SMCS1120 二轴运动控制器

用户手册

Version 1.2

目 录

版本记录：

Ver 1.0 首次发布了 SMCS1120；在该版本中阐述了其基本功能，性能指标、对其特性、使用方法进行了完整的介绍；

Ver 1.1 补充说明了光源应用事项，并给出参考电路，对光源操作方法进行了优化，使操作更简便；

Ver 1.2 对手册进行了整理，将控制器操作思路编入软件开发章节中；

免责声明：

尽管我们已付出最大努力，但由于多方面原因，产品仍然不能保证 100％完全正确无误运行，因此用户有责任设计安全有效的保护机制防止差错的出现带来的人身和财产伤害，南京雪曼公司没有责任和义务对此负责，当损失发生时，南京雪曼只提供最高相当于该产品本身 100％价格的赔偿。

名词约定：

1．本手册中所提及＂运动控制器＂如无特殊说明均指 SMCS1120。

2．本手册中所提及＂运动控制卡＂（Motion Card）如无特殊说明，均指 PCI、ISA、PC104 等接口，以插入计算机内机箱部作为与计算机连接形式的运动控制器。

3．本手册中所提及＂运动控制器＂（Motion Controller）如无特殊说明，均指 RS232、USB、Ethernet 等接口，以通讯方式与计算机连接的运动控制器；这类运动控制器也能不依赖计算机而独立工作，往往具备程序可下载，可脱机工作，可使用文本显示器作为程序、参数编辑和状态监控界面。

警告：运动机械有危险，用户有责任设计有效的异常保护装置，确保异常出现时机器能有效控制；同时在应用软件假如中加入检测和诊断模块，以保证软件在可控的范围内运行。

第一章：概述

SMCS1120 是基于先进 DSP 技术的高级专用型两轴运动控制器，是雪曼科技 SMCS 系列产品中的 AOI 及其类似设备专用运动控制器。SMCS1120 具有 2 路脉冲量运动控制、2 路编码器位置检测、4 路光源控制等，为独立式运动控制器；

SMCS1120 适用于离线式 PCB 自动光学检测仪（AOI）以及其它需要 2 轴运动控制和少量 I / O 控制的应用，由资深 AOI 专家和运动控制专家精心研制，是 AOI 行业内首款一流的专用控制器。

第二章：功能特点与技术指标

1 2 轴脉冲＋方向形式控制信号输出，标准的步进 / 伺服驱动器接口，可与大部分品牌驱动器连接，手册内附三菱交流伺服驱动器接线图。

2 2 轴增量式编码器位置检测输入，可接入电机编码器或者光栅尺，镜头位置检测更精确，有助于减小拼图缝隙。

3 专利的 4 路 LED 恒流驱动技术，亮度可分别设定，每路输出电流 0－600mA 可调，可驱动发光盘用于 CCD 照明。节省了传统的发光盘电源板，同时使系统更简洁，生产维护更方便；

4 SMCS1120 的频闪功能可减少发光盘 LED 数量至传统发光盘的一半以下，同时由于 LED 大部分时间处于熄灭状态，可延长发光盘使用寿命一倍以上，使用 SMCS1120 可立即降低系统成本。

5 可分别程序控制的 R、G、B、W 四色光源亮度，很好的解决了由于各种颜色 LED 发光效率不同带来的色差问题，同时还可在检测焊盘、丝印等不同对象时提供不同颜色的照明，使图像对比度提高从而减少误判漏判。

6 光源两种发光模式：连续发光模式用于手动操作和编程，频闪模式用于自动检测；控制器在运动停止时自动提前开启光源，无需软件干预。

7 专利的摇杆控制功能可使 AOI 仅配备一只电阻型摇杆、一台 CCD、显示器和驱动系统便可手动检测，无需 PC 机，接通电源即开始检测，方便快捷，配合步进驱动系统更可大幅降低整机成本。

8 每轴均具备软件限位功能，防止做程序时出错（指定了超行程的运动位置）导致撞机。

9 6 路光电隔离输入，可用于按钮、安全门等开关量检测。

10 4 路光电隔离输出，可用于 PCB OK / NG 判定声光提示。

11 115200Bit / S 通讯速率的 RS232 通讯，运动控制器内置高速 DSP 数字信号处理器，独立于 PC 工作，对 PC 依赖性降到最低。全 DB 座连接器易于拔插，连接可靠，很好的耐震动和耐电磁干扰性能。

12 实时嵌入式操作系统，具有自诊断，自监控功能，具有硬件级的出错处理能力，运动控制系统可靠性不再受 PC 硬件和 WINDOWS 操作系统限制。

13 小巧的立式安装设计，便于设计便携式电控箱，生产维护更便捷。

14 技术指标：

14.1 RS232 通讯波特率：115200bit / s

14.2 运动控制轴数：2 轴

14.3 脉冲频率输出：5Hz－500KHz

14.4 编码器计数轴数：2 轴

14.5 最大脉冲频率输入：10MHz

14.6 加减速时间设置范围：5ms-5000ms

14.7 位置寄存器宽度：32 位有符号数，单次运动范围：－2,147,483,648～＋2,147,483,647

14.8 I / O 功能（以下均为光电隔离部分电源电压为 24V 时的参数，为 12V 时见特别说明）

- 光电隔离最大耐压：2500V，符合 CE 测试中 EFT 要求
- 光电隔离输出口最大工作电压：36V，绝对最大电压 50V（3 秒）
- 光电隔离输出口最大电流：50mA，绝对最大电流 100 mA（3 秒）
- 光电隔离输入 / 输出最大延时 10 微秒
- 光电隔离输入高电平：21～24V 均为可靠的高电平，电源为 12V 时， 8－10V 为可靠高电平
- 光电隔离输入低电平：0－7V 均为可靠的低电平（7－21V 之间为不确定电平），电源为 12V 时，0－4V 均为可靠的低电平。
- 低电平输入开关触点流过最大电流：5.1mA；电源为 12V 时，最大电流：2.4mA；

14.9 工作条件：

- 工作温度：0～50℃
- 贮存温度：－40～80℃
- 湿度：15%～85%，非结露
- 工作电压：＋12～＋24VDC，照明灯关闭状态下功耗：电流＝100mA

14.10 机械尺寸：长×宽×高＝190 × 43 × 85 mm

第三章：硬件使用说明

3．1 建立整体了解

3．1．1 SMCS1120 典型应用连接示意图

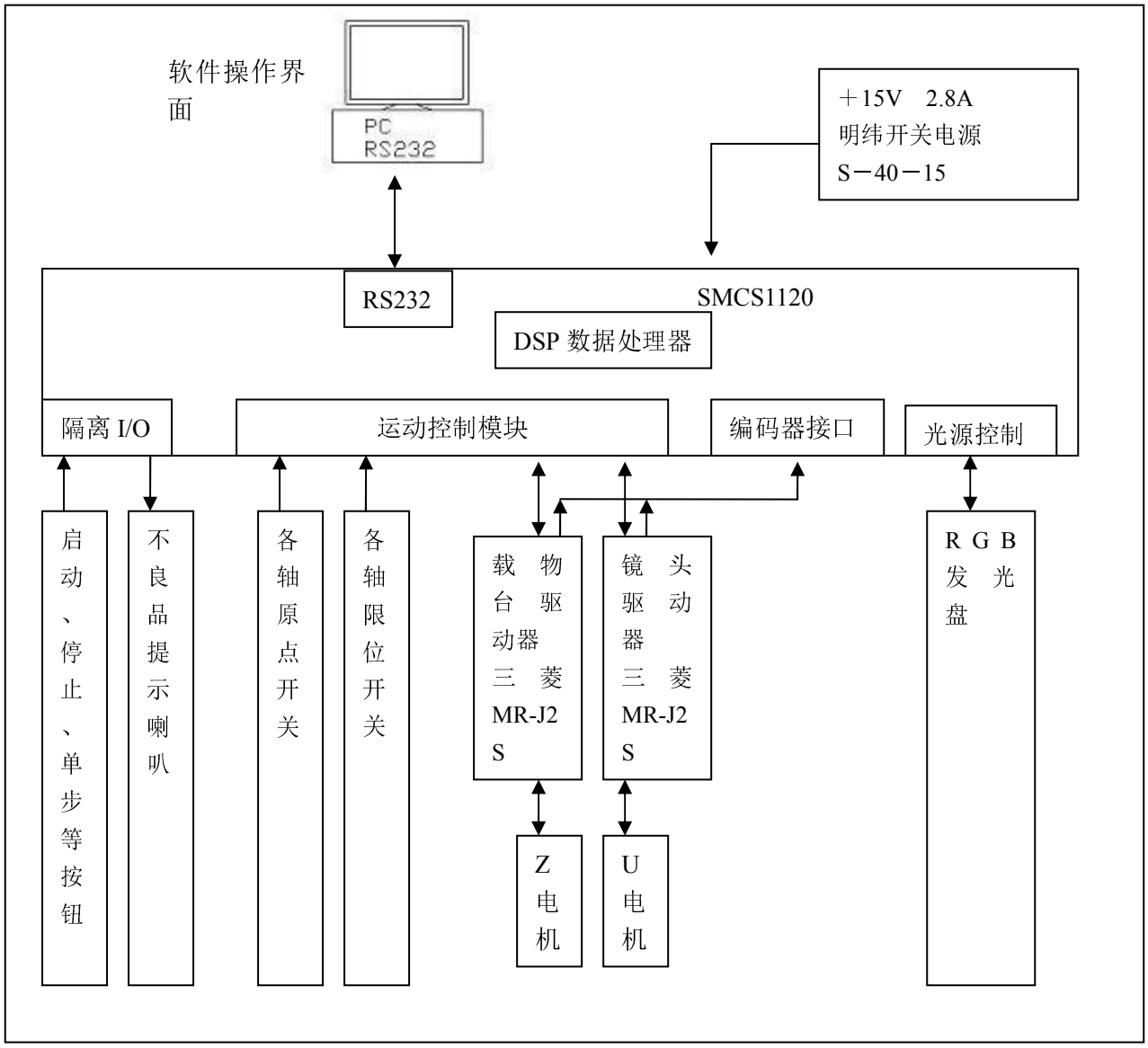

图 3－1 SMCS1120 运动控制器典型应用连接示意图

3．2 连接器引脚定义与功能介绍

3．2．1 接线座 CN1／CN2：电机驱动器信号接口

CN2 是第一轴驱动器信号接口，CN1 是第二轴驱动器信号接口，其接脚定义完全相同。

表 3－1 SMCS1120－接线座 CN1／CN2 引脚定义

引脚编号	引脚定义	引脚说明	引脚编号	引脚定义	引脚说明
1	COM－	＋12－＋24V 电源负极输出，接交流伺服驱动器 COM－	14	COM＋	＋12－＋24V 电源正极输出，接交流伺服驱动器 COM＋
2	ALARM	伺服报警输入	15	－	保留
3	OC_Out	集电极开路输出 *	16	INP	伺服定位完成
4	A－	编码器输入 A－	17	A＋	编码器输入 A＋
5	B－	编码器输入 B－	18	B＋	编码器输入 B＋
6	－	保留	19	－	保留
7	＋5V	＋5V 输出（55mA 限制）返回路径为 GND	20	GND	＋5V 输出返回路径（0V）
8	－	保留	21	GND	编码器信号参考电位
9	DIR＋	方向信号输出＋	22	DIR－	方向信号输出－
10	GND	方向信号参考电位（0V）	23	PULSE＋	脉冲信号输出＋
11	PULSE－	脉冲信号输出－	24	GND	脉冲信号参考电位（0V）
12	－	保留	25	－	保留
13	GND	0V	SHELL		接信号电缆屏蔽网

