Twido 可编程控制器

软件参考手册

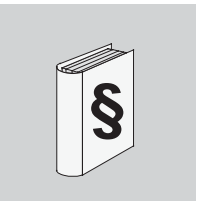
中文 V3.2

施耐德电气
旗下品牌

Telemecanique

目录

安全信息



重要信息

注意

请参照设备仔细阅读说明书，以便在安装，操作和维护之前熟悉设备。
以下特殊信息会在本手册和设备中出现，用以警告潜在的危险。

这个危险或警告符号警告用户有触电危险存在，如不遵循说明，将造成人员伤亡。

这是一个安全警告符号，警告用户存在潜在的人身伤害危险。
请遵循这个符号所有的安全信息，以免可能的伤害或死亡。

DANGER

DANGER 意味着即将发生危险，如果不避免这种情况，**必然会**引起死亡，严重伤害或设备损坏。

WARNING

WARNING 意味着存在潜在的危险，如果不避免这种情况，**将会**导致死亡，严重伤害或设备损坏。

CAUTION

CAUTION 意味着存在潜在的危险，如果不避免这种情况，**将会**导致死亡，严重伤害或设备损坏。

注意

只有合格用户才能使用电气设备。 Schneider Electric 不负担任何违反本手册操作而引起的责任。本手册仅限于受过培训的用户使用。在 Twido 硬件手册中提供了装配和安装的指导。 TWD USE 10AE。

附加安全信息

每一个对使用和安装本产品负责的用户必须确定每一个应用程序已经包含所有必要的设计考虑，并且符合所有的应用规则，性能与安全要求，规章，编码和标准。

一般性警告和注意

	**危险** **电击、燃烧或爆炸危险** 在安装，搬动，接线和维护之前请关闭电源。 **如果不遵守这个警告将导致 严重伤害或设备损坏。**

	**警告** **爆炸危险** ● 替代元件可能会损坏 Class Ⅰ， Div 2 适应性。 ● 在带电或危险区域中，请不要断开设备的连接。 **如果不注意这个警告将会导致严重伤害或设备损坏。**

警告

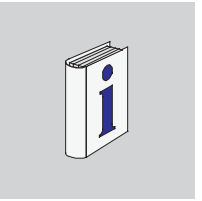
意外的设备操作

- 在安装，搬动，接线和维护之前请关闭电源。
- 本产品不适用于危急安全环境。在该环境下，人员或设备的危险依然存在，需要使用适当的硬连线安全锁紧装置。
- 请不要拆开，修理或更改模块。
- 请在机柜中使用控制器。
- 请在规定的操作环境下使用模块。
- 请在传感器与模块相连并需要供电的情况下使用传感器电源供应。
- 请在电源线和输出电路上使用 IEC60127 认证的熔断器，以满足电压和电流的要求。建议使用：5 × 20 mm 型 218 系列 / 延时 （慢熔型）熔断器 Littelfuse® 。

如果不注意这个警告将会导致严重伤害或设备损坏。

电池的安全处理

TWDLCA • 40DRF 一体化 CPU 使用一个外部的锂电池来长时间保持数据。
（注：锂电池需另外购买。）



	**警告** **爆炸和中毒危险** ● 不要燃烧锂电池，它可能爆炸和释放有毒物质。 ● 不要损坏或泄漏锂电池。 ● 废电池应正确处理，不正确的处理将造成环境污染。 ● 遵守当地的法律正确处理锂电池。如果不注意这个警告将会导致严重伤害或设备损坏。 **不按警告处理会导致死亡，严重伤害或设备损坏。**

注意

反极性晶体管输出损坏晶体管

- 确信晶体管输出接线正确。
- 反极性输出永久损坏输出电路。

如果不遵守这个警告将会导致人身伤害或设备损害。

有关本书

概览

文件范围

本书为 Twido 可编程控制器的软件参考手册，并由以下主要部分组成：

- 对 Twido 编程软件的描述及对 Twido 控制器进行编程所需基础的介绍。
- 对通信，管理模拟输入 / 输出，安装 AS-I 总线接口模块，CANopen 现场总线主模块及其他特殊功能的描述。
- 对用来编写 Twido 程序的软件语言的描述。
- 对 Twido 控制器说明及功能的介绍。

有效性注意

此手册中的信息仅适用于 Twido 可编程控制器。

修订史

版次	变更
1	Kampai 参考手册的改版
2	Kampai 1.1 版本第 1 步
3	将源语言改为法语并更新
4	对于 Twido 2.0 此版本包括对 TwidoSoft 的最新修改（2003 年 6 月）
5	对于 Twido 2.1
6	将源语言改为英语
7	Twido 3.0 版本模拟模块 + 在线编程
8	Twido 3.0 版本包括 CANopen 模块 +Twidoport+ 宏语言
9	对于 Twido v3.2 版本（BUD3）

相关文档

产品的相关警告

施耐德不对此文档中的任何错误承担责任。未经施耐德书面允许，不得已任何形式和手段复制该文档，包括使用电子版本。

用户评论

我们欢迎您对此文档进行评论。您可以通过 E-Mail 联系我们：techpub@schneider-electric.com

Twido 软件描述

I

概览

本部分的主题

本部分介绍了 Twido 可编程控制器的软件语言及创建控制程序所需要的基本信息。

本部分包含了哪些内容?

本部分包含了以下章节：

Twido 软件介绍

1

概览

本章的主题

本章简要介绍了 Twido 控制器的编程及配置语言：TwidoSoft，及 List， Ladder 和 Grafcet 编程语言。

本章包含了哪些内容?

本章包含了以下主题：

TwidoSoft 介绍

介绍

TwidoSoft 是一个图形开发环境，用于 Twido 可编程控制器应用程序的创建，配置和维护。 TwidoSoft 允许您用不同类型的语言创建程序 （见 *Twido 语言，p. 23*），然后传递应用程序到控制器运行。

TwidoSoft

TwidoSoft 是一个 32 位的基于 Windows 操作系统的软件，运行环境为个人计算机（PC）的 Microsoft Windows 98 第二版， Microsoft Windows 2000 专业版或 Microsoft Windows XP 操作系统。
TwidoSoft 软件的主要特点：

- 标准的 Windows 用户接口
- 用于 Twido 控制器的编程和配置
- 用于控制器的通信和控制

注意：控制器和 PC 接口可以采用 TCP/IP 协议。务必确保此协议在 PC 上已经安装。

最小配置

使用的最小配置是：

- Pentium 300MHz，
- 128 Mb of RAM，
- 40 Mb 硬盘可用空间。

Twido 语言介绍

介绍

一个可编程控制器基于控制程序来读输入，写输出，并且解决逻辑问题。创建一个 Twido 控制器的控制程序即在 Twido 的一种编程语言中写一系列的指令。

Twido 语言

下列语言可以用来创建 Twido 控制程序：

- 指令列表语言：
  一个指令列表程序是由布尔指令写成的一个逻辑表达序列。
- 梯形图：
  梯形图是图形方式的逻辑表达。
- Grafcet 语言：
  Grafcet 语言由一系列的步和转换组成。Twido 支持 Grafcet 指令系统，但不支持图形 Grafcet。

您能在个人计算机 （PC）上用上述编程语言创建和编辑 Twido 控制程序。
列表 / 梯形图的转换可逆特性使得您能方便地将一个程序从梯形图转换为列表或从列表转换为梯形图。

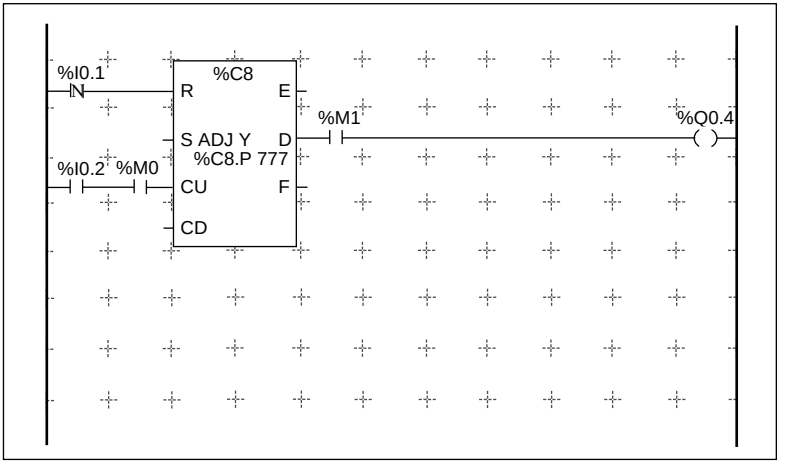
指令列表语言

指令列表语言写成的程序由一系列控制器顺序执行的指令组成。下面是一个列表程序示例：

```
0   BLK     %C8
1   LDF     %I0.1
2   R
3   LD      %I0.2
4   AND     %M0
5   CU
6   OUT_BLK
7   LD      D
8   AND     %M1
9   ST      %Q0.4
10  END_BLK
```

梯形图

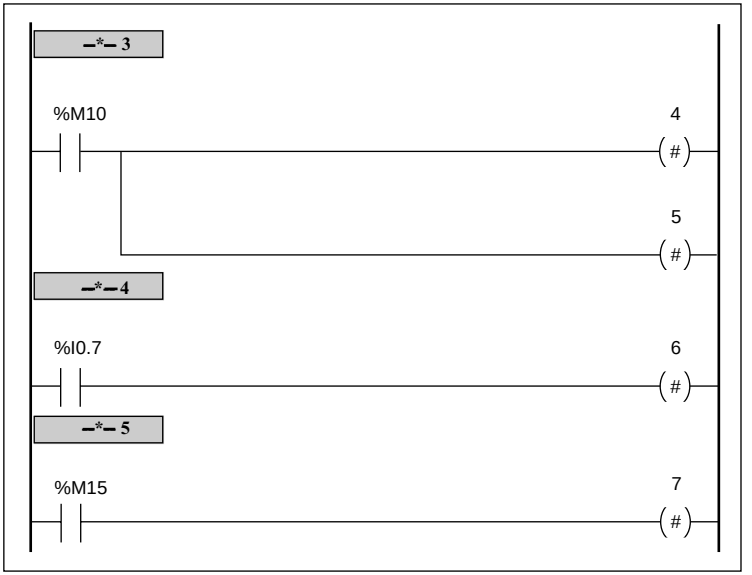
梯形图类似于表示继电器控制电路的继电器逻辑图。图形元素线圈，触点，和模块等表示指令。下面是一个梯形图示例。



Grafcet 语言

Grafcet 分析法将连续控制系统分为一系列的步，包括相关的动作，转换，和条件。下面分别是 Grafcet 指令的列表图示例和梯形图示例。

0	-*-	3
1	LD	%M10
2	#	4
3	#	5
4	-*-	4
5	LD	%I0.7
6	#	6
7	-*-	5
8	LD	%M15
9	#	7
10	...	



Twido 语言对象

2

概览

本章的主题

本章提供了用于 Twido 控制器编程的语言对象的详细资料。

本章包含了哪些内容?

本章包含了以下主题：

语言对象生效

介绍

字对象和位对象在控制器已分配它们存储空间后才有效。因此应用程序在下载到控制器之前必须用到它们。

举例

对象的有效范围是从零到此类对象的最大参数值。例如：如果存储字在您的应用程序中最大参数值是 %MW9，则 %MW0 到 %MW9 被分配空间。该例中 %MW10 无效且其内部访问和外部访问均不允许。

位对象

介绍

位对象是用作操作数且为布尔指令测试的位 - 类型软件变量。下面是位对象列表：

- I/O 位
- 内部位 （存储位）
- 系统位
- 步位
- 字的抽取位

位的操作数列表

下面表格列出并描述了所有在布尔指令中用作操作数的主要位对象。

类型	描述	地址或值	最大值	写访问 (1)
立即值	0 或 1 （假或真）	0 或 1	-	-
输入 输出	这些位是 I/O 电气状态的 “逻辑映像”。它们存储在数据存储器中且在每次程序逻辑扫描时得到更新。	%Ix.y.z (2) %Qx.y.z (2)	注解 （4）	不可以 可以
AS-Interface 输入 输出	这些位是 I/O 电气状态的 “逻辑映像”。它们存储在数据存储器中且在每次程序逻辑扫描时得到更新。	%IAx.y.z %QAx.y.z	注解 （5）	不可以 可以
内部（存储）	内部位是程序运行时存储立即数的内存区域。 **注意：**未用的 I/O 位不能用作内部位。	%Mi	128 TWDLC•A10 DRF， TWDLC•A16 DRF 256 其它所有 CPU 型号	可以
系统	系统位 %S0 到 %S127 监控控制器的正确操作及应用程序的正确运行。	%Si	128	取决于 i 的值
函数模块	函数模块位对应函数模块的输出。 这些输出或者直接被连接，或者用作一个对象。	%TMi.Q， %Ci.P 等等	注解 （4）	不可以 （3）
可逆函数模块	用可逆编程指 BLK，OUT_BLK，和 END_BLK 编程的函数模块。	E，D，F，Q， TH0，TH1	注解 （4）	不可以
字的抽取	一些 16 位的字中一位可被抽取为操作位。	不定	不定	不定

类型	描述	地址或值	最大值	写访问 (1)
Grafcet 步	位 %X1 到 %Xi 对应 Grafcet 步。步位 Xi 当对应步处于活动状态时置为 1，当对应步处于非活动状态时置为 0。	%X21	62 TWDLC•A10DRF， TWDLC•A16DRF 96 TWDLC•A24DRF， TWDLCA•40DRF 和模块型 CPU	可以

注解：

1. 被程序写或用活动表编辑器写。

2. 见 I/O 寻址。

3. 除了位 %SBRi.j 和 %SCi.j 能被读和写。

4. 数值由控制器型号决定。

5. x = 扩展模块地址 （0..7）； y = AS-Interface 地址 （0A..31B）； z = 通道数 （0..3）。（见*连接 AS-Interface V2 总线的从设备的相关 I/O 寻址，p.250*。）

字对象

介绍

字对象是存放在数据存储区中的 16 位字，它们可表示 -32768 到 32767 之间的任何整数 （除了高速计数器函数模块是 0 到 65535）。

字对象举例：

- 立即值
- 内部字 （%MWi）（存储字）
- 常量字 （%KWi）
- I/O 交换字 （%IWi， %QWi%）
- AS-Interface 模拟量 I/O 字 （%IWAi， %QWAi）
- 系统字 （%SWi）
- 功能模块 （配置数据和 / 或运行数据，具体的字对象值取决于不同的功能模块）

字的格式

字的内容或值根据下述约定以 16 位二进制码（或补码）的形式存放在用户内存中：

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	位的位置
0	1	0	1	0	0	1	0	0	1	0	0	1	1	0	1	位的状态
+	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	位的值

在带符号的二进制码中，第 15 位根据约定用于标示值的正负：

- 第 15 位为 0：字的值为正。
- 第 15 位为 1：字的值为负 （负值用二进制补码逻辑表示）。

字和立即值可以以以下形式存储和读取：

- 十进制
  最小值：-32768，最大值：32767 （例如， 1579）
- 十六进制
  最小值：16#0000，最大值：16#FFFF （例如， 16#A536）
  替换语法：#A536

字对象描述

下面表格是字对象的描述。

字	描述	地址或值	最大值	写访问 (1)
立即值	这些整数值的格式和 16 位字一样，允许这些值赋给这些字。		-	不可以
	10 进制	-32768 到 32767		
	16 进制	16#0000 到 16#FFFF		
内部（存储）	操作期间作为 “工作”字存值于数据存储区。字 %MW0 到 %MW2999 直接被程序读或写。	%MWi	3000	可以
常量	存储常量或文字数字信息。它们的内容只能通过 TwidoSoft 配置来写或修改。常量字 %KW0 到 %KW255 程序中为只读。	%KWi	256	可以，但是只能通过 TwidoSoft
系统	这些 16 位字具有功能： ● 通过读字 %Swi 为直接来自控制器的数据提供访问。 ● 完成应用程序中的操作 （例如，调节调度模块）。	%SWi	128	取决于 i 的值
功能模块	这些字对应功能模块的当前参数或值。	%TM2.P，%Ci.P，等等		可以
网络交换字	赋值给远程连接控制器。这些字用于控制器间通信：			
	网络输入	%INWi.j	4 点 / 远程连接	不可以
	网络输出	%QNWi.j	4 点 / 远程连接	可以

字	描述	地址或值	最大值	写访问 (1)
模拟量 I/O 字	用于 AS-Interface 子站模拟量的输入输出。			
	模拟量输入	%IWAx.y.z	注解 （3）	不可以
	模拟量输出	%QWAx.y.z	注解 （3）	可以
位抽取	可能从下列字中的 16 位中抽取一位：			
	内部	%MWi：Xk	2999	可以
	系统	%SWi：Xk	128	取决于 i 的值
	常量	%KWi：Xk	127	不可以
	输入	%IWi.j：Xk	注解 （2）	不可以
	输出	%QWi.j：Xk	注解 （2）	可以
	AS-Interface 子站输入	%IWAx.y.z：Xk	注解 （2）	不可以
	AS-Interface 子站输出	%QWAx.y.z：Xk	注解 （2）	可以
	网络输入	%INWi.j：Xk	注解 （2）	不可以
	网络输出	%QNWi.j：Xk	注解 （2）	可以

注意：
1. 被程序写或用活动表编辑器写。
2. 数值由配置决定。
3. x= 扩展模块地址 （0..7）； y=AS-Interface 地址 （0A-31B）； z= 通道数 （0..3）。（见*连接 AS-Interface V2 总线的从设备的相关 I/O 寻址， p.250*。）

浮点和双字对象

介绍

TwidoSoft 允许您进行浮点数和双整字对象操作。
浮点数是其表达式中含有小数的数学量。（例如：3.4E+38， 2.3 or 1.0）。
双整字是存放在数据存储区中的 4 字节字，包含介于 -2147483648 和 +2147483647 之间的一个值。

浮点数格式及值

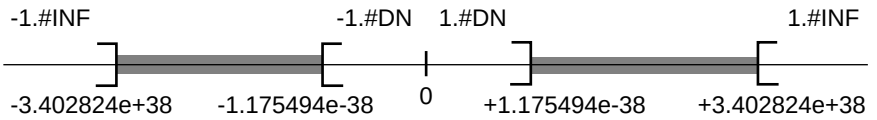
所用浮点格式是基于 IEEE STD 734-1985 标准 （等价于 IEC 559）。其字长 32 位，对应一个十进制浮点值。
浮点值格式见下表：

位 31	位 {30...23}	位 {22...0}
S	指数	小数部分

上面格式的表示值由下面方程决定：

32 位浮点值 = （-1）^{S}*2$^{（指数 -127）}$ *1. 小数部分

浮点值表达式中可有或没有指数，但它们一般必须有小数点 （浮点）。
浮点值范围从 -3.402824e+38 和 -1.175494e-38 到 1.175494e-38 和 3.402824e+38 （图中灰色值）。它们也包含值 0，记为 0.0。



当计算结果是：

- 小于 -3.402824e+38，显示符号 -1.#INF （表示负无穷），
- 大于 +3.402824e+38，显示符号 1.#INF （表示正无穷），
- 介于 -1.175494e-38 和 1.175494e-38 之间，近似为 0.0。这两个界限之间的值不是浮点值。
- 不确定 （例如负数的平方根），则显示符号 1.#NAN 或 -1.#NAN。

该表示法精度为 2-24。显示浮点数，小数点后 6 位阿拉伯数字即足够。

注意：

- 值 “1285”将作为一个整型值翻译；为了将其作为浮点值辨识，必须记做：“1285.0”

浮点数算术功能的限制

下表显示浮点数算术功能的限制

算术功能		限制和无效操作	
类型	语法	#QNAN （无效）	#INF （无限）
一个操作数的平方根	SQRT （x）	x < 0	x > 1.7E38
整数的实数指数 EXPT (%MF,%MW)	EXPT （y， x） (这里： x^y=%MW^%MF)	x < 0	y.ln （x） > 88
10 的对数	LOG （x）	x <= 0	x > 2.4E38
自然对数	LN （x）	x <= 0	x > 1.65E38
自然指数	EXP （x）	x < 0	x > 88.0

硬件兼容性

不是所有 Twido 控制器支持浮点和双字操作。

下表显示硬件兼容性：

Twido 控制器	双字支持	浮点支持
TWDLMDA40DUK	可以	可以
TWDLMDA40DTK	可以	可以
TWDLMDA20DUK	可以	不可以
TWDLMDA20DTK	可以	不可以
TWDLMDA20DRT	可以	可以
TWDLCA • 40DRF	可以	可以
TWDLC • A24DRF	可以	不可以
TWDLC • A16DRF	可以	不可以
TWDLC • A10DRF	不可以	不可以

有效性检查

当结果不在有效范围之内，系统位 %S18 将置为 1。
状态字 %SW17 的位显示浮点操作出错的原因：
字 %SW17 的不同位：

%SW17:X0	无效操作，结果不是一个数 （1.#NAN 或 -1.#NAN）
%SW17:X1	保留
%SW17:X2	被 0 除，结果为无穷 （-1.#INF 或 1.#INF）
%SW17:X3	结果绝对值大于 +3.402824e+38，视为无穷大 （-1.#INF 或 1.#INF）
%SW17:X4 到 X15	保留

该字在系统冷启动或程序重复使用时置为 0。

浮点和双字描述

下表是浮点和双字的描述：

对象类型	描述	地址	最大值	写访问	索引表
立即值	32 位同样格式的整数或小数对象。	-	[-]	不可以	-
内部浮点	对象用于操作过程中存储值于数据存储区中。	%MFi	2999	可以	%MFi[索引]
内部双字		%MDi	2999	可以	%MDi[索引]
浮点常量	用于存储常量。	%KFi	255	可以，但只能通过 TwidoSoft	%KFi[索引]
双字常量		%KDi	255	可以，但只能通过 TwidoSoft	%KDi[索引]

对象之间重迭的可能性

单字，双字和浮点字均存储于同一存储区域的数据空间。这样，浮点字 %MFi 和双字 %MDi 与单字 %MWi 和 %MWi+1 相对应 （字 %MWi 包含低位且字 %MWi+1 包含字 %MFi 的高位）。

下表显示了浮点和内部双字是怎样重迭的：

浮点双字	奇数地址	内部字
%MF0 / %MD0		%MW0
	%MF1 / %MD1	%MW1
%MF2 / %MD2		%MW2
	%MF3 / %MD3	%MW3
%MF4 / %MD4		%MW4
	...	%MW5
...		...
	%MFi / %MDi	%MWi
%MFi+1 / %MDi+1		%MWi+1

下表显示了浮点和双字常量是怎样重迭的：

浮点双字	奇数地址	内部字
%KF0 / %KD0		%KW0
	%KF1 / %KD1	%KW1
%KF2 / %KD2		%KW2
	%KF3 / %KD3	%KW3
%KF4 / %KD4		%KW4
	...	%KW5
...		...
	%kFi / %kDi	%KWi
%KFi+1 / %KDi+1		%KWi+1

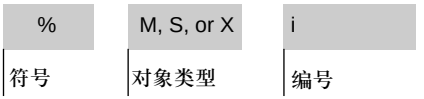
示例：

%MF0 对应 %MW0 和 %MW1。 %KF543 对应 %KW543 和 %KW544。

位对象寻址

句法

用下面格式寻址内部位，系统位，和步位对象：



描述

下面表格是对寻址格式中各元素的描述。

组	条目	描述
符号	%	百分比符号一般用于软件变量前。
对象类型	M	内部位在程序运行时存储中间值。
	S	系统位提供控制器状态和控制信息。
	X	步位提供步的活动状态。
编号	i	最大编号值取决于配置对象的编号。

位对象寻址示例：

- %M25 = 内部位编号 25
- %S20 = 系统位编号 20
- %X6 = 步位编号 6

来自字的抽取位

通过 TwidoSoft 可以从字的 16 位中抽取一位。根据下面语法字的地址加上了抽取位的排列号：

示例：

- %MW5：X6 = 内部字 %MW5 的第 6 位
- %QW5.1：X10 = 输出字 %QW5.1 的第 10 位

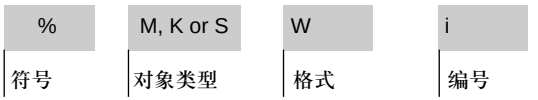
字对象寻址

介绍

字对象寻址，除了输入 / 输出寻址 （见*输入 / 输出寻址，p.447*）和功能块 （见*功能模块对象，p.47*），遵循以下规则。

句法

下面格式用于寻址内部字，常量字，和系统字：



描述

下面表格是对寻址格式中各元素的描述。

组	条目	描述
符号	%	百分比符号一般用于内部地址前。
对象类型	M	内部字在程序运行时存储中间值。
	K	常量字存储常量值或文字数字信息。它们的内容只能通过 Twidosoft 写或修改。
	S	系统字提供控制器状态和控制信息。
语法	W	16 位字。
编号	i	最大编号值取决于配置对象的编号。

字对象寻址示例：

- %MW15 = 内部字编号 15
- %KW26 = 常量字编号 26
- %SW30 = 系统字编号 30

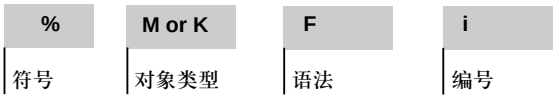
浮点对象寻址

介绍

浮点对象寻址，除了输入 / 输出寻址 （见*输入 / 输出寻址 p.44*）和功能块 （见功能模块对象， p.47），遵循以下规则。

句法

下面格式用于寻址内部和常量浮点对象：



描述

下面表格是对寻址格式中各元素的描述。

组	条目	描述
符号	%	百分比符号一般用于内部地址前。
对象类型	M	内部浮点对象在程序运行时存储中间值。
	K	浮点常量存储常量值。它们的内容只能通过 Twidosoft 写或修改。
语法	F	32 位对象。
编号	i	最大编号值取决于配置对象的编号。

浮点对象寻址示例：

- %MF15 = 内部浮点对象编号 15
- %KF26 = 常量浮点对象编号 26

双字对象寻址

介绍

双字对象寻址，除了输入 / 输出寻址 （见*输入 / 输出寻址，p.44*）和功能块 （见*功能模块对象，p.47*），遵循以下规则。

句法

下面格式用于寻址内部和常量双字：

描述

下面表格是对寻址格式中各元素的描述。

组	条目	描述
符号	%	百分比符号一般用于内部地址前。
对象类型	M	内部双字在程序运行时存储中间值。
	K	常量双字存储常量值或文字数字信息。它们的内容只能通过 Twidosoft 写或修改。
语法	D	32 位双字。
编号	i	最大编号值取决于配置对象的编号。

双字对象寻址示例：

- %MD15 = 内部双字编号 15
- %KD26 = 常量双字编号 26

输入 / 输出寻址

介绍

Twido 配置中每个输入 / 输出 （I/O）点有一个唯一地址：例如，地址 “%I0.0.4” 表示控制器本体的输入 4。

I/O 地址可赋值给下列硬件：

- 设定为远程连接主机的控制器
- 设定为远程 I/O 的控制器
- 扩展 I/O 模块

TWDNOI10M3 AS-Interface 总线接口模块和 TWDNCO1M CANopen 总线接口模块使用各自的子站 I/O 寻址系统：

- 关于 TWDNOI10M3，见 *连接 AS-Interface V2 总线的从设备的相关 I/O 寻址，p.252*。
- 关于 TWDNCO1M，见 *CANopen 主模块 PDO 寻址， p.293*。

输出或线圈的多重涉及

一个程序中，您可以对一个输出或线圈有多重涉及。但只有最后的结果才能更新硬件的输出。例如， %Q0.0.0 可在一个程序中用到多次，并且这种多次出现不会得到任何警告。因此有必要确定只有一个因素给定输出的所需状态。

	**注意** **意外操作** 不提供任何重复输出检查或警告。您在应用程序中改变输出或线圈之前，请仔细阅读它们的使用。 **如果不遵守这个警告将会导致人身伤害或设备损害。**

格式

用下面格式寻址输入 / 输出

%	I, Q	x	.	y	.	z
符号	对象类型	控制器位置	点	I/O 类型	点	通道编号

用下面格式寻址输入 / 输出交换字

%	I, Q	W	x	.	y
编号	对象类型	格式	控制器位置	点	I/O 类型

描述

下面表格描述了 I/O 寻址格式。

组	条目	值	描述
符号	%	-	百分比符号一般用于内部地址前。
对象类型	I	-	输入。控制器或扩展 I/O 模块输入电气状态的“逻辑映像”。
	Q	-	输出。控制器或扩展 I/O 模块输出电气状态的“逻辑映像”。
控制器位置	x	0 1 - 7	主控制器 （远程连接主机）。 远程控制器 （远程连接从机）。
I/O 类型	y	0 1 - 7	基本 I/O （控制器本地 I/O）。 扩展 I/O 模块。
通道编号	z	0 - 31	控制器或扩展 I/O 模块的 I/O 通道号。可用 I/O 点的编号取决于控制器类型和扩展 I/O 模块类型。

举例

下表是 I/O 寻址的一些示例。

I/O 对象	描述
%I0.0.5	控制器本体 5 号输入点 （本地 I/O）。
%Q0.3.4	本地控制器 3 号扩展 I/O 模块 4 号输出点 （扩展 I/O）。
%I0.0.3	控制器本体 3 号输入点。
%I3.0.1	3 号远程连接 I/O 控制器 1 号输入点。
%I0.3.2	本地控制器 3 号扩展 I/O 模块 2 号输入点。

网络寻址

介绍

Twido 远程连接网络中的应用程序数据通过网络字 %INW 和 %QNW 在对等控制器和主控制器间得到交换。见*通信，p.91* 以获得详细资料。

格式

下面格式用于网络寻址。

%	IN,QN	W	x	.	j
符号	对象类型	格式	控制器位置	点	字

格式描述

下面表格是网络寻址格式描述。

组	项	值	描述
符号	%	-	百分比符号一般用于内部地址前。
对象类型	IN	-	网络输入字。从从机到主机传输数据。
	QN	-	网络输出字。从主机到从机传输数据。
格式	W	-	一个 16 位字。
控制器位置	x	0 1 - 7	主控制器 （远程连接主机）。 远程控制器 （远程连接从机）。
字	j	0 - 3	每个对等控制器用一到四个字与主控制器进行数据交换。

举例

下表是网络寻址的一些示例。

网络对象	描述
%INW3.1	3 号远程控制器 1 号网络字。
%QNW0.3	基本控制器 3 号网络字。

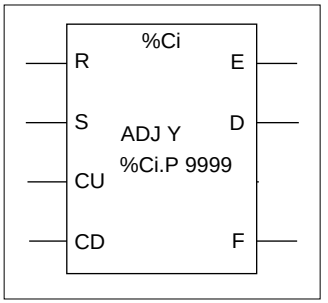
功能模块对象

介绍

功能模块提供了可供程序访问的位对象和特殊字。

功能模块示例

下面是一个计数器功能模块。



加 / 减计数功能块

位对象

位对象对应模块输出。布尔测试指令能用下面任一方法访问这些位：

- 直接方式 （例如， LD E），如果它们在可逆编程中与模块有线连接 （见*标准功能模块编程原则 p.423*）。
- 通过指定模块类型 （例如， LD %Ci.E）。

输入由指令表访问。

字对象

字对象对应指定的参数和值如下：

- 模块配置参数：一些程序可访问参数 （例如，预选参数），和一些程序非访问参数 （例如，时基）。
- 当前值：例如， %Ci.V，当前计数值。

字对象

执行系统功能时，双字增强了 Twido 的计算能力，如高速计数 （%FC），超高速计数 （%VFC）和脉冲输出 （%PLS）。
32-bit 双字的寻址只需在原有的标准字对象上加 “D”。下例显示以单字和双字方式寻址高速计数当前值：

- %FCi.V 高速计数当前值的单字形式。
- %FCi.VD 高速计数当前值的双字形式。

注意：双字不是所有的 Twido CPU 都支持 . 参见 *硬件兼容性，p.37*。 Twido controller 是否支持双字。

程序可访问对象

请参见下面相应部分列出的程序可访问对象。

- 针对基本功能模块，见*基本功能模块，p.421*。
- 针对高级功能模块，见*与高级功能模块相关的位和字对象，p.476*。

结构化对象

介绍

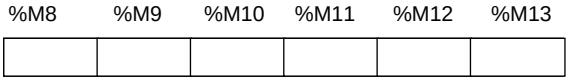
结构化对象是邻近对象的联合。 Twido 支持下列结构化对象：

- 位串
- 字表
- 双字表
- 浮点字表

位串

位串是指一系列类型相同的相邻对象位，并被定义为长度 （L）。

示例：位串 %M8:6



注意：%M8:6 是可接受的 （8 是 8 的倍数），但 %M10:16 是不可接受的 （10 不是 8 的倍数）。

位串可被用于赋值指令 （见*赋值指令，p.448*）。

位的可用类型

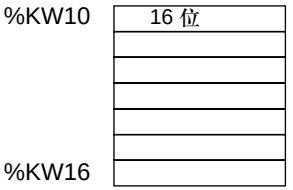
位串中位的可用类型：

类型	地址	最大值	写访问
离散输入位	%I0.0：L 或 %I1.0：L（1）	0<L<17	不可以
离散输出位	%Q0.0：L 或%Q1.0：L（1）	0<L<17	可以
系统位	%Si：L， i 为 8 的倍数	0<L<17 和 i+L ⩽ 128	取决于 i
Grafcet 步位	%Xi：L i 为 8 的倍数	0<L<17 和 i+L ⩽ 95（2）	可以 （通过程序）
内部位	%Mi：L i 为 8 的倍数	0<L<17 和 i+L ⩽ 256（3）	可以

关键：
1. 位串中 I/O 位只有 0 到 16 可读。对于 24 输入的控制器和 32 I/O 模块，位串中高于 16 的位将不可读 （如访问高于 16 的位，采用 %I0.16:8 等类似方式）。
2. TWDLCAA10DRF 和 TWDLCAA16DRF 的 i+L 最大值是 62
3. TWDLCAA10DRF 和 TWDLCAA16DRF 的 i+L 最大值是 128

字表

字表是指一系列类型相同且相邻的字，并被定义为长度 （L）。
示例：字表 %KW10：7



字表可用于赋值指令 （见*赋值指令，p.448*）。

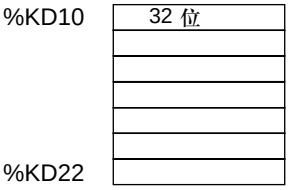
字的可用类型

字表中字的可用类型：

类型	地址	最大值	写访问
内部字	%MWi：L	0<L<256 和 i+L< 3000	可以
常量字	%KWi：L	0<L <256 和 i+L<256	不可以
系统字	%SWi：L	0<L 和 i+L<128	取决于 i

双字表

双字表是指一系列类型相同且相邻的字，并有规定的长度 （L）。
示例：双字表 %KD10：7



双字表可用于赋值指令 （见*赋值指令 p.448*）。

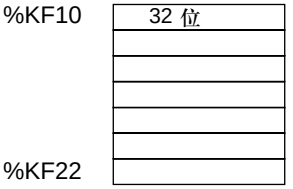
双字的可用类型

双字表中字的可用类型：

类型	地址	最大值	写访问
内部字	%MDi：L	0<L<256 和 i+L< 3000	可以
常量字	%KDi：L	0<L 和 i+L<256	不可以

浮点字表

浮点字表是指一系列类型相同且相邻的浮点字，并被定义为长度 （L）。
示例：浮点字表 %KF10：7



浮点字表可用于赋值指令 （见*先前指令*）。

浮点字的可用类型

浮点字表中浮点字的可用类型：

类型	地址	最大值	写访问
内部字	%MFi：L	0<L<256 和 i+L< 3000	可以
常量字	%KFi：L	0<L 和 i+L<256	不可以

索引对象

介绍

索引字指的是含有索引对象地址的单字，双字或浮点。对象寻址方式有两种：

- 直接寻址
- 索引寻址

直接寻址

当程序写完之后，对象的直接地址就被设定和定义。
示例：%M26 此内部位的直接地址是 26。

索引寻址

对象的索引地址通过给对象的直接地址添加一个索引，提供了一个修改对象地址的方法。索引内容被添加到对象直接地址中，索引由内部字 %MWi 定义。“索引字”的数量没有限制。
示例：%MW108[%MW2] 字的地址由直接地址 108 加上字 %MW2 的内容组成。如果字 %MW2 的值是 12，则写入 %MW108[%MW2] 等价于写入 %MW120 （108 加 12）。

可索引寻址的对象

下面是可以索引寻址的对象类型。

类型	地址	最大值	写访问
内部字	%MWi[MWj]	0 ⩽ i+%MWj<3000	可以
常量字	%KWi[%MWj]	0 ⩽ i+%MWj<256	不可以
内部双字	%MDi[MWj]	0 ⩽ i+%MWj<2999	可以
常量双字	%KDi[%MWj]	0 ⩽ i+%MWj<255	不可以
内部浮点	%MFi[MWj]	0 ⩽ i+%MWj<2999	可以
常量浮点	%KFi[%MWj]	0 ⩽ i+%MWj<255	不可以

索引对象可被用于赋值指令 （见*赋值指令，p448* 用于单 \ 双字）和比较指令 （见*比较指令 p.453* 用于单 \ 双字）。这种寻址使得通过修改程序中索引对象的内容，可以连续扫描一系列相同类型的对象 （如内部字和常量）。

索引溢出系统位 %S20

当索引对象的地址超出此类对象存储区域的限制，就会发生索引溢出。概括如下：

- 对象地址加索引内容小于 0。
- 对象地址加索引内容大于程序直接引用字的最大值。最大值是 2999 （对字 %MWi）或 255 （对字 %KWi）。

索引溢出事件发生后，系统将系统位 %S20 置为 1，且该对象索引值赋为 0。

注意：用户有责任对任何溢出进行监测。用户程序必须读位 %S20 并作可能的处理。用户必须确认将其复位到 0。

%S20 (初始状态 = 0):

- 索引溢出发生：系统将其置为 1。
- 溢出确认：用户在修改索引后，将其置为 0。

符号化对象

介绍

您能用名字或用户化的记忆符号来标注 Twido 软件语言对象。符号的使用使得程序逻辑可以快速检查和分析，并且极大地简化了应用程序的开发和测试。

举例

例如， WASH_END 这个符号可用来表示一个定时器功能模块中一个冲洗周期的结束。记忆这个名字的意义将比记忆一个程序地址如 %TM3 的意义容易得多。

符号定义原则

符号定义原则如下：

- 最多 32 个字符。
- 只能使用字母 （A-Z），数字 （0 -9），和下划线 （_）。
- 第一个字符必须是字母或重音字符。您不能使用百分号 （%）。
- 不能使用空格和特殊字符。
- 不区分大小写。例如，Pump1 和 PUMP1 视为相同符号，在一个程序中只允许用一次。

符号编辑

符号在符号编辑器中被定义并与语言对象相关联。程序里这些符号和其注释存储在 PC 硬盘而不是控制器里。因此，它们不随应用程序传递给控制器。

用户存储器

3

概览

本章的主题

本章描述了 Twido 用户存储器的结构和用法。

本章包含了哪些内容?

