深圳市雷赛控制技术有限公司
SHENZHEN LEADSHINE CONTROL TECHNOLOGY CO.,LTD

雷赛控制技术 DMC3000 系列运动控制卡

用户使用手册

Version 2.0

2023.9.27

版 权 说 明

本手册中的信息资料仅供参考。由于改进设计和功能等原因，雷赛控制技术保留对本资料的最终解释权，内容如有更改，恕不另行通知。

调试机器要注意安全！用户必须在机器中设计有效的安全保护装置，在软件中加入出错处理程序。否则所造成的损失，雷赛控制技术没有义务或责任负责。

目 录

第 1 章　产品概述

1.1　DMC3000 系列运动控制卡的特点

DMC3000系列运动控制卡是深圳市雷赛控制技术有限公司开发出具有自主知识产权的新型运动控制卡。其CPU更高级，运动控制算法更完善，控制性能更快速、更优秀。主要表现有：

（1）DMC3000系列最多可同时控制12轴电机运动及检测8轴编码器信号。

（2）DMC3000 系列支持梯形、S 形加减速的点位运动控制以及支持运动中变速、变位置功能。

（3）DMC3000 系列支持两个坐标系，每个坐标系支持 2~8 轴的直线插补运动或 2 轴圆弧插补运动，在某两轴进行圆弧插补运动的同时可进行其余多轴的直线插补运动。

（4）DMC3000 系列支持多达 5000 个位置点的 PVT 运动曲线规划高级功能，根据位置点的相关信息：时间、位置、速度，实现在准确的时间点以准确的速度到达确定的位置；可通过自定义数据实现复杂轨迹多轴连续插补运动功能。

（5）DMC3000 系列支持手轮运动配置功能，改变以往每个轴都需要配一个手轮的方式。可任意配置一个轴或多个轴按不同的倍率跟随一个手轮运动。任意配置跟随轴增强了手轮复用性能，多轴跟随手轮可实现多轴协同精确定位功能。

（6）DMC3000 系列支持异常减速停止时间设置功能，异常减速停止包括命令减速停止、硬限位减速停止、软限位减速停止、IO 触发减速停止等，根据现场实际需求情况设定减速停止时间可达到理想的减速效果。

（7）DMC3000 系列支持 IO 输出延时翻转功能，可实现精确输出脉冲信号控制执行器件动作，如：照相机曝光时间控制，点胶机出胶量控制等。

（8）DMC3000 系列支持 IO 计数及滤波功能，可实现在干扰环境下准确计数的功能，可用于统计加工件数等生产统计功能，也可用于位置比较输出次数等数据校验功能。

（9）DMC3000 系列支持一维、二维低速位置比较输出功能，多达 256 个比较缓冲区，可灵活配置比较输出模式，比较周期可达 1 毫秒以内。在确定的二维运动轨迹上确定一些位置比较点，在运动到达比较点时按设定的 IO 输出规则控制 IO 输出，可用于平面定点拍照等场合。

（10）DMC3000 系列支持多种单轴高速位置比较输出功能，在强化单点位置比较输出的基础上新增了缓冲队列式以及线性增量式的多点高速位置比较输出功能。缓冲队列式位置比较支持添加多个任意的比较位置，根据先进先出的原则依次进行比较输出。线性增量式位置比较支持按线性变化的多个比较位置，位置间时间间隔最小可达几微秒。

（11）DMC3000 系列支持多种高速位置锁存功能，在强化单次锁存功能的基础上新增了连续锁存

功能以及高速触发急停锁存功能。连续锁存可实现对多个位置依次进行高速锁存，结合高速比较输出可以实现多个位置精确检测功能。高速触发急停可以实现在接收到触发信号时锁存当前位置并在设定的时间内停止发脉冲这种特殊应用的精确定位功能。

（12）DMC3000 系列支持原点位置锁存功能，可实现精确回机械原点的功能。

（13）DMC3000 系列支持轴 IO 映射配置功能，支持将轴信号配置到任意一个硬件输入口，如：可将限位接口当原点信号。该功能可减少现场接线、换线的困难。

（14）DMC3000 系列 API 函数接口标准化，可实现本系列内任意替换控制卡而无需修改应用软件，方便用户根据不同的应用需求择优选择不同的控制卡。

（15）DMC3000 系列可配套 CAN-IO 扩展模块，以满足需求较多 IO 端口的设备。

（16）DMC3000 系列支持多达 255 个字符的密码设置及校验功能，该功能可实现控制卡的应用软件与硬件的绑定，有效的保护客户开发的系统软件。

1.2 DMC3000 系列卡主要技术指标

表 1.1 DMC3000 系列卡主要技术指标

技术指标 卡类型	DMC3C00 DMC3C00-PCIe	DMC3800 DMC3800-PCIe	DMC3600 DMC3600-PCIe	DMC3400A DMC3400A-PCIe
电机轴数	12	8	6	4
支持在PC机中同时工作的卡数	8			
控制电机的脉冲信号频率范围	1 Hz～4 MHz			
控制电机的脉冲信号频率精度	1 Hz			
脉冲信号输出最大电流	20 mA （吸入）			
脉冲信号长度	28位有符号			
直线插补精度	±0.8 pulse			
圆弧插补精度	±1.5 pulse			
支持的插补坐标系个数	2			
编码器信号输入个数	8	8	6	4
编码器计数器长度	28位有符号			
编码器输入信号频率	4 MHz （4倍频后为16MHz）			
手轮输入信号最大频率	500 kHz			
通用数字输入口数量	16（可扩展）			

技术指标 卡类型	DMC3C00 DMC3C00-PCIe	DMC3800 DMC3800-PCIe	DMC3600 DMC3600-PCIe	DMC3400A DMC3400A-PCIe
通用数字输出口数量	16（可扩展）			14（可扩展）
通用数字输入口	光电隔离，RC滤波			
通用数字输入口输入电流	5～10 mA			
通用数字输入口最高响应频率	4 kHz			
通用数字输出口	光电隔离，集电极开路			
通用数字输出口最大电流	500 mA（5～24Vdc，吸入）			
CAN-IO扩展	最多支持连接8个CAN-IO扩展模块			
高速位置锁存输入口数量（LTC）	2			1
高速位置比较输出口数量（CMP）	4			2
机械正负限位输入口数量（±EL）	24	16	12	8
机械原点信号输入口数量（ORG）	12	8	6	4
伺服到位信号输入口数量（INP）	8	8	6	4
伺服报警信号输入口数量（ALM）	8	8	6	4
伺服准备好信号输入口数量（RDY）	8	8	6	4
伺服使能信号输出口数量（SEVON）	8	8	6	4
伺服误差清除信号输出口数量（ERC）	8	8	6	4
工作温度	0～50 ℃			
贮存温度	-20～80 ℃			
湿度	5～85 %，非结露			
PCI总线插槽电源（输入）	+5VDC±5% ，最大1100 mA			
外部电源（输入）	24VDC ，10A			
尺寸大小（mm）	182.00(长)×106.68(高)			
驱动程序	支持Windows XP、Windows 7、Ubuntu操作系统			
运动控制函数库	支持C/C++、C#、VB6.0、VB.NET、LabVIEW等多种语言			
调试软件	免费提供Motion调试软件			

1.3 DMC3000 系列运动控制卡的典型应用

DMC3000系列运动控制卡已广泛应用于各行各业自动化设备中。主要设备有：

- 电子产品加工、装配设备，如：丝印机、贴片机、PCB钻孔机等
- 激光加工设备，如：激光打标机、激光切割机等
- 机器视觉及自动检测设备，如：影像测量仪、电路板自动检测设备等
- 生物、医学自动采样、处理设备
- 专用工业机器人

1.3.1 DMC3400A 卡的典型应用

图 1.1 为 DMC3400A 运动控制卡组成的 8 轴运动控制系统的典型结构图。

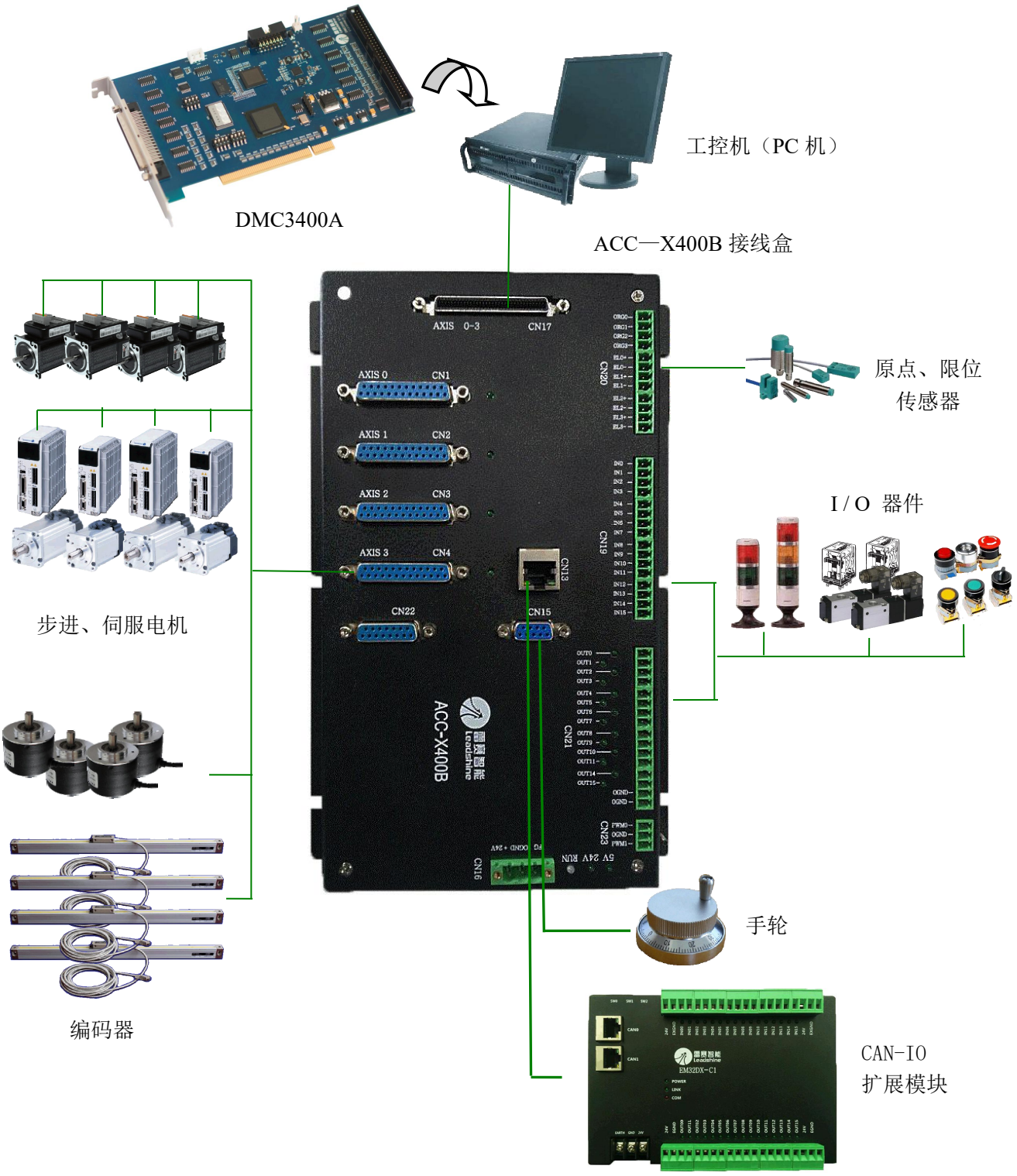


图1.1　DMC3400A运动控制卡组成的运动控制系统结构图

1.3.2 DMC3800 卡的典型应用

图 1.2 为 DMC3800 运动控制卡组成的 8 轴运动控制系统的典型结构图。

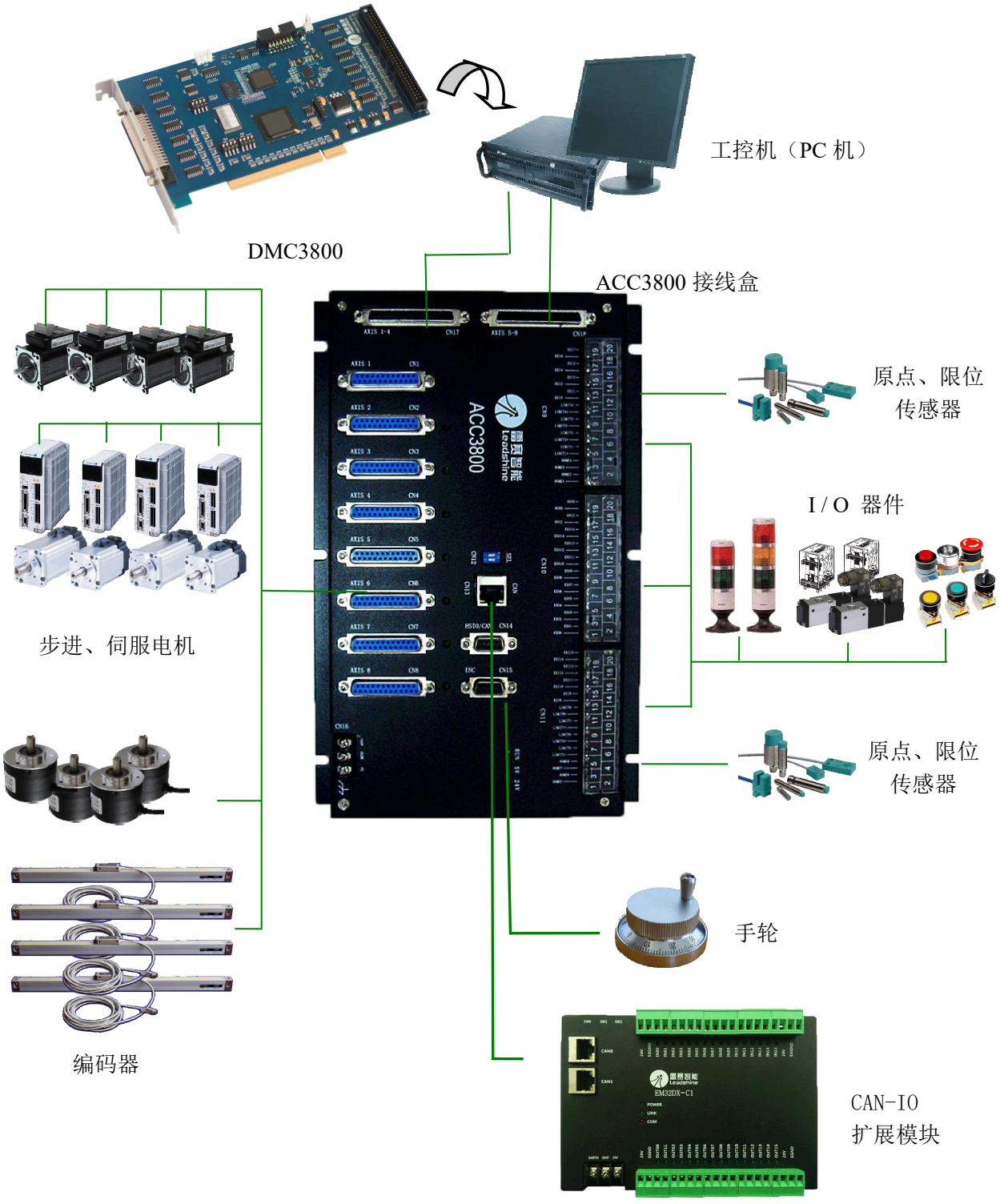


图1.2　DMC3800运动控制卡组成的运动控制系统结构图

1.3.3　DMC3600/3C00 卡的典型应用

使用 DMC3600/3C00 卡组成的运动控制系统与使用 DMC3800 卡组成的运动控制系统类似，可参考图 1.2。

1.4　订货信息

表 1.1　DMC3400A 运动控制卡及主要配件订货代码

名 称	订货代码	说 明
运动控制卡DMC3400A	8.0.03.3400A.011-000	必选
ACC-X400B 接线盒	8.3.00.X4000.001-000	必选
68芯电缆线CABLE68-NR-20（2米）	1.4.2.6868039-00	必选
CAN-IO扩展模块		选配

表 1.2　DMC3600 运动控制卡及主要配件订货代码

名 称	订货代码	说 明
运动控制卡DMC3600	8.0.03.36000.000-000	必选
接线盒ACC3600	8.3.03.36000.000-000	必选
转接板ACC64TO68组合件	8.3.68.68000.001-000	必选
68芯电缆线CABLE68-NR-20（2米）	1.4.2.6868039-00	必选
CAN-IO扩展模块		选配

表 1.3　DMC3800 运动控制卡及主要配件订货代码

名 称	订货代码	说 明
运动控制卡DMC3800	8.0.03.38000.000-000	必选
接线盒ACC3800	8.3.03.38000.000-000	必选
转接板ACC64TO68组合件	8.3.68.68000.001-000	必选
68芯电缆线CABLE68-NR-20（2米）	1.4.2.6868039-00	必选
CAN-IO扩展模块		选配

表 1.4　DMC3C00 运动控制卡及主要配件订货代码

名 称	订货代码	说 明
运动控制卡DMC3C00	8.0.03.3C000.000-000	必选
ACC-XC00 接线盒	8.3.00.XC000.000-000	必选
电缆线HPCN68P转MINI68P-20 1-35双绞线（2米）	1.4.2.6868002-00	必选

CAN-IO扩展模块		选配

表 1.5 DMC3C00-PCIe 运动控制卡及主要配件订货代码

名称	订货代码	说 明
运动控制卡DMC3C00-PCIe		必选
ACC2-XC00 接线盒		必选
电缆线HPCN68P转MINI68P-20 1-35双绞线（2米）	10.25.03.010570	必选
CAN-IO扩展模块		选配

表 1.6 DMC3800-PCIe 运动控制卡及主要配件订货代码

名称	订货代码	说 明
运动控制卡DMC3800-PCIe		必选
接线盒ACC2-3800		必选
转接板ACC64TO68组合件	80.15.99.011250	必选
68芯电缆线CABLE68-NR-20（2米）	10.25.03.010650	必选
CAN-IO扩展模块		选配

表 1.7 DMC3600-PCIe 运动控制卡及主要配件订货代码

名称	订货代码	说 明
运动控制卡DMC3600-PCIe		必选
接线盒ACC2-3600		必选
转接板ACC64TO68组合件	80.15.99.011250	必选
68芯电缆线CABLE68-NR-20（2米）	10.25.03.010650	必选
CAN-IO扩展模块		选配

表 1.8 DMC3400A-PCIe 运动控制卡及主要配件订货代码

名称	订货代码	说 明
运动控制卡DMC3400A-PCIe		必选
ACC2-X400B 接线盒		必选
68芯电缆线CABLE68-NR-20（2米）	10.25.03.010650	必选

CAN-IO扩展模块		选配

1.5 产品图片

运动控制卡 DMC3800、DMC3600、DMC3400A 以及 DMC3C00 卡的外观示意图如图 1.3、1.4 所示，其中 DMC3800 卡与 DMC3600、DMC3400A 卡的外观基本相同。

图1.3 DMC3800/3600/DMC3400A运动控制卡外观示意图

图1.4 DMC3C00动控制卡外观示意图

运动控制卡 DMC3400A-PCIe、DMC3600-PCIe、DMC3800-PCIe 及 DMC3C00-PCIe 卡的外观示意图如图 1.7、1.8 所示。其中 DMC3800-PCIe 卡与 DMC3600-PCIe、DMC3400A-PCIe 卡的外观相同。

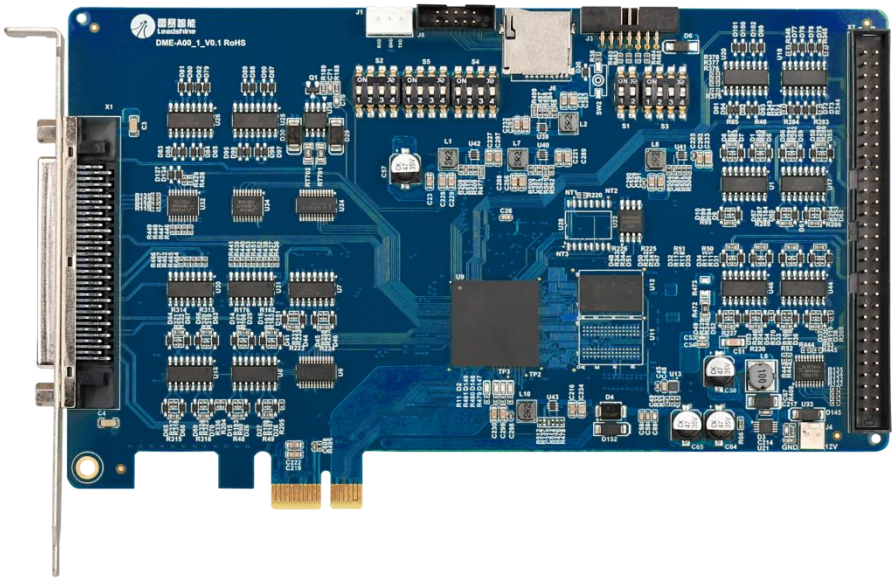


图1.7　DMC3800-PCIe/3600-PCIe/3400A-PCIe运动控制卡外观示意图

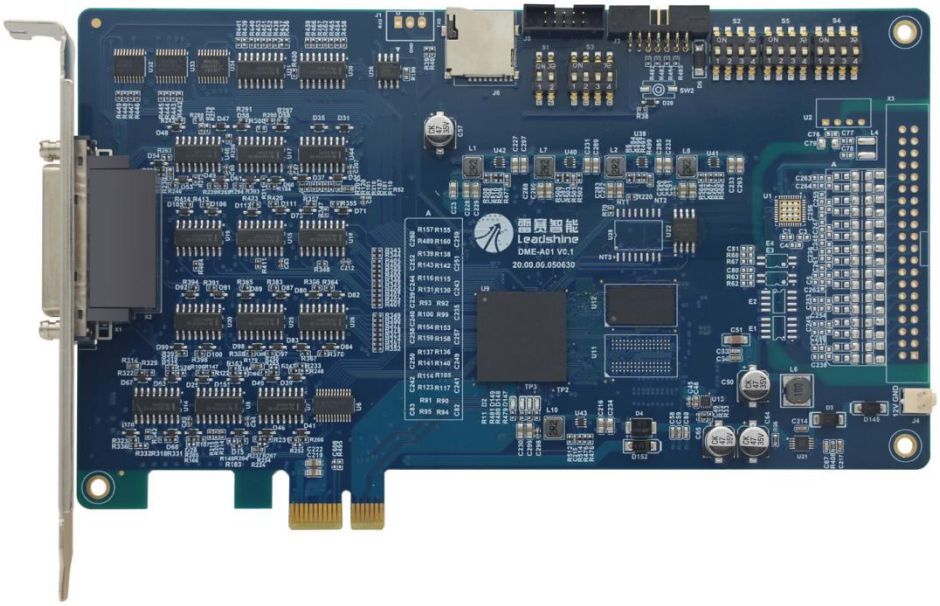


图1.8　DMC3C00-PCIe运动控制卡外观示意图

第 2 章　DMC3000 系列卡功能介绍

雷赛控制技术 DMC3000 系列运动控制卡是一款新型的 PCI/PCIe 总线运动控制卡。可以控制多个步进电机或数字式伺服电机；适合于多轴点位运动、插补运动、轨迹规划、手轮控制、编码器位置检测、IO 控制、位置比较、位置锁存等功能的应用。

DMC3000 系列卡的运动控制函数库功能丰富、易学易用，用户开发应用软件十分方便。随卡免费提供的雷赛控制卡 Motion 调试软件，不但可以演示 DMC3000 系列卡的控制功能，而且可用于控制卡及运动控制系统的硬件测试。

2.1　运动控制功能

2.1.1　点位运动

点位运动是指：运动控制器控制运动平台从当前位置开始以设定的速度运动到指定位置后准确地停止。

点位运动只关注终点坐标，对运动轨迹的精度没有要求。点位运动的运动距离由脉冲数决定，运动速度由脉冲频率决定。

PC 机执行点位运动指令，即调用点位运动函数，并自动将运动参数通过 PCI 总线接口传送至 DMC3000 系列运动控制卡，使其按设定的速度输出脉冲；当输出脉冲数等于命令脉冲数时，DMC3000 系列卡停止脉冲输出。位移与时间的关系如图 2.1 所示。

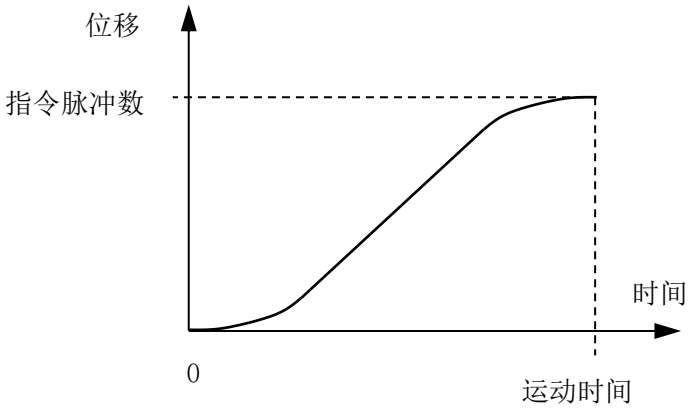


图2.1　定长运动位移曲线

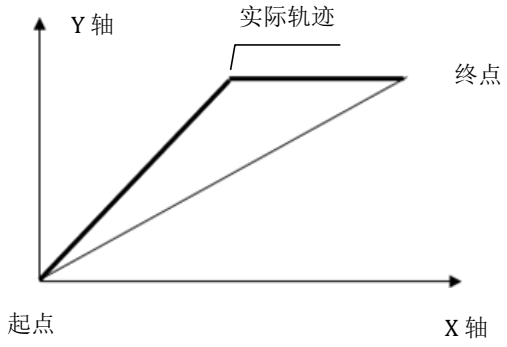


图2.2　两轴复合运动的轨迹

多轴同时做点位运动，称之为多轴联动。由于软件处理速度远比机械系统响应速度快，虽然多条运动指令连续发出，需几个微秒，可对机械系统而已，可认为是同时启动。如果每个轴的运动速度相同，则多轴运动的轨迹就可能是折线，如图 2.2 所示。

如果从起点到终点都需要按照规定的路径运动，就必须采用直线插补或圆弧插补功能。

2.1.1.1 梯形速度控制

为了让平台在运动过程中能平稳加速、准确停止，一般采用梯形速度曲线控制运动过程，如图 2.3 所示。即：电机以起始速度开始运动，加速至最大速度后保持速度不变，结束前减速至停止速度，并停止。运动的距离由点位运动指令决定。

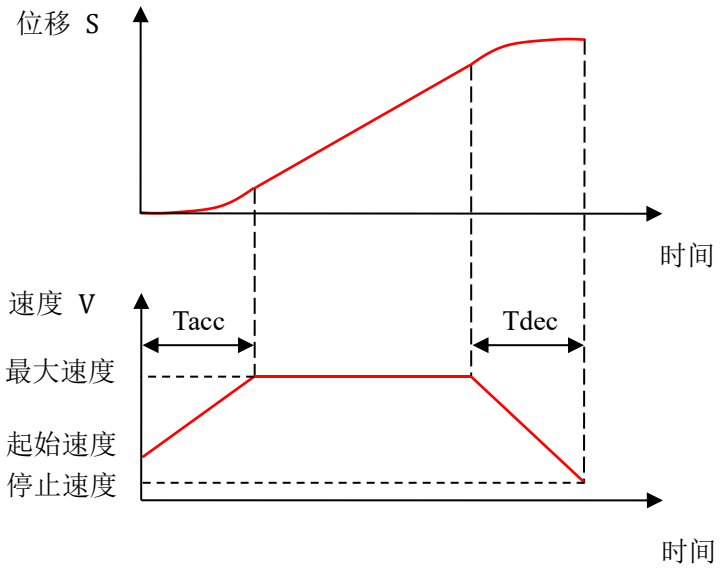


图2.3 梯形速度曲线及对应的位移曲线

Tacc：总加速时间， Tdec：总减速时间

2.1.1.2 S 形速度控制

为改善平台运动的平稳性，DMC3000 系列运动控制卡还提供了 S 形速度控制曲线。

在 S 形速度控制过程中，指令脉冲频率从一个内部设定的速度快速加速到起始速度，然后作 S 形加速运动；运动结束前，指令脉冲频率作 S 形减速运动到停止速度，然后再快速减速到一个内部设定的速度，这时脉冲输出停止，如图 2.4 所示。

2.1.1.3 运动中改变终点位置

在进行点位运动的过程中，DMC3000 系列运动控制卡可以改变运动的终点位置，且位置可增可减。如图 2.5 所示，原本点位运动的距离是 50000 个脉冲；但当运动了 550ms 时，将终点改为 100000 个脉冲，电机从减速变为加速，继续向前运动，然后减速停在新的终点位置。

该功能在固晶机上使用十分方便。首先，取料臂开始向一个理想位置运动；摄像头检测晶片的实际位置；然后给出终点的修整位置；取料臂向新的位置运动。这样可以大大提高设备的生产效率。

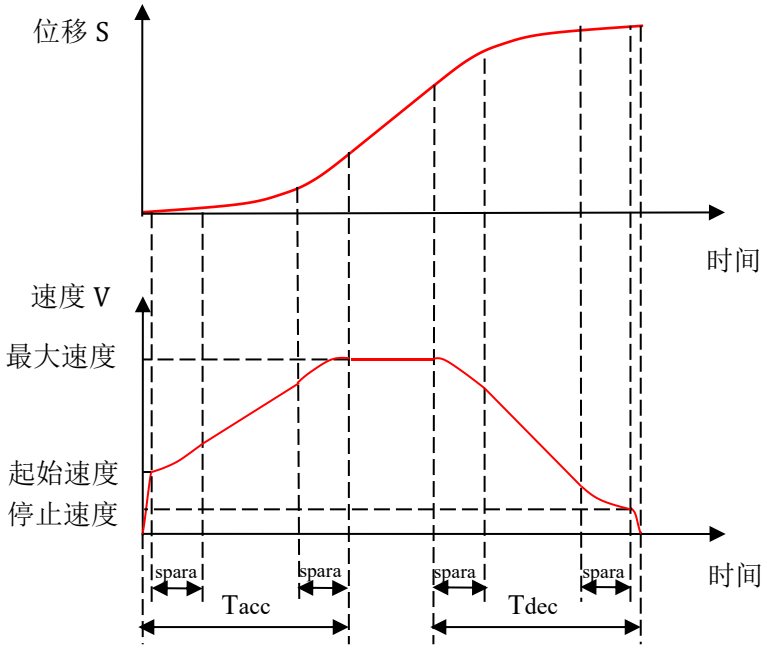
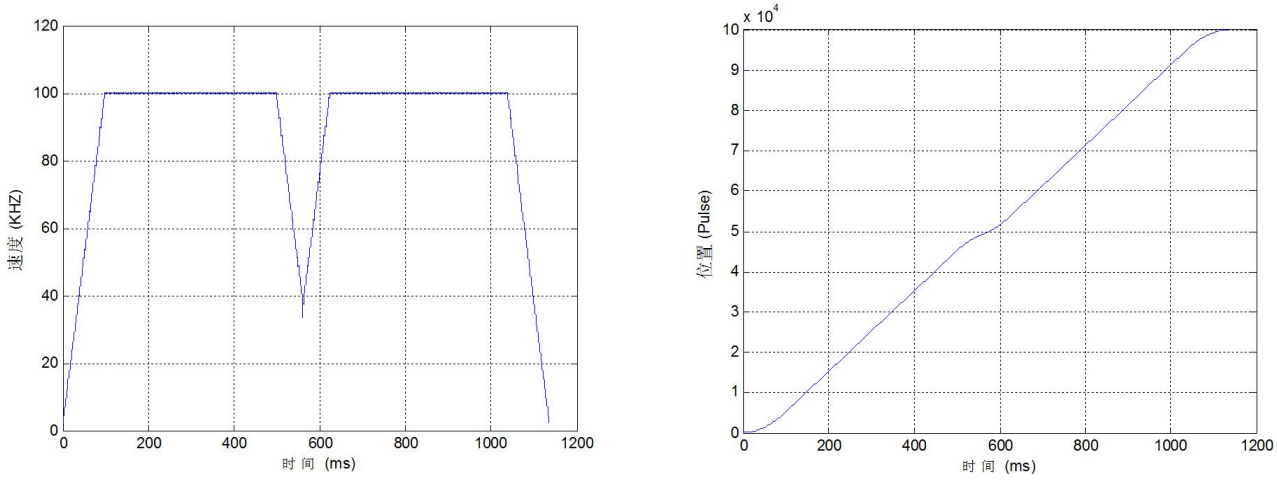


图2.4　S形速度曲线及对应的位移曲线

spara：S 段时间，Tacc：总加速时间， Tdec：总减速时间



(a)　速度曲线　　(b)　位移曲线

图2.5　DMC3000系列运动控制卡改变终点位置的过程

2.1.1.4　运动中改变当前速度

在点位运动（或连续运动）过程中，DMC3000 系列运动控制卡可以改变运动的速度，且速度变化过程和预设的梯形速度曲线或 S 型速度曲线相同，如图 2.6 所示。

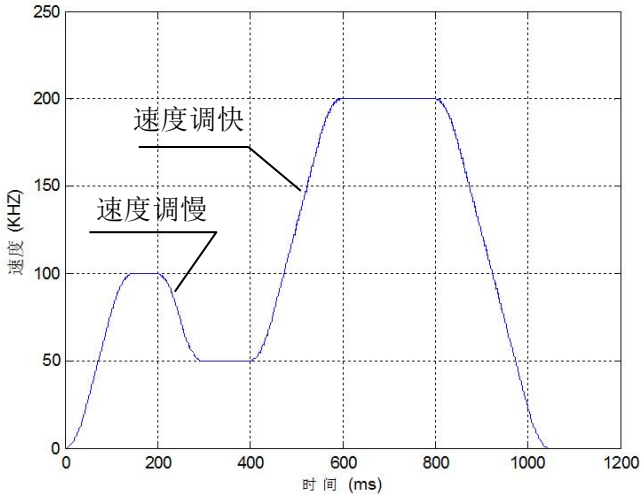


图2.6　DMC3000系列运动控制卡的调速过程

2.1.2　连续运动

连续运动是指：电机从起始速度开始运行，加速至最大速度后连续运动；只有当接收到停止指令或外部停止信号后，才减速停止。

连续运动指令其实就是速度控制指令，国外运动控制器将此指令称为 JOG 指令。

DMC3000 系列运动控制卡可以控制电机以梯形或 S 形速度曲线在指定的加速时间内从起始速度加速至最大速度，然后以该速度连续运行，直至调用停止指令或者该轴遇到限位信号、急停信号才会按启动时的速度曲线减速停止。连续运动指令的速度与时间曲线如图 2.7 所示。

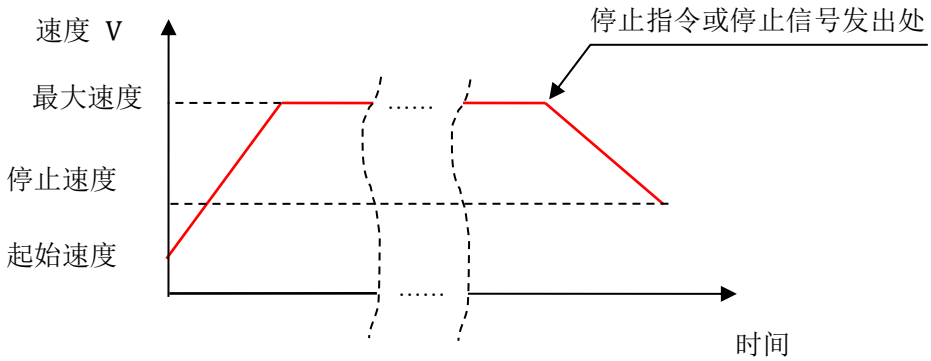


图2.7　连续运动速度曲线

该功能的主要用途是：速度控制，如：传送带的速度、包装机连续送料速度等。

2.1.3 直线、圆弧插补运动

为了实现轨迹控制，运动控制卡按照一定的控制策略控制多轴联动，使运动平台用微小直线段精确地逼近轨迹的理论曲线，保证运动平台从起点到终点上的所有轨迹点都控制在允许误差范围内。这种控制策略称为插补算法，因此轨迹运动通常称为插补运动。插补运动有许多种类，如：直线插补、圆弧插补、螺旋线插补、抛物线插补等。

DMC3000 系列运动控制卡采用积分法进行直线插补和圆弧插补。图 2.8 为直线插补的实例。理想轨迹为原点至点 Z 的直线；图中 7 条带箭头的线段为平台实际运动轨迹。

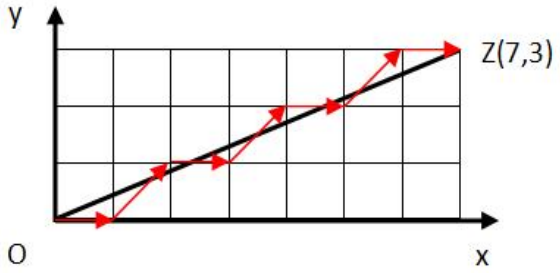


图2.8 积分法直线插补实例

3400A 卡可以进行 2 至 4 轴的直线插补运动。
DMC3C00 卡可以进行 2 至 12 轴的直线插补运动。
DMC3800 卡可以进行 2 至 8 轴的直线插补运动。
DMC3600 卡可以进行 2 至 6 轴的直线插补运动。

在进行圆弧插补之前，必须确定圆弧轨迹的参数。如果只知道圆弧的起点、终点、半径，那么就有 4 条轨迹可选，如图 2.9 所示。顺时针方向圆弧轨迹称为顺圆；逆时针方向的圆弧轨迹称为逆圆。只有获得圆弧轨迹方向、圆心坐标或者知道圆弧角是小于 180 度还是大于 180 度，才能确定唯一的圆弧插补轨迹。如图 2.10 所示，一段起点为 Q（7，0），终点为 Z（0，7），半径为 7（单位为脉冲），圆心在原点的逆园轨迹。积分法计算的插补结果如图 2.10 中 10 条带箭头的线段所示。DMC3000 系列运动控制卡只能在任意 2 轴间的坐标平面内进行圆弧插补运动。

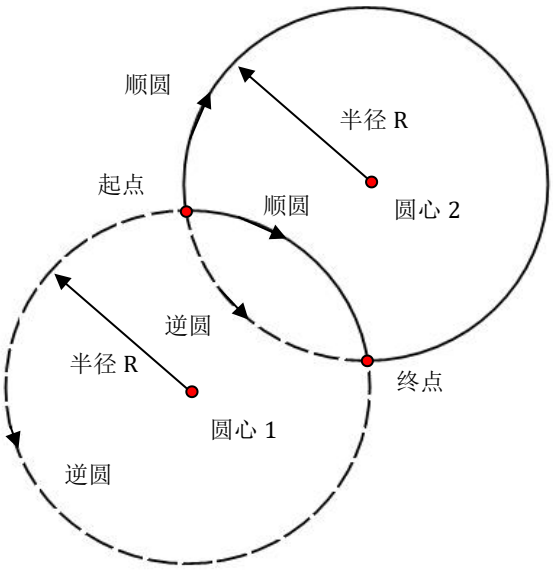


图2.9 圆弧插补轨迹

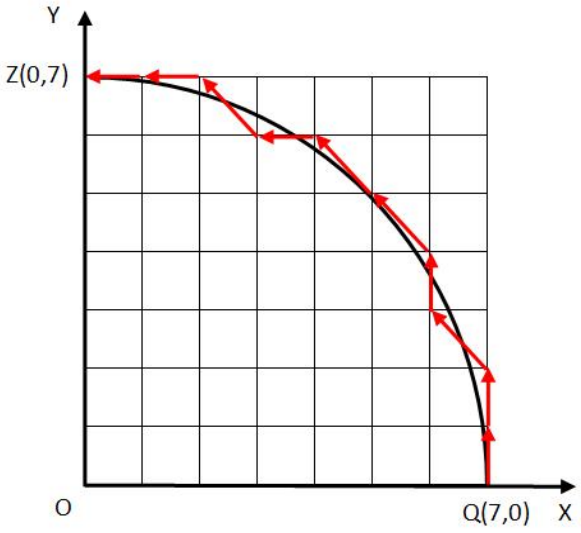


图2.10 圆弧插补实例

2.1.4 PVT 运动功能

DMC3000 系列运动控制卡共有四种 PVT 模式，分别为 PTT、PTS、PVT、PVTS 模式。其中 PTT、PTS 运动模式用于单轴速度规划；PVT、PVTS 运动模式则用于多轴轨迹规划。用户可以根据实际需求选择适合的 PVT 模式。

2.1.4.1 单轴速度规划功能

DMC3000 系列运动控制卡中提供了两种 PVT 模式来实现单轴速度规划，分别为 PTT 运动模式和 PTS 运动模式。

1. PTT 运动模式

PTT 模式中第一个字母 P 表示位置（Position）、第二个字母 T 表示时间（Time），最后一个字母 T 表示梯形（Trapezoid）；PTT 模式表示在梯形速度曲线下规划点位运动。

用户通过输入一系列位置和时间参数，自定义单轴的运动规律。首先将速度曲线分割成若干段，如图 2.11 所示。

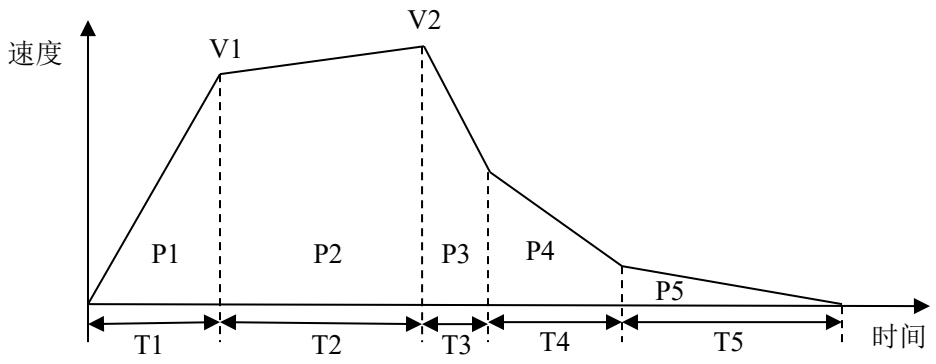


图 2.11 PTT 运动速度曲线

整个速度曲线被分割成 5 段。

第 1 段起点速度为 0；经过时间 T1 后，位移量为 P1，即速度曲线所围面积。因此第 1 段的终点速度为 V1=2×P1/T1；

第 2 段起点速度为 V1，经过时间 T2 后，位移量为 P2，即速度曲线所围面积。因此第 2 段的终点速度为 V2=2×P2/T2-V1；

第 3、4、5 段依此类推。

在定义一段完整的 PTT 运动时，第 1 段的起点位置和时间被设为 0；各段的终点位置和时间都是相对于该段起点的相对值。位置单位为脉冲，时间单位为秒。

运动控制卡执行 PTT 运动时，根据用户定义的各段位移和时间参数，计算各点的速度，形成

一条连续的速度曲线。

2. PTS 运动模式

PTS 运动模式是 PTT 的扩展功能模式。PTS 的最后一个字母 S 表示 S 形速度曲线；PTS 模式是在 S 形速度曲线下规划点位运动；和 PTT 模式相比，其各段速度过渡更加平滑。

用户通过输入一系列位置、时间、百分比参数，自定义单轴的运动规律。其中位置、时间参数定义和 PTT 模式相同；“百分比”参数是指：相邻 2 个数据点之间加速度的变化时间占速度变化时间的百分比。

以图 2.12 为例说明之。数据点 P2 和 P3 之间加速度不变，因此数据点 P2 的百分比为 0。数据点 P3 和 P4 之间加速度变化时间为 2×ta，速度变化时间为 2×ta+te，因此数据点 P3 的百分比为[2×ta/(2×ta+te)]×100%。

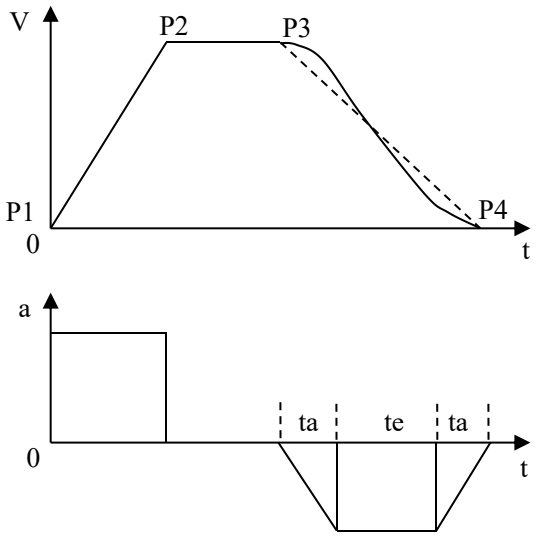


图 2.12 加速度变化时间百分比定义

调整百分比参数，可以改变 S 形速度曲线的形状。如图 2.12 所示，当数据点 P3 的百分比为 0 时，数据点 P3 和 P4 之间的速度曲线为直线（如图 2.12 中虚线所示）；当数据点 P3 的百分比不为 0 时，数据点 P3 和 P4 之间的速度曲线为 S 形曲线（如图 2.12 中实线所示）。“百分比”参数越大，S 段曲线越长。

2.1.4.2 多轴高级轨迹规划功能

当直线插补、圆弧插补不能满足轨迹规划的需求时，可以使用 DMC3000 系列运动控制卡的两种高级 PVT 运动功能。

DMC3000 系列运动控制卡提供的两种多轴轨迹规划的功能分别为 PVT、PVTS 运动模式；第二

字母 V 表示速度（Velocity），最后一个字母 S 表示平滑。

PVT 模式用于对各点的位置、速度、时间都有要求的轨迹规划；PVTS 模式用于只对各点的位置、时间有要求，而对各点的速度无严格要求的轨迹规划。

1. PVT 运动模式

PVT 模式使用一系列数据点的位置、速度、时间参数自定义运动规律。

DMC3000 系列卡采用 3 次样条插值算法对位置、速度和时间参数进行曲线拟合。即位置、速度曲线满足 3 次多项式函数关系。

位移方程为： $p = at^3 + bt^2 + ct + d$

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

如果给定轨迹上的一组“位置、速度、时间”参数，即可用 3 次样条函数逼近该轨迹。如图 7.21 所示为采用 PVT 模式拟合平面椭圆的轨迹图。

注意：每个点的位置与速度参数需要仔细设计，否则难以得到理想的运动轨迹。

2. PVTS 运动模式

PVTS 运动模式是 PVT 模式的简化模式。PVTS 运动模式只需要定义各数据点的位置、时间参数，以及起点速度和终点速度。运动控制卡根据各数据点的位置、时间参数计算运动轨迹的速度，确保各数据点速度连续和加速度连续。

由于对各点速度没有特殊要求，因此使用 PVTS 运动模式可以得到更平滑的运动轨迹。如图 7.23 所示为采用 PVTS 模式拟合空间圆弧的轨迹图。

2.1.5 回原点运动

在运动平台上，每个轴都有一个位置传感器用于设置一个位置参考点，即原点位置，以便进行位置控制。在正常运动之前，都需要用回零指令控制平台向原点方向运动，当控制卡检测到原点信号 ORG 后，平台自动停止，并将停止位置作为该轴的原点。参见图 2.13。

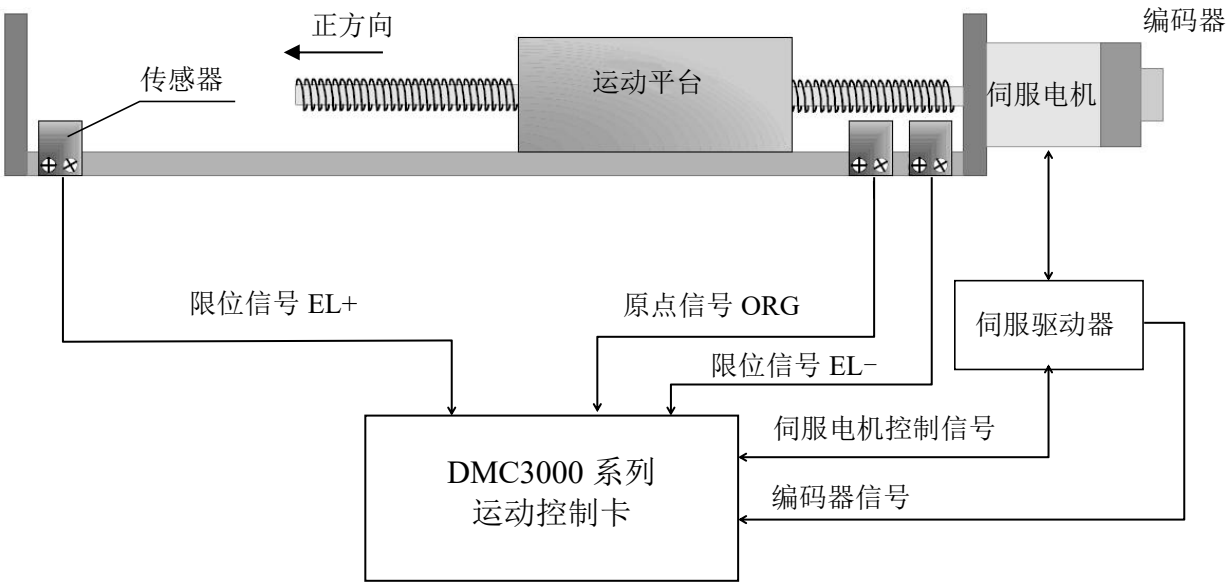


图2.13　运动平台传感器信号及电机控制信号与运动控制卡的关系

2.1.6　异常减速停止时间设置功能

DMC3000 系列卡支持异常减速停止时间设置功能，异常减速停止包括命令减速停止、硬限位减速停止、软限位减速停止、IO 触发减速停止等，用户根据现场实际需求情况设定减速停止时间可达到理想的减速效果。

2.1.7　手摇脉冲发生器的控制

为了让用户能在手动模式下方便地调整机器的位置，DMC3000 系列运动控制卡提供了手摇脉冲发生器（简称：手轮）控制功能。用户操控接在 DMC3000 系列卡上的手摇脉冲发生器（参见图 1.1、图 1.2、图 2.14），手摇得慢，电机就转动得慢；手摇得快，电机就转动得快。

该功能多用于加工轨迹起点调整、刀具对位等工作中。

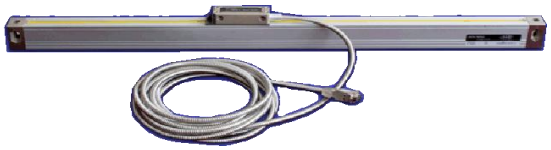
图2.14　手持式手摇脉冲发生器外形

2.2 编码器位置检测

DMC3000 系列卡每轴都有一个编码器输入接口用于检测平台的位移或电机的转角。编码器有 EA、EB、EZ 三个信号，脉冲计数信号由 EA 和 EB 端口输入；它可以接收两种类型的脉冲信号：正负脉冲输入或 A/B 相正交信号；EZ 信号是编码器零位信号。编码器外形如图 2.15 所示。

采用探针和编码器配合使用，DMC3000 系列卡通过位置触发功能，可完成对工件的位置检测工作，如图 2.16 所示。即：当探针接触到工件时，产生一个触发信号；DMC3000 系列卡接受该信号后，立即将编码器当前位置记录下来；通过记录工件的一系列数据，然后再通过软件处理，即可获得该工件的外形尺寸。

旋转编码器



光栅尺

图2.15 编码器外形

图2.16 被测工件与探针

2.3 专用 IO 和通用 IO 控制

DMC3000 系列运动控制卡除了运动控制功能外，还提供了数字式输入信号（Input）和输出信号（Output）即 IO 信号的控制功能。专用 IO 信号用于原点、限位、伺服电机等控制，有专用控制指令相对应。

2.3.1 原点信号和限位信号

常用的专用 IO 信号有：运动平台上用于正向行程限位的传感器信号 EL+、反向行程限位的传感器信号 EL-；以及用于平台定位的原点传感器信号 ORG 等。参见图 2.13。

2.3.2 伺服电机控制信号

设备中有交流伺服电机时，使用 DMC3000 系列卡上的伺服电机控制信号 SEVON、RDY、ALM、

INP 和 ERC 就显得十分方便。

SEVON 是控制卡输出给伺服电机驱动器的控制信号，当 SEVON 信号为无效状态时，伺服驱动器不使能，电机处于自由状态；当 SEVON 信号有效时，伺服驱动器使能，电机锁紧；等待指令脉冲信号。

RDY 是伺服电机驱动器发给控制卡的状态信号，当 RDY 信号为有效时，表示伺服驱动器已经准备好，控制卡接收到该信号后就可以向伺服驱动器发出运动命令；如果 RDY 信号无效，表示伺服驱动器还未准备好，这时控制卡发出指令脉冲信号，伺服驱动器也不会运动。

ALM 信号是从伺服电机驱动器发给控制卡的状态信号，用来报告伺服驱动器或电机出错。控制卡接收到 ALM 信号时，将立即停止发出脉冲，该过程是一个硬件处理过程。

INP 信号是从伺服电机驱动器发给控制卡的状态信号，告知运动控制卡伺服电机已经停止。

伺服电机驱动器通常有一个位置偏差计数器，记录指令脉冲和位置反馈脉冲之间的偏差。伺服电机驱动器将控制电机运动使位置偏差趋于 0，但是电机实际位置总是滞后于指令脉冲。所以当运动控制卡的指令脉冲发送完毕时，伺服电机并没有立即停止，而是继续运动，如图 2.17 所示，直到位置偏差趋于 0；这时驱动器将发出一个 INP 信号。

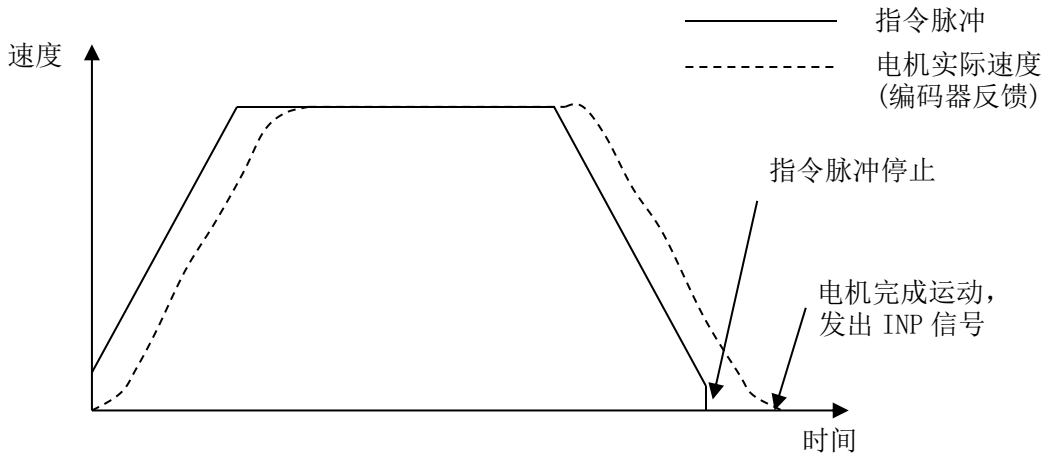


图 2.17　伺服定位完成时的 INP 信号

ERC 信号是控制卡输出给伺服电机驱动器的控制信号。

伺服驱动器依赖电机目标位置（即要求电机达到的位置）和当前位置（即电机已经到达的位置）之间的误差来驱动电机运动，若这个误差为零电机将停止运动。当伺服驱动器接收到运动控制卡发出的 ERC 信号后，会立即清除误差并停止电机运转。

2.3.3　位置比较输出

DMC3000 系列卡提供了位置触发输出信号的函数，包括单轴低速位置比较、单轴高速位置比

较和一组二维低速位置比较。当电机运动到预先设置的位置时，自动触发特定的输出口。该功能在轨迹运动中用于控制点胶阀的开关、触发照相机快门等动作十分方便。

2.3.4 高速位置锁存

DMC3000 系列卡支持多种高速位置锁存功能，包括单次锁存、连续锁存以及锁存触发延时急停功能。连续锁存可实现对多个位置依次进行高速锁存，结合高速比较输出可以实现多个位置精确检测功能。高速触发急停可以实现在接收到触发信号时锁存当前位置并在设定的时间内停止发脉冲这种特殊应用的精确定位功能。

2.3.5 通用 IO 信号

如图 1.1、图 1.2 所示，在电机运动的同时，DMC3000 系列卡还可接受按键、数字式传感器等信号，控制指示灯、继电器、电磁阀等器件。

若自动化设备上有气缸等元件。当设备上电时，一般控制器不会立即输出正确的控制信号，气缸会无规律的运动一次。这样很容易产生人身安全事故，或将气缸上安装的刀具、测量元件等损坏。

为了避免该现象发生，DMC3000 系列卡可用拨码开关设置输出端口的初始电平。确保在上电瞬间，就有正确的控制信号作用在电磁阀上，使气缸不会随意运动。

2.3.6 轴 IO 映射功能

DMC3000 系列卡支持轴 IO 映射配置功能，支持将轴专用 IO 信号配置到任意一个硬件输入口，如：可将限位接口当原点信号。该功能可减少现场接线、换线的困难。

2.3.7 虚拟 IO 映射功能

DMC3000 系列卡支持虚拟 IO 映射功能。通过该功能函数的设置可以实现专用通用 IO 输入接口的滤波功能，并且可以通过专用函数读取该端口滤波后的电平状态。

2.3.8 CAN-IO 扩展模块

当设备需求的 IO 端口数量较多时，DMC3000 系列卡可配套 CAN-IO 扩展模块对通用输入输出端口进行扩展。

2.4 多卡运行

DMC3000 系列卡的驱动程序支持最多 8 块 DMC3000 系列卡同时工作。因此，一台 PC 机可以同时控制多达 96 个电机同时运动。

DMC3000 系列卡支持即插即用模式，用户可不必去关心如何设置卡的基地址和 IRQ 中断值。在使用多块运动控制卡时，首先要用运动控制卡上的拨码开关设置卡号；系统启动后，系统 BIOS 为相应的卡自动分配物理空间。

运动控制卡的编号是 0~7 号。DMC3C00 卡上的 12 个轴的编号为 0~11 号；DMC3800 卡上的八个轴的编号为 0~7 号；DMC3600 卡上的六个轴的编号为 0~5 号；DMC3400A 卡上的四个轴的编号为 0~3 号。在多卡系统中要控制某个电机运动，需要先确定卡号，再确定轴号。

第 3 章　硬件接口电路

3.1　硬件简介

DMC3000 系列运动控制卡兼容 PCI V2.3 标准的 32Bit PCI 标准半长卡规范。硬件接口电路有：4 轴脉冲和方向控制信号、伺服电机控制信号、编码器信号、机械限位与原点信号、手轮脉冲信号以及通用输入输出信号等。具体硬件系统框图如图 3.1 所示。

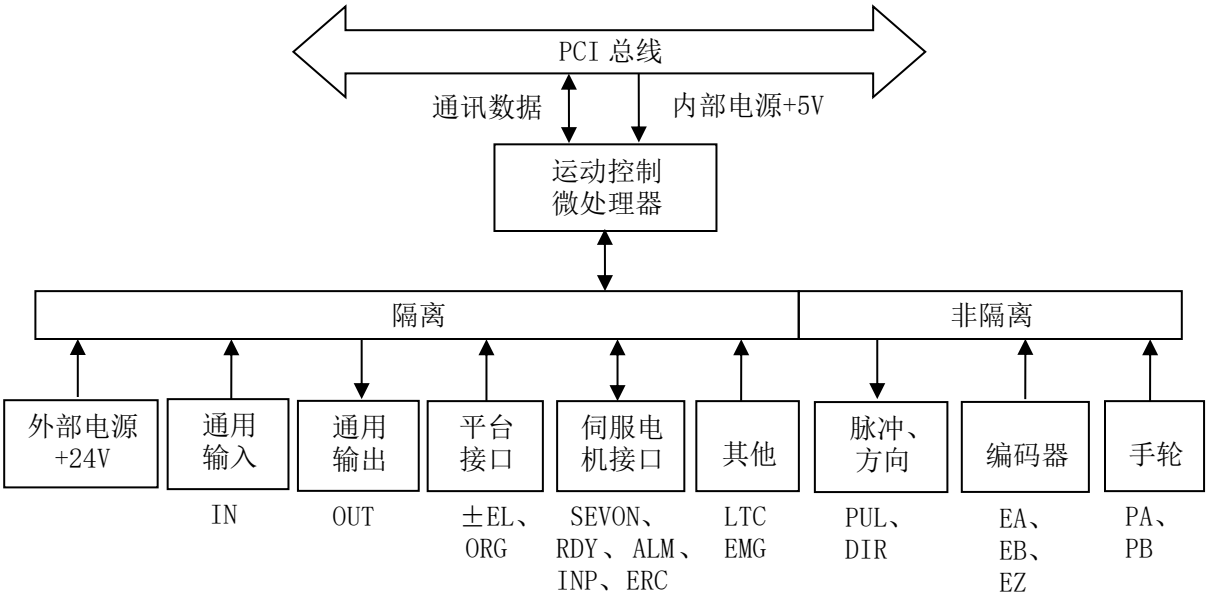


图3.1　运动控制卡硬件系统框图

DMC3000 系列运动控制卡硬件布置及尺寸如图 3.2 所示。

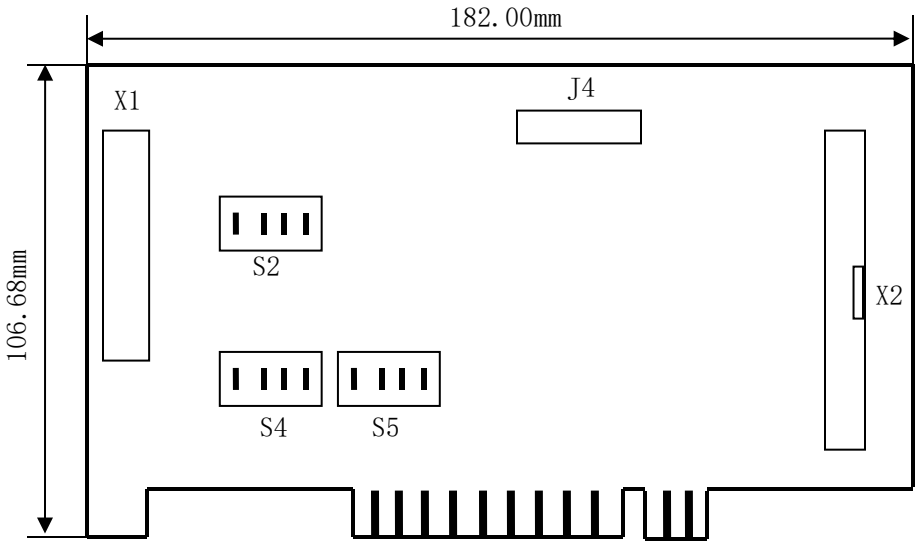


图3.2 DMC3000系列卡硬件布置及尺寸图

3.2 控制卡与配件的连接

3.2.1 DMC3C00 控制卡与配件的连接

DMC3C00 控制卡有一个必配的接线盒 ACC-XC00，连接示意图如图 3.3 所示。

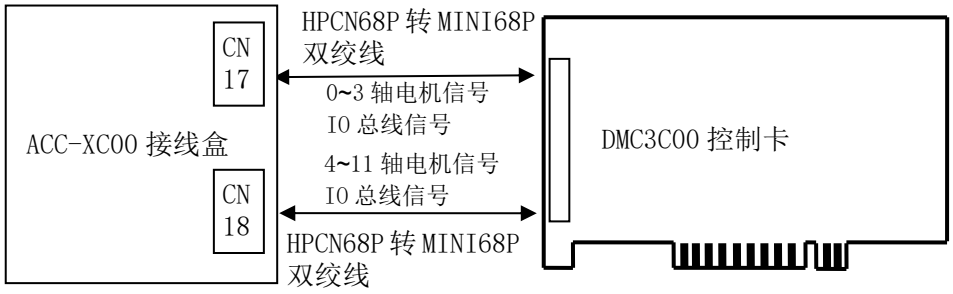


图3.3 DMC3C00与配件连接示意图

3.2.2 DMC3800 控制卡与配件的连接

DMC3800 控制卡有一个必配的接线盒 ACC3800,连接示意图如图 3.4 所示。

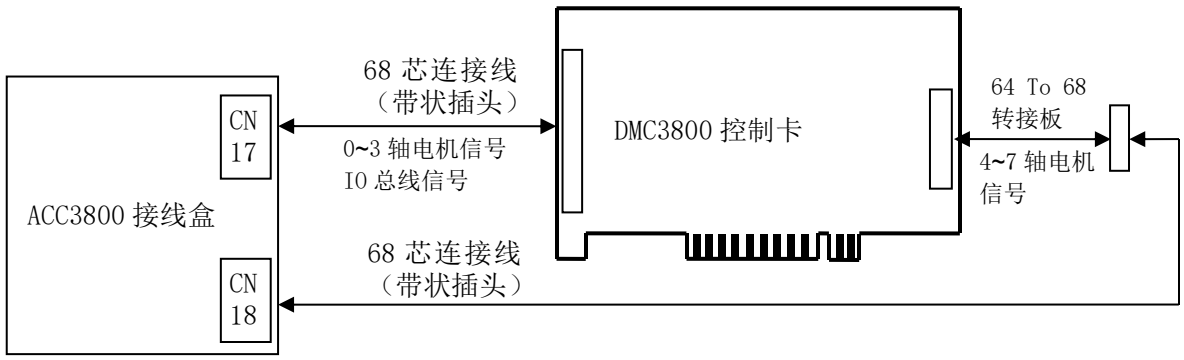


图3.4　DMC3800与配件连接示意图

3.2.3　DMC3600 卡与配件的连接

DMC3600 卡有一个必配的接线盒 ACC3600，连接示意图如图 3.5 所示。

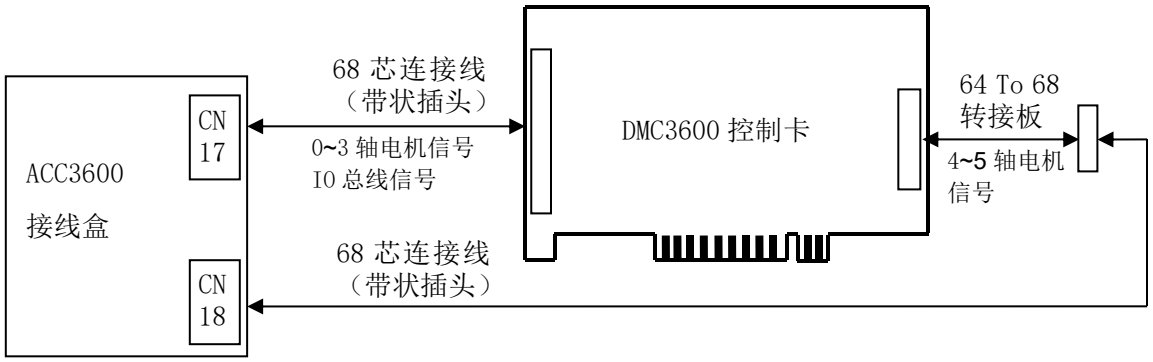


图3.5　DMC3600卡与配件连接示意图

3.2.4　DMC3400A 卡与配件的连接

DMC3400A 卡有一个必配的接线盒 ACC-X400B，连接示意图如图 3.6 所示。

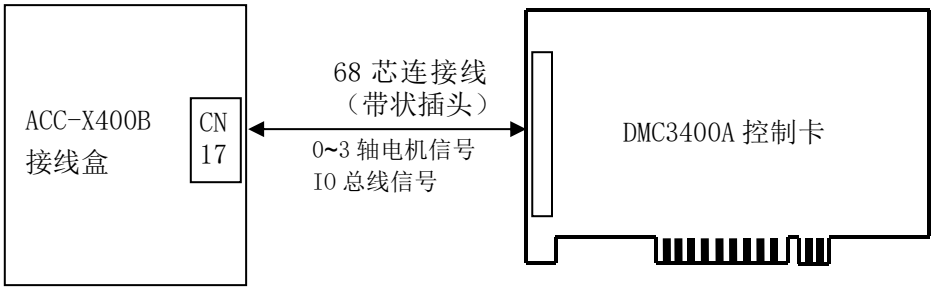


图3.6　DMC3400A卡与配件连接示意图

3.3 电机控制信号接口电路

DMC3000 系列卡提供了两种脉冲量输出模式，一种是脉冲+方向信号模式，另一种是正/负脉冲模式，如图 3.7 和图 3.8 所示。默认情况下，控制卡输出脉冲+方向信号模式。用户可以通过系统配置在这两种模式之间切换。

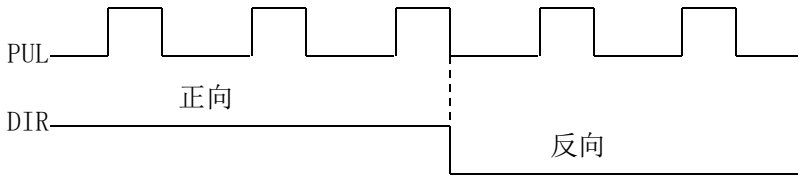


图3.7 单脉冲模式

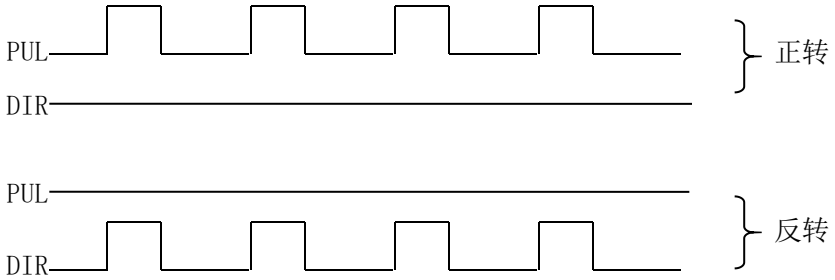


图3.8 双脉冲模式

3.3.1 DMC3C00/3800/3400A 卡的电机控制信号接口电路

DMC3C00 卡有 12 路电机控制信号接口，通过 ACC-XC00 接线盒与外部电机驱动器相连。DMC3800 卡有 8 路电机控制信号接口，通过 ACC3800 接线盒与外部电机驱动器相连。DMC3400A 卡有 4 路电机控制信号接口，通过 ACC-X400B 接线盒与外部电机驱动器相连。

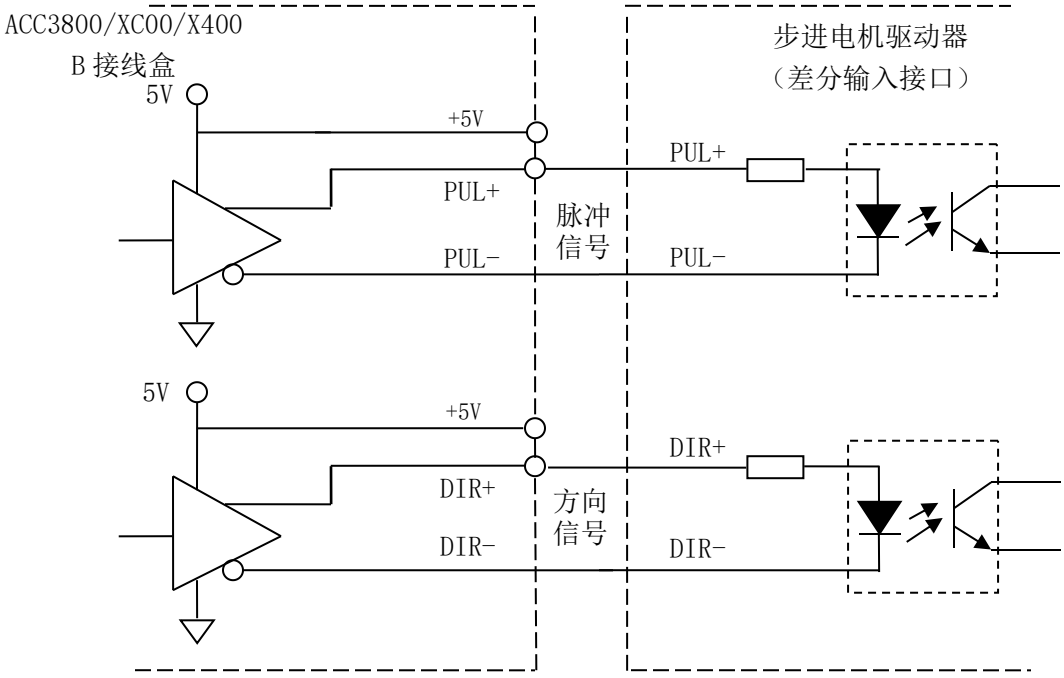


图3.9　ACC3800/X400B/XC00接线盒差分输出方式接线图

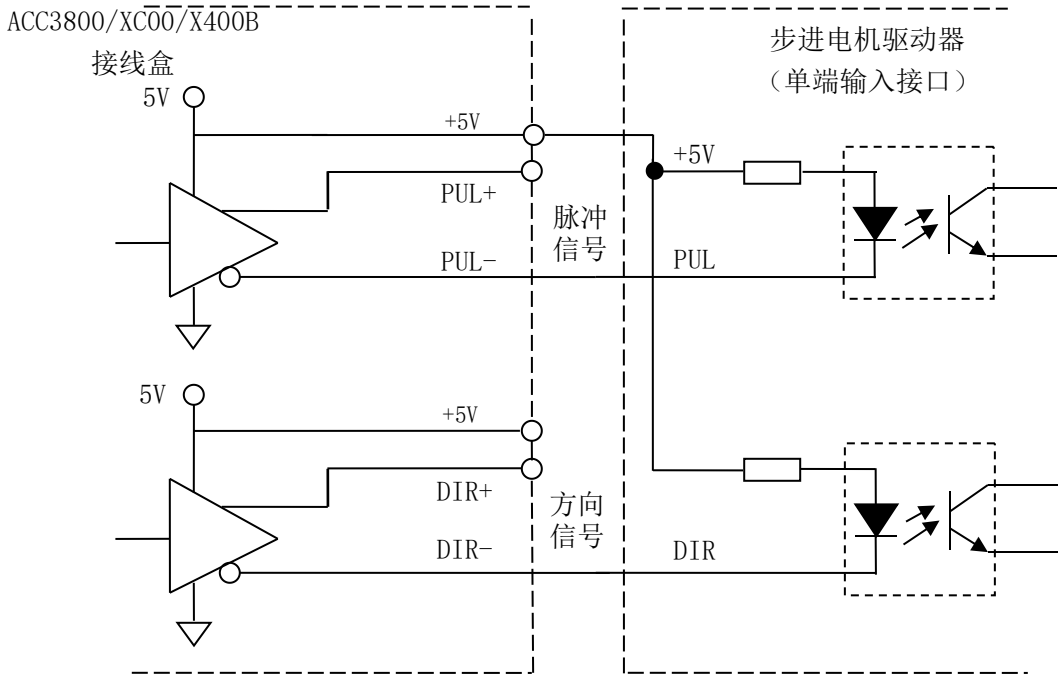


图3.10　ACC3800/X400B/XC00接线盒单端输出方式接线图

接线盒 ACC3800 上的 CN1~CN8 轴控制端口上有提供+5V 的电源信号，接线盒 ACC-X400B 上的 CN1~CN4 轴控制端口上有提供+5V 的电源信号，接线盒 ACC-XC00 上的 CN1~CN 以及 CN25~CN288 轴控制端口上有提供+5V 的电源信号。当电机驱动器为单端输入接口时，可使用+5V 和 PUL-端口输

出脉冲信号，使用+5V 和 DIR-端口输出方向信号；当电机驱动器为差分输入接口时，使用 PUL+和 PUL-输出脉冲信号，使用 DIR+和 DIR-输出方向信号。如图 3.9、3.10 所示。

3.3.2 DMC3600 卡的电机控制信号接口电路

DMC3600 卡有 6 路电机控制信号接口，通用 ACC3600 接线盒与外部电机驱动器相连。

接线盒 ACC3600 上的 CN1~CN6 轴控制端口上有提供+5V 的电源信号。当电机驱动器为单端输入接口时，可使用+5V 和 PUL-端口输出脉冲信号，使用+5V 和 DIR-端口输出方向信号；当电机驱动器为差分输入接口时，使用 PUL+和 PUL-输出脉冲信号，使用 DIR+和 DIR-输出方向信号。