*：集电极开路输出，在 CN2 中，该输出是可编程通用输出 0，在 CN1 中该输出是可编程通用输出 1；SMCS1120 上电后该输出为高电平，可通过软件设置输出状态。

以下为自动光学检查仪使用最普遍的三菱交流伺服驱动器 MR－J2S 接线图。

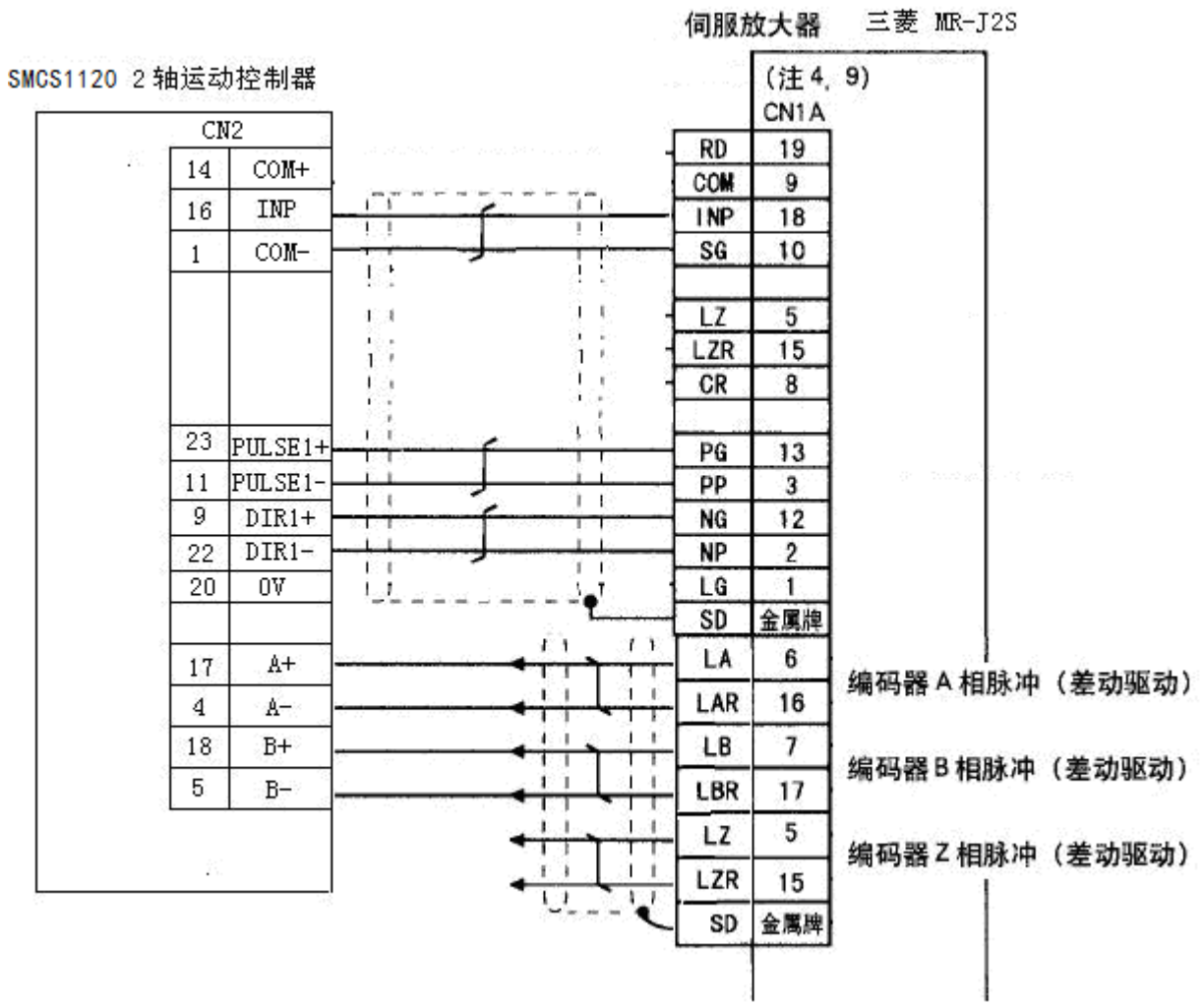

图 3－1 SMCS1120 与三菱交流伺服驱动器连接

图中虚线框表示信号线需要使用屏蔽电缆并将屏蔽层接驱动器 CN1A 外壳；⫞表示这两根信号线需使用双绞线。建议使用屏蔽双绞线制作 SMCS1120 与交流伺服驱动器连接电缆，以增强抗干扰能力提高系统可靠性。

注意：使用 MR－J2S 时，位置控制指令输入应设置为 001，错误的设置将导致定时时间延长甚至接收不到指令。

3．2．2 接线座 CN3：RS232 接口

该接口采用标准 RS232 数据线（一端公一端母，两端引脚按引脚编号一一对应连接）与计算机 RS232 接口直接相连即可。

表 3－2 SMCS1120－接线座 CN3 引脚定义

引脚编号	引脚定义	引脚说明	引脚编号	引脚定义	引脚说明
1	－	空	6	－	空
2	TXD	发送端	7	－	空

3	RXD	接收端	8	—	空
4	—	空	9	—	空
5	GND	0V	SHELL	0V	通讯电缆屏蔽网

所有信号接口均禁止带电状态拔插连接器、紧固或脱开接线端子，连接器一旦插上，请随手拧紧固定锣丝以免遗忘。

3．2．3 接线座 CN4：发光盘接口

光源的 2 种工作模式：

连续照明模式：通过软件设置为连续照明模式（见 4.3.22）后，可由 PC 软件开启光源，发光盘各颜色以设定亮度持续点亮，PC 软件可随时开启 / 关闭。在手动操作机器以及编程时使用该模式。在频闪模式时光源受 SMCS1120 控制，PC 软件不可以开启光源。

由于持续点亮 LED 将导致 LED 温度升高影响寿命，为保护发光盘 LED 使用寿命请在不使用时将模式切换到频闪模式。发光盘连续发光时间不应超过 1 小时。

频闪模式：SMCS1120 电源开启后默认为该模式。所有轴均停止运动光源处于关闭状态。SMCS1120 检测到第 0 轴或者第 1 轴启动后便准备闪光；当第 0 轴和第 1 轴即将停止时（提前 1 毫秒），开启光源，光源持续设定的闪光时间后自动关闭。若停止后软件立即令控制器再次运动，且闪光时间大于一个启停运动时间时光源将持续点亮。频闪模式发光盘的亮与灭完全由控制器自行控制，不需要 PC 软件干预。

频闪光源在从开启到亮度达到稳定状态仅历时数个微秒，因此轴停止后即可抓取图片，无需等待亮度稳定。光源闪光时间应大于 CCD 快门时间和软件查询周期，一般可设为 50－100 毫秒；由于 WINDOWS 操作系统的非实时性，并不能保证闪光结束之前 PC 软件能采集到 CCD 的图片，因此用户有必要对抓到的图片进行检测，如果图片是在闪光时间结束之后抓取的，应切换到连续照明模式开启光源抓取图片，之后再切换为频闪模式，否则容易出现误判现象。

表 3－3　SMCS1120－接线座 CN4 引脚定义

引脚编号	引脚定义	引脚说明	引脚编号	引脚定义	引脚说明
1	—	空	6	V＋	SMCS1120 的供电电压正端输出
2	LIGHT1	红色光源驱动输出	7	V＋	
3	LIGHT2	绿色光源驱动输出	8	V＋	
4	LIGHT3	蓝色光源驱动输出	9	V＋	
5	LIGHT4	白色光源驱动输出	SHELL		接光屏蔽网

光源输出电路每一路最大可带 100mA 负载，若发光盘每种颜色光源最大工作电流小于 50mA 以下，或者大于 100mA 请通知我们，我们将在出厂时修改最大负载能力，否则调光范围大大缩短，影响光源使用。

光源输出电路为吸入电流的集电极开路输出方式，发光盘应采用共阳极接法，请将 R G B W 红、绿、蓝、白各色 LED 正公共端接 V＋，负公共端接控制端，如无白色 LED 则 LIGHT4 可不接。

由于自动检测时发光盘工作于频闪模式，LED 仅瞬间工作，因此 LED 工作瞬间可给予 LED 连续工作电流数倍的电流额定峰值电流以内的工作电流，充分发挥 LED 的亮度，类似照相机闪光灯原理，所以工作在频闪方式下的发光盘只需要传统光源一半以下的 LED 数量便可达到传统光源的亮度。编程时期可用软件将光源设置为连续照明模式，亮度应适当降低，由于编程时间较短，LED 晶片温度尚未达到临界温度，不会对 LED 带来伤害；而自动检测时间则加大 LED 的亮度，达到 CCD 所需要的最佳照明要求。

