上海微敏自控技术有限公司

使用手册

DMC-2143

2/06

目　录

第一章　概述

简介

DMC21x3 系列是 GALIL 经济型多轴独立控制器。该控制器提供了强大功能，和早期型号相比，具有更高的通信速度、非易失性存储器、更高的编码器反馈速度以及功能更丰富的选件。

每个 DMC-21x3 提供两个通信通道：RS-232（最高波特率 19.2K）和 10M 以太网，且提供 4M Flash EEPROM 存储器，用于存储应用程序、参数、变量、数组。可以很容易地在现场对固件进行升级。

每个 DMC-21x3 最多可以用来控制 8 个轴。DMC-2113、DMC-2123、DMC-2133、DMC-2143、DMC-2153、DMC-2163、DMC-2173 和 DMC-2183 分别是一到八轴控制器的具体型号。

DMC-21x3 在专为解决复杂运动问题而设计，能够在包括点动、点到点定位、矢量定位、电子齿轮同步、电子凸轮、组合运动、轨迹运动的应用中使用。控制器通过对运动规划曲线的平滑处理以及加、减速过程的控制，可大大减小对机械部分的运动冲击。为了对复杂轮廓平滑跟踪，DMC-21x3 还提供无限直线、圆弧线段的矢量进给。控制器的电子齿轮也提供了多主轴以及龙门模式的操作。

DMC-21x3 提供了通用 I/O 来与外部事件同步。8 数字输入（DMC-2153 到 DMC-2183 为 16 路数字输入）、8 路数字输出（DMC-2153 到 DMC-2183 为 16 路数字输出）如果使用了 DB-28040 扩展选件，就可以再增加 40 路数字 I/O 和 8 路模拟量输入（详见《DMC-21x3 可选附（硬）件手册》）。如果不需要辅助编码器，还有更多的输入点可以使用（每轴两路）。每轴正、负向限位、急停、原点开关和可定义输入中断有专用的 TTL 输入接口。

可以用二进制或 ASCII 码格式向 DMC-21x3 传送命令。并提供用于自整定、在 PC 上观察运动轨迹、AUTOCAD 文档转换、多种环境下（如 Visual Basic、C、C++等等）开发的工具软件。并向用户提供 DOS、Linux、Windows3.x、95、98、2000、ME、NT、XP 下的驱动函数。

电机接口类型概述

DMC-21x3 提供如下电机控制信号

1、 ±10V 模拟电压信号——用于伺服电机的速度模式、扭矩模式、变频电机等等
2、 正弦波信号——用于某些无刷电机
3、 脉冲/方向信号——用于步进电机、伺服电机位置模式
4、 其它执行机构——如比例阀——的控制信号（详情请向代理商上海微敏自控技术有限公司咨询）

±10V 模拟电压信号

DMC-21x3 电机指令使用 16 位 DA 输出，PID 滤波器具有速度/加速度前馈、积分限制等特性的，因此可以达到更高的精度。

控制器在出厂时将设置成标准伺服电机工作方式。在此设置下，控制器输出±10V 模拟指令信号，与伺服放大器相连接。具体连接方式将在第二章中详述。

正弦波信号

DMC-21x3 能够提供控制某些无刷电机需要的正弦波信号。在这种设置下，控制器产生两路正弦波信号连接到特定设计的放大器上。

提示：无刷电机所需要的正弦波，有可能是在放大器上产生。如果是由放大器生成正弦波，只需要连接一路指令信号，控制器按照标准伺服的方式设置即可。

脉冲/方向信号

DMC-21x3 能够用于控制步进电机或者工作在位置模式下的伺服电机。在这种方式中，控制器输出脉冲、方向两路信号与步进电机驱动器相连接。对于脉冲输出工作方式来说，控制器不需要编码器反馈，而以开环方式工作。具体连接方式将在第二章中详述。

放大器概述

放大器应该与电机相匹配，可以是线性调制或脉宽调制（PWM）。放大器可以有电流反馈、电压反馈或速度反馈。

电流型放大器（伺服电机的扭矩模式）

电流型放大器接收±10V 模拟指令信号。按照+10V 指令产生所需最大电流（扭矩）原则来设置放大器增益。例如，如果电机峰值电流为 10A，放大器增益就应该是 1A/V。

速度型放大器（伺服电机的速度模式）

对于速度型放大器，10V 指令信号应该使电机以所需最大速度运转。

步进放大器　（伺服电机的位置模式）

位置型放大器接收指令脉冲+方向信号。

DMC-21x3 功能组件：

DMC-21x3 的结构可以理解为如图 1．1 所示的功能组件。各功能组件的内容将在下面讨论。

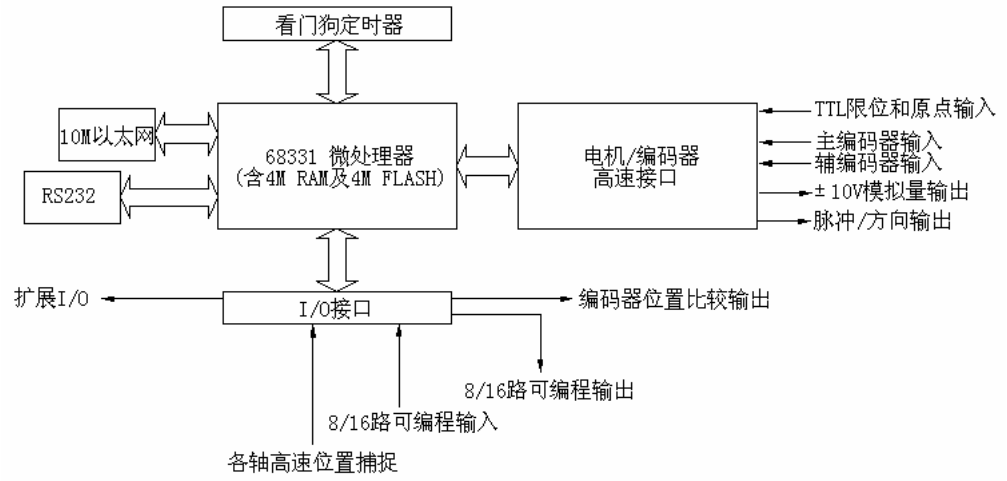


图 1.1 DMC-21x3 功能组件

微处理器

DMC-21x3 的中央处理器是专用的摩托罗拉 68331 系列微机，带有 4 兆 RAM 和 4M FLSH。RAM 为变量、数组和程序提供存储空间，FLASH 为变量、数组和程序提供断电保存，同时，控制器的固件也存储在 FLASH 中。

电机接口

Galil 的 GL1800 对于正交方波信号的处理最高可达 12MHz。对于标准伺服电机操作，它产生一个±10V 的模拟量（16 位 DA 转换）；对于正弦波的操作，它利用两个 DA 通道产生 2 个±10V 模拟量。对于步进电机操作，它产生脉冲和方向信号。

通讯

DMC-21x3 的通讯界面包括一个 10M 以太网和一个 RS232 串口。串口的波特率最高为 19.2K。

通用 I/O

DMC-21x3 提供了 8 路 TTL 输入、8 路 TTL 输出的接口。如果订购了 DB28040，就可以附加 40 个 I/O。此外，没有用的辅编码器也可以作为额外的输入使用（每轴两路）。通用输入点也可以用做各轴的高速位置捕捉。另外提供位置比较输出信号。

DMC-2152 到 DMC-2182 提供 16 路 TTL 输入、16 路 TTL 输出。

看门狗定时器

DMC-21x3 提供内部看门狗定时器，以确保微处理器工作在正常状态。定时器会触发放大器使能输出（AMPEN）。在 DMC-21x3 出现严重故障时，用 AMPEN 使放大器输出关断。AMPEN 通常为高电平，在上电期间，如果微处理器暂停工作，AMPEN 输出就会变低，报警灯也在此时点亮。此时可用复位操作使 DMC-21x3 恢复工作。

系统组件

如图 1.2 所示，DMC-21x3 是运动控制系统的一部分，这个系统中还包括放大器、编码器、电机。下面介绍这些组件。

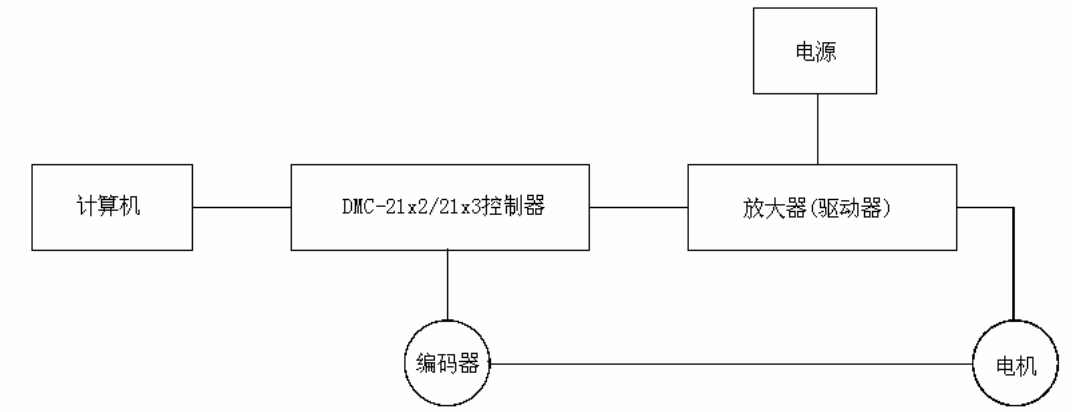


图 1.2 运动控制系统的组件

电机

电机将电流转换为产生运动的力。每轴的运动都需要一个能够按照需要的速度和加速度来拖动负载的电机。GALIL 的软件“Motion Component Selector”可以帮助用户确定电机的规格，如果需要此软件，请与 GALIL 代理商上海微敏自空技术有限公司联系。

电机可以是步进或伺服，可以是有刷或无刷，可以是旋转或直线的。对于步进电机，可以是整步、半步或者微步进（细分）控制。当使用步进电机时，不需要编码器。

放大器（驱动器）

对于每个轴而言，放大器将来自控制器的±10V 电压信号转换为驱动电机的电流。对于步进电机，就是将脉冲和方向信号转化为电流。放大器要能够为电机提供足够的电流。

编码器

编码器将运动转化为电脉冲并反馈给控制器。DMC-21x3 可以接收旋转编码器或直线编码器的信号。典型的编码器输出两路正交的信号，称为 CHA 和 CHB。编码器信号可以是单端的（只有 CHA 和 CHB），也可以是差动的（包括 CHA+和 CHA-，CHB+和 CHB-），DMC-21x3 对这两种信号都可以正常接收，并且进行 4 倍频处理。编码器还可以有用于定位的第三路信号（index）。

针对步进电机，DMC-21x3 也可以接收脉冲/方向类型的编码器信号。

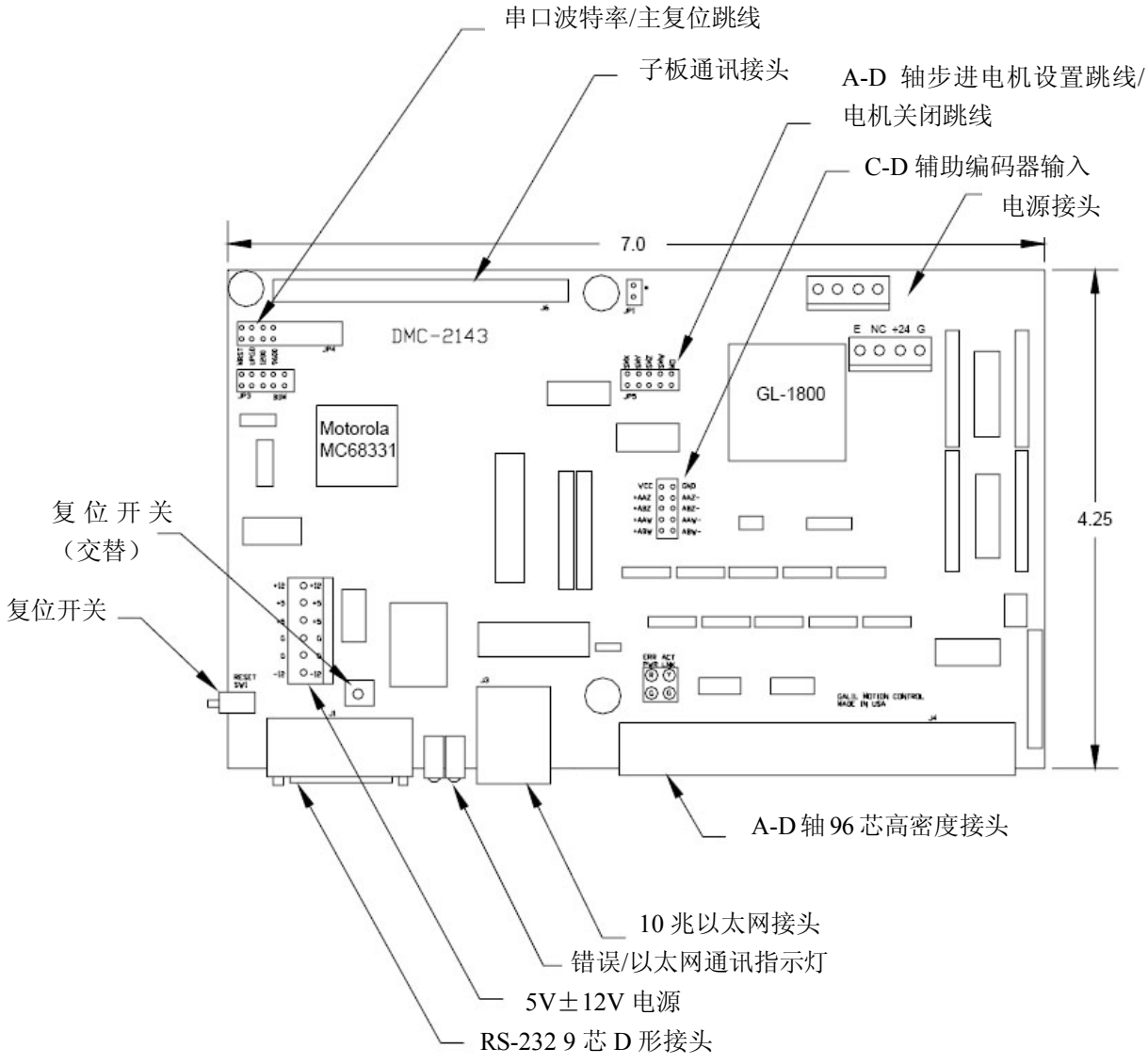
DMC-21x3 对编码器的分辨率没有任何限制，但是，输入到控制器的编码器信号频率不能超过 3,000,000 个周期/秒（12,000,000 计数/秒）。例如，编码器是 10000 线/英寸，那么最大速度就是 300 英寸/秒。如果需要更高的编码器信号频率，请与 Galil 代理商联系。

标准的电压规格是 TTL（0 到 5V），不过，电压达到 12V 也可以接受（如果是编码器是差动输出，12V 信号可以直接输入到 DMC-21x3，如果是单端 12V 输出信号，需要在负输入端接入一个偏压。

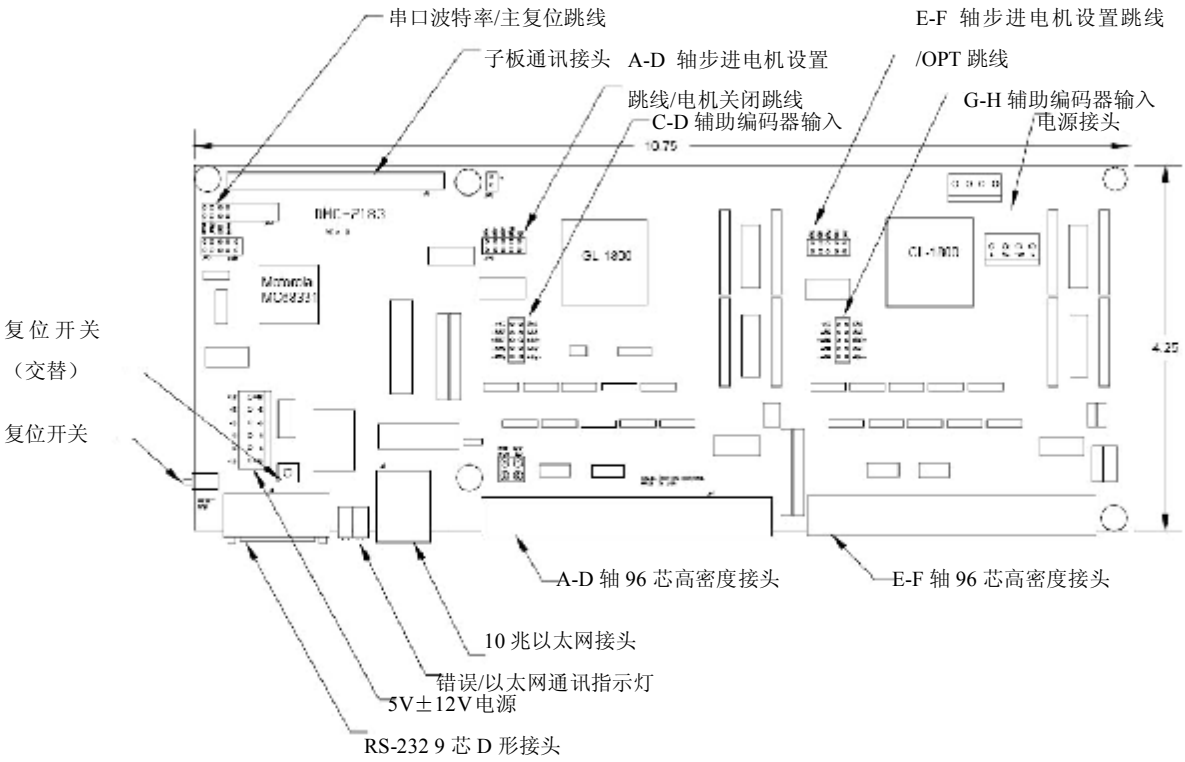
需要连接其它类型的位置传感器，比如绝对式编码器，Galil 可以定制控制器和相关指令。请与 Galil 代理商上海微敏自控技术有限公司联系并将您特定系统的需求告诉我们的工程师。

第二章　起步

DMC-2113—DMC2143



DMC-2153—DMC2183



DMC-21x3 系统所须部件

部件	数量	备注
DMC-21x3 控制卡	1	必选
ICM20105T 端子板	1 或 2	推荐，也可选择其它接线端子板或自行制作端子板
RS232 连接线	1	可选
以太网连接线	1	可选
DC 电源	1	
DB-28040 I/O 扩展卡	1	如需要扩展 I/O 接点，则必选
CB-50-80	1	如需要扩展 I/O 接点，则必选
CAB-80	1	如需要扩展 I/O 接点，则必选
IOM-1964-80	1	如需要扩展 I/O 接点，则必选
模拟量输入线	1	如需要模拟量输入，则必选

安装 DMC-21x3

要构成一套完整的 DMC-21x3 运动控制系统，需要九个步骤。

第一步： 确定电机配置

第二步： 设置 DMC-21x3 上的跳线

第三步： 设置通讯频率

第四步： 安装通讯软件

第五步： 为控制器连接电源
第六步： 与 GALIL 软件建立通讯
第七步： 连接驱动器（放大器）和编码器
第八步： a 连接模拟控制方式驱动器
b 连接脉冲控制方式驱动器
第九步： 调整伺服增益

第一步：确定电机配置

在建立控制系统之前，用户必须确定所需要的电机配置。DMC-21x3 能够控制标准伺服电机，正弦波换向无刷电机，步进电机或其中的任意组合。其它种类的执行机构（如直线电机、直接驱动电机、液压伺服马达等）也可连接使用。下面是有关电机配置方面的说明。

模拟量控制方式电机

DMC-21x3 在出厂时已经被设置成模拟量控制电机的工作方式，提供±10V 模拟指令输出信号。因此，无需再做硬、软件配置。

脉冲控制方式电机

要想将 DMC-21x3 配置成脉冲控制电机的工作方式，在硬件上对每个轴需要一个用于设定脉冲模式的跳线，而且在软件配置方面必须用 MT 指令将所对应的轴设置成脉冲方式。第二步和第九步中将对此进行详细介绍。

第二步：设置 DMC-21x3 上的跳线

主复位和升级跳线：

在 DMC-21x3 控制板上，JP2 含有 MRST 及 UPGD 跳线。MRST 用于主复位（恢复到出厂时状态），当 MRST 短接时，对控制器上电或使复位输入信号变低，控制器就执行主复位操作。无论何时对控制器进行主复位，都将擦除用户存储在 EEPROM 中的所有程序、数组、变量和运动控制参数。

UPGD 短接时可以控制器的控制软件无条件升级。在控制器正常工作时，对软件升级不需要短接此跳线，在 EEPROM 数据出错的情况下，才需要短接此跳线。通常情况下 EEPROM 不会毁坏，如果在控制软件升级期间出现电源故障（断电），有可能使 EEPROM 内数据出错。若 EEPROM 内出现数据错误，您的控制器就不会正常工作。在此情况下，短接 UPGD 跳线，使用 GALIL 智能终端（Smart Terminal）中的更新软件功能（UpdateFirmware）重新装入系统控制软件。

脉冲模式跳线：

要想使某一轴以脉冲方式工作，必须短接与之相对应的脉冲方式（SM）跳线。此跳线位于DMC-21x3控制板上，标号为JP5和JP7。各个短路棒分别以SMX~SMW来标注。当短接跳线时，对应的轴才可以被设置成“脉冲方式”。

电机关断跳线：

可以用控制器上的硬件跳线选择上电时的电机状态。短接MO位置上的跳线，就会使控制器在上电时保持“电机关断”状态。为了使电机使能，就需要执行SH指令。不短接该跳线，电源一上电，控制器就会立即使电机使能。要关断电机，就需要执行MO指令。

MO跳线位于脉冲模式跳线（SM）的旁边。

第三步：设置通讯频率

RS232口的通讯频率通过控制板上的跳线设置，在JP2上有1200和9600两个跳线，通讯频率的设定如下表

1200	9600	波特率
短接	开放	1200
开放	短接	9600
开放	开放	19200

第四步：安装通信软件

在对计算机上电后，您应该安装 GALIL 软件，使控制器和 PC 机之间能够通信。

DOS：

用 GALIL 软件 CD-ROM 盘，进入 16BitInstalls\DMCDOS\Disk1 目录，在提示符处键入“INSTALL”，并按提示操作。

Windows3.x（16bit 版）：

用 GALIL 软件 CD-ROM 盘，进入目录 16BitInstalls\DMCWIN，运行 DMCWINl6.exe，并按提示操作。

Windows95，98，NT，ME，2000，XP（32bit 版）：

插入 GALIL 软件 CD-ROM 盘时，GALIL 软件 CD-ROM 就会自动开始安装过程。将 GALIL CD-ROM 安装到您的计算机之后，您就能很容易地安装所需要的其它软件工具。要想安装基本通信软件，就运行 GALIL 软件 CD-ROM，并选择“DMCSmartTerm”。此时，就会将 GALIL 集成终端工具软件安装到您的计算机上。用此在计算机与控制器之间建立通信。

第五步：为控制器连接电源

不要在通电后连接 DMC-21x3 的各种扩展硬件。DMC-21x3 的规格中如果没有-DC 的选项，控制板需要 5V 和+/-12V 的直流电源。确认控制板上电源插针上的标注，如果电源接线错误会导致控制板损坏。如果在 DMC-21x3 的规格中包含了-DC 的选项，则需要连接相应的单一电源（-DC24 可接受 18V-36V，-DC48 可接受 36V-72V）。

第六步：与 GALil 软件建立通信

串口通信：

用 RS232 电缆（直通 9 芯线）将 DMC-21x3 串口与您的计算机串口相连接。

使用 DOS 版 GALIL 软件

要与 DMC-21x3 通信，在提示行键入 TALK2DMC，一旦您建立了通信，终端显示应出现一个冒号(：)。如果未接收到冒号，就按回车键。如果没有返回冒号提示，很大可能性是由于串口设置错误。用户必须确保在与控制器进行通信时，设置正确的通信口和波特率。请注意：为了与 GALIL 软件进行通信，控制器上的串口必须设置为硬件握手方式。用户也必须确保使用合适的串行通信电缆。串行通信电缆的引脚定义参见附录。

使用 Windows 版 GALIL 软件

为了在 Windows 环境下与 GALIL 控制器进行通信，必须在 Windows Registry（注册表）中注册控制器。要注册控制器，您必须指定控制器的型号、通信参数及其它相关资料。通过 GALIL 软件进行注册，如 SmartTerminal 和 WSDK。用“New Controller”按钮来添加一个新控制器到 Registry（注册表）。此时需要您提供 GALIL 控制器类别型号（DMC-2102/3）。点击靠右的下拉箭头就会展现出可用的控制器型号的菜单。然后，您就可以选择串行或 Ethernet 连接。注册信息会显示出缺省通信口 1，缺省速度为 19200。如果需要，就应该更改此信息，以便与计算机通信口和控制器上所设定的波特率相一致。注册过程中也显示定时器超时和延迟信息。这些属高级参数，应该由高级用户来更改（详细说明请参见软件文档）。

一旦您为控制器设置了合适的注册信息，就选择 OK 并关闭注册窗口。至此，您就能够与 DMC-21x3 进行通信了。

如果不能与控制器进行通讯，程序会暂停 3~15 秒，然后显示错误信息。出现这样的情况，原因大多是串行通信口设定不正确或通讯线有问题。当想与控制器通信时，您必须设置正确的通信口和波特率。请注意：用 GALIL 软件必须将控制器上的串口设置成“硬件握手方式”，同时确保使用“直通”的串行通信电缆。串行通信电缆的引脚定义参见附录。

一旦您建立通信，点击“Smart Term”菜单，您就会接收到一个冒号（：）提示。与控制器通信在后续章节中描述。

使用非 GALIL 通信软件

DMC-21x3 串口配置为 DATASET 方式，您的计算机或终端也必须配置为 DATATERM、全双工，无校验，8 个数据位，1 个起始位和 1 个停止位。确认波特率开关设置为上述所需之波特率。

需要将您的计算机配置为“dumb”终端，发送 ASCII 码给 DMC-21x3。如果需要回显控制器收到的指令，使用“EO”指令

Ethernet 通信

使用 Windows 版 GALIL 软件

必须在 Windows 注册表中注册控制器，以便主计算机与其通信。通过 GALIL 软件（如 SmartTerminal 或 WSDK）可以进行注册。

使用 New Controller 按钮在注册表中添加新控制器，或者用 Find Ethernet Controller 按钮自动查找连接到网络上的控制器。添加新控制器时，选择 DMC-2102 作为控制器类型，输入从您的系统管理器中得到的 IP 地址，选择您想与控制器通信的 UDP 或 TCP 协议相对应的按钮。如果没有分配 IP 地址给控制器，就点击 ASSIGN IP ADDRESS。

ASSIGN IP ADDRESS 将检查连接到网络中的控制器，查看哪一个控制器还没有 IP 地址，然后程序就会提问您是否想将您输入的 IP 地址分配给带有指定序号的控制器。点击 YES，就分配此 IP 地址，点击 NO，移动到下一个控制器，或点击 CANCEL，不保存变更。如果网络上不存在还没有分配 IP 地址的控制器，程序就会给出说明。

做完注册后，点击 OK。如果您不想保存此次修改，就点击 CANCEL。一旦控制器已注册，就从列表中选择正确的控制器，并点击 OK。如果软件成功地与控制器建立了通信，就会在显示屏顶部显示注册表信息。

注意：必须通过 Ethernet 连接来注册控制器。

向终端发送测试指令：

在您连接终端之后，按键盘上的<Return>或<Enter>。作为应答，控制器以冒号(：)回应。

现在键入

TPA<Return>

此指令让控制器返回 A 轴的当前位置。控制器应以数字回应，例如：

00000000

第七步：连接驱动器（放大器）和编码器

当您建立了 GALIL 软件与 DMC-21x3 之间的通信，就可准备连接运动控制系统中的其余部件。一般来说，运动控制系统还应该包括接口模块（如 ICM-20105）、各轴放大器以及电机组成。

下面是连接一套运动控制系统的首要步骤：

A、按放大器说明书正确连接电机与放大器，但不要与控制器相连。打开放大器电源，电机仍应静止不动，如电机出现转动，或放大器上出现错误信息，请与电机及放大器的

供应商联系。关闭放大器电源，以进行后面的接线工作

B、连接驱动器使能信号

通常在接口模块上对于驱动器使能信号进行光电隔离处理，因此，可以很方便地与放大器侧的使能信号相连接。针对不同类型驱动器的具体接线方式，请参看相关模块的接线说明。

控制器上的驱动器使能信号受看门狗定时器、电机关断指令 MO 及超差报警使能关断指令 OE 控制。

C、连接编码器

对于脉冲模式，编码器作为可选项。可以连接编码器以读取实时位置值，但不进行自动闭环补偿。

DMC-21x3 可接收带定标脉冲或不带定标脉冲的单端或差分编码器反馈信号。对于编码器刻线密度没有限制，不过反馈到控制器的输入脉冲频率不要超过 3,000,000Hz（即 12,000,000Hz 计数单位），DMC 控制器在内部对接收到的正交脉冲信号进行四倍频处理。例如，如果编码器刻线密度为 10000 周期/inch，最大速度即是 300inch/sec。

可接收的编码器输出信号标准电平是 TTL（RS422，即 0~5V），不过，也可接受电平为 12V 的脉冲信号。（如果使用差分信号，可以将 12V 的脉冲信号直接输入到 DMC-21x3，单端 12V 脉冲信号需要在反向输入端加一定的偏置电压）。

务必确认编码器与 DMC 控制器互联模块之间的连线正确。否则会导致不能正确反馈位置。

DMC-21x3 除接受正交脉冲的编码器信号外，还能够接收脉冲+方向的编码器信号。当使用脉冲+方向编码器时，将脉冲信号连接到 CHA，方向信号连接到 CHB。此时，用户必须用 CE 指令将控制器编码器反馈形式配置成脉冲+方向方式，具体详见《指令手册》。另外，如果需要连接其它信号形式的编码器，请与上海微敏自控技术有限公司或各地办事处联系。

D、验证编码器工作正常与否

先从 A 轴编码器开始，在确认连接无误之后，转动电机轴，并用指令 TPA<回车>查询位置，控制器读回的值就会随电机转动而变化。如果 TPA 读取值不随编码器转动而变化，其原因有如下三种：

a．编码器连线错误——检查相对应轴的连线。

b．编码器损坏——用示波器观察编码器信号。验证 A、B 相幅值在 5~12V 之间。若编码器损坏，请更换编码器。如果您未能观察到编码器信号，请用不同的编码器试试。

c．控制器硬件损坏——连接同一个编码器到其它轴，如果问题消失，即可判断为控制器硬件故障。

第八步 A：连接模拟控制方式驱动器

可以将电机和放大器配置成转矩或速度方式。在转矩方式下，放大器增益应该是 10V 指令电压信号对应所需要的最大电流。而在速度方式下，10V 指令电压信号应该对应电机的最大速度。

在进行以下步骤时，如有可能，将负载与电机脱离，以避免发生危险的可能！

a.检查反馈环的方向

假如电机和放大器连接正确，且编码器工作正常，则继续下一步。将电机、放大器与控制器连接之前，按下一步设定误差限制和转矩限制参数。注意：以下的指令全部以 A 轴为例。

b．设定误差限制值以确保安全

通常，关于反馈的正确极性具有不确定性。反馈极性错误会使电机暴走（飞车）。

用终端软件，设定下列参数以防系统毁坏：

输入指令：

ER2000<CR>　　设定 A 轴误差限制值为 2000 计数单位

OEl<CR>　　当超差时，断开 A 轴放大器使能　　•

此时，如果电机飞车，位置环误差超过 2000 计数单位，就会立即关断使能信号。

注意：此功能要求控制器 AMPEN 信号与放大器使能信号相连接时才有效。

c．设定转矩限制值以确保安全

为了限制输出到放大器的最大电压信号，DMC-21x3 控制器有转矩限制指令 TL。此指令设定控制器的最大电压输出，且在初始设置伺服系统时能够用来避免转矩或速度过大。

当放大器以转矩方式工作时，控制器输出电压将直接与电机的转矩输出相关；用户根据所使用的电机、放大器数据决定这种关系。设定转矩限制值就会限制电机输出转矩。

当放大器以速度方式工作时，控制器的指令输出电压将直接与电机的速度相关。用户根据电机、放大器数据决定这种关系。此时，设定转矩限制值就会限制电机的速度。

例如：以下指令就会将 A 轴指令输出电压限制为 1V：

TLl<CR>

注意：一旦确定了反馈回路的正确极性，一般来说，就应该将转矩限制值增大到缺省值 9.99。如果转矩限制值在正常工作范围以下，伺服系统不能发挥处全部的性能。详见《指令手册》。

d．连接电机

设定好参数后，即可连接模拟指令信号(ACMD)到放大器。

要想检测反馈极性，用以下指令进行一个运动：

PRl000<CR>　相对位置 1000 计数单位

BGA<CR>　　开始 A 轴运动

当反馈极性不正确时，电机就会暴走，但当位置误差超过 2000 计数单位时，控制器就会将使能关断，使电机停止。如果电机暴走，就必须转换系统中的极性。

转换系统极性：

当反馈极性不正确时（正反馈时），用户必须转换系统极性，转换系统极性的方法有多种方法。如果您连接的是 DC 伺服电机，最简单的方法是将两根电枢线进行交换（一般为红色线和黑色线）。如果您连接的是 AC 伺服电机，交换编码器反馈连线即可进行极性转换。如果您用的是单端型编码器，就将 CHA 和 CHB 相交换；如果您用的是差分型编码器，就只交换 CHA+和 CHA-。通过软件指令 MT 和 CE 也能够改变系统极性和编码器极性，详见《指令手册》。

有时，虽然反馈极性正确（电机不暴走），但运动的方向与期望的方向相反，在这种情况下，需要将电机指令和编码器极性同时转换。

如果电机以所需方向运动，但未能到达目标位置而停止，最可能的原因是由于转矩输出不够，此时，需要检查电机指令输出信号 ACMD，减小电机侧的系统磨擦可能会好一些。用如下指令确认：

TTA<Return>　报告 A 轴转矩

报告实际转矩输出信号的大小，以确认磨擦力的大小。

一旦您建立了闭环系统的正确极性，您就能够调整 PID 滤波器参数 KP，KD，KI。为了获得良好的系统响应性和高精度，就需要准确调整伺服系统参数。将在下一步描述。

有关控制器与放大器与伺服电机的连接请参考相关接线说明。

第八步 B：连接步进电机（脉冲控制）

在脉冲控制的中，指令脉冲输出信号占空比为 50%。控制器以开环方式工作，不需要编码器反馈。在此模式下，相对应轴的辅助编码器不能用于外部连接。如果使用编码器作为位置反馈，将编码器连接到对应轴的主编码器输入口。所指定的脉冲位置用指令 RP 或 TD 来查询。编码器位置反馈可以用指令 TP 查询。

能够用滤波器参数 KS 对输出脉冲频率进行平滑处理。KS 参数范围为 0.5-8，其中 8 意味着平滑量最大。详见《指令手册》中 KS 指令的说明。

所有运动指令均适用于脉冲控制工作方式，如：PR，PA，VP，CR，JG 等。在参数方面，相对简单一些，使用加减、减速、平滑参数即可。由于脉冲方式以开环运行，PID 滤波器不起作用。

要将 DMC-21x3 与在脉冲模式下与电机相连接，您必须遵循以下步骤进行：

a．短接 SM 跳线

在 DMC-21x3 控制板有与各轴以步进电机工作方式相对应的跳线。

b．连接脉冲和方向信号到放大器

c．用 MT 指令将 DMC 控制器配置为步进电机方式，输出脉冲型式分为高有效、低有效脉冲两种。使用 MT2 为低有效脉冲方式，MT-2 为高有效脉冲。详见《指令手册》中关于 MT 指令的说明。

第九步：调整伺服增益

对于使用模拟输出控制电机的方式，通过上面的步骤确认接线正确且输出控制信号与输入编码器信号方向一致后，需要调整相关参数，使系统性能得到优化，从而获得快速、精确的响应性能。对于不同类型的放大器，调整方式也略有不同，下面以 A 轴连接速度型放大器为例，介绍大致的调整过程。

对于速度型放大器，重要的参数有零点偏置 OF、速度前馈系数 FV、比例增益 KP、积分增益 KI、微分增益（阻尼系数）KD。

A、确定零点偏置

顺序输入如下指令：

MOA	关闭 A 轴使能信号
KDA=0	取消微分增益
KIA=0	取消积分增益
KPA=0.1	设置较小的比例增益
SHA	打开 A 轴使能信号

在电机稳定后，继续输入指令

TTA	查看当前输出控制信号的大小

记录下屏幕显示的数据，比如 0.0500，用 OF 指令将此数据输入

OFA=0.05	输入零点偏置

B、根据放大器的参数确定速度前馈

从放大器的依据放大器的手册，查出 1V 指令电压所对应的转速以及编码器信号每转的脉冲数。计算出驱动器接受 1V 指令信号时，编码器信号的计数频率：如某速度型放大器，1V 电压对应的转速为 500rpm，编码器每转为 2500 个脉冲，则对应频率为（由于 GALIL 会对编码器信号进行 4 倍频处理，所以计数频率是实际脉冲频率的 4 倍）：

f=500/60*2500*4≈83333Hz

然后计算相应的速度前馈 fv=820000/f，对于上面的例子：

fv=820000/83333≈10

输入计算结果

FVA=0.05　　输入速度前馈设置

MO 指令关闭电机。用

C、调整增益

将比例增益和微分常数设定一个较小值，例如：

KPl<Return>　比例增益

KDl00<Return>微分常数

为了增加阻尼就增大 KD(最大值为 4095)。渐渐增大，当电机出现振动(出现异常声音)时，停止增大。此时，若用指令 TEA<Return>反复发送几次，得到不同的返回值，在出现振荡时，此值的极性交替变化。此时，适当减小 KD，直到系统稳定。