接线盒 ACC3600 的电机控制信号接口电路与接线盒 ACC3800 的相同，关于其接口电路示意图可参考图 3.9~3.10。

3.4 编码器、手摇脉冲发生器接口电路

3.4.1 编码器信号输入接口

3.4.1.1 可接收的编码器信号类型

DMC3000 系列运动控制卡支持 2 种类型的编码器信号输入：非 AB 相脉冲输入和 A/B 相正交信号。

1. 非 AB 相脉冲输入模式

该模式为脉冲+方向模式。在此模式下 EA 端口接收脉冲信号；EB 端口接收方向信号，高电平对应于计数器计数加，低电平对应于计数减。

2. AB 相正交信号输入模式

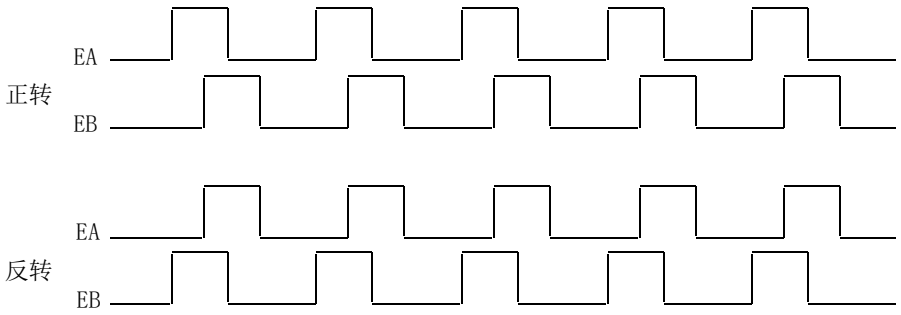


图3.11 A/B相正交信号

在这种模式下，EA 脉冲信号超前或滞后 EB 脉冲信号 90 度，而这种超前或滞后代表电机的运转方向。如图 3.11 所示，当 EA 信号超前 EB 信号 90° 时，被视为正转；当 EB 信号超前 EA 信号 90° 时，被视为反转。

为了提高编码器的分辨率，用户还可选用 4 倍、2 倍频计数模式对 EA，EB 信号进行计数设置。

1 倍频计数：若只用 EA 信号的上升沿触发计数器，一个脉冲周期就计数一次。

2 倍频计数：EA、EB 信号的上升沿都参与触发计数器，故将一个脉冲周期就分为两份。所以，计数精度提高了 2 倍。

4 倍频计数：EA、EB 信号的上升沿和下降沿都参与触发计数器，故将一个脉冲周期就分为四份。所以，计数精度提高了 4 倍。

例如：如果使用的编码器为 2500 线，即电机转一周反馈的 EA、EB 脉冲数都为 2500 个。让电机转一周，若编码器输入模式为 4 倍频计数，编码器计数器的值为 10000；若设置为 2 倍频计数，编码器计数器的值为 5000；若设置为 1 倍频计数，编码器计数器的值为 2500。

3.4.1.2 编码器信号输入接口电路

如果使用差分输出的编码器，输入信号的正端接 EA+(或 EB+，EZ+)端，负端接 EA-(或 EB-，EZ-)端。如图 3.12 所示。

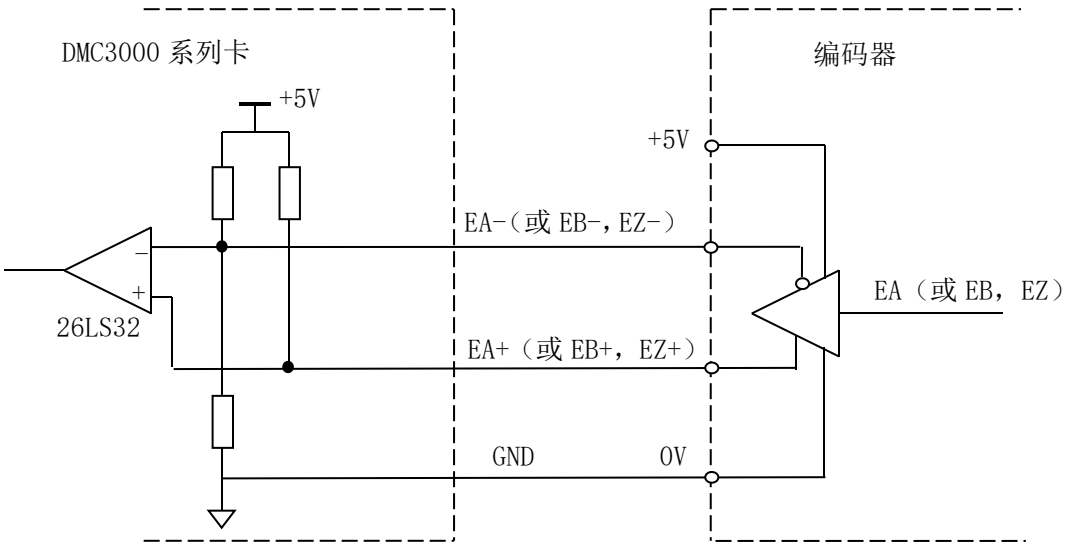


图 3.12 差分输出编码器接线原理图

如果使用集电极开路输出的编码器，则编码器输出信号接 EA+（或 EB+，EZ+）端，而 EA-（或 EB-，EZ-）端悬空。如图 3.13 所示。

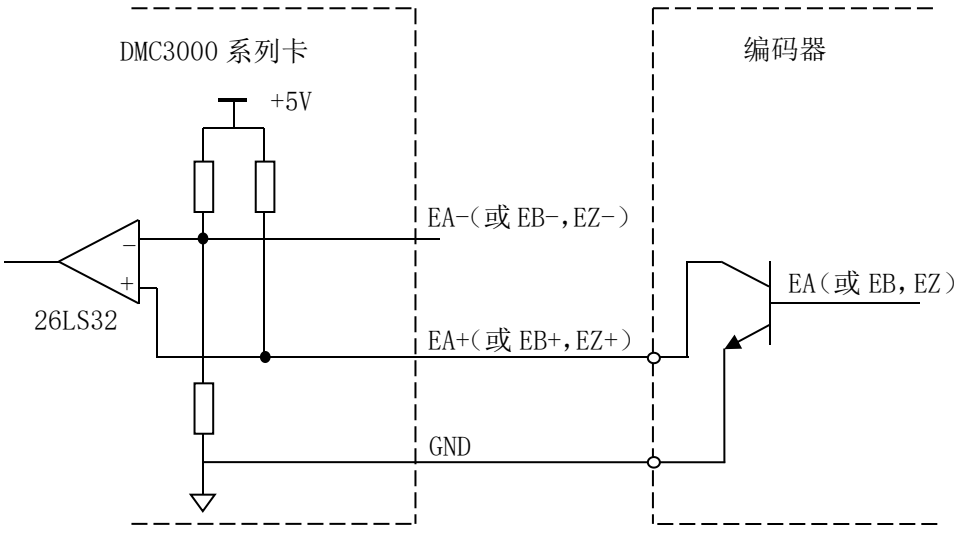


图 3.13　集电极开路输出的编码器接线原理图

注意：

1）编码器等脉冲输入信号的 EA+、EA-、EB+、EB-和 EZ+、EZ-的差分信号电压差必须高于 3.5V，小于 5V，且输出电流不应小于 6mA。

2）需要将输入设备的地线和控制卡的 GND 连接。

3）DMC3000 系列 4 款新版硬件卡最后 2 个伺服轴支持差分和单端编码器输入，剩余伺服轴只支持差分信号输入。即 DMC3400A 前两轴只支持差分信号编码器输入，后两轴支持差分和单端编码器输入；DMC3600 前四轴只支持差分信号编码器输入，后两轴支持差分和单端编码器输入；DMC3800 前六轴只支持差分信号编码器输入，后两轴支持差分和单端编码器输入；DMC3C00 前六轴只支持差分信号编码器输入，后两伺服轴支持差分和单端编码器输入。

3.4.2　手摇脉冲发生器输入接口

DMC3000 系列运动控制卡为每个轴提供了手摇脉冲发生器（简称：手轮）脉冲信号 PA、PB 的输入接口，用户可以通过手摇脉冲发生器控制电机的运动，电机的运动距离和速度由手摇脉冲发生器输入的脉冲数和脉冲频率控制。其接口原理如图 3.14 所示。

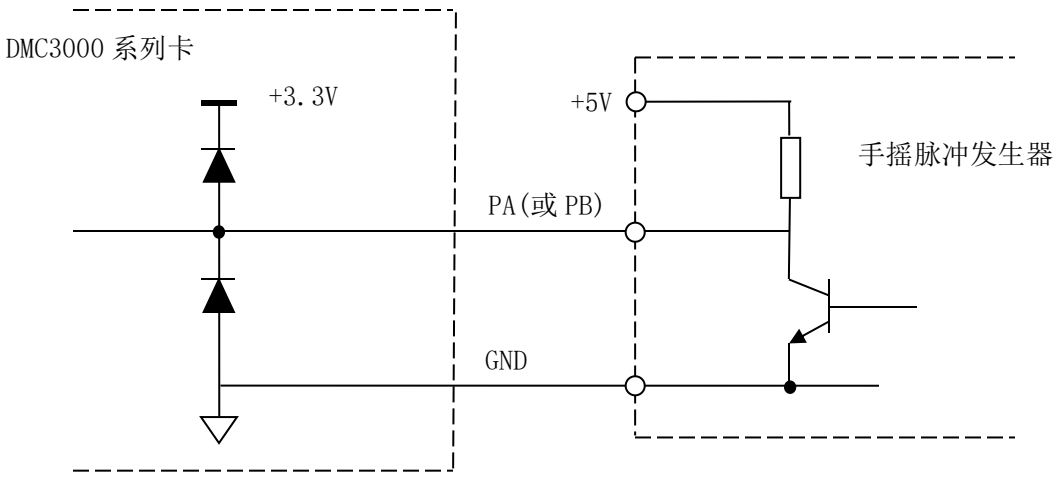


图 3.14　手轮脉冲输入接口原理图

3.5　专用 I/O 接口电路

3.5.1　原点信号输入接口

DMC3000 系列卡为每个轴都提供了 1 个原点位置传感器信号的输入端口 ORG,其接口电路如图 3.15 所示。

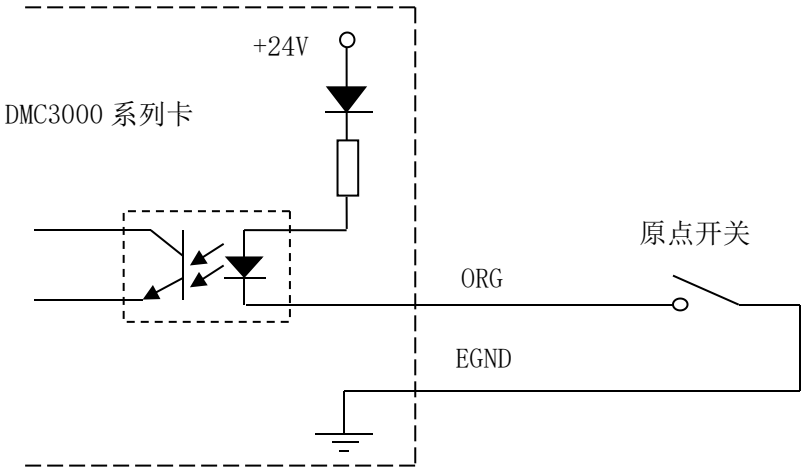


图 3.15　原点信号接口电路原理图

原点信号由 ORG 输入端口接入运动控制卡后，经过光耦后进入微处理器。光电隔离电路可以有效地将外部干扰信号隔离在控制卡之外，有效地提高了系统的可靠性。

3.5.2 限位信号输入接口

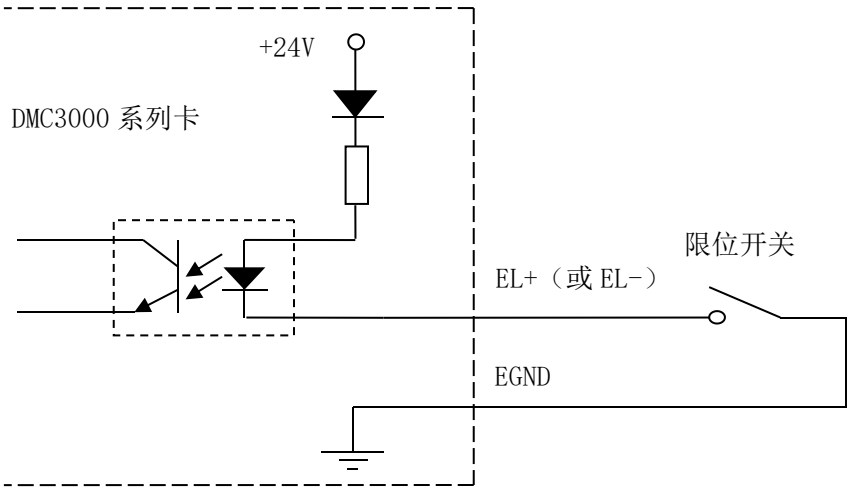


图 3.16 限位信号接口电路原理图

DMC3000 系列运动控制卡为每个轴提供了 2 个机械限位信号 EL+ 和 EL-。EL+为正向限位信号，EL-为负向限位信号。当运动平台触发限位开关时，EL+或 EL-即有效，控制卡将禁止运动平台继续向前运动。其接口电路如图 3.16 所示。

注意：用户需根据使用的限位开关类型来设置限位开关的有效工作电平。当使用常开型限位开关时，应通过软件选择 EL+、EL-信号为低电平有效；当使用常闭型限位开关时，应选择 EL+、EL-信号为高电平有效。

3.5.3 伺服电机驱动器控制信号接口

DMC3000 系列运动控制卡为每一轴均提供了伺服电机驱动器专用信号接口，其中信号 RDY、ALM 和 INP 用于监控伺服电机状态，信号 SEVON 和 ERC 用于设置伺服电机状态。

3.5.3.1 驱动器使能信号

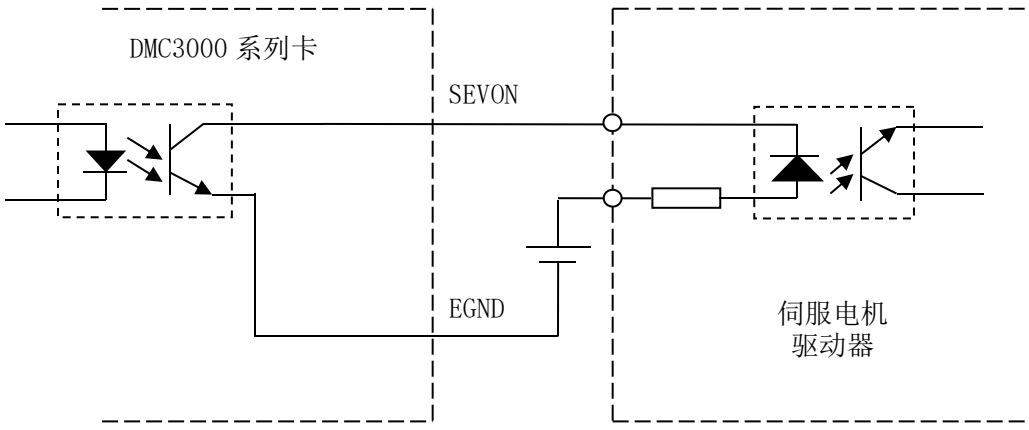


图 3.17　SEVON 信号接口原理图

当伺服电机驱动器的 SEVON 信号为无效状态时，伺服电机驱动器不工作，电机处于自由状态；当 SEVON 信号有效时，伺服电机驱动器进入工作状态，电机锁紧。其接口电路原理图如图 3.17 所示。

3.5.3.2　驱动器准备好信号

当伺服电机驱动器处于准备好状态，RDY 信号就会自动将该信号置为有效。此时，运动控制卡可以向伺服电机驱动器发出运动命令。其接口电路原理图如图 3.18 所示。

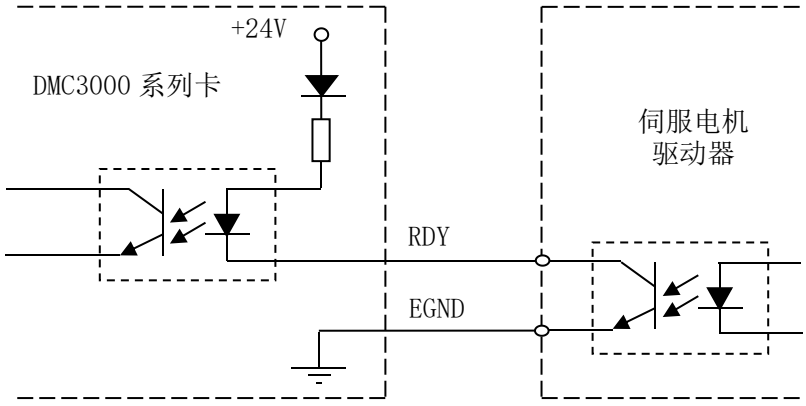


图 3.18　RDY 信号接口原理图

3.5.3.3 驱动器报警信号

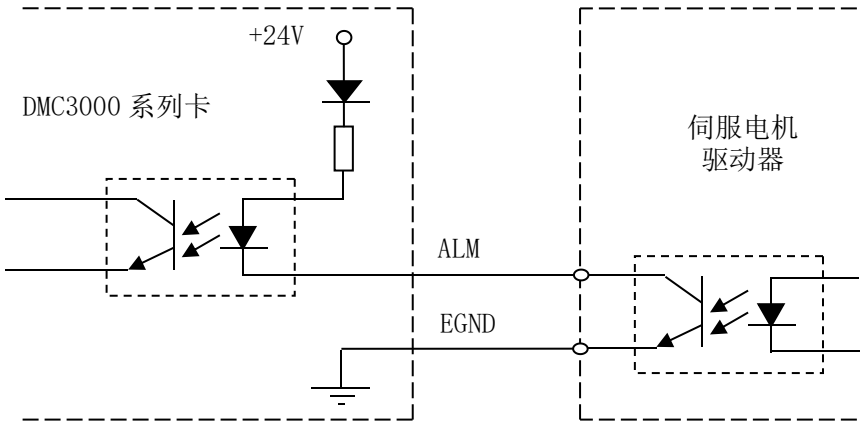


图 3.19 ALM 信号接口原理图

ALM 信号是伺服电机驱动器发出的报警信号。当运动控制卡接收到 ALM 信号后，将立即中止发送运动指令脉冲，或先减速再停止发送脉冲。其接口电路原理图如图 3.19 所示。

3.5.3.4 驱动器位置到达信号

雷赛控制技术 DMC3000 系列运动控制卡均为每一轴提供了用于监控伺服电机定位结果的 INP 信号接口。典型接口及接线原理如图 3.20。

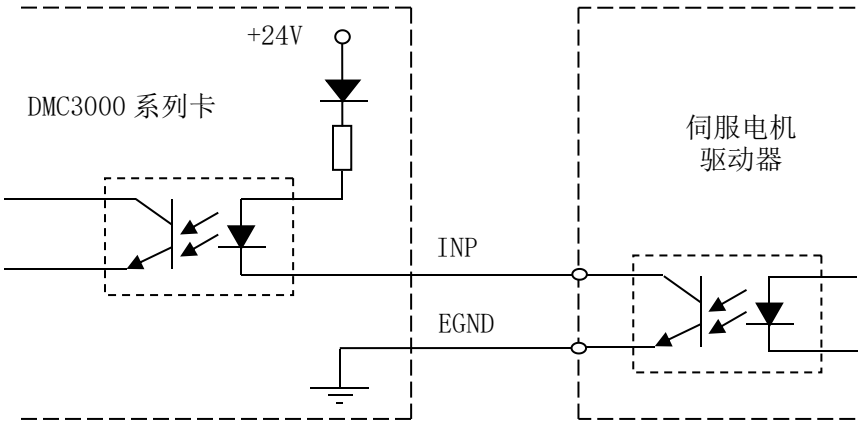


图 3.20 INP 信号接线原理图

注意：当使能了 INP 信号功能时，只有在 INP 信号为有效状态下，对应的轴才能进行运动，否则此时检测轴的状态是正在运行的（对轴运动作限制）。

3.5.3.5 驱动器位置偏差清除信号

雷赛控制技术 DMC3000 系列运动控制卡均为每一轴提供了用于清零伺服驱动位置偏差计数器

的 ERC 信号输出接口。典型接口及接线原理如图 3.21。

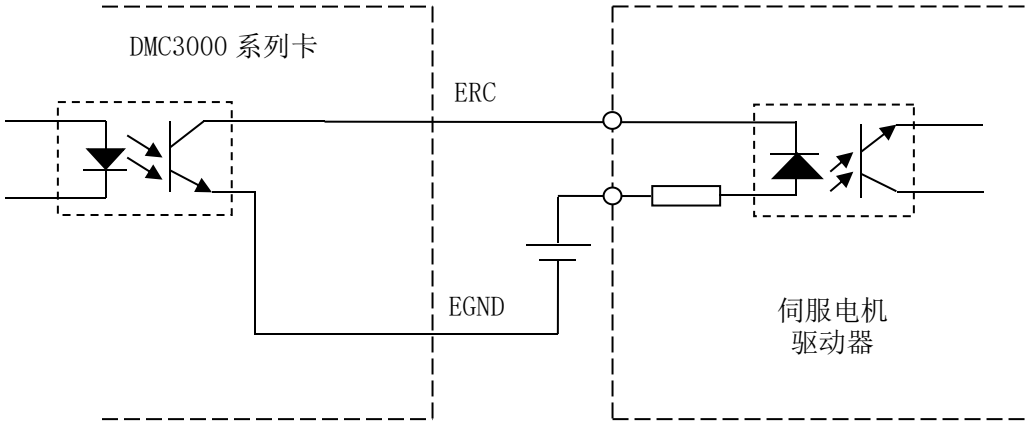


图 3.21　ERC 信号接线原理图

3.5.4　高速位置锁存输入信号接口

DMC3000 系列卡每四轴共用一个位置锁存输入信号 LTC（DMC3C00 后四轴除外），信号 LTC0 锁存 0~3 轴的当前编码器位置或指令位置，信号 LTC1 锁存 4~7 轴的当前编码器位置或指令位置。其接口电路原理如图 3.22 所示。

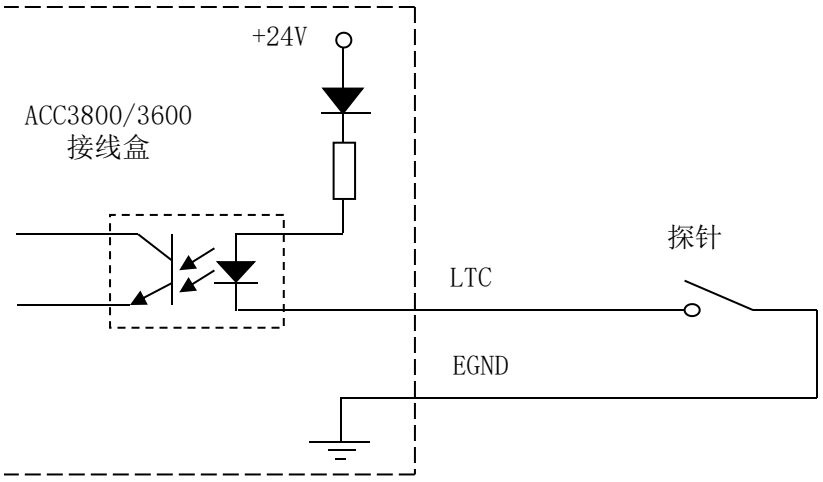


图 3.22　ACC3800/3600 位置锁存输入信号接口原理图

3.5.5　高速位置比较输出信号接口

DMC3000 系列卡共有四个高速位置比较器，每个高速位置比较器均配有 1 个硬件位置比较输出接口。通过软件使能后，可分别设置比较模式以及关联电机轴号，当该轴的指令寄存器内的数

值或编码器寄存器内数值满足触发条件时，硬件自动在 CMP 端口上输出一个开关信号。

ACC3800/3600/XC00/X400B 接线盒的 CMP 接口原理图如图 3.23 所示。

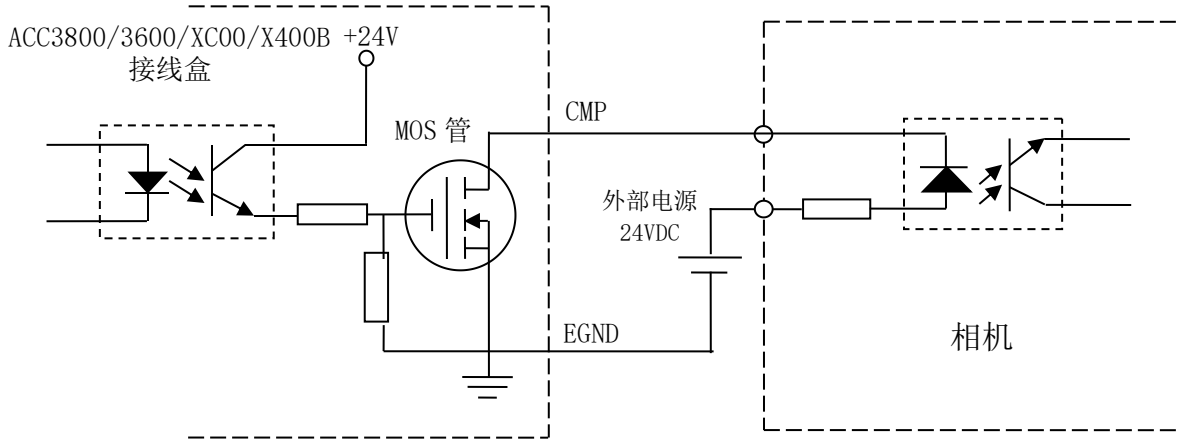


图 3.23　ACC3800/3600/XC00/X400B 高速位置比较输出信号接口原理图

3.6　通用 I/O 接口电路

DMC3000 系列控制卡除了提供专用的数字 I/O 接口外还提供了大量的通用数字 I/O 接口。

3.6.1　通用数字输入信号接口

DMC3000 系列卡有 16 路通用数字输入信号（其中 IN14、IN15 为高速输入）。所有输入接口均加有光电隔离元件，可以有效隔离外部电路的干扰，以提高系统的可靠性。通用数字输入信号接口原理图如图 3.24 所示。

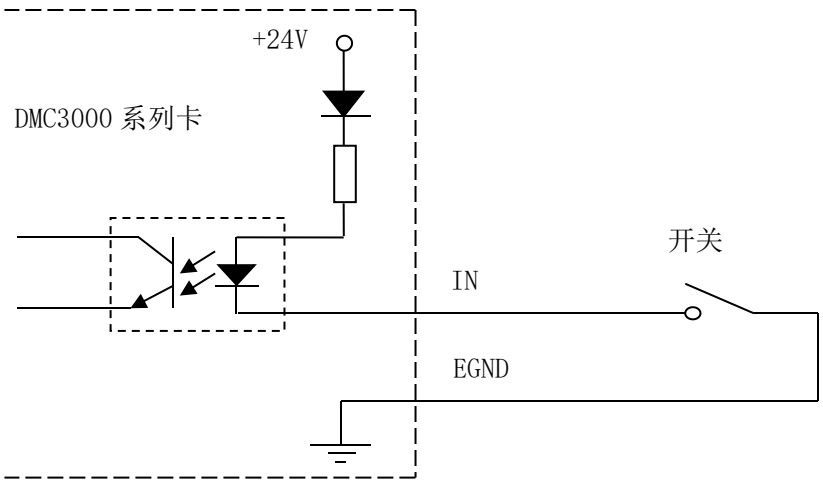


图 3.24　通用输入信号接口原理图

3.6.2 通用数字输出信号接口

DMC3800 卡通过与 ACC3800 接线盒相连，有 16 路通用数字输出信号（其中 OUT12~15 为高速输出），由 MOS 管驱动，其最大工作电流为 500 mA（5～24Vdc，吸入），可用于控制继电器、电磁阀、信号灯或其它设备。

DMC3600 卡通过与 ACC3600 接线盒相连，有 16 路通用数字输出信号（其中 OUT12~15 为高速输出），由 MOS 管驱动，其最大工作电流为 500 mA（5～24Vdc，吸入），可用于控制继电器、电磁阀、信号灯或其它设备。

DMC3C00 卡通过与 ACC-XC00 接线盒相连，有 16 路通用数字输出信号（其中 OUT12~15 为高速输出），由 MOS 管驱动，其最大工作电流为 500 mA（5～24Vdc，吸入），可用于控制继电器、电磁阀、信号灯或其它设备。

DMC3400A 卡通过与 ACC-X400B 接线盒相连，有 14 路通用数字输出信号（其中 OUT14~15 为高速输出,OUT12~13 保留），由 MOS 管驱动，其最大工作电流为 500 mA（5～24Vdc，吸入），可用于控制继电器、电磁阀、信号灯或其它设备。

下面给出了通用数字输出信号接口控制 3 种常用元器件的接线图。

1、发光二极管

通用数字输出端口控制发光二极管时，需要接一限流电阻 R，限制电流在 10mA 左右，电阻需根据使用的电源来选择，电压越高，使用的电阻值越大。接线图如图 3.25 所示。

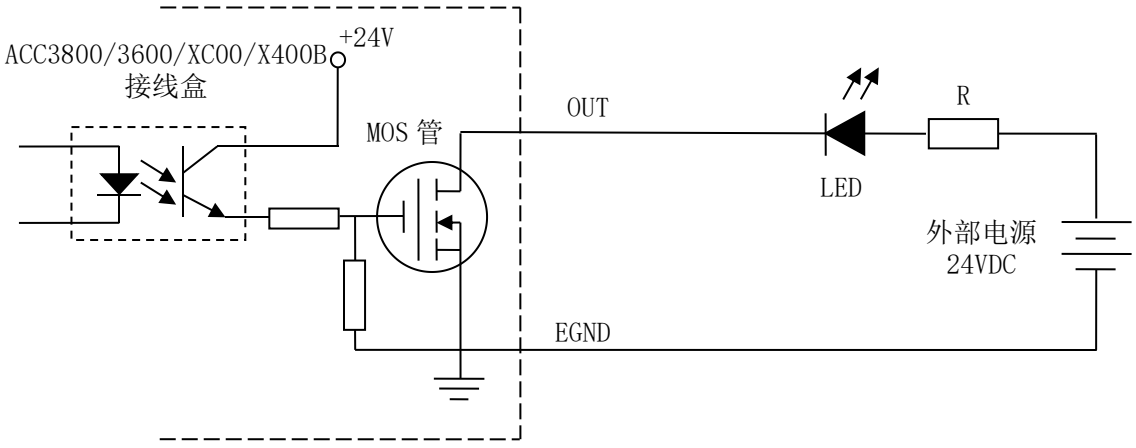


图 3.25 ACC3800/3600/XC00/X400B 输出口接发光二极管

2、灯丝型指示灯

通用数字输出端口控制灯丝型指示灯时，为提高指示灯的寿命，需要接预热电阻 R，电阻值

的大小，以电阻接上后，输出口为 1 时，灯不亮为原则。接线图如图 3.26 所示。

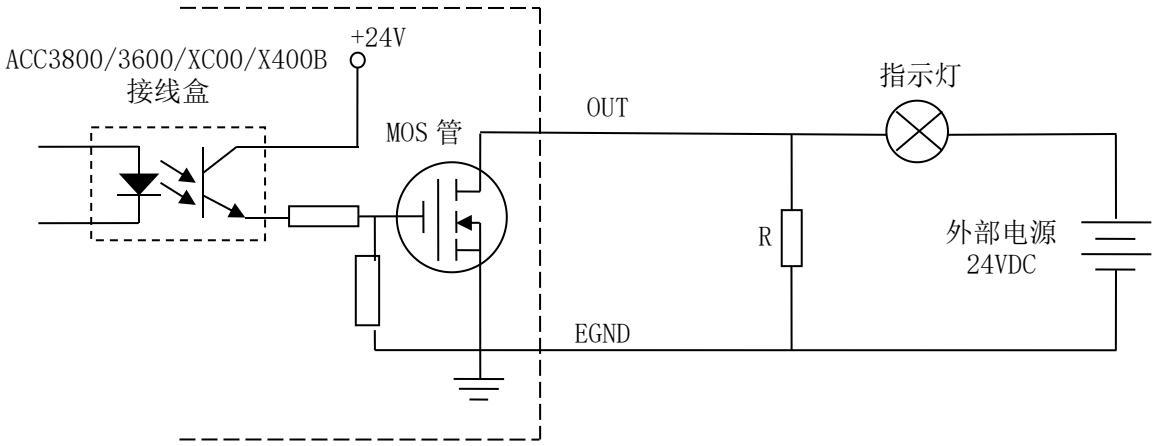


图 3.26　ACC3800/3600/XC00/X400B 灯丝型指示灯接线图

3、小型继电器

继电器为感性负载，必须并联一个续流二极管。当继电器突然关断时，继电器中的电感线圈产生的感应电动势可由续流二极管消耗，以免 ULN2803 或 MOS 管被感应电动势击穿。其接线图如图 3.27 所示。

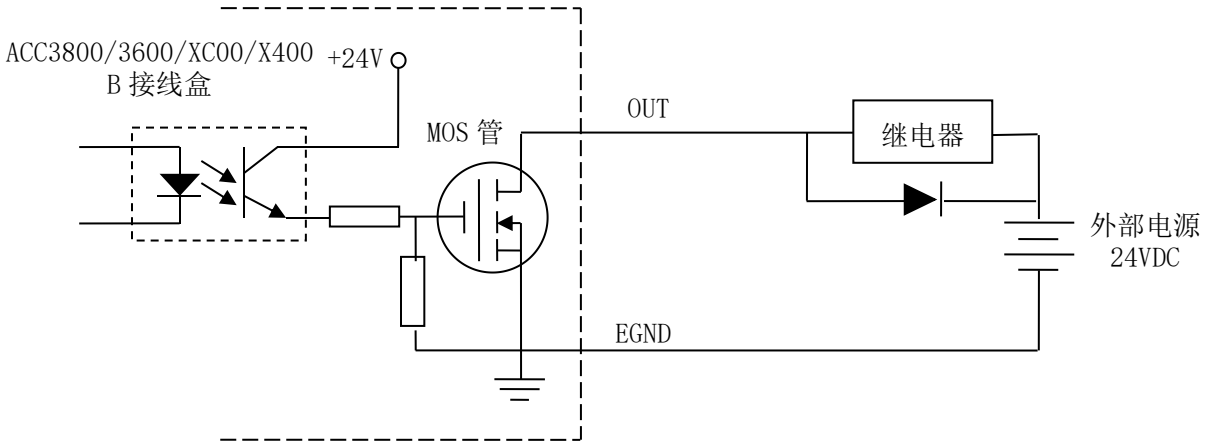


图 3.27　ACC3800/3600/XC00/X400B 接小型继电器的接线图

注意：在使用通用数字输出端口时，切勿把外部电源直接连接至通用数字输出端口上；否则，会损坏输出口。

3.7 CAN-IO 扩展模块接口电路

CAN-IO 扩展模块是 DMC3000 系列运动控制卡的配套产品，扩展 DMC3000 系列卡的 IO 端口。通过菊花链的连接方式可以将多个 CAN-IO 扩展模块挂在同一块运动控制卡下面，最多支持连接 8 个 CAN-IO 扩展模块。

CAN-IO 扩展模块各接口具体定义详见各个 CAN-IO 扩展模块手册。

3.7.1 CAN-IO 扩展模块的连接与设置

使用 CAN-IO 扩展模块时，必须先将各个扩展模块与 ACC3800/3600/XC00/X400B 接线盒连接，然后设置好各个扩展模块的节点号及终端电阻。

3.7.1.1 CAN-IO 扩展模块的连接

CAN-IO 扩展模块与 ACC3800/3600 接线盒是通过菊花链的形式连接，如图 3.28 所示。

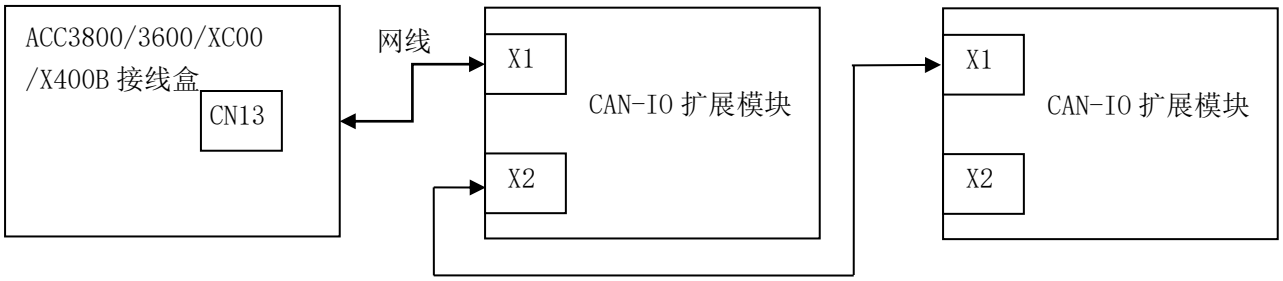


图 3.28 CAN-IO 扩展模块与 ACC3800/3600 接线盒的连接示意图

3.7.1.2 终端电阻及模块启动的设置

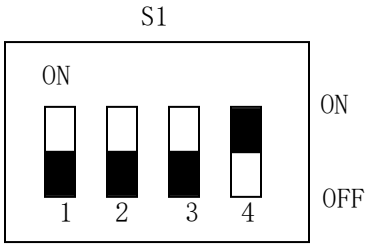


图 3.29 拔码开关示意图

表 3.1 S1 拔码开关的设置

拔码开关号	开关状态	定义
S1-1	ON	RUN
	OFF	STOP
S1-2	ON	保留
	OFF	保留
S1-3	ON	保留
	OFF	保留
S1-4	ON	终端电阻连接
	OFF	终端电阻断开

当使用多个 CAN-IO 扩展模块时，菊花链上的最后一个 CAN-IO 扩展模块的终端电阻必须设置

为连接状态。如果只使用一个 CAN-IO 扩展模块，那么此扩展模块的终端电阻也建议设置为连接状态。并且，需要运行的 CAN-IO 扩展模块必须设置为 RUN 状态。

这两个功能的设置都是通过拔码开关 S1 实现的，具体方法如表 3.1 所示。

3.7.1.3 CAN-IO 扩展模块节点号的设置

CAN-IO 扩展模块的节点号是通过 S3 旋转开关设置的，波特率是通过 S2 来设置的，须在模块上电之前设置,且需保证模块波特率与控制卡波特率相对应（具体请参考 8.24 节），其中 S2 设置为 7、8、9 时无效。S2、S3 重新设置后，模块需重新上电才有效。

当使用多个 CAN-IO 扩展模块时，菊花链上的第一个 CAN-IO 扩展模块的 CAN 节点号必须设置为 1，第二个扩展模块的 CAN 节点号必须设置为 2，第三个扩展模块的 CAN 节点号必须设置为 3，.....以此类推。

3.7.2 通用数字输入信号接口

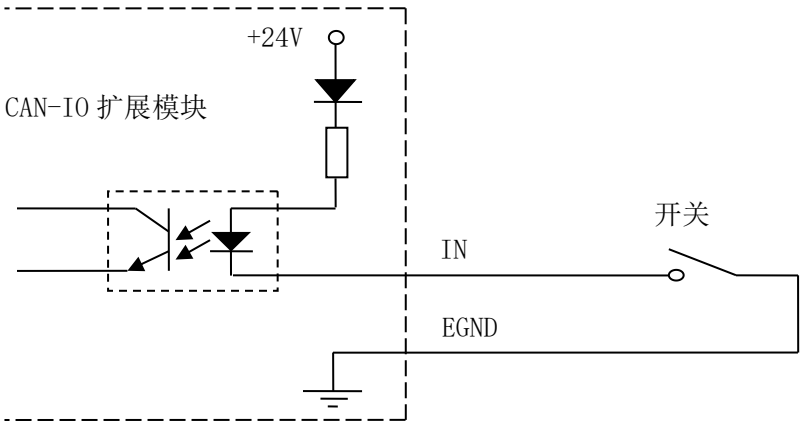


图 3.30 通用输入信号接口原理图

所有输入接口均加有光电隔离元件，可以有效隔离外部电路的干扰，以提高系统的可靠性。通用数字输入信号接口原理图如图 3.30 所示。

3.7.3 通用数字输出信号接口

所有输出接口均由 MOS 管驱动，单路输出电流可达 500mA，可用于对继电器、电磁阀、信号灯或其它设备的控制。其接口电路都加有光电隔离元件，可以有效隔离外部电路的干扰，提高了系统的可靠性。输出电路采用 OD 设计，上电默认 MOS 管关断。

注意：在使用通用数字输出端口时，切勿把外部电源直接连接至通用数字输出端口上；否则，会损坏输出口。

下面给出了通用数字输出信号接口控制 3 种常用元器件的接线图。

1、发光二极管

通用数字输出端口控制发光二极管时，需要接一限流电阻 R，限制电流在 10mA 左右，电阻需根据使用的电源来选择，电压越高，使用的电阻值越大。接线图如图 3.31 所示。

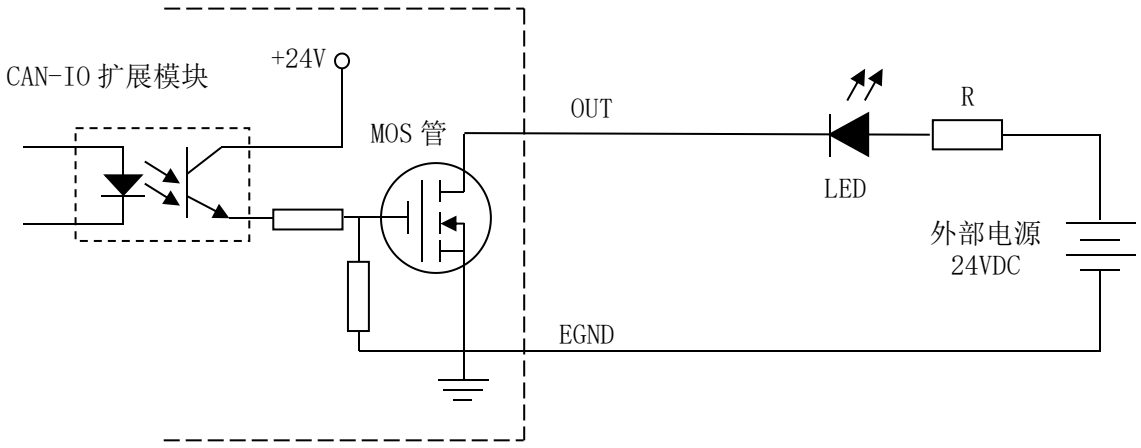


图 3.31　通用输出口接发光二极管

2、灯丝型指示灯

通用数字输出端口控制灯丝型指示灯时，为提高指示灯的寿命，需要接预热电阻 R，电阻值的大小，以电阻接上后，输出口为 1 时，灯不亮为原则。接线图如图 3.32 所示。

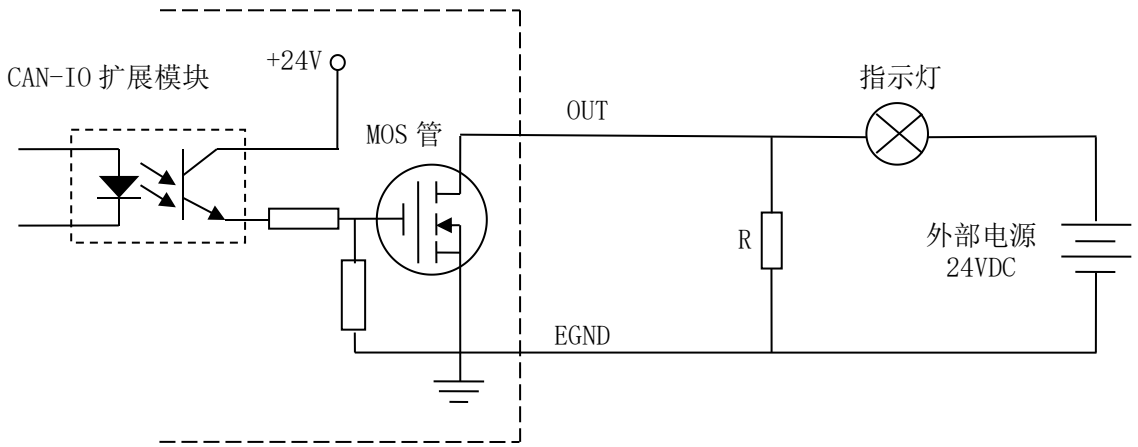


图 3.32　通用输出口接灯丝型指示灯

3、小型继电器

继电器为感性负载，必须并联一个续流二极管。当继电器突然关断时，继电器中的电感线圈产生的感应电动势可由续流二极管消耗，以免 ULN2803 或 MOS 管被感应电动势击穿。其接线图如图 3.33 所示。

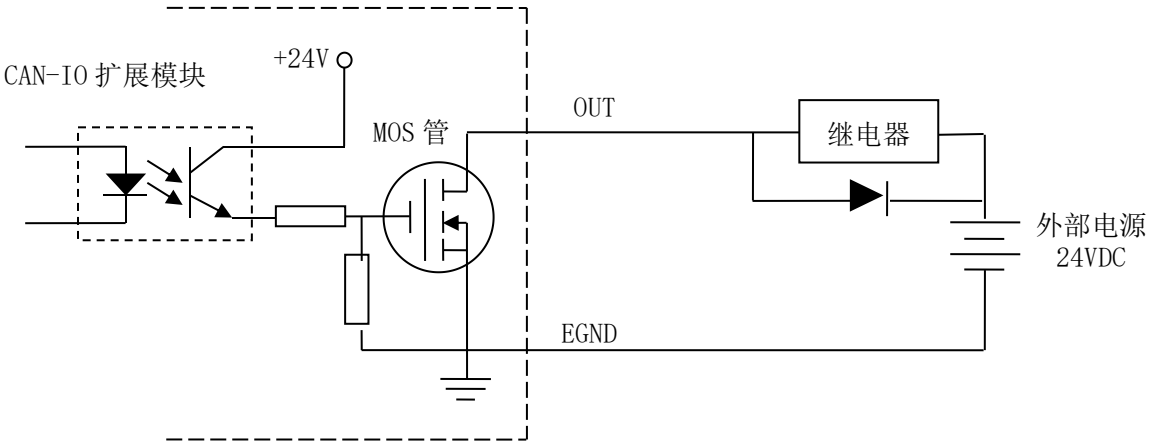


图 3.33　通用输出口接小型继电器

第 4 章　硬件及驱动程序的安装

4.1　硬件安装步骤

4.1.1　DMC3800 硬件设置

如图 4.1 所示，DMC3800 运动控制卡上有 3 组拨码开关 S2、S4 和 S5，用于设置 DMC3800 卡的工作方式和参数。

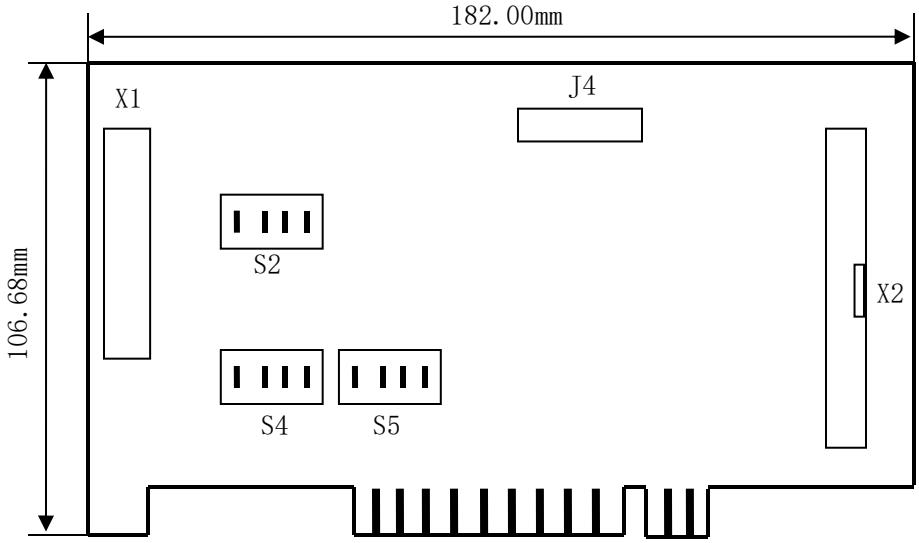


图 4.1　DMC3800 板卡跳线开关、拨码开关的位置

1.电机脉冲信号输出方式的选择

DMC3800 控制卡主板上没有跳线开关用于设置电机脉冲信号输出方式,但是在接线盒 ACC3800 上的 CN1~CN8 轴控制端口上提供+5V 电源。当使用+5V 和 PUL-端口时，则指令脉冲信号输出方式为单端输出;当使用 PUL+和 PUL-端口时,则指令脉冲信号输出方式为差分输出。参见图 3.12~3.13。

2.拨码开关设置

如图 4.2 所示，DMC3800 卡上有三个 4 位拨码开关 S2、S4 和 S5，其中 S2、S4 暂时保留，无定义。S5 用于卡号设置。具体方法如表 4.1、4.2 所示。

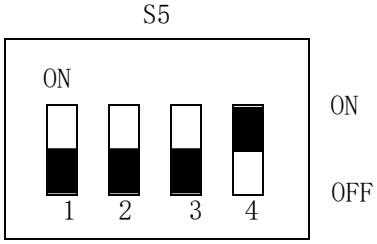


图 4.2　拨码开关示意图

表 4.1　拨码开关 S5 设置运动控制卡号

S5-4	S5-3	S5-2	S5-1	控制卡号
保留	OFF	OFF	OFF	0
保留	OFF	OFF	ON	1
保留	OFF	ON	OFF	2
保留	OFF	ON	ON	3
保留	ON	OFF	OFF	4
保留	ON	OFF	ON	5
保留	ON	ON	OFF	6
保留	ON	ON	ON	7

注意：1）拨码开关 S5-1、S5-2、S5-3 出厂默认配置为 OFF，即默认设置为 0 号卡。

2）当每张卡都设置为 0 号卡时，按靠近 CPU 的顺序自动排序。除此种情况外，如有两张卡以上设置为相同卡号，则初始化函数 dmc_board_init 会返回一个错误代码。

4.1.2　DMC3C00/3600/3400A 硬件设置

DMC3C00/3600/3400A 卡外观及硬件设置与 DMC3800 卡是相同的，关于 DMC3600/3400A 卡的硬件设置可参考章节 4.1.1　DMC3800 硬件设置。

4.1.3　DMC3800-PCIe/DMC3600-PCIe/3400A-PCIe 硬件设置

DMC3800-PCIe/DMC3600-PCIe/3400A-PCIe 硬件设置与 DMC3C00-PCIe 卡基本相同，关于 DMC3800-PCIe/DMC3600-PCIe/3400A-PCIe 的硬件设置可参考章节 4.1.4　DMC3C00-PCIe 硬件设置

4.1.4　DMC3C00-PCIe 硬件设置

如图 4.3 所示，DMC3C00-PCIe 运动控制卡上有 3 组拨码开关 S2、S4 和 S5，用于设置 DMC3C00-PCIe 卡的工作方式和参数。

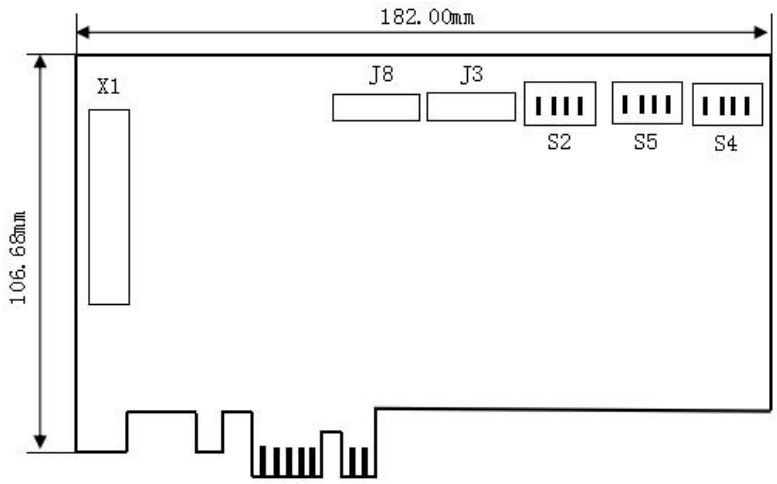


图 4.3　DMC3C00-PCIe 板卡跳线开关、拨码开关的位置

1.电机脉冲信号输出方式的选择

DMC3C00-PCIe 控制卡主板上没有跳线开关用于设置电机脉冲信号输出方式；但是在接线盒 ACC2-XC00 上的 CN1~CN8 轴控制端口上提供+5V 电源。当使用+5V 和 PUL-端口时，则指令脉冲信号输出方式为单端输出；当使用 PUL+和 PUL-端口时，则指令脉冲信号输出方式为差分输出。参见图 3.9~3.10。

2.拨码开关设置

如图 4.3 所示，DMC3C00-PCIe 卡上有三个 4 位拨码开关 S2、S4 和 S5，其中 S2、S4 暂时保留，无定义。S5 用于卡号设置。S5 具体使用方法如表 4.3 所示。

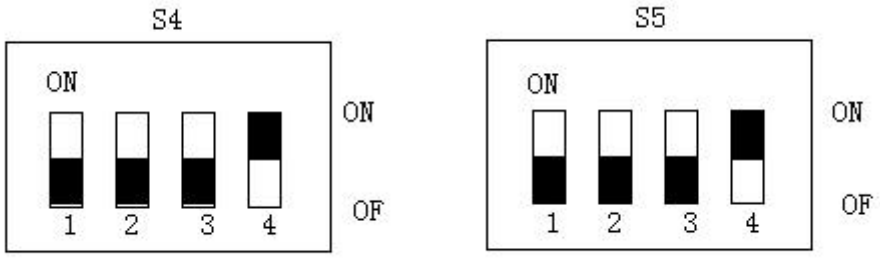


图 4.4　拨码开关示意图

表 4.3　拨码开关 S5 设置运动控制卡号

S5-3	S5-2	S5-1	控制卡号
OFF	OFF	OFF	0
OFF	OFF	ON	1
OFF	ON	OFF	2
OFF	ON	ON	3

S5-3	S5-2	S5-1	控制卡号
ON	OFF	OFF	4
ON	OFF	ON	5
ON	ON	OFF	6
ON	ON	ON	7

注意：1）拨码开关 S5-1、S5-2、S5-3 出厂默认配置为 OFF，即默认设置为 0 号卡。
当每张卡都设置为 0 号卡时，按靠近 CPU 的顺序自动排序。除此种情况外，如有两张卡以上设置为相同卡号，则初始化函数 dmc_board_init 会返回一个错误代码。

4.1.5 硬件安装

DMC3000 系列运动控制卡硬件遵从 32bit PCI 卡结构标准，其安装方法与其它 PCI 卡，如：声卡、网卡等相似。具体步骤如下：

1）打开控制卡的包装，参考相应运动控制卡硬件设置的说明，按照实际需求，完成跳线开关、拨码开关的设置；

2）操作员要带好防静电手套，并触摸一下地线，完全释放身上的静电；

3）关闭 PC 机以及一切与 PC 相连的设备；

4）打开 PC 机的机箱；

5）选择一个靠近处理器的 32bit PCI 插槽，将控制卡垂直插入插槽中；

6）将控制卡用螺钉固定在 PC 机机箱上，确保紧固可靠；

7）将接线板用电缆线与控制卡对应的插座连接，并确保连接牢固可靠。

4.2 驱动程序安装步骤

1、双击驱动文件 DMCDriver_Setup.exe，如果出现如图 4.3 所示对话框，点击“下一步”继续安装；如果出现如图 4.4 所示对话框，请选择“修复”，并点击“下一步”，跳转到第 5 步继续安装。

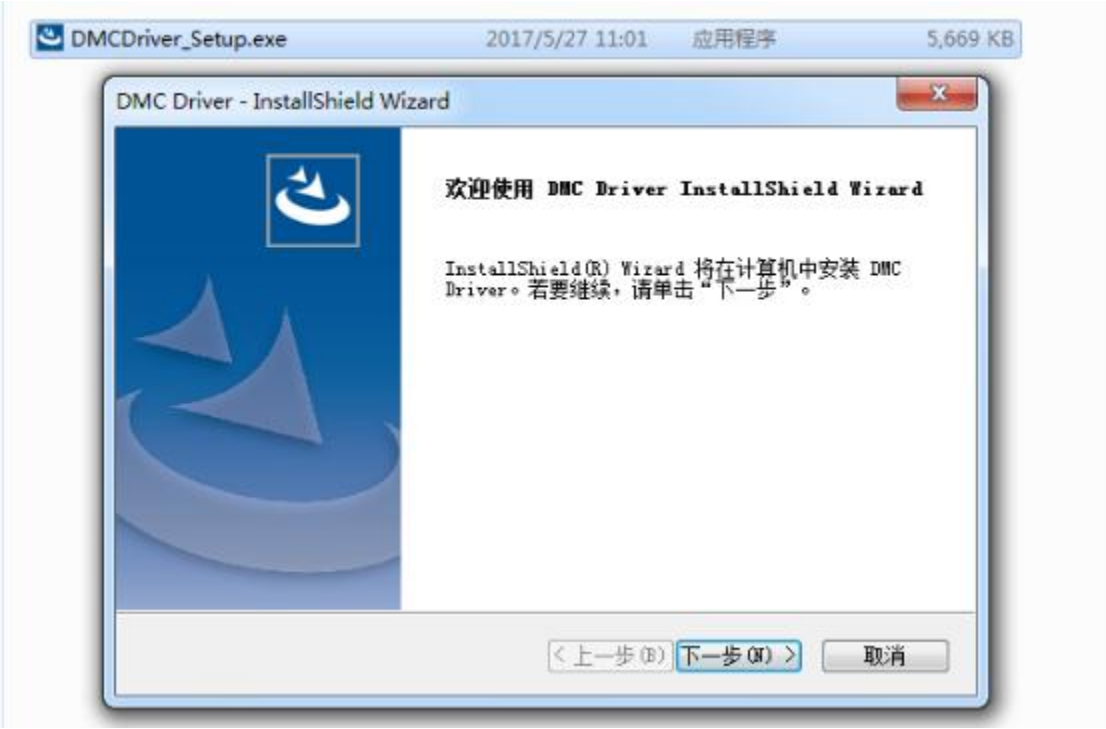


图 4.3 驱动程序开始安装

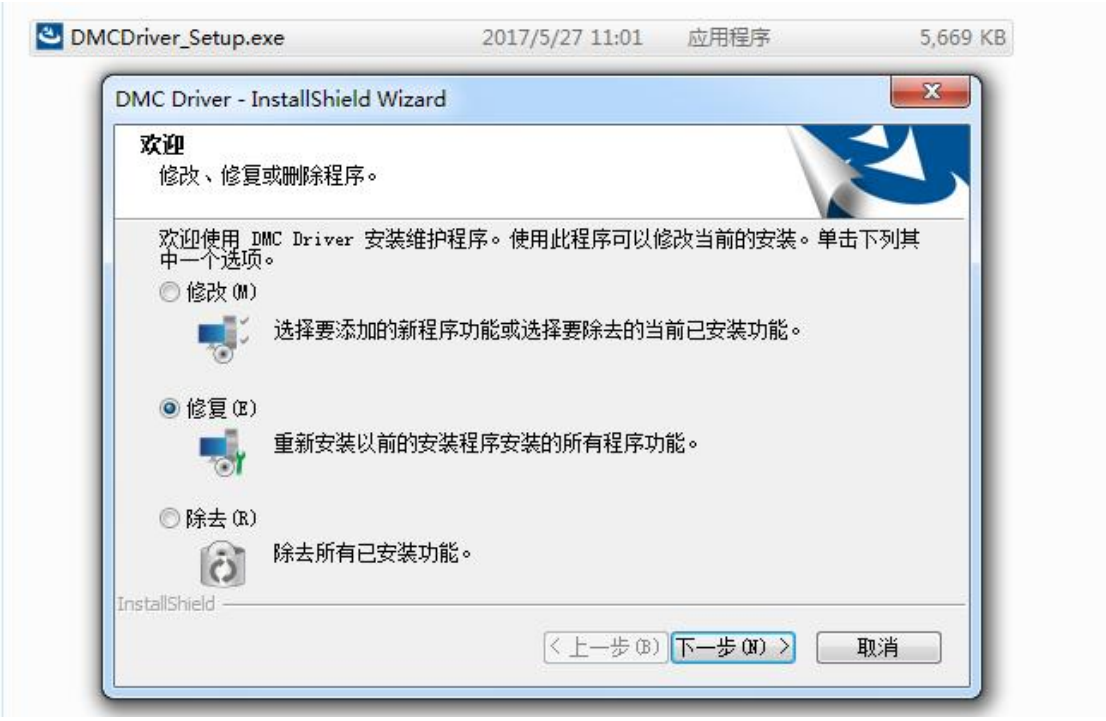


图 4.4 已安装驱动程序选项

2、如图 4.5 所示，输入用户名和公司名称，点击“下一步”继续安装。

图 4.5 输入信息对话框

3、如图 4.6 所示，请选择“全部”，然后点击“下一步”继续安装。

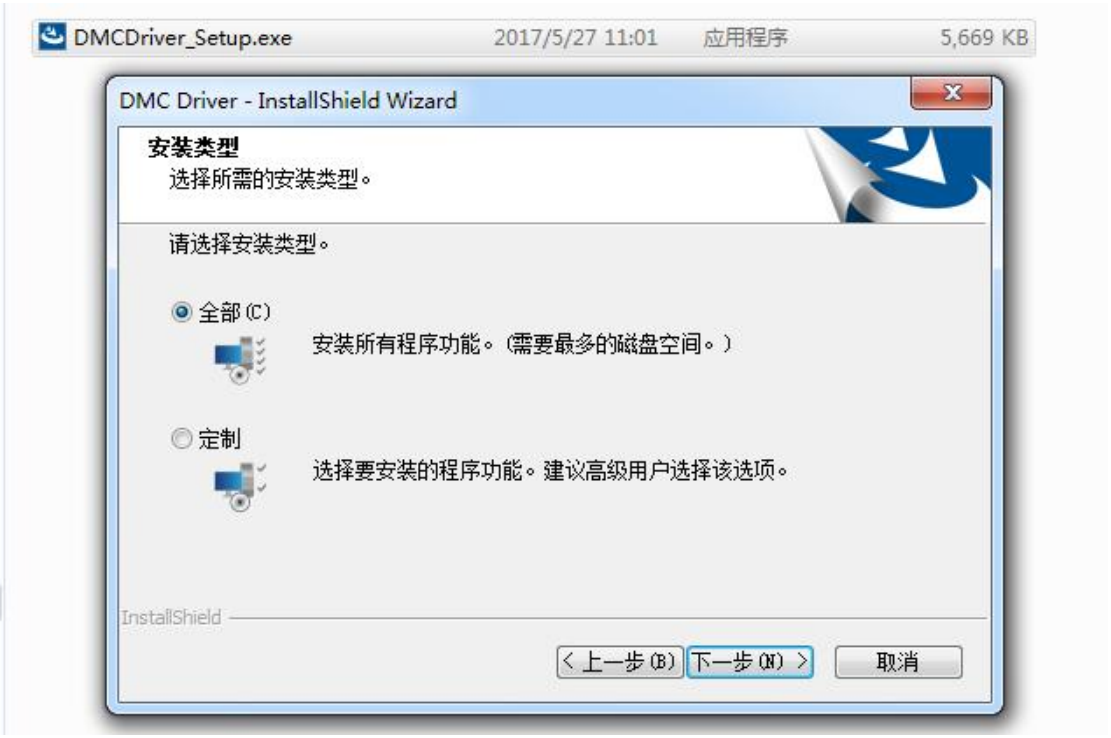


图 4.6 安装类型选择

4、如图 4.7 所示，点击“安装”按钮，开始安装，。

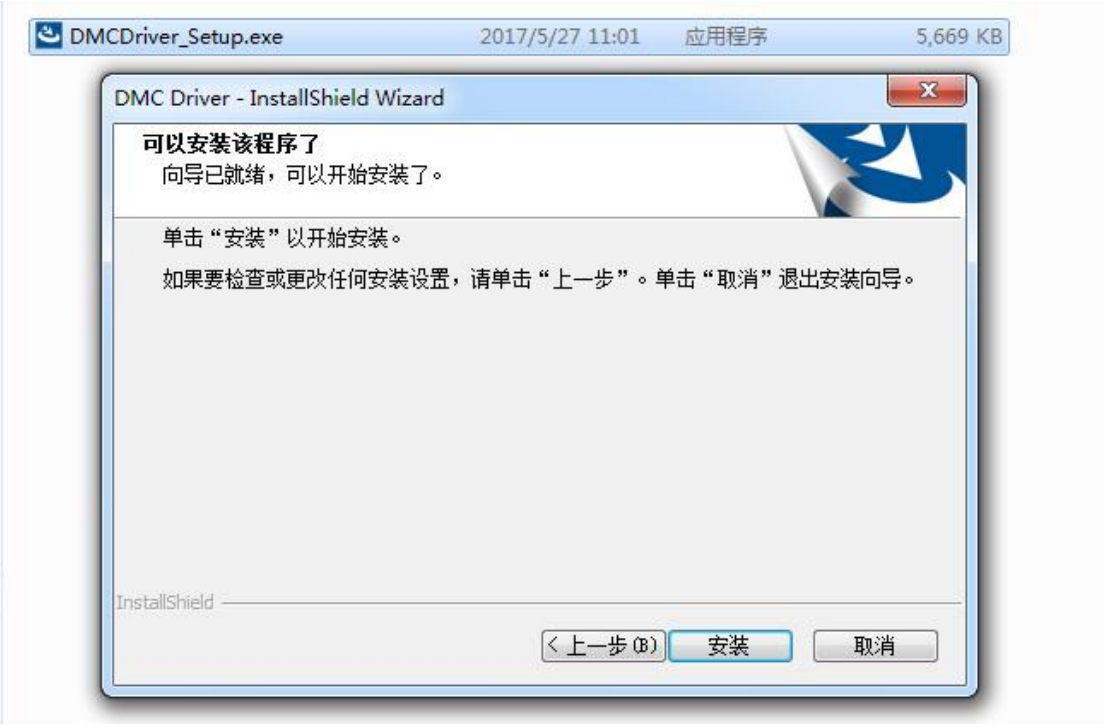


图 4.7 驱动程序开始安装

5、安装过程中会出现如图 4.8 所示的 Windows 安全提示对话框，请点击“始终安装此驱动程序软件”，并继续安装。

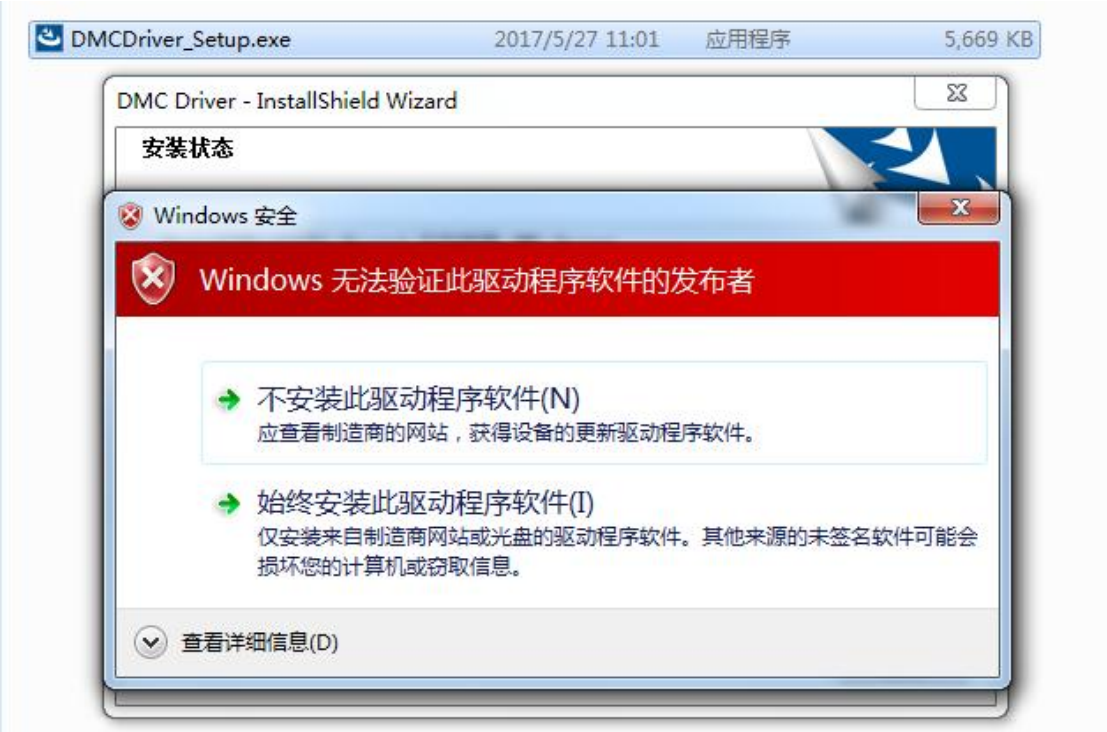


图 4.8 Windows 安全提示

6、安装完成后，显示界面如图 4.9 所示，点击“完成”。

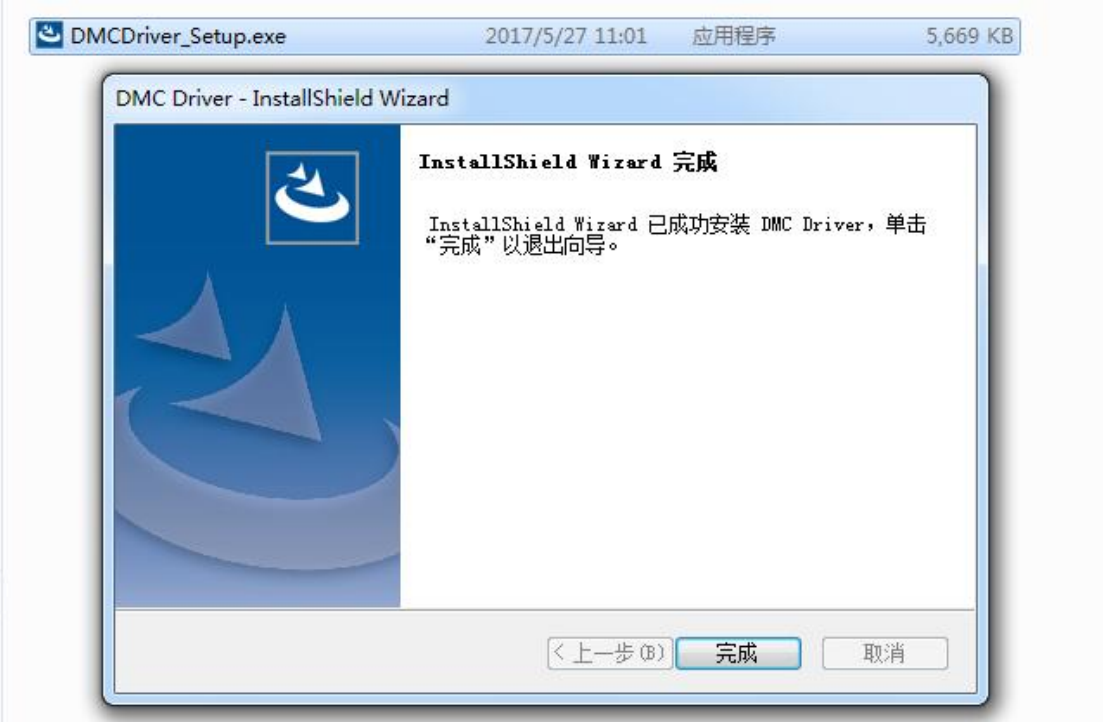


图 4.9 完成驱动程序的安装

7、打开设备管理器， 在“ Jungo” 选项下可以看到“ DMC3K5K” 和“ LeisaiDrvr1230” 注册信息。 至此， 控制卡就可以正常使用了， 如图 4. 10 所示。

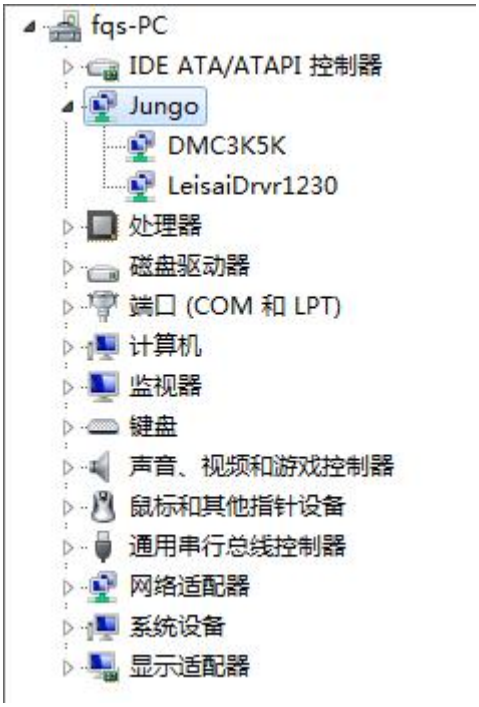


图 4.10 正确安装驱动程序后的设备信息

4.3 驱动程序卸载步骤

1、双击驱动文件 DMCDriver_Setup.exe，如果出现如图 4.11 所示对话框，请选择 “除去”，并点击“下一步”。

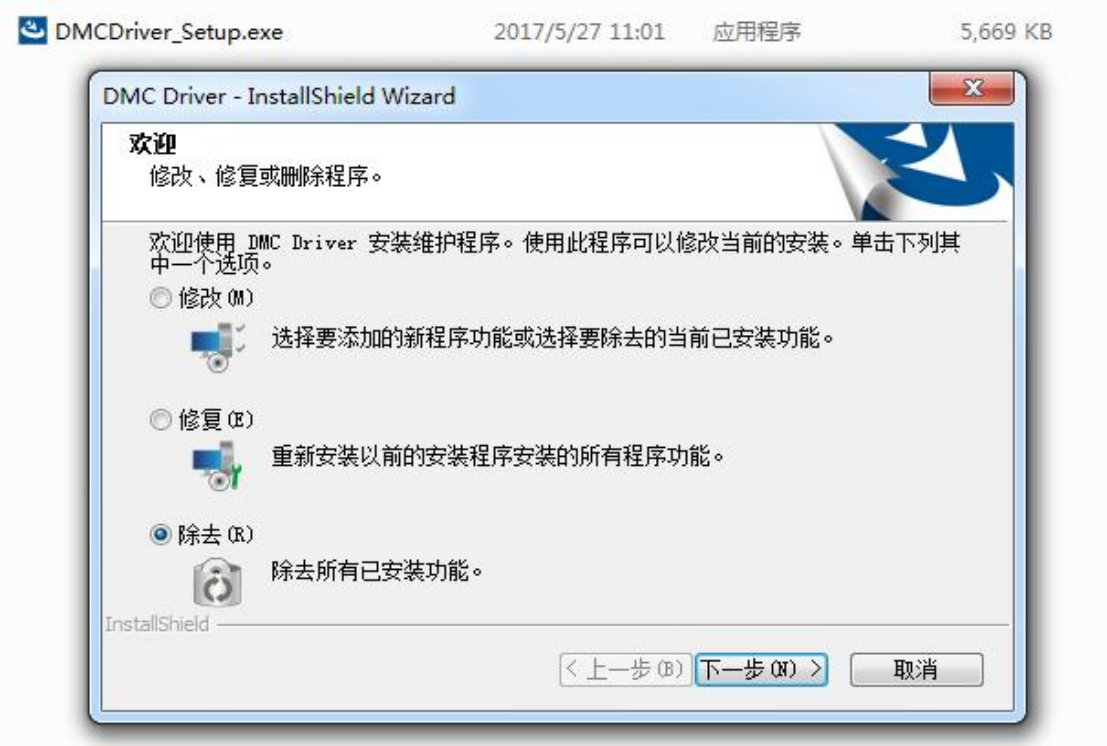


图 4.11 删除驱动选项

2、出现确认对话框，请选择“是”按钮。

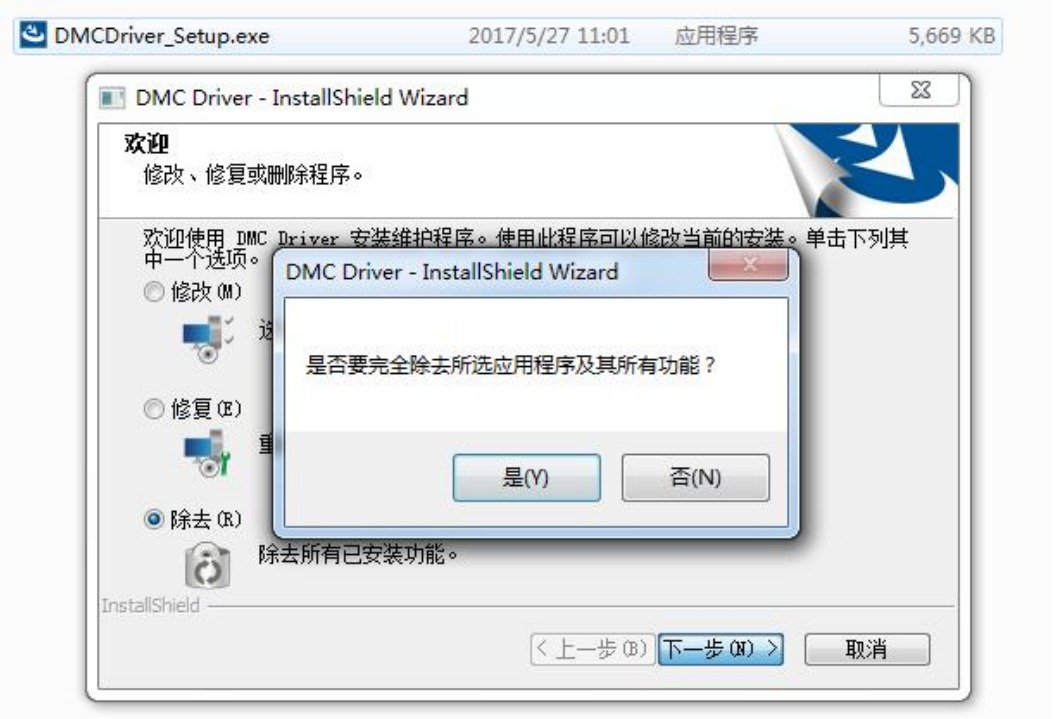


图 4.12 是否删除提示窗

3、卸载完成后，显示界面如图 4.13 所示，点击“完成”，完成卸载。

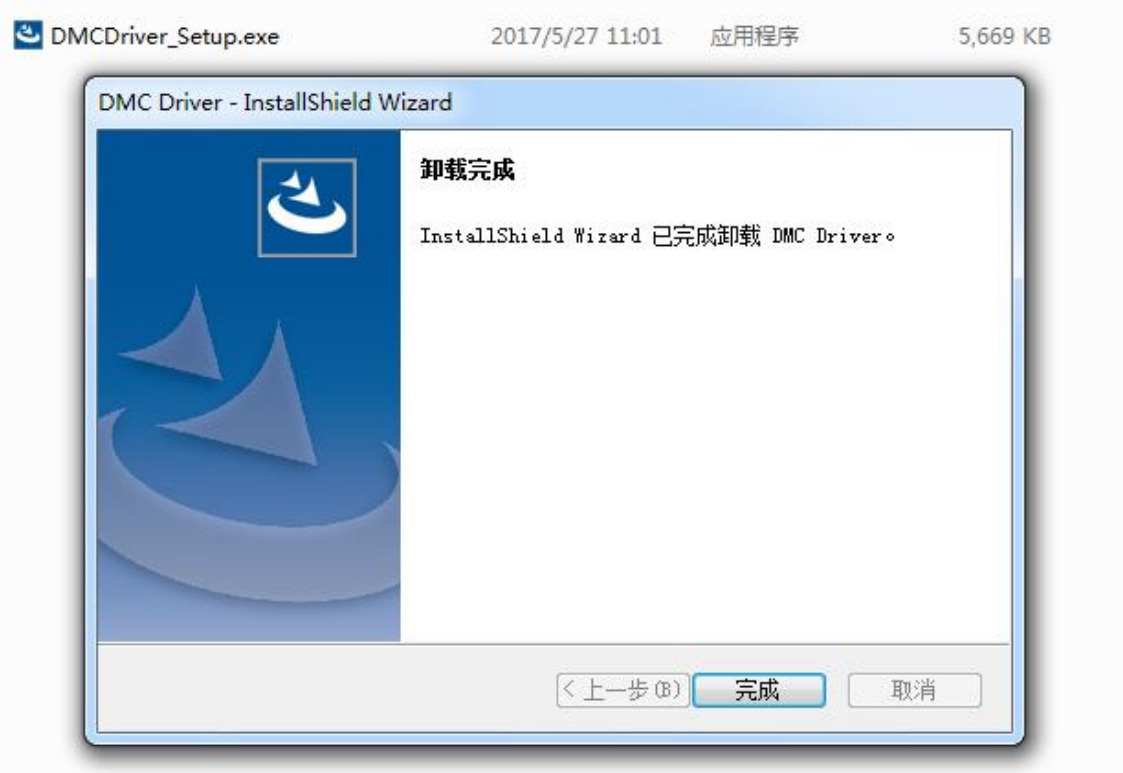


图 4.13 完成驱动程序的卸载

第 5 章　控制卡 Motion 使用手册

5.1　概述

控制卡 Motion 软件是基于 Windows 平台开发的控制卡调试工具软件，支持雷赛控制技术的多种型号控制卡，其具有调试控制卡所有功能的能力，操作方便快捷。

控制卡 Motion 软件兼容了传统的脉冲控制卡和 EtherCAT 总线控制卡的功能，不但可以调试单轴运动、多轴运动、插补运动等传统运动功能和 IO 状态、轴状态等监视功能，还可以实现 EtherCAT 总线配置下载等相关功能。