发光盘连接示意图如下：

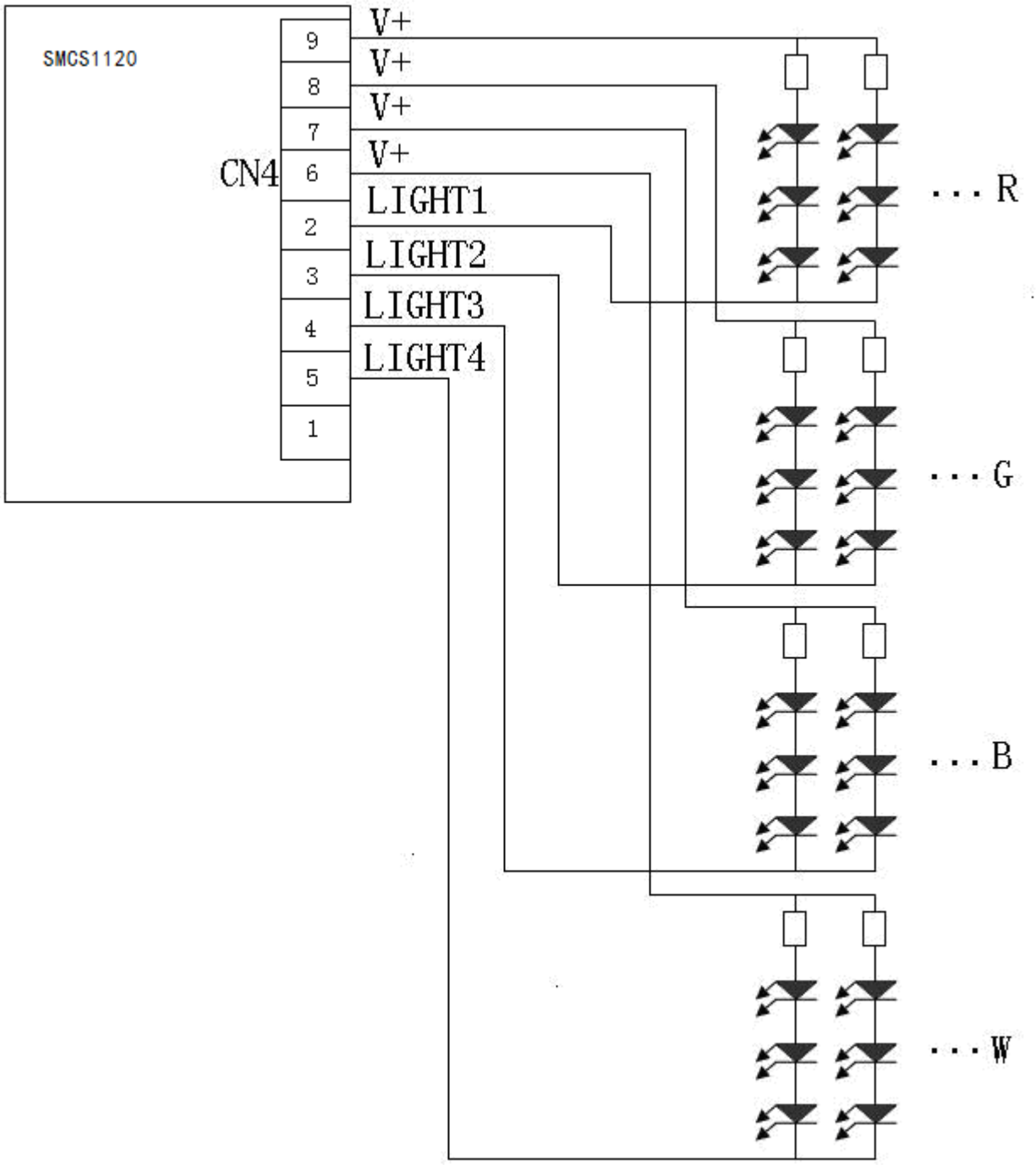

图 3－2 SMCS1120 与发光盘连接示意图

3．2．4 接线座 CN5：限位开关／按钮开关接口

表 3－4 SMCS1120－接线座 CN5 引脚定义

引脚编号	引脚定义	信号说明	引脚编号	引脚定义	信号说明
1	COM+	光电隔离电路电源正端	14	COM-	光电隔离电路电源负端
2	LM_SW1+	第1轴正限位信号输入	15	COM-	光电隔离电路电源负端
3	LM_SW1-	第1轴负限位信号输入	16	COM-	光电隔离电路电源负端
4	LM_SW2+	第2轴正限位信号输入	17	IN5	通用光电隔离输入 5
5	LM_SW2-	第2轴负限位信号输入	18	IN6	通用光电隔离输入 6
6	HOME1	第1轴原点信号输入	19	+3.3V	+3.3V 输出(最大 40mA)
7	HOME2	第2轴原点信号输入	20	COM-	光电隔离电路电源负端
8	IN1	通用光电隔离输入 1	21	COM-	光电隔离电路电源负端
9	IN2	通用光电隔离输入 2	22	COM-	光电隔离电路电源负端
10	IN3	通用光电隔离输入 3	23	NC	保留
11	IN4	通用光电隔离输入 4	24	NC	保留
12	OUT3	通用光电隔离输出 3	25	COM-	光电隔离电路电源负端
13	OUT4	通用光电隔离输出 4			

第 1 轴正限位开关接线为例连接示意图如下：

- 用微动开关作为限位／原点开关

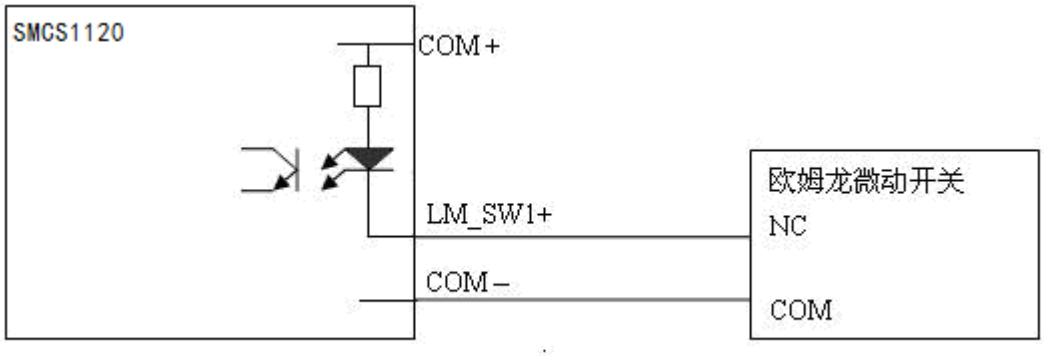

图 3－3 使用微动开关作为限位开关

- 用光电开关作为限位／原点开关

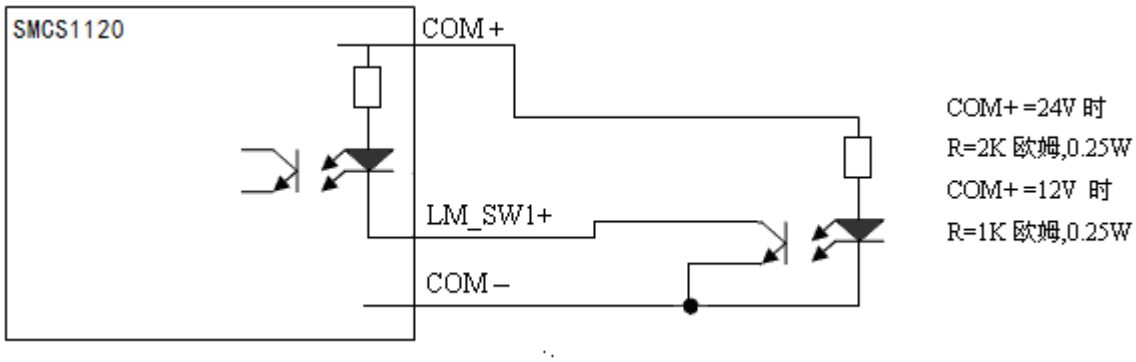

图 3－4 使用光电开关作为限位开关

- 使用其它形式开关连接方式与上述基本相同，如有疑问，请咨询技术支持人员；SMCS1120 使用常开或者常闭型限位开关 / 原点开关均可，但如果条件许可应尽量使用常闭型限位开关。作为常识，与常开型开关相比，常闭型开关上的信号更不容易被其它信号干扰到，更可以预防由于导线反复弯折导致的断线现象。

注意：CN5 与 CN1、CN2 为同一形状，如果接错可能发生故障！

3．2．5 连接电源

SMCS1120 工作电压范围比较宽，图 3－5 中使用 15V 开关电源。由于该电源同时给控制器内部光源电路供电，所以实际提供给 SMCS1120 的电源由发光盘的工作电压决定。

如果发光盘上 LED 为每 3 个串联为一组，然后多组并联的连接方式，应使用 15V 电源；如果每 6 个串联为一组，则应使用 24V 电源。

由于每颗高亮度 LED 工作电压为 3.3V 左右，加上控制器调光电路消耗 4V 左右电压，因此 SMCS1120 工作电压可由如下公式计算出来：

U＝3.3 × N +4 (V)

开关电源输出电压应等于或者大于计算结果，但不得大于 4V 以上。

控制器工作电源必须使用开关电源供电，禁止使用＂整流一滤波＂的非稳压电源。

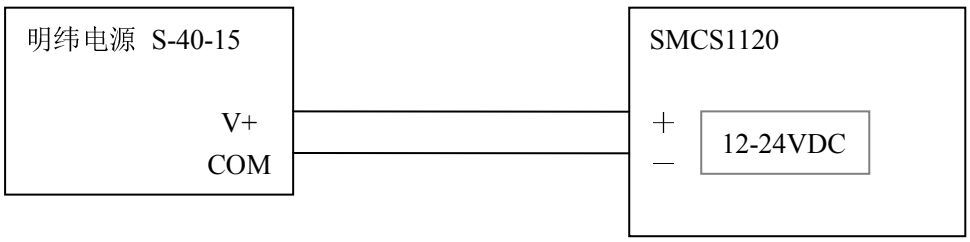

图 3－5 SMCS1120 电源供应方式

3．2．6 连接计算机

SMCS1120 采用工业标准总线 RS232 与计算机通讯，采用标准 9PD 连接电缆（一头公－母 9PD 插头）与计算机 COM 连接即可，如果 1 条电缆长度不够，可用 2 条转接，但应注意总长度不要超过 9 米。

若计算机后面板没有 COM 口，则电脑主板上一般都有一个 10PIN 双排插针形式的 COM 口插座，使用 IDC 转接线转接至计算机机箱上即可。

若计算机 COM 口不够用，可用 USB 转 RS232 接口转接线连接 SMCS1120 至计算机空闲的 USB 接口，电缆长度不够时，可在转接线的 RS232 接口与 SMCS1120 之间接入转接线，而不要在 USB 插头与 PC 之间接入 USB 中继器，这是因为 RS232 信号的抗干扰和数据纠错能力远远好过于 TTL 电平的 USB 信号。