渐渐增大 KP 值（最大值为 1023），用 TE 指令监测响应性能的改善。例如，

KPl0<Return>　比例增益

TEA<Return>　监测返回值

随着 KP 增大，误差会减小。

如果增益过大，也会出现系统振荡，此时，适当减小 KP。一般来说，KP 不应该大于 KD/4。

选择 KI，从 0 值开始渐渐增大。积分器会消除位置误差，使系统控制精度得以提高。用 TEA<Return>进行验证，最终结果是随着 KI 增大，误差变为 0。如果 KI 过大，会导致系统振荡。在系统出现振荡时，适当减小 KI。

设计编程举例

在对系统进行连接、基本参数配置之后，我们举一些例子作为设计编程的入门，以便大家能够对简单运动编程有个初步了解。

系统配置

指令	解释
KPl0,10,10,10	对 A~D 轴设置增益 KP
KP=10	对所有轴设置增益 KP 的另一种方法
KPA=10	对 A 轴设置增益
KP,20	只对 B 轴设置增益
OEl,1,1,1,1,1,1,1	所有轴设置为超差时使能自动断开
ER=1000	所有轴误差限制值设为 1000 计数单位
KPl0,10,10,10,10,10,10,10	对 A-H 轴设置增益
KP,,10	只对 C 轴设置增益
KPD=10	对 D 轴设置增益的另一种方法
KPH=10	对 H 轴设置增益的另一种方法

简单规划运动

使 A 轴以速度为 20,000 计数单位/sec，加速度、减速度为 100,000 计数单位/sec^2 运动 10,000 计数单位的距离，此例中，电机起动、运转、停止

指令	解释
PRl0000	距离
SP20000	速度
ACl00000	加速度

DCl00000	减速度
BGA	开始运动

多轴

4 个轴分别运动

指令	解释
PR500,1000,6000,400	A、B、C、D 轴的距离
SPl0000,12000,20000,10000	A、B、C、D 轴的速度
ACl0000,10000,10000,10000	A、B、C、D 轴的加速度
DC80000,40000,30000,50000	A、B、C、D 轴的减速度
BGAC	开始 A、C 轴运动
BGBD	开始 B、D 轴运动

独立运动

分别指定运动参数，如下：

指令	解释
PR,300,-600	B、C 轴距离
SP,2000	B 轴速度
DC,80000	B 轴减速度
AC,100000	B 轴加速度
AC,,100000	C 轴加速度
DC,,150000	C 轴减速度
BGC	开始 C 轴运动
BGB	开始 B 轴运动

位置查询

用 TP 指令对各轴的实际位置进行查询

指令	说明
TP	查询所有轴的实际位置
TPA	查询 A 轴的实际位置
TPB	查询 B 轴的实际位置
TPC	查询 C 轴的实际位置
TPD	查询 D 轴的实际位置

用 TE 指令查询位置误差（指令位置——实际位置）

指令	说明
TE	查询所有轴的位置误差
TEA	查询 A 轴的位置误差
TEB	查询 B 轴的位置误差
TEC	查询 C 轴的位置误差
TED	查询 D 轴的位置误差

绝对位置

目的：熟悉指定绝对位置的指令

指令	说明
DP0,2000	定义 A、B 轴当前位置为 0，2000
PA7000,4000	设置想要的绝对位置

BGA	开始 A 轴运动
BGB	开始 B 轴运动

在两个运动完成后，指令 A、B 轴回到零点：

PA0,0	移动到 0，0
BGAB	开始 A、B 轴运动

速度控制

目的：驱动 A、B 轴电机以指定速度运动：

指令	说明
JGl0000,-20000	设定 A、B 轴的 JOG 速度和方向
ACl00000,40000	设定 A、B 轴的加速度
DC50000,50000	设定 A、B 轴的减速度
BGAB	开始 A、B 轴运动

在数秒之后，指令：

JG-40000	设定 A 轴新速度和方向
TVA	读取 A 轴速度

然后：

JG,20000	设定 B 轴新速度
TVB	读取 B 轴速度

以上试验会使速度和方向发生变化。用指令 ST 能够停止运动

ST	停止运动

在转矩限制条件下工作

电机指令的输出幅值可以用指令 TL 进行限制。

指令	说明
TL0.2	设置 A 轴输出限制为 0.2V
JGl0000	设置 A 轴速度
BGA	开始 A 轴运动

在本例中，在放大器以转矩方式工作时，由于输出信号不足以克服磨擦力，A 轴电机有可能不会运动。如果开始运动，用手指就能轻松地使电机停止。注意：当放大器以速度方式工作时，此时只会降低电机转速，但决不可以尝试用手指轻松地使电机停止。

用指令 TL 渐渐增大转矩指令输出，如：

指令	说明
TLl.0	增大转矩限制值到 1V
TL9.98	增大转矩限制值到 9.98V

最大值 9.998V 对应全输出转矩。

查询

可以查询参数值，几个例子如下：

指令	说明
KP?	读取 A 轴比例增益
KP,,?	读取 C 轴比例增益
KP?,?,?,?	读取所有轴的比例增益

也可以查询其它参数，如 KI、KD、FA 等。《指令手册》列出了能够查询的所有指令。

以缓冲器方式工作

在执行下述指令之前，可以缓冲所要执行的指令

指令	说明
PR600000	距离
SPl0000	速度
WTl0000	在读取下一条指令之前，等待 10000msec
BGA	开始 A 轴运动

使用编辑器

可以对运动程序进行编辑，并存储在控制器存储器中。在 GALIL DOS 终端（DMCTERM）上用指令 ED 启动控制器的编辑器。

指令	说明
ED	编辑方式

将操作转入可以对程序进行编写、编辑的编辑器方式。编辑器提供行号。例如，对于第一条指令，其第一行为 0。

行号	指令	说明
000	#A	定义标号
001	PR700	距离
002	SP2000	速度
003	BGA	开始 A 轴运动
004	EN	程序结束

要退出编辑器方式，输入<Cntrl>Q。还可以用指令执行这个程序。

XQ#A	开始执行程序名为#A 的程序

如果从 GALIL Windows 终端软件(如：DTERM32 或 SMARTTERM)发布 ED 指令，软件就会将基于编辑器的窗口打开。通过此编辑器，能够导入、编辑、下载、上载程序。

带有循环的运动程序

可以使用条件跳转指令，对指令执行次序进行控制。

指令	说明
#A	程序标号
DP0	定义当前位置为 0
Vl=1000	设定 V1 的初值
#LOOP	循环标号
PAVl	运动 A 轴电机 V1 个计数单位
BGA	开始 A 轴运动
AMA	在 A 轴运动完成之后
WT500	等待 500msec
TPA	查询并报告 A 轴位置
V1=V1+1000	增加 V1 值
JP#LOOP,V1<10001	若 Vl<10001，重复
EN	结束

在输入完上述指令之后，退出编辑器方式，用<Cntrl>Q。要想开始执行程序，用指令

XQ#A	执行程序#A

带有条件启动的运动程序

运动程序可以包含条件启动，如下

指令	说明
#B	标号
DP0,0	定义初始位置
PR30000,60000	设置目标位置值
SP5000,5000	设置速度
BGA	开始 A 轴运动
AD4000	A 轴移动到 4000 时等待
BGB	开始 B 轴运动
AP6000	A 轴绝对位置=6000 时，等待
SP2000,50000	改变速度
AP,50000	B 轴绝对位置=50000 时，等待
SP,10000	改变 B 轴速度
EN	结束

要想启动此程序，指令：

XQ#B	执行程序#B

控制变量

目的：说明如何利用控制变量

指令	说明
#A;DP0	标号，定义当前位置为 0
PR4000	设置初始位置
SP2000	设定速度
BGA	开始 A 轴运动
AMA	等待 A 轴运动完成
WT500	等待 500msec
#B	
VI=_TPA	决定到 0 的距离
PR-V1/2	指令 A 轴移动 1/2 距离
BGA	开始 A 轴运动
AMA	在 A 轴运动完成之后
WT500	等待 500msec
V1=	报告 V1 值
JP#C,V1=0	如果 VI=0，退出
JP#B	重复执行#B 程序
#C	标号#C
EN	结束

要想执行此程序，指令：

XQ#A	执行程序#A

此程序使 A 轴移动到初始位置 4000，并将其读取回，同时将其值的 1/2 作为新的目标位置。

注意：_TPA 是读取 A 轴位置值的内部变量。其使用方法是在 DMC 指令前加下划线。

直线插补

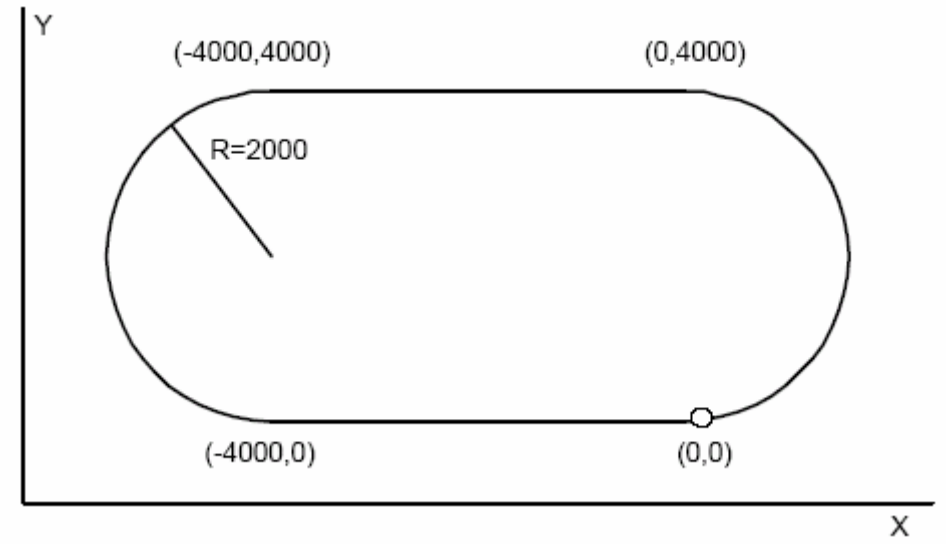
目的：分别使 A、B、C 轴以直线插补方式移动 7000，3000，6000 的距离。正常情况下，三个电机一起启动、停止。

指令	说明
LMABC	指定 A、B、C 轴为直线插补运动
LI7000,3000,6000	直线插补相对距离
LE	直线插补结束
VS6000	矢量速度
VA20000	矢量加速度
VD20000	矢量减速度
BGS	开始运动

圆弧插补

目的：以圆弧插补方式移动 A、B 轴形成下图所示的运动轨迹。注意：矢量运动从位置(0,0)点开始，定义在任何矢量运动程序的开始处。详见后续“应用编程”一章。

指令	说明
VMAB	选择 A、B 轴为圆弧插补运动
VP-4000,0	直线线段运动
CR2000,270,-180	圆弧线段运动
VP0,4000	直线线段运动
CR2000,90,-180	圆弧线段运动
VSl000	矢量速度
VA50000	矢量加速度
VD50000	矢量减速度
VE	矢量程序结束
BGS	开始矢量运动



第三章　连接

概述

DMC-21x3 为前极限、后极限、原点、急停信号提供了兼容 TTL 规格的输入端。控制器还提供了 8 个 TTL 输入和 8 个 TTL 输出用于通用 I/O 控制。对于 5-8 轴控制器，还有额外的 8 个 TTL 输入和 8 个 TTL 输出。

本章为您介绍控制器上的输入和输出信号以及信号的连接。

如果您准备使用 GALIL 或代理商为您提供的标准接口板（如 ICM20100、ICM20105），那么信号的连接与电气描述请以接口板的接线说明为准。

驱动器接口

模拟指令输出 ACMD

其输出信号为相对 GND 的 0—±10V 模拟电压，分辨率为 16bitDAC。当驱动器工作在电流（扭矩）模式下时，应将驱动器设置为所须最大电流对应 10V 指令信号。当驱动器工作在速度模式下时，应将驱动器设置为所须最大速度对应 10V 指令信号。

脉冲指令输出

当需要控制步进电机或工作在位置模式下的伺服电机时，DMC-21x3 提供脉冲/方向信号（标准 TTL 输出），信号的极性与可以通过指令 MT 设置（参见《指令手册》中对 MT 的介绍。

使能输出 AMPEN

DMC-21x3 使能输出为 TTL 信号（当使用接口板配件时，请参考该配件的说明）。当信号有效时，输出高电平。这个信号在以下情况下取消：使用了电机关闭指令 MO；看门狗定时器激活（控制器无响应时）；使用了 OE1 指令，而且位置误差超过了给定的误差极限（ER）。在一些驱动器上，使能信号输入点可能标记为 ENABLE 或 INHIBIT。

编码器信号输入

DMC-21x3 为每轴提供两路编码器信号输入，可以接收旋转编码器、光栅尺或其它增量位置编码器的 TTL 信号（RS422）。DMC-21x3 默认接收正交方波信号，也可以接收脉冲/方向信号，参见《指令手册》中的 CE 指令。

注意：若将某轴配置为脉冲输出工作方式，则该轴不再支持双编码器反馈，该轴辅助编码器输入接口不能作为编码器信号的输入端使用。

主编码器输入口含有 A+、A-、B+、B-、I+、I-接点，辅助编码器输入口只有 A+、A-、B+、B-接点。当不需要两个编码器信号时，辅助编码器信号输入接口也可以用做通用输入信号。

输入信号接口

限位信号

正向限位开关（FLSx）（x 代表轴 A、B、C、D 等）一旦生效，就会立即禁止机械体向正方向运动；而负向限位开关（RLSx）一旦生效就会立即禁止机械体向反方向运动。如果在运动期间，限位开关生效，控制器就会以 DC 指令所设置的减速度使运动减速而停止。在限位开关生效后，不会自动断开使能，电机虽然停止，但仍处于闭环状态，使电机位置保持在锁定状态。可以用 CN 指令来指定信号为高电位有效或低电位有效，详见《指令手册》。

如果用户的 DMC 程序在执行，而且编写了#LIMSWI 子程序，当正、反向限位开关生效时，正在运行的程序会被中断，控制器自动跳转到#LIMSWI 子程序。用户可以将此子程序包含在任何一个运动控制程序中。一旦限位开关生效，对于执行所规定的命令非常有用。详见第六章关于自动子程序的说明。

在限位开关生效后，如果限位开关的逻辑状态未返回到无效状态，就不可能使电机向限位开关所保护的方向进一步运动。在逻辑状态回复之前的任何运动均会导致以下报警：“022-Begin not possible due to limit switch”。

操作数_LFx 和_LRx 分别为正、反向输入限位信号的逻辑状态。1 对应输入限位信号无效，0 对应于输入限位信号有效。使用终端程序，用命令 MG_LFx 或 MG_LRx 可以将限位开关的状态显示在屏幕上。也能够用 TS 命令查询限位开关的状态。详见《指令手册》。

原点信号

原点信号用于在运动控制中提供机械原点。原点信号状态的变化提示控制运动体到达了一个特殊参考点。参考点可以是空间中的一点或编码器标志脉冲（Index 或 Zero 信号）。

原点输入状态相应的操作数为_HMx，0 与 1 分别对应不同的输入状态，详见《指令手册》中对 CN 指令的介绍。

GALIL 控制器支持 3 种原点返回方式：寻找原点开关（FE）、寻找标志脉冲（FI）和标准原点返回方式（HM）。

寻找原点开关

原点开关是整个行程中用于确定坐标的参考点，它可以是一个机械式的开关、一个光电开关或者一个接近开关。寻找原点开关的过程用如下命令来进行：

FEA<Return>

BGA<Return>

此程序会使电机加速到恒定速度运动，一旦检测到原点开关输入逻辑状态发生变化，就使电机减速停止，此点即为原点。FE 运动的方向取决于原点开关的状态与 CN 指令的设置。用户事先可以用命令，AC、DC、SP 设置加速度、减速度和速度。建议使用大的减速度，以便在检测到原点开关输入信号后，电机能够快速停止。

寻找标志脉冲

原点开关在整个行程中用作确定坐标的参考信号，精确的原点位置是运动系统重复精度的基础，而一般用作原点信号的器件，其本身的重现性不足以保证系统的重复性时，可以选择以编码器的标志脉冲位置作为坐标的参考点，一般来说，这样可以使原点的重复精度接近编码器本身的分辨率（当使用光栅尺作为编码器信号输入时，请参考光栅尺的规格说明）

寻找标志脉冲程序用如下命令：

JG1000<Return>

FIA<Return>

BGA<Return>

此程序会使电机加速到用户所指定的速度运动，一旦控制器检测到标志脉冲信号由低变高，电机就减速停止。电机的回零速度和方向用 JG 命令来指定，加速度、减速度分别用 AC、DC 命令来设置。尽管寻找标志脉冲是原点返回操作的选择，与原点机械开关输入的逻辑状态变化无关，只与标志脉冲信号的电平变化密切相关。

标准原点返回方式

对于以上两种方式，前者的以整个行程范围内唯一的开关信号来确定坐标，但重复精度较低，而后者的重复精度较高，但在很多应用中，在整个行程范围内，有多个编码器标志脉冲的位置。而标准的回原点流程通过两者的结合来确定唯一的、高重复精度较高的参考点。

标准原点返回方式用如下命令来进行：

HMA<Return>

BGA<Return>

电机以所设定的速度运动，一旦检测到原点开关输入逻辑状态变化，电机就会减速停止，然后，电机就会改变方向，以 256 计数单位/sec 的速度逼近原点开关，当此开关的逻辑状态再次变化时，电机以同样的速度向正向运动(计数值增大的方向)，控制器一检测到标志脉冲(Index)，就命令电机减速停止，并将此位置定义为 0 点(机械原点)。

相关原点返回操作的进一步说明及举例请参考“运动编程”一节和《指令手册》中的 HM、FI、FE 指令介绍

急停输入

急停输入的功能是一旦此输入点逻辑状态发生变化就立即使控制器停止运行。

注 1：急停输入的响应方式不同于限位开关输入有效时的情况。当急停输入生效时，控制器立即停止执行运动命令，而限位开关输入生效时，控制器会使电机减速逐渐停止。

注 2：急停输入的效果与各轴超差关断使能的状态设置有关。如果轴超差关断使能有效，当急停信号发生时，就会关断该轴使能。此时，由于伺服系统不再处于闭环控制之下，所以电机会依靠惯性停止。如果超差关断使能设置为无效，电机就会尽可能快地减速停止，此时，电机仍然保持在伺服闭环状态（有可能产生抖动）。

当检测到急停输入状态变化时，控制器就终止当前正在运行的所有运动程序。进一步的说明请参见《指令手册》中的 OE 命令。

复位输入

此输入为低电平时，控制器将执行复位操作，所有内部状态恢复到上电时的状态。

通用输入

DMC-21x3 有 8 个通用 TTL 输入点（对于 5-8 轴控制器，有 16 个 TTL 通用输入点），这些输入点可以用做通用输入，用户可以用这此输入点建立与外部事件的任意连接。同时，这些输入点也可以用作特定轴的专用输入。1-4 点可以作为 A-D 各轴的位置锁存输入，其中第 1 点对应 A 轴、第 2 点对应 B 轴……（关于位置锁存功能的详细介绍，参见 “编程”一章中相应的部分，以及《指令手册》中 AL、RL 指令的介绍）；第 5-8 点可以作为 A-D 各轴独立的急停输入，其中第 5 点对应 A 轴、第 6 点对应 B 轴……（详见《指令手册》中 CN 指令）。对于 5-8 轴的控制器，9-16 输入点对于 E-H 轴也有相同的作用。

这些输入点的状态可以用 TI 指令查询（详见《指令手册》，也可以函数@IN[x]分别读取这些输入点状态，其中 x 代表输入点号（1-8，或 1-16）。

辅助编码器信号接口作为通用输入

DMC-21x3 为每轴提供两路编码器信号输入，当不需要两个编码器信号时，辅助编码器信号输入接口也可以用做通用输入信号。辅助编码器输入分配为输入点 81-96。A 轴辅助编码器

的 A 通道为 81、B 通道为 82……，用函数@IN[81]-@IN[96]读取这些输入点的状态。

注意：若将某轴配置为脉冲输出工作方式，则该轴的辅助编码器输入接口不能作为编码器信号的输入端使用，但依然可以作为通用输入点使用。

输出信号接口

DMC-21x3 提供特定的专用输出和供用户定义的通用输出功能

通用输出

DMC-21x3 提供 8 点 TTL 通用输出信号（对于 5-8 轴控制器，为 16 点）。这些点定义为 OUTI~OUT8（或 OUT1-OUT8）。用指令 SB（设置位）、CB（清除位）、OB（输出位）、OP（输出端口）控制这些输出点。这些指令的具体内容，请参考《指令手册》

报警输出 ERR

当控制器出现报警时，红色 LED 灯点亮，报警输出 ERROR 变低。以下几种原因可能会导致控制报警：

a．位置误差超过设定的误差极限；

b．控制器受到复位信号；

c．控制器损坏，处理器处于自检状态；

d．驱动报警输出的 IC 芯片损坏

位置比较输出 CMP

比较输出信号由控制器主编码器的位置反馈值与 OC 命令所设置的比较值相比较而产生。每当反馈值跨越比较值时，在 CMP 端产生一个低脉冲（脉冲宽度 0.6usec）。进一步信息请参考《指令手册》中 OC 的说明。

扩展 I / O

DMC-21x3 提供可选的 40 点扩展 3.3V 的 I/O 和 8 点 12BIT 模拟输入。如需要这些功能，须订购 DB-28040 扩展卡（可能需要相应的连线接口板）。这些 I/O 点每 8 个一组，由用户定义其用作信号输入还是信号输出。具体操作参见第七章及《指令手册》中 CO 指令

第四章　通讯

说明

DMC21x3 有一个的 RS-232 接口和一个 10M 的 Ethernet 接口。其中，RS-232 是数据设备（DATASET）接口。

RS-232 接口

RS-232 口的引脚定义如下所示。

1	CTS—输出	6	CTS—输出
2	TD—输出	7	RTS—输入
3	RD—输入	8	CTS—输出
4	RTS—输入	9	NC
5	GND		

如果与 PC 的 9 芯串行接口连接，只要使用 9 芯直通的连线即可，如果与其它设备的串行接口相连，则至少需要 5 根连接线。

RS-232 配置

将您的 PC 机（或其它需要与 DMC-21x3 相连接的设备）上的串行数据格式配置成 8 个数据位，1 个起始位，1 个停止位，全双工，无校验方式。

可以用控制卡上的 JP2 跳线来设置 RS-232 通信口的波特率。如下：

1200	9600	波特率
开放	短路	1200
短路	开放	9600
短路	短路	19200

Ethernet 配置

通信协议

Ethernet 是以数据包为单位传送信息的本地网络。因此，需要通信协议来规定如何发送、接收这些包。DMC-21x3 支持两种工业标准协议，TCP/IP 和 UDP/IP。控制器以所接触的格式启动响应。

TCP/IP 是一种“应答”协议。为了开始通信，必须将主站与从站相连。当收到发送的数据包时，会告知已收到。若未收到，告知回应，就认为此包丢失，并自动重新发送。

与 TCP/IP 协议不同的是，UDP/IP 本身不需要“应答”。此协议类似于用 RS-232 的方式进行通信。如果丢失信息，控制器不返回冒号或问号。由于此协议不提供关于丢失数据的保护，因此，发送方必须自己重发数据包。

尽管 UDP/IP 更为高效，简单，但 GALIL 建议使用 TCP/IP 协议。如果在传送当中，数据

包出现丢失或破坏 TCP/IP 协议会自动重发这些数据包，可以保证通讯更为可靠。

Ethernet 通信以数据包传送信息，数据包必须限定在 470 数据字节以内。更大的包可能引起控制器通信阻塞。

注：为了不丢失传送信息，GALIL 建议用户在发送下一个包之前等待包接收的回答信息。

地址分配

Ethernet 设备的地址分配包括三层。第一层是 Ethernet 或硬件 MAC 地址。这是一个唯一且永久的 6 字节数。任何装置之间不会有相同的 Ethernet 地址。DMC-21x3 的硬件 MAC 地址由厂家设置，此地址的最后两个字节是控制器的序列号。

第二层是 IP 地址。它是一个 32 位（4 字节）数。IP 地址由各局域网络规定，可以用多种方法对控制器分配 IP 地址。

第一种方法是使用由 Ethernet 连接的 BOOT-P 软件工具（DMC-21x3）必须连接到网络中并上电）。关于 BOOT-P 的简要说明参见“第三方软件”一节。可以使用内部网络上的 BOOT-P 服务器或 GALIL 终端软件。要使 GALIL BOOT-P 工具软件，就在终端软件中选择控制器注册表。在添加新控制器中选择 DMC-21x3，然后选择 Ethernet 方式，并选择 TCP/IP 或 UDP/IP 作为通信协议。至此，点击 ASSIGN IP ADDRESS。GALIL 终端软件就搜索连接到当前局域网上所有 GALIL 控制器；用户选定控制器，就可以软件就会给控制器分配所指定的 IP 地址。然后，进入终端软件，键入 BN 并回车以保存 IP 地址到控制器的非易失性存储器中。

注意：确保只有一个 BOOT-P 服务器在运行。如果您的网络有 DHCP 或 BOOT-P 在运行，它就可以自动分配 IP 地址给连接到网络中的控制器。为了确保 IP 地址正确无误，在把控制器连接到 Ethernet 网络之前，请与您的系统管理员联系。

第二种设置 IP 地址的方法是通过 DMC-21x3 的 RS-232 串口发送 IA 命令，输入要分配的 IP 地址，地址为 4 字节数，中间用逗号分开或带符号 32 位数（例如，IA124,51,29,31 或 IA2083724575），输入 BN 并回车，将 IP 地址保存到控制器的非易失性存储器中。

注意：GALIL 特别推荐，所选择的 IP 地址是不需要通过 GATEWAY 存取的地址（GATEWAY 是管理内部网络和外部世界通信的应用软件）。在分配选定 IP 地址时，请先确认当前局域网的 IP 地址和子网掩码，确保分别给 GALIL 控制器的地址于使用的 PC 的地址在同一个域内。

Ethernet 地址分配的第三层是 UDP 或 TCP 接口号。GALIL 控制器不需要指定接口号。每次连接控制器时，接口号由终端或主站建立。

与多个设备通讯

DMC-21x3 能够支持多主多从通讯，主装置可以是多台 PC 机，发送命令给控制器；典型的从装置是外围 I/O 装置，从控制器接收命令。

注意：“主”等效于互联网的“Client”，“从”等效于互联网“server”。

Ethernet 句柄（handle）是在设备内的通讯资源。DMC-21x3 同时最多能够有 8 个 Ethernet 句柄开放。使用 TCP/IP 时，每个主或从使用各自的 Ethernet 句柄。在 UDP/IP 中，一个句柄可以用于所有主装置，但一个从装置占用一个句柄（Pings 和 ARP 不占用句柄）。如果所有 8 个句柄均被占用，而第 9 个主设备试图接入，就会发送“reset packet”，在其应用程序中产生相应的报警。

注意：有许多方法对控制器进行复位，硬件复位（按下复位按钮或对控制器断电）及软件复位（通过 Ethernet 或 RS232 发送 RS 指令）。不会使控制器断开连接的唯一复位是由 Ethernet 进行软件复位。

当 GALIL 控制器作为主设备使用时，用 IH 命令来分配句柄并连接到相应的从设备。IP 地址按 4 字节数，中间用逗号分开输入，或者输入带符号 32 位数。也可以指定端口（port）号，但如果不指定，其缺省值为 1000。同时指定使用的通讯协议（TCP/IP 或 UDP/IP）。否则，

控制器就不会连接到从设备。例如：IHB=151，25，255，9<179>2，这样就会打开句柄#2，并连接到 IP 地址 151，25，255，9，端口号为 179，使用 TCP/IP 协议。

为了与 I/O 装置对话，可以用附加协议层。Modbus 是 RS485 协议，它对信息以二进制数据包，作为 TCP/IP 数据包的一部分进行传送。在这个协议中，各从装置有 1 字节从地址。DMC-21x3 可以使用特定的从地址或句柄号的设定。Modbus 的端口号是 502。

Modbus 协议有一组称之为功能码的命令。DMC-2100 / 2200 支持 10 个主要功能码：

功能码	定义
01	读线圈状态（读位）
02	读输入状态（读位）
03	读保持寄存器（读字）
04	读输入寄存器（读字）
05	强置单线圈状态（写 1 位）
06	预置单寄存器（写 1 个字）
07	读异常状态（读错误码）
15	强行设置多线圈（写多个位
16	预置多个寄存器（写多个字）
17	报告从设备 ID

DMC-21x3 提供三级 Modbus 通信。第一级允许用户创建数据包并接收数据。它使用 MBh 指令，功能码为-1。指令格式为：

MBh=-1，Len，array[]

其中：Len 是字节数；array[]是含数据的数组

第二级包含 Modbus 结构。这需要发送配置和特殊命令给 I/O 装置。格式变化取决于调用的功能码。更为详细的说明请参考《指令手册》。

Modbus 通信的第三级使用标准 GALIL 命令。配置为从装置以后，可以使用的命令有：@IN[]，@AN[]，SB，CB，OB 和 AO。例如，AO2020,8.2 就会告诉 I/O 号 2020 输出 8.2V 电压。

如果不需要规定的从地址，要使用的 I/O 号能够用下式求得：

I/O 号：（HandleNum*1000）+（（Module-1）*4）+（BitNum-1）

其中：HandleNum 是句柄号。从 1(A)-6(F)。Module 是模块在插槽中的位置，从 1-16。BitNum 是模块中的 I/O 点，从 1~4。如果使用明确的从地址，方程式变为：

I/O 号：（SlaveAddress*1000）+（HandleNum*1000）+（（Module-1）*4）+（BitNum-1）

哪个装置接收来自控制器的什么信息取决于许多因素。如果一个装置对控制器发问，如果它不告诉控制器将此发问发送到另一个装置，它就会收到响应。如果产生响应的命令是部分下载的程序，响应就会按路线图传送到由 ENET 开关缺省规定的那个口（如果没有准确告诉去另一个口），ENET 开关“ON”指定 Ethernet 在进入最后端口与控制器进行通信，“OFF”指定主 RS232。要对信息指定规定的目的地，就将{Eh}添加到命令的尾部。例如，MG{EC}“Hello”就将信息“Hello”发送到端口#3。TP,,?{EF}将 Z 轴位置发送到端口#6。

对 ModBus 的详细介绍，请自行参考相关资料

多点传送

多点传送只用于 UDP/IP，它类似于广播（处于网络上的每个人都会获得信息），但指定给一组。换而言之，在所指定组内的所有设备都会接收到以多点方式发送的信息。网络上可以有许多个多点传送组，由其多点传送 IP 地址加以区分。要想与处于一个指定的多点发送组中的所有装置进行通信，就可以将信息发送到多点发送 IP 地址而不是各个设备的 IP 地址，所有 GALIL 控制器的多点发送缺省地址为 239，255，19，56，用 IA>u 命令能够更改控制器的多点发送 IP 地址。

使用第三方软件

GALIL 控制器支持 ARP，BOOT-P 和 Ping 等软件工具来建立 Ethernet 连接。ARP 是一个应用软件。用它来以指定的 IP 地址确定装置的 Ethernet(硬件)地址。BOOT-P 是用来确定网络上的哪个装置没有 IP 地址并将您所选择的装置分配 IP 地址。Ping 用于检查具有指定 IP 地址的装置与主计算机之间的通信。

DMC-21x3 能够通过可发送 TCP/IP 或 UDP/IP 包的任何应用软件与主计算机进行通信。一个很好的例子是 Telnet，它是与大多数 Windows 系统捆绑的工具软件。

数据记录

DMC-21x3 通过使用单个命令 QR 就能够提供许多状态信息。此命令与 QZ 命令配合使用对于读取整个控制器状态非常有用。QR 命令会返回 4 字节头信息和通过命令参数所指定的指定块信息如：QR ABCDEFGHST。各个参数按照如下数据记录表对相应的块信息。如果不给出参数，就返回整个数据记录表。注意：数据记录空间大小取决于轴数。

数据记录结构

数据类型	内容分类	块
UB	头信息的第 1 字节	头
UB	头信息的第 2 字节	头
UB	头信息的第 3 字节	头
UB	头信息的第 4 字节	头
UW	采样数	I
UB	0 组通用输入信号	I
UB	1 组通用输入信号	I
UB	2 组通用输入信号	I
UB	3 组通用输入信号	I
UB	4 组通用输入信号	I
UB	5 组通用输入信号	I
UB	6 组通用输入信号	I
UB	7 组通用输入信号	I
UB	8 组通用输入信号	I
UB	9 组通用输入信号	I
UB	0 组通用输出信号	I
UB	1 组通用输出信号	I
UB	2 组通用输出信号	I
UB	3 组通用输出信号	I
UB	4 组通用输出信号	I
UB	5 组通用输出信号	I
UB	6 组通用输出信号	I
UB	7 组通用输出信号	I
UB	8 组通用输出信号	I
UB	9 组通用输出信号	I
UB	错误代码	I
UB	通用状态	I

UW	插补 S 的线段数	S
UW	插补 S 的状态	S
SL	插补 S 运动过的距离	S
UW	插补 T 的线段数	T
UW	插补 T 的状态	T
SL	插补 T 运动过的距离	T
UW	X（A）轴运动状态	A
UB	X（A）轴 I/O 状态	A
UB	X（A）轴停止代码	A
SL	X（A）轴参考位置	A
SL	X（A）轴电机位置（主编码器位置）	A
SL	X（A）轴位置误差	A
SL	X（A）轴辅编码器位置	A
SL	X（A）轴速度	A
SW	X（A）轴输出模拟电压	A
SW	0 或 X（A）轴输入模拟量（使用 DB28040）	A
UW	Y（B）轴运动状态	B
UB	Y（B）轴 I/O 状态	B
UB	Y（B）轴停止代码	B
SL	Y（B）轴参考位置	B
SL	Y（B）轴电机位置（主编码器位置）	B
SL	Y（B）轴位置误差	B
SL	Y（B）轴辅编码器位置	B
SL	Y（B）轴速度	B
SW	Y（B）轴输出模拟电压	B
SW	0 或 Y（B）轴输入模拟量（使用 DB28040）	B
UW	Z（C）轴运动状态	C
UB	Z（C）轴 I/O 状态	C
UB	Z（C）轴停止代码	C
SL	Z（C）轴参考位置	C
SL	Z（C）轴电机位置（主编码器位置）	C
SL	Z（C）轴位置误差	C
SL	Z（C）轴辅编码器位置	C
SL	Z（C）轴速度	C
SW	Z（C）轴输出模拟电压	C
SW	0 或 Z（C）轴输入模拟量（使用 DB28040）	C
UW	W（D）轴运动状态	D
UB	W（D）轴 I/O 状态	D
UB	W（D）轴停止代码	D
SL	W（D）轴参考位置	D
SL	W（D）轴电机位置（主编码器位置）	D
SL	W（D）轴位置误差	D
SL	W（D）轴辅编码器位置	D
SL	W（D）轴速度	D
SW	W（D）轴输出模拟电压	D
SW	0 或 W（D）轴输入模拟量（使用 DB28040）	D

UW	E 轴运动状态	E
UB	E 轴 I/O 状态	E
UB	E 轴停止代码	E
SL	E 轴参考位置	E
Sl	E 轴电机位置（主编码器位置）	E
SL	E 轴位置误差	E
SL	E 轴辅编码器位置	E
SL	E 轴速度	E
SW	E 轴输出模拟电压	E
SW	0 或 E 轴输入模拟量（使用 DB28040）	E
UW	F 轴运动状态	F
UB	F 轴 I/O 状态	F
UB	F 轴停止代码	F
SL	F 轴参考位置	F
SL	F 轴电机位置（主编码器位置）	F
SL	F 轴位置误差	F
SL	F 轴辅编码器位置	F
SL	F 轴速度	F
SW	F 轴输出模拟电压	F
SW	0 或 F 轴输入模拟量（使用 DB28040）	F
UW	G 轴运动状态	G
UB	G 轴 I/O 状态	G
UB	G 轴停止代码	G
SL	G 轴参考位置	G
SL	G 轴电机位置（主编码器位置）	G
SL	G 轴位置误差	G
SL	G 轴辅编码器位置	G
SL	G 轴速度	G
SW	G 轴输出模拟电压	G
SW	0 或 G 轴输入模拟量（使用 DB28040）	G
UW	H 轴运动状态	H
UB	H 轴 I/O 状态	H
UB	H 轴停止代码	H
SL	H 轴参考位置	H
SL	H 轴电机位置（主编码器位置）	H
SL	H 轴位置误差	H
SL	H 轴辅编码器位置	H
SL	H 轴速度	H
SW	H 轴输出模拟电压	H
SW	0 或 H 轴输入模拟量（使用 DB28040）	H

注：

UB=无符号字节（1 字节）；UW=无符号整数（2 字节）

SW=整数（2 字节）；SL=长整数（4 字节）

状态信息说明

头信息的 0、1 字节

表明数据记录中包括那些轴的状态信息

BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
1	保留	保留	保留	保留	存在 I	存在 T	存在 S
BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
存在 H	存在 G	存在 F	存在 E	存在 D	存在 C	存在 B	存在 A

头信息的 2、3 字节

2、3 两字节是一个整数，描述了整个数据记录中有多少字节（包括头信息）。2 字节为低位，3 字节为高位。

通用状态信息

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
程序运行	保留	保留	保留	保留	IN 指令等待输入	跟踪开启	回响开启

轴 I/O 状态信息

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
位置已锁存	锁存输入状态	保留	保留	正限位状态	反限位状态	原点输入状态	SM 跳线已短路

轴运动状态信息

BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
运动中	PR 或 PA 模式下	PA 模式下	FE 过程中	HM 过程中	HM 第一步完成	HM 第二步完成或 FI 过程中	参与插补运动
BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
负方向运动	参与轨迹模式运动	匀速运动	由于 ST 指令或限位开关而减速中	减速停止过程中	提供位置锁存	因误差过大而关闭电机	电机关闭

插补运动状态信息

BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
运动中	保留	保留	保留	保留	保留	保留	保留
BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
保留	保留	匀速运动	由于 ST 指令或限位开关而减速中	减速停止过程中	保留	保留	保留

