Softlink
Powered By Googoltech

iDEABOX-E

智能控制器高级编程手册

V1.0

2016.06

www.softlink.cn

版权申明

注意	运动中的机器有危险！使用者有责任在机器中设计有效的出错处理和安全保护机制，上海固高欧辰智能科技有限公司没有义务或责任对由此造成的附带的或相应产生的损失负责。

联系我们

客户服务： 4006 300 321

上海固高欧辰智能科技有限公司

地　　址：上海闵行区东川路 555 号 4 号楼 1 层

电　　话：021-54708386　　54708786

传　　真：021-54708386

电子邮件：info@softlink.cn

网　　址：http://www.softlink.cn

文档版本

版本号	修订日期
1.0	2016 年 07 月 05 日

前言

感谢选用 Softlink 运动控制器

为回报客户，我们将以品质一流的运动控制器、完善的售后服务、高效的技术支持，帮助您建立自己的控制系统。

Softlink 产品的更多信息

上海固高欧辰智能科技有限公司的网址是 http://www.softlink.cn。在我们的网页上可以得到更多关于公司和产品的信息，包括：公司简介、产品介绍、技术支持、产品最新发布等等。

您也可以通过电话（*4006 300 321*）咨询关于公司和产品的更多信息。

技术支持和售后服务

您可以通过以下途径获得我们的技术支持和售后服务：

- 电子邮件： info@softlink.cn；
- 电　　话： 4006 300 321

编程手册的用途

用户通过阅读本手册，能够了解 IDEABOX-E 运动控制器的控制功能，掌握函数的用法，熟悉特定控制功能的编程实现。最终，用户可以根据自己特定的控制系统，编制用户应用程序，实现控制要求。

编程手册的使用对象

本编程手册适用于具有 C 语言编程基础或 Windows 环境下使用动态链接库的基础，同时具有一定运动控制工作经验，对伺服或步进控制的基本结构有一定了解的工程开发人员。

编程手册的主要内容

详细介绍了 IDEABOX-E 运动控制器的控制功能及编程实现。

相关文件

关于 IDEABOX-E 运动控制器调试和安装，请参见随产品配套的《IDEABOX-E 运动控制器用户手册》。

关于 IDEABOX-E 运动控制器配置文件及配置工具，请参见随产品配套的《EtherCAT 配置文件&配置工具 EthercatConfig 使用说明》。

目　　录

第1章 指令列表

提示	本章表格中右侧的数字为“页码”，其中指令右侧的为“**第 7 章指令详细说明**”中的对应页码，其他为章节页码，均可以使用“超级链接”进行索引。 本手册中所有字体为蓝色的指令（如 GT_ArcXYR）均带有超级链接，点击可跳转至指令说明。

表 1-1 指令列表

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

第2章 Softpro 中运动函数库的使用

2.1 Softpro 软件库的使用

在 Softpro 平台下使用运动控制器的 Ethercat 总线必须调用 Ethercat 专用库，方法与使用 CPAC-X00-TPX.lib 方法一致，可同时使用 CPAC-X00-TPX.lib。

Ideabox-E 控制器的库文件名为 CPAC GUC-X00-TPX ECAT.lib。如果需要调用高级功能库，方法与使用 CPAC GUC-X00-TPX ECAT.lib 方法一致，库文件名为 GUC-X00-TPX-Addition 2.2.lib

2.2 Softpro 平台的使用

(1) 启动Softpro.exe，新建一个工程；

(2) 系统自动添加CPAC GUC-X00-TPX.lib

(3) 手动在库文件管理器中添加CPAC GUC-X00-TPX ECAT.lib和CPAC GUC-X00-TPX-Addition 2.2.lib

至此，用户就可以在 Softpro 中调用函数库中的任何函数，开始编写应用程序。

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

第3章 命令返回值及其意义

IDEABOX-E 按照主机发送的指令工作。IDEABOX-E 指令封装在动态链接库中。用户在编写应用程序时，通过调用 IDEABOX-E 中运动控制库 GUC-X00-TPX-Addition 2.2.lib 指令来操纵 IDEABOX-E 的运动控制。

IDEABOX-E 在接收到主机发送的指令时，将执行结果反馈到主机，指示当前指令是否正确执行。指令返回值的定义如表 3-1。

表 3-1 运动控制器指令返回值定义

返回值	意义	处理方法
0	指令执行成功	
1	指令执行错误	1. 检查当前指令的执行条件是否满足
7	指令参数错误	1. 检查当前指令输入参数的取值
2	license 不支持	1. 如果需要此功能，请与生产厂商联系。
-1	主机和运动控制器通讯失败	1. 是否正确安装运动控制器驱动程序 2. 检查运动控制器是否接插牢靠 3. 更换主机 4. 更换控制器
-6	打开控制器失败	1. 是否正确安装运动控制器驱动程序 2. 是否调用了 2 次 GT_Open 指令 3. 其他程序是否已经打开运动控制器
-7	运动控制器没有响应	1. 更换运动控制器

建议在用户程序中，检测每条指令的返回值，以判断指令的执行状态。并建立必要的错误处理机制，保证程序安全可靠地运行。

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

第4章 系统配置

系统配置除使用 Softpro 中的配置工具外，还支持使用 CPAC GUC-X00-TPX-Addition 2.2.lib 中的指令来在程序运行中配置。

提示	本章表格中右侧的数字为“页码”，其中指令右侧的为“**第 7 章指令详细说明**”中的对应页码，其他为章节页码，均可以使用“超级链接”进行索引。 本手册中所有字体为蓝色的指令（如 GT_ArcXYR）均带有超级链接，点击可跳转至指令说明。

4.1 配置信息修改指令

用户除了可以使用上面所述的配置文件的方式实现运动控制器的初始化配置外，还可以使用指令的方式来实现初始化配置。

4.1.1 指令列表

表 4-1 配置信息指令列表

指令	说明	页码
GT_AlarmOff	控制轴驱动报警信号无效	67
GT_AlarmOn	控制轴驱动报警信号有效	67
GT_LmtsOn	控制轴限位信号有效	83
GT_LmtsOff	控制轴限位信号无效	83
GT_ProfileScale	设置控制轴的规划器当量变换值	88
GT_EncScale	设置控制轴的编码器当量变换值	78
GT_SetStopDec	设置平滑停止减速度和急停减速度	95
GT_GetStopDec	读取平滑停止减速度和急停减速度	80
GT_SetStopIo	设置平滑停止和紧急停止数字量输入的信息	95
GT_GpiSns	设置运动控制器数字量输入的电平逻辑	82
GT_SetAdcFilter	设置模拟量输入的滤波器时间参数	92

4.1.2 重点说明

1. 编码器计数方向设置

指令 GT_EncSns()可以修改运动控制器各编码器的计数方向，当指令参数的某个状态位为 1 时，将所对应的控制轴的编码器计数方向取反，指令参数的状态位定义如表 4-2 所示：

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

表 4-2 编码器计数方向设置指令参数定义

状态位	8	7	6	5	4	3	2	1	0
编码器	辅助编码器	Enc8	Enc7	Enc6	Enc5	Enc4	Enc3	Enc2	Enc1

2. 设置限位开关触发电平

运动控制器默认的限位开关为常闭开关，即各轴处于正常工作状态时，其限位开关信号输入为低电平；当限位开关信号输入为高电平时，与其对应轴的限位状态将被触发。如果使用常开开关，需要通过调用指令 GT_LmtSns()改变限位开关触发电平。

指令 GT_LmtSns()的参数设置各轴正负限位开关的触发电平，当该参数的某个状态位为 0 时，表示将对应的限位开光设置为高电平触发，当某个状态位为 1 时，表示将对应的限位开关设置为低电平触发。指令参数和各轴限位的对应关系如下所示：

表 4-3 指令参数和各轴限位对应关系

状态位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
限位开关	轴 8		轴 7		轴 6		轴 5		轴 4		轴 3		轴 2		轴 1	
	—	+	—	+	—	+	—	+	—	+	—	+	—	+	—	+

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

第5章 运动模式

除 GUC-800-TPX 控制器每个轴可以独立工作在点位、Jog、PT、电子齿轮或 Follow 运动模式下外，本章主要介绍 CPAC GUC-X00-TPX-Addition 2.2.lib 实现 PVT 运动模式，以及插补运动。

提示	本章表格中右侧的数字为“页码”，其中指令右侧的为“**第 7 章指令详细说明**”中的对应页码，其他为章节页码，均可以使用“超级链接”进行索引。 本手册中所有字体为蓝色的指令（如 GT_ArcXYR）均带有超级链接，点击可跳转至指令说明。

注意	用户需注意，每一个轴在任一时刻只能处在一种运动模式下。

5.1 插补运动模式

插补运动模式可以实现多轴的协调运动，从而完成一定的运动轨迹。该插补运动模式具有以下一些功能，可以实现直线插补和圆弧插补；可以同时有两个坐标系进行插补运动；每个坐标系含有两个缓存区，可以实现缓存区暂停、恢复等功能；具有缓存区延时和缓存区数字量输出的功能；具有前瞻预处理功能，能够实现小线段高速平滑的连续轨迹运动。

5.1.1 指令列表

表 5-1 设置插补运动指令列表

指令	说明	页码
GT_SetCrdPrm	设置坐标系参数，确立坐标系映射，建立坐标系	93
GT_GetCrdPrm	查询坐标系参数	79
GT_CrdData	向插补缓存区增加插补数据	76
GT_LnXY	缓存区指令，两维直线插补	84
GT_LnXYZ	缓存区指令，三维直线插补	85
GT_LnXYZA	缓存区指令，四维直线插补	85
GT_LnXYG0	缓存区指令，两维直线插补(终点速度始终为 0)	84
GT_LnXYZG0	缓存区指令，三维直线插补(终点速度始终为 0)	86
GT_LnXYZAG0	缓存区指令，四维直线插补(终点速度始终为 0)	86
GT_ArcXYR	缓存区指令，XY 平面圆弧插补(以终点位置和半径为输入参数)	68
GT_ArcXYC	缓存区指令，XY 平面圆弧插补(以终点位置和圆心位置为输入参数)	67

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

GT_ArcYZR	缓存区指令，YZ 平面圆弧插补(以终点位置和半径为输入参数)	69
GT_ArcYZC	缓存区指令，YZ 平面圆弧插补(以终点位置和圆心位置为输入参数)	69
GT_ArcZXR	缓存区指令，ZX 平面圆弧插补(以终点位置和半径为输入参数)	71
GT_ArcZXC	缓存区指令，ZX 平面圆弧插补(以终点位置和圆心位置为输入参数)	70
GT_BufIO	缓存区指令，缓存区内数字量 IO 输出设置指令	73
GT_BufDelay	缓存区指令，缓存区内延时设置指令	72
GT_BufDA	缓存区指令，缓存区内输出 DA 值	72
GT_BufLmtsOn	缓存区指令，缓存区内有效限位开关	74
GT_BufLmtsOff	缓存区指令，缓存区内无效限位开关	73
GT_BufSetStopIo	缓存区指令，缓存区内设置 axis 的停止 IO 信息	75
GT_BufMove	缓存区指令，实现刀向跟随功能，启动某个轴点位运动	74
GT_BufGear	缓存区指令，实现刀向跟随功能，启动某个轴跟随运动	72
GT_SetUserSegNum	缓存区指令，设置自定义插补段段号	95
GT_GetUserSegNum	读取自定义插补段段号	81
GT_GetRemainderSegNum	读取未完成的插补段段数	80
GT_CrdSpace	查询插补缓存区剩余空间	76
GT_CrdClear	清除插补缓存区内的插补数据	76
GT_CrdStart	启动插补运动	77
GT_CrdStatus	查询插补运动坐标系状态	77
GT_SetOverride	设置插补运动目标合成速度倍率	94
GT_SetCrdStopDec	设置插补运动平滑停止、急停合成加速度	94
GT_GetCrdStopDec	查询插补运动平滑停止、急停合成加速度	79
GT_GetCrdPos	查询该坐标系的当前坐标位置值	78
GT_GetCrdVel	查询该坐标系的合成速度值	79
GT_InitLookAhead	初始化插补前瞻缓存区	83

5.1.2 重点说明

1. 建立坐标系

运动控制器初始状态下，所有的规划轴都处于单轴运动模式下，两个坐标系也是无效的。所以，当需要进行插补运动时，首先需要建立坐标系，将规划轴映射到相应的坐标系中。每个坐标系最多支持四维(X-Y-Z-A)，用户根据自己的需求，也可以利用二维(X-Y)、三维(X-Y-Z)坐标系描述运动轨迹。

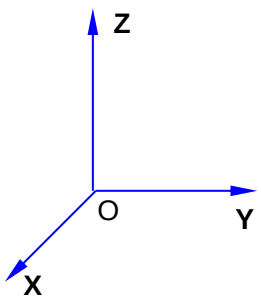

图 5-1 右手坐标系

用户通过调用 GT_SetCrdPrm 指令将在坐标系内描述的运动通过映射关系映射到相应的规划轴上。运动控制器根据坐标映射关系，控制各轴运动，实现要求的运动轨迹。调用 GT_SetCrdPrm 指令时，所映射的各规划轴必须处于静止状态。

例程 5-1 建立坐标系

```
…… …… ……
    Rtn:INT;
    CrdPrm: TCrdPrm;                        (* 定义坐标系结构体变量*)
    First:BOOL:=TRUE;
-------------------------------------------------------------------------------
IF First THEN
    SysMemSet(ADR(crdPrm),0,sizeof(crdPrm));        (* 将变量初始化为全 0 *)
    crdPrm.dimension:=2;                    (* 坐标系为二维坐标系*)
    crdPrm.synVelMax:=500;                  (* 最大合成速度：500pulse/ms*)
    crdPrm.synAccMax:=1;                    (* 最大加速度：1pulse/ms^2*)
    crdPrm.evenTime := 50;                  (* 平滑时间：50ms*)
    crdPrm.profile[0] := 1;                 (* 规划器 1 对应到 X 轴*)
    crdPrm.profile[1] := 2;                 (* 规划器 2 对应到 Y 轴*)
    crdPrm.setOriginFlag := 1;              (* 需要明确指定坐标系原点的规划位置*)
    crdPrm.originPos[0] := 100;
    crdPrm.originPos[1] := 100;
    rtn := GT_SetCrdPrm(1,ADR(crdPrm));         (* 建立 1 号坐标系，设置坐标系参数*)
    First:=FALSE;
END_IF
…… …… ……
```

例程说明：

- dimension：表示所建立的坐标系的维数，取值范围为[1,4]，该例程中所建立的坐标系是二维，即 X-Y 坐标系。
- synVelMax：表示该坐标系所能承受的最大合成速度，如果用户在输入插补段的时候所设置的目标速度大于了该速度，则将会被限制为该速度。
- synAccMax：表示该坐标系所能承受的最大合成加速度，如果用户在输入插补段的时候所设置的加速度大于了该加速度，则将会被限制为该加速度。
- evenTime：每个插补段的最小匀速时间。当用户设置的插补段比较短时，而该插补段的目标速度又设置的比较大，则会造成合成速度的曲线如图 5-2 所示，只有加速段和减速段，形成一个速度尖角，加速度在尖角处瞬间由正值变为了负值，造成较大的冲击；设置了 evenTime 之后，可以减小目标速度，使速度曲线如图 5-3 所示，减小了加速度突变的冲击。

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

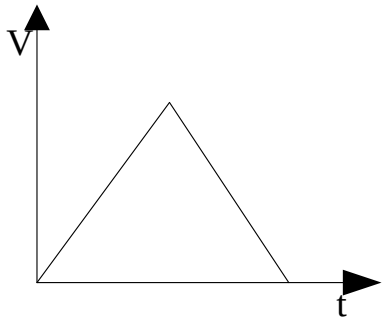

图 5-2 evenTime=0

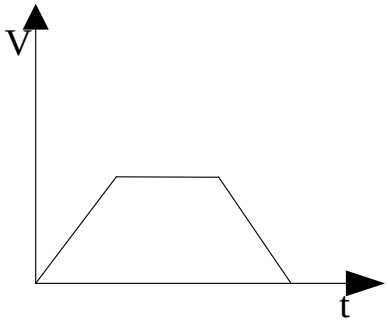

图 5-3 evenTime>0

- profile[x]：规划轴与坐标轴之间的对应关系。Profile[0..7]对应规划轴 1~8，如果规划轴没有对应到该坐标系，则 profile[x]的值为 0；如果对应到了 X 轴，则 profile[x]为 1，Y 轴对应为 2，Z 轴对应为 3，A 轴对应为 4。不允许多个规划轴映射到相同坐标系的相同坐标轴，也不允许把相同规划轴对应到不同的坐标系，否则该指令将会返回错误值。
- setOriginFlag：表示是否需要指定坐标系原点的规划位置值，该参数可以方便用户建立区别于机床坐标系的加工坐标系。如果该参数为 0，则加工坐标系的原点在当前的规划位置上，如果该参数为 1，则加工坐标系的原点在用户指定的规划位置上，通过 originPos 来指定。
- originPos[x]：加工坐标系原点的规划位置值，即相对于机床坐标系的偏移量。建立的加工坐标系如图 5-4 所示。

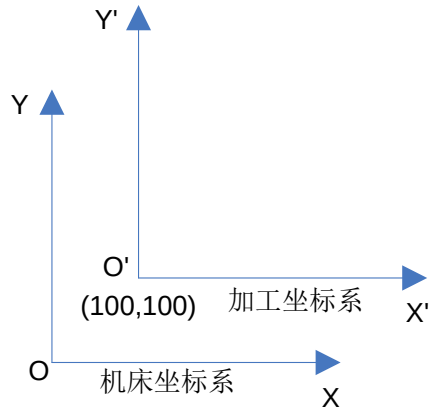

图 5-4 加工坐标系偏移量示意图

2. 坐标系运动

坐标系运动采用缓存区运动方式，即用户需要向插补缓存区中传递插补数据，然后，启动插补运动，运动控制器则会依次执行用户所传递的插补数据，直到所有的插补数据全部运动完成。

每个坐标系包含两个缓存区(FIFO)：FIFO0 和 FIFO1，其中 FIFO0 为主要运动 FIFO，FIFO1 为辅助运动 FIFO，每个 FIFO 都含有 4096 段插补数据的空间。FIFO 支持动态管理的方式，即插补数据运动完成之后，其所占用的缓存区空间将会被释放，用户可以继续传递新的插补数据，通过这种方式，就可以支持大于 4096 段的用户插补数据。

(1) 直线插补

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

例程 5-2 坐标系运动的直线插补

```
…… …… ……
    Rtn:INT;
    Run:INT;                                    (* 定义坐标系运动状态查询变量*)
    Segment:DINT;                               (* 定义坐标系运动完成段查询变量*)
    First:BOOL:=TRUE;
-----------------------------------------------------------------------------------------------
IF First THEN
    rtn := GT_AxisOn(1);
    rtn := GT_AxisOn(2);

    rtn := GT_CrdClear(1,0);                    (* 清除坐标系 1 的 FIFO0 中的数据*)
    (* 第一段插补数据*)
    rtn := GT_LnXY(1,200000,0,100,0.1,0,0);     (*向坐标系 1 的 FIFO0 传递直线插补数据*)
                                                (* 该插补段的终点坐标(200000,0)*)
                                                (* 该插补段的目标速度：100pulse/ms*)
                                                (* 插补段的加速度：0.1pulse/ms^2*)
                                                (* 终点速度为 0*)
    (*第二段插补数据*)
    rtn = GT_LnXY(1,100000,173205,100,0.1,0,0);
    (* 缓存区数字量输出*)
    rtn = GT_BufIO(1,MC_GPO,16#ffff,16#55,0);   (* 数字量输出类型：通用输出*)
                                                (* bit0~bit15 全部都输出*)
                                                (* 输出的数值为 0x55*)
    (* 第三段插补数据*)
    rtn := GT_LnXY(1,-100000,173205,100,0.1,0,0);
    (* 缓存区数字量输出*)
    rtn := GT_BufIO(1,MC_GPO,16#ffff,16#aa,0);
    (* 第四段插补数据*)
    rtn := GT_LnXY(1,-200000,0,100,0.1,0,0);
    (* 缓存区延时指令*)
    rtn := GT_BufDelay(1,400,0);                (* 该处延时 400ms*)
    (* 第五段插补数据*)
    rtn := GT_LnXY(1,-100000,-173205,100,0.1,0,0);
    (* 缓存区数字量输出*)
    rtn := GT_BufIO(1,MC_GPO,16#ffff,16#55,0);
    (* 缓存区延时指令*)
    rtn := GT_BufDelay(1,100,0);
    (* 第六段插补数据*)
    rtn := GT_LnXY(1,100000,-173205,100,0.1,0,0);
    (* 第七段插补数据*)
    rtn := GT_LnXY(1,200000,0,100,0.1,0,0);
    rtn := GT_CrdSpace(1,ADR(space),0);         (*查询坐标系 1 的 FIFO0 所剩余的空间*)
    rtn := GT_CrdStart(1,0);                    (*启动坐标系 1 的 FIFO0 的插补运动*)
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
        First:=FALSE;
    END_IF

        (* 坐标系在运动,查询到的 run 的值为 1*)
        rtn := GT_CrdStatus(1,ADR(run),ADR(segment),0);
    …… …… ……
```

例程说明：

通过 GT_LnXY 指令向插补缓存区传递数据，在该指令中包含了终点坐标、加速度、目标速度，还可以调用 GT_BufIO 指令在缓存区中进行数字量输出，调用 GT_BufDelay 指令在缓存区中进行延时操作。该例程共向插补缓存区传递了 7 段运动数据，例程的运行结果如图 5-5 所示，是一个六边形。其中，在运动过程中进行了缓存区延时和数字量输出的操作。

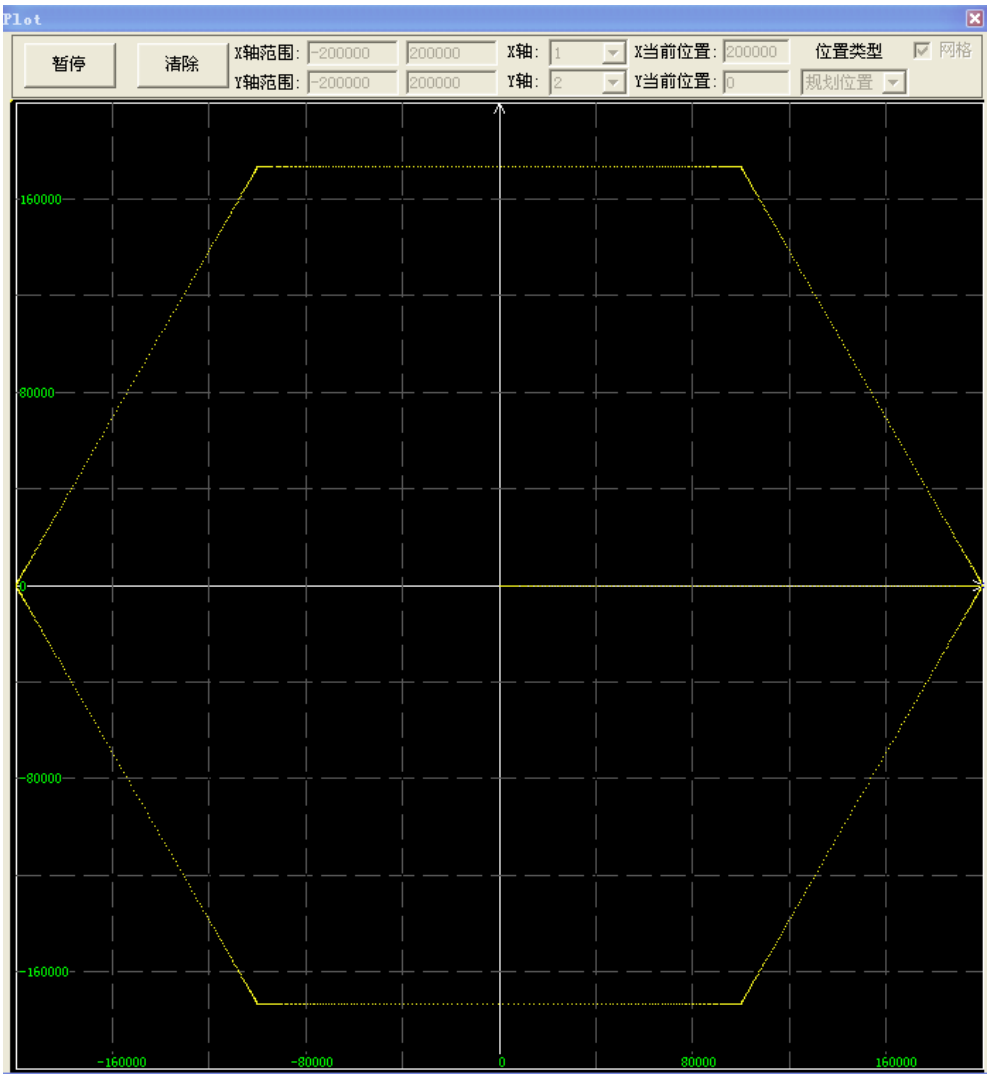

图 5-5 直线插补例程运动结果

GT_CrdStatus 指令可以用来查询坐标系指定 FIFO 的运动状态(运动或静止)，以及当前已经完成的插补运动的段数，该段数从建立坐标系之后的第一个 GT_CrdStart 的调用之后开始累加，直到销毁坐标系或者调用 GT_CrdClear 指令时才被清零。