5.2　功能描述

控制卡 Motion 软件是一款控制卡辅助调试软件，适用于初次使用雷赛控制卡入门用户或控制卡使用调试查错。主要功能如下（注意：以下功能与具体型号控制卡有关，不同类型控制卡可能功能有所不同）：

- 参数设置，支持各个轴所有参数的设置，包括脉冲模式、脉冲当量、加减速时间、回零模式、限位、伺服等参数设置。
- 运动状态，支持控制卡本地或 EtherCAT 模块的轴状态、IO 状态、轴专用信号监视。
- 单轴测试，支持单个轴定长运动、定速运动、回零运动测试。
- 多轴测试，支持直线插补、多类型圆弧/螺旋线单段插补运动测试。
- 手轮测试，支持手轮倍率、通道、模式等相关参数设置，启动、停止等相关操作。
- PVT 测试，支持 PVT 模式、PVT 数据等设置，以及 PVT 理论曲线显示等，支持多轴 PVT 同时操作。
- 单轴比较，支持比较点编辑、比较器状态监视等功能
- 二维比较，支持二维比较点编辑、比较器状态监视等功能
- 原点锁存，支持原点锁存模式、锁存源、触发方式等参数设置，配置、复位锁存器等操作。
- 高速锁存，支持锁存方式、锁存源、触发方式等参数设置，配置、复位锁存器等相关操作。
- 连续插补，支持直线插补、圆弧/螺旋线、延时、IO 输出等连续插补相关配置，以及插补速度、前瞻等参数设置，插补器状态、缓存数量、当前行号等相关状态监视。

- PWM 测试，直接设置输出 PWM，显示 PWM 曲线图。
- AD 测试，监视所有通道的 AD 电压，并显示曲线图。
- DA 测试，直接设置输出 DA 电压，显示 DA 曲线图。
- CAN 状态，支持 CAN 状态显示以及连接操作。
- EtherCAT 总线配置，扫描并配置 EtherCAT 总线参数，设置轴映射、IO 映射等相关操作。
- 板卡信息，显示控制卡型号、版本、固件类型、轴数量等相关信息。
- 固件升级，支持在线升级控制卡固件。
- 硬件复位，支持复位重启控制卡。
- 信息输出，支持输出函数调用过程，以及函数错误码。
- 错误码查询，支持查询控制卡函数调用返回的错误码对应的解释查询。

详细情况见附件：控制卡 Motion 使用手册。

第 6 章　应用软件开发方法

DMC3000 系列运动控制卡的应用软件可以在 Visual Bisic 和 Visual C++等高级语言环境下开发。

如果您对 VB、VC 语言都不熟悉，建议您花两天时间阅读一本 VB 语言的培训教材，并且通过练习掌握该语言的基本技巧，如：编写简单的程序、创建窗体和调用函数。

如果您曾用 VB 或 VC 等程序语言开发过运动控制软件，并具有丰富的经验，则可直接阅读第 8 章“函数库详解”。

6.1　基于 WINDOWS 平台的应用软件结构

使用雷赛控制技术运动控制卡的自动化设备运动控制系统构架如图 6.1 所示：

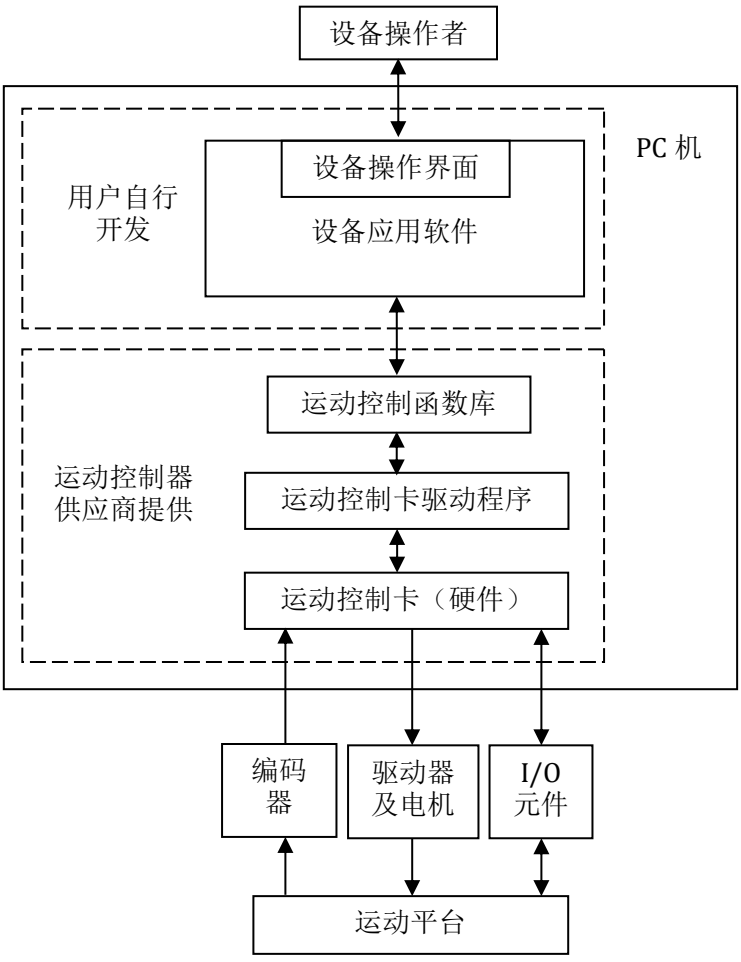


图 6.1　基于雷赛控制技术运动控制卡的自动化设备运动控制系统构架

从图 6.1 中可看出，运动控制系统的工作原理可以简单描述为：

1）操作员通过操作界面（包括显示屏和键盘）将指令信息传递给设备应用软件；

2）设备应用软件将操作者的信息以及应用软件中已有的运动流程、运动轨迹等数据转化为运动参数，并根据这些参数调用 DLL 库中运动函数；

3）运动函数通过雷赛控制技术运动控制卡驱动程序向运动控制卡发出控制指令；

4）运动控制卡根据控制指令发出相应的指令脉冲给驱动器及电机、读写通用输入输出口、读取编码器数据。

用户根据设备的工艺流程、运动轨迹和友好的人机界面等要求开发设备应用软件。雷赛控制技术已提供支持 DMC3000 系列运动控制卡的硬件驱动程序和 DLL 运动函数库，包括控制卡初始化、单轴及多轴控制、数字量输入/输出控制等多种函数。这些函数可以方便地完成与运动控制相关的功能，用户不需要更多了解硬件电路的细节以及运动控制和插补算法的细节，就能使用 VB、VC 等程序语言开发出自己的运动控制系统应用软件。

用户编写的设备应用软件的典型流程如图 6.2 所示。

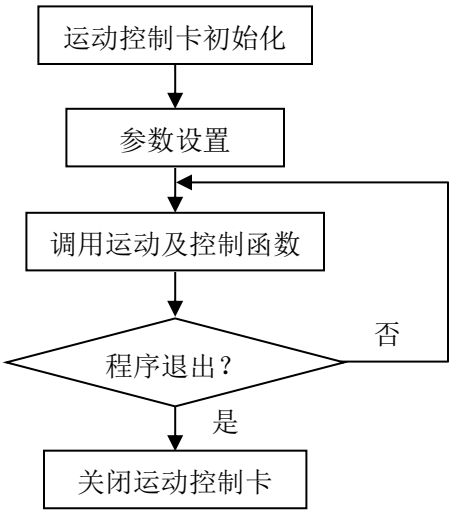


图 6.2　设备应用软件的典型流程

6.2　采用 VB 6.0 开发应用软件的方法

下面以 Visual Basic6.0 环境下编写一个点位运动的应用软件为例，讲解用 VB 开发应用软件的一般方法。

1）在磁盘上新建一个目录，如 E:\test1

2）打开 Visual Basic 6.0，新建一个“标注 EXE”工程，在对话框上添加按钮“启动”和“停止”，并将其名称分别修改为“CB_Start”和“CB_Stop”，如图 6.3 所示。

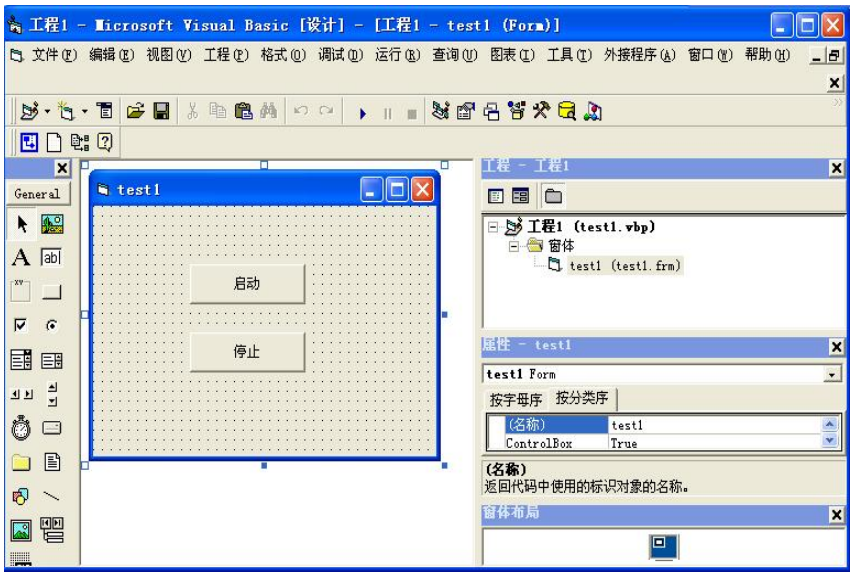


图 6.3　修改对话框（VB）

3）工程保存在 E:\test1 目录下。

4）在资料光盘相应目录下找到 LTDMC.bas、LTDMC.dll 和 PVT.dll 文件，拷贝到 test1 目录下。

5）菜单中选择“工程”->“添加模块”->“现存”，找到 test1 目录下的 LTDMC.bas 文件，添加到工程中，如图 6.4 所示。

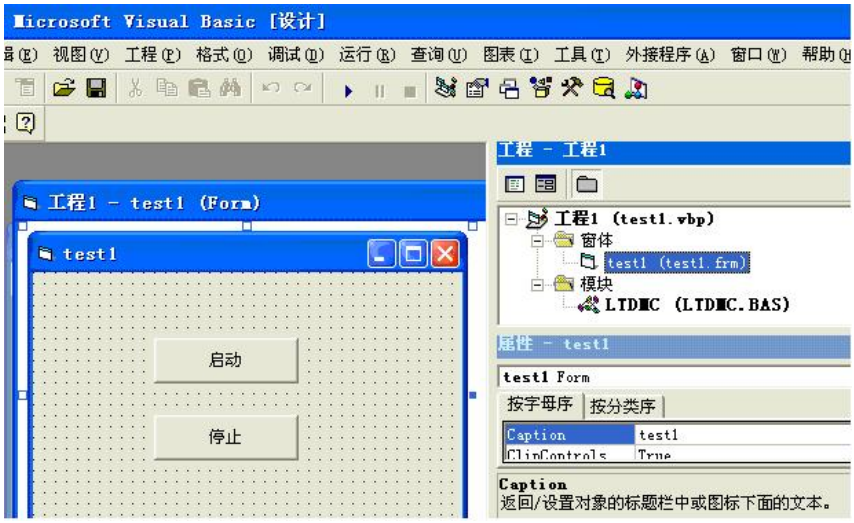


图 6.4　添加头文件

6）如图 6.5 所示，双击窗口控件，在 Form_Load 事件中添加代码 dmc_board_init。选择 UnLoad 事件，在 Form_Unload 事件中添加代码 dmc_board_close 双击“启动”按钮，在 CB_Start_Click 事件中添加代码如下：

```
dmc_set_profile 0,0,500,5000, 0.01,0.01,500
dmc_pmove 0,0,200000,0
```

双击“停止”按钮，在 CB_Stop_Click()事件中添加代码如下：

```
dmc_stop 0,0,0
```

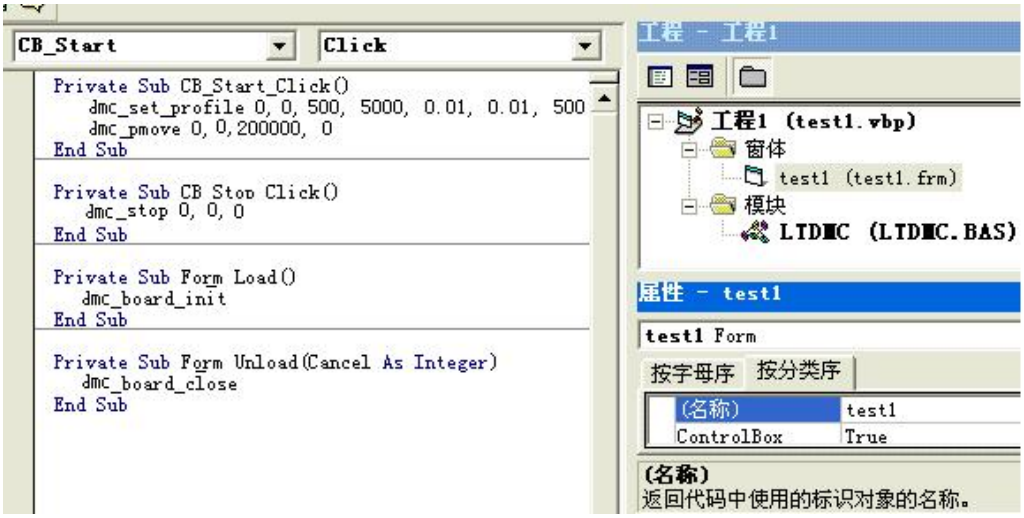


图 6.5　程序中调用运动控制卡库函数

7）程序编写完成。运行程序，显示界面如图 6.6 所示。按下“启动”按钮，第 0 轴就会输出长度为 200000 的脉冲；运动中可以按下“停止”按钮，便会减速停止脉冲输出。

图 6.6　程序运行界面（VB）

6.3 采用 VC 6.0 开发应用软件的方法

下面以 Visual C++ 6.0 环境下编写一个点位运动的应用软件为例，讲解用 VC 开发应用软件的一般方法。

1）打开 Visual C++ 6.0。

2）新建一个工程。

3）选择 MFC APPWizard(exe)。

4）选择工程保存路径，如：E:\。

5）输入工程名,如：test1。如图 6.7。

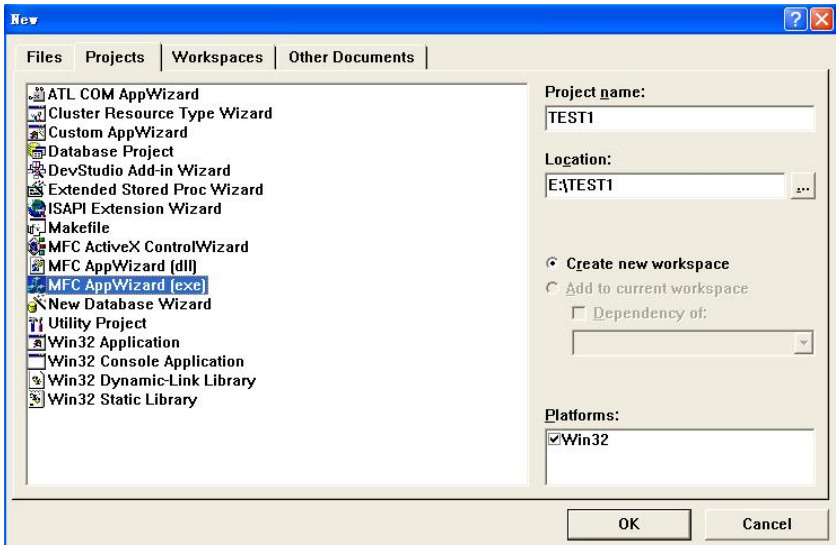


图 6.7 创建新工程

6）在应用程序类型中选择“基于对话”，按“完成”键，建立工程。

7）给对话框进行简单的修改，增加按钮“启动”和“停止”；并分别命名为“IDC_BUTTON_Start”和“IDC_BUTTON_Stop”，如图 6.8 所示。

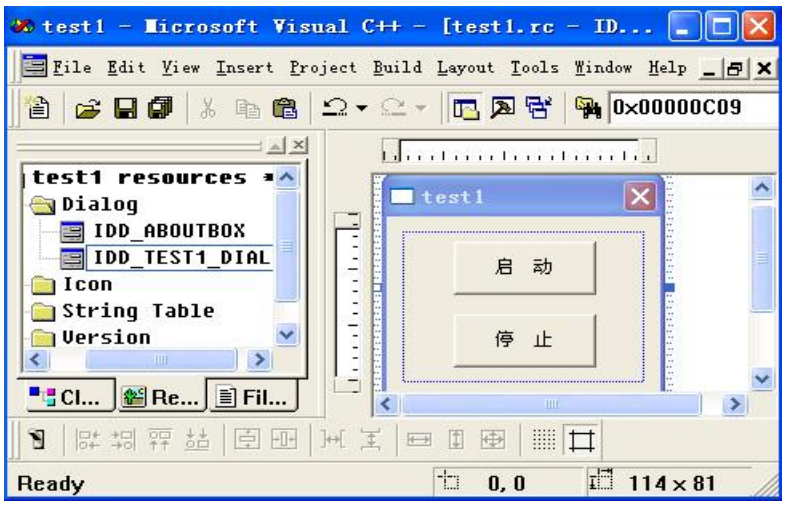


图 6.8 修改对话框

8）在相应的目录下找到 LTDMC.h、LTDMC.lib、LTDMC.dll 和 PVT.dll 文件，拷贝到 E:\tes1 目录。

9）在菜单中选择“工程”->“添加工程”->“文件”，将 LTDMC.lib 文件加入到工程中。

10）打开 test1.cpp 文件，在程序开始部分添加相应语句：#include “LTDMC.h”，如图 6.9 所示。

11）在 CTest1Dlg::OnInitDialog()函数中添加代码：dmc_board_init();如图 6.10。

12）如图 6.11 所示，在 Ctest1Dlg 中添加一个成员函数 OnCancel,在 OnCancel 函数中添加代码如下：

```
dmc_board_close();
CDialog::OnCancel();
```

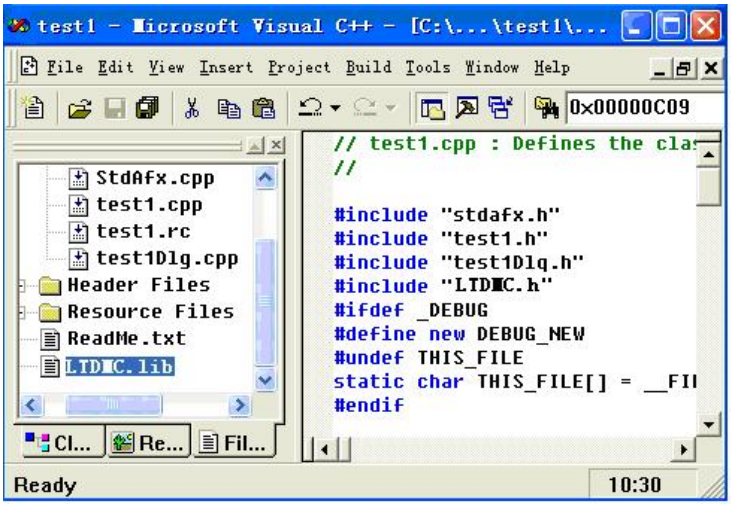


图 6.9　程序增加头文件

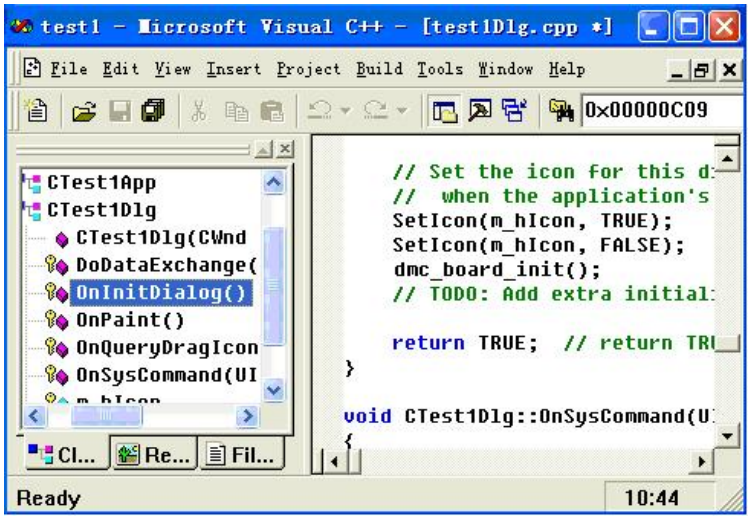


图 6.10　程序增加初始化函数

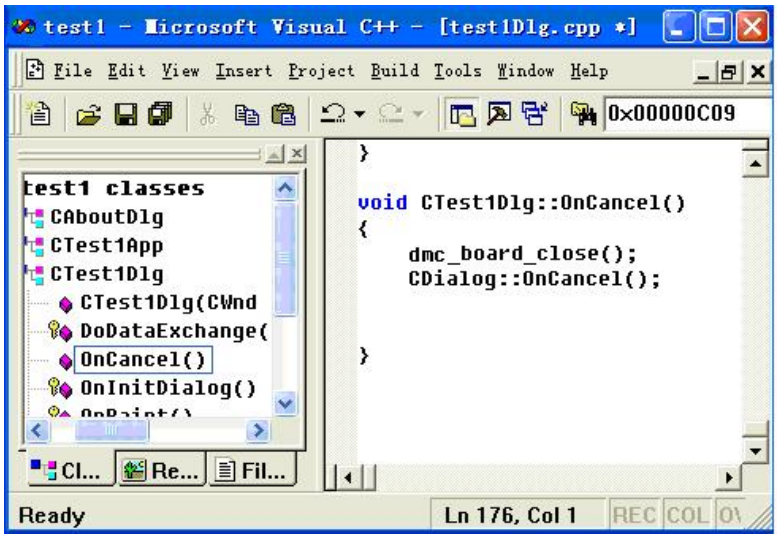


图 6.11　程序增加 OnCancel 函数

13）双击“启动”按钮在按钮点击事件中输入代码如下：

```
dmc_set_profile(0,0,500,5000,  0.01,0.01,500);
dmc_pmove(0,0,200000,0);
```

双击“停止”按钮在按钮点击事件中输入代码：

```
dmc_stop(0,0,0);
```

如图 6.12 所示。

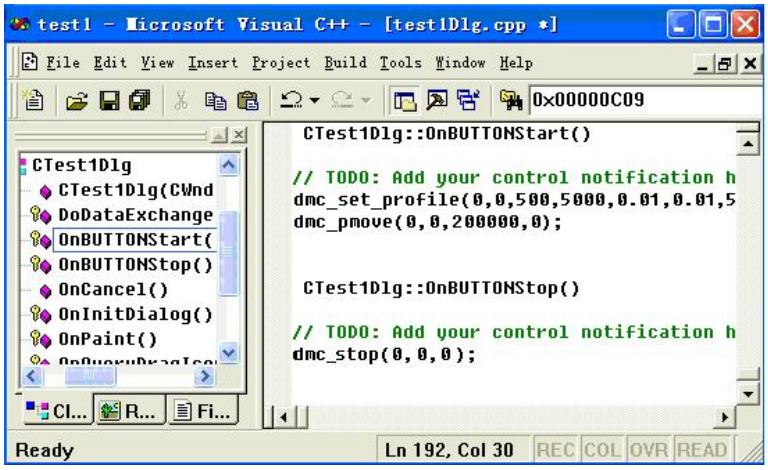


图 6.12　程序中调用运动控制卡库函数

14）编译程序后，运行程序，显示图 6.13 所示的界面。按下“启动”按钮，第 0 轴就会输出长度为 200000 的脉冲；运动中可以按下“停止”按钮便会减速停止脉冲输出。

图 6.13　程序运行界面

第 7 章　实现基本功能的方法及相关函数

本章介绍采用 Visual BASIC 语言通过调用相关函数实现 DMC3000 系列运动控制卡基本功能的方法。

注意：在编程之前，一定要用控制卡 Motion 软件检测硬件系统，确保硬件接线正确。

7.1　限位开关及急停开关的设置

在设备进行运动功能调试之前，必须确保安全机制的有效。

限位开关在运动平台出现超出行程的运动时，起到限制作用，使电机减速或紧急停止，提高设备运行时的安全性能。在使用运动控制卡进行运动控制之前，必须保证限位开关的有效性。

急停开关在运动过程中出现意外的运动时，能起到紧急停止运动的功能，提高设备运行时的安全性能。在使用运动控制卡进行运动控制之前，必须保证急停开关的有效性。

7.1.1　限位开关的设置

DMC3000 系列卡提供了限位开关设置函数 dmc_set_el_mode 来设置限位功能。用户必须根据设备的限位开关硬件接线，来设置限位开关工作的有效电平。

相关函数如下表 7.1 所示。

表 7.1　限位开关设置相关函数说明

名称	功能	参考
dmc_set_el_mode	设置限位开关信号	8.5 节
dmc_get_el_mode	读取限位开关信号设置	

假设设备正常运动时限位开关为高电平，运动平台碰到限位开关时为低电平，则此时应设置控制卡限位开关的有效电平为低电平。

例程 7.1：设置限位开关为低电平

```
……
Dim MyCardNo,Myaxis,Myel_enable,Myel_logic,Myel_mode As Integer
MyCardNo = 0          ’卡号
Myaxis = 0            ’轴号
```

```
Myel_enable = 1      ’正负限位使能
Myel_logic = 0       ’正负限位低电平有效
Myel_mode = 0        ’正负限位停止方式为立即停止
dmc_set_el_mode MyCardNo,Myaxis,Myel_enable,Myel_logic,Myel_mode   ’设置 0 号轴限位信号
……
```

7.1.2 急停开关的设置

DMC3000 系列卡提供了急停开关设置函数 dmc_set_emg_mode 来设置限位功能。用户必须根据设备的急停开关硬件接线，来设置急停开关工作的有效电平。

相关函数如下表 7.2 所示。

表 7.2 急停开关设置相关函数说明

名称	功能	参考
dmc_set_emg_mode	设置急停开关信号	8.21 节
dmc_get_emg_mode	读取急停开关信号设置	

DMC3000 系列卡没有专用于 EMG 急停开关的硬件接口，用户需要根据自己的需求对轴 IO 进行映射配置，对应接口电路进行接线，然后调用急停开关设置函数进行设置。

假设使用控制卡的通用输入口 0 作为所有轴的急停信号。设备正常运动时急停开关为高电平，当急停开关为低电平时紧急停止运动，则此时应设置控制卡急停开关的有效电平为低电平。具体设置见例程 7.3，关于轴 IO 映射功能的实现详见 7.14 节 轴 IO 映射功能的实现。

例程 7.2：设置通用输入口 0 为所有轴的急停信号，低电平有效

```
……
Dim CardNo,Axis,IoType, MapIoType, MapIoIndex,Myenable,Mylogic As Integer
Dim Filter As Double
CardNo = 0          ’卡号
For Axis = 0 To 7    ’循环，依次对 0~7 号轴进行设置（DMC3600 时为 0 To 5，因其只有 6 个轴）
    dmc_set_AxisIoMap CardNo, Axis, 3, 6, 0, 0.01
    ’设置轴 IO 映射，将通用输入 0 作为各轴的急停信号，EMG 信号滤波时间为 0.01 秒
Next Axis
Myenable = 1         ’急停信号使能
Mylogic = 0          ’急停信号低电平有效
For Axis = 0 To 7    ’循环，依次对 0~7 号轴进行设置（DMC3600 时为 0 To 5，因其只有 6 个轴）
  dmc_set_emg_mode CardNo, Axis, Myenable,Mylogic    ’设置 EMG 信号使能，低电平有效
Next Axis
……
```

7.2 回原点运动的实现

7.2.1 回原点步骤

在进行精确的运动控制之前，需要设定运动坐标系的原点。运动平台上都设有原点传感器（也称为原点开关）。寻找原点开关的位置并将该位置设为平台的坐标原点的过程即为回原点运动。DMC3000 系列控制卡共提供多种回原点方式。

回原点运动主要步骤如下：

1）使用 dmc_set_home_pin_logic 函数设置原点开关的有效电平；

2）使用 dmc_set_homemode 函数设置回原点方式；

3）设置回原点运动的速度曲线；

4）设置回零偏移量、回零完成是否清零及非限位回零方式遇限位是否反找；

5）使用 dmc_home_move 函数执行回原点运动；

7.2.2 回原点方式

DMC3000 系列控制卡提供多种回原点运动的方式（DMC3C00 后四轴只支持 0、1、2、10、11、12 六种回零模式）：

方式 0：一次原点回零

该方式以设定速度回原点；适合于行程短、安全性要求高的场合。动作过程为：电机从初始位置以恒定速度向原点方向运动，当到达原点开关位置，原点信号被触发，电机立即停止（过程 0）；将停止位置设为原点位置，如图 7.1 所示。

图 7.1 一次回零方式示意图

方式 1：一次原点回零加回找

该方式先进行方式 1 运动，完成后再反向回找原点开关的边缘位置，当原点信号第一次无效的时候，电机立即停止；将停止位置设为原点位置如图 7.2 所示。

方式 2：两次原点回零

如图 7.3 所示，该方式为方式 0 和方式 1 的组合。先进行方式 1 的回零加反找，完成后再进

行方式 0 的一次回零。可参见方式 1 和方式 2 的说明。

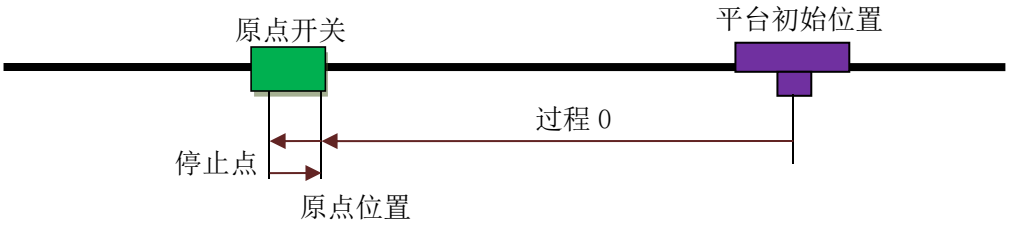


图 7.2　一次回零加回找方式示意图

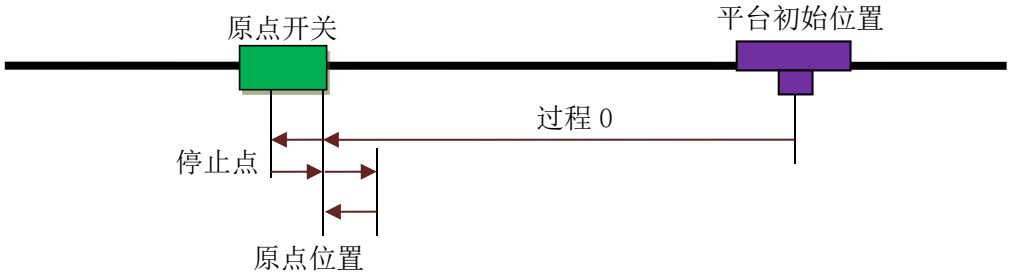


图 7.3　二次回零方式示意图

方式 3：一次原点回零后再同向找 1 个 EZ 信号后回零

该方式在回原点运动过程中，当找到原点信号后，还要等待该轴的 EZ 信号出现，此时电机停止。回原点过程如图 7.4 所示。

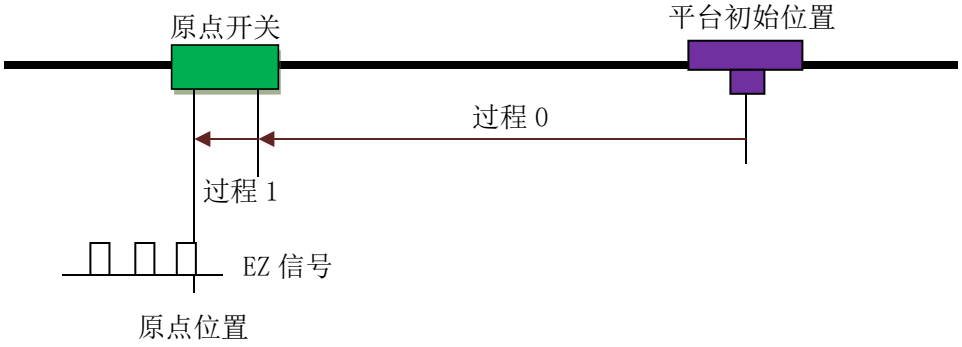


图 7.4　一次回零后再找 1 个 EZ 信号回零方式示意图

方式 4：记 1 个 EZ 信号回零

该方式在回原点运动过程中，当检测到该轴的 EZ 信号出现一次后，此时电机停止。回原点过程如图 7.5 所示。

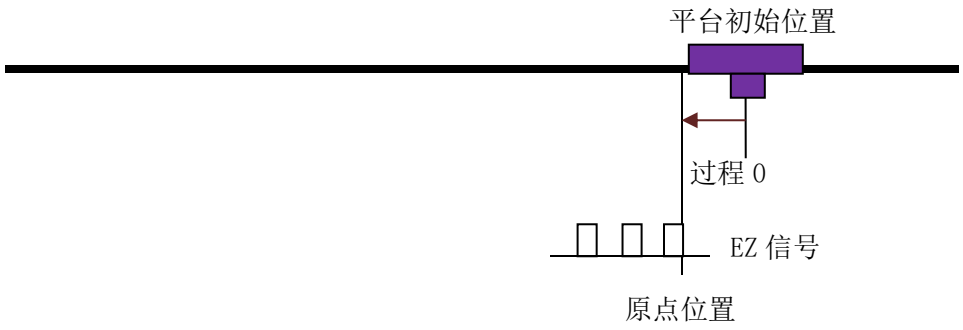


图 7.5　记 1 个 EZ 信号回零方式示意图

方式 5：一次原点回零再反找 EZ 信号

该方式在回原点运动过程中，当找到原点信号后，减速停止，然后以反找速度反向找到 EZ 生效此时电机停止。回原点过程如图 7.6 所示。

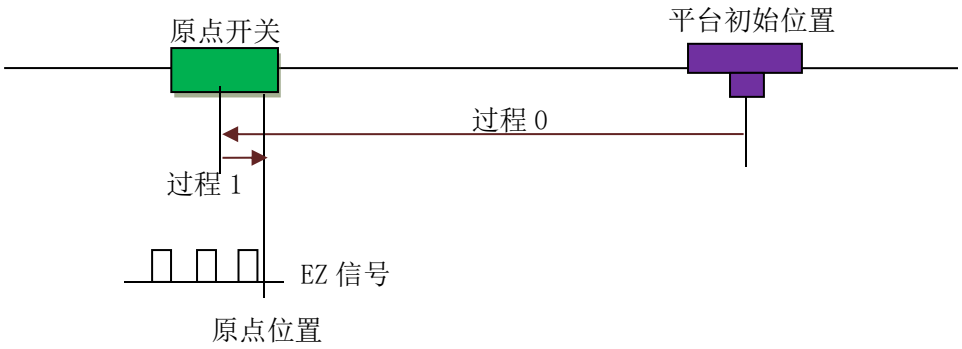


图 7.6　一次回零反找一个 EZ 进行回零

方式 6：原点锁存

如图 7.7 所示，电机先以设定速度回原点，当原点开关边沿触发时，将当前位置锁存下来，同时电机减速停止。电机减速停止完成后再反向回找锁存位置，运动到锁存位置，电机停止。

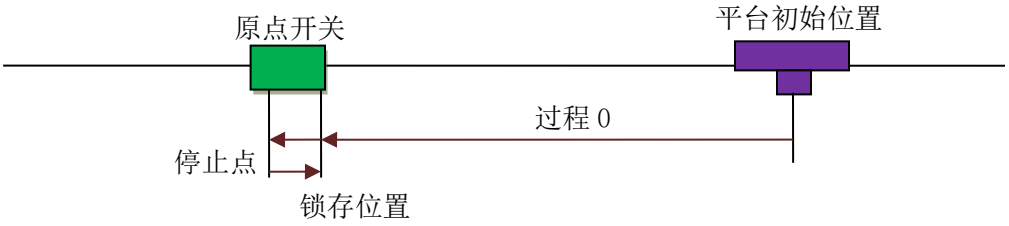


图 7.7　原点锁存回零方式示意图

方式 7：原点锁存加同向 EZ 锁存

该方式先以方式 6 执行一次原点锁存回零，完成后继续沿设定回零方向运行到 EZ 信号产生，EZ 信号产生时锁存当前位置并执行减速停，电机减速停止之后再反向回找 EZ 的锁存位置，运动

到锁存位置，电机停止。回原点过程如图 7.8 所示。

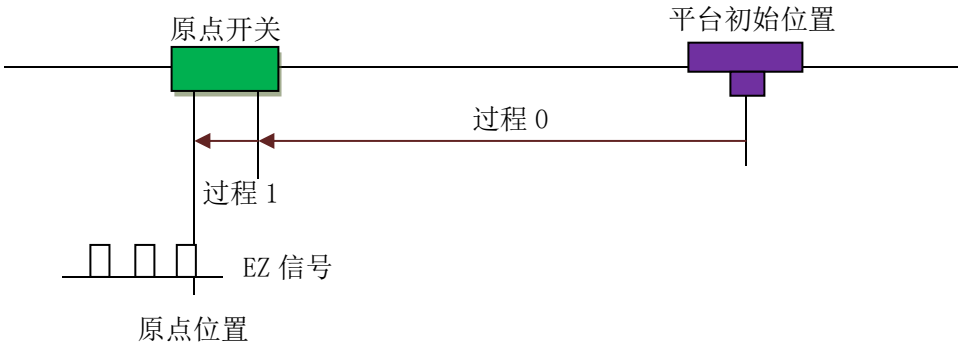


图 7.8　原点锁存加同向 EZ 锁存回零方式示意图

方式 8：单独记一个 EZ 锁存

在回零过程中检测到 EZ 有效边沿出现，锁存当前位置，执行减速停，电机减速停止之后再反向回找 EZ 的锁存位置，运动到锁存位置，电机停止。回原点过程如图 7.9 所示。

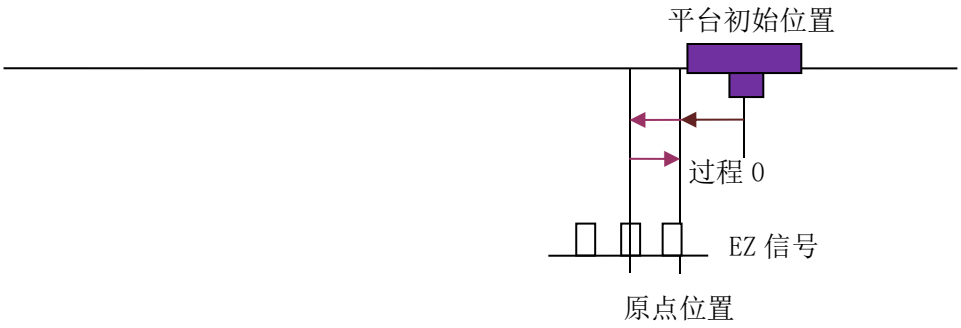


图 7.9　单独记一个 EZ 锁存回零方式示意图

方式 9：原点锁存加反向 EZ 锁存

该方式先以方式 6 执行一次原点锁存回零，完成后以与设定回零方向相反的方向运行到 EZ 信号产生，EZ 信号产生时锁存当前位置并执行减速停，电机减速停止之后再反向回找 EZ 的锁存位置，运动到锁存位置，电机停止。回原点过程如图 7.10 所示。

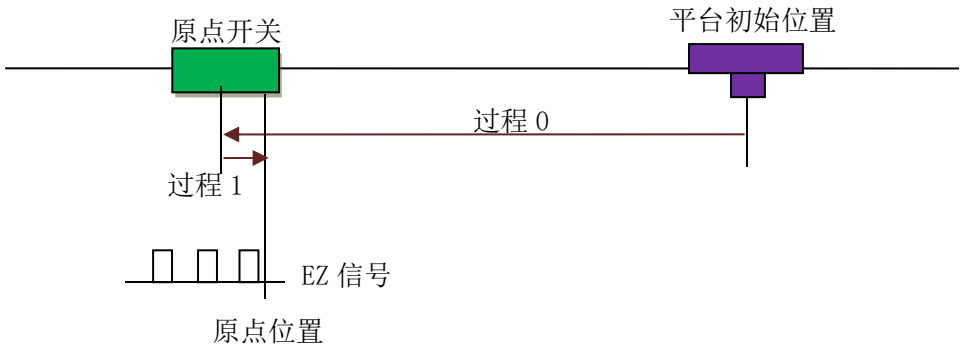


图 7.10　原点锁存加反向 EZ 锁存进行回零

方式 10：一次限位回零

该方式以设定速度回原点；适合于行程短、安全性要求高的场合。动作过程为：电机从初始位置以恒定速度向限位方向运动，当到达限位开关位置，限位信号被触发，电机立即停止（过程 0）；将停止位置设为原点位置，如图 7.11 所示。

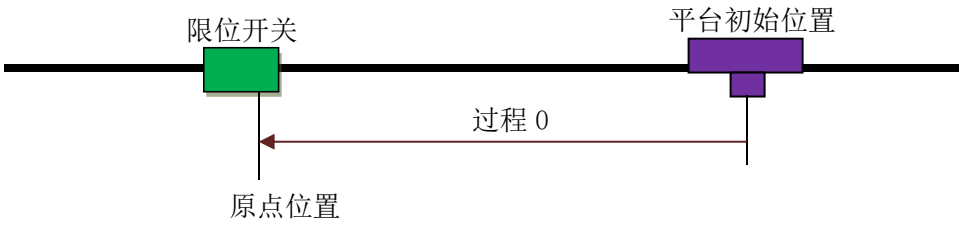


图 7.11　一次限位方式示意图

方式 11：一次限位回零加回找

该方式先进行方式 1 运动，完成后再反向回找原点开关的边缘位置，当原点信号第一次无效的时候，电机立即停止；将停止位置设为原点位置如图 7.12 所示。

方式 2：两次限位回零

如图 7.13 所示，该方式为方式 1 和方式 2 的组合。先进行方式 2 的回零加反找，完成后再进行方式 1 的一次回零。可参见方式 1 和方式 2 的说明。

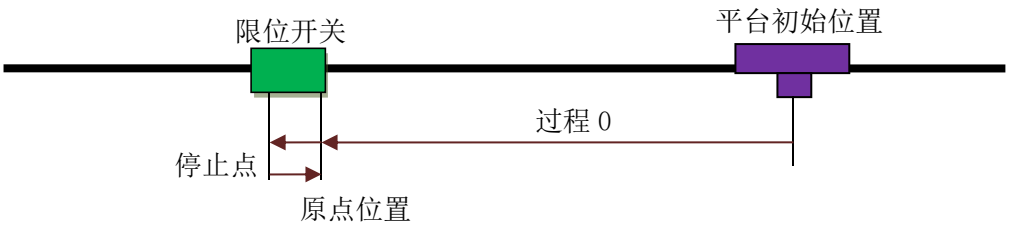


图 7.12　一次回零加回找方式示意图

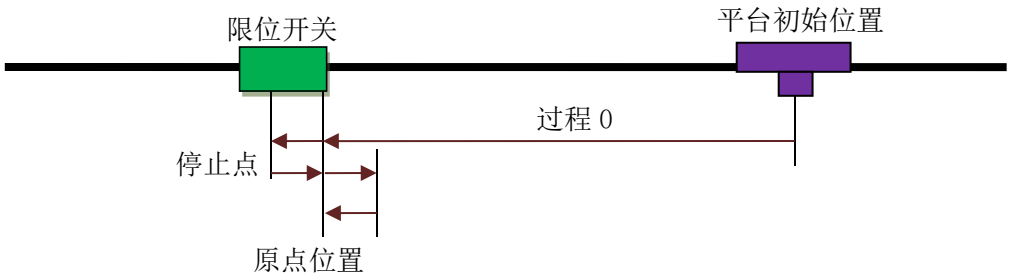


图 7.13　二次回零方式示意图

相关函数如下表 7.3 所示。

表 7.3　回原点相关函数说明

名称	功能	参考
dmc_set_home_pin_logic	设置原点信号的有效电平	8.3 节
dmc_set_homemode	选择回原点模式	
dmc_set_home_el_return	设置回零遇限位是否反找	
dmc_set_home_position	设置回零偏移量及清零模式	
dmc_home_move	按指定的方向和速度方式开始回原点	
dmc_get_home_result	读取回零状态	

注意：DMC3000 系列控制卡 3XX201615 版本及更高固件支持回零完成是否自动清零以及设置偏移量等，具体可参考 8.3 相关函数调用；

例程 7.3：方式 1 低速回原点

```
……
Dim MyCardNo,Myaxis,Myorg_logic,Myhome_dir,Mymode,MyEZ_count As Integer
Dim Myfilter,Myvel_mode As Double
MyCardNo = 0        ’卡号
Myaxis = 0          ’轴号
Myorg_logic = 0     ’原点信号低电平有效
Myfilter = 0        ’保留参数
dmc_set_home_pin_logic MyCardNo,Myaxis,Myorg_logic, Myfilter   ’设置 0 号轴原点信号
Myhome_dir = 0      ’负方向回零
Myvel_mode = 0      ’回零速度模式为低速
```

```
Mymode = 0          ’回零模式为方式 1，一次回零
MyEZ_count = 0      ’保留参数
dmc_set_homemode MyCardNo,Myaxis, Myhome_dir, Myvel_mode, Mymode, MyEZ_count
                                              ’设置 0 号轴回零模式
dmc_set_profile MyCardNo,Myaxis,500,1000,0.1,0.1,500    ’设置 0 号轴梯形速度曲线参数
dmc_home_move MyCardNo,Myaxis                    ’0 号轴按照设置的模式进行回零运动
While (dmc_check_done (MyCardNo,Myaxis) = 0)     ’检测运动状态，等待回原点动作完成
   DoEvents
Wend
dmc_set_position MyCardNo,Myaxis,0          ’设置 0 号轴的指令脉冲计数器绝对位置为 0
……
```

7.3 点位运动的实现

DMC3000 系列卡在描述运动轨迹时可以用绝对坐标也可以用相对坐标，如图 7.14 所示。两种模式各有优点，如：在绝对坐标模式中用一系列坐标点定义一条曲线，如果要修改中间某点坐标时，不会影响后续点的坐标；在相对坐标模式中，用一系列坐标点定义一条曲线，用循环命令可以重复这条曲线轨迹多次。

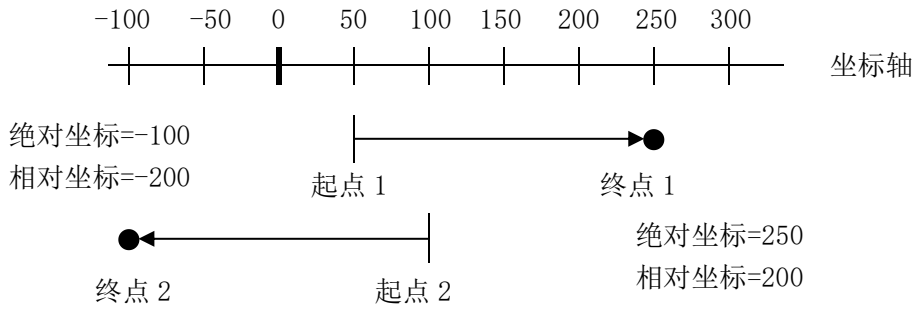


图 7.14　绝对坐标与相对坐标中轨迹终点的不同表达方式

在 DMC3000 系列函数库中距离或位置的单位为 pulse；速度单位为 pulse/s。

DMC3000 系列卡在执行点位运动控制指令时，可使电机按照梯形速度曲线或 S 形速度曲线进行点位运动。

7.3.1 梯形速度曲线下的点位运动

梯形速度曲线是位置控制中最基本的速度控制方式。参见图 2.3。

相关函数如表 7.4 所示。

表 7.4　梯形速度点位运动相关函数说明

名称	功能	参考
dmc_set_profile	设置单轴运动速度曲线	8.8 节
dmc_pmove	指定轴点位运动	8.9 节
dmc_check_done	检测指定轴的运动状态	8.7 节

例程 7.4：执行以梯形速度曲线作点位运动

```
……
Dim MyCardNo,Myaxis,Myposi_mode, Mys_mode As Integer
Dim MyMin_Vel,MyMax_Vel,MyTacc,MyTdec,MyStop_Vel, Mys_para As Double
Dim MyDist As Long
MyCardNo = 0        ’卡号
Myaxis = 0          ’轴号
MyMin_Vel = 500     ’设置起始速度为 500pulse/s
MyMax_Vel = 6000    ’设置最大速度为 6000pulse/s
MyTacc = 0.02       ’设置加速时间为 0.02s
MyTdec = 0.01       ’设置减速时间为 0.01s
MyStop_Vel = 500    ’设置停止速度为 500pulse/s
dmc_set_profile MyCardNo,Myaxis, MyMin_Vel,MyMax_Vel,MyTacc,MyTdec,MyStop_Vel
                    ’设置 0 号轴速度曲线参数
Mys_mode = 0        ’保留参数
Mys_para = 0        ’ S 段时间为 0，即没有 S 段运动
dmc_set_s_profile MyCardNo,Myaxis, Mys_mode, Mys_para   ’设置 0 号轴 S 段参数为 0
MyDist = 50000      ’设置运动距离为 50000pulse
Myposi_mode = 0     ’设置运动模式为相对坐标模式
dmc_pmove MyCardNo,Myaxis, MyDist, Myposi_mode          ’0 号轴定长运动
While(dmc_check_done(MyCardNo,Myaxis) = 0)              ’判断 0 轴运动状态
    DoEvents
Wend
……
```

在点位运行过程中，最大速度 Max_Vel 和目标位置 Dist 均可以实时改变，如图 7.15 所示。若在减速时改变目标位置，电机的速度将如图 7.16 所示发生变化。

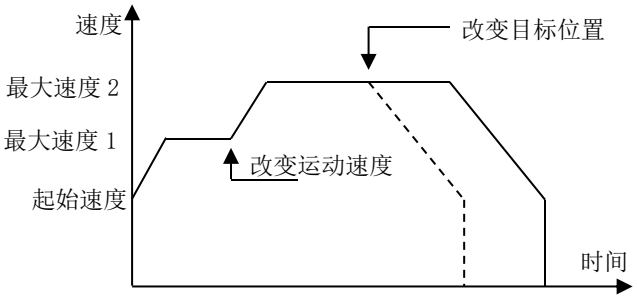


图 7.15　改变速度及改变目标位置

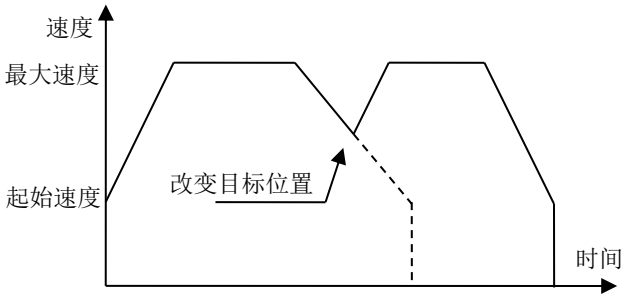


图 7.16　减速时改变目标位置

实现这 2 个功能的函数如表 7.5 所示：

表 7.5　梯形速度下改变速度、终点的相关函数说明

名称	功能	参考
dmc_change_speed	在线改变指定轴的当前运动速度	8.9 节
dmc_reset_target_position	在线改变指定轴的当前目标位置	

例程 7.5：改变速度、改变终点位置

```
……
Dim MyCardNo,Myaxis,Myposi_mode As Integer
Dim MyCurr_Vel,MyTaccdec As Double
Dim Mydist As Long
MyCardNo = 0        ’卡号
Myaxis = 0          ’轴号
dmc_set_profile MyCardNo,Myaxis,500,6000,0.01,0.02,500    ’设置梯形速度曲线参数
dmc_pmove MyCardNo,Myaxis,50000,0    ’0 号轴定长运动，运动距离 50000pulse、相对坐标模式
If(“改变速度条件”)                    ’如果改变速度条件满足，则执行改变速度命令
   MyCurr_Vel = 9000                  ’设置新的速度为 9000pulse/s
   MyTaccdec = 0                  ’保留参数
   dmc_change_speed MyCardNo,Myaxis, MyCurr_Vel, MyTaccdec       ’执行在线变速指令
End If
If(“改变终点位置条件”)          ’如果改变终点位置条件满足，则执行改变终点位置命令
   Mydist = 55000                 ’改变终点位置为 55000pulse
   Myposi_mode = 0                ’保留参数
   dmc_reset_target_position MyCardNo,Myaxis, Mydist, Myposi_mode  ’  执行在线变位指令
End If
……
```

7.3.2 S 形速度曲线运动模式

梯形速度曲线较简单；而 S 速度曲线运动更平稳。参见图 2.4。

相关函数如表 7.6 所示。

表 7.6 S 形速度控制相关函数说明

名称	功能	参考
dmc_set_profile	设置单轴运动速度曲线	8.8 节
dmc_set_s_profile	设置 S 段曲线参数值	
dmc_pmove	指定轴点位运动	8.9 节
dmc_check_done	检测指定轴的运动状态	8.7 节

例程 7.6：执行以 S 形速度曲线作点位运动

```
……
Dim MyCardNo,Myaxis,Mys_mode As Integer
Dim Mys_para As Double
MyCardNo = 0        ’卡号
Myaxis = 0          ’轴号
Mys_mode = 0        ’保留参数
Mys_para = 0.02   ’  S 段时间为 0.02s
dmc_set_profile MyCardNo,Myaxis,500,6000,0.05,0.05,500    ’设置 0 号轴速度曲线参数
dmc_set_s_profile MyCardNo,Myaxis, Mys_mode, Mys_para     ’设置 0 号轴 S 段参数
dmc_pmove 0,0,50000,0           ’ 0 号轴定长运动，运动距离为 50000pulse、相对坐标模式
While(dmc_check_done(MyCardNo,Myaxis) = 0)                ’判断 0 轴运动状态
    DoEvents
Wend
……
```

如果因为距离太短或加速太慢原因导致电机速度在加速段不能升至设定的最大值 Max_Vel 时，理论上加速段将突然切换至减速段，从而引起该轴出现较大震动。为了避免出现这种问题，DMC3000 系列运动控制卡内置有自动调整功能，使得加减速段的过渡保持平滑，如图 7.17 所示。

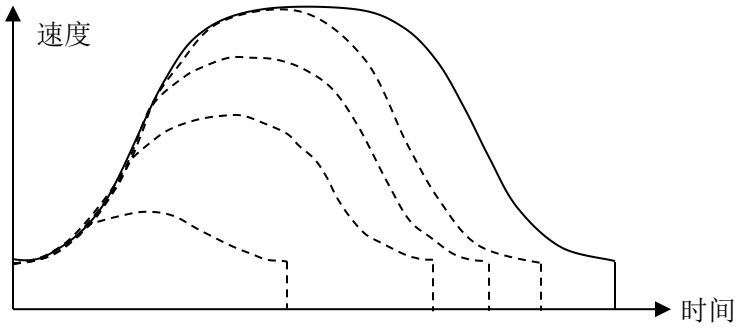


图 7.17　自动降速避免尖三角形

在 S 形速度曲线下的点位运动过程中，也可以调用 dmc_change_speed 和 dmc_reset_target_position 函数实时改变运行速度和目标位置。但多轴插补运行情况下不能实时改变运行速度和目标位置。

7.3.3　多轴联动

多轴同时做点位运动，称之为多轴联动。

DMC3000 系列卡可以控制多个电机同时执行 dmc_pmove 这类单轴运动函数。所谓同时执行，是在程序中顺序调用 dmc_pmove 等函数，因为程序执行速度很快，在几微秒内电机都开始运动，感觉是同时开始运动。

多轴联动在各轴速度设置不当时，各轴停止时间不同、在起点与终点之间运动的轨迹也不是直线。参见图 2.2。

如果从起点到终点都需要按照规定的路径运动，就必须采用插补运动功能。

7.4　连续运动的实现

连续运动中，DMC3000 系列卡可以控制电机以梯形或 S 形速度曲线在指定的加速时间内从起始速度加速至最大速度，然后以该速度一直运行，直至调用停止指令或者该轴遇到限位信号才会按启动时的速度曲线减速停止。相关函数如表 7.7 所示。

表 7.7　连续运动相关函数说明

名称	功能	参考
dmc_vmove	指定轴连续运动	8.9 节
dmc_stop	指定轴停止运动	8.7 节

在执行连续运动过程中，可以调用 dmc_change_speed 实时改变速度。注意：在以 S 形速度曲线连续运动时，改变最大速度最好在加速过程已经完成的恒速段进行。图 7.18 和图 7.19 为梯形和 S 形速度曲线下连续运动中变速和减速停止过程的速度曲线。

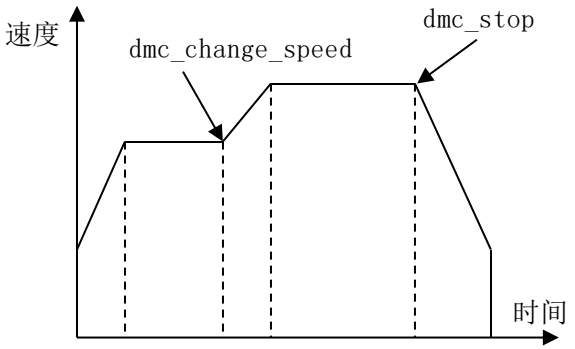


图 7.18　梯形运动中变速

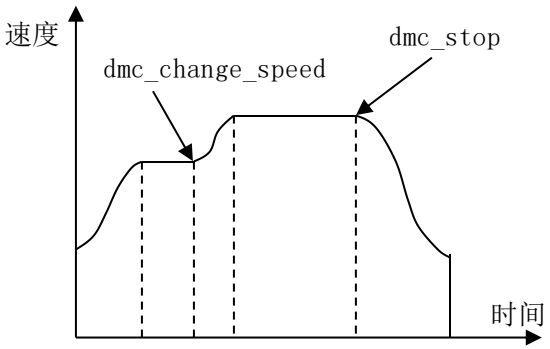


图 7.19　S 形运动中变速

例程 7.7：以 S 形速度曲线加速的连续运动及变速、停止控制

```
……
Dim MyCardNo,Myaxis,Mydir,Mystop_mode As Integer
Dim MyCurr_Vel As Double
MyCardNo = 0        ’卡号
Myaxis = 0          ’轴号
Mydir = 1           ’设置连续运动方向为正方向
dmc_set_profile MyCardNo,Myaxis,100,1000,0.1,0.1,100   ’设置 0 号轴运动参数
dmc_set_s_profile MyCardNo,Myaxis,0,0.002              ’设置 0 号轴 S 段时间为 0.002s
dmc_vmove MyCardNo,Myaxis,Mydir                        ’0 号轴执行连续运动
If(“改变速度条件”)                    ’如果改变速度条件满足，则执行改变速度命令
   MyCurr_Vel = 1200                   ’设置新的速度
   dmc_change_speed MyCardNo,Myaxis, MyCurr_Vel,0      ’执行在线变速指令
End If
If(“停止条件”)                  ’如果运动停止条件满足，则执行减速停止命令
   Mystop_mode = 0              ’设置停止模式为减速停止
   dmc_stop MyCardNo,Myaxis, Mystop_mode               ’0 号轴减速停止
End If
……
```

7.5　插补运动的实现

插补运动是为了实现轨迹控制，运动控制卡按照一定的控制策略控制多轴联动，使运动平台用微小直线段精确地逼近轨迹的理论曲线，保证运动平台从起点到终点上的所有轨迹点都控制在

允许误差范围内。

7.5.1 直线插补运动

DMC3C00 卡可以进行任意 2~11 轴直线插补，DMC3800 卡可以进行任意 2~8 轴直线插补，DMC3600 卡可以进行任意 2~6 轴直线插补，DMC3400A 卡可以进行任意 2~4 轴直线插补，插补计算由控制卡的硬件执行，用户只需将插补运动的速度、加速度、终点位置等参数写入相关函数即可。

1. 两轴直线插补

如图 7.20 所示，2 轴直线插补从 P0 点运动至 P1 点，X、Y 轴同时启动，并同时到达终点；X、Y 轴的运动速度之比为ΔX：ΔY，二轴合成的矢量速度为：

$$\frac{\Delta P}{\Delta t}=\sqrt{\left(\frac{\Delta X}{\Delta t}\right)^2+\left(\frac{\Delta Y}{\Delta t}\right)^2}$$

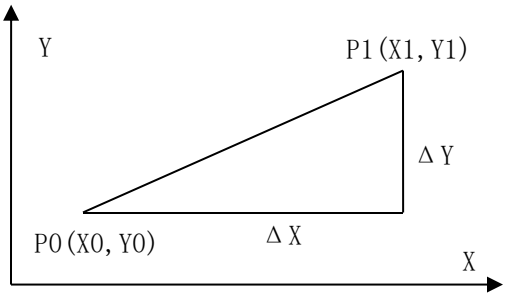


图 7.20 两轴直线插补

2. 三轴直线插补

如图 7.21 所示，在 X、Y、Z 轴内直线插补，从 P0 点运动至 P1 点。插补过程中 3 轴的速度比为ΔX：ΔY：ΔZ，三轴合成的矢量速度为：

$$\frac{\Delta P}{\Delta t}=\sqrt{\left(\frac{\Delta X}{\Delta t}\right)^2+\left(\frac{\Delta Y}{\Delta t}\right)^2+\left(\frac{\Delta Z}{\Delta t}\right)^2}$$

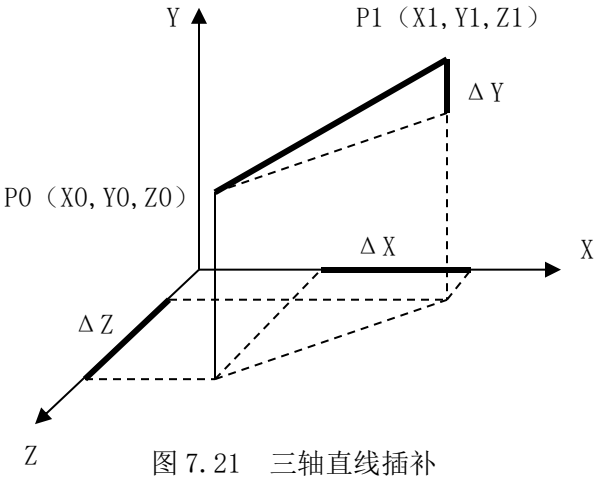


图 7.21 三轴直线插补

3. 四轴直线插补

4 轴插补可以理解为在 4 维空间里的直线插补。一般情况是 3 个轴进行直线插补，另一个旋转轴也按照一定的比例关系和这条空间直线一起运动。其合成矢量速度为：

$$\frac{\Delta P}{\Delta t}=\sqrt{\left(\frac{\Delta X}{\Delta t}\right)^2+\left(\frac{\Delta Y}{\Delta t}\right)^2+\left(\frac{\Delta Z}{\Delta t}\right)^2+\left(\frac{\Delta U}{\Delta t}\right)^2}$$

调用直线插补函数时，调用者需提供矢量速度，包括其最大矢量速度 Max_Vel 和加减速时间参数。相关函数如表 7.8 所示：

表 7.8 直线插补运动相关函数说明

名称	功能	参考
dmc_set_vector_profile_multicoor	设置插补速度	8.10 节

名称	功能	参考
dmc_line_multicoor	直线插补运动	8.11 节

例程 7.8： XY 轴直线插补

```
……
Dim MyCardNo,MyCrd,MyaxisNum,AxisArray(2),Myposi_mode As Integer
Dim MyMin_Vel,MyMax_Vel,MyTacc,MyTdec,MyStop_Vel As Double
Dim Dist(2) As Long
MyCardNo = 0            ’卡号
MyCrd = 0               ’参与插补运动的坐标系
MyMin_Vel = 0           ’保留参数
MyMax_Vel = 5000        ’插补运动最大矢量速度 5000pulse/s
MyTacc = 0.1            ’插补运动加减速时间 0.1s
MyTdec = 0              ’保留参数
MyStop_Vel = 0          ’保留参数
dmc_set_vector_profile_multicoor MyCardNo, MyCrd, MyMin_Vel, MyMax_Vel, MyTacc, MyTdec,
MyStop_Vel              ’设置插补速度曲线参数
MyaxisNum = 2           ’插补运动轴数为 2
AxisArray(0) = 0        ’定义插补 0 轴为 X 轴
AxisArray(1) = 1        ’定义插补 1 轴为 Y 轴
Dist(0) = 30000         ’定义 X 轴运动距离为 30000pulse
Dist(1) = 40000         ’定义 Y 轴运动距离为 40000pulse
Myposi_mode = 0         ’插补运动模式为相对坐标模式
dmc_line_multicoor MyCardNo,MyCrd,MyaxisNum,AxisArray(0),Dist(0),Myposi_mode
                        ’执行直线插补运动
While(dmc_check_done_multicoor(MyCardNo,MyCrd) = 0)    ’判断坐标系 0 状态
    DoEvents
Wend
……
```

该例程使 X，Y 轴进行相对坐标模式直线插补运动，其相关参数为：

ΔX = 30000 pulse

ΔY = 40000 pulse

最大矢量速度 = 5000 pulse/s （0 轴,1 轴分速度为 3000，4000 pulse/s）

梯形加减速时间 = 0.1 s

7.5.2 圆弧插补运动

DMC3000 系列卡的任意两轴之间可以进行圆弧插补，圆弧插补分为相对位置圆弧插补和绝对

位置圆弧插补，运动的方向分为顺时针（CW）和逆时针（CCW），参见图 2.6。

相关函数如表 7.9 所示。

表 7.9　圆弧插补相关函数说明

名称	功能	参考
dmc_set_vector_profile_multicoor	设置插补速度	8.10 节
dmc_arc_move_multicoor	圆弧插补运动	8.11 节

例程 7.9：XY 轴间的圆弧插补

```
……
Dim MyCardNo,MyCrd,AxisArray(2),MyArc_Dir,Myposi_mode As Integer
Dim Pos(2), Cen(2) As Long
MyCardNo = 0          ’定义卡号
MyCrd = 0             ’定义参与插补运动的坐标系
AxisArray(0)=0        ’定义 0 轴为插补 X 轴
AxisArray(1)=1        ’定义 1 轴为插补 Y 轴
Pos(0) = 5000 : Pos(1) = -5000       ’设置终点坐标
Cen(0) = 5000 : Cen(1) = 0           ’设置圆心坐标
MyArc_Dir = 0         ’设置圆弧方向为顺时针
 Myposi_mode = 1      ’设置圆弧插补模式为绝对坐标模式
dmc_set_vector_profile_multicoor MyCardNo,MyCrd,0,3000,0.1,0,0    ’设置矢量速度曲线
dmc_arc_move_multicoor MyCardNo,MyCrd, AxisArray(0),Pos(0),Cen(0),MyArc_Dir, Myposi_mode
                          ’XY 轴执行顺时针方向绝对圆弧插补运动，
While(dmc_check_done_multicoor(MyCardNo,MyCrd) = 0)    ’判断坐标系 0 状态
    DoEvents
Wend
……
```

7.6　PVT 运动功能的实现

DMC3000 系列卡共提供四种 PVT 模式，分别为 PTT、PTS、PVT、PVTS 模式。其中 PTT、PTS 运动模式用于单轴速度规划功能，PVT、PVTS 运动则用于多轴轨迹规划功能，用户可以根据实际需求选择适合的 PVT 模式。

注　意：DMC3C00 后四轴不支持 PVT 功能

7.6.1 单轴任意速度规划功能的实现

DMC3000 系列卡中提供了两种 PVT 模式来实现单轴任意速度规划的功能，分别为 PTT 运动模式和 PTS 运动模式。PTT 运动模式是用于单轴梯形速度的规划，而 PTS 运动模式则用于单轴 S 形速度的规划。

单轴任意速度规划功能的介绍可参见 2.1.4.1 节 单轴任意速度规划功能。

7.6.1.1 PTT 运动模式

PTT 模式非常灵活，能够实现单轴任意速度规划。用户通过直接输入位置和时间参数描述运动规律。相关函数如表 7.10 所示。

表 7.10 PTT 模式运动相关函数说明

名称	功能	参考
dmc_PttTable	向指定数据表传送数据，采用 PTT 描述方式	8.12 节
dmc_PvtMove	启动 PVT 运动	

例程 7.10：PTT 模式运动

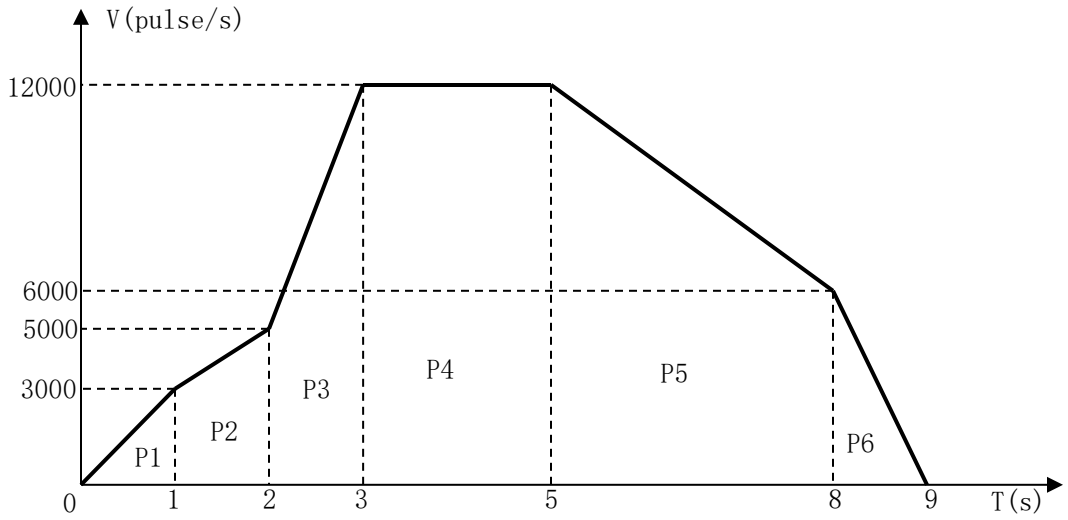


图 7.22 PTT 模式下的 V-T 曲线规划

规划如图 7.22 所示速度曲线，用 PTT 模式规划该速度曲线非常简单。

首先计算各段的位移量，即速度曲线和时间轴所围面积：P1=1500(pulse)，P2=4000(pulse)，P3=8500(pulse)，P4=24000(pulse)，P5=27000(pulse)，P6=3000(pulse)。

将各段位移量累加，得到 PTT 模式下各点的位置和时间数据，如表 7.11 所示。

表 7.11　PTT 模式数组数据

序号	位置 P(pulse)	时间 T(s)
0	0	0
1	1500	1
2	5500	2
3	14000	3
4	38000	5
5	65000	8
6	68000	9

编写程序如下：

```
……
Dim MyCardNo,My_AxisList(1), MyAxisNum As Integer
Dim MyPTime(7) As Double                          ’定义 PTT 的时间数组
Dim MyPPos(7) As Long                             ’定义 PTT 的位置数组
Dim MyCount As Integer

MyCardNo = 0                                      ’卡号为 0
My_AxisList(0) = 0                                ’0 号轴参与 PTT 运动
MyCount = 7                                       ’有 7 组数据

MyPPos(0) = 0: MyPTime(0) = 0                     ’定义 PVT 数组数据
MyPPos(1) = 1500: MyPTime(1) = 1
MyPPos(2) = 5500: MyPTime(2) = 2
MyPPos(3) = 14000: MyPTime(3) = 3
MyPPos(4) = 38000: MyPTime(4) = 5
MyPPos(5) = 65000: MyPTime(5) = 8
MyPPos(6) = 68000: MyPTime(6) = 9

dmc_PttTable MyCardNo, My_AxisList(0), MyCount, MyPTime(0), MyPPos(0)
                                          ’  以 PTT 描述方式，向 0 号轴传送 PVT 数据
MyAxisNum = 1                                     ’参与 PVT 运动的轴数为 1
dmc_PvtMove MyCardNo,MyAxisNum,My_AxisList(0)     ’启动 PVT 运动
……
```

