凯恩帝Fi系列车削类/铣削类通用　高级编程手册　C051B-0402-W55N-0002

WWW.KND.COM.CN

V5.1　2022.04

凯恩帝Fi系列

车削类/铣削类通用
高级编程手册

北京凯恩帝数控技术有限责任公司

C051B-0402-W55N-0002
© KND CO., LTD.

说明书使用注意事项

在对本产品进行安装连接、编程和操作之前，必须详细阅读本说明书以及机床厂家提供的说明书，严格按照说明书的要求进行操作，否则可能导致产品损坏、工件报废甚至人身伤害。

本说明书描述的产品功能、技术指标仅针对本产品。安装了本产品的数控机床的实际功能和技术性能由机床厂家的设计决定，关于数控机床的具体指标请以机床厂家的说明书为准。

本说明书的内容可能随产品更新而变动，恕不另行通知。

本说明书适用于 KND MFi 钻、镗、铣床、加工中心、钻攻机及雕铣机等产品。说明书上全功能，具体有的功能，参见规格表或咨询机床厂

定制系统与标准系统的区别之处，请参见具体的《定制系统补充说明》。

本产品支持电子盘功能。机床调试完毕后，请将系统参数备份在电子盘中。当机床运行过程中发生数据紊乱或工作异常等情况时，读取电子盘操作可使系统快速恢复到备份前的状态。具体操作方法请参见操作篇内容。

本说明书的版权归“北京凯恩帝数控技术有限责任公司”所有。如有疑问，请垂询我公司技术人员，或使用以下沟通渠道联系我公司。

网址： http://www.knd.com.cn E-mail： bj@knd.com.cn
总机： 010-63701999 销售部： 010-63701981

产品使用注意事项

运输与储存

产品包装箱堆叠不可超过六层
不可在产品包装箱上攀爬、站立或放置重物
不可使用与产品相连的电缆拖动或搬运产品
严禁碰撞、划伤面板和显示屏
产品包装箱应避免潮湿、暴晒以及雨淋

开箱检查

打开包装后请确认是否是您所购买的产品
检查产品在运输途中是否有损坏
对照清单确认各部件是否齐全，有无损伤
如存在产品型号不符、缺少附件或运输损坏等情况，请及时与我公司联系

接线

参加接线与检查的人员必须是具有相应能力的专业人员
产品必须可靠接地，接地电阻应小于 4 欧姆，不能使用中性线（零线）代替地线
接线必须正确、牢固，以免导致产品故障或出现意想不到的后果
插拔插头或打开产品机箱前，必须切断产品电源

检修

检修或更换元器件前必须切断电源
发生短路或过载时应检查故障，故障排除后方可重新启动
不可对产品频繁通断电，断电后若须重新通电，相隔时间至少 1 分钟

目 录

I 编程篇 5

1 控制轴 7

1.1 变量指令轴地址符 7
1.2 轴地址替换功能（G54.1） 7
1.2.1 轴替换规则 7
1.2.2 笛卡尔平面更改 9
1.2.3 G、F 接口 10
1.3 比例控制（RC）功能（G351） 11
1.3.1 RC 控制器说明 11
1.3.2 诊断接口 13

2 插补功能 15

2.1 空间三点圆弧插补（G02.1） 15
2.2 螺线插补 18
2.2.1 平面等速螺线插补（G02.2，G03.2） 18
2.2.2 空间等速螺线插补（G02.3，G03.3） 21
2.2.3 平面等角螺线插补（G02.4，G03.4） 24
2.2.4 空间等角螺线插补（G02.5，G03.5） 26
2.3 极坐标插补（G12.1，G13.1） 28
2.4 圆柱插补（G07.1） 30
2.5 假想轴插补（G07） 33
2.6 数字主轴螺纹插补（G07.2） 35
2.7 数字主轴螺纹切削循环（G78.1） 38
2.8 主插补轴功能（G07.3） 43

3 进给功能 47

3.1 加减速控制 47
3.2 速度平滑控制 50

4 倾斜平面指定 52

5 坐标值和尺寸 53

5.1 比例缩放（G50，G51） 53
5.2 坐标旋转（G68.1，G69.1） 56

6 主轴比例控制（SPRC） 57

6.1.1 应用举例 61

7 斜轴功能 64

8 并行控制器(PEXC)功能 66

8.1 通道说明 67
8.2 PEXC 功能 67
8.3 PEXC 通道调用 70
8.3.1 调用方式 1：NC 调用(CP/CN/CL 指令) 70
8.3.2 调用方式 2：PLC PEXC 指令调用 71
8.3.3 调用方式 3：PLC G 接口调用 72
8.4 通道等待功能 73
8.4.1 G4.1 通道等待命令 73
8.4.2 G4.2 等待 PEXC 通道结束命令 74
8.5 PEXC 通道 S 代码支持 74
8.6 PEXC 通道 G/F 接口 75

9 机床补偿控制器 77

9.1 相关接口 77
9.1.1 控制接口 77
9.1.2 补偿平面 78
9.1.3 诊断接口 78
9.2 应用 79

10 程序 81

10.1 屏蔽子程序显示和屏蔽单段功能指令 81

11 刀具半径补偿的详细说明 84

12 用户宏程序（宏 B） 106

12.1 变量 106
12.2 基本数据类型 109
12.2.1 整型常量 109
12.2.2 实型常量 110
12.2.3 类型转换 111
12.3 系统变量 112
12.4 算术、逻辑和关系运算 116
12.5 加工程序语句 119
12.6 控制语句 119
12.6.1 无条件转移（GOTO 语句） 119
12.6.2 条件转移（IF 语句） 120

12.6.3 循环（WHILE 语句） 121
12.7 命令语句 122
12.7.1 字符串 122
12.7.2 MSG 命令 122
12.7.3 ERR 命令 123
12.7.4 OPENWRITE 命令 123
12.7.5 CLOSEWRITE 命令 123
12.7.6 WRITE 命令 124
12.8 宏程序调用 124
12.8.1 G65 宏程序调用 126
12.8.2 G66 模态调用 127
12.8.3 自定义 G 代码宏程序调用 129
12.8.4 自定义 M 代码宏程序调用 131
12.8.5 自定义 T 代码宏程序调用 132
12.8.6 CALL 调用 135
12.9 变量索引和数组 136
12.9.1 变量索引 136
12.9.2 数组 137
12.10 中断型用户宏程序 138
12.11 用户宏程序库及实数标准 139
12.11.1 用户宏程序数学库函数 139
12.11.2 函数功能说明 140
12.11.3 实型数与 IEEE754 标准 148
12.12 常见问题解释 150

Ⅰ　编程篇

1 控制轴

1.1 变量指令轴地址符

地址符：Pi = 进行绝对地址编程，Ri = 进行相对地址编程，其中 i=1~8，代表空间内的 8 个轴

注

1 铣床和车床 B 体系时主通道不支持 Ri 指定

2 Pi=，Ri=中间不能有空格

例

P1=0;P2=0 绝对运动

R1=100,R2=100 相对运动

主通道和 PEXC 通道可以使用宏变量#9000~#9007 对空间内第 1~8 轴的轴地址进行重映射，#9000~#9007 在各空间是独立的，不相互影响，设为 0 的轴不能指定，未设定的轴不进行重映射。

例

#9000=4;#9001=5#9002=0;	空间内第 1、2 轴映射到系统的第 4、5 轴，关闭第 3 轴
P1=0;P2=0;	4、5 轴绝对运动
R1=100,R2=100;	4、5 轴相对运动
P3=0;	报警：非法地址符

1.2 轴地址替换功能（G54.1）

当由 CAM 软件生成的程序，期望在不修改程序的情况下应该到其他轴上时，可以通过 G54.1 功能实现轴地址替换。

格式

G54.1 P0~8;

P1~8：使用参数 P422-429 定义的轴替换规则

G54.1 或 G54.1 P0：关闭轴替换功能

注意事项

注

1 G54.1 不能与 G54.9 同时使用；

2 G54.1 应单独一行；

3 指令 G54.1 时，长度半径补偿，坐标变换，恒限速功能应处于无效状态；

4 指令 G54.1 后，G17、G18、G19 平面内轴可能发生改变；

5 指令 G54.1 时会取消补偿并刷新坐标，请确保 G54.1 后第一个运动段为绝对坐标指令。

1.2.1 轴替换规则

系统提供了 8 个替换规则参数 P422~P429。替换规则需满足如下条件：

1 需要替换的轴用“X=XA”的形式，以‘,’进行分割，未在规则中指定的轴不能进行运动；

例：X，Y=YA，Z；

即规则定义了：Y 替换为 YA 轴，且只有 X，Y，Z 轴可以运动；

1 等号左侧字符可以是用户绝对轴名或非轴地址的单字母字符（X,Y,Z,A,B,C,U,V,W）；

2 等号右侧字符必须是有效的用户轴名，可以是绝对轴名，仅当左侧字符是非轴地址字符时可以是相对轴名；.

3 以‘,’进行分割，不可重复指定相同轴地址；

4 ‘=’号两侧轴对应轴类型需一致，即同为直线轴或旋转轴；

5 如果笛卡尔坐标系没有第三轴，可以用-1 代替。

2 表达式前三项视为笛卡尔坐标系基本轴，如果要指定基本轴之外的替换规则，必须先指定基本轴，再指定替换规则；

例 1：X，Y=YA，Z,

笛卡尔坐标系三个基本轴决定了平面内的轴，默认值为:X-Y-Z，经过替换规则处理后，笛卡尔坐标系基本轴替换为：X-YA-Z

例 2：X，Z，Y，H=HX （注：按轴顺序编写）

前三项基本轴配置同默认配置，表明笛卡尔坐标系不发生变化即 XYZ;（车床笛卡尔坐标系依然是 XYZ，只是习惯按轴顺序写做 XZY，系统内部会自动交换为 XYZ）

例 3：X=XB，Z=ZB，-1，U=UB，V=VB;

P420：XA、ZA、XB、ZB；P421：UA、VA、UB、VB

定义了车床 G18 平面的两个轴 XB、ZB，其余平面无效不能指定；同时定义了绝对坐标和相对坐标替换规则；

3 轴地址不能重复指定；

举例

示例 1

铣床系统轴名配置为：X，Y，Z，XA，YA，ZA，C 共 7 个轴

定义规则 1(P422)：X=XA，Y=YA，Z=ZA，则：

```
G54.1 P1;
G00 G90   X0 Y0 Z0;          XA，YA，ZA 轴绝对坐标移动到 0
X100 Y100 Z100;              XA,YA,ZA 轴移动到 100.
G91 X100 Y100 Z100;          XA，YA，ZA 轴移动到 200.
```

示例 2

车床系统轴名配置为：X，ZA，Y，ZB，ZC，C 共 5 个轴，相对地址为 U，WA，V，WB，WC，B

定义规则 1(P422)：X，Z=ZA，Y，W=WA，其中 Z，W 不是系统类有效轴名，ZA 为第二轴绝对轴地址，WA 为第二轴相对轴地址，则：

```
G54.1 P1;                    笛卡尔坐标系被设定为 X-Y-ZA;
G00 Z100;                    ZA 轴绝对坐标移动到 100.;
W100;                        ZA 轴相对坐标移动 100.，绝对坐标变为 200.
```

1.2.2 笛卡尔平面更改

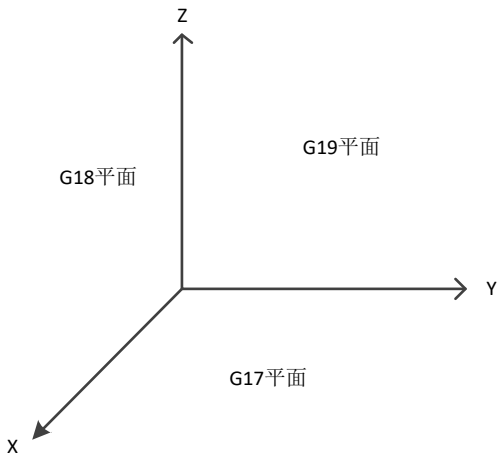
根据替换规则，G54.1 可能会更改笛卡尔坐标系内的坐标轴，即 G17、G18、G19 平面内轴发生改变。

例

铣床系统轴配置为：X、Y、Z、XA、YA、ZA；

车床系统轴配置为：X、Z、Y、XA、ZA、YA；

上电时，各平面内轴情况（注：车床的默认笛卡尔坐标系依然时 XYZ，只是习惯按轴顺序写为 XZY，系统内部会进行交换处理）：

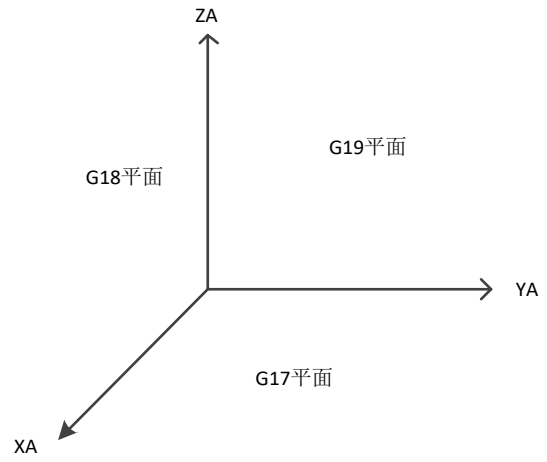


设参数 P422 规则为：

铣床：X=XA，Y=YA，Z=ZA

车床：X=XA，Z=ZA，Y=YA

执行 G54.1 P1 后，笛卡尔坐标系变更为：



则编程 G02 I100 J100；将在 XA-YA 平面绘制圆;

1.2.3 G、F 接口

G141								G142EN

G142EN： 1：使用 G142 接口指定轴交换规则

G142	轴交换规则选择：0~8

F142	G54.1 当前选择规则，范围[0，8]

注

1 G142 接口设定后，非运行状态（F0.7 = 0）立即起效；

2 G 接口控制与 G54.1 P 指令应互斥。

1.3 比例控制（RC）功能（G351）

比例控制器（RC：RC1~RC4），用于控制从动轴，使其跟随主动轴按编程比例转动。

格式

- **启动 RC 控制器**

G351 NLnl NPnp Pp Qq；

[说明]

nl：选择 RC 控制器。有效数值：1~4，对应 RC1~RC4。

未编入默认 RC1 控制器。

np：轴号选择。

十进制最高位：主动轴号（1~最大轴数）

十进制其余位：从动轴号（1~最大轴数），最多同时指定七个从动轴

注 1：主动轴与某个从动轴相同时，为对该轴进行缩放控制

注 2：多个从动轴时，与顺序无关。例：NP1234 与 NP1432 等价。

p/q：主动轴和从动轴的速度比。从动轴速度=主动轴速度*q/p。

p 应为正数(>=1)，q 应不为 0(>0 或<0)，pq 支持小数。

P 未编入时，默认 p=1

Q 未编入时，默认 q=1。q 为负值代表从动轴与主动轴移动方向相反。

- **停止 RC 控制器**

G350 NLnl Pp ；

[说明]

nl：选择 RC 控制器。有效数值：1~4，对应 RC1~RC4。

未编入默认同时停止 RC1~RC4。

p：发出停止信号，并等待 RC 控制器停止

p 有效范围：1

P 未编入时，向 RC 控制器发出减速停止信号，但不等待其停止结束

编入 P1 时，向 RC 控制器发出减速停止信号，并等待其停止结束

注

G351/G350 指令，主通道和 PEXC 通道都可使用。

1.3.1 RC 控制器说明

坐标更新

RC 控制期间，从动轴跟随主动轴运动的部分脉冲，计入机床坐标，不计入绝对坐标。但累计未计入部分，在 RC 控制结束时更新绝对坐标。

在 RC 控制器启动时刻，当从动轴和主动轴不是同一个轴时，从动轴绝对坐标清 0；并在停止时刻，

将 RC 控制期间未计入绝对坐标的数据总和计入绝对坐标。

当主动轴和从动轴为同一个轴时，无绝对坐标清 0 过程。

RC 启动说明

1 启动后系统在标题栏中间以红色字体醒目提示工作中的 RC 控制器：RC1~4

2 启动后，如果主动轴或从动轴为主轴，且处于转速控制模式，那么立即进入轮廓控制模式

例如：

配置	第 1 轴为主动轴(X 轴)，第 4 轴为从动轴(C 轴，为第 1 主轴)
程序	G351 NP14； 1:1 控制，C 轴进入轮廓模式，坐标清零 … G350 ； 虽停止 RC 控制器，但 C 轴仍处于轮廓模式
其它说明	G51.3 与 G351 等价，G50.3 与 G350 等价，下同

3 处于 RC 控制的轴，具有以下限制

(1) G27~G30/G53/G153 中编入从动轴轴地址时报警

例如：

配置	第 1 轴为从动轴(X 轴)
程序	G351 NP21； 1:1 控制，X 轴为从动轴 G28 X0 ； 报警

(2) 转动主轴时，如果主轴为主动轴或从动轴时报警

例如：

配置	第 4 轴为从动轴，也是第 1 主轴(C 轴)
程序	G351 NP14； 1:1 控制，X 轴为从动轴 M3 S200 ； 报警

4 从动轴有加减速控制时(RCi-SAAD=1)时，当 RC 启动控制后，可再次指定 G351，给入新的控制器数据。

例如：

配置	第 1 轴为主动轴(X 轴)，第 2 轴为从动轴(Y 轴)
程序	G351 NP12 ； … G351 NP12 Q2 ； … G350 ；

RC 停止说明

1 通过代码停止

例如：

配置	第 1 轴为主动轴(X 轴)，第 2 轴为从动轴(Z 轴)
程序	G0 X70. Z0. ； G351 NP12 P2 Q-1 ； … G350 ； 停止 RC 控制器，但并不等待其结束(下一段无主动轴和从动轴

	运动) Y50. ； G350 P1 ； 等待 RC 控制器结束(下一段有主动轴或从动轴运动需等待) X50. ； …

2 通过 G 接口停止

PLC 可在复位时接通 RCi-RST(i=1~4)，这样复位时就可以停止相应 RC 控制器。

3 主动或从动轴报警时，停止 RC 控制器。

从动轴软限位说明

从动轴不检查软限位

1.3.2 诊断接口

F/G 区接口

G1260	RC1-TO4	RC1-TO3	RC1-TO2	RC1-TO1			RC1-SAAD	RC1-RST
G1261	RC2-TO4	RC2-TO3	RC2-TO2	RC2-TO1			RC2-SAAD	RC2-RST
G1262	RC3-TO4	RC3-TO3	RC3-TO2	RC3-TO1			RC3-SAAD	RC3-RST
G1263	RC4-TO4	RC4-TO3	RC4-TO2	RC4-TO1			RC4-SAAD	RC4-RST

RCi-TO1~ RCi-TO4：(i=1~4)

RC 控制器启动超时秒数。数值 0 为 0.5s。

RCi-RST：(i=1~4)

1—(RCi)停止控制

注

如果给入该信号，那么应该保持为 1，直到 F126i.0(RCi-CTL)为 0。

RCi-SAAD：(i=1~4)

1/0—(RCi)从动轴有/无加减速控制

注

1 启动 RC 控制器时，如果检测到该位为 1，并且此时主动轴还在运动，那么报警“RC 错误(G351)：无加减速(RCi-SAAD=0,i=1~4)，需在控制停止后指令”。

2 启动 RC 控制器时，如果检测到该位为 0，那么停止(G350 或 RCi-RST=1)时，会等到主动轴停止时才停止 RC 控制器。

F1260							RC1-SYNC	RC1-CTL
F1261	RC1-MA8	RC1-MA7	RC1-MA6	RC1-MA5	RC1-MA4	RC1-MA3	RC1-MA2	RC1-MA1
F1262	RC1-SA8	RC1-SA7	RC1-SA6	RC1-SA5	RC1-SA4	RC1-SA3	RC1-SA2	RC1-SA1

RCi-CTL：(i=1~4)

1—(RCi)控制中

RCi-SYNC：(i=1~4)

1—(RCi)从动轴跟随中(不包括控制启动时加速过程和停止时减速过程)

RCi-MAj：(i=1~4) (j=1~8)

1—(RCi)主动轴控制标志（每轴一位）

RCi-SAj：(i=1~4) (j=1~8)

1—(RCi)从动轴控制标志（每轴一位）

F1265~F1277 为 RC2~RC4 控制器 F 诊断位

2 插补功能

2.1 空间三点圆弧插补（G02.1）

通过指定圆弧中间点和终点，可以在三维空间内进行圆弧插补。属于 01 组模态 G 代码。

格式

G02.1 NI(I)_ NJ(J)_ NK(K)_ Xp_ Yp_ Zp_ NIP_ F_ ;

NI_NJ_NK_：

圆弧中间点 Q 的绝对坐标值，分别对应笛卡尔坐标系的 Xp，Yp 和 Zp 轴。

I_ J_ K_：

圆弧中间点 Q 相对于起点 P 的增量坐标（即：矢量 PQ）。I、J 和 K 分别对应于笛卡尔坐标系 Xp，Yp 和 Zp 轴。只能用增量指定。

Xp_ Yp_ Zp_：

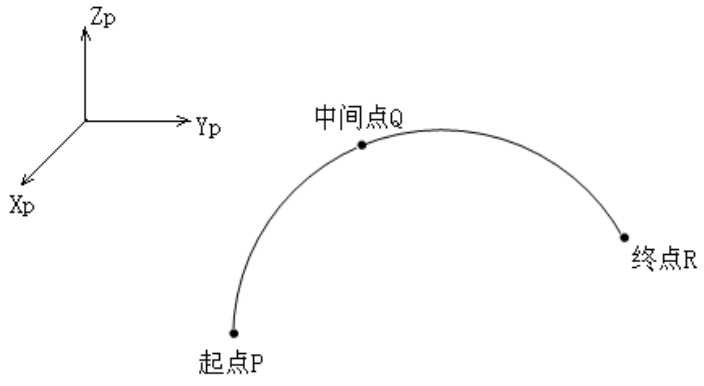
圆弧终点坐标 R。可增量指定或绝对指定。

NIP_：

笛卡尔坐标系外轴指令（非 Xp，Yp 和 Zp 轴）。插补时，与圆弧插补同时开始同时结束。可增量指定或绝对指定。

F_：

进给速度。



解释

- **半径补偿**

不支持半径补偿，如果半径补偿有效时指令了空间圆弧插补指令，那么报警（PS0048）。

- **缩放和旋转**

可以进行缩放和旋转。

对缩放的处理与圆弧插补类似，即仅针对指令向量进行缩放，而不是轨迹本身。

- **任意轴插补机能**

通过指令 G54.9 Pijk，可变更插补轴。此时，X_P、Y_P、Z_P 分别为 G54.9 设定的第 1/2/3 个轴。

注

1 当起点 P、中间点 Q、终点 R 三点共线（包括存在两点重合的情况）时，改走直线 PR（忽略中间点）。

2 无法用单条指令指定整圆。

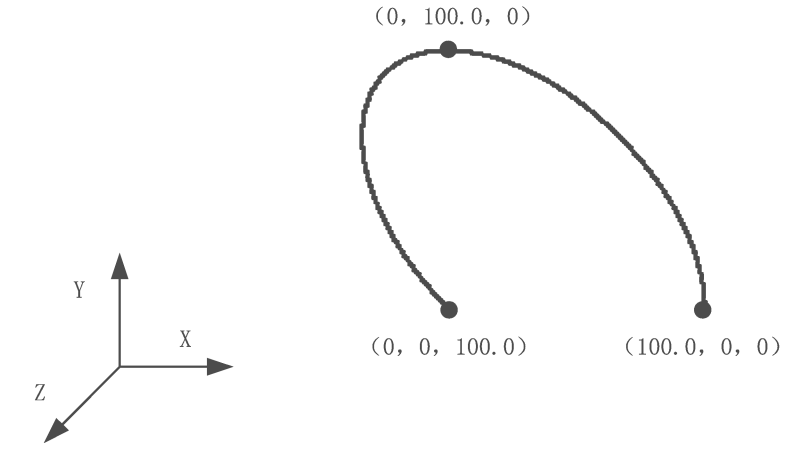
3 G102 与 G02.1 等价。

4 不建议在极坐标变换下（G16），使用空间圆弧插补。当 G16 有效时，空间圆弧中间点的 I、J、K 不会变换。

举例

以下示例列举一些特殊使用场景。

- **一般的空间圆弧插补**

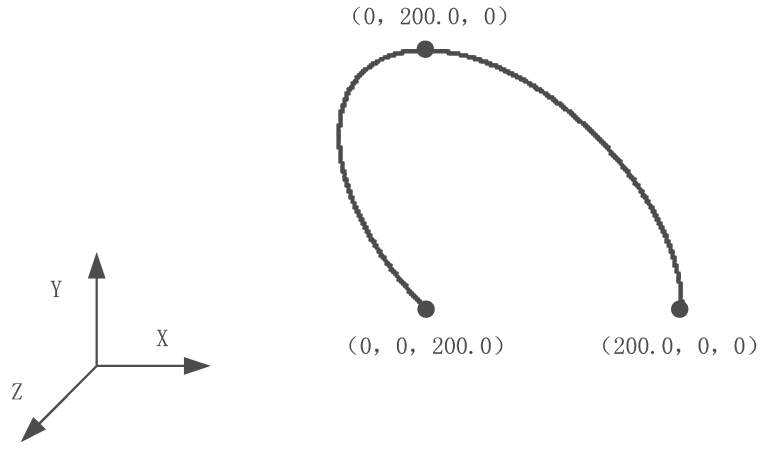


过（100.0，0，0）、（0，100.0，0）、（0，0，100.0）三点的弧。

加工程序

```
G90 G00 X100. Y0 Z0 ;
G02.1 I-100. J100. X0 Y0 Z100. F500. ;
```

- **比例缩放有效时的空间圆弧插补**

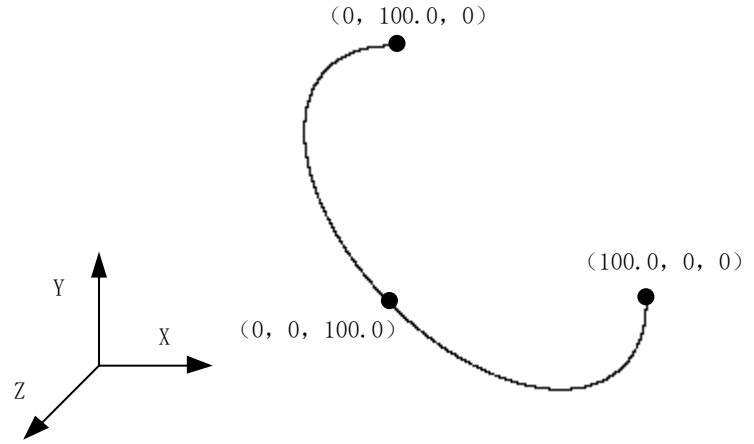


过（100.0，0，0）、（0，100.0，0）、（0，0，100.0）三点的弧，以（0，0，0）为中心放大 2 倍。等价于过（200.0，0，0）、（0，200.0，0）、（0，0，200.0）三点的弧。

加工程序

```
G90 G00 X0 Y0 Z0 ;
G51 X0 Y0 Z0 P2000 ;
G00 X100. Y0 Z0 ;
G02.1 I-100. J100. X0 Y0 Z100. F500. ;
G00 X0 Y0 Z0 ;
G50 ;
```

- **坐标旋转有效时的空间圆弧插补**



过（100.0，0，0）、（0，100.0，0）、（0，0，100.0）三点的弧，绕（0，0，0）到（1.0，1.0，1.0）点的直线旋转 120 度。等价于过（0，100.0，0）、（0，0，100.0）、（100.0，0，0）三点的弧。

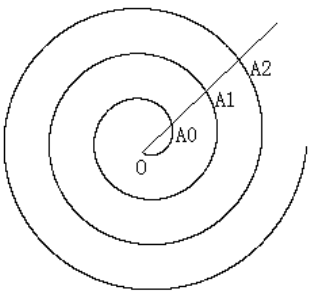
加工程序

```
G90 G00 X0 Y0 Z0 ;
G68 X0 Y0 Z0 I1. J1. K1. R120. ;
G00 X100. Y0 Z0 ;
G02.1 I-100. J100. X0 Y0 Z100. F500. ;
G00 X0 Y0 Z0 ;
G69 ;
```

2.2 螺线插补

2.2.1 平面等速螺线插补（G02.2，G03.2）

一动点以常速 v 沿一射线运动，而这一射线又以定角速度 ω 绕极点 O 转动时，该动点所描成的轨迹为“阿基米德螺线”，亦称“等速螺线”。如下图所示。



等速螺线具有等距性特点：过极点的射线与曲线交于 A0、A1、A2……，它们是等间距的。（当常速 v =0 时的等速螺线为圆弧。）

格式

- **格式 1：起点或终点为极点**

XY 平面的等速螺线

G17 $\begin{Bmatrix} G02.2 \\ G03.2 \end{Bmatrix}$ Xp_ Yp_ NIP_ P_ Q_ F_ ；

ZX 平面的等速螺线

G18 $\begin{Bmatrix} G02.2 \\ G03.2 \end{Bmatrix}$ Zp_ Xp_ NIP_ P_ Q_ F_ ；

YZ 平面的等速螺线

G19 $\begin{Bmatrix} G02.2 \\ G03.2 \end{Bmatrix}$ Yp_ Zp_ NIP_ P_ Q_ F_ ；

G17/G18/G19：平面指定。

G02.2/G03.2：G02.2 为顺时针等速螺线，G03.2 为逆时针等速螺线。两者为 01 组模态 G 代码。

Xp_Yp_/Zp_Xp_/Yp_Zp_：

G17/G18/G19 平面内等速螺线的终点（增量编程或绝对编程）。若起点与终点重合，则不运动，并产生报警（PS0272）。

NIP_：平面外轴指令。插补时，与平面内等速螺线插补同时开始同时结束。

P_：0-起点为极点，1-终点为极点，其它-报警（PS0268）。P0 可以不输入。

Q_：等速螺线绕极点转过的角度（Q>0，否则报警（PS0267）），Q 必须输入。

F_：进给速度。

- **格式 2：不包含极点**

XY 平面的等速螺线

G17 $\begin{Bmatrix} G02.2 \\ G03.2 \end{Bmatrix}$ Xp_ Yp_ I_ J_ NIP_ P_ F_ ；

ZX 平面的等速螺线

G18 $\begin{Bmatrix} G02.2 \\ G03.2 \end{Bmatrix}$ Zp_ Xp_ K_ I_ NIP_ P_ F_ ；

YZ 平面的等速螺线

G19 $\begin{Bmatrix} G02.2 \\ G03.2 \end{Bmatrix}$ Yp_ Zp_ J_ K_ NIP_ P_ F_ ；

G17/G18/G19：平面指定。

G02.2/G03.2： G02.2 为顺时针等速螺线，G03.2 为逆时针等速螺线。两者为 01 组模态 G 代码。

Xp_Yp_/Zp_Xp_/Yp_Zp_：

G17/G18/G19 平面内等速螺线的终点（增量编程或绝对编程）。当终点和起点重合时，系统按照圆弧进行插补，角度为$(p+1)\times 360$度。

I_J_/K_I_/J_K_：G17/G18/G19 平面内等速螺线的极点相对于起点的矢量（增量编程）。

NIP_：平面外轴指令。插补时，与平面内等速螺线插补同时开始同时结束。

P_：附加整圈数（<=10000000，超出报警（PS0117））。

F_：进给速度。

解释

- **半径补偿**

不支持半径补偿，如果半径补偿有效时指令了平面等速螺线插补指令，那么报警（PS0048）。

- **格式区分**

系统根据指令 Q，区分指令格式 1 和指令格式 2。

- **缩放和旋转**

进行缩放和旋转后，系统改用空间等速螺线插补。

对缩放的处理与圆弧插补类似，即仅针对指令向量进行缩放，而不是轨迹本身。

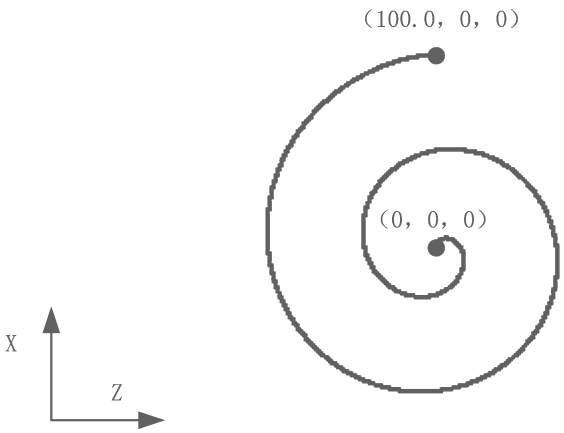
- **任意轴插补机能**

通过指令 G54.9 Pijk，可变更插补轴。此时， X_P 、 Y_P 、 Z_P 分别为 G54.9 设定的第 1/2/3 个轴。

举例

以下示例列举一些特殊使用场景。

- **格式 1**

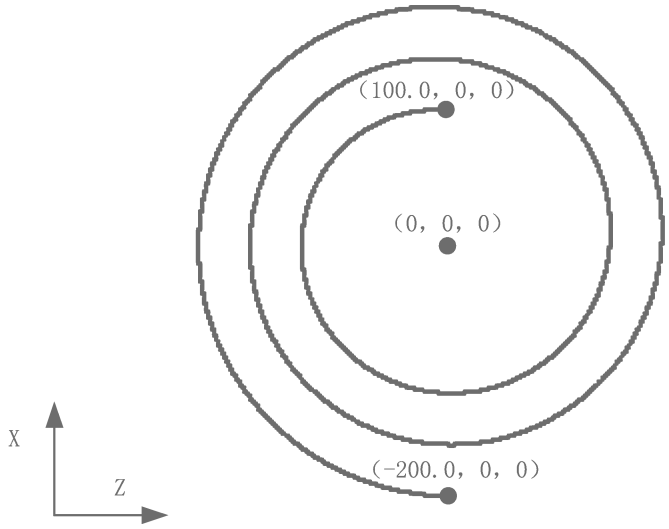


在 G18 平面（ZX 平面）上等速螺线起点为（100.0，0，0），逆时针（G03.2）转过 720 度，到达极点（0，0，0）。然后由极点顺时针（G02.2）反向运动到起点。

加工程序

```
G90 G00 X100.0 Y0 Z0 ;
G18 G03.2 X0 Q720. P1 F500.          ;运动至极点
G18 G02.2 X100. Q720. P0 F500.       ;由极点反向运动至起点
```

- **格式 2**



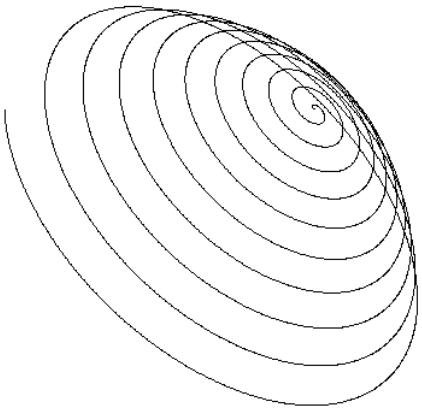
在 G18 平面（ZX 平面）上等速螺线起点为（100.0，0，0），以点（0，0，0）为极点，逆时针（G03.2）转动 2 圈半，到达终点（-200.0，0，0）。然后再顺时针（G02.2）反向运动到起点。

加工程序

```
G90 G00 X100.0 Y0 Z0 ;
G18 G03.2 X-200. I-100.0 P2 F500. ;
G18 G02.2 X100. I200.0 P2 F500.          ;上一段的反向运动
```

2.2.2 空间等速螺线插补（G02.3，G03.3）

将平面等速螺线所在平面旋转，得到空间等速螺线。空间等速螺线仍然在一个平面内，但该平面不一定是基本坐标平面(G17/G18/G19 平面)，需要指定该平面的法矢量。空间等速螺线为平面等速螺线在空间中的扩展。如下图所示。



当螺线与其所在平面的法矢量呈右手螺旋法则时，螺线为逆时针方向，否则为顺时针方向。

格式

- **格式 1：起点或终点为极点**

$$\begin{cases} NX_ \ NY_ \ NZ_ \\ NI_ \ NJ_ \ NK_ \end{cases} \begin{cases} G02.3 \\ G03.3 \end{cases} Xp_ \ Yp_ \ Zp_ \ NIP_ \ P_ \ Q_ \ F_ \ ;$$

$\begin{cases} NX_ \ NY_ \ NZ_ \\ NI_ \ NJ_ \ NK_ \end{cases}$：

指定空间等速螺线所在平面的法矢量（长度任意，应尽量提高有效位数）。其中，NX_ NY_ NZ_为一次性指定，而 NI_ NJ_ NK_为模态指定，当无一次性指定时，模态值有效。

G02.3/G03.3：

G02.3 为顺时针空间等速螺线，G03.3 为逆时针空间等速螺线。两者为 01 组模态 G 代码。

Xp_ Yp_ Zp_ ：

空间等速螺线的终点（增量编程或绝对编程）。起点和法矢量可确定空间等速螺线所在的平面。当等速螺线终点不在该平面上时，系统将终点向该平面投影得到该平面内的一条螺线，并附加从投影点到终点的直线运动。插补时，该附加直线与平面内等速螺线插补同时开始同时结束。若起点与终点重合，则不运动，并产生报警（PS0272）。

NIP_：非空间基本坐标轴指令。插补时，与空间等速螺线插补同时开始同时结束。

P_：0-起点为极点，1-终点为极点，其它-报警（PS0268）。P0 可以不输入。

Q_：等速螺线绕极点转过的角度（Q>0，否则报警（PS0267）），Q 必须输入。

F_：进给速度。

- **格式 2：不包含极点**

$$\begin{cases} NX_ \ NY_ \ NZ_ \\ NI_ \ NJ_ \ NK_ \end{cases} \begin{cases} G02.3 \\ G03.3 \end{cases} Xp_ \ Yp_ \ Zp_ \ I_ \ J_ \ K_ \ NIP_ \ P_ \ F_ \ ;$$

$\begin{cases} NX_ \ NY_ \ NZ_ \\ NI_ \ NJ_ \ NK_ \end{cases}$：

指定空间等速螺线所在平面的法矢量（长度任意，应尽量提高有效位数）。其中，NX_ NY_ NZ_为一次性指定，而 NI_ NJ_ NK_为模态指定，当无一次性指定时，模态值有效。

G02.3/G03.3：

G02.3 为顺时针空间等速螺线，G03.3 为逆时针空间等速螺线。两者为 01 组模态 G 代码。

Xp_ Yp_ Zp_ ：

空间等速螺线的终点（增量编程或绝对编程）。起点和法矢量可确定空间等速螺线所在的平面。当等速螺线终点不在该平面上时，系统将终点向该平面投影得到该平面内的一条螺线，并附加从投影点到终点的直线运动。插补时，该附加直线与平面内等速螺线插补同时开始同时结束。当终点和起点重合时，系统按照圆弧进行插补，角度为$(p+1)\times 360$度。

I_ J_ K_：

等速螺线的极点相对于起点的矢量（增量编程）。极点应该在螺线平面上。当极点距螺线平面距离不超过参数“螺线中心距螺线平面距离的允许偏差量”（P6449）时，系统将其在螺线平面上的投影点作为修正后的极点，否则系统报警（PS0275）。修正后的极点相对于起点的矢量不能为 0，否则报警（PS0263）。

NIP_：非空间基本坐标轴指令。插补时，与空间等速螺线插补同时开始同时结束。

P_：附加整圈数（<=10000000，超出报警（PS0117））。

F_：进给速度。

解释

- **半径补偿**

  不支持半径补偿，如果半径补偿有效时指令了空间等速螺线插补指令，那么报警（PS0048）。

- **格式区分**

  系统根据指令 Q，区分指令格式 1 和指令格式 2。

- **缩放和旋转**

  可以进行缩放和旋转。

  对缩放的处理与圆弧插补类似，即仅针对指令向量进行缩放，而不是轨迹本身。

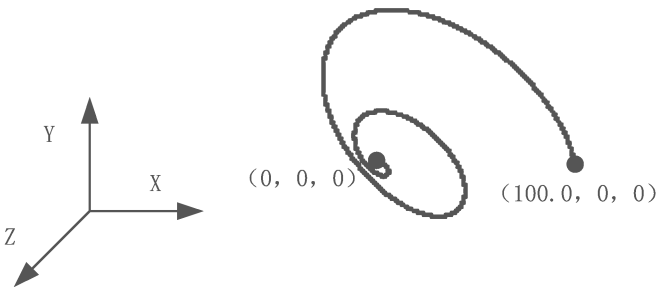
- **任意轴插补机能**

  通过指令 G54.9 Pijk，可变更插补轴。此时，X_P、Y_P、Z_P 分别为 G54.9 设定的第 1/2/3 个轴。

举例

以下示例列举一些特殊使用场景。

- **格式 1**

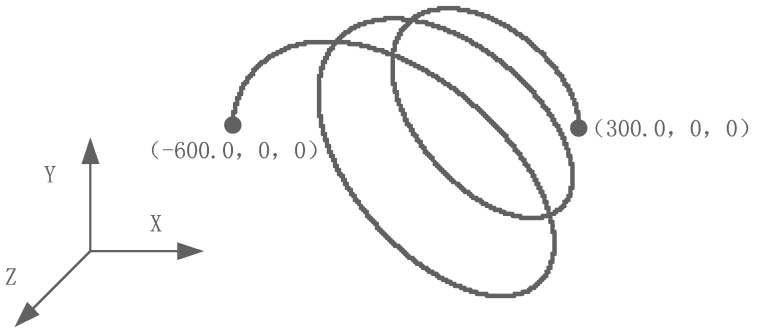


空间等速螺线起点为（100.0，0，0），所在平面法矢量为（1.0，1.0，1.0），逆时针（G03.3）转过 720 度，到达极点（0，0，0）。然后由极点顺时针（G02.3）反向运动到起点。

加工程序

```
G90 G00 X100. Y0 Z0 ;
NX1. NY1. NZ1. G03.3 X0 Q720. P1 F500.          ;运动至极点
NX1. NY1. NZ1. G02.3 X100. Q720. P0 F500.       ;由极点反向运动至起点
```