关于速度、输出模拟电压信息

在数据记录中返回的速度信息是使用 TV 指令所返回值的 64 倍。请参见《指令手册》中相关 TV 命令的说明。

输出模拟电压信息表达为±32767 范围内的数。最大反向输出为-32767。最大正向输出为 32767。零输出为 0。

QZ 命令

当使用 QR 命令时，QZ 命令就非常有用，这是因为 QZ 命令提供了有关控制器和数据记录方面的信息。QZ 命令返回如下 4 字节信息：

字节	信息
0	轴数

1	数据记录中块 I 的字节数
2	数据记录中块 S、T 的字节数
3	数据记录中各轴数据块的字节数

控制器对指令的响应

大多数 DMC-21x3 指令用跟随有参数的两个字符来表达，各指令必须用回车或分号来结束。

指令以 ASCII 码发送，DMC-21x3 对各 ASCII 字符逐一进行译码。控制器对每条指令译码大约占用 0.5msec 时间。不过，由于使用 FIFO 缓存，因此，PC 机能以更快的速率给控制器发送数据。在对指令进行译码之后，DMC-21x3 回复一个响应给发送指令的口。如果命令是有效的话，控制器就返回一个（:），而当命令是无效的时，则返回一个（?）。由 RS232 口发送的命令回复到 RS232 口；而由 Ethernet 口发送的命令回复到 Ethernet 口。

对于返回数据的那些指令，如 TP，DMC-21x3 就会返回数据，回车符，换行符和冒号。

在发送各命令之后，为防出错，实际的应用中最好用收到冒号加以确认。控制器提供回响功能就能使 DMC-21x3 与发送的数据响应相协调。通过发送命令 EO1 给控制器，就使回响功能有效。

由控制器产生的非请求信息

控制器正在执行程序时，可以通过 RS232 口或 Ethernet 发送信息。这些信息可能是程序中使用 MG、IN 指令发出的字符或指令错误信息。由于这些响应不是对外部命令的直接响应，因此，称其为非请求信息。

通过使用 CF 指令或者在使用特定端口说明（参见《指令手册》中 MG 与 IN 的说明）可以指定这些信息是通过串口还是通过 Ethernet 口发出。如果未明确指定口，那么这些信息会从串口发出。

控制器有一个特殊指令 CW，它能够影响非请求信息的格式。此指令可以让 GALIL 软件区别控制器发出的信息是指令的响应还是非请求信息相。指令 CW1 使控制器把非请求信息 ASCII 字符的高位设置成 1。这样对有些终端可能引起字符混肴。用指令 CW2 使此功能无效。有关详细说明，参见《指令手册》中的 CW 命令。

GALIL 软件工具和库文件

GALIL 向用户提供 API（应用编程接口）软件。API 软件用 C 写成，包含在 GALIL CD-ROM 中，用户能够在 WINDOWS 环境下用 API 进行二次应用开发。用 API 软件，用户能够把已存在的库函数直接合并到 C 程序中。

GALIL 也开发了 Activex Tool Kit 套件，来处理包括支持中断在内的所有通信，这些目标码直接装到 VB Labview C 或接受 Activex 控件的任何软件包中并作为实时控制软件的一部分。

第五章　指令

说明

在了解了运动控制器系统的构成、硬件连接、通讯之后，本章将对运动控制器的指令作一些基础性的介绍，以便用户在后续章节中能够深入领会运动编程及应用开发中所涉及到的指令。

DMC-21x3 为用户提供了 100 多条指令用于运动控制和参数设置。这些指令涉及到初始化、查询状态、配置数字滤波器等方面，而且这些指令能以 ASCII 码或二进制码发送。

这些指令就类似于 BASIC，使用起来非常容易。指令由两个大写字母组成，与相应的功能相对应，便于记忆。例如：指令 BG 指开始运动（Begin），ST 指停止运动（Stop）。大部分指令可以使用二进制码方式，用$80~$FF 之间的二进制码表示。

用户能够实时通过通信口发送 ASCII 指令，由 DMC-21x3 控制器立即执行，或者将整个一组指令作为程序下载到 DMC-21x3 存储器中，在需要的时候执行。将指令编写成程序执行，这方面的内容请参考“编程”一章。在应用编程中，指令必须使用 ASCII 码形式。

本章重点描述 DMC-21x3 指令集和语句。指令总集及 DMC-21x3 全部指令列表参见《指令手册》。

指令语法——ASCII 码

DMC-21x3 指令用两个 ASCII 大写字符后跟可用参数来表达，可以在指令和参数之间插入空格，用分号或回车来终止指令。

注意：如果您使用 GALIL 终端工具软件来编写程序，只有在给出<return>回车指令后，才会处理这些指令。这样就允许用户在单行内用分号把许多指令分开，直到用户给了<return>指令，才开始执行。

重要提示：所有 DMC-21x3 指令均以大写字符发送。

例如，指令

PR 4000<return>　相对位置

PR 是用于指定相对位移量的两字符指令，4000 是以计数单位表示所要运动的距离。<return>使该条指令结束。PR 和 4000 之间的空格为可选择（可有也可无）。

为了对 A,B,C,D 轴指定数据，就用逗号将轴分开，如果对一个轴没有指定数据，但仍然需要逗号，如下例所示，在此情况下，该轴对应的参数维持原有的值。

要想查看各指令的当前值，就对所提问的各轴用“?”跟随在指令之后。

PRl000	只对 A 轴指定 1000
PR,2000	只对 B 轴指定 2000
PR,,3000	只对 C 轴指定 3000
PR,,,4000	只对 D 轴指定 4000
PR2000,4000,6000,8000	对 A，B，C，D 轴指定位移值
PR,8000,,9000	只对 B，D 轴指定位移值
PR?,?,?,?	查询 A，B，C，D 轴位移值
PR,?	只查询 B 轴位移值

DMC-21x3 提供了好几种方法对各轴指定数据，下面是用单个轴指定器（如 A，B，C，D

等)单独指定数据的举例。用等号来对该轴分配数据。

举例如下：

PRA=1000	对 A 轴指定相对位移量 1000
ACB=200000	对 B 轴指定加速度 200000

与分配数据不同，有些指令会要求一个轴或多个轴动作。例如，ST AB 就使 A 轴和 B 轴停止运动，在此情况下，由于用相对应的字母 A,B,C,D 指定相应轴，因此就不需要逗号。如果没有参数跟在指令后面，那么就会使所有轴发生动作。以下是有关这方面的举例：

BG A	只启动 A 轴
BG B	只启动 B 轴
BG ABCD	启动所有轴
BG BD	只启动 BD 轴
BG	启动所有轴
BG ABCDEFGH	启动所有 8 个轴
BG D	只启动 D 轴

一轴以上的联动控制

当要求控制插补运动时，用字母 S 和 T 来指定插补自然坐标系。例如：

BG S	开始插补运动，在 S 坐标系
BG TW	开始插补运动序，在 T 坐标系，指令 W 轴运动

指令语法——二进制码

有些指令有等效的二进制数，通信方式的执行速度比 ASII 码指令一些。二进制格式只能用于从 PC 机发送指令时，他不能嵌入下载到控制器的应用程序中。

二进制指令格式

二进制指令都有 4 字节的头信息和跟随的数据域。4 个字节以十六进制格式指定。

头信息格式：

字节 1：指定 80~FF 之间的指令代码，完整的二进制指令代码表列表如下。

字节 2：指定各个域中的字节号，0，1，2，4，6，如下：

00	本指令无数据域（例如 SH 或 BG）
01	每域 1 个字节
02	每域一个字（2 字节）
04	每域一个长字（4 字节）
06	GALIL 实际格式（整数和小数）

字节 3：指定指令是否适用于插补运动控制，如下：

00	非插补运动控制
0l	插补运动控制

例如，指令 STS 使插补运动停止，等效二进制指令的第 3 字节为 01。

字节 4：指定轴号或数据域，如下：

Bit7=H 轴或第 8 数据域
Bit6=G 轴或第 7 数据域
Bit5=F 轴或第 6 数据域
Bit4=E 轴或第 5 数据域
Bit3=D 轴或第 4 数据域
Bit2=C 轴或第 3 数据域

Bitl=B 轴或第 2 数据域

Bit0=A 轴或第 1 数据域

数据域格式

数据域必须与格式字节和轴字节相一致。例如，指令 PR l000,,-500 为：A7 02 00 05 03 E8 FE 0E 其中：

A7　　是 PR 的二进制码

02　　指定各数据域为 2 字节

00　　S 对 PR 无效

05　　指定 bit0 对 A 轴有效，bit2 对 C 轴有效（$2^0+2^2=5$）

03E8　　表示 1000

FEOE　　表示-500

举例：指令 ST ABCS 为：A1 00 01 07 其中：

A1　是 ST 指令的二进制码

00　指定 0 数据域

01　指定停止联动控制轴 S

07　表示停止 X(bit0)，Y(bit1)和 Z(bit2)轴，（$2^0+2^1+2^2=7$）

二进制指令表

代码	指令	代码	指令	代码	指令	代码	指令
80	保留	A0	BG	C0	ET	E0	TI
81	KP	A1	ST	C1	EM	E1	SC
82	KI	A2	AB	C2	EP	E2	保留
83	KD	A3	HM	C3	EG	E3	保留
84	DV	A4	FE	C4	EB	E4	保留
85	AF	A5	FI	C5	EQ	E5	TM
86	KF	A6	PA	C6	EC	E6	CN
87	PL	A7	PR	C7	保留	E7	LZ
88	ER	A8	JG	C8	AM	E8	OP
89	IL	A9	MO	C9	MC	E9	OB
8A	TL	AA	SH	CA	TW	EA	SB
8B	MT	AB	保留	CB	MF	EB	CB
8C	CE	AC	保留	CC	MR	EC	II
8D	OE	AD	保留	CD	AD	ED	EI
8E	FL	AE	保留	CE	AP	EE	AL
8F	BL	AF	保留	CF	AR	EF	保留
90	AC	B0	LM	D0	AS	F0	保留
91	DC	B1	LI	D1	AI	F1	保留
92	SP	B2	VP	D2	AT	F2	保留
93	IT	B3	CR	D3	WT	F3	保留
94	FA	B4	TN	D4	WC	F4	保留
95	FV	B5	LE/VE	D5	保留	F5	保留
96	GR	B6	VT	D6	保留	F6	保留
97	DP	B7	VA	D7	保留	F7	保留
98	DE	B8	VD	D8	RP	F8	保留

99	OF	B9	VS	D9	TP	F9	保留
9A	GM	BA	VR	DA	TE	FA	保留
9B	保留	BB	保留	DB	TD	FB	保留
9C	保留	BC	保留	DC	TV	FC	保留
9D	保留	BD	CM	DD	RL	FD	保留
9E	保留	BE	CD	DE	TT	FE	保留
9F	保留	BF	DT	DF	TS	FF	保留

控制器对指令的响应

DMC-21x3 控制器对于有效指令返回冒号（:），而对无效指令返回问号（?）。例如，如果 BG 指令以小写字母发送，DMC-21x3 就会返回“?”

当控制器收到无效指令时，用户能查询错误代码，错误代码会指明出现无效指令响应的原因。要查询错误代码，键入指令 TCl。

接收到无效指令响应有多种原因，最常出现的原因是：指令拼写错误（小写字母或使用了带格式的文本）；给出时间不合适（例如在运动期间）；或超出指令值范围（例如超出最大速度）。所有出错代码列表见《指令手册》中 TC 指令部分。

查询控制器

查询指令

DMC-21x3 有一组直接查询控制器的指令，当输入指令并回车后，以十进制格式返回所要查询的数据。所返回数据的格式能够用位置格式（PF）、变量格式（VP）和前导零（LZ）指令进行修改，详见第 7 章和《指令手册》。

查询指令汇总

指令	说明
RP	参考位置
RL	锁存位置
^R^V	控制器内部固件版本
SC	停止代码
TB	状态
TC	错误代码
TD	辅编码器位置（或发送脉冲位置）
TE	位置误差
TI	通用输入点状态
TP	主编码器位置
TR	轨迹
TS	轴状态
TT	转矩值（输出模拟量）
TV	速度值（编码器反馈速度）

下面的例子说明如何显示 X 轴当前位置值：

TPA<return>　　读取 A 轴位置

0000000000	控制器响应
TP AB <return>	读取 AB 轴位置
0000000000,0000000000	控制器响应

查询当前指令值

大多数指令均能用问号(?)作为参数进行查询。对想要查询的各轴键入用?号跟随的指令，如：

PR?，?，?，?	查问A，B，C，D轴的位置值
PR，?	查问B轴的位置值

操作数

大多数DMC-21x3指令均有可用于查询相对应值的操作数，操作数必须用于有效的DMC表达式里。例如，要显示操作数的值，用户可以用指令：MG “Operand”其中“Operand”是有效DMC操作数。

所有指令操作数均以下划线符（_）开始。例如，将A轴当前位置值用指令分配给变量“V”

```
V=_TPA
```

《指令手册》中解释了有等效操作数的所有指令。也可参见第7章中关于操作数的描述。

第六章　编程

概述

DMC-21x3 提供了好几种运动方式，它包括定位控制、手动、插补控制运动、电子凸轮运动、电子齿轮等。用户可以非常方便的利用这些现有的运动方式根据控制要求编写出应用程序。另外，GALIL 还提供了一些与运动状态关系密切的特殊功能，本章将对这些运动方式和功能逐个进行讨论。

以下举一些例子以帮助您进一步理解有关的运动方式

运动实例	运动方式	涉及的指令
通过速度控制，没有定位要求	独立进给	JG、AC、DC、ST
绝对或相对的定位，各轴独立运动，速度曲线为梯形包络线	独立定位	PA、PR、SP、AC、DC
按照任意指定的位置-时间曲线运动	轨迹运动	CM、CD、DT、WC
由圆弧和线段组成的 2 维运动	平面内曲线插补	VM、VP、CR、VS、VR、VA、VD、VE
由圆弧和直线组成的 2 维运动，同时跟踪运动的方向矢量	含角度跟踪的平面曲线插补	VM、VP、CR、VS、VR、VA、VD、TN、VE
多维空间曲线运动	多维插补运动	LM、LI、LE、VS、VR、VA、VD
两轴以一定的比例同步运动	电子齿轮同步运动	GA、GR、GM
从动轴完全按照主动轴的方式运动	龙门跟随同步运动	GA、GR
按照任意数学模型所预制的轨迹运动	轨迹运动	CM、CD、DT、WC
示教、记录和重现	自动数据采集和轨迹运动	CM、CD、DT、WC、RA、RD、RC
间隙补偿	双闭环	DV
基于主动轴编码器位置的轮廓跟踪运动	电子凸轮运动	EA、EM、EP、ET、EB、EG、EQ
独立定位过程中对加速度的平滑处理	运动平滑处理	IT
插补运动中的轨迹平滑处理	矢量平滑	VT
步进电机的速度平滑处理	步进电机平滑处理	KS
龙门驱动——两轴驱动同一负载	龙门同步驱动	GR、GM

独立定位

所谓独立轴定位控制，就是说各轴之间的运动独立，与其它轴的状态无关，各轴按照程序中所规定的运动轨迹完成各自的定位控制。用户为各轴指定目标绝对位置（PA）或相对位

置（PR），速度（SP），加速度（AC），减速度（DC），让其开始运动（BG），DMC-21x3 控制器就自动产生相应的速度和位置规划，完成定位控制运动。

注意：当控制器规划的运动完成时，由于电机及负载惯性，实际电机运动并不一定立即完成，但这并不影响发送下一条运动指令。

控制器可以对所有各轴同时或分别发送 BG 指令，需要用 A，B，C，D 来选择所要运动的轴。当未指定轴号时，就意味着使所有轴开始运动。在运动期间，可以随时改变运动速度（SP）和加速度（AC），不过，直到运动完成后，才可以改变减速度（DC）和位置值（PR 或 PA）。请记住，**控制器不是在实际电机处于指定目标位置时认为运动完成，而是规划中的“指令位置”到达目标位置时，也就是电机应该到达时**。在到达终点位置之前，可以随时发送停止指令(ST)使电机减速停止。

只要附加运动与正在执行的运动方向相同，在运动期间就可以指定增量位置（IP），此时用户只指定新的位置增量 n，新的目标位置就等于原有目标位置+增量 n。一旦接收到 IP 指令，就会重新规划运动轨迹，使运动达到新的终点位置。IP 指令不需要 BG。

注意：如果电机不在运动中，IP 指令就等效于 PR 和 BG 指令的组合。

独立定位指令汇总

指令	说明
PR	指定相对位置
PA	指定绝对位置
SP	指定运动速度
AC	指定加速度
DC	指定减速度
BG	开始运动
ST	停止运动
IP	指定运动增量
IT	速度平滑的时间常数
AM	等待运动规划完成
MC	等待实际位置到达

独立定位操作数汇总

指令	说明
_PRx	当前“x”轴的增量位置值
_PAx	若“X”轴不在运动中，读取当前指令位置值，而如果处于运动状态，则读取之前停止位置的指令位置
_SPX	当前“x”轴的速度设置
_ACx	当前“x”轴的加速度设置
_DCx	当前“x”轴的减速度设置
_BGx	当前“x”轴是否在运动

举例：

1）绝对位置运动

指令

PA l0000,20000　　指定 A，B 轴绝对位置

AC l000000,1000000　　指定 A，B 轴加速度

DC l000000,1000000　　指定 A，B 轴减速度

SP 50000,30000	指定 A，B 轴速度
BG AB	开始运动

2）多个运动顺序

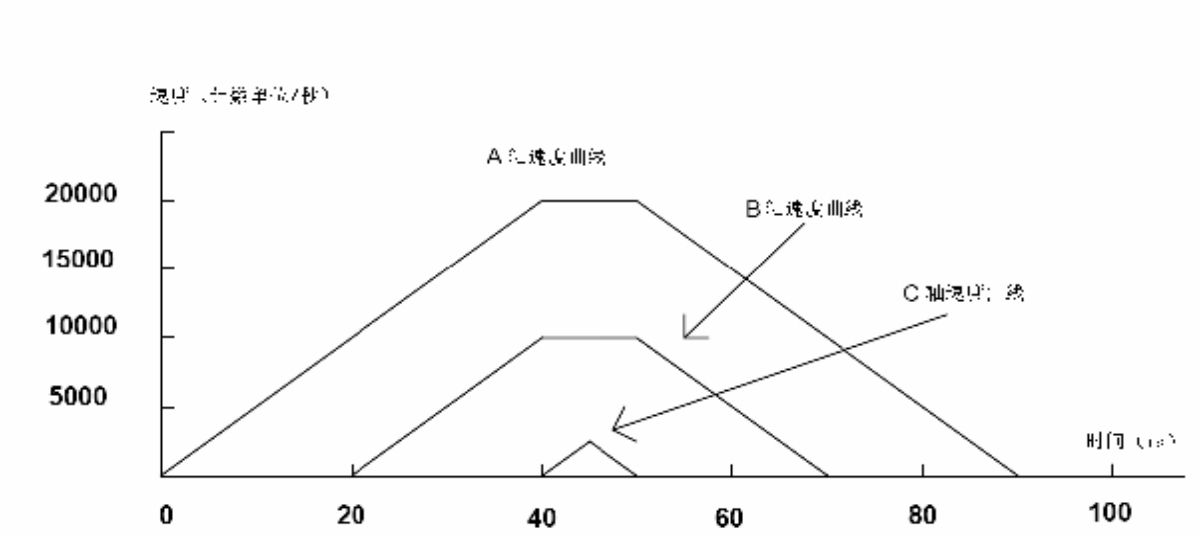
要求运动参数：

A 轴：	2000 计数单位	位置
	15000 计数单位 / sec	速度
	500000 计数单位 / sec2	加速度
B 轴：	1500 计数单位	位置
	10000 计数单位 / sec	速度
	500000 计数单位 / sec2	加速度
C 轴：	100 计数单位	位置
	5000 计数单位 / sec	速度
	500000 计数单位 / see2	加速度

此例要求 A，B，C 轴做相对位置运动，各轴运动间隔为 20msec，图 6.1 表示轴的速度轮廓。此例的程序如下：

指令	说明
#A	程序标号
PR2000,1500,100	A,B,C 轴运动距离分别为 2000，1500，100 计数单位
SPl5000,10000,5000	A,B,C 轴运动速度分别为 15000，10000，5000 计数单位/sec
AC500000,500000,500000	各轴加速度为 500000 计数单位 / sec2
DC500000,500000,500000	各轴减速度为 500000 计数单位/sec2
BGA	开始 A 轴运动
WT20	等待 20msec
BGB	开始 B 轴运动
WT20	等待 20msec
BGC	开始 C 轴运动
EN	程序结束

图 6.1 -ABC 速度轮廓



图注：A、B 两轴有一个“梯形”速度包络线，而 C 轴有一个“三角形”速度包络线。AB 轴加速度到指定速度，以恒速运动。然后减速到达指令位置所指定的终点。C 轴加速，但在到达指定速度之前，就必须减速以使该轴停在指定的位置。所有三个轴均有相同的加，减速度，因此，三个速度轮廓的上升，下降沿的斜率是相同的。

JOG 运动

JOG 运动方式非常灵活，这是因为在运动期间，可以随时变更运动速度、加速度和运动方向。用户对各轴规定 JOG 速度（JG），加速度（AC）和减速度（DC）。运动方向由 JG 参数的符号来决定：当用 BG 使运动开始时，电机加速到达所设速度，并以此速度连续运转，直到有新的速度给定或发送 ST 指令。如果在运动期间改变 JOG 速度，控制器就会使电机加速或减速达到新速度。

注意：在以 JOG 方式运行期间，控制器以位置闭环方式工作。DMC-21x3 控制器将速度轮廓转换成位置-时间包络线，每个采样周期产生新的位置目标值。这种控制方法用精确锁相进行精密速度调节。

通过使用 IP 指令能够使电机位置发生瞬时改变，一接收到此指令，控制器就指令电机到达给定增量加上当前位置的新目标位置，此指令对于在电机运行过程中，想使两个电机的位置同步是非常有用的。

注意：如果是脉冲输出的控制方式，IP 指令的参数不能过大，如果要求的瞬时速度超过控制器的极限速度（3MHz）否则会控制器停止输出脉冲。

JOG 运动指令汇总

指令	说明
AC	指定加速度
DC	指定减速度
BG	开始运动
ST	停止运动
IP	指定瞬时运动增量
IT	使加、件速过程平滑的时间常数
JG	指定 JOG 速度和方向

JOG 运动操作数汇总

指令	说明
_SPx	当前“x”轴的速度设置
_ACx	当前“x”轴的加速度设置
_DCx	当前“x”轴的减速度设置
_TVx	当前“x”轴的编码器反馈速度
_BGx	当前“x”轴是否在运动

举例：

1）只有 X 轴以 JOG 方式运动

使 A 轴以 50000 计数单位/sec 速度做 JOG 运动，在 A 轴电机到达 JOG 速度后，使 C 轴以 25000 计数单位/see 速度做反方向运动。

指令	说明
#A	程序标号
AC20000,,20000	指定 A，C 轴加速度为 20000 计数单位 / sec2
DC20000,,20000	指定 A，C 轴减速度为 20000 计数单位 / sec2
JG50000,,-25000	指定 A，C 轴 JOG 速度和方向

BGA	开始 A 轴运动
ASA	等待 A 轴速度到达
BGC	开始 C 轴运动
EN	程序结束

2）手操杆控制 JOG 运动

JOG 速度也能够用模拟输入（例如手操杆）加以改变，假设 10V 输入，对应速度必须是 50000cts/sec。

指令	说明
#JOY	程序标号
JG0	设置 JOG 方式
BGA	开始运动
#B	循环程序标号
V1=@AN[1]	读取模拟口 1 输入
Vel=V1*50000 / 10	计算速度
JG Vel	改变 JG 速度
JP#B	循环

相对位置运动，各轴运动间隔为 20msec，图 6.1 表示轴的速度轮廓。此例的程序如下：

位置跟踪

在独立轴定位控制方式下，在运动过程中可以改变运动速度、加速度但不能任意改变目标位置。而在一些应用中，电机在运动中跟踪一个随机位置，无论一个轴是否在运动，都要即刻改变最终位置。位置跟踪模式同样需要指定速度（SP）、加速度（AC）和减速度（DC）指令。启动位置跟踪模式后，随时可以指定目标位置（PA）。不需要 BG 指令来开始运动，指定位置后，控制器就会根据当前的位置、速度和目标位置来产生或修改位置规划。

启动位置跟踪所用的指令为

PT a，b，c，d。

当 a，b，c，d 为 1 时，启动对应轴的位置跟踪模式；当 a，b，c，d 为 0 时，关闭对应轴的位置跟踪模式。

在位置跟踪模式下，AM、MC 指令无效。

位置跟踪指令汇总

指令	说明
PT	开启/关闭位置跟踪功能
PA	指定目标位置
SP	指定运动速度
AC	指定加速度
DC	指定减速度

位置跟踪操作数汇总

指令	说明
_PTx	当前“x”轴位置跟踪模式的状态
_PAx	若“X”轴不在运动中，读取当前指令位置值，而如果处于运动状态，则读取之前停止位置的指令位置
_SPX	当前“x”轴的速度设置

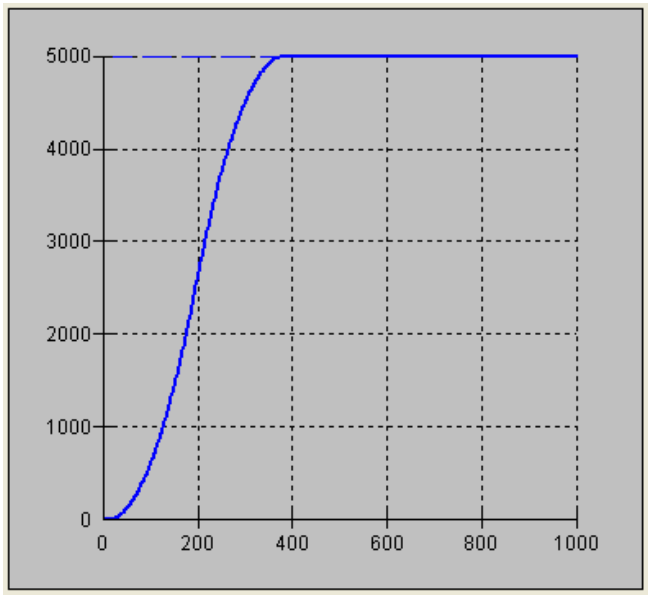
_ACx	当前“x”轴的加速度设置
_DCx	当前“x”轴的减速度设置

举例：

指令

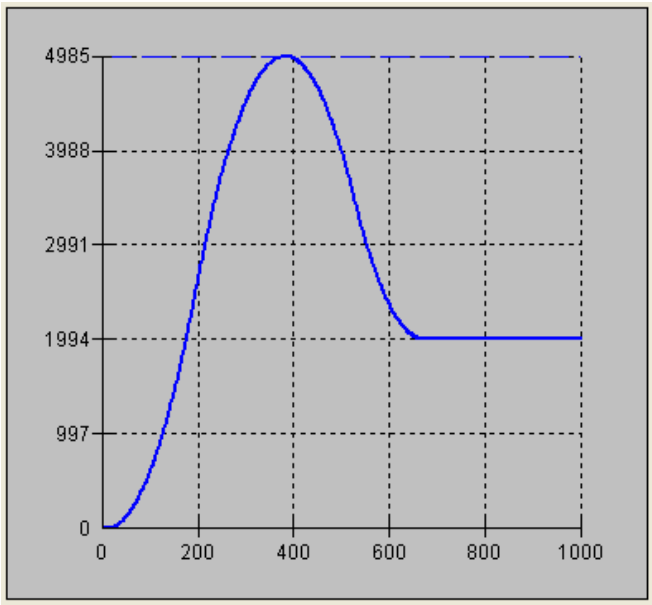
PT 1	使用位置跟踪
AC l50000	指定 A，B 轴加速度
DC l50000	指定 A，B 轴减速度
SP 50000	指定 A，B 轴速度
PA 5000	

上面程序的运动过程如下图：

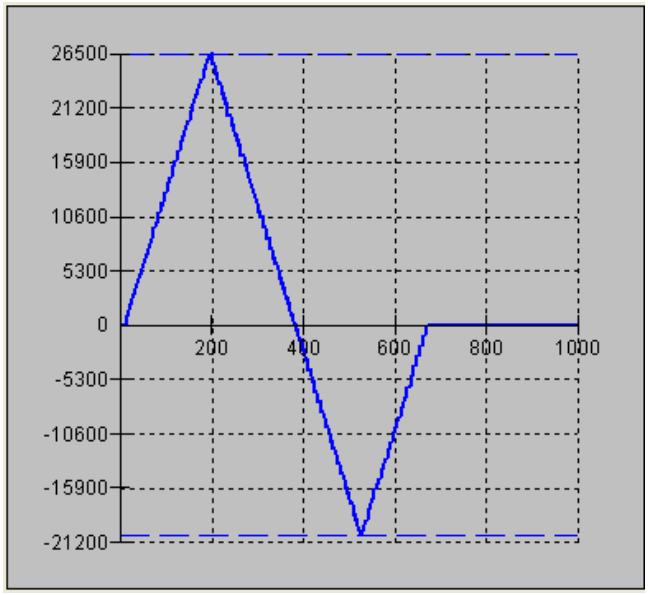


a.位置-时间（ms）曲线

如果在到达 5000 之前，发送 PA 2000 指令，则可以得到如下运动过程：

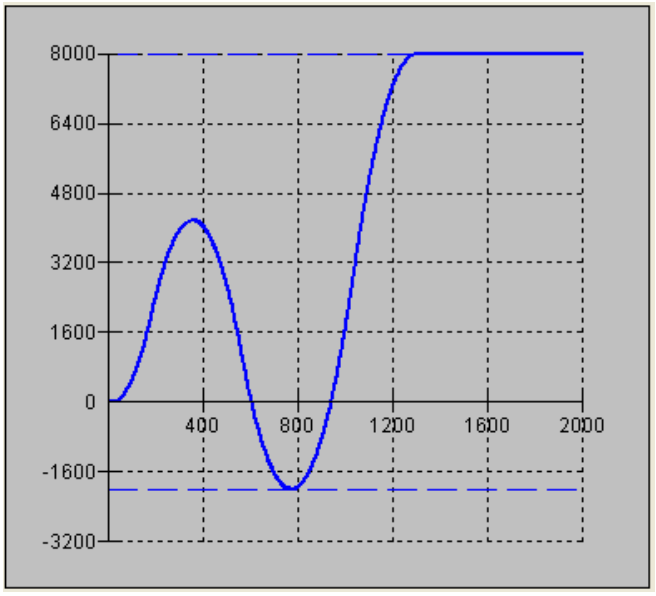


b.位置-时间（ms）曲线

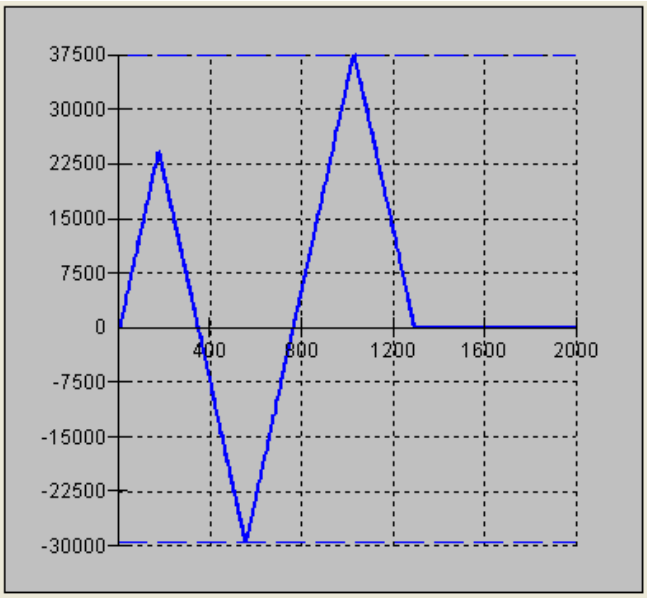


c.速度-时间（ms）曲线

如果在运动中，先将目标位置改为-2000，而后再改为 8000，则运动过程下：



d.位置-时间（ms）曲线



e.速度-时间（ms）曲线

多维直线插补

DMC-21x3 提供最多 8 轴直线插补功能。在直线插补方式中，各轴沿规定的轨迹以矢量速度，加速度，减速度联动；运动轨迹根据各轴的增量距离来产生。在连续插补运动中，可以给定无限增量线段，使直线插补方式完全跟随工件运动轨迹，对总的运动长度没有限制。

LM 指令为插补选择直线插补方式和插补轴。例如，LMBC 只选择 B 轴和 c 轴为直线插补。使用直线插补方式时，如果不补改变直线插补轴，那么只需要指定一次 LM 指令。

指定联动矢量

对于直线插补方式或矢量方式来讲，DMC-21x3 考虑到两套联动矢量，他们分别用字母 S 和 T 来分辨。

要想规定矢量指令，必须先辨别联动矢量。用 CAS 指令来辨别 S 矢量，而用 CAT 指令来辨别 T 矢量。如果未用 CA 指令改变坐标系，那么所有矢量指令都适用于当前有效的矢量。

指定直线段

用指令 LI a,b,c,d,e,f,g,h 来为各轴指定增量运动距离，BGS 指令之前，可以给出 511 个增量线段。一旦开始运动，还可以给控制器发送附加 LI 线段。

在运动启动之前，能用清除线段指令 CS 来删除存储在缓冲区中的 LI 线段。要想停止运动，就使用指令 STS 或 AB。ST 指令使电机减速停止，而指令 AB 使电机即刻停止并使控制器上的用户程序终止执行，（指令 AB1 只使运动终止，不会中断用户程序）。

必须使用直线插补结束指令 LE 来指定直线插补运动社定程序结束，此指令告诉控制器减速停止在最后一条 LI 指令之后。如果未给定 LE 指令，就必须用终止指令 AB1 来终止运动。

用户的责任在于保证 DMC-21x3 缓冲区中保持足够的 LI 线段以确保连续运动。如果控制器未接收到附加 LI 线段和 LE 指令，在最后一个矢量线段完成之后，控制器就会立即停止运动，也没有可控制的减速度。LM?或_LM 读取能够发送到缓冲区的 LI 线段的可用空间。返回值为 511 意味着缓冲区是空的，可以发送 511 个 LI 线段。若返回值为 0，则意味着缓冲区是满的，不能再发送附加 LI 线段。只要缓冲区未满，就能够以 PC 机总线速度发送附加 LI 线段。

指令_CS 读取线段计数器值，当处理线段时，_CS 增加，此计数器值以 0 开始计数。此功能使得主计算机能够确定正在处理哪条线段。

附加指令

用指令 VSn,VAn 和 VDn 来指定矢量速度，矢量加速度和减速度，DMC-21x3 以 LM 方式中所指定的轴为基础计算矢量速度。例如，LMABC 指定 A,B,C 轴为直线插补轴，此例中的矢量速度用下式进行计算：

$VS^2=AS^2+BS^2+CS^2$

其中：AS,BS,CS 分别为 A,B,C 轴的速度。

控制器总是使用由 LM 所指定的轴定义来计算速度，而不是 LI。

用 VT 来设置联动控制轴 S 曲线平滑参数；指令 AVn 是“矢量运动完成后”条件启动指令，它使程序暂停执行，直到矢量距离 n 到达之后，才重新启动执行后续程序。

对各插补线段指定矢量速度

指令 VS 具有立即效应，因此，必须在需要时刻给定。在某些应用中，例如 CNC，需要对不同的运动线段加附各种不同的速度；这可以用两个参数来进行（<n 和>m）。

例如：LIa,b,c,d<n>m

第一个指令<n 等效于在给定线段的起点指令 VSn，在其它约束下，使运动朝着新的速度变化；第二个函数>m，需要矢量速度在线段终点要达到的值 m。注意：函数>m 可在给定的线段内或在前一线段期间开始减速，在给定的 VA,VD 值条件下，满足最终速度需要。

不过，请注意：控制器一次只有一个>m 指令起作用，因此，一个参数可能会被另一个屏蔽。例如，如果函数>100000 由>5000 跟随其后，且减速距离不够，那么就不会满足第二个条件；控制器试图将速度降到 5000，但会在指定位置之后。

作为一个例子，看看以下程序。

指令	说明
#ALT	程序标号
DP ,0,0	定义 B，C 轴位置为 0
LM BC	定义 B，C 轴之间为直线插补方式
LI ,4000,0<4000>1000	指定第一个直线插补线段 B 轴走 4000 单位 C 轴走 0 单位，矢量速度为 4000，结束速度为 1000
LI ,1000,1000<4000>1000	指定第二个直线插补线段 B 轴走 1000 单位 C 轴走 1000 单位，矢量速度为 4000，结束速度为 1000
LI ,0,5000<4000>1000	指定第三个直线插补线段 B 轴走 0 单位 C 走 5000 单位，矢量速度为 4000，结束速度为 1000
LE	结束直线插补线段
BGS	开始运动
EN	程序结束

改变进给率

指令 VRn 使进给率 VS 以 0.0001 的分辨率在 0~10 之间成比例变化。此指令立即起作用并使 VS 按比例变化。VR 也适用于用“<”指定矢量速度时。此指令对于进给率修调功能非常有用。VR 不对加速度进行比例处理。例如，VR0.5 使 VS2000 减小一半。

直线插补指令汇总

指令	说明
LM	指定直线插补轴
LI	指定相对于当前位置的增量距离，并分配矢量速度

VS	指定矢量速度
VA	指定矢量加速度
VD	指定矢量减速度
VR	指定矢量速度比率
BG	开始直线插补程序
CS	清除缓冲区中程序段，
LE	直线插补结束——位于 LI 指令的最后
AM	在矢量运动完成之后的条件启动
AV	在相对矢量距离到达之后的条件启动
VT	矢量运动 S 曲线平滑常数

直线插补操作数汇总

指令	说明
_AV	读取已运动的相对矢量距离
_CS	插补线段计数器——读取程序中的插补线段数，以 0 开始
_LE	读取矢量长度(在 2147483647 之后复位)
_LM	读取控制器缓冲区中直线插补线段可用空间，0 代表缓冲区已满，511 表示缓冲区空
_VPx	读取沿轨迹最后一个数据点的绝对坐标值，（x：A，B，C，D，E，F，G，H）

