作进行控制。

调用示例

```
CALL FC19(      //将 M 型机床控制面板的信号发送至接口
   BAGNo :=          B#16#1,           //BAG 编号 1
   ChanNo :=         B#16#1,           //通道编号 1
   SpindleIFNo :=    B#16#4,           //主轴接口编号 = 4
   FeedHold :=       m22.0,            //进给停止信号，模态生效
   SpindleHold :=    db2.dbx151.0);    //主轴停止模态生效
                                       //消息 DB
```

基于此参数设置，信号会传输至 BAG 1、通道 1 和所有轴。此外主轴倍率会传输至进给轴/主轴接口 4。进给停止信号会传输至标志位 22.0，主轴停止信号传输至数据块 DB2、数据位 151.0。

轴选择配置

无需改写或重写 FC19 便可将轴选择按键灵活地指定给相应轴或主轴。只需将轴编号输入 DB10.DBB8 及之后的轴表格中。只需将轴编号输入 DB10.DBB8 及之后的轴表格中。

示例

主轴定义为第 4 机床轴，需通过轴按键 9 选择。

解决方案：

在 DB10 字节 (8 +(9-1)) 中对应第 4 轴输入 4。

```
CALL FC19(             //发送至接口的信号
   BAGNo :=            B#16#1,      //BAG 编号 1
   ChanNo :=           B#16#1,      //通道编号 1
```

```
SpindleIFNo :=      B#16#4,      //主轴接口编号 = 4

FeedHold :=         m30.0,       //进给停止信号，模态生效
SpindleHold :=      m30.1);      //主轴停止，模态生效
```

14.17.24 FC21:传输 - NC/PLC 数据交换

14.17.24.1 功能

模块 FC21 用于 PLC 与 NCK 之间的数据交换。调用 FC21 时会立即传输数据，而不是在下一个 PLC 基本程序循环开始时才进行。

通过调用带参数 "Enable" = 1 的模块 FC21 激活数据传输：

功能

此模块提供以下功能：

- 同步动作信号：PLC → NC 通道
- 同步动作信号：NC 通道 → PLC
- PLC-NC 快速数据交换（在 NC 中读取功能）
- PLC-NC 快速数据交换（在 NC 中写入功能）
- 更新发送至 NC 通道的信号
- 更新发送至轴的信号（用户接口的数据字节 2）
- 更新发送至轴的信号（用户接口的数据字节 4）

14.17.24.2 功能说明

功能说明

```
VAR_INPUT
           Enable :BOOL;
           Funct :BYTE;
           S7Var :ANY;
           IVar1 :INT;
           IVar2 :INT;
END_VAR

VAR_OUTPUT
```

```
            Error :BOOL;
            ErrCode :INT;
END_VAR
```

14.17.24.3 对形式参数的说明

对形式参数的说明

信号	方式	类型	取值范围	含义	
Enable：	E	BOOL	0 (FALSE), 1 (TRUE)	1：	传输数据
Funct：	E	BYTE	1, 2, 3, ... 7	1：	发送至通道的同步动作
				2：	通道发出的同步动作
				3：	读取数据
				4：	写入数据
				5：	发送至通道的控制信号
				6：	发送至轴的控制信号
				7：	发送至轴的控制信号
S7Var：	E	ANY	S7 数据区域	取决于“Funct”	
IVAR1：	E	INT	---	取决于“Funct”	
IVAR2：	E	INT	---	取决于“Funct”	
Error：	A	BOOL	0 (FALSE), 1 (TRUE)	1：	存在故障
ErrCode：	A	INT	---	取决于“Funct”	

14.17.24.4 功能 1、2：发送至/来自通道的同步动作信号

同步动作可由 PLC 取消或使能。

其数据区域位于 DB21, ...DBB300 ...307（发送至通道）和 DB21, ...DBB308 ...315（来自通道）中。此功能中参数“S7Var”不会被分析，但是必须设定为一个当前参数（参见调用示例）。数据会在 FC21 运行期间立即传输至 NC/从 NC 传输。

信号	方式	类型	取值范围	含义	
Enable：	E	BOOL	0 (FALSE), 1 (TRUE)	1：	传输数据
Funct：	E	BYTE	1, 2	1： 2：	发送至通道 从通道发出
S7Var：	E	ANY	S7 数据区域	未使用	
IVAR1：	E	INT	1, 2, ... 最大通道编号	通道编号	
Error：	A	BOOL	0 (FALSE), 1 (TRUE)	1：	存在故障
ErrCode：	A	INT	1, 10	1： 10：	“Funct”无效 通道编号无效

调用示例：

```
FUNCTION FC100:VOID
VAR_TEMP
        myAny :ANY;
END_VAR

BEGIN
NETWORK

//在 NC 通道 1 中通过 ID3、ID10 和 ID31 取消同步动作：
SYAK :  AUF   DB21;
        SET;
        S     DBX300. //ID3
              2;
        S     DBX301. //ID10
              1;
        S     DBX303. //ID31
              6;
        L     B#16#1;
```

```
        T     MB11;
        SPA   TRAN;

//从 NCK 通道 1 发出的信号同步动作：
SYVK :  L B#16#2;
        T MB11;
TRAN:   CALL FC21 (
             Enable   := M 10.0,      //若为 True，则 FC 21 生效
             Funct    := MB 11,
             S7Var    := #myAny,      //未使用
             IVAR1    := 1,           //通道编号
             IVAR2    := 0,
             Error    := M 10.1,
             ErrCode  := MW 12);

END_FUNCTION
```

14.17.24.5 功能 3、4：PLC-NC 快速数据交换

概述

为实现 NC 与 PLC 之间的快速信息交换，设有充当接口的专属内部数据区域。该接口包含 4096 字节。通过 FC21 由 PLC 执行访问（读/写）。该接口的内部结构仅由用户定义，且 NC 侧与 PLC 侧必须一致。

Vom NC-Programm kann auf diese Daten mit den Befehlen $A_DBB[x], $A_DBW[x], $A_DBD[x], $A_DBR[x] zugegriffen werden (Literatur:Listenhandbuch Systemvariablen).

数据区域中的具体地址通过一个字节偏移（0 至 4095）在参数 “IVAR1” 中设定。此时必须根据数据格式选择对齐，即一个 Dword 要占用 4 字节区间，一个 Word 则占用 2 字节。字节可从数据区域内的任意偏移处开始。不支持对单个位的访问。其会被 FC21 转换成字节访问。数据类型信息和数据数量从通过 S7Var 传输的 ANY 参数获取。

通过 NC 和 PLC 进行访问时，若未采取额外的编程技术措施，则只能确保 1 字节和 2 字节访问的数据一致性。2 字节一致性也仅针对 WORD 或 INT 数据类型，而不适用于 BYTE。

在使用较大的数据类型或传输数组时，必须使用参数“IVAR2”中的信号量字节，这样 FC21 便可通过该字节识别数据块的有效性或一致性。此操作必须由 NC 支持，即在 NC 程序中写入或删除信号量字节。信号量字节和有效数据位于相同的数据区域。

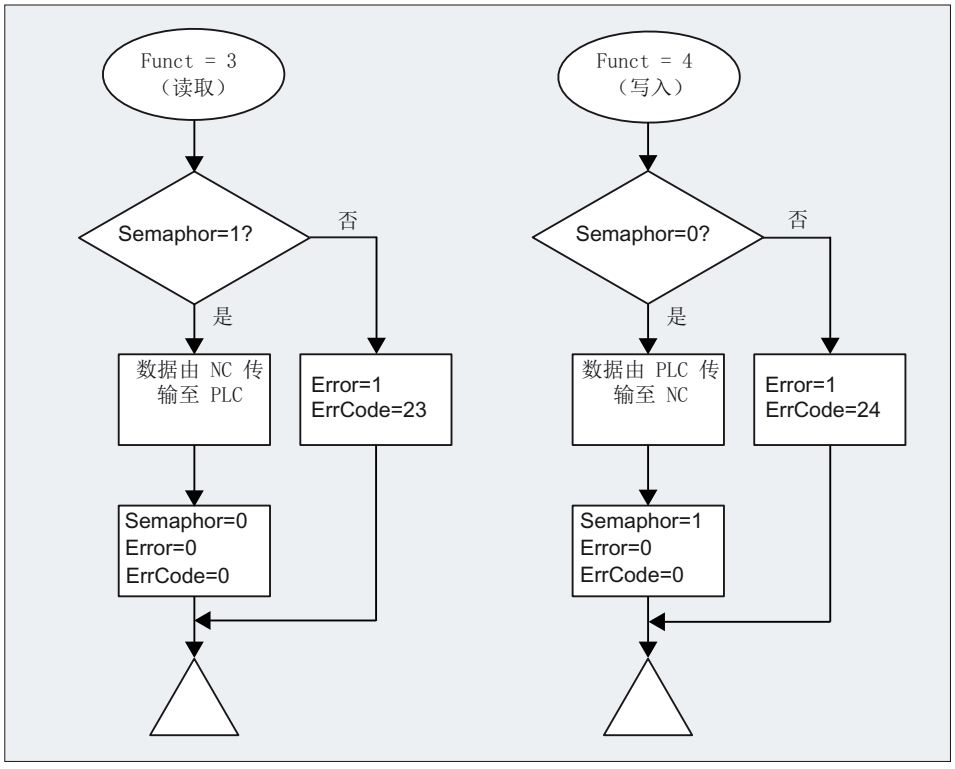
“IVAR2” 中 0 至 4095 的值即为信号量字节。

PLC 通过 FC21 在用于传输有效数据的调用中读取和写入信号量字节。PLC 编程人员只需要定义接口中的信号量变量。由 NC 通过 NC 程序访问时，必须按照下面的流程图通过单个指令编写信号量机制。读取或写入变量时须采用不同的顺序。

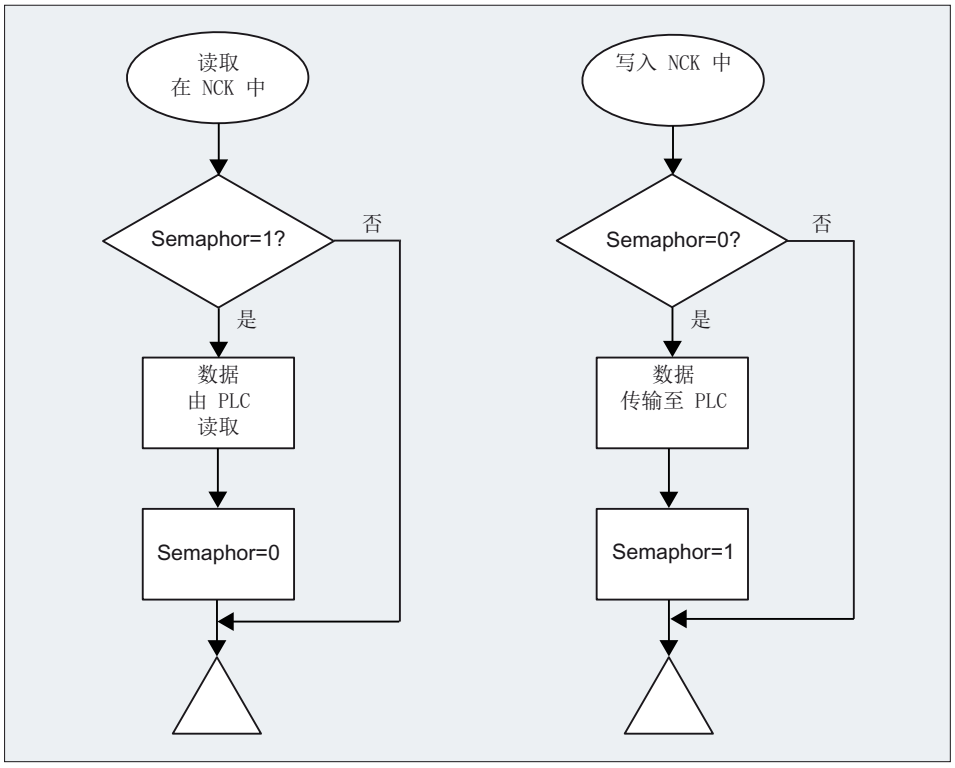
信号量机制仅直接支持单个变量或数组。其结构传输必须划分为单个任务。此时必须通过由用户自行编写的信号量机制来确保该结构的数据一致性。

若设置 IVAR2 = -1，那么数据传输将不启用信号量。

PLC 中启动信号量的数据交换（FC21 示意图）



NC 中的结构：



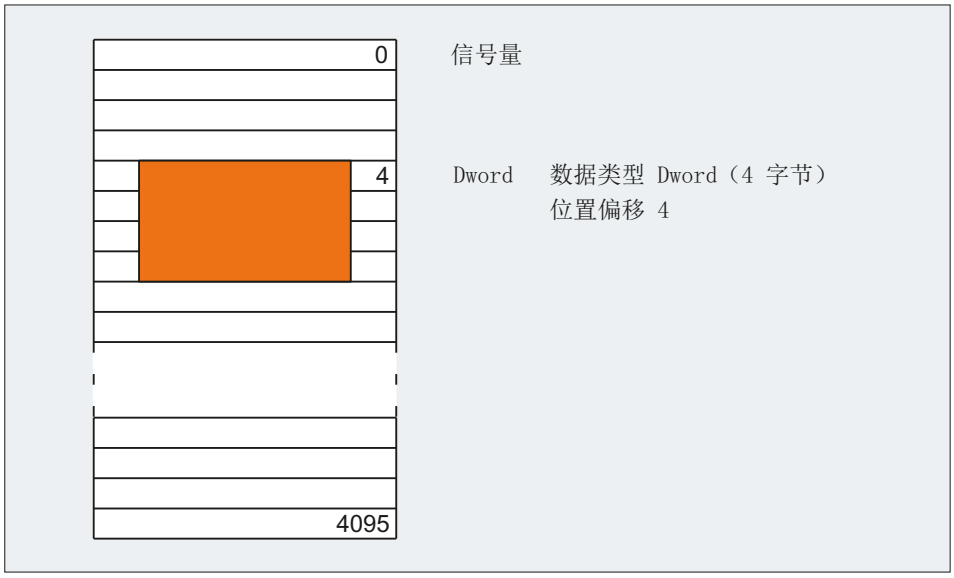
变量取值范围

信号	方式	类型	取值范围	含义	
Enable：	E	BOOL	0 (FALSE), 1 (TRUE)	1：	传输数据
Funct：	E	BYTE	3, 4	3：	读取数据
				4：	写入数据
S7Var：	E	ANY	S7 数据区域，除本地数据	源/目标数据区域	
IVAR1：	E	INT	0 ... 4095	位置偏移	
IVAR2：	E	INT	-1 ... 4095	信号量字节 传输不启用信号量：-1	
Error：	A	BOOL	0 (FALSE), 1 (TRUE)	1：	存在故障

信号	方式	类型	取值范围	含义	
ErrCode：	A	INT	20, 21, 22, 23, 24, 25	20：	对齐错误
				21：	不允许的位置偏移
				22：	不允许的信号量字节
				23：	无新数据可读取
				24：	无法写入数据
				25：	S7Var 中设置了本地数据

示例 1：启用字节 0 中的信号量读取位置偏移 4 的 DWORD，并保存在标志位双字 100 中

- 数据类型 Dword（4 字节）
- 位置偏移 4



PLC 编程

```
CALL  FC21(
        Enable     := M 10.0,                  //若为 TRUE，则 FC21 生效
```

```
        Funct      := B#16#3,              //读取数据
        S7Var      := P#M 100.0 DWORD 1,
        IVAR1      := 4,
        IVAR2      := 0,
        Error      := M 10.1,
        ErrCode    := MW12);
UN     M10.1;                 //Enable 保持 1，直至读取数据
R      M10.0;
```

通过同步动作进行 NC 编程

- 将数据写入 PLC，字节 0 用作信号量：
  ID=1 WHENEVER $A_DBB[0] == 0 DO $A_DBR[4] = $AA_IM[X] $A_DBB[0] = 1
- 从 PLC 读取数据，字节 1 用作信号量：
  ID=2 WHENEVER $A_DBB[1] == 1 DO $R1 = $A_DBR[12] $A_DBB[1] = 0

示例 2：以无信号量的方式读取位置偏移 8 的 WORD，并保存在标志位字 104 中

```
CALL   FC21(
        Enable     := M 10.0,              //若为 TRUE，则 FC21 生效
        Funct      := B#16#3,              //读取数据
        S7Var      :=P#M 104.0 WORD 1,
        IVAR1      :=8,
        IVAR2      :=-1,
        Error      := M 10.1,
        ErrCode    := MW12);
```

14.17.24.6 功能 5：更新发送至通道的控制信号

此功能用于在循环数据传输之间快速传输重要控制信号。用户接口 DB21, ... 的数据字节 6 和 7 会被传输至 NC。通道在参数“IVAR1”中设定。借助此功能可在 PLC 循环外传输例如进给禁止、读取禁止等信号。

信号	方式	类型	取值范围	含义	
Enable：	E	BOOL	0 (FALSE), 1 (TRUE)	1：	传输数据
Funct：	E	BYTE	5	5：	发送至通道的控制信号
S7Var：	E	ANY	S7 数据区域	未使用	
IVAR1：	E	INT	第 1 个最大通道	通道编号	

信号	方式	类型	取值范围	含义	
Error：	A	BOOL	0 (FALSE), 1 (TRUE)	1：	存在故障
ErrCode：	A	INT	1, 10	1：	“Funct”无效
				10：	通道编号无效

14.17.24.7 功能 6：更新发送至轴的控制信号

功能 6 用于在循环数据传输之间快速传输重要控制信号。用户接口 DB31, ... 的**数据字节 2** 会被传输至 NC。传输会针对所有激活的轴进行。借助此功能可在 PLC 循环外传输例如控制器使能等信号。

信号	方式	类型	取值范围	含义	
Enable：	E	BOOL	0 (FALSE), 1 (TRUE)	1：	传输数据
Funct：	E	BYTE	6	6：	发送至轴的控制信号
S7Var：	E	ANY	S7 数据区域	未使用	
IVAR1：	E	INT	0		
Error：	A	BOOL	0 (FALSE), 1 (TRUE)	1：	存在故障
ErrCode：	A	INT	1	1：	“Funct”无效

14.17.24.8 功能 7：更新发送至轴的控制信号

功能 7 用于在循环数据传输之间快速传输重要控制信号。用户接口 DB31, ... 的**数据字节 4** 会被传输至 NC。传输会针对所有激活的轴进行。借助此功能可在 PLC 循环外传输例如进给停止等信号。

信号	方式	类型	取值范围	含义	
Enable：	E	BOOL	0 (FALSE), 1 (TRUE)		
Funct：	E	BYTE	7	7：	发送至轴的控制信号
S7Var：	E	ANY	S7 数据区域	未使用	
IVAR1：	E	INT	0		
Error：	A	BOOL	0 (FALSE), 1 (TRUE)	1：	存在故障
ErrCode：	A	INT	1	1：	“Funct”无效

14.17.25 FC22: TM_DIR - 刀具管理的方向选择

注意
使用 模块 FC22“TM_DIR”只能与刀具管理配合使用！

功能

模块 FC22“TM_DIR”可基于位置编号（例如链式刀库或刀塔（分度轴）的刀位号）依据目标位置和当前位置，提供用于定位的最短行程和运行方向。

输出端

- 输入 FC22:“Start” = 1 ⇒ 输出端循环更新。
- 输入 FC22:“Start” = 0 ⇒ 输出端未定义。

特殊定位

Bei Richtungsauswahl mit Sonderpositionierung (Eingang FC22:"Offset" > 0) wird eine neue Zielposition aus der Zielposition, dem Offset für Sonderpositionierung, sowie der Anzahl der Magazinplätze, berechnet:

新目标位置 = (目标位置 - (特殊位置 -1)) 负 MODULO 刀库刀位数量

新的目标位置为：由用户请求的目标位置位于特殊位置的刀位编号上时，刀库必须定位至的目标刀位编号。

方向优化始终生效，与是否启用特殊定位无关。

调用

针对每个刀库都必须以相应的参数设置调用一次此模块。

文档

- 用于刀具管理的其它 PI 服务：
  - FB4:PI_SERV - 请求 PI 服务 (页 1080)
  - FC7：TM_REV - 刀塔换刀的传输模块 (页 1148)
  - FC8：TM_TRANS - 刀具管理的传输模块 (页 1152)
- 功能手册之刀具管理

功能说明

```
FUNCTION FC22:VOID
//NAME:            TM_DIR
VAR_INPUT
    MagNo :        INT;
    ReqPos :       INT;
    ActPos :       INT;
    Offset :       BYTE;
    Start :        BOOL;
END_VAR

VAR_OUTPUT
    Cw :           BOOL;
    Ccw :          BOOL;
    InPos :        BOOL;
    Diff :         INT;
    Error :        BOOL;
END_VAR
```

```
BEGIN
END_FUNCTION
```

对形式参数的说明

信号	方式	类型	取值范围	含义
MagNo：	E	INT	1, 2, 3, ...	刀库编号
ReqPos：	E	INT	1, 2, 3, ...	目标位置（刀库刀位号）
ActPos：	E	INT	1, 2, 3, ...	当前位置（刀库刀位号）
Offset：	E	BYTE	0, 1, 2, ...	用于特殊定位的偏移
Start：	E	BOOL	0 (FALSE), 1 (TRUE)	1 = 启动计算
Cw：	A	BOOL	0 (FALSE), 1 (TRUE)	1 = 刀库以顺时针运动
Ccw：	A	BOOL	0 (FALSE), 1 (TRUE)	1 = 刀库以逆时针运动
InPos：	A	BOOL	0 (FALSE), 1 (TRUE)	1 = 到达位置
Diff：	A	INT	0, 1, 2, ...	微分行程量（最短行程）
Error：	A	BOOL	0 (FALSE), 1 (TRUE)	1 = 故障

调用示例

```
CALL FC22(                          //刀具管理方向选择
                                    // 输入端
   MagNo :=      2,                 //刀库编号
   ReqPos :=     mw 20,             //目标位置
   ActPos :=     mw 22,             //当前位置
   Offset :=     b#16#0,            //用于特殊定位的偏移
   Start :=      m 30.4,            //Start 触发
                                    //输出端
   Cw :=         m 30.0,            //刀库按顺时针转动
   Ccw :=        m 30.1,            //刀库按逆时针转动
   InPos :=      m 30.2,            //刀库到达位置
```

```
Diff :=       mw 32,       //微分行程
Error :=      m 30.3       //出错
);
```

14.17.26 FC24: MCP_IFM2 - 将 MCP 信号传送到接口上

功能

模块 FC24“MCP_IFM2”（M 型，例如 MCP 310）用于将数据从机床控制面板传输至 NC/PLC 接口：

- 运行方式
- 轴选择
- WCS/MCS 切换
- 运行键
- 倍率或倍率仿真信号
- 钥匙开关位置

根据生效的运行方式或选择的坐标系，以下定义适用于**进给倍率、轴运行键**和 **INC 键**。

- **进给倍率：**
  - 进给倍率会传输至所选通道的接口和轴的接口。
  - 当 HMI 信号“快进的进给倍率生效”置位时，除了接口字节“快进补偿”（DBB 5）外，进给倍率信号还会传输至 NC 通道（特例：开关位置“零”）。此外“快进补偿生效”也会随此 HMI 信号置位。
- **机床功能 INC 键和轴运行键：**
  - 选择了 MCS 时，这些信号会被传输至所选机床轴的接口。
  - 选择了 WCS 时，这些信号会被传输至参数设置的通道的几何轴接口。
  - 在 MCS 和 WCS 间切换时，之前选择的轴一般会被取消。

对机床控制面板的 LED 的控制基于相应选择的反馈信息。

进给和主轴启动/停止不会传输至接口，而是作为信号“FeedHold”或“SpindleHold”模态输出。用户可将这些信号与其他用于引起进给停止或主轴停止的信号互联（这例如可通过 FC10:AL_MSG 的相应输入信号实现）。此外其对应的 LED 也会一同被控制。

主轴方向（+、-）也不会直接切换，而是作为输出参数“SpindleDir”提供。这例如可用于 FC18 的参数设置。此外主轴的使能通过参数“SpindleHold”切换。若需直接运行主轴，可通过相应的轴选择预选主轴，并通过（轴）方向键运行。

机床控制面板失灵时，其发出的信号会被设为零，输出信号“FeedHold”和“SpindleHold”同样如此。

在一个 PLC 循环中允许多次调用 FC24 以及 FC19、FC25、FC26。此时 LED 显示基于循环中的首次调用。此外，在首次调用中会执行参数设置的模块的所有动作。之后的调用中只会处理通道和 BAG 接口。对于几何轴，只需在循环中首次调用模块时设定方向。

单程序段模式的选择/取消只通过循环中首个调用进行控制。

当参数“BAGNo”提升 B#16#10 时，可处理第二块机床控制面板。进行参数设置时，BAG 编号包含在下部半位元组中（下面 4 个位）。

“BAGNo” = 0 或 B#16#10 ⇒ 不处理 BAG 信号。

“ChanNo” = 0 ⇒ 表示不处理通道信号。

INC 选择会被传输至 BAG 接口。此设定由此模块通过 DB10.DBX57.0（BAG 区域内的 INC 输入生效）在启动后单次激活。

此外还可通过该模块并行处理 2 个机床控制面板。此时必须将 OB1 循环中对 MCP 2 的模块调用设置到对 MCP 1 的开关位置“零”）。机床控制面板模块能够在特定界限内支持双 MCP（在为 2 个 MCP 指定了相同的轴时，不支持轴选择功能中的相互闭锁 ）。

钥匙开关位置

从软件版本 4.5 SP2 起钥匙开关信号也在 FC24 中传输至用户接口（DBX56.5 至 7）。传输与钥匙开关是否安装在 MCP 上无关。

说明

更多信息参见“FC19:MCP_IFM - 将 MCP 信号传送到接口上 (页 1186)”。

灵活的轴配置

选择轴时的指定方式以及机床轴编号的方向键都能体现出此模块的灵活性。

MCP 模块能够更好地支持 2 块机床控制面板的同时运行，特别针对 2 通道、2 BAG 的应用情形。在相应 MCP 的轴表格中，轴编号也可以 MCP 模块的 BAG 编号设定。

借助 DB10 中的轴编号表格便可实现此灵活性。
对于第 **1** 块机床控制面板（MCP），表格从字节 8 开始（符号名称：MCP1AxisTbl[1..22]）；对于第 **2** 块机床控制面板（MCP）则从字节 32 开始（符号名称：MCP2AxisTbl[1..22]）。此处必须以字节方式输入机床轴编号。可在轴表格中输入 0。系统不会检查是否有不允许的轴编号，输入错误时可能会引起 PLC 停止。

对于 **FC24**，也可对**可选择的最大轴数量**进行限制。
对于 MCP 1，此上限在 DB10.DBW30（符号名称：MCP1MaxAxis）中设置，对于 MCP 2 则在 DB10.DBW54（符号名称：MCP2MaxAxis）中设置。

其预设值为 0，此时配置的最大轴数量将作为上限生效。轴编号和限制也可启用动态调整。之后必须在 FC24 中重新执行轴选择。轴通过相应的方向键运行期间不允许切换轴编号。缺省设置为兼容模式，其中两个 MCP 的轴编号为 **1 至 6**，可选择的轴数上限为配置的轴数。

功能说明

```
FUNCTION FC24:VOID
// NAME:                MCP_IFM2

VAR_INPUT
    BAGNo :             BYTE;
    ChanNo :            BYTE;
    SpindleIFNo :       BYTE;
END_VAR

VAR_OUTPUT
    FeedHold :          BOOL;
    SpindleHold :       BOOL;
    SindleDir :         BOOL;
END_VAR

BEGIN
END_FUNCTION
```

对形式参数的说明

信号	方式	类型	取值范围	含义
BAGNo：	E	BYTE	B#16#00 - B#16#0A	**第 1 个 MCP**:运行方式信号需要传输至的 BAG 的编号。
			B#16#10 - B#16#1A	**第 2 个 MCP**:运行方式信号需要传输至的 BAG 的编号。
ChanNo：	E	BYTE	B#16#00 - B#16#0A	通道信号需要传输至的通道的编号

信号	方式	类型	取值范围	含义	
SpindleIFNo：	E	BYTE	0 - 31 (B#16#1F)	主轴数据需要传输至的进给轴/主轴的编号（对应机床轴的编号）	
FeedHold：	A	BOOL	0 (FALSE), 1 (TRUE)	MCP 进给停止，模态生效	
SpindleHold：	A	BOOL	0 (FALSE), 1 (TRUE)	MCP 主轴停止，模态生效	
SpindleDir：	A	BOOL	0 (FALSE), 1 (TRUE)	主轴旋转方向	
				0：	逆时针旋转方向
				1：	顺时针旋转方向

调用示例

```
CALL FC24(                          //M 型机床控制面板 slim 规格
                                    //发送至接口的信号
    BAGNo :=       B#16#1,          //BAG 编号 1
    ChanNo :=      B#16#1,          //通道编号 1
    SpindleIFNo :=B#16#4,           //主轴接口编号 = 4
    FeedHold :=    m22.0,           //进给停止信号，模态生效
    SpindleHold :=db2.dbx151.0,     //主轴停止在消息 DB 中模态生效
    SpindleDir := m22.1);           //主轴旋转方向反馈
```

基于此参数设置，信号会传输至 BAG 1、通道 1 和所有轴。此外主轴倍率会传输至进给轴/主轴接口 4。进给停止信号会传输至标志位 22.0，主轴停止信号传输至数据块 DB2、数据位 151.0。参数“SpindleDir”反馈的主轴旋转方向可用于 FC18 调用中的方向设定。

14.17.27 FC25: MCP_IFT - 将 MCP/BT 信号传送到接口上

功能

模块 FC25“MCP_IFT”（T 型，例如 MCP 483）用于将数据从机床控制面板传输至 NC/PLC 接口：

- 运行方式
- 四根轴的方向键

- WCS/MCS 切换
- 倍率
- 钥匙开关

根据生效的运行方式或选择的坐标系，以下定义适用于**进给倍率、轴运行键**和 **INC 键**。

- **进给倍率：**
  - 进给倍率会传输至所选通道的接口和轴的接口。
  - 当 HMI 信号“快进的进给倍率生效”置位时，除了接口字节“快进补偿”（DBB 5）外，进给倍率信号还会传输至 NC 通道（特例：开关位置“零”）。此外“快进补偿生效”也会随此 HMI 信号置位。
- **机床功能 INC 键和轴运行键：**
  - 选择了 MCS 时，这些信号会被传输至所选机床轴的接口。
  - 选择了 WCS 时，这些信号会被传输至参数设置的通道的几何轴接口。

对机床控制面板的 LED 的控制基于相应选择的反馈信息。

进给和主轴启动/停止不会传输至接口，而是作为信号“FeedHold”或“SpindleHold”模态输出。用户可将这些信号与其他用于引起进给停止或主轴停止的信号互联（这例如可通过 FC10:AL_MSG 的相应输入信号实现）。此外其对应的 LED 也会一同被控制。

机床控制面板失灵时，其发出的信号会被设为零，输出信号“FeedHold”和“SpindleHold”同样如此。

在一个 PLC 循环中允许多次调用 FC25 以及 FC19、FC24、FC26。此时 LED 显示基于循环中的首次调用。此外，在首次调用中会执行参数设置的模块的所有动作。之后的调用中只会处理通道和 BAG 接口。对于几何轴，只需在循环中首次调用模块时设定方向。

单程序段模式的选择/取消只通过首个循环进行控制。

当参数“BAGNo”提升 B#16#10 时，可处理第二块机床控制面板。进行参数设置时，BAG 编号包含在下部半位元组中（下面 4 个位）。

“BAGNo” = 0 或 B#16#10 ⇒ 不处理 BAG 信号。

“ChanNo” = 0 ⇒ 表示不处理通道信号。

灵活的轴配置

选择轴时的指定方式以及机床轴编号的方向键都能体现出此模块的灵活性。

MCP 模块能够支持 2 块机床控制面板的同时运行，特别针对“2 通道、2 BAG”的应用情形。此时必须将 OB1 循环中对 MCP 2 的模块调用设置到对 MCP 1 的开关位置“零”）。在相应 MCP 的轴表格中，轴编号也可以 MCP 模块的 BAG 编号设定。

借助 DB10 中的轴编号表格便可实现此灵活性。
对于第 **1** 块机床控制面板（MCP），表格从字节 8 开始（符号名称：MCP1AxisTbl[1..22]）；对于第 **2 块**机床控制面板（MCP）则从字节 32 开始（符号名称：MCP2AxisTbl[1..22]）。此处必须以字节方式输入机床轴编号。可在轴表格中输入 0。系统不会检查是否有不允许的轴编号，输入错误时可能会引起 PLC 停止。

对 **FC25 中可选择的最大轴数**通过轴表格中的 0 值进行限制。轴编号也可启用动态调整。在通过方向键手动运行轴的期间不允许切换轴编号。缺省设置为兼容模式，其中两个 MCP 的轴编号为 **1 至 4**，可选择的轴数上限为配置的轴数。

说明

更多信息参见“FC19:MCP_IFM - 将 MCP 信号传送到接口上 (页 1186)”。

功能说明

```
FUNCTION FC25:VOID
// NAME:              MCP_IFT

VAR_INPUT
    BAGNo :           BYTE;
    ChanNo :          BYTE;
    SpindleIFNo :     BYTE;
END_VAR

VAR_OUTPUT
    FeedHold :        BOOL;
    SpindleHold :     BOOL;
END_VAR

BEGIN
END_FUNCTION
```

对形式参数的说明

信号	方式	类型	取值范围	含义
BAGNo：	E	BYTE	B#16#00 - B#16#0A	第 1 个 MCP:运行方式信号需要传输至的 BAG 的编号。
			B#16#10 - B#16#1A	第 2 个 MCP:运行方式信号需要传输至的 BAG 的编号。
ChanNo：	E	BYTE	B#16#00 - B#16#0A	通道信号需要传输至的通道的编号
SpindleIFNo：	E	BYTE	B#16#00 - B#16#1F	主轴数据需要传输至的进给轴/主轴的编号（对应机床轴的编号）
FeedHold：	A	BOOL	0 (FALSE), 1 (TRUE)	MCP 进给停止，模态生效
SpindleHold：	A	BOOL	0 (FALSE), 1 (TRUE)	MCP 主轴停止，模态生效

调用示例

基于此参数设置，信号会传输至 BAG 1、通道 1 和所有轴。主轴倍率会传输至进给轴/主轴接口 4。进给停止会传输至标志位 22.0，主轴停止会传输至数据块 DB2, DBX151.0。

```
CALL FC25(                          //T 型机床控制面板
                                    //发送至接口的信号
  BAGNo :=         B#16#1,          //BAG 编号 1
  ChanNo :=        B#16#1,          //通道编号 1
  SpindleIFNo :=   B#16#4,          //主轴接口编号 = 4
  FeedHold :=      m22.0,           //进给停止信号，模态生效
  SpindleHold :=   db2.dbx151.0);   //主轴停止在消息 DB 中模态生效
```

14.17.28 FC26: HPU_MCP - 将 HT 8 信号传送到接口上

功能说明

```
FUNCTION FC26:VOID
//NAME:          HPU_MCP
```

```
VAR_INPUT
    BAGNo :      BYTE;
    ChanNo :     BYTE;
END_VAR

BEGIN
END_FUNCTION
```

形式参数说明

信号	方式	类型	取值范围	含义
BAGNo :	E	BYTE	B#16#00 - B#16#0A	第 1 个 MCP:运行方式信号需要传输至的 BAG 的编号。
			B#16#10 - B#16#1A	第 2 个 MCP:运行方式信号需要传输至的 BAG 的编号。
ChanNo :	E	BYTE	B#16#00 - B#16#0A	通道信号需要传输至的通道的编号

调用示例

为 MCP 1、BAG 1 和 NC 的第一个通道调用 FC26。

```
CALL FC26(                          //HT 8 的机床控制面板
          BAGNo :=    B#16#01,      //MCP 1，BAG 1
          ChanNo :=   B#16#01);     //通道 1
```

为 MCP 2、BAG 2 和 NC 的第三个通道调用 FC26。

```
CALL FC26(                          //HT 8 的机床控制面板
          BAGNo :=    B#16#12,      //MCP 2，BAG 2
          ChanNo :=   B#16#03 );    //通道 3
```

功能简介

FC26 “HPU_MCP（手持操作单元 HT 8 的机床控制面板信号）”用于在功能块 FB1 中设置的 HT 8 输入/输出数据区域（参数：MCPxIn 和 MCPxOut）和 NC/PLC 接口间传输下列功能的 HT 8 专用信号：

- 运行方式
- 机床功能 INC
- 坐标系 WCS 或 MCS
- 轴运行键
- 选择轴
- 进给倍率
- 快进倍率
- 钥匙开关信息

说明

通过 HT 8 和/或 HMI 切换运行方式

借助 FC2“GP_HP 基本程序循环部分”传输运行方式切换信号时，可选择通过 HT 8 的 MCP 或 HMI 执行。HMI 信号向 NC/PLC 接口的传输也可在功能块 FB1 中通过参数“MMCToIF”= FALSE 取消。

生效轴：

通过 HT 8 可同时响应最多 6 根轴。轴的选择须由用户/机床制造商在 PLC 用户程序中进行。

灵活的轴配置

通过 FC26 可将机床轴灵活地指定给运行键或轴选择功能。为此，DB10 中有 2 个表格可供使用：

- MCP 1 的机床轴 MCP：DB10.DBB8 至 DBB13（机床轴编号表）
  符号名称：MCP1AxisTbl[1..22]
- MCP 2 的机床轴 MCP：DB10.DBB32 至 DBB37（机床轴编号表）
  符号名称：MCP2AxisTbl[1..22]

必须将生效机床轴的编号 n（n = 1, 2, ...）以字节方式输入到这些表格中。未使用的位置必须输入 0 值。

可为 FC26 设定表格长度：

- MCP 1：DB10.DBB30（机床轴表格上限）
- MCP 2：DB10.DBB54（机床轴表格上限）

值为 4 时，表示 FC26 只考虑前 4 条记录（机床轴）。对于 FC26，最大值为 6。赋值为 0 或大于 6 时系统将隐性启用最大值。

说明

注意下列边界条件：

- 系统不会检查机床轴编号是否允许采用。无效的机床轴编号可能会引起 PLC 停止。
- 机床轴编号可动态修改。当机床轴正在通过运行键运行时，不允许对机床表格执行写入。

基于生效坐标系传输运行键信号

针对 6 根轴的运行键信号在 HT 8 输入数据区域中位于：

- EB n + 2，位 0 - 位 5（正运行方向）
- EB n + 3，位 0 - 位 5（负运行方向）

坐标系通过输入信号切换：

- EB n + 0，位 0（MCS/WCS）

输入信号在 FC26 中通过脉冲沿标志位分析。生效的坐标系在以下输出信号中显示：

- AB n + 0，位 0 （MCS/WCS），其中 0 = MCS，1 = WCS

MCS 生效时，轴 1 - 6 的运行键信号传输至机床轴表格（DB10.DBB8 至 DBB13，或 DBB32 至 DBB37）中设定的轴的轴专用接口（DB31, ...DBX4.6 和 DBX4.7（运行键 +/-））。

WCS 生效时，则会假定机床轴表中的轴 1 - 3 为几何轴。因此，

- 轴 **1 - 3**（EB n + 2 / 3，位 0 - 位 2）的运行键信号会传输至通过参数 “ChanNo” 设定的通道的几何轴接口 DB21, ...DBB12 + (n * 4)，其中 n = 0, 1, 2)，位 6 和位 7（运行键 +/-）。
  轴 1、2、3 的运行键信号与通道的几何轴 1、2、3 固定对应，不可修改。
- 轴 **4 - 6**（EB n + 2 / 3，位 3 - 位 5）的运行键信号会传输至机床轴表格中记录的轴 4 - 6（DB10.DBB**11** 至 DBB**13**，或 DBB**35** 至 DBB**37**）的轴专用接口（DB31, ...DBX4.6 和 DBX4.7（运行键 +/-））。

WCS 中不运行机床轴

WCS 生效时（AB n + 0，位 0 = 1），可闭锁机床轴的运行。为此须在 PLC 用户程序中设置以下输出信号：

- AB n + 3，位 7 = 1（WCS 中：非机床轴）
  请求 FC26 不为机床轴传输运行键信号。机床轴表格中轴 1 - 3 的运行键信号会传输至设定通道的几何轴 1 - 3。机床轴表格中轴 4 - 6 的运行键信号不会被传输。
- AB n + 2，位 6（轴 7 - n 已选择）
  请求 FC26 不传输机床轴，因为机床轴表格中的轴 1 - 6 已切换。这样轴 1 - 3 也不是几何轴，而是机床轴。

进给倍率

HT 8 倍率开关的值会作为进给倍率传输至编写的通道（参数：“ChanNo”）的通道专用接口 DB21, ...DBB4（进给倍率），以及 DB10.DBB8 至 DBB13（机床轴编号）中编写的轴的轴专用接口 DB31, ...DBB0（进给倍率）。

快进倍率

若为编写的通道（参数：“ChanNo”）设置了信号 DB21, ...DBX25.3 = 1（快进的进给倍率），那么 HT 8 倍率开关的值会作为快进倍率传输至通道专用接口 DB21, ...DBB5（快进补偿），且信号 DB21, ...DBX6.6 = 1（快进补偿生效）置位。

机床功能 INC

基于生效的坐标系 MCS 或 WCS，机床功能 INC 的 HT 8 信号会进行不同的传输：

- 生效坐标系：MCS
  针对全部 6 根轴，所选择的机床功能 INC 传输至 DB10.DBB8 至 DBB13 中编写的轴的轴专用接口 DB31, ...DBX5.0 至 DBX5.5（机床功能）。
- 生效坐标系：WCS
  对于轴 1 至 3，机床功能 INC 的信号传输至编写的通道（参数：“ChanNo”）的通道专用接口 DB21, ...DBX13.0 至 DBX13.5（机床功能）。
  对于轴 4 至 6，机床功能 INC 的信号传输至在表中在 DB10.DBB11 至 DBB13（机床轴编号）中编写的轴的轴专用接口 DB31, ...DBX5.0 至 DBX5.5（机床功能）。

INC 机床功能的选择信号会被传输至 BAG 专用接口 DB11.DBB 2 + (n * 20)，位 0 至位 5（n = 0, 1, 2, ... ）。FC26 会在启动后通过 DB10.DBX57.0（BAG 区域中 INC 输入生效）将“用于 INC 机床功能的 BAG 接口激活”这一信息提供给 NC）。

手轮选择

手轮选择信号由 HMI 分析，并传输至机床轴或几何轴的对应 NC/PLC 接口信号：

- 几何轴：DB21, ... DBB12 + (n * 4)，位 0 至位 2（n = 0, 1, 2）。
- 机床轴：DB31, ...DBX4.0 至 DBX4.2

前提条件：FB1 参数："HWheelMMC" = TRUE

一个 PLC 循环中的多次调用

在一个 PLC 循环中允许多次调用 FC26。在 PLC 循环中首次调用时，系统会：

- 执行设置的模块的所有动作
- 在输出区域写入 LED 信号
- 选择了 WCS 时，写入几何轴的运行键信号
- 处理单程序段模式的激活和取消信号

之后的 FC26 调用中，只会处理通道接口和 BAG 接口。

处理双 MCP

若在 PLC 程序的循环运行（组织块 OB1）中分别为两块 MCP 调用 FC26，必须将 MCP 2 的调用安排在 MCP 1 的调用之后。

说明

若一根轴通过这两个 MCP 均可运行，那么用户（机床制造商）须采取措施，确保相应情形下只有一个 MCP 生效。

HT 8 的 MCP 失灵

HT 8 的 MCP 失灵时，所有输入信号均会被设为 0。

14.17.28.1 HT 8 发出的 NC/PLC 接口信号一览

运行方式和机床功能

来源： MCP	目标： 编程设置的 BAG (参数 BAGNo) BAG 1 的显示
AUTOMATIC	DB11.DBX0.0
MDI	DB11.DBX0.1
JOG	DB11.DBX0.2
REPOS	DB11.DBX1.1
REF	DB11.DBX1.2
TEACH IN	DB11.DBX1.0
INC 1 ... 10 000, INC Var.	DB11.DBX2.0 - DBX 2.5

运行键和快进倍率

来源：MCP	目标：编写的通道的几何轴（参数：ChanNo）
运行键 +	DB21, ...DBX12.7
运行键 -	DB21, ...DBX12.6
快进倍率	DB21, ...DBX12.5
运行键 +	DB21, ...DBX16.7
运行键 -	DB21, ...DBX16.6
快进倍率	DB21, ...DBX16.5
运行键 +	DB21, ...DBX20.7
运行键 -	DB21, ...DBX20.6
快进倍率	DB21, ...DBX20.5

来源：MCP	目标：编写的轴，对应 DB10， DBB8 - 13（MCP 1）或 DBB32 - 37（MCP 2）中的表
运行键 +	DB31, ...DBX4.7
运行键 -	DB31, ...DBX4.6
快进倍率	DB31, ...DBX4.5

倍率

来源：MCP	目标：编写的通道（参数：ChanNo）
进给倍率	DB21, ...DBB4

来源：MCP	目标：编写的轴，对应 DB10， DBB8 - 13（MCP 1）或 DBB32 - 37（MCP 2）中的表
进给倍率	DB31, ...DBB0

通道信号

来源：MCP	目标：编写的通道（参数：ChanNo）
NC 启动	DB21, ...DBX7.1
NC 停止	DB21, ...DBX7.3
复位	DB21, ...DBX7.7
单程序段	DB21, ...DBX0.4

14.17.28.2 发送至 HT 8 的 NC/PLC 接口信号一览

运行方式和机床功能

目标：MCP	来源： 接口 DB (参数 BAGNo) BAG 1 的显示
AUTOMATIC	DB11.DBX6.0
MDI	DB11.DBX6.1
JOG	DB11.DBX6.2
REPOS	DB11.DBX7.1
REF	DB11.DBX7.2
TEACH IN	DB11.DBX7.0

14.17.29 FC19, FC24, FC25, FC26 - 源代码说明

任务

机床控制面板对用户接口的传输（FC19 M 型、FC24 Slim 型、FC25 T 型、FC26 HT 8 型）

对应模块

DB7，BAG 数量、通道、轴

DB7，机床控制面板的 Pointer

DB8，用于下一个循环的保存信息

使用的资源

无。

简介

模块 **FC19**（M 型）、**FC24**（Slim 型）、**FC25**（T 型）、**FC26**（HT 8 型）用于和用户接口交换机床控制面板信号。

参数

- 输入参数 “BAGNo”
  由数据块处理的 BAG 通过参数 “BAGNo” 选择。机床控制面板的编号也通过“BAGNo”参数选择（位 4）。
- 输入参数 “ChanNo”
  需要处理的通道通过参数 “ChanNo” 选择。
- 输入参数 “SpindleIFNo”（**非 FC26**）
  在参数 “SpindleIFNo” 中给定主轴数据需要传输至其接口的进给轴/主轴编号（对应的机床轴的编号）。主轴倍率会被传输至该主轴接口。输入参数会被检查是否存在设置错误。
- 输出参数 “FeedHold” 和 “SpindleHold”（**非 FC26**）
  输出参数 “FeedHold” 和 “SpindleHold” 由进给禁止、主轴禁止、进给使能、主轴使能这 4 个键构成。进给/主轴禁止激活时，反馈值“逻辑 1”。

用于下一个循环的信息会根据机床控制面板编号保存在 DB8，字节 0 至 3 或字节 62 至 65 中。这些信息是脉冲沿标志位、进给值和所选择的轴编号。含有效数据的模块由 DB7 中的 Pointer 参数“MCP1In”和“MCP1Out”（“MCP2In”和“MCP2Out”）提供。该 Pointer

通过 DB7 的 VAR 部分中的另一个指针间接定址，以避免绝对定址。该附加指针已在 FB1 中作为符号确定。

模块说明

所有 4 个模块结构相似，并针对子任务划分：

- Netzwerk Input（输入网络）
  在 Netzwerk Input（输入网络）中将各种参数复制到本地变量。此时机床控制信号（输入/输出区域有效数据）也会通过各种指针在 DB7 中转存。出于效率原因，这些本地变量在模块中处理。部分数值会被初始化以用于启动。
- Netzwerk Global_IN（全局输入网络）
  在 Netzwerk Global_IN（全局输入网络）中会为之后的处理确定通过脉冲沿检测进行的 MCS/WCS 切换、轴选择、方向键和快进叠加。在此程序部分中会执行用户自定义的修改，主要针对轴选择。
- Netzwerk NC（NC 网络）
  在 Netzwerk NC（NC 网络）中只会复制钥匙开关信息。
- Netzwerk BAG（BAG 网络）
  Netzwerk BAG（BAG 网络）会将运行方式作为动态信号传输至 NC。NC 的 INC 反馈会暂时为对应的 LED 保存。BAG 编号为 0 时则不处理此网络。若编号过大，则会触发消息 401901 或 402501，并切换至停止。
- Netzwerk Channel（通道网络）
  在 Netzwerk Channel（通道网络）中，NC 启动、停止、复位和单程序段功能会通过对应的反馈信息激活。进行了对应预选时将提供几何轴的方向键，反之则不提供。通道编号为 0 时则不处理此网络。若编号过大，则会触发消息 401902 或 402502，并切换至停止。
- Netzwerk Spindle（主轴网络）
  Netzwerk Spindle（主轴网络）将主轴倍率传输至通过 “SpindleIFNo” 设置的接口。
- Netzwerk Axes（轴网络）
  Netzwerk Axes（轴网络）将进给倍率传输至所选择的轴接口。方向键会被指定给所选择的轴/主轴。在之前选择的轴上方向信息会被设置为 0。
- Netzwerk Global_OUT（全局输出网络）
  Netzwerk Global_OUT（全局输出网络）中会对输出参数进行预处理，并生成 INC 机床功能的 LED 信号。
- Netzwerk Output（输出网络）
  Netzwerk Output（输出网络）将机床控制面板的输出信号从 VAR_TEMP 的映射传输至逻辑地址。之后为下一个循环保存数据。

轴选择扩展

Netzwerk Global_IN（全局输入网络）

若需选择超过 9 根轴，必须在 Netzwerk Global_IN 中进行调整。若此时还使用其他机床控制面板按键和 LED，则须采取以下措施：

1. 使用 UD DW#16#Wert（注释：清空所有轴 LED 用于显示）为轴选择删除定义的所有 LED。此时通过位掩码处理 9 个轴选择 LED。
2. 使用 UW W#16# 指令（注释："屏蔽所有轴选择按钮）检查新的方向选择是否成功。此时必须调整位域。
3. 必须为跳转栏（SPL）扩展新的跳转标记。新的跳转标记必须以降序插入标记 m009 前。必须为新的跳转标记添加选择信息，如 m009、m008 所显示的一样。

说明

根据需要，这些模块可作为 SPL 源供货。但其不一定为当前的模块版本。当前 C 环境下的研发在某些细节上已更加深入。建议注明对模块的特别要求，并将这些要求通过销售部门传达给项目管理部门。

14.17.30 FC1005：AG_SEND - 将数据传输至以太网 CP

功能

功能块 AG_SEND 会将需要通过配置的连接传输的数据传递给以太网 CP。配合功能块 AG_RECV 可通过集成“CP 840D sl”实现与另一个工作站的数据交换。为此必须在 STEP 7，“NetPro”中配置该工作站。

在基本程序中可通过功能块 FC1005 启用此功能。此功能块接近“SIMATIC_NET_CP”库中的功能块 FC5。

其支持的协议包括 TCP、UDP 和 ISO-on-TCP。

说明

启用 SINUMERIK CP 及 UDP 或 ISO-on-TCP 协议时，可传输的数据数量较少

采用 SINUMERIK CP，并启用 UDP 或 ISO-on-TCP 协议时，可通过功能块 FC1005 传输的数据仅为 **240** 字节。

形式参数说明

信号	方式	类型	取值范围	含义
ACT：	E	BOOL	0 (FALSE), 1 (TRUE)	任务触发 1)
ID：	E	INT	1, 2, 3 ...16	连接 ID
LADDR：	E	WORD	-	模块起始地址 2)
SEND：	E	ANY	-	设定地址和长度。数据区域的地址可选择参照： ● 标志位区域 ● 数据块区域
LEN：	E	INT	TCP：1, 2, ... 8192 UDP：1, 2, ... **240** ISO-on-TCP：1, 2, ... **240**	传输的字节数
DONE：	A	BOOL	0 (FALSE), 1 (TRUE)	0：任务执行中 1：任务已执行
ERROR：	A	BOOL	0 (FALSE), 1 (TRUE)	0：无故障 1：存在故障
STATUS：	A	WORD	-	状态显示
1) 参数 ACT 必须一直为 TRUE，直至：(DONE == 1) 或 (ERROR == 1) 2) 在 SINUMERIK 840D sl 上：参数 LADDR：= **W#16#8110**				

文档

对功能块的详细描述参见：

- SINUMERIK 操作界面：在线帮助
- SIMATIC 编程手册：SIMATIC NET S7-CP 的程序块
  章节：“用于工业以太网的程序块” > “用于开放式通讯服务（SEND/RECEIVE 接口）的程序块” > “AG_SEND / AG_LSEND / AG_SSEND”

14.17.31 FC1006：AG_RECV - 从以太网 CP 接收数据

功能

功能块 AG_RECV 用于从以太网 CP 接收通过配置的连接传输的数据。配合功能块 AG_SEND 可通过集成“CP 840D sl”实现与另一个工作站的数据交换。为此必须在 STEP 7，“NetPro”中配置该工作站。

在基本程序中可通过功能块 FC1006 启用此功能。此功能块接近“SIMATIC_NET_CP”库中的功能块 FC6。

其支持的协议包括 TCP、UDP 和 ISO-on-TCP。

说明

启用 SINUMERIK CP 及 UDP 或 ISO-on-TCP 协议时，可传输的数据数量较少

采用 SINUMERIK CP，并启用 UDP 或 ISO-on-TCP 协议时，可通过功能块 FC1006 传输的数据仅为 **240** 字节。

形式参数

信号	方式	类型	取值范围	含义
ID：	E	INT	1, 2 ...16	连接 ID
LADDR：	E	WORD	-	模块起始地址 1)
RECV：	E	ANY	-	设定地址和长度。数据区域的地址可选择参照： ● 标志位区域 ● 数据块区域
NDR：	A	BOOL	0 (FALSE), 1 (TRUE)	0：任务执行中 1：接收新数据
ERROR：	A	BOOL	0 (FALSE), 1 (TRUE)	0：无故障 1：存在故障
STATUS：	A	WORD	-	状态显示
LEN：	A	INT	TCP：1, 2, ... 8192 UDP：1, 2, ... **240** ISO-on-TCP：1, 2, ... **240**	传输的字节数
1) 在 SINUMERIK 840D sl 上：参数 LADDR：= **W#16#8110**				

文档

对功能块的详细描述参见：

- SINUMERIK 操作界面：在线帮助
- SIMATIC 编程手册：SIMATIC NET S7-CP 的程序块
  章节：“用于工业以太网的程序块” > “用于开放式通讯服务（SEND/RECEIVE 接口）的程序块” > “AG_RECV / AG_LRECV / AG_SRECV”

14.18 信号/数据说明

14.18.1 接口信号 NCK/PLC、HMI/PLC、MCP/PLC

文档

接口信号一览请见：

参数手册之“NC 变量和接口信号”

下面的章节中将对由 PLC 基本程序分析、处理和传输至用户接口的 NC 接口信号进行说明。

14.18.2 解码 M 信号

在零件程序、ASUB 或同步动作中编写的 M 功能由 NC 针对通道传输至 PLC：

- 通道 1 中的 M 功能：DB21
- 通道 2 中的 M 功能：DB22
- ...

DB21, ...	含义
DBX194.0 ... 7	M 功能 M0 ... M7
DBX195.0 ... 7	M 功能 M8 ... M15
DBX196.0 ... 7	M 功能 M16 ... M23

DB21, ...	含义
DBX197.0 ... 7	M 功能 M24 ... M31
DBX198.0 ... 7	M 功能 M32 ... M39
DBX199.0 ... 7	M 功能 M40 ... M47
DBX200.0 ... 7	M 功能 M48 ... M55
DBX201.0 ... 7	M 功能 M56 ... M63
DBX202.0 ... 7	M 功能 M64 ... M71
DBX203.0 ... 7	M 功能 M72 ... M79
DBX204.0 ... 7	M 功能 M80 ... M87
DBX205.0 ... 7	M 功能 M88 ... M95
DBX206.0 ... 3	M 功能 M96 ... M99

用于各个 M 功能的信号在 NC/PLC 接口中至少要持续一个 PLC 循环。

说明

以下主轴专用 M 功能不会传输至 NC/PLC 接口：M3、M4、M5 和 M70。

零件程序结束 M02/M30

输出 M 功能 M02/M30 不表示零件程序结束。输出 M 功能 M02/M30 源自异步子程序 (ASUB) 或同步动作，在这些情形下均与实际零件程序结束无关。为了在通道中明确获知零件程序结束，必须分析以下接口信号：

DB21, ...DBX35.7（通道状态：复位）

14.18.3 G 指令

在零件程序、ASUB 或同步动作中编写的 G 指令由 NC 针对通道传输至 PLC：

- 通道 1 中的 G 指令：DB21
- 通道 2 中的 G 指令：DB22
- ...

DB21, ...	含义
DBB208	G 指令组 1 中生效的 G 指令
DBB209	G 指令组 2 中生效的 G 指令
DBB210	G 指令组 3 中生效的 G 指令
...	...
DBB271	G 指令组 64 中生效的 G 指令

各个 G 指令在 NC/PLC 接口中至少要持续一个 PLC 循环。

DB21, ... 中的地址

DB21, ... 中的 G 指令组的地址 DBBx 按如下方式计算：

字节编号 = 208 + “G 指令组的编号” - 1

上电后的初始位置

上电后，系统会为所有 G 指令组传输值零，即生效的 G 指令组未定义。

零件程序结束或终止

零件程序结束或终止后，保留最后生效的 G 指令。

NC 启动

NC 启动后，以下机床数据中设定的 8 个 G 指令组的值会根据通过机床数据定义的初始设置以及零件程序中编写的值改写：
MD22510 $MC_GCODE_GROUPS_TO_PLC

文档

G 指令一览请见：

编程手册之基本原理；章节：“表格” > “G 指令”

14.18.4 DB2 中的消息信号

借助 DB2 能够在操作面板前端上为单个信号输出消息。这些信号被划分为预定义组别。消息出现、消失或被应答时，消息编号列中的编号会被传输至 HMI。针对每个消息编号都可在 HMI 保存一段文本。

文档：

- 参数手册之“NC 变量和接口信号”；章节“接口信号一览” > “PLC 报警/消息”
- 调试手册；“报警和消息文本”章节

说明

用户区域的数量可通过 FB1 设置。

配置（FB1: MsgUser）修改后必须删除 DB2 / DB3。

14.19 STEP 7 中编程的说明

以下章节中给出了 STEP 7 中复杂的编程流程和功能的简化说明：

- 复制数据 (页 1229)
- 数据类型 ANY 和 POINTER (页 1229)
- 多实例 DB (页 1235)
- 字符串 (页 1236)
- 确定数据块结构中的偏移地址 (页 1237)
- FB 调用 (页 1237)

文档

对 ANY 和 POINTER 数据类型的结构的基本说明参见：

SIMATIC STEP 7 手册；章节：“设计用户程序” > “CPU 标签和数据存储”

14.19.1 复制数据

复制方案

为了实现数据从一个 DB 到另一个的快速复制，建议：

- **在数据量较大时**使用系统功能 SFC BLKMOV 或 SFC FILL，因为这样能显著提升复制速度。
- **在数据量较小时**使用下面的程序，因此向 SFC 提供 ANY 参数需要花费额外的时间。

示例

```
                                    // DB xx.[AR1] 为复制源
                                    // DI yy.[AR2] 为目标
AUF         DB 100;                 //源 DB
LAR1        P#20.0;                 //源起始地址位于数据字节 20
AUF         DI 101;                 //目标 DB
LAR2        P#50.0;                 //目标起始地址位于数据字节 50
                                    //提前载入 AR1、AR2、DB、DI
L           4;                      //传输 8 字节
M001:
L           DBW [AR1,P#0.0];        //字方式复制
T           DIW [AR2,P#0.0];
+AR1        P#2.0;
+AR2        P#2.0;
TAK;
LOOP        M001;
```

14.19.2 ANY 和 POINTER

下面的程序示例中将演示编程机制。 其将显示如何在一个 FC 或 FB 内通过数据类型 POINTER 或 ANY 访问输入、输出及传输变量（VAR_INPUT、VAR_OUTPUT、VAR_IN_OUT）。 访问将采用部分符号的形式编写。

14.19.2.1 在 FC 中使用 POINTER 和 ANY

功能

FC99 有定义为 POINTER 或 ANY 的参数。

此示例中显示的是一个主干程序，用于访问 POINTER 或 ANY 的子分量。 此处通过 POINTER、ANY 参数设置的 DB 会被打开，地址偏移作为跨区域指针保存在地址记录器 AR1 中。 这样便可访问通过 POINTER、ANY 定址的变量的数据单元（通常为结构、数组）。

示例中此访问在相应程序串末尾显示。 对于 ANY 数据类型，在访问变量时还可基于数据类型和单元的数量执行检查或跳转。

示例

```
FUNCTION FC99:VOID
VAR_INPUT
   Row : BYTE;
   Convert : BOOL;                    //激活数字转换
   Addr : POINTER;                    //指向变量
   Addr1 : ANY;
END_VAR
VAR_TEMP
   dbchr : WORD;
   数量 :WORD;
   类型 : BYTE;
END_VAR
BEGIN
NETWORK
TITLE =
                                      //POINTER
   L            P##Addr;
   LAR1 ;                             //取回 Pointer
   L            W [AR1,P#0.0];        //取回 DB 编号
   T            #dbchr;
   L            D [AR1,P#2.0];        //Pointer 的偏移部分
LAR1 ;
AUF DB [#dbchr];                      //打开变量的 DB
L B [AR1,P#40.0];                     //通过 Pointer 取回字节值
                                      //通过地址偏移 40
                                      //ANY
   L            P##Addr1;
   LAR1 ;                             //取回 ANY
   L            B [AR1,P#1.0];        //取回类型
   T            #类型;
   L            W [AR1,P#2.0];        //取回数量
   T            #数量;
```

```
   L             W [AR1,P#4.0];        //取回 DB 编号
   T             #dbchr;
   L             D [AR1,P#6.0];        //Pointer 的偏移部分
LAR1 ;
   AUF           DB [#dbchr];          //打开变量的 DB
   L             B [AR1,P#0.0];        //通过 ANY 取回字节值
```

14.19.2.2 在 FB 中使用 POINTER 和 ANY

功能

FB99 有定义为 POINTER 或 ANY 的参数。

此示例中显示的是一个主干程序，用于访问 POINTER 或 ANY 的子分量。 此处通过 POINTER、ANY 参数设置的 DB 会被打开，地址偏移作为跨区域指针保存在地址记录器 AR1 中。 这样便可访问通过 POINTER、ANY 定址的变量的数据单元（通常为结构、数组）。

示例中此访问在相应程序串末尾显示。 对于 ANY 数据类型，在访问变量时还可基于数据类型和单元的数量执行检查或跳转。

示例

```
FUNCTIONBLOCK FB99
VAR_INPUT
   Row :BYTE;
   Convert : BOOL;                          //激活数字转换
   Addr : POINTER;                          //指向变量
   Addr1 :ANY;
END_VAR
VAR_TEMP
   dbchr :WORD;
   数量 :WORD;
   类型 :BYTE;
END_VAR
BEGIN
NETWORK
TITLE =
                                            //POINTER
   L               P##Addr;
   LAR1 ;                                   //从实例 DB 取回 Pointer
```

```
L               DIW [AR1,P#0.0];        //取回 DB 编号
T               #dbchr;
L               DID [AR1,P#2.0];        //Pointer 的偏移部分
LAR1 ;
AUF             DB [#dbchr];            //打开变量的 DB
L               B [AR1,P#40.0];         //通过 Pointer 取回字节值
                                        //通过地址偏移 40
                                        //ANY
L               P##Addr1;
LAR1 ;                                  //从实例 DB 取回 ANY
L               DIB [AR1,P#1.0];        //取回类型
T               #类型;
L               DIW [AR1,P#2.0];        //取回数量
T               #数量;
L               DIW [AR1,P#4.0];        //取回 DB 编号
T               #dbchr;
L               DID [AR1,P#6.0];        //Pointer 的偏移部分
LAR1 ;
AUF             DB [#dbchr];            //打开变量的 DB
L               B [AR1,P#0.0];          //通过 ANY 取回字节值
```

14.19.2.3 用于传输至 FC 或 FB 的 POINTER 或 ANY 变量

POINTER 或 ANY 变量

从 STEP 7 版本 1 开始，可在 VAR_TEMP 中定义一个 POINTER 或 ANY。

下面以两个示例显示如何提供 ANY。

示例 1：通过一个选择列表将 ANY 传输至另一个 FB（FC）。

在一个 FB（FC）上定义了**多个** ANY 参数。 现在需要通过一个选择列表将一个特定的 ANY 传输至另一个 FB（FC）。 这只能通过一个 VAR_TEMP 中的 ANY 实现。在参数“WelcherAny”中可设置 1 至 4，用于选择 Addr1 至 Addr4。

说明

在模块中使用地址记录器 AR2。 地址记录器 AR2 也在多实例 DB 中使用。 因此此 FB **不可**作为多实例 DB。

```
FUNCTIONBLOCK FB100
CODE_VERSION1                          //从 STEP 7 版本 2 起，用于
                                       //取消多实例 DB
VAR_INPUT
WelcherAny : INT;
   Addr1 : ANY;                        //注意固定顺序
   Addr2 : ANY;
   Addr3 : ANY;
   Addr4 : ANY;
END_VAR
VAR_TEMP
   dbchr :WORD;
   数量 :WORD;
   类型 :BYTE;
   Temp_addr :ANY;
END_VAR
BEGIN
NETWORK
TITLE =
L        WelcherAny;
DEC 1;
L        P#10.0;                       //每个 ANY 10 字节
*I;
LAR2;
L        P##Addr1;
+AR2;                                  //添加 ANY 的起始地址
L        P##Temp_addr;
LAR1 ;                                 //从 VAR_TEMP 取回 Pointer
L        DID [AR2,P#0.0];              //以 VAR_TEM 传输指针值
T        LD [AR1,P#0.0];
L        DID [AR2,P#4.0];
T        LD [AR1,P#4.0];
L        DIW [AR2,P#8.0];
T        LW [AR1,P#8.0];

CALL FB101, DB100
         (ANYPAR := #Temp_addr);       //ANYPAR 是数据类型 ANY
```

示例 2：将之前构建的 ANY 参数传输至另一个 FB（FC）

需要将一个之前构建的 ANY 参数传输至另一个 FB（FC）。 这只能通过一个 VAR_TEMP 中的 ANY 实现。

```
FUNCTIONBLOCK FB100
VAR_INPUT
   DB 编号: INT;
   DB 偏移 : INT;
   数据类型:  INT;
   数量 : INT;
END_VAR
VAR_TEMP
   dbchr :WORD;
   Temp_addr :ANY;
END_VAR
BEGIN
NETWORK
TITLE =
L        P##Temp_addr;
LAR1 ;                              //从 VAR_TEMP 取回 Pointer
L        B#16#10;                   //ANY 标识
T        LB [AR1,P#0.0];
L        数据类型;
T        LB [AR1,P#1.0];
L        数量;
T        LW [AR1,P#2.0];
L        DB 编号;
T        LW [AR1,P#4.0];
L        DB 偏移 :
SLD 3;                              //偏移为位偏移
T        LD [AR1,P#6.0];
CALL FB101, DB100
         (ANYPAR := #Temp_addr);    //ANYPAR 是数据类型 ANY
```

14.19.3 多实例 DB

功能

从 STEP 7 版本 2 起 FB 可支持多实例，即可提供多实例 DB。 多实例 DB 的特点是：一个数据块能用于各种 FB 实例（参见 STEP 7 文档）。 这样便可实现 DB 的组态范围优化。

多实例 DB 应只在需要使用的时候激活，因为其会提升 FB 的运行时间和代码大小。

说明

启用多实例的 FB 中有包含指针和地址记录器的复杂程序时，对于该 FB，用户须遵循下列规则：

启用多实例时，变量（VAR_INPUT、VAR_OUTPUT、VAR_IN_OUT、VAR）的起始地址通过 DI 数据块记录器和地址记录器 AR2 传输。 在支持多实例的 FB 中进行访问时，编译器通过地址记录器 AR2 独立控制对这些变量的访问。 但若该 FB 中的复杂程序部分必须通过数据记录器运作（例如复制数据），则必须在修改 AR2 前保存之前的内容。 访问实例变量（VAR_INPUT、VAR_OUTPUT、VAR_IN_OUT、VAR）前，必须将 AR2 记录器重新恢复为原始内容。 实例的 AR2 记录器最好保存到一个本地变量（VAR_TEMP）中。

“载入对实例变量的指针”指令会输出一个从实例数据起点开始的指针值。 为了能通过指针对此变量进行访问，必须添加上 AR2 中的偏移。

示例

```
FUNCTION_BLOCK FB99
VAR_INPUT
   varin: INT;
END_VAR
VAR
   variable1: ARRAY[0..9] OF
   INT;
   variable2: INT;
END_VAR
BEGIN
L           P##variable1;     //指向 ARRAY 开始处的 Pointer
                              //累加器中现在的值为 8500 0010
                              //AR2 中也有一个跨区域指针。
                              //若需跨区域工作，
                              //必须在将两个指针相加时
                              //抑制一个区域。
UD          DW#16#00FF_FFFF, //抑制区域
```

```
LAR1                          //载入 AR1
TAR2;
+AR1 AR2;                     //添加 AR2 实例偏移
                              //现在可通过 AR1 间接访问
                              //variable1 的 ARRAY
L           DIW [AR1, P#0.0];//例如访问第一个单元
END_FUNCTION_BLOCK
```

14.19.4 字符串

STRING 数据类型为特定的基本程序服务所需。 因此本部分会对字符串的结构及参数指定时的一般方法进行说明。

字符串的结构

STRING 类型的数据一般在数据块中保存（定义）。 其有 2 种定义方式：

1. 只将数据类型 STRING 指定给一个变量。 此时 STEP 7 编译器会生成一个 254 字符的长度。
2. 将数据类型 STRING 以及长度设定（在方括号中设定，例如 [32]）一起指定给变量。 此时 STEP 7 编译器会根据设定生成字符串长度。

STRING 类型的变量在保存时总是比定义多占用 2 个字节。 其中第 1 个字节中是可由 STEP 7 编译器保存的最大字符数。 第 2 个字节中则是实际使用的字符数。 通常此处是由 STEP 7 编译器保存的指定字符串的有效长度。 字符从第 3 个字节开始保存（每个字符 1 字节）。

基本程序模块上 STRING 参数的指定一般通过 POINTER 或 ANY 数据类型进行。指定通常采用符号编程。 此时须将待设置的字符串所处于的数据块接收至符号列表。 之后必须通过数据块的符号名称、接下来的点和 STRING 变量的符号名称执行对基本程序模块的指定。

14.19.5 确定数据块结构中的偏移地址

功能

另一项常会出现的任务是在 DB 内确定符号形式的偏移地址。 例如该 DB 中一个任意位置有一个 ARRAY 或 STRUKTUR。 此时需要在通过起始地址以符号形式载入地址记录器后，通过地址记录器访问 ARRAY 或 STRUKTUR 的单个单元。 以符号形式载入地址记录器的一种方法是借助一个将 Pointer 作为输入参数的 FC。 在程序中为该 FC 输入参数以符号指定 ARRAY 或 STRUKTUR 的地址。 FC 中的程序代码从输入参数确定偏移地址，并在地址记录器（AR1）中将该偏移地址传输至主叫功能。 这样便可在间接访问时采用符号定址。

示例

```
FUNCTION FC99:VOID
VAR_INPUT
    Addr : POINTER;          //指向变量
END_VAR
BEGIN
NETWORK
TITLE =
L        P##Addr;
LAR1 ;                       //从 Addr 取回 Pointer
L        D [AR1,P#2.0];      //变量的 Pointer 的偏移部分
LAR1 ;
END_FUNCTION
```

14.19.6 FB 调用

功能

针对所有含大量静态参数的功能块调用，例如基本程序提供的 FB2、3、4、5 和 7，为了优化其运行速度，可在启动中即通过相应的实例参数对其进行调用。 此时参数在启动（OB100）中进行预设，且之后在循环部分（OB1）不再进行修改。 这些固定的参数值在循环调用中不再进行设置，因为其已在启动中写入实例 DB。

示例： 带有实例 DB110 的 FB2 的参数设置

下面的示例将显示如何合理划分 OB100 部分和 OB1 部分。

首先显示的是循环程序中的普通调用：

```
CALL FB2, DB110(
     Req :=      M 100.0,
     NumVar :=   2,                    //读取 2 个变量
     Addr1:=     NCVAR.C1_RP_rpa0_0
     Line1 :     W#16#1
     Addr2:=     NCVAR.C1_RP_rpa0_0
     Line2 .     W#16#2
     Error :=    M1.0,
     NDR :=      M1.1,
     State :=    MW 2,
     RD1 :=      P#M 4.0 REAL 1,
     RD2 :=      P#M 24.0 REAL 1,
```

从此处起是经过修改的程序调用方案。
这里显示的是 OB100 中的调用：

```
CALL FB2, DB110(
     Req :=      FALSE,
     NumVar :=   2,                    //读取 2 个变量
     Addr1:=     NCVAR.C1_RP_rpa0_0
     Line1 :     W#16#1
     Addr2:=     NCVAR.C1_RP_rpa0_0
     Line2 .     W#16#2
     RD1 :=      P#M 4.0 REAL 1,
     RD2 :=      P#M 24.0 REAL 1,
```

这里显示 OB1 中还余下的调用：

```
CALL FB2, DB110(
     Req :=      M0.0,
     Error :=    M1.0,
     NDR :=      M1.1,
     State :=    MW 2,
```

说明

此措施能够缩短 OB1 中的运行时间，因为静态参数值不会在每个 OB1 运行中复制到实例 DB。

此方案的节约效果：

循环复制花费为 3 个整数值和 4 个基于实例 DB 的 ANY 参数，这由 3 次载入一个常量与 3 次传输至实例数据块得出。 在每次 ANY 传输中载入 4 次常量，之后传输至数据块。

14.20 数据表

14.20.1 机床数据

14.20.1.1 显示机床数据

编号	名称：$MM_	说明
9032	HMI_MONITOR	确定用于 HMI 屏幕信息的 PLC 数据

14.20.1.2 NC 专用机床数据

编号	名称： $MN_	说明
10100	PLC_CYCLIC_TIMEOUT	循环 PLC 监控时间
14504	MAXNUM_USER_DATA_INT	用户数据（INT）的数量
14506	MAXNUM_USER_DATA_HEX	用户数据（HEX）的数量
14508	MAXNUM_USER_DATA_FLOAT	用户数据（FLOAT）的数量
14510	USER_DATA_INT	用户数据（INT）
14512	USER_DATA_HEX	用户数据（HEX）
14514	USER_DATA_FLOAT[n]	用户数据（FLOAT）

说明

整数/十六进制格式的机床数据在 NC 中作为 DWORD 处理。 浮点格式的机床数据在 NC 中作为 FLOAT（8 Byte IEEE）处理。 其分别保存在 NC/PLC 接口中，在 PLC 启动中即可由 PLC 用户程序从 DB20 读取。

14.20.1.3 通道专用机床数据

编号	名称：$MC_	说明
28150	MM_NUM_VDIVAR_ELEMENTS	用于写入 PLC 变量的单元数

14.20.2 信号

14.20.2.1 来自操作面板的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
生效的 SINUMERIK 操作区	DB19.DB21	DB1900.DBB1
当前屏幕窗口号	DB19.DBW24	DB1900.DBW4

P4：SINUMERIK 828D 的 PLC 15

15.1 概述

15.1.1 PLC 固件

SINUMERIK 828D 的 PLC 是基于 SIMATIC S7-200 的指令组的集成 PLC。

PLC 用户程序的编程主要在 Windows-PC 上通过“PLC 编程工具”进行。 此外也可通过控制系统的操作界面对 PLC 进行诊断和编辑。 为此控制系统上提供了“Ladder-Add-On-Tool”。

请注意下列边界条件：

- PLC 用户程序的编程完全采用梯形图（LAD）。
- 支持 S7-200 编程语言的一个子集。
- 在载入至 CPU 时，除了运行相关的代码外，完整的项目数据（包括符号和注释）也会载入至控制系统。 这样一来控制系统总是能获取与当前运行的 PLC 用户程序相匹配的项目。
- 在从 CPU 载入时，完整的项目数据（包括符号和注释）会被载入 PLC 编程工具，并可由其处理。
- 用户必须根据类型管理数据和过程信息。 所有数据访问中都须持续使用设定的数据类型。

15.1.2 PLC 用户接口

用户接口由 PLC 固件创建，用于对 PLC 和 NC/HMI 之间的所有信号和数据交换进行组织。

用户接口由以下部分组成：

- 数据接口，用于循环交换（参见“数据接口 (页 1250)”）
- 功能接口，用于功能或位置相关的数据交换（参见“功能接口 (页 1268)”）

这些接口的结构化数据（掉电保持和非掉电保持）由固件指定给数据块从而供用户使用：PLC 用户程序的“通讯对象”是 NC（NCK、刀具管理、NC 通道、轴、主轴等）和 HMI。

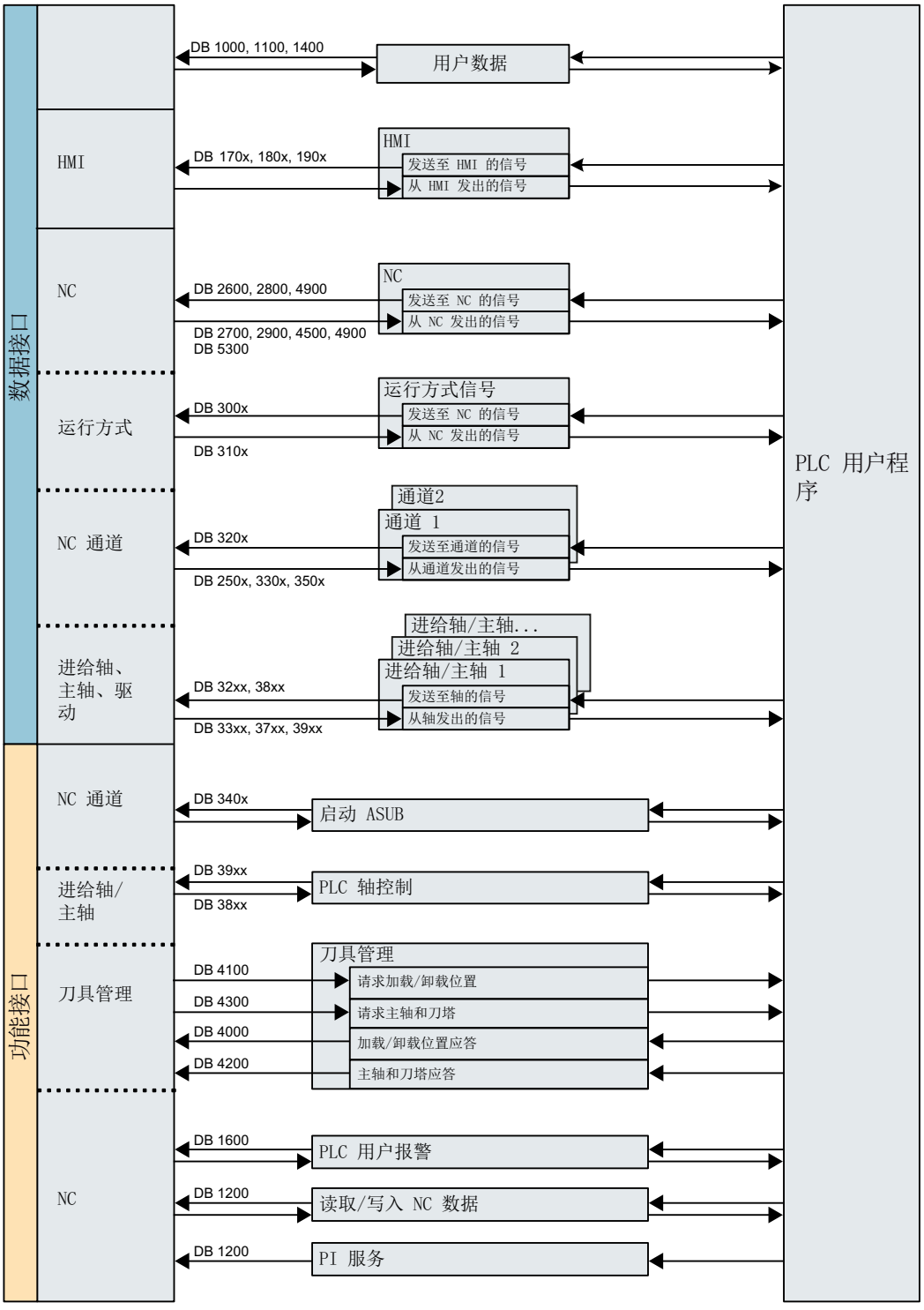


图 15-1　　PLC 828D 用户接口一览

15.1.2.1 循环交换的数据

数据一方面会在 PLC 和 NC 间，另一方面会在 PLC 和 HMI 间进行交换。

发送至 PLC 的数据由固件在用户程序的循环开始处提供。这样例如可确保 NC 发出的信号在一整个循环内保持恒定。

PLC **发出**的数据则由固件在用户程序的循环结束时传输至 NC 或 HMI。

PLC ↔ NC 接口

循环数据例如包括状态信号（“程序运行中”、“程序中断”）、控制信号（启动、停止）、辅助指令和 G 指令。

这些数据划分为用于以下功能的信号：

- 运行方式
- 通道
- 轴/主轴
- 通用 NC 信号

PLC ↔ HMI 接口

涉及的信号包括：

- 通过列表选择程序
- 控制指令 Messenger
- 来自/发送至 HMI 的通用信号
- 来自/发送至维护管理器的信号
- 操作面板信号（掉电保持区域）
- 来自/发送至 HMI 的选择/状态信号（掉电保持区域）

15.1.2.2 报警和消息

借助 DB1600 中的用户接口，可在 HMI 上显示故障消息和运行消息。

固件会对输入的信号进行分析，并将其作为“出现”和“消失”的报警和消息发送至 HMI 进行显示。 故障文本由 HMI 管理。

说明

HMI 上最多可显示八个 PLC 报警。

15.1.2.3 掉电保持数据

用户数据块 DB9000 - DB9063 和数据区域 DB1400.DBW0 - DBW127 用于掉电保持数据。 需要在断电/上电后保持生效的所有数据均可保存在该区域。 掉电保持数据会保存在非易失性存储器中，但不在数据备份中。

15.1.2.4 非掉电保持数据

非掉电保持数据（例如标志位、计时器和计数器）会在每次控制系统启动时被删除。

15.1.2.5 PLC 机床数据

PLC 机床数据位于 NC 机床数据的数据区域。这些数据会在上电时由 PLC 固件传输至 PLC 用户接口的 DB4500，并在该数据块中由 PLC 用户程序进行分析。

文档

SINUMERIK 828D 参数手册

15.1.3 PLC 基础数据

集成 PLC 通过一个程序存储器提供 24000 个 PLC 指令，这些指令都完全在一个固定的 PLC 循环中执行。

在可选择使用的中断程序 INT0 中可执行最多 500 个指令。指令同步于伺服执行，能够实现对过程事件的最快响应。因此不需要具有中断能力的 PLC I/O 模块。

数据	数量	特殊性
主程序（MAIN）	1	
子程序（SBRx）	256	
中断	2	
时间控制的中断	1	伺服同步的中断程序
报警/消息	1248	显示最多 8 个报警。

数据		数量	特殊性
标志位		4096	非掉电保持。
计数器		64	非掉电保持。
计时器， 其中：		128	非掉电保持。
	10 ms 增量间隔	112	
	100 ms 增量间隔	16	
用户模块，最大 512 字节		64	地址区域 DB9000 至 DB9063
NC ↔ PLC 数据传输			通过可固定参数设置的接口。

15.1.4 PLC I/O、快速板载输入/输出

快速板载输入/输出的属性及其响应时间请参见“快速板载输入/输出 (页 1246)”章节。

对 I/O 模块、机床控制面板，以及向 PLC 指定板载输入/输出的说明请参见：

文档：

SINUMERIK 828D PPU 设备手册

15.1.5 PLC Toolbox

15.1.5.1 星形/三角形切换

PLC Toolbox 中随附了以下用于星形/三角形切换的模块：

- StarDelta

说明

借助此模块，可通过配备 SMI 的主轴电机 1PH8 在 SINAMICS S120 上执行星形/三角形切换。

15.2 快速板载输入/输出

PPU 模块上配备了多个直接连接信号的接口，用于需要通过控制系统快速处理的数字量输入和输出信号：

- 连接器 X242：4 个输入信号，4 个输出信号
- 连接器 X252：4 个输入信号，2 个输出信号

接口	I/O 信号		定址
X242	输入：	DIN1 ... DIN4	I256.0 ... I256.3
	输出：	DOUT1 ... DOUT4	Q256.0 ... Q256.3
X252	输入：	DIN9 ... DIN12	I256.4 ... I256.7
	输出：	DOUT9 ... DOUT10	Q256.4 ... Q256.5

响应时间

根据位置控制器周期、执行位置和使用的数字量输入/输出，得出的响应时间如下：

位置控制器周期		1.5 ms	3 ms
	处理方式：	响应时间 1)	
	• 循环运行（MAIN、OB1）中的子程序 SBRx • 数字量输入/输出： **PP 72/48 I/O 模块**	12.5 ms	14 ms
	• 中断程序 INT0（伺服同步） • 数字量输入/输出：**PPU 的板载 I/O**	3 ms	4.5 ms
1) 信号发送至输入端 → 在 PLC 程序中处理 → 信号发送至输出端			

读取/写入访问

输入/输出信号的读写直接在模块的输入/输出上进行。 为了实现尽可能快的访问（伺服周期同步），建议在中断程序 **INT0** 中使用直接操作指令：

指令	符号
直接常开触点	-| I |-
直接常闭触点	-| /I |-
直接设置位值	-(SI)-

指令	符号
直接复位位值	-(RI)-
直接指定位值	-(I)-

参数设置

输入/输出信号可通过以下机床数据单独指定给 NC 或 PLC：

- MD10366 $MN_HW_ASSIGN_DIG_FASTIN[<n>]
- MD10368 $MN_HW_ASSIGN_DIG_FASTOUT[<n>]

其中 **<n>**：用于输入/输出字节定址的下标（0 = 字节 1，1 = 字节 2，...）

I/O 信号		对应关系	
		NC	PLC
输入：	DIN1 ... DIN4	MD10366[0] = 00 01 **01** 01_H	MD10366[0] = 00 01 **00** 01_H
输出：	DOUT1 ... DOUT4	MD10368[0] = 00 01 **01** 01_H	MD10368[0] = 00 01 **00** 01_H
输入：	DIN9 ... DIN12	MD10366[1] = 00 01 **01** 01_H	MD10366[1] = 00 01 **00** 01_H
输出：	DOUT9 ... DOUT10	MD10368[1] = 00 01 **01** 01_H	MD10368[1] = 00 01 **00** 01_H

文档

对接口的详细说明参见：

设备手册 SINUMERK 828D PPU

15.3 梯形图浏览器和梯形图插件

15.3.1 一览

功能

梯形图浏览器和梯形图插件的主要功能：

- 梯形图浏览器：将 PLC 用户程序中包含的触点和继电器的连接以梯形图（KOP）的形式显示
- 梯形图插件：
  - 诊断 PLC 用户程序
  - 处理或创建中断程序
  - 数据重组

文档

对梯形图浏览器和梯形图插件的功能及操作的详细说明参见相应的 SINUMERIK 操作手册：

- 操作手册“通用”
- 操作手册“车削”
- 操作手册“铣削”

各自的“梯形图浏览器和梯形图插件”章节

15.3.2 参数设置

显示地址

通过以下 NC 专用机床数据依据 SIMATIC S7 300 标识设置梯形图浏览器中的地址显示：

MD51230 $MN_ENABLE_LADDER_DB_ADDRESSES = <值>

<值>	含义
0	依据 SIMATIC S7 200 标识的地址显示（例如 Vxxxxx）
1	依据 SIMATIC S7 300 标识的地址显示（例如 DBxx.DBBxxxx）

使能/禁止编辑

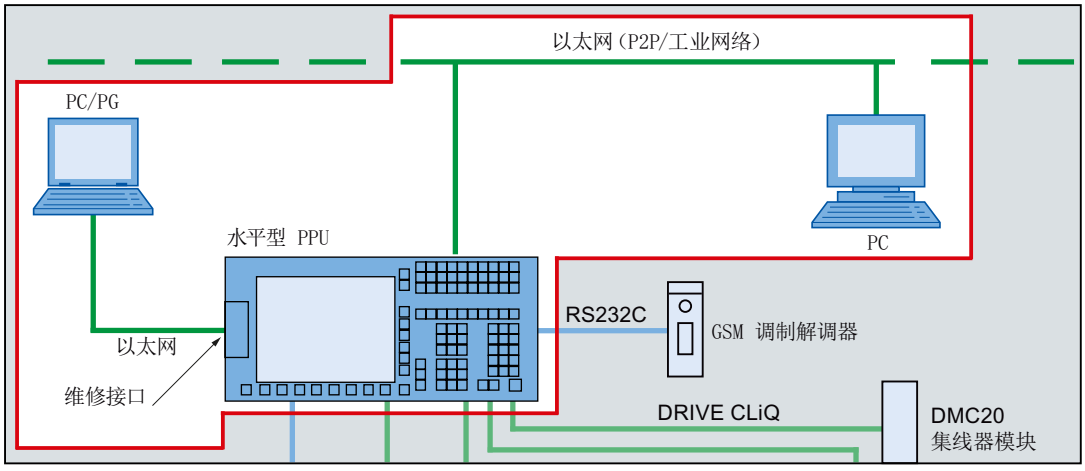
通过以下 NC 专用机床数据可使能或禁止对中断程序 INT_100 和 INT_101 的编辑：

MD51231 $MN_ENABLE_LADDER_EDITOR = <值>

<值>	含义
0	中断程序 INT_100 和 INT_101 无法编辑。
1	中断程序 INT_100 和 INT_101 可编辑。

15.4 PLC 编程工具

使用“PLC 编程工具”可方便地创建 PLC 用户程序。编程工具是一个 Windows 程序，必须安装在 Windows-PC 上。 为了能在线访问控制系统，必须通过工业以太网接口、X130（工业网络）或 X127（维护接口）将 PC 与控制系统相连。



调用“PLC 编程工具”时，若未指定现有项目，则会创建一个名为“项目 1”的新项目。 这个项目可立即用于创建 PLC 用户程序，以任意名称另存和载入控制系统。 编程工具中现有的项目可以按照 Windows 程序的典型方式打开。

“PLC 编程工具”的在线帮助可通过 Windows 下的典型方式以功能键“F1”调用。

文档

- SINUMERIK 828D 调试手册，车削和铣削分册
  章节“供货范围和前提条件”>“与控制系统通讯”>“通过编程工具与控制系统通讯”
- PLC 编程工具的在线帮助

15.5 数据接口

数据一方面会在 PLC 和 NC 间，另一方面会在 PLC 和 HMI 间循环交换。这意味着必须通过用户程序对来自 HMI 的数据和为 NC 确定的数据进行安排，从而使其生效。

发送至 PLC 的数据由固件在用户程序的循环开始处提供。这样例如可确保 NC 发出的信号在一整个循环内保持恒定。

PLC 发出的数据则由固件在用户程序的循环结束时传输至 NC 或 HMI。

对此接口的所有数据的说明参见 SINUMERIK 828D PPU 设备手册。

15.5.1 PLC-NC 接口

循环数据例如包括状态信号（“程序运行中”、“程序中断”）、控制信号（启动、停止）、辅助指令和 G 指令。

这些数据划分为用于以下功能的信号：

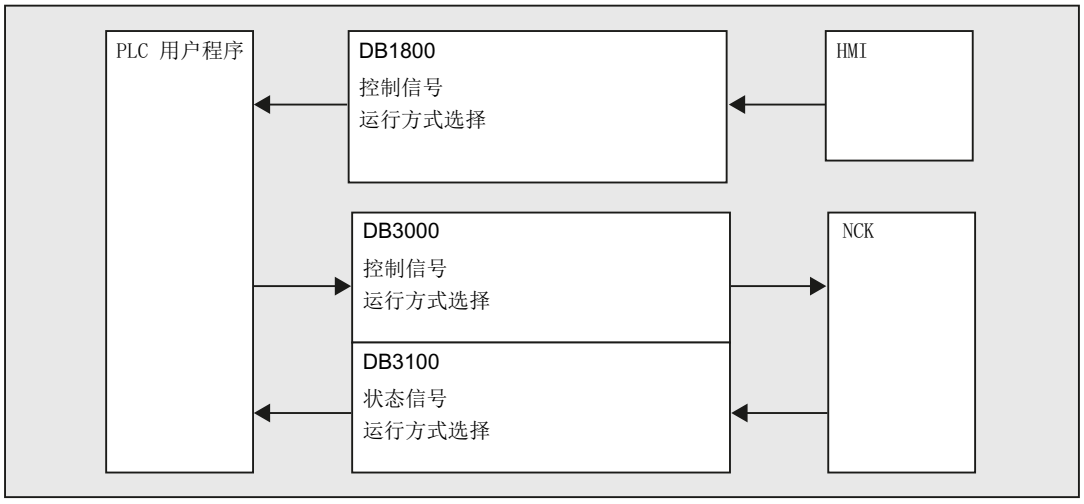
- 运行方式信号
- NC 通道信号
- 轴信号和主轴信号
- 通用 NC 信号
- PLC-NC 快速数据交换

15.5.1.1 运行方式信号

DB3000, 3100

通过机床控制面板或 HMI 设定的运行方式信号会传输至 NC。

NC 会将其当前状态发送至 PLC。

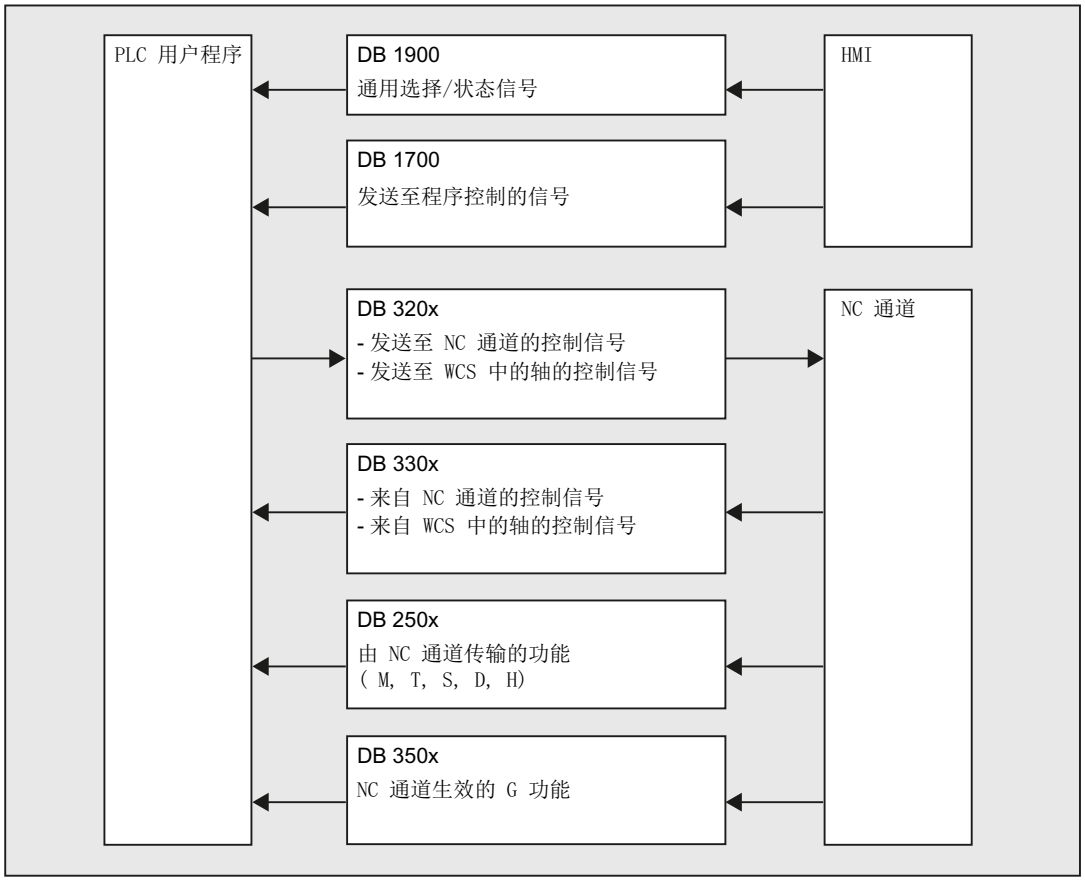

PLC 用户程序
DB1800
控制信号
运行方式选择
HMI
DB3000
控制信号
运行方式选择
NCK
DB3100
状态信号
运行方式选择

15.5.1.2 NC 通道信号

DB250x, 320x, 330x, 350x

此信号分为：

- 采用一般循环传输的控制信号/状态信号，参见“运行方式信号 (页 1250)”。
- 辅助指令和 G 指令
  这些信号通过两种方式传输至接口 DB。
  其首先会和修改信号一起被记录。
  M 信号 M0 至 M99 还会被解码，且循环对应的接口位会被设置。
  对于 G 功能，只有通过机床数据选择的功能组会被记录至接口数据块。
  S 值还会与对应的 M 信号（M03、M04、M05）一同记录至主轴专用接口。同样地，轴专用进给率也会记录至相应的轴专用接口。



15.5.1.3 轴信号和主轴信号

DB370x, 380x, 390x

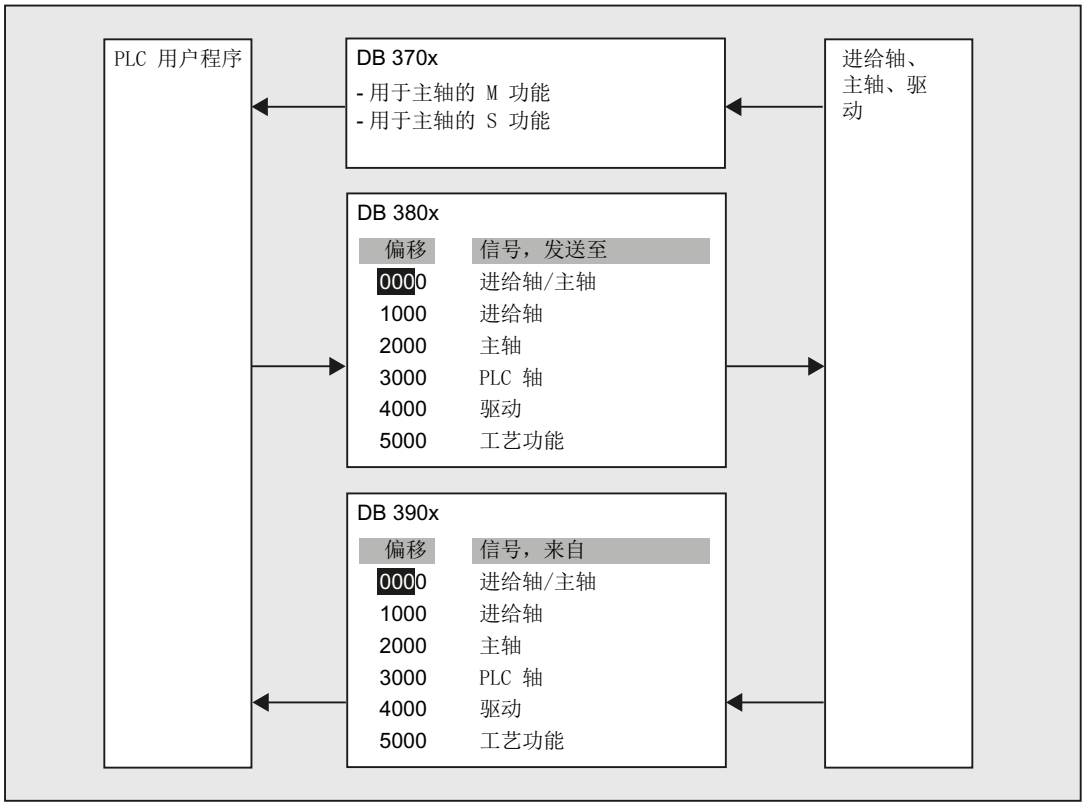
轴/主轴专用信号分为下列组别：

- 轴/主轴共用信号
- 轴信号
- 主轴信号
- 驱动信号

除了下面列出的特例外，这些信号均为循环传输。特例包括轴向 F 值、M 值和 S 值。

若轴向 F 值在 NC 程序执行过程中传输至 PLC，则会通过 M、S、F 分配器记录。

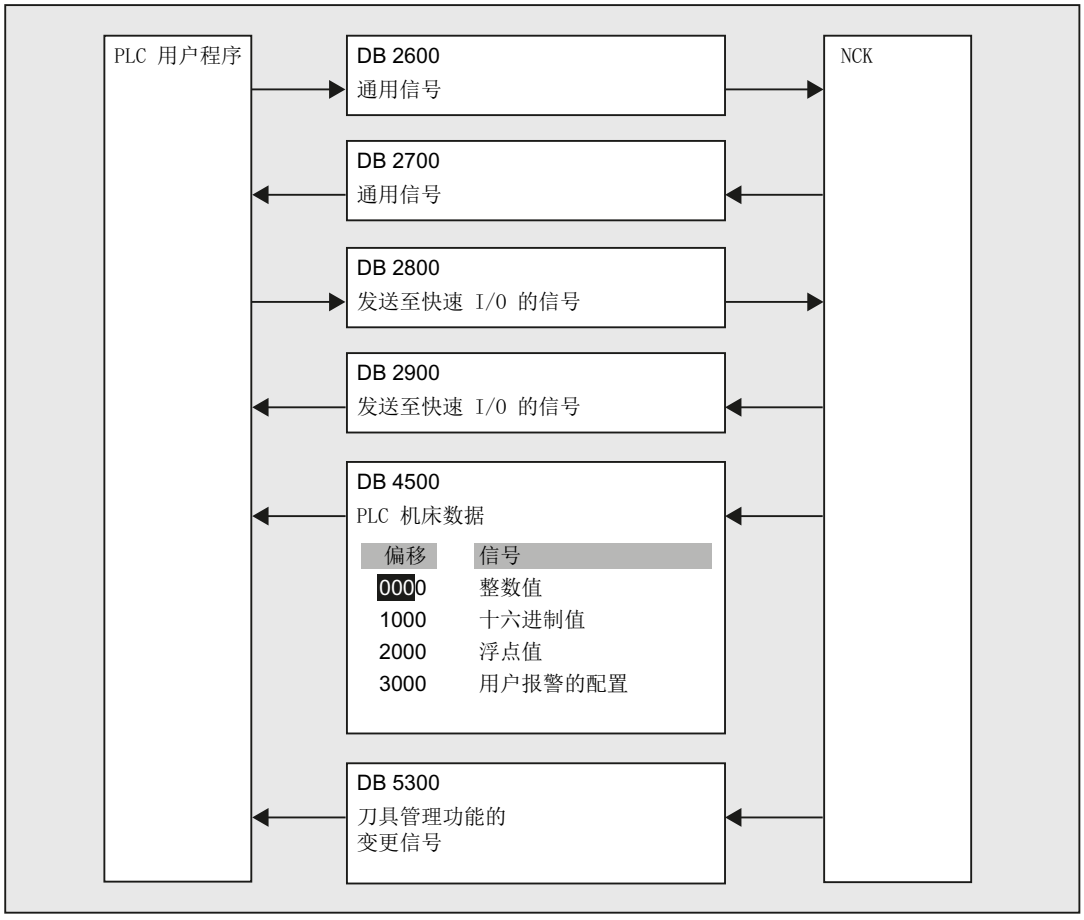
对于 M 值和 S 值，若其中一个值或两个值被执行，则同样通过 M、S、F 分配器记录。



15.5.1.4 通用 NC 信号

DB2600, 2700, 2800, 2900, 4500, 5300

- 发送至 NC 数字量和模拟量输入/输出的设定值
- 从 NC 数字量和模拟量输入/输出发出的实际值
- 钥匙开关信号和急停信号
- NC 的就绪信号和状态信号



15.5.1.5 PLC-NC 快速数据交换

DB4900

数据块 DB4900 设计用于 PLC 和 NC 间的快速信息交换，其大小为 1024 字节。

此区域（结构）在 NC 零件程序和 PLC 用户程序中必须启用相同定义。

NC 零件程序可通过 $A_DBB[x]、$A_DBW[x]、$A_DBD[x]、$A_DBR[x]（0 ≤ x ≤ 1023） 指令访问这些数据（参见：参数手册，系统变量）。

此时必须根据数据格式对齐数据，即一个 Dword 要占用 4 字节区间，一个 Word 则占用 2 字节。字节可处于数据区域内的任意位移处。

系统可保证字节、字和双字访问的数据一致性。在传输多个数据时，用户须启用信号量（用于识别数据块的有效性或一致性）来确保数据一致性。

15.5.2 PLC-HMI 接口

DB1700, 1800, 1900

这些信号已在 PLC-NC 接口 (页 1250) 章节中的图中注明。

这里再次引用 数据接口 (页 1250) 中的说明：

来自 HMI 的数据和为 NC 确定的数据不会自动传输至 NC 接口区域。这些信号和数据须通过用户程序安排。

涉及的信号包括：

- 通过列表选择程序
- 控制指令 Messenger
- 来自/发送至 HMI 的通用信号
- 来自/发送至维护管理器的信号
- 操作面板信号（掉电保持区域）
- 来自/发送至 HMI 的选择/状态信号（掉电保持区域）

15.5.2.1 操作区域编号

生效操作区域的编号显示在：DB1900.DBB1

操作区域编号

操作区	编号
加工	201
参数	205

操作区	编号
编程	203
程序管理器	202
诊断	204
调试	206

15.5.2.2 屏幕窗口号

当前屏幕窗口号显示在：DB1900.DBW4

屏幕窗口号区域

屏幕窗口号区域有：

- JOG、手动加工 (页 1256)
- 回参考点 (页 1263)
- MDA (页 1263)
- AUTO (页 1264)
- “参数”操作区 (页 1265)
- “程序”操作区 (页 1266)
- “程序管理器”操作区 (页 1267)
- “诊断”操作区 (页 1267)

屏幕窗口号：JOG、手动加工

运行方式 JOG

画面	编号
车削工艺	
用于所有可接收窗口的循环启动窗口	81
铣削工艺	
用于所有可接收窗口的循环启动窗口	3
车削/铣削工艺	
基本画面	19

画面	编号
T,S,M	2
设置零偏	21
定位	4
平面铣削	18
切削	80
用于所有用户屏幕窗口的循环启动窗口	91
一般设置	1
多通道功能设置	106
碰撞监测设置	107
测量记录设置	108
回转	60
所有 G 指令	100
实际值缩放 (MCS/WCS)	101
螺纹同步	102
退回	103
手轮	104
同步动作	105
车削工艺：工件零点	
工件零点（基本菜单）	30
用户屏幕窗口	31
用户屏幕窗口	34
用户屏幕窗口	35
用户屏幕窗口	36
用户屏幕窗口	37
用户屏幕窗口	38
用户屏幕窗口	40
测量边沿 Z	5
车削工艺：工件测量	
测量刀具（基本菜单）	50
手动 X 或用户屏幕窗口	51
手动 Y	71

画面	编号
手动 Z 或用户屏幕窗口	52
放大镜或用户屏幕窗口	53
用户屏幕窗口	54
用户屏幕窗口	55
校准测头 X 或用户屏幕窗口	56
校准测头 Z 或用户屏幕窗口	57
Z 中的长度自动设定	58
Y 中的长度自动设定	73
X 中的长度自动设定	59
铣削工艺：工件零点	
工件零点（基本菜单）	30
测量边沿 X	5
测量边沿 Y	22
测量边沿 Z	23
用户屏幕窗口	7
边对齐或用户屏幕窗口	31
2 个边沿的间距或用户屏幕窗口	32
直角	33
任意角或用户屏幕窗口	8
矩形腔	34
1 个钻孔或用户屏幕窗口	9
2 个孔	35
3 个孔	36
4 个孔	37
矩形凸台	38
1 个圆弧凸台或用户屏幕窗口	10
2 个圆弧凸台	39
3 个圆弧凸台	40
4 个圆弧凸台	41
校准平面	42
校准探头长度或用户屏幕窗口	11

画面	编号
校准探头半径	12
铣削工艺：测量工件	
测量刀具（基本菜单）	50
- 手动测量长度（用铣刀）或 - 手动测量 X 轴长度（用车刀）或 - 用户屏幕窗口	16
手动测量 Y 轴长度（用车刀）	74
手动测量 Z 轴长度（用车刀）	24
手动测量直径或用户屏幕窗口	17
- 自动测量长度（用铣刀）或 - 自动测量 X 轴长度（用车刀）或 - 用户屏幕窗口	13
自动测量 Y 轴长度（用车刀）	75
自动测量 Z 轴长度（用车刀）	25
自动测量直径或用户屏幕窗口	14
用户屏幕窗口	51
校准测头或用户屏幕窗口	15
校准固定点或用户屏幕窗口	52
RunMyScreens（仅当 JobShopIntegration 置位时）	
用于第 1 个水平软键的用户屏幕窗口	96
用于第 2 个水平软键的用户屏幕窗口	98
用于第 3 个水平软键的用户屏幕窗口	99
用于第 4 个水平软键的用户屏幕窗口	94
用于第 5 个水平软键的用户屏幕窗口	95
用于第 6 个水平软键的用户屏幕窗口	92
用于第 7 个水平软键的用户屏幕窗口	97
用于第 8 个水平软键的用户屏幕窗口	90
用于第 9 个水平软键的用户屏幕窗口	83
用于第 10 个水平软键的用户屏幕窗口	82
用于第 11 个水平软键的用户屏幕窗口	93
用于第 12 个水平软键的用户屏幕窗口	84
用于第 13 个水平软键的用户屏幕窗口	85

画面	编号
用于第 14 个水平软键的用户屏幕窗口	86
用于第 15 个水平软键的用户屏幕窗口	87
用于第 16 个水平软键的用户屏幕窗口	88

JOG、手动加工

DB19.DBB24	
画面	**画面编号**
车削/铣削工艺	
车削-锥	61
铣削-角	62
挡块	63
直线	1300
所有轴直线	1330
X alpha 直线	1340
Z alpha 直线	1350
圆弧	1360
钻削	1400
钻削-中心钻孔	1410
钻削-螺纹中心	1420
钻削-钻中心孔	1433
钻削-钻削	1434
钻削-铰孔	1435
钻削-镗孔	1436
钻削-深孔钻削	1440
钻削-深孔钻削 2	1441
钻削 – 螺纹钻削	1453
钻孔螺纹铣削	1455
位置	1473
序列位置	1474
栅格位置	1477
框架位置	1478

DB19.DBB24	
画面	**画面编号**
圆弧位置	1475
节距圆位置	1479
障碍物	1476
车削	1500
车削-切削 1	1513
车削-切削 2	1514
车削-切削 3	1515
车削-凹槽 1	1523
车削-凹槽 2	1524
车削-凹槽 3	1525
车削-E 形退刀槽	1533
车削-F 形退刀槽	1534
车削-螺纹 DIN 退刀槽	1535
车削-螺纹退刀槽	1536
车削-螺纹长度	1543
车削-螺纹锥	1544
车削-螺纹端面	1545
车削-螺纹链	1546
车削-断削	1550
铣削	1600
铣削-平面铣削	1610
铣削-矩形腔	1613
铣削-圆形腔	1614
铣削-矩形轴颈	1623
铣削-圆形轴颈	1624
铣削-长形腔	1633
铣削-圆形槽	1634
铣削-开口槽	1635
铣削-多边形	1640
铣削-螺纹铣削	1454

DB19.DBB24	
画面	**画面编号**
铣削-雕刻	1670
轮廓车削	1200
轮廓车削-新轮廓/上一个轮廓	1210
轮廓车削-轮廓切削	1220
轮廓车削-轮廓槽式车削	1230
轮廓车削-轮廓槽式车削	1240
轮廓铣削	1100
轮廓铣削-新轮廓/上一个轮廓	1110
轮廓铣削-轨迹铣削	1120
轮廓铣削-钻中心孔	1130
轮廓铣削-预钻削	1140
轮廓铣削-轮廓腔	1150
车削工艺：模拟	
侧视图	1740
正视图	1750
3D 视图	1760
2 窗口视图	1770
半剖视图	1780
车削工艺：同步记录	
侧视图	1741
正视图	1751
3D 视图	1761
2 窗口视图	1771
机床内部	1791
半剖视图	1781
铣削工艺：模拟	
顶视图	1742
3D 视图	1760
前视图	1744
后视图	1746

DB19.DBB24	
画面	**画面编号**
左视图	1748
右视图	1752
半剖视图	1780
车削视图	1782
铣削工艺：同步记录	
顶视图	1743
3D 视图	1761
前视图	1745
后视图	1747
左视图	1749
右视图	1753
机床内部	1791
半剖视图	1781
车削视图	1783

屏幕窗口号：回参考点

画面	**编号**
实际值缩放 MCS/WCS	101

屏幕窗口号：MDI

画面	**编号**
MDI	20
所有 G 指令	100
实际值缩放 MCS/WCS	101
手轮	104
同步动作	105
程序控制	210
设置	250

屏幕窗口号：AUTO

画面	编号
自动方式	200
溢出转存	202
程序控制	210
程序段搜索	220
一般设置	250
多通道功能设置	106
碰撞监测设置	107
所有 G 指令	100
实际值缩放 MCS/WCS	101
手轮	104
同步动作	105
车削工艺：同步记录	
侧视图	243
正视图	244
3D 视图	245
2 窗口视图	246
机床内部	247
半剖视图	253
铣削工艺：同步记录	
顶视图	242
3D 视图	244
前视图	248
后视图	249
左视图	251
右视图	252
机床内部	247
半剖视图	253
车削视图	254

屏幕窗口号：“参数”操作区

画面	编号
刀具表	600
刀具磨损	610
用户刀具列表	620
刀库	630
零点偏移	
零点偏移生效	642
零点偏移概述	643
基本零偏	644
零点偏移 G54 - G509	645
零点偏移生效、零点偏移概述、基本零偏或 G54 - G509 的详细信息	647
用户变量	
R 参数	650
全局 GUD 1 (SGUD)	660
全局 GUD 2 (MGUD)	661
全局 GUD 3 (UGUD)	662
全局 GUD 4	663
全局 GUD 5	664
全局 GUD 6	665
全局 GUD 7	666
全局 GUD 8	667
全局 GUD 9	668
通道 GUD 1 (SGUD)	690
通道 GUD 2 (MGUD)	691
通道 GUD 3 (UGUD)	692
通道 GUD 4	693
通道 GUD 5	694
通道 GUD 6	695
通道 GUD 7	696
通道 GUD 8	697
通道 GUD 9	698

画面	编号
局部 LUD	681
局部 LUD / PUD	684
设定数据	
工作区域限制	671
主轴数据	670
主轴卡盘数据	672
Ctrl-Energy	
Ctrl-Energy 基本菜单	6170
Ctrl-Energy 分析	6171
Ctrl-Energy 协议	6172
Ctrl-Energy 分析图形	6176
Ctrl-Energy 分析长期测量	6177
Ctrl-Energy 分析详细信息	6179
Ctrl-Energy 测量比较	6178

屏幕窗口号：“程序”操作区

画面	编号
车削工艺：模拟	
侧视图	413
正视图	414
3D 视图	415
2 窗口视图	416
半剖视图	423
铣削工艺：模拟	
顶视图	412
3D 视图	414
前视图	418
后视图	419
左视图	421
右视图	422

画面	编号
半剖视图	423
车削视图	424

屏幕窗口号：“程序管理器”操作区

画面	编号
NC 目录	300
本地驱动器	325
USB / 已设计的驱动器 1	330
已设计的驱动器 2	340
已设计的驱动器 3	350
已设计的驱动器 4	360
已设计的驱动器 5	383
已设计的驱动器 6	384
已设计的驱动器 7	385
已设计的驱动器 8	386

屏幕窗口号：“诊断”操作区

画面	编号
报警列表	500
信息	501
报警日志	502
NC/PLC 变量	503

15.6 功能接口

15.6.1 读写 NC 变量

15.6.1.1 用户接口

借助 NC/PLC 接口“读取/写入 NC 变量”可同时读取或写入最多八个 NC 变量。

任务（读取/写入）中须执行以下步骤：

1. 任务设定 (页 1268)
2. 任务管理器：启动任务 (页 1270)
3. 任务管理器：等待任务结束 (页 1270)
4. 任务管理：任务结束 (页 1271)
5. 任务分析 (页 1273)

任务的流程图参见“任务管理：流程图 (页 1272)”

15.6.1.2 任务设定

变量专用任务接口

需要在任务中处理的每个变量都必须在**变量专用任务接口**中通过其参数设定。通用名称会于随后针对每个可由接口操作的变量详细说明。

DB120x[1)]	读取/写入 NC 数据（PLC → NC）							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBB1000	变量索引							
DBB1001	区域编号							
DBW1002	NC 变量 x 的行下标							
DBW1004	NC 变量 x 的列下标							
DBW1006	---							
DBD1008	写入：NC 变量 x 的数据（变量的数据类型：1…4 字节）[2)]							
DBD1012	---							
DBD1016	写入：NC 变量 x 的数据（REAL）[3)]							
DBD1020	写入：NC 变量 x 的数据（DWORD / DINT）[3)]							
DBW1024	写入：NC 变量 x 的数据（WORD / INT）[3)]							

DB120x [1)]	读取/写入 NC 数据（PLC → NC）							
DBB1026	写入：NC 变量 x 的数据（BYTE）[3)]							
DBB1027	---	---	---	---	---	---	---	写入：NC 变量 x 的数据 [3)]

1) DB120x，其中 x = 0 ... 7，对应 1 ... 8。

2）只针对用户接口“读/写 NC 变量”的预定义变量

3) 只针对 DB9910 NC_DATA 中的变量

说明

通道专用变量

写入/读取通道专用变量时，一个任务中只允许通过**一个**通道进行变量定址。

驱动专用变量

写入/读取驱动专用变量时，一个任务中只允许通过 **一个**驱动对象 SERVO 进行变量定址。驱动对象 SERVO 必须已指定给 NC 的一根机床轴。行索引对应逻辑驱动编号。

故障情况

从不同的驱动对象读写变量，或同时从一个通道和一个驱动对象读写变量时，会显示以下故障信息：

DB1200.DBX3000.1 == 1（出现故障）

示例：将一个“刀位类型”变量作为第四个变量读取

```
DB120x.DBB1000: 77
DB120x.DBB1001: -
DB120x.DBW1002: <刀位编号>
DB120x.DBW1004: <刀库编号>
DB120x.DBW1006: -
DB120x.DBD1008: -
```

示例：将一个变量作为第四个变量写入

为了将一个数据写入 NC，必须将数值输入至双字 DBD1008 中：

```
DB120x.DBB1000: <变量索引>
DB120x.DBB1001: <区域编号>
DB120x.DBW1002: <列索引>
DB120x.DBW1004: <行索引>
DB120x.DBW1006: -
DB120x.DBD1008: <值>
```

15.6.1.3 任务管理器：启动任务

以下数据须由用户写入**全局用户接口**：

DB120x[1)]	**读取/写入 NC 数据（PLC → NC）**							
字节	**位 7**	**位 6**	**位 5**	**位 4**	**位 3**	**位 2**	**位 1**	**位 0**
DBB0							任务类型	任务：启动
DBB1	任务中须处理的变量的数量							
1) DB120**x**，其中 x = 0 ... 7，对应 1 ... 8。								

任务类型

- **读取**变量：DB1200.DBX0.1 = **0**
- **写入**变量：DB1200.DBX0.1 = **1**

启动任务

为了通过指定数量的变量启动任务，必须设置启动信号：

DB1200.DBX0.0 = **1**

说明

之前的任务已完成时，才能启动新任务。参见“任务管理器：等待任务结束 (页 1270)”章节。

任务执行可能会持续多个 PLC 循环，并根据系统负载率有所浮动。因此此功能的时间无法确定。

15.6.1.4 任务管理器：等待任务结束

任务结束由 NC 在**全局结果接口**中始终针对**整个**任务反馈。对于 PLC 用户，信号只能读取。

DB120x[1)]	**读取/写入 NC 数据（NC → PLC）**							
字节	**位 7**	**位 6**	**位 5**	**位 4**	**位 3**	**位 2**	**位 1**	**位 0**
DBB2000							任务中出错	任务结束
1) DB120**x**，其中 x = 0 ... 7，对应 1 ... 8。								

任务状态

- 任务无故障结束
  DB1200.DBX2000.0 == 1 AND DB1200.DBX2000.1 == 0
- 任务报错结束
  DB1200.DBX2000.0 == 1 AND DB1200.DBX2000.1 == 1

可能的故障原因

- 变量数（DB1200.DBB1）超出有效范围
- 变量索引（DB1200.DBB1000）超出有效范围
- 同时从不同的驱动对象 Servo 读取/写入 NC 数据

15.6.1.5 任务管理：任务结束

要求

为了结束任务，必须通过 PLC 用户程序在识别任务结束后将任务的启动信号复位：

DB1200.DBX0.0 = 0

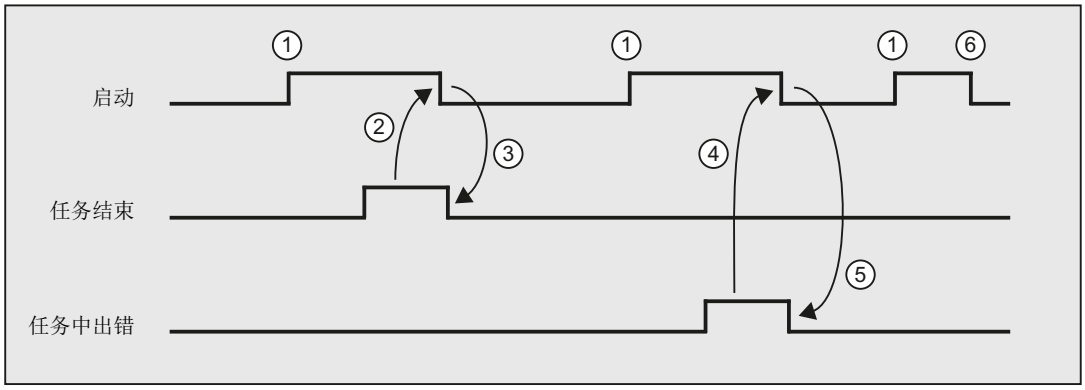
反馈信息

NC 会复位状态信号作为反馈信息：

- DB1200.DBX2000.0 == 0
- DB1200.DBX2000.1 == 0

至此任务结束。

15.6.1.6 任务管理：流程图



① 启动任务：
DB1200.DBX0.0（启动）= 1

② 等待任务结束：
DB1200.DBX2000.0（任务结束）== **1** AND
DB1200.DBX2000.1（任务中出错）== **0**
⇒ 复位任务请求：
DB1200.DBX0.0 = 0（启动）

③ 通过 DB1200.DBX0.0 == 0（启动）由 PLC 基本程序结束任务：
DB1200.DBX2000.0（任务结束）= 0

④ 等待任务结束：
DB1200.DBX2000.0（任务结束）== **0** AND
DB1200.DBX2000.1（任务中出错）== **1**
⇒ 执行故障处理
⇒ 复位任务请求：
DB1200.DBX0.0（启动）= 0

⑤ 通过 DB1200.DBX0.0 == 0（启动）由 PLC 基本程序结束任务：
DB1200.DBX2000.1（任务中出错）= 0

⑥ PLC 基本程序发出任务结束信息前，若 DB1200.DBX0.0（启动） 被复位，任务会被执行，但是无反馈信息。

15.6.1.7 任务分析

必须为任务中处理的每个变量分析变量专用结果接口。

DB120x [1)]	NC 服务（NC → PLC）							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBB3000							出错	变量有效
DBB3001	访问结果（参见以下“访问结果”部分）							
DBW3002	---							
DBD3004	读取：NC 变量 x 的数据（变量的数据类型：1…4 字节）[2)]							
DBD3008	---							
DBD3012	---							
DBD3016	读取：NC 变量 x 的数据（REAL）[3]							
DBD3020	读取：NC 变量 x 的数据（DWORD / DINT）[3]							
DBW3024	读取：NCK 变量 x 的数据（WORD / INT）[3]							
DBB3026	读取：NCK 变量 x 的数据（BYTE）[3]							
DBB3027	---	---	---	---	---	---	---	读取：NC 变量 x 的数据 [3)]

1) DB120x，其中 x = 0 ... 7，对应 1 ... 8。

2）只针对用户接口“读/写 NC 变量”的预定义变量

3) 只针对 DB9910 NC_DATA 中的变量

说明

通道专用变量

写入/读取通道专用变量时，一个任务中只允许通过一个通道进行变量定址。

驱动专用变量

写入/读取驱动专用变量时，一个任务中只允许通过 一个驱动对象 SERVO 进行变量定址。驱动对象 SERVO 必须已指定给 NC 的一根机床轴。行索引对应逻辑驱动编号。

故障情况

从不同的驱动对象读写变量，或同时从一个通道和一个驱动对象读写变量时，会显示以下故障信息：

DB1200.DBX3000.1 == 1（出现故障）

访问结果

NC 变量

DBB3001	
值	含义
0	无故障
3	不允许访问对象
5	无效地址
10	对象不存在

驱动专用变量

如果在读取/写入驱动专用变量时出错 (DB1200.DBX3000.1 == 1)，访问结果中会显示一个故障编号（该编号基于在 PROFIdrive 协议中定义的故障编号）。

DBB3001	
值	含义
x	<PROFIdrive 协议的故障编号> + 20_H 或 36_D

确定访问结果的含义：

1. 计算 PROFIdrive 协议的故障编号
   <PROFIdrive 协议的故障编号> + 访问结果 - 20_H 或 36_D。
2. 确定 PROFIdrive 协议的故障编号的含义
   有关 PROFIdrive 协议的故障编号的说明请见：
   文档
   功能手册之驱动功能 SINAMICS S120；章节 “通讯” > “PROFIdrive 通讯” > “非循环通讯” > “任务和应答结构” > “参数应答中的故障值"

示例：任务状态

任务无故障

- DB1200.DBX3000.0 == 1（变量有效）**AND**
- DB1200.DBX3000.1 == 0（未出现故障）

结果：

- DB1200.DBB3001 == 0 （访问结果:“无故障”）
- DB1200.DBD3004 == <读取的值>

任务存在故障

- DB1200.DBX3000.0 == 0（变量无效）**AND**
- DB1200.DBX3000.1 == 1（出现故障）

结果：

- DB1200.DBB3001：可能的故障原因参见上面的“访问结果”部分

15.6.1.8 可操作变量

可供使用的变量如下：

变量	含义
cuttEdgeParam (页 1275)	刀具带 D 号的补偿值参数和刀沿列表
numCuttEdgeParams (页 1276)	刀沿 P 元素的数量
linShift (页 1276)	可设定零点偏移的平移
numMachAxes (页 1278)	现有通道轴的最高编号
rpa (页 1278)	R 参数
actLineNumber (页 1279)	当前 NC 程序段的行编号
$TC_MPPx (页 1279)	刀位数据
r0078[1] (页 1281)	转矩电流的实际值
r0079[1] (页 1281)	转速控制器输出处的转矩设定值
r0081 (页 1282)	转矩利用率，百分比值
r0082[1] (页 1282)	有功功率实际值
TEMP_COMP_x (页 1283)	温度补偿数据

变量“cuttEdgeParam”

刀具带 D 号的补偿值参数和刀沿列表

单个参数的含义取决于各个刀具的类型。目前为每个刀沿预留了 25 个参数（但只有一部分已赋值）。为将来可以灵活扩展，不使用 25 个参数中的一个固定值进行计算，而是使用变量值 'numCuttEdgeParams'（变量索引 2）进行计算。

刀具参数的详细说明请参见章节 W1：刀具补偿 (页 1593)。

地址	值 / 含义
DB120x.DBB1000	1
DB120x.DBB1001	-
DB120x.DBW1002	(刀沿编号 - 1) * numCuttEdgeParams + 参数编号 (WORD)
DB120x.DBW1004	T 号 (1...32000) (WORD)
DB120x.DBD1008	写入：数据至 NC 变量 x（数据类型：REAL)
DB120x.DBW3004	读取：NC 变量 x 的数据（数据类型：REAL)

变量“numCuttEdgeParams”

刀沿 P 元素的数量

地址	值 / 含义
DB120x.DBB1000	2
DB120x.DBB1001	-
DB120x.DBW1002	-
DB120x.DBW1004	-
DB120x.DBD1008	-
DB120x.DBW3004	读取：来自 NCK 变量 x 的数据（数据类型：WORD）

变量“linShift”

可设定零点偏移（通道专用可设定框架）的平移

只有当 MD18601 MM_NUM_GLOBAL_USER_FRAMES > 0 时，该变量才存在。

有下列框架索引可供使用：

索引	含义
0	ACTFRAME = 当前生成的零点偏移
1	IFRAME = 当前可设定的零点偏移
2	PFRAME = 当前可编程的零点偏移
3	EXTFRAME = 当前外部零点偏移
4	TOTFRAME = 当前总零点偏移 = 由 ACTFRAME 和 EXTFRAME 得出的总和
5	ACTBFRAME = 当前总基本框架
6	SETFRAME = 当前第 1 系统框架（实际值设置，对刀）
7	EXTSFRAME = 当前第 2 系统框架（实际值设置，对刀）
8	PARTFRAME = 当前第 3 系统框架（可定向刀架的 TCARR 和 PAROT）
9	TOOLFRAME = 当前第 4 系统框架（TOROT 和 TOFRAME）
10	MEASFRAME = 工件和刀具测量的结果框架
11	WPFRAME = 当前第 5 系统框架（工件基准点）
12	CYCFRAME = 当前第 6 系统框架（循环）

地址	值 / 含义
DB120x.DBB1000	3
DB120x.DBB1001	-
DB120x.DBW1002	框架索引 * numMachAxes (页 1278) + 轴号
DB120x.DBW1004	-
DB120x.DBD1008	-
DB120x.DBW3004	**读取**：NC 变量 x 的数据（数据类型：REAL)

变量“numMachAxes”

现有通道轴的最高编号

若无通道轴**空位**，那么此变量的值同时也是通道中现有轴的数量。

地址	值 / 含义
DB120x.DBB1000	4
DB120x.DBB1001	-
DB120x.DBW1002	-
DB120x.DBW1004	-
DB120x.DBD1008	-
DB120x.DBW3004	**读取**：NC 变量 x 的数据（数据类型：WORD）

变量“rpa”

R 参数

地址	值 / 含义
DB120x.DBB1000	5
DB120x.DBB1001	-
DB120x.DBW1002	R 号 + 1
DB120x.DBW1004	-
DB120x.DBD1008	**写入**：数据至 NC 变量 x（数据类型：REAL)
DB120x.DBW3004	**读取**：NC 变量 x 的数据（数据类型：REAL)

变量“actLineNumber”

当前 NC 程序段的行编号

编号	含义
≥ 1	当前 NC 程序段的行编号
0	程序未启动，因此无法提供行编号
-1	无法提供行编号：故障
-2	无法提供行编号：通过 DISPLOF 激活了程序段显示抑制

地址	值 / 含义
DB120x.DBB1000	6
DB120x.DBB1001	-
DB120x.DBW1002	-
DB120x.DBW1004	-
DB120x.DBD1008	-
DB120x.DBW3004	**读取**：NC 变量 x 的数据（数据类型：INT）

刀具管理：刀位数据

刀位类型（$TC_MPP2）

地址	值 / 含义
DB120x.DBB1000	7
DB120x.DBB1001	-
DB120x.DBW1002	刀位编号（1 … 31999）
DB120x.DBW1004	刀库编号（1 … 9999）
DB120x.DBD1008	-

地址	值 / 含义		
DB120x.DBW3004	读取：NC 变量的值（数据类型：WORD）		
		值	含义
		> 0	虚拟刀位的刀位类型
		0	“match all”（缓冲区）
		9999	未定义（非虚拟刀位）

刀位状态（$TC_MPP4）

地址	值 / 含义		
DB120x.DBB1000	8		
DB120x.DBB1001	-		
DB120x.DBW1002	刀位编号（1 … 31999）		
DB120x.DBW1004	刀库编号（1 … 9999）		
DB120x.DBD1008	-		
DB120x.DBW3004	读取：NC 变量的值（数据类型：WORD）		
		值	含义
		1	禁用
		2	未占用（<> 占用）
		4	预留用于缓冲区中的刀具
		8	预留用于待装载的刀具
		16	左半刀位占用
		32	右半刀位占用
		64	上半刀位占用
		128	下半刀位占用

该刀位上刀具的 T 编号（$TC_MPP6）

地址	值 / 含义
DB120x.DBB1000	9
DB120x.DBB1001	-
DB120x.DBW1002	刀位编号（1 … 31999）

地址	值 / 含义
DB120x.DBW1004	刀库编号（1 … 9999）
DB120x.DBD1008	-
DB120x.DBW3004	读取：该刀位上刀具的 T 编号（数据类型：WORD）

变量 r0078[1]

- 驱动对象：SERVO、SERVO_AC、SERVO_I_AC
- CO:转矩生成电流实际值 [A_{eff}]
- 下标 [1]：已通过 p0045 平滑

地址	值 / 含义
DB120x.DBB1000	10
DB120x.DBB1001	驱动模块的编号
DB120x.DBW1002	-
DB120x.DBW1004	-
DB120x.DBD1008	-
DB120x.DBW3004	读取：NC 变量 x 的数据（数据类型：REAL)

变量 r0079[1]

- 驱动对象：SERVO、SERVO_AC、SERVO_I_AC
- CO:转速控制器输出端的转矩设定值（在时钟周期插补前）[Nm]
- 下标 [1]：已通过 p0045 平滑

地址	值 / 含义
DB120x.DBB1000	11
DB120x.DBB1001	驱动模块的编号
DB120x.DBW1002	-

地址	值 / 含义
DB120x.DBW1004	-
DB120x.DBD1008	-
DB120x.DBW3004	读取：NC 变量 x 的数据（数据类型：REAL)

变量 r0081

- 驱动对象：SERVO、SERVO_AC、SERVO_I_AC
- CO：转矩利用率，百分比值

地址	值 / 含义
DB120x.DBB1000	12
DB120x.DBB1001	驱动模块的编号
DB120x.DBW1002	-
DB120x.DBW1004	-
DB120x.DBD1008	-
DB120x.DBW3004	读取：NC 变量 x 的数据（数据类型：REAL)

变量 r0082[1]

- 驱动对象：SERVO、SERVO_AC、SERVO_I_AC
- CO:有功功率实际值 [kW]
- 下标 [1]：已通过 p0045 平滑

地址	值 / 含义
DB120x.DBB1000	13
DB120x.DBB1001	驱动模块的编号
DB120x.DBW1002	-

地址	值 / 含义
DB120x.DBW1004	-
DB120x.DBD1008	-
DB120x.DBW3004	读取：NC 变量 x 的数据（数据类型：REAL)

温度补偿

变量“TEMP_COMP_ABS_VALUE”（SD43900）

与位置无关的温度补偿值

地址	值 / 含义
DB120x.DBB1000	14
DB120x.DBB1001	轴编号（1, 2, …）
DB120x.DBW1002	-
DB120x.DBW1004	-
DB120x.DBD1008	写入：数据至 NC 变量 x（数据类型：REAL)
DB120x.DBW3004	读取：NC 变量 x 的数据（数据类型：REAL)

变量“TEMP_COMP_SLOPE”（SD43910）

与位置相关的温度补偿的斜率角

地址	值 / 含义
DB120x.DBB1000	15
DB120x.DBB1001	轴编号（1, 2, …）
DB120x.DBW1002	-
DB120x.DBW1004	-

地址	值 / 含义
DB120x.DBD1008	写入：数据至 NC 变量 x（数据类型：REAL)
DB120x.DBW3004	读取：NC 变量 x 的数据（数据类型：REAL)

变量“TEMP_COMP_REF_POSITION”（SD43920）

位置相关温度补偿的参考位置

地址	值 / 含义
DB120x.DBB1000	16
DB120x.DBB1001	轴编号（1, 2, …）
DB120x.DBW1002	-
DB120x.DBW1004	-
DB120x.DBD1008	写入：数据至 NC 变量 x（数据类型：REAL)
DB120x.DBW3004	读取：NC 变量 x 的数据（数据类型：REAL)

变量“TOOL_TEMP_COMP”（SD42960[...]）

与刀具相关的温度补偿

地址	值 / 含义
DB120x.DBB1000	17
DB120x.DBB1001	-
DB120x.DBW1002	索引 + 1 (1, 2, 3)
DB120x.DBW1004	-
DB120x.DBD1008	写入：数据至 NC 变量 x（数据类型：REAL)
DB120x.DBW3004	读取：NC 变量 x 的数据（数据类型：REAL)

15.6.1.9 设定所选 NC 变量

所选 NC 变量通过数据块 DB9910（所选 NC 变量）设定。数据块长度取决于在变量列表中选择的 NC 变量的数量。变量列表最多可包含 42 个 NC 变量。数据块 DB9910 包含每个 NC 变量的数据，用于变量定址和数据类型转换。

DB9910	所选 NC 变量，读取 (PLC → NC)							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBB0	**变量 1** 的变量下标							
DBB1	句法 ID							
DBB2	范围							
DBB3	单位							
DBW4	栏索引							
DBW6	行索引							
DBB8	模块							
DBB9	行数							
DBB10	类型							
DBB11	长度							
....	.....							
DBB492	**变量 42** 的变量下标							
DBB493	句法 ID							
DBB494	范围							
DBB495	单位							
DBW496	栏索引							
DBW498	行索引							
DBB500	模块							
DBB501	行数							
DBB502	类型							
DBB503	长度							

变量索引

变量索引以 NC 变量的名称为基准。变量索引由一个初始值 100 和列表（0 至 41）中 NC 变量的偏移组成。

变量索引在用户接口 DB120x RW_NCDx 中以 DBB1000 A_VarIdx 输入。

注释包含下列以空格隔开的数据组：

- 范围
- 模块
- 变量名称
- 变量类型
- 列
- 变量字节数

其它用户接口

由于数据块中的数据检查，变量值只能写入数据块中与其类型相同的地址，例如：REAL 值只能写入 REAL 地址中（例如 MOV_R）。目前为止用户接口“读/写 NC 变量”只包含一个 DWORD 类型地址用于写入值（DBD1008）。因此一个 REAL 值只能通过一个辅助变量，标志位或累加器写入。对于读取也同样适用（DBD3004）。因此用户接口“读/写NC 变量”得以扩展。分别对写入和读取的每一个类型添加一个相应类型的地址：REAL, DWORD/DINT, WORD/INT, BYTE 和 BOOL (DBD1016 … DBB1027 bzw. DBD3016 … DBB3027)。这些新地址通过 PLC 固件只用于特定变量，这些变量由 NC 变量编辑器选出并在编译时输入到 DB9910 NC_DATA 中（变量索引 ≥ 100）。目前为止确定在用户接口“读/写 NC 变量”中定义的 NC 变量继续使用原先地址（DBD1008 或 DBD3004）。

15.6.2 程序实例服务（PI 服务）

15.6.2.1 任务设定

PI 服务通过**任务接口**（DB1200，偏移 4000 起）设定。

DB1200			**PI 服务 [r/w]**					
			PLC → NC 接口					
字节	**位 7**	**位 6**	**位 5**	**位 4**	**位 3**	**位 2**	**位 1**	**位 0**
DBB 4000	-	-	-	-	-	-	-	启动 [1]
DBB 4001	PI 下标 [2]							
DBB 4002	-							
DBB 4003	-							

DB1200	PI 服务 [r/w]
DBW 4004	PI 参数 1 3)
DBW 4006	PI 参数 2
DBW 4008	PI 参数 3
DBW 4010	PI 参数 4
DBW 4012	PI 参数 5
DBW 4014	PI 参数 6
DBW 4016	PI 参数 7
DBW 4018	PI 参数 8
DBW 4020	PI 参数 9
DBW 4022	PI 参数 10
1) DB1200.DBX4000.1，启动：DBX4000.1 = 1 ⇒ PI 服务开始；DBX4000.1 = 0 ⇒ PI 服务结束 2) DB1200.DBB4001，PI 下标：指定具体的 PI 服务 3) DB1200.DBW4004 ..., PI 参数：PI 专用参数	

PI 服务一览：

- PI 服务 ASUB (页 1288)
- PI 服务 LOGOUT (页 1290)
- PI 服务 DATA_SAVE (页 1291)
- PI 服务 TMMVTL (页 1291)

15.6.2.2 任务反馈信息

PLC 在**结果接口**（DB1200，偏移 5000 起）中提供对“已启动 PI 服务成功与否”的反馈信息。

其通过以下信号提示任务结束：

- DB1200.DBX5000.0
- DB1200.DBX5000.1

这些信号由 PLC 写入，对于用户为只读。

一旦用户将信号“启动”（DB1200.DBX4000.1）复位，任务便告完成。 之后状态信号 DB1200.DBX5000.0 和 .1 会被置零。

DB1200			PI 服务 [r]					
			NCK → PLC 接口					
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBB 5000	-	-	-	-	-	-	任务中出错 2)	任务结束 1)
DBB 5001	-							
DBB 5002	-							
1) DB1200.DBX5000.0，任务状态：DBX5000.0 == 1 ⇒ 任务结束 2) DB1200.DBX5000.1，故障状态：DBX5000.1 == 0 ⇒ 无故障；DBX5000.1 == 1 ⇒ 故障								

可能的故障原因

- 参数设置的 PI 服务（DB1200.DBB4001）的下标超出有效范围
- 参数错误

15.6.2.3 PI 服务 ASUB

中断

说明

通道与中断的关系

最多只有**一个**通道的控制系统可以使用**两个**中断，最多有**两个**通道的控制系统可以使用**四个**中断。

通过 PI 服务 “AUSB” 可由 PLC 从每个中断编号中指定一个中断程序 (ASUB)。如果触发了相应的中断，系统会在 NC 上执行该中断程序。为此，这些中断程序必须以下列程序名保存在 NC 中的机床制造商目录 (_N_CMA_DIR) 下。

中断	程序名
1	PLCASUP1_SPF
2	PLCASUP2_SPF
3	PLCASUP3_SPF
4	PLCASUP4_SPF

若中断程序不存在，则须新建。之后必须在 NC 上触发 NC 复位（上电）。

控制系统启动后每个中断指定只须执行一次 PI 服务 “ASUB”。中断和中断程序的指定关系会保留至下一次启动。

任务设定

PI 服务：ASUB		
地址	**说明**	**有效值**
DB1200.DBW4001	PI 下标 1)	1, 2, 13, 14
DB1200.DBW4004	参数 1：LIFTFAST 2)	0 (FALSE), 1 (TRUE)
DB1200.DBW4006	参数 2：BLSYNC 3)	0 (FALSE), 1 (TRUE)
DB1200.DBW4008	参数 3：通道下标 4)	0, 1
DB1200.DBW4010	参数 4：中断优先级 5)	0, 1, 2, 3, 4
1) • PI 下标 = 1：中断 1 ⇒ _N_CMA_DIR / PLCASUP1_SPF，默认的中断优先级：1 • PI 下标 = 2：中断 2 ⇒ _N_CMA_DIR / PLCASUP2_SPF，默认的中断优先级： 2 • PI 下标 = 13：中断 3 ⇒ _N_CMA_DIR / PLCASUP3_SPF，默认的中断优先级： 3 • PI 下标 = 14：中断 4 ⇒ _N_CMA_DIR / PLCASUP4_SPF，默认的中断优先级： 4		
2) LIFTFAST 会导致触发中断后首先进行快速退刀。之后才执行中断程序。 **文档** 功能手册之特殊功能；章节“R3：扩展停止和回退” > “控制系统引导的 ESR” > “回退”		
3) 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。		
4) 0 → 通道 1，1 → 通道 2		
5) 使用默认的中断优先级时，必须将参数设为 0。		

机床数据

- 最低中断优先级
  在以下机床数据中确定了 NC 的最低中断优先级，执行相应的中断。如果中断的优先级低于机床数据中给定的最低优先级，则不会在控制系统中执行：
  MD11604 $MN_ASUP_START_PRIO_LEVEL

前提条件

通道状态

仅当请求服务的通道处于“复位”状态时，才允许执行 PI 服务“ASUB”。

ProgEvent“启动”

若针对事件控制程序调用 (ProgEvent) 将“启动”配置为触发事件 (MD20108 $MC_PROG_EVENT_MASK)，那么要待 ProgEvent 程序（PROG_EVENT_SPF 或 MD11620 $MN_PROG_EVENT_NAME = <Anwender_Prog_event_SPF>) 结束后才允许启动 PI 服务 “ASUB”。

参见

任务设定 (页 1286)

任务反馈信息 (页 1287)

15.6.2.4 PI 服务 LOGOUT

功能

将最近一次传输至 NC 的口令复位。

任务设定

PI 服务：LOGOUT		
地址	**说明**	**有效值**
DB1200.DBW4001	PI 下标	3（复位口令）

15.6.2.5 PI 服务 DATA_SAVE

功能

在系统 CF 卡上恢复 NC 当前状态的数据。

说明

启动

下一次启动控制系统时可在 “Startup menu” 中通过 “Reload saved user data” 将保存的状态载入 NC。

任务设定

PI 服务：DATA_SAVE		
地址	**说明**	**有效值**
DB1200.DBW4001	PI 下标	4

文档

SINUMERIK 828D 调试手册；章节 “数控调试” > “供货范围和前提条件” > “启动控制系统”

15.6.2.6 PI 服务 TMMVTL

功能

通过 PI 服务 TMMVTL 可由 PLC 请求刀具换位任务。 基于 PI 服务，刀具管理在目标刀库中为 PI 服务中设定的刀具（刀具编号或源刀位编号/源刀库编号）搜索空刀位。 之后 PLC 通过 DB41xx.DBB0 接收刀具换位任务。

任务设定

地址	**说明**	**有效值**
DB1200.DBW4001	PI 索引	5
DB1200.DBW4004	参数 1：刀具编号 1)	-1, 1 … 31999
DB1200.DBW4006	参数 2：源刀位编号 1)	-1, 1 … 31999
DB1200.DBW4008	参数 3：源刀库编号 1)	-1, 1 … 9999
DB1200.DBW4010	参数 4：目标刀位编号 2)	-1, 1 … 31999

地址	说明	有效值
DB1200.DBW4012	参数 5：目标刀库编号 3)	-1, 1 … 9999

1) 可选择通过以下方式指定刀具：

- 刀具编号（T 编号）
- 源刀位编号和源刀库编号

未采用的方案的参数必须输入值 -1。

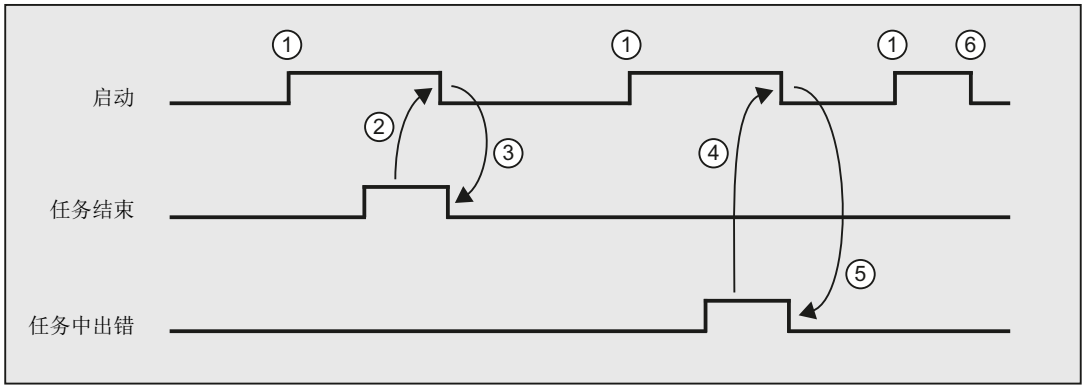
2) 目标刀位编号 = -1 时，系统会根据设置的所搜方案在整个刀库中为刀具搜索空刀位。 若设定了目标刀位，那么系统会检查该刀位是否未占用并适用于刀具。

3) 目标刀库编号 = -1 时，系统会按照从 $TC_MDP2 得出的对应关系，为所有中间存储器上的刀具进行搜索。

应用示例

- 在使用中间存储器（例如抓刀器和/或换刀机构）用于送回刀具时，在异步送回期间可能需要在刀库中进行明确的空刀位搜索。 在这种情况下 PLC 不需要标示源刀位，PI 服务 TMMVTL 会搜索适合的刀位。
- 刀具需从后台刀库移动至前台刀库。

15.6.2.7 PI 服务：周期图



① 用户设置“启动”信号，任务开始。

② PLC 固件发出“任务结束”信号后，用户复位“启动”信号。

③ “启动”信号的复位触发 PLC 固件对“任务结束”信号的复位。

④ PLC 固件发出“任务中出错”信号后，用户复位“启动”信号。

⑤ “启动”信号的复位触发 PLC 固件对“任务中出错”信号的复位。

⑥ 在 PLC 发出“任务结束”或“任务中出错”信号前，若用户无意间复位了“启动”信号，那么此任务的结果信号将不会被更新。但任务仍会被执行。

15.6.3 PLC 用户报警

15.6.3.1 用户接口

说明

尽管下文称之为用户“*报警*”，实际上要在输入**删除标准**（参见“定义用户报警 (页 1296)”）时才能确定是**消息**还是**报警**。

借助 DB1600 中的用户接口，可在 HMI 上显示故障消息和运行消息。

这包含以下服务：

- 激活用户报警 700000 至 700247 和扩展用户报警 701000 至 701999。

  说明

  使用扩展 PLC 用户报警 701000 至 701999 所适用的前提条件是：
  - 兼容模式必须关闭。
  - 数据块 DB9913 必须包含在 PLC 程序中（即：DB9913 在 PLC 编程工具中的“库” > “特殊数据块”下选中并借助复制和粘贴/双击接收到 PLC 程序中）。

- 用户报警 700000 至 700247 和 701000 至 701247 可带有一个附加的数字参数。
- 取消和应答用户报警。
- 对因用户报警而触发的系统响应进行分析。

固件会对输入的信号进行分析，并将其作为“出现”和“消失”的报警和消息发送至 HMI 进行显示。故障文本由 HMI 管理。

15.6.3.2 用户报警的激活接口

每个用户报警都通过指定的激活位激活。这些位在**激活接口**中设置。

新的用户报警通过相应位的 0/1 脉冲沿激活。

报警 700000 至 700247 的激活接口

DB1600	激活报警 [r/w] (PLC → HMI)							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
	激活报警号							
DBB0	700007	700006	700005	700004	700003	700002	700001	700000

DB1600	激活报警 [r/w] (PLC → HMI)							
	激活报警号							
DBB1	700015	700014	700013	700012	700011	700010	700009	700008
	激活报警号							
DBB2	700023	700022	700021	700020	700019	700018	700017	700016
	激活报警号							
DBB3	700031	700030	700029	700028	700027	700026	700025	700024
	激活报警号							
DBB4	700039	700038	700037	700036	700035	700034	700033	700032
	激活报警号							
DBB5	700047	700046	700045	700044	700043	700042	700041	700040
...	….							
	激活报警号							
DBB30	700247	700246	700245	700244	700243	700242	700241	700240

报警 701000 至 701999 的激活接口

DB1600	激活报警 [r/w] (PLC → HMI)							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
	激活报警编号							
DBB4000	701007	701006	701005	701004	701003	701002	701001	701000
	激活报警编号							
DBB4001	701015	701014	701013	701012	701011	701010	701009	701008
	激活报警编号							
DBB4002	701023	701022	701021	701020	701019	701018	701017	701016
	激活报警编号							
DBB4003	701031	701030	701029	701028	701027	701026	701025	701024
	激活报警编号							
DBB4004	701039	701038	701037	701036	701035	701034	701033	701032
	激活报警编号							
DBB4005	701047	701046	701045	701044	701043	701042	701041	701040
...	….							