PTT 运动结果(位置-时间曲线、加速度-时间曲线)如图 7.23、7.24 所示。

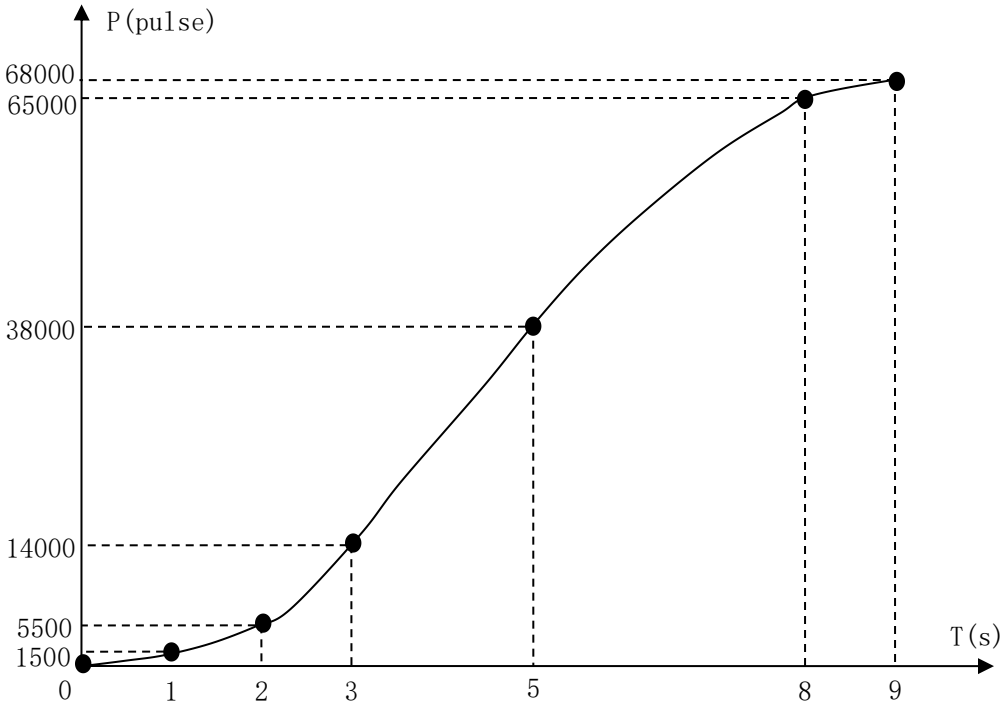


图 7.23　PTT 运动得到的位移曲线

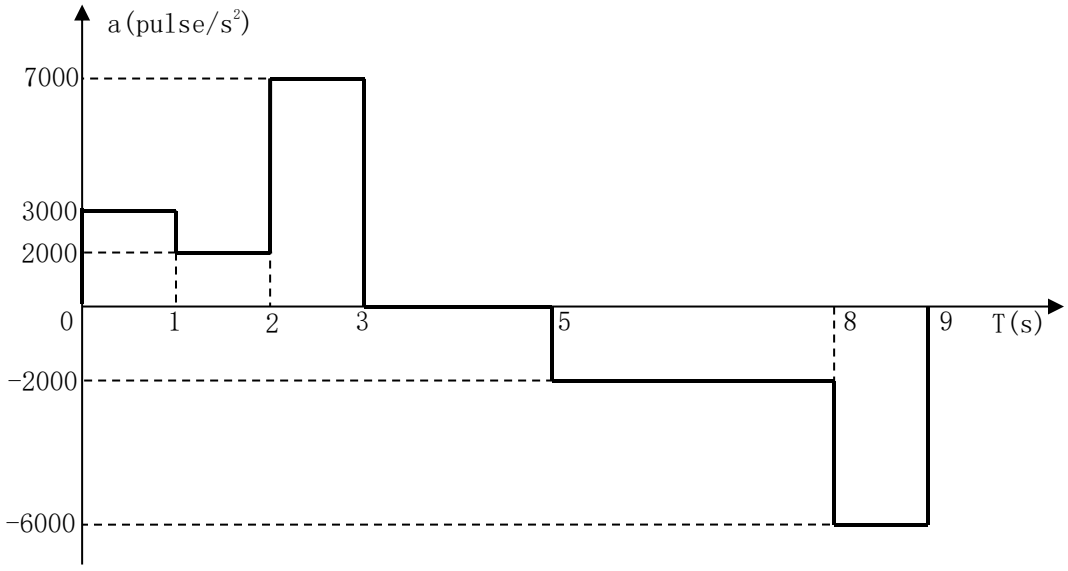


图 7.24　PTT 运动的加速度曲线

7.6.1.2　PTS 运动模式

PTS 运动模式是 PTT 的扩展功能模式，可以使各数据点的速度过渡更加平滑。用户通过输入位置、时间、百分比参数描述运动规律。数据点的百分比参数是指：相邻 2 个数据点之间加速度的变化时间占速度变化时间的百分比。相关函数如表 7.12 所示。

表 7.12　PTS 模式运动相关函数说明

名称	功能	参考
dmc_PtsTable	向指定数据表传送数据，采用 PTS 描述方式	8.12 节
dmc_PvtMove	启动 PVT 运动	

例程 7.11：PTS 模式运动

由图 7.24 可知，例程 7.11 的加速度曲线存在突变，因此，运动过程中有冲击现象。如果要获得更加平滑的速度曲线，可以使用 PTS 模式运动，其速度曲线比图 7.22 所示的速度曲线平滑。

设置各点的速度百分比参数如表 7.13 所示；其对应的加速度-时间曲线如图 7.25 所示，每段的加减速时间为该段时间值×速度百分比；每段的加速度最大值与 PTT 模式下的加速度值不一样，这是因为内部算法作了平滑处理。其对应的位置-时间曲线、速度-时间曲线如图 7.26、7.27 所示。

表 7.13　PTS 模式数组数据

序号	P(pulse)	T(s)	Percent(%)
0	0	0	0
1	1500	1	20
2	5500	2	40
3	14000	3	60
4	38000	5	0
5	65000	8	20
6	68000	9	80

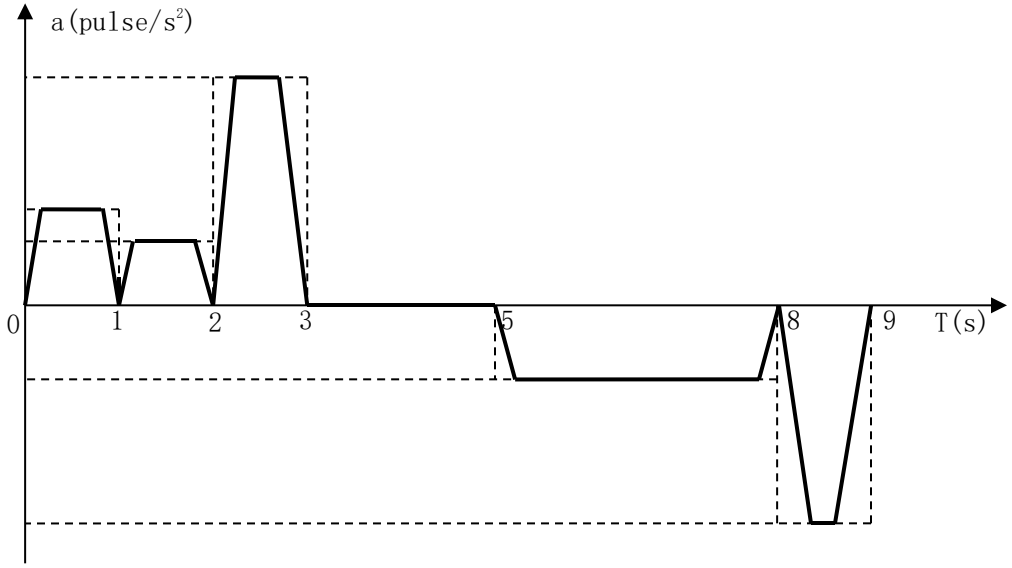


图 7.25　PTS 模式下的加速度曲线

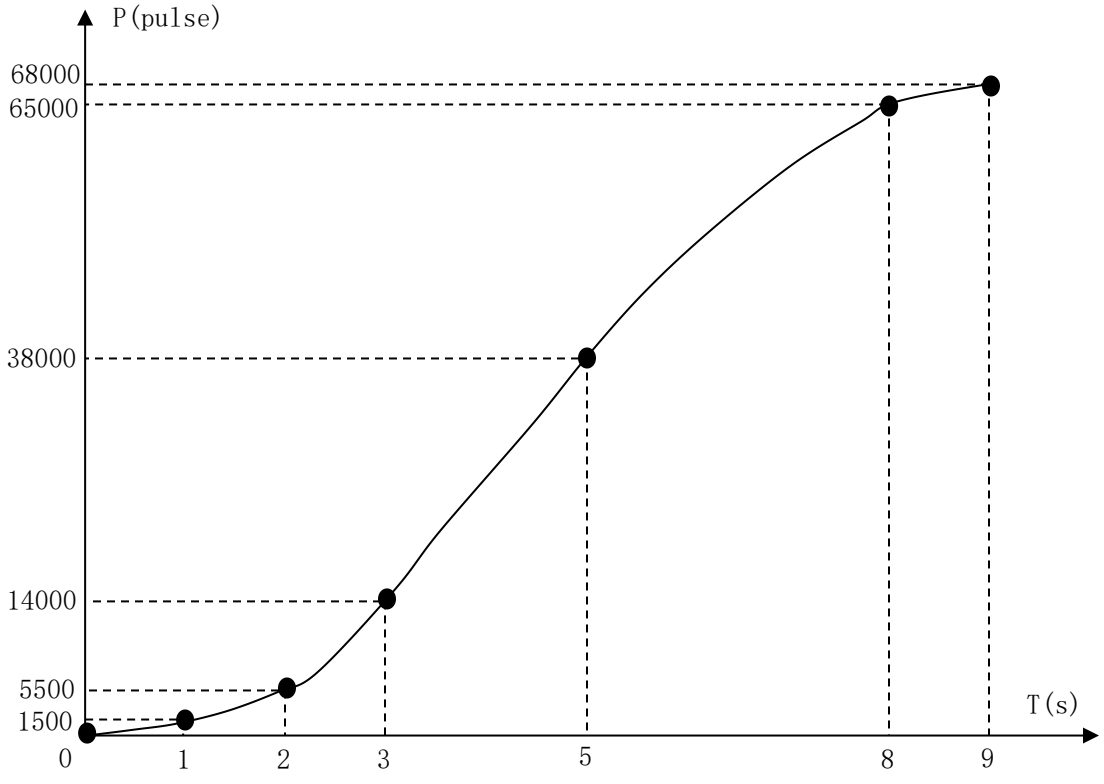


图 7.26　PTS 运动得到的位移曲线

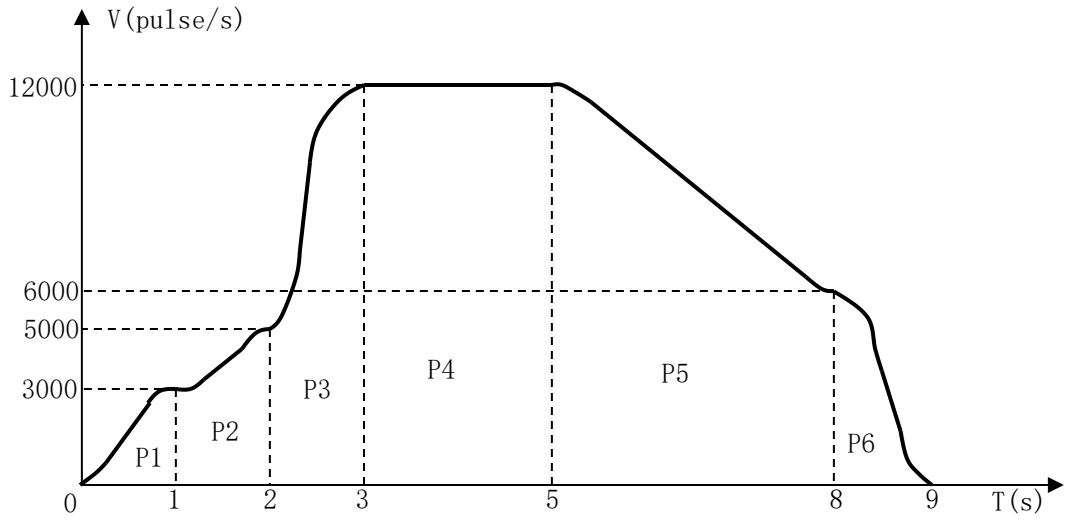


图 7.27　PTS 运动得到的速度曲线

编写程序如下：

```
……
Dim MyCardNo,My_AxisList(1), MyAxisNum As Integer
Dim MyPTime(7) As Double                          '定义 PTS 数据的时间数组
Dim MyPPos(7) As Long                             '定义 PTS 数据的位置数组
Dim MyPPer(7) As Double                           '定义 PTS 数据的百分比数组
Dim MyCount As Integer

MyCardNo = 0                                      ' 卡号为 0
My_AxisList(0) = 0                                ' 0 号轴参与 PTS 运动
MyCount = 7                                       ' 有 7 组数据
                                                  ' 定义 PTS 数据
MyPPos(0) = 0: MyPTime(0) = 0: MyPPer(0) = 0
MyPPos(1) = 1500: MyPTime(1) = 1: MyPPer(1) = 20
MyPPos(2) = 5500: MyPTime(2) = 2: MyPPer(2) = 0
MyPPos(3) = 14000: MyPTime(3) = 3: MyPPer(3) = 60
MyPPos(4) = 38000: MyPTime(4) = 5: MyPPer(4) = 0
MyPPos(5) = 65000: MyPTime(5) = 8: MyPPer(5) = 20
MyPPos(6) = 68000: MyPTime(6) = 9: MyPPer(6) = 80
dmc_PtsTable MyCardNo,My_AxisList(0), MyCount, MyPTime(0), MyPPos(0), MyPPer(0)
                                                  '以 PTS 描述方式向 0 号轴传送 PVT 数据
MyAxisNum = 1                                     ' 参与 PVT 运动的轴数为 1
dmc_PvtMove MyCardNo, MyAxisNum, My_AxisList(0)   '启动 PVT 运动
……
```

7.6.2 多轴高级轨迹规划功能的实现

DMC3000 系列卡具有 PVT 高级运动曲线规划的功能，当用户需要规划一些特殊的运动轨迹而使用单轴运动及插补运动无法满足需求时，可以尝试使用 PVT 来规划自己的运动轨迹。

DMC3000 系列卡共提供了两种 PVT 模式来实现多轴轨迹规划的功能，分别为 PVT、PVTS 运动模式。PVT 模式用于对各点的位置、时间、速度都有要求的轨迹规划，PVTS 模式用于只对各点的位置、时间有要求，而对各点的速度无太多要求的轨迹规划。

多轴高级轨迹规划功能的介绍参见2.1.4.2 节 多轴高级轨迹规划功能。

7.6.2.1 PVT 运动模式

PVT 模式使用一系列数据点的位置、速度、时间参数描述运动规律。

相关函数如表 7.14 所示。

表 7.14 PVT 模式运动相关函数说明

名称	功能	参考
dmc_PvtTable	向指定数据表传送数据，采用 PVT 描述方式	8.12 节
dmc_PvtMove	启动 PVT 运动	

注意：当设置的各点 P、V、T 数据不合理时，很难得到理想的轨迹曲线。

理想轨迹上取点越多，实际轨迹越接近理想轨迹。

例程 7.12：PVT 模式运动

如图 7.28 所示，要控制运动平台按椭圆轨迹运动，但 DMC3000 系列卡中并没有提供椭圆插补函数，用户可以使用 PVT 函数自己设计该轨迹。

设该椭圆的长半轴长 9000pulse，短半轴长 7000pulse；椭圆轨迹的角速度ω恒定，轨迹运动的总时间为 10s。

显然该椭圆的方程为：

$$\begin{cases} x = 9000\cos(\theta) + 9000 \\ y = 7000\sin(\theta) \end{cases} \quad (7.1)$$

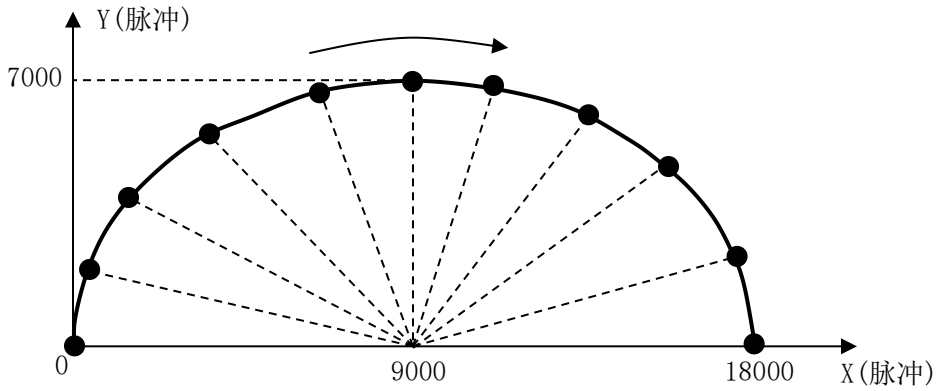


图 7.28　上半椭圆轨迹和分段

使用 PVT 模式运动上半椭圆轨迹的步骤为：

1、将该轨迹分成圆弧角相等的十段轨迹，如图 7.28 所示；计算各点坐标值，即得 P 值。程序如下：

```
Dim MyPPosX(11) As Long      '定义数组，用于存储 PVT 的位置数据（X 轴）
Dim MyPPosY(11) As Long      '定义数组，用于存储 PVT 的位置数据（Y 轴）
Dim a, b, i As Integer
a = 9000: b = 7000           ' 定义椭圆长半轴、短半轴长
For i = 0 To 10              ' 计算各点的 X、Y 坐标
    MyPPosX(i) = a * Cos((10 - i) * 3.14159 / 10) + a
    MyPPosY(i) = b * Sin((10 - i) * 3.14159 / 10)
Next i
```

2、根据各点坐标（即 P 值），计算出各点对应的速度的（V 值）和时间（T 值）。

对椭圆方程（7.1）式求导，可得 X、Y 轴方向的速度分量为：

$$\begin{cases} \dot{x} = -9000\sin(\theta)\dfrac{d\theta}{dt} \\ \dot{y} = 7000\cos(\theta)\dfrac{d\theta}{dt} \end{cases} \qquad (7.1)$$

上式中$\dfrac{d\theta}{dt}$即为角速度ω。

各点的 X、Y 轴方向的速度分量可由以下程序计算得出。程序如下：

```
Dim MyPVelX(11) As Double        ' 定义数组，用于存储 PVT 中的速度数据（X 轴）
Dim MyPTimeX(11) As Double       ' 定义数组，用于存储 PVT 中的时间数据（X 轴）
Dim MyPVelY(11) As Double        ' 定义数组，用于存储 PVT 中的速度数据（Y 轴）
```

```
Dim MyPTimeY(11) As Double          ’定义数组，用于存储 PVT 中的时间数据（Y 轴）
Dim MyWVel As Double                ’ 定义角速度

For i = 0 To 10
   MyPTimeX(i) = i                  ’ 存储 X 轴各点时间数据
   MyPTimeY(i) = i                  ’ 存储 Y 轴各点时间数据
Next i

MyWVel = -3.14159 / 10              ’ 计算角速度ω
MyPVelX(0) = 0: MyPVelX(10) = 0     ’起始点与终止点 X 轴速度设为 0
MyPVelY(0) = 0: MyPVelY(10) = 0     ’起始点与终止点 Y 轴速度设为 0
For i = 0 To 8
    MyPVelX(i + 1) = -a * Sin((10 - i - 1) * 3.14159 / 10) * MyWVel
                                    ’计算其他点 X 轴速度
    MyPVelY(i + 1) = b * Cos((10 - i - 1) * 3.14159 / 10) * MyWVel
                                    ’计算其他点 Y 轴速度
Next i
```

计算出各点 X、Y 轴的 P、V、T 数据如表 7.15 所示。

表 7.15　PVT 数据

	X 轴			Y 轴		
序号	P(pulse)	V(pulse/s)	T(s)	P(pulse)	V(pulse/s)	T(s)
0	0	0	0	0	0	0
1	440	873.731	1	2163	2091.479	1
2	1719	1661.927	2	4115	1779.117	2
3	3710	2287.443	3	5663	1292.603	3
4	6219	2689.048	4	6657	679.560	4
5	9000	2827.431	5	7000	-0.003	5
6	11781	2689.046	6	6657	-679.566	6
7	14290	2287.438	7	5663	-1292.608	7
8	16281	1661.921	8	4114	-1779.120	8
9	17560	873.724	9	2163	-2091.481	9
10	18000	0	10	0	0	10

3、使用 dmc_PvtTable 函数向数据表传递数组数据。

程序如下：

```
Dim MyCardNo,My_AxisList(1) As Integer   ’定义 PVT 运动的卡号、轴列表变量
Dim MyCountX As Integer                  ’定义 X 轴的 PVT 数据点编号变量
Dim MyCountY As Integer                  ’定义 Y 轴的 PVT 数据点编号变量
```

```
MyCardNo = 0                                          ’ 0 号卡
My_AxisList(0) = 0: My_AxisList(1) = 1                ’ 0、1 号轴(即 X、Y 轴)参与 PVT 运动
MyCountX = 11: MyCountY = 11                          ’ 11 组数据
dmc_PvtTable MyCardNo, My_AxisList(0), MyCountX, MyPTimeX(0), MyPPosX(0), MyPVelY(0)
                                                      ’ 以 PVT 描述方式向 X 轴传送 PVT 数据
dmc_PvtTable MyCardNo, My_AxisList(1), MyCountY, MyPTimeY(0), MyPPosY(0), MyPVelY(0)
                                                      ’ 以 PVT 描述方式向 Y 轴传送 PVT 数据
```

4、使用 dmc_PvtMove 函数执行 PVT 运动。

程序如下：

```
Dim My_AxisNum As Integer
My_AxisNum = 2                                        ’参与 PVT 运动的轴数为 2
dmc_PvtMove MyCardNo, My_AxisNum, My_AxisList(0)      ’启动两轴 PVT 运动
```

执行完上述程序后，得到 PVT 运动轨迹如图 7.29 所示。

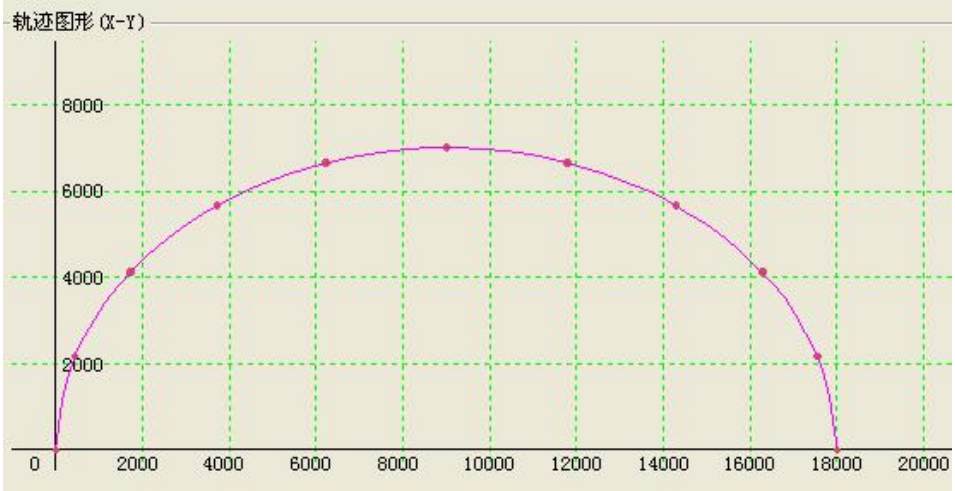


图 7.29　PVT 模式运动得到的上半椭圆轨迹

7.6.2.2　PVTS 运动模式

PVTS 运动模式只需要定义理想轨迹上数据点的位置和时间，以及起点速度和终点速度。运动控制卡根据各数据点的位置、时间参数计算各点之间的轨迹位置和速度，确保各段轨迹的速度连续、加速度连续。相关函数如表 7.16 所示。

表 7.16　PVTS 模式运动相关函数说明

名称	功能	参考
dmc_PvtsTable	向指定数据表传送数据，采用 PVTS 描述方式	8.12 节
dmc_PvtMove	启动 PVT 运动	

例程 7.13：PVTS 模式运动

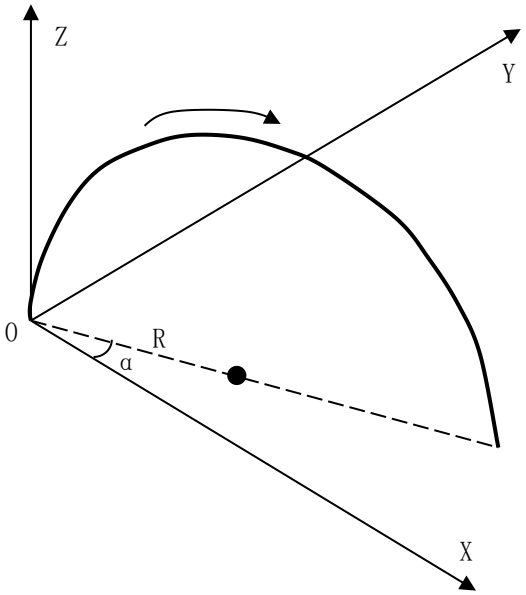


图 7.30　空间圆弧轨迹

如图 7.30 所示，设计空间圆弧轨迹。设其半径 R=15000pulse，其映射在 XY 平面上的轨迹与 X 轴的夹角 α = π /6，轨迹运动总时间为 10s。

显然该空间圆弧的方程为：

$$\begin{cases} x = 15000\cos(\frac{\pi}{6})\cos(\theta) + 15000\cos(\frac{\pi}{6}) \\ y = 15000\sin(\frac{\pi}{6})\cos(\theta) + 15000\sin(\frac{\pi}{6}) \\ z = 15000\sin(\theta) \end{cases} \qquad (7.2)$$

若对轨迹上各点速度无精确要求，采用 PVTS 模式设计该轨迹的步骤如下：

1、将该轨迹分成角度相等的十份，计算各点的位置坐标，即 P 值。

2、根据各点的 P 值，计算出各点的 T 值。设每段轨迹运动的时间相等。

3、设定起点速度与终点速度都为 0。

4、使用 dmc_PvtsTable 函数向数据表传递数组数据。

5、使用 dmc_PvtMove 函数执行 PVT 运动。

编写程序如下：

```
……
Dim MyPTimeX(11),MyPVelBeginX,MyPVelEndX As Double
Dim MyPTimeY(11),MyPVelBeginY,MyPVelEndY As Double
Dim MyPTimeZ(11),MyPVelBeginZ,MyPVelEndZ As Double
Dim MyPPosX(11),MyPPosY(11) ,MyPPosZ(11) As Long
Dim MyCountX,MyCountY,MyCountZ As Integer
Dim MyCardNo,My_AxisNum,My_AxisList(2) As Integer
Dim pi As Single        '定义圆周率
Dim R, i As Integer     '定义空间圆弧半径，循环变量

R = 15000               '定义圆半径
pi = 3.141592654        '定义圆周率
MyCountX = 11: MyCountY = 11: MyCountZ = 11    '设置 X、Y、Z 轴的数据点数

For i = 0 To 10
    MyPPosX(i) = R * Cos(pi / 6) * Cos((10 - i) * pi / 10) + R * Cos(pi / 6)
                                                '计算 X 轴各点位置坐标
    MyPPosY(i) = R * Sin(pi / 6) * Cos((10 - i) * pi / 10) + R * Sin(pi / 6)
                                                '计算 Y 轴各点位置坐标
    MyPPosZ(i) = R * Sin((10 - i) * pi / 10)    '计算 Z 轴各点位置坐标
Next i

For i = 0 To 10
    MyPTimeX(i) = i             '计算 X 轴各点时间
    MyPTimeY(i) = i             '计算 Y 轴各点时间
    MyPTimeZ(i) = i             '计算 Z 轴各点时间
Next i

MyPVelBeginX = 0: MyPVelEndX = 0        'X 轴的起点速度及终点速度为 0
MyPVelBeginY = 0: MyPVelEndY = 0        'Y 轴的起点速度及终点速度为 0
MyPVelBeginZ = 0: MyPVelEndZ = 0        'Z 轴的起点速度及终点速度为 0

MyCardNo = 0                            ' 0 号卡运动
My_AxisList(0) = 0: My_AxisList(1) = 1: My_AxisList(2) = 2
                                        '0、1、2 号轴（即 X、Y、Z 轴）参加 PVT 运动

dmc_PvtsTable MyCardNo, My_AxisList(0), MyCountX, MyPTimeX(0), MyPPosX(0), MyPVelBeginX,
MyPVelEndX                              '以 PVTS 描述方式向 X 轴传送 PVT 数据
dmc_PvtsTable MyCardNo, My_AxisList(1), MyCountY, MyPTimeY(0), MyPPosY(0), MyPVelBeginY,
MyPVelEndY                              '以 PVTS 描述方式向 Y 轴传送 PVT 数据
dmc_PvtsTable MyCardNo, My_AxisList(2), MyCountZ, MyPTimeZ(0), MyPPosZ(0), MyPVelBeginZ,
```

```
MyPVelEndZ                                          '以 PVTS 描述方式向 Z 轴传送 PVT 数据

My_AxisNum = 3                                      '3 个轴参与 PVT 运动
dmc_PvtMove MyCardNo, My_AxisNum, My_AxisList(0)    '启动 PVT 运动
……
```

以 PVTS 模式得到的空间圆弧轨迹如图 7.31 所示。

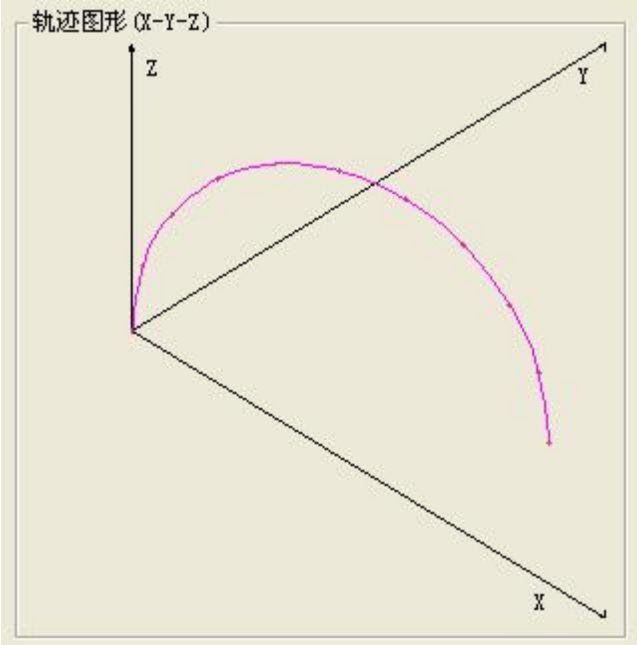


图 7.31　PVTS 模式运动得到的空间圆弧轨迹

7.7　异常减速停止时间设置功能的实现

DMC3000 系列卡支持异常减速停止时间设置功能，用户根据现场实际需求情况设定减速停止时间可达到理想的减速效果，相关函数如表 7.17 所示。

表 7.17　异常减速停止时间设置相关函数说明

名称	功能	参考
dmc_set_dec_stop_time	设置减速停止时间	8.20 节
dmc_get_dec_stop_time	读取减速停止时间设置	

注　意：当发生异常停止时，如：调用 dmc_stop 函数、限位信号（软硬件）被触发、减速停止信号(DSTP)被触发等进行减速停止时，减速停止时间都为 dmc_set_dec_stop_time 函数里设置的减速时间。

例程 7.14：设置异常减速停止时间为 0.5 秒

```
……
Dim CardNo,Axis As Integer
Dim Stop_time As Double
CardNo = 0              ’卡号
Axis = 0                ’指定轴号：第 0 轴
Stop_time = 0.5         ’异常减速停止时间为 0.5 秒
dmc_set_dec_stop_time CardNo, Axis, Stop_time    ’设置轴异常减速停止时间
……
```

7.8 手轮运动功能的实现

7.8.1 单轴手轮运动功能

DMC3000 系列运动控制卡支持单轴手轮运动功能。该功能允许用户设置一个手轮通道对应一个运动轴进行运动。相关函数如表 7.18 所示。

表 7.18 单轴手轮运动功能相关函数说明

名称	功能	参考
dmc_set_handwheel_inmode	设置单轴手轮运动控制输入方式	8.15 节
dmc_handwheel_move	启动手轮运动	
dmc_set_handwheel_channel	手轮通道选择设置	

注 意：1）DMC3C00/3400A 只有低速手轮通道，DMC3800/3600 有高、低速两种手轮通道，手轮外部硬件接口只有一个，通过该功能可以设置这个手轮通道对应的轴号（即每次只能控制一个轴运动）。

2）当启动手轮运动后，只有发送 dmc_stop 或 dmc_emg_stop 命令才会退出手轮模式。

例程 7.15：单轴手轮运动

```
……
Dim MyCardNo,Myaxis,Myinmode As Integer
Dim Mymulti As Long
Dim Myvh As Double

MyCardNo = 0        ’0 号卡
Myaxis = 0          ’设置运动轴为 0 号轴
```

```
Myinmode = 0        ’设置手轮输入方式为 AB 相
Mymulti = 10        ’设置手轮输入倍率为 10
Myvh = 0            ’保留参数
dmc_set_handwheel_inmode MyCardNo,Myaxis,Myinmode,Mymulti, Myvh    ’设置手轮运动
dmc_handwheel_move MyCardNo,Myaxis        ’启动手轮运动
dmc_stop MyCardNo,Myaxis,1                ’立即停止手轮运动
……
```

7.8.2 多轴手轮运动功能

DMC3000 系列运动控制卡支持多轴手轮运动功能。该功能允许用户设置一个手轮通道对应多个运动轴进行运动。相关函数如表 7.19 所示。

表 7.19 多轴手轮运动功能相关函数说明

名称	功能	参考
dmc_set_handwheel_inmode_extern	设置多轴手轮运动控制输入方式	8.15 节
dmc_handwheel_move	启动手轮运动	
dmc_set_handwheel_channel	手轮通道选择设置	

注 意：1）DMC3400A 只有低速手轮通道，DMC3C00/3800/3600 有高、低速两种手轮通道，手轮外部硬件接口只有一个，通过该功能设置可以使一个手轮通道控制多个轴同时运动。

2）当启动手轮运动后，只有发送 dmc_stop 或 dmc_emg_stop 命令才会退出手轮模式。

例程 7.16：多轴手轮运动

```
……
Dim MyCardNo,Myinmode,MyaxisNum,MyaxisList(7),MyIndex As Integer
Dim Mymulti(7) As Long

MyCardNo = 0        ’0 号卡
Myinmode = 0        ’设置手轮输入方式为 AB 相
MyaxisNum = 3       ’设置参与手轮运动的轴数为 3
MyaxisList(0) = 0
MyaxisList(1) = 2
MyaxisList(2) = 5   ’设置运动轴为 0、2、5 号轴
Mymulti(0) = 1
Mymulti(1) = 10
Mymulti(2) = 20     ’设置手轮输入倍率数组分别为 1，10，20
MyIndex = 0         ’设置手轮通道为高速通道

dmc_set_handwheel_channel MyCardNo, MyIndex     ’设置手轮通道为高速通道
```

```
dmc_set_handwheel_inmode_extern MyCardNo,Myinmode,MyaxisNum,MyaxisList(0), Mymulti(0)
        ’设定手轮脉冲方式为 AB 相位信号，0 轴、2 轴、5 轴参与手轮运动，分别为 1、10、20 倍输出脉冲
        控制电机
dmc_handwheel_move MyCardNo, MyaxisList(0)        ’启动手轮运动
dmc_stop MyCardNo, MyaxisList(0),1                ’立即停止手轮运动
……
```

7.9 编码器检测的实现

DMC3000 系列卡的反馈位置计数器是一个 32 位正负计数器，对通过控制卡编码器接口 EA，EB 输入的脉冲（如编码器、光栅尺反馈脉冲等）进行计数。

相关函数如表 7.20 所示。

表 7.20 编码器检测相关函数说明

名称	功能	参考
dmc_set_counter_inmode	设置编码器输入口的计数方式	8.16 节
dmc_get_encoder	读取编码器反馈的脉冲计数值	
dmc_set_encoder	设置编码器的脉冲计数值	

例程 7.17：编码器检测

```
……
Dim MyCardNo,Myaxis,Mymode As Integer
Dim Myencoder_value, MyX_Position As Long
MyCardNo = 0        ’卡号
Myaxis = 0          ’轴号
Mymode = 3          ’设置编码器的计数方式为 4 倍频，AB 相
dmc_set_counter_inmode MyCardNo,Myaxis,Mymode      ’设置 0 号轴的编码器计数方式
Myencoder_value = 0                                ’设置 0 号轴的计数初始值为 0
dmc_set_encoder MyCardNo,Myaxis, Myencoder_value   ’设置 0 号轴的计数初始值
MyX_Position = dmc_get_encoder (MyCardNo,Myaxis)   ’读轴 0 的计数器数值至变量 MyX_Position
……
```

7.10 检测轴到位状态功能的实现

DMC3000 系列卡提供了检测轴到位状态函数，使用这些函数可以设置单轴运动中允许的误差范围，并检测单轴运动是否处于允许的误差范围内。相关函数如表 7.21 所示。

表 7.21　检测轴到位状态功能相关函数说明

名称	功能	参考
dmc_set_factor_error	设置位置误差带	8.23 节
dmc_check_success_pulse	检测指令到位	
dmc_check_success_encoder	检测编码器到位	

注　意：1）该功能检测只适用于单轴运动。

2）检测函数请在 dmc_check_done 检测到轴停止后调用，函数调用后会等待轴到位后返回，如果调用函数 100ms 内未到位，函数超时返回认为不到位。

例程 7.18：检测轴到位状态

```
……
Dim MyCardNo,Myaxis As Integer
Dim MyFactor As Double
Dim MyError As Long

MyCardNo = 0        ’ 0 号卡
Myaxis = 0          ’ 设置运动轴为 0 号轴
MyFactor = 5        ’ 设置编码器系数为 5
MyError = 10        ’ 设置位置误差带为 10 pulse
dmc_set_factor_error MyCardNo, Myaxis, MyFactor, MyError    ’ 设置位置误差带
dmc_pmove MyCardNo,Myaxis,1000,1      ’ 0 号轴定长运动，运动距离为 1000 pulse、绝对模式

While (dmc_check_done(MyCardNo,Myaxis) = 0)    ’ 判断 0 号轴运动状态
    DoEvents
Wend
If dmc_check_success_encoder(MyCardNo,Myaxis) = 0 Then     ’ 检测 0 号轴到位状态
    Print  "编码器不到位!"
Else
    Print  "编码器到位!"
End If
……
```

运行结果说明：

运动的目标位置脉冲数为 1000 pulse，假设当前编码器反馈脉冲数为 199 pulse

由于误差带设为 10 pulse，编码器系数设为 5

处理如下：

199*5=995(pulse)

1000-995=5(pulse)

在误差带范围[-10,10]之内，此时则认为编码器到位，否则认为编码器不到位

7.11 通用 I/O 控制的实现

7.11.1 I/O 普通控制功能

用户可以使用 DMC3000 系列卡上的数字 I/O 口用于检测开关信号、传感器信号等输入信号，或者控制继电器、电磁阀等输出设备的信号。相关函数如表 7.22 所示。

表 7.22 通用 IO 普通控制功能相关函数说明

名称	功能	参考
dmc_read_inbit	读取指定控制卡的某一位输入口的电平状态	8.14 节
dmc_write_outbit	对指定控制卡的某一位输出口置位	
dmc_read_outbit	读取指定控制卡的某一位输出口的电平状态	
dmc_read_inport	读取指定控制卡的全部输入口的电平状态	
dmc_read_outport	读取指定控制卡的全部输出口的电平状态	
dmc_write_outport	设置指定控制卡的全部输出口的电平状态	

注意:在使用 dmc_write_outport 对运动控制卡的全部输出口进行置位，使用 dmc_read_inport、dmc_read_outport 进行 IO 电平读取显示时，应该使用十六进制数进行赋值（尽量避免使用十进制数，特别是在不支持无符号变量的开发环境下）。在对 IO 电平进行控制与读取时，使用十六进制数赋值远比使用十进制数赋值更加直观、方便。

例程 7.19：读取第 0 号卡的通用输入口 1 的电平值，并对通用输出口 3 置高电平

```
……
Dim MyCardNo,MyInbitno,MyInValue,MyOutbitno,MyOutValue As Integer
MyCardNo = 0        ’卡号
MyInbitno = 1       ’定义通用输入口 1
MyInValue = dmc_read_inbit(MyCardNo, MyInbitno)
                    ’读取通用输入口 1 的电平值，并赋值给变量 MyInValue
MyOutbitno = 3      ’定义通用输出口 3
MyOutValue = 1      ’定义输出电平为高
dmc_write_outbit MyCardNo, MyOutbitno, MyOutValue   ’对通用输出口 3 置高电平
……
```

例程 7.20：读取全部输入 IO 口的电平值并进行显示，对全部输出 IO 口的电平进行初始化

```
……
Dim MyCardNo,MyInport,MyOutport As Integer
Dim MyInportValue,MyOutportValue As Long
Dim MyInportValueTemp As String
MyCardNo = 0        ’卡号
```

```
MyInport = 0        ’输入端口组号
MyInportValue = dmc_read_inport(MyCardNo, MyInport)
                    ’读取所有输入 IO 口电平值，并赋值给变量 MyInportValue
MyInportValueTemp = Hex(MyInportValue)      ’转换成十六进制
MyInTextShow = MyInportValueTemp            ’显示在文本框 MyInTextShow 中
MyOutport = 0       ’保留参数，固定为 0
MyOutportValue = &HFFFFFFBFA
      ’&H 表示十六进制（VB），定义输出口电平值，输出口 0、2、10 为低电平，其余端口为高电平
dmc_write_outport  MyCardNo,MyOutport, MyOutportValue    ’对全部输出口进行电平赋值
……
```

7.11.2 I/O 延时翻转功能

DMC3000 系列卡支持 I/O 延时翻转功能。该函数执行后，首先输出一个与当前电平相反的信号，延时设置的时间后，再自动翻转一次电平。相关函数如表 7.23 所示。

表 7.23 IO 延时翻转相关函数说明

名称	功能	参考
dmc_reverse_outbit	IO 输出延时翻转	8.14 节

注意：该功能只适用于通用输出端口 OUT0 ~ OUT15。

例程 7.21：对通用输出 IO 端口 0 进行延时翻转动作

```
……
Dim MyCardNo,MyOutbitno As Integer
Dim MyDelayTime As Long

MyCardNo = 0        ’卡号
MyOutbitno = 0      ’通用输出口 0
MyDelayTime = 0.5 ’延时时间 0.5s
dmc_reverse_outbit MyCardNo, MyOutbitno, MyDelayTime    ’启动 IO 延时翻转
……
```

运行结果：

图 7.32 为该例程对应的输出端口 0 的电平状态时序图(假设输出端口 0 的初始电平状态为低电平)。

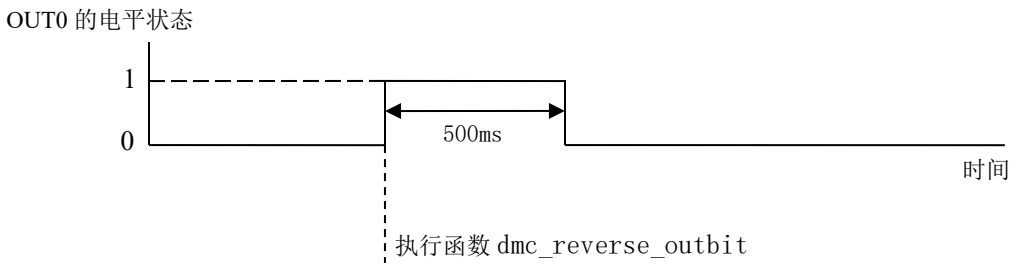


图 7.32　输出端口 0 的电平状态时序图

7.11.3　I/O 计数功能

DMC3000 系列卡支持输入 IO 计数功能，该功能允许用户设置输入 IO 作为计数器使用。相关函数如表 7.24 所示。

表 7.24　IO 计数功能相关函数说明

名称	功能	参考
dmc_set_io_count_mode	设置 IO 计数模式	8.14 节
dmc_set_io_count_value	设置 IO 计数值	
dmc_get_io_count_value	读取 IO 计数值	

例程 7.22：设置通用输入 IO 端口 0 作为计数器

```
……
Dim MyCardNo,MyInbitno,Mymode As Integer
Dim Myfilter,MyCountValue As Long

MyCardNo = 0            ’卡号
MyInbitno = 0           ’通用输入口 0
Mymode = 1              ’上升沿计数
Myfilter = 0.001        ’滤波时间 0.001s
dmc_set_io_count_mode MyCardNo,MyInbitno,Mymode,Myfilter     ’设置计数模式
MyCountValue = 0  ’计数值为 0
dmc_set_io_count_value MyCardNo,MyInbitno,MyCountValue       ’计数器清零
While(1)            ’循环
    dmc_get_io_count_value MyCardNo,MyInbitno,MyCountValue
                    ’读取计数器值，并赋值给变量 MyCountValue
Wend
……
```

7.12 位置比较功能的实现

DMC3000 系列控制卡(DMC3C00 后四轴除外）提供了位置比较功能，位置比较的一般步骤是：

1、配置比较器；

2、清除比较器；

3、添加/更新比较位置点；

4、开始运动并查看比较状态。

7.12.1 一维低速位置比较功能

DMC3000 系列控制卡对每个轴都提供了一组一维低速位置比较，每轴最多都可以添加 256 个比较点。一维低速位置比较的触发延时时间小于 1ms。相关函数如表 7.25 所示。

表 7.25 一维低速位置比较相关函数说明

名称	功能	参考
dmc_compare_set_config	设置一维位置比较器	8.18 节
dmc_compare_clear_points	清除一维位置比较点	
dmc_compare_add_point	添加一维位置比较点	
dmc_compare_get_current_point	读取当前一维比较点位置	
dmc_compare_get_points_runned	查询已经比较过的一维比较点个数	
dmc_compare_get_points_remained	查询可以加入的一维比较点个数	

注 意：（1）每轴的位置比较都是独立进行的。

（2）执行位置比较时，每个比较点的触发是按照添加的比较点顺序执行的，即如果有一个比较点没有被触发比较动作，那么后面的比较点是不会起作用的。

例程 7.23：一维低速位置比较

```
……
Dim MyCardNo,Myaxis,Myenable,Mycmp_source,Mydir,Myaction As Integer
Dim Mypos,Myactpara As Long
MyCardNo = 0        ’卡号
Myaxis = 0          ’轴号
Myenable = 1        ’设置比较器使能
Mycmp_source = 0    ’设置比较源为指令位置
dmc_compare_set_config MyCardNo,Myaxis,Myenable,Mycmp_source     ’设置 0 号轴比较器
dmc_compare_clear_points MyCardNo,Myaxis        ’清除 0 号轴的比较点及比较点个数
Mypos = 10000       ’设置比较位置为 10000pulse
```

```
Mydir = 1          ’设置比较模式为大于等于
Myaction = 3       ’设置触发功能为 IO 电平取反
Myactpara = 0      ’设置输出 IO 端口 0 触发功能
dmc_set_position MyCardNo,Myaxis,0          ’设置 0 号轴的指令脉冲计数器绝对位置为 0
dmc_compare_add_point MyCardNo,Myaxis, Mypos, Mydir, Myaction, Myactpara
            ’添加比较点，位置 10000pulse，模式大于等于，触发时动作为输出端口 0 电平取反
Mypos = 30000      ’设置比较位置为 30000pulse
Mydir = 1          ’设置比较模式为大于等于
Myaction = 1       ’设置触发功能为 IO 端口置高电平
Myactpara = 3      ’设置输出 IO 端口 3 触发功能
dmc_compare_add_point MyCardNo,Myaxis, Mypos, Mydir, Myaction, Myactpara
            ’添加比较点，位置 30000pulse，模式大于等于，触发时动作为输出端口 3 置高电平
dmc_set_profile MyCardNo,Myaxis,2000,10000,0.1,0.1,3000    ’设置梯形速度曲线参数
dmc_pmove MyCardNo,Myaxis,50000,0     ’0 号轴定长运动，运动距离为 50000pulse、相对模式
……
```

运行结果：

当运动到 10000pulse 时，通用输出口 0 电平将翻转；当运动到 30000pulse 时，通用输出口 3 输出高电平

7.12.2 二维低速位置比较功能

DMC3000 系列卡提供了一组二维低速位置比较，最多都可以添加 256 个比较点。二维低速位置比较的触发延时时间小于 1ms。相关函数如表 7.26 所示。

表 7.26 二维低速位置比较相关函数说明

名称	功能	参考
dmc_compare_set_config_extern	设置二维位置比较器	8.18 节
dmc_compare_clear_points_extern	清除二维位置比较点	
dmc_compare_add_point_extern	添加二维位置比较点	
dmc_compare_get_current_point_extern	读取当前二维位置比较点位置	
dmc_compare_get_points_runned_extern	查询已经比较过的二维比较点个数	
dmc_compare_get_points_remained_extern	查询可以加入的二维比较点个数	

注意：执行位置比较时，每个比较点的触发是按照添加的比较点顺序执行的，即如果有一个比较点没有被触发比较动作，那么后面的比较点是不会起作用的。

例程 7.24：二维低速位置比较

```
……
Dim MyCardNo,My_axis(1) As Integer      ’定义进行位置比较的卡号，轴号列表
Dim My_ComPos(1) as Long                ’定义进行位置比较的位置列表
Dim My_ComDir(1) as Integer             ’定义进行位置比较的方向列表
Dim Pos(2), Cen(2) As Long              ’定义圆弧插补的终点位置及圆心坐标列表

MyCardNo = 0       ’卡号
dmc_compare_set_config_extern  MyCardNo,1,0
                              ’设置比较器，0 号卡二维位置比较使能，比较源指令位置
dmc_compare_clear_points_extern MyCardNo ’清除位置比较点
My_axis(0)=0：My_axis(1)=1                ’设置比较轴号列表
My_ComPos(0)=-5000: My_ComPos(1)=5000     ’设置比较位置列表
My_ComDir(0)=0: My_ComDir(1)=1            ’设置比较模式列表
dmc_set_position MyCardNo, My_axis(0),0
dmc_set_position MyCardNo, My_axis(1),0    ’设置轴的指令脉冲计数器绝对位置为 0
dmc_compare_add_point_extern MyCardNo, My_axis(0), My_ComPos(0), My_ComDir(0),3,0
                              ’添加比较点，触发时动作为输出端口 0 电平取反
My_ComPos(0)=5000: My_ComPos(1)=5000      ’设置比较位置列表
My_ComDir(0)=1: My_ComDir(1)=0            ’设置比较模式列表
dmc_compare_add_point_extern MyCardNo, My_axis(0), My_ComPos(0), My_ComDir(0),1,3
                              ’添加比较点，触发时动作为输出端口 3 置为高电平
Pos(0) = 0: Pos(1) = 0        ’设置终点坐标
Cen(0) = 0: Cen(1) = 5000     ’设置圆心坐标
dmc_set_vector_profile_multicoor MyCardNo,0,0,3000,0.1,0,0          ’设置矢量速度曲线
dmc_arc_move_multicoor MyCardNo,0, My_axis(0), Pos(0), Cen(0),0,0 ’圆弧插补运动
……
```

7.12.3 一维高速位置比较功能

DMC3000 系列卡共有四个高速位置比较器，每个高速位置比较器均配有 1 个硬件位置比较输出接口。其中，CMP0、CMP1 端口电路使用的是普通光耦（速率 4KHz），CMP2、CMP3 端口电路使用的是高速光耦（1MHz）。

每个 CMP 输出端口都可以与任意轴关联，支持多种比较模式，用户只需在函数里设置便可以使用，非常灵活方便。

其中“等于”、“小于”、“大于”比较模式为单次比较模式。“队列”模式提供 127 个比较点，采用先添加先比较，比较完可追加比较点，也可一次性添加多个比较点。“线性”模式适用于每个比较点位置都是按照线性增量依次增加的情况，用户只需要设置起始比较点位置以及线性增量值

即可，非常方便。位置间时间间隔最小可达几微秒。

相关函数如表 7.27 所示。

表 7.27 一维高速位置比较相关函数说明

名称	功能	参考
dmc_hcmp_set_mode	设置高速比较模式	8.19 节
dmc_hcmp_set_config	配置高速比较器	
dmc_hcmp_set_liner	设置高速比较线性模式参数	
dmc_hcmp_clear_points	清除高速位置比较点	
dmc_hcmp_add_point	添加/更新高速比较位置	
dmc_hcmp_get_current_state	读取高速比较参数	
dmc_write_cmp_pin	控制指定 CMP 端口的输出	
dmc_read_cmp_pin	读取指定 CMP 端口的电平	

注意：（1）每个比较器的位置比较都是独立进行的。

（2）在队列及线性比较模式中，执行位置比较时，每个比较点的触发是按照添加的比较点顺序执行的，即如果有一个比较点没有被触发比较动作，那么后面的比较点是不会起作用的。

例程 7.25：单次比较模式（大于）

```
……
Dim MyCardNo,Myhcmp,Myaxis,MyCmpSource,MyCmpLogic,MyCmpMode As Integer
Dim MyTime, MyCmpPos As Long

MyCardNo = 0        ’卡号
Myhcmp = 0          ’高速比较器，对应硬件 CMP 端口
Myaxis = 2          ’轴号
MyCmpSource=0       ’比较位置源
MyCmpLogic=0        ’有效电平
MyTime=0            ’脉冲宽度,只有比较模式为 4 或 5 时起作用
dmc_hcmp_set_config MyCardNo,Myhcmp,Myaxis,MyCmpSource,MyCmpLogic,MyTime
                ’配置高速比较器 0 关联 2 号轴，比较源为指令位置，有效电平为低电平
MyCmpMode =3        ’比较模式
dmc_hcmp_set_mode MyCardNo,Myhcmp, MyCmpMode      ’设置比较器 0 比较模式为模式 3（大于）
dmc_hcmp_clear_points MyCardNo,Myhcmp      ’清除位置比较器
MyCmpPos = 50000    ’比较位置
dmc_hcmp_add_point MyCardNo, Myhcmp, MyCmpPos      ’更新高速比较位置为 50000 pulse
dmc_set_profile MyCardNo,Myaxis,2000,10000,0.1,0.1,3000    ’设置梯形速度曲线参数
dmc_pmove MyCardNo,Myaxis,100000,1      ’2 号轴定长运动，运动距离为 100000 pulse、绝对模式
……
```

运行结果：

当 2 号轴运动超过位置 50000 pulse 时，CMP0 端口输出低电平并保持。

例程 7.26：队列比较模式

```
……
Dim MyCardNo,Myhcmp,Myaxis,MyCmpSource,MyCmpLogic,MyCmpMode As Integer
Dim MyTime, MyCmpPos As Long

MyCardNo = 0        ’卡号
Myhcmp = 0          ’高速比较器，对应硬件 CMP 端口
Myaxis = 0          ’轴号
MyCmpSource=0       ’比较位置源
MyCmpLogic=0        ’有效电平
MyTime=500000       ’脉冲宽度,单位 us
dmc_hcmp_set_config MyCardNo,Myhcmp,Myaxis,MyCmpSource,MyCmpLogic,MyTime
     ’配置高速比较器 0 关联 0 号轴，比较源为指令位置，有效电平为低电平，脉冲宽度为 500ms
MyCmpMode =4        ’比较模式
dmc_hcmp_set_mode MyCardNo,Myhcmp, MyCmpMode      ’设置比较器 0 比较模式为模式 4（队列）
dmc_hcmp_clear_points MyCardNo,Myhcmp             ’清除位置比较器
MyCmpPos = 10000    ’比较位置
dmc_hcmp_add_point MyCardNo, Myhcmp, MyCmpPos     ’更新高速比较位置为 10000 pulse
MyCmpPos = 30000    ’比较位置
dmc_hcmp_add_point MyCardNo, Myhcmp, MyCmpPos     ’添加高速比较位置为 30000 pulse
MyCmpPos = 70000    ’比较位置
dmc_hcmp_add_point MyCardNo, Myhcmp, MyCmpPos     ’添加高速比较位置为 70000 pulse
dmc_set_profile MyCardNo,Myaxis,2000,10000,0.1,0.1,3000   ’设置梯形速度曲线参数
dmc_pmove MyCardNo,Myaxis,100000,1      ’0 号轴定长运动，运动距离为 100000 pulse、绝对模式
……
```

运行结果：

当 0 号轴运动经过位置 10000、30000、70000 pulse 时，CMP0 端口输出低电平，低电平持续时间为 500ms。

例程 7.27：线性比较模式

```
……
Dim MyCardNo,Myhcmp,Myaxis,MyCmpSource,MyCmpLogic,MyCmpMode As Integer
Dim MyTime, MyCmpPos,MyCmpInc,MyCmpCount As Long

MyCardNo = 0        ’卡号
Myhcmp = 1          ’高速比较器，对应硬件 CMP 端口
Myaxis = 0          ’轴号
MyCmpSource=0       ’比较位置源
```

```
MyCmpLogic=0        ’有效电平
MyTime=500000       ’脉冲宽度,单位 us
dmc_hcmp_set_config MyCardNo,Myhcmp,Myaxis,MyCmpSource,MyCmpLogic,MyTime
     ’配置高速比较器 1 关联 0 号轴，比较源为指令位置，有效电平为低电平，脉冲宽度为 500ms
MyCmpMode =5        ’比较模式
dmc_hcmp_set_mode MyCardNo,Myhcmp, MyCmpMode      ’设置比较器 1 比较模式为模式 5（线性）
dmc_hcmp_clear_points MyCardNo,Myhcmp      ’清除位置比较器
MyCmpInc = 20000    ’线性增量，单位 pulse
MyCmpCount = 5      ’比较次数
dmc_hcmp_set_liner MyCardNo,Myhcmp,MyCmpInc, MyCmpCount
     ’设置比较器 1 线性模式参数，线性增量为 20000 pulse，比较次数为 5
MyCmpPos = 10000    ’比较位置
dmc_hcmp_add_point MyCardNo, Myhcmp, MyCmpPos      ’更新高速比较初始位置为 10000 pulse
dmc_set_profile MyCardNo,Myaxis,2000,10000,0.1,0.1,3000    ’设置梯形速度曲线参数
dmc_pmove MyCardNo,Myaxis,100000,1      ’0 号轴定长运动，运动距离为 100000 pulse、绝对模式
……
```

运行结果：

当 0 号轴运动经过位置 10000、30000、50000、70000、90000 pulse 时，CMP1 端口输出低电平，低电平持续时间为 500ms。

7.13 高速位置锁存功能的实现

7.13.1 高速单次位置锁存功能

DMC3000 系列卡(DMC3C00 后四轴除外）提供了高速单次位置锁存功能，位置锁存无触发延时时间，当捕获到位置锁存信号后立即锁存当前位置。相关函数如表 7.28 所示。

表 7.28 高速单次锁存相关函数说明

名称	功能	参考
dmc_set_ltc_mode	设置指定轴的 LTC 信号	8.17 节
dmc_set_latch_mode	设置锁存方式	
dmc_get_latch_value	从控制卡内读取编码器锁存器的值	
dmc_get_latch_flag	从控制卡内读取指定卡内锁存器的标志位	
dmc_reset_latch_flag	复位指定卡的锁存器的标志位	

注意：在单次锁存中，多次触发高速锁存口只锁存第一次触发位置，只有调用函数清除锁存状态方可再次锁存。

例程 7.28：高速单次位置锁存

```
……
Dim MyCardNo,Myaxis,Myltc_logic,Myltc_mode As Integer
Dim Myall_enable,Mylatch_source,Mytriger_chunnel As Integer
Dim Myfilter As Double
Dim My_latch_Value As Long          ’定义锁存值
MyCardNo = 0                        ’卡号
Myaxis = 0                          ’轴号
Myltc_logic = 0                     ’设置 LTC 触发方式为下降沿触发
Myltc_mode = 0                      ’保留参数
Myfilter = 0                        ’保留参数
dmc_set_ltc_mode MyCardNo,Myaxis,Myltc_logic,Myltc_mode,Myfilter
                                    ’设置 0 号轴的 LTC 信号，触发方式为下降沿触发
Myall_enable = 0                    ’设置锁存方式为单次锁存
Mylatch_source = 0                  ’设置锁存源为指令位置
Mytriger_chunnel = 0                ’保留参数
dmc_set_latch_mode MyCardNo,Myaxis, Myall_enable,Mylatch_source,Mytriger_chunnel
                                        ’设置 0 号轴的锁存源是指令位置，单次锁存
dmc_reset_latch_flag MyCardNo,Myaxis  ’复位 0 号轴的锁存状态
dmc_set_profile MyCardNo,Myaxis,1000,5000,0.1,0.1,2000    ’设置梯形速度曲线参数
dmc_pmove MyCardNo,Myaxis,50000,0     ’0 号轴定长运动，运动距离为 50000pulse、相对模式
While (dmc_get_latch_flag(MyCardNo,Myaxis) = 0)    ’判断 0 号轴 LTC 锁存状态
    DoEvents
Wend
My_latch_Value = dmc_get_latch_value(MyCardNo,Myaxis)
                  ’从控制卡内读取编码器锁存器的值，并赋值给变量 My_latch_Value
……
```

7.13.2 高速连续位置锁存功能

DMC3000 系列卡提供了高速连续位置锁存功能，位置锁存无触发延时时间，当捕获到位置锁存信号后立即锁存当前位置；但相邻两个锁存点之间的间隔时间应大于 1ms。

相关函数如表 7.29 所示。

表 7.29 高速连续锁存相关函数说明

名称	功能	参考
dmc_set_ltc_mode	设置指定轴的 LTC 信号	8.17 节
dmc_set_latch_mode	设置锁存方式	
dmc_get_latch_flag_extern	从 PC 缓存中读取锁存器已锁存个数	

dmc_get_latch_value_extern	按索引号读取 PC 缓冲区中已保存的锁存值	
dmc_reset_latch_flag	复位指定卡的锁存器的标志位	

注意：在连续锁存，多次触发高速锁存口锁存所有的触发位置，超过 1000 次剔除最先的触发位置保存最新的触发位置。在每次锁存后，不需要调用函数清除锁存状态。但是在启动高速连续位置锁存之前，必须调用函数清除锁存状态。

例程 7.29：高速连续位置锁存

```
……
Dim MyCardNo,Myaxis As Integer
Dim My_latch_Value(1000) as Long          ’定义锁存值
Dim My_latch_flag as Integer              ’定义锁存个数
Dim i as Integer
MyCardNo = 0       ’卡号
Myaxis = 0         ’轴号
dmc_set_ltc_mode MyCardNo,Myaxis,0,0,0     ’设置 0 号卡 0 号轴的 LTC 信号，下降沿触发
dmc_set_latch_mode MyCardNo,Myaxis,2,0,0   ’设置 0 号轴的锁存源是指令位置，连续锁存
dmc_reset_latch_flag MyCardNo,Myaxis       ’复位 0 号轴的锁存状态
dmc_set_profile MyCardNo,Myaxis,1000,5000,0.1,0.1,2000    ’设置梯形速度曲线参数
dmc_pmove MyCardNo,Myaxis,50000,0          ’0 号轴定长运动，距离为 50000pulse、相对模式
While(dmc_check_done(MyCardNo,Myaxis) = 0)    ’判断 0 号轴状态
    DoEvents
Wend
My_latch_flag = dmc_get_latch_flag_extern(MyCardNo,Myaxis)
      ’从 PC 缓存中读取锁存器已锁存个数，并赋值给变量 My_latch_flag
For i = 0 To My_latch_flag - 1
    My_latch_Value(i) = dmc_get_latch_value_extern(MyCardNo,Myaxis,i)
      ’按索引号读取 PC 缓冲区中已保存的锁存值，并赋值给变量 My_latch_Value
Next i
……
```

7.13.3 LTC 触发延时急停功能

DMC3000 系列运动控制卡提供了 LTC 触发延时急停功能，位置锁存无触发延时时间，当捕获到位置锁存信号后立即锁存当前位置；并经过设置的延时时间后停止关联轴的运动。

相关函数如表 7.30 所示。

表 7.30 LTC 触发延时急停相关函数说明

名称	功能	参考

dmc_set_ltc_mode	设置指定轴的 LTC 信号	8.17 节
dmc_set_latch_mode	设置锁存方式	
dmc_set_latch_stop_time	设置 LTC 端口触发延时急停时间	
dmc_reset_latch_flag	复位指定卡的锁存器的标志位	

注意：1）在每次触发锁存前，必须调用函数清除锁存状态。

2）触发延时急停模式只对 0 号轴及 4 号轴起作用；LTC0 对应为 0 号轴，LTC1 对应为 4 号轴。

例程 7.30：LTC 触发延时急停

```
……
Dim MyCardNo,Myaxis,Myltc_logic,Myltc_mode As Integer
Dim Myall_enable,Mylatch_source,Mytriger_chunnel As Integer
Dim Myfilter As Double
Dim My_latch_Value,MyLtcDTime As Long        ’定义锁存值
MyCardNo = 0                          ’卡号
Myaxis = 0                            ’轴号
Myltc_logic = 0                       ’设置 LTC 触发方式为下降沿触发
Myltc_mode = 0                        ’保留参数
Myfilter = 0                          ’保留参数
dmc_set_ltc_mode MyCardNo,Myaxis,Myltc_logic,Myltc_mode,Myfilter
                                      ’设置 0 号轴的 LTC 信号，触发方式为下降沿触发
Myall_enable = 3                      ’设置锁存方式为触发延时停止
Mylatch_source = 0                    ’设置锁存源为指令位置
Mytriger_chunnel = 0                  ’保留参数
dmc_set_latch_mode MyCardNo,Myaxis, Myall_enable,Mylatch_source,Mytriger_chunnel
                                    ’设置 0 号轴的锁存源是指令位置，锁存方式为触发延时停止
MyLtcDTime = 10000                 ’延时急停时间，单位 us
dmc_set_latch_stop_time MyCardNo,Myaxis, MyLtcDTime
                                   ’设置 LTC0 端口触发延时急停时间为 10 ms
dmc_reset_latch_flag MyCardNo,Myaxis  ’复位 0 号轴的锁存状态
dmc_set_profile MyCardNo,Myaxis,1000,5000,0.1,0.1,2000    ’设置梯形速度曲线参数
dmc_pmove MyCardNo,Myaxis,50000,0      ’0 号轴定长运动，运动距离为 50000pulse、相对模式
While (dmc_get_latch_flag(MyCardNo,Myaxis) = 0)    ’判断 0 号轴 LTC 锁存状态
    DoEvents
Wend
My_latch_Value = dmc_get_latch_value(MyCardNo,Myaxis)
                  ’从控制卡内读取编码器锁存器的值，并赋值给变量 My_latch_Value
While (dmc_check_done(MyCardNo,Myaxis) = 0)    ’判断 0 号轴运动状态
    DoEvents
Wend
Print “Axis 0 is stop!”
```

……

7.13.4 LTC 反相输出功能

DMC3000 系列卡提供了 LTC 反相输出功能，该功能允许设置某一输出端口为 LTC 电平的反相输出。相关函数如表 7.31 所示。

表 7.31 LTC 反相输出相关函数说明

名称	功能	参考
dmc_SetLtcOutMode	LTC 反相输出设置	8.17 节
dmc_GetLtcOutMode	读取 LTC 反相输出设置	

注意：当某输出端口作为 LTC 反相输出后，该端口将不能通过通用 IO 函数进行操作。

例程 7.31：LTC 反相输出

```
……
Dim MyCardNo,Myaxis,MyEnable,MyOutBitno As Integer
MyCardNo = 0                    ’卡号
Myaxis = 0                      ’轴号
MyEnable = 1                    ’LTC 反相输出功能使能状态
MyOutBitno = 2                  ’关联输出端口号
dmc_SetLtcOutMode MyCardNo,Myaxis, MyEnable, MyOutBitno
   ’设置输出端口 2 为 LTC0 端口的反相输出
   ’当设置此函数后，就无法使用 dmc_write_outbit 函数控制 OUT2，但仍可以使用读取输出 IO 函数
   ’读取该端口的状态；设置此函数后 OUT2 的电平状态与 LTC0 的电平状态是相反的
```

7.14 轴 IO 映射功能的实现

DMC3000 系列卡(DMC3C00 后四轴除外）提供了轴 IO 映射功能，该功能允许用户对专用 IO 信号的硬件输入接口进行任意配置，相关函数如表 7.32 所示。

表 7.32 轴 IO 映射相关函数说明

名称	功能	参考
dmc_set_axis_io_map	设置轴 IO 映射关系	8.21 节
dmc_get_axis_io_map	读取轴 IO 映射关系设置	

例程 7.32：设置第 0 号卡的第 2 轴原点接口作为第 0 轴的正限位信号

```
……
Dim CardNo,Axis,IoType, MapIoType, MapIoIndex As Integer
Dim Filter As Double
CardNo = 0          ’卡号
Axis = 0            ’指定轴号：第 0 轴
IoType = 0          ’指定轴的 IO 信号类型为：正限位信号
MapIoType = 2       ’轴 IO 映射类型：原点信号
MapIoIndex = 2      ’  轴 IO 映射索引号：第 2 轴
Filter = 0.01       ’0.01 秒滤波
dmc_set_AxisIoMap CardNo, Axis, IoType, MapIoType, MapIoIndex, Filter ’设置轴 IO 映射
……
```

例程 7.33：设置第 0 号卡的通用输入口 0 接口作为所有轴的急停信号，并且设置急停信号为低电平有效

```
……
Dim CardNo,Axis,IoType, MapIoType, MapIoIndex As Integer
Dim Filter As Double
CardNo = 0        ’卡号
IoType = 3        ’指定轴的 IO 信号类型为：急停信号
MapIoType = 6     ’轴 IO 映射类型：通用输入端口
MapIoIndex = 0    ’轴 IO 映射索引号：通用输入 0
Filter = 0.01     ’0.01 秒滤波
For Axis = 0 To 7   ’循环，依次对 0~7 号轴进行设置（DMC3600 时为 0 To 5，因其只有 6 个轴）
    dmc_set_AxisIoMap CardNo, Axis, IoType, MapIoType, MapIoIndex, Filter ’设置轴 IO 映射
Next Axis
For Axis = 0 To 7   ’循环，依次对 0~7 号轴进行设置（DMC3600 时为 0 To 5，因其只有 6 个轴）
  dmc_set_emg_mode CardNo, Axis, 1, 0   ’设置 EMG 信号使能，低电平有效
Next Axis
……
```

7.15 原点锁存功能的实现

DMC3000 系列卡(DMC3C00 后四轴除外）提供了原点锁存功能，该功能可以实现在碰到原点信号时将当前位置锁存，使用该功能可以实现精确回零运动，一般步骤如下：

1） 使用 dmc_set_homelatch_mode 函数设置原点锁存模式；

2） 使用 dmc_reset_homelatch_flag 函数清除原点锁存标志；

3） 设置轴运动参数，并启动运动；

4） 判断原点锁存标志是否有效，有效则停止运动；

5） 重新设置轴运动速度为低速度，并使用定长运动到原点锁存位置；

6） 回到原点锁存位置后，指令脉冲计数器清零。

相关函数如表 7.33 所示。

表 7.33 原点锁存相关函数说明

名称	功能	参考
dmc_set_homelatch_mode	设置原点锁存模式	8.4 节
dmc_reset_homelatch_flag	清除原点锁存标志	
dmc_get_homelatch_flag	读取原点锁存标志	
dmc_get_homelatch_value	读取原点锁存值	

例程 7.34：使用原点锁存功能进行精确回零运动

```
……
Dim MyCardNo,MyAxis,Myenable,Mylogic,Mysource As Integer
Dim MyHomelatchValue As Long
MyCardNo = 0      ’卡号
MyAxis = 0        ’轴号
Myenable = 1      ’使能原点锁存
Mylogic = 0       ’锁存方式为下降沿锁存
Mysource = 0      ’锁存位置源为指令位置
dmc_set_homelatch_mode MyCardNo,MyAxis,Myenable,Mylogic,Mysource   ’设置原点锁存
dmc_reset_homelatch_flag MyCardNo,MyAxis    ’清除原点锁存标志位
dmc_set_profile MyCardNo,MyAxis,1000,10000,0.1,0.1,1000
                                    ’设置 0 号轴运动参数，最大速度为 10000pulse/s
dmc_vmove MyCardNo,MyAxis,0         ’0 号轴执行连续运动，负方向
While(dmc_get_homelatch_flag(MyCardNo,Myaxis) = 0)    ’判断原点锁存标志位
    DoEvents
Wend
dmc_stop MyCardNo,MyAxis,0          ’减速停止
While (dmc_check_done(MyCardNo,MyAxis) = 0)    ’判断当前运动状态
    DoEvents
Wend
dmc_set_profile MyCardNo,MyAxis,500,1000,0.1,0.1,500
                                    ’设置 0 号轴运动参数，最大速度为 1000pulse/s
MyHomelatchValue = dmc_get_homelatch_value(MyCardNo,Myaxis)
                                   ’获取原点锁存值，并赋值给变量 MyHomelatchValue
dmc_pmove MyCardNo,MyAxis, MyHomelatchValue,1   ’0 号轴定长运动到原点锁存位置，绝对模式
While (dmc_check_done(MyCardNo,MyAxis) = 0)     ’判断当前运动状态
```

```
    DoEvents
Wend
dmc_set_position MyCardNo,MyAxis,0                    ’设置 0 号轴的指令脉冲计数器绝对位置为 0
……
```

7.16 虚拟 IO 映射功能的实现

DMC3000 系列卡(DMC3C00 后四轴除外）提供了虚拟 IO 映射功能，该功能允许用户对虚拟 IO 口的硬件输入接口进行任意配置。通过该功能函数的设置可以实现专用通用 IO 输入接口的滤波功能，并且可以通过函数 dmc_read_inbit_virtual 读取该端口滤波后的电平状态。相关函数如表 7.34 所示。

表 7.34 虚拟 IO 映射相关函数说明

名称	功能	参考
dmc_set_io_map_virtual	设置虚拟 IO 映射关系	8.22 节
dmc_get_io_map_virtual	读取虚拟 IO 映射关系设置	
dmc_read_inbit_virtual	读取滤波后的虚拟 IO 口电平状态	

注 意：1）dmc_read_inbit_virtual 函数需要配合虚拟 IO 映射功能使用。

2）dmc_read_inbit_virtual 与 dmc_read_inbit 函数的区别是：dmc_read_inbit 函数是不经过滤波直接读取硬件端口的电平状态，dmc_read_inbit_virtual 函数通过虚拟 IO 映射后读取相应端口滤波后的电平状态。

例程 7.35：设置第 0 号卡的第 2 轴原点接口作为虚拟 IO 口 3 的硬件输入接口，同时设置该接口的滤波时间为 0.05 秒，并读取该接口滤波后的电平状态

```
……
Dim CardNo,bitno, MapIoType, MapIoIndex, MyVirIOLogic As Integer
Dim Filter As Double
CardNo = 0              ’卡号
bitno = 3               ’虚拟 IO 口号：3
MapIoType = 2           ’虚拟 IO 映射类型：原点信号
MapIoIndex = 2          ’  虚拟 IO 映射索引号：第 2 轴
Filter = 0.05           ’0.05 秒滤波
dmc_set_io_map_virtual CardNo,bitno, MapIoType, MapIoIndex, Filter ’设置虚拟 IO 映射
MyVirIOLogic = dmc_read_inbit_virtual(CardNo,bitno)
      ’读取第 2 轴原点接口（即虚拟 IO 口 3）滤波后的电平，并赋值给变量 MyVirIOLogic
……
```

7.17 CAN-IO 扩展模块的操作

CAN-IO 扩展模块是 DMC3000 系列卡的配套产品，扩展 DMC3000 系列卡的 IO 端口，每个模块都可扩展 16 输入 16 输出 IO 端口。通过菊花链的连接方式可以将多个 CAN-IO 扩展模块挂在同一块运动控制卡下面，最多支持连接 8 个 CAN-IO 扩展模块。

CAN-IO 通讯的波特率为 1Mbps，通讯状态的刷新频率为 4ms。

DMC3000 系列卡提供了 CAN-IO 扩展函数，用户通过这些函数的调用可以很方便地实现对扩展 IO 的操作。一般步骤如下：

1）使用 nmc_set_connect_state 函数进行 CAN 通讯连接；

2）使用 nmc_get_connect_state 函数读取 CAN 通讯状态，如果出现连接异常，则重新执行步骤 1；

3）连接成功后，则可以对 CAN-IO 扩展模块的输入输出端口进行操作。

相关函数如表 7.35 所示。

表 7.35 CAN-IO 相关函数说明

名称	功能	参考
nmc_set_connect_state	设置 CAN 通讯状态	8.24 节
nmc_get_connect_state	读取 CAN 通讯状态	
nmc_write_outbit	设置指定 CAN-IO 扩展模块的某个输出口的电平	
nmc_read_outbit	读取指定 CAN-IO 扩展模块的某个输出口的电平	
nmc_read_inbit	读取指定 CAN-IO 扩展模块的某个输入口的电平	
nmc_write_outport	设置指定 CAN-IO 扩展模块的输出端口的电平	
nmc_read_outport	读取指定 CAN-IO 扩展模块的输出端口的电平	
nmc_read_inport	读取指定 CAN-IO 扩展模块的输入端口的电平	
nmc_set_da_mode	设置指定 CAN-ADDA 扩展模块的某个输出口的模式	
nmc_get_da_mode	读取指定 CAN-ADDA 扩展模块的某个输出口的模式	
nmc_set_ad_mode	设置指定 CAN-ADDA 扩展模块的某个输入口的模式	
nmc_get_ad_mode	读取指定 CAN-ADDA 扩展模块的某个输入口的模式	
nmc_set_da_output	设置指定 CAN-ADDA 扩展模块的某个输出口的电压/电流	
nmc_get_da_output	读取指定 CAN-ADDA 扩展模块的某个输出口的电压/电流	
nmc_get_ad_input	读取指定 CAN-ADDA 扩展模块的某个输入端口的电压/电流	