为了表明查询运动状态的功能，考虑一下举例#LMOVE 中的第一个运动线段，其中，B 轴朝点 5000 运动。假设，当 B=3000 时，用指令“MG_AV"查询控制器，返回的值将是 3000，_CS，_VPB，_VPC 的值将为 0。现在假设：在第二个插补线段 C=2000 时重复查询，在此点的_AV 值是 7000，_CS 等于 1，_VPB=5000，_VPC=0。

举例：

1）直线插补运动

在本例中，要求 A，B 轴进行 90 度转向，为了在拐角附近减慢速度，我们使用 AV4000 条件启动指令，将速度减慢到 1000cts/sec,一旦电机到走过拐角，速度就增回到 4000cts/sec。

指令	说明
#LMOVE	程序标号
DP0,0	定义 AB 轴位置为 0
LMAB	定义 AB 轴为直线插补关系
LI5000,0	指定第一条直线插补线段
LI0,5000	指定第二条直线插补线段
LE	直线插补结束
VS4000	指定矢量速度
BGS	开始运动
AV4000	设置条件启动，等待到达矢量距离 4000
VSl000	改变矢量速度
AV5000	设置条件启动，等待到达矢量距离 5000
VS4000	改变矢量速度
EN	程序结束

2）直线运动

在 CD 平面（即 ZW 平面）做联动控制直线插补运动，以 100000cts/sec 矢量速度和

100000cts/sec^2 的矢量加速度移动到坐标点 40000，30000cts。

指令	说明
LMCD	指定直线插补轴
LI ,,40000,30000	指定 C,D 距离
LE	直线插补结束
VSl00000	指定矢量速度
VAl000000	指定矢量加速度
VDl000000	指定矢量减速度
BGS	开始运动

注意：上述程序指定的矢量速度 VS,并不是实际轴速度 VC 和 VD，轴速度由下式求得：

$$VS = \sqrt{VC^2 + VD^2}$$

各轮廓曲线见图 6.2

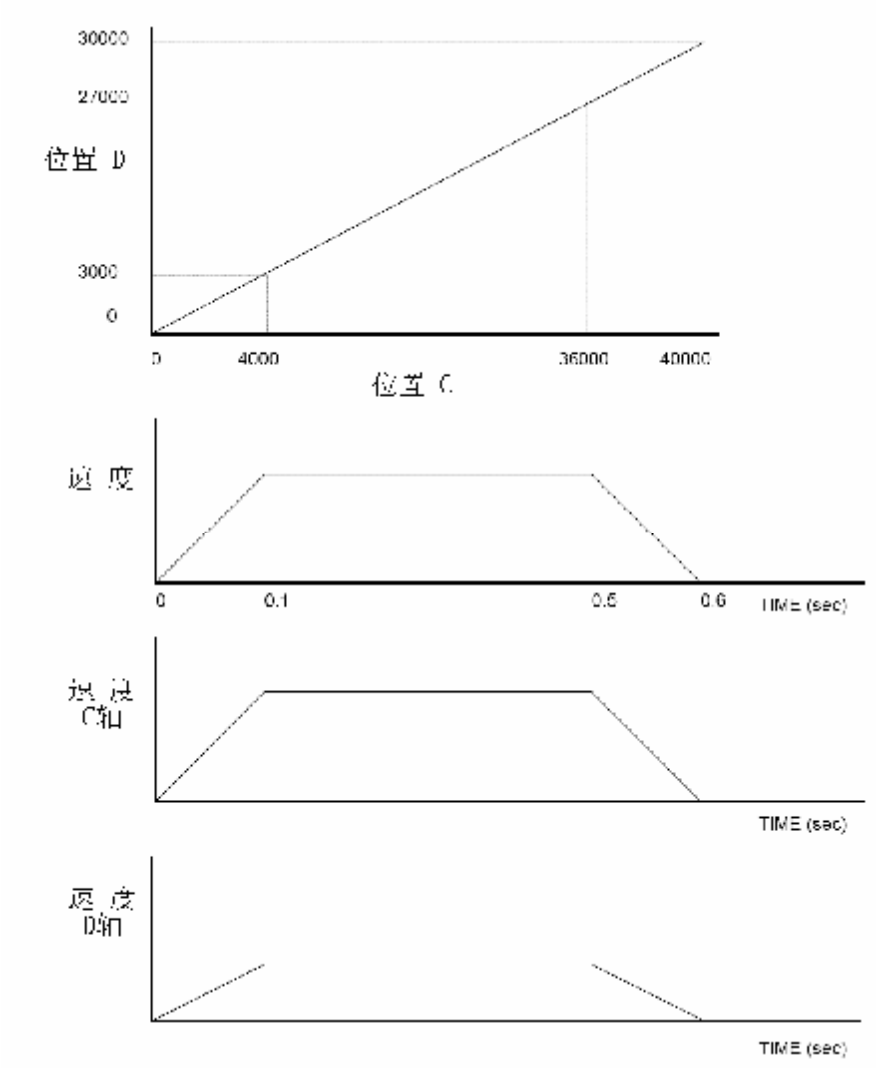


图 6.2-直线插补

3）复杂运动

本例是 AB 平面（XY 平面）的直线插补运动，使用数组 VA 和 VB 来存储由程序#LOAD 所填入的 750 个增量距离。

指令	说明
#LOAD	LOAD 程序
DAVA[750],VB[750]	定义数组

Count=0	初始化计数器
n=0	初始化位置增量
#LOOP	循环标号
VA[count]=n	填数组 VA
VB[count]=n	填数组 VB
n=n+10	增量位置
count=count+l	计数器+1
JP#LOOP,count<750	如果数组未满，则循环
#A	标号
LMAB	指定 AB 轴为直线插补方式
count=0	初始化数组计数器
#LOOP2;JP#LOOP2,_LM=0	若程序缓冲区满，则等待
JS#C,count=500	在第 500 个插补线段处开始运动
LIVA[count],VB[count]	指定直线插补线段
count=count+l	数组计数器+1
JP#LOOP2,count<750	重复循环，直到数组执行完
LE	直线插补结束
AMS	插补程序执行完后
MG"DONE"	发送信息
EN	结束程序
#C;BG;EN	开始执行运动子程序

平面直线和圆弧插补方式

DMC-21x3 控制器提供了平面内圆弧插补功能，直线插补和圆弧插补可以混合编程。即使在直线和圆弧之间的过渡处，沿轨迹的运动也是以矢量插补速度连续进行。DMC-21x3 执行所有直线、圆弧插补复杂运算，上位 PC 机可脱机去处理其他任务。而对于 3 维以上的圆弧插补，可以用高级语言（如：C / C++.、VC 等），并调用 API 函数，在开放式平台上进行第二次开发，以满足这方面的应用要求。

二维平面运动方式类似于多维直线插补方式。可以选择任意两个轴为联动轴构成直线和圆弧插补运动。此外，也能对第三轴进行控制，使其与选择的两轴运动保持角度跟随关系。注意：在任何时候，只能为联动轴指定任意两个轴构成两维圆弧插补关系。

指令 VM m,n,p 其中 m 和 n 是两个联动轴，p 是角度跟随轴，如果不需要角度跟随功能，可以只使用 VM m,n。

注意：指令中的逗号（,）不是必须的，可以省略。例如，VMABC 选择 AB 轴为联动轴，而 C 轴为角度跟随轴。

指定插补坐标平面

DMC-21x3 为直线插补和圆弧插补方式提供了两组不同的坐标系。他们分别用字母 S 和 T 加以区别。

要指定矢量指令，必须先区分坐标平面。用指令 CAS 来确认 S 平面，用指令 CAT 来确认 T 平面。所有矢量指令都适用于当前有效的坐标系，直到用 CA 指令加以改变。

指定矢量线段

运动线段用两个指令进行描述，它们分别是：VP 为直线插补线段；CR 为圆弧插补线段。

一旦规定了一组直线插补线段和（或）圆弧插补线段，就要用指令 VE 来结束。这就意味着这些指令所构成的程序为插补运动。在执行第一个插补运动之前，控制器先将插补程序中所有运动轴的当前矢量位置定义为 0。

注意：局部 0 定义不影响绝对坐标系或以后的插补运动程序。

VP x,y 指令规定矢量运动终点相对于起点的坐标值。指令 CR t,q,d 以半径 r，起始角 q 和弧角 d 来定义圆弧的弧长。起始角 q 约定为：0，对应于轴正方向，而对于 q 和 d，逆时针方向旋转为正向。

在一个程序中，在开始运动前可指定 511 个 CR 或 VP 插补线段，用 BGS 来启动矢量运动程序，一旦开始运动，就可以继续加入附加插补线段，必须用指令 VE 来结束

在运动开始之前能用 CS 指令来删除存储在缓冲区中的 VP 和 CR 指令。用指令 STS 或 ABl 来停止运动。STS 指令以规定的减速度使运动停止；ABl 立即使运动中止。

必须用矢量结束指令来指定插补运动结束，此指令需要控制器减速停止在最后一条插补运动之后。若未给定 VE 指令，就必须用 ABl 来中止插补控制程序执行。

用户有必要在 DMC-21x3 程序缓冲区中存入足够的运动程序段，以确保连续运动。如果控制器未接收到附加运动程序段，且没有 VE 指令，控制器就会立即使运动停止在最后的矢量处，也不会有可控制的减速度过程。LM?或_LM 读取能够发送到缓冲区中的运动线段的可用空间。返回值为 511 表示缓冲区是空的，此时可发送 511 个插补线段；0 表示缓冲区是满的，此时不能发送附加插补线段。只要缓冲区未满，就能以 PC 机总线速度发送附加插补线段。

用操作数_CS 来确认插补线段计数器的值。

附加指令

指令 VSn，VAn 和 VDn 用于设置矢量速度，加速度和减速度。

指令 VT 是用于联动运动的 S 曲线平滑参数。

对各插补线段指定失量速度

可以用即时指令 VS 指定矢量速度。也能将它用如下指令附加到插补线段中：

VPa,b<n>m

CRr,θ,δ<n>m

第一个参数<n，等效于在给定插补线段起点加入指令 VSn，若无其他限制，它会使运动向新指令的速度加速。

第二个参数>m 规定了到达插补线段终点处的矢量速度值。

注意：参数>m 可以使运动在给定的插补线段内或在前一插补线段期间开始减速，在给定 VA 和 VD 值条件下，满足最终的速度要求。

但是，请注意，控制器每次只能以一个>m 指令工作，因此，一个参数可以用另一个进行屏蔽；例如，如果参数．>100000 由>5000 跟随其后，且减速距离不够，那么就不满足第二个条件，此时，控制器就设法将速度减到 5000，但不会在指定位置达到。

改变进给率（进给率修调）

指令 VRn 使进给率 VS 以 0.0001 分辨率在 0~10 之间进行比例修调。此指令立刻起作用并使 VS 成比例修正。VR 也适用于用“<”操作数指定矢量速度的场合。此功能对于像 CNC 那样要求有进给率修调功能的应用非常有用。VR 不对加，减速度做比例处理。例如，VR 0.5 使 VS2000 除以 2。

以编码器分辨率补偿偏差

当使用矢量方式时，DMC-21x3 对编码器分辨率采用缺省比例系数 1：1，如果实际情况与此不同，就用指令 ES 对编码器计数进行比例处理；ES 指令附带两个参数，分别代表用于矢量运动的两个编码器的计数值。两个数的比例越小就会由更高的分辨率编码器来倍乘。通常称此为椭圆缩放系数，此指令对于椭圆加工非常有用。详见《指令手册》ES 指令。

条件启动

AVn 指令是在矢量运动完成后的条件启动，在执行程序中的下一条指令之前，等待相对矢量距离 n 到达。

角度跟随

像玻璃切割，石材切割类应用，要求第三轴(即刀片)的角度与切割平面运动轨迹保持正切，为了满足这类应用，DMC-21x3 控制器使一个轴做为角度跟随轴。在角度跟随工作之前，需要用 VM 指令来分配一个轴，并通过 TN m，n 指令定义该轴的偏置和比例系数：m 为比例系数，单位为计数单位/度，n 为联动坐标平面中等于 0 度的正切位置。用操作数_TN 来返回正切轴的初始位置。

平面插补控制指令汇总

指令	说明
VM	指定二维平面联动轴
VP	指定矢量终点坐标值
CR	指定圆弧的形状
VS	指定矢量速度
VA	指定矢量加速度
VD	指定矢量减速度
VR	指定矢量速度比率
BG	开始运动
CS	清除缓冲区中程序段，
VE	联动结束
AM	在矢量运动完成之后的条件启动
AV	在相对矢量距离到达之后的条件启动
VT	矢量运动 S 曲线平滑常数
TN	正切角度的比例和偏置
ES	两轴定标比例系数

平面插补操作数汇总

指令	说明
_AV	读取已运动的矢量距离
_CS	插补线段计数器——读取程序中的插补线段数，以 0 开始
_VE	读取矢量长度(在 2147483647 之后复位)
_LM	读取控制器缓冲区中直线插补线段可用空间，0 代表缓冲区已满，512 表示缓冲区空
_VPx	读取沿轨迹最后一个数据点的绝对坐标值，（x：A，B，C，D，E，F，G，H）

举例

1）角度跟随

假设一个 XY 工作台，带有控制切刀的 C 轴，C 轴编码器为 2000ets/rev，在上电之后它对 C 轴初始化，使切刀指向+B 方向，要进行 180° 圆弧切割，半径为 3000，中心点在坐标原点，起始点在（3000，0），以逆时针方向运动，终点在（-3000，0）。注意：XY 平面内的 0° 方向是在+A 轴方向，对应于 z 轴位置-500，定义偏置。此运动分两部分，首先，将 A，B，C 轴驱动到起点，然后，使切刀运动。假使切刀用输出位 0 来使能。

指令　　说明

#EXAMPLE	程序标号
VM ABC	具有 C 轴做角度跟随的 AB 平面运动
TN2000/360,-500	C 轴，2000cts/度，以位置-500 为 0°。
CR3000,0,180	半径为 3000，起始角为 0°，逆时针转 180°
VE	矢量结束
CB0	切刀使能断开
PA3000,0,_TN	把 AB 轴移到起点位置，C 轴提到初始正切位置。
BGABC	开始运动
AMABC	当运动完成时，
SB0	使切刀使能
WT50	使切刀使能等待 50msec
BGS	开始圆弧切割
AMS	当联动轴运动完成时
CB 0	使切刀使能关断 0
MG “ALL DONE”	
EN	程序结束

2）平面运动

运动路径如图 6.3 所示，进给率为 20000cts / sec，运动平面为 AB。

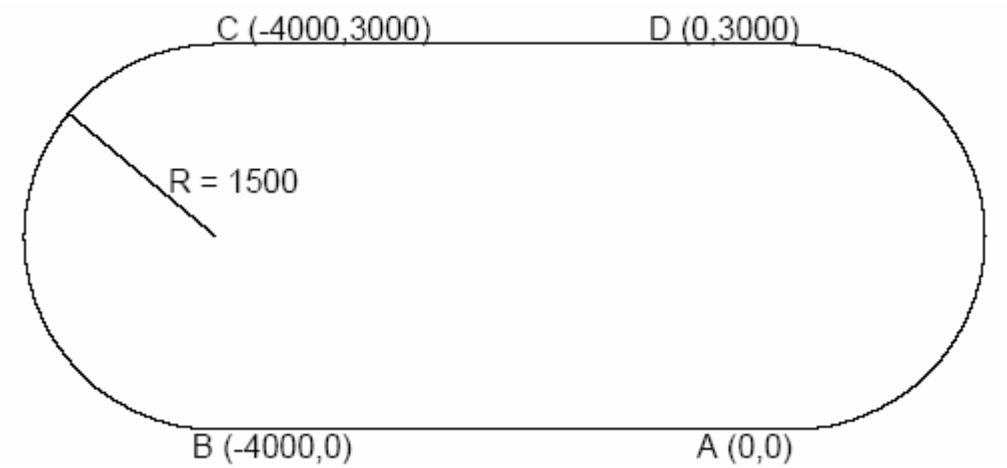


图 6.3-运动路径

指令	说明
VM AB	指定运动平面
VS20000	指定矢量速度
VA1000000	指定矢量加速度
VD1000000	指定矢量减速度
VP-4000,0	线段 AB
CRl500,270,-180	线段 BC
VP0,3000	线段 CD
CRl500,90,-180	线段 DA
VE	矢量程序结束
BGS	开始运动

此运动以 A 点开始，向点 B，C，D，A 运动，假设，运动到中途 AB 两点之间时，查询控制器。

_AV 的值是 2000

_CS 的值是 0

_VPA 和_VPB 是 A 点的绝对坐标值

假设在 C、D 两点中间再次查询：

_AV 的值是 4000+1500π+2000=10,712

_CS 的值是 2

_VPA 和_VPB 为 C 点的坐标值。

电子齿轮同步功能

此方式允许最多 8 个轴从电气上与另外的主动轴实现同步运动。主动轴可以两个方向旋转，而其它电子齿轮轴则以规定的齿轮比跟随运动。各轴的齿轮比可以不同，在运动期间可以改变。

指令 GA ABCDEFGH 指定主动轴，GRa,b,c,d 指定从动轴的齿轮比，其中，齿轮比可以是±127．9999 之间的数，分辨率是 0.0001。电子齿轮有两种方式：分别是标准电子齿轮同步和龙门同步方式。用指令 GM 使龙门同步方式有效。GR0,0,0,0 关掉所有方式的电子齿轮同步控制。在标准方式中，电子齿轮的从动轴的限位开关或 ST 指令会使电子齿轮同步功能无效，但在龙门同步方式中，从动轴的限位开关无效，ST 指令无效。

指令 GMa,b,c,d 选择在龙门同步方式下要控制的轴。参数为 1 使龙门同步方式使能，而 0 则为标准电子齿轮功能。

GR 使指定的轴与主动轴的实际位置成电子齿轮耦合关系，主动轴用运动指令 PR，PA 或 JG 等来控制，或者由外部来控制主动轴。

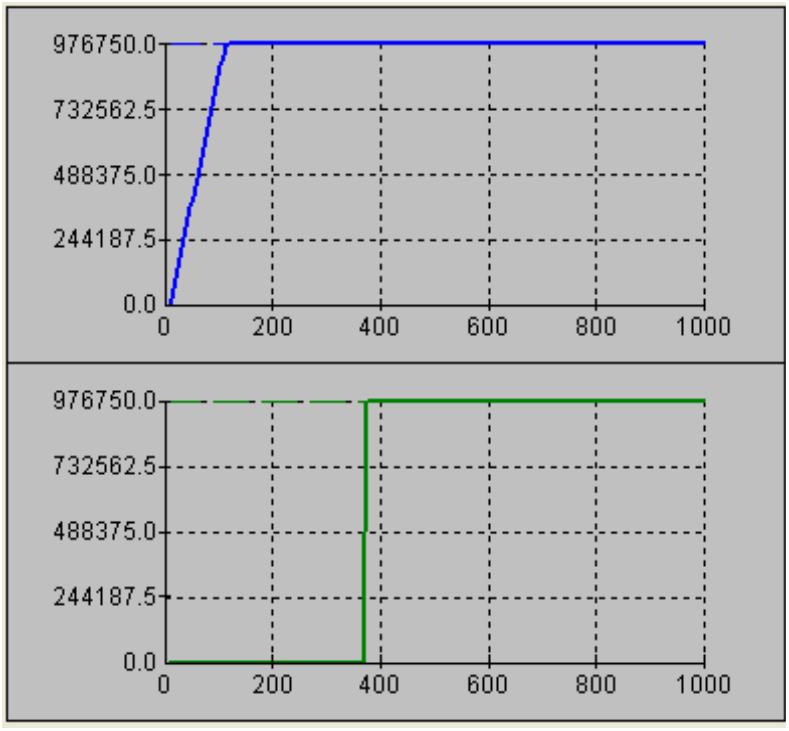
主动轴也是由控制器驱动时，还可以将主动轴定义为该轴的指令位置或是实际反馈实际位置。用字母 C 来表明以主动轴的指令位置作为同步对象。例如，GAB 表示电子齿轮轴是以 B 轴的实际反馈位置来同步，A 轴会按照 B 轴的编码器反馈情况运动；GACB 则表示电子齿轮轴是以 B 轴的指令位置来同步，A 轴会按照 B 轴的指令位置运动，也就是说，当向 B 轴下达了运动指令，即使由于某种原因，B 轴的编码器反馈信号与给等的命令不一致，A 轴也会按照指令来同步运动。

另一种电子齿轮方法，是使从动轴电机与 GAS 所规定的几个轴的矢量指令位置相同步。例如，如果 A 轴和 B 轴形成圆弧运动，C 轴可以与矢量运动成比例运动。同样，如果 A、B、C 轴执行直线插补运动，就能使 D 轴与矢量运动成电子齿轮运动关系。

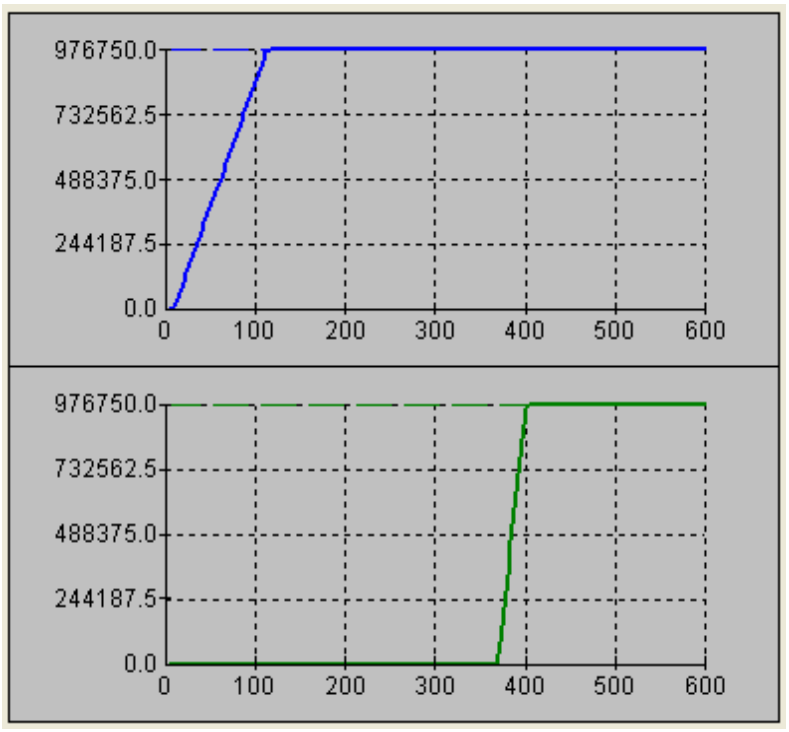
电子齿轮功能也允许电子齿轮从轴电机执行除电子齿轮之外的独立轴运动或联动运动。则该轴的运动速度、距离为两种运动的叠加。例如，当电子齿轮轴电机以 1：1 的比率跟随主动轴时，也可以用 PR、JG 指令或 VP、LI 指令使其超前一个附加的距离。

在一些应用中，需要在主轴高速运动中开/关电子齿轮功能，或者改变同步比例。例如，主轴在以 1,000,000cts/sec 的速度运行，从动轴实现 1：1 同步。在开始同步的瞬间，从动轴的速度要瞬间达到 1,000,000cts/sec，电机要以最大加速度运行，这会给系统带来很大的震动。在大多数应用中，相对于这种震动，从动轴在一定的时间内慢慢加速到达同步是可以接受的。GALIL 允许用户指定一段主轴的运动距离作为实现同步的过程。在例如，主轴同样是以 1,000,000cts/sec 的速度运行，同步比为 1：1，但是，实现同步的距离为主轴的 30,000cts，这就可以很大的消除从动轴的震动。下面两图是立即同步和逐渐同步过程的速度-时间曲线。

每个图中，下面的部分是从动轴，上面的部分是主动轴，在后一个图中，从动轴的震动明显减轻。在加速过程完成前，两个轴并没有真正的完全同步，而是从动轴明显落后。如果 从开始的一瞬间就需要精确的位置同步，则需要用在同步以外的指令对位置补偿。控制器可以用操作数_GP 跟踪这个相位延迟。



i.立即同步的速度-时间曲线



ii.逐渐同步的速度-时间曲线

用一个例子来说明_GPn 的作用。两个轴分别以 1.132 和-0.045 的比例与外部主轴同步，由于主轴在高速运动中，需要逐渐同步。使用 DMC-2143 控制器，C 轴作为外部主轴，A、B 两轴作为从动同步轴，实现同步的距离为 60 主轴的 6000cts。

MOC	关闭 C 轴，将其作为外部主轴
GAC,C	C 轴的主编码器输入作为 A、B 的主轴
GD6000,6000	指定逐渐同步距离为主轴的 6000cts。
GR1.132,-0.045	指定同步比

在从动轴逐渐同步的过程中，主轴运动了 6000cts，从动轴分别只移动了大约 3396cts、

135cts，相差的部分存储在操作数_Opn 中。如果需要精确的位置同步，用 IP 指令调整差值。

电子齿轮同步功能指令汇总

指令	说明
GA	为电子齿轮定义主动轴，参数： n=A,B,C,D,E,F,G,H 以主编码器反馈为主动轴 n=CA,CB,CC,CD,CE,CF,CG,CH 以指令位置为为主动轴 n=DA,DB,DC,DD,DE,DF,DG,DH 以辅编码器反馈为主动轴 n=S 或 T 为联动运动
GD	速度同步前的加速距离
_GPn	改变同步比时，从动轴逐渐加速带来位置滞后量。
GR	为从动轴设置齿轮比，0 使所指定轴的电子齿轮无效
GM	1 设置为龙门同步方式，0 使龙门同步无效
MR	经过指定值时，使运动反向的条件启动，只能使用 1 个域
MF	经过指定值时，使运动正向的条件启动，只能使用 1 个域

举例

1）简单主从运动

主动轴以 100000cts/sec 速度移动 10000cts，定义 B 轴为主动轴，A、C、D 轴与主动轴成电子齿轮运动，齿轮比分别为 5，-0.5，10。

指令	说明
GAB,,B,B	指定主动轴为 B 轴
GR5,,-0.5,10	设置齿轮比
PR,10000	指定 B 轴位移量
SP,100000	指定 B 轴速度
BGB	开始运动

2）电子齿轮

目的：以外部主动轴速度的 1.132 倍和-0.045 倍运行两个电子齿轮电机，主动轴以 0~1800r/min（2000cts/rev 编码器）速度驱动主动轴。

方案：使用 DMC-21x3 控制器，其中 C 轴为主动轴，A、B 轴为电子齿轮轴。

指令	说明
MOC	关断 C 轴使能，C 轴由外部控制，作为主动轴使用
GAC,C	指定 C 轴为 A、B 轴的主动轴
GRl.132,-.045	指定齿轮比

现在假设：A 轴的齿轮比要随时改变为 2，用以下指令来实现：

GR2	指定 A 轴齿轮比为 2

3）龙门同步方式

在主动轴和随动轴由 DMC-21x3 控制器的应用中，使随动轴与主动轴的指令位置同步，而不是实际位置，这样可以忽略主动轴的震荡。

例如，由 A、B 两轴在两侧驱动龙门架，对于两台电机之间的刚性连接而言，就需要龙门同步方式。A 轴是主动轴，B 轴是随动轴，要使 B 轴与 A 轴的指令位置同步，使用指令：

指令	说明
GACA	指定 A 轴指令位置为 B 轴的主动轴
GR 1	设定 B 轴的齿轮比为 1：1
GM 1	设置龙门同步方式
PR 3000	指定 A 轴位置

BGA　　启动 A 轴运动

在电子齿轮方式中，您也可以进行轮廓位置修正。例如，假设，您需要使从动轴超前 10cts，就简单地指令：

IP，10　　指定 B 轴增量位置位移量 10

在这些条件下，此 IP 指令等效于：

PR,10　　指定 B 轴相对位移量 10

BGB　　开始 B 轴运动

4）使具有轮廓速度修正功能的两个输送带同步

指令	说明
GA,A	定义 A 轴为 B 轴的主动轴
GR,2	设置 B 轴齿轮比为 2：l
PR,300	指定修正距离
SP,5000	指定修正速度
AC,100000	指定修正加速度
DC,100000	指定修正减速度
BGB	开始修正

电子凸轮

电子凸轮是一种使几个运动轴能够周期性同步的运动控制方式。7 个轴可以作为一个主动轴的从动轴。主动轴的位置必须是通过主编码器输入的实际位置。

电子凸轮是更为典型的电子齿轮应用，它以轴之间的凸轮关系表为基础，使所有控制轴实现同步。例如，DMC-2182/DMC-2183 控制器就可以有 1 个主动轴和 7 个从动轴。为了说明创建凸轮表的过程，我们假设 A 轴为主动轴，B 轴为从动轴的凸轮关系。其图形关系见图 6.4。

第 1 步．选择主动轴

电子凸轮方式的第一步是选择主动轴，用如下指令来进行：

EAp

其中：p=A、B、C、D　是选择的主动轴

为了举例方便，假设主动轴为 A 轴，因此，我们规定 EAA。

第 2 步．规定主动轴循环周期和从动轴变化

在电子凸轮方式中，主动轴的位置表达模式为 1 个循环周期。在本例中，A 轴位置总是表达为 0~6000 之间的数值。同样，也对从动轴重新定义，以便它在 0 处开始启动，在 1500 处结束。

在循环终点，当主动轴为 6000，从动轴为 1500 时，就将 A 和 B 的位置重新定义为 0，为了指定主动轴循环周期和从动轴循环周期变化，我们使用指令 EM。

EMa，b，c，d

其中：a，b，c，d 指定主动轴的循环周期和一个循环周期后从动轴的总变化量。

主动轴的循环周期限定在 8388607 以内，而每个循环周期从动轴变化量限定在 2147483647。如果变化量是负数，就取其绝对值。对于给出的例子，主动轴的循环周期为 6000cts，而从动轴的变化量为 1500。因此，我们使用指令：

EM6000,1500

第 3 步．指定主动轴间隔和起始点

下一步，我们要创建电子凸轮表，以主动轴位置的均匀间隔规定凸轮表。最多可以有 256 个间隔。用如下指令来规定主动轴间隔大小和起始点：

EPm,n

其中：m 是间隔宽度，单位为 cts，n 是起始点。

对于给定的例子，我们通过规定主动轴间隔点 0，2000，4000，6000 来构建凸轮表。因此，规定如下：

EP2000,0

第 4 步．指定从动轴位置

下一步，我们用如下指定规定从动轴位置：

ET[n]=a，b，c，d

其中：n 表示各点次序。

n 值以 0 开始，最多到 256，参数 a，b， c，d 表示相应点从动轴位置。对于此例而言，表达如下：

ET[0]=,0

ET[1]=,3000

ET[2]=,2250

ET[3]=,1500

这样就创建 ECAM 表。

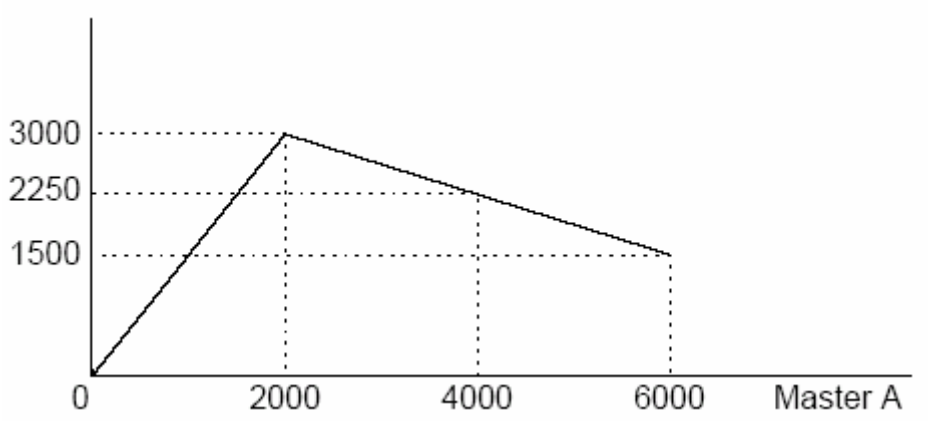


图 6.4-电子凸轮举例

第 5 步．使 ECAM 使能

为了使 ECAM 方式使能，使用指令

EBn

其中：n=1 使 ECAM 方式使能，n=0 使 ECAM 方式无效。

第 6 步．使从动轴运动啮合

要使从动轴啮合，使用指令

EGa,b,c,d

其中：a，b，c，d 是必须使相对应的从动轴啮合处的主动轴位置。

如果任意参数值在一个循环周期范围之外，凸轮就立即啮合。当凸轮啮合时，重新定义从动轴位置，模为 1 个循环周期。

第 7 步．使从动轴运动脱开

要使凸轮脱开，使用如下指令：

EQ a,b,c,d

其中：a，b，c，d 是脱开相对应从动轴处的主动轴位置。

这就使从动轴在指定的主动轴位置点脱开。如果参数在主动轴循环周期以外，就可瞬时停止。

第 8 步．创建产生 ECAM 表的程序

为了表明整个过程，用下式来表示凸轮关系：

B=0.5*A 十 100sin(0．18*A)

其中：A 是主动轴，循环周期为 2000cts

凸轮表能手动一点一点创建，或用程序自动创建。下面的程序包含配置，指定 EAA 将 A 轴定义为主动轴。主动轴的循环周期是 2000。循环一到，A 轴就变化 1000。因此，指令为：

EM2000,1000.

假设，我们用 100 个线段定义一个表，因此，增量间隔就为 20cts。

如果主动轴在 0 点开始起动，那么，所需要的指令就是：EP20,0.

