目录

第一章　通用说明

1.1 产品简介

Kinco-KS 系列 PLC 属于小型一体化 PLC，是步科公司推出的高性能薄片型产品。

KS 系列 PLC 在延续 K5/K2 系列功能丰富、高性能、高可靠性的前提下，采用了性能更高的 CPU，更提供了本体自带 CAN 总线接口、更高性能的高速输入/输出、紧凑型安装、丰富的扩展模块等更贴近用户需求的功能，能满足用户多种应用需求。

1.2 产品列表

名 称	订货号	功能描述
CPU 模块		
CPU101	KS101M-04DX	DC24V 供电，DI 4*DC24V，1*MicroUSB 编程口，1 个以太网口 通信口：1*RS232 通信口，1*RS485，2*CAN 接口 其中，CAN1 口支持扩展协议（可接扩展模块--最多可带 14 个扩展模块）、自由通信；CAN2 口支持 Kinco 运动控制指令、CANOpen 主站、自由通信
CPU105	KS105-16DT	DC24V 供电，DI 8*DC24V，DO 8*DC24V 通信口：1*RS232 通信口，1*RS485。 扩展功能：支持，最多可带 14 个扩展模块。
	KS105C1-16DT	DC24V 供电，DI 8*DC24V，DO 8*DC24V 通信口：1*RS232 通信口，1*RS485，1*CAN 接口（Kinco 运动控制指令、支持 CANOpen 主站，CANopen 从站，CAN 自由通信功能）。
	KS105C2-16DT	DC24V 供电，DI 8*DC24V，DO 8*DC24V 通信口：1*RS232 通信口，1*RS485，2*CAN 接口 其中，CAN1 口支持扩展协议（可接扩展模块--最多可带 14 个扩展模块）、自由通信；CAN2 口支持 Kinco 运动控制指令、CANOpen 主站、自由通信。

扩展模块		
PM121	KS121-16DX	DC24V 供电，DI 16*DC24V
PM122	KS122-12XR	DC24V 供电，DO 12*继电器
PM122	KS122-14DT	DC24V 供电，DO 14*晶体管
PM123	KS123-14DR	DC24V 供电，DI 8*DC24V，DO 6*继电器
PM131	KS131-04RD	4 通道热电阻输入，两线、三线制或四线制，PT100、PT1000、Cu50、R
PM133	KS133-06IV	DC24V 供电，4 通道模拟量输入/2 通道模拟量输出 可选 4-20mA/1-5V/0-20mA/0-10V

1.3 正常的工作条件

Kinco-KS 的设计符合 GB/T 15969.3-2007（idt IEC61131-2：2007）标准和测试规范。

下表简要描述了 Kinco-KS 正常的工作条件。使用时用户必须保证不超出表中规定的 PLC 工作条件。

运输和存储		
气候条件	环境温度	-40℃～+70℃ 。
	相对湿度	10%～95%，无凝露。
	大气压	相当于 0～3000 米海拔高度。
机械条件	自由跌落	带运输包装，允许从 1 米高度 5 次跌落于水泥地面。
工作条件		
气候条件	环境温度	自然通风的开放式装置，环境气温-10～55℃。
	相对湿度	10%～95%，无凝露。
	大气压	海拔高度不超过 2000 米
	污染等级	适用于污染等级 2。
机械条件	正弦振动	5<f<8.4Hz ，随机: 3.5mm 位移；连续: 1.75mm 位移。 8.4<f<150，随机: 1.0g 加速度；连续: 0.5g 加速度。
	冲击	半正弦波、15g、11ms，每轴向 6 次。
电磁兼容性	静电放电	空气放电 8KV，接触放电 4KV。性能等级 B。

(EMC)	浪涌	交流电源 2KV CM，1KV DM；直流电源 0.5KV CM，0.5KV DM；IO 和通信口：1KVCM。 性能等级 A。
	快速瞬变脉冲群	电源耦合 2KV，5KHz；IO 及通信耦合 1KV，5KHz。 性能等级 A。
	电压跌落	交流系统，50Hz 时，电压 0%持续 1 周波，40%持续 10 周波，75%持续 20 周波。 性能等级 A。
防护等级	防水防尘	IP20。

第二章 CPU 模块介绍

2.1 概述

2.1.1 部件结构图

2.1.2 CPU 类型

Kinco-KS 提供了多种规格的 CPU，供电电源均采用 DC 24V 的供电电压。

下表描述了各种类型 CPU 的主要技术参数。

参数	KS105-16DT	KS105C1-16DT	KS105C2-16DT
供电电源			
额定供电电源	DC24V		
供电电压范围	DC20.4V－28.8V		
额定功率	5W		
I/O 及通信口			
本体开关量	8*DI/ 8*DO（晶体管，源型，输出高电平）		
本体模拟量	--		
最大扩展模块	14	---	14
CAN 接口	---	有，支持 Kinco 运动控制指令、CANOpen 主站，CANopen 从站，CAN 自由通信功能	有，支持 Kinco 运动控制指令、CANOpen 主站、CAN 自由通信功能
串行通信口	PORT0 为 RS232 接口，支持编程协议、Modbus RTU 从站、自由通信； PORT1 为 RS485 接口，支持编程协议、Modbus RTU 主从站、自由通信。		
高速输入	4 路，最高频率全部高达 200KHz,支持单相和 AB 相计数。		
高速输出	4 通道 0、1 和 2 最高输出频率 200KHz（要求负载电阻不大于 1.5KΩ）。 通道 3 最高输出频率 10KHz。		
边沿中断	4 路，I0.0-I0.3 可分别设置为上升沿或者下降沿中断。		
存储区域			
用户程序	最大 4K 条指令		
用户数据	M 区 1K 字节；V 区 4K 字节。		
数据备份	E2PROM，448 字节，永久存储。		
数据保持	V 区 1K 字节（VB0-VB1023）。锂电池，常温下 3 年		

其它	
定时器	256 1ms 时基：4 10ms 时基：16 100ms 时基：236
定时中断	2 个，0.1ms 时基。
计数器	256 个
实时时钟	有， 在 25℃时误差小于 5 分钟/月

参数	KS101M-04DX
供电电源	
额定供电电源	DC24V。备注：USB 口也可以直接供电供 CPU 运行。
供电电压范围	DC20.4V－28.8V
额定功率	5W
I/O 及通信口	
本体开关量	4*DI
最大扩展模块	14
编程口	USB2.0，采用 micro USB 接口形式。
以太网口	1 个，支持编程协议、Modbus TCP Server（即从站）
CAN 接口	2 个 CAN 接口 CAN1 支持扩展协议、CAN 自由通信功能 CAN2 支持 CANOpen 主站、Kinco 运动控制指令和 CAN 自由通信功能。
串行通信口	1 个 RS232 和 1 个 RS485 串行通信口，分别命名为 PORT0、PORT1，通信波特率最高为 115.2Kbps。 PORT0 口既可用作编程口，也支持 Modbus RTU 从站协议和自由通信。 PORT1 口既可用作编程口，也支持 Modbus RTU 主站、从站协议和自由通信。
高速输入 单相 双相	2 最高计数频率：200KHz 最高计数频率：200KHz
存储区域	
用户程序	最大 8K 指令

用户数据	M 区 4K 字节；V 区 16K 字节
数据备份	E2PROM，V 区最后的 1K 字节，永久存储。
数据保持	全部 V 区。锂电池，常温下 3 年
其它	
定时器	256 个
定时中断	2 个，0.1ms 时基。
计数器	256 个
实时时钟	有， 在 25℃时误差小于 5 分钟/月

2.2 各部分功能介绍

2.2.1 CPU 状态及指示灯

CPU 模块有两种状态：运行状态和停止状态。

在运行状态下，CPU 模块正常地循环执行主扫描任务和各种中断任务。在停止状态下，CPU 模块仅处理部分通信请求(包括来自于 KincoBuilder 编程软件的编程、调试等命令，以及作为 Modbus RTU 从站响应主站的通信命令)，同时将所有的输出点（DO、AO）立即输出用户工程的【硬件配置】中定义的“停机输出”值。

- 改变 CPU 状态

PLC 的实际运行状态有两种控制方式：软件启动/停止控制和硬件拨码开关控制。

一般情况下，PLC 正常上电并联机，PLC 的状态由拨码开关和软件【调试】中启动/停止操作组合控制：拨码开关和软件设定都处于 RUN 时，PLC 的实际状态为 RUN，其他操作组合方式下 PLC 的实际运行状态都为 STOP。

以下情形，PLC 的运行状态由单个控制方式决定：

（1）PLC 的运行状态只取决于软件中 RUN/STOP 的设定, 而拨码开关所在位置不生效的情况

a.PLC 运行出错（严重错误、致命错误）导致 PLC 停止；

b.用户使用编程软件菜单中的【调试】操作使 PLC 进入了停止或运行状态；

c.用户使用了 STOP 指令将 PLC 置 STOP;

d.下载程序时中途出现下载失败，PLC 会维持软件控制的 STOP 状态。 直到下面情形中的一个事件发生。

（2）PLC 的运行状态只取决于拨码开关的控制，软件中的启动/停止设定不生效的情况

a.PLC 重新上电时，PLC 的状态取决于拨码开关当前的位置；

b.软件执行了清除 PLC 命令， 清除完成后拨码开关所在的位置即 PLC 所处的状态；

c.下载程序时，若拨码在 RUN 位置，软件会提示强制 PLC 为 STOP 从而继续下载，下载成功后，PLC 恢复 RUN 状态；下载时若拨码在 STOP 位置，下载成功后，PLC 保持 STOP 状态。

d.任何时候，用户拨动拨码开关都可以切换 PLC 的启动/停止状态。

- CPU 指示灯

Run、Err 这 2 个指示灯用于指示 CPU 当前的工作情况。

【Run】：若 CPU 正处于运行状态，则 RUN 灯点亮;若 CPU 正处于停止状态，则 RUN 灯灭.。

【Err.】：若 CPU 检测到用户程序或者模块本身发生错误，则点亮 Err 灯。错误分为三个等级：致命错误、严重错误、一般错误。当 CPU 检测到错误时，会根据错误的等级采取不同的处理措施，同时点亮 Err 灯，并将具体的错误码根据发生的先后次序依次存储下来，以供用户读取并进行分析。

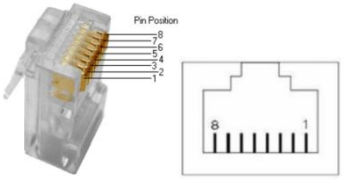
2.2.2 编程口及串行通信口

KS 提供了 2 个串行通信口，分别命名为 PORT0、PORT1。

PORT0 口为 RS232 类型，PORT1 口为 RS485 类型。这两个通信口都可以用作编程口，也支持 Modbus RTU 主站（仅 PORT1 支持）及从站协议和自由通信。波特率最高 115200bps。

各通信口位置及管脚定义见产品外壳丝印图。

RJ45 接口的管脚描述如下：



各通信口的接线图请参见章节的 2.3 接线图

⚠ 由于 RS232 的电气标准不支持带电插拔，因此在插拔电缆之前必须先断掉至少一方（CPU 或者计算机）的电，否则容易损坏通讯口。

2.2.3 以太网通信口

KS101M-04DX 提供了符合标准 IEEE802.3 规范的 Ethernet 接口。该接口支持编程协议，可以作为编程口使用，另外也支持 ModBus TCP Sever，即通常说的“从站”功能。

通信电缆采用直通电缆（直连线）或者交叉电缆（交叉线）均可。KS101M-04DX 的以太网接口提供了“自动协商”功能，当插入电缆后，KS101M-04DX 会跟通信对方自动进行协商以确定所用电缆类型。

用户可以通过 USB 口、串口或者以太网口自身来修改以太网接口的参数：在 KincoBuilder 软件中执行【工具】--【TCP/IP 参数配置】菜单命令，将弹出如下对话框，用户可以读取或者修改参数。

在局域网内，位于同一网段（即 IP 地址的前三段数字要相同，最后一段不同）的 PLC、PC 之间才能相互通信。

2.2.4 CAN 通讯口

KS105C1-16DT 提供了 1 个 CAN 接口，命名为 CAN，支持 Kinco 运动控制指令、支持 CANopen 主站、从站和自由通信。

KS105C2-16DT/KS101M-04DX 提供了 2 个 CAN 接口，分别命名为 CAN1、CAN2。其中，CAN1 口支持扩展协议（可接扩展模块--最多可带 14 个扩展模块）和自由通信；CAN2 口支持 Kinco 运动控制指令、CANOpen 主站、自由通信。

各通信口位置及管脚定义、总线终端电阻见产品外壳丝印图。

2.2.5 扩展模块接口

KS105-16DT 提供扩展模块接口，可以连接 KS 系列扩展模块。

KS105C2-16DT/KS101M-04DX 的 CAN1 接口，既可作为扩展模块接口连接 KS 系列扩展模块，也支持 CAN 通讯功能。这两种功能用户可直接使用，无需特殊设置，PLC 会自动进行识别，**但注意的是只能扩展协议和 CAN 通讯功能只能任选其一。**

CPU 连接扩展模块时，可直接使用标准屏蔽超五类双绞直连网线连接，扩展模块提供一进一出的接口，方便用户连接多个扩展模块时使用。具体连接可参考下图：

扩展模块连接无需设定地址，其实际在整个网络的位置由在 KincoBulider 软件硬件配置时确定模块顺序决定。如下图所示，硬件配置中 KS123-14DR 在 CPU 模块之后的第一个位置，KS133-06IV 在 CPU 模块的第二个位置（也就是紧随 KS123-14DR 之后的位置）。

硬件配置

	模块名称	输入区地址	输出区地址	扩展 +5V	扩展 +24V	备 注
1	KS105-16DT	0...0	0...0	---	---	CPU模块, DC24V供电, DI 8*DC24V, DO 8*DC24V
2	KS123-14DR	1...1	1...1	---	---	KS123, DI 8*DC24V, DO 6*继电器
3	KS133-06IV	0...7	0...3	---	---	KS133, AI*4, AO*2, 4-20mA/0-20mA/1-5V/0-10V

2.2.6 高速脉冲计数和高速脉冲输出

KS 提供了 4 路高速计数器，编号为 HSC0 至 HSC3。高速计数器支持多种模式，可以进行单相、双相（Up/Down）、AB 相（1 倍频和 4 倍频）等计数,最高计数频率全部为 200KHz（单相，AB 相均是）。

另外，KS 也提供了 4 路高速输出，所用通道分别为 Q0.0、Q0.1 和 Q0.4、Q0.5，都支持 PTO（脉冲串）和 PWM（脉宽调制）方式输出。其中，Q0.0 和 Q0.1、Q0.4 通道的最高输出频率可达 200KHz（要求负载电阻不大于 1.5KΩ），Q0.5 通道的最高输出频率为 10KHz。

2.2.7 边沿中断

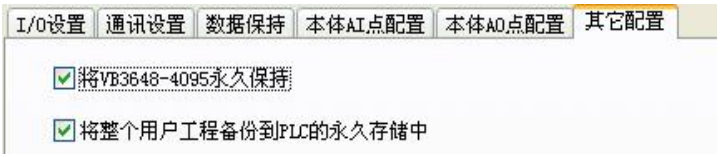
CPU 本体的输入点 I0.0—I0.3 支持边沿中断功能，可以利用输入信号的上升沿和下降沿产生中断。利用这一功能够快速捕捉到输入的上升沿或下降沿，对一些脉冲宽度小于 CPU 扫描周期的输入信号实现快速响应。

2.2.8 数据保持和数据备份

数据保持是指在 CPU 模块 RAM 中的数据在断电后保持为断电瞬间的状态，并供 CPU 在下一次上电的时候使用。CPU 模块内部均提供一个后备锂电池（不可充电，可更换）用于数据保持功能。在断电时，后备电池为 RAM 供电并保持 RAM 中的数据。用户需要使用 KincoBuilder 软件在用户工程的【PLC 硬件配置】中选择需保持的数据区类型（如 V 区、C 区等）及起止范围。其中 V 区数据保持的范围为 VB0-VB1023 总共 1024 个字节。常温下，电池典型寿命为 5 年,断电电保持的时间累计不小于 3 年。

数据备份是指 CPU 模块在永久存储器中开辟一个区域，用于存放用户数据，该区域内的数据断电永久不会丢失，并供 CPU 在下一次上电的时候使用。**CPU 模块提供了 E2PROM 存储器用于数据备份功能，由于 E2PROM 只有 100 万次的写入寿命，因此用户注意尽量避免备份那些变化频繁的数据!**

V 区中的最后 448 个字节（即 VB3648--VB4095）是数据备份区域，该区域中的数据会自动备份到永久存储器中。KS 默认的永久保持区域与 K5 保持一致，即为 VB3648--VB3902。若要使 VB3903--VB4095 也成为永久保持区域，则需要在【PLC 硬件配置】中进行配置，如不进行配置，则这个区域的数据不会自动备份。配置界面如下图：



2.2.9 实时时钟（RTC）

CPU本体内都集成了实时时钟(RTC)，可提供实时的时间/日历表示。在第一次使用RTC时，用户需要通过在KincoBuilder中执行【PLC】->【调整CPU时钟…】菜单命令来设置时钟，之后就可以使用读写实时时钟的指令（READ_RTC、SET_RTC、RTC_W、RTC_R），实现与相关的控制应用。