(2) 圆弧插补

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

例程 5-3 坐标系运动的直圆弧插补

```
        Rtn:INT;
        Run:INT;                                    (* 定义坐标系运动状态查询变量*)
        Segment:DINT;                               (* 定义坐标系运动完成段查询变量*)
        First:BOOL:=TRUE;
    ------------------------------------------------------------------------------------------------
    IF First THEN
        rtn := GT_AxisOn(1);
        rtn := GT_AxisOn(2);

        rtn := GT_CrdClear(1,0);                    (* 清除坐标系 1 的 FIFO0 中的数据*)

        (* 直线插补数据*)
        rtn := GT_LnXY(1,200000,0,100,0.1,0,0);

        (* 圆弧插补数据*)
        rtn := GT_ArcXYC(1,200000,0,-100000,0,0,100,0.1, 0,0);
                                                    (* 使用圆心描述方法描述一个整圆*)
                                                    (* 圆心坐标(100000,0)*)
                                                    (* 终点坐标与起点坐标重合(200000,0)*)
                                                    (* 顺时针圆弧*)
                                                    (* 该插补段的目标速度：100pulse/ms*)
                                                    (* 插补段的加速度：0.1pulse/ms^2*)
                                                    (* 终点速度为 0*)
        (* 圆弧插补数据*)
        rtn := GT_ArcXYR(1,0,200000,200000,1,100,0.1,0,0);
                                                    (* 使用半径描述方法描述一个 1/4 圆弧*)
                                                    (* 终点坐标为：(0,200000)*)
                                                    (* 半径：200000*)
                                                    (* 逆时针圆弧*)
        rtn := GT_LnXY(1,0,0,100,0.1,0,0);          (* 回到原点位置*)
        rtn := GT_CrdSpace(1, ADR(space),0);        (* 查询坐标系 1 的 FIFO0 所剩余的空间*)
        rtn := GT_CrdStart(1,0);                    (* 启动坐标系 1 的 FIFO0 的插补运动*)
        First:=FALSE;
    END_IF

        rtn := GT_CrdStatus(1,ADR(run),ADR(segment),0);
        (* 查询坐标系 1 的 FIFO 的插补运动状态*)
        (* 坐标系在运动,查询到的 run 的值为 1*)
    IF run=1 THEN
    …… …… ……
```

圆弧插补例程的运行结果如图 5-6 所示。

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

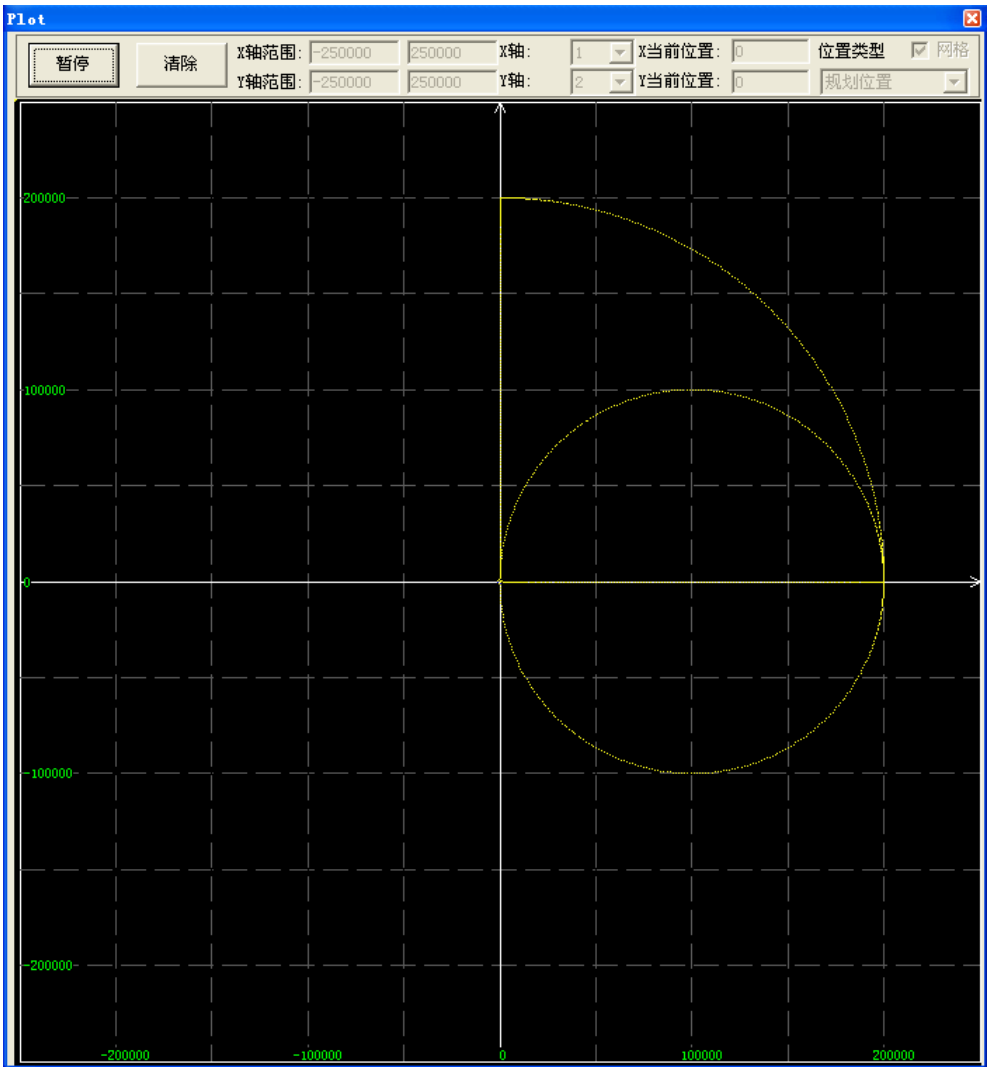

图 5-6 圆弧插补例程运动结果

该控制器支持在 XY 平面、YZ 平面和 ZX 平面的圆弧插补，其中圆弧插补的旋转方向按照右手螺旋定则定义为：从坐标平面的“上方”(即垂直于坐标平面的第三个轴的正方向)看，来确定逆时针方向和顺时针方向。可以这样简单记忆：将右手拇指前伸，其余四指握拳，拇指指向第三个轴的正方向，其余四指的方向即为逆时针方向。映射坐标系为二维坐标系(X-Y)时，XOY 坐标平面内的圆弧插补逆时针方向同样定义。

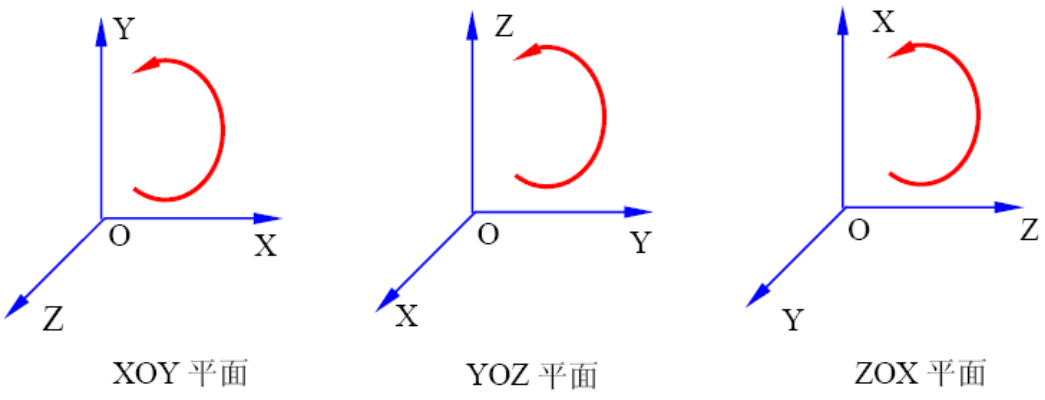

图 5-7 圆弧插补逆时针方向

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

圆弧插补有两种描述方法：半径描述方法和圆心坐标描述方法，用户可以根据加工数据选择合适的描述方法来编程，所使用的描述方法遵循 G 代码的编程标准。

1) 半径描述方法

半径描述方法，即调用指令 GT_ArcXYR、GT_ArcYZR、GT_ArcZXR 对圆弧进行描述，用户需要输入圆弧终点坐标、圆弧半径、圆弧的旋转方向、速度和加速度等。其中参数半径可为正值，也可为负值，其绝对值为圆弧的半径，正值表示圆弧的旋转角度≤180°，负值表示圆弧的旋转角度＞180°，如图 5-8 所示，半径描述方法无法描述 360°的整圆。

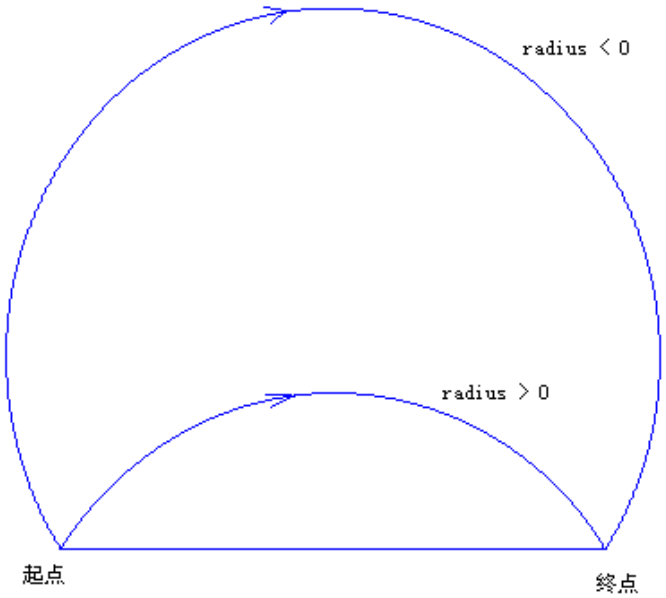

图 5-8 半径取正值/负值圆弧插补示意图

2) 圆心描述方法

圆心描述方法，即调用指令 GT_ArcXYC、GT_ArcYZC、GT_ArcZXC 对圆弧进行描述，用户需要输入圆弧终点坐标、圆心相对于起点坐标的相对位置值、圆弧的旋转方向、速度和加速度等。其中，圆心位置值参数的定义如图 5-9 所示：

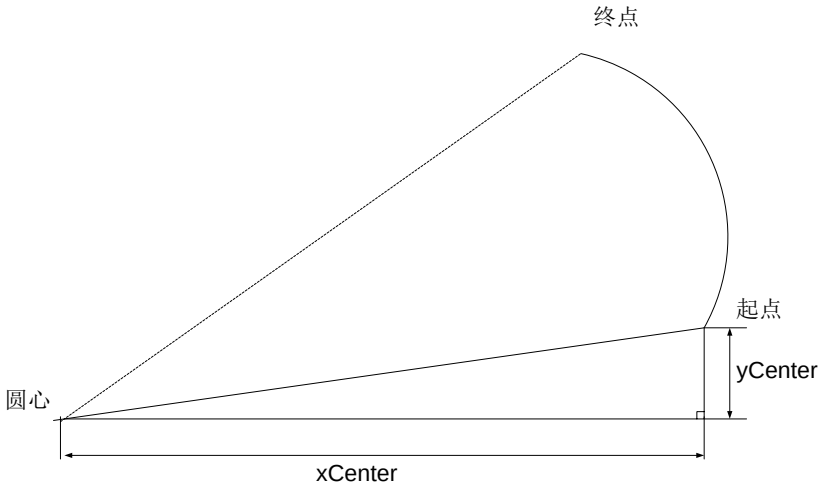

图 5-9 圆心表示方法示意图

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

圆心参数为相对于起点位置的增量值，带正负号，如果起点的坐标为(xStart,yStart)，用户设置的圆心参数为(xCenter,yCenter)，则圆心坐标值为(xStart+xCenter,yStart+yCenter)。用户设置的起点坐标和终点坐标重合时，则表示将要进行一个整圆的运动。

用户应该保证圆弧描述指令可以正确描述一段圆弧，如果用户所设置的参数不能生成一段正确的圆弧，指令会返回 7(参数错误)。

3. 前瞻预处理

小线段插补加工的特点：为保证刀具与加工工件接触面的光顺，尽量保持轨迹运动过程中切向速度的恒定，同时又必须保证一定的轨迹加工精度。

观察图 5-10，可以了解该图形中每条线段终点处都是拐点(轨迹特征发生明显改变的点)，且必须降速，但是否应该降到 0，需要根据线段长度、速度、加速度以及拐点速度变化限值等与加工工艺相关的参数来计算出各段的终点速度。

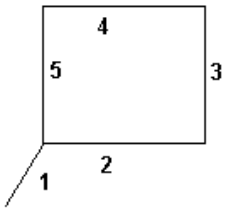

图 5-10 X-Y 平面多段轨迹图形

观察图 5-11，可以了解对于以小线段拟合曲线轨迹的线段组合，应在加工过程中尽量保持切向速度的恒定，但又必须保证在拐点(第 8 点)处将速度降到一个合理的值(合理的终点速度)，以保证加工执行机构(机械本体和电机)能够承受由于拐点处轨迹特征发生变化而带来的速度变化量。

图 5-11 X -Y 平面小线段轨迹图形

为了解决高速和高精度这对矛盾，运动控制器对运动过程中的速度规划采用基于前瞻预处理的处理方式。

用户可根据本身机床的工艺特征参数(脉冲当量、目标速度、最大加速度、允许拐弯时间等)，调用运动控制器提供的前瞻预处理模块，给出每段的终点速度，运动控制器则严格按照每段终点速度进行加减速控制。运动控制器将这套实现速度规划预处理功能的指令称为前瞻预处理(又称 LookAhead)指令。

从图 5-12 可以直观地了解，使用前瞻预处理功能模块来规划速度，在小线段加工过程中的对速度的显著提升：

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

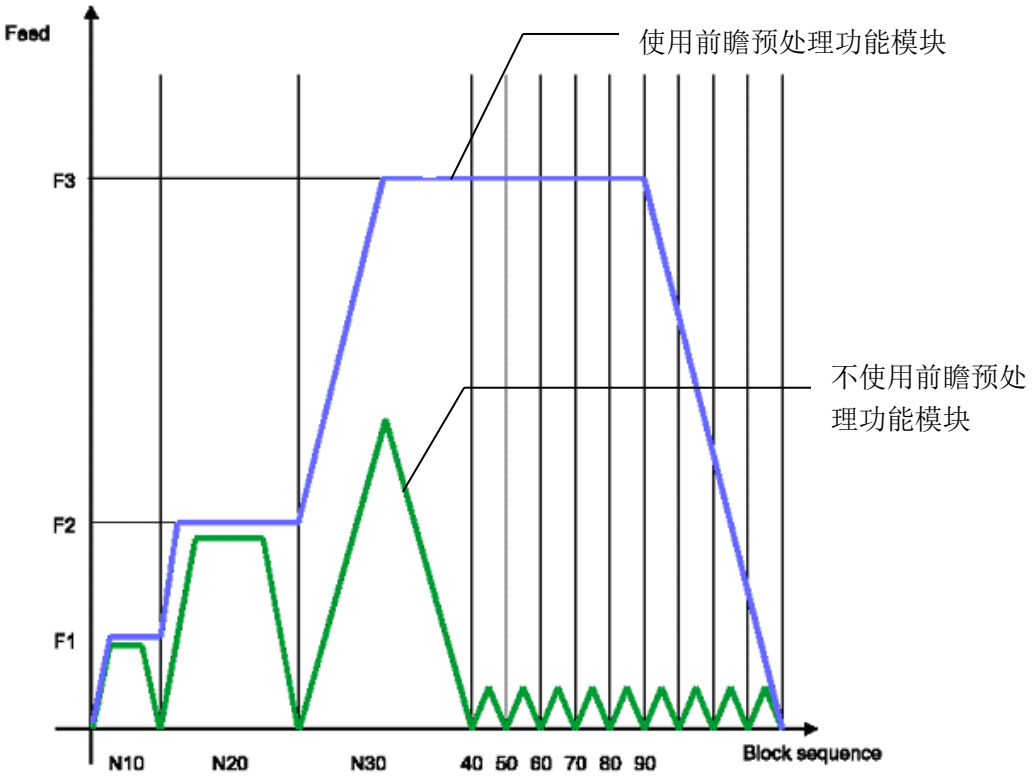

图 5-12 使用和不使用前瞻预处理功能模块的速度曲线对比图

例程 5-4 使用前瞻预处理

```
……..
    Rtn:INT;
    I:INT;
    CrdDataSend: TCrdData;
    CrdData:ARRAY[0..199] OF TCrdData;                    (* 定义前瞻缓存区内存区*)
    PosTest:ARRAY[0..1] OF DINT;
    First:BOOL:=TRUE;
IF First THEN
    rtn := GT_AxisOn(1);
    rtn := GT_AxisOn(2);

    rtn := GT_InitLookAhead(1,0,5,1,200,crdData); (* 初始化坐标系 1 的 FIFO0 的前瞻模块*)
    (* 压插补数据：小线段加工*)
    posTest[0] := 0;
    posTest[1] := 0;
FOR I:=1 TO 300 BY 1 DO
        rtn := GT_LnXY(1,8000+posTest[0],9000+posTest[1],100,0.8,0,0);
        posTest[0] := posTest[0]+1600;
        posTest[1] := posTest[1]+1852;
END_FOR
    rtn := GT_CrdData(1,0,0);                    (* 将前瞻缓存区中的数据压入控制器*)
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
            rtn := GT_CrdStart(1,0);
             First:=FALSE;
        END_IF
        …… …… ……
```

例程说明：

- 拐弯时间(T)：GT_InitLookAhead 指令的第三个参数，单位：ms。T 的经验范围是：1ms~10ms，T 越大，计算出来的终点速度越大，但却降低了加工精度；反之，提高了加工的精度，但计算出的终点速度偏低。因此要合理选择 T 值。

- 最大加速度(accMax)：GT_InitLookAhead 指令的第四个参数，单位：pulse/(ms*ms)。系统能承受的最大加速度，根据不同的机械系统和电机驱动器取值不同。

- 前瞻缓存区大小和前瞻缓存区内存区指针：该前瞻模块采用用户提供前瞻缓存区内存区的方式，因此，用户可以根据自己的需要以及计算机的条件定义合适的缓存区大小，前瞻缓存区越大，占用的内存区就越大。用户需要先定义一个插补数据数组变量，申请一定的内存区，然后通过 GT_InitLookAhead 指令把内存区的指针传递给运动控制器的前瞻模块，在进行前瞻预处理的过程中，用户不能再对该内存区进行任何操作，否则将会破坏前瞻缓存区中的数据，造成数据的错误。

  当前瞻缓存区的段数不为 0 时，用户调用缓存区指令传递的插补数据先进入前瞻缓存区，当前瞻缓存区放满之后，如果再有新的数据传入，最先进入前瞻缓存区的数据，则会进入插补缓存区。

  如果用户所有的插补数据已经输入完毕，前瞻缓存区中还有数据没有进入插补缓存区，这时，需要调用 GT_CrdData(1,0,0)，运动控制器会将前瞻缓存区的数据依次传递给插补缓存区，直到前瞻缓存区被清空为止。

  在数据量比较大的时候，用户需要配合 GT_CrdSpace 指令查询插补缓存区的剩余空间，在有空间的时候再调用缓存区指令传递数据，如果插补缓存区已满，调用缓存区指令将会返回错误，说明该段插补数据没有输入成功，需要再次输入该段插补数据。

如果不使用前瞻预处理功能，则运动控制器不会对插补段的终点速度和目标速度进行优化，运动控制器将严格按照用户指定的目标速度和终点速度进行速度规划。如果用户调用 GT_LnXYG0、GT_LnXYZG0 和 GT_LnXYZAG0 指令，则该运动指令将会完成一个完整的加减速过程，即每段运动的合成速度都是从 0 开始，结束的时候也是 0。如果调用其他插补运动指令(包括直线插补和圆弧插补指令)，则用户可以指定插补段的目标速度和终点速度，运动控制器会按照用户指定的目标速度和终点速度进行速度规划。

使用前瞻预处理功能之后，控制器会根据用户设置的参数将每段的终点速度设置为一个合理的值，该值不一定为 0，如果用户的工艺要求某段插补数据的终点速度必须为 0，则需要调用 GT_LnXYG0、GT_LnXYZG0 或者 GT_LnXYZAG0，该指令会将该直线插补段的终点速度设置为 0。如果调用其他插补运动指令(包括直线插补和圆弧插补指令)，则用户设置的终点速度将会无效，实际的终点速度是前瞻预处理模块根据用户设置的前瞻预处理参数和运动轨迹计算出来的一个合理的终点速度。另外，如果某段插补运动数据与下段插补运动数据之间存在缓存区延时，则该段插补运动的终点速度会被设置为 0。

前瞻预处理功能只支持 3 轴或者 3 轴以下的插补运动，如果建立的坐标系大于 3 轴，则在使用前瞻预处理功能时，会返回错误值，调用缓存区指令时，会返回 7(参数错误)。

如果没有进行前瞻预处理的合成速度曲线如图图 5-13 所示，合成速度在不停的变化。进行前瞻预处理的合成速度曲线如图 5-14 所示，由于例程中的插补运动都是在同一条直线上，所以速度可以一直维持

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

在目标速度，大大提高了加工效率。

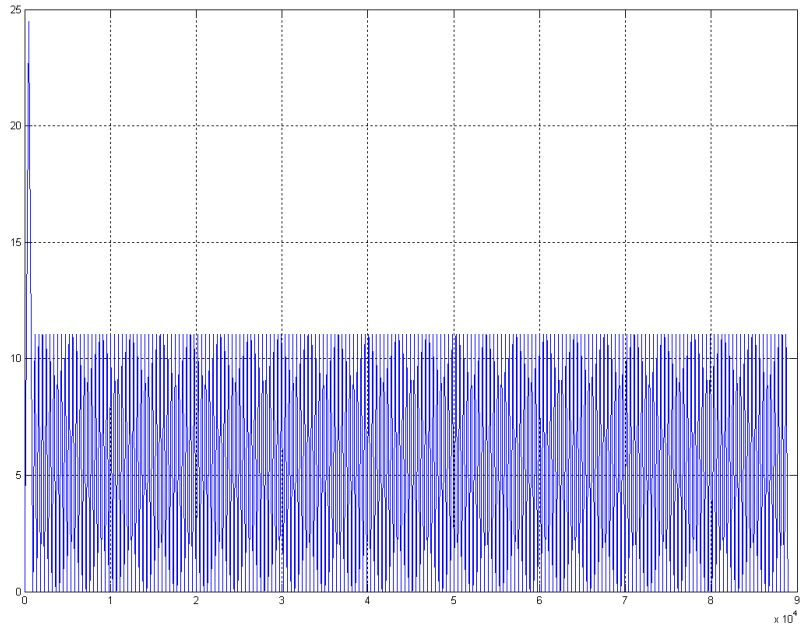

图 5-13 没有进行前瞻预处理的合成速度曲线

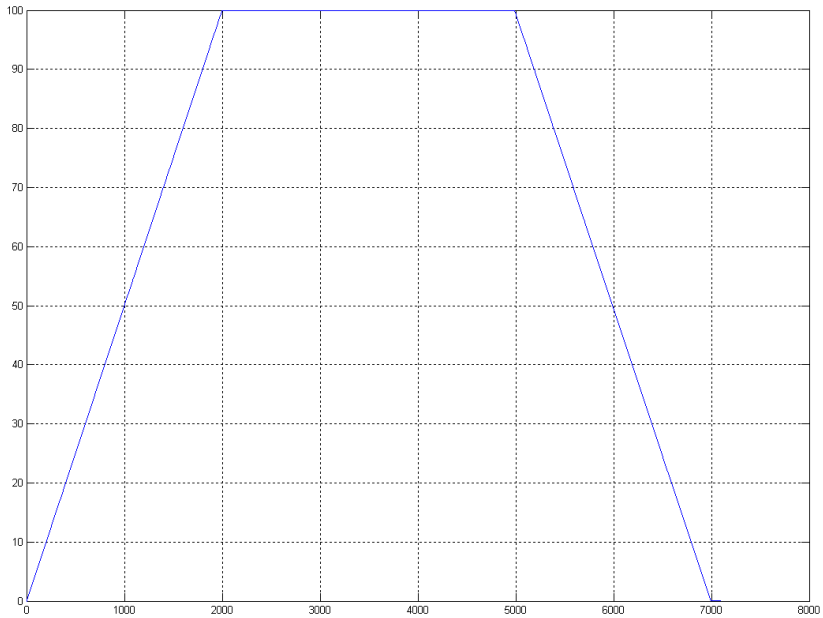

图 5-14 进行了前瞻预处理后的合成速度曲线

4. 缓存区暂停、恢复

在缓存区插补的过程中，用户可能会需要暂停加工，查看加工效果，或者在暂停之后进行其他操作，如换刀等，该运动控制器支持上述操作过程。为了实现上述的操作过程，每个坐标系提供了两个插补缓存区：FIFO0 和 FIFO1。两个缓存区都有 4096 段插补缓存区，并且可以独立设置各自的前瞻预处理缓存区。

