存储卡

S7-200 SMART CPU 支持商用MicroSD卡（支持容量为 4G、8G、16G）：可用于程序传输，CPU固件更新，恢复 CPU 出厂设置。

打开 CPU本体数字量输出点上方的端子盖，可以看到右侧有一卡槽，将MicroSD卡缺口向里插入，如图 1所示：

图 1. 插入Micro SD 卡

使用 MicroSD 卡传送程序

➪ 制作程序传输卡：

步骤一：
用户在 CPU **上电且停止**状态下插入存储卡；

注意：用户也可以在 CPU断电状态下插入一张空的存储卡然后再将 CPU 上电，但是需要注意的是存储卡必须确实是**空的**而不是旧的固件更新卡或者是程序传输卡。否则可能会更改 CPU 固件或者是内部存储的项目。

步骤二：
下载源程序到 CPU（如果 CPU中已经存在源程序则不需此步）；

步骤三：
在 Micro/WIN SMART 中，点击“ PLC”->“ 编程存储卡 ”，打开“ 编程存储卡 ”对话框，选择需要被拷贝到存储卡上的块，点击“ 编程 ”按钮，如图 2所示；

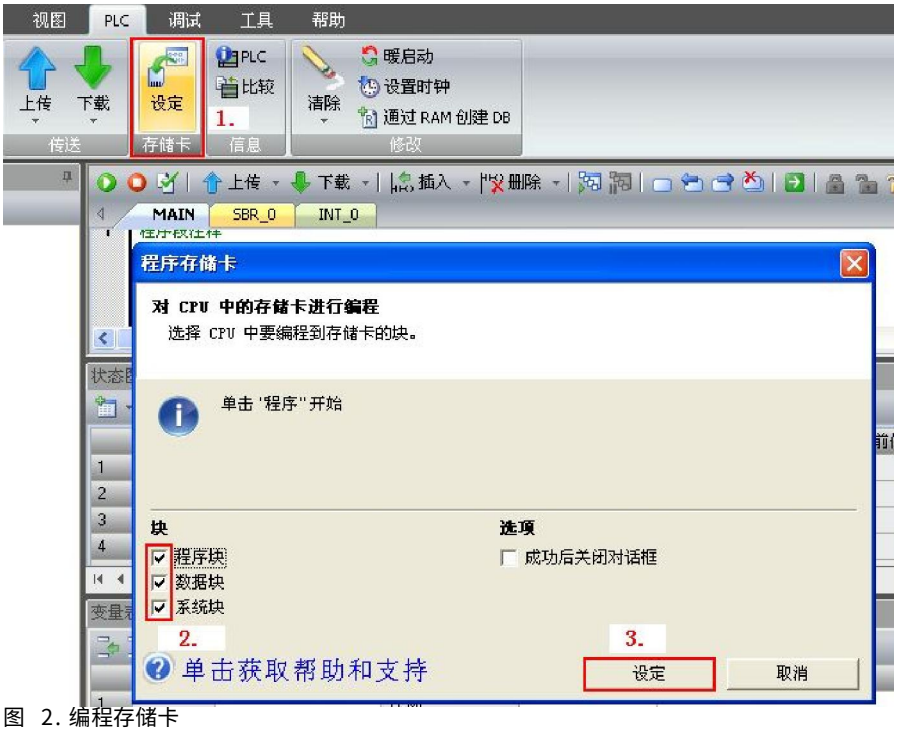


图 2. 编程存储卡

步骤四：
当 Micro/WIN SMART 显示编程操作成功执行时（“ 编程存储卡 ”对话框显示编程成功，点击“ 关闭 ”）如图 3所示，从 CPU止取下存储卡。

图 3. 编程存储卡成功

注意：在 Micro/WIN SMART 中进行“编程存储卡”操作时，是将 CPU中存储的程序拷贝至 CPU，而非软件中打开的程序。所以必须先将程序下载到 CPU中，才能执行“编程存储卡”操作。

使用已制作好的程序传输卡拷贝项目到另一个 CPU:

步骤一：
在 CPU 断电状态下插入存储卡；

步骤二：
给 CPU 上电，CPU 会自动识别存储卡为程序传输卡并且自动将其中的内容复制到 CPU 内部存储器，传输过程中 RUN 指示灯和 STOP 指示灯以 2 HZ 的频率交替点亮；

步骤三：
当 CPU 只有 STOP 灯开始闪烁，表示“程序传送”操作成功，则从 CPU 上取下存储卡。

使用 MicroSD 卡更新固件

步骤一：
用普通读卡器将固件文件拷贝到一个**空的** MicroSD 卡中。固件文件包括：工作文件 “S7_JOB.S7S” 和文件夹“FWUPDATE.S7S”（内含固件，命名方式：CPU订货号-固件版本号，扩展名为 .upd）。
使用记事本打开文件 “S7_JOB.S7S”，应只包含字符串“FWUPDATE”。

图 4.固件文件夹

步骤二：
在 CPU 断电状态下将包含固件文件的存储卡插入 CPU；

步骤三：
给 CPU 上电，CPU 会自动识别存储卡为固件更新卡并且自动更新 CPU 固件。更新过程中 RUN 指示灯和 STOP 指示灯以 2 HZ 的频率交替点亮。

步骤四：
当 CPU 只有 STOP 灯开始闪烁，表示“固件更新”操作成功，从 CPU 上取下存储卡。

步骤五：

给 CPU 重新上电，在 Micro/WIN SMART 中查看 CPU固件版本，如图 5所示。

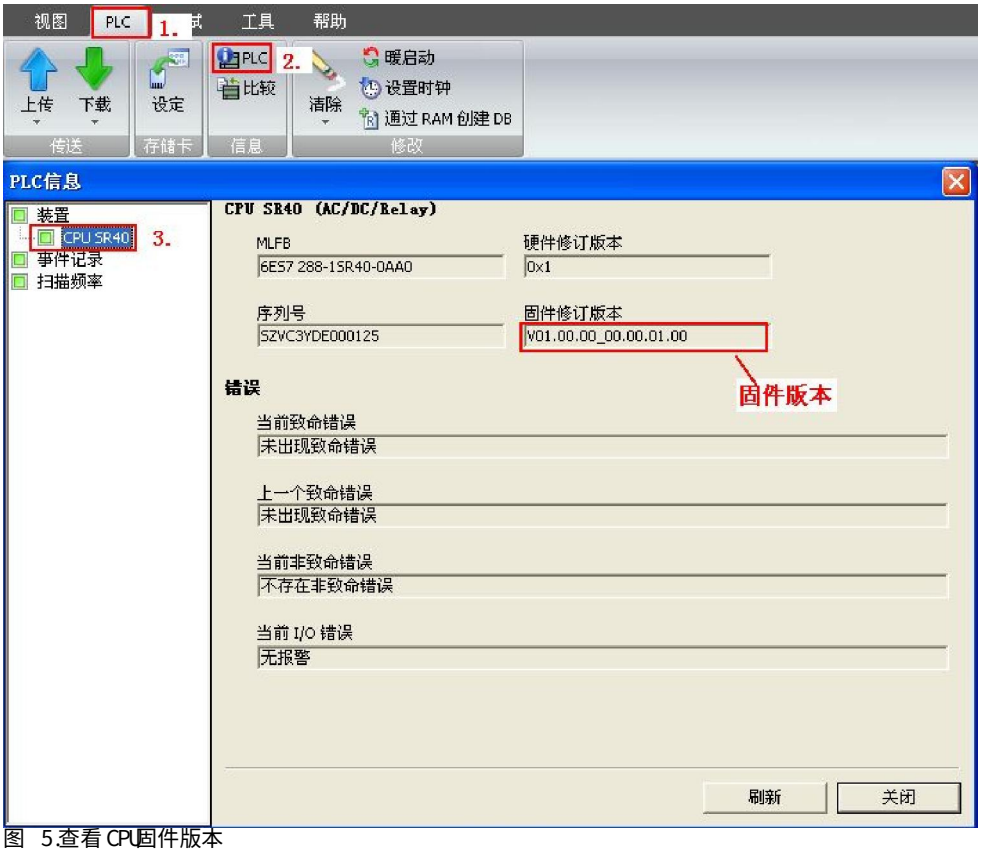


图 5.查看 CPU固件版本

使用 MicroSD卡恢复 CPU出厂设置

步骤一：
用普通读卡器将恢复出厂设置文件拷贝到一个**空的** MicroSD卡中。恢复出厂设置文件为文本文件 “ S7_JOB.S7S” 。 使用记事本打开文本文件 “ S7_JOB.S7S”，应包含字符串 “ **RESET_TO_FACTORY**”。

步骤二：
在 CPU断电状态下插入 MicroSD卡，给 CPU上电， CPU会自动识别存储卡为恢复出厂设置卡并且自动恢复 CPU出厂设置。恢复出厂设置过程中， RUN指示灯和 STOP指示灯以 2 HZ的频率交替点亮。

步骤三：
当 CPU只有 STOP灯开始闪烁，表示“ 恢复出厂设置 ”操作成功，从 CPU上取下存储卡。

步骤四：
“ 恢复出厂设置 ”操作包括以下几项操作：将 CPU IP地址恢复为出厂默认设置，清空 CPU程序块、数据库和系统块。

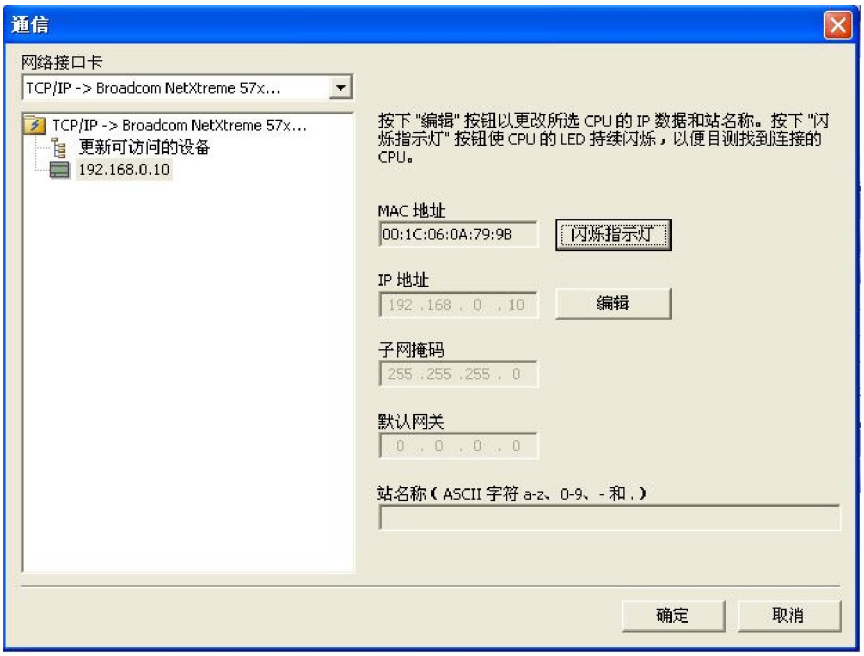


图 6.恢复出厂设置操作后 CPU 的 IP 地址

注意：恢复出厂设置不会更改 CPU的固件版本， CPU 固件版本将保持为恢复出厂设置操作之前的固件版本。

常问问题

如何将固件更新卡转换为程序传输卡？

在 CPU**止电且停止**状态下插入固件更新卡，按照制作程序传输卡的步骤进行操作，即可得到一张程序传输卡，但需注意的是原卡中的固件更新文件会被删除。

如何将程序传输卡转换为固件更新卡？

使用读卡器将固件更新文件拷贝至程序传输卡，删除原卡中的程序传输文件夹 “ SIMATIC.S7S” ，并将工作文件“ S7_JOB.S7S”中的字符串更换为“ FWUPDATE” 。

注意：建议用户不要在同一张存储卡上同时存储程序传输文件和固件更新文件。

在固件更新期间 CPU 本体上的 LED 指示灯如何显示？如果用户在固件更新期间取出存储卡， LED 指示灯如何显示？

CPU 本体上的 LED 灯在使用存储卡时的显示状态及原因：

状态一：如果用户在 CPU 运行状态下插入存储卡， CPU 会转入停止状态。无论存储卡中存储了什么这种行为都会发生。

状态二： STOP 灯以 2 HZ 的频率闪烁
a. CPU 上电后，通过存储卡的“ 固件更新 ”操作被成功执行，并且 CPU 此刻需要重新上电或者重启；
b. CPU 上电后，通过存储卡的“ 程序传输 ”操作被成功执行；
c. CPU 上电后，通过存储卡的“ 恢复出厂设置 ”操作被成功执行；
d. CPU 上电后，检测到空存储卡或者是未知卡件，无任何操作被执行；
e. 在上电状态下插入一张存储卡。

状态三： STOP 灯和 ERROR 灯同时以 2 HZ 频率闪烁
a. CPU 上电后，试图通过存储卡进行“ 固件更新 ”操作并且没有成功；
b. CPU 上电后，试图通过存储卡进行“ 程序传输 ”操作并且没有成功；
c. CPU 上电后，试图通过存储卡进行“ 恢复出厂设置 ”操作并且没有成功；
d. 当“ 固件更新 ”操作和“ 程序传输 ”操作正在进行中，用户取出存储卡。

注意：对于产生“ 状态二 ”的情况 b，c，d，e 以及产生“ 状态三 ”的情况 b，c，当取出存储卡时 LED 灯的闪烁状况会停止；对于产生“ 状态二 ”的情况 a 和产生“ 状态三 ”的情况 a，只有当 CPU 重新上电或者重启时 LED 灯才会停止闪烁。

状态四：繁忙 LED 模式（RUN 指示灯和 STOP 指示灯以 2 HZ 的频率交替点亮）
“ 固件更新 ”操作和“ 程序传输 ”操作正在进行中，当该操作停止时， LED 灯显示状态转为“ 状态二 ”或者“ 状态三 ”。

如果存储卡中同时包含程序文件和固件更新文件，哪种更新操作会被优先执行？是否有预定义的优先级？是否可以在一张固件更新存储卡上进行程序传输卡的制作？

存储卡上的固件更新文件位于一个名为 “ FWUPDATE.S7S” 的文件夹中，程序传输文件位于一个名为 “ SIMATIC.S7S” 的文件夹中。所以理论上这两个文件夹可以同时存储在存储卡上。

然而，存储卡内只有一个命名为 “ S7_JOB.S7S” 工作文件，这个文件决定了 CPU 将存储卡视为固件更新卡还是程序传输卡。如果工

作文件包含字符串 “ FWUPDATE”，那么这张存储卡便是固件更新卡。若工作文件包含字符串 “ TO_ILM”，那么这张存储卡便是程序传输卡。

此外，如果 CPU 被指示创建程序传输卡，固件实际上会在复制项目文件到存储卡上之前先删除存储卡上的以下内容：
——工作文件；
——“ FWUPDATE.S7S” 文件夹和其中的所有内容；
——“ S7_JOB.S7S” 文件夹和其中所有的内容。

如果一张程序传输卡中留有固件更新文件是存在潜在风险的。用户可能会在一张程序传输卡上拷贝固件更新文件，如果工作文件包含字符串 “ FWUPDATE” ，则 CPU会将存储卡视为固件更新卡。

S7-200 SMART 实时时钟

S7-200 SMART的硬件实时时钟可以提供年、月、日、时、分、秒的日期 /时间数据。

CPU CR40 AC/DC/Relay 没有内置的实时时钟， CPU SR20、 CPU SR40、 CPU ST40、 CPU SR60、 CPU ST60支持内置的实时时钟， CPU断电状态下可保持 7天。

S7-200 SMART CPU SR20的时钟精度是± 120 秒 / 月， CPU SR40、 CPU ST40、 CPU SR60、 CPU ST60的时钟精度是 120 秒 / 月。

S7-200 SMART CPU 靠内置超级电容为实时时钟提供电源缓冲，保持时间为典型值 7天，最小值 6天。缓冲电源放电完毕后，再次上电后时钟将停止在缺省值，并不开始走动。

注意：因为 **CPU CR40 无内置超级电容**，所以实时时钟无电源缓冲，尽管用户可以使用 READ_RTC 和 SET_RTC 指令设置日期 /时间数据，但是当 CPU CR40 断电并再次上电时，这些**日期 /时间数据会丢失**，上电后日期时间数据会被初始化为 **2000年 1月 1日**。

为了提高运算效率，应当避免每个程序周期都读取实时时钟。实际上可读取的最小时间单位是 1秒，可每秒读取一次（使用 SM0.5上升沿触发读取指令）。

使用程序读取的实时时钟数据为 BCD格式，可在状态表中使用十六进制格式查看。

要设置日期、时间值，使之开始走动，可以：

- 用编程软件（Micro/WIN SMART）的菜单命令 PLC > SET Clock，通过与 CPU的在线连接设置，完成后时钟开始走动
- 编用户程序使用 Set_RTC（设置时钟）指令设置

Micro/WIN SMART 可以通过任何编程连接实现实时时钟的设置。

用 Micro/WIN SMART 设置时钟

通过编程软件 Micro/WIN SMART 设置 CPU 的时钟，必须先建立编程通信连接。
在 Micro/WIN SMART 菜单中选择 “ PLC “ - >” 设置实时时钟 ”，打开 “ PLC 时钟操作 ” 对话框：

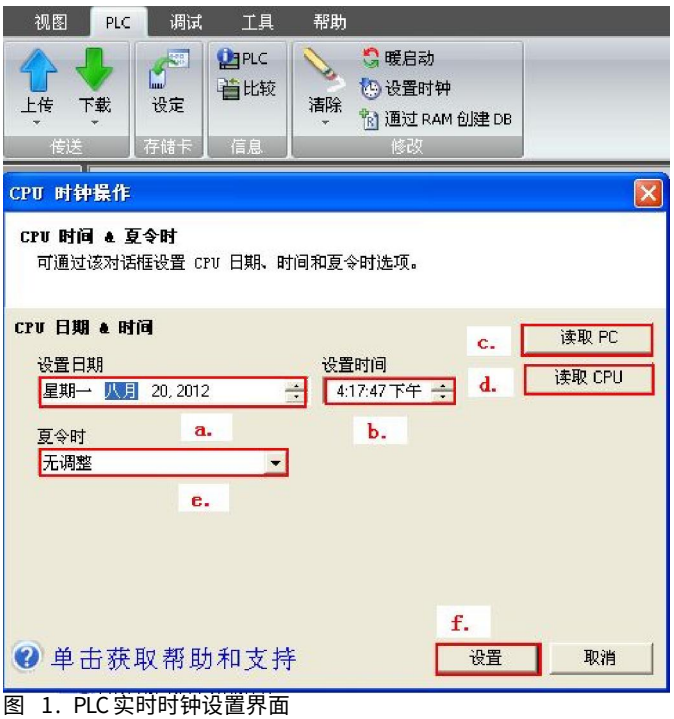


图 1. PLC 实时时钟设置界面

图中：

a. 设置日期：选择需要修改的数据字段，直接输入数字，或者使用输入框右侧的上下按钮调整
b. 设置时间：选择需要修改的数据字段，直接输入数字，或者使用输入框右侧的上下按钮调整
c. 读取 PC 时钟：按此按钮可以读取安装 Micro/WIN SMART 的 PC 机的本机时间
d. 读取 PLC 时钟：按此按钮读取 PLC 内部的实时时钟数据
e. 根据需要选择夏时制调整选项
f. 按“设置”按钮，将上面的时钟日期数据写入 PLC

时钟读写指令缓冲区格式

Read_RTC（读时钟）和 Set_RTC（设置时钟）指令靠数据缓冲区在用户程序与硬件芯片间交换数据，它们的缓冲区格式相同。

表 1. 时钟缓冲区

地址偏移	T	T+1	T+2	T+3	T+4	T+5	T+6	T+7
数据内容	年	月	日	小时	分钟	秒	0	星期
数值范围 BCD（16进制）	00-99	01-12	01-31	00-23	00-59	00-59	0	0-7*

* 1 = 星期日，7 = 星期六，0 = 表示禁止计星期

T 就是缓冲区的起始字节地址，可以由用户自由设置（在 CPU 允许的 V 存储区范围内）。如果设置 T 为 VB100，那么读取时钟后，“年”的信息就会保存在 VB100 中，“月”保存在 VB101 中。

常问问题

写时钟指令（SET_RTC）为何不能正常改写时钟内容？

1. 写时钟指令需要严格按照 8 个字节的时钟缓冲区格式，设置相应的数据单元，任何不合格的数据都可能造成不能写入的现象。注意数据的格式必须是 BCD 格式，可以说是将 10 进制数换成 16 进制表示，如 16#59（59H）就是 59（秒/分等）。

   ⚠ 执行写时钟指令要保证缓冲区所有字节都包含合法数据；仅修改某些数据时，其他字节不能包含非法数值，否则会发生错误。

2. 写时钟指令必须使用一次性的脉冲（沿）触发条件，不能持续激活写时钟指令。

读写的时钟数据如何在十进制（整数）和 BCD 数据之间转换？

要计算时间需要将 BCD 格式的时钟数据与十进制的数据之间的转换，使用相应的转换指令就可以实现。

具体转换指令参见例程

参考例程

 S7-200 SMART 时间设置与读取

⚠ **注意：**此指令库/程序的作者和拥有者对于该软件的功能性和兼容性不负任何责任。使用该软件的风险完全由用户自行承担。由于它是免费的，所以不提供任何担保，错误纠正和热线支持，用户不必为此联系西门子技术支持与服务部门。

CPU 本体集成高速计数器

S7-200 SMART CPU 具有集成的、硬件高速计数器。

CPU SR20、CPU SR40、CPU ST40、CPU SR60 和 CPU ST60 可以使用 4 个 60kHz 单相高速计数器或 2 个 40kHz 的两相高速计数器，而 CPU CR40 可以使用 4 个 30kHz 单相高速计数器或 2 个 20kHz 的两相高速计数器。

计数器共有四种基本类型：带有内部方向控制的单相计数器，带有外部方向控制的单相计数器，带有两个时钟输入的双相计数器和 A/B 相正交计数器。

表 1. 高速计数器的模式及输入点：

模式	描述	输入点		
	HSC0	I0.0	I0.1	I0.4
	HSC1	I0.1		
	HSC2	I0.2	I0.3	I1.5
	HSC3	I0.3		
0	带有内部方向控制的单相计数器	时钟		
1		时钟		复位

3	带有外部方向控制的单相计数器	时钟	方向	
4		时钟	方向	复位
6	带有增减计数时钟的双相计数器	增时钟	减时钟	
7		增时钟	减时钟	复位
9	A/B相正交计数器	时钟 A	时钟 B	
10		时钟 A	时钟 B	复位

表 2. 高速计数器的寻址

高速计数器号	HSC0	HSC1	HSC2	HSC3
新当前值（新 CV）	SMD38	SMD48	SMD58	SMD138
新预置值（新 PV）	SMD42	SMD52	SMD62	SMD142
当前计数值（仅读出）	HC0	HC1	HC2	HC3

高速计数器的具体编程及相关的中断和其它参数，请参见《S7-200 SMART 系统手册》，上面有详细的阐述及例程。

STEP 7-Micro/WIN SMART提供了一个方便实用的高速计数器指令编程向导，用户可以简单快速地配置自己的高速计数器功能。

高速输入降噪

要正确操作高速计数器，可能需要执行以下一项或两项操作：
● 调整 HSC 通道所用输入通道的“系统块”数字量输入滤波时间。 在 S7-200 SMART CPU 中。 在 HSC 通道对脉冲进行计数前应用输入滤波。 这意味着，如果 HSC 输入脉冲以输入滤波过滤掉的速率发生，则 HSC 不会在输入上检测到任何脉冲。 请务必将 HSC 的每路输入的滤波时间组态为允许以应用需要的速率进行计数的值。 包括方向和复位输入。 下表显示可检测到的每种输入滤波组态的最大输入频率。

表 3 输入滤波设置和可检测到的最大输入频率

输入滤波时间	可检测到的最大频率
0.2μs	60kHz （HSC 的最大值）
0.4μs	60kHz （HSC 的最大值）
0.8μs	60kHz （HSC 的最大值）
1.6μs	60kHz （HSC 的最大值）
3.2μs	60kHz （HSC 的最大值）
6.4μs	60kHz （HSC 的最大值）
12.8μs	39 kHz
0.2ms	2.5kHz
0.4ms	1.25kHz
0.8ms	625 Hz
1.6ms	312 Hz
3.2ms	156 Hz
6.4ms	78 Hz
12.8ms	39 Hz

输入逻辑电平有效电压范围

表 4. 输入逻辑电平有效电压范围

CPU型号	逻辑 1信号（最小）	逻辑 0信号（最大）
CPU SR20	2.5mA时 15VDC	1mA时 5VDC
CPU ST40	I0.0-I0.3：8mA时 4VDC 其他：2.5mA时 15VDC	I0.0-I0.3：1mA时 1VDC 其他：1mA时 5VDC
CPU SR40	I0.0-I0.3：8mA时 4VDC 其他：2.5mA时 15VDC	1mA时 5VDC
CPU CR40	2.5mA 时 15VDC	1mA时 5VDC
CPU ST60	I0.0-I0.3：8mA时 4VDC 其他：2.5mA时 15VDC	I0.0-I0.3：1mA时 1VDC 其他：1mA时 5VDC
CPU SR60	I0.0-I0.3：8mA时 4VDC 其他：2.5mA时 15VDC	1mA时 5VDC

●加入下拉电阻是为了使输入输出信号达到其逻辑电平有效范围。 如果设备的输出是集电极开路晶体管，则可能出现这种情况。 晶体管关闭时，没有任何因素将信号驱动为低电平状态。 信号将转换为低电平状态，但所需时间将取决于电路的输入电阻和电容。 这种情况可能导致脉冲丢失。 可通过将下拉电阻接到输入信号的方法避免这种情况，如下图所示。 由于 CPU 的输入电压是 24V，因此电阻的额定功率必须为高功率。 100 欧 5 瓦的电阻是一个合适的选择。

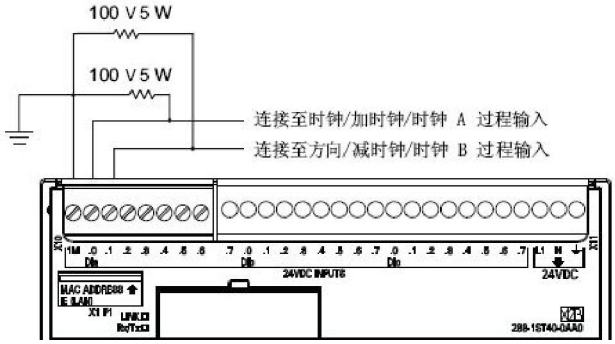


图 1. 集电极开路 HSC 输入驱动接线下拉电阻

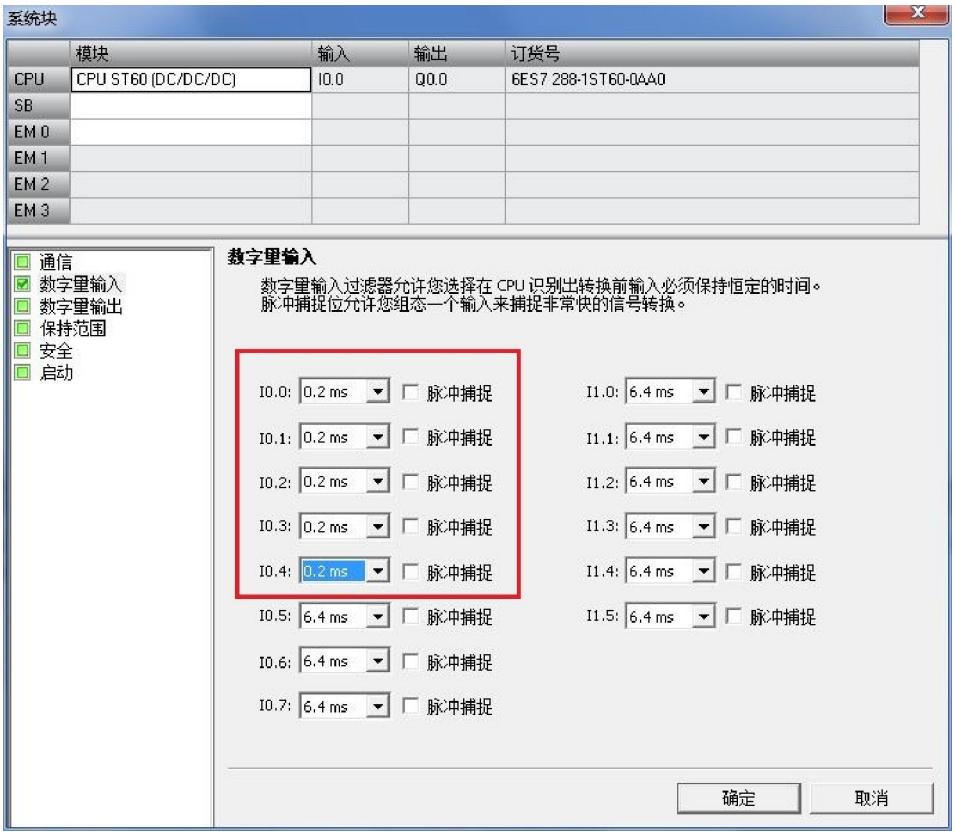
常问问题：

为什么 S7-200 SMART 高速计数器在低频率下计数正常，而在高频率下无法计数？

答：在 S7-200 中，HSC 旁路了输入滤波。而在 S7-200 SMART 中，HSC 没有旁路输入滤波，因此系统块中设置的输入滤波会影响 HSC，这样可以有助于一些用户避免高频干扰。默认的滤波时间 6.4μs 可以允许计数的最高频率是 78 HZ，如果要计数更高频率的信号，必须调整相应的滤波时间。 最大频率与滤波时间的对应关系如下表：

输入滤波时间	可检测到的最大频率
0.2μs	60kHz （HSC 的最大值）
0.4μs	60kHz （HSC 的最大值）
0.8μs	60kHz （HSC 的最大值）
1.6μs	60kHz （HSC 的最大值）
3.2μs	60kHz （HSC 的最大值）
6.4μs	60kHz （HSC 的最大值）
12.8μs	39 kHz
0.2ms	2.5kHz
0.4ms	1.25kHz
0.8ms	625 Hz
1.6ms	312 Hz
3.2ms	156 Hz
6.4ms	78 Hz
12.8ms	39 Hz

滤波时间的设置在 “ 系统块 ” 的 “ 数字量输入 ” 选项设置，如下图：



S7-200 SMART 高速计数器是否支持模式 12?

不支持。

高速计数器怎样占用输入点？

高速计数器根据被定义的工作模式，按需要占用 CPU上的数字量输入点。每一个计数器都按其工作模式占用固定的输入点。在某个模式下没有用到的输入点，仍然可以用作普通输入点；被计数器占用的输入点（如外部复位），在用户程序中仍然可以访问到。

为什么高速计数器不能正常工作？

在程序中要使用初次扫描存储器位 SM0.1来调用 HDEF指令，而且只能调用一次。如果用 SM0.0调用或者第二次执行 HDEF指令会引起运行错误，而且不能改变第一次执行 HDEF 指令时对计数器的设定。

对高速计数器如何寻址？为什么从 SMDx中读不出当前的计数值？

可以直接用 HC0；HC1；HC2；HC3对不同的高速计数器进行寻址读取当前值，也可以在状态表中输入上述地址直接监视高速计数器的当前值。 SMDx不存储当前值，参见上述表 2。

高速计数器的计数值是一个 32位的有符号整数。

高速计数器如何复位到 0?