例程 7.36：假设 DMC3000 系列卡扩展了 1 个 CAN-IO 扩展模块，读取该模块的通用输入口 1 的电

平值，并对该模块的通用输出口 3 置低电平

```
……
ushort CardNo, CANCount, CANStatus, GetCANCount, GetCANStatus, CANBaud ;
ushort CANNum, MyInBitno, MyOutBitno, MyOutLevel;
ushort MyInValue;

CardNo = 0;                          //卡号
CANCount = 1;                        //节点数为 1，因只有 1 个扩展 CAN-IO 模块
CANStatus = 1;                       //通讯状态为连接
CANBaud =0;
GetCANCount = 0;
GetCANStatus = 0;
LTDMC. nmc_set_connect_state (CardNo, CANCount, CANStatus,CANBaud);
                                     //连接 CAN-IO 扩展模块
LTDMC. nmc_get_connect_state (CardNo, ref  GetCANCount, ref  GetCANStatus);
                                     //获取当前 CAN 通讯状态
CANNum = 1;                          //节点号为 1，选择菊花链上的第 1 个扩展 CAN-IO 模块
MyInBitno = 1;                       //通用输入端口 1
LTDMC.nmc_read_inbit(CardNo, CANNum, MyInBitno，ref MyInValue);
                  //获取节点号为 1 的扩展模块上的输入端口 1 的电平，并将该值赋于变量 MyInValue
MyOutBitno = 3;  //通用输出端口 3
MyOutLevel = 0;  //低电平
LTDMC.nmc_write_outbit(CardNo, CANNum, MyOutBitno, MyOutLevel);
                 //控制节点号为 1 的扩展模块上的输出端口 3 输出低电平
……
```

例程 7.37：假设 DMC3000 系列卡扩展了 2 个 CAN-IO 扩展模块，读取节点号 1 扩展模块的通用输入口 0 的电平值，并对节点号 2 扩展模块的通用输出口 7 置低电平

```
……
ushort CardNo, CANCount, CANStatus, GetCANCount, GetCANStatus,CANBaud;
ushort CANNum, MyInBitno,MyOutBitno, MyOutLevel;
ushort  MyInValue;

CardNo = 0;                          //卡号
CANCount = 2;                        //节点数为 2，因有 2 个扩展 CAN-IO 模块
GetCANCount = 0;
GetCANStatus = 0;
CANBaud =0;
CANStatus = 1;                       //通讯状态为连接
LTDMC. nmc_set_connect_state (CardNo, CANCount, CANStatus,CANBaud );
```

```
                                    //连接 CAN-IO 扩展模块
LTDMC. nmc_get_connect_state (CardNo, ref  GetCANCount, ref  GetCANStatus);
                                    //获取当前 CAN 通讯状态
CANNum = 1;                         //节点号为 1，选择菊花链上的第 1 个扩展 CAN-IO 模块
MyInBitno = 0;                      //通用输入端口 0
LTDMC.nmc_read_inbit(CardNo, CANNum, MyInBitno，ref MyInValue);
                //获取节点号为 1 的扩展模块上的输入端口 0 的电平，并将该值赋于变量 MyInValue
CANNum = 2;     //节点号为 2，选择菊花链上的第 2 个扩展 CAN-IO 模块
MyOutBitno = 7;//通用输出端口 7
MyOutLevel = 0;//低电平
LTDMC.nmc_write_outbit(CardNo, CANNum, MyOutBitno, MyOutLevel);
                //控制节点号为 2 的扩展模块上的输出端口 7 输出低电平
……
```

例程 7.38：假设 DMC3800 卡扩展了 2 个 CAN-IO 扩展模块，读取 1 号模块的通用输入口 1 的电平值，并对 2 号模块的通用输出口 23 置低电平

```
……
short err=0;
ushort CardNo, CANCount, CANStatus, GetCANCount, GetCANStatus,CANBaud ;
ushort CANNum;
ushort MyInBitno, MyOutBitno, MyOutLevel, MyInValue;

CardNo = 0;                         //卡号
CANCount = 2;                       //节点数为 2，因有 2 个扩展 CAN-IO 模块
CANStatus = 0;                      //通讯状态为连接
CANBaud = 0;                        //波特率 1000Kbps
GetCANCount = 0;
GetCANStatus = 0;

err = LTDMC.nmc_set_connect_state(CardNo, CANCount, CANStatus, CANBaud);
                                    //连接 CAN-IO 扩展模块
err = LTDMC.nmc_get_connect_state(CardNo, ref  GetCANCount, ref  GetCANStatus);
                                    //获取当前 CAN 通讯状态
CANNum = 1;                         //节点号为 1，选择菊花链上的 1 号扩展 CAN-IO 模块
MyInBitno = 1;                      //通用输入端口 1
MyInValue = 0;
err = LTDMC.nmc_read_inbit(CardNo, CANNum, MyInBitno, ref MyInValue);
                //获取节点号为 1 的扩展模块上的输入端口 1 的电平，并将该值赋变量 MyInValue
CANNum = 2;                         //节点号为 2，选择菊花链上的 2 号扩展 CAN-IO 模块
MyOutBitno = 23;  //通用输出端口 23
MyOutLevel = 0;  //低电平
```

```
err = LTDMC.nmc_write_outbit(CardNo, CANNum, MyOutBitno, MyOutLevel);
                //控制节点号为 2 的扩展模块上的输出端口 23 输出低电平
……
```

例程 7.39：假设 DMC3800 卡扩展了 2 个 CAN-ADDA 扩展模块，模块的输入输出均为电压模式，读取节点号 1 扩展模块的 AD 输入口 1 的电平值，并对节点号 2 扩展模块的 DA 输出口 1 置 2V 电平

```
……
short err = 0;
ushort CardNo, CANCount, CANStatus, GetCANCount, GetCANStatus, CANBaud;
ushort CANNum;
ushort MyADno, MyDAno;
Int32 MyDAValue, MyADValue;

CardNo = 0;                         //卡号
CANCount = 2;                       //节点数为 2，因有 2 个扩展 CAN 模块
GetCANCount = 0;
GetCANStatus = 0;
CANBaud =0;                         //波特率 1000Kbps
CANStatus = 1;                      //通讯状态为连接
LTDMC.nmc_set_connect_state(CardNo, CANCount, CANStatus,CANBaud );
                                    //连接 CAN 扩展模块
LTDMC.nmc_get_connect_state(CardNo, ref  GetCANCount, ref  GetCANStatus);
                                    //获取当前 CAN 通讯状态
CANNum = 1;                       //节点号为 1，选择菊花链 i 上的 1 号扩展 CAN 模块
MyADno = 1;                     //AD 输入端口 1
MyADValue = 0;
err = LTDMC.nmc_get_ad_output(CardNo, CANNum, MyADno, ref MyADValue);
          //获取节点号为 1 的扩展模块上的输入端口 1 的电平，并将该值赋于变量 MyInValue
CANNum = 2;                       //节点号为 2，选择菊花链上的 2 号扩展 CAN 模块
MyDAno = 1;  //DA 输出端口 1
MyDAValue = 2000;  //2V 输出
err = LTDMC.nmc_set_da_output(CardNo, CANNum, MyDAno, MyDAValue);
          //控制节点号为 2 的扩展模块上的输出端口 1 输出 2V
```

7.18 光盘中的 VB 例程

DMC3000 系列运动控制卡为了方便用户利用 VB 或 VC 等编程工件开发应用软件，针对典型功能，如：单轴点位运动、回原点、插补运动、通用输入输出等，提供了例源代码。用户可以直接将配套 CD 中相应目录中的代码直接拷贝到您的程序工程中使用。

7.18.1 定长运动例程

图 7.33 定长运动例程主界面

定长运动例程主界面如图 7.33 所示。此例程为四个轴的定长运动（即点位运动），每个轴都可设置起始速度、运行速度、停止速度、加速时间、减速时间参数。设有“启动”、“位置清零”、“在线变速”、“在线变位”、“减速停止”、“立即停止”、“退出”功能按钮。这个例程用到了控制卡的初始化/关闭函数，读取各运动轴当前速度/位置函数，设置轴运动速度函数/定长运动函数，轴的运动状态函数，在线变位/在线变速函数，减速停函数/急停函数。通过这个例程的学习可以了解我司控制卡基本函数的调用，详细代码请参考光盘例程“例 1_定长运动”。

7.18.2 连续运动例程

连续运动例程主界面如图 7.34 所示。此例程为任意指定轴的连续运动例程。每个轴都可设置起始速度、运行速度、停止速度、加速时间、减速时间参数。设有“启动”、“位置清零”、“在线变速”、“减速停止”、“立即停止”、“退出”功能按钮。程序底层可显示每运动轴的实时速度和每

个轴当前的运动距离。详细代码请参考光盘例程“例 2_连续运动”。

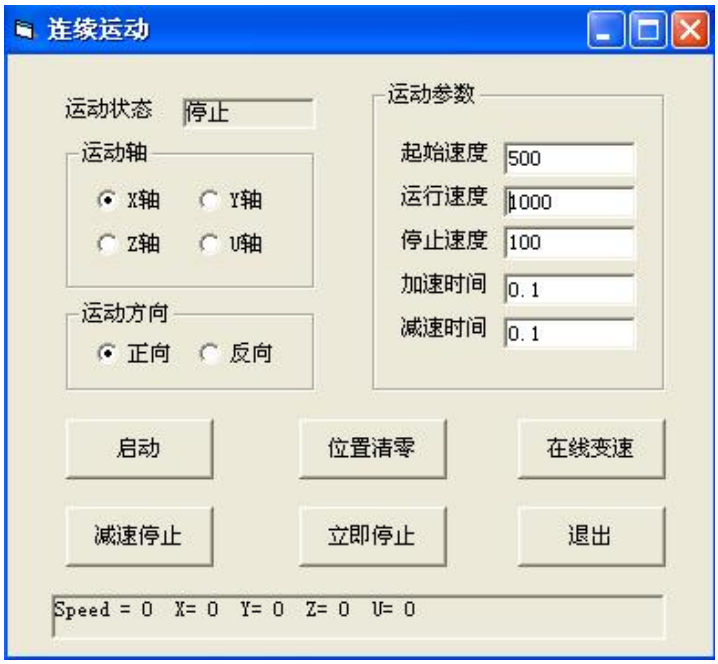


图 7.34　连续运动例程主界面

7.18.3　回原点运动例程

此例程为四个轴的回原点程序，为每个轴设置了回零运动速度，加减速时间。设有“启动”，“减速停止”，“急停停止”，“位置清零”和“退出”功能按钮。设有回零方式、回零方向、回零速度模式选择。在程序界面底层可以显示每个轴运动的实时速度及当前位置，并可以实时检测轴的运动状态。详细代码请参考光盘例程“例 3_回原点运动”。回原点运动例程主界面如图 7.35 所示。

图 7.35 回原点运动例程主界面

7.18.4 通用专用输入输出例程

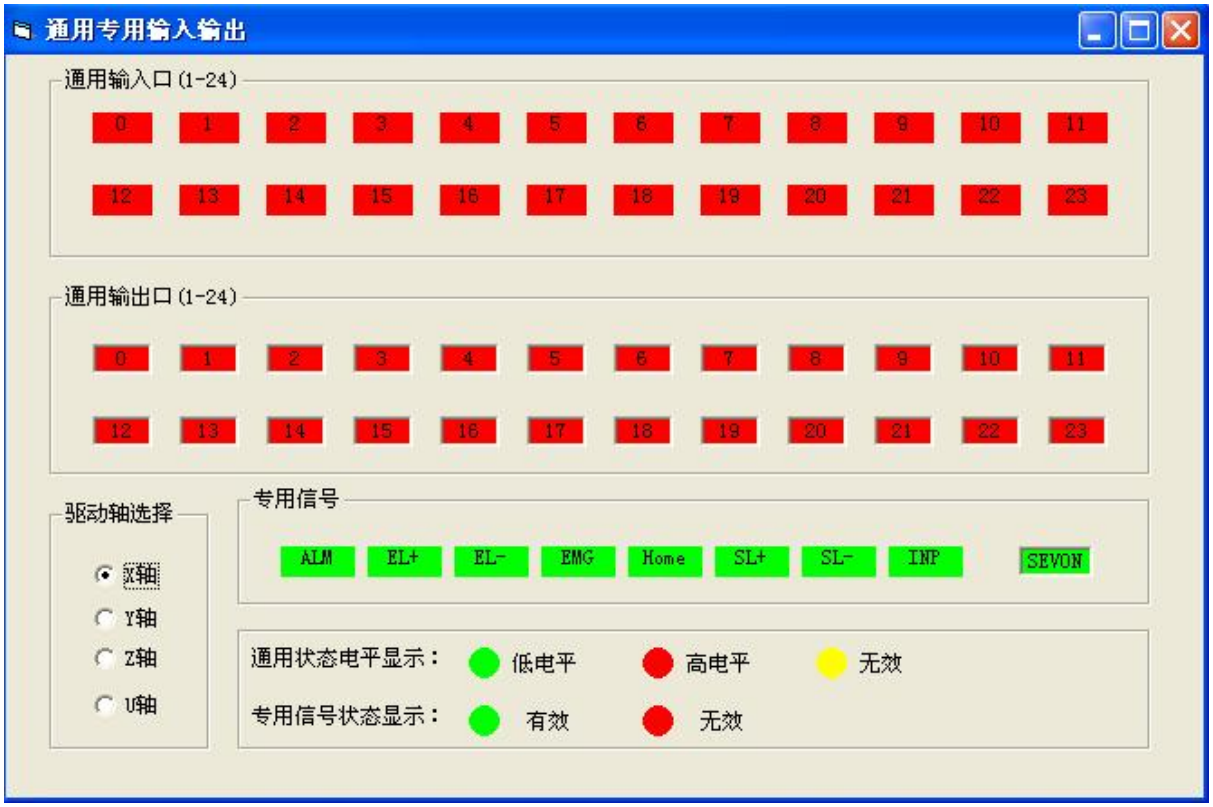


图 7.36 通用专用输入输出例程主界面

主界面如图 7.36 所示，此例程为控制卡通用数字 I/O 信号和每个轴的专用 I/O 信号的检测例程。程序提供所有通用输入/输出信号的检测图标，可以对每个输出口的输出状态进行操作，并可以检测每个轴的专用信号的输入状态。详细代码请参考光盘例程“例 4_通用专用输入输出”。

7.18.5 直线插补例程

直线插补运动例程主界面如图 7.37 所示。此例程为任意两轴至四轴的直线插补运动例程。程序为每段插补线段设有运行速度，加/减速度，每个轴设有插补运动距离，提供两到四轴插补轴的任意方式选择。设有“执行”，“减速停止”，“急停”，“位置清零”和“退出”按钮等功能，程序底层可显示插补运动时每个轴的实时插补速度。并且可以在线检测轴的运动状态。详细代码请参考光盘例程“例 5_直线插补”。

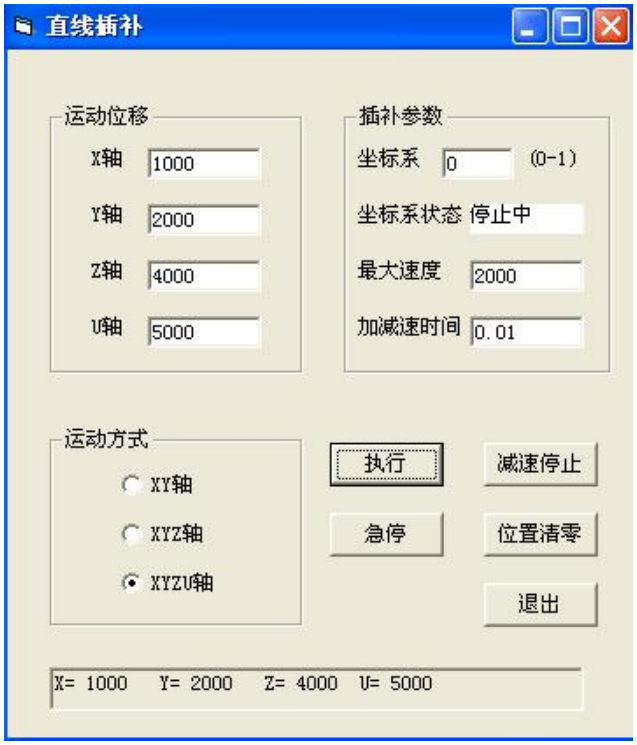


图 7.37 直线插补例程主界面

7.18.6 两轴圆弧插补例程

此例程为第 0 轴和第 1 轴两轴的圆弧插补运动例程。其它任意两轴的圆弧插补与此例程的代码类似。圆弧插补的轴号，矢量运动速度和插补距离在代码里面已设置好。程序设有“启动”、“急停”、“立即停止”、“位置清零”、“退出”按钮等功能。详细代码请参考光盘例程“例 6_两轴圆弧插补”。两轴圆弧插补例程主界面如图 7.38 所示。

图 7.38　两轴圆弧插补例程主界面

7.18.7　手轮运动例程

图 7.39 所示为手轮运动例程。此例程为控制卡使用单轴手轮运动控制输入方式例程。可设置手轮输入方式、手轮倍率，设有“启动”、“停止”、“退出”功能按钮。详细代码请参考光盘例程“例 7_手轮运动”。

图 7.39　手轮运动例程主界面

7.18.8 单轴点位运动实例

图 7.40 单轴点位运动界面

如图 7.40 所示，此例程为单轴点位运动例程。控制电机运动的方式分为模拟按钮及外部 IO 输入控制两种方式；选择模拟按钮时，即通过软件界面按钮控制电机正反转；选择外部 IO 输入时，即通过外部硬件 IO 输入来控制电机正反转。在输入参数框里，可以设定电机轴号、S 形曲线参数、外部 IO 输入的端口号及每次运行的距离。在显示框里，可以监控此时电机的速度、位置及电机状态。此外，还添加了两个定时器控件，定时器 1 主要负责读取当前速度、位置、电机状态并显示在显示框中。定时器 2 主要负责选择外部 IO 输入时，监测外部 IO 输入的状态，当读取的电平为有效电平时控制电机转动。详细代码请参考光盘例程“例 8_单轴点位运动”。

如果选用 0 号轴（即硬件接口为第 0 轴电机信号接口）控制步进驱动器，正转信号通过外部输入 IO 端口 1 输入，反转信号通过外部输入 IO 端口 2 输入，正转状态通过外部输出 IO 端口 1 输出，反转状态通过外部输出 IO 端口 2 输出。其硬件接线图如图 7.41 所示。

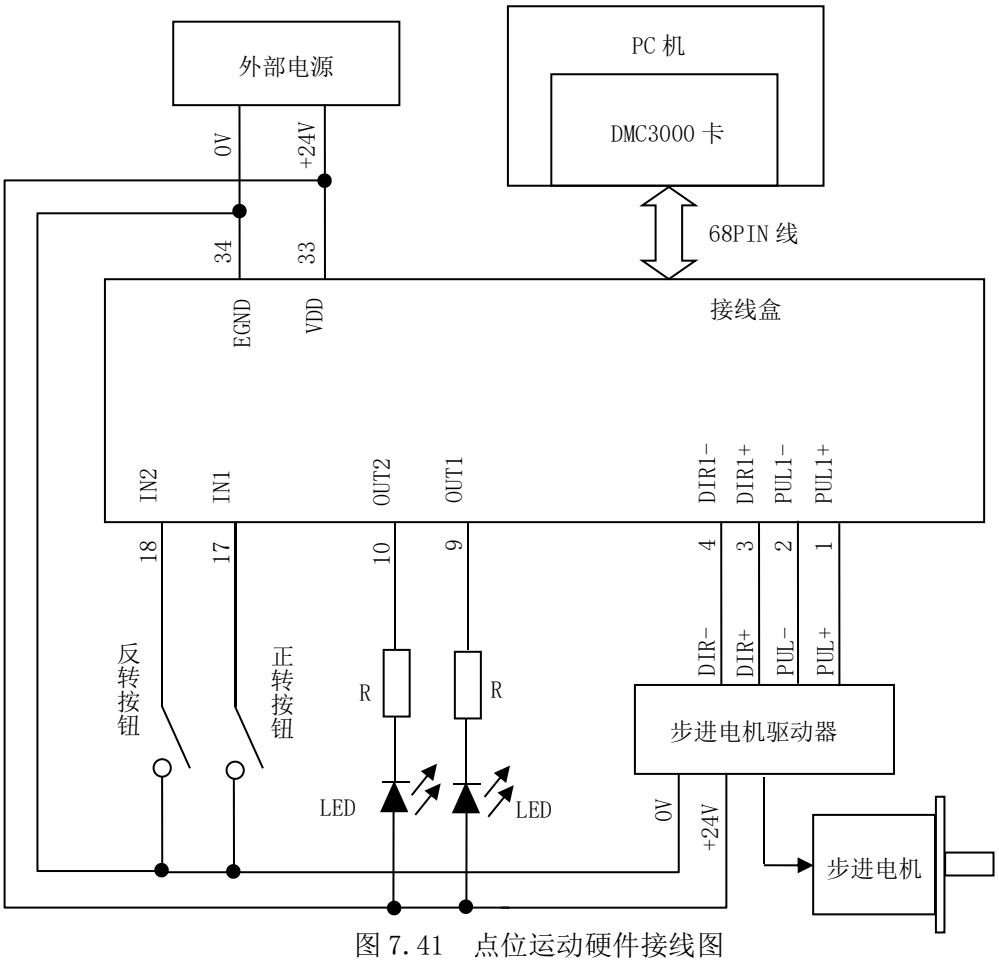


图 7.41　点位运动硬件接线图

7.18.9　轨迹运动实例

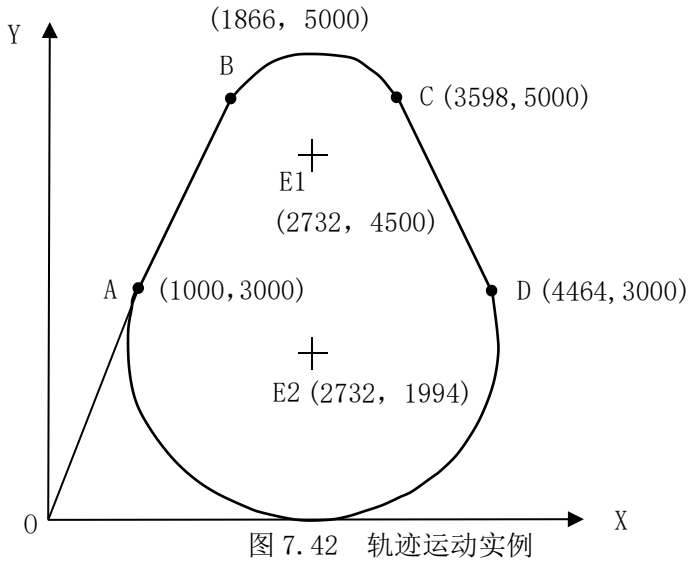


图 7.42　轨迹运动实例

运动轨迹 ABCD 如图 7.42 所示，其中 AB、CD 为直线运动；BC、DA 为圆弧运动。设当前位置位于系统原点 O 处，则先执行直线插补运动 OA，然后执行直线插补运动 AB，然后执行圆弧插补运动 BC，然后执行直线插补运动 CD，最后执行圆弧插补运动 DA。即运动步骤为 O->A->B->C->D->A。

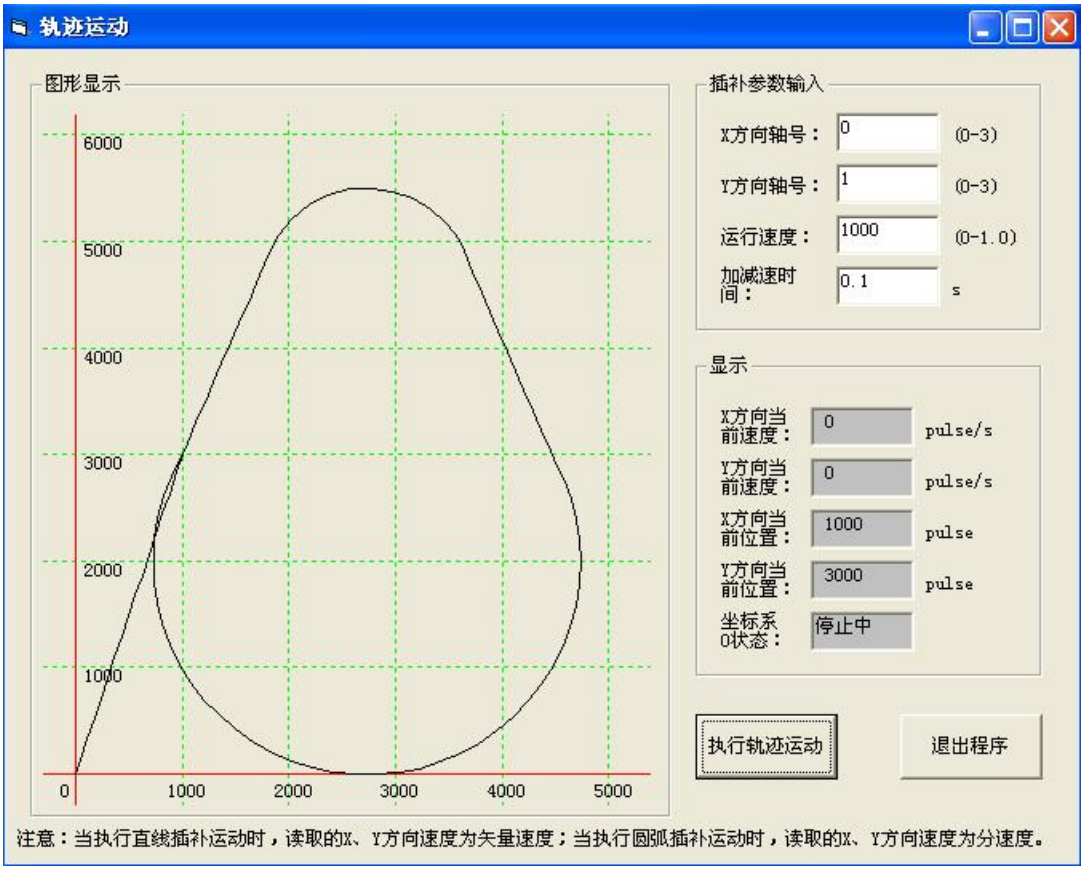


图 7.43　轨迹运动实例程序界面

程序界面如图 7.43 所示，在插补参数输入框可以输入 X、Y 方向的电机轴号，插补运动参数。显示框中可以监控当前速度、位置、电机状态信息。图形显示框中则实时显示运动图形。此外，还添加一个定时器控件，其主要负责读取当前速度、位置、电机状态并显示在显示框中；图形显示框的曲线绘制。详细代码请参考光盘例程“例 9_轨迹运动”。

7.18.10　单轴任意速度规划 PVT 实例

单轴任意速度规划实例程序界面如图 7.44 所示。PTT 模式运动的数据计算、函数调用参见例程 7.11；PTS 模式运动的数据计算、函数调用参见例程 7.12。详细代码请参考光盘例程“例 10_单轴任意速度规划 PVT”。

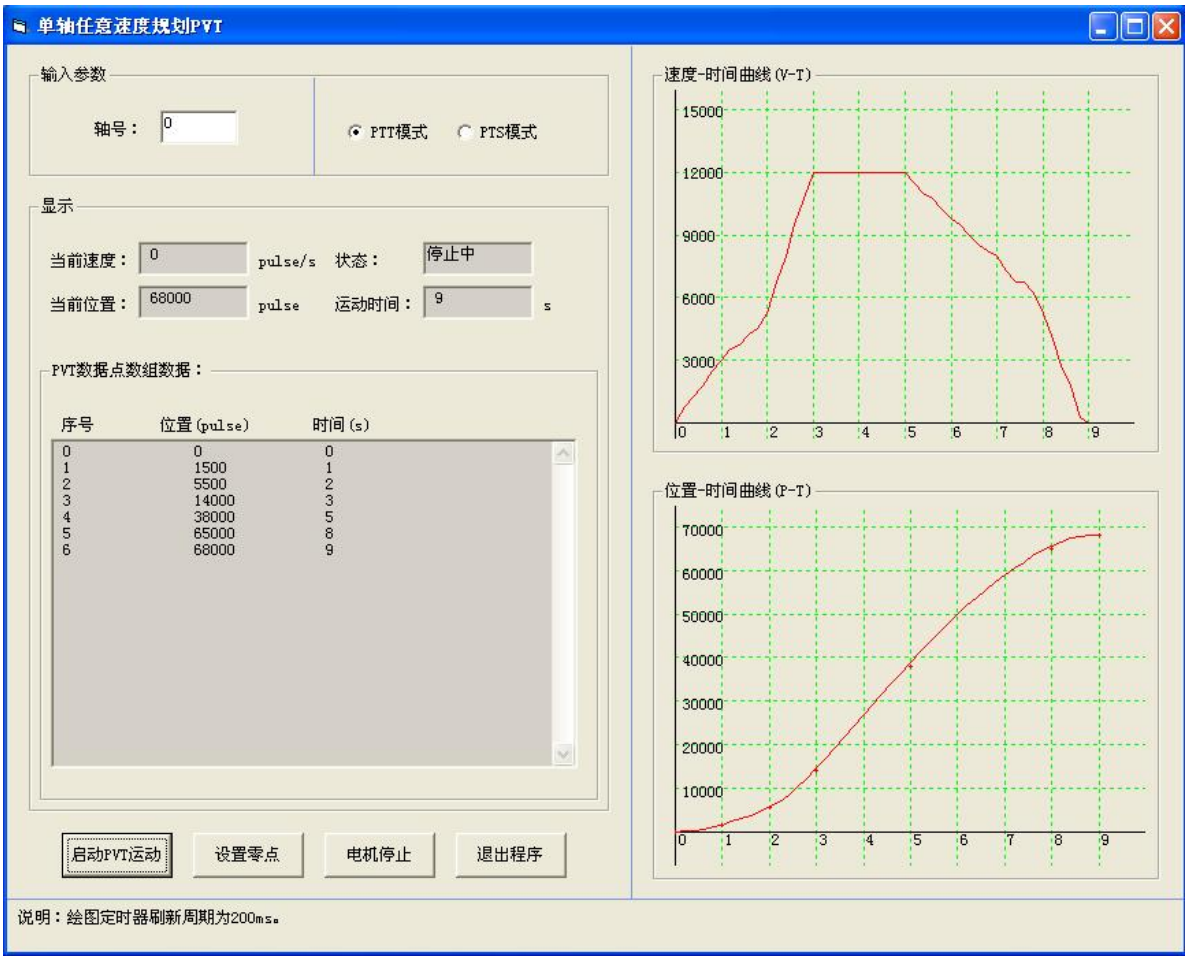


图 7.44　单轴任意速度规划 PVT 程序界面

7.18.11　两轴高级轨迹规划 PVT 实例

两轴高级轨迹实例程序界面如图 7.45 所示。PVT 模式运动的数据计算、函数调用参见例程 7.13。详细代码请参考光盘例程“例 11_两轴高级轨迹规划 PVT”。

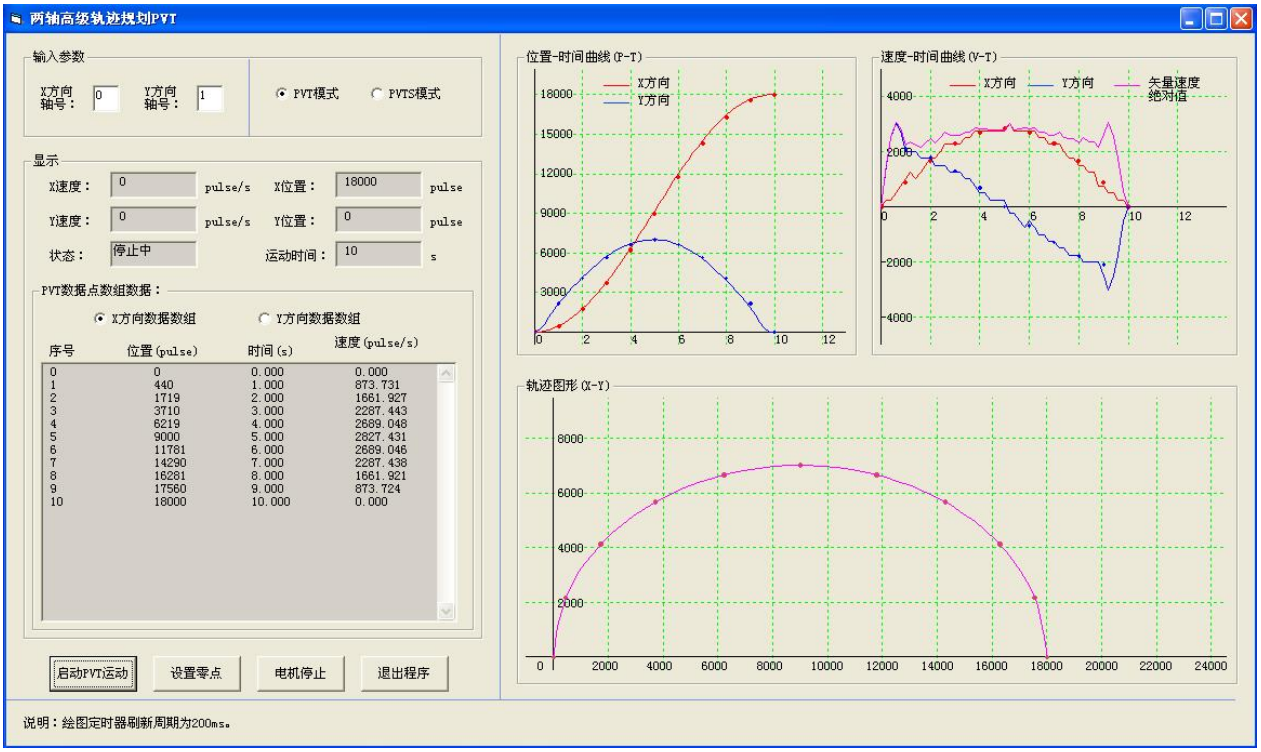


图 7.45　两轴高级轨迹规划 PVT 程序界面

7.18.12　三轴高级轨迹规划 PVT 实例

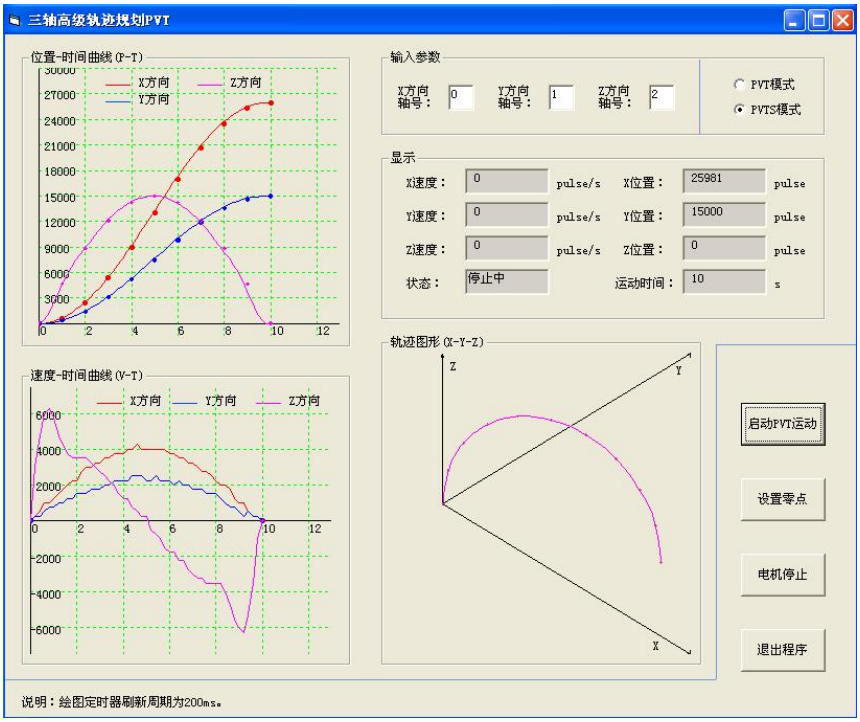


图 7.46　三轴高级轨迹规划 PVT 程序界面

三轴高级轨迹实例程序界面如图 7.46 所示。PVTS 模式运动的数据计算、函数调用参见例程 7.14。详细代码请参考光盘例程“例 12_三轴高级轨迹规划 PVT”。

7.18.13　二维低速位置比较实例

如图 7.47 所示，该程序中添加了两个二维位置比较点，运动轨迹为圆弧运动。程序中设有输出 IO 口 0、输出 IO 口 3 电平显示，X、Y 方向位置显示，并有轨迹图形显示，可以实时观察当前状态。设有“启动”、“输出 IO 口 3 置为低电平”、“退出程序”功能按钮。详细代码请参考光盘例程“例 13_二维低速位置比较”。

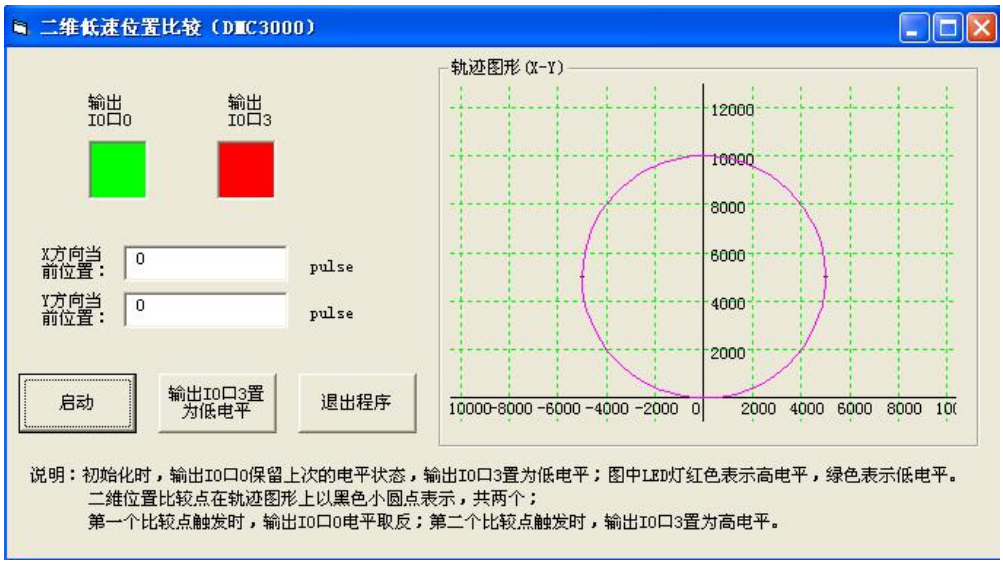


图 7.47　二维低速位置比较程序界面

7.18.14　CAN-IO 扩展模块实例

CAN-IO 扩展模块程序主界面如图 7.48 所示。通过界面，可以对 CAN-IO 扩展模块进行连接及断开操作，可以直观地观察 CAN-IO 扩展模块的输入输出端口电平状态，以及 CAN 通讯状态。通过直接点击各个输出 LED 灯，可以对各个输出端口进行操作。点击“循环点灯”按钮，可以对当前连接的所有 CAN-IO 扩展模块的输出端口进行循环点灯操作。详细代码请参考光盘例程“例 14_CAN-IO 扩展模块”。

图 7.48　CAN-IO 扩展模块程序界面

第 8 章　函数库详解

8.1　板卡设置函数

short dmc_board_init(void)
功　能：控制卡初始化函数，分配系统资源
参　数：无
返回值：0：　　没有找到控制卡，或者控制卡异常
　　1~8：　控制卡数
　　负值：表明有 2 张或 2 张以上控制卡的硬件设置卡号相同；返回值取绝对值后减 1 即为该卡号

short dmc_board_reset(void)
功　能：控制卡硬件复位函数
参　数：无
返回值：错误代码
注　意：执行复位操作后，必须等待 5 秒方可执行初始化控制卡，否则会出错。出错后必须重新执行复位操作，再等待 5 秒后执行初始化控制卡

short dmc_board_close(void)
功　能：控制卡关闭函数，释放系统资源
参　数：无
返回值：错误代码

short dmc_get_CardInfList (WORD* CardNun, DWORD* CardTypeList, WORD* CardIdList)
功　能：获取控制卡硬件 ID 号
参　数：CardNun　　返回初始化成功的卡数
　　CardTypeList　返回控制卡固件类型数组
　　CardIdList　　返回控制卡硬件 ID 号数组，卡号按从小到大顺序排列
返回值：错误代码

注　意：参数 CardTypeList 类型为十六进制

short dmc_get_card_version(WORD CardNo,DWORD *CardVersion)
功能：获取控制卡硬件版本号
参　数：CardNo　　控制卡卡号
　　CardVersion　　返回控制卡硬件版本号
返回值：错误代码

short dmc_get_card_soft_version(WORD CardNo,DWORD *FirmID,DWORD *SubFirmID)
功能：获取控制卡固件版本号
参　数：CardNo　　控制卡卡号
　　FirmID　　返回控制卡固件类型
　　SubFirmID　　返回控制卡固件版本号
返回值：错误代码
注　意：参数 FirmID 类型为十六进制

short dmc_get_card_lib_version(DWORD *LibVer)
功　能：获取控制卡动态库文件版本号
参　数：LibVer　　返回库版本号
返回值：错误代码

short dmc_get_total_axes(WORD CardNo,DWORD *TotalAxis)
功能：获取当前卡的轴数
参　数：CardNo　　控制卡卡号
　　TotalAxis　　返回当前卡的轴数
返回值：错误代码

short dmc_download_configfile(WORD CardNo,const char *FileName)
功　能：下载参数文件
参　数：CardNo　　控制卡卡号
　　FileName　　文件路径：
　　参数文件名+后缀：相对路径

完整描述参数文件的路径+文件名后缀：绝对路径

返回值：错误代码

注　意：1）当使用相对路径时，参数文件与程序必须在同一目录下

2）可以在 Motion3000 软件中“参数设置”界面下，将各轴参数设置好，然后点击“参数文件操作”-“读取”将参数文件保存。最后在编写程序时，使用函数 dmc_download_configfile 将参数文件下载

short　dmc_download_firmware(WORD CardNo,const char *FileName)

功　能：下载固件文件

参　数：CardNo　　控制卡卡号

FileName　　文件路径：

参数文件名+后缀：相对路径

完整描述参数文件的路径+文件名后缀：绝对路径

返回值：错误代码

注　意：1）当使用相对路径时，固件文件与程序必须在同一目录下

2）可以在控制卡 Motion 软件中右击主卡->“固件升级”菜单下直接升级固件

8.2　脉冲模式设置函数

short dmc_set_pulse_outmode(WORD CardNo,WORD axis,WORD outmode)

功　能：设置指定轴的脉冲输出模式

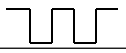
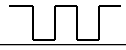
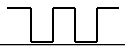
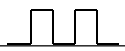
参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

outmode　　脉冲输出方式选择，其值如表 8.1 所示

返回值：错误代码

表 8.1　指令脉冲输出模式

输出脉冲类型 OUTMODE	正方向脉冲		负方向脉冲	
	PULSE 输出端	DIR 输出端	PULSE 输出端	DIR 输出端
0		高电平		低电平
1		高电平		低电平
2		低电平		高电平
3		低电平		高电平
4		高电平	高电平	
5		低电平	低电平	

注意：在调用运动函数（如：dmc_vmove 等）输出脉冲之前，一定要根据驱动器接收脉冲的模式调用 dmc_set_pulse_outmode 设置控制卡脉冲输出模式

short dmc_get_pulse_outmode(WORD CardNo,WORD axis,WORD* outmode)

功　能：读取指定轴的脉冲输出模式设置

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

outmode　　返回脉冲输出方式

返回值：错误代码

8.3　回原点运动函数

short dmc_set_home_pin_logic(WORD CardNo,WORD axis,WORD org_logic,double filter)

功　能：设置 ORG 原点信号

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

org_logic　　ORG 信号有效电平，0：低有效，1：高有效

filter　　保留参数，固定值为 0

返回值：错误代码

short dmc_get_home_pin_logic(WORD CardNo,WORD axis,WORD *org_logic,double *filter)

功　能：读取 ORG 原点信号设置
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　org_logic　　返回设置的 ORG 信号有效电平
　　filter　　保留参数
返回值：错误代码

short dmc_set_homemode(WORD CardNo,WORD axis,WORD home_dir,double vel_mode,
WORD mode,WORD EZ_count)
功　能：设置回原点模式
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　home_dir　　回零方向，0：负向，1：正向
　　vel_mode　　回零速度模式：
　　0：低速回零，即以本指令前面的 dmc_set_profile 函数设置的起始速度运行
　　1：高速回零，即以本指令前面的 dmc_set_profile 函数设置的最大速度运行
　　mode　　回零模式：
　　0：一次回零
　　1：一次回零加回找
　　2：二次回零
　　3：一次回零后再记一个同向 EZ 脉冲进行回零
　　4：记一个 EZ 脉冲进行回零
　　5：原点加反向 EZ
　　6：原点锁存
　　7：原点锁存加同向 EZ 锁存
　　8：单独记一个 EZ 锁存
　　9：原点锁存加反向 EZ 锁存
　　10. 一次限位回零

11. 一次限位回零加反找
12. 二次限位回零

EZ_count 保留参数，固定值为 0

返回值：错误代码

注意：1）当回零模式 mode=4 时，回零速度模式将固定为低速回零

2）DMC3C00 后四轴只支持 0、1、2、10、11、12 六种回零模式

3）后三种回零模式最新固件（3XX201611 及以后固件）才支持。正向回零时进行正限位回零，负向回零时进行负限位回零； 若开始回零时处于限位信号中，会先向设置的回零方向的反向运动，移出限位信号范围后，再变向，找相应的限位信号；一次限位回零遇到限位信号后急停；一次限位回零加反找在反找阶段遇到限位信号后急停；二次限位回零在第二次遇到限位信号后急停；

short dmc_get_homemode(WORD CardNo,WORD axis,WORD* home_dir, double* vel, WORD* mode, WORD* EZ_count)

功 能：读取回原点模式

参 数：CardNo 控制卡卡号

axis 指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5，DMC3400A：0~3

home_dir 返回回零方向

vel 返回回零速度模式

mode 返回回零模式

EZ_count 保留参数

返回值：错误代码

short dmc_set_home_el_return(WORD CardNo,WORD axis,WORD enable)

功 能：设置回零遇限位是否反找

参 数：CardNo 控制卡卡号

axis 指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5 DMC3400A：0~3

Enable 使能是否遇限位反找

返回值：错误代码

说　明：设置回零过程当中遇限位是否反找，需要在配置回零函数时配置，一次即可，以后均不用配置。DMC3C00，3400A 默认使能，DMC3800，3600 默认不使能。**限位反找功能只针对非限位回零方式起作用。**

short dmc_get_home_el_return(WORD CardNo,WORD axis,WORD *enable)
功　能：读取回零遇限位是否反找
参　数：CardNo　　控制卡卡号 0-7
　　axis　　指定轴号 0-11
　　Enable　　使能是否遇限位反找，0 不使能，1 使能
返回值：错误代码

short dmc_set_home_position(WORD CardNo,WORD axis,WORD enable,double position)
功　能：设置回零偏移量及清零模式
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5，DMC3400A：0~3
　　enable　　清零模式，0：不清零，1：回零完成后清零，再偏移；2：回零以及偏移完成后清零
　　position　　偏移量（正值正向偏移；负值负向偏移；0 无偏移）。正限位回零只能设置负向偏移，负限位回零只能设置正向偏移
返回值：错误代码

short dmc_get_home_position (WORD CardNo, WORD axis, WORD * enable, double * position)
功　能：读取回零偏移量及清零模式
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5　DMC3400A：0~3
　　enable　　读取清零模式
　　position　　读取回零偏移量
返回值：错误代码

short dmc_set_el_ret_deviation(WORD CardNo, WORD axis, WORD enable,double deviation)

功　能：设置开启限位反找偏移距离
参　数：CardNo:　　卡号；0-7
　　axis:　　轴号；0-11
　　enable:　　使能：1 使能；0 禁止；
　　deviation　　偏移距离，单位 pulse；
返回值：错误代码

short dmc_get_el_ret_deviation(WORD CardNo, WORD axis, WORD* enable, double* deviation)
功　能：读取限位反找偏移距离
参　数：CardNo:　　卡号；0-7
　　axis:　　轴号；0-11
　　enable:　　使能：1 使能；0 禁止；
　　deviation　　偏移距离；
返回值：错误代码

short dmc_home_move(WORD CardNo,WORD axis)
功　能：回原点运动
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
返回值：错误代码

short dmc_get_home_result(WORD CardNo,WORD axis,WORD* state)
功　能：读取回零执行状态
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　state　　回零执行状态，1：回零完成，0：回零未完成
返回值：错误代码

8.4 原点锁存函数

short dmc_set_homelatch_mode(WORD CardNo,WORD axis,WORD enable,WORD logic,
WORD source)
功　能：设置原点锁存模式
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00/3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　enable　　原点锁存使能，0：禁止，1：允许
　　logic　　触发方式，0：下降沿，1：上升沿
　　source　　位置源选择，0：指令位置计数器，1：编码器计数器
返回值：错误代码
注意:DMC3C00 后四轴不支持原点锁存功能

short dmc_get_homelatch_mode(WORD CardNo,WORD axis,WORD* enable,WORD* logic,
WORD* source)
功　能：读取原点锁存模式设置
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00/3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　enable　　返回原点锁存使能状态
　　logic　　返回触发方式
　　source　　返回位置源选择
返回值：错误代码
short dmc_reset_homelatch_flag(WORD CardNo,WORD axis)
功　能：清除原点锁存标志
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00/3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
返回值：错误代码

long dmc_get_homelatch_flag(WORD CardNo,WORD axis)

功　能：读取原点锁存标志
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00/3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
返回值：原点锁存标志，0：未锁存，1：锁存

long dmc_get_homelatch_value(WORD CardNo,WORD axis)
功　能：读取原点锁存值
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00/3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
返回值：锁存值，单位：pulse

8.5　限位开关设置函数

short dmc_set_el_mode(WORD CardNo,WORD axis,WORD el_enable,WORD el_logic,WORD el_mode)
功　能：设置 EL 限位信号
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　el_enable　　EL 信号的使能状态：
　　0：正负限位禁止
　　1：正负限位允许
　　2：正限位禁止、负限位允许
　　3：正限位允许、负限位禁止
　　el_logic　　EL 信号的有效电平：
　　0：正负限位低电平有效
　　1：正负限位高电平有效
　　2：正限位低有效，负限位高有效
　　3：正限位高有效，负限位低有效
　　el_mode　　EL 制动方式：
　　0：正负限位立即停止

1：正负限位减速停止
2：正限位立即停止，负限位减速停止
3：正限位减速停止，负限位立即停止

返回值：错误代码

short dmc_get_el_mode(WORD CardNo,WORD axis, WORD *el_enable,WORD *el_logic,WORD *el_mode)

功　能：读取 EL 限位信号设置

参　数：CardNo　　控制卡卡号
axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
el_enable　　返回设置的 EL 信号使能状态
el_logic　　返回设置的 EL 信号有效电平
el_mode　　返回 EL 制动方式

返回值：错误代码

short dmc_set_softlimit(WORD CardNo,WORD axis,WORD enable, WORD source_sel,WORD SL_action, long N_limit,long P_limit)

功　能：设置软限位

参　数：CardNo　　控制卡卡号
axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

enable　　使能状态，0：禁止，1：允许
source_sel　　计数器选择，0：指令位置计数器，1：编码器计数器
SL_action　　限位停止方式，0：立即停止，1：减速停止
N_limit　　负限位位置，单位：pulse
P_limit　　正限位位置，单位：pulse

返回值：错误代码

注　意：正、负限位位置可为正数也可为负数，但正限位位置应大于负限位位置

short dmc_get_softlimit(WORD CardNo,WORD axis,WORD* enable, WORD* source_sel,WORD*

SL_action, long* N_limit,long* P_limit)
功能：读取软限位设置
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　enable　　返回使能状态
　　source_sel　　返回计数器选择
　　SL_action　　返回限位停止方式
　　N_limit　　返回负限位脉冲数
　　P_limit　　返回正限位脉冲数
返回值：错误代码

short dmc_set_arc_zone_limit_config(WORD CardNo, WORD* AxisList, WORD AxisNum, double *Center, double Radius, WORD Source, WORD StopMode);
功　能：设置圆弧区域限位配置信息
参　数：CardNo: 卡号，0-7
AxisList: 需要限位的轴列表，0-实际轴数-1
AxisNum：需要限位的轴个数，固定为 2
Center：区域限位圆心，pulse 单位
Radius：区域限位半径，pulse 单位
Source：计数器选择，0 指令位置，1 反馈位置
StopMode: 限位触发后停止模式， 0 减速停止；1 急停；
返回值：错误代码

short dmc_get_arc_zone_limit_config(WORD CardNo, WORD* AxisList, WORD* AxisNum, double *Center, double * Radius, WORD* Source, WORD* StopMode);
功　能：读取圆弧区域限位配置信息
参　数：CardNo: 卡号，0-7
AxisList: 需要限位的轴列表，0-实际轴数-1
AxisNum：需要限位的轴个数，固定为 2
Center：区域限位圆心，pulse 单位
Radius：区域限位半径，pulse 单位

Source：计数器选择，0 指令位置，1 反馈位置
StopMode: 限位触发后停止模式， 0 减速停止；1 急停；
返回值：错误代码

short dmc_get _arc_zone_limit_axis_status(WORD CardNo, WORD AxisNo);
功　能：查询相应轴的状态
参　数：CardNo: 卡号，0-7
AxisNo: 轴号，0-实际轴数-1
返回值：-1 该轴没有被设置区域限位；0 该轴在区域内；1 该轴触发区域限位

short dmc_set_arc_zone_limit_enable(WORD CardNo, WORD enable);
功　能：配置圆弧限位功能使能状态
参　数：CardNo: 卡号，0-7
enable: 使能状态，0 未使能；1 使能
返回值：错误代码

short dmc_get_arc_zone_limit_enable(WORD CardNo, WORD* enable);
功　能：读取圆弧限位功能使能状态
参　数：CardNo: 卡号，0-7
enable: 使能状态，0 未使能；1 使能
返回值：错误代码

8.6　位置计数器控制函数

short dmc_set_position(WORD CardNo,WORD axis,long current_position)
功　能：设置指令脉冲位置
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　current_position　　指令脉冲位置，单位：pulse
返回值：错误代码
long dmc_get_position(WORD CardNo,WORD axis)

功　能：读取指令脉冲位置
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
返回值：指令脉冲位置，单位：pulse
注　意：如果开启螺距补偿功能，则该函数读取到的数值为补偿后轴的位置；

short dmc_get_position_ex(WORD CardNo,WORD axis, double * pos);
功　能：读取开启螺距补偿功能补偿前指令脉冲位置
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　Pos　　指令脉冲位置，单位：pulse
返回值：函数错误码

8.7　运动状态检测及控制相关函数

short dmc_get_stop_reason(WORD CardNo, WORD axis, long* StopReason)
功　能：读取指定轴的停止原因
参　数：CardNo　　卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　StopReason　　停止原因：
　　0：正常停止
　　1：ALM 立即停止，IMD_STOP_AT_ALM
　　2：ALM 减速停止，DEC_STOP_AT_ALM
　　3：LTC 外部触发立即停止，IMD_STOP_AT_LTC
　　4：EMG 立即停止，IMD_STOP_AT_EMG
　　5：正硬限位立即停止，IMD_STOP_AT_ELP
　　6：负硬限位立即停止，IMD_STOP_AT_ELN
　　7：正硬限位减速停止，DEC_STOP_AT_ELP
　　8：负硬限位减速停止，DEC_STOP_AT_ELN

9：正软限位立即停止，IMD_STOP_AT_SOFT_ELP
10：负软限位立即停止，IMD_STOP_AT_SOFT_ELN
11：正软限位减速停止，DEC_STOP_AT_SOFT_ELP
12：负软限位减速停止，DEC_STOP_AT_SOFT_ELN
13：命令立即停止，IMD_STOP_AT_CMD
14：命令减速停止，DEC_STOP_AT_CMD
15：其它原因立即停止，IMD_STOP_AT_OTHER
16：其它原因减速停止，DEC_STOP_AT_OTHER
17：未知原因立即停止，IMD_STOP_AT_UNKOWN
18：未知原因减速停止，DEC_STOP_AT_UNKOWN
19：外部 IO 减速停止，DEC_STOP_AT_DEC

返回值：错误代码

short dmc_clear_stop_reason(WORD CardNo, WORD axis)
功　能：清除指定轴的停止原因
参　数：CardNo　　卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3

返回值：错误代码

double　dmc_read_current_speed(WORD CardNo,WORD axis)
功 能：读取当前速度值
参 数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3

返回值：指定轴的速度，单位：pulse/s
注　意：当执行直线插补运动时，该函数读取的速度为矢量速度；当执行圆弧插补运动时，该函数读取的速度为各轴分量速度

short dmc_LinkState(WORD CardNo,WORD* State)
功　能：检测主卡与接线盒的通讯连接状态
参　数：CardNo　　控制卡卡号

State　　连接状态，0：连接，1：断开

返回值：错误代码

short dmc_set_output_status_repower (WORD CardNo, WORD enable);

功　能：接线盒断线或者断电后再次连接或者上电时，设置所有输出电平状态

参　数：CardNo：卡号

enable：1: 再次连接或者上电时候关闭所有输出口（包括伺服使能）;

0：再次连接或者上电时候保持断线或者断电之前的状态;

返回值：错误代码

注　意：enable 默认值为 0（保持接线盒断线或者断电之前的状态），且该功能 3XX201615 固件版本及更高版本固件才支持;

short　dmc_check_done(WORD CardNo,WORD axis)

功　能：检测指定轴的运动状态

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

返回值：0：指定轴正在运行，1：指定轴已停止

注　意：此函数适用于单轴、PVT 运动

short dmc_check_done_multicoor(WORD CardNo,WORD Crd)

功　能：检测坐标系的运动状态

参　数：CardNo　　控制卡卡号

Crd　　指定控制卡上的坐标系号（取值范围：0~1）

返回值：坐标系状态，0：正在使用中，1：正常停止

注　意：此函数适用于插补运动

DWORD dmc_axis_io_status(WORD CardNo,WORD axis)

功　能：读取指定轴有关运动信号的状态

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

返回值：见表 8.2

表 8.2　轴的运动信号状态

位号	信号名称	描述
0	ALM	1：表示伺服报警信号 ALM 为 ON；　0：OFF
1	EL+	1：表示正硬限位信号 +EL 为 ON；　0：OFF
2	EL-	1：表示负硬限位信号 - EL 为 ON；　0：OFF
3	EMG	1：表示急停信号 EMG 为 ON；　0：OFF
4	ORG	1：表示原点信号 ORG 为 ON；　0：OFF
6	SL+	1：表示正软限位信号+SL 为 ON；　0：OFF
7	SL-	1：表示负软件限位信号-SL 为 ON；　0：OFF
8	INP	1：表示伺服到位信号 INP 为 ON；　0：OFF
9	EZ	1：表示 EZ 信号为 ON；　0：OFF
10	RDY	1：表示伺服准备信号 RDY 为 ON；　0：OFF
11	DSTP	1：表示减速停止信号 DSTP 为 ON；　0：OFF
其他位	保留	

short　dmc_stop(WORD CardNo,WORD axis,WORD stop_mode)

功　能：指定轴停止运动

参　数：CardNo　　控制卡卡号

　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

　　stop_mode　　制动方式，0：减速停止，1：紧急停止

返回值：错误代码

注　意：此函数适用于单轴、PVT 运动

short dmc_stop_multicoor(WORD CardNo,WORD Crd,WORD stop_mode)

功　能：停止坐标系内所有轴的运动

参　数：CardNo　　控制卡卡号

　　Crd　　指定控制卡上的坐标系号（取值范围：0~1）

　　stop_mode　　制动方式，0：减速停止，1：立即停止

返回值：错误代码

注　意：此函数适用于插补运动

short dmc_emg_stop(WORD CardNo)
功 能：紧急停止所有轴
参 数：CardNo 控制卡卡号
返回值：错误代码
注 意：此函数适用于所有运动模式

long dmc_get_target_position(WORD CardNo,WORD axis)
功 能：读取正在运动的目标位置（绝对坐标）
参 数：CardNo 控制卡卡号
axis 指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
返回值：目标位置，单位：pulse

8.8 单轴运动速度曲线设置函数

short dmc_set_profile(WORD CardNo,WORD axis,double Min_Vel,double Max_Vel,double Tacc,double Tdec,double Stop_Vel)
功 能：设置单轴运动速度曲线
参 数：CardNo 控制卡卡号
axis 指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
Min_Vel 起始速度，单位：pulse/s（最大值为 4M）
Max_Vel 最大速度，单位：pulse/s（最大值为 4M）
Tacc 加速时间，单位：s（最小值为 0.001s）
Tdec 减速时间，单位：s（最小值为 0.001s）
Stop_Vel 停止速度，单位：pulse/s（最大值为 4M）
返回值：错误代码

short dmc_get_profile(WORD CardNo,WORD axis,double *Min_Vel,double *Max_Vel,double *Tacc,double *Tdec,double * Stop_Vel)
功 能：读取单轴运动速度曲线
参 数：CardNo 控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
Min_Vel　　返回起始速度，单位：pulse/s
Max_Vel　　返回最大速度，单位：pulse/s
Tacc　　返回加速时间，单位：s
Tdec　　返回减速时间，单位：s
Stop_Vel　　返回停止速度，单位：pulse/s
返回值：错误代码

short dmc_set_s_profile(WORD CardNo,WORD axis,WORD s_mode,double s_para)
功　能：设置单轴速度曲线 S 段参数值
参　数：CardNo　　控制卡卡号
axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
s_mode　　保留参数，固定值为 0
s_para　　S 段时间，单位：s；范围：0~0.5 s
返回值：错误代码

short dmc_get_s_profile(WORD CardNo,WORD axis,WORD s_mode, double *s_para)
功　能：读取单轴速度曲线 S 段参数值
参　数：CardNo　　控制卡卡号
axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
s_mode　　保留参数，固定值为 0
s_para　　返回设置的 S 段时间
返回值：错误代码

8.9　单轴运动函数

short dmc_pmove(WORD CardNo,WORD axis,long Dist,WORD posi_mode)
功　能：指定轴点位运动
参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
Dist　　目标位置，单位：pulse
posi_mode　　运动模式，0：相对坐标模式，1：绝对坐标模式

返回值：错误代码

注　意：当运动模式为相对坐标模式时，目标位置大于 0 时正向运动，小于 0 时反向运动。

short dmc_vmove(WORD CardNo,WORD axis,WORD dir)

功　能：指定轴连续运动

参　数：CardNo　　控制卡卡号
axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
dir　　运动方向，0：负方向，1：正方向

返回值：错误代码

short　dmc_change_speed(WORD CardNo,WORD axis,double Curr_Vel,double Taccdec)

功　能：在线改变指定轴的当前运动速度

参　数：CardNo　　控制卡卡号
axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
Curr_Vel　　改变后的运动速度，单位：pulse/s
Taccdec　　保留参数，固定值为 0

返回值：错误代码

注　意：1）该函数适用于单轴运动中的变速
2）变速一旦成立，该轴的默认运行速度将会被改写为 Curr_Vel，也即当调用 dmc_get_profile 回读速度参数时会发生与 dmc_set_profile 所设置的不一致的现象
3）在连续运动中 Curr_Vel 负值表示往负向变速，正值表示往正向变速。在点位运动中 Curr_Vel 只允许正值

short　dmc_reset_target_position(WORD CardNo,WORD axis,long dist,WORD posi_mode)

功　能：在线改变指定轴的当前目标位置

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
dist　　目标位置，单位：pulse
posi_mode　　保留参数，固定值为 0

返回值：错误代码

注　意：1）该函数只适用于点位运动中的变位
2）参数 dist 为绝对坐标位置值，无论当前的运动模式为绝对坐标还是相对坐标模式

short dmc_update_target_position(WORD CardNo,WORD axis,long dist,WORD posi_mode)

功　能：强行改变指定轴的当前目标位置（在线/非在线）

参　数：CardNo　　控制卡卡号
axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
dist　　目标位置，单位：pulse
posi_mode　　保留参数，固定值为 0

返回值：错误代码

注　意：1）该函数适用于指定轴停止状态或点位运动中的变位
2）参数 dist 为绝对坐标位置值，无论当前的运动模式为绝对坐标还是相对坐标模式

short dmc_t_pmove_extern(WORD CardNo, WORD axis, double MidPos, double TargetPos, double Min_Vel, double Max_Vel, double stop_Vel, double acc,double dec, WORD posi_mode);

功　能：指定轴按固定曲线做分段定长位移运动（可实现软着陆功能）

参　数：CardNo　　控制卡卡号
axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
MidPos　　第一段运动终点位置，单位：pulse
TargetPos　　第二段运动终点位置，单位：pulse
Min_Vel　　起始速度，单位：pulse/s
Max_Vel　　最大速度，单位：pulse/s
stop_Vel　　停止速度（第二段运动的运行速度），单位：pulse/s
acc　　加速时间，单位：S
dec　　减速时间，单位：S

posi_mode　0 相对模式，1 绝对模式

返回值：错误代码

8.10 插补速度曲线设置函数

short dmc_set_vector_profile_multicoor(WORD CardNo,WORD Crd, Double Min_Vel, double Max_Vel, double Tacc, double Tdec, double Stop_Vel)

功　能：设置插补速度

参　数：CardNo　控制卡卡号

Crd　指定控制卡上的坐标系号（取值范围：0~1）

Min_Vel　保留参数，固定值为 0

Max_Vel　合成最大速度，单位：pulse/s

Tacc　加减速时间，单位：s（最小值为 0.001s）

Tdec　保留参数，固定值为 0

Stop_Vel　保留参数，固定值为 0

返回值：错误代码

说　明：DMC3000 系列卡支持两个插补系（参数 Crd）。两个插补系的速度可独立设置，执行插补运动时两个插补系可独立进行插补运动（即可同时进行两组插补运动）

short dmc_get_vector_profile_multicoor(WORD CardNo, WORD Crd, Double *Min_Vel, double *Max_Vel, double *Tacc, double *Tdec, double *Stop_Vel)

功　能：读取插补速度设置

参　数：CardNo　控制卡卡号

Crd　指定控制卡上的坐标系号；取值范围：0~1

Min_Vel　保留参数

Max_Vel　返回合成最大速度，单位：pulse/s

Tacc　返回加减速时间，单位：s

Tdec　保留参数

Stop_Vel　保留参数

返回值：错误代码

8.11 插补运动函数

short dmc_line_multicoor(WORD CardNo, WORD Crd, WORD axisNum, WORD *axisList, long *DistList, WORD posi_mode)

功　能：直线插补运动

参　数：CardNo　　控制卡卡号

Crd　　指定控制卡上的坐标系号；取值范围：0~1

axisNum　　插补轴数，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

axisList　　插补轴列表

DistList　　插补轴目标位置列表，单位：pulse

posi_mode　　运动模式，0：相对坐标模式，1：绝对坐标模式

返回值：错误代码

注　意：检测直线插补状态应使用坐标系状态检测函数 dmc_check_done_multicoor；停止正在执行的直线插补运动应使用坐标系停止函数 dmc_stop_multicoor；

short dmc_arc_move_multicoor (WORD CardNo, WORD Crd ,WORD *AxisList,long *Target_Pos, long *Cen_Pos, WORD Arc_Dir, WORD posi_mode)

功　能：两轴圆弧插补运动，圆心位置+终点位置

参　数：CardNo　　控制卡卡号

Crd　　指定控制卡上的坐标系号；取值范围：0~1

AxisList　　轴列表数组

Target_Pos　　终点坐标，单位：pulse

Cen_Pos　　圆心坐标，单位：pulse

Arc_Dir　　圆弧方向，0：顺时针，1：逆时针

posi_mode　　运动模式，0：相对坐标模式，1：绝对坐标模式

返回值：错误代码

注　意：1）检测圆弧插补状态应使用坐标系状态检测函数 dmc_check_done_multicoor；停止正在执行的圆弧插补运动应使用坐标系停止函数 dmc_stop_multicoor

8.12 PVT 运动函数

short dmc_PttTable(WORD CardNo,WORD axis,DWORD count,double *pTime,long *pPos)

功　能：向指定数据表传送数据，采用 PTT 模式

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

count　　数据点个数，每个数据表具有 1000 个存储空间，每个数据点占用 1 个存储空间

pTime　　数据点时间数组，单位：s（精度：ms）；数组长度：count

pPos　　数据点位置数组，单位：pulse；数组长度：count

返回值：错误代码

注 意：1）下载的第一组（即起始点）数据中位置、时间必须为 0；数组中的数据都是以起始点的数据为参考点

2）调用该指令向数据表中传递数据时，会删除数据表中原先的数据，因此所有数据应当一次传送完毕。如果使用数据表的轴正在运动，禁止更新数据表

3）DMC3C00 后四轴不支持 PVT 功能

short dmc_PtsTable (WORD CardNo, WORD axis, DWORD count, double *pTime, long *pPos, double *pPercent)

功　能：向指定数据表传送数据，采用 PTS 模式

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

count　　数据点个数，每个数据表具有 1000 个存储空间，每个数据点占用 1 个存储空间

pTime　　数据点时间数组，单位：s（精度：ms）；数组长度：count

pPos　　数据点位置数组，单位：pulse；数组长度：count

pPercent　　数据点百分比数组，百分比的取值范围：[0,100]；数组长度：count

返回值：错误代码

注 意：1）下载的第一组（即起始点）数据中位置、时间必须为 0；数组中的数据都是以起始点的数据为参考点

2）调用该指令向数据表中传递数据时，会删除数据表中原有数据，因此所有数据应当一次传送完毕。如果使用数据表的轴正在运动，禁止更新数据表

short dmc_PvtTable(WORD CardNo,WORD axis,DWORD count,double *pTime,long *pPos,double *pVel)

功　能：向指定数据表传送数据，采用 PVT 模式

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

count　　数据点个数，每个数据表具有 5000 个存储空间，每个数据点占用 1 个存储空间

pTime　　数据点时间数组，单位：s（精度：ms）；数组长度：count

pPos　　数据点位置数组，单位：pulse；数组长度：count

pVel　　数据点速度数组，单位：pulse/s；数组长度：count

返回值：错误代码

注 意：1）下载的第一组（即起始点）数据中位置、时间、速度必须为 0；数组中的数据都是以起始点的数据为参考点

2）调用该指令向数据表中传递数据时，会删除数据表中原有数据，因此所有数据应当一次传送完毕。如果使用数据表的轴正在运动，禁止更新数据表

short dmc_PvtsTable (WORD CardNo,WORD axis,DWORD count,double *pTime,long *pPos,double velBegin,double velEnd)

功　能：向指定数据表传送数据，采用 PVTS 模式

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

count　　数据点个数，每个数据表具有 5000 个存储空间，每个数据点占用 1 个存储空间

pTime　　数据点时间数组，单位：s（精度：ms）；数组长度：count

pPos　　数据点位置数组，单位：pulse；数组长度：count

velBegin　　设置的第一点的速度，单位：pulse/s

velEnd　　设置的最后一点的速度，单位：pulse/s

返回值：错误代码

注 意：1）下载的第一组（即起始点）数据中位置、时间、速度必须为 0；数组中的数据都是以起始点的数据为参考点

2）调用该指令向数据表中传递数据时，会删除数据表中原有数据，因此所有数据应当一次传送完毕。如果使用数据表的轴正在运动，禁止更新数据表

short dmc_PvtMove(WORD CardNo,WORD AxisNum,WORD* AxisList)

功　能：启动 PVT 运动

参　数：CardNo　　控制卡卡号

AxisNum　轴数

AxisList　轴列表

返回值：错误代码

8.13　伺服驱动专用接口函数

short　dmc_write_sevon_pin(WORD CardNo,WORD axis,WORD on_off)

功　能：控制指定轴的伺服使能端口的输出

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

on_off　　设置伺服使能端口电平，0：低电平，1：高电平

返回值：错误代码

short　dmc_read_sevon_pin(WORD CardNo,WORD axis)

功　能：读取指定轴的伺服使能端口的电平

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

返回值：伺服使能端口电平，0：低电平，1：高电平

short　dmc_read_rdy_pin(WORD CardNo,WORD axis)

功　能：读取指定轴的 RDY 端口的电平

参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
返回值：RDY 端口电平，0：低电平，1：高电平

short　dmc_set_inp_mode(WORD CardNo,WORD axis,WORD enable,WORD inp_logic)
功　能：设置指定轴的 INP 信号
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　enable　　INP 信号使能，0：禁止，1：允许
　　inp_logic　　INP 信号的有效电平，0：低有效，1：高有效
返回值：错误代码
注意：当使能 INP 信号功能后，只有在 INP 信号为有效状态时，对应的轴才能进行运动，否则此时检测轴的状态是正在运行（即对轴运动作限制）

short　dmc_get_inp_mode(WORD CardNo,WORD axis,WORD *enable,WORD *inp_logic)
功　能：读取指定轴的 INP 信号设置
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　enable　　返回 INP 信号使能状态
　　inp_logic　　返回设置的 INP 信号有效电平
返回值：错误代码

short　dmc_set_alm_mode(WORD CardNo,WORD axis,WORD enable,WORD alm_logic,WORD alm_action)
功　能：设置指定轴的 ALM 信号
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　enable　　ALM 信号使能，0：禁止，1：允许

alm_logic　　ALM 信号的有效电平，0：低有效，1：高有效
alm_action　　ALM 信号的制动方式，0：立即停止（只支持该方式）
返回值：错误代码

short dmc_get_alm_mode(WORD CardNo,WORD axis,WORD *enable,WORD *alm_logic,
WORD *alm_action)
功　能：读取指定轴的 ALM 信号设置
参　数：CardNo　　控制卡卡号
axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
enable　　返回 ALM 信号使能状态
alm_logic　　返回设置的 ALM 信号有效电平
alm_action　　返回 ALM 信号的制动方式
返回值：错误代码

short　dmc_write_erc_pin(WORD CardNo,WORD axis,WORD sel)
功　能：控制指定轴的 ERC 信号输出
参　数：CardNo　　控制卡卡号
axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
sel　　输出电平，0：低电平，1：高电平
返回值：错误代码

short dmc_read_erc_pin(WORD CardNo,WORD axis)
功　能：读取指定轴的 ERC 端口电平
参　数：CardNo　　控制卡卡号
axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
返回值：ERC 端口电平，0：低电平，1：高电平

8.14 通用输入输出 IO 函数

short dmc_read_inbit(WORD CardNo, WORD bitno)
功　能：读取指定控制卡的某个输入端口的电平
参　数：CardNo　　控制卡卡号
　　bitno　　输入端口号，取值范围：0~15
返回值：指定的输入端口电平：0：低电平，1：高电平

short dmc_write_outbit(WORD CardNo, WORD bitno,WORD on_off)
功　能：设置指定控制卡的某个输出端口的电平
参　数：CardNo　　控制卡卡号
　　bitno　　输出端口号，取值范围：0~15
　　on_off　　输出电平，0：低电平，1：高电平
返回值：错误代码

short dmc_read_outbit(WORD CardNo, WORD bitno)
功　能：读取指定控制卡的某个输出端口的电平
参　数：CardNo　　控制卡卡号
　　bitno　　输入端口号，取值范围：0~15
返回值：指定输出端口的电平，0：低电平，1：高电平

DWORD dmc_read_inport(WORD CardNo,WORD portno)
功　能：读取指定控制卡的全部输入端口的电平
参　数：CardNo　　控制卡卡号
　　portno　　IO 组号，取值范围：0、1
返回值：bit0~bit31 的定义见表 8.3

表 8.3　dmc_read_inport 函数返回值各位的定义表

第 0 组（portno 参数）			第 1 组（portno 参数）		
函数返回值的 bit	输入口号	输入口名称	函数返回值的 bit	输入口号	输入口名称
0	0	IN0	0	32	ORG0
1	1	IN1	1	33	ORG1
2	2	IN2	2	34	ORG2