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

其中，FIFO0 是主运动 FIFO，用户的主体插补运动的插补数据应该放在 FIFO0 中。FIFO0 的插补运动可以被中断，中断后可以进行辅助 FIFO1 的插补运动，辅助 FIFO1 的插补运动完成后，FIFO0 可从断点处继续恢复原来的运动。

FIFO1 是辅助运动 FIFO，用户的辅助插补运动的插补数据可以放在 FIFO1 中，FIFO1 的插补数据必须在 FIFO0 的运动停止的情况下才能输入，如果 FIFO0 在运动，向 FIFO1 中传递插补数据，将会提示错误。FIFO1 的插补运动也可以暂停、恢复，但是，在暂停、恢复之间不可以进行 FIFO0 的插补运动，否则，FIFO1 缓存区将会被清空，不可以再恢复 FIFO1 的运动。

在主运动 FIFO0 的运动暂停之后，又进行了 FIFO1 的运动，如果用户希望在 FIFO1 运动结束之后，继续进行 FIFO0 的运动，则用户必须保证，FIFO1 运动结束后，坐标位置值与 FIFO0 停止时的坐标位置值(断点位置)相同，否则，FIFO0 将不接受启动运动指令，调用 GT_CrdStart 指令启动 FIFO0 的运动，将会提示错误。

5. 刀向跟随功能

刀向跟随，就是在插补运动的过程中，部分轴会随着插补运动的合成位移的变化而变化，从而实现在加工过程中，刀具始终处于合适的加工方向的工艺。在本控制器的插补模块中有两条指令来实现该工艺：GT_BufMove和GT_BufGear。GT_BufMove可以在插补运动的过程中插入模态和非模态的点位运动；GT_BufGear可以在插补过程中实现其他轴跟随插补合成位移的运动。

(1) 插补过程中的点位运动

插补过程中的点位运动通过在缓存区中压入 GT_BufMove 指令来实现，该指令的第二个参数是需要进行点位运动的轴号，这里需要注意的是，需要进行点位运动的轴不能是坐标系中的轴；当该轴的运动模式不是点位运动模式而且正在运动时，该指令将不能正常执行。该指令的第三个参数是点位运动的目标位置，该位置值是相对于机床原点的绝对位置。该指令的第四个参数是点位运动的目标速度，该值必须为正值。该指令的第五个参数是点位运动的加速度，该值必须为正值。该指令的第六个参数表示该点位运动是模态的还是非模态的，模态指令的意义是，在进行该点位运动时，后续的插补缓存区中的指令将会被暂停执行，直到该指令执行完毕后，才执行下一条指令；非模态指令的意义是，该指令启动了一个轴的点位运动后，立即取下一条缓存区中的指令执行，不会等待点位运动的结束。使用的具体例程如下：

例程 5-5 插补过程中的点位运动

```
PROGRAM PLC_PRG
VAR
   Enable : BOOL;
   Rtn : INT;
   Run : INT; (*  定义坐标系运动状态查询变量  *)
   Segment : DINT; (*  定义坐标系运动完成段查询变量  *)
END_VAR
-----------------------------------------------------------------------------------------------
IF Enable THEN
   rtn := GT_CrdClear(1,0); (*清除坐标系 1 的 FIFO0 中的数据*)
   rtn := GT_LnXY(1,200000,200000,100,0.1,0,0); (*  直线插补指令  *)
   rtn := GT_BufMove(1,4,50000,30,0.1,0,0); (*  缓存区内的点位运动指令
                                              点位运动的轴号：第 4 轴
                                              点位运动的目标位置：50000 pulse
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
                                                    点位运动的目标速度：100 pulse/ms
                                                    点位运动的目标加速度：0.1 pulse/(ms*ms)
                                                    该点位运动是非模态指令 *)
        rtn := GT_LnXY(1,200000,0,100,0.1,0,0); (* 直线插补指令*)
        rtn := GT_BufMove(1,4,100000,30,0.1,1,0); (* 缓存区内的点位运动指令
                                                    点位运动的轴号：第 4 轴
                                                    点位运动的目标位置：100000 pulse
                                                    点位运动的目标速度：100 pulse/ms
                                                    点位运动的目标加速度：0.1 pulse/(ms*ms)
                                                    该点位运动是模态指令 *)
        rtn := GT_ArcXYC(1,-200000,0,-200000,0,0,100,0.1,0,0); (* 圆弧插补指令 *)
        Enable := FALSE;
    END_IF
    … … …
```

例程的运行结果如图 5-15 所示：

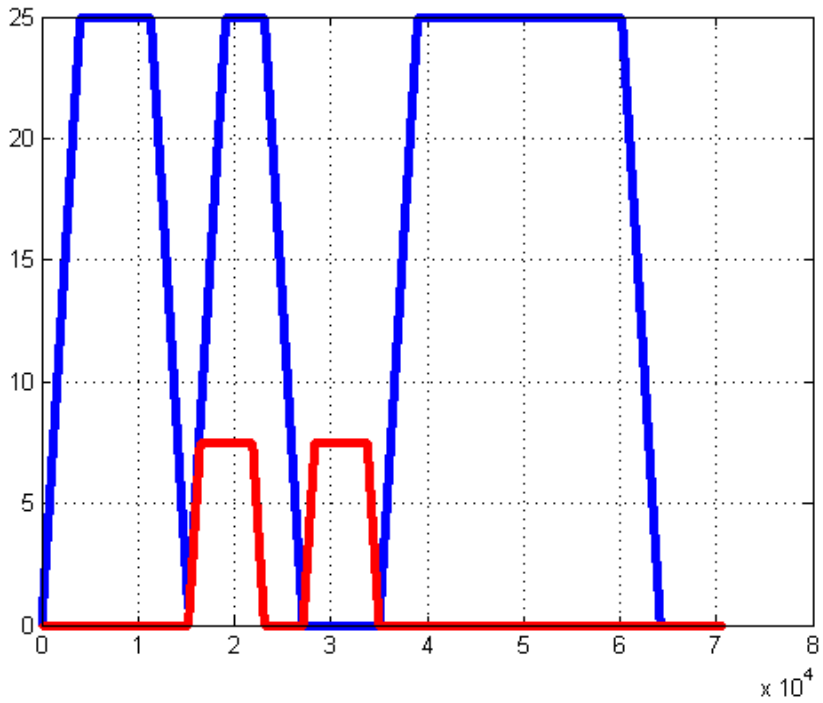

图 5-15 插补过程中的点位运动曲线

其中蓝色为插补运动的合成速度，红色为点位运动轴的速度值，可以看出第一个点位运动是非模态指令，与插补运动同时运动，而第二个点位运动是模态指令，会阻塞插补运动，等点位运动结束之后，再进行插补运动。 在使用插补缓存区中的点位运动功能时，需要注意以下内容：

a) 点位运动的目标位置是相对于机床原点的绝对位置。

b) 如果在上一次的缓存区点位运动没有运动完成，又发送了新的点位运动，则会按照新的点位运动指令进行规划，即可以在插补缓存区中修改点位运动的目标位置和目标速度。

c) 如果在运动过程中停止插补缓存区的运动，则点位运动将不会停止，如果需要停止点位运动，则

需要调用GT_Stop()指令停止响应轴的运动。恢复缓存区运动时，用户需自行保证之前点位运动的轴在合适的位置上。

d) 当在模态点位运动的过程中，进行点位运动的轴由于触发限位等异常原因停止时，插补缓存区将不会再继续运行，此时用户需排查异常情况，重新设置相应参数，使系统正常后才可以工作。

(2) 插补过程中的跟随运动

插补过程中的跟随运动通过在缓存区中压入 GT_BufGear 指令来实现，该指令的第二个参数是需要进行跟随运动的轴号，这里需要注意的是，需要进行跟随运动的轴不能是坐标系中的轴；如果在发送跟随指令 GT_BufGear 时该轴正在运动时，该指令将不能正常执行。该指令的第三个参数是跟随运动的位移量，该位移量是相对值，即下一段插补段运动过程中，跟随轴需要运动的位移量。使用的具体例程如下：

例程 5-6 插补过程中的跟随运动

```
PROGRAM PLC_PRG
VAR
   Enable : BOOL;
   Rtn : INT;
   Run : INT; (* 定义坐标系运动状态查询变量 *)
   Segment : DINT; (* 定义坐标系运动完成段查询变量 *)
END_VAR
----------------------------------------------------------------------------------------------
IF Enable THEN
   rtn = GT_CrdClear(1,0); (* 清除坐标系 1 的 FIFO0 中的数据 *)
   rtn = GT_LnXY(1,200000,200000,100,0.1,0,0); (* 直线插补指令 *)
   rtn = GT_BufGear(1,4,50000, 0); (* 缓存区内的跟随运动指令
                                      跟随运动的轴号：第 4 轴
                                      跟随运动的位移量：50000 pulse *)
   rtn = GT_LnXY(1,200000,0,100,0.1,0,0); (* 直线插补指令 *)
   rtn = GT_BufGear(1,4,50000,0); (* 缓存区内的跟随运动指令
                                     跟随运动的轴号：第 4 轴
                                     跟随运动的位移量：50000 pulse *)
   rtn = GT_ArcXYC(1,-200000,0,-200000,0,0,100,0.1,0,0); (* 圆弧插补指令 *)
   Enable := FALSE;
END_IF
```

例程的运行结果如图 5-16 所示：

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

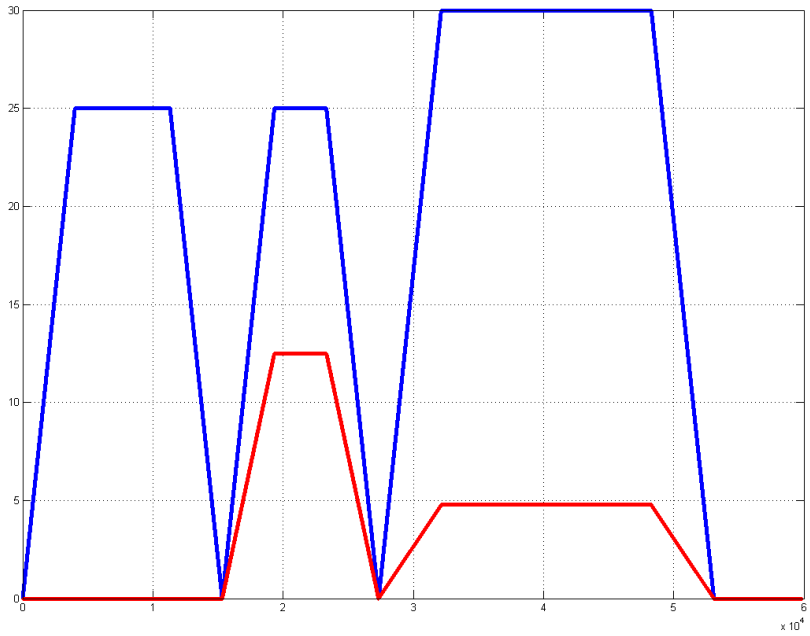

图 5-16 插补过程中的跟随运动曲线

其中蓝色为插补运动的合成速度，红色为点位运动轴的速度值，跟随轴的速度跟随插补运动的合成速度的变化而变化。 在使用插补缓存区中的跟随运动功能时，需要注意以下内容：

a) GT_BufGear 指令需要在所要跟随的插补段前，不要间隔其他种类的指令，可以同时调用多个 GT_BufGear 指令来实现多个轴跟随插补运动。

b) 当暂停坐标系运动时，当插补的合成速度减速到 0 时，跟随轴的速度也会为 0，如果希望重新恢复坐标系运动时，跟随轴仍能够实现跟随，不要调用 GT_Stop()指令停止跟随轴的运动，否则重新启动插补缓存区的运动时，跟随轴将无法完成正在进行的跟随运动。

5.2 PVT 模式

5.2.1 指令列表

表 5-2 PVT 指令列表

指令	说明	页码
GT_PrfPvt	设置指定轴为 PVT 模式	87
GT_SetPvtLoop	设置循环次数	94
GT_GetPvtLoop	查询循环次数	80
GT_PvtTable	向指定数据表传送数据，采用 PVT 描述方式	90
GT_PvtTableComplete	向指定数据表传送数据，采用 Complete 描述方式	90
GT_PvtTablePercent	向指定数据表传送数据，采用 Percent 描述方式	91
GT_PvtPercentCalculate	计算 Percent 描述方式下各数据点的速度	88
GT_PvtTableContinuous	向指定数据表传送数据，采用 Continuous 描述方式	91
GT_PvtContinuousCalculate	计算 Continuous 描述方式下各数据点的时间	88

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

GT_PvtTableSelect	选择数据表	92
GT_PvtStart	启动运动	89
GT_PvtStatus	读取状态	89

5.2.2 重点说明

PVT 模式使用一系列数据点的“位置、速度、时间”参数来描述运动规律。

位置、速度和时间满足如下函数关系：

$$p = at^3 + bt^2 + ct + d$$

$$v = \frac{dp}{dt} = 3at^2 + 2bt + c$$

如果给定相邻 2 个数据点的“位置、速度、时间”参数，可以得到如下方程组：

$$\begin{cases} at_1^3 + bt_1^2 + ct_1 + d = p_1 \\ 3at_1^2 + 2bt_1 + c = v_1 \\ at_2^3 + bt_2^2 + ct_2 + d = p_2 \\ 3at_2^2 + 2bt_2 + c = v_2 \end{cases}$$

求解该方程组，可以得到 a、b、c、d，因此相邻 2 个数据点的运动规律就可以确定下来。

运动控制器提供 32 个数据表存储数据点。每个数据表具有 1024 个存储空间。数据表和轴之间相互独立，一个数据表可以供多个轴使用。

调用 GT_PvtTable、GT_PvtTableComplete、GT_PvtTablePercent 或 GT_PvtTableContinuous 指令向数据表中传递数据。这些指令会删除数据表中原先的数据，因此所有数据应当一次传送完毕。如果使用数据表的轴正在运动，禁止更新数据表。

调用 GT_PvtTableSelect 指令选择数据表。可以在运动状态下切换数据表，但是不会立即切换。只有当前数据表执行完毕以后，才会切换到新的数据表。

调用 GT_PvtStart 启动运动。启动以后，各轴时间清 0。如果第一个数据点的时间为 0 则立即启动，否则会延时启动，延时时间等于第一个数据点的时间。

数据表可以循环执行，调用 GT_SetPvtLoop 设置循环次数，循环次数为 0 表示无限循环。当遍历完数据表以后，时间初始化为第一个数据点的时间，而不是 0。

假设有如下 4 个数据点，采用 PVT 方式进行描述。

表 5-3 PVT 模式示例数据

数据点	时间（毫秒）	位置（脉冲）	速度（脉冲/毫秒）
P1	1,000	0	0
P2	2,000	5,000	10
P3	3,000	15,000	10

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

P4	4,000	20,000	0

(1) 调用 GT_PrfPvt 将轴切换到 PVT 模式；

(2) 调用 GT_PvtTable 将 4 个数据点传递到数据表；

(3) 调用 GT_SetPvtLoop 设置为循环执行；

(4) 调用 GT_PvtStart 启动运动。

由于 P1 的时间为 1000 毫秒，因此调用 GT_PvtStart 以后延时 1000 毫秒启动。由于是循环执行，到达 P4 以后返回到 P1，速度曲线如图 5-17 所示。

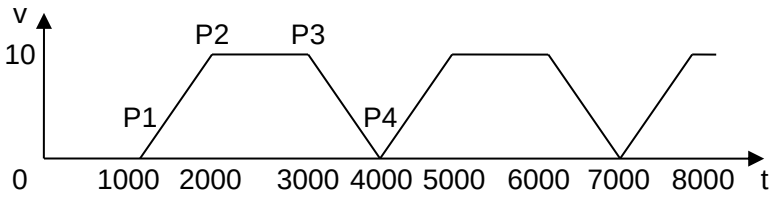

图 5-17 循环执行数据表

PVT 模式有 4 种方式描述运动规律，PVT、Complete、Percent 和 Continuous，下面对此进行详细说明。

1. PVT 描述方式

PVT 描述方式直接定义各数据点的“位置、速度、时间”。相邻 2 个数据点之间，运动控制器使用 3 次多项式对位置进行插值，使用 2 次多项式对速度进行插值。因此当给出各数据点“位置、速度、时间”参数以后，相应的运动规律也就确定下来。

例如下面 4 组数据点，采用 PVT 描述方式。

表 5-4 PVT 描述方式下的四组数据点

数据组	数据点	时间（ms）	位置（pulse）	速度（pulse/ms）
1	P1	0	0	0
	P2	1, 000	5, 000	10
	P3	2, 000	15, 000	10
	P4	3, 000	20, 000	0
2	P1	0	0	0
	P2	1, 000	5, 000	9
	P3	2, 000	15, 000	9
	P4	3, 000	20, 000	0
3	P1	0	0	0
	P2	1, 000	5, 000	7.5
	P3	2, 333	15, 000	7.5
	P4	3, 333	20, 000	0
4	P1	0	0	0
	P2	750	1, 667	6.6669

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

数据组	数据点	时间（ms）	位置（pulse）	速度（pulse/ms）
	P3	2, 250	18, 333	6.6669
	P4	3, 000	20, 000	0

这 4 组数据点对应的运动规律如图 5-18 所示。

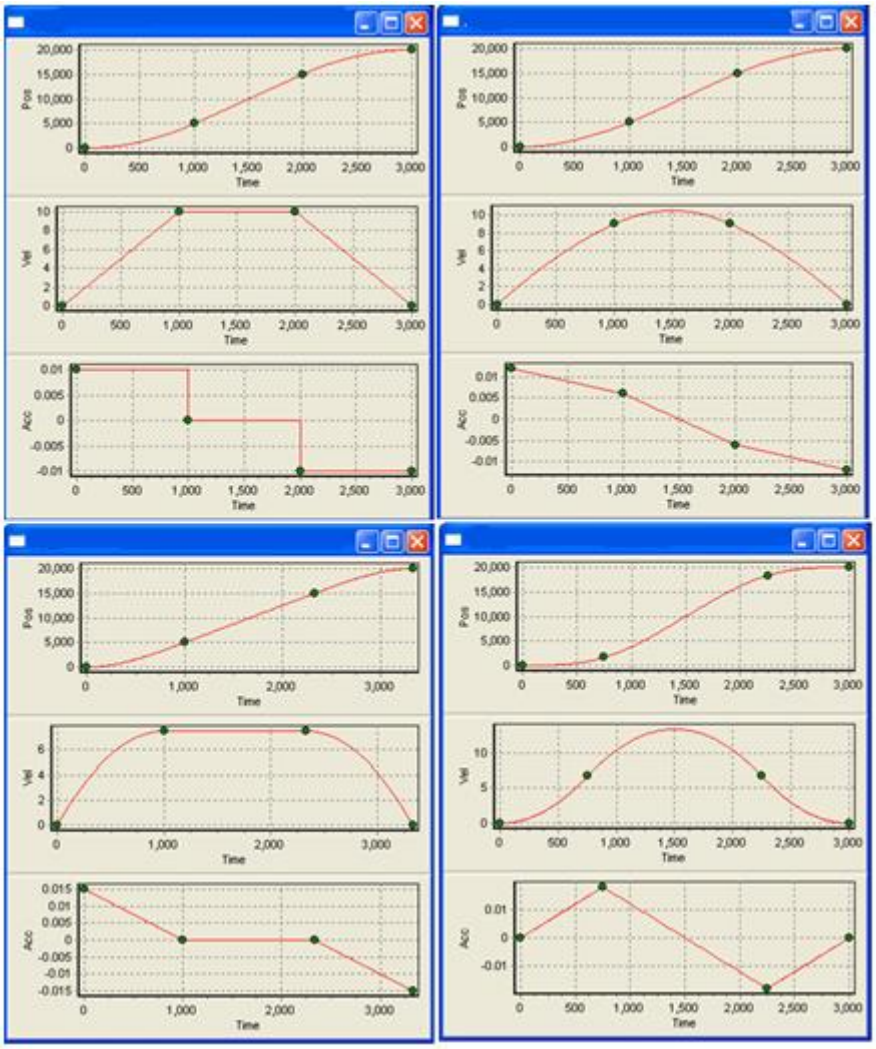

图 5-18 合理的 PVT 描述方式运动规律

可以看出，PVT 描述方式非常灵活。给定数据点的“位置、速度、时间”参数，就能够得到相应的运动规律。

需要注意的是，数据点参数需要仔细设计，否则难以得到理想的运动规律。

例如下面 2 组数据点参数不合理，得到的速度曲线不够平滑。

表 5-5 两组不合理的 PVT 描述方式下的数据点

数据组	数据点	时间（ms）	位置（pulse）	速度（pulse/ms）
1	P1	0	0	0
	P2	1, 000	5, 000	15

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

数据组	数据点	时间（ms）	位置（pulse）	速度（pulse/ms）
	P3	2, 000	15, 000	15
	P4	3, 000	20, 000	0
2	P1	0	0	0
	P2	1, 000	5, 000	5
	P3	2, 000	15, 000	5
	P4	3, 000	20, 000	0

这 2 组数据点对应的运动规律如图 5-19 所示。

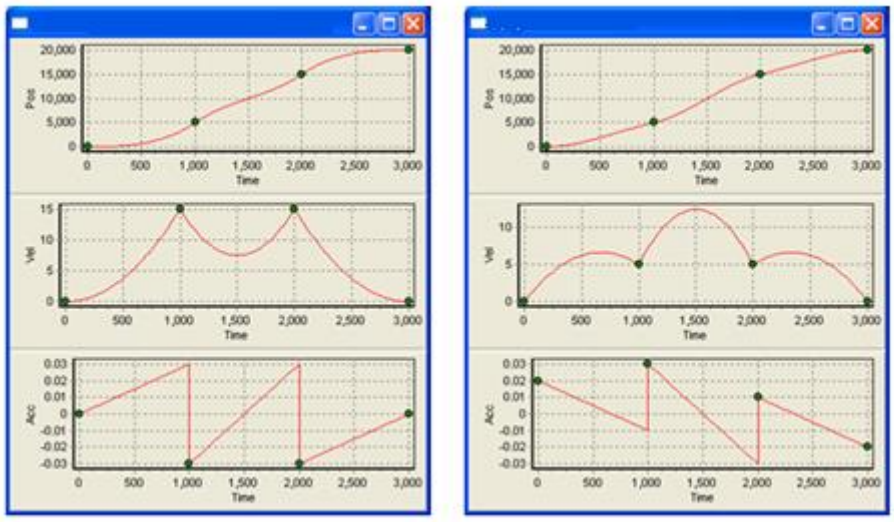

图 5-19 不合理的 PV 描述方式运动规律

2. Complete 描述方式

Complete 描述方式定义各数据点的“位置、时间”，以及起点速度和终点速度。

Complete 方式只定义了起点速度和终点速度。运动控制器根据各数据点的“位置、时间”参数计算中间各点的速度，确保各数据点速度连续和加速度连续。

例如下面这组数据点，采用 Complete 描述方式，可以轻松得到光滑的速度曲线。

表 5-6 Complete 描述方式的一组数据点

数据点	时间（毫秒）	位置（脉冲）	速度（脉冲/毫秒）
P1	0	0	0
P2	1,000	5,000	不指定
P3	2,000	15,000	不指定
P4	3,000	20,000	0