- **格式 2**



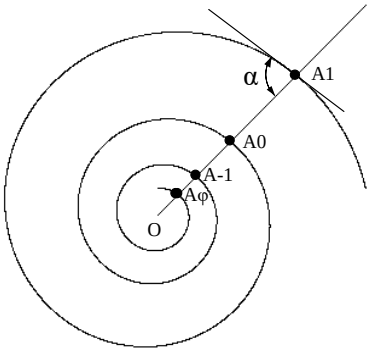
空间等速螺线起点为（300.0，0，0），所在平面法矢量为（1.0，1.0，1.0），以点（100.0，100.0，100.0）为极点，逆时针（G03.3）转动 2 圈半，到达终点（-600.0，0，0）。然后再顺时针（G02.3）反向运动到起点。

加工程序

```
G90 G00 X300. Y0 Z0 ;
NX1. NY1. NZ1. G03.3 X-600. I-200. J100. K100. P2 F500. ;
NX1. NY1. NZ1. G02.3 X300. I400. J-200. K-200. P2 F500.        ;上一段的反向运动
```

2.2.3 平面等角螺线插补（G02.4，G03.4）

曲线与所有过极点的射线的交角都相等（皆为α，$k=ctg\alpha$），当$\varphi\to-\infty$时，曲线沿顺时针或逆时针方向绕极点转动而趋于极点（但无法到达极点）。此为“对数螺线”，亦称“等角螺线”。如下图所示。



等角螺线具有等比性特点：过极点的射线与曲线交于……、A_{-1}、A_0、A_1、……，则……、OA_{-1}、OA_0、OA_1、……各线段成等比级数（公比为$e^{2k\pi}$）。（当$\alpha=\frac{\pi}{2}$时为圆弧）

格式

XY 平面的等角螺线

G17 $\begin{Bmatrix} G02.4 \\ G03.4 \end{Bmatrix}$ Xp_ Yp_ I_ J_ NIP_ P_ F_ ；

ZX 平面的等角螺线

G18 $\begin{Bmatrix} G02.4 \\ G03.4 \end{Bmatrix}$ Zp_ Xp_ K_ I_ NIP_ P_ F_ ；

YZ 平面的等角螺线

G19 $\begin{Bmatrix} G02.4 \\ G03.4 \end{Bmatrix}$ Yp_ Zp_ J_ K_ NIP_ P_ F_ ；

G17/G18/G19：平面指定。

G02.4/G03.4：G02.4 为顺时针等角螺线，G03.4 为逆时针等角螺线。两者为 01 组模态 G 代码。

Xp_Yp_/Zp_Xp_/Yp_Zp_：

G17/G18/G19 平面内等角螺线的终点（增量编程或绝对编程）。当终点和起点重合时，系统按照圆弧进行插补，角度为$(p+1)\times 360$度。如果起点到极点距离与终点到极点距离相同，那么此时$\alpha = \pi/2$，轨迹为圆弧。

I_J_/K_I_/J_K_：G17/G18/G19 平面内等角螺线的极点相对于起点的矢量（增量编程）。

NIP_：平面外轴指令。插补时，与平面内等角螺线插补同时开始同时结束。

P_：附加整圈数（<=10000000，超出报警（PS0117））。

F_：进给速度。

解释

- **半径补偿**

不支持半径补偿，如果半径补偿有效时指令了平面等角螺线插补指令，那么报警（PS0048）。

- **缩放和旋转**

进行缩放和旋转后，系统改用空间等角螺线插补。

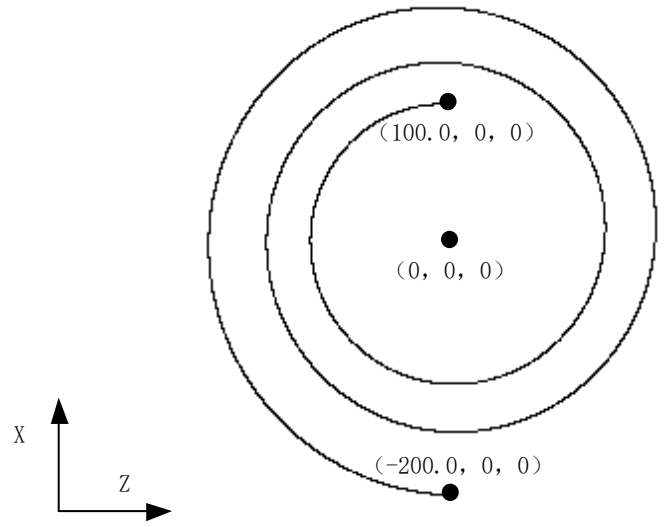
对缩放的处理与圆弧插补类似，即仅针对指令向量进行缩放，而不是轨迹本身。

- **任意轴插补机能**

通过指令 G54.9 Pijk，可变更插补轴。此时，Xp，Yp，Zp 分别为 G54.9 设定的第 1/2/3 个轴。

举例

以下示例列举一些特殊使用场景。



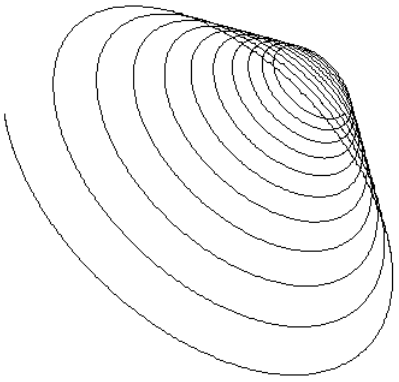
在 G18 平面（ZX 平面）上等角螺线起点为（100.0，0，0），以点（0，0，0）为极点，逆时针（G03.4）转动 2 圈半，到达终点（-200.0，0，0）。然后再顺时针（G02.4）反向运动到起点。

加工程序

```
G90 G00 X100. Y0 Z0 ;
G18 G03.4 X-200. I-100. P2 F500. ;
G18 G02.4 X100. I200. P2 F500.          ;上一段的反向运动
```

2.2.4 空间等角螺线插补（G02.5，G03.5）

将平面等角螺线所在平面旋转，得到空间等角螺线。空间等角螺线仍然在一个平面内，但该平面不一定是基本坐标平面（G17/G18/G19 平面），需要指定该平面的法矢量。空间等角螺线为平面等角螺线在空间中的扩展。如下图所示。



当螺线与其所在平面的法矢量呈右手螺旋法则时，螺线为逆时针方向，否则为顺时针方向。

格式

$\begin{cases} \mathbf{NX_\ \ NY_\ \ NZ_} \\ \boldsymbol{NI}\mathbf{_\ \ NJ_\ \ NK_} \end{cases} \begin{cases} \boldsymbol{G02.5} \\ \boldsymbol{G03.5} \end{cases} \mathbf{Xp_\ \ Yp_\ \ Zp_\ \ I_\ \ J_\ \ K_\ \ NIP_\ \ P_\ \ F_\ \ ;}$

$\begin{cases} \mathbf{NX_\ \ NY_\ \ NZ_} \\ \boldsymbol{NI}\mathbf{_\ \ NJ_\ \ NK_} \end{cases}$：指定空间等角螺线所在平面的法矢量（长度任意，应尽量提高有效位数）。

其中，NX_NY_NZ_为一次性指定，而 NI_NJ_NK_为模态值，当无一次性指定时，模态值有效。

G02.5/G03.5：G02.5 为顺时针空间等角螺线，G03.5 为逆时针空间等角螺线。两者为 01 组模态 G 代码。

Xp_Yp_Zp_：

空间等角螺线的终点（增量编程或绝对编程）。起点和法矢量可确定空间等角螺线所在的平面。当等角螺线终点不在该平面上时，系统将终点向该平面投影得到该平面内的一条螺线，并附加从投影点到终点的直线运动。插补时，该附加直线与平面内等角螺线插补同时开始同时结束。当终点和起点重合时，系统按照圆弧进行插补，角度为$(\mathrm{p}+1)\times 360$度。

I_J_K_：

等角螺线的极点相对于起点的矢量（增量编程）。极点应该在螺线平面上。当极点距螺线平面距离不超过参数“螺线中心距螺线平面距离的允许偏差量”（P6449）时，系统将其在螺线平面上的投影点作为修正后的极点，否则系统报警（PS0275）。修正后的极点相对于起点的矢量不能为 0，否则报警（PS0263）。

NIP_：非空间基本坐标轴指令。插补时，与空间等角螺线插补同时开始同时结束。

P_：附加整圈数（<=10000000，超出报警（PS0117））。

F_：进给速度。

解释

- **半径补偿**

不支持半径补偿，如果半径补偿有效时指令了空间等角螺线插补指令，那么报警（PS0048）。

- **缩放和旋转**

可以进行缩放和旋转。

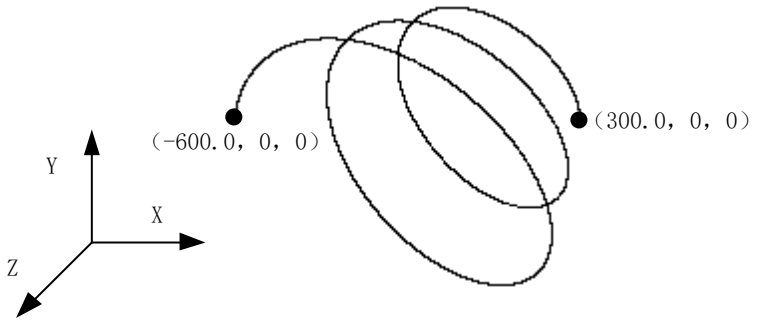
对缩放的处理与圆弧插补类似，即仅针对指令向量进行缩放，而不是轨迹本身。

- **任意轴插补机能**

通过指令 G54.9 Pijk，可变更插补轴。此时，X_P、Y_P、Z_P 分别为 G54.9 设定的第 1/2/3 个轴。

举例

以下示例列举一些特殊使用场景。



空间等角螺线起点为（300.0，0，0），所在平面法矢量为（1.0，1.0，1.0），以点（100.0，100.0，100.0）为极点，逆时针（G03.2）转动 2 圈半，到达终点（-600.0，0，0）。然后再顺时针（G02.2）反向运动到起点。

加工程序

```
G90 G00 X300. Y0 Z0 ;
NX1. NY1. NZ1. G03.5 X-600. I-200. J100. K100. P2 F500. ;
NX1. NY1. NZ1. G02.5 X300. I400. J-200. K-200. P2 F500.          ;上一段的反向运动
```

2.3 极坐标插补（G12.1，G13.1）

极坐标插补是一种轮廓控制。它把笛卡尔坐标系内的编程指令转换为直线轴的移动（刀具移动）和旋转轴的移动（工件旋转）。这种方法可用于凸轮磨削。

格式

G12.1 Pij ; **启动极坐标插补**

……

G13.1 ; **取消极坐标插补**

G12.1 和 G13.1 由单程序段指定，也可分别以 G112 和 G113 代之。

i 为直线轴轴号，j 为旋转轴轴号，数值为 1~8，分别对应第 1~8 轴。

解释

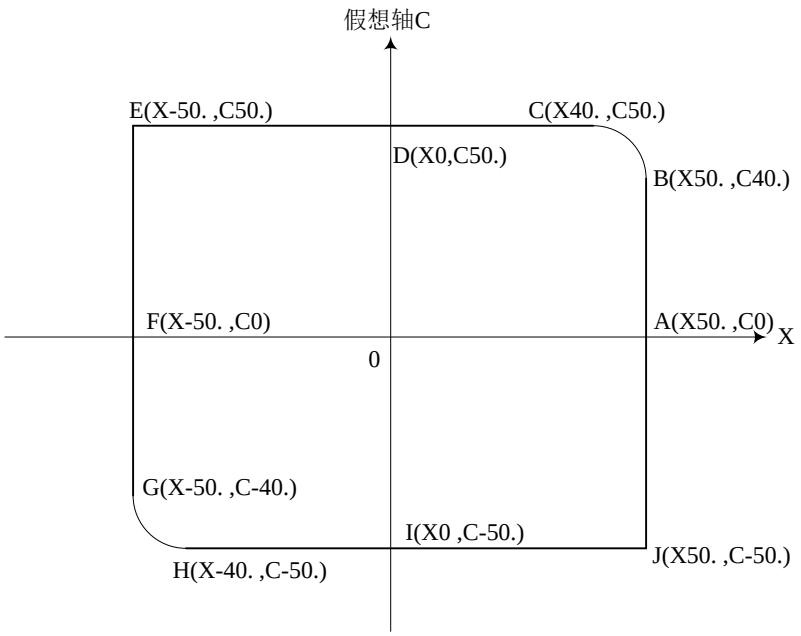
在未进入极坐标插补时，直线轴和旋转轴组成极坐标系。进入极坐标插补后，用户使用假想直角坐标系编程（笛卡尔坐标系），系统会自动将该假想直角坐标系转化为极坐标系。通过这种方式，可以达到简化编程的目的。

假想直角坐标系编程时，第 1 个轴使用的地址符为上述直线轴地址符，第 2 个轴使用的地址符为上述旋转轴地址符。编程时，第 2 个轴小数位数仍然为旋转轴的小数位数。当旋转轴与直线轴小数位数不同时，编程时应特别注意第 2 个轴的单位。

注

1 G12.1/G13.1 指令必须独占一段指令，同段中不能有其他 G 代码，否则报警（PS0009）。
2 i 指定的轴必须是直线轴，j 指定的轴必须是旋转轴，否则报警（PS0120）。
3 非 MDI 方式复位时自动取消极坐标插补，进入 G13.1 状态。
4 系统在取消极坐标插补时，自动重新设置参与极坐标插补的直线轴和旋转轴的绝对坐标。
5 极坐标插补不影响手动模式。应在进入手动模式前取消极坐标插补，否则，显示上绝对坐标更新正常，实际上机床在进行极坐标插补。
6 如果参与插补的旋转轴设定为 B/C/D 类旋转轴，那么进入极坐标插补后，自动转化为 A 类旋转轴，退出时恢复为 B/C/D 类旋转轴。
7 只有在取消状态（G13.1），才能指令 G12.1，否则 G12.1 被忽略。
8 如果在进入极坐标插补后进行刀具补偿，那么应该在退出极坐标插补前取消补偿。
9 在进入极坐标插补后不能进行工件坐标系设定。

举例



G12.1G13.1 应用示例图（路径：A->B->…->J->A）

加工程序

```
O0012 ;
G00 X50. Z0. Y0. C0.          ;定位到开始位置
G54.9 P142                    ;将默认笛卡尔坐标系 XYZ 更改为 XCZ
G17                           ;设置平面为 XC 平面
G12.1 P14                     ;启动极坐标插补，直线轴为 X 轴，旋转轴为 C 轴
G01 C40. F500.                ;直线运动
G03 X40. C50. I-10.           ; XC 平面圆弧运动
G01 X-50. ;
C-40. ;
G03 X-40. C-50. I10. ;
G01 X50. ;
C0. ;
G13.1                         ;取消极坐标插补
G54.9                         ;恢复默认笛卡尔坐标系（XYZ）
M30 ;
```

速度限制

防止极坐标插补的插补轨迹过于接近圆心导致旋转轴转速过大。在极坐标插补过程中，系统根据当前极半径长度与参数 P714，计算出安全最大进给速度。若当前进给速度超过该安全值，则进给速度被限制到该安全值上。

- **举例**

参数 P714 为 50.0 转/分，极半径为 5.0 毫米，则安全进给速度为：2 * π * 5.0 * 50.0 = 1570 毫米/分。即此时最大进给速度为 1570 毫米/分，任何大于此值的进给速度都被限制到此值。

- **超速报警**

系统会实时检查极坐标插补过程中旋转轴转速，若超过参数 P715 设定值，将触发驱动器 9 号报警，并中断该轴使能信号。

- **相关参数**

714	极坐标插补时，旋转轴控制转速

[数据类型] 浮点型

[默认设置] 50.0

715	极坐标插补时，旋转轴报警转速

[数据类型] 浮点型

[默认设置] 100.0

2.4 圆柱插补（G07.1）

圆柱插补是一种轮廓控制。它把笛卡尔坐标系内的编程指令转换为旋转轴的角度量输出。

格式

G07.1 IP r ； 启动圆柱插补

……

G07.1 IP 0 ； 取消圆柱插补

G07.1 由单程序段指定，可以 G107 代之。

IP：一个旋转轴的地址

r：工件半径

解释

在进入圆柱插补时，系统将旋转轴的角度值按侧面展开，转化为长度量。进入圆柱插补后，用户可以按展开面编程，系统自动将指令的长度量转化为角度量输出。退出圆柱插补时，又将长度量转化回角度量使用。

编程时，小数位数为旋转轴的小数位数。当旋转轴与直线轴小数位数不同时，应特别注意编程单位。

注

1 G07.1 指令必须独占一段指令，否则报警（PS0047）。

2 指令时必须处于补偿取消状态，否则报警（PS0048）。

3 指定的轴必须是旋转轴，否则报警（PS0108）。

4 非 MDI 方式复位时自动取消圆柱插补。

5 系统在取消圆柱插补时，自动重新设置参与圆柱插补的旋转轴的绝对坐标。

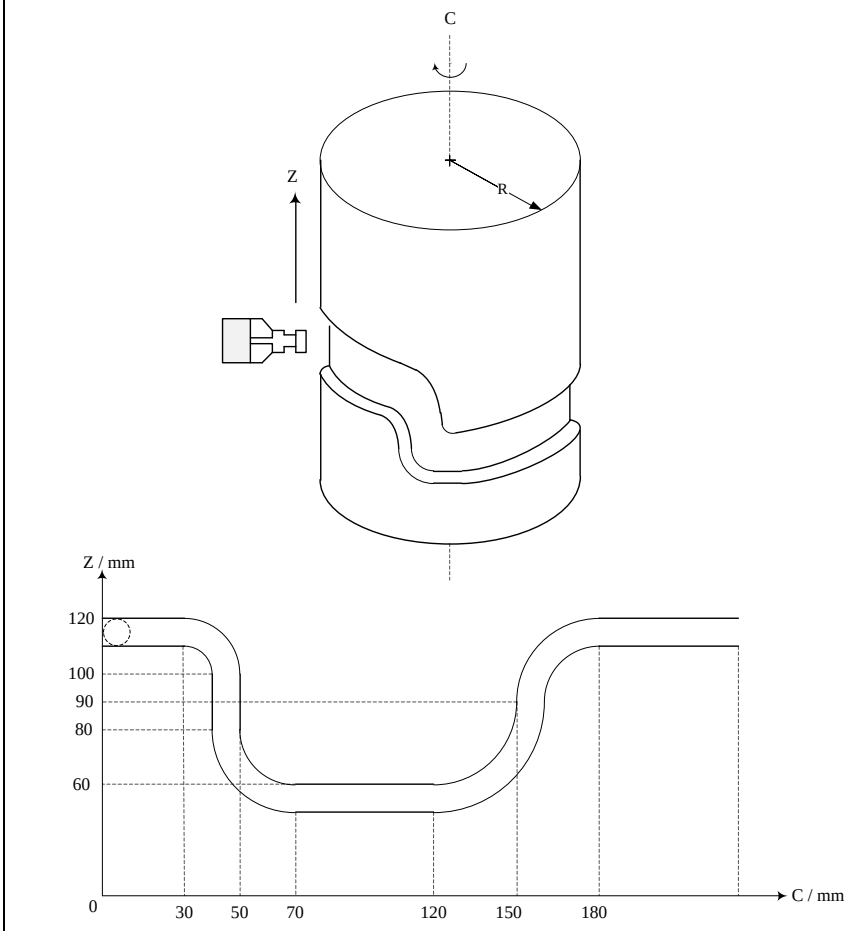
6 圆柱插补不影响手动模式。应在进入手动模式前取消圆柱插补，否则，显示上绝对坐标更新正常，实际上机床在进行圆柱插补。

7 如果参与插补的旋转轴设定为 B/C/D 类旋转轴，那么进入圆柱插补后，自动转化为 A 类旋转轴，退出时恢复为 B/C/D 类旋转轴。

8 如果在进入圆柱插补后进行刀具补偿，那么应该在退出圆柱插补前取消补偿。

9 在进入圆柱插补后不能进行工件坐标系设定。

举例



加工程序

```
O0107 ;
G00 X100. Z120. Y0. C0. ;          定位到开始位置
G54.9 P142 ;                       将默认笛卡尔坐标系 XYZ 更改为 XCZ
G07.1 C100. ;                      启动圆柱插补，半径 100.0mm
G42 G01 C30. D01 F500. ;           直线运动
G19 G02 Z100. C50. K-20. ;         CZ 平面圆弧运动
G01 Z80. ;
G03 C70. Z60. J20. ;
G01 C120. ;
G03 C150. Z90. K30. ;
G02 C180. Z120. J30. ;
……
G07.1 C0 ;                        取消圆柱插补
G54.9 ;                            恢复默认笛卡尔坐标系（XYZ）
M30 ;
```

2.5 假想轴插补（G07）

假想轴与非假想轴区别是：自动方式下，插补结束后，假想轴会清除插补输出的脉冲，因此实际不运动，机床坐标不更新，但绝对坐标更新；在取消假想轴后，恢复绝对坐标。

格式

G07 IP 0 ; **启动假想轴插补**

……

G07 IP 1 ; **取消假想轴插补**

IP：轴地址。指令其它非 0 数据，系统认为取消假想轴插补。

解释

- **半径补偿**

当假想轴位于半径补偿平面时，半径补偿自动取消。

- **假想轴的使用**

利用假想轴功能，由圆弧插补可得到正弦变化的移动速度，由螺旋线插补可得到正弦插补，由空间圆弧插补可得到椭圆插补等。

注

1 G07 指令应独占一段指令。

2 非 MDI 方式复位时自动取消假想轴插补。

3 系统在取消假想轴插补时，自动重新设置该轴的绝对坐标。

4 假想轴插补不影响手动模式。

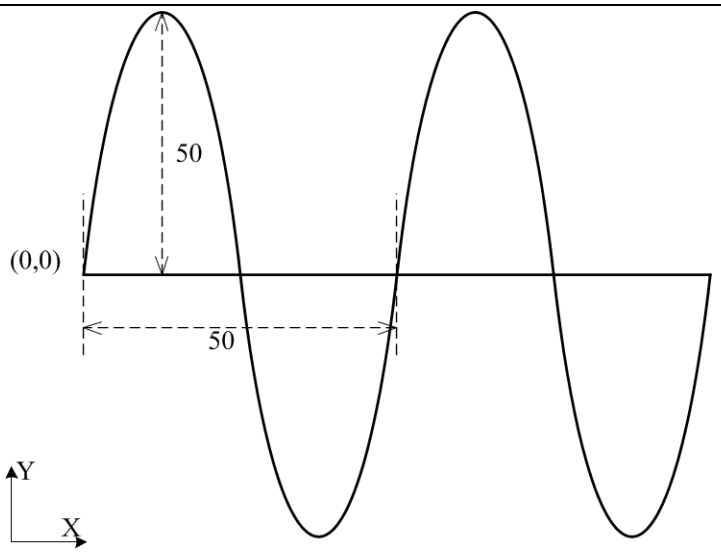
5 未指定轴自动取消假想轴插补功能。单独的 G07 指令取消所有假想轴插补。

6 设定的进给速度为假想轨迹速度，实际轨迹进给速度会小于等于该设定值。

举例

以下示例列举一些特殊使用场景。

- **正弦插补**



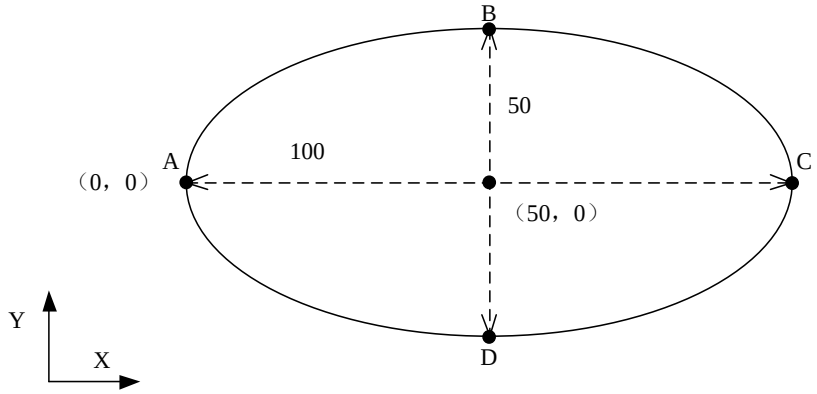
在 XY 平面上正弦插补。

起点（0，0），终点（100.0，0），长度 100.0，2 个完整周期，振幅 50，周期 50。

加工程序

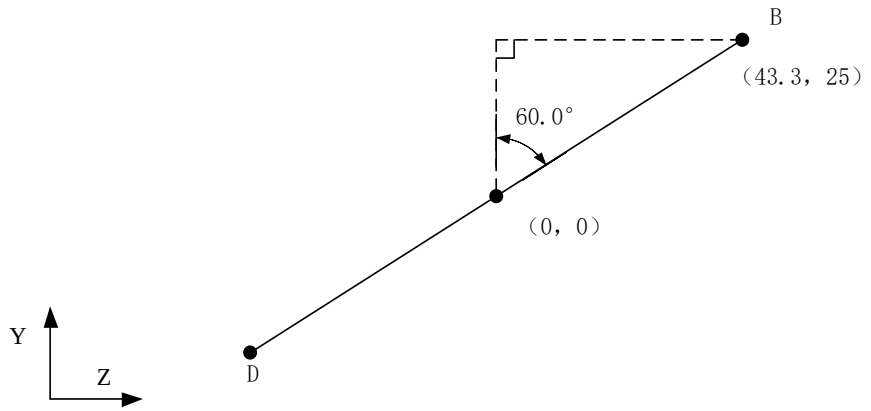
```
G90 G00 X0 Y0 Z0 ;
G07 Z0 ;
G19 G03 K50. X100. P1 F500.          ;在 YZ 平面完成两周整圆轨迹
G07 Z1 ;
```

- **椭圆插补**



在 XY 平面上椭圆插补，中心为（50.0，0），长轴为 100.0，短轴为 50.0。

利用假想轴 Z 与 X 轴 Y 轴组成假想空间，从 A 点过 B 点到 C 点完成上半圆加工，从 C 点过 D 点到 A 点完成下半圆加工。假想空间的 YZ 平面如下图所示。



加工程序

```
G90 G00 X0 Y0 Z0                ;起点（0，0）
G07 Z0 ;
G02.1 I50. J25. K43.3 X100. F500.  ;上半弧
I-50. J-25. K-43.3 X0.          ;下半弧
G07 Z1 ;
```

2.6 数字主轴螺纹插补（G07.2）

数字主轴螺纹插补是指使用数字主轴代替模拟主轴进行螺纹加工。

格式

G07.2 Pi NS_NF_ NP_ ； 启动数字主轴螺纹插补

…… NF_ ；

……

…… NF_ ；

G07.2 ； 取消数字主轴螺纹插补

Pi：i—数字螺纹导程方向直线轴轴号。数值为：1~8，分别对应第 1~8 轴。主轴使用当前主轴(NS 模态值)。

NS_：指定当前主轴。

NF_：G07.2 程序段的 NF_为启动数字主轴螺纹插补后第 1 个程序段的起点导程。后面的 NF_为当前段终点导程（也是下一段起点导程）。

NP_：主轴转动方向。

NP0：直线轴正向移动时，主轴正向运动；直线轴负向运动时，主轴反向运动。

NP1：直线轴正向移动时，主轴反向运动；直线轴负向运动时，主轴正向运动。NP0 可忽略。

注

1 Pij 两位数格式仍然保留。j 指定 Cs 主轴，如果与当前主轴(模态值 NS 指定主轴)不同，那么报警。

2 数字螺纹加工主轴在 G07.2 Pi 行记忆，后面变更当前主轴(NS)将报警。想变更数字螺纹加工主轴，必须重新指令(例如：“G07.2 Pi NSn”，n 为新主轴号)。

3 进入数字螺纹插补时，若主轴未进入位置模式，那么系统自动进入轮廓控制(插入当前主轴轮廓控制 M 代码)。

解释

数字主轴螺纹插补有效时，除定位程序段（G00）外，主轴跟随指定的直线轴运动。主轴最高转速为：主轴转速 $=\frac{\text{直线轴移动速度}}{\text{导程}}$

数字主轴螺纹插补有效时，“数字螺纹加速曲线”（P6418）和“数字螺纹减速曲线”（P6419）有效，“数字螺纹切削主轴平均加速度”（P6438）和“数字螺纹切削主轴平均减速度”（P6439）有效。快速加减速曲线和加减速度不受影响。

起点导程和终点导程不同时，导程在起点和终点间线性变化（变螺距螺纹）。

注

1 G07.2 指令应独占一段指令。

2 非 MDI 方式复位时自动取消数字主轴螺纹插补。

3 数字主轴螺纹插补不影响手动模式。

4 数字主轴螺纹插补有效，指令 NF 为 0 时，视为主轴不动（或导程无穷大）。

5 数字主轴螺纹插补有效时，系统只考虑起点到终点直线轴的移动量，而不考虑实际是否有移动。如果该移动量为 0，那么主轴不动。

6 数字主轴螺纹插补有效时，过高的直线轴移动速度或过小的导程，可能会导致主轴转速过高。

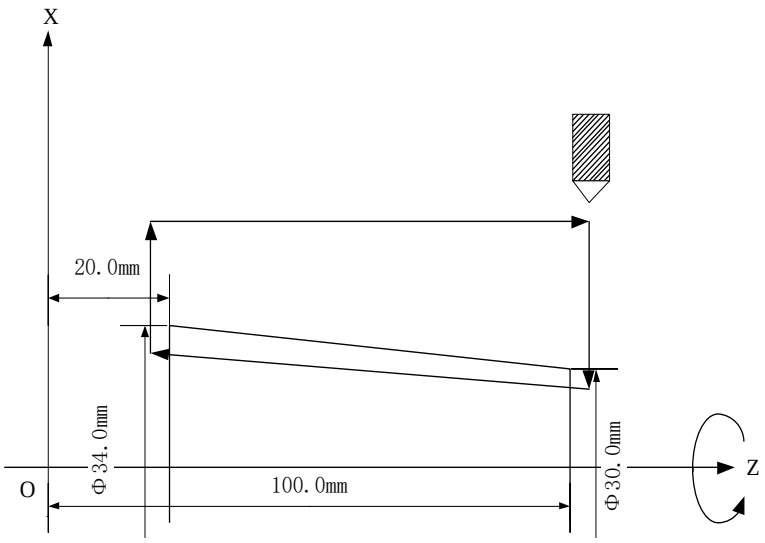
7 数字主轴螺纹插补有效时，攻丝循环（G93）、钻孔循环（包括刚性攻丝）不可使用。

8 G07.2 启动数字主轴螺纹插补时，系统自动插入当前主轴进入轮廓控制 M 代码。数字主轴螺纹插补有效时，不要再指令主轴正/反转（M03/M04），否则主轴转速为正/反转转速与数字主轴螺纹插补时主轴转速的叠加。

9 若在 G07.2 语句中使用转进给控制指令，则触发报警 PS283：“启动数字主轴螺纹加工后，不能进行转进给控制”。

举例

第一主轴螺纹插补

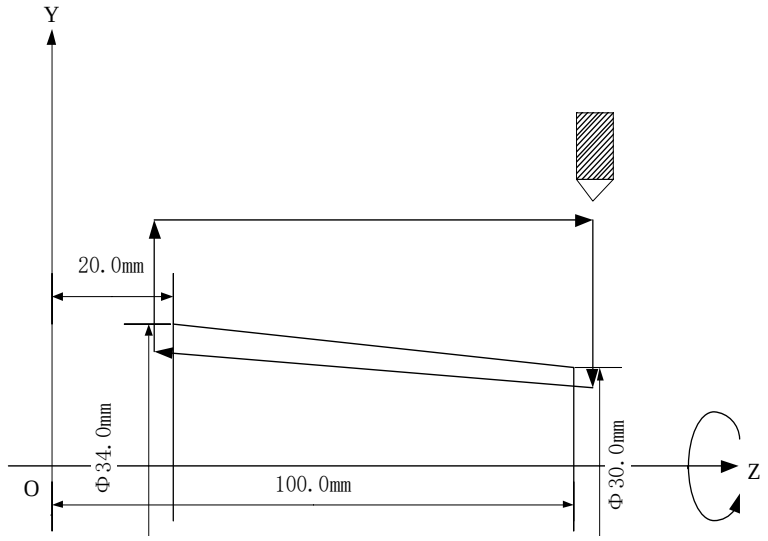


绕 Z 轴旋转的轴为第一主轴时

加工程序

```
M03 S1000;
G00 X40. Z102.0 ;
G07.2 P3 NS1 NF5.                    ;若之前未指令过 NS 指令，该段 NS1 可省略
X15. ;
G01 X17. Z18. F50. ;
G00 X40. ;
Z102. ;
G07.2 ;
```

第二主轴螺纹插补



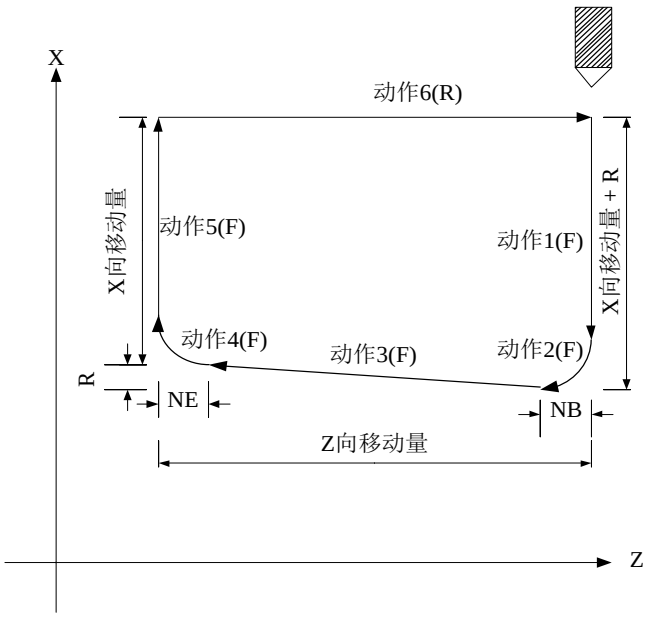
绕 Z 轴旋转的轴为第二主轴时

加工程序

```
M03.2 S2=1000;
G00 Y40. Z102.0 ;
G07.2 P3 NS2 NF5. ;
Y15. ;
G01 Y17. Z18. F50. ;
G00 Y40. ;
Z102. ;
G07.2 ;
```

2.7 数字主轴螺纹切削循环（G78.1）

数字主轴螺纹切削循环（G78.1）利用固定循环功能，在一个程序段中实现螺纹切削的进刀，加工，退刀，返回等多个动作，用于简化数字主轴螺纹插补时的编程，提高编程效率。G78.1 固定循环由 6 个动作完成，如下图所示（以 G18 Z/X 平面加工为例，其它加工方式相同）。



G78.1固定循环动作顺序：
动作1：切削进给进刀；
动作2：切削进给旋入；
动作3：切削进给加工；
动作4：切削进给旋出；
动作5：切削进给退刀；
动作6：快速进给返回；

格式

G18/G19 G78.1 IP1_ IP2_ R_ Q_ NB_ NE_ NP_ L_ F_ ;

G18/G19：指定 IP1/IP2 所在平面。平面内轴可由 G54.9 指令更改。

IP1_：非导程方向终点坐标或者移动量(绝对/相对)。平面内轴。

IP2_：导程方向终点坐标或者移动量(绝对/相对)。平面内轴，且为 G7.2 指定的直线轴。

R_：锥度螺纹最大端和最小段的半径差。

Q_：第一头螺纹的起始偏移角度。

NB_：螺纹旋入阶段的宽度。

NE_：螺纹旋出阶段的宽度。

NP_：指定 NP1 时多头螺纹的偶数头反向加工，指定 NP0 时正常加工多头螺纹。

L_：螺纹头数。

F_：切削进给速度。

解释

- **使用要点和注意事项**

1 在指定 G78.1 之前，需要先指定 G07.2 进入到数字主轴螺纹插补模式，否则报警（PS0287）。

2 G78.1 是模态指令，在未指令 01 组其他 G 代码时一直有效，指令 01 组其他 G 代码后清除 G78.1 的模态信息。

3 非 MDI 复位时，G78.1 被替换为 01 组上电时默认 G 代码。

4 不可在坐标变换（包括极坐标变换、比例缩放和坐标旋转）有效状态下指定 G78.1，否则报警（PS530/PS531/PS532）。

5 G7.2 指定的导程方向直线轴非平面内轴时，则报警（PS289）。

6 IP1_，IP2_在每个程序段都必须指令，如果只指令其中一个，则报警（PS535）；同时不允许非平面内轴输入，否则报警（PS534）。

7 如果平面内轴都没有输入时，系统将只记录模态信息而不运动。

8 固定循环从旋入阶段到旋出阶段禁止单段停止。

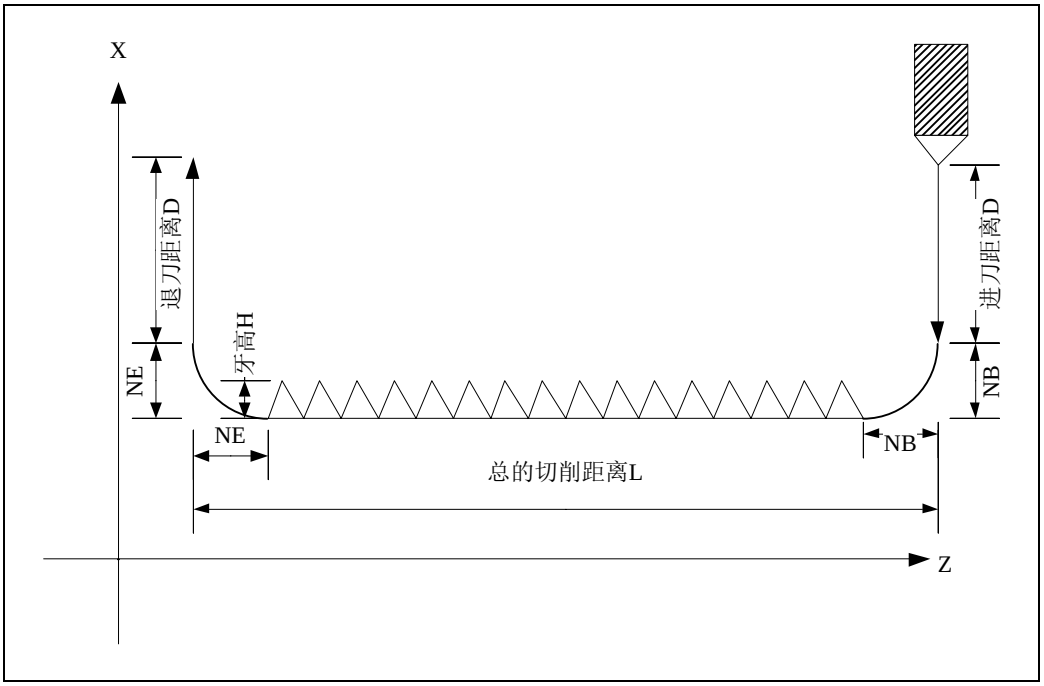
9 动作 6（快速返回到循环起点）的过程中，主轴自动旋转至零点，为下一次循环作准备。系统根

据退刀时主轴的位置，自动采取正转、反转或不转动的方法将主轴位置调整到零位。

10 不支持 G17 实现 X/Y 平面加工。如有需要，可使用 G54.9 进行坐标系变换，其第 3 轴作为纵向轴（或螺纹导程方向）。如，在车床 A 体系中，要以 Y 轴为纵向轴，X 为侧向轴，可以指令 G54.9 P123（XZY）。

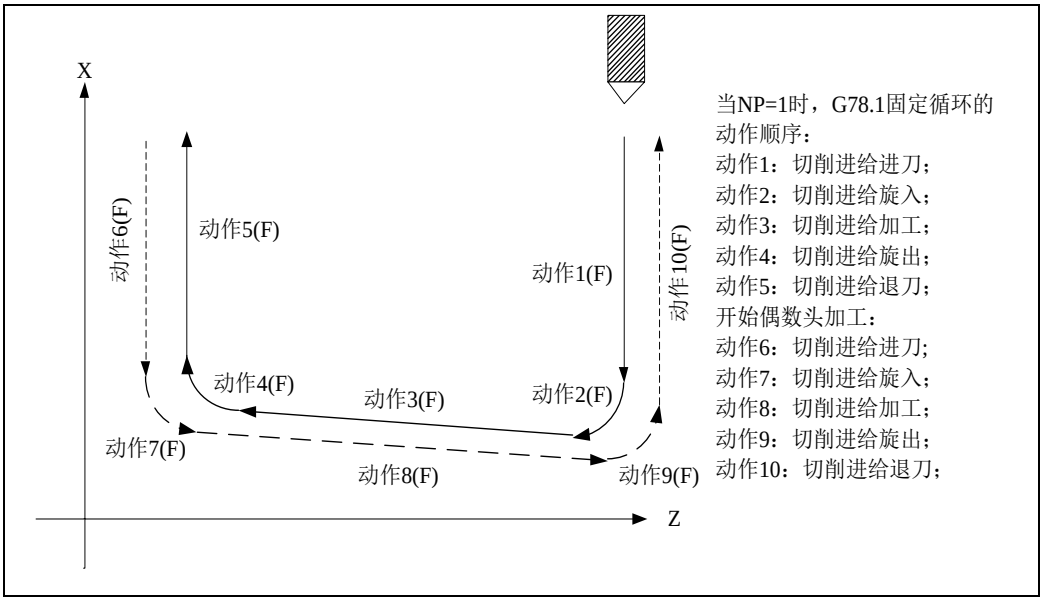
- **NB 和 NE 的意义**

在螺纹旋入阶段或者螺纹旋出阶段，由于升降速的原因需要一段加速或减速的距离，以实现刀具的旋入或旋出。系统在旋入或旋出阶段，根据 NB 或 NE 的值自动插入一段小圆弧指令实现该加减速过程，如下图所示。其中 NB 或 NE 的距离应大于螺纹牙高 H，并且远小于进刀或者退刀距离 D，同时 NB 与 NE 的和不能超过 Z 轴的总进给量 L。