- 选用带外部复位模式的高速计数器，当外部复位输入点信号有效时，高速计数器复位为 0
- 也可使用内部程序复位，即将高速计数器设定为可更新初始值，并将初始值设为 0，执行 HSC指令后，高数计数器即复位为 0

高速计数器的值在复位后是复位到初始值还是“ 0”值？

外部复位会将当前值复位到 0值而不是初始值；内部复位则将当前值复位到初始值（若初始值设为 ” 0“，则内部复位也是复位到 ” 0 “ 值）。如果你设定了可更新初始值，但在中断中未给初始值特殊寄存器赋新值，则在执行 HSC 指令后，它将按初始化时设定的初始值赋值。

为何给高速计数器赋初始值和预置值时后不起作用，或效果出乎意料？

高速计数器可以在初始化或者运行中更改设置，如初始值、预置值。其操作步骤应当是：

1. 设置控制字节的更新选项。需要更新哪个设置数据，就把控制字节中相应的控制位置位（设置为“ 1”）；不需要改变的设置，相应的控制位就不能设置
2. 然后将所需 的值送入初始值和预置值控制寄存器
3. 执行 HSC指令

高速计数器指令向导

在 Micro/WIN SMART 中的命令菜单中选择 Tools（工具）> Wizards（向导）中选择 High Speed Counter（高速计数器向导），也可以在项目树中选择 Wizards（向导）文件夹中的 High Speed Counter（高速计数器向导）按钮，如图 1所示。

图 1选择 HSC 向导

步骤一：选择 HSC 编号，如图 2所示。

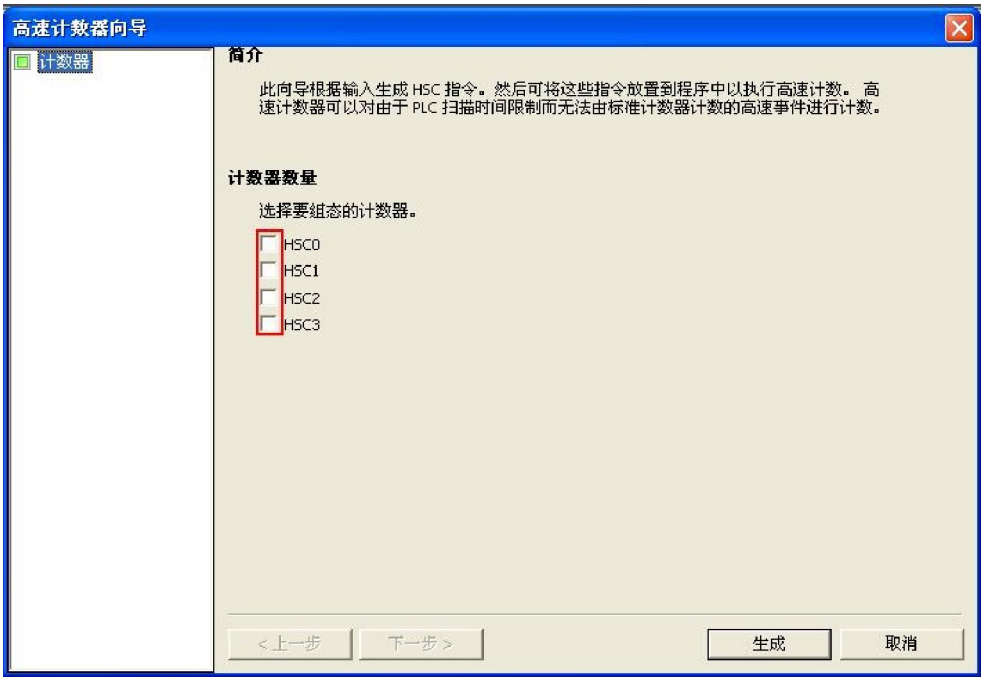


图 2选择计数器编号

步骤二：为计数器命名，在左侧树形目录中选择“ 高速计数器 ”，如图 3所示。

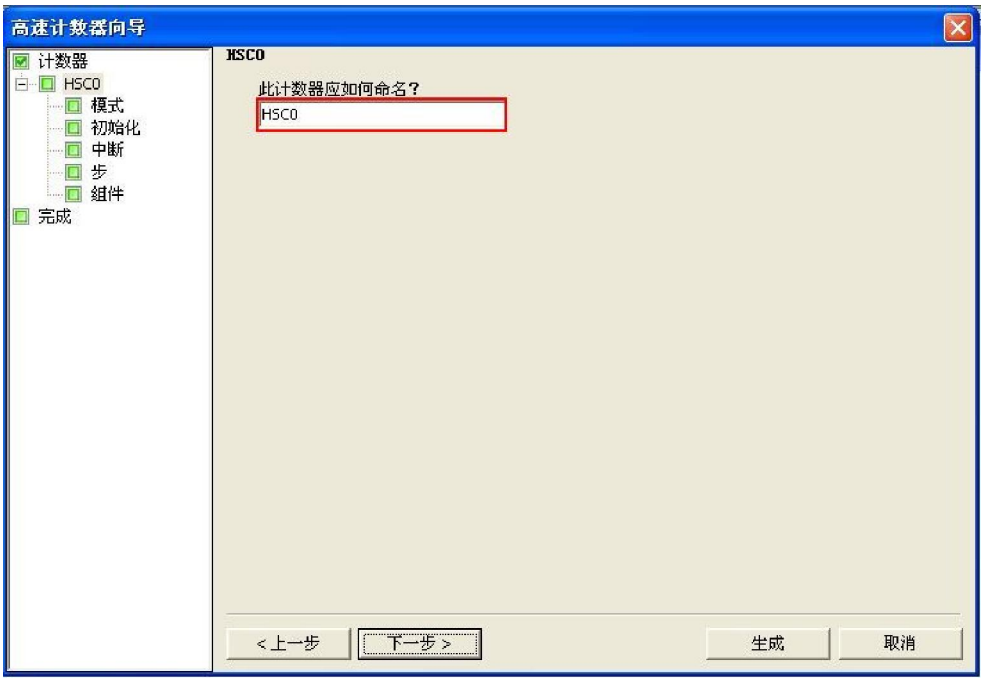


图 3.高速计数器命名

步骤三：选择计数器模式，详细信息请见“ 表 1.高速计数器的模式及输入点 ”。

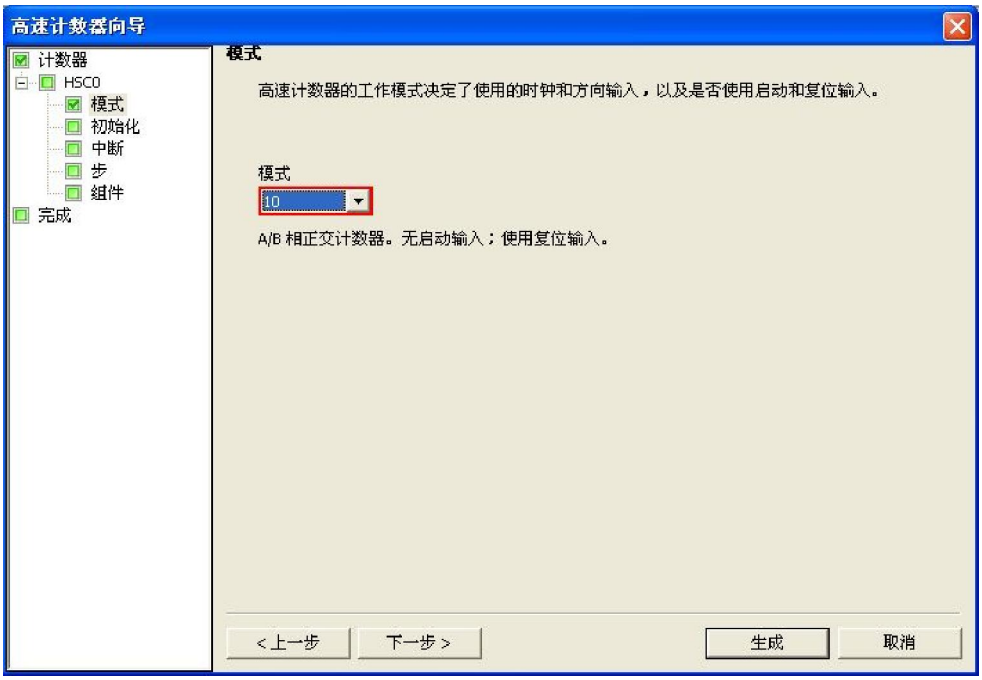


图 4.选择高速计数器模式

步骤四：配置初始化信息。

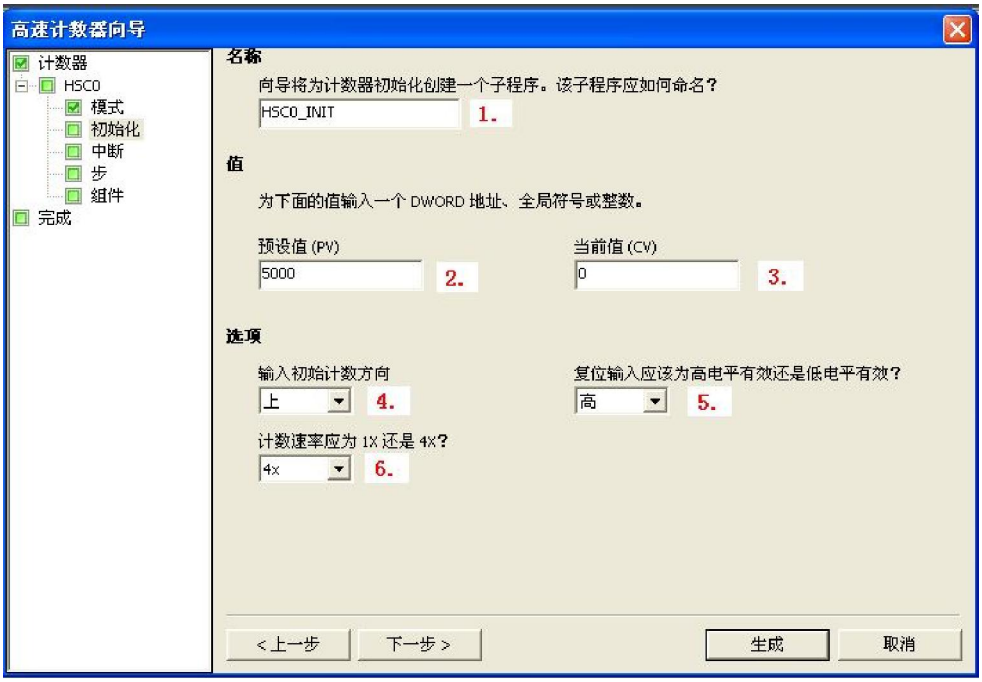
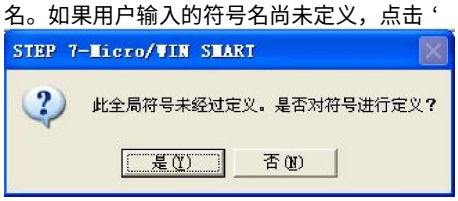


图 5. HSC 初始化选项

在上图中：

1. 为初始化子程序命名，或者使用默认名称。
2. 设置计数器预置值：可以为整数、双字地址或符号名：如 5000、VD100、PV_HC0。用户可使用全局符号表中双字整数对应的符号名。如果用户输入的符号名尚未定义，点击“ Generate （生成）”后会看到：



这个提示框显示：“ 这不是定义的全局符号。您希望定义符号吗 ”，点击“ 是 ”

填入地址和注释，注意：地址必须为双字地址， 注释可以不填。

3. 设置计数器初始值：可以为整数、双字地址或符号名： 5000、VD100、CV_HC0。
4. 初始化计数方向：增，减。
5. 对于带外部复位端的高速计数器，可以设定复位信号为高电平有效或者低电平有效。
6. 使用 A/B相正交计数器时，可以将计数频率设为 1倍速或 4倍速。使用非 A/B相正交计数器时，此项为虚。
7. S7-200 SMART 均不支持带外部启动端的高速计数器，因此此项为虚。

⚠ **注意：**所谓“ 高 /低电平有效 ”指的是在物理输入端子上的有效逻辑电平，即可以使 LED 灯点亮的电平。这取决于源型 /漏型输入接法，并非指实际电平的高、低。

步骤五：配置中断事件，如图 6所示。

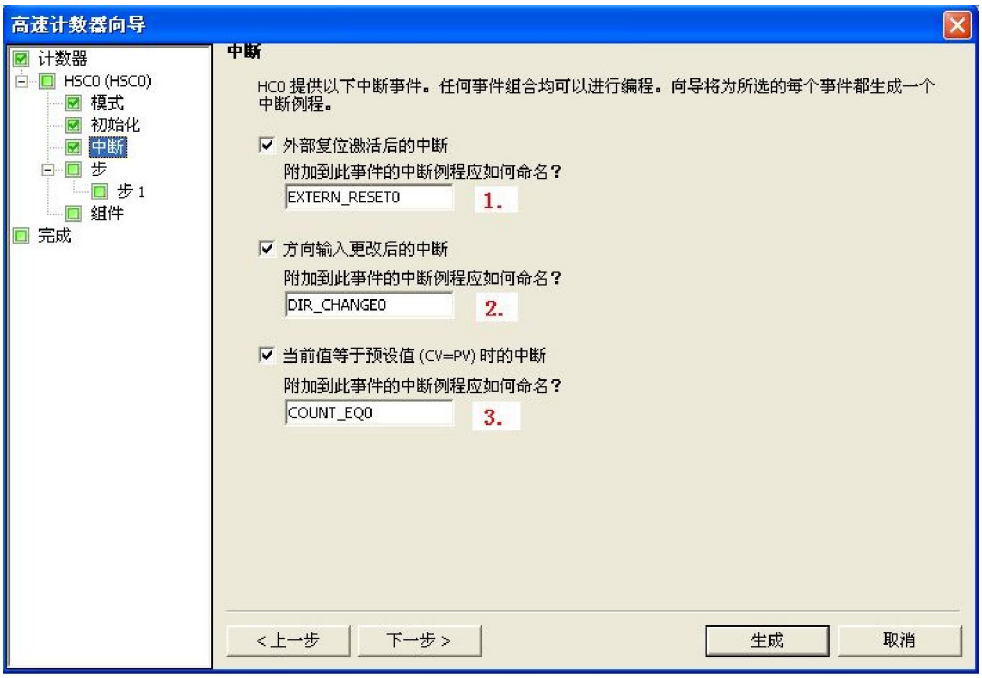


图 6 配置中断

如图 6所示，一个高速计数器最多可以有 3 个中断事件，在白色方框中填写中断服务程序名称或者使用默认名称：

⚠ 在这里配置的中断事件并非必须，系由用户根据自己的控制工艺要求选用。

1. 外部复位输入有效值时中断，如果使用的高速计数器模式不具有外部复位端，则此项为虚。
2. 方向控制输入状态改变时的中断，有以下 3 种情况会产生该中断：
   - 单项计数器的内部或外部方向控制位改变瞬间
   - 双相计数器增、减时钟交替的瞬间
   - A/B相脉冲相对相位（超前或滞后）改变时瞬间
3. 当前值等于预置值时产生的中断，通过向导，可以在该中断的服务程序中重新设置高速计数器的参数，如预置值、当前值。一个这样的过程称为‘一步‘。

步骤六：配置 HSC 步数，如图 7所示，最多可设置 10 步。

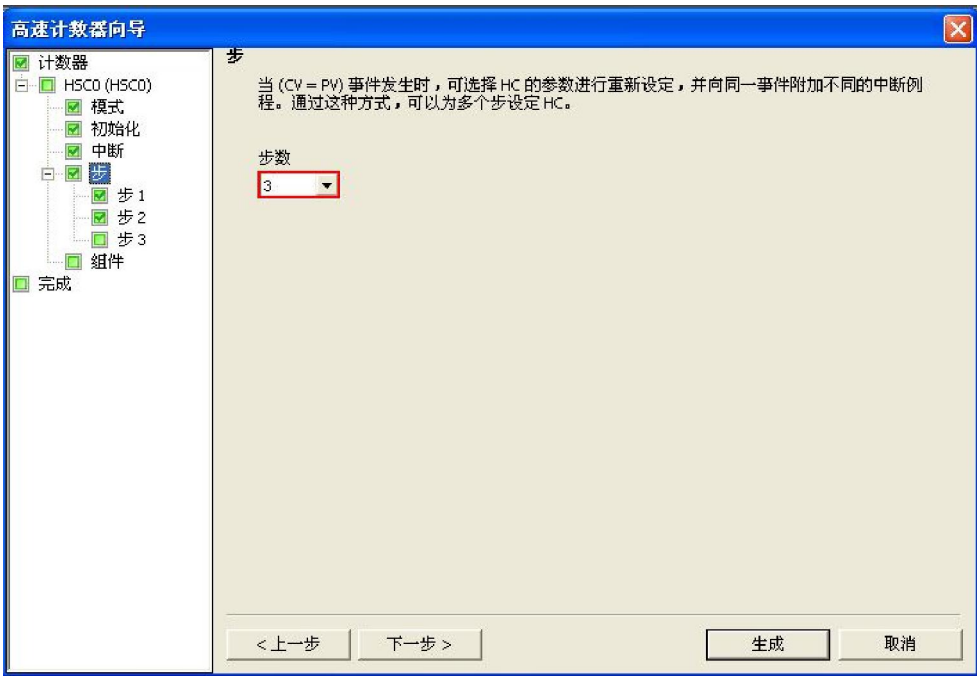


图 7. 配置 HSC 步数

步骤七：定义高速计数器每一步的操作，如图 8所示：

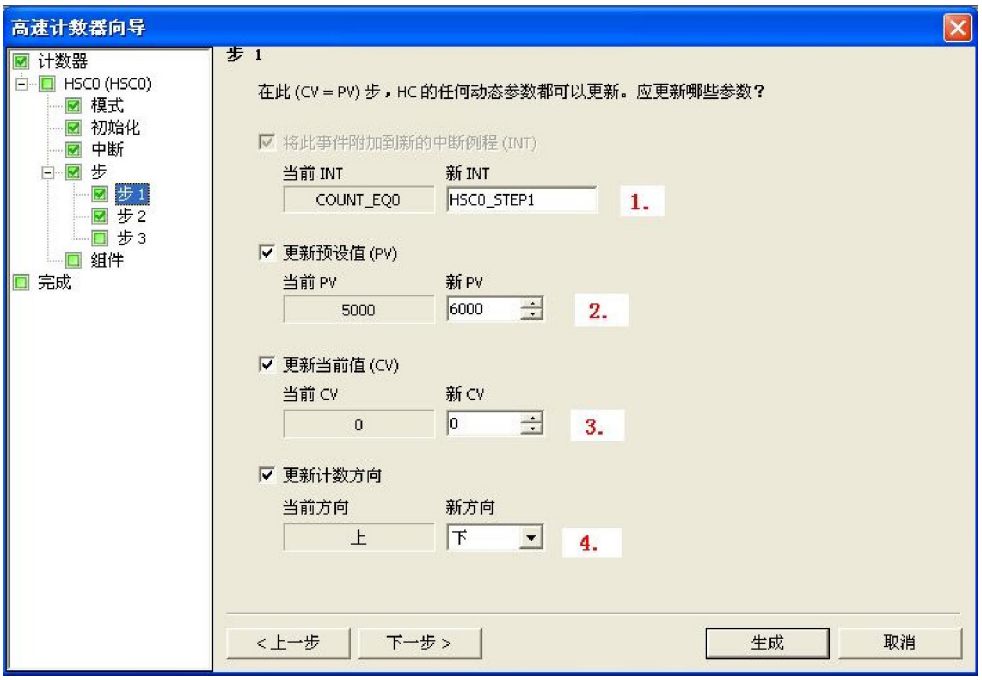


图 8. HSC 第一步

在这里配置的是当前值等于设定值中断的服务程序中的操作：

1. 向导会自动为当前值等于预置值匹配一个新的中断服务程序，用户可以对其重新命名，或者使用默认的名称。
2. 勾选后，用户在右侧输入新的预置值。
3. 勾选后，用户在右侧输入新的当前值。
4. 如果选用的高速计数器模式有内部方向控制位。
5. 使用相同的方法完成其余两步的设置

步骤八：完成向导，如图 9所示：

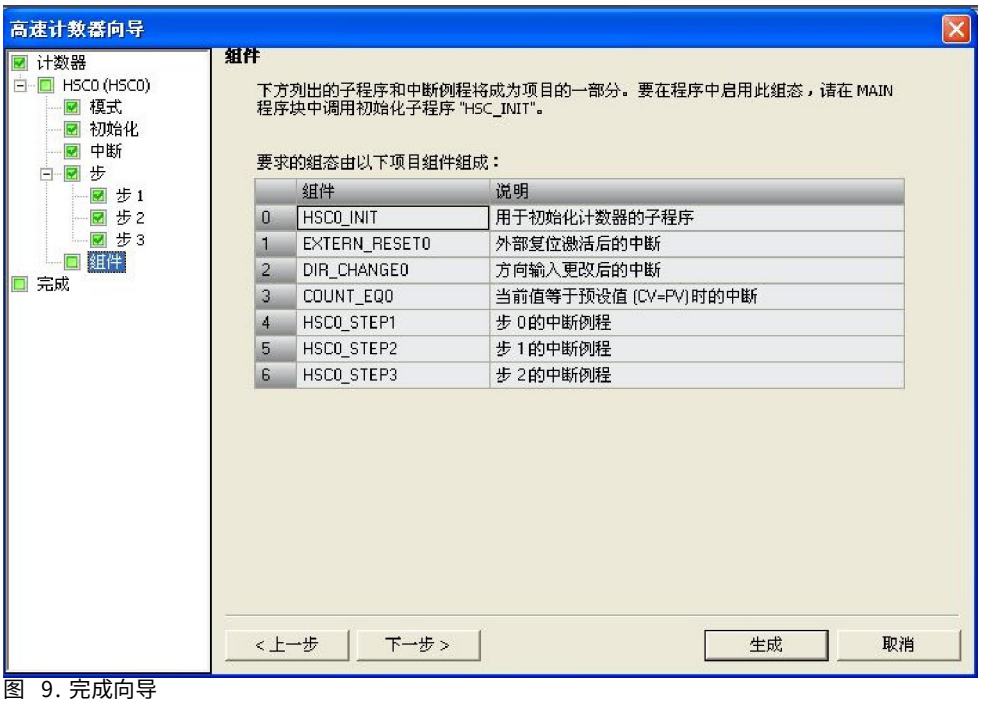
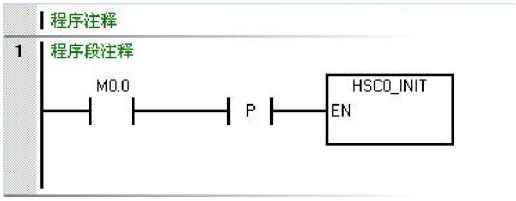


图 9. 完成向导

点击向导对话框左侧树形目录中的选项“组件（Components）”可以看到此时向导生成的子程序和中断程序名称及描述，点击“生成（Generate）”按钮，完成向导。

注意：Micro/WIN SMART 高速计数器指令向导采用树形目录的形式，用户可以直接在目录树中选择相应选项进行设置，这种方式便于用户在完成指令向导后根据实际需求进行快速修改。

步骤九：调用子程序：



⚠ **注意：**

- HSC_INIT 为初始化子程序，请在主程序块中使用 SM0.1 或一条边沿触发指令调用一次此子程序。
- 向导生成的中断服务程序及子程序都未上锁，用户可以根据自己的控制需要进行修改。

PID控制

S7-200 SMART能够进行 PID控制。 S7-200 SMART CPU最多可以支持 8个 PID控制回路（ 8个 PID指令功能块）。

PID是闭环控制系统的比例－积分－微分控制算法。

PID控制器根据设定值（给定）与被控对象的实际值（反馈）的差值，按照 PID算法计算出控制器的输出量，控制执行机构去影响被控对象的变化。

PID控制是负反馈闭环控制，能够抑制系统闭环内的各种因素所引起的扰动，使反馈跟随给定变化。

根据具体项目的控制要求，在实际应用中有可能用到其中的一部分，比如常用的是 PI（比例－积分）控制，这时没有微分控制部分。

PID算法在 S7-200 SMART中的实现

PID控制最初在模拟量控制系统中实现，随着离散控制理论的发展， PID也在计算机化控制系统中实现。

为便于实现， S7-200 SMART中的 PID控制采用了迭代算法。详细的计算方法请参考《 S7-200 SMART系统手册》中 PID指令部分的相关内容。

计算机化的 PID控制算法有几个关键的参数 K_c（Gain，增益）， T_i（积分时间常数）， T_d（微分时间常数）， T_s（采样时间）。

在 S7-200 SMART中 PID功能是通过 PID指令功能块实现。通过定时（按照采样时间）执行 PID功能块，按照 PID运算规律，根据当时的给定、反馈、比例－积分－微分数据，计算出控制量。

PID功能块通过一个 PID回路表交换数据，这个表是在 V数据存储区中的开辟，长度为 36字节。因此每个 PID功能块在调用时需要指定两个要素： PID控制回路号，以及控制回路表的起始地址（以 VB表示）。

由于 PID可以控制温度、压力等等许多对象，它们各自都是由工程量表示，因此有一种通用的数据表示方法才能被 PID功能块识别。 S7-200 SMART中的 PID功能使用占**调节范围**的百分比的方法抽象地表示被控对象的数值大小。在实际工程中，这个调节范围往往被认为与被控对象（反馈）的测量范围（量程）一致。

PID功能块只接受 0.0 - 1.0之间的实数（实际上就是百分比）作为反馈、给定与控制输出的有效数值，如果是直接使用 PID功能块编程，必须保证数据在这个范围之内，否则会出错。其他如增益、采样时间、积分时间、微分时间都是实数。

因此，必须把外围实际的物理量与 PID功能块需要的（或者输出的）数据之间进行转换。这就是所谓输入 /输出的转换与标准化处理。《 S7-200 SMART系统手册》上有详细的介绍。

- S7-200 SMART的编程软件 Micro/WIN SMART提供了 PID指令向导，以方便地完成这些转换 /标准化处理。除此之外， PID指令也同时会被自动调用。

调试 PID控制器

PID控制的效果就是看反馈（也就是控制对象）是否跟随设定值（给定），是否响应快速、稳定，是否能够抑制闭环中的各种扰动而回复稳定。

要衡量 PID参数是否合适，必须能够连续观察反馈对于给定变化的响应曲线；而实际上 PID的参数也是通过观察反馈波形而调试的。因此，没有能够观察反馈的连续变化波形曲线的有效手段，就谈不上调试 PID参数。

观察反馈量的连续波形，可以使用带慢扫描记忆功能的示波器（如数字示波器），波形记录仪，或者在 PC机上做的趋势曲线监控画面等。

- 新版编程软件 STEP 7-Micro/WIN SMART内置了一个 PID调试控制面板工具，具有图形化的给定、反馈、调节器输出波形显示，可以用于手动调试 PID参数。对于没有“ 自整定 PID”功能的老版 CPU，也能实现 PID手动调节。

PID参数的取值，以及它们之间的配合，对 PID控制是否稳定具有重要的意义。这些主要参数是：

- 采样时间：

计算机必须按照一定的时间间隔对反馈进行采样，才能进行PID控制的计算。采样时间就是对反馈进行采样的间隔。短于采样时间间隔的信号变化是不能测量到的。过短的采样时间没有必要，过长的采样间隔显然不能满足扰动变化比较快、或者速度响应要求高的场合。

- 编程时指定的PID控制器采样时间必须与实际的采样时间一致。S7-200 SMART中PID的采样时间精度用***定时中断***来保证。

- 增益（Gain，放大系数，比例常数）
  增益与偏差（给定与反馈的差值）的乘积作为控制器输出中的比例部分。过大的增益会造成反馈的振荡。

- 积分时间（Integral Time）
  偏差值恒定时，积分时间决定了控制器输出的变化速率。积分时间越短，偏差得到的修正越快。过短的积分时间有可能造成不稳定。
  积分时间的长度相当于在阶跃给定下，增益为“1”的时候，输出的变化量与偏差值相等所需要的时间，也就是输出变化到二倍于初始阶跃偏差的时间。
  如果将积分时间设为最大值，则相当于没有积分作用。

- 微分时间（Derivative Time）
  偏差值发生改变时，微分作用将增加一个尖峰到输出中，随着时间流逝减小。微分时间越长，输出的变化越大。微分使控制对扰动的敏感度增加，也就是偏差的变化率越大，微分控制作用越强。微分相当于对反馈变化趋势的预测性调整。
  如果将微分时间设置为0就不起作用，控制器将作为PI调节器工作。

常问问题

对于某个具体的PID控制项目，是否可能事先得知比较合适的参数？有没有相关的经验数据？

虽然有理论上计算PID参数的方法，但由于闭环调节的影响因素很多而不能全部在数学上精确地描述，计算出的数值往往没有什么实际意义。因此，除了实际调试获得参数外，没有什么可用的经验参数值存在。甚至对于两套看似一样的系统，都可能通过实际调试得到完全不同的参数值。

PID控制不稳定怎么办？如何调试PID?