这组数据点对应的运动规律如图 5-20 所示。

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

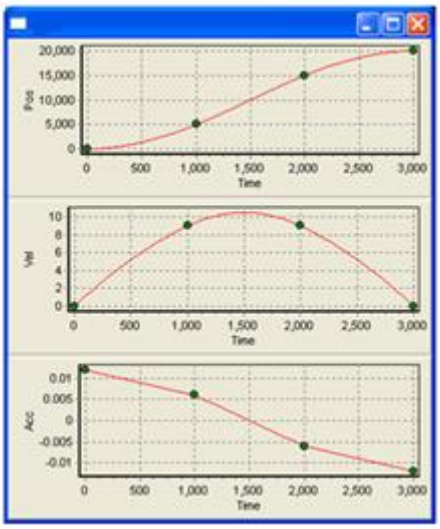

图 5-20 Complete 描述方式运动规律

Complete 适合描述光滑的速度曲线，例如三角函数等。

假设位置和时间之间的关系由函数 $P=50000\sin^2(\pi/2000*t)$确定。在一个函数周期[0,2000]内取 5 个时间点计算相应的位置，如表 5-7 所示。

表 5-7 Complete 方式描述三角函数的数据点

数据点	时间（毫秒）	位置（脉冲）	速度（脉冲/毫秒）
P1	0	0	0
P2	500	25,000	不指定
P3	1,000	50,000	不指定
P4	1,500	25,000	不指定
P5	2,000	0	0

这组数据点对应的运动规律如图 5-21 所示。

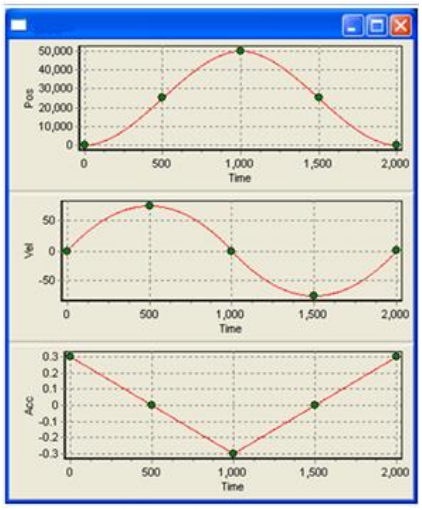

图 5-21 Complete 描述方式运动规律

增加数据点可以减小与函数 $P=50000\sin^2(\pi/2000*t)$的逼近误差。图 5-22 给出了数据点数为 5、10、50 时的位置误差。

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

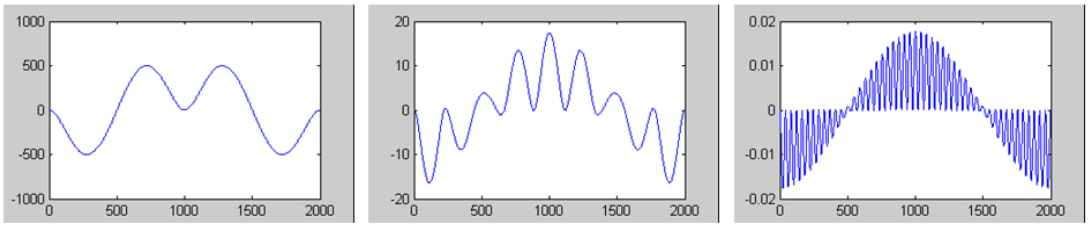

图 5-22 Complete 方式下数据点数分别为 5、10、50 时的位置误差

3. Percent 描述方式

Percent 描述方式定义各数据点的“位置、时间、百分比”，以及起点速度。

Percent 描述方式能够精确定义加速段、匀速段、减速段的位移、速度和时间。

Percent 描述方式假设相邻 2 个数据点之间速度为线性变化，利用起点速度以及各数据点的“位置、时间”参数，通过如下递推公式可以计算出各数据点的速度。

$$v_{i+1} = \frac{2(p_{i+1} - p_i)}{t_{i+1} - t_i} - v_i$$

因此指定了各数据点的“位置、时间”参数以后，各数据点的速度实际上也就已经确定下来。

通过“百分比”参数可以调整速度曲线的光滑性。数据点的百分比参数是指“相邻 2 个数据点之间加速度的变化时间占速度变化时间的百分比”。以图 5-23 为例来进行说明。

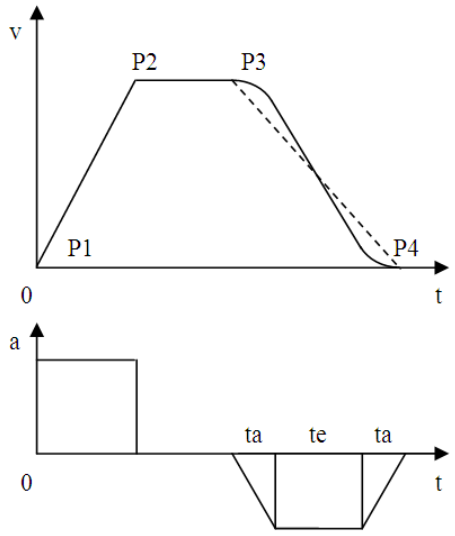

图 5-23 Percent 描述方式下的百分比定义

数据点 P1 和 P2 之间加速度不变，因此数据点 P1 的百分比为 0。

数据点 P2 和 P3 之间加速度不变，因此数据点 P2 的百分比为 0。

数据点 P3 和 P4 之间加速度变化时间为 2ta，运动时间为 2ta+te，因此数据点 P3 的百分比为 2ta/(2ta+te)*100%。

调整百分比参数，不会影响数据点的“位置、时间参数”。以上图为例，当数据点 P3 的百分比为 0

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

时，数据点 P3 和 P4 之间的速度曲线为虚线；当数据点 P3 的百分比不为 0 时，数据点 P3 和 P4 之间的速度曲线为实线。

例如下面这组数据点，采用 Percent 描述方式。

表 5-8 Percent 描述方式例程数据

数据点	时间（毫秒）	位置（脉冲）	百分比	速度（脉冲/毫秒）
P1	0	0	60	0
P2	1,000	5,000	0	不指定
P3	2,000	15,000	100	不指定
P4	3,000	20,000	0	不指定

这组数据点对应的运动规律如图 5-24 所示。

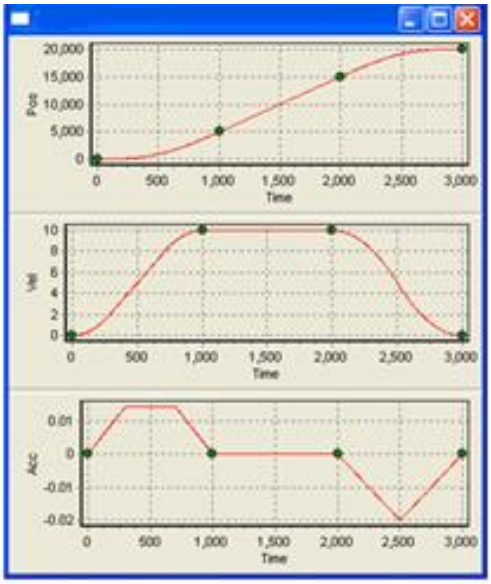

图 5-24 Percent 描述方式下的运动规律

4. Continuous 描述方式

Continuous 描述方式定义各数据点的“位置、速度、最大速度、加速度、减速度、百分比”。不用指定数据点的时间。运动控制器根据数据点参数，自动将相邻 2 个数据点之间拆分为加速段、匀速段和减速段。

数据点 P_i 的最大速度是指从数据点 P_i 到数据点 P_{i+1} 之间的速度上限。

数据点 P_i 的加速度是指从数据点 P_i 到数据点 P_{i+1} 之间的加速段所使用的加速度。

数据点 P_i 的减速度是指从数据点 P_i 到数据点 P_{i+1} 之间的减速段所使用的减速度。

数据点 P_i 的百分比是指从数据点 P_i 到数据点 P_{i+1} 之间的加减速段中，加速度变化时间占速度变化时间的百分比。

相邻 2 个数据点之间能够拆分出来的段数和这 2 个数据点的参数有关，图 5-25 示例了一些可能的分段情况。

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

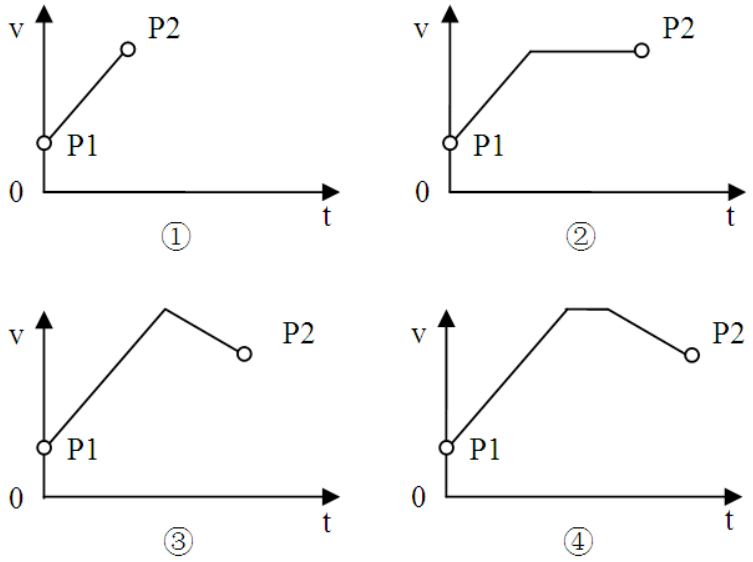

图 5-25 Continuous 描述方式

例如下面这 2 组数据点，采用 Continuous 描述方式。

表 5-9 Continuous 描述方式下的数据点

数据点	位置	速度	最大速度	加速度	减速度	百分比
P1	0	0	10	0.01	0.01	60
P2	20,000	0	10	0.01	0.01	0
数据点	位置	速度	最大速度	加速度	减速度	百分比
P1	0	0	10	0.01	0.01	60
P2	19,800	2	2	0.02	0.02	0
P3	21,800	0	2	0.02	0.02	0

这 2 组数据点对应的运动规律如图 5-26 所示。

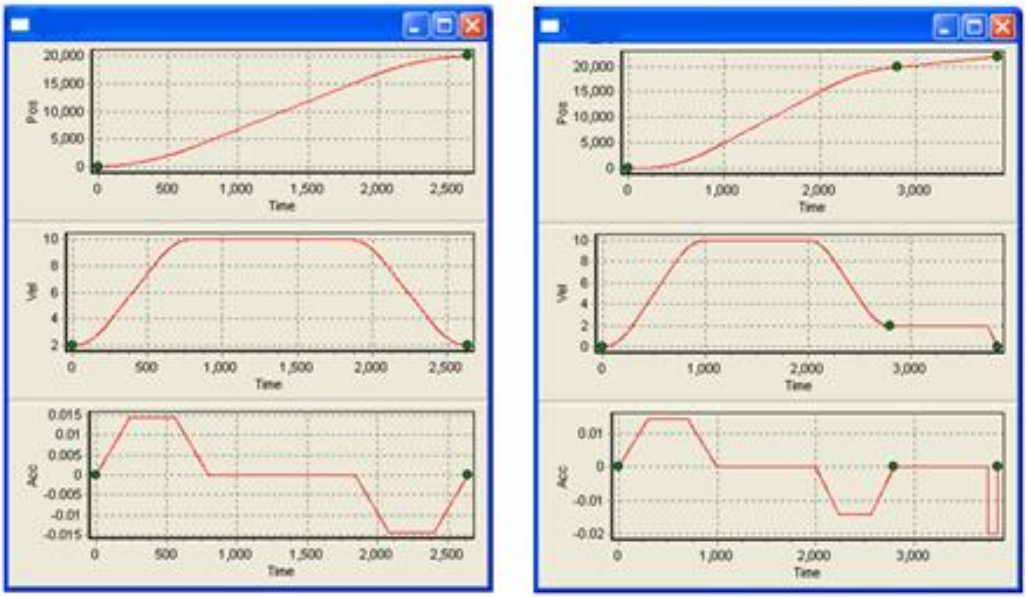

图 5-26 Continuous 描述方式下的运动规律

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

5.2.3 例程

1. PVT 描述方式

如图 5-27 所示，整个速度曲线由 5 段组成，并且带有起跳速度。

- 第一段速度增大，加速度保持不变。
- 第二段速度增大，加速度减小。
- 第三段速度不变，加速度为 0。
- 第四段速度减小，加速度增大。
- 第五段速度减小，加速度不变。

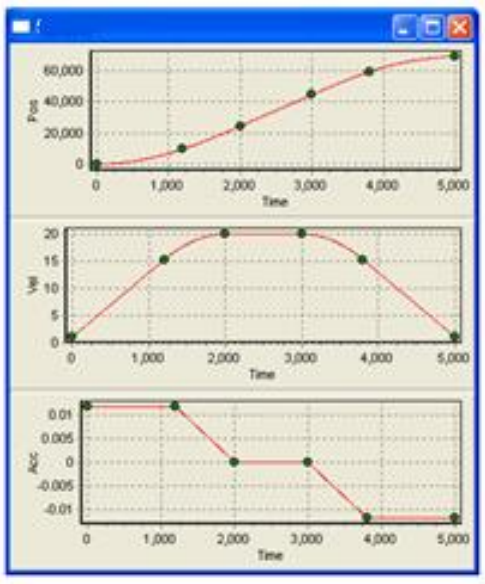

图 5-27 PVT 例程描述方式下的运动规律

可以满足上述要求的一组数据表如下。

表 5-10 PVT 描述方式例程数据点

数据点	时间（毫秒）	位置（脉冲）	速度（脉冲/毫秒）
P1	0	0	1
P2	1,200	9,750	15.25
P3	2,000	24,483	20
P4	3,000	44,483	20
P5	3,800	59,216	15.25
P6	5,000	68,966	1

例程 5-7 PVT 描述方式

```
VAR_GLOBAL CONSTANT
    AXIS:INT:=1
```

```
    TABLE:INT:=1
END_VAR

PROGRAM MAIN
VAR
    Rtn:INT;
    Mask:DINT;
    (* X 轴的数据点参数*)
    AlTime:ARRAY[0..5] OF LREAL:=0,1200,2000,3000,3800,5000;
    Pos:ARRAY[0..5] OF LREAL:=0,9750,24483,44483,59216,68966;
    Vel:ARRAY[0..5] OF LREAL:=1,15.25,20,20,15.25,1;
    prfVel,prfPos,t:LREAL;
    tableId:INT;
    First:BOOL:=TRUE;
END_VAR
-----------------------------------------------------------------------------------------------------------------
IF First THEN
    rtn := GT_AxisOn(AXIS);

    (* 设置为 PVT 模式*)
    rtn := GT_PrfPvt(AXIS);

    (* 发送数据*)
    rtn := GT_PvtTable(TABLE,6,ADR(alTime[0]),ADR(pos[0]),ADR(vel[0]));

    (* 选择数据表*)
    rtn := GT_PvtTableSelect(AXIS,TABLE);

    mask := SHL(1,(AXIS-1));
    rtn := GT_PvtStart(mask);

    First:=FALSE;
END_IF

(* 读取数据表和运动时间*)
rtn := GT_PvtStatus(AXIS,ADR(tableId),ADR(t),1);

(* 读取规划速度*)
rtn = GT_GetPrfVel(AXIS,ADR(prfVel),1,0);

(* 读取规划位置*)
rtn = GT_GetPrfPos(AXIS,ADR(prfPos),1,0);
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

2. Complete 描述方式

假设位置和时间之间的关系由函数 $P=40000\sin^2(\pi/2000*t)$确定。要求启动以后能够循环运动，按 A 键幅值增大 50%，按 B 键幅值减小 50%。

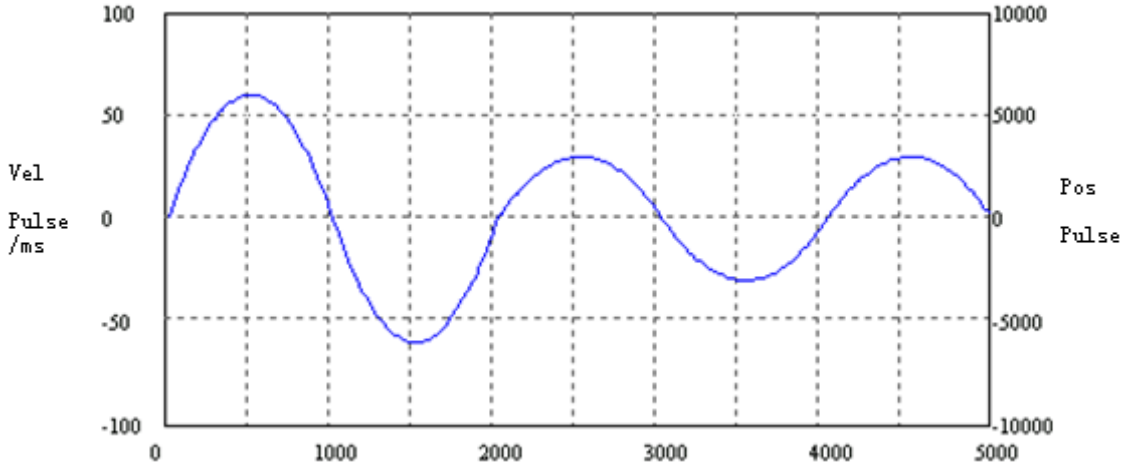

图 5-28 Complete 描述方式下的速度曲线

例程 5-8 Complete 描述方式

```
VAR_GLOBAL CONSTANT
    AXIS:INT:=1
    TABLE1:INT:=1
    TABLE2:INT:=2
    PI:LREAL:=3.1415926
END_VAR

FUNCTION Calculate : INT
VAR_INPUT
    Amplitude:LREAL;
    n:DINT;
    pTime:POINTER TO LREAL;
    pPos:POINTER TO LREAL;
END_VAR

VAR
    i:DINT;
END_VAR
----------------------------------------------------------------------------------------------------------------
FOR i:=0 TO n-1 BY 1 DO
    pPos[i] := amplitude*sin(PI/2000*pTime[i])*sin(PI/2000*pTime[i]);
END_FOR

PROGRAM MAIN
VAR
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
        Rtn:INT;
        Mask:DINT;
        (* X 轴的数据点参数*)
        AlTime:ARRAY[0..4] OF LREAL:= 0,500,1000,1500,2000;
        Pos:ARRAY[0..4] OF LREAL;
        a,b,c: ARRAY[0..4] OF LREAL;
        prfVel,prfPos,t:LREAL;
        tableId:INT;
        amplitude:LREAL:=40000;
        table:INT:= TABLE1;
        key:STRING(1);
        First:BOOL:=TRUE;
    END_VAR
    ----------------------------------------------------------------------------------------------------------------
    IF First THEN
        rtn := GT_AxisOn(AXIS);

        (* 设置为 PVT 模式*)
        rtn := GT_PrfPvt(AXIS);

        Calculate(amplitude,5,ADR(alTime[0]),ADR(pos[0]));

        (* 发送数据*)
    rtn:=GT_PvtTableComplete(table,5,ADR(alTime[0]),ADR(pos[0]),ADR(a[0]),ADR(b[0]),ADR(c[0]
),0,0);

        (* 选择数据表*)
        rtn := GT_PvtTableSelect(AXIS,table);

        (* 设置为循环执行*)
        rtn := GT_SetPvtLoop(AXIS,0);
        mask := SHL(1,(AXIS-1));
        rtn := GT_PvtStart(mask);
        First:=FALSE;
    END_IF

    (* 读取数据表和运动时间*)
    rtn := GT_PvtStatus(AXIS,ADR(tableId),ADR(t),1);

    (* 读取规划速度*)
    rtn = GT_GetPrfVel(AXIS,ADR(prfVel),1,0);

    (* 读取规划位置*)
    rtn = GT_GetPrfPos(AXIS,ADR(prfPos),1,0);
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
    IF key<>'' THEN
        IF ( 'A' = key ) THEN
            amplitude := amplitude *1.5;
        END_IF
        IF ( 'B' = key) THEN
            amplitude := amplitude *0.5;
        END_IF
    IF ( 'A' = key ) OR ( 'B' = key ) THEN
            Calculate(amplitude,5,ADR(alTime[0]),ADR(pos[0]));
            table := TABLE1 + TABLE2 - tableId;

    (*发送数据*）
            rtn := GT_PvtTableComplete(table, 5, ADR(alTime[0]), ADR(pos[0]), ADR(a[0]),
ADR(b[0]), ADR(c[0]), 0, 0);