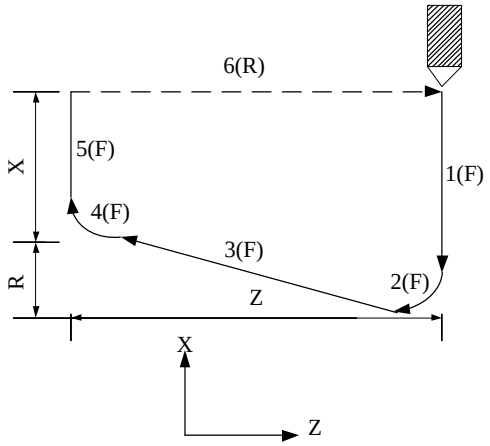
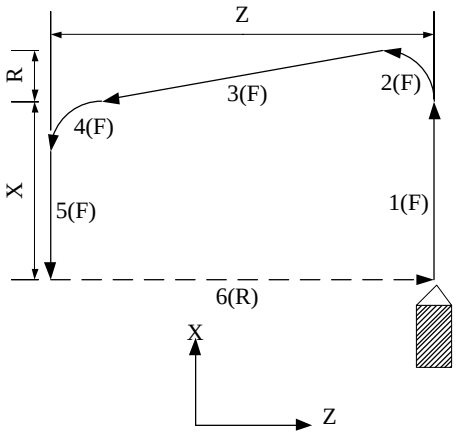
- **NP 的意义**

NP 为 1 时，多头螺纹加工的偶数头将进行反向加工，省去了快速运动回到初始点的过程，提高了加工效率。当不是多头螺纹或者 NP 等于 0 时，将正常加工多头螺纹，即快速运动回到起始点后继续正向加工下一头螺纹。指定 NP1 的加工情况如下图所示。

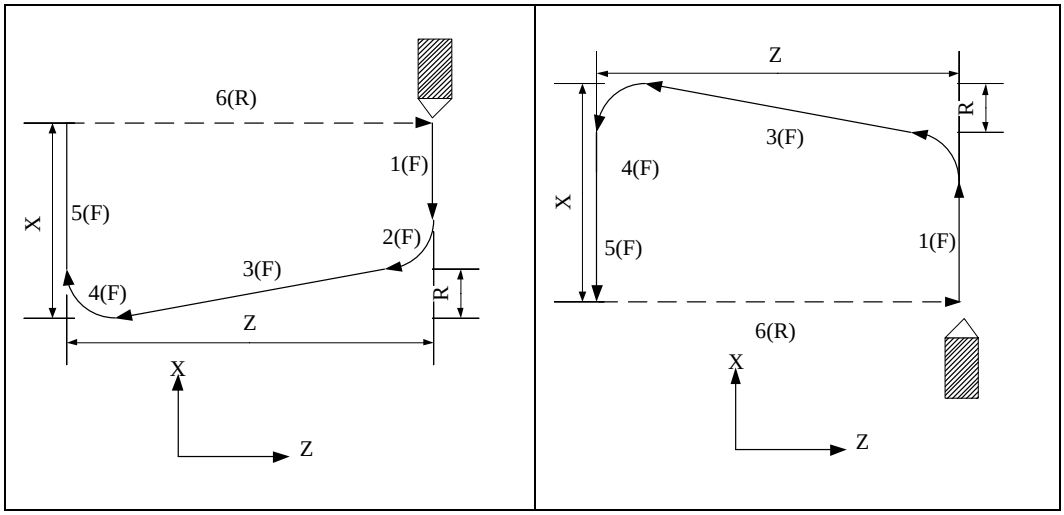


增量指定下地址 X、Z、Y、R 符号与轨迹的关系

以 X/Z 平面加工为例，增量值指定时，地址 X、Z、R 后数值的符号和刀具轨迹的关系如下所示。如需使用 Y/Z 平面，或使用 G54.9 指令任意平面加工，则方向关系与此相同，可以 Z 轴或 G54.9 第 3 轴（螺纹导程方向轴）为基准进行类推。

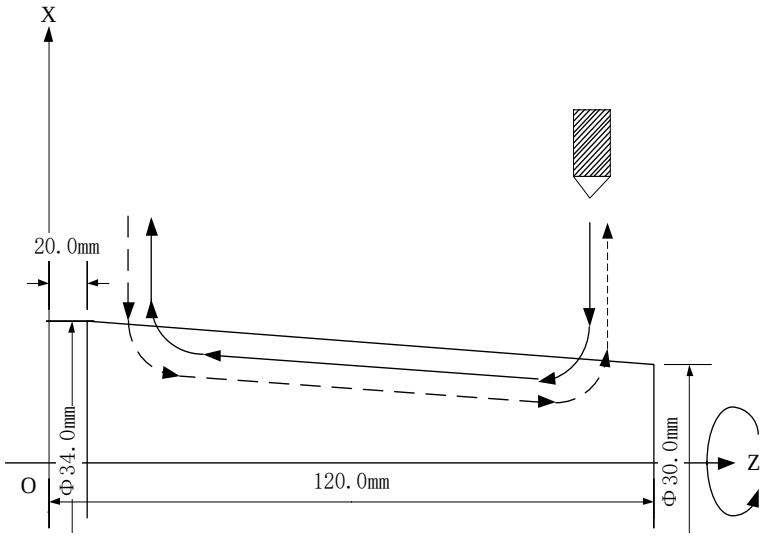
（1）X<0，Z<0，R<0	（2）X>0，Z<0，R>0
	

（3）X<0，Z<0，R>0，但｜R｜≤｜X｜	（4）X>0，Z<0，R<0，但｜R｜≤｜X｜



举例

第一主轴螺纹切削循环

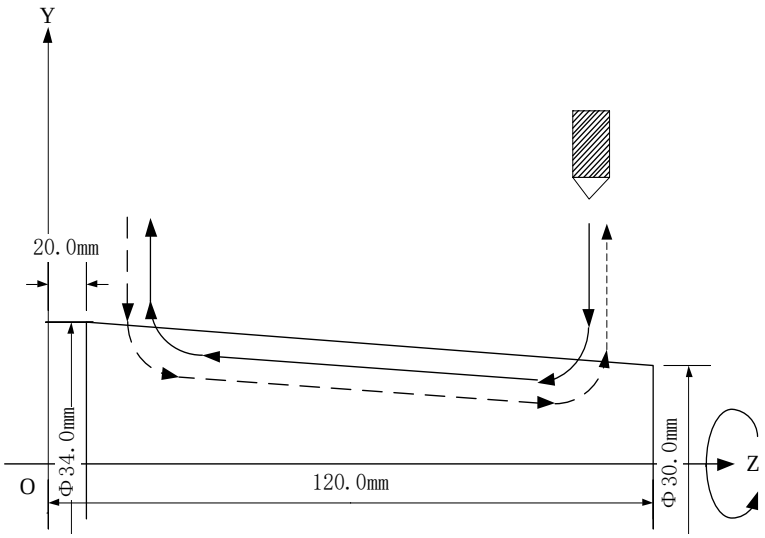


绕 Z 轴旋转的轴为第一主轴时

加工程序

```
M03 S1000 ;
G18 G00 X40. Z100.0 ;
G07.2 P3 NS1 NF5.                ;若之前未指令过 NS 指令，该段 NS1 可省略
G78.1 X17. Z20. R-2. NB3.0 NE3.0 NP1 L2 F5. ;
X16 Z20. ;
G07.2 ;
```

第二主轴螺纹切削循环



绕 Z 轴旋转的轴为第二主轴时

加工程序

```
M03.2 S2=1000 ;
G19 G00 Y40. Z100.0 ;
G07.2 P3 NS2 NF5. ;
G78.1 Y17. Z20. R-2. NB3.0 NE3.0 NP1 L2 F5. ;
Y16 Z20. ;
G07.2 ;
```

2.8 主插补轴功能（G07.3）

在某些场合中，同时运动的多个轴中，以某一个轴的运动为主要运动，其给定速度也是可确定的。则以该轴作为“主插补轴”，创建一个从主插补轴的起点到终点的虚拟运动，各个轴的实际合成运动是与该虚拟运动同时开始、同时结束的。这个功能称为“主插补轴”功能。配合 G05 功能，可应用于凸轮车、凸轮磨等。

格式

G7.3 IP p ; **启动主插补轴功能**

……

G7.3.; **取消主插补轴功能**

IP：“主插补轴”对应的轴地址。

p：取值范围：[1.0, 1000.0]。含义：程序段路径长度（名义长度）与主插补轴运动增量之间的比例限制。超出该限制后会报警。

解释

主插补轴只能用于螺纹切削以外的运动，一般用于直线运动。对于直线，可理解为“速度（F 值）直接作用于主插补轴上，其它轴跟随运动”。因此，启动主插补轴功能后，速度 F 值不再是各轴的合成速度，需注意。

1 只有当主插补轴有移动量时，该功能才生效。

2 非 MDI 方式下，按下复位按键可取消主插补轴功能。

举例

凸轮车

X/Y/Z/C 4 轴系统，C 轴控制头架（旋转轴），X 轴控制砂轮工进，3 位小数点，使用绝对指令编程。为了提高效率，C 轴采用匀速旋转。

```
%
(子程序);
G5 NP3 R200. ;
G7.3 C1.5                ;启用主插补轴功能，C 为主插补轴
G0 X0 C0 ;
G1 F360                  ;指令主插补轴速度 F 值（参考名义移动量计算）
#512 = 0 ;
#513 = 100 ;
WHILE[#512 < #513] DO1 ;
N2500 X0 C0              ;基圆部分
N3401 X0 C90.1           ;开始桃尖部分
N3402 X0 C90.2 ;
N3403 X0.00014 C90.3     ;虽然显示 3 位小数，但 NP 功能可直接解析 6 位小数
N3404 X0.0002 C90.4 ;
N3405 X0.00026 C90.5 ;
N3406 X0.00032 C90.6 ;
N3407 X0.00038 C90.7 ;
N3408 X0.00043 C90.8 ;
N3409 X0.00047 C90.9 ;
N3410 X0.0005 C91 ;
……
N5198 X0 C269.8 ;
N5199 X0 C269.9 ;
N6090 X0 C359 ;
N6091 X0 C359.999 ;
#512 += 1 ;
END1 ;
G7.3                     ;取消主插补轴功能
M99                      ;返回主程序
%
```

凸轮磨

X/Y/Z/A 4 轴系统，Y 轴控制头架（旋转轴），X 轴控制砂轮工进，3 位小数点，在 M 系统运行。为了达到平稳的磨削力，桃尖处 Y 轴的速度是变化的。

```
%
(子程序);
G5 NP3 R200;
G7.3 Y10                          ;启用主插补轴功能，Y 为主插补轴
#512=0 ;
#513=5 ;
WHILE[#512<#513]DO1 ;
N10100 G91 G01 Y1 X0 F1200        ;增量指令编程
N10101 Y1 X0 F1200                ;基圆部分
N10102 Y1 X0 F1200 ;
N10103 Y1 X0 F1200 ;
……
N10860 Y1 X0 F1039                ;F 值变化
N10870 Y1 X0 F1004 ;
N10880 Y1 X0 F969 ;
N10890 Y1 X0 F936 ;
N10900 Y1 X0 F902 ;
N10910 Y1 X0 F866 ;
……
N11610 Y1 X.151 F241              ;桃尖部分
N11620 Y1 X.143 F249 ;
N11630 Y1 X.135 F251 ;
N11640 Y1 X.127 F258 ;
N11650 Y1 X.118 F262 ;
N11660 Y1 X.111 F267 ;
……
N15000 #512=#512+1 ;
N15100 END1 ;
G7.3                              ;取消主插补轴功能
G91 G01 Y10 X0.2 F3000            ;此时的 F 为合成速度（按照名义移动量计算）
M99                               ;返回主程序
%
```

注

1 使用该功能时，一般主插补轴的运动量相对较大，是主要运动轴，否则可能导致其它跟随插补轴的速度或加速度过大。

2 因主插补轴有效时，F 值意义与无效时不同，故不需要主插补轴功能时（如上述程序返回主程序前），要编入取消指令，防止其他程序中多轴联动时速度异常，引起危险。

3 为减小其他轴跟随时的加速度，可将拐角处的轨迹变得更细密，或增加 G05 段中 R 的大小。

4 为使小线段加工更流畅、加工精度更高，可从以下几方面对程序进行优化：

(1) 增加循环内（上述程序 WHILE-END 之间）的程序段数：将轨迹密化，或将循环体展开多次；

(2) 循环内的地址值尽量用常数，少用宏表达式计算；

(3) 循环内不要使用 G05，否则会禁止预读。

3 进给功能

3.1 加减速控制

本系统自动进行加减速控制，不但能够使刀具的启动和停止更加平稳，而且在运行过程中也可以使速度平稳的变化。因此，编程时不需要对加减速进行考虑，但要提前设置好加减速相关参数。

加减速控制有插补前加减速控制和插补后加减速控制之分，本系统在插补前进行加减速控制，并可跨多段加减速。

解释

- **加减速曲线类型**

本系统提供 6 种加减速曲线类型供用户选择，分别为:（2 次）直线、2.5 次曲线、（3 次）S 曲线(无匀加速段)、4 次曲线、三角曲线、带匀加速段的 S 曲线。曲线次数是从位移与时间关系得出的。

加速和减速可选择不同的曲线类型，并可分别选用不同的平均加速度。

相关参数如下：

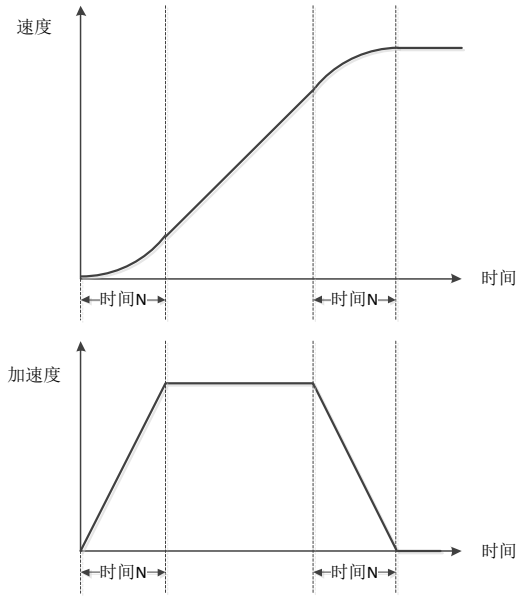
1 自动方式相关参数：

- 切削加速曲线（P6410）/切削减速曲线（P6411）
- 快速运动加速曲线（P6412）/快速运动减速曲线（P6413）
- 刚性攻丝加速曲线（P6414）/刚性攻丝减速曲线（P6415）
- 数字螺纹加速曲线（P6418）/数字螺纹减速曲线（P6419）
- 螺纹加速曲线（P6420）/数字螺纹减速曲线（P6421）
- 螺纹让刀加速曲线（P6482）

参数数值说明如下：

数值 0~4 为原曲线类型。

数值>=10(<=200)为带匀加速段的 S 曲线。且数值大小为时间 N。如下图：



- 带匀加速段的 S 曲线图

注：上述参数系统默认 30，即使用带匀加速段的 S 曲线。

其中紧急减速曲线用于复位/暂停/轴锁等情况，急停或限位报警时立即停止输出，无减速过程。

2 手动模式相关参数(目前不能使用带匀加速段的 S 曲线)：

- 手动加速曲线（P5210）/手动减速曲线（P5211）
- 单步或手轮加速曲线（P5410）/单步或手轮减速曲线（P5411）
- 回零加速曲线（P5110）/回零减速曲线（P5111）

- **加速度与减速度数值**

不管是哪一种加减速曲线，当加速度太大时，都会逼近直线，影响加工的平稳性，因此要合理设置加速度和减速度。数值太小会影响加工效率，太大会造成机床震动。

系统使用平均加/减速度方便用户使用。通过最大速度和平均加速度或平均减速度可计算出加速或减速时间。

例

最大速度 2000mm/min，平均加速度（20000mm/min）/s，平均减速度为（10000mm/min）/s，那么从停止开始加速到最大速度的加速时间为 100ms，从最大速度减速到停止的减速时间为 200ms。

平均加速度或减速度适用于各种曲线类型。

多轴联动时，加速度或减速度均指的是切线方向的平均加减速度。

相关参数如下：

1 自动方式相关参数：

- 切削平均加速度（P6430）/切削平均减速度（P6431）
- 快速运动平均加速度（P6432）/快速运动平均减速度（P6433）
- 刚性攻丝平均加速度（P6434）/刚性攻丝平均减速度（P6435）
- 紧急减速系数（P6436）
- 数字螺纹平均加速度（P6438）/数字螺纹平均减速度（P6439）
- 螺纹切削平均加速度（P6480）/螺纹切削平均减速度（P6481）

其中，紧急减速系数用于复位/暂停/轴锁等情况，急停或限位报警时立即停止输出，无减速过程。

2 手动模式相关参数：

- 手动平均加速度（P5212）/手动平均减速度（P5213）
- 单步或手轮平均加速度（P5412）/单步或手轮平均减速度（P5413）
- 回零平均加速度（P5112）/回零平均减速度（P5113）
- 手轮平滑时间（P5414）
- 手轮或单步速率限制（P5420）

- **加减速相关参数显示设置**

系统支持对加减速曲线类型和加减速度参数的显示进行设置。

1 加减速曲线类型参数是否显示受参数“FACDCT”（P0005.2）控制。

2 加减速度参数的单位由参数“ACDCMS”（P0005.4）控制。

3 加减速度参数合并功能是否有效由参数“ACDCM”（P0005.4）控制。

注

“FACDCT”（P0005.2）仅影响回零、手动、单步的加减速曲线是否显示。

1 回零加速曲线（P5110）/回零减速曲线（P5111）

2 手动加速曲线（P5210）/手动减速曲线（P5211）

3 单步或手轮加速曲线（P5410）/单步或手轮减速曲线（P5411）

- **加减速相关控制指令**

使用 CT，AD，AC 和 DC 指令在加工程序中控制加减速度和加减速曲线类型。

CT：曲线类型

AD：加减速度

AC：加速度

DC：减速度

注

1 AD，AC 和 DC 的单位与加减速度是否用时间表示无关，与公英制无关。不影响螺纹切削段、刚性攻丝切削段、进入数字螺纹控制的程序段。

2 CT 取值说明：0~4 对应不同的曲线；>=5 时为带匀速段的 S 曲线，且数值大小表示平滑时间

例

G153 Z200. NF12000. CT11 AD80. ；

编写 G153 换刀程序时，可以指定加减速曲线和加减速度，以便防止用户修改加减速度和加减速曲线后影响换刀时间。

- **加减速控制方式**

插补功能	可用加速曲线类型	可用减速曲线类型	
		正常情况	复位/暂停/轴锁/急停/限位等紧急情况
直线/圆弧/螺线插补（每分进给）	√（2 次）直线 √2.5 次曲线 √（3 次）S 曲线 √4 次曲线 √三角曲线 √带匀加速段的 S 曲线 [相关参数] P6410/P6412/P6430/P6432	√（2 次）直线 √2.5 次曲线 √（3 次）S 曲线 √4 次曲线 √三角曲线 √带匀加速段的 S 曲线 [相关参数] P6411/P6413/P6431/P6433	√（2 次）直线 √2.5 次曲线 √（3 次）S 曲线 √4 次曲线 √三角曲线 √带匀加速段的 S 曲线 [相关参数] P6436
直线/圆弧/螺线插补（每转进给）	√（2 次）直线 [相关参数] P6430/P6432 注：只有直线有效，无法选择曲线类型	√（2 次）直线 √2.5 次曲线 √（3 次）S 曲线 √4 次曲线 √三角曲线 √带匀加速段的 S 曲线 [相关参数] P6411/P6413/P6431/P6433	√（2 次）直线 √2.5 次曲线 √（3 次）S 曲线 √4 次曲线 √三角曲线 √带匀加速段的 S 曲线 [相关参数] P6436
直线/圆弧插补（自动手轮控制） 注：自动手轮控制对螺纹切削部分无效	√（2 次）直线 [相关参数] P5310 注：只有直线有效，无	√（2 次）直线 [相关参数] P5310 注：只有直线有效，无	√（2 次）直线 [相关参数] P5310

插补功能	可用加速曲线类型	可用减速曲线类型	
		正常情况	复位/暂停/轴锁/急停/限位等紧急情况
（包括 G93 攻丝）	法选择曲线类型	法选择曲线类型	
数字主轴刚性攻丝	√（2 次）直线 √2.5 次曲线 √（3 次）S 曲线 √4 次曲线 √三角曲线 √带匀加速段的 S 曲线 [相关参数] P6414/P6434	√（2 次）直线 √2.5 次曲线 √（3 次）S 曲线 √4 次曲线 √三角曲线 √带匀加速段的 S 曲线 [相关参数] P6415/P6435	√（2 次）直线 √2.5 次曲线 √（3 次）S 曲线 √4 次曲线 √三角曲线 √带匀加速段的 S 曲线 [相关参数] P6436
数字主轴螺纹插补	√（2 次）直线 √2.5 次曲线 √（3 次）S 曲线 √4 次曲线 √三角曲线 √带匀加速段的 S 曲线 [相关参数] P6418/P6438	√（2 次）直线 √2.5 次曲线 √（3 次）S 曲线 √4 次曲线 √三角曲线 √带匀加速段的 S 曲线 [相关参数] P6419/P6439	√（2 次）直线 √2.5 次曲线 √（3 次）S 曲线 √4 次曲线 √三角曲线 √带匀加速段的 S 曲线 [相关参数] P6436
手动方式/单步方式/回零方式	√（2 次）直线 √2.5 次曲线 √（3 次）S 曲线 √4 次曲线 √三角曲线 √带匀加速段的 S 曲线 [相关参数] P5110/P5210/P5410	√（2 次）直线 √2.5 次曲线 √（3 次）S 曲线 √4 次曲线 √三角曲线 √带匀加速段的 S 曲线 [相关参数] P5111/P5211/P5411	√（2 次）直线 √2.5 次曲线 √（3 次）S 曲线 √4 次曲线 √三角曲线 √带匀加速段的 S 曲线 [相关参数] P5111/P5211/P5411
手轮模式	√（2 次）直线 [相关参数] P5414/P5420	√（2 次）直线 [相关参数] P5414/P5420	√（2 次）直线 [相关参数] P5414/P5420

3.2 速度平滑控制

在加工中进行速度平滑可提高加工的精度，提高加工效率，避免一些机械冲击从而提高加工水平。系统提供了在不同加工中适用的平滑器供用户加工时进行选择使用。

各平滑器说明如下：

平滑器	参数	说明
切削平滑器	P6502 (默认切削平滑阶数) 或通过 G05R 更改	平滑切削运动，但不包括以下运动： 1 螺纹切削 2 插补型攻丝/G93 攻丝 3 G31/G36/G37（无任何平滑器）
快速平滑器	P6461	平滑快速运动

平滑器	参数	说明
	(快速平滑时间)	
螺纹平滑器	P6462 (螺纹切削平滑时间)	平滑螺纹切削 注：除大螺距螺纹加工外，不建议开启；开启螺纹平滑会降低螺纹跟随性。
刚性攻丝平滑器	P6463 (插补型攻丝、G93 攻丝平滑时间)	平滑插补型攻丝及 G93 攻丝
手动模式平滑器	P6465 (手动模式平滑时间)	平滑手动模式相关运动
PLC 轴平滑器	P6466 (PLC 轴平滑时间)	平滑 PLC 轴运动
简易 PLC 轴平滑器	P6467 (简易 PLC 轴平滑时间)	平滑简易 PLC 轴运动
主轴平滑器	P2779/P2829/… (转速控制平滑时间)	平滑主轴转动

4 倾斜平面指定

空间插补指令（包括空间等速螺线和空间等角螺线指令）可以指定一个倾斜平面。

格式

- **格式 1**

NX_ NY_ NZ_

一次性指定倾斜平面的法矢量

- **格式 2**

NI_ NJ_ NK_

模态指定倾斜平面的法矢量

注

1 NX、NY、NZ 分别代表空间笛卡尔坐标系的第 1、2、3 个轴。NI、NJ、NK 同样分别代表空间笛卡尔坐标系的第 1、2、3 个轴。

2 当 NX、NY、NZ 这三个一次性地址符中有一个输入时，那么格式 1 有效。NX0、NY0 或 NZ0 可以省略，但 NX、NY 和 NZ 不能同时为 0。

3 当 NX、NY、NZ 这三个一次性地址符都未输入时，那么格式 2 有效。NI、NJ、NK 是模态地址符，上电时清 0。

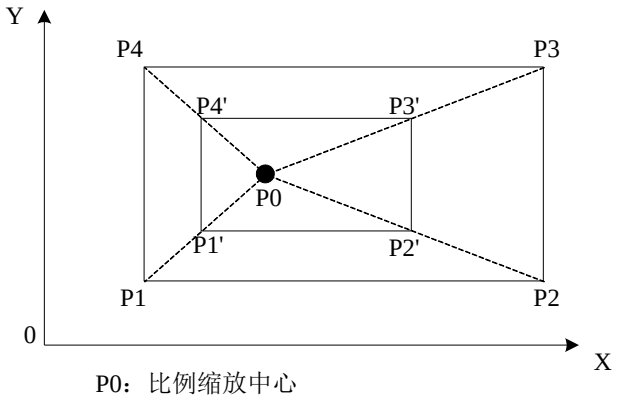
4 法矢量可以为任意长度。

5 坐标值和尺寸

5.1 比例缩放（G50，G51）

编程形状成比例的放大或缩小，称为比例缩放。G 代码体系 A 不支持此功能。

各轴放大或缩小的比例可以相同，也可以不同。比例可以在程序中指定，也可以用参数指定。如下图所示。



P0：比例缩放中心

P1~P4：加工程序的形状（比例缩放前）

P1'~P4'：比例缩放后的形状

格式

- **沿所有轴以相同比例缩放**

G51 IP__ P__ ; **缩放开始**

……

G50 ; **缩放取消**

IP__：比例缩放中心坐标值的绝对值指令。

P__：缩放比例。

- **沿各轴以不同比例缩放（镜像）**

G51 IP_ I_ J_ K_ ; **缩放开始**

……

G50 ; **缩放取消**

IP__：比例缩放中心坐标值的绝对值指令。

I_ J_ K_：X、Y、Z 各轴对应的缩放比例。

解释

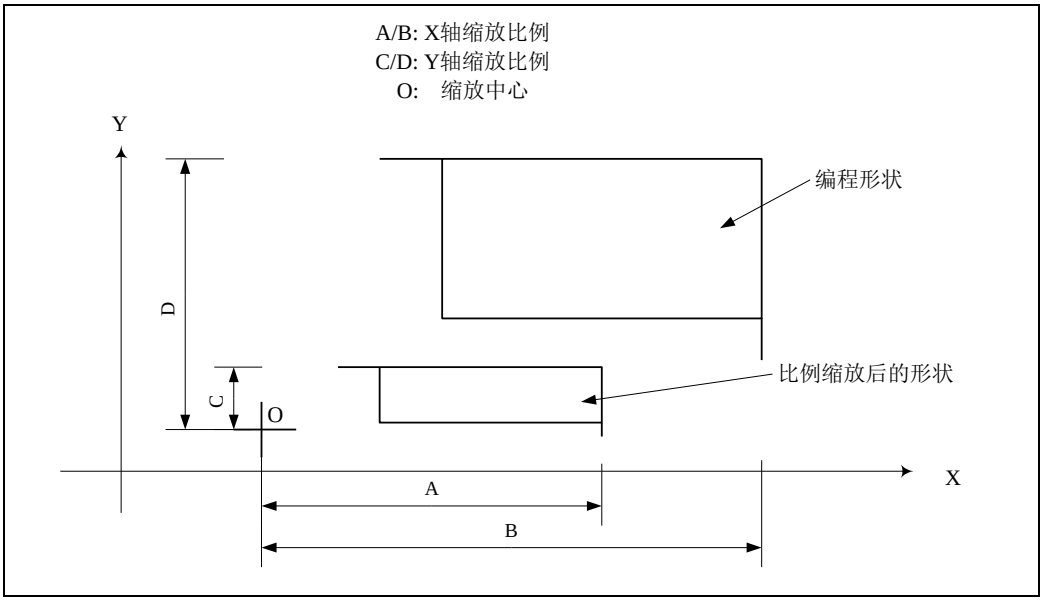
- **以相同比例沿所有轴缩放**

在这种方式下，P__必须输入，否则被认为是各轴比例缩放（I、J、K）格式。如果省略 IP__，则以指令 G51 时的刀具位置作为缩放中心。

缩放比例 P 单位为 0.001，与系统最小单位无关，与公英制无关。

- **沿各轴比例缩放与可编程镜像（负的比例）**

各轴用不同的比例缩放，如下图所示。



各轴的比例（I、J、K）的最小输入单位为该轴的最小输入单位。建议输入时带小数点，以便加工程序在最小输入单位不同的系统上运行。

如果设置负值，则成为镜像。

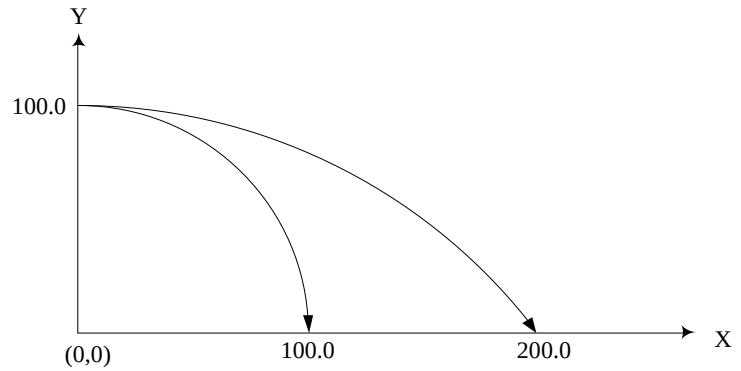
如果某一轴未使用 I/J/K 指定比例，则参数“比例缩放指令 G51 的缩放比例”（P3611）设定值有效。

使用 G54.9 变换笛卡尔坐标系以后，再使用 G51 缩放指令时，请不要使用参数 P3611，应使用 IJK 方式指令每一个轴的缩放系数，或使用 P 指令各轴一致的缩放系数，否则变换结果将与期望不同。

- **圆弧插补的比例缩放**

即使对圆弧插补的各轴指定不同的缩放比例，刀具也不会画出椭圆轨迹。

当各轴的缩放比例不同，圆弧插补用半径 R 编程时，其插补的图形如下图所示（X 轴的比例为 2，Y 轴的比例为 1）。圆弧终点根据缩放中心和比例，在 X 方向放大到 2 倍，圆弧半径取缩放比例中的最大值，放大至 2 倍。




```
G90 G00 X0.0 Y100.0 ;
G51 X0.0 Y0.0 I2.0 J1.0 ;
G02 X100.0 Y0.0 R100.0 F500.0 ;
上述指令与下面的指令效果相同
G90 G00 X0.0 Y100.0 ;
G02 X200.0 Y0.0 R200.0 F500 ;
半径R的缩放比例取I和J两者中的大值
```

当各轴的缩放比例不同，圆弧插补用 I、J、K 编程时，圆弧终点根据缩放中心和比例确定位置，圆

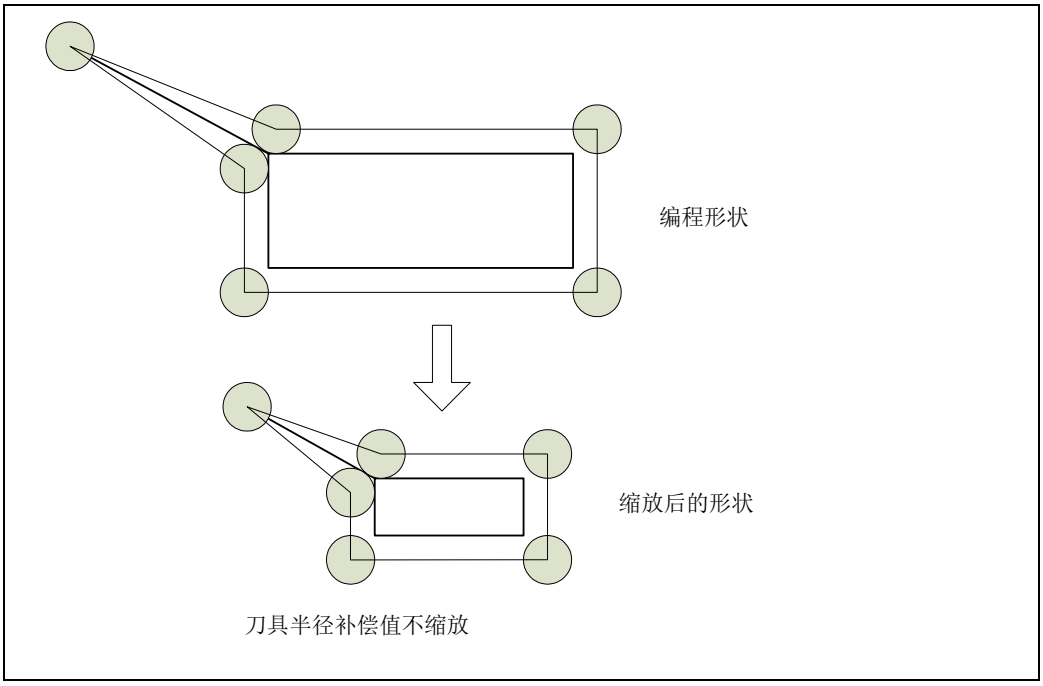
弧中心根据缩放后的 I、J、K 得到，通常这样会使圆弧起始半径和终点半径不同，易产生报警（PS0200）。因此，对于圆弧插补的比例缩放，建议用户不要使用 I、J、K 指定圆心编程。

- **参数 EMPNM 对比例缩放的影响**

  当参数“EMPNM”（P3600.0）设置为 1 时，如果未指定笛卡尔坐标轴（1~3 轴），那么不进行坐标变换，1~3 轴不移动。通常情况应将该参数设置为 1，以防止空行因坐标变换而产生移动。

- **刀具半径补偿**

  刀具半径补偿值不缩放，如下图所示。



刀具半径补偿值不缩放

- **刀具长度补偿**

  刀具长度补偿应在 G50 状态进行。G51 状态时，不应指令长度补偿指令，也不应通过 G10 修改长度补偿值。

- **比例缩放无效**

  以下几种情况，比例缩放无效：

  - 在固定循环（G83.5、G84.5、G86.5、G81~G89）中，Z 轴移动的缩放无效
  - 深孔钻循环（G83，G83.5）的切入值 q 和返回值 d 的缩放无效。
  - 精镗循环（G86.5）、反镗循环（G87）中 X 轴和 Y 轴偏移值 q 的缩放无效。
  - 手动运行时，移动距离不能用缩放功能增减。

- **关于参考点和坐标系的指令**

  在缩放状态，不能指令返回参考点的 G 代码（G27~G30）和指令坐标系的 G 代码（G52~G59、G50），也不能通过 G10 修改当前工件坐标系。若必须指令这些 G 代码，应在取消缩放功能后指令。

注

1 必须在单独程序段内指定 G51。在图形缩放后，指定 G50 取消缩放方式。

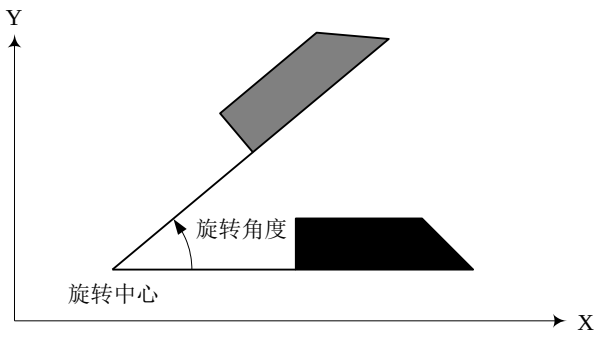
2 如果在指定 G50 取消缩放方式时 G68.1 有效，那么由于当前位置在比例缩放取消后发生变化，即便 G50 单独一段，也可能会有运动。这时应在指令 G50 前回到缩放中心或同时指令 G50 和 G69.1。

3 如果比例缩放的结果被四舍五入，移动量可能为 0。此时，该程序段被认为是不移动的程序段。

4 倒角/过渡圆尺寸值不随比例缩放变化，缩放后可能会触发报警 PS0055。

5.2 坐标旋转（G68.1，G69.1）

使用坐标旋转指令，可将工件旋转某一指定的角度，以便于加工在位置上需要旋转的工件。另外，如果工件的形状由多个相同的图形圆周排列组成，可先将图形单元编成子程序，然后用主程序的旋转指令调用。这样可节省用户编程时间和系统存储空间。示意图如下。



6 主轴比例控制（SPRC）

本系统设计有两个主轴比例控制器（SPRC）：SPRC1 和 SPRC2。SPRC 是用于控制从动主轴，使其跟随主动主轴、按编程比例转动，同时可进行相位控制和补偿控制的功能。格式及说明

格式

- **启动 SPRC 控制器**

G251 NLnl NPnp Pp Qq Rr NQnq Ee Kk

[说明]

nl：选择 SPRC 控制器。有效数值：1~2，对应 SPRC1~SPRC2。未编入默认 SPRC1 控制器。

np：主轴号选择。

十位数：主动主轴号(1~最大主轴数)，个位数：从动 Cs 主轴号(1~最大主轴数)。

未编入时，默认主动主轴为第 1 主轴，从动主轴为第 2 主轴。

p/q：主动主轴和从动主轴转速比。

(1) p 有效范围：1~1000，q 有效范围：-1000~-1 或 1~1000。pq 支持小数。

(2) P 未编入时，默认 p=1

(3) Q 未编入时，默认 q=1。q 为负值代表从动主轴与主动主轴旋转方向相反。

r：主动主轴与从动主轴相位差。r 满足条件：-360.0 < r < 360.0

若 R 编入，则进行相位控制；若 R 未编入，根据以下情况决定是否要进行相位控制：

(1) 使用 SPRC1 且位参数 SPRC1NR(P2700.0)=0，或使用 SPRC2 且位参数 SPRC2NR (P2700.1)=0 时，进行相位控制，等同于输入 R0。

(2) 使用 SPRC1 且位参数 SPRC1NR (P2700.0)=1，或使用 SPRC2 且位参数 SPRC2NR (P2700.1)=1 时，不进行相位控制。

其中，相位控制附加速度由从动主轴的“定向或整圈停止电机转速”参数（P2776/P2826/P2876）确定。

nq：补偿增益

(1) nq 有效范围：1~100，单位：Hz

(2) 当无相位控制或非 1:1 同步主轴，指定 NQ 时报警。

(3) 当同时满足以下条件时，不指定 NQ 效果更好，指定反而更差：

1) 主动和从动主轴都为总线轴

2) 位参数 DSPSCi(i--主动主轴的轴号)=1

3) 非控制方式 0 时

4) 驱动器型号及驱动器软件版本相同

e/k（选配）：滚齿加工中斜齿圆柱齿轮加工，属专用机械加工。e 为螺旋角，k 为模数。

(1) e 有效范围：-50.0~50.0 度

(2) k 有效范围：0.01~100.0mm 或 0.01/25.4~100.0/25.4inch

(3) E 和 K 同时使用，仅输入一个时报警

(4) 当“G251EK”（P2701.0）为 0 时禁用 EK

- **停止 SPRC 控制器**

G250 NLnl Pp ；

[说明]

nl：选择 SPRC 控制器。有效数值：1~2，对应 SPRC1~SPRC2。

未编入默认同时停止 SPRC1 和 SPRC2。

p：发出停止信号，并等待 SPRC 控制器停止

p 有效范围：1

P 未编入时，向 SPRC 控制器发出减速停止信号，但不等待其停止结束

编入 P1 时，向 SPRC 控制器发出减速停止信号，并等待其停止结束

注：G250 自动恢复同步前从动轴转速

解释

- **坐标更新**

SPRC 控制期间，从动主轴跟随主动主轴运动的部分脉冲，计入机床坐标，不计入绝对坐标。但累计未计入部分，在 SPRC 控制结束时更新绝对坐标。

SPRC 控制器启动时刻，从动主轴绝对坐标清 0。停止时刻，将 SPRC 控制期间未计入绝对坐标的数据总和计入绝对坐标。

- **SPRC 启动说明**

1 启动后系统在标题栏中间以红色字体醒目提示工作中的 SPRC 控制器：SPRC1/SPRC2

注：当 SPRC1 和 SPRC2 都显示时，中间没有空格显示。

2 启动后，从动主轴立即进入轮廓控制模式

例如：

配置	第 1 主轴为主动轴(C 轴)，第 2 主轴为从动轴(Y 轴)
程序	G51.2 ; 1:1 控制，Y 轴进入轮廓模式，坐标清零 … G50.2 ; 虽停止 SPRC 控制器，但 Y 轴仍处于轮廓模式
其它说明	G51.2 与 G251 等价，G50.2 与 G250 等价，下同

3 启动后，当主动主轴为数字主轴时，那么主轴停止时保持使能状态。

例如：

配置 1	第 1 主轴为总线主动轴(C 轴)，DSPSC4=1，第 2 主轴为从动轴(Y 轴)
程序	M3 S500 ; G4 P2000 ; G51.2 ; G4 P5000 ; M5 ; C 轴停止后不断使能

配置 2	第 1 主轴为总线主动轴(C 轴)，DSPSC4=0，第 2 主轴为从动轴(Y 轴)
程序	M3 S500 ; G4 P2000 ; G51.2 R0 NQ20 ; G4 P5000 ; M5 ; C 轴停止后不断使能

配置 3	第 1 主轴为脉冲主轴(C 轴，P2730=2104)，第 2 主轴为从动轴(Y 轴)
程序	M3 S500 ; G4 P2000 ; G51.2 R0 NQ20 ; G4 P5000 ; M5 ; C 轴停止后不断使能

4 处于 SPRC 从动控制中的主轴，具有以下限制

(1) G27~G30/G53/G153 中编入该轴地址时报警

例如：

配置	第 1 主轴为主动轴(C 轴)，第 2 主轴为从动轴(Y 轴)
程序	G51.2 ; 1:1 控制，Y 轴进入轮廓模式，坐标清零 G28 Y0 ; 报警

(2) 提前转速控制时断使能功能无效

(3) 再次指令该轴为另一 SPRC 控制器将报警。系统限制主轴不允许同时为两个 SPRC 控制器的从动主轴

例如：

配置	第 1 主轴为主动轴(C 轴)，第 2 主轴为从动轴(Y 轴)
程序	G51.2 ; 1:1 控制 G51.2 NL2 ; 报警

(4) SPRC 控制停止前将一直保持使能状态。如果为总线从动主轴，那么附加转速控制时，即便 DSPSCi=0(i 为从动主轴轴号)，也时无法开环。

例如：

配置	第 1 主轴为主动轴(C 轴)，第 2 主轴为从动轴(Y 轴)
程序	G51.2 ; 1:1 控制 M3.2 S2=100 ; 从动轴附加 100 转/分。若为总线从动主轴，那么闭环控制 M5.2 ; 停止附加转动 M3.2 ; 再次附加转动 M61 ; 报警，因为将无法断开使能信号。