闭环系统的调试，首先应当做开环测试。所谓开环，就是在PID调节器不投入工作的时候，观察：

- ***反馈通道的信号是否稳定***
- 输出通道是否动作正常

可以试着给出一些比较保守的PID参数，比如放大倍数（增益）不要太大，可以小于1，积分时间不要太短，以免引起振荡。在这个基础上，可以直接投入运行观察反馈的波形变化。给出一个阶跃给定，观察系统的响应是最好的方法。

如果反馈达到给定值之后，历经多次振荡才能稳定或者根本不稳定，应该考虑是否增益过大、积分时间过短；如果反馈迟迟不能跟随给定，上升速度很慢，应该考虑是否增益过小、积分时间过长……

总之，PID参数的调试是一个综合的、互相影响的过程，实际调试过程中的多次尝试是非常重要的步骤，也是必须的。

- S7-200 SMART的***新一代产品***提供了自整定的PID细调功能。

没有采用积分控制时，为何反馈达不到给定？

这是必然的。因为积分控制的作用在于消除纯比例调节系统固有的“静差”。没有积分控制的比例控制系统中，没有偏差就没有输出量，没有输出就不能维持反馈值与给定值相等。所以永远不能做到没有偏差。

如何实现PID反作用调节？

参见PID向导中的常问问题。

S7-200 SMART控制变频器，在变频器也有PID控制功能时，应当使用谁的PID功能？

可以根据具体情况使用。一般来说，如果需要控制的变量直接与变频器直接有关，比如变频水泵控制水压等，可以优先考虑使用变频器的PID功能。

参考链接

- 用PID指令向导进行PID编程
- PID自整定与PID调试面板
- PID的自动/手动切换

PID Wizard - PID向导

Micro/WIN SMART提供了 PID Wizard（PID指令向导），可以帮助用户方便地生成一个闭环控制过程的 PID算法。此向导可以完成绝大多数 PID运算的自动编程，用户只需在主程序中调用 PID向导生成的子程序，就可以完成 PID控制任务。

PID向导既可以生成模拟量输出 PID控制算法，也支持开关量输出；既支持连续自动调节，也支持手动参与控制。**建议用户使用此向导对 PID编程，以避免不必要的错误。**

建议用户使用较新的***编程软件版本***。在新版本中的 PID向导获得了改善。

PID向导编程步骤

使用以下方法之一打开 PWM向导：
●在 Micro/WIN SMART中的工具菜单中选择 **PID向导**：

图 1. 选择 PID向导

● 在项目树中打开“ 向导 ”文件夹，然后双击“ PID”，或选择“ PID”并按回车键。

图 2. 选择 PID向导

第一步：定义需要配置的 PID回路号

在此对话框中选择要组态的回路。 最多可组态 8 个回路。 在此对话框上选择回路时， PID 向导左侧的树视图随组态该回路所需的所有节点一起更新。

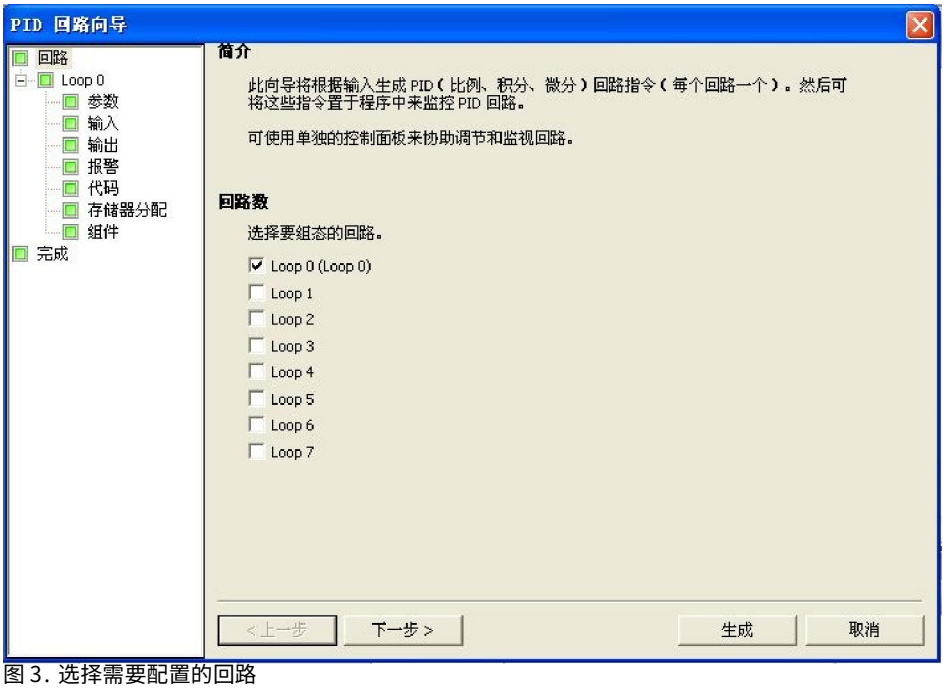


图 3. 选择需要配置的回路

第二步：为回路组态命名

可为回路组态自定义名称。 此部分的默认名称是“ 回路 x”，其中“ x”等于回路编号。

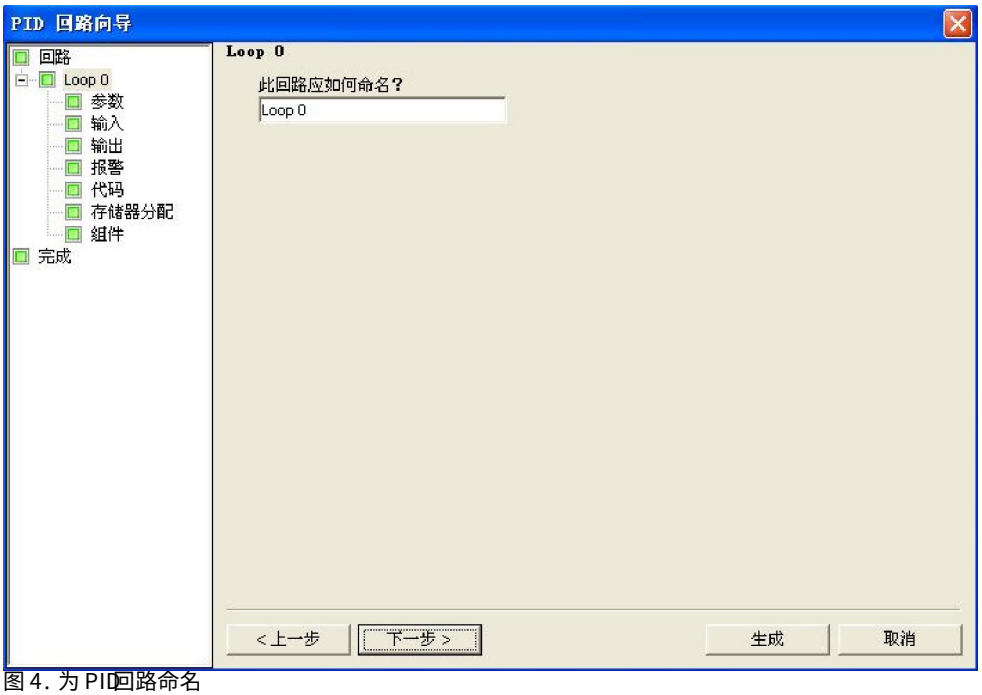


图 4. 为 PID回路命名

第三步：设定 PID回路参数

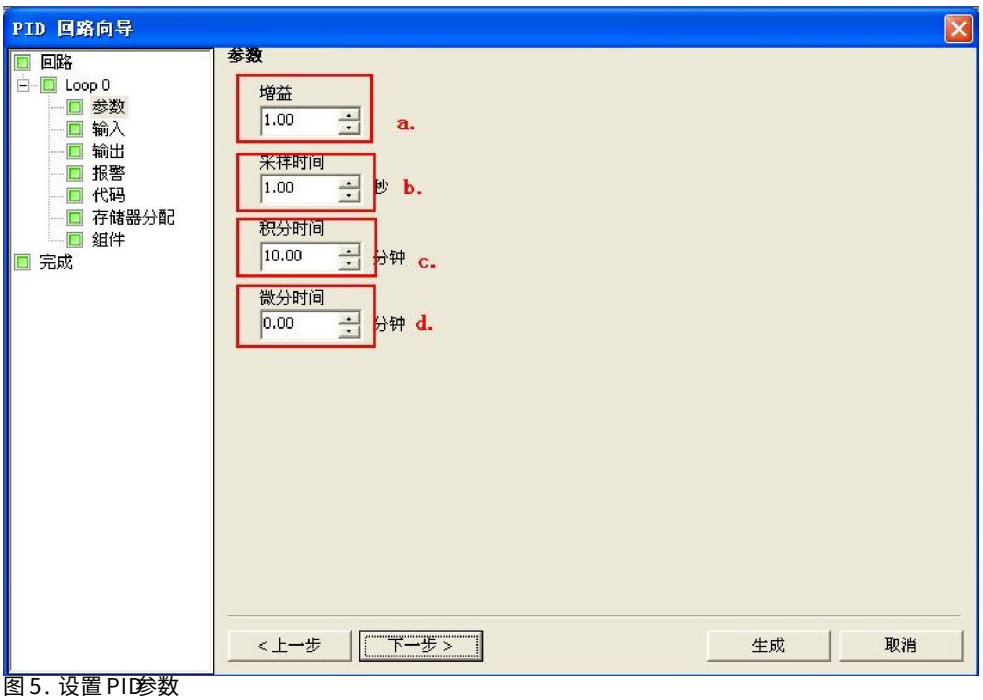


图 5. 设置 PID参数

图 5中定义了 PID回路参数，这些参数都应当是实数：

a. **增益**： 即比例常数，默认值 =1.00。
b. **积分时间**：如果不想要积分作用，默认值 =10.00。
c. **微分时间**：如果不想要微分回路，可以把微分时间设为 0 ，默认值 =0.00。
d. **采样时间**：是 PID控制回路对反馈采样和重新计算输出值的时间间隔 ,，默认值 =1.00。在向导完成后，若想要修改此数，则必须返回向导中修改，不可在程序中或状态表中修改。

⚠ **注意：**关于具体的 PID参数值，每一个项目都不一样，需要现场调试来定，没有所谓经验参数。

第四步：设定回路过程变量

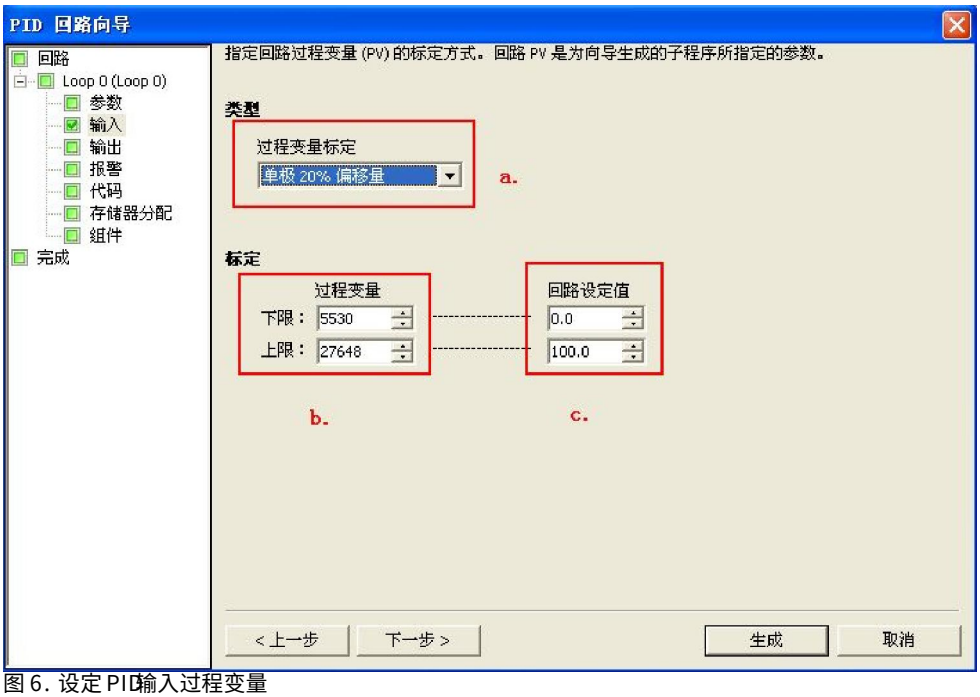


图 6. 设定 PID 输入过程变量

a. 指定回路过程变量 (PV) 如何标定。可以从以下选项中选择：
   - **单极性**：即输入的信号为正，如 0－10V 或 0－20mA 等
   - **双极性**：输入信号在从负到正的范围内变化。如输入信号为 ± 10V、± 5V 等时选用
   - **选用 20% 偏移**：如果输入为 4－20mA 则选单极性及此项，4mA 是 0－20mA 信号的 20%，所以选 20% 偏移，即 4mA 对应 5530，20mA 对应 27648。
   - **温度 x 10 ° C**
   - **温度 x 10 ° F**

b. 反馈输入取值范围
   - 在 a. 设置为单极时，缺省值为 0 - 27648，对应输入量程范围 0 - 10V 或 0 - 20mA 等，输入信号为正
   - 在 a. 设置为双极时，缺省的取值为 -27648 - +27648，对应的输入范围根据量程不同可以是 ± 10V、± 5V 等
   - 在 a. 选中 20% 偏移量时，取值范围为 5530 - 27648，不可改变

c. 在“标定”(Scaling) 参数中，指定回路设定值 (SP) 如何标定。默认值是 0.0 和 100.0 之间的一个实数。
   此反馈输入也可以是工程单位数值，参见：设置给定－反馈的量程范围。

第五步：设定输入回路输出选项

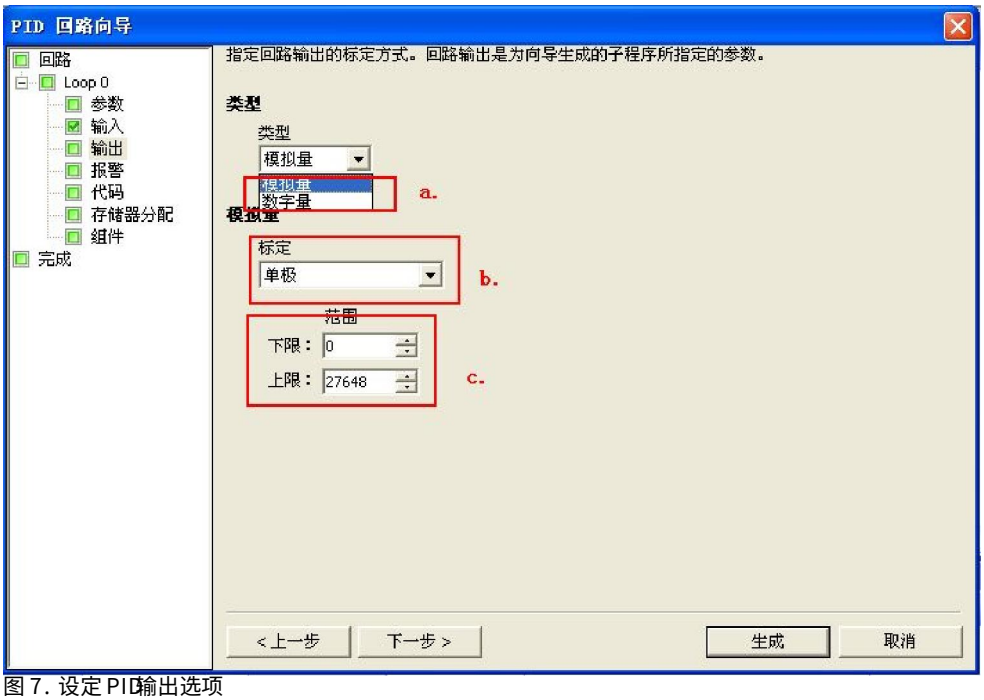


图 7. 设定 PID 输出选项

a. 输出类型
   可以选择模拟量输出或数字量输出。模拟量输出用来控制一些需要模拟量给定的设备，如比例阀、变频器等；数字量输出实际上

是控制输出点的通、断状态按照一定的占空比变化，可以控制固态继电器（加热棒等）

b. 选择模拟量则需设定回路输出变量值的范围，可以选择：
   - **单极**：单极性输出，可为 0– 10V或 0– 20mA等
   - **双极**：双极性输出，可为正负 10V或正负 5V等
   - **单极 20%偏移量**：如果选中 20%偏移，使输出为 4 - 20mA

c. 取值范围：
   - 为单极时，缺省值为 0 到 27648
   - 为双极时，取值 -27648 到 27648
   - 为 20%偏移量时，取值 5530 - 27648，不可改变

如果选择了开关量输出，需要设定此**循环周期**，如图 7所示：

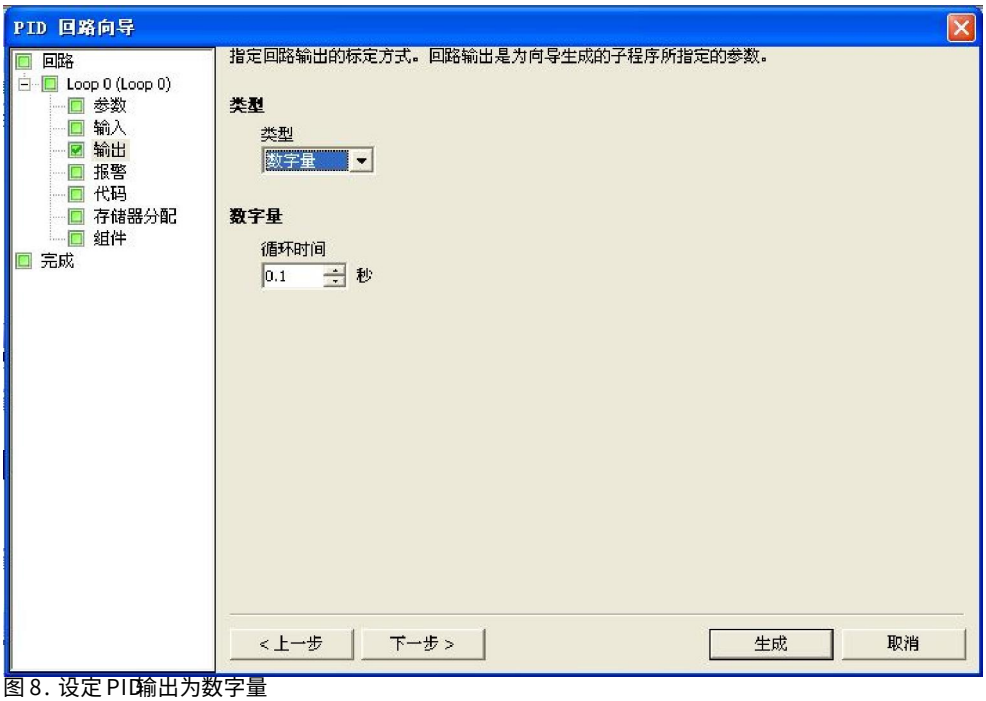


图 8. 设定 PID输出为数字量

第六步：设定回路报警选项

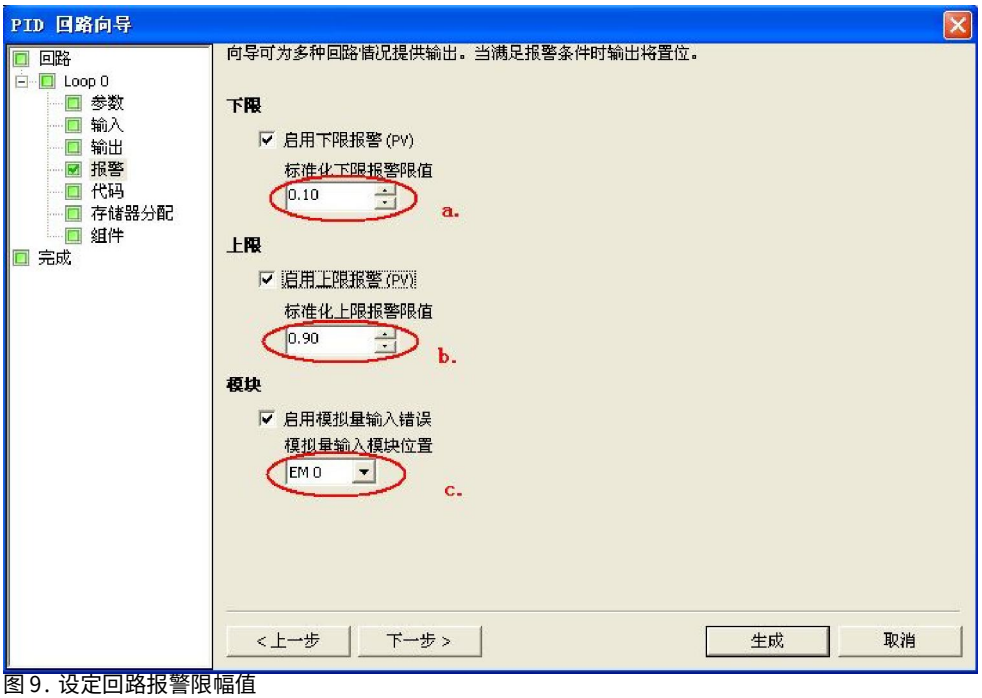


图 9. 设定回路报警限幅值

向导提供了三个输出来反映过程值 (PV)的低值报警、高值报警及过程值模拟量模块错误状态。当报警条件满足时，输出置位为 1。这些功能在选中了相应的选择框之后起作用。

a. 使能低值报警并设定过程值 (PV)报警的低值，此值为过程值的百分数，缺省值为 0.10，即报警的低值为过程值的 10%。此值最低

可设为 0.01，即满量程的 1%
b. 使能高值报警并设定过程值 (PV)报警的高值，此值为过程值的百分数，缺省值为 0.90，即报警的高值为过程值的 90%。此值最高可设为 1.00，即满量程的 100%
c. 使能过程值 (PV)模拟量模块错误报警并设定模块于 CPU连接时所处的模块位置。“ EM0”就是第一个扩展模块的位置

第七步：定义向导所生成的 PID初使化子程序和中断程序名及手 /自动模式

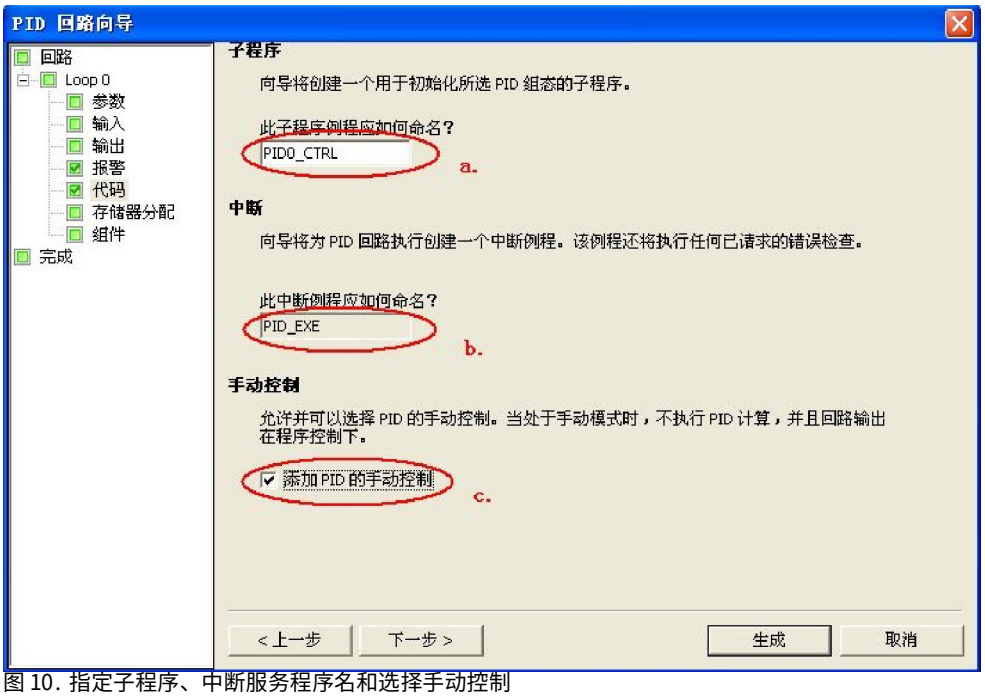


图 10. 指定子程序、中断服务程序名和选择手动控制

向导已经为初使化子程序和中断子程序定义了缺省名，你也可以修改成自己起的名字。

a. 指定 PID初使化子程序的名字。
b. 指定 PID中断子程序的名字

⚠ 注意：
1. 如果你的项目中已经存在一个 PID配置，则中断程序名为只读，不可更改。因为一个项目中所有 PID共用一个中断程序，它的名字不会被任何新的 PID所更改。
2. PID向导中断用的是 SMB34定时中断，在用户使用了 PID向导后，注意在其它编程时不要再用此中断，也不要向 SMB34中写入新的数值，否则 PID将停止工作。

c. 此处可以选择添加 PID 手动控制模式。在 PID手动控制模式下，回路输出由手动输出设定控制，此时需要写入手动控制输出参数一个 0.0— 1.0的实数，代表输出的 0%— 100%而不是直接去改变输出值。

➪ PID 控制的自动 /手动之间的无扰动切换

第八步：指定 PID运算数据存储区

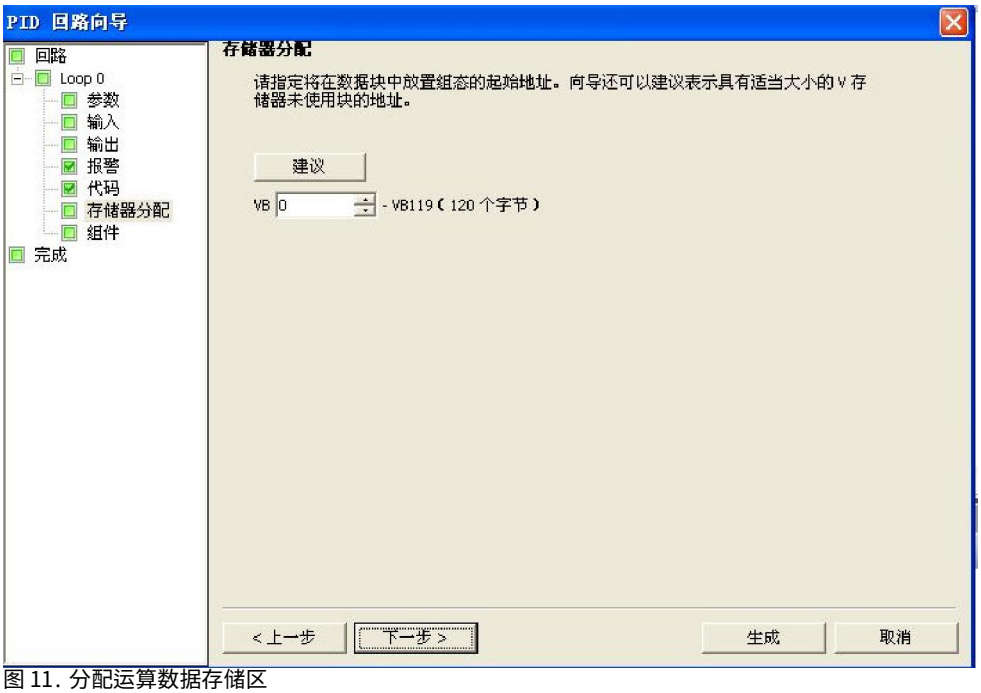


图 11. 分配运算数据存储区

PID指令（功能块）使用了一个 120个字节的 V区参数表来进行控制回路的运算工作；除此之外， PID向导生成的输入 /输出量的标准化程序也需要运算数据存储区。需要为它们定义一个起始地址，要保证该地址起始的若干字节在程序的其它地方没有被重复使用。如果点击“ 建议 ”，则向导将自动为你设定当前程序中没有用过的 V区地址。

⚠ 自动分配的地址只是在执行 PID向导时编译检测到空闲地址。向导将自动为该参数表分配符号名，用户不要再自己为这些参数分配符号名，否则将导致 PID控制不执行。

第九步：生成 PID子程序、中断程序及符号表等

一旦点击完成按钮，将在你的项目中生成上述 PID子程序、中断程序及符号表等。

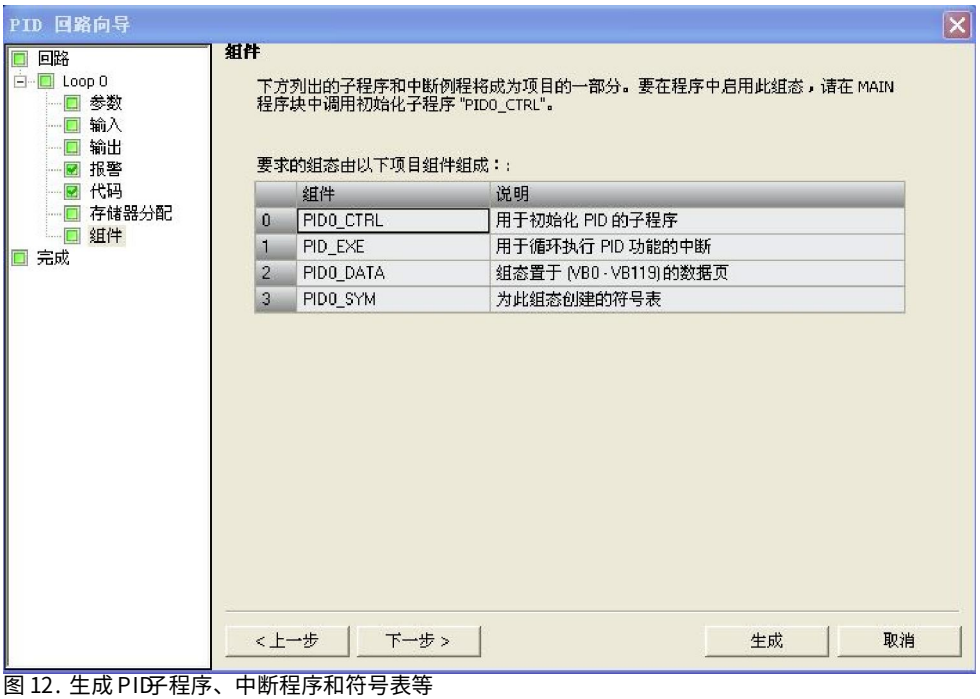


图 12. 生成 PID子程序、中断程序和符号表等

第十步：配置完 PID向导，需要在程序中调用向导生成的 PID子程序（如下图）

图 13．PID子程序

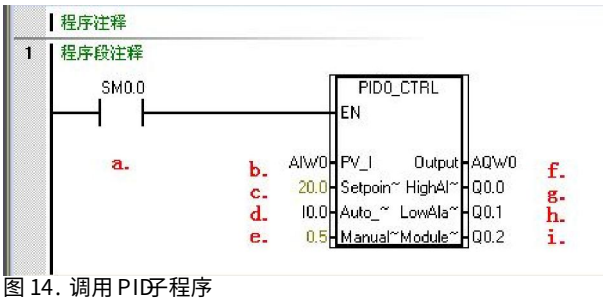


图 14．调用 PID子程序

在用户程序中调用 PID子程序时，可在指令树的程序块中用鼠标双击由向导生成的 PID子程序，在局部变量表中，可以看到有关形式参数的解释和取值范围。

a. **必须用 SM0.0来使能 PIDx_CTRL 子程序， SM0.0 后不能串联任何其他条件，而且也不能有越过它的跳转；如果在子程序中调用 PIDx_CTRL 子程序，则调用它的子程序也必须仅使用 SM0.0 调用，**以保证它的正常运行
b. 此处输入过程值（反馈）的模拟量输入地址
c. 此处输入设定值变量地址（VDxx），或者直接输入设定值常数，根据向导中的设定 0.0— 100.0，此处应输入一个 0.0— 100.0的实数，例：若输入 20，即为过程值的 20%，假设过程值 AIW0是量程为 0— 200度的温度值，则此处的设定值 20代表 40度（即 200度的 20%）；如果在向导中设定给定范围为 0.0 - 200.0，则此处的 20相当于 20度
d. 此处用 I0.0控制 PID的手 /自动方式，当 I0.0为 1时，为自动，经过 PID运算从 AQW0输出；当 I0.0为 0时， PID将停止计算， AQW0输出为 ManualOutput（VD4）中的设定值，此时不要另外编程或直接给 AQW0赋值。若在向导中没有选择 PID手动功能，则此项不会出现
e. 定义 PID手动状态下的输出，从 AQW0输出一个满值范围内对应此值的输出量。此处可输入手动设定值的变量地址（VDxx），或直接输入数。数值范围为 0.0-1.0之间的一个实数，代表输出范围的百分比。例：如输入 0.5，则设定为输出的 50%。若在向导中没有选择 PID手动功能，则此项不会出现
f. 此处键入控制量的输出地址
g. 当高报警条件满足时，相应的输出置位为 1，若在向导中没有使能高报警功能，则此项将不会出现
h. 当低报警条件满足时，相应的输出置位为 1，若在向导中没有使能低报警功能，则此项将不会出现
i. 当模块出错时，相应的输出置位为 1，若在向导中没有使能模块错误报警功能，则此项将不会出现