下面的程序计算表点，由于相位角等于 0.18A，且 A 以 20 增量变化，因此，相位变化增量为 3.6°，

然后，程序就按照方程式计算 B 的值，并将所得值用指令 ET[N]=,B 分配给表。

指令	说明
#SETUP	标号
EAA	选择 A 为主动轴
EM2000,1000	凸轮周期
EP20,0	主动轴位置增量
N=0	标志
#LOOP	循环从方程式建立凸轮表
P=N*3.6	注：3.6=0.18*20
S=@SIN[P]*100	定义正弦位置
B=N*10+S	定义从动轴位置
ET[N]=,B	定义凸轮表
N=N+1	计数器加 1
JP#LOOP,N<=100	重复循环
EN	程序结束

第 9 步．创建程序运行 ECAM 方式

现在假设，从动轴用启动信号输入点 1 啮合，但啮合和脱开点必须在循环周期终点 A=1000，B=500 处进行，这样就必须驱动 B 轴到该点以避免跳动。

用程序实现如下：

指令	说明
#RUN	标号
EBl	使凸轮使能
PA,500	起始位置
SP,5000	B 轴速度
BGB	使 B 轴电机运动
AM	在 B 轴运动之后
All	等待起动信号
EG,1000	使从动轴啮合
AI-1	等待停止信号
EQ,1000	使从动轴脱开
EN	结束

电子凸轮指令汇总

指令	说明
EA	指定 ECAM 主动轴
EB	使 ECAM 使能
EC	设置 ECAM 表的索引
EG	使 ECAM 啮合

EM	指定主动轴循环周期和从动轴一个循环周期中的变化量
EP	定义 ECAM 表的起点和间隔
EQ	使 ECAM 在指定位位置脱开
ET[n]	定义 ECAM 表

电子凸轮操作数汇总

指令	说明
_EB	读取 ECAM 状态
_EC	读取当前 ECAM 表索引
_EG	读取各轴 ECAM 状态
_EM	读取各轴循环周期
_EP	读取 ECAM 表各项的间隔
_EQ	读取各轴 ECAM 状态

举例

1）ECAM

下例表示具有主动轴为 C 轴，两个从动轴为 A、B 轴的 ECAM 程序：

指令	说明
#A;V1=0	标号：初始化变量
PA0,0;BGAB;AMAB	A、B 轴运动到位置 0，0
EAC	C 轴作为 ECAM 主动轴
EM0,0,4000	C 轴变化量为 4000，A、B 轴为 0
EP400,0	ECAM 间隔为 400cts，以 0 为始点
ET[0]=0,0	主动轴在 0 位置时，为第 1 点
ET[1]=40,20	ECAM 表中的第 2 点
ET[2]=120,60	ECAM 表中的第 3 点
ET[3]=240,120	ECAM 表中的第 4 点
ET[4]=280,140	ECAM 表中的第 5 点
ET[5]=280,140	ECAM 表中的第 6 点
ET[6]=280,140	ECAM 表中的第 7 点
ET[7]=240,120	ECAM 表中的第 8 点
ET[8]=120,60	ECAM 表中的第 9 点
ET[9]=40,20	ECAM 表中的第 10 点
ET[10]=0,0	下一个循环周期的起始点
EB1	ECAM 方式使能
JGC=4000	设置 C 轴为 JOG 方式，速度为 4000cts / sec
EG0,0	当主动轴=0 时，使 A、B 轴啮合
BGC	开始 C 轴 JOG 运动
#LOOP;JP#LOOP,V1=0	一直循环到变量被修改
EQ2000,2000	当主动轴=2000 时，使 A、B 轴脱开
MP..2000	等待主动轴到达 2000
STC	停止 C 轴运动
EB0	退出 ECAM 方式
EN	程序结束

上例表示了创建 ECAM 程序的方法及将指令发送给控制器的方法。图 6.5 是用 WSDK 工具软件捕获到的各轴运动曲线，图 6.5 中的第一个曲线是 A 轴，第二个曲线是 B 轴，第三个

是 c 轴。

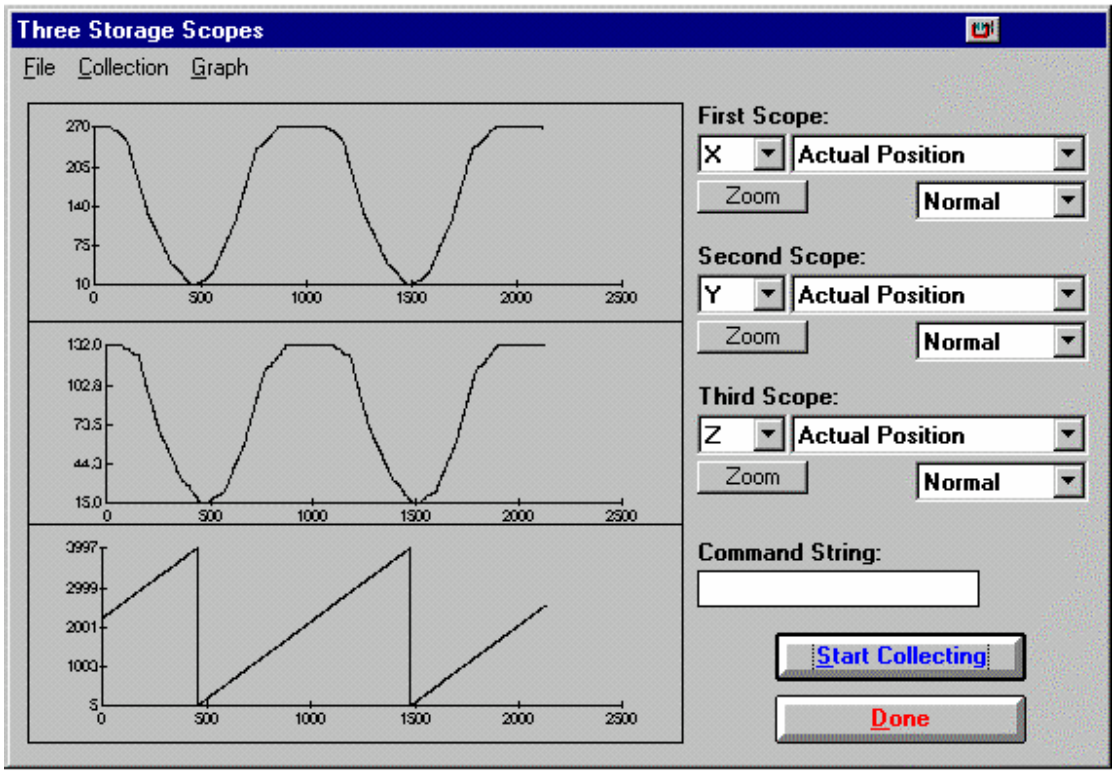


图 6.5-ABC 速度轮廓

轨迹方式

DMC-21x3 也具有轨迹功能，用此功能，用户能够在 1-8 轴内定义任意位置曲线，从而非常理想地跟踪由计算机产生的轨迹，如：抛物线、螺旋线或用户自定义的轨迹。此功能对轨迹类别无限制，轨迹长度也无限制。

指定轨迹线段

用指令 CM 来指定轨迹方式，例如，CMAC 将 A、C 轴指定为轨迹加工。没有用于轨迹方式的其它轴可以以其它方式工作。

轨迹用位置增量加以描述，用指令 CD a,b,c,d 及与之相对于时间间隔 DTn 来定义位置增量。参数 n 指定时间间隔。时间间隔定义为 2^nms，其中：n 为 1~8。控制器在指定的增量之间进行直线插补，对增量的插补按各毫秒产生一个点来进行。

例如，我们以图 6.6 为例，由以下各点来描述 A 轴位置。

点 1　　T=0ms 时，　A=0
点 2　　T=4ms 时，　A=48
点 3　　T=12ms 时，　A=288
点 4　　T=28ms 时，　A=336

同样的轨迹用增量表达如下：

增量 1　DA=48　　时间=4　　DT=2
增量 2　DA=240　　时间=8　　DT=3
增量 3　DA=48　　时间=16　　DT=4

当控制器收到沿这些点产生轨迹的指令时，它就在这些点之间进行直线插补。插补各点的结果为：在 lms 处，位置值 12，在 2ms 处，位置值 24，等等，以此类推。关于上例中的程

序如下：

指令	说明
#A	标号
CMA	指定 A 轴为轨迹方式
DT2	指定第 1 个时间间隔为 2^2ms
CD48;WC	指定第 1 个位置增量
DT3	指定第 2 个时间间隔为 2^3ms
CD240;WC	指定第 2 个位置增量
DT4	指定第 3 个时间间隔为 2^4ms
CD48；WC	指定第 3 个位置增量
DT0;CD0	退出轨迹方式
EN	程序结束

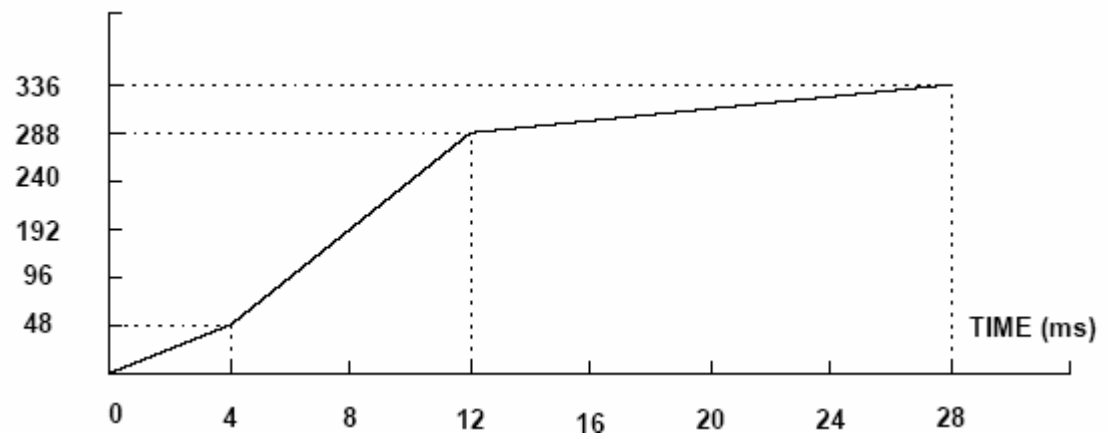


图 6.6-单轴轨迹方式位置-时间曲线

附加指令

指令 WC 用作为轨迹方式时的条件启动，它意味着等待时间到达结束前一指令执行时再执行后续指令。用 DT0 和 CD0 配合使用就退出轨迹方式。

如果控制器未发现新的数据记录，且仍在轨迹方式，控制器就等待新数据。在等待当中，控制器不产生新的运动指令。如果控制器接收到错误数据，就以问号（?）回应。

轨迹方式指令汇总

指令	说明
CM	指定用于轨迹方式的坐标轴，任何非轨迹轴可以工作在其它方式
CD	按时间间隔指定位置增量，增量范围为±32000。在 CD0 与 DT0 配合使用时，就使轨迹方式结束。
DT	对位置增量指定时间间隔，其中 n 是 1-8 之间的整数。0 使轨迹方式结束，如果 n 不改变，就不需要与各条 CD 指令一同指定。
WC	在处理下一个数据记录之前，等待前一时间间隔完成

通用速度轨迹

轨迹方式非常适用于产生任意速度轨迹。可以把速度轨迹指定为数学函数或点的汇集。设计包含两部分：产生存储数据点的数组和运行程序。

举例

1）产生数组

我们先看看图 6.7 中所示的速度和位置轨迹曲线。目的是在 120ms 内，使电机运动 6000cts，速度轨迹为正弦波曲线，以减少冲击和系统振荡。如果我们以 Bms 内的 Acts 描述位移重，我们就能用下式描述运动方程。

$$\omega = \frac{A}{B}(1 - \cos(2\pi T / B))$$

$$X = \frac{AT}{B} - \frac{A}{2\pi}\sin(2\pi T / B))$$

注：ω 是角速度；A 是位置；T 是变量。时间以 ms 为单位。

在此例中，A=6000，B=120，位置、速度轨迹方式为：

$$A = 50T - (6000 / 2\pi)\sin(2\pi T / 120)$$

注意：速度 ω 以 cts / ms 为单位，方程式如下：

$$\omega = 50(1 - \cos(2\pi T / 120))$$

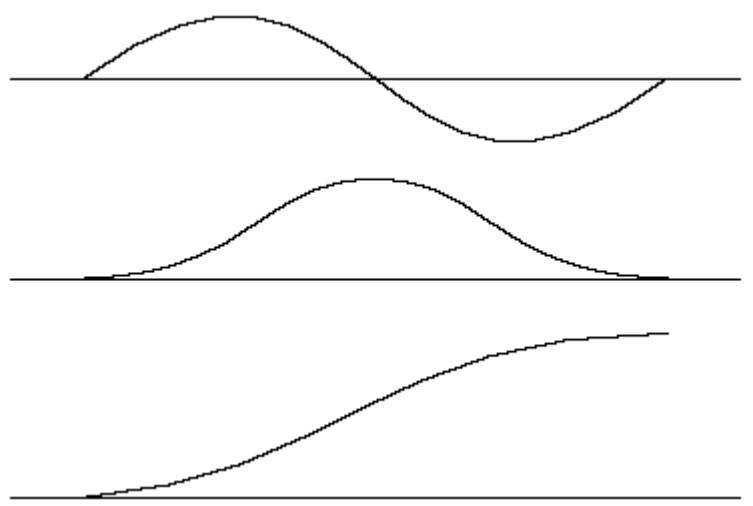


图 6.7-具有正弦波加速的速度轨迹

DMC-21x2/ DMC-21x3 具有三角函数运算功能，以度为单位。结合我们的举例，A 的方程式表达为：

A=50T-955sin3T

以下是本例中产生轨迹运动的完整程序。为产生数组，我们以 8ms 的间隔计算位置值，并将其存在数组 POS，然后，计算各位置之间的差值，存入数组 DIF，最后，以轨迹方式运转电机。

轨迹方式举例程序

指令	说明
#POINTS	标号，定义 A 轴位置点的程序
DM POS[16]	分配存储器
DM DIF[15]	
C=0	设置初始条件，C 是标志
T=0	T 是时间，以 ms 为单位
#A	
VI=50*T	

```
V2=3*T                  以度为单位的参数
V3=-955*@SIN[V2]+V1     计算位置
V4=@INT[V3]             V3 值取整
POS[C]=V4               存入数组 POS
T=T+8
C=C+1
JP#A,C<16
#B                      求位置差的程序标号
C=0
#C
D=C+1
DIF[C]=POS[D]-POS[C]    计算差值并存储
C=C+1
JP#C,C<15
EN                      第一个程序结束
#RUN                    运转电机的程序标号
CM X                    轨迹方式
DT3                     4ms 间隔
C=0
#E
CD DIF[C]               轨迹距离存入 DIF
WC                      等待完成
C=C+1
JP#E,C<15
DT0
CD0                     停止轨迹方式
EN                      程序结束
```

2）示教（录返功能）

有些应用需要对机器运动轨迹进行示教，对此，用 DMC-21x2/ DMC-21x3 数组自动捕获功能来捕获位置数据即可实现示教，然后以轨迹方式对捕获的数据进行录返。使用如下数组指令：

指令	说明
DMC[n]	规划数组
RAC[]	为自动记录规定数组（DMC-2142/ DMC-2143 为最多 4 个）
RD_TPA	指定捕获的数据(如_TPA 或_TPC)
RCn,m	指定捕获时间间隔，其中 n 为 2^n 采样时间，m 是捕获到的记录数值
RC?或_RC	如果在记录，返回值为 1

录返举例程序：

```
指令                说明
#RECORD             程序标号
DP0                 将 A 轴位置定义为 0
DA*[]               清空所有数组
DM xpos[501]        分配调用 xpos 的 501 个元素数组
RA xpos[]           将元素记录入 xpos 数组
```

RD_TPA	要记录的元素为 A 轴编码器位置
MO A	A 轴使能关断
RC2	以 2^2ms 采样周期开始记录
#LOOPl;JP#LOOP1,_RC1	循环直到所有元素都被记录
#COMPUTE	确定连续点之间差值的子程序
DM dx[500]	为保持轨迹点分配 500 个元素数组
i=0	设置循环计数器
#LOOP2	循环计算差值
dx[i]=xpos[i+1]-xpos[i]	计算差值
i=i+l	循环计数器加 1
JP#LOOP2，i<500	继续循环，直到 dx 置满
#PLAYBK	录返运动子程序
SHX	伺服使能
WTl000	等待 1sec(1000msec)
CMA	指定 A 轴为轨迹方式
DT2	设置轨迹数据采样间隔为 2^2msec
i=0	设置数组标志为 0
#LOOP3	执行轨迹点的子程序
CDdx[i];WC	轨迹数据指令，等待下一个轨迹点
i=i+1	标志加 1
JP#LOOP3,i<500	循环，直到执行完所有数组元素
DT0	设定轨迹更新率为 0
CD0	轨迹方式关断(与 DT0 配合使用)
EN	程序结束

关于自动数组捕获的其它内容，请参见第 7 章中“数组”一节

虚拟轴功能

DMC-21x2/ DMC-21x3 控制器具有虚拟轴功能，指定为 N 轴。此轴没有编码器输入也没有控制驱动器的输出。不过，能够用如下指令对其编程：

AC、DC、JG、SP、PR、PA、BG、IT、GA、VM、VP、CR、ST、DP、RP、EA

虚拟轴的主要用途是在 ECAM 方式中设置虚拟主动轴，以及执行插补方式的不需要的部分运动。这些例子通过下面一些例子加以说明。

1）ECAM 主动轴举例

假设：强制 A、B 轴沿着由 ECAM 表所描述的轨迹运动，并假定，ECAM 主动轴不是外部编码器但必须是可控制的变量。

通过用指令 EAN 把 N 轴定义为主动轴，并用指令设置主动轴的模（例如 EMN=4000）来实现这一要求，其次，创建凸轮表，要使强制轴运动，就简单地以 JOG 方式或指令 PR、PA 指令 N 轴。

例如：PAN=2000;BGN

会使 A、B 轴技运动周期运动到相对应的点。

2）正弦波运动举例

假设：X 轴必须进行 10 个循环的正弦波运动，幅值为 1000cts，频率为 20Hz，据此要求，我们让 A 轴和 N 轴进行圆弧运动，注意：VS 的值必须是：

VS=2π*R*F

其中 R 是半径或幅值，F 是频率，单位为 Hz.

设置 VA 和 VD 为最大值，以求最快的加、减速度。

指令	说明
VM AN	选择轴
VA 68000000	最大加速度
VD 68000000	最大减速度
VS l25664	VS 为 20Hz
CR l000,-90,3600	10 圈
VE	
BG S	开始矢量运动

步进电机工作方式

将控制器配置为步进电机工作方式时，个别指令与伺服方式有所不同。下面我们就对步进电机工作方式的不同点加以说明。

设置步进电机工作方式

为了使控制器工作在步进电机方式，在硬件方面，必须短接用于步进电机方式设定的跳线，详见第 2 章。

在硬件设置完成之后，用指令 MT 从软件功能上进行设置。用于指令 MT 的参数如下：

2	设置输出为负脉冲（共阳连接）
-2	设置输出为正脉冲（共阴连接）
2.5	设置输出为负脉冲（共阳连接）方向信号取反
-2.5	设置输出为正脉冲（共阴连接）方向信号取反

步进电机加减速平滑处理

指令 KS 具有步进电机加减速平滑处理功能。平滑处理的效果可以视为一个简单的阻一容（单极性）滤波器。滤波器位于运动轮廓分配器之后，使脉冲输出间隔发生变化，从而使步进电机工作更为平稳，因此起到平滑作用。当步进电机以满步或半步工作时，KS 的作用最为明显。KS 会使步进脉冲按照所规定的时间常数进行延迟。当以步进电机方式工作时，总是要用 KS 设置一定量的平滑处理：由于滤波效果出现在轮廓分配器之后，因此，在所有步进脉冲通过滤波器之前，轮廓分配器就在准备其它后续运动。也能够使用一般的运动平滑处理指令 IT，指令 IT 的用途是使运动轮廓输出更为平滑，减小由于加速度而引起的冲击。

监视产生脉冲——指令脉冲

为了控制器工作正常，就需要确保在进行其它后续运动之前，控制器已完成发送所有步进脉冲的工作。如果步进电机来回运动，这一点就尤为重要。例如，当以伺服电机方式工作时，就用条件启动指令 AM 来确认运动轮廓分配器何时完成并何时准备执行新的运动指令。不过，在步进电机方式工作下，当运动轮廓分配完成时，控制器可能还在产生步进脉冲；这是由于步进电机平滑滤波器而引起。为了更好地理解这点，我们看看控制器进行产生步进脉冲的过程。

首先，控制器按照运动指令产生运动轮廓。

第二，轮廓分配器产生由运动轮廓所指令的脉冲。由运动轮廓分配器产生的脉冲可以用指令 RP(参考位置)来监控，RP 读取的是由运动轮廓分配器所确定的位置绝对值。用 DP 指令来设置参考位置值；例如，DP0，将 A 轴的参考位置定义为 0。

第三，运动轮廓分配器的输出由步进平滑滤波器进行滤波。此滤波器将延迟加入步进电机脉冲的输出中。延迟量取决于用指令 KS 所规定的参数。如前所述，总会有一些步进电机平滑量。

KS 的缺省值是 1.313，此值对应于 4 个采样周期的时间常数。

第四，对步进电机平滑滤波器的输出进行缓冲，然后输出到步进电机驱动器。可以用指令 TD 监视由平滑滤波器产生的脉冲。TD 返回由缓冲实际输出所决定的位置绝对值，DP 指令设置步进计数寄存器的值以及参考位置值。例如，DP0 将 A 轴的参考位置设置为 0。

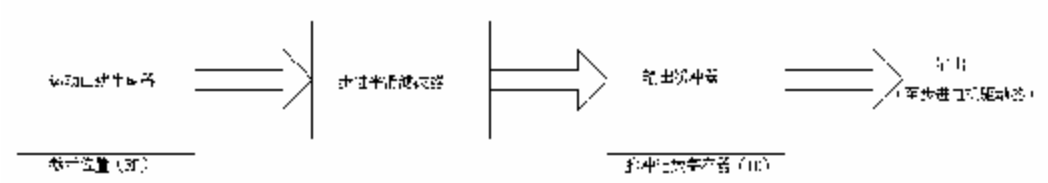


图 6.8

运动完成条件启动

在步进电机方式中，用指令 MC 使执行处理暂停，只有当指令位置中所指定的脉冲数与步进寄存器的输出脉冲数相同时，控制器才会继续执行后续指令。一般来说，MC 条件启动指令比 AM 条件启动指令更为有用，原因在于：由于步进电机平滑功能，步进脉冲从指令位置开始能够得到延迟。

编码器用于步进电机

可以将编码器用于步进电机上，来检查电机实际位置。如果使用编码器，就必须将其连接到主编码器输入口。

注意：当以步进电机方式工作时，不能使用辅助编码口。编码器的位置可以用指令 TP 来查询；用指令 DE 能定义位置值。

注意：当以步进电机方式工作时，控制器不能以闭环方式工作。

步进电机工作方式指令汇总

指令	说明
DE	定义编码器位置（当使用编码器时）
DP	定义参考位置和步进计数寄存器
IT	运动轮廓平滑时间常数——独立运动时间常数
KS	步进电机平滑常数
MT	指定电机类型（用于步进电机时，为 2，-2，2.5，-2.5）
RP	读取指令（参考）位置
TD	读取由控制器产生的步进脉冲数
TP	读取编码器位置

步进电机工作方式操作数汇总

指令	说明
_DE	脉冲计数寄存器的值
_DP	主编码器值
_IT	独立运动平滑时间常数
_KS	步进电机平滑常数
_MT	电机类型（用于步进电机时，为 2，-2，2.5，-2.5）

_RP	指令（参考）位置
_TD	脉冲计数寄存器的值
_TP	主编码器位置

双闭环（辅助编码器）

DMC-21x2/ DMC-21x3 为每个轴的第二编码器提供了接口，但对配置为步进电机工作方式及用于位置比较输出功能轴来说，第二编码器口不起作用。使用时，第二编码器可以安装在任何位置，但一般都与主编码器分别安装在电机或负载侧。第二编码器最通常的用途是背隙补偿，如下所述。

第二编码器可以是标准正交型，也可以是脉冲+方向型。控制器也提供有转换编码器旋转方向的功能设定。用 CE 指令配置主、辅编码器。指令形式为 CE a,b,c,d（或 a,b,c,d,e,f,e,h 用于 4 轴以上控制器），其中，参数 a，b，c，d 各等于两个整数 m、n 之和。m 配置主编码器，而 n 配置辅助编码器。

注意：用于配置为步进电机的轴时，不能改变辅助编码器的设置。

使用 CE 指令

m=	主编码器	n=	辅编码器
0	正交信号	0	正交信号
1	脉冲/方向信号	4	脉冲/方向信号
2	反向正交信号	8	反向正交信号
3	反向脉冲/方向信号	12	反向脉冲/方向信号

例如，要将主编码器配置为反向正交，m=2，而第二编码器为脉冲/方向，n=4，总和为 6，因此用于 A 轴的指令是：

CE6

用于辅助编码器的其它指令

指令 DE a，b，c，d 可用来定义辅助编码器的值，例如，

DE 0，500，-30，300

此指令将 A，B，C，D 轴辅助编码器的当前初始值定义为 0，500，-30，300。

辅助编码器的值可以用指令 DE?进行查询，例如，

DE?,,?

此指令返回 A，C 轴辅助编码器值。

辅助编码器位置可以用指令分配给变量，如下：

V1=_DEA

指令 TD a,b,c,d 返回辅助编码器的当前位置。

指令 DV a,b,c,d 将使用的辅助编码器配置成自动双闭环。

背隙补偿

有两种使用辅助编码器的间隙补偿方法：

1. 连续双闭环
2. 采样双闭环

为了方便起见，我们看看电机和负载之间存在传动间隙的情况。为了补偿间隙，在电机和负载侧都安装位置编码器。

第一种方法，连续双闭环把两个反馈信号相结合以达到稳定，这种方法需要仔细调整系统，且取决于间隙的大小。不过，一旦成功，这种方法就可以彻底解决间隙问题。

第二种方法，采样双闭环，只在终点读取负载编码器并进行修正。这种方法与间隙大小无关。不过它仅对在只要求终点定位精度的系统系统有效。

注意：如果间隙的大小是固定的，还有一种间隙补偿的方法，那就是用测量仪器精确测出电机与负载之间的间隙值，然后将此值作为间隙补偿量输入到控制器中，在运动过程中予以补偿。

举例

1）连续双闭环

将负载编码器连接到主编码器口，将电机侧编码器连接到第二编码器口。双闭环方法在两个编码器之间把滤波器功能进行分离。以负载编码器为基础，将 KP 和 KI 用于位置误差补偿，将 KD 用于电机侧编码器。这种方法可以构成稳定的系统。

用指令 DV(双速度环)使双闭环方法有效，其中：

DV1 ，1，1，1

此指令就使 4 个轴的双闭环功能有效。

DV0 ，0，0，0

此指令就使双闭环功能无效。

注意：双闭环补偿取决于间隙幅度大小，间隙过大，会使闭环系统不稳定。所建议的补偿步骤是以 KP=0，KI=0 开始，在 DVl 条件下，使 KD 值调至最大。一经求得 KD，就渐渐地将 KP 增至最大值，最后，如果需要，再增大 KI。

2）采样双闭环

在此例中，我们看看由旋转电机经滚珠丝杠带动直线滑台。由于滚珠丝杠有间隙，就需要用直线编码器(即光栅)来检测滑台位置；从稳定性方面来讲，在电机侧最好使用旋转编码器。将旋转编码器连接到 A 轴主编码器口，而将光栅连接到 A 轴的辅助编码器口。假定：所要的运动距离是 1 厘米，它对应于旋转编码器反馈值为 40000cts，光栅反馈值为 10000cts。

设计方法是驱动电机移动一段距离，对应 40000cts 旋转编码器计数值，运动一完成，控制器就检测到光栅反馈的位移值，并进行位置修正。

由以下程序来实现：

指令	说明
#DUALOOP	标号
CE0	配置编码器
DE0	设置初始值
PR40000	主要运动
BGA	开始运动
#CORRECT	修正循环
AMA	等待运动完成
V1=10000-_DEA	求直线编码器误差
V2=_TEA／4+V1	补偿电机误差
JP#END,@ABS[V2]<2	若误差<2，就退出
PR V2*4	修正移动量
BGA	开始修正
JP#CORRECT	重复循环
#END	标号
EN	程序结束

运动平滑(S 曲线加减速)

DMC-21x3 控制器具有速度轮廓平滑功能，通常也称之为 S 曲线加/减速功能。以此来减小系统的机械冲击。

梯形速度轮廓（即线性加减速）使加速度突然从 0 变到设定的最大值，不连续的加速度往往会引起振荡，产生冲击。加速度轮廓平滑功能产生连续加速轮廓并减小机械冲击和振荡。

用 IT 和 VT 指令使运动平滑：

IT 和 VT 指令可以实现运动平滑功能。这些指令对加、减速函数进行滤波，以产生平滑的速度轮廓。所产生的速度轮廓具有连续加速性能，从而减小机械冲击。

用以下指令来指定平滑功能：

ITa，b，c，d　　独立运动时间常数

VTn　　插补运动时间常数

指令 IT 用于平滑 JG、PR、PA 等非插补方式的运动，而指令 VT 用于平滑 VM、LM 等插补方式的矢量运动。

平滑参数 a，b，c，d 和 n 是 0~1 之间的数，它决定滤波强度。最大值 1 用于产生梯形速度轮廓的滤波器。平滑参数值越小意味着滤波作用越强，运动越平滑。

下例说明平滑效果，图 6.9 表示梯形速度轮廓及加入平滑滤波后的加速度和速度曲线。

注意：平滑处理会使运动时间延长。

举例

指令	说明
PR20000	位置
ACl00000	加速度
DCl00000	减速度
SP5000	速度
IT0.5	平滑滤波器
BGA	开始运动

用 KS 指令使运动平滑（用于步进电机）

当以步进电机方式工作时，还可以用指令 KS 能实现运动平滑功能。KS 指令对步进电机脉冲的频率进行平滑处理。与指令 IT、VT 相类似，KS 也产生平滑的速度轮廓。

用以下指令指定步进电机平滑功能：

KS a，b，c，d

其中：a，b，c，d　是 0．5~8 之间的数，它代表平滑量（滤波深度）。最小值 0.5 意味着没有滤波，仍然是梯形速度轮廓。滤波参数值越大，意味着滤波作用越强，运动就越平滑。

注意：KS 只对脉冲输出方式有效。

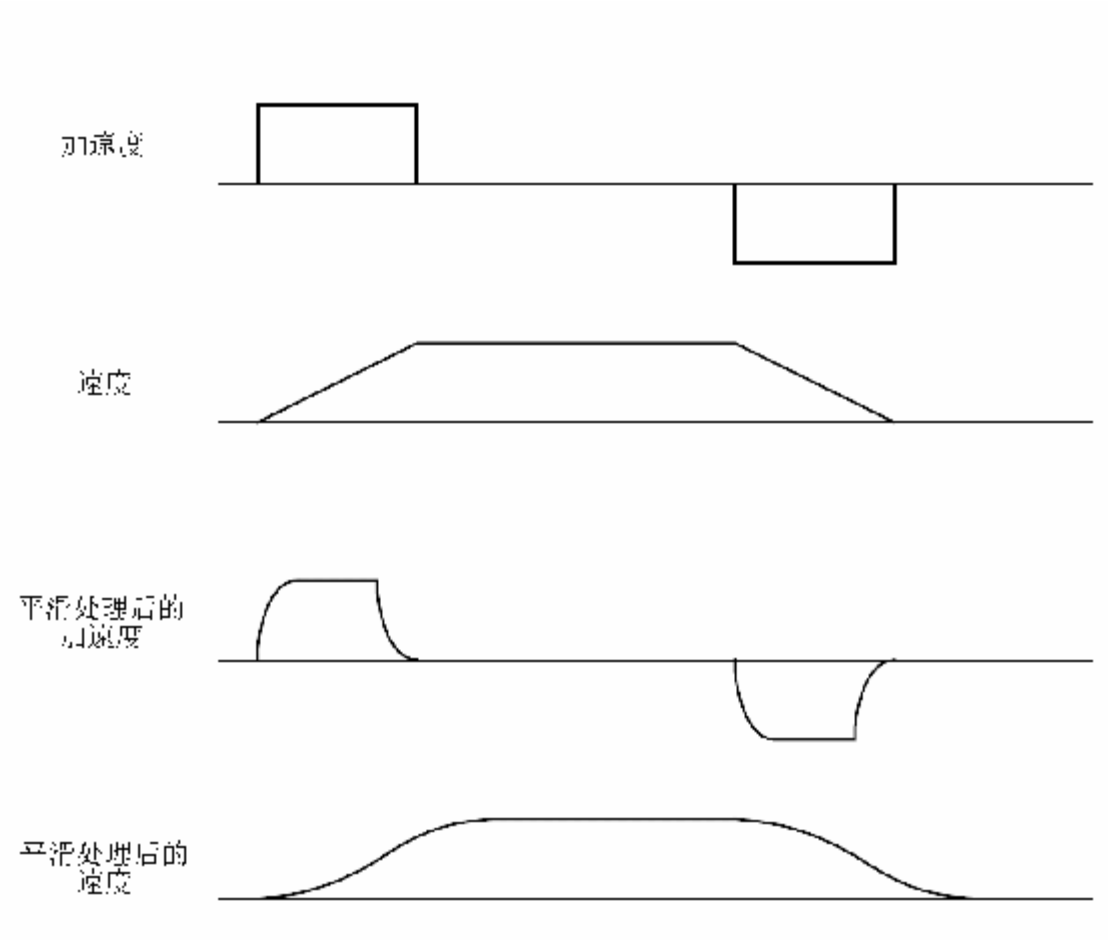


图 6.9-梯形包络速度轮廓和平滑速度轮廓

原点返回功能

可以用 FE、HM 指令使电机回到机械参考点，此参考点称之为原点开关，连接到原点输入端子。HM 指令除检测原点输入信号之外，还检测电机编码器标志脉冲（也称为零位脉冲）信号。用指令 CN 来定义原点输入信号的极性。

FE 指令对于检测原点开关很有用，原点开关连接到原点输入端。当使用 FE 和 BG 指令时，电机就会加速到指定的旋转速度，并旋转直至检测到原点开关输入信号状态变化，然后，电机就会减速停止。为了使电机在检测到原点开关之后快速减速，在执行 FE 指令之前必须输入大的减速值。产生的速度轮廓曲线示于图 6.10。

HM 指令能用来使控制器检测到原点开关之后，指令电机按标志脉冲进行定位，从而获得更为精确的初始位置。HM 和 BG 指令的动作过程如下：

1．一开始运动，电机加速到工作速度，运动方向由原点输入的状态来决定。低电平使电机以正向开始运动，高电平使电机以反方向开始运动。用 CN 指令来定义原点输入极性。

2．一检测到原点开关改变状态，电机就开始减速停止。

3．然后电机就慢慢返回，直到再次检测到原点开关状态翻转。

4．接着，电机正向运动，直至检测编码器标志脉冲。

5．DMC-21x3 将检测到标志脉冲的位置定义为原点位置(0)。

关于其它几种原点返回方法及原点返回程序的设计技巧请参考本书第 3 章中关于“原点开关输入”一节的说明及《指令手册》中关于 HM、FE、FI 指令解释。

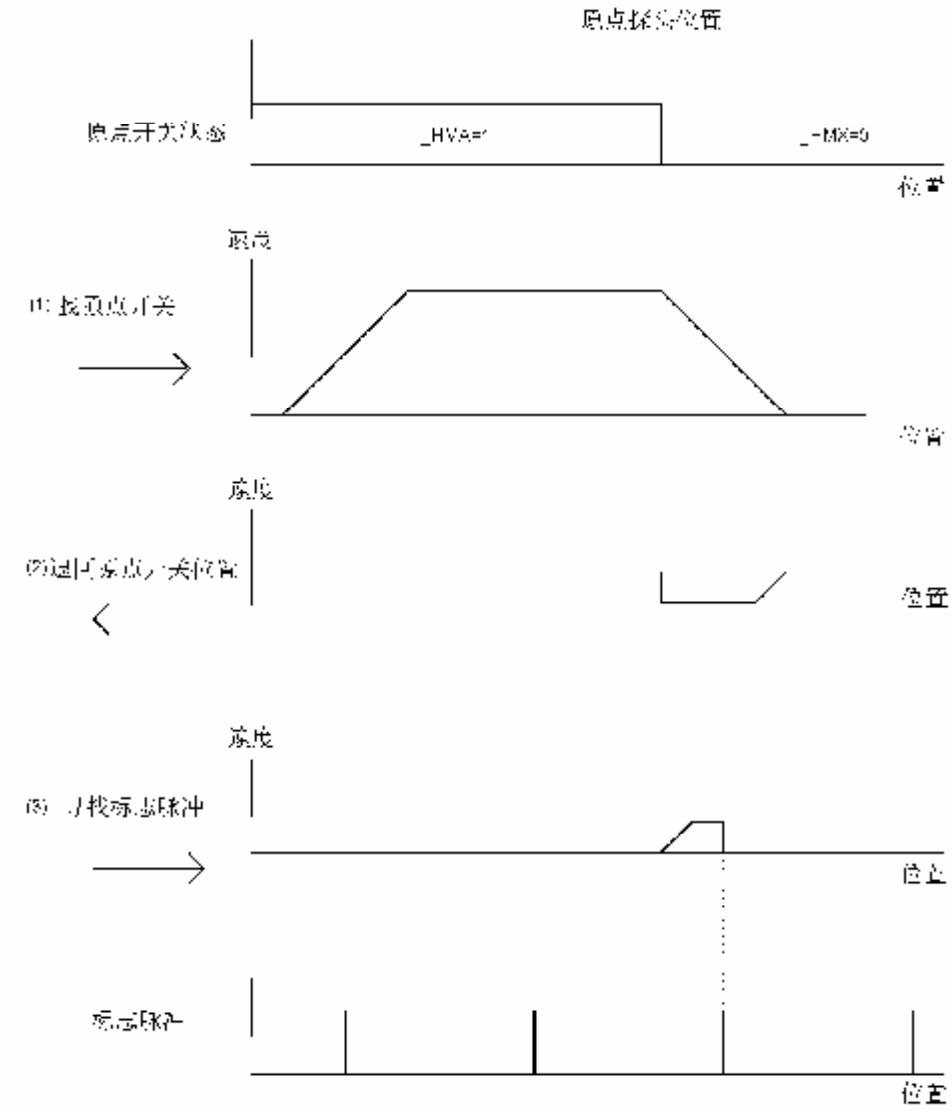


图 6.10-原点返回动作顺序

举例

指令	说明
#HOME	标号
ACl000000	加速度
DCl000000	减速度
SP5000	回原点速度
HM A	A 轴原点返回
BG A	开始运动
AM A	运动完成后
MG ”AT HOME”	发送信息
EN	结束
#EDGE	标号
AC2000000	加速度
DC2000000	减速度
SP8000	速度
FEB	搜寻原点开关边缘指令

BGB	开始运动
AMB	运动完成后
MG"FOUND HOME"	发送信息
DP,0	定义此位置为 B 轴 0 点
EN	结束

原点返回指令汇总

指令	说明
FE	搜寻原点开关边缘
FI	搜寻标志脉冲
HM	返回原点（相当于 FE 与 FI 的结合）
SC	查询停止代码
TS	查询各开关状态

原点返回指令汇总

指令	说明
_HM	原点开关状态
_SC	停止代码
_TS	各开关状态

高速位置捕捉（位置锁存功能）

对于印刷套色等需要定标应用来说，往往需要精密捕获位置。DMC-21x3 具有位置锁存功能，当锁存输入状态发生变化时，用此功能就能够捕获到各轴主编码器或辅助编码器的位置。对此功能进行配置，可以选择当锁存输入变高或变低时对位置进行捕获。当锁存功能使能为有效低电平工作时，在 12 微秒之内就会捕获到位置值，当锁存使能为有效高电平工作时，在 35 微秒之内就会捕获到位置值。各轴都有一个为位置捕获而对应的通用输入。

输入	功能	输入	功能
IN1	A 轴锁存	IN9	E 轴锁存
IN2	B 轴锁存	IN10	F 轴锁存
IN3	C 轴锁存	IN11	G 轴锁存
IN4	D 轴锁存	IN12	H 轴锁存

用 DMC-21x2/ DMC-21x3 软件指令 AL，RL 使锁存功能使能，并给出锁存的位置。使用锁存的步骤如下：

1．给出 AL ABCD 指令使锁存对主编码器使能，而 AL SASBSCSD 对辅助编码器锁存使能；

2．用_AL A 或 B 或 C 或 D 指令，测试看看锁存是否出现（输入变低电平）。例如，

VI=_AL A

将轴 A 锁存的状态放入 VI 变量，如果未出现锁存，VI=0;

3．在出现锁存之后，用 RL ABCD 或_RL ABCD 读取捕获的位置。

注意：在每次锁存发生之后，必须对锁存重新使能。

举例：

指令	说明
#LATCH	标号
JG,5000	JOG B 轴

BGB	B 轴开始运动
ALB	B 轴锁存使能
#WAIT	等待循环程序标号
JP#WalT,_ALB=1	如果锁存未出现，重复循环等待
Result=_RLB	设置 Result 等于 B 轴锁存的位置
Result=	打印结果
EN	程序结束

位置比较输出

在喷绘、以及其它形式的点阵加工的应用中，需要位置同步信号。当运动速度不高、位置精度要求不高的情况下，可以使用 AP 指令配合通用输出实现，但是当运动速度较高、位置精度要求较高的场合，这种方式的响应速度不足以达到应用要求。GALIL 为这种应用设计了专用的位置比较输出功能。

位置比较输出功能是在编码器位置到达指定值时，DMC-21x3 在专用的输出点（CMP）上输出宽度为 600ns 的负脉冲。用户可以指定在到达指定位置后输出一个宽脉冲，还是从指定位置开始，每经过一定的距离向外发送一个位置同步信号。

使用位置比较输出功能，用作位置基础的必须是某轴的主编码器输入，而且该轴还有其它一些限制：必须定义为模拟量输出方式（MTx=1 或-1）；主编码器与辅助编码器的定义必须完全一致（CEx=0、5、10 或 15）；该轴的辅助编码器不能用于编码器信号的接入。