(5) 执行 M19/M61/M38/M39 时报警，也包括定向和 C/S 开关按键

例如：

配置	第 1 主轴为主动轴(C 轴)，第 2 主轴为从动轴(Y 轴)
程序	G51.2 ; 1:1 控制 M19 ; 主动主轴定向 M19.2 ; 报警

(6) 若此时使用 PLC 轴功能控制该从动主轴(包括手动移动该轴)，那么该控制立即进入轴锁状态。

5 处于 SPRC 控制中的主动或从动主轴，提前转速控制时断使能功能无效

6 从动主轴加减速过程中（非同步跟随阶段），该主轴的多段加减速参数及转速控制平滑时间有效。

7 当 SPRC 启动控制，且主动主轴未处于停止状态，但从动主轴未处于跟随过程中，那么系统自动进行切削锁住。

例如：

配置	第 1 主轴为主动轴(C 轴)，第 2 主轴为从动轴(Y 轴)
程序	M3 S200 ; 启动主动主轴 G51.2 P1 Q2 ; 1:2 控制 G01 Z-10. F100. ; 待 Y 轴速度稳定(处于 SPRC 从动跟随过程中)后切削

注意：

配置	第 1 主轴为主动轴(C 轴)，第 2 主轴为从动轴(Y 轴)
程序	G51.2 ; 1:1 控制，主动主轴处于停止状态 G1 Z-10. F100. ; 虽从动主轴未处于跟随状态，但因主动主轴处于停止状态，不会进行切削锁住

8 当 SPRC 启动控制后，可再次指定 G251，给入新的控制器数据

例如：

配置	第 1 主轴为主动轴(C 轴)，第 2 主轴为从动轴(Y 轴)
程序	M3 S400 ; G4 P1000 ; G51.2 Q2 ; Y 轴 800 转/分 G4 P1000; G51.2 P2 ; Y 轴直接从 800 转/分直接降到 200 转/分 G4 P1000 ; G51.2 ; Y 轴直接从 200 转/分直接升到 400 转/分

- **SPRC 停止说明**

1 通过代码停止

例如：

配置	第 1 主轴为主动轴(C 轴)，第 2 主轴为从动轴(Y 轴)
程序	G0 X20. Z0. ; M3 S400 ; G51.2 P2 ; Y 轴 200 转/分 G01 Z-10. F500. ; G50.2 ; 停止，不等待 G0 X25. ; G50.2 P1 ; 等待 SPRC 控制器，可省略 G0 Y180.0; Y 轴定位到 180 度
其它说明	“G50.2 P1;”段可省略。下一段指令 Y 轴移动时，系统因为检测到 Y 轴为 SPRC 控制器控制的从动主轴且处于减速状态，所以进行了等待，以保证“G0 Y180.0;”正确运动到 Y180.0。

2 通过 G 接口停止

PLC 可在复位时接通 SPnRSPRC(n 为从动主轴轴号)，这样复位时就可以停止相应 SPRC 控制器。

3 有补偿时，且除 SPRC 控制器外有脉冲输出，报警并停止 SPRC 控制器。

4 主动或从动主轴报警时，停止 SPRC 控制器。

相关参数

2700							SPRC2NR	SPRC1NR

[数据类型] 位型

1 SPRC2NR SPRC2 控制器未输入 R 时，默认 R0/不进行相位控制

0 SPRC1NR SPRC1 控制器未输入 R 时，默认 R0/不进行相位控制

2769	(主轴 1)到位允许误差
2819	(主轴 2)到位允许误差
2869	(主轴 3)到位允许误差

指令补偿(NQ)时，从动主轴到位允许误差在补偿控制时使用。当误差绝对值大于到位允许误差时，进行补偿。

注意事项

1. G00 快速、手动和手轮运动会进行 Z 向补偿。
2. 当 G251 使用配置为 2104（带编码器反馈的脉冲主轴），且无相位控制（R），无补偿（NQ）时，使用主动轴的输出脉冲，而不是反馈脉冲。

6.1.1 应用举例

多边形加工

工件轴：第 1 主轴，为主动主轴

刀具旋转轴：第 2 主轴，为从动主轴

位参数 SPRC1NR(P2700.0)=0，目的：省略 R0 编入。

示例：

配置	第 1 主轴为主动轴(C 轴)，第 2 主轴为从动轴(Y 轴)
程序	O0251 ; M3 S100 ; 工件轴旋转速度 100 转/分 G0 X50. Z0. ; 快速定位 G51.2 P1 Q2 ; 刀具旋转轴以速度 200 转/分同步旋转，执行相位同步，但无补偿 G01 Z-10. F100. ; Z 轴进给 G0 X80. ; X 轴回退 G50.2 ; 刀具旋转轴同步解除，刀具停止 G0 Z100. ; Z 轴退回安全位置 M30 ;

同步夹持工件

工件轴：第 1 主轴，为主动主轴

刀具旋转轴：第 2 主轴，为从动主轴

示例：

配置	第 1 主轴为主动轴(C 轴)，第 2 主轴为从动轴(Y 轴)
程序	O0251 ; … 第 1 主轴卡盘关闭 M 代码 由 PLC 实现 …

	M3 S1500 ; 第 1 主轴旋转 … G51.2 R0 NQ10 ; 同步控制，且有相位同步，有补偿 … 第 2 主轴卡盘关闭 M 代码 PLC 需确认 F2208.1(SP2SYNC)=1，然后关闭卡盘
说明	在主轴同步夹持模式中，主动主轴与从动主轴同时夹持相同工件时，请勿更改主轴转速或停止主轴。否则可能因变速过程中，同步误差变大，而引发非常危险的情况。

主轴重叠控制（承载攻丝）

在主轴重叠方式下，进行攻丝/钻孔等，以提高效率。

工件轴：第 1 主轴，为主动主轴

刀具旋转轴(攻丝轴)：第 2 主轴，为从动主轴

示例：

配置	第 1 主轴为主动轴(C 轴)，第 2 主轴为从动轴(Y 轴)
程序	O0251 ; M3 S500 ; 工件轴旋转速度 500 转/分 … G51.2 ; … G0 X0 Z2. ; G84 Z-20. F2. S2=300 NS2 ; 重叠后，攻入时 800 转/分，攻出时 200 转/分 G250 ; M5 ; M30;

平齿轮加工（专用机械）

工件轴：第 1 主轴，为主动主轴

刀具旋转轴(滚齿轴)：第 2 主轴，为从动主轴

示例：

配置	第 1 主轴为主动轴(C 轴)，第 2 主轴为从动轴(Y 轴)
程序	O0251 ; M3 S500 ; 工件轴旋转速度 500 转/分 G0 X40. Z5. ; 快速定位 G51.2 P10 R0 ; 滚齿轴 50 转/分同步旋转，执行相位同步 G01 Z-10. F10. ; 第 1 次切入 G0 X44. ; Z5. ; X38. ; G01 Z-10. F10. ; 第 2 次切入 G0 X44. ; …

	G50.2 ; 刀具旋转轴同步解除，刀具停止 … M30 ;

斜齿轮加工（专用机械）

刀具旋转轴(滚齿轴)：第 1 主轴，为主动主轴

工件轴：第 2 主轴，从动主轴

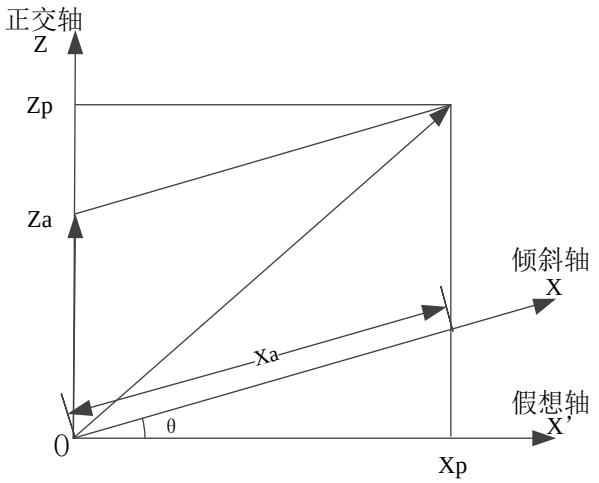
其它加工参数：齿数 P=68，滚齿条数 Q=3，螺旋角 E=18 度，模数 K=2.5mm，工件左旋，滚刀右旋。注：P/Q 不可约分。

示例：

配置	第 1 主轴为主动轴(Y 轴)，第 2 主轴为从动轴(C 轴)
程序	O0251 ; G250 ; M3 S160 ; 刀具旋转速度 500 转/分 G0 X20. Z3. ; 快速定位 G251 NP12 P68 Q-1 E18.0 K2.5 R0; 启动滚齿加工，执行相位同步 G00 X-5.5 ; G1 Z-80. F0.050; 第 1 次切入 G0 X20. ; G1 Z3. F3. ; 用 G1 回定位位置，相当于 C 轴补偿量归零(不能用 G0) G0 X-8.5 ; G01 Z-80. F0.050 ; 第 2 次切入 G0 X20. ; … G50.2 ; 刀具旋转轴同步解除，刀具停止 … M30 ;

7 斜轴功能

斜轴功能：数控机床上，由于结构设计的需要，部分坐标轴可能会采用倾斜式布局，倾斜轴的运动将导致（虚拟）笛卡儿坐标系中的两个坐标轴同时产生移动。系统在倾斜轴运动时，能够自动调整坐标尺寸，这个功能称为斜轴控制功能。



解释

以 X 轴倾斜为例，如下图所示，假设倾斜角为θ，运动起点为$(0,0)$。当编程的终点坐标为(X_p, Z_p)时，各轴实际运动距离(X_a, Z_a)的计算公式为：

当 X 轴的移动距离的计算式为：

$$X_a = \frac{X_p}{cos\theta}$$

沿 Z 轴的移动距离由 X 轴的倾角修正，按下式计算：

$$Z_a = Z_p - X_p \times \tan\theta$$

沿 X 轴的进给速度分量按下式计算：

$$F_a = \frac{F_p}{cos\theta}$$

式中X_a，Z_a，F_a：实际的距离和速度

Xp，Zp，Fp：编程的距离和速度

注

1 （虚拟）笛卡儿坐标系ZOX′为程序坐标系，斜坐标系ZOX为实际移动坐标系。

2 公式中轴坐标为半径值。

实际在移动斜轴时，必定引入正交轴坐标的移动，当用户只想假想轴移动，希望正交轴不移动时，系统内部需处理绝对坐标中正交轴的移动量。

格式

G00/G01/G31 U_ F_ NP1

NP=1 时，根据 U_（假想轴上移动量，相对编程）自动计算在正交轴上的移动量，合成斜轴上的独立运动

例

P413=21，P414=30

X、Z 回零后，执行程序：

G1 U10 F100 NP1;

结束后：

绝对坐标：X10.000 Z2.887

机床坐标：X11.547 Z0.000

斜轴移动量：

$$\frac{10}{\cos(30°)} = 11.547$$

正交轴移动量（系统内部处理）：

$$\frac{10}{2} \times \tan(30°) = 2.887$$

步骤

斜轴控制中的纵轴和斜轴必须在参数 P413 设定，十位数—正交轴轴号，个位数—倾斜轴轴号。沿各轴的移动距离根据倾斜角度 P414 设定。例如：Z 轴为正交轴，X 轴为倾斜轴，假定倾斜角度为+30 度。那么设置 P413=21，P414=30.0。

坐标说明

机床坐标总是为斜坐标系中实际坐标，相对坐标或绝对坐标总是为（虚拟）笛卡儿坐标系中的程序坐标。

1 斜轴与正交轴独立运动

以下情况独立运动：

(1) 回参考点(程序回参考点/机械回参考点，包括 G28 回参考点)

(2) 按机床坐标运动(G53/G153)

(3) 非自动和录入方式下，G1250.0(AACSEPG)=1

该情况下，指令斜轴轴运动时(例如：X 为斜轴，指令 X20.0)，斜轴机床坐标变化，正交轴机床坐标不会跟随变化(即不动)，但绝对坐标会发生变化。

注

参考点有效时，斜轴与正交轴的独立运动，不会导致需要重新回参考点。

2 按（虚拟）笛卡儿坐标系运动

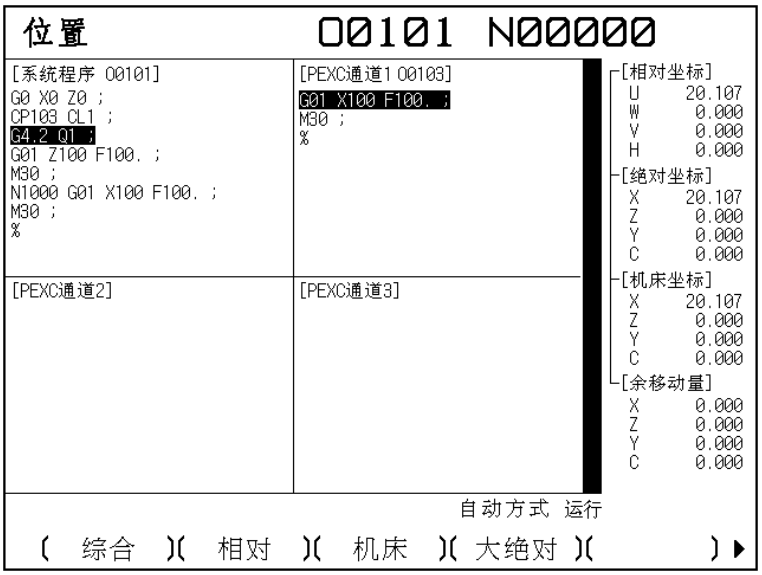
指斜轴功能有效时非独立运动情况。该情况下，指令斜轴轴运动时(例如：X 为斜轴，指令 X20.0)，斜轴绝对坐标正常变化，不会引起正交轴绝对坐标变化，但是斜轴和正交轴机床坐标都将变化(即两轴都运动)。

手动或自动返回参考点

倾斜轴和正交轴回参考点无先后次序要求。

8 并行控制器(PEXC)功能

系统设计有 8 个并行控制器(PEXC)：PEXC1/PEXC2/…/PEXC8。PEXC 是用于部分替代(今后将逐渐完善直至全部替代)原 PLC 轴控制。可由 PLC 使用，也可由 NC 指令调用。



PEXC 通道运行示意图

相关参数

P6					PECPCN			

[数据类型] 位型

3 PECPCN NC 调用 PEXC 通道功能有效/无效

0：无效

1：有效

注

当该参数设置为 0 时，NC 调用 PEXC 通道时，报警 PS775：PEXC 错误，NC 调用 PEXC 通道功能无效。

P18	PLC 轴功能接口选择

0--无效/1--G、F 接口/2--使用 PEXC 并行执行控制器

P19	DPEXC8	DPEXC7	DPEXC6	DPEXC5	DPEXC4	DPEXC3	DPEXC2	DPEXC1

[数据类型] 位型

DPEXCn 并行通道 n 无效/有效

0：有效

1：无效

P406							PEXCn	

[数据类型] 位型

1 PEXCn n 轴是/否由 PEXC 控制轴

0：否

1：是

注

当前轴该参数为 1 时，若主通道指令该轴运动，则报警 PS776：主通道不能指令 PEXC 轴运动，请查看 P406.1。

P2303	PLCPATH							

PLCPATH

1/0：是/否显示 PLC 通道页面

注：当 P18=2 时，设置该参数将显示 PEXC 页面。

P2751	PEXCIV8	PEXCIV7	PEXCIV6	PEXCIV5	PEXCIV4	PEXCIV3	PEXCIV2	PEXCIV1

PEXCIVn

1/0：PEXC 通道 n 使用独立/公用保持宏变量(#500-#999)

注：每个 PEXC 通道均支持使用各自通道的非易失性宏变量，仅支持通过程序方式或配合桁架功能使用。

8.1 通道说明

原先 NC 通道为主通道，加上 8 个 PEXC 通道，共 9 个通道。

通道编号	说明
0	主通道
1	PEXC1
2	PEXC2
3	PEXC3
4	PEXC4
5	PEXC5
6	PEXC6
7	PEXC7
8	PEXC8

8.2 PEXC 功能

G 代码支持

组号	G 代码		功能
	铣床或车床 B 体系	车床 A 体系	
01 组 (模态)	G00		直线定位
	G01		直线插补
	G02		顺时针圆弧插补(CW)
	G03		逆时针圆弧插补(CCW)
	G102(G02.1)		空间圆弧插补

00 组(一次性)	G04		暂停
	G28		自动返回参考点
	G30		返回第 2 第 3 参考点
	G31		跳跃机能
	G53		机床坐标系定位
	G153(G53.1)		快速机床坐标系定位
	G104(G4.1)		通道等待命令
	G204(G4.2)		通道结束等待命令
	G954(G54.9)		笛卡儿坐标系坐标轴设定
	G251		支持
	G250		支持
	G351		支持
	G350		支持
02 组	G17/G18/G19		加工平面选择
03 组	G90/G91		绝对值编程/增量值编程
04 组	G98	G98.1	在固定循环中返回初始平面
	G99	G99.1	在固定循环中返回 R 点平面
07 组	G40		不支持，忽略
	G41		不支持，报警
	G42		不支持，报警
08 组	G49		不支持，忽略
	G43		不支持，忽略
	G44		不支持，忽略
09 组	G80		取消钻孔循环
	G81		钻孔循环，点钻循环
	G82		钻孔循环、镗阶梯孔循环
	G83		深孔加工循环
	G84		攻丝循环
	G85		镗削循环
	G86		镗削循环
	G87		镗削循环、反镗循环
	G88		镗孔循环
	G89		镗孔循环
	G73	G83.5	高速深孔加工循环
	G74	G84.5	反攻丝循环
	G76	G86.5	精镗循环
10 组	G94	G98	每分进给(不支持，忽略)
	G95	G99	每转进给(不支持，报警)
14 组		G96	恒线速 ON(不支持，报警)
		G97	恒线速 OFF(不支持，报警)

注：PEXC 通道 G00 定位采用直线，与位参数 LPR(P4.1)无关。

其它功能

功能	是否支持	补充说明
M 代码	是	过程控制 M 代码仅支持 M99 和 M30。 以下过程控制 M 代码不支持： M00/M01/M02 M06 M91~M94(车床) M96/M97 M98
S 代码	是	
T 代码	否	
子程序	否	
手轮调试	NC 调用支持	
自动手轮	是	G 接口：G3500n.6；含义：PLC 调用的 PEXC 程序自动手轮功能是否有效（G13.6 是自动手轮功能的总开关） 说明： 1）NC 调用的 PEXC 通道程序与 NC 程序视为一体，自动手轮控制与之前相同； 2）PLC 调用的 PEXC 通道程序，根据需要打开 G3500-n.6；
暂停控制	支持	G 接口信号：PAUSE_1~ PAUSE_8
单段控制	支持	G 接口信号：SBK_1~ SBK_8
再启动控制	支持	G 接口信号：RSTART_1~ RSTART_8
宏程序	支持	限制： 局部变量不可使用，否则会与主通道局部变量混乱，后续改进支持。 全局变量不要与主通道使用相同的全局变量，否则会出现变量混乱。 译码后变量就写入。 后续改进宏程序的使用

速度和加速度说明

速度和加速度与主通道使用参数相同。需要注意的是参数 P6460/P6461/P6463 虽然对 PEXC 生效，

但实现不同，效果略有不同。参考下面说明：

1 PEXC 快速加减速时间 S 曲线时间 N=P6412+P6460+P6461

2 PEXC 刚性攻丝加减速 S 曲线时间 N=P6414+P6463

3 P6460 对 PEXC 不再是指数型，只是简单解决振动问题

4 G31 跳跃信号到来时，编入 AD 指令，进行减速控制，而不是立即停止脉冲输出；未编入 AD 会立即停止脉冲输出

宏变量说明

1 写全局变量 100~1000 时记录通道号，以便检查各通道写全局变量冲突，冲突时报警 732；

2 禁止 PEXC 通道进行子程序调用（包括 M98，M06 和自定义宏程序调用），调用时报警 733；

3 增加 PEXC 局部变量显示，输入输出功能；

8.3 PEXC 通道调用

所有 PEXC 通道，有以下三种调用方式，但需错开使用。

1 NC 调用(CP/CN/CL 指令)

2 梯级调用

3 G 接口调用(G3480~G3494)

方式 2 和方式 3 合称“PLC 调用”

8.3.1 调用方式 1：NC 调用(CP/CN/CL 指令)

格式

CPcp CNcn CLcl ;

cp：指定调用程序号 1~9999。

超出范围报警 PS700。未编入默认当前程序，但如果当前程序是 0 号程序，那么报警 PS707。

cn：指定开始段顺序号，非 0 值。

指定 0 时报警 PS701。未指定时，开始段为调用程序的第 1 段。

cl：指定 PEXC 通道编号 1~8。

超出范围报警 PS702，未编入默认使用 PEXC1。

注

1 调用语句不要求单独一行，在所在段开始执行时进行调用。

2 CP 和 CN 应至少编入 1 个。

3 开始段不仅是调用开始执行处，也是通道程序中 M99 重复执行处。

复位

以下情况引发复位(减速停止，有报警时复位报警)：

1 急停信号(G27.4 *ESP)有效

2 ERS_i 信号有效

3 复位按键有效。包括：MDI 复位按键、机床复位按键(X70.7)、PLC 复位按键(G29.4)

注：ERS(G27.7)仅对主通道有效。

示例

示例 1

主通道(NC 通道)	PEXC2 通道
O101 G0 Z0 CP103 CL2； 定位，并用 PEXC2 执行 O103 G4.2 Q2 ； 等待 PEXC2 结束 X0 Y0 Z100.； M30 ；	O103 G54.9 P124 ； G03 X40. Y20. R-30. F2000. ； M30 ；

示例 2

主通道(NC 通道)和 PEXC1 通道编在同一程序
O101 G0 Z0 CN1000； 定位至 Z0，并用 PEXC 执行 N1000 开始至程序结束之间的程序块 G4.2 Q1 ； 等待 PEXC1 结束 X0 Y0 Z100. ； M30 ； N1000 G54.9 P124 ； G03 X40. Y20. R-30. F2000. ； M30 ；

示例 3

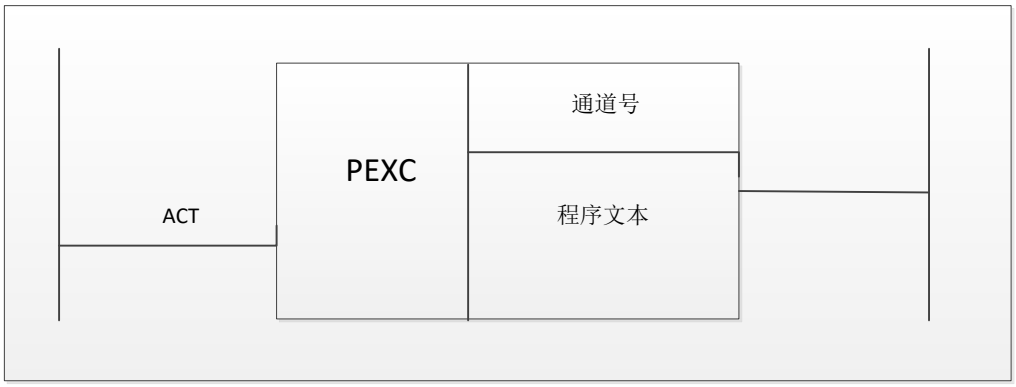
主通道(NC 通道)和 PEXC1 通道编在同一程序
O101 G0 Z0 CN1000； 定位至 Z0，并用 PEXC 执行 N1000 开始至程序结束之间的程序块 G4.2 Q1 ； 无法等到 PEXC1 结束，除非复位。 X0 Y0 Z100. ； M30 ； N1000 G1 X100. F2000. ； 循环起始段 N1001 X0. ； N1002 M99 ； 循环，跳至 N1000 继续执行

8.3.2 调用方式 2：PLC PEXC 指令调用

PEXC 指令

功能：PLC 通道程序并行执行指令函数；

- **格式**



输入说明

ACT： 输入信号，ACT 有效时 PEXC 执行通道程序（当 ACT 为 0 会停止执行）；

通道号： 常量或变量，范围 1~8；

程序文本： 通道执行程序文本；

输出说明： 无输出。

注

1 每一个通道可以最多拥有 32 个并行程序，但一次只能执行一个并行程序；

2 同一个通道的多个并行程序只能同时接通一个，同一通道内切换到另一个程序时需要间隔一个 PLC 周期；

3 并行程序暂不支持在线编辑，请在 PC 端编辑；

4 须配合 KNDPLC V5.5 以上版本使用

复位

以下情况引发复位(减速停止，有报警时复位报警)：

1 急停信号(G27.4 *ESP)有效。不复位调用逻辑错误。

2 ERS_i 信号有效。不复位调用逻辑错误。

3 ACT=0。复位调用逻辑错误。

注：ERS(G27.7)仅对主通道有效。

8.3.3 调用方式 3：PLC G 接口调用

G 接口

G3480	(PEXC1)调用程序号
G3482	(PEXC2)调用程序号
G3484	(PEXC3)调用程序号
G3486	(PEXC4)调用程序号
G3488	(PEXC5)调用程序号
G3490	(PEXC6)调用程序号
G3492	(PEXC7)调用程序号
G3494	(PEXC8)调用程序号

数值非 0 为调用程序号，数值 0 为复位调用。

注：可通过将调用程序号清零实现通道复位，通过将通道倍率清零实现通道的暂停

复位

以下情况引发复位(减速停止，有报警时复位报警)：

1 急停信号(G27.4 *ESP)有效。不复位调用逻辑错误。

2 ERS_i 信号有效。不复位调用逻辑错误。

3 调用程序号变更为 0。复位调用逻辑错误。

注：ERS(G27.7)仅对主通道有效。

8.4 通道等待功能

8.4.1 G4.1 通道等待命令

该命令可用于主通道和所有 PEXC 通道。

格式

G4.1 Pp Ll Qq ;

p：等待信号编号，1~999，未输入或超出范围报警。

l：等待通道数 1~3，未输入默认 1。

q：等待通道编号，每两位十进制为 1 个通道编号。通道编号参考前面的通道说明。

G4.1 应单独一行，包含以下地址时报警：

1 M/S/T

2 轴地址

3 其它 G 代码

示例

主通道程序	PEXC1 程序
O0001 ; G0 X0 Y0 ; CP1000 ; G1 X100. F1000. ; G4.1 P1 Q1 ; 等待 PEXC1 通道 P1 信号 X0. ; … M30 ;	O1000 ; G1 Y100. F1000. ; G4.1 P1 Q0 ; 等待主通道 P1

主通道程序	PEXC1 程序	PEXC2 程序
O0001 ; G0 X0 Y0 ; CP1000 ; CP1001 L2; G1 X100. F1000. ; 等待 PEXC1/2 通道 P1 信号	O1000 ; G1 Y100. F1000. ; 等待主通道及 PEXC2 P1 G4.1 P1 L2 Q0002 ;	O1001 ; G1 Z100. F1000. ; 等待主通道及 PEXC1 P1 G4.1 P1 L2 Q0001 ;

G4.1 P1 L2 Q0102 ; X0. ; … M30 ;		

8.4.2 G4.2 等待 PEXC 通道结束命令

该命令可用于主通道和所有 PEXC 通道。

格式

G4.2 Qq ;

q：等待通道编号，每两位十进制为 1 个 PEXC 通道编号(1~8)，但不包括主通道(通道 0)。通道编号参考前面的通道说明。

G4.2 应单独一行，包含以下地址时报警：

1 M/S/T

2 轴地址

3 其它 G 代码

Q0001 等价于 Q1，高位 0 忽略

主通道遇到 M30/M02/EOR 时，自动等待所有 NC 调用(不包括 PLC 调用)的 PEXC 通道结束

示例

主通道程序	PEXC1 程序
O0001 ; G0 X0 Y0 ; CP1000 ; G1 X100. F1000. ; G4.1 P1 Q1 ; 等待 PEXC1 通道 P1 信号 X0. ; G4.2 Q1 ; 等待 PEXC1 执行完 G0 X0 Y0 ; M30 ;	O1000 ; G1 Y100. F1000. ; G4.1 P1 Q0 ; 等待主通道 P1 M30 ;

8.5 PEXC 通道 S 代码支持

PEXC 通道中支持 S 代码编程，其使用方式与 NC 通道一致。

注 1：诊断 N511/N531/N551 为当前 S 代码值，N512/N532/N552 为当前 S 代码值发出的通道号(0--NC 主通道, !=0--PEXC 通道号)

8.6 PEXC 通道 G/F 接口

G 接口

以下接口对三种调用方式都有效。

G3500	DISAXLK_1	ENATHD_1	MDCHG_1		RSTART_1	SBK_1	PAUSE_1	ERS_1
G3502	(PEXC1)快速倍率							
G3503	(PEXC1)切削倍率							

G3520~G3643 为 PEXC2~8 通道 G 接口信号，同 PEXC1

ERS_i (i=1~8)：PEXCi 外部复位信号

PAUSE_i (i=1~8)：PEXCi 暂停信号

注

PEXC 执行 G28 时，从中间点向零点移动过程中，出现暂停信号时，按照零速处理，不显示“暂停”。G28 以后将在 PEXC 中单独增加，不在调用 PLC 通道处理，此处暂时以零速处理。

SBK_i (i=1~8)：PEXCi 单段信号

注

单段停止后，手轮仍然可以回退，以后再考虑是否禁止。此处暂不处理。

RSTART_i (i=1~8)：PEXCi 单段或暂停的再启动信号

MDCHG_i (i=1~8)：NC 调用时模式切换是否暂停

ENATHD_i (i=1~8)：PLC 调用时自动手轮功能有效

注

NC 调用的 PEXC 通道程序与 NC 程序视为一体，自动手轮控制与之相同；PLC 调用的 PEXC 通道程序，根据需要打开“ENATHD_i”。

DISAXLK_i (i=1~8)：PEXCi 禁用轴锁相关信号(G29.6/G29.5/G33)

F 接口

以下接口用于 PLC 调用(“梯级调用”和“G 接口调用”)。

F3500	NERST_1	PFIN_1		LOGE3_1	LOGE2_1	LOGE1_1	PEALM_1	START_1

F3520~F3640 为 PEXC2~8 通道 F 接口信号，同 PEXC1

START_i (i=1~8)：PEXCi 通道工作中

PEALM_i (i=1~8)：PEXCi 通道报警中

LOGE3_i~ LOGE1_i(i=1~8)：PEXCi 逻辑错误码。下表以 PEXC1 为例对逻辑错误码进行说明。

PEXC1 逻辑错误码 (LOGE3_1~ LOGE1_1)	**含义**
0	正常
1	[来源]梯级调用

	[原因]至少有两条 PEXC 指令使用通道 i，且 ACT 输入信号都有效。注：如果需要同时执行多条 PEXC 指令，请安排多个 PEXC 通道并行执行。
2	[来源]梯级调用和 G 接口调用 [原因]同时使用梯级调用和 G 接口调用。注：一个时刻只能选择一种调用。如果需要并行执行，请安排多个 PEXC 通道并行执行。
3	[来源]梯级调用或 G 接口调用 [原因]当前 PLC 调用尚未复位时，给入了新的 PLC 调用。
4	[来源]梯级调用或 G 接口调用 [原因]当前调用(三种调用都有可能)尚未复位时，给入了新的 PLC 调用。
5	[来源]梯级调用或 G 接口调用 [原因]当前调用(三种调用都有可能)有报警尚未复位时，给入了新的 PLC 调用。

PFIN_i (i=1~8)：1: PEXCi 通道完成。

NERST_i (i=1~8)：1: PEXCi 通道有逻辑错误码(BIT2~4)或报警(BIT1)需复位。PLC 根据该接口进行复位。PEALM_i 和 LOGE2_i~ LOGE1_i 仅用于判断问题来源。

9 机床补偿控制器

用于补偿机床位置或磨损(如：砂轮磨损)。

9.1 相关接口

9.1.1 控制接口

机床补偿控制器通过宏变量接口完成。

变量说明：

变量号	读取	写入
9900	无效	写入 33335555 时，开启机床补偿控制器。之后若需要关闭改功能，只能重新上电。开启后，再次写入的数值被忽略。
9906	主通道最后半径补偿号。 返回主通道 G41/G42 有效时最后一段的半径补偿号	无效
9907	主通道当前补偿方向 -1—G42 0— G40 1— G41 注：#9920 半径补偿功能根据该方向确定半径补偿方向。	无效
9908	补偿平滑时间，单位：ms	补偿平滑时间，单位：ms。范围：16~200 注：超出数值范围时，写入失败
9910	无效	所有轴补偿速度，单位 mm/min。 数值不能小于 100mm/min，也不能超过切削进给上限速度 P713。 注 1：超出数值范围时，写入失败。 注 2：等同于 9911~9918 写入相同值。
9911~9918	1~8 轴补偿速度，单位：mm/min。	1~8 轴补偿速度，单位：mm/min。 数值不能小于 100mm/min，也不能超过切削进给上限速度 P713。 注 1：超出数值范围时，写入失败。 注 2：目前旋转轴不支持
9920	补偿半径，单位：mm	刀具(或砂轮等)半径补偿值，单位：mm。 注 1：此为绝对值，实际补偿值为该变量的变化量。写入 0 将取消半径补

变量号	读取	写入
		偿量。以下情况不计算半径补偿矢量，保持最后的补偿矢量： 1) 攻丝过程中(主通道) 2) G31/G36/G37/G53/G153(主通道) 注：G41/G42 半径补偿时，无法执行 G31，会报“刀偏时指令了 G31”。 3) 螺纹(主通道) 4) 快速运动(主通道) 5) 处于 G40 状态(主通道) 注 2：处于 G41/G42 状态时，数值>0，增大半径补偿量，数值<0 时，减小半径补偿量。 注 3：快速绘图时写入忽略
9921~9928	长度补偿值，单位：mm	长度补偿值，单位：mm。 注 1：此为绝对值，实际补偿值为该变量的变化量。写入 0 将取消该轴长度补偿量。 注 2：快速绘图时写入忽略

限制

1 目前旋转轴和被拆除的轴不支持

2 有一个控制轴产生伺服报警时，其它控制轴也报警，同时清除长度补偿和半径补偿量。

3 绝对编码器 TU2 时如果有该轴的补偿量(半径或长度)，那么报警(报警会清除补偿量)，复位后重新执行即可。

4 MACC 补偿运动虽可视为切削运动，但绘图时同 PLC 轴运动一样按快速绘制。

9.1.2 补偿平面

可以通过 G141(G142EN)/G142 接口指定补偿平面，也可以通过 G154 代码在程序中指定补偿平面。

注：G17/G18/G19 代码有效

9.1.3 诊断接口

主通道切削切矢量诊断接口

N320								TVVALID

TVVALID： (MACC)1/0—切矢量有效/无效

N330	(MACC)切矢量 1 轴分量
N331	(MACC)切矢量 2 轴分量
…	…
N337	(MACC)切矢量 8 轴分量

补偿诊断接口

N321	(MACC)半径补偿值(单位：mm)
N322	(MACC)半径补偿补偿平面第 1 轴轴索引
N323	(MACC)半径补偿补偿平面第 2 轴轴索引
N324	(MACC)补偿平滑时间

N340	(MACC)1 轴长度补偿值(单位：mm)
N341	(MACC)2 轴长度补偿值(单位：mm)
…	…
N347	(MACC)8 轴长度补偿值(单位：mm)

N350	(MACC)半径补偿矢量 1 轴分量(单位：mm)
N351	(MACC)半径补偿矢量 2 轴分量(单位：mm)
…	…
N357	(MACC)半径补偿矢量 8 轴分量(单位：mm)

N360	(MACC) 1 轴长度补偿量(单位：mm)
N361	(MACC) 2 轴长度补偿量(单位：mm)
…	…
N367	(MACC) 8 轴长度补偿量(单位：mm)

N370	(MACC) 1 轴余补偿量(单位：mm)
N371	(MACC) 2 轴余补偿量(单位：mm)
…	…
N377	(MACC) 8 轴余补偿量(单位：mm)

N380	(MACC) 1 轴本周期补偿脉冲(插补单位)
N381	(MACC) 2 轴本周期补偿脉冲(插补单位)
…	…
N387	(MACC) 8 轴本周期补偿脉冲(插补单位)

9.2 应用

示例程序：

```
%
O1 ;
#9900 = 33335555 ;          // 功能开启
G17 ;                       // 铣床 XY 平面
（注：也可以通过 G142 接口或者 G154 指令改变补偿平面）
#9910 = 1000;               // 补偿速度 1000mm/min
```

```
CP2 ;                          // 使用 PEXC 并行执行 O2 程序
G42 D1 G0 X0. Y-20. ;
G1 X20. F2000. NR6. ;
Y20. NR6. ;
X-20. NR6. ;
Y-20. NR6. ;
X0. ;
G40 G0 X20. ;
G4.2 Q1 ;                      // 等待 PEXC1 执行完
#98001 = #98001 - #9920 ;      // 将补偿量计入半径磨损
#9920 = 0;                     // 清除补偿量
M30 ;
%

%
O2
#1 = 1 ;
N10 G4 X0.5 ;
#9920 += 0.1 ;                 // 每 0.5s 增加半径补偿 0.1mm，实际加工可能使用主轴转矩控制等
#1 += 1 ;
IF [#1 < 10] GOTO 10 ;
M30 ;
%
```

10 程序

10.1 屏蔽子程序显示和屏蔽单段功能指令

功能描述

该功能用于调用子程序时，通过使用屏蔽指令，实现对子程序特定段进行显示屏蔽与程序单段运行功能的屏蔽。

格式

- **屏蔽子程序显示 MD 指令**

```
%
Oxxxx
MD1；          屏蔽子程序显示开始指令
•••
•••
•••
MD0；          屏蔽子程序显示取消指令
%
```

注

屏蔽显示指令仅对子程序有效或 M98L 功能在当前程序内调用的子程序块有效。

MD1/MD0 应在同一程序配对使用。

- **举例**

例：自定义 M 代码 M999 调用子程序 O9400

```
O0001
M999;
M30;

O9400
MD1;                    开始屏蔽显示
N10 G04 X100.0;
N20 G01 X500.0 F100.0;
N30 M99;
MD0;                    结束屏蔽显示
```

注

1 示例程序使用了屏蔽显示指令，开始屏蔽的 MD1 指令放在第一个 EOB 之前，结束屏蔽指令 MD0 可放在 M99 段前行、与 M99 同行、M99 段后，都可实现对整个子程序的屏蔽显示。N10-N30 段被屏蔽显示，屏幕显示状态为 O0001 界面，无法看到 O9400 内容。

2 屏蔽显示部分从 MD1 指令之后开始。

3 子程序执行结束后，调用子程序的程序顺序执行。

4 MD 状态复位即清除。

- **屏蔽程序单段 MS 指令**

```
%
Oxxxx
MS1;                    屏蔽程序单段功能指令
•
•
•
MS0;                    取消屏蔽程序单段功能
%
```

注

1 该指令不仅限于子程序中的使用，程序均可使用

2 MS1/MS0 需要在同一程序段配对使用

- **举例**

例：自定义 M 代码 M999 调用子程序 O9400

```
O0001
M999;
M30;

O9400
MS1;                    开始屏蔽程序单段功能
N10 G04 X100.0;
N20 G04 X100.0;
N30 G04 X100.0;
N40 G04 X100.0;
N50 M99;
MS0;                    结束屏蔽程序单段功能
```

注

1 示例程序使用了屏蔽单段功能指令，在单段方式下运行该程序，O9400 中 N10-N50 段单段功能被屏蔽，单段功能无效，程序在 N10-N50 段顺序执行。

2 子程序中使用时，如果希望从设置 MS1 处开始直至子程序返回都屏蔽单段，那么分两种情况：

(a) MS0 编写在 M99 的前一段；

(b) MS0 与 M99 编写在同一段（禁止 M99 快速跳转）。

3 对于因其他设置引起禁止 M99 快速跳转的情况（如参数 M99O=1、M99 段编写了 S/T 指令等），应采用情况 b，否则采用情况 a。

11 刀具半径补偿的详细说明

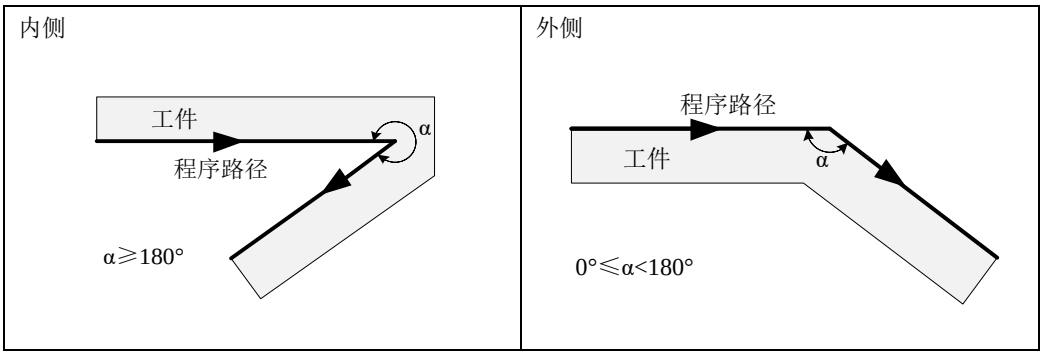
解释

- **拐角**

两个相邻移动程序段交点处的切线在工件实体一侧的夹角

- **内侧和外侧**

拐角大于或等于 180 度时称为“内侧”，在 0~180 度之间时称为“外侧”



- **补偿取消状态**

当系统初上电、复位或者执行了 M02、M30 指令，系统处于补偿取消状态。在补偿取消状态下，补偿向量的大小始终为 0，刀具中心路径与编程路径一致。程序的最后，必须以补偿取消状态结束。