            (*选择数据表*)
            rtn := GT_PvtTableSelect(AXIS,table);
        END_IF

        IF ( 'Q' = key) THEN
            (*停止轴运动*)
            mask := SHL(1,(AXIS-1));
            GT_Stop(mask,0);
        END_IF
    END_IF
```

3. Percent 描述方式

X 轴往复运动，Y 轴正向进给。X 轴加减速时 Y 轴开始进给，X 轴匀速运动时，Y 轴保持静止。

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

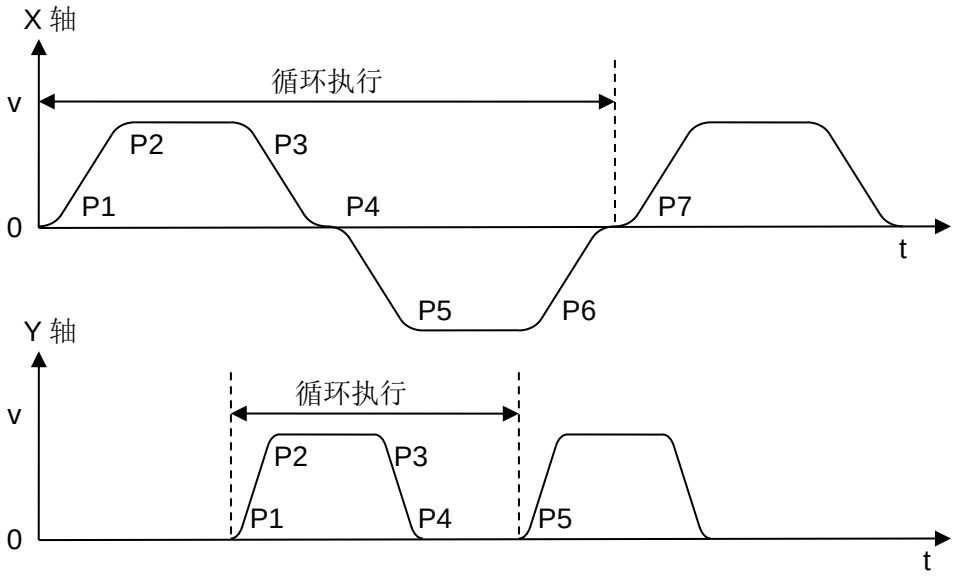

图 5-29 Percent 描述方式下 X 轴和 Y 轴的运动规律

X 轴取 7 个数据点，设置为循环模式。数据点参数如下：

表 5-11 Percent 描述方式下的数据点 1

数据点	时间（毫秒）	位置（脉冲）	百分比	速度（脉冲/毫秒）
P1	0	0	60	0
P2	1,000	5,000	0	不指定
P3	2,000	15,000	60	不指定
P4	3,000	20,000	60	不指定
P5	4,000	15,000	0	不指定
P6	5,000	5,000	60	不指定
P7	6,000	0	0	不指定

根据 X 轴数据点参数，可以计算出各数据点的速度，百分比参数对数据点的速度计算没有影响。

表 5-12 Percent 描述方式下的数据点 2

数据点	时间（毫秒）	位置（脉冲）	速度（脉冲/毫秒）
P1	0	0	0
P2	1,000	5,000	2(5000-0)/(1000-0)-0=10
P3	2,000	15,000	2(15000-5000)/(2000-1000)-10=10
P4	3,000	20,000	2(20000-15000)/(3000-2000)-10=0
P5	4,000	15,000	2(15000-20000)/(4000-3000)-0=-10
P6	5,000	5,000	2(5000-15000)/(5000-4000)-(-10)=-10
P7	6,000	0	2(0-5000)/(6000-5000)-(-10)=0

Y 轴取 5 个数据点，设置为循环模式。X 轴启动以后到达数据点 P3 时 Y 轴才启动，因此第 1 个数据点的时间设置为 2000 毫秒。当 Y 轴到达 P5 以后，返回到 P1 循环执行。数据点参数如下：

表 5-13 Percent 描述方式下的数据点 3

数据点	时间（毫秒）	位置（脉冲）	百分比	速度（脉冲/毫秒）

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

P1	2,000	0	60	0
P2	2,500	2,500	0	不指定
P3	3,500	12,500	60	不指定
P4	4,000	15,000	0	不指定
P5	5,000	15,000	0	不指定

根据 Y 轴数据点参数，可以计算出各数据点的速度，百分比参数对数据点的速度计算没有影响。

表 5-14 Percent 描述方式下的数据点 4

数据点	时间（毫秒）	位置（脉冲）	速度（脉冲/毫秒）
P1	2,000	0	0
P2	2,500	2,500	2(2500-0)/(2500-2000)-0=10
P3	3,500	12,500	2(12500-2500)/(3500-2500)-10=10
P4	4,000	15,000	2(15000-12500)/(4000-3500)-10=0
P5	5,000	15,000	2(15000-15000)/(5000-4000)-0=0

X 轴循环 n 次，Y 轴需要循环 2n-1 次。**错误!未找到引用源。**是当 X 轴的循环次数为 2，Y 轴循环次数为 3 时的 XY 位置图。横轴是 X 轴的位置，纵轴是 Y 轴的位置。黄线是实际的运动轨迹。

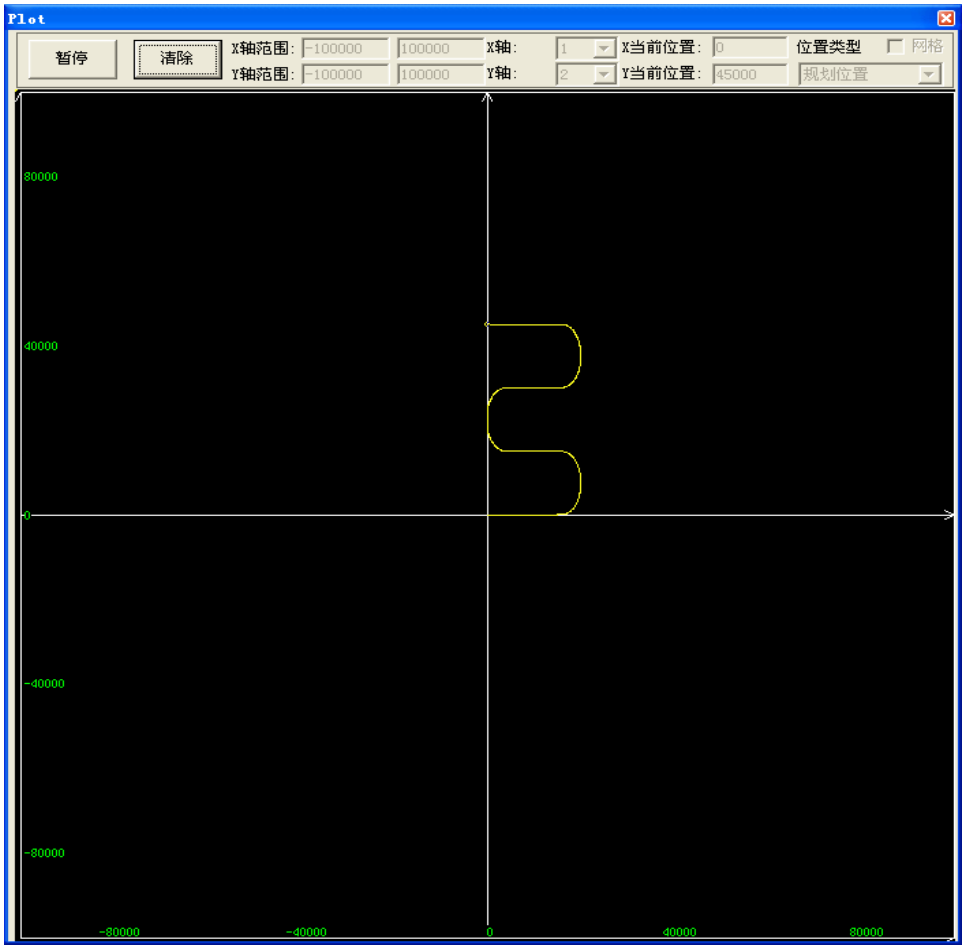

图 5-30 X-Y 位置图

例程 5-9 Percent 描述方式

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
VAR_GLOBAL CONSTANT
   AXIS_X:INT:=1
   AXIS_Y:INT:=2

   TABLE_X:INT:=1
   TABLE_Y:INT:=2
   LOOP_COUNT:INT:=2;
END_VAR

PROGRAM MAIN
VAR
      Rtn:INT;
      Mask:DINT;
      (* X 轴的数据点参数*)
      Time_x:ARRAY[0..6] OF LREAL:= 0,1000,2000,3000,4000,5000,6000;
      pos_x:ARRAY[0..6] OF LREAL:= 0,5000,15000,20000,15000,5000,0;
      percent_x:ARRAY[0..6] OF LREAL:= 60,0,60,60,0,60,0;
      Time_y:ARRAY[0..4] OF LREAL:= 2000,2500,3500,4000,5000;
      pos_y:ARRAY[0..4] OF LREAL:= 0,2500,12500,15000,15000;
      percent_y:ARRAY[0..4] OF LREAL:= 60,0,60,0,0;

      prfVel, prfPos, alTime:ARRAY[0..1] OF LREAL;
      tableId:INT;
      First:BOOL:=TRUE;
END_VAR
-----------------------------------------------------------------------------------------------------------
IF First THEN

      rtn := GT_AxisOn(AXIS_X);
      rtn := GT_AxisOn(AXIS_Y);

      (*  将 X 轴设置为 PVT 模式*)
      rtn := GT_PrfPvt(AXIS_X);

      (*  将 Y 轴设置为 PVT 模式*)
      rtn := GT_PrfPvt(AXIS_Y);

      (*  向 X 轴的数据表发送数据*)
      rtn := GT_PvtTablePercent(TABLE_X,7,ADR(time_x[0]),ADR(pos_x[0]),ADR(percent_x[0]),0);

      (*  向 Y 轴的数据表发送数据*)
      rtn := GT_PvtTablePercent(TABLE_Y,5,ADR(time_y[0]),ADR(pos_y[0]),ADR(percent_y[0]),0);

      (* X 轴选择数据表 TABLE_X*)
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
        rtn := GT_PvtTableSelect(AXIS_X,TABLE_X);

        (* Y 轴选择数据表 TABLE_Y*)
        rtn := GT_PvtTableSelect(AXIS_Y,TABLE_Y);

        (* 设置循环次数*)
        rtn := GT_SetPvtLoop(AXIS_X,LOOP_COUNT);

        (* 设置循环次数*)
        rtn := GT_SetPvtLoop(AXIS_Y,2*LOOP_COUNT-1);

        (* 同时启动 X 轴和 Y 轴*)
        (* 由于 Y 轴的第 1 个数据点时间为 2000ms*)
        (* 因此 X 轴启动 2000ms 以后，Y 轴才开始运动*)
        mask := SHL(1,(AXIS_X-1));
        mask := mask OR SHL(1,(AXIS_Y-1));
        rtn := GT_PvtStart(mask);
        First:=FALSE;
    END_IF

    (* 读取数据表和运动时间*)
    rtn := GT_PvtStatus(AXIS_X,ADR(tableId[0]),ADR(alTime[0]),1);
    rtn := GT_PvtStatus(AXIS_Y,ADR(tableId[1]),ADR(alTime[1]),1);

    (* 读取规划速度*)
    rtn := GT_GetPrfVel(AXIS_X,ADR(prfVel[0]),1,0);
    rtn := GT_GetPrfVel(AXIS_Y,ADR(prfVel[1]),1,0);

    (* 读取规划位置*)
    rtn := GT_GetPrfPos(AXIS_X,ADR(prfPos[0]),1,0);
    rtn := GT_GetPrfPos(AXIS_Y,ADR(prfPos[1]),1,0);
```

4. Continuous 描述方式

X 轴从 A 点运动到 B 点，Y 轴从 C 点运动到 D 点。要求 X 轴到达 B 点时，Y 轴同时到达 D 点。

首先调用 GT_PvtContinuousCalculate 指令计算 X 轴的运动时间 t_x 和 Y 轴的运动时间 t_y。该指令不会把数据点发送到运动控制器。如果 $t_x>t_y$，Y 轴延时 t_x-t_y 启动；如果 $t_x<t_y$，X 轴延时 t_y-t_x 启动。

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

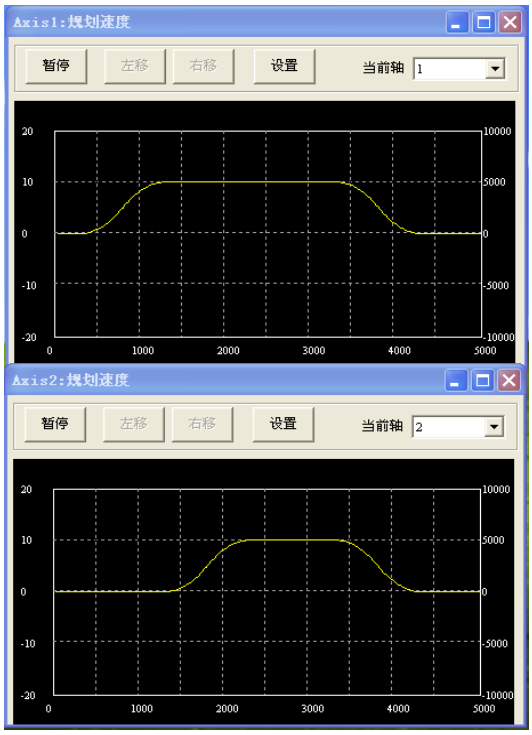

图 5-31 X 轴和 Y 轴的速度曲线

例程 5-10 Continuous 描述方式

```
VAR_GLOBAL CONSTANT
    AXIS_X:INT:=1
    AXIS_Y:INT:=2
    TABLE_X:INT:=1
    TABLE_Y:INT:=2
END_VAR
----------------------------------------------------------------------------------
PROGRAM MAIN
VAR
    Rtn:INT;
    Mask:DINT;
    (* X 轴的数据点参数*)
    pos_x:ARRAY[0..1] OF LREAL:= 0,30000;
    vel_x:ARRAY[0..1] OF LREAL:= 0,0;
    velMax_x:ARRAY[0..1] OF LREAL:= 10,10;
    percent_x:ARRAY[0..6] OF LREAL:=100,100;
    acc_x:ARRAY[0..1] OF LREAL:= 0.01,0.01;
    dec_x:ARRAY[0..1] OF LREAL:= 0.01,0.01;
    time_x:ARRAY[0..1] OF LREAL;
    timeBegin_x:LREAL;

    (* Y 轴的数据点参数*)
    pos_y:ARRAY[0..1] OF LREAL:= 0,20000;
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
        vel_y:ARRAY[0..1] OF LREAL:= 0,0;
        velMax_y:ARRAY[0..1] OF LREAL:= 10,10;
        percent_y:ARRAY[0..6] OF LREAL:=100,100;
        acc_y:ARRAY[0..1] OF LREAL:= 0.01,0.01;
        dec_y:ARRAY[0..1] OF LREAL:= 0.01,0.01;
        time_y:ARRAY[0..1] OF LREAL;
        timeBegin_y:LREAL;

        prfVel, prfPos, alTime:ARRAY[0..1] OF LREAL;
        tableId: ARRAY[0..1] OF INT;
        First:BOOL:=TRUE;
    END_VAR
    ----------------------------------------------------------------------------------------------------------------
    IF First THEN
        rtn := GT_AxisOn(AXIS_X);
        rtn := GT_AxisOn(AXIS_Y);

        (* 将 X 轴设置为 PVT 模式*)
        rtn := GT_PrfPvt(AXIS_X);
        (* 将 Y 轴设置为 PVT 模式*)
        rtn := GT_PrfPvt(AXIS_Y);

        (* 计算 X 轴运动时间*)
        rtn := GT_PvtContinuousCalculate(2, ADR(pos_x[0]), ADR(vel_x[0]), ADR(percent_x[0]),
ADR(velMax_x[0]),ADR(acc_x[0]), ADR(dec_x[0]), ADR(time_x[0]));
        (* 计算 Y 轴运动时间*)
        rtn := GT_PvtContinuousCalculate(2, ADR(pos_y[0]), ADR(vel_y[0]), ADR(percent_y[0]),
ADR(velMax_y[0]), ADR(acc_y[0]), ADR(dec_y[0]), ADR(time_y[0]));

        (* 计算启动延时*)
        IF time_x[1] < time_y[1] THEN
            timeBegin_x := time_y[1] - time_x[1];
            timeBegin_y := 0;
        ELSE
            timeBegin_x := 0;
            timeBegin_y := time_x[1] - time_y[1];
        END_IF

        (* 发送 X 轴数据点*)
        rtn := GT_PvtTableContinuous(TABLE_X,2, ADR(pos_x[0]), ADR(vel_x[0]),
ADR(percent_x[0]), ADR(velMax_x[0]), ADR(acc_x[0]), ADR(dec_x[0]),timeBegin_x);
        (* 发送 Y 轴数据点*)
        rtn := GT_PvtTableContinuous(TABLE_Y,2, ADR(pos_y[0]), ADR(vel_y[0]),
ADR(percent_y[0]), ADR(velMax_y[0]), ADR(acc_y[0]), ADR(dec_y[0]),timeBegin_y);
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
        (* X 轴选择数据表 TABLE_X*)
        rtn := GT_PvtTableSelect(AXIS_X,TABLE_X);
        (* Y 轴选择数据表 TABLE_Y*)
        rtn := GT_PvtTableSelect(AXIS_Y,TABLE_Y);

        (*  同时启动 X 轴和 Y 轴*)
        (*  由于 Y 轴的第 1 个数据点时间为 2000ms*)
    (*  因此 X 轴启动 2000ms 以后，Y 轴才开始运动*)
        mask := SHL(1,(AXIS_X-1));
        mask := mask OR SHL(1,(AXIS_Y-1));
        rtn := GT_PvtStart(mask);
        First:=FALSE;
    END_IF

    (*  读取数据表和运动时间*)
    rtn := GT_PvtStatus(AXIS_X,ADR(tableId[0]),ADR(alTime[0]),1);
    rtn := GT_PvtStatus(AXIS_Y,ADR(tableId[1]),ADR(alTime[1]),1);

    (*  读取规划速度*)
    rtn := GT_GetPrfVel(AXIS_X,ADR(prfVel[0]),1,0);
    rtn := GT_GetPrfVel(AXIS_Y,ADR(prfVel[1]),1,0);

    (*  读取规划位置*)
    rtn := GT_GetPrfPos(AXIS_X,ADR(prfPos[0]),1,0);
    rtn := GT_GetPrfPos(AXIS_Y,ADR(prfPos[1]),1,0);
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

第6章 运动程序

6.1 简介

为了表述方便，直接在 PC 机上调用动态链接库发送指令访问控制器的程序称为“应用程序”，下载到运动控制器上执行的程序称为“运动程序”。

C 语言编写的运动程序编译以后能够下载到运动控制器中执行。运动程序能够脱离主机在运动控制器上独立执行，从而将主机从繁琐的运动逻辑管理中解放出来。一方面主机能够将 CPU 资源分配给其它任务，另一方面运动程序能够直接访问运动控制器，不需要进行频繁通讯，从而具有更高的实时性。

当然，如果需要，主机仍然可以在任何时候向控制器发送指令，即使运动控制器上的运动程序正在执行。注意：当主机指令和运动控制器上的运动程序控制相同的轴时，需要仔细设计运动逻辑，以免造成混乱。

运动控制器允许多达 32 个运动程序在运动控制器上同时执行。

运动控制器内建的线程调度机制保证多线程环境下运动程序所有指令的执行都是完整的。

在多线程环境下，一个线程中连续的 2 条指令在执行时有可能被插入其它线程的指令。当启动多个线程并行执行时，应当仔细考虑线程之间是否会相互影响。

6.2 编写运动程序

提示	本章表格中右侧的数字为“页码”，其中指令右侧的为“**第 7 章指令详细说明**”中的对应页码，其他为章节页码，均可以使用“超级链接”进行索引。 本手册中所有字体为蓝色的指令（如 GT_ArcXYR）均带有超级链接，点击可跳转至指令说明。

6.2.1 指令列表

表 6-1 运动控制指令列表

指令	说明	页码
GT_Download	下载运动程序到运动控制器	78
GT_GetFunId	读取运动程序中函数的标识	80
GT_GetVarId	读取运动程序中变量的标识	82
GT_Bind	绑定线程、函数、数据页	71
GT_RunThread	启动线程	92
GT_StopThread	停止正在运行的线程	96
GT_PauseThread	暂停正在运行的线程	87

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

GT_GetThreadSts	读取线程的状态	81
GT_SetVarValue	设置运动程序中变量的值	96
GT_GetVarValue	读取运动程序中变量的值	82

6.2.2 重点说明

编写运动程序的方法如下：

1) 使用文本编辑器编写运动程序的 C 语言源程序。

2) 使用 MCT2008 编译运动程序，生成目标程序文件（*.bin）和符号文件（*.ini）。

3) 调用 GT_Download 指令将运动程序目标程序下载到运动控制器中。

4) 调用 GT_GetFunId 指令获取函数 ID

5) 调用 GT_GetVarId 指令获取变量 ID

6) 调用 GT_Bind 指令绑定线程、函数和数据页。

7) 调用 GT_SetVarValue 指令初始化局部变量和全局变量。

8) 调用 GT_RunThread 指令，启动线程。

9) 调用 GT_GetThreadSts 指令查询线程状态，或者调用 GT_GetVarValue 指令查询变量。

调用 GT_Download 指令可以将运动程序下载到运动控制器的 SDRAM 中。当下载新的运动程序时会覆盖原有的运动程序。运动控制器每次上电以后需要重新下载运动程序。在发布应用程序时，应同时发布目标文件和符号文件。否则运动程序无法正确下载执行。

运动程序下载到运动控制器以后还不能立即执行，必须调用 GT_Bind 指令绑定线程、函数和数据页，然后调用 GT_RunThread 指令启动线程。运动控制器支持 32 个线程同时运行，一个线程只能分配一个函数，但是一个函数可以分配给多个线程同时执行，例如多轴回零时，可以让多个线程绑定同一个回零函数，然后同时启动这些线程就可以实现多轴同时回零。在线程执行过程中不允许绑定新的函数，除非线程执行完毕。

各函数的局部变量放在相互独立的数据页中。运动控制器提供 32 个数据页。在绑定线程和函数时，必须指明所使用的数据页。一个数据页只能分配给一个线程，但是一个线程可以使用多个数据页。线程在执行过程中可以切换数据页。

应用程序可以随时调用 GT_GetThreadSts 指令查询线程的执行状态。

应用程序可以随时调用 GT_SetVarValue 指令更新运动程序中所有变量的值。

应用程序可以随时调用 GT_GetVarValue 指令查询运动程序中所有变量的值。

6.2.3 例程

1. 单线程累加求和

例程 6-1 运动程序单线程累加求和

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

运动程序完成累加求和任务。定义了全局变量 sum 用于保存累加和，局部变量 begin 用于保存累加起点，局部变量 end 用于保存累加终点。累加完成以后程序结束。

```
//-----------------------------------------------
// 累加求和
// begin  累加起点
// end    累加终点
//-----------------------------------------------
int sum;

int add(int begin,int end)
{
    int i;
    int cc;

    i=begin;
lbl_loop:
    cc = i > end;
    if(cc) goto lbl_end;
    sum = sum + i;
    i = i + 1;
    goto lbl_loop;
lbl_end:
    return sum;
}
```

应用程序负责编译、下载、初始化、启动运动程序。

```
PROGRAM MAIN
VAR
    rtn: INT;
    compile: TCompileInfo;
    filename:STRING:='sum.c';
    downloadfilename:STRING:='sum.bin';
    funname:STRING:='add';
    funid: INT;
    varname:STRING:='sum';
    sum:TVarInfo;
    begin:TVarInfo;
    end:TVarInfo;
    value:DINT;
    thread:TThreadSts;
    First: BOOL := TRUE;
END_VAR
-------------------------------------------------------------------------------
IF First THEN
    (*复位运动控制器*)
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
        rtn:=GT_Reset();

        (*编译运动程序 sum.c*)
        (*编译成功之后生成 sum.bin 和 sum.ini*)
        (*必须保证 error.ini 文件位于工程文件夹中*)
        rtn:=GT_Compile(ADR(filename), ADR(compile));

        (*下载运动程序 sum.bin*)
        rtn:= GT_Download(ADR(downloadfilename));

        (*获取函数 ID*)
        rtn:= GT_GetFunId(ADR(funname), ADR(funid));

        (*获取全局变量 sum 的 ID*)
        rtn:= GT_GetVarId(0, ADR(varname), ADR(sum));

        (*获取局部变量 begin 的 ID*)
        varname:='begin';
rtn:= GT_GetVarId(ADR(funname), ADR(varname), ADR(begin));

        (*获取局部变量 end 的 ID*)
        varname:='end';
rtn:= GT_GetVarId(ADR(funname), ADR(varname), ADR(end));

        (*绑定线程，函数，数据页*)
        rtn:= GT_Bind(0, funId, 0);

        value:=0;
        (*初始化运动程序的全局变量 sum*)
        rtn:= GT_SetVarValue(-1, ADR(sum), ADR(value), 1);

        value:=1;
        (*初始化运动程序的局部变量 begin*)
        rtn:= GT_SetVarValue(0, ADR(begin), ADR(value), 1);

        value:=100;
        (*初始化运动程序的局部变量 end*)
        rtn:= GT_SetVarValue(0, ADR(end), ADR(value), 1);

        (*启动线程*)
        rtn: GT_RunThread(0);

        First:=FALSE;
END_IF
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
(*查询线程状态*)
rtn:= GT_GetThreadSts(0,ADR(thread));

(*查询全局变量 sum 的值*)
rtn:= GT_GetVarValue(-1, ADR(sum), ADR(value), 1);
```

2. 多线程累加求和

例程 6-2 运动程序多线程累加求和

运动程序代码和例程 1 相同。

应用程序负责编译、下载、初始化、启动运动程序。和例程 1 不同之处在于启动 2 个线程完成累加运算任务。

```
PROGRAM MAIN
VAR
    rtn:INT;
    compile:TCompileInfo;
    funId:INT;
    sum,begin,end:TVarInfo;
    value:DINT;
    thread:TThreadSts;

    First: BOOL:=TRUE;
    filename: STRING:='sum.c';
    downloadfilename: STRING:='sum.bin';
    funname: STRING:='add';
    varname:STRING:='sum';
END_VAR
-------------------------------------------------------------------------------
IF First THEN
    (*初始化控制器*)
    rtn:=GT_Reset();

    (*编译运动程序 sum.c*)
    (*编译成功之后生成 sum.bin 和 sum.ini*)
    (*必须保证 error.ini 文件位于工程文件夹中*)
    rtn:=GT_Compile(ADR(filename), ADR(compile));

    (*下载运动程序 sum.bin*)
    rtn:= GT_Download(ADR(downloadfilename));

    (*获取函数 ID*)
    rtn:= GT_GetFunId(ADR(funname), ADR(funId));
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
        (*获取全局变量 sum 的 ID*)
        rtn:= GT_GetVarId(0, ADR(varname), ADR(sum));

        (*获取局部变量 begin 的 ID*)
        varname:='begin';
        rtn:= GT_GetVarId(ADR(funname), ADR(varname), ADR(begin));

        (*获取局部变量 end 的 ID*)
        varname:='end';
        rtn:= GT_GetVarId(ADR(funname), ADR(varname), ADR(end));

        (*绑定线程，函数，数据页*)
        rtn:= GT_Bind(0, funId, 0);
        rtn:= GT_Bind(1, funId, 1);

        (*初始化运动程序的全局变量 sum*)
        value:=0;
        rtn:= GT_SetVarValue(-1, ADR(sum), ADR(value), 1);

        (*初始化线程 0 运动程序的局部变量 begin*)
        value:=1;
        rtn:= GT_SetVarValue(0, ADR(begin), ADR(value), 1);

        (*初始化线程 0 运动程序的局部变量 end*)
        value:=50;
        rtn:= GT_SetVarValue(0, ADR(end), ADR(value), 1);

        (*初始化线程 1 运动程序局部变量 begin*)
        value:=51;
        rtn:= GT_SetVarValue(1, ADR(begin), ADR(value), 1);