本章包含了以下主题：

用户存储器结构

介绍

应用程序可以访问的控制器存储器分为两个不同的集：

- 位值
- 单字值 （16-bit 有符号）和双字值 （32-bit 有符号）

位存储器

位存储器位于处理器的内置 RAM。它含有 256 位对象图。

存储字

字存储器 （16 位）支持：

- **动态字**：运行存储 （仅存储于 RAM）。
- **内存字 （%MW）和双字 （%MD）**：动态系统数据和系统数据。
- **程序**：任务描述符和执行码。
- **配置数据**：常量字，初始值，和输入 / 输出配置。

存储器存储类型

下面是 Twido 控制器存储器存储的不同格式。

- 随机存取存储器。
  内部暂时存储器：包含动态字，存储字，程序和动态数据。
- EEPROM
  一个整合的 32KB EEPROM 提供内部程序和数据备份。保护程序不因电池失效或停止外部供电超过 30 天而导致崩溃。包含程序和配置数据。保存最多可达 512 个存储字。如果使用 64K 扩展存储卡程序将不再备份在这里，且 Twido 已配置成接受 64K 扩展存储卡。
- 32K 备份卡
  一个可选择的外部卡，用来保存程序及传递程序给其它 Twido 控制器。能用来更新控制器 RAM 中的程序。包含程序和常量，但不包含存储字。
- 64K 扩展存储卡
  一个可选择的外部卡，可存储多达 64K 的程序。卡必须插在控制器中卡中程序才可使用。

备份存储器

PLC 的程序和内存字能被保存在以下内存中：

- RAM （好的电池最多可达 30 天）
- EEPROM （最大容量 32 KB）

RAM 中程序丢失 （或没有电池）时，程序能自动从 EEPROM 存储器传递给 RAM 存储器。
通过 TwidoSoft 也可手动完成传递。

存储器配置

下表描述了 Twido 一体型和模块型 CPU 可配置的内存类型。

存储器类型	一体型控制器				
	10DRF	**16DRF**	**24DRF**	**40DRF (32k)**	**40DRF** (64k)**
内部 RAM Mem 1*	10KB	10KB	10KB	10KB	10KB
外部 RAM Mem 2*		16KB	32KB	32KB	64KB
内部 EEPROM	8KB	16KB	32KB	32KB	32KB***
外部 EEPROM	32KB	32KB	32KB	32KB	64KB
最大程序容量	8KB	16KB	32KB	32KB	64KB
最大外部备份	8KB	16KB	32KB	32KB	64KB

存储器类型	模块型控制器		
	20DUK 20DTK	**20DRT 40DUK 40DTK (32k)**	**20DRT 40DUK 40DTK** (64k)**
内部 RAM Mem 1*	10KB	10KB	10KB
外部 RAM Mem 2*	32KB	32KB	64KB
内部 EEPROM	32KB	32KB	32KB***
外部 EEPROM	32KB	32KB	64KB
最大程序容量	32KB	32KB	64KB
最大外部备份	32KB	32KB	64KB

（*） Mem 1 和 Mem 2 位于存储器使用。
（**）在本情况中， 64KB 备份卡必须安装在 Twido 上并且要在配置中申明，
（***）保留用于备份前 512 %MW 单字或前 256 %MD 双字。

没有备份卡和扩展存储器时的备份和恢复

介绍

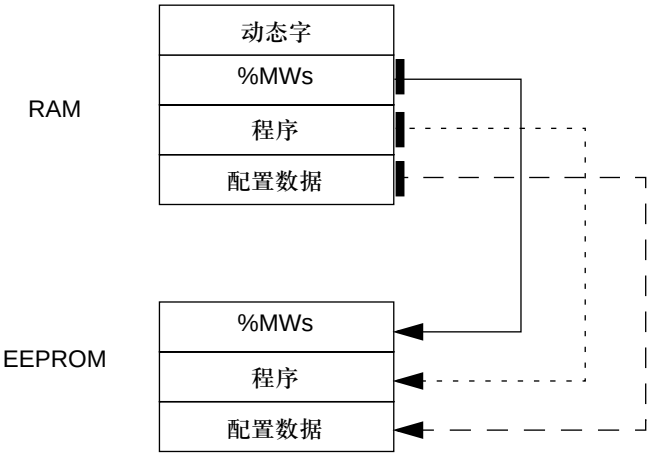
下面信息详细叙述了模块型和一体型控制器在没有备份卡和扩展存储器插入情况下的备份和恢复。

概览

Twido 程序，存储字和配置数据可用控制器内部 EEPROM 备份。因为保存程序到内部 EEPROM 将清除以前所有备份的存储字，因此程序和已配置的存储字必须备份。动态数据可存储在存储字里然后备份到 EEPROM。存储字只能在程序之后保存到内部 EEPROM。

存储器结构

下图是 PLC 的内存结构图 ,. 箭头显示了哪些内容可从 RAM 备份到 EEPROM：



程序备份

这里是程序备份到 EEPROM 的步骤：

步骤	动作
1	下面必须为真： RAM 中有一个有效程序。
2	在 Twido software window 的主菜单中选择 “控制器” 点击 “备份”，在 TwidoSoft 3.1 版本以上的软件中，备份到内部 EEPROM 的操作可以自动完成。

程序恢复

上电时有一种情况程序将从 EEPROM 恢复到 RAM （假设没有卡和扩展存储器存在）：

- RAM 中的程序无效。

为从 EEPROM 手动恢复程序按下面指示来做：

- 在 Twido software window 的主菜单中选择 ‘控制器’下拉到 ‘恢复’并点击。

数据 （%MWs）备份

这里是备份数据 （存储字）到 EEPROM 的步骤：

步骤	动作
1	但必须确保下面为真： RAM 中有一个有效程序 （%SW96：X6=1）。 相同的有效程序已备份到 EEPROM。 程序已配置存储字。
2	将 %SW97 置为将要保存的存储字的长度。 注意：长度不能超过存储字的配置长度，且必须大于 0，不超过 512。
3	将 %SW96：X0 置为 1。

数据 （%MWs）恢复

手动置系统位 %S95 为 1 即恢复 %MWs。

但必须确保下面为真：

- EEPROM 存在有效备份程序
- RAM 中程序与 EEPROM 备份程序匹配
- 备份的存储字有效

使用 32K 备份卡备份和恢复

介绍

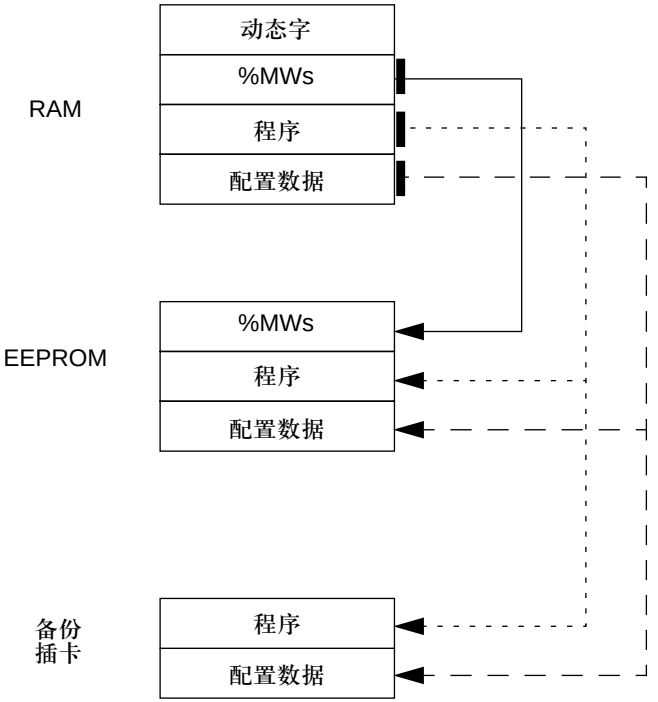
下面信息详细叙述了模块型和一体型控制器怎样使用 32K 备份卡进行备份和恢复。

概览

备份卡用来保存程序及传递程序到其它 Twido 控制器。一旦程序安装或保存完毕，卡应从控制器卸载并放到旁边。卡只能保存程序和配置字 （%MWs 不能保存在 32K 备份卡里）。动态数据可存储在存储字里然后备份到 EEPROM。当程序安装完成，任何在安装之前备份到内部 EEPROM 的 %MWs 都将丢失。

存储器结构

以下是装有备份卡的控制器内存结构图。箭头显示了哪些内容可从 RAM 备份到 EEPROM 和卡：



程序备份

这里是程序备份到备份卡的步骤：

步骤	动作
1	控制器断电。
2	插入备份卡。
3	控制器上电。
4	在 Twido software window 的主菜单中选择 “控制器” 点击 “备份”。
5	控制器断电。
6	从控制器卸载备份卡。

程序恢复

下面是加载备份卡保存的程序到控制器：

步骤	动作
1	控制器断电。
2	插入备份卡。
3	控制器上电。 （如果配置了自动运行，您必须重启进入运行模式）。
4	控制器断电。
5	从控制器卸载备份卡。

数据 （%MWs） 备份

这里是备份数据 （存储字）到 EEPROM 的步骤：

步骤	动作
1	但必须确保下面为真： RAM 中有一个有效程序。 相同的有效程序已备份到 EEPROM。 程序已配置存储字。
2	将 %SW97 置为将要保存的存储字的长度。 注意长度不能超过存储字的配置长度，且必须大于 0，不超过 512。
3	将 %SW96：X0 置为 1。

数据 （%MWs） 恢复

手动置系统位 %S95 为 1 即恢复 %MWs。
但必须确保下面为真：

- EEPROM 存在有效备份程序
- RAM 中程序与 EEPROM 备份程序匹配
- 备份的存储字有效

64K 扩展存储卡的使用

介绍

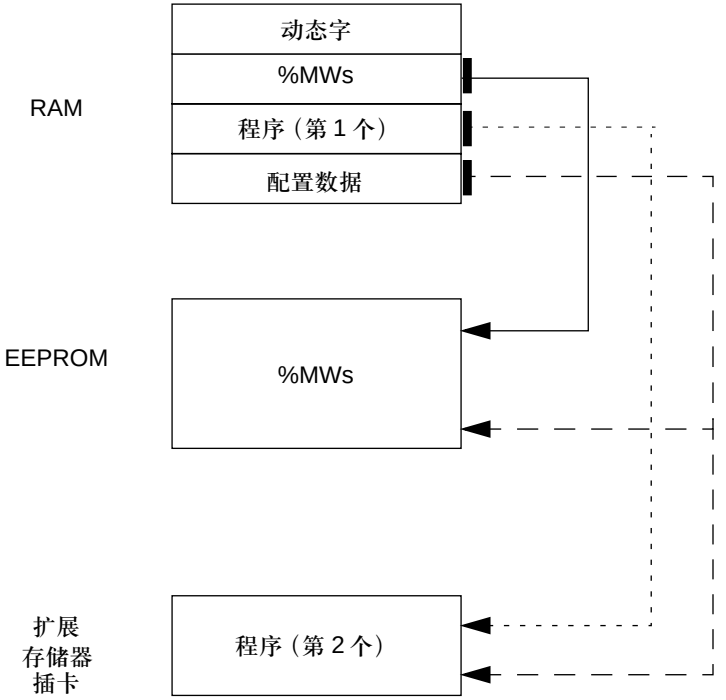
下面信息详细叙述了模块型控制器安装 64K 扩展存储卡后存储器的功能使用。

概览

64K 扩展存储卡将 Twido 控制器的程序存储容量从 32K 扩展到 64K。扩展程序使用时卡必须插在控制器里。如果卡被卸载，控制器将进入停止状态。存储字仍然备份到控制器的 EEPROM。动态数据可存储在存储字里然后备份到 EEPROM。 64K 扩展存储卡的上电动作和 32K 备份卡相同。

存储器结构

下图是使用了扩展内存卡的控制器内存结构图，箭头显示了哪些内容可从 RAM 备份到 EEPROM 和 64K 扩展内存卡：



软件配置和扩展存储器安装

在写入您的扩展程序之前，您必须安装到 64K 扩展内存卡到控制器上。下面是操作步骤：

步骤	动作
1	在 Twido software window 的硬件选件菜单中输入 ‘TWDXCPMFK64’。
2	控制器断电。
3	插入 64K 扩展内存卡。
4	控制器上电。

保存您的程序

一旦您的 64K 扩展内存卡安装完毕且您的程序已写完：

- 在 Twido software window 的主菜单中选择 “控制器”点击 “备份”。

数据 （%MWs） 备份

这里是备份数据 （存储字）到 EEPROM 的步骤：

步骤	动作
1	但必须确保下面为真： 一个有效程序存在 程序已配置存储字。
2	将 %SW97 置为将要保存的存储字的长度。 **注意**：长度不能超过存储字的配置长度，且必须大于 0，不超过 512。
3	将 %SW96：X0 置为 1。

数据 （%MWs） 恢复

手动置系统位 %S95 为 1 即恢复 %MWs。

但必须确保下面为真：

- 存在有效程序
- 备份的存储字有效

控制器操作模式

4

概览

本章的主题

本章描述了控制器工作模式和程序执行的循环和周期。包含了有关电源中断和恢复的详细信息。

本章包含了哪些内容?

本章包含了以下主题：

循环扫描

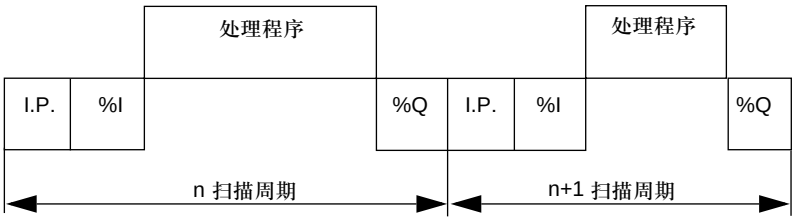
介绍

循环扫描是控制器工作于一个扫描循环接一个扫描循环的模式。在执行输出更新（任务循环的第三阶段）之后，系统执行自己任务的一个特定编号且立即触发另一个任务循环。

注意：用户程序的扫描时间由控制器的看门狗定时器监测且不能超过 500 ms。否则将出错并导致控制器立即停止转入到暂停模式。该模式将强制输出为它们默认的缺省安全状态。

操作

下图显示了循环扫描时间的运行状态。



循环状态描述

下表描述了循环状态。

地址	阶段	描述
I.P.	内部处理	系统后台监控控制器 （管理系统位和字，更新当前定时器值，更新状态指示灯，检测 RUN/STOP 开关，等等）及处理 TwidoSoft 请求 （修改和激活）。
%I, %IW	输入采集	将离散状态和程序特殊模块输入写入存储器。
-	程序处理	运行用户应用程序。
%Q，%QW	输出更新	写与离散和程序特殊模块关联的输出位或字。

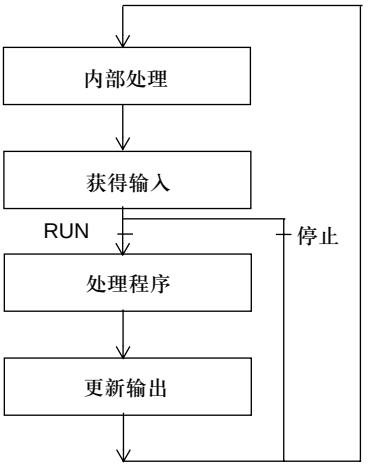
工作模式

控制器在运行状态，处理器执行：
- 内部处理
- 输入采集
- 应用程序运行
- 输出更新

控制器在停止状态，处理器执行：
- 内部处理
- 输入采集

图解

下面图例显示了操作循环。



循环的检查

看门狗检查循环。

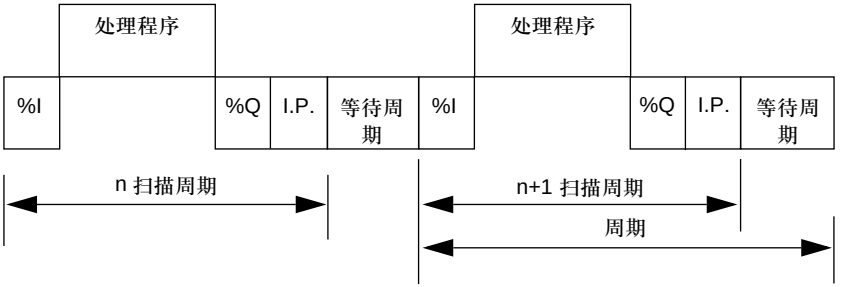
周期扫描

介绍

在此运行方式下，输入采集，应用程序运行，及输出更新根据配置定义时间 （2-150 ms）被周期性地执行。
在控制器扫描开始时，一个定时器开始倒计时，其值在配置定义周期里初始化。控制器扫描必须在定时结束之前完成扫描并开始一个新的扫描。

运行

下图显示了周期扫描时间的运行状态。



运行阶段描述

下表描述了运行状态。

地址	阶段	描述
I.P.	内部处理	系统后台监控控制器 （管理系统位和字，更新当前定时器值，更新状态指示灯，检测 RUN/STOP 开关，等等）及处理 TwidoSoft 请求 （修改和激活）。
%I，%IW	输入采集	将离散状态和程序特殊模块输入写入存储器。
-	程序处理	运行用户应用程序。
%Q，%QW	输出更新	写与离散和程序特殊模块关联的输出位或字。

工作模式

控制器在运行状态，处理器执行：

- 内部处理
- 输入采集
- 应用程序运行
- 输出更新

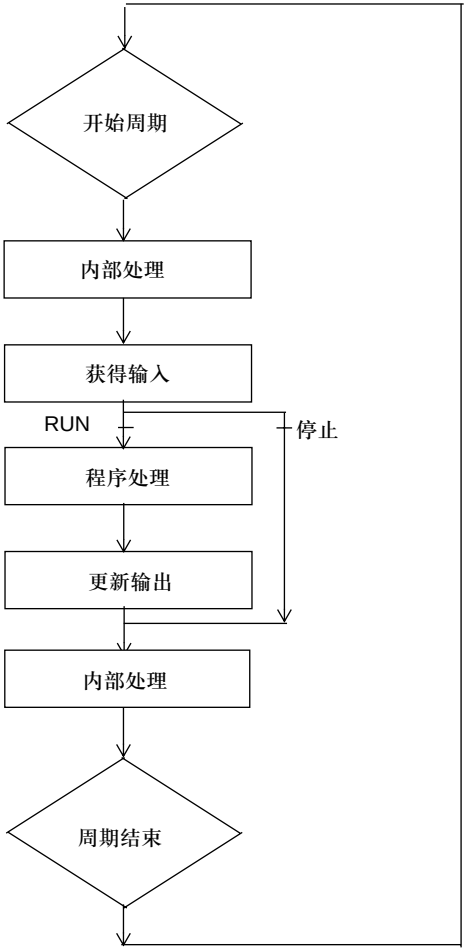
如果周期没有结束，处理器将完成操作循环直至内部处理周期结束。如果运行时间比分配的周期长，则控制器将系统位 %S19 置为 1 表示周期被超出。继续处理且运行完毕。但是不能超过看门狗的时间限制。后续扫描在写输出扫描后以后台运行的方式被连接进来。

控制器在停止状态，处理器执行：

- 内部处理
- 输入采集

图解

下面图例显示了操作循环。



循环的检查

可执行两种检查：

- 周期溢出
- 看门狗

扫描时间检查

概览

任务循环由看门狗定时器 Tmax （任务循环的最大持续时间）所监测。它允许显示程序错误 （无限循环，等等）且保证输出刷新的最大持续时间。

软件看门狗 （周期或循环运行）

周期或循环运行中，触发看门狗将导致一个软件错误。应用程序进入暂停状态且系统位 %S11 被置为 1。有必要重启任务与 Twido 软件连接以便分析错误原因，修改程序改正错误，然后重启程序运行。

注意：暂停状态是指由于程序软件错误如扫描溢出导致程序立即停止。数据保持当前值，以便错误原因分析。程序在处理指令处停止。与控制器的通信被打开。

周期运行检查

周期运行中附加检查用来检测周期被超出：

- **%S19** 指示周期被超出。设为：
  - 1 当扫描时间高于任务周期时由系统设定，
  - 0 由用户设定。
- **%SW0** 包含周期值 （0-150 ms）。它：
  - 冷启动时由配置选择值初始化，
  - 能被用户修改。

主任务运行时间使用

下面系统字用于控制器扫描循环时间信息：

- **%SW11** 最大看门狗时间初始化 （10 到 500 ms）。
- **%SW30** 包含最近一个控制器扫描循环的执行时间。
- **%SW31** 包含自最近一次冷启动以来最长的一个控制器扫描循环的执行时间。
- **%SW32** 包含自最近一次冷启动以来最短的一个控制器扫描循环的执行时间。

注意：这些不同信息也可从配置编辑器查到。

工作模式

介绍

Twido 软件用于考虑三类主要工作模式组：

- 检查
- 运行或生产
- 停止

从 Grafcet 开始

这些不同的工作模式可从下面 Grafcet 方法的开始或使用得到：

- Grafcet 初始化
- 步的预置
- 情形保持
- 图表固定

预处理和系统位的使用保证了有效的工作模式管理，而不使用户程序复杂和过载。

Grafcet 系统位

位 %S21，%S22 和 %S23 仅为预处理保留。这些位被系统自动复位。它们必须被置位指令 S 也只能被其写入。

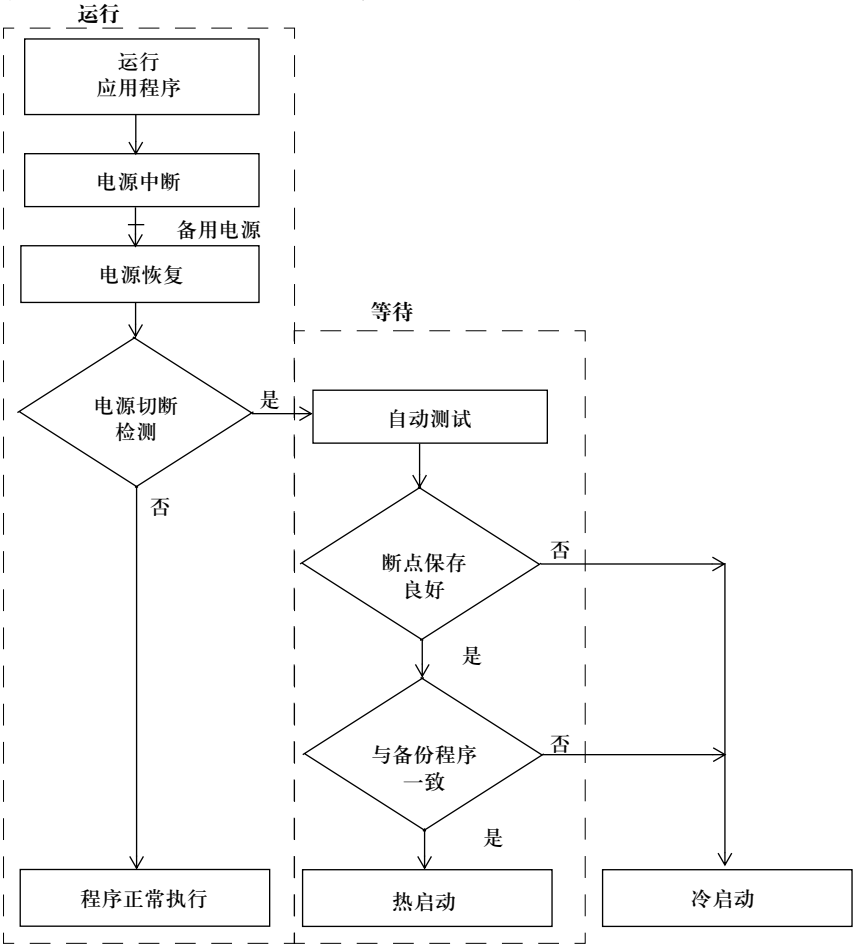
下表提供了与 Grafcet 相关的系统位：

位	功能	描述
%S21	GRAFCET 初始化	一般置为 0，下面将其置为 1： ● 冷启动，%S0=1 ； ● 用户，仅在部分预处理程序中，用置位指令 **S%S21** 或设置线圈 **- （S） - %S21**。 结果： ● 释放所有活动步。 ● 激活所有初始步。
%S22	GRAFCET 复位	一般置为 0，能且只能被程序预处理时置为 1。 结果： ● 释放所有活动步。 ● 扫描已停止的时序处理。
%S23	GRAFCET 预置和冻结	一般置为 0，能且只能被程序预处理时置为 1。 ● 通过设置 %S22 为 1 预置。 ● 通过一系列 S Xi 指令预置将要活动的步。 ● 通过设置 %S23 为 1 能够预置。 冻结情形： ● 在初始情形：通过程序将 %S21 维持为 1。 ● 在一个 “空”情形：通过程序将 %S22 维持为 1。 ● 在一个情形决定于将 %S23 维持为 1。

电源中断和电源恢复处理

图解

下面图例显示了系统检测到的各种电源重启。如果中断时间小于电源供应转换时间（交流电约为 10 ms，直流电 1 ms），程序将忽略并正常运行。



注意：断点将保存在电池备份 RAM 中。上电时，系统检查电池和存储的断点状态以决定是否可视为热启动。

运行 / 停止位对自动运行

运行 / 停止输入位的优先级比 “自动运行” 选项高，该选项可从扫描模式对话框得到。如果运行 / 停止位被设置，控制器在电源恢复时将在运行模式下重启。
控制器模式取决如下：

运行 / 停止输入位	自动运行	结果状态
0	0	停止
0	1	停止
上升沿	忽略	运行
1	忽略	运行
软件没有配置	0	停止
软件没有配置	1	运行

注意：对所有软件版本 V1.0 的一体型控制器，如果控制器在电源中断时处于运行模式，且扫描模式对话框中 “自动运行” 标志未设置，控制器在电源恢复时重新启动进入停止模式。否则将执行冷重启。

注意：对所有软件版本 V1.11 的模块型和一体型控制器，如果控制器的电池在电源中断时工作正常，控制器将以电源中断时的工作模式启动。电源恢复时，扫描模式对话框设置的 “自动运行” 标志对控制器模式不起影响。

操作

下表描述了电源中断的处理阶段。

阶段	描述
1	在电源中断事件中系统存储应用程序断点和中断时间。
2	所有输出被置为可靠状态 （0）。
3	电源恢复时，被保存的断点用来与一个进程比较，该进程定义了开始到运行的类型： ● 如果应用程序断点改变 （系统断点丢失或新的应用程序），控制器初始化应用程序：冷重启 （一体型体系）。 ● 如果应用程序断点相同，控制器重启且不初始化数据：热重启。

热重启处理

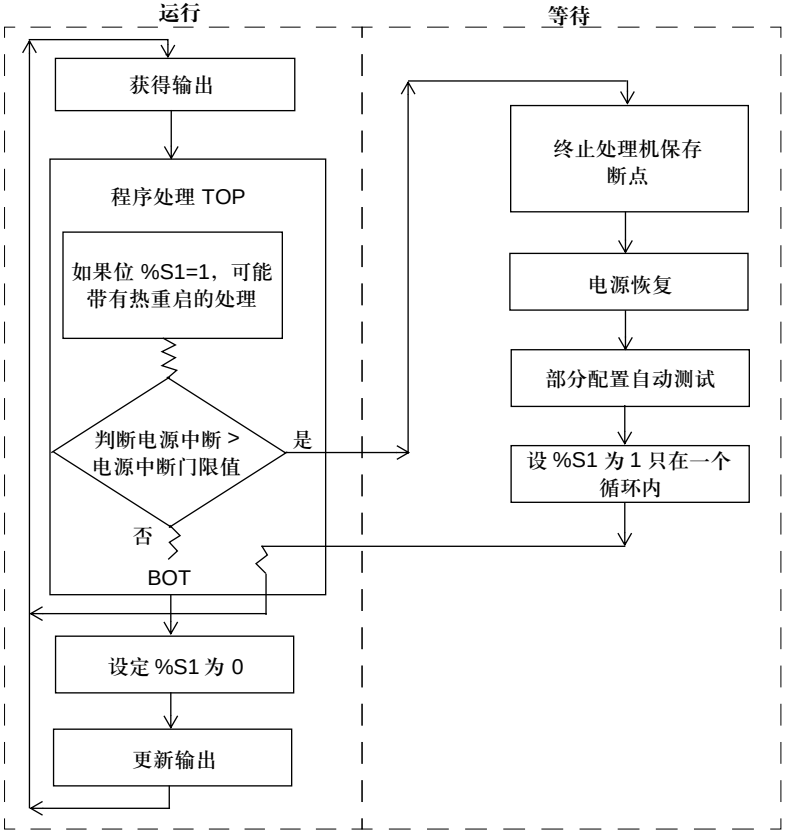
热重启原因

热重启将发生：

- 当电源恢复且应用程序断点没有丢失，
- 当位 %**S1** 被程序置为状态 1，
- 当控制器处于停止模式时操作显示

图解

下图描述了运行模式下的热重启操作。



程序执行的重启

下表描述了热重启后运行程序的重启阶段。

阶段	描述
1	程序执行从电源中断前的相同环境继续开始，但不更新输出。 **注意：**只有来自用户代码的相同环境可被重启。系统代码 （例如，输出更新）不能被重启。
2	重启循环结束时，系统： ● 释放被保留的应用程序 （并且在调试情况下引起应用程序停止） ● 讯息重新初始化
3	系统执行重启循环时： ● 重启任务置位 **%S1**（热启动指示位）和 **%S13**（运行中的第一次循环）为 1 ● 第一个任务循环结束时，重置位 **%S1** 和 **%S13** 为 0

热启动过程

热启动事件中，如果需要一个特殊的应用处理，位 **%S1** 在任务循环的开始必须被检测，并且调用相应的程序。

电源故障后的输出

一旦检测到电源中断，输出将被置为 （默认）可靠状态 （0）。
当电源恢复，输出将处于上次的状态直到它们被任务再次更新。

冷启动处理

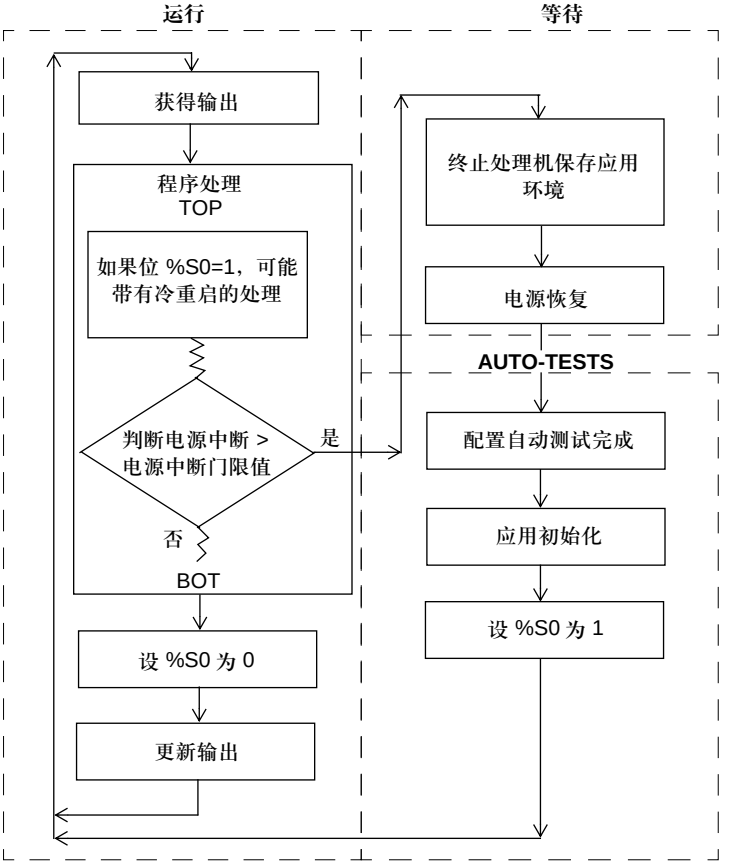
冷启动原因

冷启动将发生：

- 当装载新的应用程序到 RAM
- 当电源恢复且应用程序断点丢失
- 当系统位 **%S0** 被程序置为状态 1
- 当控制器处于停止模式时操作显示

图解

下图描述了运行模式下的冷启动操作。



运行

下表描述了冷启动后运行程序的重启阶段。

阶段	描述
1	启动时控制器处于运行模式。 若冷重启出错而停止，系统将强制冷重启。程序在任务循环开始时执行重启动。
2	系统： ● 复位内部位和字及 I/O 映像到 0 ● 初始化系统位和字 ● 从配置数据初始化功能模块
3	对第一个重启循环，系统： ● 重启任务置位 **%S0**（冷启动指示位）和 **%S13**（运行中的第一次循环）为 1 ● 重置位 **%S0** 和 **%S13** 在第一个任务循环结束时到 0 ● 设置位 **%S31** 和 **%S38**（事件控制指示）初始状态 1。 ● 重置位 **%S39**（事件控制指示）和字 **%SW48**（记录所有事件，除了周期事件）。

冷启动过程

冷启动事件中，如果需要一个特殊的应用处理，位 **%S0**（为 1）在第一次任务循环中必须检测。

电源故障后的输出

一旦检测到电源中断，输出将被置为（默认）可靠状态（0）。
当电源恢复，输出将处于零状态直到它们被任务再次更新。

控制器初始化

介绍

控制器初始化可由 Twido 软件通过设置系统位 **%S0** （冷启动）和 **%S1** （热启动）。

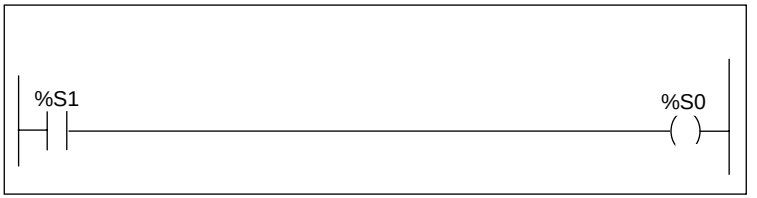
冷启动初始化

对于冷启动，系统位 **%S0** 必须置为 1。

用 %S0 和 %S1 进行热启动初始化（相对于冷启动）

在上电时初始化对象，系统位 **%S0** 和 **%S1** 必须置为 1。

下面示例显示了怎样用系统位对热重启初始化编程。




```
LD  %S1   如果 %S1 = 1 ( 热启动), 置 %S0 为 1 初始化控制器。
ST  %S0   在扫描周期结束时，系统把这两位置为 0。
```

注意：不要置 **%S0** 为 1 超过一个控制器扫描。

事件任务管理

5

摘要

概览

本章描述了事件任务以及它们在控制器中是怎样被执行的。

注意：Twido Brick 10 控制器 （TWDLCAA10DRF）不能管理事件任务。

本章包含了哪些内容?

本章包含了以下主题：

事件任务概述

介绍

前面的章节介绍了周期任务 （见*周期扫描，p.70*）和循环任务 （见*循环扫描，p.68*），对象在这些任务开始和结束时被更新。为使对象能够得到更快的更新，可以利用事件源中断一个确定任务的运行，以执行更高优先级的 （事件）任务。
所谓事件任务：

- 是在特定条件 （事件源）满足时，所执行的一部分程序，
- 它具有比主程序更高的优先级，
- 它具有更快的响应时间，以使得系统总的响应时间减少。

一个事件的描述

一个事件由以下组成：

- 一个事件源，定义为软件中断或硬件中断，用于中断主程序 （见*对不同事件源的描述，p.85*），
- 一个事件相关的，独立编程的实体，
- 一个事件队列，它用来存储事件列表直至事件被执行，
- 一个优先级，它指定了事件执行的顺序。

不同事件源的描述

不同事件源的概述

一个事件源由特定的软件管理，以保证主程序正确中断，并调用与事件关联的程序。应用程序的扫描时间对事件的执行没有影响。
下面 9 个事件源是被允许的：

- 4 个与 VFC 功能模块阈值相关联的条件 （每个 %VFC 有两个事件）
- 4 个与控制器本体物理输入相关联的条件，
- 1 个周期条件。

一个事件源只能对应一个事件，并且必须被 TwidoSoft 迅速地检测到。一旦被检测到，软件即转到执行与该事件对应的程序部分：每个事件对应一个子程序，标志为 **SRi**：由事件源配置决定。

本地控制器的物理输入事件

输入 %I0.2，%I0.3，%I0.4 和 %I0.5 可以用作事件源，如果它们没被锁定并且这些事件在配置中得到许可。
事件处理可以被控制器本体 （位置 0）的输入 2 到输入 5 在上升沿或下降沿时所激活。
欲知此事件配置更为详尽的信息，请参考 “硬件配置 -> 输入配置” 章节，此章节位于 “TwidoSoft 操作指导” 在线帮助中。

%VFC 函数模块的输出事件

%VFC 函数模块的输出 TH0 和 TH1 是事件源。输出 TH0 和 TH1 分别设置为：

- 1，当当前值大于阀值 S0 和 S1，
- 0，当当前值小于阀值 S0 和 S1。

这些输出的上升沿或下降沿能激活事件程序。
欲知此事件配置更为详尽的信息，请参考 “软件配置 -> 超高速计数器” 章节，此章节位于 “TwidoSoft 操作指导” 在线帮助中。

周期事件	此事件周期性地执行某一程序部分。该任务具有比主任务 （主程序）高的优先级。然而，该事件源的优先级比其它事件源的优先级要低。 该任务的周期在配置中设定，范围 5 到 255 ms。只能使用一个周期事件。 欲知此事件配置更为详尽的信息，请参考 “程序参数配置 -> 扫描模式”章节，此章节位于 “TwidoSoft 操作指导”在线帮助中。

事件管理

事件队列和优先级

事件有两种可能的优先级：高和低。但是只有一类事件 （即只有一类事件源）具有高优先级。其余事件都是低优先级，它们的执行顺序依赖于他们被检测到的顺序。为管理事件任务的执行顺序，存在两个事件队列：

- 一个队列里，最多可存储 16 个高优先级事件 （来自同一事件源），
- 另一个队列里，最多可存储 16 个低优先级事件 （来自其它事件源）。

这些队列的管理基于 FIFO 原理：存储在前的事件执行也在前。但是它们只能保存 16 个事件，多余的事件将会被丢失。

低优先级队列仅仅在高优先级队列为空时才得到执行。

事件队列管理

每当一个中断发生 （检测到事件源）时，下面步骤依次执行：

步骤	描述
1	中断管理： ● 物理中断识别， ● 事件被存储到对应的事件队列， ● 确认没有相同优先级的事件处于等待状态 （如果有，该事件将在队列中处于等待状态）。
2	断点保存。
3	执行连接到相应事件的程序部分 （标记 SRi：的子程序）。
4	输出更新
5	断点恢复

在断点恢复之前，队列中所有的事件必须执行完。

事件检查

系统位和字用来检查事件：（见 *系统位和系统字，p.657*）：

- %S31：用来执行或延迟一个事件，
- %S38：用来决定是否放置事件到事件队列中，
- %S39：用来查出事件是否丢失，
- %SW48：显示冷启动后执行的事件任务的总数 （周期中断出外）。

冷启动后 %S39 和 %SW48 复位到零， %S31 和 %S38 设置到初始状态 1，但热启动时保持不变。上述情况下，事件队列都将被复位。

特殊功能

II

概览

本部分的主题

本章介绍了通信，内置模拟量，模拟量 I/O 模块管理， AS-I V2 总线和 CANopen 现场总线功能。

本部分包含了哪些内容?

本部分包含了以下章节：

章节	章节名	页码
6	通信	91
7	内置式模拟功能	203
8	模拟模块管理	207
9	AS-I V2 总线安装	221
10	安装和配置 CANopen 现场总线	259
11	配置 TwidoPort 以太网网关	301
12	操作显示单元操作	331

通信

6

概览

本章的主题

本章提供了 Twido 控制器通信的配置，编程和管理概述。

本章包含了哪些内容?

本章包含了以下主题：

通信的各种类型介绍

概览

Twido 提供 1 或 2 个串行通信口，用于和远程 I/O，对等 PLC，或其它一般设备通信。但只有第一个端口能和 TwidoSoft 通信。每个 Twido 控制器支持三种不同的基本协议：远程连接， ASCII，或 Modbus （Modbus 主协议或 Modbus 从协议）。此外， TWDLCAE40DRF 一体型控制器提供 1 个 RJ-45 以太网通信口。 Modbus TCP/IP 客户端 / 服务器协议，用于在以太网中的控制器之间实现点对点的通信。

远程连接

远程连接协议是一种高速主 / 从总线，它支持一个主控制器和最多七个远程（从）控制器之间的少量数据通信。根据远程控制器的配置，传送相应的应用或 I/O 数据。远程控制器的类型可以是远程 I/O 或对等控制器。

ASCII

ASCII 协议是一个简单的半双工字符模式协议，用于传输和 / 或接收一个字符串到 / 自一个简单设备 （打印机或终端）。此协议只能通过 “EXCH” 指令得到支持。

Modbus

Modbus 协议是一个主 / 从协议，它允许一个并且只能一个主机发送命令，查询从机的响应。主机可单独对一个从机发送命令，也可以广播方式对所有从机发送命令。从机对每一个单独发送给它们的查询返回讯息 （响应）。但对广播方式的查询不做响应。

Modbus 主机模式 - Modbus 主机模式允许 Twido 控制器向从机发出 modbus 查询并等待响应。Modbus 主机模式只能通过 “EXCH” 令得到支持。 Modbus ASCII 和 RTU 均为 modbus 主机模式所支持。

Modbus 从机模式 - Modbus 从机模式允许 Twido 控制器响应主机的 modbus 查询。 Twido 控制器支持供对象访问的标准 modbus 数据，控制功能和服务扩展。 Modbus ASCII 和 RTU 均为 modbus 从机模式所支持。

注意：RS-485 网络 （没有中继器的情况下）可安装 32 个设备 （1 个主机和最多 31 个从机），它们的地址可在 1 和 247 之间选择。

Modbus TCP/IP

注意：只有 TWDLCAE40DRF 系列带有内置以太网接口的一体型控制器支持 Modbus TCP/IP。

以下信息描述了 Modbus 应用协议 （MBAP）。
Modbus 应用协议 （MBAP）是一个在 LAN 上支持可编程控制器和其他节点之间进行通信的 7 层协议。
当前的 Twido 控制器 TWDLCAE40DRF 实现以太网上的传输是通过基于 TCP/IP 上的 Modbus 应用协议。 Modbus 协议传输是典型的请求 – 响应信息对。 PLC 可用作客户端或服务器，取决于其发送或接收信息。

TwidoSoft 与控制器通信

概览

每个 Twido 控制器在它的端口 1 上有一个内置的 EIA RS-485 端口。它由内部电源供应。端口 1 必须用于和 TwidoSoft 编程软件通信。
选件卡或通信模块均不能用于这个连接。不过，调制解调器可以使用这个端口。
将 PC 连接到 Twido 控制器 RS-485 的端口 1 有几种方法：

- 通过 TSXPCX 电缆线，
- 通过电话线：调制解调器连接。

另外，TWDLCAE40DRF 一体型控制器有一个内置 RJ-45 以太网连接端口，可以与运行 TwidoSoft 软件的 PC 以太网进行通信。
可连接到以太网的 PC 与 TWDLCAE40DRF 控制器的 RJ-45 端口有两种方法：

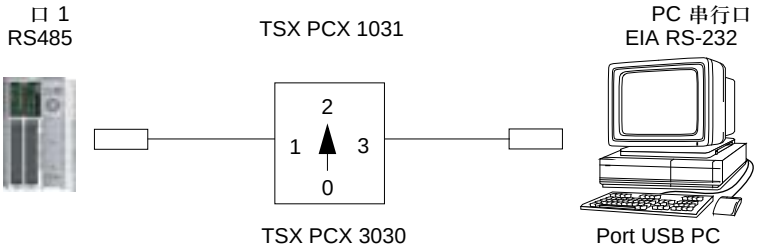
- 通过直接电缆连接：UTP Cat5 RJ45 以太网交叉电缆 （不推荐）
- 通过 Schneider Electric 电缆：SFTP Cat5 RJ45 以太网电缆 （电缆参数：490NTW000••）。

	**注意** **设备损坏** 当通信电缆 TSXPCX1031 或 TSX PCX 3030 从一个控制器物理移除并很快插入另一个控制器时， TwidoSoft 不能辨识这种断开连接。为了避免这种情况，请在移除电缆之前使用 TwidoSoft 断开连接。 **如果不遵守这个警告将会导致人身伤害或设备损害。**

TSXPCX 电缆连接

个人计算机的 EIA RS-232C 或 USB 端口通过 TSXPCX1031 或 TSX PCX 3030 多功能通信电缆与控制器的端口 1 相连接。电缆 TSX PCX 1031 转换 EIA RS-232 和 EIA RS-485 间的信号，电缆 TSX PCX 3030 转换 USB 和 EIA RS-485 间的信号。电缆上设有 4 – 位置的旋转开关可供选择不同模式的操作。开关对应的四个位置是“0-3”， TwidoSoft 与 Twido 控制器连接的正确设置是位置 2。

连接图如下所示。



注意：此电缆的 5 号引脚 DPT 信号不等于 0V。这表示控制器的当前连接是 TwidoSoft 连接。对执行固件来说，该内部上拉信号表示这是个 TwidoSoft 连接。

公和母连接器引脚　下面是 8- 引脚 miniDIN 公连接器的引脚图：

Mini DIN

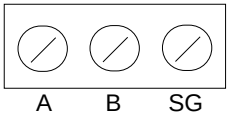
TWD NAC232D, TWD NAC485D

引脚输出	本体 RS485	RS485 可选	RS232-C
1	D1 (A+)	D1 (A+)	RTS
2	D0 (B-)	D0 (B-)	DTR
3	NC	NC	TXD
4	/DE	NC	RXD
5	/DPT	NC	DSR
6	NC	NC	GND
7	0 V	0 V	GND
8	5 V	5 V	5 V

注意：5V 最大（8 引脚）：180mA

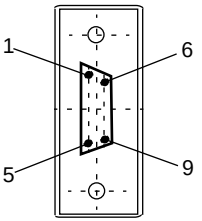
端子排

TWD NAC485T
TWD NOZ485T



引脚输出	RS485
A	D1 (A+)
B	D0 (B-)
SG	0V

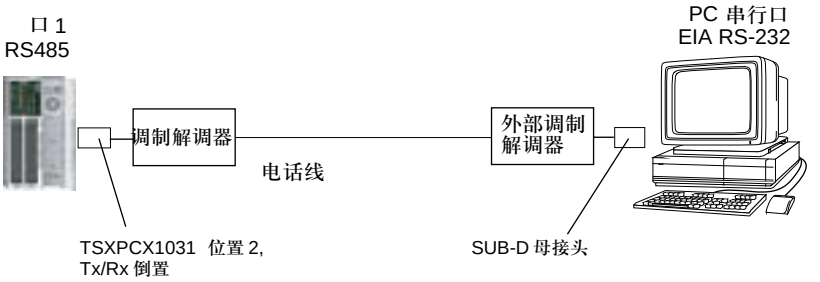
下面是 TSX PCX 1031 的 9- 引脚 SubD 母连接器的引脚图。



引脚输出	RS232
1	DCD
2	RX
3	TX
4	DTR
5	SG
6	NC
7	RTS
8	CTS
9	NC

电话线连接

调制解调器 （见 *TwidoSoft 和 Modem 间的通信，p.101*）通过电话线的连接可以对控制器编程，和控制器通信。
与控制器相连的调制解调器是外置的调制解调器与控制器端口 1 相连。
与 PC 相连的可以是内部调制解调器，也可以是连接 COM 串行口的外部调制解调器。
连接图如下所示。



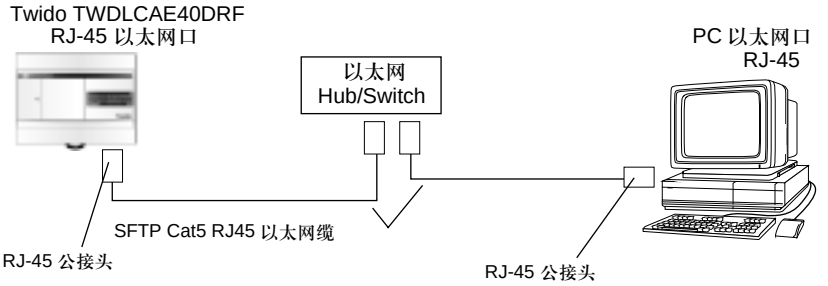
注意：只允许一个调制解调器与控制器的端口 1 相连。

注意：警告。请注意安装调制解调器所提供的驱动程序软件，因 TwidoSoft 只考虑已安装的调制解调器。

以太网连接

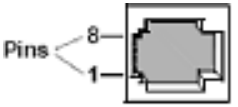
注意：尽管 Twido TWDLCAE40DRF 和运行 TwidoSoft 编程软件的 PC 之间的连接支持使用电缆 （以太网交叉电缆）直接连接，但是我们不推荐使用此方式。因此，用户应该通过集线器 / 交换机连接到以太网

下图显示了使用以太网集线器 / 交换机的 PC 到 Twido 控制器的连接：



注意：运行 TwidoSoft 应用程序的 PC 必须可连接到以太网。

Twido TWDLCAE40DRF 有一个 RJ-45 连接器可以连接到 100 BASE-TX 以太网上，并带有自协商功能。它能支持 100Mbps 和 10 Mbps 的网络速度。
下图为 Twido 控制器的 RJ-45 连接器：