DB1600	激活报警 [r/w] (PLC → HMI)							
	激活报警编号							
DBB4124	701999	701998	701997	701996	701005	701994	701993	701992

15.6.3.3 用户报警的变量接口

可为用户报警 700000 至 700247 和 701000 至 701247 指定一个变量作为参数。为此，在**变量接口**中为每个报警预留一个双字。因此有效的偏移必须能被 4 整除。

报警 700000 至 700247 的变量接口

DB1600	报警变量 [r32/w32] (PLC → HMI)							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBD1000	报警 700000 的变量							
DBD1004	报警 700001 的变量							
DBD1008	报警 700002 的变量							
	...							
DBD1980	报警 700245 的变量							
DBD1984	报警 700246 的变量							
DBD1988	报警 700247 的变量							

报警 701000 至 701247 的变量接口

DB1600	报警变量 [r32/w32] (PLC → HMI)							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBD5000	报警 701000 的变量							
DBD5004	报警 701001 的变量							
DBD5008	报警 701002 的变量							
	...							
DBD5980	报警 701245 的变量							
DBD5984	报警 701246 的变量							
DBD5988	报警 701247 的变量							

15.6.3.4 定义用户报警

可为每个报警定义以下属性：

- 报警响应：出现报警时控制系统的响应方式。
- 删除标准：为了删除或应答报警所须采取的措施。删除标准同时还决定了报警的类型和优先级。
- 通道分配：报警所分配的通道。

用户报警 700000 至 700247

通过机床数据定义用户报警 700000 至 700247。

报警响应和删除标准

通过以下机床数据确定报警响应和删除标准：

MD14516 $MN_USER_DATA_PLC_ALARM [x] = <报警响应和删除标准>

其中 x = 用户报警编号 - 700000；取值范围：0 ≤ x ≤ 247

<报警响应和删除标准>	
位	含义
报警响应	
0	NC 启动禁止
1	读取禁止
2	所有轴的进给禁止
3	急停
4	PLC 停止
5	报警记录
删除标准	
6	使用 DB1600 DBX3000.0 终止
7	上电
位 x = 1：激活功能 位 x = 0：撤销功能	

通道分配

通过以下机床数据进行通道分配：

MD14518 $MN_USER_DATA_PLC_ALARM_ASSIGN [x] = <通道分配>

其中 x = 用户报警编号 - 700000；取值范围：0 ≤ x ≤ 247

<通道分配>	
位	含义
0	NC 通道 1 的适用范围
1	NC 通道 2 的适用范围
位 x = 1：通道已选 位 x = 0：通道撤销	

用户报警 701000 至 701999

在数据块 DB9913 (ALARM_INI) 中定义用户报警 701000 至 701999。每个报警需要两个字节的配置数据。一个字节用于报警响应和删除标准，另一个字节用于通道分配。这两个字节汇总在一个字中。

DB9913	配置的用户报警 701000 到 701999 [r]
DBW0	报警 701000
DBW2	报警 701001
DBW4	报警 701002
	...
DBW1998	报警 701999

位	含义
报警响应	
0	NC 启动禁止
1	读取禁止
2	所有轴的进给禁止
3	急停
4	PLC 停止
5	报警记录
删除标准	
6	使用 DB1600 DBX3000.0 终止
7	上电

位	含义
通道分配	
8	NC 通道 1 的 PLC 用户报警
9	NC 通道 2 的 PLC 用户报警
10	预留
11	预留
12	预留
13	预留
14	预留
15	预留

删除标准和优先级

通过位 6 和 7 设置用户报警的删除标准和所含的类型和优先级：

位 7	位 6	删除标准：	类型	优先级
0	0	复位激活位	消息	低
0	1	在 DB1600.DBX3000.0 中应答（参见“用户报警的应答接口 (页 1299)”）	报警	中
1	0	上电	报警	高
1	1	预留（内部值：位 7 = 1，位 6 = 0）	-	-

显示消息

如果上述用户报警中没有激活报警响应（机床数据位 0 至位 4 = 0），其将被定义为对系统无影响的“显示消息”。这意味着相应机床数据的删除标准（机床数据位 6 和位 7）也将不会被分析。

15.6.3.5 读取生效的报警响应和删除标准

当前**生效的报警响应**（即实际响应）、**生效的删除标准**和**生效的通道分配**可从接口全局读取：

DB1600	生效的报警响应 [r]							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBB2000	上电	使用 DB1600 DBX3000.0 终止		PLC 停止	急停	通道 1		
						所有轴进给禁止	读取禁止	NC 启动禁止
DBB2001						通道 2		
						所有轴的进给禁止	读取禁止	NC 启动禁止

至少为一个生效的报警配置了对应的响应或删除标准时，一个位会被置位。若之后不再为存在的报警配置响应或删除条件，那么该位将被删除。

15.6.3.6 用户报警的应答接口

应答用户报警的前提条件是对应激活位的复位。

- 删除标准为 {0,0} 的消息会自动从现实中消失。
- 删除标准为 {0,1} 的报警由应答位 *Quit* 删除。
- 删除标准为 {1,0} 的报警不受应答位置位影响，只能通过上电删除。

DB1600	报警应答 [r/w]							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBB3000								Quit
DBB3001								
DBB3002								
DBB3003								

15.6.3.7 HMI 接口

PLC 可向 HMI 传输八条消息或报警用于显示，显示采用消息或报警出现的顺序。

出现其他消息/报警时，前七条保留在 HMI 中，最近期的消息或报警则会按照以下规则由刚出现的替换：

- 系统消息/报警会替换用户消息/报警。
- 具有更高优先级的消息/报警会替换低优先级的消息/报警。

保留前七条消息/报警是因为他们很有可能是造成问题的原因，而之后的消息/报警只是次要原因。但是，如果一条或多条消息/报警被应答并借此删除，那么 HMI 中会补充对应数量的新报警/消息。

15.6.4 PLC 轴控制

15.6.4.1 简介

使用 PLC 可通过用户接口数据块对轴或主轴进行控制，此时轴/主轴通过 DB 编号指定：

- **DB380x** 接口 PLC → NC (到轴/主轴)
- **DB390x** 接口 NC → PLC (来自轴/主轴)

轴索引 x：0 ≤ x ≤ 最大轴索引；轴索引 = 轴编号 - 1

可支持以下功能：

- 定位轴
- 主轴定位
- 主轴旋转
- 主轴往复
- 分度轴

文档

- 功能手册之扩展功能分册；定位轴（P2）和分度轴（T1）
- 功能手册之基本功能分册；主轴（S1）

前提条件

待控制的轴必须已被指定给 PLC（PLC 轴）。通过用户接口“*跨通道取轴*”（DB3800.DBB8/DB3900.DBB8）可实现 NC 和 PLC 间的轴交换。

功能启动

每种功能都通过对应“启动”信号的上升沿触发。该信号必须保持在逻辑“1”，直至该功能被积极或消极应答（例如通过*到达位置*= “1”或*故障* =“1”）。信号“定位轴生效”=“1”表示该功能生效且输出字节有效。

终止

无法通过复位启动信号来**终止**功能，而只能通过其他接口信号进行（例如通过轴专用信号*删除剩余行程/主轴复位*，DB380x DBX2.2）。

轴接口会反馈轴的状态信号，必要时须分析这些信号（例如*准停*、*运行指令*→ DB390x）。

在调用 PLC 轴控制时，若轴/主轴由于通过 NC 运行被占用（存在运行指令），那么要到该运动结束后才能启动调用。在此情形下系统不会输出故障代码。

轴禁止

设置了**轴禁止**时，由 PLC 控制的轴无法运行。此时只会生成仿真的实际值。（特性与 NC 编程时相同）。

15.6.4.2 用户接口：将 NC 轴作为 PLC 轴

发送至轴/主轴的请求信号（节选）

首先须通过 PLC 请求轴/主轴：

DB380x	发送到进给轴/主轴的信号 (PLC → NC) [r/w]							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBB8	请求 PLC 轴/主轴			更改此字节时的激活信号			为通道指定 NC 轴/主轴	
							B	A

修改请求信号 (DB380x.DBX8.7 oder 8.0) 时，必须通过激活信号 (DB380x.DBX8.4) 的 0→1 脉冲沿通知 NC。PLC 循环结束后必须再次复位激活信号。

来自轴/主轴的状态信号（节选）

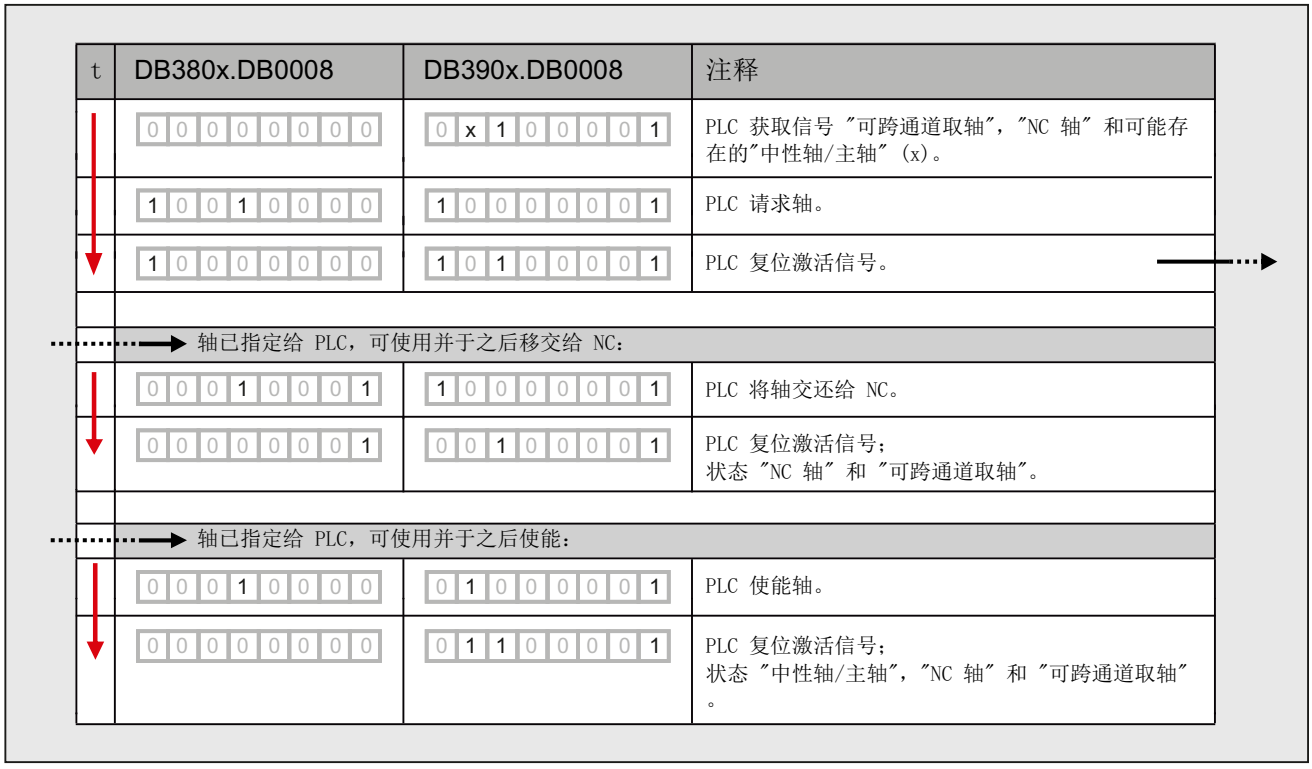
DB390x	来自进给轴/主轴的信号 (NC → PLC) [r]							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBB8	PLC 轴/主轴	中性轴/主轴	允许跨通道取轴	PLC 请求新类型			通道中 NC 轴/主轴的当前分配	
							B	A

说明

模拟

如需激活接口信号，必须在模拟时为每个所需轴置位机床数据 MD30350 $MA_SIMU_AX_VDI_OUTPUT。

请求和交还 PLC 轴



15.6.4.3 用户接口：功能

下面两个表为可供使用的接口信号一览。信号的详细说明及适用范围将在下文中介绍。

发送至 PLC 轴的信号

DB380x	到 PLC 轴的信号 (PLC → NC) [r/w]							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBB3000	启动定位轴	启动主轴定位	启动主轴旋转	启动主轴往复	-	-	-	-
DBB3001	-	-	停止主轴旋转	停止主轴往复	-	-	-	-
DBB3002	自动齿轮档选择	恒定切削速度	旋转方向同 M4	-	启用手轮叠加	运行尺寸 Inch（非公制）	行程条件：最短行程（DC）	行程条件：增量（IC）
DBB3003	分度位置	-	-	-	-	-	行程条件：沿正方向逼近（ACP）	行程条件：沿负方向逼近（ACN）
DBD3004	位置 (REAL, 对于分度轴:DINT)							
DBD3008	进给轴（REAL），若 < 0，则从机床数据 POS_AX _VELO 取值。							

行程条件的位和方向设定的位用于定义相应的定位或运行模式，只能设置其中一个位：

含义	待设置的行程条件
绝对定位	未设定模式位
增量定位	DBB3002.0 = 1
以最短行程定位	DBB3002.1 = 1
沿正方向绝对定位	DBB3003.1 = 1
沿负方向绝对定位	DBB3003.0 = 1
旋转方向同 M4	DBB3002.5 = 1

其他位用于指定和启动相应功能，这些功能位及地址和速度会在各功能中详细说明。

PLC 轴发出的信号

DB390x	来自 PLC 轴的信号 (NC → PLC) [r]							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBB3000	定位轴生效	到达位置		-	-	-	运行时出现故障	轴无法启动
DBB3001	-	-	-	-	-	-	-	-
DBB3002	-	-	-	-	-	-		-
DBB3003	故障编号							

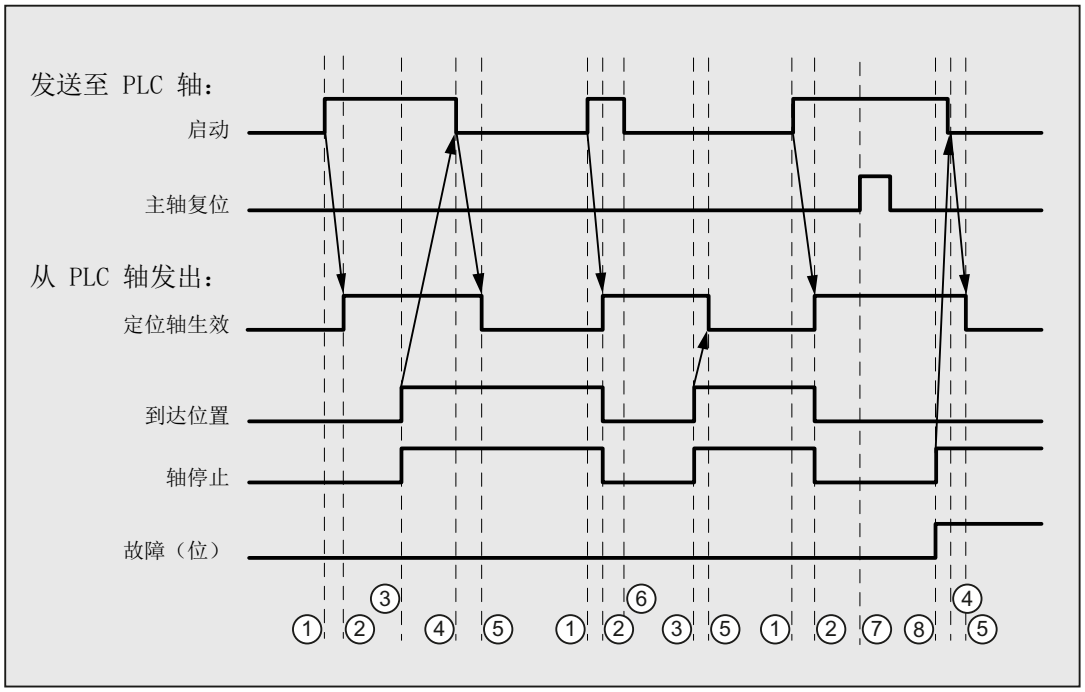
为了使用下面列出的功能，必须满足以下前提条件：

- 轴或主轴按照规定指定给了 PLC。
- 控制器使能和脉冲使能已设置。
- 设置所有控制信号后，启动信号中只有一个在 DB380x.DBB3000 中设置。

15.6.4.4 主轴定位

DB380x	控制信号 PLC → NC	注释
DBX3002.0	增量运行	同一时间始终只能置位四个信号当中的一个。 所有信号 0：绝对定位
DBX3002.1	以最短行程运行	
DBX3003.0	绝对定位，负方向	
DBX3003.1	绝对定位，正方向	
DBD3004	设定位置/设定行程	设定行程，当 DBX3002.0 == 1 时
DBD3008	进给速度	0：以 MD35300 $MA_SPIND_POSCTRL_VELO 中的值运行
DBX3000.6	启动	**提示**复位信号不会导致停止！
DBX2.2	删除剩余行程，主轴复位	终止信号，用于结束功能

DB390x	状态信号 NC → PLC	注释
DBX3000.7	定位轴生效	倍率 = 0 或到达设定位置时置 1 启动时 = 1
DBX3000.6	到达位置	1：通过“精准停”到达设定位置
DBX3000.0	主轴无法启动	
DBX3000.1	运行时出现故障	1：运行时出现故障，故障编号在 DBB3003 中分析！
DBB3003	故障编号	
DBX1.4	进给轴/主轴停止	1：当 $n < n_{min}$



① 用户通过*启动*的上升沿触发功能。

② *定位轴生效*消息显示该功能生效且输出信号有效，*到达位置*和*轴停止*同样会被复位。赋值 = 0 时信号不会被复位。

③ 到达位置时发送*到达位置*信号，*主轴停止*置位。

④ 之后用户复位*启动*。

⑤ 之后信号*定位轴生效*复位。

⑥ 得到*定位轴生效*信号后，用户立即复位*启动*信号。

⑦ 定位通过设置*主轴复位*终止。此信号必须保持至少一个 PLC 周期。

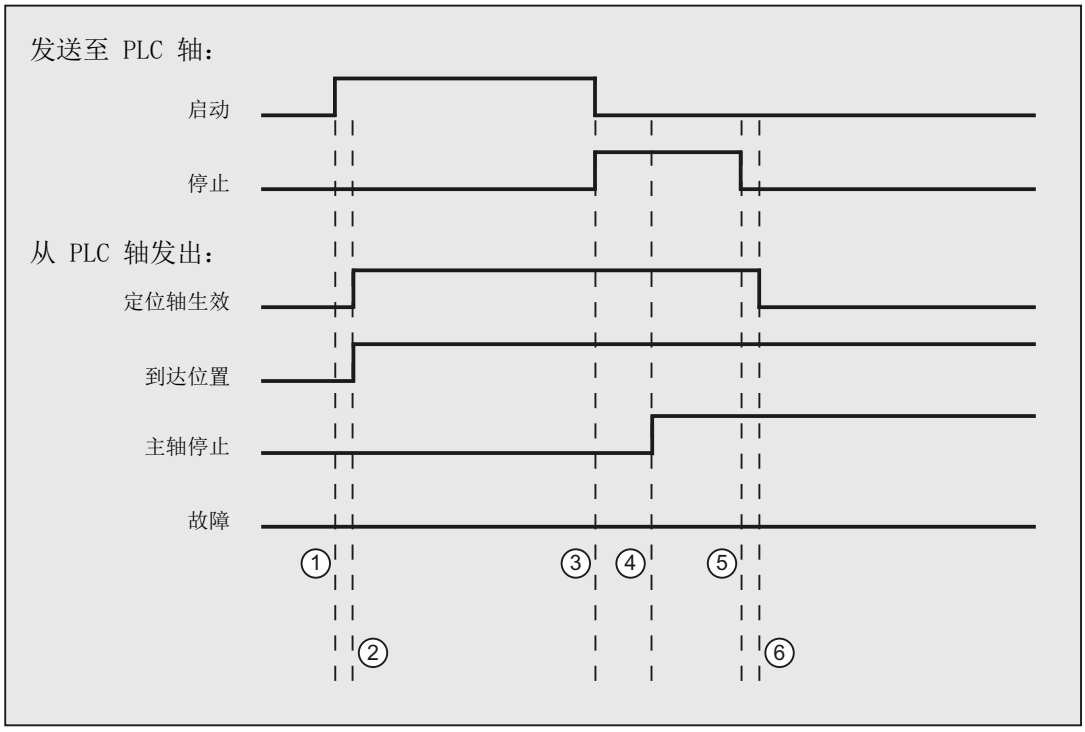
⑧ 主轴进入停止状态（*主轴停止*），信号*故障*置位。（在此情形下系统会输出故障编号 115）

15.6.4.5 主轴旋转

DB380x	控制信号 PLC → NC	注释
DBX3002.0	增量运行	-
DBX3002.1	以最短行程运行	-

DB380x	控制信号 PLC → NC	注释
DBX3002.5	旋转方向同 M4	1：旋转方向已通过 M4 设定 0：旋转方向已通过 M3 设定
DBX3003.0	绝对定位，负方向	-
DBX3003.1	绝对定位，正方向	-
DBD3008	进给速度	主轴转速
DBX3000.5	启动主轴旋转	-
DBX3001.5	停止主轴旋转	-

DB390x	状态信号 NC → PLC	注释
DBX3000.7	定位轴生效	1：启动或停止 == 1
DBX3000.6	到达位置	1：该功能已无故障启动
DBX3000.1	故障	1：运行时出现故障，故障编号在 DBB3003 中分析！
DBB3003	故障编号	-
DBX1.4	进给轴/主轴停止	-



① 用户通过*启动*的上升沿触发功能。

② *定位轴生效*和*到达位置*复位，*到达位置*此处不相关。

③ 用户复位*启动*并设置*停止*，从而停止主轴旋转。

④ 主轴停止，且信号*主轴停止*置位。

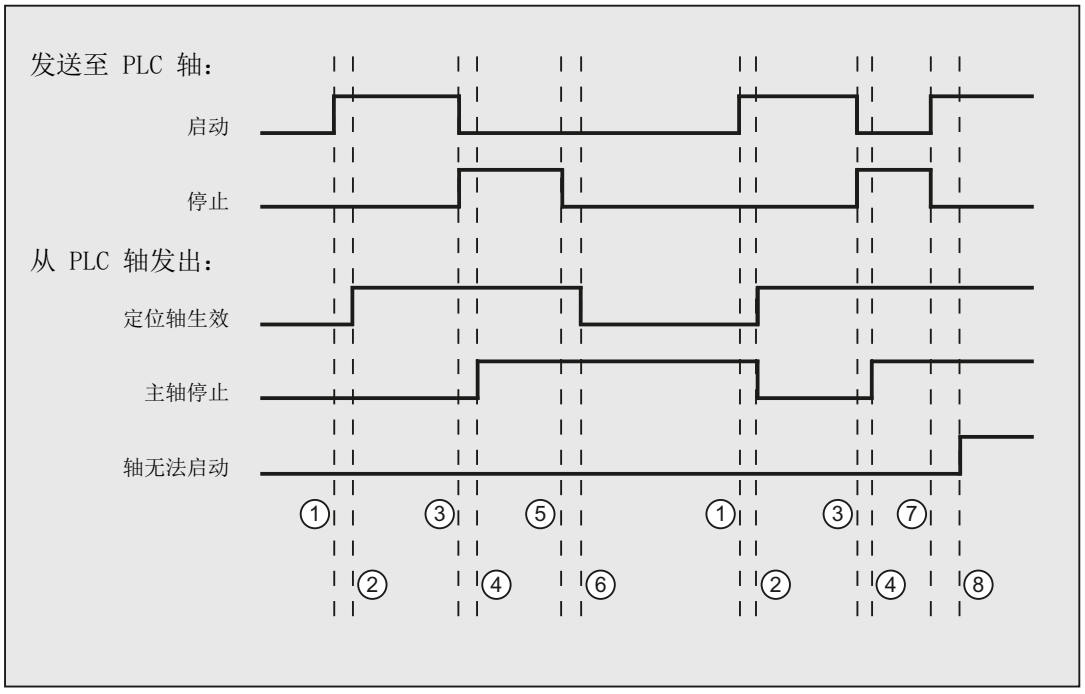
⑤ 之后用户复位*停止*。

⑥ *停止*的复位会引起*定位轴生效*的复位。

15.6.4.6 主轴往复

DB380x	控制信号 PLC → NC	注释
DBX3002.0	增量	这些位均不允许置位。
DBX3002.1	最短行程	
DBX3002.5	旋转方向同 M4	
DBX3003.0	绝对定位，负方向	
DBX3003.1	绝对定位，正方向	
DBD3004	设定齿轮级	MD35010 $MA_GEAR_STEP_CHANGE_ENABLE = 0 0 - 5：往复 MD35010 $MA_GEAR_STEP_CHANGE_ENABLE = 1 0：往复 1：往复运行和齿轮档切换 M41 2：往复运行和齿轮档切换 M42 3：往复运行和齿轮档切换 M43 4：往复运行和齿轮档切换 M44 5：往复运行和齿轮档切换 M45
DBD3008	进给速度	在往复运行中无含义！往复转速取自以下机床数据：MD35400 $MA_SPIND_OSCILL_DES_VELO
DBX3000.5	启动主轴往复	不允许在停止后直接启动。首先须复位停止（均置 0）。
DBX3001.5	停止主轴往复	

DB390x	状态信号 NC → PLC	注释
DBX3000.7	定位轴生效	1：启动或停止 == 1
DBX3000.6	到达位置	1：启动后
DBX3000.1	故障	1：运行时出现故障，故障编号在 DBB3003 中分析！
DBX3000.0	轴无法启动	1：启动时出现故障，故障编号在 DBB3003 中分析！
DBB3003	故障编号	
DBX1.4	进给轴/主轴停止	



① 用户通过*启动*的上升沿触发功能。

提示：仅在*定位轴生效*信号复位时才可执行！

② *定位轴生效*和*到达位置*复位，*到达位置*此处不相关，不会被考虑。

③ 用户复位*启动*并设置*停止*，从而停止主轴往复。

④ 主轴停止，且信号*主轴停止*置位。

⑤ 之后用户复位*停止*。

⑥ *停止*的复位会引起*定位轴生效*的复位。

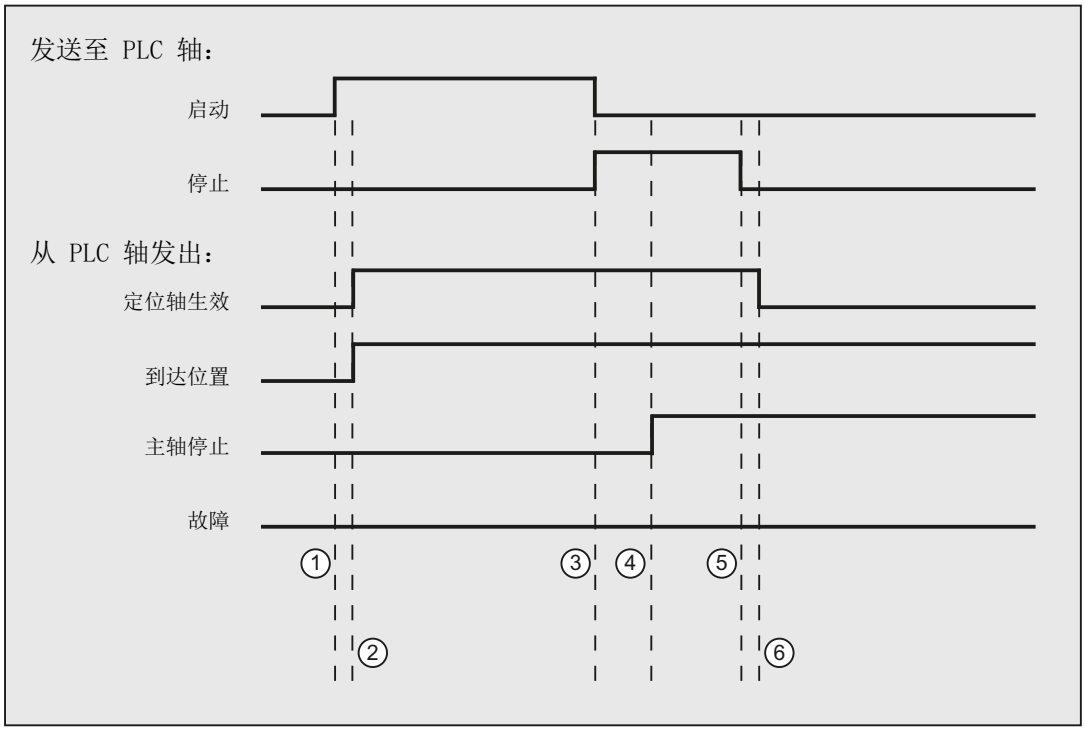
⑦ 用户程序中*停止*复位，*启动*在同一循环中错误地重新置位。这样*定位轴生效*不会被复位，而是会设置

⑧ 信号*轴无法启动*（故障编号 106）。

15.6.4.7 分度轴

DB380x	控制信号 PLC → NC	注释
DBX3002.0	增量运行	同一时间始终只能置位其中一个信号。 所有信号 0：绝对定位
DBX3002.1	以最短行程运行	
DBX3002.5	旋转方向同 M4	
DBX3003.0	绝对定位，负方向	
DBX3003.1	绝对定位，正方向	
DBX3003.7	分度位置	分度轴 ON
DBD3004	设定位置/设定行程	设定行程，当 DBX3002.0 == 1 时
DBD3008	进给速度	0：以 MD32060 $MA_POS_AX_VELO 中的值运行
DBX3000.7	启动定位轴	**提示**复位信号不会导致停止！
DBX2.2	删除剩余行程，主轴复位	终止信号，用于结束功能

DB390x	状态信号 NC → PLC	注释
DBX3000.7	定位轴生效	倍率 = 0 或到达设定位置时置 1。
DBX3000.6	到达位置	1：通过“精准停”到达设定位置。
DBX3000.1	故障	1：运行时出现故障，故障编号在 DBB3003 中分析！
DBB3003	故障编号	-
DBX1.4	进给轴/主轴停止	-



① 用户通过*启动*的上升沿触发功能。

提示：仅在*定位轴生效*信号复位时才可执行！

② *定位轴生效*和*到达位置*复位，*到达位置*此处不相关。

③ 用户复位*启动*并设置*停止*，从而停止主轴往复。

④ 主轴停止，且信号*主轴停止*置位。

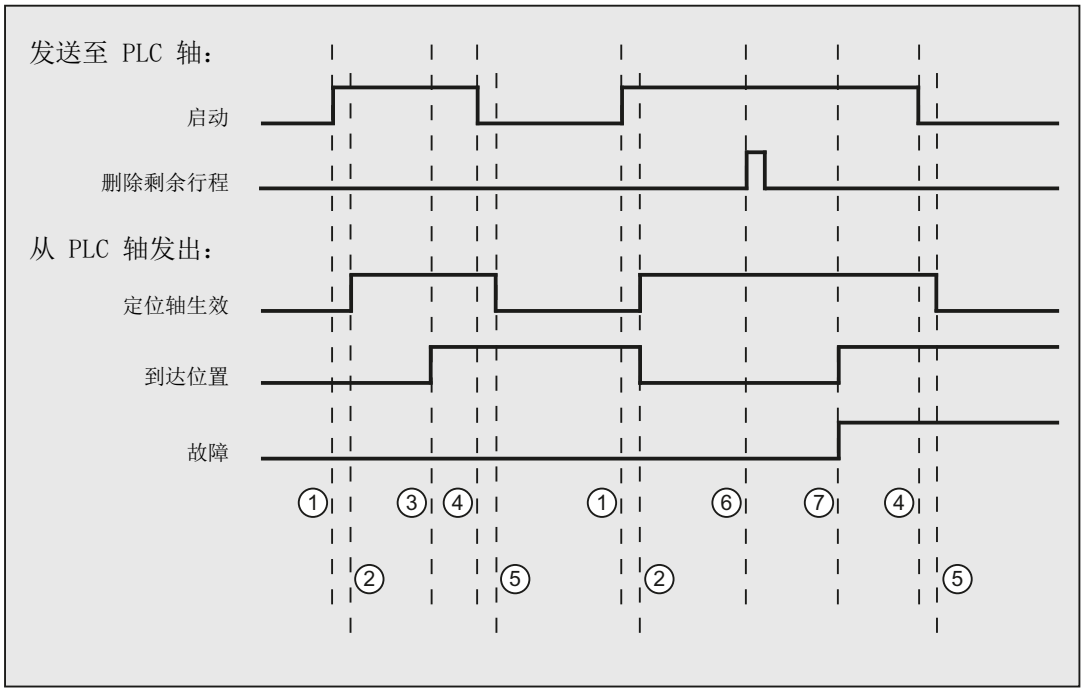
⑤ 之后用户复位*停止*。

⑥ *停止*的复位会引起*定位轴生效*的复位。

15.6.4.8 定位轴（公制）

DB380x	控制信号 PLC → NC	注释
DBX3002.0	增量运行	同一时间始终只能置位其中一个信号。 所有信号 0：绝对定位
DBX3002.1	以最短行程运行	
DBX3002.5	旋转方向同 M4	
DBX3003.0	绝对定位，负方向	
DBX3003.1	绝对定位，正方向	
DBD3002.2	英制运行尺寸	0：公制运行尺寸
DBX3002.3	手轮叠加	0：叠加 OFF
DBD3004	设定位置/设定行程	设定行程，当 DBX3002.0 == 1 时
DBD3008	进给速度	0：以 MD32060 $MA_POS_AX_VELO 中的值运行
DBX3000.7	启动定位轴	**提示**复位信号不会导致停止！
DBX2.2	删除剩余行程，主轴复位	终止信号，用于结束功能

DB390x	状态信号 NC → PLC	注释
DBX3000.7	定位轴生效	倍率 = 0 或到达设定位置时置 1。
DBX3000.6	到达位置	1：轴已到达设定位置。
DBX3000.1	故障	1：运行时出现故障，故障编号在 DBB3003 中分析！
DBB3003	故障编号	-



① 先通过*启动*的上升沿触发功能

② *定位轴生效* = 1 显示该功能生效且输出信号有效，*到达位置*和*轴停止*同样会被复位。

③ 积极应答*到达位置*= 1 且 *定位轴生效* = 1

④ 收到应答后复位功能启动

⑤ 通过功能进行信号转换

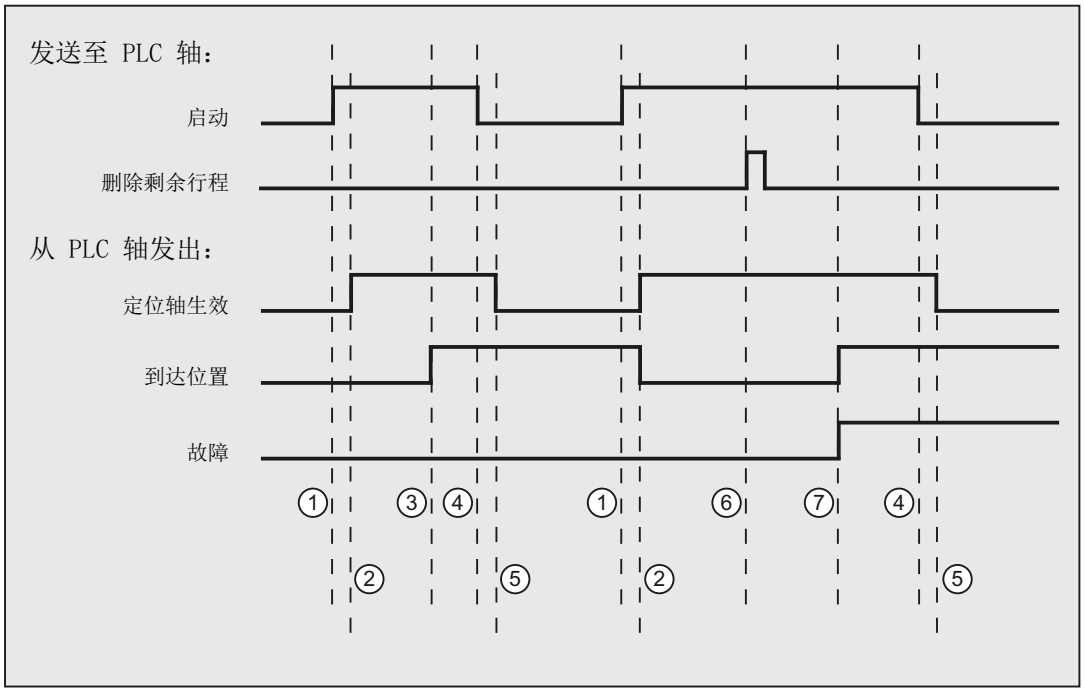
⑥ 定位通过删除剩余行程终止，信号持续时间至少为 1 PLC 循环。

⑦ *到达位置*和*故障*信号置位，可读取*故障编号*（此情形下为 30）。

15.6.4.9 定位轴（英制）

DB380x	控制信号 PLC → NC	注释
DBX3002.0	增量运行	同一时间始终只能置位其中一个信号。 所有信号 0：绝对定位
DBX3002.1	以最短行程运行	
DBX3002.5	旋转方向同 M4	
DBX3003.0	绝对定位，负方向	
DBX3003.1	绝对定位，正方向	
DBX3002.2	英制运行尺寸	0：公制运行尺寸
DBX3002.3	手轮叠加	0：叠加 OFF
DBD3004	设定位置/设定行程	设定行程，当 DBX3002.0 == 1 时
DBD3008	进给速度	0：以 MD32060 $MA_POS_AX_VELO 中的值运行
DBX3000.7	启动定位轴	**提示**复位信号不会导致停止！
DBX2.2	删除剩余行程，主轴复位	终止信号，用于结束功能

DB390x	状态信号 NC → PLC	注释
DBX3000.7	定位轴生效	倍率 = 0 或到达设定位置时置 1。
DBX3000.6	到达位置	1：轴已到达设定位置。
DBX3000.1	故障	1：运行时出现故障，故障编号在 DBB3003 中分析！
DBB3003	故障编号	-



① 先通过*启动*的上升沿触发功能

② *定位轴生效* = 1 显示该功能生效且输出信号有效，*到达位置*和*轴停止*同样会被复位。

③ 积极应答*到达位置*= 1 且 *定位轴生效* = 1

④ 收到应答后复位功能启动

⑤ 通过功能进行信号转换

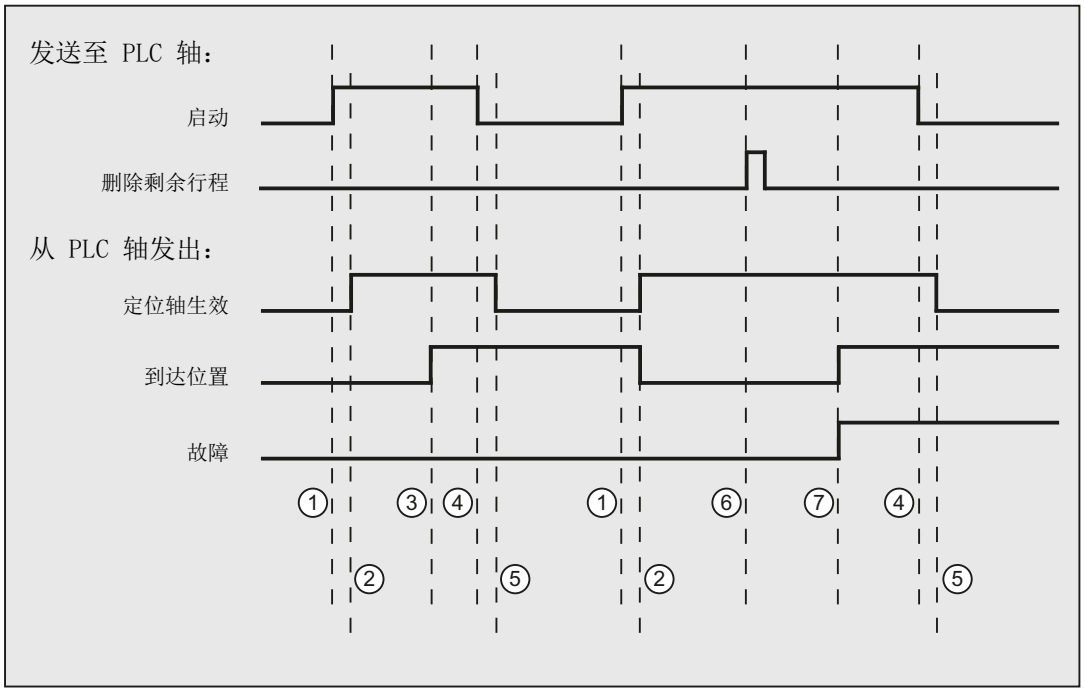
⑥ 定位通过删除剩余行程终止，信号持续时间至少为 1 PLC 循环。

⑦ *到达位置*和*故障*信号置位，可读取*故障编号*（此情形下为 30）。

15.6.4.10 带手轮叠加的公制定位轴

DB380x	控制信号 PLC → NC	注释
DBX3002.0	增量运行	同一时间始终只能置位其中一个信号。 所有信号 0：绝对定位
DBX3002.1	以最短行程运行	
DBX3002.5	旋转方向同 M4	
DBX3003.0	绝对定位，负方向	
DBX3003.1	绝对定位，正方向	
DBX3002.2	英制运行尺寸	0：公制运行尺寸
DBX3002.3	手轮叠加	0：叠加 OFF
DBD3004	设定位置/设定行程	设定行程，当 DBX3002.0 == 1 时
DBD3008	进给速度	0：以 MD32060 $MA_POS_AX_VELO 中的值运行
DBX3000.7	启动定位轴	**提示**复位信号不会导致停止！
DBX2.2	删除剩余行程，主轴复位	终止信号，用于结束功能

DB390x	状态信号 NC → PLC	注释
DBX3000.7	定位轴生效	倍率 = 0 或到达设定位置时置 1。
DBX3000.6	到达位置	1：轴已到达设定位置。
DBX3000.1	故障	1：运行时出现故障，故障编号在 DBB3003 中分析！
DBB3003	故障编号	-



① 先通过*启动*的上升沿触发功能

② *定位轴生效* = 1 显示该功能生效且输出信号有效，*到达位置*和*轴停止*同样会被复位。

③ 积极应答*到达位置*= 1 且 *定位轴生效* = 1

④ 收到应答后复位功能启动

⑤ 通过功能进行信号转换

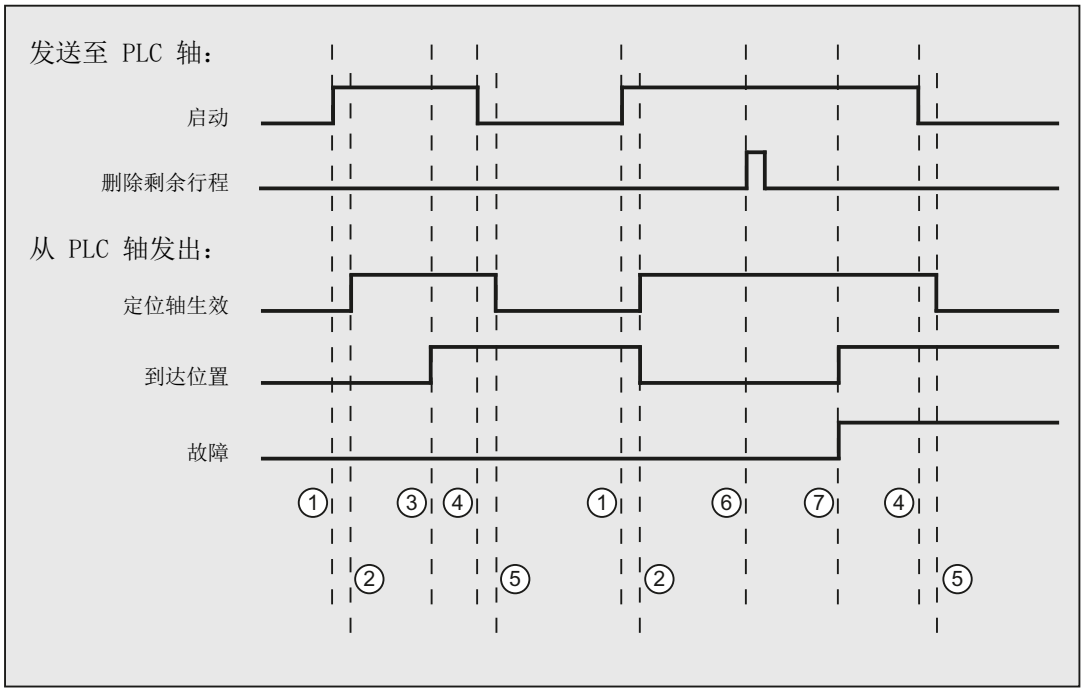
⑥ 定位通过删除剩余行程终止，信号持续时间至少为 1 PLC 循环。

⑦ *到达位置*和*故障*信号置位，可读取*故障编号*（此情形下为 30）。

15.6.4.11 带手轮叠加的英制定位轴

DB380x	控制信号 PLC → NC	注释
DBX3002.0	增量运行	同一时间始终只能置位其中一个信号。 所有信号 0：绝对定位
DBX3002.1	以最短行程运行	
DBX3002.5	旋转方向同 M4	
DBX3003.0	绝对定位，负方向	
DBX3003.1	绝对定位，正方向	
DBX3002.2	英制运行尺寸	0：公制运行尺寸
DBX3002.3	手轮叠加	0：叠加 OFF
DBD3004	设定位置/设定行程	设定行程，当 DBX3002.0 == 1 时
DBD3008	进给速度	0：以 MD32060 $MA_POS_AX_VELO 中的值运行
DBX3000.7	启动定位轴	**提示**复位信号不会导致停止！
DBX2.2	删除剩余行程，主轴复位	终止信号，用于结束功能

DB390x	状态信号 NC → PLC	注释
DBX3000.7	定位轴生效	倍率 = 0 或到达设定位置时置 1。
DBX3000.6	到达位置	1：轴已到达设定位置。
DBX3000.1	故障	1：运行时出现故障，故障编号在 DBB3003 中分析！
DBB3003	故障编号	-



① 先通过*启动*的上升沿触发功能

② *定位轴生效* = 1 显示该功能生效且输出信号有效，*到达位置*和*轴停止*同样会被复位。

③ 积极应答*到达位置*= 1 且 *定位轴生效* = 1

④ 收到应答后复位功能启动

⑤ 通过功能进行信号转换

⑥ 定位通过删除剩余行程终止，信号持续时间至少为 1 PLC 循环。

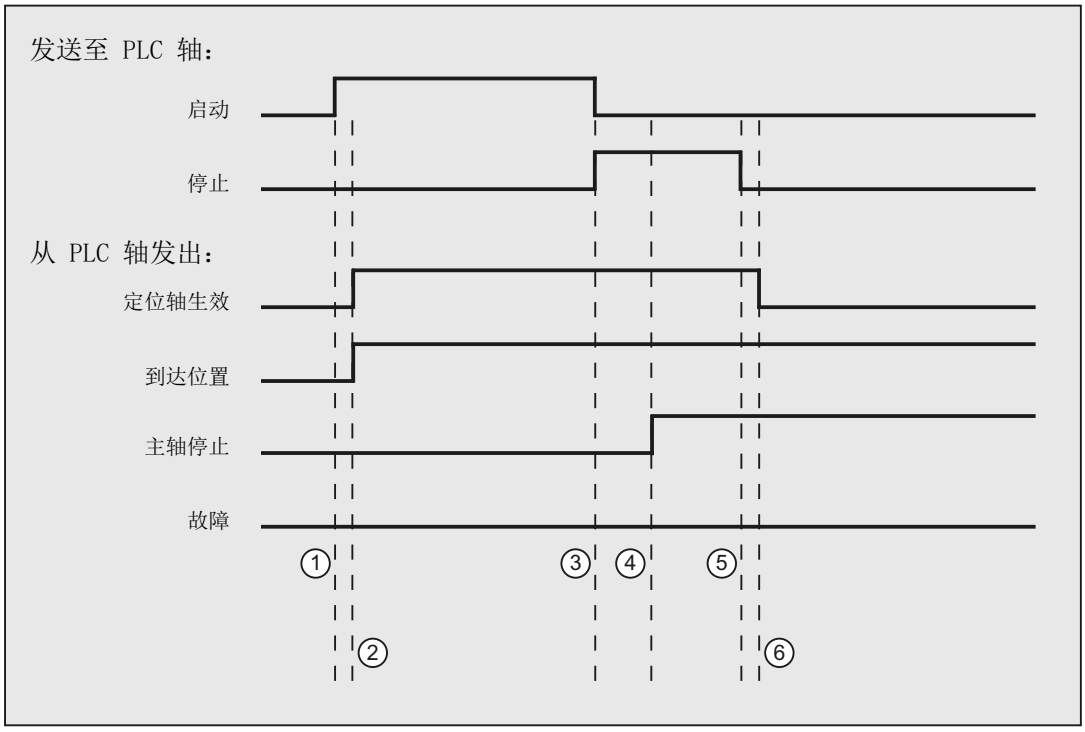
⑦ *到达位置*和*故障*信号置位，可读取*故障编号*（此情形下为 30）。

15.6.4.12 主轴旋转与自动齿轮档选择

DB380x	控制信号 PLC → NC	注释
DBX3002.0	增量运行	-
DBX3002.1	以最短行程运行	-
DBX3002.5	旋转方向同 M4	1：旋转方向已通过 M4 设定 0：旋转方向已通过 M3 设定
DBX3003.0	绝对定位，负方向	-

DB380x	控制信号 PLC → NC	注释
DBX3003.1	绝对定位，正方向	-
DBX3002.7	自动齿轮档选择	1：自动齿轮档选择 ON
DBD3008	进给速度	主轴转速
DBX3000.5	启动主轴旋转	-
DBX3001.5	停止主轴旋转	-

DB390x	状态信号 NC → PLC	注释
DBX3000.7	定位轴生效	1：启动或停止 == 1，
DBX3000.6	到达位置	1：设定转速已输出
DBX3000.1	故障	1：运行时出现故障，故障编号在 DBB3003 中分析！
DBB3003	故障编号	-
DBX1.4	进给轴/主轴停止	-



① 用户通过*启动*的上升沿触发功能。

② *定位轴生效*和*到达位置*复位，*到达位置*此处不相关。

③ 用户复位*启动*并设置*停止*，从而停止主轴旋转。

④ 主轴停止，且信号*主轴停止*置位。

⑤ 之后用户复位*停止*。

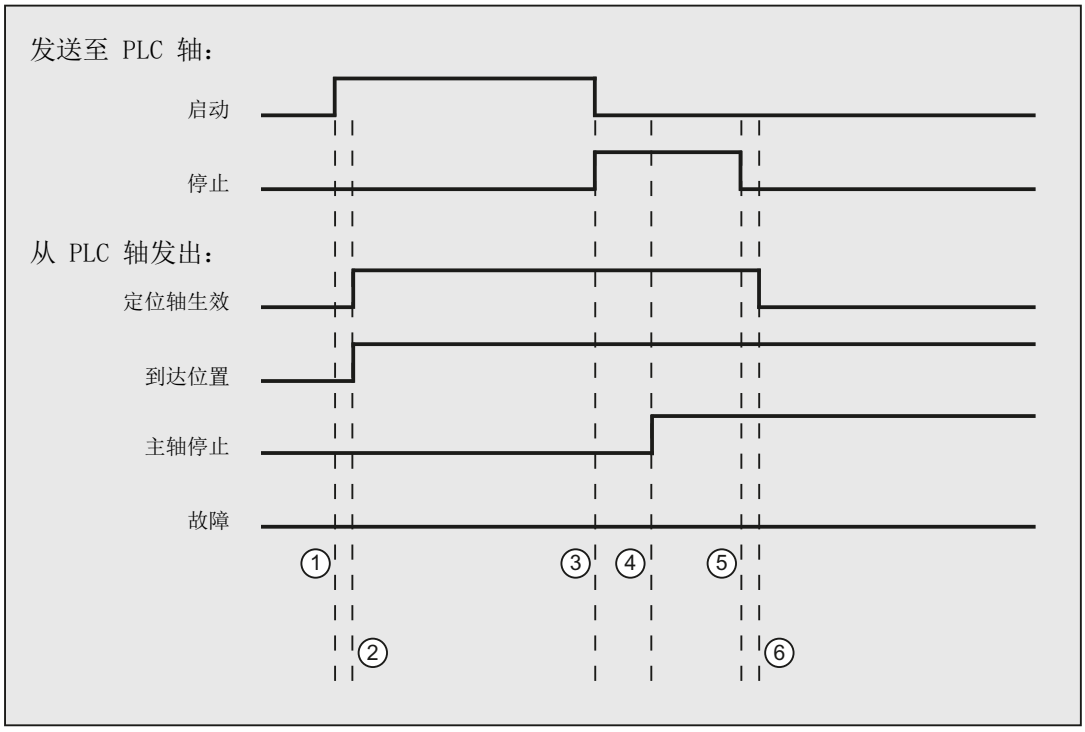
⑥ *停止*的复位会引起*定位轴生效*的复位。

15.6.4.13 通过恒定切削速度 [m/min] 旋转主轴

DB380x	控制信号 PLC → NC	注释
DBX3002.0	增量运行	-
DBX3002.1	以最短行程运行	-
DBX3002.5	旋转方向同 M4	1：旋转方向已通过 M4 设定 0：旋转方向已通过 M3 设定
DBX3003.0	绝对定位，负方向	-

DB380x	控制信号 PLC → NC	注释
DBX3003.1	绝对定位，正方向	-
DBX3002.2	英制运行尺寸	0 公制运行尺寸
DBX3002.6	恒定切削速度	1 恒定切削速度 “ON”
DBD3008	进给速度	主轴转速
DBX3000.5	启动主轴旋转	-
DBX3001.5	停止主轴旋转	-

DB390x	状态信号 NC → PLC	注释
DBX3000.7	定位轴生效	1：启动或停止 == 1，
DBX3000.6	到达位置	1：设定转速已输出
DBX3000.1	故障	1：运行时出现故障，故障编号在 DBB3003 中分析！
DBB3003	故障编号	-
DBX1.4	进给轴/主轴停止	-



① 用户通过*启动*的上升沿触发功能。

② *定位轴生效*和*到达位置*复位，*到达位置*此处不相关。

③ 用户复位*启动*并设置*停止*，从而停止主轴旋转。

④ 主轴停止，且信号*主轴停止*置位。

⑤ 之后用户复位*停止*。

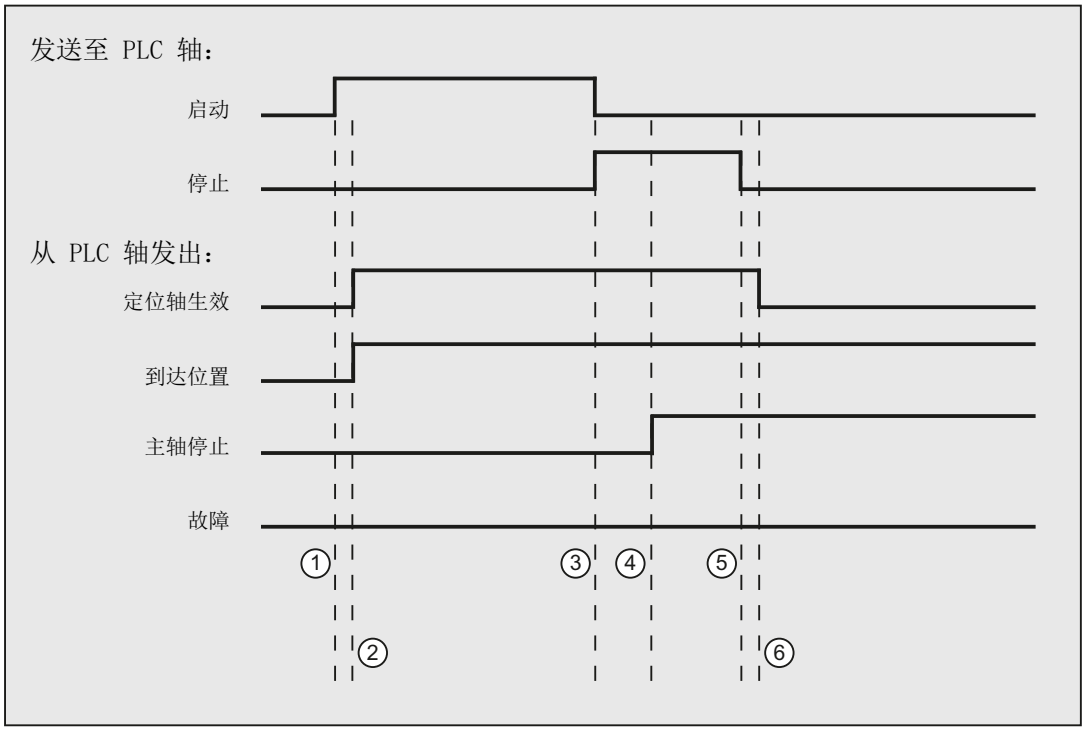
⑥ *停止*的复位会引起*定位轴生效*的复位。

15.6.4.14 通过恒定切削速度 [feet/min] 旋转主轴

DB380x	控制信号 PLC → NC	注释
DBX3002.0	增量运行	-
DBX3002.1	以最短行程运行	-
DBX3002.5	旋转方向同 M4	1：旋转方向已通过 M4 设定 0：旋转方向已通过 M3 设定
DBX3003.0	绝对定位，负方向	-

DB380x	控制信号 PLC → NC	注释
DBX3003.1	绝对定位，正方向	-
DBX3002.2	英制运行尺寸	1：英制运行尺寸
DBX3002.6	恒定切削速度	1：恒定切削速度 ON
DBD3008	进给速度	主轴转速
DBX3000.5	启动主轴旋转	-
DBX3001.5	停止主轴旋转	-

DB390x	状态信号 NC → PLC	注释
DBX3000.7	定位轴生效	1：启动或停止 == 1
DBX3000.6	到达位置	1：设定转速已输出
DBX3000.1	故障	1：运行时出现故障，故障编号在 DBB3003 中分析！
DBB3003	故障编号	-
DBX1.4	进给轴/主轴停止	-



① 用户通过*启动*的上升沿触发功能。

② *定位轴生效*和*到达位置*复位，*到达位置*此处不相关。

③ 用户复位*启动*并设置*停止*，从而停止主轴旋转。

④ 主轴停止，且信号*主轴停止*置位。

⑤ 之后用户复位*停止*。

⑥ *停止*的复位会引起*定位轴生效*的复位。

15.6.4.15 故障消息

如果功能无法执行，应根据故障置位以下信号：

- DB390x .DBX3000.0 == 1（轴无法启动）
- DB390x.DBX3000.1 == 1（运行中的故障）

详细的故障原因通过以下内容显示：

- DB390x.DBB3003 (故障编号)

故障编号		含义
十进制	十六进制	
1	01	已同时激活多个轴/主轴功能
20	14	启动了一个功能，但未到达位置
30	1E	轴/主轴的控制权在运动结束前即已移交给 NC
40	28	通过 NC 程序对轴进行编程，NC 内部故障
50	32	固定分配的 PLC 轴：运行（JOG）或回参考点
60	3C	固定分配的 PLC 轴：通道状态不允许开始
100	64	为轴/主轴编写的位置错误（报警编号 1) 16830）
101	65	编写的转速过高
102	66	恒定切削速度的取值范围错误（报警编号 1) 14840）
104	68	副主轴:不允许的编程（报警编号 1) 22030）
105	69	无测量系统（报警编号 1) 16770）
106	6A	轴定位尚在生效（报警编号 1) 22052）
107	6B	未找到参考标记（报警编号 1) 22051）
108	6C	从转速闭环控制到位置闭环控制没有过渡（报警编号 1) 22050）
109	6D	未找到参考标记（报警编号 1) 22051）
110	6E	速度/转速为负
111	6F	转速设定值为零
112	70	无效的齿轮档
115	73	未到达编程的位置
117	75	在 NC 中 G96/G961 未生效
118	76	在 NC 中 G96/G961 仍有效
120	78	该轴不是分度轴（报警编号 1) 20072）
121	79	分度位置故障（报警编号 1) 17510）
125	7D	DC（最短行程）无法启用（报警编号 1) 16800）
126	7E	无法启用负绝对值（报警编号 1) 16820）
127	7F	无法启用正绝对值（报警编号 1) 16810）
128	80	无用于直径编程的平面轴（报警编号 1) 16510）
130	82	软件限位开关正（报警编号 1) 20070）
131	83	软件限位开关负（报警编号 1) 20070）

故障编号		含义
十进制	十六进制	
132	84	工作区域限制正（报警编号 1) 20071）
133	85	工作区域限制负（报警编号 1) 20071）
134	85	框架不可用于分度轴
135	87	配备“端面齿”的分度轴生效（报警编号 1) 17501）
136	88	配备“端面齿”的分度轴生效，且轴未回参考点（报警编号 1) 17503）
137	89	不可为经过转换的主轴/轴启动主轴模式（报警编号 1) 22290）
138	8A	针对轴超出了生效的坐标系专用工作区域正限制（报警编号 1) 20082）
139	8B	针对轴超出了生效的坐标系专用工作区域负限制（报警编号 1) 20082）
200	C8	系统报警编号 1) 450007
1）详细的报警说明参见：报警诊断手册；SINUMERIK 828D，SINAMICS S120		

15.6.5 启动 ASUB

15.6.5.1 任务启动

DB340x	ASUB：任务 [r/w] (PLC → NC)							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBB0	-	-	-	-	-	-	-	启动 INT1
DBB1	-	-	-	-	-	-	-	启动 INT2
DBB2	-	-	-	-	-	-	-	启动 INT3
DBB3	-	-	-	-	-	-	-	启动 INT4

中断

“启动 INT1”：

- DBX0.0 = 1：请求启动对应 INT1 的中断程序 (ASUB)。
- DBX0.0 = 0：结果接口 DB340x.DBB1000.0 - 3 中应答完成后，复位 ASUB 请求。

“启动 INT2”：

- DBX1.0 = 1：请求启动对应 INT2 的中断程序（ASUB）。
- DBX1.0 = 0：结果接口 DB340x.DBB1001.0 - 3 中应答完成后，复位 ASUB 请求。

“启动 INT3”：

- DBX2.0 = 1：请求启动对应 INT3 的中断程序 (ASUB)。
- DBX2.0 = 0：结果接口 DB340x.DBB1002.0 - 3 中应答完成后，复位 ASUB 请求。

“启动 INT4”：

- DBX3.0 = 1：请求启动对应 INT4 的中断程序（ASUB）。
- DBX3.0 = 0：结果接口 DB340x.DBB1003.0 - 3 中应答完成后，复位 ASUB 请求。

15.6.5.2 任务结果

DB340x	ASUB：结果 [r] (NC → PLC)							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBB1000	用于启动 INT1 的信号 (DB340x.DBX0.0)							
					ASUB 无法执行 1)	未分配中断编号 2)	执行 ASUP	ASUB 结束 3)
DBB1001	用于启动 INT2 的信号 (DB340x.DBX1.0)							
					ASUB 无法执行 1)	未分配中断编号 2)	执行 ASUP	ASUB 结束 3)
DBB1002	用于启动 INT3 的信号 (DB340x.DBX2.0)							
					ASUB 无法执行 1)	未分配中断编号 2)	ASUB 执行中	ASUB 结束 3)
DBB1003	用于启动 INT4 的信号 (DB340x.DBX3.0)							
					ASUB 无法执行 1)	未分配中断编号 2)	ASUB 执行中	ASUB 结束 3)

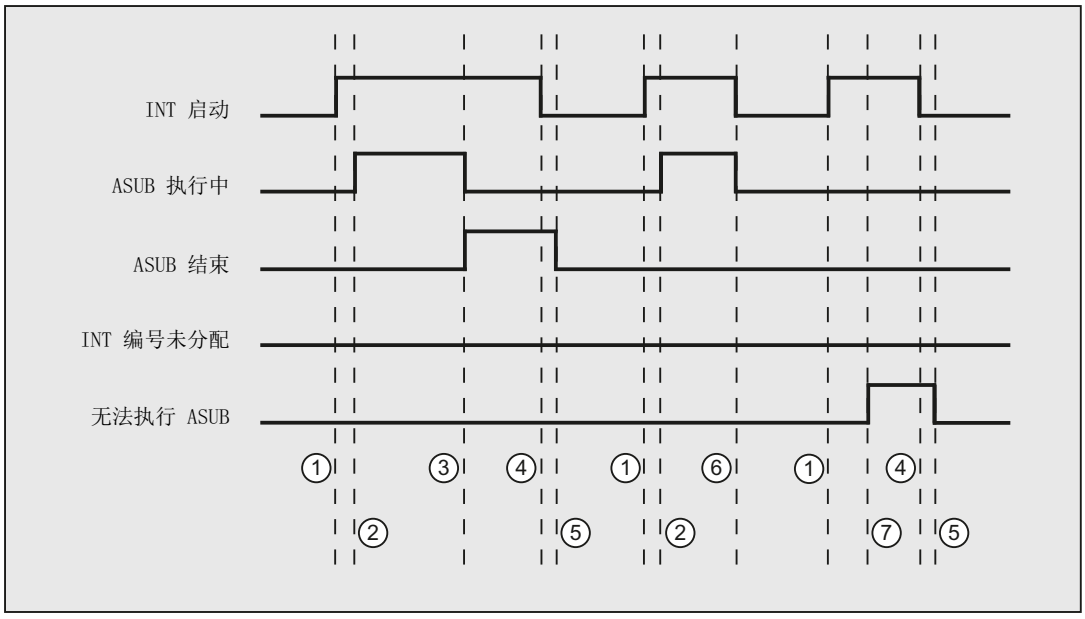
1) 消极应答：例如急停或通道复位请求时。

2) 消极应答：编号指定尚未进行。解决方法：执行 PI 服务 “ASUB”。

3) 积极应答：ASUP 成功完成。复位启动信号：DB340x.:DBXn.0 = 0，其中 n = 0, 1, 2, 3

15.6.5.3 信号流

信号流



① 用户通过*启动*的上升沿触发功能。

② 软件反馈 *ASUB 执行中*。

③ 应答 *ASUB 结束*显示成功执行，*ASUB 执行中*复位。

④ 用户在收到应答后复位功能触发信号。

⑤ 通过固件切换信号

⑥ 不允许！若接收到应答前功能触发被复位，那么输出信号不会更新，对触发功能的运行不会产生影响。

⑦ *ASUB 无法执行*：消极应答，出错。

图 15-2 示例：信号流

15.6.6 在 HMI 上执行通道选择

功能

HMI 上显示的通道可由 PLC 用户程序通过 HMI/PLC 接口在机床基本画面中选择。

前提条件

在 NC 中设置了多个通道。

任务和应答接口

DB1900	通道选择 [r/w] (PLC ↔ HMI)							
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBB5011	故障标识（见下） (HMI → PLC)							
...	...							
DBB5021	状态 (HMI → PLC)	请求 (PLC ↔ HMI)	功能编号 (PLC → HMI)					
DBB5022	通道编号（见下） (PLC → HMI)							

通道编号

- 通道编号：1, 2, ... 最大通道数
- 下一个通道：FF_H

故障标识

- 0：无故障
- 1：故障号无效 (DBX32.0 - .5)
- 2：参数无效 (DBB33 - DBB35)
- 3：HMI 内部变量的写入异常中断
- 10：无通道（DBB33）

功能流程

PLC → HMI

PLC 用户程序必须遵循以下流程：

1. 检查是否有空余接口用于新任务：
   - DB1900.DBX**5021**.6 == 0（功能请求）
   - DB1900.DBX**5021**.7 == 0（状态）
2. 如果接口未占用，则须输入任务数据并置位功能请求：
   - DB1900.DBB**5022** = <通道编号>
   - DB1900.DBX**5021**.0 - .5 = 1（功能编号：通道选择）
   - DB1900.DBX**5021**.6 = 1（功能请求）

HMI → PLC

如果参数设置**正确**，HMI 会作出以下响应：

1. HMI 识别出通道选择的功能请求后，会将状态设为“执行功能”并复位功能请求：
   - DB1900.DBX**5021**.7 = 1（状态）
   - DB1900.DBX**5021**.6 = 0（功能请求）
2. 执行完通道选择后，HMI 会再次复位状态并设置值 0 作为故障标识：
   - DB1900.DBX**5021**.7 = 0（状态）
   - DB1900.DBX**5011** = 0（故障标识）

如果参数设置**错误**，HMI 会作出以下响应：

- 复位功能请求并设置相应的故障标识：
  - DB1900.DBX**5021**.6 = 0（功能请求）
  - DB1900.DBX**5021**.7 = 0（状态）
  - DB1900.DBX**5011** = <故障标识>

15.7 数控禁用功能（选件）

15.7.1 功能

说明

“数控禁用功能”是一个需要购买许可证的选件（产品编号：6FC5800-0AP76-0YB0）。使用数控禁用功能必须在西门子购买相应的许可证。通过试用许可证**无法**使用禁用功能。

注意
许可证 数控禁用功能的调试人员（机床制造商或代理商）必须具有该选件 (CoL) 的许可证！ 在忘记了 PIN 的情况下，借助该许可证仍可得到西门子的授权。许可证（CoL）的拥有者可以取消机床的禁用。

通过“数控禁用功能”，机床制造商可借助其通过 SINUMERIK Integrate 应用程序 Access MyMachine (AMM) 创建的加密文件激活控制系统中的禁用日期。这样便可将设备的使用期限限制在禁用日期内。禁用日期届满后，控制系统的 NC 启动功能被禁用。

通过另一个加密文件可以延长或取消数控禁用功能。该文件由机床制造商负责交给最终用户。

15.7.2 前提条件

使用“数控禁用功能”必须满足下列条件：

- 必须设置选件“数控禁用功能”。
- 必须使用 CPU 型号为 “828D Step 2 x.yy” 的 PLC 程序。
  兼容模式必须关闭。
- 必须装有 SINUMERIK Integrate 应用程序 Access MyMachine/ P2P (PC)。

15.7.3 限制

数控禁用功能支持有使用期限的商业模式。以防止在超出规定时间后未经授权使用。直接访问数控系统便可实现功能操作。数控禁用功能不具备绝对的防篡改保护功能。通过关闭数控系统的自动运行功能防止未经授权使用。由于此时不会中断正在运行的自动程序，因此，可导致运行时间超出禁用日期。SINUMERIK 控制系统的所有其他功能仍然可用。

只有机床制造商提供支持，数控禁用功能才能发挥其作用。使用数控禁用功能时，必须遵循以下说明和边界条件！

说明

数控禁用功能是以连接了 PLC 项目的 SINUMERIK 控制系统为基础的。SINUMERIK 控制系统由 PPU 和装有系统软件的 CF 卡组合而成。

边界条件

- 数控禁用功能可因篡改尝试或冲突而导致机床停机。
- 使用数控禁用功能时，客户可能还需要购买机床制造商或代理商的附加维修项目。
- 提供给客户的 PLC 项目必须存有 OEM-PIN。使用 “自由” PLC 项目可实现对功能的操作！
- 重新执行相应机床的 PLC 项目可实现对功能的操作！
- 必须对 SINUMERIK PLC 中的程序组织单元进行保护。可在 PLC 编程工具中进行激活。必须对 PLC 项目的编码进行加密。
- 必须为每台机床单独给指定一个 OEM-PIN，以实现更好的保护。
- 首次设置禁用功能时（参见章节“数控禁用功能的初次调试 (页 1335)”）调试人员必须正确设置 SINUMERIK 控制系统中的日期和时间。如果日期已经错过，则机床会根据与实际日期的差值延长运行时间。

- 数控禁用功能是以 SINUMERIK 828D 的实时时钟为基础的。由于 SINUMERIK 828D 采用免维护结构，因此，时间可能会丢失。数控禁用功能会执行时间的合理性检测。该检测可能会受到实时时钟的电压损耗的影响。此时不会算入掉电时间。
- 数控禁用功能的调试人员具有相应的许可证 (CoL)。证明其为西门子指定的功能调试人员，具备相应的操作技能和资质。
- 软件误操作可导致控制系统被意外禁用。

15.7.4 防止篡改

数控禁用功能用于确保只能在设置的时间范围内使用数控控制系统。尽管现有的保护机制能够防止数控禁用功能被篡改，但仍存在漏洞和风险。数控禁用功能会周期性检查 PPU、CF 卡和 PLC 项目中所安装的组合。所有三个组件更换时，禁用功能取消。为了防止篡改，必须强制遵守章节“限制 (页 1334)”中的边界条件。

说明

数控禁用功能采用的是密码运行。所采用的运行符合首次投放市场时的技术状态。随着时间的流逝，运行可能逐渐普及。

15.7.5 数控禁用功能的初次调试

数控禁用功能的初次调试结合了控制系统硬件、PPU、CF 卡上的系统软件和属于机床的 PLC 项目。同时设置了首个禁用日期。

调试禁用功能需要创建一个与硬件相匹配的加密 Lockset 文件 (lockset.clc) 。为此，可借助 SINUMERIK Integrate 应用程序 Access MyMachine (AMM) 进行创建。

创建 Lockset 文件

创建 Lockset 文件需要以下数据：

- CF 卡的序列号
- 控制系统 (PPU) 的序列号
- OEM-PIN
- 禁用日期

说明

CF 卡和控制系统 (PPU) 的**序列号**位于 SINUMERIK Operate 操作界面上的“诊断” > “版本” > “硬件” > “NCU/PLC” > “ncu1” 操作区中：

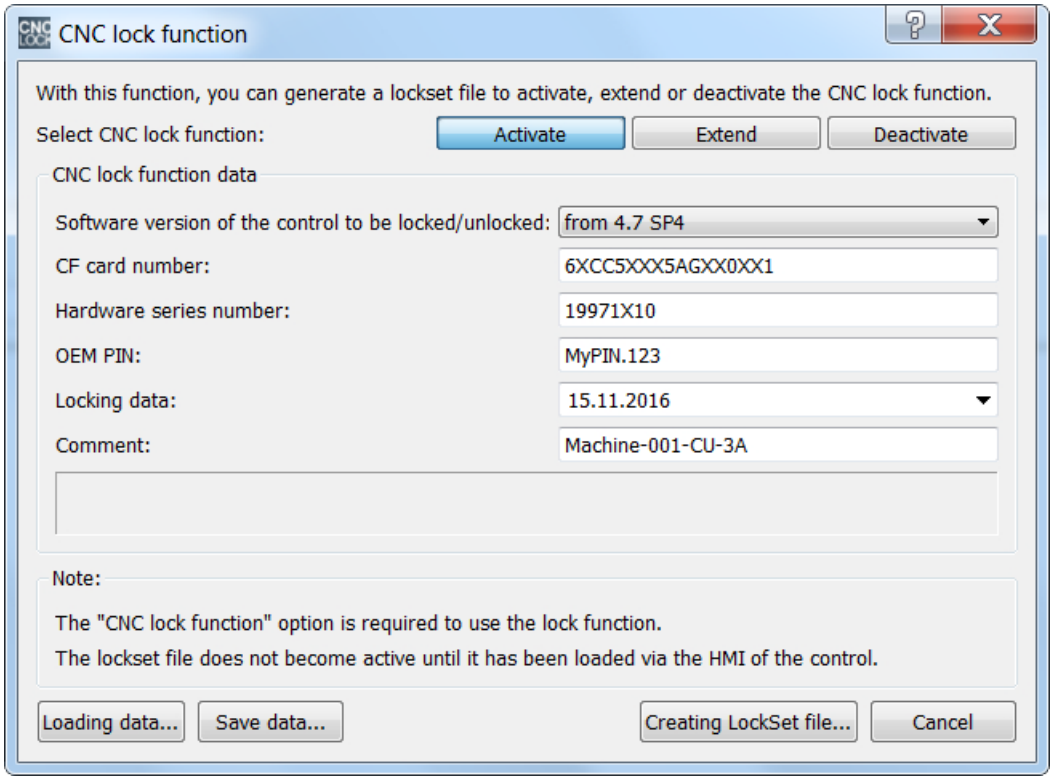
- **CF 卡**：“CF 卡” > “序列号” 区域
- **PPU**：“SINUMERIK 828D PPU” > “序列号” 区域

说明

OEM-PIN 用于提高防止数控禁用功能被篡改的安全性。

OEM-PIN 由系统在激活数控禁用功能时保存到 PLC 用户程序中。OEM-PIN 无法由用户在 PLC 用户程序中查看、修改或删除。

通过 Access MyMachine 在对话框“数控禁用功能”中输入数据（通过主菜单：“其他”>“数控禁用功能”调用）。“**激活**”按钮必须已选择。



在输入所需所有数据后，通过“创建 Lockset 文件...”按钮创建用于激活的 Lockset 文件。

传输 Lockset 文件

之后必须通过以太网连接直接或通过存储介质（例如：USB 闪存）将 Lockset 文件传输至控制系统。

文件位于：/System-CF-Card/User/sinumerik/data/license。

读取 Lockset 文件

读取 Lockset 文件前，机床制造商必须正确设置控制系统的时间，因为在激活数控禁用功能的时间点上的时间是保存为监控的起始值的。之后可通过操作界面启动读取激活文件：操作区“调试” > “许可证” > “读入许可证密钥”。

读取无需访问权限。

如果在读取 Lockset 文件时未出错，则数控禁用功能在控制系统中生效。

说明

如果在读取 Lockset 文件时出错，系统会显示一个故障报警。数控禁用功能的状态保持不变。

说明

建议在调试机床和激活数控禁用功能后由机床制造商创建一个含所有控制系统组件的完整**调试存档**。这样就能确保数控禁用功能数据的一致性。必要时，可借助于该调试存档重新调试控制系统，而不需要维修就能重新激活数控禁用功能。

检查禁用日期

激活数控禁用功能后，操作界面上便会显示记录在 Lockset 文件中的禁用日期：

SINUMERIK Operate 操作界面：操作区“调试” > “许可证”

设置预警时间

预警时间是指达到禁用日期（自该禁用日期起报警 8063 每天显示一次）前的时间区间。该报警表示即将达到禁用日期且控制系统的 NC 启动功能被禁用。预警时间通过以下机床数据设置：

MD17300 $MN_CNC_LOCK_WARNING_TIME

重要说明

正确并安全地使用数控禁用功能必须遵循以下说明！

说明

在激活数控禁用功能前必须正确设置控制系统上的**时间**。

说明

必须为 PLC 用户程序设置 **POE 密码保护**。防止用户复制专有技术并用在自有 PLC 用户程序中，不同于机床制造商的 PLC 用户程序，该 PLC 用户程序含有数控禁用功能的 PLC 密钥，可进行更换。

说明

只有激活了数控禁用功能**后**，才可读取最终用户的**调试文档**。

说明

取消数控禁用功能（参见“取消数控禁用功能 (页 1340)”）后，必须读取一个**新调试文档**并更换原始文档。只有这样才能确保之后不会意外重新激活数控禁用功能。否则会触发报警并禁用 NC 启动。如需撤销该操作，必须重新读取用于取消数控禁用功能的 Lockset 文件。

说明

机床制造商须承担正确并安全地使用数控禁用功能的责任。

15.7.6 延长数控禁用功能

为延长数控禁用功能，机床制造商必须通过 Access MyMachine（AMM）创建一个**含新的禁用日期**的 Lockset 文件（lockset.clc）。

创建 Lockset 文件

创建 Lockset 文件需要以下数据：

- CF 卡的序列号
- 控制系统 (PPU) 的序列号
- OEM-PIN
- **新的禁用日期**

说明

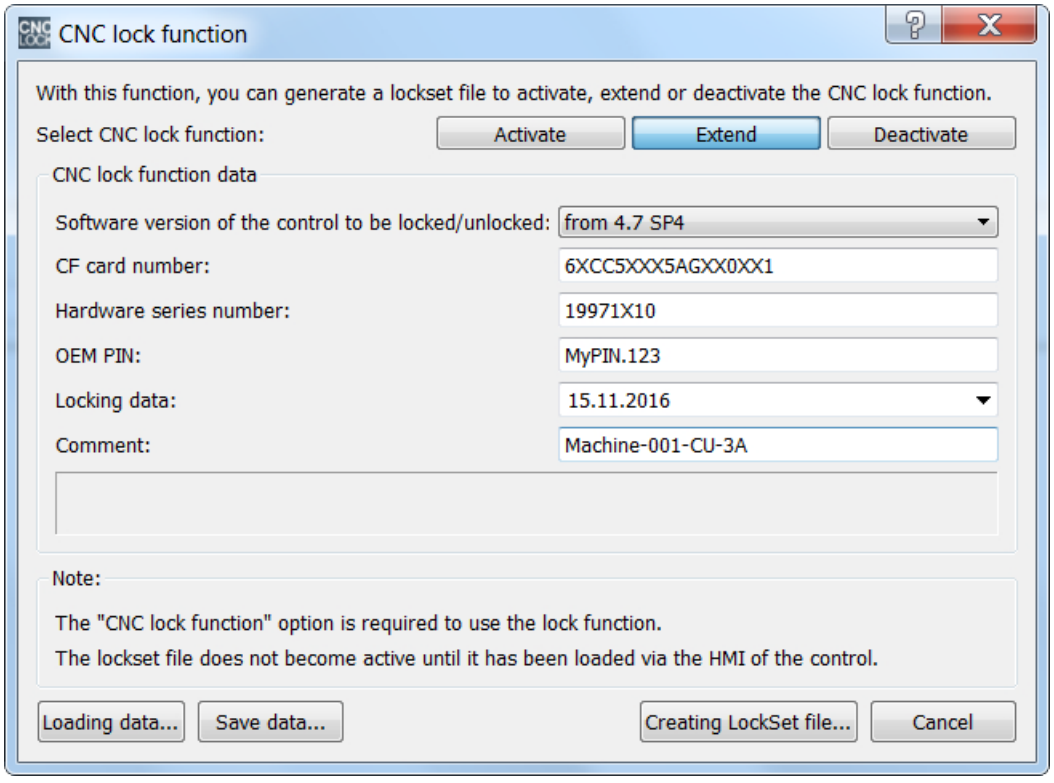
CF 卡和控制系统 (PPU) 的序列号及 OEM-PIN 必须与初次调试时用于激活数控禁用功能的值相一致。

说明

CF 卡和控制系统 (PPU) 的**序列号**位于 SINUMERIK Operate 操作界面上的“诊断” > “版本” > “硬件” > “NCU/PLC” > “ncu1” 操作区中：

- **CF 卡**：“CF 卡” > “序列号” 区域
- **PPU**：“SINUMERIK 828D PPU” > “序列号” 区域

通过 Access MyMachine 在对话框“数控禁用功能”中输入数据（通过主菜单：“其他”>“数控禁用功能”调用）。“**延长**”按钮必须已选择。



在输入所需所有数据后，通过“创建 Lockset 文件...”按钮创建用于延长的 Lockset 文件。

传输 Lockset 文件

必须将新的 Lockset 文件传输至控制系统。此时，可由机床制造商通过以太网连接直接传输至控制系统。

文件位于：/System-CF-Card/User/sinumerik/data/license。

或者将新的 Lockset 文件发送给最终用户并由最终用户将其传输至控制系统。

读取 Lockset 文件

通过操作界面启动对 Lockset 文件的读取：操作区“调试” > “许可证” > “读入许可证密钥”。

读取无需访问权限。

如果在读取 Lockset 文件时未出错，则配置了新禁用日期的数控禁用功能在控制系统中生效。

说明

如果在读取 Lockset 文件时出错，系统会显示一个故障报警。数控禁用功能的状态保持不变。

检查修改的禁用日期

为确保禁用日期修改成功，可在操作界面上进行检查：

SINUMERIK Operate 操作界面：操作区“调试” > “许可证”

15.7.7 取消数控禁用功能

为取消数控禁用功能，机床制造商必须通过 Access MyMachine（AMM）创建一个**不含禁用日期**的 Lockset 文件（lockset.clc）。

创建 Lockset 文件

创建 Lockset 文件需要以下数据：

- CF 卡的序列号
- 控制系统 (PPU) 的序列号
- OEM-PIN

说明

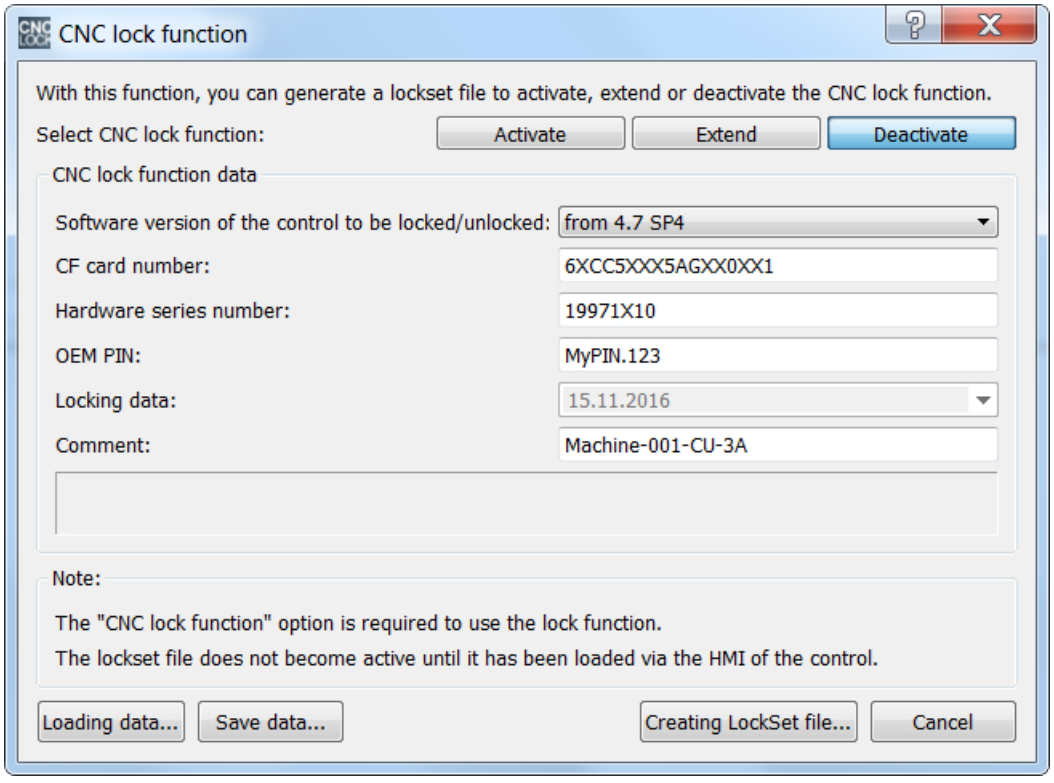
CF 卡和控制系统 (PPU) 的序列号及 OEM-PIN 必须与初次调试时用于激活数控禁用功能的值相一致。

说明

CF 卡和控制系统 (PPU) 的**序列号**位于 SINUMERIK Operate 操作界面上的“诊断” > “版本” > “硬件” > “NCU/PLC” > “ncu1” 操作区中：

- **CF 卡**：“CF 卡” > “序列号” 区域
- **PPU**：“SINUMERIK 828D PPU” > “序列号” 区域

通过 Access MyMachine 在对话框“数控禁用功能”中输入数据（通过主菜单：“其他”>“数控禁用功能”调用）。“**取消**”按钮必须已选择。



在输入所需所有数据后，通过“创建 Lockset 文件...”按钮创建用于取消的 Lockset 文件。

传输 Lockset 文件

必须将新的 Lockset 文件传输至控制系统。此时，可由机床制造商通过以太网连接直接传输至控制系统。

文件位于：/System-CF-Card/User/sinumerik/data/license。

或者将新的 Lockset 文件发送给最终用户并由最终用户将其传输至控制系统。

读取 Lockset 文件

通过操作界面启动对 Lockset 文件的读取：操作区“调试” > “许可证” > “读入许可证密钥”。

读取无需访问权限。

如果在读取 Lockset 文件时未出错，系统会取消数控禁用功能。

说明

如果在读取 Lockset 文件时出错，系统会显示一个故障报警。数控禁用功能的状态保持不变。

说明

建议**在取消**禁用日期后由最终用户创建一个含所有控制系统组件的完整**调试存档**。必要时，可借助于该调试存档重新调试控制系统，而不需要再次激活数控禁用功能。

检查

如果操作界面上不再显示禁用日期，则表示禁用日期取消成功：

SINUMERIK Operate 操作界面：操作区“调试” > “许可证”

15.7.8 更换损坏的控制系统硬件 (PPU)

在更换损坏的控制系统硬件 (PPU) 时，必须保留机床上的系统 CF 卡，以确保其可在新的控制系统硬件中使用。

在初次启动装配了新硬件的控制系统后，必须读取通过之前的控制系统硬件创建的调试存档。下一次启动控制系统后，系统会显示报警 8062：“数控禁用功能：功能运行异常：原因 2”（更换了硬件）并禁用 NC 启动。报警原因在于新控制系统硬件的新序列号。

请求新 Lockset 文件

为重新解禁控制系统，最终用户需要向数控禁用功能的调试人员（机床制造商或代理商）请求与当前控制系统匹配的新 Lockset 文件（lockset.clc）。

创建 Lockset 文件

创建 Lockset 文件需要以下数据：

- 保留在机床上的 CF 卡的序列号
- **新**控制系统 (PPU) 的序列号

- 初始 OEM-PIN
- 最后的禁用日期或者新的禁用日期

说明

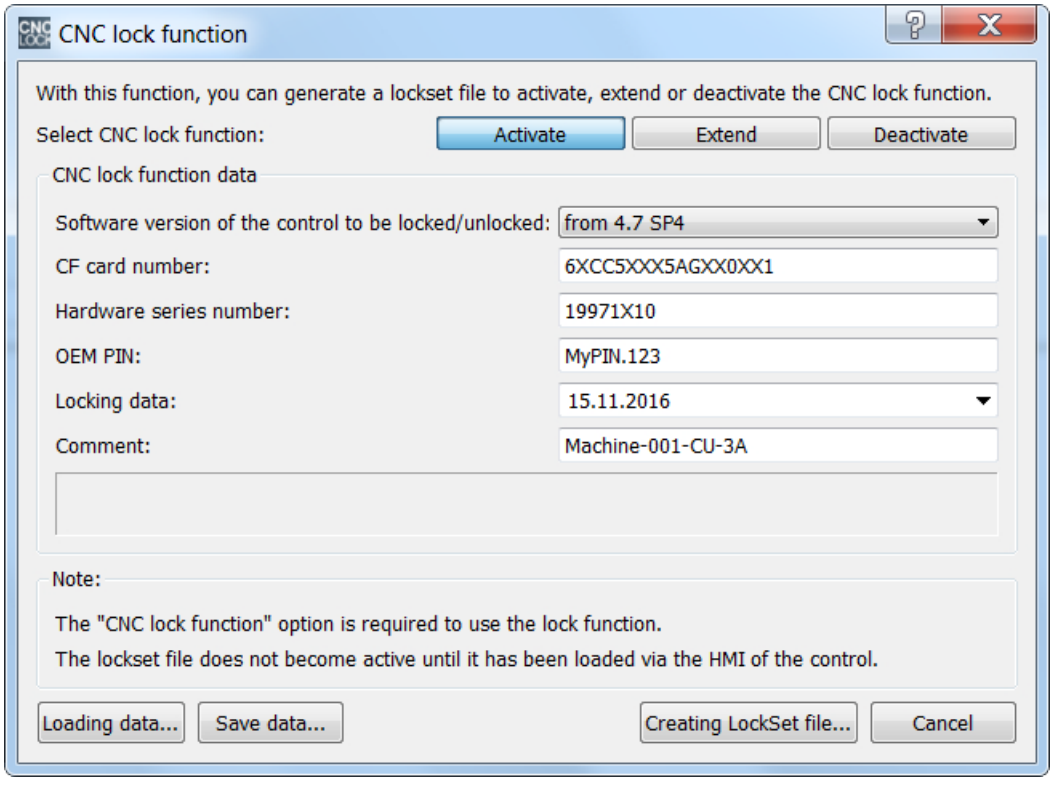
数控禁用功能的调试人员必须知道初始 **OEM-PIN**，以更换损坏的控制系统硬件 (PPU)。

说明

CF 卡和控制系统 (PPU) 的**序列号**位于 SINUMERIK Operate 操作界面上的“诊断” > “版本” > “硬件” > “NCU/PLC” > “ncu1” 操作区中：

- **CF 卡**：“CF 卡” > “序列号”区域
- **PPU**：“SINUMERIK 828D PPU” > “序列号”区域

通过 Access MyMachine 在对话框“数控禁用功能”中输入数据（通过主菜单：“其他”>“数控禁用功能”调用）。“**激活**”按钮必须已选择。



在输入所需所有数据后，通过“创建 Lockset 文件...”按钮创建用于解禁的 Lockset 文件。

传输 Lockset 文件

必须将新的 Lockset 文件传输至控制系统。此时，可由机床制造商通过以太网连接直接传输至控制系统。

文件位于：/System-CF-Card/User/sinumerik/data/license。

或者将新的 Lockset 文件发送给最终用户并由最终用户将其传输至控制系统。

读取 Lockset 文件

说明

读取 Lockset 文件**前**必须正确设置新控制系统硬件上的**时间**，因为在激活数控禁用功能时间点上的时间是保存为监控的起始值的。数控禁用功能的调试人员可通过更新和调整时间确保过程顺利。

通过操作界面启动对 Lockset 文件的读取：操作区“调试” > “许可证” > “读入许可证密钥”。

读取无需访问权限。

如果在读取 Lockset 文件时未出错，则数控禁用功能在新的控制系统中生效。

15.7.9 更换损坏的 CF 卡

更换损坏的系统 CF 卡时，必须将损坏的 CF 卡寄给西门子。相反，西门子会寄给您一个新 CF 卡。

在初次启动装配了新 CF 卡的控制系统后，必须读取通过之前的控制系统硬件创建的调试存档。下一次启动控制系统后，系统会显示报警 8062：“数控禁用功能：功能运行异常：原因 1”（更换了 CF 卡）并禁用 NC 启动。报警原因在于新 CF 卡的新序列号。

请求新 Lockset 文件

为重新解禁控制系统，最终用户需要向数控禁用功能的调试人员（机床制造商或代理商）请求与当前控制系统匹配的新 Lockset 文件（lockset.clc）。

创建 Lockset 文件

创建 Lockset 文件需要以下数据：

- **新** CF 卡的序列号
- 控制系统 (PPU) 的序列号

- 初始 OEM-PIN
- 依据数控禁用功能状态的禁用日期（最后一个禁用日期、新禁用日期、无禁用日期）

说明

数控禁用功能的调试人员必须知道初始 **OEM-PIN**，以更换损坏的 CF 卡。

说明

CF 卡和控制系统 (PPU) 的**序列号**位于 SINUMERIK Operate 操作界面上的“诊断” > “版本” > “硬件” > “NCU/PLC” > “ncu1” 操作区中：

- **CF 卡**：“CF 卡” > “序列号”区域
- **PPU**：“SINUMERIK 828D PPU” > “序列号”区域

通过 Access MyMachine 在对话框“数控禁用功能”中输入数据（通过主菜单：“其他”>“数控禁用功能”调用）。“**延长**”或“**取消**”按钮必须已选择：

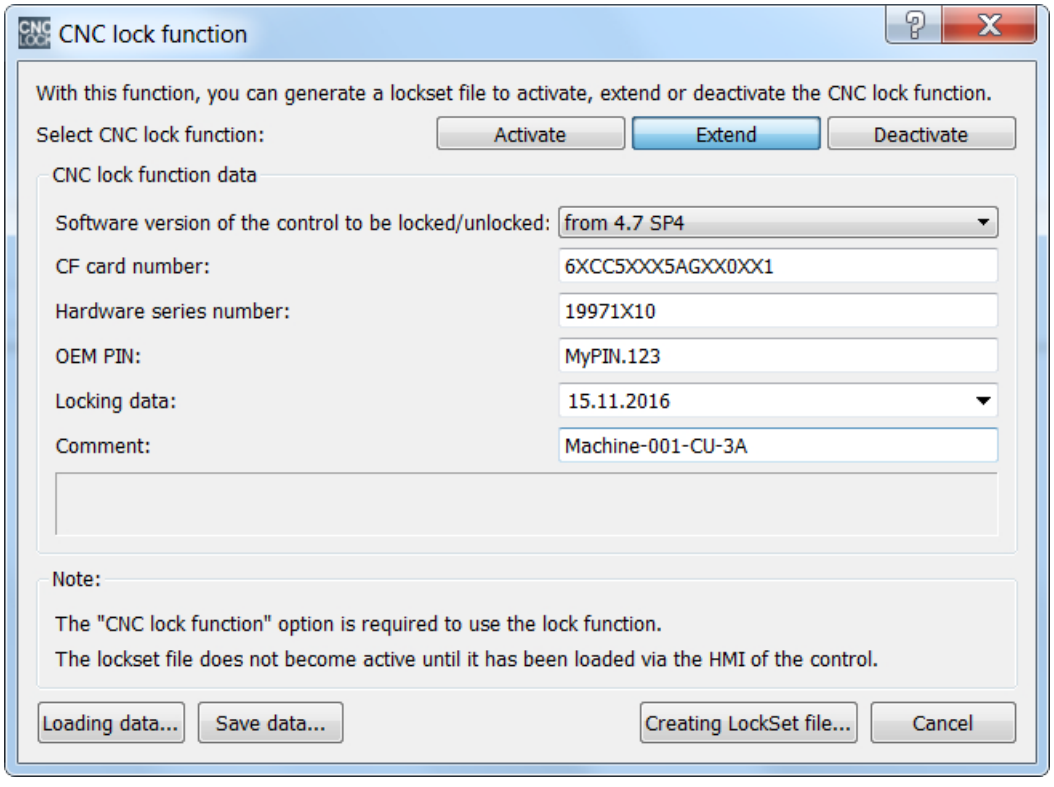


图 15-3　　更新数控禁用功能

在输入所需所有数据后，通过“创建 Lockset 文件...”按钮创建用于解禁的 Lockset 文件。

传输 Lockset 文件

必须将新的 Lockset 文件传输至控制系统。此时，可由机床制造商通过以太网连接直接传输至控制系统。

文件位于：/System-CF-Card/User/sinumerik/data/license。

或者将新的 Lockset 文件发送给最终用户并由最终用户将其传输至控制系统。

读取 Lockset 文件

说明

读取 Lockset 文件**前**必须正确设置新控制系统硬件上的**时间**，因为在激活数控禁用功能时间点上的时间是保存为监控的起始值的。数控禁用功能的调试人员可通过更新和调整时间确保过程顺利。

通过操作界面启动对 Lockset 文件的读取：操作区“调试” **>** “许可证” **>** “读入许可证密钥”。

读取无需访问权限。

如果在读取 Lockset 文件时未出错，则数控禁用功能在新的控制系统中生效。

15.7.10 忘记 OEM-PIN

数控禁用功能的调试人员（机床制造商或代理商）忘记了由其设定的 OEM-PIN，故无法再为相应的控制系统创建有效的 Lockset 文件。

激活设备

为确保数控禁用功能的调试人员能够再次进行操作，必须拨打西门子热线获取以下相关信息：

- “数控禁用功能”选件的许可证 (CoL)
- CF 卡的序列号
- 控制系统 (PPU) 的序列号
- 数控软件的软件版本

说明

CF 卡和控制系统 (PPU) 的**序列号**位于 SINUMERIK Operate 操作界面上的“诊断” **>** “版本” **>** “硬件” **>** “NCU/PLC” **>** “ncu1” 操作区中：

- CF-Card:Bereich "CF-Karte" > "SerialNo."
- PPU:Bereich "SINUMERIK 828D PPU" > "SerialNo."

数控软件的**软件版本**位于 SINUMERIK Operate 操作界面上的“诊断” **>** “版本” **>** “实际版本”操作区中：

必须将原版许可证（CoL）发送给西门子。由西门子提供免费激活服务。

数控禁用功能的调试人员通过拨打热线获取一个用于解禁设备的 Lockset 文件。但激活仅限于控制系统硬件。不包括 PLC 程序。因此，必须有与设备相匹配的原厂 PLC 程序。

而且必须由数控禁用功能的调试人员在最终用户（设备操作员）现场进行操作。

有关该流程的更多信息请拨打西门子热线。

15.7.11 其它信息

配置文件

说明

由机床制造商负责创建序列号与 OEM-PIN 的对应关系文档。

Access MyMachine 可用于创建包含以下数据的非加密配置文件（用于 “**U**ser-**C**NC-**L**ock-**S**et” 文件的 “ucls”）：

- CF 卡的序列号
- 控制系统 (PPU) 的序列号
- OEM-PIN
- 创建日期
- 禁用日期

在对话框“数控禁用功能”中通过按钮“保存数据...”调用该功能。

通过按钮“载入数据 ...”可重新读取保存在项目文件中的非加密数据。

日期或时间设置错误

若在数控禁用功能激活的情况下确定了错误的日期或时间设置，则会触发以下动作：

- Ausgabe von Alarm 8065:"CNC-Sperrfunktion:Bitte Datum/Uhrzeit korrekt einstellen!"
- NC 启动禁用。

为取消禁用，必须在关闭控制系统前将错误的设置更正。

注意
永久禁用 必须在关闭控制系统前将错误的日期或时间设置更正。Ansonsten droht die dauerhafte Sperre durch Alarm 8064:"CNC-Sperrfunktion:Das Sperrdatum ist erreicht, kein NC-Start möglich!" 解决方法：在关闭控制系统**前**正确地设置日期/时间。

Wird bei aktivierter CNC-Sperrfunktion das Datum versehentlich in die Zukunft gestellt, wird der folgende Alarm ausgegeben:

Alarm 8066 "CNC-Sperrfunktion:Durch Änderung des Datums wurde die verbleibende Laufzeit verringert!"