第 0 组（portno 参数）			第 1 组（portno 参数）		
函数返回值的 bit	输入口号	输入口名称	函数返回值的 bit	输入口号	输入口名称
3	3	IN3	3	35	ORG3
4	4	IN4	4	36	ORG4
5	5	IN5	5	37	ORG5
6	6	IN6	6	38	ORG6
7	7	IN7	7	39	ORG7
8	8	IN8	8	40	ALM0
9	9	IN9	9	41	ALM1
10	10	IN10	10	42	ALM2
11	11	IN11	11	43	ALM3
12	12	IN12	12	44	ALM4
13	13	IN13	13	45	ALM5
14	14	IN14/LTC0	14	46	ALM6
15	15	IN15/LTC1	15	47	ALM7
16	16	EL0+	16	48	RDY0
17	17	EL1+	17	49	RDY1
18	18	EL2+	18	50	RDY2
19	19	EL3+	19	51	RDY3
20	20	EL4+	20	52	RDY4
21	21	EL5+	21	53	RDY5
22	22	EL6+	22	54	RDY6
23	23	EL7+	23	55	RDY7
24	24	EL0-	24	56	INP0
25	25	EL1-	25	57	INP1
26	26	EL2-	26	58	INP2
27	27	EL3-	27	59	INP3
28	28	EL4-	28	61	INP4
29	29	EL5-	29	61	INP5
30	30	EL6-	30	62	INP6
31	31	EL7-	31	63	INP7

DWORD dmc_read_outport(WORD CardNo,WORD portno)

功　能：读取指定控制卡的全部输出口的电平

参　数：CardNo　　控制卡卡号

　　portno　　保留参数，固定值为 0

返回值：bit0~bit31 的定义见表 8.4

short dmc_write_outport(WORD CardNo, WORD portno,DWORD port_value)

功　能：设置指定控制卡的全部输出口的电平

参　数：CardNo　　控制卡卡号

portno　　保留参数，固定值为 0

port_value　　bit0~bit31 的定义见表 8.4

返回值：错误代码

表 8.4　dmc_read_outport、dmc_write_outport 函数返回值各位的定义表

函数返回值的 bit	输出口号	输出口名称	函数返回值的 bit	输出口号	输出口名称
0	0	OUT0	16	16	SEVON0
1	1	OUT1	17	17	SEVON1
2	2	OUT2	18	18	SEVON2
3	3	OUT3	19	19	SEVON3
4	4	OUT4	20	20	SEVON4
5	5	OUT5	21	21	SEVON5
6	6	OUT6	22	22	SEVON6
7	7	OUT7	23	23	SEVON7
8	8	OUT8	24	24	ERC0
9	9	OUT9	25	25	ERC1
10	10	OUT10	26	26	ERC2
11	11	OUT11	27	27	ERC3
12	12	OUT12/CMP0	28	28	ERC4
13	13	OUT13/CMP1	29	29	ERC5
14	14	OUT14/CMP2	30	30	ERC6
15	15	OUT15/CMP3	31	31	ERC7

short dmc_reverse_outbit(WORD CardNo,WORD bitno,double reverse_time)

功　能：IO 输出延时翻转

参　数：CardNo　　控制卡卡号

bitno　　输出端口号,取值范围：0~15

reverse_time　　延时翻转时间，单位：s

返回值：错误代码

注　意：1）该函数只能对 OUT0 ~ OUT15 端口进行操作

2）当延时翻转时间参数设置为 0 时，此时延时翻转时间将为无限大

short dmc_set_io_count_mode(WORD CardNo,WORD bitno,WORD mode,double filter_time)
功　能：设置 IO 计数模式
参　数：CardNo　　控制卡卡号
　　bitno　　输入端口号，取值范围：0~15
　　mode　　IO 计数模式，0：禁用，1：上升沿计数，2：下降沿计数
　　filter_time　　滤波时间，单位：s
返回值：错误代码

short dmc_get_io_count_mode(WORD CardNo,WORD bitno,WORD *mode,double* filter_time)
功　能：读取 IO 计数模式设置
参　数：CardNo　　控制卡卡号
　　bitno　　输入端口号，取值范围：0~15
　　mode　　返回 IO 计数模式
　　filter_time　　返回滤波时间，单位：s
返回值：错误代码

short dmc_set_io_count_value(WORD CardNo,WORD bitno,DWORD CountValue)
功　能：设置 IO 计数值
参　数：CardNo　　控制卡卡号
　　bitno　　输入端口号，取值范围：0~15
　　CountValue　　IO 计数值
返回值：错误代码

short dmc_get_io_count_value(WORD CardNo,WORD bitno,DWORD* CountValue)
功　能：读取 IO 计数值
参　数：CardNo　　控制卡卡号
　　bitno　　输入端口号，取值范围：0~15
　　CountValue　　返回 IO 计数值
返回值：错误代码

8.15 手轮功能函数

short dmc_set_handwheel_inmode (WORD CardNo,WORD axis,WORD inmode,long multi, double vh)

功　能：设置单轴手轮运动控制输入方式

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

inmode　　手轮输入方式，0：A、B 相位正交信号；1：脉冲+方向信号

multi　　手轮倍率，正数表示默认方向，负数表示与默认方向反向

vh　　保留参数，固定值为 0

返回值：错误代码

注　意：DMC3C00/3400A 只有低速手轮通道，DMC3800/3600 有高、低速两种手轮通道，手轮外部硬件接口只有一个，通过该函数可以设置这个手轮通道对应的轴号（即每次只能控制一个轴运动）

short dmc_get_handwheel_inmode (WORD CardNo,WORD axis,WORD* inmode,long＊multi, double* vh)

功　能：读取单轴手轮运动控制输入方式

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

inmode　　返回手轮输入方式

multi　　返回手轮倍率

vh　　保留参数

返回值：错误代码

short dmc_handwheel_move(WORD CardNo,WORD axis)

功　能：启动手轮运动

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

返回值：错误代码

注　意：当启动手轮运动后，只有发送 dmc_stop 或 dmc_emg_stop 命令后才会退出手轮模式

short　dmc_set_handwheel_channel(WORD CardNo,WORD index)

功　能：手轮通道选择设置

参　数：CardNo　　控制卡卡号

　　index　　0：高速通道

　　1：低速通道

返回值：错误代码

手轮通道的说明：使用高速通道与低速通道的效果是一样的，其区别是：高速通道是通过 ACC3800/3600/XC00 接线盒 CN18 口连接到控制卡；低速通道是通过 ACC3800/3600/X400B/XC00 接线盒 CN17 口连接到控制卡

short dmc_get_handwheel_channel(WORD CardNo,WORD* index)

功　能：读取手轮通道选择设置

参　数：CardNo　　控制卡卡号

　　index　　返回设置的手轮通道

返回值：错误代码

short　dmc_set_handwheel_inmode_extern(WORD　CardNo,WORD　inmode,WORD　AxisNum,WORD* AxisList,long* multi)

功能：设置多轴手轮运动控制输入方式

参　数：CardNo　　控制卡卡号

　　inmode　　手轮输入方式：0：A、B 相位正交信号；1：脉冲+方向信号

　　AxisNum　　参与手轮运动的轴数

　　AxisList　　参与手轮运动的轴号数组

　　multi　　手轮倍率数组:正数表示默认方向，负数表示与默认方向反向

返回值：错误代码

注　意：通过该函数设置可以使一个手轮通道控制多个轴同时运动

short　dmc_get_handwheel_inmode_extern(WORD　CardNo,WORD　*inmode,WORD　*AxisNum,WORD* AxisList,long* multi)

功能：读取多轴手轮运动控制输入方式

参　数：CardNo　　控制卡卡号

　　inmode　　返回手轮输入方式

AxisNum　　返回参与手轮运动的轴数
AxisList　　返回参与手轮运动的轴号数组
multi　　返回手轮倍率数组
返回值：错误代码

8.16 编码器函数

short dmc_set_counter_inmode(WORD CardNo,WORD axis,WORD mode)
功　能：设置编码器的计数方式
参　数：CardNo　　控制卡卡号
axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
mode　　编码器的计数方式：
0：非 A/B 相(脉冲/方向)
1：1×A/B
2：2×A/B
3：4×A/B
返回值：错误代码

short dmc_get_counter_inmode(WORD CardNo,WORD axis,WORD *mode)
功　能：读取编码器的计数方式
参　数：CardNo　　控制卡卡号
axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
mode　　返回编码器的计数方式
返回值：错误代码

short dmc_set_encoder_dir(WORD CardNo, WORD axis,WORD dir);
功　能：设置编码器方向
参　数：CardNo　　控制卡卡号 0-7
axis　　指定轴号 0-8
dir　　编码器配置方向：0：A 在前为正 1：B 在前为正

返回值：错误代码

short dmc_set_encoder(WORD CardNo,WORD axis,long encoder_value)
功　能：设置指定轴编码器脉冲计数值
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　encoder_value　　编码器计数值，单位：pulse
返回值：错误代码
说　明：此函数 axis 参数为 8 时可以设置手轮编码器计数值，DMC3400A 无高速手轮通道，不支持设置手轮编码器计数值；

long dmc_get_encoder(WORD CardNo,WORD axis)
功　能：读取指定轴编码器脉冲计数值
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
返回值：编码器计数值，单位：pulse
说　明：此函数 axis 参数为 8 时可以读手轮编码器计数值，DMC3400A 无高速手轮通道，不支持读取手轮编码器计数值；如果开启螺距补偿功能，则该函数读取到的数值为补偿后轴的编码器位置；

short dmc_get_encoder_ex(WORD CardNo,WORD axis, double * pos)
功　能：读取指定轴螺距补偿开启补偿前编码器脉冲计数值
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　Pos　　编码器计数值
返回值：函数错误码

short dmc_set_ez_mode(WORD CardNo,WORD axis,WORD ez_logic,WORD ez_mode,double filter)
功　能：设置指定轴的 EZ 信号

参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　ez_logic　EZ 信号有效电平，0：低有效，1：高有效
　　ez_mode　　保留参数，固定值为 0
　　filter　　保留参数，固定值为 0
返回值：错误代码

short　dmc_get_ez_mode(WORD　CardNo,WORD　axis,WORD　*ez_logic,WORD　*ez_mode,double *filter)
功　能：读取指定轴的 EZ 信号设置
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　ez_logic　返回设置的 EZ 信号有效电平
　　ez_mode　　保留参数
　　filter　　保留参数
返回值：错误代码

8.17　高速位置锁存函数

short　dmc_set_ltc_mode(WORD　CardNo,WORD　axis,WORD　ltc_logic,WORD　ltc_mode,double filter)
功　能：设置指定轴的 LTC 信号
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　ltc_logic　LTC 信号的触发方式，0：下降沿锁存，1：上升沿锁存，**2：双边沿锁存**
　　ltc_mode　保留参数，固定值为 0
　　filter　　保留参数，固定值为 0
返回值：错误代码
注　意：DMC3C00 后四轴不支持高速位置锁存功能

short dmc_get_ltc_mode(WORD CardNo,WORD axis,WORD *ltc_logic,WORD *ltc_mode,double *filter)

功　能：读取指定轴的 LTC 信号设置

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

ltc_logic　返回设置的 LTC 信号的触发方式

ltc_mode　保留参数

filter　　保留参数

返回值：错误代码

short dmc_set_latch_mode(WORD CardNo,WORD axis,WORD all_enable,WORD latch_source, WORD triger_chunnel)

功　能：设置锁存方式

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

all_enable　锁存方式：
0：单次锁存
1：保留
2：连续锁存
3：触发延时急停

latch_source　锁存源，0：指令位置计数器，1：编码器计数器

triger_chunnel　保留参数，固定值为 0

返回值：错误代码

注　意：1）DMC3400A 只有一个高速锁存通道 LTC0，DMC3C00/3800/3600，有两个高速锁存通道 LTC0，LTC1；LTC0 锁存 0~3 号轴，LTC1 锁存 4~7 号轴

2）DMC3400A 中触发延时急停模式只对 0 号轴起作用，DMC3C00/3800/3600 中，触发延时急停模式只对 0 号轴及 4 号轴起作用；LTC0 对应为 0 号轴，LTC1 对应为 4 号轴

short dmc_get_latch_mode(WORD CardNo, WORD axis, WORD* all_enable, WORD* latch_source,

WORD* triger_chunnel)

功　能：读取锁存方式

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

all_enable　　返回锁存方式设置

latch_source　　返回锁存源设置

triger_chunnel　　保留参数

返回值：错误代码

short dmc_set_latch_stop_time(WORD CardNo,WORD axis,long time)

功　能：设置 LTC 端口触发延时急停时间

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3400A：0，DMC3C00/3800/3600：0、4

time　　触发延时停止时间，单位：us，取值范围：1us~50ms

返回值：错误代码

注　意：触发延时急停模式只对 0 号轴及 4 号轴起作用；LTC0 对应为 0 号轴，LTC1 对应为 4 号轴

short dmc_get_latch_stop_time(WORD CardNo,WORD axis,long* time)

功　能：读取 LTC 端口触发延时急停时间

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3400A：0，DMC3C00/3800/3600：0、4

time　　返回触发延时停止时间设置，单位：us

返回值：错误代码

short dmc_SetLtcOutMode(WORD CardNo,WORD axis,WORD enable,WORD bitno)

功　能：LTC 反相输出设置

参　数：CardNo　控制卡卡号

Axis　指定轴号，取值范围：DMC3C00/3800：0~7，DMC3600：0~5,DMC3400A：0~3

Enable　使能状态：0：禁止，1：使能

bitno　通用输出 IO 口号，取值范围：0~15

返回值：错误代码

注　意：当某输出端口作为 LTC 反相输出后，该端口将不能通过通用 IO 函数设置输出值；

short dmc_GetLtcOutMode(WORD CardNo,WORD axis,WORD *enable,WORD* bitno)

功　能：读取 LTC 反相输出设置

参　数：CardNo　控制卡卡号

axis　指定轴号，取值范围：DMC3C00/3800：0~7，DMC3600：0~5,DMC3400A：0~3

enable　返回使能状态：0：禁止，1：使能

bitno　返回设置的通用输出 IO 口号

返回值：错误代码

long dmc_get_latch_value(WORD CardNo,WORD axis)

功　能：从控制卡内读取锁存器的值

参　数：CardNo　控制卡卡号

axis　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

返回值：锁存值

注　意：当选择锁存方式为单次锁存时，用此函数读取锁存值

short dmc_get_latch_flag(WORD CardNo,WORD axis)

功　能：从控制卡内读取指定卡内锁存器的标志位

参　数：CardNo　控制卡卡号

axis　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

返回值：0：未触发锁存，1：已触发锁存

注　意：当选择锁存方式为单次锁存时，用此函数读取锁存器标志位

short dmc_reset_latch_flag(WORD CardNo,WORD axis)

功　能：复位指定卡的锁存器的标志位

参　数：CardNo　控制卡卡号

axis　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

返回值：错误代码

注　意：当使用锁存功能前，必须先调用此函数复位锁存器的标志位

short dmc_get_latch_flag_extern(WORD CardNo,WORD axis)

功　能：从 PC 缓存中读取锁存器已锁存个数

参　数：CardNo　　控制卡卡号

　　axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

返回值：当前已锁存个数，0 表示此时无锁存值

注　意：当选择锁存方式为连续锁存时，用此函数读取已锁存个数

long dmc_get_latch_value_extern(WORD CardNo,WORD axis,WORD index)

功　能：按索引号读取 PC 缓冲区中已保存的锁存值

参　数：CardNo　　控制卡卡号

　　axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

　　index　　索引号

返回值：锁存值

注　意：当选择锁存方式为连续锁存时，用此函数读取锁存值。索引号按锁存顺序从 0 开始排列（即第一次锁存的位置值存在索引号为 0 处，第二次锁存的位置值存在索引号为 1 处，以此类推）

8.18　位置比较函数

short dmc_compare_set_config(WORD CardNo, WORD axis,WORD enable, WORD cmp_source)

功　能：设置一维位置比较器

参　数：CardNo　　控制卡卡号

　　axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

　　enable　　比较功能状态，0：禁止，1：使能

　　cmp_source　　比较源，0：指令位置计数器，1：编码器计数器

返回值：错误代码

注　意：DMC3C00 后四轴不支持位置比较功能

short dmc_compare_get_config(WORD CardNo, WORD axis,WORD* enable,WORD* cmp_source)

功　能：读取一维位置比较器设置

参　数：CardNo　　控制卡卡号

　　axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

　　enable　　返回比较功能状态

　　cmp_source　　返回比较源

返回值：错误代码

short dmc_compare_clear_points(WORD CardNo,WORD axis)

功　能：清除已添加的所有一维位置比较点

参　数：CardNo　　控制卡卡号

　　axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

返回值：错误代码

short dmc_compare_add_point(WORD CardNo,WORD axis, long pos, WORD dir, WORD action, DWORD actpara)

功　能：添加一维位置比较点

参　数：CardNo　　控制卡卡号

　　axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

　　pos　　比较位置，单位：pulse

　　dir　　比较模式，0：小于等于，1：大于等于

　　action　　比较点触发功能编号，见表 8.5

　　actpara　　比较点触发功能参数，见表 8.5

返回值：错误代码

表 8.5　dmc_compare_add_point 函数 action, actpara 参数值

action	actpara	功能
1	IO 号	IO 置为高电平
2	IO 号	IO 置为低电平
3	IO 号	取反 IO

action	actpara	功能
5	IO 号	输出 500us 脉冲
6	IO 号	输出 1ms 脉冲
7	IO 号	输出 10ms 脉冲
8	IO 号	输出 100ms 脉冲
13	轴号	停止指定轴
15	变速值（单位：pulse/s）	在线变速

short dmc_compare_get_current_point(WORD CardNo,WORD axis,long* pos)
功　能：读取当前一维比较点位置
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　pos　　返回当前比较点位置，单位：pulse
返回值：错误代码

short dmc_compare_get_points_runned(WORD CardNo,WORD axis,long* PointNum)
功　能：查询已经比较过的一维比较点个数
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　PointNum　　返回已经比较过的点数
返回值：错误代码

short dmc_compare_get_points_remained(WORD CardNo,WORD axis,long* PointNum)
功　能：查询可以加入的一维比较点个数
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
　　DMC3400A：0~3
　　PointNum　　返回可以加入的比较点数
返回值：错误代码

short dmc_compare_set_config_extern (WORD CardNo, WORD enable, WORD cmp_source)

功　能：设置二维位置比较器
参　数：CardNo　　控制卡卡号
　　enable　　二维位置比较功能状态，0：禁止，1：使能
　　cmp_source　　二维位置比较源，0：指令位置计数器，1：编码器计数器
返回值：错误代码

short dmc_compare_get_config_extern(WORD CardNo, WORD* enable,WORD* cmp_source)
功　能：读取二维位置比较器设置
参　数：CardNo　　控制卡卡号
　　enable　　返回比较功能状态
　　cmp_source　　返回比较源
返回值：错误代码

short dmc_compare_clear_points_extern(WORD CardNo)
功　能：清除已添加的所有二维位置比较点
参　数：CardNo　　控制卡卡号
返回值：错误代码

short dmc_compare_add_point_extern(WORD CardNo,WORD* axis, long* pos, WORD* dir, WORD action, DWORD actpara)
功　能：添加二维位置比较点
参　数：CardNo　　控制卡卡号
　　axis　　指定卡上的即将进行位置比较的轴列表(两个轴)
　　pos　　二维位置比较位置列表，单位：pulse
　　dir　　比较模式列表，0：小于等于，1：大于等于
　　action　　二维位置比较点触发功能编号，见表 8.6
　　actpara　　二维位置比较点触发功能参数，见表 8.6
返回值：错误代码

表 8.6　dmc_compare_add_point_ex 函数 action, actpara 参数值

action	actpara	功能
1	IO 号	IO 置为高电平
2	IO 号	IO 置为低电平

action	actpara	功能
3	IO 号	取反 IO
5	IO 号	输出 500us 脉冲
6	IO 号	输出 1ms 脉冲
7	IO 号	输出 10ms 脉冲
8	IO 号	输出 100ms 脉冲
13	轴号	停止指定轴

short dmc_compare_get_current_point_extern(WORD CardNo,long *pos)
功　能：读取当前二维位置比较点位置
参　数：CardNo　　控制卡卡号
　　pos　　返回当前二维位置比较点位置，单位：pulse
返回值：错误代码

short dmc_compare_get_points_runned_extern(WORD CardNo,long *PointNum)
功　能：查询已经比较过的二维比较点个数
参　数：CardNo　　控制卡卡号
　　PointNum　　返回已经比较过的二维位置比较点数
返回值：错误代码

short dmc_compare_get_points_remained_extern(WORD CardNo, long *PointNum)
功　能：查询可以加入的二维比较点个数
参　数：CardNo　　控制卡卡号
　　PointNum　　返回可以加入的二维位置比较点数
返回值：错误代码

8.19　高速位置比较函数

short dmc_hcmp_set_mode(WORD CardNo,WORD hcmp, WORD cmp_mode)
功　能：设置高速比较模式
参　数：CardNo　　控制卡卡号
　　hcmp　　高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）
　　cmp_mode　比较模式：

0：禁止（默认值）
1：等于
2：小于
3：大于
4：队列，提供 127 个点比较空间，采用先添加先比较，比较完可追加比较点，也可一次性添加多个比较点
5：线性，提供起始比较点，位置增量，比较次数

返回值：错误代码

注　意：1）当选择模式 1 时，只有当前位置等于比较位置时，CMP 端口才输出有效电平
2）当选择模式 2 时，只要当前位置小于比较位置时，CMP 端口就一直保持有效电平
3）当选择模式 3 时，只要当前位置大于比较位置时，CMP 端口就一直保持有效电平
4）当选择模式 4 或 5 时，CMP 端口输出有效电平的时间通过 dmc_hcmp_set_config 函数的 time 参数（脉冲宽度）设置
5）DMC3C00 后四轴不支持高速位置比较功能

short dmc_hcmp_get_mode(WORD CardNo,WORD hcmp, WORD* cmp_mode)
功　能：读取高速比较模式设置
参　数：CardNo　　控制卡卡号
hcmp　　高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）
cmp_mode　返回比较模式设置
返回值：错误代码

short dmc_hcmp_set_config(WORD CardNo,WORD hcmp,WORD axis, WORD cmp_source, WORD cmp_logic,long time)
功　能：配置高速比较器
参　数：CardNo　　控制卡卡号
hcmp　　高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）
axis　　关联轴号，取值范围：DMC3C00：0~7，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3
cmp_source　比较位置源：0：指令位置计数器，1：编码器计数器
cmp_logic　有效电平：0：低电平，1：高电平
time　　脉冲宽度，单位：us，取值范围：1us~20s

返回值：错误代码

注　意：该函数的 time 参数（脉冲宽度）只对队列和线性比较模式起作用

short dmc_hcmp_get_config(WORD CardNo,WORD hcmp,WORD* axis, WORD* cmp_source, WORD* cmp_logic,long* time)

功　能：读取高速比较器配置

参　数：CardNo　　控制卡卡号

　　hcmp　　高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）

　　axis　　返回关联轴号设置

　　cmp_source　　返回比较位置源设置

　　cmp_logic　　返回有效电平设置

　　time　　返回脉冲宽度设置

返回值：错误代码

short dmc_hcmp_clear_points(WORD CardNo,WORD hcmp)

功　能：清除已添加的所有高速位置比较点

参　数：CardNo　　控制卡卡号

　　hcmp　　高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）

返回值：错误代码

short dmc_hcmp_add_point(WORD CardNo,WORD hcmp, long cmp_pos)

功　能：添加/更新高速比较位置

参　数：CardNo　　控制卡卡号

　　hcmp　　高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）

　　cmp_pos　　队列模式下：添加比较位置，单位：pulse

　　线性模式下：更新起始比较位置，单位：pulse

　　其他模式下：更新比较位置，单位：pulse

返回值：错误代码

short dmc_hcmp_set_liner(WORD CardNo,WORD hcmp, long Increment,long Count)

功　能：设置高速比较线性模式参数

参　数：CardNo　　控制卡卡号

hcmp　　高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）
Increment　　位置增量值，单位：pulse（正值表示位置递增，负值表示位置递减）
Count　　比较次数，取值范围：1~32767
返回值：错误代码

short dmc_hcmp_get_liner(WORD CardNo,WORD hcmp, long* Increment,long* Count)
功　能：读取高速比较线性模式参数设置
参　数：CardNo　　控制卡卡号
hcmp　　高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）
Increment　　返回位置增量值设置
Count　　返回比较次数设置
返回值：错误代码

short dmc_hcmp_get_current_state(WORD CardNo,WORD hcmp,long *remained_points,long *current_point,long *runned_points)
功　能：读取高速比较参数
参　数：CardNo　　控制卡卡号
hcmp　　高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）
remained_points　　返回可添加比较点数
current_point　　返回当前比较点位置，单位：pulse
runned_points　　返回已比较点数
返回值：错误代码

short dmc_write_cmp_pin(WORD CardNo,WORD hcmp,WORD on_off)
功　能：控制指定 CMP 端口的输出
参　数：CardNo　　控制卡卡号
hcmp　　高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）
on_off　　设置 CMP 端口电平，0：低电平，1：高电平
返回值：错误代码
注　意：该函数只在 CMP 禁止功能的状态下起作用

short dmc_read_cmp_pin(WORD CardNo,WORD hcmp)

功　能：读取指定 CMP 端口的电平
参　数：CardNo　　控制卡卡号
　　hcmp　　高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）
返回值：CMP 端口电平
队列缓存模式添加比较点
short dmc_hcmp_fifo_set_mode(WORD ConnectNo,WORD hcmp, WORD fifo_mode);
功 能：启用缓存方式添加比较位置
参 数：CardNo　　控制卡卡号 0-7
　　hcmp　　高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）
　　fifo_mode　　是否启用缓存方式，0：不启用，1：启用
返回值：错误代码

short dmc_hcmp_fifo_get_mode(WORD ConnectNo,WORD hcmp, WORD* fifo_mode);
功 能：缓存方式添加比较位置模式回读
参 数：CardNo　　控制卡卡号 0-7
　　hcmp　　高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）
　　fifo_mode　　回读是否启用缓存方式，0：不启用，1：启用
返回值：错误代码

short dmc_hcmp_fifo_get_state(WORD ConnectNo,WORD hcmp,long *remained_points);
功 能：读取剩余缓存状态，上位机通过此函数判断是否继续添加比较位置
参 数：CardNo　　控制卡卡号 0-7
　　hcmp　　高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）
　　remained_points　剩余缓存
返回值：错误代码
备注：实际控制卡最大 25000+127 个缓存点，如果添加 25000 个点，读剩余缓存为 127；

short dmc_hcmp_fifo_add_table(WORD ConnectNo,WORD hcmp, WORD num,double *cmp_pos);
功　能：按数组的方式批量添加比较位置
参　数：CardNo　　控制卡卡号 0-7
　　hcmp　　高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）
　　num　　数据点数，单次可添加 30000 点

cmp_pos 比较位置数组

返回值：错误代码

备注：添加大数据，会堵塞一段时间，直到数据添加完成；

short dmc_hcmp_fifo_clear_points(WORD ConnectNo,WORD hcmp);

功 能：清除比较位置,也会把 FPGA 的位置同步清除掉

参 数：CardNo 控制卡卡号 0-7

hcmp 高速比较器，取值范围：0~3（对应硬件 CMP0~CMP3 端口）

返回值：错误代码

8.20 二维高速位置比较 PWM 输出功能函数

short dmc_hcmp_2d_set_enable(WORD CardNo,WORD 2dhcmp, WORD cmp_enable)

功 能：设置高速比较使能

参 数：CardNo 卡号

2dhcmp 保留，参数值为 0

cmp_enable 二维高速比较器使能，0：禁止，1：使能

返回值：错误代码

short dmc_hcmp_2d_get_enable(WORD CardNo,WORD 2dhcmp, WORD *cmp_enable)

功 能：读取高速比较使能

参 数：CardNo 控制卡卡号

2dhcmp 保留，参数值为 0

cmp_enable 返回二维高速比较器使能

返回值：错误代码

short dmc_hcmp_2d_set_config(WORD CardNo,WORD hcmp,WORD cmp_mode,WORD x_axis, WORD x_cmp_source, WORD y_axis, WORD y_cmp_source, long error,WORD cmp_logic,long time,WORD pwm_enable,double duty,long freq,WORD port_sel,WORD pwm_number);

功 能：配置高速比较器

参 数：CardNo 控制卡卡号

2dhcmp 保留，参数值为 0

cmp_mode	比较模式： 0：进入误差带后触发 1：进入误差带单轴等于后再触发
x_axis	x 轴关联轴号 轴号范围：DMC3400A：0~3，DMC3600：0~6，DMC3800/DMC3C00：0~8
x_cmp_source	x 轴比较位置源：0：指令位置，1：反馈位置
y_axis	y 轴关联轴号 轴号范围：DMC3400A：0~3，DMC3600：0~6，DMC3800/DMC3C00：0~8
y_cmp_source	y 轴比较位置源：0：指令位置，1：反馈位置
Error	x/y 轴误差带设置，单位：pulse
cmp_logic	有效电平：0：低电平，1：高电平
time	脉冲宽度，单位：us，取值范围：1us~20s
pwm_enable	pwm 模式使能
duty	占空比
freq	频率
port_sel	输出口选择
pwm_number	输出的 pwm 脉冲数

返回值：错误代码

注　意：如果 2 次高速比较输入相距很近，则自动重合为一个脉冲。

short dmc_hcmp_2d_get_config(WORD CardNo,WORD hcmp,WORD *cmp_mode,WORD *x_axis, WORD *x_cmp_source, WORD *y_axis, WORD *y_cmp_source, long *error,WORD *cmp_logic,long *time,WORD *pwm_enable,double *duty,long *freq,WORD *port_sel,WORD *pwm_number);

功　能: 读取高速比较器

参　数：

CardNo	控制卡卡号
2dhcmp	保留，参数值为 0
x_axis	返回 x 轴关联轴号
x_cmp_source	返回 x 轴比较位置源：0：指令位置，1：反馈位置
y_axis	返回 y 轴关联轴号
y_cmp_source	返回 y 轴比较位置源：0：指令位置，1：反馈位置
error	返回 x/y 轴误差带设置，单位：pulse
cmp_logic	返回有效电平：0：低电平，1：高电平

time 返回脉冲宽度，单位：us，取值范围：1us~20s
pwm_enable pwm 模式使能
duty 占空比
freq 频率
port_sel 输出口选择
pwm_number 输出的 pwm 脉冲数
返回值：错误代码

short dmc_hcmp_2d_clear_points(WORD CardNo,WORD 2dhcmp)
功 能：清除所有缓冲区高速位置缓冲比较值，并退出当前比较状态
参 数：CardNo 控制卡卡号
2dhcmp 保留，参数值为 0
返回值：错误代码

short dmc_hcmp_2d_add_point(WORD CardNo,WORD 2dhcmp, long x_cmp_pos, long y_cmp_pos)
功 能：添加/更新高速比较位置
参 数：CardNo 控制卡卡号
2dhcmp 保留，参数值为 0
x_cmp_pos 队列模式下：添加 x 比较位置，单位：pulse
y_cmp_pos 队列模式下：添加 x 比较位置，单位：pulse
返回值：错误代码

short dmc_hcmp_2d_get_current_state(WORD CardNo,WORD 2dhcmp, long *remained_points, long* x_current_point, long* y_current_point, long *runned_points, WORD *current_state)
功 能：读取高速比较参数
参 数：CardNo 控制卡卡号
2dhcmp 保留，参数值为 0
remained_points 返回可添加比较点数
x_current_point 返回当前 x 比较点位置
y_current_point 返回当前 y 比较点位置
runned_points 返回已比较点数
current_state 比较器状态 1 正在输出 0 输出完成

返回值：错误代码

short dmc_hcmp_2d_force_output(WORD CardNo,WORD hcmp,WORD enable);

功　能：该函数用于强制二维比较输出，输出按照配置好的脉冲模式或者 pwm 模式

参　数：CardNo　　控制卡卡号

2dhcmp　　保留，参数值为 0

Force_output_en　　强制二维比较输出，设置为 1 使能

返回值：错误代码

8.21　异常信号接口函数

short dmc_set_emg_mode(WORD CardNo,WORD axis,WORD enable,WORD emg_logic)

功　能：设置 EMG 急停信号

参　数：CardNo　　控制卡卡号

axis　　DMC3C00/3800 卡：指定轴号，取值范围：0~7

DMC3600 卡：指定轴号，取值范围：0~5

DMC3400A:指定轴号，取值范围：0~3

enable　　允许/禁止信号功能，0：禁止，1：允许

emg_logic　　EMG 信号有效电平，0：低有效，1：高有效

返回值：错误代码

注　意：DMC3C00 后四轴不支持异常停止功能。

short dmc_get_emg_mode (WORD CardNo, WORD axis,WORD *enbale,WORD * logic)

功　能：读取 EMG 急停信号设置

参　数：CardNo　　控制卡卡号

axis　　DMC3C00/3800 卡：指定轴号，取值范围：0~7

DMC3600 卡：指定轴号，取值范围：0~5

DMC3400A:指定轴号，取值范围：0~3

enable　　返回 EMG 信号功能状态

logic　　返回设置的 EMG 信号有效电平

返回值：错误代码

short dmc_set_io_dstp_mode(WORD CardNo,WORD axis,WORD enable,WORD logic)

功　能：设置减速停止信号
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00/3800：0~7，DMC3600：0~5,DMC3400A：　0~3
　　enable　　允许/禁止硬件信号功能，0：禁止，1：允许
　　logic　　外部减速停止信号有效电平，0：低有效，1：高有效
返回值：错误代码
注　意：减速停止信号（DSTP）的减速时间由函数 dmc_set_dec_stop_time 设置

short dmc_get_io_dstp_mode(WORD CardNo,WORD axis,WORD *enable,WORD *logic)
功　能：读取减速停止信号设置
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00/3800：0~7，DMC3600：0~5,DMC3400A：　0~3
　　enable　　返回 DSTP 硬件信号功能状态
　　logic　　返回设置的外部减速停止信号有效电平
返回值：错误代码

short dmc_set_dec_stop_time(WORD CardNo,WORD axis,double stop_time)
功　能：设置减速停止时间
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00/3800：0~7
　　DMC3600：0~5,DMC3400A：0~3
　　stop_time　　减速时间，单位：s
返回值：错误代码
注　意：当发生异常停止时，如：调用 dmc_stop 函数、限位信号（软硬件）被触发、减速停止信号(DSTP)被触发等进行减速停止时，减速停止时间都为 dmc_set_dec_stop_time 函数里设置的减速时间

short dmc_get_dec_stop_time(WORD CardNo,WORD axis,double *stop_time)
功　能：读取减速停止时间设置
参　数：CardNo　　控制卡卡号
　　axis　　指定轴号，取值范围：DMC3C00/3800：0~7
　　DMC3600：0~5,DMC3400A：　0~3

stop_time　　返回设置的减速时间，单位：s

返回值：错误代码

8.22　轴 IO 映射函数

short　dmc_set_axis_io_map(WORD　CardNo,WORD　Axis,WORD　IoType,WORD　MapIoType,WORD MapIoIndex,double filter_time)

功　能：设置轴 IO 映射关系

参　数：CardNo　　控制卡卡号

Axis　　指定轴号，取值范围：DMC3C00/DMC3800：0~7，DMC3600：0~5，　DMC3400A：0~3

IoType　　指定轴的 IO 信号类型：

0：正限位信号，AxisIoInMsg_PEL

1：负限位信号，AxisIoInMsg_NEL

2：原点信号，AxisIoInMsg_ORG

3：急停信号，AxisIoInMsg_EMG

4：减速停止信号，AxisIoInMsg_DSTP

5：伺服报警信号，AxisIoInMsg_ALM

6：伺服准备信号，AxisIoInMsg_RDY（保留）

7：伺服到位信号，AxisIoInMsg_INP

MapIoType　　轴 IO 映射类型：

0：正限位输入端口，AxisIoInPort_PEL

1：负限位输入端口，AxisIoInPort_NEL

2：原点输入端口，AxisIoInPort_ORG

3：伺服报警输入端口，AxisIoInPort_ALM

4：伺服准备输入端口，AxisIoInPort_RDY

5：伺服到位输入端口，AxisIoInPort_INP

6：通用输入端口，AxisIoInPort_IO

MapIoIndex　　轴 IO 映射索引号：

1）当轴 IO 映射类型设置为 6 时，此参数可设置为 0~15 整数，表示该映射对应的具体通用输入端口号

2）当轴 IO 映射类型设置为 0~5 时，此参数可设置 0~7 整数，表示该

映射所对应的具体轴号

filter_time　　轴 IO 信号滤波时间，单位：s

返回值：错误代码

注　意：1）该函数可以实现对专用 IO 信号的硬件输入接口进行任意配置

2）dmc_set_axis_io_map 函数用法详见 7.14 节 轴 IO 映射功能的实现

3）DMC3C00 后四轴不支持 IO 映射功能

short dmc_get_axis_io_map(WORD CardNo,WORD Axis,WORD IoType,WORD* MapIoType,WORD* MapIoIndex,double* filter_time)

功　能：读取轴 IO 映射关系设置

参　数：CardNo　　控制卡卡号

Axis　　指定轴号，取值范围：DMC3C00/3800：0~7、DMC3600：0~5，DMC3400A：0~3

IoType　　轴 IO 信号类型

MapIoType　　返回轴 IO 映射类型

MapIoIndex　　返回轴 IO 映射索引号

filter_time　　返回轴 IO 信号滤波时间，单位：s

返回值：错误代码

short dmc_set_special_input_filter(WORD CardNo,double filter_time)

功　能：统一设置所有专用 IO 的滤波时间

参　数：CardNo　　控制卡卡号

filter_time　　轴 IO 信号滤波时间，单位：s

返回值：错误代码

8.23　虚拟 IO 映射函数

short dmc_set_io_map_virtual(WORD CardNo,WORD bitno,WORD MapIoType,WORD MapIoIndex,double filter_time)

功　能：设置虚拟 IO 映射关系

参　数：CardNo　　控制卡卡号

bitno　　虚拟 IO 口号,取值范围：0~15

MapIoType　　虚拟 IO 映射类型：
0：正限位输入端口，AxisIoInPort_PEL
1：负限位输入端口，AxisIoInPort_NEL
2：原点输入端口，AxisIoInPort_ORG
3：伺服报警输入端口，AxisIoInPort_ALM
4：伺服准备输入端口，AxisIoInPort_RDY
5：伺服到位输入端口，AxisIoInPort_INP
6：通用输入端口，AxisIoInPort_IO
7：急停信号输入端口，AxisIoInPort_EMG
8：通用输出端口

MapIoIndex　　虚拟 IO 映射索引号：
1）当虚拟 IO 映射类型设置为 6 时，此参数可设置为 0~15 整数，表示该映射对应的具体通用输入端口号
2）当虚拟 IO 映射类型设置为 0~5 时，此参数可设置 0~11 整数，表示该映射所对应的具体轴号
3）取消虚拟 IO 映射关系，设置该值为 255；

filter_time　　虚拟 IO 信号滤波时间，单位：s

返回值：错误代码

注　意：1）该函数可以实现专用通用 IO 输入接口的滤波功能
2）dmc_set_io_map_virtual 函数用法详见 7.16 节 虚拟 IO 映射功能的实现

short dmc_get_io_map_virtual(WORD CardNo,WORD bitno,WORD* MapIoType,WORD* MapIoIndex,double* filter_time)

功　能：读取虚拟 IO 映射关系设置

参　数：CardNo　　控制卡卡号
bitno　　虚拟 IO 口号，取值范围：0~15
MapIoType　　返回虚拟 IO 映射类型
MapIoIndex　　返回虚拟 IO 映射索引号
filter_time　　返回虚拟 IO 信号滤波时间，单位：s

返回值：错误代码

short dmc_read_inbit_virtual(WORD CardNo,WORD bitno)

功　能：读取滤波后的虚拟 IO 口电平状态

参　数：CardNo　　控制卡卡号

　　bitno　　虚拟 IO 口号，取值范围：0~15

返回值：指定的虚拟 IO 口电平：0：低电平，1：高电平

注　意：1）此函数需要配合虚拟 IO 映射功能使用

2）dmc_read_inbit_virtual 与 dmc_read_inbit 函数的区别是：dmc_read_inbit 函数是不经过滤波直接读取硬件端口的电平状态，dmc_read_inbit_virtual 函数通过虚拟 IO 映射后读取相应端口滤波后的电平状态

3）dmc_read_inbit_virtual 函数用法详见 7.16 节 虚拟 IO 映射功能的实现

8.24　检测轴到位状态函数

short dmc_set_factor_error(WORD CardNo,WORD axis,double factor,long error)

功　能：设置位置误差带

参　数：CardNo　　控制卡卡号

　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

　　factor　　编码器系数

　　error　　位置误差带，单位：pulse

返回值：错误代码

编码器系数的说明：当使用 dmc_check_success_encoder 函数检测编码器是否到位时，其用于判断的编码器位置为：**编码器计数值**乘以**编码器系数**的值。关于检测轴到位状态函数的用法详见章节 7.10 检测轴到位状态功能的实现

short dmc_get_factor_error(WORD CardNo,WORD axis,double* factor,long* error)

功　能：读取位置误差带设置

参　数：CardNo　　控制卡卡号

　　axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

　　factor　　返回编码器系数设置

　　error　　返回位置误差带设置

返回值：错误代码

short dmc_check_success_pulse(WORD CardNo,WORD axis)

功　能：检测指令到位

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

返回值：0：表示指令位置在设定的目标位置的误差带之外

1：表示指令位置在设定的目标位置的误差带之内

注　意：1）该函数只适用于单轴运动

2）检测函数请在 dmc_check_done 检测到轴停止后调用，函数调用后会等待轴到位后返回，如果调用函数 100ms 内未到位，函数超时返回认为不到位

short dmc_check_success_encoder(WORD CardNo,WORD axis)

功　能：检测编码器到位

参　数：CardNo　　控制卡卡号

axis　　指定轴号，取值范围：DMC3C00：0~11，DMC3800：0~7，DMC3600：0~5
DMC3400A：0~3

返回值：0：表示编码器位置在设定的目标位置的误差带之外

1：表示编码器位置在设定的目标位置的误差带之内

注　意：1）该函数只适用于单轴运动

2）检测函数请在 dmc_check_done 检测到轴停止后调用，函数调用后会等待轴到位后返回，如果调用函数 100ms 内未到位，函数超时返回认为不到位

8.25　CAN 扩展函数

short nmc_set_connect_state(WORD CardNo,WORD NodeNum,WORD state,WORD Baud)

功　能：设置 CAN 通讯状态

参　数：CardNo　　控制卡卡号

NodeNum　　CAN 节点数，取值范围：1~8

state　　设置通讯状态，0：断开，1：连接

Baud　　设置控制卡波特率

波特率参数：0,1,2,3,4,5

对应波特率：1000Kbps，800Kbps,500Kbps,250Kbps,125Kbps,100Kbps

返回值：错误代码

注　意：1）当关闭运动控制卡时，CAN 通讯不会被自动断开；当再次初始化运动控制卡时，CAN-IO 通讯依然保持之前的状态；

2）当连接 CAN 通讯时，必须使用 nmc_get_can_state 函数读取 CAN-IO 的通讯状态，确认 CAN 通讯已正常连接。当连接出现异常时，可再次调用 nmc_set_can_state 函数进行连接；

3）设置波特率时需保证控制卡波特率与模块波特率相对应；

short nmc_get_connect_state(WORD CardNo,WORD* NodeNum,WORD* state)

功　能：读取 CAN 通讯状态

参　数：CardNo　　控制卡卡号

NodeNum　　返回 CAN 节点数

state　　返回 CAN-IO 通讯状态，0：断开，1：连接，2：异常

返回值：错误代码

short nmc_write_outbit(WORD CardNo,WORD NodeID,WORD IoBit,WORD IoValue)

功　能：设置指定 CAN-IO 扩展模块的某个输出端口的电平

参　数：CardNo　　控制卡卡号

Node　　CAN 节点号，取值范围：1~8

bitno　　输出口号

on_off　　输出电平，0：低电平，1：高电平

返回值：错误代码

short nmc_read_outbit(WORD CardNo,WORD NodeID,WORD IoBit,WORD* IoValue)

功　能：读取指定 CAN-IO 扩展模块的某个输出端口的电平

参　数：CardNo　　控制卡卡号

Node　　CAN 节点号，取值范围：1~8

IoBit　　输出口号

IoValue　　指定输出端口的电平，0：低电平，1：高电平

返回值：错误代码

short nmc_read_inbit(WORD CardNo, WORD NoteID, WORD IoBit,WORD* IoValue)

功　能：读取指定 CAN-IO 扩展模块的某个输入端口的电平
参　数：CardNo　　控制卡卡号
　　NoteID　　CAN 节点号，取值范围：1~8
　　IoBit　　输入端口号
　　IoValue　　指定输入端口的电平，0：低电平，1：高电平
返回值：错误代码

short nmc_write_outport(WORD CardNo, WORD Node, WORD PortNo, DWORD outport_val)
功　能：设置指定 CAN-IO 扩展模块的输出端口的电平
参　数：CardNo　　控制卡卡号
　　Node　　CAN 节点号，取值范围：1~8
　　PortNo　　端口号，0 表示 0~31 号口，1 表示 32~63 号口
　　outport_val　输出值，bit0~bit32 的值分别代表第 0~32 号输出口的电平
返回值：错误代码

short nmc_read_outport(WORD CardNo, WORD Node, WORD PortNo, DWORD* outport_val)
功　能：读取指定 CAN-IO 扩展模块的输出端口的电平
参　数：CardNo　　控制卡卡号
　　Node　　CAN 节点号，取值范围：1~8
　　PortNo　　端口号
　　outport_val　输出端口的值，bit0~bit31 的值分别代表第 0~31 号输出口的电平
返回值：错误码

short nmc_read_inport(WORD CardNo, WORD Node, WORD PortNo, DWORD* inport_val)
功　能：读取指定 CAN-IO 扩展模块的输入端口的电平
参　数：CardNo　　控制卡卡号
　　Node　　CAN 节点号，取值范围：1~8
　　PortNo　　端口号
　　inport_val　输入端口的值，bit0~bit31 的值分别代表第 0~31 号输入口的电平
返回值：错误码

short nmc_set_da_output(WORD CardNo, WORD NoteID, WORD channel,double Value)

功　能：设置指定 CAN-ADDA 扩展模块的某个输出端口的电平
参　数：CardNo　　控制卡卡号
　　NoteID　　CAN 节点号，取值范围：1~8
　　channel　　输出端口号
　　Value　　输出电平，单位：mV/mA
返回值：错误代码

short nmc_get_da_output(WORD CardNo, WORD NoteID, WORD channel,double* Value)
功　能：读取指定 CAN-ADDA 扩展模块的某个输出端口的电平
参　数：CardNo　　控制卡卡号
　　NoteID　　CAN 节点号，取值范围：1~8
　　channel　　输出端口号
　　Value　　输出端口的电平，单位：mV/mA
返回值：错误代码

short nmc_get_ad_input(WORD CardNo, WORD NoteID, WORD channel,double* Value)
功　能：读取指定 CAN-ADDA 扩展模块的某个输入端口的电平
参　数：CardNo　　控制卡卡号
　　NoteID　　CAN 节点号，取值范围：1~8
　　channel　　输入端口号
　　Value　　输入端口的电平，单位：mV/mA
返回值：错误代码

short nmc_set_ad_mode(WORD CardNo,WORD NoteID,WORD channel,WORD mode,DWORD buffer_nums)
功　能：设置指定 CAN-ADDA 扩展模块的某个输入端口的模式
参　数：CardNo　　控制卡卡号
　　NoteID　　CAN 节点号，取值范围：1~8
　　channel　　输入端口号
　　mode　　输入模式，0：电压模式，1：电流模式
　　buffer_nums　　保留参数
返回值：错误代码

short nmc_get_ad_mode(WORD CardNo,WORD NoteID,WORD channel,WORD* mode,DWORD buffer_nums)

功　能：读取指定 CAN-ADDA 扩展模块的某个输入端口的模式

参　数：CardNo　　控制卡卡号

NoteID　　CAN 节点号，取值范围：1~8

channel　　输入端口号

mode　　输入模式，0：电压模式，1：电流模式

buffer_nums　　保留参数

返回值：错误代码

short nmc_set_da_mode(WORD CardNo,WORD NoteID,WORD channel,WORD mode,DWORD buffer_nums)

功　能：设置指定 CAN-ADDA 扩展模块的某个输出端口的模式

参　数：CardNo　　控制卡卡号

NoteID　　CAN 节点号，取值范围：1~8

channel　　输出端口号

mode　　输出模式，0：电压模式，1：电流模式

buffer_nums　　保留参数

返回值：错误代码

short nmc_get_da_mode(WORD CardNo,WORD NoteID,WORD channel,WORD* mode,DWORD buffer_nums)

功　能：读取指定 CAN-ADDA 扩展模块的某个输出端口的模式

参　数：CardNo　　控制卡卡号

NoteID　　CAN 节点号，取值范围：1~8

channel　　输出端口号

mode　　输出模式，0：电压模式，1：电流模式

buffer_nums　　保留参数

返回值：错误代码

short nmc_write_to_flash (WORD CardNo,WORD PortNum,WORD NodeNum)

功　能：保存模式设置到模块 FLASH

参　数：CardNo　　控制卡卡号

PortNum　　保留参数，设为 0
NodeNum　　节点号，1-8

返回值：错误代码

注　意：1）保存后模块会断开连接，需要重新连接才能进行正常控制。
2）设置的模式会断电保存，上电后电压或电流模式为最后一次断电前设置的模式。

8.26　AD/DA 功能

short dmc_set_da_enable(WORD CardNo,WORD enable);
功　能：设置 DA 输出使能
参　数：CardNo　　控制卡卡号
WORD enable　DA 使能状态，0：禁止，1：使能
返回值：错误代码

short dmc_get_da_enable(WORD CardNo,WORD* enable);
功　能：读取 DA 输出使能设置
参　数：CardNo　　控制卡卡号
WORD enable　DA 使能状态，0：禁止，1：使能
返回值：错误代码

short dmc_set_da_output(WORD CardNo, WORD channel,double Vout);
功　能：设置 DA 输出
参　数：CardNo　　控制卡卡号
Channel　　DA 输出口号，取值范围 0、1
Vout　　DA 输出电压，输出电压范围 -10V ~ 10V
返回值：错误代码

short dmc_get_da_output(WORD CardNo, WORD channel,double* Vout);
功　能：读取 DA 输出
参　数：CardNo　　控制卡卡号
Channel　　DA 输出口号，取值范围 0、1
Vout　　DA 输出电压，输出电压范围 -10V ~ 10V

返回值：错误代码

short dmc_get_ad_input(WORD CardNo,WORD channel,double* Vout);

功　能：读取 AD 输入

参　数：CardNo　　控制卡卡号

　　Channel　　AD 输入口号，取值范围 0 ~ 7

　　Vout　　AD 输入电压，输入电压范围 -10V ~ 10V

返回值：错误代码

8.27　螺距补偿函数

short dmc_enable_leadscrew_comp(WORD CardNo, WORD axis,WORD enable);

功　能：使能螺距补偿功能

参　数：CardNo　　控制卡卡号

　　axis　　轴号

　　enable　　使能螺距补偿功能，1：使能，0：不使能

返回值：错误代码

short dmc_set_leadscrew_comp_config(WORD CardNo, WORD axis,WORD n, int startpos,int lenpos,int *pCompPos,int *pCompNeg);

功　能：设置螺距补偿配置参数

参　数：CardNo　　控制卡卡号

　　axis　　轴号

　　n　　补偿点数

　　startpos　　补偿起始点的规划位置（单位：pulse）

　　lenPos　　测量段的总长（单位：pulse);

　　pCompPos　　对应为正方向运动时，各点位置需要补偿的脉冲数;

　　pCompNeg　　对应为负方向运动时，各点位置需要补偿的脉冲数;

返回值：错误代码

注　意：边界点是不补偿的;

8.28 圆弧区域限位功能

运动仅在用户设定的圆形范围内进行，超出这个范围，运动会依据设定的减速模式停止，且禁止任何运动操作；

short dmc_set_arc_zone_limit_config(WORD CardNo, WORD* AxisList, WORD AxisNum, double *Center, double Radius, WORD Source, WORD StopMode);

功　能：设置圆弧区域限位配置信息

参　数：CardNo: 卡号

AxisList: 需要限位的轴列表

AxisNum：需要限位的轴个数，固定为 2

Center：区域限位圆心

Radius：区域限位半径

Source：计数器选择 0 脉冲，1 反馈

StopMode: 限位触发后停止模式 0 减速停；1 急停

返回值：错误代码

short dmc_get_arc_zone_limit_config(WORD CardNo, WORD* AxisList, WORD* AxisNum, double *Center, double * Radius, WORD* Source, WORD* StopMode);

功　能：读取圆弧区域限位配置信息

参　数：CardNo: 卡号

AxisList: 需要限位的轴列表

AxisNum：需要限位的轴个数，固定为 2

Center：区域限位圆心

Radius：区域限位半径

Source：计数器选择 0 脉冲，1 反馈

StopMode: 限位触发后停止模式 0 减速停；1 急停

返回值：错误代码

short dmc_get _arc_zone_limit_axis_status(WORD CardNo, WORD AxisNo);

功　能：查询相应轴的状态

参　数：CardNo：卡号
　　AxisNo：轴号
返回值：-1 该轴没有被设置区域限位；0 该轴在区域内；1 该轴触发区域限位

short dmc_arc_zone_limit_enable(WORD CardNo, WORD enable);
功　能：配置圆弧限位功能使能状态
参　数：CardNo：卡号
　　enable：使能状态，0 未使能；1 使能
返回值：错误代码

short dmc_get_arc_zone_limit_enable(WORD CardNo, WORD* enable);
功　能：读取圆弧限位功能使能状态
参　数：CardNo：卡号
　　enable：使能状态，0 未使能；1 使能
返回值：错误代码
注　意：1. 由于触发圆弧区域限位运动停止，获取停止原因返回值为 108；
　　2. 轴在运动中，不可设置限位参数；
　　3. 可通过重复调用 dmc_get_arc_zone_limit_config()函数来修改参数；
　　4. 除 VMOVE 外，其它运动形式若开始运动时在限位区域外，则禁止运动；
　　5. 使用前需先配置，再使能。

8.29　密码管理函数

short　dmc_enter_password_ex(WORD CardNo, const char* spass);
功　能：密码登录
参　数：CardNo：卡号
　　spass：　旧密码
返回值：错误代码
注　意：修改旧密码前需要调用此密码进行验证，验证成功后才可调用写密码函数写入新的密码，3XX201621 版本固件及更高版本支持；

功能使用：在板卡为出厂状态时，可直接调用 dmc_write_sn 来首次写入密码，后续要使用该函数修改密码时，需要先调用 dmc_enter_password_ex 进行旧密码验证，验证成功后才能写入新的密码。密码验证失败连续超过 5 次，不能继续验证，卡断电后，才可重新验证。

short dmc_write_sn(WORD CardNo,const char* new_sn)
功　能：修改密码
参　数：CardNo　　控制卡卡号
　　new_sn　　新密码，密码长度不大于 255 个字符
返回值：错误代码

short dmc_check_sn(WORD CardNo,const char* check_sn)
功　能：密码校验
参　数：CardNo　　控制卡卡号
　　check_sn　　旧密码
返回值：校验状态，0：失败，1：成功
注　意：1）密码校验连续失败 5 次后，无法再次校验
　　2）用户可以在系统软件开启时加入密码校验动作，以此对系统软件进行加密

8.30　打印输出函数

short dmc_set_debug_mode(WORD mode,const char *FileName)
功　能：函数调用打印输出设置
参　数：mode　　打印输出模式，0：只打印报错函数，1：全部打印，2：全部不打印
　　FileName　　文件保存路径：
　　参数文件名+后缀：相对路径
　　完整描述参数文件的路径+文件名后缀：绝对路径
返回值：错误代码
说　明：使能打印输出后，可监控运动函数库的调用情况。在用户调用函数时，将输出相关信息，并保存在指定文件路径中；**函数设置模式 2 全部不打印需配合最新动态库（20181030 及以后动态库）使用。**

short dmc_get_debug_mode(WORD mode,char *FileName)

功　能：读取函数调用打印输出设置

参　数：mode　　返回打印输出使能状态

　　FileName　　返回文件保存路径

返回值：错误代码

8.31　运动函数错误码说明

DMC3000 系列运动控制卡的运动函数错误码见表 8.7。

表 8.7　运动函数错误码

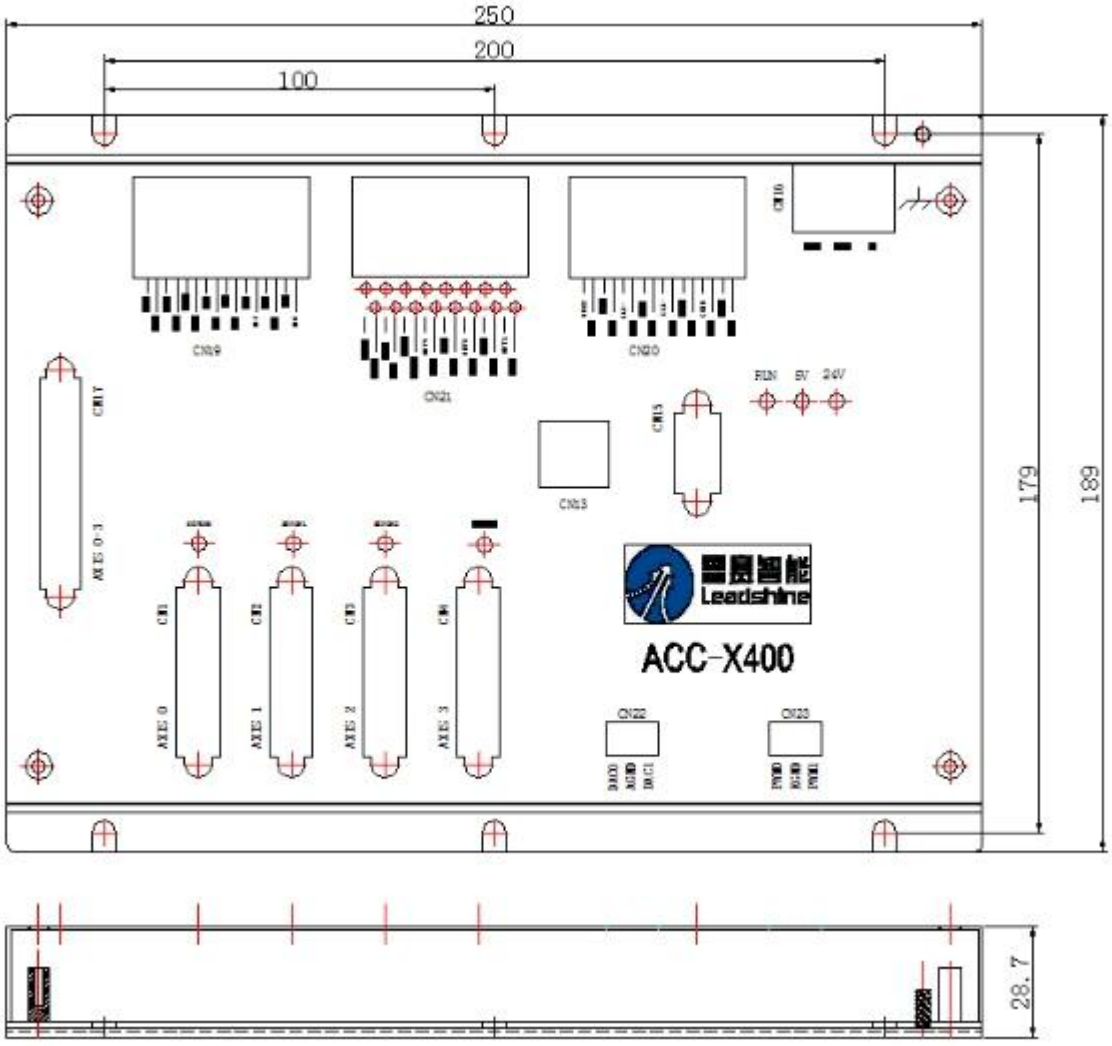
错误码	名称	含义
0	ERR_NOERR	成功
1	ERR_UNKNOWN	未知错误
2	ERR_PARAERR	参数错误
3	ERR_TIMEOUT	操作超时
4	ERR_CONTROLLERBUSY	控制卡状态忙
6	ERR_CONTILINE	连续插补错误
8	ERR_CANNOT_CONNECTETH	无法连接错误
9	MOTION_ERR_HANDLEERR	卡号错误，可能由于函数参数中的卡号或轴号超出范围产生，比如有两张卡分别为 0 号和 1 号卡，当轴号为 8 时根据计算应该为 2 号卡，此时就会产生错误。
10	ERR_SENDERR	数据传输错误， 请检查 PCI 槽位是否松动
12	ERR_FIRMWAREERR	固件文件错误
14	ERR_FIRMWAR_MISMATCH	固件不匹配
20	ERR_FIRMWARE_INVALID_PARA_ERR	固件参数错误
22	ERR_FIRMWARE_STATE_ERR	固件当前状态不允许操作
24	ERR_FIRMWARE_CARD_NOT_SUPPORT	不支持的功能
1967		旧密码输入错误
1968		旧密码验证次数超限，不能继续验证
1969		拒绝写入新密码

附　录

附录 1　ACC-X400 接线盒接口说明

ACC-X400 接线盒是 DMC3400A 运动控制卡的配套产品，扩展 DMC3400A 控制卡的所有用户端子。

ACC-X400 接线盒安装尺寸图如图 F4.1 所示：



F4.1　ACC-X400 接线盒安装尺寸图

ACC-X400 接线盒外形结构图如图 F4.2 所示：

F4.1 ACC-X400 接线盒外形结构图

1、 ACC-X400 接线盒接口及脚位

表 F4.1 ACC-X400 接线盒接口功能简述

名称	功能介绍
CN1~CN4	第 1~4 轴轴控制信号
CN19	数字量通用输入端子
CN21	数字量输出端子
CN20	数字量专用输入端子
CN13	CAN 总线接口（RJ45）
CN15	辅助编码器接口
CN16	24V 直流电源输入端子
CN17	运动控制卡连接端子（1~4 轴）

2、 CN1~CN4 轴控制端子信号定义

1）ACC-X400 接线盒 CN1 至 CN4 为电机控制信号端口，采用 DB25 母头，其引脚号和信号对应关系见表 F4.2 所示。其中 24V 电源端口主要为控制卡的电机控制信号（伺服专用信号）供电。

2）电机控制信号端（CN1 至 CN4）的 24V 为控制信号电源，不能用于驱动器等动力负载供电。

3）根据 ACC-X400 接线盒的 PCB 板的铜线宽度与厚度，4 个电机控制信号端口的 24V 总输出电流需要控制在 1.5A 以内。

表 F4.2 接口 CN1~CN4 引脚号和信号关系表

序	名称	I/O	说 明	序	名称	I/O	说 明
1	OGND	O	24V 电源地	14	24V	O	+24V 输出
2	ALM	I	驱动报警	15	ERC	O	驱动报警复位
3	SEVON	O	驱动允许	16	INP	I	到位信号
4	EA-	I	编码器输入	17	EA+	I	编码器输入
5	EB-	I	编码器输入	18	EB+	I	编码器输入
6	EZ-	I	编码器输入	19	EZ+	I	编码器输入
7	+5V	O	内部 5V	20	GND	O	内部 5V 地
8	保留	-	保留	21	GND	O	内部 5V 地
9	DIR+	O	方向输出	22	DIR-	O	方向输出
10	GND	O	内部 5V 地	23	PUL+	O	脉冲输出
11	PUL-	O	脉冲输出	24	GND	O	内部 5V 地
12	RDY	I	伺服准备完成	25	保留		保留
13	GND	O	内部 5V 地				

注意：（1）当使用+5V 和 PUL-端口时，则选择电机指令脉冲信号输出方式为单端输出；当使用 PUL+和 PUL-端口时，则选择电机指令脉冲信号输出方式为差分输出。

3、 IO 扩展接口定义

CN13 为 IO 扩展接口，采用 RJ45 接口，可连接 CAN-IO 扩展模块。

4、 CN15 辅助编码器/手轮接口定义

CN15 为辅助编码器/手轮接口，采用 DB9 母头。其引脚号和信号对应关系见表 F4.3 所示。

表 F4.3 接口 CN15 引脚号和信号关系表

序	名称	I/O	说 明
1	EA+	I	编码器/手轮输入

2	EB+	I	编码器/手轮输入
3	EZ+	I	编码器输入
4	保留	-	NC
5	+5V	O	内部 5V 输出
6	EA-	I	编码器/手轮输入
7	EB-	I	编码器/手轮输入
8	EZ-	I	编码器输入
9	GND	O	内部 5V 地

5、 CN16 电源定义

接口 CN16 是接线盒的电源输入接口，板上标有 24V 的端子接+24，标有 OGND 的端子接外部电源地。

6、 CN17 接口定义

CN17 接口为 0~3 轴电机的控制信号、IO 信号及低速手轮通道等与控制卡的接口。

7、 CN19 数字量输入端子定义

CN19 为数字量输入接口，其引脚号和信号对应关系见表 4.4 所示：

F4.4　接口 CN9 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	IN0	I	通用输入（低速）	9	IN8	I	通用输入（低速）
2	IN1	I	通用输入（低速）	10	IN9	I	通用输入（低速）
3	IN2	I	通用输入（低速）	11	IN10	I	通用输入（低速）
4	IN3	I	通用输入（低速）	12	IN11	I	通用输入（低速）
5	IN4	I	通用输入（低速）	13	IN12	I	通用输入（低速）
6	IN5	I	通用输入（低速）	14	IN13	I	通用输入（低速）
7	IN6	I	通用输入（低速）	15	IN14	I	通用输入（LTC0 高速）
8	IN7	I	通用输入（低速）	16	IN15	I	通用输入（LTC1 高速）

8、 CN20 数字量输入接口定义

CN20 为数字量输入接口，其引脚号和信号对应关系见表 F4.5 所示。

表 F4.5　接口 CN20 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	OGND	O	24V 电源地	9	EL2+	I	2 轴正限位
2	OGND	O	24V 电源地	10	EL2-	I	2 轴负限位
3	OGND	O	24V 电源地	11	EL3+	I	3 轴正限位
4	OGND	O	24V 电源地	12	EL3-	I	3 轴负限位
5	EL0+	I	0 轴正限位	13	ORG0	I	0 轴原点输入
6	EL0-	I	0 轴负限位	14	ORG1	I	1 轴原点输入
7	EL1+	I	1 轴正限位	15	ORG2	I	2 轴原点输入
8	EL1-	I	1 轴负限位	16	ORG3	I	3 轴原点输入

9、 CN21 数字量输出接口定义

CN21 为数字量输出接口，其引脚号和信号对应关系见表 F4.6 所示。

表 F4.6　接口 CN21 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	OUT0	O	通用输出（低速）	9	OUT8	O	通用输出（低速）
2	OUT1	O	通用输出（低速）	10	OUT9	O	通用输出（低速）
3	OUT2	O	通用输出（低速）	11	OUT10	O	通用输出（低速）
4	OUT3	O	通用输出（低速）	12	OUT11	O	通用输出（低速）
5	OUT4	O	通用输出（低速）	13	OUT12	O	保留
6	OUT5	O	通用输出（低速）	14	OUT13	O	保留
7	OUT6	O	通用输出（低速）	15	OUT14	O	通用输出/CMP2（高速）
8	OUT7	O	通用输出（低速）	16	OUT15	O	通用输出/CMP3（高速）

10、 CN22 模拟量输出接口定义

DMC3400A 提供了两路 12 位 0~10V DA 输出口，CN22 为模拟量输出接口，其引脚号和信号对应关系如表 F4.7 所示。

表 F4.7　接口 CN22 引脚号和信号关系表

序	名称	I/O	说　明
1	DAC0	O	模拟量输出接口 1，输出电压范围 0~10V
2	AGND	O	模拟地
3	DAC1	O	模拟量输出接口 1，输出电压范围 0~10V

11、 CN23 PWM 输出接口定义

CN23 为 PWM 输出接口，其引脚号和信号对应关系如表 F4.8 所示

F4.8 接口 CN23 引脚号和信号关系表

序	名称	I/O	说 明
1	PWM0	O	PWM(CMP0)输出 0
2	EGND	O	24V 电源地
3	PWM1	O	PWM(CMP1)输出 1

12、 指示灯定义

ACCX400 接线盒表面有 3 个指示灯，分别为：

24VLED：外部电源指示灯；

5VLED：内部电源指示灯；

RUNLED：连接状态指示灯。绿色时表示接线盒与控制卡处于通讯状态；红色闪烁时表示接线盒与控制卡未连接成功。

附录 2　ACC-X400B 接线盒接口说明

ACC-X400B 接线盒是 DMC3400A 运动控制卡的配套产品，扩展 DMC3400A 控制卡的所有用户端子。

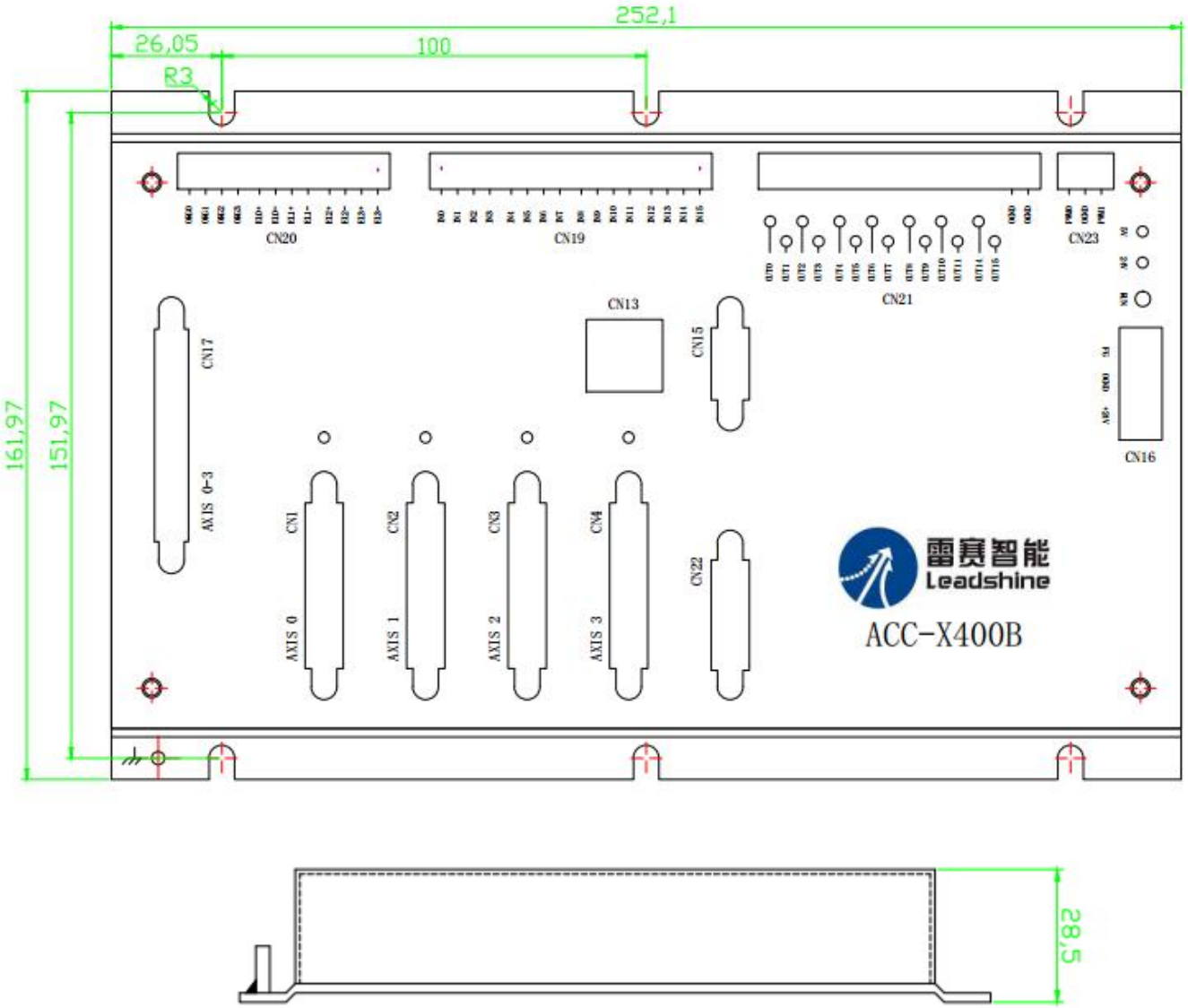


图 2.1　ACC-X400B 接线盒尺寸图

ACC-X400B 接线盒外形结构图如图 2.2 所示：

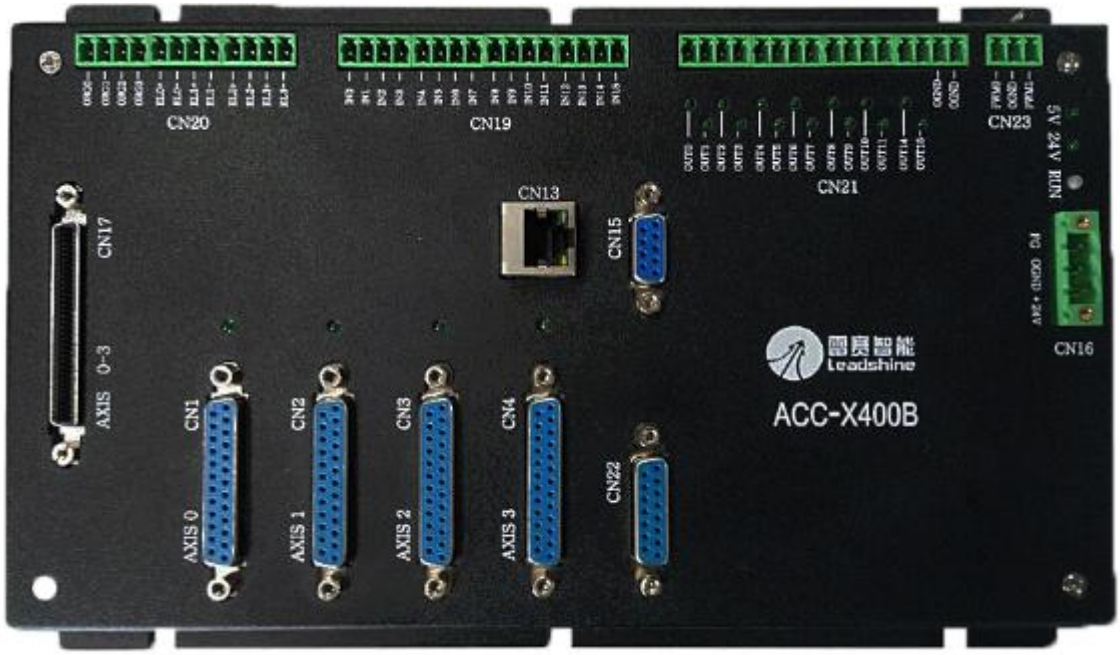


图 2.2　ACC-X400B 接线盒外形结构图

1、 ACC-X400B 接线盒接口及脚位

表 2.1　ACC-X400B 接线盒接口功能简述

名称	功能介绍
CN1~CN4	第 1~4 轴轴控制信号
CN13	CAN 总线接口（RJ45）
CN15	辅助编码器接口
CN16	24V 直流电源输入端子
CN17	运动控制卡连接端子（1~4 轴）
CN19	数字量通用输入端子
CN20	数字量专用输入端子
CN21	数字量输出端子
CN23	PWM 输出端子

2、 CN1~CN4 轴控制端子信号定义

1）ACC-X400B 接线盒 CN1 至 CN4 为电机控制信号端口，采用 DB25 母头，其引脚号和信号对应关系见表 2.2 所示。其中 24V 电源端口主要为控制卡的电机控制信号（伺服专用信号）供电。

2）电机控制信号端（CN1 至 CN4）的 24V 为控制信号电源，不能用于驱动器等动力负载供电。

3）根据 ACC-X400B 接线盒的 PCB 板的铜线宽度与厚度，4 个电机控制信号端口的 24V 总输出电流需要控制在 1.5A 以内。

表 2.2　接口 CN1~CN8 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	OGND	O	24V 电源地	14	24V	O	+24V 输出
2	ALM	I	驱动报警	15	ERC	O	驱动报警复位
3	SEVON	O	驱动允许	16	INP	I	到位信号
4	EA-	I	编码器输入	17	EA+	I	编码器输入
5	EB-	I	编码器输入	18	EB+	I	编码器输入
6	EZ-	I	编码器输入	19	EZ+	I	编码器输入
7	+5V	O	内部 5V	20	GND	O	内部 5V 地
8	保留	-	保留	21	GND	O	内部 5V 地
9	DIR+	O	方向输出	22	DIR-	O	方向输出
10	GND	O	内部 5V 地	23	PUL+	O	脉冲输出
11	PUL-	O	脉冲输出	24	GND	O	内部 5V 地
12	RDY	I	伺服准备完成	25	保留		保留
13	GND	O	内部 5V 地				

注意：（1）当使用+5V 和 PUL-端口时，则选择电机指令脉冲信号输出方式为单端输出；当使用 PUL+和 PUL-端口时，则选择电机指令脉冲信号输出方式为差分输出。