- **补偿开始条件**

在补偿取消状态下，当满足以下条件的程序段开始执行，系统进入补偿模式。

1 含有 G41 或 G42 指令，或控制进入 G41 或 G42 模式。

2 刀具补偿的补偿号不为 0。

3 至少指令了补偿平面上一个轴（I、J、K 除外）的移动，且移动量不为 0。

注

在补偿开始程序段，不能指令圆弧指令 G02 和 G03，否则产生报警（PS0034）。

- **补偿结束条件**

在刀具半径补偿模式下，当指令了 G40 IP_或者 H00 或者 D00，刀具半径补偿即被取消。

注

刀具半径补偿取消时，不能指令圆弧指令，否则会产生报警（PS0034）且刀具停止。

- **系统内部程序段数**

补偿开始段，系统连续读入两个程序段，第一个程序段读入并执行，第二个程序段进入刀具补偿缓冲区。

1 在单程序段方式下，读入二个程序段，执行第一个程序段，然后停止。

2 在连续执行时，通常预先读入二个程序段，因此在 CNC 内部有三个程序段，一个为正在执行的程序段，下面的二个程序段进入缓冲区。

在补偿取消时，系统控制当前程序段以及刀具半径补偿缓冲寄存器中的程序段的执行。此时，如果选择单程序段模式，系统执行一个程序段后停止，再次按下起动按扭，系统直接执行下一个程序段而不用再预读。

图中符号

下文描述中，图中出现的各符号含义如下：

1 S 表示执行单程序段 1 次的位置

2 SS 表示执行单程序段 2 次的位置

3 SSS 表示执行单程序段 3 次的位置

4 L 表示刀具沿直线移动

5 C 表示刀具沿圆弧运动

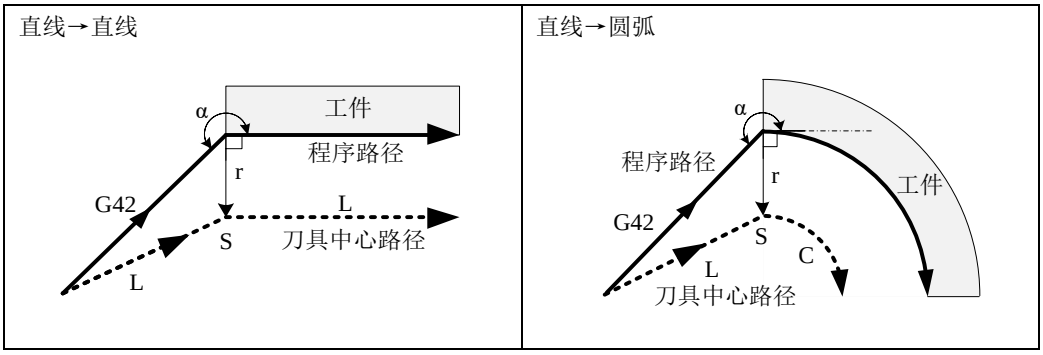
6 r 表示刀具半径补偿量

7 X 表示补偿路径交点

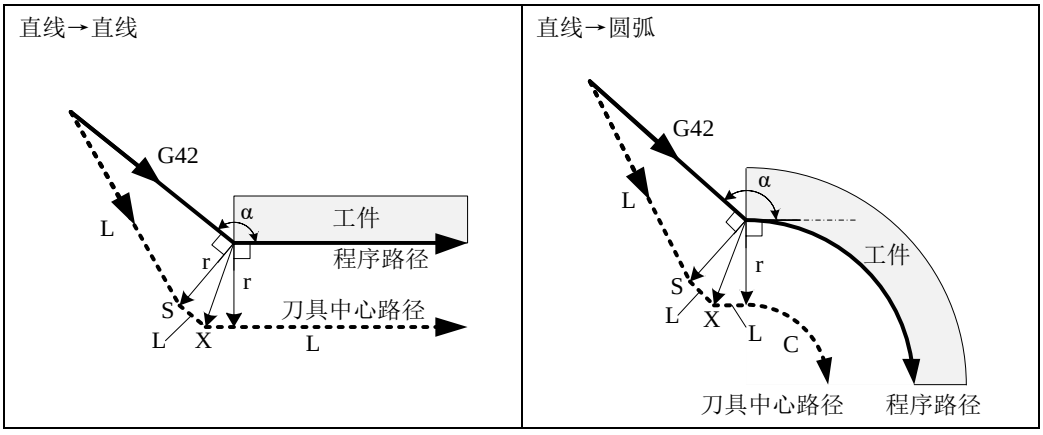
11.1.1.1 补偿开始时的刀具移动

以下说明从补偿取消状态进入补偿状态时的刀具移动方式。

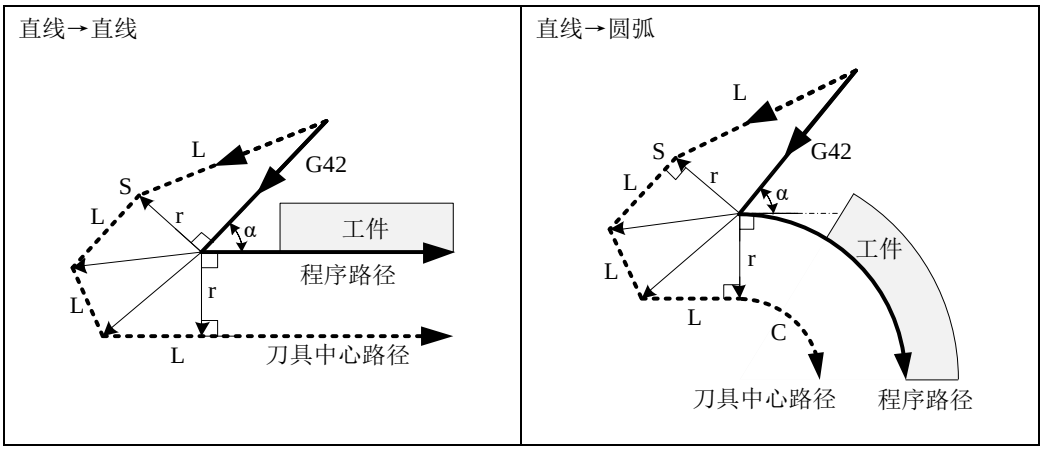
沿拐角的内侧移动（$\alpha \geq 180°$）



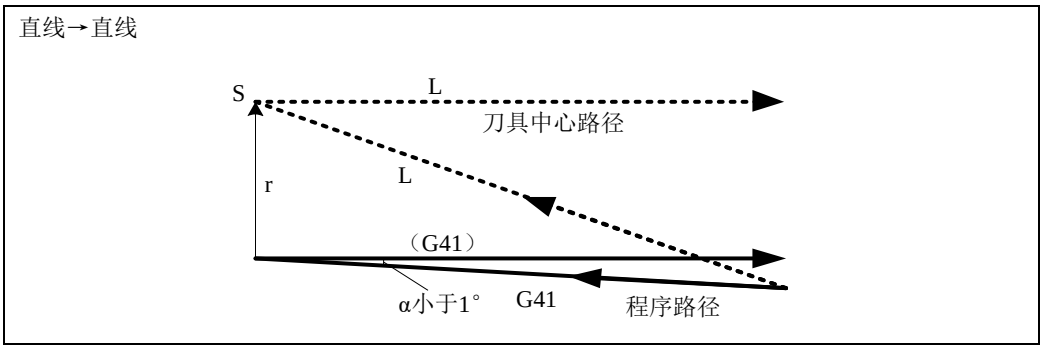
沿着拐角为钝角的外侧移动（$90° \leq \alpha < 180°$）



沿着拐角为锐角的外侧移动（α＜90°）



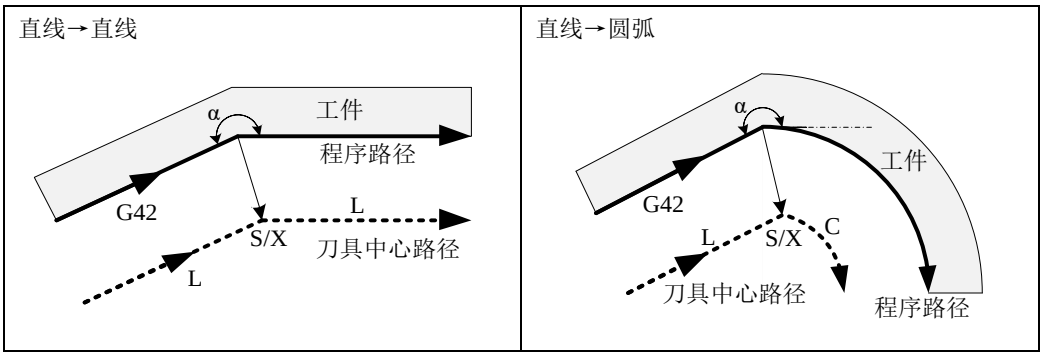
沿着拐角为小于1°的锐角外侧移动（α＜1°）

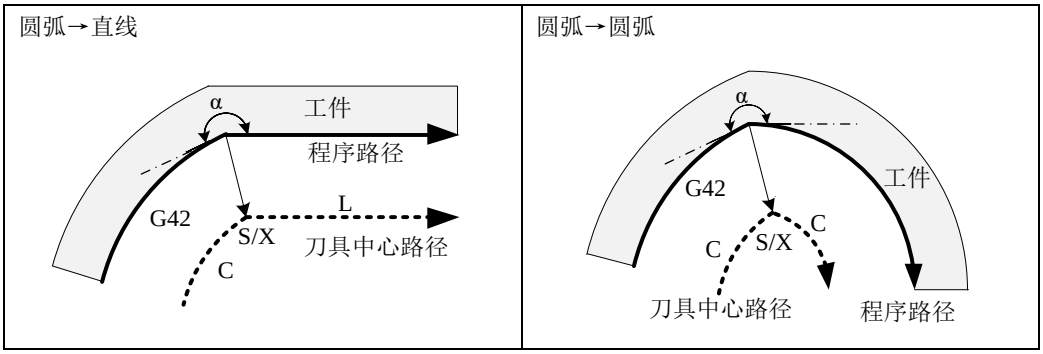


11.1.1.2 补偿模式下的刀具移动

在补偿模式，如果没有连续指定 2 个或 2 个以上的非移动程序段（辅助机能或暂停等），补偿仍会正确执行，否则可能会产生过切或切削不足。在补偿模式执行中不可变更补偿平面，否则会产生报警，同时刀具停止。

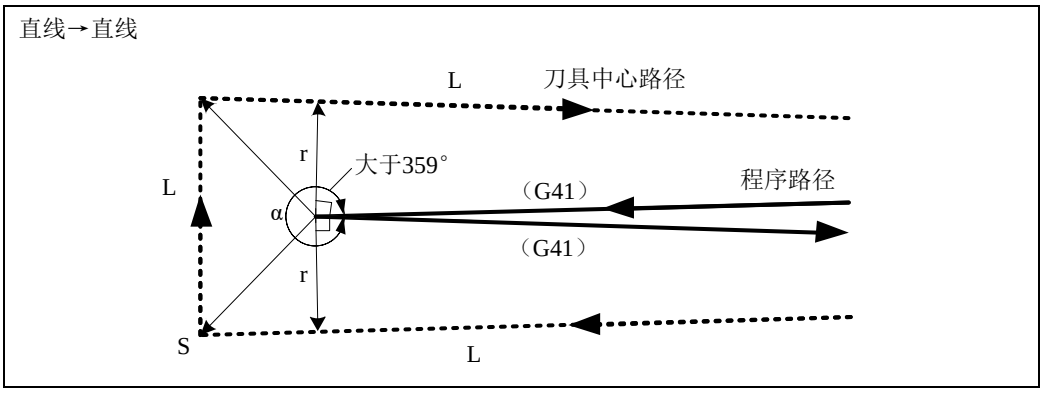
沿拐角的内侧移动（α≥180°）



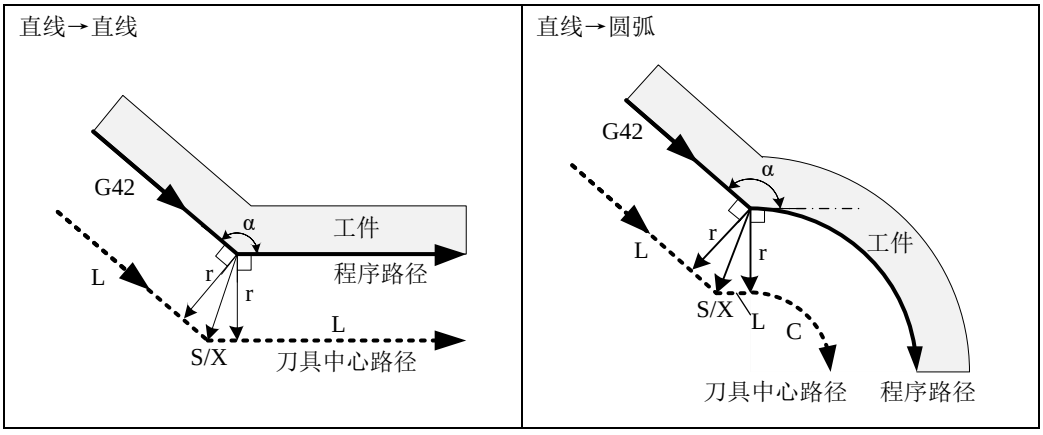


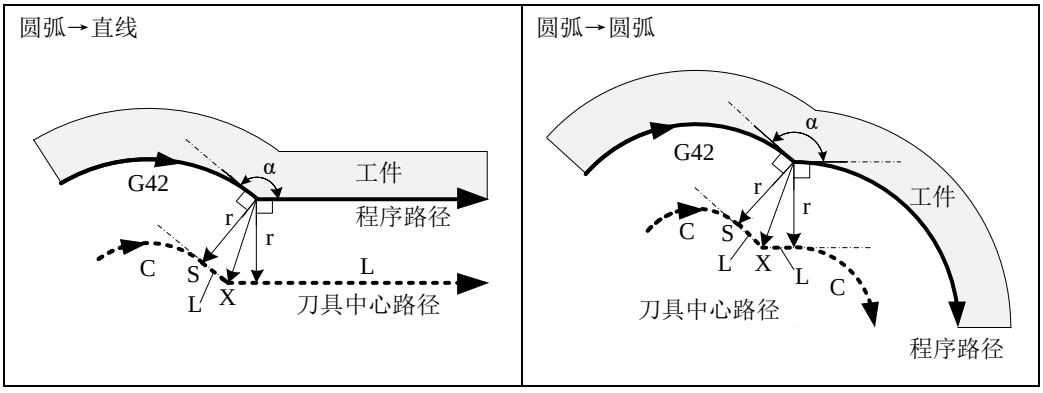
沿着拐角小于1°的内侧移动（α > 359°）

这里只说明直线→直线的情况，其他情况依此类推。

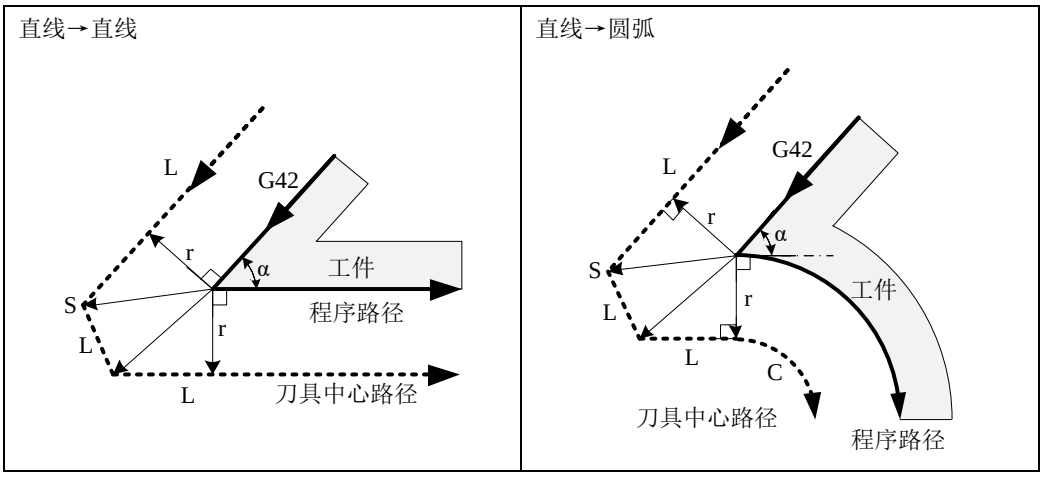


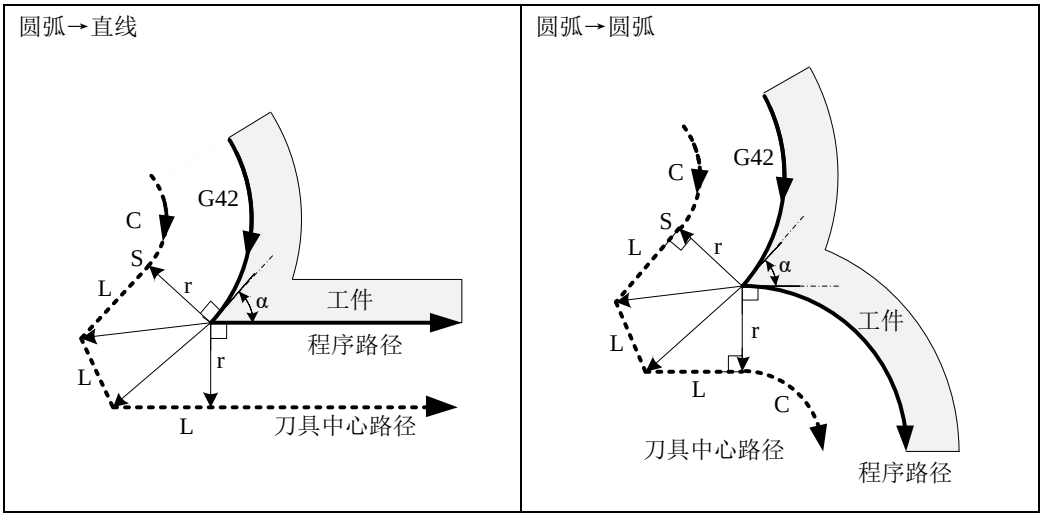
沿着拐角为钝角的外侧移动（90° ≤ α < 180°）





沿着拐角为锐角的外侧移动（α < 90°）

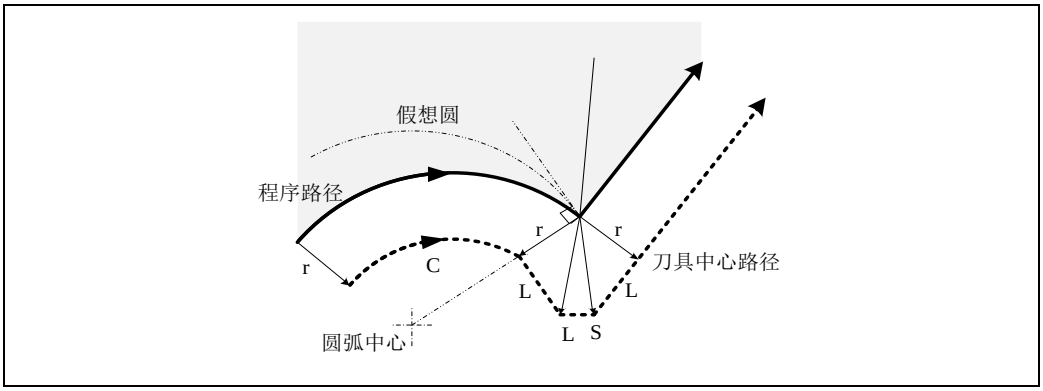




特殊情况

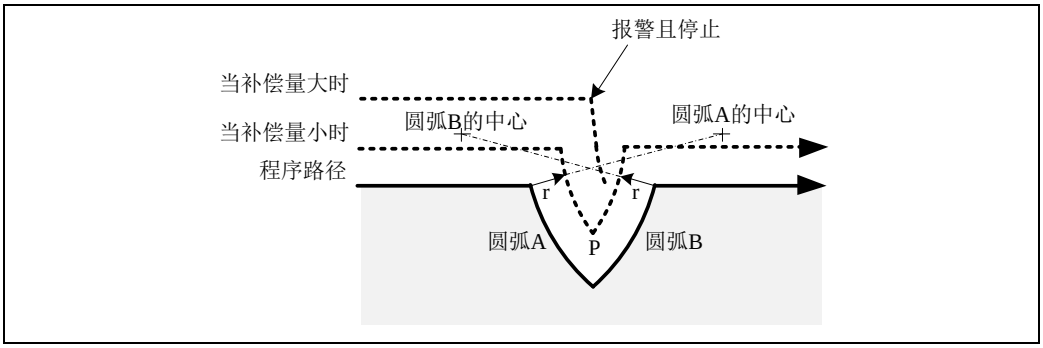
- **编程圆弧终点不在圆弧上**

如下图，当编程圆弧终点不在圆弧上时，系统则假设刀具半径补偿相对于一个假想圆进行，该假想圆通过圆弧终点。根据这种假设，系统创建一个矢量，并进行补偿。“圆弧→圆弧”情况也相同。



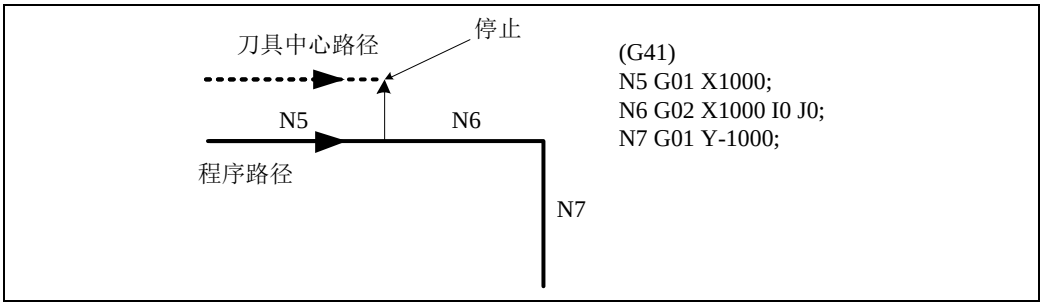
- **没有交叉点**

当刀具半径值较小时，圆弧的补偿路径有交点，但是当刀具半径变大，可能交点不存在。如下图所示，遇到这种情况，刀具将停止在前一程序段的终点并显示报警（PS0033）。



- **圆弧的中心与起点或终点一致**

如果圆心和起点或终点相同，则会产生报警 PS0038，并停止在前一程序段终点。如下图所示。

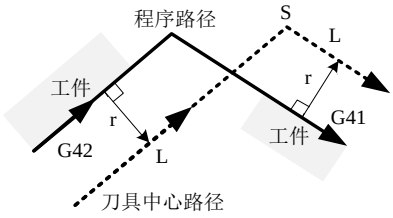


补偿模式下改变补偿方向

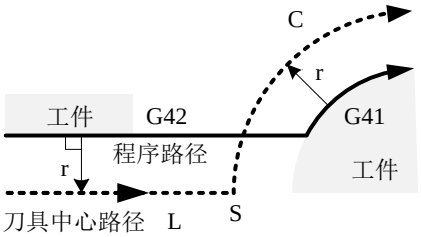
在补偿模式中可变更补偿方向。如果补偿方向在一个程序段中改变，则将该程序段的刀具中心路径和上一程序段的刀具中心路径的交点，作为上一程序段的终点和该程序段的起点。但不可在起动开始程序段及其后面的程序段变更。补偿方向变更时，没有内侧和外侧的概念。下文的解释中补偿量假设为正。

- **具有交点的补偿路径**

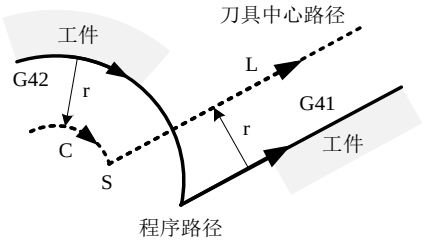
直线→直线



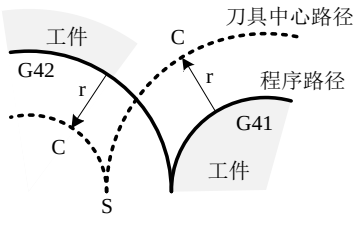
直线→圆弧



圆弧→直线



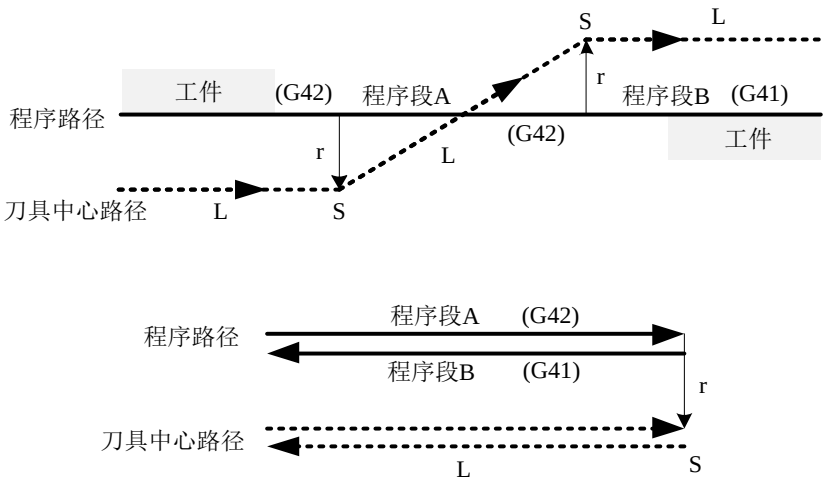
圆弧→圆弧



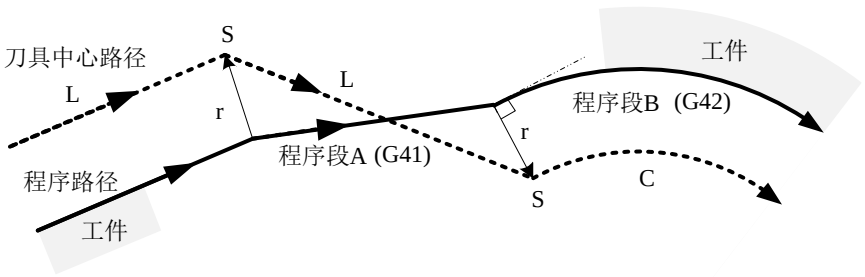
- **没有交点的补偿路径**

当用 G41，G42 改变程序段 A 至程序段 B 的补偿方向时，如果没有交点，则创建一个与程序段 B 的起点垂直的向量，作为程序段 A 的终点和程序段 B 的起点。

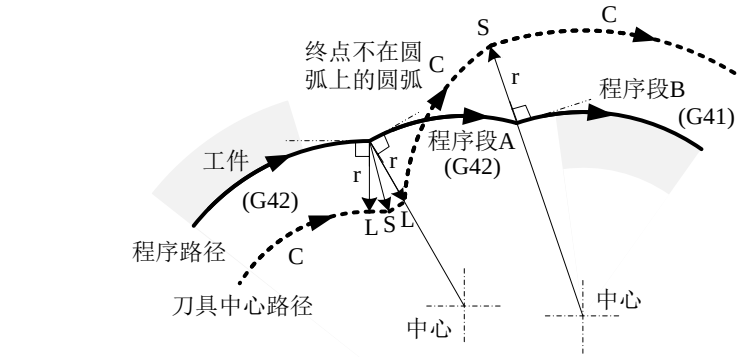
直线→直线



直线→圆弧



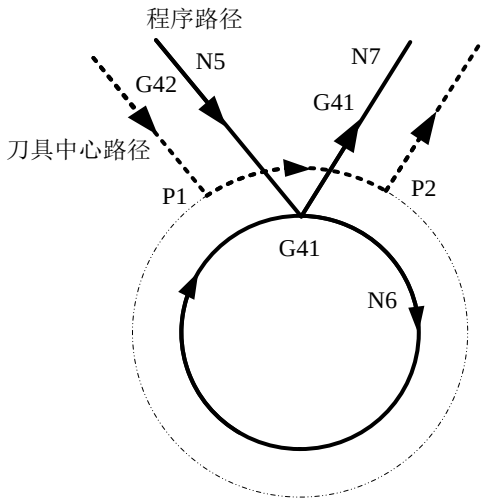
圆弧→圆弧



- **大于 1 周的圆弧**

通常不会产生这种情况。但当 G41 及 G42 变更时，或用 I，J，K 指令 G40 时，则可能会出现。如下图，刀具中心路径不是圆弧，而是 P1 至 P2 的一段弧。并且在某些情况下，可能会因为干涉检查出现报警。

如希望刀具按全圆移动，必须分段指令。

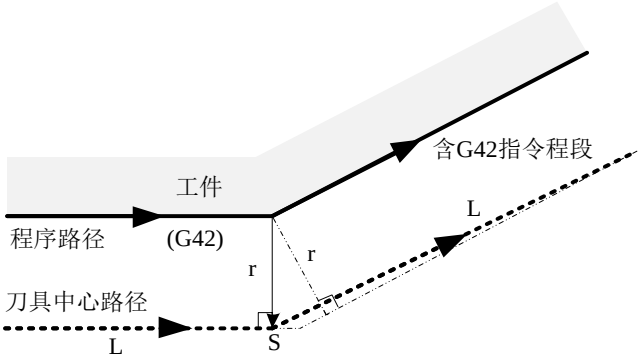


(G42)
N5 G01 G91 X5000 Y-7000;
N6 G41 G02 J-5000;
N7 G42 G01 X5000 Y7000;

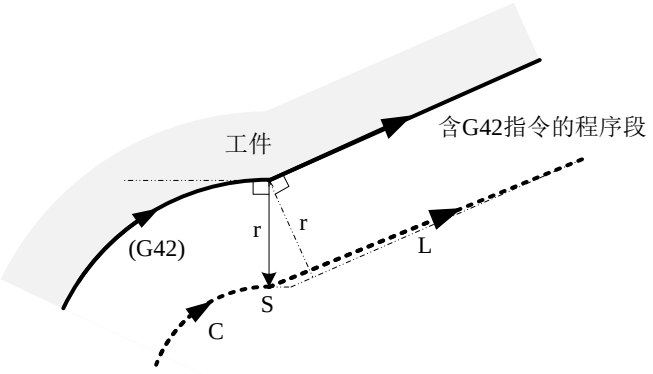
补偿模式下重复指定刀具半径补偿 G 代码

在补偿模式中，重复指定刀具半径补偿 G 代码时（G41/G42 模式时指定 G41/G42），相对于移动方向会形成一个与前程序段成直角的向量，与加工内侧和外侧无关。如果在圆弧指令中指定此种 G 代码，则不能得到正确的圆弧。

直线→直线



直线→圆弧

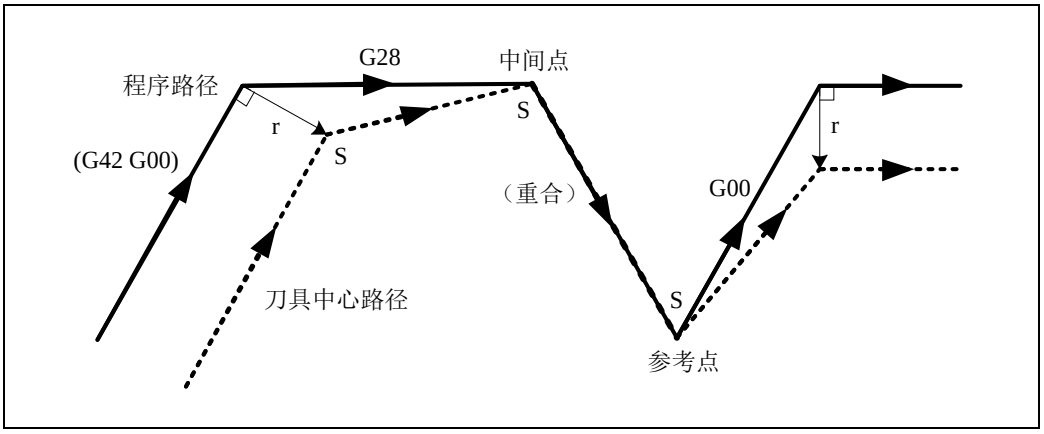


补偿模式下暂时取消补偿的指令

在补偿模式中，如果指定以下的指令，补偿会暂时取消，之后系统会将补偿模式自动恢复。请参见“补偿取消时的刀具移动”及“补偿开始时的刀具移动”的说明。

- **G28 自动返回参考点**

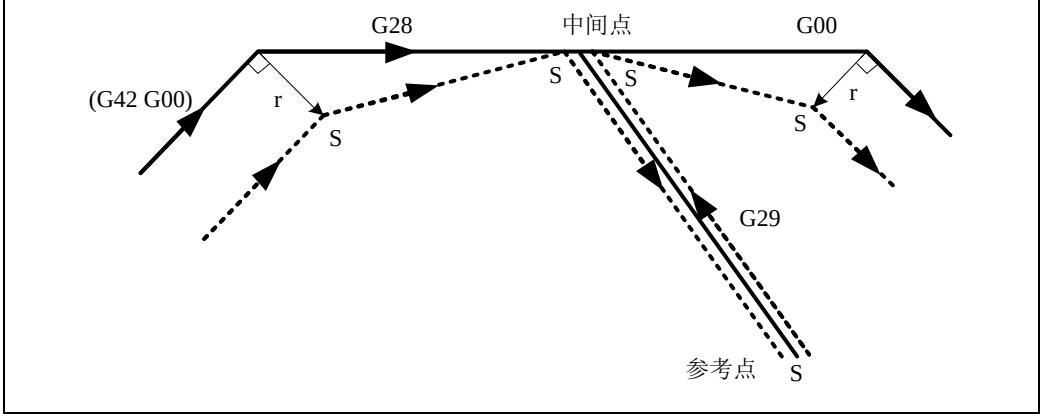
在补偿模式中，如果指令 G28，补偿将在中间点取消，在参考点返回后补偿模式自动恢复



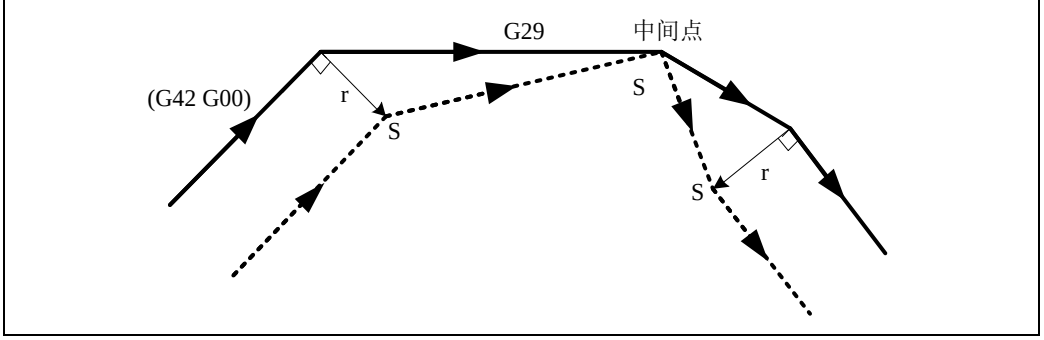
- **G29 从参考原点自动返回**

在补偿模式中，如指令 G29，补偿将在中间点取消，补偿模式将在下一个程序段自动恢复。

在 G28 后立刻指令 G29 时



不在 G28 后立刻指令 G29 时

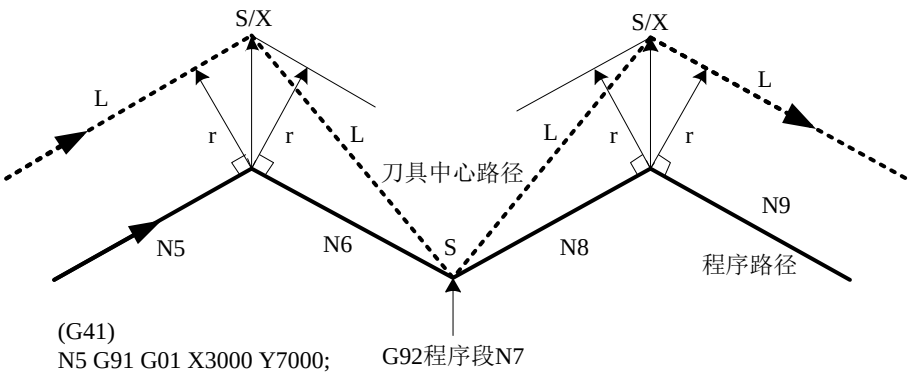


- **G92 工件坐标系设定**

在补偿模式中，如果指定了 G92（工件坐标系设定），补偿向量会暂时取消，之后补偿向量会自动恢复。

此时，不同于补偿取消模式，刀具直接从交点移动到补偿向量取消的指令点。在补偿模式恢复时，

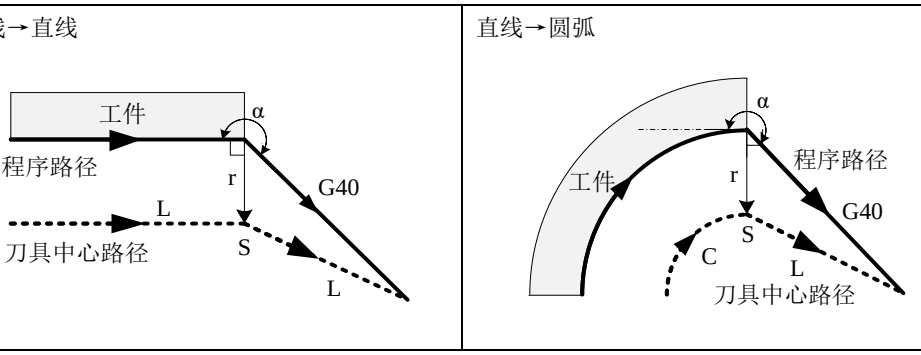
刀具又直接移动到交点。



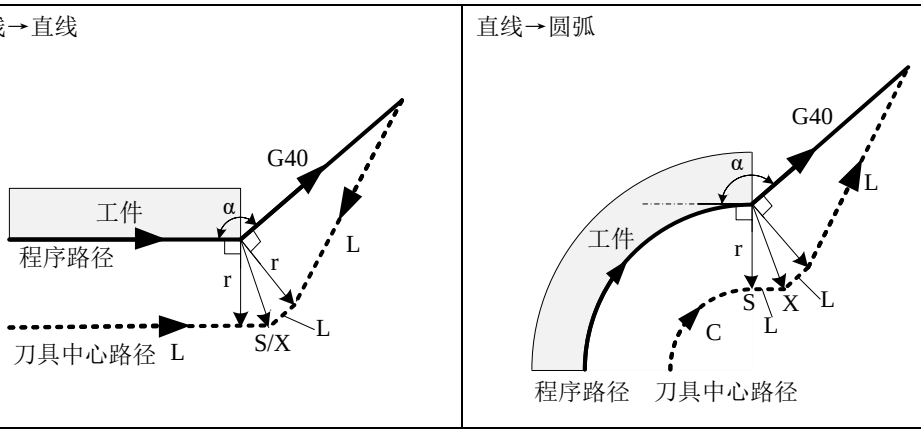
11.1.1.3 补偿取消时的刀具移动

以下说明从补偿状态退出到补偿取消状态时的刀具移动方式。

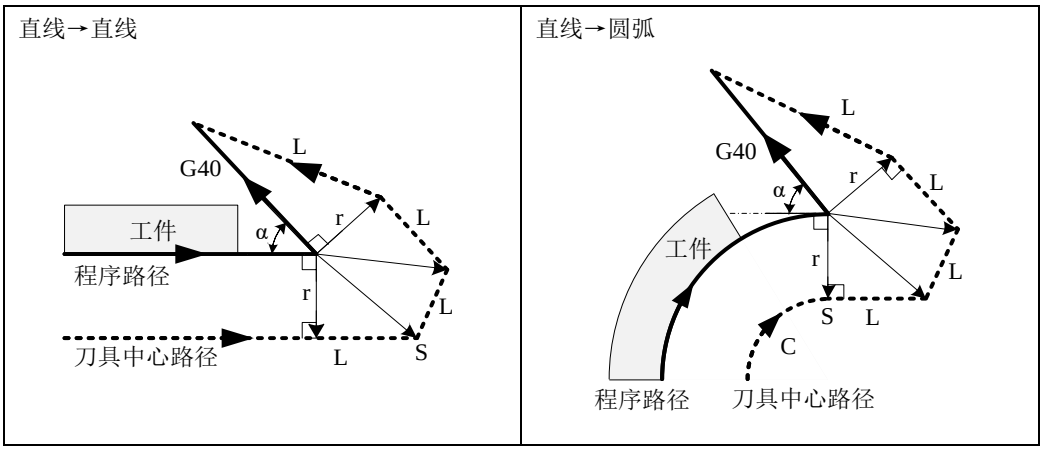
沿拐角的内侧移动（$\alpha \geq 180°$）



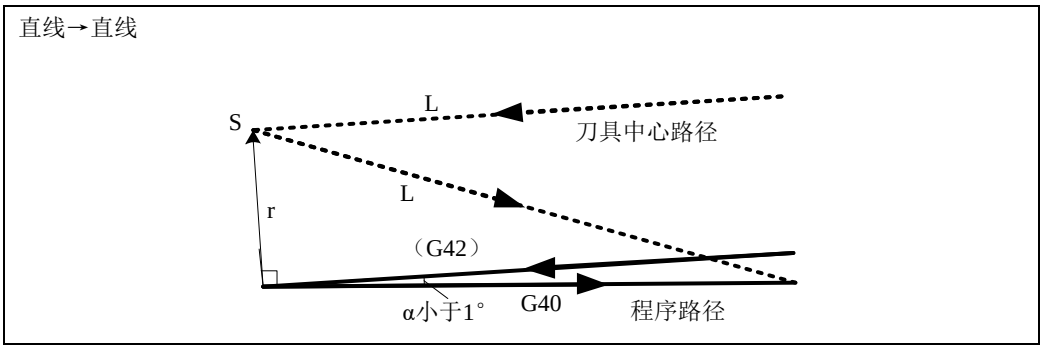
沿着拐角为钝角的外侧移动（$90° \leq \alpha < 180°$）



沿着拐角为锐角的外侧移动（α＜90°）



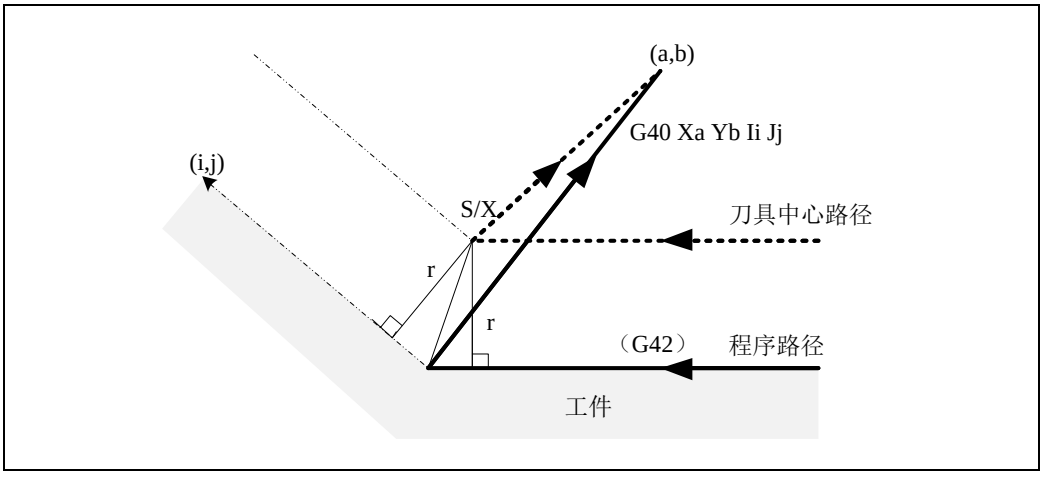
沿着拐角为小于1°的锐角外侧移动（α＜1°）



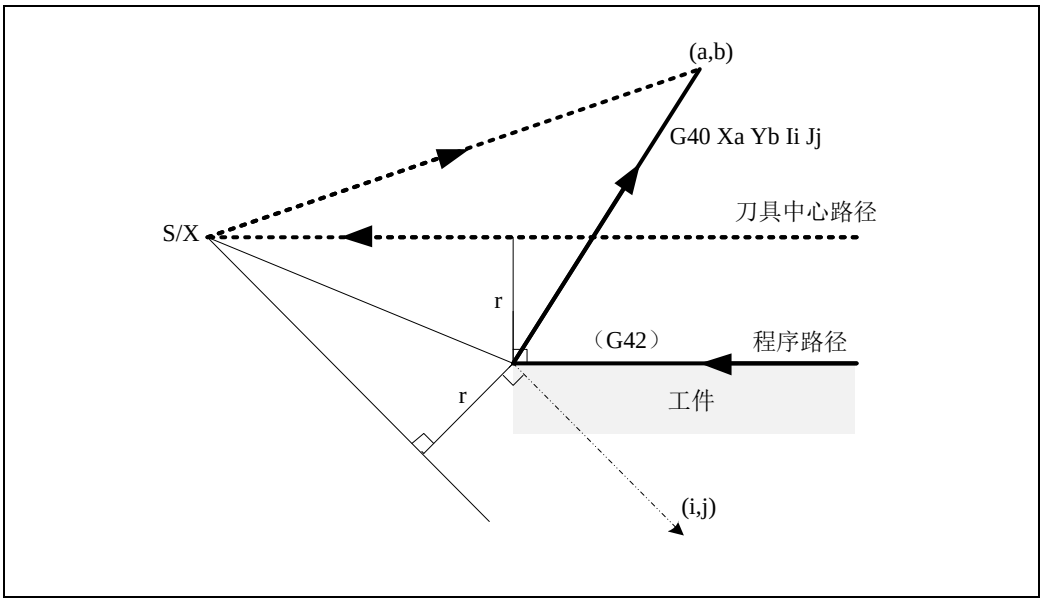
G40 和 I_ J_ K_同时指定

如果在指定 G40 和 I_ J_ K_的程序段之前为 G41 或 G42 模式，则系统假设已经指定从前面程序段的终点到向量(I, J)、(I, K)或(J, K)的方向，补偿方向（G41、G42）与前面程序段相同。