指令 OC 用来指定位置同步信号的输出，OC 指令的常用格式如下：

OCx=m,n

其中　x 指定用于位置比较的轴，对于 DMC-2183 而言，可以是 A-H 中任何一个轴

m 是输出首个同步脉冲的位置（绝对坐标）

n 是同步信号之间的距离间隔，在+/-65535 之间

指令 OCx=0 用于取消位置同步信号输出。

指令使用中要注意，n 的符号应与编码器信号的方向一致，如果方向不一致，则相当于设置为 65536-|n|。因此，如果编码器的速度较低且不稳定，|n| 接近 65535 时，可能产生错误的输出。（当编码器信号速度较低且不稳定时，可能会在某个瞬间的速度与平均速度的方向向反，如果 65536-|n| 的值很小，比如 n 为 65534，则瞬时的编码器信号瞬时反向也可能产生不必要的输出。

此外，使用此功能时，由于输出信号为宽度不足 1us 的脉冲信号，所有此信号相关的传输、接收应选择带宽 10MHz 以上的器件。

第七章　应用

概述

在第 6 章中，我们对 DMC21x3 所具有的基本运动控制功能做了全面介绍。以此为基础，用户即可以利用 API 函数用 VB、VC、C / C++、Labview、delph 等高级语言编写满足特殊需要的应用程序，也可以利用控制器已具有的运动控制功能编写这些应用程序。一般来说，利用已有的运动控制功能就可编写出满足特殊需要的应用程序；只不过在某些应用场合，需要具有良好的人机界面，在此情况下，用户有必要采用高级语言进行编程。

我们已知道，DMC21x3 提供了功能强大、简单易记的编程语言，使用户能够精心打造满足特殊应用要求的、具有自身特点的专用控制器。本章以前 6 章内容为基础，全面介绍创建应用程序的方法和技巧，使用户能够灵活而充分地运用 DMC21x3 的基本功能。

GALIL 运动控制器本身具有 32bit 高速处理器及 Flash、RAM 存储器，进行程序、指令解释、运动模式规划、I/O 事件处理、高速插补运算、闭环控制、通信管理等。因此我们可以把程序下载到 DMC21x3 存储器中，然后脱离主机运行，主机可以从事其它任务处理。不过，在需要的情况下，主机仍然能够随时给控制器发送指令，即使在程序正在执行当中，也是如此。对于这种应用编程而言，只能使已提供的 ASCII 格式的指令。

除标准运动指令之外，DMC21x3 还提供了一些指令，使 DMC21x3 能根据运行中的一些条件进行自身判断。这些指令包括条件跳转、事件触发及子程序等。例如，指令 JP#LOOP，n<10，如果变量 n<10，程序就跳转到标号#LOOP。

为使编程更为灵活，DMC21x3 具有用户自定义变量、数组及算术运算功能。例如，在定长裁断应用中，在程序中，能够把长度指定为变量，从而操作者就可以根据现场需要随时加以改变。

用户程序存储区大小为 1000 行×80 字符。

程序编辑

GALIL 提供了两种版本的编辑器，一种是 DOS 版的内置编辑器，另一种是 Windows 版本，使用户能够很方便地在控制器的存储器中创建并编辑程序。在 DOS 环境下，内编辑器用指令 ED 打开；而在 Windows 环境下，只要运行 SmartTerm 文件，就会自动打开基于编辑器的窗口。基于编辑器的窗口具有更多的功能，而且容易使用，因此，建议用户尽可能使用基于编辑器的窗口，而当使用简单终端时，才使用内置编辑器。

一给出 ED 指令，各程序行就自动按序从 000 开始编号。如果没有参数跟随在 ED 指令之后，编辑器提示符缺省出现在存储器中的最后一个程序的最后一行。如果需要的话，通过在 ED 后面指定行号或标号，用户就能对指定的行号或标号进行编辑。这种内置编辑器几乎与 DOS 版中的行编辑器软件相同。

注意：ED 指令只接受 DOS 窗口中的参数(如#BEGIN)。为了通用起见，本节中的编辑功能，不是 DOS 方式时就不能适用。

指令	说明
：ED	将编辑器放到上个程序的结尾处
：ED5	将编辑器放到第 5 行
：ED#BEGIN	将编辑器放到标号#BEGIN 处

行号表示为 000，001，002 以此类推，跟着行号输入程序指令。一行中可以编辑多个指令，只要总字符不要超过 80 个即可。

在编辑方式下，为了存储、插入、删除程序行，程序员可以调用特殊指令。这些特殊指令列表如下：

编辑方式指令

<RETURN>　回车指令

键入回车键使输入指令的当前行保存起来，编辑器会自动转入下一行。因此，敲击多个<RETURN>会使编辑器前进多行。注意，如果不给<RETURN>，就不会保存程序行的更改。

<Cntrl> P

<Cntrl> P 指令使编辑器移到前一行。

<Cntrl> I

<Cntrl> I 指令在当前行之前插入一行。例如，如果编辑器处于第 2 行，并用<Cntrl> I 指令，就会将新行插入第 1 行和第 2 行之间，新行的行号为 2，原第 2 行变为第 3 行。

<Cntrl> D

<Cntrl> D 删除当前正在编辑的行。例如，如果编辑器处在第 2 行。用<Cntrl> D 就会删除该行，原第 3 行又变为第 2 行。

<Cntrl> Q

<Cntrl> Q 退出编辑方式，DMC-2000 会返回一个冒号(：)作为回应。

在编辑完结束后，用户可以对编写的程序用 LS 指令进行列表，如果没有参数跟随在 LS 指令之后，就会列出整个程序。用户可以用参数 n 对所指定的行或标号进行列表。跟在起始列表参数之后的指令和新行号或标号指定列表要停止的位置。

举例：

指令	说明
：LS	对整个程序列表
：LS5	在第 5 行开始列表
：LS5，9	PJ 出 5~9 行
：LS#A，9	列出#A-9 行
：LS#A，#A+5	列出标号#A 和其余 5 行

注意：这种编辑器不适用于 DMC-2100，不过，可以使用任何其它终端(如 Telnet)。

程序格式

DMC 程序由配合解决机械设备控制要求的 DMC21x3 指令组成。起动、停止运动等动作性指令，程序流程指令相结合来形成完整的程序。程序流程指令对实时条件，如延迟时间或运动完成（到位）等进行评估判断，并以此改变程序流程。程序中的每条指令必须用分号（;）分开或用回车键结束。用分号把一行中的多条指令分开，每行中最多不能超过 80 个字符。用回车键使程序行结束。

在程序中使用标号

建议所有 DMC21x3 程序以标号开始，以 EN 语句结束。标号以#号开始，后面最多可跟随 7 个字符。第一个字符必须是字母，其后，允许使用数字，但不允许有空格。对于 1.0c 以上版本的控制软件（Fireware），最多可以定义 510 个标号。

有效标号举例：

#BEGIN
#SQUARE
#X1
#BEGIN1

无效标号举例：

#1Square
#123

举例：

指令	说明
#START	程序标号
PRl0000,20000	指定 A、B 轴相对运动距离
BGAB	开始运动
AM	等待运动完成
WT2000	等待 2sec
JP#START	跳转到标号 START
EN	程序结束

上述程序使 A、B 两轴分别移动 10000 和 20000 计数单位，在运动完成后，电机停 2sec，循环无限重复直到发送停止指令。

特殊标号

DMC21x3 有一些特殊标号，以此来定义输入中断子程序、限位开关处理子程序、报警处理子程序、指令出错子等程序。参见“自动启动子程序”一节。

DMC21x3 有一个特殊标号用于自动程序执行。在上电或复位时，就自动执行已存入控制器非易失性存储器并以标号#AUTO 开始的程序。必须用指令 BP 将程序存入非易失性存储器。

对条件进行监测的自动子程序见如下说明。

标号	说明
#ININT	输入中断子程序
#LIMSWI	限位开关处理子程序
#POSERR	存取位置误差子程序
#MCTIME	运动完成条件启动超时子程序
#CMDERR	指令出错子程序
#COMINT	通信中断子程序
#TCPERR	TCP/IP 通信出错子程序
#AMPERR	放大器错误子程序（只在使用 AMP-20540 时有效）
#AUTOERR	#AUTO 错误处理子程序

注释程序

注释程序有两种方法，第一种方法使用 NO 指令，可以在 GALIL 程序内直接使用；第二种方法是使用 REM 语句，这需要使用 GALIL 软件来进行。

NO 指令

DMC 控制器提供 NO 指令用于对程序注释。在 NO 指令之后，一行中可以写入 78 个字符，让程序员对程序进行注释，举例如下：

指令	说明
#PATH	标号
NO2-D CIRCULAR PATH	注释——无操作
VM AB	矢量方式

```
NO VECTOR MOTION ON A AND B    注释——无操作
VSl0000                        矢量速度
NO VECTOR SPEED IS l0000       注释——无操作
VP-4000,0                      矢量位置
NO BOTTOM LINE                 注释——无操作
CRl500,270,-180                圆弧运动
NO HALF CIRCLE MOTION          注释——无操作
VP0,3000                       矢量位置
NO TOP LINE                    注释——无操作
CRl500,90,-180                 圆弧
NO HALF CIRCLE MOTION          注释——无操作
VE                             矢量结束
NO END VECTOR SEQUENCE         注释——无操作
BGS                            开始运动
NO BEGIN SEQUENCE MOTION       注释——无操作
EN                             结束程序
NO END OF PROGRAM              注释——无操作
```

注意：NO 指令是实际控制器指令，因此，NO 指令会占用控制器的处理时间。

提示：有些用户用字“NOTE”对程序加注，在“NO”之后的每个字均视为注释。

REM 指令

如果您使用 GALIL 软件与 DMC21x3 控制器通信，就可以用 REM 语句。‘REM’语句以字‘REM’开始，在同一行中可以跟随任何注释。当下载程序到控制器时，GALIL 终端软件会去掉这些语句。例如：

```
#PATH
REM2-DCIRCULAR PATH
VMAB
REM VECTOR MOTIONON A AND B
VS 10000
REM VECTOR SPEED IS 10000
VP-4000，0
REM BOTTOM LINE
CR 1500,270,-180
REM HALF CIRCLE MOTION
VP0,3000
REM TOP LINE
CRl500,90,180
REM HALF CIRCLE MOTION
VE
REM END VECTOR SEQUENCE
BGS
REM BEGIN SEQUENCE MOTION
EN
REM END OF PROGRAM
```

当下载此程序到控制器时，会去掉这些 REM 语句。

执行程序——多任务

DMC21x3 可以同时运行 8 个独立程序。把这些程序称之为通道。分别从 0—7 进行编号，其中 0 是主通道。多任务功能对于执行功能相对独立的操作，如 PLC 功能等非常有用。

主通道与其它通道在以下方面有所不同：

（1）只有主通道（0 通道）可以使用输入指令 IN

（2）限位开关处理、位置超差、指令出错等子程序均在 0 通道执行。

要开始执行程序，使用如下指令：

XQ#A,n

其中：n 表示通道号，要暂停执行任意通道，用指令：

HX n

其中：n 是通道号

注意：XQ 和 HX 指令能通过运行中的程序来执行。

下例是在输出点 1 端产生波形的独立运动程序。

指令	说明
#TASK1	TASK1 标号
AT0	初始化参考时间
CB1	清除输入点 1
#LOOP1	LOOP1 标号
ATl0	从参考时间开始等待 10ms
SB1	输出点 1 置高电子
AT-40	从参考时间开始等待 40ms，然后初始化参考时间
CB1	清除输出点 1
JP#LOOP1	重复 LOOP1
#TASK2	TASK2 标号
XQ#TASK1,1	执行 TASK1
#LOOP2	LOOP2 标号
PRl000	定义相对距离
BGX	开始运动
AMX	在运动完成后
WTl0	等待 10ms
JP#LOOP2,@IN[2]	输入点 2 不变低，就一直重复运动
HX	暂停所有任务

用 XQ#TASK2,0 指令执行上述程序，TASK2 在主通道（即通道 0），在 TASK2 之内执行 #TASK1。

调试程序

DMC21x3 提供了用于调试应用程序的指令和操作数。这些指令有用于监测程序执行，确认控制器状态和控制器程序、数组和变量空间大小的查询指令。操作数也包含能够有助于调试程序的重要状态信息。

跟踪指令

跟踪指令会在程序执行之前，使控制器将程序中的每一行立即发送到主计算机。用指令

TR1 使跟踪功能使能，TR0 使跟踪功能关断。

从控制器送出的数据存储在输出 FIFO 缓冲区中。输出 FIFO 缓冲区能存储 512 个字符信息。在正常运行中，控制器把输出内容放入 FIFO 缓冲区，主计算机侧的软件监测缓冲区并读取所需信息。当跟踪方式使能时，控制器就以非常高的速率把这些信息发送到 FIFO 缓冲区。一般来说，由于软件读取信息较慢，因此 FIFO 就可能会放满，当 FIFO 放满时，如不清除，就会使程序执行速度减慢。如果用户想避免这种不必要的延迟，就发送指令 CW,1，此指令使控制器把那些不能放入 FIFO 的数据扔掉。在此情况下，控制器就不会延迟程序执行。

错误代码指令

当有程序错误时，DMC21x3 程序在错误发生点暂停。要显示程序执行的最后一行的行号就用指令 MG_ED。

用户用指令 TCl 能得到错误发生条件的类型，此指令报告错误代码号和描述出错条件的文本信息，指令 TC0 或 TC 返回不带文本信息的错误代码。详见《指令手册》中有关 TC 指令的说明。

停止代码指令

用停止代码指令 SC 能确认各轴的运动状态，当某轴运动莫名其妙停止时，此指令很有用。SC 指令会返回代表运动状态的代码号。详见《指令手册》。

RAM 存储器查询指令

为了调试程序存储器，数组存储器或变量存储器的状态，DMC21x3 有几个很有用的指令。指令 DM?返回当前可用的数组元素数；指令 DA?返回当前能被定义的数组数。例如，一台标准 DMC21x3 最多有 30 个数组，8000 个数组元素。如果已定义了一个 100 个元素的数组，DM?指令返回值就为 7900，DA?指令返回值就为 29。

为了列出变量区的内容，使用查询指令 LV（变量列表）即可；为了列出数组区中的内容，就使用查询指令 LA（数组列表）；为了列出程序区的内容，使用查询指令 LS（列表）；为了只列出应用程序标号，就使用查询指令 LL（标号列表）。

操作数

一般来说，所有操作数在调试应用程序中均可提供有用的相关信息。下面列出的这些操作数对程序调试特别有用。为了显示操作数的值，可以使用 MG 指令。例如，由于操作数_ED 读取程序执行的最后一行，那么，指令 MG_ED 就会显示此行号。

_ED	读取程序执行的最后一行。用于确定程序停止的地方。
_DL	读取可用的标号数
_UL	读取可用的变量数
_DA	读取可用的数组数
_DM	读取可用的数组元素数
_AB	读取急停输入状态
_FLa	读取‘a’轴正向限位开关的状态
_RLa	读取‘a’轴负向限位开关的状态

举例

下例有一个错误，在 A 轴处于运动中时，试图指定新的相对位移量。当执行此程序时，控制器停在 003 行，然后，用户就可以用指令 TEl 对控制器发问查询，控制器就会以相对的说明给予回应：

指令	说明

：ED	编辑方式
000#A	标号
001PRl000	相对位置 1000
002BGA	开始
003PR5000	相对位移 5000
004EN	END
<Cntrl>Q	退出编辑方式
：XQ#A	执行#A
?003PR5000	第 3 行有错误
：TCl	查询错误代码
?7Command not valid while running	在运行当中，指令无效
：ED3	编辑第 3 行
003AMX；PR5000；BGA	在运动完成后加入
<Cntrl>Q	退出编辑方式
：XQ#A	执行#A

程序流程指令

DMC21x3 提供了控制程序流程的指令，DMC21x3 通常按序执行程序指令。使用事件触发器、条件启动和条件跳转语句就能改变程序流程。

事件触发器和条件启动

要使控制器离开主计算机独立运行，就要对 DMC21x3 进行编程，根据出现事件做出决策。这样的事件有如下几种；等待运动完成、等待定时到达、等待输入逻辑电平改变。

DMC21x3 提供了好几种事件触发器，使程序暂停，直到所指定的事件出现才会继续程序执行。通常，自动按序逐行执行程序。不过，对事件触发指令译码时，暂停实际程序正常执行，直到事件触发器触发，程序才会继续执行。例如，能够用运动完成触发器把程序中的两个运动顺序分开；直到运动按第一个运动顺序完成才会执行用于第二个运动顺序的指令。在这种方式，没有主计算机干预，DMC21x3 也能按照自身状态或外部事件进行决策。

事件触发指令

指令	说明
AM	直到所指定轴或运动按序完成程序才继续执行。不带参数的 AM 适用于所有轴的运动完成，此指令对分开程序中的运动顺序很有用。
AD	使程序执行暂停，直到位置指令到达所指定的相对于运动始点的距离。一次只能指定一个轴
AR	使程序执行暂停，直到由上一个 AR 或 AD 指令所指定的距离过后，才执行后续指令。一次只能指定一个轴。
AP	使程序执行暂停，直到绝对位置出现之后，才继续执行。每次只能指定一个轴。
MF	使程序执行暂停，直到正向运动到达绝对位置后才继续执行。一次只能指定一个轴。如果位置已经越过指定位置点，MF 会立即释放。此功能对齿轮轴或辅助编码器输入起作用。
MR	使程序执行暂停，直到反向运动到达绝对位置之后才继续执行。一次指定一个轴。如果位置已越过指定位置点，MR 会立

	即释放。此功能对齿轮轴或辅助编码器输入起作用。
MC	使程序执行暂停，直到运动规划已经完成且编码器实际位置进入或越过指定位置才继续执行。如果不到位，TW A，B，C，D 设置超时报警。如果出现超时，条件启动就会清除，并将停止码设置为 99。应用程序会跳转到标号#MCTIME。
AI	使程序执行暂停，直到所指定的输入处于所规定的逻辑电平，n 指定输入点。正为高电平，负为低电平。n=l~8（用 DMC-2x10~2x40 时），n=l-24（用 DMC-2x50~2x80 时）。
AS	使程序执行暂停，直到所指定轴到指定速度
AT	使程序执行暂停，直到从起始计时 n msec 到达才继续执行。AT0 设置参考始点，ATn 从参考始点开始等待 n rnsec。AT-n 从参考始点开始等待 n ms 并在时间到达之后设置新的参考始点。
AV	使程序执行暂停，直到沿联动轨迹指定的距离出现
WT	使程序执行暂停，直到所指定的等待时间到达，以 ms 为单位。

举例

1）多重运动顺序

用 AM 条件启动指令把两个 PR 运动分开，如果不使用 AM 指令，由于只有在运动完成后才指定新 PR 值，所以对第二个 PR 指令，控制器会返回问号（?）

指令	说明
#TWOMOVE	标号
PR2000	相对位置值 2000
BGA	开始运动
AMA	等待运动完
PR4000	下一位置运动
BGA	开始第二个运动
EN	程序结束

2）在距离之后设置输出

在离开运动始点 1000cts 距离之后设置输出位 1 为高电平。条件启动的精度是采样周期乘以速度。

指令	说明
#SETBIT	标号
SPl0000	速度为 10000
PA20000	绝对位置
BGA	开始运动
ADl000	等待 1000cts 到达
SBl	设置输出位 1
EN	程序结束

3）重复性位置触发

在运动期间，每 10000cts 设置输出位，用 AR 条件启动。

指令	说明
#TRIP	标号
JG50000	指定 JOG 速度
BGA：n=0	开始运动
#REPEAT	重复循环#REPEAT
ARl0000	等待 10000cts 到达

TPA	报告位置
SB1	设置输出 1
WT50	等待 50ms
CB1	清除输出 1
n=n+l	循环计数+1
JP#REPEAT，n<5	重复循环 5 次
STA	停止
EN	程序结束

4）按外部输入开始运动

此例等待输入点 1 变低，然后开始运动

注意：AI 指令实际上使程序执行暂停，只有当输入出现时，才继续执行。如果您不想使程序执行顺序暂停，那么，您就要使用输入中断功能(Ⅱ)或条件跳转功能如：JP#GO，@IN[1]=1。

指令	说明
#INPUT	程序标号
AL1	等待输入点 1 变低
PRl0000	相对位置 10000
BGA	开始运动
EN	程序结束

5）当速度到达时，设置输出点

指令	说明
#ATSPEED	程序标号
JP50000	指定 JOG 速度
ACl0000	加速度
BGA	开始运动
ASA	等待速度到达 50000
SB1	输出点 1 置高
EN	程序结束

6）改变矢量轨迹速度

下列程序使进给率或矢量速度在指定的矢量运动距离点发生改变。矢量距离从运动始点或上一个 AV 指令开始计算：

指令	说明
#VECTOR	标号
VMAB;VS5000	联动轨迹
VPl0000,20000	矢量位置
VP20000,30000	矢量位置
VE	矢量结束
BGS	开始运动
AV5000	在矢量距离 5000 到达之后
VSl000	减速
EN	结束

7）带有等待的多重运动

本例通过在执行新的运动之前等待各个运动完成进行多重相对位移运动：

指令	说明
#MOVES	标号
PRl2000	距离
SP20000	速度

Acl00000	加速度
BGA	开始运动
ADl0000	等待 10000cts 距离到达
SP5000	新速度
AMA	等待运动完成
WT200	暂停 200ms
PR-10000	新位置
SP30000	新速度
Acl50000	新加速度
BGA	开始运动
EN	结束

8）用 AT 定义输出波形

下列程序在输出点 1 产生 10ms 高电平，40ms 低电平的脉冲列，每个循环每 50ms 重复一次。

指令	说明
#OUTPUT	程序标号
AT0	初始化时间参考点
SB1	输出点 1 置高
#LOOP	循环
Atl0	从参考点计时 10ms 之后
CB1	输出点 1 置低
AT-40	从参考点计时等待 40ms 并重新设置参考点
SB1	输出点 1 置高
JP#LOOP	循环
EN	结束

条件跳转

DMC21x3 具有条件跳转和条件跳转到子程序功能；分别对应指令 JP 和 JS，使程序根据所指定的条件跳转到新的分支程序中去执行。条件跳转确认条件是否满足，然后进入新位置或子程序。与事件触发不同，条件跳转指令不暂停程序。条件跳转功能对于实时检测事件很有用。此功能使 DMC21x3 不用主计算机干预即可做出判断。例如，DMC21x3 能以输入点状态为条件在两种运动轮廓之间做出选择。

指令格式——JP 和 JS

格式	说明
JS 目标，逻辑条件	如果逻辑条件满足，跳转到指定目标子程序
JP 目标，逻辑条件	如果逻辑条件满足，跳转到指定目标位置

目标是当指定条件满足时程序定序器将要跳转的程序行号或标号。注意：程序存储区的首行行号为 0。

逻辑算子

运算子	说明
<	小于
>	大于
=	等于
<=	小于等于
>=	大于等于

<>	不等于

条件语句

如果对不为 0 的任意值进行评估，条件语句被满足。条件语句可以是任何有效的 DMC21x3 数字操作数，包括变量、数组元素、数值、函数、键字、算术表达式等。如果不给出条件语句，

即为无条件跳转。

数字	V1=6
数字表达式	V1=V7*6
	@ABS[V1]>10
数组元素	Vl<Count[2]
变量	V1<V2
操作数	_TPA=0
	_TVA>500
I / O	V1>@AN[2]
	@AN[1]=0

多重条件语句

DMC21x3 可以在一个跳转语句中接受多个条件。使用“&”和“|”操作数就能把多个条件语相迭加，任意两个条件之间的“&”操作数要求两个语句都必须为真。任意两个条件之间的“|”操作数要求只要任何一个语句为真。

注意：必须把各个条件放在圆括弧内，以便控制器进行正确判断。此外，DMC21x3 从左到右执行运算。关于数字表达式及位操作算子‘&’和‘|’的详细介绍，请参见“数字表达式”一节。

例如，使用以 V1、V2、V3、V4 命名的变量：

JP#TEST,(V1<V2)&(V3<V4)

在此例中，若 V1 小于 V2 且 V3 小于 V4，程序就跳转到标号为#TEST。为了进一步说明，我们再看看下例：

JP#TEST,((V1<V2)&(V3<V4))|(V5<V6)

如果满足两个条件：1、如果 V1 小于 V2 且 V3 小于 V4，或者 2、(V5<V6)，程序就会跳转到标号#TEST。

举例

1）如果满足 JP 指令的条件，控制器就跳转到所指定的标号或行号，并从此点开始继续执行指令。如果条件不满足，控制器就按序继续执行下一条指令。

指令	说明
JP#LOOP,count<10	如果变量 count 小于 10，跳转到#LOOP
JS#MOVE2,@IN[1]=1	如果输入点 1 为逻辑高电平，跳转到子程序#MOVE2。在执行完#MOVE2 子程序之后，程序分序器返回到调用子程序处的主程序位置。
JP#BLUE,@ABS[V2]>2	如果变量 V2 的绝对值大于 2，跳转到#BLUE
JP#C,V1*V7<=V8*V2	如果 V1 值乘 V7 小于或等于 V8 乘 V2 的值，跳转到#C
JP#A	无条件跳转到#A

2）使 A 轴电机运动到绝对位置 1000cts，并返回到 0，重复 10 次，在每两次运动之间暂停 100ms。

指令	说明
#BEGIN	程序标号
Count=10	循环计数器置初值

#LOOP	循环程序标号
PA1000	绝对位置 1000
BGA	开始运动
AMA	等待运动完成
WTl00	暂停 l00ms
PA0	返回绝对位置 0 点
BGA	开始运动
AMA	等待运动完成
WTl00	等待 l00ms
Count=Count-1	循环计数器减 1
JP#LOOP,Count>0	通过循环测试 10 次
EN	程序结束

IF、ELSE、ENDIF

DMC21x3 提供了对使用 IF、ELSE、ENDIF 指令的条件语句的结构化方法。

使用 IF、ENDIF 指令

IF 条件语句由 IF 和 ENDIF 指令组合来构成，IF 指令具有一个或更多的条件语句作为其判断。如果条件语句判断为真，指令解释器就会继续执行跟随在 IF 指令之后的指令；如果条件语句判断为假，控制器就会忽略掉在 IF 指令之后的指令，去执行与之相配合的 ENDIF 指令或者去执行在程序中出现的 ELSE 指令（见以下相关 ELSE 指令介绍）。

注意：对于已执行的每个 IF 指令，总是必须执行一次 ENDIF 指令。建议用户不要在 IF 条件语句里包含跳转指令，否则会使程序执行不知去向。在此情况下，指令解释器就可能不执行 ENDIF 指令。

使用 ELSE 指令

ELSE 指令是 IF 条件语句的选择部分，只有当 IF 指令的参数判断为假时才执行 ELSE 指令。ELSE 指令必须出现在 IF 指令之后，且没有参数。如果用于 IF 指令的参数判断为假，控制器就会跳过后续指令去执行 ELSE 指令。如果用于 IF 指令的参数判断为真，控制器就会执行 IF 和 ELSE 指令之间的指令。

嵌套 IF 条件语句

DMC21x3 允许 IF 条件语句包含在其它 IF 语句之内，此技术称之为“嵌套”，DMC21x3 允许 255 个 IF 条件语句进行嵌套。这是一项功能非常实用的技术，使用户能对许多种不同的分支情况进行指定。

指令格式　IF　ELSE　ENDIF

格式	说明
IF 条件表达式	如果条件为真，就执行 IF 指令后（直到 ELSE 指令）的指令，否则，继续在 IF 指令或可选的 ELSE 指令处执行。
ELSE	可选择指令，当 IF 指令的参数判断为假时，转而执行此指令，只能与 IF 指令配合使用。
ENDIF	用来使 IF 条件语句结束的指令。对于每条 IF 指令，程序必须有一个 ENDIF 指令与之对应。

指令	说明
#TEST	开始主程序“TEST'’
Ⅱ,,3	输入点 1 和 2 时终端使能
MG”WAITING FOR INPUTl,INPUT2”	输出信息
#LOOP	无休止循环用标号
JP#LOOP	无休止循环

EN	主程序结束
#ININT	输入中断子程序
IF(@IN[1]=0)	以输入点 1 为基础的III条件语句
IF(@IN[2]=0)	若第 1 个 IF 条件为真，执行第 2 个 IF
MG”INPUTl AND INPUT2 ARE ACTIVE”	若第 2 个 IF 条件为真，执行信息发送
ELSE	用于第 2 个语句的 ELSE 指令
MG”ONLY INPUTl IS ACTIVE”	若第 2 个 IF 条件为假，执行信息发送
ENDIF	第 2 个条件语句结束
ELSE	用于第 1 个语句的 ELSE 指令
MG”ONLY INPUT2 IS ACTIVE”	若第 1 个 IF 条件为假，执行信息发送
ENDIF	第 1 个条件语句结束
#WAIT	用于循环的标号
JP#WAIT,(@IN[1]=0)| (@IN[2]=0)	循环直到输入点 1 和 2 均无效
RI0	中断返回

子程序

所谓子程序就是一组以标号开始，且以 EN 指令结束的指令。子程序从主程序中用跳转子程序指令 JS，后跟标号或行号和条件语句进行调用。可以嵌套 8 个子程序，在执行完子程序之后，程序返回到调用子程序的地方。但子程序堆栈被变换的情况除外。

例如画出边长为 500cts 正方形的子程序举例如下，正方形以矢量位置 1000，1000 处开始绘制。

指令	说明
#M	主程序标号
CB1	清除输出点 1（抬笔）
VP1000,1000;LE;BGS	定义矢量位置：移动笔
AMS	等待运动完成
SB1	输出点 1 置高电平（下笔）
JS#SQUARE;CB1	跳到 SQUARE 子程序
EN	结束子程序
#SQUARE	SQUARE 子程序
V1=500;JS#L	定义边长
V1=-V1;JS#L	切换方向
EN	结束子程序
#L;PRV1,V1;BGA	定义 A、B 轴位置，开始 A 轴运动
AMA;BGB;AMB	在 A 轴运动完成后，开始 B 轴运动
EN	结束子程序

堆栈变换

可以用 ZS 指令使子程序堆栈进行变换。每次 JS 指令、执行一次中断或自动子程序（如#POSERR 或#LIMSWI），子程序堆栈指针就加 1。通常是 EN 指令结束子程序时使堆栈减 1。偶尔，希望不返回到子程序或中断被调用的程序行。ZS1 指令使堆栈指针减 1，这样就能使程序定序器继续执行下一行。ZS0 指令把堆栈复位到它的初始值。例如，如果限位发生，执行#LIMSWI 子程序，往往希望重新开始程序顺序，而取代返回到限位发生的地方。要满足这种要求，就在#LIMSWI 子程序的末尾加上 ZS0 指令。

自动启动子程序

DMC 控制器有一个特殊标号用于自动程序执行。已经存入控制器非易失存储器中以标号#AUTO 开始的程序在上电或复位时能够自动被运行。此程序必须用 BP 指令存入非易失性存储器中。

用于监测条件的自动子程序

经常需要不用主机或 DMC21x3 程序介入而连续监测某些条件，DMC21x3 能够在后台监测几个重要条件。这些条件有：检查限位开关、定义的输入、位置误差或指令错误等。在应用程序中插入特殊、预先定义的标号使自动监测功能生效。

预先定义的自动子程序标号有

子程序标号	说明
#LIMSWI	任意轴限位开关有效
#ININT	由 II 所规定的输入有效
#POSERR	超出 ER 指定的误差极限
#MCTIME	定位超时，定位时间由指令 TW 设定
#CMDERR	指令错误
#COMINT	通讯中断
#TCPERR	TCP/IP 通讯错误
#AMPERR	驱动器错误
#AUTOERR	执行#AUTO 程序错误

例如，当任意一个轴超过位置误差上限时，就会自动执行#POSERR 子程序。#POSERR 子程序中的指令检查出哪个轴处于报警中并采取恰当行动。在另一个例子中，用#ININT 标号来指向输入中断子程序。当指定的输入出现时，就会执行该程序。

举例

1）限位开关

限位开关一出现，此程序就打印信息。注意，对于#LIMSWI 子程序来说，DMC21x3 应该正在执行着来自存储器的应用程序。这是一个非常简单的程序，不做任何事但只循环一个语句，例如#LOOP：JP#LOOP；EN。即使在执行“虚拟”应用程序当中，仍然可以从 PC 发送像 JP5000 那样的运动指令。

指令	说明
：ED	编辑方式
000#LOOP	虚拟程序
001JP#LOOP：EN	跳转到循环程序
002#LIMSWI	限位开关标号
003MG”LIMITOCCURRED”	打印信息
004RE	返回主程序
<control>Q	退出编辑方式
：XQ#LOOP	执行虚拟程序
：JG5000	Jog 运动
：BGA	开始运动

现在，当 A 轴正向限位开关有效时，就会执行#LIMSWI。

几点注意事项：

1）用 RE 指令从#LIMSWI 于程序返回

2）如果限位开关保持有效状态，就会再次执行 LIMSWI 子程序

3）当正在指令电机运动时，就只执行#LIMSWI 子程序。

2）位置误差

指令	说明
：ED	编辑方式
000#LOOP	虚拟程序
001JP#LOOP;EN	循环
002#POSERR	位置误差子程序
003V1=_TEX	读位置误差
004MG"EXCESS POSITION ERROR"	打印信息
005MG"ERROR=",V1=	打印误差
006RE	从误差子程序返回
<control>Q	退出编辑方式
：XQ#LOOP	执行虚拟程序
：JG100000	以高速进行 Jog 运动
：BGX	开始运动

3）输入中断

指令	说明
#A	标号
II1	由输入点 1 产生输入中断
JG30000,,,60000	Jog 运动
BGAD	开始运动
#LOOP;JP#LOOP;EN	循环
#ININT	输入中断
STXW;AM	停止运动
#TEST;JP#TEST,@IN[1]=0	若输入点 1 仍为低电子，循环测试
JG3000,,,6000	Jog 运动
BGAD	开始运动
RI0	从中断程序返回到主程序，并不对条件启动重新使能

4）运动完成超时

指令	说明
#BEGIN	开始主程序
TWl000	设置超时基准 1000ms
PAl0000	绝对位移 10000
BGA	开始运动
MCA	运动完成条件启动
EN	结束主程序
#MCTIME	运动完成子程序
MG”A fell short”	发送信息
EN	结束程序

如果 A 轴在运动结束后的 1 秒之内没有到达指令位置，此程序就会送出信息“A fell short”。

5）指令错误

指令	说明
#BEGIN	开始主程序
IN”ENTERSPEED”，SPEED	提示输入速度
JG SPEED;BGA	开始运动
JP#BEGIN	重复循环

EN	结束主程序
#CMDERR	指令错误子程序
JP#DONE,_ED<>2	检查第 2 行是否出错
ZS1	调整堆栈
JP#BEGIN	返回到主程序
#DONE	如果有其它错误，结束程序
ZS0	0 堆栈
EN	结束程序

上述程序提示操作人员输入 JOG 速度值，如果操作人员输入的数值超出范围，大于 12M（模拟输出伺服）或 3M（脉冲），通过执行#CMDERR 子程序，提示操作人员输入一个新速度值。

在多任务应用中，有另外的方法用于处理不同通道中的指令错误。下表所列的特殊操作数与 XQ 指令配合使用就能使控制器跳过或重新处理无效指令。

特殊操作数

操作数	说明
_ED1	产生错误的通道号
_ED2	无效指令所在位置
_ED3	无效指令之后的指令所在位置

以下列格式与 XQ 指令一起使用操作数：

XQ_ED2（或_ED3）,_ED1,1

其中：在指令行末尾的“,1”表示重新开始；因此，当执行上面的格式时，不会去掉已有的程序堆栈。

下例示出使用操作数的错误修正子程序。

6）指令错误（使用多任务时）

指令	说明
#A	开始通道 0（连续循环）
JP#A	
EN	通道 0 结束
#B	开始通道 1
N=-1	建立新变量
KPN=n	设置 KP 为 N 的值，此为无效值
TY	发布无效指令
EN	通道 1 结束
#CMDERR	开始指令错误子程序
IF_TC=6	如果错误，误差超出范围(KP-1)
N=1	设置 N 为有效数值
XQ_ED2，_ED1，1	重试 KPN 指令
ENDIF	
IF_TC=1	如果错误属无效指令(TY)
XQ_ED3,_ED1，1	跳过无效指令
ENDIF	
EN	指令错误子程序结束

7）Ethernet 通信错误

在 DMC-21x3 中运行这个简单程序并指出（通过串口）通信端口何时失败。通过监控串口，若有必要，用户能够重新建立通信。

指令	说明

#LOOP	简单程序循环
JP#LOOP	
EN	
#TCPERR	Ethernet 通信错误自动子程序
MG{P1}_IA4	将信息发送到串口，指明哪个端口没有接收正确的响应。
RE	

数学及函数表达式

数学运算符

为了进行数据运算变换，DMC21x3 提供如下数学运算功能

运算符	功能
+	加
-	减
*	乘
/	除
&	与
|	或
()	括弧，用于改变运算顺序，可以嵌套

加、减、乘的运算数值范围为±2147483647.9999，除法运算的精确度为 1/65000。

注意：数学表达式的运算完全遵循从左向右的顺序，只有圆括弧可以改变这个顺序。GALIL 并不遵守通常先乘除后加减的运算优先级顺序。

SPEED=7.5*V1/2	变量 SPEED 等于 V1 乘以 7.5 再除以 2
COUNT=COUNT+2	变量 COUNT 等于当前值加 2
RESULT=_TPA-(@COS[45]*40)	把 A 轴位置值减去 28.28 放入变量 RESULT。40*COS45°为 28.28
TEMP=@IN[1]&@IN[2]	只有当输入点 1 和输入点 2 均为高电平时，变量 TEMP 等于 1

位运算符

数学算子&和丨是位运算符。运算符&是逻辑与，运算符丨是逻辑或。这些运算符对任意有效的 DMC21x3 数字操作数（包括变量、数组元素、数字值、函数、键字和算术表达式）进行位运算。位运算符也可以与字符串一起使用。这对于把输入字符串拆分字符来说非常有用。采用输入指令进行字符串输入时，输入变量会拥有多达 6 个字符。这些字符合并成一个单值，表示为 32bit 整数和 16bit 小数。每个 ASCII 字符表达为 1 个字节（8bit），因此，输入变量就拥有 6 个字符。字符串的第一个字符置于变量的最高字节，而最后一个字符放置于小数点后的最低有效字节。