        (*初始化线程 1 运动程序局部变量 end*)
        value:=100;
        rtn:= GT_SetVarValue(1, ADR(end), ADR(value), 1);

        (*启动线程 0, 1*)
        rtn:= GT_RunThread(0);
        rtn:= GT_RunThread(1);

        First:=FALSE;
    END_IF

    (*查询线程状态*)
    rtn:= GT_GetThreadSts(0, ADR(thread));
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
rtn:= GT_GetThreadSts(1, ADR(thread));

(*查询全局变量 sum 的值*)
rtn:= GT_GetVarValue(-1, ADR(sum), ADR(value), 1);
……
```

3. 组合运动

例程 6-3 组合运动

该例程实现 3 个运动轴协调运动，其应用场景为半导体加工设备，定义 3 个运动轴分别为摆臂轴、送料轴和点胶轴，其中各个轴的运动关系如下所述：

- 摆臂轴：点胶轴离开工作区域时，启动摆臂轴向工作区域运动，完成工作后无需等待任何信号，即可撤离工作区域；
- 送料轴：摆臂轴离开工作区域时，即可启动送料轴运动；
- 点胶轴：摆臂轴离开工作区域之后，启动点胶轴向工作区域运动；当送料轴运动到位时，点胶轴完成点胶工作，然后离开工作区域。

运动程序将三个轴的运动分别用三个函数来实现：ArmMotion 控制摆臂轴，GlueMotion 控制点胶轴，PieceMotion 控制物料轴，通过四个全局同步变量，来实现各个运动之间的互斥和关联：

- pieceArrival：标志送料轴已经到达，可以启动点胶轴的运动离开工作区域；
- pieceStart：启动送料轴标志位，标志送料轴可以启动；
- glueStart：启动点胶轴标志位，标志点胶轴可以开始运动到工作区域；
- armStart：启动摆臂轴标志位，标志摆臂轴可以开始向工作区域运动。

ArmMotion 函数与线程 0 和数据页 0 绑定；GlueMotion 函数与线程 1 和数据页 1 绑定；PieceMotion 函数与线程 2 和数据页 2 绑定，每个线程独立控制一个轴的运动。运动程序的变量初始值设置如下：

- 全局变量 pieceArrival 值：0；(物料轴未到达)
- 全局变量 armStart 值：0；(初始状态下摆臂轴运动条件不满足)
- 全局变量 pieceStart 值：1；(初始状态下物料轴运动条件满足)
- 全局变量 glueStart 值：1；(初始状态下点胶轴运动条件满足)

图 6-1 中 1 轴是摆臂轴规划速度，2 轴是点胶轴规划速度，3 轴是物料轴规划速度。

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

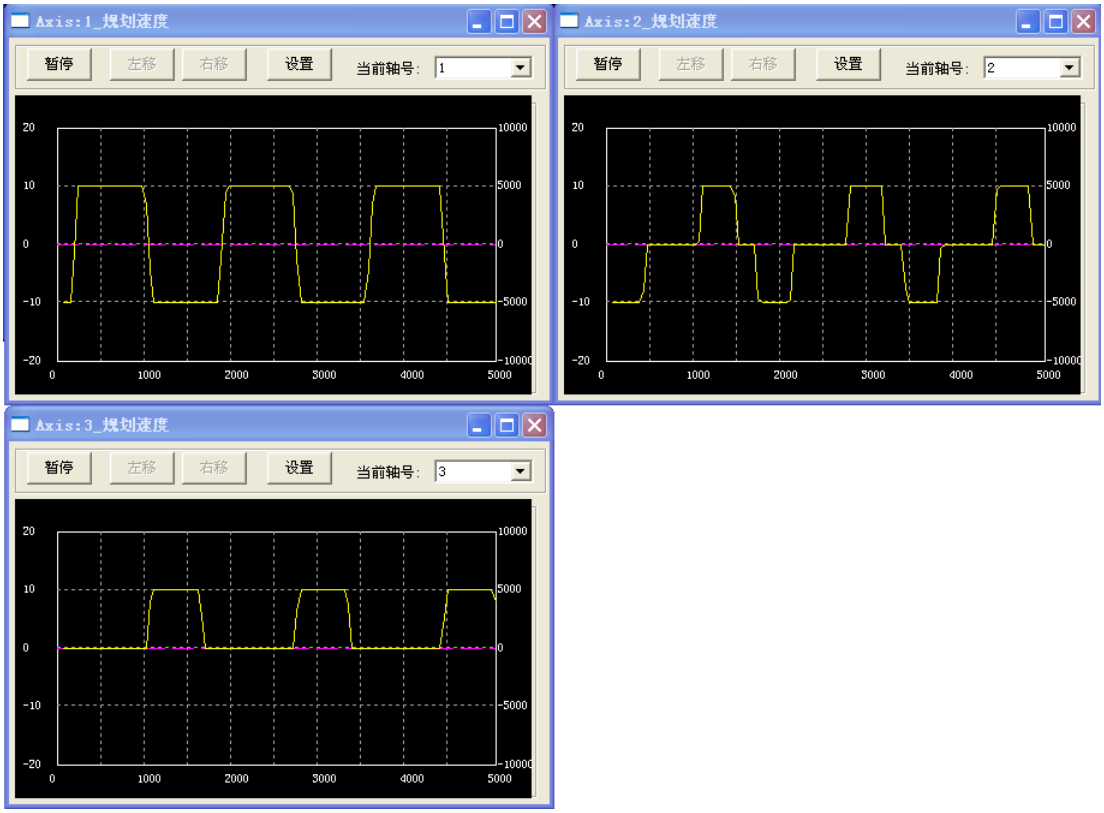

图 6-1 组合运动时序图

运动程序代码如下：

```
//-----------------------------------------------
// 组合运动
//-----------------------------------------------

int pieceArrival;        //物料轴到达标志
int pieceStart;              //物料轴启动标志
int glueStart;           //点胶轴启动标志
int armStart;            //摆臂轴启动标志

// 摆臂轴运动
 void ArmMotion(short arm)
 {
      long armStep;
      short cc;
       long clock;
       long sts;
       short axis;

 lbl_loop:
      axis=arm-1;
      axis=1<<axis;
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
    // 等待摆臂启动标志
lbl_wait_for_arm_start:
    cc = !armStart;
    if(cc) goto lbl_wait_for_arm_start;

    // 清除摆臂启动标志
    armStart = 0;

    // 摆臂轴运动
    GT_SetPos(arm,armStep);
    GT_Update(axis);

    // 等待摆臂轴运动停止
lbl_wait_for_arm_stop:
    GT_GetSts(arm,&sts,1,&clock);
    cc = sts & 0x400;
    if(cc) goto lbl_wait_for_arm_stop;

    // 摆臂轴返回
    GT_SetPos(arm,0);
    GT_Update(axis);

    // 置起点胶轴启动标志
    glueStart = 1;

    // 置起物料轴启动标志
    pieceStart = 1;

    // 等待摆臂轴运动停止
lbl_wait_for_arm_return:
    GT_GetSts(arm,&sts,1,&clock);
    cc = sts & 0x400;
    if(cc) goto lbl_wait_for_arm_return;

     goto lbl_loop;
}

// 点胶轴运动
void GlueMotion(short glue)
{
    long glueStep;
    short cc;
     long clock;
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
        long sts;
        short axis;

lbl_loop:
        axis=glue-1;
        axis=1<<axis;

        // 等待点胶轴启动标志
lbl_wait_for_glue_start:
        cc = !glueStart;
        if (cc) goto lbl_wait_for_glue_start;

        // 清除点胶轴启动标志
        glueStart = 0;

        // 点胶轴运动
        GT_SetPos(glue,glueStep);
        GT_Update(axis);

        // 等待点胶轴运动停止
lbl_wait_for_glue_stop:
        GT_GetSts(glue,&sts,1,&clock);
        cc = sts & 0x400;
        if(cc) goto lbl_wait_for_glue_stop;

        // 等待物料轴到达
lbl_wait_for_piece_arrival:
        cc = !pieceArrival;
        if(cc) goto lbl_wait_for_piece_arrival;

        // 点胶轴返回
        GT_SetPos(glue,0);
        GT_Update(axis);

        // 置起摆臂轴启动标志
        armStart = 1;

        // 等待运动停止
lbl_wait_for_glue_return:
        GT_GetSts(glue,&sts,1,&clock);
        cc = sts & 0x400;
        if(cc) goto lbl_wait_for_glue_return;

        goto lbl_loop;
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
}

// 物料轴运动
void PieceMotion(short piece)
{
    long pieceStep;
    short cc;
     long clock;
     long sts;
     long pos;
     short axis;
     pos = 0;

lbl_loop:
    axis=piece-1;
    axis=1<<axis;

    // 等待启动物料标志
lbl_wait_for_piece_start:
    cc = !pieceStart;
    if (cc) goto lbl_wait_for_piece_start;

    // 清除物料轴启动标志
     pieceStart = 0;

    // 清除物料轴到达标志
    pieceArrival = 0;

     // 启动物料轴运动
     pos = pos + pieceStep;
     GT_SetPos(piece,pos);
     GT_Update(axis);

    // 等待物料轴运动停止
lbl_wait_for_piece_stop:
     GT_GetSts(piece,&sts,1,&clock);
     cc = sts & 0x400;
     if(cc) goto lbl_wait_for_piece_stop;

    // 置起物料轴到达标志
     pieceArrival = 1;

     goto lbl_loop;
}
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

应用程序负责编译、下载、初始化、启动运动程序。代码如下：

```
PROGRAM MultiMotion
VAR
    ARMAxis:INT:=1;
    GLUEAxis:INT:=2;
    PIECEAxis:INT:=3;
    ARM_VEL:LREAL:=10;
    GLUE_VEL:LREAL:=10;
    PIECE_VEL:LREAL:=10;
    ARM_STEP:DINT:=6000;
    GLUE_STEP:DINT:=4000;
    PIECE_STEP:DINT:=8000;

    rtn:INT;
    trap:TTrapPrm;
    compile:TCompileInfo;
    armMotion,glueMotion,pieceMotion:INT;
    pieceArrival,pieceStart,glueStart,armStart:TVarInfo;
    arm,armStep,glue,glueStep,piece,pieceStep:TVarInfo;
    value:DINT;
    prfPos:ARRAY [1..8] OF DINT;

    cmpfilename:STRING:='led.c';
    dlfilename:STRING:='led.bin';
    funname:STRING;
    varname:STRING;

    First: BOOL:=TRUE;
END_VAR
-----------------------------------------------------------------------------------------
IF First THEN
    (*复位运动控制器*)
    rtn:=GT_Reset();

    (*清除报警和限位*)
    rtn:=GT_ClrSts(1,8);

    (*设置 ARM 运动参数*)
    rtn:=GT_PrfTrap(ARMAxis);
    rtn:=GT_GetTrapPrm(ARMAxis,ADR(trap));
    trap.acc:=0.25;
    trap.dec:=0.25;
    rtn:=GT_SetTrapPrm(ARMAxis,ADR(trap));
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
rtn:=GT_SetVel(ARMAxis,ARM_VEL);
rtn:=GT_Update(SHL(DWORD#1,ARMAxis-1));

(*设置 GLUE 运动参数*)
rtn:=GT_PrfTrap(GLUEAxis);
rtn:=GT_GetTrapPrm(GLUEAxis, ADR(trap));
trap.acc:=0.25;
trap.dec:=0.25;
rtn:=GT_SetTrapPrm(GLUEAxis, ADR(trap));
rtn:=GT_SetVel(GLUEAxis, GLUE_VEL);
rtn:=GT_Update(SHL(DWORD#1,GLUEAxis-1));

(*设置 PIECE 运动参数*)
rtn:=GT_PrfTrap(PIECEAxis);
rtn:=GT_GetTrapPrm(PIECEAxis, ADR(trap));
trap.acc:=0.25;
trap.dec:=0.25;
rtn:=GT_SetTrapPrm(PIECEAxis, ADR(trap));
rtn:=GT_SetVel(PIECEAxis, PIECE_VEL);
rtn:=GT_Update(SHL(DWORD#1,PIECEAxis-1));

(*编译运动程序*)
(*编译成功后生成扩展名为.bin 和.ini 的文件*)
(*必须保证 error.ini 文件位于工程文件夹中*)
rtn:=GT_Compile(ADR(cmpfilename), ADR(compile));

(*下载运动程序*)
rtn:= GT_Download(ADR(dlfilename));

(*获取函数 ID*)
funname:='armMotion';
rtn:= GT_GetFunId(ADR(funname), ADR(armMotion));
funname:='glueMotion';
rtn:= GT_GetFunId(ADR(funname), ADR(glueMotion));
funname:='pieceMotion';
rtn:= GT_GetFunId(ADR(funname),ADR(pieceMotion));

(*获取全局变量 ID*)
varname:='pieceArrival';
rtn:= GT_GetVarId(0, ADR(varname), ADR(pieceArrival));
varname:='pieceStart';
rtn:= GT_GetVarId(0, ADR(varname), ADR(pieceStart));
varname:='glueStart';
rtn:= GT_GetVarId(0, ADR(varname), ADR(glueStart));
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
varname:='pieceStart';
rtn:= GT_GetVarId(0, ADR(varname), ADR(pieceStart));

(*获取局部变量 ID*)
funname:='ArmMotion';
varname:='arm';
rtn:= GT_GetVarId(ADR(funname), ADR(varname), ADR(arm));
varname:='armStep';
rtn:= GT_GetVarId(ADR(funname), ADR(varname), ADR(armStep));

funname:='GlueMotion';
varname:='glue';
rtn:= GT_GetVarId(ADR(funname), ADR(varname), ADR(glue));
varname:='glueStep';
rtn:= GT_GetVarId(ADR(funname), ADR(varname), ADR(glueStep));

funname:='PieceMotion';
varname:='piece';
rtn:= GT_GetVarId(ADR(funname), ADR(varname), ADR(piece));
varname:='pieceStep';
rtn:= GT_GetVarId(ADR(funname), ADR(varname), ADR(pieceStep));

(*绑定线程，函数，数据页*)
rtn:= GT_Bind(0, armMotion, 0);
rtn:= GT_Bind(1, glueMotion, 1);
rtn:= GT_Bind(2, pieceMotion, 2);

(*初始化运动程序的全局变量*)
value:=0;
rtn:= GT_SetVarValue(-1, ADR(pieceArrival), ADR(value), 1);
rtn:= GT_SetVarValue(-1, ADR(armStart), ADR(value), 1);

value:=1;
rtn:= GT_SetVarValue(-1, ADR(pieceStart), ADR(value), 1);
rtn:= GT_SetVarValue(-1, ADR(glueStart), ADR(value), 1);

(*初始化运动程序局部变量*)
value:=ARMAxis;
rtn:= GT_SetVarValue(0, ADR(arm), ADR(value), 1);
value:=ARM_STEP;
rtn:= GT_SetVarValue(0, ADR(armStep), ADR(value), 1);

value:=GLUEAxis;
rtn:= GT_SetVarValue(1, ADR(glue), ADR(value), 1);
```


上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

```
        value:=GLUE_STEP;
        rtn:= GT_GetVarValue(1, ADR(glueStep), ADR(value), 1);

        value:=PIECEAxis;
        rtn:= GT_SetVarValue(2, ADR(piece), ADR(value), 1);
        value:=PIECE_STEP;
        rtn:= GT_SetVarValue(2, ADR(pieceStep), ADR(value), 1);

        (*启动线程*)
        rtn:= GT_RunThread(0);
        rtn:= GT_RunThread(1);
        rtn:= GT_RunThread(2);

        First:=FALSE;
    END_IF

    GT_GetPrfPos(1, ADR(prfpos), 8 ,0);
```

6.3 语言元素

1. 数据类型

支持整型和浮点型 2 种数据类型。

- 整型 32 位，取值范围是-2,147,483,648 ~ 2,147,483,647。
- 浮点型采用定点格式，32 位整数，16 位小数。所能表示的最小精度为 (1/2)^16=0.0000152587890625。

2. 常量

可以在程序中直接使用立即数和宏。立即数可以是 10 进制整数、16 进制整数和浮点数。

3. 变量

可以声明局部变量和全局变量。每个函数最多可声明 1024 个局部变量。全局变量最多可声明 1024 个。整型类型说明符为 int。浮点型类型说明符为 double。

4. 数组

支持一维数组，支持常量下标索引和变量下标索引。

不支多维数组，不支持用数组元素进行下标索引。

5. 函数

函数可以定义返回值类型和输入形参类型。

不支持在函数中调用自定义函数，但是可以调用 GT 运动控制指令。

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

6. 数据类型转换

支持强制数据类型转换。强制数据类型转换符有 int,double。

- 数据类型转换符必须加括号，如 a=(int)b
- 数据类型转换不会改变变量本身的数据类型定义

7. 运算指令

支持算术运算、逻辑运算、关系运算、位运算，语法规则和 C 语言相同，但是不支持复杂表达式，只能使用 2 个操作数进行运算，而且这 2 个操作数的数据类型必须相同。

注意：由于运动程序中的浮点数据类型只有 16 位小数精度，请不要在运动程序中进行高精度浮点运算。

8. 算数运算

用于各类数值运算。包括加(+)、减(-)、乘(*)、除(/)、求余(或称模运算，%)共五种。

9. 逻辑运算

逻辑运算包括与(&&)、或(||)、非(!)三种。参数可以是整型变量、或者整型常数。

10. 关系运算

关系运算符用于比较运算。包括大于(>)、小于(<)、等于(==)、 大于等于(>=)、小于等于(<=)和不等于(!=)六种。参与比较的参数类型必须一致。

11. 位运算

参与运算的量，按二进制位进行运算。包括位与(&)、位或(|)、位非(~)、位异或(^)、左移(<<)、右移(>>)六种。

6.4 流程控制

支持条件跳转、条件返回，语法规则和 C 语言相同。

1) 条件跳转如下所示：

```
if(var) goto label;
```

当条件变量 var 非 0 时，跳转到标记为 label 的指令。

不支持表达式作为判断条件。

2) 条件返回如下所示：

```
if(var) return value;
```

当条件变量 var 非 0 时，程序返回，返回值为 value。

不支持表达式作为判断条件。

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

第7章 指令详细说明

提示	以下表格中的“章节页码”即为此指令在章节中的位置。可以使用“超级链接”进行索引。“指令示例”即为与此指令相关的例程，可以使用“超级链接”进行索引。

指令 1 GT_AlarmOff

指令原型	short GT_AlarmOff(short axis)		
指令说明	控制相应轴驱动报警信号无效。		
指令类型	立即指令，调用后立即生效。	章节页码	12
指令参数	该指令共有 1 个参数，参数的详细信息如下。		
axis	控制轴号。		
指令返回值	若返回值为 1：请检查相应轴在配置文件中是否已经配置了报警无效。 其他返回值：请参照指令返回值列表。		
相关指令	GT_AlarmOn		
指令示例	无		

指令 2 GT_AlarmOn

指令原型	short GT_AlarmOn(short axis)		
指令说明	控制轴驱动报警信号有效。		
指令类型	立即指令，调用后立即生效。	章节页码	12
指令参数	该指令共有 1 个参数，参数的详细信息如下。		
axis	控制轴号。		
指令返回值	若返回值为 1：请检查相应轴在配置文件中是否已经配置了报警无效。 其他返回值：请参照指令返回值列表。		
相关指令	GT_AlarmOff		
指令示例	无		

指令 3 GT_ArcXYC

指令原型	short GT_ArcXYC(short crd, long x, long y, double xCenter, double yCenter, short circleDir, double synVel, double synAcc, double velEnd=0, short fifo=0)		
指令说明	XY 平面圆弧插补。使用圆心描述方法描述圆弧。		
指令类型	缓存区指令。	章节页码	14
指令参数	该指令共有 10 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
x	圆弧插补 x 轴的终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
y	圆弧插补 y 轴的终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
xCenter	圆弧插补的圆心 x 方向相对于起点位置的偏移量。		
yCenter	圆弧插补的圆心 y 方向相对于起点位置的偏移量。		
circleDir	圆弧的旋转方向。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

	0：顺时针圆弧。 1：逆时针圆弧。
synVel	插补段的目标合成速度。取值范围：(0, 32767)，单位：pulse/ms。
synAcc	插补段的合成加速度。取值范围：(0, 32767)，单位：pulse/ms^2。
velEnd	插补段的终点速度。取值范围：[0, 32767)，单位：pulse/ms。该值只有在没有使用前瞻预处理功能时才有意义，否则该值无效。默认值为：0。
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。
相关指令	无。
指令示例	**例程 5-3 坐标系运动的直圆弧插补**

指令 4　GT_ArcXYR

指令原型	short GT_ArcXYR (short crd, long x, long y, double radius, short circleDir, double synVel, double synAcc, double velEnd=0, short fifo=0)		
指令说明	XY 平面圆弧插补。以终点位置和半径为输入参数。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 9 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
x	圆弧插补 x 轴的终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
y	圆弧插补 y 轴的终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
radius	圆弧插补的圆弧半径值。取值范围：[-1073741823, 1073741823]，单位：pulse。 半径为正时，表示圆弧为小于等于 180°圆弧。 半径为负时，表示圆弧为大于 180°圆弧。 半径描述方式不能用来描述整圆。		
circleDir	圆弧的旋转方向。 0：顺时针圆弧。 1：逆时针圆弧。		
synVel	插补段的目标合成速度。取值范围：(0, 32767)，单位：pulse/ms。		
synAcc	插补段的合成加速度。取值范围：(0, 32767)，单位：pulse/ms^2。		
velEnd	插补段的终点速度。取值范围：[0, 32767)，单位：pulse/ms。该值只有在没有使用前瞻预处理功能时才有意义，否则该值无效。默认值为：0		
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。		

相关指令	无。
指令示例	**例程 5-3 坐标系运动的直圆弧插补**

指令 5　GT_ArcYZC

指令原型	short GT_ArcYZC (short crd, long y, long z, double yCenter, double zCenter, short circleDir, double synVel, double synAcc, double velEnd=0, short fifo=0)		
指令说明	YZ 平面圆弧插补。以终点位置和圆心位置为输入参数。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 10 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
y	圆弧插补 y 轴的终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
z	圆弧插补 z 轴的终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
yCenter	圆弧插补的圆心 y 方向相对于起点位置的偏移量。		
zCenter	圆弧插补的圆心 z 方向相对于起点位置的偏移量。		
circleDir	圆弧的旋转方向。 0：顺时针圆弧。 1：逆时针圆弧。		
synVel	插补段的目标合成速度。取值范围：(0, 32767)，单位：pulse/ms。		
synAcc	插补段的合成加速度。取值范围：(0, 32767)，单位：pulse/ms^2。		
velEnd	插补段的终点速度。取值范围：[0, 32767)，单位：pulse/ms。该值只有在没有使用前瞻预处理功能时才有意义，否则该值无效。默认值为：0。		
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	**无。**		

指令 6　GT_ArcYZR

指令原型	short GT_ArcYZR (short crd, long y, long z, double radius, short circleDir, double synVel, double synAcc, double velEnd=0, short fifo=0)		
指令说明	YZ 平面圆弧插补。以终点位置和半径为输入参数。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 9 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
y	圆弧插补 y 轴的终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
z	圆弧插补 z 轴的终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
radius	圆弧插补的圆弧半径值。取值范围：[-1073741823, 1073741823]，单位：pulse。 半径为正时，表示圆弧为小于等于 180°圆弧。 半径为负时，表示圆弧为大于 180°圆弧。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

	半径描述方式不能用来描述整圆。
circleDir	圆弧的旋转方向。 0：顺时针圆弧。 1：逆时针圆弧。
synVel	插补段的目标合成速度。取值范围：(0, 32767)，单位：pulse/ms。
synAcc	插补段的合成加速度。取值范围：(0, 32767)，单位：pulse/ms^2。
velEnd	插补段的终点速度。取值范围：[0, 32767)，单位：pulse/ms。该值只有在没有使用前瞻预处理功能时才有意义，否则该值无效。默认值为：0。
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。
相关指令	无。
指令示例	**无。**