3．3 SMCS1120 演示程序

我们提供了一个调试程序，可供熟悉控制器和调试 AOI 主机，运行界面如下：

图 3－6　SMCS1120 功能演示程序启动界面

右边三个按钮，分别打开：参数设置界面、运动控制界面。SMCS1120 的大部分功能将这 3 个操作界面得到展示。

界面下方提示框显示该演示程序与控制的连接情况，如果连接异常，则会有相应的提示，请按提示进行操作。

- 参数设置画面

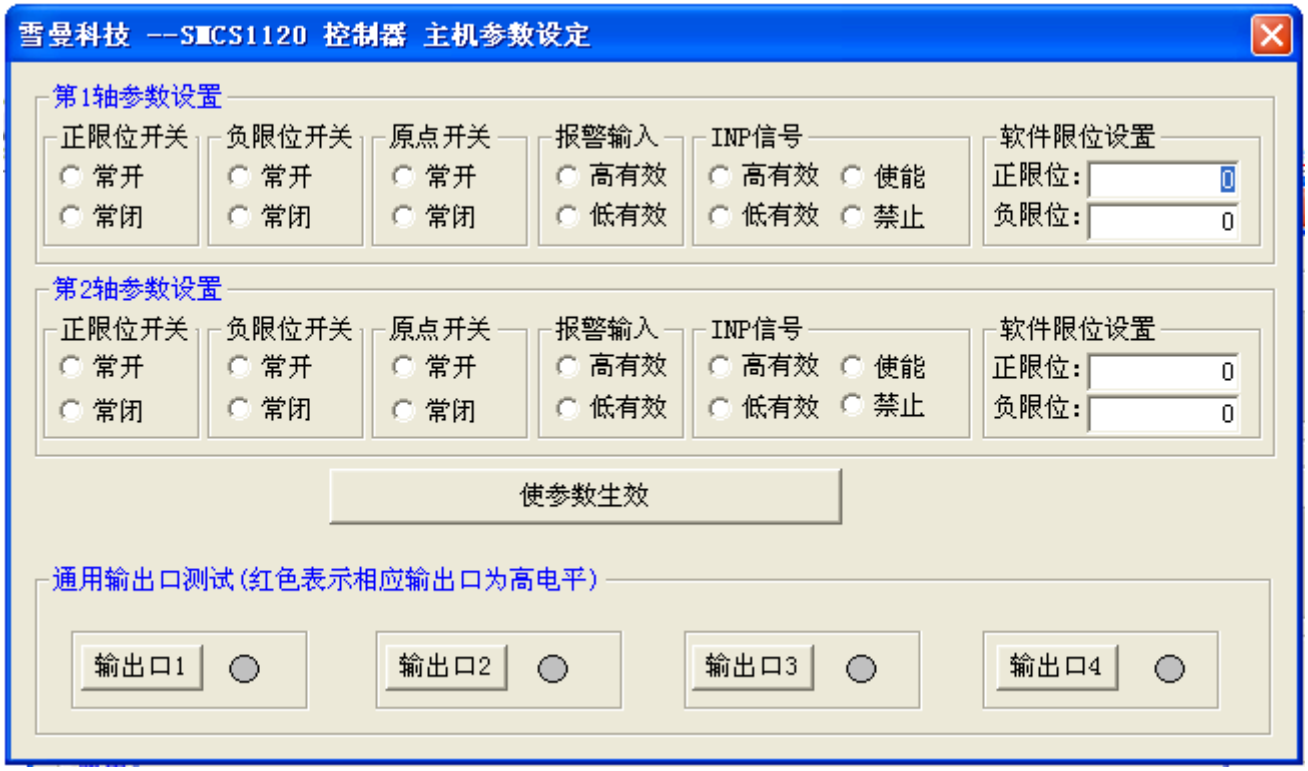

图 3－7 SMCS1120 演示程序参数设置界面

一个典型的运动控制轴连接示意图如下：

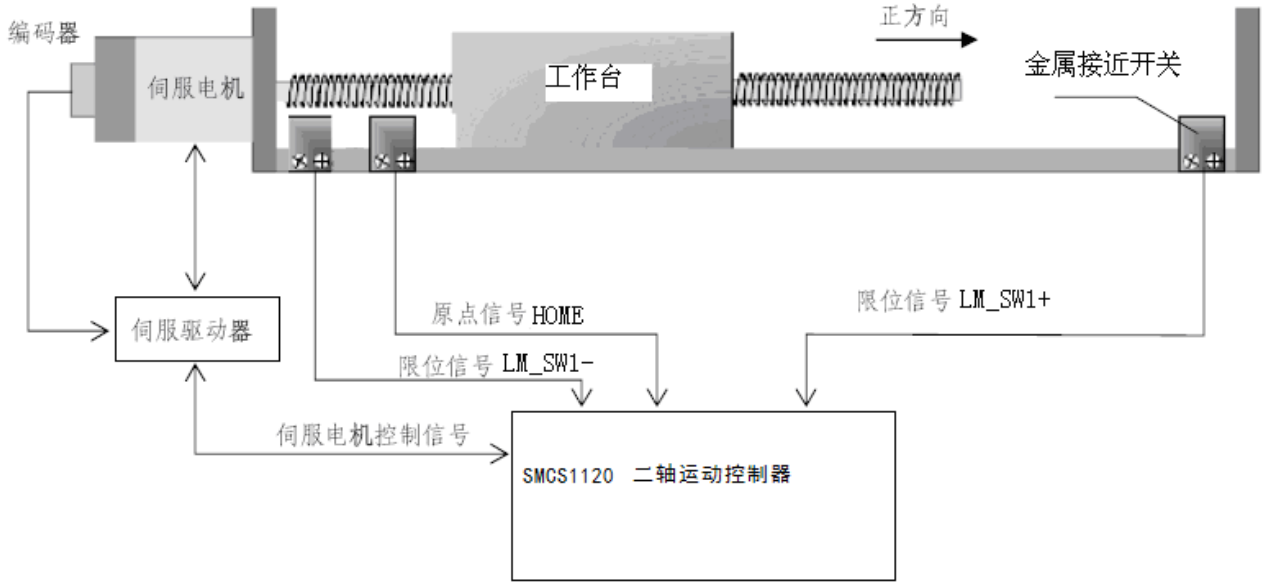

图 3－8 单个运动轴电气连接示意图

图中各种开关类型和伺服电机接口信号有效电平配置形式均需要设置。

参数设置窗口可设置各轴限位开关、原点开关为常开类型或者是常闭（对于金属接近开关，则是设置 NPN 或 PNP）类型；同时也可以设置伺服电机驱动器接口信号有效电平，INP 信号是否使能。当伺服驱动器 INP 信号正确连接后，可将 INP 信号使能，这样每个点位运动结束后运动控制器等待伺服驱动器的 INP（定位结束）信号到来后才置轴状态寄存器 " 运动完成 " 标志位为 1，否则继续等待，INP 信号为正脉冲或者负脉冲均可，持续时间须大于 1mS 以上。错误的设置可能导致 INP 信号无效或者运动完成标志无法置 1。

回原点模式时控制器将忽略软件限位设定值。软件限位一般设置在两限位开关以内的某个范围，速度模式时运动超出软件限位或者限位开关范围时控制器将使该轴减速停止，点位运动模式时若设定目标位置超出软件限位范围，控制器将拒绝执行该命令，同时警告目标位置设置错误。软件限位作为一种辅助手段，可提高系统可靠性，尤其是调机阶段可方便的设置软件限位位置，确保调机时机器在安全范围内运动。

在该画面还可以测试输出口，调试声光提示装置。

参数设置完毕后需单击【使参数生效】按钮才会起作用。

- 运动控制演示界面

图 4－19 SMCS1120 演示程序运动控制演示界面

左上角为轴状态显示区，与轴状态寄存器各位状态一一对应。

需要注意的是，限位信号灯泡亮不代表限位信号引脚输入电平的状态，该标志代表该限位已触发，因为限位开关的电平状态是可以在＂参数设置界面＂设置的。

右上角为指令位置显示区，显示当前轴指令位置（即运动控制器发送出去的脉冲数）和实际位置（即光栅尺或者编码器计数值），一般情况下实际位置与指令位置相差 1 个脉冲数左右（与驱动器参数调试有关）。回原点结束后该位置显示为 0；使用【位置清零】按钮请注意，该按钮按下后控制器将当前位置设置为原点，因此软件限位值失去意义，需再次使机器回原点。

＂回原点模式＂栏可用于各轴回原点演示。速度设置范围为 5Hz～100，000Hz，加减速时间设置范围为 0.001 秒～5 秒。

＂梯形速度曲线设置＂栏用于设置各轴加减速时间（加速度）和点位模式以及速度模式的初速度和最大运行速度，速度设置范围为：5Hz～500，000Hz，初速度一般为最大速度的 1 / 10～1 / 3。超范围的设置将导致函数返回值非 0，下方状态栏显示调用函数名以及函数返回值，只能在停止状态下设置速度曲线。

右侧两个大按钮【全部减速停止】和【全部立即停止】用于停止当前轴的运动，也是退出速度模式唯

一的方法。

" 点位运动模式 " 栏可演示各轴相对 / 绝对点位运动，在当前轴点位运动未结束前，当前轴的其它轴模式运动不可同时进行，停止按钮则不受任何模式的限制。

" 速度模式 " 按钮可演示各轴速度模式运动，这在手动编程时非常有用。在停止模式下，单击【启动 0 轴】按钮，对应轴的滑块变为可用，朝正方向拉动滑块时，机器朝正方向运动。滑块拉到最大位置时，运动速度达到 " 梯形速度曲线 " 栏设置的最大速度，松开滑块后滑块自动回到中间位置，速度在 " 加速时间 " 栏的设置时间内从最大速度减速到初速度，若滑块拖动速度为最大速度的 1 / 2，则减速到 0 的时间也变为设置时间的 1 / 2。退出速度模式的唯一方法是单击右侧两个大按钮【全部减速停止】或者【全部立即停止】。

" 光源控制 " 栏用于测试 / 调试光源亮度、颜色等。光源的工作模式分为频闪模式和连续照明模式：

频闪模式：在每一次运动结束前 5 个毫秒内以较连续照明高一倍左右的亮度开启光源，持续设定时间后将光源关闭，从而减少 LED 无谓的工作时间，延长其寿命。

第四章：软件开发

4．1 一般开发步骤

1. 熟读函数介绍部分，了解函数功能、参数以及返回值、调用方法。
2. 在 VC 工程中加入 SMCS1120 .H 和 SMCS1120 .LIB。
3. 这样，SMCS1120.DLL 中的 API 函数就可以像系统 API 一样调用了。注意：如果需要多线程编程，请使用互斥对象进行同步，避免同一 API 函数在多个线程中同时调用，如果在不同线程中不可能调用同一 API 函数，则不需要线程同步。
4. 在软件初始化部分调用初始化函数，在软件退出部分调用关闭 SMCS1120 的函数，操作比较简单，典型流程如下：