CPU断电后，实时时钟依靠后备电池的供电来维持运行，常温下，电池典型寿命为5年,断电保持的时间累计不小于3年。

2.2.10 后备电池

KS允许使用特定规格的的锂电池作为后备电池。当断电时，后备电池用于给实时时钟供电来维持时钟的运行，同时也给RAM供电来进行数据保持。

后备电池可以拆卸，但需要开盖更换，打开外壳后即可看到如右图的电池，用户可以自行更换。

电池为CR2032带连接器的3V锂电池，形状如右图，用户可以单独订购电池。

2.3 接线图

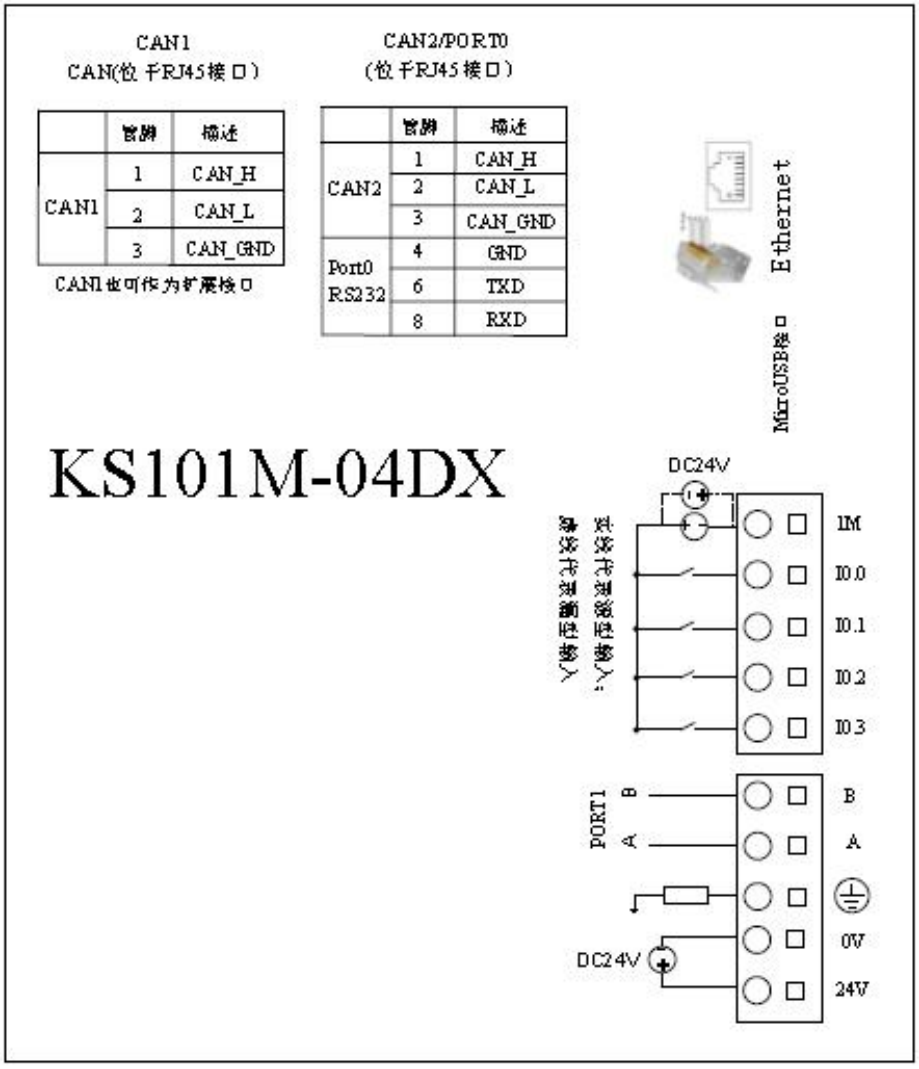


图 2-1　KS101M-04DX 接线图

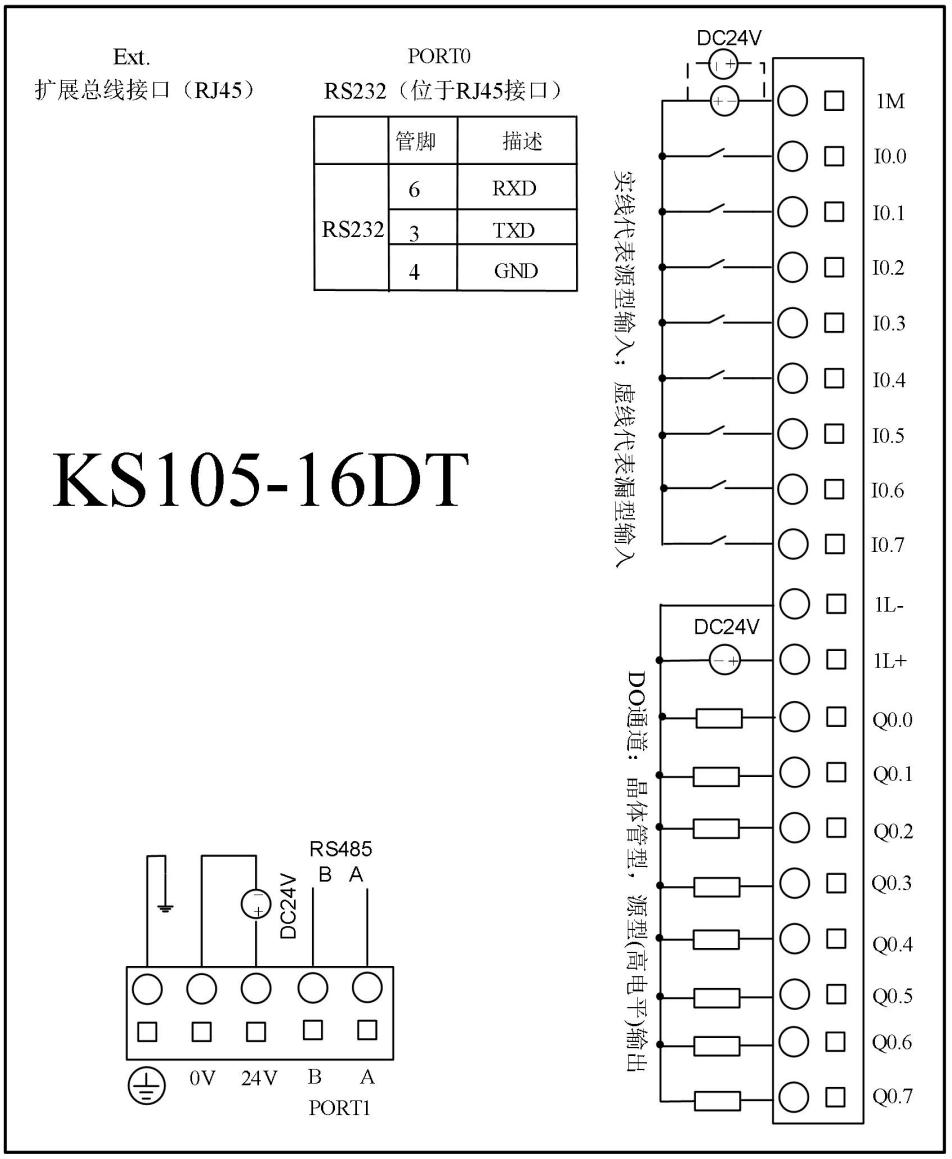


图 2-2　KS105-16DT 接线图

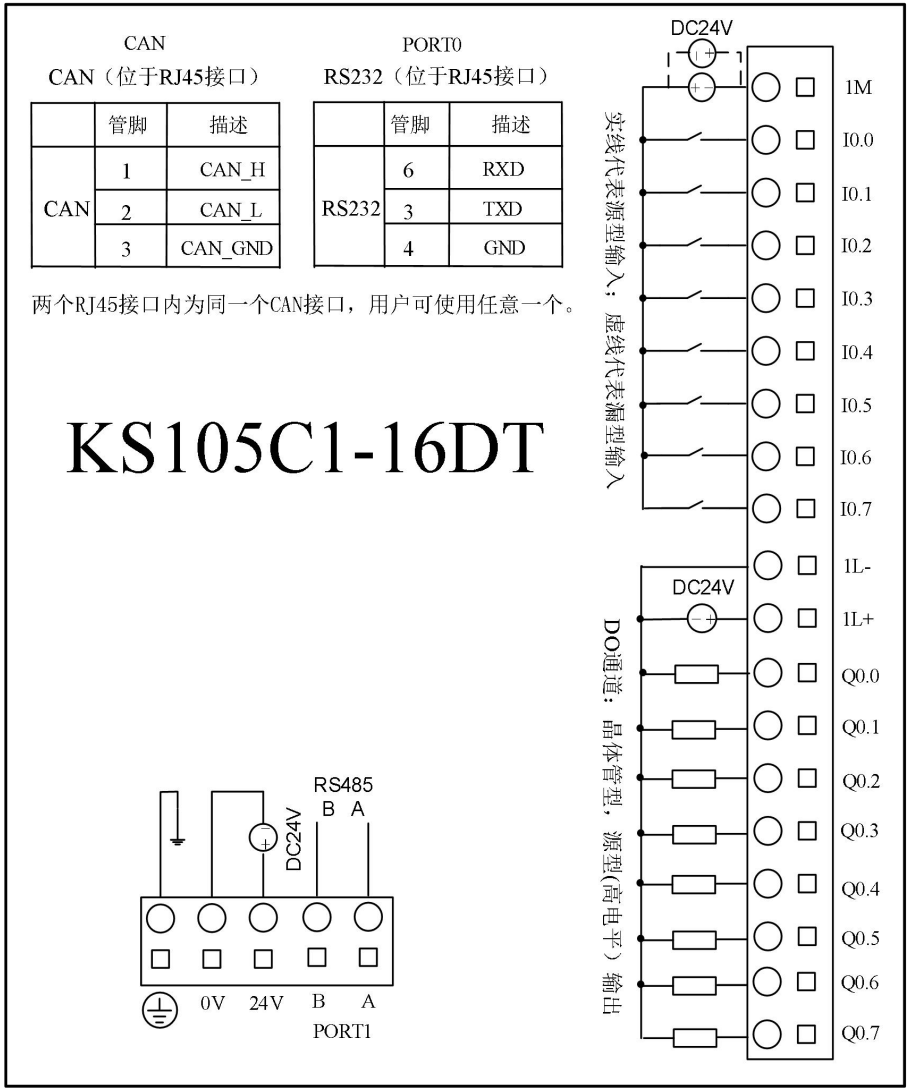


图 2-3　KS105C1-16DT 接线图

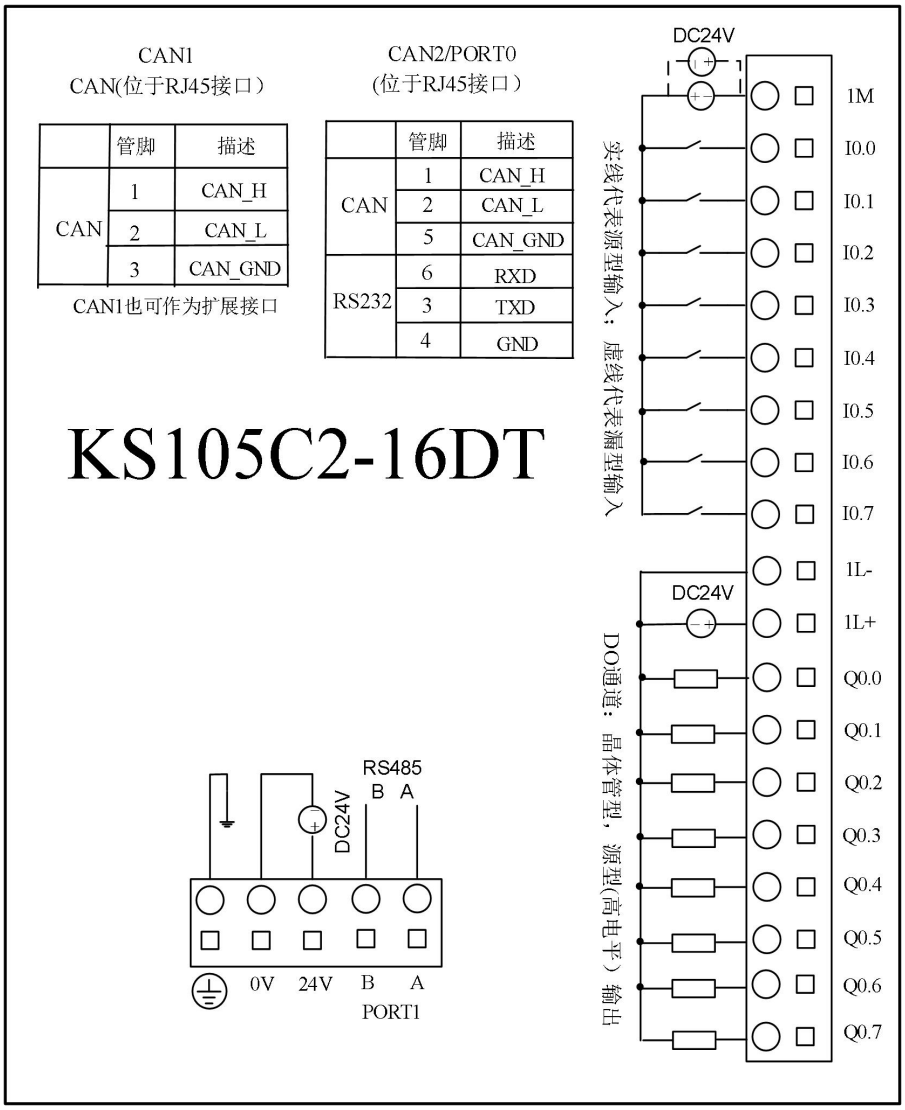
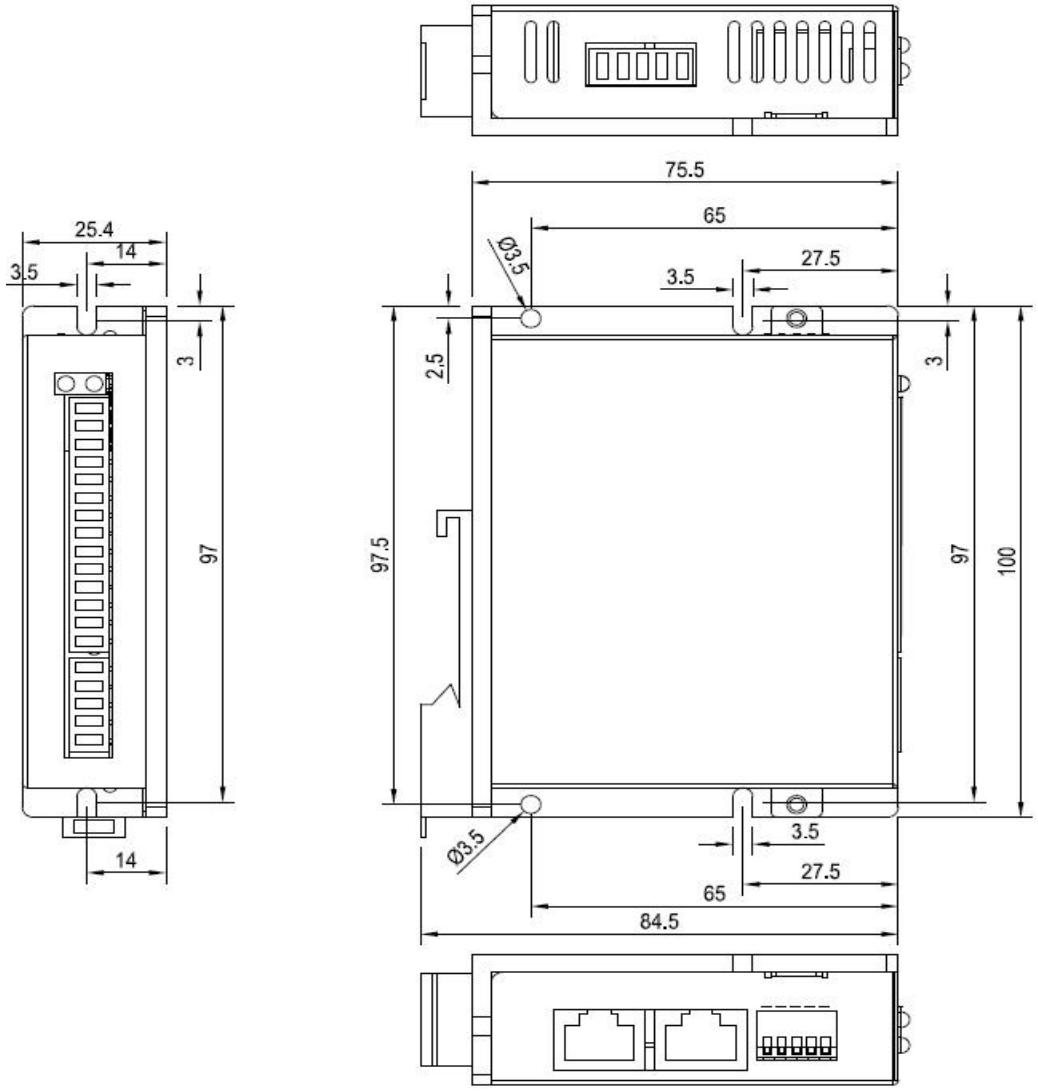


图 2-4　KS105C2-16DT 接线图

2.4 尺寸图



2.5 技术数据

- 本体 DI 通道参数

输入类型	源型/漏型
额定输入电压	DC24V，最高允许 DC30V。
额定输入电流	3.5mA@24VDC
逻辑“0”最大输入电压	5V@0.7mA
逻辑“1”最小输入电压	11V@2.0mA
输入延迟时间 • 接通延时 • 断开延时	 1.2μs 0.5μS
输入与内部逻辑电路的隔离 • 隔离方式 • 隔离电压	 光电耦合器 500VAC/1 分钟

- 本体 DO 通道参数（晶体管型）

输出类型	高电平输出
额定输出电压	DC24V。允许范围: DC20.4V—DC28.8V（与供电电压一致）。 **注意：由于受限于产品尺寸，晶体管输出的公共电源内部没有滤波及限幅电路，如果接入电源超过允许范围，可能会造成内部器件损坏；在电源恶劣的环境中请外部增加电路以保证 1L+ 1L-间电压在允许范围内，平均值不超过 28.8V,最大尖峰电压不超过 32V。**
每通道输出电流	额定 200mA，最大 300mA @24VDC
每通道瞬时冲击电流	1A,不超过 1S
输出漏电流	最大 0.5μA
输出阻抗	最大 0.2Ω
输出延迟时间 • 接通延时 • 断开延时	 普通通道 12μs，高速通道 0.5μs 普通通道 35μS，高速通道 1μS
保护功能： • 电源接入极性保护 • 感性负载输出保护	 无 有

• 短路保护 • 输出极性反向保护	有 有，允许在输出端施加反极性信号不超过 10s。
输出与内部逻辑电路的隔离 • 隔离方式 • 隔离电压	 光电耦合器 500VAC/1 分钟

第三章 扩展模块

3.1 概述

KS 系列扩展模块可作为 KS 系列 CPU 的扩展，具体使用方法参考 2.2.4 扩展模块接口。需要注意的是，KS 系列扩展模块供电电源为 DC24V，无需从扩展总线供电，详见各扩展模块接线图

同时 KS 系列扩展模块提供一个标准 RS485 接口，可作为标准的 Modbus 从站供主站访问。各模块 Modbus 地址见下表。

*注：某些 Modbus RTU 主站的寄存器从 1 开始编号，此时将表中的数据直接加 1 即可。

扩展模块型号	类型	ModBus 功能码	内存区域	ModBus 寄存器号
KS122-14DT	DO（开关量输出，0XXXX）	01，05，15	Q0.0-Q1.5	0-13
KS122-12XR			Q0.0-Q1.3	0-11
KS123-14DR	DI（开关量输入，1XXXX）	02	I0.0 --- I0.7	0-7
	DO（开关量输出，0XXXX）	01，05，15	Q0.0-Q0.5	0-5
KS133-06IV	AI（模拟量输入，3XXXX）	04	AIW0 --- AIW6	0-3
	AO（模拟量输出，4XXXX）	03，06，16	AQW0 --- AQW2	0-1
KS121-16DX	DI（开关量输入，1XXXX）	02	I0.0 --- I1.7	0-15
KS131-04RD	AI（模拟量输入，3XXXX）	04	AIW0 --- AIW6	0-3

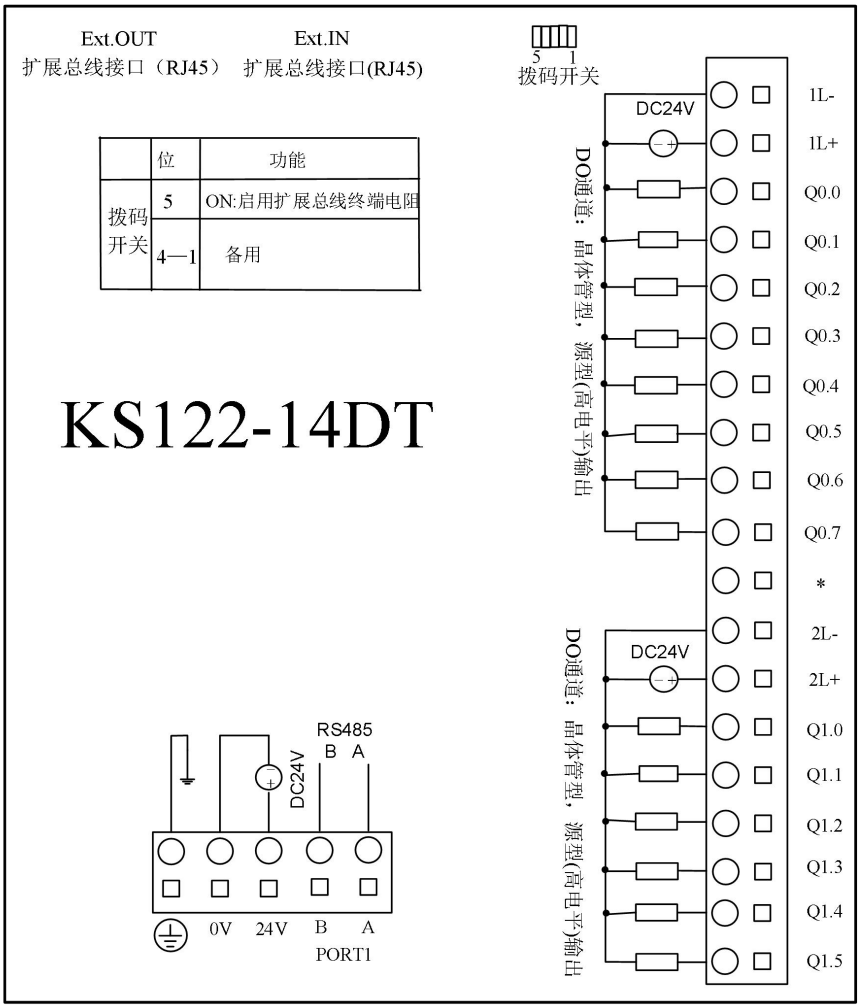
3.2 DO 扩展模块

3.2.1 DO 14×晶体管

该模块的订货号是：Kinco-KS122-14DT。

这是具有 14 个通道的 DO 模块，其中 DO 14×晶体管。

3.2.1.1 接线图



3.2.1.2 技术数据

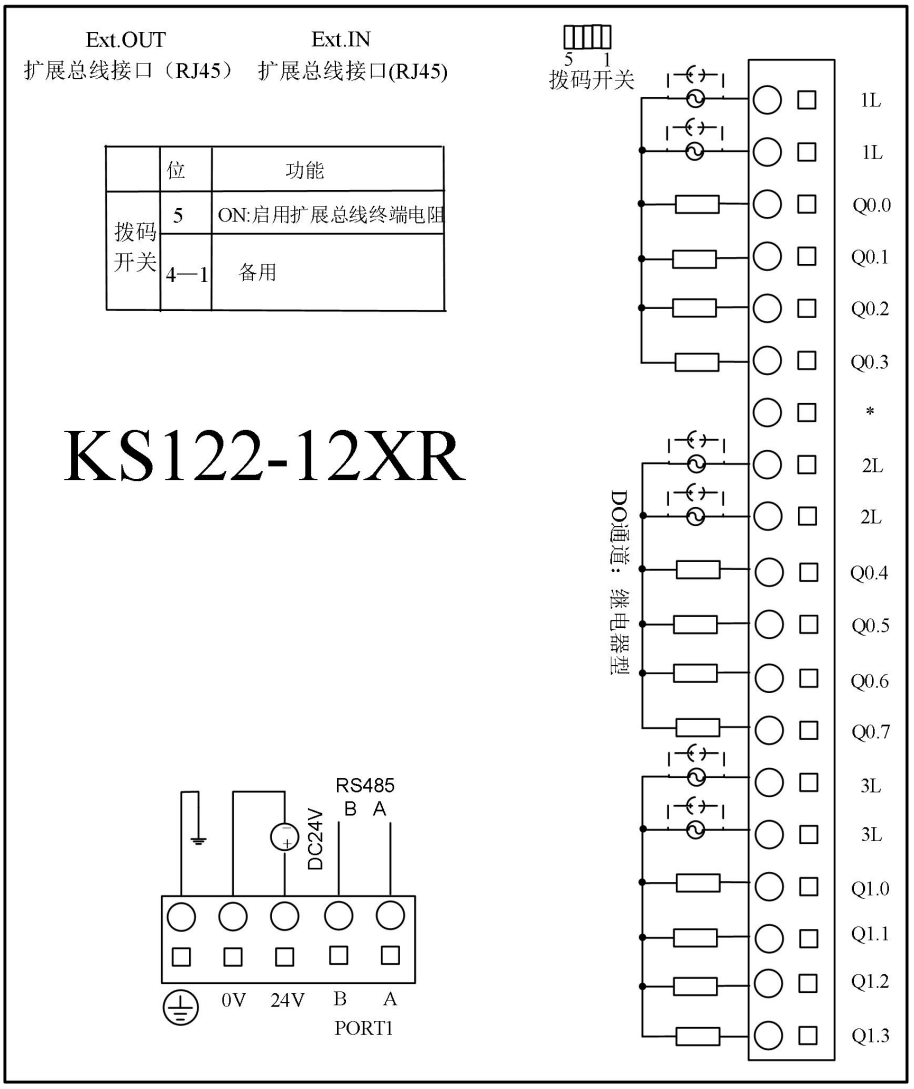
额定供电电源	DC 24V，≥100mA
电气参数	
输出通道数	14（8 通道/组）
输出类型	源型
额定输出电压	DC24V。允许范围: DC20.4V—DC28.8V（与供电电压一致）。
每通道输出电流	最大 500mA@24VDC
输出漏电流	最大 0.5μA
输出阻抗	最大 0.2Ω
输出延迟时间 • 接通延时 • 断开延时	 0.3--5μs 5μs
保护功能： • 电源接入极性保护 • 感性负载输出保护 • 短路保护 • 输出极性反向保护	 有 有 有 有，允许在输出端施加反极性信号不超过 10s。
输出与内部逻辑电路的隔离 • 隔离方式 • 隔离电压	 光电耦合器 500VAC/1 分钟
尺寸和重量	
尺寸(长×宽×高)	100×84.5×25.4mm
净重	200g

3.2.2 DO，DO 12×继电器

该模块的订货号是：Kinco-KS122-12XR。

这是具有 12 个通道的 DO 模块，其中 DO 12×继电器。

3.2.2.1 接线图



3.2.2.2 技术数据

额定供电电源	DC 24V，≥100mA
电气参数	
输出通道数	12 继电器（4 通道/组）
允许负载电压	DC 30V/AC250V

允许负载电流	2A（DC 30V/AC250V）
每组最大输出电流	10A
输出接通延迟时间	10ms（最大值）
输出断开延迟时间	5ms（最大值）
继电器触点预期寿命 • 机械寿命（空载） • 电气寿命（额定负载）	 20，000，000 次（1200 次/分钟） 100，000 次（6 次/分钟）
输出隔离特性 • 隔离方式 • 线圈与触点的隔离电压	 继电器 2000Vrms
尺寸和重量	
尺寸(长×宽×高)	100×84.5×25.4mm
净重	200g

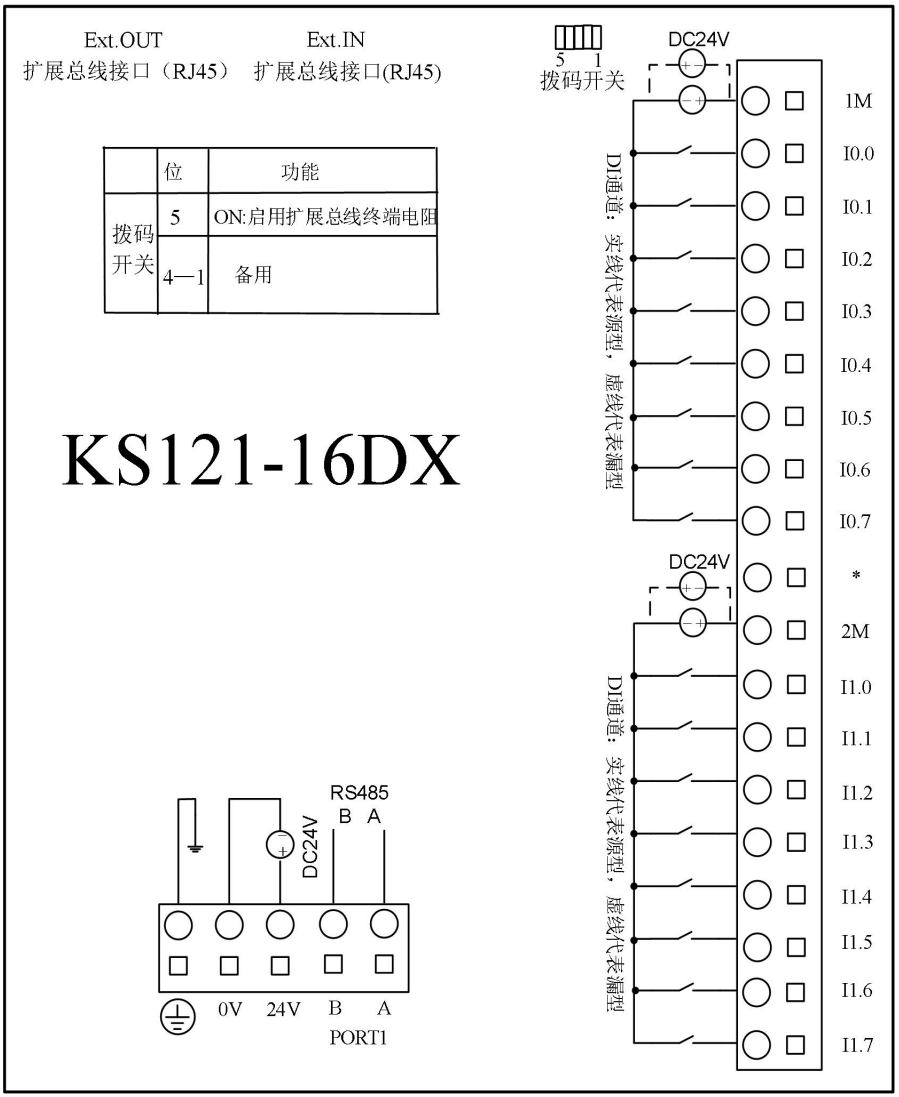
3.3 DI 扩展模块

3.3.1 DI 16×DC24V

该模块的订货号是：Kinco-KS121-16DX。

这是具有 16 个通道的 DI 模块，其中 DI 16×DC24V。

3.3.1.1 接线图



	位	功能
拨码开关	5	ON:启用扩展总线终端电阻
	4—1	备用

3.3.1.2 技术数据

额定供电电源	DC 24V，≥100mA
电气参数	
输入通道数	16（8 通道/组）

输入类型	源型/漏型
额定输入电压	DC 24V
额定输入电流	3.5mA@24VDC
逻辑“0”最大输入电压	5V
逻辑“1”最小输入电压	11V@2.0mA
输入延迟时间 • 接通延时 • 断开延时	 12μs 40μS
输入与内部逻辑电路的隔离 • 隔离方式 • 隔离电压	 光电耦合器 500VAC/1 分钟
尺寸和重量	
尺寸(长×宽×高)	100×84.5×25.4mm
净重	200g