RJ-45 连接器的八个引脚垂直分布，编号顺序由下至上。 RJ-45 连接器的引出如下表所示：

引出	功能	极性
8	NC	
7	NC	
6	RxD	(-)
5	NC	
4	NC	
3	RxD	(+)

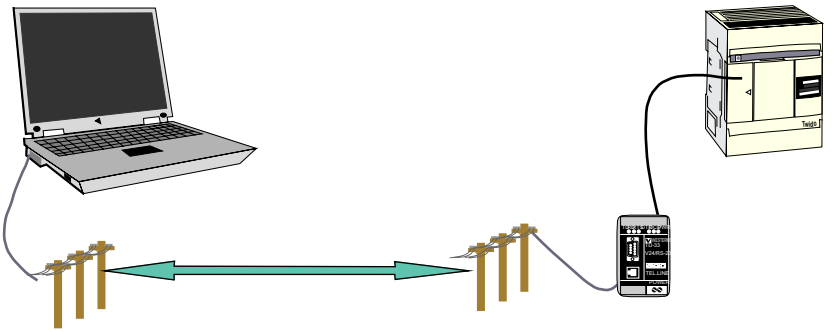
引出	功能	极性
2	TxD	(-)
1	TxD	(+)

注意：
- 同样的连接器和引出可用在 10Base-T 和 100Base-TX 网络中。
- 在将Twido控制器连接到100Base-TX网络上时，至少应该使用5类以太网电缆。

TwidoSoft 和调制解调器间通信

概览

运行 Twidosoft 的个人计算机可以连接到 Twido 控制器上，以转移应用程序，动态监控程序和执行操作模式命令。 Twido 控制器也可连接到其他设备上，如另一个 Twido 控制器，以便通过程序进程建立通信。

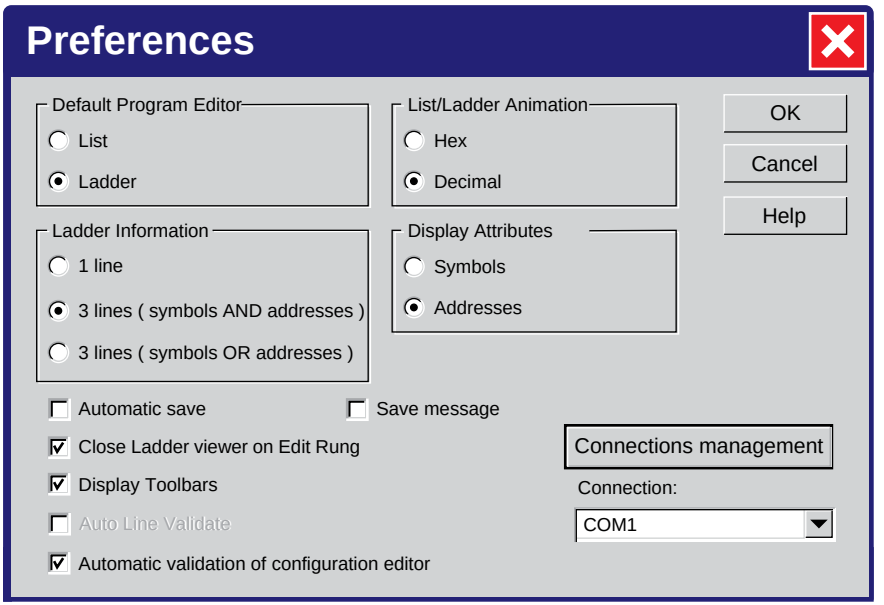


安装调制解调器

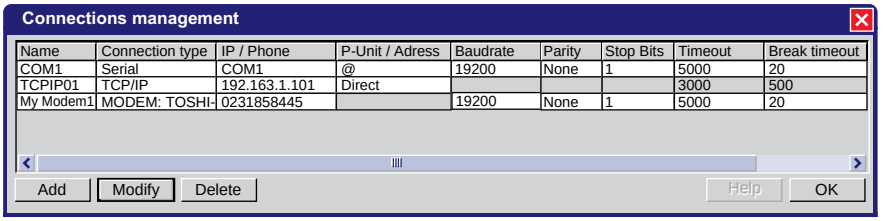
用户希望能和 Twidosoft 进行通信的调制解调器，必须安装在运行 Windows 的个人计算机上。
在 Windows 下安装调制解调器，请遵照 Windows 文档说明。
此安装与 Twidosoft 无关。

建立连接

Twidosoft 和 Twido 控制器的默认通信连接由串行通信端口完成，使用 TSX PCX 1031 电缆和交叉适配器 （见*附录 1，p.110*）。
如果调制解调器用来连接个人计算机，则必须在 Twidosoft 软件中指出。
通过 Twidosoft 选择连接，点击 “文件”，然后 “参数”。



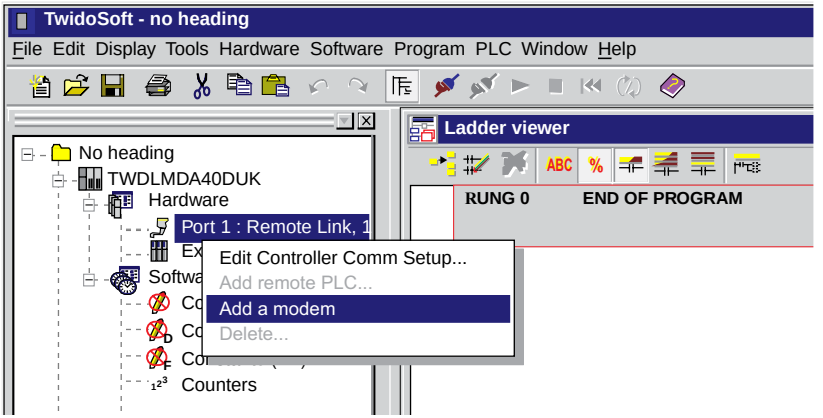
在此屏幕下可以选择连接或管理连接 （建立，修改，等等）。
使用已存在的连接，则从下拉列表框中进行选择。
如果要填加，修改或删除连接，点击 “管理连接”，则所有连接及其属性的列表会在新的窗口中出现。



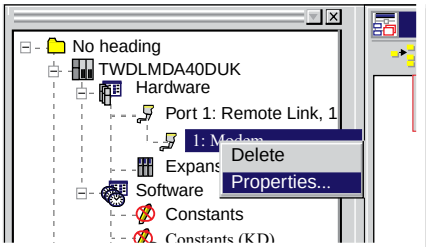
在这种情况下，显示出两个通讯连接端口 （COM1 和 TCPIP01），同时显示一个调制解调器连接，为 TOSHIBA V.90 型，通过配置组成编号：0231858445 （内部调用）。
您可以为了应用维护需要修改每个连接的名称 （COM1 不能修改）。
以上即为如何定义和选择个人计算机和调制解调器之间连接。
然而，这仅仅是在计算机和 Twido 控制器间建立连接全部过程的一部分。
下一部包括 Twido 控制器。远程 Twido 必须连接到调制解调器上。
所有的调制解调器必须被初始化以建立连接。 Twido 控制器包含至少 V2.0 版本固件，在上电时，如果调制解调器在应用中已配置，则 Twido 控制器会向调制解调器发送初始化串。

调制解调器的配置

Twido 控制器中调制解调器的配置过程如下：

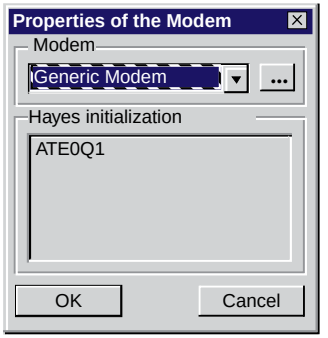


一旦调制解调器在端口 1 被配置，其属性必须被定义。右击调制解调器显示选项“删除”或“属性”。点击“属性”，可以选择已知调制解调器，新建或修改调制解调器。

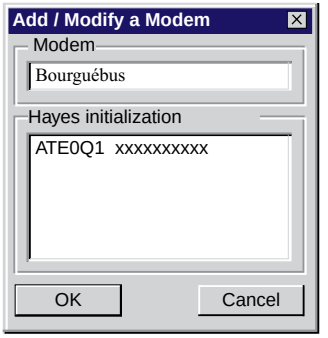


注意：调制解调器由 Twido 控制器通过端口 1 进行管理。这意味着可以在通信端口 1 上连接调制解调器，但是所有的调制解调器的工作模式和其初始化队列必须被手工操作，并且不能和端口 1 的配置重复。

第二步，选择 “属性”，然后：



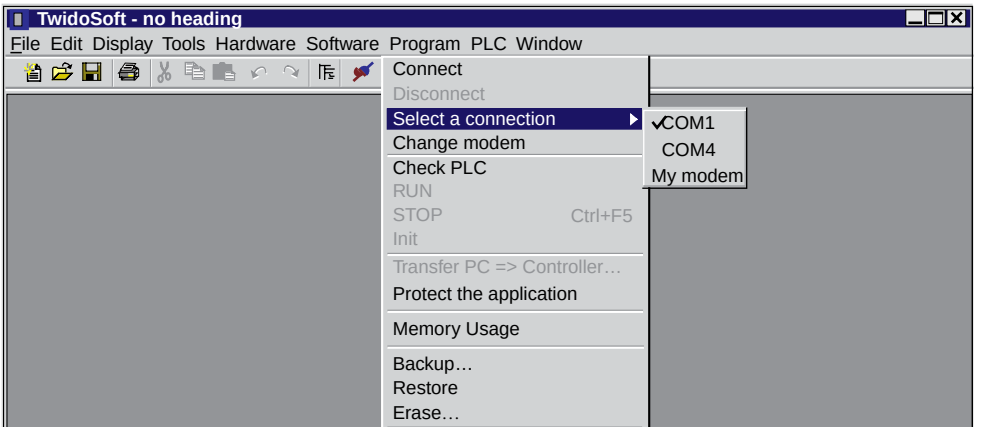
可以选择一个预定义的调制解调器，或点击 “...” 建立一个新的。



然后给新的文件命名，并按照调制解调器文档进行初始化。
“xxxxxx”代表了必须进入的初始化队列以使调制解调器准备适当的通信操作，如，波特率，奇偶校验，停止位，和接收模式。
完成此序列，请参阅调制解调器文档。
最大字长：127 个字符。
当应用程序已完成，或至少通信端口 1 已完全描述，使用 “点到点连接”传输应用程序。
这样， Twido 控制器已经可以通过调制解调器连接到个人计算机上来执行 Twidosoft。

连接队列

一旦 Twidosoft 和 Twido 控制器准备好，建立连接如下：

步骤	动作
1	Twido 控制器和调制解调器上电。
2	开启计算机并运行 Twidosoft。
3	选择 “PLC” 菜单，然后点击 “选择一个连接”，并选择 “我的调制解调器”（或您为您的调制解调器连接的命名，见 “创建连接”）。 
4	连接 TwidoSoft

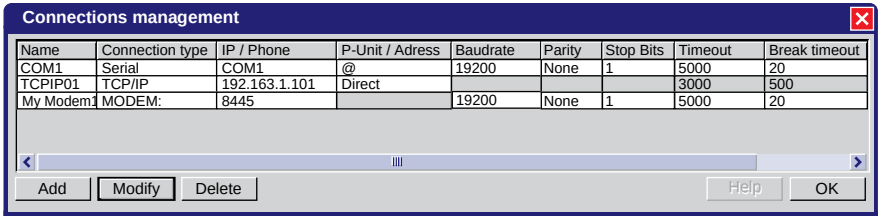
注意：如果希望一直使用调制解调器连接，点击 “文件”，“参数”，并选择 “我的调制解调器”（或您为您的调制解调器连接的命名）。 Twidosoft 将会记住您的选择。

运行模式

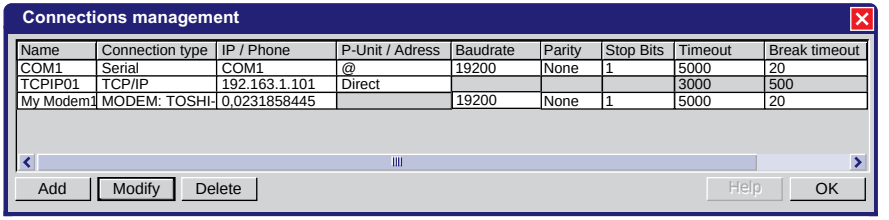
Twido 控制器将初始化串发送给已连接的，上电状态下的调制解调器。当调制解调器在 Twido 应用中被配置，则控制器首先发送 “FF” 命令来测试调制解调器是否被连接。如果控制器接到应答，则初始化串发送到调制解调器。

内线，外线和国际长途

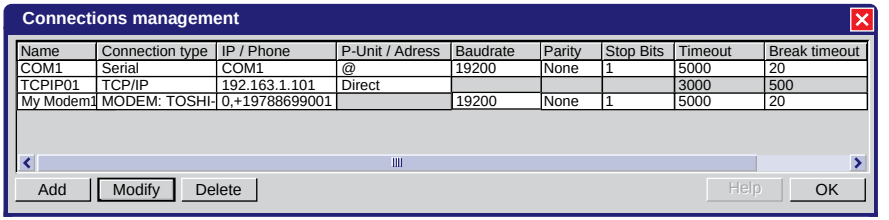
如果和公司内部的 Twido 控制器进行通信，只需使用分机号进行拨号，如：8445



如果通过外线交换台进行对公司外拨号且首先拨 “0” 或 “9”，则使用以下语法：0,0231858445 或 9,0231858445



对于国际长途，语法为：举例为 +19788699001 若通过交换台则为：0,+19788699001



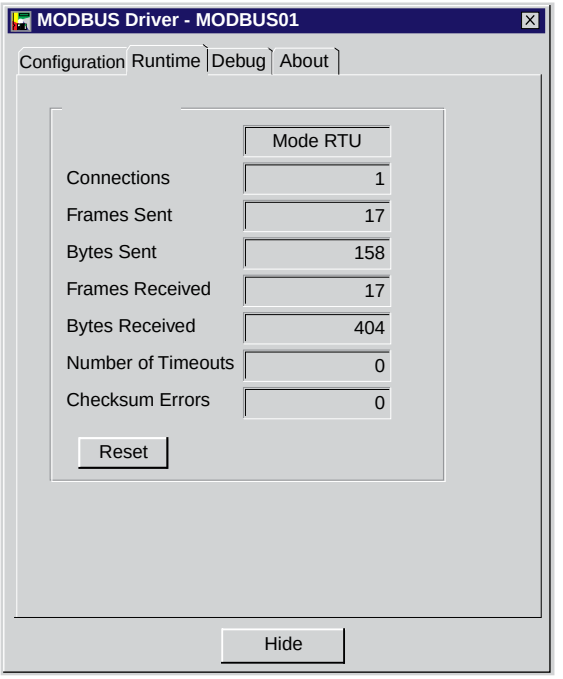
常见问题

当通信已经建立了一段时间时，一些通信错误可能出现。如若错误发生，则必须调整通信参数。

Twidosoft 使用 modbus 驱动通过串口或内部调制解调器来通信。当通信开始时，modbus 驱动在工具栏中可见。双击 modbus 驱动图标打开窗口。访问 modbus 驱动参数，“运行时间”提供了有关与远程控制器帧交换的信息。

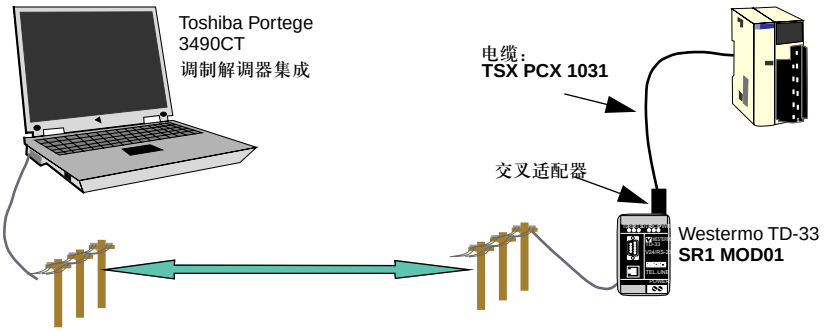
如果 “超时次数”增加或不是 1，用 “连接管理”改变其值，通过 Twidosoft 点击 “文件”然后 “参数”，选择 “连接管理”。点击 “超时”，然后按修改按钮，输入一个新的更大的值。默认值为 “5000”微秒。

利用一个新的连接重试。调整参数值直至连接稳定。



举例

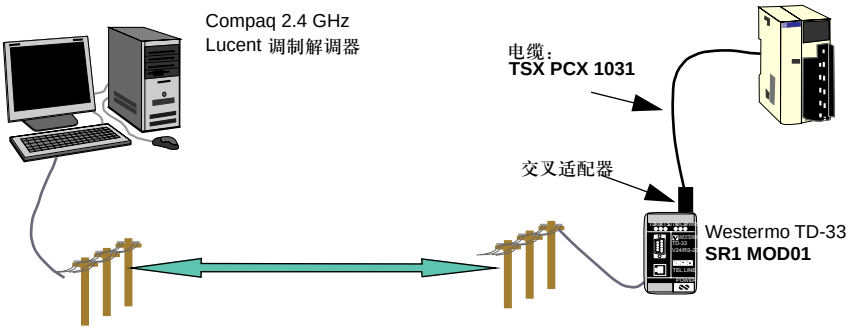
- **示例 1**：Twidosoft 连接 TWD LMDA 20DRT （Windows 98 SE） .
  - PC：Toshiba Portege 3490CT 运行 Windows 98，
  - 调制解调器 （PC 内置）：Toshiba 内置 V.90 调制解调器，
  - Twido 控制器：TWD LMDA 20DRT 2.0 版，
  - 调制解调器 （连接至 Twido）：类型 Westermo TD-33 / V.90，型号 SR1 MOD01，于新 Twido 目录 （03 年九月）（见*附录 2，p.111*），
    （**仅对北美客户**：modem 型号是 TD-33/V.90 US），
  - 电缆：TSX PCX 1031，连接到 Twido 通信端口 1，和适配器：9 针公 /9 针母，为了在 Westermo 调制解调器和 Twido 控制器连接中交叉 Rx 和 Tx （见*附录 1，p.110*）。也可使用 TSX PCX 1130 电缆 （RS485/232 反转和 Rx/Tx 交叉）。



第一个测试使用 2 根公司内部模拟电话线，不使用全部号码，只用分机号。
对于本测试，连接参数 （Twidosoft 菜单 “参数”然后 “连接管理”）使用设置，超时为 5000 超时返回为 20。

- **示例 2**：Twidosoft 连接 TWD LMDA 20DRT （windows XP Pro）
  - PC：Compaq Pentium 4， 2.4GHz，
  - 调制解调器：Lucent Win modem， PCI bus，
  - Twido 控制器：TWD LMDA 20DRT 2.0 版，
  - 调制解调器 （连接至 Twido）：类型 WESTERMO TD-33 / V.90，型号 SR1 MOD01，于新 Twido 目录 （03 年九月）（见附录 2 第 111 页）
    （**仅对北美用户**：在您当地可得到的 Modem 型号为 TD-33/V.90US）

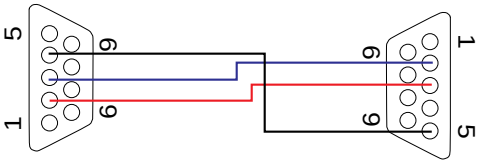
- 电缆：TSX PCX 1031，连接到 Twido 通信端口 1，和适配器：9 针公 / 9 针母，为了在 Westermo 调制解调器和 Twido 控制器连接中交叉 Rx 和 Tx （见附录 1， p.110）。也可使用 TSX PCX 1130 电缆 （RS485/232 反转和 Rx/Tx 交叉）。



第一个测试使用 2 根公司内部模拟电话线，不使用全部号码，只用分机号。（因此对于内部东芝 V.90 调制解调器的电话号码只有 4 位）。
对于本测试，连接参数 （Twidosoft 菜单 “参数”然后 “连接管理”）使用设置，超时为 5000 超时返回为 20。

附录 1

交叉适配器，适用于 TSX PCX 1031 和 Westermo TD-33 调制解调器：



附录 2

调制解调器 Westermo TD-33，Schneider 参考值 **SR1 MOD01** (1)。此调制解调器可管理四 DIP 开关，必须设置为 **OFF**：

厂方设置

使用储存配置 （速度和格式等等）
DTR 无效，自动波特率

注意：
1. 某些产品可能不兼容，请咨询当地施耐德办事处。

附录 3

Wavecom WMOD2B 调制解调器，Schneider 参考值 SR1 MOD02 (1) 双波 （900/1800Hz）：

注意：
1. 某些产品可能不兼容或在所有区域无效，请咨询当地施耐德办事处。

附录 4

本文档中使用的产品参考值：

- Twido 产品：TWD LMDA 20DRT，
- Twidosoft 软件：TWD SPU 1002 V10M，
- TSX PCX 1031 电缆，
- TSX PCX 1130 电缆，
- RTU 调制解调器：Westermo TD-33 / V90 **SR1 MOD01** (1)，
- GSM 调制解调器：Wavecom WMOD2B **SR1 MOD02** (1).

注意：
1. 某些产品可能不兼容或在所有区域无效，请咨询当地施耐德办事处。

远程连接通信

介绍

远程连接协议是一种高速主 / 从总线，它支持一个主控制器和最多七个远程 （从）控制器之间的少量数据通信。根据远程控制器的配置，传送相应的应用或 I/O 数据。远程控制器的类型可以是远程 I/O 或对等控制器。

注意：主机包含有关远程 I/O 地址的信息。它不知道地址中的具体控制器。这样，主机不能确认用户程序中用到的远程输入和输出是否实际存在。注意这些远程输入或输出的实际存在。

注意：远程 I/O 总线和协议属于专用，第三方设备不允许出现在网络中。

注意

意外设备操作

- 确信远程连接中只有一个主控制器且每个从机都有唯一地址。如果不遵守这个警告将会导致数据崩溃或意料不到的结果。
- 确信所有从机都有唯一地址。没有两个从机具有相同地址。如果不遵守这个警告将会导致数据崩溃或意料不到的结果。

如果不遵守这个警告将会导致人身伤害或设备损害。

注意：远程连接需要 EIA RS-485 连接，并且同时只能有一个通信端口被配置为远程连接。

硬件配置

一个远程连接必须使用最少 3- 线的 EIA RS-485 端口。通过配置，可是用第一个端口或第二个端口，如果存在第二个端口的话。

注意：一次只能有一个通信端口配置成远程连接。

下表列出了可以使用的设备：

远程	端口	说明
TWDLC • A10/16/ 24DRF, TWDLCA•40DRF, TWDLMDA20/40DUK, TWDLMDA20/40DTK, TWDLMDA20DRT	1	基本控制器用 miniDIN 连接器装置一个 3- 线 EIA RS-485 端口。
TWDNOZ485D	2	通信模块用miniDIN连接器装置一个3-线EIA RS-485端口。 **注意：**该模块只对模块型控制器有用。附加该模块后，控制器不能有操作显示扩展模块。
TWDNOZ485T	2	通信模块用一个端子块装置一个 3- 线 EIA RS-485 端口。 **注意：**该模块只对模块型控制器有用。附加该模块后，控制器不能有操作显示扩展模块。
TWDNAC485D	2	通信适配器用 miniDIN 连接器装置一个 3- 线 EIA RS-485 端口。 **注意：**该适配器只对一体型 16、 24 I 和 40 I/O 控制器及操作显示扩展模块有用。
TWDNAC485T	2	通信适配器用一个端子块装置一个 3- 线 EIA RS-485 端口。 **注意：**该适配器只对一体型 16、 24 和 40 I/O 控制器及操作显示扩展模块有用。
TWDXCPODM	2	操作显示扩展模块用miniDIN连接器装置一个 3- 线 EIA RS-485 端口或用一个端子块装置一个 3- 线 EIA RS-485 端口。 **注意：**该模块只对模块型控制器有用。附加该模块后，控制器不能有通信扩展模块。

注意：端口 2 的存在和配置 （RS232 或 RS485）只在上电时被检查，或被固件可执行程序重置。

每个设备的电缆连接

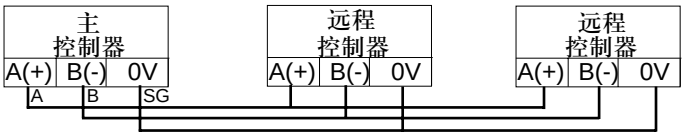
注意：引脚 5 的 DPT 信号必须与引脚 7 的 0V 相连以表示远程连接通信使用。当此信号不与地相接时，控制器无论主机或从机都默认到编程模式，并试图与 TwidoSoft 建立通信。

注意：DPT 与 0V 相连仅为连接到主控制器的端口 1 的必要条件。

每个设备的电缆连接图如下。

Mini-DIN 接头

端子排连接



软件配置

远程连接中只能定义一个主控制器。另外，每个远程控制器必须具有唯一的从机地址。多个主机或从机使用同样地址将会导致传输失败或造成意外。

注意

意外设备操作

确信远程连接中只有一个主控制器且每个从机具有唯一地址。如果不遵守这个警告将会导致数据崩溃或意料不到的结果。

如果不遵守这个警告将会导致人身伤害或设备损害。

主控制器配置

主控制器由 TwidoSoft 配置，管理最多有七个远程控制器的远程连接网络。这七个远程控制器可配置成远程 I/O 或对等控制器。由 TwidoSoft 配置的主机地址对应着地址 0。

为了将一个控制器配置成主控制器，需用 TwidoSoft 将端口 1 或端口 2 配置成远程连接且选择地址 0 （主机）。

然后，从 “添加远程控制器”窗口，您能指定从控制器为远程 I/O 或对等控制器以及它们的地址。

远程控制器配置

通过 TwidoSoft 配置端口 1 或 2 为远程连接或分配地址 1 到 7 到端口，完成远程控制器的配置。

下表概括了各种控制器配置的不同和限制：

类型	应用程序	数据访问
远程 I/O	没有 甚至没有简单的 “END” 声明运行模式与主机相连。	%I 和 %Q 仅控制器的本地 I/O 可供访问（I/O 扩展不可以）。
对等控制器	有 运行模式与主机无关。	%INW 和 %QNW 每个对等控制器可传输一个最多 4 个字的输入和 4 个字的输出。

远程控制器扫描同步

远程连接的更新循环与主控制器的扫描不同步。与远程控制器的通信由中断驱动且作为与主控制器的扫描运行平行的后台任务发生。扫描循环结束时，最新的数值读进程序数据，被下次程序执行使用。远程 I/O 和对等控制器有着相同的处理。

所有控制器可通过系统位 %S111 检查总的连接活动。为了达到同步，主机或对等控制器必须使用系统位 %S110。当一个完整的更新循环发生时，这个位被置为 1。应用程序负责将它重置为 0。

主机可通过系统位 %S112 使远程连接有效或无效。控制器可通过 %S113 检查配置的正确性并更正操作。在 %S100 系统读取并报告端口 1 上的 DPT 信号 （用于决定 TwidoSoft 是否被连接）。

系统位	状态	表示
%S100	0	主 / 从：DPT 非活动的 （TwidoSoft 电缆没有连接）
	1	主 / 从：DPT 活动的 （TwidoSoft 电缆连接）
%S110	0	主 / 从：被程序置为 0
	1	主：所有远程连接交换完成 （仅远程 I/O） 从：和主机交换完成
%S111	0	主：一个远程连接交换完成 从：一个远程连接交换被检测
	1	主：一个远程连接交换正在处理 从：一个远程连接交换被检测
%S112	0	主：不能远程连接
	1	主：可以远程连接
%S113	0	主 / 从：远程连接配置 / 操作正确
	1	主：远程连接配置 / 操作错误 从：远程连接操作错误

主控制器重启

如果主控制器重启，将会发生下列某个事件：

- 冷启动 （%S0 = 1）强制通信重新初始化。
- 热启动 （%S1 = 1）强制通信重新初始化。
- 在停止模式，主机继续和从机通信。

从控制器重启

如果从控制器重启，将会发生下列某个事件：

- 冷启动 （%S0 = 1）强制通信重新初始化。
- 热启动 （%S1 = 1）强制通信重新初始化。
- 在停止模式，从机继续和主机通信。如果主机指示停止状态；
  - 远程 I/O 应用停止状态。
  - 对等控制器继续它的当前状态。

主控制器停止

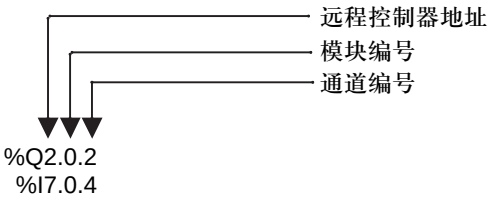
当主控制器进入停止模式，所有的从设备继续与主机通信。当主机指示停止状态时，远程 I/O 将停止，但是对等控制器继续它们当前的运行或停止状态。

远程 I/O 数据访问

配置为远程 I/O 的远程控制器没有也不执行自己的应用程序。远程控制器本体的数字输入和输出只是主控制器的扩展。应用程序必须也只能使用三位数字的寻址方式。

注意：远程 I/O 的模块号一般为 0。

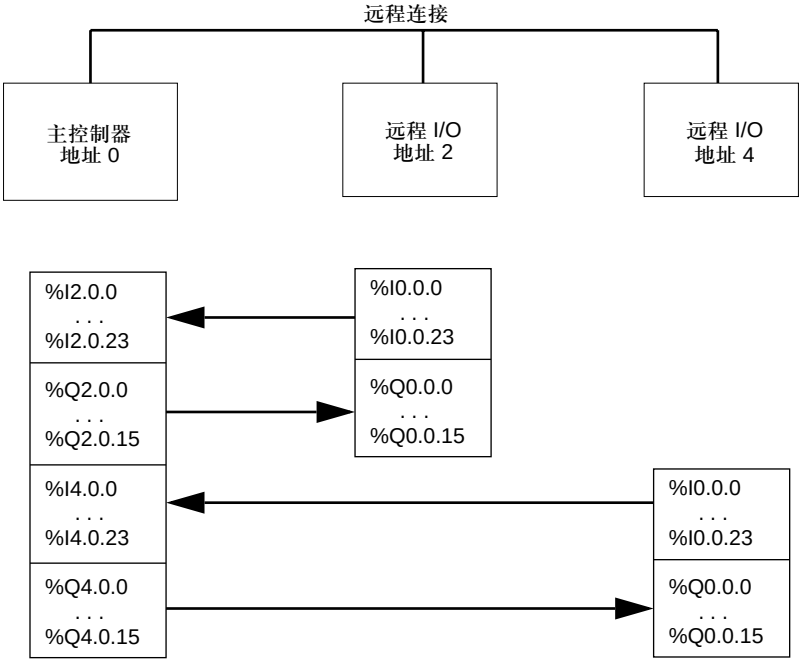
图例



为与远程 I/O 通信，主控制器使用标准输入和输出符号 %I 和 %Q。指令 %Q2.0.2 可以输出到地址为 2 的远程 I/O 的第三个输出位。类似的，指令 %I7.0.4 为读取 7 号位置的远程 I/O 的第五个输入位。

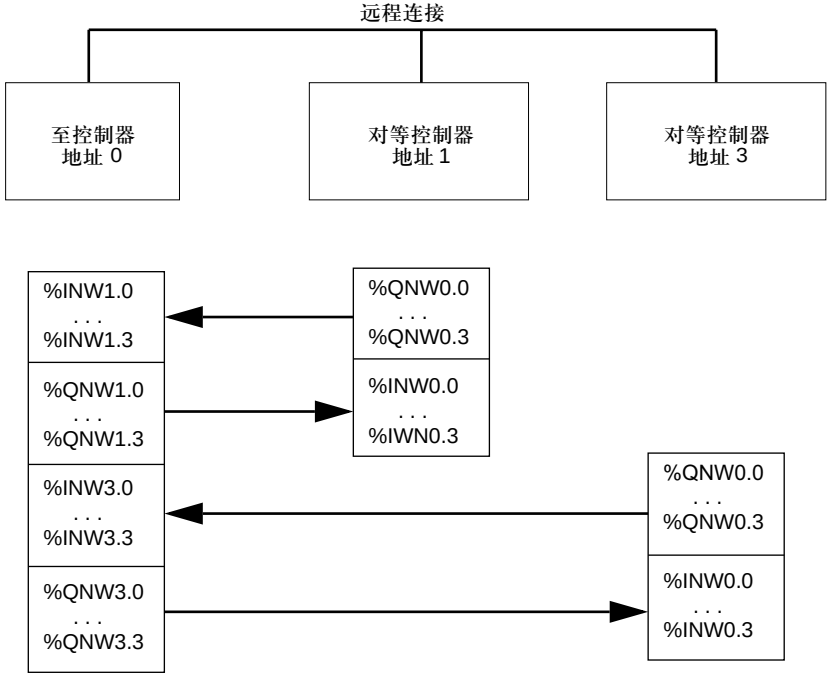
注意：主机限定为只能访问数字 I/O，这些 I/O 只是远程控制器本体 I/O 的一部分。模拟和扩展 I/O 不能被传递，除非使用对等通信。

图例


远程连接
主控制器
地址 0
远程 I/O
地址 2
远程 I/O
地址 4
%I2.0.0
. . .
%I2.0.23
%Q2.0.0
. . .
%Q2.0.15
%I4.0.0
. . .
%I4.0.23
%Q4.0.0
. . .
%Q4.0.15
%I0.0.0
. . .
%I0.0.23
%Q0.0.0
. . .
%Q0.0.15
%I0.0.0
. . .
%I0.0.23
%Q0.0.0
. . .
%Q0.0.15

对等控制器数据访问

为与对等控制器通信，主机用网络字 %INW 和 %QNW 交换数据。网络中每个对等控制器由其远程地址 “j” 通过字 %INWj.k 和 %QNWj.k 被访问。每个对等控制器使用 %INW0.0 到 %INW0.3 和 %QNW0.0 到 %QNW0.3 访问主机数据。控制器在运行或停止模式下网络字被自动更新。
下面是一个主机与两个对等控制器数据交换的图例。



远程连接中没有对等消息。主机应用程序可用来管理网络字，为了实现远程控制器间的信息传递，可将主机作为桥梁使用。

状态信息

除了前面解释的系统位，主机还保存当前状态和远程控制器配置。这由系统字 %SW111 和 %SW113 来完成。远程控制器和主机都能从系统字 %SW112 获得远程连接通信时最近一次出错值。

系统字	用法	
%SW111	远程连接状态：每个远程控制器两位 （仅主机）	
	x0-6	0- 远程控制器 1-7 不存在
		1- 远程控制器 1-7 存在
	x8-14	0- 远程控制器 1-7 检测为远程 I/O
		1- 远程控制器 1-7 检测为对等控制器
%SW112	远程连接配置 / 操作错误码	
		0- 操作成功
		1- 检测到超时 （从机）
		2- 检测到校验位出错 （从机）
		3- 配置不匹配 （从机）
%SW113	远程连接配置：每个远程控制器两位 （仅主机）	
	x0-6	0- 远程控制器 1-7 没有被配置
		1- 远程控制器 1-7 已被配置
	x8-14	0- 远程控制器 1-7 配置为远程 I/O
		1- 远程控制器 1-7 配置为对等控制器

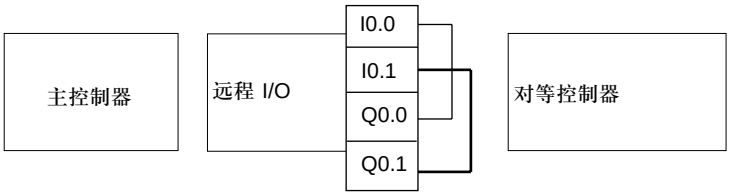
远程连接举例

为配置一个远程连接，您必须：

1. 配置硬件。
2. 连接控制器。
3. 连接 PC 和控制器间的通信电缆。
4. 配置软件。
5. 写应用程序。

下面是远程 I/O 与一个对等控制器远程连接使用图例。

步骤 1：配置硬件：



硬件配置是三个任意类型的基本控制器。端口 1 用于两种通信模式。一种模式通过 TwidoSoft 配置和传输应用程序。另一种模式用于远程连接网络。如果可行，可以使用控制器的可选端口 2，但是一个控制器只支持一个远程连接。

注意：此例中，远程 I/O 的前两个输入口和前两个输出口用硬连线连接。

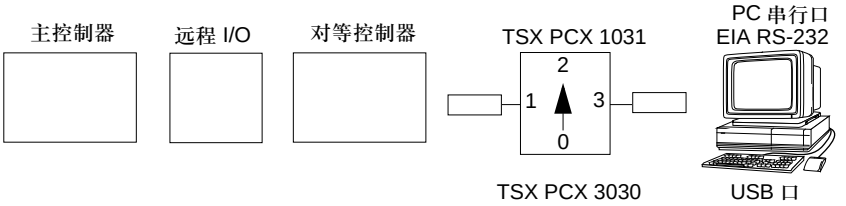
步骤 2：连接控制器

Mini-DIN 接头

端子排连接

将 A （+）和 B （-）信号线分别连在一起。并且每个控制器的 DPT 信号与地相接。虽然使用端口 2 （可选卡或通信模块）进行远程连接时不需要此信号与地相接，不过最好养成这个好习惯。

步骤 3：连接 PC 和控制器间的通信电缆：



多功能可编程电缆 TSXPCX1031 或 TSX PCX 3030 用于三个基本控制器的通信。确信电缆上的开关处于位置 2。为了对每个控制器编程，每个控制器都需建立点对点通信。这样建立这种通信：与第一个控制器的端口 1 连接，传递配置和应用数据，然后将控制器设置成运行状态。对每个控制器重复这个过程。

注意：每个控制器在配置和应用传递完毕后需要移除电缆。

步骤 4：配置软件：
三个控制器均用 TwidoSoft 创建配置，和应用程序，如果可以的话。
对于主控制器，在控制器通信安装编辑中设置协议为 “远程连接”，地址为 “0（主机）”。

控制器公共设置
类型: 远程连接
地址: 0（主站）

主机通过添加 “远程 I/O” 地址 “1” 和 “对等 PLC” 地址 “2” 配置远程控制器。

添加远程控制器

控制器使用: 远程 I/O
远程地址: 1

控制器使用: 对等控制器
远程地址: 2

对于配置成远程 I/O 的控制器，确认控制器通信安装设置成 “远程连接” 且其地址设为 “1”。

控制器公共设置
类型: 远程连接
地址: 1

对于配置成对等机的控制器，确认控制器通信安装设置成 “远程连接” 且其地址设为 “2”。

控制器公共设置
类型: 远程连接
地址: 2

步骤 5: 写应用程序：
对主控制器，写如下应用程序：

```
LD 1

[%MW0 := %MW0 +1]
[%QNW2.0 := %MW0]
[%MW1 := %INW2.0]

LD %I0.0
ST %Q1.0.0
LD %I1.0.0
ST %Q0.0

LD %I0.1
ST %Q1.0.1
LD %I1.0.1
ST %Q0.1
```

对配置成远程 I/O 的控制器，不用写任何应用程序。
对配置成对等机的控制器，写如下应用程序：

```
LD 1
[%QNW0.0 := %INW0.0]
```

此例中，主机程序对一个内部存储字做自加运算且用一个网络字传递给对等控制器。对等控制器从主机接收该字并将其返回。主机将接收从机返回的字放入一个不同的存贮字并保存这次传输。
为与远程 I/O 控制器通信，主机将自己本地输入传送给远程 I/O 输出。当远程 I/O 外部 I/O 硬连线时，主机可返回并重新得到信号。

ASCII 通信

介绍

ASCII 协议为 Twido 控制器提供了一个半双工字符模式协议，用于和一个设备传输和 / 或接收数据。该协议由 EXCHx 指令支持，由 %MSGx 功能模块控制。
ASCII 协议提供了三种通信方式：

- 只发送
- 发送 / 接收
- 只接收

EXCHx 指令发送和 / 或接收帧的最大值是 256 字节。

硬件配置

ASCII 连接 （见*系统位 %S103* 和 %S104，（见*系统位 （%S）*，*p.658*）可以通过 EIA RS-232 或 EIA RS-485 端口建立，并且可以同时在两个通信端口上运行。
下表列出了可以使用的设备：

远程	端口	说明
TWDLC • A10/16/24DRF, TWDLCA•40DRF, TWDLMDA20/40DTK, TWDLMDA20DRT	1	基本控制器用 miniDIN 连接器装置一个 3- 线 EIA RS-485 端口。
TWDNOZ232D	2	通信模块用 miniDIN 连接器装置一个 3- 线 EIA RS-232 端口。 **注意：**该模块只对模块型控制器有用。附加该模块后，控制器不能有操作显示扩展模块。
TWDNOZ485D	2	通信模块用 miniDIN 连接器装置一个 3- 线 EIA RS-485 端口。 **注意：**该模块只对模块型控制器有用。附加该模块后，控制器不能有操作显示扩展模块。
TWDNOZ485T	2	通信模块用一个端子块装置一个 3- 线 EIA RS-485 端口。 **注意：**该模块只对模块型控制器有用。附加该模块后，控制器不能有操作显示扩展模块。
TWDNAC232D	2	通信适配器用 miniDIN 连接器装置一个 3- 线 EIA RS-232 端口。 **注意：**该适配器只对一体型 16、 24 和 40 I/O 控制器及操作显示扩展模块有用。
TWDNAC485D	2	通信适配器用 miniDIN 连接器装置一个 3- 线 EIA RS-485 端口。 **注意：**该适配器只对一体型 16、 24 和 40 I/O 控制器及操作显示扩展模块有用。
TWDNAC485T	2	通信适配器用一个端子块装置一个 3- 线 EIA RS-485 端口。 **注意：**该适配器只对一体型 16、 24 和 40 I/O 控制器及操作显示扩展模块有用。

远程	端口	说明
TWDXCPODM	2	操作显示扩展模块用 miniDIN 连接器装置一个 3- 线 EIA RS-232 端口，用 miniDIN 连接器装置一个 3- 线 EIA RS-485 端口或用一个终端装置一个 3- 线 EIA RS-485 端口。注意：该模块只对模块型控制器有用。附加该模块后，控制器不能有通信扩展模块。

注意：端口 2 的存在和配置 （RS232 或 RS485）只在上电时被检查，或被固件可执行程序重置

定义电缆

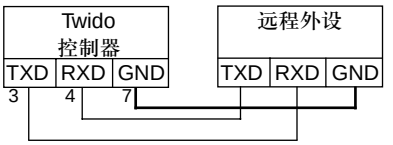
下面是 EIA RS-232 和 EIA RS-485 型的电缆连接定义图。

注意：如果 Twido 控制器使用端口 1， 5 号引脚的 DPT 信号必须与 7 号引脚的 0V 相接。这意味着 Twido 控制器的端口 1 通信是 ASCII 而不是和 TwidoSoft 软件通信。