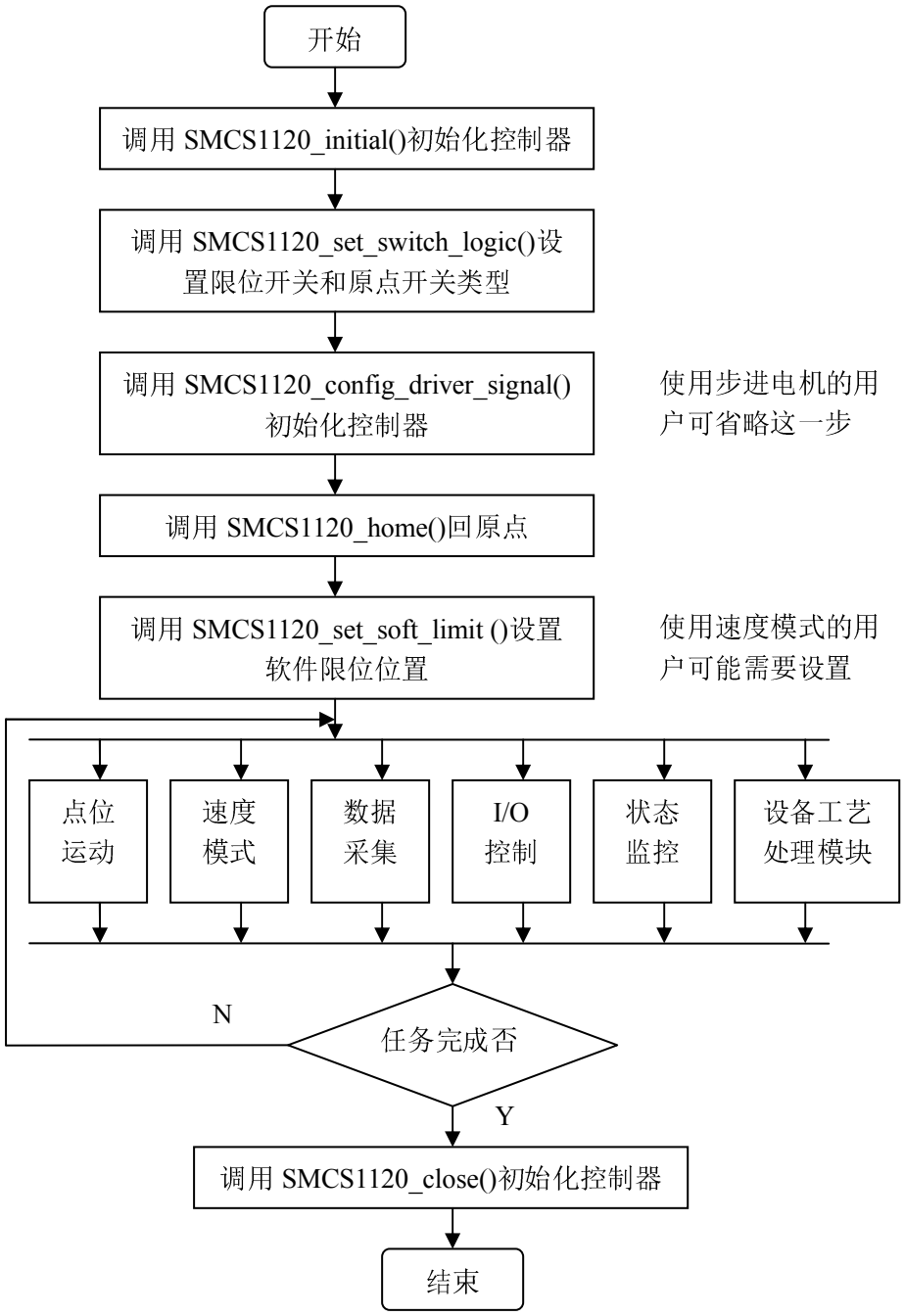

图 4-1 SMCS1120 运动控制 系统软件开发处理流程

4．2 SMCS1120 软件编程思路介绍

SMCS1120 的 I / O 操作和光源控制比较简单，见第 4 章软件开发相关的 API 函数介绍章节即可。运动控制部分每个轴的操作是完全相同的，SMCS1120 使用的关键是了解运动轴的 4 种工作模式：

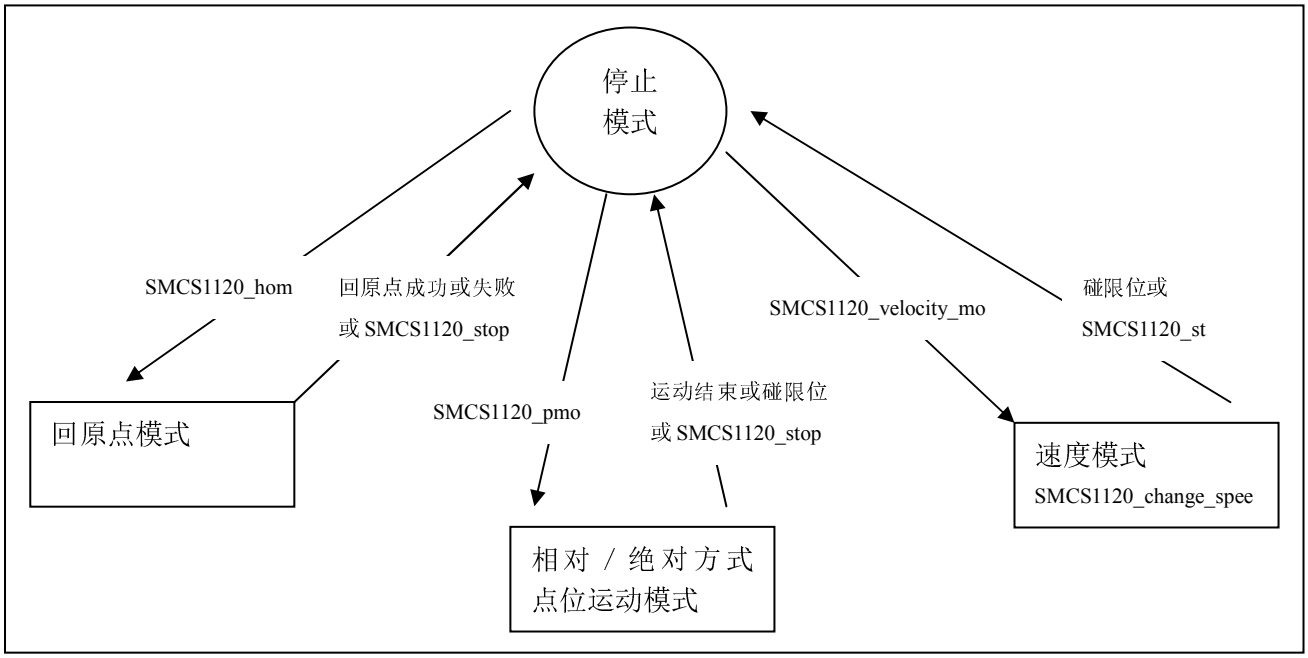

图 3－2 SMCS1120 工作模式转换图

从图中可以看出，停止模式是 4 种模式中的过渡，回原点模式、点位运动模式和速度模式这三个模式的切换必须经历停止模式；控制器电源开启后处于停止模式。

- 控制器处于停止模式的标志是轴状态字 bit10 为 1，用 SMCS1120_get_status()可以查询到该状态字。
- 从停止模式到回原点模式的方法是调用 SMCS1120_home()函数；退出回原点模式的方法是调用停止函数 SMCS1120_stop()，或者等待回原点结束。
- 从停止模式到点位运动模式的方法是调用 SMCS1120_pmove()函数；退出点位运动模式的方法是调用停止函数 SMCS1120_stop()，或者等待点位运动结束，如果点位运动目标位置设置超过限位（限位开关），运动触发限位时也会退出点位运动模式。
- 从停止模式到速度模式的方法是调用 SMCS1120_velocity_move()函数；退出点位运动模式的方法同样是调用停止函数 SMCS1120_stop()，如果速度模式运动时碰到限位（限位开关和软件限位位置），也会退出速度模式。

4．2．1 停止模式

SMCS1120 电源开启后进入该模式，各轴均保持停止状态；从停止模式转入其它模式前需做如下工作：

- 设置限位开关和原点开关是常开型还是常闭型，每个开关均可通过软件设置；相关函数：SMCS1120_set_switch_logic()；
- 配置驱动器接口信号；相关函数：SMCS1120_config_driver_signal()；
- 设置梯形运动速度曲线；相关函数：SMCS1120_set_move_speed()；
- 设置 I / O 口状态。相关函数：SMCS1120_set_output()；

4．2．2 回原点模式

确认将要操作的轴处于停止模式后，调用 SMCS1120_home(unsigned int axis,int dir,long speed,double acc_time)函数启动回原点，回原点速度曲线为梯形速度曲线，参数 speed 和 acc_time 确定梯形速度曲线参数，如下：

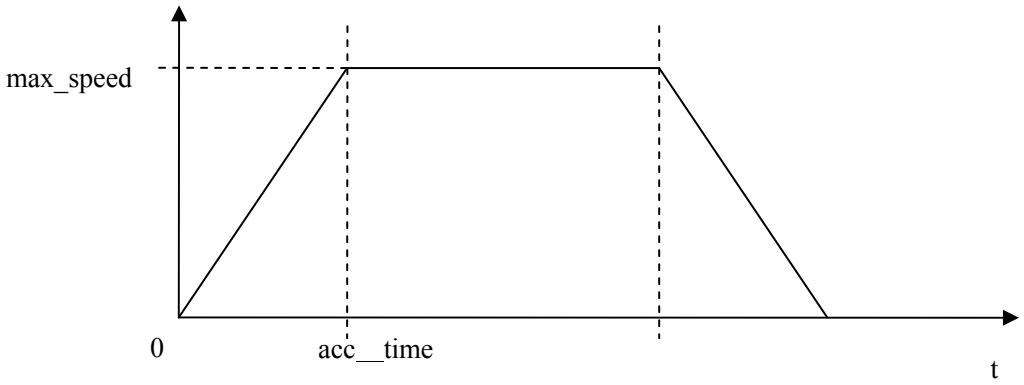

图 3－3 回原点运动时梯形速度曲线的设定

设调用代码为：

SMCS1120_home(1,-1,5000,0.2);

该函数调用后立即返回，第 1 轴状态字 bit10 首先清为 0，之后开始运动，将出现如下几种运动情形：

A. 以 5000 个脉冲每秒（pps）的速度朝负方向找原点开关，如果原点开关触发，则该轴将减速－－停止－－低速（speed 的五分之一）回退，直到原点开关刚好处于未触发状态时立即停止，并将第 1 轴指令位置寄存器清 0，回原点成功结束，轴状态字 Bit10 置 1。

B. 若朝负方向找原点时触发了负限位开关，则该轴将减速－－停止－－朝正方向找原点开关，若找到原点开关时，情形与 A 相同，回原点成功。若正方向运行时在找到原点开关之前触发了正限位开关，则减速停止，找原点运动以失败结束，轴状态字 Bit10 保持为 0。

C. 若朝负方向找原点时，在找到原点开关之前触发了正限位，则该轴减速停止，找原点运动以失败结束，轴状态字 Bit10 保持为 0。

D. 调用 SMCS1120_stop()函数，该轴减速停止，找原点运动以失败结束，轴状态字 Bit10 保持为 0。

朝正方向回原点的过程与负方向类似，不再赘述，在轴回原点期间可调用其它轴运动函数或者 I / O 操作函数等。

4．2．3 点位运动模式

确认将要操作的轴处于停止模式后，调用 SMCS1120_set_move_speed(unsigned int axis,unsigned long start_speed,unsigned long max_speed,double acc_time)设置点位运动梯形速度曲线参数，参数 start_speed、max_speed 和 acc_time 确定梯形速度曲线参数，如图 3－5 所示：

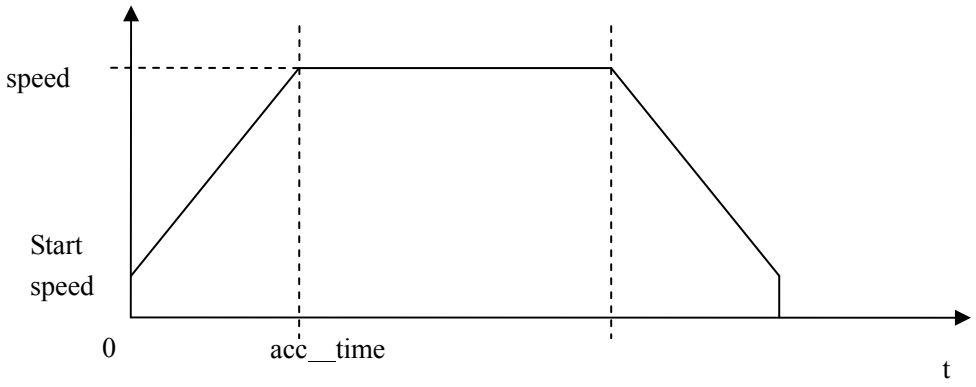

图 3－4 点位运动时梯形速度曲线的设定

合理的设置 start_speed，可减少加／减速过程持续时间而又不对机构造成冲击，可减少单位次数运动持续时间，这在频繁的大量的点位运动应用中尤为有效。Start_speed 一般为 max_speed 的 1／10 至 1／3，越高刚性的传动机构其初始速度设置越小；如采用高刚性联轴器与丝杆直连的情形应设置为 1／10，而采用皮带传动的轴则可设置高达 1／3。