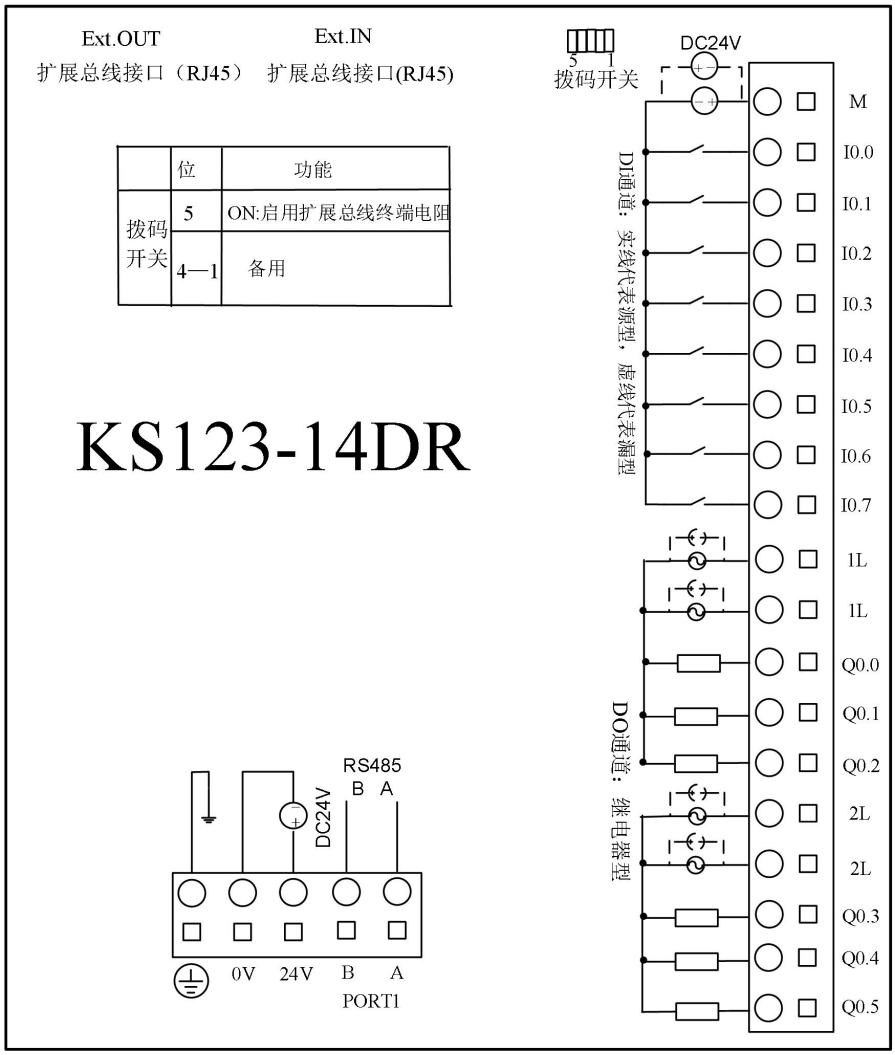
3.4 DI/O 扩展模块

3.4.1 DI/O， DI 8×DC24V DO 6×继电器

该模块的订货号是：Kinco-KS123-14DR。

这是具有 14 个通道的 IO 混合模块，其中 DI 8×DC24V，DO 6×继电器。

3.4.1.1 接线图



	位	功能
拨码开关	5	ON:启用扩展总线终端电阻
	4—1	备用

3.4.1.2 技术数据

额定供电电源	DC 24V，≥100mA
电气参数	
输入通道数	8（8 通道/组）
输入类型	源型/漏型

额定输入电压	DC 24V
额定输入电流	3.5mA@24VDC
逻辑“0”最大输入电压	5V
逻辑“1”最小输入电压	11V@2.0mA
输入与内部逻辑电路的隔离 • 隔离方式 • 隔离电压	 光电耦合器 500VAC/1 分钟
输出通道数	6 继电器（3 通道/组）
允许负载电压	DC 30V/AC250V
允许负载电流	2A（DC 30V/AC250V）
每组最大输出电流	10A
输出接通延迟时间	10ms（最大值）
输出断开延迟时间	5ms（最大值）
继电器触点预期寿命 • 机械寿命（空载） • 电气寿命（额定负载）	 20，000，000 次（1200 次/分钟） 100，000 次（6 次/分钟）
输出隔离特性 • 隔离方式 • 线圈与触点的隔离电压	 继电器 2000Vrms
尺寸和重量	
尺寸(长×宽×高)	100×84.5×25.4mm
净重	200g

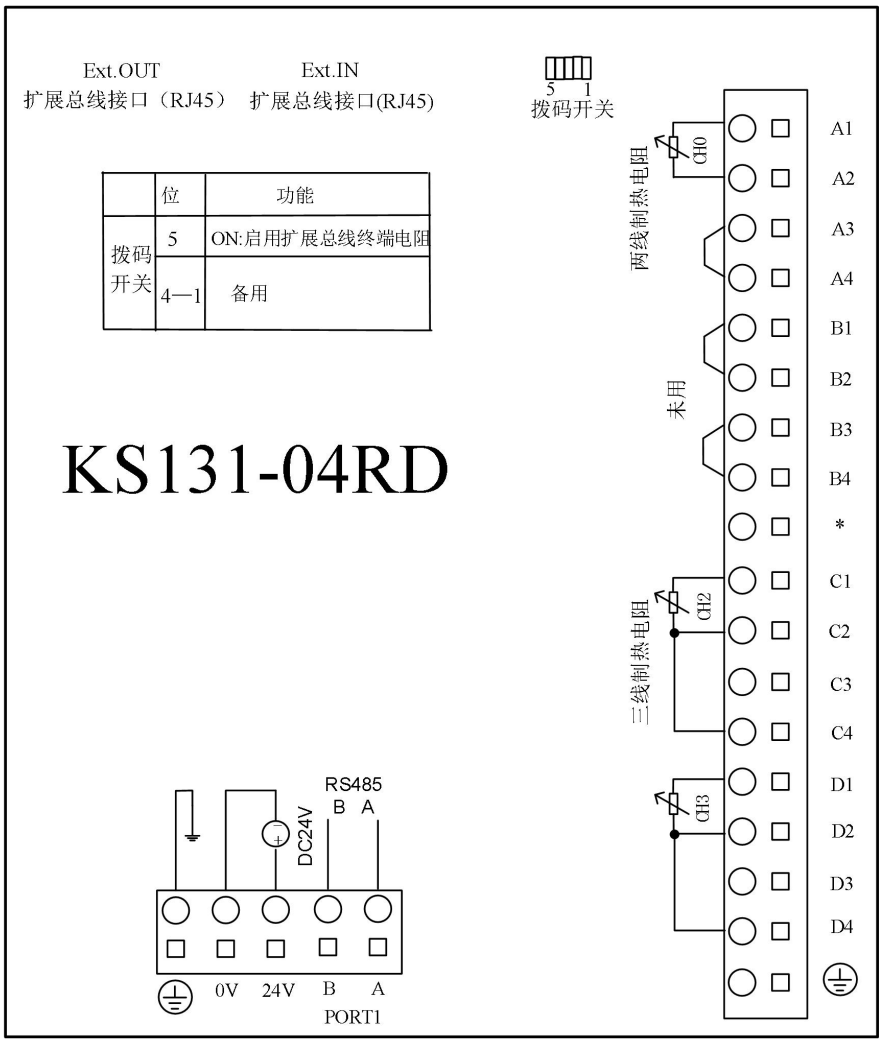
3.5 AI 扩展模块

3.5.1 AI 4×RD 热电阻输入

该模块的订货号是：Kinco-KS131-04RD。

该模块具有 4 个通道，可以接入热电阻（Pt100、Pt1000、Cu50、R）来测量温度，也可以直接测量电阻的阻值。通道的信号形式在编程软件中进行配置。各通道可以混合接入不同的电阻型号，并且支持两线制和三线制两种接线形式。

3.5.1.1 接线图



3.5.1.2 测量范围和测量值表示格式

各通道的输入信号经过 ADC 采样、计算后得到电阻值，然后根据所选热电阻的温度-电阻特性公式计算得到相应的温度值，并经过扩展总线送往 CPU 模块的 AI 区中以供用户程序访问。

各种信号形式均有一定的测量范围，若被测值超出测量范围，则 AI 值保持为相应的上限值或者下限值，同时模块将会报警，同时通过扩展总线向 CPU 模块发送故障报文。**建议用户将未用通道的**

端子短接起来，并在编程软件中将其信号形式配置为【电阻】，那么这些未用通道将不会引起报警。

下表是测量范围和测量值表示格式，其中 T 代表被测温度值，R 代表被测电阻值。

信号形式	测量范围	测量值表示格式
Pt100	-200～850℃	T×10
Cu50	-50～150℃	
Pt1000	-50～300℃	
电阻	0～2000Ω	R×10

3.5.1.3 技术数据

额定供电电源	DC 24V，≥100mA
电气参数	
通道数	4
信号形式	Pt100、Cu50、Pt1000、电阻
接线形式	两线制或者三线制
分辨率（含符号位）	24 位
测量精度	温度：±0.6℃； 电阻：±1Ω
转换速率（每通道）	约 1 次/秒
尺寸和重量	
尺寸(长×宽×高)	100×84.5×25.4mm
净重	200g

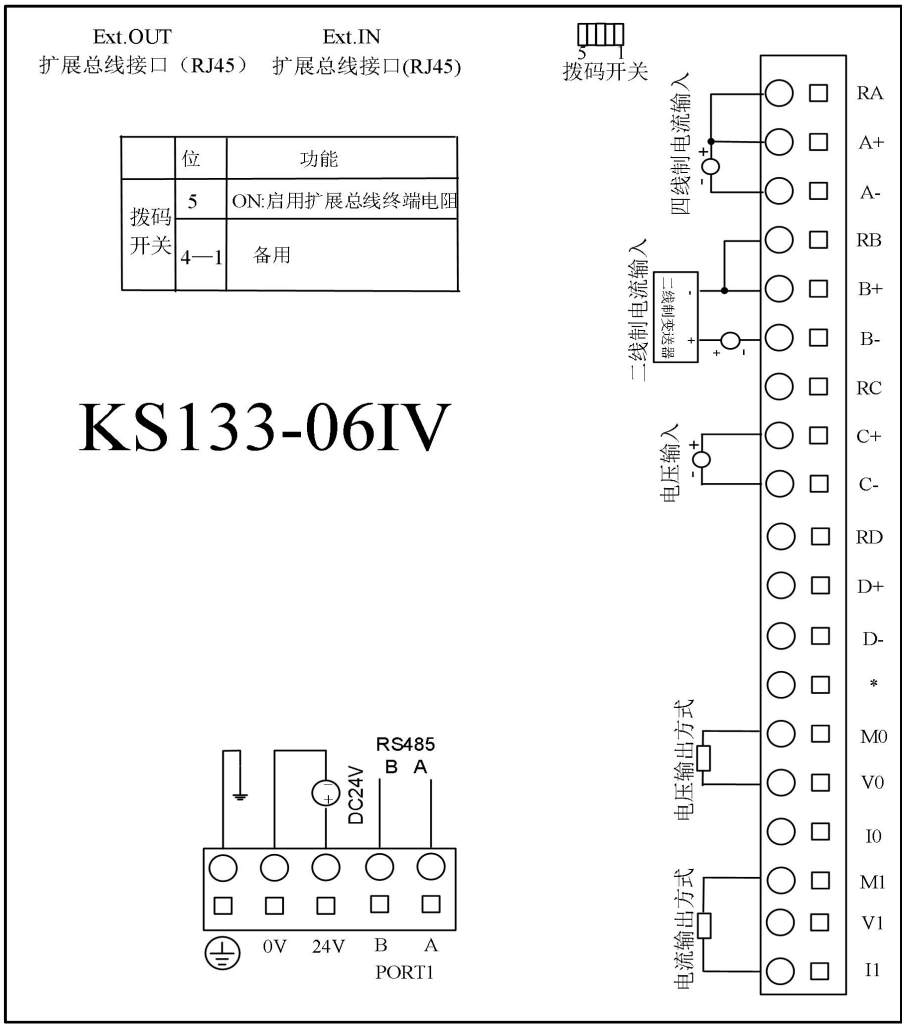
3.6 AI/O 扩展模块

3.6.1 AI/O，AI 4×IV AO 2×IV，电流/电压输入/输出

该模块的订货号是：Kinco-KS133-06IV。

该模块具有 4 个 AI 通道和 2 路 AO 通道，分别刻有测量、输出标准的电压或电流信号（4-20mA、1-5V、0-20mA、0-10V）。在一个模块中各通道允许选择不同的信号形式(具体参考 K5 系列使用手册）

3.6.1.1 接线图



3.6.1.2 AI 测量范围和测量值表示格式

各通道的输入信号经过 ADC 采样、线性计算后，计算结果作为测量值经过扩展总线送往 CPU 模块的 AI 映像区中以供用户程序访问。

各种信号形式均有一定的测量范围，若被测值超出测量范围，则模块将会报警：相应通道的 LED 点亮，同时通过扩展总线向 CPU 模块发送故障报文。**建议用户将未用通道的端子短接起来，并在编**

程软件中将其信号形式设置为【0-20mA】或者【0-10V】，那么这些未用通道将不会引起报警。

下表是测量范围和测量值表示格式，其中 I 代表输入电流值，V 代表输入电压值。

信号形式	测量范围	测量值表示格式
4-20mA	3.92-20.4mA	I×1000
0-20mA	0-20.4mA	
1-5V	0.96-5.1V	V×1000
0-10V	0-10.2V	

3.6.1.3 AO 输出范围和输出值表示格式

用户程序中指定的 AQ 输出值首先经过扩展总线送到相应的 AO 模块中，然后经过计算、变换并通过 DAC 在指定的通道输出。

各种信号形式的输出范围是有限制的，若用户程序中指定的输出值超出了所选范围的上、下限，则的输出信号保持上、下限值不变。

下表是输出范围和输出值表示格式，其中 I 代表实际电流值，V 代表实际电压值。

信号形式	输出范围	用户程序中指定的输出值
4-20mA	3.92-20.4mA	I×1000
0-20mA	0-20.4mA	
1-5V	0.96-5.1V	V×1000
0-10V	0-10.2V	

3.6.1.4 技术数据

电气参数	
AI 通道数	4
额定供电电源	DC 24V，≥100mA
信号形式	4-20mA、1-5V、0-20mA、0-10V
分辨率	12 位
测量精度	0.3% F.S.
信号限值	电流输入不超过 24mA，电压输入不超过 12V
转换速率（每通道）	约 15 次/秒

输入阻抗	电流模式：<=250Ω 电压模式：>4MΩ
AO 通道数	2
额定供电电源	DC 24V ≥100mA
信号形式	4-20mA、1-5V、0-20mA、0-10V
分辨率（含符号位）	12 位
输出信号精度	0.3% F.S.
转换速率（每通道）	约 15 次/秒
外部负载	电流模式： 最大 500Ω 电压模式： 最小 10KΩ
尺寸和重量	
尺寸(长×宽×高)	100×84.5×25.4mm
净重	200g

第四章 软件功能及使用

4.1 概述

Kinco-KS 系列延续了 Kinco K5 等老产品的设计，同样采用 Kincobuilder 编程软件及相同的指令集。所以对于绝大部分通用功能及指令请参考 Kinco K5/K2 系列软件使用手册，不同的地方主要是部分功能有所增强，所以请仔细参考本手册的中列出的所有内容。

4.2 使用 ModBus TCP 协议与第三方设备通讯

ModBus TCP 主站可访问的内存区域分类如下：

类型	Modbus 功能码	对应的 PLC 内存区域
DO（开关量输出，0XXXX）	01，05，15	Q 区，M 区
DI（开关量输入，1XXXX）	02	I 区，M 区
AO（模拟量输出，4XXXX）	03，06，16	AQ 区，V 区

AI （模拟量输入，3XXXX）	04	AI 区，V 区
错误记录（16 位无符号整数）	03,04	PLC 错误记录区

一次命令最大访问的寄存器数量如下：

1. 读取位，一次最大读取 1600 个位(200 字节)。(01，02 功能码)
2. 写入位，一次最大写入 800 个位。(15 功能码)
3. 读取字，一次最大读取 100 个字。(03, 04 功能码)
4. 写入字，一次最大写入 100 个字。(16 功能码)
5. 当操作的内存区小于上面的最大值时， 只允许一次操作整个内存区。比如报文中读取 AI 区 90 个字，这是错误的，因为 AI 区最大 32 个字。

4.2.1 ModBus 寄存器编号

由于各种规格的 CPU 的内存区域不同，所以允许访问的范围也有限制，其他规格的参考编程软件 KincoBuilder 里的帮助主题中的附录 A 使用 Modbus RTU 协议通讯。

若 Modbus TCP 主站的寄存器从 1 开始编号，那么将将下表中的寄存器号直接加 1 即可。

- 适用于 KS101M-04DX

内存区域	范围	类型	对应的 Modbus 寄存器号*
I	I0.0 --- I0.3	DI	0 --- 3
M	M0.0 --- M4095.7	DI/DO	320 -- 33087
V	VW0 --- VW16382	AI/AO	100 -- 8291

- 除以上内存区域支持 ModBus 访问外，K 系列 PLC 还提供了错误记录的内存区域供用户通过 MODBUS 读取查看，详见编程软件 KincoBuilder 里的帮助主题中的附录 D 错误诊断功能

4.3 高速计数器的使用

KS 提供了 4 路高速计数器，编号为 HSC0 至 HSC3，最高计数频率全部为 200KHz。

高速计数器具有多种工作模式，可以进行单相、双相（Up/Down）、AB 相（1 倍频和 4 倍频）等计数。所有的高速计数器在相同的工作模式下均具有相同的功能。

所有高速计数器均允许指定最大 32 个预置值（PV），每个 PV 值均支持“计数值=预置值”中断。

PV 值可以指定为相对值或者绝对值方式，若选择为相对值方式，那么“计数值=预置值”中断允许选择循环发生。

4.3.1 高速计数器工作模式和输入信号

高速计数器的输入信号包括如下几种：时钟（即输入脉冲）、方向、启动和复位信号。

在不同的工作模式下，所需要的输入信号也有所不同。下面各表详细描述了各个高速计数器所支持的工作模式及其输入信号的分配。

HSC 0				
模式	**描述**	**I0.1**	**I0.0**	**I0.5**
0	带内部方向控制的单相增/减计数器 方向控制位：SM37.3	时钟		
1			复位	
2			复位	启动
3	带外部方向控制的单相增/减计数器	时钟		方向
4			复位	方向
6	带增/减计数时钟输入的双相计数器	时钟（减）	时钟（增）	
9	A/B 相正交计数器	时钟 A 相	时钟 B 相	

HSC1					
模式	**描述**	**I0.4**	**I0.6**	**I0.3**	**I0.2**
0	带内部方向控制的单相增/减计数器 方向控制位：SM47.3			时钟	
1		复位			
2		复位	启动		
3	带外部方向控制的单相增/减计数器			时钟	方向
4		复位			方向
6	带增/减计数时钟输入的双相计数器			时钟（减）	时钟（增）
7		复位			
9	A/B 相正交计数器			时钟 A	时钟 B
10		复位			

HSC 2			
模式	**描述**	**I0.4**	**I0.5**
0	带内部方向控制的单相增/减计数器。方向控		时钟

	制位：SM57.3		
9	A/B 相正交计数器	时钟 B 相	时钟 A 相

HSC 3			
模式	**描述**	**I0.6**	**I0.7**
0	带内部方向控制的单相增/减计数器。方向控制位：SM127.3		时钟
9	A/B 相正交计数器	时钟 B 相	时钟 A 相

4.3.2 控制寄存器和状态寄存器

- 控制寄存器

在 SM 区中为每个高速计数器均提供了如下控制寄存器用于存放配置数据。其中，当前值用于修改计数器当前的计数值，若将当前值写入高速计数器，那么高速计数器就会立即从这个新数值开始计数。下表详细描述了这些寄存器。

HSC0	HSC1	HSC2	HSC3	描 述
SM37.0	SM47.0	SM57.0	SM127.0	复位信号的有效电平：0=高电平；1=低电平
SM37.1	SM47.1	SM57.1	SM127.1	启动信号的有效电平：0=高电平；1=低电平
SM37.2	SM47.2	SM57.2	SM127.2	正交计数器速率：0=1x 速率；1=4x 速率*
SM37.3	SM47.3	SM57.3	SM127.3	计数方向：0=减计数；1=增计数。
SM37.4	SM47.4	SM57.4	SM127.4	是否向 HSC 中写入计数方向：0=否；1=是。
SM37.5	SM47.5	SM57.5	SM127.5	是否向 HSC 中写入新预置值：0=否；1=是。
SM37.6	SM47.6	SM57.6	SM127.6	是否向 HSC 中写入新当前值：0=否；1=是。
SM37.7	SM47.7	SM57.7	SM127.7	是否允许该高速计数器：0=禁止；1=允许。
HSC0	**HSC1**	**HSC2**	**HSC3**	**描 述**
SMD38	SMD48	SMD58	SMD128	当前值
SMD42	SMD52	SMD62	SMD132	预置值

HSC0	HSC1	HSC2	HSC3	描 述
SM141.0	SM151.0	SM161.0	SM171.0	是否使用多段预置值：0=否；1=是
SM141.1	SM151.1	SM161.1	SM171.1	预置值是相对值还是绝对值：0 =绝对；1=相对
SM141.2	SM151.2	SM161.2	SM171.2	预置值比较（“CV=PV”）中断是否循环产生： 0=否；1=是。 注意：只有相对值方式才允许设定为循环产生。

SM141.3	SM151.3	SM161.3	SM171.3	保留
SM141.4	SM151.4	SM161.4	SM171.4	是否更新段数及预置值：0=否；1=是
SM141.5	SM151.5	SM161.5	SM171.5	是否复位中断变量：0=是；1=否
SM141.6	SM151.6	SM161.6	SM171.6	保留
SM141.7	SM151.7	SM161.7	SM171.7	保留
HSC0	**HSC1**	**HSC2**	**HSC2**	**描 述**
SMW142	SMW152	SMW162	SMW172	预置值表的起始位置（用相对于 VB0 的字节偏移来表示），必须为奇数。