指令 7　GT_ArcZXC

指令原型	short GT_ArcZXC (short crd, long z, long x, double zCenter, double xCenter, short circleDir, double synVel, double synAcc, double velEnd=0, short fifo=0)		
指令说明	ZX 平面圆弧插补。以终点位置和圆心位置为输入参数。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 10 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
z	圆弧插补 z 轴的终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
x	圆弧插补 x 轴的终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
zCenter	圆弧插补的圆心 z 方向相对于起点位置的偏移量。		
xCenter	圆弧插补的圆心 x 方向相对于起点位置的偏移量。		
circleDir	圆弧的旋转方向。 0：顺时针圆弧。 1：逆时针圆弧。		
synVel	插补段的目标合成速度。取值范围：(0, 32767)，单位：pulse/ms。		
synAcc	插补段的合成加速度。取值范围：(0, 32767)，单位：pulse/ms^2。		
velEnd	插补段的终点速度。取值范围：[0, 32767)，单位：pulse/ms。该值只有在没有使用前瞻预处理功能时才有意义，否则该值无效。默认值为：0。		
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

相关指令	无。
指令示例	**无。**

指令 8　GT_ArcZXR

指令原型	short GT_ArcZXR (short crd, long z, long x, double radius, short circleDir, double synVel, double synAcc, double velEnd=0, short fifo=0)		
指令说明	ZX 平面圆弧插补。以终点位置和半径为输入参数。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 9 个参数，参数的详细信息如下。		
Crd	坐标系号。正整数，取值范围：[1, 2]。		
Z	圆弧插补 z 轴的终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
X	圆弧插补 x 轴的终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
radius	圆弧插补的圆弧半径值。取值范围：[-1073741823, 1073741823]，单位：pulse。 半径为正时，表示圆弧为小于等于 180°圆弧。 半径为负时，表示圆弧为大于 180°圆弧。 半径描述方式不能用来描述整圆。		
circleDir	圆弧的旋转方向。 0：顺时针圆弧。 1：逆时针圆弧。		
synVel	插补段的目标合成速度。取值范围：(0, 32767)，单位：pulse/ms。		
synAcc	插补段的合成加速度。取值范围：(0, 32767)，单位：pulse/ms^2。		
velEnd	插补段的终点速度。取值范围：[0, 32767)，单位：pulse/ms。该值只有在没有使用前瞻预处理功能时才有意义，否则该值无效。默认值为：0。		
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	**无。**		

指令 9　GT_Bind

指令原型	short GT_Bind(short thread, short funId, short page)		
指令说明	绑定线程、函数、数据页。		
指令类型	立即指令，调用后立即生效。	**章节页码**	51
指令参数	该指令共有 3 个参数，参数的详细信息如下。		
thread	线程编号，取值范围：[0, 31]。		
funId	函数标识，可以调用 GT_GetFunId 查询。		
page	数据页编号，取值范围：[0, 31]。		
指令返回值	若返回值为 1：请检查绑定的线程号是否已经有线程绑定并正在运行。 其他返回值：请参照指令返回值列表。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

相关指令	无。
指令示例	**例程 6-1 运动程序单线程累加求和**

指令 10 GT_BufDA

指令原型	short GT_BufDA (short crd, short chn, short daValue, short fifo=0)		
指令说明	缓存区内输出 DA 值。		
指令类型	缓存区指令。	章节页码	14
指令参数	该指令共有 4 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
chn	模拟量输出的通道号。取值范围：[1, 8]。		
daValue	模拟量输出的值。取值范围：[-32768, 32767]，其中：-32768 对应-10V，32767 对应+10V。		
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	无。		

指令 11 GT_BufDelay

指令原型	short GT_BufDelay (short crd, unsigned short delayTime, short fifo=0)		
指令说明	缓存区内延时设置指令。		
指令类型	缓存区指令。	章节页码	14
指令参数	该指令共有 3 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
delayTime	延时时间。取值范围：[0, 16383]，单位：ms。		
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 5-2 坐标系运动的直线插补**		

指令 12 GT_BufGear

指令原型	short GT_BufGear (short crd, short gearAxis, long pos, short fifo=0)		
指令说明	实现刀向跟随功能，启动某个轴跟随运动。		
指令类型	缓存区指令。	章节页码	14

指令参数	该指令共有 4 个参数，参数的详细信息如下。
crd	坐标系号。正整数，取值范围：[1, 2]。
gearAxis	需要进行跟随运动的轴号，取值范围：[1, 8]。该轴不能处于坐标系中。
pos	跟随运动的位移量，单位：pulse。
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。
相关指令	无。
指令示例	**例程 5-6 插补过程中的跟随运动**

指令 13 GT_BufIO

指令原型	short GT_BufIO (short crd, unsigned short doType, unsigned short doMask, unsigned short doValue, short fifo=0)		
指令说明	缓存区内数字量 IO 输出设置指令。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 5 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
doType	数字量输出的类型。 MC_ENABLE(该宏定义为 10)：输出驱动器使能。 MC_CLEAR(该宏定义为 11)：输出驱动器报警清除。 MC_GPO(该宏定义为 12)：输出通用输出。		
doMask	从 bit0~bit15 按位表示指定的数字量输出是否有操作。 0：该路数字量输出无操作。1：该路数字量输出有操作。		
doValue	从 bit0~bit15 按位表示指定的数字量输出的值。		
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 5-2 坐标系运动的直线插补**		

指令 14 GT_BufLmtsOff

指令原型	short GT_BufLmtsOff (short crd, short axis, short limitType, short fifo=0)		
指令说明	缓存区内无效限位开关。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 4 个参数，参数的详细信息如下。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

crd	坐标系号。正整数，取值范围：[1, 2]。
axis	需要将限位无效的轴的编号。取值范围：[1, 8]。
limitType	需要无效的限位类型。 MC_LIMIT_POSITIVE(该宏定义为 0)：需要将该轴的正限位无效。 MC_LIMIT_NEGATIVE(该宏定义为 1)：需要将该轴的负限位无效。 -1：需要将该轴的正限位和负限位都无效，默认为该值。
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。
相关指令	GT_BufLmtsOn
指令示例	无。

指令 15 GT_BufLmtsOn

指令原型	short GT_BufLmtsOn(short crd, short axis, short limitType, short fifo=0)		
指令说明	缓存区内有效限位开关。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 4 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
axis	需要将限位有效的轴的编号，取值范围：[1, 8]。		
limitType	需要有效的限位类型。 MC_LIMIT_POSITIVE(该宏定义为 0)：需要将该轴的正限位设置为有效。 MC_LIMIT_NEGATIVE(该宏定义为 1)：需要将该轴的负限位设置为有效。 -1：需要将该轴的正限位和负限位都设置为有效，默认为该值。		
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。		
相关指令	GT_BufLmtsOff		
指令示例	无。		

指令 16 GT_BufMove

指令原型	short GT_BufMove (short crd, short moveAxis, long pos, double vel, double acc, short modal, short fifo=0)		
指令说明	实现刀向跟随功能，启动某个轴点位运动。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 7 个参数，参数的详细信息如下。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

crd	坐标系号。正整数，取值范围：[1, 2]。
moveAxis	需要进行点位运动的轴号，取值范围：[1, 8]。该轴不能处于坐标系中。
pos	点位运动的目标位置，单位：pulse。
vel	点位运动的目标速度，单位：pulse/ms。
acc	点位运动的加速度，单位：pulse/ms^2。
modal	点位运动的模式。 0：该指令为非模态指令，即不阻塞后续的插补缓存区指令的执行。 1：该指令为模态指令，将会阻塞后续的插补缓存区指令的执行。
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。
相关指令	无。
指令示例	**例程 5-5 插补过程中的点位运动**

指令 17 GT_BufSetStopIo

指令原型	short GT_BufSetStopIo (short crd, short axis, short stopType, short inputType, short inputIndex, short fifo=0)		
指令说明	缓存区内设置 axis 的停止 IO 信息。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 6 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
axis	需要设置停止 IO 信息的轴的编号。取值范围：[1, 8]。		
stopType	需要设置停止 IO 信息的停止类型。 0：紧急停止类型。 1：平滑停止类型。		
inputType	设置的数字量输入的类型。 MC_LIMIT_POSITIVE(该宏定义为 0)：正限位。 MC_LIMIT_NEGATIVE(该宏定义为 1)：负限位。 MC_ALARM(该宏定义为 2)：驱动报警。 MC_HOME(该宏定义为 3)：原点开关。 MC_GPI(该宏定义为 4)：通用输入。 MC_ARRIVE(该宏定义为 5)：电机到位信号。		
inputIndex	设置的数字量输入的索引号，取值范围根据 inputType 的取值而定。 当 inputType= MC_LIMIT_POSITIVE 时，取值范围：[1, 8]。 当 inputType= MC_LIMIT_NEGATIVE 时，取值范围：[1, 8]。 当 inputType= MC_ALARM 时，取值范围：[1, 8]。 当 inputType= MC_HOME 时，取值范围：[1, 8]。 当 inputType= MC_GPI 时，取值范围：[1, 16]。 当 inputType= MC_ARRIVE 时，取值范围：[1, 4]。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。
相关指令	无。
指令示例	无。

指令 18 GT_CrdClear

指令原型	short GT_CrdClear(short crd, short fifo)		
指令说明	清除插补缓存区内的插补数据。		
指令类型	立即指令，调用后立即生效。	**章节页码**	14
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
fifo	所要清除的插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 5-2 坐标系运动的直线插补**		

指令 19 GT_CrdData

指令原型	short GT_CrdData (short crd, TCrdData *pCrdData, short fifo=0)		
指令说明	用于在使用前瞻时。调用该指令表示后续没有新的数据，将会一次性把前瞻缓存区的数据压入运动缓存区。		
指令类型	立即指令，调用后立即生效。	**章节页码**	14
指令参数	该指令共有 3 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
pCrdData	只能设置为：NULL。		
fifo	插补缓存区号。正整数，取值范围：[0, 1]。默认值为：0。		
指令返回值	若返回值为非零值，说明前瞻缓存区还有数据没有被压入运动缓存区，而运动缓存区没有空间了。此时需要检查运动缓存区的空间（调用 GT_CrdSpace 检查）。当检查运动缓存区有空间时，再次调用 GT_CrdData 指令，直至返回值为 0 时，前瞻缓存区的数据才被完全送入运动缓存区。		
相关指令	无。		
指令示例	**例程 5-4 使用前瞻预处理**		

指令 20 GT_CrdSpace

指令原型	short GT_CrdSpace (short crd, long *pSpace, short fifo=0)

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

指令说明	查询插补缓存区剩余空间。		
指令类型	立即指令，调用后立即生效。	**章节页码**	14
指令参数	该指令共有 3 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
pSpace	读取插补缓存区中的剩余空间。		
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1：检查当前坐标系是否映射了相关轴。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 5-2 坐标系运动的直线插补**		

指令 21 GT_CrdStart

指令原型	short GT_CrdStart (short mask, short option)		
指令说明	启动插补运动。		
指令类型	立即指令，调用后立即生效。	**章节页码**	14
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
mask	从 bit0~bit1 按位表示需要启动的坐标系。 bit0 对应坐标系 1，bit1 对应坐标系 2。 0：不启动该坐标系，1：启动该坐标系。		
option	从 bit0~bit1 按位表示坐标系需要启动的缓存区的编号。 bit0 对应坐标系 1，bit1 对应坐标系 2。 0：启动坐标系中 FIFO0 的运动，1：启动坐标系中 FIFO1 的运动。		
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 若使用了辅助 fifo1 运动，检查当前坐标系位置没有恢复到 fifo0 断点坐标系位置。 (3) 检查参数设置是否启动了坐标系。 (4) 检查坐标系是否在运动。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 5-2 坐标系运动的直线插补**		

指令 22 GT_CrdStatus

指令原型	short GT_CrdStatus (short crd, short *pRun, long *pSegment, short fifo=0)		
指令说明	查询插补运动坐标系状态。		
指令类型	立即指令，调用后立即生效。	**章节页码**	14
指令参数	该指令共有 4 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
pRun	读取插补运动状态。0：该坐标系的该 FIFO 没有在运动；1：该坐标系的该 FIFO 正在进行插补运动。		
pSegment	读取当前已经完成的插补段数。当重新建立坐标系或者调用 GT_CrdClear 指令后，该值会被清零。		
fifo	所要查询运动状态的插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1：检查当前坐标系是否映射了相关轴。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

	其他返回值：请参照指令返回值列表。
相关指令	无。
指令示例	**例程 5-2 坐标系运动的直线插补**

指令 23 GT_Download

指令原型	short GT_Download(char *pFileName)		
指令说明	下载运动程序到运动控制器。注：文件包含的线程数不能大于 32。		
指令类型	立即指令，调用后立即生效。	**章节页码**	51
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
pFileName	下载到运动控制器的运动程序文件名		
指令返回值	若返回值为 2007： (1) 请检查文件名是否太长。 (2) 请检查需要下载的文件是否存在。 若返回值为 2008：表示打开文件失败。 (1) 请检查文件是否损坏，编译是否成功。 (2) 请检查 ini 和 bin 文件是否在同一根目录。 若返回值为 7：请检查线程数量是否大于 32。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 6-1 运动程序单线程累加求和**		

指令 24 GT_EncScale

指令原型	short GT_EncScale(short axis, short alpha, short beta)		
指令说明	设置控制轴的编码器当量变换值。		
指令类型	立即指令，调用后立即生效。	**章节页码**	12
指令参数	该指令共有 3 个参数，参数的详细信息如下。		
axis	控制轴号。		
alpha	规划器当量的alpha值，取值范围：(-32767, 0)和(0, 32767)。		
beta	规划器当量的beta值，取值范围：(-32767, 0)和(0, 32767)。		
指令返回值	若返回值为 1：若当前轴在规划运动，请调用 GT_Stop()停止运动再调用该指令。 其他返回值：请参照指令返回值列表。		
相关指令	GT_ProfileScale		
指令示例	无。		

指令 25 GT_GetCrdPos

指令原型	short GT_GetCrdPos (short crd, double *pPos)		
指令说明	查询该坐标系的当前坐标位置值。获取的坐标值可能和规划位置不一致，取决于建立坐标系的原点是否为零。		
指令类型	立即指令，调用后立即生效。	**章节页码**	14
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
pPos	读取的坐标系的坐标值。单位：pulse。该参数应该为一个数组首元素的指针，数组的元		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

	素个数取决于该坐标系的维数。
指令返回值	请参照指令返回值列表
相关指令	GT_GetCrdVel
指令示例	无。

指令 26 GT_GetCrdPrm

指令原型	short GT_GetCrdPrm (short crd, TCrdPrm *pCrdPrm)		
指令说明	查询坐标系参数。		
指令类型	立即指令，调用后立即生效。	**章节页码**	14
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
pCrdPrm	读取坐标系的相关参数 结构体的成员含义参照 GT_SetCrdPrm 指令说明。		
指令返回值	请参照指令返回值列表.		
相关指令	GT_SetCrdPrm		
指令示例	无。		

指令 27 GT_GetCrdStopDec

指令原型	short GT_GetCrdStopDec (short crd, double *pDecSmoothStop, double *pDecAbruptStop)		
指令说明	查询插补运动平滑停止、急停合成加速度。		
指令类型	立即指令，调用后立即生效。	**章节页码**	14
指令参数	该指令共有 3 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
pDecSmoothStop	查询坐标系合成平滑停止加速度，单位：pulse/ms^2。		
pDecAbruptStop	查询坐标系合成急停加速度，单位：pulse/ms^2。		
指令返回值	若返回值为 1：检查当前坐标系是否映射了相关轴。 其他返回值：请参照指令返回值列表。		
相关指令	GT_SetCrdStopDec		
指令示例	无。		

指令 28 GT_GetCrdVel

指令原型	short GT_GetCrdVel(short crd, double *pSynVel)		
指令说明	查询该坐标系的当前坐标速度值。		
指令类型	立即指令，调用后立即生效。	**章节页码**	14
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
pSynVel	读取的坐标系的合成速度值，单位：pulse/ms。		
指令返回值	请参照指令返回值列表。		
相关指令	GT_GetCrdPos		
指令示例	无。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

指令 29 GT_GetFunId

指令原型	short GT_GetFunId (char *pFunName, short *pFunId)		
指令说明	读取运动程序中函数的标识。		
指令类型	立即指令，调用后立即生效。	**章节页码**	51
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
pFunName	运动程序函数名称。		
pFunId	根据运动程序函数名称查询函数标识。		
指令返回值	若返回值为 1，2007 或者 2008： 请检查重新检查 GT_Download 是否调用成功。 若失败，请根据 GT_Download 返回值提示操作，直至成功。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 6-1 运动程序单线程累加求和**		

指令 30 GT_GetPvtLoop

指令原型	short GT_GetPvtLoop(short profile, long *pLoopCount, long *pLoop)		
指令说明	查询 PVT 运动模式循环次数。		
指令类型	立即指令，调用后立即生效。	**章节页码**	30
指令参数	该指令共有 3 个参数，参数的详细信息如下。		
profile	规划轴号。		
pLoopCount	查询已经循环的次数。		
pLoop	查询循环执行的总次数。		
指令返回值	若返回值为 1：请检查当前轴是否为 PVT 模式，若不是，请先调用 GT_PrfPvt 将当前轴设置为 PVT 模式。 其他返回值：请参照指令返回值列表。		
相关指令	GT_SetPvtLoop		
指令示例	无。		

指令 31 GT_GetRemainderSegNum

指令原型	short GT_GetRemainderSegNum(short crd, long *pSegment, short fifo=0)		
指令说明	读取未完成的插补段段数。		
指令类型	立即指令，调用后立即生效。	**章节页码**	14
指令参数	该指令共有 3 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
pSegment	读取的剩余插补段的段数。		
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1：检查当前坐标系是否映射了相关轴。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	无。		

指令 32 GT_GetStopDec

指令原型	short GT_GetStopDec(short profile, double *pDecSmoothStop, double *pDecAbruptStop)		
指令说明	读取平滑停止减速度和急停减速度。		
指令类型	立即指令，调用后立即生效。	**章节页码**	12
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
profile	规划器的编号		
pDecSmoothStop	读取的平滑停止减速度。单位：pulse/ms^2。		
pDecAbruptStop	读取的急停减速度。单位：pulse/ms^2。		
指令返回值	请参照指令返回值列表。		
相关指令	GT_SetStopDec		
指令示例	无。		

指令 33 GT_GetThreadSts

指令原型	short GT_GetThreadSts(short thread, TThreadSts *pThreadSts)		
指令说明	读取线程的状态。		
指令类型	立即指令，调用后立即生效。	**章节页码**	51
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
thread	线程编号。取值范围：[0, 31]。		

pThreadSts	读取线程状态
	typedef struct ThreadSts
	{
	short run; // 运行状态
	short error; // 当前执行的指令返回值
	double result; // 函数返回值
	short line; // 当前执行的指令行号
	} TThreadSts;		

指令返回值	请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 6-1 运动程序单线程累加求和**		

指令 34 GT_GetUserSegNum

指令原型	short GT_GetUserSegNum (short crd, long *pSegment, short fifo=0)		
指令说明	读取自定义插补段段号。		
指令类型	立即指令，调用后立即生效。	**章节页码**	14
指令参数	该指令共有 3 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
pSegment	读取的用户自定义的插补段段号。		
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1：检查当前坐标系是否映射了相关轴。 其他返回值：请参照指令返回值列表。		
相关指令	GT_SetUserSegNum		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

指令示例	无。

指令 35 GT_GetVarId

指令原型	short GT_GetVarId(char *pFunName, char *pVarName, TVarInfo *pVarInfo)		
指令说明	读取运动程序中变量的标识。		
指令类型	立即指令，调用后立即生效。	章节页码	51
指令参数	该指令共有 3 个参数，参数的详细信息如下。		
pFunName	全局变量输入 NULL。 局部变量所在函数的名称。		
pVarName	运动程序变量名称。		
pVarInfo	根据运动程序函数名称和变量名称查询变量标识。		
指令返回值	若返回值为 1，2007 或者 2008： 请检查重新检查 GT_Download 是否调用成功。 若失败，请根据 GT_Download 返回值提示操作，直至成功。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 6-1 运动程序单线程累加求和**		

指令 36 GT_GetVarValue

指令原型	short GT_GetVarValue(short page, TVarInfo *pVarInfo, double *pValue, short count=1)		
指令说明	读取运动程序中变量的值。		
指令类型	立即指令，调用后立即生效。	章节页码	51
指令参数	该指令共有 4 个参数，参数的详细信息如下。		
page	数据页编号。 全局变量为-1。 局部变量取值范围：[0, 31]。		
pVarInfo	需要访问的变量标识。		
pValue	需要读取的变量值。		
count	需要读取的变量值的数量，取值范围：[1, 8]。		
指令返回值	请参照指令返回值列表。		
相关指令	GT_SetVarValue		
指令示例	**例程 6-1 运动程序单线程累加求和**		

指令 37 GT_GpiSns

指令原型	short GT_GpiSns(unsigned short sense)		
指令说明	设置运动控制器数字量输入的电平逻辑。		
指令类型	立即指令，调用后立即生效。	章节页码	12
指令参数	该指令共有 1 个参数，参数的详细信息如下：		
sense	按位表示各数量输入的电平逻辑，从 bit0~bit15，分别对应数字量输入 1 到 16。影响 GT_GetDi 指令读取电平的结果。 0：输入电平不取反，通过 GT_GetDi()指令读取到 0 表示输入低电平，通过 GT_GetDi()		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

	指令读取到 1 表示输入高电平； 1：输入电平取反，通过 GT_GetDi()指令读取到 0 表示输入高电平，通过 GT_GetDi()指令读取到 1 表示输入低电平；
指令返回值	请参照指令返回值列表。
相关指令	无。
指令示例	无。

指令 38 GT_InitLookAhead

指令原型	short GT_InitLookAhead (short crd, short fifo, double T, double accMax, short n, TCrdData *pLookAheadBuf)		
指令说明	初始化插补前瞻缓存区。		
指令类型	立即指令，调用后立即生效。	**章节页码**	14
指令参数	该指令共有 6 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
T	拐弯时间。单位：ms。		
accMax	最大加速度。单位：pulse/ms^2。		
n	前瞻缓存区大小。取值范围：[0, 32767)。		
pLookAheadBuf	前瞻缓存区内存区指针。		
指令返回值	请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 5-4 使用前瞻预处理**		

指令 39 GT_LmtsOff

指令原型	short GT_LmtsOff(short axis, short limitType=-1)		
指令说明	控制轴限位信号无效。		
指令类型	立即指令，调用后立即生效。	**章节页码**	12
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
axis	控制轴号。		
limitType	需要有效的限位类型。 MC_LIMIT_POSITIVE(该宏定义为0)：将该轴的正限位无效。 MC_LIMIT_NEGATIVE(该宏定义为1)：将该轴的负限位无效。 -1：将该轴的正限位和负限位都无效，默认为该值。		
指令返回值	若返回值为 1：请检查相应轴限位报警，配置文件是否已经配置了限位无效。 其他返回值：请参照指令返回值列表。		
相关指令	GT_LmtsOn		
指令示例	**无。**		

指令 40 GT_LmtsOn

指令原型	short GT_LmtsOn(short axis, short limitType=-1)
指令说明	控制轴限位信号有效。

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

指令类型	立即指令，调用后立即生效。	**章节页码**	12
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
axis	控制轴号。		
limitType	需要有效的限位类型。 MC_LIMIT_POSITIVE(该宏定义为0)：将该轴的正限位有效。 MC_LIMIT_NEGATIVE(该宏定义为1)：将该轴的负限位有效。 -1：将该轴的正限位和负限位都有效，默认为该值。		
指令返回值	若返回值为 1：请检查相应轴限位报警，配置文件是否已经配置了限位无效。 其他返回值：请参照指令返回值列表。		
相关指令	GT_LmtsOff		
指令示例	无。		

指令 41 GT_LnXY

指令原型	short GT_LnXY (short crd, long x, long y, double synVel, double synAcc, double velEnd=0, short fifo=0)		
指令说明	XY 平面二维直线插补。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 7 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
x	插补段 x 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
y	插补段 y 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
synVel	插补段的目标合成速度。取值范围：(0, 32767)，单位：pulse/ms。		
synAcc	插补段的合成加速度。取值范围：(0, 32767)，单位：pulse/ms^2。		
velEnd	插补段的终点速度。取值范围：[0, 32767)，单位：pulse/ms。该值只有在没有使用前瞻预处理功能时才有意义，否则该值无效。默认值为：0。		
fifo	插补缓存区号。取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 5-2 坐标系运动的直线插补**		

指令 42 GT_LnXYG0

指令原型	short GT_LnXYG0 (short crd, long x, long y, double synVel, double synAcc, short fifo=0)		
指令说明	二维直线插补，且终点速度始终为 0。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 6 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
x	插补段 x 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

y	插补段 y 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。
synVel	插补段的目标合成速度。取值范围：(0, 32767)，单位：pulse/ms。
synAcc	插补段的合成加速度。取值范围：(0, 32767)，单位：pulse/ms^2。
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。
相关指令	无。
指令示例	无。