- **指定加工内侧**

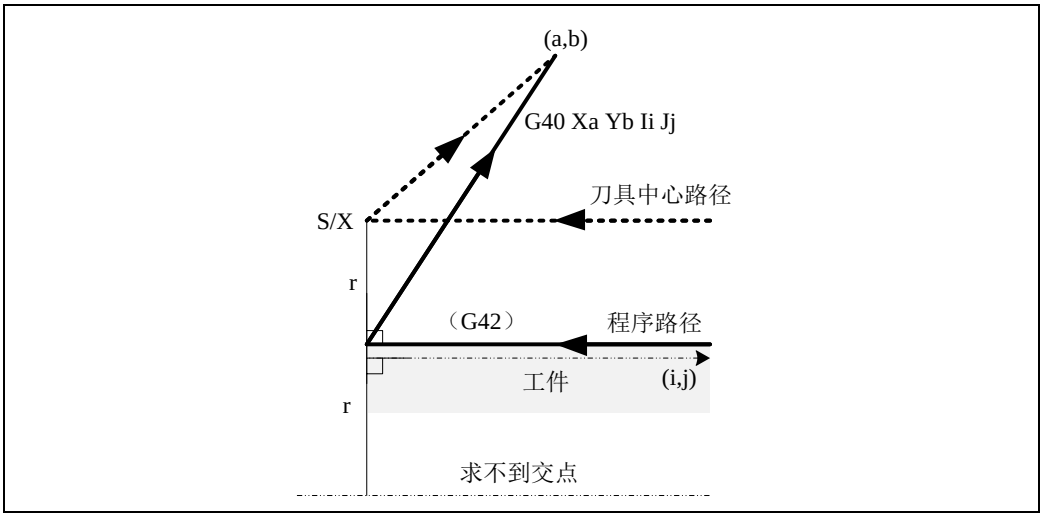


- **指定加工外侧**



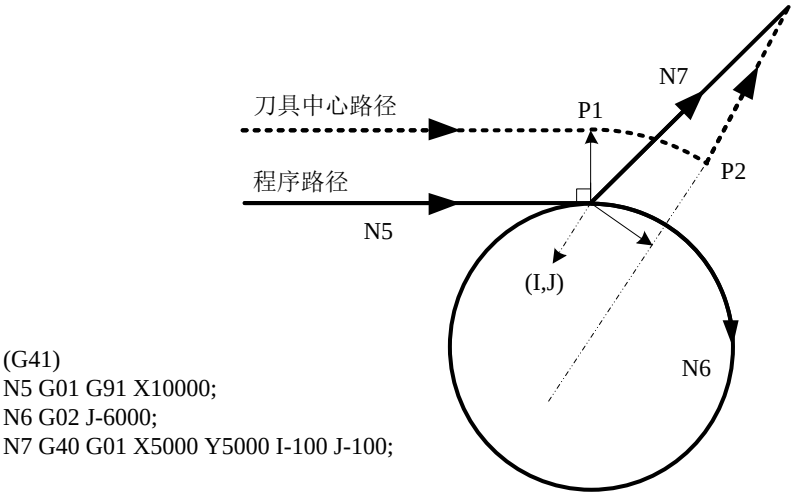
- **没有交点**

当求不到交点时，系统产生一个起点位于 G40 前面的程序段的终点，大小等于补偿量的一个偏移向量。刀具移动到该偏移向量处。



- **大于一周的圆弧**

在下图中，刀具中心路径不沿圆周移动，而沿圆弧从 P1 至 P2。在某些情况下，可能会因干涉检查产生报警（PS0041）。如想沿圆周移动，圆弧指令必须分割。



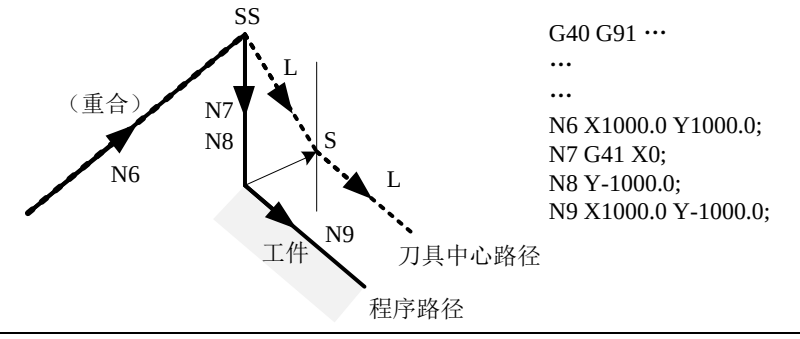
11.1.1.4 不产生刀具移动的程序段

下表中各类型的程序段都是不会产生刀具移动的程序段。

类型	举例	解释
1	M05 ;	M 代码输出
2	S01 ;	S 代码输出
3	G04 X10.000 ;	暂停指令
4	G10 P01 X100 ;	补偿量设定
5	(G17) Z100 ;	补偿平面内无移动的指令
6	G90 ;	只有 G 代码
7	G01 G91 X0 ;	移动量为零

程序段位于补偿开始时

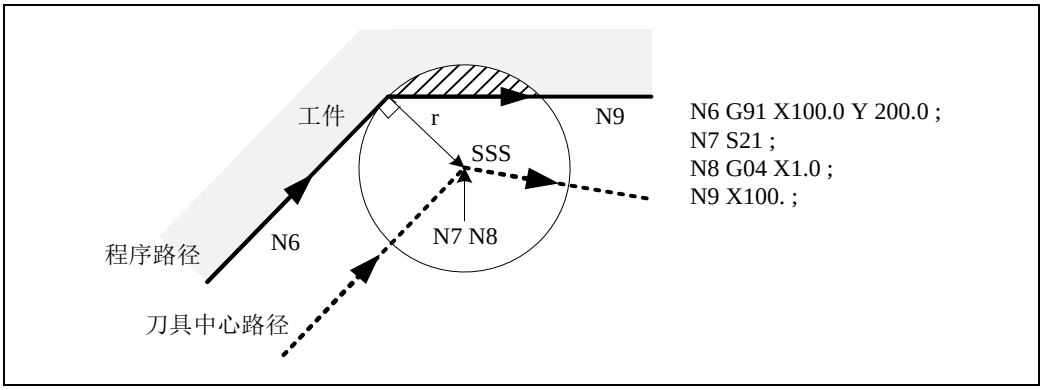
如果在补偿开始的指令没有刀具移动，不会产生补偿向量。



程序段位于补偿模式时

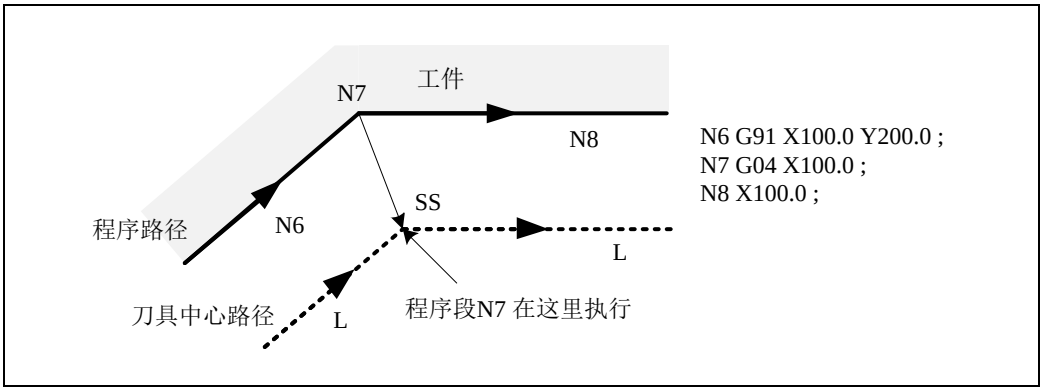
- **连续指令两个及两个以上无刀具移动的程序段时**

  两个及以上没有刀具移动的程序段不可连续指令。如果这样指令，会形成长度为补偿量，方向垂直于前程序段移动方向的向量，可能产生过切。

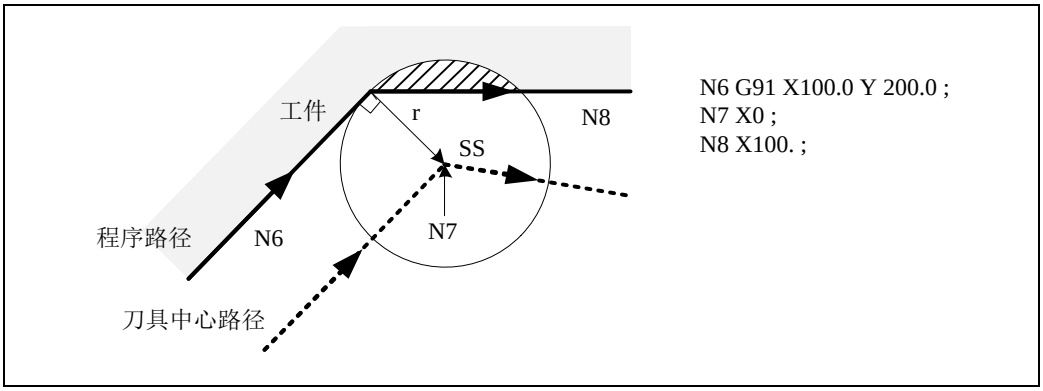


- **仅指令了一个无刀具移动的程序段时**

此时，向量及刀具中心路径与没有指令该程序段时一样。该无刀具移动程序段在单程序段停止点执行。

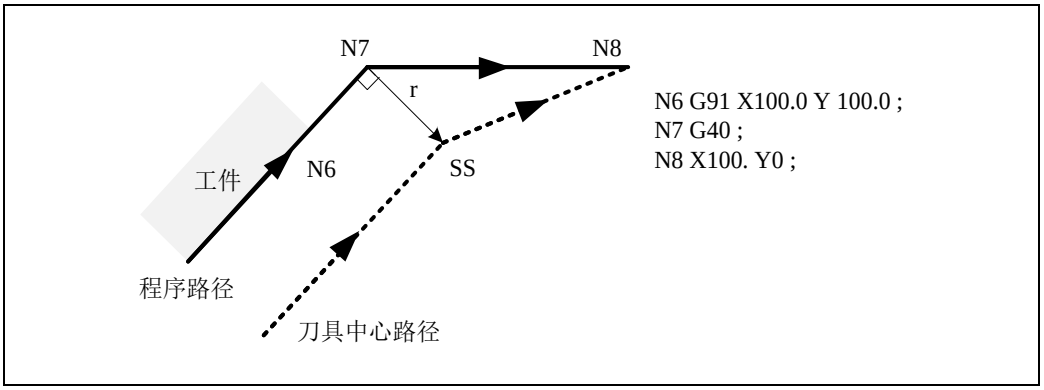


但是，对于移动量为零的程序段（类型 7，参见上表），即使只指定一个程序段，刀具移动与连续指定两个及两个以上没有刀具移动指令的程序段时的情形一样。



程序段位于补偿取消时

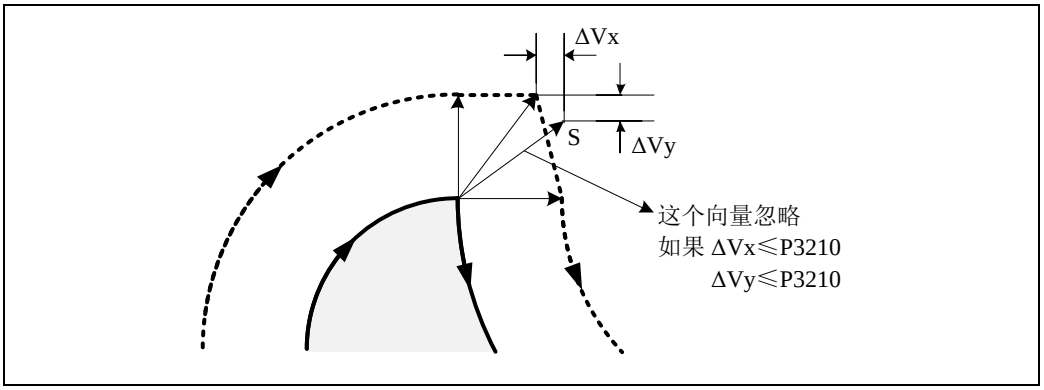
当没有刀具移动的程序段与补偿取消一起指令时，会形成长度为补偿量，方向垂直于前程序段移动方向的向量，这个向量在下一个移动指令取消。



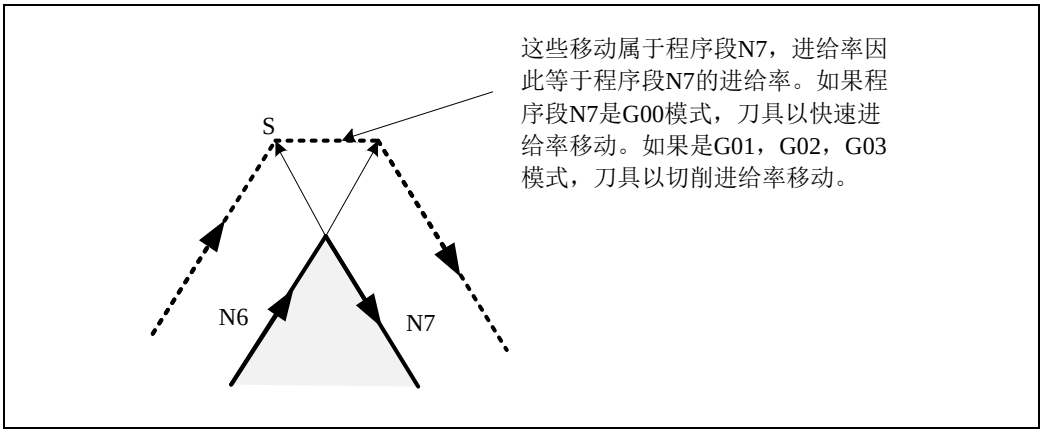
11.1.1.5 转角移动

在一个程序段的终点创建两个或更多个补偿向量时，刀具则线性地从一个向量移动到另一个向量。这种移动称为转角移动。

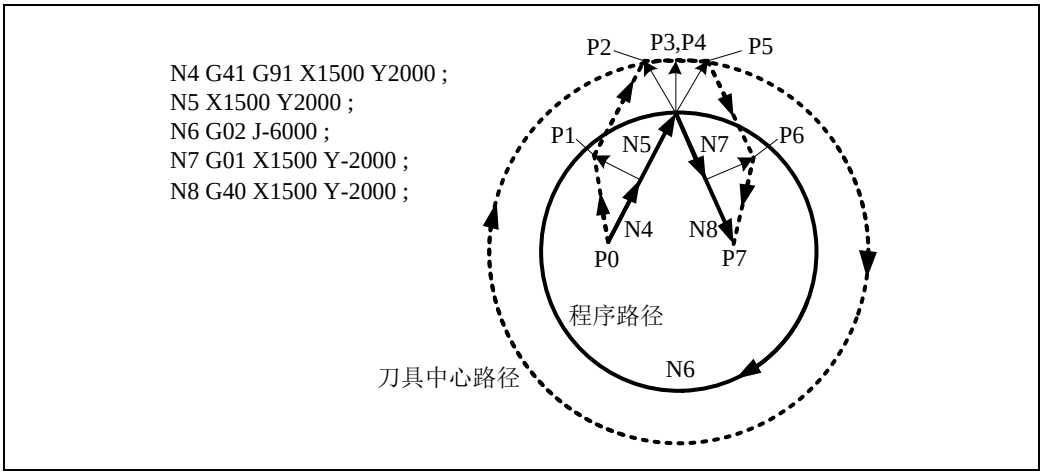
当向量几乎一致时，即向量间的转角移动的距离被判定为较短时（根据参数 P3210），不执行转角移动。此时后面的向量被忽略。通过忽略因为刀具半径补偿而产生的微小移动，从而防止由于缓冲的中断而引起的速度波动。



如果这些向量不一致（未被忽略），则产生一个沿转角的移动。这个移动属于后面的程序段。



但是，如果下一个程序段的路径超过半圆时，则不执行以上机能，即始终不忽略向量。主要是为了避免如下情况。



如果向量未忽略，则刀具中心路径为 P0→P1→P2→P3→整圆→P4→P5→P6→P7；但是，如果 P2 及 P3 间的向量可忽略，则 P3 被忽略。刀具路径为 P0→P1→P2→P4→P5→P6→P7，程序段 N6 的圆弧切削整个被忽略。

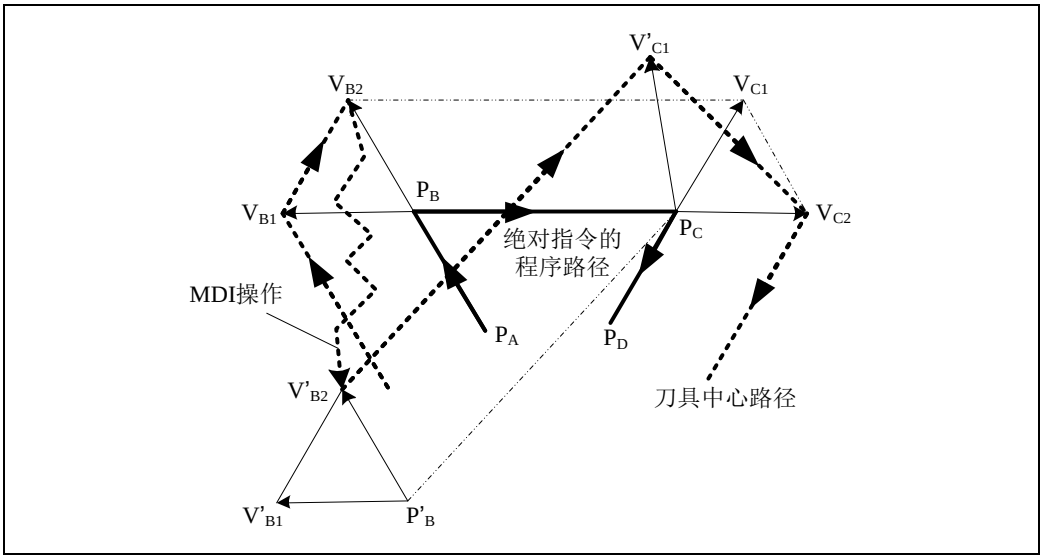
11.1.1.6 MDI 方式下刀具半径补偿

MDI 运行

复位后，在 MDI 方式下执行程序，和自动运行相同，也进行刀具半径补偿。

MDI 干预

当绝对指令编程的 NC 程序在自动运行过程中执行了单程序段停止，插入执行 MDI 操作，然后再次起动自动运行时，此时，系统传送下一个程序段起点的向量，并根据下两个程序段形成其它向量。所以，如下图所示，从点 PC 后补偿可正确地执行。

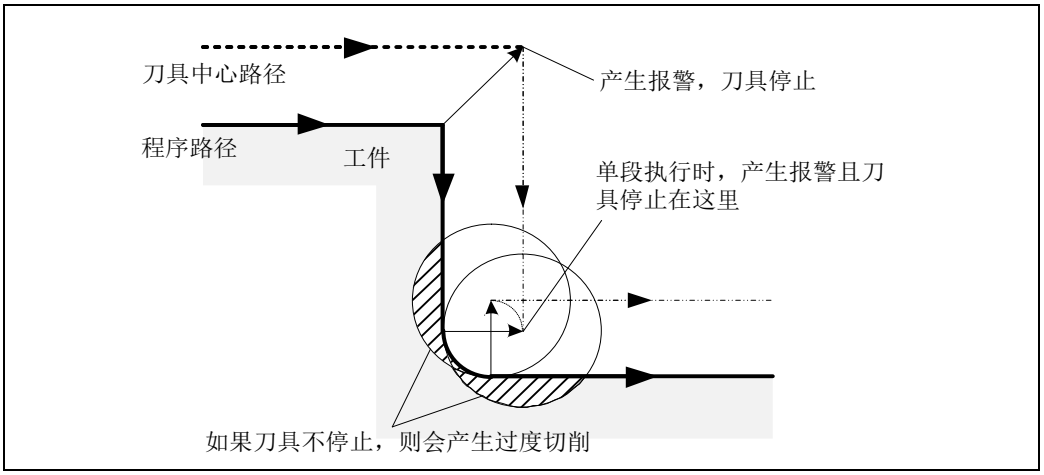


当点P_A，P_B，P_C以绝对指令编程时，程序段从P_A至P_B执行后用单段执行停止，插入 MDI 方式移动刀具。向量V_{B1}及V_{B2}传送至V'_{B1}及V'_{B2}，在程序段$P_B \rightarrow P_C$及$P_C \rightarrow P_D$间的向量V_{C1}及V_{C2}重新计算。但是，因为向量V_{B2}没有再度计算，所以从点P_C后补偿才可正确地执行。

11.1.1.7 利用刀具半径补偿防止过切

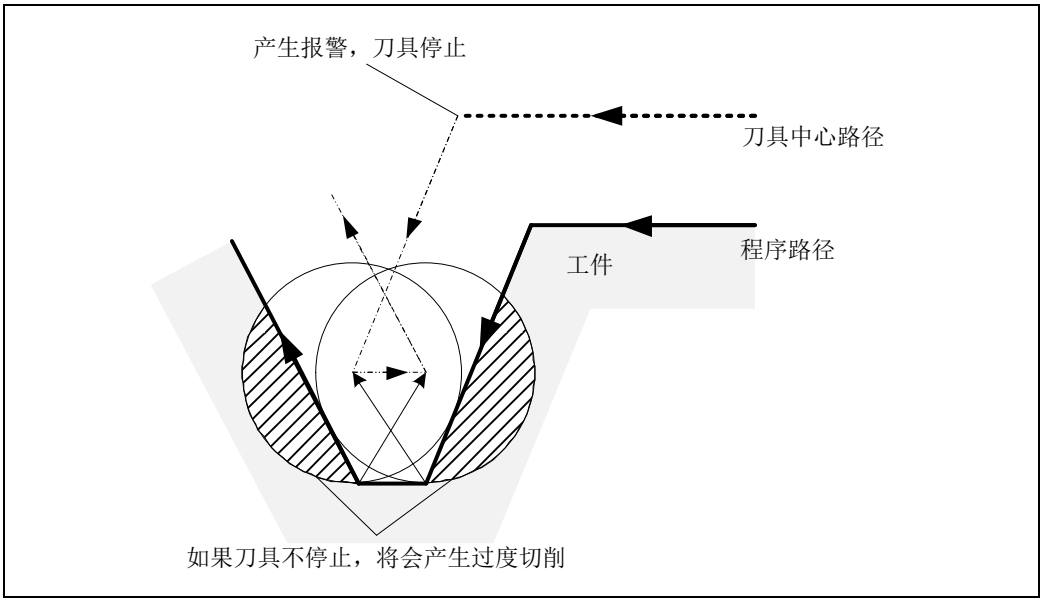
加工比刀具半径小的圆弧内侧

当转角半径小于刀具半径时，因为刀具的内侧补偿将产生过度切削，会产生报警，刀具停在单段程序开始位置，因此在单段执行时，会产生过度切削。



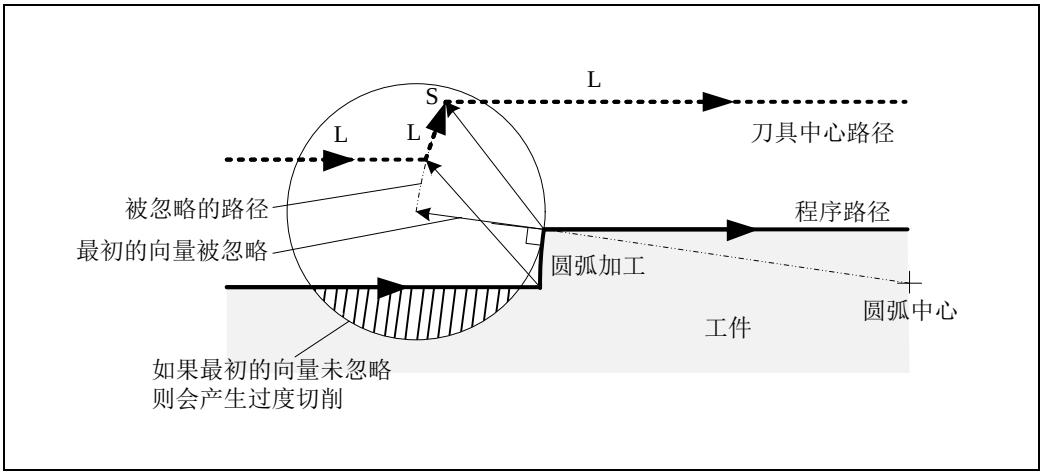
加工刀具半径小的截沟

因为刀具半径补偿强制刀具中心路径向程序路径反向移动，会产生过度切削。



加工刀具半径小的段差

如果在程序中有比刀具半径小的段差时，用圆弧加工指令这个段差的加工时，正常补偿的刀具中心路径变成与程序方向相反。此时忽略最初的向量，刀具直线移动到第二个向量。单段执行在这里停止。如不在单段模式下加工，自动运行会继续。如果段差是直线，不会产生报警，作正确切削。但会残留未切削部分。

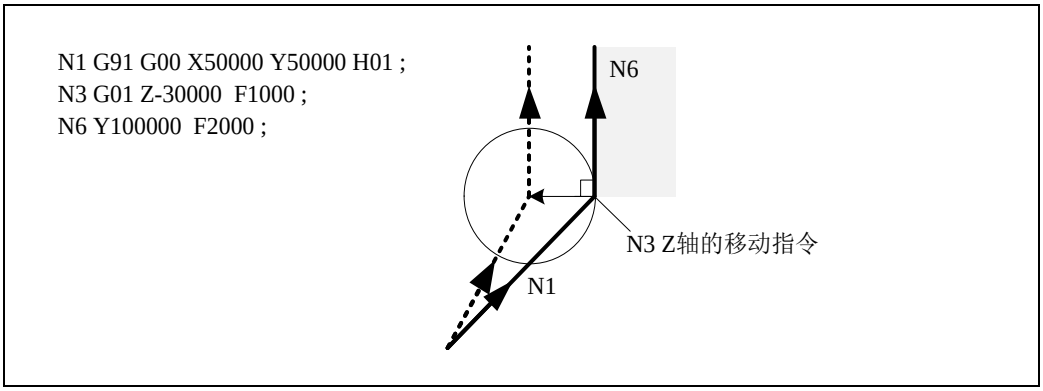


刀具半径补偿开始时分段 Z 轴移动

一般是在加工开始时，刀具半径补偿有效后（补偿平面为 XY），刀具会沿 Z 轴移动到距工件一段距离。此时，如想将沿 Z 轴的移动分为快速进给及切削进给两段，应参照以下例子。

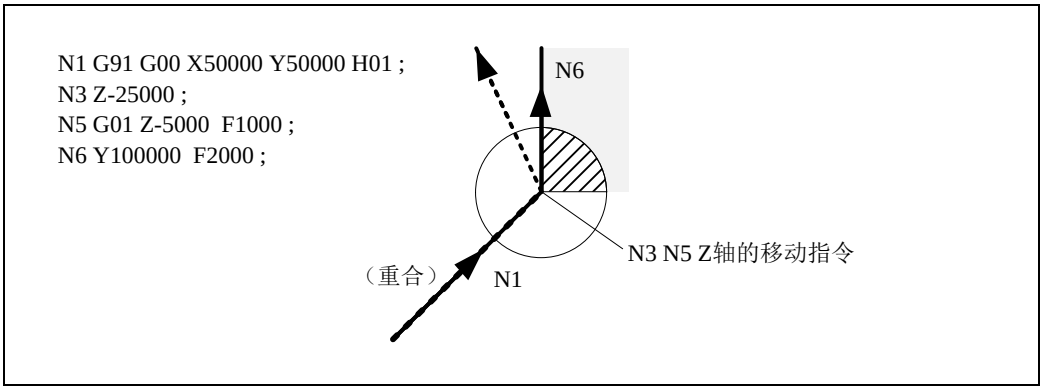
- **原始程序**

如下图，执行 N3 时，N6 也进入缓冲区，系统能够正确的执行补偿。



- **错误修改后的程序**

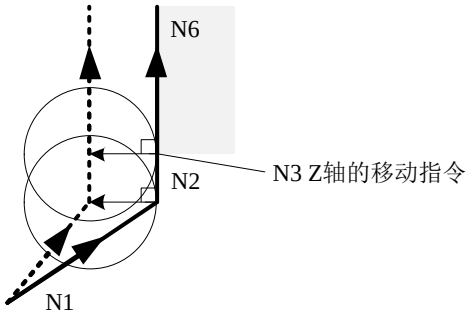
如下图，刀具在刀补平面内连续两个程序段没有移动指令，N6 不能进入刀补缓冲区，补偿向量不在补偿开始时计算，结果产生过度切削。



- **正确修改后的程序**

如下图所示，虽然 N3 和 N5 也连续指定了两个在刀补平面内无移动的程序段，但是根据刀补规则，系统将在无移动指令处，产生方向垂直于前程序段移动方向的向量，由于 N2 和 N6 方向相同，所以能够正确补偿，不会产生过切。

N1 G91 G00 X50000 Y40000 H01 ;
N2 Y10000 ;
N3 Z-25000 ;
N5 G01 Z-5000 F1000 ;
N6 Y100000 F2000 ;



11.1.1.8 干涉检查

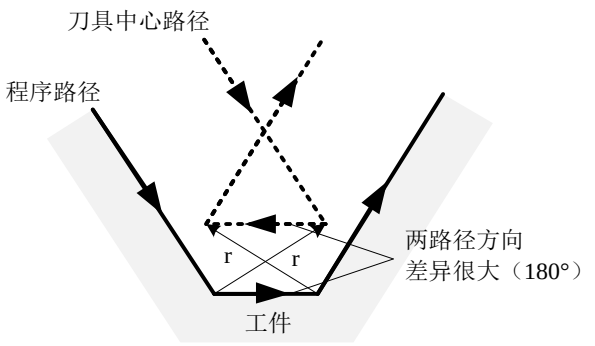
刀具过切现象称为“干涉”。干涉检查功能可以预测出可能的刀具过切情况，但是不能检查出所有的干涉情形，即使过切未发生也可能会进行干涉检查，并报警。

干涉的基本条件

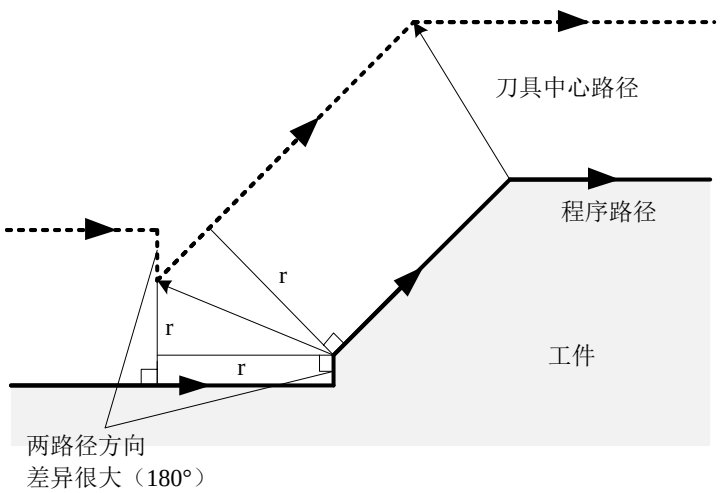
1 刀具路径方向与程序路径方向不同（路径间的夹角在 90 度~270 度之间）。

2 圆弧加工时，除以上条件外，刀具中心路径的起点和终点间的夹角与程序路径起点和终点间的夹角有很大的差异（180 度以上）。

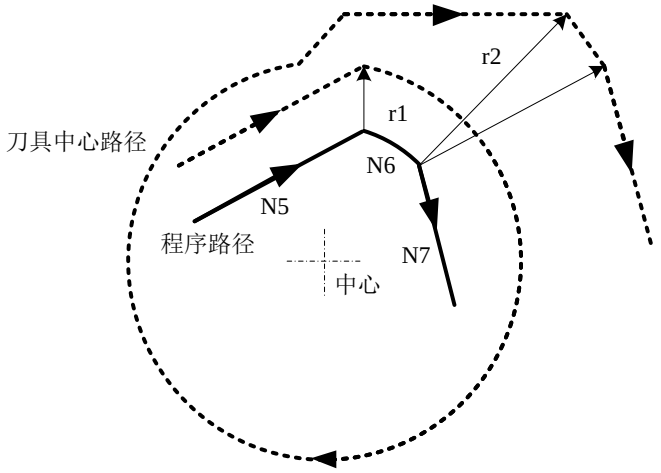
例 1



例 2



例 3



(G41)
N5 G01 G91 X8000 Y2000 H01 ;
N6 G02 Y-1600 X3200 I-2000 J-8000 H02 ;
N7 G01 X2000 Y-5000 ;

（H01刀具半径补偿量 r1 =2000)
（H02刀具半径补偿量 r2 =6000)

以上范例，程序段 N6 的圆弧在一个象限内。但是在刀具补偿后，圆弧位于 4 个象限。

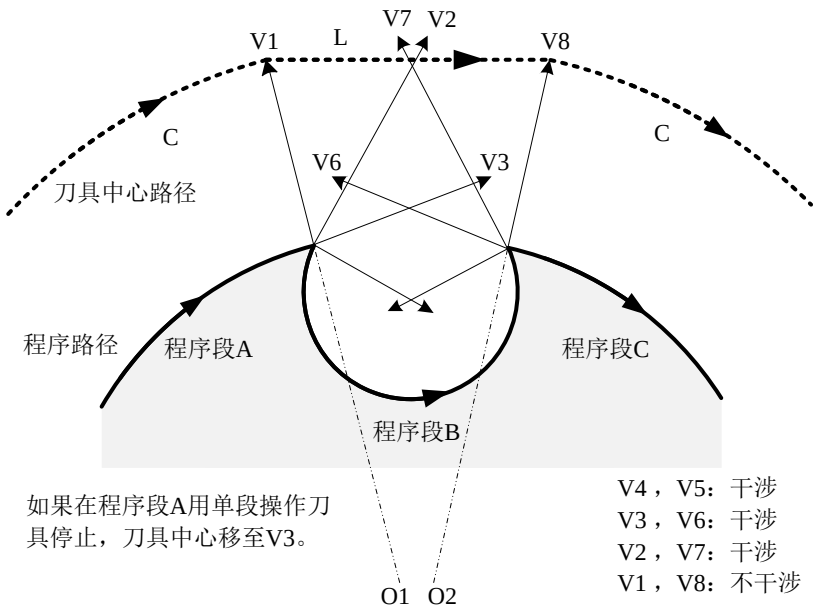
干涉的预先处理

补偿模式中，相邻的程序段 A、B 及 C 执行时，在 A 及 B 间产生向量 V1、V2、V3 及 V4，在 B 及 C 间产生向量 V5、V6、V7 及 V8（参见下文例子中的图），首先检查最近的向量。如果发生干涉，则将它们消去。但是如果要忽略的向量在拐角的最后，它们不能消去。即：

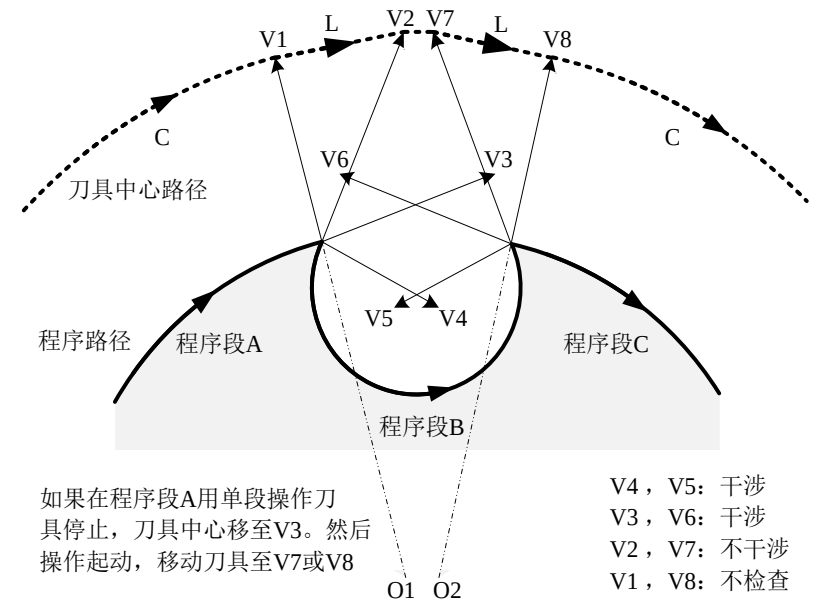
V4 和 V5 间 ----- 干涉 ----- V4，V5 消去
V3 和 V6 间 ----- 干涉 ----- V3，V6 消去
V2 和 V7 间 ----- 干涉 ----- V2，V7 消去
V1 和 V8 间 ----- 干涉 ----- V1，V8 不能消去

如果在检查中，某一对向量不干涉，则不检查此后的向量。如果程序段 B 是圆弧移动，向量干涉会产生直线移动。

例 1 刀具直线移动（V1→V8）

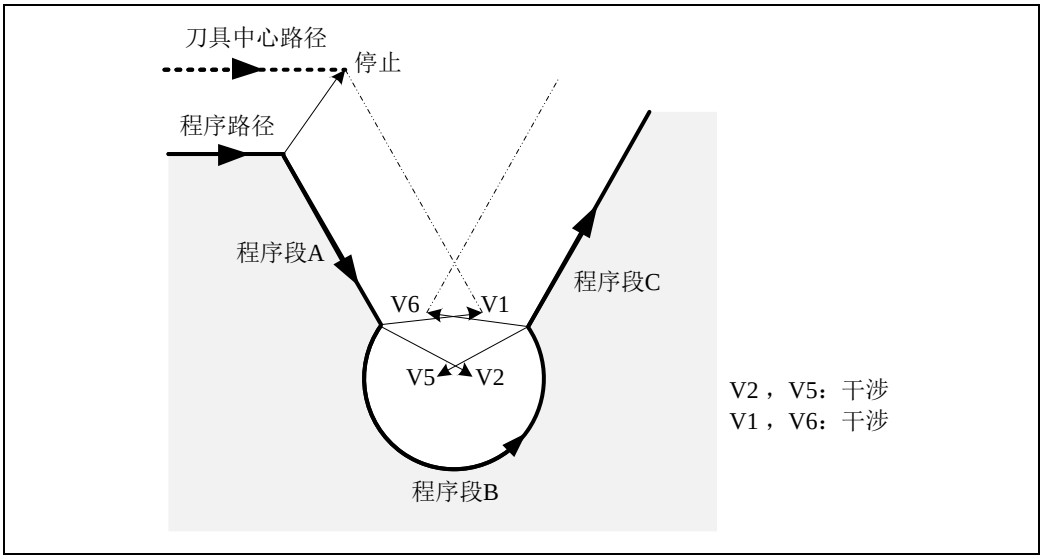


例 2 刀具直线移动（V1→V2→V7→V8）



预处理后仍有干涉发生

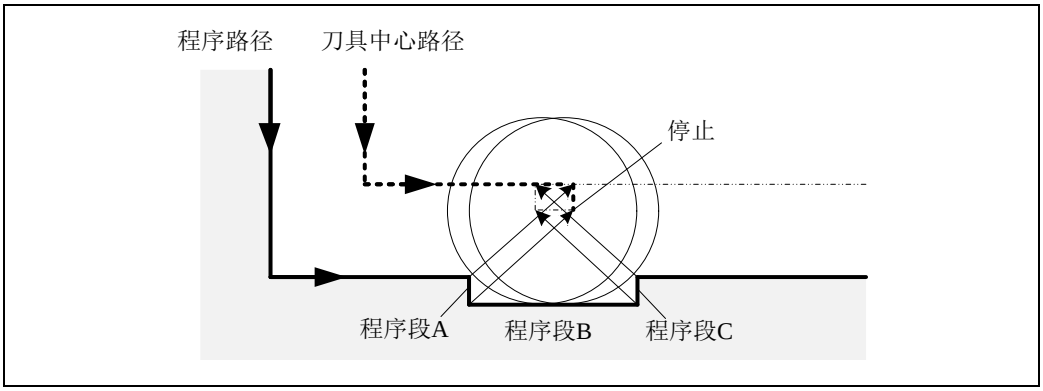
如果干涉在预处理后发生或检查开始只有一组向量而这组向量干涉，刀具在前面程序段执行后立即停止（如果用单程序段操作执行，刀具在程序段结束时停止），显示报警（PS0041）。