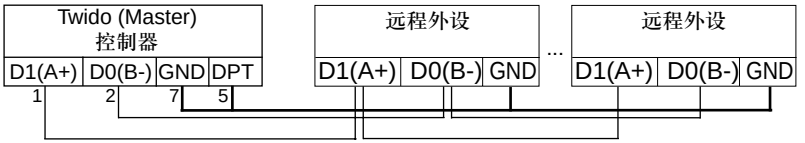
每个设备的电缆连接图如下。

Mini-DIN 接头

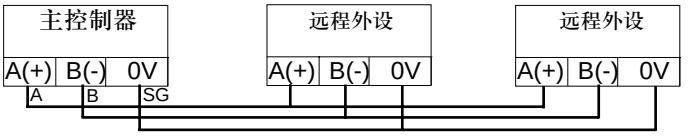
RS-232 EIA 电缆



RS-485 EIA 电缆



端子排连接



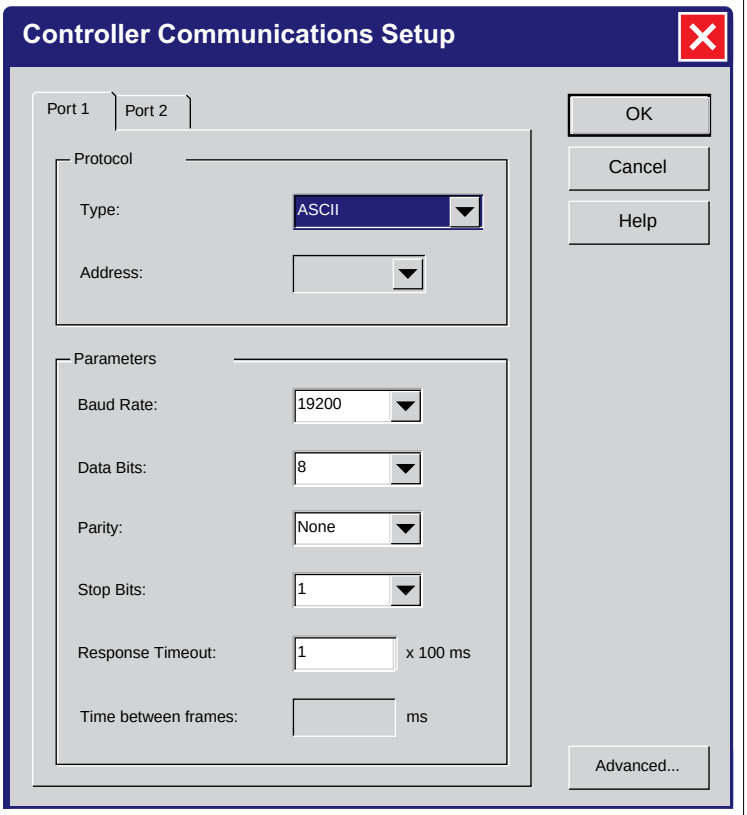
软件配置

为了配置控制器使用 ASCII 协议通过串行接口发送和接收字符，您必须：

步骤	描述
1	用 TwidoSoft 配置 ASCII 串行口。
2	在应用程序中创建发送 / 接收表以供 EXCHx 指令使用。

配置端口

Twido 控制器可用基本端口 1 或可选端口 2 来使用 ASCII 协议。配置 ASCII 串行口：

步骤	动作
1	定义本地控制器配置的附加通信适配器或模块。
2	右键点击端口再点击**控制器通信编辑…** **结果：**出现下窗口。 
3	从 “协议类型” 列表框中选择 ASCII 协议类型。
4	设置相关通信参数。
5	点击高级按钮设定高级参数。

ASCII 模式发送 / 接收表配置

发送和 / 或接收帧的最大值是 256 字节。与 EXCHx 指令相关的字表由发送和接收控制表组成。

	高字节	低字节
控制表	命令	长度 （发送 / 接收）
	保留 （0）	保留 （0）
发送表	发送字节 1	发送字节 2
	...	...
		发送字节 n
	发送字节 n+1	
接收表	接收字节 1	接收字节 2
	...	...
		接收字节 p
	接收字节 p+1	

控制表

此**长度**字节包含发送表的长度 （最大 250 字节），如果接收被请求，它将被接收结束时收到的字符数覆盖。

此**命令**字节必须包含下面之一：

- 0：只发送
- 1：发送 / 接收
- 2：只接收

发送 / 接收表

在只发送模式，控制和发送表在 EXCHx 指令执行之前被填写，类型可以是 %KW 或 %MW。只发送模式不需要接收字符空间。一旦所有字节发送完毕， %MSGx.D 将置为 1，然后可以执行新的 EXCHx 指令。
在发送 / 接收模式，控制和发送表在 EXCHx 指令执行之前被填写，并且类型必须是 %MW。发送表的末端需要最多 256 接收字节的空间。一旦所有字节发送完毕，Twido 控制器转换到接收模式并等待接收任何字节。
在只接收模式，控制表在 EXCHx 指令执行之前被填写，并且类型必须是 %MW。控制表的末端需要最多 256 接收字节的空间。 Twido 控制器立即进入接收模式并等待接收任何字节。
当收到帧结束字节或接收表满时接收结束。这种情况下，字 %SW63 和 %SW64 出现错误 （接收表溢出）。如果配置了一个非零停止时间，接收在停止时间到时也将结束。如果停止时间值选为零，则没有接收停止时间。这样为了停止接收，必须激活 %MSGx.R 输入。

消息交换

语言提供了两种通信服务：

- **EXCHx 指令**：发送 / 接收消息
- **%MSGx 功能模块**：控制消息交换。

Twido 控制器处理 EXCHx 指令时使用端口配置协议。

注意：每个通信端口可配置不同或相同协议。通过设置端口号 （1 或 2）路径 EXCHx 指令和 %MSGx 功能模块可以访问每个通信端口。

EXCHx 指令

EXCHx 指令允许 Twido 控制器发送和 / 或接收信息到 / 自 ASCII 设备。用户定义一个字表 （%MWi:L 或 %KWi:L），其中包含用来发送和 / 或接收的控制信息和数据（发送和 / 或接收最多 256 字节）。字表格式如前描述。
使用 EXCHx 指令完成消息交换：

句法：[EXCHx %MWi:L]
其中：x = 端口号 (1 或 2)
L = 在控制字、传输和接收表中的字数

Twido 控制器必须在第二条 EXCHx 指令执行之前由第一条指令完成交换。发送不止一条消息时，必须使用 %MSGx 功能模块。
当任何传输处于中断控制情形时 （数据接收也处于中断控制情形），EXCHx 列表指令的处理立即开始，并被视为后台处理。

%MSGx 功能模块

%MSGx 功能模块的使用不是必需的；它能用于管理数据交换。 %MSGx 功能模块有三种用途：

- **通信错误校验**
  错误校验确认 EXCHx 指令编程参数 L （字表长度）足够大，能包含发送消息的长度。与存储在字表第一个字的低字节中的长度相比较。
- **多消息协调**
  %MSGx功能模块提供前面消息传输完成的时间信息，以保证多消息发送的协调。
- **传输优先消息**

%MSGx 功能模块允许当前消息传输停止以发送紧急消息。

%MSGx 功能模块有一个输入，两个输出：

输入 / 输出	定义	描述
R	输入复位	置为 1：通信重新初始化或模块重置（%MSGx.E = 0 和 %MSGx.D =1）。
%MSGx.D	通信完成	0：程序请求。 1：通信完成，如果：传输完毕，字符接收完毕，出错，或模块重置。
%MSGx.E	错误	0：消息长度正确且连接正确。 1：命令错误，表配置错误，接收字符错误（速率，奇偶校验，等等。），或接收表满。

限制

请务必注意下列限制：

- 端口 2 的可用性和类型 （见 %SW7）只在上电或重置时被检查
- 当连接 TwidoSoft 时端口 1 将不能进行任何消息处理
- EXCHx 和 %MSGx 不能被配置为远程连接的端口所处理
- EXCHx 将停止 Modbus 从机处理
- EXCHx 指令的处理在错误事件中不会得到重试
- 输入复位 （R）可用来中断 EXCHx 指令接收处理
- EXCHx 指令可配置超时时间来中断接收
- 多消息通过 %MSGx.D 得到控制

错误和工作模式环境

当使用 EXCHx 指令时如果出错，位 %MSGx.D 和 %MSGx.E 将置为 1 且系统字 %SW63 包含端口 1 的错误代码， %SW64 包含端口 2 的错误代码。

系统字	用法
%SW63	EXCH1 错误代码： 0 - 操作成功 1 - 传送的子节数太大 （> 250） 2 - 发送表太小 3 - 字表太小 4 - 接收表溢出 5 - 超时时间到 6 - 发送错误 7 - 表中错误命令 8 - 所选端口没有配置 / 不可用 9 - 接收错误 10 - 接收时不能用 %KW 11 - 发送偏移量大于发送表 12 - 接收偏移量大于接收表 13 - 控制器停止 EXCH 处理
%SW64	EXCH2 错误代码：见 %SW63。

通信过程中控制器重启结果

如果控制器重启，下面事件中的一个将会发生：

- 冷启动 （%S0 = 1）强制通信重新初始化。
- 热启动 （%S1 = 1）强制通信重新初始化。
- 在停止模式，控制器停止所有 ASCII 通信。

ASCII 连接举例

为配置一个 ASCII 连接，您必须：

1. 配置硬件。
2. 连接 ASCII 通信电缆。
3. 配置端口。
4. 写应用程序。
5. 初始化动态数据表编辑器。

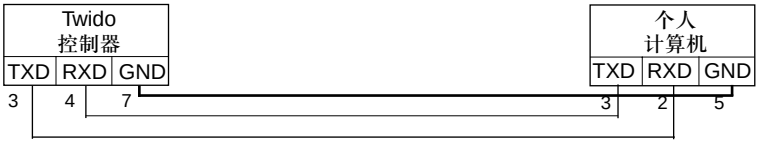
下面图描述了怎样使用 ASCII 和 PC 终端仿真器通信。

步骤 1：配置硬件：

硬件配置是指两个串行连接，从 PC 到 Twido 控制器的可选 EIA RS-232 端口 2。对于模块型控制器，可选端口 2 是指 TWDNOZ232D 或 TWDXCPODM 上的 TWDNAC232D。对于一体型控制器，可选端口 2 是 TWDNAC232D。

为了配置控制器，需连接电缆 TSXPCX1031 （图中没有显示）至 Twido 控制器的端口 1。然后连接电缆至 PC 的 COM 1 口。确信电缆开关处于位置 2。最后，连接 PC 的 COM 2 口和 Twido 控制器的可选 EIA RS-232 端口 2。连线说明见下一步。

步骤 2：ASCII 通信电缆 （EIA RS-232）连线说明：



ASCII 通信电缆的最少使用线数是 3。发送和接收信号线交叉。

注意：电缆的 PC 端，可能需要额外的连接 （如数据终端准备和数据设置准备）来满足握手信号。 Twido 控制器不需要额外连接。

步骤 3：端口配置：

硬件 -> 添加选项
TWDNOZ232D

串行口 2	
协议	ASCII
地址	
波特率	19200
数据位	8
校验	无
停止位	1
响应超时 (x100ms)	100
帧间隔时间 (ms)	
开始字符	
首个结束字符	65
第二个结束字符	
无收发停止 (ms)	
接受到字节数时停止	

PC 终端	
端口:	COM2
波特率:	19200
数据:	8 位
奇偶校验:	无
停止:	1 位
流控制:	无

用一个 PC 终端仿真程序配置 COM2 口并确保不存在流控制。
用 TwidoSoft 配置控制器的端口。首先，配置硬件选项。此例中， TWDNOZ232D 加到模块型基本控制器。
其次，用 PC 终端仿真器的所有相同参数设置初始化控制器通信安装。此例中，选择大写字母 “A” 表示 “帧结束”，用于停止字符接收。“响应超时” 参数选用 10 秒。这两个参数只有一个被调用，取决于哪个先发生。

步骤 4：写应用程序：

```
LD 1
[%MW10 := 16#0104 ]
[%MW11 := 16#0000 ]
[%MW12 := 16#4F4B ]
[%MW13 := 16#0A0D ]
LD 1
AND %MSG2.D
[EXCH2 %MW10:8]
LD %MSG2.E
ST %Q0.0
END
```

用 TwidoSoft 创建一个应用程序包括三个主要部分。首先，初始化控制和发送表以便 EXCH 指令使用。此例中，设置命令发送和接收数据。数据发送数量设为 4 字节，像在应用程序中定义的那样，由已使用过的成帧字符未尾跟着 （在这例中，首个结尾字符为 “A”）。起动和结束值不会显示在活动表上，只有数据可以显示。不管怎样，那些字符是在使用接收时 （用 %SW63 和 %SW64），自动传输和调节。然后，检查相关通信位 %MSG2，如果端口准备好就发送 EXCH2 指令。EXCH2 指令指定了一个 8 字值。它们是 2 个控制字 （%MW10 和 %MW11）， 2 个字用于传输信息 （%MW12 和 %MW13）， 4 个字接收数据 （%MW14 到 %MW16）。
最后，错误状态 %MSG2 被得到并存储于本地基本控制器 I/O 的第一个输出位。还可以使用 %SW64 进行附加的错误检查以使结果更加准确。

步骤 5：初始化动态数据表编辑器：

地址	当前	保留格式
1 %MW10	0104	十六进制
2 %MW11	0000	十六进制
3 %MW12	4F4B	十六进制
4 %MW13	0A0D	十六进制
5 %MW14	TW	ASCII
6 %MW15	ID	ASCII
7 %MW16	O	ASCII

最后一步是下载应用程序到控制器并运行它。初始化活动表编辑器以激活和显示字 %MW10 到 %MW16。字符 “O” - “K” -CR-LF 被显示到终端仿真器上。当 EXCH 模块的响应时间到且新 EXCH 模块没有开始之前，字符 “O” - “K” -CR-LF 可被多次显示。终端仿真器敲下 “T” - “W” - “I” - “D” - “O” - “A”。它们将被交换到 Twido 控制器并被活动表编辑器所显示。

Modbus 通信

介绍

Modbus 协议是一个主 - 从协议，它允许一个并且只能一个主机发送命令，查询从机的响应。主机可单独对一个从机发送命令，也可以广播方式对所有从机发送命令。从机对每一个单独发送给它们的查询返回讯息 （响应）。但对广播方式的查询不做响应。

注意

意外设备操作

- 确定在总线上只有一个 Modbus 主控制器并且每个 Modbus 从站有一个唯一的地址。如果不遵守这个警告将会导致数据崩溃或意料不到的结果。
- 确定所有 Modbus 从站有唯一的地址。没有两个从机具有相同地址。如果不遵守这个警告将会导致数据崩溃或意料不到的结果。

如果不遵守这个警告将会导致 人身伤害或设备损害。

硬件配置

Modbus 连接可以通过 EIA RS-232 或 EIA RS-485 端口建立，并且可以同时在两个通信端口上运行。每个端口可指定为自己的 Modbus 地址，用 %S101 和系统字 %SW101 和 %SW102。（见*系统位 （%S）*，*p.658*）（又见*系统字 （%SW）*，*p.667*）。

下表列出了可以使用的设备：

远程	端口	说明
TWDLC • A10/16/24DRF, TWDLCA•40DRF, TWDLMDA20/40DTK, TWDLMDA20DRT	1	基本控制器用 miniDIN 连接器装置一个 3- 线 EIA RS-485 端口。
TWDNOZ232D	2	通信模块用 miniDIN 连接器装置一个 3- 线 EIA RS-232 端口。 **注意：**该模块只对模块型控制器有用。附加该模块后，控制器不能有操作显示扩展模块。
TWDNOZ485D	2	通信模块用 miniDIN 连接器装置一个 3- 线 EIA RS-485 端口。 **注意：**该模块只对模块型控制器有用。附加该模块后，控制器不能有操作显示扩展模块。
TWDNOZ485T	2	通信模块用一个端子块装置一个 3- 线 EIA RS-485 端口。 **注意：**该模块只对模块型控制器有用。附加该模块后，控制器不能有操作显示扩展模块。
TWDNAC232D	2	通信适配器用 miniDIN 连接器装置一个 3- 线 EIA RS-232 端口。 **注意：**该适配器只对一体型 16、24 和 40 I/O 控制器及操作显示扩展模块有用。
TWDNAC485D	2	通信适配器用 miniDIN 连接器装置一个 3- 线 EIA RS-485 端口。 **注意：**该适配器只对一体型 16、24 和 40 I/O 控制器及操作显示扩展模块有用。
TWDNAC485T	2	通信适配器用一个端子块装置一个 3- 线 EIA RS-485 端口。 **注意：**该适配器只对一体型 16、24 和 40 I/O 控制器及操作显示扩展模块有用。

远程	端口	说明
TWDXCPODM	2	操作显示扩展模块用 miniDIN 连接器装置一个 3- 线 EIA RS-232 端口，用 miniDIN 连接器装置一个 3- 线 EIA RS-485 端口或用一个终端装置一个 3- 线 EIA RS-485 端口。 **注意：**该模块只对模块型控制器有用。附加该模块后，控制器不能有通信扩展模块。

注意：端口 2 的存在和配置 （RS232 或 RS485）在上电时检查，由固件执行程序复位。

定义电缆

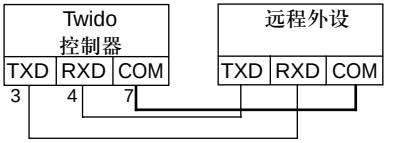
下面是 EIA RS-232 和 EIA RS-485 型的电缆连接定义图。

注意：如果使用 Twido 控制器端口 1，第 5 针 DPT 信号必须与第 7 针 （COM）短接。这意味着 Twido 控制器端口 1 使用 Modbus 通信并且没有协议用于与 TwidoSoft 软件通信。

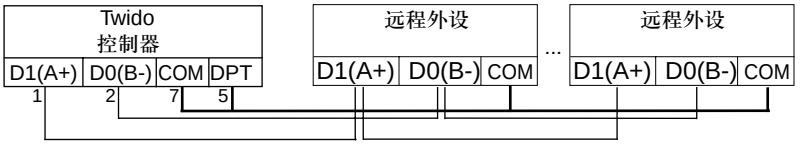
每个设备的电缆连接图如下。

Mini-DIN 接头

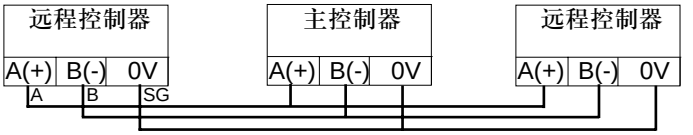
RS-232 EIA 电缆



RS-485 EIA 电缆



端子排连接



EIA RS-485 线极化在 TWDLCA • 40DRF 控制器

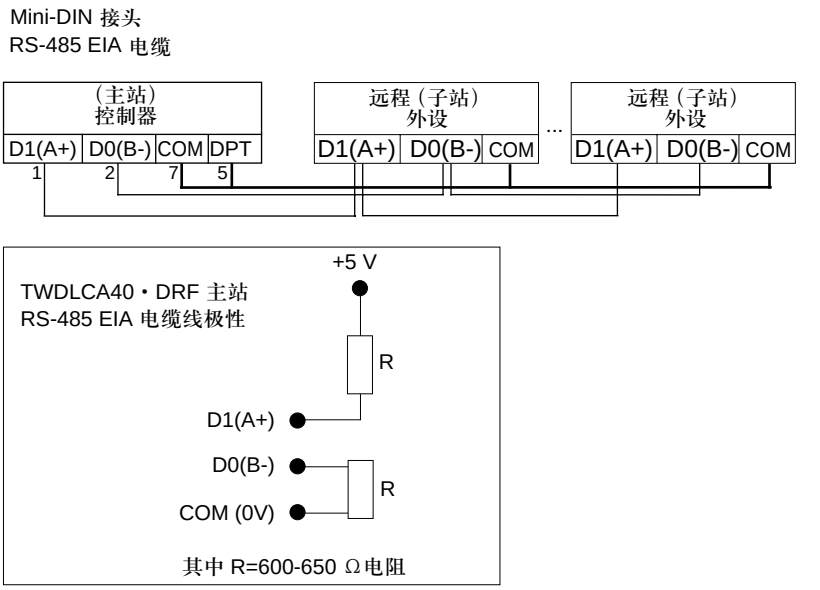
在 TWDLCA • 40DRF 控制器里没有内部预极化。所以，当连接 TWDLCA?40DRF Modbus 主控制器到 EIA-485 Modbus 网络时，需要外部线极化。（当在 EIA-485 对称线没有数据活动时，线没有被驱动，所以，容易受外部噪声和干扰的影响。为了保证接收者在正常状态下稳定，当没有数据信号存在时，Modbus 主设备需要通过外部线极化来偏移网络。）

注意：EIA RS-485 外部线极化仅在 Modbus 主控制器上必须得以实施；你没有必要在任何从设备上使用。

集成在 TWDLCA • 40DRF mini-DIN RS-485 EIA 的外部线极化组成：

- 一个上拉电阻到 5V 电压在 D1 （A+）回路，
- 一个下拉电阻到公共回路在 D0 （B-）。

下图描述了集成在 TWDLCA?40DRF mini-DIN RS-485 EIA 的外部线极化：



外部极化可以通过以下两种方法中的任意一种实施：

- 通过 mini-DIN 飞线外部连接用户提供的极化组件。（请参考连接器管脚定义。）
- 使用一个极化接头 （配置为 2- 线极化）和极化组件 （可以从目录快速查到）。

软件配置

为了配置控制器使用 Modbus 协议通过串行接口发送和接收字符，您必须：

步骤	描述
1	用 TwidoSoft 配置Modbus 串行口。
2	在应用程序中创建发送 / 接收表以供 EXCHx 指令使用。

配置端口

Twido 控制器可用基本端口 1 或可选端口 2 来使用 Modbus 协议。配置 Modbus 串行口：

步骤	动作
1	定义本地控制器配置的附加通信适配器或模块。
2	右键点击端口，再点击控制器通信配置并将协议类型改为 Modbus。
3	设置相关通信参数。

Modbus 主模式

Modbus 主模式允许控制器向从机发送一个 Modbus 查询，并等待其响应。Modbus 主模式只能通过 EXCHx 指令得到支持。 Modbus ASCII 和 RTU 均被 Modbus 主模式支持。
发送和 / 或接收帧的最大值是 250 字节。
另外与 EXCHx 指令相关的字表由控制，发送和接收表组成。

	高字节	低字节
控制表	命令	长度 （发送 / 接收）
	接收偏移	发送偏移
发送表	发送字节 1	发送字节 2
	...	...
	...	发送字节 n
	发送字节 n+1	
接收表	接收字节 1	接收字节 2
	...	...
	...	接收字节 p
	接收字节 p+1	

注意：除了查询个别从设备， Modbus 主控制器可对所有从设备发起广播查询。此命令子节在广播查询时必须设为 00，从设备地址必须置为 0。

控制表

此长度字节包含发送表的长度 （最大 250 字节），如果接收被请求，它将被接收结束时收到的字符数覆盖。
该参数是发送表的字节长度。如果 Tx 偏移参数等于 0，该参数将等于发送帧的长度。如果 Tx **偏移**参数不等于 0，发送表的一个字节 （由偏移值决定）将不被发送且该参数等于帧长度加 1。
此命令字节在 Modbus RTU 查询 （除了广播）情形下必须总是等于 1 （Tx 和 Rx）。
此 **Tx 偏移**字节包含字节发送时被忽略的字节在发送表中的排列号 （1 表示第一个字节， 2 表示第二个字节，等等）。它用于处理 Modbus 协议中与字节 / 字的值有关的问题。例如，如果此字节包含 3，则第三个字节将被忽略，使得表中第四个字节在发送时变为第三个字节。
此 **Rx 偏移**字节包含信息包发送时加入的字节在接收表中的排列号 （1 表示第一个字节， 2 表示第二个字节，等等）。它用于处理 Modbus 协议中与字节 / 字的值有关的问题。例如，如果此字节包含 3，则表中第三个字节将被填为零，使得实际接收到的第三个字节在表中变为第四个字节。

发送 / 接收表

在任一模式 （Modbus ASCII 或 Modbus RTU），发送表在 EXCHx 指令执行之前被填写。在执行时间，控制器决定数据链路层是什么，并完成所有必要的转换以处理传输和响应。发送 / 接收表不存储开始，结束和检查字符。
一旦所有字节发送完毕，控制器转换到接收模式并等待接收任何字节。
接收通过下面某种方式完成：

- 字符或帧超时被检测到，
- 以 ASCII 模式接收到的帧字符结束，
- 接收表满。

发送字节 X 条目包含 Modbus 协议 （RTU 编码）数据，这些数据将被发送。如果通信端口配置成 Modbus ASCII，发送时将附加合适的帧字符。第一个字节包含设备地址 （特殊或广播），第二个字节包含功能代码，剩下的字节包含功能代码相关的信息。

注意：这是一个典型应用，但没有定义所有可能性。数据发送时将不进行确认工作。

接收字节 X 条目包含 Modbus 协议 （RTU 编码）数据，这些数据将被接收。如果通信端口配置成 Modbus ASCII，响应时将移除对应的帧字符。第一个字节包含设备地址，第二个字节包含功能代码 （或响应代码），剩下的字节包含功能代码相关的信息。

注意：这是一个典型应用，但没有定义所有可能性。数据接收时除了校验，将不进行别的确认工作。

Modbus 从模式

Modbus 从模式允许控制器回应标准 Modbus 的查询。
当电缆 TSXPCX1031 与控制器相连时，端口开始 TwidoSoft 通信，电缆连接之前所运行的通信模式将被临时停止。
Modbus 协议支持两种数据链路层格式：ASCII 和 RTU。每种格式都由物理层定义， ASCII 使用 7 个数据位， RTU 使用 8 个数据位。
当使用 Modbus ASCII 模式时，消息的每个字节作为两个 ASCII 发送。 Modbus ASCII 帧从一个起始字符 （‘：’）开始，可用两个终止字符 （CR 和 LF）表示结束。帧结束字符默认为 0x0A （换行），用户可在配置中修改这个字节。 Modbus ASCII 帧的校验值是除去起始和终止字符后帧的二进制补码。
Modbus RTU 模式在消息发送之前不重新定义格式；然而，它使用一个不同的校验计算模式 CRC。
Modbus 数据链路层有下列限制：

- 地址 1-247
- 位：请求可有 128 位
- 字：请求可有 125 个 16 位的字

消息交换

语言提供了两种通信服务：

- **EXCHx 指令**：发送 / 接收消息
- **%MSGx 功能模块**：控制消息交换。

Twido 控制器处理 EXCHx 指令时使用端口配置协议。

注意：每个通信端口可配置不同或相同协议。通过选择端口号 （1 或 2） EXCHx 指令和 %MSGx 功能模块可以访问每个通信端口。

EXCHx 指令

EXCHx 指令允许 Twido 控制器发送和 / 或接收信息到 / 自 Modbus 设备。用户定义一个字表 （%MWi:L），其中包含用来发送和 / 或接收的控制信息和数据 （发送和 / 或接收最多 250 字节）。字表格式如前描述。

使用 EXCHx 指令完成消息交换：

句法：[EXCHx %MWi:L]

其中：x = 端口号 (1 或 2)

L = 在控制字，传输和接收表中的字数

Twido 控制器必须在第二条 EXCHx 指令执行之前由第一条指令完成交换。发送不止一条消息时，必须使用 %MSGx 功能模块。

当任何传输处于中断控制情形时 （数据接收也处于中断控制情形），EXCHx 列表指令的处理立即开始，并被视为后台处理。

%MSGx 功能模块

%MSGx 功能模块的使用不是必需的；它能用于管理数据交换。 %MSGx 功能模块有三种用途：

- **通信错误校验**
  错误校验确认 EXCHx 指令编程参数 L （字表长度）足够大，能包含发送消息的长度。与存储在字表第一个字的低字节中的长度相比较。
- **多消息协调**
  %MSGx 功能模块提供前面消息传输完成的信息，以保证多消息发送的协调。
- **传输优先消息**
  %MSGx 功能模块允许当前消息传输停止以发送紧急消息。
  %MSGx 功能模块有一个输入，两个输出：

输入 / 输出	定义	描述
R	输入复位	置为 1：通信重新初始化或模块重置（%MSGx.E = 0 和 %MSGx.D =1）。
%MSGx.D	通信完成	0：程序请求。 1：通信完成，如果：传输完毕，字符接收完毕，出错，或模块重置。
%MSGx.E	错误	0：消息长度正确且连接正确。 1：命令错误，表配置错误，接收字符错误（速率，奇偶校验，等等），或接收表满。

限制

请务必注意下列限制：

- 端口 2 的存在和配置 （RS232 或 RS485）在上电或复位时检查。
- 当连接 TwidoSoft 时端口 1 将不能进行任何消息处理
- EXCHx 和 %MSG 不能被配置为远程连接的端口所处理
- EXCHx 将停止 Modbus 从机处理
- EXCHx 指令的处理在错误事件中不会得到重试
- 输入复位 （R）可用来中断 EXCHx 指令接收处理
- EXCHx 指令可配置超时时间来中断接收
- 多消息通过 %MSGx.D 得到控制

错误和工作模式环境

当使用 EXCHx 指令时如果出错，位 %MSGx.D 和 %MSGx.E 将置为 1 且系统字 %SW63 包含端口 1 的错误代码， %SW64 包含端口 2 的错误代码。

系统字	用法
%SW63	EXCH1 错误代码： 0 - 操作成功 1 - 传送的字节数太大 （> 250） 2 - 传送表太小 3 - 字表太小 4 - 接收表溢出 5 - 超时 6 - 传送 7 - 字表命令错误 8 - 选择的端口没有配置或无效 9 - 接收错误 10 - 接收时不能使用 %KW 11 - 传送偏移量大于传送表 12 - 接收偏移大于接收表 13 - 控制器停止 EXCH 处理
%SW64	EXCH2 错误代码：见 %SW63。

主控制器重启

如果主 / 从控制器重启，下面事件中的一个将会发生：

- 冷启动 （%S0 = 1）强制通信重新初始化。
- 热启动 （%S1 = 1）强制通信重新初始化。
- 在停止模式，控制器停止所有 Modbus 通信。

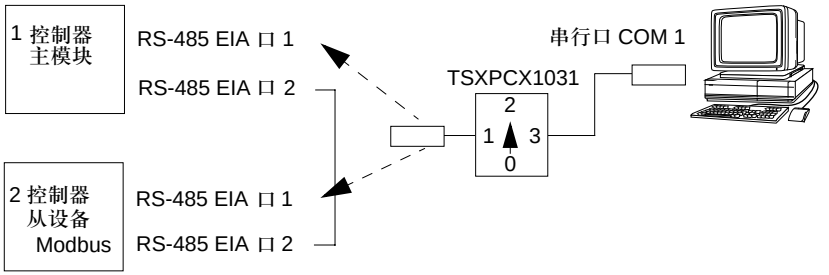
Modbus 连接举例 1

为配置一个 Modbus 连接，您必须：

1. 配置硬件。
2. 连接 Modbus 通信电缆。
3. 配置端口。
4. 写应用程序。
5. 初始化动态数据表编辑器。

下面图描述了怎样使用 Modbus 请求代码 3 读取一个从机的输出字，此例使用两个 Twido 控制器。

步骤 1：配置硬件：



硬件配置是两个 Twido 控制器。一个配置成 Modbus 主机，另一个配置成 Modbus 从机。

注意：此例中，每个控制器配置成使用 EIA RS-485 端口 1 和可选 EIA RS-485 端口 2。模块型控制器的端口 2 可以是 TWDNOZ485D 或 TWDNOZ485T，如果您使用 TWDXCPODM，它可以是 TWDNAC485D 或 TWDNAC485T。一体型控制器的端口 2 可以是 TWDNAC485D 或 TWDNAC485T。

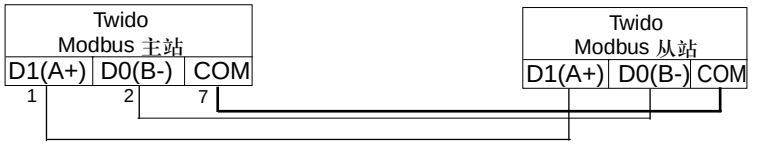
为了配置每个控制器，需连接电缆 TSXPCX1031 至控制器的端口 1。

注意：电缆 TSXPCX1031 一次只能连接到一个控制器，并且只能连接到 EIA RS-485 端口 1。

然后连接电缆至 PC 的 COM 1 口。确信电缆开关处于位置 2。下载和监控应用程序。对第二个控制器重复此过程。

步骤 2：连接 Modbus 通信电缆：

Mini-DIN 接头



端子排连接

此例中连线是点对点连接。三个信号 D1 （A+）， D0 （B-），和 COM （0V）按照图接线。

如果使用 Twido 控制器的端口 1， DPT 信号 （第 5 脚）必须与公共回路短接 （第 7 脚）。 DPT 的这个条件决定 TwidoSoft 是否被连接。当与地相接时，控制器将使用程序中的端口配置设置决定通信方式。

步骤 3：端口配置：

硬件 → 添加选项 TWDNOZ485-	硬件 → 添加选项 TWDNOZ485-
硬件 => 控制器通信设置 串行口 2 协议 Modbus 地址 1 波特率 19200 数据位 8 (RTU) 校验 无 停止位 1 响应超时 (× 100ms) 10 帧间隔时间 (ms) 10	硬件 => 控制器通信设置 串行口 2 协议 Modbus 地址 2 波特率 19200 数据位 8 (RTU) 校验 None 停止位 1 响应超时 (× 100ms) 100 帧间隔时间 (ms) 10

在主和从的应用里，可选的 EIA RS-485 端口被配置。确信控制器通信参数在 Modbus 协议和不同地址中得到改正。

此例中，主机地址设为 1，从机地址设为 2。位数设为 8，表示我们将使用 Modbus RTU 模式。如果它被设为 7，则表示使用 Modbus-ASCII 模式。其它的默认修改是将响应的停止时间增加到 1 秒。

注意：因为选择的是 Modbus RTU 模式，所以 “帧结束”参数被忽略。

步骤 4：写应用程序：

```
LD 1
[%MW0 := 16#0106 ]
[%MW1 := 16#0300 ]
[%MW2 := 16#0203 ]
[%MW3 := 16#0000 ]
[%MW4 := 16#0004 ]
LD 1
AND %MSG2.D
[EXCH2 %MW0:11]
LD %MSG2.E
ST %Q0.0
END
```

```
LD 1
[%MW0 := 16#6566 ]
[%MW1 := 16#6768 ]
[%MW2 := 16#6970 ]
[%MW3 := 16#7172 ]
END
```

使用 TwidoSoft 为主机和从机写应用程序。对于从机，我们简单写一些存储字到一个已知值集合。主机中， EXCHx 指令的字表初始化为从 Modbus 地址为 2 的从机读取 4 个字，从位置 %MW0 开始。

注意：注意 Modbus 主机的 %MW1 中 RX 偏移设置的使用。偏移 3 将在表的接收区域的第三个位置加入一个字节 （值 = 0）。主机中字的这种排列使得它们正确设置字的边界。如果没有这个偏移，每个数据字将被交换模块的两个字分开。偏移使得用法方便。

在执行 EXCH2 之前，程序检查和 %MSG2 相关的通信位。最后， %MSG2 的错误状态被得到并存储于本地基本控制器 I/O 的第一个输出位。另外还可以使 %SW64 进行错误检查，以便结果更为准确。

步骤 5：初始化主机的动态数据表编辑器：

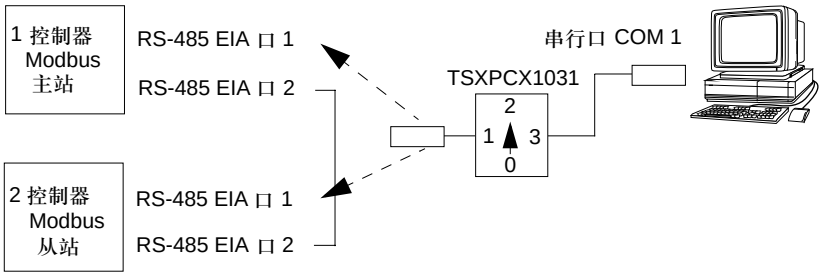
地址	当前	保留	格式
1 %MW5	0203	0000	十六进制
2 %MW6	0008	0000	十六进制
3 %MW7	6566	0000	十六进制
4 %MW8	6768	0000	十六进制
5 %MW9	6970	0000	十六进制
6 %MW10	7172	0000	十六进制

在下载并设置每个控制器运行之后，打开主机的动态数据表编辑器。检查表的响应部分，确认响应代码是 3 且被读字节数正确。此例中，注意从从机读取的字 （开始于 %MW7）在主机中排列正确，包括字的边界。

Modbus 连接举例 2

下面图描述了怎样使用 Modbus 请求代码 16 给一个从机写输出字。此例使用两个 Twido 控制器。

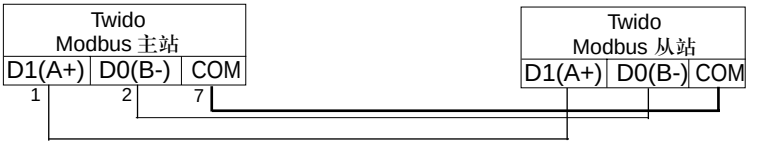
步骤 1：配置硬件：



硬件配置与上例相同。

步骤 2：连接 Modbus 通信电缆 （RS-485）：

MiNi-DIN 接头



Terminal block connection

Modbus 通信电缆与上例相同。

步骤 3：端口配置：

硬件→添加选项 TWDNOZ485-	硬件→ 添加选项 TWDNOZ485-
硬件 => 控制器通信设置 串行口 2 协议 Modbus 地址 1 波特率 19200 数据位 8 (RTU) 校验 无 停止位 1 响应超时 (× 100ms) 10 帧间隔时间 (ms) 10	硬件 => 控制器通信设置 串行口 2 协议 Modbus 地址 2 波特率 19200 数据位 8 (RTU) 校验 无 停止位 1 响应超时 (× 100ms) 100 帧间隔时间 (ms) 10

端口配置与上例相同。

步骤 4：写应用程序：

```
LD 1
[%MW0 := 16#010C ]
[%MW1 := 16#0007 ]
[%MW2 := 16#0210 ]
[%MW3 := 16#0010 ]
[%MW4 := 16#0002 ]
[%MW5 := 16#0004 ]
[%MW6 := 16#6566 ]
[%MW7 := 16#6768 ]
LD 1
AND %MSG2.D
[EXCH2 %MW0:11]
LD %MSG2.E
ST %Q0.0
END
```

```
LD 1
[%MW18 := 16#FFFF ]
END
```

使用 TwidoSoft 为主机和从机写应用程序。对于从机，写一个简单存储字 %MW18。它将给从机分配空间，存储地址 %MW0 到 %MW18。如果没有分配这个空间， Modbus 请求将试图写到从机中不存在的位置。
主机中， EXCH2 指令的字表初始化为写 4 个字节到 Modbus 地址为 2 的从机，地址是 %MW16 （十六进制 10）。

注意：注意 Modbus 主机程序的 %MW1 中发送 TX 偏移设置的使用。偏移 7 将使第六个字的高字节 （%MW5 中十六进制值 00）禁止。这样使得字表的发送表中数据值排列在一起，使得它们在数据分界处正确分开。

在执行 EXCH2 之前，程序检查和 %MSG2 相关的通信位。最后， %MSG2 的错误状态被得到并存储于本地控制器本体 I/O 的第一个输出位。另外还可以使用 %SW64 进行错误检查，以便结果更为准确。

步骤 5：初始化动态数据表编辑器：
创建主机的活动表如下：

地址	当前	保留	格式
1 %MW0	010C	0000	十六进制
2 %MW1	0007	0000	十六进制
3 %MW2	0210	0000	十六进制
4 %MW3	0010	0000	十六进制
5 %MW4	0002	0000	十六进制
6 %MW5	0004	0000	十六进制
7 %MW6	6566	0000	十六进制
8 %MW7	6768	0000	十六进制
9 %MW8	0210	0000	十六进制
10 %MW9	0010	0000	十六进制
11 %MW10	0004	0000	十六进制

创建从机的活动表如下：

地址	当前	保留	格式
1 %MW16	6566	0000	十六进制
2 %MW17	6768	0000	十六进制

在下载并设置每个控制器运行之后，打开从控制器的动态数据表编辑器。 %MW16 和 %MW17 的两个值被写入从机。主机中，动态数据表编辑器用于检查交换数据的接收表部分。该数据显示上述示例中的从机地址，响应代码，写入的第一个字，从 %MW8 开始写入的接收表。

标准 Modbus 请求

介绍

您能使用请求来交换设备间的数据以访问位和字信息。RTU 和 ASCII 模式使用相同的表格式。

格式	参考
Bit	%Mi
Word	%MWi

Modbus 主模式：读 N 位

下表是请求 01 和 02 描述。

		表索引	高字节	低字节
控制表		0	（发送 / 接收）	（发送长度）（*）
		1	03 （接收偏移）	00 （发送偏移）
发送表		2	从站 @ （1..247）	01 或 02 （请求码）
		3	读取的第一位的地址	
		4	N_1= 读取位的个数	
接收表（响应之后）		5	从站 @ （1..247）	01 或 02 （响应码）
		6	00 （由 Rx 偏移加入的字节）	N_2 = 写的数据字节数 = [1+ （N_1-1） /8]，其中 [] 表示整数部分
		7	读取的第一个字节	读取的第二个字节（如果 N_1>1）
		8	读取的第三个字节（如果 N_1>1）	
		...		
	（N_2/2） +6 （如果 N_2 是偶数），（N_2/2+1） +6 （如果 N_2 是奇数）		读取的第 N_2 个字节（如果 N_1>1）	

(*) 本字节也接收响应后传送的串长度

Modbus 主模式：读 N 字

下表是请求 03 和 04 描述。

	表索引	高字节	低字节
控制表	0	01 （发送 / 接收）	06 （发送长度） (*)
	1	03 （接收偏移）	00 （发送偏移）
发送表	2	从站 @ （1..247）	03 或 04 （请求码）
	3	读取的第一个字的地址	
	4	N = 读取的字数	
接收表（响应之后）	5	从站 @ （1..247）	03 或 04 （响应码）
	6	00 （由 Rx 偏移加入的字节）	2*N （读取的字节数）
	7	读取的第一个字	
	8	读取的第二个字 （if N>1）	
	...		
	N+6	读取的第 N 个字 （if N>2）	

(*) 本字节也接收响应后传送的串长度

注意：接收偏移 Rx 将在接收表的第三个位置加入一个字节 （值 = 0）。这样保证了读取字节数和读取字的值能存放在接收表的正确位置。

Modbus 主模式：写 1 位

下表是请求 05 描述。

	表索引	高字节	低字节
控制表	0	01 （发送 / 接收）	06 （发送长度）（*）
	1	00 （接收偏移）	00 （发送偏移）
发送表	2	从站 @ （1..247）	05 （请求码）
	3	要写的位地址	
	4	要写的位的值	
接收表（响应之后）	5	从站 @ （1..247）	05 （响应码）
	6	被写的位地址	
	7	被写的值	

(*) 本字节也接收响应后传送的串长度

注意：
- 此请求不需要用到偏移。
- 这里响应帧和请求帧一样 （正常情况下）。
- 为了写一位 1，发送表中相关字，即要写的位的值必须包含值 FF00H，值为 0 的位写 0。

Modbus 主模式：写 1 字

下表是请求 06 描述。

	表索引	高字节	低字节
控制表	0	01 （发送 / 接收）	06 （发送长度）（*）
	1	00 （接收偏移）	00 （发送偏移）
发送表	2	从站 @ （1..247）	06 （请求码）
	3	要写字的地址	
	4	要写的字值	
接收表（响应之后）	5	从站 @ （1..247）	06 （响应码）
	6	被写字的地址	
	7	被写的字值	

(*) 本字节也接收响应后传送的串长度

注意：
- 此请求不需要用到偏移。
- 这里响应帧和请求帧一样 （正常情况下）。

Modbus 主模式：写 N 位

下表是请求 15 描述。

	表索引	高字节	低字节
控制表	0	01 （发送 / 接收）	8+ 字节数 （发送）
	1	00 （接收偏移）	07 （发送偏移）
发送表	2	从站 @ （1..247）	15 （请求码）
	3	读取的第一位的编号	
	4	N_1 = 读取的位数	
	5	00 （不发送，偏移结果）	N_2 = 写的数据字节数 = [1+ （N_1 -1） /8], 其中 [] 表示整数部分
	6	第一个字节的值	第二个字节的值
	7	第三个字节的值	第四个字节的值
	...		
	（N_2 /2） +5 （如果 N_2 是偶数），（N_2 / 2+1） +5 （如果 N_2 是奇数）第 N_2 个字节的值	第 N_2 个字节的值	
接收表（响应之后）		从站 @ （1..247）	15 （响应码）
		被写的第一位的地址	
		被写位的个数 (= N_1)	

注意：

- 发送偏移 Tx =7 将取消发送帧中的第 7 个字节。这样也是为了发送表中的字值有一个好的对应

Modbus 主模式：写 N 字

下表是请求 16 描述。

	表索引	高字节	低字节
控制表	0	01 （发送 / 接收）	8 + (2*N) （发送长度）
	1	00 （接收偏移）	07 （发送偏移）
发送表	2	从站 @ （1..247）	16 （请求码）
	3	要写的第一字地址	
	4	N = 写的字数	
	5	00 （不发送，偏移结果）	2*N= 写的字节数
	6	写的第一个字值	
	7	写的第二个值	
	...		
	N+5	写的第 N 个值	
接收表（响应之后）	N+6	从站 @(1..247)	16 （响应码）
	N+7	被写的第一字的地址	
	N+8	被写的字数 （= N）	