调用 PID子程序时，不用考虑中断程序。子程序会自动初始化相关的定时中断处理事项，然后中断程序会自动执行。

第十一步：实际运行并调试 PID参数

没有一个 PID项目的参数不需要修改而能直接运行，因此需要在实际运行时调试 PID参数。

查看数据块以及符号表相应的 PID符号标签的内容，可以找到包括 PID核心指令所用的控制回路表，包括比例系数、积分时间等等。将此表的地址复制到状态表中，可以在监控模式下在线修改 PID参数，而不必停机再次做配置。

参数调试合适后，用户可以在数据块中写入，也可以再做一次向导，或者编程向相应的数据区传送参数。

常问问题

做完 PID向导后，如何知道向导中设定值，过程值及 PID等参数所用的地址？

- 做完 PID向导后可在符号表中，查看 PID向导所生成的符号表 (上例中为 PID0_SYM)，可看到各参数所用的详细地址，及数值范围。
- 在数据块中，查看 PID指令回路表的相关参数。如图所示：

符号表

			符号	地址	注释
1			PID0_Low_Alarm	VD116	下限报警限值
2			PID0_High_Alarm	VD112	上限报警限值
3			PID0_Mode	V122.0	
4			PID0_WS	VB122	
5			PID0_D_Counter	VW120	
6			PID0_D_Time	VD24	微分时间
7			PID0_I_Time	VD20	积分时间
8			PID0_SampleTime	VD16	采样时间（要进行修改，请重新运行 PID 向...
9			PID0_Gain	VD12	回路增益
10			PID0_Output	VD8	计算得出的标准化回路输出
11			PID0_SP	VD4	标准化过程设定值
12			PID0_PV	VD0	标准化过程变量
13			PID0_Table	VB0	PID 0的回路表起始地址

图 15．PID数据块

做完 PID向导后，如何在调试中修改 PID参数？

可以在状态表中，输入相应的参数地址，然后在线写入用户需要的 PID参数数值，这样用户就可根据工艺需要随时对 PID参数、设定

值等进行调整。

PID已经调整合适，如何正式确定参数？

可以在数据块中直接写入参数。

做完PID向导后，能否查看PID生成的子程序，中断程序？

PID向导生成的子程序，中断程序用户是无法看到的，也不能对其进行修改。没有密码能够打开这些子程序，一般的应用也没有必要打开查看。

PID参数有经验值吗？

每一个项目的 PID 参数都不一样，没有经验参数，只能现场调试获得。

我的PID向导生成的程序为何不执行？

- 必须保证用 SM0.0无条件调用 PID0_CTRL程序
- 在程序的其它部分不要再使用 SMB34定时中断，也不要对 SMB34赋值

如何实现PID反作用调节？

在有些控制中需要 PID反作用调节。例如：在夏天控制空调制冷时，若反馈温度（过程值）低于设定温度，需要关阀，减小输出控制（减少冷水流量等），这就是 PID反作用调节（在 PID正作用中若过程值小于设定值，则需要增大输出控制）。

若想实现 PID反作用调节，需要把 PID回路的增益设为负数。对于增益为 0的积分或微分控制来说，如果指定积分时间、微分时间为负值，则是反作用回路。

如何根据工艺要求有选择地投入PID功能？

可使用“ 手动 /自动 ”切换的功能。 PID向导生成的 PID功能块只能使用 SM0.0的条件调用。

参考链接

PID功能简介

PID 向导 中的给定－反馈设置

完成 PID Wizard配置后，会为每个 PID回路生成一个子程序 PIDx_CTRL（x = 0 - 7）。在用户程序中，必须使用 SM0.0始终调用这个子程序才能实现 PID功能。

下图是一个最简单的 PID子程序调用程序段：

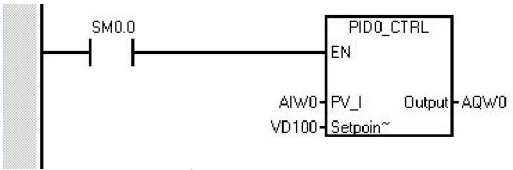


图 1. 调用 PID子程序

其中：

- PV_I： 过程反馈参数值的入口
- Setpoint： 给定参数值的入口
- Output： PID调节器的输出值

在这里，给定、反馈的入口参数不是 PID指令功能块所需要的 0.0 - 1.0之间的实数，而可以是实际的反馈地址，或是其他变量。例如， PV_I可以是模拟量输入地址 AIW0，也可以是存储器地址 VW100等；Setpoint则往往来自 V变量存储区，这样可以从人机操作界面（HMI）设备输入给定值。

注意：

对于 PID控制系统来说，必须保证给定与过程反馈的一致性：

- 给定与反馈的物理意义一致

  这取决于被控制的对象，如果是压力，则给定也必须对应于压力值；如果是温度，则给定也必须对应于温度。

- 给定与反馈的数值范围对应

如果给定直接是摄氏温度值，则反馈必须是对应的摄氏温度值；如果反馈直接使用模拟量输入的对应数值，则给定也必须向反馈的数值范围换算。

如果给定与反馈的换算有特定的比例关系也可以。如给定也可以表示为以反馈的数值范围的百分比数值。

给定与反馈的数值具体是什么数值，其取值范围究竟如何，完全取决于我们在使用“ PID向导 ”编程时指定的给定与反馈的数值范围。其中，反馈量的数值范围不能随便自己定义，而要取决于具体应用的模拟量输入模块。

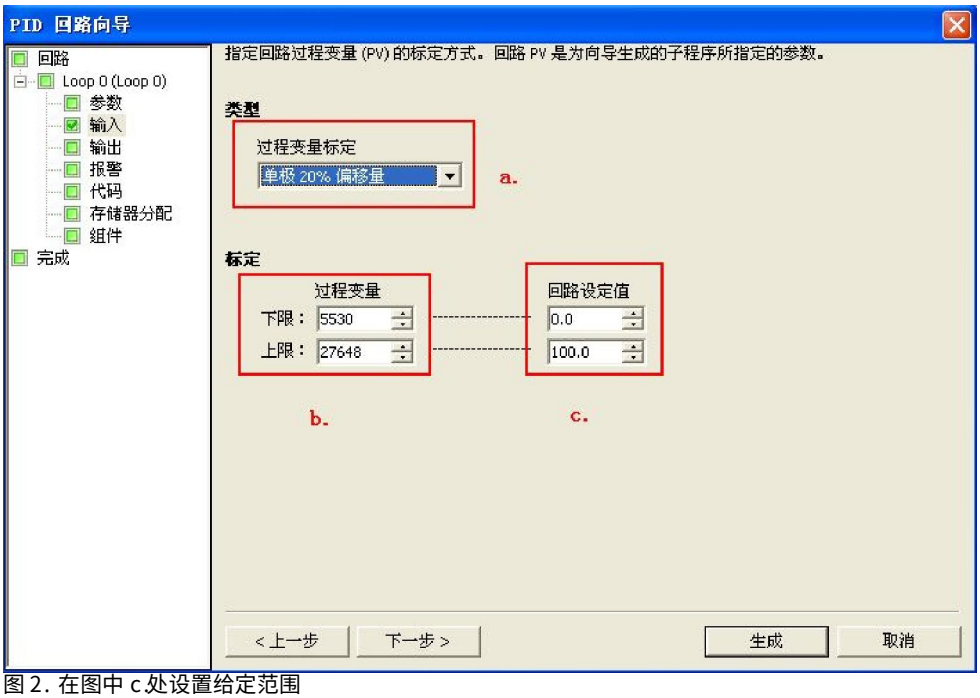


图 2. 在图中 c 处设置给定范围

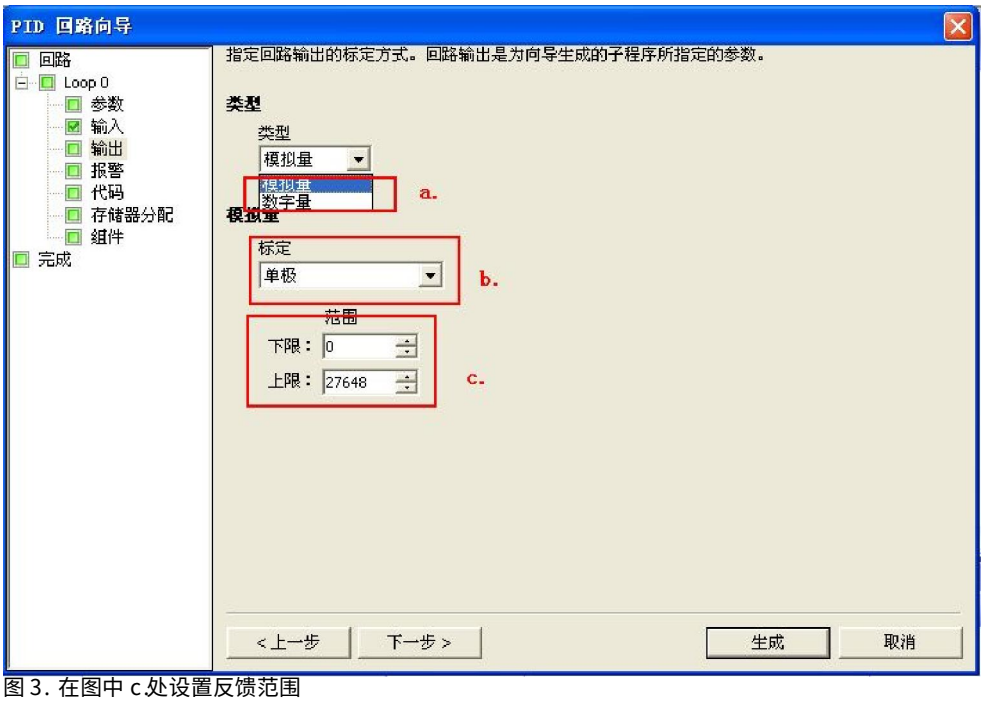


图 3. 在图中 c 处设置反馈范围

实例

假定一个 PID控制系统的控制对象是压力，反馈元件的测量范围为 0 - 16MPa，反馈器件的信号经过变换，以 0 - 20mA（或 4 - 20mA）电流信号的形式输入到 EM231模拟量输入模块中。据此，我们可以按下表设置给定、反馈的范围。

表 1.

	反馈（单极性）		给定	
	实际物理量	模拟量输入数值	百分比形式（占 0 - 16MPa的百分比）	物理工程单位形式
高限	16 MPa	32000	100.0	n× 16.0

低限	0 MPa	0 （0 - 20mA） 6400 （4 - 20mA）	0.0	0.0

n 为比例系数，为了精度高些可以设置 n=10等等

又如一个温度控制的 PID系统，温度值直接由热电偶测量，输入到 EM231 TC（热电偶）模块转换为温度 值。热电偶为 J型，其测量范围为 -150.0° C - 1200.0 ° C。则可按如下设置给定的范围。

表 2.

	反馈（双极性）		给定	
	实际物理量	**模拟量输入数值**	**百分比形式（占 -150° C - 1200.0° C的百分比）**	**物理工程单位形式**
高限	1200.0 ° C	12000	100.0	1200
低限	-150.0 ° C	-1500	0.0	-150

在上面的例子中，反馈和给定可以按照如下方法设置

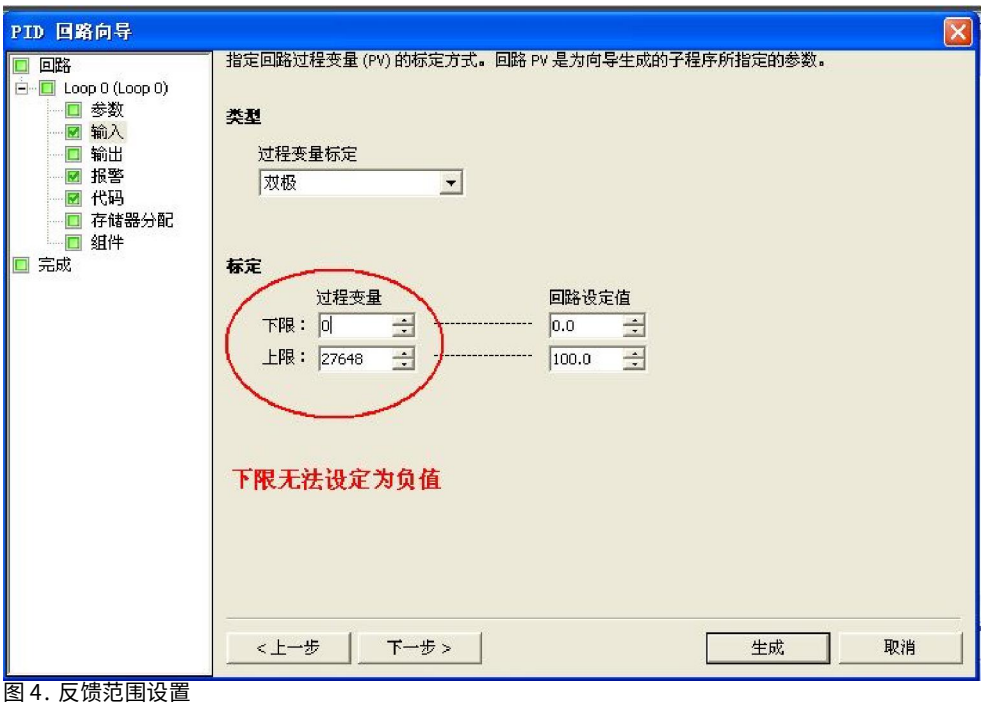


图 4. 反馈范围设置

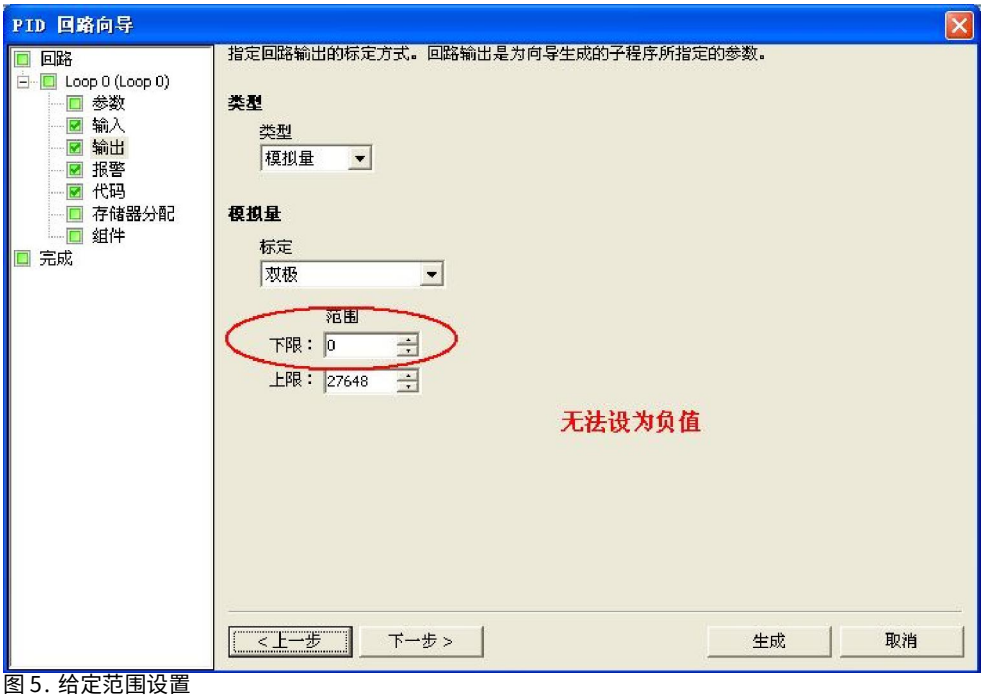


图 5. 给定范围设置

PID自整定

新的 S7-200 SMART CPU支持 PID自整定功能，在 STEP 7-Micro/WIN SMART中也添加了 PID调节控制面板。

用户可以使用用户程序或 PID调节控制面板来启动自整定功能。在同一时间最多可以有 8个 PID回路同时进行自整定。 PID调节控制面板也可以用来手动调试老版本的（不支持 PID自整定） CPU的 PID控制回路。

用户可以根据工艺要求为调节回路选择快速响应、中速响应、慢速响应或极慢速响应。 PID自整定会根据响应类型而计算出最优化的比例、积分、微分值，并可应用到控制中。

PID调节控制面板

STEP 7-Micro/WIN SMART中提供了一个 PID控制面板，可以用图形方式监视 PID回路的运行，另外从面板中还可以启动、停止自整定功能。

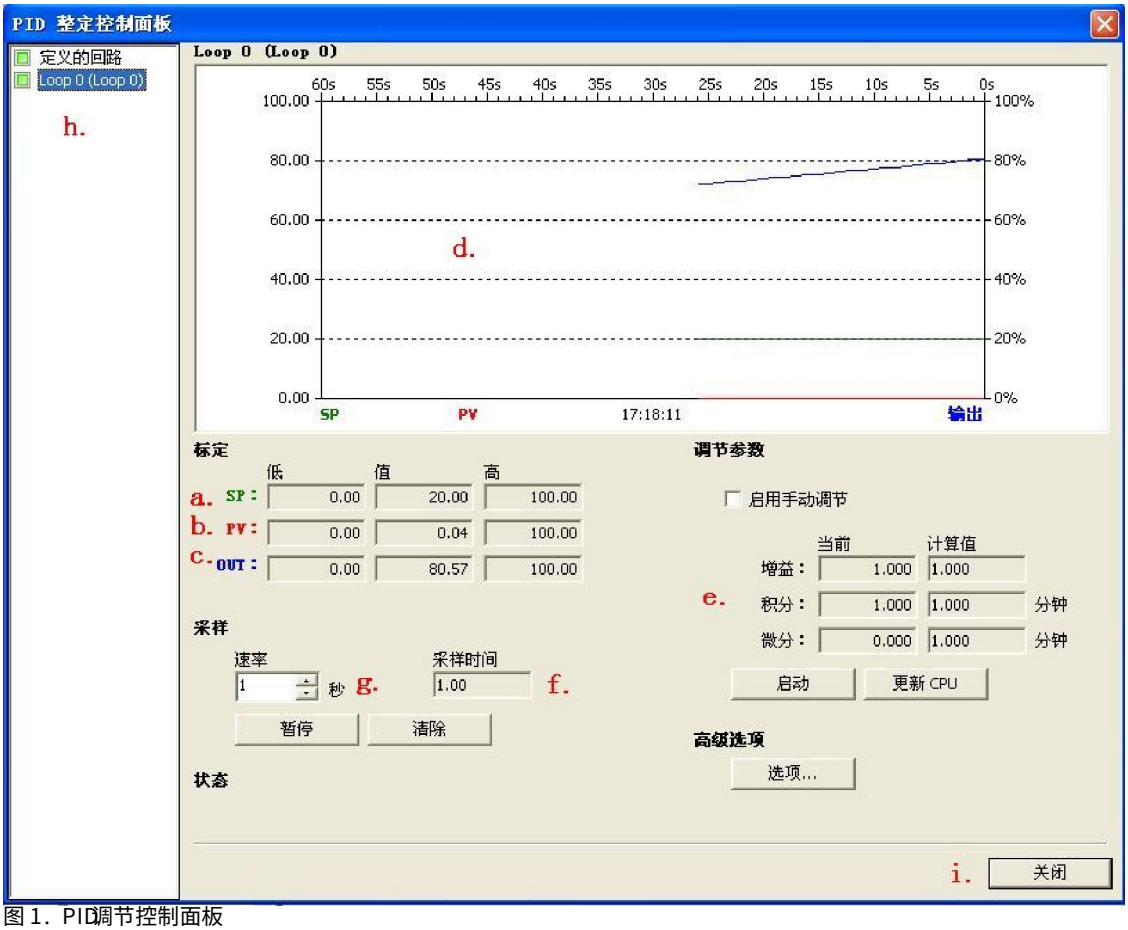


图 1. PID调节控制面板

在图 1中：

a. 当前设定值指示
显示当前使用的设定值

b. 过程值指示
显示过程变量的值

c. 当前的输出值指示
显示当前的输出值

d. 可显示过程值、设定值及输出值的 PID趋势图

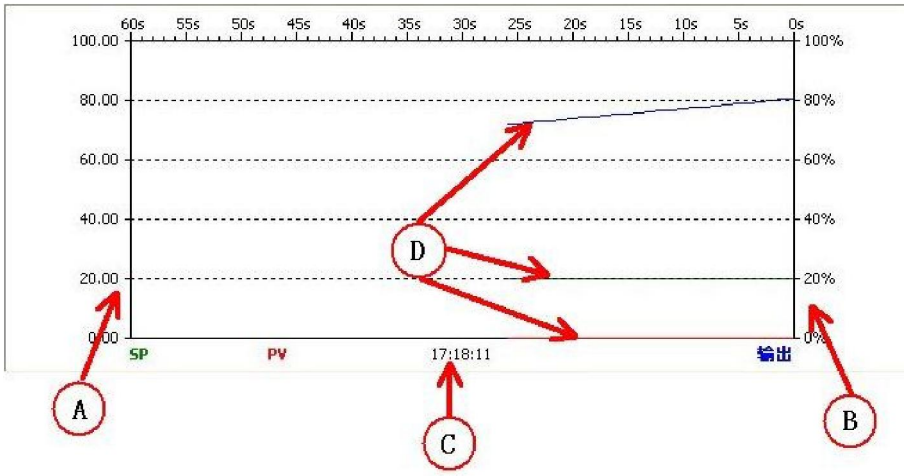


图 2. 图形显示区

图中：
A. 过程变量和设定值的取值范围及刻度
B. PID输出的取值范围及刻度
C. 实际PC时间
D. 以不同颜色表示的设定值、过程变量及输出的趋势图

e. 调节参数
这里你可以：
- 选择 PID参数的显示：当前参数、推荐参数或手动输入值
- 在手动调节模式下，可改变 PID参数，并按更新 PLC按钮来更新 PLC中的参数
- 启动 PID自整定功能
- 选择高级选项按钮进入高级参数设定

f. 当前采样时间
指示当前使用的采样时间

g. 时间选项设定
这里你可以设定趋势图的时基，时基以秒为单位

h. 当前的 PID回路号
这里你可以选择需要监视或自整定的 PID回路

i. 关闭 PID调节面板

⚠ 要使用 PID调节控制面板， PID编程必须使用 PID向导完成。

PID自整定步骤

第一步：在 PID向导中完成 PID功能配置

⚠ 要想使用 PID自整定功能， PID编程必须用 PID向导来完成

第二步：打开 PID调节控制面板，设置 PID回路调节参数

在Micro/WIN SMART在线的情况下，从主菜单工具中点击 

进入 PID调节控制面板中。

在 PID调节面板的 h.区查看已选择的 PID回路号，在 e.区启动手动调节，调节 PID参数并点击更新，使新参数值起作用，监视其趋势图，根据调节状况改变 PID参数直至调节稳定。

⚠ 为了使 PID自整定顺利进行，应当做到：

- 使 PID调节器基本稳定，输出、反馈变化平缓，并且使反馈比较接近给定
- 设置合适的给定值，使 PID调节器的输出远离趋势图的上、下坐标轴，以免 PID自整定开始后输出值的变化范围受限制

➪ 参见：手动调整 PID回路参数

第三步：在 e.区点击高级选项按钮，设定 PID自整定选项。如果不是很特殊的系统，也可以不加理会。

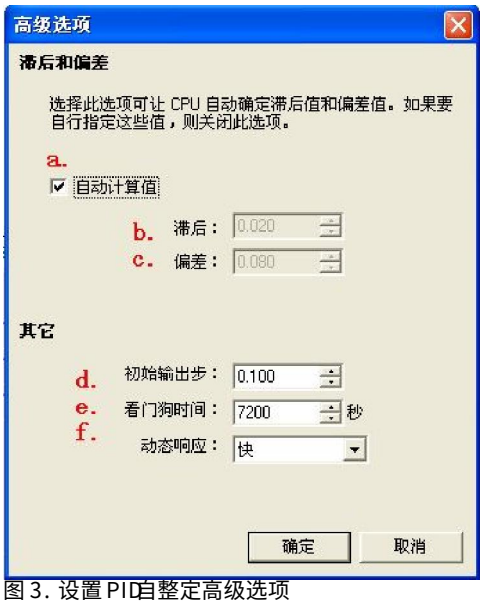


图 3. 设置 PID自整定高级选项

在此允许你设定下列参数：

a. 你可以选中复选框，让自整定来自动计算滞后值和偏值

对于一般的 PID系统，建议使用自动选择。

b. 滞后：

滞后值规定了允许过程值偏离设定值的最大（正负）范围，过程反馈在这个范围内的变化不会引起 PID自整定调节器改变输出，或者使 PID自整定调节器“ 认为 ”这个范围内的变化是由于自己改变输出进行自整定调节而引起的。 PID自整定开始后，只有过程反馈值超出了该区域， PID自整定调节器才会认为它对输出的改变发生了效果。这个值用来减少过程变量的噪声对自整定的干扰，从而更精确地计算出过程系统的自然振动频率。如果选用自动计算，则缺省值为 2%。

如果过程变量反馈干扰信号较强（噪声大）自然变化范围就大，可能需要人为设置一个较大的值。但这个值的改变要与下面的偏差值保持 1:4的关系。

c. 偏差：

偏差值决定了允许过程变量偏离设定值的峰峰值。如果选择自动计算该值，它将是死区的 4倍，即 8%。

有些非常敏感的系统不允许过程量偏离给定值很多，也可以人工设置为比较小的值，但是要和上述“ 死区 ”设置保持比例关系。这就是说，一个精度要求高的系统，其反馈信号必须足够稳定。

d. 初始输出步： PID调节的初始输出值

PID自整定开始后， PID自整定调节器将主动改变 PID的输出值，以观察整个系统的反应。初始步长值就是输出的变动第一步变化值，以占实际输出量程的百分比表示。

e. 看门狗时间：过程变量必须在此时间（时基为秒）内达到或穿越给定值，否则会产生看门狗超时错误。

PID自整定调节器在改变输出后，如果超过此时间还未观察到过程反馈（从下至上或从上至下）穿越给定曲线，则超时。如果能够事先确定实际系统响应非常慢，可以加长这个时间。

f. 动态响应选项：根据回路过程（工艺）的要求可选择不同的响应类型：快、中、慢、非常慢

- 快：可能产生超调，属于欠阻尼响应
- 中：在产生超调的边缘，属于临界阻尼响应
- 慢：不会产生任何超调，属于过阻尼响应
- 非常慢：不会产生任何超调，属于严重过阻尼响应

用户在这里指定需要达到的系统控制效果，而不是对系统本身响应快慢的判断。

g. 设定完参数点击 OK键回到 PID调节控制面板的主画面

第四步：在手动将 PID调节到稳定状态后，即过程值与设定值接近，且输出没有不规律的变化，并最好处于控制范围中心附近。此时可点击 e区内的启动按钮启动 PID自整定功能，这时按钮变为停止。这时只需耐心等待，系统完成自整定后会自动将计算出的 PID参数显示在 e区。当按钮再次变为启动时，表示系统已经完成了 PID自整定。

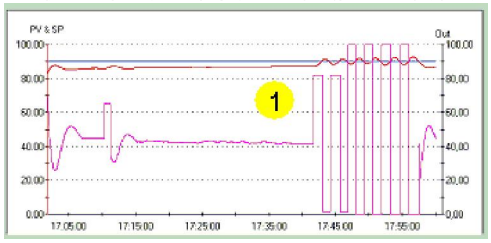
要使用自整定功能，必须保证 PID回路处于自动模式。开始自整定后，给定值不能再改变。

第五步：如果用户想将 PID自整定的参数应用到当前 PLC中，则只需点击更新 PLC。

完成 PID调整后，最好下载一次整个项目（包括数据块），使新参数保存到 CPU的 EEPROM中。

PID自整定失败的原因

1. PID输出在最大值与最小值之间振荡（曲线接触到坐标轴）



解决方法：降低 PID初始输出步长值

2. 经过一段时间后， PID自整定面板显示如下信息：“ 自整定计算因为等待反馈穿越给定值的看门狗超时而失败 ”。

解决方法： 确定在启动 PID自整定前，过程变量和输出值已经稳定。并检查看门狗时间的值，将其适当增大。

对于其它错误，可参考手册中表格 11－ 3中的错误代码的描述。

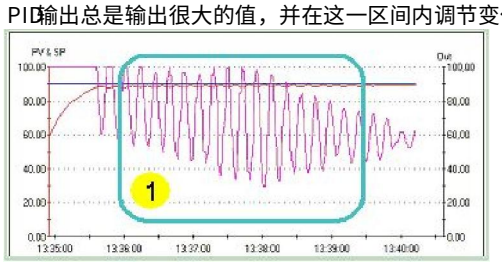
如何获得一个稳定的 PID回路

在开始 PID自整定调整前，整个 PID控制回路必须工作在相对稳定的状态。

稳定的 PID是指过程变量接近设定值，输出不会不规则的变化，且回路的输出值在控制范围中心附近变化。

问题与解决方法：

1. PID输出总是输出很大的值，并在这一区间内调节变化

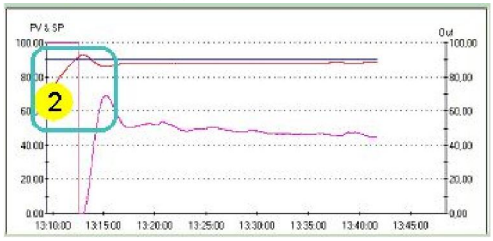


产生原因：

- 增益值太高
- PID扫描时间太长（对于快速响应 PID的回路）

解决方法：降低增益值并且 /或选择短一些的扫描时间

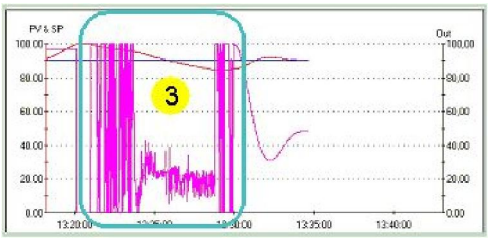
2. 过程变量超过设定值很多（超调很大）



产生原因：积分时间可能太高

解决方法：降低积分时间

3. 得到一个非常不稳定的 PID



产生原因：

- 如果用了微分，可能是微分参数有问题
- 没有微分，可能是增益值太高

解决方法：

- 调整微分参数到 0— 1的范围内
- 根据回路调节特性将增益值降低，最低可从 0.x 开始逐渐增大往上调，直到获得稳定的 PID。

PID 自动 /手动调节的无扰动切换

有些工程项目中可能需要根据工艺要求在不同的时刻投入、或者退出 PID 自动控制；退出 PID 自动控制时，控制器的输出部分可以由操作人员直接手动控制。这就是所谓的 PID 手动 /自动切换。

PID 控制处于自动方式时， PID 控制器（ S7-200 SMART 中的 PID 调节功能）会按照 PID 算法，自动通过输出的作用使过程反馈值跟随给定值变化，并保持稳定。这是一个自动的闭环控制系统。操作人员可以根据现场工艺的要求，改变给定（即设定值）的值。