能通过下例所示的位操作将字符逐个分开。

指令	说明
#TEST	开始主程序
IN“ENTER”，LEN{S6}	将 6 个字符的字符串放入变量“LEN”
FLEN=@FRAC[LEN]	将变量“FLEN'’定义为变量“LEN”的小数部分

```
FLEN=$10000*FLEN                    FLEN 左移 32 位 bits(将小数部分 FLEN 转换
                                    成整数))
LEN1=(FLEN&$00FF)                   屏蔽 FLEN 最高字节并将此值设置成变量
                                    “LEN1”
LEN2=(FLEN&$FF00)/$l00              让变量“LEN2”=FLEN 的最高字节
LEN3=LEN&$FF                        让变量“LEN3”=LEN 的最低字节
LEN4=(LEN&$FF00)/$l00               让变量“LEN4”=LEN 的第 2 字节
LEN5=(LEN&$FF0000)/$10000           让变量“LEN5”=LEN 的第 3 字节
LEN6=(LEN&$FF000000)/$1000000       让变量“LEN6”=LEN 的第 4 字节
MG LEN6{S4}                         显示“LEN6”为 4 字符的字符串信息
MG LEN5{S4}                         显示“LEN5”为 4 字符的字符串信息
MG LEN4{S4}                         显示“LEN4”为 4 字符的字符串信息
MG LEN3{S4}                         显示“LEN3”为 4 字符的字符串信息
MG LEN2{S4}                         显示“LEN2”为 4 字符的字符串信息
MG LENl{S4}                         显示“LENl”为 4 字符的字符串信息
EN
```

此程序将接受 6 个字符的字符串输入，解析各个字符，然后显示各字符。也要注意，用于屏蔽的值以十六进制表示（由引导号符$标注），对于更多的相关信息，参见“发送信息”一节。

为了进一步说明，如果用户在输入提示处输入字符串“TESTME”控制器就会响应如下：

```
T       来自指令 MG LEN6{S4}的响应
E       来自指令 MG LENS{S4}的响应
S       来自指令 MG LEN4{S4}的响应
T       来自指令 MG LEN3{S4}的响应
M       来自指令 MG LEN2{S4}的响应
E       来自指令 MG LENl{S4}的响应
```

函数

函数	说明
@SIN[n]注	正弦
@COS[n]注	余弦
@TAN[n]注	正切
@ASIN[n]	反正弦，返回值在±90°范围内，分辨率为 1/64000
@ACOS[n]	反余弦，返回值在 0°~180°范围内，分辨率为 1/64000
@ATAN[n]	反正切，返回值在±90°范围内，分辨率为 1/64000
@COM[n]	补码
@ABS[n]	绝对值
@FRAC[n]	取小数部分
@INT[n]	取整数部分
@RND[n]	四舍五入取整
@SQR[N]	平方根
@IN[n]	通用输入点的状态（n 从 1 开始）
@OUT[n]	通用输出点的状态（n 从 1 开始）
@AN[n]	通用模拟输入点的输入值（n 从 1 开始）

注：n 以角度为单位，分辨率为 1/64000 度，

函数可以与数学表达式结合使用，数学表达式的执行次序是从左到右，且能够用圆括号相括而优先执行。

指令	说明
V1=@ABS[V7]	变量 V1 等于变量 V7 的绝对值.
V2=5*@SIN[POS]	变量 V2 等于 5 倍变量 POS 的正弦
V3= @IN[1]	变量 V3 等于输入点 1 的数字值
V4=2*(5+@AN[5])	变量 V4 等于模拟输入点 5 的值加 5，然后乘以 2

变量

对于要求参数为变量的应用，DMC21x3 提供了 254 个变量，这些变量可以是数值或字符串。用户可以把某些参数（如位置或速度等）定义为变量，以变量形式写入程序；其后，再由操作人员分配变量值或者通过程序计算确定变量值。例如，定长裁断应用就需要切长为变量。

指令	说明
PR POSA	把变量 POSA 分配给 PR 指令
JG RPMB*70	把变量 RPMB 乘以 70 分配给 JG 指令

变量名

DMC21x3 允许用户创建 254 个变量，各变量可以用 8 个字符构成的变量名加以定义。变量名必须以英文字母开头，不过，在变量名的其余部分允许用数字，但不允许有空格。变量可以是大写或小写，或者任意组合。变量名对大小写敏感，SPEEDC 与 SpeedC 是两个不同的变量。变量名不应该与 DMC21x3 指令相同。例如，PR 就不适合用作变量名。

有效变量名：

POSA
POSl
SPEEDC

无效变量名

REALLONGNAME	不能多于 8 个字符
123	不能以数字开始
SPEED C	变量名中不能有空格

变量赋值

赋给变量的值可以是数字、其它变量、操作数、函数、控制参数和字符串；数字变量值的范围是 4 字节整数后跟 2 字节小数（±2,147,483,647.9999）。

可以用等号把数值分配给变量，可以用任意有效函数把值分配给变量。例如，V1=@ABS[V2]或 V2=@IN[1]。也允许有算术运算。

要分配字符串值，字符串必须在引号内。字符串变量可以含有 6 个字母，这些字母必须在引号中。

指令	说明
POSA=_TPA	把从 TPA 指令返回的值分配给变量 POSA
SPEED=5.75	把 5.75 分配给变量 SPEED
INPUT=@IN[2]	把输入点 2 的逻辑值分配给变量 INPUT
V2=V1+V3*V4	将（V1+V3）*V4 的值分配给变量 V2，
VAR=”CAT”	将字符串 CAT 分配给 VAR

将变量值赋给控制器参数

可以将变量值赋给控制器参数（如 GN 或 PR）

PR V1	将 V1 分配给 PR 指令
SP _VSS*2000	将_VSS*2000 分配给 SP 指令

在终端上显示变量值

可以用格式：变量=，将变量发送到显示屏。例如，V1=就会返回变量 V1 的值。

变量应用举例

将变量用于操纵杆（Joystick）下例读取 A-B 轴操纵棒的电压值，并将它分配给变量 VA 和 VB，以比例速度驱动电机运动，其中：

10V=3000rpm=200000cts / sec

速度 / 模拟输入(比例速度)=200000 / 10=20000

指令	说明
#JOYSTIK	标号
JG0,0	设定 JOG 方式
BGAB	开始运动
#LOOP	循环
VA=@AN[1]*20000	读取操纵棒 A
VB=@AN[2]*20000	读取操纵棒 B
JG VA,VB	以变量 VA,VB 进行 JOG 运动
JP#LOOP	重复循环
EN	结束

操作数

操作数允许把 DMC 控制器运动或状态参数并入可变量和表达式。大多数 DMC21x3 指令都有一个等效的操作数，通过在 DMC21x3 指令之前添加下划线即指定为操作数。《指令手册》中明确了哪些指令有相应的操作数。

状态指令，如 TP 读回实际位置值，而设置指令，如 KP 或 SP 就读回 DMC 寄存器中的值；跟随指令之后需要指定轴。

指令	说明
POSA=_TPA	将来自 A 轴实际位置寄存器的值分配给变量 POSA.
JP#LOOP,_TEA>5	如果 A 轴位置误差寄存器中的值大于 5，就跳转到#LOOP
JP#ERROR,_TC=1	如果错误代码等于 1，就跳转到#ERROR

可以在表达式中使用操作数，并将其赋值给变量，但不能对其赋值，例如：_TPA=2 是无效的。

特殊操作数（关键字）

DMC21x3 提供了一些附加操作数，用于存取那些不能用标准 DMC21x3 指令读取的内部状态。

关键字	说明
_BGn	如 n 轴运动完成,返回 1,否则为 0
_BN	返回控制卡的序号
_DA	反回数组变量数

_DL	返回可用标号数
_DM	返回可用的数组空间大小
_HMn	返回原点开关状态
_LFn	返回前极限开关状态
_LRn	返回后极限开关状态
_UL	返回可用变量数
TIME	返回上电后执行的伺服周期数

数组

为了存储、采集数据，DMC21x3 提供了 8000 个元素的数组空间。数组是一维的，可以定义 30 个不同数组。各个数组元素都有一个 4 字节整数，后跟 2 字节小数的数值范围（±2147483647.9999），可以用数组来捕获实时数据，如位置、转矩和模拟输入值。在轨迹方式下，在记录和录返应用中，可以很方便地用数组来保存位置轨迹的各点坐标值。

定义数组

用指令 DM 定义数组，用户必须指定要放入的数组名和入口号。数组名可以含有 8 个字符，以大写字母开头，再定义数组中的元素数，元素数在[]中。

DM POSA[7]	定义带有 7 个元素的数组名 POSA
DM SPEED[l00]	定义带有 100 个元素的数组名 SPEED
DM POSA[0]	清空数组空间

数组赋值

像变量一样，可以给各数组元素赋值。分配的值可以是数值或从指令、函数、关键字来的返回值。

数组元素地址从 0 开始，例如，POSA 数组中的第 1 个元素（用 DMPOSA[7]定义）就指定为 POSA[0]。

注意：在给数组赋值之前，必须用指令 DM 定义数组。

DM SPEED[10]	定义数组 Speed 大小
SPEED[1]=7650.2	给数组的第 1 个元素赋值 7650.2
SPEED[1]=	读回数组第一个元素的值
POSXA[10]=_TPA	给第 10 个元素赋值 A 轴实际位置值
CON[2]=@COS[POS]*2	给第 2 个数组元素赋值为 POS 的余弦值乘以 2
TIMER[1]=TIME	给第 1 个数组元素赋值为 TIME 值

用变量给数组元素分配地址

数组元素号也可以是一个变量。这样，就可以用一个计数器按序给数组元素赋值。

指令	说明
#A	程序标号
COUNT=0;DMPOS[10]	初始化计数器并定义数组
#LOOP	开始循环
WTl0	等待 10msec
POS[COUNT]=_TPA	将位置值记录入数组元素
POS[COUNT]=	报告位置
COUNT=COUNT+1	计数器加 1

JP#LOOP,COUNT<l0　　循环，直到保存完 10 个元素

EN　　结束

上例表示，以每 l0ms 采集一个数值的速率将 10 个位置值存入 10 个数组元素，这 10 个值存入数组名 POS 中，变量 count 用来使数组元素计数器增 1。上例也可用下述的自动数据采集功能来实现。

上载、下载数组到板上存储器

也可以用 QU 和 QD 指令上载和下载数组。

QUarray[]，start，end，delim

QDarray[]，start，end

其中：array 是数组名，例如：A[]；start 是第一个数组元素（缺省值=0）；end 是最后一个数组元素（缺省值；最后一个元素）；delim 定义数组数据用逗号“,”（delim=1）还是用回车（delim=0）来分开。用<control>Z，<control>Q，<control>D 或＼来终止文件。

将数据自动采集到数组中

DMC21x3 控制器有一个用于数据自动采集的特殊功能，可以用于对诸如位置、位置误差、输入、转矩等进行自动采集，并存入数组。这对于示教运动轨迹或观察系统性能而言非常有用。可以同时采集 4 种数据，并存入 4 个数组。可以指定采集速率或时间间隔，可以按一次事件或一个循环连续记录进行记录。

指令汇总——自动数据采集

指令	说明
RA n[],m[],o[],p[]	为数据采集选择 4 个数组，必须用 DM 指令定义数组
RD tp1,tp2,tp3,tp4	选择要记录的数据类型，其中：tpl，tp2，tp3，tp4 代表各种数据类型（见下表），数据类型的次序很重要，与 RA 指令中 n，m，o，p 数组的次序相对应。
RC n,m	RC 指令开始数据采集，设置数据捕获时间间隔，其中：n 是 1-8 的整数，指定数据之间相隔 2^nms；m 是选择项，规定要被捕获的元素数量：若不定义 m，元素数量就缺省为由 DM 指令定义的最小数组。当 m 是负数时，就以循环方式连续进行记录。_RD 是记录指针，指示下一个数组元素的地址。n=0 则停止记录。
RC?	返回值为 0 或 1，其中：0 表明不记录，1 表示记录在进行。

用于记录的数据

数据类型	说明
_DEA	辅编码器位置或发送脉冲位置
_TPA	主编码器位置
_TEA	位置误差
_SHA	规划位置
_RLA	缩存的位置
_TI	输入点
_OP	输出点
_TSA	开关点
_SCA	停止代码
_NOA	状态位
_TTA	输出模拟量（±10V 对应数据±32544）
_AFA	模拟输入值

操作数汇总

操作数	说明
_RC	返回值为 0 或 1；0 表示不记录，1 表示记录在进行
_RD	返回值为下一个数组元素的地址

举例

指令	说明
#RECORD	标号
DM APOS[300],BPOS[300]	定义 A、B 轴位置数组注
DM AERR[300],BERR[300]	定义 A、B 轴误差数组
RA APOS[1],AERR[],BPOS[],BERR[]	为捕获选择数组
RD _TPA,_TEA,_TPB,_TEB	选择数据类型
PR l0000,20000	指定运动距离
RC1	以 2msec 速率开始记录
BGAB	开始运动
#A:JP#A,RC=1	循环
MG”DONE”	发送信息
EN	结束程序
#PLAY	录返程序标号
N=O	初始化计数器
JP#DONE，N>300	如果完成，退出
N=	打印计数器值
APOS[N]=	打印 A 轴位置
BPOS[N]=	打印 B 轴位置
AERR[N]=	打印 A 轴误差
BERRtN]=	打印 B 轴误差
N=N+1	计数器加 1
#DONE	标号
EN	结束程序

取消数组空间分配

用 DA 指令后跟数组名就可以取消数组空间分配，DA*[0]将所有数组分配取消。

数据输入(数字和字符串)

用指令 IN 来提示用户输入数字或字符串，使用 IN 指令，用户可以把信息放在双引号中，指定信息提示内容。控制器执行 IN 指令时，就会等待输入数据。把输入分配给所指定的变量或数组元素。

1）数据输入

```
#A
IN "Enter Length",LenA
EN
```

在此例中，信息“Enter Length”显示在计算机屏幕上，控制器等待操作人员输入一个长度值，操作人员输入分配给变量 LenA 的值。（注意：在输入信息末尾的逗号和变量名之间不能有空格）。

2）定长裁断

在此例中，材料的长度要事先规定好，当运动完成时，使刀头工作，切断材料。长度值

是一个变量，提示操作者以英寸输入。切割运动由连接到输入点 1 的启动按钮启动。负载与螺距为 2 英寸的丝杠相连接，2000cts/rev 编码器安装在电机上，分辨率为 4000cts/英寸。下面的程序对长度用变量 LEN，用 IN 指令来提示操作人员输入长度值，然后输入的长度值就分配给变量 LEN。

指令	说明
#BEGIN	标号
AC 800000	加速度
DC 800000	减速度
SP5000	速度
LEN=3.4	初始长度
#CUT	Cut 子程序
AI1	等待启动信号
IN”enter Length”,LEN	提示操作人员以英寸为单位输入长度值
PR LEN*4000	以 cts 指定位置值
BGX	开始运动移动材料
AMX	等待运动完成
SB1	对切刀设置输出，使切刀动作
WTl00;CB1	等待 100msec，然后关断切刀
JP#CUT	重复切割子程序
EN	结束

数据输出(数字和字符串)

可以用几种方法从控制器输出数字和字符串数据。信息指令 MG 能输出字符串和数字数据。还有，能用查询指令对控制器进行指令，以返回变量和数组值以及其它信息。详见第 5 章所描述的查询指令。

发送信息

用信息指令 MG 可以将信息发送到总线，此指令将指定的文本、数字或字符串数据从变量或数组发送到屏幕。

用双引号指定文本字符串，而用变量或数组名指定变量或数组数据。例如：

MG”TheFinalValueis”,RESULT

除变量、函数和指令之外，在信息指令中还可以使用操作数。例如：

MG”Analoginputis”,@AN[1]

MG”The Position Of Ais”,_TPA

格式化信息

用{sn}对字符串变量进行格式化，其中，n 是字符数，1-6。例如：

MG STR {S3}

这条语句返回字符串变量名为 STR 的 3 个字符。用{Fn.m}表达式后跟完整的 MG 语句就可以对数字数据进行格式化。{Sn.m}以十六进制而不是十进制对数据进行格式化，将实际数值格式化成小数点左面 n 个字符，小数点右面 m 个字符。有效值前面的 0 用来显示指定的格式。

例如：

MG”The Final Value is”,RESULT{F5.2}

如果变量 RESULT 的值等于 4.1，上述语句返回如下：

The Final Value is 00004.10

如果变量 RESULT 的值等于 999999.999，上述语句返回如下：

The Final Value is 99999．99

信息指令通常发送跟着语句的回车和换行。在语句末尾发送{N}就可以阻止回车和换行。当文本字符串需要处于数值前后时，这就很有用。

例如：

```
#A
JG 50000：BGA；ASA
MG”TheSpeedis”，_TVA{F5．1}{N}
MG”counts／sec”
EN
```

当执行上述程序时，屏幕上就会显示出：

The speed is 50000 counts／sec

用 MG 指令配置终端

能用 MG 指令对终端进行配置。可以用格式{^n}发送任意 ASCII 字符，

其中：n 是 1~255 之间的任意整数。

例如：

MG{^07}{^255}

将用 7 和 255 表示的 ASCII 字符发送到总线。

12．1．4 信息函数格式化字符汇总

格式化字符	说明
“”	输出文本字符
{Fn.m}	格式化数值，小数点左面 n 位十进制数，右面 m 位
{P1}或{En}	发送到串口或网口
{$n.m}	以 16 进制对数值进行格式化
{^n}	发送 ASCII 码为 n 的字符
{N}	阻止回车/换行
{Sn}	发送字符串的前 n 个字符

显示变量和数组

可以用格式，变量= 或数组[x]=．将变量或数组发送到屏幕上。例如，V1=就返回 V1 的值。

举例——显示变量和数组元素

```
指令        说明
#DISPLAY        标号
DM POSA[7]      定义有 7 个元素的数组 POSA
PR l000         位置指令
BGX             开始运动
AMX             在运动完成后
V1=_TPA         变量 V1 赋值
POSA[1]=_TPA    第 1 个元素赋值
V1=             显示 V1
```

查询指令

DMC21x3 有一组直接查询控制器的指令。当输入这些指令时，就以十进制返回所要的数据，通过回车和换行而进入下一行。返回的数据格式能用位置格式化指令（PF）和前位充 0

指令（LZ）进行修改。关于查询指令的完整内容参见第 5 章。

用 PF 指令对查询指令所产生的回应进行格式化

PF 指令能修改由查询指令所返回值的格式：

BL?	LE?
DE?	PA?
DP?	PR?
EM?	TN?
FL?	VE?
IP?	TE
TP	

用 PF 指令可以对十进制或十六进制数值格式化，使其该数的小数点右面和左面具有所规定的位数。指令格式如下：

PF m．n

其中：m 是小数点左面的位数（0-10），n 是小数点右面的位数（0-4），负的 m 则指定为十六进制格式。

返回十六进制值由$开头，以 2 的补码表示。十六进制值应该按带符号 2 的补码进行输入，其中，负数有负号。缺省格式为 PFl0．0。如果由 PF 所规定的十进制位数小于实际值，那么 9 就出现在所有的十进制位。

举例：

指令	说明
：DP21	定义位置
：TPA	报告位置
0000000021	缺省格式
：PF4	修改格式为 4 位
：TPA	报告位置
0021	新格式
：PF-4	修改成十六进制格式
：TPA	报告位置
$0015	十六进制值
：PF2	修改格式为 2 位
：TPA	报告位置
99	如果位置值大于 99，就返回 99

从查询指令响应中去掉前面的 0

可以用指令 LZ 去掉由查询指令所返回值前面的 0。

指令	说明
LZ0	使 LZ 功能无效
TP	报告位置
-0000000009,0000000005	返回值(带有前 0)
LZ1	使 LZ 功能使能
TP	报告位置
-9,5	返回值（无前 0）

查询指令返回值的局部格式化

可以对查询指令返回值进行局部格式化。要想进行局部格式化，在查询指令的同一行使用指令{Fn.m}或{$n..m}。符号 F 指出应该以十进制格式返回响应，而$指定以十六进制返回。n 是小数点左面的位数，m 是小数点右面的位数。

指令	说明

TP{F2．2}	以十进制格式 2.2 报告位置
-05.00,05.00,00.00,07.00	查询所得结果
TP{S4.2}	以十六进制格式 4．2 报告位置
FFFB.00,$0005.00,$0000.00,$0007.00	查询所得结果

对变量和数组元素格式化

用变量格式化(VF)指令对变量和数组元素进行格式化。

VF 指令格式如下：

VFm.n

其中：m 是小数点左侧位数（0-10），n 是小数点右侧位数（0-4）。m 的负号表示为十六进制格式。VF 的缺省格式为 VF10.4。

十六进制返回值前缀 S，以 2 的补码表示。

指令	说明
V1=10	V1 赋值
V1=	返回 V1
：0000000010.0000	查得结果—缺省格式
VF2.2	更改格式
V1=	返回 V1
：10.00	查得结果——新格式
VF-2.2	指定十六进制格式
VI=	返回 V1 值
$0A.00	查得结果—十六进制值
VF1	修改格式
V1=	返回 V1 值
：9	查得结果——溢出

变量的局部格式化

PF 和 VF 指令是广域格式化指令，它对所有相关返回值和变量的格式起作用。也可以对变量进行局部格式化。要想进行局部格式化，就使用指令{Fn.m}或{Sn.m}后跟变量名和“=”。F 规定十进制，$规定十六进制。n 是小数点左侧位数，m 是小数点右侧的位数。

指令	说明
V1=10	V1 赋值
V1=	返回 V1 值
：0000000010.0000	缺省格式
V1={F4.2}	规定局部格式
：0010.00	新格式值
V1={S4.2}	规定十六进制局部格式
：$00A0.00	十六进制值
V1=“ALPHA”	V1 赋值为“ALPHA”
V1={S4}	指定字符串为前 4 个字符
：ALPH	

局部格式也可用于 MG 指令。

硬件 I/O

数字输出

DMC21x3 有 8 点通用数字输出口和 40 点扩展数字 I/O（使用扩展选件 DB-28040）用于控制外部事件。用 CO 指令可以将扩展 I/O 配置成输入或输出，DMC-2152/DMC2153~DMC-2182/DMC2183 还有 8 个附加输出点。用软件指令 SB（置位）、CB（位清除）或 OB（输出位）对输出口的各位进行置位或清除操作。

1）置位和清除位

指令	说明
SB6	将输出口 6 置于高电平
CB4	将输出口 4 清除为低电平

2）输出位

输出位（OB）指令常根据变量、数组、输入或表达式的值用于设置或清除输出。任何非 0 值都会使输出置位。

指令	说明
OB1,POS	如果变量 POS 不为 0，则对输出点 1 置位，如果 POS 等于 0，则清除输出点 1。
OB2,@IN[l]	如果输入点 1 为高电平，则对输出点 2 置位，如果输入点 1 为低电子，则清除输出点 2
OB3,@IN[1]&@IN[2]	如果输入点 1 和输入点 2 均为高电子，则对输出点 3 置位
OB4,COUNT[1]	如果数组 COUNT 中的元素 1 不为 0，则对输出点 4 置位

通过用指令 OP（输出口）指定 8 位字来设置输出口。这条指令使用户能用单条指令对整个 8 位输出口的状态进行定义，其中，2^0 是输出点 1，2^1 是输出点 2，以此类推。1 表示输出点接通。

常常用输出口来设置运动进行期间各继电器状态或外部开关和事件

3）在运动之后接通输出

指令	说明
#OUTPUT	标号
PR2000	位置指令
BG	开始
AM	运动之后
SB1	输出点 1 置高电平
WT1000	等待 1000msec
CB1	输出点 1 置低
EN	结束

数字输入

用@IN[n]函数或 TI 指令可以对通用数字输入点状态进行读取。@IN[n]函数返回所指定输入点 n 的逻辑电平，其中，n 是 1-96。

1）用输入点来控制程序流程

指令	说明
JP#A,@IN[1]=0	如果输入点 1 为低，则跳转到#A
JP#B,@IN[2]=1	如果输入点 2 为高，则跳转到#B
AI7	等待直到输入点 7 为高

AI-6	等待直到输入点 6 为低

2）由开关按钮启动运动

当用户将面板开关按为 ON 时，A 轴电机必须以 4000cts/sec 速度运转，当面板开关转换到 OFF 时，A 轴电机必须停止运转。

方案：将面板开关接到 DMC-21x3 的输入点 1，输入点 1 为高电平意味着开关处于 ON 位置。

指令	说明
#S;JG4000	设置速度
AI1;BGX	在输入点 1 变高后开始运动
AI-1;STX	在输入点 1 变低后停止运动
AMX;JP#S	运动完成后，重复循环
EN	

辅助编码器用作通用输入

辅助编码器输入口可以用作为通用输入，对于各轴来说，控制器都有一个辅助编码器接口，每个辅助编码器口由两个输入 CHA 和 CHB 组成。辅助编码器输入地址分配为输入点 81-96。

来自于辅助编码器的各路输入接口是差分线接收器，能够承受±12V 之间的电压。输入已被配置成接受 TTL 电平信号。要连接 TTL 电平信号，只需简单地把信号连接到正输入端，各输入端不要连接。对于其它信号电子，应该把负输入端连接到 1 / 2 全程电压范围（例如，如果信号是 0-12V 逻辑，那么就把负输入端连到 6V）。

例如：DMC2113 有一个辅助编码器接口，此接口有两个输入(CHA 和 CHB)。给 CHA 输入分配输入点 81，CHB 输入分配输入点 82。要将此输入用作为 2 个 TTL 信号，第一个信号连接到 AA+，第二个信号连接到 AB+，AA-、AB-不要连接。要读取这个输入，用函数@IN[81]和@IN[82]即可。

输入中断功能

DMC21x3 提供有输入中断功能，此功能会使程序自动执行跟在#ININT 标号之后的指令。用 II m,n,o 指令使此功能有效。m 代表起始输入，n 代表范围内的最后输入，参数 o 是中断屏蔽。如果不用 m 和 n，o 就表示屏蔽号。1 表示对中断进行使能的输入，其中，2^0 是 bitl，2^1 是 bit2，以此类推。例如，II,,5 使输入点 1 和 3（$2^0+2^2=5$）有效。

任何一个指定输入点的低电平输入都会使#ININT 子程序自动执行。用中断返回指令（RI）使程序运行从这个子程序返回到中断发生的地方。如果在执行#ININT 子程序之后，要想返回到程序中的其它地方，就用零堆栈（ZS）指令后面跟着无条件跳转语句。

重要提示：要从#ININT 子程序返回，使用 RI 指令（而不是 EN）

举例——输入中断

指令	说明
#A	标号#A
II1	由输入点 1 产生中断
JG30000,-20000	设定 A、B 轴速度
BG AB	开始 A、B 轴运动
#B	标号#B
TP AB	报告 A、B 轴位置
WT 1000	等待 1000ms
JP#B	跳转到#B
EN	程序结束

#ININT	中断子程序
MG"Interrupt has occurred"	显示信息
ST AB	停止 A、B 轴运动
#LOOP;JP#LOOP,@IN[1]=0	循环直至中断清除
JG 15000,10000	指定新速度
WT 300	等待 300ms
BG AB	开始 A、B 轴运动
RI	中断返回

模拟输入

DMC21x3 在使用了 DB-28040 扩展 I/O 选件后，提供 8 路模拟量输入。用指令@AN[n]函数可以读取这些点的电压值，其中：n 代表模拟输入点 1~8。AD 转换的分辨率是 12bit（16bitADC 为可选功能）。用模拟输入来用来读取特殊传感器的值，如温度、张力或压力等。

下例示出使电机跟随模拟信号运动的程序，第 1 个例子是点到点(PTP)运动，第 2 个例子表示连续运动。

1）位置跟随器（点——点）

目的：电机必须跟随模拟信号运动，当模拟信号变化到 1OV 时，电机必须移动 10000cts。

方法：读取模拟输入值，并指令 A 轴移动到 10000cts 处。

指令	说明
#POINTS	标号
SP 700	速度
AC 80000;DC 80000	加、减速度
#LOOP	
VP=@AN[1]*1000	读入模拟输入值并计算位置
PA VP	指令位置
BGA	开始运动
AMA	运动完成后
JP#LOOP	重复循环
EN	结束

2）位置跟随器（连续运动）

方法：读入模拟输入值，计算指令位置值和位置误差。指令电机以与位置误差成比例的速度运转。

指令	说明
#CONT	标号
AC 80000;DC 80000	加减速度
JG 0	开始 JOG 方式
BGX	开始运动
#LOOP	
VP=@AN[1]*1000	计算需要的位置
VE=VP-_TPX	求出位置误差
VEL=VE*20	计算速度
JGVEL	改变速度
JP#LOOP	重复循环
EN	结束

DMC21x3 控制器的扩展 I/O

DMC21x3 控制器可以通过选件扩展 40 个数字 I/O 点，可以通过软件将这些 I/O 点以 8 个一组配置成输入或输出点。

配置 DMC21x3 的 I/O 点

将 DMC21x3 系列控制器的 40 个扩展 I/O 点每相邻 8 个为一组进行配置。扩展 I/O 表示为 2-6 组或 bits l7~56。

用指令 CO 将扩展 I/O 配置成输入或输出。CO 指令有一个参数：

COn

其中，n 是一个数值，转化二进制数，每一位代表扩展 I/O 的一组。当某位为 1 时，对应的组就配置为输出。

最低有效位代表第 2 组。可以用下式求出十进制值：$n=n_2+2*n_3+4*n_4+8*n_5+16*n_6$，其中，$n_x$ 代表组。如果 n_x 值是 1，那么这一组 8 点就要配置成输出。如果 n_x 值是 0，那么，这一组 8 点就要配置为输入。例如，如果要把 4 组和 5 组配置成输出，就写 CO12。

8 位一组的 I/O	组	2 进制表达	十进制的值
17-24	2	2^0	
25-32	3	2^1	
33-40	4	2^2	4
41-48	5	2^3	8
49-56	6	2^4	16

确定 n 的最简单方法：

第 1 步：确定哪 8 位组 I/O 块要配置成输出

第 2 步：从表中，确定要配置为输出的各 I/O 块的十进制值

第 3 步：将第 2 步确定的所有值相加，即为用于 n 的值

例如，如果要将块 2 和块 3 配置为输出，那　n=3，并指令 CO3。

将输出状态存入非易失性存储器中

扩展 I/O 的配置和输出点的状态都能用 BN 指令存入 EEPROM 中。如果未做任何设置，就采用缺省值 CO0（所有块均为输入）。

存取扩展 I/O

当配置为输出时，可以用 SBn 和 CBn 指令对各个 I/O 点进行定义。也可以用条件指令 OBn 定义输出。对于 1 至 4 轴控制器，n=l~8 和 17~56，对于 5 轴以上控制器，n=1~56。

也可以用指令 OP 来设置输出位，指定为数据块。OP 指令接纳 5 个参数。第 1 个参数设置控制器主输出口的值（输出位 1-8，块 0）。其余 4 个参数设置外部扩展 I/O 值：

OP m,a,b,c

其中 m 是代表 bits l-8 的十进制值(0-255)，而 a，b，c，d 代表扩展 I/O，以 16 位连续组为单位，（其值从 0~65535）。对于配置成为输入点的 I/O 点来说，忽略所给的参数，下表说明了用来配置输出点状态的参数。

参数	组	位	说明
m	0	1-8	基本通用输出
a	2，3	17-32	扩展 I/O
b	4，5	33-48	扩展 I/O
c	6，	49-56	扩展 I/O

例如，如果将块 6 配置为输出，就可以用如下指令：

OP7,,,7

此指令会将位 1，2，3（0 组）和位 65，66，67（8 组）置为 1。位 4-8 和位 68~80 置为 0。其它所有位不受影响。

当读取配置为输入的 I/O 块时，使用 TIn 指令。参数‘n’对应要读取的组（n=0，2，3，4，5，6）。读回的值是相对应位的十进制代表值，能用@IN[n]函数查询各位。对于 1 至 4 轴控制器，n=l~8 和 17~56，对于 5 轴以上控制器，n=1~56。如果输入以下指令：

MG @IN[17]

控制器就会返回 2 组的最低有效位的状态(假设 2 组设置为输入)。

应用举例

线缆裁切器

操作人员按下启动开关，使电机前进 10 英寸线距，当运动停止时，控制器产生输出信号，启动切刀。切割完成周期为 100ms。

假设电机通过一个直径为 2 英寸的滚子驱动线缆，同时假定编码器分辨率为 1000 线/rev。由于滚轮的圆周等于 2π 英寸，它对应于 4000 记数值，因此，1 英寸行程就等于：

4000/2π=637cts／inch

这就是说，10 英时的距离等于 6370cts，如旋转速度为 5inches/sec，即等于 3185cts/sec。

输入信号可以用输入点 1，而输出信号选为输出点 1，电机速度轮廓和相关输入、输出信号示于图 7.1。

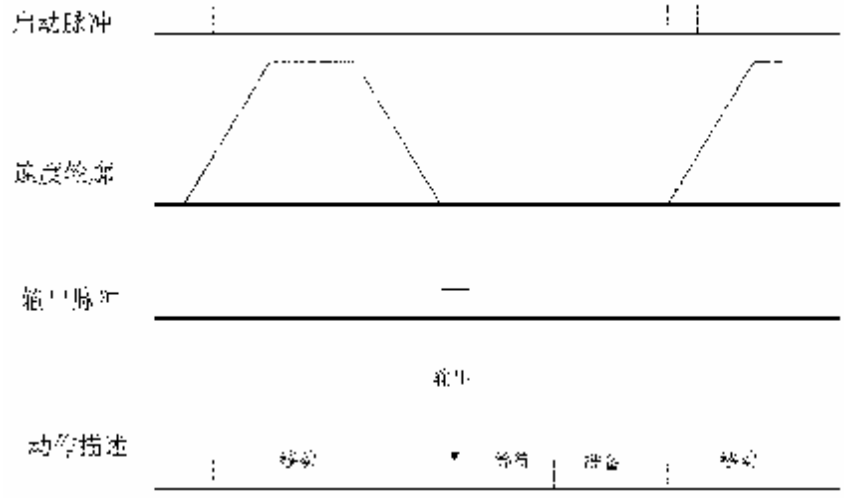


图 7.1 -电机速度和相关输入/输出信号

程序按我们在程序#A 中所定义的状态起动，这里，控制器等待输入点 1 的输入脉冲，一旦脉冲给出，控制器就开始正向运动。

正向运动一完成，控制器就输出 20ms 的脉冲，然后在返回#A 进行下一个新循环之前等待另外 80ms。

指令	说明
#A	标号
AI1	等待输入点 1
PR 6370	距离
SP 3185	速度
BGA	开始运动

AMA	运动完成后
SB1	输出点 1 置位
WT 20	等待 20ms
CB1	输出点 1 清 0
WT 80	等待 80ms
JP#A	重复循环

15．2 X-Y-Z 工作台控制器（3 轴应用）

一台 X-Y-Z 三轴系统必须切割出图 7.2 所示的形状，X-Y 工作台做平面运动，而 Z 轴使切割刀具做上、下运动。

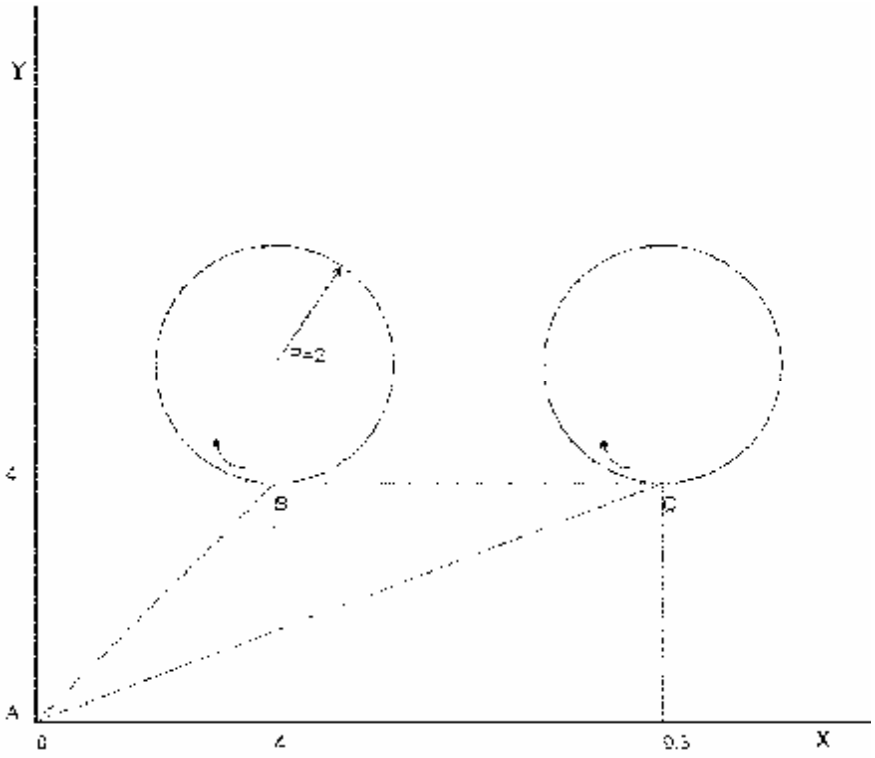


图 7.2 -XY 平面内的切割轨迹

图 7.2 中的实线表示刀具行走的切割路径，要求进给速率为 1inch / sec；虚线表示刀具的非切割路径，运动速率为 5inch/sec，加速度为 0.1g。

运动从 A 点开始，Z 轴抬起，X-Y 轴移动到 B 点的同时，Z 轴向下运动，开始沿圆弧进行切割。圆弧加工运动一完成，Z 轴上升抬起刀具，X、Y 轴继续运动到 C 点，以此类推。