之后调用 SMCS1120_pmove(unsigned int axis,long pos,unsigned int mode)函数启动点位运动。

设该函数调用之前位置为 2000，调用代码为：

SMCS1120_pmove (1, 5000,0);

即相对坐标方式运动，该函数调用后立即返回，第 1 轴状态字 bit10 首先清为 0，之后开始运动，将出现如下几种运动情形：

- 运动到指令位置寄存器为 7000 的位置时运动完成，bit10 置 1，退出点位运动模式，进入停止模式。
- 限位开关被触发，减速停止并退出点位运动模式，进入停止模式。
- 调用 SMCS1120_stop()函数，该轴减速停止，轴状态字 Bit10 变为 1，进入停止模式。

若调用代码为 SMCS1120_pmove (1, 5000,1)，即绝对坐标运动方式，该轴运动到指令位置寄存器为 5000 时运动完成。

4．2．4 速度模式

确认将要操作的轴处于停止模式后，与点位模式一样调用 SMCS1120_set_move_speed()函数设置点位运动梯形速度曲线参数。

之后调用 SMCS1120_velocity_move(unsigned int axis)函数进入速度模式，进入速度模式后运动速度为 0，接着调用 SMCS1120_change_speed(unsigned int axis, long speed)函数改变运动速度为其它值。

进入速度模式后第 1 轴状态字 bit10 首先清为 0，之后开始运动，可能出现如下运动情形：

- 使用 SMCS1120_change_speed（）函数将速度设为非零值，若速度为正，则该轴正向运动，若速度为负，则减速为 0 之后朝负方向加速到设定速度后运行。
- 限位开关或者软件限位被触发，减速停止并退出点位运动模式，进入停止模式。
- 使用 SMCS1120_change_speed（）函数将速度设为零值，该轴减速为零并保持速度模式不变。
- 调用 SMCS1120_stop()函数，该轴减速停止，轴状态字 Bit10 变为 1，进入停止模式。

4．2．5 操纵杆控制模式

该模式为 NC 机所特有。在该模式下将通用输入口 6 输入低电平后控制器检测操纵杆位置，并以操纵杆偏离中心位置的偏移量作为速度和方向控制的依据，使得 AOI 不必配置 PC 便可运动。

该模式适用于不配置 PC，并采用步进驱动系统的经济型 AOI。

4．3 SMCS1120 API 函数介绍、函数汇总表

SMCS1120 遵循严谨的编程思路，综合点位运动设备和需要较强采集／控制功能的应用特点设计了一套简单易用的 API 函数，这些函数对传入参数做严格的检查（取值可能值、取值范围和取值和法性检查），若参数无规则错误，则予以继续执行，并返回执行结果；无论如何，API 函数都会返回一个值，以确保程序执行在程序员掌控之中，返回值为 0 表明执行成功，可继续执行，返回其它值表明执行失败，可能是传入参数异常，也可能是通讯线路中断，或者根本是 SMCS1120 电源没有开启，查表 4－1 可知出错原因，应进行相应的处理。

作为简单的硬件控制程序接口，函数和变量类型比较简单，命名方式并没有采用 C＋＋编程中流行的匈牙利命名法，而是采用全小写字母的下划线命名法，在控制程序中这种方式比匈牙利法更易于阅读，不用来回输入大小写字母，减少代码录入的劳动强度。

对返回值进行检测的建议：

1） SMCS1120 初始化函数必须检查返回值，如果返回值表明操作失败，则后续调用初始化函数之外的所有 API 函数均会以失败返回；函数返回失败时，引用参数所传回的值无意义。

2） 参数设置、参数保存保存函数的返回值需要进行检测。

3） 在很短时间(数十到数百个毫秒)便可执行完的一连串的函数调用时（比如读数线程中对各轴位置进行读取），为减少程序复杂性，可只在第一条函数调用时对返回值进行检查并作必要的处理。

SMCS1120 API 函数返回值对异常现象进行了编码，所有 API 函数均遵循这个规则：

表 4－1 SMCS1120API 函数返回值含义表

返回值	含义
0	操作成功
1	第1个参数错误
2	第2个参数错误
3	第3个参数错误
4	第4个参数错误
5－6	保留
7	当前状态下不允许进行的操作；
8	设备没有初始化或者已确定通讯中断
9	初始化失败，有空闲的COM口，但没有发现连接有处于通电状态的SMCS1120
10	没有空闲的COM口
41	本次通讯超时。SMCS1120的超时时间为1秒
42	本次通讯遭受严重干扰，通讯数据出错，但通讯还未完全中断

正常情况下，返回值均为 0；当 SMCS1120 未初始化时便调用该函数或者软件正在运行中，通讯电缆拔除或者 SMCS1120 电源被关闭，返回值为 8；返回值 41～43 基本上不会出现，但作为一种可能，仍然有必要罗列出来。其它 API 函数与此类似，不再逐条说明。

为使用户建立整体了解，现对 API 函数汇总如下，并作简要介绍，详细请参见各个函数介绍。

表 4－2 SMCS1120API 函数汇总表

编号	函数原型	函数说明
1	int SMCS1120_initial(long baudrate);	初始化 SMCS1120。该函数以指定的波特率初始化串口并在空闲串口（包括由 USB 转接而成的串口）中查找处于通电状态的 SMCS1120。
2	int SMCS1120_close(void);	关闭 SMCS1120
3	int SMCS1120_set_switch_logic(unsigned int logic);	设置所有轴限位开关和原点开关是常开类型还是常闭类型

4	int SMCS1120_get_switch_logic(unsigned int *logic);	读取所有轴限位开关和原点开关类型设置
5	int SMCS1120_config_driver_signal (unsigned int axis,unsigned int mode);	配置指定轴驱动器接口信号有效电平和是否使能
6	int SMCS1120_get_driver_config (unsigned int axis,unsigned int *mode);	读取指定轴驱动器接口信号配置
7	int SMCS1120_get_status (unsigned int axis,unsigned int *status);	读取指定轴状态字
8	int SMCS1120_home(unsigned int axis,int dir,long speed,double acc_time);	命令指定轴以指定的方向和速度回原点
9	int SMCS1120_set_move_speed (unsigned int axis, unsigned long start_speed,unsigned long max_speed,double acc_time);	设置指定轴梯形速度曲线参数，包括最大速度和加／减速时间
10	int SMCS1120_pmove(unsigned int axis,long pos,unsigned int mode)	命令指定轴以绝对/相对方式运动
11	int SMCS1120_velocity_move(unsigned int axis)	命令指定轴以速度模式运行
12	int SMCS1120_change_speed(unsigned int axis, long speed)	指定轴以速度模式运行时，改变该轴运行速度和方向
13	int SMCS1120_stop (unsigned int axis,unsigned int mode)	命令指定轴减速停止或者立即停止
14	int SMCS1120_get_position(unsigned int axis,long *pos)	读取指定轴当前命令位置
15	int SMCS1120_set_position(unsigned int axis,long pos)	重设指定轴当前命令位置
16	int SMCS1120_set_soft_limit(unsigned int axis,long pos_limit,long neg_limit)	设置指定轴正的和负的软件限位位置
17	int SMCS1120_get_soft_limit(unsigned int axis,long *pos_limit,long *neg_limit)	读取指定轴正的和负的软件限位位置
18	int SMCS1120_get_input(unsigned int *input)	读取指定通道编号的 16 个输入口高低电平状态
19	int SMCS1120_set_output(unsigned int status)	设置指定通道编号的 16 个输出口高低电平状态
20	int SMCS1120_get_output(unsigned int *status)	读取指定通道编号的 16 个输出口高低电平状态
21	int SMCS1120_reset_timeouts(void)	复位连接中断超时标记
22	int SMCS1120_set_light(unsigned int *light_value,unsigned int time,unsigned int mode,unsigned int on_off);	设置光源 R／G／B／W 亮度、工作模式以及开启或关闭光源（开启或关闭光源只有在连续照明模式才能进行）
23	int SMCS1120_get_encoder(unsigned int axis,long *pos);	读取编码器计数值
24	int SMCS1120_set_encoder(unsigned int axis,long pos);	设置编码器计数值

各函数详细介绍如下：

4.3.1 int SMCS1120_initial(long baudrate);

函数功能：该函数逐个查找PC上所有空闲的RS232接口是否连接有处于通电状态的SMCS1120计数器，如果找到便对其初始化。

参数说明：baudrate：以指定的波特率初始化计算机串行接口，该波特率应与SMCS1120相同。SMCS1120波特率出厂设定值为115200。

返回值说明:表4－1

使用举例:

```
int SMCS1120return;
 SMCS1120return = SMCS1120_initial (115200);
if(SMCS1120return !=0) {出错处理}
```

提示：

从该函数可以看出，仅当SMCS1120连接到计算机空闲的串口且处于通电状态时，调用该函数方可成功返回，若SMCS1120处于未连接或者未通电状态时，该函数将提示初始化失败。初始化失败后，继续执行SMCS1120_initial()之外的操作是无效的，将返回非0值。

4.3.2 int SMCS1120_close();

函数功能:关闭SMCS1120并释放其占用的串口。

参数说明:无

返回值说明: 表4－1

4.3.3 int SMCS1120_set_switch_logic(unsigned int logic);

函数功能：设置所有轴限位开关和原点开关是常开类型还是常闭类型；开关是常开类型时，输入低电平有效；是常闭类型时，输入高电平有效，如限位开关输入引脚输入高电平或者悬空，控制器认为限位开关已触发。该配置操作应该在运动控制器初始化之后，任何运动开始之前进行，否则可能导致机器认为限位开关一直触发而无法运动。

参数说明：

logic：该变量各位代表了限位开关和原点开关类型，见表4－3

表 4－3 开关类型配置

位编号	位定义	取值说明
Bit0	1 轴 正限位	0-常闭开关;1-常开开关
Bit1	1 轴 负限位	0-常闭开关;1-常开开关
Bit2	2 轴 正限位	0-常闭开关;1-常开开关
Bit3	2 轴 负限位	0-常闭开关;1-常开开关
Bit4-7	保留	保留，读为 0
Bit8	1 轴 原点	0-常闭开关;1-常开开关
Bit9	2 轴 原点	0-常闭开关;1-常开开关
Bit10-15	保留	保留，读为 0

返回值说明：表4－1

使用举例：

```
int SMCS1120return;
unsigned int logic=0xfff;//设置所有限位开关和原点开关为常开类型
SMCS1120return = SMCS1120_set_switch_logic(logic);
if(SMCS1120return !=0) {出错处理}
```

4.3.4 SMCS1120_get_switch_logic(unsigned int *logic);

函数功能：读取所有轴限位开关和原点开关类型设置。

参数说明：见表 4－3

使用举例：

```
int SMCS1120return;
unsigned int logic;
SMCS1120return = SMCS1120_get_switch_logic(＆logic); //读取所有轴限位开关和原点类型设定值，将
                                                     logic的地址传递给API函数以接收该函数传回的设定值
if(SMCS1120return !=0) {出错处理}
```