PID 控制处于手动方式时， PID 控制器不再起自动计算的作用。这时，控制回路的输出是由操作人员手动控制、调整，由操作人员观察现场的控制效果，从而构成人工闭环控制。

> 所谓 PID 自动 /手动控制，就是看控制系统的**输出**是由 PID 控制器自动控制，还是由操作人员手动控制。

有些控制系统的执行机构不能承受较大的冲击，这就要求在进行 PID 自动 /手动切换时，保持控制**输出**的稳定。这就是要求无扰动切换。

为了达到 PID 自动 /手动控制的无扰动切换，需要在编程时注意一些相关事项。下面分别就直接使用 PID 指令编程，和使用 PID 向导编程两种情况作一介绍。

直接使用 PID 指令编程时的 PID 自动 /手动无扰切换

直接使用 PID 指令块编写 PID 控制程序时，可以简单地使用“ 调用 /不调用 ”指令的方式控制自动 /手动模式。因为 PID 指令本身已经具有实现无扰动切换的能力，此时在 PID 指令控制环节之外编程没有多大必要。

PID 指令的 EN 输入端使能（为“ 1”）时，我们认为是自动控制模式； EN 输入端未使能（为“ 0”）时，我们认为是手动控制模式。

PID 指令本身有一个“ 能流历史状态位 ”，以记录指令的状态切换。在 EN 端从“ 0”变为 ”“ 1”时， PID 指令认为这是从“ 手动 ”模式向“ 自动 ”模式切换。 PID 指令此时会自动执行一系列动作，以配合无扰动切换：

- 使设定值（ SPn） = 当前过程反馈变量（ PVn）
- 设置上次采样过程变量（ PVn-1） = 当前过程反馈变量（ PVn）
- 设置积分偏差和（或所谓积分前项）（ Mx） = 当前输出值（ Mn）

使设定值等于当前反馈值可以避免出现偏差，使之不存在调整的要求；当然如果有工艺要求，也可以后续调整设定值。其他的动作都是为了使 PID 在后续的操作中不改变输出的值。

在编程时要注意：

- 从自动模式向手动模式切换时， PID 指令的 EN 端不再有能流，计算停止，输出值 Mn 不再变化。此时如果需要操作人员人工观察控制的结果，手动控制输出量，就可以通过用户程序直接改变回路表中的输出值存储单元内容（见数据块或系统手册的相关部分内容）。如果有必要，操作人员的操作可能要进行一些标准化换算。
- 为保证从手动模式向自动模式的切换无扰动，需要在手动控制时，或在切换过程中，禁止对 PID 回路表中设定值的更新，以便切换时 PID 指令用当前过程反馈值替代设定值。切换完成后，操作人员可以调整设定值。

使用 PID 向导编程时的 PID 自动 /手动无扰切换

使用 PID 指令向导编程时，指令向导会自动调用 PID 指令，并且编写外围的控制变量标准化换算、定时采样等功能。用户在使用 PID 指令向导时，需要在用户程序中用 SM0.0 调用指令向导生成的子程序（如 PIDx_CTRL 子程序）。 PID 向导可以生成带自动 /手动切换功能的子程序，这个子程序使用一个数字量点为“ 1”、“ 0”的状态来控制是否投入 PID 自动控制。

> 到目前为止（ STEP 7-Micro/WIN SMART） ，使用 PID 向导生成的子程序时，由于用户程序不能直接使用 PID 指令，它的无扰切换能力因为隔了外壳子程序，所以受到了局限。如果对无扰切换要求比较严格，需要另外编一些程序加以处理。

考察如下 PID 控制子程序。

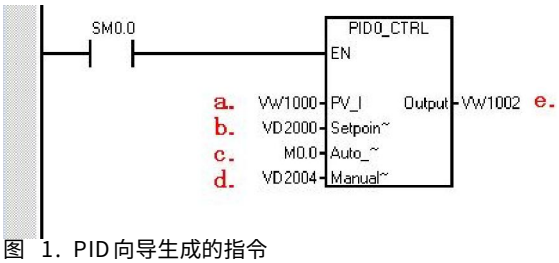


图 1. PID 向导生成的指令

图中：

a. 过程反馈量
b. 设定值，实数
c. 自动 /手动控制，“ 1”=自动，“ 0”=手动
d. 手动控制输出值， 0.0 - 1.0 之间的一个实数
e. PID 控制输出值

要实现无扰动切换，必须：

- 在从自动向手动切换时，使手动输出值（ VD2004）等于当前的实际控制输出值；
- 在从手动向自动切换使，使设定值相当于当前的过程反馈值。

为此，可编写类似下图所示的程序，放在 PID 控制子程序之前：

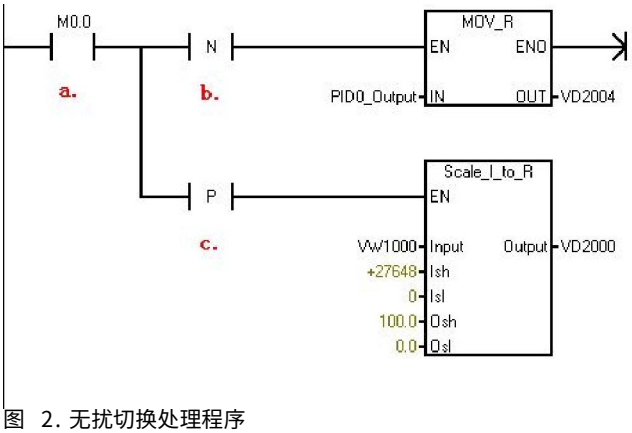


图 2. 无扰切换处理程序

图中：

a. 自动 /手动切换控制点
b. 从自动向手动切换时，使手动输出值等于实际当前值
c. 从手动向自动切换时，把当前反馈量换算为相应的给定值

上述程序中的 Scale_I_to_R 就是量程变换指令库中的子程序，如何将该指令库导入到 Micro/WIN SMART 中可参见如何将 Micro/WIN 的库文件导入到 Micro/WIN SMART 中。这是为了解决过程反馈与设定值之间的换算问题。用户也可以自己编程换算，或者根据反馈与给定的取值范围决定是否需要换算。

脉宽调制（ PWM）

PWM 提供了占空比可变周期固定的输出。 PWM 输出以指定频率（循环时间）启动之后将连续运行。 脉宽根据所需要的控制要求进行变化。 占空比可表示为周期的百分比或对应于脉冲宽度的时间值。 脉宽的变化范围为 0%（无脉冲，始终为低电平 ）到 100%（无脉冲，始终为高电平）。 请参见下图。

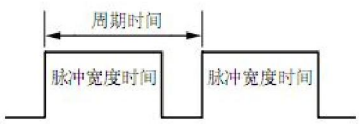


图 1

由于 PWM 输出可从 0%变化到 100%，因此在很多情况下，它可以提供一个类似于模拟量输出的数字量输出。 例如， PWM 输出可用于电机从静止到全速的速度，或用于阀从关闭到全开的位置控制。

PWM 功能产生一个占空比变化周期固定的脉冲输出。你可以为其设定周期和脉宽（以微秒或毫秒为单位）：

- 周期：10μs到65535μs或者2ms到65535ms（S7-200 SMART CPU支持最高100KHz脉冲输出）
- 脉宽：0μs到65535μs或者0ms到65535ms（最低4μs，设置为0μs等于禁止输出。）

PWM发生器的详细参数和例程请参见《S7-200 SMART系统手册》。

使用PWM输出

S7-200 SMART CPU提供三个数字量输出（Q0.0、Q0.1和Q0.3），这三个数字量输出可以通过PWM向导组态为PWM输出。

为PWM操作组态输出时，输出的周期是固定的，脉宽或脉冲占空比可通过程序进行控制。

为了简化应用中运动控制的使用，STEP 7-Micro/WIN SMART提供了“PWM向导”来组态PWM。

更改PWM特性

只能使用同步更新更改PWM波形的特性。执行同步更新时，信号波形特性的更改发生在周期交界处，这样可实现平滑转换。

使用SM特殊寄存器组态控制PWM操作

PLS指令读取存储在指定SM存储单元的数据，并相应地对PWM发生器进行编程。SMB67控制PWM 0，SMB77控制PWM 1，而SMB567控制PWM 2。
可以更改PWM波形的特性，方法是修改SM存储区中的位置（包括控制字节），然后执行PLS指令。可以随时禁止生成PWM波形，方法是将0写入控制字节的PWM使能位（SM67.7、SM77.7或SM567.7），然后执行PLS指令。

- 加载新的脉冲计数（SMD72或SMD82）、脉冲宽度（SMW70或SMW80）或周期时间（SMW68或SMW78）时，也会在执行PLS指令之前置位控制寄存器中的相应更新位。

- 中止PWM操作后，应经过一个周期时间，然后再重新启用PWM通道进行操作。如果未经过此时间就重新启用PWM通道，则可能因完成原始PWM命令而导致初始脉冲出现脉冲失真。

- 如果在PWM正在执行时尝试更改PWM的时基，则该请求将被忽略，并会出现非致命错误（0x001B - ILLEGAL PWM TIMEBASE CHG）。

PWM控制寄存器的设置详见《S7-200 SMART系统手册》。

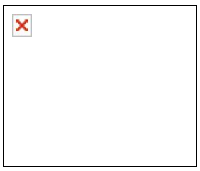
组态PWM输出

要为PWM组态其中一个内置输出，请使用PWM向导。

图1. PWM向导

使用以下方法之一打开PWM向导：
● 在“工具”(Tools)菜单的“向导”(Wizards)区域单击“PWM”按钮。

● 在项目树中打开“向导”(Wizards)文件夹，然后双击“PWM”，或选择“PWM”并按回车键。



组态PWM向导的步骤：

1. 选择脉冲发生器

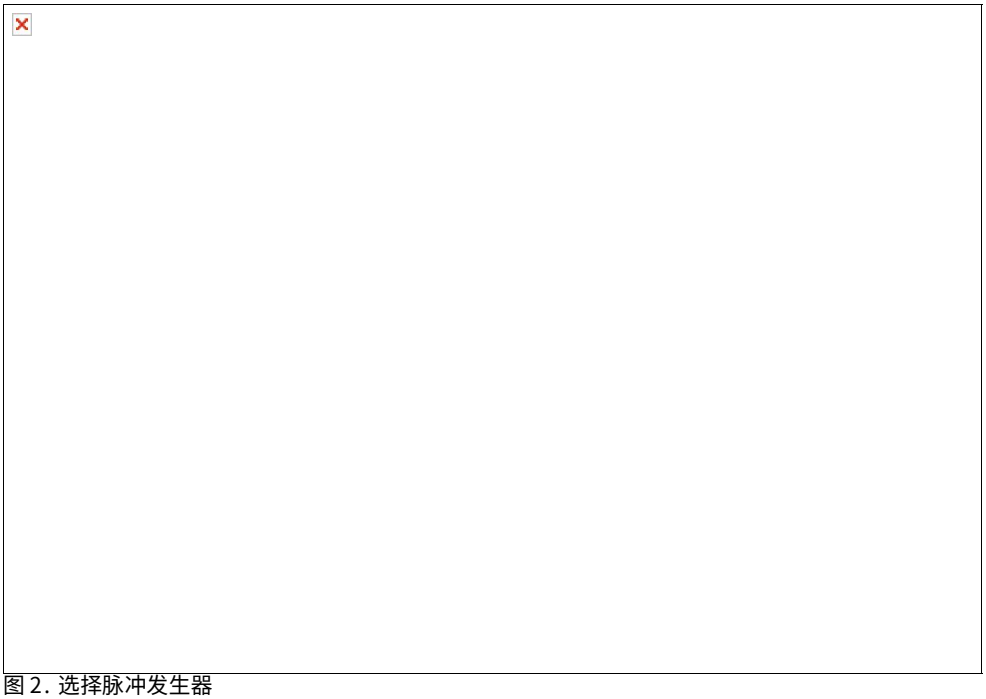


图2. 选择脉冲发生器

2. 必要时，更改 PWM通道的名称

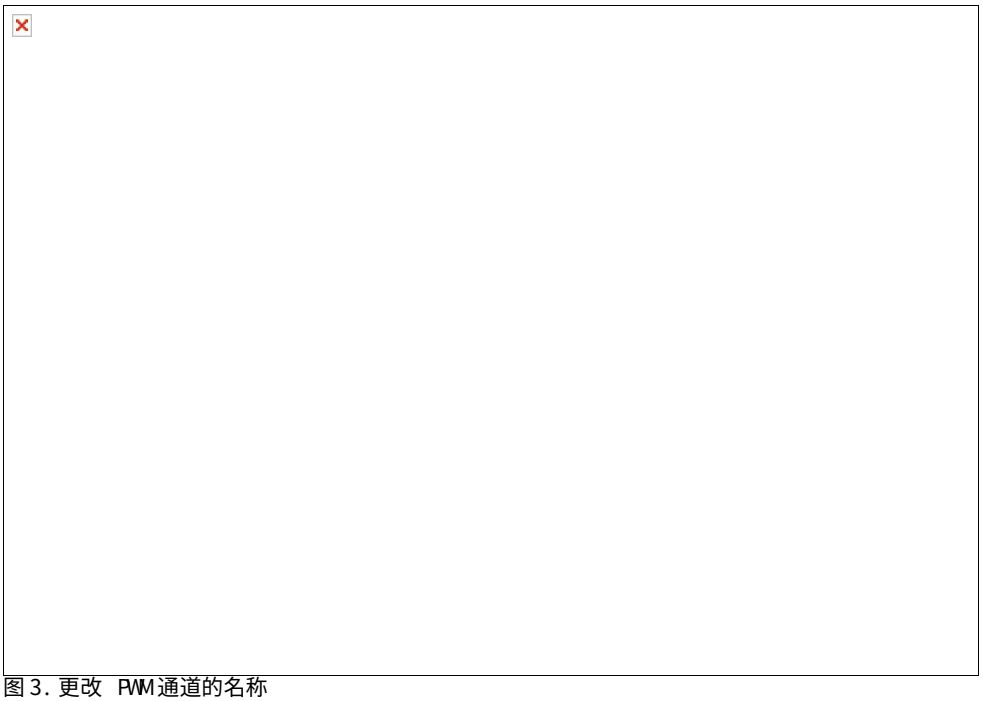


图3. 更改 PWM通道的名称

3. 组态 PWM通道输出时基

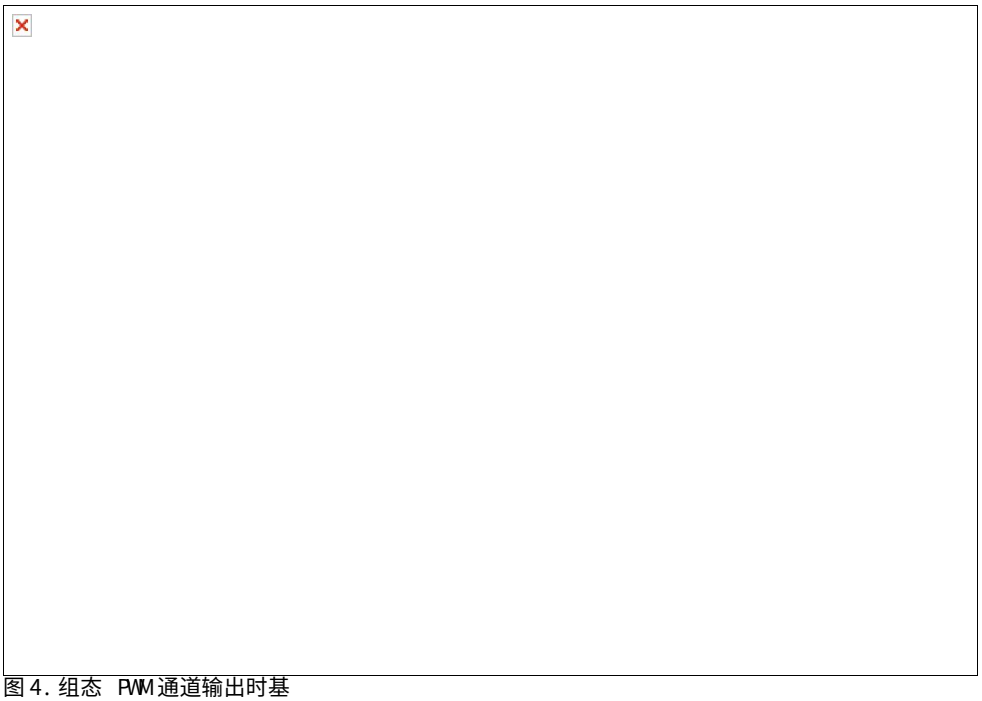


图 4. 组态 PWM 通道输出时基

4. 生成项目组件

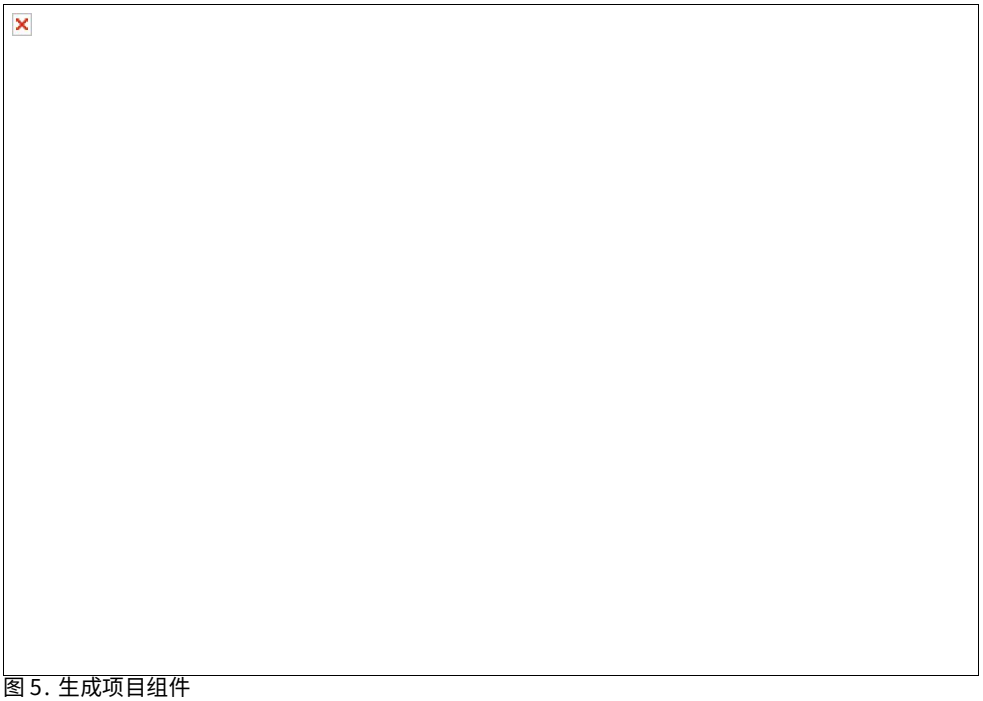


图 5. 生成项目组件

5. 使用 PWMx_RUN 子例程控制 PWM 输出的占空比

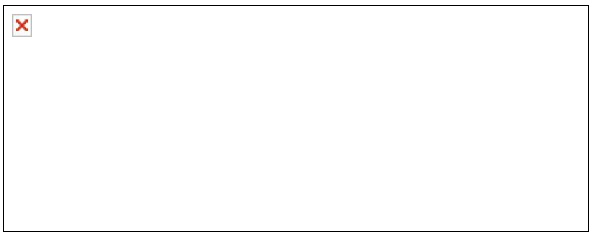
注意：

PWM 通道已硬编码：

- PWM0 已分配到 Q0.0。
- PWM1 已分配到 Q0.1。
- PWM3 已分配到 Q0.3

PWMx_RUN子程序编程

PWMx_RUN子程序允许您通过使用改变脉冲宽度（从0到周期时间的脉冲宽度）来控制输出占空比。



其中：

a. PWMx_RUN执行控制：控制脉冲发生的产生

b.Cycle:写入脉冲周期

c.Pulse:写入脉冲宽度

常问问题

使用 PWM输出功能应使用什么类型的 CPU?

应使用 24VDC晶体管输出的 CPU。如果使用继电器类型 CPU， PWM输出频率不能高于继电器响应频率，即使 PWM输出频率不过高，继电器频繁通断会影响 CPU使用寿命。

PWM输出的幅值是多少？

PWM 输出的幅值为 24V（高电平有效，共负端连接），若想实现输出其他电压的幅值，需自己加转换器来实现。 S7-200 SMART CPU的高速输出点所在的数字量输出点可以支持 20.4 － 28.8V 电压幅值。

如何强制停止 PWM 输出？

可以通过编程将控制字节中的使能位 SM67.7 、 SM77.7和 SM567.7 清零，然后执行 PLS 指令，便可立即停止 PWM0、 PWM1和 PWM2输出。

PWM输出周期和脉宽有哪些限制？

因为限制 PWM输出的因素有两个：

- 硬件输出电路响应速度的限制，对于Q0.0、 Q0.1和Q0.3 从断开到接通为 1.0μs，从接通到断开 3.0μs，因此最小脉宽不可能小于 4.0μs。
- 最大的频率为 100K，因此最小周期为 10μs。

不论是连续脉冲，还是相对较长周期内的单个脉冲，其脉冲宽度限制都是相同的。

如何改变 PWM输出的周期 /脉冲宽度？

PWM功能可以在初始化时设置脉冲的周期和宽度，也可以在连续输出脉冲时很快地改变上述参数。可以通过使用写入 SM特殊寄存器和使用 PWM向导两种方法更改 PWM输出的周期和脉宽。

使用写入 SM寄存器方式的操作步骤为：

1. 设置控制字节，以允许写入（或者更新）相应的参数
2. 将相应的特殊存储器写入新的周期 /脉宽值
3. 执行 PLS指令，对 PWM发生器进行硬件设置变更

使用 PWM向导方法为：调用 PWMx_RUN子程序，将周期值写入 Cycle管脚，将脉宽值写入 Pulse管脚。

开环运动控制

S7-200 SMART CPU 提供运动轴（Axis of Motion）控制：内置于 CPU，用于速度和位置控制。

CPU 本体集成的三个数字量输出点（Q0.0、 Q0.1、 Q0.3）可通过 Micro/Win SMART 软件中的运动控制向导组态运动控制输出。

注意：高速脉冲输出操作只能在晶体管输出的 CPU 上进行，不可以在继电器输出的 CPU 上进行。

运动轴（Axis of Motion）

内置于 S7-200 SMART CPU 的运动控制功能使用运动轴（Axis of Motion）进行步进电机和伺服电机的速度和位置控制。

S7-200 SMART 提供 3个轴的开环位置控制所需要的功能和性能：

1. 提供高速控制，速度从每秒 2个脉冲到每秒 100,000个脉冲（2HZ到 100KHZ）；
2. 提供可组态的测量系统，既可以使用工程单位（例如英寸和厘米）也可以使用脉冲数；

3. 提供可组态的反冲补偿；
4. 支持绝对、相对和手动位控方式；
5. 提供连续操作；
6. 提供多达 32 组移动曲线，每组最多可有 16 步；
7. 提供 4 种不同的参考点寻找模式，每种模式都可对起始的寻找方向和最终的接近方向进行选择。

S7-200 SMART CPU 运动控制输入/输出点定义见表 1：

表 1 运动控制输入/输出定义

类型	信号	描述	CPU 本体 I/O 分配		
输入	STP	STP 输入可让 CPU 停止脉冲输出。在位控向导中可选择您所需要的 STP 操作。	在位控向导中可被组态为 I0.0-I0.7，I1.0-I1.3 中的任意一个，但是同一个输入点不能被重复定义		
	RPS	RPS（参考点）输入可为绝对运动操作建立参考点或零点位置。			
	LMT+	LMT+和 LMT-是运动位置的最大限制。位控向导中可以组态 LMT+和 LMT-输入。			
	LMT-				
	ZP(HSC)	ZP（零脉冲）输入可帮助建立参考点或零点位置。通常，电机驱动器/放大器在电机的每一转产生一个 ZP 脉冲	CPU 本体高速计数器输入可被组态为 ZP 输入 HSC0（I0.0） HSC1（I0.1） HSC2（I0.2） HSC3（I0.3）		
输出			Axis0	Axis1	Axis2
	P0	P0 和 P1 是源型晶体管输出，用以控制电机的运动和方向。	Q0.0	Q0.1	Q0.3
	P1		Q0.2	Q0.7 or Q0.3*	Q1.0
	DIS	DIS 是一个源型输出，用来禁止或使能电机驱动器/放大器。	Q0.4	Q0.5	Q0.6

*如果 Axis1 组态为脉冲加方向，则 P1 分配到 Q0.7。如果 Axis1 组态为双向输出或者 A/B 相输出，则 P1 被分配到 Q0.3，但此时 Axis2 将不能使用。

Micro/WIN SMART 为运动控制提供了方便快捷的工具，遵守以下步骤即可：

1. 组态 Axis of Motion：
   Micro/WIN SMART 提供了运动控制向导，可生成组态/曲线表和位控指令。

   详细信息参看：位置控制向导

2. 测试 Axis of Motion：
   Micro/WIN SMART 提供一个运动控制面板，用以测试输入输出的接线、Axis of Motion 的组态以及运动曲线的运行。

   详细信息参看：运动控制面板

3. 创建 CPU 执行程序：
   位控向导自动生成位控指令。您可以将这些指令插入您的程序中。将以下指令插入您的用户程序当中：
   - 要使能位控模块，请插入一个 AXISx_CTRL 指令。用 SM0.0（始终接通）以确保这条指令在每一个循环周期中都得到执行。
   - 要将电机移动到一个指定位置，使用一条 AXISx_GOTO 指令或一条 AXISx_RUN 指令。AXISx_GOTO 指令使电机运动到您在程序中输入的指定位置。AXISx_RUN 指令则使电机按照您在位控向导中所组态的路线运动。
   - 要使用绝对坐标进行运动，您必须为您的应用建立零位置。使用一条 AXISx_RSEEK 或一条 AXISx_LDPOS 指令建立零位置。
   - 位控向导生成的其它指令为典型应用提供需要的功能，对于您的特定应用，这些指令是可选的。

   详细信息参看：程序编制

4. 编译您的程序并将系统块、数据块和程序块下载到 S7-200 SMART CPU 中。

运动控制向导

1. 打开“运动控制”向导，“工具”->“向导”->“运动控制”