（2）ACCX400B 接线盒编码器口 1~2 轴仅支持差分接法，3~4 轴同时支持单端和差分接法

3、 CN13 IO 扩展接口定义

CN13 为 IO 扩展接口，采用 RJ45 接口，可连接 CAN-IO 扩展模块。

4、 CN15 辅助编码器/手轮接口定义

CN15 为辅助编码器/手轮接口，采用 DB9 母头。其引脚号和信号对应关系见表 2.3 所示。

表 2.3　接口 CN15 引脚号和信号关系表

序	名称	I/O	说　明
1	EA+	I	编码器/手轮输入
2	EB+	I	编码器/手轮输入
3	EZ+	I	编码器输入
4		-	NC
5	+5V	O	内部 5V 输出

6	EA-	I	编码器/手轮输入
7	EB-	I	编码器/手轮输入
8	EZ-	I	编码器输入
9	GND	O	内部 5V 地

5、 CN16 电源定义

接口 CN16 是接线盒的电源输入接口，板上标有 24V 的端子接+24，标有 OGND 的端子接外部电源地,标有 FG 的端子为机壳地接口。

6、 CN17 接口定义

CN17 接口为 0~3 轴电机的控制信号、IO 信号及低速手轮通道等与控制卡的接口。

7、 CN19 数字量输入端子定义

CN19 为数字量输入接口，其引脚号和信号对应关系见表 2.4 所示：

表 2.4　接口 CN9 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	IN0	I	通用输入（低速）	9	IN8	I	通用输入（低速）
2	IN1	I	通用输入（低速）	10	IN9	I	通用输入（低速）
3	IN2	I	通用输入（低速）	11	IN10	I	通用输入（低速）
4	IN3	I	通用输入（低速）	12	IN11	I	通用输入（低速）
5	IN4	I	通用输入（低速）	13	IN12	I	通用输入（低速）
6	IN5	I	通用输入（低速）	14	IN13	I	通用输入（低速）
7	IN6	I	通用输入（低速）	15	IN14	I	通用输入/LTC0（高速）
8	IN7	I	通用输入（低速）	16	IN15	I	通用输入/LTC1（高速）

8、 CN20 数字量输入接口定义

CN20 为数字量输入接口，其引脚号和信号对应关系见表 2.5 所示。

表 2.5　接口 CN20 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	ORG0	I	0 轴原点输入	7	EL1+	I	1 轴正限位
2	ORG1	I	1 轴原点输入	8	EL1-	I	1 轴负限位
3	ORG2	I	2 轴原点输入	9	EL2+	I	2 轴正限位
4	ORG3	I	3 轴原点输入	10	EL2-	I	2 轴负限位
5	EL0+	I	0 轴正限位	11	EL3+	I	3 轴正限位
6	EL0-	I	0 轴负限位	12	EL3-	I	3 轴负限位

9、 CN21 数字量输出接口定义

CN21 为数字量输出接口，其引脚号和信号对应关系见表 2.6 所示。

表 2.6 接口 CN21 引脚号和信号关系表

序	名称	I/O	说 明	序	名称	I/O	说 明
1	OUT0	O	通用输出（低速）	9	OUT8	O	通用输出（低速）
2	OUT1	O	通用输出（低速）	10	OUT9	O	通用输出（低速）
3	OUT2	O	通用输出（低速）	11	OUT10	O	通用输出（低速）
4	OUT3	O	通用输出（低速）	12	OUT11	O	通用输出（低速）
5	OUT4	O	通用输出（低速）	13	OUT14	O	通用输出/CMP2（高速）
6	OUT5	O	通用输出（低速）	14	OUT15	O	通用输出/CMP3（高速）
7	OUT6	O	通用输出（低速）	15	OGND	O	外部电源地
8	OUT7	O	通用输出（低速）	16	OGND	O	外部电源地

10、CN22 模拟量输入、输出接口定义（选配）

CN22 为模拟量输入、输出接口，其引脚号和信号对应关系见表 2.7 所示。

表 2.7 接口 CN22 引脚号和信号关系表

序	名称	I/O	说 明	序	名称	I/O	说 明
1	AIN0	I	模拟量输入	9	AOUT0	O	模拟量输出
2	AIN1	I	模拟量输入	10	AOUT1	O	模拟量输出
3	AIN2	I	模拟量输入	11	GND	O	内部电源地
4	AIN3	I	模拟量输入	12	GND	O	内部电源地
5	AIN4	I	模拟量输入	13	GND	O	内部电源地
6	AIN5	I	模拟量输入	14	GND	O	内部电源地
7	AIN6	I	模拟量输入	15	GND	O	内部电源地
8	AIN7	I	模拟量输入	16			

说明： 1）支持 8 路模拟量输入和 2 路模拟量输出；

2）AIN0~AIN7 输入电压为-10V~＋10V，12bit 精度；

3）ADC 输入阻抗不小于 100K 欧姆；

4）待测信号的地与接线盒模拟输入端子的 11~15 脚（GND）连接，被采样信号接 AIN0~AIN7；

5）DAC 输出电压范围-10V~＋10V，12bit 精度，默认输出电压 0V。

11、CN23 输出接口定义

CN23 为 PWM 输出接口，其引脚号和信号对应关系如表 F3.8 所示

F3.8 接口 CN23 引脚号和信号关系表

序	名称	I/O	说 明
1	PWM0	O	PWM 输出 0
2	OGND	O	24V 电源地
3	PWM1	O	PWM 输出 1

12、 指示灯定义

ACCX400B 接线盒表面有 3 个指示灯，分别为：

24VLED：外部电源指示灯；

5VLED：内部电源指示灯；

RUNLED：连接状态指示灯。绿色时表示接线盒与控制卡处于通讯状态；红色闪烁时表示接线盒与控制卡未连接成功。

附录 3　ACC3600 接线盒接口说明

ACC3600 接线盒是 DMC3600 运动控制卡的配套产品，扩展 DMC3600 卡的所有用户端口。ACC3600 接线盒外观如图 3.1 所示，接口示意图如图 3.2 所示，尺寸图如图 3.3 所示。

图 3.1　ACC3600 接线盒外观照片

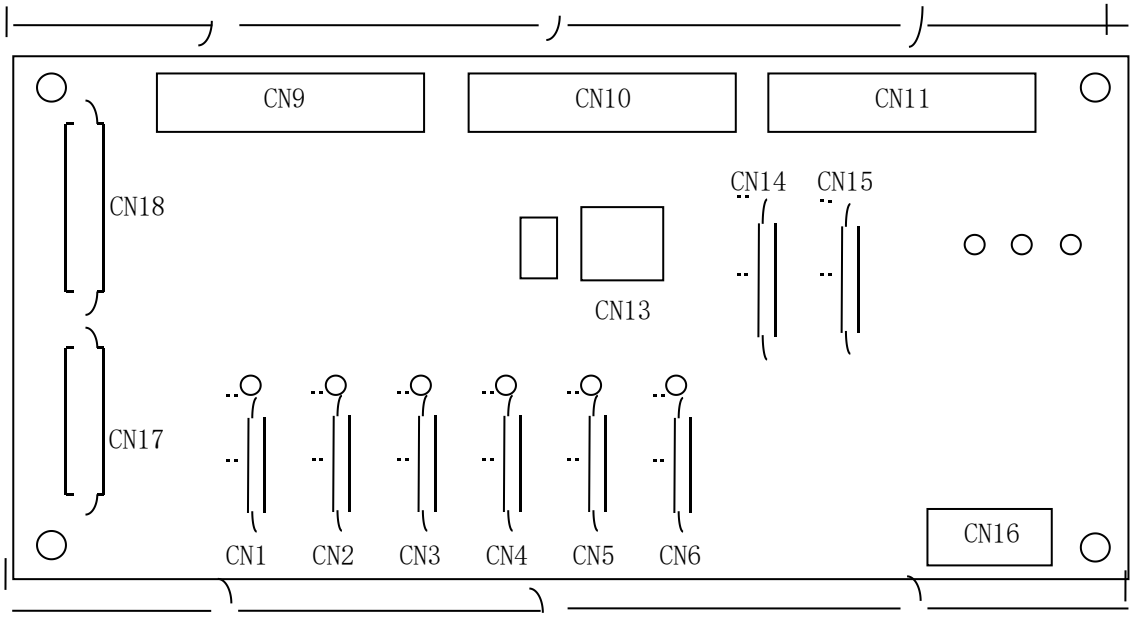


图 3.2　ACC3600 接线盒接口布置示意图

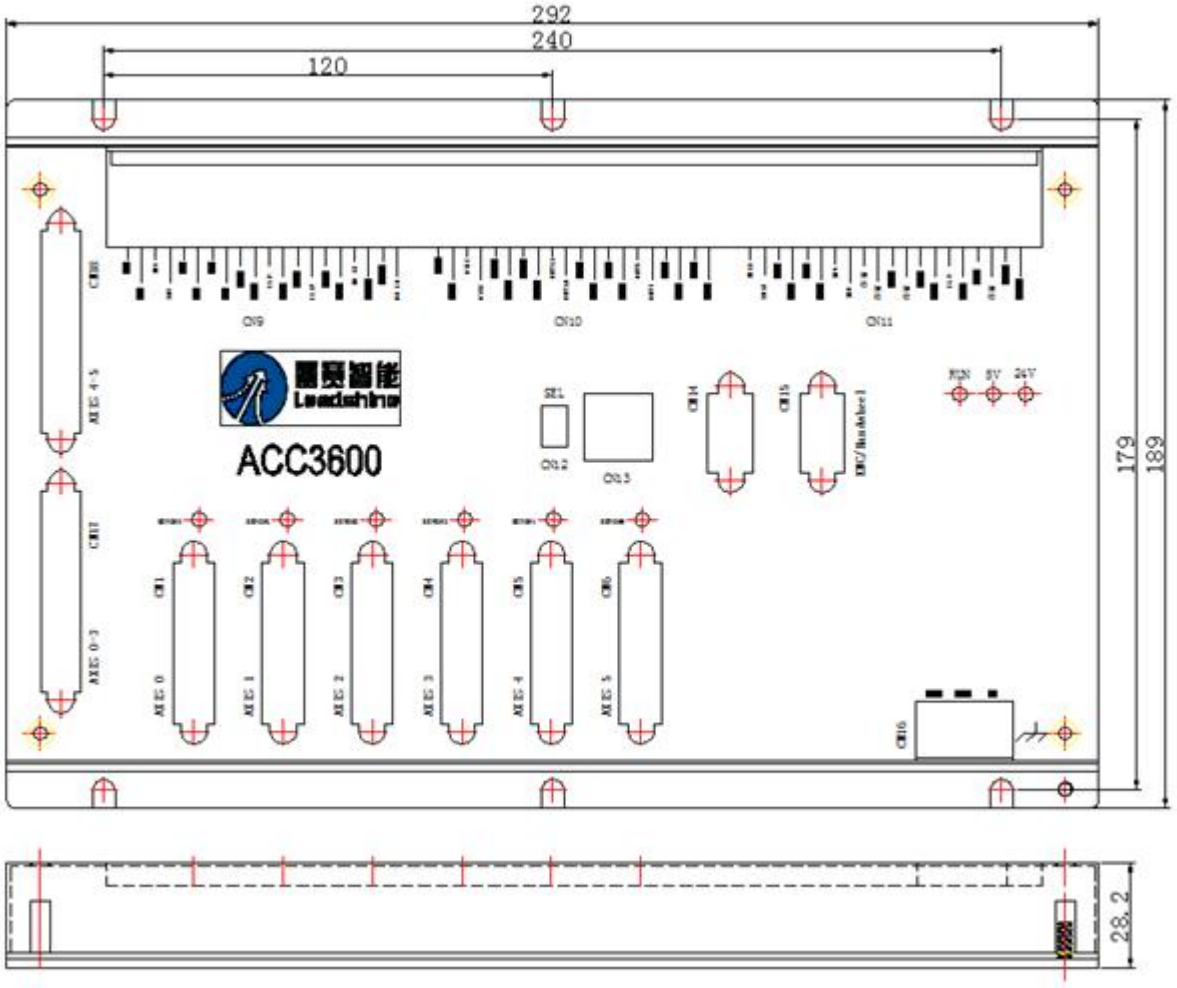


图 3.3　　ACC3600 接线盒尺寸图

表 3.1 ACC3600 接线盒接口功能简述

名称	功能介绍
CN1~CN6	第 0~5 轴轴控制信号
CN9	数字量输入端口
CN10	数字量输出端口
CN11	数字量输入端口
CN12	通用 IO 口初始电平选择开关
CN13	IO 扩展接口
CN14	保留
CN15	辅助编码器接口
CN16	24V 直流电源输入端口
CN17	运动控制卡连接端口（0~3 轴）
CN18	运动控制卡连接端口（4~5 轴）

1、 CN1~CN6 轴控制端口信号定义及使用说明

1）ACC3600 接线盒 CN1 至 CN6 为电机控制信号端口，采用 DB25 母头，其引脚号和信号对应关系见表 3.2 所示。其中 24V 电源端口主要为控制卡的电机控制信号（伺服专用信号）供电。

2）电机控制信号端（CN1 至 CN6）的 24V 为控制信号电源，不能用于驱动器等动力负载供电。

3）根据 ACC3600 的 PCB 板的铜线宽度与厚度，6 个电机控制信号端口的 24V 总输出电流需要控制在 1.5A 以内。

表 3.2 接口 CN1~CN6 引脚号和信号关系表

序	名称	I/O	说 明	序	名称	I/O	说 明
1	OGND	O	24V 电源地	14	24V	O	+24V 输出
2	ALM	I	驱动报警	15	ERC	O	驱动报警复位
3	SEVON	O	驱动允许	16	INP	I	到位信号
4	EA-	I	编码器输入	17	EA+	I	编码器输入
5	EB-	I	编码器输入	18	EB+	I	编码器输入
6	EZ-	I	编码器输入	19	EZ+	I	编码器输入
7	+5V	O	内部 5V	20	GND	O	内部 5V 地
8	保留	-	保留	21	GND	O	内部 5V 地
9	DIR+	O	方向输出	22	DIR-	O	方向输出
10	GND	O	内部 5V 地	23	PUL+	O	脉冲输出
11	PUL-	O	脉冲输出	24	GND	O	内部 5V 地
12	RDY	I	伺服准备完成	25	保留		保留
13	GND	O	内部 5V 地				

注意：（1）当使用+5V 和 PUL-端口时，则选择电机指令脉冲信号输出方式为单端输出；当使用 PUL+和 PUL-端口时，则选择电机指令脉冲信号输出方式为差分输出。

（2）ACC3600 接线盒编码器口 1~4 轴仅支持差分接法，5~6 轴同时支持单端和差分接法

2、 CN9 数字量输入端口定义

CN9 为数字量输入接口，其引脚号和信号对应关系见表 3.3 所示。

表 3.3 接口 CN9 引脚号和信号关系表

序	名称	I/O	说 明	序	名称	I/O	说 明
1	HOME0	I	0 轴原点输入	11	EL3+	I	3 轴正限位
2	HOME1	I	1 轴原点输入	12	EL3-	I	3 轴负限位
3	HOME2	I	2 轴原点输入	13	IN0	I	通用输入（低速）

序	名称	I/O	说　明	序	名称	I/O	说　明
4	HOME3	I	3 轴原点输入	14	IN1	I	通用输入（低速）
5	EL0+	I	0 轴正限位	15	IN2	I	通用输入（低速）
6	EL0-	I	0 轴负限位	16	IN3	I	通用输入（低速）
7	EL1+	I	1 轴正限位	17	IN4	I	通用输入（低速）
8	EL1-	I	1 轴负限位	18	IN5	I	通用输入（低速）
9	EL2+	I	2 轴正限位	19	IN6	I	通用输入（低速）
10	EL2-	I	2 轴负限位	20	IN7	I	通用输入（低速）

3、 CN10 数字量输出接口定义

CN10 为数字量输出接口，其引脚号和信号对应关系见表 3.4 所示。

表 3.4　接口 CN10 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	OUT0	O	通用输出（低速）	11	OUT10	O	通用输出（低速）
2	OUT1	O	通用输出（低速）	12	OUT11	O	通用输出（低速）
3	OUT2	O	通用输出（低速）	13	OUT12	O	通用输出/CMP0（高速）
4	OUT3	O	通用输出（低速）	14	OUT13	O	通用输出/CMP1（高速）
5	OUT4	O	通用输出（低速）	15	OUT14	O	通用输出/CMP2（高速）
6	OUT5	O	通用输出（低速）	16	OUT15	O	通用输出/CMP3（高速）
7	OUT6	O	通用输出（低速）	17	OVCC	O	+24V 输出
8	OUT7	O	通用输出（低速）	18	OVCC	O	+24V 输出
9	OUT8	O	通用输出（低速）	19	OGND	O	外部电源地
10	OUT9	O	通用输出（低速）	20	OGND	O	外部电源地

注意：当 CMP0~3 端口不设置为高速位置比较器输出端口时，可以设置为 OUT12~15 通用输出端口。

4、 CN11 数字量输入接口定义

CN11 为数字量输入接口，其引脚号和信号对应关系见表 3.5 所示。

表 3.5　接口 CN11 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	HOME4	I	4 轴原点输入	11	EL7+	I	保留
2	HOME5	I	5 轴原点输入	12	EL7-	I	保留
3	HOME6	I	保留	13	IN8	I	通用输入（低速）
4	HOME7	I	保留	14	IN9	I	通用输入（低速）
5	EL4+	I	4 轴正限位	15	IN10	I	通用输入（低速）
6	EL4-	I	4 轴负限位	16	IN11	I	通用输入（低速）
7	EL5+	I	5 轴正限位	17	IN12	I	通用输入（低速）

序	名称	I/O	说　明	序	名称	I/O	说　明
8	EL5-	I	5 轴负限位	18	IN13	I	通用输入（低速）
9	EL6+	I	保留	19	IN14	I	通用输入/LTC0（高速）
10	EL6-	I	保留	20	IN15	I	通用输入/LTC1（高速）

5、 CN12 拔码开关定义（保留，固定为 ON）

6、 CN13 IO 扩展接口定义

CN13 为 IO 扩展接口，采用 RJ45 接口，可连接 CAN-IO 扩展模块。

7、 CN14 接口定义（保留）

8、 CN15 辅助编码器/手轮接口定义

CN15 为辅助编码器/手轮接口，采用 DB9 母头。其引脚号和信号对应关系见表 3.6 所示。

表 3.6　接口 CN15 引脚号和信号关系表

序	名称	I/O	说　明
1	EA+	I	编码器/手轮输入
2	EB+	I	编码器/手轮输入
3	EZ+	I	编码器输入
4		-	NC
5	+5V	O	内部 5V 输出
6	EA-	I	编码器/手轮输入
7	EB-	I	编码器/手轮输入
8	EZ-	I	编码器输入
9	GND	O	内部 5V 地

9、 CN16 电源定义

接口 CN16 是接线盒的电源输入接口，板上标有 24V 的端子接+24，标有 OGND 的端子接外部电源地。

10、 CN17 接口定义

CN17 接口为 0~3 轴电机的控制信号、IO 信号及低速手轮通道等与控制卡的接口。

11、 CN18 接口定义

CN18 接口为 4~5 轴电机的控制信号及高速手轮通道与控制卡的接口。

12、 指示灯定义

ACC3600 模块表面有 3 个指示灯，分别为：

24VLED：外部电源指示灯；

5VLED：内部电源指示灯；

RUNLED：连接状态指示灯。绿色时表示接线盒与控制卡处于通讯状态；红色闪烁时表示接线盒与控制卡未连接成功。

附录 4　ACC3800 接线盒接口说明

ACC3800 接线盒是 DMC3800 卡的配套产品，扩展 DMC3800 卡的所有用户端口。ACC3800 接线盒外观如图 4.1 所示，接口示意图如图 4.2 所示，尺寸图如图 4.3 所示。

图 4.1　ACC3800 接线盒外观照片

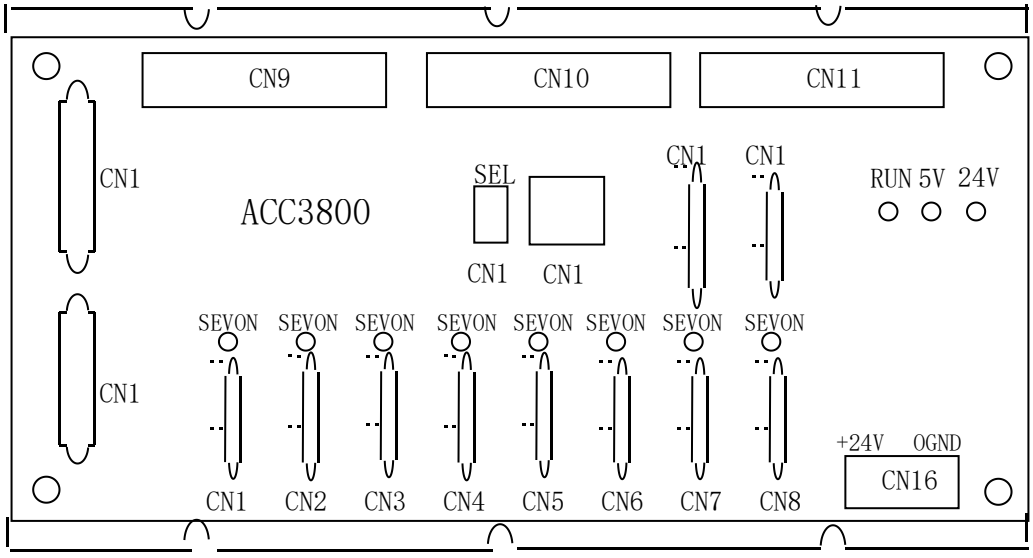


图 4.2　ACC3800 接线盒接口布置示意图

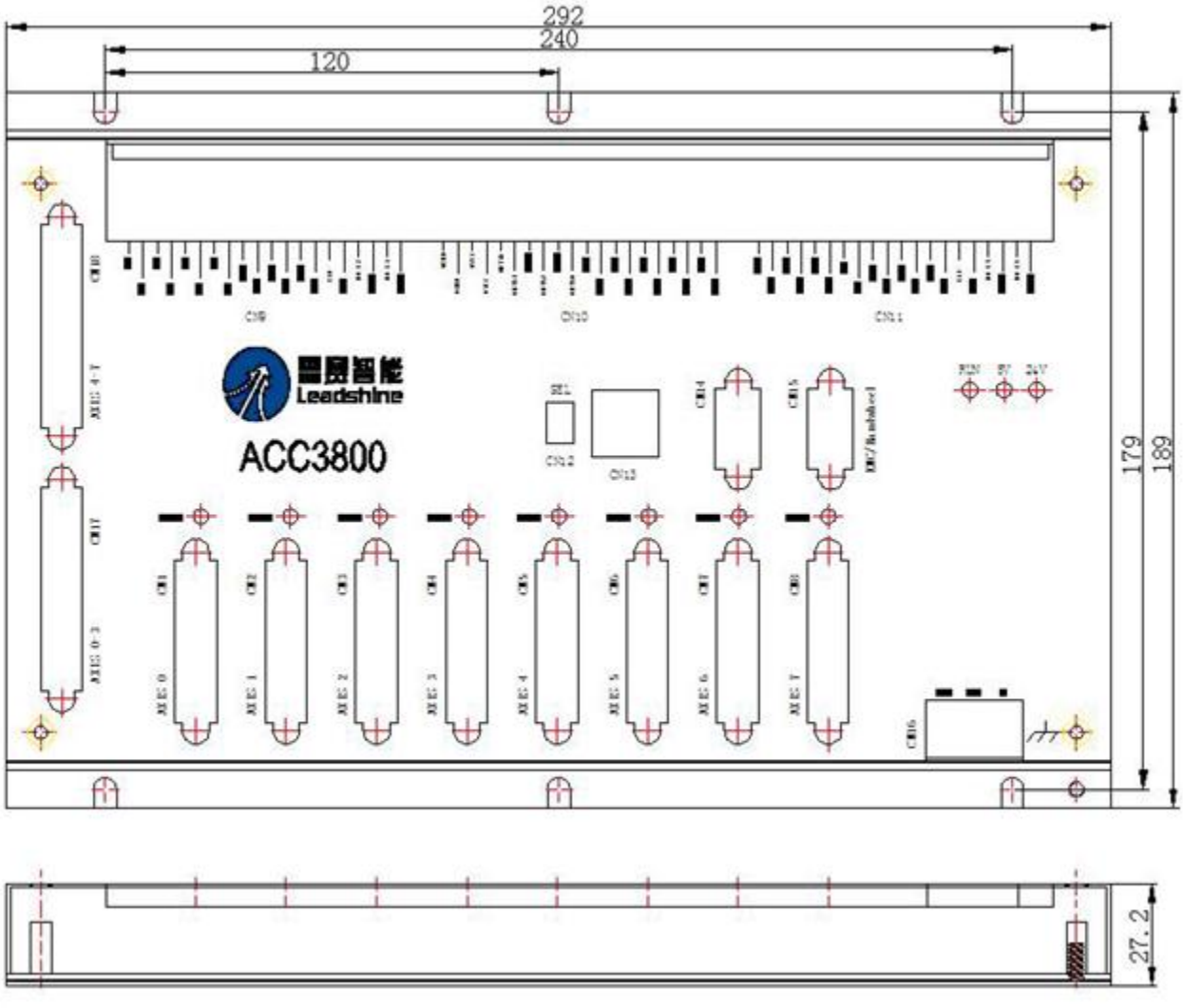


图 4.3　　ACC3800 接线盒尺寸图

表 4.1　ACC3800 接线盒接口功能简述

名称	功能介绍
CN1~CN8	第 0~7 轴轴控制信号
CN9	数字量输入端口
CN10	数字量输出端口
CN11	数字量输入端口
CN12	通用 IO 口初始电平选择开关
CN13	IO 扩展接口
CN14	保留
CN15	辅助编码器接口
CN16	24V 直流电源输入端口
CN17	运动控制卡连接端口（0~3 轴）
CN18	运动控制卡连接端口（4~7 轴）

1、 CN1~CN8 轴控制端口信号定义

1）ACC3800 接线盒 CN1 至 CN8 为电机控制信号端口，采用 DB25 母头，其引脚号和信号对应关系见表 4.2 所示。其中 24V 电源端口主要为控制卡的电机控制信号（伺服专用信号）供电。

2）电机控制信号端（CN1 至 CN8）的 24V 为控制信号电源，不能用于驱动器等动力负载供电。

3）根据 ACC3800 的 PCB 板的铜线宽度与厚度，,8 个电机控制信号端口的 24V 总输出电流需要控制在 1.5A 以内。

表 4.2　接口 CN1~CN8 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	OGND	O	24V 电源地	14	24V	O	+24V 输出
2	ALM	I	驱动报警	15	ERC	O	驱动报警复位
3	SEVON	O	驱动允许	16	INP	I	到位信号
4	EA-	I	编码器输入	17	EA+	I	编码器输入
5	EB-	I	编码器输入	18	EB+	I	编码器输入
6	EZ-	I	编码器输入	19	EZ+	I	编码器输入
7	+5V	O	内部 5V	20	GND	O	内部 5V 地
8	保留	-	保留	21	GND	O	内部 5V 地
9	DIR+	O	方向输出	22	DIR-	O	方向输出
10	GND	O	内部 5V 地	23	PUL+	O	脉冲输出
11	PUL-	O	脉冲输出	24	GND	O	内部 5V 地
12	RDY	I	伺服准备完成	25	保留		保留
13	GND	O	内部 5V 地				

注意：（1）当使用+5V 和 PUL-端口时，则选择电机指令脉冲信号输出方式为单端输出；当使用 PUL+和 PUL-端口时，则选择电机指令脉冲信号输出方式为差分输出。

（2）ACC3800 接线盒编码器口 1~6 轴仅支持差分接法，7~8 轴同时支持单端和差分接法

2、 CN9 数字量输入端口定义

CN9 为数字量输入接口，其引脚号和信号对应关系见表 4.3 所示。

表 4.3　接口 CN9 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	HOME0	I	0 轴原点输入	11	EL3+	I	3 轴正限位

序	名称	I/O	说　明	序	名称	I/O	说　明
2	HOME1	I	1 轴原点输入	12	EL3-	I	3 轴负限位
3	HOME2	I	2 轴原点输入	13	IN0	I	通用输入（低速）
4	HOME3	I	3 轴原点输入	14	IN1	I	通用输入（低速）
5	EL0+	I	0 轴正限位	15	IN2	I	通用输入（低速）
6	EL0-	I	0 轴负限位	16	IN3	I	通用输入（低速）
7	EL1+	I	1 轴正限位	17	IN4	I	通用输入（低速）
8	EL1-	I	1 轴负限位	18	IN5	I	通用输入（低速）
9	EL2+	I	2 轴正限位	19	IN6	I	通用输入（低速）
10	EL2-	I	2 轴负限位	20	IN7	I	通用输入（低速）

3、 CN10 数字量输出接口定义

CN10 为数字量输出接口，其引脚号和信号对应关系见表 4.4 所示。

表 4.4　接口 CN10 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	OUT0	O	通用输出（低速）	11	OUT10	O	通用输出（低速）
2	OUT1	O	通用输出（低速）	12	OUT11	O	通用输出（低速）
3	OUT2	O	通用输出（低速）	13	OUT12	O	通用输出/CMP0（普通）
4	OUT3	O	通用输出（低速）	14	OUT13	O	通用输出/CMP1（普通）
5	OUT4	O	通用输出（低速）	15	OUT14	O	通用输出/CMP2（高速）
6	OUT5	O	通用输出（低速）	16	OUT15	O	通用输出/CMP3（高速）
7	OUT6	O	通用输出（低速）	17	OVCC	O	+24V 输出
8	OUT7	O	通用输出（低速）	18	OVCC	O	+24V 输出
9	OUT8	O	通用输出（低速）	19	OGND	O	外部电源地
10	OUT9	O	通用输出（低速）	20	OGND	O	外部电源地

注意：当 CMP0~3 端口不设置为高速位置比较器输出端口时，可以设置为 OUT12~15 通用输出端口。

4、 CN11 数字量输入接口定义

CN11 为数字量输入接口，其引脚号和信号对应关系见表 4.5 所示。

表 4.5　接口 CN11 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	HOME4	I	4 轴原点输入	11	EL7+	I	7 轴正限位
2	HOME5	I	5 轴原点输入	12	EL7-	I	7 轴负限位
3	HOME6	I	6 轴原点输入	13	IN8	I	通用输入（低速）
4	HOME7	I	7 轴原点输入	14	IN9	I	通用输入（低速）
5	EL4+	I	4 轴正限位	15	IN10	I	通用输入（低速）

序	名称	I/O	说　明	序	名称	I/O	说　明
6	EL4-	I	4 轴负限位	16	IN11	I	通用输入（低速）
7	EL5+	I	5 轴正限位	17	IN12	I	通用输入（低速）
8	EL5-	I	5 轴负限位	18	IN13	I	通用输入（低速）
9	EL6+	I	6 轴正限位	19	IN14	I	通用输入/LTC0（高速）
10	EL6-	I	6 轴负限位	20	IN15	I	通用输入/LTC1（高速）

5、 CN12 拔码开关定义（保留，固定为 ON）

6、 CN13 IO 扩展接口定义

CN13 为 IO 扩展接口，采用 RJ45 接口，可连接 CAN-IO 扩展模块。

7、 CN14 接口定义（保留）

8、 CN15 辅助编码器/手轮接口定义

CN15 为辅助编码器/手轮接口，采用 DB9 母头。其引脚号和信号对应关系见表 4.6 所示。

表 4.6　接口 CN15 引脚号和信号关系表

序	名称	I/O	说　明
1	EA+	I	编码器/手轮输入
2	EB+	I	编码器/手轮输入
3	EZ+	I	编码器输入
4		-	NC
5	+5V	O	内部 5V 输出
6	EA-	I	编码器/手轮输入
7	EB-	I	编码器/手轮输入
8	EZ-	I	编码器输入
9	GND	O	内部 5V 地

9、 CN16 电源定义

接口 CN16 是接线盒的电源输入接口，板上标有 24V 的端子接+24，标有 OGND 的端子接外部电源地。

10、 CN17 接口定义

CN17 接口为 0~3 轴电机的控制信号、IO 信号及低速手轮通道等与控制卡的接口。

11、 CN18 接口定义

CN18 接口为 4~7 轴电机的控制信号及高速手轮通道与控制卡的接口。

12、 指示灯定义

ACC3800 模块表面有 3 个指示灯，分别为：

24VLED：外部电源指示灯；

5VLED：内部电源指示灯；

RUNLED：连接状态指示灯。绿色时表示接线盒与控制卡处于通讯状态；红色闪烁时表示接线盒与控制卡未连接成功。

附录 5　ACC-XC00 接线盒接口说明

ACC-XC00 接线盒是 DMC3C00 运动控制卡的配套产品，扩展 DMC3C00 控制卡的所有用户端子。

ACC-XC00 接线盒尺寸图如图 5.1 所示。

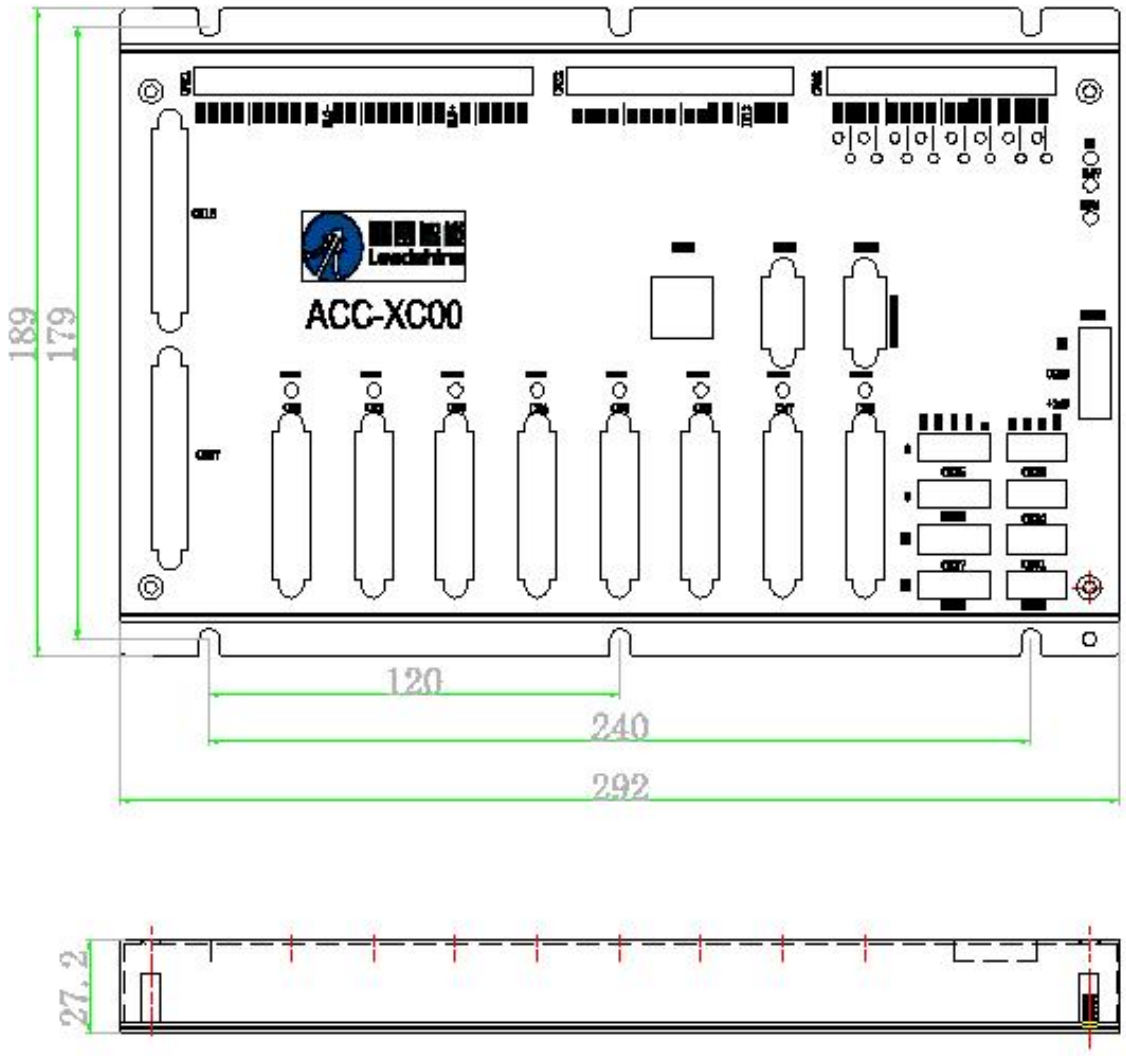


图 5.1　ACC-XC00 尺寸图

ACC-XC00 接线盒外形结构图如图 5.2 所示：

图 5.2　ACC-XC00 接线盒外形结构图

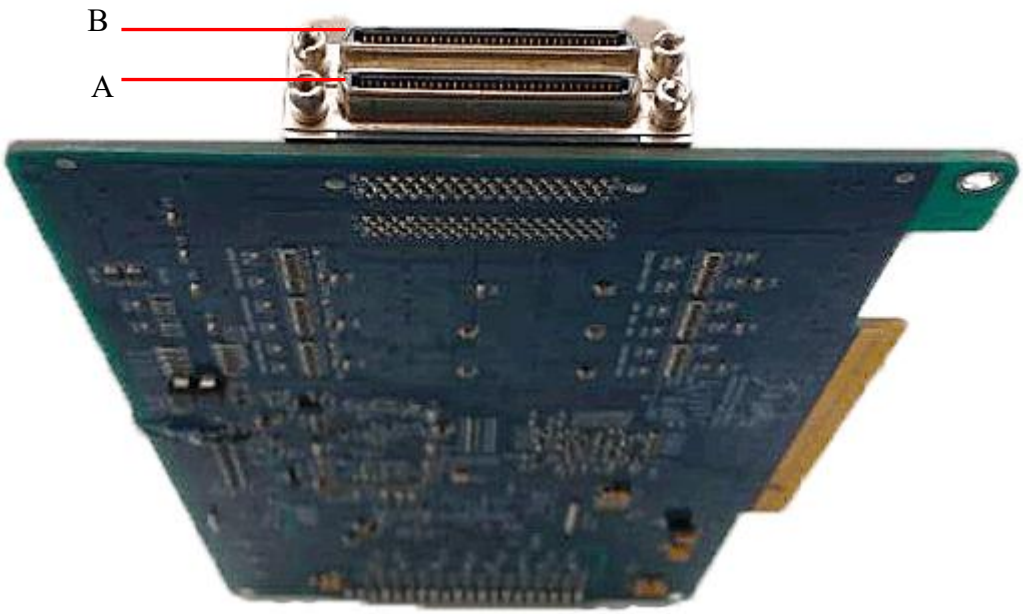


图 5.3　DMC5C10 侧视图

注　意：1）如图 5.3 所示，A为 0~3 轴的电机控制信号以及IO等信号端口，通过电缆线与ACC-XC00 的CN17 端口相连；B为 4~11 轴电机的控制信号以及IO等信号端口，通过电缆线与ACC-XC00 的CN18 端口相连。

2）DMC3C00 与 ACC-XC00 接线盒的连接示意图请参考第 3.2 节

1、 ACC-XC00 接线盒接口及脚位

表 5.1 ACC-XC00 接线盒接口功能简述

名称	功能介绍
CN1~CN8	第 1~8 轴轴控制信号
CN13	CAN 总线接口（RJ45）
CN14	保留
CN15	辅助编码器接口
CN16	24V 直流电源输入端子
CN17	运动控制卡连接端子（1~4 轴）
CN18	运动控制卡连接端子（5~12 轴）
CN21~CN23	第 1~8 轴原点、限位以及通用 IO 信号接口
CN25~28	第 9~12 轴轴控制信号
CN29~32	第 9~12 轴控制信号原点、限位信号接口

2、 CN1~CN8 轴控制端子信号定义

1）ACC-XC00 接线盒 CN1 至 CN8 以及 CN25 至 CN28 为电机控制信号端口，CN1 至 CN8 采用 DB25 母头，其引脚号和信号对应关系见表 F5.2 所示。其中 24V 电源端口主要为控制卡的电机控制信号（伺服专用信号）供电。

2）电机控制信号端（CN1 至 CN8）的 24V 为控制信号电源，不能用于驱动器等动力负载供电。

3）根据 ACC-XC00 接线盒的 PCB 板的铜线宽度与厚度，4 个电机控制信号端口的 24V 总输出电流需要控制在 1.5A 以内。

表 5.2 接口 CN1~CN8 引脚号和信号关系表

序	名称	I/O	说 明	序	名称	I/O	说 明
1	OGND	O	24V 电源地	14	24V	O	+24V 输出
2	ALM	I	驱动报警	15	ERC	O	驱动报警复位
3	SEVON	O	驱动允许	16	INP	I	到位信号
4	EA-	I	编码器输入	17	EA+	I	编码器输入
5	EB-	I	编码器输入	18	EB+	I	编码器输入
6	EZ-	I	编码器输入	19	EZ+	I	编码器输入
7	+5V	O	内部 5V	20	GND	O	内部 5V 地
8	保留	-	保留	21	GND	O	内部 5V 地

9	DIR+	O	方向输出	22	DIR-	O	方向输出
10	GND	O	内部 5V 地	23	PUL+	O	脉冲输出
11	PUL-	O	脉冲输出	24	GND	O	内部 5V 地
12	RDY	I	伺服准备完成	25	保留		保留
13	GND	O	内部 5V 地				

注意：（1）当使用+5V 和 PUL-端口时，则选择电机指令脉冲信号输出方式为单端输出；当使用 PUL+和 PUL-端口时，则选择电机指令脉冲信号输出方式为差分输出。

（2）ACC-XC00 接线盒 1~6 轴编码器口仅支持差分接法，7~8 轴同时支持单端和差分接法

3、 CN13 IO 扩展接口定义

CN13 为 IO 扩展接口，采用 RJ45 接口，可连接 CAN-IO 扩展模块。

4、 CN14 接口定义（保留）

5、 CN15 辅助编码器/手轮接口定义

CN15 为辅助编码器/手轮接口，采用 DB9 母头。其引脚号和信号对应关系见表 5.3 所示。

表 5.3　接口 CN15 引脚号和信号关系表

序	名称	I/O	说　明
1	EA+	I	编码器/手轮输入
2	EB+	I	编码器/手轮输入
3	EZ+	I	编码器输入
4		-	NC
5	+5V	O	内部 5V 输出
6	EA-	I	编码器/手轮输入
7	EB-	I	编码器/手轮输入
8	EZ-	I	编码器输入
9	GND	O	内部 5V 地

6、 CN16 电源定义

接口 CN16 是接线盒的电源输入接口，板上标有 24V 的端子接+24，标有 OGND 的端子接外部电源地，标有 FG 的端子为机壳地接口。

表 5.4　接口 CN16 引脚号和信号关系表

序号	名称	功能

1	+24V	外部 24V 电源输入
2	OGND	外部 24V 电源地
3	FG	机壳地

7、 CN17 接口定义

CN17 接口为 0~3 轴电机的控制信号、IO 信号及低速手轮通道等与控制卡的接口。

8、 CN18 接口定义

CN18 接口为 4~11 轴电机的控制信号及高速手轮通道与控制卡的接口。

9、 CN21 专用输入接口定义（0~7 轴）

CN21 为数字量专用输入接口，其引脚号和信号对应关系见表 5.5 所示：

表 5.5　接口 CN21 引脚号和信号关系

序	名称	I/O	说明	序	名称	I/O	说明
1	ORG0	I	0 轴原点输入	13	EL2+	I	2 轴正限位输入
2	ORG1	I	1 轴原点输入	14	EL2-	I	2 轴负限位输入
3	ORG2	I	2 轴原点输入	15	EL3+	I	3 轴正限位输入
4	ORG3	I	3 轴原点输入	16	EL3-	I	3 轴负限位输入
5	ORG4	I	4 轴原点输入	17	EL4+	I	4 轴正限位输入
6	ORG5	I	5 轴原点输入	18	EL4-	I	4 轴负限位输入
7	ORG6	I	6 轴原点输入	19	EL5+	I	5 轴正限位输入
8	ORG7	I	7 轴原点输入	20	EL5-	I	5 轴负限位输入
9	EL0+	I	0 轴正限位输入	21	EL6+	I	6 轴正限位输入
10	EL0-	I	0 轴负限位输入	22	EL6-	I	6 轴负限位输入
11	EL1+	I	1 轴正限位输入	23	EL7+	I	7 轴正限位输入
12	EL1-	I	1 轴负限位输入	24	EL7-	I	7 轴负限位输入

10、CN22 通用输入接口定义

CN22 为数字量通用输入接口，其引脚号和信号对应关系见表 5.6 所示：

表 5.6　接口 CN21 引脚号和信号关系

序	名称	I/O	说　明	序	名称	I/O	说　明

1	IN0	I	通用输入（低速）	9	IN8	I	通用输入（低速）
2	IN1	I	通用输入（低速）	10	IN9	I	通用输入（低速）
3	IN2	I	通用输入（低速）	11	IN10	I	通用输入（低速）
4	IN3	I	通用输入（低速）	12	IN11	I	通用输入（低速）
5	IN4	I	通用输入（低速）	13	IN12	I	通用输入（低速）
6	IN5	I	通用输入（低速）	14	IN13	I	通用输入（低速）
7	IN6	I	通用输入（低速）	15	IN14	I	通用输入/LTC0（高速）
8	IN7	I	通用输入（低速）	16	IN15	I	通用输入/LTC1（高速）

11、CN23 数字量输出接口定义

CN23 为数字量输出接口，其引脚号和信号对应关系见表 5.7 所示。

表 5.7　接口 CN21 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	OUT0	O	通用输出（低速）	9	OUT8	O	通用输出（低速）
2	OUT1	O	通用输出（低速）	10	OUT9	O	通用输出（低速）
3	OUT2	O	通用输出（低速）	11	OUT10	O	通用输出（低速）
4	OUT3	O	通用输出（低速）	12	OUT11	O	通用输出（低速）
5	OUT4	O	通用输出（低速）	13	OUT12	O	通用输出/CMP0（高速）
6	OUT5	O	通用输出（低速）	14	OUT13	O	通用输出/CMP1（高速）
7	OUT6	O	通用输出（低速）	15	OUT14	O	通用输出/CMP2（高速）
8	OUT7	O	通用输出（低速）	16	OUT15	O	通用输出/CMP3（高速）

12、CN25 ~ CN28 电机接口端子定义

CN25~CN28 为 8~11 轴电机控制信号端口，其引脚号和信号对应关系见表 5.8 所示。

表 5.8　接口 CN25~CN28 引脚号和信号关系表

序号	名称	说明
1	DIR+	方向输出
2	DIR-	方向输出
3	PUL+	脉冲输出
4	PUL-	脉冲输出
5	5V	内部 5V 输出

说明：CN25 对应第 8 轴；CN26 对应第 9 轴；CN27 对应第 10 轴；CN28 对应第 11 轴

13、CN29 ~ CN32 专用输入接口定义（8~11 轴)

CN29~CN32 为 8~11 轴专用输入信号接口，其引脚号和信号对应关系见表 5.9 所示。

表 5.9　接口 CN25~CN28 引脚号和信号关系表

序号	名称	说明
1	ORG	原点输入
2	EL+	正限位输入
3	EL-	负限位输入
4	OGND	24V 电源地

说明：CN29 对应第 8 轴；CN30 对应第 9 轴；CN31 对应第 10 轴；CN32 对应第 11 轴

14、 指示灯定义

ACC-XC00 模块表面有 3 个指示灯，分别为：

24VLED：外部电源指示灯；

5VLED：内部电源指示灯；

RUNLED：连接状态指示灯。绿色时表示接线盒与控制卡处于通讯状态；红色闪烁时表示接线盒与控制卡未连接成功。

附录 6　ACC2-X400B 接线盒接口说明

ACC2-X400B 接线盒是 DMC3400A-PCIe 运动控制卡的配套产品，扩展 DMC3400A-PCIe 控制卡的所有用户端子。

ACC2-X400B 接线盒尺寸图如图 6.1 所示。

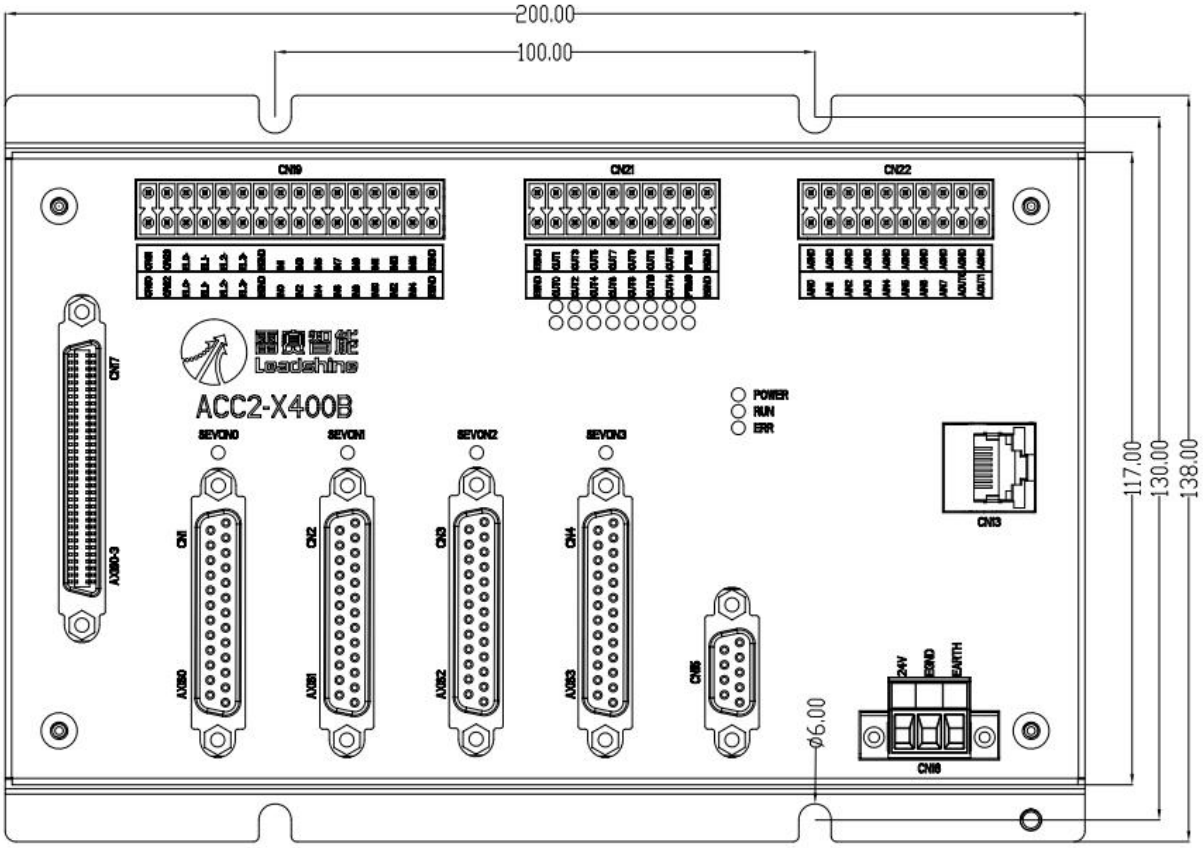


图 6.1　ACC2-X400B 接线盒尺寸图

ACC2-X400B 接线盒外形结构图如图 6.2 所示：

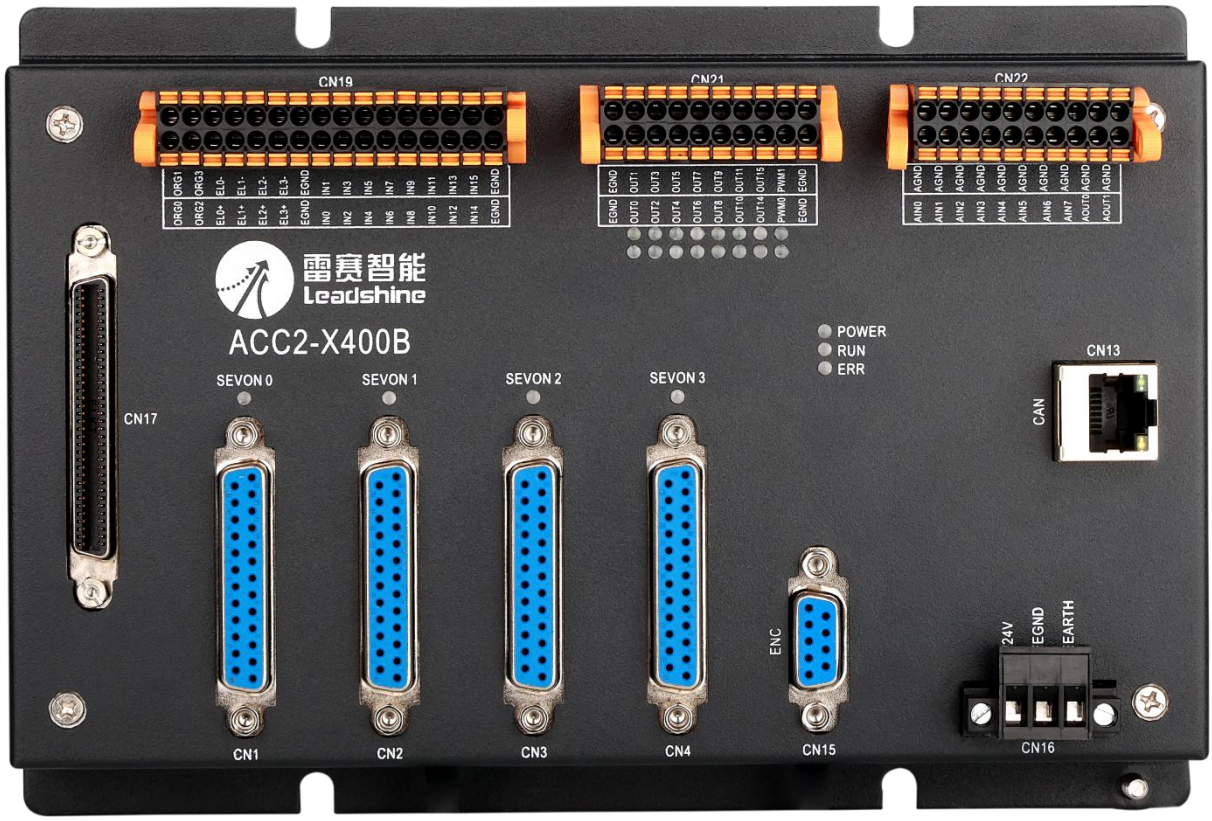


图 6.2　ACC2-X400B 接线盒外形结构图

1、 ACC2-X400B 接线盒接口及脚位

表 6.1　ACC2-X400B 接线盒接口功能简述

名称	功能介绍
CN1~CN4	第 1~4 轴轴控制信号
CN13	CAN 总线接口（RJ45）
CN15	辅助编码器接口
CN16	24V 直流电源输入端子
CN17	运动控制卡连接端子（1~4 轴）
CN19	第 1-4 轴原点及限位信号输入接口和数字量通用输入接口、
CN21	数字量输出端子和 PWM 输出接口

CN22	模拟量输入输出接口

2、 CN1~CN4 轴控制端子信号定义

1）ACC2-X400B 接线盒 CN1 至 CN4 为电机控制信号端口，采用 DB25 母头，其引脚号和信号对应关系见表 6.2 所示。其中 24V 电源端口主要为控制卡的电机控制信号（伺服专用信号）供电。

2）电机控制信号端（CN1 至 CN4）的 24V 为控制信号电源，不能用于驱动器等动力负载供电。

3）根据 ACC2-X400B 接线盒的 PCB 板的铜线宽度与厚度，4 个电机控制信号端口的 24V 总输出电流需要控制在 1.5A 以内。

表 6.2　接口 CN1~CN4 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	OGND	O	24V 电源地	14	24V	O	+24V 输出
2	ALM	I	驱动报警	15	ERC	O	驱动报警复位
3	SEVON	O	驱动允许	16	INP	I	到位信号
4	EA-	I	编码器输入	17	EA+	I	编码器输入
5	EB-	I	编码器输入	18	EB+	I	编码器输入
6	EZ-	I	编码器输入	19	EZ+	I	编码器输入
7	+5V	O	内部 5V	20	GND	O	内部 5V 地
8	保留	-	保留	21	GND	O	内部 5V 地
9	DIR+	O	方向输出	22	DIR-	O	方向输出
10	GND	O	内部 5V 地	23	PUL+	O	脉冲输出
11	PUL-	O	脉冲输出	24	GND	O	内部 5V 地
12	RDY	I	伺服准备完成	25	保留		保留
13	GND	O	内部 5V 地				

注意：（1）当使用+5V 和 PUL-端口时，则选择电机指令脉冲信号输出方式为单端输出；当使用 PUL+和 PUL-端口时，则选择电机指令脉冲信号输出方式为差分输出。

（2）ACC2-X400B 接线盒编码器口 1~2 轴仅支持差分接法，3~4 轴同时支持单端和差分接法

3、 CN13 IO 扩展接口定义

CN13 为 IO 扩展接口，采用 RJ45 接口，可连接 CAN-IO 扩展模块。

4、 CN15 辅助编码器/手轮接口定义

CN15 为辅助编码器/手轮接口，采用 DB9 母头。其引脚号和信号对应关系见表 6.3 所示。

表 6.3　接口 CN15 引脚号和信号关系表

序	名称	I/O	说　明
1	EA+	I	编码器/手轮输入
2	EB+	I	编码器/手轮输入
3	EZ+	I	编码器输入
4		-	NC
5	+5V	O	内部 5V 输出
6	EA-	I	编码器/手轮输入
7	EB-	I	编码器/手轮输入
8	EZ-	I	编码器输入
9	GND	O	内部 5V 地

5、 CN16 电源定义

接口 CN16 是接线盒的电源输入接口，板上标有 24V 的端子接+24，标有 EGND 的端子接外部电源地,标有 EARTH 的端子为机壳地接口。

6、 CN17 接口定义

CN17 接口为 0~3 轴电机的控制信号、IO 信号及低速手轮通道等与控制卡的接口。

7、 CN19 原点及限位信号输入接口和数字量输入接口定义

CN19 为第 1-4 轴原点及限位信号输入接口和数字量通用输入接口，其引脚号和信号对应关系见表 6.4 所示：

表 6.4　接口 CN9 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	ORG0	I	0 轴原点输入	17	ORG1	I	1 轴原点输入
2	ORG2	I	2 轴原点输入	18	ORG3	I	3 轴原点输入
3	EL0+	I	0 轴正限位输入	19	EL0-	I	0 轴负限位输入
4	EL1+	I	1 轴正限位输入	20	EL1-	I	1 轴负限位输入
5	EL2+	I	2 轴正限位输入	21	EL2-	I	2 轴负限位输入
6	EL3+	I	3 轴正限位输入	22	EL3-	I	3 轴负限位输入
7	EGND	O	24V 电源地	23	EGND	O	24V 电源地
8	IN0	I	通用输入（低速）	24	IN1	I	通用输入（低速）
9	IN2	I	通用输入（低速）	25	IN3	I	通用输入（低速）
10	IN4	I	通用输入（低速）	26	IN5	I	通用输入（低速）
11	IN6	I	通用输入（低速）	27	IN7	I	通用输入（低速）
12	IN8	I	通用输入（低速）	28	IN9	I	通用输入（低速）

13	IN10	I	通用输入（低速）	29	IN11	I	通用输入（低速）
14	IN12	I	通用输入（低速）	30	IN13	I	通用输入（低速）
15	IN14	I	通用输入/LTC0（高速）	31	IN15	I	通用输入/LTC1（高速）
16	EGND	O	24V 电源地	32	EGND	O	24V 电源地

8、 CN21 数字量输出接口和 PWM 输出定义

CN21 为数字量输出端子和 PWM 输出接口，其引脚号和信号对应关系见表 6.6 所示。

表 6.6　接口 CN21 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	EGND	O	24V 电源地	11	EGND	O	24V 电源地
2	OUT0	O	通用输出（低速）	12	OUT1	O	通用输出（低速）
3	OUT2	O	通用输出（低速）	13	OUT3	O	通用输出（低速）
4	OUT4	O	通用输出（低速）	14	OUT5	O	通用输出（低速）
5	OUT6	O	通用输出（低速）	15	OUT7	O	通用输出（低速）
6	OUT8	O	通用输出（低速）	16	OUT9	O	通用输出（低速）
7	OUT10	O	通用输出（低速）	17	OUT11	O	通用输出（低速）
8	OUT14	O	通用输出/CMP2（高速）	18	OUT15	O	通用输出/CMP3（高速）
9	PWM0	O	PWM 输出 0	19	PWM1	O	PWM 输出 1
10	EGND	O	24V 电源地	20	EGND	O	24V 电源地

9、 CN22 模拟量输入、输出接口定义

CN22 为模拟量输入、输出接口，其引脚号和信号对应关系见表 6.7 所示。

表 6.7　接口 CN22 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	AIN0	I	模拟量输入	11	GND	O	内部电源地
2	AIN1	I	模拟量输入	12	GND	O	内部电源地
3	AIN2	I	模拟量输入	13	GND	O	内部电源地
4	AIN3	I	模拟量输入	14	GND	O	内部电源地
5	AIN4	I	模拟量输入	15	GND	O	内部电源地
6	AIN5	I	模拟量输入	16	GND	O	内部电源地
7	AIN6	I	模拟量输入	17	GND	O	内部电源地
8	AIN7	I	模拟量输入	18	GND	O	内部电源地
9	AOUT0	O	模拟量输出	19	GND	O	内部电源地
10	AOUT1	O	模拟量输出	20	GND	O	内部电源地

说明： 1）支持 8 路模拟量输入和 2 路模拟量输出；

2）AIN0~AIN7 输入电压为-10V~＋10V，16bit 精度；

3）ADC 输入阻抗不小于 100K 欧姆；

4）待测信号的地与接线盒模拟输入端子的 11~18 脚（GND）连接，被采样信号接 AIN0~AIN7；

5）DAC 输出电压范围-10V~＋10V，16it 精度，默认输出电压 0V。

10、 指示灯定义

ACC2-X400B 接线盒表面有 3 个指示灯，分别为：

POWER：24V 外部电源指示灯；

RUN：连接状态指示灯，绿色时表示接线盒与控制卡处于通讯状态；

ERR：错误指示灯，红色闪烁时表示接线盒与控制卡未连接成功；

附录 7　ACC2-3600 接线盒接口说明

ACC2-3600 接线盒是 DMC3600-PCIe 运动控制卡的配套产品，扩展 DMC3600-PCIe 控制卡的所有用户端子。

ACC2-3600 接线盒尺寸图如图 7.1 所示。

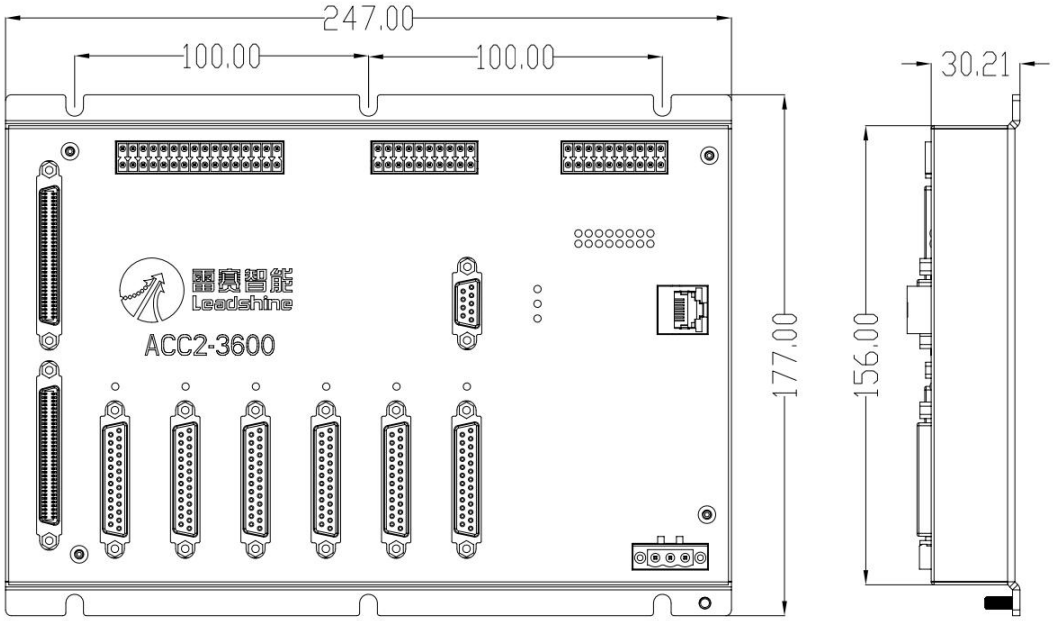


图 6.1　ACC2-3600 接线盒尺寸图

ACC2-3600 接线盒外形结构图如图 6.2 所示

图 6.2　ACC2-3600 接线盒外形结构图

1、 ACC2-3600 接线盒接口及脚位

表 7.1 ACC2-3600 接线盒接口功能简述

名称	功能介绍
CN1~CN6	第 1~6 轴轴控制信号
CN9	第 1-6 轴原点及限位信号输入接口
CN10	数字量输入端子
CN11	数字量输出端子
CN13	CAN 总线接口（RJ45）
CN15	辅助编码器接口
CN16	24V 直流电源输入端子
CN17	运动控制卡连接端口（0~3 轴）
CN18	运动控制卡连接端口（4~5 轴）

2、 CN1~CN6 轴控制端子信号定义

1）ACC2-3600 接线盒 CN1 至 CN6 为电机控制信号端口，采用 DB25 母头，其引脚号和信号对应关系见表 7.2 所示。其中 24V 电源端口主要为控制卡的电机控制信号（伺服专用信号）供电。

2）电机控制信号端（CN1 至 CN6）的 24V 为控制信号电源，不能用于驱动器等动力负载供电。

3）根据 ACC2-3600 接线盒的 PCB 板的铜线宽度与厚度，4 个电机控制信号端口的 24V 总输出电流需要控制在 1.5A 以内。

表 7.2 接口 CN1~CN6 引脚号和信号关系表

序	名称	I/O	说 明	序	名称	I/O	说 明
1	OGND	O	24V 电源地	14	24V	O	+24V 输出
2	ALM	I	驱动报警	15	ERC	O	驱动报警复位
3	SEVON	O	驱动允许	16	INP	I	到位信号
4	EA-	I	编码器输入	17	EA+	I	编码器输入
5	EB-	I	编码器输入	18	EB+	I	编码器输入
6	EZ-	I	编码器输入	19	EZ+	I	编码器输入
7	+5V	O	内部 5V	20	GND	O	内部 5V 地
8	保留	-	保留	21	GND	O	内部 5V 地
9	DIR+	O	方向输出	22	DIR-	O	方向输出
10	GND	O	内部 5V 地	23	PUL+	O	脉冲输出
11	PUL-	O	脉冲输出	24	GND	O	内部 5V 地

12	RDY	I	伺服准备完成	25	保留		保留
13	GND	O	内部 5V 地				

注意：（1）当使用+5V 和 PUL-端口时，则选择电机指令脉冲信号输出方式为单端输出；当使用 PUL+和 PUL-端口时，则选择电机指令脉冲信号输出方式为差分输出。

（2）ACC2-3600 接线盒编码器口 1~4 轴仅支持差分接法，5~6 轴同时支持单端和差分接法

3、 CN9 原点及限位信号输入接口定义

CN9 为第 1-6 轴原点及限位信号输入接口，其引脚号和信号对应关系见表 7.3 所示：

表 7.3　接口 CN9 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	EGND	O	24V 电源地	17	EGND	O	24V 电源地
2	ORG0	I	0 轴原点输入	18	ORG1	I	1 轴原点输入
3	ORG2	I	2 轴原点输入	19	ORG3	I	3 轴原点输入
4	ORG4	I	4 轴原点输入	20	ORG5	I	5 轴原点输入
5	NC		保留	21	NC		保留
6	EGND	O	24V 电源地	22	EGND	O	24V 电源地
7	EL0+	I	0 轴正限位输入	23	EL0-	I	0 轴负限位输入
8	EL1+	I	1 轴正限位输入	24	EL1-	I	1 轴负限位输入
9	EL2+	I	2 轴正限位输入	25	EL2-	I	2 轴负限位输入
10	EL3+	I	3 轴正限位输入	26	EL3-	I	3 轴负限位输入
11	EGND	O	24V 电源地	27	EGND	O	24V 电源地
12	EL4+	I	4 轴正限位输入	28	EL4-	I	4 轴负限位输入
13	EL5+	I	5 轴正限位输入	29	EL5-	I	5 轴负限位输入
14	NC		保留	30	NC		保留
15	NC		保留	31	NC		保留
16	EGND	O	24V 电源地	32	EGND	O	24V 电源地

4、 CN10 数字量输入接口定义

CN10 为数字量输入接口，其引脚号和信号对应关系见表 7.4 所示。

表 7.4　接口 CN10 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	EGND	O	24V 电源地	11	EGND	O	24V 电源地
2	IN0	I	通用输入（低速）	12	IN1	I	通用输入（低速）
3	IN2	I	通用输入（低速）	13	IN3	I	通用输入（低速）
4	IN4	I	通用输入（低速）	14	IN5	I	通用输入（低速）

序	名称	I/O	说　明	序	名称	I/O	说　明
5	IN6	I	通用输入（低速）	15	IN7	I	通用输入（低速）
6	IN8	I	通用输入（低速）	16	IN9	I	通用输入（低速）
7	IN10	I	通用输入（低速）	17	IN11	I	通用输入（低速）
8	IN12	I	通用输入（低速）	18	IN13	I	通用输入（低速）
9	IN14	I	通用输入/LTC0（高速）	19	IN15	I	通用输入/LTC1（高速）
10	EGND	O	24V 电源地	20	EGND	O	24V 电源地

5、 CN11 数字量输出接口定义

CN11 为数字量输出接口，其引脚号和信号对应关系见表 7.5 所示。

表 7.5　接口 CN11 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	EGND	O	24V 电源地	11	EGND	O	24V 电源地
2	OUT0	O	通用输出（低速）	12	OUT1	O	通用输出（低速）
3	OUT2	O	通用输出（低速）	13	OUT3	O	通用输出（低速）
4	OUT4	O	通用输出（低速）	14	OUT5	O	通用输出（低速）
5	OUT6	O	通用输出（低速）	15	OUT7	O	通用输出（低速）
6	OUT8	O	通用输出（低速）	16	OUT9	O	通用输出（低速）
7	OUT10	O	通用输出（低速）	17	OUT11	O	通用输出（低速）
8	OUT12	O	通用输出/CMP0（高速）	18	OUT13	O	通用输出/CMP1（高速）
9	OUT14	O	通用输出/CMP2（高速）	19	OUT15	O	通用输出/CMP3（高速）
10	EGND	O	24V 电源地	20	EGND	O	24V 电源地

注意：当 CMP0~3 端口不设置为高速位置比较器输出端口时，可以设置为 OUT12~15 通用输出端口。

6、 CN13 IO 扩展接口定义

CN13 为 IO 扩展接口，采用 RJ45 接口，可连接 CAN-IO 扩展模块。

7、 CN15 辅助编码器/手轮接口定义

CN15 为辅助编码器/手轮接口，采用 DB9 母头。其引脚号和信号对应关系见表 7.6 所示。

表 7.6　接口 CN15 引脚号和信号关系表

序	名称	I/O	说　明
1	EA+	I	编码器/手轮输入
2	EB+	I	编码器/手轮输入
3	EZ+	I	编码器输入
4		-	NC

5	+5V	O	内部 5V 输出
6	EA-	I	编码器/手轮输入
7	EB-	I	编码器/手轮输入
8	EZ-	I	编码器输入
9	GND	O	内部 5V 地

8、 CN16 电源定义

接口 CN16 是接线盒的电源输入接口，板上标有 24V 的端子接+24，标有 EGND 的端子接外部电源地,标有 EARTH 的端子为机壳地接口。

9、 CN17 接口定义

CN17 接口为 0~3 轴电机的控制信号、IO 信号及低速手轮通道等与控制卡的接口。

10、CN18 接口定义

CN18 接口为 4~5 轴电机的控制信号及高速手轮通道与控制卡的接口。

11、指示灯定义

ACC2-3600 接线盒表面有 3 个指示灯，分别为：

POWER：24V 外部电源指示灯；

RUN：连接状态指示灯，绿色时表示接线盒与控制卡处于通讯状态；

ERR：错误指示灯，红色闪烁时表示接线盒与控制卡未连接成功；

附录 8　ACC2-3800 接线盒接口说明

ACC2-3800 接线盒是 DMC3800-PCIe 运动控制卡的配套产品，扩展 DMC3800-PCIe 控制卡的所有用户端子。

ACC2-3800 接线盒尺寸图如图 8.1 所示。

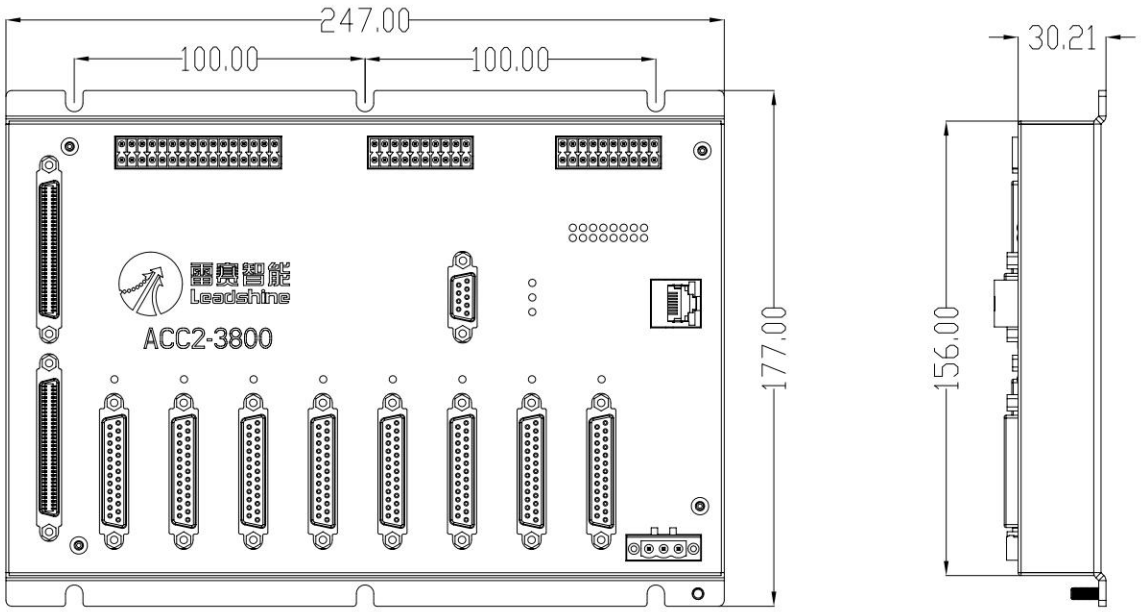


图 8.1　ACC2-3800 接线盒尺寸图

ACC2-3800 接线盒外形结构图如图 8.2 所示：

图 8.2　ACC2-3800 接线盒外形结构图

1、 ACC2-3800 接线盒接口及脚位

表 8.1 ACC2-3800 接线盒接口功能简述

名称	功能介绍
CN1~CN8	第 1~8 轴轴控制信号
CN9	第 1-8 轴原点及限位信号输入接口
CN10	数字量输入端子
CN11	数字量输出端子
CN13	CAN 总线接口（RJ45）
CN15	辅助编码器接口
CN16	24V 直流电源输入端子
CN17	运动控制卡连接端口（0~3 轴）
CN18	运动控制卡连接端口（4~7 轴）

2、 CN1~CN8 轴控制端子信号定义

1）ACC2-3800 接线盒 CN1 至 CN8 为电机控制信号端口，采用 DB25 母头，其引脚号和信号对应关系见表 7.2 所示。其中 24V 电源端口主要为控制卡的电机控制信号（伺服专用信号）供电。

2）电机控制信号端（CN1 至 CN6）的 24V 为控制信号电源，不能用于驱动器等动力负载供电。

3）根据 ACC2-3600 接线盒的 PCB 板的铜线宽度与厚度，4 个电机控制信号端口的 24V 总输出电流需要控制在 1.5A 以内。

表 8.2 接口 CN1~CN8 引脚号和信号关系表

序	名称	I/O	说 明	序	名称	I/O	说 明
1	OGND	O	24V 电源地	14	24V	O	+24V 输出
2	ALM	I	驱动报警	15	ERC	O	驱动报警复位
3	SEVON	O	驱动允许	16	INP	I	到位信号
4	EA-	I	编码器输入	17	EA+	I	编码器输入
5	EB-	I	编码器输入	18	EB+	I	编码器输入
6	EZ-	I	编码器输入	19	EZ+	I	编码器输入
7	+5V	O	内部 5V	20	GND	O	内部 5V 地
8	保留	-	保留	21	GND	O	内部 5V 地
9	DIR+	O	方向输出	22	DIR-	O	方向输出
10	GND	O	内部 5V 地	23	PUL+	O	脉冲输出
11	PUL-	O	脉冲输出	24	GND	O	内部 5V 地