需要注意的是，控制字节中并非所有的控制位都适用于所有的工作模式。比如，“计数方向”和“是否向 HSC 中写入计数方向”这两个控制位就只用于模式 0、1 和 2（带内部方向控制的单相增/减计数器），若高速计数器所用的工作模式是采用外部的方向控制信号，那么这两个控制位就会被忽略。

控制字节、当前值和预置值上电后的缺省值均为 0。

- 状态寄存器

每个高速计数器都在 SM 区中提供了状态寄存器用于指明高速计数器当前的状态信息。

HSC0	**HSC1**	**HSC2**	**HSC3**	**描 述**
SM36.0	SM46.0	SM56.0	SM126.0	保留
SM36.1	SM46.1	SM56.1	SM126.1	保留
SM36.2	SM46.2	SM56.2	SM126.2	保留
SM36.3	SM46.3	SM56.3	SM126.3	多段 PV 值表设置是否有错误：0=否，1=是。
SM36.4	SM46.4	SM56.4	SM126.4	保留
SM36.5	SM46.5	SM56.5	SM126.5	当前计数方向：0=减；1=增。
SM36.6	SM46.6	SM56.6	SM126.6	当前计数值是否等于预置值：0=否；1=是。
SM36.7	SM46.7	SM56.7	SM126.7	当前计数值是否大于预置值：0=否；1=是。
HSC0	HSC1	HSC2	HSC3	描 述
SMB140	SMB150	SMB160	SMB170	正在运行的 PV 值段序号（从 0 开始）。

4.3.3 预置值（PV 值）设定

KS 允许每个高速计数器设定最多 32 个 PV 值，允许选择 PV 之间的关系是相对值或者绝对值，允许“CV=PV”中断循环产生。同时，KS 也兼容老产品的单 PV 值设定方式。

下面将以 HSC0 为例来对 PV 值的功能和设定方法进行详细的描述。

➢ **如何选择多段 PV 值**

每个高速计数器的控制寄存器里都提供了一个控制位，用于选择是否使用多段预置值。

HSC0 的这个控制位是 SM141.0。

若 SM141.0 为 0，则表示采用单 PV 值方式，与 K5 的用法一致：SMD42 指定了新 PV 值，SM37.5 指明是否使用这个新的 PV 值。

若 SM141.0 为 1，则表示采用多段 PV 值方式，此时 SM37.5、SMD42 无效。各 PV 值存放于 PV 值表中（SMW142 为表起始地址），SM141.4 指明是否使用 PV 值表中的数据。若 SM141.4 为 1，则表示本次启动后，高速计数器时将采用 PV 值表中的数据。若 SM141.4 为 0，则表示本次启动后，高速计数器将采用上一次的 PV 值数据，而忽略 PV 值表中的数据。

➢ **多段 PV 值表**

若使用多段 PV 值，那么各 PV 值将采用 PV 值表中的数据。

每个高速计数器的控制寄存器里都提供一个控制字，用于存放 PV 值表的起始地址，**表的起始地址必须为 V 区中的奇数地址，例如 301（表示 VB301）。**

PV 值表的格式如下。

字节偏移 (1)	数据类型	描述
0	BYTE	PV 值个数
1	DINT	第 1 个 PV 值
5	DINT	第 2 个 PV 值
…	DINT	…

（1） 所有偏移量均是相对于表起始位置的偏移字节数。

（2） 当采用相对值方式时，PV 值的数学绝对值必须大于 1，否则 PLC 认为段数到此结束，并以此统计 PV 值个数（优先于个数设定值）；

当采用绝对值方式时，相邻两个 PV 值之间的差值的数学绝对值必须大于 1，否则 PLC 认为段数到此结束，并以此统计 PV 值个数（优先于个数设定值）；

（3） 用户设定 PV 值时需要注意：“CV=PV”中断必须依次产生。也就是说，当计数值达到第 1 个 PV 值并产生中断后，接下来 PLC 将会与第 2 个 PV 值进行比较，依次类推。

（4） PV 值设置必须合理。以相对值为例，当增计数时，PV 值必须大于 0，否则该值对应的“CV=PV”中断也许永远不会产生；当减计数时，PV 值必须小于 0，否则该值对应的“CV=PV”中断也许永远不会产生。

➢ **相对值和绝对值方式**

每个高速计数器的控制寄存器里都提供了一个控制位，用于选择 PV 值是相对值或者绝对值方式。HSC0 的这个控制位是 SM141.1。

若 SM141.1 为 0，则表示 PV 值是绝对值方式。当计数值等于 PV 值时，将会产生相应的“CV=PV”中断。例如，若设定 3 个 PV 值，依次是 1000、2000、3000，那么计数值达到 1000 时，将产生第 1 个“CV=PV”中断；当计数值达到 2000 时，将产生第 2 个“CV=PV”中断；后面依次类推。

若 SM141.1 为 1，则表示 PV 值是相对值方式，若计数器以当前的计数值为基准，继续计数使得差值等于 PV 值时，将会产生相应的“CV=PV”中断。例如，若设定 3 个 PV 值，分别为 10、1000、1000，而且在高速计数器启动时的计数值是 100，那么当计数值分别达到 110、1110、2110 时，将分别产生“CV=PV”中断。

➢ **“CV=PV”中断循环产生。**

只有 PV 值是相对值方式时，才允许设定为循环产生中断，否则无效。

若 SM141.0 为 0，则表示“CV=PV”中断只产生一次。当所有 PV 值对应的中断都发生完成后就会停止。若要继续产生，则必须修改相应的寄存器值并再次调用 HSC 指令。

若 SM141.0 为 1，则表示“CV=PV”中断会循环产生。当最后一个 PV 值对应的中断发生完成时，PLC 将以当前的计数值为基准，与各个 PV 值再次相加，得到新的中断所需数值，然后继续与计数值进行比较，产生相应的“CV=PV”中断。这个过程会一直循环进行，永不停止。

例如，若设定 3 个 PV 值，分别为 10、1000、1000，而且在高速计数器启动时的计数值是 100，则各个中断每次产生所需数值如下：

当前计数值	中断次数	第 1 个值	第 2 个值	第 3 个值
100	第 1 次	110	1110	2110
2110	第 2 次	2120	3120	4120
4120	第 3 次	4130	5130	6130
…	第 n 次	…	…	…

4.3.4 “CV=PV”中断编号

当采用单 PV 值方式时，那么高速计数器完全兼容 K5，包括“CV=PV”中断的编号与 K5 中的一致。

当采用多段 PV 值方式时，高速计数器为 32 个 PV 值均分配了一个新的中断编号，如下表。

高速计数器	中断事件号	描述
HSC0	64	第 1 个 PV 值的“CV=PV”中断
	65	第 2 个 PV 值的“CV=PV”中断
	…	…（依次加 1）
	95	第 32 个 PV 值的“CV=PV”中断
HSC1	96	第 1 个 PV 值的“CV=PV”中断
	97	第 2 个 PV 值的“CV=PV”中断
	…	…（依次加 1）
	127	第 32 个 PV 值的“CV=PV”中断
HSC2	128	第 1 个 PV 值的“CV=PV”中断
	129	第 2 个 PV 值的“CV=PV”中断
	…	…（依次加 1）
	159	第 32 个 PV 值的“CV=PV”中断
HSC3	160	第 1 个 PV 值的“CV=PV”中断
	161	第 2 个 PV 值的“CV=PV”中断
	…	…（依次加 1）
	191	第 32 个 PV 值的“CV=PV”中断

4.3.5 高速计数器的使用方法

- **方法一：使用相关指令进行编程**

这种方法也是在 K3 和 K5 中使用的方法，KS 也支持这种方法。总体步骤如下：

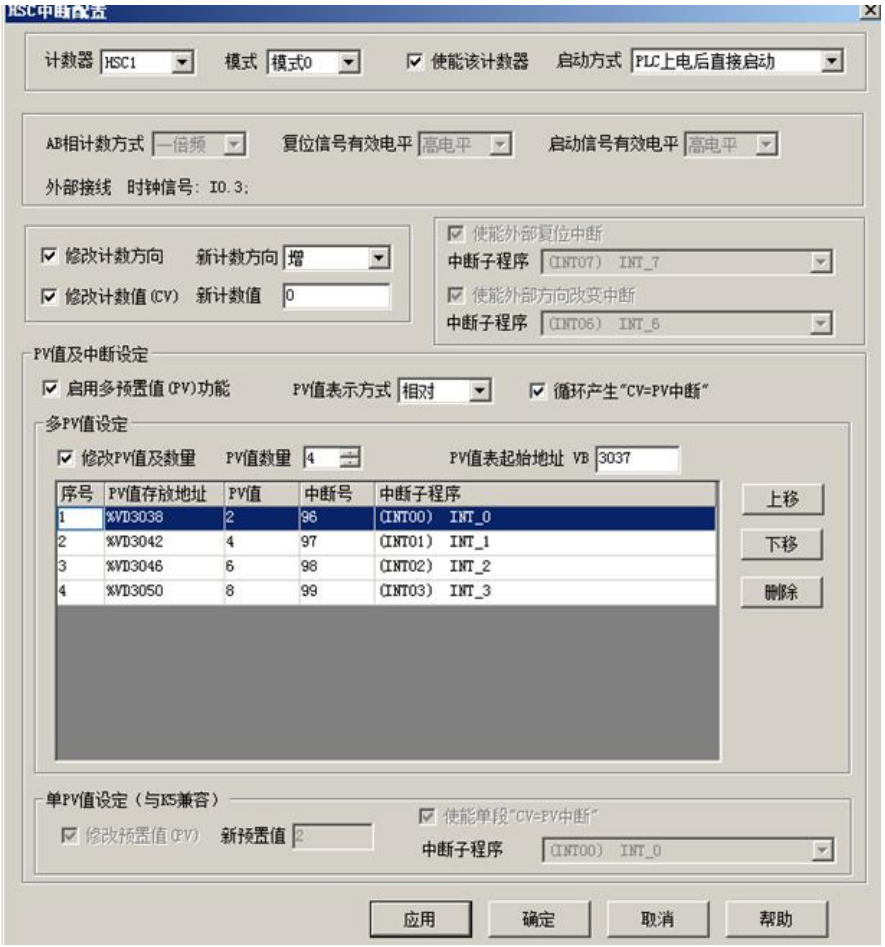
1) 配置该高速计数器的控制字节，并指定当前值（也就是计数的起始值）和预置值；
2) 使用 HDEF 指令来定义一个高速计数器及其工作模式；
3) （可选）使用 ATCH 指令为高速计数器中断连接相应的中断服务程序；
4) 使用 HSC 指令来配置并启动高速计数器。

- **方法二：使用 HSC 向导**

在 KS 中，为高速计数器提供了配置向导。用户可以直接利用该向导对所有的高速计数器进行配置，无需再进行复杂的编程。向导如下图。

即使通过向导对 HSC 进行了配置，用户也可以在程序中按照“方法一”来随时对高速计数器进行

参数修改、启动、停止。



HSC 向导的使用方法如下：

1） 在【计数器】中，选择将要使用的计数器。

2） 选中【使能该计数器】，然后将会允许进行后续的配置。

3） 在【模式】中，选择将要使用的计数器模式。

4） 在【启动方式】中，选择该高速计数器的启动方式。

启动方式有如下两种：

“在程序中调用 HSC 指令”： 若选择这种方式，那么在用户程序中，通过调用 HSC 指令来启动该计数器。在调用 HSC 指令之前，无需再配置各寄存器和调用 HDEF 指令。

“PLC 上电后直接启动”：若选择这种方式，那么该高速计数器在 PLC 上电后就自动运行，无需调用任何指令。

5） 若要使用多段 PV 值方式，则选中【启用多预置值（PV）功能】，然后可以对 PV 值、数量、关联的中断子程序等进行配置。若选中【修改 PV 值及数量】，则可以调整【PV 值数量】中的数值，从而修改 PV 值个数。

6） 若要使用单 PV 值方式，则首先选中“单 PV 值设定（与 K5 兼容）”中的【修改 PV 值】，然后可以修改 PV 值及关联的中断子程序。

7） 其它的配置项，请参考前文的描述，按实际需求进行配置。

4.4 高速脉冲输出功能的使用

KS 也提供了 4 路高速输出，所用通道分别为 Q0.0、Q0.1 和 Q0.4、Q0.5，都支持 PTO（脉冲串）和 PWM（脉宽调制）方式输出。其中，Q0.0 和 Q0.1、Q0.4 通道的最高输出频率可达 200KHz（要求负载电阻不大于 3KΩ），Q0.5 通道的最高输出频率为 10KHz。

针对定位控制指令，Kinco-KS 为每路高速输出均指定了一个方向输出通道，同时还在 SM 区中提供了一个方向使能控制位。如下表。

	Q0.0	Q0.1	Q0.4	Q0.5
方向输出通道	Q0.2	Q0.3	Q0.6	Q0.7
方向使能控制位	SM201.3	SM231.3	SM251.3	SM221.3

方向输出通道用于输出电机的方向控制信号，正转时输出为 0，反转时输出为 1。

方向使能控制位用来禁止或者允许使用相应的方向输出通道。方向使能控制位具有最高的优先级，若设置为禁止，那么定位控制指令执行时将不会输出方向控制信号，相应的输出通道就可以作为普通的 DO 点使用。

4.4.1 高速脉冲输出指令

KS 的指令集中提供了如下 3 种指令用于高速输出功能：

1） PLS 指令：可以实现 PTO（单段或者多段）和 PWM 输出功能。

2） 定位控制指令：共计 5 条指令，包括 PREL（相对运动）、PABS（绝对运动）、PHOME（回原点）、PJOG（点动）、PSTOP（急停）指令，用户能够很方便地实现简单的定位控制功能。**注意：当使用定位控制指令时，输出脉冲频率不能低于 80Hz！**

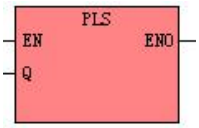
3） 跟随指令 PFLO_F：在输入参数中有输入频率（*F*）、电子齿轮比（*NUME*、*DENOM*）、脉冲个数(*COUNT*)等，这些参数均可以使用变量。该指令的输出脉冲频率等于输入频率乘以电子齿轮比，当输出脉冲个数达到 *COUNT* 个则停止输出并设置完成标志位。**注意：当使用跟随指令时，输出脉冲频率不能低于 30Hz！**

4.4.2 PLS 指令的使用

PLS 指令可以实现 PTO 或者 PWM 输出功能。

- PTO：Pulse Train Output，脉冲串输出。
- PWM：Pulse-Width Modulation，脉宽调制。

➢ 指令及其操作数说明

	名称	指令格式	影响 CR 值	适用于
LD	PLS			• K5 • K2 • K6 • KS • KW
IL	PLS	PLS *Q*	U	

参数	输入/输出	数据类型	允许的内存区
Q	输入	INT	常量（0、1 或者 2）

PLS 指令的作用是：读取 SM 区中相应控制寄存器的值并配置高速脉冲输出的特性，然后启动高速脉冲输出，直到完成指定的脉冲输出功能。脉冲输出通道由参数 *Q* 指定，0 表示使用 Q0.0 输出，1 表示使用 Q0.1 输出，2 表示使用 Q0.4 输出，3 表示使用 Q0.5 输出。

注意：用户程序中，仅在需要时执行一次 PLS 指令即可，建议利用边沿指令的输出结果来调用 PLS 指令。若 PLS 的 EN 端一直保持为 1，那么 PLS 指令将无法正常输出。

• LD

若 *EN* 值为 1，则 PLS 指令被执行。

• IL

若 CR 值为 1，则 PLS 指令被执行。该指令的执行不影响 CR 值。

4.4.2.1 高速脉冲输出功能

KS1 支持 4 路高速脉冲输出，相应地就提供了 4 个 PTO/PWM 脉冲发生器用于产生 PTO/PWM 输出。其中，第一个脉冲发生器分配在 Q0.0，称为 PWM0 或者 PTO0；第二个分配在 Q0.1，称为 PWM1 或者 PTO1；第三个分配在 Q0.4，称为 PWM2 或者 PTO2；第四个分配在 Q0.5，称为 PWM3 或者 PTO3。

PTO/PWM 发生器和 DO 映像寄存器共同使用内存地址 Q0.0、Q0.1 和 Q0.4、Q0.5。如果用户程序中调用了某个通道的高速输出指令，那么 PTO/PWM 发生器将控制输出通道，并禁止普通 DO 的输出。

⚠ *注意: 若Q0.0、Q0.1、 Q0.4、Q0.5是继电器类型的，则避免使用高速脉冲输出功能！*

- **PWM**

PWM 功能提供占空比可调的连续脉冲输出。用户可以控制输出的周期和脉宽。

周期和脉宽的单位可以选择微秒（μs）或毫秒（ms），最大周期值为 65535。当脉宽大于等于周期时，占空比自动地被设为 100%，输出一直接通。当脉宽为 0 时，占空比为 0%，输出断开。

- **PTO**

PTO 功能能够产生指定脉冲个数的脉冲串方波（50%占空比）。用户可以控制输出方波的周期和输出脉冲的个数。脉冲周期的单位是微秒（μs ）或者毫秒（ms），最大周期值为 65535。脉冲个数的范围是：2～4,294,967,295。如果指定脉冲数小于 2，则 KS1 将设置相应的错误标志位并禁止输出。

PTO 功能提供了单段操作和多段操作两种模式。

- **单段操作**

在单段操作模式下，每次执行 PLS 指令后仅会进行一次脉冲串输出。

- **多段操作**

在多段操作模式下，CPU 自动从 V 区的包络表中读出每个 PTO 段的设定值并依据设定值执行该段 PTO。

各段在包络表中的设置均占用 8 个字节，包括一个周期值（16 位无符号整数）、保留值（暂时未用到，16 位符号整数）和一个脉冲个数值（32 位无符号双整数）。也就是说，在同一段中，所有脉冲的输出频率是相同的。多段操作使用 PLS 指令来配置并启动。

包络表的起始位置存储在 SMW168（对应 PTO0）、SMW178（对应 PTO1）、SMW218（对应 PTO2）和 SMW248（对应 PTO3）中，时基通过 SM67.3（对应 PTO0）、SM77.3（对应 PTO1）、SM97.3（对应 PTO2）、SM107.3（对应 PTO3）设置，可以选择微秒或毫秒。包络表中的所有周期值必须使用同一个时基，并且在包络执行时不能改变。

包络表的格式如下表所示。

字节偏移 (1)	长 度	段 数	描 述
0	8 位		段数（1 到 64）
1	16 位	第 1 段	初始周期（2 到 65535 时基）
3	16 位		保留
5	32 位		脉冲个数（1 到 4,294,967,295）
9	16 位	第 2 段	初始周期（2 到 65535 时基）
11	16 位		保留
13	32 位		脉冲数（1 到 4,294,967,295）
…		…	…

（1）所有偏移量均是相对于包络表起始位置的偏移字节数。

注意: 包络表的起始位置必须为 V 区中的奇数地址，如 VB3001。

4.4.2.2 PTO/PWM 寄存器

在 SM 区中为每个 PTO/PWM 发生器均提供了一些控制寄存器用于存放其配置数据。如下表。

Q0.0	Q0.1	Q0.4	Q0.5	描 述	
SM67.0	SM77.0	SM97.0	SM107.0	PTO/PWM	是否更新周期值：0=否；1=是
SM67.1	SM77.1	SM97.1	SM107.1	PWM	是否更新脉宽值：0=否；1=是
SM67.2	SM77.2	SM97.2	SM107.2	PTO	是否更新脉冲个数：0=否；1=是
SM67.3	SM77.3	SM97.3	SM107.3	PTO/PWM	时基：0=1μs；1=1ms
SM67.4	SM77.4	SM97.4	SM107.4	PWM	更新方法：0=异步更新；1=同步更新
SM67.5	SM77.5	SM97.5	SM107.5	PTO	操作方式：0=单段操作；1=多段操作
SM67.6	SM77.6	SM97.6	SM107.6	功能选择：0= PTO；1=PWM	

SM67.7	SM77.7	SM97.7	SM107.7	PTO/PWM 允许或禁止此功能：0=禁止；1= 允许
Q0.0	**Q0.1**	**Q0.4**	**Q0.5**	**描 述**
SMW68	SMW78	SMW98	SMW108	PTO/PWM 周期值，范围 2~65535
SMW70	SMW80	SMW100	SMW110	PWM 脉宽值，范围 0~65535
SMD72	SMD82	SMD102	SMD112	PTO 脉冲个数，范围 1~4,294,967,295
SMW168	SMW178	SMW218	SMW248	包络表的起始位置（用相对于 VB0 的字节偏移来表示），仅用于 PTO 多段操作。

所有控制字节、周期、脉冲数的缺省值都是 0。用户修改 PTO/PWM 波形的特性的方法是：首先设置相应的控制寄存器，如果是 PTO 多段操作，包络表也得先设置好，然后再执行 PLS 指令。

在 SM 区中也为每个 PTO/PWM 发生器均提供了一个状态字节，用户可以通过访问状态字节来了解 PTO/PWM 发生器的当前状态信息。如下表。

Q0.0	**Q0.1**	**Q0.4**	**Q0.5**	**描 述**
SM66.0	SM76.0	SM96.0	SM106.0	保留
SM66.1	SM76.1	SM96.1	SM106.1	保留
SM66.2	SM76.2	SM96.2	SM106.2	保留
SM66.3	SM76.3	SM96.3	SM106.3	PWM 是否空闲：0=否；1=是
SM66.4	SM76.4	SM96.4	SM106.4	PTO 周期值、脉冲个数设置是否有错误：0=否；1=是 **注：周期值、脉冲个数值必须大于 1。**
SM66.5	SM76.5	SM96.5	SM106.5	PTO 是否由于用户命令而终止：0=否；1=是
SM66.6	SM76.6	SM96.6	SM106.6	保留
SM66.7	SM76.7	SM96.7	SM106.07	PTO 是否空闲：0＝忙；1＝空闲

PTO 空闲位、PWM 空闲位指明了 PTO 输出、PWM 输出是否已经结束。

4.4.2.3 使用 PTO 功能

下面以 PTO0 为例来介绍如何编程使用 PTO 功能。

总体上，使用 PTO 包括两个步骤：设置相关的控制寄存器，初始化 PTO；执行 PLS 指令。

建议用户在工程中尽量编写单独的初始化子程序，这样可以使整个用户工程具有良好的结构。另

外，若有可能的话，尽量在主程序中以 SM0.1 为条件来调用这个初始化子程序，这样该子程序将只在 CPU 上电后的首次扫描中调用并执行一次，可以减少 CPU 的扫描时间。

➢ **执行 PTO（单段操作）**

1) 根据期望的操作来设置控制字节 SMB67。

例如，SMB67 = B#16#85 表明了：

- 允许 PTO/PWM 功能；
- 选择使用 PTO 功能，单段操作；
- 时基选择为 1μs；
- 允许更新脉冲个数和周期值。

2) 将期望的周期值赋给 SMW68。

3) 将期望的脉冲个数赋给 SMD72。

4) （可选）使用 *ATCH* 指令为“PTO0 完成”中断事件（事件号 27）连接一个中断服务程序以实现对该中断事件的快速响应。

5) 执行 PLS 指令来配置并启动 PTO0。

➢ **改变 PTO 周期（单段操作）**

按照如下步骤来改变 PTO0 周期值：

1) 根据期望的操作来设置控制字节 SMB67:

例如，SMB67 = B#16#81 表明了：

- 允许 PTO/PWM 功能；
- 选择使用 PTO 功能，单段操作；
- 时基选择为 1μs；
- 允许更新周期值。

2) 将期望的周期值赋给 SMW68。

3) 执行 PLS 指令来配置并启动 PTO0，具有新周期值的 PTO 就会立即接着启动。

➢ **改变 PTO 脉冲个数（单段操作）**

按照如下步骤来改变 PTO0 输出的脉冲个数：

1) 根据期望的操作来设置控制字节 SMB67:

例如，SMB67 = B#16#84 表明了：

- 允许 PTO/PWM 功能；
- 选择使用 PTO 功能，单段操作；
- 时基选择为 1μs；
- 允许更新脉冲个数。

2) 将期望的脉冲个数赋给 SMD72。

3) 执行 PLS 指令来配置并启动 PTO0，就会立即接着输出新指定个数的脉冲。

➢ **执行 PTO（多段操作）**

1) 根据期望的操作来设置控制字节 SMB67。

例如，SMB67 = B#16#A0 表明了：

- 允许 PTO/PWM 功能；
- 选择使用 PTO 功能
- 选择多段操作；
- 时基选择为 1μs；

2) 将包络表的起始位置（奇数，表示包络表起始地址相对于 VB0 的字节偏移）赋给 SMW168。

3) 设置包络表中的相关数值。

4) （可选）使用 *ATCH* 指令为“PTO0 完成”中断事件（事件号 27）连接一个中断服务程序以实现对该中断事件的快速响应。

5) 执行 PLS 指令来配置并启动 PTO0。

4.4.2.4 使用 PWM 功能

下面以 PWM0 为例来介绍如何编程使用 PWM 功能。

总体上，使用 PWM 包括两个步骤：设置相关的控制寄存器；执行 PLS 指令。

建议用户在工程中尽量编写单独的初始化子程序，这样可以使整个用户工程具有良好的结构。另外，若有可能的话，尽量在主程序中以 SM0.1 为条件来调用这个初始化子程序，这样该子程序将只在 CPU 上电后的首次扫描中调用并执行一次，可以减少 CPU 的扫描时间。

➢ **使用 PWM**

1) 根据期望的操作来设置控制字节 SMB67。

例如，SMB67 = B#16#D3 表明了：

- 允许 PTO/PWM 功能；
- 选择使用 PWM 功能；
- 选择使用同步更新方式；
- 时基选择为 1μs；
- 允许更新脉宽值和周期值。

2) 将期望的周期值赋给 SMW68。

3) 将期望的脉宽值赋给 SMW70。

4) 执行 PLS 指令来配置并启动 PWM0。

➢ **改变脉宽**

下面描述了如何改变 PWM0 的脉宽。

1) 根据期望的操作来设置控制字节 SMB67。

例如，SMB67 = B#16#D2 表明了：

- 允许 PTO/PWM 功能；
- 选择使用 PWM 功能；
- 选择使用同步更新方式；
- 时基选择为 1μs；
- 允许更新脉宽值。

2) 将期望的脉宽值赋给 SMW70。

3) 执行 PLS 指令来配置并启动 PWM0。

4.4.3 定位控制指令的使用

4.4.3.1 如何修改定位控制指令的经过值

➢ **控制寄存器和状态寄存器**

针对定位控制指令， KS1 在 SM 区中为每路高速输出均分配了一个控制字节，在应用中用户需要注意设置该控制字节。另外，还分配了一个当前值（DINT 型）寄存器，用于存放当前已经输出的脉冲个数（正转时增加，反转时减少）。下表详细描述了这些寄存器。

Q0.0	Q0.1	Q0.4	Q0.5	描 述
SMD212	SMD242	SMD262	SMD226	只读。当前值（正转时增加，反转时减少），表示当前已经输出的脉冲个数。
SMD208	SMD238	SDM258	SMD222	读写。 新当前值。与相应标志位配合，用于修改当前值。
Q0.0	**Q0.1**	**Q0.4**	**Q0.5**	**描 述**
SM201.7	SM231.7	SM251.7	SM221.7	读写。急停标志位。若该位为 1，则表示处于急停状态，不执行任何定位控制指令。 当 PSTOP（急停）指令执行时，该位将自动置 1。用户需要使用程序将该位清 0。
SM201.6	SM231.6	SM251.6	SM221.6	读写。用于决定是否复位当前值 1 - 将当前值清零。 0 - 当前值保持不变。
SM201.5	SM231.5	SM251.5	SM221.5	保留
SM201.4	SM231.4	SM251.4	SM221.4	读写。用于决定是否修改当前值 1 - 将当前值清零。 0 - 当前值保持不变。
SM201.3	SM231.3	SM251.3	SM221.3	方向使能控制位。 1- 禁止方向输出，方向通道作为普通 D0。 0 - 使能方向输出。
SM201.0 ~ SM201.2	SM231.0 ~ SM231.2	SM251.0 ~ SM251.2	SM221.0 ~ SM221.2	保留

- **如何修改当前值**

4 路高速输出通道各有一个当前值寄存器，分别为 SMD212、SMD242、SMD262 和 SMD226，其中存放着相应通道已经输出的脉冲个数。 当前值寄存器是只读值，不允许在程序中直接进行修改。若需要修改当前值，则可以采取如下方法：

- **方法一**

利用复位控制位来将当前值清除为 0。

4 个通道的复位控制位分别为 SM201.6、SM231.6、SM251.6 和 SM221.6。

只要复位控制位为 1， PLC 就会将相应的当前值寄存器清 0。因此复位控制位仅需要保持一个扫描周期即可发挥作用，使用时注意避免复位控制位长时间保持为 1，也尽量避免在运动过程中（包括 PHOME、PREL、PABS、PJOG、PFLO_F 指令正在执行）来复位当前值，以免计数出现误差。

下面以通道 0 为例来说明如何复位当前值：

```
(* Network 0 *)
(*以原点信号为基准，当运动到原点时，要求将当前值清 0.*)
LD      %SM0.0
PHOME
0, %M0.0, %M0.1, %M0.2, %VW0, %VW2, %VW4, %VD6, %VW10, %M0.4, %M0.5, %MB1
(* Network 1 *)
(*PHOME 指令完成后，利用 DONE 标志位将当前值清 0。*)
LD      %M0.4
R_TRIG
ST      %SM201.6
```

- **方法二**

利用下述寄存器，可以将当前值设置为任意值。

Q0.0	Q0.1	Q0.4	Q0.5	描 述
SMD208	SMD238	SDM258	SMD222	读写。新当前值。与相应标志位配合，用于修改当前值。
SM201.4	SM231.4	SM251.4	SM221.4	读写。用于决定是否修改当前值 1 - 将当前值清零。 0 - 当前值保持不变。

以通道 0 为例来说明使用方法：若 SM201.4 为 0，则保持当前值 SMD212 不变。若 SM201.4 为 1，则将 SMD208 中的值赋值给当前值 SMD212。注意尽量避免在运动过程中（包括 PHOME、PREL、PABS、PJOG、PFLO_F 指令正在执行）来修改当前值寄存器，以免当前值计数出现误差。

下面的示例程序是以通道 0 为例，说明如何修改当前值：

```
(* Network 0 *)
(*以原点信号为基准，当运动到原点时，要求将当前值设置为 100.*)
LD      %SM0.0
PHOME
0, %M0.0, %M0.1, %M0.2, %VW0, %VW2, %VW4, %VD6, %VW10, %M0.4, %M0.5, %MB1
(* Network 1 *)
(*PHOME 指令完成后，利用 DONE 标志位来修改当前值。*)
LD      %M0.4
R_TRIG
```

```
MOVE      DI#100, %SMD208
ST        %SM201.4
```

4.4.3.2 定位控制指令运行过程中是否可以改变最高输出频率？

PREL（相对运动）和 PABS（绝对运动）在脉冲输出过程中不会去改变最高输出频率。它们在启动时会读取当前最低频率、最高频率和加减速时间参数值，并根据这些值自动计算适当的加减速段数，然后启动脉冲输出。在脉冲输出过程中，PREL 和 PABS 不会再读取上述参数值，因此这些参数值的变化对于脉冲输出没有影响。

PJOG（点动）指令在执行过程中会实时读取输入频率参数（*MAXF*）值，并根据新的频率值来调整输出脉冲的频率。

PHOME（回原点）指令在运行到最高频率段之后、在回原点和原点信号产生之前，会实时读取最高频率参数（*MAXF*）值，并根据新的频率值自动计算加速或者减速段数，然后加速或者减速运行到新的频率值再维持匀速输出。

4.5 CAN 总线的使用

<table>
<tr><th>型号</th><th colspan="2">CAN 通讯口</th><th>扩展协议</th><th>自由通信</th><th>Kinco 运动控制协议</th><th>CANOpen 主站</th><th>CANOpen 从站</th></tr>
<tr><td>KS105C1-16DT</td><td>1*CAN 接口（命名为 CAN）</td><td>CAN1</td><td>不支持</td><td colspan="4">支持
注意：只能任选其一使用。</td></tr>
<tr><td rowspan="3">KS105C2-16DT</td><td rowspan="3">2*CAN 接口（分别命名为 CAN1、CAN2）</td><td rowspan="2">CAN1</td><td>支持（最多可带 14 个扩展模块）</td><td colspan="2">支持</td><td colspan="2">不支持</td></tr>
<tr><td colspan="5">注意：CAN1 接口连接扩展模块时，CAN 通讯功能失效，
即 CAN1 接口的扩展协议和 CAN 通讯功能只能任选其一。</td></tr>
<tr><td>CAN2</td><td>不支持</td><td colspan="3">支持
注意：只能任选其一使用。</td><td>不支持</td></tr>
</table>

4.5.1 硬件接线

KS105C1-16DT CPU 模块提供了一个 CAN 接口，在两个 RJ45 内直连相通，用户可任意使用其中的一个。接线图参考 2.3 接线图。

KS105C2-16DT CPU 模块提供了二个 CAN 接口，分部在两个 RJ45 内。接线图参考 2.3 接线图。

CAN 总线 PLC 端提供拨码开关选择终端电阻是否启用，详情参考产品丝印标示！

需要注意一个误区，模块提供的接口是 RJ45 接口，当多个模块本身组网时，连接电缆需使用直连网线相连，而不是交叉网线。

4.5.2 扩展总线功能

CAN 通信口支持扩展总线协议，若在用户工程的【硬件配置】中配置了扩展模块，则 CAN 接口将作为扩展总线接口工作，此时用户程序中就只允许同时使用自由通信指令。

CPU 最多允许连接 14 个扩展模块，各扩展模块允许分布式安装，CPU 到最末端一个扩展模块之间的通信电缆总长度不允许超过 30 米。使用扩展总线时，建议将首端的 KS 及最末端的扩展模块的终端电阻都加上，以避免信号反射，增强通信稳定性。另外，采用长距离分布式安装时，扩展通信电缆推荐采用屏蔽双绞线且屏蔽层单端良好接地（控制地），并且通信电缆应远离强干扰源、各种大功率线（包括设备的动力电缆）、开关频繁的脉冲信号线等。

在出厂默认的设置中，CPU 模块在上电时会为每个扩展模块自动分配一个唯一的 ID 并配置各种参数，因此要求 CPU 与所有扩展模块同时上电或者扩展模块全部都先于 CPU 模块上电，否则可能会导致程序执行错误。但为了方便用户的分布式应用，CPU 提供了 EX_ADDR 指令，用户可以通过调用这条指令来修改上述默认配置，从而使扩展模块的使用更灵活。

4.5.2.1 如何使用 EX_ADDR 指令实现扩展模块的分布式应用

EX_ADDR 指令的说明请参见 4.4.7.4 扩展总线指令。

在实际应用中，若扩展模块与 CPU 模块相距很远或者安装于不同的设备上时，可能无法保证默认所需的上电次序。在这种情况下，用户可以按照下述步骤使用 EX_ADDR 指令来修改出厂默认的配置，使得各扩展模块可以在任意时刻上电或者断电而不会引起 PLC 执行程序错误。

1) 在用户工程中，在【硬件配置】中按所需次序依次加入各个扩展模块并按实际需求配置好，并在

程序调用 EX_ADDR 指令（位于指令集的【CAN 指令】组中）。

2) 按【硬件配置】中的次序，将真实的 CPU 和所有扩展模块都连接好，然后按默认的次序上电（CPU 与所有扩展模块同时上电或者扩展模块全部都先于 CPU 模块上电）。

3) 将用户工程下载到 CPU 中。CPU 正常运行后，将 EX_ADDR 指令的参数值修改为 181（十进制），然后让 EX_ADDR 指令执行一次。指令成功执行后，各个扩展模块会自动保存好自己的 ID 和各种参数（比如信号形式、滤波方式等）。

4) 给本 PLC 系统断电。然后用户就可以将扩展模块安装于所需的位置，注意各模块的次序（从 CPU 开始）依然要与【硬件配置】中的次序一致。此后，扩展模块再上电时就会自动读取保存好的数据并自动进入运行状态，无需 CPU 进行配置，因此可以独立于 CPU 在任意时间上电或断电。

5) 若用户需要恢复出厂默认的上电次序，则将程序中 EX_ADDR 指令的参数值修改为 99（十进制），然后让 EX_ADDR 指令执行一次。指令成功执行后，各个扩展模块会清除保存的 ID 和通道参数，以后再上电时就会等待 CPU 自动分配 ID 并配置参数。

4.5.3 Kinco 运动控制功能

Kinco 运动控制功能用于控制 Kinco 公司具有 CAN 接口的运动控制产品（伺服和步进驱动器）。它基于 CANOpen 协议，将与驱动器的 CANOpen 通信细节等进行了封装，并结合实际应用需求，为用户提供了提供了一组运动控制指令和相应的网络配置工具。本功能使用简便，用户即使不熟悉 CANOpen 协议细节，也可以很方便地实现与驱动器的通信并进行定位控制。

本功能最大可以控制 32 台运动控制产品。在实际应用中，用户可根据需要程序空间、网络负荷率等决定实际连接台数。

本功能支持对运动控制产品进行参数上传（下载）、电机锁轴、松轴、回原点、点动（速度模式）、绝对定位、相对定位等操作，暂不支持力矩模式和主从跟随模式等操作。另外，本功能原则上可以用于所有支持标准 CANopen 协议的第三方运动控制产品，**使用前请咨询步科技术人员**。

用户按照如下步骤使用 Kinco 运动控制功能：

1) 在用户工程中，进入【Kinco 运动控制网络配置】向导窗口中完成网络的配置。

2) 根据实际需求调用运动控制指令进行编程。

运动控制指令的说明请参见 4.4.7.1 Kinco 运动控制指令。

3) 将工程下载到 PLC 中，则该 PLC 启动后将作为主站运行，管理整个网络的通信，并且执行定位控制程序。

4.5.3.1 Kinco 运动控制网络配置

Kinco 运动控制功能采用 CANOpen 协议，PLC 作为主站，各个驱动器作为从站。在调用指令之前，用户必须先对实际所用的 CANOpen 网络进行配置。按现场应用的习惯，我们在软件中把从站称为“轴”。

在 Kincobuilder 软件的【工程管理器】中，双击【Kinco 运动控制网络配置】节点即可进入配置窗口，在该窗口中完成网络的配置。

在窗口分了三部分区域:网络节点的树状列表、主站参数和轴（从站）的参数。

- **网络节点树的操作**

在网络节点树中，根节点是【CANOpen 主站】，下面的各个子节点是网络中的轴（从站）。

下方提供了【添加】、【删除】、【复制】和【删除】4 个按钮，同时软件也提供了相应的快捷键和右键菜单功能。用户可以利用这些功能对网络节点进行操作。

- 添加一个新的轴

单击【添加】按钮；或者在任一节点上单击右键，然后执行【添加】菜单命令；或者使用 ALT+N 快捷键。使用上述 3 种方法新增的轴在初始时均采用默认的参数。

- 复制、粘贴

用户可以先复制一个已有的轴，然后粘贴到网络中生成新的轴，新轴除了轴号（从站地址）之外，其它参数都跟被复制的轴保持一致。对于那种网络中所有轴的功能都一样的项目来说，这个功能很方

便。

先单击树中的某个轴选中它，然后单击【复制】按钮，或使用 Ctrl+C 快捷键；或者在某个轴上单击右键，执行【复制】菜单命令。这几种方法都可以复制这个轴。

复制完成后，再单击【粘贴】按钮，或者使用 Ctrl+P 快捷键，或者在任一轴上单击右键并执行【粘贴】菜单命令，都可以在网络中生成新的轴。

- 删除一个轴

  先单击某个轴选中它，然后单击【删除】按钮，或者使用 DELETE 快捷键，都可以删除这个轴。在某个轴上单击右键并执行【删除】菜单命令，也可以删除这个轴

➢ **主站参数**

单击【CANOpen 主站】节点，主站的所有参数将会可以修改，轴（从站）的所有参数将会变灰且不能修改。

- 【波特率】：选择主站所用的波特率。注意网络上所有节点（主站及从站）的波特率都必须一致。
- 【SDO 超时】：主站 PLC 发送 SDO 请求报文之后的超时等待时间，若超过这个时间没有收到相应从站的应答报文，则会报告超时错误。当选择不同波特率时，软件会自动推荐一个 SDO 超时时间，用户可以在这个值基础上修改。

➢ **轴（从站）的参数**

单击某个轴节点，该轴的所有参数将会可以修改，主站的所有参数将会变灰且不能修改。

- 【轴号】：轴的 CANOpen 从站地址，**本系统中从站站号必须从 1 开始连续分配。**
- 【类型】：根据轴的功能不同，用户可选择直线轴或者旋转轴。
- 【编码器分辨率】：轴或者步进驱动器的编码器的分辨率，即编码器旋转一圈所发出的脉冲个数。
- 【每圈机械当量】：电机轴每转动一圈，机械负载所移动的长度（直线轴，mm）或者转动的角度（旋转轴，°）。
- 【节点保护】：设置该轴的节点保护时间。用户可以采用默认值，也可以点击“高级”自行修改。
- 【PDO 禁止时间】：PLC 内为各轴自动建立了多个 PDO 用于传输位置、速度、状态等信息，因轴的位置、速度等变化很快，所以 PDO 发送非常频繁，必须设置 PDO 禁止时间。用户可以采用默认值，也可以自行修改。

➢ **其它操作**

- 【确定】：保存当前界面所配置的参数并退出界面
- 【取消】：只保存当前界面所配置**并且已经点击应用的**参数，然后退出界面
- 【应用】：保存当前界面所配置的参数
- 【轴一览表】：轴一览表主要是用来方便查看**所有已经配置且使能**的轴配置的参数，以便核对


轴一览表

轴号	轴类型	编码器分辨率	机械当里	节点保护时间	节点保护因子	PDO禁止时间
1	直线轴	10000	10.000000毫米	1000 ms	3	10 ms
2	旋转轴	10000	45.000000度	2000 ms	3	20 ms

4.5.4 CANOpen 主站功能

CANOpen 总线具有开放性好、可靠性高、实时性较好、抗干扰能力强、成本低等优势，是工业控制中一种常用的现场总线，目前应用越来越广泛。

4.5.4.1 CANOpen 通信对象简介

CANOpen 应用层和通信规范（CiA DS301）是 CANOpen 协议的核心，适用于所有的 CANopen 设备。 在 DS301 中定义了多种 CANOpen 通信对象，同时也详细描述了这些对象的服务和协议。为了方便用户的应用，下面我们将介绍几种关键的对象及其通信协议。

4.5.4.1.1 网络管理（NMT）

网络管理（NMT）面向 CANOpen 设备，采用了主从模式。NMT 服务可以初始化、启动、监视、复位或者停止 CANOpen 设备。在一个网络内必须存在一个 NMT 主站，主站拥有整个网络的控制权，即网络管理类（NMT）功能。下面介绍几种常用的 NMT 服务。

4.4.4.1.1.1 NMT 节点控制（NMT Node Control）

NMT 主站通过 NMT Node Control 报文来控制各从站的 NMT 状态（包括停止、预操作、操作和初始化）。从站设备必须支持 NMT 节点控制服务。NMT 节点控制报文格式如下：

COB-ID	Byte 0	Byte 1
0x000	CS(Command Specifier)	Node ID

其中， Node ID：目标从站的 ID。若 Node ID 为 0，则表示网络上所有的从站都需要执行本命令。

CS： 命令字，不同数值的含义为：1 表示 启动目标节点；

2 表示 停止目标节点；

128 表示 目标节点进入预操作状态；

129 表示 复位目标节点

130 表示 目标节点复位通信参数

4.5.4.1.1.2 NMT 错误控制（NMT Error Control）

错误控制服务用于检测网络故障,包括节点保护（Node Guarding）和心跳（Heartbeat）两种方式。在实际应用中，必须为一个节点选择一种错误控制方式。

顺便提一下，心跳服务是在 DS301 后期的版本中新增加的，CiA 推荐使用。

- **NMT 节点保护（NMT Node Guarding）**

NMT 主站发送远程帧（无数据）：

COB-ID
0x700 + Node ID

NMT 从站发送如下应答报文：

COB-ID	Byte 0
0x700 + Node ID	Bit7：触发位，必须在每次节点保护应答中交替置“0”或者“1”。 Bit0-6：组合的数值表示从站状态。其中，0 表示 Boot-up，4 表示 STOPPED；5 表示 Operational；127 表示 Pre-Operational。

- **心跳（NMT Node Guarding）**

若一个节点被配置为心跳生产者，它会周期性地发送心跳报文。网络中另外一个或者多个节点作为心跳消费者，来处理各生产者的心跳报文。通常，主站作为心跳消费者，其它从站作为心跳生产者。心跳报文格式如下：

COB-ID	Byte 0
0x700 + Node ID	本节点的状态值。其中，0 表示 Boot-up，4 表示 STOPPED；5 表示 Operational；127 表示 Pre-Operational。

4.5.4.1.2 服务数据对象（SDO，Service Data Object）

SDO 通信是基于“客户机-服务器”模型。

通过使用索引（index）和子索引（sub-index），SDO 使一个 CANOpen 设备（作为客户机）可以直接访问其它 CANOpen 设备（作为服务器）的对象字典中的对象。通常，主站作为客户机。

SDO 有两种传输机制：加速传输，每次最多传输 4 字节数据；分段传输，允许分段传输超过 4 个字节的数据。下面简单介绍一下加速传输机制 SDO 的报文格式。

请求报文，Client -> Server：

COB-ID	Byte 0	Byte 1-2	Byte 3	Byte 4-7
0x600 + Node ID	SDO 命令字	对象索引	对象子索引	数据

应答报文，Server -> Client：

COB-ID	Byte 0	Byte 1-2	Byte 3	Byte 4-7
0x580 + Node ID	SDO 命令字	对象索引	对象子索引	数据

4.5.4.1.3 过程数据对象（PDO，Process Data Object）

PDO 用于传输实时的数据，一个 PDO 报文中最多包含 8 个字节的数据。

PDO 通信是基于“生产者-消费者”模型。以发送数据或者接收数据来区分，PDO 分为发送 PDO（TPDO）和接收 PDO（RPDO）。生产者支持 TPDO，消费者支持 RPDO。

PDO 通信没有协议规定，一个 PDO 报文中包含的内容是预先定义好的。在网络组态时，用户就定义了每个 PDO 的 COB-ID 和其中映射的对象，因此，生产者和消费者都能知道相应 PDO 的内容，从而对报文进行解析。

每个 PDO 在对象字典中由通信参数和映射参数来描述。下面介绍 PDO 的通信参数。

- **COB-ID**

  指明了该 PDO 使用的 COB-ID。

➢ **传输类型**

指明了该 PDO 发送（或接收）的触发方式。它是一个 8 位无符号整数值。

传输类型分为如下几类：

- 同步方式：根据 SYNC 对象的计数值来触发发送（或接收）。传输类型的值为 0 表示“同步，非循环”方式，值为 1-240 表示“同步，循环”方式。
- RTR-Only：仅适用于 TPDO，由接收到的 RTR 报文来触发 PDO 的发送。传输类型的值为 252 意味着接收到 SYNC 和 RTR 之后就发送 PDO。值为 253 意味着接收到 RTR 之后立即发送 PDO。
- 事件驱动：当 CANOpen 设备内部事件发生之后就立即发送 PDO。传输类型值为 254 表示是设备制造商自定义的事件。值为 255 表示是设备子协议和应用层协议定义的事件，一般是指 PDO 内的数据值改变或者定时器定时时间到。