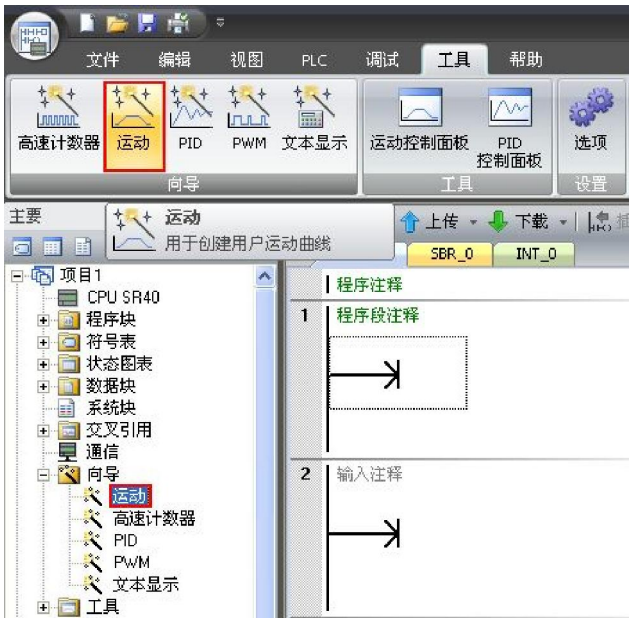


图 1打开“运动控制”向导

2. 选择需要配置的轴

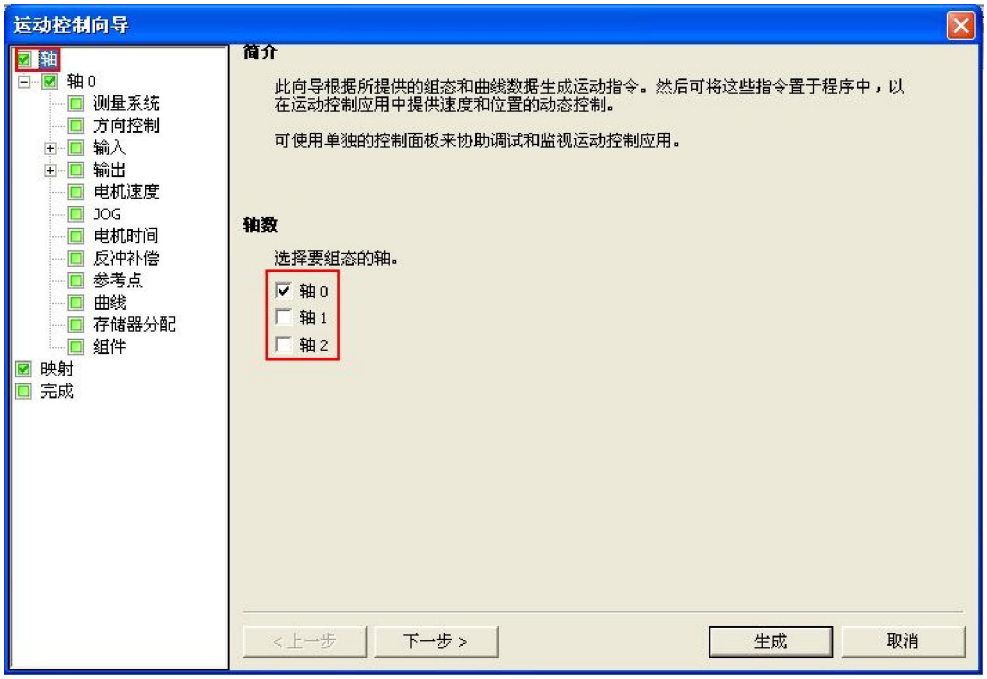


图 2选择需要配置的轴

3. 为所选择的轴命名

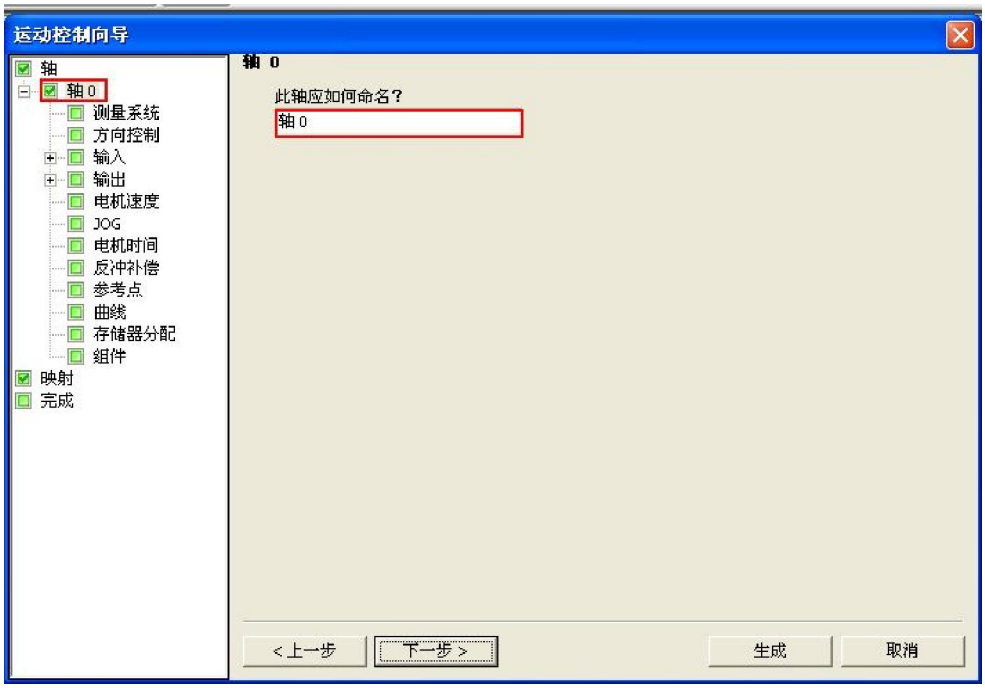


图 3.为所选择的轴命名

4. 输入系统的测量系统（“ 工程量 ”或者“ 脉冲数 /转 ”）

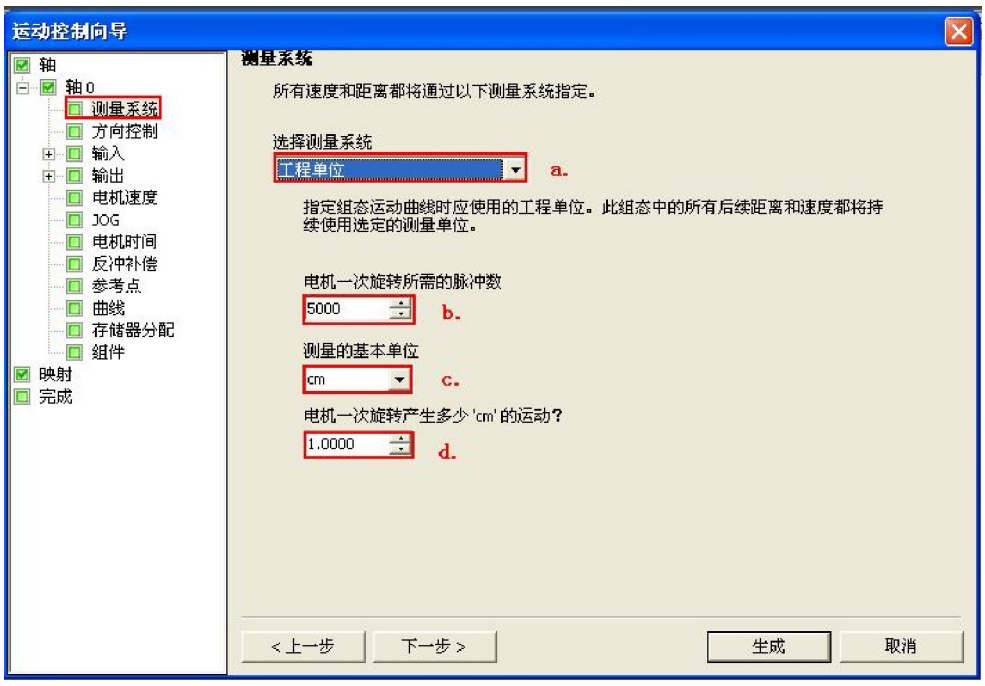


图 4.选择测量系统

a.选择工程单位或者是脉冲数；
b.选择电机每转脉冲数；
c.选择基本单位；
d.输入电机每转运行距离

5. 设置脉冲方向输出

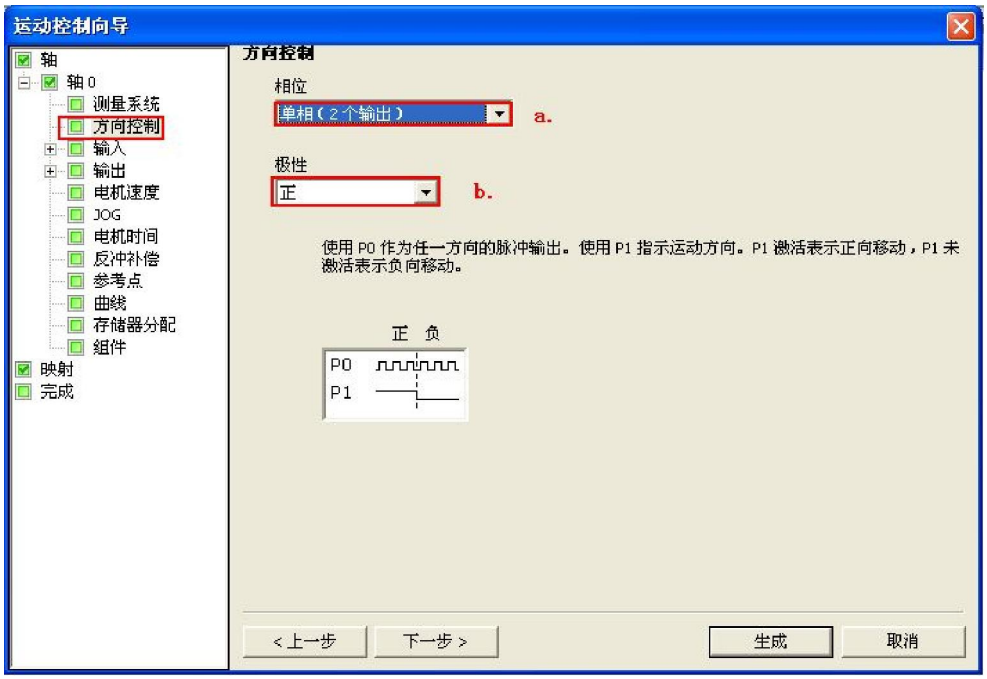


图 5.设置脉冲方向输出

a.设置有几路脉冲输出（单相： 1路、双向： 2路、正交： 2路）；
b.设置脉冲输出极性和控制方向。

6. 分配输入点

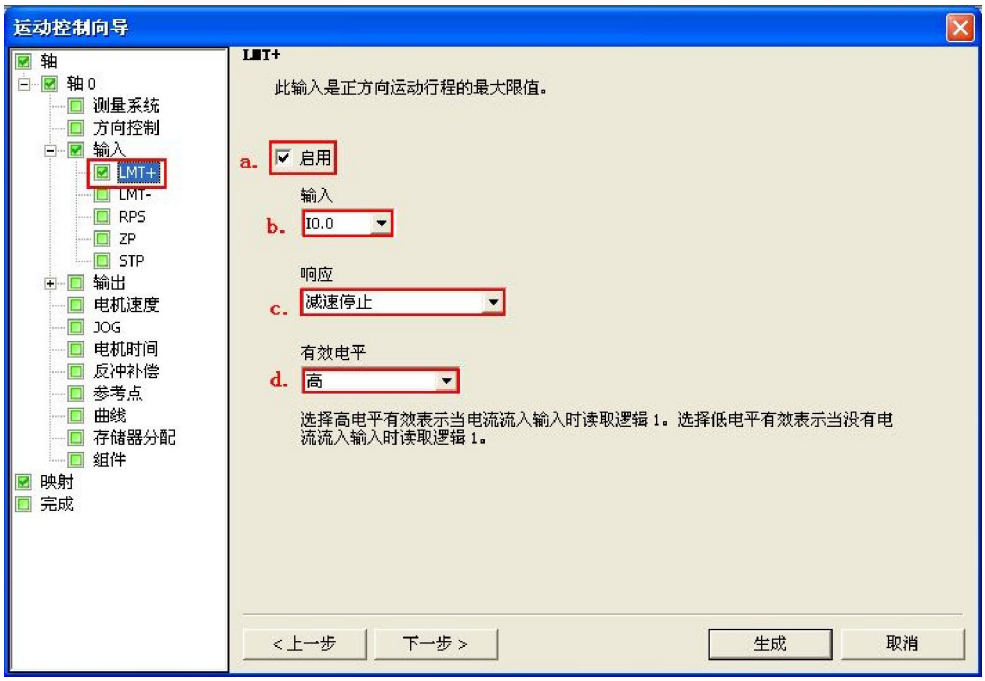


图 6.配置正限位输入点

a.正限位使能；
b.正限位输入点；
c.指定相应输入点有效时的响应方式；
d.指定输入信号有效电平（低电平有效或者高电平有效）。

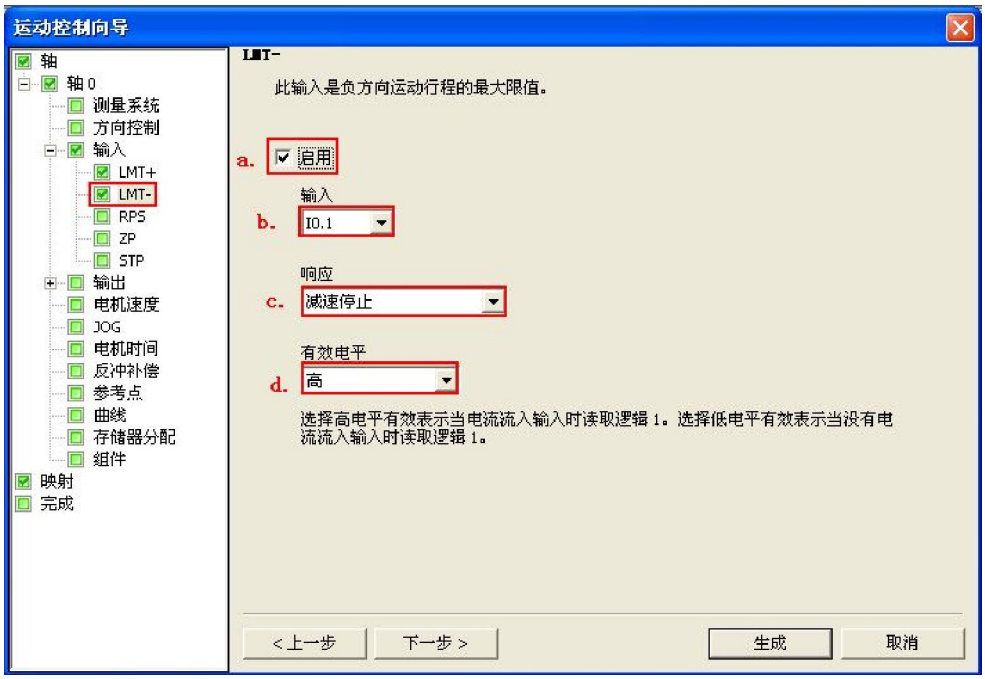


图 7配置负限位输入点

a.负限位使能；
b.负限位输入点；
c.指定相应输入点有效时的响应方式；
d.指定输入信号有效电平（低电平有效或者高电平有效）。

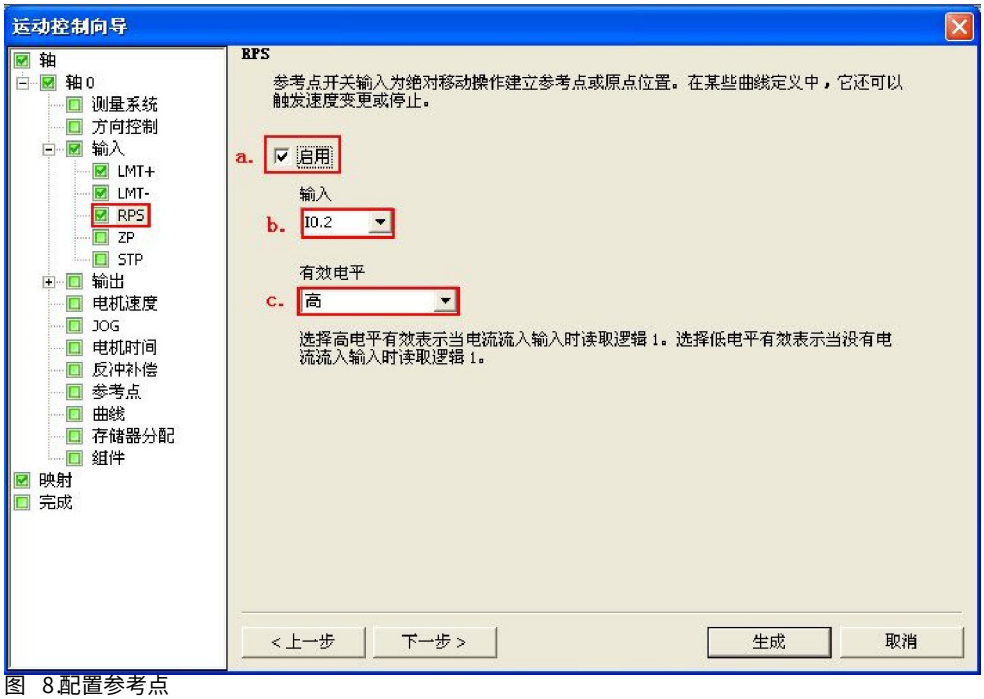


图 8配置参考点

a.使能参考点；
b.参考点输入点；
c.指定输入信号有效电平（低电平有效或者高电平有效）。

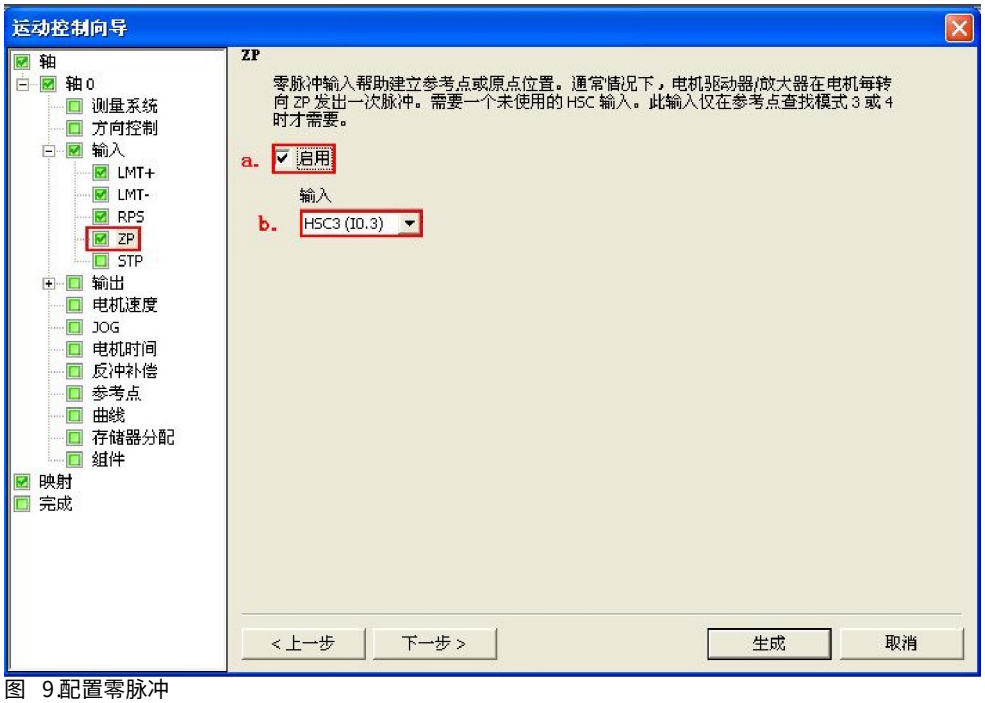


图 9.配置零脉冲

a.使能零脉冲：
b.零脉冲输入点。

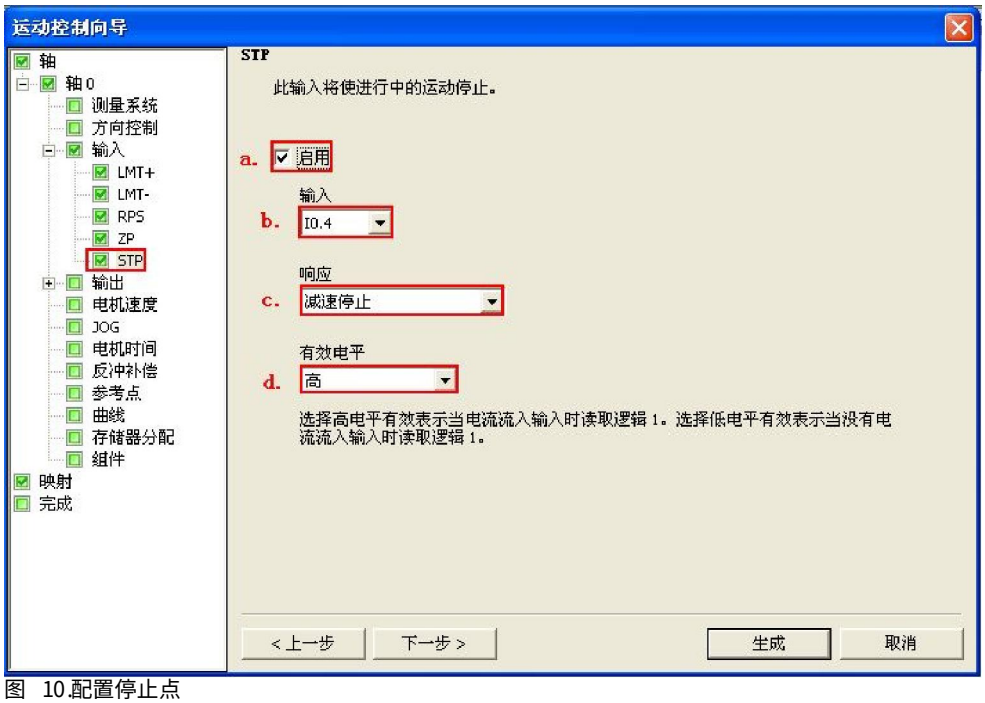


图 10.配置停止点

a.使能停止点；
b.停止输入点；
c.指定相应输入点有效时的响应方式；
d.指定输入信号有效电平（低电平有效或者高电平有效）。

注意：关于输入点的分配与定义请参看 运动控制输入 /输出定义

7. 定义输出点

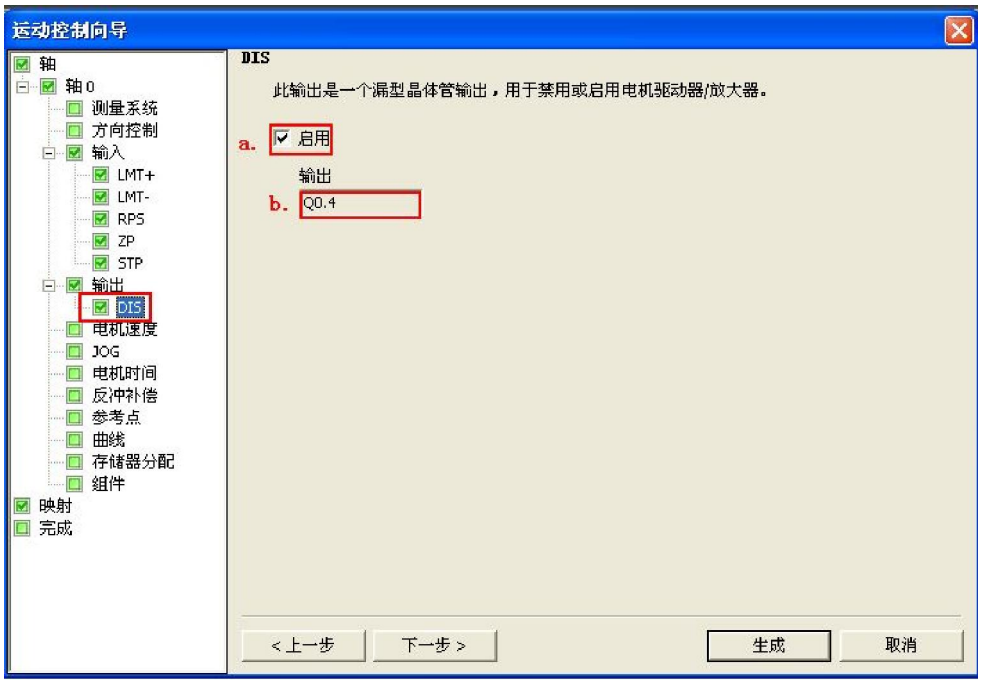


图 11.定义输出点

注意：每个轴的输出点都是固定的用户不能对其进行修改，但是可以选择使能 /不使能 DIS。
关于输出点的定义请参看 ➪运动控制输入 /输出定义

8. 定义电机的速度

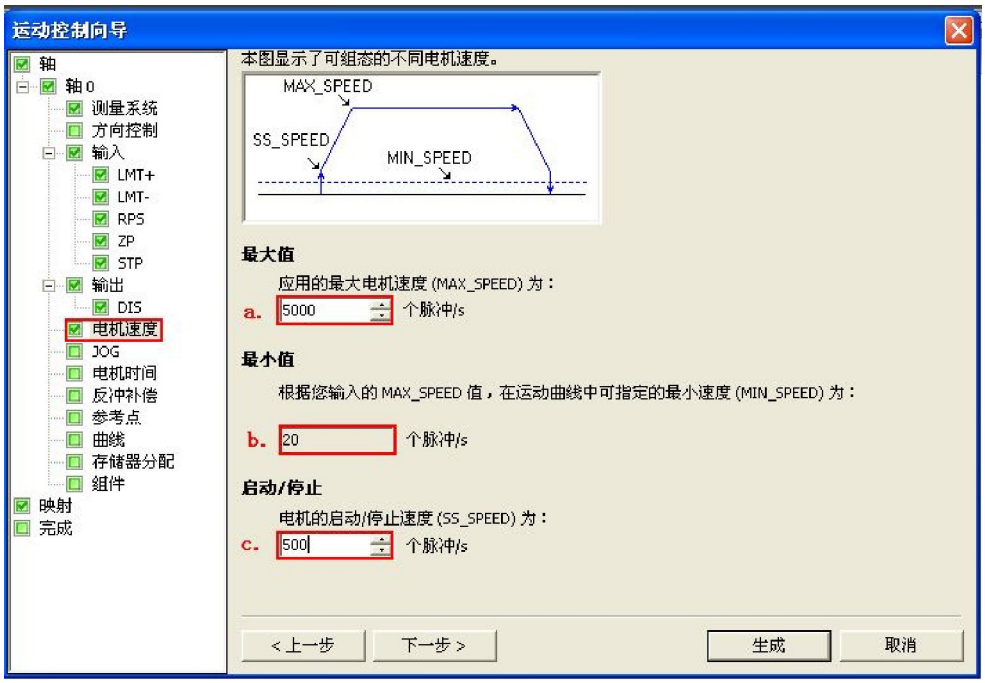


图 12.定义电机的速度

a.定义电机运动的最大速度“ MAX_SPEED”；
b.根据定义的最大速度，在运动曲线中可以指定的最小速度；
c.定义电机运动的启动 /停止速度“ SS_SPEED” 。

9. 定义点动参数

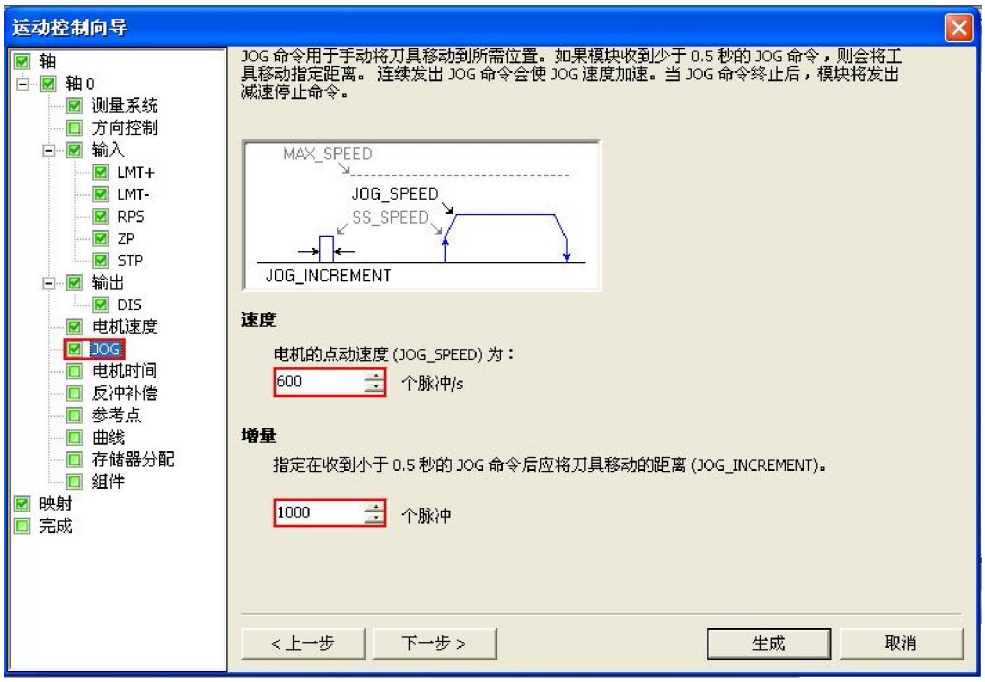


图 13.定义点动参数

a.定义点动速度“ JOG_SPEED”（电机的点动速度是点动命令有效时能够得到的最大速度）；
b.定义点动位移“ JOG_INCREMENT”（点动位移是瞬间的点动命令能够将工件运动的距离）。

注意：当 CPU 收到一个点动命令后，它启动一个定时器。如果点动命令在 0.5秒到时之前结束， CPU 则以定义的 SS_SPEED速度将工件运动 JOG_INCREMENT数值指定的距离。当 0.5秒到时时，点动命令仍然是激活的， CPU 加速至 JOG_SPEED速度。继续运动直至点动命令结束 ,随后减速停止。

10. 加 /减速时间设置

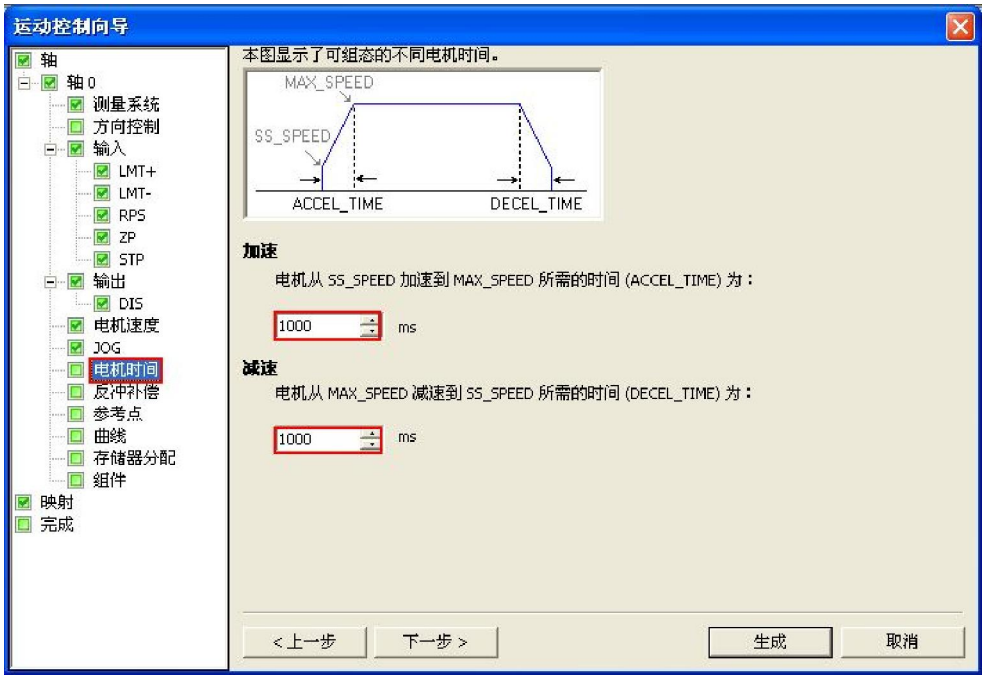


图 14.加 /减速时间设置

a.设置从启动 /停止速度“ SS_SPEED”到最大速度“ MAX_SPEED'的加速度时间“ ACCEL_TIME”；
b.设置从最大速度“ MAX_SPEED”到启动 /停止速度“ SS_SPEED”的减速度时间“ DECEL_TIME ”。