实际上没有干涉，但作干涉检查

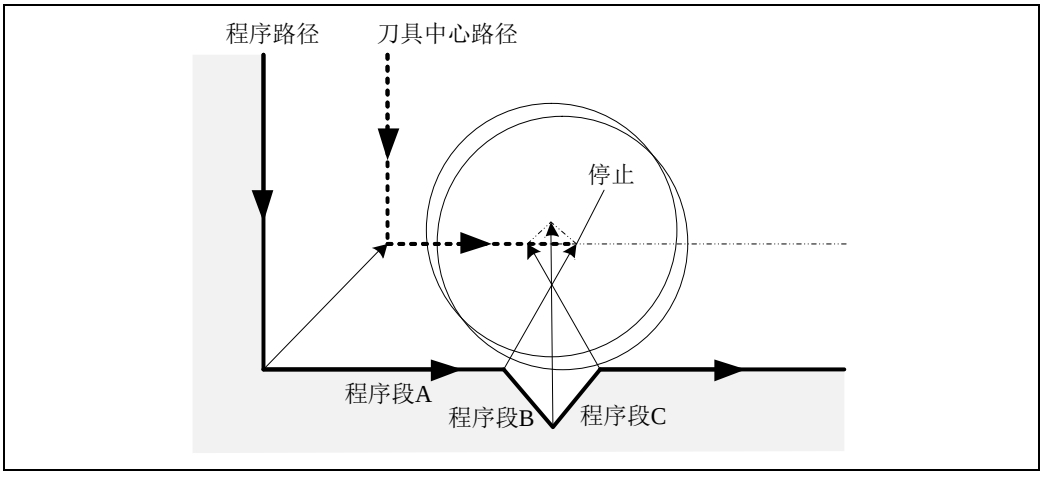
- **凹处深度小于补偿量**

实际上没有干涉，但是因为在程序 B 段，程序的方向与刀具半径补偿的路径相反，刀具停止并显示报警。



- **截沟深度小于补偿量**

刀具路径与程序路径方向相反。



12 用户宏程序（宏 B）

用户宏程序（如未特别说明简称宏 B）允许用户使用变量、算术运算、逻辑运算、位操作、条件转移、循环控制和程序调用等基本程序语言特征，使得编制程序更方便、灵活、容易、快捷。可以极大的提高程序的通用性，只需要对不同的主程序赋值，即可调用相同的子程序加工同类型的零件（如模具行腔、阵列孔等）。另外，宏程序中函数的运用也使各种曲线加工更加流畅，同时程序占用系统空间更小。

本章节将全面介绍标准宏 B 的功能，简易宏 B 功能仅在标准宏 B 功能基础上简化了个别指令和操作，本节将在被简化的内容处特别注明。

12.1 变量

普通加工程序直接用数值指定 G 代码和移动距离（例如，G01 X100.0）。使用用户宏程序时，数值可以直接指定或用变量指定。当用变量时，变量值可用程序赋值或用 MDI 面板操作设定。

例

```
#1=#2+100 ;
G01 X#1 F300.0 ;
```

解释

- **变量的表示**

一般程序语言需要对变量命名，但用户宏程序不需要，用户可以直接使用变量，即用变量符号（#）和后面的变量号指定。

例

#1、#2、#101 等均表示变量，即变量由“#+数值”组成。

表达式也可用于指定变量号。此时，表达式必须封闭在括号中。

例

当 COMP=1 时，#[#1+#2-12]

当 COMP=0 时，#(#1+#2-12)

注

当参数“COMP”（P4100.5）设为 1 时，方括号用于变量索引、改变优先级、函数或命令等语句中，而圆括号用于注释；当 COMP 设为 0 时，圆括号代替方括号功能，而方括号则用于数组功能，注释使用 C/C++语言注释风格（//或/* */）。如无特别说明，默认 COMP 为 1。

- **变量的类型**

所有变量根据变量号可以分成四种类型，如下所示。

变量号	别名	变量类型	功能
#0~#63	r0~r63	局部变量	局部变量只能在宏程序内局部使用，例如，运算结果。断电后开机，局部变量初始化为空。调用宏程序时，自变量对局部变量赋值。
#100~#499 #500~#999	——	公用变量	公用变量在不同的宏程序中意义相同。断电后开机，变量#100~#499 初始化为空，变量#500~#999

变量号	别名	变量类型	功能
			的数据与断电前保持一致，即使断电也不丢失。
#1000~	——	系统变量	系统变量用于读写系统内部的各种数据，例如，刀具的当前位置和补偿值等。

- **变量的显示**

对于局部变量和公用变量（#0~#999），系统在“宏变量”界面予以显示，但不显示系统变量。变量号右侧一栏是该变量的数值，可直接用 MDI 面板操作设定。如果变量未曾赋值，则变量值显示为空。

- **变量的引用**

在地址后指定变量号即可引用其变量值。当用表达式指定变量时，要把表达式放在括号中。

例

G01 X#100 ;

G01 X[#1+#2] F#3 ;

被引用变量的值根据地址的最小输入单位自动地舍入。

例

系统公制编程时，最小输入单位默认为 0.001mm，当把 12.3456 赋给#1，指令 G00X#1 时，实际在指令 G00 X12.346。

如果要改变引用变量值的符号，应将负号（-）放在#的前面。

例

G01 X-#100 ;

G01 X-[#1+#2] F#3 ;

引用系统变量

例

#500=#4001	读取当前 01 组 G 代码的模态信息，读取数值为当前 G 代码的变量号
#1 = #500	若当前#1=1，则说明当前 01 组 G 代码的模态为 G01

使用 G/M/S/T/RS 指令调用宏变量地址时，赋值数据必须为 BCD 码。或参数“BADR”（P2502.6）设置使用 G/M/S/T 通过变量赋值时不需要转为 BCD 码。

例

#1=90;

G[BCD[#1]] X10 Z10 ; 使用 A 体系圆柱切削循环 G90 指令

注

引用宏变量地址传递数据时，宏变量地址需被定义。若引用未定义的宏变量地址时，系统会产生报警（PS1318）。

- **变量名称编程功能**

使用宏变量时可根据需要将变量地址赋予别名，编程可使用别名进行数据传递，配合变量提示信息功能使用，可使程序编写条理更加清晰，调用更加方便。

1 使用宏变量自定义名称编程

0006								
			MVAP					

[数据类型] 位型

MVAP

0：无宏变量别名编程机能

1：有宏变量别名编程机能

例：

5 号宏变量的别名为“长度”，那么："#长度"等价于"#5"。

2 在宏程序界面可以自定义变量名称和添加变量提示信息

4101		MVNP						

[数据类型]　位型

MVNP

0：不允许修改宏变量名称和提示

1：允许修改宏变量名称和提示

注：

1 该参数有效时可以修改宏程序名称和提示，P4101.6=0 时修改宏变量名称，发生 P/S 报警 703：由于权限或参数 4101.6 保护，禁止修改宏变量名称与提示

2 名称，提示可以支持英文和中文表述

3 可通过操作面板按键直接更改或通过 U 盘，串口等方式传输宏变量文件

例：开启功能，并定义以下宏变量名称：#500（孔深），#501（索引 1），#502（索引 2）。下面程序可以正常运行：

```
O0010
孔深 = 100.0 ;
G1 X0. ;
X [ 孔深 ] ;
索引 1 = 300 ;
索引 2 = 200 ;
X0. ;
X [ # [ 索引 1 + 索引 2 ] ] ;
M30 ;
```

- **变量的初始化**

使用宏变量时常会用到地址初始化的操作，对于所有宏变量：

用户可将宏变量赋值为 NULL，表示该变量为空或未定义，用于初始化变量。

例

```
1  #1 = NULL ;
2  r1 = NULL ;
3  #1 = 5 ;
4  #[#1] = NULL ;          相当于#5=NULL
5  #2 = NULL ;
6  #1 = #2 ;               #2 为空，赋值将#1 置空
```

为了兼容其他厂家的宏程序设计，本系统支持自动将#0 变量替换为 NULL，由参数“ZVRBN”（P4100.1）控制，ZVRBN 设为 1 时该机能有效。

功能中的替换操作等价于文本替换，即将“#0”替换为“NULL”，因此以下情况将不做处理：

1 #0 被视为空变量，非 0 号变量。r0 仍然作为 0 号变量被访问，不会替换为 NULL。

2 在#[表达式]中，即便“表达式”结果为 0，也表示 0 号变量，不会被替换为 NULL。

3 “#0…0”也被视为#0 情况，被替换为 NULL。例如，“#00”、“#0000”。

4 空变量不允许进行除赋值、作为 CALL 函数参数及 ISNULL 函数参数以外的操作。例如：

```
#1 = NULL == NULL ;      //报警
#1 += NULL ;             //报警
#1 = NULL + 1 ;          //报警
#1 = SIN(NULL) ;         //报警
```

当参数“ZVRBN”（P4100.1）设置为 1 时，允许使用#1 EQ #0 或 #1 NE #0 来判断变量是否为空。

对于公用变量，可以使用上述方法初始化变量，也可以通过设置参数“GVNUL”（P4101.7）在每次开机时将公用变量初始化。

可设置为每次开机将公用变量初始值设为“0”，或设置为开机不更改公用变量的方式。

4101	GVNUL							

[数据类型] 位型

GVNUL

0：#100~#499 号公用变量初始值为 0

1：#100~#499 号公用变量初始值为原方式

- **限制**

程序号和顺序号不能使用宏变量。例如，下面情况不能使用变量：

例

下面的情况不能使用宏变量：

```
O#1 ;
N#3 Y200.0 ;
```

12.2 基本数据类型

用户宏程序有 6 种基本数据类型：整型、无符号整型、长整型、无符号长整型、单精度实型和双精度实型。常量赋值时，系统会自动选择合适的数据类型存储。变量无需定义数据类型，可直接使用。一般情况下，可以赋给变量任何基本数据类型的常量。下表给出了 6 种数据类型的长度和范围。

类型	长度	范围
INT（整型）	4 字节	$-2^{31} \sim 2^{31}-1$
UINT（无符号整型）	4 字节	$0 \sim 2^{32}-1$
LONG（长整型）	8 字节	$-2^{63} \sim 2^{63}-1$
ULONG（无符号长整型）	8 字节	$0 \sim 2^{64}-1$
FLOAT（单精度实型）	4 字节	1.4012985e-045~1.7014118e+038 约精确到 6 位数
DOUBLE（双精度实型）	8 字节	2.2250738585072014e-308~1.7976931348623157e+308 约精确到 15 位数

12.2.1 整型常量

整型、无符号整型、长整型和无符号长整型常数统称为整型常量，简称整数。

解释

- **整型常量的表示**

整型常量具有以下两种表示方法：

1 十进制整数

编程时通常以习惯的十进制整数形式指定。

例

10、-345、0、+5

2 十六进制整数

十六进制整数是以 0x 开头的十六进制字符串。十六进制字符有 0~9，a~f（或 A~F），其中 a~f（或 A~F）表示十进制值 10~15。

例

0x12	（十进制 18）
0xb	（十进制 11）
0xa8	（十进制 168）
0xFFFF	（十进制 65535）

- **存储类型修饰符**

系统设定了 s、u 和 l 三种整型修饰符。s 表示有符号数，u 表示无符号数，l 表示 8 字节长整型或无符号长整型数。无修饰符，默认为 4 字节有符号整型数。这些修饰符有的可以相互组合。

例

5s	（4 字节有符号整型数）
5u	（4 字节无符号整型数）
5l	（8 字节有符号长整型数）
5sl 或 5ls	（8 字节有符号长整型数）
5ul 或 5lu	（8 字节无符号长整型数）

注

如果设定了修饰符，应该保证数值在规定的范围内，系统不进行溢出判断。

- **系统定义整型常量**

系统定义整型常量 true（TRUE）和 false（FALSE）。其中，true（TRUE）表示 32 位无符号整型数 1，false（FALSE）表示 32 位无符号整型数 0。

12.2.2 实型常量

单精度实型和双精度实型常数统称为实型常量，简称实数。

解释

- **实型常量的表示**

实型常量又称浮点常量，是一个十进制表示的符号实数。符号实数的值包括整数部分、尾数部分和指数部分。实型常量的形式如下：

```
[+|-][digits][.digits][e[+|-]digits]
```

此处，digits 是一位或多位十进制数字（0~9），e 是指数符号。小数点之前是整数部分，小数点之后是尾数部分。整数部分和尾数部分可以省略，但不能同时省略。指数部分用 e 开头，基数为 10。实型常量如果不加修饰，则默认为双精度类型。详情参见 I-12.11.3。

- **存储类型修饰符**

系统设定了 f 和 d 用于修饰单精度和双精度实型常量。修饰符不能修饰带指数符号的实型常量。

例

5.2	（无修饰符，默认 8 字节双精度实型）
5.2e2	（带指数符号默认 8 字节双精度实型）
5.2f	（4 字节单精度实型）
5.2d	（8 字节双精度实型）
5.2e2f 或 5.2e2d	（非法）

- **系统定义实型常量**

为方便使用，系统预定义了部分实型常量，如下表所示。

名称	意义
PI	8 字节双精度实型，值为 3.1415926535897932。
INF 或（PINF）	8 字节双精度实型，值为＋∞（存储为 0x7FF0000000000000），供测试用。
NINF	8 字节双精度实型，值为－∞（存储为 0xFFF0000000000000），供测试用。
ZERO（PZERO）	8 字节双精度实型，值为+0（存储为 0x0000000000000000），供测试用。
NZERO	8 字节双精度实型，值为-0（存储为 0x8000000000000000），供测试用。
NAN	8 字节双精度实型，值为 NAN（存储为 0xFFFFFFFFFFFFFFFF），供测试用。

12.2.3 类型转换

以上 6 种数据类型，可以互相转换。用户宏程序支持显式类型转换和隐式类型转换。

解释

- **显式类型转换**

显示类型转换格式如下：

[类型][表达式]　　当 COMP=1 时

或

(类型)(表达式)　　当 COMP=0 时

类型：INT、UINT、LONG、ULONG、FLOAT 或 DOUBLE。

表达式：当为变量时，可以去掉两侧括号。

例

语句：#1=[INT]5.5 ;

结果：#1=5。

- **隐式类型转换**

对于同一表达式中的常量和变量，如果数据类型不一致，则系统首先进行隐式类型转换。将较小的数据类型提升到较大的数据类型。数据类型从小到大排列如下：

INT、UINT、LONG、ULONG、FLOAT、DOUBLE

例

语句：#1=5.5+8 ;

结果：#1=5.5+[DOUBLE]8。

5.5 为 DOUBLE 型，8 为 UINT 型，DOUBLE>UINT，那么计算时首先将 8 转换为 DOUBLE 型，然后进行计算。

12.3 系统变量

系统变量用于读写系统内部的数据，例如，刀具补偿和当前位置数据等。其中，某些系统变量只能读不能写。系统变量是自动化程序和通用程序设计的基础。

解释

- **接口信号**

接口信号是可编程机床控制器（PLC）与用户宏程序之间交换的信号。

变量号	功能
#1000~#1015 #1032	把 16 位信号从 PLC 传送到用户宏程序。变量#1000~#1015 用于按位读信号，变量#1032 用于一次读整个 16 位信号。
#1100~#1115 #1132	把 16 位信号从用户宏程序传送到 PLC。变量#1100~#1115 用于按位写信号，变量#1132 用于一次写整个 16 位信号。

系统读取到作为接口信号的系统变量#1000~#1015 的值后，便可知接口输入信号的状态（0 为打开，1 为闭合）。其对应 DI 地址定义表如下。

序号	15	14	13	12	11	10	9	8
接口	UI15	UI14	UI13	UI12	UI11	UI10	UI09	UI08
G 区地址	G15.7	G15.6	G15.5	G15.4	G15.3	G15.2	G15.1	G15.0
宏地址	#1015	#1014	#1013	#1012	#1011	#1010	#1009	#1008

序号	7	6	5	4	3	2	1	0
接口	UI07	UI06	UI05	UI04	UI03	UI02	UI01	UI00
G 区地址	G14.7	G14.6	G14.5	G14.4	G14.3	G14.2	G14.1	G14.0
宏地址	#1007	#1006	#1005	#1004	#1003	#1002	#1001	#1000

读取系统变量#1032，可以一次读取全部输入信号，表达式如下：

$$\#1032=\sum_{i=0}^{15}\#(1000+i)\times 2^{i}$$

用户也可通过写系统变量#1100~#1115 来控制接口输出信号的状态（0 为打开，1 为闭合）。其对应 DO 地址定义表如下。

序号	15	14	13	12	11	10	9	8
接口	UO15	UO14	UO13	UO12	UO11	UO10	UO09	UO08
F 区地址	F15.7	F15.6	F15.5	F15.4	F15.3	F15.2	F15.1	F15.0
宏地址	#1115	#1114	#1113	#1112	#1111	#1110	#1109	#1108

序号	7	6	5	4	3	2	1	0

序号	7	6	5	4	3	2	1	0
接口	UO07	UO06	UO05	UO04	UO03	UO02	UO01	UO00
F 区地址	F14.7	F14.6	F14.5	F14.4	F14.3	F14.2	F14.1	F14.0
宏地址	#1107	#1106	#1105	#1104	#1103	#1102	#1101	#1100

通过给系统变量#1132 赋值，可以一次输出全部输出信号，表达式如下：

$$\#1132=\sum_{i=0}^{15}\#(1100+i)\times 2^{i}$$

注

1 不能赋值给系统变量#1000~#1032。

2 通过诊断界面的 G 接口可以显示#1000~#1032，其中 G014 对应 UI00~UI07，G015 对应 UI08~UI15。

3 当非 0 或 1 的值赋给系统变量#1100~#1115 时，被认为是 1。

4 可以读取系统变量#1100~#1132 的值。

5 通过诊断界面的 F 接口可以显示#1100~#1132，其中 F014 对应 UO00~UO07，F015 对应 UO08~UO15。

6 系统宏输入或输出接口信号的具体意义请参见操作手册或连接调试手册。

- **刀具补偿值**

通过系统变量可以读写刀具补偿值，可使用的变量数取决于刀补数。刀具补偿号与变量号的关系如下所示。

补偿号	1 轴长度补偿形状变量号	2 轴长度补偿形状变量号	3 轴长度补偿形状变量号	4 轴长度补偿形状变量号	5 轴长度补偿形状变量号	6 轴长度补偿形状变量号	7 轴长度补偿形状变量号	8 轴长度补偿形状变量号	半径补偿形状变量号
1	#10001	#20001	#30001	#40001	#50001	#60001	#70001	#80001	#90001
2	#10002	#20002	#30002	#40002	#50002	#60002	#70002	#80002	#90002
……	……	……	……	……	……	……	……	……	……
256	#10256	#20256	#30256	#40256	#50256	#60256	#70256	#80256	#90256

补偿号	1 轴长度补偿磨损变量号	2 轴长度补偿磨损变量号	3 轴长度补偿磨损变量号	4 轴长度补偿磨损变量号	5 轴长度补偿磨损变量号	6 轴长度补偿磨损变量号	7 轴长度补偿磨损变量号	8 轴长度补偿磨损变量号	半径补偿磨损变量号
1	#18001	#28001	#38001	#48001	#58001	#68001	#78001	#88001	#98001
2	#18002	#28002	#38002	#48002	#58002	#68002	#78002	#88002	#98002
……	……	……	……	……	……	……	……	……	……
256	#18256	#28256	#38256	#48256	#58256	#68256	#78256	#88256	#98256

- **0 号刀补变量号**

	1 轴	2 轴	3 轴	4 轴	5 轴	6 轴	7 轴	8 轴
工件偏移	#19000	#29000	#39000	#49000	#59000	#69000	#79000	#89000
刀具偏移	#19001	#29001	#39001	#49001	#59001	#69001	#79001	#89001

- **时间信息**

时间信息相关宏变量如下表所示。

变量号	数据类型	功能
#3001	UINT	该变量为一个计时器，以 1 毫秒为计时单位。当电源接通时，该变量复位为 0。当达到 4294967296 毫秒时，该计时器返回到 0。
#3002	FLOAT	该变量为一个计时器，当循环启动灯亮时以一小时为单位计时。当电源接通时，该变量复位为 0。
#3011	UINT	该变量用于读取当前日期（年/月/日），年/月/日信息转换成十进制数。例如，2006 年 9 月 21 日表示为 20060921。若存储不正确的日期，会产生报警（PS1303）。
#3012	UINT	该变量用于读取当前时间（时/分/秒），时/分/秒信息转换成十进制数。例如，下午 5 时 22 分 18 秒表示为 172218。若存储不正确的时间，会产生报警。（PS1303）。

- **转矩信息**

变量号	变量类型	功能
#3020~#3027	INT	轴转矩值，轴未接时无法读取。

- **已加工的零件数**

已加工的零件数（加工件数）可以被宏程序读取。

变量号	变量类型	功能
#3901	UINT	已加工零件数（加工件数）（不大于 65535 的非负数）。
#3902	UINT	已加工单批计数值（不大于 65535 的非负数）。

- **模态信息**

当前程序段的模态信息可以被宏程序读出（只读）。对应模态 G 代码的变量号=组号+4000，宏变量值为 G 代码值。具体宏变量如下所示。

变量号	变量类型	功能
#4001	UINT	G00、G01、G02、G03、G93
#4002	UINT	G17、G18、G19
#4003	UINT	G90、G91
#4004	UINT	G98、G99
#4005	UINT	G54、G55、G56、G57、G58、G59
#4006	UINT	G20、G21
#4007	UINT	G40、G41、G42
#4008	UINT	G49、G43、G44
#4009	UINT	G80、G81、G82、G83、G84、G85、G86、G87、G88、G89、G73、G74、G76
#4010	UINT	G94、G95
#4011	UINT	G15、G16
#4012	UINT	G50、G51
#4013	UINT	G68、G69
#4016	UINT	G12.1、G13.1
#4099	UINT	G 代码体系 A/B
#4105	UINT	F 代码

变量号	变量类型	功能
#4107	UINT	H 代码
#4111	UINT	L 代码
#4112	UINT	M 代码
#4113	UINT	顺序号
#4114	UINT	程序号
#4118	UINT	S 代码
#4119	UINT	T 代码
#4126	UINT	第一主轴 S 指令值
#4127	UINT	第二主轴 S 指令值
#4128	UINT	第三主轴 S 指令值

注

1 #4001~#4128 读取出来的数值默认为 BIN 码。

2 某些地址的赋值数据必须是 BCD 码，如 G、S（两位数）、T、M 等。使用时可提前通过 BCD 函数进行数制转换。例如:

(1) 若#4007=41，指令 G[#4007]，则等价于 G25，通过变量赋值会自动转为 BCD 码。

(2) 若指令 G[BCD[#4007]]，则等价于 G41，提前使用 BCD 函数进行了数制转换。

3 除数制转换外还可通过参数“BADR”（P2502.6）设置 G/M/S/T 通过变量赋值时是否需要转为 BCD 码。如果 BADR 设为 1，则使用 G/M/S/T 通过变量赋值时不需要转为 BCD 码。

2502		BADR						

[数据类型] 位型

BADR

0：G/M/T/S 等代码通过宏变量赋值时需要转为 BCD 码

1：G/M/T/S 等代码通过宏变量赋值时不需要转为 BCD 码

- **位置信息**

读取系统变量#5001~#5088 后，便可知当前刀具的位置信息。当公制输入时最小单位为 0.001 毫米，英制输入时最小单位为 0.0001 英寸。

系统变量	变量类型	位置信息	移动中读取	刀具半径、长度补偿
#5001~5008	DOUBLE	前段 1~8 轴程序段终点位置（ABSIO）	可	不考虑刀尖位置（程序指令位置）
#5011~5018	DOUBLE	1~8 轴机床位置（ABSMT）	不可	
#5041~5048	DOUBLE	1~8 轴现在位置（ABSOT）	不可	考虑刀具基准点位置（与绝对坐标值相同）
#5061~#5068	DOUBLE	1~8 轴 G31 指令跳跃时的绝对坐标	可	
#5080	DOUBLE	刀具半径补偿量	可	
#5081~5088	DOUBLE	刀具长度补偿量（1~8 轴）	可	

注

系统变量#5001~#5088 为只读，不能对其赋值。

- **工件坐标系**

工件坐标系信息系统变量可通过宏变量进行读写

系统变量	变量类型	坐标系
#5541~#5548	DOUBLE	G54 坐标系
#5551~#5558	DOUBLE	G55 坐标系
#5561~#5568	DOUBLE	G56 坐标系
#5571~#5578	DOUBLE	G57 坐标系
#5581~#5588	DOUBLE	G58 坐标系
#5591~#5598	DOUBLE	G59 坐标系

附加工件坐标系系统变量读写

系统变量	变量类型	坐标系
#6001~#6008	DOUBLE	G54P0
#6011~#6018	DOUBLE	G54P1
……	……	……
#6981~#6988	DOUBLE	G54P98
#6991~#6998	DOUBLE	G54P99

地址计算规则：G54Pn 坐标系（n=0~99），对应系统变量#(6000 + n * 10 + 1)~#(6000 + n * 10 + 8)

注：

1 使用附加工件坐标系（EXT 坐标系），需 P500.4 控制开启

2 上述变量只有在工件坐标系功能有效时才能访问，否则报警“加载变量失败”或“存储变量失败”。

- **驱动器阶跃指令**

系统变量	变量类型	功能
#9800	DOUBLE	写入 34343434 开启阶跃速度输出功能
#9801~#9808	DOUBLE	写入 1~8 轴速度指令 VREF 值，范围 -0x40000000~0x40000000。超过该范围时，自动限定到此范围。0x40000000 对应最大转速。
#9811~#9818	DOUBLE	写入 1~8 轴阶跃速度输出持续时间，单位：ms。超过 500 时，变量写入失败。

[示例](设置系统参数 P1121-1=0，开环)

```
#9800 = 34343434 ;
#9801 = 0x40000000 ;    // 写入最大速度
#9811 = 2 ;                    // 持续 2ms
```

12.4 算术、逻辑和关系运算

下表列出的运算可以在宏变量之间进行。赋值运算符右侧表达式中的变量（#j 和#k）可以由常量

或另外一个表达式代替，左侧表达式中的变量号也可以使用表达式。

功能	格式	备注
赋值	#i = #j ;	赋值运算。
加法 减法 乘法 除法	#i = #j + #k ; #i = #j - #k ; #i = #j * #k ; #i = #j / #k ;	算术运算。 如果 j 与 i 相等，可使用简化符号（+=、-=、*=、/=）。如#i = #i + #k 可简化为#i += #k。
逻辑与 逻辑或 逻辑非	#i = #j && #k ; #i = #j | | #k ; !	逻辑运算，常用于判定条件之间的运算关系。 逻辑与运算，即判定两个条件均为真时为真，任意条件为假时运算结果为假。 逻辑或运算，即判定两个条件中任一条件符合，即运算结果为真。 逻辑非，对逻辑变量的否定。
左移 右移 异或	#i = #j << #k ; #i = #j >> #k ; #i = #j ^ #k ; 或 #i = #j XOR #k ;	如果 j==i，那么可使用简化符号（&=、^=、|=、<<=、>>=）。如#i = #i & #k 可简化为#i &= #k。
等于 不等于 大于 大于等于 小于 小于等于	#i = #j == #k ; 或 #i = #j EQ #k ; #i = #j != #k ; 或 #i = #j NE #k ; #i = #j > #k ; 或 #i = #j GT #k ; #i = #j >= #k ; 或 #i = #j GE #k ; #i = #j < #k ; 或 #i = #j LT #k ; #i = #j <= #k ; 或 #i = #j LE #k ;	关系运算。 结果为 32 位无符号整数 0（FALSE）或 1（TRUE）。
位运算与 位运算或 位运算取反	#i = #j & #k ; 或 #i = #j AND #k ; #i = #j | #k ; 或 #i = #j OR #k ; #i = ~#j	位操作。只能对整型数进行位操作，使用实型数时将报警。位操作是按整型数的二进制形式操作的。结果的数据类型为无符号整型或无符号长整型。当有长整型数据参与位运算时，结果是无符号长整型，否则结果是无符号整型。
从 BCD 转为 BIN 从 BIN 转为 BCD	#i = BIN[#j] ; #i = BCD[#j] ;	主要用于与 PLC 的信号交换。 详情可参考 I-12.11.1。
平方根 绝对值 四舍五入 上取整 下取整 自然对数 指数函数 余数	#i = SQRT[#j] ; #i = ABS[#j] ; #i = FABS[#j] ; #i = ROUND[#j] ; #i = FUP[#j] ; 或 #i = CEIL[#j] ; #i = FIX[#j] ; 或 #i = FLOOR[#j] ; #i = LN[#j] ; 或 #i = LOG[#j] ; #i = EXP[#j] ; #i = #j REM #k;	详情可参考 I-12.11.1。
正弦	#i = SIN[#j] ;	三角函数。当以角度指定时，如$90^{\circ}30'$

功能	格式	备注
反正弦 余弦 反余弦 正切 反正切	#i = ASIN[#j] ; #i = COS[#j] ; #i = ACOS[#j] ; #i = TAN[#j] ; #i = ATAN[#j]/[#k] ;	表示为 90.5 度。 常数或表达式可以代替#j。 详情可参考 I-12.11.1。

解释

- **注意事项**

1 由参数“DEG”（P4100.6）决定以角度指定还是弧度编程。当 DEG 设为 0 时，所有三角函数都以弧度指定；当 DEG 设为 1 时，所有三角函数都以角度指定。

2 上表中列出的函数为常用函数，除此之外，C 函数库中绝大部分函数在系统上都能使用，详情可参考 I-12.11.1。

3 当使用双精度进行计算时，可以精确到约 15 位有效数字。当使用单精度计算时，可以精确到约 6 位有效数字。双精度运算精度高，但运算速度比单精度慢，而单精度运算则相反。

4 可以使用以 GE/LE/EQ/GT/LT/NE 为开头并加一个以上字母构成的别名，例如：“LTSS”可以作为别名使用

- **优先级**

运算符优先级如下表所示。

优先级	运算符	含义
1	()、[]	括号表达式
2	+、-、!、~、(类型)	单目运算符
3	*、/	双目运算符
4	+、-	
5	<<、>>	
6	<、<=、>、>=	
7	==、!=	
8	&、^、|	
9	&&、||	
10	=、+=、-=、*=、/=、<<=、>>=	赋值运算符

其中，括号表达式优先级最高，而赋值运算符优先级最低。计算时，先计算优先级高的。如果优先级一样，则采用从左到右的顺序计算。

例

语句：#1 = [1.0 + 2.0] * 3.0 / 4.0 – [8 >> 2 + 3] ;的计算顺序如下：

1 1.0 + 2.0

2 [1.0 + 2.0] * 3.0

3 [1.0 + 2.0] * 3.0 / 4.0

4 2 + 3

5 8 >> 2 + 3

6 [1.0 + 2.0] * 3.0 / 4.0 – [8 >> 2 + 3]

7 #1 = [1.0 + 2.0] * 3.0 / 4.0 – [8 >> 2 + 3]

12.5 加工程序语句

系统供用户使用的加工程序语句包括：NC 语句、赋值语句、控制语句和命令语句。其中，赋值语句、控制语句和命令语句为宏语句。

解释

- **赋值语句**

赋值语句包含赋值运算符。

例

#1 = #2 ;

- **控制语句**

控制语句包括：GOTO、IF、WHILE 等语句。

例

IF[#1 >= 100] GOTO #2 ;

- **命令语句**

命令语句包括：CALL、MSG、ERR、OPENWRITE、CLOSEWRITE、WRITE 等语句。

例

MSG[“Error!”] ;

- **宏语句与 NC 语句的区别**

宏语句与 NC 语句的不同点如下：

1 即使处于单程序段运行方式，机床也不停止运行。此功能由参数“SBKM”（P4100.7）控制，当 SBKM 设为 1 时，宏指令在单程序段方式下停止；当 SBKM 设为 0 时，宏指令在单程序段方式下不停止。

2 在刀具半径补偿方式中宏程序语句段做为不移动程序段处理。

12.6 控制语句

程序是顺序、选择、循环三种结构的复杂组合。宏程序语言使用一组相关的控制语句，来实现选择结构和循环结构：

1 选择控制语句 IF

2 循环控制语句 WHILE

3 转移控制语句 GOTO

12.6.1 无条件转移（GOTO 语句）

格式

GOTO n ;

n：要转移到的程序段顺序号。

解释

GOTO 语句表示转移到标有顺序号|N|（n 的绝对值）的程序段。

可用括号表达式或宏变量指定顺序号，但计算结果不能为非整型数据，否则报警。n 可以为负整数，表示负向优先或负向搜索。如果要向后跳转，可指定 n 为负值，节省搜索时间。

例

……

N10 M03 ;

#1=10 ; （给#1 赋值 10）

GOTO #1 ; （无条件跳转到 N10）

……

执行后，程序跳转到 N10，输出 M03 代码。

注

直接指定 n，且 n 为负数时，需加中括号。例如，GOTO [-1] ;

12.6.2 条件转移（IF 语句）

IF 语句用来判断给定的条件是否满足，根据结果（真或假）来选择执行相应的操作。有两种格式的 IF 语句。

格式

- **格式 1**

IF[表达式] GOTO n ;

n：表达式为真时转移到的程序段顺序号。

- **格式 2**

IF[表达式] THEN 赋值语句 ;

解释

- **格式 1**

如果表达式结果不为 0，则认为结果为真，转移到有顺序号为 n 的程序段。

如果表达式结果为 0，则认为结果为假，执行下个程序段。表达式结果如果为非整数，则应强制转换为整数，否则报警。

例

……

#1=1 ; （给#1 赋值 1）

N10 #2+=#1 ; （#2 恒加#1）

IF[#2<100] GOTO 10 ; （如果#2 小于 100 则程序跳转到 N10）

……

该程序对#2 累加，当等于 100 时程序向下顺序执行。

注

关系运算表达式结果为 32 位整数。

- **格式 2**

如果表达式结果不为 0，则认为结果为真，执行赋值语句，否则执行下个程序段。表达式结果如果为非整数，则应强制转换为整数，否则报警。

例

```
……
#1=10 ;                          （给#1 赋值 10）
N10 #1-=1 ;                      （#1 恒减 1）
IF [#1>1] GOTO 10 ;              （如果#1 大于 1，则继续执行 N10）
IF [#1<=2] THEN #1=10 ;          （如果#1 小于等于 2，给#1 赋值 10）
……
```

该程序执行时，#1 每次都从 10 减到 1。

12.6.3 循环（WHILE 语句）

格式

- **格式 1**

```
WHILE[表达式] DO n ;
……
END n ;
```

n：指定循环执行范围的标号（1~1023）。

- **格式 2**

```
DO n ;
……
END n ;
```

n：指定循环执行范围的标号（1~1023）。

解释

- **格式 1**

如果表达式结果不为 0，则认为结果为真，循环执行 WHILE 到 END 之间的语句，直到表达式结果为假。当表达式结果为零时，跳转到 END 语句的下一个程序段执行。表达式结果如果为非整数，则应强制转换为整数，否则报警。n 为正整数，仅仅为了匹配 DO 和 END 语句。

例

```
……
N1 #1 = 1 ;
N2 #2 = 0 ;
N3 WHILE[#1 <= 100] DO 5 ;
N4 #2 += #1 ;          （计算 1+2+3+·····+100）
N5 #1 += 1 ;
N6 END 5 ;
……
```

该程序执行时，#1 从 1 加到 100，#2 得到的是 1+2+3+·····+100 的和。

- **格式 2**

当 DO 语句中没有指定 WHILE 判断条件语句时，则为无限循环。

如格式 1 中例程，如果不指定 WHILE 判断条件，则#1 和#2 的值无限加下去直至数据溢出报警。

例

……

N1 #1 = 1 ;

N2 #2 = 0 ;

N3 DO 5 ;

N4 #2 += #1 ; （计算 1+2+3+……）

N5 #1 += 1 ;

N6 END 5 ;

……

- **标号 n 与循环嵌套**

标号 n 可重复使用，循环也可以嵌套使用，但有以下限制：

1 DO n 和 END n 必须成对使用（n 值相同），且 DO 一定要在 END 之前出现。

2 两个循环不能交叉。

3 如果循环嵌套，那么子循环的标号不能与父循环相同。

4 GOTO 语句可以从循环体内转移到循环体外，但不能从循环体外转移到循环体内，否则执行到 END 语句会报警。

12.7 命令语句

命令语句是用户宏程序的一部分，包括 MSG、ERR、OPENWRITE、CLOSEWRITE、WRITE。命令语句仅在标准宏 B 功能中有效，在简易宏 B 功能中执行上述语句将触发 PS 报警。

12.7.1 字符串

以一对双引号括起来的字符序列为字符串。字符串不是基本数据类型，只在部分命令语句中使用。例如：“message”。目前不支持中文字符串。

12.7.2 MSG 命令

格式

MSG[参数 1，参数 2，……，参数 n] ;

向消息区输出消息（一条）。

解释

参数可以为任意个（包括 0 个）。参数 1~n 可以是字符串或者表达式。参数若为字符串，则输出字符串；若为表达式，则将计算结果转换为相应的字符串，然后输出。

例

语句：#1=5+2

MSG[“5+2=”, #1] ;

结果：在消息区显示[MSG]5+2=7

12.7.3 ERR 命令

格式

ERR[报警号，参数 1，参数 2，……，参数 n] ;

报警并停止运行，同时向消息区连续显示消息（一条）。

解释

报警页面显示的报警号为 ERR 命令指定的报警号加上 3000。报警号可以是表达式，但结果必须是小于 1000 的整型数，否则报警。参数可以为任意个（包括 0 个），参数 1~n 可以是字符串或者表达式。参数若为字符串，则输出字符串，若为表达式，则将计算结果转换为相应字符串，然后输出。

例

语句：ERR[5, “error”] ;

结果：报 3005 号 P/S 报警，并在报警页面的提示区域显示 error，同时向消息区显示该内容。

12.7.4 OPENWRITE 命令

格式

OPENWRITE [handle] ;

打开 handle 号句柄，准备写入数据。

handle =0，指向内存。

handle =1，指向串口。

handle =2，指向 USB。

当前版本只有 handle =1 有效。

12.7.5 CLOSEWRITE 命令

格式

CLOSEWRITE [handle] ;

关闭 handle 号句柄，结束数据写入。

handle =0，指向内存。

handle =1，指向串口。

handle =2，指向 USB。

当前版本只有 handle =1 有效。

12.7.6 WRITE 命令

格式

WRITE [handle, 参数 1, 参数 2, ……, 参数 n] ;

向 handle 号句柄依次写入数据。

handle =0，指向内存。

handle =1，指向串口。

handle =2，指向 USB。

当前版本只有 handle =1 有效。

解释

参数可以为任意个（包括 0 个）。参数 1~n 可以是字符串或者表达式。参数若为字符串，则输出字符串；若为表达式，则将计算结果转换为相应的字符串，然后输出。

例

语句：WRITE[1, “Caculate:\n5+2=”, 5+2] ;

结果：如果串口通讯软件已经准备好接收数据，则将在串口通讯软件中显示：

Caculate:

5+2=7

注

1 使用 WRITE 指令时，可以在字符串中使用转义字符“\n”和“\t”，所以“\n”和“\t”都将作为一个字符输出。

2 第一个 WRITE 指令使用前应该使用 OPENWRITE 指令打开串口。最后一个 WRITE 指令结束后，应该使用 CLOSEWRITE 指令关闭串口。

3 要使用 WRITE 功能，首先应该设置串口参数，并让串口通讯软件处于接收状态，如同接收程序一样。

12.8 宏程序调用

程序调用包括：NC 程序调用（M98）和宏程序调用。宏程序调用包括以下几种方法：

- G65 宏程序调用
- 自定义 G 代码宏程序调用
- 自定义 M 代码宏程序调用
- 自定义 T 代码宏程序调用
- CALL 调用

NC 程序调用不可以像宏程序调用那样传递参数，数据只能通过全局变量进行传递。

上述宏程序调用方法仅在标准宏 B 功能中有效，在简易宏 B 功能中无效。在简易宏 B 功能中执行 G65 将触发“非法 G 代码”报警；屏蔽自定义 G/M/T 代码参数；执行 CALL 指令将触发 PS 报警。

自变量

- **单一地址自变量**

调用宏程序时，可能需要给被调用函数传入若干变量，这些变量称为自变量。使用自变量时，CNC 将指定值赋给对应的局部变量。当参数“MCDEC”（P4101.4）设置为 1 时，宏程序调用时自变量

支持小数。

注

1 当变量地址输入小数点时，自动按固定位数的小数处理，小数位数为编程单位小数位数（即 P40 数值，英制+1）。

2 以下地址作为特殊含义的地址不支持该功能，即按原方式处理：F，NF，M，S，T。

3 模态地址作为宏程序调用的自变量地址时，模态值会保存且可能因为小数点扩大 1000 倍，编程时注意处理模态值。

4 建议使用 NA-NZ 作为传递小数的变量地址。

自变量使用的地址符以及与局部变量的关系如下表所示：

地址	变量号	地址	变量号	地址	变量号	地址	变量号
A	#1	H	#11	Q	#17	W	#23
B	#2	I	#4	R	#18	X	#24
C	#3	J	#5	S	#19	Y	#25
D	#7	K	#6	T	#20	Z	#26
E	#8	M	#13	U	#21		
F	#9	P	#16	V	#22		