➢ **禁止时间**

禁止时间定义了该PDO连续发送时的最小间隔时间。配置禁止时间是为了避免由于高优先级PDO发送过于频繁，始终占据总线，而使其它优先级较低的报文无法使用总线的问题。

➢ **事件定时器**

用于指定一个定时发送的周期值。它是一个 16 位无符号整数，单位是 ms。PDO 将以该定时值为周期来触发发送。若该数值为 0，则表示不使用事件定时器。

4.5.4.2 使用 CANOpen 主站功能

KS 的 CANOpen 主站功能具有如下特点：

- 采用 CAN2.0A 标准。符合 CANOpen 标准协议 DS301 V4.2.0。
- 支持 NMT 网络管理服务，包括 NMT Node Control 和 NMT Error Control，并作为 NMT 主站。
- 最大支持 32 个 CANOpen 从站。允许用户在 KincoBuilder 中为每个从站配置启动过程；
- 每个从站最多支持 8 个 TPDO 和 8 个 RPDO；总共最多支持 128 个 TPDO 和 128 个 RPDO。
- 支持客户端 SDO，并提供 SDO 读、写指令，SDO 指令支持标准的加速传输模式；
- 支持 CANOpen 预定义的紧急报文。

4.5.4.2.1 CANOpen 网络配置工具

在 KincoBuilder 中，进入【PLC 硬件配置】，在窗口上部的表格中选中 CPU 模块，然后在窗口下部的页面中单击【CANOpen 主站】，就进入了 CANOpen 的网络配置页面。

4.5.4.2.2 处理 EDS 文件

在【CANOpen 主站】->【网络配置】页面中，提供了如下按钮可以对 EDS 文件进行操作：

- 【导入 EDS】：单击此按钮，选择所需的 EDS 文件，就可以将其导入到 Kincobuilder 中并存储。导入一个 EDS 之后，相应的从站设备就在显示在【所有从站模块列表】中，之后才可以进行组态。
- 【删除】：　在下方的【所有从站模块列表】中选择一个从站设备，然后单击【删除】按钮，就可以将该设备从列表中删除，同时也将它的 EDS 文件从 Kincobuilder 中删除。
- 【导出所有 EDS】：可以将 Kincobuilder 中已有的全部从站 EDS 文件合并导出到一个文件中（扩展名为 .ALLEDS）。这个功能在卸载 Kincobuilder 时会比较有用，用户在卸载前可以使用这个功能将所有从站的 EDS 文件备份，以后可以将备份的.ALLEDS 文件直接导入即可。
- 【导入所有 EDS】：可以将一个 EDS 备份文件（扩展名为 .ALLEDS ）导入到 Kincobuilder 中，导入之后该文件中包含的所有从站设备都将显示在下方的【所有从站模块列表】中。

4.5.4.2.3 CANOpen 网络配置过程

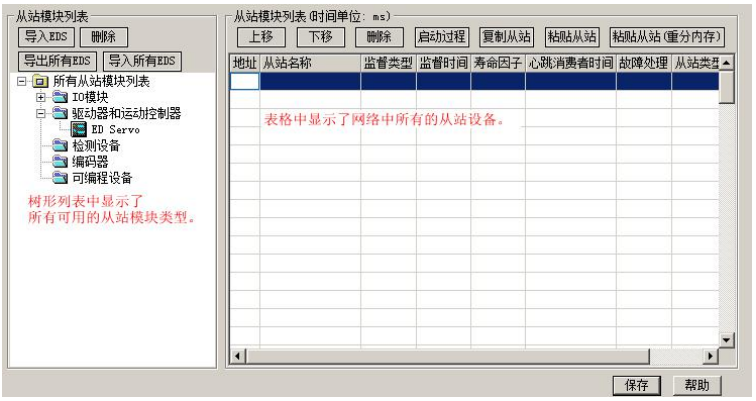
1)　配置全局参数

进入【主站及全局配置】页面，如下图：

- 【波特率】：选择主站所用的波特率。注意网络上所有节点的波特率必须一致。
- 【SDO 超时】：设定主站发送 SDO 请求报文之后的超时等待时间，若超过这个时间没有收到相应从站的应答报文，则会报告错误。SDO 超时值设定一般不需要超过 100ms。
- 【启动时配置各从站】：若选中此项，那么主站除了控制各个从站的 NMT 状态转换外，在启动时主站还会根据各个从站的参数组态情况依次发送相应的配置命令来对各个从站进行配置（如从站的错误控制方式、PDO 映射等）。若不选中此项，则主站仅仅控制各个从站的 NMT 状态转换。

2)　配置各从站

进入【网络配置】页面，继续配置网络上的从站节点及其参数，如下图：



页面中所有的功能按钮，都有相应的右键菜单命令。用户在相关位置单击鼠标右键，就会弹出相应的右键菜单，此时可以使用菜单命令。下面描述配置一个从站的常用过程。

a) 向网络中添加一个从站设备

从左侧的树形列表中双击需要加入网络的从站类型，就会添加一个该类型的从站设备到网络中去，并显示在右侧的表格中。

b) 配置从站设备的站号（ID）、监督类型等参数：

右侧表格中的【地址】列就是从站的站号（ID）。第 1 行是 1 号站的位置。

添加一个从站设备后，就会显示出它的默认配置参数。添加时，Kincobuilder 默认是将设备从上到下依次添加到表格中。用户可以鼠标单击表格中的行来选中一个从站，然后可以单击【上移】、【下移】按钮来调整它的站号，也可以单击【删除】按钮将此设备从网络中删除。

【监督类型】用于配置该节点的 NMT Error Control 方式，包括节点保护和心跳两种方式。若从站设备同时支持这两种方式，推荐优先选择使用心跳方式。

【监督时间】表示前面所选的节点保护方式或者心跳方式的周期值。建议在实际应用中，这个周期值设置不要太小，比如可以设置在 2000 以上。

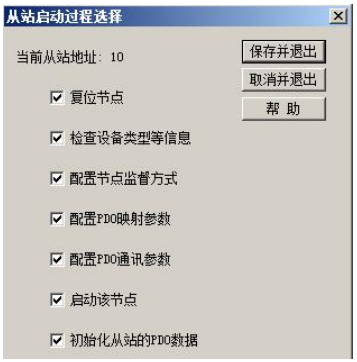
【心跳消费者时间】主站会定时查询是否收到从站的心跳报文，若超过这个“心跳消费者”时间仍然没有收到，则认为该从站已经离线并进行相应的故障处理。建议在实际应用中，这个周期值设置不要太小，比如可以设置在 2000 以上。

【故障处理】用于选择当主站检测到该从站故障后采用的处理方式，包括“无”、“停止节点”和“停止网络”三种选项。主站能够检测的故障包括 SDO 命令超时没有应答、节点保护或者心跳报文

超时、收到从站发送的部分类型的紧急报文等。

c) 配置从站的启动过程：

鼠标单击表格中的一个从站，然后单击【启动过程】，就可以选择在网络启动过程中主站需要对该从站进行何种配置。



【复位节点】：主站在向从站发送配置命令之前，是否先发送“复位节点”命令。

【检查设备类型】：主站在向从站发送配置命令之前，是否先读取设备信息进行检查。

【配置节点监督方式】：主站是否需要配置从站的监督类型及其参数。

【配置 PDO 映射参数】：主站是否需要配置从站的 PDO 映射参数。

【配置 PDO 通信参数】：主站是否需要配置从站的 PDO 通信参数。

【启动该节点】：配置完成后，主站是否需要向该从站发送“启动节点”命令。

【初始化从站的 PDO 数据】：在启动该从站后，主站是否需要把该从站全部 RPDO 中的数据清 0 并立即发送一次。

d) 配置各从站的 PDO：

进入【对象字典映射】页面，为网络中所有的从站配置 PDO。

页面左侧部分【已添加从站列表】显示了已经加入到网络的所有从站，以及各从站对象字典中允许映射到 PDO 中的对象。其中，【发送 PDO】中的对象只能映射到该从站的 TPDO 中，【接收 PDO】列表中的对象只能映射到该从站的 RPDO 中。

在【已添加从站列表】中鼠标单击一个从站，那么在右侧就会显示出该从站所有的 PDO，用户可以对各 PDO 的进行配置：

- 通信参数

在右侧的表格中，选中一个 PDO，可以修改其定时时间、禁止时间等通信参数。

其中，从站中前 4 个的 TPDO 和 RPDO 的 COB-ID 不允许修改，采用了 DS301 中预定义连接集中的默认值。后 4 个的 TPDO 和 RPDO 允许用户自己输入合法的 COB-ID 值。

- 映射参数

在左侧的对象列表中，双击一个对象，就会将这个对象加入到了当前 PDO 中，同时 Kincobuilder 会自动为这个对象分配一个 PLC 的 V 区地址，比如 VW1006，用户在程序中操作这个 V 区地址就相当于操作相应的对象了。

3） 复制从站、粘贴从站

在【网络配置】页面中，提供了【复制从站】、【粘贴从站】和【粘贴从站（重分内存）】这 3 个按钮。如下图。

【复制从站】：选中一个已经配置好的从站，然后单击此按钮，就会复制该从站的所有信息（其中包括所有 PDO 的通信参数、映射参数等）。如果所选从站没有配置任何 PDO，那么复制失败并提示相应信息。

【粘贴从站】：复制成功一个从站后，单击选中表格中的一个空行，然后单击此按钮，就会将刚才复制的从站信息粘贴到该行并生成一个新从站。注意：新从站 PDO 中的各映射对象对应的 PLC 内存地址，还是保持与源从站中的一致，没有重新分配，用户需要自己修改。

【粘贴从站（重分内存）】：操作方法与【粘贴从站】一样，但是不同之处是新从站 PDO 中各映射对象的 PLC 内存地址会自动进行分配，无需用户修改。

4.5.5 CANOpen 从站功能

4.5.5.1 概述

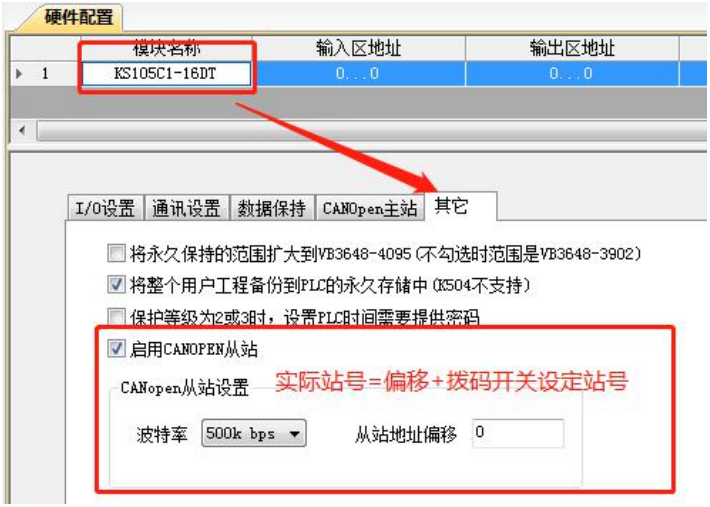
KS 支持 CANOpen 从站功能，具有如下特点：

- CAN 接口采用 CAN2.0A 标准。
- 符合 DS301 V4.2.0 和 DS405 V2.0 协议
- 支持 NMT 网络管理服务，支持心跳协议。
- 支持 SDO Server，加速传输模式。
- 支持最多 16 个 TPDO 和 16 个 RPDO，PDO 的通信参数和映射参数均可自由配置。

注意：使用时，PDO的序号必须连续，中间不允许有间隔，比如，使用TPDO1、TPDO2、TPDO3 是合法的，但是若配置为使用TPDO1、TPDO3、TPDO4则是非法的。

4.5.5.2 启用 CANOpen 从站功能

在用户工程中，进入【PLC 硬件配置】，在 CPU 模块的【其它】页面中，用户可以配置 CANOpen 从站功能。



【启用 CANOPEN 从站】：若选中此项，则本 PLC 将作为 CANOpen 从站。

【波特率】：选择 CAN 接口所用的波特率。一个网络中所有节点的波特率都必须一致。

【从站地址偏移】：本从站的真实站号等于“第 1 至 3 位拨码开关的组合值”加上“从站地址偏移”。例如，假如第 1 到 3 位拨码开关的组合值是 5，从站地址偏移设置为 8，则本从站站号就是 13。

4.5.5.3 对象字典

用户可以到步科公司的官网免费下载本从站设备的 eds 文件。

在 KincoBuilder 软件中已经集成了本设备的 EDS 文件：在硬件配置的【CANOpen 主站】->【网络配置】页面中，本设备就显示在“所有从站模块列表”的“运动控制器和 PLC”组中，名字显示为“Kinco KPLC for Kinco”。用户若使用 Kinco PLC 作为主站及从站，那么无需再次导入 eds 文件。

本设备暂时开放了如下内存区域作为 CANOpen 通信对象，可以映射到 PDO 中使用。若需要使用更大的内存区域，请联系步科公司提供另外的 eds 文件即可。

对象索引	数据类型	PLC 内存区域	数量	读写属性
0xA040	BYTE	IB0～IB9	10	只读
0xA0C0	INT16	AIW0～AIW18	10	只读
0xA4C0	BYTE	QB0～QB9	10	可读可写
0xA4C1	BYTE	MB0～MB9	10	可读可写
0xA4D2	BYTE	VB508～VB517	10	可读可写

0xA540	INT16	AQW0～AQW18	10	可读可写
0xA550	INT16	VW0～VW18	10	可读可写
0xA641	INT32	VD1016～VD1052	10	可读可写
0xA6C2	FLOAT	VR2032～VR2068	10	可读可写

4.5.6 CAN 自由通信功能

KS 里提供了一组 CAN 通信指令，可以初始化 CAN 口、通过 CAN 口收发数据等，用户可以使用这些指令来自由编写通信程序与其它设备进行通信。CAN 通信指令支持 CAN2.0A 和 CAN2.0B 标准，另外这些指令只支持数据帧，不支持远程帧。CAN 数据帧格式如下：

ID	字节 1-8
11 位（CAN2.0A，标准帧）或 29 位（CAN2.0B，扩展帧）	1-8 字节长度的数据

CAN 自由通信功能可以与其它的通信功能（扩展总线、Kinc 运动控制、CANOpen 主站和从站）同时使用，但需要注意通信波特率必须保持一致。

注意：自由通信报文的 ID 号不允许使用 CANOpen 协议中的 COB-ID 号！

CAN 通信指令的详细说明请参见 4.4.7.3　CAN 自由通信指令

4.5.7 CAN 总线相关指令

4.5.7.1　Kinco 运动控制指令

4.5.7.1.1 综述

下述指令位于指令集的【Kinco 运动控制】组中。

名称	功能描述
MC_RPARAS	读取轴驱动器内的参数（具体见下文的参数表）
MC_WPARAS	修改轴驱动器内的参数
MC_POWER	控制锁轴、松轴
MC_RESET	复位轴上的错误信息，将轴状态置为静止等待状态
MC_HOME	控制轴回原点
MC_JOG	控制轴点动
MC_MABS	控制轴进行绝对定位运动

MC_MREL	控制轴进行相对定位运动
MC_MIOT	读取目标轴的序列号、软件版本、IIT、温度等设备信息

➢ **注意事项**

用户在使用这些指令时，需要注意如下几点：

- 在一个用户工程中，最大允许的轴数：KS 和 KW 系列 16 个，KM 系列 128 个。
- 在一个用户工程中，使用的专用指令总数量限制：KS 和 KW 系列最大 192 个，KM 系列最大 1024 个。其中，MC_MIOT 指令每个轴只允许使用 1 个。
- 对于同一个轴，当某个专用指令正在执行、尚未完成时，不允许再启动执行另外的专用指令。假如此时用户程序启动了另外一个专用指令，那么这个指令会直接结束并报告错误。
- 对于同一个轴，MC_MIOT 指令优先级最低：若其它指令正在运行，则 MC_MIOT 指令不会执行；若 MC_MIOT 指令正在执行过程中，程序又启动里的其它指令，则 MC_MIOT 指令将直接终止。
- 对于同一个轴，用户程序执行运动指令（不含读写参数指令）之前，必须首先执行 MC_POWER 指令进行锁轴，在锁轴成功后才可以继续执行回原点、相对运动、绝对运动或者点动指令。若没有锁轴，那么执行这几种指令时会直接结束并报告错误。
- 对于同一个轴，用户程序使用 MC_RESET 指令进行复位，复位成功后，轴将处于松轴的静止等候状态，需先执行 MC_POWER 指令进行锁轴才可以继续执行回原点、相对运动、绝对运动或者点动指令。
- 对于回原点、相对运动、绝对运动或者点动指令，其使用的加减速度为驱动器内部设定的加减速度，用户也可通过 MC_WPARAS 指令设定。
- 在用户程序中调用各个运动控制指令的输出结果互不相干。若某条指令执行出错，则它的输出参数 ERRID 将给出错误代码，且这个错误结果直到下一次该指令再次执行后才会再次刷新，其它指令执行的结果不会影响到指令的执行结果！
- 总线掉线后（MC_STATE 指令的 ONLINE 输出结果为 1），为安全考虑，本组指令不会自动重连执行！用户必须排除错误后，断电重启 PLC 才可重新执行指令！

➢ **指令输出参数 ERRID**

每个指令均提供了 ERRID 输出参数。若指令执行成功，则 ERRID 输出为 0。若指令执行失败，则 ERRID 会被设置为不同的错误码值来说明错误的原因。

下面是各错误码值的说明（注意，**此处的错误码不适合于 MC_RPARAS 和 MC_WPARAS 指令，**

这两个指令的错误码另有特殊含义，请另外参考指令说明）：

错误码	描 述
0	无错误
1	目标轴没有使能，或者网络中不存在该轴
2	目标轴没有处于锁轴状态。
3	目标轴正在执行其它运动控制指令，没有处于静止状态。
4	PLC 内部的 CAN 报文发送缓冲区满，不能发送 CAN 报文
5	PLC 向目标轴发送了 SDO 请求报文，但超时没有收到回应
6	PLC 向目标轴发送了 SDO 请求报文，但是收到了错误的回应报文
7	指令正常执行了，但是 PLC 持续检测目标轴返回的状态，最终没有检测到正确的状态值

4.5.7.1.2 MC_RPARAS（读取参数）和 MC_WPARAS（修改参数）

本组指令的目的是方便用户批量操作驱动器参数，比如用户可以在调试初期一次性设定驱动器的参数。具体参数如何设置请查询驱动器操作手册，**设置不当则有可能运转异常，请谨慎操作。**

1) 可操作的驱动器参数列表

通过驱动器读写指令，可对驱动器的下列参数进行操作，所有参数均可读可写。每个指令一次最多操作 32 个参数。表中工艺数据类型，REAL 表示单精度浮点数，UINT32 表示无符号 32 位数，INT32 表示有符号 32 位数，其它以此类推。

表中的“序号”值是**固定的**，每个参数均有一个序号，用户在指令中输入序号就可以操作相应的参数。“工艺单位”指的是在指令参数中采用的单位，“驱动器取值范围”指的是驱动器内部的取值范围（本指令在内部会自动将用户需要的实际工艺参数值转换为驱动器内部使用的数据格式，比如加速度、速度、位置等）。

序号	参数名称	CANOpen 对象	工艺数据类型	工艺单位	驱动器内取值范围
0	**梯形加速度**	0x60830020	REAL	直线轴：mm/s² 旋转轴：1/s²	[0,268435455]
1	**梯形减速度**	0x60840020			
2	**找原点速度**	0x60990120	REAL	直线轴：mm/min 旋转轴：度/min	[-2147483648, 2147483647]
3	**找原点模式**	0x60980008	INT8	DEC	[-128,127]

4	**速度环比例增益 0**	0x60F90110	UINT16	DEC	[0,32767]
5	**速度环积分增益 0**	0x60F90210	UINT16	DEC	[0,32767]
6	**位置环比例增益 0**	0x60FB0110	REAL	HZ	[0,32767]
7	**位置环速度前馈**	0x60FB0210	REAL	%	[0,1024]
8	**速度环比例增益 1**	0x23400410	UINT16	DEC	[0,32767]
9	**速度环积分增益 1**	0x23400510	UINT16	DEC	[0,32767]
10	**位置环比例增益 1**	0x23400610	REAL	HZ	[0,32767]
11	**目标电流限制**	0x60730010	UINT16	DEC	[0,2048]
12	**最大速度限制**	0x607F0020	REAL	直线轴：mm/min 旋转轴：度/min	[-2147483648, 2147483647]
13	**原点偏移模式**	0x60990508	UINT8	无单位	[0,255]
14	**电机方向**	0x607E0008	UINT8	无单位	0 和 1
15	**电机型号**	0x64100110	UINT16	无单位	[0,65535]
16	**软限位正设置**	0x607D0120	REAL	直线轴：mm 旋转轴：度	[-2147483648, 2147483647]
17	**软限位负设置**	0x607D0220			
18	**平滑滤波**	0x60FB0510	UINT8	无单位	[0,255]
19	**最大跟随误差**	0x60650020	UINT32	DEC	[0,268435455]
20	**目标位置窗口**	0x60670020	UINT32	DEC	[0,268435455]
21	**位置窗口时间**	0x60680010	UINT16	DEC	[0,32767]
22	**速度反馈滤波**	0x60F90508	REAL	HZ	[0,45]
23	**速度反馈模式**	0x60F90608	UINT8	无单位	[0,85]
24	**输入口极性**	0x20100110	UINT8	无单位	[0,255]
25	**输入口 1 功能**	0x20100310	UINT16	无单位	[0,65535]
26	**输入口 2 功能**	0x20100410	UINT16	无单位	[0,65535]
27	**输入口 3 功能**	0x20100510	UINT16	无单位	[0,65535]
28	**输入口 4 功能**	0x20100610	UINT16	无单位	[0,65535]
29	**输入口 5 功能**	0x20100710	UINT16	无单位	[0,65535]

30	**输入口 6 功能**	0x20100810	UINT16	无单位	[0,65535]
31	**输入口 7 功能**	0x20100910	UINT16	无单位	[0,65535]
32	**输入口 8 功能**	0x20101D10	UINT16	无单位	[0,65535]
33	**输出口极性**	0x20100D10	UINT8	无单位	[0,255]
34	**输出口 1 功能**	0x20100F10	UINT16	无单位	[0,65535]
35	**输出口 2 功能**	0x20101010	UINT16	无单位	[0,65535]
36	**输出口 3 功能**	0x20101110	UINT16	无单位	[0,65535]
37	**输出口 4 功能**	0x20101210	UINT16	无单位	[0,65535]
38	**输出口 5 功能**	0x20101310	UINT16	无单位	[0,65535]
39	**输出口 6 功能**	0x20101E10	UINT16	无单位	[0,65535]
40	**输出口 7 功能**	0x20101F10	UINT16	无单位	[0,65535]
41	**齿轮前脉冲数据**	0x25080420	INT32	DEC	[-2147483648, 2147483647]
42	**脉冲模式**	0x25080308	UINT8	无单位	[0,255]
43	**保存参数**	0x10100120	UINT32	无单位	仅 16#65766173 有效
44	**初始化参数**	0x10110120	UINT32	无单位	仅 16#64616f6C 有效

2) ERRID 参数说明

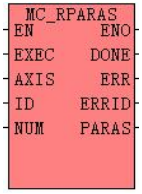
读、写参数指令均提供了 *ERRID*（DWORD 型）输出参数。

这个参数值是错误码，表示了指令执行过程中发生过的错误。

错误码	**含 义**
0xFFFFFFFF	发生过导致指令无法执行的错误，包括： 1）用户输入的轴号错误、参数数量错误 2）有其它 Kinco 专用指令正在运行 3）指令要操作 32 个参数，结果这 32 个参数均操作失败

其它值	ERRID 的每个位均表示了对应参数的操作结果，每个位与 *ID* 参数序号表中指定的参数一一对应：bit0 表示本次操作第 1 个参数的结果，bit1 表示第 2 个参数的操作结果，以此类推。某位值为 1，表示对应参数操作失败，否则表示对应参数操作成功。