11. 定义反冲补偿

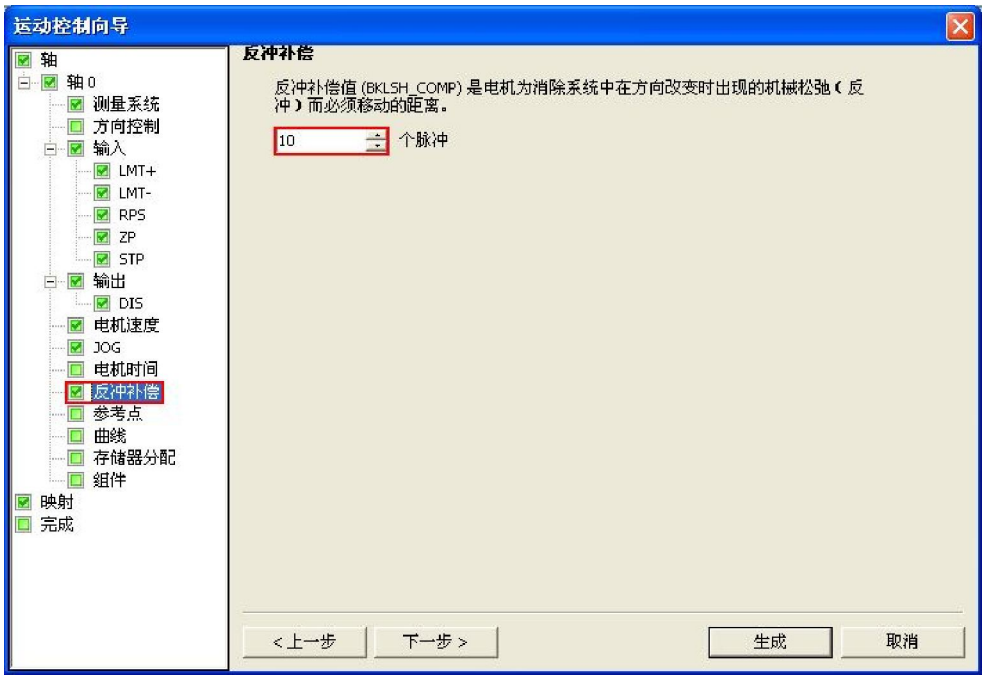


图 15.定义反冲补偿

注意：反冲补偿为当方向发生变化时，为消除系统中因机械磨损而产生的误差，电机必须运动的距离。反冲补偿总是正值。（缺省 =0）

12. 使能寻找参考点位置

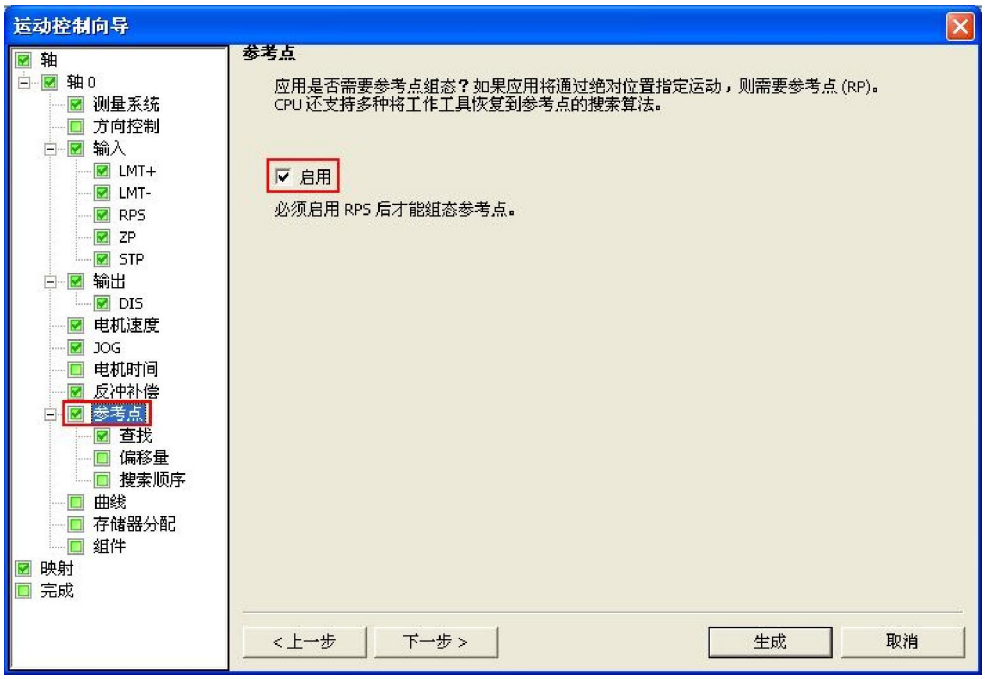


图 16. 使能寻找参考点位置

注意：若您的应用需要从一个绝对位置处开始运动或以绝对位置作为参考，您必须建立一个参考点（RP）或零点位置，该点将位置测量固定到物理系统的一个已知点上。

13. 设置寻找参考点位置参数

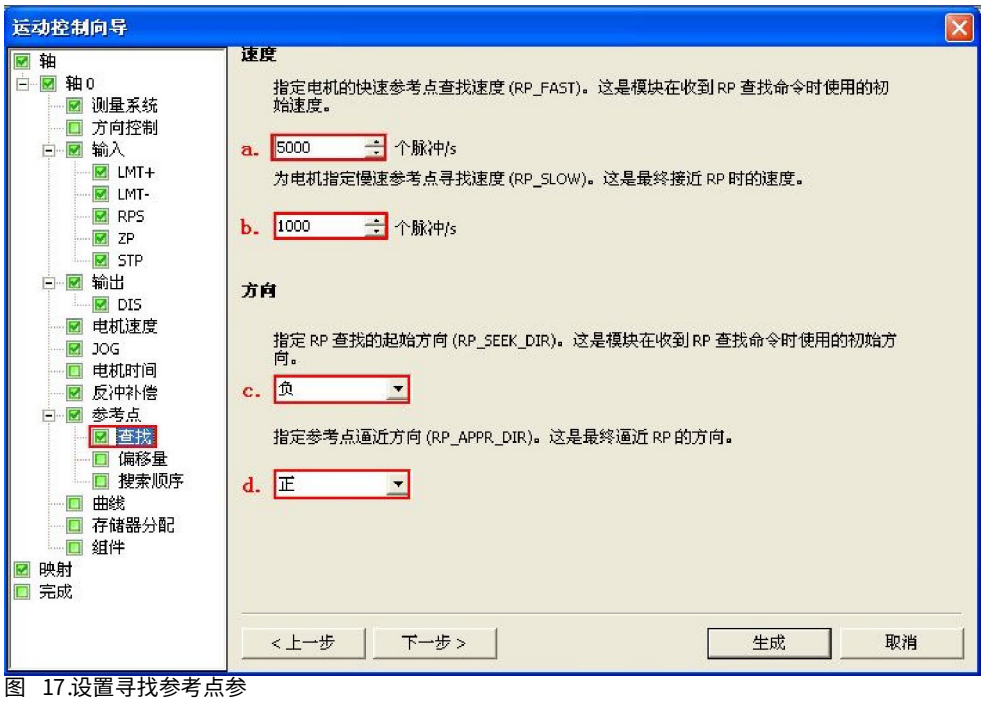


图 17.设置寻找参考点参

a.定义快速寻找速度“ RP_FAST”（快速寻找速度是模块执行 RP寻找命令的初始速度，通常 RP_FAST是 MAX_SPEED的 2/3左右）；
b.定义慢速寻找速度“ RP_SLOW”（慢速寻找速度是接近 RP的最终速度，通常使用一个较慢的速度去接近 RP以免错过， RP_SLOW的典型值为 SS_SPEED）；
c.定义初始寻找方向“ RP_SEEK_DIR”（初始寻找方向是 RP寻找操作的初始方向。通常，这个方向是从工作区到 RP附近。限位开关在确定 RP的寻找区域时扮演重要角色。当执行 RP寻找操作时，遇到限位开关会引起方向反转，使寻找能够继续下去，默认方向 =反向）；
d.定义最终参考点接近方向“ RP_APPR_DIR”，（最终参考点接近方向是为了减小反冲和提供更高的精度，应该按照从 RP移动到工作区所使用的方向来接近参考点，默认方向 =正向）。

14. 设置参考点偏移量

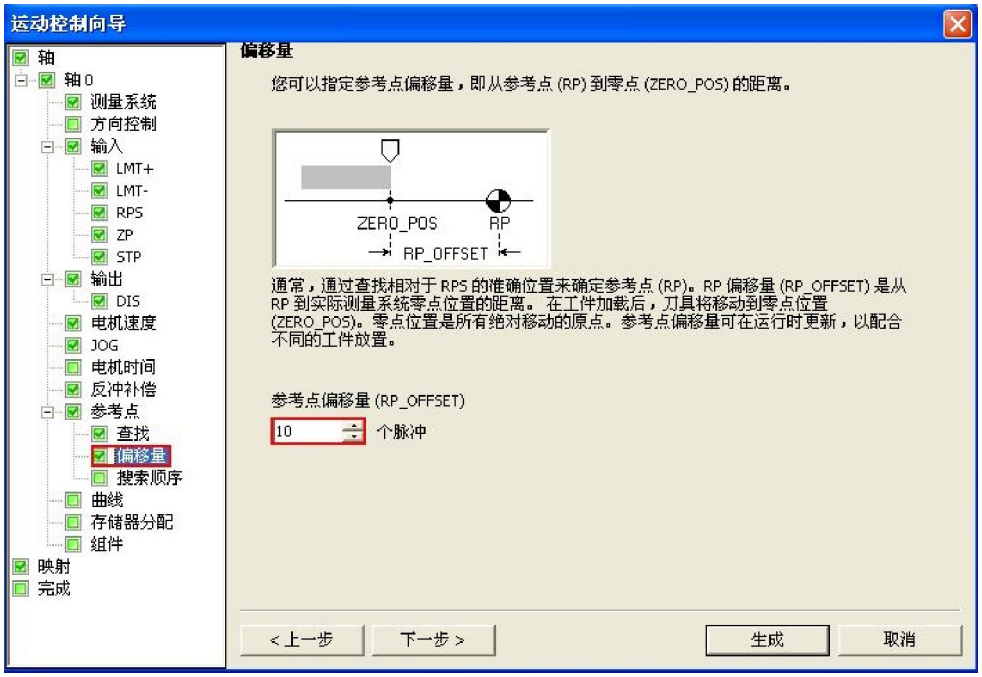


图 18.设置参考点偏移量

注意：参考点偏移量“ RP_OFFSET是 ”在物理的测量系统中 RP到零位置之间的距离，缺省 =0。

15. 设置寻找参考点顺序

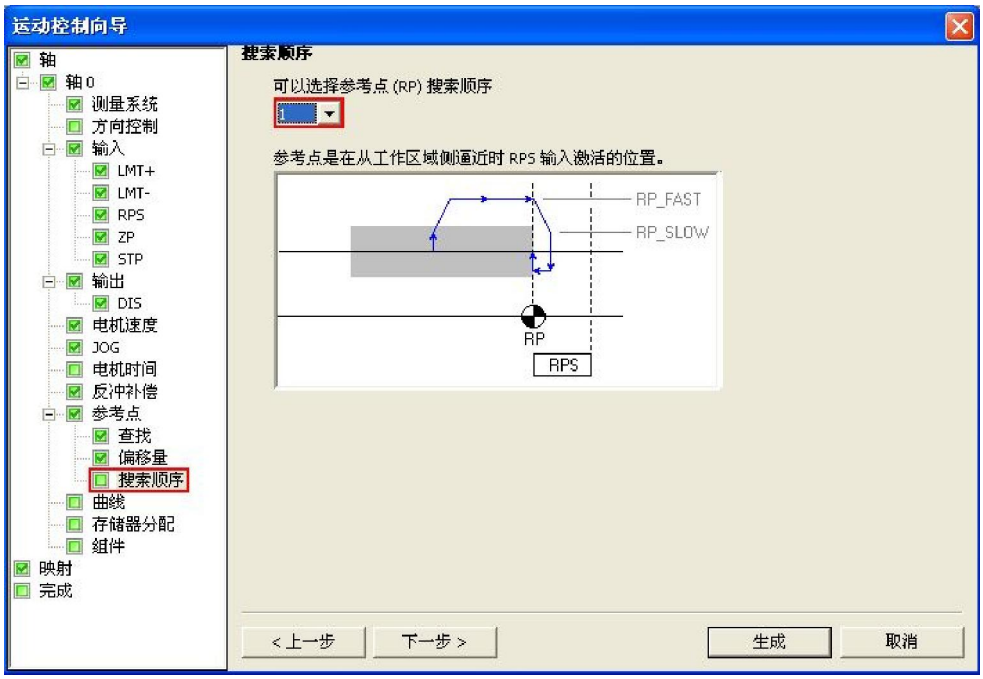


图 19.设置寻找参考点顺序

S7-200 SMART 提供 4 中寻找参考点顺序模式，每种模式定义如下：
RP 寻找模式 1：RP 位于 RPS 输入有效区接近工作区的一边开始有效的位置上；
RP 寻找模式 2：RP 位于 RPS 输入有效区的中央；
RP 寻找模式 3：RP 位于 RPS 输入有效区之外，需要指定在 RPS 失效之后应接收多少个 ZP（零脉冲）输入；
RP 寻找模式 4：RP 通常位于 RPS 输入的有效区内，需要指定在 RPS 激活后应接收多少个 ZP（零脉冲）输入。

关于寻找参考点模式的详细信息请参考：
➪《S7-200 SMART 系统手册》

16. 新建运动曲线并命名

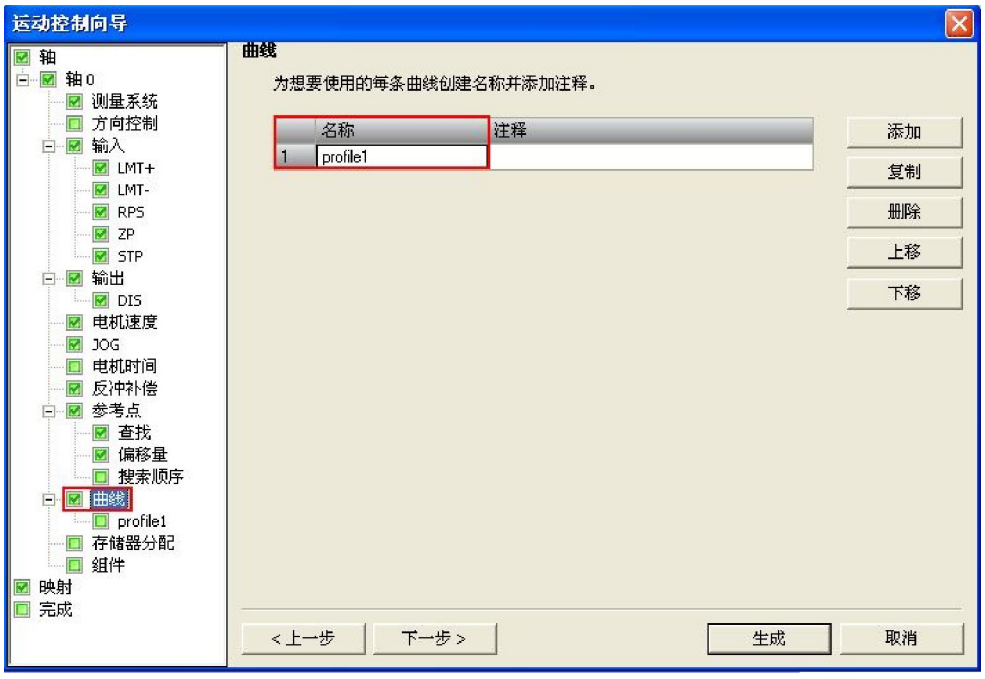


图 20.新建运动曲线并命名

通过点击“添加（Add）”按钮添加移动曲线并命名。

注意：S7-200 SMART 支持最多 32 组移动曲线。运动控制向导提供移动曲线定义，在这里，您可以为您的应用程序定义每一个移动曲线。运动控制向导中可以为每个移动曲线定义一个符号名，其做法是您在定义曲线时输入一个符号名即可。

17. 定义运动曲线

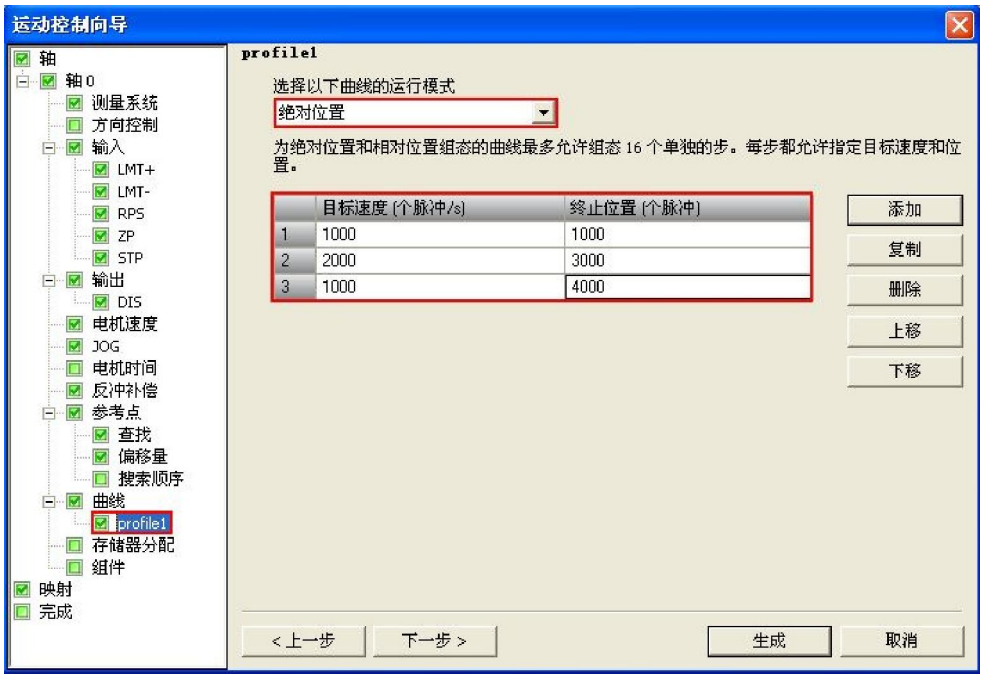


图 21.定义运动曲线

a.选择移动曲线的操作模式（支持四种操作模式：绝对位置、相对位置、单速连续旋转、两速连续转动）；
b.定义该移动曲线每一段的速度和位置（S7-200 SMART 每组移动曲线支持最多 16步）。

18. 为配置分配存储区

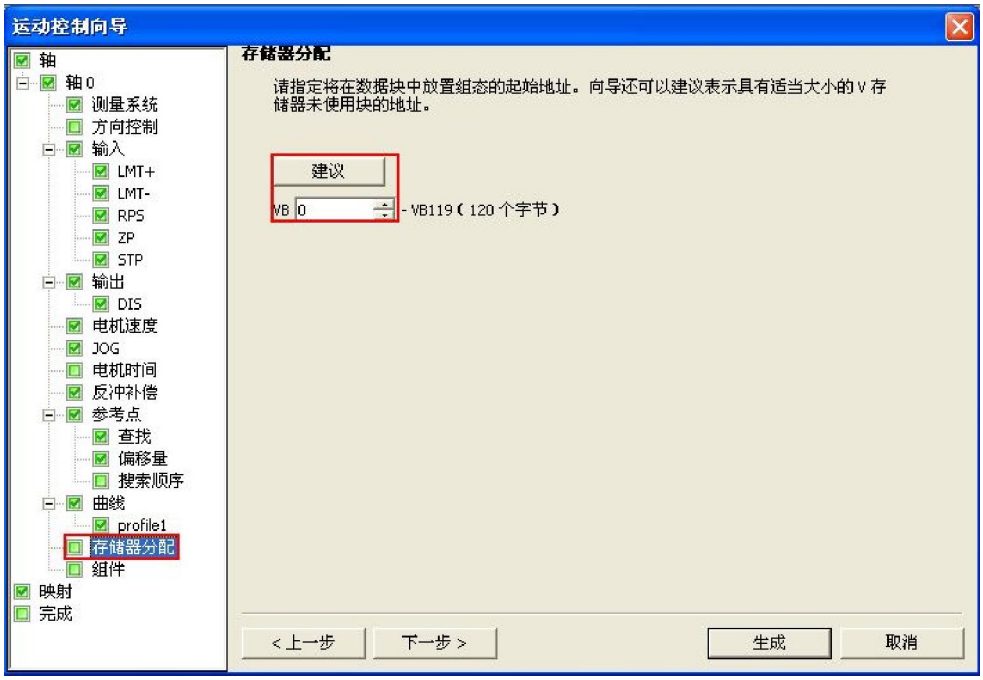


图 22.为配置分配存储区

通过点击“ 建议（ Suggest） ”按钮分配存储区

⚠注意：程序中其他部分不能占用该向导分配的存储区。

19. 完成组态

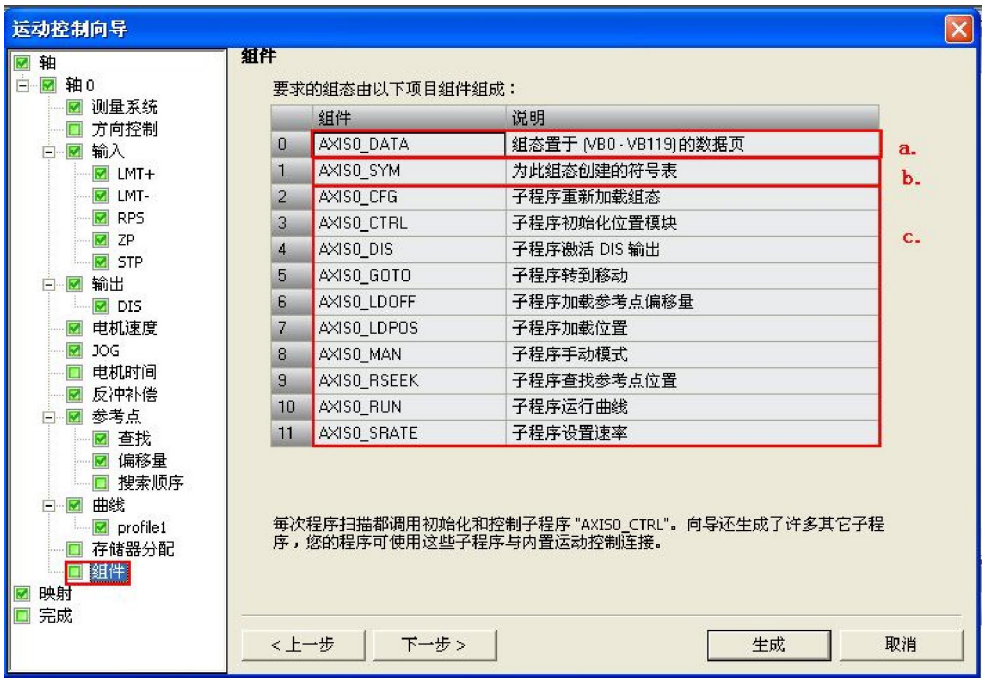


图 23.向导生成的组件

当您完成对运动控制向导的组态时，只需点击生成（Generate），然后运动控制向导会执行以下任务：
a. 将组态和曲线表插入到您的 S7-200 SMART CPU 的数据块（AXISx_DATA）中；
b. 为运动控制参数生成一个全局符号表（AXISx_SYM）；
c. 在项目的程序块中增加运动控制指令子程序，您可在应用中使用这些指令；
要修改任何组态或曲线信息，您可以再次运行运动控制向导。

⚠注意：由于运动控制向导修改了程序块、数据块和系统块，要确保这三种块都下载到 S7-200 SMART CPU 中。否则，CPU 可能会无法得到操作所需的所有程序组件。

20. 查看输入输出点分配

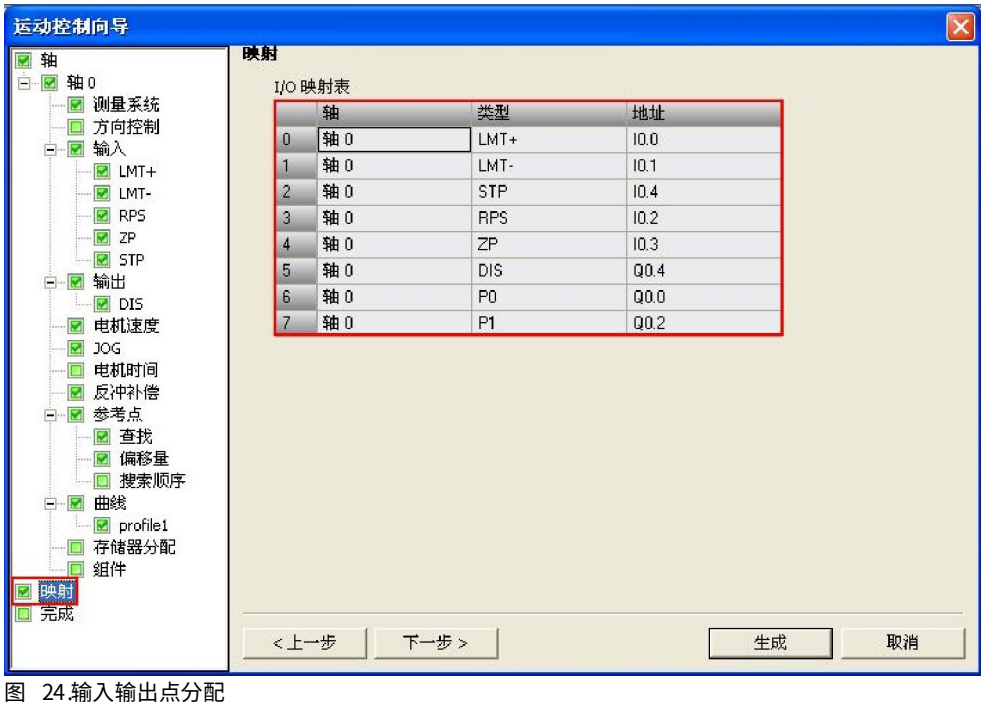


图 24.输入输出点分配

完成配置后运动控制向导会显示运动控制功能所占用的 CPU 本体输入输出点的情况。

使用运动控制面板进行调试

为了帮助用户更好的开发 S7-200 SMART 的运动控制功能，STEP-7 Micro/WIN SMART 提供了一个调试界面“运动控制面板”。通过“操作界面、配置参数界面和配置曲线参数界面”，可以帮助用户方便地调试、操作和监视 S7-200 SMART 的工作状态，验证控制系统

接线是否正确，调整配置运动控制参数，测试每一个预定义的运动轨迹曲线。

⚠ 注意：使用运动控制面板之前请确保已经完成以下操作：

a.将运动控制向导生成的所有组件（包括程序块、数据块和系统块）下载到 CPU 中。否则 CPU 无法得到操作所需要的有效程序组件；
b.将 CPU 的运行状态设置为“ STOP”。

步骤一：打开运动控制面板。

1. 通过菜单栏或者左侧树形目录打开“运动控制面板”

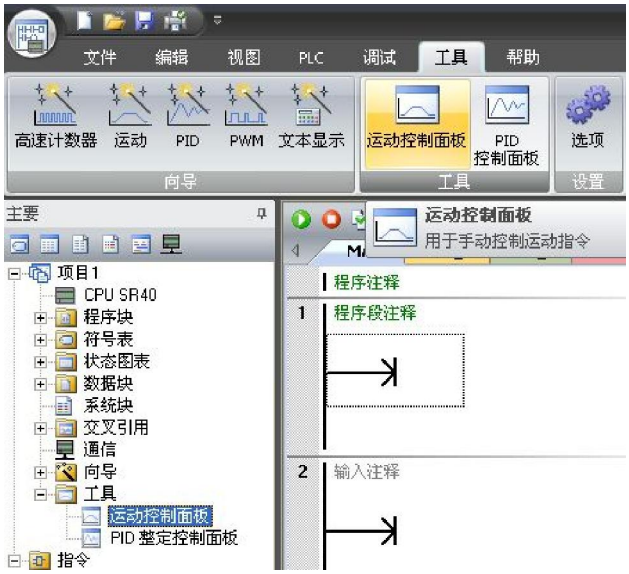


图 1.打开“运动控制面板”

2. 点击“运动控制面板”按钮，会弹出一个对话框，其作用是比较 STEP-7 Micro/WIN SMART 当前打开的程序与 CPU 中的程序是否一致（如图 2所示）。当程序比较通过后点击“继续”按钮（若未通过请重新下载程序块、数据库和系统块至 CPU）。

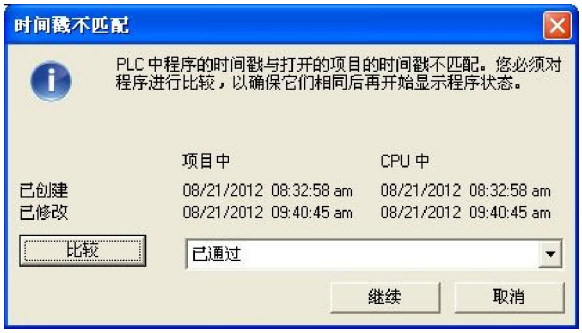


图 2.程序比较对话框

步骤二：在“操作”界面中监视和控制运动轴

“操作”界面允许用户以交互的方式，非常方便地操作、控制运动轴。该界面友好地显示当前设备运行速度、位置和方向信息，监控到输入、输出点状态信息。“操作”界面如图 3所示：

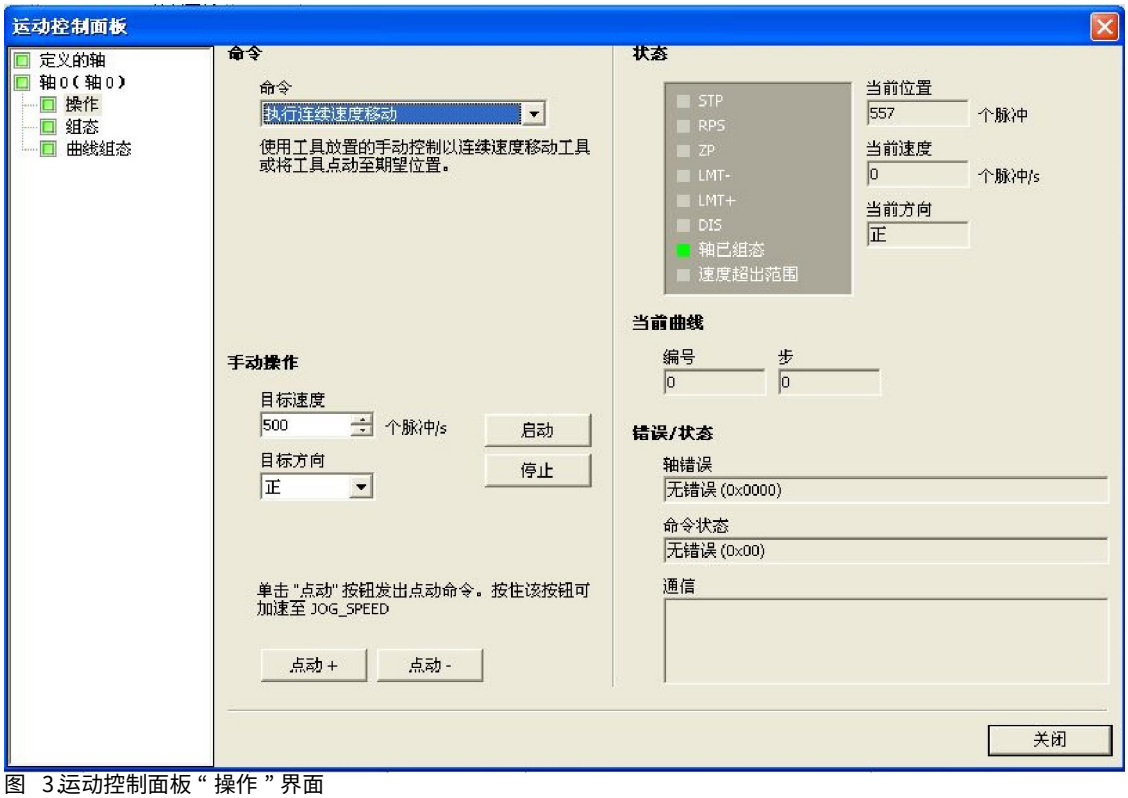


图 3.运动控制面板“ 操作 ”界面

1. 选择“ 激活 DIS输出 ”， 点击“ 执行 ”，使能电机驱动器 .

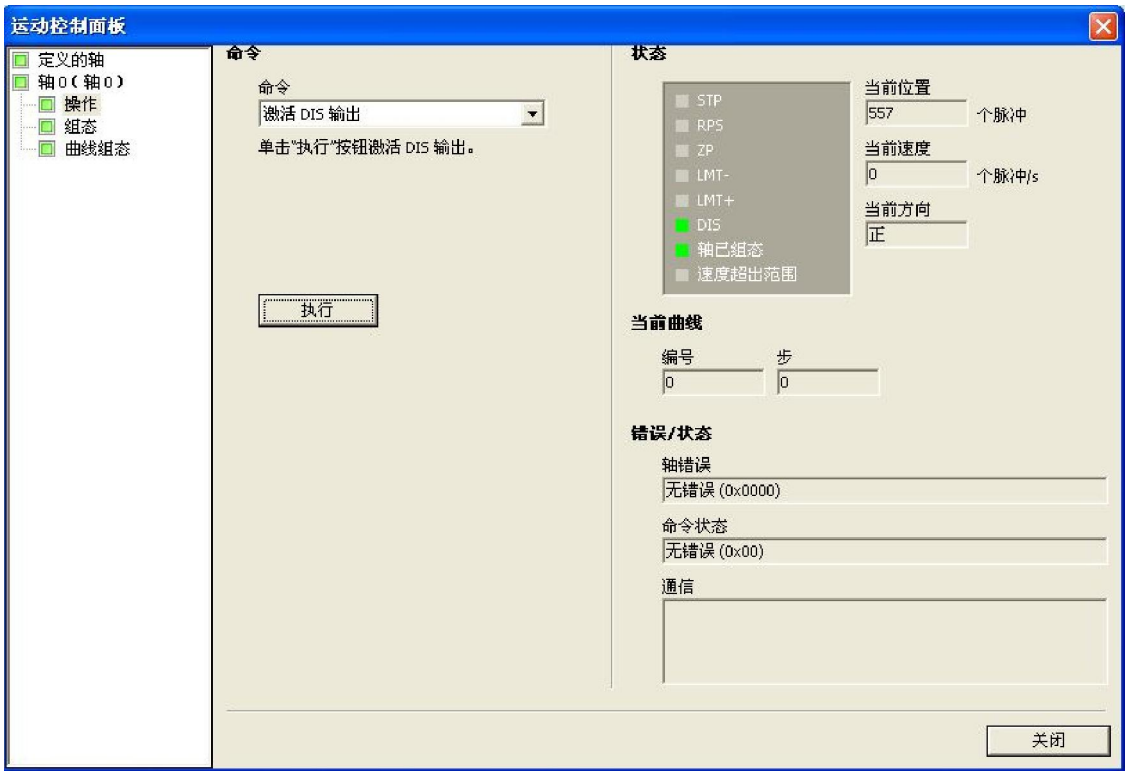


图 4.激活“ DIS”输出

2. 选择“ 执行连续速度移动 ”，可以使电机连续运转 .