只要控制系统未关闭，就还能将日期更正。

注意
缩短的使用寿命 在关闭控制系统后，被设置成未来日期的日期会被视作真实日期，且无法再复位。这会使至禁用日期为止的使用寿命缩短。 解决方法：在关闭控制系统**前**正确地设置日期。

更多信息

参见：

- SINUMERIK Integrate Access MyMachine /P2P (PC) 的在线帮助
- PLC 编程工具的在线帮助

R1：回参考点 16

16.1 简要说明

功能

在机床轴回参考点时，机床轴坐标系与机床坐标系同步。为此会将机床轴运行至机床零点，之后将机床轴的实际位置设为零。

若受结构限制而无法直接驶向机床零点，则须使用一个在机床轴运行范围内定义的参考点来实现机床轴的同步。其相对于机床零点的位置必须已知。回参考点时机床轴的实际位置会设为该值。

测量系统和回参考点方式

回参考点时可采用以下测量系统和回参考点方式使机床轴同步：

- 测量系统
  - 至少有一个零脉冲的增量旋转测量系统
  - 增量线性测量系统（长度测量系统）
  - 使用距离编码参考脉冲的旋转测量系统（Heidenhain 公司）
  - 使用距离编码参考脉冲的线性测量系统（Heidenhain 公司）
  - 绝对式旋转测量系统
  - 绝对式线性测量系统（长度测量系统）
- 回参考点方式
  - 使用增量测量系统通过无接触式接近开关和单/双脉冲沿识别回参考点
  - 基于增量测量系统使用参考凸轮通过无接触式接近开关回参考点
  - 使用增量测量系统在主轴应用中通过无接触式接近开关以配置的逼近速度回参考点
  - 基于使用距离编码参考脉冲的测量系统通过越过 2 个或 4 个零脉冲回参考点
  - 通过测量系统补偿执行被动测量系统回参考点
  - 跟踪模式下回参考点
  - 通过驱动上的凸轮开关回参考点

启动

机床轴的回参考点运行可手动启动，或通过零件程序启动：

- 手动：JOG 和 MDI 运行方式，机床功能 REF
- 零件程序：指令 G74

16.2 轴专用回参考点

进行轴专用回参考点时，必须为每根需要回参考点的机床轴单独触发回参考点运行。

选择运行方式和机床功能

在进行机床轴回参考点前，首先切换至 JOG 或 MDI 运行方式：

DB11, ... DBX0.2（JOG 运行方式）

DB11, ... DBX0.1（MDI 运行方式）

接下来必须选择机床功能 REF（回参考点）：

DB11, ... DBX1.2（机床功能 REF）

启动回参考点

使用轴专用回参考点时，必须为每根轴分别启动回参考点。

回参考点运行通过轴专用运行键启动：

DB31, ... DBX4.6（运行键 -）

DB31, ... DBX4.7（运行键 +）

方向使能

为了避免误操作，必须对方向使能进行参数设置。通过方向使能确定通过哪个运行键启动回参考点：

MD34010 $MA_REFP_CAM_DIR_IS_MINUS = <值>

<值>	含义
0	沿正方向回参考点
1	沿负方向回参考点

长按模式

通过以下机床数据设定，是在按下一次方向键后完全执行回参考点，还是必须由机床操作人员长按方向键（长按模式）来执行（出于安全原因）：

MD11300 $MN_JOG_INC_MODE_LEVELTRIGGRD（INC/REF 长按模式）

机床操作人员在长按模式下松开方向键后，机床轴会制动至静止状态。回参考点运行不会终止。再次按下方向键便可继续执行回参考点。

回参考点状态

回参考点启动时，系统会反馈机床轴的回参考点状态：

DB31, ... DBX60.4（已回参考点/已同步 1）

DB31, ... DBX60.5（已回参考点/已同步 2）

DB21, ... DBX36.2（所有需要回参考点的轴均已回参考点）

距离编码测量系统

使用距离编码测量系统时，可通过任意运行键启动回参考点。

顺序：

进行机床轴回参考点时，机床操作人员或机床制造商（通过 PLC 用户程序）须确保遵循特定的顺序。

- 机床操作人员
  机床操作人员必须按照所要求的顺序启动机床轴。
- 机床制造商
  机床制造商的 PLC 用户程序更能够确保机床轴按照相应的顺序启动。

多根机床轴同时回参考点

根据控制系统，可同时为多根机床轴执行回参考点：

回参考点完成

若机床轴回参考点成功完成，系统会设置回参考点状态来进行反馈：

DB31, ... DBX60.4（已回参考点/已同步 1）

DB31, ... DBX60.5（已回参考点/已同步 2）

回参考点终止

使用轴专用回参考点时，机床轴在设定为机床轴主通道的通道中运行：

MD30550 $MA_ AXCONF_ASSIGN_MASTER_CHAN

因此，为了终止回参考点运行，必须为机床轴的主通道触发运行方式组复位或通道复位：

DB11, ... DBX0.7（BAG 复位）

DB21, ... DBX7.7（通道复位）

运行终止时尚未完成回参考点的机床轴都会保持“未回参考点”状态：

DB31, ... DBX60.4（已回参考点/已同步 1）

DB31, ... DBX60.5（已回参考点/已同步 2）

16.3 通道专用回参考点

进行通道专用回参考点时，触发回参考点运行后，系统将按照参数设置的顺序为通道的所有机床轴执行回参考点。

选择运行方式和机床功能

在进行机床轴回参考点之前，首先应将相应的 BAG 切换至 JOG 或 MDI 运行方式：

DB11, ... DBX0.2（JOG 运行方式）

DB11, ... DBX0.1（MDI 运行方式）

接下来必须选择机床功能 REF（回参考点）：

DB11, ... DBX1.2（机床功能 REF）

参数设置轴顺序

通道中的机床轴执行回参考点的顺序通过以下机床数据设置：

MD34110 $MA_REFP_CYCLE_NR = <编号>

<编号>	含义
-1	对于通道中的 NC 启动，机床轴不必为已回参考点状态。
0	机床轴不参加通道专用回参考点。
1 - 15	通道专用回参考点的顺序编号

机床轴会按照编号的升序执行回参考点。

相同编号的机床轴会同时执行回参考点

多根机床轴同时回参考点

根据控制系统，可同时为多根机床轴执行回参考点：

启动回参考点

通道专用回参考点通过以下方式启动：

DB21, ... DBX1.0（激活回参考点）

通道通过以下信号提示通道专用回参考点的状态：

DB21, ... DBX33.0（回参考点生效）

回参考点状态

回参考点启动时，系统会反馈机床轴的回参考点状态：

DB31, ... DBX60.4（已回参考点/已同步 1）

DB31, ... DBX60.5（已回参考点/已同步 2）

回参考点完成

参与通道专用回参考点的所有机床轴均成功回参考点后，系统将进行相应反馈：

DB21, ... DBX36.2（所有需要回参考点的轴均已回参考点）

回参考点终止

使用通道专用回参考点时，机床轴在其作为通道轴实际指定到的通道中运行。

因此，为了终止回参考点运行，必须为对应通道触发运行方式组复位或通道复位：

DB11, ... DBX0.7（BAG 复位）

DB21, ... DBX7.7（通道复位）

运行终止时尚未完成回参考点的机床轴都会保持“未回参考点”状态：

DB31, ... DBX60.4（已回参考点/已同步 1）

DB31, ... DBX60.5（已回参考点/已同步 2）

16.4 在零件程序中写入回参考点（G74）

功能

借助指令 G74 可通过零件程序或同步动作为机床轴执行首次回参考点，或重新回参考点。

例如在以下情形下便需要重新回参考点：

- 实际值偏移 PRESETON
  文档：编程手册之工作准备分册，章节“坐标转换（框架）”>“通过 PRESETON 启用预设定偏移”
- “驻留”机床轴 (页 131)
- 超出位置测量系统的编码器极限频率

编程

句法

G74 <机床轴> {<机床轴>}

含义

G74：	用于机床轴回参考点的指令	
	独占一个程序段：	是
	有效性：	逐段生效
<机床轴>：	必须指定机床轴的名称（MD10000 $MN_AXCONF_MACHAX_NAME_TAB）。 该机床轴必须为执行相应零件程序或同步动作的通道的通道轴。	

复位特性

运行方式组复位或通道复位会使编写的所有机床轴回参考点运行终止：

- DB11, ... DBX0.7（BAG 复位）
- DB21, ... DBX7.7（通道复位）

复位时尚未完成回参考点的机床轴都会保持“未回参考点”状态：

- DB31, ... DBX60.4 (已回参考点/已同步 1) == 0
- DB31, ... DBX60.5（已回参考点/已同步 2) == 0

16.5 使用增量测量系统回参考点

16.5.1 硬件信号

根据机床结构和所使用的增量编码器的属性，必须连接不同的硬件信号。

参考凸轮

- 连接
  参考凸轮信号可连接至外部 PLC I/O 模块的数字量输入，或 NCU 接口 X142 的快速输入。
- NC/PLC 接口信号
  参考凸轮信号必须由 PLC 用户程序传输至轴专用 NC/PLC 接口。
  DB31, ... DBX12.7（回参考点减速）

零脉冲选择

若在轴或主轴回参考点期间识别出多个测量系统零脉冲（例如电机和编码器间的测量齿轮箱），必须通过一个附加的无接触式接近开关信号选择指定的零脉冲。

- 连接
  无接触式接近开关必须连接至 NCU 接口 X122 或 X132 的一个快速数字量输入。
- 激活
  为了分析无接触式接近开关信号，必须在驱动参数 p0493 中为轴/主轴选择连接无接触式接近开关的数字量输入。

替代零脉冲

若使用的测量系统不能提供零脉冲信号，可通过无接触式接近开关信号生成一个替代零脉冲。

- 连接
  无接触式接近开关必须连接至 NCU 接口 X122 或 X132 的一个快速数字量输入。
- 激活
  为了分析无接触式接近开关信号，必须在驱动参数 p0494 或 p0495 中为轴/主轴选择连接无接触式接近开关的数字量输入。

概述

信号	接口：数字量输入通过	设置
参考凸轮	外部 PLC I/O 模块或 NCU：X142	PLC 用户程序：DB31, ... DBX12.7
零脉冲选择	NCU:X122 或 X132	驱动参数：p0493
外部零脉冲或替代零脉冲	NCU:X122 或 X132	驱动参数：p0494 或 p0495

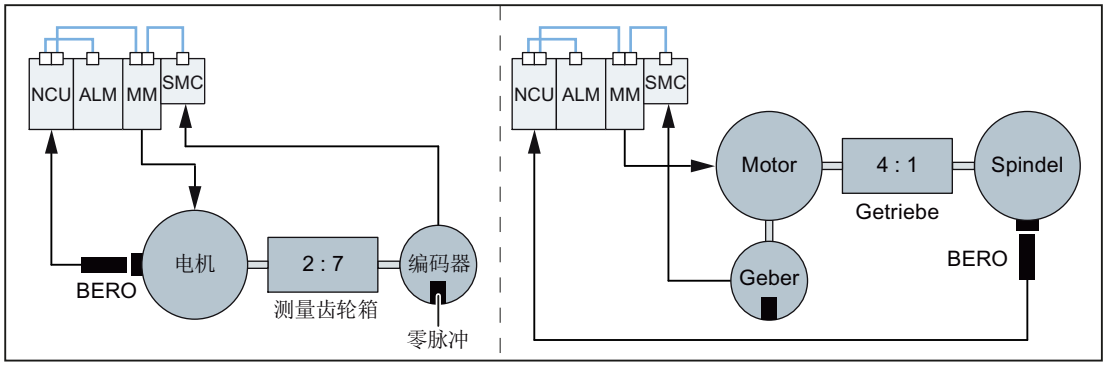
文档

- NCU 接口：SINUMERIK 840D sl 设备手册 NCU7x0.3 PN，
  章节：“连接”>“数字量输入/输出”
- 驱动参数：SINAMICS S120/S150 参数手册

16.5.2 零脉冲选择

功能

增量测量系统的回参考点基于编码器零脉冲相对机床轴整个运行范围的唯一位置。若在特定的机床条件下能够在机床轴的运行范围内识别出多个编码器零脉冲（示例见下图），则须在机床上启用无接触式接近开关来确定唯一的零脉冲。此时参考点的位置由无接触式接近开关信号和编码器零脉冲的信号组合得出。



BER 无接触式接近开关
O

图 16-1 电机和编码器间的测量齿轮箱，或电机和主轴间的降档齿轮箱

参数设置

NC：回参考点模式

回参考点模式必须设置为“增量、旋转或线性测量系统的回参考点：编码器信号上的零脉冲”：

MD34200 $MA_ENC_REFP_MODE[<轴>] = 1

驱动：零脉冲选择

在参数 p0493 中必须设置连接无接触式接近开关的 NCU 接口数字量输入。

说明

零脉冲选择

无接触式接近开关信号只在驱动中处理。连接和参数设置参见“硬件信号 (页 1355)”章节。

16.5.3 时序

使用增量测量系统回参考点的时序分为三个阶段：

- 阶段 1：“阶段 1：运行到参考凸轮 (页 1359)”
- 阶段 2：“阶段 2：与零脉冲同步 (页 1362)”
- 阶段 3：“阶段 3：运行至参考点 (页 1367)”

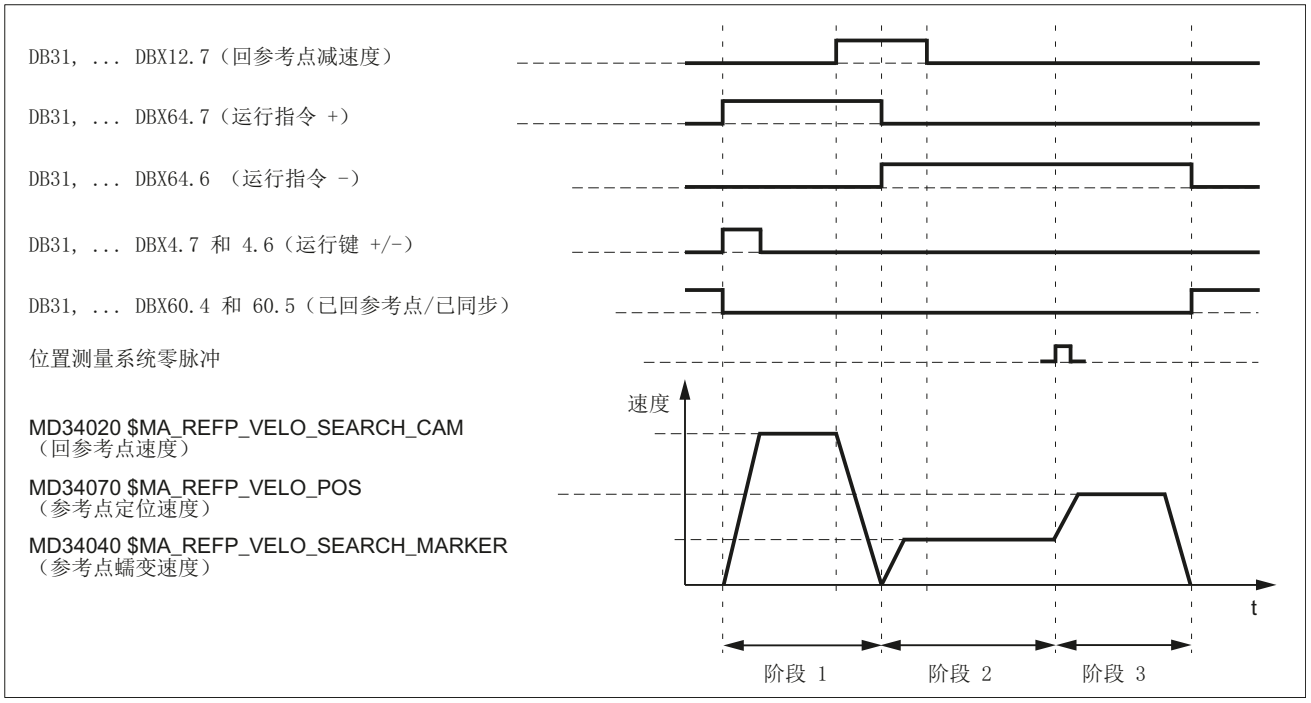


图 16-2　使用增量测量系统回参考点时的时序（示例）

16.5.4 阶段 1：运行到参考凸轮

阶段 1：图形显示

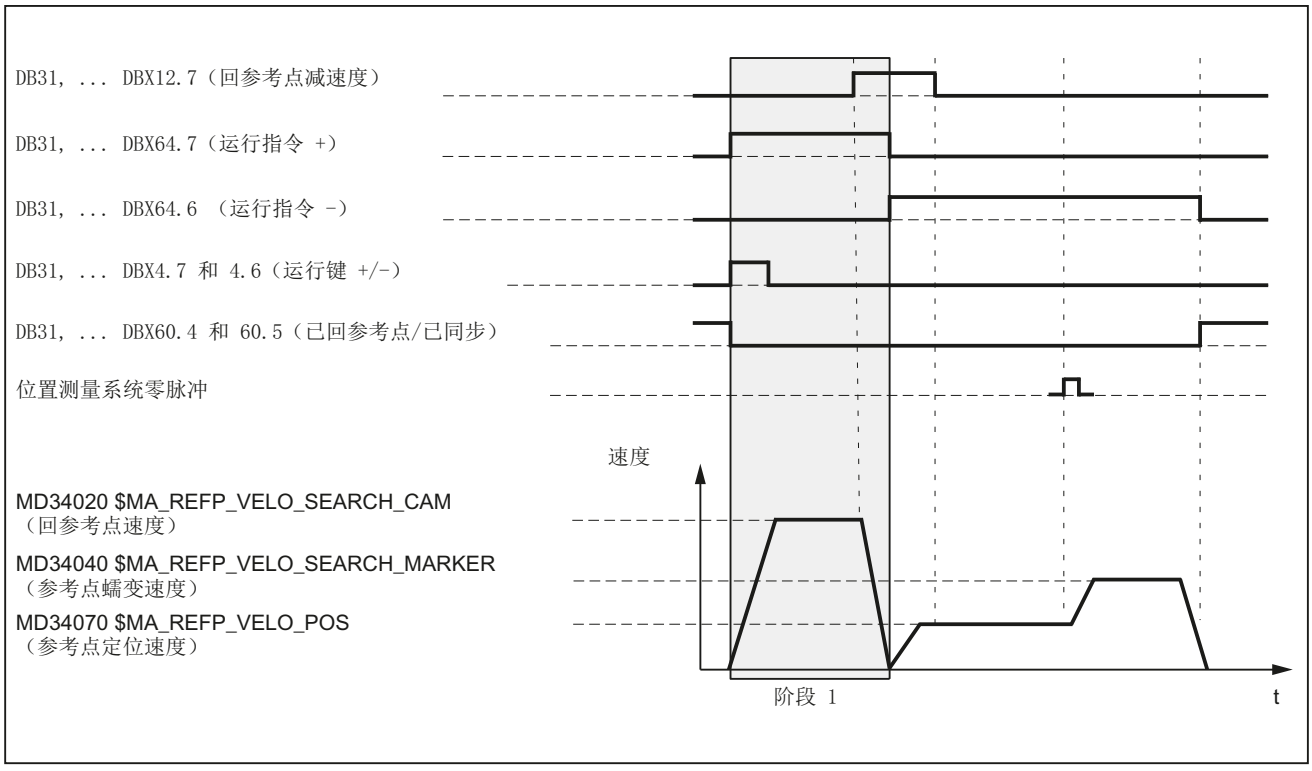


图 16-3 阶段 1：运行到参考凸轮

阶段 1：启动

回参考点运行的启动参见“轴专用回参考点 (页 1350)”和“轴专用回参考点 (页 1350)”章节。

阶段 1：过程

在阶段 1 中，根据机床轴相对参考凸轮的位置分为三种情形：

1. 机床轴位于参考凸轮前
2. 机床轴位于参考凸轮上
3. 机床轴无参考凸轮

情况 1：机床轴位于参考凸轮前

启动回参考点运行后，机床轴沿设定的方向加速至参数设置的回参考点速度：

- MD34010 $MA_REFP_CAM_DIR_IS_MINUS（沿负方向逼近参考点）
- MD34020 $MA_REFP_VELO_SEARCH_CAM（回参考点速度）

必须在 PLC 用户程序中查询数字量输入来识别“到达参考凸轮”，并通过以下接口信号将此信息提供给 NC：

DB31, ... DBX12.7 = 1（回参考点减速）

识别出 NC/PLC 接口信号时，机床轴将制动直至到达静止状态。此时至少还会运行 s_{min} 的行程。这样可确保在阶段 2 中机床轴能够以参数设置的参考点停止速度离开参考凸轮。

$$s_{min} = \frac{(\text{MD34040 \$MA\_REFP\_VELO\_SEARCH\_MARKER})^2}{2 * \text{MD32300 \$MA\_MAX\_AX\_ACCEL}}$$

这样一来阶段 1 结束。之后将继续执行回参考点阶段 2。

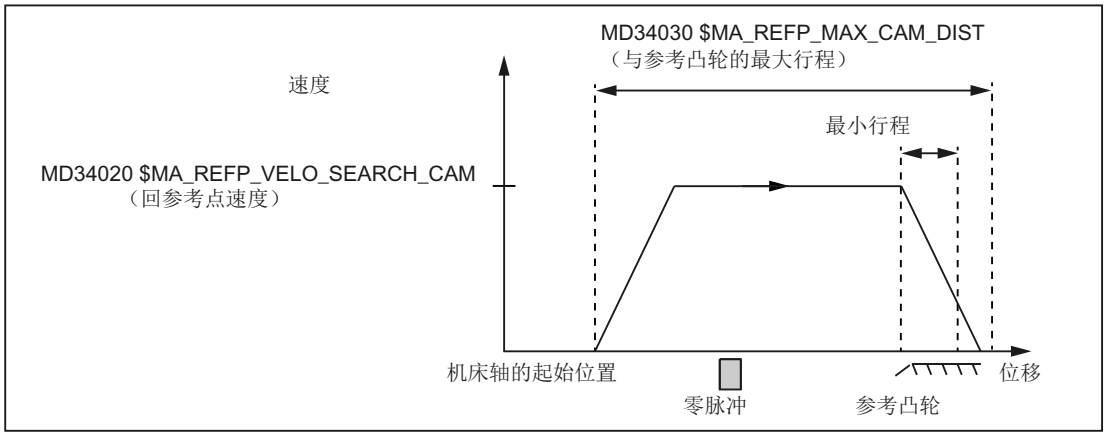


图 16-4　　减速的最小行程

情况 2：机床轴位于参考凸轮上

机床轴停留在初始位置。
这样一来阶段 1 结束。之后将继续执行回参考点阶段 2。

情况 3：机床轴无参考凸轮

无参考凸轮的机床轴将停留在初始位置。

其中例如包括：

- 整个运行范围内只有一个零脉冲的机床轴
- 每转只有一个零脉冲的回转轴

对于无参考凸轮的机床轴，以下机床数据必须设置为零：

MD34000 $MA_REFP_CAM_IS_ACTIVE = 0（带参考凸轮的轴）

这样一来阶段 1 结束。之后将继续执行回参考点阶段 2。

阶段 1：属性

- 进给倍率生效。
- 进给停止 (通道专用和轴专用) 生效。
- NC 停止和 NC 启动均生效。
- 在参数设置的最大行程内无法到达参考凸轮时，机床轴将停止：
  MD34030 $MA_REFP_MAX_CAM_DIST（至参考凸轮的最大行程距离）

参见

通道专用回参考点 (页 1352)

16.5.5 阶段 2：与零脉冲同步

阶段 2：图形显示

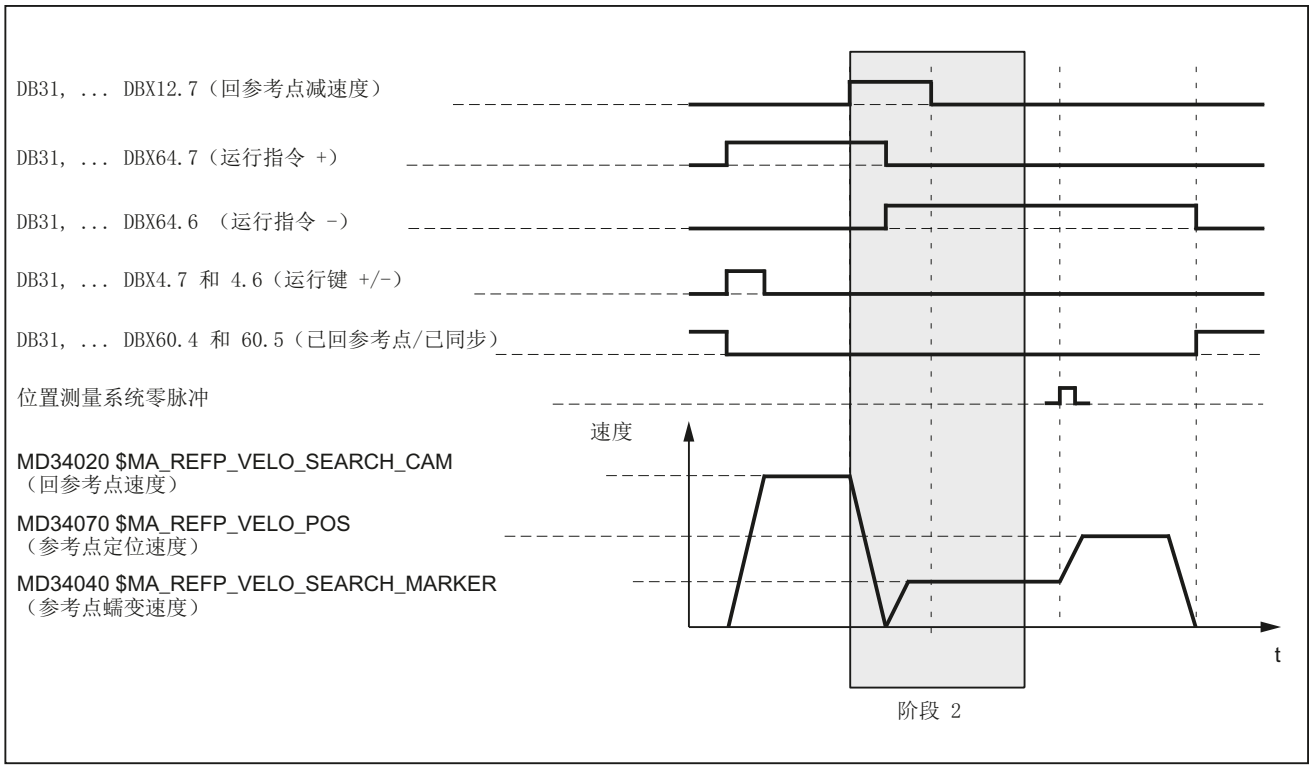


图 16-5 阶段 2：与零脉冲同步

阶段 2：启动

若第 1 阶段未发生报警且成功完成，则自动启动第 2 阶段。

初始情况：

机床轴位于参考凸轮上。

零脉冲搜索方向：

零脉冲搜索方向取决于以下机床数据中的设置：

- MD34010 $MA_REFP_CAM_DIR_IS_MINUS（沿负方向逼近参考点）
- MD34050 $MA_REFP_SEARCH_MARKER_REVERSE（参考凸轮上换向）

阶段 2：过程

阶段 2 中的同步可通过参考凸轮的下降沿或上升沿实现。这通过以下机床数据设置：

MD34050 $MA_REFP_SEARCH_MARKER_REVERSE[<轴>] = <值>

值	含义
0	通过参考凸轮下降沿同步
1	通过参考凸轮上升沿同步

说明

若机床轴逼近参考凸轮时的实际速度尚未达到阶段 2 设定速度（参数设置的公差限值内），则会重新启动阶段 1。例如，启动回参考点时机床轴位于参考凸轮上时便会出现此情形。

MD35150 $MA_SPIND_DES_VELO_TOL（主轴转速公差）

情形 1：通过参考凸轮下降沿同步

通过参考凸轮下降沿同步时，机床轴会逆向于设置的回参考点方向（阶段 1 的运行方向）加速至参数设置的参考点停止速度。

离开参考凸轮后会等待下一个编码器零脉冲：
DB31, ... DBX12.7 == 0

识别出编码器零脉冲时，阶段 2 结束。机床轴以恒定速度运行，并继续执行回参考点阶段 3。

- MD34040 $MA_REFP_VELO_SEARCH_MARKER（参考点停止速度）

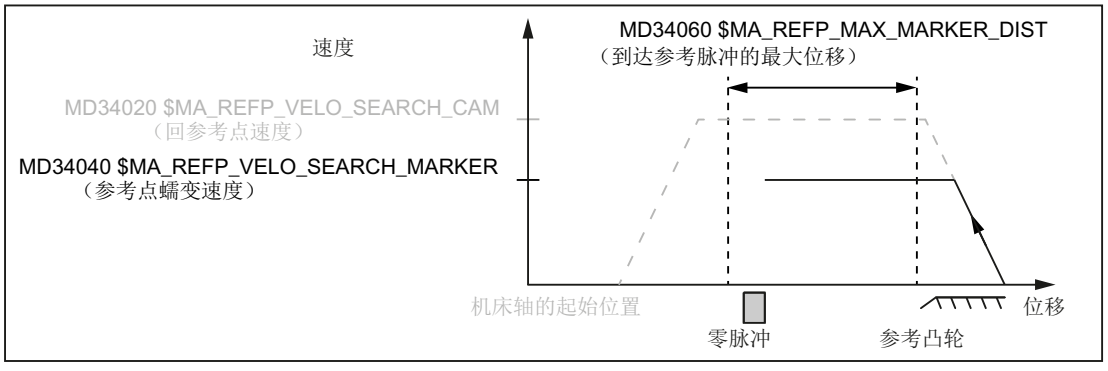


图 16-6　　通过参考凸轮下降沿同步

情形 2：通过参考凸轮上升沿同步

通过参考凸轮上升沿同步时，机床轴会逆向于设置的回参考点方向（阶段 1 的运行方向）加速至参数设置的回参考点速度。

- MD34020 $MA_REFP_VELO_SEARCH_CAM（回参考点速度）
- MD34010 $MA_REFP_CAM_DIR_IS_MINUS（沿负方向逼近参考点）

离开参考凸轮后机床轴会向静止状态制动：
DB31, ... DBX12.7 == 0

之后机床轴通过参数设置的参考点停止速度返回参考凸轮：

MD34040 $MA_REFP_VELO_SEARCH_MARKER（参考点停止速度）

到达参考凸轮（DB31, ... DBX12.7 = 1）后，等待下一个编码器零脉冲。

识别出编码器零脉冲时，阶段 2 结束。机床轴以恒定速度运行，并继续执行回参考点阶段 3。

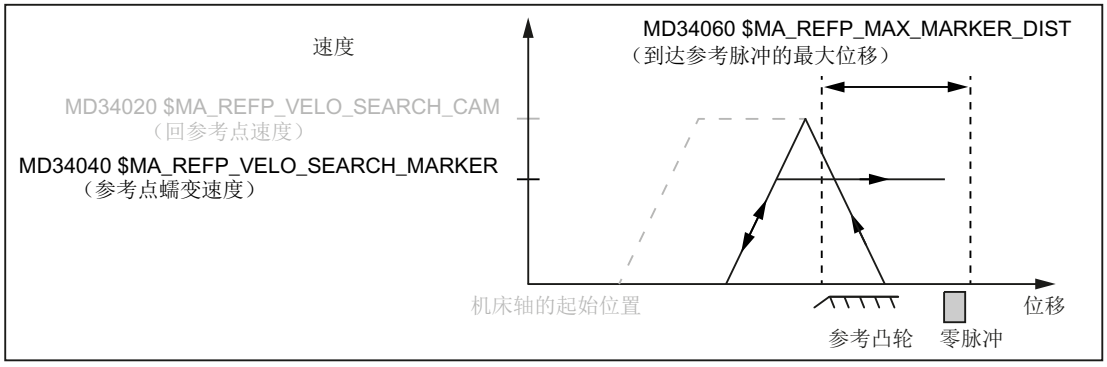


图 16-7　　通过参考凸轮上升沿同步

电子参考凸轮偏移

电子参考凸轮偏移用于补偿温度造成的参考凸轮长度延展，从而确保始终与相同的编码器零脉冲同步。

MD34092 $MA_ REFP_CAM_SHIFT（提供等距零脉冲的增量测量系统的电子参考凸轮偏移）

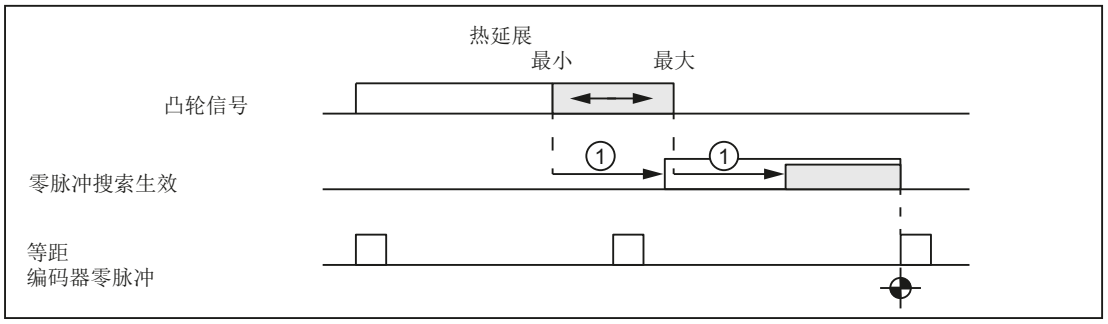
启用电子参考凸轮偏移时，识别出参考凸轮脉冲沿后不会立即与下一个编码器零脉冲同步，而是在经过设置的偏移后再进行同步。

生效的偏移 s_{shift} 基于自识别出参考凸轮脉冲沿起插补周期中经过的位移得出：

s_{shift_min} = MD34092 $MA_ REFP_CAM_SHIFT

s_{shift_max} = MD34092 $MA_ REFP_CAM_SHIFT +
MD34040 $MA_REFP_VELO_SEARCH_MARKER * 插补周期

电子参考凸轮偏移始终在零脉冲搜索方向上生效。



① 参考凸轮偏移

图 16-8 电子参考凸轮偏移

前提条件

电子参考凸轮偏移仅适用于带参考凸轮的机床轴：

MD34000 $MA_REFP_CAM_IS_ACTIVE == 1

参考凸轮校准

提供等距零脉冲的编码器

对于等距提供零脉冲的编码器，必须精确校准参考凸轮，确保回参考点运行中始终能识别正确的零脉冲。

时间特性

从到达机床上的参考凸轮，直到识别出从 PLC 用户程序传输至 NC 的参考凸轮信号，这期间的时间特性受以下因素影响：

- 参考凸轮开关的精度
- 参考凸轮开关的延时（常闭触点）
- PLC 输入上的延时
- PLC 循环时间
- NC/PLC 接口更新所需的循环时间

- 插补周期
- 位置控制周期

设置说明

- 参考凸轮
  实际应用中，在两个零脉冲中央校准参考凸轮的脉冲沿是可靠方案。
- 电子参考凸轮偏移

> **⚠ 警告**
>
> **碰撞危险**
>
> 参考凸轮的校准错误或不精确时，可能会分析错误的零脉冲。此时控制系统会因此确定错误的机床零点。这样一来机床轴会运行至错误的位置。软件限位开关、保护区和工作区域限制也将对错误的位置生效，这样便无法保护机床。位移差相当于机床轴两个零脉冲间经过的行程的 +/- 值。

以下只读机床数据用作设置电子参考凸轮偏移时的参考信息：
MD34093 $MA_REFP_CAM_MARKER_DIST（参考凸轮/参考标记距离）
该机床数据显示的值对应从离开参考凸轮到出现参考标记时之间的距离。若该值过小，则存在温度影响或凸轮信号运行时间浮动造成参考点不确定的危险。

阶段 2：属性

- 进给倍率不生效。
  内部按进给倍率 = 100% 运行。
  若进给倍率设为 0%，则终止运行。
- 进给停止 (通道专用和轴专用) 生效。
- NC 停止和 NC 启动均不生效。
- 若阶段 2 中机床轴未在参数设置的行程内到达参考脉冲（编码器零脉冲），那么机床轴将停止：
  MD34060 $MA_REFP_MAX_ MARKER_DIST（至参考脉冲的最大行程）

16.5.6 阶段 3：运行至参考点

阶段 3：图形显示

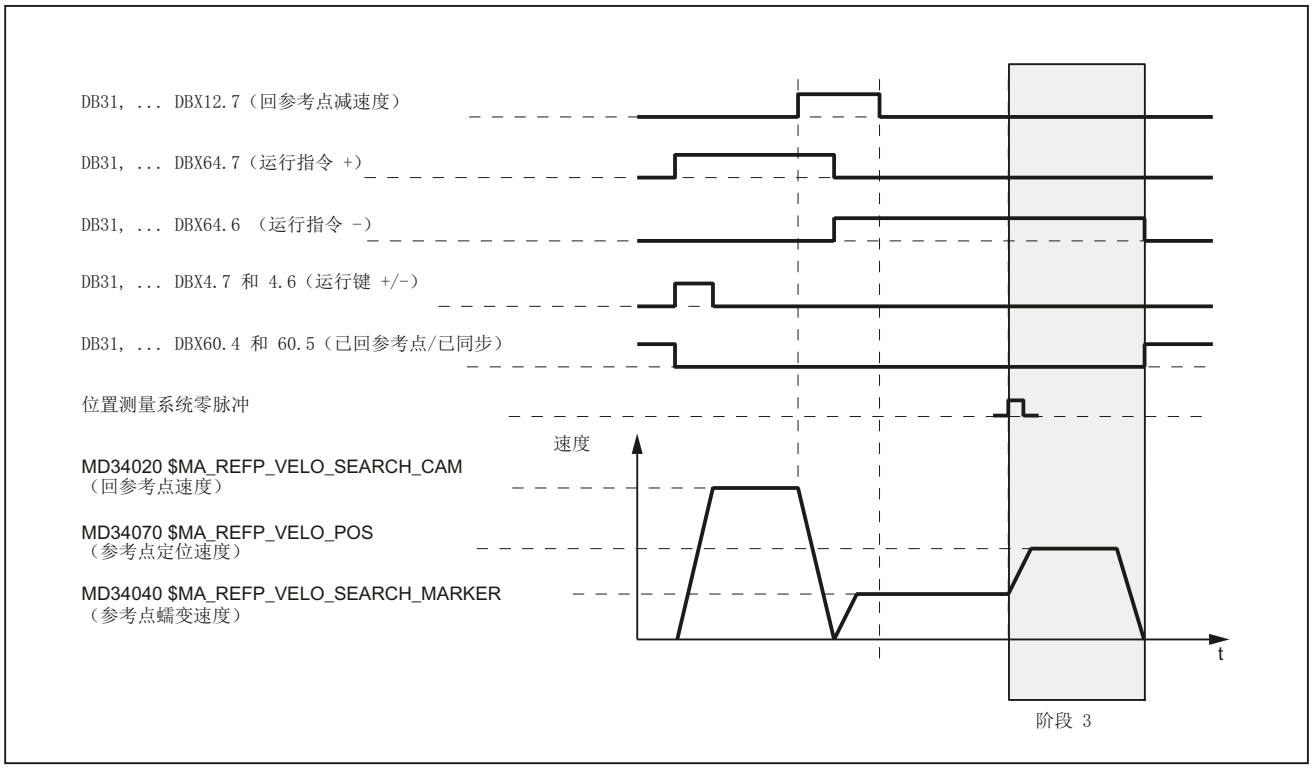


图 16-9 阶段 3：运行至参考点

阶段 3：启动

在阶段 2 的末尾，机床轴以参考点停止速度运行。因此，一旦阶段 2 在未触发报警的情形下顺利结束，便会立即切换至阶段 3。

初始状态

已识别编码器零脉冲。

阶段 3：过程

机床轴通过参数设置的参考点定位速度：
MD34070 $MA_REFP_VELO_POS（参考点定位速度）
从阶段 2 中识别的编码器零脉冲运行至参考点。

此时待运行的行程 s_{ref} 为参考点距离和参考点偏移之和：

MD34080 $MA_REFP_MOVE_DIST（参考点距离）

MD34090 $MA_REFP_MOVE_DIST_CORR（参考点偏移）

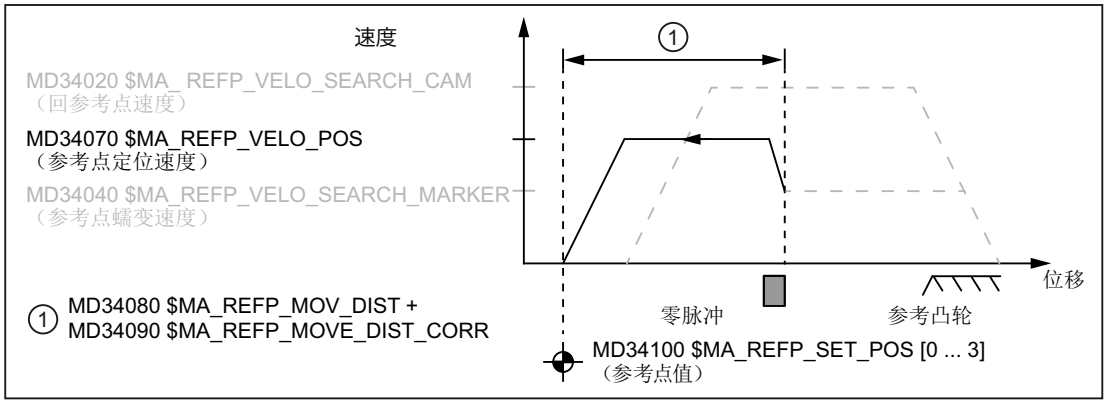


图 16-10　参考点位置

到达参考点后，机床轴将停止，实际值系统会与通过 NC/PLC 接口设定的参考点值 n 进行同步：

MD34100 $MA_ REFP_SET_POS[<n>]（参考点值）

参考点通过以下 NC/PLC 接口选择：

DB31, ... DBX2.4 ... 7（参考点值 1 ... 4）

实际值系统会与阶段 1 中到达参考凸轮（DB31, ... DBX12.7 == 1）时选择的参考点值进行同步。

就此机床轴已完成回参考点。根据生效的测量系统，系统会设置以下接口信号作为对 PLC 用户程序的反馈信息：

DB31, ... DBX60.4 / .5（已回参考点/已同步 1 / 2）= 1

阶段 3 的属性

- 进给倍率生效。
- 进给停止 (通道专用和轴专用) 生效。
- NC 停止和 NC 启动均生效。

阶段 3 的特殊性

在下列情形下，机床轴在识别出零脉冲后先会停止，并于随后返回参考点：

- 参考点距离和参考点偏移之和小于所需的制动行程（基于参考点定位速度）：
  MD34080 + MD34090 <“所需制动行程（基于 MD34070）”
- 在与当前运行方向相反的方向上，参考点位于参考凸轮“后”。

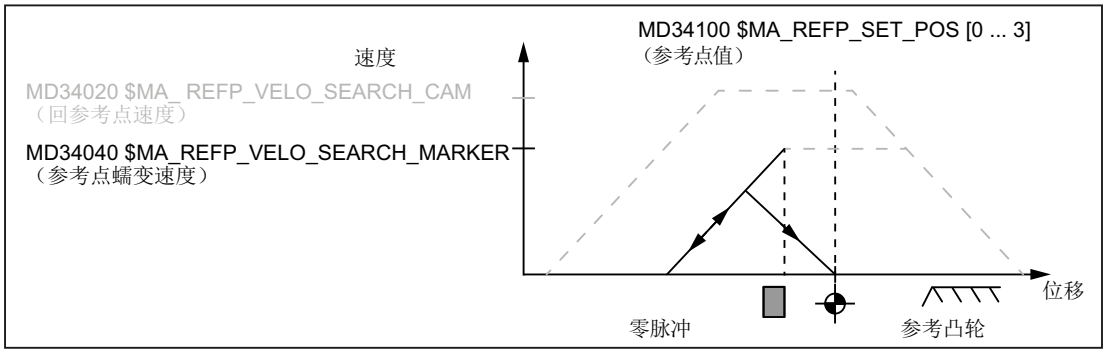


图 16-11　参考点距离和参考点偏移之和小于制动行程

16.6 通过距离编码的参考脉冲回参考点

16.6.1 概述

距离编码的参考脉冲

采用距离编码参考脉冲的测量系统由两个并行的标度信号构成：

- 增量光栅
- 参考脉冲信号

每两个相邻参考脉冲之间的距离已定义。当运行超过两个相邻参考脉冲后，可由此确定机床轴的绝对位置。 例如：假设参考脉冲间距大约为 10 mm，则机床轴回参考点时仅需大约 20 mm 的运行路径即可。

可从任意的轴位置出发沿正方向或负方向回参考点（特例：运行区域末端）。

16.6.2 基础参数设置

线性测量系统

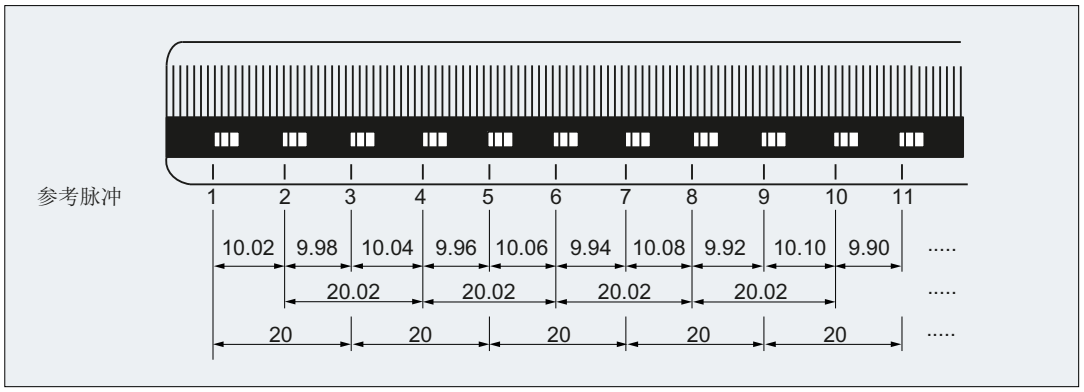


图 16-12 采用距离编码参考脉冲的玻璃量具，分度区段：20 mm

通过以下数据参数设置线性测量系统：

- 机床零点和长度测量系统第一个参考脉冲位置之间的绝对偏移量：
  MD34090 $MA_REFP_MOVE_DIST_CORR（参考点偏移/绝对偏移）
  参见：确定绝对偏移
- 长度测量系统相对于机床坐标系的方向设置（相同或相反）：
  MD34320 $MA_ENC_INVERS（长度测量系统与机床坐标系方向相反）

旋转测量系统

旋转测量系统的规则与线性测量系统相同（参见上文）。

确定绝对偏移

确定机床零点和机床轴第一个参考脉冲位置之间的绝对偏移量时，建议如下操作：

1. 将绝对偏移量数值设置为零：
   MD34090 $MA_REFP_MOVE_DIST_CORR = 0
2. 执行回参考点操作。
   提示：回参考点的执行位置应根据以下标准选择：例如能使用激光干涉仪轻松测得机床轴相对于机床零点的准确位置。
3. 通过操作界面（例如 SINUMERIK Integrate）读取显示在机床坐标系 MCS 中的机床轴实际位置。

4. 以机床零点为基准，测量机床轴的当前位置。
5. 计算出绝对偏移量并输入 MD34090 中。
   根据测量系统相对机床坐标系的定向，绝对偏移量如下计算：

测量系统的方向	绝对偏移量
相同	（所测得的位置）+（显示的实际位置）
相反	（所测得的位置）-（显示的实际位置）

⚠ 警告

参考点偏差

得到绝对偏移量并填入 MD34090 后，必须重新启动机床轴的回参考点运行。

回参考点模式

通过距离编码的参考脉冲回参考点时，可采用以下两种模式：

- 分析计算相邻**两个**参考脉冲：
  MD34200 $MA_ENC_REFP_MODE = 3
  优点：
  – 运行路径较短
- 分析计算相邻**四个**参考脉冲：
  MD34200 $MA_ENC_REFP_MODE = 8
  优点：
  – 可通过 NC 进行合理性检查
  – 提高回参考点结果的安全性

16.6.3 时序

时序

通过距离编码的参考脉冲回参考点时，过程分为 2 个阶段：

- 阶段 1：同步越过参考脉冲
- 阶段 2：运行至固定目标点

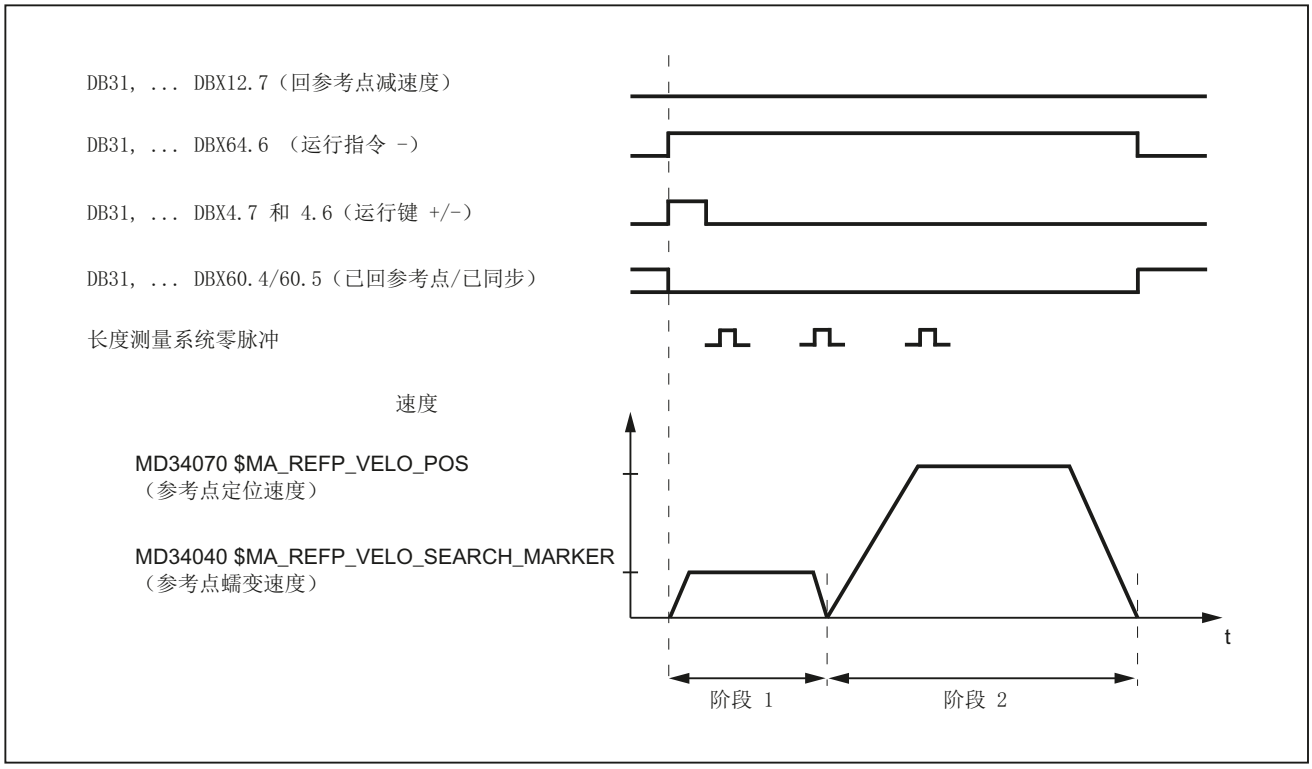


图 16-13 距离编码的参考脉冲

16.6.4 阶段 1：同步越过参考脉冲

阶段 1：启动

回参考点运行的启动参见“轴专用回参考点 (页 1350)”和“通道专用回参考点 (页 1352)”章节。

参考凸轮

对于使用距离编码的参考脉冲的测量系统，回参考点本来无需任何参考凸轮。出于功能性考虑，在通道专用回参考点运行和使用零件程序回参考点运行中（G74）须在机床轴的运行范围末端安装参考凸轮。

阶段 1：过程

未触碰到参考凸轮的过程

启动回参考点运行后，机床轴加速至设置的参考点停止速度：

MD34040 $MA_REFP_VELO_SEARCH_MARKER（参考点停止速度）

越过设置数量的参考脉冲后，机床轴再次停止，其实际值系统同步至 NC 获得的绝对位置。

在参考凸轮上启动的过程

若启动回参考点运行时机床轴位于参考凸轮上，则其将加速至设置的参考点停止速度，运行方向与设置的回参考点运行方向相反：

MD34040 $MA_REFP_VELO_SEARCH_MARKER（参考点停止速度）

MD34010 $MA_CAM_DIR_IS_MINUS（沿负方向回参考点）

如此可保证机床轴在未越过设定数量的参考脉冲前不会撞上运行范围限制点。

越过设置数量的参考脉冲后，机床轴再次停止，其实际值系统同步至 NC 获得的绝对位置。

回参考点途中触碰到参考凸轮的过程

启动回参考点运行后，机床轴加速至设置的参考点停止速度：

MD34040 $MA_REFP_VELO_SEARCH_MARKER（参考点停止速度）

机床轴在越过设定数量的参考脉冲前即触碰到了参考凸轮。此种情况下，机床轴会反向运行并沿相反的方向重新启动参考脉冲搜索。

越过设置数量的参考脉冲后，机床轴再次停止，其实际值系统同步至 NC 获得的绝对位置。

参考脉冲间距合理性检查

若 NC 在回参考点运行过程中探测到两个相邻参考脉冲的间距大于编写的参考脉冲间距的两倍，则表示存在错误：

MD34300 $MA_ENC_REFP_MARKER_DIST（参考脉冲间距）

此种情况下，机床轴会以设置的参考点停止速度（MD34040）的一半沿反方向运行并重新启动参考脉冲搜索。

若又再次发现参考脉冲间距有误，则机床轴会停止并终止回参考点运行（报警 20003“测量系统故障”）。

终止标准

若在编写的行程中未识别出设定数量的参考脉冲，机床轴会停止并终止回参考点运行：

MD34060 $MA_REFP_MAX_ MARKER_DIST（至参考脉冲的最大行程）

阶段 1 的属性

阶段 1 成功结束后，机床轴实际值系统即完成同步。

16.6.5 阶段 2：运行到目标点

阶段 2：启动

若第 1 阶段未发生报警且成功完成，则自动启动第 2 阶段。

初始情况：

- 机床轴直接停在设定数量的参考脉冲中的最后一个上。
- 机床轴的实际值系统已同步。

阶段 2：过程

为完成回参考点运行，机床轴将在阶段 2 中运行至定义的目标位置（参考点）。若需要缩短参考点的运行路径，也可抑制此过程进行：

MD34330 $MA_STOP_AT_ABS_MARKER = <值>

值	含义
0	运行至目标位置
1	不运行至目标位置

运行至目标位置（通常情况）

机床轴加速至设置的参考点定位速度并运行至编写的目标点（参考点）：

MD34070 $MA_REFP_VELO_POS（参考点定位速度）

MD34100 $MA_REFP_SET_POS（参考点值）

就此机床轴已完成回参考点。根据生效的测量系统，NC 会设置对应的接口信号作为提示：

DB31, ... DBX60.4 / 60.5（已回参考点/已同步 1 / 2）= 1

不运行至目标位置

就此机床轴已完成回参考点。根据生效的测量系统，NC 会设置对应的接口信号作为提示：

DB31, ... DBX60.4 / 60.5（已回参考点/已同步 1 / 2）= 1

阶段 2 的属性

根据是否为机床轴编写了参考凸轮，阶段 2 将具有不同特性。

无参考凸轮的机床轴

MD34000 $MA_REFP_CAM_IS_ACTIVE（带参考凸轮的轴）= 0

属性：

- 进给倍率生效。
- 进给停止 (通道专用和轴专用) 生效。
- NC 停止和 NC 启动均生效。

带参考凸轮的机床轴

MD34000 $MA_REFP_CAM_IS_ACTIVE（带参考凸轮的轴）= 1

属性：

- 进给倍率不生效。
  内部按进给倍率 = 100% 运行。
  若进给倍率设为 0%，则终止运行。
- 进给停止 (通道专用和轴专用) 生效。
- NC 停止和 NC 启动均不生效。
- 离开参考凸轮后，若未在编写的行程中识别出设定数量的参考脉冲，那么机床轴将停止：
  MD34060 $MA_REFP_MAX_ MARKER_DIST（至参考脉冲的最大行程）

旋转测量系统的特点

在使用距离编码的旋转测量系统中，只能在旋转范围内确定唯一的绝对位置。由于编码器机械结构的差别，硬件方面的绝对位置溢出并不总是与回转轴的运行范围吻合。

模数回转轴上的特殊性

对于模数回转轴，参考点位置映射至编写的模数区域：

MD30330 $MA_MODULO_RANGE（模数区域的大小）

MD30340 $MA_MODULO_RANGE_START（模数区域起点）

说明

也可通过轴功能“在配置的模数区域内确定使用距离编码的旋转编码器的参考点位置”将参考点位置映射至编写的（虚拟）模数区域：

MD30455 $MA_MISC_FUNCTION_MASK（轴功能），位 1 = 1

16.7 通过实际值补偿回参考点

16.7.1 对回参考点测量系统进行实际值补偿

功能

在对回参考点测量系统进行实际值补偿中，根据成功执行的机床轴测量系统回参考点运行，系统会将得到的绝对位置实际值直接传输至该机床轴的其他测量系统，并将其作为已回参考点：

DB31, ... DBB60.4 / 60.5（已回参考点/已同步 1/2）= 1

优点

在从实际回参考点的测量系统转换为通过实际值补偿回参考点的测量系统时，机床轴可继续保持在闭环控制下（伺服使能生效），因为补偿的实际值不会产生实际值跃变。

说明

为了获取测量系统专用的编码器信息来提升定位精度，建议在测量系统切换后对通过实际值补偿回参考点的测量系统重新执行回参考点。

激活

对回参考点测量系统的实际值补偿针对机床激活：

MD34102 $MA_REFP_SYNC_ENCS = 1

16.7.2 使用距离编码参考脉冲的测量系统中的实际值补偿

功能

为了获取测量系统专用的编码器信息来提升定位精度，建议在测量系统切换后对通过实际值补偿回参考点的测量系统重新执行回参考点。

若将使用距离编码参考脉冲的编码器用于被动测量系统，那么在满足以下前提条件时可不执行回参考点：

1. 主动测量系统：间接测量系统（电机测量系统），例如使用绝对值编码器
2. 被动测量系统：使用距离编码参考脉冲的直接测量系统
3. 切换测量系统前，对配备已回参考点的间接测量系统的机床轴执行运行，使其越过回参考点所需数量的零脉冲。此时被动直接测量系统自动成为已回参考点状态。

参数设置

除了针对各测量系统的回参考点专用机床数据外，还须设置以下机床数据：

- 使能实际值补偿：
  MD34102 $MA_REFP_SYNC_ENCS = 1
- 使用距离编码参考标记的直接测量系统：
  - MD34200 $MA_ENC_REFP_MODE[*测量系统*] = 3
    距离编码的参考脉冲
  - MD30242 $MA_ENC_IS_INDEPENDENT[*测量系统*] = 2
    实际值补偿中，系统将为被动直接测量系统补偿生效的间接测量系统的位置实际值，但是不会将其标记为已回参考点。在越过设定数量的参考脉冲后，该被动直接测量系统自动成为已回参考点状态。回参考点可在任意运行方式下进行。

过程

1. 初始情况：两个测量系统均未回参考点：
   DB31, ... DBX60.4 = 0（已回参考点/已同步 1）
   DB31, ... DBX60.5 = 0（已回参考点/已同步 2）
2. 根据测量系统类型对间接测量系统执行回参考点：
   DB31, ... DBX60.4 = 1（已回参考点/已同步 1）
   DB31, ... DBX60.5 = 0（已回参考点/已同步 2）
3. 运行机床轴使其越过设定数量的参考脉冲。
   借此直接测量系统自动回参考点：
   DB31, ... DBX60.4 = 1（已回参考点/已同步 1）
   DB31, ... DBX60.5 = 1（已回参考点/已同步 2）

16.8 跟踪模式下回参考点

功能

增量测量系统和使用距离编码参考脉冲的测量系统也可在机床轴的跟踪模式下进行回参考点。前提条件是根据使用的测量系统对回参考点运行进行了正确地设置（参见“使用增量测量系统回参考点 (页 1355)”和“通过距离编码的参考脉冲回参考点 (页 1369)”章节）。

在跟踪模式下的回参考点中，机床轴不是通过 NC，而是借助穿过编码器零脉冲或设定数量的距离编码参考脉冲的外部运行来移动。识别出编码器零脉冲或设定数量的距离编码参考脉冲时，测量系统即完成回参考点。

说明

回参考点结果的可重复性

在 NC 控制的回参考点运行中，在运行期间遵循参数设置的运行速度即可保证回参考点结果的可重复性。在跟踪模式下回参考点时，机床制造商/用户必须采取合适的措施保证回参考点结果的可重复性。

零脉冲的唯一性

增量测量系统的回参考点基于编码器零脉冲相对机床轴整个运行范围的唯一位置。

在跟踪模式下回参考点时不会由 NC 分析参考凸轮信号，因此在未采取其他措施时，只能在以下情形下确定唯一的参考点：

- 机床轴运行范围内只有一个编码器零脉冲。
- 使用绝对编码参考脉冲的线性测量系统。
- 模数回转轴（一次旋转内的绝对位置）

存在多个零脉冲信号时的零脉冲选择

若在特定的机床条件下（例如编码器和负载间的降档齿轮箱）能够在机床轴的运行范围内识别出多个编码器零脉冲，则须将一个无接触式接近开关信号连接至 NCU 接口的数字量输入，从而通过该信号来确定唯一的零脉冲。

说明

无接触式接近开关信号：零脉冲选择

无接触式接近开关信号只在驱动中处理。连接和参数设置参见“硬件信号 (页 1355)”章节。

使能

“跟踪模式下回参考点”功能通过以下机床数据使能：

MD34104 $MA_REFP_PERMITTED_IN_FOLLOWUP = TRUE

启动回参考点运行

启动回参考点运行时，若机床轴处于跟踪模式下（DB31, ... DBX61.3 == TRUE），那么测量系统会在跟踪模式下进行回参考点。

若机床轴在启动回参考点运行时不处于跟踪模式下，则会执行“一般的”由 NC 控制的回参考点运行。

跟踪模式下的回参考点可在以下运行方式中启动：

- JOG-REF：运行键
- AUTOMATIC：零件程序指令 G74

回参考点运行的过程（JOG-REF 运行方式）

1. 激活机床轴跟踪模式：
   DB31, ... DBX1.4（跟踪运行）= 1
   DB31, ... DBX2.1（伺服使能）= 0
2. 等待激活跟踪运行：
   DB31, ... DBX61.3（跟踪生效）== 1
3. 切换至 JOG 运行方式，机床功能 REF。

4. 外部运行机床轴，使其越过编码器零脉冲或设定数量的距离编码参考脉冲。机床轴移动后，立即在 NC 内部启动回参考点：系统会复位以下 NC/PLC 接口信号作为反馈信息：
   DB31, ... DBX61.4（轴/主轴停止）== 0
5. 成功识别出编码器零脉冲或设定数量的距离编码参考脉冲时，测量系统即完成回参考点。系统会置位以下 NC/PLC 接口信号作为反馈信息：
   DB31, ... DBX60.4/60.5（已回参考点/已同步 1/2）== 1

终止回参考点运行

以下情形下，生效的回参考点运行将终止：

- 取消跟踪模式
- NC 复位

已回参考点的测量系统的特性

已回参考点的测量系统只能在 AUTOMATIC 运行方式下通过零件程序指令 G74 重新执行回参考点。

回参考点运行的过程（AUTOMATIC 运行方式）

1. 切换至 AUTOMATIC 运行方式。
2. 启动零件程序。
3. 激活机床轴跟踪模式：
   DB31, ... DBX1.4（跟踪运行）= 1
   DB31, ... DBX2.1（伺服使能）= 0
4. 等待激活跟踪运行：
   DB31, ... DBX61.3（跟踪生效）== 1
5. 执行指令 G74 后，立即在 NC 内部启动回参考点：
6. 外部运行机床轴，使其越过编码器零脉冲或设定数量的距离编码参考脉冲。
7. 成功识别出编码器零脉冲或设定数量的距离编码参考脉冲时，测量系统即完成回参考点。系统会置位以下 NC/PLC 接口信号作为反馈信息：
   DB31, ... DBX60.4 / 60.5（已回参考点/已同步 1/2）== 1
8. 回参考点运行完成后执行程序段切换。

终止回参考点运行

以下情形下，生效的回参考点运行将终止：

- 取消跟踪模式
- NC 复位

已回参考点的测量系统的特性

已回参考点的测量系统可重新执行回参考点。

16.9 使用绝对值编码器回参考点

16.9.1 校准信息

使用绝对值编码器的机床轴

使用绝对值编码器的机床轴的优势在于：在经过一次校准后，机床轴的实际值系统将立即与确定的绝对位置同步，无需像增量测量系统一样执行回参考点运行（例如控制系统启动、取消机床轴“驻留”等）。

校准

校准绝对值编码器时，编码器的实际值会通过机床零点一次性补偿，之后设置生效。

机床轴连接的绝对值编码器的当前校准状态会在以下轴专用机床数据中显示：

MD34210 $MA_ENC_REFP_STATE（绝对值编码器的状态）

值	含义
0	编码器未校准
1	编码器校准已使能
2	编码器已校准

校准方式

支持以下校准方式：

- 通过输入参考点偏移校准
- 通过输入参考点值校准
- 通过测头自动校准
- 通过无接触式接近开关校准

重新校准

以下情形下必须重新校准绝对值编码器：

- 负载和绝对值编码器之间的齿轮档切换
- 拆卸/安装绝对值编码器

- 拆卸/安装带绝对值编码器的电机
- 静态 NC 存储器中数据丢失
- 电池掉电
- 实际值设置（PRESETON）

⚠ 警告

数据备份

备份机床 A 的机床数据时，机床轴的编码器状态（MD34210）也会一同被备份。

在将该数据组载入至相同类型的机床 B 时（例如批量调试或维修后），NC 会将相关机床轴自动视为以校准/已回参考点。在此类型情形下，用户/机床制造商须自行负责执行重新校准。

另见机床数据说明：

MD30250 $MA_ACT_POS_ABS（关闭时间点的绝对编码器位置）

说明

控制系统只会在以下情形下识别出绝对值编码器需要重新校准：

- 齿轮箱切换，传动比切换
- 零脉冲监控响应
- 更换绝对值编码器后出现新的编码器序列号

在这些情形下控制系统会将绝对值编码器的状态设置为“0”。

MD34210 $MA_ENC_REFP_STATE = 0（编码器未校准）

此时会显示以下报警：
报警 25022“轴 <轴名称> 编码器 <编号> 警告 0”
零脉冲监控响应时还会额外显示以下报警：
报警 25020“轴 <轴名称> 零脉冲监控生效编码器”

在所有其他情形下（例如 PRESETON），用户须自行通过将编码器状态**手动**设置为“0”来取消校准，之后重新执行校准。

16.9.2 通过输入参考点偏移校准

功能

在通过输入参考凸轮偏移校准时，先会确定操作界面中显示的位置实际值和机床中的实际值的差值，并将其作为参考点偏移提供给 NC。

步骤

1. 测定仪机床零点为基准的机床轴位置，例如通过：
   – 测量位置（例如激光干涉仪）
   – 将机床轴运行至一个已知位置（例如固定停止点）
2. 在操作界面上读取显示的机床轴位置实际值。
3. 计算参考点偏移（第 1 步和第 2 步中获取的位置实际值的差值）并输入机床数据：
   MD34090 $MA_REFP_MOVE_DIST_CORR（参考点偏移）
4. 此时该绝对值编码器被标记为已校准：
   MD34210 $MA_ENC_REFP_STATE = 2

   说明

   编码器校准要在下一次激活编码器（例如控制系统启动）时才生效。

5. 上电，触发复位。
6. 在操作界面上检查显示的机床轴位置。

   说明

   间隙补偿

   若为使用绝对值编码器的测量系统设置了间隙补偿，则须注意：

   将机床轴运行至校准机床位置时不得存在间隙！

   持续激活参考点偏移

   上电复位后，输入的参考点偏移（MD34090）才会持续生效。 校准完绝对值编码器后，若在未上电复位的情况下运行机床轴，那么机床数据中输入的参考点偏移可能会在内部溢出补偿中被改写。

   检查位置实际值

   在校准绝对值编码器后，建议在下一次控制系统启动（上电）后检验机床轴的位置实际值。

16.9.3 通过输入参考点值校准

功能

在通过输入参考点值校准时，会以机床零点为基准测定机床轴的绝对位置，例如通过：

- 测量位置（例如激光干涉仪）
- 将机床轴运行至一个已知位置（例如固定停止点）

测得的值作为参考点值提供给 NC。之后 NC 会通过编码器绝对值和参考点值的差值计算出对应的参考点偏移。

步骤

1. 检查回参考点模式是否被设为“接收参考点值”：
   MD34200 $MA_ENC_REFP_MODE == 0
   如果没有，则在机床数据中输入值 0 并触发上电复位。
2. 在 JOG 运行方式下将机床轴运行至待测位置（例如：机床干涉仪）或已知位置（例如：固定挡块）。

   说明

   可通过运行键沿“为回参考点运行使能的方向”运行机床轴：

   MD34010 $MA_REFP_CAM_DIR_IS_MINUS（沿负方向逼近参考点）

   为了避免驱动组中的间隙引起的位置错误，必须以较低的速度逼近已知位置。
3. 将以机床零点为基准的机床轴位置作为参考点值输入到以下机床数据中：
   MD34100 $MA_REFP_SET_POS = *位置*
4. 使能编码器校准：
   MD34210 $MA_ENC_REFP_STATE = 1
5. 触发上电复位以接收输入的机床数据值。
6. 切换至 JOG-REF 运行方式。
7. 按下步骤 2 中使用过的、使能用于回参考点的运行键。
   按下运行键时机床轴不会移动！
   NC 通过输入的参考点值和绝对值编码器发送的值计算出参考点偏移。计算结果记录至以下机床数据：
   MD34090 $MA_REFP_MOVE_DIST_CORR（参考点偏移）
   该绝对值编码器的状态会被设置为“编码器已校准”：
   MD34210 $MA_ ENC_REFP_STATE = 2
   机床轴的实际值系统会进行同步。
   就此机床轴已完成回参考点。根据生效的测量系统，NC 会置位对应的结构信号作为反馈：
   DB31, ... DBB60.4 / 60.5（已回参考点/已同步 1/2）= 1
8. 上电，触发复位。

   说明

   持续激活参考点偏移

   上电复位后，输入的参考点偏移（MD34090）才会持续生效。

   校准完绝对值编码器后，若在未上电复位的情况下运行机床轴，那么机床数据中输入的参考点偏移可能会在内部溢出补偿中被改写。

   检查位置实际值

   在校准绝对值编码器后，建议在下一次控制系统启动（上电）后检验机床轴的位置实际值。

16.9.4 通过测头自动校准

功能

使用测头自动校准时，机床轴会通过零件程序逼近机床中的一个已知位置。 该位置值作为参考点值保存在 NC 中。 一旦测头切换，即表示到达该位置，此时 NC 会通过编码器值和参考点值的差值计算出对应的参考点偏移。

说明

用于自动校准的零件程序

机床制造商/用户必须根据机床规格自行创建用于通过测头自动校准的零件程序。

无碰撞

由于没有为待回参考点的机床轴启用实际值监控，因此机床轴运行期间操作人员尤其要确保机床中不会发生碰撞。

零件程序

用于通过测头自动校准绝对值编码器的零件程序必须能为每根待回参考点的轴按顺序执行以下操作：

1. 逼近机床轴的校准位置，该位置通过测头响应进行识别。
   逼近应从相同的方向以逐级降低的速度分多次执行，从而获取尽可能高的测量精度。 测量值保存在系统变量 $AA_IM 中。
2. 计算和写入参考点偏移：

   MD34090 $MA_REFP_MOVE_DIST_CORR = MD34100 $MA_REFP_SET_POS – $AA_IM

3. 将绝对值编码器状态设置为“编码器已校准”。
   MD34210 $MA_ ENC_REFP_STATE = 2

过程

通过测头自动校准时，执行以下步骤：

1. 使能零件程序启动，包含未回参考点的机床轴：
   MD20700 $MC_REFP_NC_START_LOCK = 0
2. 为所有相关机床轴输入测头切换时机床轴相对机床零点的位置，作为参考点值：
   MD34100 $MA_REFP_SET_POS = 参考点值
3. 触发 NC 复位以接收输入的机床数据值。
4. 启动零件程序。

5. 零件程序运行完毕后，在未回参考点的机床轴上重新禁用零件程序启动：
   MD20700 $MC_REFP_NC_START_LOCK = 1
6. 触发上电复位，从而使通过零件程序写入的参考点偏移持续生效：
   MD34090 $MA_REFP_MOVE_DIST_CORR（参考点偏移）

说明

持续激活参考点偏移

上电复位后，输入的参考点偏移（MD34090）才会持续生效。

校准完绝对值编码器后，若在未上电复位的情况下运行机床轴，那么机床数据中输入的参考点偏移可能会在内部溢出补偿中被改写。

检查位置实际值

在校准绝对值编码器后，建议在下一次控制系统启动（上电）后检验机床轴的位置实际值。

16.9.5 通过 BERO 校准

功能

在通过无接触式接近开关校准时，和使用增量测量系统时一样需要向定义的机床位置执行回参考点运行。此时无接触式接近开关用于代替使用绝对值编码器时不存在的编码器零脉冲。回参考点运行完成后，NC 会通过编码器绝对值和参数设置的参考点值的差值自动计算出参考点偏移。

参数设置

NC：回参考点模式

回参考点模式必须设置为“增量、旋转或线性测量系统的回参考点：编码器信号上的零脉冲”：

MD34200 $MA_ENC_REFP_MODE[<轴>] = 1

NC：参考点值

参考点值通过以下机床数据设置：

MD34100 $MA_REFP_SET_POS[<轴>] = <参考点值>

驱动：替代零脉冲

在参数 p0494 或 p0495 中必须设置连接无接触式接近开关的 NCU 接口数字量输入。

执行

回参考点运行可在 JOG-REF 运行方式下手动启动，或在 AUTOMATIC 或 MDI 运行方式下通过零件程序（G74）启动。

回参考点运行完成后，绝对值编码器变为已校准状态，机床轴的实际值系统也完成同步。

根据生效的测量系统，NC 会为机床轴置位以下 NC/PLC 接口信号，作为对 PLC 用户程序的反馈：

DB31, ... DBB60.4 / 60.5（已回参考点/已同步 1/2）= 1

说明

若在绝对值编码器校准后移除无接触式接近开关，则须将回参考点模式修改为“使用绝对值编码器回参考点”：

MD34200 $MA_ENC_REFP_MODE[<轴>] = 0

16.9.6 使用绝对值编码器回参考点

设置

运行使能

机床轴将校准过的绝对值编码器作为生效测量系统时，若为其触发回参考点运行（在 JOG-REF 运行方式下手动触发，或通过零件程序指令 G74 自动触发），那么该轴的运动特性取决于设置的运行使能：

MD34330 $MA_REFP_STOP_AT_ABS_MARKER = <值>

值	含义
0	运行已**使能**。 触发回参考点后，机床轴运行至回参考点位置。到达参考点后，回参考点运行结束。
1	运行**未使能**。 触发回参考点后，机床轴不运行，回参考点运行立即结束。

16.9.7 通过旋转绝对值编码器启用替代零脉冲回参考点

功能

为了在使用绝对值编码器时也能采用零脉冲回参考点（参见章节“使用增量测量系统回参考点 (页 1355)”），则在使用绝对值编码器时不要根据硬件对现有零脉冲进行模拟。这样，在编码器的每转中控制系统都会在同一位置上生成一个替代零脉冲信号。

与通过增量编码器回参考点的区别

启用替代零脉冲的绝对值编码器不能视作完全等同于增量编码器。绝对值编码器的所有属性将保留。下表列出了增量编码器和绝对值编码器的不同属性：

表格 16-1 增量编码器和绝对值编码器的属性

属性	增量编码器	绝对值编码器
编码器类型	MD30240 $MA_ENC_TYPE =	
	1	4
内部编码器位置	MD30250 $MA_ACT_POS_ABS =	
	值仅在 MD34210 ≥ 1 时更新	值仅在 MD30270 = 0 时更新
运行范围扩展	MD30270 $MA_ENC_ABS_BUFFERING =	
	无作用	= 0（默认）：生效
参考点偏移	MD34090 $MA_REFP_MOVE_DIST_CORR =	
	允许输入数值	只由控制系统更新数值
不支持的回参考点类型	MD34200 $MA_ENC_REFP_MODE =	
	1, 3, 4, 8	0, 1
校准状态	MD34210 $MA_ENC_REFP_STATE = 0, 1, 2	
	运行中关闭时，编码器校准自动失效	包含位置跃变的参数值切换或序列号变更时，编码器校准自动失效
模数区域绝对位置	MD34220 $MA_ENC_ABS_TURNS_MODULO =	
	0	1 - 4096
编码器序列号	MD34230 $MA_ENC_SERIAL_NUMBER =	
	0	每次编码器切换时必须由 PLC 更新数值，否则校准和报警会丢失。

属性	增量编码器	绝对值编码器
接收批量调试文件	无限制	由于编码器属性（MD30250、MD30270、MD34090、MD34210、MD34220、MD34230）必须加以限制。
激活时间	0 秒	数秒
零脉冲	编码器每转 1 个	无
零脉冲监控	硬件	软件
上电后无实际值缓存时的位置	0.0	MD34220 内的最后位置
	MD34210 = 0	MD30270 = 1
上电后有实际值缓存时的位置	关闭前最后的静态位置	断电期间最后位置，含微小运行
	MD34210 = 1	MD30270 = 0
上电后的回参考点状态	取决于校准状态	

前提条件

此功能只适用于旋转绝对值编码器：

- MD31000 $MA_ENC_IS_LINEAR == 0
- MD30240 $MA_ENC_TYPE == 4

参数设置

通过替代零脉冲回参考点：

MD34200 $MA_ENC_REFP_MODE = 1

功能的基本条件

- 参考点偏移（MD34090 $MA_REFP_MOVE_DIST_CORR）不得进行参数设置：
  在使用绝对值编码器时，此参数描述的是机床零点和绝对值编码器零点之间的偏移，此处含义则发生变化。
- 负载侧零脉冲搜索速度 MD34040 $MA_REFP_VELO_SEARCH_MARKER 不可超出编码器绝对信号极限频率 MD36302 $MA_ENC_FREQ_LIMIT_LOW。
  速度过大时会无法读取绝对信息，也就无法生成替代零脉冲。

- 如果在所设置的路径上没有找到零脉冲（MD34060 $MA_REFP_MAX_MARKER_DIST），则会显示报警：
- 除了最后位置外，若还需在绝对值编码器上掉电保持回参考点状态，必须设置以下机床数据：
  MD34210 $MA_ENC_REFP_STATE = 2

数据备份与批量调试

绝对值编码器的某些属性会限制调试存档向其他机床的传输。在控制系统上运行了调试存档后，必须检查以下机床数据并进行必要的修改：

- MD30250 $MA_ACT_POS_ABS（内部编码器位置）
- MD30270 $MA_ENC_ABS_BUFFERING（运行范围扩展）
- MD34090 $MA_REFP_MOVE_DIST_CORR（绝对偏移）
- MD34210 $MA_ENC_REFP_STATE（校准状态）
- MD34220 $MA_ENC_ABS_TURNS_MODULO（模数区域）
- MD34230 $MA_ENC_SERIAL_NUMBER（编码器序列号）

16.9.8 激活测量系统

在以下情形下，机床轴的测量系统会被激活：

- 控制系统启动（上电）
- 通过接口信号激活测量系统（取消“驻留”）：
  DB31, ... DBB1.5 / 1.6（位置测量系统 1/2）
  DB31, ... DBB2.1（控制器使能）
- 低于参数设置的编码器极限频率（主轴）：
  MD36300 $MA_ENC_FREQ_LIMIT

测量系统激活时，NC 会将机床轴的实际值系统同步为当前绝对值。 同步期间轴运行会被禁止，主轴则不会。

参数设置编码器极限频率（主轴）

Heidenhain 公司的绝对值编码器 EQN 1325 具有一个增量信号和一个绝对信号。

若使用超出增量信号编码器极限频率的转速运行主轴，必须将相比要小得多的绝对信号极限频率设置为编码器极限频率：

MD36300 $MA_ENC_FREQ_LIMIT

否则，测量系统激活时低于设置的编码器极限频率将导致读取的绝对位置错误。 这会引起机床轴实际值系统中的位置偏移。

确定编码器极限频率

待设置的编码器极限频率通过以下极限转速中较小的一个得出，或通过两个转速得出：

- 编码器
  极限转速或编码器极限频率从编码器数据页获取（例如极限转速 2000 [rpm]）
- NC
  取决于 NC 内部的分析过程，在确保 NC 能正确测算绝对值的情形下，极限转速为每个插补周期 4 编码器旋转。
  例如，在插补周期为 12 ms 时：极限转速 = 4 / 12 ms = 20000 rpm
  极限转速对应的极限频率如下计算：

$$MD36300 = \frac{4 * MD31020}{MD10050 * MD10070}$$

MD31020 $MA_ENC_RESOL（每转的编码器线数）
MD10050 $MN_SYSCLOCK_CYCLE_TIME（系统基本周期）
MD10070 $MN_IPO_SYSCLOCK_TIME_RATIO（插补周期系数）

说明

主轴相关的位置闭环控制激活转速必须根据主轴的绝对值编码器的编码器极限频率设置：
MD35300 $MA_SPIND_POSCTRL_VELO（位置闭环控制激活转速）
MD36300 $MA_ENC_FREQ_LIMIT（编码器极限频率）

16.9.9 不支持的回参考点方式

下列回参考点方式不适用于绝对值编码器：

- 通过编码器零脉冲回参考点/校准
- 距离编码的参考脉冲
- 包含两个脉冲沿分析的无接触式接近开关

16.10 自动恢复机床参考

若未定义机床参考，那么在运行机床轴时将无法执行位置相关功能，例如转换或刀具框架。 在很多情形下，编码器激活时就必须启用这些功能用于轴的运行，例如控制系统激活后或退出“‘驻留’机床轴 (页 131)”后。但同时又不适合（或者无法）运行机床轴来重新回参考点。

绝对值编码器

对于使用校准过的绝对值编码器的测量系统，无需采取额外措施便可通过读取编码器值立即恢复机床参考。

增量编码器

对于增量测量系统，在不运行轴的情况下可通过“自动回参考点”或“恢复实际位置”恢复机床参考。

前提条件

⚠ 警告

因机床轴实际位置偏差造成的位置测量系统同步故障

机床轴测量系统关闭期间，不允许对其进行机械移动。 否则会使缓存中保存的最后实际位置和机床轴的实际位置之间产生偏移。 这可能会造成测量系统同步错误，并对人员和机械造成危险。

机床制造商须在机床上提供相应措施保证实际位置不变，用户须遵循这些措施。 相关责任由机床制造商/用户自行担负。

若机床的构造决定了关闭状态下轴的移动无法避免，则必须使用绝对值编码器，或在上电后重新执行回参考点。

说明

编码器模块 SMExx

自动回参考点以及重启后将实际位置恢复为缓存中的最后位置这两个功能都必须有编码器模块 SMExx（Externaly Mounted，外部安装）作为支持。 使用编码器模块 SMCxx（Cabinet，机柜安装）或 SMIxx（集成）时，在控制系统重启（上电）后无法恢复实际位置。 机床轴的测量系统必须重新执行回参考点。

16.10.1 自动回参考点

功能

启用自动回参考点功能时，控制系统接通后，生效的机床轴测量系统的实际位置会被设置为缓存中最后保存的位置，且编码器状态会被设置为“已回参考点”。这样一来，在 AUTOMATIC 和 MDI 运行方式下便可在控制系统启动后直接启动程序。

前提条件

- 控制系统接通后生效的测量系统必须已在系统关闭前执行了首次回参考点。
- 控制系统关闭时，机床轴必须处于“精准停”（DB31, ... DBX60.7 == 1）状态。

说明

若系统关闭时机床轴不处于“精准停”状态，那么上电后实际位置将以“0”值初始化。编码器状态则会显示“未回参考点”。

参数设置

自动回参考点通过将编码器状态设置为“自动回参考点已使能，但编码器尚未回参考点”来使能：

MD34210 $MA_ENC_REFP_STATE[<编码器>] = 1

测量系统回参考点后，编码器状态中会显示“在下一次激活编码器时执行自动回参考点”：

MD34210 $MA_ENC_REFP_STATE[<编码器>] == 2

NC/PLC 接口信号

自动回参考点后，生效测量系统的编码器状态会显示为“已回参考点”：

DB31, ... DBX60.4 / .5 == 1（已回参考点/已同步 1 / 2）

前提条件

MD34210 $MA_ENC_REFP_STATE[<编码器>] == 1 时的编码器激活

激活编码器时，编码器状态等于“1”表示“自动回参考点”已使能。但测量系统尚未回参考点，或在机床轴静止时未在“精准停”状态下关闭。此时对机床轴或生效的测量系统会进行以下设置：

- 实际位置 = 0
- 生效的系统，编码器状态 = “未回参考点”：
  DB31, ... DBX60.4 / .5 = 0（已回参考点/已同步 1 / 2）

文档

功能手册之基本功能分册，章节“R1 回参考点” >“使用增量测量系统回参考点 (页 1355)”

16.10.2 恢复实际位置

功能

将实际位置恢复为缓存中最后保存的位置时，生效测量系统的编码器状态会被设置为“**已恢复**”。轴只可手动运行。

AUTO 运行方式

为了在 AUTOMATIC 运行方式下使能 NC 启动来实现程序的自动执行，必须对机床轴的测量系统重新执行回参考点。

MDI 运行方式和溢出转存

在 MDI 运行方式和溢出转存中，也可采用恢复后的位置进行加工，而无需执行轴回参考点。为此，必须针对特定通道明确使能系统恢复后的位置启动：

MD20700 $MC_REFP_NC_START_LOCK = 2

前提条件

控制系统接通后生效的测量系统必须已在系统关闭前执行了首次回参考点。

参数设置

使能：恢复实际位置

恢复实际位置通过将编码器状态设置为“恢复关闭前缓存中最后保存的位置，不自动回参考点”来使能：

MD34210 $MA_ENC_REFP_STATE[<编码器>] = **3**

使能：NC 启动，用于“MDI”和“溢出转存”运行方式

NC 启动通过以下机床数据使能，用于在“MDI”和“溢出转存”运行方式下执行零件程序或零件程序段，此时测量系统状态为“位置已恢复”：

MD34110 $MA_REFP_CYCLE_NR **≠ -1**（通道专用回参考点运行时的回参考点）

MD20700 $MC_REFP_NC_START_LOCK = **2**（不带参考点的 NC 启动禁用）

NC/PLC 接口信号

恢复的实际位置不被视作与回参考点后的实际位置有同等质量。因此机床轴测量系统的状态会显示“位置已恢复”，而不是“已回参考点/已同步”。

实际位置已恢复：

- DB31, ... DBX60.4 / .5 = **0**（已回参考点/已同步 1 / 2）
- DB31, ... DBX71.4 / .5 = **1**（**位置已恢复**，编码器 1 / 2）

测量系统已回参考点：

- DB31, ... DBX60.4 / .5 = 0 → **1**（**已回参考点/已同步** 1 / 2）
- DB31, ... DBX71.4 / .5 = 1 → **0**（位置已恢复，编码器 1 / 2）

说明

在“位置已恢复”状态下，运行范围限制监控（软件限位开关、工作区域限制等）已生效。

前提条件

主轴

超出编码器极限频率时，主轴会被复位为“未回参考点/同步”状态：

- DB31, ... DBX60.4 / .5 = 1 → **0**（已回参考点/已同步 1 / 2）
- DB31, ... DBX71.4 / .5 = 1 → **0**（位置已恢复，编码器 1 / 2）

文档

功能手册之基本功能分册，章节“R1 回参考点” >“使用增量测量系统回参考点 (页 1355)”

16.11 前提条件

16.11.1 较大运行范围

直线轴，运行范围 > 4096 编码器转数，配备旋转绝对值编码器 EQN 1325，设置的绝对值编码器区域 MD34220 $MA_ENC_ABS_TURNS_MODULO = 4096：

最大运行范围等同于增量编码器。

使用绝对值编码器的连续旋转回转轴：

- 支持任意的整数传动比。
- 建议将使用绝对值编码器的连续旋转回转轴参数设置为模数轴（运行范围 0...360 度）：
  MD30310 $MA_ROT_IS_MODULO = 1
  否则测量系统激活后该回转轴可能需要非常大的运行行程，用于运行至绝对零。

使用绝对值编码器的机床轴：

为了确保测量系统重新激活后控制系统能够正确测算位置实际值，测量系统关闭时机床轴只允许移动小于绝对值编码器区域的一半的位移。

MD34220 $MA_ENC_ABS_TURNS_MODULO

对编码器位置唯一性的提示

说明

线性绝对值编码器

线性编码器（例如 Heidenhain LC181）的绝对值在可提供的标度长度内始终为唯一值。

旋转绝对值编码器

旋转绝对值编码器的绝对值只在最大编码器转数范围内具有唯一性。

例如，Heidenhain 公司的旋转绝对值编码器 EQN 1325 能够在 0 至 4096 的转数范围内提供唯一绝对值。

根据安装条件将得出：

- 负载上带编码器的回转轴：4096 负载转数
- 电机上带编码器的回转轴：4096 电机转数
- 电机上带编码器的直线轴：4096 电机转数

示例：

旋转绝对值编码器 EQN 1325 安装到了一根直线轴的电机上。 有效主轴螺距为 10 mm 时，运行范围内的唯一绝对值为 -20.48 m 至 +20.48 m。

16.12 数据表

16.12.1 机床数据

16.12.1.1 NC 专用机床数据

编号	名称：$MN_	说明
11300	JOG_INC_MODE_LEVELTRIGGRD	长按模式/点按模式中 INC/REF

16.12.1.2 通道专用机床数据

编号	名称：$MC_	说明
20700	REFP_NC_START_LOCK	不带参考点的 NC 启动禁用

16.12.1.3 进给轴/主轴专用机床数据

编号	名称：$MA_	说明
30200	NUM_ENCS	编码器数量
30240	ENC_TYP	实际值编码器类型
30242	ENC_IS_INDEPENDENT	编码器是独立的
30250	ACT_POS_ABS	绝对编码器位置至关闭时间点
30270	ENC_ABS_BUFFERING	绝对值编码器：运行范围扩展
30300	IS_ROT_AX	回转轴/主轴
30310	ROT_IS_MODULO	回转轴/主轴的模数转换
30330	MODULO_RANGE	模数区域的大小
30340	MODULO_RANGE_START	模数区域的起始位置
30355	MISC_FUNCTION_MASK	轴功能
31122	BERO_DELAY_TIME_PLUS	正方向上的 BERO 延时
31123	BERO_DELAY_TIME_MINUS	负方向上的 BERO 延时
34000	REFP_CAM_IS_ACTIVE	带参考凸轮的轴
34010	REFP_CAM_DIR_IS_MINUS	沿负方向回参考点
34020	REFP_VELO_SEARCH_CAM	回参考点速度
34030	REFP_MAX_CAM_DIST	到达参考凸轮的最大位移
34040	REFP_VELO_SEARCH_MARKER	参考点蠕变速度
34050	REFP_SEARCH_MARKER_REVERSE	参考凸轮上换向
34060	REFP_MAX_MARKER_DIST	到达参考脉冲的最大位移 启用 2 个距离编码参考脉冲时的最大行程
34070	REFP_VELO_POS	参考点定位速度
34080	REFP_MOVE_DIST	启用距离编码系统时的参考点距离/目标点
34090	REFP_MOVE_DIST_CORR	参考点/绝对值偏移，距离编码
34092	REFP_CAM_SHIFT	提供等距零脉冲的增量测量系统的电子参考凸轮偏移
34093	REFP_CAM_MARKER_DIST	参考凸轮与参考脉冲间距离
34100	REFP_SET_POS	参考点值
34102	REFP_SYNC_ENCS	对回参考点测量系统进行实际值补偿
34104	REFP_PERMITTED_IN_FOLLOWUP	使能跟踪模式下回参考点

编号	名称：$MA_	说明
34110	REFP_CYCLE_NR	通道相关轴顺序回参考点
34120	REFP_BERO_LOW_ACTIVE	BERO 凸轮的极性切换
34200	ENC_REFP_MODE	回参考点方式
34210	ENC_REFP_STATE	绝对值编码器状态
34220	ENC_ABS_TURNS_MODULO	旋转编码器的绝对值范围
34230	ENC_SERIAL_NUMBER	编码器序列号
34232	EVERY_ENC_SERIAL_NUMBER	编码器序列号生效范围
34300	ENC_REFP_MARKER_DIST	使用距离编码编码器时参考脉冲的基本距离
34310	ENC_MARKER_INC	使用距离编码测量系统时两个参考脉冲的距离差
34320	ENC_INVERS	长度测量系统与机床系统方向相反
34330	REFP_STOP_AT_ABS_MARKER	无目标点的距离编码长度测量系统
35150	SPIND_DES_VELO_TOL	主轴转速公差
36302	ENC_FREQ_LIMIT_LOW	编码器极限频率重新同步
36310	ENC_ZERO_MONITORING	零脉冲监控

16.12.2 信号

16.12.2.1 发送至 BAG 的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
BAG 复位	DB11.DBX0.7	DB3000.DBX0.7
机床功能 REF	DB11.DBX1.2	DB3000.DBX1.2

16.12.2.2 从 BAG 发出的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
机床功能 REF 生效	DB11.DBX5.2	DB3100.DBX1.2

16.12.2.3 发送至通道的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
激活回参考点	DB21, ...DBX1.0	DB320x.DBX1.0
OEM 通道信号：(HMI → PLC) REF	DB21, ...DBX28.7	-

16.12.2.4 来自通道的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
回参考点生效	DB21, ...DBX33.0	DB330x.DBX1.0
复位	DB21, ...DBX35.7	DB330x.DBX3.7
所有需要回参考点的轴均已回参考点	DB21, ...DBX36.2	DB330x.DBX4.2

16.12.2.5 发送至进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
跟踪模式（请求）	DB31, ...DBX1.4	DB380x.DBX1.4
位置测量系统 1/位置测量系统 2	DB31, ...DBX1.5-6	DB380x.DBX1.5-6
参考点值 1 至 4	DB31, ...DBX2.4-7	DB380x.DBX2.4-7
运行键 -/+	DB31, ...DBX4.6-7	DB380x.DBX4.6-7
回参考点减速度	DB31, ...DBX12.7	DB380x.DBX1000.7

16.12.2.6 来自进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
已回参考点，已同步 1 / 2	DB31, ...DBX60.4-5	DB390x.DBX0.4-5
跟踪生效	DB31, ...DBX61.3	DB390x.DBX1.3
运行指令 -/+	DB31, ...DBX64.6-7	DB390x.DBX4.6-7
已恢复 1 / 2	DB31, ...DBX71.4-5	DB390x.DBX11.4-5

S1：主轴

17

17.1 简要说明

主轴主要用于使刀具或工件进入旋转运动，从而为切削加工创造前提条件。

为此，根据机床的类型主轴须支持以下功能：

- 设定主轴旋转方向（M3、M4）
- 设定主轴转速（S、SVC）
- 无定向主轴停止（M5）
- 定向主轴停止 / 主轴定位
  （SPOS、M19 和 SPOSA）
- 齿轮档切换（M40 至 M45）
- 主轴模式/轴模式（主轴作为回转轴，或反之）
- 螺纹切削（G33、G34、G35）
- 刚性攻丝（G331、G332）
- 带补偿夹具的攻丝（G63）
- 旋转进给率（G95）
- 恒定切削速度（G96、G961、G97、G971）
- 可编程的主轴转速限制（G25、G26、LIMS）
- 主轴或主轴电机上可安装位置测量编码器
- 主轴最小/最大转速监控，以及
  最大编码器极限频率和主轴目标点监控
- 激活/取消位置闭环控制（SPCON、SPCOF、M70）
- 编写主轴功能：
  - 通过零件程序
  - 通过同步动作
  - 通过 PLC 使用 FC18，或通过特殊主轴接口用于简单主轴控制

17.2 运行方式

17.2.1 概述

主轴为无限旋转的回转轴，其能够在各运行模式中在位置闭环控制或转速闭环控制下运行。

运行模式

- 控制模式（转速闭环控制）
- 往复模式（转速闭环控制）
- 定位模式（位置闭环控制）
- 同步模式
  文档：
  功能手册，扩展功能；同步主轴（S3）。
- 刚性攻丝（G331 / G332）
  文档：
  编程手册，基本原理；章节：“行程指令” > “攻丝” > “刚性攻丝（G331 / G332）”
- 轴模式（位置闭环控制）

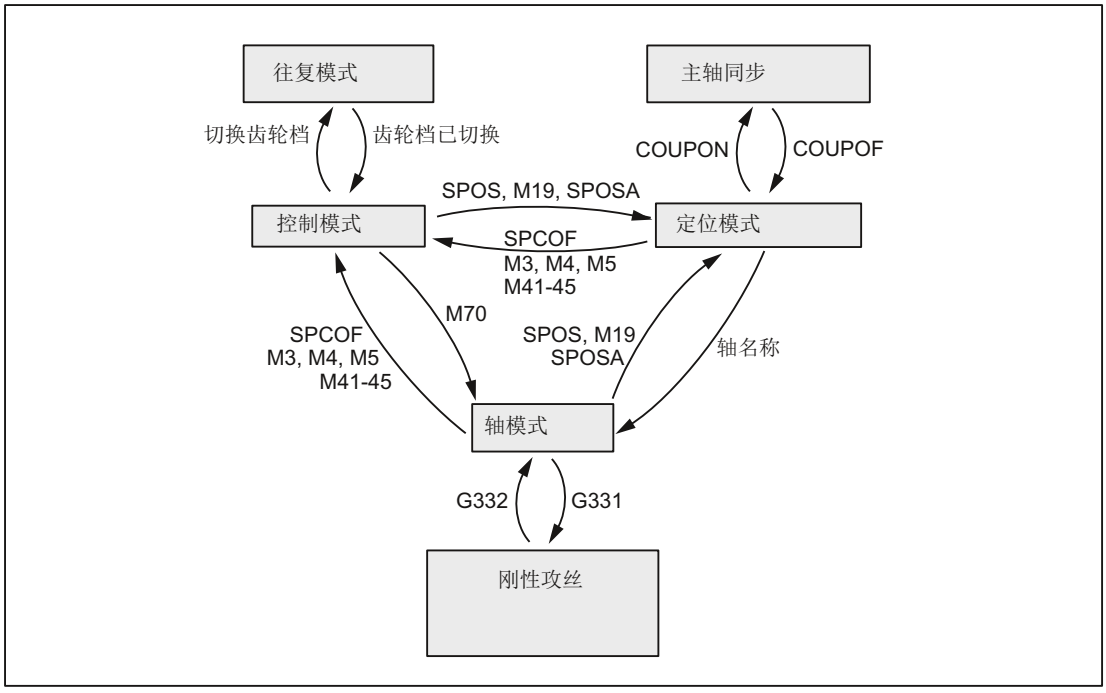
说明

轴/主轴模式切换

在轴模式和主轴模式共用一个电机的情况下，主轴可在轴模式与主轴模式之间切换。

17.2.2 运行方式切换

在主轴运行方式和轴运行方式之间可以如下切换：



- 控制模式 → 往复模式
  通过自动齿轮档切换（M40）和新的 S 值，或通过 M41 至 M45 设定了新的齿轮档时，主轴切换至往复模式。只有当新的齿轮档不等于当前实际齿轮档时，主轴才会切换至往复模式。

- 往复模式 → 控制模式
  如果创建了一个新的齿轮档，则接口信号：
  DB31, ... DBX84.6（往复模式）
  会被复位，并使用接口信号：
  DB31, ... DBX16.3（齿轮档已切换）
  切换至控制模式。
  最后编程的主轴转速（S 值）再次生效。

- 控制模式 → 定位模式

  如果要使主轴由旋转状态（M3 或 M4）定位停止，或者从静止状态（M5）重新定向，通过 SPOS、M19 或 SPOSA 切换至定位模式。

- 定位模式 → 控制模式
  若需退出主轴定向，则用 M3、M4 或 M5 转换到控制模式。最后编程的主轴转速（S 值）再次生效。

- 定位模式 → 往复模式
  若需退出主轴定向，可通过 M41 至 M45 切换至往复模式。齿轮档切换结束后，最后编程的主轴转速（S 值）和 M5（控制模式）再次生效。
- 定位模式 → 轴模式
  如果主轴已通过定向停止，则通过编写指定的轴名称切换到轴模式。齿轮档仍然保留。
- 控制模式 → 轴模式
  如果要从控制模式切换到轴模式，则也可以通过编程 M70 进行。 此时旋转的主轴如 M5 一样制动，位置闭环控制激活，且选择参数组零。
- 轴模式 → 控制模式
  若需结束轴模式，则可以通过 M3、M4 或 M5 切换到运行模式。最后编程的主轴转速（S 值）再次生效。
- 轴模式 → 往复模式
  若需结束轴模式，则可以通过 M41 至 M45 切换到往复模式（仅在编程的齿轮档不等于实际齿轮档时）。齿轮档切换结束后，最后编程的主轴转速（S 值）和 M5（控制模式）再次生效。

17.2.3 控制模式

何时为控制模式？

执行下列功能时，主轴处于控制模式下：

- 恒定主轴转速：
  - S...S... M3/M4/M5 和 G93, G94, G95, G97, G971
  - S...S... M3/M4/M5 和 G33, G34, G35
  - S...S... M3/M4/M5 和 G63
- 恒定切削速度：
  - G96/G961 S...M3/M4/M5

主轴必须未同步。

前提条件

将 M3/M4/M5 和下列功能配合使用时，必须使用主轴位置实际值编码器：

- 旋转进给率（G95）
- 恒定切削速度（G96、G961、G97、G971）

- 螺纹切削（G33、G34、G35）
- 刚性攻丝（G331、G332）
- 激活位置闭环控制（SPCON、M70）

将 M3/M4/M5 和下列功能配合使用时，不需要使用主轴位置实际值编码器：

- 反比时间进给率编码（G93）
- 进给速度，单位 mm/min 或 inch/min（G94）
- 带补偿夹具的攻丝（G63）

转速控制运行

需要恒定的主轴转速，而主轴位置不重要时，尤其适合使用转速控制运行（例如恒定的铣刀转速用于生成均匀的工件表面）

- 转速控制运行在零件程序中通过 M3、M4、M5 或 SPCOF 激活。
- 以下 NC/PLC 接口信号置位：
  DB31, ... DBX84.7（控制模式）
- 不启用位置闭环控制作业时，
  NC/PLC 接口信号：
  DB31, ... DBX61.5（位置控制器生效）复位。
- 转速控制运行中的加速度根据齿轮档在以下机床数据中定义：
  MD35200 $MA_GEAR_STEP_SPEEDCTRL_ACCEL
  数值应尽可能对应物理条件。

位置闭环控制运行

主轴位置需要在较长时间保持为已知，或需要激活同步主轴设定值耦合时，尤其适合使用位置闭环控制运行。

- 位置闭环控制运行在零件程序中通过以下指令激活：SPCON(<主轴编号>)
- 以下 NC/PLC 接口信号置位：
  DB31, ... DBX61.5（位置控制器生效）
- 位置闭环控制运行中的加速度根据齿轮档在以下机床数据中定义：
  MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL

独立主轴复位

复位或程序结束（M2、M30）后的株洲特性通过以下机床数据设置：

MD35040 $MA_SPIND_ACTIVE_AFTER_RESET（独立主轴复位）

值	含义
0	复位或程序结束时，主轴立即启用有效加速度制动至静止状态。最后编写的主轴转速和主轴旋转方向会被删除。
1	复位或程序结束时，保留最后编写的主轴转速（S 值）和主轴旋转方向（M3、M4、M5）。主轴不会被制动。

如果在复位或程序结束前恒定切削速度 G96、G961 生效，则当前的主轴转速（基于 100% 主轴倍率）会在系统内部作为最后编写的主轴转速接收。

只能通过 NC/PLC 接口信号使主轴静止：

DB31, ... DBX2.2（删除剩余行程/主轴复位）

出现为主轴生成快速停止的报警时，旋转方向会被删除。最后编程的主轴转速（S 值）保留。消除报警原因后必须重新启动主轴。

G96/G961 下的实际转速显示和主轴特性

DB31, ... DBX61.4（轴/主轴停止）

主轴被作为“停止”时的转速通过以下机床数据设置：

MD36060 $MA_STANDSTILL_VELO_TOL

指定该值时，须确保静止状态下以下 NC/PLC 接口信号保持：

DB31, ...DBX61.4（轴/主轴停止）

当系统提示 DB31,...DBX61.4（轴/主轴停止），且没有为主轴启用位置闭环控制时，操作界面上转速实际值会显示零值，且通过系统变量 $AA_S[<n>] 将读取零值。

恒定切削速度 G96/G961 下的主轴特性

- 加工开始时（从 G0 过渡至 Gx），以及 NC 停止、G60（模态准停）和 G9（逐段准停）后，系统会等待轨迹启动，直至转速实际值到达转速设定值的公差范围：
  DB31, ... DBX83.5（转速实际值 = 转速设定值）
- 即使设定了大幅度的转速变化（平面轴向位置 0 运行），NC/PLC 接口信号
  DB31, ... DBX83.5（转速实际值 = 转速设定值）
  和
  DB31, ... DBX83.1（转速设定值受限）
  也会设置为定义的值。

- 在低于最低转速
  或识别出 NC/PLC 接口信号
  DB31, ... DBX61.4（轴/主轴停止）
  时，NC/PLC 接口信号：
  DB31, ... DBX83.5（转速设定值 = 转速实际值）
  会被复位（例如用于机床紧急策略）。
- 已开始的轨迹加工（G64，精磨）不会被中断。

此外主轴特性还受以下机床数据影响：

MD35500 $MA_SPIND_ON_SPEED_AT_IPO_START（主轴在设定区域时的进给使能）

齿轮档切换结束时的主轴特性

- NC/PLC 接口信号：
  DB31, ... DBX16.3（齿轮档已切换）
  会提示 NC，新的齿轮档
  （NC/PLC 接口信号 DB31, ... DBX16.0-16.2 （实际齿轮档 A 至 C））
  生效，且往复模式结束。
  此时以下 NC/PLC 接口信号设置与否则不重要：
  DB31, ... DBX18.5（往复转速）

  实际齿轮档应与设定齿轮档相一致。
  选择参数组时，信号发送的实际齿轮档为相关信息。
- 由 PLC（DB31, ... DBX16.3）应答齿轮档切换（GSW）后，主轴处于转速控制运行（DB31, ... DBX84.7 = 1）中。
  若在 GSW 前编写了旋转方向（M3、M4、M5 或 FC18：“启动主轴旋转”）或主轴转速（S 值），那么最后编写的的旋转方向和转速将在 GSW 后重新生效。

17.2.4 往复模式

对于主轴，往复模式在齿轮档切换期间激活。

对其原理的详细说明请见“通过往复模式进行齿轮档切换 (页 1463)”。

17.2.5 定位模式

17.2.5.1 一般功能

何时为定位模式？

在定位模式下，主轴将在定义的位置处停止。此时位置闭环控制会被激活，并保持生效直至被取消。

执行下列功能时，主轴处于定位模式下：

- SPOS[<n>]=...
- SPOS[<n>]=ACP(...)
- SPOS[<n>]=ACN(...)
- SPOS[<n>]=AC(...)
- SPOS[<n>]=IC(...)
- SPOS[<n>]=DC(...)
- SPOSA[<n>]=ACP(...)
- SPOSA[<n>]=ACN(...)
- SPOSA[<n>]=AC(...)
- SPOSA[<n>]=IC(...)
- SPOSA[<n>]=DC(...) 与 SPOSA[<n>]=... 相同
- M19 或者 M[<n>]=19

其中 <n> = 主轴编号，对于主主轴可省去地址扩展 [<n>]。

SPOS[<n>]=AC(...)

将主轴定位至一个绝对位置（0 至 359.999 度）定位方向由当前的主轴旋转方向（主轴旋转）决定，或由剩余行程决定。

SPOS[<n>]=IC(...)

以编写的最后一个位置为基准将主轴定位至一个相对位置（+/- 999999.99 度）。定位方向通过待运行行程的符号确定。

SPOS[<n>]=DC(...)

将主轴沿最短行程定位至一个绝对位置（0 至 359.999 度）
定位方向由当前的主轴旋转方向决定（主轴旋转），或由控制系统自动确定（主轴停止）。

SPOS[<n>]=...

功能流程与 SPOS[<n>]=DC(...) 相同。

SPOS[<n>]=ACP(...)

从正方向逼近位置。

在从负方向定位时，主轴会制动直至转速为零，并沿相反方向加速，从而执行正逼近方向。

SPOS[<n>]=ACN(...)

从负方向逼近位置。

在从正方向定位时，主轴会制动直至转速为零，并沿相反方向加速，从而执行逼近负方向。

M19 (DIN 66025)

可通过 M19 定位主轴。此时位置和位置逼近模式通过下列设定数据读取：

SD43240 $SA_M19_SPOS[<n>]（M19 主轴定位中的主轴位置）

SD43250 $SA_M19_SPOSMODE[<n>]（M19 主轴定位中的主轴位置逼近模式）

M19 的定位作用与以下指令相同：

SPOS = <逼近模式> <位置/行程>

M19 会作为辅助功能输出至 NC/PLC 接口，和 M3、M4、M5 和 M70 一样。定位期间，M19 程序段在插补器中保持生效（同 SPOS）。

将 M19 作为宏（例如 `DEFINE M19 AS SPOS=0`）或子程序使用的零件程序将保持可执行状态。出于与之前版本控制系统的兼容性原因，可按照下面的示例取消 M19（NC 定位主轴）的内部处理：

```
MD22000 $MC_AUXFU_ASSIGN_GROUP[0] = 4        ; 辅助功能组：4
MD22010 $MC_AUXFU_ASSIGN_TYPE[0] = "M"       ; 辅助功能类型："M"
MD22020 $MC_AUXFU_ASSIGN_EXTENSION[0] = 0    ; 辅助功能扩展：0
MD22030 $MC_AUXFU_ASSIGN_VALUE[0] = 19       ; 辅助功能数值：19
```

隐性生成辅助功能 M19

为了使 M19 和 SPOS/SPOSA 在 NC/PLC 接口上的特性方面保持一致，可在使用 SPOS 和 SPOSA 时将辅助功能 M19 输出至 NC/PLC 接口。

若需激活该功能，可采用以下两种方案：

- 通过以下机床数据针对通道为通道中所有主轴激活：
  MD20850 $MC_SPOS_TO_VDI（启用 SPOS/SPOSA 时将 M19 输出至 PLC）

位	值	含义
0	0	若 MD35035 $MA_SPIND_FUNCTION_MASK 位 19 也被设置为“0”，那么在启用 SPOS 和 SPOSA 时不会生成辅助功能 M19。这样一来也可省去对辅助功能的应答时间。
	1	在零件程序中编写 SPOS 和 SPOSA 时会在零件程序中生成辅助功能 M19，并将其输出至 PLC。地址扩展对应主轴编号。

- 通过以下机床数据针对主轴和跨通道激活：
  MD35035 $MA_SPIND_FUNCTION_MASK（主轴功能）

位	值	含义
19	0	若 MD20850 $MC_SPOS_TO_VDI 位 0 也被设置为“0”，那么在启用 SPOS 和 SPOSA 时不会生成辅助功能 M19。这样一来也可省去对辅助功能的应答时间。
	1	在零件程序中编写 SPOS 和 SPOSA 时会隐性生成辅助功能 M19，并将其输出至 PLC。地址扩展对应主轴编号。

说明

在将一根主轴应用于多个通道的情形下（跨通道取轴/主轴），建议优先通过 MD35035 激活。

这两个机床数据中有一个的配置 = 1 时，将隐形生成辅助功能 M19。

此功能激活时，SPOS/SPOSA 程序段的最小执行时间会由 PLC 提升至辅助功能辅助功能的输出和应答时间。

隐形生成的辅助功能 M19 具有“Quick（快速）”和“运行中输出”属性。这些属性为固定设置，且与辅助功能专用机床数据（MD..._$M..._**AUXFU**_...）中 M19 的配置无关。

通过 FC 18 发出主轴定位指令时将**不会**生成辅助功能 M19。

定位结束

定位结束可如下编写：

FINEA[S<n>]：	在到达“精准停”时运行结束（DB31, ... DBX60.7）
COARSEA[S<n>]：	在到达“粗准停”时运行结束（DB31, ... DBX60.6）
IPOENDA[S<n>]：	在到达“插补停止”时运行结束

此外可通过 IPOBRKA 为单轴插补在制动斜坡(100-0%)中设置程序段切换的运行结束标准。

文档：
功能手册之扩展功能分册；定位轴（P2）

程序段切换

如果已经达到所有在程序段中编程的主轴或轴的运行结束标准，并且也达到了轨迹插补的程序段切换标准，那么将执行程序段切换。这既适用于零件程序程序段，也适用于工艺循环程序段。

SPOS、M19 和 SPOSA 功能相同，区别在于程序段切换特性：

- SPOS 和 M19
  程序段中编写的所有功能达到其程序段结束标准（例如，所有辅助功能由 PLC 应答，所有轴到达终点），且主轴已到达定位终点时，执行程序段切换。
- SPOSA
  程序段中编写的所有功能（主轴除外）达到其程序段结束标准后，执行程序段切换。若 SPOSA 单独编写在程序段中，则会立即执行程序段切换。主轴定位可能会跨越过个程序段（参见 WAITS）。

协调

运动过程中的协调可通过以下方式实现：

- PLC
- 机床数据配置
- 零件程序中的编程

PLC

当以下 NC/PLC 接口信号不存在时：
DB31, ... DBX83.5（主轴处于设定区域）

可设置通道专用 NC/PLC 接口信号：
DB21, ... DBX 6.1（读取禁止）
用于等待特定的主轴位置。

机床数据配置

通过以下机床数据设置：
MD35500 $MA_SPIND_ON_SPEED_AT_IPO_START = 1
并启用公差：
MD35150 $MA_SPIND_DES_VELO_TOL
实现：主轴加速到预设的转速后再进行轨迹插补。

采用以下设置时：
MD35500 $MA_SPIND_ON_SPEED_AT_IPO_START = 2
运行的轨迹轴会在加工开始前在最后一个 G0 处停止。

满足以下条件时继续执行加工：

- 收到下一个运行指令。
- 到达设定的主轴转速。
- 当 MD35510 $MA_SPIND_STOPPED_AT_IPO_START = 1
  （主轴停止时，使能轨迹进给）

零件程序中的编程

零件程序中的协调措施有以下优点：

- 零件程序的创建者可决定在哪个程序位置完成主轴加速，例如用于开始加工工件。
- 避免不必要的等待时间。

零件程序中的协调通过编写 WAITS 指令实现：

WAITS:	用于主主轴
WAITS[<n>]:	用于编号为 **<n>** 的主轴
WAITS[<n>,<m>,...]:	用于多根主轴，可支持最大主轴数的主轴

⚠ 小心

协调故障

零件程序的创建者必须为 WAITS 输入下列等待条件中的一个：

- 到达位置
- 主轴停止
- 主轴达到编写的转速

在跨通道主轴应用中，零件程序的创建者须确保：当其他通道的主轴已开始加速或制动，以实现设定的新转速或新旋转方向时，WAITS 必须在该阶段中最先启动。

执行后续程序段时将等待，直至：

- 到达使用 SPOSA 编写的位置
- 到达 M5 主轴停止状态：
  DB31, ..., DBX 61.4（主轴停止）
  此时将启用公差：
  MD36060 $MA_STANDSTILL_VELO_TOL
  识别出第一个信号的出现时，WAITS 将结束并切换至下一个程序段。
- 在 M3/M4（速度控制运行）中转速处于设定区域内：
  DB31, ..., DBX83.5（主轴位于设定区域内）
  此时将启用公差：
  MD35150 $MA_SPIND_DES_VELO_TOL
  识别出第一个信号的出现时，WAITS 将结束并切换至下一个程序段。
  此 WAITS 功能在编写的通道中生效。
  通过 WAITS 可等待该通道已知的所有主轴，包括已在其他通道中启动的主轴。

特殊情况

- **主轴速度的公差：**
  采用以下机床数据设置时：
  MD35150 $MA_SPIND_DES_VELO_TOL = 0
  NC/PLC 接口信号：
  DB31, ... DBX83.5（主轴处于设定区域）
  总是设置为 1。
  一旦主轴在转速或旋转方向变化（M3/M4）中达到设定值，WAITS 立即结束。
- **缺少使能信号：**
  若 WAITS 在转速控制运行中等待“主轴处于设定区域”，但另一方面主轴由于缺少使能信号（轴进给使能、控制器使能、脉冲使能等）停止或不旋转，那么要等到使能信号恢复，并存在“主轴处于设定区域”信号时，该程序段才会结束。
- **NC 停止和 BAG 停止时的特性：**
  若在 WAITS 期间触发了一个 NC 停止或 BAG 停止程序段，那么 NC 启动后在上述边界条件下将继续等待。

说明

对于跨通道主轴应用，在编写程序时尤其需要注意：当其他通道中的主轴尚以“旧”转速旋转时，不可在通道中启动 WAITS（过早）。
在此情形下存在“主轴设定区域”信号，因此 WAITS 会过早结束。
为了避免此情形，建议一定要在 WAITS 前设置一个 WAITM。

进给率

定位速度在以下机床数据中配置：

MD35300 $MA_SPIND_POSCTRL_VELO（位置闭环控制激活转速）

配置的定位速度可通过编程或同步动作进行修改。
`FA[S<n>]=<值>`

其中：`<n>`：　主轴号
　　`<值>`：　定位速度，单位 度/分钟

`FA[S<n>]=0` 时配置的速度生效。

加速度

加速度在以下机床数据中配置：

MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL（位置闭环控制运行下的加速度）

MD35200 $MA_GEAR_STEP_SPEEDCTRL_ACCEL（转速控制运行下的加速度）

配置的定位加速度可通过编程或同步动作进行修改。
ACC[S<n>]=<值>

其中： <n>： 主轴号
<值>： 加速度，以基于配置的加速度的百分比设定

ACC[S<n>]=0 时配置的加速度生效。

定位终止

定位通过以下方式终止：

- 通过 NC/PLC 接口信号：
  DB31, ... DBX2.2（删除剩余行程/主轴复位）
- 每次复位时（例如操作面板前端复位）
- 通过 NC 停止。

终止特性与以下机床数据无关：

MD35040 $MA_SPIND_ACTIVE_AFTER_RESET（独立主轴复位）

特殊性

主轴倍率开关有效。

17.2.5.2 从旋转状态定位主轴

初始状态

启动定位（程序中的 SPOS、M19 或 SPOSA 指令）时，主轴可能处于转速控制运行或位置闭环控制运行下。

此时须区分下列情形：

情形 1：	主轴处于转速控制运行下，超出编码器极限频率
情形 2：	主轴处于转速控制运行下，未超出编码器极限频率
情形 3：	主轴处于位置闭环控制运行下
情形 4：	主轴转速 ‹ 位置闭环控制激活转速

过程

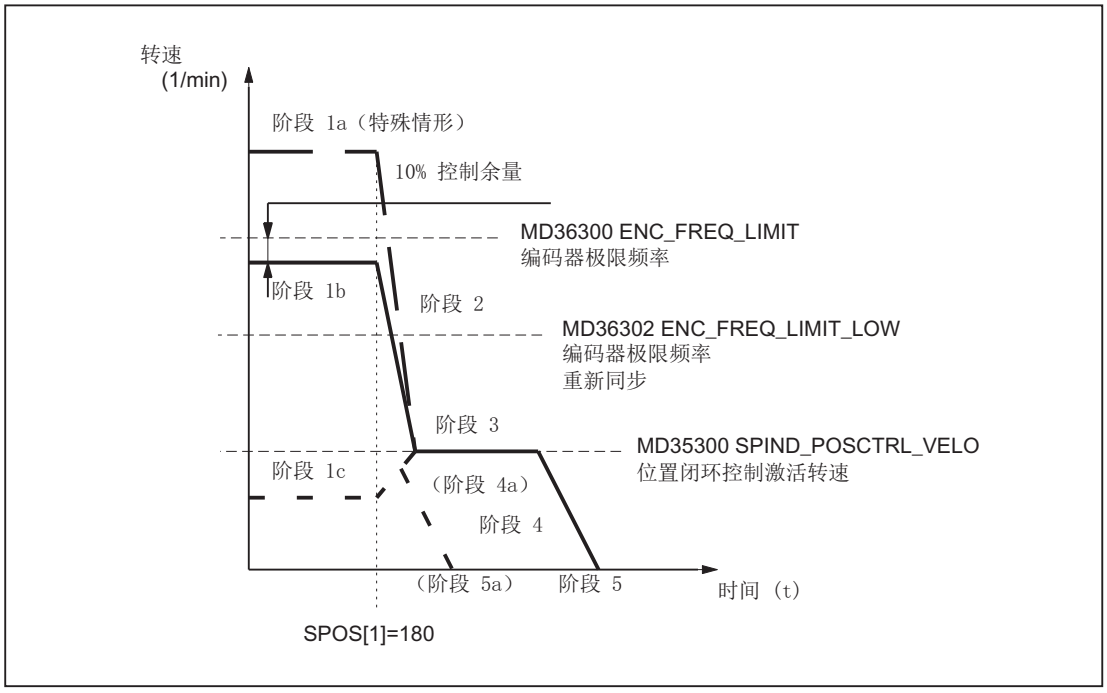


图 17-1　　从旋转状态定位主轴

说明

为编码器重新同步配置的编码器极限频率（MD36302 $MA_ENC_FREQ_LOW）所对应的转速必须大于位置闭环控制激活转速（MD35300 $MA_SPIND_POSCTRL_VELO）。

阶段 1

从阶段 1a 定位：

主轴以大于编码器极限频率的转速旋转。 主轴未同步。

从阶段 1b 定位：

主轴以小于编码器极限频率的转速旋转。主轴已同步。

说明

若激活了位置闭环控制，则转速只可为主轴最大转速或编码器极限频率的 90%（需要 10% 的控制余量）。

从阶段 1c 定位：

主轴以编写的主轴转速旋转，转速小于配置的位置闭环控制激活转速：

MD35300 $MA_SPIND_POSCTRL_VELO

主轴已同步。

阶段 2

主轴转速 > 位置闭环控制激活转速

随着 SPOS、M19 或 SPOSA 指令生效，主轴通过配置的加速度开始制动，直至达到位置闭环控制激活转速。

MD35200 $MA_GEAR_STEP_SPEEDCTL_ACCEL

低于编码器极限频率时主轴会进行同步。

主轴转速 < 位置闭环控制激活转速

SPOS、M19 或 SPOSA 指令使主轴切换至位置闭环控制运行（当其尚不处于位置闭环控制下时）。

配置的位置闭环控制运行中的加速度生效：

MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL

系统会计算与目标位置之间的行程。

主轴以时间最优的方式运行到编写的目标位置。 也就是说采用最高速度逼近目标点（最大速度 MD35300 $MA_SPIND_POSCTRL_VELO ）。 根据相应的边界条件，按 2 - 3 - 4 - 5，或者 2 - 4a - 5a 的阶段顺序运行。

阶段 3

主轴转速 > 位置闭环控制激活转速

达到配置的位置闭环控制接通转速（MD35300 $MA_SPIND_POSCTRL_VELO）时：

- 位置闭环控制激活（未处于位置闭环控制下时）
- 计算（与目标位置的）剩余行程
- 在位置闭环控制运行下切换至配置的加速度（或保持该加速度）：
  MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL

主轴转速 < 位置闭环控制激活转速

为了到达目标点，主轴会先加速至配置的位置闭环控制激活转速（MD35300 $MA_SPIND_POSCTRL_VELO）。 此时不会超出该转速。

“制动启用点计算”能够确定，通过配置的位置闭环控制中的加速度（MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL）何时能够精确到达编写的主轴位置。

阶段 4

主轴转速 > 位置闭环控制激活转速

主轴从计算得出的“制动点”开始采用机床数据：
MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL
中设定的加速度制动，直至到达目标位置。

主轴转速 < 位置闭环控制激活转速

在阶段 3 中“制动启用点计算”所确定的时间点，主轴以配置的位置闭环控制中的加速度（MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL）制动，直至到达停止状态。

阶段 4a：

在 SPOS 指令生效时，目标点过近以至于主轴无法加速到配置的位置闭环控制激活转速（MD35300 $MA_SPIND_POSCTRL_VELO）。

此时主轴会以配置的位置闭环控制中的加速度（MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL）制动，直至到达停止状态。

阶段 5

主轴转速 > 位置闭环控制激活转速

位置闭环控制保持生效，主轴停止在编写的位置。

说明

控制系统会对主轴位置实际值编码器的最大编码器极限频率进行监控（可能超出）；在位置闭环控制运行下，转速设定值会被降低到测量系统极限转速的 90%。

此时以下 NC/PLC 接口信号置位：

DB31, ... DBX83.1（编写的转速过高）

若在降低转速设定值后仍检测到“超出测量系统极限频率”，则会触发报警。

主轴转速 < 位置闭环控制激活转速（阶段 5、5a）

主轴到达位置并停止。 位置闭环控制生效，主轴停止在编写的位置上。

若主轴实际位置和编写的位置（主轴设定位置）之间的距离小于配置的精/粗准停限值，那么系统会置位以下 NC/PLC 接口信号：

DB31, ... DBX60.6（采用粗准停到达位置）

DB31, ... DBX60.7（采用精准停到达位置）

准停限值通过以下机床数据定义：

MD36010 $MA_STOP_LIMIT_FINE（精准停）

MD36000 $MA_STOP_LIMIT_COARSE（粗准停）

说明

达到定位结束标准并以信号提示后，定位结束。

前提条件是“精准停”。 通过零件程序中的 SPOS、M19 或 SPOSA 指令、同步动作、以及通过 PLC 使用 FC 18 进行的主轴定位可实现精准停。

17.2.5.3 从静止状态定位主轴

过程

需要对静止状态下的主轴执行定位时，须区分两种情形：

- 情形 1：主轴未同步。
  在控制系统和驱动接通后，或者齿轮档切换后需要定位主轴时（例如用于换刀）便是这种情形。
  MD31040 $MA_ENC_IS_DIRECT = 0
- 情形 2：主轴已同步。
  控制系统和驱动接通后，在执行首次定位前通过 M3 或 M4 进行了至少一次主轴旋转，之后通过 M5 停止主轴（与零脉冲同步）时，便是这种情形。

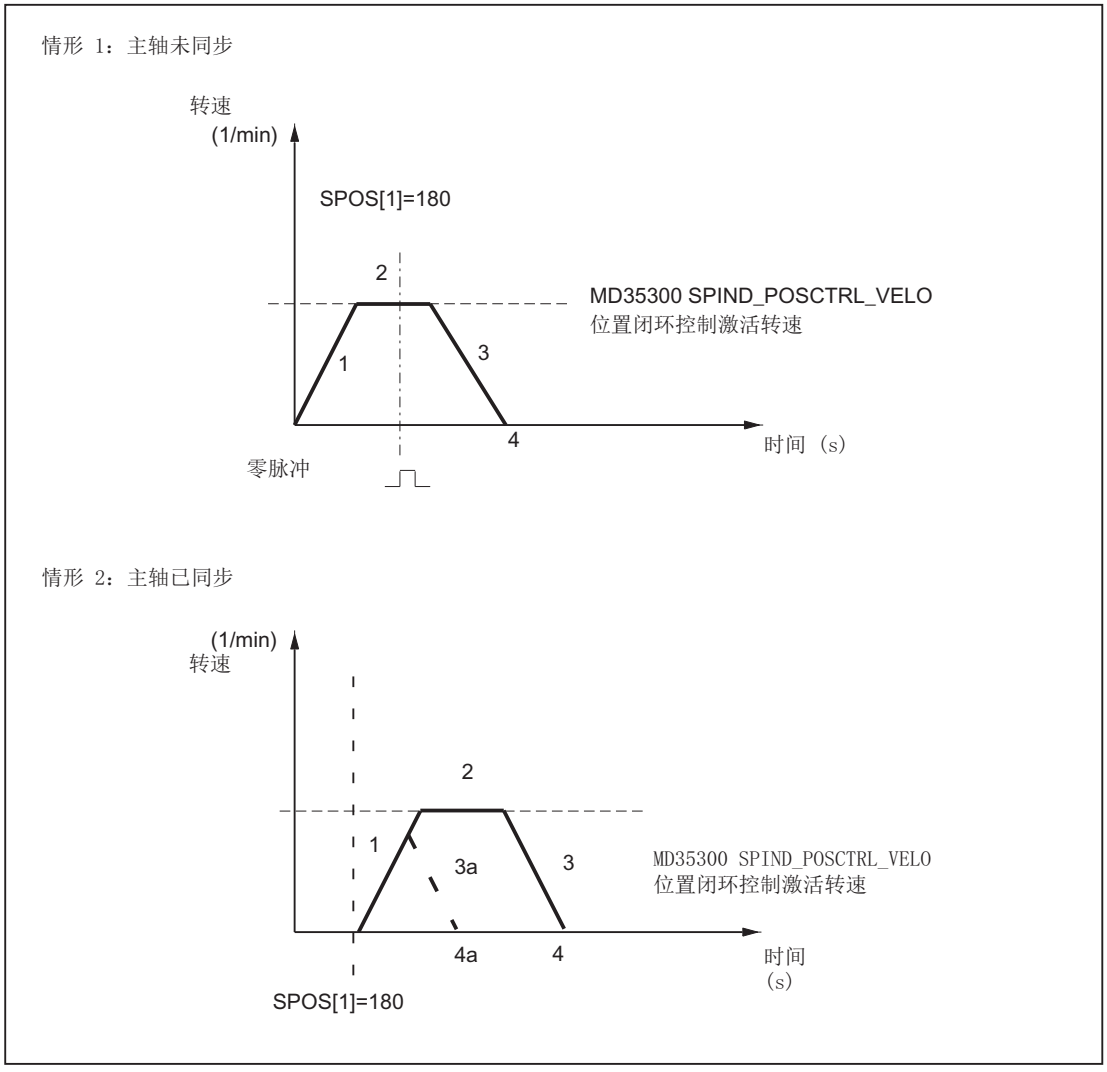


图 17-2 对停止主轴的定位

阶段 1

情形 1：主轴未同步

主轴通过编写的 SPOS、M19 或 SPOSA 指令采用以下机床数据中设定的加速度进行加速：

MD35200 $MA_GEAR_STEP_SPEEDCTRL_ACCEL（转速控制运行下的加速度）

旋转方向通过以下机床数据定义：

MD35350 $MA_SPIND_POSITIONING_DIR（从静止状态定位时的旋转方向）

特例：

若通过 ACN、ACP、IC 定位，那么编写的运行方向会被激活。

主轴与主轴位置实际值编码器的下一个零脉冲进行同步，并进入位置闭环控制运行。

此时系统会监控，在设定的行程内能否找到零脉冲（IC 除外）：

MD34060 $MA_REFP_MAX_MARKER_DIST（至参考脉冲的最大行程）

在主轴未同步的情况下达到以下机床数据中输入的转速时：
MD35300 $MA_SPIND_POSCTRL_VELO（定位转速）
主轴将以位置闭环控制激活转速继续旋转（不再进行加速）。

情形 2：主轴已同步

主轴通过 SPOS、M19 或 SPOSA 指令切换至位置闭环控制运行。

此时启用以下机床数据中设定的加速度：

MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL（位置闭环控制运行下的加速度）

旋转方向通过编写的运动（ACP、ACN、IC、DC）或剩余行程确定。

此时不会超出以下机床数据中输入的转速：
MD35300 $MA_SPIND_POSCTRL_VELO（位置闭环控制激活转速）

系统会计算与目标位置之间的行程。

主轴以时间最优的方式运行到编写的目标位置。 也就是说采用最高速度逼近目标点（最大速度 MD35300 $MA_SPIND_POSCTRL_VELO ）。 根据相应的边界条件，按 1 - 2 - 3 - 4，或者 1 - 3a - 4a 的阶段顺序运行。

阶段 2

情形 1：主轴未同步

若主轴同步，那么位置闭环控制会被激活。

主轴以以下机床数据中设定的转速一直运行：
MD35300 $MA_SPIND_POSCTRL_VELO
直到“制动启用点计算”识别出通过定义的加速度何时能够精确到达编写的主轴位置。

情形 2：主轴已同步

为了到达目标点，主轴会先加速至以下机床数据中输入的转速：
MD35300 $MA_SPIND_POSCTRL_VELO

此时不会超出该转速。

“制动启用点计算”能够确定，通过以下机床数据中定义的加速度何时能够精确到达编写的主轴位置：
MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL

在阶段 1 中“制动启用点计算”所确定的时间点，主轴启用以下机床数据中定义的加速度制动，直至到达停止状态。

MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL

阶段 3

在阶段 2 中“制动启用点计算”所确定的时间点，主轴启用以下机床数据中定义的加速度制动，直至到达停止状态。

MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL

阶段 3a：

在 SPOS 指令生效时，目标点过近以至于主轴无法加速到以下机床数据中设定的速度：
MD35300 $MA_SPIND_POSCTRL_VELO
。

主轴通过以下机床数据中定义的加速度制动，直至到达停止状态：

MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL

阶段 4、4a

主轴到达位置并停止。 位置闭环控制生效，主轴停止在编写的位置上。

若主轴实际位置和编写的位置（主轴设定位置）之间的距离小于精/粗准停限值，那么系统会置位以下 NC/PLC 接口信号：
DB31, ... DBX60.6（采用粗准停到达位置）
和

DB31, ... DBX60.7（采用精准停到达位置）

准停限值在以下机床数据中定义：

MD36010 $MA_STOP_LIMIT_FINE

MD36000 $MA_STOP_LIMIT_COARSE

阶段 3：

在阶段 2 中“制动启用点计算”所确定的时间点，主轴启用以下机床数据中定义的加速度制动，直至到达停止状态。

MD35210 $MA_GEAR_STEP_ POSCTRL_ACCEL

阶段 4：

主轴到达位置并停止。 位置闭环控制生效，主轴停止在编写的位置上。

若主轴实际位置和编写的位置（主轴设定位置）之间的距离小于精/粗准停限值，那么系统会置位以下 NC/PLC 接口信号：
DB31, ... DBX60.6（采用粗准停到达位置）
和
DB31, ... DBX60.7（采用精准停到达位置）

准停限值在以下机床数据中定义：

MD36010 $MA_STOP_LIMIT_FINE

MD36000 $MA_STOP_LIMIT_COARSE

17.2.5.4 换刀信号“主轴到达位置”

功能

换刀时的运动过程（特别是对于铣床）主要包括主轴定位，以及随后（通过优化时也可实现时间同步）通过轨迹轴进行的换刀位置逼近运行。因此执行换刀必须满足的前提条件是：在逼近换刀位置前到达主轴位置。

若换刀循环由于人为操作（例如通过 NC 停止、轴和主轴的 NC 停止、运行方式组停止等）中断，则须确保主轴不会以错误的位置进入机械手。

为此，在主轴定位中通过“精准停”到达最后编写的主轴位置时，系统会输出以下 NC/PLC 接口信号用于位置检查：

DB31, ... DBX85.5（主轴到达位置）

说明

此信号只会在“主轴定位”功能中输出。

其中包括：

- 零件程序中编写的 SPOS、SPOSA 和 M19
- 同步动作中的 SPOS 和 M19
- 使用 FC18 进行的主轴定位
- 通过 PLC 接口（DB31, ... DBX30.4）进行的主轴定位

信号置位

系统输出信号 DB31, ... DBX85.5（主轴到达位置）的前提条件是：

- 主轴的回参考点状态为：
  DB31, ... DBX60.4/5（已回参考点/已同步 1/2）= 1

  说明

  由于主轴定位中会自动搜索零脉冲，因此定位顺利完成时始终会在结束处提供回参考点信号。

- 实现“精准停”：
  DB31, ... DBX60.7（精准停）= 1
  此外设定值方面必须已到达编写的主轴位置。

删除信号

移除信号 DB31, ... DBX60.7（精准停）时，信号 DB31, ... DBX85.5（主轴到达位置）也会随之复位。

其他属性

- 若定位后主轴已位于编写的位置上，那么 NC/PLC 接口信号 DB31, ... DBX85.5（主轴到达位置）将保持置位。
- 若定位后（信号“主轴到达位置”已输出）例如在 JOG 运行方式下运行主轴，那么 NC/PLC 接口信号 DB31, ... DBX85.5（主轴到达位置）会被删除。
  之后在该运行方式下将主轴运行回到原始位置时，NC/PLC 接口信号 DB31, ... DBX85.5（主轴到达位置）又将重新置位。最后编写的位置设定保留。

17.2.6 进给轴模式

17.2.6.1 一般功能

功能

对于特定的加工任务，例如在车床上加工端面时，若转速控制下通过 M3、M4、M5 执行的主轴运行和通过 SPOS、M19 或 SPOSA 进行的主轴定位不能满足需求，则可将主轴切换至位置控制下的轴运行方式，并将其用作回转轴。

回转轴功能示例：

- 通过轴名称编程
- 零点偏移（G54，G55，TRANS，...）
- G90、G91、IC、AC、DC、ACP、ACN
- 动态转换（例如 TRANSMIT）
- 轨迹插补
- 作为定位轴运行

文献：
功能手册 扩展功能；章节“回转轴（R2）”

前提条件

- 对于主轴运行方式和进给轴运行方式，主轴电机是同一个。
- 位置测量系统对于主轴运行方式和进给轴运行方式可以是同一个，或者也可以使用独立的位置测量系统。
- 对于进给轴运行方式，务必需要一个位置实际值编码器。
- 为了采用轴运行方式，主轴必须进行回参考点，例如通过 G74 回参考点。
  示例：

程序代码	注释
M70	; 主轴切换到轴运行方式
G74 C1=0 Z100	; 轴回参考点
G0 C180 X50	; 在位置控制下运行轴

可设计的 M 功能

对于可以将主轴接通到轴运行方式的 M 功能，可通过下列机床数据针对通道进行设计：

MD20094 $MC_SPIND_RIGID_TAPPING_M_NR

说明

控制系统会根据编程顺序自动识别到轴运行方式的过渡（参见“隐性过渡至进给轴模式 (页 1428)”）。因此在零件程序中不必明确编程 M 功能将主轴切换至轴运行方式。也可编程 M 功能，以提高零件程序的可读性。

特点

- 进给倍率开关有效。
- NC/PLC 接口信号正常情况下不会结束轴运行：
  DB21, ... DBX7.7（复位）。
- NC/PLC 接口信号:
  DB31，... DBB16 至 DBB19 以及 DBB82 至 DBB91
  无意义，当：
  DB31，... DBX60.0 == 0（主轴/回转轴）时
- 可以在任何传动级激活进给轴方式。
  如果已在电机上安装位置实际值编码器（间接测量系统），可在不同传动级之间得出不同的定位和轮廓精度。
- 如果进给轴方式有效，则不能更换传动级。如需切换齿轮档，首先必须通过 M41 ... M45 或 M5、SPCOF 将主轴切换到受控方式中。
- 在进给轴方式中，第一个参数组生效（机床数据系下标为零）。
  文档：
  功能手册之基本功能分册；“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”

动力

在轴运行方式中，轴的动态限值生效。例如：

- MD32000 $MA_MAX_AX_VELO[<轴>]（最大轴速度）
- MD32300 $MA_MAX_AX_ACCEL[<轴>]（最大轴加速度）
- MD32431 $MA_MAX_AX_JERK[<轴>]（轨迹运动时的最大轴向急动）

前馈控制

保持对轴生效的前馈模式。

“动态前馈控制”功能的详细说明请见：

文献

功能手册 扩展功能；章节“补偿（K3）> “动态前馈控制（滞后误差补偿）”

示例：采用模拟控制器时的分辨率切换

切换至轴运行方式

编程	注释
SPOS=...	
M5	; 控制器使能关闭（由 PLC） → 输出至 PLC
M70	; 控制器切换（由 PLC，基于 M70） 控制器使能打开（由 PLC）
C=...	; NC 以进给轴参数组运行

切换至主轴运行方式

编程	注释
C=...	
M71	; → 输出至 PLC 控制器使能关闭（由 PLC） 控制器切换（由 PLC） 在 NC 内部切换到主轴参数组（1-5）上，控制器使能开（由 PLC）
M3/4/5 或 SPOS=...	; NC 以主轴参数组运行

在主轴运行方式下切换

根据生效的传动级选择参数组 1...5。

原则上除了带补偿衬套的攻丝时外都需要接通前馈控制：

MD32620 $MA_FFW_MODE（前馈控制方式） ≠ 0

参数组	轴运行	主轴运行
1	生效	-
2	-	生效
3	-	生效
4	-	生效

参数组	轴运行	主轴运行
5	-	生效
6	-	生效
主轴方式：不同齿轮档下的参数组		

17.2.6.2 隐性过渡至进给轴模式

功能

控制系统能够基于编程顺序自动识别出向进给轴模式的过渡，并于系统内部生成必要的 M70 过程。执行的步骤取决于具体情形，其中主要包括：

1. 停止主轴
2. 激活位置闭环控制，处理前馈控制和参数组切换
3. 程序段预处理的位置同步（必要时内部预处理停止）

该功能始终生效。因此基本不需要在零件程序中编写 M70 指令。

过程

隐性过渡至进给轴模式的步骤（零件程序中未编写 M70）：

- 从转速控制运行（M3、M4、M5、SPCOF、...）过渡至进给轴模式：
  控制系统内部识别出过渡，并在请求进给轴模式的程序段前插入一个中间程序段。生成的程序段包含 M70 功能。该程序段的执行时间与程序中编写的 M70 程序段的执行时间基本相同。放弃隐性生成辅助功能 M70 以及将其输出至 PLC 时（参见 MD35035），停止主轴上的短暂切换可能会有区别（无制动时间）。
- 从定位模式（M19、SPOS、SPOSA）过渡至进给轴模式：
  过渡将立即执行，不会生成中间程序段。在进行了相应配置时（参见 MD35035），隐性生成的辅助功能 M70 会在切换至主轴处于进给轴模式的程序段时输出至 PLC。

辅助功能向 PLC 的输出

向进给轴模式的隐性过渡可以辅助功能输出的形式发送至 PLC。

激活/取消激活

此功能通过以下机床数据激活/取消：

MD35035 $MA_SPIND_FUNCTION_MASK（主轴功能）

位	值	含义
20	0	控制系统内部生成 M70 时，不向 PLC 输出辅助功能
	1	控制系统内部生成 M70 时，隐性生成辅助功能 M70 并输出至 PLC。地址扩展对应主轴编号。

说明

零件程序中编写的辅助功能 M70 始终会输出至 PLC。

属性

隐性生成的辅助功能 M70 具有“Quick（快速）”和“运行中输出”属性。这些属性为固定设置，且与辅助功能专用机床数据（MD..._$M..._**AUXFU**_...）中 M70 的配置无关。

过渡至进给轴模式时只会生成一次 M70。在之后的将主轴作为轴运行的程序段中，系统将不再生成和输出 M70 辅助功能。直到通过 POS、M3、M4、M5、SPCOF 等指定退出进给轴模式，并于之后重新过渡至进给轴模式时，系统才会重新生成隐性 M70 并输出。

前提条件

同步动作

在同步动作中将主轴编写为一般轴时，必须确保满足过渡至进给轴模式的前提条件。

当主轴处于转速控制运行中时，在将其定义为轴前必须编写 M70 或 SPOS 指令。否则轴编程可能会触发报警。

FC 18

和同步动作一样，在使用 FC 18 时也须确保过渡至进给轴模式的应用条件，例如通过过渡预备定位指令。否则 PLC 会以 FC 18 状态字中的故障位对 FC 18 调用进行应答。

使用 FC 18 编程过渡至进给轴模式时，**不会**隐性生成辅助功能 M70。

示例

示例 1：

零件程序：将旋转主轴过渡至进给轴模式

配置：MD35035 $MA_SPIND_FUNCTION_MASK，位 20 = 1

程序代码	注释
N05 M3 S1000	
N10 ...	
N15 POS[C]=77	; 切换至 N15 前生成一个 M70 中间程序段，通过该程序段停止主轴，并将 M70 输出至 PLC。
…	

示例 2：

零件程序：从定位模式过渡至进给轴模式

配置：MD35035 $MA_SPIND_FUNCTION_MASK，位 20 = 1

程序代码	注释
N05 SPOS=0	
N10 ...	
N15 C77	; 将隐性 M70 输出至 PLC，无中间程序段
…	

示例 3：

同步动作：从主轴定位模式过渡至进给轴模式

配置：MD35035 $MA_SPIND_FUNCTION_MASK，位 20 = 1

程序代码	注释
WHEN COND1==TRUE DO SPOS=180	
WHEN COND2==TRUE DO POS[C]=270	; 将隐性 M70 输出至 PLC

示例 4：

同步动作：通过 M70 从转速控制运行过渡至进给轴模式

配置：MD35035 $MA_SPIND_FUNCTION_MASK，位 20 = 1

程序代码	注释
WHEN COND11==TRUE DO M3 S1000	
WHEN COND12==TRUE DO M70	; 将 M70 输出至 PLC
WHEN COND13==TRUE DO POS[C]=270	; 不生成隐性 M70，因为已启用进给轴模式

示例 5：

同步动作：不允许的从转速控制运行到进给轴模式的过渡

配置：MD35035 $MA_SPIND_FUNCTION_MASK，位 20 = 1

程序代码	注释
WHEN COND21==TRUE DO M3 S1000	
WHEN COND22==TRUE DO POS[C]=270	; 报警 20141!

17.2.7 主轴初始设置

主轴初始设置

可以使用下列机床数据将主轴运行方式确定为初始设置：

MD35020 $MA_SPIND_DEFAULT_MODE

值	主轴初始设置
0	转速控制运行，取消位置调节
1	转速控制运行，打开位置调节
2	定位运行
3	轴运行

主轴初始设置的生效时间

在下列机床数据中可以设置主轴初始设置的生效时间：

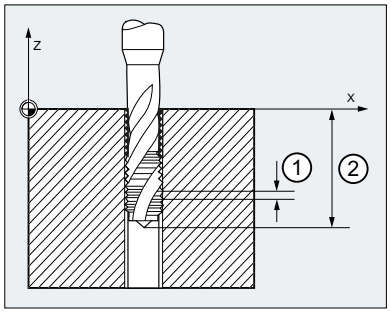
MD35030 $MA_SPIND_DEFAULT_ACT_MASK

值	生效时间
0	上电
1	上电并启动程序
2	上电并复位（M2 / M30）

17.2.8 刚性攻丝

17.2.8.1 功能

在刚性攻丝中，系统在位置闭环控制下对线性轴和主轴的运行进行插补。前提条件是**主轴**带**行程测量系统**并受**位置闭环控制**。



① 螺距

② 螺纹深度

螺纹旋转方向

螺纹旋转方向（右旋螺纹或者左旋螺纹）通过螺距的符号确定：

- 正螺距 → 右旋螺纹（旋转方向对应 M3）
- 负螺距 → 左旋螺纹（旋转方向对应 M4）

定义的主轴起始位置

在刚性攻丝中，在下列情形下必须编写定义的主轴起始位置（SPOS）：

- 在多重加工中制造的螺纹
- 需要既定的螺纹起始位置的螺纹

```
SPOS = <起始位置>
G331 ...
```

17.2.8.2 编程

使用指令 G331 和 G332 编程不带补偿夹具的攻丝时，会执行以下运行：

- G331:在钻削方向上攻丝，直至螺纹终点
- G332:通过主轴旋转方向自动反向，回退至成套丝锥 G331

句法

```
G331 <轴> <螺距>
G331 <轴> <螺距> S...
G332 <轴> <螺距>
```

含义

G331:	攻丝 通过轴运行（钻深）和螺距描述攻丝。	
	生效方式：	模态
G332:	攻丝的回退 回退应以与攻丝时相同的螺距执行 (G331)。主轴自动换向。	
	生效方式：	模态
<轴>：	几何轴（X、Y 或 Z）至螺纹终点的运行行程/位置，例如 Z50	
<螺距>:	螺距 I (X)、J (Y) 或 K (Z)： ● 正螺距：右螺纹，例如 K1,25 ● 负螺距：左螺纹，例如 K-1,25	
	值域：	±0.001 至 ±2000.00 mm/rev
S...:	主轴转速 如未给定主轴转速，则以上次生效的主轴转速运行。	

说明

第二齿轮级数据组

为了在攻丝时有效地调节主轴转速和电机转矩并提高加速度，可以在轴专用的机床数据中配置第二齿轮级数据组，以及另外两个可定义的开关阈值（最大转速和最小转速），该数据组不同于第一齿轮级数据组，且与其转速开关阈值无关。请务必注意机床制造商的规定。

文档：
功能手册之基本功能；主轴（S1），章节：“可定义的齿轮级匹配”

示例

- 示例：使用 G331 / G332 攻丝 (页 1434)
- 示例：给出当前齿轮级已编程的钻削转速 (页 1434)
- 示例：使用第二齿轮级数据组 (页 1435)
- 示例：未进行转速编程，齿轮级监控 (页 1435)

- 示例：无法进行齿轮换档，齿轮级监控 (页 1435)
- 示例：不带 SPOS 的编程 (页 1436)

17.2.8.3 示例：使用 G331 / G332 攻丝

程序代码	注释
N10 SPOS[n]=0	; 主轴：位置闭环控制运行
	; 起始位置 0 度
N20 G0 X0 Y0 Z2	; 轴：运行至起始位置
N30 G331 Z-50 K-4 S200	; 在 Z 轴上攻丝，
	; 螺距 K-4 负 =>
	; 主轴旋转方向：逆时针，
	; 主轴转速 200 转/分钟
N40 G332 Z3 K-4	; 沿 Z 轴回退，
	; 螺距 K-4 负（逆时针），
	; 自动反向旋转 =>
	; 主轴旋转方向顺时针
N50 G1 F1000 X100 Y100 Z100 S300 M3	; 主轴处于主轴运行中。

17.2.8.4 示例：给出当前齿轮级已编程的钻削转速

程序代码	注释
N05 M40 S500	; 编程设置的主轴转速：500 转/分钟 =>
	; 齿轮级 1 (20 至 1028 转/分钟)
...	
N55 SPOS=0	; 定位主轴
N60 G331 Z-10 K5 S800	; 攻丝
	; 主轴转速 800 转/分钟 => 齿轮级 1

使用 M40 时通过第一齿轮级数据组测定与编程的主轴转速 S500 匹配的齿轮级。在当前齿轮级中输出编程的钻削转速 S800，并且在必要时将其限制为齿轮级的最大转速。进行 SPOS 后不能进行自动齿轮换档。自动齿轮换档的前提条件是主轴的转速控制运行。

说明

如果主轴转速为 800 rpm，选择了齿轮级 2，此时必须在相应的第二齿轮级数据组的机床数据中设计最大转速和最小转速的开关阈值（参见以下例子）。

17.2.8.5 示例：使用第二齿轮级数据组

第二齿轮级数据组中最大转速和最小转速的开关阈值将在编程 G331/G332 时，以及编程有效主主轴 S 值时计算。自动齿轮换档 M40 必须有效。由此测定的齿轮级会和生效的齿轮级相比较。如果两齿轮级有差别，则进行齿轮换档。

程序代码	注释
N05 M40 S500	；编程设置的主轴转速：500 转/分钟
...	
N50 G331 S800	；主主轴：选择齿轮级 2
N55 SPOS=0	；定位主轴
N60 G331 Z-10 K5	；攻丝
	；从第 2 齿轮级数据组加速主轴

17.2.8.6 示例：未进行转速编程，齿轮级监控

如果使用第二齿轮级数据组时，G331 指令中没有编程转速，则会以上次编程的转速加工螺纹。齿轮换档不生效。在此情况下会监控，上次编程的转速是否在当前齿轮级规定的转速范围（最大转速和最小转速开关阈值）内。若不在范围内，则显示报警 16748。

程序代码	注释
N05 M40 S800	；编程设置的主轴转速：800 转/分钟
...	
N55 SPOS=0	；定位主轴
N60 G331 Z-10 K5	；攻丝
	；监控主轴转速 800 转/分钟
	；齿轮级 1 生效
	；齿轮级 2 必须生效 => 报警 16748

17.2.8.7 示例：无法进行齿轮换档，齿轮级监控

如果使用第二齿轮级数据组时，在 G331 程序段中除了几何数据之外，还编程了主轴转速，而该转速不在当前齿轮级的规定转速范围（最大转速和最小转速的开关阈值）内，那么将不会执行齿轮换档，因为此时无法保持主轴和进给轴的路径运行。

同上述例子，在 G331 程序段中也会对转速和齿轮级进行监控，并相应地显示报警 16748。

程序代码	注释
N05 M40 S500	；编程设置的主轴转速：500 转/分钟 =>
	；齿轮级 1
...	
N55 SPOS=0	；定位主轴

程序代码	注释
N60 G331 Z-10 K5 S800	；攻丝 ；无法进行齿轮换档， ；监控主轴转速 800 转/分钟 ；带第 1 组齿轮级数据：齿轮级 2 ；必须生效 => 报警 16748

17.2.8.8 示例：不带 SPOS 的编程

程序代码	注释
N05 M40 S500	；编程设置的主轴转速：500 转/分钟 => ；齿轮级 1 (20 至 1028 转/分钟)
...	
N50 G331 S800	；主主轴：选择齿轮级 2
N60 G331 Z-10 K5	；攻丝 ；从第 2 齿轮级数据组加速主轴

主轴从当前位置开始执行螺纹插补，该位置由之前执行的零件程序部分决定，比如在执行齿轮换档时。因此可能无法再对螺纹进行再加工。

说明

必须注意的是，在使用多主轴加工时钻削主轴必须为主主轴。通过编程 SETMS (<主轴编号>) 可将钻削主轴设置为主主轴。

17.2.8.9 特殊情形：在 NC 程序中通过 NC/PLC 接口信号将旋转方向反转

在正常情况下，必须在 NC 程序启动**前**通过 PLC 用户程序借助 NC/PLC 接口信号“M3 / M4 反向”（DB31, ... DBX17.6）请求主轴的旋转方向反转。NC 仅会在启动时自动分析该接口信号的状态，并视情况改变主轴的旋转方向。

若需要在程序执行期间通过 NC/PLC 接口信号“M3/M4 反向”（DB31, ... DBX17.6）再次修改旋转方向，则可在 NC 程序中通过以下过程实现：

N110 SPCOF 或 M5	停止主轴并进入转速闭环控制
N111 Mxx 或 Hxx	输出一个用户专用辅助功能，用于在 PLC 用户程序中触发以下动作： • Setzen der Einlesesperre im Kanal:DB21, ... DBX6.1 = 1 • Invertieren des Nahtstellensignals "M3/M4 invertieren" der Spindel:DB31, ... DBX17.6 • Rücksetzen der Einlesesperre im Kanal:DB21, ... DBX6.1 = 0

N112 STOPRE	预处理停止
N113 SPOS=IC(0.001)	主轴在位置闭环控制下增量运行。 基于主轴在位置闭环控制下的增量运行，NC 重新分析接口信号 DB31, ... DBX17.6（M3/M4 反向）。 自此程序段起，在主轴运行中，基于该接口信号的旋转方向反转生效。
N114 SPOS=0	**可选**：若工艺上需要既定的初始位置（例如 0°），对主轴进行定位。

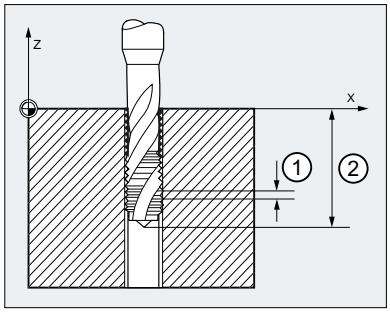
示例

程序代码	注释
N010 G0 X0 Z0	; 轴的初始设置
N020 G1 G94 F1000 Z10	; Z 轴运行至初始位置
N030 M3 S100	; 主轴在转速闭环控制下运行
	; 转速为 100 rpm
	; 采用程序启动时确定的主轴旋转方向
...	; 任意加工
N040 M5	; 停止主轴并进入转速闭环控制
N050 H10	; Hilfsfunktionsausgabe:H0=10
	; 在 PLC 用户程序中：
	; - Einlesesperre setzen:DB21, ... DBX6.1=1
	, - Drehrichtung invertieren:DB31, ... DBX17.6
	; - Einlesesperre rücksetzen:DB21, ... DBX6.1=0
N060 STOPRE	; 预处理停止
N070 SPOS=IC(0.001)	; 主轴在位置闭环控制下增量运行
N080 SPOS=0	; 对主轴进行定位
N090 G331 Z-10 K1 S500	; 刚性攻丝
N100 G332 Z2 K1	; Gewindebohren ohne Ausgleichsfutter:后退运行
N110 G1 G94 F1000 Z10	; 返回 Z 轴的初始位置
N120 SPCOF	; 将主轴切换至转速闭环控制
N130 H10	; 见上
N140 STOPRE	; 预处理停止
N150 SPOS=IC(.001)	; 主轴在位置闭环控制下增量运行
N160 SPOS=0	; 对主轴进行定位
N170 G331 Z-10 K1 S500	; 刚性攻丝
N180 G332 Z2 K1	; Gewindebohren ohne Ausgleichsfutter:后退运行
N190 G1 G94 F1000 Z10	; Z 轴运行至初始位置
...	