注意：发送偏移 Tx =7 将取消发送帧中的第 7 个 MMSB 字节。这样也是为了发送表中的字值有一个好的对应。

“透明就绪”执行级 （Twido 串行 A05，以太网 A15）

概述

以下的 Modbus 功能代码都被串行 Modbus 和 TCP/IP Modbus 支持。关于 Modbus 协议，请参阅文档 Modbus Application Protocol （Modbus 应用协议）在 http://www.modbus-ida.org

Twido 支持的 Modbus 功能代码（MB FC）

下表描述了 Twido 串行和 TCP/IP Modbus 都支持的功能代码：

支持 MB FC	支持 Sub-fc 代码	功能
1	—	读多个内部位 %M
2	—	读多个内部位 %M
3	—	读多个内部寄存器 %MW
4	—	读多个内部寄存器 %MW
5	—	强制单个内部位 %M
6	—	写单个内部寄存器 %MW
8	00 仅有	Echo 诊断
15	—	写多个内部位 %M
16	—	写多个内部寄存器 %MW
23	—	读 / 写多个内部寄存器 %MW
43	14	读设备标识

以太网 TCP/IP 通信概述

以太网特点

以下信息描述了 Twido 控制器本体 TWDLCAE40DRF 的以太网可用特征。TWDLCAE40DRF 控制器本体是一个可连接至以太网的设备，以便执行 Modbus 应用协议而不是 TCP/IP。Modbus TCP/IP 在客户端 / 服务器拓扑结构网络中提供对等通信。

帧格式

Twido TWDLCAE40DRF 一体型控制器只支持 Ethernet II 帧格式。与 IEEE802.3 的帧格式不相容。注意 Schneider Electric 的其他可编程控制器，如 Premium 和 Quantum 系列，都支持 Ethernet II 和 IEEE802.3 帧格式。因此，如果用户计划使用 Twido 系列的 Premium 或 Quantum PLC，用户应将其配置为使用 Ethernet II 帧格式或允许其最优化兼容。

TCP 连接

TWDLCAE40DRF 一体型控制器是一种支持同时连接 4 个信道设备，可在 100Base-TX 以太网上通信。它可以完成 100Base-TX 自协商功能，也可以在 10Base-T 网络上工作。而且它还允许标记 IP 的连接，如在 TwidoSoft 应用程序中配置的那样。它支持 100Base-TX 自适应，也支持 10Base-T。而且，允许标记 IP 连接。（见标记 IP 表，以参照更多关于标记 IP 的信息）由 Twido 控制器支持的服务器处理的最大数字是 1 （每个 TCP 连接）。

IP 地址

每个 TWDLCAE40DRF 控制器默认赋给唯一静态 IP 地址。为了增加网络的灵活性，除了使用默认的 IP 地址， TwidoSoft 应用程序允许用户为设备配置不同的静态 IP 地址，也可定义子网和网关的 IP 地址。此外，如果 TWDLCAE40DRF 控制器未能从 BootP 服务器上获得有效 IP 地址 （或如果当您指定一个静态 IP 地址时，它进行重复 IP 地址检测），那么控制器转到后退方式并使用默认 IP 地址。
每一个 TWDLCAE40DRF 控制器被分配唯一一个 MAC 物理地址 （IEEE 全球地址），将永久存储在一体型本体控制器中。设备默认 IP 地址来自控制器的 MAC 物理地址 （IEEE Global Address）。

注意：当使用默认 IP 地址时， BootP 客户服务关闭。

Modbus TCP 客户端 / 服务器

TWDLCAE40DRF 控制器可以是 Modbus TCP/IP 客户端和服务器，分别取决于它是查询还是响应其他远程设备。
TCP 信息服务通过 TCP 端口 502 执行。

- Modbus 服务器执行施耐德透明准备标准 TR A15。
- Modbus客户端通过EXCH3指令和%MSG3功能实现。用户可以对多条EXCH3指令进行编程，但一次只能有一条 EXCH3 指令被激活。 TCP 连接自适应 Modbus 客户端执行施耐德透明准备标准 TR A10。

PC-Controller 以太网通信快速 TCP/IP 设置指南

范围

快速 TCP/IP 设置向导是为了提供以太网连接信息和 TCP/IP 配置信息，以快速设置运行 TwidoSoft 应用的 PC 和 Twido 控制器之间通过以太网的通信。

检查当前用户 PC 的 IP 设置

以下的信息描述了如何检查用户 PC 的 IP 设置。此外，这个过程对所有的 Windows 操作系统版本都有效：

步骤	动作
1	选择运行从 **Windows** 开始访问它。
2	键入 “**command**”在打开的运行对话框中 **结果：**此 **C:\WINDOWS\system32\command.com** 出现。
3	键入 “**ipconfig**”在命令行中 .
4	Windows IP 配置出现，显示如下参数： IP 地址…… 子网掩码……： 默认网关……： **注意：**以上 IP 设置无法直接通过命令提示被直接修改。只适用于咨询。如果用户希望修改用户 PC 的 IP 配置，请参照如下步骤。

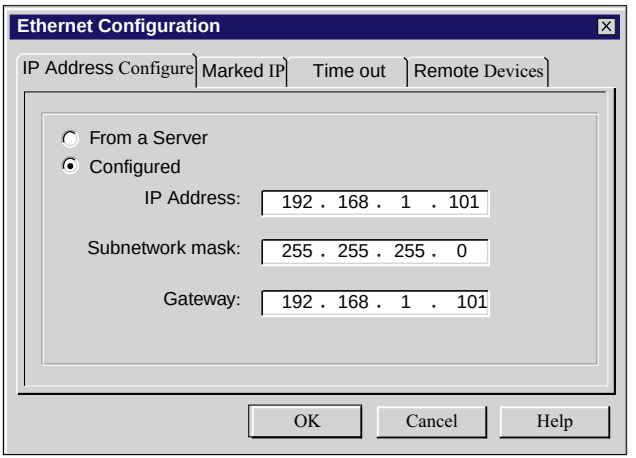
配置用户 PC 的 TCP/IP 设置

以下信息将帮助用户配置其 PC 的 TCP/IP 设置，用户的 PC 须运行 TwidoSoft 应用程序以对 Twido 控制器通过网络进行编程和控制。以下标记出的过程可在装有 Windows XP 操作系统的 PC 上运行，且仅做示例。（另外，对于其他操作系统，请参照用户手册中为其他安装在 PC 上的特定操作系统的 TCP/IP 设置指令的概述。）

步骤	动作
注意：如果用户的 PC 已经安装而且以太网卡已通过现存网络对其进行了配置，则无需更改 IP 地址设置（略过 1 – 6 步，继续以下步骤）。本过程中的 1 – 6 步用于为用户修改其 IP 设置。	
1	选择控制面板 > 网络连接从 Windows 开始访问它。
2	右键点击局域网连接（孤立网络）在用户希望连接 Twido 控制器的网络上，然后选择属性。
3	选择 **TCP/IP** 位于已安装的网络组件列表中，然后点击属性。 **注意：**如果 TCP/IP 协议未出现在已安装组件的列表中，请参考操作系统的用户手册，以查找如何安装 TCP/IP 网络组件。
4	此 **TCP/IP** 属性对话框出现，显示用户 PC 当前的 TCP/IP 设置，包括 **IP** 地址和子网掩码。 **注意：**在单一网络中，不需要使用自动获得 **IP** 地址选项，此指定 **IP** 地址框必须被选中， IP 地址和子网掩码域必须包含有效的 IP 设置。
5	输入一个有效静态 **IP** 地址。格式为点十进制。对于独立网络，建议指定 C 级 IP 地址。（见 *IP 寻址第 176 页*）。举例， 192.168.1.198 是一个 C 类 IP 地址。 **注意：**用户指定的 IP 地址必须与当前网络的 ID 兼容。例如，如果当前网络地址支持 192.168.1.xxx IP 地址 （当 192.168.1 为网络 ID，则 xxx = 0-255 是主机 ID，则用户可以指定 191.168.1.198 为其 PC 的有效 IP。确认主机 ID 198 在网络上是唯一的）。
6	输入一个有效子网掩码格式为点十进制。如果用户的 C 类网络无子网，则建议指定 C 类网络默认的子网掩码，如 255.255.255.0。

配置用户 Twido 控制器的 TCP/IP 设置

一旦用户配置了其运行 TwidoSoft 应用的 PC 的 TCP/IP 设置，若希望在网络上用 TwidoSoft 通信，则需要配置 Twido 控制器的 TCP/IP 设置，如下所示：

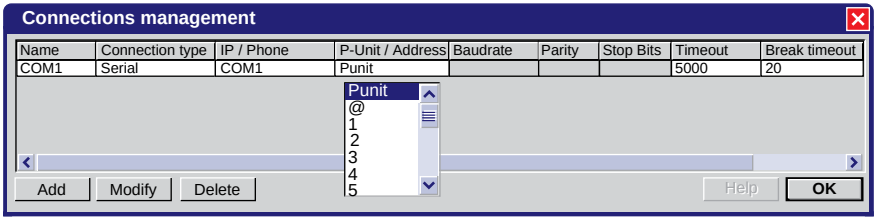
步骤	动作
1	连接电缆 （TSXPCX1031），从运行 TwidoSoft 的 PC 到 Twido 控制器的 RS-485 控制台端口。
2	在用户 PC 上运行 **TwidoSoft** 应用程序。
3	选择一个新的硬件从 TwidoSoft 应用浏览器中，并选择 **TWDLCAE40DRF** 控制器。
4	从 **TwidoSoft** 菜单栏中选择 **PLC** ＞选择一个连接，并选择 **COM 1** 端口。
5	双击以太网端口快捷键位于 TwidoSoft 应用浏览器中 （或者选择硬件 > 以太网位于菜单栏中）以调用以太网配置对话框，如下所示： 
6	由 **IP** 地址配置页中： ● 选择服务器的单选按钮，用 BootP 客户支持，以从服务器上自动获得动态 IP 地址 （直接到第 10 步） **注意：**TWDLCAE40DRF 控制器执行三次，每 200 ms 再次发送 BootP 请求给服务器。如果没收到有效响应，控制器退回使用默认 IP 地址。 ● 从 IP 地址配置表中，选择配置，开始配置 IP 地址，子网掩码和网关地址域，如以下 7 – 9 步。 **注意：**本阶段只涉及 PC 到控制器通过以太网通信的基本配置。因此，用户无需配置标记 IP，超时检查和远程设备表。

步骤	动作
7	输入一个有效的 **IP** 地址，以点十进制格式。本 IP 地址必须与上一步中配置的 PC 的 IP 地址兼容。 **注意：**Twido 控制器和 PC 的 IP 地址的网络 ID 必须相同。然而， Twido 控制器的主机 ID 必须与 PC 的主机 ID 不同，且在网络中唯一。例如，如果 PC 的 C 类 IP 地址为 192.168.1.198，则 Twido 控制器的有效地址为 192.168.1.xxx（当 192.168.1 为网络 ID，则 xxx = 0-197， 199-255 为主机 ID）。
8	输入一个有效子网掩码格式为点十进制。因此，用户输入的子网掩码必须与 PC 指定的相一致。 Twido 控制器和运行 TwidoSoft 的 PC 必须在同一网段上。 **注意：**如果用户的 C 类网络无子网，则建议指定 C 类网络默认的子网掩码，如 255.255.255.0。
9	输入一个有效网关地址，以点十进制格式。 **注意：**如果独立网络上没有网关设备，则输入 Twido 控制器自己的 IP 地址，此地址在第 6 步中被配置。
10	点击 **OK** 保存用户 Twido 控制器的以太网配置的设置。

在 TwidoSoft 中设置一个新的 TCP/IP 连接

用户将在 TwidoSoft 应用中新设置一个 TCP/IP 连接。这个新的特定的 TCP/IP 连接将允许运行 TwidoSoft 的 PC 与 Twido 控制器在以太网上通信。

在文件菜单选择首选项再点击连接管理对话框：



步骤	动作
1	单击增加按钮，位于连接管理对话框。 **结果：**添加了一个新的连接行。新行显示的为默认连接设置。用户需要修改这些设置。 **注意：**要在域中设置一个新的值，用户有两种选择： ● 选择要修改的区域，点击修改。 ● 双击所选域。
2	在名字域中，输入连接的描述性名称。有效的名称最多包含 32 个字母字符。
3	在连接方式区域，点击展开下拉菜单，包括：TCP/IP，串口， Modem （如有）和 **USB** （如有）。 选择 TCP/IP 在 PC 和可连接以太网的 Twido 控制器之间建立一个新的以太网连接。
4	在 IP / Phone 域中输入一个有效的 IP 地址作为你希望连接到的 Twido TWDLCAE40DRF 的 IP 地址。 **IP** 地址：输入上一步中用户为 Twido 控制器指定的 IP 地址。
5	此 Punit 地址域能填写，如 IP / Phone 区域已选择。对于 TCP/IP，默认值是直接对于串口，默认值是 Punit。当这些被选择时，接下来的三项 （波特率，奇偶校验和停止位）被禁止。 如不知道控制器的地址， @ 允许你稍后选择，当下载程序时再选择。*（在第一次连接前，有一个弹出菜单让你选择要传输的控制器，地址在 1-247 的范围，1 是默认值。）*
6	使用默认设置，位于超时和断开时间域，除非用户有特定超时需求。*（请参见以太网连接管理，* *p.194**）*

步骤	动作
7	单击 **OK** 按钮保存新的连接设置并关闭连接管理对话框。 结果：所有新增的连接名被增加到文件→参考项的下拉菜单中→选择对话框或在 **PLC** →选择一个连接菜单。

连接控制器到网络

概览

以下信息描述了如何在用户以太网上安装 TDWLCAE40DRF 一体型控制器网络。

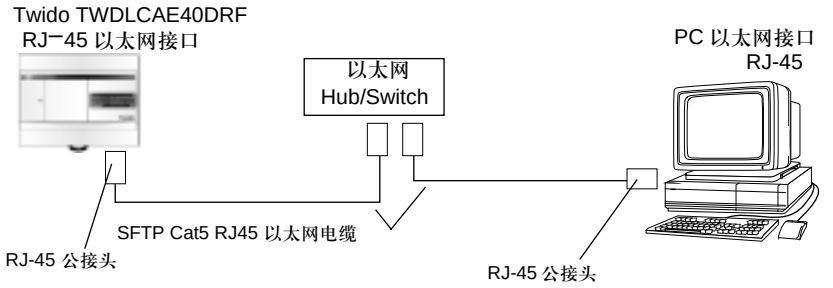
决定适当的 IP 地址设置

请咨询您的网络管理员以决定是否必须配置新的设备 IP，网关和子网掩码地址。如果管理员指定了新的 IP 地址参数，用户需要在 TwidoSoft 应用中手工输入此信息。今后请遵照此说明，在 *TCP/IP 设置， p.182*。

以太网连接

注意：尽管 Twido TWDLCAE40DRF 和运行 TwidoSoft 编程软件的 PC 之间的连接支持使用电缆 （以太网交叉电缆）直接连接，但是我们不推荐使用此方式。因此，用户的连接应该通过网络以太网集线器 / 交换机。

下图显示了一个通过以太网集线器 / 交换机进行连接的 Twido 网络：



Twido TWDLCAE40DRF 具有一个 RJ-45 连接器，可以连接到 100BASE-TX 以太网，并具有自适应功能。它可以兼容 100Mbps 和 10 Mbps 两种网速。

注意：当把 Twido 控制器连接到 100BASE-TX 网络中时，至少需要 5 类以太网电缆。

IP 寻址

概览

本部分提供给用户有关 IP 寻址符号，子网和网关的概念的有关信息。

IP 地址

IP 地址有 32 位，以点十进制表示。它包括四组从 0 到 255 之间的数字，每组之间用圆点隔开。例如， 192.168.2.168 就是一个以点十进制表示的 IP 地址 （注意这是一个保留的 IP 地址，仅用作示例）
一般网络中，IP 地址分为三类，分别为 A 类，B 类和 C 类。类与类之间可以通过第一个数字的范围来区分，如下表所示：

第一组十进制数	IP 类别
0-127	A 类
128-191	B 类
192-223	C 类

IP 子网掩码

一个 IP 地址包含两个部分，网络 ID 和主机 ID。子网掩码用来将 IP 地址的网络部分分离开，以使用更多的主机 ID 人工构建子网。因此，子网构建是将多个物理网络连接到逻辑网络的一个手段。同一个子网上的所有设备网络 ID 相同。

注意：如果用户所在的是一个大型组织的分部，则在用户公司内部网络上实现子网构建是一个很好的机会。在现存网络上安装新的 Twido 控制器时，请从您的网络管理员那里获取更多关于子网构建的信息。

网关地址

网关设备也叫做路由器，用在用户公司的全局网上，使用户所在网段能够访问其他网段，因特网，或远程局域网。
网关地址同样使用点十进制格式，同上述 IP 地址格式相同。

注意：在现存网络上安装新的 Twido 控制器时，请从您的网络管理员那里获取更多网关的信息。

分配 IP 地址

概览

本部分将提供给用户为安装到网络的 Twido TWDLCAE40DRF 控制器分配 IP 地址方法的信息。

安装到独立网络

Twido TWDLCAE40DRF 控制器应安装到一个独立的以太网网络。

注意：独立网络意味着此网络未与因特网或公司的内部网相连。

从 BootP 获得地址

BootP 服务地址：如果您在 **IP** 地址配置表中选择服务器， Twido 控制器将试图首先从 BootP 服务器中获得 IP 地址。
BootP 处理要从 BootP 服务器上获得响应。如果在 BootP 请求发出后仍未得到有效 IP 地址， Twido 将使用从 MAC 地址转来的默认 IP 地址配置。
（参照控制器的 MAC 地址和默认 IP 地址， p.178）

MAC 地址和默认 IP 地址

MAC 地址：每个 Twido TWDLCAE40DRF 控制器都有一个出厂时设置的 MAC 地址，是赋给每个以太网设备的全球唯一的 48 位地址。
默认 **IP** 地址：Twido 控制器的默认以太网接口 IP 地址来自其唯一 MAC 地址。
默认 **IP** 地址的格式为点十进制，如下所示：

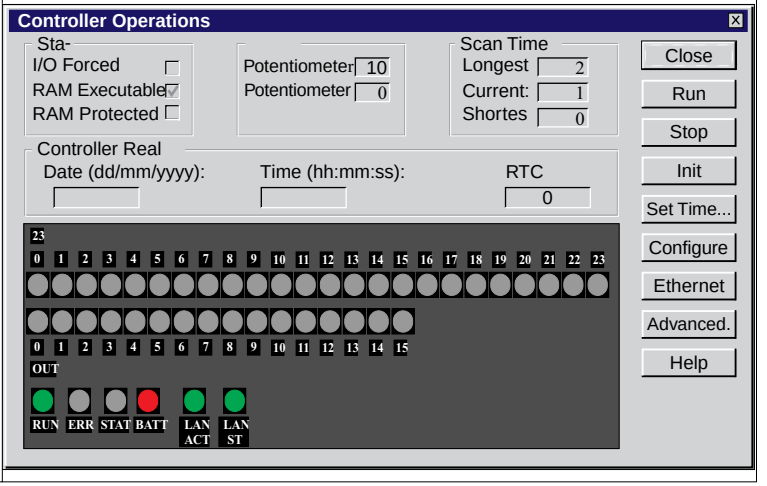
085.016.xxx.yyy, 其中：

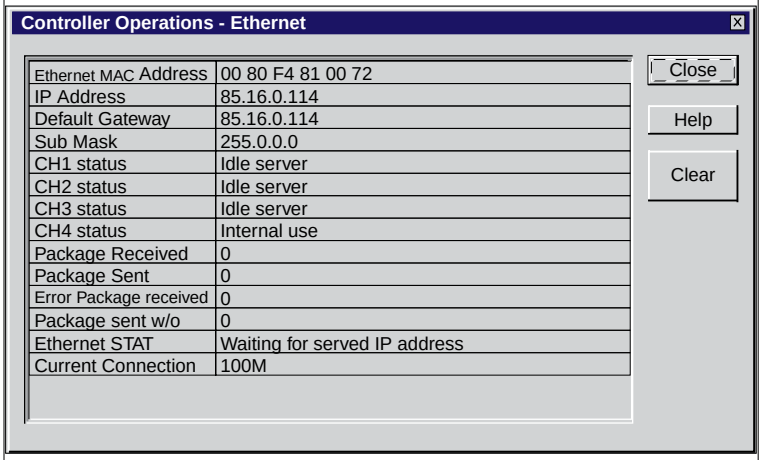
- 085.016. 是所有来自 MAC 地址的 IP 地址集合头，
- xxx 和 yyy 是设备的 MAC 地址的最后两个数字。

例如， IP 地址来自于 MAC 地址 00.80.F4.81.01.11 是 085.016.001.17.

检查控制器 MAC 地址和当前的 IP 地址

若要检查 Twido 控制器的 MAC 地址，当前的 IP 地址， IP 设置 （子网掩码和网关地址），以太网连接状态，请依照如下步骤：

步骤	动作
1	在 TwidoSoft 应用程序中，从菜单栏选择 **PLC**。
2	选择查看 **PLC** **结果：**此控制器操作对话框弹出，在软件前面板上显示 Twido LED ； 如下图所示： 

步骤	动作
3	单击在屏幕右边的以太网按钮访问连接参数。 **结果：**此控制器以太网操作表格弹出，显示 MAC，当前 IP，子网和网关信息，还有以太网连接信息，如下图所示： 
4	注意 Twido 控制器唯一的 **MAC** 地址显示在以太网表格的第一行。
5	IP 信息在表中的显示位置取决于用户 **IP** 配置表设置以太网配置对话框 （见 *IP 地址配置表，p.184*）： ● 如果在 IP 配置表上从一个服务器中选择，以上的表将显示 Twido 控制器的默认的 IP 地址 （来自 MAC 地址），默认子网和网关。 如果从服务器端无法得到有效的 BootP 服务 IP 地址，默认 IP 地址仅在后退方式下应用。当一个通道用作 BootP UDP，那么这个通道状态显示为内部使用。 ● 如果选择配置，则以上的表将显示以前输入到 IP 配置表中的当前的 IP 地址，子网和网关设置 **注意：**余下的区域显示有关以太网连接的当前状态信息。

私有 IP 地址

如果用户的网络是一个独立网络（与因特网不相连），则用户可以为其网络节点（Twido 控制器）指定任意的 IP 地址（只要此 IP 地址符合 IANA 格式规则并不与任何网络上已经存在的其他设备的 IP 相冲突）。
私有 IP 地址满足在独立网络上任意 IP 寻址的要求。注意私有地址范围内的地址只适用于组织内部。
下表显示了私有 IP 地址的范围：

网络	私有 IP 地址的有效范围
A 类	10.0.0.0 -> 10.255.255.255
B 类	172.16.0.0 -> 172.31.255.255
C 类	192.168.0.0 -> 192.168.255.255

为用户的 IP 地址赋值

现今的网络很少有完全隔离于因特网或公司的其他部分的以太网。因此，如果需要在已存网络上安装并连接 Twido 基本控制器，不要任意为其指定 IP 地址，而应事先咨询网络管理员。在为控制器的 IP 赋值时，应遵循如下步骤。

注意：在独立网络中最好使用 C 类 IP 地址。

TCP/IP 设置

概览

以下是如何为 Twido TWDLCAE40DRF 一体型控制器设定以太网 TCPI/IP 配置的详细信息。

注意：TCP/IP 设置只能在 TwidoSoft 应用程序处于离线模式下完成。

注意

意外的设备操作

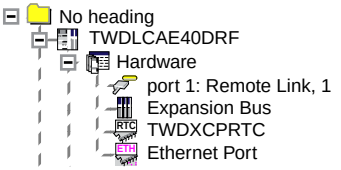
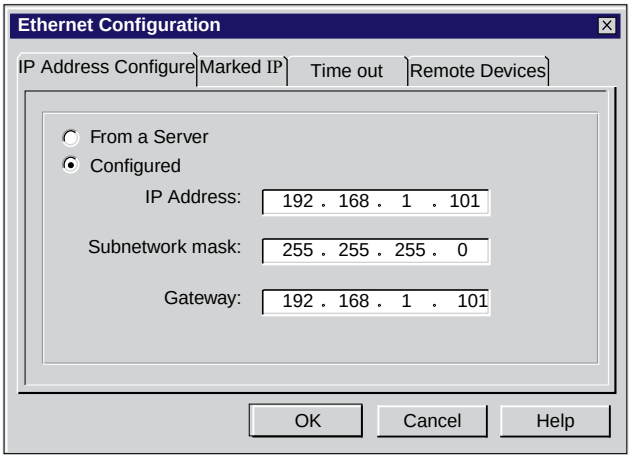
如有两个设备地址相同会对网络造成不可预知的操作。

- 确信设备有唯一的地址。
- 从系统管理员获得 IP 地址，以避免双重地址的可能性。

如果不遵守这个警告将会导致人身伤害或设备损害。

调用以太网配置对话框

以下步骤详细描述了如何调用以太网配置对话框：

步骤	动作
1	打开应用浏览器，如下图所示。 **结果：**  **注意：**确认可连接至以太网的设备如 TWDLCAE40DRF 被选择为当前硬件，否则以太网端口硬件选项将不会出现。
2	双击以太网端口快捷键图标，打开以太网配置对话框，如下图所示。 **结果：**  **注意：**有两种可选方法来调用以太网配置屏幕： **1.** 右击以太网端口快捷键图标并选择编辑。 **2.** 从 TwidoSoft 菜单栏选择硬件 > 以太网。

TCP/IP 设置

以下部分详细描述了如何配置 Twido TWDLCAE40DRF TCP/IP 参数，通过使用 **IP** 地址配置，标记 **IP**，超时和远程设备表。

IP 地址配置表

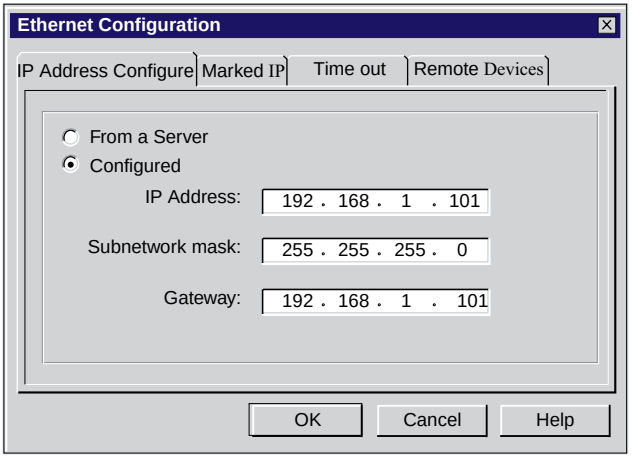
概览

以下信息描述了如何设置 IP 地址配置表和以太网配置对话框。

注意：Twido 控制器的 IP 地址仅在 TwidoSoft 应用程序离线时可配置。

IP 地址配置表

以下是一个 IP 地址配置表的样本屏幕，显示了用户手工配置的 IP，子网和网关地址的示例：



配置 IP 地址表　　以下信息是关于如何在 IP 地址配置表中配置不同的区域：

区域	配置
默认 IP 地址	如果用户不希望手动设置 Twido 控制器的 IP 地址，则选中此按钮 （IP 地址，子网掩码和网关文本框变灰）。 Twido 控制器 （BootP 客户）将使用从其服务器中自动分配来的 IP 地址。 在三次每 200 ms 的间隔重试之后如果不能获得一个有效的服务 IP 地址。 Twido 控制器将选择使用默认 IP 地址 （撤回状态）。（注意 Twido 控制器每 15 s 间隔周期性地发送要求给服务器，直到获得一个有效的 IP 地址。） 默认以太网界面 IP 地址来自于它的 MAC 地址。 （注意当任何 PLC 通道 （除了内部使用通道）是活动状 态时，默认 IP 地址不会自动改变。） **注意：**若要获取更多有关 MAC 地址的信息，请参照*分配 IP 地址 p. 178*。
配置	选中此按钮手动配置 IP，子网和网关地址。 **注意：**请咨询系统或网络管理员以获取有效的网络 IP 参数。
IP 地址	输入 Twido 控制器的静态 IP 地址，格式为点十进制。 **注意：**若要设备通信良好，运行 TwidoSoft 应用的 PC 的 IP 地址和 Twido 控制器的网络 ID 必须一致。 **注意：**若要网络的通信良好，每个连接在网络上的设备的 IP 地址必须唯一。当 Twido 控制器连接到网络上时， 将进行是否有重复 IP 的检验。如果网络中有重复的 IP 地址， Twido 控制器的 LAN ST LED 将发出 4 个周期性的闪烁。用户必须在此域中输入一个新的不重复的 IP 地址。

区域	配置
子网掩码	输入有效的子网掩码并通过网络管理员将其赋给控制器。请注意此域不能留空，必须输入值。 默认情况下， TwidoSoft 应用程序将自动根据用户在以上 IP 地址域中的 IP 类别计算并显示一个默认的子网掩码。根据 Twido 网络 IP 地址类别，默认的子网掩码值遵循如下规则： A 类网络 -> 默认子网掩码：255.0.0.0 B 类网络 -> 默认子网掩码：255.255.0.0 C 类网络 -> 默认子网掩码：255.255.255.0 **注意：**若要设备通信良好，运行 TwidoSoft 应用的 PC 和 Twido 控制器的子网掩码必须一致。 **注意：**若用户的 Twido 控制器无特殊子网连接要求，请使用默认的子网掩码。
网关	输入网关的 IP 地址。在 LAN 中，网关必须同用户的 Twido 控制器在同一网段。此信息可以从用户的网络管理员处获得。请注意应用程序并不提供默认值，所以用户必须在此域中输入一个有效的网关地址。 **注意：**如果用户的网络中无网关设备，只需在网关域中输入 Twido 控制器的 IP 地址。

标记 IP 表

概览

以下信息描述了如何配置以太网配置对话框中的标记 IP 表。

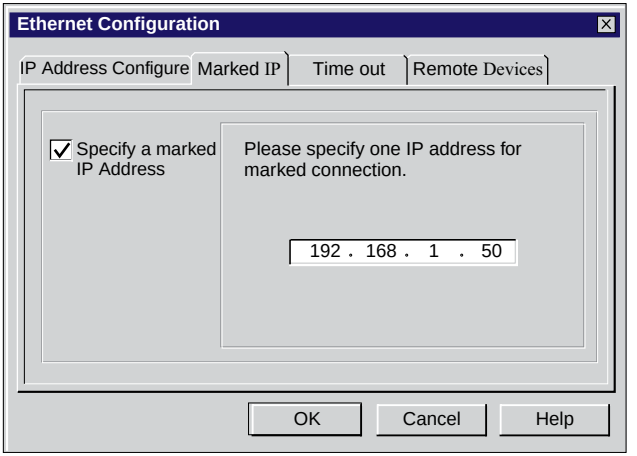
注意：标记 IP 仅可在 TwidoSoft 应用程序处于离线模式下被配置。

标记 IP 功能的定义

本功能允许用户保留 Twido 控制器支持的四个以太网 TCP 连接信道中的一个，以被指定标记 IP 的特定用户的主机使用。
标记 IP 可以保证 TCP 信道保留且可用，可与特定远程设备通信，即使空闲超时不可用 （空闲超时设为 “0”）。

标记 IP 表

下图给出了标记 IP 表的简单屏幕的图像，显示了用户输入标记 IP 地址的示例：



配置标记 IP 表

要配置标记 IP 表，请遵循如下步骤:

步骤	动作
1	检查被标记的表确定一个标记 IP 地址来使能标记 IP 功能。注意在默认状态下，标记 IP 不可用。 **结果：**IP 地址框在表格的右侧激活，如上面图中所示。
2	在所提供的 IP 地址表中输入用户希望标记 IP 的用户主机地址。 **注意：**本域中无默认值。用户必须输入所标记设备的 IP 地址，或者不要选中指定一个标记 IP 地址按钮，使本功能无效。

超时表

概览

以下信息描述如何配置超时表。

注意：只有当 TwidoSoft 应用程序处于离线模式时， Twido 控制器才配置超时表。

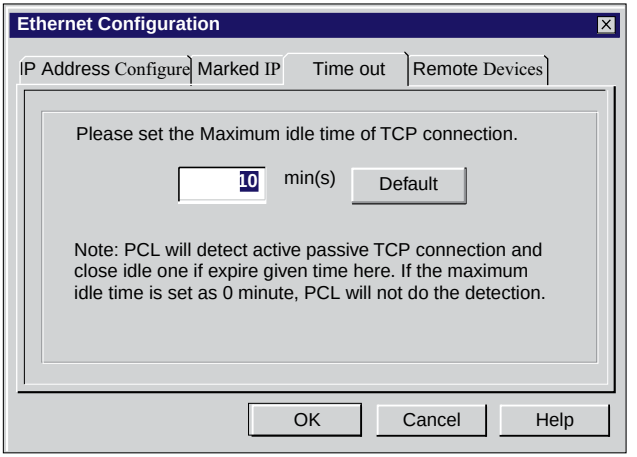
定义超时表

超时检查对所有 Twido 控制器的当前的以太网 TCP 连接给出一个空闲超时。空闲超时是四个以太网连接通道中的任何一个在与远程客户主机断开之前，此通道可以保持空闲的时间长度。

注意：空闲定时器在所监测的连接通道上有数据传输时被复位。

超时表

下图是一个空闲定时器默认值为 10 分钟时的空闲检查表的示例屏幕：



配置超时表

要设置空闲定时器，直接输入以分钟为单位的时间量在分钟文本框中，如上面图标所示。

注意：

1. 默认时间为 10 分钟。在用户输入一个值之后，要重置配置时间为 10 分钟，则点击默认。
2. 为不能使用超时功能，设置耗费时间为 0。 Twido 控制器不再进行空闲检查。因此， TCP 连接将保持。
3. 允许的空闲时间最长为 255 分钟。

远程设备表

概览

以下信息描述了在用户希望使用 EXCH3 指令以使 Twido 控制器用作 TCP/IP 客户端的情形下，如何配置以太网配置对话框中的远程设备表。

注意：Twido 控制器的远程设备表仅在 TwidoSoft 应用程序处于离线模式下可配置。

用户首先应该知道的

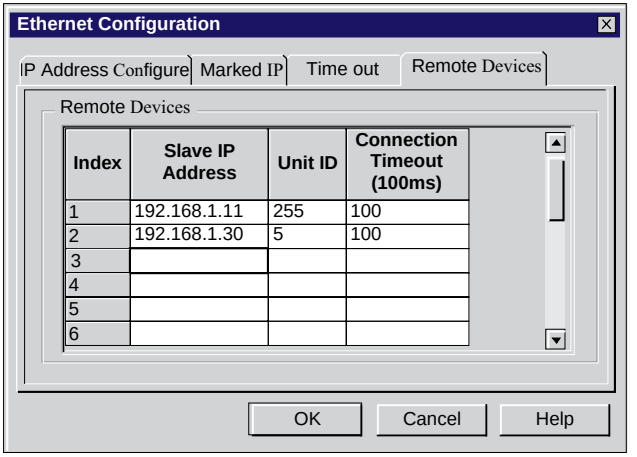
用户只需在希望使用 Modbus TCP/IP 客户端 （合法 Modbus 主机）指令（EXCH3）的控制器上配置远程设备。

远程设备表

远程设备表存储有关以太网上的远程控制器 （用作 Modbus TCP/IP 服务器）的信息，此控制器可被 Modbus TCP/IP 客户端使用 EXCH3 指令查询。因此，用户必须合理地配置远程设备表，以使 Modbus TCP/IP 客户端服务器能够在网络上对 Modbus TCP/IP 服务器控制器进行访问。

远程设备表

以下显示了在一个用作 Modbus TCP/IP 客户端的 Twido 控制器上配置的远程设备表的图例屏幕：



配置远程设备表　以下信息描述了如何配置远程设备表的不同域：

区域	配置
Index	这是一个只读域，包含与远程设备 （在从设备 IP 地址域中指定的 Modbus TPC/IP 服务器）以太网 IP 地址相关的 MBAP 索引。 MBAP 索引由 EXCH3 指令调用，用作确定表中哪一个远程控制器正在被 Modbus TCP/IP 客户端访问。 **注意：**表中可最多指定 16 个远程设备，索引从 1 到 16。
从设备 IP 地址	在此域中输入远程设备 （Modbus TCP/IP 服务器）控制器的 IP 地址。 **注意：**用户必须从索引 1 开始配置从设备 IP 地址，并以索引升序进行逐一配置。例如，在配置从设备 IP 索引为 1 然后就为 3 是不允许的，应在索引 3 之前首先输入索引 2。
单元 ID	在此域中输入 Modbus 单元 ID （或协议地址）。一个有效的单元 ID 的范围为 0 到 255。默认设置为 255。 单元 ID （除了 255）使得与远程设备通过 Modbus 网桥或网关进行通信成为可能。如果目标设备是另一个 Twido 控制器或是一个有效的 Modbus 设备，其通过网关连接到另一个总线串行口连接地址上，则用户可以相应地设置远程设备的单元 ID。 在此域中，用户可以将从设备 IP 地址设为网关或网桥的 IP 地址，单元 ID 设为目标设备的 Modbus 串行口连接地址。
连接超时 （100 ms）	确定时间单位为 100 ms， Twido 控制器将尝试与远程设备建立 TCP 连接。如果连接在超时后仍未能建立，则 Twido 控制器将放弃尝试，直至由指令发起下一次 EXCH3 的连接请求。 有效的超时设置范围为 0 到 65535 （转换为 0 到 6553.5 s）。 默认设置为 100。

显示以太网配置

概览

用户可以使用 TwidoSoft 配置编辑器查看当前 Twido 控制器的以太网配置。

查看以太网配置

要使用配置编辑器查看当前的以太网配置，请遵循如下步骤：

步骤	动作
1	从 TwidoSoft 菜单栏选择程序 > 配置编辑器。
2	点击位于配置编辑器任务栏的快捷图标 **ETH** 或双击以太网端口快捷键。
3	以太网 TCP/IP 配置参数将出现在表格中，如下图所示：  Ethernet Configuration IP Address Configuration IP address 192 . 168 . 1 . 101 Subnetwork mask 255 . 255 . 255 . 0 Gateway address 192 . 168 . 1 . 101 Marked IP 192 . 168 . 1 . 50 Remote Server Slave IP address / Unit ID / Connection Timeout 192 . 168 . 1 . 11 — 255 — 100 192 . 168 . 1 . 30 — 5 — 100 192 . 168 . 1 . 50 — 255 — 1500 192 . 168 . 1 . 16 — 255 — 1500 192 . 168 . 1 . 20 — 255 — 100
4	在此步中，如果用户刚刚对 Twido 的以太网 TCP/IP 配置设置做了修改，仍可以选择保存修改或放弃修改返回先前配置，如下述说明： ● 选择工具 > 接受更改以保存用户对 TCP/IP 以太网配置所做的修改。 ● 选择工具 > 取消更改放弃修改恢复先前的 TCP/IP 以太网配置设置。 ● 选择工具 > 编辑 ... 返回到以太网配置对话框修改 TCP/IP 配置设置。 ● 选择 **PLC** > 传送 **PC** => **PLC**... 下载 PLC 配置到 Twido。

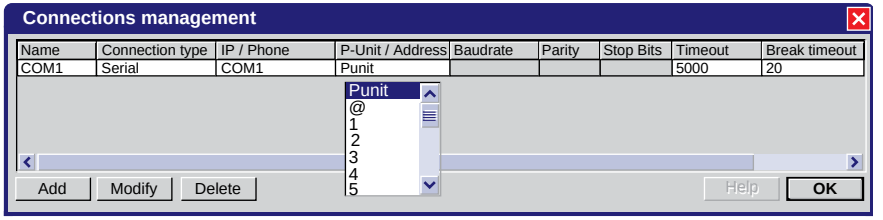
以太网连接管理

概览

以下信息描述了如何配置 / 添加 / 删除 / 选择一个 PC 到控制器的以太网 TPC/IP 连接。

设置新的 TCP/IP 连接

要在运行 TwidoSoft 应用程序的用户 PC 和网络上的 TWDLCAE40DRF 控制器之间建立一个以太网 TCP/IP 连接，请遵循如下步骤：
选择文件→首选项调用连接管理对话框：



步骤	动作
1	单击增加按钮，位于连接管理对话框。 结果：添加了一个新的连接行。新行显示的为默认连接设置。用户需要修改这些设置。 **注意：**要在域中设置一个新的值，用户有两种选择： ● 选择要修改的区域点击修改。 ● 双击所选域。
2	在名字域中，输入连接的描述性名称。有效的名称最多包含 32 个字母字符。
3	在连接方式域，点击下拉菜单，包括：**TCP/IP**，串口， Modem （如有）和 USB （如有）。 选择 **TCP/IP** 在用户希望在 PC 和可连接以太网的 Twido 控制器之间建立一个新的以太网连接时。
4	在 **IP / Phone** 域中输入一个有效的 **IP** 地址，这是希望连接到的 Twido TWDLCAE40DRF 控制器的 IP 信息。 **IP** 地址：输入上一步中用户为 Twido 控制器指定的 IP 地址。
5	此 **Punit / Address** 域能填写，如 IP / Phone 区域已选择。对于 TCP/IP，默认值是直接对于串口，默认值是 **Punit**。当这些被选择时，接下来的三项 （波特率，奇偶校验和停止位）被禁止。 如不知道控制器的地址， @ 允许你稍后选择，当下载程序时再选择。（在第一次连接前，有一个弹出菜单让你选择要传输的控制器，地址在 1-247 的范围，1 是默认值。）

步骤	动作
6	在超时域中，输入一个 Twido 与控制器建立连接的超时值，单位为毫秒（ms）。超时结束后， **PC** 与控制器的连接失败，则 TwidoSoft 应用程序将放弃试图建立连接。要继续尝试建 立新的连接，请选择 **PLC** → 选择连接。 **注意：**默认值是 **500** ms。最大的超时是 65535 × 100 ms (6553.5 s)。
7	此断开时间是 Modbus TCP/IP 请求和接收响应帧之间所允许的最长时间。如果未接收到响应帧，而中断超时时间已过，则 TwidoSoft 应用将中断 PC 和控制器之间的连接。 **注意：**默认中断超时值是 20 ms。需设定非零值。
8	单击 **OK** 按钮保存新的连接设置并关闭连接管理对话框。 **结果：**所有新增的连接名被增加到文件 → 首选项 – 连接的下拉菜单中。或在 **PLC** → 选择连接访问它。

修改或删除 TCP/IP 连接

现存的以太网 TCP/IP 连接可被删除或对其参数进行修改，如下所示：

- 要在以太网管理对话框中删除一个连接，单击要删除的连接的名称，然后点击删除按钮。注意删除之后，所有的连接参数将永久性丢失。
- 要修改已存的连接参数，单击要修改的连接的名称，点击修改按钮。然后，用户可以在所选域中输入新的值。

以太网 LED 指示灯

概览

两个以太网通信 LED 指示灯位于 TWDLCAE40DRF 控制器前面板，也可以通过 **PLC** > 查看 **PLC** 访问 TwidoSoft 应用程序。其标记如下：

- LAN ACT
- LAN ST

以太网 LED 为以太网端口的状态和诊断提供连续的监控。

LED 状态

下表描述的状态为 **LAN ACT** 和 **LAN ST** 以太网 **LED** 指示灯。

LED	状态	颜色	描述
LAN ACT	关闭	-	RJ-45 端口上无以太网信号。
	稳定	绿色	10BASE-TX 链接信号指示 10 Mbps 连接。
	闪烁		数据包在 10BASE-TX 连接上进行发送或接收。
	稳定	琥珀色（黄色）	100BASE-TX 链接信号指示 100 Mbps 连接。
	闪烁		数据包在 100BASE-TX 连接上进行发送或接收。
LAN ST	稳定	绿色	基本控制器上电。以太网端口准备在网络上通信。
	快速闪烁		以太网上电时初始化。
	2 次闪烁，然后关闭		无有效的 MAC 地址。
	3 次闪烁，然后关闭		可能是以下任何一种情况： ● 未检测到链接。 ● 以太网电缆未正确插入或电缆故障。 ● 网络设备 （集线器 / 交换机）故障或未正确配置。
	4 次闪烁，然后关闭		网络上检测到了重复 IP。（要修复此情况，请为 Twido 控制器指定一个不同的 IP 地址。）
	5 次闪烁，然后关闭		没有接收到服务 IP 地址；等待服务 IP 地址。
	6 次闪烁，然后关闭		使用有效的默认 IP 地址； FDR 安全模式。
	9 次闪烁，然后关闭		以太网硬件故障。

TCP Modbus 消息

概览

用户可以使用 TCP Modbus 通讯功能使 Modbus TCP 客户端 （主控制器）与 Modbus TCP 服务器 （从控制器）之间发送和接收以太网信息。由于 TCP Modbus 是一个端对端通信协议， Twido 可连接以太网的控制器既可以作客户端也可以作服务器，这取决于发出还是应答请求。

以太网上的信息交换

以太网上的信息由 EXCH3 指令和 %MSG3 功能模块处理：EXCH3 也支持路由到以太网主机或通过网关通信。

- **EXCH3 指令**：发送 / 接收消息
- **%MSG3 功能块**：控制消息交换。

EXCH3 指令

EXCH3 指令允许 Twido 控制器发送和 / 或接收信息到 / 从以太网节点。用户自定义一个字表 （%MWi:L），包含控制信息和要发送和 / 或接收的数据 （发送和 / 或接收最多 128 个字节），字表的格式由下面部分描述。
使用 EXCH3 指令进行信息交换：

语法：[EXCH3 %MWi:L]
在这里：L = 在控制字，传输和接收表中的字数

Twido 控制器必须完成来自第一条 EXCH3 指令的信息交换，在第二条指令开始之前。 %MSG3 功能模块在发送多条信息时必须使用。
EXCH3 指令的处理过程立即开始，任何传输都在中断控制下进行 （数据接收也受中断控制），即为后台处理。

注意：EXCH3 的使用方法与一般的 Modbus 使用 EXCHx （当 x = 1 或 2）相同。指令语法也相同。然而，发送和接收表中 Byte1 携带的信息却不同。一般的 Modbus 的 Byte1 传输从控制器的串行口连接地址，而 TCP Modbus 的 Byte1 传输索引号，此索引号属于 Modbus TCP 客户端控制器。索引被指定和存储在 TwidoSoft 以太网配置的远程设备表 *（见远程设备表， p. 191）*。

EXCH3 字表

发送和 / 或接收帧的最大长度为 128 个字节 （注意此限制只适用于 TCP Modbus 客户端，而 TCP Modbus 服务器端支持标准 Modbus PDU 长度：256 个字节）。
EXCH3 指令的字表由控制，发送和接收表组成，如下所示：

	高字节	低字节
控制表	命令	长度 （发送 / 接收）
	接收偏移	发送偏移
发送表	发送字节 1 （索引号，与 TwidoSoft 以太网配置对话框中的远程设备表中指定的相一致）	发送字节 2 为 Modbus 编号
	...	发送字节 n
	发送字节 n+1	
接收表	接收字节 1 （索引号，与 TwidoSoft 以太网配置对话框中的远程设备表中指定的相一致）	接收字节 2 为 Modbus 编号
	...	接收字节 p
	接收字节 p+1	

%MSG3 功能块

%MSG3 功能的使用方法与一般的 Modbus 使用 %MSGx 相一致。%MSG3 用作管理数据交换，如下所示：

- 通信错误校验
- 多消息协调
- 传输优先消息

%MSGx 功能模块有一个输入，两个输出：

输入 / 输出	定义	描述
R	输入复位	置为 1：通信重新初始化或模块重置（%MSGx.E = 0 和 %MSGx.D = 1）。
%MSGx.D	通信完成	0：程序请求。 1：通信完成，如果：传输完毕，字符接收完毕，出错，或模块重置。
%MSGx.E	错误	0：消息长度正确且连接正确。 1：命令错误，表配置错误，接收字符错误（速率，奇偶校验，等等。），或接收表满。

EXCH3 错误编码

EXCH3 指令发生错误时：

- 位 **%MSG3.D** 和 %**MSG3.E** 被设为 **1**，和
- 以太网通信错误编码被记录在系统字 **%SW65**。

下表显示了 EXCH3 错误编码：

EXCH3 错误编码 （记录在系统字 %SW65 中）
适用于所有 EXCHx(x = 1, 2, 3) 的标准错误编码： 0 - 操作成功 1 传输字节数过大 （> 128) 2 - 发送表太小 3 - 字表太小 4 - 接收表溢出 5 - 超时 （注意，错误编码 5 在 EXCH3 指令中无效，由以太网指定错误编码 109 和 122 代替，如下所示。） 6 - 发送 7 - 表中的错误命令 8 - 所选端口未配置 / 不可用 9 - 接收错误 10 - %KW 不可用于接受 11 - 发送偏移超出发送表 12 - 接收偏移超出接收表 13 - 控制器终止 EXCH 处理
EXCH3 的以太网专用错误编码： 101 - 无此 IP 地址 102 - TCP 连接中断 103 - 无 socket 可用 （所有连接信道都处于繁忙状态） 104 - 网络关闭 105 - 网络不可用 106 - 重启时网络中断连接 107 - 由对等设备中断了连接 108 - 由对等设备重置连接 109 - 连接超时 110 - 连接请求失败 111 - 主机关闭 120 - 未知索引 （远程设备在配置表中无索引） 121 - 致命错误 （MAC ；芯片；重复 IP） 123 - 以太网正在初始化

内置式模拟功能

7

概览

本章的主题

本章描述了怎样管理内置式模拟通道和电位器。

本章包含了哪些内容?