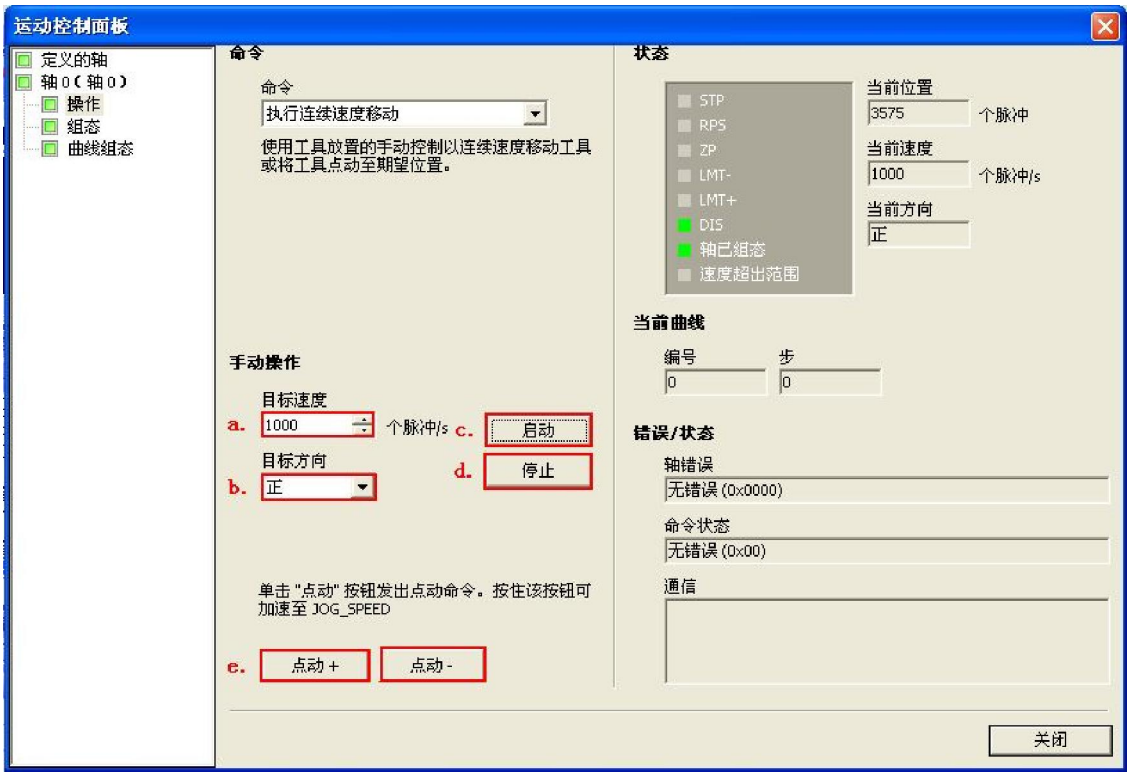


图 5.连续速度运转指令

a.输入目标速度；
b.输入目标方向；
c.点击“启动”，执行连续速度运转指令；
d.点击“停止”，终止连续速度运转指令；
e.点击“点动 +”按钮执行正向点动命令，点击“点动 -”按钮执行负向点动命令，点击时间超过 0.5 秒电机会加速到点动速度（JOG_SPEED）。

3. 选择“查找参考点”，点击“执行”，可以完成寻找机械坐标系参考点的操作。

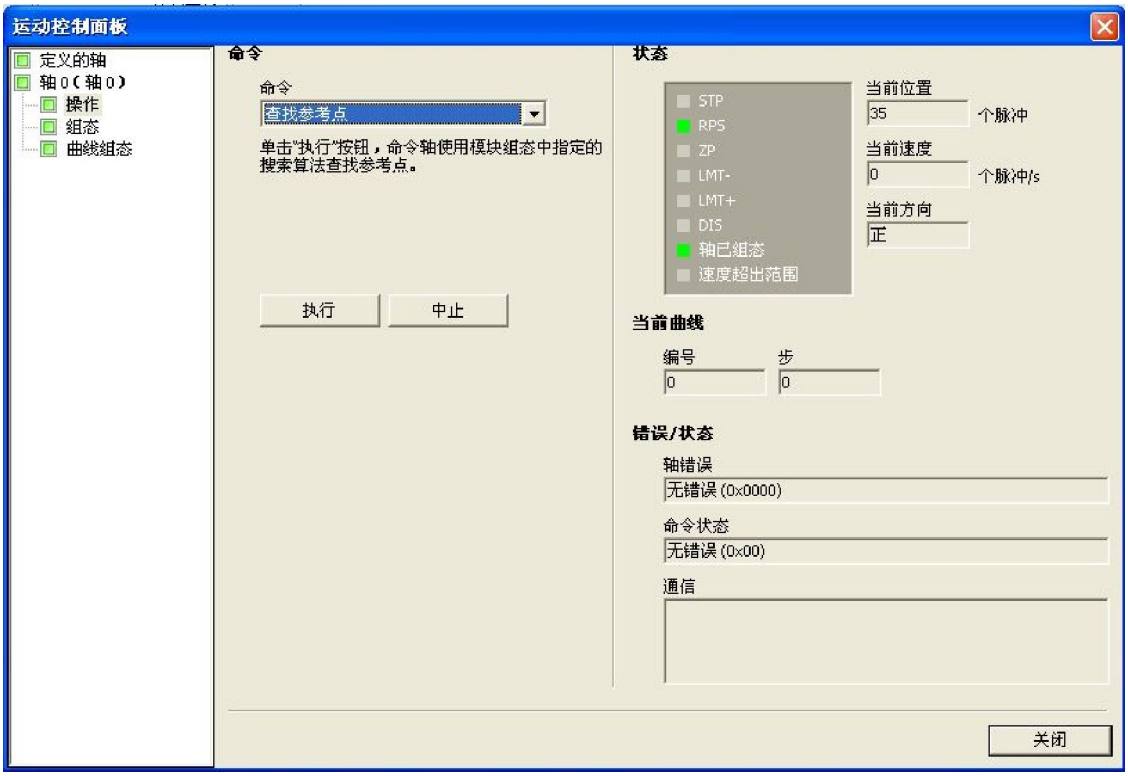


图 6.寻找参考点指令

4. 选择“执行曲线”，可以完成配制运动轨迹曲线的操作。

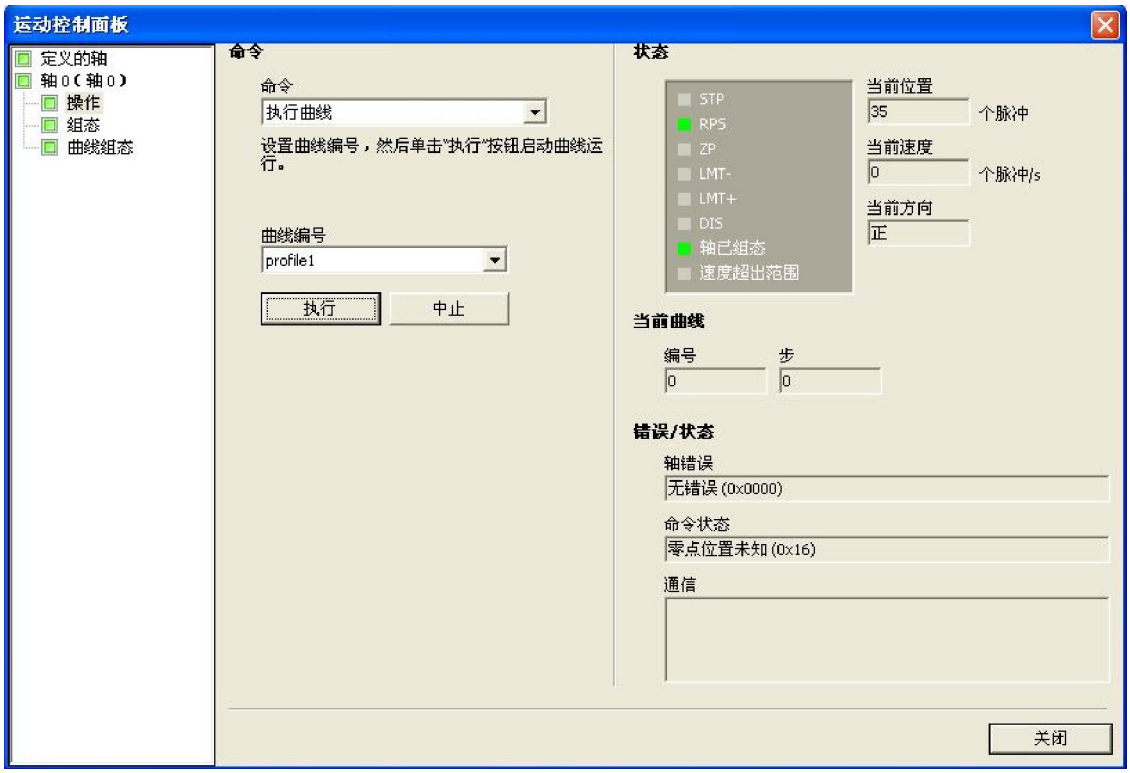


图 7.执行运动曲线指令

通过下拉列表选择已组态曲线的符号名，点击“ 执行 ”，运行指定曲线。

步骤三：在“ 组态 ”界面中显示、修改运动控制参数

在“ 组态 ”界面中，可以帮助用户方便地监控、修改存储在 S7- 200 CPU 数据块中的配置参数信息。 修改过组态设置以后，只需要先点击“ 允许更新 PLC中的轴组态 ” ，再点击“ 写入 ”即可。

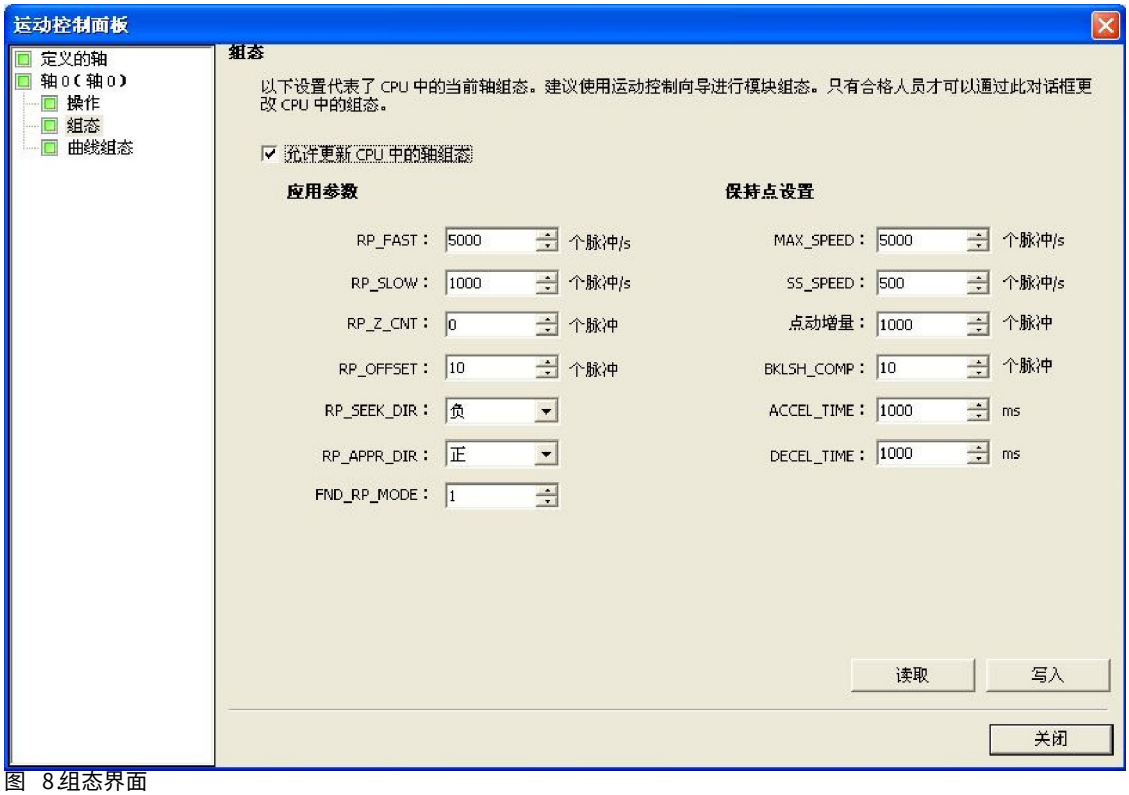


图 8.组态界面

步骤四：在“ 曲线组态 ”界面修改已组态的曲线参数并更新到 CPU中

用户可以使能更新 CPU中的轴组态功能

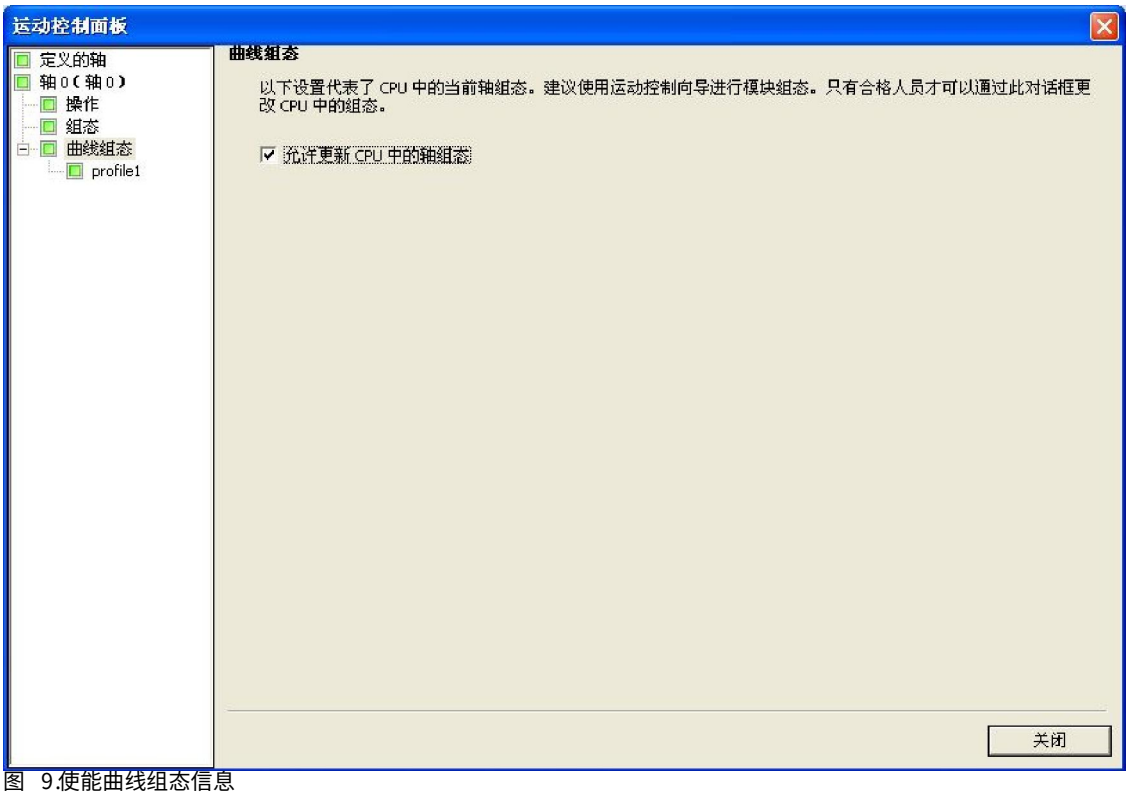


图 9.使能曲线组态信息

图 10.读取 /修改曲线信息

用户可以通过点击“ 读取 ”，读取 CPU 中存储的已组态的曲线信息，点击“ 写入 ”可以将修改后的曲线信息更新到 CPU 中。

运动控制向导为运动轴创建的指令

运动向导根据所选组态选项创建唯一的指令子程序，从而使运动轴的控制更容易。 各运动指令均具有“ AXIS*x*_”前缀，其中 *x* 代表轴通道编号。 由于每条运动指令都是一个子程序，所以 11 条运动指令使用 11 个子程序。

说明：运动指令使程序所需的存储空间增加多达 1700 个字节。 可以删除未使用的运动指令来降低所需的存储空间。 要恢复删除的运动指令，只需再次运行运动向导。

表 1.运动控制指令

指令名称	指令功能
AXISx_CTRL	启用和初始化运动轴
AXISx_MAN	手动模式
AXISx_GOTO	命令运动轴转到所需位置
AXISx_RUN	运行包络
AXISx_RSEEK	搜索参考点位置
AXISx_LDOFF	加载参考点偏移量
AXISx_LDPOS	加载位置
AXISx_SRATE	设置速率
AXISx_DIS	使能 /禁止 DIS输出
AXISx_CFG	重新加载组态
AXISx_CACHE	缓冲包络

➪ 详细的运动控制指令介绍请参考：S7-200 SMART 系统手册

运动控制指令使用准则

必须确保在同一时间仅有一条运动指令激活。

可在中断例程中执行 AXISx_RUN 和 AXISx_GOTO。 但是，如果运动轴正在处理另一命令时，不要尝试在中断例程中启动指令。 如果在中断程序中启动指令，则可使用 AXISx_CTRL 指令的输出来监视运动轴是否完成移动。

运动向导根据所选的度量系统自动组态速度参数（Speed 和 C_Speed）和位置参数（Pos 或 C_Pos）的值。 对于脉冲，这些参数为 DINT 值。 对于工程单位，这些参数是所选单位类型对应的 REAL 值。例如： 如果选择厘米 (cm)，则以厘米为单位将位置参数存储为 REAL 值并以厘米 /秒 (cm/sec) 为单位将速度参数存储为 REAL 值。

有些特定位置控制任务需要以下运动指令：

- 要在每次扫描时执行指令，请在程序中插入 AXISx_CTRL 指令并使用 SM0.0 触点。
- 要指定运动到绝对位置，必须首先使用 AXISx_RSEEK 或 AXISx_LDPOS 指令建立零位置。
- 要根据程序输入移动到特定位置，请使用 AXISx_GOTO 指令。
- 要运行通过位置控制向导组态的运动包络，请使用 AXISx_RUN 指令。

其它位置指令为可选项。

常用运动控制指令介绍

1. AXISx_CTRL
   功能：启用和初始化运动轴，方法是自动命令运动轴每次 CPU 更改为 RUN 模式时加载组态 /包络表。

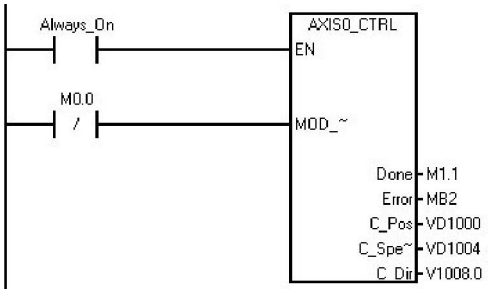


图 1. AXISx_CTRL指令

⚠ **注意：**
在您的项目中只对每条运动轴使用此子例程一次，并确保程序会在每次扫描时调用此子例程。使用 SM0.0（始终开启）作为 EN 参数的输入。

MOD_EN 参数必须开启，才能启用其它运动控制子例程向运动轴发送命令。 如果 MOD_EN 参数关闭，运动轴会中止所有正在进行的命令；
Done 戠戠参数会在运动轴完成任何一个子例程时开启；
Error 戠戠 参数存储该子程序运行时的错误代码；
C_Pos 戠 参数表示运动轴的当前位置。 根据测量单位，该值是脉冲数 (DINT) 或工程单位数 (REAL)；
C_Speed 参数提供运动轴的当前速度。 如果您针对脉冲组态运动轴的测量系统，C_Speed 是一个 DINT 数值，其中包含脉冲数 / 每秒。如果您针对工程单位组态测量系统，C_Speed 是一个 REAL 数值，其中包含选择的工程单位数 /每秒 (REAL)。
C_Dir 戠戠 参数表示电机的当前方向：信号状态 0 = 正向；信号状态 1 = 反向；

2. AXISx_DIS

功能：运动轴的 DIS 输出打开或关闭。这允许您将 DIS 输出用于禁用或启用电机控制器。

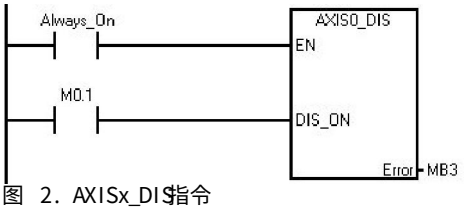


图 2. AXISx_DIS 指令

EN 位打开以启用子例程时，DIS_ON 参数控制运动轴的 DIS 输出。

⚠ **注意：**
如果您在运动轴中使用 DIS 输出，可以在每次扫描时调用该子例程，或者仅在您需要更改 DIS 输出值时进行调用。若实际 DIS 连接了电机驱动器的 DIS 输入，如果不使能则可能导致电机不运转。

3. AXISx_MAN
功能：将运动轴置为手动模式。 这允许电机按不同的速度运行，或沿正向或负向慢进。

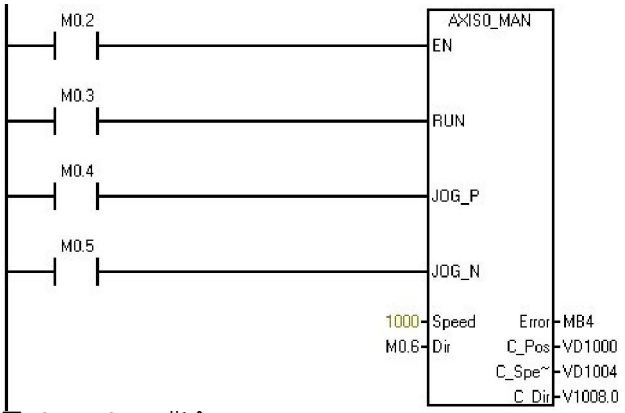


图 3. AXISx_MAN 指令

RUN 栈参数会命令运动轴加速至指定的速度（Speed 参数）和方向（Dir 参数）。 **您可以在电机运行时更改 Speed 参数，但 Dir 参数必须保持为常数。** 禁用 RUN 参数会命令运动轴减速，直至电机停止；
JOG_P（点动正向旋转）或 JOG_N（点动反向旋转）参数会命令运动轴正向或反向点动。 如果 JOG_P 或 JOG_N 参数保持启用的时间短于 0.5 秒，则运动轴将通过脉冲指示移动 JOG_INCREMENT 中指定的距离。 如果 JOG_P 或 JOG_N 参数保持启用的时间为 0.5 秒或更长，则运动轴将开始加速至指定的 JOG_SPEED；
Speed 参数决定启用 RUN 时的速度。 如果您针对脉冲组态运动轴的测量系统，则速度为 DINT 值（脉冲数 /每秒）。 如果您针对工程单位组态运动轴的测量系统，则速度为 REAL 值（单位数 /每秒）。

⚠**注意：**同一时间仅能启用 RUN、JOG_P 或 JOG_N 输入之一。

4. AXISx_RSEEK
功能：使用组态 /包络表中的搜索方法启动参考点搜索操作。当运动轴找到参考点且移动停止时，运动轴将 RP_OFFSET 参数值载入当前位置。

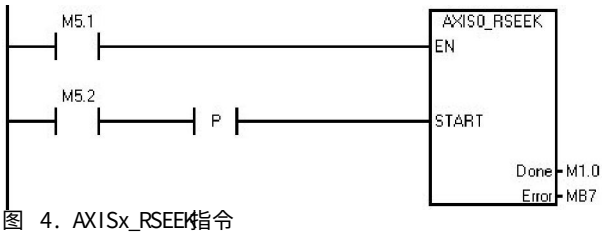


图 4. AXISx_RSEEK 指令

RP_OFFSET 的默认值为 0。 可使用运动控制向导、运动控制面板或 AXISx_LDOFF（加载偏移量）子例程来更改 RP_OFFSET 值；
EN 位开启会启用此子例程。确保 EN 位保持开启，直至 Done 位指示子例程执行已经完成；
START 参数开启将向运动轴发出 RSEEK 命令。 对于在 START 参数开启且运动轴当前不繁忙时执行的每次扫描，该子例程向运动轴发送一个 RSEEK 命令。为了确保仅发送了一个命令，**请使用边沿检测元素用脉冲方式开启 START 参数。**

5. AXISx_GOTO
功能：命令运动轴转到所需位置。

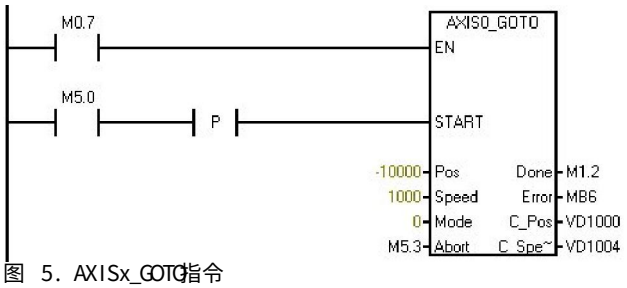


图 5. AXISx_GOTO指令

START 参数开启会向运动轴发出 GOTO 命令。 对于在 START 参数开启且运动轴当前不繁忙时执行的每次扫描，该子例程向运动轴发送一个 GOTO 命令。为了确保仅发送了一个 GOTO 命令，请使用边沿检测元素用脉冲方式开启 START 参数；
Pos 参数包含一个数值，指示要移动的位置（绝对移动）或要移动的距离（相对移动）。 根据所选的测量单位，该值是脉冲数 (DINT) 或工程单位数 (REAL)；
Speed 参数确定该移动的最高速度。 根据所选的测量单位，该值是脉冲数 /每秒 (DINT) 或工程单位数 /每秒 (REAL)；
Mode 参数选择移动的类型：
0：绝对位置
1：相对位置
2：单速连续正向旋转
3：单速连续反向旋转
Abort 参数启动会命令运动轴停止当前包络并减速，直至电机停止。

⚠**注意：**若 Mode 参数设置为 0，则必须首先使用 AXISx_RSEEK 或 AXISx_LDPOS 指令建立零位置。

6. AXISx_RUN
功能：命令运动轴按照存储在组态 /包络表的特定包络执行运动操作。

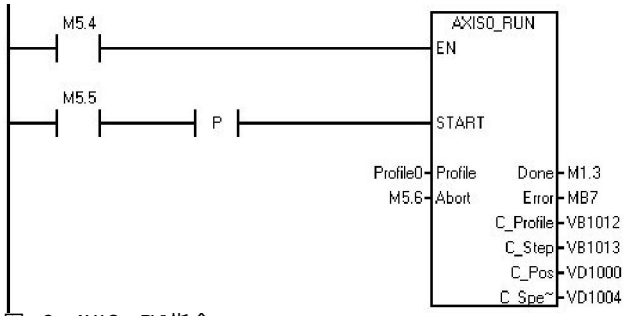


图 6. AXISx_RUN指令

START 参数开启将向运动轴发出 RUN 命令。 对于在 START 参数开启且运动轴当前不繁忙时执行的每次扫描，该子例程向运动轴发送一个 RUN 命令。为了确保仅发送了一个命令，请使用边沿检测元素用脉冲方式开启 START 参数；
Profile 参数包含运动包络的编号或符号名称。 “Profile”输入必须介于 0 - 31。否则子例程将返回错误；
Abort 参数会命令运动轴停止当前包络并减速，直至电机停止；
C_Profile 参数包含运动轴当前执行的包络；
C_Step 参数包含目前正在执行的包络步。

7. AXISx_LDOFF
功能：建立一个与参考点处于不同位置的新的零位置。

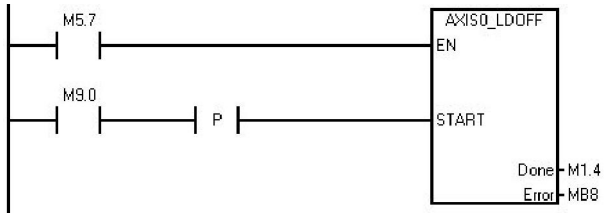


图 7. AXISx_LDOFF指令

开启 START 参数将向运动轴发出 LDOFF 命令。 对于在 START 参数开启且运动轴当前不繁忙时执行的每次扫描，该子例程向运动轴发送一个 LDOFF 命令。为了确保仅发送了一个命令，请使用边沿检测元素用脉冲方式开启 START 参数。

⚠**注意：**
在执行该子例程之前，您必须首先确定参考点的位置。您还必须将机器移至起始位置。 当子例程发送 LDOFF 命令时，运动轴计算起始位置（当前位置）与参考点位置之间的偏移量。 运动轴然后将算出的偏移量存储到 RP_OFFSET 参数并将当前位置设为 0。这将起始位置建立为零位置。
如果电机失去对位置的追踪（例如断电或手动更换电机的位置），您可以使用 AXISx_RSEEK 子例程自动重新建立零位置。

8. AXISx_LDPOS

功能：将运动轴中的当前位置值更改为新值。您还可以使用本子例程为任何绝对移动命令建立一个新的零位置。

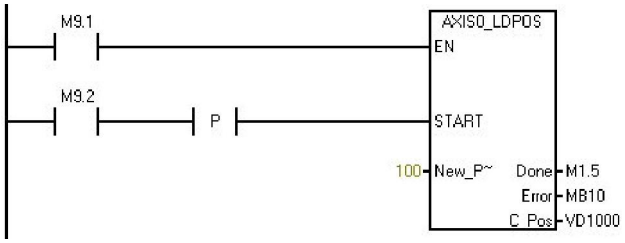


图 8. AXISx_LDPOS指令

START 参数开启将向运动轴发出 LDPOS 命令。 对于在 START 参数开启且运动轴当前不繁忙时执行的每次扫描，该子例程向运动轴发送一个 LDPOS 命令。为了确保仅发送了一个命令，请使用边沿检测元素用脉冲方式开启 START 参数；
New_Pos 参数提供新值，用于取代运动轴报告和用于绝对移动的当前位置值。 根据测量单位，该值是脉冲数 (DINT) 或工程单位数 (REAL)