17.2.9 带弹性卡头的攻丝

17.2.9.1 功能

在带弹性卡头的攻丝（G63）中，不随线性轴对主轴进行插补，而是在转速闭环控制下运行。必须计算并明确编写取决于主轴转速及螺距的线性轴进给率。由于不一同对主轴和线性轴进行插补，例如为对主轴的转速波动进行补偿，需要位于弹性卡头中的丝锥。



回退

回退同样通过 G63 编写，但采用相反的主轴旋转方向。

轴及主轴倍率值

在选择了“**带**弹性卡头的攻丝”功能的情况下，生效的轴及主轴倍率值为 100 %。

进给率

待编写的线性轴进给率 F 为主轴转速与丝锥螺距的乘积：

F [mm/min] = 主轴转速 S [rpm] * 螺距 [mm/rev]

17.2.9.2 编程

使用指令 G63 编程带补偿夹具的攻丝时，会执行以下运行：

- G63：在钻削方向上攻丝，直至螺纹终点
- G63：设置了主轴旋转方向反向的回退

说明

在 G63 程序段之后，上次生效的插补方式 G0、G1、G2 会再次生效。

句法

```
G63 <轴> <旋转方向> <转速> <进给率>
```

含义

G63:	带补偿夹具的攻丝	
	生效方式：	非模态
<轴>：	几何轴（X、Y 或 Z）至螺纹终点的运行行程/位置，例如 Z50	
<旋转方向>：	主轴旋转方向： • M3：顺时针旋转，右螺纹 • M4：逆时针方向，左螺纹	
<转速>：	攻丝时允许的最大主轴转速，例如 S100	
<进给率>：	钻削轴的进给率 F，其中 F = 主轴转速 * 螺距	

示例

钻削 M5 螺纹：

- 螺距符合标准：0.8 毫米/转
- 主轴转速 S：200 转/分钟
- 进给率 F = 200 转/分钟 * 0.8 毫米/转 = 160 毫米/分钟。

程序代码	注释
N10 G1 X0 Y0 Z2 F1000 S200 M3	; 逼近起始点
	; 主轴以顺时针方向旋转，200 转/分钟
N20 G63 Z-50 F160	; 带补偿夹具的攻丝
	; 钻削深度：绝对 Z=50mm
	; 进给率：160 mm/min
N30 G63 Z3 M4	; 回退：绝对 Z=3mm
	; 旋转方向反向
	; 主轴以逆时针方向旋转，200 转/分钟

17.3 回参考点/同步

为何进行同步?

为了确保控制系统在上电后能够精确识别主轴位置，必须将控制系统与主轴的位置测量系统同步。

仅在经过同步后，主轴才能支持下列功能：

- 螺纹切削
- 刚性攻丝
- 轴编程

对主轴同步的更多说明请见“R1：回参考点 (页 1349)”章节。

为何要回参考点？

为了确保控制系统在上电后能够精确识别机床零点，必须将控制系统与回转轴的位置测量系统同步。该操作称为回参考点。回参考点的过程被称为回参考点运行。

仅在执行回参考点后轴才能精确逼近编程的机床上的位置。

对回转轴回参考点的更多说明请见“R1：回参考点 (页 1349)”章节。

位置测量系统的安装位置

位置测量系统可如下安装：

- 直接安装在电机上，且主轴上设有充当零脉冲编码器的无接触式接近开关
- 通过测量齿轮箱安装在电机上，且主轴上设有充当零脉冲编码器的无接触式接近开关
- 直接安装在主轴上
- 通过测量齿轮箱安装在主轴上，且主轴上设有充当零脉冲编码器的无接触式接近开关（仅在传动比不等于 1:1 时）

若存在两个位置测量系统，可将其安装在相同的位置，或者分别安装在不同位置。

同步过程

控制系统上电后，主轴按如下过程进行同步：

- 主轴以设定的主轴转速（S 值）和主轴旋转方向（(M3 或 M4）启动，并与下一个位置测量系统零脉冲或下一个无接触式接近开关的信号同步。
- 通过 SPOS、M19 或 SPOSA 对静止状态的主轴进行定位。主轴与下一个位置测量系统零脉冲或下一个无接触式接近开关的信号同步。之后定位至编写的位置。

- 可通过 SPOS、M19 或 SPOSA 对运动中（M3 或 M4）的主轴进行同步。
  此时具有以下特性：
  - 使用 SPOS=<位置>、SPOS=DC(<位置>) 和 SPOS=AC(<位置>) 时，保持运动方向逼近位置。
  - 使用 SPOS=ACN(<位置>) 或 SPOS=ACP(<位置>) 时将始终采用正方向或负方向逼近。必要时需在定位钱反转运动方向。
- 在 JOG 运行方式下通过方向键在转速控制运行中越过零脉冲。

说明

通过零件程序、FC 18 或同步动作触发的同步具有相同的效果。

说明

在主轴同步中，所有四个参考点值根据所选测量系统生效。测量系统偏移也以同样的方式生效。

请注意下列机床数据：

- MD34080 $MA_REFP_MOVE_DIST
  （使用距离编码系统时的参考点距离/目标点）
- MD34090 $MA_REFP_MOVE_DIST_CORR
  （距离编码的参考点偏移/绝对偏移）
- MD34100 $MA_REFP_SET_POS
  （参考点值，对距离编码系统无意义）

若通过 SPOS=IC(...) 和 < 360 度的行程定位未回参考点的主轴，可能无法越过零脉冲，主轴位置仍未和零脉冲同步。这可能出现于以下情形：

- 上电后
- 通过设置轴专用 NC/PLC 接口信号：
  DB31, ... DBX17.5（定位 2 中重新同步主轴）
  DB31, ... DBX17.4（定位 1 中重新同步主轴）

通过无接触式接近开关同步时的特殊性

无接触式接近开关的信号延时所引起的位置错误可于 NC 内部通过输入信号运行时间补偿进行修正。

信号运行时间补偿通过以下机床数据设置：

- MD31122 $MA_BERO_DELAY_TIME_PLUS
  （无接触式接近开关的延时，正运动方向）
- MD31123 $MA_BERO_DELAY_TIME_MINUS
  （无接触式接近开关的延时，负运动方向）

作用取决于设置的回参考点模式：

MD34200 $MA_ENC_REFP_MODE = <值>

<值>	含义
2	在不设定特定定位速度/转速的情况下执行位置同步。
7	将以下机床数据中的速度用作位置同步过程中的定位速度。 MD34040 $MA_REFP_VELO_SEARCH_MARKER（关闭速度） 零脉冲搜索不会自动执行，而是需要通过以下 NC/PLC 信号的 0/1 脉冲沿显性请求： DB31, ... DBX16.4 / 5（重新同步主轴，测量系统 1 / 2）

说明

供货状态下已对信号运行时间补偿进行了相应预设，通常不需要修改。

回参考点过程

控制系统启动后，若需直接在进给轴模式下编写主轴，必须确保该轴已回参考点。

控制系统上电后，可对主轴执行回参考点（前提条件时每转一个零脉冲）。

回参考点的步骤参见“R1：回参考点 (页 1349)”章节。

同步主轴（参见“同步过程”部分）时，若主轴的位置测量系统也用于回转轴，那么回转轴也会同时进行回参考点。

主轴位置测量系统

为主轴模式和进给轴模式使用的是一个电机时，主轴可从主轴模式切换至进给轴模式（回转轴）。

主轴（主轴模式和进给轴模式）可配备一到两个位置测量系统。启用两个位置测量系统时，可将其中的一个指定给主轴，将另一个指定给回转轴，或者将两个位置测量系统都指定给主轴。若存在两个位置测量系统，则其均由控制系统更新，不过只能有一个位置测量系统生效。

通过以下 NC/PLC 接口信号选择生效的位置测量系统：

DB31, ... DBX1.5 / 6（位置测量系统 1 / 2）

生效的位置测量系统应用于以下功能：

- 主轴位置闭环控制（SPCON）
- 主轴定位（SPOS、M19 和 SPOSA）

- 螺纹切削（G33、G34、G35）
- 刚性攻丝（G331、G332）
- 旋转进给率（G95）
- 恒定切削速度（G96、G961、G97、G971）
- 主轴转速实际值显示
- 进给轴模式
- 同步主轴设定值耦合

重新同步主轴的位置测量系统

在以下情形下，必须重新对主轴的位置测量系统执行同步：

- 位置测量编码器安装在电机上，无接触式接近开关安装在主轴上，并进行了齿轮档切换。主轴以新的齿轮档旋转时，系统内部触发同步（参见同步过程部分）。
- 机床可在垂直主轴和水平主轴之间进行切换。此时使用两个不同的位置测量编码器（一个用于水平主轴，一个用于垂直主轴），但只使用一个控制系统上的实际值输入。若在水平主轴和垂直主轴之间进行了切换，必须重新执行同步。
  同步通过以下 NC/PLC 接口信号触发：
  DB31, ... DBX16.4（重新同步主轴 1）
  或
  DB31, ... DBX16.5（重新同步主轴 2）
  为此主轴必须处于控制模式下。

在热启动后恢复位置

对于使用增量位置测量系统的主轴，可在断电后将实际值继续保存在缓存中，并在上电后恢复为断电前最后保存的位置，这样一来例如可对转换等位置相关功能执行恢复（参见“自动恢复机床参考 (页 1392)”章节）。其应用例如有：使用刀具定向加工时在热启动后退刀（参见“上电后通过定向转换回退刀具 (页 710)”章节）。

位置恢复后，位置测量系统的状态通过以下 NC/PLC 接口信号显示：

DB31, ... DBX71.4 （“已恢复 1”），对应位置测量系统 1

DB31, ... DBX71.5 （“已恢复 2”），对应位置测量系统 2

在 JOG 运行方式下退刀后，必须对恢复位置的轴执行回参考点。这样一来信号 DB31, ... DBX71.4/5（“已恢复 1/2”）会被删除，信号 DB31, ... DBX60.4/5（“已回参考点/已同步 1/2”）会被置位。

说明

若机床数据 MD20700 $MC_REFP_NC_START_LOCK 设置为“2”，则可使用“恢复”的轴位置进行 NC 启动（MDI 运行方式下或溢出转存时）。

17.4 可配置的齿轮档调整

17.4.1 主轴的齿轮档和齿轮档切换

为什么要启用齿轮档？

主轴的齿轮档用于实现电机转速降档，从而在较低的主轴转速下生成较高的转矩，或用于升档，从而获取较高的转速。

齿轮档数量

可为每根主轴配置 5 个齿轮档。

使用的齿轮档的数量通过以下机床数据定义：

MD35090 $MA_NUM_GEAR_STEPS

齿轮档的参数设置

齿轮档 1 至 5 可通过下列机床数据进行设置：

机床数据	含义
MD35012 $MA_GEAR_STEP_CHANGE_POSITION[<n>]	齿轮档切换位置
MD35110 $MA_GEAR_STEP_MAX_VELO[<n>]	自动齿轮档切换的最大转速
MD35120 $MA_GEAR_STEP_MIN_VELO[<n>]	自动齿轮档切换的最小转速
MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[<n>]	齿轮档的最大转速

机床数据	含义
MD35135 $MA_GEAR_STEP_PC_MAX_VELO_LIMIT[<n>]	位置闭环控制下齿轮档的最大转速
MD35140 $MA_GEAR_STEP_MIN_VELO_LIMIT[<n>]	齿轮档的最小转速
MD35200 $MA_GEAR_STEP_SPEEDCTRL_ACCEL[<n>]	转速控制运行下的加速度
MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL[<n>]	位置闭环控制运行下的加速度
MD35300 $MA_SPIND_POSCTRL_VELO[<n>]	位置闭环控制激活转速
MD35310 $MA_SPIND_POSIT_DELAY_TIME[<n>]	定位延时
MD35550 $MA_DRILL_VELO_LIMIT[<n>]	刚性攻丝的最大转速

齿轮档切换类型

齿轮档切换的类型通过以下机床数据设置：

MD35010 $MA_GEAR_STEP_CHANGE_ENABLE

位	值	含义
0	0	主轴电机直接（1:1）或以不可变的传动比安装至主轴（缺省设置）。 第一齿轮档的机床数据生效。
	1	主轴电机具有最多 5 个齿轮档。 齿轮档切换如下进行： ● 往复模式下 ● 在不确定的切换位置上
1	0	含义同“位 0 = 0”。
	1	含义同“位 0 = 1”，但是齿轮档切换在配置的主轴位置上进行。 切换位置在以下机床数据中设置： MD35012 $MA_GEAR_STEP_CHANGE_POSITION 齿轮档切换前会以当前齿轮档逼近该位置。 位 1 置位时，则可忽略位 0！
3	1	对 NC 和 PLC 间的齿轮档切换信号对话进行仿真。
5	1	在 G331/G332 攻丝中使用第二齿轮档数据组（参见下文中的“第二齿轮档数据组”部分）。必须为攻丝时使用的主主轴设置该位。

齿轮档切换的前提条件

原则上，仅在请求的齿轮档不同于生效齿轮档时，才会执行齿轮档切换。

齿轮档切换时的参数组选择

启用以下设置时，伺服参数组也会随齿轮档切换：

MD35590 $MA_PARAMSET_CHANGE_ENABLE = 0 或 1

更多信息参见章节“齿轮档切换时的参数组选择 (页 1458)”。

请求齿轮档切换

可通过以下方法请求齿轮档切换：

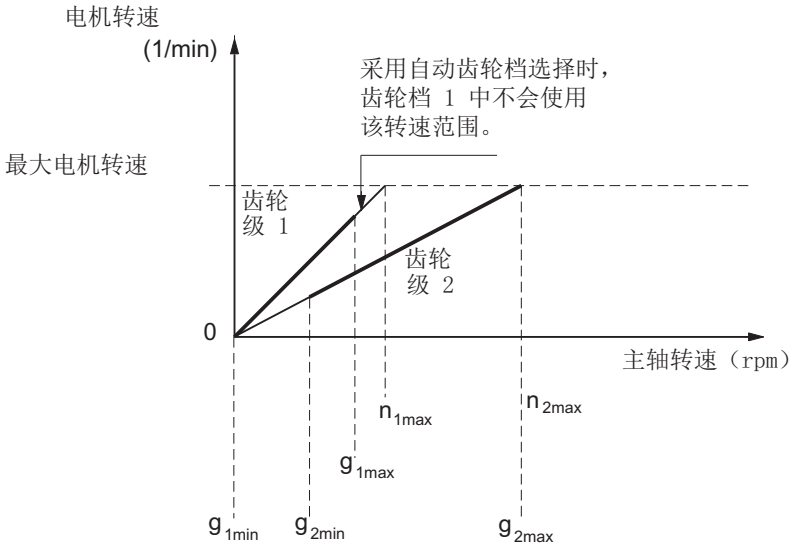
- 在零件程序中通过：
  - `M40 S...`
    为编写的转速 `S...` 自动选择齿轮档
  - `M41 ... M45`
    直接选择齿轮档 1 ... 5
  - `M70`
    基于 MD35014 $MA_GEAR_STEP_USED_IN_AXISMODE = 1 ... 5
    （参见“执行 M70 时启用的齿轮档 (页 1477)”）
  - `G331 S...`
    基于 MD35010 $MA_GEAR_STEP_CHANGE_ENABLE，位 5 = 1
- 在同步动作中通过：
  - `DO M40 S...`
    为编写的转速 `S...` 自动选择齿轮档
  - `DO M41...M45`
    直接选择齿轮档 1 ... 5
  - `DO M70`
    基于 MD35014 $MA_GEAR_STEP_USED_IN_AXISMODE = 1 ... 5
- 通过 PLC 使用功能块 FC18
- 在复位状态下通过描述 NC/PLC 接口：
  DB31, ... DBX16.0-16.2（实际齿轮档 A 至 C）
  特别是在上电后，可借此将机械生效的齿轮档信息提供给 NC。

说明

主轴电机直接（1:1）或以不可变的传动比安装至主轴时（MD35010 = 0），辅助功能 M40 和 M41 ... M45 对该主轴无意义。

齿轮档选择

下面的示例中显示的主轴转速设定为最大值时两个齿轮档之间的选择：



可通过 MD35110 GEAR_STEP_MAX_VELO 设定：

n_{1max} ... 齿轮档 1 的最大主轴转速

g_{1min} ... 采用自动齿轮档选择时齿轮档 1 的最小主轴转速

g_{1max} ... 采用自动齿轮档选择时齿轮档 1 的最大主轴转速

n_{2max} ... 齿轮档 2 的最大主轴转速

g_{2min} ... 采用自动齿轮档选择时齿轮档 2 的最小主轴转速

g_{2max} ... 采用自动齿轮档选择时齿轮档 2 的最大主轴转速

图 17-3　齿轮档切换，在两个齿轮档之间选择

齿轮档切换过程

预选了新的齿轮档时，切换过程如下：

1. 切换
   以下两个 NC/PLC 接口信号置位：
   DB31, ... DBX82.0-82.2（设定齿轮档 A 至 C）
   DB31, ... DBX82.3（切换齿轮档）
   根据以下 NC/PLC 接口信号置位的时间：
   DB31, ... DBX18.5（往复转速）
   主轴启用往复加速度或转速控制运行/位置闭环控制运行加速度制动，直至到达停止状态。
   最迟在主轴达到静止状态后：
   DB31, ... DBX61.4（轴/主轴停止）
   可通过以下 NC/PLC 接口信号激活往复运行：
   DB31, ... DBX18.5（往复转速）

   原则上不使用往复运行也能切换至新的齿轮档。
   切换至新的齿轮档后，PLC 程序会将下面两个 NC/PLC 接口信号置位：
   DB31, ... DBX16.0-16.2（实际齿轮档 A 至 C）
   DB31, ... DBX16.3（齿轮档已切换）
2. 齿轮档切换结束
   以下 NC/PLC 接口信号置位时，齿轮档切换结束（主轴运行方式“往复模式”取消）：
   DB31, ... DBX16.3（齿轮档已切换）
   在电机静止状态下，参数组切换为新的实际齿轮档的伺服参数组和插补参数组。
   通过 NC/PLC 接口信号：
   DB31, ... DBX16.3（齿轮档已切换）
   NC 得到反馈信息：新齿轮档生效，可结束往复运行。
   NC/PLC 接口信号：
   DB31, ... DBX82.3（切换齿轮档）
   由 NC 复位，
   并触发 PLC 程序对以下 NC/PLC 接口信号的复位：
   DB31, ... DBX16.3（齿轮档已切换）

   此时以下 NC/PLC 接口信号设置与否则不重要：
   DB31, ... DBX18.5（往复转速）

   选择参数组时，信号提示的实际齿轮档为相关信息，
   其应与设定齿轮档相同。
   若非此情形，则会在启用以下机床数据设置时：
   MD11410 $MN_SUPPRESS_ALARM_MASK，位 3 = 0
   触发报警 22010。
   PLC 通过以下 NC/PLC 接口信号应答齿轮档切换后：
   DB31, ... DBX16.3（齿轮档已切换）
   主轴处于转速控制运行中（DB31, ... DBX84.7）。

   PLC 和 NC 间信号交换的更多信息参见“A2: 不同的 NC/PLC 接口信号与功能 (页 43)”。

第二齿轮档数据组

自动齿轮档切换 M40 可通过另一个可配备的齿轮档数据组扩展。

第二齿轮档数据组只和刚性攻丝（G331、G332）功能配合使用，用于实现对主轴转速和电机转矩的高效调整。

为主主轴设置以下位来激活此功能：

MD35010 $MA_GEAR_STEP_CHANGE_ENABLE，位 5 = 1

第二齿轮档数据组使用的齿轮档的数量通过以下机床数据定义：

MD35092 $MA_NUM_GEAR_STEPS2

启用以下设置时，第二齿轮档数据组将被取消激活：

MD35092 $MA_NUM_GEAR_STEPS2 = 0（缺省设置）

此时若 M40 生效，那么齿轮档选择将基于第一齿轮数据组进行。

说明

第二数据组中的齿轮档数量可与第一个数据组中不同。启用 M40 时，若未找到与编写的转速相匹配的齿轮档，则不执行齿轮档切换（特例参见“M40：转速超出配置的开关阈值时的自动齿轮档选择 (页 1509)”）。

刚性攻丝 G331/G332 的典型程序流程的更多信息请参见：

文档：

编程手册之基本原理分册；行程指令

第二齿轮档数据组的齿轮档 1 至 5 可通过下列机床数据进行设置：

机床数据	含义
MD35112 $MA_GEAR_STEP_MAX_VELO2[n]	自动齿轮档切换的最大转速
MD35122 $MA_GEAR_STEP_MIN_VELO2[n]	自动齿轮档切换的最小转速
MD35212 $MA_GEAR_STEP_POSCTRL_ACCEL2[n]	位置闭环控制运行下的加速度

说明

机械条件相关的伺服参数组的数量保持不变。还可为主轴配置五个机械齿轮档，为进给轴模式配置一个。

转速限制针对每个齿轮档通过以下机床数据配置，与各种开关阈值无关。

机床数据	含义
MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[n]	齿轮档的最大转速
MD35140 $MA_GEAR_STEP_MIN_VELO_LIMIT[n]	齿轮档的最小转速

对于刚性攻丝（G331、G332），还可将转速限制在电机的线性加速度范围内。为此，在以下机床数据中针对齿轮档设定线性电机特性曲线范围的转速上限：

MD35550 $MA_DRILL_VELO_LIMIT[n]

在零件程序中设定齿轮档

M40 生效时自动选择齿轮档

齿轮档由控制系统自动选择。此时系统会检查哪一个齿轮档适用于编写的主轴转速（S...）。若能确定一个与当前实际齿轮档不同的齿轮档，则会置位以下 NC/PLC 接口信号：

DB31, ... DBX82.3（切换齿轮档）

DB31, ... DBX82.0-82.2（设定齿轮档 A 至 C）

确定匹配的齿轮档后，仅当新转速不在当前齿轮档所允许的范围内时，系统才会请求齿轮档切换。

必要时，系统会将转速限制为当前齿轮档的最大转速，或提升至最小转速，此时以下 NC/PLC 接口信号将置位：

DB31, ... DBX83.1（设定转速已限制）

DB31, ... DBX83.2（设定转速已提升）

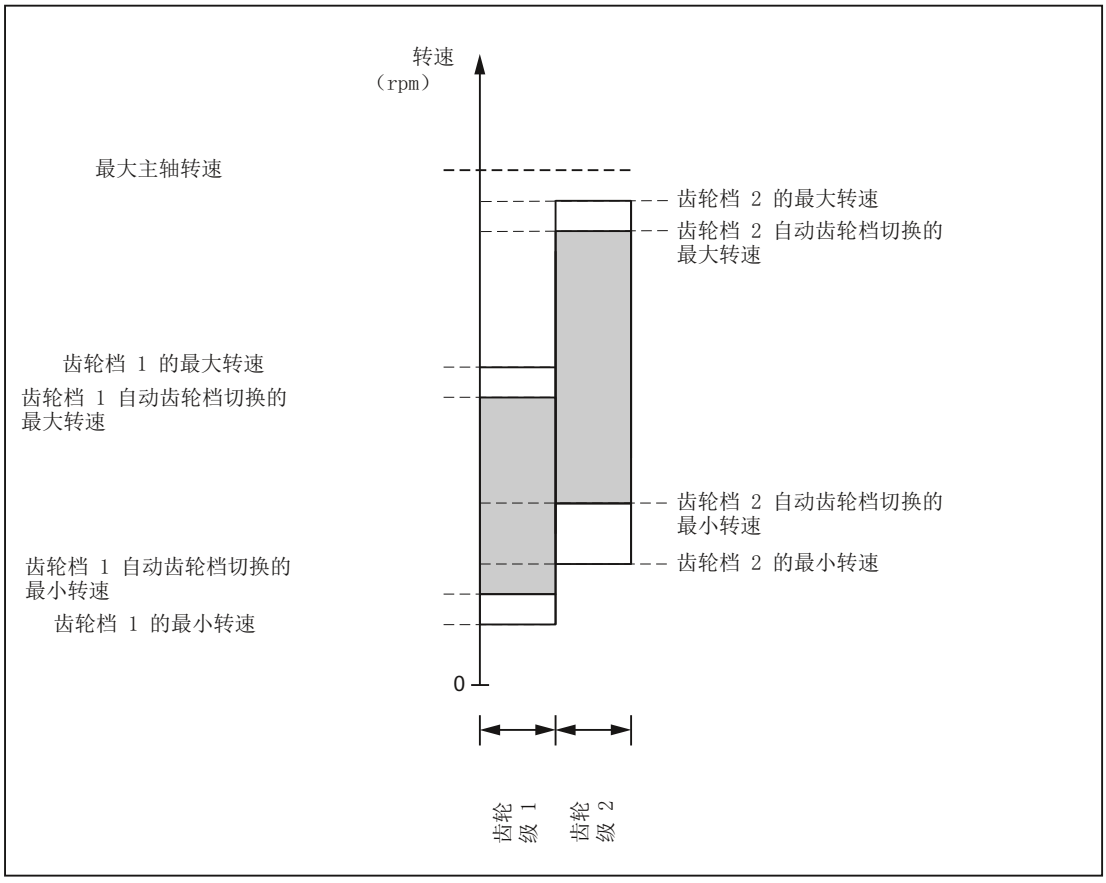


图 17-4　　示例：自动齿轮档选择（M40）中两个转速范围相叠的齿轮档

说明

使用 M40 进行自动齿轮档切换时，主轴必须处于控制模式下并启用一个 S 字。否则齿轮档切换会被拒绝，并发出以下报警：

Alarm 22000“无法切换齿轮档”

说明

选择自动齿轮档切换时，生效的附加中间齿轮比不会被考虑在内。

通过 M41 至 M45 固定设定齿轮档

齿轮档可在零件程序中通过 M41 至 M45 固定设定。

若通过 M41 至 M45 设定了一个与当前实际齿轮档不同的齿轮档，那么以下 NC/PLC 接口信号将置位：

DB31, ... DBX82.3（切换齿轮档）

DB31, ... DBX82.0-82.2（设定齿轮档 A 至 C）

此时，编写的主轴转速（S...）将基于该固定设定的齿轮档：

- 若编写了一个超出固定设定齿轮档转速上限（MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT）的主轴转速，系统会将其限制为上限值，并置位以下 NC/PLC 接口信号：
  DB31, ... DBX83.1（设定转速已限制）
- 若编写了一个低于固定设定齿轮档转速下限（MD35140 $MA_GEAR_STEP_MIN_VELO_LIMIT）的主轴转速，系统会将其提升为下限值，并置位以下 NC/PLC 接口信号：
  DB31, ... DBX83.2（设定转速已提升）

程序段切换

在零件程序段中编写齿轮档切换时，齿轮档切换程序段一直保持生效，直至 PLC 结束齿轮档切换。

此效果相当于以下 NC/PLC 接口信号置位：

DB21, ... DBX6.1（读取禁止）

通过 PLC 使用 FC18 设定齿轮档

可使用功能块 FC18 在复位状态下以及所有运行方式下设定齿轮档切换。

使用 FC18 设定转速和旋转方向时，可向 NC 请求选择与转速相匹配的齿轮档。这相当于 M40 下的自动齿轮档切换。

以下情形下齿轮档不会切换：

- 主轴通过 FC18 定位。
- 主轴在进给轴模式下运行。

功能块 FC18 的更多详细信息参见“P3：SINUMERIK 840D sl 的 PLC 基本程序 (页 945)”章节。

在同步动作中设定齿轮档

在同步动作中可通过以下方式请求齿轮档切换：

- `DO M40 S...`
  为编写的转速 `S...` 自动选择齿轮档
- `DO M41...M45`
  直接选择齿轮档 1 ... 5
- `DO M70`
  基于 MD35014 $MA_GEAR_STEP_USED_IN_AXISMODE = 1 ... 5
  （参见“执行 M70 时启用的齿轮档 (页 1477)”）

以下情形下齿轮档**不会**切换：

- 主轴通过同步动作定位。
- 主轴在进给轴模式下运行。

说明

更多信息请见“在零件程序中设定齿轮档”部分。
特例：
在同步动作中设定齿轮档不会影响程序段切换。

手动设定齿轮档

不借助零件程序，不通过 NC 请求也可在机床上切换齿轮档。这例如可通过直接手动切换齿轮档来实现。

为了选择匹配的参数组，必须将当前齿轮档的信息提供给 NC。为此，控制系统或零件程序必须处于复位状态。

前提条件

修改以下 NC/PLC 接口将齿轮档接收至 NC：
DB31, ... DBX16.0-16.2（实际齿轮档 A 至 C）

这三个位必须在运行期间保持稳定。

接收完成时会通过以下 NC/PLC 接口信号对 PLC 进行应答：
DB31, ... DBX82.0-82.2（设定齿轮档 A 至 C）
。

以下 NC/PLC 接口信号不可置位：
DB31, ... DBX16.3（齿轮档已切换）

若通过 PLC 使用 DB31, ... DBX16.0-16.2 设定信齿轮档时位置闭环控制生效，则须在切换期间将其取消。

齿轮档切换期间的 NC 停止

以下情形下将无法通过 NC/PLC 接口信号：
DB21, ... DBX7.4（NC 停止）
实现主轴停止：

- 主轴尚不处于用于齿轮档切换的往复模式下。
- 以下 NC/PLC 接口信号不存在：
  DB31, ... DBX16.3（齿轮档已切换）

说明

终止方法：
DB31, ... DBX2.2（删除剩余行程/主轴复位）
或
DB31, ... DBX16.3（齿轮档已切换）
与对应的实际齿轮档反馈信息：
DB31, ... DBX16.0-16.2（实际齿轮档）

齿轮档切换后的主轴特定

齿轮档切换完成后，主轴的特性取决于以下初始条件：

- 若齿轮档切换前主轴处于停止状态（M5、FC18：“停止主轴旋转”），处于定位模式或进给轴模式下，那么齿轮档切换结束后 M5（主轴停止）将生效。
- 若在齿轮档切换前编写了旋转方向
  （M3、M4、FC18：“启动主轴旋转”），那么最后编写的的旋转方向和转速将在齿轮档切换后重新生效。主轴会以新的齿轮档加速至最后编写的主轴转速（S...）。
- 若齿轮档切换前位置闭环控制生效（SPCON），那么其会在齿轮档切换后被重新激活。可开始执行零件程序中的下一个程序段。

特殊性

对于齿轮档切换须注意以下几点：

- 选择以下 NC/PLC 接口信号时：
  DB31, ... DBX20.1（V/f 运行启动切换）
  齿轮档切换不会结束。

  此时会输出设定值 0。
  齿轮档切换的应答通过以下 NC/PLC 接口信号进行：
  DB31, ... DBX16.3（齿轮档已切换）
- 通过 PLC 结束齿轮档切换前，必须将信号“斜坡函数发生器快速停止”复位。
- NC 复位时，齿轮档切换过程结束，不输出报警。
  NC/PLC 接口信号：
  DB31, ... DBX16.0-16.2（实际齿轮档 A 至 C）
  发送的齿轮档会由 NC 接收。

通过 FC17 进行的星形/三角形切换

对于数字化主主轴驱动，在主轴运转时也可通过 FC17 沿两个方向从星形切换为三角形连接。此自动切换操作由 FC17 中定义的切换逻辑控制，其切换时间可由用户针对主轴设置。

功能块 FC17 的更多详细信息参见“P3：SINUMERIK 840D sl 的 PLC 基本程序 (页 945)”章节。

17.4.2 主轴齿轮档 0

技术背景

对于使用可切换主轴负载齿轮箱的机床，有时可能会出现电机和负载（工件/刀具）间的传动链断开的情形。 例如在执行齿轮档切换期间按下复位或急停键，或者安装期间首次调试时都可能出现此状态。 控制系统必须能够识别出传动链断开的状态，并执行下一次齿轮档切换。

功能

对于脱接的齿轮档，PLC 会通过接口信号 DB31, ... DBX16.0-2（实际齿轮档 A 至 C）将二进制编码值“0”（≙ 齿轮档 0）传输至 NC：

DB31, ... DBX16.0-2 = 0

该值用于向控制系统提示传动链断开的状态。

对齿轮档切换的影响

零件程序中的齿轮档切换

PLC 发出的实际齿轮档信号在零件程序开始时由 NC 读取。 若此时读取到的实际齿轮档为“0”，那么 NC 将执行下一次齿轮档切换，并执行与 PLC 的齿轮档切换信号对话。 若读取的值大于“0”，那么编程时即完成了请求的齿轮档和生效齿轮档之间的比较。 若两个齿轮档相同，则不执行齿轮档切换；若编写有轨迹运行，则其也不会中断。

同步动作、FC18 和 DBB30 中的齿轮档切换

进行齿轮档切换时，PLC 发出的实际齿轮档信号总是会由 NC 进行分析。 读取值为“0”时，NC 总是会执行齿轮档切换。若读取值大于“0”，则会比较请求的齿轮档和生效齿轮档。仅当两个值不同时，才会通过 PLC 执行齿轮档切换，并输出 NC/PLC-接口信号 DB31, ... DBX82.3（切换齿轮档）。

前提条件

- **DB31, ... DBX16.0-2 = 0 的输出**
  对于脱接的齿轮档，必须通过 PLC 将齿轮档 0 输入至 NC/PLC 接口 DB31, ... DBX16.0-2（实际齿轮档 A 至 C）。
- **使能齿轮档切换**
  读取到齿轮档 0 后，齿轮档切换的前提条件是齿轮档切换的总体使能：
  MD35010 $MA_GEAR_STEP_CHANGE_ENABLE（参数设置齿轮档切换）
  MD35090 $MA_NUM_GEAR_STEPS（建立的齿轮档的数量）
  MD35092 $MA_NUM_GEAR_STEPS2（第 2 齿轮档数据组：建立的齿轮档的数量），基于 MD35010 $MA_GEAR_STEP_CHANGE_ENABLE，位 5 = 1（刚性攻丝）
- **PLC 用户程序/上电 ASUB**
  通过 PLC 用户程序或上电 ASUB，能够在齿轮档脱接（齿轮档 0）的情形下确保在主轴运动前编写一个齿轮档切换请求。 这例如可通过在 ASUB 中编写 M41 实现。 主轴运动（例如 JOG 运行方式下或进给轴模式下）本身不会生成齿轮档切换。

示例

示例：上电后切换至第一齿轮档

1. 上电。
2. PLC 用户程序在机械环境中确定“齿轮档已脱接”状态。

3. PLC 通过置位以下信号将“齿轮档已脱接”状态传输至 NC：
   DB31, ... DBX16.0-2 = 0
4. 零件程序启动或上电 ASUB。
5. 执行 N05（零件程序见下文）：
   执行向齿轮档 1 的切换。
   NC：
   – 置位以下 NC/PLC 接口信号：
     DB31, ... DBX82.3（切换齿轮档）
   – 将设定齿轮档 1 的信息发送给 PLC：
     DB31, ... DBX82.0 = 1
     DB31, ... DBX82.1 = 0
     DB31, ... DBX82.2 = 0
6. 齿轮档的机械切换，应答
   若切换了齿轮档，则由 PLC：
   – 置位以下 NC/PLC 接口信号：
     DB31, ... DBX16.3（齿轮档已切换）
   – 将实际齿轮档 1 的信息发送给 NC：
     DB31, ... DBX16.0 = 1
     DB31, ... DBX16.1 = 0
     DB31, ... DBX16.2 = 0
7. 执行 N80：
   基于优化齿轮档切换频率的原则，不执行齿轮档切换。

零件程序：

程序代码	注释
N05 M41	; 选择齿轮档 1
...	
N80 M41	; 若切换至齿轮档 1，则不执行齿轮档切换。

主轴 1（AX5）的配置数据：

MD35010 $MA_GEAR_STEP_CHANGE_ENABLE[AX5] = 1（使能齿轮档切换）

17.4.3 确定主轴齿轮档

主轴的当前齿轮档可通过系统变量读取：

- 在同步动作或零件程序（会触发预处理停止）中写入以下系统变量，状态显示在操作界面中：

$VC_SGEAR[<n>]	当前换入的主轴齿轮档 $VC_SGEAR 读取的是 PLC 发送的实际齿轮档。 取值范围： 0 ... 5
$AC_SGEAR[<n>]	生效的主轴齿轮档 $AC_SGEAR 读取的是主处理中的设定齿轮档。 取值范围： 1 ... 5 根据此齿轮档为主轴激活数据组。

说明

程序段搜索时，实际齿轮档（$VC_SGEAR[<n>]）可能有别于设定齿轮档（$AC_SGEAR[<n>]），这是因为程序段搜索期间未进行齿轮档切换。因此借助 $VC_SGEAR[<n>] 和 $AC_SGEAR[<n>] 也可查询程序段搜索后是否需要执行齿轮档切换。

- 在不带预处理停止的零件程序中，通过系统变量：

$P_SGEAR[<n>]	设定齿轮档 $P_SGEAR 读取的是零件程序中编写的（M41 ... M45）、M40 下选择的或 M70 下配置的齿轮档。
$P_SEARCH_SGEAR[<n>]	程序段搜索:齿轮档 M 功能 $P_SEARCH_SGEAR 读取的是程序段搜索中收集的、最后编写的齿轮档 M 功能。

17.4.4 齿轮档切换时的参数组选择

伺服参数组

伺服参数组 1 至 6 用于在主轴齿轮档切换时根据机床的属性变化调整位置控制器。

齿轮档切换时的参数组选择

齿轮档切换时，齿轮档参数组（插补参数组）也会随之切换。此外根据以下机床数据中的设置，伺服数据组也可能切换：

MD35590 $MA_PARAMSET_CHANGE_ENABLE（参数组切换使能）

值	含义
0	系统内部的参数组选择 伺服参数组采用固定分配。 其中： ● 对于进给轴模式下的轴和主轴，一般情况下第一个参数组生效。 特例： 使用 G33、G34、G35、G331 和 G332 时，为相关轴激活下列编号的数据组： 主主轴齿轮档 + 1（对应参数组编号 2 ... 6） ● 对于主进给轴模式下的主轴，系统会根据齿轮档设置匹配的参数组。
1	除了系统内部的参数组选择外，也可进行“外部”参数组选择： ● 通过 PLC（DB31, ... DBX 9.0 - 9.2） ● 通过在零件程序或同步动作中写入指令 SCPARA 不过系统内部参数组选择具有更高优先级。 提示：值 1 只适用于轴。
2	伺服参数组只通过 PLC（DB31, ... DBX 9.0 - 9.2）或在零件程序或同步动作中编写 SCPARA 来设定（用于轴和主轴）。 上电后选择参数组 1。

主轴模式

MD35590 $MA_PARAMSET_CHANGE_ENABLE = 0 或 1

参数组按照“齿轮档 + 1”选择。

生效的齿轮档通过以下接口获取：

DB31, ... DBX16.0-16.2（实际齿轮档 A 至 C）

生效的参数组在以下接口输出：

DB31, ... DBX69.0-69.2 （控制器参数组 A 至 C）

5 个齿轮档每个都有一个参数组，由 NC 如下指定：

数据组	NC/PLC 接口 DBX 69.2 / 69.1 / 69.0	参数组 编号	参数组 索引 [n]
进给轴模式	最后生效的齿轮档	1	0
齿轮档 1	001	2	1
齿轮档 2	010	3	2
齿轮档 3	011	4	3
齿轮档 4	100	5	4
齿轮档 5	101 110 111	6	5

进给轴模式下的主轴

若主轴处于进给轴模式下，那么伺服中会选择参数组索引“0”（注意 MD35590 $MA_PARAMSET_CHANGE_ENABLE！）

齿轮档切换特性取决于以下机床数据中的设置：

MD35014 $MA_GEAR_STEP_USED_IN_AXISMODE（M70 下进给轴模式的齿轮档）

若没有为进给轴模式配置齿轮档（MD35014 = 0），M70 下将不会自动执行齿轮档切换（缺省设置）。最后的齿轮档会保存在系统内部，并在下一次主轴编程时与对应参数组一起重新生效。

相反的情形下，若为进给轴模式配置了齿轮档（MD35014 = 1 ... 5），那么在执行 M70 期间会切换至齿轮档 1 ... 5。在从进给轴模式切换至主轴模式时，通过 M70 换入的齿轮档保持生效。主轴模式下在 M70 前生效的齿轮档不会被自动重新换入。

另见“执行 M70 时启用的齿轮档 (页 1477)”。

负载齿轮箱系数

可在进给轴模式下为每个齿轮档配置一个正的或负的**负载齿轮箱系数**。

此系数分为分子和分母通过以下机床数据设置：

MD31050 $MA_DRIVE_AX_RATIO_DENOM[n]（负载齿轮比分母）

MD31060 $MA_DRIVE_AX_RATIO_NUMERA[n]（负载齿轮比分子）

正负系数的设置范围一样大。

不可输入“0”值。

说明

若配置了一个间接编码器，且负载齿轮比系数变更，那么参考将丢失且 NC/PLC 接口信号：DB31, ... DBX60.4/60.5（已回参考点/已同步 1/2)
针对相关测量系统复位。

文档

闭环控制和伺服参数组的更多信息参见：

- 功能手册之基本功能分册；章节“速度，设定值/实际值系统，闭环控制（G2）”
- 编程手册之工作准备分册；章节：可编程的伺服参数组

17.4.5 中间齿轮比

应用和功能

通过可配置的中间齿轮比可调整各种动力刀具。刀具侧的中间齿轮比以乘法作用于电机齿轮比/负载齿轮比。

中间齿轮比通过以下机床数据设置：

MD31066 $MA_DRIVE_AX_RATIO2_NUMERA（中间齿轮比分子）

MD31064 $MA_DRIVE_AX_RATIO2_DENOM（中间齿轮比分母）

用于中间齿轮比的刀具侧编码器
通过以下机床数据配置：
MD31044 $MA_ENC_IS_DIRECT2（中间齿轮比上的编码器）

该机床数据被修改后，可通过调试软件 SinuCOM-NC 或通过 HMI 操作面板上的软键使用“激活机床数据”功能来激活经修改的参数设置。现有的电机齿轮比/负载齿轮箱则是上电后生效。

换刀

若需在换刀时一并切换中间齿轮比，那么用户必须通过中间齿轮比的机床数据重新配置其分母与分子的传动比。

示例：

⚠ 小心

配置故障

用户须负责在适宜的时间段停止、立即执行机床数据修改并通过“激活机床数据”功能激活。

为换入的传动比为 2:1 的刀具配置一个匹配的中间齿轮比，并在零件程序中通过 NEWCONF 指令立即激活。

程序代码

```
N05 $MA_DRIVE_AX_RATIO2-NUMERA[AX5] = 2
M10 $MA_DRIVE_AX_RATIO2-DENOM[AX5] = 1
N15 NEWCONF
```

切换

借助“激活机床数据”功能立即切换至新的传动比。技术上而言，对应的机械切换要持续一些时间，因为要通过动力刀具换入另一个辅助刀头。

说明

在静止状态下切换无急动。因此用户须采取相应的预防措施。

对于运动中需要进行切换，以及需要平滑或柔和转速过渡的应用，可通过驱动端现有的转速设定值滤波器应对。

对控制技术关联的更多说明请见“G2：速度、设定值/实际值系统、闭环控制 (页 363)”。

17.4.6 未应答的齿轮档切换

运行方式切换

未应答的齿轮档切换无法通过运行方式切换（例如切换至 JOG）中断。

切换最多可延迟以下机床数据中输入的时间值：
MD10192 $MN_GEAR_CHANGE_WAIT_TIME

若未在此时间内应答齿轮档切换，那么 NC 将触发报警。

其他事件

触发重整的事件同样会等待齿轮档切换结束。

机床数据：
MD10192 $MN_GEAR_CHANGE_WAIT_TIME
用于设置对齿轮档切换的等待时间。
此时间结束后，若齿轮档切换仍未结束，那么 NC 将响应并发出报警。

以下事件具有类似的特性：

- 用户 ASUB
- 模式切换
- 删除剩余行程
- 跨通道取轴
- 激活 PI 用户数据
- 激活 PI 服务机床数据
- 切换跳转，切换 DryRun
- 在运行方式下编辑
- 补偿组报警
- 溢出转存
- G33、G34、G35 下的快速退刀
- 子程序级终止，子程序终止

上电后特性

上电后，在 NC 复位状态下可通过 PLC 控制系统设定机床上生效的齿轮档。

NC 会选择对应的参数组并通过 NC/PLC 接口信号：
DB31, ... DBX82.0-82.2（设定齿轮档 A 至 C）
向 PLC 反馈。

17.4.7 通过往复模式进行齿轮档切换

在通过 M40（自动选择齿轮档）或 M41 至 M45 设定了新的齿轮档时，主轴处于往复模式中。

在往复运行中，主轴电机的旋转方向以较小的时间间隔交替反转。借助电机的往复运动，例如能够更顺利地切换至新的齿轮档。原则上不使用往复运行也能切换至新的齿轮档。

可采用下列往复类型：

- 通过 NC 往复
- 通过 PLC 往复
- 使用 FC 18 往复
  文档：
  功能手册之基本功能分册；PLC 基本程序（P3）

调试：机床数据

- MD35010 $MA_GEAR_STEP_CHANGE_ENABLE（参数设置齿轮档切换）
- MD35410 $MA_SPIND_OSCILL_ACCEL（往复时的加速度）
- MD35430 $MA_SPIND_OSCILL_START_DIR（往复时的起始方向）
- MD35400 $MA_SPIND_OSCILL_DES_VELO（往复转速）
- MD35450 $MA_SPIND_OSCILL_TIME_CCW（M4 方向的往复时间）
- MD35440 $MA_SPIND_OSCILL_TIME_CW（M3 方向的往复时间）

调试：NC/PLC 接口信号

- DB31, ... DBX16.0 - 2（实际齿轮档）
- DB31, ... DBX16.3（齿轮档已切换）
- DB31, ... DBX18.4（通过 PLC 往复）
- DB31, ... DBX18.5（往复使能）
- DB31, ... DBX18.6（设定旋转方向，顺时针）
- DB31, ... DBX18.7（设定旋转方向，逆时针）
- DB31, ... DBX61.4（主轴静止）
- DB31, ... DBX82.0 - 2（设定齿轮档）
- DB31, ... DBX82.3（切换齿轮档）
- DB31, ... DBX83.5（主轴位于设定区域内）
- DB31, ... DBX84.6 (Aktive Spindelbetriebsart:Pendelbetrieb)

过程说明：齿轮档切换

参数设置：MD35010 $MA_GEAR_STEP_CHANGE_ENABLE，位 0 = 1

过程：

- 主轴制动。
  此时制动过程对应 M5 运动。
- NC/PLC 接口信号 NC → PLC：
  - DB31, ... DBX84.6 == 1（往复模式）
  - DB31, ... DBX82.3 == 1（切换齿轮档）
  - DB31, ... DBX82.0 - 2 ==（设定齿轮档）
  - DB31, ... DBX61.5 == 0（位置控制器生效）
- 现在可“脱接”负载齿轮箱。
- 现在可通过 NC 或 PLC 执行往复运行（见下文）。
- 在 PLC 向 NC 反馈“齿轮档切换”和“新齿轮档生效”后，继续执行程序：
- DB31, ... DBX16.3 = 1（齿轮档已切换）
- DB31, ... DBX16.0 - 2（实际齿轮档）
- 对于采用间接编码器（电机编码器）的测量系统，回参考点状态会被删除：
  DB31, ... DBX60.4 / 5 = 0

通过 NC 往复

请求

- DB31, ... DBX18.5 = 1（往复使能）

状态反馈信息

- DB31, ... DBX84.6 == 1（往复模式）
- DB31, ... DBX82.3 == 1（切换齿轮档）
- DB31, ... DBX82.0 - 2 ==（设定齿轮档）
- DB31, ... DBX61.5 == 0（位置控制器生效）

现在可“脱接”负载齿轮箱。

启动往复

现在可通过 PLC 用户程序将往复使能置位：

- DB31, ... DBX18.5 = 1（往复使能）

在取决于旋转方向的往复时间内，电机以起始旋转方向以往复加速度加速至往复转速：

- MD35410 $MA_SPIND_OSCILL_ACCEL（往复时的加速度）
- MD35430 $MA_SPIND_OSCILL_START_DIR（往复时的起始方向）
- MD35400 $MA_SPIND_OSCILL_DES_VELO（往复转速）
- MD35450 $MA_SPIND_OSCILL_TIME_CCW（M4 方向的往复时间）
- MD35440 $MA_SPIND_OSCILL_TIME_CW（M3 方向的往复时间）

切换旋转方向

在取决于旋转方向的往复时间届满后，电机以往复加速度制动，并在另一往复时间内以相反的旋转方向加速至往复转速。

通过 PLC 往复

请求

- DB31, ... DBX18.4 = 1（通过 PLC 往复）
- DB31, ... DBX18.5 = 1（往复使能）

状态反馈信息

- DB31, ... DBX84.6 == 1（往复模式）
- DB31, ... DBX82.3 == 1（切换齿轮档）
- DB31, ... DBX82.0 - 2 ==（设定齿轮档）
- DB31, ... DBX61.5 == 0（位置控制器生效）

现在可“脱接”负载齿轮箱。

启动往复

现在可通过 PLC 用户程序将“往复使能”置位：

- DB31, ... DBX18.7 = 1（设定旋转方向，顺时针）或 DB31, ... DBX18.6 = 1（设定旋转方向，逆时针）

电机以设定旋转方向以往复加速度加速至往复转速：

- MD35410 $MA_SPIND_OSCILL_ACCEL（往复时的加速度）
- MD35400 $MA_SPIND_OSCILL_DES_VELO（往复转速）

切换旋转方向

往复时间以及旋转方向切换均需要在 PLC 用户程序中实现。

往复模式结束

往复模式结束后，主轴重新处于控制模式，并自动进入通过 SPCON 或 SPCOF 定义的模式。

所有针对齿轮档的限值（齿轮档的转速上限/下限等）对应设定的实际齿轮档的值。

程序段切换

若将主轴切换至往复模式（DB31, ... DBX82.3 = 1 （切换齿轮档）），则 NC 程序的执行将会停止。

在齿轮档切换完成后（DB31, ... DBX16.3 == 1（齿轮档已切换）），系统会继续执行该 NC 程序。

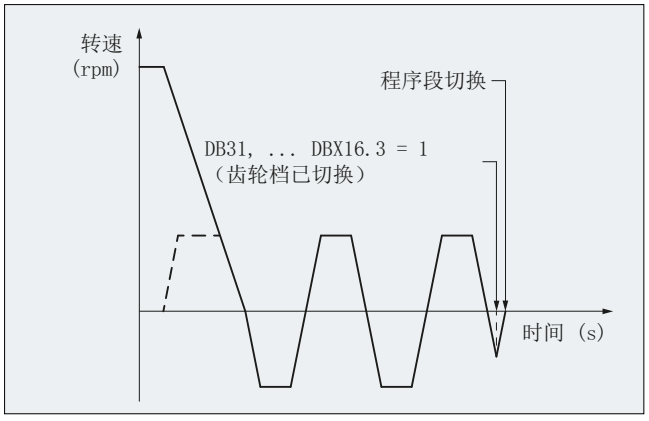
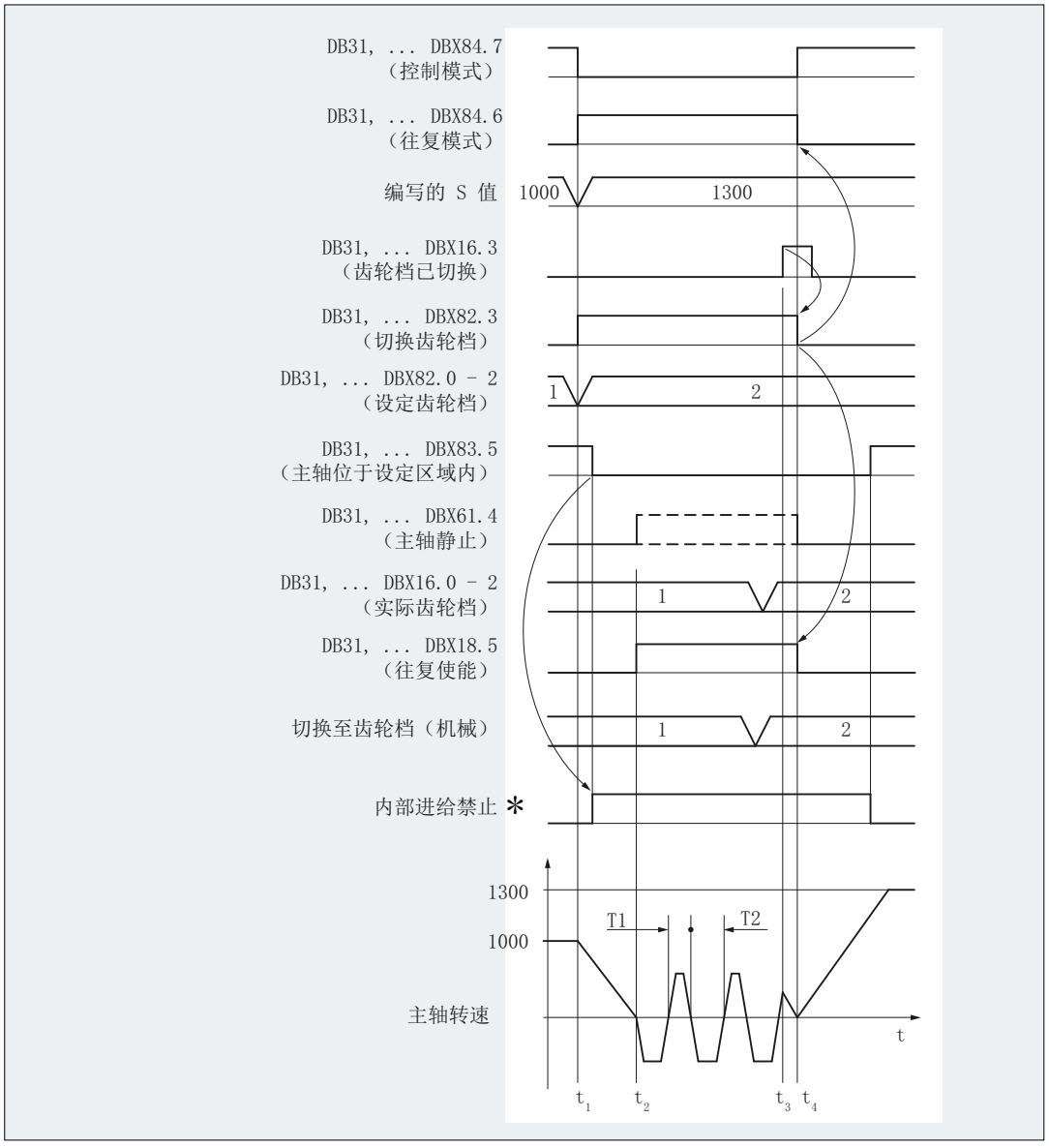


图 17-5　　往复模式后的程序段切换

示例性信号图



t_1： Die NC erkennt aufgrund des neuen programmierten **S-Wertes S1300** im NC-Programm, die Notwendigkeit zum Umschalten der Getriebestufe (1. → 2.Getriebestufe) und fordert die Freigabe von der PLC an:

- NC → PLC：DB31, ... DBX82.3 = 1（切换齿轮档）

NC 将程序执行禁止（内部进给禁止*）

t_2： NC 向 PLC 发送**主轴转速 == 0** 信息：

- DB31, ... DBX61.4 = 1 (主轴静止)

PLC 用户程序给予往复使能。

- PLC → NC：DB31, ... DBX18.5 = 1（往复使能）
  提示
  往复使能最迟须在时间点 t_2 置位。

NC 开始往复。

t_3： 在机床上执行齿轮档切换。

PLC 用户程序将切换信息反馈给 NC，并删除往复使能：

- PLC → NC：DB31, ... DBX16.0 - 2 = 2（实际齿轮档）
- PLC → NC：DB31, ... DBX16.3 = 1（齿轮档已切换）
- PLC → NC：DB31, ... DBX18.5 = 0（往复使能）

t_4： NC 通过将对 PLC 的请求复位结束齿轮档切换，并通过将主轴运行模式从往复模式切换至控制模式来结束往复：

- NC → PLC：DB31, ... DBX82.3 = 0（切换齿轮档）
- DB31, ... DBX84.6 = 0 (Aktive Spindelbetriebsart:Pendelbetrieb)
- DB31, ... DBX84.7 = 1 (Aktive Spindelbetriebsart:Steuerbetrieb)

NC 使能下一程序段用于加工，并将主轴加速至新的 S 值 S1300。

*： 以下情形下，系统将内部进给禁止置位：

- 因在 NC 程序中编写 S 值而发生齿轮档切换**并且**
- **加工**程序段生效（非快速移动 G0）

提示

通过同步动作进行齿轮档切换，或通过 PLC 使用 FC 18 设定时，内部进给禁止**不会**置位。

图 17-6 静止主轴上的齿轮档切换

边界条件

- 为进行主轴制动，不必将以下 NC/PLC 接口信号置位：
  DB31, ... DBX4.3（主轴停止）
  请求齿轮档切换时，控制系统内部使得主轴静止。
- 总是通过以下 NC/PLC 接口信号的置位来结束齿轮档切换：
  DB31, ... DBX16.3 = 1（齿轮档已切换）

- 通过以下 C/PLC 接口信号的复位来停止 NC 中的往复运动：
  DB31, ... DBX18.5（往复使能）
  但主轴处于“往复模式”中。
- 在采用齿轮档切换和间接测量系统（MD31050 $MA_ENC_IS_DIRECT = 0 电机编码器）时，主轴同步会丢失。
  齿轮档切换 ⇒ DB31, ... DBX60.4 / 5 = 0（已回参考点/已同步）
  切换齿轮档后，在下一次越过零脉冲时重新对主轴进行同步。

17.4.8 在固定位置切换齿轮档

应用和优点

标准化主轴齿轮越来越多地应用于机床，一方面是为了减少齿轮档切换时的技术时滞，另一方面则是为了享受使用标准化机械组件所带来的成本优势。

固定位置上的齿轮档切换功能能够支持对负载齿轮箱的“定向齿轮档切换”，并会通过 NC 请求另一种控制方案。此时齿轮档切换只能在一个定义的主轴位置上执行。传统负载齿轮箱所启用的往复运动将不再需要。

在固定位置切换齿轮档的过程

在固定位置切换齿轮档

机床数据配置：
MD35010 $MA_GEAR_STEP_CHANGE_ENABLE = 2
下的后续过程：

- 主轴从静止状态或运动状态定位至以下机床数据中配置的位置：
  MD35012 $MA_GEAR_STEP_CHANGE_POSITION
  若从运动状态开始执行齿轮档切换，则保持当前的旋转方向。定位期间主轴处于定位模式下。
  以下 NC/PLC 接口信号会输出：
  DB31, ... DBX84.5（定位模式）

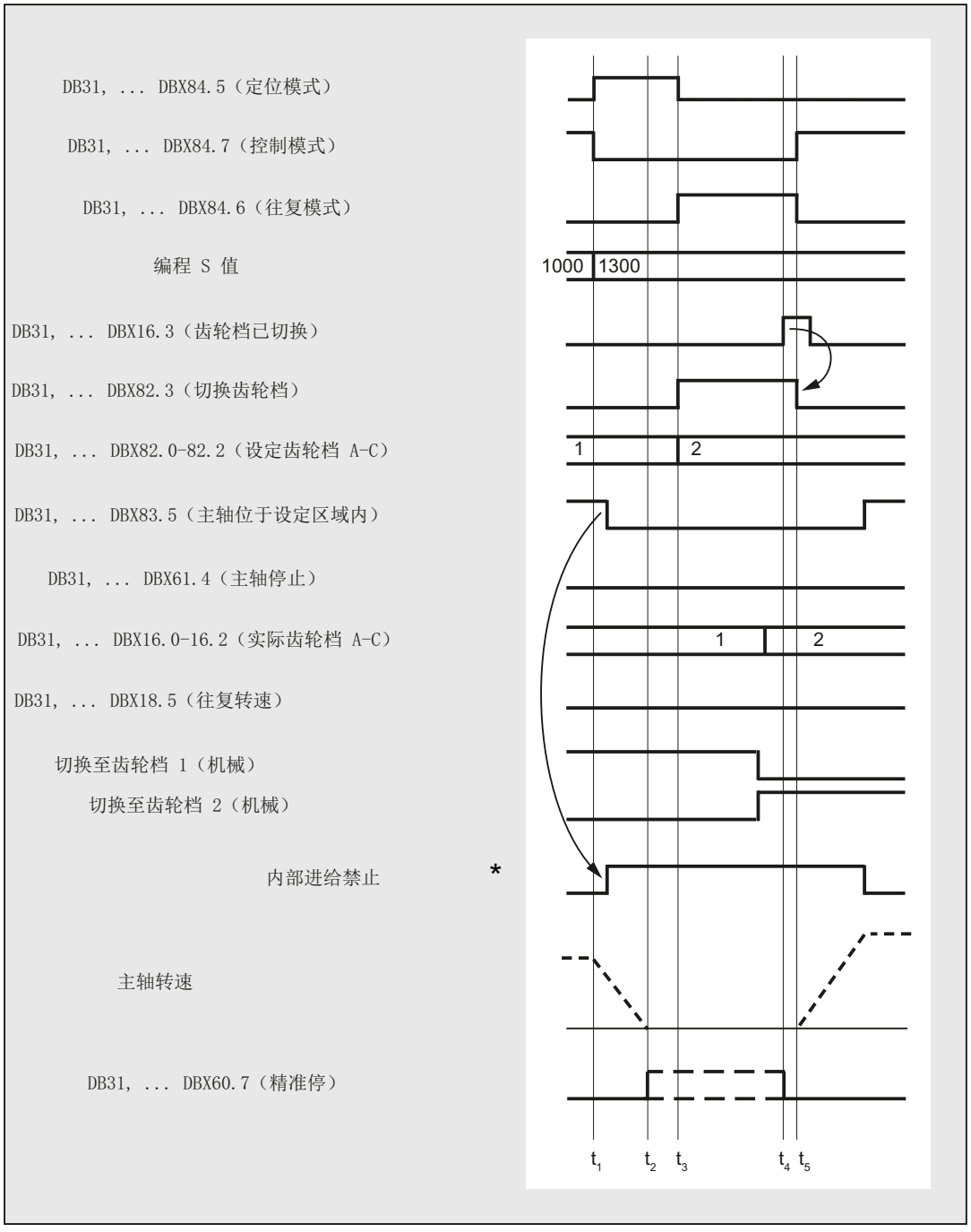
  若未回参考点：
  DB31, ... DBX60.4/5 = 0
  或以下 NC/PLC 接口信号置位：
  DB 31, ... DBX17.4/5（定位时测量系统 1/2 重新同步）
  则定位将延长搜索零脉冲的过程。
- 到达以下机床数据中配置的换刀位置后：
  MD35012$MA_GEAR_STEP_CHANGE_POSITION
  在切换至往复模式前，等待以下机床数据中设定的时间：
  MD35310 $MA_SPIND_POSIT_DELAY_TIME
  并开始齿轮档切换对话。
- 输出 NC/PLC 接口信号：
  DB31, ... DBX84.6（往复模式）
  DB31, ... DBX82.3（切换齿轮档）
  DB31, ... DBX82.0-82.2（设定齿轮档 A 至 C）
- 使用间接编码器（电机编码器）的测量系统生效时，位置闭环控制不取消：
  MD31040 $MA_ENC_IS_DIRECT = 0
  使用直接编码器的测量系统生效时，取消位置闭环控制：
  DB31, ... DBX61.5 = 0
  这是因为至负载的力流中断，无法通过其实现位置闭环控制。
- 若无法启用位置闭环控制，可通过复位“控制器使能”将其取消：
  DB31, ... DBX2.1 = 0

- 机床上齿轮档的机械切换
  为此不需要电机的往复运行。
  NC/PLC 接口信号：
  DB31, ... DBX18.5（往复使能）
  和
  DB31, ... DBX18.4（通过 PLC 往复）
  不用置位。
  原则上现在还是可以启用往复运行。
- 通过 PLC 写入 NC/PLC 接口信号：
  DB31, ... DBX16.0-16.2（实际齿轮档 A 至 C）。

- 发送以下信号后：
  DB31, ... DBX16.3（齿轮档已切换）
  若存在最后生效的运动，则会继续执行。
  间接编码器（电机编码器）的回参考点状态会被删除：
  DB31, ... DBX60.4/5 = 0
  主轴处于转速控制运行下，且以下 NC/PLC 接口信号被输出：
  DB31, ... DBX84.7（控制模式）

齿轮档切换规定位置

在固定位置切换齿轮档的典型时序过程：



t_1： 通过编写 S1300，NC 识别一个新的齿轮档（齿轮档 2），置位接口信号 DB31, ... DBX84.5（定位模式）并禁止下一个零件程序段的执行（= 内部进给禁止*）。

t_2： 主轴停止，并发出准停提示信号。

t_3： 齿轮档切换等待时间

t_4： 新齿轮档接合。PLC 用户将新的（实际）齿轮档传输至 NC，并置位接口信号 DB31, ... DBX16.3（齿轮档已切换）。

t_5： NC 复位接口信号 DB31, ... DBX82.3（切换齿轮档），使能下一个零件程序段用于加工，并将主轴加速至新的 S 值（S1300）。

* ： 以下情形下，内部进给禁止：

- 主轴齿轮档切换通过零件程序编写**并且**
- 一个加工程序段生效（即 G0 不生效）

通过同步动作进行齿轮档切换，或通过 PLC 使用 FC 18 设定时，内部进给禁止**不会**置位。

图 17-7 静止主轴上的齿轮档切换

齿轮档切换位置 MD35012

齿轮档切换位置在以下机床数据中针对每个齿轮档配置：
MD35012 $MA_GEAR_STEP_CHANGE_POSITION

齿轮档切换等待时间 MD35310

定位后，等待以下机床数据中设定的时间：
MD35310 $MA_SPIND_POSIT_DELAY_TIME
直至接收到齿轮档切换请求：
DB31, ... DBX84.6（往复模式）
DB31, ... DBX82.3（切换齿轮档）
和
DB31, ... DBX82.0-82.2（设定齿轮档 A 至 C）

定位特性值/位置

主轴总是沿最短路径（等同于 DC）逼近齿轮档位置。

若未执行回参考点，且主轴处于静止状态（例如上电后），
那么逼近方向取决于以下机床数据：
MD35350 $MA_SPIND_POSITIONING_DIR

若需启用可变的齿轮档切换位置，可通过写入机床数据并于随后通过“激活机床数据”激活修改来实现。
机床数据的修改可通过零件程序或 HMI 进行。

若未到达配置的位置，则会触发报警 22020，且不会发生 NC 和 PLC 间的齿轮档切换信号对话，防止齿轮箱损坏。此报警具有很高权重，因此无法继续执行零件程序，必须消除故障原因。根据经验，定位终止的原因可能是机床数据设置错误，或 PLC 信号不兼容。

速度

定位速度在以下机床数据中针对齿轮档配置：

MD35300 $MA_SPIND_POSCTRL_VELO

NC/PLC 接口信号“主轴补偿/进给补偿”
DB31, ... DBX17.0=0:DB31, ... DBB19
以及
DB31, ... DBX17.0=1:DB31, ... DBB0
在定位中照常生效。
定位速度可通过程序指令 OVRA[Sn] 按百分比修改。

说明

OVRA[Sn] 为模态生效。在齿轮档切换结束后，应将其设置为适宜用于加工的值。

零件程序指令 FA[Sn] 不会影响齿轮档切换中的定位速度。

加速度

加速度值针对齿轮档在以下机床数据中定义：

MD35200 $MA_GEAR_STEP_SPEEDCTRL_ACCEL

与

MD35210 $MA_GEAR_STEP_POSCTRL_ACCEL

加速度可通过程序指令 ACC[Sn] 按比例修改。

说明

ACC[Sn] 为模态生效。在齿轮档切换结束后，应将其设置为适宜用于加工的值。

取决于转速的加速度

“折线式加速度特性曲线”同使用 SPOS 或 FC18 定位时一样生效。

急动度

目前尚无法限制加速度变化。

定位结束

定位结束（DB31, ... DBX84.5）和
往复模式开始（DB31, ... DBX84.6）之间的过渡
取决于达到“准停”（DB31, ... DB60.7）
以及以下机床数据中输入的时间：
MD3510 $MA_SPIND_POSIT_DELAY_TIME

过渡条件的定义一方面会影响齿轮档切换时间，另一方面也会影响到达齿轮档切换位置时的精度。

程序段切换

程序段切换将停止，直至 PLC 发出齿轮档切换结束信号（DB31, ... DBX16.3）。

齿轮档切换结束

齿轮档切换结束后，主轴重新回到控制模式，并自动进入通过 SPCON 或 SPCOF 定义的闭环控制模式。

所有针对齿轮档的限值（齿轮档的转速上限/下限等）对应反馈的实际齿轮档。

前提条件

- 主轴至少须配备一个测量系统。
- 必须能启用位置闭环控制运行，并对其进行过调试。
- 通过零件程序、同步动作或 FC18：“开始主轴定位”调用的 SPOS 必须能无故障执行。

若无法满足所有前提条件，此功能可能无法成功应用。

激活

“在固定位置切换齿轮档”功能通过以下机床数据设置激活：
MD35010 $MA_GEAR_STEP_CHANGE_ENABLE = 2

17.4.9 执行 M70 时启用的齿轮档

技术背景

在某些机床上，可能需要在轴模式期间启用特定齿轮档。

可能的原因包括：

- 在轴模式的伺服参数组中（索引 0），对应一个齿轮档只匹配了一项优化（K_V、前馈控制、滤波器）。同时这些参数组的机床数据不能修改。
- 只有一个机械传动比，和其他齿轮档相比只有较小或没有反向间隙。只有启用该齿轮档时，主轴才能同其他轴一起实现轨迹运动或转换（例如 TRANSMIT）。

功能

此功能激活时，过渡至轴模式时将自动切换为预设的齿轮档。

齿轮档切换集成在 M70 进程中，并在主轴制动后和换入伺服参数组前通过索引 0 的参数组进行（注意 MD35590 $MA_PARAMSET_CHANGE_ENABLE 的设置！）

启用此功能时，齿轮档切换期间 NC 和 PLC 的信号对话的执行方式同编写齿轮档切换（M41 ... M45）时相同。

前提条件

过渡至轴模式时切换齿轮档的前提条件是对齿轮档切换的总体使能：

MD35010 $MA_GEAR_STEP_CHANGE_ENABLE（参数设置齿轮档切换）

MD35090 $MA_NUM_GEAR_STEPS（建立的齿轮档的数量）

激活/取消激活

此功能通过以下机床数据激活/取消：

MD35014 $MA_GEAR_STEP_USED_IN_AXISMODE（M70 下轴模式的齿轮档）

值	含义
0	执行 M70 时不进行隐性齿轮档切换。保留当前齿轮档（缺省设置！）
1 ... 5	执行 M70 时切换至齿轮档 1 ... 5。

前提条件

在固定位置切换齿轮档（MD35010 $MA_GEAR_STEP_CHANGE_ENABLE = 2）

此功能支持“在固定位置切换齿轮档”。此时，执行 M70 时将增加主轴定位的过程。逼近齿轮档切换位置时采用当前齿轮档。

在未编写 M70 的情形下过渡至轴模式

控制系统能够基于编程顺序自动识别出向轴模式的过渡（参见“隐性过渡至进给轴模式 (页 1428)”），并于系统内部生成必要的包含齿轮档切换的 M70 过程。

通过 FC 18 过渡至轴模式

通过 FC 18 过渡至轴模式（“启动轴”）时，不支持隐性齿轮档切换。此时必须通过 PLC 应用程序在切换至轴模式前换入匹配的齿轮档。也可通过 FC 18 进行齿轮档切换（“启动齿轮档切换”）。

从轴模式切换至主轴模式

在从轴模式切换至主轴模式时，通过 M70 换入的齿轮档保持生效。主轴模式下在 M70 前生效的齿轮档不会被自动重新换入。伺服参数组会从参数组 1（索引 0）切换至与齿轮档相匹配的参数组 2 ... 6（索引 1 ... 5）（MD35590 $MA_PARAMSET_CHANGE_ENABLE < 2 时）。

示例

主轴过渡至轴模式时需要换入齿轮档 4。

配置：MD35014 $MA_GEAR_STEP_USED_IN_AXISMODE[<主轴名称>] = 4

程序代码	注释
N05 M3 S1000	
N10 G1 X100 F1000	
N15 M70	; 换入齿轮档 4。
N20 POS[C]=77	
N25 ...	

说明

MD35014 可通过“激活机床数据”功能改变。这样一来，在过渡至轴模式前可根据需要在零件程序修改要换入的齿轮档。

17.4.10 在 DryRun、程序测试和 SERUPRO 下抑制齿轮档切换

功能

在空运行进给（DryRun）、程序测试和 SERUPRO 模式下通常不需要进行齿轮档切换。因此可为这些模式抑制该功能。此配置通过以下机床数据的位 0 ... 2 进行：

MD35035 $MA_SPIND_FUNCTION_MASK

空运行进给（DryRun）	
位 0 = 0	DryRun 功能激活时，为含 M40、M41 至 M45 的零件程序段、FC 18 编程和同步动作启用齿轮档切换。
位 0 = 1	DryRun 功能激活时，为含 M40、M41 至 M45 的零件程序段、FC 18 编程和同步动作抑制齿轮档切换。
程序测试和 SERUPRO	
位 1 = 0	程序测试/SERUPRO 功能激活时，为含 M40、M41 至 M45 的零件程序段、FC 18 编程和同步动作启用齿轮档切换。
位 1 = 1	程序测试/SERUPRO 功能激活时，为含 M40、M41 至 M45 的零件程序段、FC 18 编程和同步动作抑制齿轮档切换。
DryRun、程序测试和 SERUPRO	
位 2 = 0	在 DryRun、程序测试和 SERUPRO 功能取消后，在 REPOS 中不针对编写的齿轮档追加执行齿轮档切换。
位 2 = 1	在 DryRun、程序测试和 SERUPRO 功能取消后，在条件允许时针对编写的齿轮档追加执行齿轮档切换。

过程

齿轮档切换被抑制时，编写的主轴转速同样会由插补器限制在生效的齿轮档所允许的范围内。

此类限制中生成的 NC/PLC 接口信号 DB31, ... DBX83.2（设定转速已提升）和 DB31, ... DBX83.1（设定转速已限制）将被抑制。

DryRun 和空运行进给中不必通过 PLC 程序进行监控。

抑制齿轮档切换时，新的设定齿轮档（DB31,...DBX82.0-82.2）不会输出至 PLC。

齿轮档切换请求 DB31, ... DBX82.3（切换齿轮档）也会被抑制。

这样可防止 PLC 程序处理齿轮档切换信息。

确定最后生效的齿轮档

系统变量 $P_GEAR 提供的是通过零件程序编写的齿轮档（可能未输出至 PLC）。

系统变量 $AC_SGEAR 提供的是零件程序、同步动作中最后生效的齿轮档，并可在操作界面上读取。

取消时的特性

DryRun 功能可在程序段运行期间取消。该功能取消后，必须确定零件程序所请求的齿轮档并将其换入。

只有在启用正确的齿轮档后才能确保剩余零件程序的顺利执行。主轴处于转速控制运行时，追加的齿轮档切换将在 DryRun 取消后启动的系统 REPOS 中执行。此时会与 PLC 进行完整的齿轮档切换信号对话，并换入最后编写的齿轮档。

在 REPOS 期间，若零件程序中编写的齿轮档与 NC/PLC 接口提供的当前齿轮档一致，则不会执行齿轮档切换。

上述情形同样适用于 SERUPRO 功能。

程序段搜索 SERUPRO 的更多相关信息请参见：
文档：
功能手册之基本功能分册；章节“BAG，通道，程序运行，复位特性（K1）”

前提条件

齿轮档切换被抑制时，输出的主轴转速被限制在为当前齿轮档设定的转速范围内。

对 REPOS 期间追加执行齿轮档切换的限制：

- 主轴在取消程序段或目标程序段中作为命令主轴（同步动作）或 PLC 主轴（FC18)时，将不会追加执行齿轮档切换。
- 由于主轴处于定位模式或轴模式，或由于耦合生效导致无法换入齿轮档时，将触发报警 22011“通道%1 程序段%3 主轴 2% 无法切换至编写的齿轮档”。

示例

空运行进给（DryRun）下的齿轮档切换

```
; 第 1 个用于初始状态
N00 M3 S1000 M41                    ; 换入齿轮档 1
```

```
M0                                          ; 零件程序停止

; PI 服务：激活空运行进给（DryRun）
                                            ; （配置）
N10 M42                                     ; 请求齿轮档 2，齿轮档切换不执行
N11 G0 X0 Y0 Z0                             ; 轴定位
N12 M0                                      ; 零件程序停止

; PI 服务：取消空运行进给（DryRun）
                                            ; 执行 REORG 和 REPOS
                                            ; 现在换入齿轮档 2，执行齿轮档切换
N20 G1 Z100 F1000
...
N99 M30                                     ; 零件程序结束
```

17.5 其他主轴功能调整设置

通过机床数据可激活下列主轴功能：

MD35035 $MA_SPIND_FUNCTION_MASK，<位> = <值>

<位>	<值>	含义
0 ... 2	1	激活：空运行进给（DryRun）、程序测试和 SERUPRO 模式下的齿轮档切换特性 参见“在 DryRun、程序测试和 SERUPRO 下抑制齿轮档切换 (页 1479)”。
4	1	将 NC 程序中编写的主轴转速 S... 接收至 SD43200 $SA_SPIND_S。这也适用于通过 PLC 用户程序借助 FC18 和同步动作进行的转速设定。 参见“功能 (页 1492)”。
5	0	在 JOG 运行模式下运行主轴时，采用当前有效的转速设定。 参见 SD41200 $SN_JOGSPIND_SET_VELO
	1	在 JOG 运行模式下运行主轴时，SD43200 $SA_SPIND_S 中设置的转速生效
8	1	将 NC 程序中编写的切削速度 S... 接收至 SD43202 $SA_SPIND_CONSTCUT_S。 这也适用于通过 PLC 用户程序借助 FC18 进行的切削速度设定。 参见“功能 (页 1492)”。

<位>	<值>	含义
10	0	SD 43206 $SA_SPIND_SPEED_TYPE 不因 NC 程序或通道设置而改变。
	1	Ist die Spindel Masterspindel wird der Vorschubtyp (aktiver Wert der 15.G-Gruppe) nach SD43206 $SA_SPIND_SPEED_TYPE übernommen. 若主轴不是主主轴，则 SD43206 $SA_SPIND_SPEED_TYPE 保持不变。 参见“功能 (页 1492)”。
12	零脉冲搜索时主轴倍率的有效性，M19、SPOS、SPOSA = 0：	
	0	主轴倍率**不**生效。
	1	主轴倍率生效。
19	与隐性辅助功能输出 M19 相关的特性：	
	0	**并且** MD20850 $MC_SPOS_TO_VDI == 0： 在 SPOS 和 SPOSA 中不将辅助功能 M19 输出至 PLC。
	1	在 SPOS 和 SPOSA 中将辅助功能 M19 输出至 PLC。 地址扩展对应主轴编号。 参见“定位模式 (页 1408)”
20	与隐性辅助功能输出 M70 相关的特性：	
	0	不将隐性辅助功能 M70 输出至 PLC。 **提示**：编写的辅助功能 M70 始终会输出至 PLC。
	1	在从主轴模式过渡至轴模式时，将辅助功能 M70 输出至 PLC。 地址扩展对应主轴编号。 参见“隐性过渡至进给轴模式 (页 1428)”。
22	在刚性攻丝（**G331**、**G332**）中用于将主轴旋转方向反转的轴专用 NC/PLC 接口信号 **DB31, ... DBX17.6**（M3/M4 反向）的**关闭**。	
	0	接口信号 DB31, ... DBX17.6 **生效**。 **提示** • **默认情形**：在 NC 程序启动时确定该接口信号的值。该接口信号随后的变化在 NC 程序内无影响。 • **特殊情形**：若在 NC 程序执行期间也需要将该接口信号的变化考虑在内，则须以针对具体用户的方式实现。参见章节“特殊情形：在 NC 程序中通过 NC/PLC 接口信号将旋转方向反转 (页 1436)”。
	1	接口信号 DB31, ... DBX17.6 **不生效**。

17.6 可切换主轴

功能

借助“可切换的主轴”功能，能够就通道主轴写入可应用在不同通道中的通用 NC 程序。其中通过主轴编号转换器将 NC 程序中使用的逻辑主轴编号转换成物理主轴编号。因此，在 NC 程序中编写的主轴编号相同的情况下，在不同通道中运行不同的物理主轴（机床轴）。

这样一来，通过“跨通道取轴”功能换入或交还的物理主轴便不必在零件程序中显性指定。

机床数据

物理主轴编号

每个主轴必须借助明确的可配置的编号（物理主轴编号）被指定给机床轴：

MD35000 $MA_SPIND_ASSIGN_TO_MACHAX[<机床轴>] = <物理主轴编号>

激活主轴编号转换器

主轴编号转换器通过以下机床数据针对具体通道激活：

MD20092 $MC_SPIND_ASSIGN_TAB_ENABLE = TRUE

设定数据

主轴编号转换器

为设置通道专用的主轴编号转换器，为每个逻辑主轴编号或编写的地址扩展（借此在 NC 程序中响应主轴（通道主轴））指定一个物理主轴编号（机床轴）。

SD42800 $SC_SPIND_ASSIGN_TAB[<逻辑主轴编号>] = <物理主轴编号>

更多信息：

- 可自 SD42800 $SC_SPIND_ASSIGN_TAB[**0**] 读取当前主主轴的逻辑主轴编号。该数据不允许写入。在编写 SETMS 时，该数据会更新。
  对于未使用的主轴必须在 SD42800 中输入 0 值。
- 在辅助功能输出中，系统将物理主轴编号作为地址扩展输出。
- 主轴编号转换器适用于 NC 程序中的主轴编程以及同步动作。

- 主轴编号转换器不适用于通过 NC/PLC 接口使用功能块 FC18 进行的主轴设定，因为在此情况下直接对物理主轴进行定址。
- 主轴转换涉及的系统变量有：$P_S、$P_SDIR、$P_SMODE、$P_GWPS、$AC_SDIR、§AC_SMODE、$AC_MSNUM、$AA_S

边界条件

- 可切换的通道主轴不能替代“跨通道取轴”功能。
- 可通过 NC 程序、PLC 用户程序或 PI 服务写入设定数据（SD42800 $SC_SPIND_ASSIGN_TAB）来修改主轴转换。该修改立即生效。
- 在主轴编号转换器中只可使用指定给该通道的物理主轴（MD20070 $MC_AXCONF_MACHAX_USED）。
  若在主轴编号转换器中指定在另一通道中生效的物理主轴，则根据 MD30552 $MA_AUTO_GET_TYPE 中的设置，或是为该通道请求物理主轴，或是显示报警 16105“指定的主轴不存在”。
- 若通过 PLC 或 HMI 写入 SD42800 $SC_SPIND_ASSIGN_TAB[<n>]，那么对应通道必须处于复位状态，或者必须确保待切换主轴未在运行中的零件程序中使用。
  可以通过预处理停止 STOPRE 来实现同步特性。
- 可以将一个逻辑主轴转换成多个物理主轴。但是在操作界面上显示逻辑主轴时会依据转换表产生多重含义。

示例

指定主轴编号和机床轴：

```
MD35000 $MA_SPIND_ASSIGN_TO_MACHAX [AX4] = 1
MD35000 $MA_SPIND_ASSIGN_TO_MACHAX[ AX5 ] = 2
MD35000 $MA_SPIND_ASSIGN_TO_MACHAX [AX6] = 3
MD35000 $MA_SPIND_ASSIGN_TO_MACHAX [AX7] = 5
```

在通道中接收机床轴：

```
MD20070 $MC_AXCONF_MACHAX_USED[0] = 4
MD20070 $MC_AXCONF_MACHAX_USED[1] = 5
MD20070 $MC_AXCONF_MACHAX_USED[2] = 6
MD20070 $MC_AXCONF_MACHAX_USED[3] = 7
```

定义主主轴：

```
MD20090 $MC_SPIND_DEF_MASTER_SPIND = 1
```

主轴编号转换器

```
MD20092                              ; 激活主轴编号转换器
$MC_SPIND_ASSIGN_TAB_ENABLE=1
SD42800                              ; 主主轴如配置
$SC_SPIND_ASSIGN_TAB[0]=1
SD42800                              ; 表的初始设置
$SC_SPIND_ASSIGN_TAB[1]=1
SD42800
$SC_SPIND_ASSIGN_TAB[2]=2
SD42800
$SC_SPIND_ASSIGN_TAB[3]=3
SD42800                              ; 逻辑主轴未指定
$SC_SPIND_ASSIGN_TAB[4]=0
; 地址扩展为 1 输出：M1=3 S1=1000
; 主轴 1（物理主主轴）旋转
M3 S1000
...
; 逻辑主轴 1 被指定给物理主轴 5
SD42800
$SC_SPIND_ASSIGN_TAB[1]=5

; 逻辑主轴 2 被指定给物理主轴 3
SD42800
$SC_SPIND_ASSIGN_TAB[2]=3
; 注意：die physikalische Spindel 3 ist jetzt zweimal vergeben.
; 编写逻辑主轴 2 和 3 时
; 始终会对物理主轴 3 进行响应。
; 在机床初始画面中两根主轴均旋转。

SETMS(2)                             ; SD42800 $SC_SPIND_ASSIGN_TAB[0] = 2
; 主主轴 = 地址扩展 = 2，输出未转换的主轴编号 M3 = 5
; 配置编号“3”的物理主轴停止。
M5
GET(S4)                              ; Alarm 16105:logische Spindel 4
                                     ; 无法转换
RELEASE (S1)                         ; Kanalspindel 1 = physik.Spindel 5
                                     ; 被使能。
```

17.7 编程

17.7.1 在零件程序中编程

编程指令

指令	说明	
SETMS：	主主轴为以下机床数据中设定的主轴： MD20090 $MC_SPIND_DEF_MASTER_SPIND（通道中主主轴的缺省设置）	
SETMS(<n>)：	编号 **<n>** 的主轴为主主轴 （可与缺省设置不同： MD20090 $MC_SPIND_DEF_MASTER_SPIND） 必须为以下功能定义主主轴：	
	• G95：	旋转进给率
	• G96 S.../G961 S...：	恒定切削速度，单位 m/min 或 ft/min
	• G97/G971：	取消 G96/G961，冻结最后的主轴转速
	• G63：	带弹性卡头的攻丝
	• G33/G34/G35：	螺纹切削
	• G331/G332：	刚性攻丝
	• G4 S...：	暂停时间，单位：主轴转数
	• 未指定主轴编号的 M3、M4、M5、S、SVC、SPOS、M19、SPOSA、M40、M41 至 M45 和 WAITS 编程	
	可跨越复位/零件程序结束和零件程序开始保留当前的主主轴设置。 这通过以下机床数据设置： • MD20110 $MC_RESET_MODE_MASK • MD20112 $MC_START_MODE_MASK	
M3： M<n>=3：	主主轴顺时针方向旋转 编号为 **<n>** 的主轴顺时针方向旋转	
M4： M<n>=4：	主主轴逆时针方向旋转 编号为 **<n>** 的主轴逆时针方向旋转	
M5： M<n>=5：	主主轴无定向停止 编号为 **<n>** 的主轴无定向停止	

指令	说明
S...: S<n>=...:	主主轴的转速，单位 rpm 编号为 <n> 的主轴的转速，单位 rpm
SVC=...: SVC[<n>]=...:	主主轴切削速度，单位 m/min 或 ft/min 编号为 <n> 的主轴的切削速度，单位 m/min 或 ft/min
SPOS=...: SPOS[<n>]=...:	主主轴定位 编号为 <n> 的主轴定位 主轴到达位置以后才执行程序段切换。
SPOSA=...: SPOSA[<n>]=...:	主主轴定位 编号为 <n> 的主轴定位 立即执行程序段切换。 主轴独立于之后的零件程序执行进行定位，直至到达位置。
SPOS=DC(...): SPOS[<n>]=DC(...): SPOSA=DC(...): SPOSA[<n>]=DC(...):	从运动状态开始定位时，运动方向保持不变，并逼近位置。 从静止状态开始定位时，沿最短路径逼近位置。
SPOS=ACN(...): SPOS[<n>]=ACN(...): SPOSA=ACN(...): SPOSA[<n>]=ACN(...):	始终采用负方向逼近位置。 必要时在定位前反转运动方向。
SPOS=ACP(...): SPOS[<n>]=ACP(...): SPOSA=ACP(...): SPOSA[<n>]=ACP(...):	始终采用正方向逼近位置。 必要时在定位前反转运动方向。
SPOS=IC(...): SPOS[<n>]=IC(...): SPOSA=IC(...): SPOSA[<n>]=IC(...):	设定运行行程。 运行方向取决于行程的符号。 如果主轴已在运行中，必要时会反转运行方向，从而能够以编写的方向运行。 在运行期间越过零脉冲时，若尚未执行回参考点，或通过接口信号请求回参考点，那么主轴将自动与零脉冲同步。
M19: M[<n>]=19:	将主主轴定位至 SD43240 中的位置。 将编号为 <n> 的主轴定位至 SD43240 中的位置。 主轴到达位置以后才执行程序段切换。

指令	说明
M70： M<n>=70：	停止主轴并激活位置闭环控制，选择参数组零，激活轴模式 针对主主轴 针对编号为 **<n>** 的主轴
SPCON： SPCON(<n>)： SPCON(<n>,<m>)：	主轴位置闭环控制 ON 针对主主轴 针对编号为 **<n>** 的主轴 针对编号为 **<n>** 和 **<m>** 的主轴
PCOF： SPCOF(<n>)： SPCOF(<n>,<m>)：	主轴位置闭环控制 OFF，激活转速控制运行 针对主主轴 针对编号为 **<n>** 的主轴 针对编号为 **<n>** 和 **<m>** 的主轴
FPRAON(S<n>)：	主轴 **<n>** 旋转进给率 ON，从主主轴推导
FPRAON(S<n>,S<m>)：	主轴 **<n>** 旋转进给率 ON，从主轴 **<m>** 推导 旋转进给值必须通过 FA[S<m>] 设定。
FPRAOF(S<n>)：	主轴 **<n>** 旋转进给率 OFF
C30 G90 G1 F3600	回转轴 C（轴模式下的主轴）以 3600 degrees/min = 10 rev/min 的速度运行至 30 度位置
G25 S...： G25 S<n>：	可编程的主轴转速下限 针对主主轴 针对编号为 **<n>** 的主轴
G26 S...： G26 S<n>：	可编程的主轴转速上限 针对主主轴 针对编号为 **<n>** 的主轴
LIMS=...： LIMS[<n>]=...：	G96、G961、G97 下可编程的主轴转速限值 针对主主轴 针对编号为 **<n>** 的主轴

指令	说明
VELOLIM[<Spindel>]=...:	可编程的、针对配置的齿轮档的转速上限 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 是同时对轴模式和主轴模式生效（位 6 = 1），还是需要为两种模式分别编写（位 6 = 0）。 若分别编写，在编程中通过名称选择模式： ● 主轴名称 S<n> 用于主轴模式 ● 轴名称，例如“C”，用于轴模式 补偿值的基准： ● 轴模式下的主轴（MD30455 位 6 = 0）： 以配置的最大轴速度为基准（MD32000 $MA_MAX_AX_VELO）。 ● 主轴模式或轴模式下的主轴（MD30455 位 6 = 1）： 以生效的齿轮档的最大转速为基准 （MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[<n>]） VELOLIM 编程的更多详细信息参见： **文档：** 编程手册之工作准备分册
WAITS:	对主主轴的同步指令 执行以下程序段时系统将会等待，直至之前的 NC 程序段中通过 SPOSA 编写的主轴到达其位置（精准停）。 M5 后 WAITS： 等待，直至主轴停止。 M3/M4 后 WAITS： 等待，直至主轴达到设定转速。
WAITS(<n>): WAITS(<n>,<m>):	对编号为 **<n>** 的主轴的同步指令 对编号为 **<n>** 和 **<m>** 的主轴的同步指令
FA[S<n>]:	为主轴 **<n>** 以 [degrees/min] 为单位编写定位速度（轴进给） FA[S<n>]=0 时，为以下机床数据配置的值重新生效： MD35300 $MA_SPIND_POSCTRL_VELO
OVRA[S<n>]:	为主轴 **<n>** 以 [%] 编写轴倍率值
ACC[S<n>]:	为主轴 **<n>** 以 [%] 编写轴加速度

指令	说明
SPI(<n>)：	通过 SPI(<n>) 根据以下机床数据： MD35000 $MA_SPIND_ASSIGN_TO_MACHAX[ ] 将一个主轴编号转换为 AXIS 数据类型。 SPI 用于通过主轴主轴编写轴功能。 以下指令可与 SPI 配合使用： ● 框架指令： – CTRANS() – CFINE() – CMIRROR () – CSCALE () ● 副主轴的速度和加速度值： – FA[SPI(<n>)] – ACC[SPI(<n>)] – OVRA[SPI(<n>)] ● 系统变量： – $P_PFRAME[SPI(<n>),TR]=<值> – $P_PFRAME= CTRANS(X,<轴值>,Y,<轴值>,SPI(<n>),<轴值>) – $P_PFRAME= CSCALE(X,<标度>,Y,<标度>,SPI(<n>),<标度>) – $P_PFRAME=CMIRROR(S<n>,Y,Z) – $P_UBFR=CTRANS((A,10) : CFINE (19,0.1) SPI 编程的更多详细信息参见： **文档：** 编程手册之工作准备分册
M40： M<n>=40：	为主主轴自动选择齿轮档 为编号为 **<n>** 的主轴自动选择齿轮档
M41 至 M45： M<n>=41 至 M<n>=45：	为主主轴选择齿轮档 1 至 5 为编号为 **<n>** 的主轴选择齿轮档 1 至 5

说明

通道中配置了主轴时，M 功能 M3、M4、M5 和 M70 不在 DB21, ... DBB194 和 DBB202 中输出。 这些 M 功能作为扩展 M 功能在 DB21, ... DBB68 ff. 和相应的轴数据块 DB31, ... DBB86 ff. 中提供。

文档

对主轴编程的更多详细说明参见：

- 编程手册之基本原理分册

17.7.2 通过同步动作编程

M 功能 M40 至 M45 也可在同步动作中编程。

此时须注意：

- 零件程序段中编写的 M40 ... M45 不会对同步动作的自动齿轮档切换的当前状态产生影响，反之亦然。
- 使用 S 值和 M40 编写的自动齿轮档切换分别对同步动作和零件程序生效。
- 上电后 M40 取消。
  通过同步动作设定 S 值时不会进行齿轮档调整。
- 通过同步动作编写的 M40 对同步动作始终生效（模态生效），复位后仍保留。
- 依据零件程序中的编程，M41 ... M45 选择的是第一个到第五个齿轮档。
  为执行功能需进行跨通道取轴。
  执行完齿轮档切换后，主轴处于中性状态（与 M3、M4、M5 编程的特性相似）。

文档

通过同步动作编程主轴及主轴运动的更多详细信息参见：

- 编程手册之工作准备分册
- 功能手册之同步动作分册

17.7.3 通过 PLC 使用 FC18 编写主轴控制 - 仅针对 840D sl

通过 PLC 使用 FC18 设定旋转方向和转速时，NC 能确定与转速相匹配的齿轮档并将其换入。这相当于零件程序编程中的 M40 功能。

为了激活齿轮档切换，必须在使用 PLC 用户程序调用 FC18 时设置对应的启动代码。

文档

通过 PLC 使用 FC18 编写主轴控制的更多详细信息参见：

- 功能手册之基本功能分册；PLC 基本程序（P3）

17.7.4 通过 NC/PLC 接口信号编程

17.7.4.1 功能

说明

此功能基于 SINUMERIK Operate。

针对主轴，可通过 PLC 用户程序借助任务接口 DB31, … DBX30.0 - 4 执行以下任务：

- 主轴停止
- 主轴启动，顺时针旋转
- 主轴启动，逆时针旋转
- 选择齿轮档
- 主轴启动，定位

任务识别

在 NC 侧基于相应接口信号的上升**脉冲沿切换**（0 → 1）识别主轴任务。

前提条件

在主轴任务的时间点上，在通道状态、程序状态和主轴状态方面必须满足下列前提条件。

- 通道状态：
  - DB21, ... DBX35.6 == 1（通道状态“中断”）**或者**
  - DB21, ... DBX35.7 == 1（通道状态“复位”）
- 程序状态：
  - DB21, ... DBX35.3 = 1（程序状态“中断”）**或者**
  - DB21, ... DBX35.4 = 1（程序状态“终止”）
- 主轴状态
  - （“通道轴”**或者**“中性轴”）**并且无**
  - （通过 FC18 **或者**同步动作进行的定位运动）

说明

故障情形下的特性

在故障情形下，即在无效的状态下将任务信号置位时，系统会忽略相应任务并且**不反馈**。

通道分配

主轴任务在以下通道中执行：任务时间点上该主轴被分配给的通道。

可通过以下 NC/PLC 接口信号确定该主轴被分配给的通道。

DB31, ... DBX68.0 - 3（NC 轴/主轴的通道分配）

运行方式切换

在主轴任务范围内进行的设定在运行方式切换后仍保留。

程序启动

在主轴被分配给的通道中启动 NC 程序时，当前主轴设定保留。可通过 NC 程序中的指令或通过同步动作修改主轴设定。

参见

M40：转速超出配置的开关阈值时的自动齿轮档选择 (页 1509)

G2：速度、设定值/实际值系统、闭环控制 (页 363)

17.7.4.2 调试：机床数据

将主轴设定自动接收至设定数据

通过以下机床数据设置将主轴设定从 NC 程序、同步动作和 PLC 的主轴控制（FC18）接收至相应的设定数据：
MD35035 $MA_SPIND_FUNCTION_MASK，位 x = <值>

位	值	含义
4	1	将通过 NC 程序、同步动作或 FC18 进行的**转速设定**接收至以下主轴专用设定数据： **SD43200** $SA_SPIND_S[<主轴>] **提示** 并非转速编程的 S 值不会被接收。例如： • 恒定切削速度（G96/G961）下的 S 值 • 旋转相关暂停时间（G4）下的 S 值
	0	不将转速设定接收至该设定数据。

位	值	含义
8	1	将“恒定切削速度”功能（G96、G961、G962）生效时通过 NC 程序、同步动作或 FC18 进行的**切削速度**设定接收至以下主轴专用设定数据： **SD43202** $SA_SPIND_CONSTCUT_S[<主轴>] **提示** 并非切削速度编程的 S 值不会被接收。例如： ● 旋转相关暂停时间（G4）下的 S 值
	0	**不**将切削速度设定接收至该设定数据。
10	1	针对**主主轴**，将**进给类型**接收至以下机床数据： **SD43206** $SA_SPIND_SPEED_TYPE[<主轴>]
	0	**不**将针对主主轴的进给类型设定接收至该设定数据。

借助接收至这些设定数据的主轴设定，在主轴任务范围内运行主轴。

17.7.4.3 调试：NC/PLC 接口信号

任务接口

- DB31, … DBX30.0（主轴停止）
- DB31, … DBX30.1（主轴启动，顺时针旋转）
- DB31, … DBX30.2（主轴启动，逆时针旋转）
- DB31, … DBX30.3（选择齿轮档）
- DB31, … DBX30.4（主轴启动，定位）

相关状态信号

- DB21, ... DBX35.3（程序状态“中断”）
- DB21, ... DBX35.4（程序状态“终止”）
- DB21, ... DBX35.7（通道状态“复位”）
- DB21, ... DBX35.6（通道状态“中断”）
- DB31, ... DBX68.0 - 3（NC 轴/主轴的通道分配）
- DB31, ... DBX 84.0（恒定切削速度）

文档：

对 NC/PLC 接口信号的详细说明请参见参数手册，NC 变量和接口信号。

17.7.4.4 转速设定（SD43200）

在通过主轴任务启动主轴时，以下设定数据中当前记录的主轴专用值作为主轴转速生效：

SD43200 $SA_SPIND_S[<主轴>]（通过 PLC 任务接口启动主轴时的转速）

写入新的转速值

在以下情形下，系统将新的转速值写入该设定数据：

- 通过 NC 程序、同步动作或 FC18 设定转速
  边界条件：
  - MD35035 $MA_SPIND_FUNCTION_MASK (页 1493)，位 4 == 1
  - DB31, ... DBX84.0 == 0（恒定切削速度）
- 在 NC 程序或同步动作中写入该设定数据
- 通过 HMI（OPI）写入该设定数据 @@@

边界条件

齿轮档切换

当设定的转速处于当前齿轮档的转速范围外时，将不会触发齿轮档切换。

特例参见章节“M40：转速超出配置的开关阈值时的自动齿轮档选择 (页 1509)”。

17.7.4.5 恒定切削速度的设定（SD43202）

在主轴转速类型“恒定切削速度”生效的情况下通过主轴任务启动主主轴时，以下设定数据中当前记录的主轴专用值作为恒定切削速度生效：

SD43202 $SA_SPIND_CONSTCUT_S[<主轴>]（通过 PLC 任务接口启动主轴时的恒定切削速度）

写入新的切削速度值

在以下情形下，系统将主主轴的新的恒定切削速度值写入该主轴专用设定数据：

- 通过 NC 程序、同步动作或 FC18 设定恒定切削速度
  边界条件：
  - MD35035 $MA_SPIND_FUNCTION_MASK (页 1493)，位 8 == 1
  - DB31, ... DBX84.0 == 1（恒定切削速度生效）
- 在 NC 程序或同步动作中写入该设定数据
- 通过 HMI（OPI）写入该设定数据 @@@

单位制

在写入设定数据时，根据下列边界条件对值进行解释：

- 通过 NC 程序或同步动作写入：
  - G700 aktiv:feet/min
  - G710 aktiv:米/分钟
  - G70、G71 生效：取决于 MD10240 $MN_SCALING_SYSTEM_IS_METRIC 中的设置
- 通过 SINUMERIK Operate 写入：
  - 取决于 MD10240 $MN_SCALING_SYSTEM_IS_METRIC 中的设置
- 通过 FC18 写入：
  - 功能编号 10：“恒定切削速度（m/min）”
  - 功能编号 11：“恒定切削速度（feet/min）”

通过系统变量或 OPI 变量读取

可在 NC 程序、同步动作中，或者通过 HMI 借助以下系统数据读取当前为恒定切削速度设定的值：

系统变量	OPI 变量	含义
$P_CONSTCUT_S	---	最后编写的切削速度
$AC_CONSTCUT_S	acConstCutS	当前的恒定切削速度

17.7.4.6 针对主主轴的主轴转速类型设定（SD43206）

零件程序、FC18 或同步动作中的**主主轴**转速类型设定会从所有常见数据源写入以下设定数据：

SD43206 $SA_SPIND_SPEED_TYPE（通过 PLC 接口启动主轴时的主轴转速类型）

Der Wertebereich und die Funktionalität entsprechen der 15.G-Gruppe (Vorschubtyp).

允许的值为 G 功能值：93、94、95、96、961、97 和 971。

根据设置，DB31, … DBX30.1/2（主轴启动，顺时针/逆时针旋转）会启用 SD43200 $SA_SPIND_S 中的转速或 SD43202 $SA_SPIND_CONSTCUT_S 中的切削速度。

93、 94、95、97 和 971：	主主轴以 SD43200 中的转速启动。
96 和 961：	主主轴的转速由设定的切削速度（SD43202）和平面轴半径得出。

17.7.5 外部编程（PLC、HMI）

SD43300 和 SD42600

借助轴专用设定数据：
SD43300 $SA_ASSIGN_FEED_PER_REV_SOURCE（主轴的旋转进给率）
可在 JOG 运行方式下通过以下通道专用设定数据：
SD42600 $SC_JOG_FEED_PER_REV_SOURCE（JOG 下旋转进给控制）
从外部选择采用旋转进给的运行。

通过该设定数据可进行以下设置：

>0:	用于推导旋转进给率的回转轴/主轴的机床轴编号。
-1:	旋转进给率从轴/主轴所在通道的主主轴推导。
0:	功能取消。

FPRAON (S2)

主轴 S2 启用旋转进给率（ON），从主主轴推导

FPRAON (S2, A)

主轴 S2 启用旋转进给率（ON），从轴 A 推导
旋转进给值必须以 `FA[Sn]` 设定。

FPRAOF (S2)

主轴 S2 不启用旋转进给率（OFF）。

SPI（n）

也可编写 `SPI(n)` 来代替 `SPI(Sn)`。

17.8 主轴监控

17.8.1 允许的转速范围

决定主轴转速范围的因素有：参数设置或编写的转速限值，以及生效的轴功能（G94、G95、G96、G961、G97、G971、G33、G34、G35、G331、G332 等）。

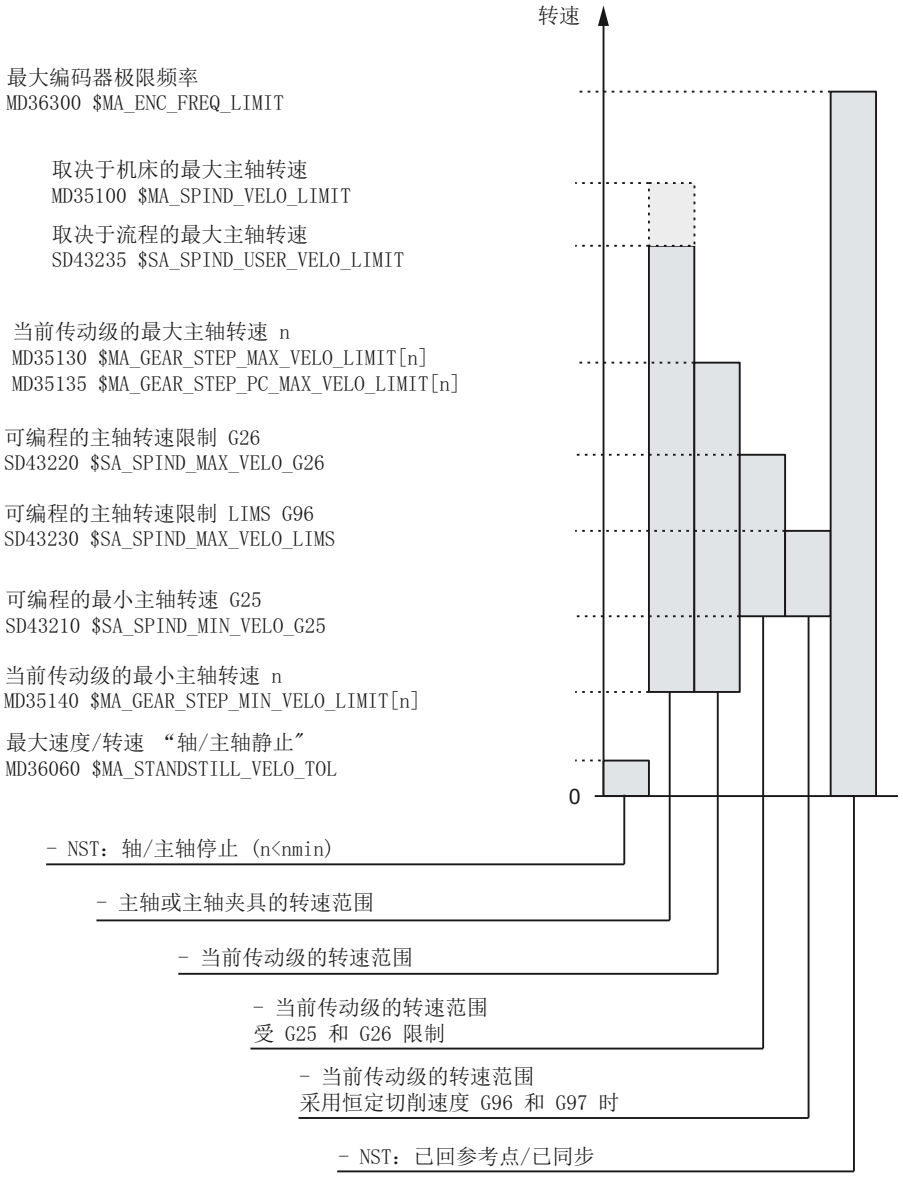


图 17-8 主轴监控范围/转速范围

17.8.2 进给轴/主轴停止

仅当主轴停止时，才能使用换刀、打开机床门、使能轨迹进给等功能。

功能

当不再生成设定值，且主轴转速实际值低于为“进给轴/主轴停止”配置的阈值时，便达到“进给轴/主轴停止”状态。

MD36060 $MA_STANDSTILL_VELO_TOL（“进给轴/主轴停止”最大速度/转速）

主轴进入停止状态时，以下 NC/PLC 接口信号置位：

DB31, ... DBX61.4（进给轴/主轴停止）

有效性

主轴静止状态监控在所有主轴运行方式以及轴模式下生效。

禁止轨迹进给

主轴在控制模式下停止时（M5），若以下机床数据置位，那么轨迹进给会被禁止：

MD35510 $MA_SPIND_STOPPED_AT_IPO_START（主轴停止时进给率使能）

主轴进入停止状态时，轨迹进给重新使能。

17.8.3 主轴位于设定区域内

功能

主轴监控“主轴处于设定区域”用于检查：

- 是否达到编写的主轴转速
- 主轴是否停止：
  DB31, ... DBX61.4（进给轴/主轴停止）= 1
- 主轴是否仍处于加速或制动阶段

在控制模式下，系统会将设定转速和实际转速进行比较。若实际转速的偏离值大于通过机床数据设定的主轴转速公差（见下文），则：

- 将以下 NC/PLC 接口信号置“0”：
  DB31, ... DBX83.5（主轴位于设定区域内）= 0
- 不使能下一个加工程序段（取决于机床数据 MD35500 $MA_SPIND_ON_SPEED_AT_IPO_START 中的设置，参见“进给轴/主轴停止 (页 1499)”）。

主轴设定转速

主轴设定转速由编写的转速接合主轴倍率和生效的限制得出。

编写的转速被限制为上限值或提升至下限值的情形会通过 DB31, ... DBX83.1（设定转速已限制）或 DB31, ... DBX83.2（设定转速已提升）显示（另见“齿轮档的最小/最大转速 (页 1501)”）。设定转速到达公差范围的情形则不会被阻止。

设定转速的公差范围

设定转速的公差范围通过主轴转速公差系数定义：

MD35150 $MA_SPIND_DES_VELO_TOL

示例：

MD35150 $MA_SPIND_DES_VELO_TOL = 0.1

⇒ 允许主轴实际转速和设定转速有 ±10% 的偏差。

若主轴实际转速处于公差范围内，则以下 NC/PLC 接口信号会置“1”：

DB31, ... DBX83.5（主轴位于设定区域内）= 1

特殊情况：

若主轴转速公差被设为“0”，那么 DB31, ... DBX83.5（主轴位于设定区域内）将永久置“1”，此时将不会进行轨迹控制。

转速变化

系统仅会在运行程序段开始时以及编写了转速变化时进行轨迹控制。例如，过载导致超出转速公差范围时，轨迹运动不会自动停止。

17.8.4 齿轮档的最小/最大转速

最小转速

主轴齿轮档的最小转速在以下机床数据中配置：

MD35140 $MA_GEAR_STEP_MIN_VELO_LIMIT[<n>]

基于倍率生成的转速设定值不会低于该最小转速。

若编写了小于最小转速的 S 值，那么该设定转速会被提升至最小转速，且以下 NC/PLC 接口信号置位：

DB31, ... DBX83.2（设定转速已提升）

齿轮档的最小转速仅在转速运行中生效，且只有以下情形下转速才会低于该值：

- 主轴倍率 0%
- M5
- S0
- DB31, ... DBX4.3（主轴停止）
- DB31, ... DBX2.1（控制器使能）取消
- DB21, ... DBX7.7（复位）
- DB31, ... DBX2.2（删除剩余行程/主轴复位）
- DB31, ... DBX18.5（往复转速）
- DB21, ... DBX7.4（轴和主轴 NC 停止）
- DB31, ... DBX1.3（轴/主轴禁止）
- DB31, ... DBX16.7（删除 S 值）

最大转速

主轴齿轮档的最大转速在以下机床数据中配置：

MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[<n>]

基于倍率生成的转速设定值会被限制在该转速以下。

设定转速被限制为最大转速时，以下 NC/PLC 接口信号置位：

DB31, ... DBX83.1（设定转速已限制）

17.8.5 主轴转速限制诊断

系统变量

生效/受限的主轴参数可通过以下系统变量读取。这些系统变量只提供转速控制运行和主轴定位运行的相关数值，使用时将主轴的编号作为索引输入。

系统变量	含义
$AC_SMAXVELO[<n>]	可能的最大转速 [rev/min]
$AC_SMAXVELO_INFO[<n>]	日期标识（转速降低）
$AC_SMINVELO[<n>]	可能的最小转速 [rev/min]
$AC_SMINVELO_INFO[<n>]	日期标识（转速提升）
$AC_SMAXACC[<n>]	可能的最大主轴加速度 [rev/s^2]
$AC_SMAXACC_INFO[<n>]	日期标识（加速度减小）
$AC_SPIND_STATE[<n>]	转速控制运行中的主轴状态
<n>: 0, 1, 2, ... 最大主轴编号（0：当前通道主主轴）	

文档

有关系统变量的详细说明请见：

参数手册之系统变量

NC/PLC 接口信号

系统会通过以下 NC/PLC 接口信号显示主轴转速受到限制或提升。

- DB31, ... DBX83.1（设定转速已限制）
- DB31, ... DBX83.2（设定转速已提升）

前提条件

主轴运行方式

系统变量提供的值取决于主轴运行方式：

- 转速控制运行：
  所有系统变量均提供当前值。
- 定位模式：
  系统变量 $AC_SMAXVELO、$AC_SMAXACC 和 $AC_SPIND_STATE 提供有效值。
  系统变量 $AC_SMINVELO 和 $AC_SMINVELO_INFO 提供切换至转速控制运行时生效的数据。
- 轴模式（例如主轴参与 TRANSMIT、TRACYL 等转换功能，或作为轨迹运动的辅助轴）：
  系统变量 $AC_SPIND_STATE 也可在轴模式下使用。此外轴模式下还可使用动态数据专用的系统变量：
  $AA_VMAXM、$AA_VMAXB 和 $AA_VLFCT。

程序段搜索 SERUPRO

在程序段搜索 SERUPRO 下可通过这些系统变量获取以下控制特性：

- 系统变量 $AC_SMAXVELO / $AC_SMAXACC 提供可显示的最大速度/加速度。
- $AC_SMAXVELO_INFO 和 $AC_SMAXACC_INFO 提供值 “0”（无限制生效）。
- $AC_SMINVELO 和 $AC_SMINVELO_INFO 提供的数据和一般程序执行时相同。
- $AC_SPIND_STATE 提供的是对应 SERUPRO 功能的状态。

示例

生效/受限的主轴 1 的主轴参数被循环写入 R 参数中，在 HMI 用户界面的 “参数” > “用户变量” 操作区中显示为 R 参数。

程序代码

```
N05 IDS=1 WHENEVER TRUE DO $R10=$AC_SMAXVELO[1]
N10 IDS=2 WHENEVER TRUE DO $R11=$AC_SMAXVELO_INFO[1]
N15 IDS=3 WHENEVER TRUE DO $R12=$AC_SMINVELO[1]
N20 IDS=4 WHENEVER TRUE DO $R13=$AC_SMINVELO_INFO[1]
N25 IDS=5 WHENEVER TRUE DO $R14=$AC_SPIND_STATE[1]
```

参见

主轴位于设定区域内 (页 1499)

17.8.6 主轴转速上限

最大主轴转速：受机床条件制约的、可参数设置的限值

受机床条件制约的最大主轴转速，例如用于保护主轴卡盘或刀具，通过以下机床数据设置：

MD35100 $MA_SPIND_VELO_LIMIT（最大主轴转速）

NC 会对主轴转速进行实际值监控，即基于当前生效的齿轮档进行监控。

说明

机床制造商

如需修改机床数据 MD35100 中的最大主轴转速，建议只在主轴停止的情形下进行。 对 NC 复位后保持生效的主轴尤其需要遵循这一点（另见 MD35040 SPIND_ACTIVE_AFTER_RESET）。 否则可能会触发报警 22100。

文档

- 制造商文档：参数手册“详细机床数据说明”
- 用户文档：诊断手册

超出限值时的响应

主轴转速实际值超出所设置的主轴最大转速的幅度高于公差值时，

“主轴转速实际值” > MD35100 $MA_SPIND_VELO_LIMIT（最大主轴转速）
+ MD35150 $MA_SPIND_DES_VELO_TOL（主轴转速公差值）

NC 会进行以下响应：

- DB31, ... DBX83.0 = 1（超出转速限值）
- 显示报警 22100“超出卡盘转速”
- 对通道中的所有轴和主轴进行制动，使其停止

说明

回转轴/主轴

若主轴还短暂地作为回转轴运行，那么以下情形下会显示报警 22100“超出卡盘转速”：

“回转轴转速实际值” > MD35100 $MA_SPIND_VELO_LIMIT（最大主轴转速）
+ MD35150 $MA_SPIND_DES_VELO_TOL（主轴转速公差值）

解决办法：主轴作为回转轴运行期间，将最大主轴转速调整为最大回转轴转速

MD35100 $MA_SPIND_VELO_LIMIT = MD32000 $MA_MAX_AX_VELO（最大回转轴转速）

必须触发复位才能使此修改生效。

最大主轴转速：受过程制约的、可设置的限值

受过程制约的最大主轴转速通过以下设定数据设置：

SD43235 $SA_SPIND_USER_VELO_LIMIT（最大主轴转速）

可通过以下方式写入设定数据来修改该值：

- 在 NC 程序中编程
- 由机床操作人员通过操作界面手动修改

修改将立即生效。

对设定转速的限制

必要时，控制系统会将编写的主轴转速设定值限制在设定数据中定义的限值范围内。 之后此限制会通过以下系统变量显示：

$AC_SMAXVELO_INFO[<主轴编号>] == 21

17.8.7 最大编码器极限频率

⚠ 小心

超限

控制系统会对主轴位置实际值编码器的最大编码器极限频率进行监控（可能会超出）。机床制造商在设计主轴电机、齿轮箱、测量齿轮箱、编码器，以及对应的机床数据时，须确保不会超出主轴位置实际值编码器的最大转速（机械极限转速）。

超出最大编码器频率

若主轴在控制模式下达到超出最大编码器极限频率的转速（编写了较大的 S 值；此时不得超出编码器的最大转速），那么同步将丢失。主轴会继续旋转，但功能降低。

在下列功能中主轴转速会被降低，直至生效的测量系统重新在编码器极限频率下工作：

- 螺纹切削（G33、G34、G35）
- 刚性攻丝（G331、G332）
- 旋转进给率（G95）
- 恒定切削速度（G96、G961、G97、G971）
- SPCON（主轴的位置闭环控制运行）

超出编码器极限频率时，系统会针对相关测量系统复位以下
NC/PLC 接口信号：
DB31, ... DBX60.4（已回参考点/已同步，编码器 1）
或
DB31, ... DBX60.5（已回参考点/已同步，编码器 2）
并置位以下 NC/PLC 接口信号：
DB31, ... DBX60.2（超出编码器极限频率 1）
或
DB31, ... DBX60.3
（超出编码器极限频率 2）。

若主轴处于轴模式下，则不允许超出最大编码器极限频率。
最大速度（MD32000 $MA_MAX_AX_VELO）必须低于最大编码器极限频率，否则会触发报警 21610 并停止轴运行。

低于最大编码器极限频率

在超出了最大编码器极限频率之后，若之后又重新达到低于最大编码器极限频率的转速（编写较低的 S 值，修改主轴倍率开关等），那么主轴将自动与下一个零脉冲或下一个无接触式接近开关信号进行同步。对于之前同步丢失，且当前最大编码器极限频率高于主轴转速的位置测量编码器总是会进行新同步。

特殊性

若下列功能生效，则不允许超出最大编码器极限频率：

- 主轴定位模式、轴模式
- 螺纹切削（G33、G34、G35）

- 刚性攻丝（G331、G332，不适用于 G63）
- 旋转进给率（G95）
- 恒定切削速度（G96、G961、G97、G971）
- SPCON

17.8.8 目标点监控

目标点监控

在主轴进行定位时（定位模式下），控制系统会监控主轴（其实际位置）和编写的主轴设定位置（目标点）之间的距离。

为此可在以下机床数据中设定两个限值，作为以主轴目标位置为起点的增量行程：
MD36000 $MA_STOP_LIMIT_COARSE（粗准停限值）
和
MD36010 $MA_STOP_LIMIT_FINE（精准停限值）

在不启用这两个限值的情况下，主轴定位也能有很好的精度，这取决于连接的主轴测量编码器、间隙、齿轮比等因素。

基于参数组的准停窗口

可基于参数组配置不同的准停窗口。
这样便可在轴模式和主轴定位中启用不同的精度进行工作。 针对主轴定位，可为每个齿轮档分别配置准停窗口。

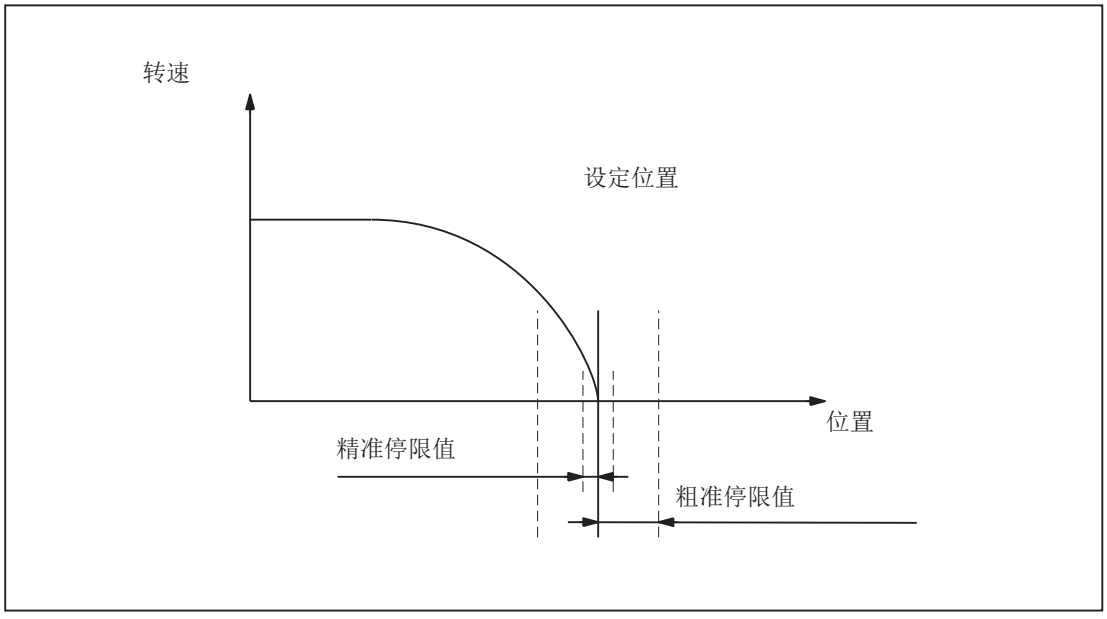


图 17-9　　主轴的准停区域

DB31, ... DBX60.7 和 DB31, ...DBX60.6（采用粗/精准停到达位置）

通过机床数据：
MD36000 $MA_STOP_LIMIT_COARSE（粗准停限值）
和
MD36010 $MA_STOP_LIMIT_FINE（精准停限值）
定义的两个限值通过 NC/PLC 接口信号：
DB31, ... DBX60.7（采用粗准停到达位置）
和
DB31, ... DBX60.6（采用精准停到达位置）
输出至 PLC。

SPOS 和 M19 下的程序段切换

在通过 SPOS 或 M19 定位主轴时，程序段切换根据时间监控
通过以下 NC/PLC 接口信号进行：
DB31, ... DBX60.6（采用精准停到达位置）。

此时程序段中编写的所有其他功能必须都已达到程序段结束标准（例如所有辅助功能均由 PLC 应答）。

使用 SPOSA 定位时，程序段切换独立于目标点监控进行。

17.8.9 M40：转速超出配置的开关阈值时的自动齿轮档选择

功能

M40 生效时，即使编写的主轴转速 S... 超出配置的开关阈值，系统仍会自动选择齿轮档。

此时分为下列情形：

- **编写的转速过高**
  编写的主轴转速大于为第 n 齿轮档配置的转速上限：
  S... > MD35110 $MA_GEAR_STEP_MAX_VELO[<n>]
  在此情形下会切换至**最高齿轮档**（根据 MD35090 $MA_NUM_GEAR_STEPS）。
- **编写的转速过低**
  编写的主轴转速小于为第一齿轮档配置的最小转速：
  S... < MD35120 $MA_GEAR_STEP_MIN_VELO[1]
  在此情形下会切换至**第一齿轮档**。
- **编写的转速 = 0**
  编写转速 0 时（S0），特性取决于第一齿轮档最小转速的配置 MD35120 $MA_GEAR_STEP_MIN_VELO[1]：
  - 若配置了 MD35120 $MA_GEAR_STEP_MIN_VELO[1] **= 0**，那么在编写 S0 时将切换至**第一齿轮档**。
  - 若配置了 MD35120 $MA_GEAR_STEP_MIN_VELO[1] **> 0**，那么在编写 S0 时将**不切换齿轮档**，最后的齿轮档将保持生效。这样便可在不触发齿轮档切换的情况下通过 S0 停止主轴（替代 M5）。

有效性

通过零件程序、同步动作或 PLC FC18 设定了主轴转速 S... 时，自动齿轮档选择（M40）下向最高齿轮档或第一齿轮档的切换便会生效。

在 G331 攻丝的零件程序中进行转速编程时，也可通过此特性实现向第二齿轮组的切换（前提条件：MD35010 $MA_GEAR_STEP_CHANGE_ENABLE，位 5 = 1）。

前提条件

使能齿轮档切换

此功能的前提条件是：通过下列机床数据对齿轮档切换进行了总体使能：

MD35010 $MA_GEAR_STEP_CHANGE_ENABLE（参数设置齿轮档切换）

MD35090 $MA_NUM_GEAR_STEPS（建立的齿轮档的数量）

示例

自动齿轮档选择 M40 是 NC 复位后的缺省设置。

零件程序：

程序代码	注释
...	
N15 S3500 M3	; S3500 大于齿轮档 2 的 MD35110。切换至齿轮档 2。切换至齿轮档 2。
...	
N50 S0 M3	; 主轴停止，S0 不请求齿轮档切换（S0 的特殊处理）。
...	
N95 S5 M3	; S5 小于齿轮档 1 的 MD35120。切换至齿轮档 1。
...	