本章包含了以下主题：

模拟电位器

介绍

Twido 控制器具有：

- 一个模拟电位器，对于 TWDLC•A10DRF， TWDLC•A16DRF 控制器和所有的模块型控制器 （TWDLMDA20DTK, TWDLMDA20DUK, TWDLMDA20DRT, TWDLMDA40DTK 和 TWDLMDA40DUK,
- 两个模拟电位器，对于 TWDLC•A24DRF 和 TWDLCA•40DRF 控制器。

编程

模拟电位器 1 的数值从 0 到 1023，模拟电位器 2 的数值从 0 到 511，对应着电位器提供的模拟值，并包含在下面两个输入字中：

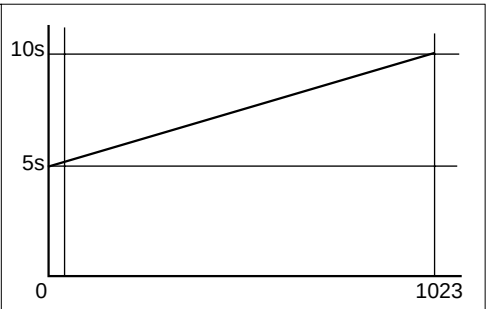
- %IW0.0.0 对应模拟电位器 1 （左边）
- %IW0.0.1 对应模拟电位器 2 （右边）

这些字可用于算术操作。它们可用于任何形式的调节，例如，预置延迟时间或计数器，调节脉冲发生器的频率或机器预热时间。

举例

用模拟电位器 1 将延迟时间从 5 s 调为 10 s：

这个调节实际上用到了模拟电位器 1 的全部调节范围：从 0 到 1023。

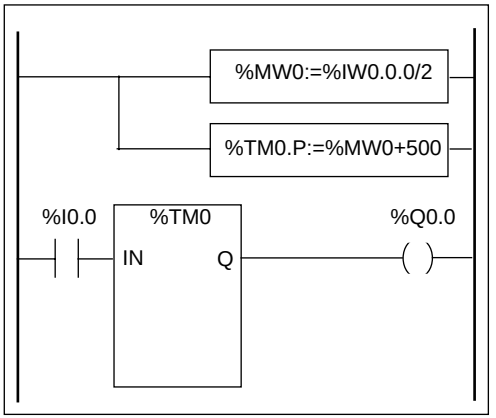


下面参数被选择用于配置时间延迟模块 %TM0：

- 类型 TON
- 时基：10 ms

时间延迟的预置值从电位器的调节值计算出来，计算方程为 %TM0.P: = (%IW0.0.0/2)+500。

上面示例的代码为：




```
LD      1
[%MW0:=%IW0.0.0/2]
[%TM0.P:=%MW0+500]
BLK     %TM0
LD      %I0.0
IN
OUT_BLK
LD      Q
ST      %Q0.0
END_BLK
.....

END
```

模拟通道

介绍

所有的模块型控制器 （TWDLMDA20DTK, TWDLMDA20DUK,TWDLMDA20DRT, TWDLMDA40DTK 和 TWDLMDA40DUK）都有一个内置式模拟通道。其电压输入范围 0 到 10 V，数字信号 0 到 511。模拟通道做八路采样，最后采用它们的平值。

原理

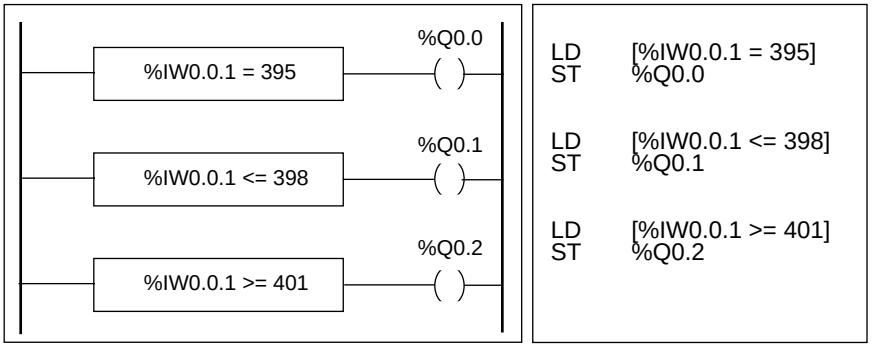
模数转换器将 0 到 10 V 的输入电压采样为 0 到 511 的数字值。该值存储于系统字 %IW0.0.1 中。由于值的转换是线性的，所以每个增加的数字相当于 20 mV （10V/512）。一旦系统检测到值 511，通道即视为饱和。

编程举例

控制一个烤炉的温度：烹饪的温度设为 350 ℃，温度波动 +/- 2.5 ℃，分别导致输出 %Q0.0 和 %Q0.2 的状态变化。此例中用到了模拟通道的全部范围 0 到 511。各温度点的模拟设置为：

温度 （℃）	电压	系统字 %IW0.0.1
0	0	0
347.5	7.72	395
350	7.77	398
352.5	7.83	401
450	10	511

上面示例的代码为：



模拟模块管理

8

概览

本章的主题

本章提供了 Twido 控制器模拟模块管理概述。

本章包含了哪些内容?

本章包含了以下主题：

模拟模块概述

介绍

除了内置式 10 位电位器和 9 位模拟通道，所有支持扩展 I/O 的 Twido 控制器都能配置模拟 I/O 模块，并和它们通信。
这些模拟模块是：

名字	点数	信号范围	编码
TWDAMI2HT	2 输入	0 - 10 V 或 4 - 20 mA	12 位
TWDAMO1HT	1 输出	0 - 10 V 或 4 - 20 mA	12 位
TWDAMM3HT	2 输入，1 输出	0 - 10 V 或 4 - 20 mA	12 位
TWDALM3LT	2 输入，1 输出	输入 Th 或 PT100，输出 0-10V 或 4-20mA	12 位
TWDAVO2HT	2 输出	+/- 10 V	11 位 + sign
TWDAMI4LT	4 输入	0 - 10 V，0 - 20 mA，NI 或 PT3- 线制传感器	12 位
TWDAMI8HT	8 输入	0 - 10 V 或 0 - 20 mA	10 位
TWDARI8HT	8 输入	NTC 或 PTC 传感器	10 位

模拟模块操作

输入和输出字 （%IW 和 %QW）用于用户程序和模拟模块之间的数据交换。这些字的更新在运行模式下与控制器扫描同步完成。

注意

意外的设备启动

当控制器置于停止模式时，模拟输出将置于可靠位置。本例中，默认设定点是零。

如果不遵守这个警告将会导致人身伤害或设备损害。

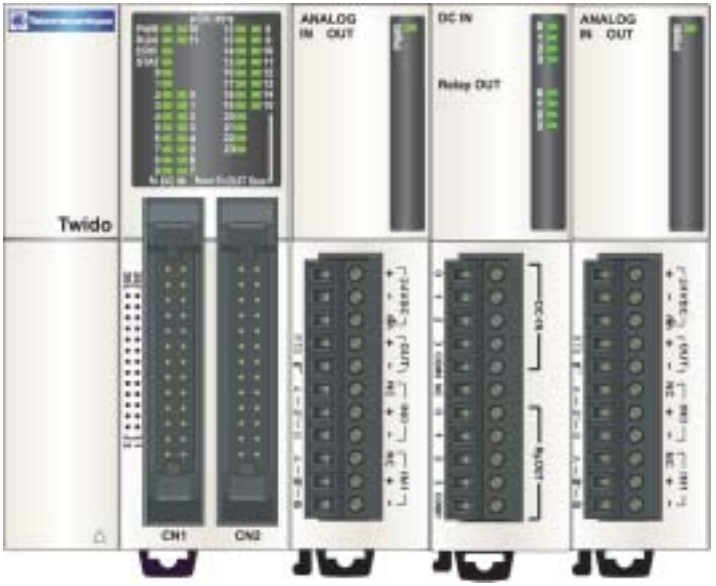
模拟输入和输出寻址

介绍

模拟通道的分配地址取决于它们在扩展总线中的位置。

模拟 I/O 寻址示例

此例中， TWDLMDA40DUK 有一个内置式模拟调节 10 位电位器，一个 9 位内置式模拟通道。扩展总线上配置有：一个 TWDAMM3HT 模拟模块，一个 TWDDMM8DRT 输入 / 输出继电器模块，和另外一个 TWDAMM3HT 模拟模块。



本体单元　模块 1　模块 2　模块 3

下表是每个输出寻址的详细情况。

描述	本体单元	模块 1	模块 2	模块 3
电位器 1	%IW0.0.0			
内置式模拟通道	%IW0.0.1			
模拟输入通道 1		%IW0.1.0		%IW0.3.0
模拟输入通道 2		%IW0.1.1		%IW0.3.1
模拟输出通道 1		%QW0.1.0		%QW0.3.0
数字输入通道			%I0.2.0 - %I0.2.3	
数字输入通道			%Q0.2.0 -%Q0.2.3	

模拟输入和输出配置

介绍

本部分提供了模拟输入和输出的配置信息。

模拟 I/O 配置

此模块配置对话框用于管理模拟模块的参数。
你能通过应用管理器或硬件访问它。

从应用管理器	在硬件菜单
1. 选择模块。	1. 选择配置模块。
2. 右键点击配置，直接打开配置模块 （模块 **ref.** 和位置）对话框。	2. 从配置模块 - 选择模块对话框选择一模块。
	3. 从打开的配置模块对话框 （模块 **ref.** 和位置）调整参数。

注意：当控制器没有连接时，您可以离线修改参数。

标题栏和内容

标题栏显示在扩展总线中的模块参数和位置。
对话框上部显示描述区域。
表格显示：地址，符号，类型，范围，最小，最大和单位

- 对于 TWDAMI4LT 和 TWIDAMI8HT，本表前有一个输入类型列表。
- 对于 TWDAVO2HT 和 TWDAMI8HT，此类型栏用栏检查框替代。
- 对于 TWDARI8HT，每一通道 （0-7）能在表内分别配置，能选择有图表或公式配置方法。这个表可以在 **Recap** 页中看到。

描述

此描述域显示一个模块摘要。

地址

电子表格的每一行表示模块的一个输入和输出通道。
每一个地址都在下表中标出，“i”表示扩展总线模块地址。

模块名称	地址
TWDALM3LT	2 输入 （%IWi.0, %IWi.1）， 1 输出 （%QWi.0）
TWDAMM3HT	2 输入 （%IWi.0, %IWi.1）， 1 输出 （%QWi.0）
TWDAMI2HT	2 输入 （%IWi.0, %IWi.1）
TWDAMO1HT	1 输出 （%QWi.0）
TWDAVO2HT	2 输出 （%QWi.0, %QWi.1）
TWDAMI4LT	4 输入 （%IWi.0 至 %IWi.3）
TWDAMI8HT	8 输入 （%IWi.0 至 %IWi.7）
TWDARI8HT	8 输入 （%IWi.0 至 %IWi.7）

符号

如果赋值的话，这显示的是一个只读地址符号。

输入输出类型

这标明一个通道模式。通道模式取决于模块型号和通道。
对于 TWDAMO1HT，TWDAMM3HT 和 TWDALM3LT，能如下配置单一输出通道：

类型
不被使用
0 - 10 V
4 – 20 mA

对于 TWDAMI2HT 和 TWDAMM3HT，能如下配置两个输入通道：

类型
不被使用
0 - 10 V
4 – 20 mA

对于 TWDALM3LT，能如下配置两个输入通道：

类型
不被使用
热电偶 K
热电偶 J
热电偶 T
PT 100

对于 TWDAVO2HT，不能调整类型。
对于 TWDAMI4LT，能如下配置四个输入通道：

输入类型	类型
电压	不使用 0-10 V
电流	不使用 0-20 mA
温度	不使用 PT 100 PT 1000 NI 100 NI 1000

对于 TWDAMI8HT，能如下配置八个输入通道：

输入类型
0 - 10 V
0 - 20 mA

对于 TWDARI8HT，能分别配置每一通道 （0-7），从操作域，直接选择模式和范围，如需要，能显示所有信息的摘要。用类型栏显示：

类型
不被使用
NTC / CTN
PTC / CTP

注意

设备损坏

如果您将输入与电压测量值相接，而您的 TwidoSoft 配置却是电流类型，您将永久性地损害模拟模块。确信连线与 TwidoSoft 配置一致。

如果不遵守这个警告将会导致 人身伤害或设备损害。

范围

用来确定通道的数值范围，取决于模块和通道类型的选择。
一旦类型配置完成，就能设定相应的范围。如需要，与单元一起用来表示可接受的最小和最大值 - 或者是固定的或是用户定义的。

范围 （NTC 传感器）	最小值	最大值	单位	输入 / 输出模拟模块
正常	0	4095	无	TWDALM3LT TWDAMO1HT TWDAMM3HT TWDAMI2HT TWDAMI4LT
	-2048	2047		TWDAVO2HT
	0	1023		TWDAMI8HT TWDARI8HT
自定义	用户定义最小值为 -32768	用户定义最大值为 32767	无	所有输入 / 输出模拟模块
˚C	-1000	5000	0.1˚C	TWDALM3LT
	根据用户设定的参数由 TwidoSoft 动态更新			TWDARI8HT
	-2000	6000		TWDAMI4LT（铂传感器）
	-500	1500		TWDAMI4LT（镍传感器）
˚F	-1480	9320	0.1˚F	TWDALM3LT
	根据用户设定的参数由 TwidoSoft 动态更新			TWDARI8HT
	-3280	11120		TWDAMI4LT（铂传感器）
	-580	3020		TWDAMI4LT（铂传感器）
电阻	100	10000	Ω	TWDARI8HT
	74	199		TWDAMI4LT (Ni100)
	742	1987		TWDAMI4LT (Ni1000)
	18	314		TWDAMI4LT (Pt100)
	184	3138		TWDAMI4LT (Pt1000)

图表或公式方式

对于 TWDARI8HT，每一通道 （0-7）能单独配置。检查使用框，然后在图表和公式配置方式中选择。

- **图表方式**
  （**R1, T1**）和 （**R2, T2**）对应曲线的两点。
  R1（默认 8700）和 **R2**（默认 200），单位是Ω。
  T1（默认 233.15）和 **T2**（默认 398.15），单位在单位列表中选择：
  单位列表框：绝对温度 （默认），˚C 或˚F。
  注意：设定 T1 和 T2 后，再改变温度单位， T1 和 T2 不会对应新的温度单位重新计算。
- **公式方式**
  如果知道 **Rref**， **Tref** 和 **B**，能用此方法定义传感器特性。
  Rref（默认 330）单位Ω。
  B 默认 3569 （最小 1，最大 32767）。
  Tref（默认 298.15）单位按设定值单位列表框：绝对温度 （默认），˚C 或˚F。
  这是相应最小 / 最大值的表，单元间的 **Tref** 值为：

单位	最小值	最大值
绝对温度	1	650
摄氏温度	-272	376
华氏温度	-457	710

在图表和公式中，允许在当前的配置通道中导入另一个通道的值：
1. 在通道编号框中选择通道号。
2. 按导入值按钮导入值。

一些错误和警告消息与这些窗口有关。

注意：如果在开始设定参数值后从图表切换到公式或从公式切换到图表，将有一警告消息弹出：所有的修改的参数将丢失，将返回默认值。

模拟模块状态信息

状态表　　下表是您需要监控的模拟 I/O 模块的状态信息。

系统字	功能	描述
%SW80	基本 I/O 状态	位 [0] 通道处于正常操作 （对于通道所有操作） 位 [1] 模块初始化 （或所有通道的信息初始化） 位 [2] 硬件故障 （外部电源故障，所有通道的普通故障） 位 [3] 模块配置错误 位 [4] 通道 0 输入数据转换处理中 位 [5] 通道 1 输入数据转换处理中 位 [6] 通道 0 输入热电偶没有配置 位 [7] 通道 1 输入热电偶没有配置 位 [8] 不被使用 位 [9] 没有使用 位 [10] 通道 0 模拟输入数据超出范围 位 [11] 通道 1 模拟输入数据超出范围 位 [12] 连线错误 （通道 0 模拟输入数据低于电流范围，电流回路开路） 位 [13] 连线错误 （通道 1 模拟输入数据低于电流范围，电流回路开路） 位 [14] 没有使用 位 [15] 输入通道不可用
%SW81	扩展 I/O 模块 1 状态：定义与 %SW80 相同	
%SW82	扩展 I/O 模块 2 状态：定义与 %SW80 相同	
%SW83	扩展 I/O 模块 3 状态：定义与 %SW80 相同	
%SW84	扩展 I/O 模块 4 状态：定义与 %SW80 相同	
%SW85	扩展 I/O 模块 5 状态：定义与 %SW80 相同	
%SW86	扩展 I/O 模块 6 状态：定义与 %SW80 相同	
%SW87	扩展 I/O 模块 7 状态：定义与 %SW80 相同	

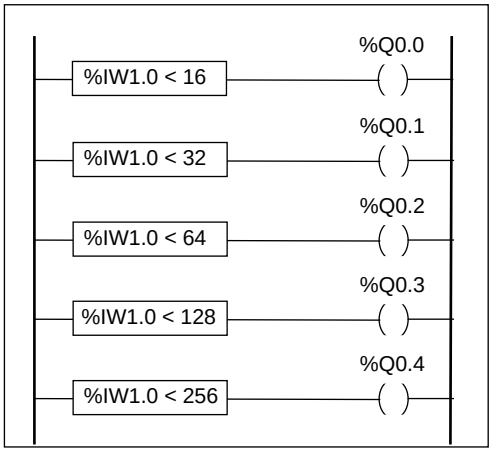
模拟模块使用示例

介绍

本部分提供了一个 Twido 可用模拟模块的使用示例。

示例：模拟输入

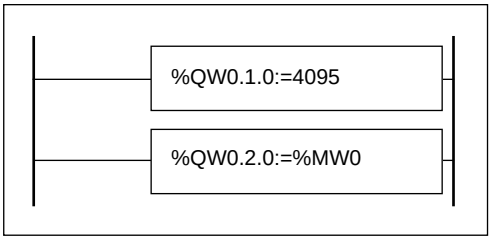
此例将模拟输入信号与五个阈值进行比较。模拟输入被比较后，如果它小于或等于某个阈值，基本控制器将置上对应位。




```
LD [%IW1.0 < 16]
ST %Q0.0

LD [%IW1.0 < 32]
ST %Q0.1

LD [%IW1.0 < 64]
ST %Q0.2

LD [%IW1.0 < 128]
ST %Q0.3

LD [%IW1.0 < 256]
ST %Q0.4
```

示例：模拟输出

以下程序使用 2 个模拟输出模块。模块 1 为 “常规” 范围 10v 输出，模块 2 为自定义范围输出：




```
LD   1
[%QW0.1.0:=4095
LD   1
[%QW0.2.0:=%MW0
```

- 输出值为 %QW1.0=4095 （正常情况）的示例：

下表为 %QW1.0 的输出电压值：

	数字值	模拟值 （v）
最小值	0	0
最大值	4095	10
值 1	100	0.244
值 2	2460	6

- 自定义范围 （最小值 =0，最大值 =1000）的输出值示例：

下表为 %QW2.0 的输出值：

	数字值	模拟值 （v）
最小值	0	0
最大值	1000	10
值 1	100	1
值 2	600	6

AS-I V2 总线安装

9

概览

本章的主题

本章介绍 AS-I 主模块 TWDNOI10M3 和它的从模块的软件安装信息。

本章包含了哪些内容?

本章包含了以下主题：

AS-I V2 总线介绍

介绍

AS-I 总线 （Actuator Sensor-Interface，执行器传感器接口）允许传感器设备 / 执行器在自动控制的最底层通过一根电缆互联。

这些传感器 / 执行器在此文档中定义为从设备。

为了实现 AS-I 应用，您需要定义应用所在的物理环境 （扩展总线，电源，处理器，模块，连接到总线的 AS-I 从设备），然后保证软件的可执行性。

第二个方面即软件可通过不同的 TwidoSoft 编辑器来实现：

- 或者本地模式，
- 或者在线模式。

AS-I V2 总线

AS-I 接口主模块 **TWDNOI10M3** 包含下列功能：

- M3 表：此表包含 AS-I V2 标准定义的所有功能，但不支持 S7-4 模拟表
- 每个模块有一个 AS-I 通道
- 自动寻址地址为 0 的从设备
- 管理表和参数
- 总线输入的极性反向保护

AS-I 总线允许：

- 最多 31 个标准地址从设备和 62 个扩展地址从设备
- 最多 248 输入和 186 输出
- 最多 7 个模拟量从设备 （每个从设备最多 4 I/O）
- 最大循环时间 10 ms

最多 2 个 AS-I 主模块能用于模块型控制器或 TWDLC•A24DRF、 TWDLCA•40DRF 一体型控制器。

总的功能描述

总的介绍

对于 AS-I 配置， TwidoSoft 软件允许用户：

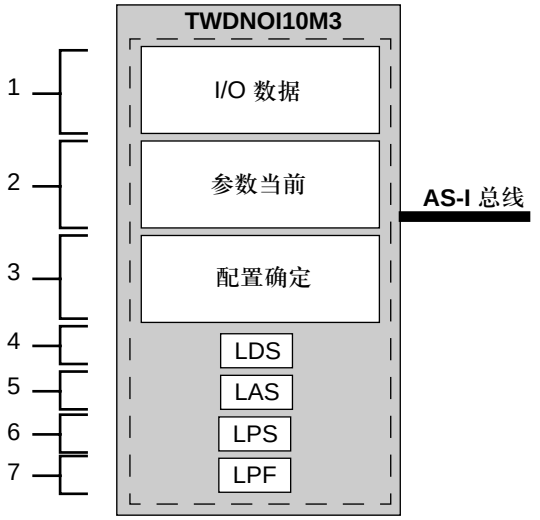
- 手动配置总线 （从设备声明和总线中地址分配）
- 根据总线当前情况修改配置
- 了解从机参数
- 控制总线状态

基于这个原因，所有来或去 AS-I 主模块的数据将存储在特定对象 （字和位）中。

AS-I 主机结构

AS-I 模块包含数据区，允许您管理从设备列表和输入 / 输出数据映像。这些信息存储在 RAM 存储器中。

下图显示了 **TWDNOI10M3** 模块结构。



注释：

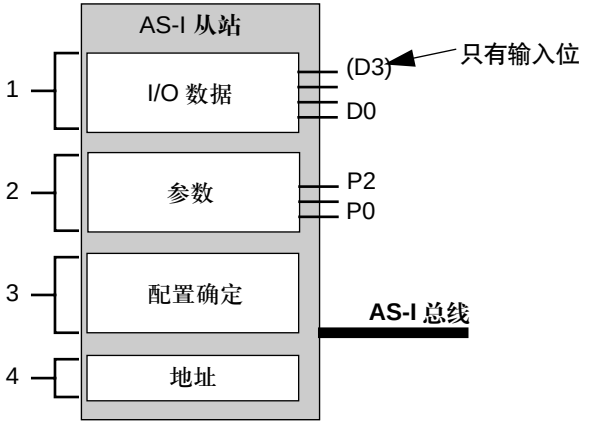
地址	条目	描述
1	I/O 数据 (IDI, ODI)	AS-I V2 总线 248 个输入和 186 个输出映像。
2	当前参数 (PI, PP)	所有从设备参数映像。
3	配置 / 辨识 (CDI, PCD)	此区域包含所有检测到的从设备的 I/O 代码和辨识代码。
4	LDS	总线上所有检测到的从设备列表。
5	LAS	总线上所有活动从设备列表。
6	LPS	总线上提供且通过 TwidoSoft 配置的从设备列表。
7	LPF	具有设备故障的从设备列表。

从设备结构

标准地址从设备均具有：
- 4 个输入 / 输出位
- 4 个参数位

扩展地址的从设备均具有：
- 4 个输入 / 输出位 （最后一位保留，仅供登录使用）
- 3 个参数位

每个从设备都有自己的地址，表和子表 （定义变量交换）。
下图显示了一个扩展地址从设备的结构：



注释：

地址	条目	描述
1	输入 / 输出数据	输入数据被从设备存储并可为 AS-I 主模块使用。 输出数据由主模块更新。
2	参数	参数用于控制和转换内部工作模式到传感器或执行器。
3	配置 / 辨识	此区域包含： ● 对应 I/O 配置的代码， ● 从设备辨识 （ID）代码， ● 从设备辨识代码 （ID1 和 ID2）。
4	地址	从设备的物理地址。
注意：工作参数，地址，配置和辨识数据都存储在永久性存储器中。		

软件安装原则

概览

根据 TwidoSoft 规则，用户应该采用渐进的方法来创建 AS-I 应用程序。

安装原则

用户必须知道怎样从功能上配置他的 AS-I 总线。（参见*如何插入从设备到现存 AS-I V2 配置，p. 248*。）

下表显示了 AS-I 总线软件执行的不同阶段。

模式	阶段	描述
本地	模块声明	AS-I 主模块 TWDNOI10M3 在扩展总线上的位置选择。
	模块通道配置	“主”模式选择。
	从设备声明	为每个设备选择： ● 总线上的位置编号， ● 从设备标准或扩展地址类型。
	配置参数确认	从设备级确认。
	应用程序全局确认	应用程序级确认。
本地或被连接	符号化 （可选）	从设备相关变量符号化。
	编程	AS-I V2 功能编程。
被连接	传递	传递应用程序到 PLC。
	调试	通过下面帮助调试应用程序： ● 调试屏幕，一方面用于显示从设备 （地址，参数），另一方面用于分配它们相应的地址， ● 诊断屏幕允许辨识错误。

注意：扩展总线上 AS-I 主模块的声明和删除与其它扩展模块一样。但是，一旦扩展总线上声明了两个 AS-I 主模块， TwidoSoft 将不允许再声明其它 AS-I 主模块。

连接之前注意

在连接 （通过软件） PC 到控制器之前，并为了避免出现问题：

- 确信总线上不物理存在地址为 0 的从设备
- 确信不在物理上存在 2 个地址相同的从设备。

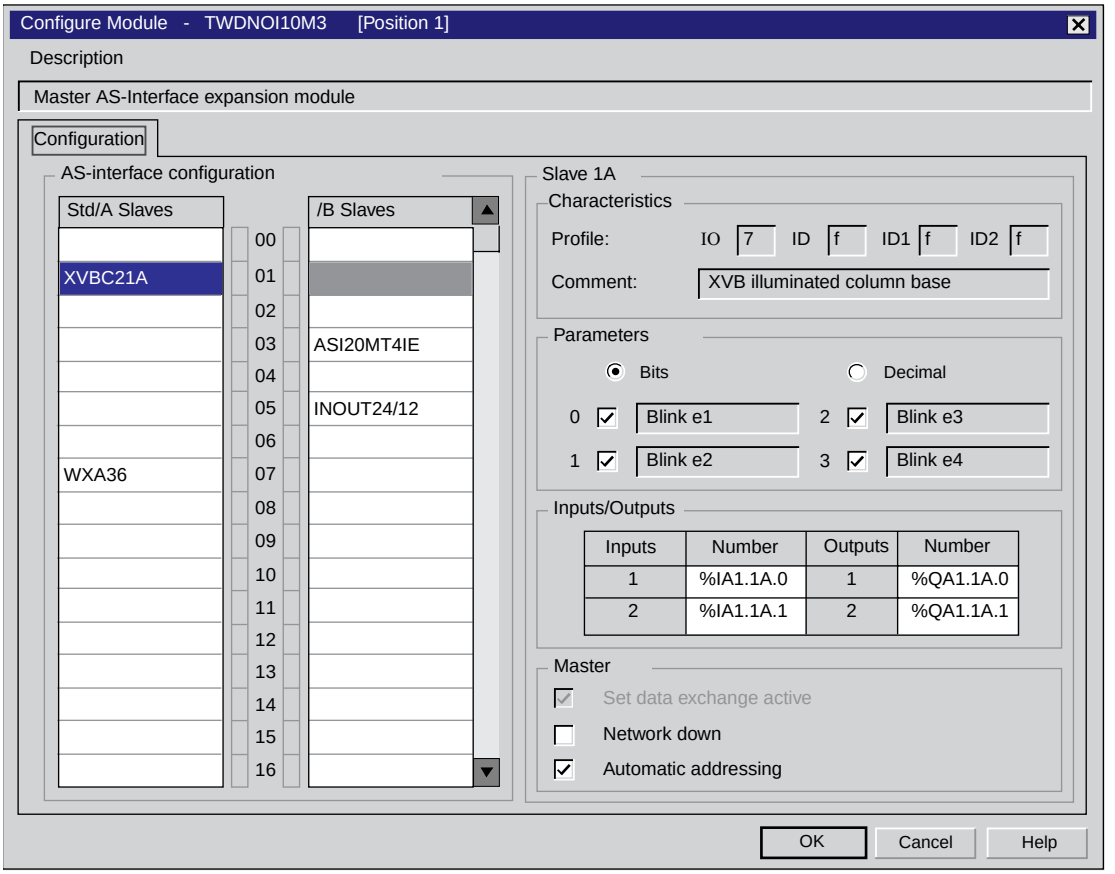
AS-I 总线配置屏幕描述

概览

AS-I 主模块配置屏幕可以访问模块和从设备相关参数。
它可以在离线模式下显示和修改参数。

离线模式图例

离线模式下配置屏幕图例：



离线模式下屏幕描述

该屏幕将总线所有数据组成三块信息：

块	描述
AS- 接口配置	用户期望的总线映像：总线上从设备的标准和扩展地址设置查看。向下移动垂直条光标到达下面地址。 变成灰色的地址对应从设备配置中不可用的地址。 例如，如果一个新的从设备标准地址设置被声明为 1A，则地址 1B 自动变成灰色。
从设备 xxA/B	被选从设备配置： ● 特性：IO 码，ID 码，ID1 和 ID2 码（表），以及从设备注释， ● 参数：参数列表 （可修改），根据用户判断用二进制 （4 个检查框）或十进制 （1 个检查框）表， ● 输入 / 输出：有效 I/O 列表和它们各自的地址。
主模式	AS-I 模块的 2 种可用功能是可以激活或不激活的（例如，自动寻址）。 “网络故障”可强迫 AS-I 总线进入离线模式。 “自动寻址”模式默认被选中。 注意：“数据交换激活”功能尚不可用。

屏幕还包含 3 个按钮：

按钮	描述
OK	用于保存配置屏幕上显示的 AS-I 总线配置。 然后返回到主屏幕。 配置可被传送到 Twido 控制器。
Cancel	不承认修改并返回到主屏幕。
Help	打开屏幕帮助窗口。

注意：配置屏幕中的修改只能在离线模式下进行。

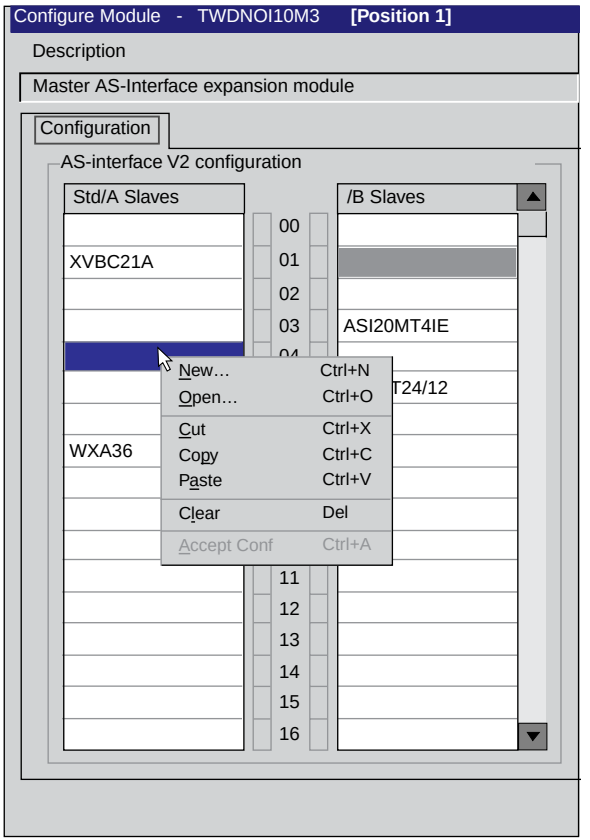
AS-I 总线配置

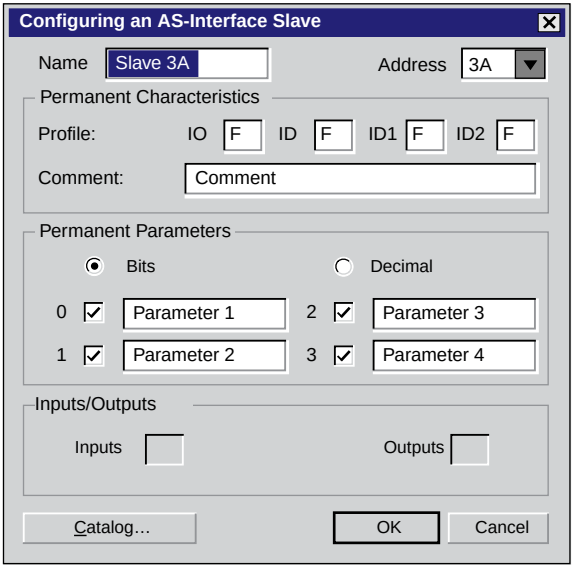
介绍

AS-I 总线配置在本地模式下通过配置屏幕完成。
一旦 AS-I 主模块和主模块模式被选定， AS-I 总线配置将由从设备配置组成。

从设备声明和配置过程

AS-I V2 总线从设备创建或修改过程：

步骤	动作
1	在总线映像的期望地址单元 （没有变成灰色）： ● 双击：到达步骤 3 或者 ● 右击： **结果：**  **注意：** 出现一个快捷菜单。用于： ● 配置总线新的从设备 ● 修改相应从设备的配置 ● 复制 （或 Ctrl+C），剪切 （或 Ctrl+X），粘贴一个从设备 （或 Ctrl+V） ● 删除一个从设备 （或 Del）

步骤	动作
2	在快捷菜单中选择： ● “新建”创建一个新的从设备：一个从设备配置屏幕显示出来；“地址”区域显示被选地址，“标识”区域被默认设置为 F，屏幕其它区域均为空白。 ● “打开”创建一个新的从设备或修改被选从设备的配置。对于新的从设备，一个新的从设备配置屏幕显示出来，“地址”区域显示被选地址，“标识”区域被默认设置为 F，屏幕其它区域均为空白。对于修改，从设备配置屏幕区域显示被选从设备以前的定义值。 新的从设备配置屏幕图例： 
3	在显示的从设备配置屏幕中，输入或修改： ● 新表的名字 （13 字符以内）， ● 注释 （可选）。 或单击 “目录 ..”从 AS- 接口表族内选择一个从设备。
4	输入： ● IO 代码 （对应输入 / 输出配置）， ● ID 代码 （标识符），（加上 ID1 表示扩展类型）。 **注意：** “输入”和 “输出”区域显示输入和输出通道数。当 IO 代码输入后，它们将被自动执行。

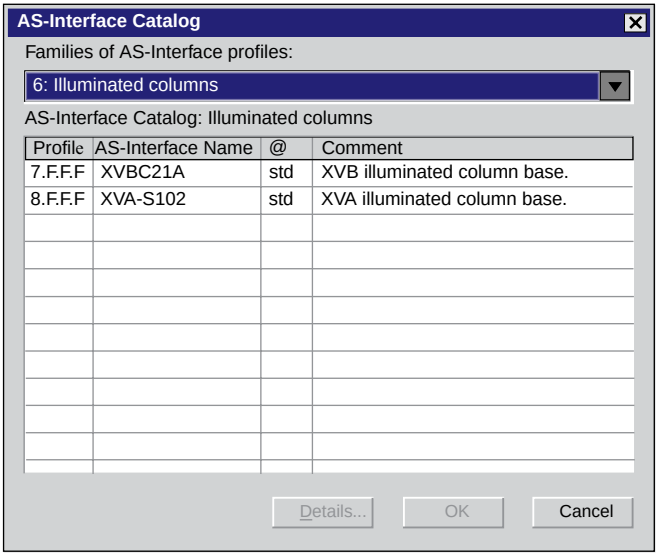
步骤	动作
5	对每个参数定义： ● 系统确认 （“二进制”视图被选框，或“十进制”视图中 0 和 15 之间的十进制值）， ● 比 “参数 X”更有明确意义的名字 （可选）。 **注意：** 被选参数是提供给 AS-I 主模块的永久参数的映像。
6	如果需要，可以通过点击地址右边的下拉列表框 （给定被授权的地址）或用键盘输入地址，来修改 “地址”（在总线可用地址限制之内）。
7	点击 “OK”按钮确认从设备配置。 结果将确认： ● IO 和 ID 被授权， ● 根据 ID 代码（如果 ID 代码等于 A，只有“bank”/B 从设备可用）授权从地址 （如果使用键盘）。 如果出错，将有一个错误消息警告用户 （例如：“本子站不能配置位该地址”），且屏幕重新显示初始化值 （根据错误出现表值或地址值）。

注意：软件限定模拟从设备声明数目不超过 7。

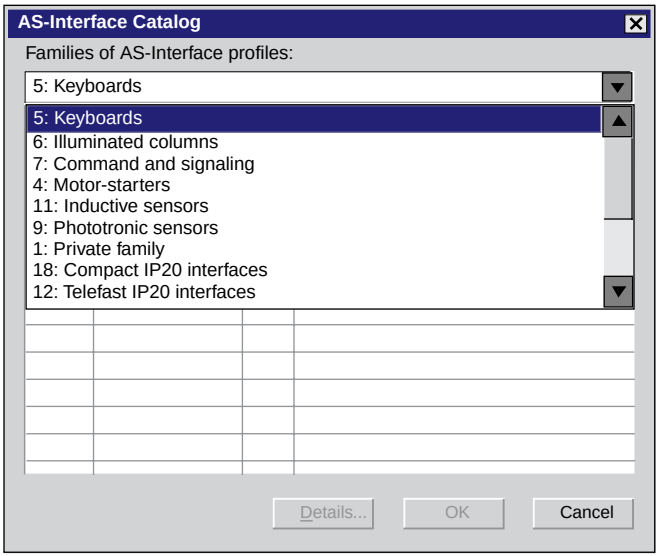
注意：关于 Schneider AS-I 目录：当点击目录时，可以建造和配置 “私有家族”中的从设备 （而不是 Schneider AS-I 目录中的从设备）。