4.3.5 int SMCS1120_config_driver_signal(unsigned int axis,unsigned int mode);

函数功能:配置指定轴驱动器接口信号有效电平和是否使能这些信号，该配置操作应该在运动控制器初始化之后，任何运动开始之前进行，否则可能导致机器认为伺服报警一直触发而无法运动。

参数说明：

axis：轴编号，取值范围0-1；错误的取值将导致函数返回1。

mode：驱动器接口信号有效电平和是否使能，见表4－4

表4－4 驱动器接口信号配置

位编号	位定义	取值说明
Bit0	使能信号	0-低电平有效,1-高电平有效,默认低有效
Bit1	保留	保留，读为 0
Bit2	报警输入	0-低电平有效,1-高电平有效,默认低有效
Bit3	定位信号	0-低电平有效,1-高电平有效,默认低有效
Bit4－6	保留	保留，读为 0
Bit7	定位信号	0-禁止,1-使能
Bit8-15	保留	保留，读为 0

例：

Bit0：0-当调用函数使能驱动器时，该信号输出低电平。

＂bit2：0-低电平有效＂,即：该轴伺服报警输入脚为高电平时，控制器认为驱动器没有报警，输入为低电平时，控制器认为驱动器已报警；

Bit3：0：当伺服驱动器输入低电平时，控制器认为驱动器定位已完成。

返回值说明:表4－1

使用举例：

```
int SMCS1120return;
unsigned int mode=0;
```

```
SMCS1120return = SMCS1120_config_driver_signal (0, mode);
if(SMCS1120return !=0) {出错处理}
```

4.3.6 int SMCS1120_get_driver_config(unsigned int axis,unsigned int *mode);

函数功能：读取指定轴驱动器接口信号配置。

参数说明：

axis：轴编号，取值范围0-1；错误的取值将导致函数返回1。

mode：指向保存配置值的变量的指针。各位的定义见表4－4

返回值说明:表4－1

使用举例:

```
int SMCS1120return;
unsigned int mode;
SMCS1120return = SMCS1120_get_driver_config(0, &mode);
if(SMCS1120return !=0) {出错处理}
```

4.3.7 int SMCS1120_get_status(unsigned int axis,unsigned int *status);

函数功能：读取指定轴状态字，机器的运行状态主要表现在该状态字上，这也是控制程序运行的重要状态字之一。该状态字每轴均有一个，各轴独立。查询状态字常常用于检测点位运动是否完成，伺服是否运动到位，限位开关等信号的状态。

参数说明：

axis：轴编号，取值范围0-1；错误的取值将导致函数返回1。

status：各位含义如下：

表 4－5 轴状态寄存器

位编号	位定义	取值说明
Bit0	运动状态	0-当前速度为 0;1-当前速度不为 0
Bit1	当前运动方向	0-负方向;1-正方向
Bit2	保留	保留
Bit3	回原点标记	0-未回原点,或回原点记录已清除;1-已回原点
Bit4	保留	保留
Bit5	正限位状态	0-未触发,1-已触发
Bit6	负限位状态	0-未触发,1-已触发
Bit7	原点开关状态	0-未触发,1-已触发
Bit8	Inp 信号状态	0-无效,1-有效
Bit9	驱动报警	0-未报警,1-正处于报警状态
Bit10	运动完成	0-运动未完成,1-运动已完成
Bit11	正软件限位	0-未触发,1-已触发
Bit12	负软件限位	0-未触发,1-已触发

由于该寄存器对运动控制器的理解和使用非常重要，各位定义介绍如下：

Bit0：该位实时的反映轴的运行速度，无论该轴处于什么模式，速度为 0 时该位便为 0。

Bit1：该位实时的反映轴的方向信号，0 表示当前的方向信号为低电平（CN1 接线座上 dir+为低，dir-为高），1 表示当前的方向信号为高电平状态（dir+为高，dir-为低）。

Bit2 和 bit4：保留未用，读为 0。

Bit3：回原点标记；SMCS1120 电源开启后该位为 0，表明尚未回过原点；如果用户成功启动回原点操作，并回原点成功，该位将为 1 并一直保持，不成功为 0。SMCS1120_initial()函数对该位无影响，因此，如果软件启动后发现该位为 1，说明控制器已回过原点且到现在电源未曾关闭过，可以不必回原点直接投入工作状态，减少不必要的回原点时间消耗。如果用户再次启动回原点操作，控制器将该位设为 0 后启动回原点操作，直到找到原点或者调用 SMCS1120_stop（）取消运动。

Bit5 / Bit6 / Bit7：该位实时反映正限位 / 负限位 / 原点开关状态；0 表示未触发，1 表示已触发，与限位开关类型和限位开关类型设置有关，见 SMCS1120_set_switch_logic()函数。

Bit8：该位实时反映伺服驱动器 " 定位完成 " 信号状态，0 表示定位未完成，1 表示定位已完成，与驱动器接口信号配置有关，见 SMCS1120_config_driver_signal()函数。

Bit9：该位实时反映伺服驱动器 " 报警 " 信号状态，0 表示未报警，1 表示已报警，与驱动器接口信号配置有关，见 SMCS1120_config_driver_signal 函数。

Bit10：点位运动启动后该位为 0。若 INP 信号配置为使能时，运动控制器指令位置寄存器达到设定位置后，控制器等待伺服驱动器的 INP 信号变为有效，当 INP 信号有效后，控制器才将该位置 1。若 INP 信号配置为禁止状态，则控制器将脉冲指令发送完毕后立即置该位为 1。

速度模式运动启动后，无论速度是否为 0，该位均为 0，只有碰到限位开关（包括软件限位）或者调用 SMCS1120_stop（）后，该为才为 1。

Bit11/bit12：该位实时反映正 / 负软件限位的状态，0 表示当前位置未越过软件设置的位置，1 表示当前位置已越过软件设置的位置。该软件限位处理由运动控制器低层软件实现，不由 PC 软件实现，因此具有较高的实时性和可靠性。

需要提醒的是：判断 1 个轴是否停止，最稳妥的方法是判断 bit10，因为 bit0 在速度模式运动中时，如果速度为 0，bit0 也会为 0，但该轴还未 " 停止 "，运动控制器认为该轴处于 " 零速度运行 " 状态，与调用 SMCS1120_stop（）后的空闲状态是有本质区别的，这一点请务必加以注意。

返回值说明:表 4－1

使用举例:

```
int SMCS1120return;
unsigned int status;
SMCS1120return = SMCS1120_get_status (0, &status);
if(SMCS1120return !=0) {出错处理}
```

4.3.8 int SMCS1120_home(unsigned int axis,int dir,long speed,double acc_time);

函数功能：命令控制器指定轴以指定的方向和速度以及加减速时间找原点。原点开关被触发后将减速停止，并低速回退，直到原点开关恢复常态后立即停止运动，并将该轴位置设置为0；回原点的精度取决于原点开关的重复性，采用光电开关和霍尔开关具有较好的重复性，回原点精度约为0.05mm左右。

参数说明：

axis：轴编号，取值范围0-1；

dir：回原点方向；-1：朝负方向找原点，若未找到原点开关，但找到了负限位开关，控制器将减速并反向找原点开关，若反向找到原点开关之前碰到了正限位开关，控制器将终止找原点操作，轴状态寄存器Bit3保持为0；

speed：回原点的速度。对于SMCS1120而言，回原点的最大速度为100Kpps。

acc_time：回原点过程中速度从0到设定值的加 / 减速时间，单位为秒。

返回值说明:表4－1

使用举例:

```
int SMCS1120return;
SMCS1120return = SMCS1120_home (0,-1,10000,0.2); //令第 0 轴朝负方向以 10Kpps 速度找原点，加减速
```

时间为 0.2 秒；

if(SMCS1120return !=0) {出错处理} //出错原因一般为当前的点位运动未完成或者速度模式运动未停止

相关函数：SMCS1120_set_switch_logic（）；

4.3.9 int SMCS1120_set_move_speed(unsigned int axis,unsigned long start_speed,unsigned long max_speed,double acc_time)；

函数功能:设置指定轴点位运动模式和速度模式的梯形速度曲线，即设置起始速度、最大速度和加减速时间（即加速度）。在该函数只可以在停止模式调用。

参数说明:

axis：轴编号，取值范围0-1；

start_speed:起始速度；单位为pps（pulse per second），该速度应小于acc_time×90%。

max_speed：最大运行速度；单位为pps（pulse per second）。

acc_time：加减速时间，单位为秒；该参数设置小于0.01时，加减速时间固定为10ms；速度曲线接近矩形，只有非常短暂的加减速过程。

返回值说明:表4－1

使用举例:

```
int SMCS1120return;
SMCS1120return = SMCS1120_set_move_speed (0,150000,0.25); //第 0 轴最大速度为 150Kpps，加减速时间为 0.25 秒
if(SMCS1120return !=0) {出错处理}
```

相关函数：SMCS1120_pmove();SMCS1120_change_speed();

4.3.10 int SMCS1120_pmove(unsigned int axis,long pos,unsigned int mode)

函数功能: 命令指定轴以绝对/相对方式点位运动；运动中不可以改变速度。

参数说明:

axis：轴编号，取值范围0-1。

pos：当mode=0时，pos是相对运动量；当mode=1时，pos是绝对位置。

mode:0-相对运动，1-绝对方式运动。

返回值说明:表4－1

使用举例:

```
int SMCS1120return;
SMCS1120return = SMCS1120_set_position(0,0);
SMCS1120return = SMCS1120_pmove ( 0， 500， 0); //命令第 0 轴运动到距离当前位置正方向 500 个脉冲的位置,轴将停止在 500 的位置
if(SMCS1120return !=0) {出错处理}
SMCS1120return = SMCS1120_pmove ( 0， -700， 0); //命令第 0 轴运动到距离当前位置负方向 700 个脉冲的位置，轴将停在-200 的位置
if(SMCS1120return !=0) {出错处理}
SMCS1120return = SMCS1120_pmove ( 0， 600， 1); //命令第 0 轴运动到距离原点开关正方向 600 个脉冲的位置，轴将停在 600 的位置
```

```
if(SMCS1120return !=0) {出错处理}
```

相关函数：SMCS1120_set_move_speed();SMCS1120_get_status();

4.3.11 int SMCS1120_velocity_move(unsigned int axis);

函数功能:当指定轴处于停止状态时，用来启动该轴速度模式运动；速度模式启动后，该轴速度为0，之后调用SMCS1120_change_speed()函数可连续改变运动速度；这非常适合于通过鼠标或者操纵杆手动将轴定位在某一个位置，当距离目标位置较远时可高速移动，当距离比较接近时扳动操纵杆降低速度，便于观察和控制。

参数说明:

axis：轴编号，取值范围0-1。

返回值说明:表4－1

使用举例:

```
int SMCS1120return;
SMCS1120return = SMCS1120_velocity_move (1); //命令第 1 轴进入速度模式
SMCS1120_change_speed(50000);                //改变速度为 50kpps
//应用软件代码
SMCS1120_change_speed(-250000);              //改变速度为反方向运动，速度为 250kpps
if(SMCS1120return !=0) {出错处理}
```