主轴 1（AX5）的配置数据：

```
MD35010 $MA_GEAR_STEP_CHANGE_ENABLE[AX5] = 1  ; 使能齿轮档切换
MD35090 $MA_NUM_GEAR_STEPS[AX5] = 2           ; 现有齿轮档数量
MD35110 $MA_GEAR_STEP_MAX_VELO[1,AX5] = 500   ; 齿轮档 1 的开关阈值上限
MD35120 $MA_GEAR_STEP_MIN_VELO[1,AX5] = 10    ; 齿轮档 1 的开关阈值下限
MD35110 $MA_GEAR_STEP_MAX_VELO[2,AX5] = 2000  ; 齿轮档 2 的开关阈值上限
MD35120 $MA_GEAR_STEP_MIN_VELO[2,AX5] = 500   ; 齿轮档 2 的开关阈值下限
```

17.9 配备 SMI 24 的主轴（Weiss 主轴）

17.9.1 简介

为了在控制系统中处理主轴的传感器数据，到目前为止都需要将传感器连接至 I/O 模块，或通过现场总线（PROFIBUS DP 或 PROFINET IO）与 PLC 通讯。

使用配备 SMI 24 的主轴（Weiss 主轴）则可将传感器数据通过 DRIVE-CLiQ 传输至驱动，并作为驱动参数提供。 使用循环驱动报文 139 又可将传感器数据从驱动传输至控制系统。这些数据在控制系统中作为以下系统数据提供：

- 系统变量
- OPI 变量
- NC/PLC 接口信号

前提条件

- 主轴使用编码器模块 SMI 24 通过 DRIVE-CLiQ 连接至驱动。
- 为主轴配置了驱动报文 139。

说明

驱动报文 139

配备编码器模块 SMI 24 的主轴一般也可通过另一个驱动报文运行。 但是传感器数据只通过驱动报文 139 传输至控制系统。

17.9.2 传感器数据

主轴电机的传感器

主轴的传感器提供夹紧装置和电机轴角度位置的信息：

- 模拟量传感器 S1：夹紧状态
  电压值在 0 - 10 V 范围内，取决于导杆位置。
- 数字量传感器 S4：活塞末端位置
  - 0 = 活塞未到达位置
  - 1 = 活塞到达位置，即活塞未占用
- 数字量传感器 S5：电机轴的角度位置
  - 0 = 电机轴未对准
  - 1 = 电机轴位置对准（前提条件：主轴停止）

说明

配备编码器模块 SMI 24 的主轴和轴容器

配备编码器模块 SMI 24 和驱动报文 139（用于将传感器数据传输至控制系统）的主轴不允许作为轴容器的组成部分。轴容器用于通过 NCU-Link 将轴分配给多个通道。

传输传感器数据

传感器数据由编码器模块 SMI 24 在循环驱动报文 139 中作为过程数据 11 - 14 传输至控制系统。驱动报文 139 基于驱动报文 136，不过其传输的是传感器数据，而不是第 2 编码器的数据。对驱动报文 139 的详细说明参见：

文档：
SINAMICS S120/S150 参数手册；功能图，章节：PROFIdrive

系统数据：传感器数据

传感器数据可在控制系统中通过以下系统数据读取：

含义	系统变量 $VA_	NC/PLC 接口 DB31, ...	OPI 变量	驱动参数
传感器配置	MOT_SENSOR_CONF[<轴>]	DBB132, DBB133	vaMotSensorConf	r5000
夹紧状态 1)	MOT_CLAMPING_STATE[<轴>]	DBW134	vaMotClampingState	r5001
模拟传感器 S1 测量值 1) 2)	MOT_SENSOR_ANA[<轴>]	DBW136	vaMotSensorAna	r5002
数字量传感器状态	MOT_SENSOR_DIGI[<轴>]	DBB138, DBB139	vaMotSensorDigi	r5003
<轴>：机床轴名称：AX1 ... AXn 或者主轴名称：S1 ... Sm				

1) 参见章节“夹紧状态 (页 1513)”

2) 传感器 S1：0 - 10 V

模拟量实际值：10000 增量，分辨率 1 mV

示例：

SIMATIC S7 输入模块：27648 增量，分辨率 0.36 mV

转换至配备 SMI 24 主轴时的匹配系数：2.7648(10000 增量 * 1 mV) / (27648 增量 * 0.36 mV) = 1.00469393

详细系统数据说明

NC/PLC 接口信号：	参数手册，NC 变量和接口信号
系统变量：	参数手册“系统变量”
OPI 变量：	参数手册 2；变量
驱动参数：	SINAMICS S120/S150 参数手册

17.9.3 夹紧状态

传感器 S1 根据夹紧装置的位置提供一个 0 V - 10 V 的模拟电压值。该电压值通过系统数据提供给用户，供其分析夹紧状态。

说明

仅当驱动参数 r5000 中显示以下状态值时，下文说明的“分析传感器 S1 来生成夹紧状态的状态值”和“限制主轴转速”才会进行。

- r5000.0 == 1：配有传感器
- r5000.1 == 1：传感器 S1（夹紧状态）存在
- r5000.10 == 1：生成状态值，转速限制 p5043 生效

另见章节“传感器数据 (页 1511)”，段落：“系统数据：传感器数据”

状态值

为了简化分析，夹紧状态在系统数据中国也作为状态字 0 - 11 提供。

夹紧范围对应特定的夹紧状态。电压范围可通过驱动参数 p5041[0...5] 设置。

针对电压范围，还可通过驱动参数 p5040 额外设置一个电压公差。

说明

“电压范围 ± 电压公差”不允许重叠。

转速限值

对于状态值为 3 - 10 的夹紧状态，可通过驱动参数 p5043[0...6] 设定转速限值。在其他夹紧状态下（状态值 1、2 和 11），转速限值固定设置为 0 [rpm]。

控制系统会在各种夹紧状态下降主轴转速限制在相应限值内。

说明

转速限值的修改

满足以下条件时，在驱动参数 p5043[0...6] 中进行的转速限值修改才会在控制系统中生效（将主轴转速限制在新的转速限值范围内）：

- 上电复位，或控制系统的关闭/接通
- 为主轴取消“驻留”状态（参见““驻留”机床轴 (页 131)”章节）

关联：状态值、电压范围和转速限值之间

状态值 [1]	夹紧状态	电压范围 [2]		转速限值
		上限	下限	
0	传感器 S1 不存在，或状态值无效	---	---	---
1	状态初始化中	---	---	[3]
2	已松开，发出消息（故障状态）	---	p5041[0] + p5040	[3]
3	已松开	p5041[0]	p5041[1]	p5043[0]
4	夹紧，带刀具	---	---	p5043[1]
5	松开，带刀具	---	---	p5043[2]
6	松开，无刀具	---	---	p5043[3]
7	已夹紧，带刀具 AND S4 == 0	p5041[2]	p5041[3]	p5043[4]
8	已夹紧，带刀具 AND S4 == 1			p5043[4]
9	夹紧，无刀具	---	---	p5043[5]

状态值 [1]	夹紧状态	电压范围 [2]		转速限值
		上限	下限	
10	已夹紧，无刀具	p5041[4]	p5041[5]	p5043[6]
11	已夹紧，发出消息（故障状态）	p5041[5] - p5040	---	[3]

[1] 状态值可在控制系统中通过以下系统数据读取：

- 系统变量： $VA_MOT_CLAMPING_STATE[<轴>]
- NC/PLC 接口： DB31, ... DBW134
- OPI 变量： vaMotClampingState
- 驱动参数： r5001

[2] p5041[0...5]：电压阈值，p5040：电压阈值公差

[3] 转速限值固定设定：0 [rpm]

17.9.4 其他驱动参数

p5042：过渡时间

在驱动参数 p5042 中可为夹紧状态识别设置以下时间：

- p5042[0]：“已夹紧，带刀具”的稳定时间
  夹紧状态“已夹紧，带刀具”必须在主轴电机中保持至少为设置值的稳定时间，之后才能将状态发送至控制系统。
- p5042[1]：最大夹紧时间
  从“已松开”状态过渡到“已夹紧，带道具”或“已夹紧，无道具”状态最多可持续的时间。

r5044：夹紧循环中的转速限制

驱动参数 r5044 中显示在“已夹紧，无刀具”状态下生效的转速限值（p5043[6]）。

值 65535 表示转速限值未生效。

17.10 前提条件

17.10.1 闭环控制参数的修改

主轴不处于位置闭环控制下时，通过 NEWCONF 指令进行的机床数据修改在主轴未停止时同样生效。

修改闭环控制参数时，新值生效后可能会引起转速设定值跃变。 闭环控制参数例如包括：

- MD32200 $MA_POSCTRL_GAIN（KV 系数）
- MD32210 $MA_POSCTRL_INTEGR_TIME（位置控制器积分时间）
- MD32410 $MA_AX_JERK_TIME（轴向急动滤波器的时间常量）

17.11 示例

17.11.1 自动齿轮档选择（M40）

示例

显示新程序段搜索变量的内容：
启用自动齿轮档选择（M40）：

S0...500	1. 齿轮档
S501..1000	2. 齿轮档
S1001..2000	3. 齿轮档

系统变量的内容：

```
$P_SEARCH_S          ; 收集的 S 值
$P_SEARCH_DIR        ; 收集的旋转方向
$P_SEARCH_GEAR       ; 收集的齿轮档
```

收集的	S 值：	旋转方向：	齿轮档：
	; 0/最后的转速	-5	40/最后的副主轴
N05 G94 M40 M3 S1000	; 1000	3	40
N10 G96 S222	: 222	3	40
N20 G97	; f (PlanAxPosPCS)*	3	40
N30 S1500	; 1500	3	40
N40 SPOS=0**	; 1500	-19	40
N50 M19**	; 1500	-19	40
N60 G94 G331 Z10 S300	; 300	-19	40
N70 M42	; 300	-19	42
N80 M4	; 300	4	42
N90 M70	; 300	70	42
N100 M3 M40	; 300	3	40
N999 M30			

* f (PlanAxPosPCS)：转速取决于工件坐标系中平面轴的当前位置。

** （$P_SEARCH_SPOS 和 $P_SEARCH_SPOSMODE 会被写入）

17.12 数据表

17.12.1 机床数据

17.12.1.1 NC 专用机床数据

编号	名称：$MN_	说明
10192	GEAR_CHANGE_WAIT_TIME	重整时对齿轮档切换应答的等待时间
10714	M_NO_FCT_EOP	复位后生效的主轴用 M 功能
12060	OVR_SPIND_IS_GRAY_CODE	主轴倍率，格雷编码
12070	OVR_FACTOR_SPIND_SPEED	主轴倍率开关的权重系数
12080	OVR_REFERENCE_IS_PROG_FEED	倍率参考速度

编号	名称：$MN_	说明
12082	OVR_REFERENCE_IS_MIN_FEED	定义轨迹倍率的参考基准
12090	OVR_FUNCTION_MASK	选择倍率设定

17.12.1.2 通道专用机床数据

编号	名称：$MC_	说明
20090	SPIND_DEF_MASTER_SPIND	通道中主主轴的缺省设置
20092	SPIND_ASSIGN_TAB_ENABLE	使能/禁用主轴转换器
20850	SPOS_TO_VDI	启用 SPOS/SPOSA 时将 M19 输出至 PLC
22400	S_VALUES_ACTIVE_AFTER_RESET	S 功能复位后有效

17.12.1.3 进给轴/主轴专用机床数据

编号	名称：$MA_	说明
30300	IS_ROT_AX	回转轴
30310	ROT_IS_MODULO	模数转换
31044	ENC_IS_DIRECT2	中间齿轮比上的编码器
31050	DRIVE_AX_RATIO_DENOM	负载齿轮比分母
31060	DRIVE_AX_RATIO_NUMERA	负载齿轮比分子
31064	DRIVE_AX_RATIO2_DENOM	中间齿轮比分母
31066	DRIVE_AX_RATIO2_NUMERA	中间齿轮比分子
31070	DRIVE_ENC_RATIO_DENOM	测量齿轮比分母
31080	DRIVE_ENC_RATIO_NUMERA	测量齿轮比分子
31122	BERO_DELAY_TIME_PLUS	正方向上的 BERO 延时
31123	BERO_DELAY_TIME_MINUS	负方向上的 BERO 延时
32200	POSCTRL_GAIN	K_V 系数
32800	EQUIV_CURRCTRL_TIME	用于前馈控制的电流环等效时间常数
32810	EQUIV_SPEEDCTRL_TIME	用于前馈控制的转速闭环等效时间常数
32910	DYN_MATCH_TIME	动态响应自适应的时间常数
34040	REFP_VELO_SEARCH_MARKER	参考点蠕变速度

编号	名称：$MA_	说明
34060	REFP_MAX_MARKER_DIST	零脉冲距离监控
34080	REFP_MOVE_DIST	使用距离编码系统时的参考点距离/目标点
34090	REFP_MOVE_DIST_CORR	距离编码的参考点偏移/绝对偏移
34100	REFP_SET_POS	参考点值
34200	ENC_REFP_MODE	回参考点模式
35000	SPIND_ASSIGN_TO_MACHAX	分配主轴至机床主轴
35010	GEAR_STEP_CHANGE_ENABLE	齿轮档切换类型
35012	GEAR_STEP_CHANGE_POSITION	齿轮档切换位置
35014	GEAR_STEP_USED_IN_AXISMODE	M70 下轴模式的齿轮档
35020	SPIND_DEFAULT_MODE	主轴初始设置
35030	SPIND_DEFAULT_ACT_MASK	激活主轴初始设置
35035	SPIND_FUNCTION_MASK	设置主轴专用功能
35040	SPIND_ACTIVE_AFTER_RESET	主轴复位后有效
35090	NUM_GEAR_STEPS	建立的齿轮档的数量
35092	NUM_GEAR_STEPS2	第 2 齿轮档数据组：建立的齿轮档的数量
35100	SPIND_VELO_LIMIT	最大主轴转速
35110	GEAR_STEP_MAX_VELO[n]	自动齿轮档切换的最大转速
35112	GEAR_STEP_MAX_VELO2[n]	第 2 齿轮档数据组：自动齿轮档切换的最大转速
35120	GEAR_STEP_MIN_VELO[n]	自动齿轮档切换的最小转速
35122	GEAR_STEP_MIN_VELO2[n]	第 2 齿轮档数据组：自动齿轮档切换的最小转速
35130	GEAR_STEP_MAX_VELO_LIMIT[n]	齿轮档的最大转速
35135	GEAR_STEP_PC_MAX_VELO_LIMIT[n]	位置闭环控制下齿轮档的最大转速
35140	GEAR_STEP_MIN_VELO_LIMIT[n]	齿轮档的最小转速
35150	SPIND_DES_VELO_TOL	主轴转速公差
35160	SPIND_EXTERN_VELO_LIMIT	PLC 主轴转速限制
35200	GEAR_STEP_SPEEDCTRL_ACCEL[n]	转速控制运行下的加速度
35210	GEAR_STEP_POSCTRL_ACCEL[n]	位置闭环控制运行下的加速度
35212	GEAR_STEP_POSCTRL_ACCEL2[n]	第 2 齿轮档数据组：位置闭环控制运行下的加速度
35220	ACCEL_REDUCTION_SPEED_POINT	降低加速度转速限值
35230	ACCEL_REDUCTION_FACTOR	降低的加速度

编号	名称：$MA_	说明
35300	SPIND_POSCTRL_VELO	位置闭环控制激活转速
35350	SPIND_POSITIONING_DIR	未同步的主轴定位时的旋转方向
35400	SPIND_OSCILL_DES_VELO	往复转速
35410	SPIND_OSCILL_ACCEL	往复时的加速度
35430	SPIND_OSCILL_START_DIR	往复时的起始方向
35440	SPIND_OSCILL_TIME_CW	M3 方向的往复时间
35450	SPIND_OSCILL_TIME_CCW	M4 方向的往复时间
35500	SPIND_ON_SPEED_AT_IPO_START	主轴在设定区域时的进给使能
35510	SPIND_STOPPED_AT_IPO_START	主轴停止时的进给使能
35550	DRILL_VELO_LIMIT[n]	攻丝的最大转速
35590	PARAMSET_CHANGE_ENABLE	通过 PLC 可进行参数组规定
36060	STANDSTILL_VELO_TOL	“轴/主轴停止”阈值速度
36200	AX_VELO_LIMIT	速度监控阈值

17.12.2 设定数据

17.12.2.1 通道专用设定数据

编号	名称：$SC_	说明
42600	JOG_FEED_PER_REF_SOURCE	JOG 方式下的旋转进给控制
42800	SPIND_ASSIGN_TAB	主轴编号转换器
42900	MIRROR_TOOL_LENGTH	镜像刀具长度补偿
42910	MIRROR_TOOL_WEAR	镜像刀具长度补偿磨损值
42920	WEAR_SIGN_CUTPOS	镜像加工平面磨损值
42930	WEAR_SIGN	对所有磨损值的符号进行反向
42940	TOOL_LENGTH_CONST	在更换加工平面（G17 至 G19）时保留刀具长度分量的分配

17.12.2.2 进给轴/主轴专用设定数据

编号	名称：$SA_	说明
43200	SPIND_S	通过 PLC 接口启动主轴时的转速
43202	SPIND_CONSTCUT_S	通过 PLC 接口启动主轴时的切削速度
43206	SPIND_SPEED_TYPE	通过 PLC 接口启动主轴时的主轴类型
43210	SPIND_MIN_VELO_G25	程序 主轴转速限制 G25
43220	SPIND_MAX_VELO_G26	程序 主轴转速限制 G26
43230	SPIND_MAX_VELO_LIMS	编写的主轴转速限制 G96/G961
43235	SPIND_USER_VELO_LIMIT	最大主轴转速
43240	M19_SPOS	M19 主轴定位中的主轴位置
43250	M19_SPOSMODE	M19 主轴定位中的逼近模式
43300	ASSIGN_FEED_PER_REF_SOURCE	轴/主轴旋转进给率

17.12.3 信号

17.12.3.1 发送至进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
进给倍率 A 至 H	DB31, ...DBX0.0-7	DB380x.DBX0.0-7
进给轴/主轴禁止	DB31, ...DBX1.3	DB380x.DBX1.3
跟踪运行	DB31, ...DBX1.4	DB380x.DBX1.4
位置测量系统 1	DB31, ...DBX1.5	DB380x.DBX1.5
位置测量系统 2	DB31, ...DBX1.6	DB380x.DBX1.6
补偿生效	DB31, ...DBX1.7	DB380x.DBX1.7
伺服使能	DB31, ...DBX2.1	DB380x.DBX2.1
主轴复位/删除剩余行程	DB31, ...DBX2.2	DB380x.DBX2.2
速度/主轴转速限制	DB31, ...DBX3.6	DB380x.DBX3.6
程序测试轴/主轴使能	DB31, ...DBX3.7	DB380x.DBX3.7
实际齿轮档 A 至 C	DB31, ...DBX16.0-2	DB380x.DBX2000.0-2
齿轮档已切换	DB31, ...DBX16.3	DB380x.DBX2000.3

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
主轴重新同步 1	DB31, ...DBX16.4	DB380x.DBX2000.4
主轴重新同步 2	DB31, ...DBX16.5	DB380x.DBX2000.5
齿轮档切换时无 n 监控	DB31, ...DBX16.6	DB380x.DBX2000.6
删除 S 值	DB31, ...DBX16.7	DB380x.DBX2000.7
主轴进给倍率有效	DB31, ...DBX17.0	DB380x.DBX2001.0
定位中重新同步主轴 1	DB31, ...DBX17.4	DB380x.DBX2001.4
定位中重新同步主轴 2	DB31, ...DBX17.5	DB380x.DBX2001.5
M3/M4 取反	DB31, ...DBX17.6	DB380x.DBX2001.6
通过 PLC 往复	DB31, ...DBX18.4	DB380x.DBX2002.4
往复使能（往复转速）	DB31, ...DBX18.5	DB380x.DBX2002.5
往复方向：顺时针（设定旋转方向为顺时针）	DB31, ...DBX18.6	DB380x.DBX2002.6
往复方向：逆时针（设定旋转方向为逆时针）	DB31, ...DBX18.7	DB380x.DBX2002.7
主轴倍率 A 至 H	DB31, ...DBX19.0-7	DB380x.DBX2003.0-7

17.12.3.2 来自进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
主轴/回转轴	DB31, ...DBX60.0	DB390x.DBX0.0
超出编码器极限频率 1	DB31, ...DBX60.2	DB390x.DBX0.2
超出编码器极限频率 2	DB31, ...DBX60.3	-
已回参考点/已同步 1	DB31, ...DBX60.4	DB390x.DBX0.4
已回参考点/已同步 2	DB31, ...DBX60.5	DB390x.DBX0.5
采用粗准停到达位置	DB31, ...DBX60.6	DB390x.DBX0.6
采用精准停到达位置	DB31, ...DBX60.7	DB390x.DBX0.7
进给轴/主轴停止（$n < n_{min}$）	DB31, ...DBX61.4	DB390x.DBX1.4
位置控制器生效	DB31, ...DBX61.5	DB390x.DBX1.5
转速控制器生效	DB31, ...DBX61.6	DB390x.DBX1.6
电流控制器生效	DB31, ...DBX61.7	DB390x.DBX1.7
已恢复 1	DB31, ...DBX71.4	DB390x.DBX11.4
已恢复 2	DB31, ...DBX71.5	DB390x.DBX11.5

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
设定齿轮档 A 至 C	DB31, ...DBX82.0-2	DB390x.DBX2000.0-2
切换齿轮档	DB31, ...DBX82.3	DB390x.DBX2000.3
超出转速限值	DB31, ...DBX83.0	DB390x.DBX2001.0
设定转速已限制	DB31, ...DBX83.1	DB390x.DBX2001.1
设定转速已提升	DB31, ...DBX83.2	DB390x.DBX2001.2
主轴位于设定区域内	DB31, ...DBX83.5	DB390x.DBX2001.5
实际旋转方向：顺时针	DB31, ...DBX83.7	DB390x.DBX2001.7
刚性攻丝生效	DB31, ...DBX84.3	DB390x.DBX2002.3
主轴同步模式生效	DB31, ...DBX84.4	DB390x.DBX2002.4
主轴定位模式生效	DB31, ...DBX84.5	DB390x.DBX2002.5
主轴往复模式生效	DB31, ...DBX84.6	DB390x.DBX2002.6
主轴控制模式生效	DB31, ...DBX84.7	DB390x.DBX2002.7
主轴实际到达位置	DB31, ...DBX85.5	DB390x.DBX2003.5
用于主轴的 M 功能	DB31, ...DBB86-87	DB370x.DBD0000
用于主轴的 S 功能	DB31, ...DBB88-91	DB370x.DBD0004
传感器系统存在	DB31, ...DBX132.0	DB390x.DBX7000.0
传感器 S1 存在（夹紧状态）	DB31, ...DBX132.1	DB390x.DBX7000.1
传感器 S4 存在（活塞末端）	DB31, ...DBX132.4	DB390x.DBX7000.4
传感器 S5 存在（电机轴角度位置）	DB31, ...DBX132.5	DB390x.DBX7000.5
生成状态值，转速限制 p5043 生效	DB31, ...DBX133.2	DB390x.DBX7001.2
夹紧状态	DB31, ...DBW134	DB390x.DBW7002
模拟值：夹紧状态	DB31, ...DBW136	DB390x.DBW7004
传感器 S4 （活塞末端）	DB31, ...DBX138.4	DB390x.DBX7006.4
传感器 S5 （电机轴角度位置）	DB31, ...DBX138.5	DB390x.DBX7006.5

V1：进给率

18

18.1 简要说明

进给方式

进给率用于确定加工速度（轴进给率或轨迹进给率），每一种插补方式下的轮廓或刀具中心点轨迹都会遵循特定的进给率（取决于 G 指令），在启用刀具补偿时也是如此。

为了实现对各种工艺应用（车削、铣削、钻孔等）的最优匹配，可在下列进给类型中进行选择：

- 快进进给率（G0）
- 反比时间进给率（G93）
- 线性进给率（G94）
- 旋转进给率（G95）
- 恒定切削速度（G96、G961）
- 恒定转速（G97、G971）
- 螺纹切削（G33、G34、G35）时的进给率
- 刚性攻丝（G331、G332）中的进给率
- 带补偿夹具攻丝（G63）中的进给率
- 倒角/倒圆进给率 FRC、FRCM
- 逐段进给率 FB

进给率对轴的指定

为了适应各种技术需求，可以多变的形式将进给率应用于轴。

可供采用的方案例如包括：

- 对工作平面和进给轴分别启用不同的进给率
- 轨迹进给中轴的灵活指定
- 定位轴进给

进给控制

为了适应加工期间变化的技术条件以及进行测试，可通过以下方式修改编写的进给率：

- 通过机床控制面板
- 通过操作面板前端
- 通过 PLC
- 通过程序指令

18.2 轨迹进给率 F

轨迹进给率 F

轨迹进给率表示的是相关轴速度分量的几何总和。其通过相互插补的轴的单次运动得到。

缺省设置下使用的是所编写的几何轴的轴向速度。通过 FGROUP 指令也可将其他几何轴和/或同步轴纳入轨迹进给率的计算。

轨迹进给率 F 决定了加工速度，在每种插补方式都须遵循，在启用刀具补偿时也是如此。地址 F 下编写的值在一个程序中一直保留，直至编写新的 F 值或新的进给类型。

进给率 F 的取值范围

参见功能说明 G2：“速度，设定值/实际值系统，闭环控制”，章节：“速度 (页 363)”。

PLC 接口上的 F 值

当前轨迹进给率的 F 值会记录至用于辅助功能的通道专用 PLC 接口（DB21, ... DBB158 至 193）。

对应接口信号（修改信号，F 值）的详细说明请见 “H2：输出到 PLC 的辅助功能 (页 429)”。

过渡圆弧处的进给率

文档：
编程手册之基本原理分册

向内和向外弯曲的轨迹段的进给率

对于圆弧程序段、同向弯曲的样条程序段，以及激活刀具半径补偿（G41/G42）时，编写的进给率可在中心点轨迹或轮廓上生效（取决于向内弯曲和向外弯曲的轨迹段）。

为此，有一个 G 指令组可供使用：

- CFTCP
  编写的进给率在中心点轨迹上生效。
- CFC
  编写的进给率在轮廓上生效。
- CFCIN
  编写的进给率在轮廓上向内弯曲部分生效。

文档：
编程手册之基本原理分册

最大轨迹速度

最大轨迹速度由参与进给的直线轴或回转轴的最大速度（MD32000 $MA_MAX_AX_VELO）得出，也就是说将几个最大值中最低的一个用作最大轨迹速度。不可超出该速度。

若编写了 G0，则会采用受到 MD32000 $MA_MAX_AX_VELO 限制的轨迹速度运行。

轨迹轴的极限速度

此外对于轨迹轴（几何轴和同步轴），还可通过 FL[<轴>] 指令为相应轴编写一个极限速度。

这样便可分别为工作平面和进给轴编写进给率。这表示同时为轨迹相关插补和进给轴设定一个进给速度。进给轴指的是垂直于所选加工平面生效的轴。可编写进给轴专用进给率来限制轴速度以及轨迹速度。此时切勿启用通过框架进行的坐标旋转。也就是说进给轴必须是基本坐标系中的轴。此功能例如用于应对以下情形：铣刀在端面上的切削效率小于通过整个铣刀圆周进行切削时的效率。

编程示例：

程序代码	注释
...G94 ...	选择进给类型 (mm/min)
X30 Y20 F200	；轨迹进给率 = 200 mm/s
FL[Z]=50 Z-30	Z 轴的最大进给率：50 mm/s

低分辨率编码器

使用低分辨率的编码器时，可通过平滑实际值来获取更稳定的轨迹运动或轴运动。所设置的时间常量越大，达到的实际值平滑效果也就越佳，但滞后行程也会越大。

MD34990 $MA_ENC_ACTVAL_SMOOTH_TIME[<轴>]（实际值的平滑时间常量）

平滑实际值应用于：

- 螺纹切削（G33、G34、G35）
- 旋转进给率（G95、G96、G97、FPRAON）
- 显示转速、实际位置和实际速度

18.2.1 进给类型 G93、G94、G95

有效性

进给类型 G93、G94、G95 在 AUTO 运行方式下对 G 指令组 1（除 G0）生效。

在 JOG 模式下可通过 G94 或 G95 运行。

文档：
功能手册之扩展功能分册；手动运行和通过手轮进行手动运行（H1）

反比时间进给率（G93）

编写程序段退出时，若编写持续时间比编写进给率更简单，则可使用反比时间进给率。

反比时间进给率通过以下公式计算：

```
F = v / s
```

其中 F: 反比时间进给率，单位 rpm
v: 所需的轨迹速度，单位 mm/min 或 inch/min
s: 轨迹长度，单位 mm 或 inch

编程举例

程序代码	注释
N10 G1 G93 X100 Y200 F2	编程的轨迹行程在 0.5 分钟内运行完毕。
...	

说明

G41/G42 生效时不允许使用 G93。如果各程序段的长度差别很大，那么在启用 G93 时应在每个程序段中编写一个新的 F 值。

线性进给率（G94）

线性进给率采用以下单位基于直线轴或回转轴编写：

- 采用公制系统时：[mm/min，degrees/min]
- 采用英制系统时：[inch/min，degrees/min]

旋转进给率（G95）

旋转进给率采用以下单位基于主主轴编写：

- 采用公制系统时：[mm/rev]
- 采用英制系统时：[inch/rev]
- 对于回转轴：[degrees/rev]

轨迹速度由主轴的实际转速按以下公式得出：

```
V = n * F
```

其中 V: 轨迹速度，单位 mm/min 或 inch/min
n: 主主轴转速，单位 rev/min
F: 编写的旋转进给率，单位 mm/rev 或 inch/rev

说明

若在进给类型 G93、G94、G95 之间切换，那么编写的 F 值将被删除。

每齿进给量

可编写实际应用中更为常用的齿进给值 FZ（每齿进给量）来替代旋转进给值 F，特别是对于铣削加工。

通过生效刀具补偿数据组的刀具参数 $TC_DPNT（每转齿数），控制系统根据每个运行程序段中编写的每齿进给量计算生效的旋转进给率：

```
F = FZ * $TC_DPNT
```

其中 F: 旋转进给率，单位 mm/rev 或 inch/rev

FZ： 每齿进给量，单位 mm/tooth 或 inch/tooth

$TC_DP 刀具参数：齿数/转

NT:

示例：5 齿铣刀($TC_DPNE = 5)

程序代码	注释
N10 G0 X100 Y50	
N20 G1 G95 FZ=0.02	；每齿进给率 0.02 毫米/齿
N30 T3 D1	；切换刀具，并激活刀具补偿数据组。
M40 M3 S200	；主轴转速 200 转/分钟
N50 X20	; 以 FZ = 0.02 毫米/齿铣削
	; 生效的旋转进给率：
	; F = 0.02 毫米/齿 * 5 齿/转 = 0.1 毫米/转
	; 或者
	; F = 0.1 毫米/转 * 200 转/分钟 = 20 毫米/分钟
...	

设定数据

- 在 JOG 运行方式下启用旋转进给率
  JOG 运行方式下，关系到通道（轴目前分配于该通道）主主轴的、与旋转进给率相关的轴的特性取决于 NC 特定设定数据的设置：
  SD41100 $SN_JOG_REV_IS_ACTIVE，位 <x>

位	值	含义
0	0	该特性取决于其他设定数据，取决于轴的功能： ● **进给轴 / 主轴**:SD43300 $SA_ASSIGN_FEED_PER_REV_SOURCE ● **带旋转框架**的**几何轴**：SD42600 $SC_JOG_FEED_PER_REV_SOURCE ● **定向轴**：SD42600 $SC_JOG_FEED_PER_REV_SOURCE
	1	以旋转进给率运行轴。 **前提条件** 如果满足以下条件，则不以旋转进给率运行： ● **进给轴 / 主轴**:主主轴静止 **且** SD43300 $SA_ASSIGN_FEED_PER_REV_SOURCE == -3 ● **带旋转框架**的**几何轴**：主主轴静止 **且** SD42600 $SC_JOG_FEED_PER_REV_SOURCE == -3 ● **定向轴**：主主轴静止 且 SD42600 $SC_JOG_FEED_PER_REV_SOURCE == -3
1 [1)]	0	在**急速运行模式**中，**以**旋转进给率运行。
	1	在**急速运行模式**中，**不以**旋转进给率运行。
2 [1)]	0	在 **JOG 方式下手轮运行**时，**以**旋转进给率运行。
	1	在 **JOG 方式下手轮运行**时，**不以**旋转进给率运行。
3 [1)]	0	在 **DRF 手轮运行**时，**以**旋转进给率运行。
	1	在 **DRF 手轮运行**时，**不以**旋转进给率运行。
1) 该功能可额外用于设置位 1。		

NC/PLC 接口信号

- 旋转进给有效（**通道专用的**）
  通过接口信号显示通道中的轴是以旋转进给率运行的：
  – 运行方式 AUTOMATIC：轨迹轴或同步轴
  – 运行方式 JOG：几何轴
  DB31, ... DBX62.2 == 1（旋转进给率生效）
- 旋转进给有效（轴专用的）
  通过接口信号显示轴是以旋转进给率运行的：
  DB31, ... DBX33.2 == 1（旋转进给率生效）

18.2.2 进给类型 G96、G961、G962、G97、G971

恒定切削速度（G96、G961）

恒定切削速度在车床上使用，目的是独立于工件的工作半径保持恒定的切削条件。这样可保证刀具始终在最佳的切削效率范围运行，从而延长其寿命。

使用 G96、G961

编写 G96、G961 时，对应的 S 值会作为切削速度在平面轴中被解译为 m/min 或 ft/min。加工期间平面轴中的工作半径减小时，转速将一直提升，直至到达恒定的切削速度。

在零件程序中首次选择 G96、G961 时必须以 m/min 或 ft/min 输入一个恒定切削速度，重新选择时则可选择输入方法。

启用 G96 时，控制系统会自动切换至旋转进给率（和 G95 下相同），也就是说，编写的进给率 F 会被解译为 mm/rev 或 inch/rev。

在编写 G961 时会自动选择线性进给率（和 G94 下相同）。编写的进给率 F 会被解译为 mm/min 或 inch/min。

确定主轴转速

基于编写的切削速度（S_{G96} 或 S_{G961}）以及当前 TCP 上平面轴（半径）的坐标位置，控制系统按照以下公式确定主轴转速：

$$n = \frac{S_{Speed}}{2 * \pi * r}$$

n: 主轴转速

S_{Speed}: 编写的切削速度

π 圆弧常数

r: 半径（旋转中心与 TCP 的距离）

为了确定半径，此时会假定下列条件：

- WCS 中的平面轴位置 0 为旋转中心。
- 位置补偿（例如在线刀具补偿、外部零点偏移、$AA_OFF、DRF 偏移和编译循环）和耦合产生的位置分量（例如 TRAIL 下的跟随轴）在确定半径时不会被考虑。

框架（例如可编程框架 SCALE、TRANS 或 ROT）会在确定主轴转速时被考虑在内，且会引起转速变化，前提条件是 TCP 上的生效直径变化。

为通道中的多根平面轴编写直径和参考轴：

可同时或分别激活多根平面轴。

- 编程和显示在操作界面中通过直径进行。
- 设定的参考轴通过 `SCC[<Achse>]` 指定给恒定切削速度 G96、G961、G962

更多信息参见功能说明“P1：端面轴 (页 933)”。

示例

S_{G96} = 230 m/min

- 若 r = 0.2 m → n = 183.12 rev/min
- 若 r = 0.1 m → n = 366.24 rev/min

⇒ 工件直径越小，转速就越高。

必须为 G96、G961 或 G962 将一根几何轴定义为平面轴。

平面轴的位置会影响主主轴转速，其通过以下通道专用机床数据定义：

MD20100 $MC_DIAMETER_AX_DEF（带平面轴功能的几何轴）

G96、G961 或 G962 功能的前提条件是：机床零点或平面轴的工件零点位于主轴的旋转中心。

恒定转速（G97、G971）

启用 G97、G971 功能时，“恒定切削速度”（G96、G961）功能会被取消，最后计算的主轴转速会被保存。G97 下进给率会被解译为旋转进给率（同 G95）。在编写 G971 时会选择线性进给率（和 G94 下相同）。进给率 F 会被解译为 mm/min 或 inch/min。

G97、G971 生效时，可通过重新编写一个 S 值设定一个新的主轴转速。G96、G961 下编写的切削速度将不受其影响。

通过 G97、G971 可在无加工的情况下避免平面轴中运动时的转速变化。

说明

在工件上进行加工时（G1、G2、G3、样条插补等进给率 F 生效的过程），G96、G961 才生效。

G96、G961 以及 G0 程序段生效时主轴转速的特性可在以下通道专用机床数据中设置：

MD20750 ALLOW_G0_IN_G96（G96、G961 下的 G0 逻辑）

采用恒定切削速度 G96、G961 时不会发生齿轮档切换。

主轴倍率开关对确定的主轴转速生效。

平面轴中的 DRF 偏移不会影响主轴转速设定值计算。

再加工开始时（G0 后）时和 NC 停止、G60、G09 后，系统会等待轨迹启动“转速实际值 = 转速设定值”。

接口信号“转速实际值 = 转速设定值”和“设定转速已限制”不会因为内部转速设定而被修改。

在低于最低转速，或识别出“进给轴/主轴停止”信号时，“转速实际值 = 转速设定值”会被复位。

已开始的轨迹加工（G64，精磨）不会被中断。

G96、G961 下的转速限制

可为“恒定切削速度”功能设定一个最大主轴转速。

- 设定数据中：
  SD43230 $SA_SPIND_MAX_VELO_LIMS（启用 G96/G961 时的主轴转速限制）
- 在零件程序中（为主主轴）通过编程指令 LIMS

最后修改的值（LIMS 或 SD）将生效。

LIMS 适用于 G96、G961、G97，且可在一个零件程序的一个程序段中设定最多四个转速限制。为此可在零件程序指令 LIMS[<Sn>] 中通过主轴编号 <Sn> = 1、2、3 或 4 设定相应的主主轴。

当程序段进入主处理时，编写的所有值都会被接收至设定数据 SD43230 $SA_SPIND_MAX_VELO_LIMS。

根据以下机床数据：
MD10710 $MN_PROG_SD_RESET_SAVE_TAB[n]（待更新设定数据）
写入的包含 LIMS 的转速限制在控制系统断电后继续保存。
重新激活 G96、G961、G97 时该转速限制重新生效。

不允许超出通过 G26 或通过以下设定数据设定的主轴转速限值：
SD43220 $SA_SPIND_MAX_VELO_G26（最大主轴转速）

编程错误导致超出转速限值（G26 或 SD43220 $SA_SPIND_MAX_VELO_G26）时，以下 NC/PLC 接口信号将置位：

DB31, ... DBX83.1（编写的转速过高）

为了确保在零件直径较大时能完成回转，转速不得低于定义的最小主轴转速。
该转速可通过以下设定数据设置：
SD43210 $SA_SPIND_MIN_VELO_G25（最小主轴转速）
以及针对齿轮档通过以下机床数据设置：
MD35140 $MA_GEAR_STEP_MIN_VELO_LIMIT（齿轮档的最小转速）

最小主轴转速可以在零件程序中用 G25 进行更改。编程低于转速限值（G25 或 SD43210 $SA_SPIND_MIN_VELO_G25）时，以下 NC/PLC 接口信号将置位：

DB31, ... DBX83.2（设定转速过低）

主轴转速限制的其它信息参见功能描述 S1：“主轴”，章节："主轴监控 (页 1498)"。

说明

零件程序中通过 G25/G26/LIMS 修改的转速限值会纳入设定数据并在程序结束后仍然保留。

如果不愿在程序结束之后采用通过 G25/G26/LIMS 修改的转速限值，则必须在机床制造商 GUD 模块中增加以下定义：

REDEF $SA_SPIND_MIN_VELO_G25 PRLOC

REDEF $SA_SPIND_MAX_VELO_G26 PRLOC

REDEF $SA_SPIND_MAX_VELO_LIMS PRLOC

G96、G961 下的主主轴切换

在 G96、G961 生效的情形下切换主主轴时，原主主轴的转速将保留。这相当于从 G96 过渡至 G97。通过 SETMS 重新定义的主主轴将执行如此生成的“恒定切削速度”功能。

报警

恒定切削速度 G96、G961、G962

- 若未编写 F 值，则会输出报警 10860“未编写进给率”。G0 程序段不会生成该报警。
- 编写了负向的轨迹速度时，会触发报警 14800“编写的轨迹速度小于等于零”。
- 若 G96、G961、G962 生效时未通过以下机床数据定义平面轴：
  MD20100 $MC_DIAMETER_AX_DEF（带平面轴功能的几何轴）
  则会触发报警 10870“未定义平面轴”。
- G96、G961 生效时，若通过编程指令 LIMS 为最大主轴转速编写了负值，则会输出报警 14820“为 G96、G961 编写的最大主轴转速为负值”。
- 若首次选择 G96、G961 时未编写恒定切削速度，则会输出报警 10900“没有为恒定切削速度编写 S 值”。

18.2.3 螺纹切削（G33, G34, G35, G335, G336）中的进给率

18.2.3.1 G33 中的进给率

G33

使用 G33 功能可以加工螺距恒定的螺纹。

转速 S、进给率 F、螺距

G33 螺纹加工中启用一个旋转进给率 [mm/rev]，其通过螺距 [mm/rev] 编程来设定。

螺纹长度所对应的轴速度通过编写的主轴转速和螺距计算：

进给率 F [mm/min] = 转速 S [rev/min] * 螺距 [mm/rev]

在加速斜坡末端，系统将建立主轴位置实际值（启用 SPCON 时主主轴的设定位置）与轴位置设定值的位置耦合。 从此时轴相对主轴零脉冲（零脉冲偏移已计算在内）的位置看来，就像轴在程序段开始处越过螺纹起始位置（零脉冲 + SF）时进行了急剧加速一样。 此时轴的跟随误差被补偿。

最小主轴转速

为了确保能在较低的转速下完成回转，不可低于最小主轴转速。

该限值可通过

- 设定数据设置：
  SD43210 $SA_SPIND_MIN_VELO_G25（最小主轴转速）
- 或针对齿轮档通过以下机床数据设置：
  MD35140 $MA_GEAR _STEP_MIN_VELO_LIMIT（齿轮档切换的最小转速）

最小主轴转速可以在零件程序中用 G25 进行更改。

NC 停止、单程序段

NC 停止和单程序段模式直到螺纹链末尾才生效。 所有相连的 G33 程序段及后面跟随的第一个非 G33 程序段会像一个程序段一样被连续执行。

提前无损终止

在螺纹切削加工到达终点前，可实现不损坏螺纹的中断。 这可通过激活回退运动来实现。

含 ROT 框架的螺纹切削

在 G33、G34、G35 螺纹切削中启用 ROT 框架时，若该框架会导致螺纹长度乃至螺距变化，则会输出报警 10607“无法执行含框架螺纹”。 允许启用围绕螺纹轴的旋转。

若在应用中一定要使用 ROT 指令，可在机床数据 MD11410 $MN_SUPPRESS_ALARM_MASK 中设置位 12 来抑制报警 10607“无法执行含框架螺纹”。

所有其他框架均会被 NC 接受，不会触发报警。 系统会对 SCALE 引起螺距变化的特性进行提示。

18.2.3.2 G34 和 G35 中线性递增式/递减式的螺距变化

功能

螺距增加量（G34）描述的是螺距增大的数值。螺距越大，工件上螺纹线之间的距离越大. 在启用恒定主轴转速的情形下，螺纹轴的速度将提升。

螺距降低量（G35）则是相反的情形。

针对螺距变化引入下面两个概念：

- G34：螺距递增变化
- G35：螺距递减变化

G34 和 G35 都会隐性激活 G33，并且提供在 F 下编写螺距变化的功能。 如果已知一个螺纹的起始螺距和最终螺距，那么就可以根据下面的等式计算出需要编写的螺距变化：

$$F = \frac{|k_e^2 - k_a^2|}{2 * l_G}$$

含义：

F: 待编写的螺纹螺距变化 [mm/rev ²]

k_e: 螺纹轴轴目标点坐标的螺距 [mm/rev]

k_a: 螺纹起始螺距 (在 I、J、K 下编程) [mm/rev]

l_G: 螺纹长度 [mm]

根据需要增加还是降低螺距，相应选择 G34 或 G35 来编写 F 值。

螺纹长度 l_G、螺纹变化 F、起始螺距 k_a 为已知时，可通过等式换算如下推导出程序段末尾的螺纹螺距 k_e：

- 启用 G34（螺距增加）时：

$$k_e = \sqrt{k_a^2 + F * 2 * l_G}$$

- 启用 G35（螺距降低）时：

$$k_e = \sqrt{k_a^2 - F * 2 * l_G}$$

说明

若得出的结果是一个负数根值，则无法加工该螺纹！
在此情形下 NC 会发出报警 10605 或报警 22275。

应用

G34 和 G35 功能可用于实现自切螺纹。

示例

螺纹切削 G33，启用递减螺距 G35

程序代码	注释
N1608 M3 S10	; 主轴转速
N1609 G0 G64 Z40 X216	; 运行到起点
N1610 G33 Z0 K100 SF=R14	; 恒定螺距（100 mm/rev）螺纹
N1611 G35 Z-220 K100 F17.045455	; 螺距递减 17.045455 mm/rev2
	; 程序段结束处螺距：50 mm/rev
N1612 G33 Z-240 K50	; 无急动运行螺纹程序段
N1613 G0 X218	
N1614 G0 Z40	
N1616 M17	

程序段预处理期间监控

在程序段预处理中可提前识别出引起螺纹轴过载的 G34 螺距变化和引起轴停止的 G35 螺纹变化， 并输出相应报警：报警 10604“螺距增加量过高”或报警 10605“螺距降低量过高”。

在特定的实际应用中，可能需要在螺纹切削中进行主轴转速的补偿。 在此情形下，操作人员须确保螺纹轴的速度在允许范围内。

为此可采用以下设置抑制报警 10604 和 10605 的输出：

MD11410 $MN_SUPPRESS_ALARM_MASK 位 10 = 1

之后程序段预处理继续正常进行。

执行期间监控

执行（插补）螺纹程序段期间，对以下情形进行循环监控：

- 超出螺纹轴的最大速度
- G35 中螺纹轴达到停止状态

系统会输出以下报警作为响应：

- 报警 22270“达到螺纹轴的最大速度”，或
- 报警 22275“螺纹轴达到速度零”

18.2.3.3 G33、G34 和 G35 中轴的加速度特性

在通过 G33、G34 或 G35 进行螺纹切削时，进给轴的加速度特性可通过以下通道专用设定数据设置：

SD42010 $SC_THREAD_RAMP_DISP[<n>] = <值>

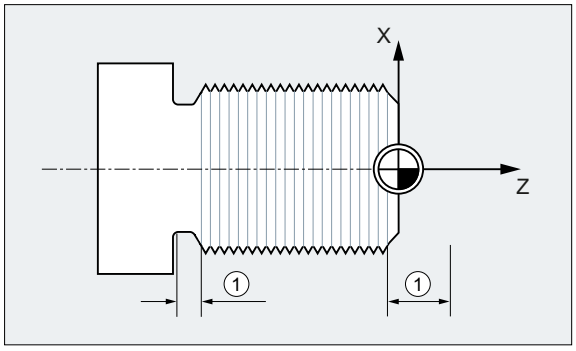
<n>	含义	<值>	含义
0	螺纹导入时的加速度特性	< 0 （缺省设置：-1）	在螺纹导入时，螺纹轴根据当前的 BRISK / SOFT 编程加速。
		= 0	在螺纹导入时，螺纹轴跃变式加速（≙ BRISK）。
		> 0	设定最大导入距离。 也可在零件程序中通过地址 DITS 设定最大导入距离（参见“G33、G34、G35 中可编程的导入和导出行程（DITS、DITE） (页 1541)”）。在主处理中切换至该程序段时，系统将编写的导入距离接收至设定数据。 **提示** 距离过短可能会导致轴的加速度过载。
1	螺纹导出时的加速度特性	< 0 （缺省设置：-1）	在螺纹导出时，螺纹轴根据当前的 BRISK / SOFT 编程制动。
		= 0	在螺纹导出时，螺纹轴跃变式制动（≙ BRISK）。
		> 0	设定最大导出距离。 也可在零件程序中通过地址 DITE 设定最大导出距离（参见“G33、G34、G35 中可编程的导入和导出行程（DITS、DITE） (页 1541)”）。在主处理中切换至该程序段时，系统将编写的导出距离接收至设定数据。 **提示** 距离过短可能会导致轴的加速度过载。

<n>	含义	<值>	含义
2	在螺纹链内的螺纹过渡处的加速度特性	= -1（缺省设置）	在两个螺纹程序段之间的角部处进行平滑，以遵循动态特性极限。 在机床的动态特性允许硬过渡的情况下不进行平滑，例如基于轴速度跃变的过载系数的有效性（MD32310 $MA_MAX_ACCEL_OVL_FACTOR）。
		= 0	非平滑地经过两个螺纹程序段之间的角部，轴按照控制回路的设定跟随。
		> 0	预留

18.2.3.4 G33、G34、G35 中可编程的导入和导出行程（DITS、DITE）

可在零件程序中通过地址 DITS 和 DITE 设定螺纹的导入和导出行程。

螺纹轴会在设定的行程内进行加速或制动。



① 导入或导出行程，根据加工方向

较短的导入行程

螺纹导入处有套环，给刀具启动斜坡留下的空间较小。
因此必须通过 DITS 缩短斜坡的长度。

较短的导出行程

由于连接至螺纹导出段，刀具减速斜坡的空间不够，可能会导致工件和刀沿之间的碰撞危险。可通过 DITE 设定较短的刀具减速斜坡。不过机械惯性的存在使得仍有发生碰撞的危险。

解决方法：编程更短的螺纹，减少主轴转速。

说明

DITE 在螺纹末端处作为精磨间距生效。从而使轴的运行平稳改变。

影响

编写的导入和导出行程只会提升轨迹上的加速度。若两个行程的其中一个大于螺纹轴启用生效加速度时所需的行程，那么螺纹轴会以最大加速度加速或制动。

句法

DITS=<值> DITE=<值>

含义

DITS：	确定螺纹导入行程
DITE：	确定螺纹导出行程
<值>：	DITS 和 DITE 下只编写行程，不编写位置！ 编写的导入/导出行程会依据当前生效的单位制（英制或公制）接受处理。

示例

程序代码	注释
...	
N40 G90 G0 Z100 X10 SOFT M3 S500	
N50 G33 Z50 K5 SF=180 DITS=1 DITE=3	；在 Z=53 时开始精磨。
N60 G0 X20	

其他信息

SD42010 $SC_THREAD_RAMP_DISP

在主处理中切换至包含 DITS 和/或 DITE 的程序段时，编写的导入或导出行程会被接收至设定数据 SD42010 $SC_THREAD_RAMP_DISP (页 1540)：

- SD42010 $SC_THREAD_RAMP_DISP[0] = 编写的 DITS 值
- SD42010 $SC_THREAD_RAMP_DISP[1] = 编写的 DITE 值

若在第一个螺纹程序段前未编写导入或导出行程，则会启动设定数据中的当前值。

通道/BAG/程序结束复位后的特性

通道/BAG/程序结束复位后，被 DITS 和/或 DITE 覆写的 SD42010 值仍然保持生效。

热启动后的特性

热启动时，设定数据恢复为被 DITS 和/或 DITE 覆写之前生效的值（标准特性）。

热启动后，如果也要将以 DITS 和 DITE 编程的值设为生效，则必须在机床数据 MD10710 $MN_PROG_SD_RESET_SAVE_TAB 中列出设定数据 SD42010 $SC_THREAD_RAMP_DISP：

MD10710 $MN_PROG_SD_RESET_SAVE_TAB[<n>] = 42010

导入和/或导出行程非常短时的特性

如果导入和/或导出行程非常短，则螺纹轴的加速度要大于配置值。这会导致轴因加速而过载。

在导入螺纹时将会发出报警 22280 “编程的导入行程过短”（在机床数据 MD11411 $MN_ENABLE_ALARM_MASK 中进行相应的设置）。报警只是一种信息，它对于零件程序的执行没有影响。

18.2.3.5 螺纹切削期间快速返回

功能

“螺纹切削时的快速返回”功能可在以下情况下中断螺纹切削，并且不会损坏设备：

- NC 停止（NC/PLC 接口信号）
- 隐性触发 NC 停止的报警
- 接通快速输入
  文档
  编程手册之工作准备分册，章节“从轮廓快速退刀”

返回运行的编程可通过：

- 返回位移和返回方向（相对）
- 返回位置（绝对）

说明

攻丝

不能在攻丝（G331/G332）时使用快速返回功能。

编程

句法

使能快速返回，通过返回位移和返回方向编写返回运动：
```
G33 ... LFON DILF=<值> LFTXT/LFWP ALF=<值>
```

使能快速返回，通过返回位置编写返回运动：

```
POLF[<轴名称>]=<值> LFPOS
POLFMASK/POLFMLIN(<轴名称 1>,<轴名称 2>,...)
G33 ... LFON
```

禁用螺纹切削快速返回：
```
LFOF
```

含义

LFON:	使能螺纹切削快速返回（G33）
LFOF:	禁用螺纹切削快速返回（G33）
DILF=:	确定返回行程的长度 可在零件程序中编程 DILF 来修改机床数据（MD21200 $MC_LIFTFAST_DIST）中预设的值。 **提示：** NC 复位后，设置的机床数据值总是生效。
LFTXT	使用 G 指令 LFTXT 和 LFWP 与 ALF 一起对返回方向进行控制。
LFWP:	LFTXT: 执行返回运行的平面通过轨迹切线和刀具方向计算（缺省设置）。 LFWP: 执行返回运行的平面是有效的工作平面。

ALF=： 在返回平面中，使用 ALF 以不连续的角度编程返回方向。

使用 LFTXT 时，通过 ALF=1 确定返回方向为刀具方向。

使用 LFWP 时，工作平面中的方向被分配如下：

- G17（X/Y 平面）
  ALF=1；以 X 方向返回
  ALF=3；以 Y 方向返回
- G18（Z/X 平面）
  ALF=1；以 Z 方向返回
  ALF=3；以 X 方向返回
- G19（Y/Z 平面）
  ALF=1；以 Y 方向返回

ALF=3；以 Z 方向返回

文档：

ALF 编程的相关内容请参见编程手册工作准备中的章节“从轮廓快速退刀时的运行方向”。

LFPOS： 将使用 POLFMASK 或 POLFMLIN 指定的轴退回到使用 POLF 编程的绝对轴位置

POLFMASK： 使能轴 (<轴名称 1>,<轴名称 1>,...)，使它独立退回至绝对位置

POLFMLIN： 使能轴，使它退回至线性关联的绝对位置。

提示：

受所有参与轴的动态特性的影响，到达退刀位置时不是总能建立起线性关联。

POLF[]： 确定目录中给定的几何轴或加工轴的绝对返回位置

有效性： 模态有效

=<值>： 在几何轴上，赋值被视为工件坐标系中的位置（WCS），在加工轴上，赋值被视为机床坐标系中的位置（MCS）。

赋值也可编程为增量值：

=IC<值>

<轴名称>： 几何轴或加工轴的名称

说明

LFON 或 LFOF 总是可以编程，但只在螺纹切削（G33）时运用。

说明

POLF 和 POLFMASK/POLFMLIN 不仅仅限于在螺纹切削时使用。

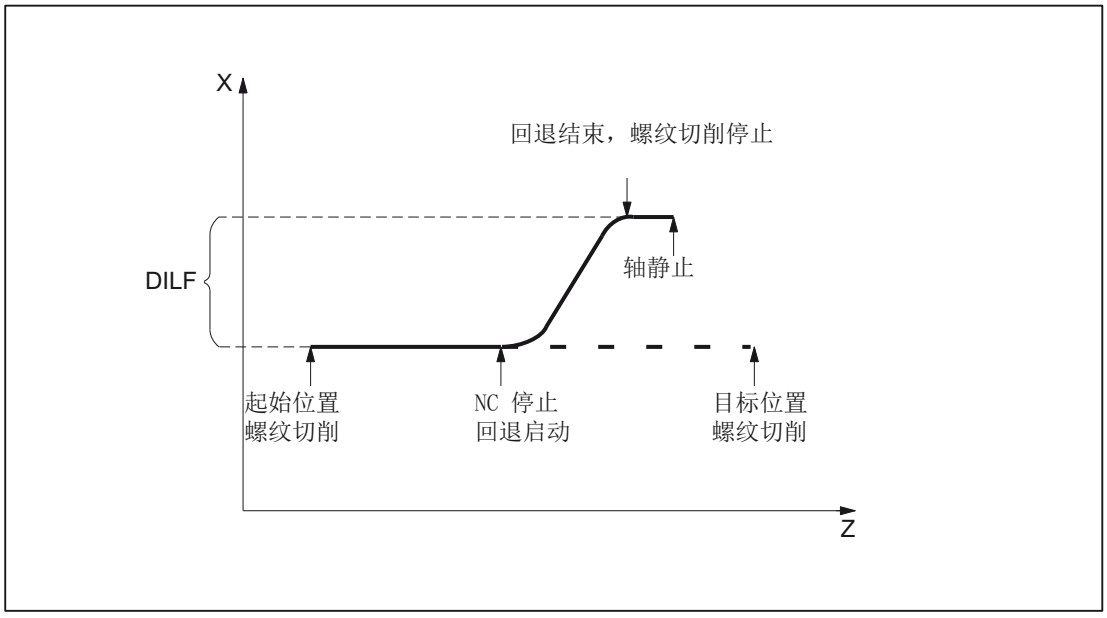


图 18-1　通过返回运动中断 G33

返回运动的动态特性

返回运动会以最大动态特性执行：

- MD32000 $MA_MAX_AX_VELO[<轴>]（速度）
- MD32300 $MA_MAX_AX_ACCEL[<轴>]（加速度）
- MD32431 $MA_MAX_AX_JERK[<轴>]（急动）

示例

程序代码	注释
N55 M3 S500 G90 G18	; 设置生效的加工平面
...	
N65 MSG （“螺纹切削”）	
MM_THREAD:	
N67 $AC_LIFTFAST=0	; 在螺纹开始前复位。
N68 G0 Z5	
N69 X10	
N70 G33 Z30 K5 LFON DILF=10 LFWP ALF=7	; 使能螺纹切削快速返回。
	; 返回行程=10 mm
	; 返回平面：Z/X（G18）
	; 返回方向 -X（ALF=3：返回方向 +X）

```
N71 G33 Z55 X15
N72 G1                                  ; 取消螺纹切削。
N69 IF $AC_LIFTFAST GOTOB MM_THREAD     ; 螺纹切削中断时。
N90 MSG("")
...
N70 M30
N55 M3 S500 G90 G0 X0 Z0
...
N87 MSG ("攻丝")
N88 LFOF                                ; 在攻丝前取消快速返回
N89 CYCLE...                            ; G33 攻丝循环。
N90 MSG("")
...
N99 M30
```

重新上电/复位后的特性

上电和复位后以下设置生效：

- Löschstellungen für die Rückzugsbewegung (LFON / LFOF) und Rückzugsrichtung (LFTXT / LFWP):MD20150 $MC_GCODE_RESET_VALUES
- 返回行程：MD21200 $MC_LIFTFAST_DIST

18.2.3.6 球螺纹（G335，G336）

功能

借助 G 指令（G335 和 G336）可以车削球螺纹（即非柱形螺纹）。该功能应用在机床上因自重而下垂的超大型部件的加工。如果使用平行于轴的螺纹，会导致部件中心的螺纹牙过小。使用球螺纹可弥补该缺陷。

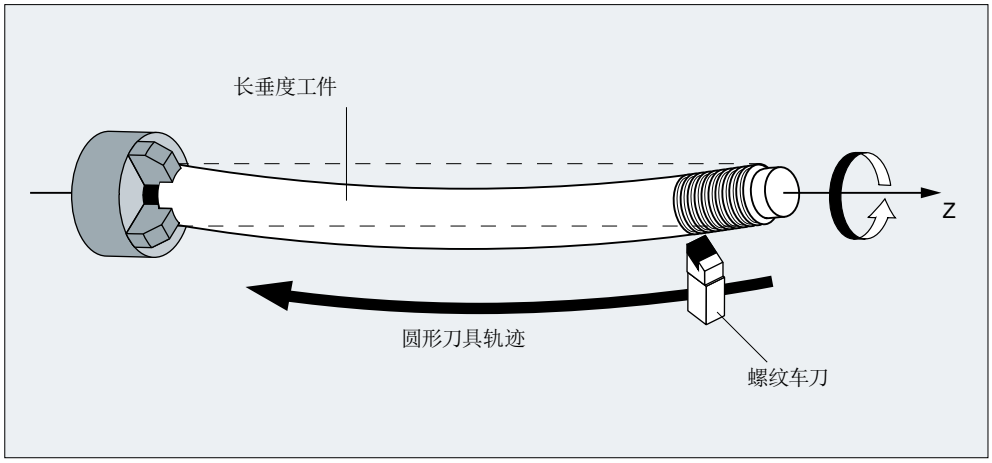


图 18-2 车削球螺纹

编程

车削球螺纹可通过 G335 或 G336 进行编程：

G335:	以**顺时针**圆弧刀具轨迹车削球螺纹
G336:	以**逆时针**圆弧刀具轨迹车削球螺纹

首先如同编写一个直螺纹一样，通过参数 I、J 和 K 指定轴终点和螺距。

此外还要指定一段圆弧。该圆弧可如同 G2/G3 一样，通过圆心、半径、张角或中间点坐标进行编程。使用圆心编程球螺纹时要注意以下几点：因为在进行螺纹切削需要指定 I、J 和 K 以计算出螺距，因此在使用圆心编程时必须用 IR=...、JR=...和 KR=... 指定圆弧参数。

IR=...:	X 方向上的圆心直角坐标
JR=...:	Y 方向上的圆心直角坐标
KR=...:	Z 方向上的圆心直角坐标

说明

IR、JR 和 KR 是球螺纹插补参数的缺省名称，可通过机床数据 MD10651 $MN_IPO_PARAM_THREAD_NAME_TAB 设置。

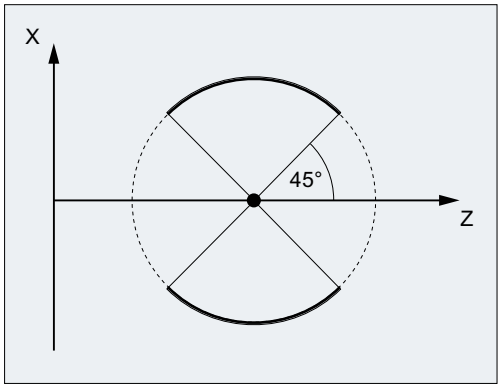
或者也可以指定起始点偏移 SF。

句法

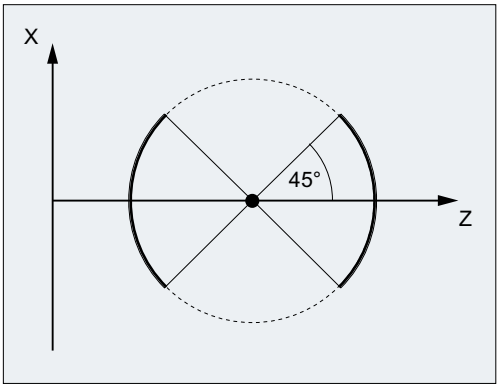
编程球螺纹的句法具有以下常规格式：
G335/G336 <轴目标点坐标> <螺距> <圆弧> [<起始点偏移>]

允许的圆弧范围

G335/G336 中编程的圆弧必须在规定的范围内，即整个圆弧段都必须包含在指定的螺纹轴区间内（I、J 或 K）。



Z 轴允许的区域（通过 K 编程螺距）



X 轴允许的区域（通过 I 编程螺距）

下图所示的圆弧段是**不**允许的：

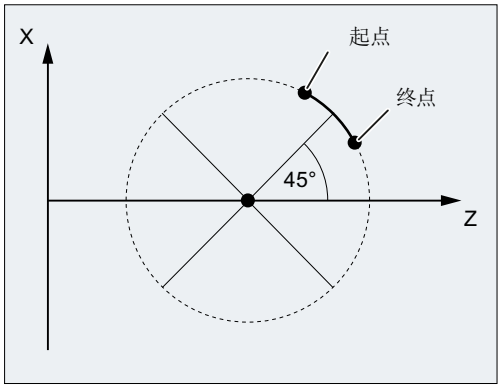


图 18-3　　球螺纹：不允许的范围

前提条件

框架

在框架激活时也可使用 G335 和 G336。然而还是得注意遵守基本坐标系（BCS）中允许的圆弧范围。

特性

以下情形下 G335/G336 的特性：

- 上电/断电
- 运行方式切换
- NCK / BAG / 通道 / 零件程序结束复位
- 程序段搜索 / REPOS / ASUB
- 报警 / 急停 / 故障情形

对应 G33/G34/G35 的特性。
无特定限制。

示例

示例 1：使用终点和圆心编程顺时针球螺纹

程序代码	注释
N5 G0 G18 X50 Z50	; 逼近起始点。
N10 **G335 Z100** K=3.5 **KR=25 IR=-20** SF=90	; 车削顺时针球螺纹。

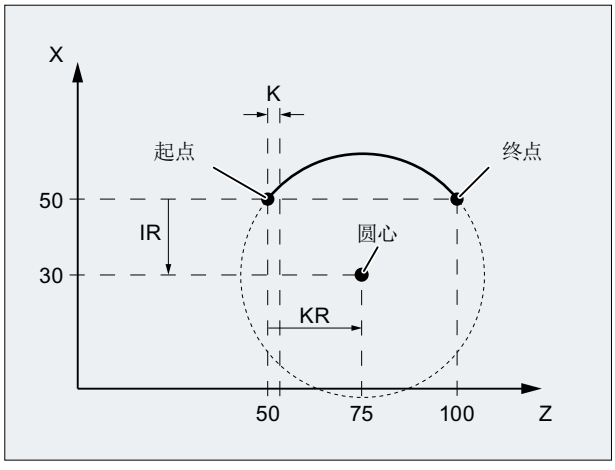


图 18-4　使用终点和圆心编程顺时针球螺纹

示例 2：使用终点和圆心编程逆时针球螺纹

程序代码	注释
N5 G0 G18 X50 Z50	; 逼近起始点。
N10 **G336 Z100** K=3.5 **KR=25 IR=20** SF=90	; 车削逆时针球螺纹。

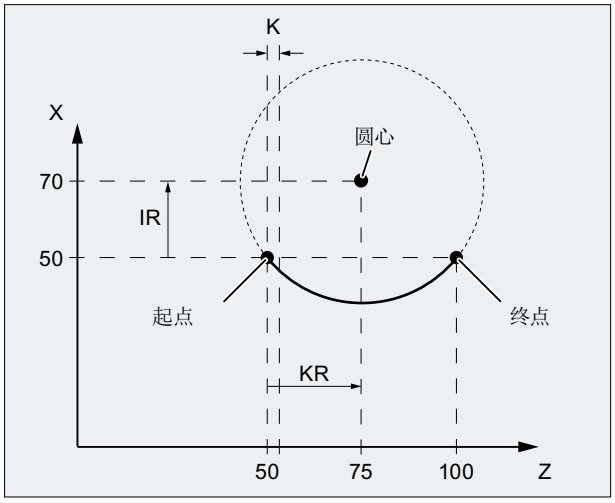


图 18-5　　使用终点和圆心编程逆时针球螺纹

示例 3：使用终点和半径编程顺时针球螺纹

程序代码
N5 G0 G18 X50 Z50
N10 **G335 Z100** K=3.5 **CR=32** SF=90

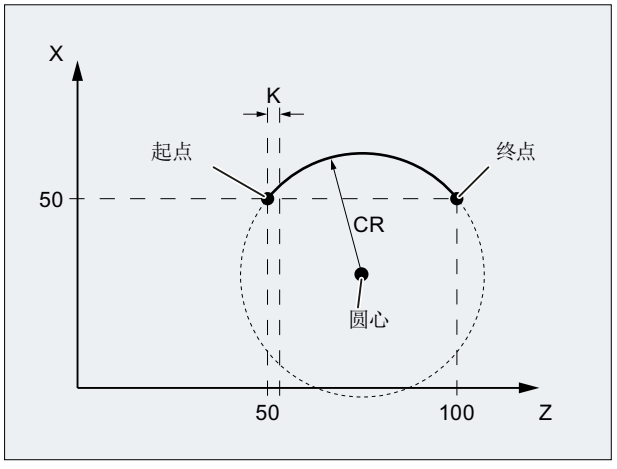


图 18-6　　使用终点和半径编程顺时针球螺纹

示例 4：使用终点和张角编程顺时针球螺纹

程序代码

```
N5 G0 G18 X50 Z50
N10 G335 Z100 K=3.5 AR=102.75 SF=90
```

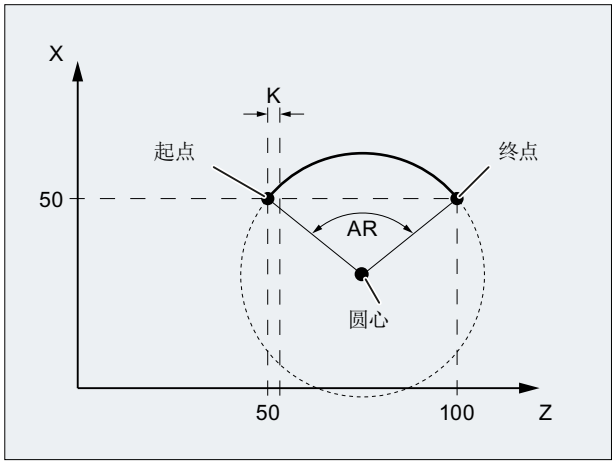


图 18-7　　使用终点和张角编程顺时针球螺纹

示例 5：使用圆心和张角编程顺时针球螺纹

程序代码

```
N5 G0 G18 X50 Z50
N10 G335 K=3.5 KR=25 IR=-20 AR=102.75 SF=90
```

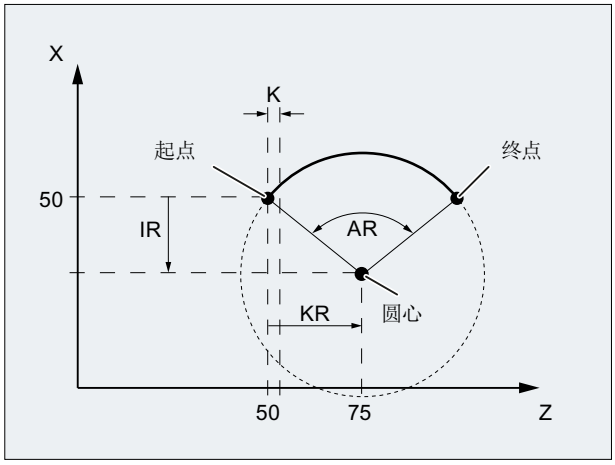


图 18-8　　使用圆心和张角编程顺时针球螺纹

示例 6：使用终点和中间点编程顺时针球螺纹

程序代码
N5 G0 G18 X50 Z50
N10 **G335 Z100** K=3.5 **I1=60 K1=64**

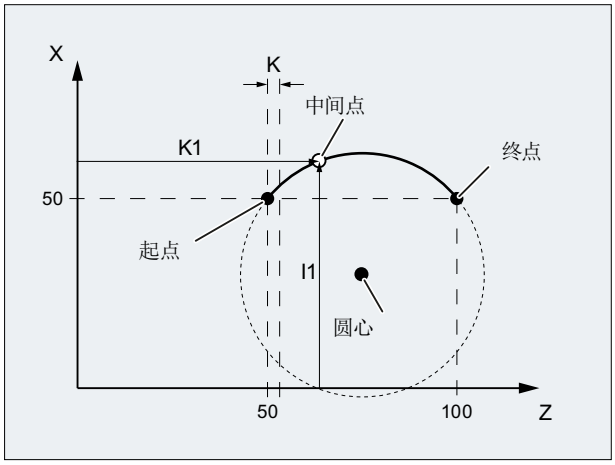


图 18-9　　使用终点和中间点编程顺时针球螺纹

18.2.4　刚性攻丝（G331、G332）中的进给率

功能

通过 G331（攻丝）和 G332（攻丝回退）可进行刚性攻丝。

轨迹进给率

在使用 G331/G332 时，用于攻丝相关的轴的轨迹进给率 F 由编程的主轴转速 S 和编程的螺距大小决定：

F [mm/min] = S [rpm] * 螺距 [mm/rev]

前提条件

刚性攻丝的前提是，主轴带位移测量系统并处于**位置闭环控制**中。

机床数据

- 防止停止事件
  通过机床数据确定 G331 / G332 生效时的停止特性：
  MD11550 $MN_STOP_MODE_MASK, 位 **<x> = <**值**>**

位	值	含义
0	0	G331 / G332 生效时，轨迹运动或停留期间（G4）**不会**停止。
	1	G331 / G332 生效时，在轨迹控制运行中断（例如：G60 或 G4）时停止。

如果要在任何情况下都能防止 G331 / G332 生效时运动停止，则必须通过指令 DELAYFSTON 和 DELAYFSTOF 确定停止延迟区。

文档：
功能手册之基本功能；章节 “通过停止延迟区控制停止事件 (页 628)”

说明

单程序段

如果在停止延迟区中激活单程序段，NC 会在停止延迟区外的第一个程序段处停止。如果在停止延迟区前已经选择了单程序段，NC 会在每个程序段交界处停止，在停止延迟区中也是如此！停止延迟区已取消。

倍率变化

如果在停止延迟区**前**调整倍率，倍率会在停止延迟区**中**生效。
如果在停止延迟区**中**调整倍率，倍率会在停止延迟区**后**生效。

- 倍率
  可通过以下设置刚性攻丝时哪个倍率生效：
  MD12090 $MN_OVR_FUNCTION_MASK, 位 **<x> = <**值**>**

位	值	含义
0	0	在 G331/G332 中，**主轴倍率**生效 根据以下机床数据中的设置，倍率可基于编写的主轴转速，或者基于配置的主轴转速限制： MD12080 $MN_OVR_REFERENCE_IS_PROG_FEED
	1	在 G331/G332 中，**轨迹倍率**生效 根据以下机床数据中的设置，倍率可基于编写的轨迹进给率，也可基于配置的轨迹进给限制： MD12082 $MN_OVR_REFERENCE_IS_MIN_FEED **提示** 当限值低于编程的轨迹进给率时才生效。

边界条件

无效补偿

以下补偿在刚性攻丝时无效：

- 可编程的轨迹进给倍率 OVR
- 快速移动倍率

18.2.5 带补偿夹具攻丝（G63）中的进给率

功能

G63 是攻丝的子功能，采用带补偿夹具的螺纹丝锥进行。 该功能中不需要使用编码器（位移测量系统）。

转速 S、进给率 F、螺距

执行 G63 时，必须为主轴编写转速 S，为进给轴（加工螺纹长度的轴）编写进给率 F。

进给率须由用户通过转速 S 和螺距来计算：

进给率 F [mm/min] = 转速 S [rev/min] * 螺距 [mm/rev]

文档

G63 的更多信息参见：编程手册之基本原理分册。

18.2.6 FGROUP 和 FGREF

编程

在刀具、工件或两者都被回转轴移动的加工过程中（例如：激光加工旋转的中空管），需要按照通用的方式在 F 值下作为轨迹进给率对生效的加工进给率进行编程。

为此，必须为参与运行的每根回转轴给定一个有效的半径（参考半径）。 这可通过编程模态有效的 NC 地址实现：
FGREF[<回转轴>]=<参考半径>

参考半径的单位取决于 G70/G71/G700/G710 的设置。

为了参与计算轨迹进给率，和以前一样，所有一同生效的轴都必须在 FGROUP 指令中进行定义。

为了在不进行 FGREF 编程的情况下与特性保持兼容，系统启动后及复位时，“1 度 = 1 毫米”的换算生效。

也就是说参考半径：

FGREF = 360 毫米/(2π) = 57.296 毫米

该缺省设置与生效的基本单位制 (MD10240 $MN_SCALING_SYSTEM_IS_METRIC) 和当前生效的 G70/G71/G700/G710 设置无关。

FGROUP 中回转轴进给率值的特性：

程序代码

```
N100 FGROUP(X,Y,Z,A)
N110 G1 G91 A10 F100
N120 G1 G91 A10 X0.0001 F100
```

在上述程序中，程序段 N110 中编程的 F 值是回转轴进给率，单位为度/分，而在程序段 N120 中编程的进给率值取决于当前生效的英寸/公制设置，要么是 100 英寸/分，要么是 100 毫米/分。

注意

不同的单位制

如果只在程序段中编程了回转轴，FGREF 值也会生效。 此时，常用的以度/分为单位的 F 值插补只适用于参考半径符合 FGREF 缺省设置的情况。

- 使用 G71/G710 时：FGREF[A]=57.296
- 使用 G70/G700 时：FGREF[A]=57.296/25.4

示例

下面的例子说明了 FGROUP 对轨迹行程和轨迹进给率的作用。 变量 $AC_TIME 包含从程序段开始的以秒为单位的时间。 它只能在同步动作中使用。

程序代码	注释
N100 G0 X0 A0	
N110 FGROUP(X,A)	
N120 G91 G1 G710 F100	; 进给率 = 100 毫米/分或 100 度/分
N130 DO $R1=$AC_TIME	
N140 X10	; 进给率 = 100 毫米/分，轨迹行程 = 10 毫米，R1 = 约 6 秒
N150 DO $R2=$AC_TIME	
N160 X10 A10	; 进给率 = 100 毫米/分，轨迹行程 = 14.14 毫米，R2 = 约 8 秒
N170 DO $R3=$AC_TIME	
N180 A10	; 进给率 = 100 度/分，轨迹行程 = 10 度，R3 = 约 6 秒
N190 DO $R4=$AC_TIME	
N200 X0.001 A10	; 进给率 = 100 毫米/分，轨迹行程 = 10 毫米，R4 = 约 6 秒
N210 G700 F100	; 进给率 = 2540 毫米/分或 100 度/分
N220 DO $R5=$AC_TIME	
N230 X10	; 进给率 = 2540 毫米/分，轨迹行程 = 254 毫米，R5 = 约 6 秒
N240 DO $R6=$AC_TIME	
N250 X10 A10	; 进给率 = 2540 毫米/分，轨迹行程 = 254.2 毫米，R6 = 约 6 秒
N260 DO $R7=$AC_TIME	
N270 A10	; 进给率 = 100 度/分，轨迹行程 = 10 度，R7 = 约 6 秒
N280 DO $R8=$AC_TIME	
N290 X0.001 A10	; 进给率 = 2540 毫米/分，轨迹行程 = 10 毫米，R8 = 约 0.288 秒
N300 FGREF[A]=360/(2*$PI)	; 1 度 = 1 英寸，通过有效的半径进行设置。
N310 DO $R9=$AC_TIME	
N320 X0.001 A10	; 进给率 = 2540 毫米/分，轨迹行程 = 254 毫米，R9 = 约 6 秒
N330 M30	

诊断

读取参考半径

回转轴参考半径的值可由以下系统变量读取：

- 在同步动作或零件程序（会触发预处理停止）中写入以下系统变量，状态显示在操作界面中：

  $AA_FGREF[<轴>]　　当前主运行值

- 在不带预处理停止的零件程序中，通过系统变量：

  $PA_FGREF[<轴>]　　编程值

如果未编程值，则在两个回转轴变量中读取缺省值 360 毫米/(2π) = 57.296 毫米（1 度对应 1 毫米）。

在线性轴上，两个变量中的值始终为 1 毫米。

读取影响速度的轨迹轴

参与到轨迹插补的轴可由以下系统变量读取：

- 在操作界面、同步动作或带有预处理停止的零件程序中显示，通过系统变量：

$AA_FGROUP[<轴>]	当指定的轴通过初始设置或 FGROUP 编程而影响当前主运行程序段中的轨迹速度时，变量提供值“1”。 无影响时，变量提供值“0”。
$AC_FGROUP_MASK	提供一个使用 FGROUP 编程、会影响轨迹速度的通道轴的位码。

- 在不带预处理停止的零件程序中，通过系统变量：

$PA_FGROUP[<轴>]	当指定的轴通过初始设置或 FGROUP 编程而影响轨迹速度时，变量提供值“1”。无影响时，变量提供值“0”。
$P_FGROUP_MASK	提供一个使用 FGROUP 编程、会影响轨迹速度的通道轴的位码。

18.3 定位轴的进给率（FA）

功能

轴专用进给率 FA 用于编写定位轴的速度。

FA 为模态有效。

其进给类型始终为 G94。

说明

不会超出最大轴速度（MD32000 $MA_MAX_AX_VELO）。

编程

每个零件程序段最多可编写 5 个轴专用进给率。

句法：

FA[<定位轴>]=<进给轴>

<定位轴>:	通道轴的名称 (MD20080 $MC_AXCONF_CHANAX_NAME_TAB)	
<进给值>:	进给速度	
	取值范围：	0.001…999 999.999 mm/min, degrees/min 或者 0.001…39 999.9999 inch/min

缺省设置

若未编写轴专用进给率 FA，以下机床数据中的缺省设置将生效：

MD32060 $MA_POS_AX_VELO（定位轴速度的初始设置）

输出到 PLC

进给轴可输出到 PLC：

- 输入至通道专用 NC/PLC 接口：
  DB21, ... DBB158 - DBB193
- 输入至轴专用 NC/PLC 接口：
  DB31, ... DBB78 - DBB81

输出时间可通过以下机床数据定义：

MD22240 $MC_AUXFU_F_SYNC_TYP（F 功能的输出时间）

缺省设置下输出被抑制（MD22240 = 3），因为在连续路径运行中将 F 功能输出至 NC/PLC 接口可能会造成速度跃变。

更多信息参见功能说明“H2：输出到 PLC 的辅助功能 (页 429)”。

复位特性

程序结束或 NC 复位后的特性通过以下机床数据定义：

MD22410 $MC_F_VALUES_ACTIVE_AFTER_RESET（F 功能复位后保持生效）

值	含义
0	NC 复位后缺省值生效。
1	NC 复位后，最后一次编写的 FA 值生效。

18.4 进给控制

18.4.1 进给禁止和进给/主轴停止

功能

在“进给禁止”或者“进给/主轴停止”状态下，轴将遵循制动特性曲线进行减速，直至达到停止状态。除螺纹加工 G33 外，遵循编写的轮廓。

通道专用进给禁止

借助以下通道专用 NC/PLC 接口信号，在所有运行模式下使一个通道的所有轴（几何轴和辅助轴）进入静止状态：

DB21, ... DBX6.0（进给禁止）

JOG 模式下针对几何轴的通道专用“进给停止”

借助以下通道专用 NC/PLC 接口信号，以针对具体轴的方式，将通道的相应几何轴的运行停止：

- DB21, ... DBX12.3（进给停止，几何轴 1）
- DB21, ... DBX16.3（进给停止，几何轴 2）
- DB21, ... DBX20.3（进给停止，几何轴 3）

这些接口信号仅在 JOG 运行模式下生效。

针对机床轴的轴专用“进给停止/主轴停止”

借助以下轴专用 NC/PLC 接口信号将机床轴的运行停止：

DB31, ... DBX4.3（进给停止/主轴停止）

AUTO 和 MDA 运行模式

在 AUTO 和 MDA 运行模式下：

- 若使一个**轨迹轴**“进给停止”，则所有在当前程序段中运行的轴以及参与轨迹组的轴均进入静止状态。
- 若使一个**定位轴**“进给停止”，则仅该轴进入静止状态。

螺纹加工

螺纹加工	有效性
G33、G34、G35	生效 **注意** 轮廓偏差
G331、G332	生效
G63	轴：生效 主轴：**不**生效

针对机床轴的轴专用进给轴/主轴禁止

借助以下轴专用 NC/PLC 接口信号将机床轴的运行停止：

DB31, ... DBX1.3（进给轴/主轴禁止）

18.4.2 通过机床控制面板设置进给倍率

功能

“通过机床控制面板设置进给倍率”时，操作人员可在设备现场使用半分比值提升或降低轨迹进给率，调整将在机床上立即生效。此时调整的方法是将编写的进给率与 NC/PLC 上的对应倍率值相乘。

定位轴的进给率可针对轴修改。

主轴转速和切削速度（G96、G961）可通过“主轴倍率”修改。

进给率被修改时，系统仍将遵循加速度和速度限值。轨迹上不会出现轮廓错误。

可为轨迹轴和定位轴分别修改进给倍率。

倍率会对编写的值或限值（例如主轴转速限值 G26、LIMS）生效。

通道专用进给倍率和快速移动倍率

针对进给倍率和快速移动倍率，NC/PLC 接口中提供专属的使能信号和倍率系数：

DB21, ... DBX6.7（进给倍率生效）

DB21, ... DBB4（进给倍率）

DB21, ... DBX6.6（快速移动倍率生效）

DB21, ... DBB5（快速移动倍率）

倍率系数可通过 PLC 以二进制编码或格雷码设定。启用的格式通过以下机床数据告知 NC：

MD12020 $MN_OVR_FEED_IS_GRAY_CODE（轨迹进给倍率开关，格雷编码）

MD12040 $MN_OVR_RAPID_IS_GRAY_CODE（快速移动倍率开关，格雷编码）

下列固定指定关系适用于二进制码：

二进制码	十进制	倍率系数
00000000	0	0.00 ≙ 0%
00000001	1	0.01 ≙ 1%
00000010	2	0.02 ≙ 2%
00000011	3	0.03 ≙ 3%
00000100	4	0.04 ≙ 4%
...	...	...
01100100	100	1.00 ≙ 100%
...	...	...
11001000	200	2.00 ≙ 200%

采用格雷码时，必须将开关位置所对应的倍率系数输入以下机床数据：

MD12030 $MN_OVR_FACTOR_FEEDRATE [<n>]（轨迹进给倍率开关的权重系数）

MD12050 $MN_OVR_FACTOR_RAPID_TRA [<n>]（快速移动倍率开关的权重系数）

激活的进给倍率对当前指定给通道的所有轨迹轴生效。激活的快速移动倍率对当前指定给通道的、且启用快速移动运行的所有轴生效。

无快速移动倍率开关存在

若无独立的快速移动倍率开关，可在快速移动倍率和进给倍率之间进行切换。此时可通过 PLC 或操作面板前端选择启用哪一种倍率。快速移动倍率生效时，进给倍率值被限制在 100% 的范围内。

- 通过操作面板前端激活快速移动倍率时，PLC 基本程序
  - 将为快速移动选择的进给倍率传输至快速移动倍率的激活信号：
    DB21, ... DBX6.6 = DB21, ... DBX25.3
  - 将进给倍率值传输至快速移动倍率值：
    DB21, ... DBB5 = DB21, ... DBB4
- 通过 PLC 选择快速移动倍率时，PLC 用户程序
  - 将快速移动倍率的激活信号置位：
    DB21, ... DBX6.6 = 1
  - 将进给倍率值传输至快速移动倍率值：
    DB21, ... DBB5 = DB21, ... DBB4

通道专用进给倍率和快速移动倍率的生效范围：	
● 对于生效的 G33、G34、G35：	不生效
● 对于生效的 G63：	不生效
● 对于生效的 G331、G332：	不生效

轨迹进给倍率的基准速度

“通过机床控制面板设置轨迹进给倍率”的基准速度可通过以下机床数据设置为和标准值（= 编写的进给率）不同的值：

MD12082 $MN_OVR_REFERENCE_IS_MIN_FEED

轴专用进给倍率

对于每根定位轴，在 NC/PLC 接口中都有一个对应的使能信号，以及一个对应的用于进给倍率系数的字节：

DB31, ... DBX1.7（倍率生效）

DB31, ... DBB0（进给倍率）

倍率系数可通过 PLC 以二进制编码或格雷码设定。启用的格式通过以下机床数据告知 NC：

MD12000 $MN_OVR_AX_IS_GRAY_CODE（轴进给倍率开关，格雷编码）

下列固定指定关系适用于二进制码：

二进制码	十进制	倍率系数
00000000	0	0.00 ≙ 0%
00000001	1	0.01 ≙ 1%
00000010	2	0.02 ≙ 2%
00000011	3	0.03 ≙ 3%
00000100	4	0.04 ≙ 4%
...	...	...
01100100	100	1.00 ≙ 100%
...	...	...
11001000	200	2.00 ≙ 200%

采用格雷码时，必须将开关位置所对应的倍率系数输入以下机床数据：

MD12010 $MN_OVR_ FACTOR_AX_ SPEED [<n>]（轴进给倍率开关的权重系数）

轴专用进给倍率的生效范围：	
• 对于生效的 G33、G34、G35：	不生效
• 对于生效的 G63：	不生效 （倍率在 NC 中固定设为 100%）
• 对于生效的 G331、G332：	不生效 （倍率在 NC 中固定设为 100%）

主轴倍率

对于每根主轴，在 NC/PLC 接口中都有一个对应的使能信号，以及一个对应的用于主轴倍率系数的字节：

DB31, ... DBX1.7（倍率生效）

DB31, ... DBB19（主轴倍率）

倍率系数可通过 PLC 以二进制编码或格雷码设定。启用的格式通过以下机床数据告知 NC：

MD12060 $MN_OVR_SPIND_IS_GRAY_CODE（主轴倍率开关，格雷编码）

下列固定指定关系适用于二进制码：

二进制码	十进制	倍率系数
00000000	0	0.00 ≙ 0%
00000001	1	0.01 ≙ 1%
00000010	2	0.02 ≙ 2%
00000011	3	0.03 ≙ 3%
00000100	4	0.04 ≙ 4%
...	...	...
01100100	100	1.00 ≙ 100%
...	...	...
11001000	200	2.00 ≙ 200%

采用格雷码时，必须将开关位置所对应的倍率系数输入以下机床数据：

MD12070 $MN_OVR_FACTOR_SPIND_SPEED [<n>]（主轴倍率开关的权重系数）

“主轴倍率”的生效范围：	
• 对于生效的 G33、G34、G35：	生效
• 对于生效的 G63：	不生效
• 对于生效的 G331、G332：	生效

主轴倍率的基准

主轴倍率可以通过机床数据或设定数据限制的转速为基准，也可以编写的转速为基准。这通过以下机床数据设置：

MD12080 $MN_OVR_REFERENCE_IS_PROG_FEED（倍率基准速度）

对倍率系数的限制

启用二进制编码倍率系数时，可对用于轨迹进给、轴进给和主轴转速的最大倍率进行限制：

MD12100 $MN_OVR_FACTOR_LIMIT_BIN（二进制编码的倍率开关的限制）

倍率生效

倍率已使能时，通过机床控制面板设定的倍率值在所有运行方式下和所有机床功能中立即生效。

倍率无效

未激活倍率时，系统内部启用 100% 的倍率系数。NC/PLC 接口上的倍率系数不会被分析。

特例情形是二进制接口的零位置和格雷码接口的开关位置 1。输入到 NC/PLC 接口的倍率值会在这两个接口中接受分析。对于二进制接口，倍率系数始终为 0%。对于格雷码接口，机床数据中为开关位置 1 输入值将作为倍率值输出。其赋值应为“0”。

18.4.3 可编程的进给倍率

功能

借助“可编程的进给倍率”功能可通过零件程序修改轨迹轴和定位轴的速度。

编程

句法	含义
OVR=<值>	修改轨迹进给率 F 的倍率
OVRA[<轴>=<值>	修改定位进给 FA 的倍率

可编程范围为 0 - 200%

缺省设置：100%

有效性

NC/PLC 接口信号 DB21, ... DBB6（快速移动或进给倍率生效）和 DB31, ... DBX1.7（轴专用倍率生效）不以可编程的进给倍率为基准。这些信号取消时，可编程的进给倍率继续生效。

生效的倍率为“可编程的进给倍率”与“通过机床控制面板设置进给倍率 (页 1561)”的乘积。

“可编程的进给倍率”的缺省设置为 100%。

缺省设置在以下情形下生效：

- 未编写进给倍率，或
- 复位后，当以下机床数据未置位时：
  MD22410 $MC_F_VALUES_ACTIVE_AFTER_RESET（F 功能复位后继续生效）

说明

OVR 对 G33、G34、G35 无效。

18.4.4 空运行进给率

功能

在零件程序试车时，可使用空运行进给率功能来提升轨迹进给率，从而加快程序或部分程序的执行速度。此功能例如可应用于不加工工件的试运行。

激活

空运行进给率可在 AUTO 运行方式下选择，并通过 PLC 或操作面板前段激活。

通过操作面板前端激活时，以下接口信号会置位：
DB21, ... DBX24.6（空运行进给率已选择）
并通过 PLC 基本程序传输至接口信号：
DB21, ... DBX0.6（激活空运行进给率）

通过 PLC 选择时，须由 PLC 用户程序置位接口信号 DB21, ... DBX0.6（激活空运行进给率）。

有效性

接口信号“激活空运行进给率”置位的情形下，通过 SD42100 $SC_DRY_RUN_FEED 设置的进给值将替代零件程序中编写的进给值生效，生效方式取决于设定数据 SD42101 $SC_DRY_RUN_FEED_MODE 中的定义（参见参数设置）。

空运行进给率始终作为线性进给（G94）。

参数设置

激活试运行进给

该功能的激活时间点取决于以下机床数据中的设置：

MD10704 $MN_DRYRUN_MASK（激活空运行进给率）

值	含义
0	空运行进给率只允许在程序段末尾激活和取消（缺省设置）
1	空运行进给率也可在程序执行期间（在零件程序段中）激活。 **注意：** 在程序执行期间激活会引起控制系统内部的重整操作，此时轴会短暂停止。这肯能会对加工工件的表面质量产生影响。
2	空运行进给率可在不会触发轴停止的任意阶段激活和取消。但是其会随程序运行中一个“较晚”的程序段生效。

修改空运行进给率

空运行的进给率在以下设定数据中输入：

SD42100 $SC_DRY_RUN_FEED（空运行进给率）

该设定数据可通过操作面板前端在“参数”操作区修改。

选择被 NC 接受时，以下 NC/PLC 接口信号将置位：

DB21, ... DBX318.6（空运行进给率生效）

满足以下条件时，生效的空运行进给率会在操作面板前段的状态栏以“DRY”显示：

- 选择在程序停止时在程序段末尾进行，或
- 机床数据 MD10704 $MN_DRYRUN_MASK 在程序执行期间置“1”。

空运行进给率的生效方式

SD42100 中输入的空运行进给率的生效方式可通过以下设定数据设置：

SD42101 $SC_DRY_RUN_FEED_MODE

值	含义
0	将编写的进给率与 SD42100 中的空运行进给率相比较，之后启用其中较大的一个运行（缺省设置）。
1	将编写的进给率与 SD42100 中的空运行进给率相比较，之后启用其中较小的一个运行。
2	SD42100 中输入的空运行进给率直接生效，不考虑编写的速度。
3 -9	预留
10	除螺纹切削（G33、G34 和 G35）和攻丝（G331、G332 和 G63）外，与配置 0 相同。执行上述功能采用编写的进给率。
11	除螺纹切削（G33、G34 和 G35）和攻丝（G331、G332 和 G63）外，与配置 1 相同。执行上述功能采用编写的进给率。
12	除螺纹切削（G33、G34 和 G35）和攻丝（G331、G332 和 G63）外，与配置 2 相同。执行上述功能采用编写的进给率。

18.4.5 一个程序段中的多个进给值

功能

通过“一个程序段中的多个进给值”功能，可根据外部数字量和/或模拟量输入运行同步地激活一个 NC 程序段中的 6 个进给轴、暂停时间以及返回。

如果暂停时间或返回行程的输入被激活，那么轨迹轴或相关单个轴的剩余行程将被删除，并启动暂停时间或返回。

插补周期内会引导返回。

信号

此功能的输入信号汇总在一个输入字节中。该字节内的功能为固定指定。

表格 18-1　“一个程序段中多个进给值”功能的输入字节

	位							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
输入编号	E7	E6	E5	E4	E3	E2	E1	E0
进给地址	F7	F6	F5	F4	F3	F2	ST	SR

E7 至 E2：激活进给 F7 至 F2

E1：激活暂停时间 ST/STA（单位：s）

E0：激活返回运动 SR/SRA

信号的优先级

信号的询问顺序从 E0 开始升序排列。因此返回运行（SR）的优先级最高，进给率 F7 最低。

SR 和 ST 可以终止使用 F2 到 F7 激活的进给运行。
SR 也会终止 ST，即完整功能。

最高优先级信号决定当前的进给率。

最高优先级输入（F2 - F7）丢失时的特性可通过以下机床数据定义：

MD21230 $MC_MULTFEED_STORE_MASK（“一个程序段中多个进给值”功能的存储特性）

位	值	含义
2 ... 7	0	最高优先级输入丢失时，对应进给率不保留（缺省设置）
	1	位 2 ... 7 置位时，最高优先级输入信号选择的对应进给率（F2 至 F7）将保留，即使该输入信号丢失，且一个较低优先级的输入存在。

以下情形下满足程序段结束标准：

- 达到编写的终点位置
- 返回运动（SR）结束
- 暂停时间（ST）结束

硬件指令

可向“一个程序段中多个进给值”功能的输入字节指定最多两个 NC I/O 的数字量输入字节或比较器字节：

MD21220 $MC_MULTFEED_ASSIGN_FASTIN（为“一个程序段中多个进给值”功能指定 NC I/O 的输入字节），位 0 ... 15

此外还可取反输入位：

MD21220 $MC_MULTFEED_ASSIGN_FASTIN，位 16 ... 31

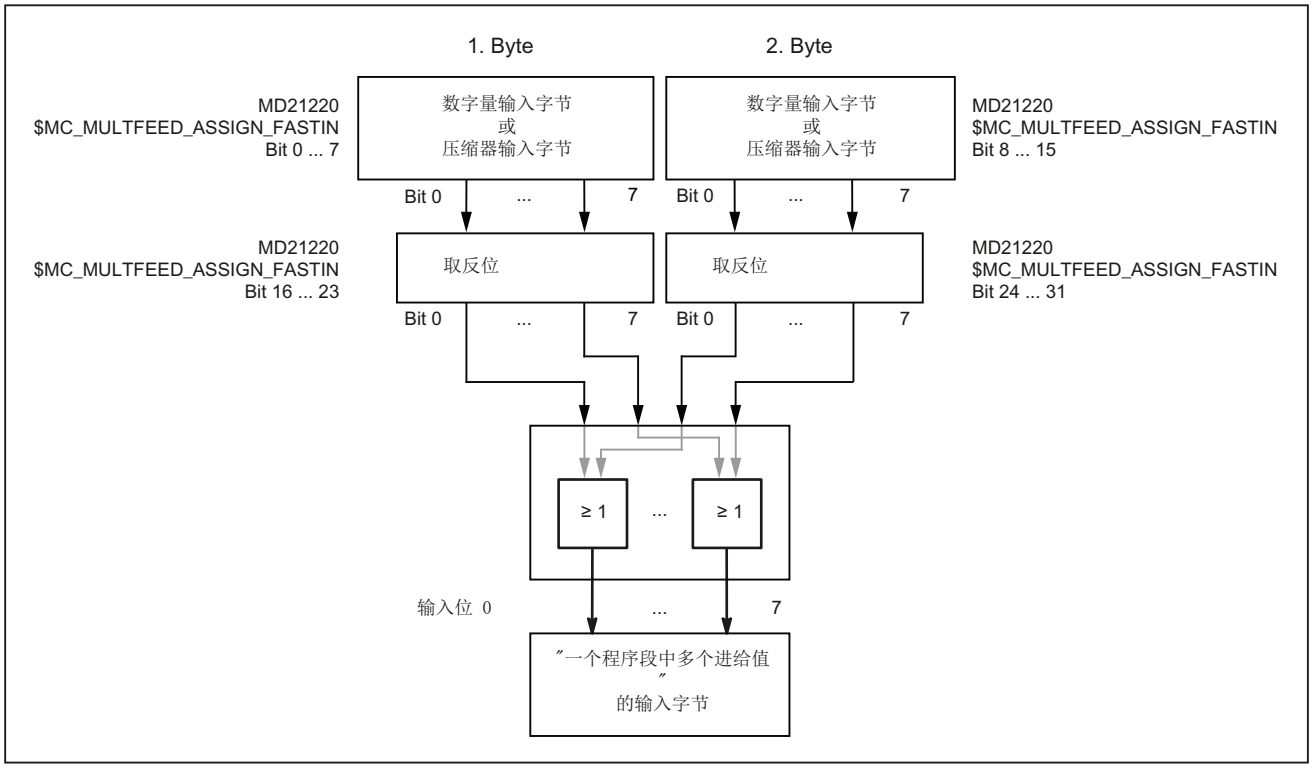


图 18-10　为“一个程序段中多个进给值”功能安排信号

数字量输入字节的安排以及比较器参数设置等相关信息参见：

文档：

功能手册之扩展功能分册；数字量和模拟量 NC I/O（A4）

编程

轨迹运行

在地址 F 中编程轨迹进给率，未出现输入信号时该值一直有效。其为模态有效。

除了轨迹进给率外，还可通过 F2=... 至 F7=... 在程序段中编写最多 6 个进给率。数字扩展给出了输入的位编号，改变它可以激活进给率：

示例：

```
F7=1000          ；7 与输入位 7 相对应
```

编写的值逐段生效。后续程序段中 F 下编写的轨迹进给率生效。

餐厅时间和返回行程在程序段中的附加地址下编写：

ST=...	暂停时间（磨削时为修光时间）
SR=...	返回行程

这些地址逐段生效。

轴运行

在地址 FA 中编程轴向进给率，未出现输入信号时该值一直有效。这些进给率模态有效。

也可以用 FMA[2,<轴>]=... 至 FMA[7,<轴>]=... 在程序段内给每个轴编程最多 6 个进给率。

第一个尖括号中给出了输入的位编号，改变它可以激活进给率：第二个注明了进给率适用于哪根轴。

示例：

```
FMA[3,Y]=1000          ；Y 轴的轴进给，对应输入位 3
```

FMA 下编写的值逐段生效。后续程序段中 FA 下编写的进给率生效。

可为各轴设定额外设定暂停时间和返回行程：

STA[<轴>]=...	轴暂停时间（修光时间）
SRA[<轴>]=...	轴向返回行程

尖括号中的内容用于指定修光时间或返回行程适用于的轴。

示例：

```
STA[X]=2.5          ; 轴 X 的修光时间为 2.5 s。
SRA[X]=3.5          ; 轴 X 的返回行程为 3.5（单位例如为 mm）。
```

这些地址逐段生效。

说明

返回行程

返回行程的单位基于当前生效的单位制（mm 或 inch）

返回行程的方向始终与当前运行方向相反。使用 SR/SRA 编写返回行程的绝对值。不需要编写正负号。

说明

POS 替代 POSA

如果基于一个外部输入为一个轴编程了进给率、暂停时间（修光时间）或返回行程，那么在该程序段中不能将该轴编程为 POSA 轴（超过程序段限制的定位轴）。

说明

状态询问

也可以为不同轴的同步指令询问输入状态。

说明

预读

程序段预读功能对一个程序段内的多个进给率有效。如此就可以使用程序段预读功能来限制当前进给率。

应用

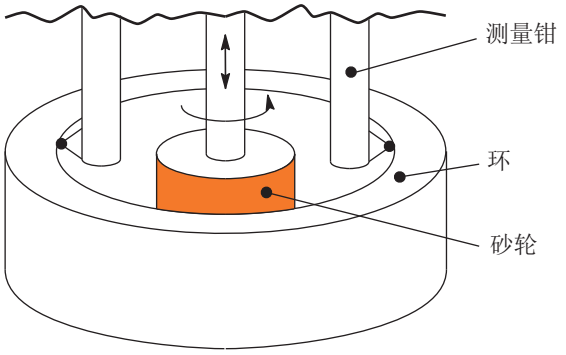
“一个程序段中多个进给值”功能主要应用于磨削工艺。但同时也不仅限于此功能。

其典型应用例如包括：

- 模拟量或数字量测量钳
  基于外部模拟量和数字量输入，可激活不同的进给值、一个暂停时间以及一个返回行程，同时阈值通过设定数据设定。
- 通过接近开关从深进给切换至工作进给。

示例

滚珠轴承环的内部磨削，其中实际直径通过测量钳采集，并根据阈值为粗加工、精加工和修光激活必要的进给值。测量钳位置也提供终点位置。因此程序段结束标准不仅取决于进给轴的轴位置，也由测量钳决定。




```
G00 X40 F=1000                                          ；起始位置
G01 X48 F=20 F7=5 F3=2.5 F2=0.5 ST=1.5 SR=0.5
                                                  回退
                                              修光时间
                                          修光进给率
                                      精加工进给率
                                  粗加工进给率
                              空磨进给率
```

针对 SINUMERIK 828D 的功能限制

MD21220=	1H	第 1 字节（X242 上 4 个位，4 个位通过接口）
	2H	第 2 字节（X252 上 4 个位，4 个位通过接口）
	102H	8 个位仅通过接口

就 SINUMERIK 828D 而言，由于硬件输入端有限，无法对整个字节进行响应。这意味着：

- 基于硬件，只能编写 4 个不同的进给率（X242 上 4 个位，或者 PPU 的 X252 上 4 个位）
- 剩余的 4 个位通过接口实现。
- “仅”使用非硬件输入的接口。这样便不再在插补周期中进给。

下表列出进给地址与输入或接口的对应关系：

进给地址	第 1 字节	$A_IN[x]	接口	第 2 字节	$A_IN[x]	接口
SR	0	1	DB2800.DBX1.0	0	9	DB2800.DBX1001.0
ST	1	2	DB2800.DBX1.1	1	10	DB2800.DBX1001.1
F2	2	3	DB2800.DBX1.2	2	11	DB2800.DBX1001.2
F3	3	4	DB2800.DBX1.3	3	12	DB2800.DBX1001.3
F4	4	5	DB2800.DBX1.4	4	13	DB2800.DBX1001.4
F5	5	6	DB2800.DBX1.5	5	14	DB2800.DBX1001.5
F6	6	7	DB2800.DBX1.6	6	15	DB2800.DBX1001.6
F7	7	8	DB2800.DBX1.7	7	16	DB2800.DBX1001.7

示例：

G90 G0 X600

POS[X]=700 FA[X]=1000

POS[X]=700 FA[**3**,X]=2000 -> （$A_IN[**4**]=**1** -> 故进给值为 2000，否则生效的进给值为 1000。）

调试：

4 个进给地址通过硬件端子，4 个通过接口

- -X242 = IN1 至 IN4 布线 (PIN 3 – 6) -> SR, ST, F2, F3
- 接口：DB2800.DBX1.4 – 1.7 = IN5 – IN8 -> F4, F5, F6, F7
- MD21220 =1H

18.4.6 固定进给值

功能

借助“固定进给值”功能，可通过机床数据激活已定义的固定进给值（最多 4 个）来取代编写的进给值或配置的 JOG 速度。

此功能可在 AUTO 和 JOG 运行方式下使用。

AUTO 运行方式下的特性

运行轮廓时启用激活的固定进给值代替编写的进给率。

JOG 运行方式下的特性

启动激活的固定进给值代替配置的 JOG 速度/JOG 快速移动速度来运行轴。 运行方向通过接口信号设定。

参数设置

固定进给值如下设置：

- 针对直线轴通过机床数据：
  MD12202 $MN_PERMANENT_FEED[<n>]
- 针对回转轴通过机床数据：
  MD12204 $MN_PERMANENT_ROT_AX_FEED[<n>]

其中 <n> = 0、1、2、3（对应固定进给值 1、2、3、4）

说明

固定进给值始终为线性进给值。 旋转进给率将在系统内部切换为线性。

激活

固定进给值通过 NC/PLC 接口激活：

- AUTO 运行方式下针对轨迹轴/几何轴，通过通道专用接口信号：
  DB21, ... DBX29.0（激活固定进给值 1）
  DB21, ... DBX29.1（激活固定进给值 2）
  DB21, ... DBX29.2（激活固定进给值 3）
  DB21, ... DBX29.3（激活固定进给值 4）
- JOG 运行方式下针对机床轴，通过轴专用接口信号：
  DB31, ... DBX3.2（激活固定进给值 1）
  DB31, ... DBX3.3（激活固定进给值 2）
  DB31, ... DBX3.4（激活固定进给值 3）
  DB31, ... DBX3.5（激活固定进给值 4）

前提条件

有效性

以下情形下，“固定进给值”功能不生效：

- 主轴上
- 定位轴上
- 攻丝

倍率＝ 0

倍率 = 0 时的运行特性取决于以下机床数据中的设置：

MD12200 $MN_RUN_OVERRIDE_0

DRF 偏移

选择固定进给值时无法激活 DRF 偏移。

18.4.7 可编程的进给特性曲线

功能

为了实现进给特性曲线的灵活设定，根据 DIN 66025 对进给特性曲线编程增加了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。

编程

可编写以下进给特性曲线：

- FNORM
  特性符合 DIN 66025（缺省设置）
  程序段中编写的 F 值在程序段的整个轨迹恒定设定，之后作为固定模态值。
- FLIN
  程序段中编写的 F 值从程序段开始处的当前值，直至程序段末尾沿轨迹线性运行，之后作为固定模态值。
- FCUB
  逐段编写的 F 值基于程序段终点通过样条连接。 样条开始时与前一进给设定呈切向，结束时与后一进给设定呈切向。 若一个程序段中缺少 F 地址，则为其使用最后编写的 F 值。
- FPO
  F 地址 [句法：`F=FPO(...,...,...)`] 通过一个多项式从当前值到编写其的程序段的末尾，结束值作为模态值。

参数设置

将 FLIN 和 FCUB 配合压缩器 COMPON 使用时，可为轨迹进给率定义一个公差：

MD20172 $MC_COMPRESS_VELO_TOL（启用压缩器时允许的最大轨迹进给偏差）

前提条件

FLIN/FCUB

通过 FLIN 或 FCUB 编写的轨迹速度曲线不和 G95 旋转进给率一起生效，也不和恒定切削速度 G96/G961、G97/G971 一起生效。

文档

可编程的进给特性曲线的更多信息参见：编程手册之工作准备分册。

18.4.8 倒角/倒圆进给率 FRC、FRCM

在从平面过渡至倒角/倒圆时，切屑条件可能会本质变化。 因此，为了达到所需的表面质量，需要为轮廓单元倒角/倒圆启用专用的、经过优化的进给值。

功能

倒角/倒圆的进给率可通过 NC 地址编写。

编程

句法：

```
... FRC/FRCM=<值>
```

含义：

FRC: **逐段生效**的倒角/倒圆进给率

FRCM: **模态生效**的倒角/倒圆进给率

<值>: 根据生效的进给类型，进给值如下解译：

- G94、G961、G971： 进给率，单位 mm/min, inch/min 或 °/min
- G95、G96、G97： 旋转进给率，单位 mm/rev 或 inch/rev

说明

只有在程序段中编程了倒圆/倒角，或者激活了 RNDM 时，FRC 才生效。

FRC 会覆盖当前程序段中的 F 值或 FRCM 值。

FRC 中编程的进给率必须大于零。

通过 FRCM=0 激活 F 中编程的进给用于倒角/倒圆。

参数设置

将倒角/倒圆工艺指定给前一/后一程序段

在 F 和 FRC/FRCM 程序段中，进给类型（G94、G95、G96、G961 ...）及其转换为的内部格式必须统一。 为此可启用以下机床数据：

MD20201 $MC_CHFRND_MODE_MASK（倒角/倒圆特性）

位	值	含义
0	0	倒角/倒圆工艺（进给率、进给类型、M 指令）由后一程序段决定（缺省设置）。
	1	倒角/倒圆工艺由前一程序段决定。

最大空程序段数量

倒角/倒圆生效时，两个包含运行信息的程序段间的空程序段（补偿级中无运行信息的程序段）的数量受限。 最大数量通过以下机床数据定义：

MD20200 $MC_CHFRND_MAXNUM_DUMMY_BLOCKS（倒角/倒圆中的空程序段）

前提条件

FLIN/FCUB

进给插补 FLIN 和 FCUB 不适用于倒角和倒圆。

G0

通过 G0 运行倒角时，FRC/FRCM 无效。 可根据 F 值编程，不会触发故障。

切换 G94 ↔ G95

如果编程了 FRCM，在 G94 ↔ G95 切换后必须对 F 和 FRCM 的值都进行重新编程。如果只重新编程了 F 值，且在进给类型转换前 FRCM > 0，则输出故障信息。

示例

示例 1：MD20201 位 0 = 0；从后一程序段接收进给值（缺省设置）

程序代码	注释
N10 G0 X0 Y0 G17 F100 G94	
N20 G1 X10 CHF=2	; 倒角 N20-N30，F=100 mm/min
N30 Y10 CHF=4	; 倒角 N30-N40，FRC=200 mm/min
N40 X20 CHF=3 FRC=200	; 倒角 N40-N60，FRCM=50 mm/min
N50 RNDM=2 FRCM=50	
N60 Y20	; 模态倒圆 N60-N70，FRCM=50 mm/min
N70 X30	; 模态倒圆 N70-N80，FRCM=50 mm/min
N80 Y30 CHF=3 FRC=100	; 倒角 N80-N90，FRC=100 mm/min
N90 X40	; 模态倒圆 N90-N100，F=100 mm/min（取消 FRCM）
N100 Y40 FRCM=0	; 模态倒圆 N100-N120，G95 FRC=1 mm/rev
N110 S1000 M3	
N120 X50 G95 F3 FRC=1	
...	
M02	

示例 2：MD20201 位 0 = 1；从前一程序段接收进给值（推荐设置！）

程序代码	注释
N10 G0 X0 Y0 G17 F100 G94	

程序代码	注释
N20 G1 X10 CHF=2	; 倒角 N20-N30，F=100 mm/min
N30 Y10 CHF=4 FRC=120	; 倒角 N30-N40，FRC=120 mm/min
N40 X20 CHF=3 FRC=200	; 倒角 N40-N60，FRC=200 mm/min
N50 RNDM=2 FRCM=50	
N60 Y20	; 模态倒圆 N60-N70，FRCM=50 mm/min
N70 X30	; 模态倒圆 N70-N80，FRCM=50 mm/min
N80 Y30 CHF=3 FRC=100	; 倒角 N80-N90，FRC=100 mm/min
N90 X40	; 模态倒圆 N90-N100，FRCM=50 mm/min
N100 Y40 FRCM=0	; 模态倒圆 N100-N120，F=100 mm/min
N110 S1000 M3	
N120 X50 CHF=4 G95 F3 FRC=1	; 倒角 N120-N130 使用 G95 FRC=1 mm/rev
N130 Y50	; 模态倒圆 N130-N140，F=3 mm/rev
N140 X60	
...	
M02	

18.4.9 逐段进给率 FB

功能

使用“逐段进给率”功能可为一个零件程序段设定一个专用的进给率。在此程序段之后，之前模态生效的轨迹进给率将重新生效。

编程

句法：

```
...FB=<值>
```

含义：

FB： 当前程序段的专用进给率

<值>： 根据生效的进给类型，进给值如下解译：

- G94、G961、G971： 进给率，单位 mm/min, inch/min 或 °/min
- G95、G96、G97： 旋转进给率，单位 mm/rev 或 inch/rev

说明

FB 中编程的进给率必须大于零。

若该程序段中未编写运行（例如：计算程序段），那么 FB 将不生效。

如果没有给倒角/倒圆编程明确的进给率，那么 FB 的值还适用于该程序段中的一个倒角/倒圆轮廓元素。

不能同时编程 FB 和 FD（启用进给倍率的通过手轮进行的手动运行）或者 F（模态轨迹进给）。

18.4.10 单轴动态响应的控制

单轴

可以在零件程序、同步动作和 PLC 对单轴进行编程：

- 零件程序：
  - `POS[<轴>]=...`
  - `POSA[<轴>]=...`
  - `SPOS[<轴>]=...`
  - `SPOSA[<轴>]=...`
  - `OS[<轴>]=...`
  - `OSCILL[<轴>]=...`
- 同步动作：
  - `EVERY ... DO`
  - `POS[<轴>]=...`
  - `SPOS[<主轴>]=...`
  - `MOV[<轴>]=...`
- PLC：
  - FC18

动态响应特性

通过下列方式控制轴的动态响应特性：

- MD32060 $MA_POS_AX_VELO（定位轴速度）
  可以修改有效的定位轴速度：
  – 零件程序/同步动作：轴进给率 FA 或者百分比的进给量修正 OVRA
  – PLC：预先设定进给比率 FRate 或者覆盖轴倍率
- MD32300°$MA_MAX_AX_ACCEL（最大轴加速度）
  可以修改有效的最大轴加速度：
  – 零件程序间接修改：写入机床数据，随后“激活机床数据”
  – 零件程序直接修改：百分比的加速度倍率 ACC
  – 同步动作间接修改：写入机床数据，并触发用于激活“激活机床数据”功能的异步子程序
  – 同步动作直接修改：百分比的加速度倍率 ACC （不能由 PLC 设定）。
  通过 PLC 同样可以进行同步动作中的操作。
- 零件程序指令：BRISKA, SOFTA, DRIVEA, JERKA
  不能在同步动作中编程 （只能间接通过 ASUP）。
  不能由 PLC 设定 （只能间接通过异步子程序设定）。
- 有效的伺服参数组
  可以修改有效的伺服参数组：
  – 零件程序/同步动作：SCPARA
  – PLC：DB31, … DBX9.0-2 （调节器参数组）
  有关伺服参数组的详细资料参见“位置控制器的参数组 (页 410)”。

说明

动态响应修改

在零件程序中进行的动态响应修改不会影响指令轴或 PLC 轴的运行。在同步动作中进行的动态响应修改不会影响零件程序中已编程的程序运行。

前馈控制

前馈控制的类型以及哪些轨迹轴应当采用前馈控制，可以在零件程序中直接使用 FFWON/FFWOF 进行编程。在同步动作以及 PLC 控制中只能通过异步子程序间接进行编程。

百分比的加速度倍率 （ACC）

在零件程序或同步动作中可以使用指令 ACC 对机床数据中：
MD32300 $MA_MAX_AX_ACCEL （最大轴加速度）
设定的加速度在 0% – 200% 的范围内进行修改。

句法：
ACC[<轴>]=<值>

含义：

ACC：	百分比加速度倍率的编程关键字
<轴>：	通道轴或主轴的名称
<值>：	加速度变化率，相对于 MD32300 相关
	取值范围： 0 ... 200

当前轴的加速度值可以通过系统变量 $AA_ACC 进行读取。该值取决于：

$AA_ACC[<轴>] = (MD32300 $MA_MAX_AX_ACCEL[<轴>]) * ACC[<轴>] / 100

通过 MD32320 $MA_DYN_LIMIT_RESET_MASK 可以确定通道复位时或者零件程序结束 M30 处 ACC 编程值的初始设置。

说明

用 ACC 编程的加速度倍率可以通过系统变量 $AA_ACC 进行读取。然而在零件程序中读取 $AA_ACC 要在其他时间进行，不同于在同步动作中读取。

系统变量 $AA_ACC 只包含在零件程序中使用 ACC 编程的数值，如果在此期间没有在同步动作中编程 ACC 来修改加速度倍率。反之亦然。

百分比的加速度倍率与主运行轴

根据系统变量 $AA_ACC 是在零件程序中或是在同步动作中被读取，分别输出用于 NC 轴或主运行轴（指令轴、 PLC 轴、 异步摆动轴等）的使用 ACC 编程的加速度倍率值。

为了保证结果正确，必须始终从原先 ACC 编程所在的位置（零件程序或同步动作）读取系统变量 $AA_AC。

示例：

在零件程序中由 ACC 写入：

```
N80 G01 POS[X]=100 FA[X]=1000 ACC[X]=90 IPOENDA[X]
```

在同步动作中由 ACC 写入：

```
N100 EVERY $A_IN[1] DO POS[X]=50 FA[X]=2000 ACC[X]=140 IPOENDA[X]
```

在零件程序中由 ACC 写入并读取 $AA_ACC ：

```
ACC[X]=50                                   ; 写
RO=$AA_ACC[X]                               ;读
```

在零件程序中由 ACC 写入并读取 $AA_ACC ：

```
IF (RO <> $MA_MAX_AX_ACCEL[X] * 0,5)                ; 检查
   SETAL(61000)
ENDIF
```

在同步动作中由 ACC 写入并读取 $AA_ACC ：

```
WHEN TRUE DO ACC[X]=25 R0=$AA_ACC[X]                ; 写入并读取
G4 F1

IF (RO <> $MA_ MAX_AX_ACCEL[X] * 0,25)              ; 检查
   SETAL(61001)
ENDIF
```

单轴的运动结束条件

类似于轨迹插补时的程序段切换标准（G601，G602, G603），也可以在零件程序/同步动作中对单轴运动结束条件进行编程：

程序指令	运动结束条件
FINEA[<轴>]	“精准停”
COARSEA[<轴>]	“粗准停”
IPOENDA[<轴>]	“插补器停止” (IPO-Stopp)

在程序结束或 NC 复位后会保留最后编程的数值。

可通过轴专用的系统变量 $AA_MOTEND 读取当前生效的运动结束条件。

说明

根据系统变量 $AA_MOTEND 是在零件程序中或是在同步动作中被读取，其中分别包含有用于 NC 轴或主运行轴的数值。

示例：

零件程序：

```
N80 G01 POS[X]=100 FA[X]=1000 ACC[X]=90 COARSEA[X]
```

同步动作：

```
N100 EVERY $A_IN[1] DO POS[X]=50 FA[X]=2000 ACC[X]=140
IPOENDA[X]
```

文档：
更多关于程序段切换以及运动结束条件（ FINEA、 COARSEA 和 IPOENDA）的信息请参见：
功能手册之扩展功能分册；定位轴 （P2）， 章节：程序段切换

可编程的伺服参数组 (SCPARA)

在零件程序/同步动作中可以使用 SCPARA 设定伺服参数组。

句法
SCPARA[<轴>]=<参数组编号>

含义

SCPARA：	激活预设伺服参数组的关键字
<轴>：	通道轴的名称
<参数组编号>：	待激活伺服参数组的编号

说明
可以从 PLC 用户程序禁止激活由 SCPARA 设定的参数组：
DB31, … DBX9.3 = 1（禁止通过 SCPARA 设定参数组）
此时 不 显示信息。

可通过系统变量 $AA_SCPAR 读取当前有效参数组的编号。

前提条件

不同的运动结束条件
当运动结束条件不同时零件程序段会以不同速度结束。这可能会在工艺循环和 PLC 用户程序中导致边际效应。

参数组切换
如果要在某个零件程序中、某个同步动作或 PLC 中切换伺服参数组，就必须对 PLC 用户程序进行扩展。

上电

上电之后设定下列初始设置值：

- 用于所有单轴插补的百分比加速度倍率：100%
- 用于所有单轴插补的运动结束条件：FINEA
- 伺服参数组：1

运行方式切换

在运行方式从 AUTO 切换为 JOG 时已编程的动态响应修改会保持有效。

复位

在复位时会保留用于零件程序设定的最后一个编程值。用于主运行插补的设定值不变。

程序段搜索

最后一次编写的轴运动结束条件会被记录，并在一个动作程序段中输出。程序段搜索中处理的最后一个含运行结束条件的程序段用于保存所有轴已编程的运动结束条件。

18.5 前提条件

尺寸单位

进给率的有效尺寸单位取决于所设定的单位制以及所输入的轴类型：

MD10240 $MN_SCALING_SYSTEM_IS_METRIC （控制器的初始单位制：英制或公制）

MD30300 $MA_IS_ROT_AX （回转轴或线性轴）

进给率类型的初始设置

进给率类型的初始设置定义在机床数据中：

MD20150 $MC_GCODE_RESET_VALUES（G 功能组的初始设置）

缺省设置为 G94。

只有当启动一个零件程序时才会显示进给率类型的初始设置。

复位后的有效性

最后所编程的 F、 FA、 OVR、 OVRA 的值在复位之后是否有效，取决于机床数据中的设置：

MD22410 $MC_F_VALUES_ACTIVE_AFTER_RESET（F 功能复位后保持生效）

主轴定位

当 G95、 G96、 G961、 G97、 G971、 G33、 G34、 G35 有效时不应进行主轴定位，因为进行主轴定位之后推导出的轨迹进给率为 0 。 如果此时尚未到达所编程的轴位置，则程序段会无法结束。

18.6 数据表

18.6.1 机床数据

18.6.1.1 NC 专用机床数据

编号	名称：$MN_	说明
10651	IPO_PARAM_THREAD_NAME_TAB	球螺纹插补参数的名称
10704	DRYRUN_MASK	激活试运行进给
10710	PROG_SD_RESET_SAVE_TAB	待更新的设定数据
11410	SUPPRESS_ALARM_MASK	封锁特定报警
11550	STOP_MODE_MASK	确定停止特性
12000	OVR_AX_IS_GRAY_CODE	轴进给倍率开关，格雷编码
12010	OVR_FACTOR_AX_SPEED	轴进给倍率开关的权重系数
12020	OVR_FEED_IS_GRAY_CODE	轨迹进给倍率开关，格雷编码
12030	OVR_FACTOR_FEEDRATE	轨迹进给倍率开关的权重系数
12040	OVR_RAPID_IS_GRAY_CODE	快速移动倍率开关，格雷编码
12050	OVR_FACTOR_RAPID_TRA	快速移动倍率开关的权重系数
12060	OVR_SPIND_IS_GRAY_CODE	主轴倍率开关，格雷编码
12070	OVR_FACTOR_SPIND_SPEED	主轴倍率开关的权重系数

编号	名称：$MN_	说明
12080	OVR_REFERENCE_IS_PROG_FEED	倍率参考速度
12082	OVR_REFERENCE_IS_MIN_FEED	定义轨迹倍率的参考基准
12090	OVR_FUNCTION_MASK	选择倍率设定
12100	OVR_FACTOR_LIMIT_BIN	二进制编码的倍率开关的限制
12200	RUN_OVERRIDE_0	在倍率为 0 时运行
12202	PERMANENT_FEED	用于线性轴的固定进给率
12204	PERMANENT_ROT_AX_FEED	用于回转轴的固定进给率

18.6.1.2 通道专用机床数据

编号	名称：$MC_	说明
20100	DIAMETER_AX_DEF	具有机床 X 轴功能的几何轴
20150	GCODE_RESET_VALUES	G 功能组的初始设置
20172	COMPRESS_VELO_TOL	压缩时允许的最大轨迹进给率偏差
20200	CHFRND_MAXNUM_DUMMY_BLOCKS	相位/半径处的空程序段
20201	CHFRND_MODE_MASK	倒角/倒圆的特性
20750	ALLOW_GO_IN_G96	G96、G961 时的 G0 逻辑
21200	LIFTFAST_DIST	从轮廓快速退回时的运行行程
21220	MULTFEED_ASSIGN_FASTIN	“在一个程序段内编写多个进给率值”功能的 NC I/O 输入字节指定
21230	MULTFEED_STORE_MASK	“在一个程序段内编写多个进给率值”功能的存储器特性
22240	AUXFU_F_SYNC_TYPE	F 功能的输出时间
22410	F_VALUES_ACTIVE_AFTER_RESET	F 功能复位后保持生效

18.6.1.3 进给轴/主轴专用机床数据

编号	名称：$MA_	说明
30300	IS_ROT_AX	回转轴/主轴
32000	MAX_AX_VELO	最大轴速度
32060	POS_AX_VELO	定位轴速度的初始设置

编号	名称：$MA_	说明
32300	MAX_AX_ACCEL	轴加速度
32320	DYN_LIMIT_RESET_MASK	动态响应限制的复位特性
34990	ENC_ACTIVAL_SMOOTH_TIME	实际值的平滑时间常数
35100	SPIND_VELO_LIMIT	最大主轴转速
35130	GEAR_STEP_MAX_VELO_LIMIT	齿轮档的最大转速
35140	GEAR_STEP_MIN_VELO_LIMIT	齿轮档的最小转速
35160	SPIND_EXTERN_VELO_LIMIT	PLC 主轴转速限制

18.6.2 设定数据

18.6.2.1 通道专用设定数据

编号	名称：$SC_	说明
42000	THREAD_START_ANGLE	螺纹加工起始角
42010	THREAD_RAMP_DISP	攻丝中轴的加速性能
42100	DRY_RUN_FEED	空运行进给率
42101	DRY_RUN_FEED_MODE	空运行进给的模式
42110	DEFAULT_FEED	进给率缺省值
42600	JOG_FEED_PER_REV_SOURCE	JOG 方式下的旋转进给率
43300	ASSIGN_FEED_PER_RES_SOURCE	定位轴/主轴的旋转进给率

18.6.2.2 进给轴/主轴专用设定数据

编号	名称：$SA_	说明
43210	SPIND_MIN_VELO_G25	已编程的主轴转速限制 G25
43220	SPIND_MAX_VELO_G26	已编程的主轴转速限制 G26
43230	SPIND_MAX_VELO_LIMS	G96 时的主轴转速限制

18.6.3 信号

18.6.3.1 发送至通道的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
激活空运行进给率	DB21, ...DBX0.6	DB320x.DBX0.6
进给倍率	DB21, ...DBB4	DB320x.DBB4
快速移动倍率	DB21, ...DBB5	DB320x.DBB5
进给禁止	DB21, ...DBX6.0	DB320x.DBX6.0
快速移动倍率有效	DB21, ...DBX6.6	DB320x.DBX6.6
进给倍率生效	DB21, ...DBX6.7	DB320x.DBX6.7
进给停止，几何轴 1	DB21, ...DBX12.3	DB320x.DBX1000.3
进给停止，几何轴 2	DB21, ...DBX16.3	DB320x.DBX1004.3
进给停止，几何轴 3	DB21, ...DBX20.3	DB320x.DBX1008.3

18.6.3.2 来自通道的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
空运行进给率已选择	DB21, ...DBX24.6	DB170x.DBX0.6
快速移动进给倍率已选择	DB21, ...DBX25.3	DB170x.DBX1.3
激活用于轨迹/几何轴的固定进给率 1	DB21, ...DBX29.0	DB320x.DBX13.0
激活用于轨迹/几何轴的固定进给率 2	DB21, ...DBX29.1	DB320x.DBX13.1
激活用于轨迹/几何轴的固定进给率 3	DB21, ...DBX29.2	DB320x.DBX13.2
激活用于轨迹/几何轴的固定进给率 4	DB21, ...DBX29.3	DB320x.DBX13.3
试运行进给率有效	DB21, ...DBX318.6	DB330x.DBX4002.6

18.6.3.3 发送至进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
进给/主轴倍率	DB31, ...DBB0	DB380x.DBB0
补偿生效	DB31, ...DBX1.7	DB380x.DBX1.7

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
激活用于机床轴的固定进给率 1	DB31, ...DBX3.2	DB380x.DBX3.2
激活用于机床轴的固定进给率 2	DB31, ...DBX3.3	DB380x.DBX3.3
激活用于机床轴的固定进给率 3	DB31, ...DBX3.4	DB380x.DBX3.4
激活用于机床轴的固定进给率 4	DB31, ...DBX3.5	DB380x.DBX3.5
进给停止/主轴停止	DB31, ...DBX4.3	DB380x.DBX4.3

18.6.3.4 来自进给轴/主轴的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
旋转进给率生效	DB31, ...DBX62.2	DB390x.DBX2.2
用于定位轴的 F 功能	DB31, ...DBB81	-
编程的转速过高	DB31, ...DBX83.1	DB390x.DBX2001.1

W1：刀具补偿

19

19.1 简要说明

刀具补偿数据的计算

SINUMERIK 840D sl 控制器能够计算下列刀具补偿数据：

- 长度补偿
- 半径补偿
- 刀具数据存放在灵活的刀具补偿存储器中：
  - 通过 T 号(0 到 32000)标识刀具
  - 最多可以通过 9 个刀沿来定义一把刀具
  - 刀沿可最多用 25 个刀具参数来描述
- 刀具选择选项： 立即选择或者通过可选的 M 功能选择
- 刀具半径补偿：
  - 选择和取消方案选项： 正常或者基于轮廓
  - 作用于所有插补方式的补偿：
    直线
    圆弧
    螺线
    样条
    多项式
    渐开线
  - 外角补偿选项：
    过渡圆弧 / 椭圆 （G450） 或者等距线交点 （G451）
  - 功能 G450/G451 可以根据轮廓由参数来调整
  - 空运行至外角，在 G450 时使用参数 DISC
  - 补偿平面内无轴运行时的中间程序段数量可选
  - 碰撞监控选项：
    会预先识别出可能发生的轮廓破坏，当：
    - 轨迹路径小于刀具半径时
    - 内角的宽度小于刀具直径时
  - 刀具半径补偿保持恒定
  - 用于多项式的交点程序

可定向刀架

如果刀架运动能够稳定地（无 NC 轴）完成刀具定向，则该功能能够在考虑刀具长度补偿的情况下进行斜面加工。 在这种情况下无需进行较为费时的 5 轴转换。

文档：
功能手册之特殊功能分册；多轴转换 (F2)

请选择合适的刀具数据和刀架数据，以便控制系统考虑刀具长度补偿。 其中一些说明数据可以直接被控制系统接收到当前框架中。

说明

更多关于刀具和刀具补偿的说明与完整的刀具补偿（WLK 和 WRK）编程方案及其特例请查阅：

文档：
编程手册之基本原理分册

平面/单一 D 号码结构

通过单一 D 号码可以选择补偿，含管理功能。

刀具补偿 - 特殊操作

对刀具长度和磨损进行符号计算时使用设定数据：

SD42900 $SC_MIRROR_TOOL_LENGTH （镜像时的刀具长度符号变换）

SD42960 $SC_TOOL_TEMP_COMP （与刀具相关的温度补偿）

这同样适用于几何轴镜像时的磨损分量，或者通过设定数据更换加工平面时的磨损分量特性。

文档：
编程手册之基本原理分册： 刀具补偿

G461/G462

为了在激活与取消刀具半径补偿的特定情况下也能够清理内角，系统引入了指令 G461 和 G462 ，增加了刀具半径补偿时的逼近/回退方案。

- G461
  如果最后的刀具半径补偿程序段与前一程序段不可能产生交点，则控制系统会计算出一个交点，方法是用一个圆弧对该程序段的偏移曲线进行延长，使其圆心位于未补偿程序段的终点，半径与刀具半径相同。
- G462
  如果最后的刀具半径补偿程序段与前一程序段不可能产生交点，则控制系统会计算出一个交点，方法是在带刀具半径补偿的最后一个程序段的终点处插入一条直线（该程序段通过其终点切线延长）。

从 G40 切换到 G41/42

在带有相关刀沿位置的刀具（车刀与磨具）上，从 G40 到 G41/G42 的切换也不再作为“换刀”进行处理，反之亦然。

刀具补偿环境

软件版本 7.1 提供了和刀具数据当前状态相关的各种功能：

- 保存
- 删除
- 读取
- 修改

其中一些功能已经在测量循环中实现。 现在已变成通用功能。

使用另一项功能可以确定关于当前刀具横坐标、纵坐标和垂直坐标的信息。

19.2 刀具

19.2.1 简介

选择刀具

使用 T 功能在程序中选择一个刀具。

是否立即使用 T 功能换刀，取决于机床数据中的设置：

MD22550 $MC_TOOL_CHANGE_MODE （M 功能时新的刀具补偿）

立即换刀

MD22550 $MC_TOOL_CHANGE_MODE = 0 （ M 功能时新的刀具补偿）

立即用 T 功能进行换刀。

在有刀具转塔的车床中主要使用这种方式。

使用 M06 换刀

MD22550 $MC_TOOL_CHANGE_MODE = 1 （ M 功能时新的刀具补偿）

T 功能仅仅用作换刀准备。

在有刀库的铣床中，为了在加工的同时做好换刀准备，主要使用这种换刀方式。

通过在机床数据中输入的 M 功能从主轴卸下旧刀具并换入新刀具：

- MD22560 $MC_TOOL_CHANGE_M_CODE （用于换刀的 M 功能）

根据 DIN 66025 应当使用 M 功能 M06 进行换刀编程。

换入的新刀具通过机床数据预选：

- MD20121 $MC_TOOL_PRESEL_RESET_VALUE（复位时预选的刀具）

应当在 复位 和启动时考虑其刀具长度补偿值，按照机床数据：

- MD20110 $MC_RESET_MODE_MASK （确定复位/零件程序结束后控制系统的初始设置）

T 功能或刀具编号的取值范围

T 功能或刀具编号可以采用下列整数值：

- 最小值：T0 （无刀具）
- 最大值：T32000 （编号为 32000 的刀具）

刀沿 D

可为一把刀具指定多个刀沿 D1 ... Dn。

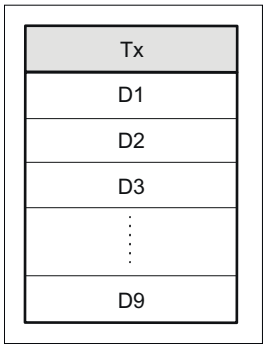


图 19-1　　示例：有 9 个刀沿 （D1 到 D9）的刀具 Tx

每把刀具的最大刀沿数

每把刀具的刀沿（D 编号）数上限通过以下机床数据定义：

- MD18106 $MN_MM_MAX_CUTTING_EDGE_PERTOOL

D 功能

通过编写对应的 D 功能来选择需要用于加工的刀沿 1 ... n，例如在零件程序中通过 D1 ... Dn。所选择的刀沿始终基于生效的刀具。 无生效刀具时（当前 T 编号为 T0 ⇒ 未选择刀具），刀沿无效。

编写 D0 会取消生效刀具 Tx 的所有刀具补偿。

换刀时选择刀沿

通过新的 T 编号选择新刀具并换入后，可通过以下方式选择刀沿：

- 编写刀沿编号 Dx。
- 通过机床数据设定刀沿号：
  MD20270 $MC_CUTTING_EDGE_DEFAULT = <值>（未编程时刀沿的缺省设置）

值	含义
= 0	不会根据 M06 自动选择刀沿。
≠ 0	根据 M06 选择的刀沿号。
= -1	保留旧刀具的刀沿号，并根据 M06 选择新刀具的刀沿号。

激活刀具补偿

编写 D1 到 Dn 会激活对当前生效刀具的刀沿补偿。 然而刀具长度补偿和刀具半径补偿在不同的时间点生效：

- **刀具长度补偿**从激活了该补偿的轴的第一次进给运动开始进行。
  运动必须为线性插补 （G0,G1， POS,POSA） 或多项式插补（POLY）。 如果 POS/POSA 轴为有效的几何轴，则刀具长度补偿从激活了该补偿的轴的第一次进给运动开始进行。
- **刀具半径补偿**通过编程 G41/G42 在有效平面（G17、 G18 或 G19） 中生效。
  在通过 G41/G42 激活刀具半径补偿的 NC 程序段中，必须激活 G0（快速运行）或 G1（线性插补），且至少须编写一根位于生效工作平面中的几何轴。
  如果在激活时仅给定了一根轴，那么系统会自动补充第二根轴的最后位置，之后在两根轴上运行。
  若该轴当前不是几何轴，则须通过编写 GEOAX 将其转换为几何轴。
  用 G41/G42 选择刀具半径补偿 时，只允许在一个程序段中使用 G0 （快速运行）或者G1（性线插补）。

19.2.2 补偿存储器结构

刀具补偿存储器容量

每个通道都可以拥有自己的刀具补偿存储器 （TO 单元）。

哪个刀具补偿存储器用于哪个通道，由机床数据进行设置：

MD28085 $MC_MM_LINK_TOA_UNIT（TO 单元至通道的分配）

所有由 NC 管理的刀具，其刀沿的最大数量通过机床数据进行设置：

MD18100 $MN_MM_NUM_CUTTING_EDGES_IN_TOA（NC 中刀沿的数量）

刀具

刀具补偿存储器由编号 T1 到 T32000 的刀具构成。

每把刀具可以通过 TOA（刀具补偿有效）文件或者分别使用软键“新刀具”来创建。不需要的补偿可以赋值为零。这也是创建补偿存储器时的缺省设置。补偿存储器中的各值（刀具参数）可通过系统变量由程序读写。

说明

刀具 （T1 到 T32000）不需要按升序、连续以及用 T1 开头的方式输入到刀具补偿存储器中。

刀沿

每把刀具可以最多有 9 个刀沿 （D1 到 D9）。如果将一把新的刀具放入刀具补偿存储器中，会自动创建第一个刀沿（D1）。使用软键“新刀沿”可以按升序连续设定更多的刀沿（最多 8 个）。用这种方式可以为每把刀具分配不同数量的刀沿。