3) MC_RPARAS（读取参数）

	名称	指令格式	适用于
LD	MC_RPARAS		• KS • KW103 • KW203

参数	输入/输出	数据类型	允许的内存区	描　述
EXEC	输入	BOOL	M、V、L、SM	若检测到 *EXEC* 的上升沿，则指令被触发执行。
AXIS	输入	INT	V、M、L、常量	目标轴的轴号（也就是 CANOpen 从站的地址）
ID	输入	BYTE	V、M、L	要读取的参数的序号表的起始地址。
NUM	输入	INT	V、M、L、常量	要读取的参数数量
DONE	输出	BOOL	M、V、L	完成标志位。指令执行完成时，*DONE* 由 0 跳变到 1。
ERR	输出	BOOL	M、V、L	错误标志位。若指令执行时发生错误，则被置 1。
ERRID	输出	DWORD	V、M、L	错误码
PARAS	输出	DWORD	V、M、L	读取到的所有参数值的存放起始地址。

AXIS 和 NUM 必须同时为常量类型或同时为内存类型，另外，ID 和 PAPAS 参数共同组成了一个长度可变的内存块，此内存块必须全部位于合法的内存区域，否则结果不可预期。

ID、*PARAS*、*NUM* 这 3 个参数共同组成了一个参数表。其中 *ID* 是序号表的起始地址，从这个地址开始连续依次存储着待操作的各个参数的序号（即前文参数列表中的“序号”），每个序号占用 1 个

字节；*PARAS* 是参数值表的起始地址，从这个地址开始连续依次存储着读取到的各个参数的数值，每个数值均占用 4 个字节；*NUM* 是待操作的参数的数量。 例如，假定 *ID* 参数是 VB100，*PARAS* 参数是 VD1200，*NUM* 参数是 3，那么 VB100、VB101、VB102 分别存放着本次要操作的 3 个参数的序号，当指令执行完成后，读取的 3 个参数值分别存放在 VD1200、VD1204、VD1208 中。

对于 *PARAS* 要注意，虽然参数值表统一采用了 DWORD 地址，但各个工艺参数的实际数据类型并不相同，因此表中在用户程序中用户要根据实际数据类型来处理参数表中的数据。

- 若实际工艺数据类型为 REAL 型，那么直接以浮点数地址来操作参数内存即可。比如，参数值存放于 VD1200，那么可以直接操作 VR1200。因为 VD1200 与 VR1200 在 PLC 中实际占用的是同一片内存地址。
- 若实际工艺数据类型是 REAL 之外的其它数据类型，并且相应的参数内存没有在全局变量表中强制定义数据类型，那么直接读取参数内存即可，因为指令会自动处理各种有符号和无符号整数。比如，参数值存放于 VD1200，而实际类型是 INT32 或者 UINT32，那么直接操作 VD1200。

- LD 格式指令说明

若 *EN* 为 1，那么在 *EXEC* 输入的上升沿，该指令被触发执行，指令根据 *ID*、*NUM* 指定的待读取的参数表，依次发送 SDO 给驱动器来读取相应的对象，并将读到的数据依次放入 *PARAS* 指定的数值表中，同时将 *ERRID* 相应位设置为 0。若某个参数的 SDO 响应错误或者超时无响应，则 *PARAS* 中相应地址的数据保持不变，同时将 *ERRID* 相应位设置为 1，然后继续读取下一个参数。 当读取完成所有参数后，DONE 被置 1，ERR、ERRID 根据执行结果来设置为不同的值。

若 EN 为 0，则指令不执行。 在指令执行过程中 EXEC 变为 0，则指令会停止读取尚未完成的参数,并将 DONE 置 1，ERR、ERRID 维持已经执行的结果。

若在指令启动时，PLC 检测到错误（比如轴未使能、轴正在执行其它指令等），则直接退出，将 DONE、ERR 置 1，ERRID 设置为相应的错误码。

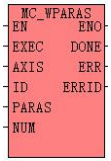
➢ 示例

本例子采用 IL 格式。在 Kincobuilder 中，先在【工程】菜单中选择【IL】格式，然后可以将示例复制粘贴到编辑器中，然后再选择【LD 格式】，程序就可以显示为 LD 格式了。

```
(* Network 0 *)
(*设置参数表，指明本次要读取参数 0、3 和 8.*)
```

```
LD          %SM0.0
MOVE        B#0, %VB100
MOVE        B#3, %VB101
MOVE        B#8, %VB102
(* Network 1 *)
(*调用指令。此次，AXIS、NUM 参数均为常量，它们也支持全内存地址的格式。*)
LD          %M0.0
MC_RPARAS %M1.1, 1, %VB100, 3, %M1.2, %M1.3, %MD8, %VD1200
(* Network 2 *)
(*读取的参数值依次存放于 PARAS 参数知道的参数值表中。表中第 1 个数据是读取的第 1 个参数值，即参数 0，因为它是 REAL 型，所以读取浮点型内存地址。*)
LD          %SM0.0
MOVE        %VR1200, %VR300
(* Network 3 *)
(*表中第 2 个数据是读取的第 2 个参数值，即参数 0。这个参数是有符号 8 位数，因为 PLC 中没有提供这种数据类型，所以按整数进行处理。*)
LD          %SM0.0
DI_TO_I %VD1204, %VW304
(* Network 4 *)
(*表中第 3 个数据是读取的第 3 个参数值，即参数 8。这个参数是无符号 16 位数，但最大范围是 32767，所以程序中按 INT 或者 WORD 型处理均可，但最好先判断一下数值是否在允许范围内。*)
LD          %SM0.0
DI_TO_I %VD1208, %VW308
NE          %VW308, 0
ST          %M3.0
```

4) MC_WPARAS（修改参数）

	名称	指令格式	适用于
LD	MC_WPARAS		• KS • KW103 • KW203

参数	输入/输出	数据类型	允许的内存区	描　述
EXEC	输入	BOOL	M、V、L、SM	若检测到 *EXEC* 的上升沿，则指令被触发执行。
AXIS	输入	INT	V、M、L、常量	目标轴的轴号（也就是 CANOpen 从站的地址）
ID	输入	BYTE	V、M、L	要修改的参数的序号表的起始地址。
PARAS	输出	DWORD	V、M、L	读修改的所有参数值的存放起始地址。
NUM	输入	INT	V、M、L、常量	要修改的参数数量
DONE	输出	BOOL	M、V、L	完成标志位。指令执行完成时，*DONE* 由 0 跳变到 1。
ERR	输出	BOOL	M、V、L	错误标志位。若指令执行时发生错误，则被置 1。
ERRID	输出	DWORD	V、M、L	错误码

AXIS 和 NUM 必须同时为常量类型或同时为内存类型，另外，ID 和 PAPAS 参数共同组成了一个长度可变的内存块，此内存块必须全部位于合法的内存区域，否则结果不可预期。

ID、*PARAS*、*NUM* 这 3 个参数共同组成了一个参数表。其中 *ID* 是序号表的起始地址，从这个地址开始连续依次存储着待操作的各个参数的序号（即前文参数列表中的“序号”），每个序号占用 1 个字节；*PARAS* 是参数值表的起始地址，从这个地址开始连续依次存储着各个参数的数值，每个数值均占用 4 个字节；*NUM* 是待操作的参数的数量。 例如，假定 *ID* 参数是 VB100，*PARAS* 参数是 VD1200，*NUM* 参数是 3，那么 VB100、VB101、VB102 分别存放着本次要操作的 3 个参数的序号，VD1200、VD1204、VD1208 分别存放着待修改的参数值。

对于 *PARAS* 要注意，虽然参数值表统一采用了 DWORD 地址，但各个工艺参数的实际数据类型并不相同，因此表中在用户程序中用户要根据实际数据类型来给参数表中相应地址赋值。

- 若实际工艺数据类型为 REAL 型，那么直接以浮点数地址来操作参数内存即可。比如，参数值期望存放于 VD1200，那么可以直接操作 VR1200。VD1200 与 VR1200 在 PLC 中实际占用的是同一片内存地址，该指令会自动做类型转换。
- 若实际工艺数据类型是 REAL 之外的其它数据类型，那么直接操作参数内存即可，指令会根据参数的数据类型自动做类型转换。比如，参数数据类型是 UINT16，那么就直接赋一个合法的数值给 VD1200 即可。

- LD 格式指令说明

若 *EN* 为 1，那么在 *EXEC* 输入的上升沿，该指令被触发执行，指令根据 *ID*、*PARAS*、*NUM* 指定的参数表，依次将 PARAS 中的数值通过 SDO 发送给驱动器来修改相应的对象，同时将 *ERRID* 相应位设置为 0。若某个参数的 SDO 响应错误或者超时无响应，则将 *ERRID* 相应位设置为 1，然后继续写入下一个参数。 当写入完成所有参数后，DONE 被置 1，ERR、ERRID 根据执行结果来设置为不同的值。

若 EN 为 0，则指令不执行。若在指令执行过程中 EN 变为 0，则指令会停止写入尚未完成的参数, 并将 DONE 置 1，ERR、ERRID 维持已经执行的结果。

若在指令启动执行时，PLC 检测到错误（比如轴未使能、轴正在执行其它指令等），则直接退出，将 DONE、ERR 置 1，ERRID 设置为相应的错误码。

➢ **示例**

本例子采用 IL 格式。在 Kincobuilder 中，先在【工程】菜单中选择【IL】格式，然后可以将示例复制粘贴到编辑器中，然后再选择【LD 格式】，程序就可以显示为 LD 格式了。

```
(* Network 0 *)
(*设置参数表，指明本次要读取参数 0、3 和 8.*)
LD          %SM0.0
MOVE        B#0, %VB100
MOVE        B#3, %VB101
MOVE        B#8, %VB102
(* Network 1 *)
(*设置各个参数待写入的数值。注意数据类型。*)
LD          %SM0.0
MOVE        1200.0, %VR1000
MOVE        DI#8, %VD1004
MOVE        DI#2000, %VD1008
(* Network 2 *)
(*调用指令*)
LD          %SM0.0
MC_WPARAS %M0.1, 1, %VB100, %VD1000, 8, %M0.2, %M0.3, %MD4
```

4.5.7.1.3 MC_POWER（锁轴和松轴）

	名称	指令格式	适用于
LD	MC_POW ER		• KS • KW103 • KW203

参数	输入/输出	数据类型	允许的内存区	描　述
EXEC	输入	BOOL	M、V、L、SM	上升沿触发锁轴命令，下降沿触发松轴命令。
AXIS	输入	INT	V、M、L、常量	目标轴的轴号（也就是 CANOpen 从站的地址）
DONE	输出	BOOL	M、V、L	完成标志位。指令执行完成时，*DONE* 由 0 跳变到 1。
ERR	输出	BOOL	M、V、L	错误标志位。若指令执行时发生错误，则被置 1。
ERRID	输出	BYTE	V、M、L	错误码

- LD 格式指令说明

若*EN*为1，那么在*EXEC*的上升沿会触发执行锁轴命令，在*EXEC*的下降沿会触发执行松轴命令。

在指令执行时，PLC首先发送命令控制轴进入待操作状态，并在5S超时时间内检查驱动器实际返回状态，若成功执行则表示指令执行成功，则DONE被置1，ERR被置0，ERRID被置0。若发生错误（可能是指令本身执行错误，也可能是执行过程中驱动器未正确执行动作的错误，详见错误码）则指令执行失败，指令都会停止执行，同时将DONE被1，ERR置1，ERRID赋值为相应的错误代码。

若 *EN* 为 0，则指令不执行。

4.5.7.1.4 MC_RESET（复位驱动器报警）

	名称	指令格式	适用于
LD	MC_RESET		• KS • KW103 • KW203

参数	输入/输出	数据类型	允许的内存区	描　述
EXEC	输入	BOOL	M、V、L、SM	上升沿触发本指令执行一次。
AXIS	输入	INT	V、M、L、常量	目标轴的轴号（也就是 CANOpen 从站的地址）
DONE	输出	BOOL	M、V、L	完成标志位。指令执行完成时，*DONE* 由 0 跳变到 1。
ERR	输出	BOOL	M、V、L	错误标志位。若指令执行时发生错误，则被置 1。
ERRID	输出	BYTE	V、M、L	错误码

当轴在运行过程中出错时，可以调用本指令，将轴上的错误信息复位，同时将轴置为松轴静止等待状态。复位成功后如需继续执行其他运动指令，则应首先调用 MC_POWER 指令锁轴！

注意：本指令仅复位驱动器的报警错误信息，并不复位各指令的输出结果！

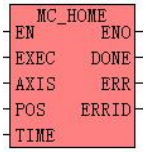
- LD 格式指令说明

若*EN*为1，那么在*EXEC*的上升沿会触发执行本指令。

在指令执行时，PLC首先发送指令复位驱动器报警，并在2秒钟超时时间内检查驱动器实际状态，若成功复位，则表示指令执行成功，则DONE被置1，ERR被置0，ERRID被置0。若发生错误（可能是指令本身执行错误，也可能是执行过程中驱动器未正确执行动作的错误，详见错误码）则指令执行失败，指令都会停止执行，同时将DONE被1，ERR置1，ERRID赋值为相应的错误代码。

若 *EN* 为 0，则指令不执行。

4.5.7.1.5 MC_HOME（回原点）

	名称	指令格式	适用于
LD	MC_HOME		• KS • KW103 • KW203

参数	输入/输出	数据类型	允许的内存区	描　述
EXEC	输入	BOOL	M、V、L、SM	上升沿触发本指令执行一次；下降沿触发暂停运动
AXIS	输入	INT	V、M、L、常量	目标轴的轴号（也就是 CANOpen 从站的地址）
POS	输入	REAL	V、M、L、常量	原点的偏移位置，单位：mm 或者°。
TIME	输入	DWORD	V、M、L、常量	超时时间，若在此时间内没有找到原点，则报错退出。
DONE	输出	BOOL	M、V、L	完成标志位。指令执行完成时，*DONE* 由 0 跳变到 1。
ERR	输出	BOOL	M、V、L	错误标志位。若指令执行时发生错误，则被置 1。
ERRID	输出	BYTE	V、M、L	错误码

执行本指令，可以使目标轴回原点。*POS*参数设置了原点坐标的偏移值。

注意：本指令使用驱动器内部回原点模式，需在驱动器首先设好 60980008 回原点模式（也可通过 MC_WPARAS 指令写入），详情请参考驱动器使用手册。

- LD 格式指令说明

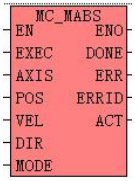
若*EN*为1，那么在*EXEC*的上升沿会触发执行本指令。

在指令执行时，PLC首先发送命令让轴开始找原点；发送完成后，检查驱动器返回的状态。检查将持续*TIME*（用户设定的超时时间，单位ms），若在此时间内轴成功找到原点，则表示指令执行成功，此时DONE被置1，ERR被置0，ERRID被置0。若发生错误（可能是指令本身执行错误，也可能是执行过程中驱动器未正确执行动作的错误，详见错误码）则指令执行失败，指令都会停止执行，，同时将DONE

被1，ERR置1，ERRID赋值为相应的错误代码。

若 *EN* 为 0，则指令不执行。若在执行过程中 *EN* 变为 0，则指令将停止执行，轴处于静止锁轴等候状态。

4.5.7.1.6 MC_MABS（绝对运动）

	名称	指令格式	适用于
LD	MC_MABS		• KS • KW103 • KW203

参数	输入/输出	数据类型	允许的内存区	描　述
EXEC	输入	BOOL	M、V、L、SM	上升沿触发本指令执行一次；下降沿触发暂停运动
AXIS	输入	INT	V、M、L、常量	目标轴的轴号（也就是 CANOpen 从站的地址）
POS	输入	REAL	V、M、L、常量	绝对目标位置，单位：mm 或者°
VEL	输入	REAL	V、M、L、常量	运动中增加的最大速度（>0），单位：mm/min 或°/min。
DIR	输入	INT	V、M、L、常量	运动方向。预留用，暂未实现功能，保持为 0 即可。
MODE	输入	INT	V、M、L、常量	运动模式：单次执行或者永久执行。 0 表示单次执行，轴执行完本次绝对定位后指令就退出。 1 表示永久执行，当轴执行完一次绝对定位后，指令并不退出，若发现新的目标位置就会发送命令让轴继续执行新一次绝对定位。
DONE	输出	BOOL	M、V、L	完成标志位。指令执行完成时，*DONE* 由 0 跳变到 1。
ERR	输出	BOOL	M、V、L	错误标志位。若指令执行时发生错误，则被置 1。
ERRID	输出	BYTE	V、M、L	错误码

ACT	输出	BOOL	M、V、L	MODE=0,单次执行时，ACT 表示单次定位指令是否被正确激活。1 表示激活，0 表示未激活。 MODE=1,永久执行时，ACT 表示永久定位指令是否被正确激活。1 表示激活（单次定位完成时也会一直处于 1），0 表示未激活。

本指令控制目标轴运动到目标位置（绝对位置）。运动时，速度从当前值开始，到达目标位置时速度为零。本指令允许暂停。

- LD 格式指令说明

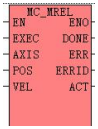
若*EN*为1，那么在*EXEC*的上升沿会触发执行本指令。

在指令执行时，PLC控制轴按照用户输入的目标位置（*POS*）、运动速度（*VEL*）参数值，让轴开始绝对定位。在运动过程中，指令将不停扫描目标位置和和目标速度参数值，若有变化则立即发送给轴，也就是随时可以接受新的速度参数和位置参数值（例如要执行暂停，在运动过程中将速度置为0即可暂停，重新给速度值则恢复运动）。同时，PLC将不停检查检查轴的返回状态，如果成功到达本次定位的目标位置，表示本次定位完成，则DONE被置1，ERR被置0，ERRID被置0。本次定位完成后，指令将判断模式（*MODE*）值，若设置为单次运行模式，那么指令直接退出；若设置为永久运行模式，那么指令并不退出，随时扫描目标位置值，若目标位置发生变化就会将其发送给轴，让轴进行新一次绝对定位。

若发生错误（可能是指令本身执行错误，也可能是执行过程中驱动器未正确执行动作的错误，详见错误码）则指令执行失败，指令都会停止执行，同时将DONE被1，ERR置1，ERRID赋值为相应的错误代码。

若 *EN* 为 0，则指令不执行。若在执行过程中 *EN* 变为 0，则指令将停止执行，轴处于静止锁轴等候状态。

4.5.7.1.7 MC_MREL（相对运动）

	名称	指令格式	适用于
LD	MC_MREL		• KS • KW103 • KW203

参数	输入/输出	数据类型	允许的内存区	描　述
EXEC	输入	BOOL	M、V、L、SM	上升沿触发本指令执行一次；下降沿触发暂停运动
AXIS	输入	INT	V、M、L、常量	目标轴的轴号（也就是 CANOpen 从站的地址）

POS	输入	REAL	V、M、L、常量	要运动的相对距离，单位：mm 或者°。 正数表示正方向运动；负数表示负方向运动。
VEL	输入	REAL	V、M、L、常量	运动中增加的最大速度（>0），单位：mm/min 或°/min。
DONE	输出	BOOL	M、V、L	完成标志位。指令执行完成时，*DONE* 由 0 跳变到 1。
ERR	输出	BOOL	M、V、L	错误标志位。若指令执行时发生错误，则被置 1。
ERRID	输出	BYTE	V、M、L	错误码
ACT	输出	BOOL	M、V、L	该指令是否被正确激活。1 表示激活，0 表示未激活。

本指令控制目标轴来运动指定的距离 *POS*（以当前位置作为参考，也就是将当前位置做为起始位置）。运动时，速度从当前值开始，到达目标位置时速度为零。本指令允许暂停。

- LD 格式指令说明

若*EN*为1，那么在*EXEC*的上升沿会触发执行本指令。

在指令执行时，PLC控制轴按照用户输入的目标位置（*POS*）、运动速度（*VEL*）参数值，让轴开始相对定位（以当前位置作为参考）。在运动过程中，指令将不停扫描目标速度参数值，若有变化则立即发送给轴，也就是随时可以接受新的速度参数值（例如要执行暂停，在运动过程中将速度置为0即可，然后重新给速度值则恢复运动）。同时，PLC将不停检查检查轴的返回状态，如果成功到达本次定位的目标位置，表示本次定位完成，则*DONE*被置1，*ERR*被置0，*ERRID*被置0。若发生错误（可能是指令本身执行错误，也可能是执行过程中驱动器未正确执行动作的错误，详见错误码）则指令执行失败，指令都会停止执行，同时将*DONE*被1，*ERR*置1，*ERRID*赋值为相应的错误代码。

若 *EN* 为 0，则指令不执行。若在执行过程中 *EN* 变为 0，则指令将停止执行，轴处于静止锁轴等候状态。

4.5.7.1.8 MC_JOG（点动）

	名称	指令格式	适用于

LD	MC_JOG		• KS • KW103 • KW203

参数	输入/输出	数据类型	允许的内存区	描　述
EXEC	输入	BOOL	M、V、L、SM	上升沿触发本指令执行一次；下降沿触发暂停运动
AXIS	输入	INT	V、M、L、常量	目标轴的轴号（也就是 CANOpen 从站的地址）
VEL	输入	REAL	V、M、L、常量	运动速度，单位：mm/min 或°/min。 正数表示正方向，负数表示负方向。
DIR	输入	INT	V、M、L、常量	运动方向。预留用，暂未实现功能，保持为 0 即可。
DONE	输出	BOOL	M、V、L	完成标志位。指令执行完成时，*DONE* 由 0 跳变到 1。
ERR	输出	BOOL	M、V、L	错误标志位。若指令执行时发生错误，则被置 1。
ERRID	输出	BYTE	V、M、L	错误码
ACT	输出	BOOL	M、V、L	该指令是否被正确激活。1 表示激活，0 表示未激活。

本指令控制目标轴以 Vel 指定的目标速度运行。

- LD 格式指令说明

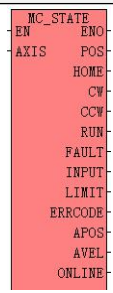
若*EN*为1，那么在*EXEC*的上升沿会触发执行本指令。

在指令执行时，PLC控制轴按照用户输入运动速度（*VEL*）参数值，让轴开始点动运行。轴运动过程中，指令将不停扫描目标速度参数值，若有变化则立即发送给轴，也就是随时可以接受新的速度参数值。

若发生错误（可能是指令本身执行错误，也可能是执行过程中驱动器未正确执行动作的错误，详见错误码）则指令执行失败，指令都会停止执行，同时将DONE被1，ERR置1，ERRID赋值为相应的错误代码。

若 *EN* 为 0，则指令不执行。若在执行过程中 *EXEC* 变为 0，则指令将停止执行，轴处于静止锁轴等候状态。

4.5.7.1.9 MC_STATE（读取驱动器的各状态数值）

	名称	指令格式	适用于
LD	MC_STATE		• KS • KW103 • KW203

参数	输入/输出	数据类型	允许的内存区	描　述
AXIS	输入	INT	V、M、L、常量	目标轴的轴号（也就是 CANOpen 从站的地址）
POS	输出	BOOL	V、M、L	“位置到”信号
HOME	输出	BOOL	V、M、L	“原点找到”信号
CW	输出	BOOL	M、V、L	“电机正转”信号
CCW	输出	BOOL	M、V、L	“电机反转”信号
RUN	输出	BOOL	M、V、L	“电机运行中”标志
FAULT	输出	BOOL	M、V、L	“轴报警”标志
INPUT	输出	WORD	V、M、L	轴的开关量输入口状态，BIT0 对应轴的 DIN1，依次顺序存放，具体数字量输入口数量查询驱动器手册
LIMIT	输出	BOOL	M、V、L	“限位到”标志
ERRCODE	输出	WORD	V、M、L	轴的报警错误码
APOS	输出	REAL	M、V、L	机械的当前实际位置，mm 或者°。
AVEL	输出	REAL	M、V、L	机械的当前实际速度，mm/min 或°/min。
ONLINE	输出	BYTE	M、V、L	“轴在线”标志。1 表示轴不在线，0 表示轴在线。