相关函数：SMCS1120_set_move_speed();SMCS1120_get_status();SMCS1120_change_speed()

4.3.12 int SMCS1120_change_speed(unsigned int axis, long speed)

函数功能:当指定轴处于速度模式时，用来改变该轴运行速度以及运行方向，控制器将按照SMCS1120_set_move_speed()函数设定的加速度/减速度改变当前速度至该函数设定的速度。

若该函数设定的速度与当前速度相反，速度会减速到0并改变方向信号以后再加速至该函数设定的速度；该函数可在运动中连续调用，以达到连续改变运行速度和方向的目的。

当该轴触发限位开关或者超过软件限位位置设置时，控制器将减速为0，并退出速度模式，如需再次运行并改变速度，需调用SMCS1120_velocity_move()函数再次进入速度模式。

参数说明:

axis：轴编号，取值范围0-1。

speed：新的速度值，符号为正表示朝正方向运动，符号为负表示朝负方向运动，为0值将使该轴速度为0并保持在速度模式，再次将速度设置为非0值时，该轴又开始运动。speed与SMCS1120_set_move_speed(unsigned int axis,unsigned long max_speed,double acc_time)所设定的max_speed关系为：

$$\max\_speed \geq speed \geq \frac{\max\_speed}{640} \quad \text{.....speed为正整数}$$

当speed大于max_speed时，函数将出错返回，速度保持原来速度；当speed小于max_speed/640时，速度为0。

返回值说明:表4－1

使用举例:

```
int SMCS1120return;
```

```
SMCS1120return = SMCS1120_velocity_move (0); //命令第 0 轴进入速度模式
SMCS1120_change_speed(50000);                //改变速度为 50kpps
//应用软件代码
SMCS1120_change_speed(-250000);              //改变速度为反方向运动，速度为 250kpps
if(SMCS1120return !=0) {出错处理}
```

相关函数：

SMCS1120_velocity_move();SMCS1120_set_move_speed();SMCS1120_get_status();

4.3.13 int SMCS1120_stop(unsigned int axis,unsigned int mode);

函数功能:停止指定轴任何模式的运动，处于停止状态时调用该函数无效；该函数常用于终止点位运动、速度模式运动、回原点运动；该函数调用之后轴停止时控制器还将轴状态寄存器bit10（运动完成标志）置1。

参数说明:

axis：轴编号，取值范围0-1。

mode：停止方式：0－以SMCS1120_set_move_speed()函数设定的加速度减速停止运动；1－运动控制器接收到该命令后立即停止运动，没有任何减速过程。在运动速度比较高时应谨慎使用，防止立即停止时的冲击引起机构磨损；如需在异常情况下紧急停车，须设计紧急停止控制装置切断电机驱动器电源。

返回值说明:表4－1

使用举例:

```
int SMCS1120return;
SMCS1120return = SMCS1120_stop (0,0);  //无论第 0 轴处于什么状态，使第 0 轴减速停止进入停止模式
if(SMCS1120return !=0) {出错处理}
```

相关函数：SMCS1120_set_move_speed();

4.3.14 int SMCS1120_get_position(unsigned int axis,long *pos);

函数功能:读取指定轴指令位置寄存器；

参数说明:

axis：轴编号，取值范围0-1。

pos：指向保存位置的变量的指针。

返回值说明:表4－1

使用举例:

```
int SMCS1120return;
long x_pos;
SMCS1120return = SMCS1120_get_position (0, &x_pos); //读取第 0 轴指令位置寄存器，将变量 x_pos 的地
                                                    址传递给 API 函数以接收该函数传回的位置值
if(SMCS1120return !=0) {出错处理}
```

相关函数：SMCS1120_get_resolution ();

4.3.15 int SMCS1120_set_position(unsigned int axis,long pos);

函数功能:重设指定轴的指令寄存器，该操作只允许在停止状态下进行。

参数说明:

axis：轴编号，取值范围0-1。

pos：新的位置值。

返回值说明:表4－1

使用举例:

```
int SMCS1120return;
SMCS1120return = SMCS1120_set_position (0,0); //设置第0轴位置为0
if(SMCS1120return !=0) {出错处理}
```

相关函数： SMCS1120_home();

4.3.16 int SMCS1120_set_soft_limit(unsigned int axis,long pos_limit,long neg_limit);

函数功能:设置指定轴正负方向软件限位值。当限位值设定为0时，限位功能被禁止。比如第1轴pos_limit设置为零，则第1轴正方向的软件限位功能失效，如果确实需要限制在0，pos_limit可设置为1。注意，正的限位位置必须大于负的限位位置，否则将导致该轴无法运动。该设置对点位运动和速度模式运动有效，对回原点运动无效。因为回原点成功后，控制器会把原点开关所在位置时的指令位置寄存器设置为0，因此软件限位位置是相对于原点开关的，除非使用了SMCS1120_set_position()函数重设了指令位置寄存器。

为什么要设置软件位置限制功能？

1. 设置了软件位置限位功能，在点位运动时设置了超范围的目标点，控制器将拒绝执行并报错，从一定程度提高机器安全性。
2. 若限位开关可触发的行程比较小，机器触发限位开关后往往会减速停止，而减速停止的移动距离便已超过限位开关可触发区，限位开关又处于未触发状态，这样便造成了限位开关未触发的假像，这在使用速度模式运行时不够安全，设置了软件位置限位功能后，软件限位和限位开关一起起作用，可消除限位开关行程比较小的缺点。开关安装方式和机械遮挡方式设计不合理或者受条件限制会出现可触发行程小的现象。
3. 使用软件限位可方便的修改限位点。

参数说明:

axis：轴号，0－1。

pos_limit：正方向软件限位位置

neg_limit：负方向软件限位位置

返回值说明:表4－1

使用举例:

```
int SMCS1120return;
SMCS1120return = SMCS1120_set_soft_limit (1, 120000, -1000);      //将正方向的软件限位位置设置在距离原点开关120000处,负方向的设置在距离原点开关负方向1000的地方。
if(SMCS1120return !=0) {出错处理}
```

相关函数：SMCS1120_velocity_move();SMCS1120_pmove();SMCS1120_get_soft_limit();

4.3.17 int SMCS1120_get_soft_limit(unsigned int axis,long *pos_limit,long *neg_limit);

函数功能:读取指定轴软件限位位置设定值。

参数说明:

axis：轴号，0－1。

pos_limit：正方向软件限位位置

neg_limit：负方向软件限位位置

返回值说明:表4－1

使用举例:

```
int SMCS1120return;
long pos_limit，neg_limit;
SMCS1120return = SMCS1120_get_soft_limit (1, &pos_limit, &neg_limit);    //
if(SMCS1120return !=0) {出错处理}
```

相关函数：SMCS1120_velocity_move();SMCS1120_pmove();SMCS1120_get_soft_limit();

4.3.18 int SMCS1120_get_input(unsigned int *input);

函数功能:读取通用输入口状态;

参数说明:

input：指向保存输入口状态的变量的指针。Bit0为通用输入1的状态，bit5为通用输入6的状态，bit6－bit15保留，读为0。

返回值说明:表4－1

使用举例:

```
int SMCS1120return;
unsigned int input;
SMCS1120return = SMCS1120_get_input (&input); //读取通用输入口状态
if(SMCS1120return !=0) {出错处理}
```

4.3.19 int SMCS1120_set_output(unsigned int status);

函数功能:设置通用输出口状态。

参数说明:

status：新的输出状态值。Bit0为通用输出1的状态，bit3为通用输入4的状态，bi4－bit15保留，读为0。

返回值说明:表4－1

使用举例:

```
int SMCS1120return;
SMCS1120return = SMCS1120_set_output (0x000e);  //使输出口1输出低电平，输出口2－4输出高电平
if(SMCS1120return !=0) {出错处理}
```

相关函数：SMCS1120_get_output（）

4.3.20 int SMCS1120_get_output(unsigned int *status);

函数功能:读取通用输出口状态。

参数说明:

status：新的输出状态值。

返回值说明:表4－1

使用举例:

```
int SMCS1120return;
unsigned int output;
SMCS1120return = SMCS1120_get_output (1, &output);  //读取光电隔离输出口状态
```

if(SMCS1120return !=0) {出错处理}

相关函数：SMCS1120_set_output（）

4.3.21 int SMCS1120_reset_timeouts(void);

函数功能:重新建立通讯连接并重置通讯超时标志。SMCS1120超时时间为2秒，当由于通讯电缆拔除等原因造成通讯中断后（任一函数返回值为8说明通讯已中断），需调用该函数清除超时标志。清除超时标志后其它函数调用才有效，否则调用其它函数后立即返回8。

参数说明：无

返回值说明:返回值固定为0，无意义。

使用举例:

```
int SMCS1120return;
 SMCS1120return = SMCS1120_reset_timeouts();
```

4.3.22 int SMCS1120_set_light(unsigned int *light_value,unsigned int time,unsigned int mode,unsigned int on_off);

函数功能:设置光源各种颜色亮度、发光模式、闪光时间以及开关光源；光源一旦设定为频闪模式，发光盘的亮与灭便由控制器自行控制，不需要软件干预，但软件可以切换到连续照明模式。另请参见3.2.3章节。

参数说明：light_value：依R／G／B／W顺序保存亮度的数组首地址，亮度值取值范围为0－31。

Time：闪光时间，单位为毫秒，取值范围为1－5000。

Mode：工作模式：0－频闪模式，1－连续照明模式。

On_off:0－关闭光源，1－开启光源；只有在工作模式为1时才允许开启光源，否则函数不予执行并返回非0值。

返回值说明: 表4－1

使用举例:

```
int SMCS1120return;
unsigned int light[4]={19,21,20,0};    //绿色led发光效率低于红色led,将绿色亮度设置高一些以保证
                                       //合成光的色温,无白色光
SMCS1120return = SMCS1120_set_light(light,100,1,1); //设置为连续发光模式，以观察光源亮度和色温
SMCS1120return = SMCS1120_set_light(light,100,1,0);   //连续发光模式，关闭光源
SMCS1120return = SMCS1120_set_light(light,100,0,0);   //切换到频闪模式
```

4.3.22 int SMCS1120_get_encoder(unsigned int axis,long *pos);

函数功能:读取指定轴编码器（或光栅尺）计数值。

参数说明：axis：轴号：0－1

Pos：指向保存读取到的计数值的变量的指针

返回值说明: 表4－1

使用举例:

```
int SMCS1120return;
long pos;
```

SMCS1120return = SMCS1120_get_encoder (0,&pos); //读取第0轴编码器计数值

相关函数：SMCS1120_set_encoder（） ，SMCS1120_get_position（）

4.3.22 int SMCS1120_set_encoder(unsigned int axis,long pos);

函数功能:设置指定轴编码器（或光栅尺）计数值。

参数说明：axis：轴号：0－1

Pos：指向保存读取到的计数值的变量的指针

返回值说明：表4－1

使用举例:

```
int SMCS1120return;
long pos＝0;
SMCS1120return = SMCS1120_set_encoder (0,pos); //j将第0轴编码器计数值清0
```

相关函数：SMCS1120_get_encoder（），SMCS1120_set_position（）