注

1 D、H、M、S、T 不可带小数点使用。

2 地址 M、S、T 按 BIN 码指定自变量。如下例子：

```
O8500;
G65 P8600 T11;
M30;

O8600;
#120=0.02;
#[98000+#20]=#120;          此程序段的含义是设定 11 号刀补#98011 的刀具半径磨损
M99;
```

- **扩展 NC 字地址自变量**

调用宏程序时，为了避免自变量数量不够或与数据地址冲突等问题，添加扩展 NC 字地址自变量，与单一地址自变量相同，使用自变量时，CNC 将指定值赋给对应的局部变量。自变量使用的地址符以及与局部变量的关系如下表所示：

地址	变量号	地址	变量号	地址	变量号
NA	#27	NJ	#36	NS	#45
NB	#28	NK	#37	NT	#46
NC	#29	NL	#38	NU	#47
ND	#30	NM	#39	NV	#48
NE	#31	NN	#40	NW	#49
NF	#32	NO	#41	NX	#50

地址	变量号	地址	变量号	地址	变量号
NG	#33	NP	#42	NY	#51
NH	#34	NQ	#43	NZ	#52
NI	#35	NR	#44		

注：

NG、NL、NM、NN、NP、NS、NT 不可带小数点使用。

例：

```
O0001
G65 P0002 NA10 NB11 NC 12 NF 12.7；
M30；

O0002
#100=#27+#28+#29-#32；
#500=#27；
#501=#28；
#502=#29；
#503=#32；
M99；
```

运行结果：

查看宏变量，#100=20.3，#500=10，#501=11，#502=12，#503=12.7

地址	变量号	地址	变量号
CT	#53	AC	#55
AD	#54	DC	#56
S1	#19	S2	#57
S3	#58	RS	#21

注

1 CT，RS 不可带小数点使用；AD，AC，DC 支持带小数点使用

2 S1，S2，S3 同 S 类似，使用时地址按 BCD 码指定自变量

3 S1，S2，S3 使用格式：

使用“字母+数字”类型的扩展 NC 字地址自变量时，使用时需使用“自变量”+“=”+“传递数据”格式使用，且等号前不能插入空格。

例：G65 P1 S2=100； //使用 G65 调用 1 号程序，将数据 100 传入“S2“对应的宏变量地址 #57，此时#57=256，地址按 BCD 码指定自变量。

12.8.1 G65 宏程序调用

G65 指令仅在标准宏 B 功能中有效，在简易宏 B 功能中执行 G65 将触发“非法 G 代码”报警。

格式

G65 Pp Ln <自变量指定> ;

p：要调用的程序号（标准的 4 位程序号名）。

n：重复次数，需要调用子程序的次数。

自变量指定：要传给被调用宏程序的数据，指定需要定义的自变量数值。参考第 124 页关于“自变量”的说明。

举例

加工程序

```
O156 ;
#100=#1+#2+#3+#5+#6+#4+#7+#8+#9+#11+#17+#18+#19+#21+#22+#23+#24+#25+#26 ;
M99 ;
O0001 ;
G65 P156 A10.0 B11.0 C12.0 D13 E14.0 F15.0 H16 I17.0 J18.0 K19.0 Q20.0 R21.0 S22 U23.0
V24.0 W25.0 X26.0 Y27.0 Z28.0 ;
M30 ;
```

运行结果

程序 O0001 执行时，G65 将数据传到对应的变量中，由程序 O156 计算出#100 的值。

注

G65 P99 A#1 B#2 C#3，当#1，#2，#3 为空时允许调用子程序

12.8.2 G66 模态调用

在执行沿轴移动的程序段后，调用一个 G66 指定的宏程序，这个过程继续直到 G67 取消模态调用

位置； G66 Pxxxx Ln <自变量>

其中：P：调用的程序号

L：重复次数，省略时为 1 次

<自变量>：传递给宏程序的数据

说明

1. G66 的指定方法同 G65,但是与 G65 不同的是,指定 G66 的程序段不会立即调用对应宏程序，而是等待执行沿轴移动的程序段后才调用；
2. 当指定 G67 时，在下一个程序段中不再执行模态宏程序调用；
3. G66 指定允许嵌套，嵌套深度为 15 层；

限制

1. G66 和 G67 的程序段必须成对出现在相同的程序中。如果不在 G66 方式下指定 G67，则会有报警；
2. 在 G66 程序段，不进行宏程序调用，但是局部变量已经完成设定；
3. 没有轴移动指令的程序段不会调用 G66 指定的宏程序；
4. 应在取消刀补的情况下（G40 状态下）指定 G66；
5. 嵌套调用 G66 宏程序时，只有第一个 G66 宏程序允许刀补，其余应取消刀补；

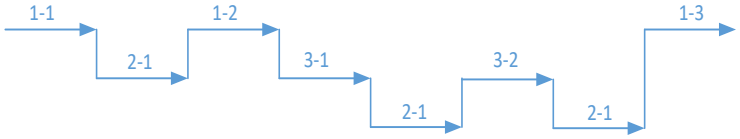
模态调用的嵌套

当 G66 多次嵌套时，在执行移动段后调用宏程序，调用最近的 G66 宏程序，如果在宏程序中有移动执行，则调用次近的 G66 宏程序，如此一层层展开下去。

【示列】

```
G66 P9100;          O9100;          O9200;
X10.0;  (1-1)       Z50.0;  (2-1)   X60.0;  (3-1)
G66 P9200;          M99;            Y70.0;  (3-2)
X150;   (1-2)
G67;
G67;
X-25.0;  (1-3)
M30;
```

上述程序的执行顺序为：



应用 G66 实现钻孔功能示列

主程序：进行孔定位

```
O1;
G28 X0 Y0 Z0;
G00 G90 Z50;
G00 G90 X100. Y50.
G66 P9100 Z-20. R5.0 F500;
X20. Y20.;
Y50.;
X70. Y80.
G67;
M30;
```

宏程序：实现钻孔；

```
O9100;
G00 G90 Z#18;
G01 Z#25F#9;
G00Z#18;
M99;
```

G66 实现阵列打孔示列

主程序：进行行定位

```
O1;
G28 X0 Y0 Z0;
G00 G90 Z50;
```

G66 P9100 Z-20. R5.0 F500;

G66 P9110;

G00 G90 X20.

X30.

X40.

G67;

G67;

M30;

宏程序：实现钻孔；

O9100；

G00 G90 Z#18;

G01 Z#25F#9；

G00Z#18;

M99;

宏程序：阵列钻孔；

O9110；

G00 G90 Y0;

Y50;

Y100;

Y150;

Y200;

M99;

12.8.3 自定义 G 代码宏程序调用

自定义 G 代码及其调用的宏程序号如下表所示。在简易宏 B 功能中自定义 G 代码参数将被屏蔽。

自定义 G 代码	代码设定参数	调用的宏程序号参数
1	4120	4320
2	4121	4321
3	4122	4322
4	4123	4323
5	4124	4324
6	4125	4325
7	4126	4326
8	4127	4327
9	4128	4328
10	4129	4329
11	4130	4330
12	4131	4331
13	4132	4332
14	4133	4333
15	4134	4334
16	4135	4335

自定义 G 代码	代码设定参数	调用的宏程序号参数
17	4136	4336
18	4137	4337
19	4138	4338
20	4139	4339

格式

G[自定义号] Ln <自变量指定>

自定义号：定义在参数 P4120~P4139 中。

n：重复次数。

自变量指定：要传到被调用宏程序的数据。参考第 124 页关于“自变量”的说明。

解释

- **自定义号**

最多可自定义 20 个 G 代码，通过参数 P4110 设置自定义 G 代码个数，在自定义参数 P4120~P4139 设置自定义号。如果原来系统中已有指定的 G 代码，则自定义 G 代码将代替原有 G 代码的功能。

注

自定义 G 代码不允许与 G 代码同一行执行。

- **调用的宏程序号**

定义在参数 P4320~4339 中。使得每个自定义 G 代码对应一个要调用的宏程序号，因此宏程序号对应的宏程序必须存在。

注

参数 P4320~4339 在出厂时一般为只读属性，被固化为 9320~9339。

- **自变量指定**

与 G65 自变量指定方式完全一致。

多级嵌套调用

调用子程序时，自定义的 G 代码在子程序中将作为普通代码执行，在多级嵌套的子程序中同样作为普通 G 代码执行。

例

参数设置自定义 G 代码 G04 调用 O9320，那么在 O9320 程序中，以及 O9320 可能嵌套调用的子程序中，G04 将作为普通 G 代码执行。

举例

如果要自定义 G60 指令，调用的宏程序号为 9320，应事先设置自定义 G 代码个数 P4110=1，然后设置自定义号 P4120=60，被调用的宏程序号参数 P4320，固化为 9320 无需设置。

加工程序

```
O9320 ;
#101=#24+#25+#26+#4+#5+#6+#18+#9 ;
M99 ;
O1000 ;
G60 L1 X100.0 Y200.0 Z300.0 I400.0 J500.0 K600.0 R700.0 F800 ;
M30 ;
```

运行结果

程序 O1000 中执行 G60 指令时，会调用 O9320 程序，并且将 100.0、200.0、300.0、400.0、500.0、600.0、800 分别传给#24、#25、#26、#4、#5、#6、#18、#9，然后由 O9320 计算出#101 的值。

12.8.4 自定义 M 代码宏程序调用

自定义 M 代码及其调用的宏程序号如下表所示。在简易宏 B 功能中自定义 M 代码参数将被屏蔽。

自定义 M 代码	代码设定参数	调用的宏程序号参数
1	4200	4400
2	4201	4401
3	4202	4402
4	4203	4403
5	4204	4404
6	4205	4405
7	4206	4406
8	4207	4407
9	4208	4408
10	4209	4409
11	4210	4410
12	4211	4411
13	4212	4412
14	4213	4413
15	4214	4414
16	4215	4415
17	4216	4416
18	4217	4417
19	4218	4418
20	4219	4419

格式

M[自定义号] Ln <自变量指定>

自定义号：定义在参数 P4200~P4219 中。

n：重复次数。

自变量指定：要传到被调用宏程序的数据。参考第 124 页关于“自变量”的说明。

解释

- **自定义号**

最多可自定义 20 个 M 代码，通过参数 P4111 设置自定义 M 代码个数，在自定义参数 P4200~P4219 设置自定义号。如果原来系统中已有指定的 M 代码，则自定义 M 代码将代替原有 M 代码的功能。

注

自定义 M 代码不允许与 G 代码同一行执行。

- **调用的宏程序号**

定义在参数 P4400~4419 中。使得每个自定义 M 代码对应一个要调用的宏程序号，因此宏程序号对应的宏程序必须存在。

注

参数 P4400~4419 在出厂时一般为只读属性，被固化为 9400~9419。

- **自变量指定**

与 G65 自变量指定方式完全一致。

- **多级嵌套调用**

调用子程序时，自定义的 M 代码在子程序中将作为普通代码执行，在多级嵌套的子程序中同样作为普通 M 代码执行。

例

参数设置自定义 M 代码 M04 调用 O9400，那么在 O9400 程序中，以及 O9400 可能嵌套调用的子程序中，M04 将作为普通 M 代码执行。

举例

如果要自定义 M60 指令，调用的宏程序号为 9400，应事先设置自定义 M 代码个数 P4111=1，然后设置自定义号 P4200=60，被调用的宏程序号参数 P4400，固化为 9400 无需设置。

加工程序

```
O9400 ;
#101=#24+#25+#26+#4+#5+#6+#18+#9 ;
M99 ;
O1000 ;
M60 L1 X100.0 Y200.0 Z300.0 I400.0 J500.0 K600.0 R700.0 F800 ;
M30 ;
```

运行结果

程序 O1000 中执行 M60 指令时，会调用 O9400 程序，并且将 100.0、200.0、300.0、400.0、500.0、600.0、800 分别传给#24、#25、#26、#4、#5、#6、#18、#9，然后由 O9400 计算出#101 的值。

12.8.5 自定义 T 代码宏程序调用

自定义 T 代码及其调用的宏程序号如下表所示。在简易宏 B 功能中自定义 T 代码参数将被屏蔽。

自定义 T 代码	代码设定参数	调用的宏程序号参数
1	4250	4450
2	4251	4451
3	4252	4452
4	4253	4453
5	4254	4454
6	4255	4455
7	4256	4456
8	4257	4457
9	4258	4458
10	4259	4459

格式

T[自定义号] Ln <自变量指定>

自定义号：定义在参数 P4250~P4259 中。

n：重复次数。

自变量指定：要传到被调用宏程序的数据。参考第 124 页关于“自变量”的说明。

解释

- **自定义号**

最多可自定义 10 个 T 代码，通过参数 P4112 设置自定义 T 代码个数，在参数 P4250~P4259 设置自定义号。如果原来系统中已有指定的 T 代码，则自定义 T 代码将代替原有 T 代码的功能。

注

自定义 T 代码不允许与 G 代码同一行执行。

调用的宏程序号

定义在参数 P4450~4459 中。使得每个自定义 T 代码对应一个要调用的宏程序号，因此宏程序号对应的宏程序必须存在。

注

参数 P4450~4459 在出厂时一般为只读属性，被固化为 9450~9459。

- **自变量指定**

与 G65 自变量指定方式完全一致。

- **多级嵌套调用**

调用子程序时，自定义的 T 代码在子程序中将作为普通代码执行，在多级嵌套的子程序中同样作为普通 T 代码执行。

例

参数设置自定义 T 代码 T04 调用 O9450，那么在 O9450 程序中，以及 O9450 可能嵌套调用的子程序中，T04 将作为普通 T 代码执行。

- **举例**

如果要自定义 T60 指令，调用的宏程序号为 9450，应事先设置自定义 T 代码个数 P4112=1，然后设置自定义号 P4250=60，被调用的宏程序号参数 P4450，固化为 9450 无需设置。

加工程序

```
O9450 ;
#101=#24+#25+#26+#4+#5+#6+#18+#9 ;
M99 ;
O1000 ;
T60 L1 X100.0 Y200.0 Z300.0 I400.0 J500.0 K600.0 R700.0 F800 ;
M30 ;
```

运行结果

程序 O1000 中执行 T60 指令时，会调用 O9450 程序，并且将 100.0、200.0、300.0、400.0、500.0、600.0、800 分别传给#24、#25、#26、#4、#5、#6、#18、#9，然后由 O9450 计算出#101 的值。

- **所有 T 代码调用同一子程序功能**

2511	T 代码调用的子程序号(0:功能无效)

[数据类型] 字型

[数据范围] 0-9999

[默认设置] 0

功能开启，设置程序号，任意 T 代码均可调用设置的程序。

T 代码调用检查顺序：

1 P4112 功能有效时，如果通过系统参数 P4250~P4259 将某一 T 代码指定为自定义 T 代码，则调用该 T 代码时将执行该自定义 T 代码指定的子程序；

2 P2511 功能有效时，执行任意 T 代码，将调用该参数指定的子程序，并且 T 代码数值被传入 20 号局部变量，在子程序中再执行同一 T 代码时，将作为普通 T 代码执行

3 P4112 与 P2511 功能均开启时，P4112 的优先级高于 P2511。因此使用 P4250~P4259 自定义的 T 代码时，只调用 O9450~O9459。指定 P4112 定义外的 T 代码调用 P2511 设定的子程序。

4 传入#20 号局部变量的 T 代码数值为 2 进制结果。

例： T20，#20=32

- **同行编入多个 T 代码并调用子程序功能**

2501					SMT			

[数据类型] 位型

SMT

0：P2511 不为一行多个 T 代码功能的子程序号

1：P2511 为一行多个 T 代码功能的子程序号

当参数 SMT（P2501.3）=1，P2511=0 时

1 编程时，一行中可编入多个 T 代码，且 T 代码位数最高为 4 位，即 T1-T9999

2 一行中编入 T 代码数量最多为 32 个，超过个数后触发 PS1216：“T 代码数量过多”

3 执行时将调用由参数 2511 指定的宏程序，宏程序的局部变量从#32 开始将被依次设置为 T 代码的值，局部变量中出现第一个 0 值表示局部变量传参结束。例如 程序段 T11 T12 T1400; 该段执行结束后，局部变量为 #32=11，#33=12，#34=1400，#35=0，#35 作为结束标志。如果编入 32 个 T 代码，由于局部变量最大号为#63，那么将没有结束标志的 0 值出现。

4 使用该功能时，原 T 代码换刀功能无效，若需换刀则需要宏程序配合 PLC 进行换刀，换刀完成时必须置位 G12.7，否则 NC 不会更新刀具号。宏变量对应的换刀指令地址为 #1480~#1767，宏变量输出、输入地址对应为 F480~F767、G480~G767。

5 当 SMT=0 时，P2511 为之前的功能：自定义 T 代码以外的 T 代码执行时调用的子程序号

12.8.6 CALL 调用

CALL 调用可以传递参数，但不能包含 NC 指令。

CALL 调用如果传递了参数，则还必须对被调用程序进行参数列表声明，即在主程序中使用 CALL 指令调用子程序，并在子程序中使用 PARAMS 指令接收参数。如果没有传递参数，则不需要使用 PARAMS 指令。

CALL 调用没有返回值，如有需要可借助全局变量来实现。即在被调用子程序中为指定全局变量赋值，在主程序中读取该全局变量即可。

CALL 和 PARAMS 指令仅在标准宏 B 功能中有效，在简易宏 B 功能中执行它们将触发 PS 报警。

格式

- **CALL 指令**

CALL[程序号] ;

或

CALL[程序号，参数 1，参数 2，……，参数 n] ;

程序号：被调用程序的程序号。

n：参数个数（n<=32）。

- **PARAMS 指令**

PARAMS[参数 1，参数 2，……，参数 n] ;

n：参数个数（n<=32）。

解释

- **CALL 指令**

程序号可用表达式表示，其计算结果必须为整数，否则报警。

参数 1~n 为要传入被调用程序的参数，最多 32 个。参数可以为变量、表达式或数组。

- **PARAMS 指令**

参数 1~n 声明可传入的参数，最多 32 个。参数声明只能使用局部变量的别名（r0~r63）。

举例

加工程序

```
O1100 ;                    （主程序）
CALL[1000, 50.0, 100.0] ;  （调用子程序 1000，并传入参数 50.0 和 100.0。）
CALL[1000, 150.0, 200.0] ; （调用子程序 1000，并传入参数 150.0 和 200.0。）
CALL[1000, 250.0, 300.0] ; （调用子程序 1000，并传入参数 250.0 和 300.0。）
M30 ;

O1000 ;                    （子程序）
PARAMS[r1, r5] ;           （参数列表声明，分别用局部变量 r1（#1）和 r5（#5）接收参数）
G04 X0.5 ;
N10 G00 X[r1] Z[r5] ;
G04 X0.5 ;
M99 ;
```

12.9 变量索引和数组

为简化宏程序编程，系统提供了两种变量使用方法：变量索引和数组。

12.9.1 变量索引

用户宏程序可以使用变量索引表示一个变量，以方便循环程序的使用。

格式

变量[索引]　　**当 COMP=1 时**

或

变量(索引)　　**当 COMP=0 时**

表示一个变量，其变量号为指定变量的变量号加上索引值。

举例

把 100~199 号变量设置为 0~99。

加工程序

```
O1500 ;
#1 = 0 ;
WHILE[#1 < 100] DO 1 ;
#100[#1] = #1 ;      （使用变量索引）
#1 += 1 ;
END 1 ;
M30 ;
```

注

局部变量的别名也可以使用索引，如 r5[#1]。

12.9.2 数组

用户宏程序支持简单的数组操作，可以处理连续多个变量。

当参数“COMP”（P4100.5）设为 1 时，方括号用于变量索引、改变优先级、函数或命令等语句中，而圆括号用于注释；当 COMP 为 0 时，圆括号代替方括号功能，而方括号则用于数组功能，注释使用 C/C++语言注释风格（//或/*　*/）。如无特别说明，默认 COMP=1。

格式

变量[数组大小]

表示以指定变量为起始变量的连续多个变量，变量个数由指定数组大小决定。

解释

在用户宏程序中，数组主要有以下两种使用方法。

- **连续变量赋值**

变量 1[数组大小 1] op 变量 2[数组大小 2] ;

op：操作符，包括=、+、-、*、/、&=、^=、|=、<<=、>>=10 种操作。

变量 1 和变量 2：分别为左操作数的起始变量和右操作数的起始变量。

数组大小 1 和数组大小 2：可以为表达式，但结果必须为整数，且大小必须相同，否则报警。

例

```
语句：#100[3]+=r2[3] ;
等价于以下三条语句：
#100+=r2 ;
#101+=r3 ;
#102+=r4 ;
```

- **CALL 指令参数**

例

```
O1100 ;                          //主程序
#100(0)=50.0 ;                   //变量索引
#100(1)=100.0 ;
CALL(1000, #100[2]) ;            //调用子程序 1000，并传入数组
#100=150.0 ;
#101=200.0 ;
CALL(1000, #100[2]) ;            //调用子程序 1000，并传入数组
M30 ;
O1000 ;                          //子程序
PARAMS(r5[2]) ;                  //接收参数
G04 X0.5 ;
N10 G00 X(r5(0)) Z(r5(1)) ;      //变量索引
G04 X0.5 ;
M99 ;
```

12.10 中断型用户宏程序

当程序正在执行时，来自机床的中断输入信号可以调用另一个程序。该功能称为中断型用户宏程序功能。

格式

M96 Pxxxx ;	**用户宏程序中断有效**
M97 ;	**用户宏程序中断无效，中断信号被忽略**

xxxx：中断时被调用的程序的程序号。

解释

- **用途**

中断型用户宏程序功能允许在程序执行期间的任一程序段调用另一个程序。这就可使程序根据随时变化的条件实时地运行。例如：

1 当检测到刀具的异常状态时，用外部信号启动异常过程的处理。

2 加工运行的顺序由另一个加工操作中断而不取消现在的运行。

3 以固定的间隔，读取与现在加工有关的信息。

进入宏中断子程序时，自动复位各模态 G 代码组，在中断子程序返回时恢复模态数据。上面叙述的中断型用户宏程序类似于适应控制的应用。

- **中断条件**

用户宏程序中断只能在程序执行期间有效。即在下列条件下有效：

1 存储器运行或 MDI 运行时

2 启动指示灯亮时

3 目前不处理用户宏中断时

- **指令的使用**

一般情况下，指定 M96 使中断信号的输入有效，可以执行用户宏程序中断功能。M97 使信号无效。一旦指定 M96，中断信号的输入是用户宏程序中断开始。直到指定 M97 或 NC 复位后，中断型宏程序功能取消。在指定 M97 或 NC 复位后，即使中断信号输入，用户宏程序中断也不执行，中断信号被忽略直到指定另一个 M96。

- **用户宏程序中断和 NC 语句**

当执行用户宏程序中断时，用户可能希望中断正在执行的 NC 语句，或者希望执行完当前的程序段后再执行中断。这时，可通过参数“INTT”（P4100.3）选择是在程序段中间执行中断，还是等到程序段结束后再执行中断。

1 I 型（在程序段中间执行中断，INTT=1）

(1) 当中断信号输入时，任何正在执行的移动或暂停都被立即停止，转而执行中断程序。

(2) 当控制返回到被中断的程序时，如果在中断程序执行过程中坐标没有变化，则程序继续执行完被中断程序段；如果在中断程序执行过程中坐标发生了变化，则程序重新执行被中断程序段。

(3) 当控制返回到被中断的程序时，若参数“MINTISKP”（P4101.0）为 1，跳过当前段，若为 0 则不跳过当前段

2 II 型（程序段结束后执行中断，INTT=0）

(1) 如果正在执行的程序段不是循环操作，则当中断信号输入时，在当前程序段执行完之前，不执行宏程序的语句。

(2) 如果正在执行的程序段是循环操作，则当中断信号输入时，在整个循环操作完成之后，才执行宏程序的语句。

- **循环操作期间的用户宏程序中断**

对于复合型固定循环，如果被中断，循环有可能无法正常结束。所以请在循环之前使用 M97 指令禁止用户宏程序中断，在循环结束之后使用 M96 指令重新使能用户宏程序中断。

- **用户宏程序中断信号**

用户宏程序中断信号（诊断信号 X4.7 或 G11.4）的输入有两种方式：状态触发方式和边沿触发方式。使用状态触发方式，当它接通时（X4.7 或 G11.4 为 1），信号有效；使用边沿触发方式，当它从断开状态切换到接通状态时，该信号变为有效（X4.7 或 G11.4 由 0 变为 1 的上升沿有效）。用户可通过参数“TSE”（P4100.2）选择这两种方式。

- **限制**

1 M98 或 M99 会临时禁止用户宏程序中断，直到下一个指令段开始。因此，当系统在执行 M98 或 M99 指令段时，无法响应用户宏程序中断输入信号。

2 用户宏程序中断在 DNC 加工过程中不能使用。

3 使用用户宏程序中断功能时，WHILE、END 等宏控制语句之前应该使用 M97 禁止宏中断，在这类语句之后再使用 M96 重新使能宏中断。

4 宏中断有效时，指令 G90~G94/G178 报警

12.11 用户宏程序库及实数标准

12.11.1 用户宏程序数学库函数

数学运算库基于 IEEE754 标准的浮点库，实现了常用的数学函数。函数名称与计算机上 C 语言运行时库中相同功能的函数同名，小写形式为大写形式的别名。

常用数学函数

功能	单精度	双精度
获取浮点类型	FPCLASSIFYF	FPCLASSIFY
判断是否是有限数	FINITEF	FINITE
判断是否是无穷大 INF	ISINFF	ISINF
判断是否是非数字	ISNANF	ISNAN
绝对值	FABSF	FABS
下取整	FLOORF	FLOOR
上取整	CEILF	CEIL
截除小数部分	TRUNCF	TRUNC
圆整四舍五入	ROUNDF	ROUND
平方根	SQRTF	SQRT
反正弦	ASINF	ASIN
反余弦	ACOSF	ACOS
正弦	SINF	SIN
余弦	COSF	COS
正切	TANF	TAN

功能	单精度	双精度
反正切	ATANF	ATAN
以 2 为底的对数	LOG2F	LOG2
以 10 为底的对数	LOG10F	LOG10
自然对数	LOGF	LOG
自然对数底数的幂	EXPF	EXP
2 的幂	EXP2F	EXP2
幂	POWF	POW
取余数	REM	REM

其它函数

功能	函数名
十进制转二进制	BIN
二进制转十进制	BCD
整型数绝对值	ABS
判断变量是否为空	ISNULL

12.11.2 函数功能说明

FPCLASSIFYF、FPCLASSIFY

- **函数格式**

#i = FPCLASSIFYF[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = FPCLASSIFY[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
NAN	0
±INF	1
0	2
非规格化小数	3
规格化有限数	4

注

有关规格化数与非规格化数的区别，参见下一节

FINITEF、FINITE

- **函数格式**

#i = FINITEF[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = FINITE[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
NAN	0

输入值	返回值
±INF	0
有限数	1

ISINFF、ISINF

- **函数格式**

#i = ISINFF[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = ISINF[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
NAN	0
+INF	1
-INF	-1
有限数	0

ISNANF、ISNAN

- **函数格式**

#i = ISNANF[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = ISNAN[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
NAN	1
±INF	0
有限数	0

FABSF、FABS

- **函数格式**

#i = FABSF[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = FABS[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
NAN	NAN
±INF	+INF
有限数	绝对值

- **举例**

#1 = FABS[-8.5] 计算结果：#1 为 8.5

FLOORF、FLOOR（别名 FIX）

- **函数格式**

#i = FLOORF[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = FLOOR[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
NAN	NAN
±INF	±INF
有限数	返回浮点值表示不大于#j 的最大整数

- **举例**

#1 = FLOOR[-8.1] 计算结果：#1 为-9

#1 = FLOOR[8.9] 计算结果：#1 为 8

CEILF、CEIL（别名 FUP）

- **函数格式**

#i = CEILF[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = CEIL[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
NAN	NAN
±INF	±INF
有限数	返回浮点值表示不小于#j 的最大整数

- **举例**

#1 = CEIL[-8.9] 计算结果：#1 为-8

#1 = CEIL[8.1] 计算结果：#1 为 9

TRUNCF、TRUNC

- **函数格式**

#i = TRUNCF[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = TRUNC[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
NAN	NAN
±INF	±INF
有限数	返回浮点值表示#j 去掉小数部分后的整数

- **举例**

#1 = TRUNC[-8.9] 计算结果：#1 为-8

#1 = TRUNC[8.1] 计算结果：#1 为 8

ROUNDF、ROUND

- **函数格式**

#i = ROUNDF[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = ROUND[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
NAN	NAN

输入值	返回值
±INF	±INF
有限数	返回浮点值表示#j 四舍五入后的整数

- **举例**

#1 = ROUND[5.1] 计算结果：#1 为 5

#1 = ROUND[5.6] 计算结果：#1 为 6

ASINF、ASIN

- **函数格式**

#i = ASINF[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = ASIN[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
|#j|<=1	弧度值：$[-\frac{\pi}{2},\frac{\pi}{2}]$ 角度值：[-90,90]
|#j|>1	NAN
±INF	NAN
NAN	NAN

ACOSF、ACOS

- **函数格式**

#i = ACOSF[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = ACOS[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
|#j|<=1	弧度值：[0,π] 角度值：[0,180]
|#j|>1	NAN
±INF	NAN
NAN	NAN

ATANF、ATAN

- **函数格式 1**

#i = ATANF[#j]/[#k] 其中，#j、#k 被自动转化为 FLOAT 型数据

#i = ATAN[#j]/[#k] 其中，#j、#k 被自动转化为 DOUBLE 型数据

- **返回值 1**

输入值	返回值
（#k, #j）为平面上的坐标点	由该点构成角度的反正切值

注

当参数“NAT”（P4100.0）设为 0 时，返回值范围为弧度值：[0,2π] 角度值：[0,360]

- **函数格式 2**

#i = ATANF[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = ATAN[#j]　　其中，#j 被自动转化为 DOUBLE 型数据

- **返回值 2**

输入值	返回值
有限数	弧度值：$[-\frac{\pi}{2},\frac{\pi}{2}]$　角度值：[-90,90]
±INF	$\pm\frac{\pi}{2}$
NAN	NAN

注
如果 ATAN[#j]后有除号，则系统认为是格式 1，因此 ATAN[#j]/#k 会报警，应改为[ATAN[#j]]/#k

SINF、SIN

- **函数格式**

#i = SINF[#j]　　其中，#j 被自动转化为 FLOAT 型数据

#i = SIN[#j]　　其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
有限数（弧度值或角度值）	正弦值
±INF	NAN
NAN	NAN

COSF、COS

- **函数格式**

#i = COSF[#j]　　其中，#j 被自动转化为 FLOAT 型数据

#i = COS[#j]　　其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
有限数（弧度值或角度值）	余弦值
±INF	NAN
NAN	NAN

TANF、TAN

- **函数格式**

#i = TANF[#j]　　其中，#j 被自动转化为 FLOAT 型数据

#i = TAN[#j]　　其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
有限数（弧度值或角度值）	正切值
±INF	NAN
NAN	NAN

LOGF、LOG（别名 LN）

- **函数格式**

#i = LOGF[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = LOG[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
+INF	+INF
有限正数	自然对数
0	-INF
有限负数	NAN
-INF	NAN
NAN	NAN

LOG2F、LOG2

- **函数格式**

#i = LOG2F[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = LOG2[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
+INF	+INF
有限正数	以 2 为底的对数
0	-INF
有限负数	NAN
-INF	NAN
NAN	NAN

LOG10F、LOG10

- **函数格式**

#i = LOG10F[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = LOG10[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
+INF	+INF
有限正数	以 10 为底的对数
0	-INF
有限负数	NAN
-INF	NAN
NAN	NAN

EXPF、EXP

- **函数格式**

#i = EXPF[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = EXP[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
+INF	+INF
有限数	e 的幂值
-INF	0
NAN	NAN

EXP2F、EXP2

- **函数格式**

#i = EXP2F[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = EXP2[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
+INF	+INF
有限数	2 的幂值
-INF	0
NAN	NAN

SQRTF、SQRT

- **函数格式**

#i = SQRTF[#j] 其中，#j 被自动转化为 FLOAT 型数据

#i = SQRT[#j] 其中，#j 被自动转化为 DOUBLE 型数据

- **返回值**

输入值	返回值
+INF	+INF
有限非负数	平方根值
有限负数	NAN
-INF	NAN
NAN	NAN

POWF、POW

- **函数格式**

#i = POWF[#j,#k] 其中，#j，#k 被自动转化为 FLOAT 型数据

#i = POW[#j,#k] 其中，#j，#k 被自动转化为 DOUBLE 型数据

- **返回值**

$\#i = \#j^{\#k}$

- **规则**

1	POWF[x,0]=1	其中：x 为任意值
2	POWF[x,1]=x	其中：x 为任意值
3	POWF[x,NAN]=NAN	其中：x 为任意值
4	POWF[NAN,y]=NAN	其中：y!=0
5	POWF[x,+INF]=+INF	其中：|x|>1
6	POWF[x,-INF]=+0	其中：|x|>1
7	POWF[x,+INF]=+0	其中：|x|<1
8	POWF[x,-INF]=+INF	其中：|x|<1
9	POWF[x,±INF]=NAN	其中：x = ±1
10	POWF[+0,y]=+0	其中：$y \in \{y > 0 \mid y != +0, y != NAN\}$
11	POWF[-0,y]=+0	其中：$y \in \{y > 0 \mid y != 0, y != NAN, y$不是奇数$\}$
12	POWF[+0,y]=+INF	其中：$y \in \{y < 0 \mid y != -0, y != NAN\}$
13	POWF[-0,y]=+ INF	其中：$y \in \{y < 0 \mid y != -0, y != NAN, y$不是奇数$\}$
14	POWF[-0,y]=-POWF[+0, y]	其中：y 是任意奇数
15	POWF[+INF,y]=+INF	其中：$y \in \{y > 0 \mid y != +0, y != NAN\}$
16	POWF[+INF,y]=+0	其中：$y \in \{y < 0 \mid y != -0, y != NAN\}$
17	POWF[-INF,y]=-POWF[-0,-y]	
18	POWF[-x,y]=POW[-1,y]×POW[x,y]	其中：y 是任意整数
19	POWF[x,y]=NAN	其中：$x \in \{x < 0 \mid x != -0, x != -INF\}$，y 不是整数

BIN

- **函数格式**

#i = BIN[#j]　　十进制转二进制

BCD

- **函数格式**

#i = BCD[#j]　　二进制转十进制

ABS

- **函数格式**

#i = ABS[#j]　　取绝对值

- **返回值**

输入值	返回值
32 位有/无符号整数	|#j|，类型为 32 位无符号整数
64 位有/无符号整数	|#j|，类型为 64 位无符号整数

ISNULL

- **函数格式**

#i = ISNULL[#j]　　当#j 为空变量时，#i 为 1；否则，#i 为 0。
其中#i 和#j 可以为变量表达式或寄存器（r0、r1、……）。

例 1

```
#1 = NULL ;
#2 = 0 ;                    //注意 0 并不是空变量
#3 = ISNULL[#1] ; //结果#3=1
#4 = ISNULL[#2] ; //结果#4=0
```

例 2

```
O502 ;                                  //子程序
IF[ISNULL[#24]] THEN #24 = 50.0 ;       //如果#24 未定义，那么默认为 50.0
IF[ISNULL[#25]] THEN #25 = 100.0 ;      //如果#25 未定义，那么默认为 100.0
G00 X[#24] Y[#25] ;
M99 ;
O503 ;                                  //主程序
G65 P502 X-100. Y-100. ;                //调用 O502，定位到（-100.，-100.）
G65 P502 Y-50. ;                        //调用 O502，定位到（50.，-50.）。未指令 X，那么#24
                                        //取默认值 50.0
G65 P502 X-30. ;                        //调用 O502，定位到（-30.，100.）。未指令 Y，那么
                                        //#25 取默认值 100.0
G65 P502 ;                              //调用 O502，定位到（50.，100.）。#24 和#25 取默认
                                        //值
M30 ;
```

例 3

```
O500 ;
PARAMS[r1, r2] ;
IF[ISNULL[r1]] THEN r1 = 50. ;
IF[ISNULL[r2]] THEN r2 = 100. ;
G00 X[r1] Y[r2] ;
M99 ;
O501 ;
CALL[500, -100., -100.] ;
CALL[500, NULL, -50.] ;
CALL[500, -30., NULL] ;
CALL[500, NULL, NULL] ;
M30 ;
//程序 O501、O500 完成与 O503、O502 相同功能
```

12.11.3 实型数与 IEEE754 标准

浮点数格式

IEEE754 定义了三种类型的浮点数格式：

1 单精度（FLOAT）

2 双精度（DOUBLE）

3 扩展双精度（DOUBLE-EXTEND）

浮点数存储格式

IEEE754 浮点数存储格式如下：

1 FLOAT：1 位符号、8 位指数、23 位尾数

2 DOUBLE：1 位符号、11 位指数、52 位尾数

3 DOUBLE-EXTEND：1 位符号、15 位指数、64 位尾数

浮点数指数部分存储格式

IEEE754 浮点数指数部分在存储时偏移可能值的一半，实际的指数是从存储指数中减去此偏差获得的。如果存储指数小于此偏差，则它实际上是负指数。指数偏移量如下：

1 FLOAT（8 位指数）：127

2 DOUBLE（11 位指数）：1023

3 DOUBLE-EXTEND（15 位指数）：16383

浮点数精度

IEEE754 浮点数表示精度如下：

格式	最小 非规格化数	最小规格化数	最大有限数	有效 数字
单精度	1.4^{-45}	1.17549435^{-38}	3.40282347^{38}	约 6 位
双精度	4.9^{-324}	$2.2250738585072014^{-308}$	$1.79769313486231 57^{308}$	约 15 位
扩展双精度	$\leq 3.6^{-4951}$	$\leq 3.4^{-4932}$	$\geq 1.2^{4932}$	

浮点数特例

IEEE754 浮点数表示特例：

1 +0： 符号位、指数、尾数全为 0

2 -0： 符号位为 1、指数和尾数全 0

3 +INF： 符号位为 0、指数全 1、尾数全 0

4 –INF： 符号位为 1、指数全 1、尾数全 0

5 NAN： 指数全 1、尾数不全为 0

注

有关 IEEE754 细节请参考 IEEE754 标准文档。宏语言支持 IEEE754 标准中单精度和双精度数据类型，不支持扩展双精度。

宏程序支持的实数分类

记符号位为 s，指数为 e，尾数为 m，则

1 FLOAT： s、e、m 分别占用 1 位、8 位、23 位

2 DOUBLE： s、e、m 分别占用 1 位、11 位、52 位

- **NAN**

e 全 1，m 不全 0，表示 NAN（Not A Number，非数字）

- **±INF**

1 e 全 1，m 全 0，s=0 表示+INF（正无穷大）

2 e 全 1，m 全 0，s=1 表示-INF（负无穷大）

- **规格化数**

e 即不全为 0，也不全为 1，称为规格化数，它们的值为：

1 FLOAT：$(-1)^s \times 2^{e-127} \times (1+\frac{m}{2^{23}})$

2 DOUBLE：$(-1)^s \times 2^{e-1023} \times (1+\frac{m}{2^{52}})$

- **非规格化数**

e 全 0，m 不全 0，称为非规格化数，它们的值为：

1 FLOAT：$(-1)^s \times 2^{-126} \times \frac{m}{2^{23}}$

2 DOUBLE：$(-1)^s \times 2^{-1023} \times \frac{m}{2^{52}}$

- **±0**

e 全 0，m 全 0，s=0 表示+0

e 全 0，m 全 0，s=1 表示-0

注

1 在进行比较时，$+0$等于-0

2 由于十进制数的二进制表示形式可能不精确等原因，浮点运算可能丢失精度

12.12 常见问题解释

问题 1

程序

#101=TANF[-64*PI] ; //参数 DEG（P4100.6）=0

结果

#101=-5.5950577e-006

分析

结果是正确的。理论上，应该是#101=0.0，但由于 PI 本身是一个近似值（约等于 3.1415926535897932），同时计算-64*PI 时误差被放大，因此计算结果会产生一定的误差，得到一个接近于 0 的数（-5.5950577e-006）。TANF 是单精度函数，这在误差允许范围内。

问题 2

程序

#118=ATAN[-1/16] ;

结果

#118=0.000000000000000

分析

结果是正确的。-1/16 是整数运算，结果为 0（整数与整数四则运算结果为整数）。应该修改程序为#118=ATAN[-1/16.0]或#118=ATAN[-1.0/16]。

问题 3

程序

#100=5.1f + 8 - 2.0 ;

结果

#100=1.110000038146973e+001

分析

结果是正确的。理论值为 1.11，这里误差的主要原因是浮点数 5.1f 在存储单元中无法精确表示，产生了微小的误差，并在计算过程中累积下来，最后在计算[DOUBLE][5.1f + 8]时表现出来（由于 2.0 为双精度，而 5.1f+8 结果为单精度数据，系统自动进行隐式类型转换）。

问题 4

程序

#103=FPCLASSIFY[EXP2[-149]] ;

结果

#103=4

分析

结果是正确的。注意这里用的是双精度函数 FPCLASSIFY 而不是单精度函数 FPCLASSIFYF。

问题 5

程序

#103= FPCLASSIFY[EXP2F[-149]] ;

结果

#103=2

分析

结果是正确的。注意这里用的是单精度函数 EXP2F，EXP2F[-149]计算结果为 0。注意单精度的数据范围。

问题 6

程序

#103= 5/2 ;

结果

#103=2

分析

结果是正确的。整数除于整数结果还是整数。

问题 7

程序

#1=41;

G[#1] X20 Z10;

结果

报警：非法 G 代码

分析

使用 G/M/S/T 指令调用宏变量地址时，赋值数据必须为 BCD 码，使用时可提前进行数制转换，可将程序更改为：G[BCD[#1]]，指令等价于 G41。或使用参数“BADR”将 BADR 设为 1，则使用 G/M/S/T 通过变量赋值时不需要转为 BCD 码。