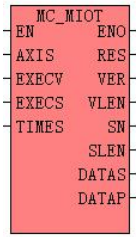
本指令一直扫描驱动器状态，获取各种不同状态的标志并输出到相应的输出参数中。

注意：“位置到”和“原点找到”这两个信号在执行动作过程中（定位或找原点）会重新变为 0，直到动作正确执行完后才会重新置 1！

- LD 格式指令说明

若*EN*为1，则本指令执行。若*EN*为0，则指令不执行，也不会刷新各种输出参数。

4.5.7.1.10 MC_MIOT（读取设备信息）

	名称	指令格式	适用于
LD	MC_HOME		• KS

参数	输入/输出	数据类型	允许的内存区	描　述
AXIS	输入	INT	V、M、L、常量	目标轴的轴号（也就是 CANOpen 从站的地址）
EXECV	输入	BOOL	M、V、L、SM	上升沿触发读取一次序列号、软件版本
EXECS	输入	BOOL	M、V、L、SM	上升沿触发读取一次 IIT、温度、运行时间
TIMES	输入	INT	V、M、L	定时器，定时时间到则读取一次 IIT、温度、运行时间。若为 0，则定时器不启动。
RES	输出	BYTE	M、V、L	执行结果。
VER	输出	BYTE	M、V、L	软件版本信息的存放起始地址
VLEN	输出	BYTE	M、V、L	软件版本信息的总长度，单位：字节
SN	输出	BYTE	M、V、L	序列号信息的存放起始地址
SLEN	输出	BYTE	M、V、L	序列号信息的总长度，单位：字节
DATAS	输出	BYTE	M、V、L	IIT、温度、运行时间信息的存放起始地址
DATAP	输出	BYTE	M、V、L	状态字、电流、速度等信息的存放起始地址

本指令用于读取目标轴的产品、运行状态等信息。每个轴只允许使用 1 个本指令。

对于同一个轴，MC_MIOT 指令优先级最低：若其它运动指令正在运行，则 MC_MIOT 不会执行；若其它运动指令被启动执行，则 MC_MIOT 指令会被打断，直接终止。

本指令读取的信息分为 3 类，下面将详细说明。

- **序列号、软件版本信息**

由 *EXECV* 参数的上升沿来触发读取一次这些信息。这些是固定信息，一般在上电时读取一次即可。

VER 参数指定了软件版本信息的存放起始地址，软件版本信息连续存放在该地址开始的区域。*VLEN* 参数值指明了软件版本信息的总长度，即占用的字节个数。

SN 参数指定了产品序列号信息的存放起始地址，序列号信息连续存放在该地址开始的区域。*SLEN* 参数值指明了序列号信息的总长度，即占用的字节个数。

每次触发后，PLC 执行一次读取过程，若全部读取成功，则更新输出参数中的数据、长度信息；若读取失败，则不更新输出参数。 无论成功还是失败，当指令完成后，都会刷新 *RES* 中的相应位。

RES 中的位	**描 述**
Bit 7	指明是否读取完成本组参数。 0 表示正在读取过程中；1 表示读取完成（无论成功或者失败）。
Bit 6	指明读取本组参数是否发生错误。 0 表示成功读取；1 表示读取过程中出现错误。

• IIT、驱动器温度、运行时间信息

这些是运行过程中的信息，需要实时读取，但是读取不能太频繁，否则可能会影响其它运动。

这些参数的读取有 2 种触发条件：*EXECS* 参数的上升沿来触发读取一次；*TIMES* 指定的定时读取周期，PLC 每隔这个时间就会触发读取一次。 若 *TIMES* 参数值为 0，则会停止定时读取。

DATAS 参数指定了这些信息的存放起始地址，各个参数信息按下表进行存放：

参数名称	**对象**	**数据类型**	**长度**	**在存放区域中的字节偏移量**
IIT	0x2FF010	UINT8	1	0
驱动器温度	0x60F70B	UINT16	2	1
运行时间	0x2FF700	UINT32	4	3

每次触发后，PLC 执行一次读取过程，若全部读取成功，则更新输出参数中的数据信息；若读取失败，则不更新输出参数。 无论成功还是失败，当指令完成后，都会刷新 *RES* 中的相应位。

RES 中的位	**描 述**
Bit 5	指明是否完成读取本组参数。 0 表示正在读取过程中；1 表示读取完成（无论成功或者失败）。
Bit 4	指明读取本组参数是否发生错误。 0 表示成功读取；1 表示读取过程中出现错误。

• 状态字、错误字、实际电流等信息

这些信息由指令通过 PDO 自动进行读取，无需用户在程序中触发。

DATAP 参数指定了这些信息存放的起始地址，各个参数信息按下表进行存放：

参数名称	对象	数据类型	长度	在存放区域中的字节偏移量
状态字	0x604100	UINT16	2	0
错误字	0x260100	UINT16	2	2
错误字 1	0x260200	UINT16	2	4
实际电流	0x607800	INT16	2	6
实际速度	0x606C00	INT32	4	8
实际位置	0x606300	INT32	4	12

- **执行结果参数：*RES***

RES 中的位	描 述
Bit 7	指明是否读取完成软件版本、序列号参数。 0 表示正在读取过程中；1 表示读取完成（无论成功或者失败）。
Bit 6	指明读取软件版本、序列号参数是否发生错误。 0 表示成功读取；1 表示读取过程中出现错误。
Bit 5	指明是否完成读取 IIT、温度、实际参数。 0 表示正在读取过程中；1 表示读取完成（无论成功或者失败）。
Bit 4	指明读取本组 IIT、温度、实际是否发生错误。 0 表示成功读取；1 表示读取过程中出现错误。
Bit 3…0	组合值表示了执行错误： 0 ---　表示无错误 1 ---　表示目标轴号错误 2 ---　表示其它运动指令正在执行，本指令不能被运行。

- **LD 格式指令说明**

若*EN*为1，那么根据EXECV、EXECS、TIMES参数条件分别触发读取相应的设备信息。

若 *EN* 为 0，则指令不执行。若在执行过程中 *EN* 变为 0，则指令将停止执行。

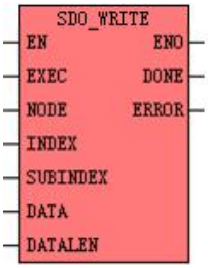
4.5.7.2　SDO 指令

SDO 指令位于指令集的【CAN 指令】组中。

当使用 Kinco 运动控制功能或者 CANOpen 主站功能时，可以使用 SDO 指令。

在一个用户工程中，最多允许使用 64 个 SDO 指令。

4.5.7.2.1 SDO_WRITE

	名称	指令格式	适用产品
LD	SDO_WRITE		• KS • KW103 • KW203

参数	输入/输出	数据类型	允许使用的内存区
EXEC	输入	BOOL	I、Q、V、M、L、SM
NODE	输入	BYTE	I、Q、V、M、L、SM、常量
INDEX	输入	WORD	I、Q、V、M、L、SM、常量
SUBINDEX	输入	BYTE	I、Q、V、M、L、SM、常量
DATA	输入	BYTE	I、Q、V、M、L、SM
DATALEN	输入	BYTE	I、Q、V、M、L、SM、常量
DONE	输出	BOOL	Q、M、V、L、SM
ERROR	输出	DWORD	Q、M、V、L、SM

注意：NODE, INDEX, SUBINDEX, DATALEN 必须同时为常量或同时为变量；DATA 与 DATALEN 参数组成了一个长度可变的内存块，此内存块必须全部位于合法的内存区域，否则结果不可预期。

各个参数的具体使用说明，见下表：

参数	功能
EN	使能端。若 *EN* 为 1 时，则该指令被使能，允许执行。
EXEC	启动端。*EXEC* 的上升沿触发本指令执行一次，需保证让 *EN* 先于 *EXEC* 导通。
NODEID	待访问节点的地址
Index	待访问对象在 OD 中的索引

SubIndex	待访问对象在 OD 中的子索引
Data	待发送数据存放的起始字节
DataLen	接发送数据的长度，单位：字节
DONE	执行结果指示。 若 SDO 正在执行，DONE 为 0；若 SDO 通信结束（收到回应或者超时），DONE 为 1。
ERROR	错误信息。见下表

SDO 错误信息说明，见下表：

错误码	说明
0	无错误。
1	主站没有启用
2	目标节点不存在
3	输入参数值错误（比如数据长度）
4	目标节点上一次的命令尚未得到回应
5	PLC 的发送或者接收缓冲区已满
6	指令超时没有回应
7	收到的回应报文错误（不是期望的回应报文、长度错误等等）
8	收到终止报文
9	本工程中，SDO 指令数量超出限制

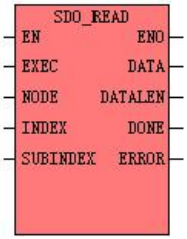
- LD 格式指令说明

如果 EN 为 1，则该指令被扫描，若检测到 EXEC 的上升沿，则启动执行一次。

如果 EN 为 0，则指令不被扫描，也不会被执行。

4.5.7.2.2 SDO_READ

	名称	指令格式	适用产品

LD	SDO_READ		• KS • KW103 • KW203

参数	输入/输出	数据类型	允许使用的内存区
EXEC	输入	BOOL	I、Q、V、M、L、SM
NODE	输入	BYTE	I、Q、V、M、L、SM、常量
INDEX	输入	WORD	I、Q、V、M、L、SM、常量
SUBINDEX	输入	BYTE	I、Q、V、M、L、SM、常量
DATA	输出	BYTE	I、Q、V、M、L、SM
DATALEN	输出	BYTE	I、Q、V、M、L、SM
DONE	输出	BOOL	Q、M、V、L、SM
ERROR	输出	DWORD	Q、M、V、L、SM

注意：NODE, INDEX, SUBINDEX, DATALEN 必须同时为常量或同时为变量；DATA 与 DATALEN 参数组成了一个长度可变的内存块，此内存块必须全部位于合法的内存区域，否则结果不可预期。

各个参数的具体使用说明，见下表：

参数	功能
EN	使能端。若 *EN* 为 1 时，则该指令被使能，允许执行。
EXEC	启动端。*EXEC* 的上升沿触发本指令执行一次，需保证让 *EN* 先于 *EXEC* 导通。
NODEID	待访问节点的地址
Index	待访问对象在 OD 中的索引
SubIndex	待访问对象在 OD 中的子索引
Data	接收到数据存放的起始字节

DataLen	接收到数据的长度，单位：字节
DONE	执行结果指示。 若 SDO 正在执行，DONE 为 0；若 SDO 通信结束（收到回应或者超时），DONE 为 1。
ERROR	错误信息。见下表

SDO 错误信息说明，见下表：

错误码	说明
0	无错误。
1	主站没有启用
2	目标节点不存在
3	输入参数值错误（比如数据长度）
4	目标节点上一次的命令尚未得到回应
5	PLC 的发送或者接收缓冲区已满
6	指令超时没有回应
7	收到的回应报文错误（不是期望的回应报文、长度错误等等）
8	收到终止报文
9	本工程中，SDO 指令数量超出限制

- LD 格式指令说明

如果 EN 为 1，则该指令被扫描，若检测到 EXEC 的上升沿，则启动执行一次。

如果 EN 为 0，则指令不被扫描，也不会被执行。

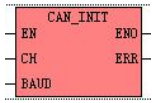
4.5.7.3 CAN 自由通信指令

CAN 通信指令位于指令集的【CAN 指令】组中。

注意：若与其它协议混用，则自由通信报文的 ID 号不允许使用 CANOpen 协议中的 COB-ID 号！

4.5.7.3.1 CAN_INIT（初始化 CAN 接口）

	名称	指令格式	适用产品

LD	CAN_INIT		• KS • KW103 • KW203

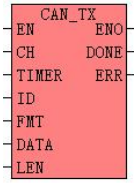
参数	输入/输出	数据类型	允许的内存区
EN	输入	BOOL	I、Q、V、M、L、SM
CH	输入	INT	常量
BAUD	输入	INT	L、M、V、常量
ERR	输出	BOOL	L、M、V、常量

参数	描　述
EN	使能端。
CH	所用的 CAN 接口。0 表示 CAN1，1 表示 CAN2，2 表示 K541 模块
BAUD	CAN 的波特率。 8 --- 1000K 7 --- 800K 6 --- 500K 5 --- 250K 4 --- 125K 3 --- 50K 2 --- 20K 1--- 10K
ERR	指令执行是否成功。0 表示成功，1 表示有错误（比如参数错误）

EN 输入端的上升沿跳变会触发执行该指令，用于初始化指定的 CAN 接口（*CH*），并将 CAN 波特率设置为 *BAUD* 值。

4.5.7.3.2 CAN_TX（自动发送 CAN 报文）

	名称	指令格式	适用产品

LD	CAN_TX		• KS • KW203

参数	输入/输出	数据类型	允许的内存区
CH	输入	INT	常量
TIMER	输入	INT	L、M、V、常量
ID	输入	DWORD	L、M、V、常量
FMT	输入	INT	L、M、V、常量
DATA	输出	BYTE	M、V
LEN	输出	BYTE	L、M、V
DONE	输出	BOOL	L、M、V
ERR	输出	BOOL	L、M、V

参数	描　述
EN	使能端。
CH	所用的 CAN 接口。0 表示 CAN1，1 表示 CAN2，2 表示 K541 模块
TIMER	待发送报文定时发送的周期，单位 ms。0 表示不启用定时发送功能。
ID	待发送报文的 ID
FMT	待发送报文的格式。0 表示标准帧，1 表示扩展帧。
DATA	待发送报文数据存放的首地址。
LEN	待发送报文的数据长度，单位：字节。
DONE	发送完成标志位。每次发送成功后，DONE 自动置 1 并维持至少一个扫描周期的时间。
ERR	发送错误标志位。1 表示本次发送失败。

注意：ID, FMT, TIMER 和 LEN 参数必须同时为常量或同时为变量；DATA 与 LEN 参数组成了一个长度可变的内存块，此内存块必须全部位于合法的内存区域，否则结果不可预期。

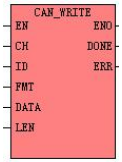
PLC 内部维护着一个报文自动发送列表，当表中的报文满足发送条件后，PLC 会将此报文自动发送出去。某报文自动发送的条件为：若报文中的数据发生了变化，则立即发送一次；若设置的定时发送周期时间到了，则立即发送一次。报文发送完成后，输出参数 *DONE* 自动置 1 并维持一个扫描后

自动变为 0，若发送失败（原因是发送缓冲区已满或者报文发送失败），则输出参数 *ERR* 自动置 1。该发送报文列表最大长度为 48 条。

CAN_TX 指令用于向该发送报文列表中添加一条报文，报文由报文 *ID* 号、格式（*FMT*，指明扩展帧或者标准帧）、数据 (*DATA*，指明报文数据存放的起始地址)、长度 *LEN* 来指定。*TIMER* 参数值指明了定时发送的周期（ms），若 *TIMER* 值为 0，则表示不会定时发送。

EN 输入端的上升沿跳变会触发执行该指令。本指令执行后，则 PLC 立即将该指令参数指定的报文加入到自动发送列表中。**因此在一个工程中，一条 CAN_TX 指令仅需要执行一次即可。另外，一个工程中调用的 CAN_TX 指令最多允许 48 条！CAN_TX 指令和 CAN_WRITE 指令的总条数最多允许 64 条！**

4.5.7.3.3 CAN_WRITE（发送一次 CAN 报文）

	名称	指令格式	适用产品
LD	CAN_WRITE		• KS • KW103 • KW203

参数	输入/输出	数据类型	允许的内存区
CH	输入	INT	常量
ID	输入	DWORD	L、M、V、常量
FMT	输入	BYTE	L、M、V、常量
DATA	输入	BYTE	L、M、V
LEN	输入	BYTE	L、M、V、常量
DONE	输出	BOOL	L、M、V
ERR	输出	BOOL	L、M、V

参数	描　述
EN	使能端。
CH	所用的 CAN 接口。0 表示 CAN1，1 表示 CAN2，2 表示 K541 模块
ID	待发送报文的 ID 号。

FMT	报文格式。0 表示标准帧，1 表示扩展帧。
DATA	待发送数据存放的起始字节地址。
LEN	待发送数据的长度。单位：字节。
DONE	报文是否发送完成。在执行时 DONE 被置 0，发送完成则 DONE 被置 1。
ERR	报文发送是否出错。若发送失败，则 ERR 被置 1。

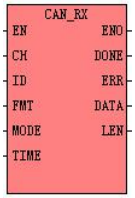
注意：ID, FMT 和 LEN 参数必须同时为常量或同时为变量；DATA 与 LEN 参数组成了一个长度可变的内存块，此内存块必须全部位于合法的内存区域，否则结果不可预期。

待发送的 CAN 报文由 *ID* 号、格式（*FMT*，指明扩展帧或者标准帧）、数据 (*DATA*，指明报文数据存放的起始地址)、长度 *LEN* 来指定。

EN 输入端的上升沿跳变会触发执行一次该指令，将待发送的报文写入 PLC 内部的发送缓冲区中，再由 PLC 调度通过通过指定的 CAN 接口 *CH* 发送出去。

若指令成功将报文发送出去，则将 *DONE* 置为 1，*ERR* 置为 0。若发送缓冲区已满或者报文发送失败，则同时将 *DONE* 和 *ERR* 置为 1。

4.5.7.3.4 CAN_RX（接收特定 ID 号 CAN 报文）

	名称	指令格式	适用产品
LD	CAN_RX		• KS • KW103 • KW203

参数	输入/输出	数据类型	允许的内存区
CH	输入	INT	常量
ID	输入	DWORD	L、M、V、常量
FMT	输入	INT	L、M、V、常量
MODE	输入	INT	L、M、V、常量
TIME	输入	INT	L、M、V、常量
DONE	输出	BOOL	L、M、V
ERR	输出	BOOL	L、M、V
DATA	输出	BYTE	M、V

LEN	输出	BYTE	L、M、V

参数	描　述
EN	使能端。
CH	所用的 CAN 接口。0 表示 CAN1，1 表示 CAN2，2 表示 K541 模块
ID	待接收报文的 ID
FMT	待接收报文的格式。0 表示标准帧，1 表示扩展帧。
MODE	接收模式。0 表示永久接收模式，1 表示单次接收模式
TIME	接收超时时间。单位 ms
DONE	在单次接收模式下，DONE 是接收成功标志位。
ERR	接收超时标志位。
DATA	最近一次接收的报文数据存放的首地址。
LEN	最近一次接收的报文的数据长度，单位：字节。

注意：ID, FMT, MODE 和 TIME 参数必须同时为常量或同时为变量；DATA 与 LEN 参数组成了一个长度可变的内存块，此内存块必须全部位于合法的内存区域，否则结果不可预期。

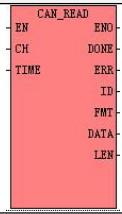
该指令用于接收指定 ID 号（*ID*）和格式（*FMT*）的报文。

MODE 参数指明了接收方式，若 *MODE* 为 1，则为单次接收模式，指令只接收一次指定报文，收到后则退出；若 *MODE* 为 0，则为永久接收模式，指令会一直接收指定报文。

EN 输入端的上升沿跳变会触发执行该指令。本指令执行后，PLC 立即进入接收状态，同时将 *DONE*、*ERR* 清 0。若为单次接收模式，那么如果在时间 *TIME* 内接收到指定报文，则将 *DONE* 置 1，指令退出接收状态，如果在时间 *TIME* 内没有收到指定报文，那么 *DONE* 和 *ERR* 被置 1，指令退出接收状态；若为永久接收模式，那么本指令启动后就会一直监视 CAN 接口 *CH* 并接收所有的指定报文，若在一次成功接收之后，在 *TIME* 内没有再次接收到指定报文，则将 ERR 置 1，之后若再次成功接收，那么就会将 ERR 清 0。

注意：在永久接收模式下（*MODE* 为 0），CPU 内部会建立一个报文 ID 过滤列表，每个 CAN_RX 指令执行后，其 *ID* 参数值都会存入到过滤列表中。CPU 会对接收的的 CAN 报文进行过滤，只有报文 ID 符合列表值的报文才会被本指令接收下来。因此，每个 CAN_RX 指令仅需要上电时执行一次即可，无需反复执行。

4.5.7.3.5 CAN_READ（接收一次 CAN 报文）

	名称	指令格式	适用产品
LD	CAN_READ		• KS • KW103 • KW203

参数	输入/输出	数据类型	允许的内存区
CH	输入	INT	常量（1 和 2）
TIME	输入	INT	L、M、V、常量
DONE	输出	BOOL	L、M、V
ERR	输出	BOOL	L、M、V
ID	输出	DWORD	L、M、V
FMT	输出	BYTE	L、M、V
DATA	输出	BYTE	M、V
LEN	输出	BYTE	L、M、V

参数	描　述
EN	使能端。
CH	所用的 CAN 接口。0 表示 CAN1，1 表示 CAN2，2 表示 K541 模块
TIME	超时时间。启动接收后，若在指定的这个时间内没有收到任何报文，则超时退出接收状态，并将 ERR 置 1
ID	接收到的报文的 ID 号。
FMT	接收到的报文格式。0 表示标准帧，1 表示扩展帧。
DATA	接收到的报文数据存放的起始字节地址。
LEN	接收到的报文数据长度。单位：字节。
DONE	是否接收完成。在执行时 DONE 被置 0，发送完成则 DONE 被置 1。
ERR	接收报文是否出错。若接收失败则 ERR 被置 1。

注意： DATA 与 LEN 参数组成了一个长度可变的内存块，此内存块必须全部位于合法的内存区域，否则结果不可预期。

EN 输入端的上升沿跳变会触发执行该指令：CAN 接口 *CH* 进入接收状态，接收任何一条来自总线上的报文。

启动接收后，若在指定的超时时间 *TIME* 内接收到一条 CAN 报文，那么 PLC 就将报文相关数据分别置于输出参数 *ID*（ID 号）、*FMT*（格式，指明接收到的是扩展帧或者标准帧）、*DATA* (接收到的报文数据存放的起始地址)、LEN（长度）中，同时将 DONE 置为 1，退出接收。若在指定的超时时间 *TIM*）内没有收到任何报文，则超时退出接收状态，并将 *DONE* 和 ERR 置 1。

CAN_READ 指令启动后，将会接收指定 CAN 接口的任何一条报文，因此，与其它协议（比如 CANOpen）混用时需要注意。

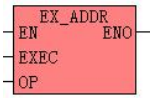
注意：1） CAN_READ 指令的优先级低于 CAN_RX 指令。

2）当 CAN_READ 指令启动之后，总线上的任何一条报文都会被其接收，因此若程序中调用了多条 CAN_READ 指令，那么最后启动的指令将会生效。

4.5.7.4 扩展总线指令

扩展总线指令位于指令集的【CAN 指令】组中。

4.5.7.4.1 EX_ADDR（修改扩展模块配置）

	名称	指令格式	适用产品
LD	EX_ADDR		• KS • KW103 • KW203

参数	输入/输出	数据类型	允许的内存区
EXEC	输入	BOOL	M、V、L、SM
OP	输入	INT	M、V、L、常量

参数	描　述
EXEC	启动端。*EXEC* 的上升沿触发本指令执行一次，需保证让 *EN* 先于 *EXEC* 导通。
OP	操作码。 181 --- 命令所有扩展模块保存好自身的 ID 和各种参数。

	99 --- 命令所有扩展模块清除已保存的 ID 和各种参数。

本指令会根据 *OP* 值向所连的扩展模块发送相应的命令：

- 若 *OP* 值为 181，则扩展模块接收到命令后会自动保存好自己的 ID 和各种参数（比如信号形式、滤波方式等）。以后再上电时扩展模块会自动读取保存好的数据并进入运行状态，不需要 CPU 进行配置，因此可以独立于 CPU 在任意时间上电或断电。
- 若 *OP* 值为 99，则扩展模块收到命令后会清除保存的 ID 和通道参数，以后再上电时就会等待 CPU 自动分配 ID 并配置参数。

➢ **LD 格式指令说明**

如果 EN 为 1，则该指令被扫描，若检测到 EXEC 的上升沿，则启动执行一次。

如果 EN 为 0，则指令不被扫描，也不会被执行。