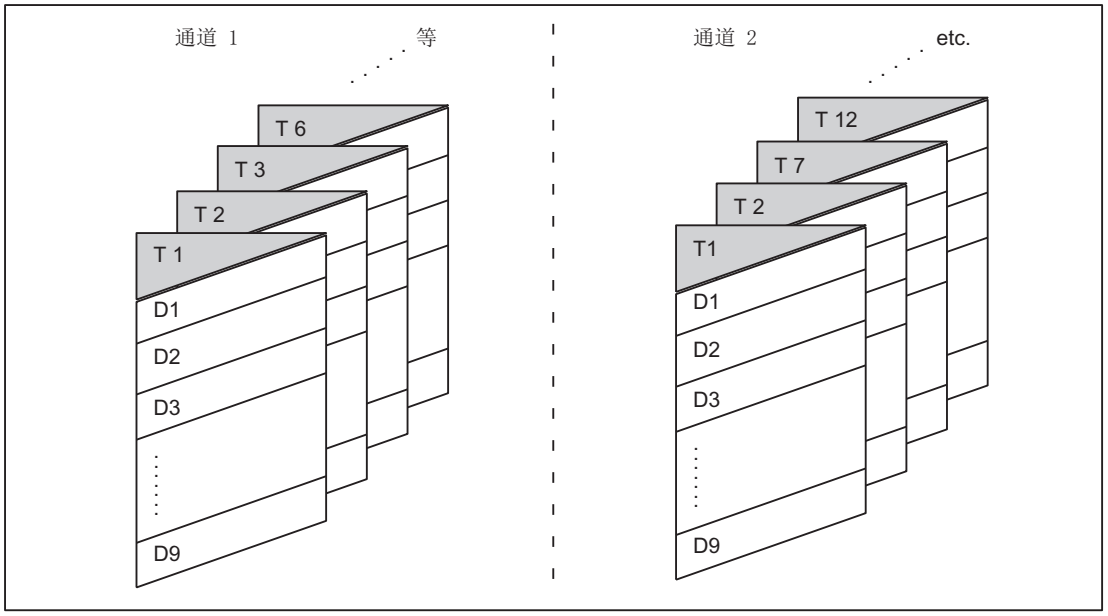
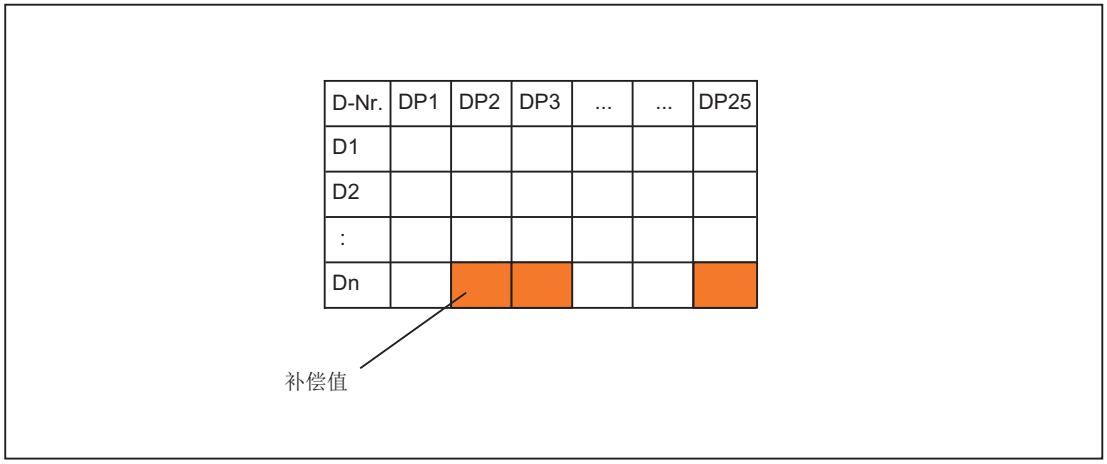


图 19-2　　用于 2 个通道的刀具补偿存储器结构示例

19.2.3 刀具补偿计算

D 号

D 号 足够用于进行刀具补偿计算 （可通过 MD 设定）。



编程

上面的补偿程序段应当在 NC 中进行计算。

在零件程序中调用：

```
...
Dn
```

19.2.4 用于 NC 地址 T 和 M 的地址扩展

MD20096

是否应当也在**未**激活刀具管理时将 T 和 M 的地址扩展视为“主轴号”， 可以通过机床数据来选择：

MD20096 $MC_T_M_ADDRESS_EXT_IS_SPINO（主轴编号作为地址扩展）

D 号和 T 号的参照规定和功能“刀具管理”激活时一样。

对 D 号的影响

通过 D 号来确定补偿数据组。

不能使用地址扩展对 D 地址进行编程。

D 地址的运用始终取决于当前有效的刀具。

编程的 D 地址始终针对主主轴上的有效刀具 （类似于刀具管理功能），当机床数据设置为：

MD20096 $MC_T_M_ADDRESS_EXT_IS_SPINO = TRUE（主轴编号作为地址扩展）

对 T 号的影响

在刀具管理生效时，主主轴（或主刀具夹具）相关的编程值显示为已编程/有效的 T 号。

刀具管理无效时， **所有**已编程的 T 值显示为已编程/有效（与使用哪些地址扩展进行编程无关）。

只有与主主轴相关的编程 T 值显示为已编程/有效，当：

MD20096 $MC_T_M_ADDRESS_EXT_IS_SPINO = TRUE（主轴编号作为地址扩展）

示例

在下面的示例中会展示出 MD20096 的影响。

观察两根主轴。主轴 1 为主主轴。M6 被确定为换刀信号。

T1 = 5	
M1= 6	
T2 = 50	
M2 = 6	
D4	

- 当刀具管理有效时 D4 针对刀具“5”。
  T2=50 确定用于副主轴的刀具，其刀具不会影响轨迹校正。轨迹仅仅通过用于主主轴的编程刀具来确定。
- D4 针对刀具“50”，刀具管理无效并且使用机床数据：
  MD20096 $MC_T_M_ADDRESS_EXT_IS_SPINO = **FALSE** （ T，M 换刀时地址扩展的含义）
  在 NC 中既不分析 T 的地址扩展也不分析 M 的地址扩展。
  每个换刀指令确定一个新的轨迹补偿。
- D4 针对刀具“5”（如同在刀具管理有效时），刀具管理无效并且使用机床数据：
  MD20096 $MC_T_M_ADDRESS_EXT_IS_SPINO = **TRUE**.
  地址扩展 1 （T1=..., M1=...） 用于说明主主轴。

说明

到目前为止，当刀具管理无效时，每次编程换刀（使用 T 或 M）都会导致在轨迹中重新计算刀具补偿。这时不进一步定义地址扩展。扩展的含义由用户来确定（通过 PLC 用户程序）。

19.2.5 任意 D 号设定

“相对” D 号

在 NC 中可将用于刀具补偿数据组的 D 号作为 “相对” D 号使用。此时会将每个 T 号分配给相应的 D 号。

功能

设定 D 号时的功能扩展：

- 最大允许使用的 D 号通过机床数据来确定：
  MD18105 $MN_MM_MAX_CUTTING_EDGE_NO （ D 号的最大数值 （DRAM））
  缺省值为 9，这能保证现有的应用可以继续运行。
- **每把刀具**的刀沿的数量（或补偿数据组）可以通过机床数据来确定：
  MD18106 $MN_MM_MAX_CUTTING_EDGE_PERTOOL （ 每把刀具 D 号的最大数量 （DRAM））
  借此可以根据实际可能的刀沿数量调整每把刀具需要编程刀沿的数量，对其进行监控。
- 此外还能在 NC 中对 D 号重新命名并为刀沿设定任意的 D 号。

说明

除了设定相对 D 号之外，D 号也可设为“平面”或者“绝对” D 号（1-32000），而不参照任意一个 T 号（在“平面 D 号结构”功能内）。

刀沿号 CE

在对 D 号重命名时，会遗失通过刀具目录确定的刀沿号的相关信息。因此在重命名后无法再看出它到底是目录中的哪个刀沿。

如果在设备改造时需要该信息，则要为每个刀沿引入一个 **CE 号**，在对 D 号重命名时它会被保留。

在零件程序中 D 号也称为刀沿补偿。该**补偿 D 号**和 **CE** 号 （刀具目录中的编号） 分开管理并且为一个任意数字，用于零件程序中的标识和屏幕上的显示。

在设备改造时 CE 号指出了具体的物理刀沿。在换刀选择补偿时 NC 不分析刀沿编号 CE（仅通过 OPI 可用）。

CE 号通过系统变量 **$TC_DPCE[t,d]** 来定义：

- **t** 代表内部 T 号
- **d** 代表 D 号。

定义时系统会进行冲突检查；也就是说一把刀具的刀沿编号必须各不相同。变量 $TC_DPCE 是刀沿参数数据组 $TC_DP1,..., $TC_DP25 的组成部分。

只有当最大的刀沿号（MD18105）大于每把刀具最大的刀沿数量（MD18106）时，才建议设置 $TC_DPCE。

此时刀沿号等于刀沿的序列号。创建的刀具补偿从 1 开始，递增到每把刀具的最大刀沿数量（MD18106）。

CE 号等于 D 号（与先前特性兼容），如果：

MD18105 ≤ MD18106。

读操作返回 CE=D。写操作被忽略，且不引发报警。

说明

对于有效刀具补偿的补偿值 $TC_DP1，……， $TC_DP25 可以使用系统变量 $P_AD[n] 读取，其中 n=1……25 。n=26 时，会获得有效补偿的 CE 号。

指令

若最大刀沿编号**大于**每把刀具的刀沿数上限，可使用下表中列出的指令

- (MD18105 $MN_MM_MAX_CUTTING_EDGE_NO) > (MD18106 $MN_MM_MAX_CUTTING_EDGE_PERTOOL)

指令	含义
CHKDNO	检查现有 D 号的单一性； 所有在一个 TO 单元内定义的刀具的 D 编号只能出现一次。备用刀具在此不考虑。
GETDNO	确定刀具某刀沿的 D 号。 如果输入的参数不存在 D 号，则设定 d=0 。 如果该 D 号无效，将返回一个大于 32000 的值。
SETDNO	设置或修改刀具 T 的刀沿 CE 的 D 号。 如果输入的参数不存在数据组，则会返回 FALSE 。句法错误会导致报警。不可以明确地将 D 号设为 0。
GETACTTD	确定绝对 D 号的相应 T 号。 不检查单一性。如果在一个 TO 单元内有多个相同的 D 号，将会返回第一个所找到刀具的 T 号。 在使用“平面” D 号时使用该指令就没有意义，因为此时始终返回数值 1（没有 T 号在数据管理器中）。
DZERO	将 TO 单元的所有 D 号标识为无效； 该指令用于设备改造。 如此标识的补偿数据组不再由语言指令 CHKDNO 来检查。为了重新对其进行访问，必须重新使用 SETDNO 设定 D 号。

说明

若最大刀沿编号**小于**每把刀具的刀沿数上限，那么表中列出的指令无效。出于兼容性原因，此设置为系统中的预设。

单个指令的详细说明参见：

文档：
编程手册之基本原理分册

激活

为了能够使用单一 D 号以及上述语言指令，刀具的 D 号必须能够自由命名。

为此必须满足下列条件：

- MD18105 > MD18106
- “平面 D 号”功能不能被激活。
  MD18102 $MN_MM_TYPE_OF_CUTTING_EDGE （编程 D 号的类型（SRAM））

示例

MD18105 $MN_MM_MAX_CUTTING_EDGE_NO = 1 （ D 号的最大值）

此时每把刀具最多可以定义一个补偿 （ D 号 = 1）。

说明

当“平面 D 号”功能被激活时只能在 TO 单元来定义一个 D 补偿。

MD18105 $MN_MM_MAX_CUTTING_EDGE_NO = 9999

此时可以为刀具配置单一的 D 号。

示例：为 T 编号（刀具）指定 D 编号（刀沿），并说明检查唯一性的方法：

- 指定对应关系：
  - T 编号 1 ⇒ 指定 D 编号 1、 2、 3
  - T 编号 2 ⇒ 指定 D 编号 10、 20、 30、40、50
  - T 编号 3 ⇒ 指定 D 编号 100、 200、 30（**错误**：设定成 30，原应为 300）
- 检查唯一性：
  一个 TO 单元内定义的所有刀具的 D 编号须为唯一。
  检查通过指令 CHKDNO 进行。若未设定参数，那么系统会检查所有刀具的 D 编号并进行比较。本示例中返回值 == FALSE，因为 T2 和 T3 中均有 D30。

指令	检查	检查结果
CHKDNO	检查所有刀具的所有 D 编号，并相互比较	返回值 == FALSE，因为重复编写了 D=30。
CHKDNO (2, 3, 30)	刀具 2 和 3 中的 D 编号 30	返回值 == FALSE，因为重复编写了 D=30。
CHKDNO (1, 3)	刀具 1 和 3 的所有 D 编号	返回值 == TRUE, 尽管第三把刀具的 D=30 与第二把 D=30 冲突。

说明

每把刀具的刀沿

若启用有 **n** 个刀沿的刀具，请将机床数据的值设置为 **n**。这样可避免为刀具定义不允许的刀沿。

MD18106 $MN_MM_MAX_CUTTING_EDGE_PERTOOL = **n**

编程示例

重新命名一个 D 号

现在要将一个 CE = 3 的刀沿的 D 号从 2 重命名为 17。为此需要进行下列设定：

- $TC_DPx[<刀具 Tn>, <刀沿 Dm>]
- 内部 T 编号 Tn = 1

- D 编号 Dm = 2
- 刀具使用一刀沿并使用：

程序代码	注释
$TC_DP2[1, 2] = 120	刀具长度 = 120
$TC_DP3[1, 2] = 5.5	刀具半径 = 5.5
$TC_DPCE[1, 2] = 3	刀沿编号 CE = 3

- MD18105 $MN_MM_MAX_CUTTING_EDGE_NO = 20 （ D 号的最大值）

在零件程序中该补偿默认使用 T1,....D2 进行编程。

刀沿 3 的当前 D 号会被设定给一个变量（DNrAlt）并且为新的 D 号定义变量 DNrNeu：

程序代码
def int DNrAlt, DNrNeu = 17
DNrAlt = GETDNO(1, 3)
SETDNO(1, 3, DNrNeu)

在此为刀沿 CE=3 设定新的 D 值 17。

现在可以通过 D 号 17 来访问该刀沿的数据； 既可以通过系统变量也可以在编程中使用 NC 地址 D。

此时在零件程序中该补偿默认使用 T1,....D17 进行编程并且数据访问方式如下：

程序代码	注释
$TC_DP2[1, 17] = 120	
$TC_DP3[1, 17] = 5.5	
$TC_DPCE[1, 17] = 3	; 刀沿编号 CE

说明

在刀具还定义另一刀沿的情况下，例如 $TC_DPCE[1, 2] = 1；= CE， 则刀沿 1 的 D 号 2 的命名不能与刀沿 3 的 D 号相同，亦即：SETDNO(1, 1, 17) 会提供状态 = FALSE 作为返回值。

DZERO - 将 D 号设置为无效

激活该指令会使 TO 单元中刀具所有的 D 号无效。补偿无法被激活，直到 NC 中再次有有效 D 号。D 号通过指令 SETDNO 重新设定。

会设定下列刀具 （全部使用刀沿编号 1）：

T1, D1	刀沿 CE=1 的 D 号
T2, D10	刀沿 CE=1 的 D 号
T3, D100	刀沿 CE=1 的 D 号

此时进行编程：

程序代码
DZERO

如果这时激活了一种补偿（例如使用 T3 D100），则会产生报警， 因为 D100 当前未被定义。

重新定义 D 号，通过：

程序代码	注释
SETDNO(1, 1, 100)	; T=1，刀沿 1 获得（新的）D 编号 100
SETDNO(2, 1, 10)	; T=2，刀沿 1 获得（旧的） D 编号 10
SETDNO(3, 1, 1)	; T=3，刀沿 1 获得（新的）D 编号 1

说明

指令 DZERO 能够在掉电时将 NC 中的 D 号保留为未定义状态。要将其清除，请在重新通电后重复指令 DZERO。

改造程序的基本流程

需要确定，所需的刀具和刀沿已经存在。刀库和 NC 的刀具存储器处于任意状态。新加工零件程序中的 D 号通常与具体刀沿的 D 号不一致。改造程序可以为如下结构：

程序代码	注释
DZERO	; 将 TO 单元的所有 D 号标识为无效
....	; 一个或多个刀库刀位上的一次或多次循环，检查刀具及其刀沿号。
	; 如果找到的刀具未被禁用（$TC_TP8）并且具有所需的刀沿号 CE （GETDNO），则使用新的 D 号设置刀沿（SETDNO）。
....	; 执行卸刀与装刀进程
	; 可以使用具有‘待卸载’或‘待装载’状态的刀具进行加工。
CHKDNO	; 卸刀/装刀以及 D 号的重命名完成。
	; 可以对各刀具和/或 D 号进行检查并且根据返回值自动处理所出现的冲突。

19.2.6 换刀中出现故障时的补偿程序段

MD22550

如果在零件程序中编程了刀具准备指令而 NC 发现了一个故障，例如用于 T 号编程的数据组在 NC 中不存在，则操作员可以对故障情况进行判断并进行相应的操作，以便继续完成加工。

可以独立进行换刀编程，相关的机床数据：

MD22550 $MC_TOOL_CHANGE_MODE （M 功能时新的刀具补偿）

MD22550 $MC_TOOL_CHANGE_MODE = 0

```
T= 'T 号'          ; 在一个 NC 程序段中进行刀具准备 + 换刀，
                   ; 即编程 T 会引起 NC 中
                   ; 新的 D 补偿被激活。参见机床数据：
                   ; MD20270 $MC_CUTTING_EDGE_DEFAULT （未编程的刀沿初始设置）
```

MD22550 $MC_TOOL_CHANGE_MODE = 1

```
T= 'T 号'          ; 刀具准备
M06                ; 换刀
                   ; （换刀 M 代码的号可设置）
                   ; 即编程 M06 会引起 NC 中
                   ; 新的 D 补偿被激活；（对此参见
                   ; 机床数据 MD20270 $MC_CUTTING_EDGE_DEFAULT）
```

当刀具管理未激活时可以发现下列问题：

- 缺少 D 补偿数据组
- 零件程序错误

说明

“刀具不在刀库中”的问题不能被发现，因为在刀具补偿时 NC 不存在刀库信息。

缺少 D 补偿数据组

程序加工停留在 D 值出错的程序段处（与机床数据 MD22550 的值无关）。操作员要么对程序进行修正，要么重新装载出错的程序段。

为此在使用平面 D 号功能需要 D 号，而在其他情况下还额外需要 T 号。这些参数通过触发的报警一起进行传输。

零件程序错误

故障情况下的加工可能性取决于换刀是如何编程的，通过机床数据来确定：

MD22550 $MC_TOOL_CHANGE_MODE （M 功能时新的刀具补偿）

使用 T 编程换刀 (MD22550 = 0)

对于这种情况会通过 NC 中所提供的“补偿程序段”功能完成加工。NC 程序停止在 T 值编程错误的 NC 程序段处。继续运行程序需要重新执行“补偿程序段”。

操作员可以进行下列操作：

- 可以修正出错的零件程序。
- 缺少的刀沿补偿数据可以由 HMI 补充提供。
- 缺少的刀沿补偿数据可以在“覆盖”状态下在 NC 中导入。

在操作员干预之后按下 启动 键并重新执行被拒绝的程序段。如果错误被正确修正，则程序继续运行。否则会重新产生一个报警。

使用 T 和 M06 编程换刀 (MD22550 = 1)

这种情况下在刀具准备（ T 编程）程序段中发现一个错误，但一开始会将其忽略。继续进行加工，直到 NC 程序中提出换刀请求（通常是 M06）。程序应该停在该位置。

编程的 T 地址可以位于 M06 指令之前的任意程序行，或者这两个语句可以处于不同的（子）程序中。因此通常情况下不能通过补偿程序段来修改一个已执行完毕的程序段。

此时的加工可能性与值为 0 相同。

能够重新装载缺少的数据。然后必须在“覆盖”状态下对 T 进行编程。

只要存在程序错误，就不能对出错行进行修正（Txx），而只能修正程序加工停止并产生报警的那一行。只有当机床数据：

MD22562 $MC_TOOL_CHANGE_ERROR_MODE Bit0 = 1 （换刀中出现故障时的特性）

过程如下：

```
Txx                ; 错误！带有 xx 的数据组不存在
                   ; 记录状态；记录 xx ；
                   ; 在程序中继续进行
....
```

```
M06                ; 识别“缺少 xx”标志位 → 输出报警，
                   ; 程序停止
                   ; 使用例如 Tyy M06 修正程序段，启动，
                   ; 编译程序段 Tyy M06，一切正常。
                   ; 继续进行加工。
```

重新执行该程序位置时过程如下：

```
Txx                ; 错误！带有 xx 的数据组不存在，
                   ; 记录状态：记录 xx ；
                   ; 在程序中继续进行
....
Tyy M06            ; 识别“缺少 xx”标志位 → 拒绝，不作其他响应，
                   ; 因为 Tyy M06 是正确的 → 程序不停止（正确）。
```

如有需要可以在程序结束之后对 T 指令的原始位置进行修正。如果换刀逻辑在机床上不能进行处理，则必须中断程序并对出错的位置进行修正。

如果只是缺少数据组，则将其传输给 NC，在“覆盖”状态下编程 Txx 并接着继续运行程序。

和“缺少 D 号”一样在“缺少 T 号”时也可以通过相应的报警（17191）将所需的参数（T 号） 提供给操作员。

说明

为了能够实现程序修正，程序会立即停止在出错的 Txx 程序段上。

程序测试运行时也会停止，使用机床数据：

MD22562 $MC_TOOL_CHANGE_ERROR_MODE Bit0=1 （换刀中出现故障时的特性）

19.2.7 定义刀具参数的作用

MD20360

刀具参数对于车床 X 轴的作用可以和直径编程一起通过机床数据设置：

MD20360 $MC_TOOL_PARAMETER_DEF_MASK（定义刀具参数）

详细信息在上述机床数据中进行说明。

通过手轮以一半行程进行 DRF 手动运行

在 DRF 手轮运行时，车床 X 轴只能使用设定增量的一半行程按如下方式运行：

通过机床数据将手轮工作方式设置为“行程设定”：

MD11346 $MN_HANDWHEEL_TRUE_DISTANCE = 1（手轮工作方式：行程设定/速度设定）

使用机床数据定义车床 X 轴上的 DRF 偏移（直径补偿值）：

MD20360 $MC_TOOL_PARAMETER_DEF_MASK 位 9 = 1（定义刀具参数）

在针对某轴撤销 DRF 偏移时（DRFOF），会一同删除现有的刀具补偿（刀具方向上的手轮叠加）。

说明

更多关于手轮叠加运动的信息请查阅：

文档：

功能手册之扩展功能分册；手动运行和通过手轮进行手动运行（H1）

编程手册之基本原理分册

（编程指南描述了如何针对某轴撤销 DRF 偏移的完整编程方法。）

19.3 平面 D 号结构

19.3.1 简介

对于**车床**，可以通过平面 D 号结构（不仅是通过 D 号）实现简单的刀具管理。

这样便得到下列边界条件：

- 仅在**无**激活的通用刀具管理时可用
- 无姊妹刀具可定义
- 无刀库可定义
- 无磨具可定义

激活

可通过以下参数来确定哪种 D 号管理方式有效：

MD18102 $MN_MM_TYPE_OF_CUTTING_EDGE = <值>

值	含义
0	无平面 D 号管理有效
1	使用绝对的**直接 D 编程**的平面 D 号结构有效

19.3.2 新建一个 D 号（补偿程序段）

编程

使用系统变量 $TC_DP1... $TC_DP25 能够对刀具补偿进行编程。其含义与原先相同。

写入方式变为：不再给定 T 号。

- “平面 D 号”功能 **有效**：
  $TC_DPx[d] = 值 ;其中 x=参数号, d=D 号
  也就是说，在“平面 D 号”功能激活的情况下，使用该句法的数据只能装载到 NC 中。
- ‘平面 D 号’功能 **无效**：
  $TC_DPx[t][d] = 值 ; 其中， t=T 号, d=D 号

对于每个刀具，一个 D 号仅能分配一次；即每个 D 号仅代表一个补偿数据组。

第一次编程一个不存在的 D 号时， NC 会在存储器中创建新的数据组。

D 数据组或者补偿数据组的最大数量 （最多 600）通过机床数据来设置：

MD18100 $MN_MM_NUM_CUTTING_EDGES_IN_TOA （TO 范围内的刀具补偿）

数据备份

数据备份以同一种格式进行；即一个使用“平面 D 号”功能生成的备份文件不能被录入到未激活该功能的控制系统的 NC 中，

反之亦然。

D 号范围

1 - 99 999 999

19.3.3 D 号编程

MD18102 = 1

如果 MD18102 $MN_MM_TYPE_OF_CUTTING_EDGE = 1， 则 D 补偿激活，不参照特定的刀具。

D0 含义和以前一样，即“撤销 NC 中有效的补偿”。

D 的地址扩展

不能对 D 进行地址扩展。对于刀具轨迹，在一个时间点只可能有一个有效的补偿数据组。

编程

在零件程序中同原先一样编程。只是编程 D 号的取值范围扩大了。

示例 1：

MD 编程		含义
MD22550 $MC_TOOL_CHANGE_MODE	= 0	通过编程 T 进行换刀。
MD18102 $MN_MM_TYPE_OF_CUTTING_EDGE	= 1	不参照特定刀具的 D 号编程。
MD20270 $MC_CUTTING_EDGE_DEFAULT	= -1	在换刀时 D 补偿保持不变。

程序代码	注释
...	
D92	
X0	; 运行中使用 D92 的补偿。
T17	; 输出 T=17 至 PLC 。
X1	; 运行中使用 D92 的补偿。
D16	
X2	; 运行中使用 D16 的补偿。
D32000	
X3	; 运行中使用 D32000 的补偿。
T29000	; 输出 T=29000 至 PLC 。
X4	; 运行中使用 D32000 的补偿。
D1	
X5	; 运行中使用 D1 的补偿。
...	

示例 2：

MD22550 = 0

程序代码	注释
T1	
T2	
T3	
D777	; 不等待，激活 D777，T3= 显示的编写并有效的刀具， D777= 编写并有效的补偿。

说明

NC 程序负责换刀并将 D 补偿分配给具体的刀具，必要时由 PLC 程序负责。

通过零件程序删除 D 号

- **使用** 平面 D 号：
  $TC_DP1[d] = 0
  TO 单元内的补偿数据组以及 D 号被删除。
  存储器被清空，可以用来定义其他的 D 号。
- **不使用** 平面 D 号：
  $TC_DP1[t][d] = 0
  删除刀具 t 的刀沿 d。
- $TC_DP1[0]=0
  删除 T0 单元的所有 D 补偿。

可以删除有效的补偿数据组（D 号）。也就是说在删除之前可能会需要进行 D0 编程。

刀具机床数据

下列机床数据对 NC 中刀具和刀沿（D 号）的处理产生影响：

机床数据	含义
MD20270 $MC_CUTTING_EDGE_DEFAULT	换刀之后的标准刀沿
MD20130 $MC_CUTTING_EDGE_RESET_VALUE	刀沿 - 启动（复位/零件程序结束）时的长度补偿
MD20120 $MC_TOOL_RESET_VALUE	刀具 - 启动（复位/零件程序结束）时的长度补偿
MD20121 $MC_TOOL_PRESEL_RESET_VALUE	复位时的预选刀具

机床数据	含义
MD22550 $MC_TOOL_CHANGE_MODE	使用 M 功能而不是 T 功能换刀
MD22560 $MC_TOOL_CHANGE_M_CODE	用于换刀的 M 功能
MD20110 $MC_RESET_MODE_MASK	确定复位/零件程序结束后控制系统的初始设置
MD20112 $MC_START_MODE_MASK	确定 NC 启动时控制系统的初始设置

19.3.4 T 号编程

当“平面 D 号结构”功能有效时会继续计算 NC 地址 T，即在显示中，编程的 T 号和有效的 T 号是一样的。 但 NC 确定的 D 号与所编程的 T 值无关。

NC 可识别出每个通道的一根主主轴 （通过机床数据可以设定主轴编号）。 补偿和指令 M6（换刀）只能参照主主轴进行计算。

T 的地址扩展会被视为主轴编号（例如 T2 = 1； 应当在主轴 2 上选择刀具 1 ），只有当主轴 2 为主主轴时，换刀才能被识别。

19.3.5 M6 编程

MD22550 和 MD22560

NC 可识别出每个通道的一根主主轴 （通过机床数据可以设定主轴编号）。补偿和指令 M6（换刀）只能参照主主轴进行计算。

通过机床数据来确定是否使用 M 功能来完成换刀指令：

MD22550 $MC_TOOL_CHANGE_MODE （M 功能时新的刀具补偿）

此时 T 为刀具准备指令。

通过机床数据来确定用于换刀的 M 功能名称：

MD22560 $MC_TOOL_CHANGE_M_CODE （用于换刀的 M 功能）

标准设置为 M6。 M6 的地址扩展被视为主轴编号。

示例

定义了两根主轴，主轴 1 和主轴 2，分别是：

```
MD20090 = 2          ; 主轴 2 为主主轴。
M6                   ; 需要换刀，指令针对主主轴
M1= 6                ; 不换刀，因为主轴 2 为主主轴
M2 = 6               ; 进行换刀，因为主轴 2 为主主轴
```

19.3.6 程序测试

MD20110

想采用同下面一样的有效刀具和刀具补偿，可以使用机床数据进行设置：

MD20110 $MC_RESET_MODE_MASK, 位 3（确定复位/零件程序结束后控制系统的初始设置）

值		含义
位 3	= 1	来自测试运行中最后完成的测试程序
	= 0	来自程序测试开始之前最后完成的程序

前提条件

位 0 和 6 必须由机床数据来设置：

MD20110 $MC_RESET_MODE_MASK, 位 3（确定复位/零件程序结束后控制系统的初始设置）

19.3.7 刀具管理或者“平面 D 号结构”

特性

刀具管理

NC 上激活的刀具管理在下列假设条件基础上工作：

1. 在刀库中管理刀具。
2. 刀沿被监控，如达到极限值会导致刀具被禁用。
3. 姊妹刀具为了便于刀具选择，仅使用刀具名称进行编程。NC 选择正确的刀具。

说明

该功能在 SINUMERIK 828D 上仅作为选件提供！

这意味着只有在定义了具体刀具并且该刀具由 NC 负责管理时，使用刀具管理才有意义。

平面 D 号结构

使用平面 D 号结构意味着，在 NC 外进行刀具管理（PLC）并且不参考 T 号。

刀具管理或者“平面 D 号结构”

同时使用刀具管理（NC）和“平面 D 号结构”（PLC）没有意义，因为采用哪种刀具管理方式的主要标准在于**时间**。这只有在由 NC 执行管理任务时才能实现。使用“平面 D 号结构”时无法实现。

说明

如果刀具管理和“平面 D 号结构”都被激活，则“刀具管理”具有更高的优先级。

19.4 刀具刀沿

19.4.1 简介

刀具刀沿

刀沿通过下列数据进行明确说明：

- 刀具类型 （带柄铣刀，钻头，等等）
- 几何尺寸说明
- 工艺说明

刀具参数

每个刀沿的几何尺寸说明、工艺说明和刀具类型都对应有一个刀具参数。

不同刀具类型有下列刀具参数可用：

刀具参数	含义	注释
1	刀具类型	
2	刀沿位置	用于车刀或在轮廓刀具 2D 刀具半径补偿时用于铣刀/磨具
几何数据 - 刀具长度		
3	长度 1	
4	长度 2	
5	长度 3	
几何尺寸 - 刀具形状		
6	半径 1/长度 1	在 3D 端面铣削时
7	长度 2	在 3D 端面铣削时
8	半径 1	在 3D 端面铣削时
9	半径 2	在 3D 端面铣削时
10	角 1 / 最小的极限角	在 3D 端面铣削时 轮廓刀具的 2D 半径补偿

刀具参数	含义	注释
11	角 2 / 最大的极限角	在 3D 端面铣削时 轮廓刀具的 2D 半径补偿
磨损尺寸 - 刀具长度		
12	长度 1	
13	长度 2	
14	长度 3	
磨损尺寸 - 刀具形状		
15	半径 1/长度 1	在 3D 端面铣削时
16	长度 2	在 3D 端面铣削时
17	半径 1	在 3D 端面铣削时
18	半径 2	在 3D 端面铣削时
19	角 1 / 最小极限角度	在 3D 端面铣削时 轮廓刀具的 2D 半径补偿
20	角 2 / 最大极限角度	在 3D 端面铣削时 轮廓刀具的 2D 半径补偿
基本尺寸 / 适配器尺寸		
21	基本长度 1	
22	基本长度 2	
23	基本长度 3	
工艺		
24	后角	仅用于车刀
25	预留*	

* “预留”表示该刀具参数不能使用并保留用于扩展功能。

3D 端面铣削

在 3D 端面铣削时，铣刀类型 111， 120， 121， 130， 155， 156 和 157 会经过特殊处理，其中可能会计算刀具参数 1-23。

文档

有关不同刀具类型的更多信息参见：

- 功能手册之基本功能分册；刀具补偿（W1），章节：“刀具类型 （刀具参数）”
- 编程手册之基本原理分册；章节：“刀具补偿” > “刀具类型列表”
- 功能手册之特殊功能分册； 3D 刀具半径补偿 （W5）

19.4.2 刀具参数 1：刀具类型

含义

刀具类型（3 位数）确定了刀具的类型。 选择了刀具类型会预先确定其他的数据如几何尺寸、磨损尺寸或基本尺寸。

条件

参数“刀具类型”的设置规定为：

- 必须为每个刀沿指定刀具类型。
- 刀具类型只允许采用规定的值。
- 刀具类型“0”表示未定义有效刀具。

刀具类型和刀具参数

在下表中列出了不同的刀具类型以及重要的刀具参数。 用于特定刀具类型的刀具参数用“x”进行标识：

最重要的可用刀具参数表（只有刀具类型是必要的）																			
1	刀具参数 $TC_DP	2	3	4	5	6	7	8/9	12/13/14	15/16	21/22/23	24	TPG 3	TPG 4/5	TPG 6	TPG 7	TPG 8
刀具类型	通过 NCK 计算	刀沿位置	几何数据 - 长度 1	- 长度 2	- 长度 3	几何数据 - 半径 1	槽宽/ - 半径 2	超出高度/ - 长度 4/5	磨损 - 长度 1/2/3	磨损 - 半径 1/2	适配/基础尺寸 - 长度 1/2/3	后角	- 最小砂轮半径	- 最小/当前砂轮宽度	- 最大转速	- 最大圆周速度	- 斜砂轮角度
铣刀和钻刀（所有其它）																			
100	铣刀，符合 CLDATA*[1]		x	x	x	x			x	x	x						
110	球头铣刀		x	x	x	x			x	x	x						
120	立铣刀		x	x	x	x			x	x	x						
121	立铣刀（带刀尖倒圆）		x	x	x	x	x		x	x	x						
130	卧铣刀		x	x	x	x			x	x	x						
131	角度铣刀，带角度倒圆		x	x	x	x	x		x	x	x						
140	面铣刀		x	x	x	x			x	x	x						
145	螺纹铣刀		x	x	x	x			x	x	x						
150	盘形铣刀		x	x	x	x			x	x	x						
155	截锥铣刀		x	x	x	x			x	x	x						
200	麻花钻		x	x	x	x			x		x						
205	整体钻头		x	x	x	x			x		x						
210	镗杆		x	x	x	x			x		x						
220	中心钻		x	x	x	x			x		x						
230	尖头锪钻		x	x	x	x			x		x						
231	平头锪钻		x	x	x	x			x		x						
240	螺纹丝锥		x	x	x	x			x		x						
241	细螺纹丝锥		x	x	x	x			x		x						
242	惠氏螺纹丝锥		x	x	x	x			x		x						
250	铰刀		x	x	x	x			x		x						
磨具和车刀（400 - 599）																			
400	周边磨砂轮	x	x	x	x	x			x	x	x					x	x
401	周边磨削砂轮，带有监控	x	x	x	x	x			x	x	x		x	x	x	x	x
403	与 401 相似但是没有 SUG* 基础尺寸[2]	x	x	x	x	x			x	x	x		x	x	x		x
410	花盘	x	x	x	x	x			x	x	x					x	
411	花盘，带有监控	x	x	x	x	x			x	x	x		x	x	x	x	
413	与 411 相似但是没有 SUG* 基础尺寸[2]	x	x	x	x	x			x	x	x		x	x	x		
490	修整器	x	x	x	x	x			x	x	x						
500	粗车刀	x	x	x	x	x			x	x	x	x					
510	精车刀	x	x	x	x	x			x	x	x	x					
520	车槽刀	x	x	x	x	x			x	x	x	x					
530	切断刀	x	x	x	x	x			x	x	x	x					
540	螺纹车刀	x	x	x	x	x			x	x	x	x					
550	蘑菇状成型车刀/成型车刀（WZV）	x	x	x	x	x			x	x	x	x					
特殊刀具（700）																			
700	槽锯		x	x	x	x	x	x	x	x	x						

CLDATA*[1] "刀具位置数据" — 刀具位置数据符合 DIN 66215，
B1 1SUG*[2] — 砂轮圆周速度

说明

在车刀组别中刀具类型没有意义。

尤其是磨具（400-499）中允许使用上文未列举出的编号。

刀具补偿数据

刀具补偿数据（TOA 数据）保存在系统变量中。

示例： 刀具类型：槽锯 （类型 700）

	几何尺寸	磨损	基准	单位
长度补偿				
长度 1	$TC_DP3	$TC_DP12	$TC_DP21	毫米
长度 2	$TC_DP4	$TC_DP13	$TC_DP22	毫米
长度 3	$TC_DP5	$TC_DP14	$TC_DP23	毫米
锯片补偿				
直径 d	$TC_DP6	$TC_DP15		毫米
槽宽 b	$TC_DP7	$TC_DP16		毫米
超出规定范围 k	$TC_DP8	$TC_DP17		毫米

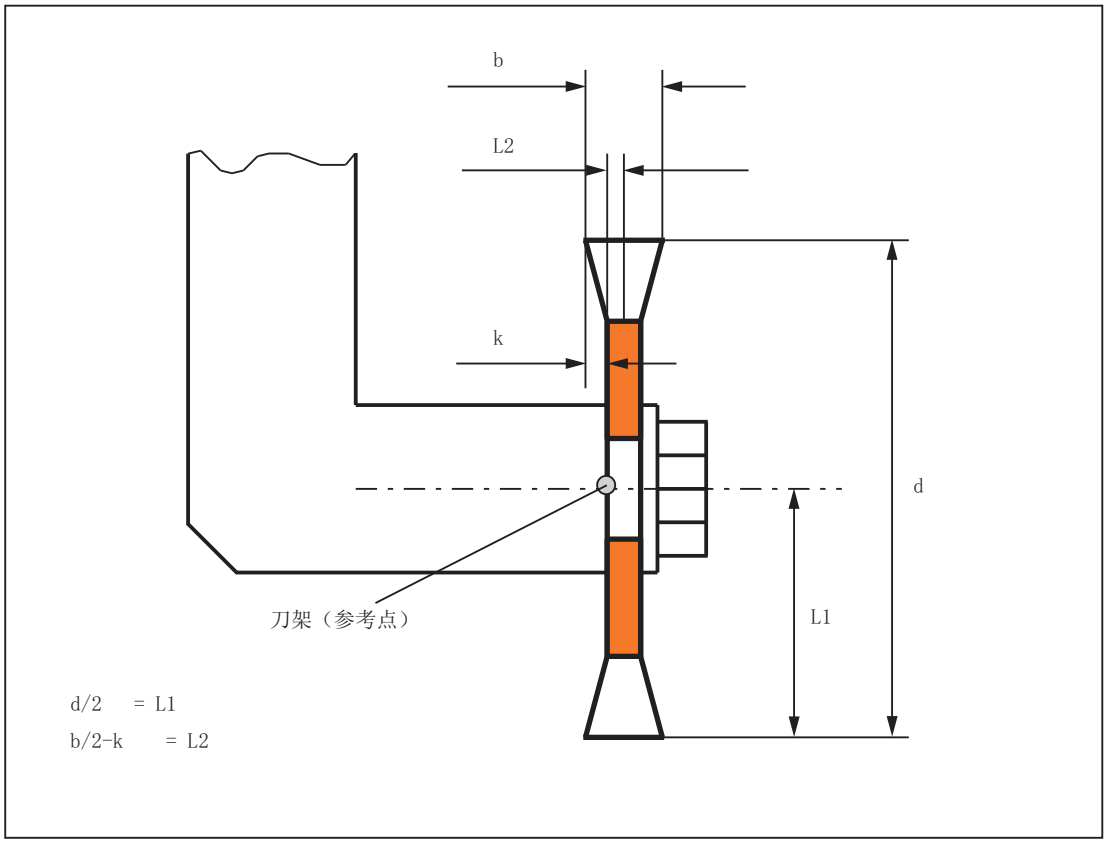


图 19-3 槽锯的几何尺寸 （类似于角头铣刀）

锯片的宽度在刀具半径补偿（G40 至 G42）时按如下方式考虑：

指令	含义
G40	无锯片补偿
G41	左侧锯片补偿
G42	右侧锯片补偿

19.4.3 刀具参数 2：刀沿位置

含义

刀沿位置用于说明刀尖 P 相对于刀沿中心点 S 的位置。 它被记录在刀具参数 2 中。

在车刀上（刀具类型 5xx），需要使用刀沿位置与刀沿半径（刀具参数 8）来共同计算刀具半径补偿。

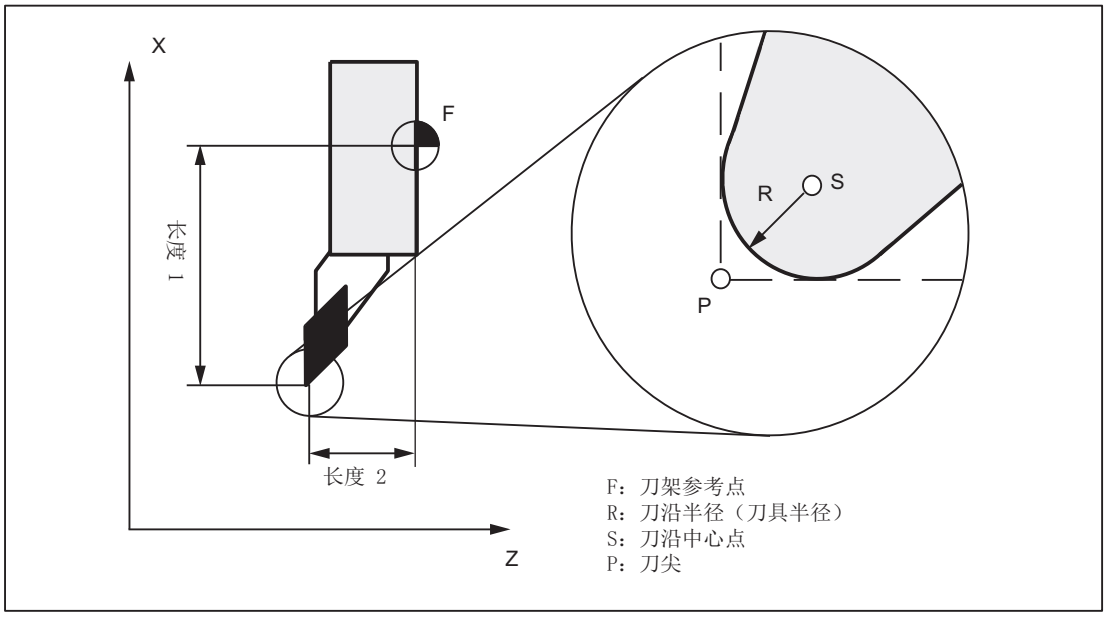


图 19-4　　使用车刀时的尺寸： 车刀

刀沿位置参数值

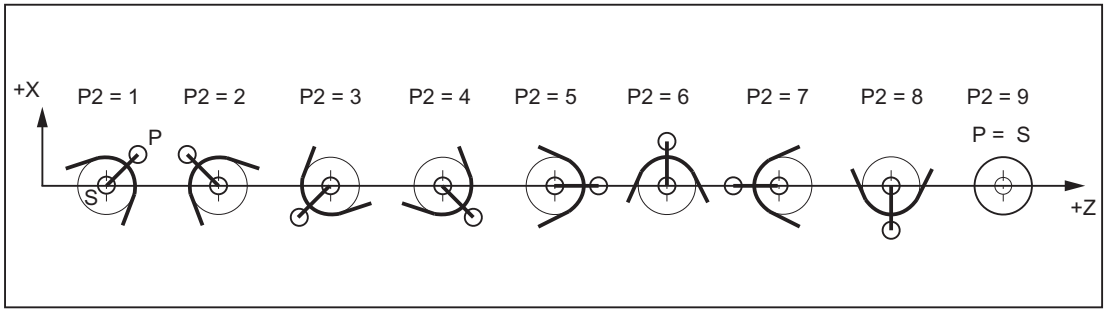


图 19-5　　刀具参数 2 （P2）： 在车削中心后侧加工时

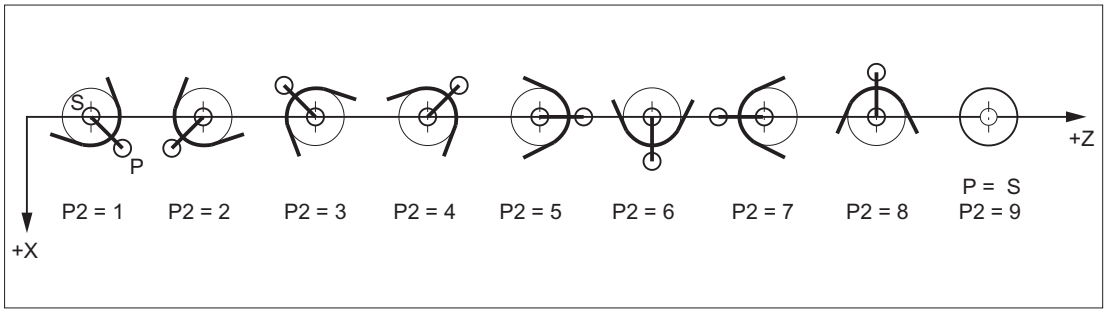


图 19-6　　刀具参数 2 （P2）： 在车削中心前侧加工时

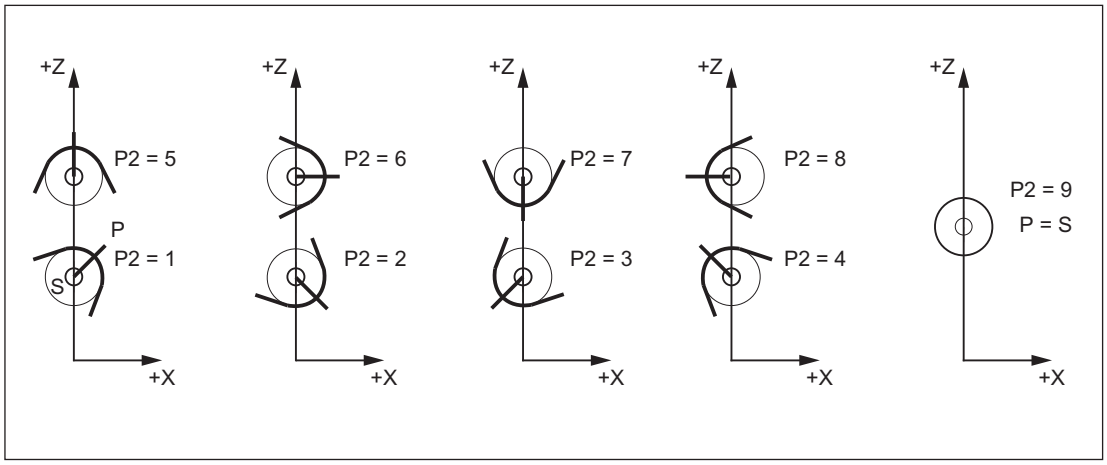


图 19-7　　刀具参数 2 （P2）： 立式车床上的刀沿位置

特殊性

- 如果为了计算刀具长度补偿不用点 P 而用刀沿中心点 S 来作为参考点，则要输入标识 9 作为刀具的刀沿位置。
- 不允许使用标识 0 （零）作为刀具的刀沿位置。

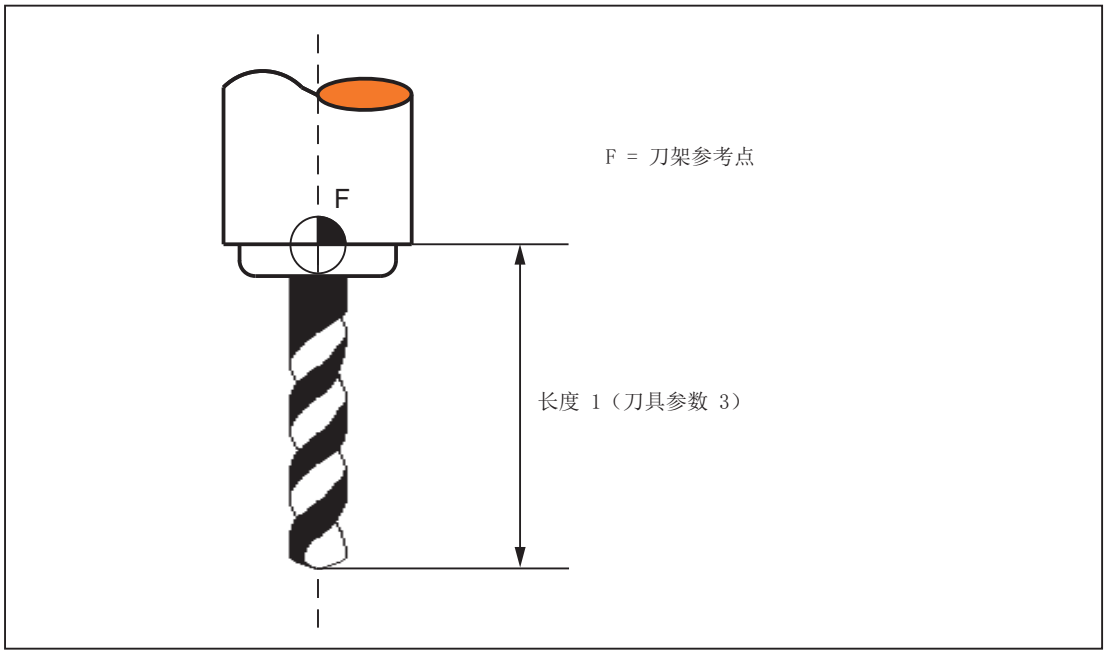
19.4.4　刀具参数 3 - 5： 几何数据 - 刀具长度

含义

进行刀具长度补偿时需要刀具的长度。 在刀具参数 3 至 5 中输入 长度 1 至 3。 根据刀具类型至少需要下列长度说明：

刀具类型	所需的刀具长度
刀具类型 12x, 140, 145, 150：	刀具长度 1
刀具类型 13x：	刀具长度 1-3 （根据平面 G17-G19）
刀具类型 2xx：	刀具长度 1
刀具类型 5xx：	刀具长度 1-3

示例： 麻花钻 （刀具类型 200），含刀具长度 1 （刀具参数 3）



说明

不管是哪种刀具类型，系统总是计算三根几何轴上的所有三个刀具参数 3 至 5 （刀具长度 1 至 3） 。

如果为某一刀具类型在刀具参数 3 至 5 中输入了更多刀具长度，系统总会计算几何轴上这些额外的刀具长度，而 不产生报警。

特点

首先将“几何尺寸 -刀具长度补偿”（刀具参数 3 至 5）和“磨损尺寸 -刀具长度补偿”（刀具参数 12 至 14）相加来确定有效的刀具尺寸。 为了计算几何轴上整个的刀具长度补偿还要加上“基本尺寸/适配器尺寸-刀具长度补偿”。

文档

关于如何在刀具参数 3 至 5（刀具长度 1 至 3）中输入刀具尺寸（长度）以及在如何在三根几何轴上计算参见 → 操作手册。

19.4.5 刀具参数 6 - 11： 几何尺寸 - 刀具形状

含义

使用刀具参数 6 至 11 来确定刀具的形状。这些数据用于“几何尺寸-刀具半径补偿”。

在多数情形下仅使用刀具参数 8（刀具半径 1）：

<table>
<tr><th>刀具参数</th><th>含义</th><th colspan="2">使用</th></tr>
<tr><td>6</td><td>刀具长度 1</td><td colspan="2">未使用</td></tr>
<tr><td>7</td><td>刀具长度 2</td><td colspan="2">未使用</td></tr>
<tr><td rowspan="4">8</td><td rowspan="4">刀具半径 1</td><td colspan="2">必须为下列刀具类型给定刀具半径：</td></tr>
<tr><td>刀具类型</td><td>含义</td></tr>
<tr><td>1xx</td><td>铣刀</td></tr>
<tr><td>5xx</td><td>车刀
对于车刀还需要给定刀具参数 2：刀沿位置 (页 1624)。</td></tr>
<tr><td colspan="3">提示
刀具类型 2xx：无需为钻头给定刀具半径。</td></tr>
<tr><td>9</td><td>刀具半径 2</td><td colspan="2">未使用</td></tr>
<tr><td>10</td><td>刀具角 1</td><td colspan="2">未使用</td></tr>
<tr><td>11</td><td>刀具角 2</td><td colspan="2">未使用</td></tr>
</table>

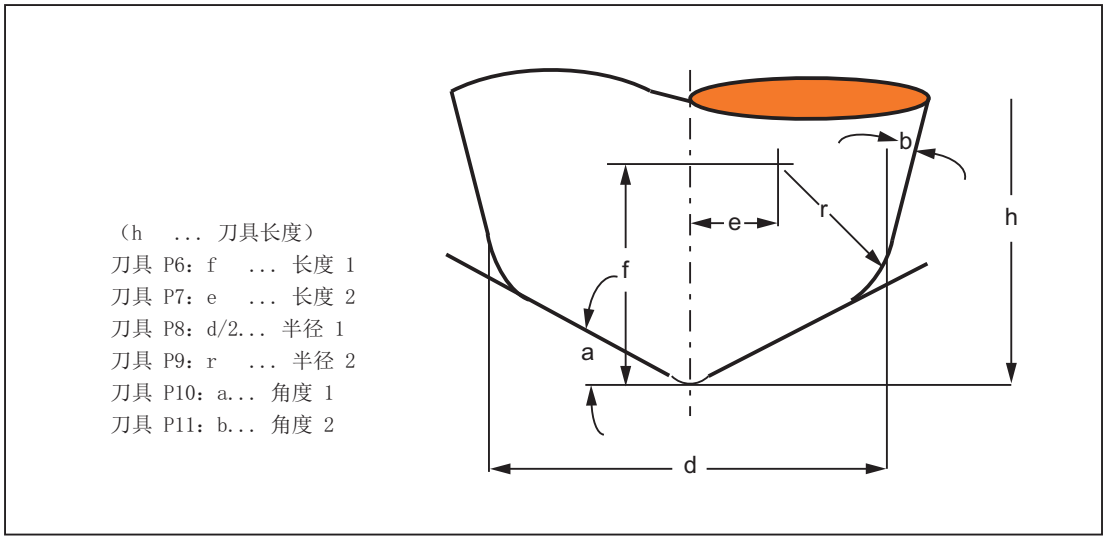
轮廓刀具的 2D 刀具半径补偿

可以规定最小和最大极限角来定义带有多个刀沿的轮廓刀具。两个极限角分别指刀沿中心点到刀沿参考点的矢量并以逆时针方向计数。

刀具角 1	每个刀沿的最小极限角
刀具角 2	每个刀沿的最大极限角

3D 端面铣削

在 3D 端面铣削中描述刀具时，需要给定所有刀具参数 6 至 11：



文档

关于如何在刀具参数 6 至 11 中输入刀具形状（用于刀具半径补偿的半径）以及如何计算三根几何轴上的“几何尺寸 -刀具半径补偿”参见：

- 编程手册之基本原理分册；章节：“刀具补偿” > “2½ D 刀具补偿”
- 功能手册之特殊功能分册； 3D 刀具半径补偿 （W5）

关于 3D 端面铣削参见：

- 编程手册之工作准备分册；章节：“转换” > “三轴、四轴和五轴转换 （TRAORI）”

19.4.6 刀具参数 12 - 14： 磨损尺寸 - 刀具长度

含义

当使用“几何尺寸-刀具长度补偿”（刀具参数 3 至 5）来设定刀具尺寸时，可以使用“磨损尺寸 -刀具长度补偿”对有效刀具尺寸的变化进行补偿。

有效刀具尺寸变化的原因有：

- 刀具测量仪器上的刀架上与机床上刀架之间存在不同
- 刀具在加工期间产生磨损
- 定义精加工余量

有效的刀具尺寸

将“几何尺寸 -刀具长度补偿”（刀具参数 3 至 5）和“磨损尺寸 -刀具长度补偿”（刀具参数 12 至 14）相加，比如：“几何尺寸 -刀具长度 1” 加上“ 磨损尺寸 -刀具长度 1”， 从而得出有效的刀具尺寸。

19.4.7 刀具参数 15 - 20： 磨损尺寸 - 刀具形状

含义

当使用“几何尺寸 -刀具半径补偿”（刀具参数 6 至 11）来设定刀具形状时，可以使用“磨损尺寸 -刀具半径补偿”对有效刀具形状的变化进行修正。

有效刀具尺寸变化的原因有：

- 刀具在加工期间产生磨损
- 定义精加工余量

有效的刀具形状

将“几何尺寸 -刀具半径补偿”（刀具参数 6 至 11）和“磨损尺寸 -刀具半径补偿”（刀具参数 15 至 20）相加，比如：“几何尺寸 -刀具半径 1” 加上“ 磨损尺寸 -刀具半径 1”， 从而得出有效的刀具形状。

19.4.8 刀具参数 21 - 23： 基本尺寸 / 适配器尺寸

含义

当刀具夹具（刀具尺寸）的参考点与刀架的参考点不一致时，可以使用基本尺寸/适配器尺寸。

在下列应用中会出现这种情况：

- 刀具与刀具适配器分开测量，但在机床上已经装配在了一起（在刀沿中分别输入刀具尺寸和适配器尺寸）。
- 刀具装入了另一个位置上的第二刀架中（比如： 垂直主轴和水平主轴）。
- 一个刀塔的刀架有不同的位置。

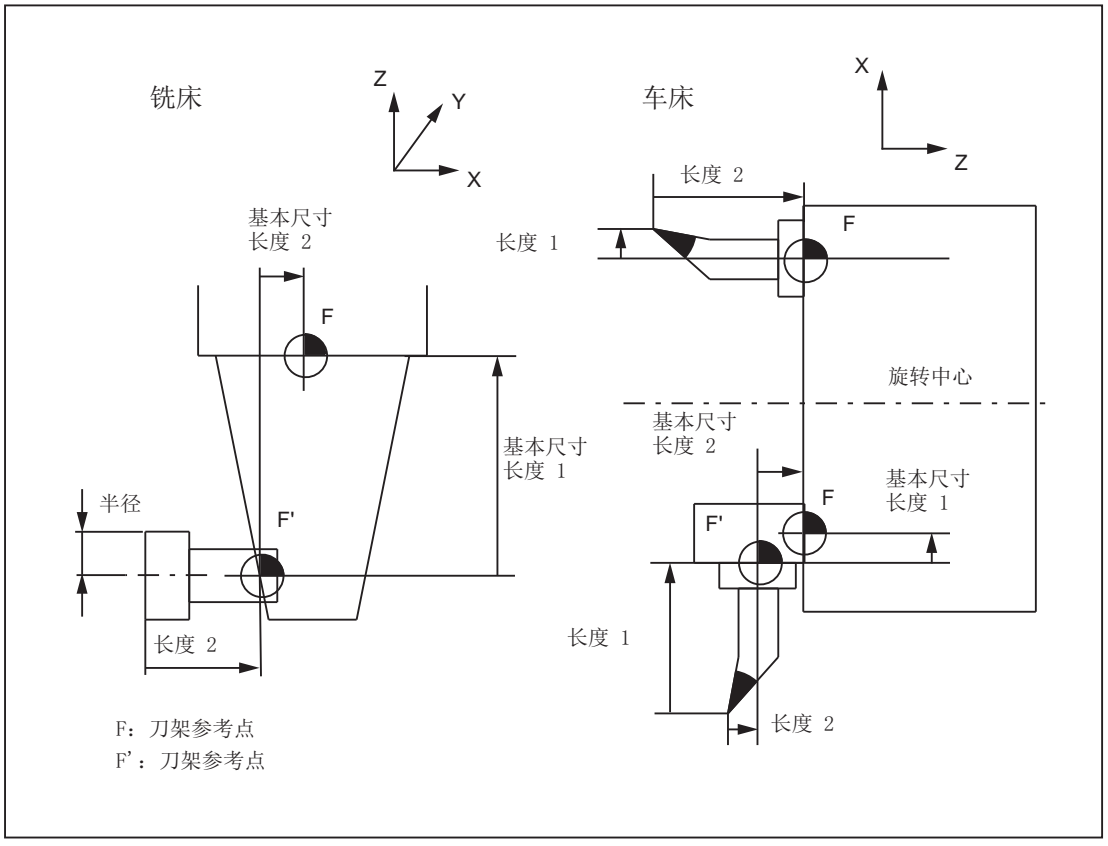


图 19-8 “基本尺寸/适配器尺寸 -刀具长度补偿”的应用示例

基本长度 1 至 3 （刀具参数 21 至 23）

为了能够在三根几何轴上（三维）参照刀架参考点 F' 对刀夹参考点 F 进行修正，全部 3 个基本长度总是有效，不管何种刀具类型无关，也就是说，即使是含一个刀具长度补偿（长度 1）的麻花钻（刀具类型 200）在 3 根轴上也会有基本尺寸/适配器尺寸。

文档

关于详细的“基本尺寸/适配器尺寸 -刀具长度”补偿参见：

- 编程手册之基本原理分册

19.4.9 刀具参数 24：后角

含义

在某些循环中会生成带底切的运动，这些循环会监控当前工作的刀具后角是否会造成轮廓损伤。

取值范围

在刀具参数 24 中输入为后角（0 到 90° 不带符号）作。

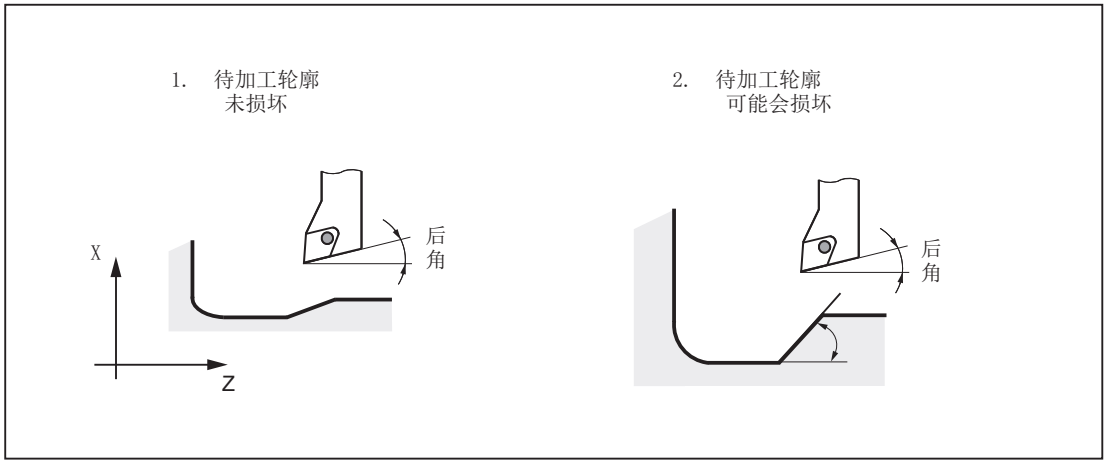


图 19-9 底切时车刀的后角

纵向或横向加工方式

根据纵向或横向加工方式输入不同的后角。 如果纵向加工和横向加工需要使用同一把刀具，则后角不同时必须定义两个刀沿。

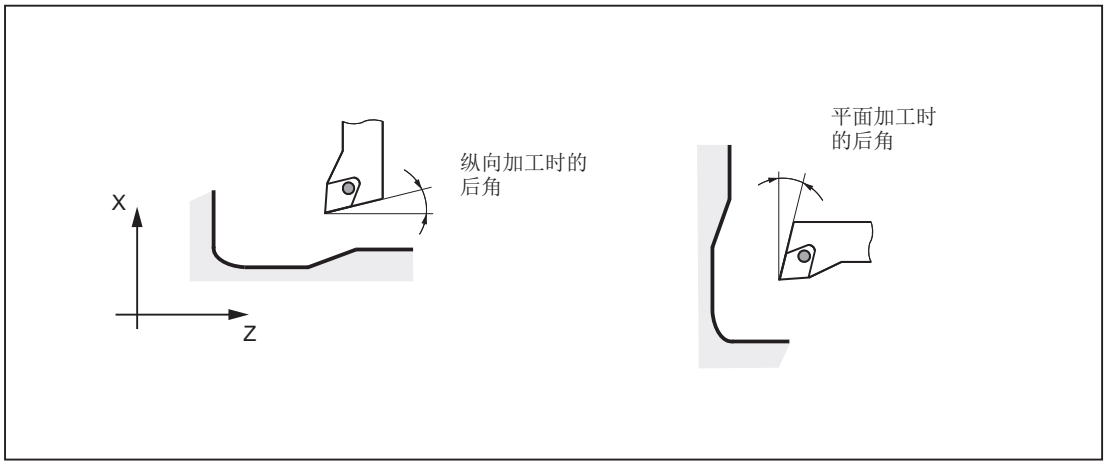


图 19-10　用于纵向和横向加工的后角

说明

如果输入零作为后角（刀具参数 24)， 则在底切时不会对车削循环进行监控。

文档

关于后角的详细说明参见：

- 编程手册 循环

19.4.10 带相关刀沿位置的刀具

对于带相关刀沿位置的刀具需要注意下列事项：

- 使用程序段起点处与终点处刀沿中心构成的直线来计算逼近程序段或退回程序段的交点。刀沿基准点和中心点之间的差值会叠加到该运动中。
  在使用 KONT 逼近或回退时会在对应的线性子程序段中产生叠加。 因此在带和未带相关刀沿位置的刀具上，几何尺寸是相同的。
- 如果刀具补偿生效，而刀沿中心和刀沿基准点之间的距离改变，则在圆弧程序段和含 4 阶以上的有理多项式的程序段中，不允许更换刀具。 使用其它的插补方式中与之相反，即使在坐标转换有效时（例如 TRANSMIT）也允许切换回从前的状态。
- 如果刀具半径补偿使用可变的刀具定向，则从刀沿参考点到刀沿中心的转换，就不能再简单地通过一个零点偏移来实现。因此在 3D 圆周铣削时禁止刀具带相应的刀沿（给出报警）。

说明

对于端面铣削，这一点就无关紧要了，因为反正只有定义过的、不带相关刀沿位置的刀具类型才允许使用。（没有明确允许某刀具类型，则刀具作为有半径参数的球头铣刀处理。刀沿位置被忽略。）

19.5 2D 刀具半径补偿（2D-WRK）

19.5.1 简介

说明

关于刀具半径补偿参见
文档：
编程手册之基本原理分册
只有编程指南描述了刀具半径补偿的完整编程方法及其特例。

为什么进行刀具半径补偿？

在零件程序所编程的工件轮廓（几何数据）应当与生产加工中所使用的刀具无关。 为此需要从一个当前的补偿存储器中获取刀具长度和刀具半径的数值。 通过刀具半径补偿可以使用当前的刀具半径来确定平面内编程轮廓的等距线。

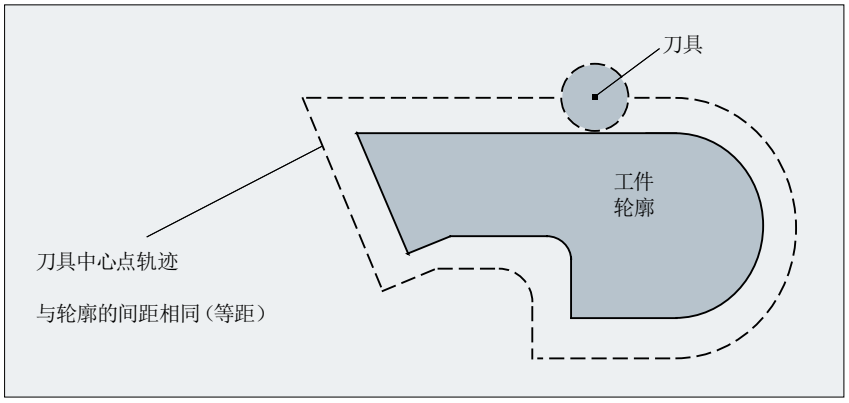


图 19-11　工件轮廓（几何数据）以及等距线

平面内的刀具半径补偿

在当前平面内（G17 至 G19）使用下列插补方式时刀具半径补偿有效：

• 线性插补	...	G0，G1
• 圆弧插补	...	G2，G3，CIP
• 螺旋线插补	...	G2，G3
• 样条插补	...	ASPLINE，BSPLINE，CSPLINE
• 多项式插补	...	POLY

19.5.2 选择刀具半径补偿 （G41/G42）

补偿方向

刀具半径补偿会计算出一条与编程轮廓等距的轨迹。 可以按运行方向在所编程轮廓的左侧或右侧进行补偿。

指令	含义
G41	按运行方向在左侧进行刀具半径补偿
G42	按运行方向在右侧进行刀具半径补偿
G40	取消刀具半径补偿

中间程序段

当刀具半径补偿有效时，通常程序段内只能编程当前平面内的几何轴位置。 尽管如此，在刀具半径补偿有效时也可以编程单个中间程序段。 中间程序段是指那些不包含当前平面内几何轴位置的程序段：

- 进给轴上的位置
- 辅助功能
- 等

中间程序段的最大数量可以在机床数据中进行设定：

MD20250 $MC_CUTCOM_MAXNUM_DUMMY_BLOCKS （刀具半径补偿时无运行动作的程序段数量）

特殊性

- 只能在含 G0 （快速移动）或 G1（线性插补）的程序段中选择刀具半径补偿。
- 最迟必须在选择刀具半径补偿的程序段中换入刀具（T 功能）并激活刀沿/刀具补偿（D1 至 D9）。
- 当刀沿/刀具补偿为 D0 时不选择刀具半径补偿。
- 如果在选择刀具半径补偿时只编程了平面内的一根轴，则系统会自动添加平面内的第二根轴（最后的编程位置）。
- 如果在选择刀具半径补偿的程序段中没有编程当前平面内的任何几何轴，则选择无效。

- 如果在一条程序段中先选择了刀具半径补偿，然后又取消了刀具半径补偿 （G40)， 则选择无效。
- 在选择刀具半径补偿时逼近特性通过指令 NORM/KONT 来确定。

19.5.3 逼近和回退特性 （NORM/KONT/KONTC/KONTT）

NORM 和 KONT

使用指令 NORM 和 KONT 可以确定逼近特性（选择刀具半径补偿时使用 G41/G42）和回退特性（选择刀具半径补偿时使用 G40）：

指令	含义
NORM	起点/终点处的正常设置 （初始设置）
KONT	在起点/终点处绕行轮廓
KONTC	恒定曲率方式逼近/回退
KONTT	恒定切线方式逼近/回退

特殊性

- KONT 与 NORM 的不同之处仅在于刀具的起始位置位于轮廓后侧。

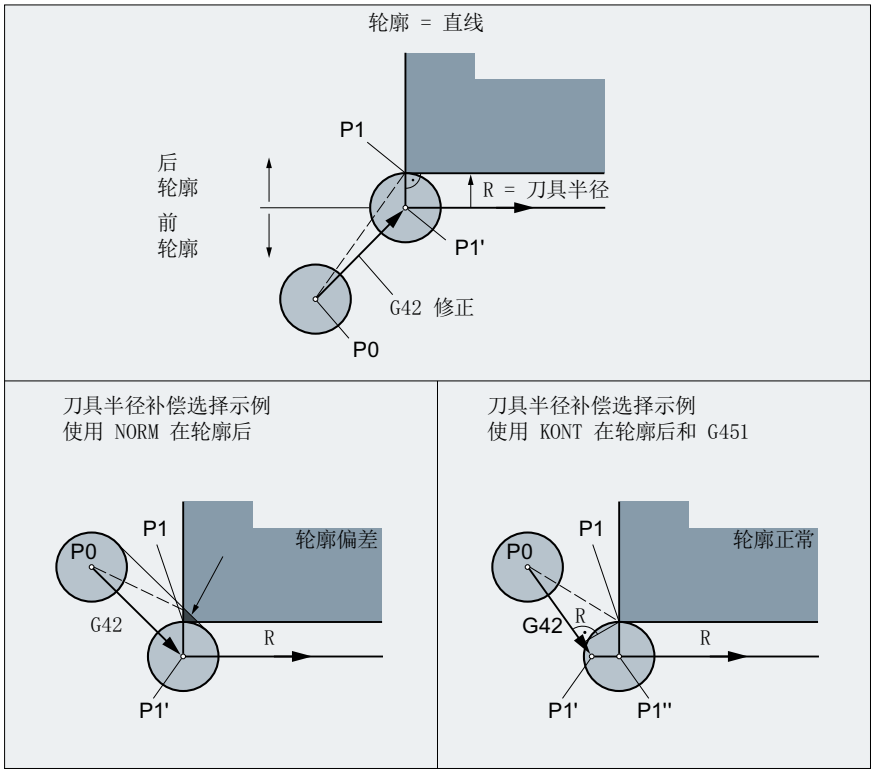


图 19-12　示例：选择刀具半径补偿，KONT 或 NORM 位于轮廓前侧和后侧

- KONT 和 G450/G451 （外角处的特性）共同起作用，确定刀具半径补偿时的逼近和回退特性。
- 在取消刀具半径补偿时通过指令 NORM/KONT 来确定回退特性。

前提条件

使用下列两种方案时逼近或回退程序段为多项式。因此它只在已使能多项式插补的控制方案中可供使用：

- KONTT
  使用 KONTT 以恒定切线方式逼近轮廓或从轮廓回退。程序段过渡的曲率通常并不恒定。
- KONTC
  使用 KONTC 从逼近程序段过渡或过渡到回退程序段时不仅切线恒定，而且曲率也恒定，以至于在这种情况下进行激活/取消时 **加速跳转不会**再出现。

虽然 KONTC 同样包括有 KONTT 的特性，但只有切线恒定的方案 KONTT 可供使用，因为所要求的曲率恒定在 KONTC 时可能会偶尔生成不希望出现的轮廓。

轴

在**三根**轴上都会遵守恒定条件。由此可以在逼近/回退时同时编程一个垂直于补偿平面的行程分量。

含 KONTT/KONTC 的原始逼近或回退程序段只允许为**线性程序段**。这些编程的线性程序段会在内部转换成对应的多项式曲线。

排除

KONTT 和 KONTC 不能用于 3D 刀具半径补偿 （CUT3DC， CUT3DCC， CUT3DF） 。

如果仍然编程了这些指令，控制系统内部会切换至 NORM，不输出故障信息。

举例 KONTC

在下面两张图中显示了用于恒定曲率逼近和回退的典型应用：

从圆心开始，沿整圆运行。这时在逼近程序段的终点处，其方向和曲率半径和后一个圆弧相同。在这两个逼近/回退程序段中，同时在 Z 方向进给。

相应的 NC 程序段如下：

```
$TC_DP1[1,1]=121                                铣刀
$TC_DP6[1,1]=10                                 半径 10 毫米
N10 G1 X0 Y0 Z60 G64 T1 D1 F10000
N20 G41 KONTC X70 Y0 Z0
N30 G2 I-70                                     整圆
N40 G40 G1 X0 Y0 Z60
N50 M30
```

说明:

在该示例中，在 XY 平面内加工一个半径为 70 毫米的整圆。因为刀具的半径为 10 毫米，所以得出的刀具中心点轨迹的半径为 60 毫米。起点/终点位于 X0 Y0 Z60，这样可以在轮廓平面内逼近/回退的同时完成 Z 方向上的动作。

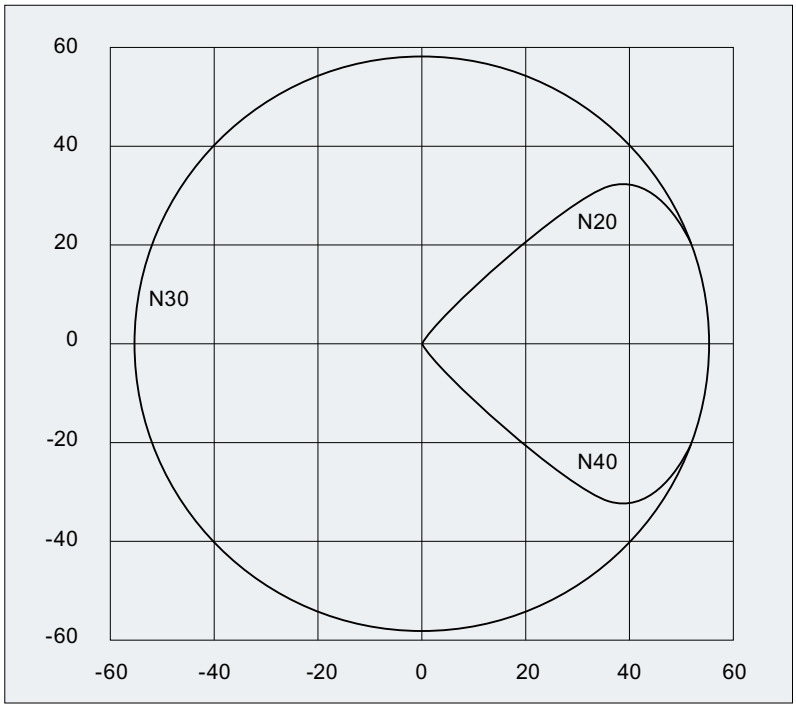


图 19-13　内部加工一个整圆时恒定曲率方式逼近和回退：投影至 XY 平面

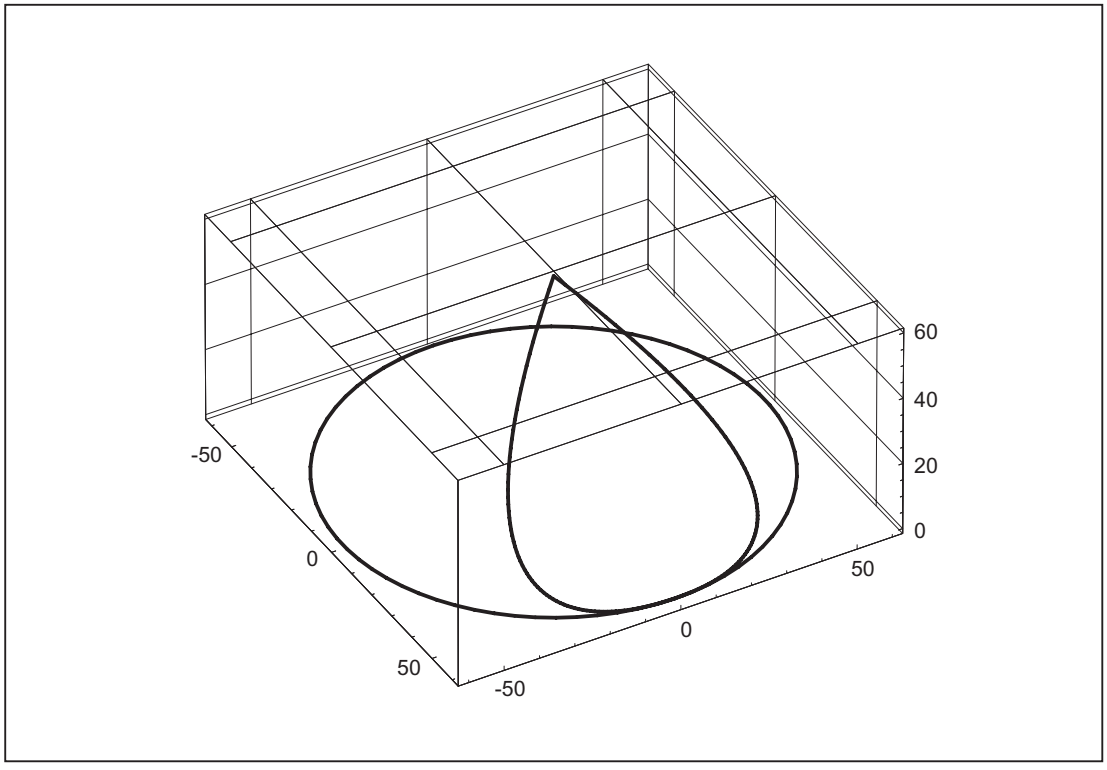


图 19-14　内部加工一个整圆时恒定曲率方式逼近和回退：立体图

KONTT 和 KONTC 的比较

在下图中显示了使用 KONTT 和 KONTC 时不同的逼近与回退特性。使用半径 20 毫米的刀具对半径 20 毫米，圆心为 X0 Y40 的圆弧进行外部补偿。因此就形成一个半径为 40 毫米的刀具中心点的圆弧运动。回退程序段的终点在 X40 Y30。圆弧程序段和逼近程序段之间的过渡位于零点。由于在 KONTC 中要求恒定曲率，回退程序段首先执行负向 Y 分量的运行。通常不希望出现该情况。使用 KONTT 编程的回退程序段无此特性。但是在这种情况下，在程序段过渡处会出现一个加速度跃变。

如果 KONTT 或 KONTC 程序段不是回退程序段，而是逼近程序段，则会生成完全相同的轮廓，该轮廓仅以相反的方向运行，即 **逼近与回退的特性是对称的。**

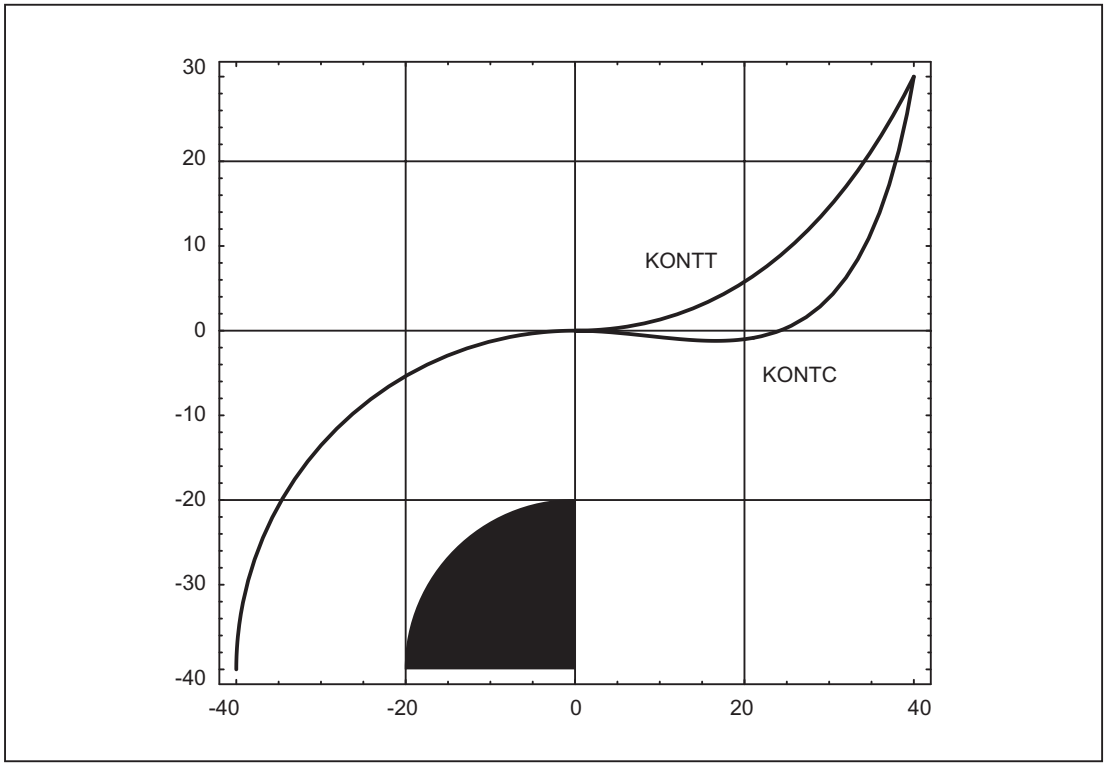


图 19-15　KONTT 和 KONTC 的区别

说明

从图中可以明显看出，使用 KONTC 时，在四分圆后紧跟着一条直线，从 X0 Y0 到 X20 Y-20，该直线会损坏轮廓。

19.5.4 平滑逼近和退回

19.5.4.1 功能

含义

“平滑逼近和回退（SAR）”功能主要用于切向逼近轮廓的起点，而不管出发点在何处。

此时可以通过一系列补充参数来改变逼近特性，使之与特殊的要求相匹配。

“平滑逼近”和“平滑回退”这两个功能几乎是相互对称的。 因此下面只对逼近进行详细说明而对于回退则只指明其不同之处。

子运动

在平滑逼近和回退时最多可以进行涉及下列位置的 4 个子运动：

- 运动的起点 P_0
- 中间点 P_1， P_2, 和 P_3
- 终点 P_4

点 P_0, P_3 和 P_4 始终是定义过的。 中间点 P_1 和 P_2 可以省略，视参数设定和几何数据而定。

在回退时按相反的顺序运行，即开始于 P_4 并结束于 P_0 。

19.5.4.2 参数

平滑逼近和回退功能的特性最多可以通过 9 个参数来确定。

用于确定逼近或回退轮廓的非模态 G 指令

该 G 指令不能省略。

- G147:沿一条直线逼近
- G148:沿一条直线回退
- G247:沿一个四分之一圆弧逼近
- G248:沿一个四分之一圆弧回退

- G347:沿半圆逼近
- G348:沿半圆回退

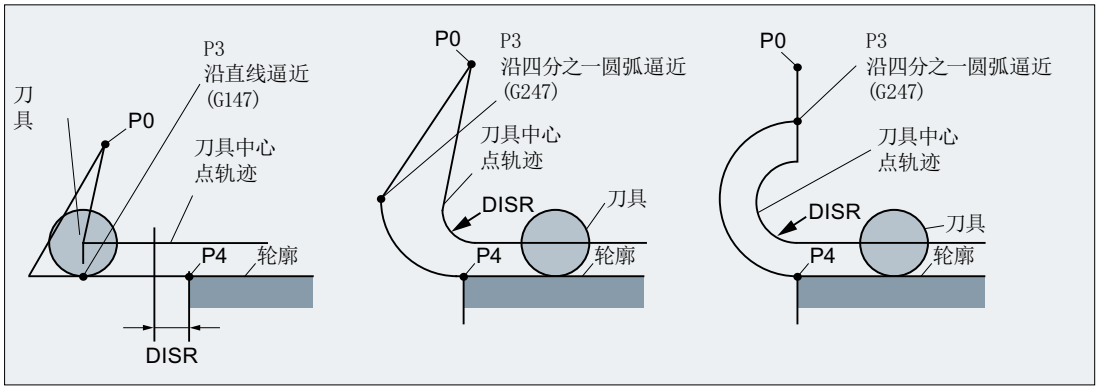


图 19-16　逼近的特性与 G147 至 G347 以及 DISR 有关（当同时激活刀具半径补偿时）

用来确定逼近或回退方向的模态 G 指令

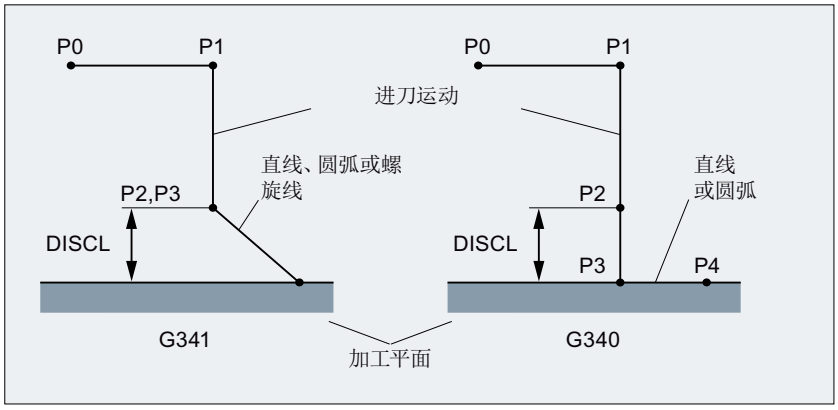
只有当逼近轮廓为四分圆或半圆时，该 G 指令才有意义。可以按下列方式确定逼近或回退方向：

- G140:
  借助有效的刀具半径补偿来确定逼近或回退方向。（G140 为初始设定值。）
  在刀具半径为正时：
  - G41 有效 → 从左侧逼近
  - G42 有效 → 从右侧逼近
    如果没有激活刀具半径补偿（G40），则特性与 G143 时相同。在这种情况下不会输出报警。如果有效刀具的半径为 0，则将其视为正刀具半径来确定逼近或回退方向。
- G141:
  从左侧逼近轮廓或向左侧离开轮廓。
- G142:
  从右侧逼近轮廓或向右侧离开轮廓。
- G143:
  自动确定逼近方向，即从一侧逼近轮廓，而相对于后续程序段起点（P_4）切线的起始点位于该侧。

说明

在**回退**时会相应使用前面程序段的终点切线。如果在回退时未对终点进行编程，即应当通过隐含方式确定终点时，则不允许在回退时使用 G143，因为逼近侧与终点位置之间是相互依赖的关系。如果在这种情况下还是编程了 G143，则会输出报警。这同样也适用于当 G140 有效时因刀具半径补偿未被激活而自动切换至 G143 的情况。

模态 G 指令（G340，G341），确定了如何将起点和终点之间的运行分到各个程序段中



G34 在图中描绘了从 P_0 到 P_4 的典型逼近过程。

0: 如果 G247 或 G347 有效（四分圆或半圆）并且起点 P_3 不位于通过终点 P_4 定义的加工平面内，则会插入螺旋线而不是圆弧。点 P_2 未定义，或者与 P_3 重合。

此时圆弧平面以及螺线轴将通过 SAR 程序段中生效的平面（ (G17 - G19） 来确定，也就是说，在下一个程序段中不会使用起始切线本身来定义圆弧，而是使用其对生效平面的投影。

从 P_0 至点 P_3 的运动分为两段直线，且采用 SAR 程序段前生效的速度。

G34 在图中描绘了从 P_0 到 P_4 的典型逼近过程。

1: P_3 和 P_4 均位于加工平面中，因此在使用 G247 或 G347 时将不会插入螺旋线，而始终插入圆弧。

图 19-17 逼近运行的过程取决于 G340/G341

说明

牵涉到有效加工平面 G17 - G19 的位置时（圆弧平面、螺线轴、垂直于有效加工平面的进给运动），系统会考虑到可能被激活的旋转框架。

DISR

DISR 用来说明逼近直线的长度或逼近圆弧的半径。

沿直线逼近/回退

在使用直线逼近/回退时 DISR 用来说明铣刀边沿与轮廓起点之间的距离，即在刀具半径补偿有效时直线的长度为刀具半径与 DISR 编程值之和。

在以下条件下会发出报警：

- 当 DISR 为负数并且绝对值大于刀具半径时（生成的逼近直线长度小于或等于零）。

沿圆弧逼近/回退

使用圆弧逼近/回退时 DISR 用来说明刀具中心点轨迹的半径。如果刀具半径补偿被激活，则产生一个圆弧，此时刀具中心点轨迹以编程的半径计算生成。

在使用圆弧逼近或回退时会输出报警：

- 当内部生成的圆弧半径为零或负数时。
- 当未对 DISR 进行编程时。
- 当半径的值 ≤ 0 时。

DISCL

DISCL 用来说明点 P2 到加工平面的距离。

如果点 P_2 的位置需要以垂直于圆弧平面的轴上的绝对值设定，则该值必须以 DISCL = AC（....） 的形式编程。

如果未对 DISCL 进行编程，则点 P_1、 P_2 和 P_3 在 G340 时相同并且逼近轮廓由 P_1 至 P_4 构成。

系统会对通过 DISCL 所定义的位于 P_1 和 P_3 之间的点进行监控，也就是说，在所有包含垂直于加工平面分量的运动（进刀运动、从 P_3 向 P_4 逼近）中，这些分量的符号必须相同。不允许出现反向，否则会输出报警。

在识别换向时允许公差，可以通过机床数据来定义：

MD20204 $MC_WAB_CLEARANCE_TOLERANCE（SAR 时换向）

如果 P_2 位于通过 P_1 和 P_3 所定义的范围以外，而偏差值小于或等于所编程公差，则认为 P_2 位于通过 P_1 或 P_3 所定义的平面内。

示例：

使用 G17 时从点 P_1 的位置 Z=20 进行逼近。通过 P_3 所定义的 SAR 平面为 Z=0。因此通过 DISCL 所定义的点必须位于这两点之间。MD20204=0.010。如果 P_2 位于 20.000 和 20.010 之间或者位于 0 和 -0.010 之间，则认为它的编程值为 20.0 或 0.0 。如果 P_2 的 Z 位置大于 20.010 或者小于 -0.010，就会输出报警。

A 在考虑加工平面的情况下依据起点 P_0 和终点 P_4 的相对位置，会沿负方向（逼近时的正常情况）或正方向（回退时的正常情况）完成进刀运动，也就是说例如在使用 G17 时允许终点 P_4 的 Z 分量比起点 P_0 的大。

终点 P_4（或回退时 P_0）的编程通常使用 X...Y...Z...

逼近时允许的终点 P_4 的编程方式

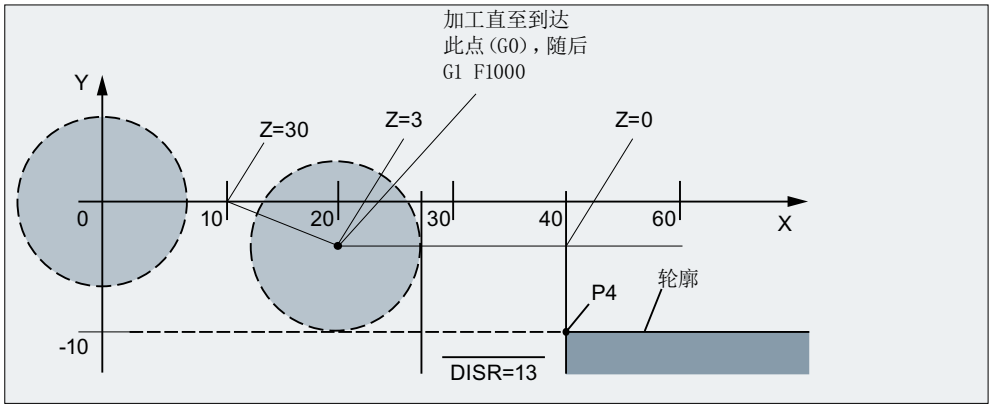
终点 P_4 可在 SAR 程序段中单独编程。

P_4 可以通过下一个运行程序段的终点来确定。

在 SAR 程序段和下一个运行程序段之间可以插入其它的程序段（虚拟程序段），而不插入几何轴运行。

逼近时的终点被当作在 SAR 程序段中自行编程，如果至少对加工平面（X 或 Y 在使用 G17 时)的一根几何轴进行了编程的话。如果在 SAR 程序段中只编程了轴的位置垂直于加工平面（Z 在使用 G17 时），则从 SAR 程序段中取出该分量，而在平面中的位置相反从后续程序段中获取。如果在这种情况下后续程序段中又另外编程了轴与加工平面垂直，则会输出报警。

示例：



程序代码	注释
$TC_DP1[1,1]=120	；铣刀定义 T1/D1
$TC_DP6[1,1]=7	；采用 7 毫米半径的刀具
N10 G90 G0 X0 Y0 Z30 D1 T1	
N20 X10	
N30 G41 G147 DISCL=3 DISR=13 Z=0 F1000	
N40 G1 X40 Y-10	
N50 G1 X50	
...	
...	

N30/N40 可以用以下语句代替：

```
N30 G41 G147 DISCL=3 DISR=13 X40 Y-10 Z0 F1000
```

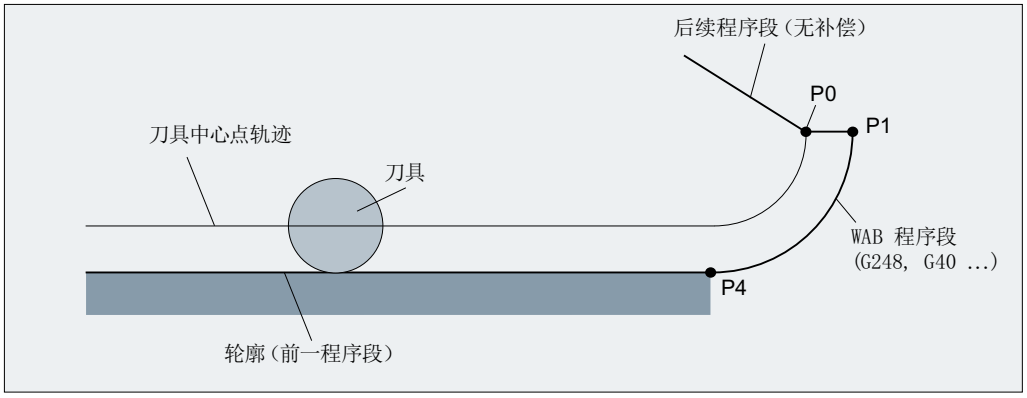
或者：

```
N30 G41 G147 DISCL=3 DISR=13 F1000
N40 G1 X40 Y-10 Z0
```

回退时允许的终点 P_0 的编程方式

终点位置总是从 SAR 程序段中自行获取，与编程了几根轴无关。需要区分下列情况：

1. 在 SAR 程序段中未编程几何轴。
   在这种情况下轮廓结束于点 P_2（或者于点 P_1 当 P_1 与 P_2 重合时）。构成加工平面的轴的位置由回退轮廓产生（直线或圆弧的终点）。轴分量与之垂直并通过 DISCL 来定义。如果这种情况下 DISCL = 0，则要在平面内完整地完成运行。
2. 在 SAR 程序段中仅编程了垂直于加工平面的轴。
   此时轮廓在点 P_1 处终止。其他两根轴的位置通过 1 中所介绍的方式得出。



   使用 SAR 回退并同时取消刀具半径补偿
   若 SAR 程序段同时为刀具半径补偿的取消程序段，则在 1 和 2 中要插入从 P_1 到 P_0 的额外位移，以确保刀具半径补偿取消后不会在轮廓末端产生运动；即此点定义的不是待补偿轮廓上的位置，而是刀具中心点。
3. 至少编程了一个加工平面内的轴。
   可能缺少的第二条加工平面轴将通过其在上一个程序段中的最终位置模态添加。垂直于加工平面的轴的位置按照 1 或 2 中描述的方式生成（取决于是否编程了该轴）。这样一来所生成的位置便定义了终点 P_0。
   取消刀具半径补偿不需要特别对待，因为所编程的点 P_0 即已定义了整体轮廓末端刀具中心点的位置。
   SAR 轮廓的起点和终点（P_0 或 P_4）既允许在逼近时重合也允许在回退时重合。

前一程序段的速度（典型 G0）

使用这个速度进行所有从点 P_0 到点 P_2 的运动，也就是说运动平行于加工平面，并且进刀运动的部分一直要达到安全距离。

使用 FAD 的进给速度编程

FAD 编程	
G340	进给速度，从 P_2 或 P_3 到 P_4。
G341	垂直于加工平面进刀的进给速度，从 P_2 到 P_3。

如果没有编程 FAD，则此轮廓段以前一程序段编程的、模态有效的速度运行，假如在 SAR 程序段中没有编程用来确定速度的 F 字。

编程特性：

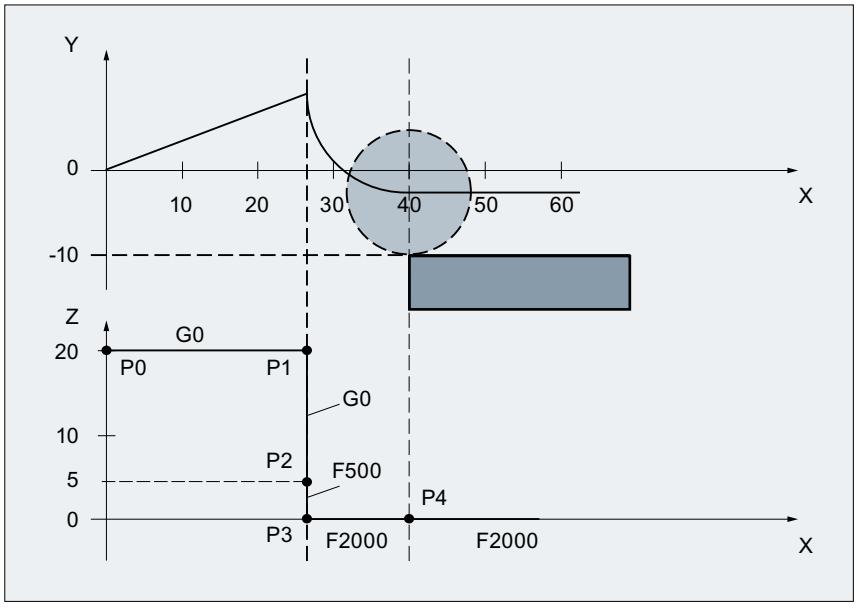
FAD=0 或为负 → 输出报警

FAD=... → 根据生效的 G 功能组 15 的 G 指令，编写的值生效（进给类型；G93、G94 等）

FAD=PM(...) → 与生效的 G 功能组 15 的 G 指令无关，编写的值作为线性进给率（如 G94）

FAD=PR(...) → 与生效的 G 功能组 15 的 G 指令无关，编写的值作为旋转进给率（如 G95）

示例：



程序代码	注释
$TC_DP1[1,1]=120	; 铣刀定义 T1/D1
$TC_DP6[1,1]=7	; 采用 7 毫米半径的刀具

程序代码	注释
N10 G90 G0 X0 Y0 Z20 D1 T1	
N20 G41 G341 G247 DISCL=AC(5) DISR=13FAD 500 X40 Y-10 Z=0 F2000	
N30 X50	
N40 X60	
...	

编程进给率 F

该进给值从点 P_3 起（或者从点 P_2 起，假如 FAD 没有进行编程）生效。如果在 SAR 程序段中没有编程 F 字，则前一程序段中的速度继续生效。通过 FAD 定义的速度不用于后续程序段。

19.5.4.3 速度

逼近时的速度

在下列两副图示中逼近的出发条件为， SAR 的后续程序段中未编程新的速度。 如果不是这种情况，则从点 P_4 起新的速度生效。

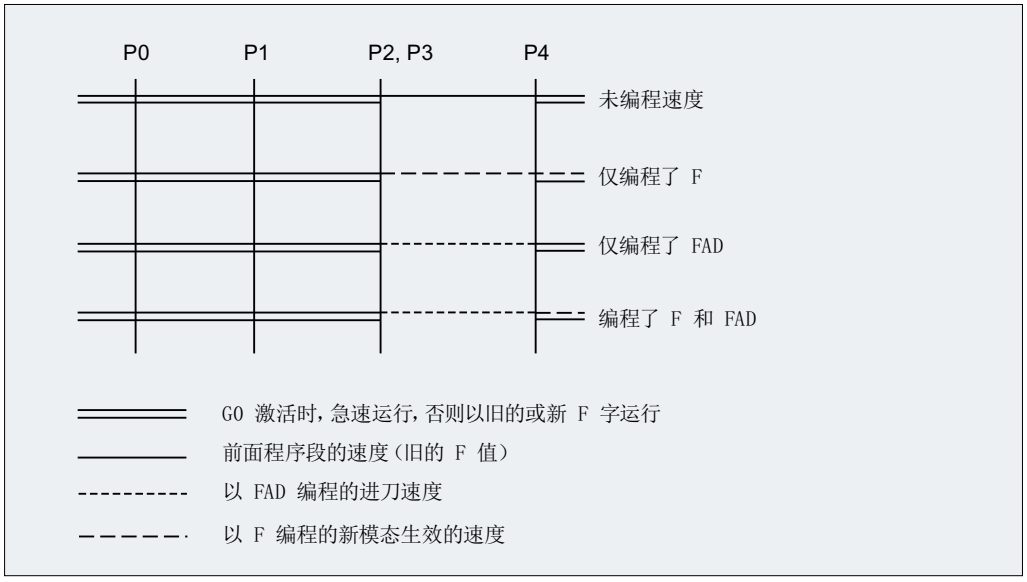


图 19-18 SAR 零件程序段中的速度，在使用 G340 逼近时

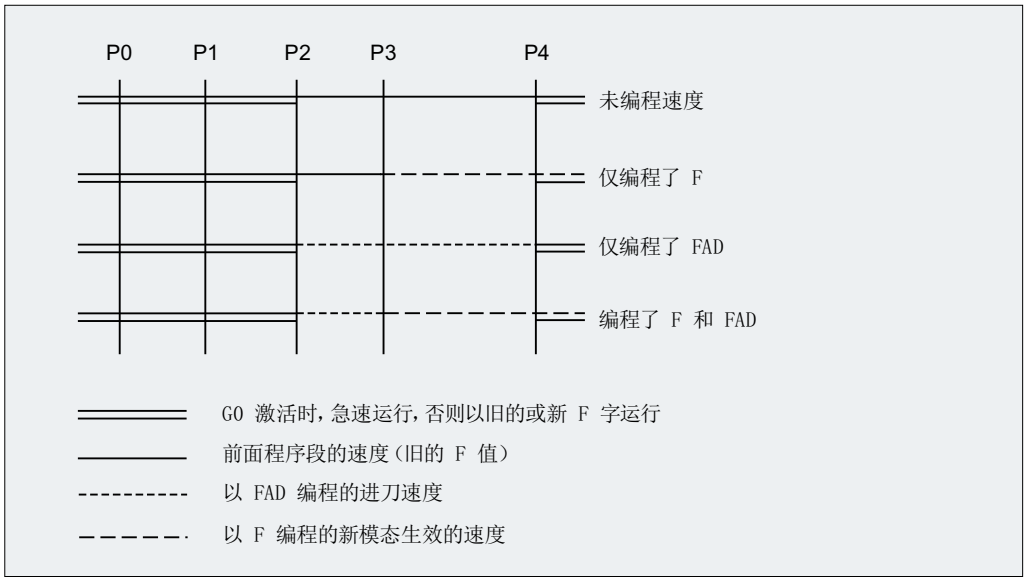


图 19-19　SAR 零件程序段中的速度，在使用 G341 逼近时

回退时的速度

在回退时，前一程序段中模态有效的进给率与在 SAR 程序段中编程的进给值其角色进行调换，也就是说使用旧的进给率来运行本身的回退轮廓（直线、圆弧、螺旋线），而用 F 字新编程的速度则相应的自点 P_2 生效直至点 P_0。

如果平面离开有效并且 FAD 已编程，则 P_3 到 P_2 的距离使用 FAD 来运行，否则使用旧的速度运行。 对于 P_4 到 P_2 的距离始终适用前一程序段中所编程的 F 字。 G0 在这些程序段中无效。

从 P_2 至 P_0 使用 SAR 程序段中所编程的 F 字来运行或者，假如未编程 F 字，则使用前面一个程序段中模态有效的 F 字。 适用的前提条件是 G0 无效。

要在回退时快速运行程序段从 P_2 到 P_0，则必须在 SAR 程序段之前或者在 SAR 程序中自行激活 G0。 此时可能另外会出现的编程 F 字在 SAR 程序段中没有意义。 然而它对后续程序段仍然模态有效。

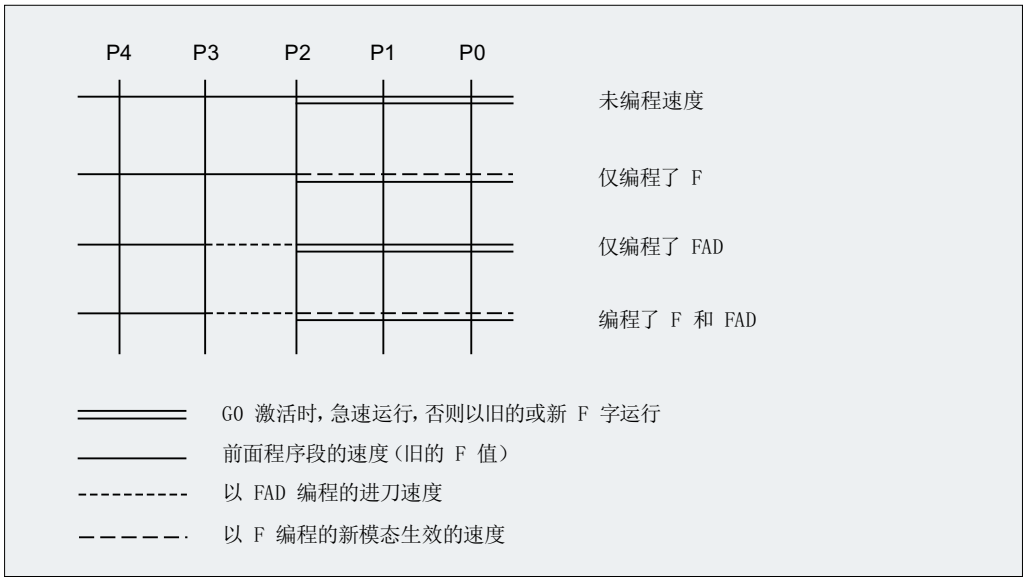


图 19-20　回退时 SAR 零件程序段中的速度

19.5.4.4　系统变量

点 P_3 和 P_4 可以在逼近时作为系统变量在 WCS 中读取：

$P_APR:	在 WCS 中读取 P_3（加载点）	
$P_AEP:	在 WCS 中读取 P_4（轮廓起点）	
$P_APDV	=1	如果 $P_APR 和 $P_AEP 内容有效，即它们还包含有最后所编程 SAR 程序段的相应位置值。
	=0	读取前一个 SAR 程序段的位置值

SAR 程序段和读取操作之间 WCS 的变化不会影响位置值。

19.5.4.5 前提条件

前提条件

- 在一个 SAR 程序段中可以任意编程其他 NC 指令（例如辅助功能输出、同步轴运行、定位轴运行等）。
  在逼近时会在第一个零件程序段中而在回退时会在最后一个零件程序段中执行这些指令。
- 如果终点 P_4 不是取自 SAR 程序段而是来自一个后续的运行程序段，则在该程序段中开始运行实际的 SAR 轮廓（直线、四分圆或半圆）。
  那么原始 SAR 程序段的最后一个零件程序段就不包含用于几何轴的移动信息。 但还是会输出该程序段，因为其中可能有其他必须要执行的动作（例如单轴）。
- 必须始终观察至少两个程序段：
  – SAR 程序段自身
  – 定义了逼近或回退方向的程序段
  在这两个程序段之间可以编程其他程序段。
  可能出现的中间程序段的数量通过以下机床数据限制：
  MD20202 $MC_WAB_MAXNUM_DUMMY_BLOCKS （SAR 时无运行动作的程序段最大数量）
- 如果在逼近程序段中同时激活了刀具半径补偿，则 SAR 轮廓的第一个直线程序段为激活程序段。
  由 SAR 功能生成的整个轮廓在使用刀具半径补偿进行处理时，被当作显编程的方式来进行（防撞监控、交点计算、逼近特性 NORM / KONT）。
- 进刀运动的方向和圆弧面或螺旋线的位置仅仅通过有效的平面（G17 - G19） - 必要时使用有效框架进行旋转 - 来确定。
- 在逼近时 SAR 程序段和定义了切线方向的后续程序段之间不允许插入预处理程序停止。这种情况下的预处理程序停止 - 无论是显编程还是由控制器自动插入 - 会导致报警。

REPOS 时的特性

如果中断一个 SAR 循环并重新定位，则使用 RMIBL 时会在中断点重新回到轮廓。 使用 RMEBL 时的重新起动点为上一个 SAR 程序段的终点，使用 RMBBL 时则为相应的第一个 SAR 程序段的起点。

如果将 RMIBL 与 DISPR （在中断点前以 DISPR 距离重新逼近）一起编程，则可以在中断程序段之前在 SAR 循环的零件程序段中设置重新返回点。

19.5.4.6 示例

示例 1

给定了下列条件：

- 平滑逼近在程序段 N20 中被激活
- X=40 (终点); Y=0; Z=0
- 沿四分圆逼近运行 (G247)
- 未编程逼近方向，G140 生效，也就是说因为刀具半径补偿效（G42）并且轮廓值为正（10），则从右侧逼近轮廓
- 内部生成的逼近圆弧（SAR 轮廓）半径为 20，使得刀具中心点轨迹的半径等于编程值 DISR=10
- 由于使用了 G341 所以在平面内的一个圆中逼近运行，这样可以得出起点（20, -20, 0）
- 由于 DISCL=5 点 P2 位置为（20, -20, 5），点 P1 由于 N10 中的 Z30 而位于（20, -20, 30）

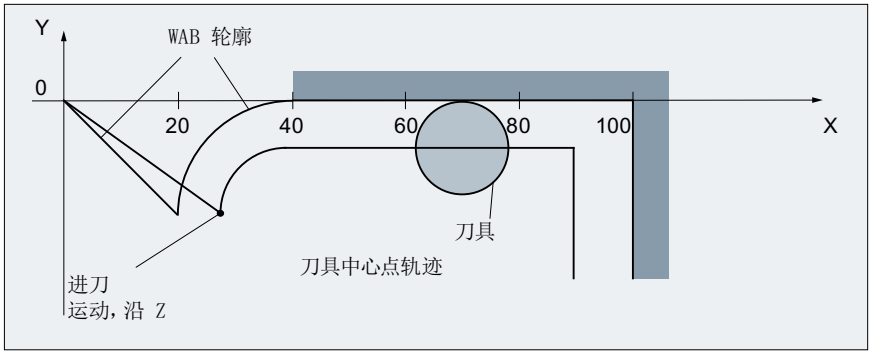


图 19-21 轮廓示例 1

零件程序：

程序代码	注释
$TC_DP1[1,1]=120	；刀具定义 T1/D1
$TC_DP6[1,1]=10	；半径
N10 G0 X0 Y0 Z30	
N20 G247 G341 G42 NORM D1 T1 Z0 FAD=1000 F=2000 DISCL=5 DISR=10	
N30 X40	
N40 X100	
N50 Y-30	
...	

示例 2

逼近时给定了下列条件：

- 平滑逼近在程序段 N20 中被激活
- 沿四分圆逼近运行 (G247)
- 未编程逼近方向，G140 生效，也就是说因为刀具半径补偿有效（G41），从左侧逼近轮廓
- 轮廓偏移 OFFN=5 (N10)
- 当前的刀具半径 = 10，因此有效的 TRC 补偿半径 =15，SAR 轮廓的半径等于 25，这样刀具中心点轨迹的半径为 DISR=10
- 圆弧的终点由 N30 得出，因为在 N20 中只编程 Z 位置
- 进刀运动
  - 从 Z20 向 Z7 (DISCL=AC(7)) 快速运行
  - 然后运行到 Z0， 使用 FAD=200
  - X-Y 平面内的逼近圆弧和使用 F1500 的后续程序段
    （为了使该速度在后续程序段中生效，必须将有效的 G 代码 G0 在 N30 中用 G1 进行覆盖。否则会使用 G0 继续进行加工轮廓。）

回退时给定了下列条件：

- 平滑回退在程序段 N60 被激活
- 沿四分圆（G248）和螺旋线（G340）进行回退
- FAD 未编程，因为在 G340 时没有意义
- Z=2 在起点；Z=8 在终点，因为 DISCL=6

- 当 DISR=5 时，SAR 轮廓的半径=20，刀具中心点轨迹的半径=5
- 圆弧程序段之后会从 Z8 向 Z20 离开，平行于 X-Y 平面运行直到终点 X70 Y0

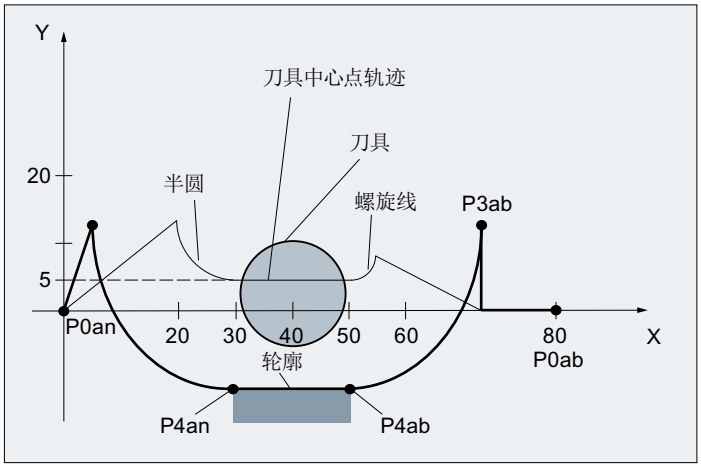


图 19-22　轮廓示例 2

零件程序：

程序代码	注释
$TC_DP1[1,1]=120	; 刀具定义 T1/D1
$TC_DP6[1,1]=10	; 半径
N10 G0 X0 Y0 Z20 G64 D1 T1 OFFN = 5	; (P0an)
N20 G41 G247 G341 Z0 DISCL = AC(7) DISR = 10 F1500 FAD=200	; (P3an)
N30 G1 X30 Y-10	; (P4an)
N40 X40 Z2	
N50 X50	; (P4ab)
N60 G248 G340 X70 Y0 Z20 DISCL = 6 DISR = 5 G40 F10000	; (P3ab)
N70 X80 Y0	; (P0ab)
N80 M 30	

说明

这种方式生成的轮廓要通过刀具半径补偿进行修正，在 SAR 逼近程序段中激活补偿并在 SAR 回退程序段中取消。

刀具半径补偿所考虑的实际有效半径为 15，这是由刀具半径（10）和轮廓偏移值（5）相加得到的。因此所得到的刀具中心点轨迹半径在逼近时为 10，在回退时为 5。

19.5.5 取消刀具半径补偿 (G40)

指令 G40

取消刀具半径补偿使用指令 G40。

特殊性

- 只能在一个程序段中使用 G0 （快速运行）或 G1（线性插补）来取消刀具半径补偿。
- 如果在刀具半径补偿 D0 有效时进行编程，则不会出现取消选择，而会出现故障信息 10750。
- 如果在取消了刀具半径补偿的程序段中对几何轴进行编程，则刀具半径补偿被取消，即使其并不在当前平面内。

19.5.6 外角的补偿

G450/G451

使用 G 指令 G450/G451 可以在外角处进行不连续程序段过渡时来控制其特性：

指令	含义
G450	使用过渡圆弧的不连续程序段过渡
G451	使用等距线交点的不连续程序段过渡

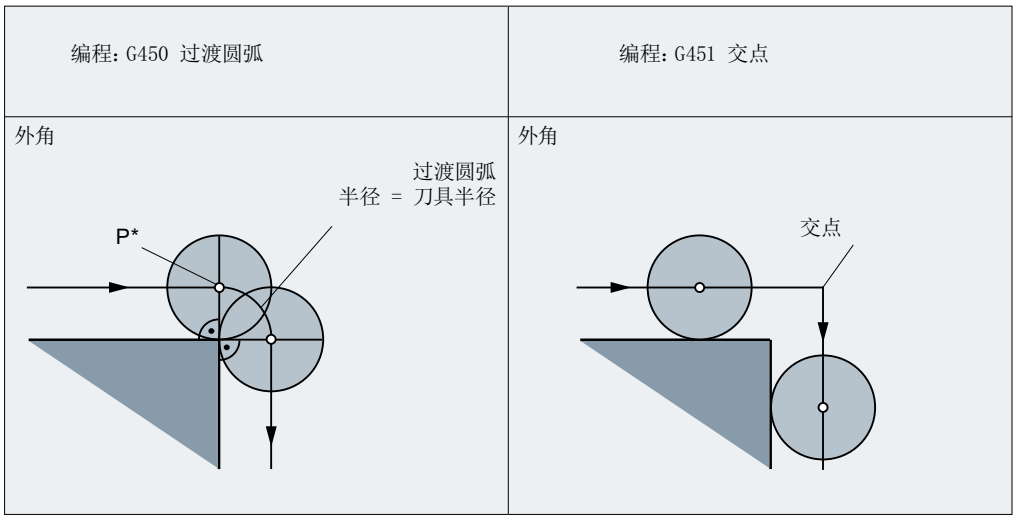


图 19-23　示例，90° 外角使用 G450 和 G451

G450 （过渡圆弧）

当 G 指令 G450 有效时，刀具中心点会在外角处使用刀具半径运行一个圆弧轨迹。该圆弧轨迹从前一轨迹块终点的正常位置（垂直于轨迹切线）开始并结束于新的轨迹块（程序段）起始点的正常位置.