指令 43 GT_LnXYZ

指令原型	short GT_LnXYZ (short crd, long x, long y, long z, double synVel, double synAcc, double velEnd=0, short fifo=0)		
指令说明	三维直线插补。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
x	插补段 x 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
y	插补段 y 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
z	插补段 z 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
synVel	插补段的目标合成速度。取值范围：(0, 32767)，单位：pulse/ms。		
synAcc	插补段的合成加速度。取值范围：(0, 32767)，单位：pulse/ms^2。		
velEnd	插补段的终点速度。取值范围：[0, 32767)，单位：pulse/ms。该值只有在没有使用前瞻预处理功能时才有意义，否则该值无效。默认值为：0。		
fifo	插补缓存区号，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	无。		

指令 44 GT_LnXYZA

指令原型	short GT_LnXYZA (short crd, long x, long y, long z, long a, double synVel, double synAcc, double velEnd=0, short fifo=0)		
指令说明	四维直线插补。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 9 个参数，参数的详细信息如下。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

crd	坐标系号。正整数，取值范围：[1, 2]。
x	插补段 x 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。
y	插补段 y 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。
z	插补段 z 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。
a	插补段 a 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。
synVel	插补段的目标合成速度。取值范围：(0, 32767)，单位：pulse/ms。
synAcc	插补段的合成加速度。取值范围：(0, 32767)，单位：pulse/ms^2。
velEnd	插补段的终点速度。取值范围：[0, 32767)，单位：pulse/ms。该值只有在没有使用前瞻预处理功能时才有意义，否则该值无效。默认值为：0。
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。
相关指令	无。
指令示例	无。

指令 45 GT_LnXYZAG0

指令原型	short GT_LnXYZAG0 (short crd, long x, long y, long z, long a, double synVel, double synAcc, short fifo=0)		
指令说明	四维直线插补，且终点速度始终为 0。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 8 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
x	插补段 x 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
y	插补段 y 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
z	插补段 z 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
a	插补段 a 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
synVel	插补段的目标合成速度。取值范围：(0, 32767)，单位：pulse/ms。		
synAcc	插补段的合成加速度。取值范围：(0, 32767)，单位：pulse/ms^2。		
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	无。		

指令 46 GT_LnXYZG0

指令原型	short GT_LnXYZG0 (short crd, long x, long y, long z, double synVel, double synAcc, short fifo=0)		
指令说明	三维直线插补，且终点速度始终为 0。		
指令类型	缓存区指令。	**章节页码**	14
指令参数	该指令共有 7 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
x	插补段 x 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
y	插补段 y 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
z	插补段 z 轴终点坐标值。取值范围：[-1073741823, 1073741823]，单位：pulse。		
synVel	插补段的目标合成速度。取值范围：(0, 32767)，单位：pulse/ms。		
synAcc	插补段的合成加速度。取值范围：(0, 32767)，单位：pulse/ms^2。		
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		
指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	无。		

指令 47 GT_PauseThread

指令原型	short GT_PauseThread(short thread)		
指令说明	暂停正在运行的线程。		
指令类型	立即指令，调用后立即生效。	**章节页码**	51
指令参数	该指令共有 1 个参数，参数的详细信息如下。		
thread	线程编号。取值范围：[0, 31]。		
指令返回值	若返回值为 1： (1) 请检查线程号是否已经绑定。 (2) 请检查相应的数据页中是否超出范围。 其他返回值：请参照指令返回值列表。		
相关指令	GT_RunThread; GT_StopThread		
指令示例	无。		

指令 48 GT_PrfPvt

指令原型	short GT_PrfPvt(short profile)		
指令说明	设置指定轴为 PVT 运动模式。		
指令类型	立即指令，调用后立即生效。	**章节页码**	30
指令参数	该指令共有 1 个参数，参数的详细信息如下。		
profile	规划轴号。		
指令返回值	若返回值为 1：若当前轴在规划运动，请调用 GT_Stop()停止运动再调用该指令。 其他返回值：请参照指令返回值列表。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

相关指令	无。
指令示例	**例程 5-7 PVT 描述方式**

指令 49 GT_ProfileScale

指令原型	short GT_ProfileScale(short axis, short alpha, short beta)		
指令说明	设置控制轴的规划器当量变换值。注意：alpha的绝对值要大于beta的绝对值。		
指令类型	立即指令，调用后立即生效。	章节页码	12
指令参数	该指令共有 3 个参数，参数的详细信息如下。		
axis	控制轴号。		
alpha	规划器当量的alpha值，取值范围：(-32767, 0)和(0, 32767)。		
beta	规划器当量的beta值，取值范围：(-32767, 0)和(0, 32767)。		
指令返回值	若返回值为 1：若当前轴再规划运动，请调用 GT_Stop()停止运动在调用该指令。 其他返回值：请参照指令返回值列表。		
相关指令	GT_EncScale		
指令示例	无。		

指令 50 GT_PvtContinuousCalculate

指令原型	short GT_PvtContinuousCalculate(long count, double *pPos, double *pVel, double *pPercent, double *pVelMax, double *pAcc, double *pDec, double *pTime)		
指令说明	计算 PVT 运动模式 Continuous 描述方式下各数据点的时间。		
指令类型	立即指令，调用后立即生效。	章节页码	30
指令参数	该指令共有 8 个参数，参数的详细信息如下。		
count	数据点个数。该指令用来计算各数据点时间，不会将数据点下载到运动控制器。		
pPos	数据点位置数组。单位：pulse，数组长度为 count。		
pVel	数据点速度数组。单位：pulse/ms，数组长度为 count。		
pPercent	数据点百分比数组。数组长度为 count。 百分比的取值范围：[0, 100]。		
pVelMax	数据点最大速度数组。单位：pulse/ms，数组长度为 count。		
pAcc	数据点加速度数组。单位是“pulse/ms^2”，数组长度为 count。		
pDec	数据点减速度数组。单位是“pulse/ms^2”，数组长度为 count。		
pTime	返回各数据点的时间。单位：ms。		
指令返回值	请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 5-10 Continuous 描述方式**		

指令 51 GT_PvtPercentCalculate

指令原型	short GT_PvtPercentCalculate (long count, double *pTime, double *pPos, double *pPercent, double velBegin, double *pVel)		
指令说明	计算 PVT 运动模式 Percent 描述方式下各数据点的速度。		
指令类型	立即指令，调用后立即生效。	章节页码	30
指令参数	该指令共有 6 个参数，参数的详细信息如下。		
count	数据点个数。该指令用来计算各数据点的速度，不会将数据点下载到运动控制器。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

pTime	数据点时间数组。单位：ms，数组长度为 count。
pPos	数据点位置数组。单位：pulse，数组长度为 count。
pPercent	数据点百分比数组。数组长度为 count。 百分比的取值范围：[0, 100]。
velBegin	起点速度。单位：pulse/ms。
pVel	返回各数据点的速度。单位：pulse/ms。
指令返回值	若返回值为 1： (1) 请检查当前轴是否为 PVT 模式，若不是，请先调用 GT_PrfPvt 将当前轴设置为 PVT 模式。 (2) 若当前轴在规划运动，请调用 GT_Stop()停止运动再调用该指令。 (3) 请检查传递的数据点是否大于 1024 个。 其他返回值：请参照指令返回值列表。
相关指令	无。
指令示例	**例程 5-9 Percent 描述方式**

指令 52 GT_PvtStart

指令原型	short GT_PvtStart(long mask)		
指令说明	启动 PVT 运动。		
指令类型	立即指令，调用后立即生效。	**章节页码**	30
指令参数	该指令共有 1 个参数，参数的详细信息如下。		
mask	按位指示需要启动的轴号。当 bit 位为 1 时表示启动对应的轴。 在启动运动之前，可以调用 GT_PvtTableSelect 选择数据表。如果没有选择数据表，默认使用数据表 1。如果数据表为空，则启动失败。		
指令返回值	若返回值为 1： (1) 目标数据表不应为空。请检查目标数据表是否空。 (2) 请检查当前轴是否为 PVT 模式，若不是，请先调用 GT_PrfPvt 将当前轴设置为 PVT 模式。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 5-7 PVT 描述方式**		

指令 53 GT_PvtStatus

指令原型	short GT_PvtStatus(short profile, short *pTableId, double *pTime, short count)		
指令说明	读取 PVT 运动状态。		
指令类型	立即指令，调用后立即生效。	**章节页码**	30
指令参数	该指令共有 4 个参数，参数的详细信息如下。		
profile	规划轴号。		
pTableId	当前正在使用的数据表 ID。		
pTime	当前轴已经运动的时间，单位：ms。		
count	读取的轴数。		
指令返回值	若返回值为 1：请检查当前轴是否为 PVT 模式，若不是，请先调用 GT_PrfPvt 将当前轴设置为 PVT 模式。 其他返回值：请参照指令返回值列表。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

相关指令	无。
指令示例	**例程 5-7 PVT 描述方式**

指令 54 GT_PvtTable

指令原型	short GT_PvtTable (short tableId, long count, double *pTime, double *pPos, double *pVel)		
指令说明	向 PVT 运动模式指定数据表传送数据，采用 PVT 描述方式。		
指令类型	立即指令，调用后立即生效。	**章节页码**	30
指令参数	该指令共有 5 个参数，参数的详细信息如下。		
tableId	指定数据表。取值范围：[1, 32]。		
count	数据点个数。每个数据表具有 1024 个存储空间。 每个数据点占用 1 个存储空间		
pTime	数据点时间数组，单位：ms，数组长度为 count。		
pPos	数据点位置数组，单位：pulse，数组长度为 count。		
pVel	数据点速度数组，单位：pulse/ms，数组长度为 count。		
指令返回值	若返回值为 1： (1) 请检查当前轴是否为 PVT 模式，若不是，请先调用 GT_PrfPvt 将当前轴设置为 PVT 模式。 (2) 若当前轴在规划运动，请调用 GT_Stop()停止运动再调用该指令。 (3) 请检查传递的数据点是否大于 1024 个。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 5-7 PVT 描述方式**		

指令 55 GT_PvtTableComplete

指令原型	short GT_PvtTableComplete (short tableId, long count, double *pTime, double *pPos, double *pA, double *pB, double *pC, double velBegin, double velEnd)		
指令说明	向 PVT 运动模式指定数据表传送数据，采用 Complete 描述方式。		
指令类型	立即指令，调用后立即生效。	**章节页码**	30
指令参数	该指令共有 9 个参数，参数的详细信息如下。		
tableId	指定数据表。取值范围：[1, 32]。		
count	数据点个数。每个数据表具有 1024 个存储空间。 每个数据点占用 1 个存储空间。		
pTime	数据点时间数组。单位：ms，数组长度为 count。		
pPos	数据点位置数组。单位：pulse，数组长度为 count。		
pA、pB、pC	工作数组，内部使用，数组长度为 count。 该数组用户不必赋值。		
velBegin	起点速度。单位：pulse/ms。		
velEnd	终点速度。单位：pulse/ms。		
指令返回值	若返回值为 1： (1) 请检查当前轴是否为 PVT 模式，若不是，请先调用 GT_PrfPvt 将当前轴设置为 PVT 模式。 (2) 若当前轴在规划运动，请调用 GT_Stop()停止运动再调用该指令。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

	(3) 请检查传递的数据点是否大于 1024 个。 其他返回值：请参照指令返回值列表。
相关指令	无。
指令示例	**例程 5-8 Complete 描述方式**

指令 56 GT_PvtTableContinuous

指令原型	short GT_PvtTableContinuous (short tableId, long count, double *pPos, double *pVel, double *pPercent, double *pVelMax, double *pAcc, double *pDec, double timeBegin)		
指令说明	向 PVT 运动模式指定数据表传送数据，采用 Continuous 描述方式。		
指令类型	立即指令，调用后立即生效。	章节页码	30
指令参数	该指令共有 9 个参数，参数的详细信息如下。		
tableId	指定数据表。取值范围：[1, 32]。		
count	数据点个数。每个数据表具有 1024 个存储空间。 每个数据点占用 1~8 个存储空间。		
pPos	数据点位置数组。单位：pulse，数组长度为 count。		
pVel	数据点速度数组。单位：pulse/ms，数组长度为 count。		
pPercent	数据点百分比数组。数组长度为 count。 百分比的取值范围：[0, 100]。		
pVelMax	数据点最大速度数组。单位：pulse/ms。数组长度为 count。		
pAcc	数据点加速度数组。单位是“pulse/ms²”。数组长度为 count。		
pDec	数据点减速度数组。单位是“pulse/ms²”。数组长度为 count。		
timeBegin	起点时间。单位：ms。		
指令返回值	若返回值为 1： (1) 请检查当前轴是否为 PVT 模式，若不是，请先调用 GT_PrfPvt 将当前轴设置为 PVT 模式。 (2) 若当前轴在规划运动，请调用 GT_Stop()停止运动再调用该指令。 (3) 请检查传递的数据点是否大于 1024 个。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 5-10 Continuous 描述方式**		

指令 57 GT_PvtTablePercent

指令原型	short GT_PvtTablePercent (short tableId, long count, double *pTime, double *pPos, double *pPercent, double velBegin)		
指令说明	向 PVT 运动模式指定数据表传送数据，采用 Percent 描述方式。		
指令类型	立即指令，调用后立即生效。	章节页码	30
指令参数	该指令共有 6 个参数，参数的详细信息如下。		
tableId	指定数据表。取值范围：[1, 32]。		
count	数据点个数。每个数据表具有 1024 个存储空间。 每个数据点占用 1~3 个存储空间。		
pTime	数据点时间数组。单位：ms，数组长度为 count。		
pPos	数据点位置数组。单位：pulse，数组长度为 count。		
pPercent	数据点百分比数组。数组长度为 count。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

	百分比的取值范围：[0, 100]。
velBegin	起点速度。单位：pulse/ms。
指令返回值	若返回值为 1： (1) 请检查当前轴是否为 PVT 模式，若不是，请先调用 GT_PrfPvt 将当前轴设置为 PVT 模式。 (2) 若当前轴在规划运动，请调用 GT_Stop()停止运动再调用该指令。 (3) 请检查传递的数据点是否大于 1024 个。 其他返回值：请参照指令返回值列表。
相关指令	无。
指令示例	**例程 5-9 Percent 描述方式**

指令 58 GT_PvtTableSelect

指令原型	short GT_PvtTableSelect(short profile, short tableId)		
指令说明	选择 PVT 运动模式数据表。		
指令类型	立即指令，调用后立即生效。	**章节页码**	30
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
profile	规划轴号。		
tableId	指定数据表。 PVT 模式提供 32 个数据表，取值范围：[1, 32]。		
指令返回值	若返回值为 1：目标数据表不应为空。请检查目标数据表是否空。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	**例程 5-8 Complete 描述方式**		

指令 59 GT_RunThread

指令原型	short GT_RunThread(short thread)		
指令说明	启动线程。		
指令类型	立即指令，调用后立即生效。	**章节页码**	51
指令参数	该指令共有 1 个参数，参数的详细信息如下。		
thread	线程编号，取值范围：[0, 31]。		
指令返回值	若返回值为 1： (1) 请检查线程号是否已经绑定。 (2) 请检查相应的数据页中是否超出范围。 其他返回值：请参照指令返回值列表。		
相关指令	GT_StopThread; GT_PauseThread		
指令示例	**例程 6-1 运动程序单线程累加求和**		

指令 60 GT_SetAdcFilter

指令原型	short GT_SetAdcFilter(short adc, short filterTime)		
指令说明	设置模拟量输入的滤波器时间参数。		
指令类型	立即指令，调用后立即生效。	**章节页码**	12
指令参数	该指令共有 2 个参数，参数的详细信息如下。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

adc	模拟量输入通道的编号，取值范围：[1, 8]。
filterTime	模拟量输入信号的滤波器时间参数，取值范围：[0, 50]。无单位。
指令返回值	请参照指令返回值列表。
相关指令	无。
指令示例	无。

指令 61 GT_SetCrdPrm

指令原型	short GT_SetCrdPrm(short crd, TCrdPrm *pCrdPrm)		
指令说明	设置坐标系参数，确立坐标系映射，建立坐标系。		
指令类型	立即指令，调用后立即生效。	**章节页码**	14
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
crd	坐标系号，取值范围：[1, 2]。		
pCrdPrm	设置坐标系的相关参数。 typedef struct CrdPrm { short dimension; short profile[8]; double synVelMax; double synAccMax; short evenTime; short setOriginFlag; long originPos[8]; } TCrdPrm; dimension：坐标系的维数。取值范围：[1, 4]。 Profile[8]：坐标系与规划器的映射关系。Profile[0..7]对应规划轴 1~8，如果规划轴没有对应到该坐标系，则 profile[x]的值为 0；如果对应到了 X 轴，则 profile[x]为 1，Y 轴对应为 2，Z 轴对应为 3，A 轴对应为 4。不允许多个规划轴映射到相同坐标系的相同坐标轴，也不允许把相同规划轴对应到不同的坐标系，否则该指令将会返回错误值。每个元素的取值范围：[0, 4]。 synVelMax：该坐标系的最大合成速度。如果用户在输入插补段的时候所设置的目标速度大于了该速度，则将会被限制为该速度。取值范围：(0, 32767)。单位：pulse/ms。 synAccMax：该坐标系的最大合成加速度。如果用户在输入插补段的时候所设置的加速度大于了该加速度，则将会被限制为该加速度。取值范围：(0, 32767)。单位：pulse/ms^2。 evenTime：每个插补段的最小匀速段时间。取值范围：[0, 32767)。单位：ms。 setOriginFlag：表示是否需要指定坐标系的原点坐标的规划位置，该参数可以方便用户建立区别于机床坐标系的加工坐标系。0：不需要指定原点坐标值，则坐标系的原点在当前规划位置上。1：需要指定原点坐标值，坐标系的原点在 originPos 指定的规划位置上。 originPos[8]：指定的坐标系原点的规划位置值。		
指令返回值	若返回值为 1： (1) 若坐标系下各轴在规划运动，请调用 GT_Stop()停止运动再调用该指令。 (2) 请检查映射到 Profile 有没被激活，若无，则返回错误。 (3) 请见检查相应轴是否在坐标系下。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

	其他返回值：请参照指令返回值列表。
相关指令	GT_GetCrdPrm
指令示例	**例程 5-1 建立坐标**

指令 62 GT_SetCrdStopDec

指令原型	short GT_SetCrdStopDec (short crd, double decSmoothStop, double decAbruptStop)		
指令说明	设置插补运动平滑停止、急停合成加速度		
指令类型	立即指令，调用后立即生效。	章节页码	14
指令参数	该指令共有 3 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
decSmoothStop	设置的坐标系合成平滑停止加速度。取值范围：(0, 32767)，单位：pulse/ms^2。		
decAbruptStop	设置的坐标系合成急停加速度。取值范围：(0, 32767)，单位：pulse/ms^2。		
指令返回值	若返回值为 1：检查当前坐标系是否映射了相关轴。 其他返回值：请参照指令返回值列表。		
相关指令	GT_GetCrdStopDec		
指令示例	无。		

指令 63 GT_SetOverride

指令原型	short GT_SetOverride (short crd, double synVelRatio)		
指令说明	设置插补运动目标合成速度倍率。		
指令类型	立即指令，调用后立即生效。	章节页码	14
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
synVelRatio	设置的插补目标速度倍率，取值范围：(0, 1]，系统默认该值为：1。		
指令返回值	若返回值为 1：检查当前坐标系是否映射了相关轴。 其他返回值：请参照指令返回值列表。		
相关指令	无。		
指令示例	无。		

指令 64 GT_SetPvtLoop

指令原型	short GT_SetPvtLoop(short profile, long loop)		
指令说明	设置 PVT 运动模式循环次数。		
指令类型	立即指令，调用后立即生效。	章节页码	30
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
profile	规划轴号。		
loop	指定循环执行的次数。 0 表示无限循环。		
指令返回值	若返回值为 1：请检查当前轴是否为 PVT 模式，若不是，请先调用 GT_PrfPvt 将当前轴设置为 PVT 模式。 其他返回值：请参照指令返回值列表。		
相关指令	GT_GetPvtLoop		
指令示例	**例程 5-8 Complete 描述方式**		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

指令 65 GT_SetStopDec

指令原型	short GT_SetStopDec(short profile, double decSmoothStop, double decAbruptStop)		
指令说明	设置平滑停止减速度和急停减速度。		
指令类型	立即指令，调用后立即生效。	章节页码	12
指令参数	该指令共有 3 个参数，参数的详细信息如下。		
profile	规划器的编号		
decSmoothStop	平滑停止减速度，取值范围：(0, 32767]。单位：pulse/ms^2。		
decAbruptStop	急停减速度，取值范围：(0, 32767]。单位：pulse/ms^2。		
指令返回值	请参照指令返回值列表。		
相关指令	GT_GetStopDec		
指令示例	无。		

指令 66 GT_SetStopIo

指令原型	short GT_SetStopIo(short axis, short stopType, short inputType, short inputIndex)		
指令说明	设置平滑停止和紧急停止数字量输入的信息。		
指令类型	立即指令，调用后立即生效。	章节页码	12
指令参数	该指令共有 2 个参数，参数的详细信息如下。		
axis	需要设置停止IO信息的轴的编号。		
stopType	需要设置停止 IO 信息的停止类型。 0：紧急停止类型。 1：平滑停止类型。		
inputType	设置的数字量输入的类型。 MC_LIMIT_POSITIVE(该宏定义为 0)，正限位。 MC_LIMIT_NEGATIVE(该宏定义为 1)，负限位。 MC_ALARM(该宏定义为 2)，驱动报警。 MC_HOME(该宏定义为 3)，原点开关。 MC_GPI(该宏定义为 4)，通用输入。 MC_ARRIVE(该宏定义为 5)，电机到位信号。		
inputIndex	设置的数字量输入的索引号，取值范围根据 inputType 的取值而定。		
指令返回值	请参照指令返回值列表。		
相关指令	无。		
指令示例	无。		

指令 67 GT_SetUserSegNum

指令原型	short GT_SetUserSegNum(short crd, long segNum, short fifo=0)		
指令说明	设置自定义插补段段号。		
指令类型	缓存区指令。	章节页码	14
指令参数	该指令共有 3 个参数，参数的详细信息如下。		
crd	坐标系号。正整数，取值范围：[1, 2]。		
segNum	设置用户自定义的插补段段号。		
fifo	插补缓存区号。正整数，取值范围：[0, 1]，默认值为：0。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

指令返回值	若返回值为 1： (1) 检查当前坐标系是否映射了相关轴。 (2) 检查是否向 fifo1 中传递数据，若是，则检查 fifo0 是否使用并运动，若运动，则返回错误。 (3) 检查相应的 fifo 是否已满。 其他返回值：请参照指令返回值列表。
相关指令	GT_GetUserSegNum
指令示例	无。

指令 68 GT_SetVarValue

指令原型	short GT_SetVarValue (short page, TVarInfo *pVarInfo, double *pValue, short count=1)		
指令说明	设置运动程序中变量的值。		
指令类型	立即指令，调用后立即生效。	**章节页码**	51
指令参数	该指令共有 4 个参数，参数的详细信息如下。		
page	数据页编号。 全局变量为-1。 局部变量取值范围：[0, 31]。		
pVarInfo	需要访问的变量标识。		
pValue	需要写入的变量值。		
count	需要写入的变量值的数量，取值范围：[1, 8]。		
指令返回值	请参照指令返回值列表。		
相关指令	GT_GetVarValue		
指令示例	**例程 6-1 运动程序单线程累加求和**		

指令 69 GT_StopThread

指令原型	short GT_StopThread(short thread)		
指令说明	停止正在运行的线程。		
指令类型	立即指令，调用后立即生效。	**章节页码**	51
指令类型	立即指令，调用后立即生效。		
指令参数	该指令共有 1 个参数，参数的详细信息如下。		
thread	线程编号，取值范围：[0, 31]。		
指令返回值	若返回值为 1： (1) 请检查线程号是否已经绑定。 (2) 请检查相应的数据页中是否超出范围。 其他返回值：请参照指令返回值列表。		
相关指令	GT_RunThread; GT_PauseThread		
指令示例	无。		

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

第8章 索引

8.1 指令详细说明

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

8.2 例程索引

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

8.3 表格索引

8.4 图片索引

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层

上海固高欧辰智能科技有限公司 www.softlink.cn 86-21-54708786 86-21-54708386 上海市闵行区东川路555号4号楼一层