AS-I 目录

目录按钮用于简化总线上从设备的配置。在使用 Schneider 家族的从设备时，使用此按钮可以简化和加速配置。
在窗口 “配置 AS-I 从设备”中点击 “目录”打开以下窗口：



通过下拉菜单可以访问 Schneider AS-I 目录中的所有家族：



一旦选中一个家族，相应的从设备列表即会出现。点击所需从设备通过点击 “OK” 使之生效。

注意：点击 “详细内容” 可以获得从设备的特性。

注意：也可添加和配置不属于 Schneider 目录中的从设备。只需选择私有家族并配置新的从设备。

调试屏幕描述

概览

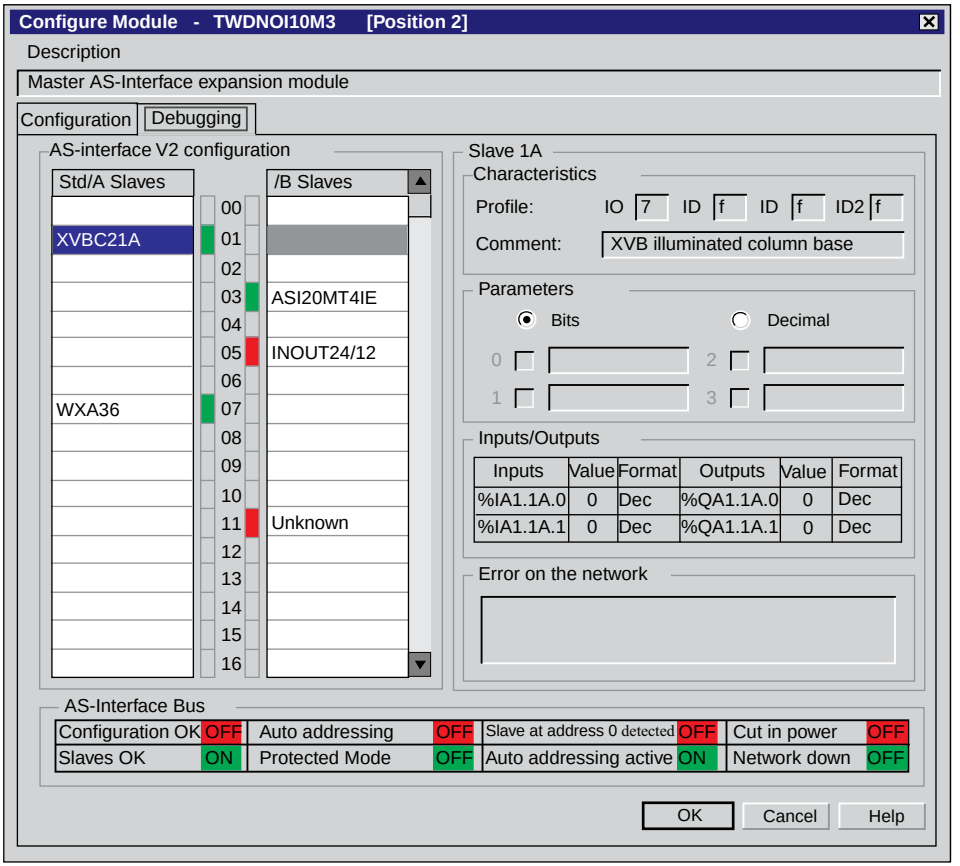
当 PC 被连接到控制器 （在下载应用程序到控制器之后），“Debug”标签出现在“Configuration”的右边；它允许进入调试屏幕。
调试屏幕动态地提供物理总线的映像，包括：

- 配置中指定从设备 （被输入）列表及它们的名字，和被检测的从设备 （名字未知，但其它均已指定）列表，
- AS-I 模块和从设备状态，
- 被选从设备的表，参数和输入 / 输出值的映像。

它可以让用户：

- 获得出错从设备的诊断 （参见*显示从设备状态，p. 238*），
- 可在连接模式下修改从设备地址 （参见*修改从设备地址，p. 239*），
- 传输从设备映像给配置屏幕 （参见*在在线模式下更新 AS-I 总线配置，p. 242*），
- 用指定地址寻址所有从设备 （第一次调试时）。

“调试”屏幕图例　调试屏幕（仅连接模式下）图例如下：



调试屏幕描述

“调试”屏幕提供的信息同配置屏幕。（参见*离线模式下屏蔽描述，p. 229*）。
下表列出了不同之处：

目录	描述
AS-I V2 配置	物理总线映像。 包括从设备状态： ● 绿色指示灯：此地址从设备处于活动状态。 ● 红色指示灯：此地址从设备出错，且消息通知您错误类型在“Error on the network”中。
从设备 xxA/B	被选从设备配置映像： ● 特性：检测到的表映像 （变成灰色，不能修改）， ● 参数：检测到的参数映像。用户只能选择参数显示格式， ● 输入 / 输出：检测到的输入 / 输出值显示，不能修改。
网络错误	通知您错误类型，如果被选从设备出错。
AS-I 总线	“Read Status”命令结果通知。 ● 显示总线状态：例如，“Configuration OK = OFF”表示用户指定配置与总线物理配置不对应， ● 显示 AS-I 主模块被授权的功能：例如，“Automatic addressing active = ON”表示主模块自动寻址被授权。

从设备状态显示

当对应地址的指示灯变红时，说明该地址的从设备出错。“Error on the network”窗口提供了被选从设备的诊断。
错误描述：

- 用户对给定地址的配置中的指定表与总线中该地址被检测到的实际表不一致 （诊断：“Profile error”），
- 总线检测到一个新的设备，没有在配置中指定：红色指示灯显示该地址且从设备名字显示 “未知”（诊断：“Slave not projected”），
- 外围设备故障，如果从设备支持 （诊断：“Peripheral fault”），
- 配置表被指定，但总线没有检测到该地址的从设备 （诊断：“Slave not detected”）。

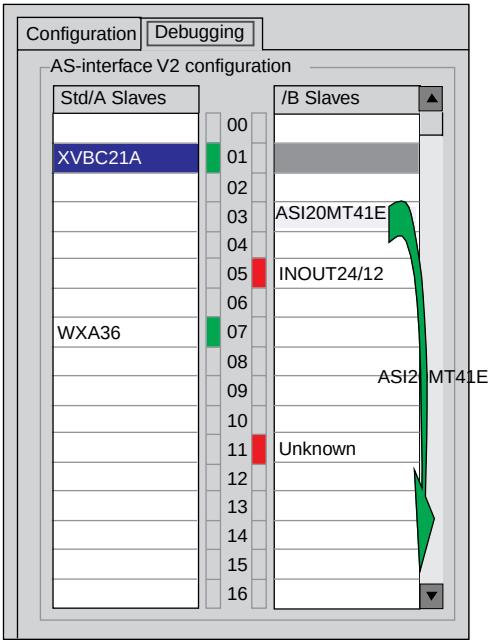
从设备地址修改

概览

从调试屏幕，用户可以在在线模式下修改从设备的地址。

从设备地址修改

下表显示了从设备地址修改过程：

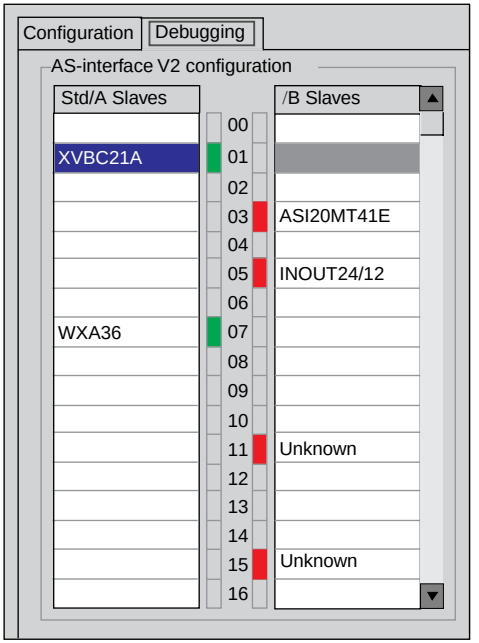
步骤	描述
1	进入 “Debug” 屏幕。
2	在 “AS-Interface V2 Configuration” 选择从设备。
3	拖放从设备到期望地址对应的单元。 图例：拖放从设备 3B 到地址 15B 

步骤	描述

结果：

所有的从设备参数被自动检查以示该操作是否可行。

结果图例：



该操作完成后，地址 3B 的从设备诊断显示 “slave not detected”表示该地址对应的从设备不再存在。选择地址 15B，被移动的从设备的表和参数被重新部署，但是从设备的名字为未知，因为该地址名字没有被指定。

注意：一个从设备的表和参数不与其名字关联在一起。不同名字的几个从设备可以具有相同的表和参数。

在线模式下 AS-I 总线配置更新

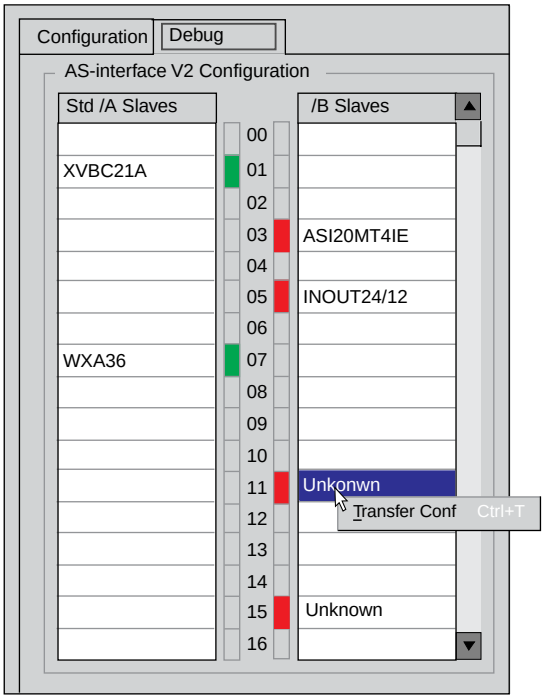
概览

在线模式下，配置屏幕的修改不被授权且物理配置和软件配置可以不同。配置屏幕可以考虑已配置或未配置从设备的表或参数的不同；实际上，在传递新的应用程序到控制器之前，可以发送修改到配置屏幕。
下面是跟踪考虑物理配置的过程：

步骤	描述
1	传递期望从设备配置到配置屏幕。
2	接受配置屏幕中的配置。
3	新配置确认。
4	传递应用程序到模块。

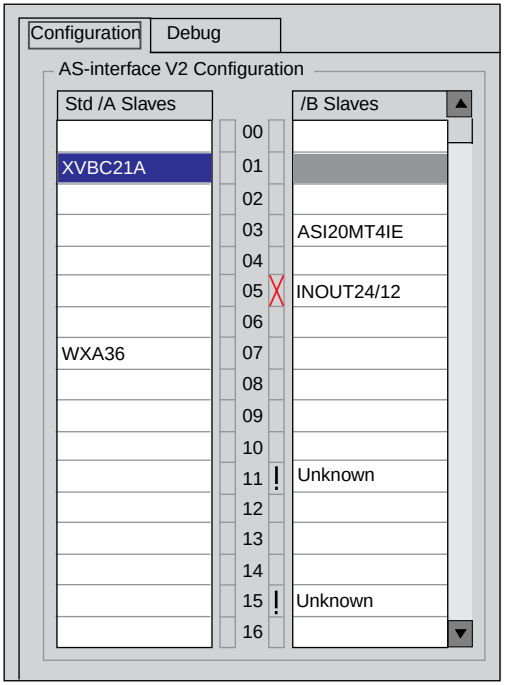
传递从设备映像到配置屏幕

当配置中未指定的从设备被总线检测到时，调试屏幕的 “AS-I V2 配置区域” 中该检测地址将出现一个 “未知” 从设备。
下表描述了传递 “未知” 从设备映像到配置屏幕的过程：

步骤	描述
1	进入 “Debug” 屏幕。
2	在 “AS-I V2 配置” 区域选择期望的从设备。
3	右击鼠标选择 “传输配置”。 图例：  **结果：** 被选从设备的映像 （表和参数的映像）被传递到配置屏幕。
4	对每个您想传递其映像到配置屏幕的从设备重复上述操作。 .

返回到配置屏幕

当用户返回到配置屏幕，所有被传递的新的从设备 （没被指定）均可见。
传递所有从设备后的配置屏幕图例：



注释：

- 叉号表示被传递的从设备的表映像和配置屏幕开始期望的表有差别。
- 感叹号表示配置屏幕中加入了新表。

解释：

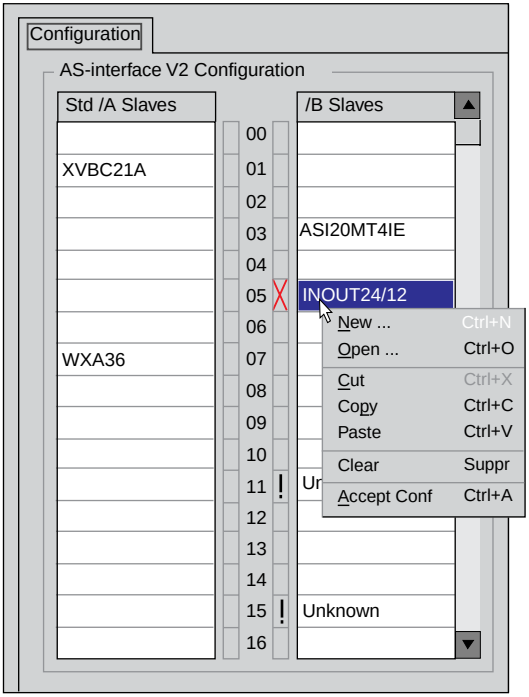
配置屏幕一般显示由总线现有映像完成的期望配置的永久映像，这是尽管地址发生了改变但却仍然存在 3B 的原因 （见从地址的修改， p. 240）。

期望的从设备的表和参数显示与期望值一致。未知从设备的表和参数显示与被检测到的映像一致。

传递最终应用程序到模块的过程

在传递新的应用程序到模块之前，用户对于每个从设备可以接受检测到的表和参数（被传递到配置屏幕），或修改配置，以手动方式 （见*声明和配置从设备过程，p. 231*）。

下表描述了确认和传递最终配置到模块的步骤：

步骤	动作
1	通过软件，断开 PC 和模块连接。 **注意：** PC 与模块相连时配置屏幕不能实现修改。
2	右击期望的从设备。
3	2 个选择： ● 选择 “接受配置” 接受被检测到的表，此表属于被选从设备。 图例：  对每个标记叉号的从设备，将有一个消息警告用户此操作将覆盖从设备的初始表 （显示在屏幕上）。 ● 在右击菜单上选择其它选项手动配置被选从设备。

步骤	动作
4	对每个配置中的期望从设备重复上述操作。
5	点击 “OK” 按钮确认和创建新的应用程序。 **结果：**自动返回到主屏幕。
6	传输应用程序到模块。

AS-I V2 从设备自动寻址

概览

每个 AS-I 总线从设备必须分配 （通过配置）一个唯一的物理地址。它必须与 TwidoSoft 声明的一样。

TwidoSoft 软件提供了从设备自动寻址功能，因此没有必要使用 AS-I 控制台。自动寻址功能用于：

- 更换故障从设备，
- 插入新的从设备。

过程

下表显示了如何设置自动寻址参数。

步骤	动作
1	进入 AS-I V2 主模块配置屏幕。
2	点击自动寻址检查框，它位于主模式区域。 **结果：**此自动寻址功能被激活 （框被选中）或停止 （框没被选中）。 **注意：**默认方式下，自动寻址参数在配置屏幕中已被选择。

怎样在现有 AS-I V2 配置中插入从设备

概览

可以不用手持编程器而插入一个设备到 AS-I V2 配置中。
此操作可以进行，一旦：

- 自动寻址配置模式被激活 （见 *AS-I V2 从设备自动寻址， p. 247*），
- 物理配置中不存在单一的从设备，
- 将要插入的从设备已在屏幕配置中指定，
- 从设备具有配置期望的表，
- 从设备地址为 0 （A）。

AS-Interface V2 模块将自动赋给从设备配置预置值。

过程

下表显示了自动插入一个新的有效从设备的过程。

步骤	动作
1	本地模式下在配置屏幕中加入新的从设备。
2	连接模式下完成配置传输给 PLC。
3	物理连接地址为 0 （A）的新的从设备到 AS-I V2 总线。

注意：如果需要，应用程序可被上面操作执行修改多次。

故障 AS-I V2 从设备的自动替换

原理

当一个从设备被声明出错，它能被一个相同类型的从设备自动替代。
这个过程不需要 AS-Interface V2 总线停止，也不需要任何操作，如果配置模式的自动寻址功能被激活 （见 *AS-I V2 从设备自动寻址，p. 247*）。

两个选择可用：

- 使用手持编程器对替代从设备编程，与故障从设备地址相同，表和子表也相同。这样自动插入到被检测从设备列表 （LDS）和活动从设备列表 （LAS），
- 替代从设备空白 （地址 0 （A），新的从设备）且和故障从设备表相同。替换从设备的地址将自动设定，然后插入到被检测从设备列表 （LDS）和活动从设备列表 （LAS）。

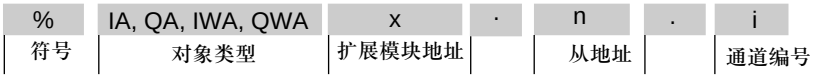
连接 AS-I V2 总线的从设备的相关 I/O 寻址

概览

这页介绍了有关从设备数字或模拟 I/O 寻址的详细资料。
为了避免远程 I/O 的混乱，可用新的符号，并采用 AS-I 语法：%IA 特此举例。

图解

寻址原理表示：



详细值

下表给出了 AS-I V2 从设备对象的详细值：

条目	值	注释
IA	-	从设备物理数字输入映像。
QA	-	从设备物理数字输出映像。
IWA	-	从设备物理模拟输入映像。
QWA	-	从设备物理模拟输出映像。
x	1 到 7	扩展总线 AS-I 模块地址。
n	0A 到 31B	位置 0 不能被配置。
i	0 到 3	-

举例

下表是一些 I/O 寻址示例：

I/O 对象	**描述**
%IWA4.1A.0	扩展总线上位置为 4 的 AS-I 模块的地址 1A 的从设备的 0 号模拟输入。
%QA2.5B.1	扩展总线上位置为 2 的 AS-I 模块的地址 5B 的从设备的 1 号数字输出。
%IA1.12A.2	扩展总线上位置为 1 的 AS-I 模块的地址 12A 的从设备的 2 号数字输入。

后台交换

下面描述的对象在后台交换，换句话说，它们在每个 PLC 循环被自动交换。

AS-I V2 总线编程和诊断

显式交换

AS-I 总线的相关对象 （字和位）为实现 AS-I 功能的高级编程贡献数据 （例如：总线操作，从设备状态，等等）和其它命令。
这些对象通过扩展总线在 Twido 控制器和 AS-I 主模块之间直接交换：

- 程序用户用指令的方式请求：ASI_CMD （见下面指令 “ASI_CMD 介绍”）
- 通过调试屏幕或活动表。

保留的特殊系统字

Twido 控制器为主模块保留的系统字使您判断网络的状态：%SW73 保留给第一个 AS-I 扩展模块， %SW74 保留给第二个。这些字只有前 5 位可用；它们只读。
下表显示了可使用的位：

系统字	位	描述
%SW73 和 %SW74	0	系统状态 （= 1 如果配置正确，否则为 0）
	1	数据交换 （= 1 可以数据交换， 0 如果处于数据交换关闭模式见 *AS-I V2 总线接口模块工作模式*， *p. 257*）。
	2	系统被停止 （= 1 如果离线模式可用 （见 *离线模式*， *p. 257*）否则 0）
	3	ASI_CMD 指令被终止 （=1 如果被终止， 0 如果进行中）
	4	ASI_CMD 错误指令 （=1 如果指令出错，否则 0）

使用示例 （对第一个 AS-I 扩展模块）：
在使用 ASI_CMD 指令之前，必须检查位 %SW73:X3 以示指令是否不处于进行状态：确认 %SW73:X3 = 1。
为了确定指令是否已被正确执行，确认位 %SW73:X4 等于 0。

ASI_CMD 指令介绍

对每个用户程序，ASI_CMD 指令允许用户对其网络编程并获得从设备诊断。指令参数的传递通过内部字 （存储字） **%MWx**。

指令语法如下：

ASI_CMDn %MWx:l

标注：

符号	描述
n	AS-I 扩展模块地址 （1 到 7）。
x	参数传递的第一个内部字 （存储字）的编号 （0 到 254）。
l	按字数计指令长度 （2）。

ASI_CMD 指令使用

下表描述了必要时对应参数 %MW(x) 和 %MW(x+1) 值的 ASI_CMD 指令动作。对从设备诊断请求，结果通过 %MW(x+1) 返回。

%MWx	%MWx+1	动作
1	0	退出离线模式。
1	1	转换到离线模式。
2	0	禁止主模块及其从设备间数据交换（进入数据交换关闭模式）。
2	1	允许主模块及其从设备间数据交换（退出数据交换关闭模式）。
3	保留	-
4	结果	读活动从设备列表 （LAS 表），地址从 0A 到 15A （每个从设备 1 位）。
5	结果	读活动从设备列表 （LAS 表），地址从 16A 到 31A （每个从设备 1 位）。
6	结果	读活动从设备列表 （LAS 表），地址从 0B 到 15B （每个从设备 1 位）。
7	结果	读活动从设备列表 （LAS 表），地址从 16B 到 31B （每个从设备 1 位）。
8	结果	读被检测从设备列表 （LDS 表），地址从 0A 到 15A （每个从设备 1 位）。
9	结果	读被检测从设备列表 （LDS 表），地址从 16A 到 31A （每个从设备 1 位）。
10	结果	读被检测从设备列表 （LDS 表），地址从 0B 到 15B （每个从设备 1 位）。
11	结果	读被检测从设备列表 （LDS 表），地址从 16B 到 31B （每个从设备 1 位）。
12	结果	读从设备外围故障列表 （LPF 表），地址从 0A 到 15A （每个从设备 1 位）。
13	结果	读从设备外围故障列表 （LPF 表），地址从 16A 到 31A （每个从设备 1 位）。
14	结果	读从设备外围故障列表 （LPF 表），地址从 0B 到 15B （每个从设备 1 位）。
15	结果	读从设备外围故障列表 （LPF 表），地址从 16B 到 31B （每个从设备 1 位）。
16	结果	读总线状态。 见下节详细结果。

注意：总线状态在每个 PLC 扫描时被更新。但是 ASI_CMD 总线读取指令的结果仅在下一个 PLC 扫描结束时才可得到。

ASI_CMD 指令读取总线状态的详细结果

ASI_CMD 指令读取总线状态 （参数 %MWx 值等于 16）时，字 %MWx+1 的结果格式如下 ：

%MWx+1		**表示 （1= 是， 0= 不是）**
低字节	第 0 位	配置正确
	第 1 位	LDS.0 （地址 0 从设备存在）
	第 2 位	自动寻址激活
	第 3 位	自动寻址可用
	第 4 位	配置激活
	第 5 位	正常操作激活
	第 6 位	APF （电源供应问题）
	第 7 位	离线准备
高字节	第 0 位	外围故障
	第 1 位	数据交换激活
	第 2 位	离线模式
	第 3 位	正常模式 （1）
	第 4 位	与 AS-I 主模块通信故障
	第 5 位	ASI_CMD 指令进行中
	第 6 位	ASI_CMD 指令错误

ASI_CMD 指令读取从设备状态的详细结果

ASI_CMD 指令诊断从设备 （%MWx 值在 4 和 15 之间）时，从设备状态通过字 %MWx+1 的位 （1= 正常）来返回。下表给出了对应字 %MWx 值的详细结果：

%MWx	%MWx+1															
值	高字节								低字节							
	第 7 位	第 6 位	第 5 位	第 4 位	第 3 位	第 2 位	第 1 位	第 0 位	第 7 位	第 6 位	第 5 位	第 4 位	第 3 位	第 2 位	第 1 位	第 0 位
4, 8, 12	15A	14A	13A	12A	11A	10A	9A	8A	7A	6A	5A	4A	3A	2A	1A	0A
5, 9, 13	31A	30A	29A	28A	27A	26A	25A	24A	23A	22A	21A	20A	19A	18A	17A	16A
6, 10, 14	15B	14B	13B	12B	11B	10B	9B	8B	7B	6B	5B	4B	3B	2B	1B	0B
7, 11, 15	31B	30B	29B	28B	27B	26B	25B	24B	23B	22B	21B	20B	19B	18B	17B	16B

为了读取从设备 20B 是否处于活动状态，必须执行 ASI_CMD 指令，内部字 %MWx 值为 7。结果通过内部字 %MWx+1 返回；从设备 20B 的状态由低字节的第 4 位的值给定：如果第 4 位等于 1，从设备 20B 处于活动状态。

ASI_CMD 指令编程示例

强制 AS-I 主模块 （扩展总线上位置为 1）变换到离线模式：

```
LD 1
[%MW0 := 16#0001 ]
[%MW1 := 16#0001 ]
LD %SW73:X3          // 如果没有 ASI_CMD 指令在进行中，那么继续
[ASI_CMD1 %MW0:2]    // 强制转换到离线模式
```

读取从设备活动表，地址 0A 到 15A：

```
LD 1
[%MW0 := 16#0004 ]
[%MW1 := 16#0000          //optional]
LD %SW73:X3  // 如果没有 ASI_CMD 指令在进行中，那么继续
[ASI_CMD1 %MW0:2]    // 读 LAS 表，地址 0A 到 15A
```

AS-I V2 总线接口模块工作模式：

概览

AS-I 总线接口模块 TWDNOI10M3 有三种工作模式，每种对应着特殊的需求。这些模式是：

- 保护模式
- 离线模式，
- 数据交换关闭模式。

使用 ASI_CMD（见 *ASI_CMD 指令介绍，p. 253*）指令可以使用户程序进入或退出这些模式。

保护模式

受保护的工作模式是应用程序运行的一般使用模式。假定 AS-I V2 模块在 TwidoSoft 中配置。它：

- 不断地检查被检测从设备列表与期望从设备列表相同，
- 监控电源供应。

这个模式下，从设备只有在配置中被声明和被检测到后才能被激活。
上电或配置阶段， Twido 控制器 AS-I 强制模块进入保护模式。

离线模式

当模块进入离线模式，它首先将所有存在的从设备复位到零且停止总线上的交换。离线模式下，输出被强制到零。
除了使用 TWDNOI10M3 AS-I 模块的 PB2 按钮，还可以通过软件使用 ASI_CMD 指令 （见 *ASI_CMD 指令编程示例，p. 256*）进入离线模式，这种方法也可以退出该模式并返回到保护模式。

数据交换关闭模式

当数据交换关闭模式运行时，总线交换继续进行，但数据不再被更新。
要进入该模式只能使用 ASI_CMD 指令 （见 *ASI_CMD 指令使用，p. 254*）。

安装和配置 CANopen 现场总线

10

概览

本章的主题

本章描述了如何在 CANopen 现场总线上安装和配置 TWDNCO1M CANopen 主模块和和它的从设备。

本章包含了哪些内容?

本章包含了以下几节：

10.1 CANopen 现场总线概述

概览

本节主题

本节主要为你提供了关于 CANopen 现场总线常识并且介绍了本章其他部分将要用到的 CAN 技术术语。

本节包含了哪些内容?

本节包含了以下主题：

CANopen 基础知识

介绍	通过解释技术术语，以对 CANopen 网络通信基础知识能更好地理解。
EDS 文件	**EDS （Electronic Data Sheet）** EDS 文件描述了 CAN 网络上设备通信属性 （波特率，输出类型， I/O 提供 ...）。它由设备制造商提供。用于在配置工具中配置节点 （就好像操作系统里的驱动程序）。
PDO	**PDO （过程数据对象）** CANopen 帧包含 I/O 数据。 区分在： ● 传送 -PDO （TPDO 带有节点提供的数据）和 ● 接收 -PDO （RPDO 节点消耗的数据）。 传送方向从节点的观点看总是可见的。 PDO 没有必要包含所有的节点数据映象（对于 TPDO 和 RPDO）。通常，模拟量输入数据和数字量输入数据被分成不同的 TPDO。对输出也同样如此。
SDO	**SDO （服务数据对象）** CANopen 帧包含参数。 SDO 主要用来在应用程序运行时，从驱动器读参数或写参数到驱动器。
COB-ID	**COB-ID （通信对象标示符）** 每个 CANopen 帧以一个 COB-ID 开头， COB-ID 作为 CAN 帧的标示码。 在配置阶段，每个节点在接收 COB-ID 时，对帧来说，他是供给者或者消费者。

关于 CANopen

介绍

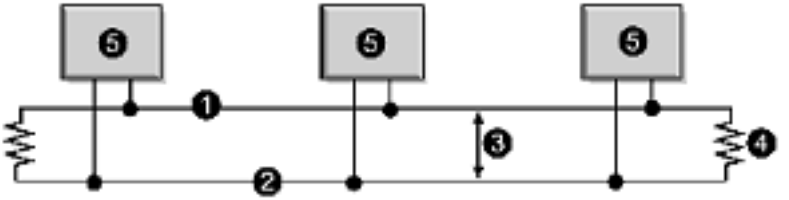
CANopen 是一个标准的工业控制系统现场总线协议。它特别适合实时控制的 PLC，因为它为集成的和可传输的工业应用提供了一个高效，低成本的解决方案。

CANopen 协议

CANopen 协议是被建立在 CAL 协议基础上的一个子协议。通过定义设备规范，它甚至更加适合用于标准工业组件。 CANopen 是一个 CiA （CAN in Automation）标准，投入市场之初，它就迅速被推广。在欧洲， CANopen 现在已经被公认为基于 CAN 设计的工业系统的工业标准。

物理层

CAN 使用差分驱动两线总线 （公共回路）。 CAN 信号是 CAN-high 和 CAN-low 线之间的电压差值。（见下图）
下图显示了两线 CAN 总线物理层组件：



1 CAN-high 线

2 CAN-low 线

3 CAN-high/CAN-low 信号间的电位差

4 120Ω 终端电阻

5 节点

根据电磁兼容要求，总线可以使用平行，绞合或者屏蔽方式走线。单线结构使反射最小化。

CANopen 规范文件

通信规范文件
CANopen 规范文件是基于 “通信规范文件”的，用来规定主要通信机制和他们的描述 （DS301）。
设备规范文件
在设备规范文件中描述了工业自动化领域最重要的设备类型，同时也定义了设备功能性质。
标准设备描述例子为：
- 数字量和模拟量输入 / 输出模块 （DS401），
- 电机 （DS402），
- 控制设备 （DSP403），
- 闭环控制器 （DSP404），
- PLC （DS405），
- 编码器 （DS406）。

通过 CAN 总线配置设备

通过 CAN 总线配置设备的可能性是制造商要求自治的基本原则之一 （对每个规范文件系列）。

CANopen规范文件的通用规格

CANopen 是符合以下规格文件的，用于 CAN 系统的一套规范：
- 开放的总线系统，
- 无协议超负荷的实时数据交换，
- 可重新定义尺寸的模块化设计，
- 设备互用性和可交换性，
- 被大量的国际制造商支持，
- 标准的网络配置，
- 访问所有的设备参数，
- 同步和循环过程数据与 / 或事件驱动数据 （可能缩短系统响应时间）。

CANopen 产品认证

在市场上提供 CANopen 认证产品的所有制造商都是 CiA 组织成员。作为一个 CiA 组织的活跃成员，施耐德电气工业 SAS 开发了与该组织标准建议兼容的产品。

CAN 标准

CANopen 协议被 CiA 组织定义并且可访问该组织 （需遵守某些限制）站点 http://www.can-cia.com。对主和从设备的源代码可以从不同的供应商得到。

注意：要了解更多关于 CANopen 标准规范和机制，请访问 CiA 主页 （http://www.can-cia.de/）。

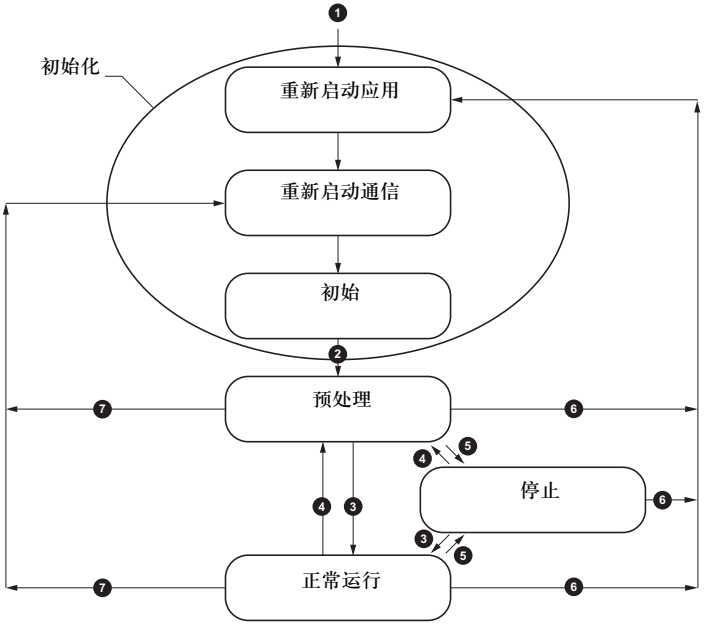
与 CANopen 网络通信

通信规范文件基于 CAL 服务和协议。
它提供用户对两种交换类型的访问：SDO 和 PDO。
在上电时，设备进入一个初始化阶段，接着进入预处理状态。在这个阶段，只有 SDO 通信可以运行。在接收到一个启动命令后，设备切换到工作状态。 PDO 交换开始可以被使用，并且 SDO 通信仍然有效。

CANOpen 启动 （boot-up）

启动过程

最小设备配置指定了简化的启动程序。这个过程用下图来描述：



图例

编号	描述
1	模块上电
2	初始化后，模块自动进入预处理状态。
3	NMT 服务指示：启动远程节点
4	NMT 服务指示：预处理
5	NMT 服务指示：停止远程节点
6	NMT 服务指示：重启节点
7	NMT 服务指示：重新启动通信

通过状态机激活 CANopen 对象

下表给出了状态机的状态与 CANopen 通信对象之间的关系。 X 表示 CANopen 通讯对象可用。

	初始化	预处理	操作	被停止
PDO 对象			X	
SDO 对象		X	X	
紧急		X	X	
启动	X		X	
NMT		X	X	X

复位应用

设备进入 “复位应用”状态：
- 设备启动后，
- 或者通过使用 “复位节点”网络管理 （NMT）服务。

在这种状态，设备规范被初始化，并且所有的设备规范信息被复位到默认值。初始化完成后，设备自动进入 “复位通信”状态。

复位通信

设备进入 “复位通信”状态：
- 在 “复位应用”状态后，
- 或者通过使用 “复位通信”网络管理 （NMT）服务。

在这种状态，所有被支持的通信对象 （与设备标识符相关的对象，例如设备类型，心跳检测，等等。 1000H - 1FFFH）的参数 （标准值，决定于设备配置）被存入对象目录里。接着设备自动进入 “初始”状态。

初始

进入 “重新启动通信”后，设备进入 “初始”模式。
该状态允许你：
- 定义要求的通信对象 （SDO， PDO，紧急事件处理），
- 安装相关的 CAL 服务
- 配置 CAN- 控制器。

设备初始化完成并且设备自动进入 “预处理”状态。

注意：TWDNCO1M CANopen 主模块不支持同步 （SYNC）模式。

预处理

设备进入 “预处理”状态：
- 在 “初始”状态后，
- 如果在正常工作模式，接收 “进入预处理” NMT 命令。

在这种状态，设备的配置可以被修改。然而，只有 SDO 可用于读或写设备相关参数。
当配置完成后，通过接收相关的命令，设备进入以下状态的其中之一：
- “停止”，当接收到 “停止远程节点” NMT 命令，
- “运行”，当接收到 “启动远程节点” NMT 命令。

停止

如果设备处于 “预处理”或 “正常工作”状态，接收到 “节点停止”命令 （NMT 服务），设备进入 “停止”状态。
在这种状态，设备不能被配置。不能读和写设备相关参数 （SDO）。只有从设备监视功能 （“节点保护”）保持有效。

操作

当设备在 “预处理状态”接收到 “启动远程节点”指令，设备进入 “操作状态”。在 “运行”状态，当使用 “节点启动” NMT 服务启动 CANopen 网络，所有的设备功能性被使用。可以使用 PDO 或 SDO 进行通信。

注意：在 “运行”模式对配置的修改可能会发生不可预测的结果，所以应该只在 “预处理”模式下进行设备的配置。

过程数据对象 （PDO）传送

PDO 定义

PDO 是与过程数据通信相关的通信对象，它能够保证过程数据的实时交换。一个 CANOpen 设备的 PDO 对象定义了该设备与网络上其他 CANopen 设备之间的隐式数据交换。
当设备在 “运行”模式， PDO 交换被激活。

PDO 类型

有两种 PDO 类型：

- PDO 被设备传送 （通常标示为：传送 PDO 或者 Tx-PDO 或者 TPDO），
- PDO 被设备接收 （经常标示为：接收 PDO 或者 Rx-PDO 或者 RPDO）。

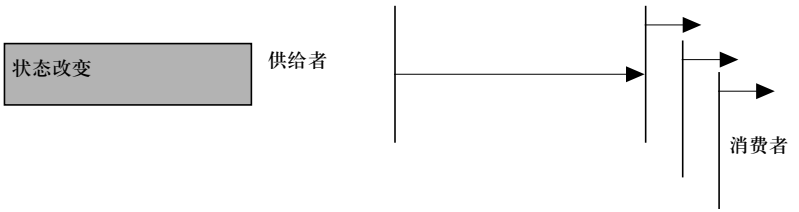
PDO 供给者和消费者

PDO 是基于 “供给者 / 消费者”模型。传送的叫 “供给者”，而接收的叫 “消费者”。
所以，写一个输出到 TWDNCO1M 主模块等于发送一个与主模块关联的包含被更新输出值的 TPDO。在这种情况下，主模块就是 PDO “供给者”（而从设备就是 PDO “消费者”）。
相反，一个输入被主模块传送的 RPDO 更新，主模块叫 “消费者”。

PDO 传送模式

除了数据被传送外，也可以配置每个 PDO 的传送模式。
主模块可配置 PDO 在以下几种传送模式下进行通信：

模式号	模式类型	模式名字
254 或 255	异步	状态改变



状态的改变 （模式 254 和 255）

“状态改变”对应于输入值的改变 （事件控制）。状态改变后，数据被立即送上总线。事件控制可能会更好的使用总线的带宽，因为只有改变被传送，而不是整个过程映象。这可能达到一个非常短的响应时间，因为当输入值被修改后，没有必要等到下一个来自主设备的请求。

当选择 “状态改变” PDO 传送时，你应该记住可能同时会有许多事情发生，等待低权限被传送时总线会产生延时。你也应该避免对高权限输入的连续修改，否则会导致总线堵塞 （这是知名的 “babbling idiot”）。

注意：作为一个通用规则，如果 Delta 模式 （对象 6426H）或者限制时间 （对象 1800H 到 1804H，子 - 索引 3）被设置以避免总线超载，你应该只为带模拟量输入的模块选择使用 PD0 传输。

通过显式交换访问数据 （SDO）

什么是 SDO?

服务数据对象 （SDO）允许使用显式请求访问数据。
当设备处于 “运行”或 “预处理”状态， SDO 服务是有效的。

SDO 类型

有两种 SDO：

- 读 SDO （下载 SDO），
- 写 SDO （上传 SDO）。

客户端 / 服务器模型

SDO 协议是基于 “客户端 / 服务器”模型。

*对于下载 **SDO***

客户端发送一个标识被读对象的请求。
服务器返回被读对象的数据。

*对于上传 **SDO***

客户端发送一个标识被写对象和要写入数据期望值的请求。
对象被更新后，服务器返回一个确认信息。

*对一个未被处理的 **SDO***

对于以上两种情况，如果一个 SDO 不能被处理，服务器返回一个错误代码（忽略代码）。

“节点保护”和“寿命保护”

使用期限的定义

“使用期限”参数是按以下来计算的：
使用期限 = 保护时间 × 使用期限因数
对象 100CH 含有以毫秒表示的 “' 保护时间”参数。对象 100DH 含有 “使用期限因数”参数。

监控激活

如果这两个参数其中一个被置为 “0”（默认配置），模块不执行监控 （无 “寿命保护”）。
为了激活监控，你必须在对象 100DH 输入一个值而且在对象 100CH 指定一个以毫秒为单位的时间。

保证可靠工作

为了保证可靠工作，建议输入 “使用期限因数”的值为 2。
如果不是这样，主模块会产生延时 （例如，由于在 “节点保护”时高优先级信息的处理或者内部处理），模块会转入 “预处理”状态而不产生错误。

监控的重要性

这两种监控机制对 CANopen 系统是特别重要的，特别对于通常不工作于事件 - 被控制模式的设备。

从设备监控

监控按以下方法执行：

阶段	描述
1	主设备设置 “远程帧”（远程传送请求）在被监控从设备的 “保护 COB-ID”。
2	从设备关心被 “保护”信息发送的响应。它包含从设备 “状态代码”和 “锁住位”，且 “锁住位”必须在每次通知后改变。
3	主设备在 “状态”和 “锁住位”信息间比较： 如果他们不在 NMT 主设备期望的状态里或者如果没有接收到响应，主设备认为从设备出错。

主设备监控

如果主设备基于严格循环请求 “保护”信息，从设备可以检测到主设备故障。
如果从设备在已定义的 “使用期限”间隔内没有接收到主设备的请求 （保护错误），他会认为主设备故障 （“看门狗”功能）。
在这种情况，相应输出进入出错状态并且从设备切换回 “预处理”模式。

注意：即使没有值输入 “保护时间”和 “时间寿命因数”对象，从主设备的 “远程”请求也会得到响应。时间监视只有当两个对象的值大于 0 时才会被激活。“保护时间”参数典型值在 250 毫秒与 2 秒之间。

“保护”协议

在第一个 “保护”信息里，“锁住位” (t) 的值是 “0”。
然后，在每个后来的 “保护”信息里该位会改变 （“锁住”），这样可以显示信息是否丢失。
总线头部用 7 个剩余的位来指示网络状态：

网络状态	响应
被停止	0 × 04 或 0 × 84
预处理	0 × 7F 或 0 × FF
操作	0 × 05 或 0 × 85

内部总线管理

把内部总线切换到“停止”状态	当通信模块从 “预处理”切换到 “处理”状态，内部总线自动从 “停止”切换到“运行”状态。 当内部总线切换到 “停止”状态，所有的扩展模块输出被置 0。 通信模块输出保留在他们的当前状态。
配置扩展模块	内部总线用于更新离散量和模拟量扩展模块参数的配置。 当总线处于 “停止”状态，参数被发送到通信模块。 当总线进入 “运行”状态，这些新的配置参数被确认。

10.2 实现 CANopen 总线

概览

介绍

本节描述了如何在 Twido PLC 系统上使用 TWDNCO1M CANopen 主模块实现 CANopen 现场总线。

本节包含了哪些内容?