假定，3 个轴的丝杆螺距都为 0.1inch/rev，且编码器的分辨率为 1000 线 / rev，由此得出：1 inch=40,000cts

速度：1in / sec=40,000cts/sec；5in/sec=200,000cts/sec

加速度：0.1g 等于 0.1g=38.6in/sec^2=1,544,000ets/sec^2

注意：圆弧半径为 2 英寸或 80000cts，加工起始点角度为 2700，以 CW（反方向）旋转 360°。这样，就用如下指令规定轨迹：

CR80000,270,-360

进一步假定，Z 釉移动 2 英寸，移动速度为 2 英寸/see。用以下指令完成所要求的运动：

指令	说明
#A	标号
VM XY	X、Y 轴圆弧插补
VP l60000,160000	矢量位置

VE	结束矢量运动
VS 200000	矢量速度
VA l544000	矢量加速度
BGS	开始运动
AMS	运动完成时
PR,,-80000	Z 轴下降
SP,,80000	Z 柚速度
BG Z	开始 Z 轴运动
AM Z	等待 Z 轴运动完成
CR 80000,270,-360	圆弧
VE	结束矢量运动
VS 40000	进给率
BGS	开始圆弧运动
AMS	等待圆弧运动完成
PR,,80000	抬起刀具(Z 轴)
BGZ	开始 Z 轴运动(抬 7J)
AMZ	等待 Z 轴运动完成
PR –21600	移动 X 轴
SP 20000	X 轴速度
BGX	开始 X 轴运动
AMX	等待 X 轴运动完成
PR,,-80000	落下 Z 轴（下刀）
BG Z	
AM Z	
CR 80000,270,-360	第 2 个圆弧切割
VE	
VS 40000	
BG S	
AM S	
PR,,80000	抬刀
BG Z	
AM Z	
VP –37600,-16000	X、Y 轴返回到起点
VE	
VS 200000	
BG S	
AM S	
EN	

用操控杆（Joystick）进行速度控制

用操控杆控制电机的速度，操控杆产生-10V~+10V 之间的电压信号，作为速度信号控制电机运转。

假定，满 10V 电压对应电机速度为 3000rpm，编码器分辨率为 1000 线/rev 即 4000cts/rev，此速度等于：

3000rpm=50rev/sec=200000cts/sec

程序周期性地读取输入电压，并将此值分配给变量 VIN，要得到对应 10V 的速度

200,000cts/sec，我们选择速度为：

Speed：20000*VIN

将对应的电机速度分配给变量 VEL。

```
指令
#A
JG0
BGX
#B
VIN=@AN[1]
VEL=VIN*20000
JG VEL
JP#B
EN
```

用操控杆进行定位控制

此系统要求电机的位置与操控棒角度成比例，也就是说，两个位置之间比例必须是可设置的。例如，如果控制比例系数为 5：1，就意味着，当操控棒电压为 5V 时，对应 1024cts，所要求的电机位置必须是 5120cts。变量 V3 改变位置比例系数。

指令	说明
#A	标号
V3=1024	初始位置比例系数
DP0	定义起始位置
JG0	设定 JOG 方式速度为 0
BGX	开始运动
#B	
V1=@AN[1]	读模拟输入
V2=V1*V3	计算目标位置
V4=V2-_TPA-_TEA	求随动误差
V5=V4*20	计算比例速度
JG V5	改变 JOG 速度
JP#B	重复循环
EN	结束

用采样双闭环进行间隙补偿

连续双闭环由 DV1 函数进行使能，它是一种进行间隙补偿的有效方法。不过，在某些场合，当间隙较大时，就很难使系统稳定。此时，采用下述的采样双闭环方法就相对容易一些。

这里，针对运动控制系统中间隙所带来的麻烦，提出解决方案。目的是能精确地控制直线滑台的位置。滑台由旋转电机控制，电机通过丝杠机与滑台相连。这里，丝杠间隙为 4μm，所要求的定位精度为 0.5μm。

问题在于传感器安装在何处。如果采用旋转编码器，那么就存在 4μm 的间隙误差。另一方面，若采用直线编码器（光栅），由于不稳定性，反馈回路的间隙就会引起系统振荡。

另一种途径是采用双闭环，这里，我们采用两个传感器，旋转编码器和光栅。旋转编码器确保系统稳定性（因为在间隙之前进行位置闭环），而光栅提供精确的负载位置信息。其工作原理是驱动电机到终点附近的给定旋转位置。一旦到达，就读取负载位置，求出位置误差，控制器指令电机移动到消除此位置误差的新的旋转位置。

由于所要求的精度是 0.5μm，光栅的分辨率应该高于其两倍。分辨率为 0.25μm 的光栅能

得到±2cts 的位置误差。

双闭环方法要求旋转编码器的分辨率等于或好于直线编码器的分辨率。假定，丝杠螺距为 2.5mm（约 10 圈/inch），旋转编码器为 2500 线/rev，即 10000cts/rev，即相当于 0.25μm / cts 脉冲当量，这样就使光栅和旋转编码器具有相同的分辨率。

为了说明这种控制方法，假定，旋转编码器用于 X 轴反馈，读取光栅反馈信号并存在变量 LINPOS 中，再假设，在起点处，x 轴的位置和 LINPOS 的值都为 0，现在，我们假定，要把直线负载移动到位置 1000。

第一步是指令 X 轴电机移动到旋转位置 1000，一旦到达，我们再检查负载位置，例如，若负载位置为 980cts，这就意味着必须进行 20cts 的修正。不过，当把 X 轴移动到位置 1000 时，假设实际位置只时 995，意味着 X 轴位置误差为 5cts，一旦电机稳定下来，就会消除。这意味着只需要修正 15cts，由于 X 轴已修正了 20cts 中的 5cts。据此，运动修正就应该是：

修正值=负载位置误差—旋转位置误差

进行数次修正，直到误差小于±2cts。往往在一个修正循环中即可完成。

指令	说明
#A	标号
DP0	定义起始位置为 0
LINPOS=0	
PRl000	距离
BGA	开始运动
#B	
AMA	等待运动完成
WT 50	暂停 50ms
LINPOS=_DEA	读直线位置
ER=1000-LINPOS-_DEA	求修正值
JP#C,@ABS[ER]<2	若误差小于 2，则退出
PR ER	定义修正
BGA	
JP#B	重复循环
#C	
EN	

第八章　防护

说明

DMC-21x3 提供了几种硬件、软件保护功能，以检查外部错误，并禁止电机运动。这些功能有助于保护各种系统元件免受损坏。

警告：机器在运动中可能有危险！用户有责任设计有效的报警处理及安全保护。由于 DMC-21x3 是机器中的一个集成部分，因此，工程师应该设计具有保护措施的完整系统，以免造成 DMC-21x3 元件损坏。对于任何伤害或损坏，GALIL 不承担任何责任。

硬件防护

DMC-21x3 有硬件输入和输出保护，用于报警和机械限位保护。

输出保护

AMPEN（放大器使能）：控制器给出关断电机命令（MO）时，此信号就变为低电平；当报警关断条件使能时（OEl），还有其它一些条件都会使 AMPEN 变低，而切断使能。这些条件如下：

（1） 位置误差超过由 ER 命令（误差限制）所设定的限制值时（即位置超差时）
（2） 给出急停命令时。各轴放大器都有各自的放大器使能线（AMPEN），
（3） 当看门狗定时器触发或复位时，此信号也变低从而关断伺服输出。

注意：AMPEN 信号的标准配置为 TTL 低有效。由于在 ICM-1900ITC 和 ICM-20105T 中，对 AMPEN 输出信号已进行光隔处理，并采用 OC 输出，因此，可以与任何电压的伺服放大器或步进驱动器直接接口。

报警输出：在 DMC-21x3 及其它 DMC 控制器中，报警输出为 TTL 信号，以此表示控制器中的报警状态。此信号在互联模块（接口板）侧，称作 ERROR。以下几种情况可能导致报警信号变低而出现报警输出：

（1） 至少一个轴出现位置超差；误差限制值由 ER 命令进行设定。
（2） 控制器侧的复位线一直为低或受干扰而影响。
（3） 控制器有故障且处理器正在自身复位。
（4） 驱动报警信号的输出 IC 芯片有故障。

注意：当用户选用 ICM-1900ITC 或 ICM-20105T 接口板与 GALIL 控制器配套时，报警输出信号 ERROR 已经光隔处理或 OC 输出形式，因此，可以很方便地与任何外部接口线路相连接。

输入保护

通用急停（ABORT）：此输入信号一变低，就立即停止运动，而且是不经过减速处理的停止。对于报警关断功能使能的任何轴来说，都会使其使能信号（AMPEN）关断。这样会使电机依靠自身惯性自由滑行而停止。如果没有使报警关断功能使能，由于此时，控制器输出命令突然变为 0，因此，电机会立即停止，锁定在当前位置。

选择性急停：也可以对控制器进行配置，对各轴提供各自的急停。其作用与 ABORT 输入相同，但只对指定轴有效。要想把控制器配置成选择性急停，使用命令 CN,,,1 即可。这样就将

输入点 5，6，7，8，13，14，15，16 配置成分别对应轴 A，B，C，D，E，F，G，H 的选择性急停。

正向限位开关：此信号变低禁止正向运动。当此限位开关生效时，电机正在以正方向运动，那么，就会使运动减速而停止。此外，如果电机正在以正方向运动，控制器就会自动跳转到限位开关子程序#LIMSWI（在用户将此子程序写入应用程序的情况下）。可以用 CN 命令改变限位开关的极性。

反向限位开关：此信号变低禁止反向运动。当此限位开关生效时，电机正在以反方向运动。那么，就会使运动减速而停止。此外，如果电机正在以反方向运动，控制器就会自动跳转到限位开关子程序#LIMSWI（在用户将此子程序写入应用程序的情况下）。可以用 CN 命令改变限位开关的极性。

软件防护

DMC-21x3 提供了可编程误差限制功能。用 ERn 命令设置误差限制值。其中：n=1~32767。ER 的缺省值为 16384。

ER 200,300,400,500　　设置 A 轴误差限制为 200，B 轴为 300，C 轴为 400，D 轴为 500

ER,1,,10　　设置 B 轴误差限制为 lcts，D 轴为 10cts.

误差限制值的单位为正交计数脉冲（cts）。误差是命令位置与实际编码器反馈值之差。如果误差的绝对值超过由 ER 所规定的值，DMC-21x3 就会产生以下几种信号，警告主系统：

信号或功能	报警出现时的状态
#POSERR	跳转到超差自动处理子程序
Error Light	点亮
OE 功能	若为 OE1，则关断电机使能
AEN 输出线	变低

设置条件语句进行跳转有助于根据不同报警在程序内部跳转。在运行期间，用 TE 命令可以对 A，B，C，D 轴的位置误差进行监控。

可设置位置限制（软件限位）功能

DMC-21x3 具有软件限位功能，可以用 BL 和 FL 软件命令对正、反向限位值进行软件设置。一旦指定位置限位值，DMC-21x3 就不会接受超过此限制值的位置命令。当然，也防止了超过限制值的运动。

举例

指令	说明
DP0,0,0	定义位置
BL-2000,-4000,-8000	设定反向位置限制值
FL 2000,4000,8000	设定正向位置限制值
JG 2000,2000,2000	Jog 运动
BG XYZ	开始

运行此程序，运动停止在正向限制值处

报警关断功能（Off-On-Error）

DMC-21x3 控制器具有在某些报警条件下，关断电机的内置功能。此功能称之为“Off-On-Error”（报警关断）。要想使 OE 功能对各轴起作用，就指定 OE1，要使此功能无效，就指定 OE0。

当此功能使能时，在以下 3 种条件下，都会使指定的电机使能关断。

（1） 指定轴的位置误差超过用 ER 命令设定的限制值；

（2） 给了急停命令(AB)

（3） 急停输入变低而起作用

注意：如果在电机运行期间，关断使能，那么，由于此时电机不再工作在伺服控制方式下，因此会依靠自身惯量而慢慢停止。

要想重新使系统使能，请使用伺服使能（SH）命令。

举例：

OE1,1,1,1	对 A、B、C、D 轴，报警关断功能使能
OE0,1,0,1	对 B、D 轴报警关断功能使能，对 A、C 轴，报警关断功能无效

超差自动处理子程序

如果任何一个轴的误差超过了由命令 ER 所指定的误差极限值，#POSERR 标号就使跟随其后的语句自动被执行。误差子程序必须用 RE 命令来结束。RE 命令会使程序执行从误差子程序返回到主程序。

注意：如果不消除产生超差的条件，还会再次进入误差子程序。

举例：

指令	说明
#A;JP#A;EN	“虚拟”程序
#POSERR	用超差起动子程序
MG”error”	发送信息
SB1	输出点 1 置位（使继电器吸合）
ST A	停止 A 轴
AM A	在停止之后
SH A	伺服使能以清除误差
RE	返回主程序

注意：只有在应用程序执行期间，#POSERR 子程序才会起作用。

限位信号处理子程序

DMC-21x3 具有硬件正、反向限位功能。对于限位开关子程序，也有一个特殊标号，#LIMSWI 标号指定开始限位开关子程序。如果任何一个限位开关起作用，而且该轴电机正在以同方向运动，那么此标号就便跟随其后的语句自动被执行。RE 命令使此子程序结束。在条件跳转期间，也可以测试正、反向限位开关的状态。_LR 指定反向限位，而_LF 指定正向限位。跟随 LR 或 LF 的 A，B，C，D 指定轴。用 CN 命令配置限位开关的极性。

举例：

指令	说明
#A;IP#A;EN	虚拟程序
#LIMSWI	限位开关处理子程序
V1=_LFA	检查是否正向限位
V2=_LRA	检查是否反向限位
JP#LF，V1=0	如果处于正向限位，跳转到#LF
JP#LR，V2=0	如果处于负向限位，跳转到#LR
JP#END	跳转到#END
#LF	标号
MG”FORWARD LIMIT”	发送信息
STX; AMA	停止运动
PR-1000;BGA;AMA	反向运动

JP#END	跳转到#END
#LR	标号
MG”REVERSE LIMIT”	发送信息
STX;AMA	停止运动
PR 1000;BGA;AMA	正向运动
#END	结束
RE	返回到主程序

注章：只有在应用程序执行期间，#LIMSWI 才会起作用。

驱动器错误处理子程序

DMC-21x3 使用 GALIL 提供的驱动器选件时，控制器可以与驱动器通讯，监视驱动器的工作状态。如果驱动器出现过流、过压异常情况，应用程序会自动跳转到#AMPERR 处，在这里，用户可以编写自己的处理程序，向使用者提示驱动器错误、关闭驱动器或输出报警信号等等。

附录

电气规格

伺服控制

模拟控制输出	±10V 模拟量信号，16 位 DA 转换，最大电流 3mA
A+,A-,B+,B-,I+,I-主、辅编码器信号	TTL 兼容信号，但最大接受电压为±12V。可接受单端（A+,B+）或差动信号（A+,A-,B+,B-）的正交方波信号或脉冲（A）方向（B）信号。对于正交方波信号，最大记数频率 12MHz（四倍频以后），对于脉冲方向信号，最大频率 3MHz。索引脉冲最小宽度 80nsec

脉冲控制

脉冲	TTL 电平，50%方波输出，最高频率 3000000Hz
方向	TTL 电平

输入/输出

限位开关输入、原点输入 IN[1]-IN[8]通用输入和急停输入 IN[9]-IN[16]通用输入(5 轴以上控制器）	TTL 电平，内部通过 4.7K 电阻接到 5VDC
OUT[1]-OUT[8]通用输出 OUT[9]-OUT[16]通用输出（5 轴以上控制器）	TTL 电平
IN[81]、IN[82]	A 轴的辅助编码器输入，可接收±12V 的单端或差动信号
IN[83]、IN[84]（2 轴以上控制器）	B 轴的辅助编码器输入，可接收±12V 的单端或差动信号
IN[85]、IN[86]（3 轴以上控制器）	C 轴的辅助编码器输入，可接收±12V 的单端或差动信号
IN[87]、IN[88]（4 轴以上控制器）	D 轴的辅助编码器输入，可接收±12V 的单端或差动信号
IN[89]、IN[90]（5 轴以上控制器）	E 轴的辅助编码器输入，可接收±12V 的单端或差动信号
IN[91]、IN[92]（6 轴以上控制器）	F 轴的辅助编码器输入，可接收±12V 的单端或差动信号
IN[93]、IN[94]（7 轴以上控制器）	G 轴的辅助编码器输入，可接收±12V 的单端或差动信号
IN[95]、IN[96]（8 轴控制器）	H 轴的辅助编码器输入，可接收±12V 的单端或

	差动信号
IN[83]、IN[84]（2 轴以上控制器）	B 轴的辅助编码器输入，可接收±12V 的单端或差动信号

电源

+5V	1.1A
+12V	40mA
-12V	40Ma

性能规格

最小伺服更新时间

型号	最小伺服更新时间（单位：1/1024ms）
DMC-2112/DMC-2113	250
DMC-2122/DMC-2123	250
DMC-2132/DMC-2133	375
DMC-2142/DMC-2143	375
DMC-2152/DMC-2153	500
DMC-2162/DMC-2163	500
DMC-2172/DMC-2173	625
DMC-2182/DMC-2183	625

位置精度：	±1 个计数单位
速度精度：	
长程	优于 0.005%
短程	取决于包括机构在内的整个系统
位置范围	每次移动±2147483647
速度范围	12MHz 模拟量闭环控制
	3MHz 脉冲控制
速度分辨率	2 计数单位/秒
模拟量分辨率	16 位，0.0003V
变量取值范围	±20 亿
变量精度	1/65536
数组空间大小	8000 个元素，30 个数组
程序空间大小	1000 行*80 列

高速模式

DMC-21x3 在高速模式下，可以得到更快的伺服更新速度。在高速模式下，控制器的伺服更新速度如下：

1-2 轴	125
3-4 轴	250
5-6	375

7-8 500

使控制器在高速模式下工作，必须将高速固件（fast firmware）下载到控制器上。可以用 GALIL 提供的终端程序，如智能终端（Smart Terminal）来完成这个工作。如需要高速固件，您可以和上海微敏自控技术有限公司联系。更新固件以后，就可以用 TM 指令设置更快的伺服更新速度。

当控制器工作在快速模式下时，以下功能将无法使用：

电子齿轮同步
电子凸轮同步
脉冲输出控制
第 2 到第 8 个程序通道中事件触发
当前速度查询指令（TV）
PL 指令

DMC-21x3 控制卡连接口

DMC-21x2 上高密度接头 J4（A-D 轴）

1	保留	51	保留
2	GND	52	GND
3	+5V	53	+5V
4	误差输出	54	保留
5	复位	55	D 轴原点输入
6	位置比较输出	56	D 轴反向极限输入
7	GND	57	D 轴正向极限输入
8	GND	58	C 轴原点输入
9	D 轴模拟控制输出	59	C 轴反向极限输入
10	D 轴方向输出	60	C 轴正向极限输入
11	D 轴脉冲输出	61	B 轴原点输入
12	C 轴模拟控制输出	62	B 轴反向极限输入
13	C 轴方向输出	63	B 轴正向极限输入
14	C 轴脉冲输出	64	A 轴原点输入
15	B 轴模拟控制输出	65	A 轴反向极限输入
16	B 轴方向输出	66	A 轴正向极限输入
17	B 轴脉冲输出	67	GND
18	A 模拟控制输出	68	+5V
19	A 轴方向输出	69	保留
20	A 轴脉冲输出	70	A 轴锁存/通用输入 1
21	D 轴驱动器使能输出	71	B 轴锁存/通用输入 2
22	C 轴驱动器使能输出	72	C 轴锁存/通用输入 3
23	B 轴驱动器使能输出	73	D 轴锁存/通用输入 4
24	A 轴驱动器使能输出	74	A 轴急停/通用输入 5
25	A 轴主编码器 A+	75	B 轴急停/通用输入 6
26	A 轴主编码器 A-	76	C 轴急停/通用输入 7

27	A 轴主编码器 B+	77	D 轴急停/通用输入 8
28	A 轴主编码器 B-	78	急停输入
29	A 轴主编码器 I+	79	通用输出 1
30	A 轴主编码器 I-	80	通用输出 2
31	B 轴主编码器 A+	81	通用输出 3
32	B 轴主编码器 A-	82	通用输出 4
33	B 轴主编码器 B+	83	通用输出 5
34	B 轴主编码器 B-	84	通用输出 6
35	B 轴主编码器 I+	85	通用输出 7
36	B 轴主编码器 I-	86	通用输出 8
37	C 轴主编码器 A+	87	+5V
38	C 轴主编码器 A-	88	GND
39	C 轴主编码器 B+	89	GND
40	C 轴主编码器 B-	90	GND
41	C 轴主编码器 I+	91	A 轴辅编码器 A+
42	C 轴主编码器 I-	92	A 轴辅编码器 A-
43	D 轴主编码器 A+	93	A 轴辅编码器 B+
44	D 轴主编码器 A-	94	A 轴辅编码器 B-
45	D 轴主编码器 B+	95	B 轴辅编码器 A+
46	D 轴主编码器 B-	96	B 轴辅编码器 A-
47	D 轴主编码器 I+	97	B 轴辅编码器 B+
48	D 轴主编码器 I-	98	B 轴辅编码器 B-
49	+12V	99	-12V
50	+12V	100	-12V

DMC-21x2 上高密度接头 J5（E-H 轴）

1	保留	51	保留
2	GND	52	GND
3	+5V	53	+5V
4	误差输出	54	保留
5	复位	55	H 轴原点输入
6	位置比较输出	56	H 轴反向极限输入
7	GND	57	H 轴正向极限输入
8	GND	58	G 轴原点输入
9	H 轴模拟控制输出	59	G 轴反向极限输入
10	H 轴方向输出	60	G 轴正向极限输入
11	H 轴脉冲输出	61	F 轴原点输入
12	G 轴模拟控制输出	62	F 轴反向极限输入
13	G 轴方向输出	63	F 轴正向极限输入
14	G 轴脉冲输出	64	E 轴原点输入
15	F 轴模拟控制输出	65	E 轴反向极限输入
16	F 轴方向输出	66	E 轴正向极限输入
17	F 轴脉冲输出	67	GND

18	E 轴模拟控制输出	68	+5V
19	E 轴方向输出	69	保留
20	E 轴脉冲输出	70	E 轴锁存输入/通用输入 9
21	H 轴驱动器使能输出	71	F 轴锁存/通用输入 10
22	G 轴驱动器使能输出	72	G 轴锁存/通用输入 11
23	F 轴驱动器使能输出	73	H 轴锁存/通用输入 12
24	E 轴驱动器使能输出	74	E 轴急停/通用输入 13
25	E 轴主编码器 A+	75	F 轴急停/通用输入 14
26	E 轴主编码器 A-	76	G 轴急停/通用输入 15
27	E 轴主编码器 B+	77	H 轴急停/通用输入 16
28	E 轴主编码器 B-	78	急停输入
29	E 轴主编码器 I+	79	通用输出 9
30	E 轴主编码器 I-	80	通用输出 10
31	F 轴主编码器 A+	81	通用输出 11
32	F 轴主编码器 A-	82	通用输出 12
33	F 轴主编码器 B+	83	通用输出 13
34	F 轴主编码器 B-	84	通用输出 14
35	F 轴主编码器 I+	85	通用输出 15
36	F 轴主编码器 I-	86	通用输出 16
37	G 轴主编码器 A+	87	+5V
38	G 轴主编码器 A-	88	GND
39	G 轴主编码器 B+	89	GND
40	G 轴主编码器 B-	90	GND
41	G 轴主编码器 I+	91	E 轴辅编码器 A+
42	G 轴主编码器 I-	92	E 轴辅编码器 A-
43	H 轴主编码器 A+	93	E 轴辅编码器 B+
44	H 轴主编码器 A-	94	E 轴辅编码器 B-
45	H 轴主编码器 B+	95	F 轴辅编码器 A+
46	H 轴主编码器 B-	96	F 轴辅编码器 A-
47	H 轴主编码器 I+	97	F 轴辅编码器 B+
48	H 轴主编码器 I-	98	F 轴辅编码器 B-
49	+12V	99	-12V
50	+12V	100	-12V

DMC-21x3 上高密度接头 J4（A-D 轴）

1	GND	33	GND	65	GND
2	D 轴脉冲输出	34	D 轴方向输出	66	D 轴模拟控制输出
3	C 轴脉冲输出	35	C 轴方向输出	67	C 轴模拟控制输出
4	B 轴脉冲输出	36	B 轴方向输出	68	B 轴模拟控制输出
5	A 轴脉冲输出	37	A 轴方向输出	69	A 轴模拟控制输出
6	D 轴使能输出	38	GND	70	位置比较输出
7	A 轴使能输出	39	B 轴使能输出	71	C 轴使能输出
8	D 轴原点输入	40	D 轴反向极限输入	72	D 轴正向极限输入

9	C 轴原点输入	41	C 轴反向极限输入	73	C 轴正向极限输入
10	B 轴原点输入	42	B 轴反向极限输入	74	B 轴正向极限输入
11	A 轴原点输入	43	A 轴反向极限输入	75	A 轴正向极限输入
12	A 轴锁存/通用输入 1	44	B 轴锁存/通用输入 2	76	C 轴锁存/通用输入 3
13	D 轴锁存/通用输入 4	45	A 轴急停/通用输入 5	77	B 轴急停/通用输入 6
14	C 轴急停/通用输入 7	46	D 轴急停/通用输入 8	78	急停输入
15	通用输出 3	47	通用输出 2	79	通用输出 1
16	通用输出 5	48	GND	80	通用输出 4
17	通用输出 8	49	通用输出 7	81	通用输出 6
18	A 轴主编码器 A+	50	A 轴主编码器 A-	82	A 轴主编码器 B+
19	A 轴主编码器 B-	51	A 轴主编码器 I+	83	A 轴主编码器 I-
20	B 轴主编码器 A+	52	B 轴主编码器 A-	84	B 轴主编码器 B+
21	B 轴主编码器 B-	53	B 轴主编码器 I+	85	B 轴主编码器 I-
22	C 轴主编码器 A+	54	C 轴主编码器 A-	86	C 轴主编码器 B+
23	C 轴主编码器 B-	55	C 轴主编码器 I+	87	C 轴主编码器 I-
24	D 轴主编码器 A+	56	D 轴主编码器 A-	88	D 轴主编码器 B+
25	D 轴主编码器 B-	57	D 轴主编码器 I+	89	D 轴主编码器 I-
26	GND	58	GND	90	GND
27	A 轴辅编码器 A+	59	A 轴辅编码器 A-	91	A 轴辅编码器 B+
28	A 轴辅编码器 B-	60	B 轴辅编码器 A+	92	B 轴辅编码器 A-
29	B 轴辅编码器 B+	61	B 轴辅编码器 B-	93	C 轴辅编码器 A+
30	C 轴辅编码器 B+	62	D 轴辅编码器 A+	94	误差输出
31	-12V	63	复位输入	95	+12V
32	+5V	64	+5V	96	+5V

DMC-21x3 上高密度接头 J5（E-H 轴）

1	GND	33	GND	65	GND
2	H 轴脉冲输出	34	H 轴方向输出	66	H 轴模拟控制输出
3	G 轴脉冲输出	35	G 轴方向输出	67	G 轴模拟控制输出
4	F 轴脉冲输出	36	F 轴方向输出	68	F 轴模拟控制输出
5	E 轴脉冲输出	37	E 轴方向输出	69	E 轴模拟控制输出
6	H 轴使能输出	38	GND	70	位置比较输出
7	E 轴使能输出	39	F 轴使能输出	71	G 轴使能输出
8	H 轴原点输入	40	H 轴反向极限输入	72	H 轴正向极限输入
9	G 轴原点输入	41	G 轴反向极限输入	73	G 轴正向极限输入
10	F 轴原点输入	42	F 轴反向极限输入	74	F 轴正向极限输入
11	E 轴原点输入	43	E 轴反向极限输入	75	E 轴正向极限输入
12	E 轴锁存/通用输入 9	44	F 轴锁存/通用输入 10	76	G 轴锁存/通用输入 11
13	H 轴锁存/通用输入 12	45	E 轴急停/通用输入 13	77	F 轴急停/通用输入 14
14	G 轴急停/通用输入 15	46	H 轴急停/通用输入 16	78	急停输入
15	通用输出 11	47	通用输出 10	79	通用输出 9
16	通用输出 13	48	GND	80	通用输出 12
17	通用输出 16	49	通用输出 15	81	通用输出 14

18 E 轴主编码器 A+	50 E 轴主编码器 A-	82 E 轴主编码器 B+
19 E 轴主编码器 B-	51 E 轴主编码器 I+	83 E 轴主编码器 I-
20 F 轴主编码器 A+	52 F 轴主编码器 A-	84 F 轴主编码器 B+
21 F 轴主编码器 B-	53 F 轴主编码器 I+	85 F 轴主编码器 I-
22 G 轴主编码器 A+	54 G 轴主编码器 A-	86 G 轴主编码器 B+
23 G 轴主编码器 B-	55 G 轴主编码器 I+	87 G 轴主编码器 I-
24 H 轴主编码器 A+	56 H 轴主编码器 A-	88 H 轴主编码器 B+
25 H 轴主编码器 B-	57 H 轴主编码器 I+	89 H 轴主编码器 I-
26 GND	58 GND	90 GND
27 E 轴辅编码器 A+	59 E 轴辅编码器 A-	91 E 轴辅编码器 B+
28 E 轴辅编码器 B-	60 F 轴辅编码器 A+	92 F 轴辅编码器 A-
29 F 轴辅编码器 B+	61 F 轴辅编码器 B-	93 G 轴辅编码器 A+
30 G 轴辅编码器 B+	62 H 轴辅编码器 A+	94 误差输出
31 -12V	63 复位输入	95 +12V
32 +5V	64 +5V	96 +5V

DMC21x2/21x3 上 CD 轴辅助编码器接线 JP6

1	+5V
2	GND
3	C 轴辅编码器 A+
4	C 轴辅编码器 A-
5	C 轴辅编码器 B+
6	C 轴辅编码器 B-
7	D 轴辅编码器 A+
8	D 轴辅编码器 A-
9	D 轴辅编码器 B+
10	D 轴辅编码器 B-

DMC21x2/21x3 上 GH 轴辅助编码器接线 JP8

1	+5V
2	GND
3	G 轴辅编码器 A+
4	G 轴辅编码器 A-
5	G 轴辅编码器 B+
6	G 轴辅编码器 B-
7	H 轴辅编码器 A+
8	H 轴辅编码器 A-
9	H 轴辅编码器 B+
10	H 轴辅编码器 B-

RS232 接口

标准 DB9 芯公插座

1	CTS——输出

2	TD——输出
3	RD——输入
4	RTS——输入
5	GND
6	CTS——输出
7	RTS——输入
8	CTS——输出
9	保留

以太网接口

1	TXP
2	TXN
3	RXP
4	保留
5	保留
6	RXN
7	保留
8	保留

DMC21x2/DMC21x3 信号描述

输出信号

模拟控制输出	±10V 模拟量信号，16 位 DA 转换，用于控制伺服驱动器。在伺服状态下，信号每个周期都会变化，在电机关闭的情况下（MO），电压保持在 OF 指定的数值上。
伺服使能	用于控制伺服驱动器的使能
脉冲输出	1、出脉冲信号，与方向信号一同控制步进电机或工作在位置模式下的伺服电机。（对应轴的 SM 跳线短接，并设置 MT 为 2，-2，2.5 或 -2.5） 2、输出频率为 25KHz 的 PWM 波，以这种方式发送控制模拟量。PWM 输出有两种模式：SM 跳线放开，占空比 0.2%对应-10V，99.8%对应 +10V；SM 跳线短接，占空比 0%对应 0V，99.6%对应 10V，方向信号输出表示极性
方向	在对应轴的 SM 跳线短接时，输出方向信号
误差输出	当输出信号变为低电平时，说明某个轴的控制误差大于 ER 设置值。此外，当控制器上电、复位时，输出亦为地电平
通用输出	TTL 输出用于用户定义的通用输出。指令 SB 设置单个输出；指令 CB 清除单个输出；指令 OP 设置所有通用输出状态

输入信号

编码器 A+、B+	接收增量式编码器的位置反馈信号。对于正交 A/B 信号的编码器，GALIL 进行四倍频处理，最高计数频率为 12,000,000Hz。对于脉冲/

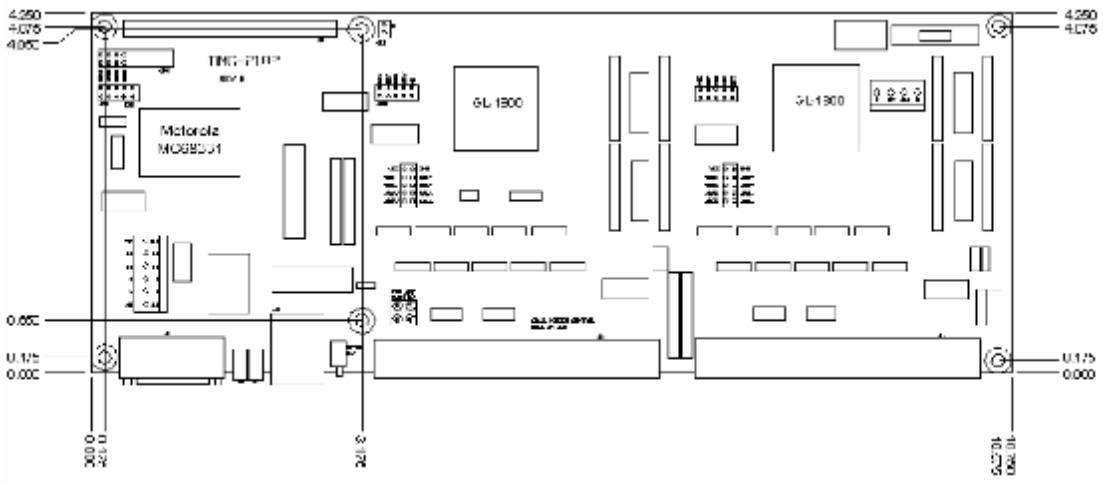
	方向的信号，用 CE 指令设置接受模式以后，也可以处理，但必须脉冲接 A，方向接 B。关于 CE 指令的细节，参考《指令手册》
编码器 I+	编码器的单圈脉冲，对于旋转编码器，该信号一圈有只在固定的位置产生一个，对于线性编码器，可能在固定的位置产生一个、多个该信号，也可能没有该信号，请参考该编码器的资料
编码器 A-,B-,I-	编码器信号的差动输出端。
辅编码器 A+, A-, B+ , B-	每轴附加的编码器输入，对于设置为脉冲输出的轴无效。
急停	输入低电平，立即停止运动指令，没有减速过程，同时停止控制卡上的程序
复位	输入低电平，控制器恢复到上电时的状态。
正向极限	输入有效时，停止正向运动，触发#LIMSWI 程序。信号极性可以通过 CN 指令定义。
反向极限	输入有效时，停止反向运动，触发#LIMSWI 程序。信号极性可以通过 CN 指令定义。
原点	用于回原点指令（HM、FE）的输入，在 HM 或 FE 指令后，指令 BG 使电机加速到运行速度，原点信号的变化会使电机减速停止。信号极性可以通过 CN 指令定义
通用输入	用于用户定义的通用输入
锁存输入	用于高速锁存瞬间的编码器的值。对于锁存信号是响应约为 20 纳秒

DMC-21x3 控制器上跳线描述

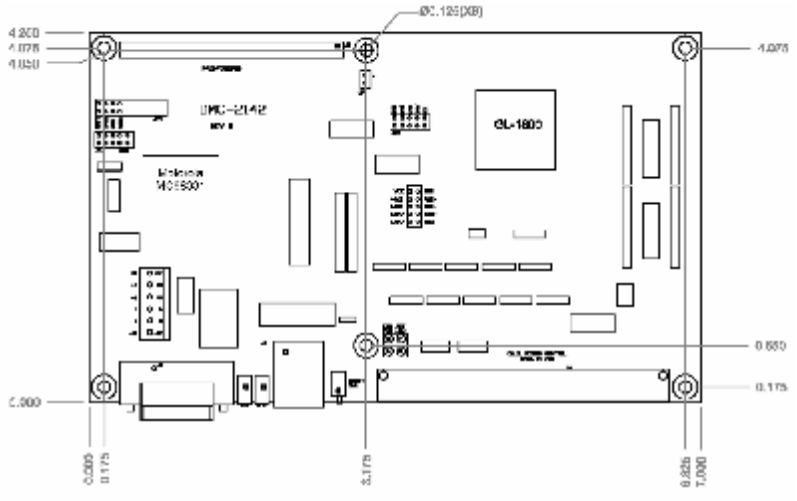
JP5	MO	在上电或复位时强制使能信号输出无效
	SMX	选择使用方向信号控制电机驱动器。如使用步进电机或伺服电机的位置模式，必须短接此跳线。
	SMY	
	SMZ	
	SMW	
JP7	SME	
	SMF	
	SMG	
	SMH	
	OPT	保留
JP2	MRST	在上电或复位时将控制器强制回复到出厂状态
	UPGD	在控制器内固件（Firmware）出现错误时，用于更新控制器内的固件

DMC21x2/3 的尺寸大小

DMC2152/3——DMC2182/3 的尺寸图



DMC2112/3——DMC2142/3 的尺寸图



附件和选件

CAB-100-10T	用于 DMC-21x2 上 J4 或 J5 的连接线，长度 1M
CAB-100-22T	用于 DMC-21x2 上 J4 或 J5 的连接线，长度 2M
CAB-100-30T	用于 DMC-21x2 上 J4 或 J5 的连接线，长度 3M
ICM-1900IT-C	用于 DMC-21x2 上 J4 或 J5 的接线端子板并提供输入信号的光耦隔离和输出信号的驱动
ICM-20105T	用于 DMC-21x3 上信号的接线端子板并提供输入信号的光耦隔离和输出信号的驱动
DB-28040	用于 DMC-21x3 上的 I/O 扩展板，提供 40 路数字 I/O 和 8 路模拟量输入

更多功能扩展选件及其详细介绍，请参考《DMC21x3 附件手册》

上海微敏自控技术有限公司
地址：上海市静安区淮安路 681 号 406 室 电话：+86 -21-32271250
E-mail :ssh@mttc.com.cn +86-21-32271252
http://www.mttc.com.cn 邮编：200041