当外角非常扁平时，会使用 G450 （过渡圆弧）和 G451 （交点）不断靠近（参见章节“非常扁平的外角”）。

如果要生成尖的外轮廓角，必须从轮廓提刀（参见章节“DISC”）。

DISC

使用 G450 - 过渡圆弧不能生成锋利的外轮廓角，因为通过过渡圆弧运行刀具中心点轨迹时会将刀具刀沿停留在外角上（编程位置）。如果在使用 G450 时仍然需要加工出锋利的外角，则可以使用指令 DISC 在程序中进行提刀编程。借此可以通过过渡圆弧形成一个锥形截面并且刀具刀沿会从外角退开。

指令 DISC 的取值范围在步骤 1 中为 0 到 100。

值	含义
DISC = 0	关闭提刀，过渡圆弧有效
DISC = 100	提刀的大小应使得其特性理论上与交点（G451）时相同。

用于 DISC 的编程最大值可以通过机床数据进行设置：

MD20220 $MC_CUTCOM_MAX_DISC（DISC 的最大值）

对于 DISC 有意义的值通常不超过 50。

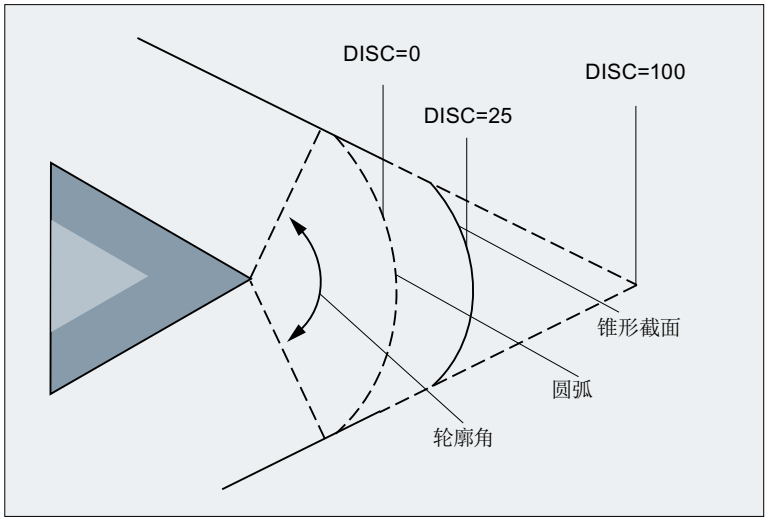


图 19-24　示例：提刀，使用 DISC=25

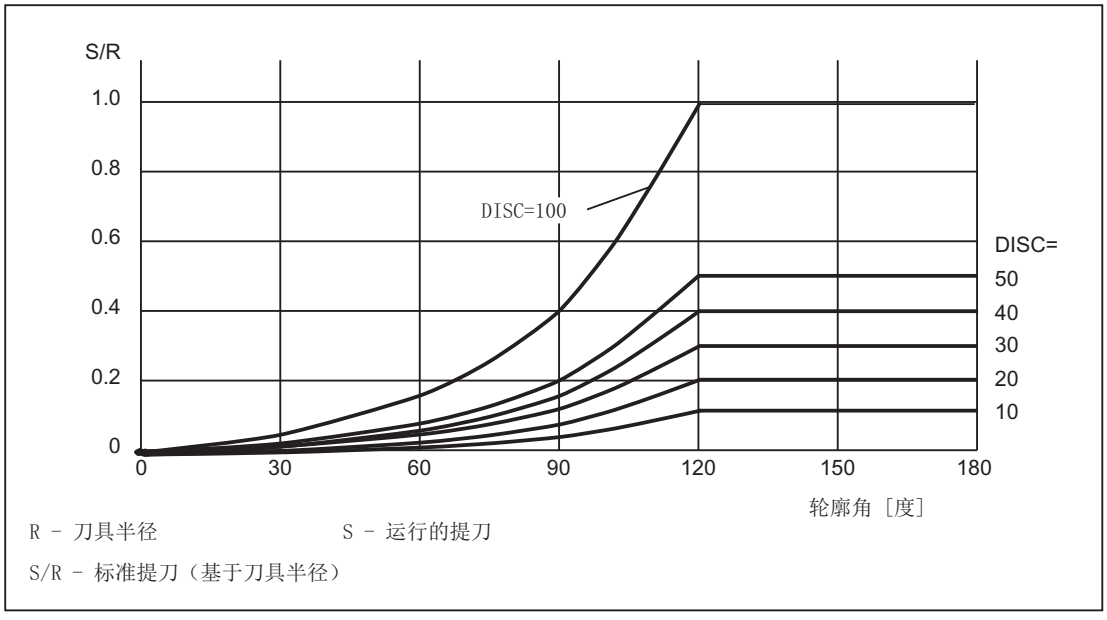


图 19-25　提刀，使用 DISC 与轮廓角有关

G451（交点）

在 G 指令 G451 有效时逼近由轨迹线（仅在直线、圆弧或螺旋线时）生成的位置（交点），该位置与编程轮廓间的距离为刀具半径（刀具的中心点轨迹）。样条与多项式原则上不得延长。

非常尖锐的外角

在非常尖锐的外角处使用 G451 可能会导致长时间的空行程。因此在非常尖锐的外角处会自动从 G451 （交点）切换到 G450 （过渡圆弧，必要时使用 DISC）。

用于自动切换（交点 → 过渡圆弧）的极限角（轮廓角）可以在机床数据中预设：

MD20210 $MC_CUTCOM_CORNER_LIMIT（刀具半径补偿时补偿程序段的最大角度）

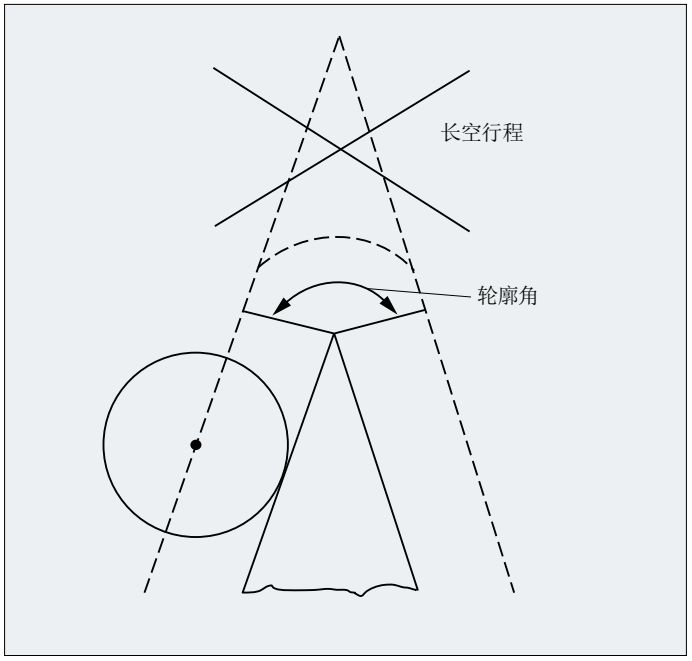


图 19-26　用于自动切换至过渡圆弧的示例

非常扁平的外角

当外角非常扁平时，会使用 G450 （过渡圆弧）和 G451 （交点）不断靠近。在这种情况下插入一个过渡圆弧不再有意义。尤其是在 5 轴加工时不允许在这类外角处插入过渡圆弧，因为这反而会在连续轨迹控制运行（G64） 损失速度。因此在非常扁平的外角处会自动从 G450 （过渡圆弧，必要时使用 DISC）切换到 G451 （交点）。

用于自动切换（过渡圆弧 → 交点）的极限角（轮廓角）可以在机床数据中预设：

MD20230 $MC_CUTCOM_CURVE_INSERT_LIMIT（刀具半径补偿时用于交点计算的最大角度）

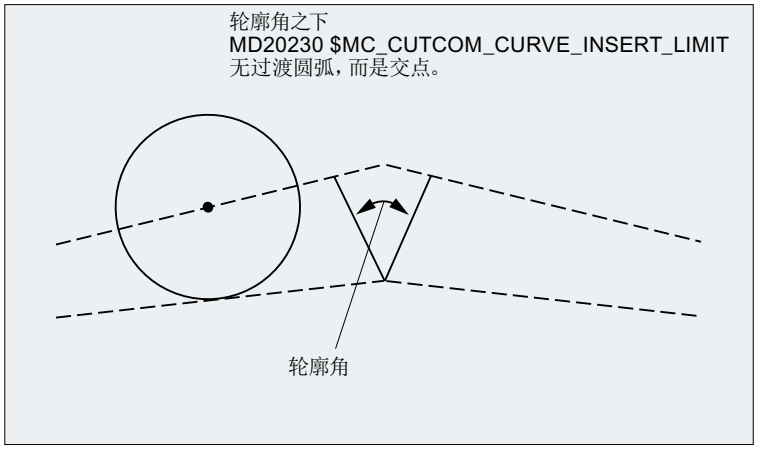


图 19-27　用于自动切换至交点的示例

19.5.7 内角的补偿

交点

如果两个依次连续的程序段构成了一个内角，则可以尝试找到两条等距线的交点。 如果找到了交点，则编程轮廓可以缩减至交点（第一个程序段在结束处缩减，第二个程序段在开始处缩减）。

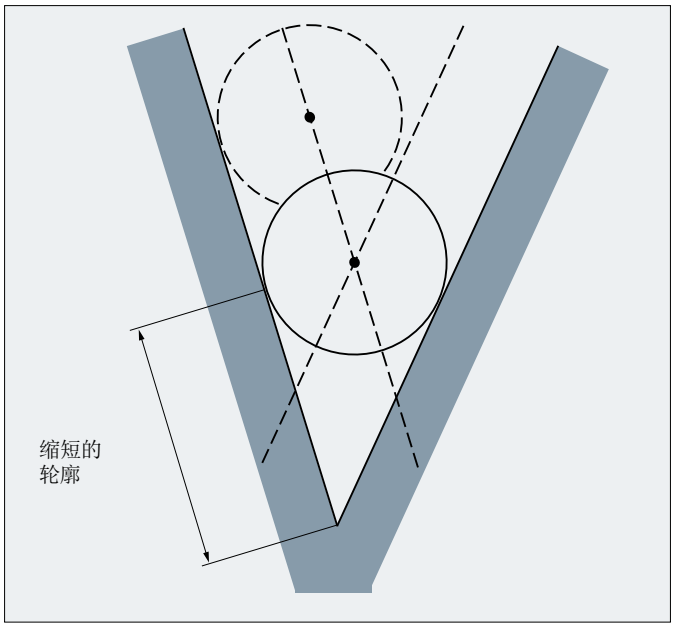


图 19-28 用于缩减轮廓的示例

无交点

对于内角有可能出现这种情况，在两个依次连续的程序段之间找不到交点（参见下图）。

前瞻性轮廓计算

如果在两个依次连续的程序段之间找不到交点，则控制器会自动观察下一个程序段并尝试利用该程序段的等距线去寻找交点（参见下图：交点 S）。 观察后续程序段（前瞻性轮廓计算）会一直进行，直至达到通过机床数据所设定的程序段数量为止：

MD20240 $MC_CUTCOM_MAXNUM_CHECK_BLOCKS （刀具半径补偿时用于前瞻性轮廓计算的程序段）

如果在预先观察的程序段中找不到交点，则程序处理停止并发出报警。

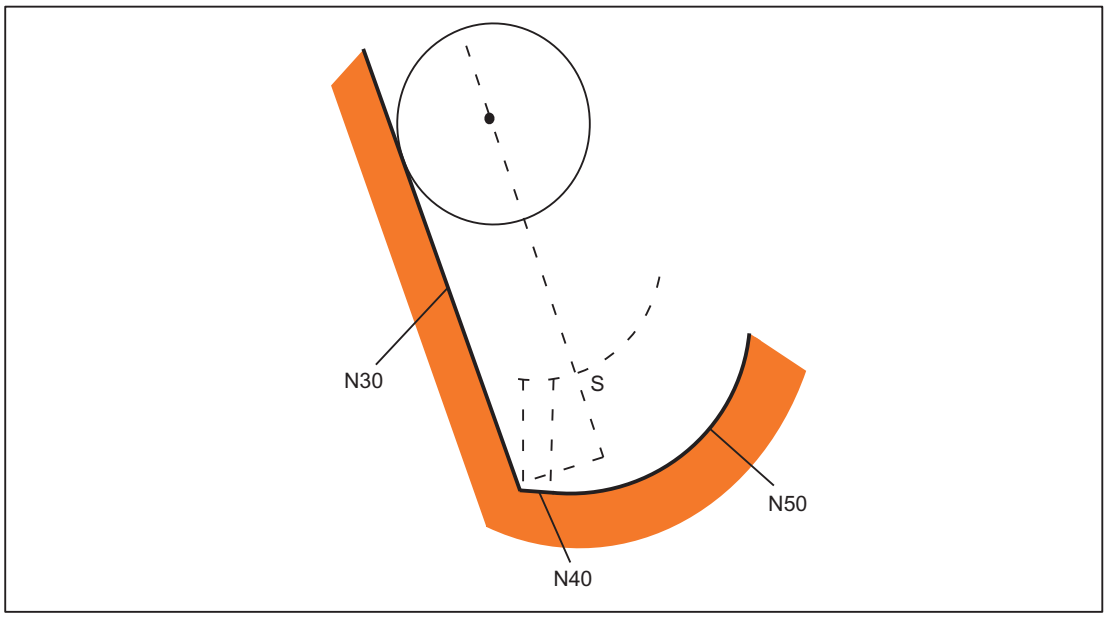


图 19-29　如果在程序段 N30 和程序段 N40 之间没有交点，则计算程序段 N30 和程序段 N50 之间的交点。

多个交点

对于内角有可能出现这种情况，前瞻性轮廓计算在多个连续的程序段中找到了多个等距线交点。 此时总是将最后一个交点确定为有效交点：

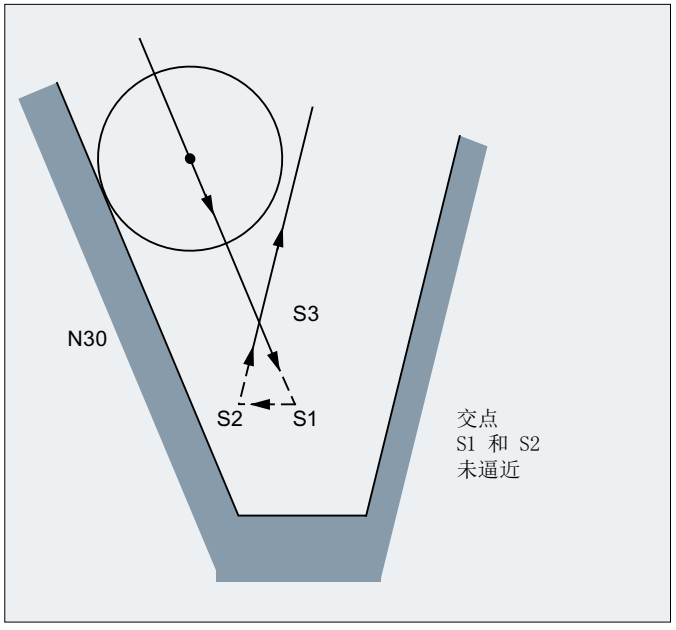


图 19-30　示例： 内角使用刀具半径补偿不破坏轮廓 （预先观察 3 个程序段）

更多信息另参见“防撞监控和瓶颈识别 (页 1663)” 章节 “防撞监控”。

特殊性

如果使用下一程序段找到了多个交点，则距离第一个程序段起始点最近的那个交点有效。

19.5.8 防撞监控和瓶颈识别

功能

防撞监控（瓶颈识别）用来检查等距线是否会与不相邻的程序段相交。 如果找到这样的交点，则处理方式与内角出现多个交点相同： 最后找到的那个交点有效。

预先观察的程序段的最大数量可以通过以下指令设置：

MD20240 $MC_CUTCOM_MAXNUM_CHECK_BLOCKS （刀具半径补偿时用于前瞻性轮廓计算的程序段）

句法

```
CDON
CDOF
CDOF2
```

含义

CDON：	启用碰撞监控
CDOF：	关闭碰撞监控
CDOF2：	关闭碰撞监控，3D 圆周铣削时。 对于所有其它的加工类型（例如 3D 端面铣削），CDOF2 与 CDOF 的含义相同。

其他信息

在 CDOF 时在两个依次连续的程序段之间寻找交点。 如果找到了交点，就不再观察其他程序段。 在外角时总是能在两个依次连续的程序段之间找到一个交点。

因此在 CDON 时以及在 CDOF 时也能够预先观察两个以上相邻的程序段。

忽略程序段

如果在两个不相邻的程序段之间识别出交点，则不会执行这两者之间在轮廓平面内的运行动作。 被忽略程序段中所包含的所有其他动作和可执行指令（M 指令，定位轴运行等）会在交点的位置按照它们在 NC 程序中的顺序来执行。

警告 10763：“补偿平面中的程序段轨迹分量变为零。”

如果由于防撞监控或瓶颈识别忽略了一个程序段，则显示警告 10763。 NC 程序不会中断。

警告显示可通过以下方式抑制：

MD11410 $MN_SUPPRES_ALARM_MASK，位 1 = 1（特殊报警的支持位）

瓶颈识别与外角

在对不相邻程序段进行交点检查时不检查已编程的原始轮廓，而是检查与之相应的、计算出的等距线。 这会导致在外角处识别出“瓶颈”，虽然实际情况并非如此。 其原因是，在 DISC > 0 时所计算出的刀具轨迹与所编程的原始轮廓并非等距运行。

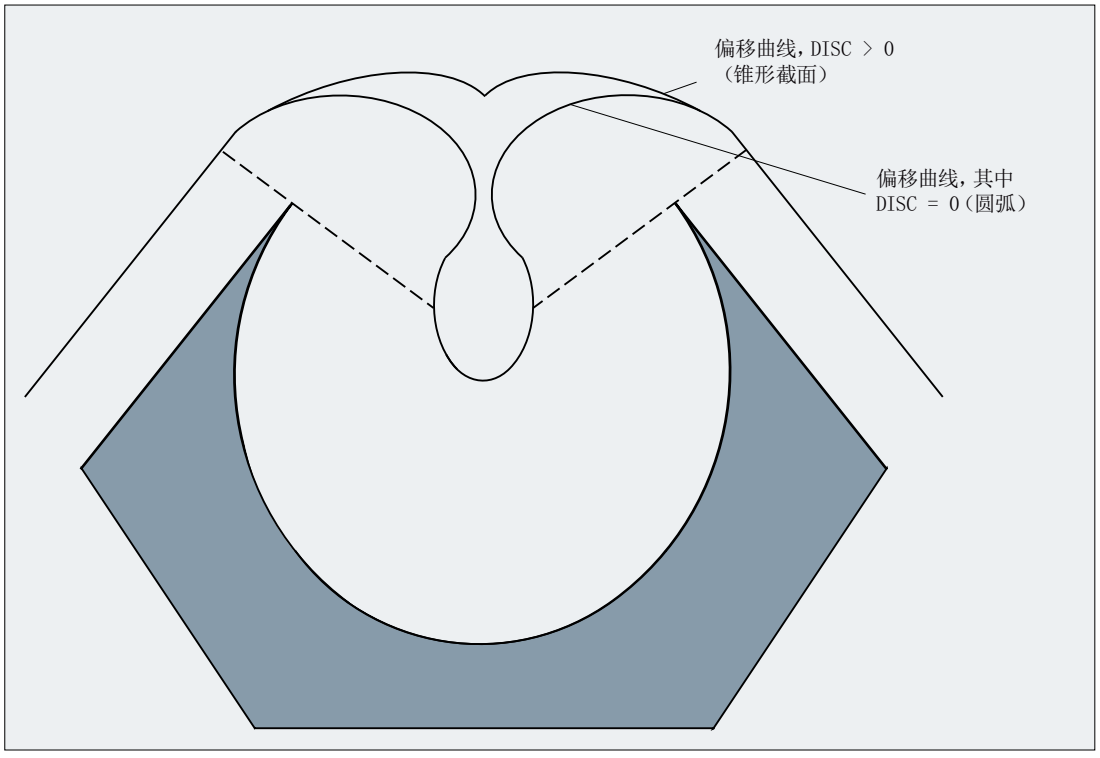


图 19-31　瓶颈识别与外角

19.5.9 槽形识别（选件）- 仅 840D sl

刀具半径补偿激活且碰撞监控 (页 1663)激活时，系统可能检测出已编程的槽宽小于刀具直径（可能导致轮廓破坏），从而在加工时忽略。在特定的工艺应用（例如：使用激光切割机创建弯曲线时）中还要求在刀具半径（激光束）稍微大于编程的槽宽时也能允许进行槽的加工。为了确保用户能进行这些操作，刀具半径补偿扩展了“槽形识别”功能。

说明

“槽形识别”功能是针对 SINUMERIK 840D sl 的一个授权选件。

产品编号：6FC5800-0AS18-0YB0

功能

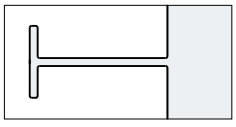
“槽形识别”功能激活时，系统会自动识别预定形状的槽并确保其加工。

识别到：

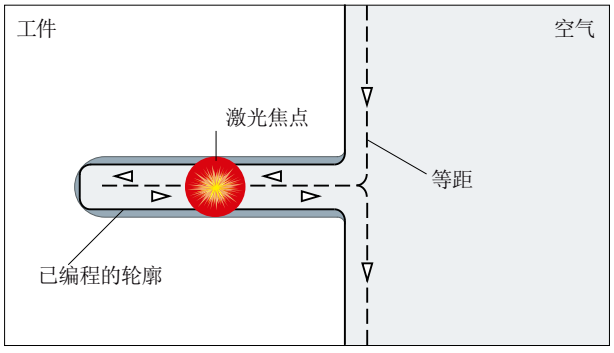
- I 形槽



- T 形槽



如果槽宽大于或等于刀具直径，则槽按照编程的开始运动。如果槽宽小于刀具直径，则系统会计算等距，使得刀具能在编程的槽的中心位置运行：



必须满足下列条件：

- 新刀具的半径（R2）不允许超出原来刀具半径（R1）的 2 倍：
  R2 < 2 * R1
- 槽必须含有奇数个程序段。
- 中间程序段必须由偶数个组成。
- 槽的起始端圆整半径或位于槽前面的程序段的长度不允许低于 R2 - R1 的差值。

激活

激活“槽形识别”功能时必须将以下通道专用的设定数据的值设为“1”：

SD42977 $SC_SLOT_FORM_RECOGN = 1

参数设置

预读标志识别的程序段数量

只有以下通道专用的机床数据设为大于零的值时，槽形识别才生效：

MD28620 $MC_MM_NUM_FEATURE_BLOCKS > 0

通过 MD28620 可以确定在槽形标识时可同时关注的最小程序段数量。

推荐值：15（该值与 T 形槽（两条直线之间有圆弧）上的轮廓元素数量相符。）

示例

以下零件程序截面上编程了一条可通过激光束切割的弯曲线。

程序代码	注释
N1 G01 G90 G71 F10000	
N2 $TC_DP1[1,1] = 120	
N3 $TC_DP3[1,1] = 0.0	
N4 $TC_DP6[1,1] = **0.2**	; 刀具半径 = 0.2 mm
N6 G91 **G41** G01 T=1 D=1	; TRC 打开。
N7 X100.00	
N8 Y-24.500	
N9 G02 X-0.200 Y-0.200 I-0.200	; 弯曲线轮廓起始端上的圆整。
N10 G01 X-49.700	
N11 G03 X-0.100 Y-0.100 J-0.100	
N12 G01 Y-0.100	
N13 G03 X0.100 Y-0.100 I0.100	
N14 G01 X49.700	

程序代码	注释
N15 G02 X0.200 Y-0.200 J-0.200	；弯曲线轮廓末端上的圆整。
N16 G01 Y-24.500	



槽形识别未激活时（SD42977 $SC_SLOT_FORM_RECOGN = 0），刀具半径补偿会将已编程的弯曲线识别为所用刀具（激光束）不允许的轮廓并忽略，因为槽宽为 0.3 mm，而刀具半径只有 0.2 mm。相反，如果槽形识别激活（SD42977 $SC_SLOT_FORM_RECOGN = 1），刀具半径补偿会将已编程的弯曲线识别为 I 形槽并使用激光束沿着已编程槽的中心切割弯曲线。

由于此示例中的槽由七个轮廓元素组成（4 个圆弧，3 条直线），必须在 MD28620 $MC_MM_NUM_FEATURE_BLOCK 设置以下最小值：

MD28620 $MC_MM_NUM_FEATURE_BLOCK ≥ 7

19.5.10 使用可变补偿值的程序段

前提条件

在所有的插补类型中都允许使用可变补偿值（也包括圆弧插补和样条插补）。

同样也允许变换符号（此时要变换补偿侧）。

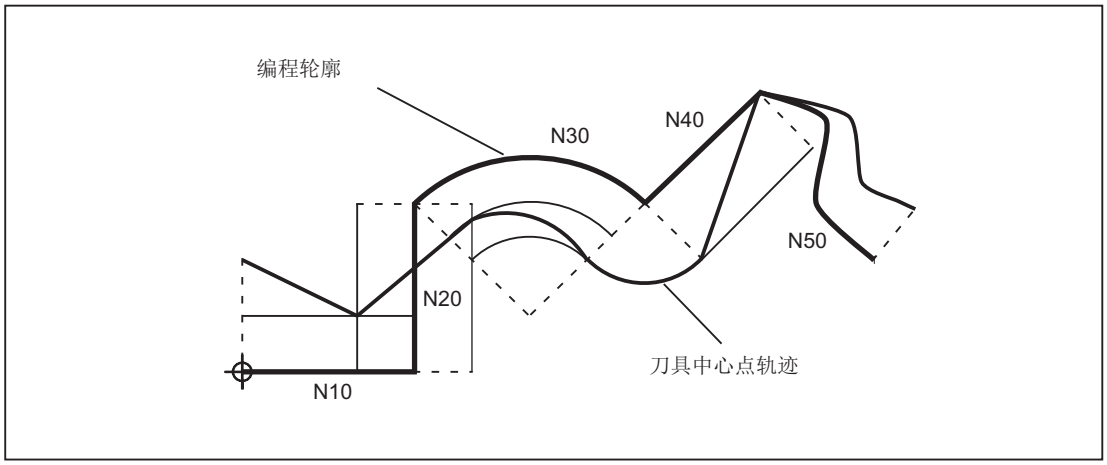


图 19-32　补偿值可变时的刀具半径补偿

交点计算

在使用可变补偿值的程序段中进行交点计算时，偏移曲线（刀具轨迹）始终参照一个恒定的补偿值进行计算。

如果使用可变补偿值的程序段是预先所观察两个程序段在运行方向上的第一个，则在程序段结束时计算补偿值，否则在程序段开始时计算补偿值。

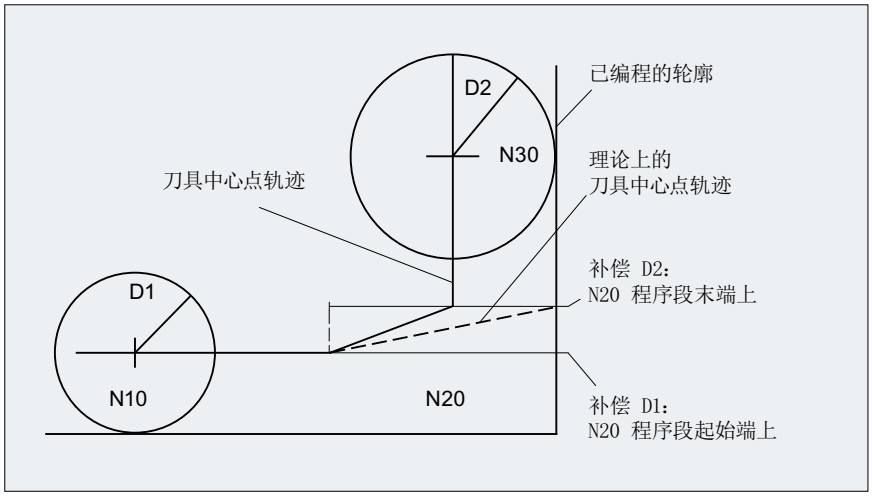


图 19-33　补偿值可变时的交点计算

功能的局限性

如果在圆内侧进行加工时补偿半径大于所编程的圆半径，则会使用下列报警拒绝进行加工：

报警 10758 “使用可变补偿值的曲率半径过小”

保持封闭轮廓稳定

如果两个圆的半径有些微增加，则其可以要求第三个程序段，用来保持封闭轮廓的稳定。然后会出现这种情况，如果两个相邻的程序段在一个封闭的轮廓内生成了两个可能的交点，会基于补偿将其跳过。

不选择第二个可能的交点而选择第一个交点，可以形成一个稳定的封闭轮廓：

SD42496 $SC_CUTCOM_CLSD_CONT ≠ 0 （封闭轮廓时刀具半径补偿的特性）

因此在跳过第一个程序段的情况下总是会到达第二个可能的交点。 此时第三个程序段就多余了。

19.5.11 报警特性

预处理时报警

如果在预处理时刀具半径补偿出现报警，则主运行中的加工会在下一个能到达的程序段结尾处暂停，即通常会在当前插补程序段的结尾处(在 Look-Ahead 预读功能有效时，需要等轴都变为停止状态）。

预处理程序停止时报警和有效的刀具半径补偿

为了确定一个程序段的终点，刀具半径补偿通常需要至少一个后续运行程序段（在瓶颈处需要更多）。 因为在有效预处理程序停止时没有这样的程序段可供使用，所以会运行至上一程序段的偏移点。 相应的，在预处理程序停止之后的第一个程序段中会返回起始点的偏移点。

这时所生成的轮廓可能会与没有预处理程序停止时的轮廓有明显的偏差。 特别可能破坏轮廓。 因此要输入下列设定数据：

SD42480 $MC_STOP_CUTCOM_STOPRE （刀具半径补偿和预处理程序停止的报警反应）

依靠该值刀具半径补偿的特性相对于之前的状态保持不变，或者在刀具半径补偿有效情况下预处理程序停止时输出报警并暂停程序处理。

操作员可以应答该报警并使用 NC 启动不作改变继续执行 NC 程序或者使用 复位 进行中断。

19.5.12 用于多项式的交点程序

功能

刀具半径补偿有效时两根曲线会形成一个外角，这与 18 组的 G 代码有关（刀具补偿时角的特性； G450 / G451）而与参与曲线的类型（直线、圆弧、多项式）无关：

- 插入用于角绕行的锥形截面
  或者
- 参与曲线外插时要使得它们会形成一个交点

如果在 G451 有效时找不到交点或者两条参与曲线所形成的角度过尖，则会自动切换至插入程序。

可以使用机床数据使能用于多项式的交点程序：

MD20256 $MC_CUTCOM_INTERS_POLY_ENABLE （能够使用用于多项式的交点程序）

说明

如果该机床数据被设置为无效，就会始终插入一个（也许非常短的）程序段（即使过渡非常接近于切线）。这些短的程序段在 G64 运行中必然会导致不希望出现的速度骤降。

19.5.13 G461/G462: 逼近/回退方案扩展

功能

在一些特殊的几何情况下，相对于现有的实现方法，还需要扩展的逼近/回退措施（激活或者不激活刀具半径补偿时）（参见下面的图形）。

说明

下面所描述的都是刀具半径补偿取消时的情形。逼近时的特性与此完全对称。

示例

```
G42 D1 T1                              ；刀具半径 20 毫米
...
G1 X110 Y0
N10 X0
N20 Y10
N30 G40 X50 Y50
```

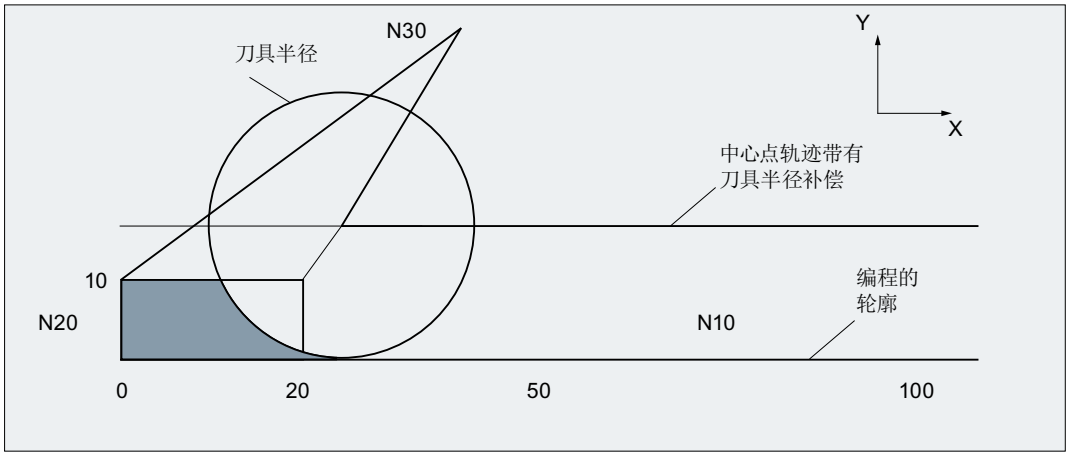


图 19-34　回退特性，在 G460 时

最后的程序段带有效的刀具半径补偿（N20），由于此程序段很短，从而在当前的刀具半径情况下，其补偿线不再与前一程序段（或者一个很前面的程序段）形成交点。因此，在后续程序段的补偿线和前一程序段之间寻找一个交点，在示例中就是在 N10 和 N30 之间。在此，出发程序段所用到的补偿线并不是真正的补偿线，而是一条从位于程序段 N20 终点的补偿点到 N30 的编程终点之间的直线。如果找到一个交点，则必须返回此点。图中彩色显示的区域不加工，尽管使用该刀具可以进行加工。

G460

在 G460 时逼近或回退特性与先前一样。

G461

如果最后的 TRC 程序段与前一程序段不可能有一个交点，则该程序段的补偿线用一个圆弧延长，其圆心位于未补偿程序段的终点，半径与刀具半径相同。

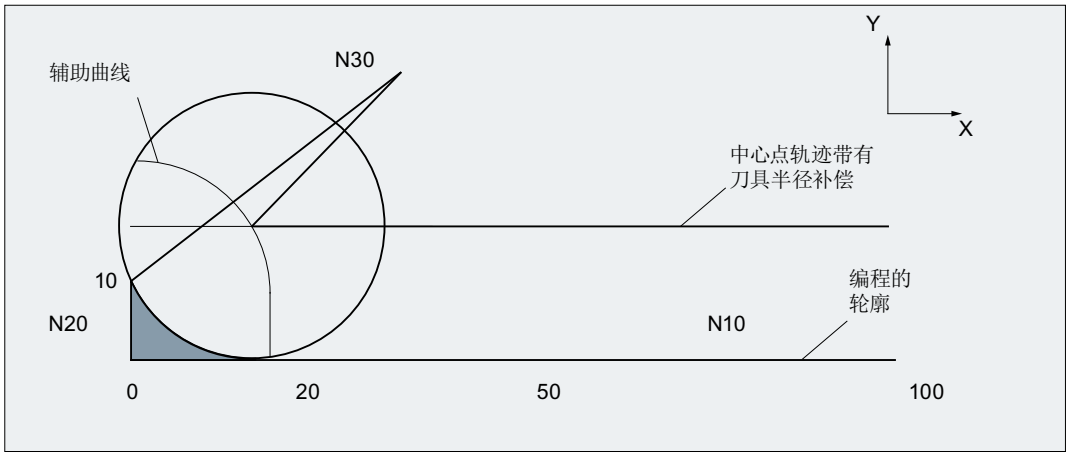


图 19-35　在 G461 时的回退特性

控制系统尝试用前面的一个程序段切削该圆弧。如果事先找到一个交点，则在有效的 CDOF CDOF 存在时停止这种寻找，这就是说对是否在很前面的程序段中还存在一个交点不再进行检测。

在 CDON 有效时，如果已经找到一个交点，则也会在后面继续寻找其它的交点。

这样找到的交点是以前程序段的新终点和取消程序段的起始点。所插入的圆弧仅用于计算交点，自身并不会引起运行。

说明

如果没找到交点，则输出下列报警：
报警“10751 碰撞危险”

G462

如果最后的 TRC 程序段与一个前面的程序段不可能产生交点，则在使用 G462 （初始设定）逼近时，在带刀具半径补偿的最后程序段的终点处插入一条直线（该程序段通过其终点切线延长）：

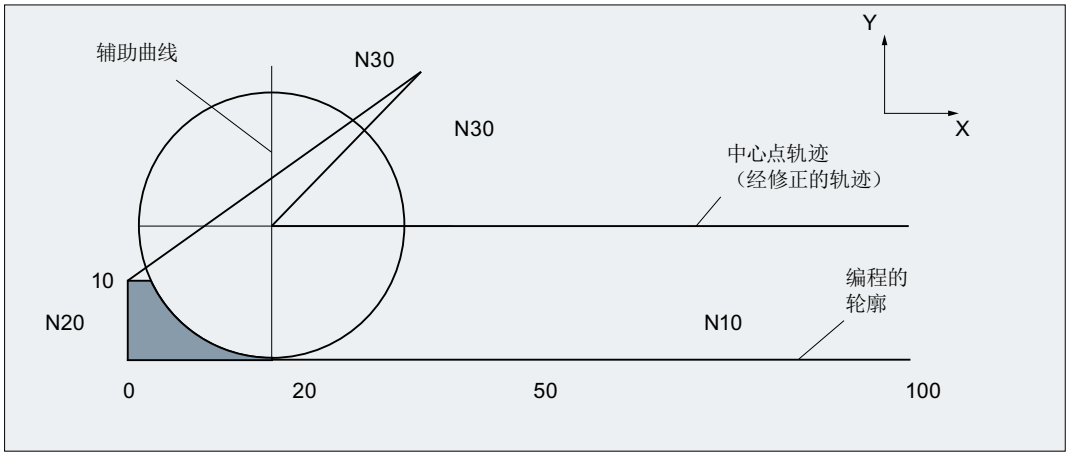


图 19-36　在 G460 时的回退特性

交点的寻找过程与在 G461 时一样。

使用 G462 时，N10 和 N20 示例程序中所形成的拐角没有完全加工到其刀具允许加工的范围。但是这种特性可能是必要的，如果部分轮廓（偏离编程的轮廓），在此示例中在 N20 左侧，即使 y 值大于 10 毫米也不允许受到损伤。

如果 KONT 有效（轮廓在起始点或者终点绕行），则其特性不一样，取决于终点是在轮廓之前或者之后。

终点在轮廓之前

如果终点位于轮廓前侧，则回退特性与 NORM 时相同。该属性不发生变化，即使在 G451 时使用直线或圆弧对最后一个轮廓程序段进行延长。因此，无需为了防止轮廓终点附近出现碰撞而采用附加的绕行方案。

终点在轮廓之后

如果终点在轮廓后侧，则根据 G450 / G451 插入圆弧或直线。G460 - G462 此时没有作用。

在这种情况下如果最后运行的程序段与前面的一个程序段没有交点，则可能会与所插入的轮廓单元产生一个交点，或者与绕行圆弧的终点到编程的终点的直线段产生一个交点。

如果所插入的轮廓单元是一个圆弧 （G450），它与前一个程序段构成一个交点，则这个交点与 NORM 和 G461 时生成的交点相同。通常情况下，还必须要运行一个附加的圆弧段。对于回退程序段的线性部分不再要求交点的计算。

在第二种情况下（如果所插入的轮廓单元与前面的程序段没有交点）运行到回退直线与前一个程序段之间的交点。

因此在 G461 或 G462 有效时只会出现相对于 G460 时不同的特性，不管是在 NORM 有效时还是在 KONT 特性与 NORM 特性的几何条件相同时。

说明

逼近运行性能与回退运行性能对称。

逼近或回退的特性由逼近程序段或回退程序段的状态确定。因此逼近特性可以单独设定，而不受回退特性的影响。

示例：

逼近时运用 G461 的程序：

```
N10 $TC_DP1[1,1]=120                    ; 刀具类型铣刀
N20 $TC_DP6[1,1]=10                     ; 半径
N30 X0 Y0 F10000 T1 D1
N40 Y20
N50 G42 X50 Y5 G461
N60 Y0 F600
N70 X30
N80 X20 Y-5
N90 X0 Y0 G40
N100 M30
```

19.6 刀具半径补偿保持恒定

含义

“保持恒定刀具半径补偿”功能用来抑制一定数量程序段的刀具半径补偿，但同时也会将先前程序段中由刀具半径补偿构成的差数，即刀具中心点已编程轨迹和实际运动轨迹之差作为偏移保留。

例如：在逐行铣削时，返向点中需要多个运动程序段、但这些运动程序段不为刀具半径补偿生成的轮廓（绕行方案）所需，此时，该功能可以发挥极大的作用。

激活

“刀具半径补偿保持恒定”功能可以使用 G 指令 CUTCONON (CUTter compensation CONstant ON) 激活并使用 G 指令 CUTCONOF (CUTter compensation CONstant OFF) 取消激活。

CUTCONON 和 CUTCONOF 构成了一个模态 G 代码组。

初始设定值为 CUTCONOF。

该功能可独立于刀具半径补偿类型 （$2^1/_2$D，3D-端面铣削, 3D-圆周铣削) 进行使用。

通常情况

在通常情况下在激活补偿抑制之前，刀具半径补偿已经是有效的，并且如果取消补偿抑制的话依旧有效。

在 CUTCONON 之前的上一个运动程序段中向程序段终点中的偏移点运动。所有后续的，并且在其中补偿抑制当前有效的程序段可以在没有补偿的情况下运行。然而它们在运行时会从最后补偿程序段的终点偏移一定的矢量到它的偏移点。这些程序段的插补类型（线性、圆周形、多项式）为任意类型。

取消补偿抑制的程序段（即含有 CUTCONOF 的程序段）被正常修改。从起始点的偏移点处开始。在上一个程序段的终点（即带有激活的 CUTCONON 的上一个已编程的运动程序段的终点）和该点之间插入一个线性程序段。

那些圆平面垂直于补偿平面的圆形程序段（垂直的圆）被处理成就像是在其中已经编程了 CUTCONON 的形式。补偿抑制的隐含的激活在第一运行程序段中自动清除，这个运行程序段在补偿平面中包含一个运行程序段并且不是这样的圆弧。在这个意义上垂直的圆弧只可能在圆周铣削时出现。

示例：

N10	；定义刀具 d1
N20 $TC_DP1[1,1]= 110	；类型
N30 $TC_DP6[1,1]=	；半径
N40	
N50 X0 Y0 Z0 G1 G17 T1 D1 F10000	
N60	

N70 X20 G42 NORM	
N80 X30	
N90 Y20	
N100 X10 CUTCONON	；启用补偿抑制
N110 Y30 KONT	；在关闭补偿抑制时 必要时插入绕行圆弧
N120 X-10 CUTCONOF	
N130 Y20 NORM	；关闭刀具半径补偿时没有绕行圆弧
N140 X0 Y0 G40	
N150 M30	

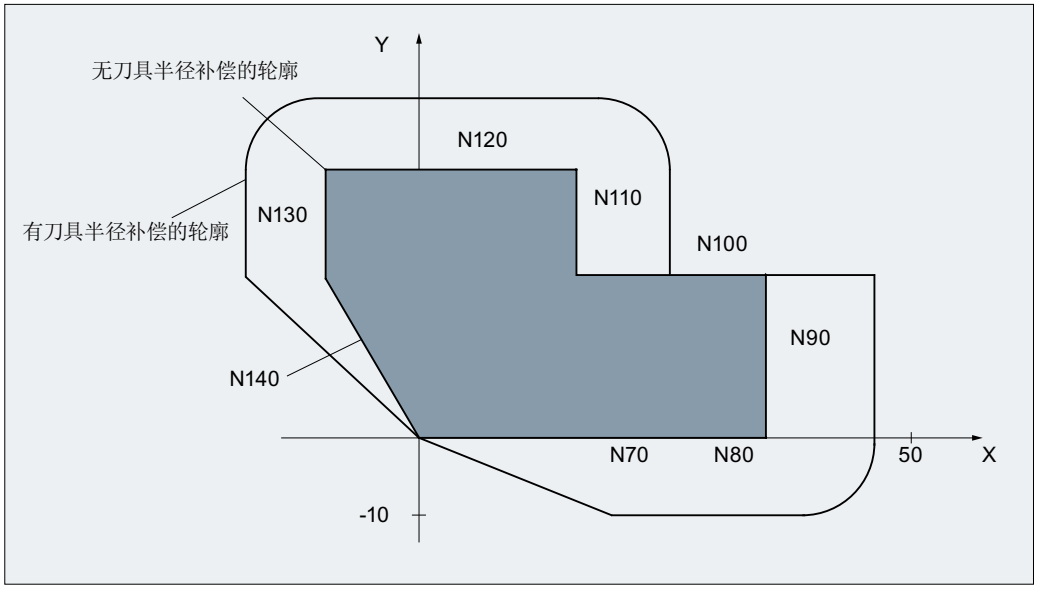


图 19-37　用于补偿抑制的编程示例

特殊情况

- 如果未激活刀具半径补偿（G40），则 CUTCONON 无效。不产生报警。但 G 指令保持生效。
  只有当后面的程序段使用 G41 或者使用 G42 启用了刀具半径补偿时，它才有意义。
- 允许在第 7 个 G 代码组中更换 G 代码（刀具半径补偿；G40 / G41 / G42），在 CUTCONON 有效时。更换到 G40 会立即生效。
  此时会将前面程序段运行时所使用的偏移移出。
- 如果在一个程序段中未编程 CUTCONON 或者 CUTCONOF 在有效补偿平面内的运行动作，则其有效性会延续至下一个使用这种运行动作的程序段。
- 如果在刀具半径补偿时编程了 CUTCONON 并且在程序结束前未将其取消，则相关的运行程序段使用最后一个有效的偏移来运行。
  同样的情况也适用于在程序的最后一个运行程序段中重新编程 G41 或 G42 时。
- 如果使用 G41 或 G42 激活了刀具半径补偿并且同时 CUTCONON 已经被激活，则补偿的激活会被延迟到第一个使用 CUTCONOF 的运行程序段。
- 在使用 CUTCONOF 再次返回轮廓时会运用第 17 个 G 代码组（刀具补偿时的逼近和回退特性；NORM / KONT），即在 KONT 时如有必要将插入绕行圆弧。插入绕行圆弧的条件和使用 G41 或 G42 激活刀具半径补偿时的条件相同。
- 使用受抑制刀具半径补偿程序段的数量可以进行限制：
  MD20252 $MC_CUTCOM_MAXNUM_SUPPR_BLOCKS （使用补偿抑制的最大程序段数量）
  如果超过限制，会发出故障报警并中断加工。
  限制是必要的，因为在重新定位时内部程序段加工必须重新接收 CUTCONON 之前的最后一个程序段。
- 刀具半径补偿有效时重新编程 G41 或 G42 的特性与补偿抑制时的情况类似。
  有以下差异：
  - 只允许线性程序段
  - 会修正单个包含有 G41 或 G42 的运行程序段，使其结束于后续程序段起始点的偏移点处。因此不用再考虑插入一个中间程序段。同样的也适用于一组运行程序段中的最后一个，而这些程序段每个都包含有 G41 或 G42 。
  - 重新逼近轮廓与第 17 组的 G 代码无关（刀具补偿时的逼近和回退特性； NORM / KONT)，而总是使用 NORM。
- 如果在依次连续的运行程序段中多次编程 G41 / G42 ，则会象使用 CUTCONON 时一样直到最后一个程序段才对所有程序段进行加工。

- 只在一组程序段的第一个运行程序段中对轮廓抑制的类型进行计算。
  如果在第一个程序段中既编程了 CUTCONON 又编程了 G41 或 G42，则其特性在轮廓抑制被取消时由 CUTCONON 来确定。
  在这种情况下从 G41 更换到 G42 或者反向更换对重新逼近时更换补偿侧（轮廓的左侧或右侧）非常有意义。
  在轮廓抑制时也可以在后面的程序段中对更换补偿侧（G41 / G42）进行编程。
- 对于所有使用有效轮廓抑制的程序段都关闭了防撞监控或瓶颈识别。

19.7 可定向刀架

19.7.1 简介

引言

在一类机床上可以改变刀具的方向（例如**通过重新调整**）。然而在运行中，设置过一次的方向保持 **固定**，特别是在运行时不能改变。由于这个原因，对于这类机床而言运动关系方向转换（3 轴， 4 轴 或 5 轴转换， TRAORI） 即没有必要也没有意义。但还是有必要考虑通过变向所确定的刀具长度组件，而无需让用户去承担所需的计算。由控制器接管这些计算。

可用性

在 SINUMERIK 828D 上“可定向刀架”功能仅供方案使用。

数据说明

如果控制器要在使用可定向刀架时考虑刀具补偿，则必须有：

- 刀具数据 （几何数据，磨损 ...）
- 刀架数据 （对可定向刀架几何数据的说明）

选择刀架

对于“可定向刀架”功能必须向控制器说明已定义的刀架。为此提供有 NC 程序指令：

```
TCARR = m
```

m:刀架编号

可以将刀架与说明了其几何数据的刀架数据组连接。

激活刀架及其数据组会立即生效，即从下一个运行程序段开始。

分配刀具/刀架

将先前有效的刀具分配给新选择的刀架。

从控制器的角度，刀架编号 m 和刀具编号 T 可以自由组合。在实际应用中可能会出于加工以及机械方面的原因排除这些组合。控制器不会检查组合是否有意义。

刀架的运动说明

可定向刀架的运动由总共 33 个参数组进行说明。

数据组的数据可以由用户处理。

可定向刀架

示例：使用两根轴进行刀具定向的万向刀架

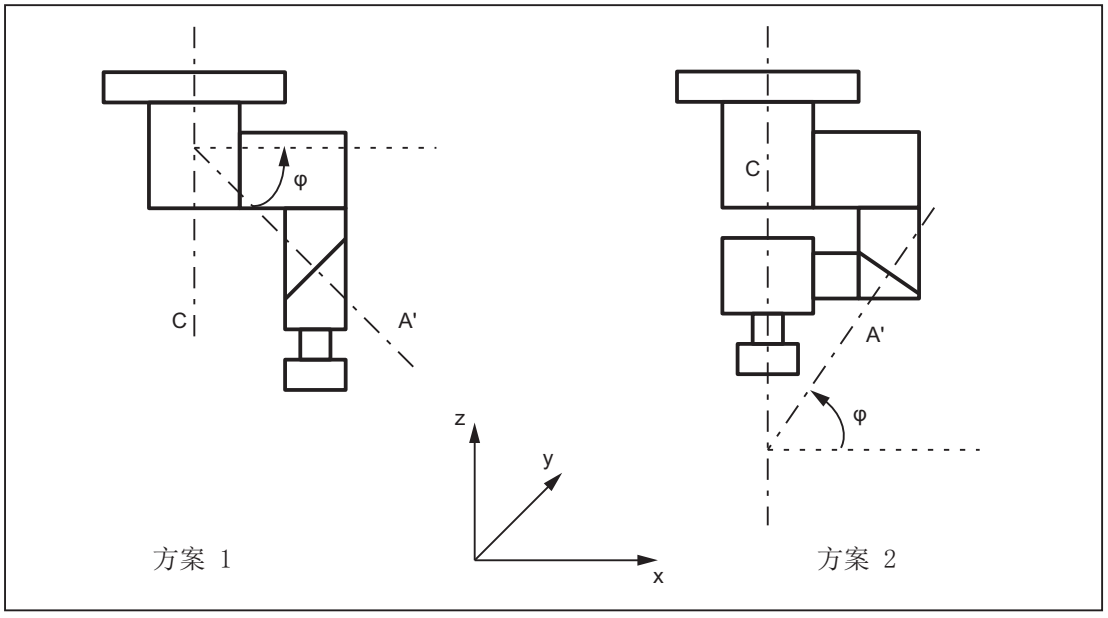


图 19-38　使用两根轴的万向刀架

处理刀架数据组

提供有两种可能：

- 在零件程序的刀架数据组中显性记录
- 自动接收来自框架的一些值（角度）
  此处前提条件是，在选择刀架时另外设定了 TCOFR (Tool Carrier Orientation FRame)。
  在更换刀架时会重新由该时间点有效的框架中获取用于刀具长度计算的刀具方向。

Z 方向上的定向

G 指令 TOFRAME 会定义一个框架，使得该框架内的 Z 方向等于当前的刀具方向。

如果未激活刀架或者激活的刀架没有方向转换，则新框架中的 Z 方向为：

- 在 G17 时等于原来的 Z 方向。
- 在 G18 时等于原来的 Y 方向。
- 在 G19 时等于原来的 X 方向。

TCOABS 在框架有效时

设定绝对的刀架方向要通过：

TCOABS (Tool Carrier Orientation ABSolut)

刀具长度补偿时所考虑的方向 **不取决于** 有效框架的方向。

指令 TCOABS 或 TCOFR 每次只能有一个有效。

框架转换

用户可以在选择刀具之后转换框架。这不会对刀具长度补偿的分量量造成影响。

刀架数据中的角度：

可以通过使用框架所确定的旋转角度来影响刀架数据中所保存的编程旋转角度。当从 TCOFR 转换至 TCOABS 时刀架数据中原始的（编程的）旋转角度重新有效。

刀具补偿类型

刀具半径补偿会考虑当前的刀具方向，如果 CUT2D 或 CUT3DFS 有效的话。

所有其他刀具补偿类型

这是 G 指令组 22 除了 CUT3DC 和 CUT3DF 之外的所有补偿类型。其特性与用于所使用平面的补偿相比没有变化。它们与刀具方向无关而是由有效的框架来确定。

在 CUT2DF 和 CUT3DFF 时用于刀具半径补偿的补偿平面由框架采集而 **不取决于** 当前的刀具方向。要考虑有效的平面（G17/G18/G19）。

CUT3DC 和 CUT3DF

用于圆周铣削的 3D 刀具补偿

在 5 轴转换有效时用于端面铣削的 3D 刀具补偿与“可定向刀架”功能无关。

通过有效的运动 5 轴转换来获取方向信息。

受限制的刀架方向

如果通过框架确定了一个方向，而使用所定义的刀架运动却无法达到该方向，则会输出报警。

下列运动无法达到每个方向：

- 当用来定义运动的、必要的两根旋转轴不相互垂直时并且定义刀具方向的刀具轴与第二根旋转轴不垂直时，
  或者
- 当定义的轴少于两根时。

没有旋转的刀架

内部使用的刀具方向仅与刀具的初始方向以及有效的平面（G17 - G19）有关。

多重含义

通过框架预设的刀具方向通常可以使用两根轴通过 **两对** 不同的旋转角度进行设置。控制器会从这两种可能的设置中选出旋转角度与所编程旋转角度尽可能接近的那个设置。

在刀架数据中保存角度

在差不多所有出现多重含义的情况下，都需要将从框架得出的所需角度保存至刀架数据中。

参数组

用于可定向刀架的一个完整的参数组由 33 个值组成。

有下列系统变量可供使用：

- $TC_CARR1 到 $TC_CARR33
- $TC_CARR34 到 $TC_CARR65 精偏移时额外供用户自由使用

单个参数的含义有下列区别。

机床的运动：

$TC_CARR1 bis $TC_CARR20 以及 $TC_CARR23

其中 $TC_CARR18 到 $TC_CARR20 定义一个扩展矢量 l_4，用来在分解运动时（不仅刀具可旋转而且工件也可旋转）进行机床说明。

$TC_CARR21 和 $TC_CARR22 包含有回转轴的通道轴名称，在其位置上可能会在确定可定向刀架方向时进行存取。

运动类型：

$TC_CARR23 借助字母 T、 P 或 M

在运动类型时有下列三种可能，允许使用大、小写：

T: 只有刀具 （**T**ool） 是可旋转的 （基准值）。

P： 只有工件 （**P**art）是可旋转的。

M: 刀具和工件可旋转 （**M**ixed 模式）。

除了列出的三个符号之外，其他尝试激活可定向刀架的符号都会引起报警：

报警 “14153 通道 %1 程序段 %2 无法识别的刀架类型：%3”

旋转轴的参数设定：

$TC_CARR24 到 $TC_CARR33

在 $TC_CARR24 到 $TC_CARR33 中使用系统变量可以对偏移、角度偏移、切端面齿以及轴极限进行参数设置。

说明

这些系统变量不仅在刀具管理有效时可用而且在其无效时也可以使用。

链/数据组的分量与预占用

刀架数据组中 $TC_CARR1 到 $TC_CARR20 与 $TC_CARR24 - $TC_CARR33 的值为 NC 语言格式类型 REAL。

$TC_CARR21 和 $TC_CARR22 的值，用于第一根旋转轴（$v_1$） 和第二根旋转轴（$v_2$）轴名称，为 NC 语言格式类型 AXIS。它们全部被预置为零。

$TC_CARR23 的值使用大写字母 “T”（只有刀具是可旋转的）预置。

$TC_CARRn[m]

$TC_CARR[0]= 0 有特殊含义

用于可定向刀架的系统变量

$TC_CARRn[m]

n: 参数 1...33

m: 可定向刀架的编号 1...机床数据的值：

MD18088 $MN_MM_NUM_TOOL_CARRIER （可定义刀架的最大数量）

说明	NC 变量	语言 格式	缺省设置
x 分量，偏移矢量 l_1	$TC_CARR1	REAL	0
y 分量，偏移矢量 l_1	$TC_CARR2	REAL	0
z 分量，偏移矢量 l_1	$TC_CARR3	REAL	0
x 分量，偏移矢量 l_2	$TC_CARR4	REAL	0
y 分量，偏移矢量 l_2	$TC_CARR5	REAL	0
z 分量，偏移矢量 l_2	$TC_CARR6	REAL	0
x 分量，旋转轴 v_1	$TC_CARR7	REAL	0
y 分量，旋转轴 v_1	$TC_CARR8	REAL	0
z 分量，旋转轴 v_1	$TC_CARR9	REAL	0
x 分量，旋转轴 v_2	$TC_CARR10	REAL	0
y 分量，旋转轴 v_2	$TC_CARR11	REAL	0
z 分量，旋转轴 v_2	$TC_CARR12	REAL	0
旋转角度 α1 （以度为单位）	$TC_CARR13	REAL	0
旋转角度 α_2 （以度为单位）	$TC_CARR14	REAL	0
x 分量，偏移矢量 l_3	$TC_CARR15	REAL	0
y 分量，偏移矢量 l_3	$TC_CARR16	REAL	0
z 分量，偏移矢量 l_3	$TC_CARR17	REAL	0
x 分量，偏移矢量 l_4	$TC_CARR18	REAL	0
y 分量，偏移矢量 l_4	$TC_CARR19	REAL	0
z 分量，偏移矢量 l_4	$TC_CARR20	REAL	0

说明	NC 变量	语言 格式	缺省设置
轴名称，旋转轴 v_1	$TC_CARR21	AXIS	0
轴名称，旋转轴 v_2	$TC_CARR22	AXIS	0
运动类型	$TC_CARR23	CHAR	T
偏移，旋转轴 v_1	$TC_CARR24	REAL	0
偏移，旋转轴 v_2	$TC_CARR25	REAL	0
角度偏移，旋转轴 v_1 （切端面齿）	$TC_CARR26	REAL	0
角度偏移，旋转轴 v_2 （切端面齿）	$TC_CARR27	REAL	0
角度增量，旋转轴 v_1 （切端面齿）	$TC_CARR28	REAL	0
角度增量，旋转轴 v_2 （切端面齿）	$TC_CARR29	REAL	0
最小位置，旋转轴 v_1 （软件限位）	$TC_CARR30	REAL	0
最小位置，旋转轴 v_2 （软件限位）	$TC_CARR31	REAL	0
最大位置，旋转轴 v_1 （软件限位）	$TC_CARR32	REAL	0
最大位置，旋转轴 v_2 （软件限位）	$TC_CARR33	REAL	0

精偏移时用户使用的系统变量

- $TC_CARR34 到 $TC_CARR40
  包含供用户自由使用的参数。
- $TC_CARR41 到 $TC_CARR65
  含有可以添加给基本参数中的数值的精密位移参数。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。
- $TC_CARR47 到 $TC_CARR54 以及 $TC_CARR61 到 $TC_CARR63
  如果未定义并且尝试在其上进行读或写的存取操作会导致报警。

说明	NC 变量	语言 格式	缺省设置
刀架名称	$TC_CARR34	字符串[32]	""
轴名称 1	$TC_CARR35 *)	字符串[32]	""
轴名称 2	$TC_CARR36 *)	字符串[32]	""
标记	$TC_CARR37 *)	INT	0
位置分量 X	$TC_CARR38 *)	REAL	0
位置分量 Y	$TC_CARR39 *)	REAL	0

说明	NC 变量	语言 格式	缺省设置
位置分量 Z	$TC_CARR40 *)	REAL	0
精偏移 x 分量，偏移矢量 l_1	$TC_CARR41	REAL	0
精偏移 y 分量，偏移矢量 l_1	$TC_CARR42	REAL	0
精偏移 z 分量，偏移矢量 l_1	$TC_CARR43	REAL	0
精偏移 x 分量，偏移矢量 l_2	$TC_CARR44	REAL	0
精偏移 y 分量，偏移矢量 l_2	$TC_CARR45	REAL	0
精偏移 z 分量，偏移矢量 l_2	$TC_CARR46	REAL	0
精偏移 x 分量，偏移矢量 l_3	$TC_CARR55	REAL	0
精偏移 y 分量，偏移矢量 l_3	$TC_CARR56	REAL	0
精偏移 z 分量，偏移矢量 l_3	$TC_CARR57	REAL	0
精偏移 x 分量，偏移矢量 l_4	$TC_CARR58	REAL	0
精偏移 y 分量，偏移矢量 l_4	$TC_CARR59	REAL	0
精偏移 z 分量，偏移矢量 l_4	$TC_CARR60	REAL	0
精偏移的偏移值，旋转轴 v_1	$TC_CARR64	REAL	0
精偏移的偏移值，旋转轴 v_2	$TC_CARR65	REAL	0
注：			
*)	系统变量 $TC_CARR35 到 $TC_CARR40 可以在测量循环以及 ShopMill 和 ShopTurn 中使用。		

19.7.2 运动关系与机床结构

运动链显示

运动链用来说明参考点和刀尖之间的运动关系。

关系链以概要方式给出了刀架数据组所需要的全部数据。对于一个特定运动关系的具体情况，必须使用实际的矢量、长度和角度来占用关系链中相应的分量。关系链可以显示所允许的最大阵列。在简单的情况下单个分量可能为零（例如只有一根旋转轴或者根本没有旋转轴的运动关系）。

在机床上不得存在会使刀具和/或工件旋转的轴。只有当例如手动通过手轮设定方向或者改变方向时，该功能才可以使用。

通过下列数据说明机床结构：

- 两根旋转轴（v_1 和 v_2）每次使用一个旋转角度（α_1 或 α_2）， 该角度在顺时针旋转和指向旋转矢量方向时为正。
- 直到有四个偏移矢量（l_1 到 l_4） 用于相关的机床尺寸（轴距，与机床参考点或刀具参考点的距离）。

零矢量

矢量 v_1 或 v_2 可能为零。相应的旋转角度（显性编程或者从有效的框架计算）必须同样为零，因为未定义旋转轴的方向。如果该条件不满足，在激活刀架时会输出报警。

少于两根旋转轴

当所说明的刀架只能在一个平面内旋转刀具时，有一根旋转轴未定义的情况才有意义。因此刀架数据中一个有意义的最小数据组可以由一个单独的零偏记录构成，即用来说明轴向旋转轴的分量 v_1 或 v_2 中的一个值，此时相应的旋转角度 α1 或 α2 由框架来确定。

其他特殊情况

两个矢量 v_1 或 v_2 有可能共线。此时方向不再有自由角度，也就是说这类的运动关系与只定义了一根旋转轴的情况相同。所有可能的方向位于一个圆锥曲面上。圆锥曲面会退化为一条直线，如果另外刀具方向 t 和 v_1 或 v_2 也共线的话，即在特别情况下不能再改变方向。圆锥曲面会退化为一个圆面（也就是说所有的方向可能都在一个平面内），如果刀具方向 t 和 v_1 或 v_2 相互垂直的话。

允许两个矢量 v_1 和 v_2 都变为零。这时当然不能再改变方向。在这种特别情况下可能不等于零的长度 l_1 和 l_2 生效如附加的刀具长度补偿，转换平面（G17 - G19）对单根轴上的分量没有影响。

运动数据扩展

- 为了通过回转轴位置来定义刀架的设置，能够直接在现有的机床数据上存取。
- 使用可旋转工件的运动扩展或使用可旋转刀具和可旋转工件的运动扩展。
- 用于回转轴位置在标度中仅允许使用离散值的可能（切端面齿）。

扩展与原先的软件版本兼容并包括运动数据组 $TC_CARR18 bis $TC_CARR23。

使用可旋转刀具的车床

在使用可旋转刀具的车床上运动的定义与旧的软件版本相比没有变化。特别是新插入的矢量 l_4 没有意义。未计算 l_4 中可能不等于零的内容。

根本不再测量用于新的运动类型的“可定向刀架”概念，此时刀具台能够独自或者额外向刀具旋转。但出于兼容性的原因仍将其保留。

下面的图中显示了用来说明使用可旋转刀具机床（通常情况）的运动链：

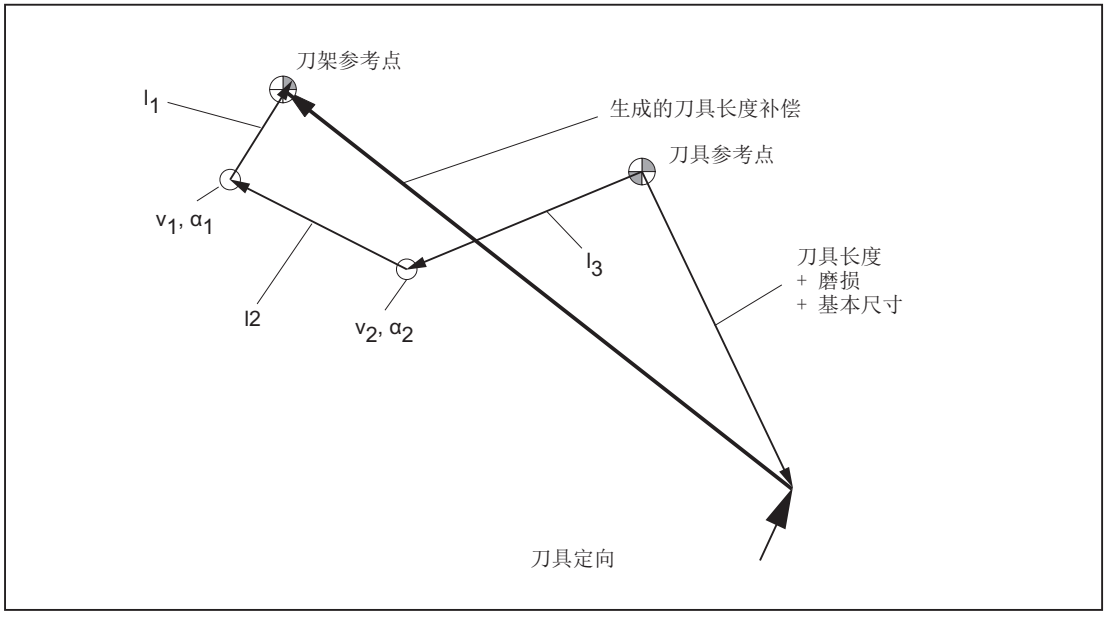


图 19-39　用来说明定向刀具的运动链

说明旋转头中偏移的矢量，从刀尖到刀架参考点的方向被定义为正方向。

对于使用可旋转刀具的机床可以定义下列运动类型：

$TC_CARR23 利用字母 T

使用可旋转工件的车床

在使用可旋转工件的车床上矢量 l_1 没有意义。不计算可能不为零的内容。

在下图中显示了使用可旋转工件机床的相应运动链

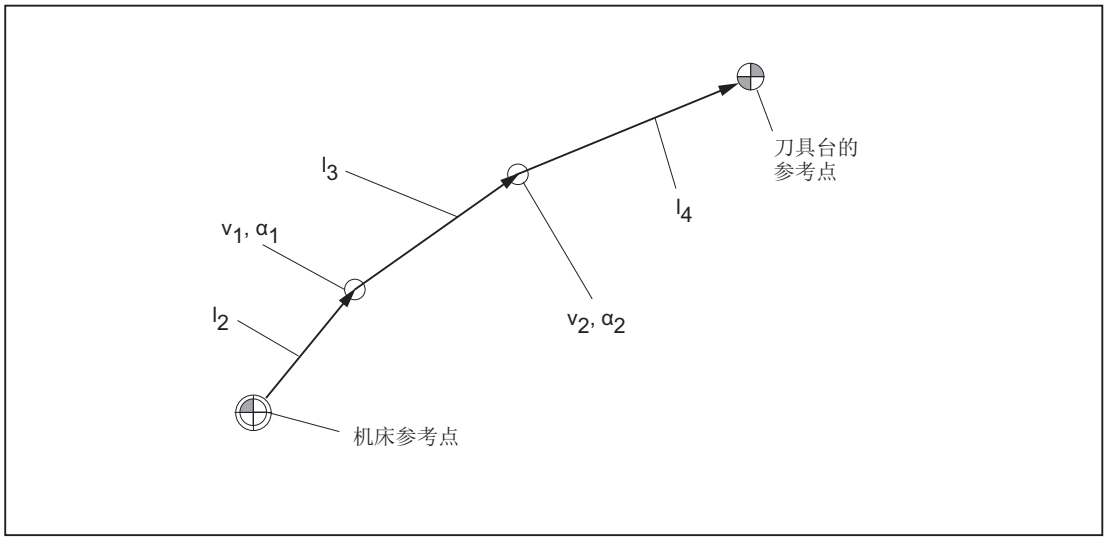


图 19-40　运动链用来说明一个可旋转的刀具台

说明台上偏移的矢量，从机床参考点到台子的方向被定义为正方向。

对于使用可旋转工件的机床可以定义下列运动类型：

$TC_CARR23 利用字母 P

说明

在使用可旋转工件的机床上，将机床参考点和刀具台参考点选择为同一点通常是有意义的。这样选择参考点的优点是，当激活旋转台时初始设置（即在不旋转回转轴时）中的工件零点位置不发生变化。然后关闭（未占用的）运动链（参见图示）。

因此在这种特殊情况下适用：$l_2 = - (l_3 + l_4)$

使用分解运动的机床

在使用分解运动的机床上（不仅刀具可旋转而且工件也可旋转）这两个分量的每一个只能使用一根轴进行旋转。

使用第一根旋转轴（v_1）和两个矢量 l_1 与 l_2 来说明可旋转刀具的运动，使用第二根旋转同（v_2）和两个矢量 l_3 与 l_4 来说明可旋转的平台。分链下图中显示了使用可旋转刀具和工件的机床上的两个运动关系。

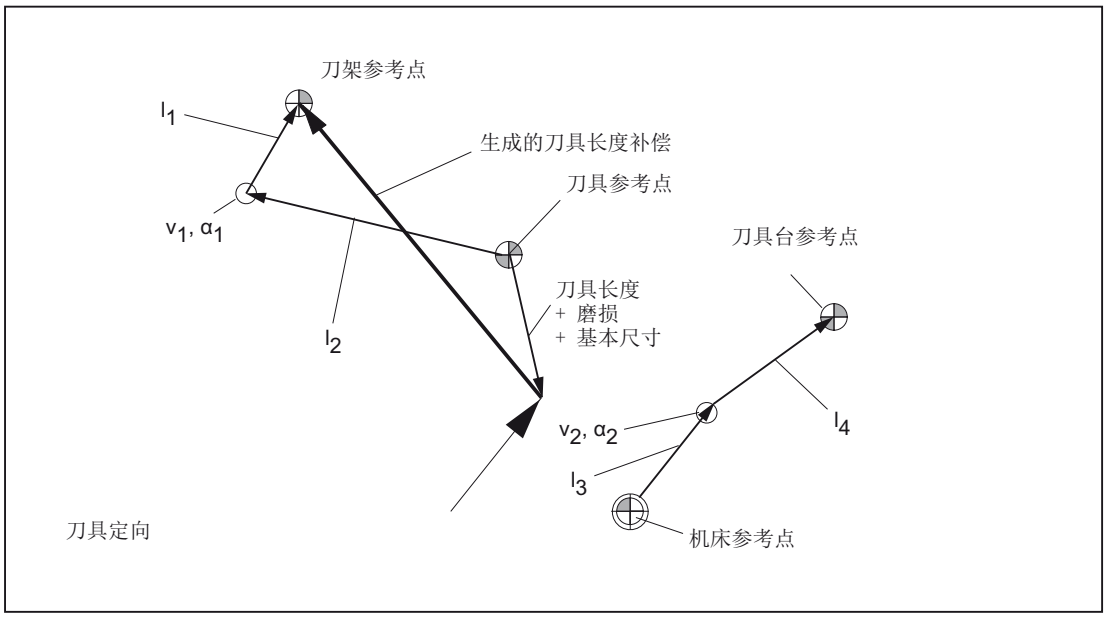


图 19-41　分解运动的运动链

对于使用可旋转刀具和可旋转工件的机床可以定义下列运动类型：

$TC_CARR23 利用字母 M （分解运动）

说明

出于同样的原因，机床运动在使用分解运动时和只有刀具台可旋转的情况一样，将机床参考点和刀具台参考点选择为同一点通常是有意义的。然后关闭（未占用的）用于说明刀具台的关系分链（参见图示）。

在特殊情况下适用：$l_3 = - l_4$

刀具可旋转， 类型 T 和 M

在使用可旋转刀具（类型 T 和 M）的机床运动中可定向刀架部分生效，它用于说明刀具部分和刀头部分（与刀具部分相反），并与有效刀具一起象一把新的整体刀具一样起作用。

精偏

偏移矢量 l_1 到 l_4 以及旋转轴的偏移 v_1 和 v_2 可以分别显示为基准值与精偏移之和。可以通过 **将基准值索引加上数值 40** 得出分配给基准值的精偏移参数。

示例：

分配给参数 $TC_CARR5 精偏移值 $TC_CARR45 。

说明

精偏移可用的系统变量 $TC_CARR41 到 $TC_CARR65， 其含义请参见：

文档:

编程手册之工作准备分册；刀具补偿

激活

使用下列设置将精偏移值加到基准值之上：

SD42974 $SC_TOCARR_FINE_CORRECTION = 1（精偏移 TCARR 打开/关闭）

前提条件

所允许使用的精偏移数值是有限制的。

确定所允许的最大数值：

用于：	通过机床数据：
● 矢量的分量 l_1 到 l_4:	MD20188 $MC_TOCARR_FINE_LIM_LIN
● 两根旋转轴的偏移 v_1 和 v_2:	MD20190 $MC_TOCARR_FINE_LIM_ROT

不允许使用的精偏移值才会被识别出来，当：

- 激活的可定向刀架也包含同样的数值时
  与
- 同时设置下列设定数据：
  SD42974 $SC_TOCARR_FINE_CORRECTION

说明旋转

E 用于说明旋转的数据组分别由初始状态旋转轴方向上给定的矢量 v_1 或 v_2 以及角度 α1 或 α2 构成。旋转角度以顺时针方向和旋转矢量的观察方向为正方向。

借助一个框架来确定两个刀架角度 α1 和 α2，与当前选择的有效平面（G17 - G19）无关。

初始设置中的刀具方向（两个角度 α1 和 α2 等于零）为（和标准情况下一样）：

- 在 G17 时平行于 Z。
- 在 G18 时平行于 Y。
- 在 G19 时平行于 X。

分配刀架数据

使用可旋转刀架机床上的示例

在一台使用可定向刀架、运动类型为 T 的机床上，在所示铣刀头上要进行下列设置：

偏移矢量 l_1 的分量 =	(-200, 0, 0)
偏移矢量 l_2 的分量 =	(0, 0, 0)
偏移矢量 l_3 的分量 =	(-100, 0, 0)
旋转轴 v_1 的分量 =	(1, 0, 0)
旋转轴 v_2 的分量 =	(-1, 0, 1)
刀具参考点基础尺寸	(0, 0, 250)

说明

通过机床上的参考点来确定用于基础尺寸的刀具参考点
关于工作空间内参考点的更多信息参见章节“K2：轴、坐标系、框架 (页 765)”。

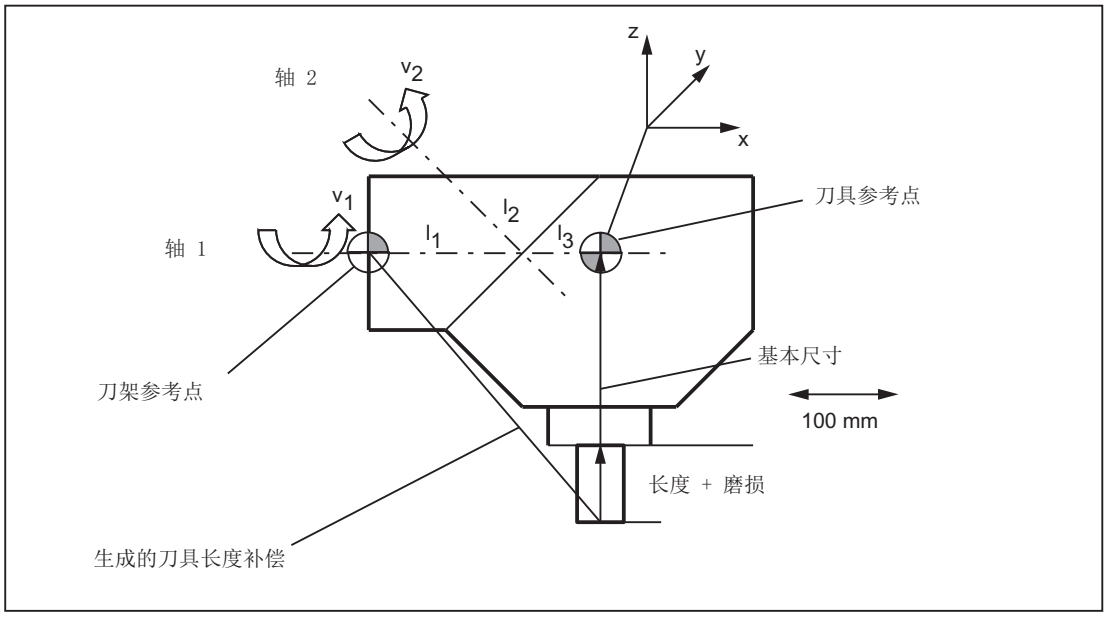


图 19-42　分配刀架数据

对于下列数据组的数值要采用合适的值：

- 两根旋转轴交于一点。
  因此 l_2 的所有分量为零。
- 第一根旋转轴位于 x-z 平面内，第二根回转轴平行于 x 轴。
  通过该条件来确定 v_1 或 v_2 的方向（其长度 - 只要不等于零 - 没有意义）。
- 刀架的参考点位于从两根回转轴交点朝 x 负方向移动 200 毫米处。
  通过该条件确定 l_1。

给出相应的数据组数值

对于使用可旋转刀架机床的图示刀架需要设定下列相应的数据组数值：

说明	NC 变量	值
x 分量，偏移矢量 l_1	$TC_CARR1	- 200
y 分量，偏移矢量 l_1	$TC_CARR2	0
z 分量，偏移矢量 l_1	$TC_CARR3	0
x 分量，偏移矢量 l_2	$TC_CARR4	0
y 分量，偏移矢量 l_2	$TC_CARR5	0
z 分量，偏移矢量 l_2	$TC_CARR6	0
x 分量，旋转轴 v_1	$TC_CARR7	1

说明	NC 变量	值
y 分量，旋转轴 v_1	$TC_CARR8	0
z 分量，旋转轴 v_1	$TC_CARR9	0
x 分量，旋转轴 v_2	$TC_CARR10	-1
y 分量，旋转轴 v_2	$TC_CARR11	0
z 分量，旋转轴 v_2	$TC_CARR12	1
旋转角度 α1 （以度为单位）	$TC_CARR13	0
旋转角度 α2 （以度为单位）	$TC_CARR14	0
x 分量，偏移矢量 l_3	$TC_CARR15	-100
y 分量，偏移矢量 l_3	$TC_CARR16	0
z 分量，偏移矢量 l_3	$TC_CARR17	0

说明

在示例中所选择的刀架运动中两根旋转轴形成角度为 45°，这样就不能采用任意值作为方向。具体在该示例中没有使用负的 X 分量的方向可以显示。

基础尺寸的 x 分量： 0

基础尺寸的 y 分量： 0

基础尺寸的 z 分量： 250

说明

从刀架的几何数据不能明确确定所需的数据，即用户在一定程度上可以自由选择，将哪些数据保存在哪里。这样就可能出现示例中的可能，只给定了距离两根轴的 z 分量基础尺寸。这种情况下 l_2 将不再为零，而是会包含第二根轴上的该点到第一根轴上另外一点的距离分量。第一根轴上的这个点同样可以自由选择。与点的选择有关 l_1 的选择必须使其可以到达参考点（同样可自由选择）。

通常情况下下列关系成立：通过轴的旋转会发生变化的矢量分量可以任意分布在旋转“之前”和“之后”的矢量上。

19.7.3 使用 3 + 2 轴进行斜面加工

功能性说明

使用 3 + 2 轴进行斜面加工可以对可定向刀架的方案进行扩展，以及说明如何将该方案传送到使用可旋转刀具台的机床和刀具与工作台可以同时变向的机床上。

“使用 3 + 2 轴进行斜面加工”功能用来在空间内进行平面加工，该平面可以相对于主平面 [X-Y 平面 (G17)、 Z-X 平面 (G18)、 Y-Z 平面 (G19)] 进行任意旋转。

通过旋转刀具或工件抑或同时旋转刀具和工件，来实现刀具相对于工件的定向。

此时软件会自动由刀具长度、杠杆臂和旋转轴角度计算出所需的平衡运动。 这时总是要首先设定所需要的方向，然后在加工过程中，例如在斜面上铣削一个形腔时，不再发生改变。

此外还要说明下列 3 个在斜面加工中必需的功能：

- **行程编程** 按刀具的方向，与有效的框架无关
- 通过设定空间角度定义一个 **框架旋转**
- 在可编程框架中保持其余框架部分的情况下定义 **刀具方向上的旋转部分**

5 轴转换的界线

如果需要使刀尖（TCP-Tool Center Point 刀具中心点） 在换向时相对于工件以及在插补时保持不变，则需要使用 5 轴软件。

更多关于 5 轴转换的说明参见：

文档：
功能手册之特殊功能分册； 3 至 5 轴转换 （F2）

确定可定向刀架

可以通过普通的 5 轴运动来模拟可定向刀架，通过刀具补偿存储器中一个总共有 33 个 REAL 值的数据组进行说明。 在一个刀架上（例如铣刀头）如果有两个用于方向设置的回转轴，那么该数值恒定为 31。

在当前的软件版本中，使用总共 47 个 Real 值来说明刀具补偿存储器中的一个数据组。 在刀架上如果有两个用于方向设置的回转轴，那么该数值恒定为 45。

剩余的两个数值可变并用于确定方向。 恒定的数值用来说明偏移、方向和旋转轴的设置可能，可变数值用来说明旋转轴的角度。

19.7.4 使用可旋转工作台的车床

系统变量

在使用 TCOABS 计算有效刀具长度时，到目前为止总是对 $TC_CARR13 或 $TC_CARR14 中所保存的角度进行存取。只要 $TC_CARR21 或 $TC_CARR22 参照的不是回转轴，就一如既往生效。如果 $TC_CARR21 或 $TC_CARR22 指向通道中的回转轴，则将当前程序段开始处相关轴的轴位置作为角度来替代 $TC_CARR13 或 $TC_CARR14 中的记录项。

允许混合的运行方式，也就是说可以从系统变量 $TC_CARR13 或 $TC_CARR14 的记录项获取用于一根轴的角度而从通道位置获取用于其他轴的角度。

因此在机床上 NC 已知用来设置可定向刀架的轴时,可以直接在相应的位置进行存取,而此前则必须对此进行固定操作例如读取系统变量 $AA_IM[Achse] 并将读取操作的结果写入 $TC_CARR13/14 中。因而在读取轴位置时也不用特别考虑隐性的预处理程序停止。

MD20180

回转轴使用编程值或计算出的数值，当机床数据：

MD20180 $MC_TOCARR_ROT_ANGLE_INCR[i] = 0（可定向刀架的回转轴增量）

相反如果该机床数据不等于零，则使用下一个标度点的位置，可以选择合适的整数 n 由下列方式得出：

φ = $MC_TOCARR_ROT_ANGLE_OFFSET[i] + n * $MC_TOCARR_ROT_ANGLE_INCR[i]

当回转轴必须被指数化并因此不能接受任意位置时会需要该功能（例如在切端面齿时）。在系统变量 $P_TCANG[i] 中会提供标度值，在系统变量 $P_TCDIFF[i] 中会提供准确值与标度值之间的差值。

定向框架 TCOFR

使用 TCOFR 时（通过有效框架定义的方向来确定角度）确定角度之后通过有效的框架旋转进行标度。如果因为机床运动而无法设定所要求的方向，则发出报警中断加工。当额定方向与可达到的方向过近时，这也同样适用。尤其是在这样的情况下不能阻止由角度标度所发出的报警。

TCARR 框架偏移

刀架切换引起的框架偏移会在通过 TCARR=... 进行选择时立即生效。而对刀具长度的修改仅在刀具生效时才会立即生效。

TCOFR / TCOABS 框架旋转

激活不会触发框架旋转，已生效的旋转也不会改变。和 T 情形（仅刀具可旋转）相似，用于计算的回转轴位置根据 G 指令 TCOFR / TCOABS 从生效框架的旋转分量或 $TC_CARRn 的记录得出。

激活框架会使工件坐标系中的位置相应改变，而不会引起机床的补偿运动。情形如下图所示：

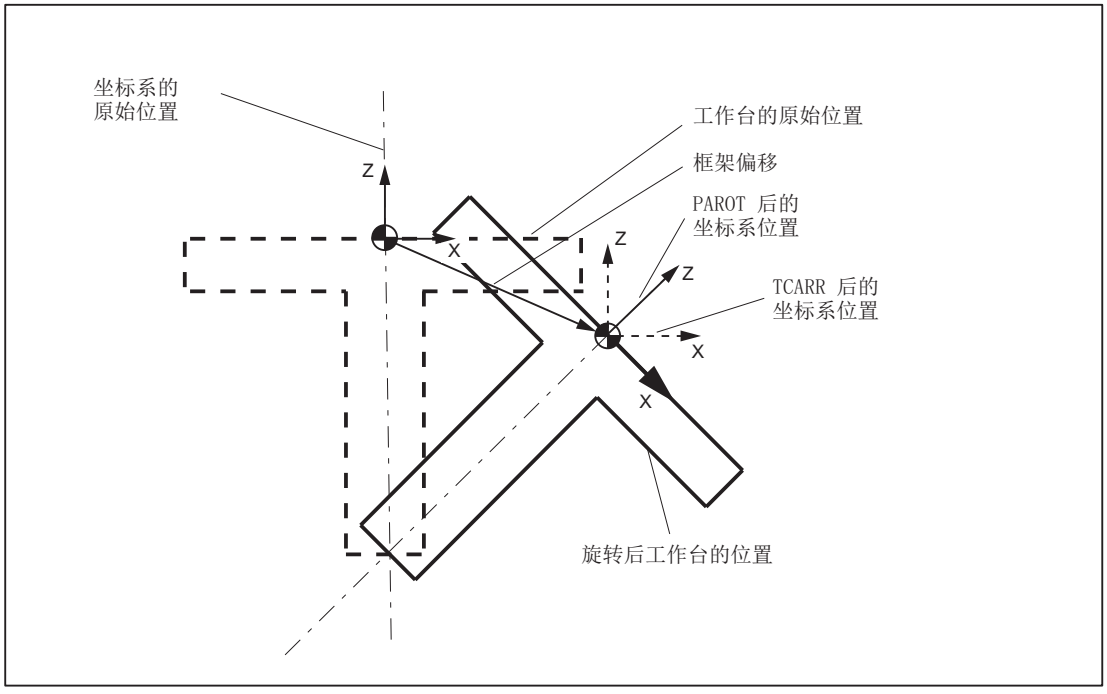


图 19-43　使用 TCARR 激活可旋转刀具台时的零点偏移

示例

图中所示的机床上，工作台的旋转轴指向正的 Y 方向。工作台旋转了 +45 度。使用 PAROT 定义一个框架，使它同样绕 Y 轴旋转 45 度。相对于外界未旋转的坐标系（在图中用“ TCARR 之后的坐标系位置”进行标识）相对于一起运动的坐标系（PAROT 之后的位置）却旋转了 -45 度。若通过 ROT Y-45 定义此坐标系，之后在 TCOFR 生效时选择刀架，则会为刀架的旋转轴得到 +45 度的角度。

可旋转的工作台

在可旋转的工作台上（运动类型 P 和 M）使用 TCARR 进行激活一开始同样不会引起坐标系旋转（见图），也就是说坐标系的零点虽然参考机床发生了偏移并固定以工件零点为基准，但方向仍然保持不变在空间上固定。

运动类型 P 和 M 时激活

在运动类型为 P 和 M 时选择刀架会激活一个添加的框架（可定向刀架的工作台偏移），它会将零点偏移作为工作台旋转的结果进行考虑。

可以在自身的系统变量 $P_PARTFR 中写入零点偏移。为此必须在机床数据中设置位 2：

MD28082 $MC_MM_SYSTEM_FRAME_MASK（系统框架（SRAM））

通过下列机床数据进行标识的基础框架对于零点偏移而言不再是必需的：

MD20184 $MC_TOCARR_BASE_FRAME_NUMBER（用于接收工作台偏移的基础框架编号）

运动类型 M 时激活

在运动类型为 M 时（刀具和工作台可以分别绕轴旋转），使用 TCARR 激活刀架会导致有效的刀具长度（如果刀具有效的话）和零点偏移同时发生变化。

旋转

根据加工任务，在使用可旋转刀架或工作台时，不仅需要考虑零点偏移（作为框架或作为刀具长度），还需考虑旋转。然而激活一个可定向刀架在任何情况下都不会直接导致坐标系旋转。

TOROT

如果刀具可以定向，则能够利用 TOFRAME 或 TOROT 来定义一个框架，使它的 Z 轴指向刀具方向。

PAROT

如果需要一个以工件为固定基准的坐标系，即相对于原始位置不仅位移而且也进行相应的工作台旋转，则与用 PAROT 进行可旋转刀具加工的情形类似，可以激活相应的旋转。

用 PAROT 使有效框架中的平移、比例与镜像保持不变，然而其旋转部分仍通过可定向刀架的旋转部分旋转，与工作台相适应。

PAROT 和 TOROT 在工作台或刀具使用两根旋转轴定向的情况下会考虑整体的方向变化。在混合运动中只考虑由旋转轴引起的、相应部分的情况。例如为了能够使用 TOROT 来旋转工件，使得斜面平行于空间固定的 X-Y 平面，在加工时需要考虑刀具的旋转，例如必须加装不垂直于该平面的钻孔时。

无可定向刀架生效时，语言指令 PAROT•不会被系统拒绝。这不会在编程框架中引起变化。

说明

关于 TCARR 和 TOROT 以及 PAROT 功能相对于通道专用系统框架的的更多说明参见章节“K2：轴、坐标系、框架 (页 765)”。

19.7.5 使用可定向刀架的步骤

新建一个刀架

NC 中可用刀架数据组的数量可以使用机床数据来定义：

MD18088 $MN_MM_NUM_TOOL_CARRIER （可定义刀架的最大数量）

1. 按如下方式计算数值：
   MD18088 = **“TO 单元的数量” * “一个 TO 单元刀架数据组的数量”**
   每个 TO 单元通过 **MD18088 / “ TO 单元”** 进行固定分配。

   说明

   有关 TO 单元定义和分配的其他说明可以通过机床数据：

   MD28085 $MC_MM_LINK_TOA_UNIT （为一个通道分配 TO 单元 （SRAM））

   文档：
   功能手册之扩展功能分册；存储器配置 （S7）

2. 刀架数据归零：
   可以使用指令 $TC_CARR1[0] = 0 将所有数据组的全部值归零。
   可以选择性的使用语言指令 DELTC 或者通过 PI 服务 _N_DELTCAR 来清除单个刀架数据组。

3. 在刀架数据上存取：
   – 零件程序
     → $TC_CARRn[m] = 值
     此时会用新的“值”来覆盖用于刀架 m 的、系统变量原来的值 n 。
     → Wert = $TC_CARRn[m]
     使用“定义实际值” - 这时只要刀架的参数 m 已经定义，就能够将其读取（例如 设置 MD18088）。否则会引发报警。
   – BTSS 接口
   可定向刀架的参数可以通过变量服务 NC-HMI（OPI）使用系统变量 $P_TCANG[<n>] 进行读写。
4. 数据备份：
   上述系统变量可以在通用 NC 数据备份范围内进行存储。

选择刀架

使用编号 m 借助 NC 程序指令 TCARR = m 来选择刀架（TCARRTool Carrier）。

使用 TCARR = 0 可以取消选择一个有效的刀架。

新刀具或新刀架

就象装配在有效刀架上一样对新激活的刀具进行处理。

新刀架在对其编程时即刻生效，无需更换或者重新编程有效刀具。刀架（编号）与刀具（编号）互不相关并且可以任意相互组合。

G 功能组 42 的 G 指令中的刀架

绝对的刀具定向 TCOABS （Tool Carrier Orientation ABSolute）：

如果系统变量 $TC_CARR13 或 $TC_CARR14 中记录了相应的值，并且在 G 功能组 42 中激活了 G 指令 TCOABS，则显性确定刀具定向。

刀具定向框架 TCOFR (Tool Carrier Orientation FRame）：

在选择刀具时同样可以通过有效框架的当前定向自动确定刀具定向，如果在选择刀具时 G 功能组 42 中下列 G 指令之一生效：

- TCOFR 或 TCOFRZ
  设置可定向刀架，使得刀具指向 Z 方向。
- TCOFRX
  设置可定向刀架，使得刀具指向 X 方向。
- TCOFRY
  设置可定向刀架，使得刀具指向 Y 方向。

TCOFR 表示在斜面上进行加工时要显性考虑刀具补偿，如同刀具与此平面垂直。

说明

刀具定向未与框架定向刚性连接。在框架有效并且 G 指令 TCOABS 有效时可以选择刀具，其中刀具的定向与有效框架的定向无关。

在选择刀具之后用户可以更换框架，而这对刀具长度补偿分量没有影响。另外也不再保证刀具垂直于加工平面。因此用户应当预先检查期望的刀具定向在斜面上的保持。

当更换刀架时 TCOFR 等有效，则始终由该时间点有效的框架重新采集用于刀具长度计算的刀具方向。

G 功能组 53 的 G 指令中的刀架

刀具定向可以借助第 53 组 G 指令 （TOFRAME， TOROT 等等）来定义一个框架，在该框架中轴方向（Z、 Y 或 X）等于当前的刀具定向。

在调用 TOFRAME 时第 6 组 G 指令（G17 - G19）对于刀具方向有决定性作用。

如果没有激活刀架，或者激活了一个不会引起刀具定向变化的刀架，则在新框架中 Z 方向：

- 在 G17 时等于原来的 Z 方向。
- 在 G18 时等于原来的 Y 方向。
- 在 G19 时等于原来的 X 方向。

在旋转刀架时该方向会相应修正。相应的适用于新的 X 或 Y 方向。

可以使用 G 指令 TOFRAMEX、TOFRAMEY、TOROTX 或 TOROTY 来代替 TOFRAME 或 TOROT 。这时轴的含义会相应互换。

更换组别

更换第 42 组的 G 指令（TCOABS， TCOFR 等等）会导致重新计算相关的刀具长度分量。

保存在刀架数据（已编程中）的旋转角度不受影响，当从 TCOFR 更换到 TCOABS 时保存在刀架数据中的原始角度重新生效。

读取旋转角度 （α_1 或 α_2）：

用于计算方向的当前角度可以通过系统变量 $P_TCANG[n] 使用 n = 1 或者 n = 2 进行读取。

如果对于一个指定的方向有两个允许的方案（二次生效的一对角度），则可以使用 $P_TCANG[3] 或 $P_TCANG[4] 在相应的值上进行存取。可以使用 $P_TCSOL 读取有效的方案 0 到 2。

使用 CUT2D 或 CUT3DFS 进行刀具半径补偿：

在刀具半径补偿时要考虑当前的刀具方向，当 G 代码组 22 (刀具补偿类型）中 CUT2D 有效或者 CUT3DFS 有效时。

在不旋转刀架上特性仅与第 6 组 G 指令（G17 - G19）的有效平面相关，因此与原来的特性相同。

所有其他刀具补偿类型：

所有其他刀具补偿类型时的特性不变。

尤其在 CUT2DF 和 CUT3DFF 时用于刀具半径补偿的补偿平面由框架采集而不取决于当前的刀具方向。此时要考虑有效平面（G17 - G19），使其特性与原先一致。

第 22 组剩余的两个 G 指令 CUT3DC 和 CUT3DF 不涉及刀架功能，因为这种情况下由有效的运动转换来提供刀具方向信息。

两根旋转轴

对于两根旋转轴通常存在两种方案。控制器会自行从这两个方案中进行选择，使得由框架计算出的方向角度尽可能接近预设的角度。

有两种方法预设角度：

1. $TC_CARR21 或 $TC_CARR22 参照一根回转轴，则在激活了刀架的程序段开始处使用该轴的位置作为预设值。
2. $TC_CARR21 或 $TC_CARR22 不参照回转轴，则使用 $TC_CARR13 或 $TC_CARR14 所包含的值。

示例

控制器首先计算用于一根轴的 10 度角。预设的角度为 750 度。因此输出角度要增加 720 度 （= 2 * 360 度），这样最终的角度为 730 度。

回转轴偏移

借助两个系统变量 $TC_CARR24 和 $TC_CARR25 可以预设回转轴偏移。参数其中一个的值不为零表示，相应回转轴在基础设置中位置不是零，而在通过参数设定的位置上。所有角度参数与根据该进行偏移的坐标系相关。

在更换加工平面时（G17 - G19）只交换有效刀具的刀具长度分量。不交换刀架分量。最终的刀具长度矢量按当前刀架进行旋转并同样使用刀架的偏移值进行修正。

借助一个框架来确定两个刀架角度 α_1 和 α_2，与当前选择的有效平面（G17 - G19）无关。

极限值限制

在系统变量程序块（$TC_CARR30 到 $TC_CARR33）中可以为每根旋转轴设定极限角度（软件限制）用来说明可定向刀架。如果不仅最小值为零而且最大值也为零，则不会运用该极限。

如果两个极限值中有一个不等于零，则会检查先前采集到的方案是否在所允许的范围内。如果不是这种情况，则会尝试, 按照所允许的轴位置增加或减少 360 度的倍数使设置有效。如果无法实现并且存在两个不同的方案，则放弃第一个方案而使用第二个。参考轴极限值按第一个方案的情形来处理第二个方案。

如果放弃了第一个方案并使用第二个替代它，则 $P_TCANG[1/2] 和 $P_TCANG[3/4] 中的内容要互换，这样可以使这种情况下实际运用的方案处于 $P_TCANG[1/2] 中。

如果未计算而是预设了轴角度，则也要监控轴极限值。如果在 TCOABS 有效时激活可定向刀架，就会出现这种情况。

19.7.6 编程

刀架选择

用刀架编号 **m** 选择刀具，使用：

```
TCARR = m
```

在刀架数据组上存取

从零件程序中能够进行下列存取：

参数原来的值 **n**， 用于刀架 **m**， 被新的 “值” **覆盖** ，使用：

```
$TC_CARRn[m] = 值
```

只要已经定义了刀架数据组，就可以 **读取** 刀架 m 的参数，使用：

```
值 = $TC_CARRn[m]
```
（值必须为 REAL 型变量）

刀架数据组编号所在的范围可以通过机床数据来定义：

MD18088 $MN_MM_NUM_TOOL_CARRIER（可定义刀架数据组的总数）

对于 NC 通道可以将该数量分别通过有效通道的数量和刀架数据组的数量来定义。

特例：

如果通过机床数据选择与标准有所偏差的设定：

MD28085 $MC_MM_LINK_TOA_UNIT（为一个通道分配一个 TO 单元）

所有刀架数据组归零

从零件程序中可以使用一条指令清除所有刀架数据组的全部值。

$TC_CARR1[0] = 0

数值不由用户设置，使用 值=0 进行预设。

激活

如果不仅激活了 **刀架** 而且激活了 **刀具**，则刀架有效。单独选择刀架没有效果。刀架选择是否有效取决于 G 指令 TCOABS / TCOFR （刀架模态 G 代码组）。

当刀架有效时，在代码组中 TCOABS / TCOFR 更换 G 指令会导致重新计算刀具长度分量。在 TCOABS 时使用保存在刀架数据中、用于两个旋转角度 α1 和 α2 的值来获取刀具方向。

在 TCOFR 时从当前的框架获取这两个角度。保存在刀架数据中的值仍保持不变。此外，当从框架得出的旋转角度有多重含义时,它们还被用来进行分解。此时要从多个可能的角度中选择与编程角度偏差最小的那个。

说明

关于使用刀架运动刀具补偿编程和关于系统变量的更多说明参见：

文档：
编程手册之工作准备分册

19.7.7 用于定向的前提条件和控制器特性

完全定向

使用说明了指定运动的数据组只能显示所有可以想像的空间方向，当：

- 定义了用于说明旋转轴的两个矢量 v_1 和 v_2 时（也就是说矢量必须不等于零）。
- 两个矢量 v_1 和 v_2 互相垂直时。
- 刀具方向垂直于第二根旋转轴时。

未定义的方向

如果未满足该条件并使用 TCOFR 时通过有效框架所确定的方向不能达到时，会输出报警。

矢量/旋转角度的相关性

如果所描述矢量 v_1 或 v_2 一根旋转轴的方向等于零，则相应的旋转角度 α1 或 α2 也必须等于零。 否则会输出报警。 只有当刀架有效时才会输出报警，即在更换刀架时。

联合使用方向进行刀具精细补偿

不 允许将刀具精细补偿和刀架组合使用。 在刀架有效时激活刀具精细补偿以及反过来在刀具精细补偿有效时激活刀架都会导致报警。

自动选择刀架，复位

在 RESET 复位 或者程序启动时可以通过机床数据自动选择刀架：

MD20126 $MC_TOOL_CARRIER_RESET_VALUE （复位时刀架有效）

类似于受控制的刀具选择通过机床数据来处理：

MD20120 $MC_TOOL_RESET_VALUE （刀具长度补偿，引导启动（复位/零件程序结束））

在 复位 或者程序启动时的特性与刀具选择时一样通过机床数据中的位 6 来控制：

MD20110 $MC_RESET_MODE_MASK （确定复位/零件程序结束后控制器的初始设置）

或者：

MD20112 $MC_START_MODE_MASK （确定 NC 启动时控制器的初始设置）

更多信息参见章节“K1：BAG、通道、程序运行、复位特性 (页 519)”。

自软件版本 SW6.3 起

如果在复位之前最后一次选择时 TCOABS 有效，则特性相对于前面的状态保持不变。 其他的 G 代码有效会导致，使用复位之前最后进行选择时有效的框架来激活可定向刀架。 此时也要考虑改变了的刀架数据（$TC_CARR...）。 如果该数据未改变，则刀架被激活至复位前相同的状态。 如果复位前最后一次选择刀架时改变了刀架数据，则选择无法始终与旧的框架保持一致。 在这种情况下，会根据该时间点有效的第 42 组 G 代码值以及有效的框架来选择可定向刀架。

MD22530 辅助功能输出至 PLC

选择刀架时可以选择输出常数或者 M 代码，其号码由刀架编号来推导，这可以使用机床数据进行设置：

MD22530 $MC_TOCARR_CHANGE_M_CODE（刀架切换时的 M 代码）

更多信息参见章节“K1：BAG、通道、程序运行、复位特性 (页 519)”。

刀架的运动

对于刀架运动要注意下列前提条件：

- 初始设置中的刀具方向，两个角度 α_1 和 α_2 为零，和标准情况下一样：
  - G17 时平行于 Z
  - G18 时平行于 Y
  - G19 时平行于 X
- 参考轴极限值必须要达到所允许的设置。
- 只有当两根回转轴相互垂直时才可以任意设置方向。
  另外在两根轴能够转动工作台的机床上，刀具方向与第一根旋转轴垂直。
  在使用混合运动的机床上，刀具方向必须与转动刀具的轴垂直，也就是说同样垂直于第一根旋转轴。

对于 **在一个框架中** 确定方向适用下列条件：

- 使用定义过的刀架运动必须要达到框架中所确定的方向，否则会输出报警。
  当两根用于定义运动的旋转轴不相互垂直时，总是会发生这种情况。
  当定义的旋转轴少于两根时这也适用，情况如下：
  - 当 **运动类型为 T 并使用可旋转刀具**时，定义刀具方向的刀具轴不垂直于 **第二** 根轴。
  - 当 **运动类型为 M 和 P 并使用可旋转工件**时，定义刀具方向的刀具轴不垂直于 **第一** 根轴。
- 旋转轴为了达到指定位置需要使用预设了刀具方向的框架，而只有在一根旋转轴的情况下才能清楚的确定它们。 对于两根旋转轴通常存在两种方案。
- 在所有需要考虑多重含义的情形中，尤为重要的是, 将由框架数据得出的预期角度保存在刀具数据中或者旋转轴要位于预期位置的附近。

ASUP， REPOS 时的特性

可以在异步子程序 ASUP 中更换刀架。 使用 REPOS 继续被中断的程序时需要考虑逼近运行时的新刀架并用它继续运行程序。 此处的处理与 ASUP 中的换刀处理类似。

更多信息参见章节“K1：BAG、通道、程序运行、复位特性 (页 519)”。

19.7.8 示例

19.7.8.1 示例：可定向刀架

要求

在后续示例中所使用的刀架可以通过绕 Y 轴旋转被完整描述。因此它能够使用一个值的记录项来定义旋转轴 （程序段 N20）。

在程序段 N50 到 N70 中说明了一把长度为 20 毫米并且半径为 5 毫米的带柄铣刀。

在程序段 N90 中定义了一个绕 Y 轴 37 度的旋转。

在程序段 N120 中激活了刀具半径补偿，并且为了能够在后续程序段中、在绕 Y 轴旋转 37 度的情况下进行补偿加工，进行了所有的设置。

```
N10                                          ; 定义刀架 1
N20 $TC_CARR8[1] = 1                         ;  Y 方向上第一根旋转轴的分量
N30
N40                                          ; 定义刀具补偿存储器 T1/D1
N50 $TC_DP1[1,1] = 120                       ; 带柄铣刀
N60 $TC_DP3[1,1] = 20                        ; 长度 1
N70 $TC_DP6[1,1] = 5                         ; 半径
N80
N90 ROT Y37                                  ; 绕 y 轴旋转 37 度
N100
N110 X0 Y0 Z0 F10000
N120 G42 CUT2DF TCOFR TCARR = 1 T1 D1 X10
N130 X40
N140 Y40
N150 X0
N160 Y0
N170 M30
```

19.7.8.2 使用可旋转平台的可定向刀架示例

使用 MOV 指令

使用 MOVT 指令要假定程序在 5 轴机床上运行，并且运行时刀具的 B 轴绕 Y 轴旋转：

```
N10 TRAORI()
N20 X0 X0 Z0 B45 F2000                       ; 设置刀具方向
```

```
N30 MOVT=-10                                ; 在刀具方向上进给运行 10 毫米
                                            ; （在 Y-Z 平面内少于 45 度）
N40 MOVT=AC(20)                             ; 在刀具方向上返回到距离
                                            ; 零点 20 毫米处
```

使用可旋转平台的车床

使用可旋转平台的可定向刀架的完整定义：

```
N10 $TC_DP1[1,1]=120
N20 $TC_DP3[1,1]=13                         ; 刀具长度 13 毫米

; 定义刀架 1 :

N30 $TC_CARR1[1] = 0                        ; 偏移矢量 1 的 X 分量偏移矢量
N40 $TC_CARR2[1] = 0                        ; 偏移矢量 1 的 Y 分量偏移矢量
N50 $TC_CARR3[1] = 0                        ; 偏移矢量 1 的 Z 分量偏移矢量

N60 $TC_CARR4[1] = 0                        ; 偏移矢量 2 的 X 分量偏移矢量
N70 $TC_CARR5[1] = 0                        ; 偏移矢量 2 的 Y 分量偏移矢量
N80 $TC_CARR6[1] = -15                      ; 偏移矢量 2 的 Z 分量偏移矢量

N90 $TC_CARR7[1] = 1                        ; 第 1 根轴的 X 轴
N100 $TC_CARR8[1] = 0                       ; 第 1 根轴的 Y 轴
N110 $TC_CARR9[1] = 0                       ; 第 1 根轴的 Z 轴

N120 $TC_CARR10[1] = 0                      ; 第 2 根轴的 X 轴
N130 $TC_CARR11[1] = 1                      ; 第 2 根轴的 Y 轴
N140 $TC_CARR12[1] = 0                      ; 第 2 根轴的 Z 轴

N150 $TC_CARR13[1] = 30                     ; 第 1 根轴的轴
N160 $TC_CARR14[1] =-30                     ; 第 2 根轴的轴

N170 $TC_CARR15[1] = 0                      ; 偏移矢量 3 的 X 分量偏移矢量
N180 $TC_CARR16[1] = 0                      ; 偏移矢量 3 的 Y 分量偏移矢量
N190 $TC_CARR17[1] = 0                      ; 偏移矢量 3 的 Z 分量偏移矢量

N200 $TC_CARR18[1] = 0                      ; 偏移矢量 4 的 X 分量偏移矢量
N210 $TC_CARR19[1] = 0                      ; 偏移矢量 4 的 Y 分量偏移矢量
N220 $TC_CARR20[1] = 15                     ; 偏移矢量 4 的 Z 分量偏移矢量

N230 $TC_CARR21[1] = A                      ; 用于第 1 根轴的轴
N240 $TC_CARR22[1] = B                      ; 用于第 2 根轴的轴
N250 $TC_CARR23[1] = "P"                    ; 刀架的类型
```

```
N260 X0 Y0 Z0 A0 B45 F2000
N270 TCARR=1 X0 Y10 Z0 T1 TCOABS
N280 PAROT
N290 X0 Y0 Z0
N300 G18 MOVT=AC(20)
N310 G17 X10 Y0 Z0
N320 MOVT=-10
N330 PAROTOF
N340 TCOFR
N350 X10 Y10 Z-13 A0 B0
N360 ROTS X-45 Y45
N370 X20 Y0 Z0 D0
N380 Y20
N390 X0 Y0 Z20
N400 M30
```

给出了可定向刀架的完整定义。包含数值 0 的分量不必要给定，因为本来就会使用零对其进行预置。

在 N270 中激活刀架。

因为在 *$TC_CARR21* 中或者在 *$TC_CARR22* 中涉及加工轴 A 和 B 并且 TCOABS 有效，所以 *$TC_CARR13* 和 *$TC_CARR14* 中的记录项被忽略，即对于旋转使用轴位置 A0 B45 。

第 4 个偏移矢量（在 Z 方向上长度 15 毫米）绕 B 轴的旋转会引起零点偏移，偏移大小为 X10.607 [= 15 * sin(45)] 和 Z-4.393 [= -15 * (1. - cos(45))]。自动说明的基准框架或系统框架会考虑该零点偏移，以至于逼近位置 X10.607 Y10.000 Z8.607 。此时在 Z 方向上通过刀具选择得出一个额外的 13 毫米的偏移，平台旋转不影响 Y 分量。

在 N280 中根据可定向刀架平台的转动来定义一个旋转。因此新的 X 方向指向第 4 象限中的等分角，而新的 Z 轴沿着第 1 象限中等分角的方向。

在 N290 中逼近零点，即机床位置 X10.607 Y0 Z-4.393，因为旋转没有改变零点的位置。

在 N300 中按 Y 方向运行至位置 Y33.000 ，因为 G18 有效并且 Y 分量不受有效框架的影响。X 和 Z 位置保持不变。

在 N310 中逼近位置 X17.678 Y0 Z1.536 。

在 N320 中只有 Z 位置作为 MOVT 指令的结果变成数值 -8.464。因为只有平台可以旋转，所以刀具方向继续平行于机床 Z 方向，即使有效框架的 Z 方向已经旋转了 45 度。

N330 删除基准或系统框架，则 N280 中的框架定义也被取消。

在 N340 中使用 TCOFR 给定，可定向刀架应当按照有效框架进行定位。由于 PAROTOF 指令在 N330 中旋转不再有效，所以将初始设定当作生成的结果。框架偏移变为 0 。

因此在 N350 中逼近位置 X10 X10 Z0 （= Z-13 + 刀具长度）。注意：通过同时编程两根回转轴 A 和 B 可以使可定向刀架的实际位置与 N340 中所使用的位置一致。由三根线性轴逼近的位置与此无关。

在 N360 中借助空间角度定义一个平面，它在 X-Z 平面和 Y-Z 平面中的交线分别与 X 轴或 Y 轴成 +45 度或 -45 度角。这样定义的平面位置如下：平面法线指向空间对角线方向。

在 N370 中，在新的坐标系内运行至位置 X20 Y0 Z0 。因为同时取消了使用 D0 的刀具选择所以在 Z 方向上不再有额外的偏移。因为新的 X 轴位于从前的 X-Z 平面内，所以在这个程序段中会到达机床位置 X14.142 Y0 Z-14.142 。

在 N380 中，在旋转过的坐标系内只处理 Y 轴。这会引起全部三根加工轴的运动。机床位置为 X5.977 Y16.330 Z-22.307。

在 N390 中逼近位于新的 Z 轴上的一个点。依照机床数据它处于空间对角线上。因此全部三根轴会到达位置 11.547。

19.7.8.3 计算使用位置和工件专用的补偿

使用适配器的刀具

在后续的程序示例中定义了一把使用适配器和可定向刀架的刀具。 为了保持关系的清晰，在刀具上、在总量和安装补偿时以及在适配器上各自只有长度 L1 和零不一样。 可定向刀架的偏移矢量都为零。

N10	$TC_TP2[1] = "MillingTool"	; 标识符
N20	$TC_TP7[1] = 9	; 位置类型
N30	$TC_TP8[1] = 2	; 状态： 使能和未禁止
; D 校验 D=1		
N40	$TC_DP1[1,1]=120	; 刀具类型 - 铣削
N50	$TC_DP3[1,1]=; 长度补偿矢量	
N60	$TC_DP12[1,1]= ; 磨损	
N70	$TC_SCP13[1,1]=0.1	; 总补偿 DL=1
N80	$TC_ECP13[1,1]=0.01	; 安装补偿 DL=1
N90	$TC_ADPTT[1]=5	; 适配器转换
N100	$TC_ADPT1[1]=0.001	; 适配器尺寸

		; 刀库数据
N110	$TC_MAP1[1]=3	; 刀库的类型： 刀塔
N120	$TC_MAP2[1]="Revolver"	; 刀库的名称
N130	$TC_MAP3[1]=17	; 刀库的状态
N140	$TC_MAP6[1]=1	; 维度 - 行
N150	$TC_MAP7[1]=2	; 维度 - 列 -> 2 个位置
N160	$TC_MPP1[1,1]=1	； 刀位类型
N170	$TC_MPP2[1,1]=9	; 位置类型
N180	$TC_MPP4[1,1]=2	； 刀位状态
N190	$TC_MPP7[1,1]=1	; 将适配器带到位置上
N200	$TC_MPP6[1,1]=1	; T 号码 “MillingTool 铣刀”
N210	$TC_MAP1[9999]=7	; 刀库的类型： Zw.存储器
N220	$TC_MAP2[9999]="Zw.speicher"	; 刀库的名称
N230	$TC_MAP3[9999]=17	; 刀库的状态
N240	$TC_MAP6[9999]=1	; 维度 - 行
N250	$TC_MAP7[9999]=1	; 维度 - 列 -> 1 个位置
N260	$TC_MPP1[9999.1]=2	； 刀位类型
N270	$TC_MPP2[9999.1]=9	; 位置类型
N280	$TC_MPP4[9999.1]=2	； 刀位状态
N290	$TC_MPP5[9999,1]=1	; 主轴编号 1
N300	$TC_MDP2[1,1]=0	; 主轴到刀库 1 的距离
		; 定义刀架 1
N310	$TC_CARR10[1] = 1	; X 方向上第 2 旋转轴的分量
N320	$TC_CARR14[1] = 45	; 第 2 根轴的旋转角度
N330	$TC_CARR23[1] = "T"	; Tool Mode 刀具类型
N340	Stopre	
N350	$SC_WEAR_TRANSFORM = 'B101'	
N360	T0 D0 DL=0	
N370	ROT X30	
N380	G90 G1 G17 F10000 X0 Y0 Z0	

N390	T="MillingTool" X0 Y0 Z0 TOWSTD	; X 0.000 Y11.110 Z 0.001
N400	T="MillingTool" X0 Y0 Z0 TOWMCS	; X 0.000 Y10.100 Z 1.011
N410	T="MillingTool" X0 Y0 Z0 TOWWCS	; X 0.000 Y 9.595 Z 0.876
N420	TCARR=1 X0 Y0 Z0	; X 0.000 Y 6.636 Z 8.017
N430	G18 X0 Y0 Z0	; X10.100 Y-0.504 Z 0.876
N440	m30	

上面示例的说明

从程序段 N390 开始按不同的方案分别逼近位置 X0 Y0 Z0 。 所到达的机床位置在程序段中以注释的方式给出。 接着程序进行说明，如何完成这些位置。

N390： 适配器转换 5 （程序段 N90）将长度 L1 转换为长度 L2。 适配器尺寸自身不用服从该转换。 由刀具长度（10）、刀具磨损（1）、总补偿（0.1）和安装补偿（0.01）的和得出 Y 值（L2 在 G17 时） 。 适配器尺寸（0.001）在 Z 上（L1）。

N400: 在程序段 N350 中设置了设定数据中的位 0 和 2 ：

SD42935 $SC_WEAR_TRANSFORM （用于刀具分量的转换）

这表示，由于 TOWMCS 在程序段 N400 中的刀具磨损和安装补偿不服从于适配器转换。 这两个补偿值的和为 1.01。 因此 Z 位置要增加该尺寸而 Y 位置相对于程序段 N390 要减少相应的值。

在 N410 中 TOWWCS 有效。 所以刀具磨损和安装补偿的和在有效的工件坐标系中生效。 在程序段 N370 中激活了绕 X 轴的 30 度旋转。 因此由原来在 Z 方向上补偿值 1.01 得出新的 Z 分量 0.875 （ = 1.01 * cos(30)）以及新的 Y 分量 -0.505 （ = 1.01 * sin(30)）。 如同程序段 N390 中， 由刀具长度、总补偿和安装补偿累加形成的和，得到程序注释中所给出的尺寸。

在 N420 中额外激活了一个可定向刀架。 它执行绕 X 轴旋转 45 度的操作（见 N310 - N330）。 因为刀架的全部偏移矢量等于零，所以不会得出另外的零点偏移。 可定向刀架对由刀具长度、总补偿与安装补偿的和起作用。 生成的部分矢量为 X0 Y7.141 Z7.142。 将程序段 N410 中 WCS 内由刀具磨损与使用补偿所计算出的和累加到上面。

在 N430 中 G18 被激活。 由刀具长度总补偿和适配器尺寸构成的和的分量进行相应的交换。 可定向刀架仍对新的矢量生效（绕 X 轴旋转 45 度）。 此时得到所生成的部分矢量 X10.100 Y0.0071 Z0.0071。 由刀具磨损和安装补偿构成的矢量（X0 Y-0.505 Z0.875）不受平面转换影响。 两个矢量的和可以得出 N430 注释中所给定的尺寸。

19.8 可旋转刀具上的刀沿数据修正

19.8.1 功能

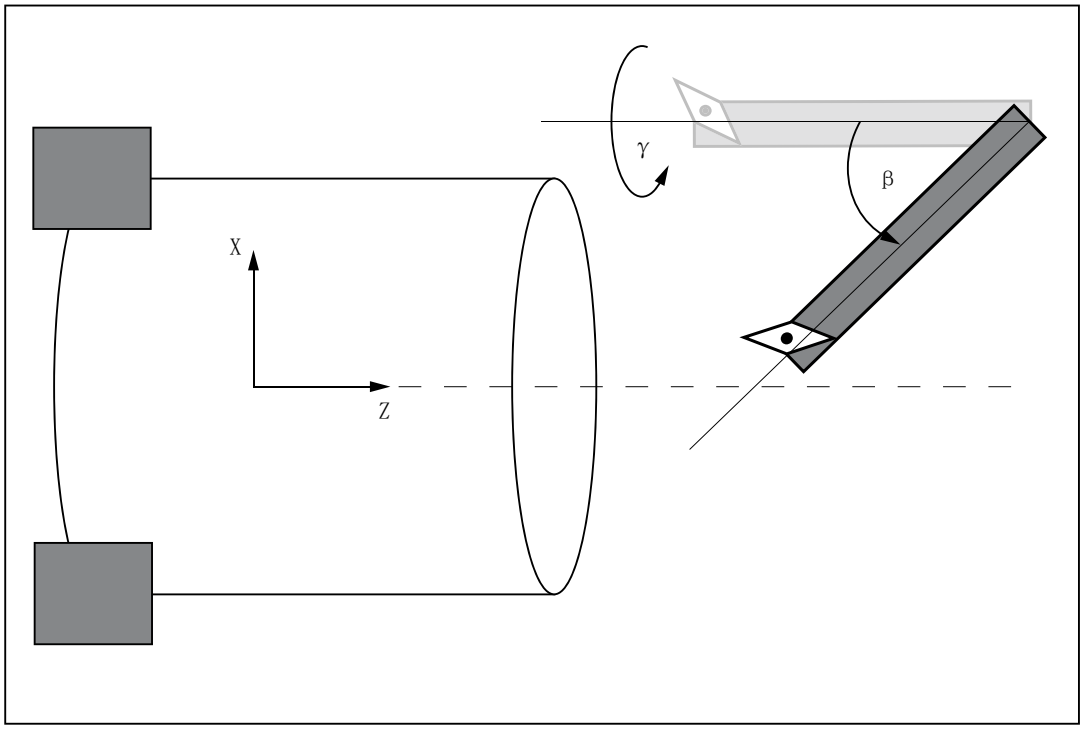
通过“可旋转刀具上的刀沿数据修正”功能可以考虑刀具（主要是车刀，但是也包括钻头和铣刀）旋转时相对于待加工工件所发生的几何数据变化。

19.8.2 计算旋转角度

总是由当前有效的可定向刀架来计算刀具的当前旋转（参见章节“可定向刀架 (页 1678)”）。

此时通常（但不是必须）使用指令 TCOFR 从一个有效的框架来定义可定向刀架的旋转角度。通过这些步骤可以不依赖刀具旋转的实际运动、借助两个角度将刀具方向始终定义为同一类型。

定义刀具旋转要用到两个与机床无关的定位角 β （Beta） 和 γ （Gamma）。 β 指此时绕应用轴的旋转角度（典型为使用 G18 时的 B 轴）而 γ 指绕横坐标的旋转（典型为使用 G18 时的 C 轴）。 首先执行绕 γ 的旋转，接着执行绕 β 的旋转，也就是说 γ 轴通过 β 轴来转动：



19.8.3 使用车刀时的刀沿位置、切削方向和角度

车刀

车刀是指刀具类型（$TC_DP1）值在 500 到 599 范围内的下列刀具。 磨具（刀具类型 400 到 499）与车刀的设置相同。

刀具的处理与刀具类型如车刀无关，当：

SD42950 $SC_TOOL_LENGTH_TYPE = 2 时

刀沿位置和切削方向

车刀通过主刀沿和副刀沿进行限制。 通过这两个刀沿相对于坐标轴的位置来定义刀具参数“刀沿位置”。 下图中概要显示了其对应关系：

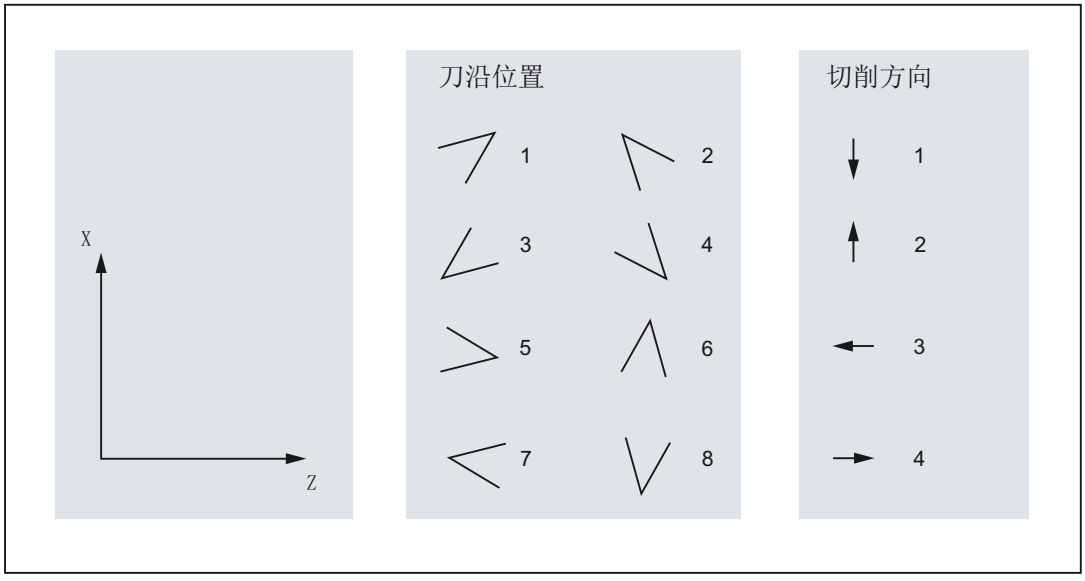


图 19-44 使用车刀时的刀沿位置和切削方向

值 1 到 4 用来标识两个刀沿处于同一象限内的情况，值为 5 到 8 时两个刀沿位于相邻的象限内或者坐标轴处于两个刀沿之间。 刀沿位置被保存在刀具参数 $TC_DP2 中。

可以为每把刀具定义一个切削方向。 该方向被保存在刀具参数 $TC_DP11 中。 可以在 1 到 4 之间取值，分别表示坐标轴的正方向或负方向：

值：	含义：
1	纵坐标 -
2	纵坐标 +

3　　横坐标 -

4　　横坐标 +

每个刀沿位置可以分配两个不同的切削方向：

刀沿位置：	1	2	3	4	5	6	7	8
切削方向：	2, 4	2, 3	1, 3	1, 4	1, 2	3, 4	1, 2	3, 4

主偏角和后角

在下图中显示了用于刀沿位置为 3 的车刀的两个角度（主偏角和后角），需要使用它们来说明刀具刀沿的几何数据。 该示例中的切削方向为 3，即标识为负的 Z 方向（= G18 时的横坐标方向）。

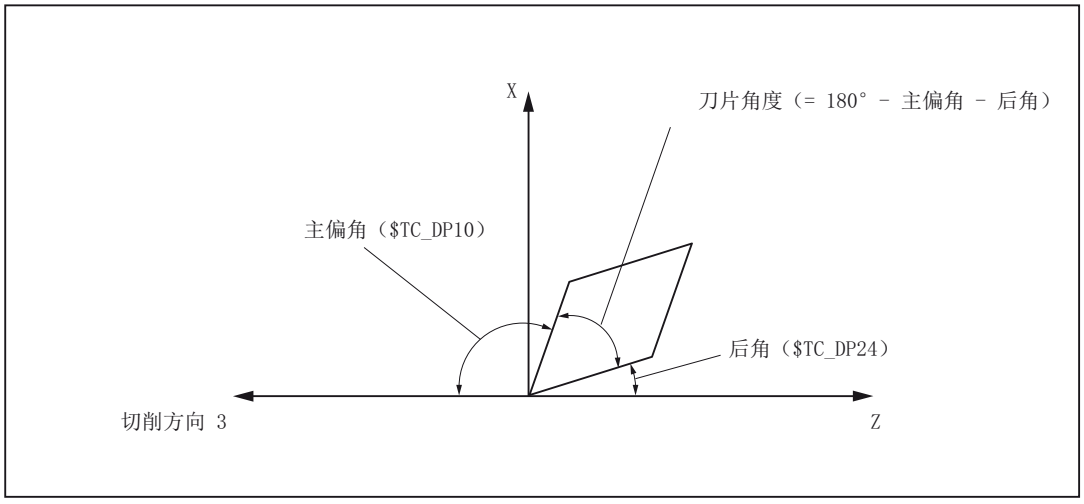


图 19-45　刀沿位置为 3 的车刀上的角度和切削方向

切削方向指明用了于主偏角的参考方向。 后角指逆转的切削方向与相邻刀沿（正）之间测量到的角度。 主偏角和后角被保存在刀具参数 $TC_DP10 或 $TC_DP24 中。

说明

切削方向和刀具角度只有在刀沿位置为 1 到 8 时才有意义。

19.8.4 车刀旋转时的修正

刀具定向

与铣刀相反，车刀不是旋转对称的。 这表示，通常需要 3 个自由度或者三根旋转轴来说明刀具方向时。 因此在所需方向可以完全设置的范围内，与具体的机床运动无关。 必要时可以通过旋转工件坐标系来替代第三个自由度。

说明

由应用程序负责将方向划分为通过可旋转刀架生成的部分和通过旋转坐标达到的部分。 控制器对此不提供更多的功能。

刀沿形状

如果车刀相对于加工平面（即绕着加工平面中的轴，典型情况为 C 轴）旋转了不是 180° 倍数的角度，则（圆形）刀沿在加工平面上的投影会变为椭圆。 如果这样的旋转所产生的偏差与圆形的区别很小，则可以将其忽略（偏转角 < 5°），也就是说控制器总是从圆形的刀沿出发，与刀具方向无关。

这最后还表示，与有效平面相关的控制器接受 180° 的旋转作为初始设定的唯一偏差设置。该限制仅适用于刀沿形状。 在进行任意空间旋转时始终要正确考虑刀具长度。

刀沿位置，切削方向和角度

绕加工平面上的轴旋转 180° 表示，在同一地点使用刀具时主轴的旋转方向必须反向于未旋转的刀具。

当刀具向平面外旋转了 +/- 90°（容差约为 1°）时，切削方向和刀沿位置和刀沿参考点（见下文）一样不进行修正，因为此时未定义刀沿在当前平面中的投影。

如果刀具在平面中旋转（绕着垂直于加工平面的轴或者在 G18 时绕着 Y 轴旋转），则由后角和主偏角所得出的最终角度来计算刀沿位置。 如果未给定用于刀具的这两个角度（即 $TC_DP10 和 $TC_DP24 都为零），则通过旋转角度独自确定新的刀沿位置。 这时的特殊情况是刀沿位置仅以 90° 的步长进行变化，也就是说刀沿位置要么在取值范围 1 到 4 内要么在取值范围 5 到 8 内与初始状态保持相关。如果不允许使用主偏角和后角的预设值（负数值，得出的刀片角度为负或者大于 90°），那么也通过旋转角度来计算新的刀沿位置。在所有这些情况下不修正后角和主偏角。

修正与旋转相关的切削方向，使得最终的后角小于 90° 。 如果原始的切削方向和原始的刀沿位置不匹配，则不修正刀具旋转时的切削方向（参见章节 “使用车刀时的刀沿位置、切削方向和角度 (页 1713)”）。

和通过可旋转刀架计算旋转角度时一样，在系统变量 $P_CUTMOD_ANG 或 $AC_CUTMOD_ANG 中有平面内的旋转角度可供使用。 该角度为没有以 45° 或 90° 倍数进行后续倒圆的原始角度。

极限情况

如果使用车刀时刀沿、切削方向、后角和主偏角的值有效、可以在平面内通过合适的旋转得到所有的切削位置（ 1 到 8 ），在两个刀沿中的一个（主刀沿或副刀沿）离开某一坐标轴的距离小于半个输入增量（输入精度为小数点后 3 位时为 0.0005° ）时，相对于刀沿位置 5 到 8 优先考虑刀沿位置 1 到 4 。

在所有其他情况下（铣刀或无有效刀沿参数的车刀），只能以 90° 步长进行旋转时，适用：如果旋转角度的数值小于 45° + 0.5 个输入增量（输入精度为小数点后 3 位时相当于 45.0005° ），刀沿位置和切削方向保持不变，即这种情况下与旋转小于 45° 时的处理相同。与之类似，当旋转数值与 180° 的偏差小于 45° + 0.5 个输入增量时，处理方式与 135° 到 225° 范围内的旋转相同。

刀沿参考点

使用车刀时一方面要定义刀沿中心点而另一方面要定义刀沿参考点。 此时通过刀沿位置来确定这两个点之间的相互位置。

两点间的距离在刀沿位置 1 到 4 时等于 √2 倍的刀沿半径，在刀沿位置 5 到 8 时等于 1 倍的刀沿半径。 刀沿参考点在第一种情况中相对于刀沿中心点、位于加工平面中的等分角上，在第二种情况中位于坐标轴上。

将刀具绕着垂直于加工平面的轴旋转任意角度，如果刀沿参考点相对于刀具位置固定的话，则会跟着旋转。 这时就无法再满足上述条件（位置在轴或者等分角上）。 这不是期待的结果。 相反总是会修正刀沿参考点，使得刀沿参考点和刀沿中心点之间的距离矢量处于称为 8 的方向上。 必要时必须为此修正刀沿位置。

下图中举例显示了其对应关系：

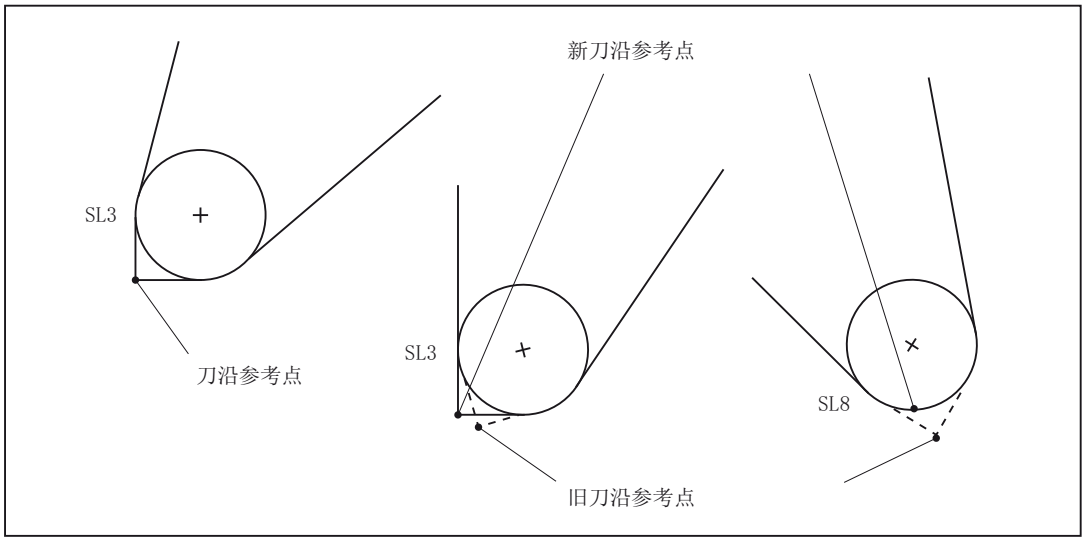


图 19-46　刀具旋转时的刀沿参考点和刀沿位置（SL）

刀沿位置为 3 的刀具，按照后角 22.5° 和 主偏角 112.5° 进行旋转。 旋转至 22.5° 时刀沿位置保持不变，然而要修正刀沿参考点相对于刀具的位置，使这两个点的相对位置保留在加工平面内。 在旋转增加（到 67.5°）时刀沿位置变为数值 8 。

说明

因为可以通过刀具长度矢量来定义刀沿参考点，通过修正刀沿参考点使有效的刀具长度发生改变。

19.8.5　使用铣刀和钻头时的刀沿位置

铣刀和钻头

铣刀和钻头是指刀具类型（$TC_DP1）值在 100 到 299 范围内的下列刀具。

刀具的处理与刀具类型如铣刀和钻头无关，当：

SD42950 $SC_TOOL_LENGTH_TYPE = 1 时

刀沿位置

对于这样定义的铣刀和钻头同样会输入一个刀沿位置，在旋转时按照下列说明对其进行修正。

也许设定刀沿位置的刀具不是上面所定义的铣刀和钻头或车刀，则不使用该刀沿位置。

钻头和铣刀的刀沿位置与在车刀上一样储存在 $TC_DP2 中。 借鉴车刀上刀沿位置的定义可以使用该参数的数值 5 到 8 。 此时刀沿位置指明了刀具的方向（旋转轴的方向）：

刀沿位置	刀具旋转轴的方向
5	横坐标 +
6	纵坐标 +
7	横坐标 -
8	纵坐标 -

示例：

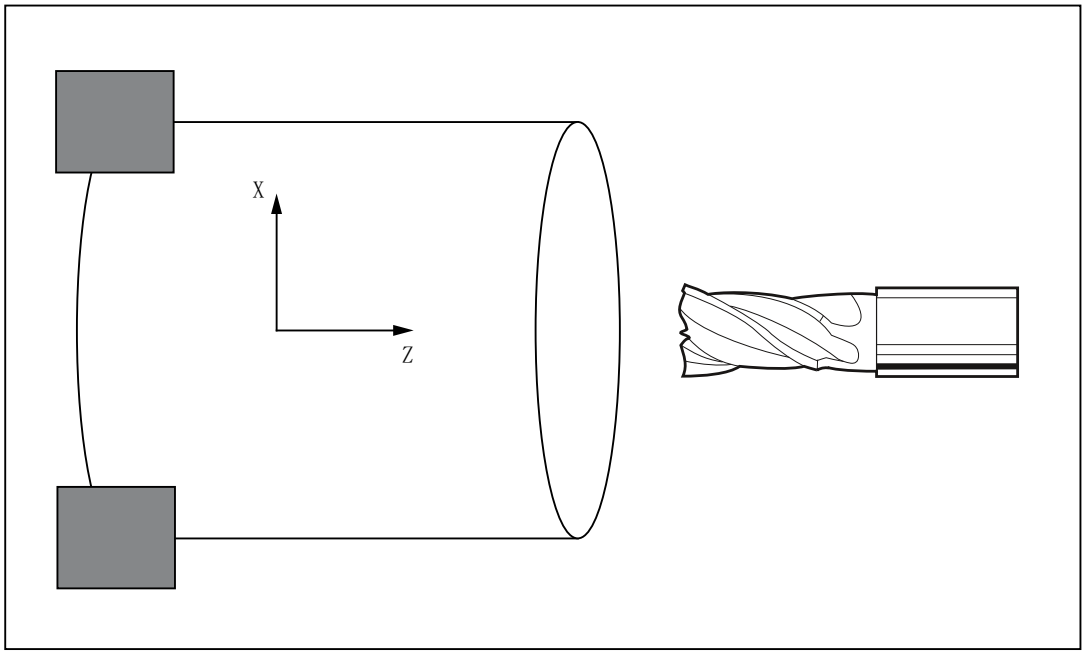


图 19-47　使用刀沿位置 7 的铣刀

19.8.6 铣刀和钻头旋转时的修正

在铣刀和钻头旋转时可以对刀沿位置进行相应的换算。 不用为铣刀和钻头定义切削方向和刀具角度（后角或主偏角），这样可以专门通过旋转推导出刀沿位置的变化。 由此还可以得出，当旋转的数值相对于零位置大于 45° 时，铣刀和钻头上的刀沿位置才发生改变。

19.8.7 参数设置

故障情况中的特性

当“可旋转刀具上的刀沿数据修正”功能有效时（通过显性调用 CUTMOD 或者通过刀具选择）可能会出现不同的故障状态。

对于这种故障状态中的每一个都可以设置故障是否会引发报警，如果引发，那么是只显示这类报警（警告提示）还是会额外中断零件程序的编译。 通过下列机床数据进行设置：

MD20125 $MC_CUTMOD_ERR

机床数据分配给每个故障状态两个位 Bit：

故障状态	位	含义
没有为有效刀具定义有效的切削方向。	0	出现故障“无效切削方向”时输出报警
	1	出现故障“无效切削方向”时程序停止
有效刀具的刀沿角度（后角和主偏角）都为零。	2	出现故障“未定义刀沿角度”时输出报警
	3	出现故障“未定义刀沿角度”时程序中断
有效刀具后角的值不允许使用（< 0° 或者 > 180°）。	4	出现故障“无效后角”时输出报警
	5	出现故障“无效后角”时程序中断
有效刀具主偏角的值不允许使用（< 0° 或者 > 90°）。	6	出现故障“无效主偏角”时输出报警
	7	出现故障“无效主偏角”时程序中断
有效刀具刀片角度的值不允许使用（< 0° 或者 > 90°）。	8	出现故障“无效刀片角度”时输出报警
	9	出现故障“无效刀片角度”时程序中断
刀沿位置与有效刀具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°）。	10	出现故障“无效的刀沿位置与主偏角组合”时输出报警
	11	出现故障“无效的刀沿位置与主偏角组合”时程序中断
禁止的有效刀具旋转（刀具从有效加工平面向外旋转 ± 90°（容差约为 1°））。 因此在加工平面中无法定义刀沿位置。	12	出现故障“无效旋转”时输出报警
	13	出现故障“无效旋转”时程序停止

重新上电后的特性

“可旋转刀具上的刀沿数据修正”功能（CUTMOD）在重新上电后会自动使用机床数据中所保存的值进行初始化：

MD20127 $MC_CUTMOD_INIT

如果该机床数据的值为“-2”，则将 CUTMOD 设置为该值，这可以在机床数据中设置：

MD20126 $MC_TOOL_CARRIER_RESET_VALUE （复位时刀架有效）

19.8.8 编程

“可旋转刀具上的刀沿数据修正”功能可以通过指令 CUTMOD 来激活。

句法

```
CUTMOD=<值>
```

含义

CUTMOD　用于打开功能“可旋转刀具的刀沿数据修改”的指令

<值>　可以为 CUTMOD 指令赋予下列值：

0　取消激活该功能。

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被激活，则该功能也被激活，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。

说明

SD42984 $SC_CUTDIRMOD

指令 CUTMOD 可以激活的功能替代了原来由设定数据 SD42984 $SC_CUTDIRMOD 激活的功能。然而，原先的功能仍被保留。但是同时使用两个功能又没有意义，因此，只有当 CUTMOD 等于零时，它才被激活。

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。

影响激活的工作平面

为了确定修改的刀沿位置，切削方向和主偏角和后角，有必要注意各个激活级面 (G17 - G19) 中的刀沿。

如果设置数据 SD42940 $SC_TOOL_LENGTH_CONST（在级面更换时更换刀具长度分量）还包括一个有效值不等于零（正或负 17、18 或 19），则该级面确定其内容，在该级面中注意相关的尺寸。

系统变量

可提供下列系统变量：

系统变量	含义
$P_CUTMOD_ANG / $AC_CUTMOD_ANG	在激活的加工层面中提供（未倒圆）角，其对于修改刀沿数据（刀沿位置，切削方向，后角和主偏角）在用 CUTMOD 或 $SC_CUTDIRMOD 激活的功能时作为基础。 $P_CUTMOD_ANG 与进给时的当前状态有关， $AC_CUTMOD_ANG 与当前主运行程序段有关。
$P_CUTMOD / $AC_CUTMOD	读取当前有效值，该值最近通过指令 CUTMOD 已编程（要激活刀沿数据修改的刀架号码）。 如果最近编程的 CUTMOD 值 = -2（用当前激活的可定向的刀架激活），则在 $P_CUTMOD 中值不返回 -2，而返回用于编程时间点的激活的可定向刀架号码。 $P_CUTMOD 与进给时的当前状态有关，$AC_CUTMOD 与当前主运行程序段有关。
$P_CUT_INV / $AC_CUT_INV	如果刀具旋转，主轴旋转方向必须反转，则值为真。为此在相关各个读取操作的程序段中必须满足下列 4 个条件： 1. 车刀或铣刀有效 （刀具类型 400 到 599 并且/或者 SD42950 $SC_TOOL_LENGTH_TYPE = 2）。 2. 刀沿影响通过语言指令 CUTMOD 激活。 3. 一个可定向的刀架激活，该刀架通过数值由 CUTMOD 标出。 4. 可定向的刀架使刀具绕着加工级面中的轴旋转（典型方式为 C 轴），得出的正常刀具刀沿相对于输出位置旋转超过 90°（典型方式为 180°）。 如果四个所述条件中至少有一个不满足，则变量内容为假。对于未定义刀沿位置的刀具，变量值总是为假。 $P_CUT_INV 与进给时的当前状态有关，$AC_CUT_INV 与当前主运行程序段有关。

所有主运行变量 ($AC_CUTMOD_ANG, $AC_CUTMOD und $AC_CUT_INV) 可以读取在同步动作中。从进给读入存取生成一个进给停止。

修正过的刀沿数据

如果刀具旋转激活，修改的数据可提供下列系统变量：

系统变量	含义
$P_AD[2]	刀沿位置
$P_AD[10]	主偏角
$P_AD[11]	切削方向
$P_AD[24]	后角

说明

当功能“可旋转刀具的刀沿数据修改”通过指令 CUTMOD 激活且一个可定向的影响刀具旋转的刀架激活时，数据总是根据相应的刀具参数（$TC_DP2[..., ...] 等）修改。

19.8.9 示例

下面的例子采用的是一个带有切削位置 3 的刀具和一个可定向的刀架，该刀架可以使刀具绕着 B 轴旋转。

括号中的数值总是按顺序 X, Y, Z 规定了机床坐标系 (MCS) 中的程序段位置。

```
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
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
```

程序代码	注释		
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			

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，到达经过修改的刀沿位置 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。

19.9 增量编程补偿值

19.9.1 G91 扩展

前提条件

使用 G91 可以定义增量尺寸编程，使得在选择刀具补偿时补偿值运行会加上增量编程值。

应用

在一些应用中比如划痕，要求用增量尺寸，仅运行编程的位移。 没有运行有效刀具补偿。

过程

选择使用增量尺寸编程的刀具补偿

- 用刀尖在工件上对刀。
- 将当前的实际位置减去基准框架中的刀具补偿值 （实际值设置）。
- 从零位置开始增量运行。

激活

在使用 FRAME 和轴的增量编程时是否会超出修改过的刀具长度或者只运行所编程的行程，可以使用设定数据进行设置：

SD42442 $SC_TOOL_OFFSET_INCR_PROG （刀具长度补偿）

零点偏移 / 框架 G91

当值 = 1 时并使用 FRAME 和轴的增量编程时是否会超出标准的零点偏移，或者当值 = 0 时只运行编程值，可以使用设定数据进行设置：

SD42440 $SC_FRAME_OFFSET_INCR_PROG （框架中的零点偏移）

更多信息参见章节“K2：轴、坐标系、框架 (页 765)”。

前提条件

如果设置特性时使得位置在程序结束以及 复位 后仍然有效
MD20110 $MC_RESET_MODE_MASK， 位 6=1 （确定复位/零件程序结束后的控制器初始设置）
并且在第 1 个零件程序段中编程了增量行程，则增加至编程行程总会超出补偿。

说明

在该配置中零件程序总是使用绝对值编程开始。

19.9.2 朝刀具定向运行（MOVT）

典型应用）

在使用可定向刀架的机床上可以在刀具方向上运行（典型情况是在钻削时），而不用激活框架（例如利用 TOFRAME 或 TOROT）用另一根轴来指明刀具方向。

同样在使用有效框架进行斜面加工的机床上,有效框架定义了斜面而刀具不能设定为相之精确垂直，由于使用了指数化刀架（切端面齿）而不能任意设置刀具方向。

在这种情况下 - 与原来所要求的、垂直于平面的运行动作有所偏差时 - 必须在刀具方向钻削，否则钻头不能朝纵向轴方向运行，通常会导致钻头加工中断。

MOVT

此类运动的终点通过 MOVT=... 编写。 缺省设置下，编写的值沿刀具方向增量生效。 此时正方向定义为刀尖指向刀架的的方向。 因此，进给（钻孔）时 MOVT 通常为负值，回退时则为正值。 这与普通的轴向平行加工情况一致，例如使用 G91Z...