12	RDY	I	伺服准备完成	25	保留		保留
13	GND	O	内部 5V 地				

注意：（1）当使用+5V 和 PUL-端口时，则选择电机指令脉冲信号输出方式为单端输出；当使用 PUL+和 PUL-端口时，则选择电机指令脉冲信号输出方式为差分输出。

（2）ACC2-3800 接线盒编码器口 1~6 轴仅支持差分接法，7~8 轴同时支持单端和差分接法

3、 CN9 原点及限位信号输入接口定义

CN9 为第 1-8 轴原点及限位信号输入接口，其引脚号和信号对应关系见表 8.3 所示：

表 8.3　接口 CN9 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	EGND	O	24V 电源地	17	EGND	O	24V 电源地
2	ORG0	I	0 轴原点输入	18	ORG1	I	1 轴原点输入
3	ORG2	I	2 轴原点输入	19	ORG3	I	3 轴原点输入
4	ORG4	I	4 轴原点输入	20	ORG5	I	5 轴原点输入
5	ORG6	I	6 轴原点输入	21	ORG7	I	7 轴原点输入
6	EGND	O	24V 电源地	22	EGND	O	24V 电源地
7	EL0+	I	0 轴正限位输入	23	EL0-	I	0 轴负限位输入
8	EL1+	I	1 轴正限位输入	24	EL1-	I	1 轴负限位输入
9	EL2+	I	2 轴正限位输入	25	EL2-	I	2 轴负限位输入
10	EL3+	I	3 轴正限位输入	26	EL3-	I	3 轴负限位输入
11	EGND	O	24V 电源地	27	EGND	O	24V 电源地
12	EL4+	I	4 轴正限位输入	28	EL4-	I	4 轴负限位输入
13	EL5+	I	5 轴正限位输入	29	EL5-	I	5 轴负限位输入
14	EL6+	I	6 轴正限位输入	30	EL6-	I	6 轴负限位输入
15	EL7+	I	7 轴正限位输入	31	EL7-	I	7 轴负限位输入
16	EGND	O	24V 电源地	32	EGND	O	24V 电源地

4、 CN10 数字量输入接口定义

CN10 为数字量输入接口，其引脚号和信号对应关系见表 8.4 所示。

表 8.4　接口 CN10 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	EGND	O	24V 电源地	11	EGND	O	24V 电源地
2	IN0	I	通用输入（低速）	12	IN1	I	通用输入（低速）
3	IN2	I	通用输入（低速）	13	IN3	I	通用输入（低速）
4	IN4	I	通用输入（低速）	14	IN5	I	通用输入（低速）

序	名称	I/O	说　明	序	名称	I/O	说　明
5	IN6	I	通用输入（低速）	15	IN7	I	通用输入（低速）
6	IN8	I	通用输入（低速）	16	IN9	I	通用输入（低速）
7	IN10	I	通用输入（低速）	17	IN11	I	通用输入（低速）
8	IN12	I	通用输入（低速）	18	IN13	I	通用输入（低速）
9	IN14	I	通用输入/LTC0（高速）	19	IN15	I	通用输入/LTC1（高速）
10	EGND	O	24V 电源地	20	EGND	O	24V 电源地

5、 CN11 数字量输出接口定义

CN11 为数字量输出接口，其引脚号和信号对应关系见表 8.5 所示。

表 8.5　接口 CN11 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	EGND	O	24V 电源地	11	EGND	O	24V 电源地
2	OUT0	O	通用输出（低速）	12	OUT1	O	通用输出（低速）
3	OUT2	O	通用输出（低速）	13	OUT3	O	通用输出（低速）
4	OUT4	O	通用输出（低速）	14	OUT5	O	通用输出（低速）
5	OUT6	O	通用输出（低速）	15	OUT7	O	通用输出（低速）
6	OUT8	O	通用输出（低速）	16	OUT9	O	通用输出（低速）
7	OUT10	O	通用输出（低速）	17	OUT11	O	通用输出（低速）
8	OUT12	O	通用输出/CMP0（高速）	18	OUT13	O	通用输出/CMP1（高速）
9	OUT14	O	通用输出/CMP2（高速）	19	OUT15	O	通用输出/CMP3（高速）
10	EGND	O	24V 电源地	20	EGND	O	24V 电源地

注意：当 CMP0~3 端口不设置为高速位置比较器输出端口时，可以设置为 OUT12~15 通用输出端口。

6、 CN13 IO 扩展接口定义

CN13 为 IO 扩展接口，采用 RJ45 接口，可连接 CAN-IO 扩展模块。

7、 CN15 辅助编码器/手轮接口定义

CN15 为辅助编码器/手轮接口，采用 DB9 母头。其引脚号和信号对应关系见表 7.6 所示。

表 7.6　接口 CN15 引脚号和信号关系表

序	名称	I/O	说　明
1	EA+	I	编码器/手轮输入
2	EB+	I	编码器/手轮输入
3	EZ+	I	编码器输入
4		-	NC

5	+5V	O	内部 5V 输出
6	EA-	I	编码器/手轮输入
7	EB-	I	编码器/手轮输入
8	EZ-	I	编码器输入
9	GND	O	内部 5V 地

8、 CN16 电源定义

接口 CN16 是接线盒的电源输入接口，板上标有 24V 的端子接+24，标有 EGND 的端子接外部电源地,标有 EARTH 的端子为机壳地接口。

9、 CN17 接口定义

CN17 接口为 0~3 轴电机的控制信号、IO 信号及低速手轮通道等与控制卡的接口。

10、CN18 接口定义

CN18 接口为 4~5 轴电机的控制信号及高速手轮通道与控制卡的接口。

11、指示灯定义

ACC2-3600 接线盒表面有 3 个指示灯，分别为：

POWER：24V 外部电源指示灯；

RUN：连接状态指示灯，绿色时表示接线盒与控制卡处于通讯状态；

ERR：错误指示灯，红色闪烁时表示接线盒与控制卡未连接成功；

附录 9　ACC2-XC00 接线盒接口说明

ACC2-XC00 接线盒是 DMC3C00-PCIe 运动控制卡的配套产品，扩展 DMC3C00-PCIe 控制卡的所有用户端子。

ACC2-XC00 接线盒尺寸图如图 9.1 所示。

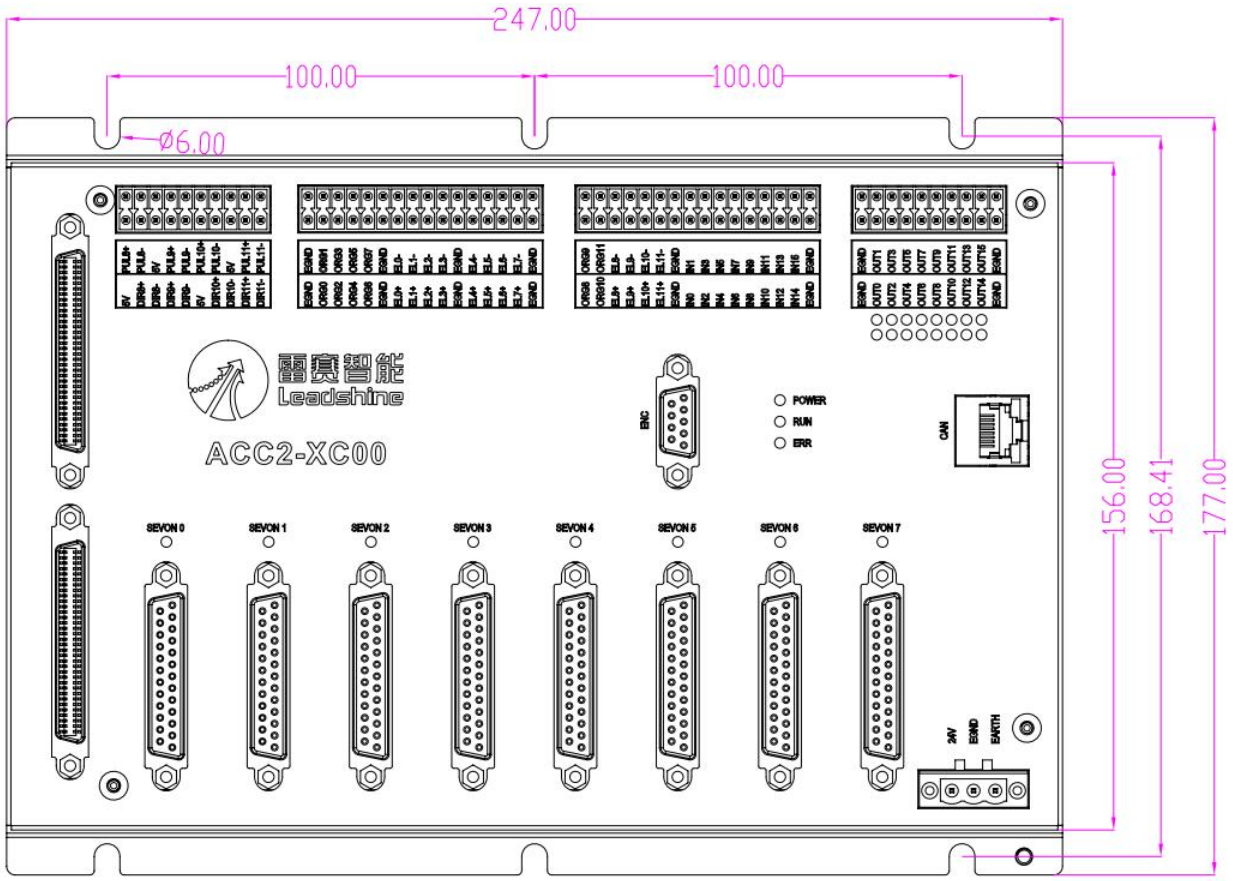


图 9.1　ACC2-XC00 尺寸图

ACC2-XC00 接线盒外形结构图如图 9.2 所示：

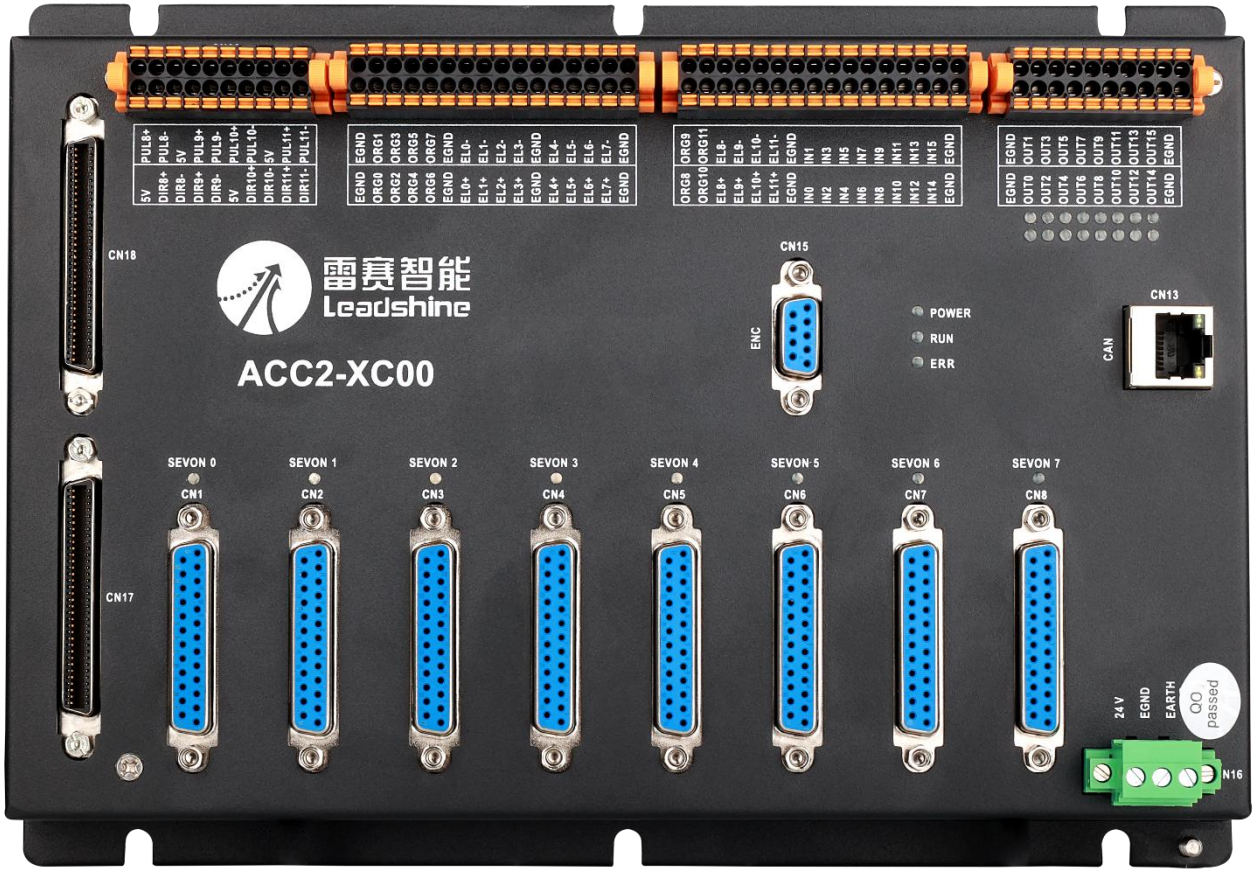


图 9.2　ACC2-XC00 接线盒外形结构图

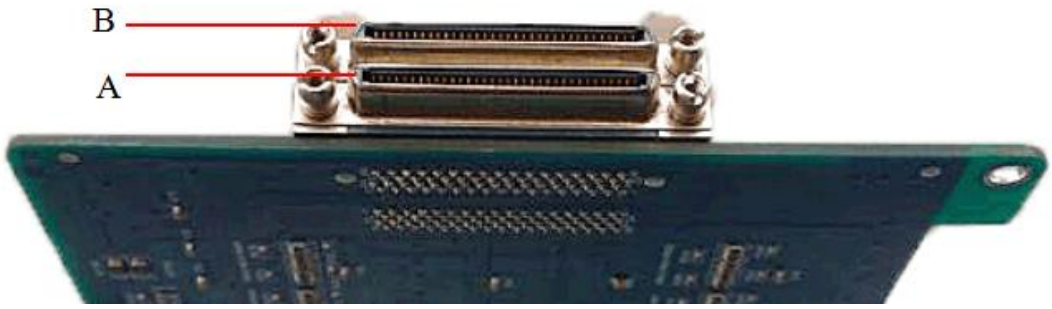


图 9.3　DMC5C10-PCIe侧视图

注　意：1）如图 9.3 所示，A为 0~3 轴的电机控制信号以及IO等信号端口，通过电缆线与ACC2-XC00 的CN17 端口相连；B为 4~11 轴电机的控制信号以及IO等信号端口，通过电缆线与ACC2-XC00 的CN18 端口相连。

2）DMC3C00-PCIe 与 ACC2-XC00 接线盒的连接示意图请参考第 3.2 节

1、 ACC2-XC00 接线盒接口及脚位

表 9.1 ACC2-XC00 接线盒接口功能简述

名称	功能介绍
CN1~CN8	第 1~8 轴轴控制信号
CN13	CAN 总线接口（RJ45）
CN15	辅助编码器接口
CN16	24V 直流电源输入端子
CN17	运动控制卡连接端子（1~4 轴）
CN18	运动控制卡连接端子（5~12 轴）
CN21~CN23	第 1~12 轴原点、限位以及通用 IO 信号接口
CN25	第 9~12 轴轴控制信号

2、 CN1~CN8 轴控制端子信号定义

1）ACC2-XC00 接线盒 CN1 至 CN8 以及 CN25 为电机控制信号端口，CN1 至 CN8 采用 DB25 母头，其引脚号和信号对应关系见表 9.2 所示。其中 24V 电源端口主要为控制卡的电机控制信号（伺服专用信号）供电。

2）电机控制信号端（CN1 至 CN8）的 24V 为控制信号电源，不能用于驱动器等动力负载供电。

3）根据 ACC2-XC00 接线盒的 PCB 板的铜线宽度与厚度，4 个电机控制信号端口的 24V 总输出电流需要控制在 1.5A 以内。

表 9.2 接口 CN1~CN8 引脚号和信号关系表

序	名称	I/O	说 明	序	名称	I/O	说 明
1	OGND	O	24V 电源地	14	24V	O	+24V 输出
2	ALM	I	驱动报警	15	ERC	O	驱动报警复位
3	SEVON	O	驱动允许	16	INP	I	到位信号
4	EA-	I	编码器输入	17	EA+	I	编码器输入
5	EB-	I	编码器输入	18	EB+	I	编码器输入
6	EZ-	I	编码器输入	19	EZ+	I	编码器输入
7	+5V	O	内部 5V	20	GND	O	内部 5V 地
8	保留	-	保留	21	GND	O	内部 5V 地
9	DIR+	O	方向输出	22	DIR-	O	方向输出
10	GND	O	内部 5V 地	23	PUL+	O	脉冲输出
11	PUL-	O	脉冲输出	24	GND	O	内部 5V 地
12	RDY	I	伺服准备完成	25	保留		保留

13	GND	O	内部 5V 地				

注意：（1）当使用+5V 和 PUL-端口时，则选择电机指令脉冲信号输出方式为单端输出；当使用 PUL+和 PUL-端口时，则选择电机指令脉冲信号输出方式为差分输出。

（2）ACC2-XC00 接线盒 1~6 轴编码器口仅支持差分接法，7~8 轴同时支持单端和差分接法

3、 CN13 IO 扩展接口定义

CN13 为 IO 扩展接口，采用 RJ45 接口，可连接 CAN-IO 扩展模块。

4、 CN15 辅助编码器/手轮接口定义

CN15 为辅助编码器/手轮接口，采用 DB9 母头。其引脚号和信号对应关系见表 9.3 所示。

表 9.3　接口 CN15 引脚号和信号关系表

序	名称	I/O	说　明
1	EA+	I	编码器/手轮输入
2	EB+	I	编码器/手轮输入
3	EZ+	I	编码器输入
4		-	NC
5	+5V	O	内部 5V 输出
6	EA-	I	编码器/手轮输入
7	EB-	I	编码器/手轮输入
8	EZ-	I	编码器输入
9	GND	O	内部 5V 地

5、 CN16 电源定义

接口 CN16 是接线盒的电源输入接口，板上标有 24V 的端子接+24，标有 OGND 的端子接外部电源地，标有 FG 的端子为机壳地接口。

表 9.4　接口 CN16 引脚号和信号关系表

序号	名称	功能
1	+24V	外部 24V 电源输入
2	EGND	外部 24V 电源地
3	EARTH	机壳地

6、 CN17 接口定义

CN17 接口为 0~3 轴电机的控制信号、IO 信号及低速手轮通道等与控制卡的接口。

7、 CN18 接口定义

CN18 接口为 4~11 轴电机的控制信号及高速手轮通道与控制卡的接口。

8、 CN21 专用输入接口定义

CN21 为数字量专用输入接口，其引脚号和信号对应关系见表 9.5 所示：

表 9.5　接口 CN21 引脚号和信号关系

序	名称	I/O	说明	序	名称	I/O	说明
1	EGND	O	24V 电源地	17	EGND	O	24V 电源地
2	ORG0	I	0 轴原点输入	18	ORG1	I	1 轴原点输入
3	ORG2	I	2 轴原点输入	19	ORG3	I	3 轴原点输入
4	ORG4	I	4 轴原点输入	20	ORG5	I	5 轴原点输入
5	ORG6	I	6 轴原点输入	21	ORG7	I	7 轴原点输入
6	EGND	O	24V 电源地	22	EGND	O	24V 电源地
7	EL0+	I	0 轴正限位输入	23	EL0-	I	0 轴负限位输入
8	EL1+	I	1 轴正限位输入	24	EL1-	I	1 轴负限位输入
9	EL2+	I	2 轴正限位输入	25	EL2-	I	2 轴负限位输入
10	EL3+	I	3 轴正限位输入	26	EL3-	I	3 轴负限位输入
11	EGND	O	24V 电源地	27	EGND	O	24V 电源地
12	EL4+	I	4 轴正限位输入	28	EL4-	I	4 轴负限位输入
13	EL5+	I	5 轴正限位输入	29	EL5-	I	5 轴负限位输入
14	EL6+	I	6 轴正限位输入	30	EL6-	I	6 轴负限位输入
15	EL7+	I	7 轴正限位输入	31	EL7-	I	7 轴负限位输入
16	EGND	O	24V 电源地	32	EGND	O	24V 电源地

9、 CN22 通用输入接口定义

CN22 为数字量通用输入和步进轴专用信号接口，其引脚号和信号对应关系见表 9.6 所示：

表 9.6　接口 CN21 引脚号和信号关系

序	名称	I/O	说　明	序	名称	I/O	说　明

1	ORG8	I	8 轴原点输入	17	ORG9	I	9 轴原点输入
2	ORG10	I	10 轴原点输入	18	ORG11	I	11 轴原点输入
3	EL8+	I	8 轴正限位输入	19	EL8-	I	8 轴负限位输入
4	EL9+	I	9 轴正限位输入	20	EL9-	I	9 轴负限位输入
5	EL10+	I	10 轴正限位输入	21	EL10-	I	10 轴负限位输入
6	EL11+	I	11 轴正限位输入	22	EL11-	I	11 轴负限位输入
7	EGND	O	24V 电源地	23	EGND	O	24V 电源地
8	IN0	I	通用输入（低速）	24	IN1	I	通用输入（低速）
9	IN2	I	通用输入（低速）	25	IN3	I	通用输入（低速）
10	IN4	I	通用输入（低速）	26	IN5	I	通用输入（低速）
11	IN6	I	通用输入（低速）	27	IN7	I	通用输入（低速）
12	IN8	I	通用输入（低速）	28	IN9	I	通用输入（低速）
13	IN10	I	通用输入（低速）	29	IN11	I	通用输入（低速）
14	IN12	I	通用输入（低速）	30	IN13	I	通用输入（低速）
15	IN14	I	通用输入/LTC0（高速）	31	IN15	I	通用输入/LTC0（高速）
16	EGND	O	24V 电源地	32	EGND	O	24V 电源地

10、CN23 数字量输出接口定义

CN23 为数字量输出接口，其引脚号和信号对应关系见表 9.7 所示。

表 9.7　接口 CN23 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	OUT0	O	通用输出（低速）	9	OUT1	O	通用输出（低速）
2	OUT2	O	通用输出（低速）	10	OUT3	O	通用输出（低速）
3	OUT4	O	通用输出（低速）	11	OUT5	O	通用输出（低速）
4	OUT6	O	通用输出（低速）	12	OUT7	O	通用输出（低速）
5	OUT8	O	通用输出（低速）	13	OUT9	O	通用输出（低速）
6	OUT10	O	通用输出（低速）	14	OUT11	O	通用输出（低速）
7	OUT12	O	通用输出/CMP0（高速）	15	OUT13	O	通用输出/CMP1（高速）
8	OUT14	O	通用输出/CMP2（高速）	16	OUT15	O	通用输出/CMP3（高速）

11、CN25 电机接口端子定义

CN25 为 8~11 轴电机控制信号端口，其引脚号和信号对应关系见表 9.8 所示。

表 9.8　接口 CN25 引脚号和信号关系表

序	名称	I/O	说　明	序	名称	I/O	说　明
1	5V	O	内部 5V 输出	11	PUL8+	O	轴 8 脉冲输出
2	DIR8+	O	轴 8 方向输出	12	PUL8-	O	轴 8 脉冲输出
3	DIR8-	O	轴 8 方向输出	13	5V	O	内部 5V 输出

4	DIR9+	O	轴 9 方向输出	14	PUL9+	O	轴 9 脉冲输出
5	DIR9-	O	轴 9 方向输出	15	PUL9-	O	轴 9 脉冲输出
6	5V	O	内部 5V 输出	16	PUL10+	O	轴 10 脉冲输出
7	DIR10+	O	轴 10 方向输出	17	PUL10-	O	轴 10 脉冲输出
8	DIR10-	O	轴 10 方向输出	18	5V	O	内部 5V 输出
9	DIR11+	O	轴 11 方向输出	19	PUL11+	O	轴 11 脉冲输出
10	DIR11-	O	轴 11 方向输出	20	PUL11-	O	轴 11 脉冲输出

12、指示灯定义

ACC2-3600 接线盒表面有 3 个指示灯，分别为：

POWER：24V 外部电源指示灯；

RUN：连接状态指示灯，绿色时表示接线盒与控制卡处于通讯状态；

ERR：错误指示灯，红色闪烁时表示接线盒与控制卡未连接成功；

附录 10　运动控制函数索引

函数分类	函数名	描 述	索引
板卡设置函数	dmc_board_init	控制卡初始化函数	8.1 节
	dmc_board_reset	控制卡硬件复位函数	
	dmc_board_close	控制卡关闭函数	
	dmc_get_CardInfList	获取控制卡硬件 ID 号	
	dmc_get_card_version	获取控制卡硬件版本号	
	dmc_get_card_soft_version	获取控制卡固件版本号	
	dmc_get_card_lib_version	获取控制卡动态库文件版本号	
	dmc_get_total_axes	获取当前卡的轴数	
	dmc_download_configfile	下载参数文件	
	dmc_download_firmware	下载固件文件	
脉冲模式设置函数	dmc_set_pulse_outmode	设置指定轴的脉冲输出模式	8.2 节
	dmc_get_pulse_outmode	读取指定轴的脉冲输出模式设置	
回原点运动函数	dmc_set_home_pin_logic	设置 ORG 原点信号	8.3 节
	dmc_get_home_pin_logic	读取 ORG 原点信号设置	
	dmc_set_homemode	设置回原点模式	
	dmc_get_homemode	读取回原点模式	
	dmc_home_move	回原点运动	
原点锁存函数	dmc_set_homelatch_mode	设置原点锁存模式	8.4 节
	dmc_get_homelatch_mode	读取原点锁存模式设置	
	dmc_reset_homelatch_flag	清除原点锁存标志	
	dmc_get_homelatch_flag	读取原点锁存标志	
	dmc_get_homelatch_value	读取原点锁存值	
限位开关设置函数	dmc_set_el_mode	设置 EL 限位信号	8.5 节
	dmc_get_el_mode	读取 EL 限位信号设置	
	dmc_set_softlimit	设置软限位	
	dmc_get_softlimit	读取软限位设置	
位置计数器控制函数	dmc_set_position	设置指令脉冲位置	8.6 节
	dmc_get_position	读取指令脉冲位置	
运动状态检测及控制函数	dmc_read_current_speed	读取当前速度值	8.7 节
	dmc_LinkState	检测主卡与接线盒的通讯连接状态	
	dmc_check_done	检测指定轴的运动状态	
	dmc_check_done_multicoor	检测坐标系的运动状态	
	dmc_axis_io_status	读取指定轴有关运动信号的状态	
	dmc_stop	指定轴停止运动	
	dmc_stop_multicoor	停止坐标系内所有轴的运动	
	dmc_emg_stop	紧急停止所有轴	
单轴运动速度曲线设置	dmc_set_profile	设置单轴运动速度曲线	8.8 节
	dmc_get_profile	读取单轴运动速度曲线	

函数分类	函数名	描 述	索引
函数	dmc_set_s_profile	设置单轴速度曲线 S 段参数值	
	dmc_get_s_profile	读取单轴速度曲线 S 段参数值	
单轴运动函数	dmc_pmove	指定轴点位运动	8.9 节
	dmc_vmove	指定轴连续运动	
	dmc_change_speed	在线变速	
	dmc_reset_target_position	在线变位	
	dmc_update_target_position	强行变位（在线/非在线）	
插补速度曲线设置函数	dmc_set_vector_profile_multicoor	设置插补速度	8.10 节
	dmc_get_vector_profile_multicoor	读取插补速度设置	
插补运动函数	dmc_line_multicoor	直线插补运动	8.11 节
	dmc_arc_move_multicoor	圆弧插补运动	
PVT 运动函数	dmc_PttTable	向指定数据表传送数据，采用 PTT 模式	8.12 节
	dmc_PtsTable	向指定数据表传送数据，采用 PTS 模式	
	dmc_PvtTable	向指定数据表传送数据，采用 PVT 模式	
	dmc_PvtsTable	向指定数据表传送数据，采用 PVTS 模式	
	dmc_PvtMove	启动 PVT 运动	
伺服驱动专用接口函数	dmc_write_sevon_pin	控制指定轴的伺服使能端口的输出	8.13 节
	dmc_read_sevon_pin	读取指定轴的伺服使能端口的电平状态	
	dmc_read_rdy_pin	读取指定轴的 RDY 端口的电平状态	
	dmc_set_inp_mode	设置指定轴的 INP 信号	
	dmc_get_inp_mode	读取指定轴的 INP 信号设置	
	dmc_set_alm_mode	设置指定轴的 ALM 信号	
	dmc_get_alm_mode	读取指定轴的 ALM 信号设置	
	dmc_write_erc_pin	控制指定轴的 ERC 信号输出	
	dmc_read_erc_pin	读取指定轴的 ERC 端口电平状态	
通用输入输出 IO 函数	dmc_read_inbit	读取指定控制卡的某一位输入口的电平状态	8.14 节
	dmc_write_outbit	设置指定控制卡的某一位输出口的电平状态	
	dmc_read_outbit	读取指定控制卡的某一位输出口的电平状态	
	dmc_read_inport	读取指定控制卡的全部输入口的电平状态	
	dmc_read_outport	读取指定控制卡的全部输出口的电平状态	
	dmc_write_outport	设置指定控制卡的全部输出口的电平状态	

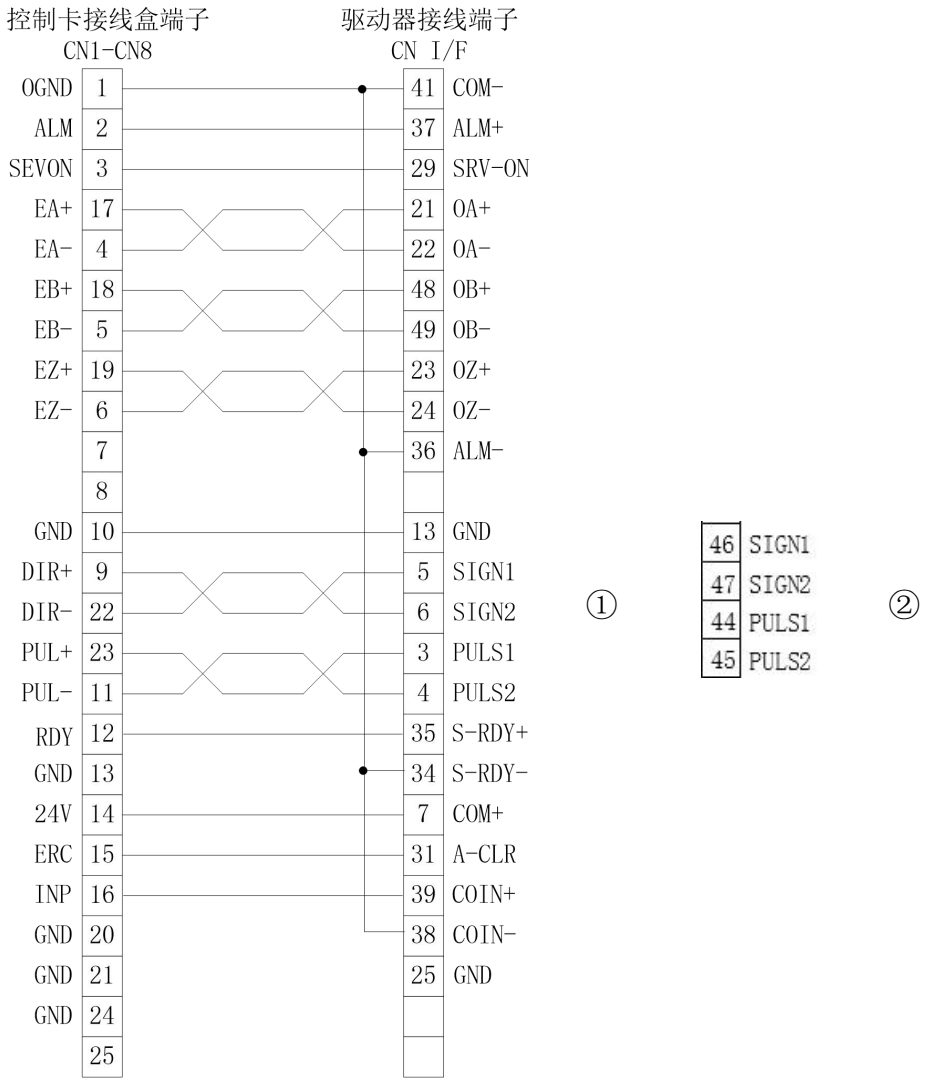
函数分类	函数名	描 述	索引
	dmc_reverse_outbit	IO 输出延时翻转	
	dmc_set_io_count_mode	设置 IO 计数模式	
	dmc_get_io_count_mode	读取 IO 计数模式设置	
	dmc_set_io_count_value	设置 IO 计数值	
	dmc_get_io_count_value	读取 IO 计数值	
手轮功能函数	dmc_set_handwheel_inmode	设置单轴手轮运动控制输入方式	8.15 节
	dmc_get_handwheel_inmode	读取单轴手轮运动控制输入方式	
	dmc_handwheel_move	启动手轮运动	
	dmc_set_handwheel_channel	手轮通道选择设置	
	dmc_get_handwheel_channel	读取手轮通道选择设置	
	dmc_set_handwheel_inmode_extern	设置多轴手轮运动控制输入方式	
	dmc_get_handwheel_inmode_extern	读取多轴手轮运动控制输入方式	
编码器函数	dmc_set_counter_inmode	设置编码器的计数方式	8.16 节
	dmc_get_counter_inmode	读取编码器的计数方式	
	dmc_set_encoder	设置指定轴编码器反馈位置脉冲计数值	
	dmc_get_encoder	读取指定轴编码器反馈位置脉冲计数值	
	dmc_set_ez_mode	设置指定轴的 EZ 信号	
	dmc_get_ez_mode	读取指定轴的 EZ 信号设置	
高速位置锁存函数	dmc_set_ltc_mode	设置指定轴的 LTC 信号	8.17 节
	dmc_get_ltc_mode	读取指定轴的 LTC 信号设置	
	dmc_set_latch_mode	设置锁存方式	
	dmc_get_latch_mode	读取锁存方式	
	dmc_set_latch_stop_time	设置锁存触发延时急停时间	
	dmc_get_latch_stop_time	读取锁存触发延时急停时间设置	
	dmc_SetLtcOutMode	LTC 反相输出设置	
	dmc_GetLtcOutMode	读取 LTC 反相输出设置	
	dmc_get_latch_value	从控制卡内读取编码器锁存器的值	
	dmc_get_latch_flag	从控制卡内读取指定卡内锁存器的标志位	
	dmc_reset_latch_flag	复位指定卡的锁存器的标志位	
	dmc_get_latch_flag_extern	从 PC 缓存中读取锁存器已锁存个数	
	dmc_get_latch_value_extern	按索引号读取 PC 缓冲区中已保存的锁存值	
位置比较函数	dmc_compare_set_config	设置一维位置比较器	8.18 节
	dmc_compare_get_config	读取一维位置比较器设置	
	dmc_compare_clear_points	清除一维位置比较点	
	dmc_compare_add_point	添加一维位置比较点	
	dmc_compare_get_current_point	读取当前一维比较点位置	
	dmc_compare_get_points_runned	查询已经比较过的一维比较点个数	
	dmc_compare_get_points_remained	查询可以加入的一维比较点个数	

函数分类	函数名	描 述	索引
	dmc_compare_set_config_extern	设置二维位置比较器	
	dmc_compare_get_config_extern	读取二维位置比较器设置	
	dmc_compare_clear_points_extern	清除二维位置比较点	
	dmc_compare_add_point_extern	添加二维位置比较点	
	dmc_compare_get_current_point_extern	读取当前二维位置比较点位置	
	dmc_compare_get_points_runned_extern	查询已经比较过的二维比较点个数	
	dmc_compare_get_points_remained_extern	查询可以加入的二维比较点个数	
高速位置比较函数	dmc_hcmp_set_mode	设置高速比较模式	8.19 节
	dmc_hcmp_get_mode	读取高速比较模式设置	
	dmc_hcmp_set_config	配置高速比较器	
	dmc_hcmp_get_config	读取高速比较器配置	
	dmc_hcmp_add_point	添加/更新高速比较位置	
	dmc_hcmp_set_liner	设置高速比较线性模式参数	
	dmc_hcmp_get_liner	读取高速比较线性模式参数设置	
	dmc_hcmp_clear_points	清除高速位置比较点	
	dmc_hcmp_get_current_state	读取高速比较参数	
	dmc_write_cmp_pin	控制指定 CMP 管脚的输出	
	dmc_read_cmp_pin	读取指定 CMP 管脚的电平状态	
异常信号接口函数	dmc_set_emg_mode	设置 EMG 急停信号	8.21 节
	dmc_get_emg_mode	读取 EMG 急停信号设置	
	dmc_set_io_dstp_mode	设置减速停止信号	
	dmc_get_io_dstp_mode	读取减速停止信号设置	
	dmc_set_dec_stop_time	设置减速停止时间	
	dmc_get_dec_stop_time	读取减速停止时间设置	
轴 IO 映射函数	dmc_set_axis_io_map	设置轴 IO 映射关系	8.22 节
	dmc_get_axis_io_map	读取轴 IO 映射关系设置	
	dmc_set_special_input_filter	设置所有专用 IO 滤波时间	
虚拟 IO 映射函数	dmc_set_io_map_virtual	设置虚拟 IO 映射关系	8.23 节
	dmc_get_io_map_virtual	读取虚拟 IO 映射关系设置	
	dmc_read_inbit_virtual	读取滤波后的虚拟 IO 口电平状态	
检测轴到位状态函数	dmc_set_factor_error	设置位置误差带	8.24 节
	dmc_get_factor_error	读取位置误差带设置	
	dmc_check_success_pulse	检测指令到位	
	dmc_check_success_encoder	检测编码器到位	
	nmc_set_can_state	设置 CAN-IO 通讯状态	8.25 节
	nmc_get_can_state	读取 CAN-IO 通讯状态	
	nmc_write_outbit	设置指定 CAN-IO 扩展模块的某个输出端口的电平	

函数分类	函数名	描 述	索引
CAN-IO 扩展函数	nmc_read_outbit	读取指定 CAN-IO 扩展模块的某个输出端口的电平	
	nmc_read_inbit	读取指定 CAN-IO 扩展模块的某个输入端口的电平	
	nmc_write_outport	设置指定 CAN-IO 扩展模块的全部输出口的电平	
	nmc_read_outport	读取指定 CAN-IO 扩展模块的全部输出口的电平 A	
	nmc_read_inport	读取指定 CAN-IO 扩展模块的全部输入端口的电平	
AD/DA 函数	dmc_set_da_enable	设置 DA 输出使能	8.26 节
	dmc_get_da_enable	读取置 DA 输出使能设置	
	dmc_set_da_output	设置 DA 输出	
	dmc_get_da_output	设置 DA 输出	
	dmc_get_da_input	读取 AD 输入	
密码管理函数	dmc_write_sn	修改密码	8.29 节
	dmc_check_sn	密码校验	
打印输出函数	dmc_set_debug_mode	函数调用打印输出设置	8.30 节
	dmc_get_debug_mode	读取函数调用打印输出设置	

附录 11　控制卡和主流驱动器电气接线图

1、控制卡与松下 Panasonic MSDA 系列驱动器接线

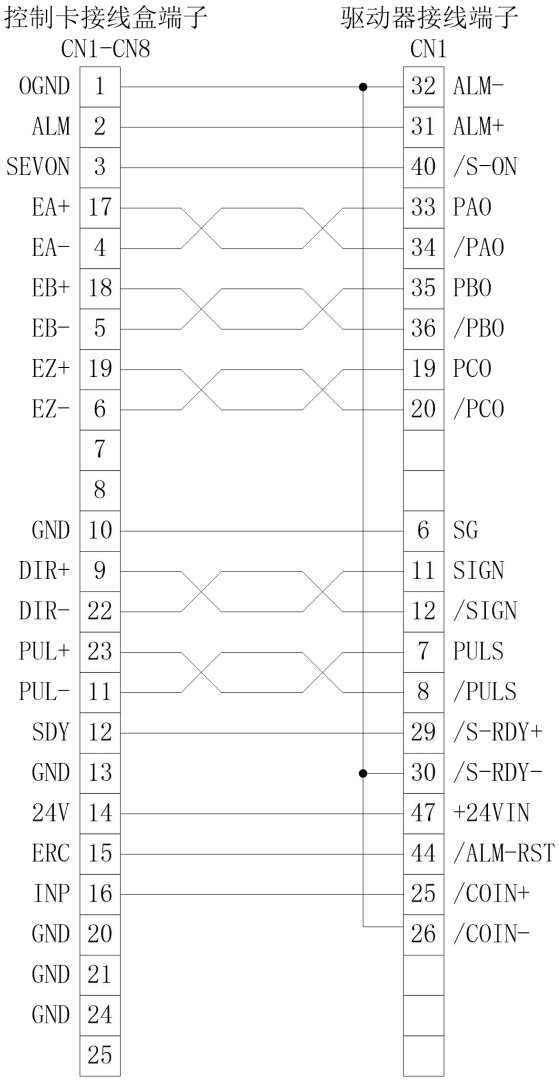


F7.1　轴端口与松下 Panasonic MSDA 系列驱动器接线

①号脉冲信号线相关接法为集电极开路接法，脉冲频率需在 500KHZ 以下；②号脉冲信号线相关接法为差分电路接法，脉冲频率可支持到 2MHZ；

注意：控制卡脉冲和编码器信号数字地 GND 需与驱动器的地 GND 连接，控制卡的 24V 地 OGND 需与驱动器的 24V 地 COM-连接。

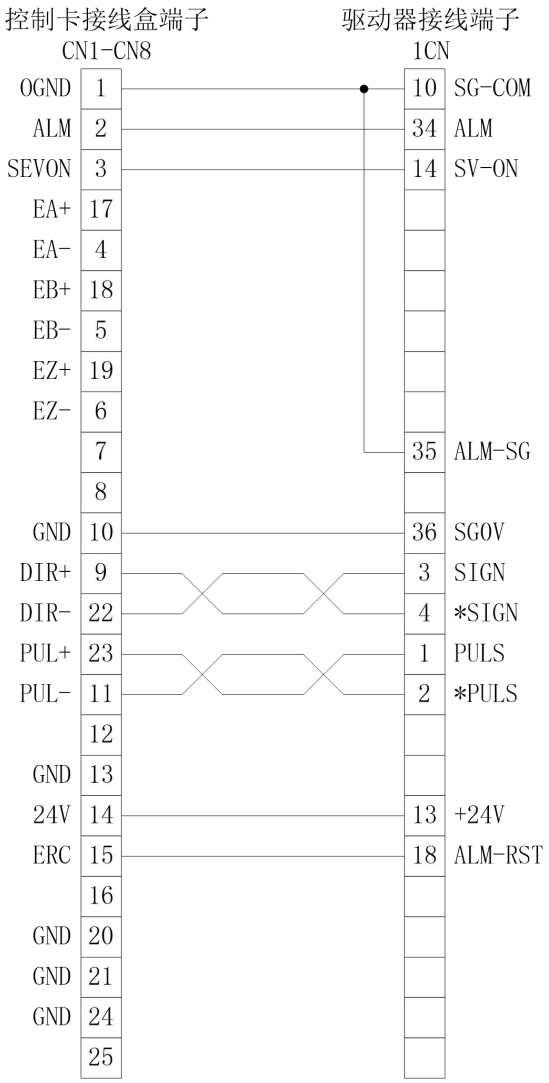
2、控制卡与安川 YASKAWA SGDM 系列驱动器接线



F7.2　轴端口与安川 YASKAWA SGDM 系列驱动器接线

注意：控制卡脉冲和编码器信号数字地 GND 需与驱动器的地 SG 连接，控制卡的 24V 地 OGND 需与驱动器的 24V 地 ALM-、/S-RDY-、/COIN-连接。

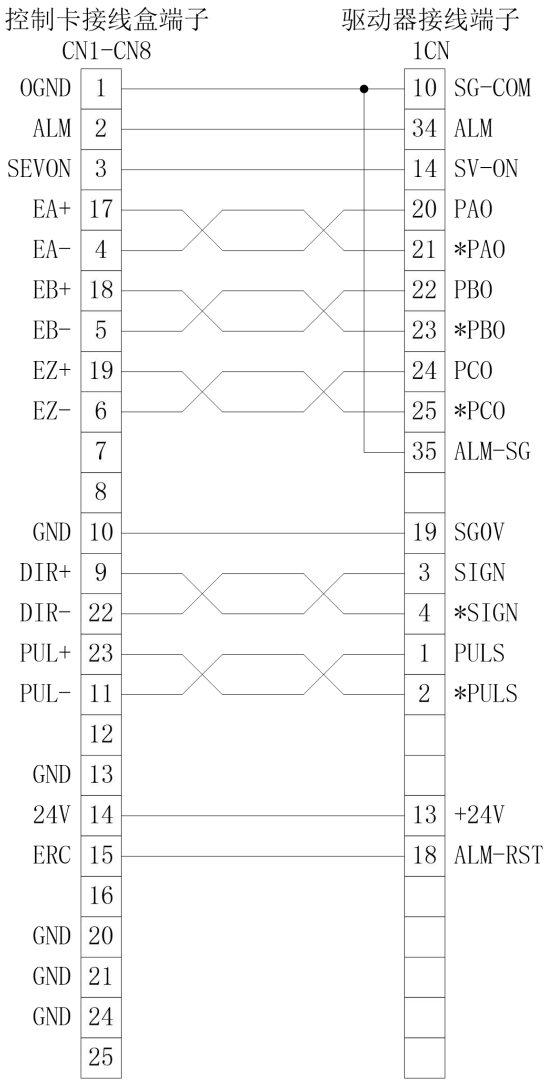
3、控制卡与安川 YASKAWA SGDE 系列驱动器接线



F7.3　轴端口与安川 YASKAWA SGDE 系列驱动器接线

注意：控制卡脉冲和编码器信号数字地 GND 需与驱动器的地 SG 0V 连接，控制卡的 24V 地 OGND 需与驱动器的 24V 地 SG-COM 连接。

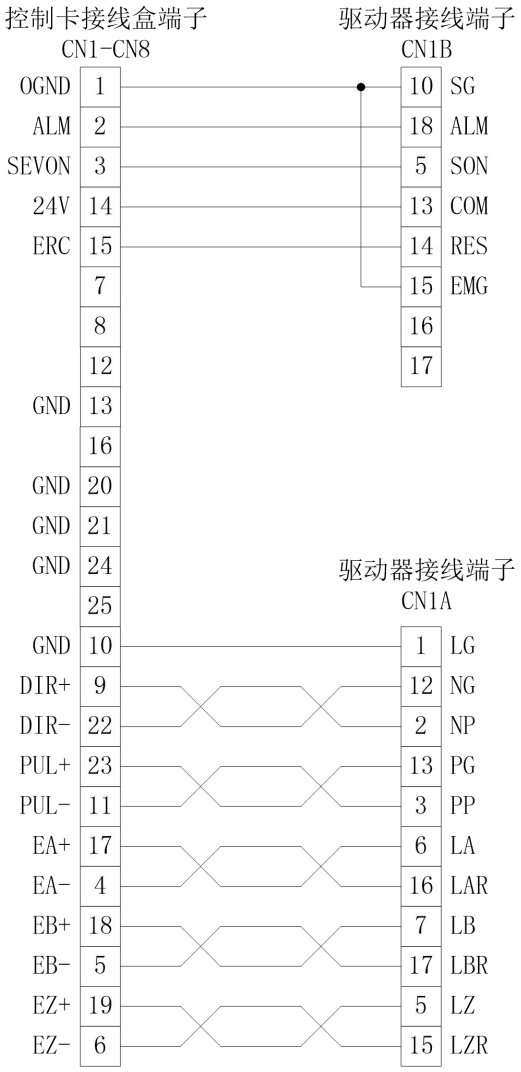
4、控制卡与安川 YASKAWA SERVOPACK 系列驱动器接线



F7.4　轴端口与安川 YASKAWA SERVOPACK　系列驱动器接线

注意：控制卡脉冲和编码器信号数字地 GND 需与驱动器的地 SG 0V 连接，控制卡的 24V 地 OGND 需与驱动器的 24V 地 SG-COM 连接。

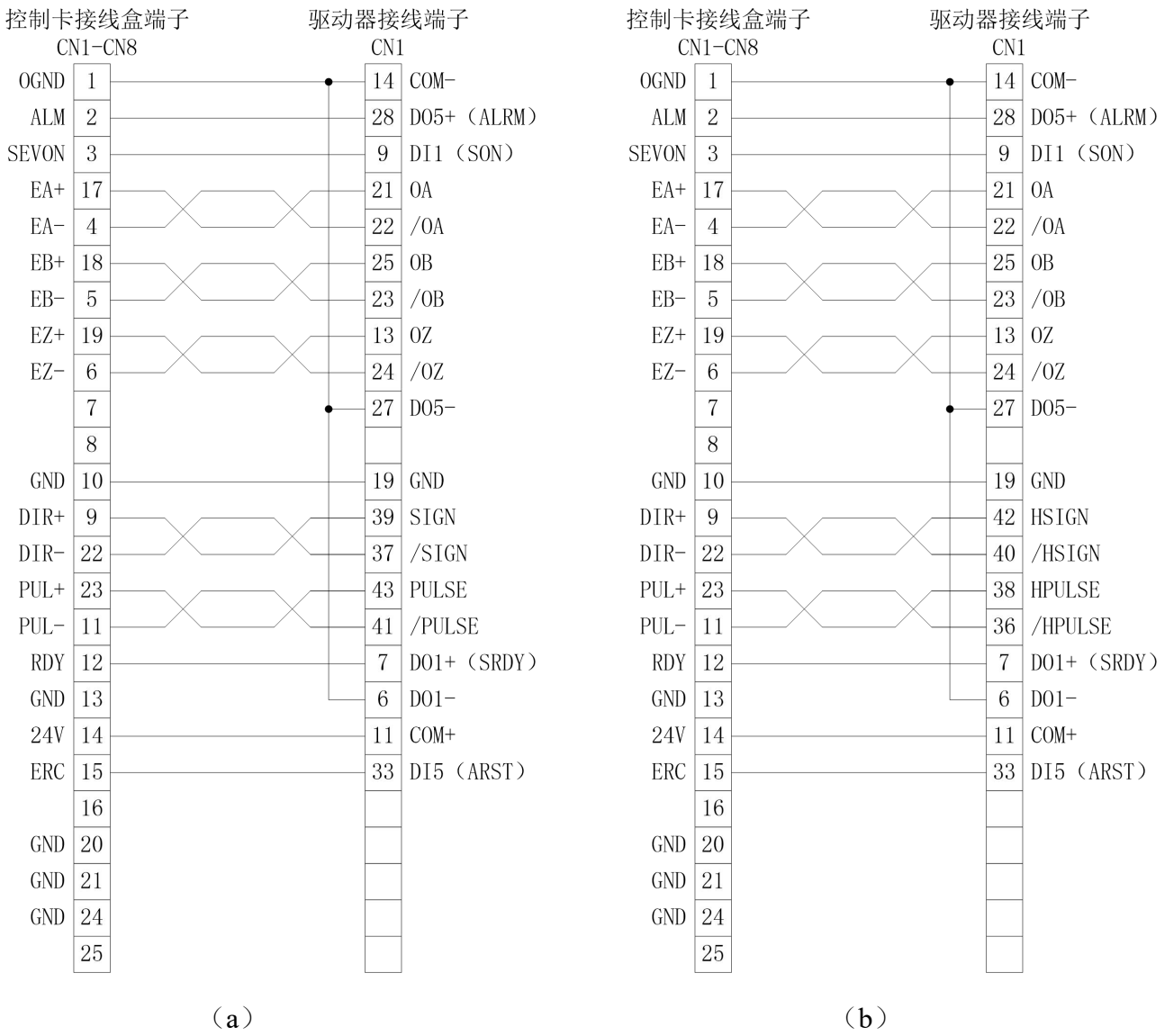
5、控制卡与三菱 MELSERVO-J2-Super 系列驱动器接线



F7.5　轴端口与三菱 MELSERVO-J2-Super 系列驱动器接线

注意：控制卡脉冲和编码器信号数字地 GND 需与驱动器的地 LG 连接，控制卡的 24V 地 OGND 需与驱动器的 24V 地 SG 连接。

6、控制卡与台达 ASDA-B2 系列驱动器接线



（a） （b）

F7.6 轴端口与台达 ASDA-B2 系列驱动器接线

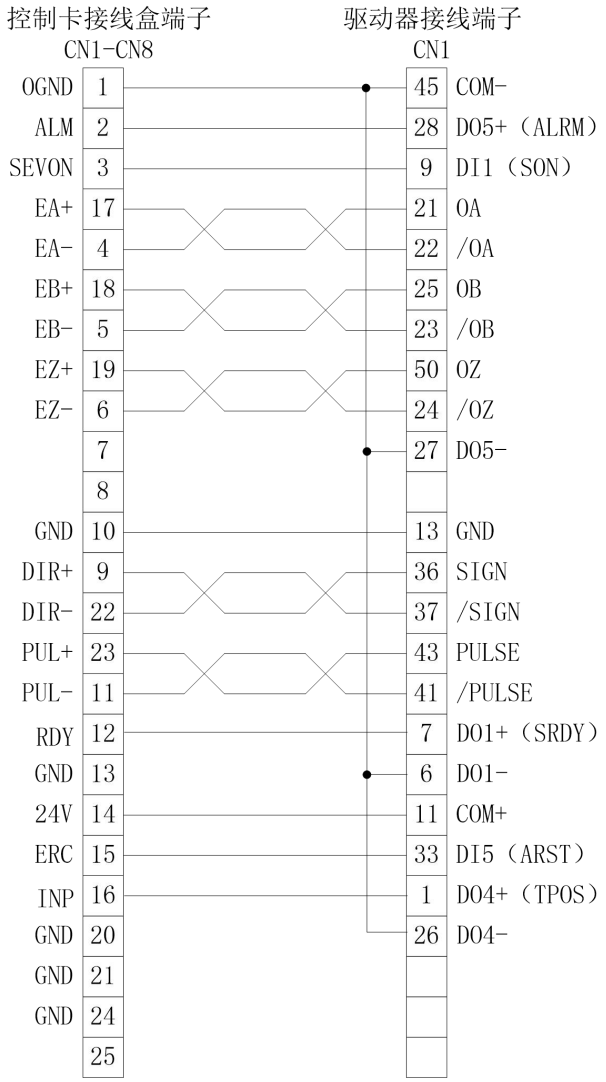
注意：

1） 当脉冲频率在 500KHZ 以下时，PULSE+，PULSE-，DIR+，DIR-可接在驱动器端 43，41，39，37 引脚上，如上图（a）所示；

2） 当脉冲频率在 500KHZ~4MHZ 时，PULSE+，PULSE-，DIR+，DIR-可接在驱动器端 38，36，42，40 引脚上，如上图（b）所示；

3） 控制卡脉冲和编码器信号数字地 GND 需与驱动器的地 GND 连接，控制卡的 24V 地 OGND 需与驱动器的 24V 地 COM-连接。

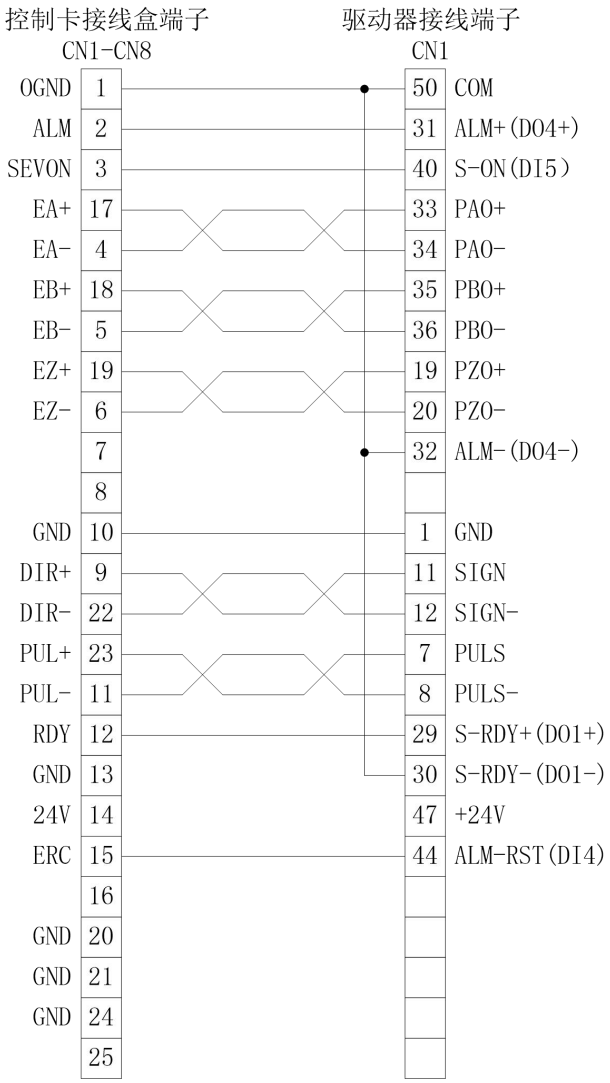
7、控制卡与台达 ASDA-AB 系列驱动器接线



F7.7　轴端口与台达 ASDA-AB 系列驱动器接线

注意：控制卡脉冲和编码器信号数字地 GND 需与驱动器的地 GND 连接，控制卡的 24V 地 OGND 需与驱动器的 24V 地 COM-连接。

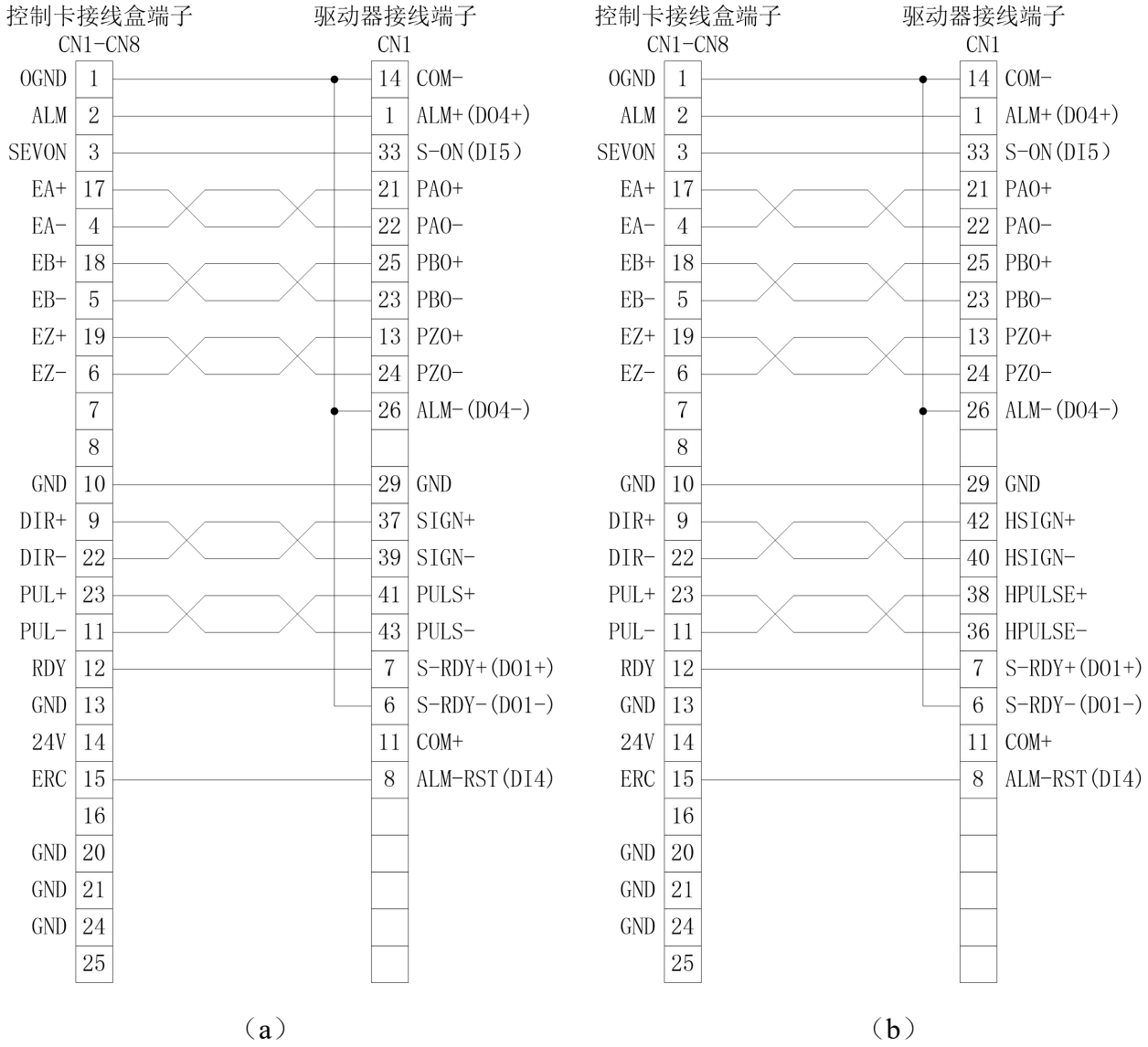
8、控制卡与汇川 IS500 系列驱动器接线



F7.8　轴端口与汇川 IS500 系列驱动器接线

注意：控制卡脉冲和编码器信号数字地 GND 需与驱动器的地 GND 连接，控制卡的 24V 地 OGND 需与驱动器的 24V 地 COM 连接。

9、控制卡与汇川 IS650P 系列驱动器接线



F7.9　轴端口与汇川 IS650P 系列驱动器接线

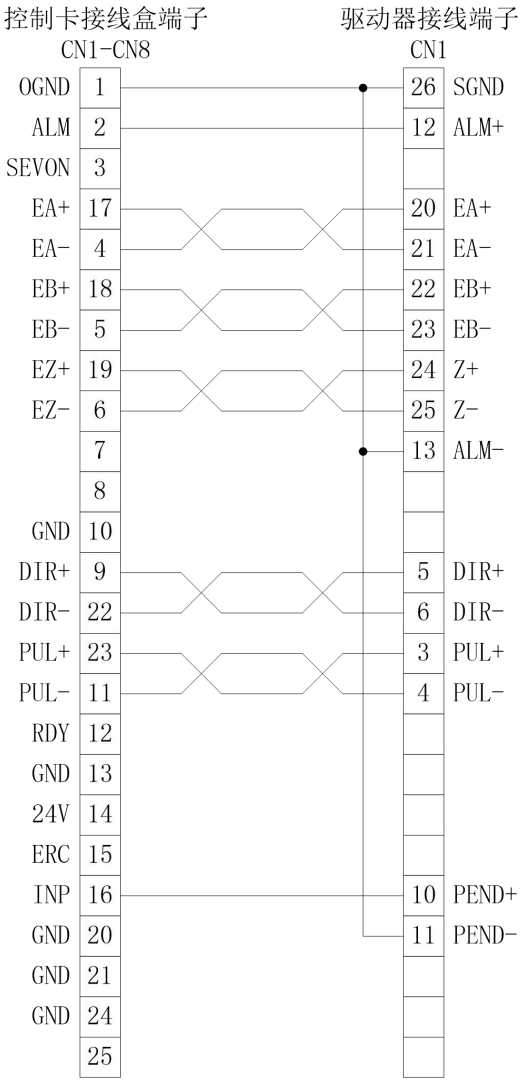
注意：

1）当脉冲频率在 500KHZ 以下时，PULSE+，PULSE-，DIR+，DIR-可接在驱动器端 41，43，37，39 引脚上，如上图（a）所示；

2）当脉冲频率在 500KHZ~4MHZ 时，PULSE+，PULSE-，DIR+，DIR-可接在驱动器端 38，36，42，40 引脚上，如上图（b）所示；

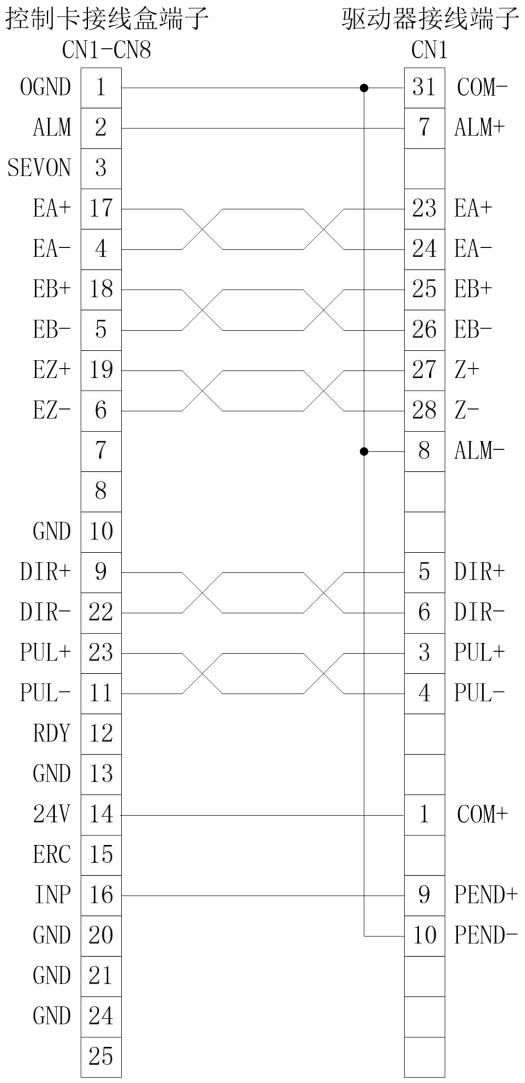
3）控制卡脉冲和编码器信号数字地 GND 需与驱动器的地 GND 连接，控制卡的 24V 地 OGND 需与驱动器的 24V 地 COM-连接。

10、控制卡与雷赛 H2-506 驱动器接线



F7.10　轴端口与雷赛 H2-506 驱动器接线

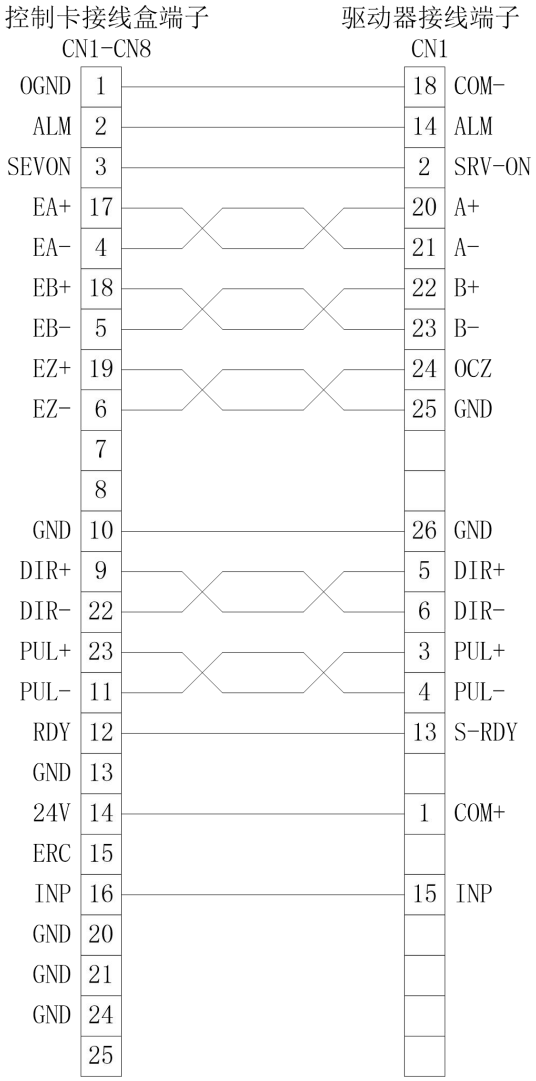
11、控制卡与雷赛 H2-758/1108/2206 驱动器接线



F7.11　轴端口与雷赛 H2-758/1108/2206 驱动器接线

注意：控制卡的 24V 地 OGND 需与驱动器的 24V 地 COM-连接。

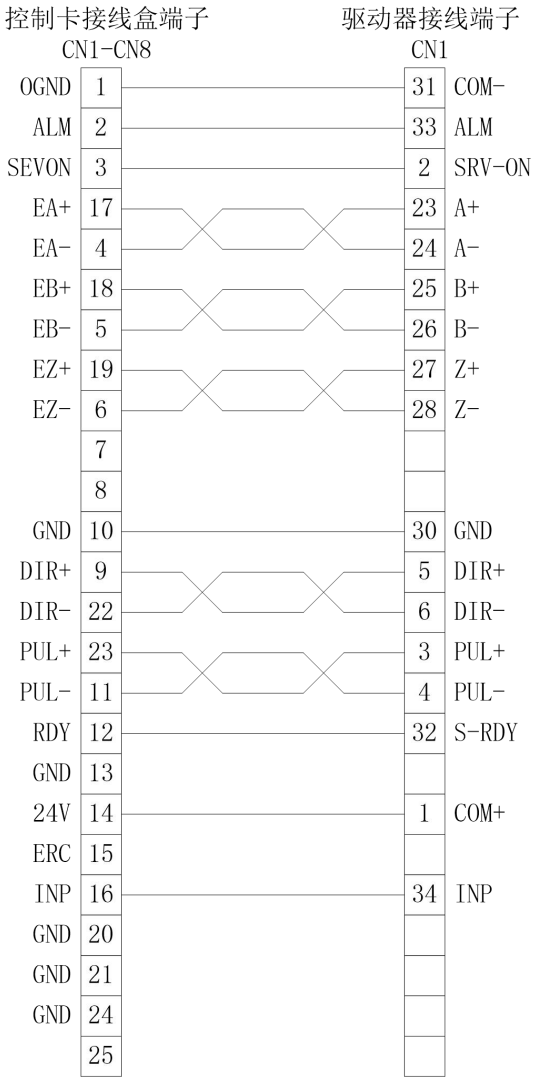
12、控制卡与雷赛 LD5 系列驱动器接线



F7.12　轴端口与雷赛 LD5 系列驱动器接线

注意：控制卡脉冲和编码器信号数字地 GND 需与驱动器的地 GND 连接，控制卡的 24V 地 OGND 需与驱动器的 24V 地 COM-连接。

13、控制卡与雷赛 L5Z 系列驱动器接线



F7.13　轴端口与雷赛 L5Z 系列驱动器接线

注意：控制卡脉冲和编码器信号地 GND 需与驱动器的地 GND 连接，控制卡的 24V 地 OGND 需与驱动器的 24V 地 COM-连接。

14、接线注意事项

1、布线建议

（1）不要将 110V/220V 交流电源电缆与直流 24V、IO 信号线缆、通信线缆等捆扎在一起，在空间允许范围内保持较远距离；

（2）不同类的电缆（如通讯信号线、电源线、数字 IO 信号线）布线时要分开，一般不能交叉重叠，当不可避免交叉时，应以直角交叉；

（3）高速 I/O、模拟量 I/O、现场总线、通信信号的电缆使用屏蔽线缆。

2、接线注意事项

（1）在安装及配线过程中，一定要确保外部电源处于关闭状态，防止触电及模块损坏；

（2）在连接端子处应扭绞导线，并以较短的长度接入端子，防止螺丝松动等情况下造成短路；

（3）电源接通后，24V 指示灯亮表明电源处于工作状态，如不亮，请考虑电源输入异常及模块故障可能。

3、接地处理

（1）ACC 系列接线盒，电源端子 24V 引脚与外部直流电源的正极连接。电源端子 OGND 引脚为 24V 电源输入地，接外部直流电源的负极。FG 为机壳地，应就近与机箱外壳相连。

（2）接地线应使用粗线，保证接地阻抗较小。

（3）ACC 系列接线盒上标记为 OGND 的引脚均为 24V 地，与电源端子 OGND 相通。标记为 GND 的引脚为 5V 信号地。

4、控制端口接线注意事项

（1）EtherCAT 和 Rtex 总线端口

主从站连接的网线应使用带屏蔽的五类双绞网线，RI45 接头带金属屏蔽。主从站的总线端口有输入和输出之分，不可接反。

（2）CAN 扩展接口

CAN 扩展总线上的所有从站波特率应保持一致，每个从站的节点号应不同。总线上的最后一个从站应接通终端电阻，中间的从站终端电阻不接通。

（3）编码器输入和脉冲输出端口

编码器输入和脉冲输出提供了差分接口，所以推荐用户以差分方式接线，差分信号使用屏蔽双绞线连接。差分信号两端数字地务必连通，即驱动器脉冲输入信号地与编码器输出信号地需与控制卡接线盒轴端口上的 GND 连接。

（4）手轮输入

控制卡手轮接口提供 5V 电源输出可作为手轮的电源输入，手轮的信号地需与手轮接口的 GND 地信号相连。

（5）数字输入输出

数字输入设备一端连接输入口一端连接到接线盒的 OGND（24V 地）端口。输出电路最大电流不能超过输出口最大工作电流，输出器件的地线需连接到接线盒的 OGND（24V 地）。输出口接感性负载时，需并联续流二极管。不允许将 24V 电源直接接到数字输出端口。

输入输出扩展电缆布线时，避免与动力线（高电压，大电流）等传输强干扰信号的电缆捆在一起，应该分开走线并且避免平行走线。

高速 IO 接口扩展电缆的总延长距离应该在 3.0m 以内。

（6）模拟量信号接口

模拟量信号连接时使用带屏蔽的线缆，固定线缆时不要将线缆与交流线缆、高压线缆等捆扎在一起，以减小噪声、电涌及感应的影响。

（7）PWM 信号接口

DMC3600、DMC3800、DMC3C00 卡的 PWM 信号与最后一个轴的脉冲输出信号引脚复用，PWM 输出 5V 信号，接线参照脉冲输出信号接线。

DMC3400A 的 PWM 输出功能需配合 ACC-X400B 接线盒使用，PWM 输出 24V 信号，PWM 接收设备地线与控制卡 OGND 连接。

附录 12　常见问题解决方法

出现问题	解决建议
板卡插上后，PC 机系统还不能识别控制卡	检查板卡驱动是否正确安装，在 WINDOWS 的设备管理器（可参看 WINDOWS 帮助文件）中查看驱动程序安装是否正常。如果发现有相关的黄色感叹号标志，说明安装不正确，需要按照软件部分安装指引，重新安装； 计算机主板兼容性差，请咨询主板供应商； PCI 插槽是否完好； PCI 金手指是否有异物，可用酒精清洗。
PC 机不能和控制卡通讯	PCI 金手指是否有异物，可用酒精清洗； 参考软件手册检查应用软件是否编写正确。
板卡和驱动器电机连接后，发出脉冲时，电机不转动	板卡上的设置脉冲发送方式和驱动器的输入脉冲方式是否匹配，跳线 J1－J8 是否正确； 可以用 Motion3000 演示软件进行测试，观察脉冲计数等是否正常； 是否已经接上供给脉冲和方向的外部电源。
控制卡已经正常工作，正常发出脉冲，但电机不转动	检查驱动器和电机之间的连接是否正确。可以使用 Motion3000 演示软件进行测试。 确保驱动器工作正常，没有出现报警。
电机可以转动，但工作不正常	检查控制卡和驱动器是否正确接地，抗干扰措施是否做好； 脉冲和方向信号输出端光电隔离电路中使用的限流电阻过大，工作电流偏小。
能够控制电机，但电机出现振荡或是过冲	可能是驱动器参数设置不当，检查驱动器参数设置； 应用软件中加减速时间和运动速度设置不合理。
能够控制电机，但工作时，回原点定位不准	检查屏蔽线是否接地； 原点信号开关是否工作正常； 所有编码信号和原点信号是否受到干扰。
限位信号不起作用	限位传感器工作不正常； 限位传感器信号受干扰； 应用程序紊乱。
不能读入编码器信号	请检查编码器信号类型是否是脉冲 TTL 方波； 参看所选编码器说明书，检查接线是否正确； 编码器供电是否正常； 检查函数调用是否正确。
对编码器的读数不准确	检查全部编码器及触发源的接线； 做好信号线的接地屏蔽。
不能锁存编码器读数	检查触发源的接线； 检查函数的调用是否正确。
锁存数据的重复精度差	检查函数调用； 程序中是否进行了去抖动处理； 触发信号的设定。
数字输入信号不能读取	接线是否正常；检查函数调用。
数字输出信号不正常	接线是否正常；检查函数调用。

深圳市雷赛控制技术有限公司
SHENZHEN LEADSHINE CONTROL TECHNOLOGY CO.,LTD

深圳市雷赛控制技术有限公司
地　址：深圳市南山区沙河西路 3157 号南山智谷产业园 B 栋 15-20 楼
邮　编：518055
电　话：0755-26415968
传　真：0755-26417609
Email：info@szleadtech.com.cn
网　址：http://www.szleadtech.com.cn