以 MOVT=AC(...) 形式编写运动时，MOVT 以绝对值方式生效。 为此定义一个穿过当前零点的平面，其表面一般矢量平行于刀具定向。 MOVT 在此时用来说明与平面相关的位置：

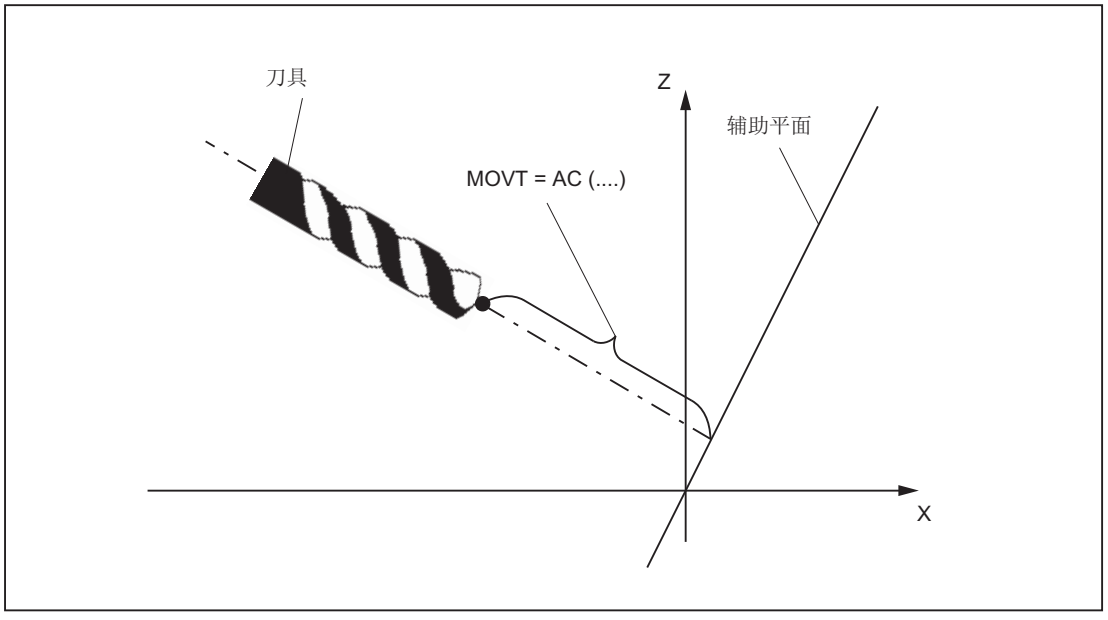


图 19-48 在刀具方向上进行绝对值编程时定义位置

与辅助平面间的参考用于计算终点位置。 生效框架不受这些内部计算影响。

用来代替 MOVT=... 也可以写入 MOVT=IC(...)，用来明确清楚的说明， MOVT 以增量方式生效。 这两种写入格式的功能无区别。

前提条件

使用 MOVT 编程要遵守下列前提条件：

- 编程与是否存在可定向刀架无关。 运动方向取决于生效的平面。 其沿垂直坐标的方向运行，即 G17 中沿 Z 轴方向，G18 中沿 Y 轴方向，G19 中沿 X 轴方向。 这既适用于没有激活可定向刀架的情况，也适用于可定向刀架上刀具不可旋转或者初始设置中使用可旋转刀具的情况。
- 定向转换（3-4-5 轴转换）生效时，MOVT 以相同方式生效。
- 如果在一个程序段中使用 MOVT 同时修改了刀具方向（例如在 5 轴转换有效时通过同时插补回转轴），则以用于 MOVT 运动方向的程序段开始处的方向为准。 刀尖（TCP-Tool Center Point）的轨迹不受方向变化影响。
- 线性或样条插补 （G0， G1， ASPLINE， BSPLINE， CSPLINE） 必须有效。 否则会输出报警。 若有样条插补方式生效，那么得到的轨迹通常非直线，因为由 MOVT 得到的终点作为通过 X、Y、Z 显性编写的终点处理。
- 在包含 MOVT 的程序段中不允许编写几何轴（报警 14157），

19.10 刀具基本定向

19.10.1 功能

缺省情形下，分配给刀具自身的定向仅取决于生效的加工平面。例如，刀具定向在使用 G17 时平行于 Z， 在使用 G18 时平行于 Y，在使用 G19 时平行于 X。

借助“为刀沿分配定向”功能，能够为每个刀沿分配不同于缺省设置的基本定向。为此，该功能提供以下系统变量：

系统变量	含义	数据类型	缺省设置
$TC_DPV[<t>, <d>]	刀沿定向	INT	0
$TC_DPV3[<t>, <d>]	刀沿定向的 L1 分量	REAL	0

系统变量	含义	数据类型	缺省设置
$TC_DPV4[<t>, <d>]	刀沿定向的 L2 分量	REAL	0
$TC_DPV5[<t>, <d>]	刀沿定向的 L3 分量	REAL	0

<t>:刀沿的 T 编号

<d>：刀沿的 D 编号

说明

选择名称 $TC_DPV3 到 $TC_DPV5 时与刀具长度分量的名称 $TC_DP3 到 $TC_DP5 类似。

仅当用于激活该功能（$MN_MM_ENABLE_TOOL_ORIENTATION，参见“参数设置 (页 1731)”）的机床数据不等于零时，这些系统变量才可用。若该机床数据的值为 1，则仅系统变量 $TC_DPV1[<t>, <d>] 可用，若值为 2，则四个系统变量均可用。

如果所有四个系统变量均等于零 ，则通过生效的加工平面定义刀具基本定向。

如果系统变量 $TC_DPV[<t>, <d>] 等于零，则其余三个系统变量（若存在）定义一个方向矢量。此时矢量的值无关紧要。

示例：

$TC_DPV[1, 1] = 0

$TC_DPV3[1, 1] = 1.0

$TC_DPV4[1, 1] = 0.0

$TC_DPV5[1, 1] = 1.0

在此示例中，刀具定向指向 L1-L3 平面中的平分线，即就铣刀而言在平面 G17 生效的情况下，指向 Z-X 平面的平分线。

$TC_DPV[<t>, <d>]

系统变量 $TC_DPV[<t>, <d>] 用于以经简化的方式给定特定的常用刀具基本定向（平行于坐标轴）。下表中列出允许使用的数值。此时第一栏中的值与第二栏或第三栏中的值等效。

$TC_DPV[<t>, <d>]	($TC_DPV2[<t>, <d>], $TC_DPV3[<t>, <d>], $TC_DPV4[<t>, <d>]) 在 $TC_DPV[<t>, <d>] = 0 时	
	铣刀	车刀 [1]
≤ 0 或者 > 6	($TC_DPV5[<t>, <d>], $TC_DPV4[<t>,<d>], $TC_DPV3[<t>, <d>]) [2]	($TC_DPV3[<t>, <d>], $TC_DPV5[<t>, <d>], $TC_DPV4[<t>, <d>]) [2]
1	(0, 0, V)	(0, V, 0)
2	(0, V, 0)	(0, 0, V)
3	(V, 0, 0)	(V, 0, 0)
4	(0, 0, -V)	(0, -V, 0)
5	(0, -V, 0)	(0, 0, -V)
6	(-V, 0, 0)	(-V, 0, 0)

[1] 在此，车刀指所有刀具类型 ($TC_DP1[<t>, <d>]) 为 400 至 599 的刀具。所有其他刀具类型当作铣刀。

[2] 若所有三个值 $TC_DPV3[<t>, <d>]、$TC_DPV4[<t>, <d>]、$TC_DPV5[<t>, <d>] 均等于零，则通过生效的加工平面确定刀具基本定向（缺省设置）。

V:Steht für einen positiven Wert in der entsprechenden Systemvariablen.

示例：

- 铣刀

  $TC_DPV[<t>, <d>] = 3　含义等同于　($TC_DPV3[<t>, <d>] = 0, $TC_DPV4[<t>, <d>] = 0, $TC_DPV5[<t>, <d>] = V)

- 车刀

  $TC_DPV[<t>, <d>] = 2　含义等同于　($TC_DPV3[<t>, <d>] = 0, $TC_DPV4[<t>, <d>] = V, $TC_DPV5[<t>, <d>] = 0)

19.10.2 参数设置

机床数据

激活

仅当以下机床数据不等于零时，用于描述刀具基本定向的系统变量才可用：

MD18114 $MN_MM_ENABLE_TOOL_ORIENT （为刀沿分配定向）

值	含义
0	功能不生效。
1	为刀具<t>的每个刀沿<d>分配系统参数 $TC_DPV[<t>, <d>]，借此便可定义 6 个沿正或负坐标方向的刀具基本定向。
2	除系统参数 $TC_DPV[<t>, <m>] 以外，为刀具<t>的每个刀沿<d>分配另三个系统参数 $TC_DPV3[<t>, <m>]、$TC_DPV4[<t>, <m>] 和 $TC_DPV5[<t>, <m>]，借此便可定义任意的空间刀具基本定向。

设定数据

仅当通过系统变量 $TC_DPVx[<t>, <d>] 中的至少一个定义了刀具基本定向时，下文述及的设定数据才生效。在仅通过平面选择 G17 - G19 确定刀具基本定向的情况下，这些设定数据无作用。

基本定向的平面分配，独立于刀具类型

在 G18 中启用车刀及磨具的基本定向，在 G17 中启用铣刀的基本定向。在这些情形下，生效的刀具定向较 $TC_DPVx[<t>, <d>] 中的设定无变化。此时基本定向总是指垂直于平面的方向，在该平面内可能会进行刀具半径补偿。就车刀而言，刀具定向通常不与刀具纵轴重合。

借助以下设定数据可定义：在基本定向的平面分配中，是否独立于 $TC_DP1 中的记录，像铣刀或者车刀或磨具那样处理刀沿。

SD42950 $SC_TOOL_LENGTH_TYPE

值	含义
0	根据缺省设置进行分配。将车刀及磨具（刀具类型 400 至 599）与其他刀具（铣刀）区分。
1	独立于实际刀具类型，始终像铣刀那样分配。

值	含义
2	独立于实际刀具类型，始终像车刀及磨具那样分配。
3	分别针对车刀及磨具（刀具类型 400 至 599）和针对所有其他刀具（铣刀）进行分配。

其中通过以下设定数据确定基本定向的平面分配：

- 铣刀：
  SD42940 $SC_TOOL_LENGTH_CONST
- 车刀及磨具：
  SD42942 $SC_TOOL_LENGTH_CONST_T

参见“平面切换时的特性”

平面切换时的特性

平面切换会导致定向改变，

此时会执行下列旋转：

平面切换	旋转
G17 → G18	绕 Z 轴旋转 -90 度并接着绕 X 轴旋转 -90 度。
G18 → G19	
G19 → G17	
G17 → G19	绕 X 轴旋转 90 度并接着绕 Z 轴旋转 90 度。
G18 → G17	
G19 → G18	

这些旋转与在平面切换时为交换刀具长度矢量的分量而需执行的旋转相同。

在适配器转换生效时，刀具基本定向会随之旋转。

若以下设定数据不等于零，则刀具基本定向在平面切换时不旋转：

- 铣刀：
  SD42940 $SC_TOOL_LENGTH_CONST
- 车刀及磨具：
  SD42942 $SC_TOOL_LENGTH_CONST_T

19.10.3 示例

定义一铣刀，长度 L1=10 ，其刀具基本定向处于 X-Z 平面的平分线中。

程序代码	注释
N10 $TC_DP1[1,1]=120	; 刀具类型：铣刀
N20 $TC_DP3[1,1]=10	; 长度补偿矢量 L1=10
N30 $TC_DPV[1,1]=0	; N30 bis N60:Programmierung der Werkzeuggrundorientierung.
N40 $TC_DPV3[1,1]=1	
N50 $TC_DPV4[1,1]=0	
N60 $TC_DPV5[1,1]=1	
N70 G17 F1000 X0 Y0 Z0 T1 D1	; 激活刀具并逼近零点。
	; 由于刀具长度而得出机床位置 X0 Y0 Z10。
N80 MOVT=10	; 沿刀具方向增量运行。
	; 产生的轴位置为 X7.071 Y0 Z17.071。
N80 M30	

19.11 刀具补偿的特殊处理

19.11.1 相关的设定数据

SD42900- 42960

通过设定数据 SD42900 - SD42940 可以进行下列与刀具补偿相关的设置：

- 刀具长度的符号
- 磨损的符号
- 在用几何轴进行镜像时磨损分量的特性
- 通过设定数据转换加工平面时磨损分量的特性

- 刀具长度分量的分配与实际的刀具类型无关
- 将磨损分量转换至合适的坐标系中会影响有效的刀具长度

说明

在下列说明中磨损被理解为下列分量的数值总和：

- 磨损量： $TC_DP12 到 $TC_DP20
- 补偿和，组成成分包括：
  - 磨损量： $SCPX3 到 $SCPX11
  - 设置值： $ECPX3 到 $ECPX11

关于补偿和与刀具补偿的详细信息请查阅：

文档:
功能手册之刀具管理

编程手册之基本原理分册；刀具补偿

要求的设定数据

- SD42900 $SC_MIRROR_TOOL_LENGTH （刀具长度分量和基础尺寸分量的镜像）
- SD42910 $SC_MIRROR_TOOL_WEAR （刀具长度分量磨损值的镜像）
- SD42920 $SC_WEAR_SIGN_CUTPOS （磨损分量的符号计算）
- SD42930 $SC_WEAR_SIGN (磨损量尺寸的反向符号)
- SD42940 $SC_TOOL_LENGTH_CONST （分配刀具长度分量到几何轴）
- SD42950 $SC_TOOL_LENGTH_TYPE （刀具长度分量的分配与刀具类型无关）
- SD42935 $SC_WEAR_TRANSFORM （磨损值转换）
- SD42960 $SC_TOOL_TEMP_COMP （刀具长度偏移）

19.11.2 刀具长度镜像 （SD42900 $SC_MIRROR_TOOL_LENGTH）

激活

通过下列设定数据激活刀具长度镜像：

SD42900 $SC_MIRROR_TOOL_LENGTH <> 0 (TRUE) （镜像时刀具长度的符号转换）

功能

通过符号反向可以对下列分量进行镜像：

- 刀具长度： $TC_DP3， $TC_DP4， $TC_DP5
- 基础尺寸： $TC_DP21， $TC_DP22， $TC_DP23

所有相关轴发生镜像的基础尺寸都要进行镜像处理。 磨损值 不一起镜像。

磨损值镜像

磨损值镜像需要使用下列设定数据：

SD42910 $SC_MIRROR_TOOL_WEAR <> 0 （镜像时刀具磨损的符号转换）

相关轴发生镜像的刀具长度分量，通过符号反向对其磨损量进行镜像。

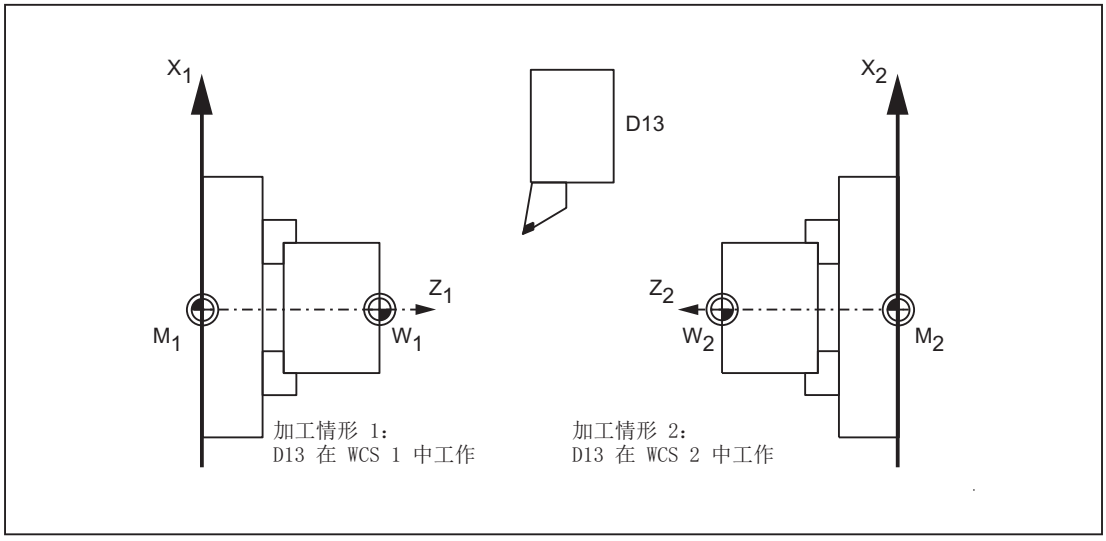


图 19-49 应用示例： 双主轴车床

19.11.3 磨损长度镜像 （SD42920 $SC_WEAR_SIGN_CUTPOS）

激活

可以通过下列数据激活磨损长度镜像：

SD42920 $SC_WEAR_SIGN_CUTPOS <> 0 (TRUE) （使用刀沿位置的刀具上磨损的符号）

功能

刀沿位置	长度 1	长度 2
1	---	---
2	---	反向
3	反向	反向
4	反向	---
5	---	---
6	---	---
7	---	反向
8	反向	---
9	---	---

在没有相关刀沿位置的刀具类型上不能进行磨损长度镜像。

说明

可以通过同时激活下列功能来取消在一个或多个分量中的镜像（符号反向）：

刀具长度镜像 （SD42900 <> 0）

和：

磨损长度镜像 （SD42920 <> 0）

SD42930 $SC_WEAR_SIGN

设定数据不等于零：

所有磨损量尺寸的符号都反向。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。

如果给出一个正值的磨损尺寸，则刀具会变“短”变“薄”。

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。

示例：

```
N10 $SC_WEAR_SIGN = 0                ; 没有磨损值符号反转
```

```
N20 $TC_DP1[1,1]= 120                ; 立铣刀
N30 $TC_DP6[1,1] = 100               ; 刀具半径 100 毫米
N40 $TC_DP15[1,1] = 1                ; 刀具半径的磨损尺寸为 1 毫米，生成的刀具半径为 101 毫米
N100 T1 D1 G41 X150 Y20
....
N150 G40 X300N10
....
N200 $SC_WEAR_SIGN = 1               ; 所有磨损值符号反转，通过重新选择（D1） 使新的、 99 毫
                                       米的半径生效。 没使用 D1 则有效半径仍为 101 毫米。
N300 D1 G41 X350 Y-20
N310 ....
```

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。

19.11.4 刀具长度和平面转换 （SD42940 $SC_TOOL_LENGTH_CONST）

平面切换

当工作平面转换时（G17-G19），刀具长度分量（长度，磨损和基础尺寸）到几何轴的分配没有改变。

刀具的分配

将刀具长度分量分配给车刀和磨具的几何轴（刀具类型 400 到 599）按照下表通过设定数据的值得出：

SD42940 $SC_TOOL_LENGTH_CONST （在平面转换时变换刀具长度分量）

平面	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

平面	长度 1	长度 2	长度 3
*) 每个不为 0 的值，又不等同于所列出的六个值中的一个，则作为 18 求值。			

下表说明了在使用 **所有其他刀具** （刀具类型 < 400 或 > 599）时刀具长度分量到几何轴的分配：

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
*) 每个不为 0 的值，又不等同于所列出的六个值中的一个，则作为 17 求值。			

说明

表中的说明基于几何轴 1 到 3 用 X、Y、Z 命名。 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。

三个刀具长度分量可以设置为上述 6 种不同的类型。

19.11.5 刀具类型 (SD42950 $SC_TOOL_LENGTH_TYPE)

特性

确定将刀具长度分量（长度、磨损和基础尺寸）分配到几何轴与刀具类型无关。

设定数据**不等于**零： （按照标准来确定）

会区分车刀、磨具（刀具类型 400 至 599）以及其它的刀具（铣刀）。

可以采用值 0 到 2 。其它的数值将视为数值 0 处理。

刀具长度分量的分配不考虑实际的刀具类型：

- 在值为 1 时： 始终与使用铣刀时一样
- 在值为 2 时： 始终与使用车刀时一样

可定向刀架

设定数据 SD42900 - SD42950

设定数据 SD42900 － SD42950 对一个可能有效的可定向刀架分量不起作用。 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。 在计算合成的总长度时要考虑所有由设定数据引起的变动。

说明

在使用可定向刀架时经常要求在没有镜像的基础系统中定义所有的刀具，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）。

当机床类型在物理学上不允许旋转带不同方向的、固定安装的刀具时，使用可定向刀架就非常有意义了。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。

19.11.6 考虑方向时工件坐标系中的刀具长度

更换刀具或者工作平面

所显示的刀具值与工件坐标系中的大小一致。 如果要使用带有斜置刀具挂架的刀架，则需要注意，所进行的转换能够支持该刀架。 如果不是这种情况，就会显示错误的刀具尺寸。 将工作平面从 G17 转换到 G18 或 G19 时同样要进行确认，转换也能够用于该工作平面。 如果涉及到一个只能用于 G17 加工的转换，则在平面转换时还会继续显示 Z 方向上的刀具尺寸。

在转换关闭时会根据工作平面在 X、 Y 或 Z 方向上显示基本刀具。 会考虑已编程的刀架。 该刀具尺寸在无转换的运行中不发生变化。

19.11.7 刀具方向上的刀具长度偏移

实时温度补偿

在使用活动刀具的 5 轴机床上加工头可能会出现温度变化, 这一方面会引起膨胀变化而且还会以长度延展的形式被传导至刀具主轴上。 5 轴加工头的典型情形，例如主轴纵向轴方向上的热膨胀。

当存在刀具定向时也可以对热膨胀进行补偿，方法是不再将温度补偿值分配给加工轴而是分配给刀具。这样就可以在刀具方向变化时对长度变化进行补偿。

借助与当前刀具方向相关的方向转换，可以实现动作的实时叠加与同时跟随旋转。此时补偿值一直在刀具坐标系中使用。

只有当待补偿轴也实际回参考点时，温度补偿才有效。

激活

可以将下列机床数据的值设置为 **不等于零** 来激活刀具方向上的温度补偿：

MD20390 $MC_TOOL_TEMP_COMP_ON（激活刀具长度的温度补偿）

另外必须在机床数据中为每根相关的通道轴设置位 2 ：

MD32750 $MA_TEMP_COMP_TYPE[<轴索引>]（温度补偿类型）

作为几何轴转换或转换切换的结果可能暂时会有不止三根通道轴变为几何轴，此时可能会超过三根轴。如果该位没有被设置用于指定的通道轴，则在该轴中有没补偿值可以生效。然而其他的轴不受此影响。在这种情况下不会输出报警。

作用范围

刀具方向上的温度补偿只作用于进行类 5 轴转换，转换时使用：

- 转换类型 24
  两根轴用来旋转刀具。
- 转换类型 56
  一根轴用来旋转刀具，另一根轴用来旋转无温度补偿的工件。

在类 5 轴转换时，使用：

- 转换类型 40
  在工件可旋转时刀具方向恒定，在这种情况下机床回转轴的运动不会对温度补偿的方向造成影响。

此外在方向转换时（不是类 5 轴转换）刀具方向上的温度补偿与同样起作用：

- 转换类型 64 到 69
  可旋转线性轴。

说明

在所有其他转换类型时可以激活温度补偿。但它不受刀具方向的变化的影响。轴就好象没有激活使用了温度补偿的方向转换一样运行。

极限值

通过机床数据可以限制补偿的最大值：

MD20392 $MC_TOOL_TEMP_COMP_LIMIT[0...2]（最大的刀具长度温度补偿）

极限值的预设值为 1 毫米。如果预设的温度补偿值大于该极限值，会对其进行限制但不发出报警。

SD42960

三个温度补偿值构成一个补偿矢量并被包含在设定数据中：

SD42960 $SC_TOOL_TEMP_COMP[0...2]（与刀具相关的温度补偿）

可以借助同步动作或者由用户通过 PLC 来说明设定数据。因此补偿值也能够被用于其他的补偿目的。

在初始设置中或者当关闭了方向转换时全部三个补偿值在三根几何轴的方向上起作用（典型的顺序 X、 Y、 Z）。将分量分配到几何轴与刀具类型（车刀/铣刀或磨具）以及所选择的加工平面 G17 到 G19 无关。设定数据中值的变化会立即生效。

可定向刀架

如果一个可定向刀架有效，那么温度补偿矢量在方向变化时会跟随旋转。这与可能有效的方向转换无关。

如果一个可定向刀架和类 5 轴转换或者使用可旋转线性轴的转换共同有效，则温度补偿矢量要服从这两种旋转。

说明

使用可旋转线性轴的转换虽然会考虑刀具矢量（长度）的变化，但是 不 考虑由可定向刀架所引起的方向变化。

温度补偿值直接随着有效发生的方向变化而改变。特别是在打开或者关闭方向转换时，也同样适用。

在将几何轴分配到通道轴出现变化时也同样适用。例如一根轴在转换切换之后不再是几何轴，则它的温度补偿值会（插补式）被减少为零。反过来，如果一根轴在转换之后成为了几何轴，则会立即建立该轴上可能存在的温度补偿值。

示例

刀具方向上的温度补偿

这是一台使用可旋转刀具的 5 轴机床，刀具能够绕 C 轴和 B 轴旋转。

在初始设置中刀具平行于 Z 轴。将 B 轴旋转 90 度，则刀具指向 X 方向。

因此在下列设定数据中，当转换有效时温度补偿值同样在机床 X 轴的方向上生效：

SD42960 $SC_TOOL_TEMP_COMP[2]（与刀具相关的温度补偿）

如果在该设置中关闭了转换，那么刀具方向会按照定义重新与 Z 轴平行并因此与实际的方向生产偏差。所以要撤销 X 轴方向上的温度偏移并同时在 Z 方向上建立温度偏移。

这是一台使用可旋转刀具的 5 轴机床（转换类型 24）。主要的机床数据显示如下：

- 第一根回转轴绕 Z C 轴旋转。
- 第二根回转轴绕 Y B 轴旋转。

下表中列出了主要的机床数据：

机床数据	值	注释
MD20390 $MC_TOOL_TEMP_COMP_ON	= TRUE	温度补偿有效
MD32750 $MA_TEMP_COMP_TYPE[AX1]	= 4	刀具方向上的补偿
MD32750 $MA_TEMP_COMP_TYPE[AX2]	= 4	刀具方向上的补偿
MD32750 $MA_TEMP_COMP_TYPE[AX3]	= 4	刀具方向上的补偿
		转换类型 24 的分配：
MD24100 $MC_TRAFO_TYPE_1	= 24	转换类型 24 ， 在第 1 个通道
MD24110 $MC_TRAFO_AXES_IN_1[0]	= 1	转换的第 1 根轴
MD24110 $MC_TRAFO_AXES_IN_1[1]	= 2	转换的第 2 根轴
MD24110 $MC_TRAFO_AXES_IN_1[2]	= 3	转换的第 3 根轴
MD24110 $MC_TRAFO_AXES_IN_1[3]	= 5	转换的第 5 根轴

机床数据	值	注释
MD24110 $MC_TRAFO_AXES_IN_1[4]	= 4	转换的第 4 根轴
MD24120 $MC_TRAFO_GEOAX_ASSIGN_TAB_1[0]	= 1	几何轴到通道轴 1
MD24120 $MC_TRAFO_GEOAX_ASSIGN_TAB_1[1]	= 2	几何轴到通道轴 2
MD24120 $MC_TRAFO_GEOAX_ASSIGN_TAB_1[2]	= 3	几何轴到通道轴 3
MD24570 $MC_TRAFO5_AXIS1_1[0]	= 0.0	
MD24570 $MC_TRAFO5_AXIS1_1[1]	= 0.0	方向
MD24570 $MC_TRAFO5_AXIS1_1[2]	= 1.0	第 1 根回转轴平行于 Z
MD24572 $MC_TRAFO5_AXIS1_2[0]	= 0.0	方向
MD24572 $MC_TRAFO5_AXIS1_2[1]	= 1.0	第 2 根回转轴平行于 Y
MD24572 $MC_TRAFO5_AXIS1_2[2]	= 0.0	
MD25574 $MC_TRAFO5_BASE_ORIENT_1[0]	= 0.0	
MD25574 $MC_TRAFO5_BASE_ORIENT_1[1]	= 0.0	刀具基本定向
MD25574 $MC_TRAFO5_BASE_ORIENT_1[2]	= 1.0	在 Z 方向上

NC 程序中的温度补偿值

分配给 X 和 Z 这两根轴的补偿值不等于零，并且在温度补偿时要根据刀具长度来考虑该值。将每次所达到的加工轴位置作为注释在程序行里进行说明。

程序代码	注释
$SC_TOOL_TEMP_COMP[0] = -0.3	; X 上的补偿值
$SC_TOOL_TEMP_COMP[1] = 0.0	
$SC_TOOL_TEMP_COMP[2] = -1.0	; Z 上的补偿值
	; 机床轴的设定位置
N10 G74 X0 Y0 Z0 A0 B0	; X Y Z
N20 X20 Y20 Z20 F10000	; 20.30 20.00 21.00
N30 TRAORI()	; 20.30 20.00 21.00

程序代码	注释
N40 X10 Y10 Z10 B90	; 11.00 10.00 9.70
N50 TRAFOOF	; 10.30 10.00 11.00
N60 X0 Y0 Z0 B0 C0	; 0.30 0.00 1.00
N70 M30	

通过例外程序段 N40 温度补偿始终在原始方向上生效，因为刀具指向初始方向。这在程序段 N50 中也适用。刀具实际还指向 X 轴方向，因为 B 轴仍然位于 90 度。但是由于转换已经关闭，所以需要考虑的方向重又平行于 Z 。

机床数据	值	注释
MD20390 $MC_TOOL_TEMP_COMP_ON	= TRUE	温度补偿有效
MD32750 $MA_TEMP_COMP_TYPE[AX1]	= 4	刀具方向上的补偿
MD32750 $MA_TEMP_COMP_TYPE[AX2]	= 4	刀具方向上的补偿
MD32750 $MA_TEMP_COMP_TYPE[AX3]	= 4	刀具方向上的补偿

其他文档

更多关于“温度补偿”的说明参见：

文档：
功能手册之扩展功能分册；补偿 (K3)

关于“类 5 轴转换”的信息参见：

文档：
功能手册之特殊功能分册； 3 至 5 轴转换 （F2）

19.12 总补偿与设置补偿

19.12.1 简介

总补偿

可以将总补偿视作加工中的 **可编程过程补偿** ，它由所有与额定尺寸形成工件偏差的误差值（包括磨损）组成。

总补偿是磨损的一种普遍形式。它是刀具刀沿数据的组成部分。总补偿的参数取决于刀沿的几何数据。

总补偿的补偿数据通过 **DL 号码** 进行响应（**DL**：Location dependend 相关位置；与各自使用位置相关的补偿）。

相反 D 号码的磨损值说明了刀沿的物理磨损情况，也就是说在特殊情况下总补偿可能与刀沿的磨损一致。

通常都能够使用总补偿； 即使用有效或无效刀具管理时以及使用扁平 D 号码功能时。

通过设置相应的机床数据来分配总补偿：

- 精细总补偿
- 粗略总补偿 （设置补偿）

设置补偿

设置补偿指调试人员在加工之前进行调试的补偿。这些值保存在 NC 专设的存储器中。然后操作员通过 HMI 只能使用“精细总补偿”。

“精细总补偿”和“粗略总补偿”在 NC 内部叠加。该值在下面会被称为总补偿。

说明

可以通过机床数据设置来打开这个功能：

MD18080 $MN_MM_TOOL_MANAGEMENT_MASK, Bit 8=1 （用于刀具管理的逐级存储器保留）

在运动有效时（例如 5 轴转换），计算刀具长度首先要按所说明的方式考虑不同的磨损分量。然后将所有的刀具长度输入到转换中。故磨损值总是跟随转换，与第 56 组的 G 指令无关（不同于可定向刀架）。

19.12.2 功能性说明

总补偿

每个 D 号码可以定义多个总补偿（DL 号码）。此时会采集例如 **与工件位置有关** 的补偿值并赋值给刀沿。总补偿生效与磨损一样（也就是说累加到 D 号码的补偿值上）。数据被固定分配给一个 D 号码。

设置

通过机床数据可以进行下列设置：

- 激活总补偿
- 确定 NC 中待创建 DL 数据组的最大数量
- 确定分配给一个 D 号码的 DL 号码的最大数量
- 确定在数据备份时是否应当一起备份总补偿（精细/粗略）
- 确定应当激活哪种总补偿，当：
  - 激活一个新的刀沿补偿时
  - 发生操作面板 复位 时
  - 发生操作面板 启动 时
  - 到达程序末尾时

此时名称以刀具和刀沿相应机床数据的逻辑为基准。

“设置补偿”和“精细总补偿”可以通过系统变量和相应的 BTSS 服务来读写。

说明

在刀具管理有效时可以通过机床数据来确定，编程换刀时所激活的刀具总补偿是否保持不变或者应当设置为零。

补偿参数 $TC_DPx 的总结

到目前为止定义了下列常用的系统变量用来说明刀沿：

$TC_DP1	刀具类型
$TC_DP2	刀沿位置

用于几何数据和磨损的参数

系统变量 $TC_DP3 到 $TC_DP11 分配有刀具几何数据补偿值。系统变量 $TC_DP12 到 $TC_DP20 允许为其中每一个参数命名一个磨损量。

几何尺寸	磨损	长度补偿
$TC_DP3	$TC_DP12	长度 1
$TC_DP4	$TC_DP13	长度 2
$TC_DP5	$TC_DP14	长度 3
几何尺寸	磨损	半径补偿

几何尺寸	磨损	长度补偿
$TC_DP6	$TC_DP15	半径
$TC_DP7	$TC_DP16	拐角半径 （刀具类型 700； 槽锯）
几何尺寸	磨损	其他补偿
$TC_DP8	$TC_DP17	长度 4 （刀具类型 700； 槽锯）
$TC_DP9	$TC_DP18	长度 5
$TC_DP10	$TC_DP19	角度 1 （刀具端面与环面之间的角度）
$TC_DP11	$TC_DP20	角度 2 （刀具纵向轴与环面上端之间的角度）

基础尺寸或适配器尺寸

$TC_DP21	适配器 - 长度 1
$TC_DP22	适配器 - 长度 2
$TC_DP23	适配器 - 长度 3

工艺

系统变量	后角
$TC_DP24	• 保存后角用于 ManualTurn； 刀具类型 5xx。与车刀标准循环中的含义相同。 • 保存钻头的刀尖角用于 ShopMill； 刀具类型 2xx。 • 在车刀的标准循环中使用；刀具类型 5xx。在使用这些刀具时为副刀沿上的角度。
$TC_DP25	• 保存切削速度的值用于 ManualTurn 。 • 保存 1xx 和 2xx 类型刀具不同状态的位编码值用于 ShopMill 。

总补偿和设置补偿的参数（$TC_SCPxy-， $TC_ECPxy）

参数的编号以系统变量 $TC_DP3 到 $TC_DP11 的编号为准。

参数的生效方式与磨损类似（累加到刀具几何数据上）。每个刀沿参数最多可以定义六个总量/安装参数。

刀具几何数据参数，补偿值累加到该参数上。	总量/安装参数 - 长度补偿	刀具磨损参数
$TC_DP3	长度 1 $TC_SCP**1**3, $TC_SCP**2**3,$TC_SCP**3**3, $TC_SCP**4**3,$TC_SCP**5**3,$TC_SCP**6**3 $TC_ECP**1**3, $TC_ECP**2**3,$TC_ECP**3**3, $TC_ECP**4**3,$TC_ECP**5**3,$TC_ECP**6**3 粗体号码 1, 2, ... 6 用来将最多六个（与位置相关或诸如此类）使用 DL =1, ... 6 进行编程补偿的参数标识给第一栏中所命名的参数。	$TC_DP12
$TC_DP4	长度 2 $TC_SCP**1**4, $TC_SCP**2**4,$TC_SCP**3**4, $TC_SCP**4**4,$TC_SCP**5**4,$TC_SCP**6**4 $TC_ECP**1**4, $TC_ECP**2**4,$TC_ECP**3**4, $TC_ECP**4**4,$TC_ECP**5**4,$TC_ECP**6**4	$TC_DP13
$TC_DP5	长度 3 等等 ...	$TC_DP14
	半径补偿	
$TC_DP6	半径	$TC_DP15
$TC_DP7	倒圆半径	$TC_DP16
	其他补偿	
$TC_DP8	长度 4	$TC_DP17
$TC_DP9	长度 5	$TC_DP18

刀具几何数据参数，补偿值累加到该参数上。	总量/安装参数 - 长度补偿	刀具磨损参数
$TC_DP10	角 1 ...等等	$TC_DP19
$TC_DP11	角 2 $TC_SCP**2**1, $TC_SCP**3**1,$TC_SCP**4**1, $TC_SCP**5**1,$TC_SCP**6**1,$TC_SCP**7**1 $TC_ECP**2**1, $TC_ECP**3**1,$TC_ECP**4**1, $TC_ECP**5**1,$TC_ECP**6**1,$TC_ECP**7**1 粗体号码 2, 3, ... 7 用来将最多六个（与位置相关或诸如此类）使用 DL =1, ... 6 进行编程补偿的参数标识给第一栏中所命名的参数。	$TC_DP20

前提条件

可以通过机床数据确定一个刀沿 DL 数据组的最大数量和 NC 中总补偿的总数。按照标准该值等于零，也就是说不能编程总补偿。

使用激活的“监控功能”可以对刀具的磨损或者也可以对“总补偿”进行监控。

额外的总量/安装数据组要另外占用缓冲存储器。每个参数需要 8 个字节。

一个总补偿数据组需要：8 字节 * 9 参数 = 72 字节

一个设置补偿数据组同样需要这么多字节。另外还需要一些字节用于内部的管理数据。

19.12.3 激活

功能

该功能必须通过机床数据来激活：

MD18108 $MN_MM_NUM_SUMCORR （TO 范围内的总补偿）

设置补偿和总补偿（“精细”）通过系统变量 $TC_ECPx 和 $TC_SCPx 或者通过 BTSS 接口进行定义，可以在零件程序中激活。

通过编程语言指令 DL="nr" 来完成。

激活新的 D 号码要么编程新的 DL 号码，要么通过下列机床数据所确定的 DL 号码有效：

MD20272 $MC_SUMCORR_DEFAULT （未编程总补偿初始设置）

DL 编程

总补偿的编程总是与有效的 D 号码相关并使用指令：

```
DL = "n"
```

总补偿“n”被累加到有效 D 号码的磨损上。

说明

如果使用“设置补偿”和“精细总补偿”进行加工，会将两者合在一起然后累加到刀具的磨损上。

可以使用指令取消总补偿：

```
DL = 0
```

说明

DL0 不允许使用。取消补偿（D0 和 T0） 时总补偿同样无效。

编程一个不存在的总补偿会释放报警，类似于编程一个不存在的 D 补偿。

此时只有定义过的磨损为补偿的组成部分（在系统变量 $TC_DP12 到 $TC_DP20 中定义）。

如果在 D 补偿有效时编程一个总补偿（也包括取消时，在轨迹上产生的效果与对 D 编程时相同。这时刚才有效的半径补偿会失去到相邻程序段的参考。

配置

MD18112 $MN_MM_KIND_OF_SUMCORR, 位 4=0：（TO 范围内总补偿的属性） 标准设置：

每个 DL 号码只存在 一个 总补偿数据组。

它通常由总补偿来说明。

这时表示通过 $TC_SCPx 所显示的数据。

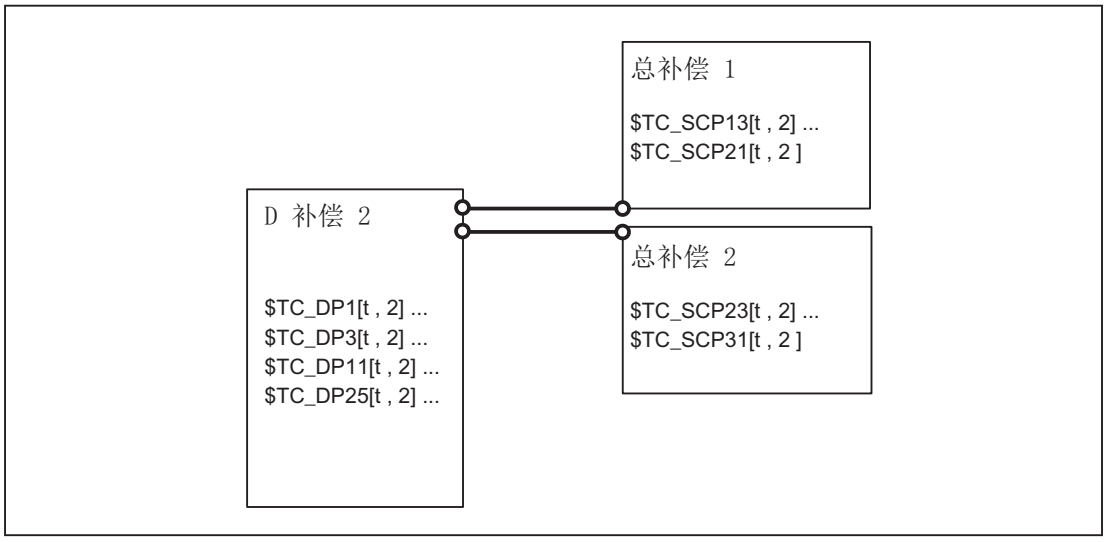


图 19-50　MD18112 $MN_MM_KIND_OF_SUMCORR, 位 4 = 0

刀具使用 T = t 有效。使用图中的数据进行编程：

D2	;	切削补偿； 即 $TC_DP3,... $TC_DP11 + 磨损 ($TC_DP12,...$TC_DP20) + 适配器尺寸
...		
DL=1	;	另外在先前 D2 的补偿上加上总补偿 1 ， 即 $TC_SCP13,...$TC_SCP21
...		
DL=2	;	D2 的补偿上不再累加总补偿 1 而是总补偿 2 ， 即 $TC_SCP23,...$TC_SCP31
...		
DL=0	;	取消总补偿； 只有 D2 的数据仍然有效

MD18112 $MN_MM_KIND_OF_SUMCORR, 位 4=1:设置补偿可供使用

此时总补偿由“精细总补偿”（通过 $TC_SCPx 显示）和设置补偿（通过 $TC_ECPx 显示）构成。因此一个 DL 号码就对应两个数据组。总补偿由相关的分量累加得出（$TC_ECPx + $TC_SCPx）。

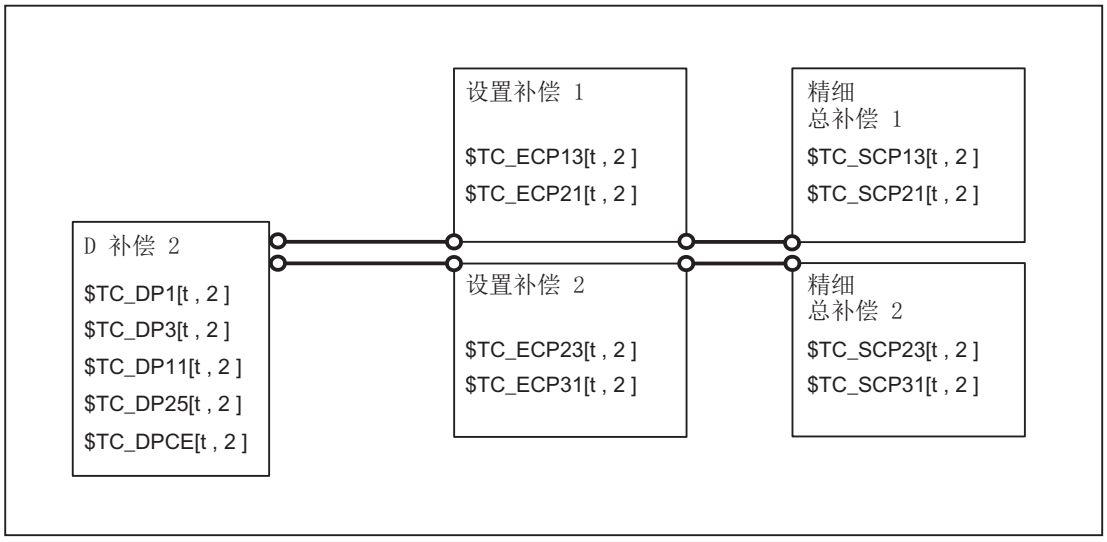


图 19-51　MD18112 $MN_MM_KIND_OF_SUMCORR, 位 4 = 1 “设置补偿” + “精细总补偿”

刀具使用 T = t 有效。使用图中的数据进行编程：

D2	;	切削补偿； 即 $TC_DP3,... $TC_DP11 + 磨损 ($TC_DP12,...$TC_DP20) + 适配器尺寸
...		
DL=1	;	另外在先前 D2 的补偿上加上总补偿 1 ； 即 $TC_ECP13 + $TC_SCP13,...$TC_ECP21 + $TC_SCP21
...		
DL=2	;	D2 的补偿上不再累加总补偿 1 而是总补偿 2 ； 即 $TC_ECP23 + $TC_SCP23,...$TC_ECP31 + $TC_SCP31
...		
DL=0	;	取消总补偿。只有 D2 的数据仍然有效。

在零件程序中读/写

单个总补偿数据组通过系统变量 $TC_SCP 的编号范围进行区分。

单个变量的含义类似于几何数据变量 $TC_DP3 到 $TC_DP11。只设置了长度 1、长度 2、长度 3 用于初始功能 （变量 $TC_SCP13 - $TC_SCP15 用于刀沿的第一个总补偿）。

R5 = $TC_SCP13[t, d]	;	将 R 参数的值设置为刀沿（d）总补偿 1 第一个分量的值 刀具 （t）。
R6 = $TC_SCP21[t, d]	;	将 R 参数的值设置为刀具（t）刀沿（d）总补偿 1 最后一个分量的值。
R50 = $TC_SCP23[t, d]	;	将 R 参数的值设置为刀具（t）刀沿（d）总补偿 2 第一个分量的值。
$TC_SCP43[t, d] = 1.234	;	将刀具（t）刀沿（d）总补偿 4 第一个分量的值设置为 1.234。

用于设置补偿的说明（如果此时 NCK 进行了配置）与之类似，即：

R5 = $TC_ECP13[t, d]	;	将 R 参数的值设置为刀具（t）刀沿（d）设置补偿 1 第一个分量的值。
R6 = $TC_ECP21[t, d]	;	将 R 参数的值设置为刀具（t）刀沿（d）设置补偿 1 最后一个分量的值。
等		

使用设置补偿工作时通过系统变量 $TC_SCPx 来说明“精细总补偿”。

新建 - 总补偿

如果补偿数据组（x）还不存在，则使用第一个写操作在其中一个参数（y）创建数据组。

$TC_SCPxy[t, d] = r.r	;	总补偿 x 的参数 y 获得数值“r.r”。x 其他参数的值为零。

使用设置补偿工作时通过系统变量 $TC_SCPx 来说明“精细总补偿”。

说明

使用设置补偿工作时，假如那时还不存在关于 [t, d] 的数据组，则在为“精细总补偿”创建一个数据组同时建立一个相应的设置补偿数据组。

新建 - 设置补偿

如果补偿数据组（x）还不存在，则使用第一个写操作在其中一个参数（y）创建数据组。

$TC_ECPxy[t, d] = r.r	;	设置补偿 x 的参数 y 获得数值“r.r”。x 其他参数的值为零。

说明

使用设置补偿工作时，假如那时还不存在关于 [t, d] 的数据组，则在为设置补偿创建一个数据组同时建立一个相应的“精细总补偿”数据组。

DELDL - 删除总补偿

大多数情况下，只有当使用一个刀沿在确定的时间、确定的工件位置上进行加工时总补偿才有意义。使用 NC 语言指令 DELDL 可以从刀沿中删除总补偿（释放必需的存储器）。

status = DELDL(t, d)	;	删除刀具 t 刀沿 d 的所有总补偿
	;	t， d 为可选参数

如果未给定 d，则会删除刀具 t 全部刀沿的所有总补偿。

如果 d 和 t 都未给定，则会删除 TO 单元的所有刀具全部刀沿的所有总补偿（用于编程该指令的通道）。

如果使用设置补偿进行工作，使用 DELDL 指令不但会删除相关刀沿的设置补偿，而且会删除它（们）的“精细总补偿”。

说明

在“删除”时会释放用于数据组的存储器。
接下来不能再激活或编程被删除的总补偿。

不能删除有效刀具的总补偿和设置补偿（类似于 D 补偿或刀具数据的清除特性）。

“status 状态”返回值用来显示删除指令的结果：

0：	删除成功
-1：	未删除（一个刀沿）或者未完全删除（多个刀沿）

数据备份

在常用的刀具数据备份框架中备份数据（作为 D 号码数据组的组成部分）。

总补偿的数据备份非常有意义，因为必须要能够在出现严重问题时对瞬间状态进行备份。通过机床数据可以将总补偿排除在数据备份之外（可以对“设置补偿”和“精细总补偿”分别进行设置）。

说明

与程序段搜索和重新定位相关的总补偿特性类似于 D 补偿。可以通过机床数据来确定复位和上电时的特性。

如果通过下列机床设定，在上电之后应当激活最后有效的刀具补偿号码（D），然而最后有效的 DL 号码不再有效：

MD20110 $MC_RESET_MODE_MASK （确定复位/零件程序结束后的控制器初始设定）

19.12.4 示例

示例 1

没有补偿也没有总补偿有效，应当在换刀时通过机床数据来确定：

- MD20270 $MC_CUTTING_EDGE_DEFAULT=0 （未编程的刀具刀沿初始设置）
- MD20272 $MC_SUMCORR_DEFAULT=0 （未编程的总补偿初始设置）

```
T5 M06              ; 换入编号 5 的刀具 - 没有补偿有效
D1 DL=3             ; 补偿 D1 + D1 的总补偿 3 生效
X10
DL=2                ; 补偿 D1 + 总补偿 2 生效
X20
DL=0                ; 取消总补偿，只有补偿 D1 仍然有效
D2                  ; 补偿 D2 生效 - 补偿没有总补偿的份额
X1
DL=1                ; 补偿 D2 + 总补偿 1 生效
X2
```

```
D0                  ; 取消补偿
X3
DL=2                ; 没有影响 - D0 的 DL2 为零（类似于编程 T0 D2）
```

示例 2

在换刀时应当确定，补偿 D2 和总补偿 DL=1 有效，通过机床数据：

MD20270 $MC_CUTTING_EDGE_DEFAULT=2 （未编程的刀具刀沿初始设置）

MD20272 $MC_SUMCORR_DEFAULT=1 （未编程总补偿初始设置）

```
T5 M06              ; 换入编号 5 的刀具 - D2 + DL=1 有效 （=机床数据的值）
D1 DL=3             ; 补偿 D1 + D1 的总补偿 3 生效
X10
DL=2                ; 补偿 D1 + 总补偿 2 生效
X20
DL=0                ; 取消总补偿，只有补偿 D1 仍然有效
D2                  ; 补偿 D2 生效 - 激活总补偿 DL=1
X1
DL=2                ; 补偿 D2 + 总补偿 2 生效
D1                  ; 补偿 D1 + 总补偿 1 生效
```

19.12.5 刀具长度测定的扩展

19.12.5.1 计算使用位置和工件专用的补偿

有效刀具长度的组成

进行没有有效运动转换的刀具补偿时，有效的刀具长度由最多 8 个矢量组成：

- 刀具长度（几何数据） ($TC_DP3 - $TC_DP5)
- 磨损 ($TC_DP12 - $TC_DP14)
- 基础尺寸（参见说明） ($TC_DP21 - $TC_DP23)
- 适配器尺寸（参见说明） ($TC_ADPT1 - $TC_ADPT3)
- 精补偿总和 ($TC_SCPx3 - $TC_SCPx5)
- 粗略总补偿或设置补偿 ($TC_ECPx3 - $TC_ECPx5)
- 可定向刀架的偏移矢量 l_1 ($TC_CARR1 - $TC_CARR3)

- 可定向刀架的偏移矢量 l_2 ($TC_CARR4 - $TC_CARR6)
- 可定向刀架的偏移矢量 l_3 ($TC_CARR15 - $TC_CARR17)

说明

基础尺寸和适配器尺寸只能二选一有效。

单个矢量的生效方式

此时单个矢量或矢量组的生效方式与下列其他尺寸有关：

影响量	作用
G 指令	生效的加工平面
刀具类型	铣刀或车刀或者磨具
机床数据	刀具管理有效/无效，存在/不存在可定向刀架
设定数据	镜像或平面转换时刀具长度分量的特性
可定向刀架	可定向刀架的设定值
适配器转换	转换过的刀具补偿值

分配到几何轴分量

如何将参加矢量的部分和的矢量分量分配到三根几何轴分量上，由下列尺寸决定：

影响量	关联性
有效的加工平面： G17 X/Y 方向 G18 Z/X 方向 G19 Y/Z 方向	进给平面： Z Y X
刀具类型： 铣刀、钻头、磨具、车刀	参见章节“刀具参数 1：刀具类型 (页 1621)”，表格“最少的必要刀具参数”

影响量	关联性
SD42900 $SC_MIRROR_TOOL_LENGTH SD42910 $SC_MIRROR_TOOL_WEAR SD42920 $SC_WEAR_SIGN_CUTPOS SD42930 $SC_WEAR_SIGN SD42940 $SC_TOOL_LENGTH_CONST SD42950 $SC_TOOL_LENGTH_TYPE	参见章节“刀具补偿的特殊处理 (页 1733)”和“设定数据 (页 1798)”。
适配器转换	**文档:** 功能手册之刀具管理

得出的刀具方向始终与三根轴的方向 X、 Y 或 Z 平行并且仅仅与有效的加工平面 G17-G19 相关，因为到目前为止没有为刀具分配方向。

刀具方向的无级变化

可定向刀具借助偏移矢量 I_1 - I_3 能够进行刀具方向的无级变化，用来继续偏移或者改变长度。

更多说明参见章节“可定向刀架 (页 1678)”。

较小的操作员补偿

与之相反，较小的操作员补偿必须能够在普通的生产运行中进行修改。

其原因有例如：

- 刀具磨损
- 夹紧故障
- 机床的温度变化

此时按下列方式定义该补偿：

定义	磨损分量
磨损	$TC_DP12 - $TC_DP14,
精补偿总和	$TC_SCPx3 - $TC_SCPx5,
粗略总补偿或设置补偿	$TC_ECPx3 - $TC_ECPx5

特别是涉及刀具长度计算的补偿，要记录到也能被测量的坐标中。

借助 G 功能组 56，借助三个值 TOWSTD、TOWMCS 和 TOWWCS 以及设定数据，能够更简单地进行这些工件专用补偿：

SD42935 $SC_WEAR_TRANSFORM （刀具分量转换）

SD42935

磨损分量中的哪些：

- 磨损 （$TC_DP12 - $TC_DP14）：
- 设置补偿或粗略总补偿（$TC_ECPx3 - $TC_ECPx5）
- 总补偿精 ($TC_SCPx3 - $TC_SCPx5)

在转换时：

- 适配器转换
- 可定向刀架

应当或者不应当转换，可以通过设定数据来确定：

SD42935 $SC_WEAR_TRANSFORM （磨损值转换）

在设定数据的初始设置中会转换所有的磨损值。

使用下列功能时要考虑到设定数据：

- 机床坐标系中的磨损值
  零件程序指令：TOWMCS
- 工件坐标系中的磨损值
  零件程序指令：TOWWCS

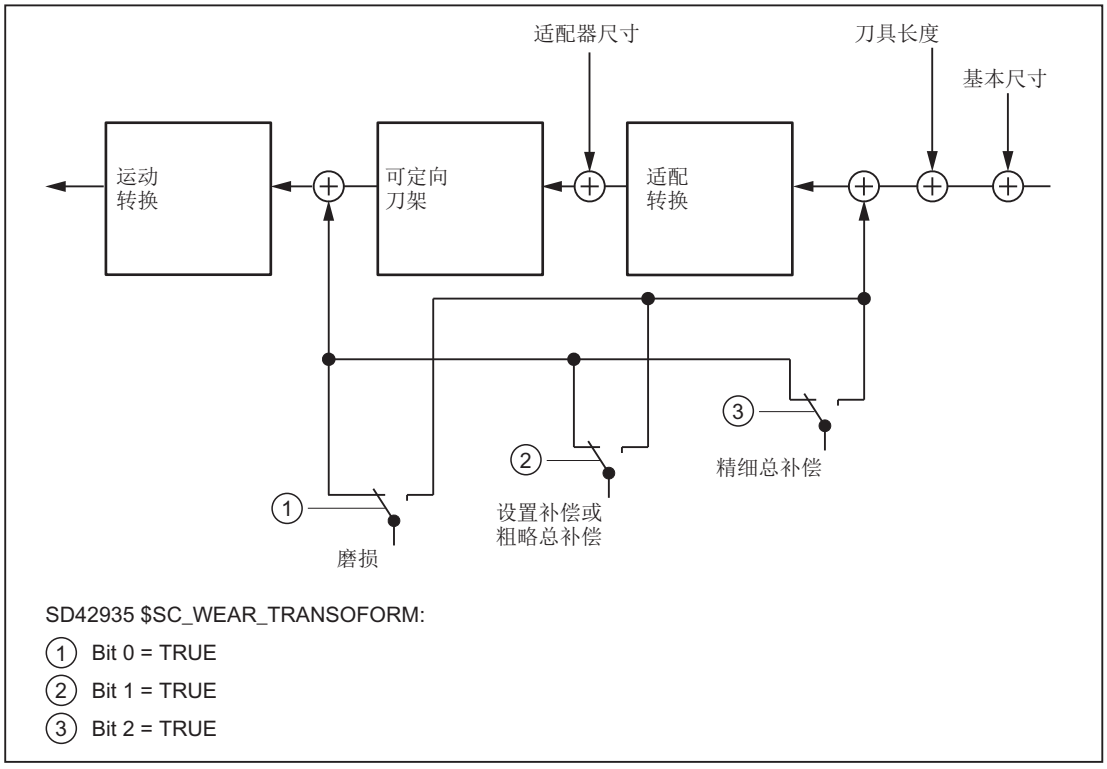


图 19-52　磨损数据的转换取决于 SD42935

编程

使用 G 代码组 56 能够确定下列数值：

句法	更正
TOWSTD	刀具长度中偏移的初始设定值
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	刀具坐标系（TCS）中刀架夹持装置（T 刀架参考点）上的磨损值
TOWKCS	V 刀具坐标系中切削头运动转换（KCS）的磨损值

用于刀具长度偏移的坐标系

使用 G 指令 TOWMCS、 TOWWCS、 TOWBCS、 TOWTCS 和 TOWKCS 例如可以在五个不同坐标系中测量刀具长度分量磨损。

1. 机床坐标系	MCS
1. 基准坐标系	BCS
1. 工件坐标系	WCS
1. 运动转换的刀具坐标系	KCS
1. 刀具坐标系	TCS

可以使用预定义功能 GETTCOR (页 1771) 在这些坐标系中的一个中显示并读取计算出的刀具长度或者刀具长度分量。

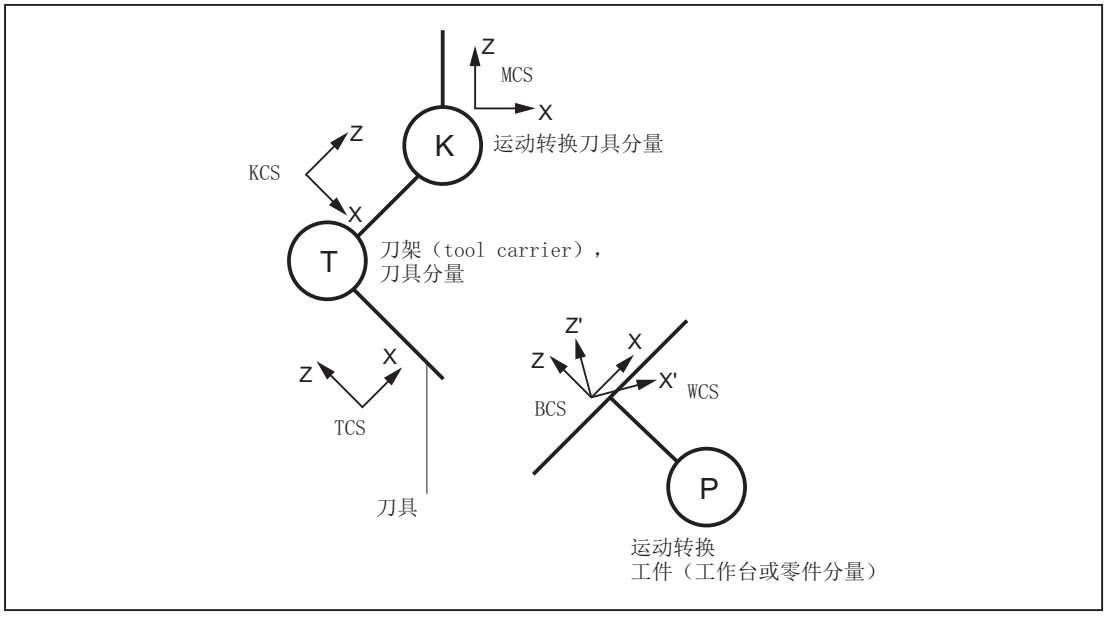


图 19-53　计算刀具长度时的坐标系

19.12.5.2　单个磨损值的功能性

TOWSTD

初始设定值（标准处理）：

- 将磨损值累加到其余的刀具长度分量上。
  然后由此生成的刀具总长度在必要时会参与进一步的计算。

对可定向刀架有效的情况：

- 磨损值服从于相应的旋转。

TOWMCS

机床坐标系中的磨损值（MCS）：

对于通过可定向刀架进行有效旋转的情况：

- 刀架仅旋转所生成刀具长度的矢量，不考虑磨损。
  接着将这样旋转过的刀具长度矢量和磨损量相加。磨损不服从于旋转。

如果 **没有** 有效的可定向刀架或者刀架不引起旋转，则 TOWMCS 和 TOWSTD 相同。

线性转换

如果 MCS 是从 BCS 中通过一个线性平移而产生的，才能在 MCS 中明确定义刀具长度。

同样有可能是这种情况：

- 当没有运动转换有效时
- 或者方向转换（3、 4 和 5 轴转换）有效时。

TOWWCS

工件坐标系中的磨损值（WCS）：

- 使用有效的可定向刀架时计算刀具矢量不用考虑磨损，如同使用 TOWMCS 。
- 磨损量在工件坐标系中进行说明。

将工件坐标系中的磨损矢量换算到机床坐标系中并加到刀具矢量上。

TOWBCS

基本坐标系中的磨损值（BCS）：

- 使用有效的可定向刀架时计算刀具矢量不用考虑磨损，如同使用 TOWMCS 。
- 磨损量在工件坐标系中进行说明。

将基本坐标系中的磨损矢量换算到工件坐标系中并加到刀具矢量上。

非线性转换

I 如果例如使用 TRANSMIT 的非线性转换有效并且将机床坐标系 MCS 给定为所需的坐标系，则会在 MCS 的位置上自动使用 BCS 。

可定向刀架

与运动转换的平台（或部件）部分相反，可定向刀架可能存在的平台部分不直接进入坐标系中。通过这样一个部分进行说明的旋转会映像在基础或者系统框架中并借此一起从 WCS 过渡到 BCS 中。

运动转换

运动转换的平台（或部件）部分通过从 BCS 到 MCS 的过渡进行说明。

TOWTCS

刀具坐标系中的磨损值 （TCS）：

- 使用有效的可定向刀架时计算刀具矢量不用考虑磨损，如同使用 TOWMCS 。
- 磨损量在刀具坐标系 TCS 中进行说明。

将刀具坐标系（TCS）中的磨损矢量通过运动转换的刀具坐标系（KCS） 换算到机床坐标系中并加到刀具矢量上。

TOWKCS

运动转换的磨损值数据在相应的刀具坐标系（KCS）中进行说明。

将磨损矢量通过运动转换的刀具坐标系换算到机床坐标系中并加到刀具矢量上。

刀具生效时的 G 指令更换

更换 G 功能组中的 G 指令 TOWSTD、 TOWMCS、 TOWWCS、 TOWBCS、 TOWTCS 和 TOWKCS 不会影响已生效的刀具，而是在下一次选择刀具时才生效。

如果在选择刀具的同一个程序段中编写，那么该 G 功能组的新 G 指令也生效。

计算单个磨损分量

各个磨损量分量（分配到几何轴，符号赋值）的赋值受以下影响：

- 有效的平面
- 适配器转换
- 下列表格中所列出的五个设定数据

设定数据	磨损分量		
SD42910 $SC_MIRROW_TOOL_WEAR	TOWSTD	TOWMCS	TOWWCS
SD42920 $SC_WEAR_SIGN_CUTPOS	X	X	—

设定数据	磨损分量		
SD42930 $SC_WEAR_SIGN	X	—	—
SD42940 $SC_TOOL_LENGTH_CONST	X	X	X
SD42950 $SC_TOOL_LENGTH_TYPE	X	X	X

说明

需要服从于通过适配器转换的有效旋转或者服从于可定向刀架的磨损分量被标识为不转换的磨损分量。

特殊性

如果 TOWMCS 或者 TOWWCS 有效，则下列设定数据不影响不转换的磨损分量：

SD42920 $SC_WEAR_SIGN_CUTPOS （在使用刀沿位置的刀具上磨损的符号）

另外在使用 TOWWCS 时下列设定数据也不影响不转换的磨损分量：

SD42910 $SC_MIRROR_TOOL_WEAR （镜像时的刀具磨损符号变换）

在这种情况下，引起磨损分量计算的框架中已经包含了可能有效的镜像。

在平面切换时会保存不转换的磨损分量到几何轴的分配，即其不像其余长度分量一样要进行交换。分量的分配与选择刀具时的有效平面相关。

示例

已知一把铣刀，上面只有分配给长度 L1 的磨损值 $TC_DP12 不等于零。

如果 G17 有效，则该长度在 Z 轴方向生效。

在选择刀具之后进行平面转换时该值也始终在 Z 方向上有效，如果 TOWMCS 或 TOWWCS 有效并且设置了下列设定数据的位 1 ：

SD42935 $SC_WEAR_TRANSFORM （用于刀具分量的转换）

如果在选择刀具时例如 G18 有效，则在其他条件相同时该分量反而会始终在 Y 方向上有效。

19.13 使用刀具环境工作

功能一览

- 保存刀具环境 (TOOLENV) (页 1765)
- 删除刀具环境 (DELTOOLENV) (页 1768)
- 读取 T 号、D 号和 DL 号 (GETTENV) (页 1769)
- 读取刀具长度或刀具长度分量 (GETTCOR) (页 1771)
- 修改刀具分量 (SETTCOR) (页 1778)

系统变量一览

- 读取保存的刀具环境的信息 ($P_TOOLENVN, ($P_TOOLENV) (页 1770)

19.13.1 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。

19.13.2 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	

19.13.3 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。	

19.13.4 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）”			

19.13.5 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。

19.13.6 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用于第 1 个分量 (Val_1) 而第二个字符用于第 2 个分量 (Val_2)，按照后续参数 **<CorComp>** 和 **<CorMode>** 对其进行加工。		
			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>：	给定几何轴的索引，在该索引中测量补偿值 **<CorVal>**[0]（可选）	
		数据类型：	INT
		值域：	0 ... 2
		索引 0 ... 2 与在当前刀具环境的生效平面 (G17/G18/G19) 中的横坐标、纵坐标、竖坐标有关。 该参数的内容仅在参数 **<CorComp>** 的值为 “0” 时评估。	
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索引为 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 中对索引为 “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：

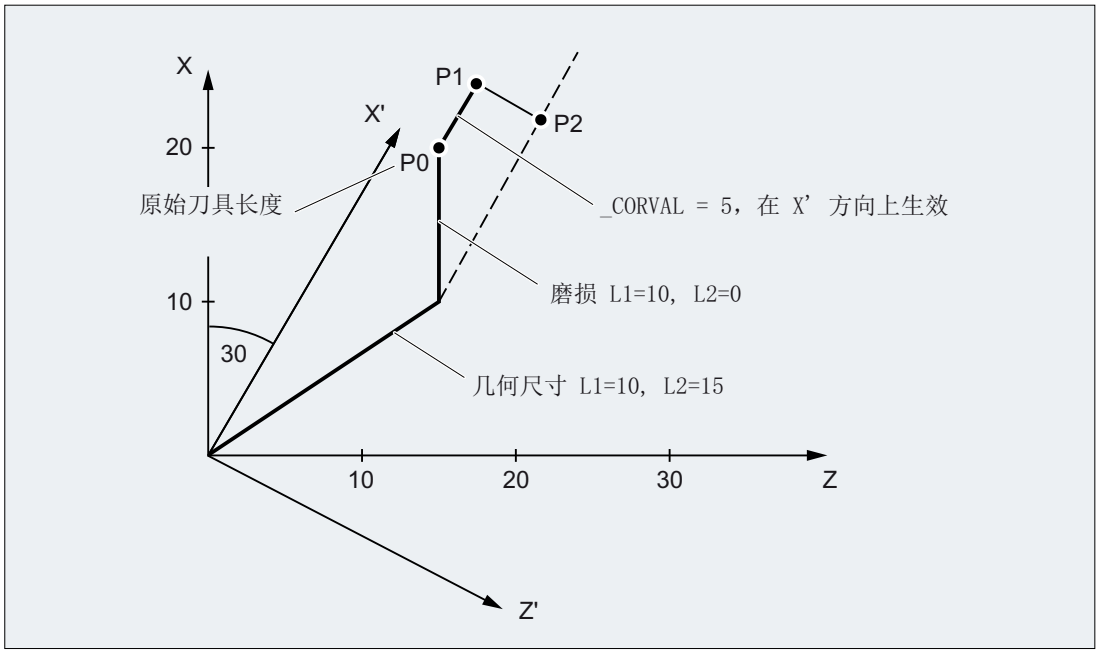
$TC_DP3[1,1]：21.830；几何数据 L1

$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	

程序代码	注释
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
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	

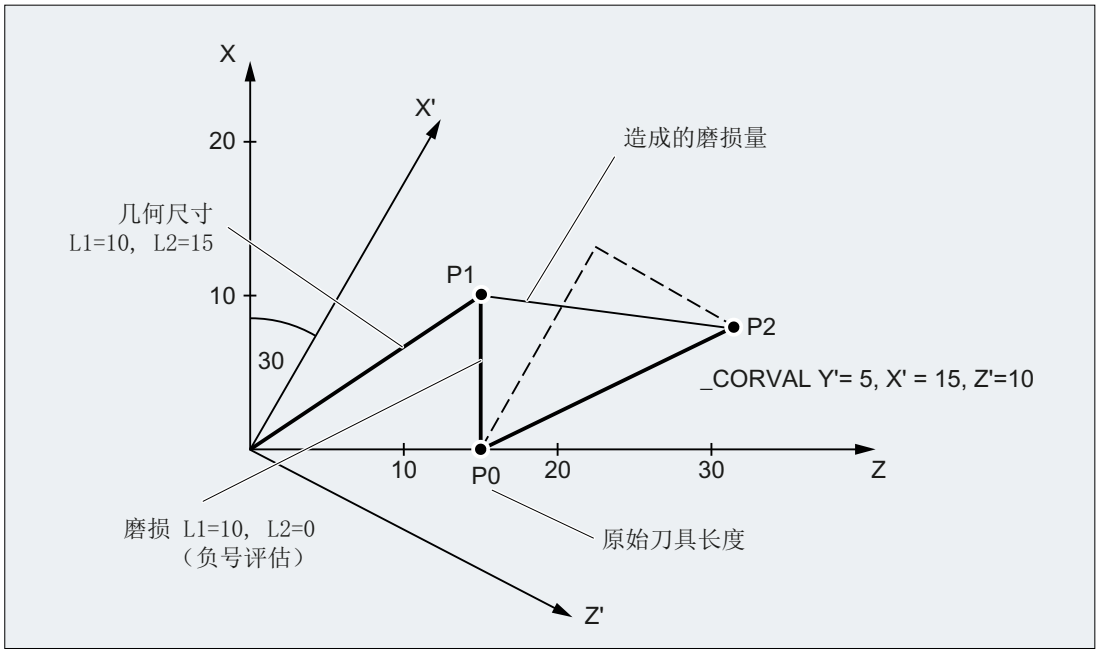
与示例 6 相比，此处参数 <CorComp> = 3，因此可省去对参数 <GeoAx> 的设定。_CORVAL[0] 中包含的值现在直接对刀具分量 L1 生效，N80 中的旋转对结果无影响，$TC_DP12 中的磨损分量与 _CORVAL[0] 一同接收至几何数据分量，这样一来 $TC_DP13 使得刀具总长度在 N100 中首次调用 SETTCOR 后即已处于刀具的几何分量中。

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	

程序代码	注释
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，也就是说磨损必须以负号评估。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，磨损必须以负号评估，因此得到的补偿必须以符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
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。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（在直径轴中刀具长度必须以系数 0.5 评估）

X 为直径轴

程序代码	注释
N10 DEF REAL _LEN[11]	
N20 DEF REAL _CORVAL[3]	
N30 $TC_DP1[1,1]=500	
N40 $TC_DP2[1,1]=2	
N50 $TC_DP3[1,1]=3.	
N60 $TC_DP4[1,1]=4.	
N70 $TC_DP5[1,1]=5.	
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

程序代码	注释
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。

19.14 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 或者 -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 至几何轴 1 到 3 的映像可以通过第 2 个参数**<Matrix>** 进行完整描述。		
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：		实数
		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：

$$<\text{Matrix}> = \begin{pmatrix} 0 & \sin 60° & \cos 60° \\ 0 & \cos 60° & -\sin 60° \\ 1 & 0 & 0 \end{pmatrix}$$

19.15 前提条件

19.15.1 平面 D 号结构

程序段搜索

可以通过以下参数设置在程序段搜索结束时将辅助功能输出到 PLC：

- MD22080 $MC_AUXFU_PREDEF_SPEC
- MD22035 $MC_AUXFU_ASSIGN_SPEC
- MD11110 $MN_AUXFU_GROUP_SPEC

19.15.2 SD42935 扩展

转换磨损量

哪些与 TOWMCS（机床坐标系中的磨损值）和 TOWWCS（工件坐标系中的磨损值）有关的磨损分量应当进行转换或不应当进行转换，可以通过以下数据来确定：

SD42935 $SC_WEAR_TRANSFORM （磨损值转换）

19.15.3 对刀

对刀

如果在复位状态下使用“对刀”功能，在初始设定值方面需要注意下列事项：

- 为评估磨损分量，G 组 TOWSTD、TOWMCS 和 TOWWCS 的初始设定值有决定性作用。
- 若为正确计算而需要不同于初始设定值的值，则仅允许在停止状态下进行对刀。

19.16 数据表

19.16.1 机床数据

19.16.1.1 NC 专用机床数据

编号	名称：$MN_	说明
18082	MM_NUM_TOOL	NC 可管理的刀具的数量（SRAM）
18088	MM_NUM_TOOL_CARRIER	定义刀架的最大数量
18094	MM_NUM_CC_TDA_PARAM	刀具数据数量（SRAM）
18096	MM_NUM_CC_TOA_PARAM	用于编译循环的每个刀沿的数据数量（SRAM）
18100	MM_NUM_CUTTING_EDGES_IN_TOA	TOA 区域中的刀具补偿（SRAM）
18102	MM_TYPE_OF_CUTTING_EDGE	D 号编程方式（SRAM）
18105	MM_MAX_CUTTING_EDGE_NO	D 号最大值
18106	MM_MAX_CUTTING_EDGE_PERTOOL	每把刀具的最大 D 号数
18108	MM_NUM_SUMCORR	NC 中所有总补偿的数量
18110	MM_MAX_SUMCORR_PER_CUTTEDGE	每个刀沿的总补偿数量
18112	MM_KIND_OF_SUMCORR	TO 区域中的总补偿特性（SRAM）
18114	MM_ENABLE_TOOL_ORIENT	指定刀具刀沿定向
18116	MM_NUM_TOOL_ENV	TO 区域中的刀具环境（SRAM）

19.16.1.2 通道专用机床数据

编号	名称：$MC_	说明
20096	T_M_ADDRESS_EXT_IS_SPINO	T、M 换刀时地址扩展的含义
20110	RESET_MODE_MASK	确定复位/零件程序结束后控制系统的初始设置
20120	TOOL_RESET_VALUE	启动（复位/零件程序结束）时的刀具长度补偿
20121	TOOL_PRESEL_RESET_VALUE	复位时的预选刀具
20125	CUTMOD_ERR	用于 CUTMOD 功能的故障处理
20126	TOOL_CARRIER_RESET_VALUE	复位时生效的刀架
20127	CUTMOD_INIT	上电时初始化 CUTMOD
20130	CUTTING_EDGE_RESET_VALUE	启动（复位/零件程序结束）时的刀沿长度补偿
20132	SUMCORR_RESET_VALUE	复位时生效的总补偿
20140	TRAFO_RESET_VALUE	启动（复位/零件程序结束）中的转换数据组
20180	TOCARR_ROT_ANGLE_INCR[i]	可定向刀架的回转轴增量
20182	TOCARR_ROT_ANGLE_OFFSET[i]	可定向刀架的回转轴偏移
20184	TOCARR_BASE_FRAME_NUMBER	用于接收工作台偏移的基本框架的编号
20188	TOCARR_FINE_LIM_LIN	线性精偏 TCARR 的限制
20190	TOCARR_FINE_LIM_ROT	回转精偏 TCARR 的限制
20202	WAB_MAXNUM_DUMMY_BLOCKS	WAB 中无运行时的最大程序段数量
20204	WAB_CLEARANCE_TOLERANCE	WAB 中的方向反转
20210	CUTCOM_CORNER_LIMIT	TRC 中平衡程序段的最大角度
20220	CUTCOM_MAX_DISC	DISC 的最大值
20230	CUTCOM_CURVE_INSERT_LIMIT	TRC 中交点计算的最大值
20240	CUTCOM_MAXNUM_CHECK_BLOCKS	TRC 中预读轮廓计算的程序段
20250	CUTCOM_MAXNUM_DUMMY_BLOCKS	TRC 中不含运行指令程序段的最大数量
20252	CUTCOM_MAXNUM_SUPPR_BLOCKS	含轮廓抑制的程序段的最大数量
20256	CUTCOM_INTERS_POLY_ENABLE	使能用于多项式的交点程序
20270	CUTTING_EDGE_DEFAULT	未作设定时刀沿的初始设置
20272	SUMCORR_DEFAULT	未作设定时总补偿的初始设置
20360	TOOL_PARAMETER_DEF_MASK	刀具参数的定义
20390	TOOL_TEMP_COMP_ON	激活用于刀具长度的温度补偿
20392	TOOL_TEMP_COMP_LIMIT	用于刀具长度的最大温度补偿

编号	名称：$MC_	说明
20610	ADD_MOVE_ACCEL_RESERVE	为叠加运动保留的加速度余量
21080	CUTCOM_PARALLEL_ORI_LIMIT	3D 刀具半径补偿中的最小角度（轨迹切线和刀具定向）
22530	TOCARR_CHANGE_M_CODE	刀架切换时的 M 代码
22550	TOOL_CHANGE_MODE	使用 M 功能时的新刀具补偿
22560	TOOL_CHANGE_M_CODE	用于换刀的 M 功能
22562	TOOL_CHANGE_ERROR_MODE	换刀中出现故障时的特性
24558	TRAFO5_JOINT_OFFSET_PART_1	平台中的运动学偏移矢量，转换 1
24658	TRAFO5_JOINT_OFFSET_PART_2	平台中的运动学偏移矢量，转换 2
28085	MM_LINK_TOA_UNIT	TO 单元对通道的指定（SRAM）

19.16.1.3 进给轴/主轴专用机床数据

编号	名称：$MA_	说明
32750	TEMP_COMP_TYPE	温度补偿类型

19.16.2 设定数据

19.16.2.1 通道专用设定数据

编号	名称：$SC_	说明
42442	TOOL_OFFSET_INCR_PROG	刀具长度补偿
42470	CRIT_SPLINE_ANGLE	压缩器的棱角临界角度
42480	STOP_CUTCOM_STOPRE	刀具半径补偿和预处理停止时的报警响应
42494	CUTCOM_ACT_DEACT_CTRL	刀具半径补偿时的逼近/回退特性
42496	CUTCOM_CLSDT_CONT	封闭轮廓中刀具半径补偿的特性
42900	MIRROR_TOOL_LENGTH	镜像中的刀具长度符号切换
42910	MIRROR_TOOL_WEAR	镜像中的刀具磨损符号切换
42920	WEAR_SIGN_CUTPOS	有刀沿位置的刀具的磨损符号
42930	WEAR_SIGN	磨损的符号

编号	名称：$SC_	说明
42935	WEAR_TRANSFORM	用于刀具分量的转换
42940	TOOL_LENGTH_CONST	平面切换时切换刀具长度分量
42950	TOOL_LENGTH_TYPE	刀具长度补偿的指定与刀具类型无关
42960	TOOL_TEMP_COMP	基于刀具的温度补偿值
42974	TCARR_FINE_CORRECTION	激活/取消精偏 TCARR
42984	CUTDIRMOD	$P_AD[2] 或者 $P_AD[11] 的修改

19.16.3 信号

19.16.3.1 来自通道的信号

信号名称	SINUMERIK 840D sl	SINUMERIK 828D
T 功能 1 更改	DB21, ...DBX61.0	-
D 功能 1 更改	DB21, ...DBX62.0	-
T 功能 1	DB21, ...DBB116-119	DB250x.DBD2000
D 功能 1	DB21, ...DBB128-129	DB250x.DBD5000
G 指令组 7 中生效的 G 指令	DB21, ...DBB214	DB350x.DBB6
G 指令组 16 中生效的 G 指令	DB21, ...DBB223	DB350x.DBB15
G 指令组 17 中生效的 G 指令	DB21, ...DBB224	DB350x.DBB16
G 指令组 18 中生效的 G 指令	DB21, ...DBB225	DB350x.DBB17
G 指令组 23 中生效的 G 指令	DB21, ...DBB230	DB350x.DBB22

Z1:NC/PLC 接口信号

20

从版本 05/2017 起，对 NC/PLC 接口信号的详细说明参见参数手册，NC 变量和接口信号。

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

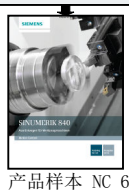
Z	
ZOA	Zero Offset Active：零点偏移标识
ZSW	（驱动）状态字

A.2 手册一览

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



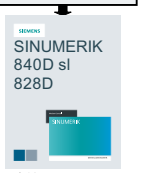
产品样本 NC 82
SINUMERIK 828D



产品样本 PM 21
SIMOTION,
SINAMICS S120



选型手册
- EMC 指令
- 工业安全



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
Leitfaden für die 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

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）。

功能

最大轨迹急动度可通过编程设定数据来启用/关闭。

参数：*值*

- 值范围：TRUE、FALSE

可用于：

- 零件程序
- 静态同步动作

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。

句法

$SC_IS_SD_MAX_PATH_JERK = *值*

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”。

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。

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

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

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。

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_ACC, 1584
$AA_ATOL, 275
$AA_FGREF, 1557
$AA_FGROUP, 1558
$AA_MOTEND, 1585
$AA_S, 1406
$AA_SCPAR, 1586
$AA_VLFCT, 1503
$AA_VMAXB, 1503
$AA_VMAXM, 1503
$AC_ACT_PROG_NET_TIME, 741
$AC_ACTUAL_PARTS, 748, 753
$AC_ASUP, 652
$AC_AUXFU_EXT, 502
$AC_AUXFU_M_EXT, 502
$AC_AUXFU_M_STATE, 503
$AC_AUXFU_M_TICK, 482
$AC_AUXFU_M_VALUE, 502
$AC_AUXFU_PREDEF_INDEX, 487, 501
$AC_AUXFU_SPEC, 499, 502
$AC_AUXFU_STATE, 503
$AC_AUXFU_TYPE, 501
$AC_AUXFU_VALUE, 502
$AC_CTOL, 275
$AC_CUT_INV, 1723
$AC_CUTMOD, 1723
$AC_CUTMOD_ANG, 1716, 1723
$AC_CUTTING_TIME, 744
$AC_CYCLE_TIME, 744
$AC_DELAYFST, 634
$AC_FGROUP_MASK, 1558
$AC_OLD_PROG_NET_TIME, 741
$AC_OLD_PROG_NET_TIME_COUNT, 742
$AC_OPERATING_TIME, 744
$AC_OTOL, 275
$AC_PATHACC, 297, 309
$AC_PATHJERK, 309, 310
$AC_PROG_NET_TIME_TRIGGER, 742
$AC_REQUIRED_PARTS, 748, 753
$AC_SGEAR, 1458, 1480
$AC_SMAXACC, 1502
$AC_SMAXACC_INFO, 1502
$AC_SMAXVELO, 1502
$AC_SMAXVELO_INFO, 1502
$AC_SMINVELO, 1502
$AC_SMINVELO_INFO, 1502
$AC_SPECIAL_PARTS, 748, 753
$AC_SPIND_STATE, 1502
$AC_STOLF, 279
$AC_TOTAL_PARTS, 748, 753
$AN_AUXFU_LIST_ENDINDEX, 492
$AN_POWERON_TIME, 740
$AN_SETUP_TIME, 740
$C_AUX_EXT, 719
$C_AUX_IS_QUICK, 719
$C_AUX_VALUE, 719
$C_D, 720
$C_D_PROG, 720
$C_DL, 720
$C_DL_PROG, 720
$C_DUPLO, 720
$C_DUPLO_PROG, 720
$C_M, 719
$C_M_PROG, 719
$C_ME, 719
$C_MTL, 716, 720
$C_MTL_PROG, 716, 720
$C_T, 719
$C_T_PROG, 719
$C_TCA, 719
$C_TE, 719
$C_THNO, 720
$C_THNO_PROG, 720
$C_TS, 719
$C_TS_PROG, 719
$P_AD, 1724
$P_CHANNO, 626
$P_CTOL, 275
$P_CUT_INV, 1723
$P_CUTMOD, 1723
$P_CUTMOD_ANG, 1716, 1723
$P_DELAYFST, 634
$P_FGROUP_MASK, 1558
$P_GEAR, 1480
$P_GFRNUM, 843
$P_IFRAME, 839
$P_ISTEST, 538
$P_OTOL, 275
$P_PROG_EVENT, 625
$P_REPINF, 645
$P_SEARCH_S, 484, 551
$P_SEARCH_SDIR, 484, 551
$P_SEARCH_SGEAR, 484, 551, 1458
$P_SEARCH_SMODE, 551
$P_SEARCH_SPOS, 484, 551

$P_SEARCH_SPOSMODE, 485, 551
$P_SGEAR, 1458
$P_SIM, 545
$P_STOLF, 279
$P_SUB_AUTOGEAR, 729
$P_SUB_AXFCT, 729
$P_SUB_CA, 729
$P_SUB_GEAR, 729
$P_SUB_LA, 729
$P_SUB_M19, 730
$P_SUB_SPOS, 730
$P_SUB_SPOSA, 730
$P_SUB_SPOSIT, 730
$P_SUB_SPOSMODE, 730
$P_SUB_STAT, 718, 720
$P_TOOLENV, 1771
$P_TOOLENVN, 1770
$P_UIFR, 839
$P_UIFRNUM, 840
$P_WORKAREA_CS_COORD_SYSTEM, 126
$PA_ATOL, 275
$PA_FGREF, 1557
$PA_FGROUP, 1558
$SC_IS_SD_MAX_PATH_ACCEL, 295
$SC_IS_SD_MAX_PATH_JERK, 308
$SC_PA_ACTIV_IMMED, 157, 170
$SC_PA_CENT_ABS, 158
$SC_PA_CENT_ORD, 158
$SC_PA_CONT_ABS, 158
$SC_PA_CONT_NUM, 157
$SC_PA_CONT_ORD, 158
$SC_PA_CONT_TYP, 158
$SC_PA_LIM_3DIM, 157
$SC_PA_MINUS_LIM, 157
$SC_PA_ORI, 157
$SC_PA_PLUS_LIM, 157
$SC_PA_T_W, 157
$SC_SD_MAX_PATH_ACCEL, 295
$SC_SD_MAX_PATH_JERK, 307
$SN_PA_ACTIV_IMMED, 157, 170
$SN_PA_CENT_ABS, 158
$SN_PA_CENT_ORD, 158
$SN_PA_CONT_ABS, 158
$SN_PA_CONT_NUM, 157
$SN_PA_CONT_ORD, 158
$SN_PA_CONT_TYP, 158
$SN_PA_LIM_3DIM, 157
$SN_PA_MINUS_LIM, 157
$SN_PA_ORI, 157
$SN_PA_PLUS_LIM, 157
$SN_PA_T_W, 157
$TC_DP1, 1713
$TC_DP10, 1714
$TC_DP11, 1713
$TC_DP2, 1713, 1717
$TC_DP24, 1714
$TC_DPCE[t,d], 1603
$TC_DPNT, 1530
$TC_DPV, 1728
$TC_DPV3, 1728
$TC_DPV4, 1729
$TC_DPV5, 1729
$VA_ABSOLUTE_ENC_DELTA_INIT, 116
$VA_ENC_ZERO_MON_ERR_CNT, 114, 116
$VA_TORQUE_AT_LIMIT, 358
$VC_SGEAR, 1458

_

_N_STRTLK, 536
_N_STRTUL, 536

“

“电机-负载”齿轮箱, 392

2

2D 刀具半径补偿, 1634
  逼近和回退特性, 1637
  防撞监控装置, 1663
  过渡圆弧, 1657
  几何, 1628
  交点 G451, 1659
  可变补偿值, 1667
  内角的补偿, 1661
  平滑逼近和退回, 1642
  取消, 1656
  外角的补偿, 1656
  选择, 1636
  已修改的报警特性, 1669

3

3D 端面铣削, 1629

A

ACC, 1489, 1583
Access MyMachine (AMM), 1333
ACN, 1487
ACP, 1487

ADDFRAME, 884
AG_SEND、AG_RECV, 985
ALF, 647, 1543
ANY, 1229
ASUB
  SERUPRO 末尾, 492
  存在用户报警的情形下, 653
  激活, 637
  内部, 650
  优先级, 647
  重新整理, 638
ATOL, 272
ATRANS, 769
AUXFUDEL, 493
AUXFUDELG, 494
AUXFUMSEQ, 482
AUXFUSYNC, 493

B

BAG, 978
BLSYNC, 647
BRISK, 290
BRISKA, 290

C

Cancel 型报警, 48
CFC, 1527
CFCIN, 1527
CFINE, 769
CFTCP, 1527
CLRINT, 648
COMPCAD, 256, 267
COMPCURV, 256, 267
COMPOF, 267
COMPON, 256, 267
COMPSURF, 256, 267
CPROT, 167
CPROTDEF, 163
CTOL, 272
CTRANS, 769

D

D 功能, 434, 1597
D 号
  设定任意 ..., 1602
D 号结构
  - 平面 （无刀具管理）, 1612
D/DL 功能替换, 716
DB10
  DBB4-7, 928
  DBB71, 376
  DBX103.0, 46
  DBX104.3, 998
  DBX104.4, 998
  DBX104.7, 45, 981
  DBX106.1, 927, 929, 930
  DBX107.7, 376
  DBX108.3, 45
  DBX108.5, 45
  DBX108.6, 45
  DBX108.7, 46
  DBX109.0, 46
  DBX109.6, 46
  DBX109.7, 46
  DBX56.1, 55, 927, 928, 930
  DBX56.2, 927, 929
  DBX56.4, 74
  DBX56.5, 74
  DBX56.6, 74
  DBX56.7, 74
  DBX92.0, 980
  DBX92.1, 980
  DBX92.2, 980
  DBX92.4, 981
  DBX92.5, 981
  DBX92.6, 981
DB11
  DBX0.0, 527, 529
  DBX0.1, 527, 1350, 1352
  DBX0.2, 527, 1350, 1352
  DBX0.4, 532
  DBX0.5, 524
  DBX0.6, 524
  DBX0.7, 525, 604, 929, 1352, 1353, 1354
  DBX1.0 - DBX1.2, 528
  DBX1.2, 1350, 1352
  DBX26.4, 530
  DBX26.5, 530
  DBX4.0 - DBX4.2, 526
  DBX46.4, 530
  DBX46.5, 530
  DBX5.0 - DBX5.2, 528
  DBX6.0, 527, 530
  DBX6.1, 527
  DBX6.2, 527
  DBX6.3, 525, 927, 929
  DBX6.4, 530
  DBX6.5, 530
  DBX6.7, 525

DBX7.0, 530
DBX7.0 - DBX7.2, 528
DB1600, 1243
DB19
DBB13, 48
DBB16, 48
DBB17, 48
DBB26, 48
DBB27, 48
DBB33, 1011
DBB36, 1011
DBX 0.3, 48
DBX 0.4, 48
DBX0.0, 47
DBX0.1, 47
DBX0.2, 47
DBX0.7, 48
DBX20.3, 48
DBX20.4, 48
DBX32.0 - .5, 1010
DBX32.6, 1010
DBX32.7, 1010
DB1900
DBB5011, 1332
DBB5021, 1332
DBB5022, 1332
DBX5021.0-.5, 1332
DBX5021.6, 1332
DBX5021.7, 1332
DB2 中的消息信号, 1228
DB21
DBX24.4, 468
DBX378.1, 645
DB21, ...
D35.0, 601
D35.5, 601
DBB116 - DBB136, 475
DBB140 - DBB190, 475
DBB194, 1490
DBB194 - DBB206, 475
DBB202, 1490
DBB376, 620
DBB4, 1562
DBB5, 1562
DBB58 - DBB67, 475
DBB68 - DBB112, 475
DBB68ff., 1490
DBX0.4, 540
DBX0.6, 542, 1567
DBX1.0, 1353
DBX1.1, 186
DBX1.6, 547, 550
DBX1.7, 537
DBX10.0 - DBX11.1, 160
DBX12.3, 1560
DBX16.3, 1560
DBX2.0, 544, 601, 630, 631
DBX2.0 - 7, 671
DBX20.3, 1560
DBX24.6, 542, 1567
DBX25.7, 537
DBX26.0, 543
DBX272.0 - DBX273.1, 160
DBX274.0 - DBX275.1, 160
DBX276.0 - DBX277.1, 161, 184
DBX278.0 - DBX279.1, 161, 184
DBX29.0, 1577
DBX29.1, 1577
DBX29.2, 1577
DBX29.3, 1577
DBX30.5, 468
DBX31.0 - 2, 567
DBX31.0 - DBX31.2, 568
DBX31.0-31.2, 573
DBX31.4, 566, 567, 568, 569
DBX31.6 - 7, 671
DBX317.1, 748, 753
DBX318.0, 638, 639
DBX318.5, 468
DBX318.6, 1568
DBX319.0, 568, 569
DBX319.1 - DBX319.3, 567, 568, 569, 570
DBX319.5, 568, 572
DBX32.3, 547
DBX32.4, 547, 548
DBX32.6, 485, 547
DBX33.0, 1353
DBX33.4, 547
DBX33.7, 538
DBX35.0, 600, 604
DBX35.0 - DBX35.4, 645
DBX35.1, 604
DBX35.2, 604
DBX35.3, 540, 604
DBX35.4, 604
DBX35.5, 600, 606
DBX35.5 - DBX35.7, 645
DBX35.6, 606
DBX35.7, 604, 606
DBX36.2, 1351, 1353
DBX36.6, 46
DBX36.7, 46, 550
DBX384.0, 609
DBX39.0, 160

DBX4.3, 1560
DBX6.0, 1560
DBX6.1, 1412, 1452
DBX6.2, 49
DBX6.6, 1562
DBX6.7, 1562
DBX7.1, 538, 603
DBX7.2, 601, 630
DBX7.3, 601, 630
DBX7.4, 601, 630, 1454, 1501
DBX7.5, 536
DBX7.7, 604, 929, 1352, 1353, 1354
DBX8.0 - DBX9.1, 160
DB21, ... DB32.6, 550
DB21, ... DBB4, 103
DB21, ... DBX0.4, 548, 644
DB21, ... DBX36.6, 550
DB21, ... DBX6.1, 644
DB21, …
DBX378.0, 637
DB31
DBX84.0, 1495
DB31, ...
DBB0, 1563
DBB1.5, 1390
DBB1.6, 1390
DBB19, 1564
DBB2.1, 1390
DBB60.4, 1376, 1384, 1387
DBB60.5, 1376, 1384, 1387
DBB68ff., 1490
DBX1.0, 49, 58
DBX1.1, 345
DBX1.2, 341
DBX1.3, 49, 341, 1501
DBX1.4, 50, 52, 56, 106, 1379, 1380
DBX1.5, 53, 112, 131, 385, 1442
DBX1.6, 53, 112, 131, 385, 1442
DBX1.7, 1563, 1564
DBX10.0, 564, 567, 568, 569, 573
DBX102.5, 131, 134
DBX102.6, 131, 134
DBX12.0, 117
DBX12.1, 117
DBX12.2, 118
DBX12.3, 118
DBX12.7, 1360, 1363
DBX16.0 - 2, 1465
DBX16.0 - DBX16.2, 1407, 1448, 1453, 1454, 1455, 1459, 1472
DBX16.3, 1403, 1407, 1448, 1454, 1455, 1465, 1467, 1469, 1472, 1476
DBX16.4, 1443
DBX16.5, 1443
DBX16.7, 1501
DBX17.0, 1475
DBX17.4 - DBX17.5, 1471
DBX18.4, 1472
DBX18.5, 1407, 1448, 1472, 1501
DBX2.1, 50, 52, 54, 55, 131, 133, 341, 1379, 1380, 1471
DBX2.2, 57, 565, 1406, 1415, 1454, 1501
DBX2.3, 106
DBX2.4 - DBX2.7, 1368
DBX20.1, 1455
DBX21.0-4, 59, 61
DBX21.5, 60
DBX21.6, 60
DBX21.7, 55, 60, 345
DBX28.7, 535
DBX3.1, 340, 344, 345
DBX3.2, 1577
DBX3.3, 1577
DBX3.4, 1577
DBX3.5, 1577
DBX3.7, 590
DBX30.0, 1494
DBX30.1, 1494
DBX30.2, 1494
DBX30.3, 1494
DBX30.4, 1494
DBX39.0, 183, 187
DBX4.3, 1469, 1501
DBX4.6, 1350
DBX4.7, 1350
DBX60.2, 112, 1506
DBX60.3, 112, 1506
DBX60.4, 53, 131, 1351, 1352, 1353, 1354, 1368, 1375, 1378, 1380, 1506
DBX60.4 - DBX60.5, 1461, 1465, 1470, 1471, 1472
DBX60.5, 53, 131, 929, 1351, 1352, 1353, 1354, 1368, 1375, 1378, 1380, 1506
DBX60.6, 53, 96, 106, 1411, 1418, 1423, 1508
DBX60.7, 53, 96, 106, 1419, 1423, 1476, 1508
DBX61.0, 49, 58
DBX61.3, 50, 58, 1379, 1380
DBX61.4, 57, 58, 1380, 1406, 1407, 1413, 1448, 1499
DBX61.5, 55, 58, 1405, 1465, 1466, 1471
DBX61.6, 55, 59
DBX61.7, 59
DBX62.4, 340, 344, 345
DBX62.5, 345, 348

DBX64.6, 96, 480
DBX64.7, 96, 480
DBX69.0, 59, 84
DBX69.0 - DBX69.2, 1459
DBX69.1, 59, 84
DBX69.2, 59, 84
DBX7.7, 1501
DBX70.0, 568, 571
DBX70.1, 568, 570, 571
DBX70.2, 568, 569, 570
DBX71.4, 1443
DBX71.5, 1443
DBX72.0, 568
DBX76.0, 59
DBX76.4, 568, 572
DBX82.0 - DBX82.2, 1448, 1450, 1452, 1453, 1463, 1465, 1466, 1471, 1474
DBX82.3, 1448, 1450, 1451, 1465, 1466, 1467, 1471, 1474
DBX83.1, 1406, 1418, 1450, 1452, 1500, 1501
DBX83.2, 1450, 1452, 1500, 1501
DBX83.5, 1406, 1407, 1412, 1413, 1414, 1500
DBX84.5, 1471, 1476
DBX84.6, 1403, 1465, 1466, 1471, 1474, 1476
DBX84.7, 1405, 1448, 1472
DBX85.5, 1424
DBX9.0, 57, 59, 83
DBX9.1, 57, 59, 83
DBX9.2, 57, 59, 83
DBX9.3, 58
DBX92.1, 60
DBX93.5, 46, 55, 61
DBX93.6, 60, 61
DBX93.7, 60, 61
DBX94.0, 61
DBX94.1, 62
DBX94.2, 62
DBX94.3, 62
DBX94.4, 62
DBX94.5, 63
DBX94.6, 63
DB31, ... DBX1.1, 342
DB31, ... DBX1.4, 106
DB31, ... DBX102.3, 97, 99
DB31, ... DBX2.3, 97, 98
DB31, ... DBX61.3, 99
DB31, ... DBX62.5, 342
DB31, ... DBX64.6, 100, 101, 103
DB31, …
DBX9.3, 1586
DB4500, 1244
DB9000 - DB9063, 1244
DC, 1487
DELAYFSTOF, 632
DELAYFSTON, 632
DELTOOLENV, 1768
DIACYCOFA, 941
DIAM90, 940
DIAM90/DIAM90A[AX], 935
DIAM90A, 941
DIAMCHAN, 941
DIAMCHANA, 941
DIAMCYCOF, 940
DIAMCYCOF/DIACYCOFA[AX], 935
DIAMOF, 940
DIAMOFA, 941
DIAMON, 940
DIAMON/DIAMONA[AX], 935
DIAMONA, 941
DILF, 1543
DISABLE, 648
DISC, 1657
DITE, 1541
DITS, 1541
DL 功能, 435
DRIVE, 323
DRIVEA, 323
DYNFINISH, 249
DYNNORM, 249
DYNPOS, 249
DYNROUGH, 249
DYNSEMIFIN, 249

E

EES, 685
ENABLE, 648
ENDLABEL, 611
EXTCLOSE, 699
EXTOPEN, 699

F

F 功能, 435
FA, 1489, 1558
FA 功能, 436
FB, 1581
FB 中的 ANY, 1231
FB 中的 POINTER, 1231
FB1 RUN_UP（基本程序、启动部分）, 1051
FB10 安全继电器, 1128
FB11 制动测试, 1131
FB2 GET（读取 NC 变量）, 1061

FB29 Diagnose（信号记录器和数据触发器）, 1136
FB3 PUT（写入 NC 变量）, 1072
FB4 PI_SERV（PI 服务）, 1080
    可用 PI 服务, 1085, 1086
FB5 GETGUD（读取 GUD 变量）, 1113
FB7 PI_SERV2（PI 服务）, 1122
FB9 MtoN（操作单元切换）, 1123
FC 中的 ANY, 1229
FC 中的 POINTER, 1229
FC10 AL_MSG, 1163
FC1005 AG_SEND, 1222
FC1006 AG_RECV, 1224
FC12 AUXFU, 1165
FC13 BHGDisp, 1166
FC17 YDelta, 1170
FC18 SpinCtrl, 1175
FC19 MCP_IFM, 1186
FC2 GP_HP（基本程序循环部分）, 1141
FC21 Transfer, 1194
FC22 TM_DIR, 1203
FC24 MCP_IFM2, 1206
FC25 MCP_IFT, 1209
FC26 HPU_MCP, 1212
FC3 GP_PRAL（基本程序报警控制部分）, 1142
FC5 GP_DIAG（基本程序，诊断）, 1145
FC6 TM_TRANS2, 1147
FC7 TM_REV, 1148
FC8 TM_TRANS, 1152
FC9 ASUP, 1160
FCUB, 1578
FGROUP, 775, 778
FIFO 缓存, 681
FL, 1527
FLIN, 1578
FNORM, 1578
FOC, 338
FOCOF, 338
FOCON, 338
FPO, 1578
FPRAOF, 1488
FPRAON, 1488
FRC, 1579
FRCM, 1579
FXS, 338
FXS-REPOS, 349
FXST, 338
FXSW, 338
FZ, 1529

G

G 功能组, 594
G0 公差系数, 276, 277
G0 时的公差 ";", 276
G25, 123, 1488
G26, 123, 1488
G33, 1536
G331, 1553
G332, 1553
G335, 1548
G336, 1548
G40, 1636
G41, 1636
G42, 1636
G450/G451, 1656
G451, 1659
G460, 1671
G461, 1595, 1672
G462, 1595, 1673
G58, 769
G59, 769
G60, 206
G601, 207
G602, 207
G603, 207
G63, 1438, 1555
G64, 213
G642, 218
G643, 218
G644, 222
G645, 226
G70, 380
G700, 380
G71, 380
G710, 380
G74, 1354
G9, 206
G91 扩展, 1725
    零点偏移, 1727
G93, 1528
G94, 1529
G95, 1529
G96, 1532
G96、G961 下的主从切换, 1536
G96、G961 下的转速限制, 1534
G961, 1532
G97, 1534
G971, 1534
GETTCOR, 1771
GETTENV, 1769
GFRAME0 ... GFRAME100, 844

H

H 功能, 433
HHH"; "UUU, 99
HMI 监示器, 1028

I

IC, 1487
IR, 1548

J

JR, 1548

K

KONT, 1637
KONTC, 1637
KONTT, 1637
KR, 1548
Kv 系数, 408

L

LENTOAX, 1792
LFOF, 1543
LFON, 1543
LFPOS, 1543
LFTXT, 1543
LFWP, 1543
LIFTFAST, 647
LIMS, 1488
Lockset 文件
    用于激活数控禁用功能, 1335
    用于解禁控制系统, 1342, 1344
    用于取消数控禁用功能, 1340
    用于延长数控禁用功能, 1338

M

M 功能替换, 713
M1, 506
M17, 505
M19, 1409, 1487
M2, 505
M3, 1486
M30, 505
M4, 1486
M40, 1490, 1509
M41, 1490
M42, 1490
M43, 1490
M44, 1490
M45, 1490
M5, 1486
M70, 1488
MCP 识别, 999
MCS, 48
Md < Mdx, 62
MD10000, 766
MD10010, 522
MD10050, 1391
MD10070, 1391
MD10125, 693
MD10131, 63
MD10192, 1462, 1463
MD10200, 364, 365, 366, 368, 378
MD10210, 364, 365, 366, 368, 402, 405
MD10220, 370
MD10230, 370
MD10240, 372, 376, 1587
MD10250, 377
MD10260, 377
MD10270, 379
MD10290, 379
MD10292, 379
MD10366, 1247
MD10368, 1247
MD10600, 815, 892
MD10602, 855, 857, 865, 870
MD10610, 770, 849
MD10612, 850
MD10615, 899
MD10618, 162
MD10651, 1549
MD10680, 254
MD10682, 254
MD10700, 675
MD10702, 541, 548, 644, 662
MD10704, 1568
MD10707, 561
MD10708, 561
MD10710, 124, 1535, 1543
MD10712, 251, 738
MD10713, 476
MD10714, 451, 715
MD10715, 714
MD10716, 714
MD10719, 717

MD10735, 529
MD10804, 715
MD10806, 715
MD10814, 715
MD1103, 352
MD1104, 352
MD1105, 352
MD11100, 464
MD11110, 471, 1795
MD11220, 378
MD11300, 1351
MD11346, 1612
MD11410, 1448, 1537, 1664
MD11411, 536
MD11450, 484, 550, 551, 552, 554, 562, 588
MD11470, 563, 565, 590
MD11550, 632, 1554
MD11600, 641
MD11602, 640, 642, 643, 644
MD11604, 588, 640, 643
MD11610, 651
MD11620, 621
MD11625, 693
MD11626, 693
MD12000, 1563
MD12010, 1564
MD12020, 1562
MD12030, 230, 231, 1562
MD12040, 1562
MD12050, 1562
MD12060, 1564
MD12070, 1565
MD12080, 1565
MD12082, 1563
MD12090, 1554
MD12100, 230, 231, 1565
MD12200, 1577
MD12202, 1576
MD12204, 1576
MD1230/1231, 352
MD14504, 991
MD14506, 991
MD14508, 991
MD14516, 1296
MD14518, 1296
MD15700, 726
MD15702, 727
MD17200, 675
MD17300, 1337
MD18080, 1745
MD18088, 1683, 1698, 1702
MD18100, 1598, 1613
MD18102, 1605, 1613, 1614
MD18104, 1767
MD18105, 1603, 1604, 1605, 1607
MD18106, 1603, 1604
MD18108, 1749
MD18112, 1750, 1765
MD18114, 1731
MD18116, 1767, 1768, 1769, 1772, 1779
MD18150, 67
MD18190, 162
MD18360, 681
MD18362, 682
MD18600, 813
MD18602, 838, 841, 845, 846
MD18960, 289, 306
MD20000, 533
MD20050, 709, 766, 876
MD20060, 766
MD20070, 766
MD20080, 766, 779
MD20090, 504, 1486
MD20092, 1483
MD20094, 450, 715
MD20095, 450, 715
MD20096, 1600, 1601, 1602
MD20100, 936, 1533, 1536
MD20105, 588, 649
MD20106, 552, 622, 662
MD20107, 553, 623
MD20108, 621
MD20109, 622
MD20110, 374, 640, 898, 899, 900, 901, 910, 939, 1596, 1616, 1617, 1704, 1726, 1755
MD20112, 561, 640, 708, 904, 910, 939, 1616, 1704
MD20115, 588, 643, 649
MD20116, 644
MD20117, 644, 662
MD20118, 709
MD20120, 708, 1615, 1704
MD20121, 708, 1596, 1615
MD20124, 504
MD20125, 1719
MD20126, 1704, 1720
MD20127, 1720
MD20130, 708, 1615
MD20140, 708
MD20144, 710
MD20150, 124, 279, 290, 305, 323, 324, 379, 533, 594, 708, 830, 902, 903, 939, 1587
MD20152, 708, 902, 939
MD20170, 258
MD20171, 258

MD20172, 258, 1578
MD20173, 258
MD20180, 1695
MD20184, 894, 1697
MD20188, 1690
MD20190, 1690
MD20191, 645
MD20192, 624
MD20193, 624
MD20194, 640, 654
MD20200, 1580
MD20201, 1579
MD20202, 1652
MD20204, 1645
MD20210, 1659
MD20220, 1658
MD20230, 1660
MD20240, 1661, 1663
MD20250, 1636
MD20252, 1677
MD20256, 1670
MD20270, 434, 1597, 1614, 1615, 1755, 1756
MD20272, 435, 1749, 1755, 1756
MD20310, 576
MD20360, 937, 1611, 1612, 1765, 1776, 1791
MD20390, 1740, 1742, 1744
MD20392, 1741
MD20400, 230
MD20430, 231
MD20440, 231
MD20443, 234
MD20450, 232
MD20460, 238, 239
MD20462, 238
MD20465, 242
MD20480, 219, 222, 271
MD20482, 258, 271
MD20484, 258
MD20485, 258
MD20486, 258
MD20487, 258
MD20488, 270
MD20490, 213
MD20500, 291, 292
MD20550, 209
MD20552, 209
MD20560, 277
MD20600, 304
MD20602, 300, 301
MD20606, 253
MD20610, 294
MD20624, 939
MD20700, 643, 1385, 1386
MD20750, 1534
MD20800, 431, 505, 506
MD20850, 1410
MD21015, 65
MD21016, 66
MD21020, 121
MD21158, 326
MD21159, 326
MD21166, 325
MD21168, 326
MD21220, 1571
MD21230, 1570
MD21330, 708
MD22000, 464, 1409
MD22010, 465, 1409
MD22020, 465, 1409
MD22030, 466, 1409
MD22035, 466, 1795
MD22040, 456, 465
MD22050, 457
MD22060, 457
MD22070, 458
MD22080, 458, 728, 1795
MD22100, 476
MD22110, 433, 436
MD22200, 469
MD22210, 432, 469
MD22220, 434, 469
MD22230, 433, 469
MD22240, 435, 436, 469, 1559
MD22250, 434, 469
MD22252, 435, 469
MD22254, 450, 467, 715
MD22256, 451, 467, 715
MD22410, 1560, 1567, 1588
MD22510, 594, 1227
MD22530, 499, 1704
MD22532, 499
MD22534, 499
MD22550, 717, 1596, 1609, 1610, 1614, 1616
MD22560, 450, 504, 716, 1596, 1616
MD22562, 1610, 1611
MD22600, 574
MD22620, 592
MD24004, 899
MD24006, 805, 830, 900
MD24007, 903
MD24008, 805, 899
MD24010, 849
MD24020, 836
MD24040, 876

MD24100, 1742
MD24110, 1742
MD24120, 1743
MD24558, 1778
MD24570, 1743
MD24572, 1743
MD24658, 1778
MD24805, 865
MD24855, 865
MD24905, 858
MD24955, 858
MD25574, 1743
MD26008, 451, 715
MD26012, 715
MD27100, 676, 939
MD27800, 534
MD27850, 745
MD27860, 610, 744, 745
MD27880, 610, 749
MD27882, 750
MD28060, 672, 675
MD28070, 270
MD28071, 258
MD28072, 258
MD28082, 830, 851, 885, 886, 894, 1697
MD28085, 1598, 1698, 1703
MD28150, 69
MD28200, 162
MD28210, 162
MD28212, 162
MD28400, 675
MD28402, 675
MD28530, 67, 221
MD28533, 234
MD28560, 905
MD28600, 125
MD28610, 253
MD28620, 1666
MD30100, 387
MD30110, 387
MD30120, 387
MD30130, 386, 388
MD30200, 385, 396
MD30210, 388, 396
MD30220, 388, 396
MD30230, 389, 396
MD30240, 386, 389, 396
MD30242, 386, 390, 396, 1377
MD30244, 397
MD30250, 397, 1382
MD30260, 397
MD30270, 397
MD30300, 396, 398, 402, 405, 1587
MD30310, 119, 396
MD30320, 396
MD30330, 396, 1376
MD30340, 396, 1376
MD30350, 386, 1302
MD30455, 1376, 1489
MD30460, 937, 940
MD30550, 1352
MD31000, 398, 402, 405
MD31010, 398
MD31020, 398, 402, 405, 1391
MD31025, 402
MD31030, 150, 396, 398, 402
MD31040, 398, 402, 405, 420, 1420, 1471
MD31044, 393, 398, 1461
MD31046, 134
MD31050, 83, 150, 392, 398, 399, 402, 405, 1460, 1470
MD31060, 83, 150, 392, 398, 399, 402, 405, 1460
MD31064, 392, 396, 1461
MD31066, 392, 396, 1461
MD31070, 150, 398, 402, 405
MD31080, 150, 398, 402, 405
MD31090, 379
MD31122, 1441
MD31123, 1441
MD31200, 379
MD32000, 92, 322, 363, 396, 1506, 1527
MD32060, 590, 1583
MD32074, 899
MD32100, 394
MD32200, 57, 83, 92, 95, 408
MD32210, 422
MD32220, 422
MD32250, 150, 395
MD32260, 150
MD32300, 92, 99, 289, 298, 299, 300, 301, 322, 1583
MD32301, 325
MD32310, 213, 299, 300
MD32320, 1584
MD32400, 316, 416
MD32402, 316, 416
MD32410, 317, 416
MD32420, 291, 305, 306
MD32430, 305
MD32431, 304, 311
MD32432, 311
MD32433, 299
MD32434, 298, 299, 310
MD32435, 311

MD32436, 325
MD32439, 312
MD32440, 239, 242
MD32610, 92
MD32620, 412
MD32630, 412
MD32640, 415
MD32711, 380
MD32750, 1740, 1742, 1744
MD32800, 83, 92
MD32810, 83, 92, 150, 412
MD32890, 418, 419
MD32900, 408
MD32910, 83, 408, 414
MD32950, 420
MD33050, 59
MD33100, 219, 254, 257, 271
MD33120, 226, 271
MD34000, 1365, 1375
MD34010, 1350, 1360, 1362, 1364, 1373, 1384
MD34020, 1360, 1364
MD34040, 1362, 1364, 1365, 1373, 1374, 1442
MD34050, 1362
MD34060, 1366, 1374, 1375, 1421
MD34070, 1367, 1375
MD34080, 1368
MD34090, 397, 1368, 1370, 1371, 1383, 1384, 1385, 1386
MD34092, 1365
MD34093, 1366
MD34100, 1368, 1375, 1384, 1385
MD34102, 1377
MD34104, 1379
MD34110, 1352
MD34200, 1371, 1377, 1379, 1384, 1387, 1442
MD34210, 1381, 1382, 1383, 1384, 1385
MD34300, 1373
MD34320, 397, 398, 1370
MD34330, 1387
MD34990, 1528
MD35000, 766, 825, 1483, 1490
MD35010, 1445, 1449, 1465, 1471, 1476
MD35012, 1444, 1471, 1474
MD35014, 1477
MD35020, 1431
MD35030, 1431
MD35035, 1410, 1479, 1481
MD35040, 1406
MD35090, 1444
MD35092, 1449
MD3510, 1476
MD35100, 363, 1504
MD35110, 1444, 1509
MD35112, 1449
MD35120, 1444, 1509
MD35122, 1449
MD35130, 83, 1444, 1450, 1452, 1501
MD35135, 1445
MD35140, 1445, 1450, 1452, 1501, 1535, 1537
MD35150, 1363, 1412, 1414, 1500, 1504
MD35200, 1405, 1415, 1417, 1421, 1445, 1475
MD35210, 1405, 1414, 1417, 1418, 1421, 1422, 1445, 1475
MD35212, 1449
MD35220, 319, 322
MD35230, 319, 322
MD35240, 221, 322
MD35242, 318
MD35300, 1391, 1414, 1416, 1417, 1421, 1422, 1445, 1475, 1489
MD35310, 1445, 1471, 1474
MD35350, 1421, 1474
MD35400, 1177, 1464, 1466
MD35410, 1464, 1466
MD35430, 1464, 1466
MD35500, 1407, 1412
MD35510, 1412, 1499
MD35550, 1445, 1450
MD35590, 57, 83, 1446, 1459
MD36000, 206, 1419, 1423, 1507, 1508
MD36010, 95, 206, 1419, 1423, 1507, 1508
MD36012, 97, 208
MD36020, 95
MD36030, 96, 97
MD36040, 96, 358
MD36042, 358
MD36050, 97, 98, 106
MD36051, 97, 98
MD36052, 98, 99, 101, 103, 105
MD36060, 57, 58, 1406, 1413, 1499
MD36100, 118
MD36110, 118
MD36120, 118
MD36130, 118
MD36200, 109
MD36210, 107, 395
MD36220, 108
MD36300, 112, 1390, 1391
MD36302, 1416
MD36310, 110, 113
MD36312, 114
MD36400, 92
MD36500, 385
MD36510, 386

MD36600, 117
MD36610, 55, 93, 95, 96, 99, 109, 110, 112, 114, 927
MD36620, 55, 928
MD37002, 345
MD37010, 341
MD37012, 345
MD37020, 342, 352
MD37030, 341
MD37040, 341
MD37060, 340, 344, 345
MD37080, 357
MD51029, 671
MD51074, 750
MD51230, 1248
MD51231, 1249
MD9004, 367, 677
MD9006, 47
MD9010, 677
MD9011, 367, 677
MD9424, 677, 808
MD9440, 909
MTL, 716

N

nact, 62
NC
    -Start, 600
    读写变量, 984
    故障, 981
    语言范围, 595
NC/PLC 接口, 533
NCK 电池报警, 46
NCK-CPU 就绪, 45
NCK-PLC 启动和同步, 977
NC-VAR-Selector, 1039
    操作选项, 1043
    调试、安装, 1051
NORM, 1637
NPROT, 167
NPROTDEF, 163

O

OTOL, 272
OVR, 1566
OVRA, 1489, 1566

P

PCOF, 1488
PLC
    - 读写变量, 67
    HMI 监示器, 1028
    按键禁用, 1015
    版本, 947
    程序列表, 1010, 1012, 1331
    基本程序功能, 947
    轴, 776
POINTER, 1229
POLF, 1543
POLFMASK, 1543
POLFMLIN, 1543
POS, 775, 776
POSA, 775, 776
PRESETON, 790, 1354
PRESETONS, 794
Process DataShare, 694, 699
PROFIBUS
    连接, 1001
PROFIBUS 连接, 1001
PROFIBUS 诊断, 980
PROFINET
    连接, 1004

R

Recall 型报警, 48
REPEAT, 611
REPEATB, 611
REPOS 偏移, 348
RESET
    指令, 604

S

S 功能, 432
S..., 1487
SAVE, 648
SBL2 的排故模式, 662
SBLOF, 663
SBLON, 663
SCC, 939
SCPARA, 1586
SD41100, 1531
SD42010, 1540, 1542
SD42100, 541, 1568
SD42101, 1569

SD42200, 663
SD42440, 771, 1726
SD42442, 1726
SD42444, 548
SD42465, 219, 271
SD42466, 219, 271
SD42470, 258
SD42471, 258
SD42472, 258
SD42473, 259
SD42475, 259
SD42476, 259
SD42477, 259
SD42480, 1669
SD42496, 1669
SD42500, 295
SD42502, 295
SD42510, 307
SD42512, 307
SD42600, 1497, 1531
SD42676, 272
SD42678, 272
SD42700, 683
SD42750, 674
SD42800, 1483
SD42900, 1594, 1734, 1758, 1766, 1794
SD42910, 1734, 1735, 1758, 1763, 1764, 1766
SD42920, 1734, 1735, 1758, 1763, 1764, 1766
SD42930, 1734, 1736, 1758, 1764, 1766
SD42935, 1711, 1734, 1759, 1764, 1766, 1795
SD42940, 1722, 1732, 1734, 1737, 1758, 1764, 1766
SD42942, 1732, 1766
SD42950, 1713, 1731, 1734, 1758, 1764, 1766
SD42954, 1766
SD42956, 1766
SD42960, 1594, 1734, 1741, 1742
SD42974, 1690
SD42977, 1666
SD42984, 1721
SD42990, 672
SD43200, 1493, 1495
SD43202, 1494, 1495
SD43206, 1494, 1496
SD43210, 1535, 1537
SD43220, 1535
SD43230, 1534
SD43235, 1505
SD43240, 1409, 1487
SD43250, 1409
SD43300, 1497, 1531
SD43400, 123
SD43410, 123
SD43420, 122
SD43430, 122
SD43500, 352
SD43510, 352
SD43520, 352
SERUPRO
    可编写的中断指示, 580
    末尾 ASUB, 492
    自动的中断指示, 582
SERUPRO ASUB, 348
    特殊性, 575
SERUPRO 定位
    通过 PLC 控制, 568
seruproMasterChan, 578
SETINT, 636, 646
SETMS, 1486
SETTCOR, 1778
SOFT, 304
SOFTA, 306
SPCOF, 1488
SPCON, 1488
SPI, 1490
SPOS, 1408, 1487
SPOSA, 1487
STOLF, 277
STRINGIS, 595
SVC, 1487

T

T 功能, 433, 1596
T 功能替换, 716
TCARR, 1696
TCOABS, 1696
TCOFR, 1695, 1696
TCP-Tool Center Position, 783
TEACH IN, 526
Tick 计数器, 488
TOA
    单元, 1598
    数据, 1623
TOOLENV, 1765
TOWBCS, 1762
TOWKCS, 1763
TOWMCS, 1762
TOWSTD, 1761
TOWTCS, 1763
TOWWCS, 1762
TRANS, 769
TRANSMIT, 800

U

UDT 指定, 986
UTD 模块, 985

V

VELOLIM, 1489

W

WAITENC, 710
WAITS, 1489
WALCS0, 127
WALIMOF, 124
WALIMON, 124
WCS, 48

X

XE * MERGEFORMAT,

按

按键禁用, 47, 1015

包

包
    计数器, 488

保

保护等级
    可设置的, 75
保护级, 71
    用户 ASUB 中, 651
保护区, 155, 163
    限制, 187

闭

闭环控制, 407

编

编程设备
    硬件前提条件, 1036
编码器监控, 112
    编码器频率, 112

标

标签, 611
标准, 925

并

并行轴, 983

补

补偿号, 1603

不

不输出设定值的程序执行, 537

参

参考点, 783
参考点 R, 783
参考轴
    用于 G96 / G961 / G962, 933
参数组
    可定向刀架, 1681
    在轴上, 411

操

操作面板上连接的 HMI - CPU 就绪, 45

槽

槽锯, 1623

测

测量系统, 385, 1349

插

插补结束, 207
插补轴组, 55

程

程序
    测试, 537
    动作, 607
    显示状态, 604
    运行时间, 740
    状态, 604
程序部分
    重复, 611
程序段
    跳转, 670
程序段搜索
    不进行计算（类型 1）, 545
    级联, 546
    在程序测试模式中计算 SERUPRO（类型 5）, 546
    在程序段终点计算（类型 4）, 546
    在轮廓处计算（类型 2）, 545
程序段搜索 SERUPRO
    REPOS 应答, 568
    齿轮档切换, 591
    叠加运动, 591
    轨迹轴, 567
    跨通道取轴后的 REPOS 偏移, 572
    缺省设置, 592
    设定值耦合和实际值耦合, 586
    设置 REPOS 特性, 563
    时序, 560
    通过 NC/PLC 接口信号控制 REPOS, 568
    同步主轴耦合中的 REPOS 偏移, 572
    有效范围内的 REPOS 偏移, 571
    重定位定位轴, 566
    轴功能的条件, 589
程序控制，接口信号, 670
程序显示模式, 673
程序运行, 594

齿

齿轮档, 1444
    M70, 1477
    手动设定, 1453
    通过 PLC 设定, 1452
齿轮档选择
    自动, 1509

出

出现 NCK 报警, 46
出现导致程序处理停止的 NCK 报警, 46
出现通道专用的 NCK 报警, 46

垂

垂直轴, 351

粗

粗偏, 769

带

带计算的程序段搜索
    收集的主轴功能, 551

单

单程序段
    SBL1, 661
    SBL2, 661
    SBL3, 661
    模式, 539
    抑制, 663
单位制, 372, 380
单轴动态响应, 1582

刀

刀架，可定向, 1678
    编程, 1702
    复位时的控制器特性，编程方式，再定位, 1703
    可旋转工作台, 1695
    前提条件, 1703
    斜面加工, 1694
    新建, 1698
刀架参考点 T, 783
刀具, 1596
    2D 刀具半径补偿, 1635
    DISC, 1657
    T 功能, 1597
    补偿数据, 1623
    参数, 1619

尺寸，有效的, 1630
刀具刀沿, 1597
刀沿, 1619
更换, 1596
管理, 522
基本尺寸 / 适配器尺寸, 1630
类型, 1621
磨损, 1629, 1630
使用 M06 更换, 1596
退回, 710
形状, 1628
形状，有效的, 1630
选择, 1596
长度, 1626
刀具半径补偿
保持为常数, 1674
磨损-, 1630
刀具补偿
类型, 1680
在 NC 中计算, 1600
刀具刀库轴, 775
刀具基本定向, 1728
刀具长度补偿
工件专用计算, 1756
工件专用计算，示例, 1709
几何, 1626
磨损-, 1629
刀具转塔轴, 775
刀沿
参考点, 1716
位置, 1713, 1717
形状, 1715
中心点, 1716
刀沿号, 1603
刀沿位置, 1624
相关, 1634

地

地址扩展取负值, 506

点

点动
在 AUTOMATIC 运行方式下, 528

电

电流控制器生效, 59

定

定位精度, 366
定位轴, 365, 776
定向
公差, 271

动

动态响应
自适应, 240
动态响应自适应功能, 408
动作程序段, 548
动作单程序段, 539

端

端面轴, 933
尺寸设定, 940

多

多刀, 1109
多个端面轴
跨通道取轴, 941
多实例 DB, 1235
多项式，交点程序, 1670

反

反比时间进给率（G93）, 1528

防

防撞监控装置, 1663

仿

仿真, 544

访

访问保护, 71
访问权限, 71
访问特性, 72

非

非模态限制 (FOC), 357
非线性信号失真, 91

分

分配刀具/刀架, 1679

符

符号编程, 985

辅

辅助功能
    地址扩展, 457
    定义, 430
    关联, 467
    计数器, 488
    类型, 457
    输出特性, 458
    用户专用, 464
    用户自定义, 429
    预定义, 429
    预定义的, 437
    值, 457
辅助功能汇总, 457
辅助功能输出, 594
辅助轴, 775

复

复位
    特性, 704
    指令, 604

高

高速数据通道, 67

割

割线误差, 254

跟

跟踪生效, 58

工

工件
    仿真, 544
    计数器, 748, 752
工件零点 W, 783
工件坐标系（WCS）, 48, 768, 810
工件坐标系中的实际值, 48
工作区域限制, 120
    在 BCS 中, 122
    在 WCS/AZS 中, 125
工作区域限制组, 125

公

公差
    主轴转速, 1500

功

功能接口, 963

攻

攻丝
    带弹性卡头, 1438

固

固定进给值, 1576
固件, 1241

轨

轨迹
    进给率 F, 1526
    最大速度, 1527
轨迹进给率, 365
轨迹轴, 775

过

过程报警, 980

过载系数, 213

恒

恒定切线的, 1638
恒定切削速度, 1532
恒定转速, 1534

后

后角, 1632, 1714

换

换刀
    D 功能, 1597
    补偿存储器, 1598
换刀时选择刀沿, 1597

回

回参考点
    使用增量测量系统, 1358
    旋转绝对值编码器, 1388
回参考点方式, 1349
回退
    方向，螺纹切削, 1544
回转轴, 775

机

机床的运动, 1682
机床零点, 1349
机床零点 M, 783
机床轴, 773
机床坐标系（MCS）, 48, 767, 788

基

基本程序段显示
    激活, 675
    配置, 674
基本尺寸, 1630
基本单位制, 372
基本显示
    显示缓存的大小, 675
基本坐标系（BCS）, 767, 800
基于半径的数据, 936
基于列表的 M 解码, 987

激

激活
    机床控制面板、手持操作设备, 996

级

级联程序段搜索, 554

极

极限
    轨迹轴速度, 1527

急

急动度
    上调，随速度提高的, 311
急动度限制, 224
急停
    过程, 927
    接口, 927
    应答, 929
急停控制元件, 926

几

几何轴, 774, 775, 800

计

计数脉冲, 750
计算单个磨损分量, 1763
计算精度, 367

加

加工时间, 744
加速过程结束, 62

角

角
    后-, 1714
    夹持-, 1714

接

接口
MCP/PLC 和 HHU/PLC, 961
PLC/HMI, 961, 969
PLC/HMI 消息, 969
PLC/MCP, 973
PLC/NCK, 962

进

进给倍率
可编程, 1566
进给禁止
通道专用, 1560
进给率
（G93、G94、G95）, 1528
-倍率, 1561
倒角/倒圆, 1579
反比时间进给率（G93）, 1528
轨迹进给率 F, 1526
进给/主轴停止, 1560
进给禁止, 1560
空运行进给率, 1567
-类型, 1525
每齿, 1529
线性（G94）, 1529
旋转（G95）, 1529
进给轴/主轴停止, 58
进给轴禁止, 49, 1561
进料轴, 775

禁

禁用日期, 1333

精

精插补, 407
精偏, 769

镜

镜像
框架, 824

可

可定向刀架, 1594, 1678
计算有效的刀具长度, 1695
示例, 1706
新建, 1698
运动链, 1685
可上锁的数据区, 75

空

空运行进给率, 541, 1567

控

控制器参数组切换, 82
控制系统特性
复位时, 704
零件程序结束时, 704
零件程序开始时, 704
启动时, 704

口

口令, 72

快

快速移动
-倍率, 1562

框

框架, 774
框架旋转, 770
通过空间角, 889
沿刀具方向, 892, 893
框架转换, 1680

连

连续路径运行, 210
隐含, 213

零

零点, 783

零点偏移
外部零点偏移, 802
零件程序
跳过程序段, 543
选择, 600
零脉冲诊断, 113
零矢量, 1686

轮

轮廓
采样时间, 253
采样系数, 253
公差, 271
轮廓误差, 91

螺

螺纹
切削, 1543
螺纹切削 G33, 1536
螺旋线插补, 779

每

每齿进给量, 1529

密

密码, 73

模

模拟轴, 386
模态激活 (FOCON/FOCOF), 357

默

默认口令, 73

目

目标程序段中的特殊之处
STOPRE 程序段, 583

挠

挠率, 252

平

平滑, 214
轨迹速度, 237
平滑逼近和退回
含义, 1642
子运动, 1642
平面 D 号结构, 1612

屏

屏幕待机生效, 47
屏幕待机未生效, 47

气

气温报警, 46

切

切削方向, 1713

球

球螺纹, 1548

驱

驱动测试：运行请求, 58
驱动测试：运行使能, 49
驱动处于循环运行中, 45
驱动就绪, 45

曲

曲率, 252
曲率恒定的, 1638

任

任意形状表面, 252
模式, 204, 252

软

软件限位开关, 118

润

润滑脉冲, 59

删

删除剩余行程, 49

设

设定齿轮档, 1446
设定值输出, 385
设定值系统, 384

剩

剩余时间
    工件, 743

识

识别校验, 750

实

实际值补偿, 386
实际值采集, 385
实际值处理, 394
实际值同步, 56
实际值系统
    工件相关, 905

使

使用分解运动的机床, 1688
使用可旋转刀具的车床, 1687
使用可旋转工件的车床, 1687

事

事件控制的程序运行, 617

适

适配器尺寸, 1630

受

受限制的刀架方向, 1681

输

输出
    辅助功能特性, 458
    计数器, 488
    外部设备/文件上, 699
    序列, 488
输入精度, 367

数

数据交换
    与操作面板的, 959
数据接口, 962
数据通道，高速, 67
数控禁用功能, 1333

说

说明旋转, 1690

伺

伺服使能, 54

速

速度, 363

所

所需存储容量
    PLC 基本程序的, 1032

替

替代零脉冲, 1388
替换子程序, 712

调

调节方向, 394
调整电机/负载齿轮比, 391

调整系数
轨迹动态响应, 242

跳

跳转标记
程序部分重复时, 611
跳转级, 670

停

停止事件, 629
停止延迟区, 628

通

通道
当前, 626
配置, 533
缺省设置, 594
属性, 533
显示状态, 605
状态, 606
通道轴, 774
通道状态
通道复位, 528
通道生效, 528
通道中断, 528
通过 FC17 进行的星形/三角形切换, 1455
通过 JOG 到达 LEAD 的仿真目标点, 587
通过 PLC 使用主轴功能, 534

同

同步轴, 778

外

外部程序存储器, 681
外部零点偏移, 802

未

未应答的齿轮档切换, 1462

位

位移条件, 216
位置测量系统, 53
位置差接通, 420
位置控制回路, 407
位置控制器生效, 58

物

物理量, 369

显

显示程序段，结构（DIN）, 677
显示精度, 367

限

限位开关监控, 117

线

线性进给率（G94）, 1529
线性信号失真, 91

相

相位滤波器, 418

信

信号
NCK / PLC, 965
PLC / BAG, 965
PLC / NCK 通道, 966
PLC / 轴、主轴, 967
PLC/NCK, 964
报警信号, 46
编译循环, 964
进给轴/主轴专用 (DB31, ...), 44
就绪信号, 45
通道专用 (DB21, ...), 44
信号交换
事件控制的, 945
循环, 945
信号失真, 91

星

星形/三角形切换, 1170

旋

旋转分量, 887
旋转进给率（G95）, 1529
旋转轴的参数设定, 1682

选

选择刀架, 1678

循

循环信号交换, 44
循环运行, 977

样

样条, 205

钥

钥匙开关, 74

以

以太网
    连接, 996

译

译码单程序段, 539

隐

隐含的连续路径运行, 213
隐含的准停功能, 211
隐性预处理停止, 584

硬

硬件限位开关, 117

用

用户自定义 ASUB
    SERUPRO 进程后, 562

预

预处理, 533
预处理程序停止时报警, 1669
预读, 227
    选择和取消, 229

远

远程诊断, 46

运

运动结束条件
    单轴时, 1585
运动类型, 1682
运动转换, 800
运行到固定挡块, 337
    程序段搜索, 346
    到达固定挡块, 341
    定位轴, 351
    功能中断, 350
    监控窗口, 342
    轮廓监控, 351
    取消, 345
    选择, 340
运行范围, 365
运行方式
    AUTOMATIC, 526
    JOG, 526
    JOG in AUTOMATIC, 526
    MDI, 526
    监控, 531
    切换, 525, 531
    锁定, 531
    同步动作, 526
    优先级, 527
    运行方式组, 526
运行方式切换
    从/向运行方式 AUTOMATIC、JOG、MDI, 532
运行方式组（BAG）, 522
运行时间
    程序, 740
运行使能, 1387
运行状态, 528

在

在刀架数据中保存角度, 1681

在刀具方向上加工, 1727

增

增量编程补偿值, 1725

执

执行外部子程序, 681

直

直接控制按键
    报警, 999
    地址, 998
    以太网总线上的 OP, 998
直接连接在刀具上的编码器, 393
直径编程
    通道专用, 940
    轴专用, 941

中

中断
    程序, 636
    禁止, 648
    信号, 636
中断程序
    结束, 638
中间齿轮比, 392, 1461

轴

轴监控
    跟随误差, 92
    静态, 96
    前提条件, 150
    实际速度, 109
    转速设定值, 107
轴配置, 780

主

主处理, 533
主处理轴, 777
主偏角, 1714
主运行
    轴, 1584
主轴
    -倍率, 1561, 1564
    齿轮档 0, 1455
    设定转速, 1500
    转速限值, 1502
    最大转速, 1501
    最小转速, 1501
主轴禁止, 49, 1561
主轴转速, 365

驻

驻留, 131, 1354

转

转速
    前馈控制, 412
转速控制回路, 407
转速控制器生效, 59
转速设定值输出, 394
转速设定值调整, 394

准

准停, 205
    隐含, 211
准停条件, 206, 207, 208

自

自动执行的 SERUPRO, 578
自控单轴运动, 590

字

字符串, 1236

坐

坐标系和参考点的位置, 785