本节包含了以下主题：

总的介绍

硬件和软件要求

以下被要求的硬件和软件用来在你的 Twido PLC 系统上实现 CANopen 总线：

硬件	要求
Twido PLC 一体型或模块型本体控制器	**一体型本体：** ● TWDLC•24DRF ● TWDLCA•40DRF **模块型本体：** ● TWDLMDA20••• ● TWDLMDA40•••
CANopen 主站	1 个 CANopen 主模块：TWDNCO1M
CANopen 从设备	最大 16 个
CANopen 连接器和电缆	
Twido PLC 编程电缆	

软件	要求
Twido PLC 配置软件	TwidoSoft V3.0 或更高

CANopen 实现步骤

以下步骤将引导你安装，配置和 CANopen 网络的使用：

步骤	描述
1	硬件安装
2	配置方法
3	CANopen 主设备声明
4	网络 CANopen 从设备声明
5	CANopen 对象映射
6	CANopen 对象链接
7	CANopen 对象符号化
8	网络 CANopen 诊断
以下子节提供了每个步骤的具体描述。	

硬件安装

安装 TWDNCO1M 主模块

在 Twido PLC 系统 （DIN - 导轨或面板安装）安装 TWDNCO1M 主模块并且把它连接到 Twido PLC 内部总线 （更多细节，见 TwidoHW - 安装扩展模块），以下为步骤：

步骤	动作	描述
1	安装准备工作	参考 Twido 可编程控制器硬件参考手册（TWD USE 10AE）使用说明： ● Twido 模块的正确安装位置， ● 从 DIN 导轨增加和拆卸 Twido 组件， ● 直接固定在面板表面， ● 控制器箱内模块的最小间隙。
2	固定 TWDNCO1M 模块	安装 TWDNCO1M 主模块在 DIN 导轨或面板。更多细节，见 TwidoHW 安装一个扩展模块。
3	模块连接到 Twido PLC 的总线	连接 CANopen 主模块到 Twido PLC 内部总线 （更多细节，见 TwdoHW - 安装扩展模块）。
4	CANopen 接线和连接	按照 CANopen 接线和连接说明要点连接 CAN 总线电源和信号线。

配置方法

概览

通过 TwidoSoft V3.0 或更高的 CANopen 配置工具对 CANopen 进行配置。

注意：
1. CANopen 网络，主和从配置，以及通信参数的配置只能在离线模式下进行。
2. 在线模式不允许改变 CANopen 配置。
3. 在线模式，只有某几个参数可以调节，例如 %IWC 和 %QWC PDO 寻址参数。

配置方法

下表描述了 CANopen 总线不同软件执行阶段：

模式	阶段	描述
本地	TWDNCO1M 模块的声明	选择一个有效的槽位把 TWDNCO1M 主模块安装到 Twido 扩展总线上。
	CANopen 网络配置	CANopen 网络配置通过： ● 把所有设备的 EDS 文件导入网络目录， ● 从目录里增加从设备到 CANopen 网络。
	PDO 映射	在网络上声明每个从设备的 TPDO 和 RPDO 对象映射。
	PDO 链接	链接每个从 PDO 到相关的主模块 PDO。
本地或被连接	符号化 （可选）	从设备相关变量符号化。
	编程	对 CANopen 功能编程。
被连接	传递	传递应用程序到 PLC。
	调试	通过下面帮助调试应用程序： ● 调试屏幕，一方面用于显示从设备 （地址，参数），另一方面用于分配它们相应的地址， ● 诊断屏幕允许辨识错误。

注意：TWDNCO1M CANopen 主模块与其他任何扩展模块相同。然而，在 Twido 扩展总线上，只允许有一个 CANopen 主模块。 TwidoSoft 用户接口程序不允许增加任何其他 CANopen 模块。

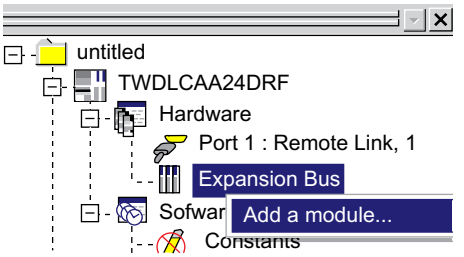
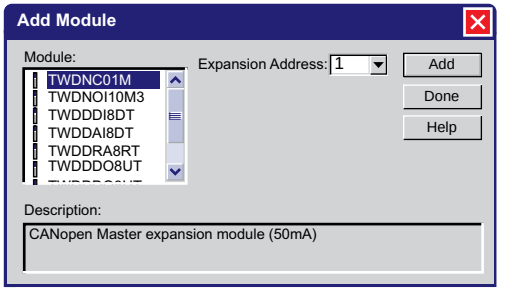
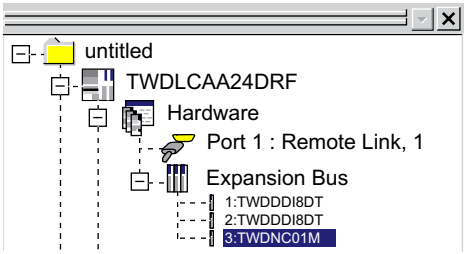
连接前的警告

在连接 （通过软件） PC 到控制器之前，并为了避免出现问题：

- 确保在总线上没有地址为 127 的从设备 （127 保留，工厂设定地址分配给 TWDNCO1M 主模块）。
- 确保在 CANopen 总线上没有重复地址的从设备被安装。

CANopen 主设备声明

步骤 下表显示了在声明 CANopen 主设备时不同的步骤。

步骤	动作	注释
1	从 TwidoSoft 应用浏览器里，右击扩展总线→增加一个模块…	
2	当增加一个模块对话框出现： ● 选择 **TWDNCO1M**。 ● 点击增加。 ● 在这个阶段，你可以继续增加任何其他你想包含进你的 Twido 系统的扩展模块（到 7 个）。 注意：只允许有一个 TWDNCO1M CANopen 主模块。 ● 点击完成。	只有 TWDLC?A24DRF， TWDLCA?40DRF，TWDLMDA20??? 和 TWDLMDA40??? 控制器支持 
3	一个扩展总线结构出现（如右例）注意：TWDNCO1M 主模块可以插入 Twido 总线上从 1 到 7 任何有效的扩展槽。	

网络 CANopen 从设备声明

概览

网络 CANopen 设备声明有三个步骤，包括：

1. 把 CANopen 从设备 EDS 文件导入 Twido CANopen 配置器的目录，
2. 通过从目录增加多达 16 个从设备到网络来建立 CANopen 网络，
3. 配置网络管理参数 （网络速度和错误控制协议参数）。

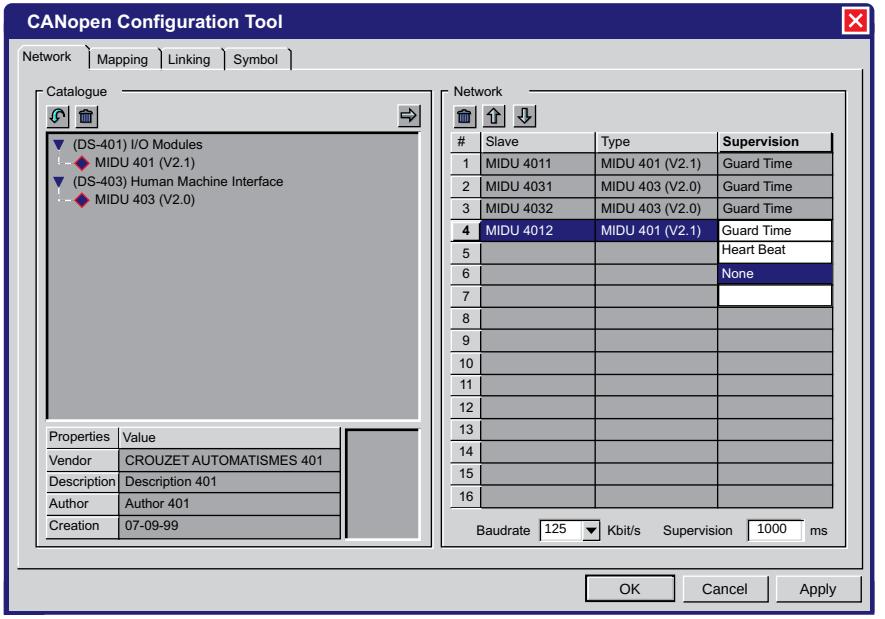
CANopen 配置器

在 TwidoSoft 应用浏览器，右击主模块名并选择硬件→扩展总线→ **TWDNCO1M** → 配置

结果：CANopen 配置工具在屏幕上出现，如下节所示。

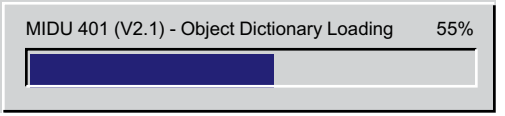
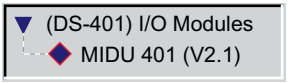
网络对话框

通过 TwidoSoft CANopen 配置器进行网络 CANopen 配置和从设备声明网络对话框，如下所示：



导入从设备规范表

下表描述了如何导入 CANopen 从设备规范表 （.EDS 文件）到 CANopen 配置工具目录：

步骤	动作
1	从目录区域，点击导入 icon  。. **结果：**操作系统打开的对话框出现。
2	找到你想加到目录里的 CANopen 从设备 EDS 文件所在文件夹的位置。 **结果：**可用的 EDS 文件出现在打开的对话框中：
3	选择一个 EDS 文件 （“文件名” **.EDS**）点击打开。 **结果：**CANopen 配置工具装载所选设备的对象目录。 **注意：**注意这个过程可能会花费几分钟时间，时间长短取决于所选 EDS 文件的尺寸。一个进度条显示装载过程完成的状态，如下例子所示： 
4	等到装载过程完成，然后重复步骤 2 到 3 以装入你想增加到目录的新设备规定文件。 **注意：**你只需执行一次这个过程，所有列在装载目录里的设备规定文件和对象字典就会被 TwidoSoft 存储。
5	为了显示 CANopen 从设备的设备属性： **1.** 双击列在目录里的设备类型。 **示例：**  . **结果：**  . **2.** 点击子站表 （举例， MIDU 401 V2.1）。 **结果：**所选 CANopen 从设备的设备属性出现在目录区的较低部分，见： ● 供应商的名字 （例如， Crouzet Automatismes 401）， ● 从设备规范文件 （例如，描述 401）， ● 作者名字 （例如， Author 401）， ● 所有规定范件的创建日期 （例如， 07-09-99）。

步骤	动作
6	要从目录里删除从设备规定文件，在目录窗选择设备名称并点击删除图标  。 **注意：**你可以在网络 CANopen 目录里存储比你实际需要还要多的规范文件。已经装入目录里的规范文件留作将来使用。
7	按应用按钮以确认对目录的改变并把从设备规定文件存入 TwidoSoft 项目。

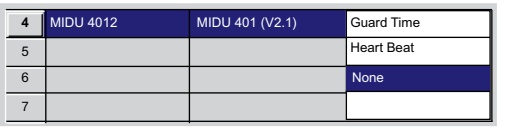
建立 CANopen 网络

下表描述了如何在 Twido CANopen 网络声明从设备。（注意你仅可以声明 EDS 规定文件已经加入或已经预先存入目录里的设备）。

步骤	动作
1	从目录区，从被存入目录的可用设备列表选择从设备规范文件。 **结果：**此增加图标  出现在目录框的右上角。
2	单击增加图标  一次。 **结果：**从设备被加入网络从设备表。 **注意：** ● 在 Twido CANopen 网络上最多可以声明 16 个从设备。 ● 新声明的从设备占用最低可用索引节点地址。（例如，如果从设备已经声明为节点地址 1， 2 和 4，新增加的从设备占用节点地址 3，默认。）
3	你可以分配给一个从设备任何一个可用节点地址 （1 到 16）。为了移动一个从设备到期望的节点地址，使用上移 / 下移 （**Move up/down**） 箭头图标  。
4	对你想要在 CANopen 网络上声明的任何新从设备，请重复步骤 1 到 3。
5	要从网络上删除一个从设备，在从设备表中选择设备名称并点击删除图标  。
6	按应用按钮以确认改变并把网络配置存入 TwidoSoft 项目。

配置网络管理参数

以下步骤描述了如何配置网络管理参数，例如波特率 （网络速度），使用期限和错误控制协议。）

步骤	动作
1	在网络对话框里，从下拉列表选择波特率 （网络速度）：**10, 20, 50, 100, 125**（默认值）， **250, 500, 800** 或 **1000 Kbit/s**。 **注意：**确定网络上已经被声明的每个从设备已经被分别配置，以确保他自己的波特率和以上定义的网络速度严格统一，否则 CANopen 网络通信将不能正常运行。
2	配置使用期限周期。这个参数定义了通信循环时间周期，用在每个从设备的监控区域，在以下步骤 3 解释。 **注意：**在该区域不要输入 0。
3	单击监控区域，以配置在网络从设备表中声明的每个从设备的错误控制协议选项。 **结果：**所选设备支持的可用监控选项出现在列表框内，如下例所示： 
4	在 TWDNCO1M 主模块和所选从设备之间选择你希望用于管理通信的错误控制协议： ● 保护时间 （Guard Time） ● 心跳 (Heartbeat) ● 没有 (None)
5	如果在网络从设备表里监控选项被设为没有，那么在这个从设备与 TWDNCO1M 主模块之间发生连接断开 （*），输出将不会返回他们的反馈值。 （*）连接断开可能由以下引起： ● 连接 TWDNCO1M CANopen 主模块到 Twido PLC 本体的扩展总线电缆断开 ● CANopen 从设备从 Twido CANopen 总线断开， ● 一根无效的总线电缆， ● TwidoSoft “复位” 命令 （在线→ Firmware/ 复位）， ● TwidoSoft 装载配置命令 （在线→下载）， ● 通过 TwidoSoft 把 firmware 下载到 TWDNCO1M 主模块命令 （在线→ Firmware 下载）。
6	按应用按钮以确认改变并把网络配置存入 TwidoSoft 项目。

CANopen 对象映射

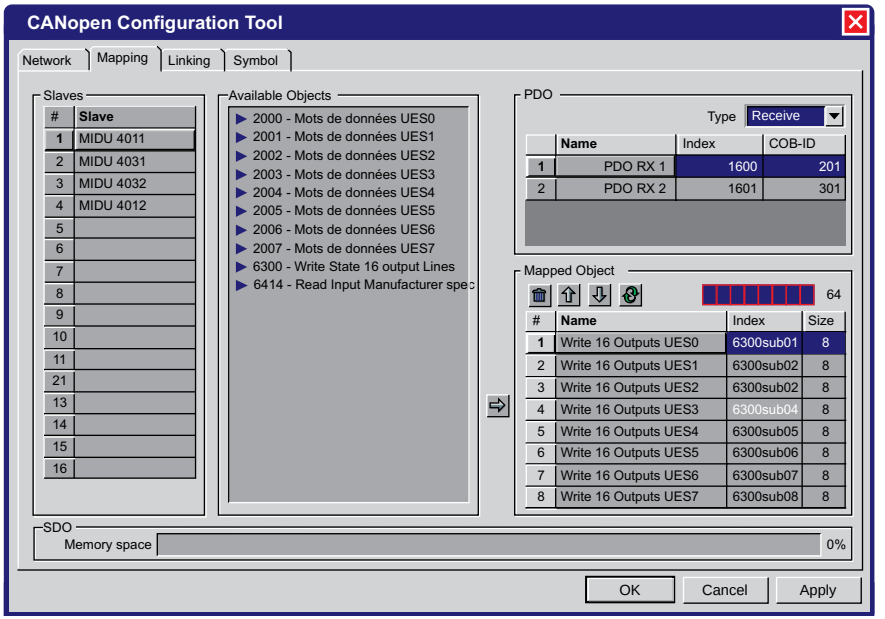
概览

此映射对话框允许你配置网络上已声明的每个从设备的 PDO。

映射对话框

在 TwidoSoft 应用浏览器，右击主模块名并选择硬件→扩展总线→ **TWDNCO1M** → 配置，然后从 CANopen 配置工具里选择映射标签。

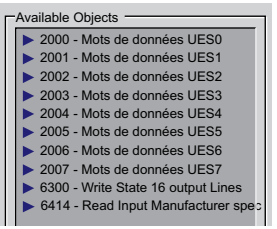
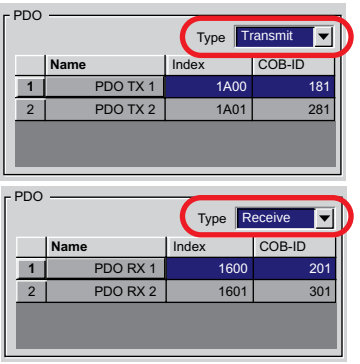
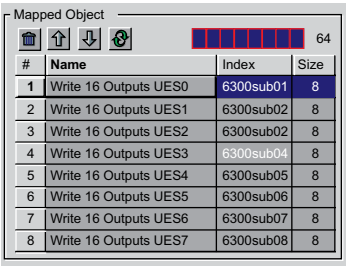
结果：屏幕上出现 CANopen 配置工具，如下图所示：



对象映射

要找出如何使用映射对话框以配置每个从设备的 TPDO 和 RPDO，按照如下步骤：

步骤	动作
1	在从设备框里，单击设备名称以选择你希望配置 PDO 的从设备。
2	**示例：**DS-401 I/O 模块标为 MIDU 4011。注意同网络配置的前几个步骤一样，从设备名称和节点地址会出现在这个框里 （见*网络 CANopen 从设备声明，p. 281*）。 

步骤	动作
3	**结果：** **1.** 所有的所选设备支持的 CANopen 对象出现在可用对象列表框，如以下例子所示：  **2.** 此 **PDO** 框显示了对所选从设备预先定义的传送 -PDO （PDO TX），作为默认值。最好还是使用类型下拉列表显示预先定义的接收 -PDO （PDO RX）。在这个例子里， MIDU 4011 DS-401 I/O 模块支持两个传送 -PDO （PDO TX）和两个接收 -PDO （PDO RX），如下所示：  **3.** 每个所选 PDO 预先定义的映射显示在映射对象框里： 

步骤	动作
4	可以使用映射对象框选择定制 PDO 映射。 RPDO 或 TPDO 是一个 64-byte 对象，能容纳高达 8 个 8-byte 字对象或者 4 个 16-byte 字对象或者这两种字对象的任何组合。不要超过全部 64-byte PDO 的限制。 要定制 PDO 映射，继续第 5。
5	对所期望的从设备 （见步 2），从 PDO 框选择你希望改变映射的 PDO。 **示例：**选择第一个传送 -PDO （PDO TX 1）。 **结果：**预先定义的 **PDO** 映射 （或者目前已经定制的映射）出现在映射对象框里。
6	要从 PDO 结构删除一个未使用的字对象，选择字对象 （索引 1 到 8）并点击删除图标  。
7	从可用对象选择你希望的字对象，并点击增加图标  以添加字对象到映射对象结构。 **注意：**对所选的 PDO，恢复默认的映射结构，点击默认图标  。.
8	要在映射的 PDO 结构里改变一个字对象地址，使用上移 / 下移箭头  /  。.
9	按应用按钮以确认对映射 PDO 结构的改变并把 PDO 映射存入 TwidoSoft 项目。
10	重复步骤 5 到 9，对每个你希望配置的 PDO 映射。
11	**注意内存使用：** ● **PDO 内存使用：** PDO 内存的使用可以通过位于映射对象框右上角的内存状态栏来监控  。 ● **SDO 附加的内存使用：** 预先定义的 PDO 和字对象不使用附加的 SDO 内存。 然而，移除和增加字对象到 PDO 映射结构需要使用附加的系统内存。 SDO 内存的当前使用在位于映射对话框底部的状态栏里：  。

CANopen 对象链接

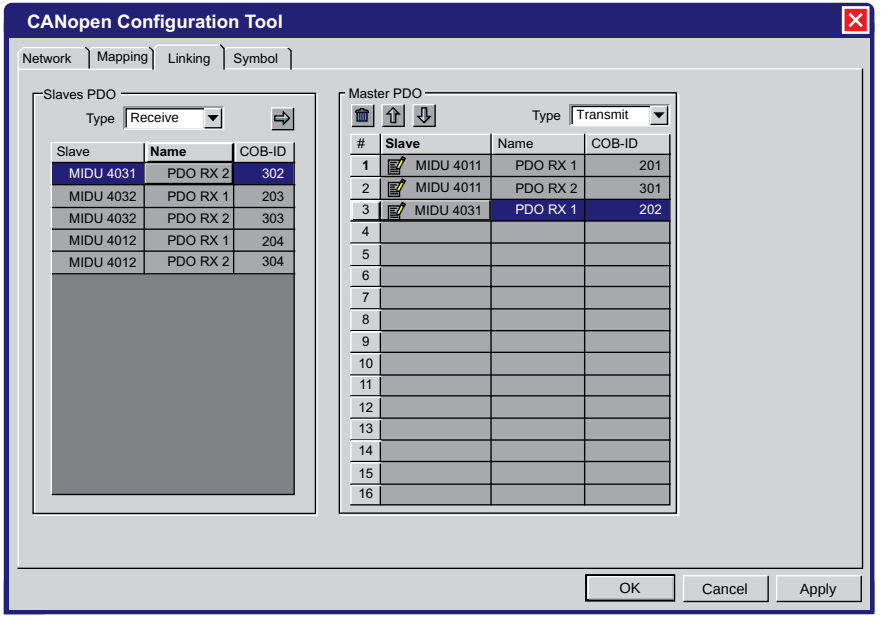
概览

此链接对话框用于定义在所选从设备 PDO 与 TWDNCO1M CANopen 主模块 PDO 间的物理链接。

链接对话框

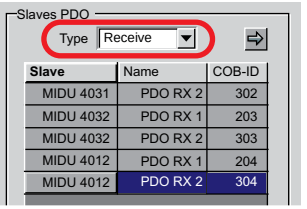
在 TwidoSoft 应用浏览器，右击主模块名并选择硬件→扩展总线→ **TWDNCO1M** → 配置，然后 CANopen 配置工具里选择链接标签。

结果：屏幕上出现 CANopen 配置工具，如下图所示：



对象链接

为了明白如何使用链接对话框在从设备与主模块 PDO 间定义物理链接，按照以下指导：

步骤	动作
1	在从 **PDO** 框，选择 PDO 类型：接收或传送。 **结果：**所有被选类型的 PDO 从设备出现在从设备 PDO 框里， 如下例所示：  **注意：**在从设备 PDO 框里选择接收或传送，会自动地把主设备 PDO 变为相反类型：传送或接收。
2	从从 **PDO** 框，选择你希望链接到 TWDNCO1M CANopen 主模块的 PDO 并点击增加图标  以添加 PDO 到主 **PDO** 链接列表。 **注意：**TWDNCO1M 主模块最多支持 16 个 TPDO 链接和 16 个 RPDO 链接。
3	要在主 PDO 框里改变 PDO 链接的地址索引，使用上下移动箭头  。
4	要删除在主 PDO 框里未使用的 PDO 链接，选择期望的 PDO （索引 1 到 16）并点击删除图标  。
5	按应用按钮以确认对映射 PDO 结构的改变并把 PDO 映射存入 TwidoSoft 项目。
6	对每个你希望链接到 CANopen 主模块的从 PDO，重复步骤 1 到 5。

CANopen 对象符号化

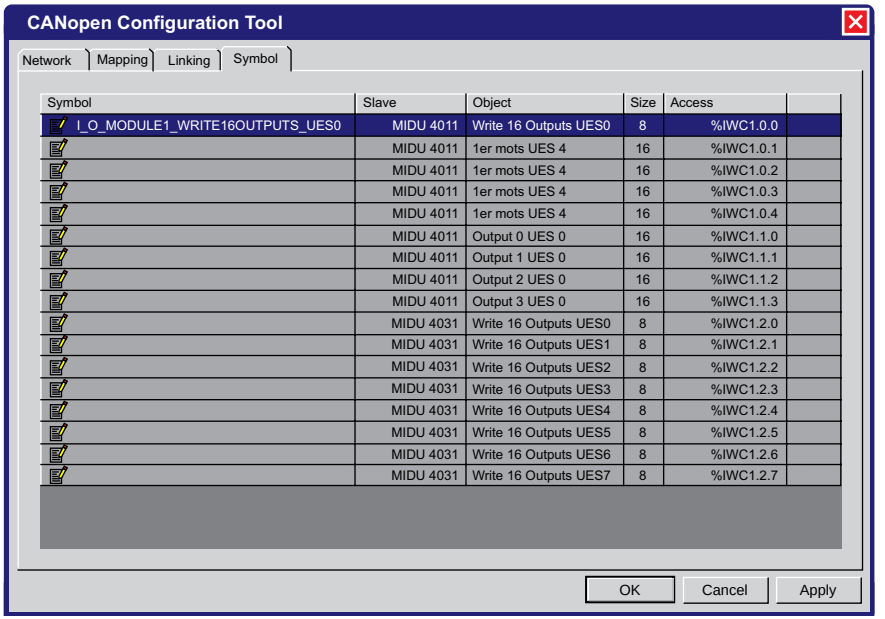
概览

此符号对话框允许你定义和 CANopen 主模块相关变量的符号。

符号对话框

在 TwidoSoft 应用浏览器，右击主模块名并选择硬件→扩展总线→ **TWDNCO1M** → 配置，然后 CANopen 配置工具里选择符号标签。

结果：屏幕上出现 CANopen 配置工具，如下图所示：



对象符号化

要使用符号框对 CANopen 对象变量定义符号，按照如下步骤：

步骤	动作
1	在符号区，双击符号编辑图标  在你所希望符号化变量的同一行。 **结果：**符号文本框被激活并且指针在文本框里右对齐。
2	填入一个描述名称。 一个有效的符号可以多达 32 字符：只允许字母 A-Z，数字 0-9 和下划线 “_”）（不允许 “/”，“%”，空格或任何其他特殊字符。） **注意：**对于编辑符号的更多细节，请参阅*符号对象*，*p. 54*。
3	按应用按钮以确定对符号表的改变并存入 TwidoSoft 项目。
4	对每个你希望的符号化的变量，重复步 1 到 3。

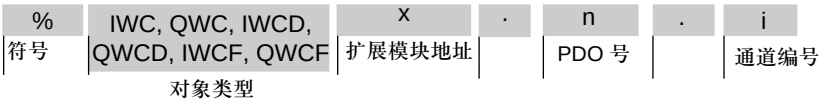
CANopen 主模块 PDO 寻址

概览

这个子节描述了 CANopen 主模块 PDO 输入和 PDO 输出的寻址。
为了避免远程 I/O 混乱，对 CANopen 对象使用新的语法：%I**WC**。举例如下。

图解

寻址规则回顾：



特殊值

下表给出了对 CANopen 从设备对象的特殊值：

条目	值	注释
IWC	-	物理 PDO 输入映像。
QWC	-	物理 PDO 输出映像。
IWCD	-	用法如 IWC，但是用双字格式。
QWCD	-	用法如 QWC，但是用双字格式。
IWCF	-	用法如 IWC，但是用浮点格式。
QWCF	-	用法如 QWC，但是用浮点格式。
x	1 到 7	TWDNCO1M CANopen 主模块的地址。
n	0 到 15	PDO 号 （根据 PDO 索引。）
i	0 到 7	通道号 （根据 PDO 子索引。）

举例

下表显示了 PDO 寻址例子：

I/O 对象	**描述**
%IWC4.1.0	PDO 号 1，子索引 0，Twido 扩展总线上位于地址 4 的 CANopen 模块输入。

隐式交换

下面描述的对象在后台交换，换句话说，它们在每个 PLC 循环被自动交换。

对 CANopen 现场总线编程和诊断

显式交换

和 CANopen 现场总线相关的对象 （字和位）使用数据 （例如：总线操作，从状态，等等。）和附加命令完成 CANopen 功能的高级编程。
这些对象通过扩展总线在 Twido 控制器与 CANopen 主模块间作显式交换：

- 用户程序请求通过指令：CAN_CMD （见下面 “CAN_CMD”指令介绍）
- 通过调试画面或动态数据表。

为 CANopen 主模块保留的特殊系统字

在 Twido 控制器里为 CANopen 保留的系统字使你能够确定网络状态：%SW8x (x=1-7) 为 CANopen 主模块保留， x 表示在 Twido 总线上的安装位置。这个字只有前 7 位可被使用；他们是只读的。
下表显示了可使用的位：

系统字	位	描述
%SW8x (x=1-7)	0	CANopen 主模块配置状态 （ = 1 如果配置好，否则为 0）
	1	CANopen 工作模式 （= 1 数据交换使能，否则为 0）
	2	系统停止 （= 1 如果离线模式使能，否则为 0）
	3	CAN_CMD 指令完成 （= 1 命令完成， = 0 命令执行中）
	4	CAN_CMD 指令错误 （= 1 指令错误， =0 指令无错误）
	5	初始化错误 （= 1）
	6	丢失信息或停电错误 （= 1）

使用例子 （对安装在 Twido 总线上扩展地址为 1 的 CANopen 主模块）：
在使用 CAN_CMD 指令前，必须检查 %SW81:X3 位以知道指令是否在执行中：检查 SW81:X3 = 1。
为了确定指令是否已经被正确执行，检查确认 %SW81:X4 位等于 0。

CANopen 从设备用特定系统字

%SW20 到 %SW27 被保留为系统字，这些字允许你知道节点地址范围从 1 到 16 的 16 个 CANopen 从设备的当前状态。这些内存字的内容为只读。

下表列出系统字 %SW20 到 %SW27：

系统字	节点地址（从设备号）		字内容 / 描述
	位 [0-7]	位 [8-15]	
%SW20	1	2	当 %SW2x 取下列值： ● = 1 => 网络上出现未预料的模块。在从系统中被删除前，它会显示 “有错误”。 ● = 2 => 节点状态操作（模块处于操作状态）： - “无错误”。 ● = 3 => 节点状态操作 （模块处于操作状态）： - “有错误”。 ● = 4 => 节点状态预操作 （模块处于预操作状态）： - 仅为预料模块 （正如配置表中所示）； - 模块可设置为操作； - “无错误”。 ● = 5 => 节点状态预操作 （模块处于预操作状态）： - 仅为预料模块 （正如配置表中所示）； - 模块可设置为操作； - “有错误”。
%SW21	3	4	
%SW22	5	6	
%SW23	7	8	
%SW24	9	10	
%SW25	11	12	
%SW26	13	14	
%SW27	15	16	

系统字	节点地址（从设备号）		字内容 / 描述
	位 [0-7]	位 [8-15]	
			● = 6 => 节点状态预操作 （模块处于预操作状态）： - 仅为预料模块 （正如配置表中所示）； - 模块存在但现有状态不可设置为操作； - “无错误”。 ● = 7 => 节点状态预操作 （模块处于预操作状态）： - 仅为预料模块 （正如配置表中所示）； - 模块存在但现有状态不可设置为操作； - “有错误”。 ● = 8 => 错误模块 （模块由不同的设备识别信息所检测）： - “无错误” ● = 9 => 错误模块 （模块由不同的设备识别信息所检测）： - “有错误”。 ● = 10 => 从设备配置错误 （模块已通过错误的确认回答 SDO 命令表中的 SDO 写请求或未按照 SDO 协议的规则）： - “无错误” ● = 11 => 从设备配置错误： - “有错误”。 ● = 12 => 丢失模块 / 错误控制超时 /SDO 超时（被配置的模块无效，或在操作过程中消失或未应答 SDO 访问）： - “无错误”。

系统字	节点地址（从设备号）		字内容 / 描述
	位 [0-7]	位 [8-15]	
			● = 13 => 丢失模块 / 错误控制超时 /SDO 超时（被配置的模块无效，或在操作过程中消失或未应答 SDO 访问）： - “有错误”。 ● = 14 => 未预料模块 （被检测的模块不在配置表中）： - “无错误”。 ● = 15 => 未预料模块 （被检测的模块不在配置表中）： - “有错误”。

CAN_CMD 指令介绍

对每个用户程序， CAN_CMD 指令允许用户对网络编程并获得从设备诊断。指令参数通过内部字 （存储字） %**MW**x 传递。

指令语法如下：

CAN_CMDn %**MW**x:l

标注：

符号	描述
n	Twido 总线上 CANopen 主模块扩展地址 （1 到 7）。
x	参数传递的第一个内部字 （存储字）的编号 （0 到 254）。
l	按字数计算的指令长度 （2）。

使用 CAN_CMD 指令

CAN_CMD 指令允许你对 CANopen 进行编程和管理并对各个从设备执行诊断检查。通过内存字 %MWx 传递命令参数。

下表描述了 CAN_CMD 指令动作，根据所需要的参数值 %MW(x) 到 %MW(x+5)：

<table>
<tr><th rowspan="2">%MWx</th><th colspan="2">%MWx+1</th><th colspan="2">%MWx+2</th><th colspan="2">%MWx+3</th><th colspan="2">%MWx+4</th><th colspan="2">%MWx+5</th><th rowspan="2">动作</th></tr>
<tr><th>位 [0-7]</th><th>位 [8-15]</th><th>位 [0-7]</th><th>位 [8-15]</th><th>位 [0-7]</th><th>位 [8-15]</th><th>位 [0-7]</th><th>位 [8-15]</th><th>位 [0-7]</th><th>位 [8-15]</th></tr>
<tr><td>1</td><td colspan="2">0</td><td colspan="8" rowspan="4">—</td><td>重置 CANopen 通信。</td></tr>
<tr><td>1</td><td colspan="2">1</td><td>重置 CANopen 节点。</td></tr>
<tr><td>2</td><td colspan="2">0</td><td>从运行模式切换到预处理模式。</td></tr>
<tr><td>2</td><td colspan="2">1</td><td>切换到运行模式。</td></tr>
<tr><td rowspan="6">3 或 4</td><td colspan="2"></td><td colspan="8" rowspan="2"></td><td>3=> 开始读 SDO 命令。
4=> 开始写 SDO 命令。</td></tr>
<tr><td colspan="2">节点</td><td>节点 =1-16=> 节点地址</td></tr>
<tr><td colspan="2"></td><td>索引</td><td colspan="7"></td><td>PDO 对象索引。</td></tr>
<tr><td colspan="2"></td><td colspan="2"></td><td>Len</td><td>Sub</td><td colspan="4"></td><td>Sub=0-255=> 对象子索引
Len= 数据子节长度</td></tr>
<tr><td colspan="2"></td><td colspan="4"></td><td colspan="2">数据 1</td><td colspan="2"></td><td>依照长度区域 （Len）值的有效负荷</td></tr>
<tr><td colspan="2"></td><td colspan="4"></td><td colspan="2"></td><td colspan="2">数据 2</td><td>依照长度区域 （Len）值的有效负荷</td></tr>
</table>

注意：在每次 PLC 扫描时总线状态被更新。然而， CAN_CMD 总线读指令的结果只有在 PLC 扫描结束后才可用。

CAN_CMD 指令编程举例

例子 1：

强制 CANopen 主 （位于 Twido 扩展总线地址为 1）切换到初始模式：

```
LD 1
[%MW0 := 16#0001 ]
[%MW1 := 16#0001 ]
LD %SW81:X3// 如无 CAN_CMD 指令在运行，则继续
[CAN_CMD1 %MW0:2]  // 强迫 CANopen 主站切换到初始模式
LD %SW81:X4// （可选的）在发送新的指令前，检查是否
CAN_CMD 已成功结束。
```

例子 2：

读以下变量：SDO_ 从 :1_ 索引 :24576_ 子索引 :1_ 长度 :4

```
LD 1
[%MW6 := %MW4]// 存储最后一个 SDO 命令结果
[%MW7 := %MW5]// 存储最后一个 SDO 命令结果
LD %SW81:X3 // 如果没有 CAN_CMD 指令在执行中，那么继续
[%MW0 := 16#0003]
[%MW1 := 16#0101]//SDO 读地址节点 1
[%MW2 := 16#6000]// 访问索引号 24576
[%MW3 := 16#0104]// 访问子索引号 1 和长度值 4
[CAN_CMD1 %MW0:6]  // 启动 SDO 命令
```

例子 3：

写以下变量：SDO_ 从 :1_ 索引 :24576_ 子索引 :1_ 长度 :4

```
LD 1
[%MW0 := 16#0004]
[%MW1 := 16#0001]//SDO 写地址节点 1
[%MW2 := 16#6000]// 访问索引号 24576
[%MW3 := 16#0104]// 访问子索引号 1 和长度值 4
[%MW4 := 16#1234]// 数据 1 值
[%MW5 := 16#1234]// 数据 2 值
LD %SW81:X3   // 如果没有 CAN_CMD 指令在执行中，那么继续
[CAN_CMD1 %MW0:6]  // 启动 SDO 命令
```

配置 TwidoPort 以太网网关

11

概览

本章主题

本章介绍 ConneXium TwidoPort 以太网网关模块的软件配置。

本章包含了哪些内容?

本章包含了以下几节：

11.1 TwidoPort 连接和配置

概览

本章主题

本章介绍如何实现 ConneXium TwidoPort 模块和 TwidoSoft 应用程序的连接，以及模块可连接性和 BootP 配置信息。

本节包含了哪些内容?

本节包含了以下主题：

TwidoSoft 配置

前言

如你有 TwidoSoft (v. 3.0 或更高)，用这些指令配置 TwidoPort：

注意：即插即用特点
TwidoPort 用 TwidoSoft 配置时， TwidoPort 的 IP 配置存储于 Twido 控制器中。因此维护时能替换 TwidoPorts 而不需要另外的配置。
要使用即插即用特点，使用 TwidoSoft 3.0 或更高的版本并升级 Twidofirmware 到 3.0 或更高。使用低版本 Twidosoft 时，使用 Telnet 手动配置 TwidoPort。

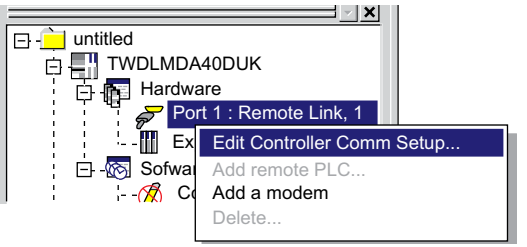
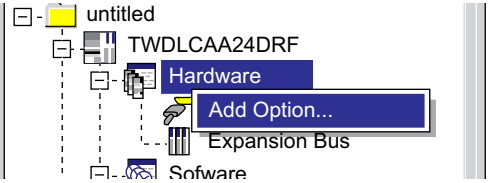
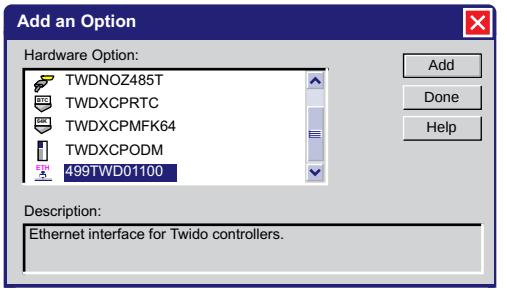
安装 499TWD01100 TwidoPort 模块

安装 TwidoPort 模块到 Twido PLC 系统 （DIN- 导轨和面板）并连接到 Twido PLC 内部总线，按以下步骤操作：

步骤	描述	动作
1	安装准备工作	参考 Twido 可编程控制器硬件参考手册 （TWD USE 10AE）使用说明： ● Twido 模块的正确安装位置， ● 增加和拆卸 Twido 组件从 DIN 导轨， ● 直接固定在面板表面， ● 控制器箱内模块的最小间隙。
2	安装 499TWD01100 TwidoPort 模块	在 DIN 导轨和面板上安装模块，更多细节参看（TwdoHW 硬件参考指南）
3	保护地 (PE) 接地	连接一根接地线到 TwidoPort 底部的 M3 螺丝端子。
4	串行口和以太网连接 顶部插头 （来自 Twido 串行口）  底部插头 （来自以太网直连或者是交叉线电缆）	连接 （所提供的） TwidoPort-to-Twido 电缆尾端的模块插头到 TwidoPort 的串行口上，并且另一端连到 Twido PLC 的 RS 485 串行口 连接 RJ-45 以太网插头到 TwidoPort 以太网口。（本电缆由用户自备，应符合系统和以太网的要求。）

申明 499TWD01100 TwidoPort 模块

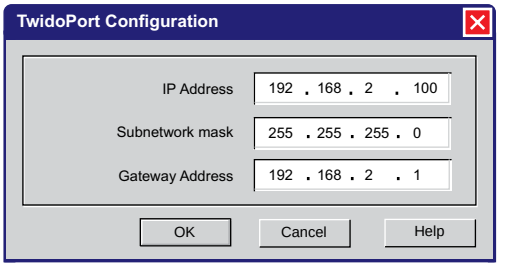
下表显示了申明 499TWD01100 TwidoPort 模块的不同阶段。

步骤	动作	注释
1	使用 TwidoSoft (v. 3.0 或更高) 配置 Twido 控制器的通信选件，右键点击 **Port 1：xxxxxx， 1 →** 编辑控制器通讯配置 ... (见注 1。)	
2	从控制器通讯设置对话框，设定通讯协议为 **Modbus**。	注意：如通讯协议未设为 MODBUS， 499TWD01100 TwidoPort 模块不能被添加到 Twido 硬件上去。
3	配置 Modbus 通讯参数。	Twido 控制器的 RS-485 Modbus 口必须配置成 **9600**, **19200**, 或 **38400** 的波特率 以支持 TwidoPort 的自动波特率功能。（见注 1 和 2。）
4	从 TwidoSoft 应用浏览器里，右击硬件→添加选件	
5	当添加选件对话框出现后： ● 选择 **499TWD01100** ● 点击增加。 ● 在这里可以添加其他选件模块。 注意：只有一块 499TWD01100 TwidoPort 模块可用。 ● 点击完成。	所有的 Twido 控制器都支持 (除了 TWDLCAE40DRF 内置以太网。) 
注意 1	**Twido 的 RS-485 Modbus 口可用。**	
注意 2	**对于最快的初始自动波特率，选择 38400-8-N-1 Twido Modbus 的地址为 1。**	

配置 499TWD01100 TwidoPort 模块

注意：TwidoPort 的以太网参数只能在 TwidoSoft 离线模式下设置。

按以下步骤配置 TwidoPort 的以太网参数：

步骤	动作	注释
前言	要了解更多的有关 IP 参数的信息 （IP 地址，子网掩码和网关地址），参见 IP 寻址，*p.176 和私有 IP 地址， p. 181*。	
1	右键点击 TwidoPort 图标配置 TwidoPort IP 参数。	**结果：**此以太网配置出现，如下所示。 
2	输入 TwidoPort 的静态 **IP** 地址格式为点十进制。 （见注释 1 和 2）	**注意：**为了更好的设备通讯，运行 Twidosoft 的 PC 和 TwidoPort 的 IP 地址应在同一网段。
注 1	**咨询网络或系统管理员以获得网络的有效 IP 参数。**	
注 2	**每个设备必须有唯一的 IP 地址以在网络上获得正常通信。连接 TwidoPort 到网络时，将检查是否 IP 地址重复。如在网络上有重复 IP 地址，将有状态 LED 周期性闪烁四次。然后用户必须在此域输入一个新的非重复 IP 地址。**	
注 3	**除非 TwidoPort 对子网有特殊需要，使用默认子网掩码。**	
注 4	**如网络中无网关设备，把 TwidoPort 的 IP 地址输入网关地址域即可。**	

步骤	动作	注释
3	输入网络管理员分配给 TwidoPortr 的合法的子网掩码。请注意不能让此域空白，必须输入一个值。 （见注释 1 和 3）	**注意：**为了更好的设备通讯，运行 Twidosoft 的 PC 和 Twido 控制器的子网掩码必须匹配。 默认情况下， TwidoSoft 应用程序将自动根据用户在以上 IP 地址域中的 IP 类别计算并显示一个默认的子网掩码。根据 Twido 网络 IP 地址类别，默认的子网掩码值遵循如下规则： A 类网络 -> 默认子网掩码：255.0.0.0 B 类网络 -> 默认子网掩码：255.255.0.0 C 类网络 -> 默认子网掩码：255.255.255.0
4	输入网关。 （见注 1 和 4。）	在 LAN 中，网关应和 TwidoPort 在同一网段。这一信息是由网络管理者提供 TWIDOSOFT 不提供默认网关值，必须手工在网关地址栏内输入有效的网关地址。
5	确认配置，下载到 Twido 控制器。	
注 1	**咨询网络或系统管理员以获得网络的有效 IP 参数。**	
注 2	**每个设备必须有唯一的 IP 地址以在网络上获得正常通信。连接 TwidoPort 到网络时，将检查是否 IP 地址重复。如在网络上有重复 IP 地址，将有状态 LED 周期性闪烁四次。然后用户必须在此域输入一个新的非重复 IP 地址。**	
注 3	**除非 TwidoPort 对子网有特殊需要，使用默认子网掩码。**	
注 4	**如网络中无网关设备，把 TwidoPort 的 IP 地址输入网关地址域即可。**	

BootP 配置

BootP 过程

TwidoPort 在 2 分钟内等 Bootp 服务器对它的 Bootp 请求传输给以响应。
如无响应， TwidoPort 将以 MAC 地址构建默认 IP 地址：

MAC 地址

MAC 地址结构如下：
MAC[0] MAC[1] MAC[2] MAC[3] MAC[4] MAC[5]。

举例，如 **MAC** 地址是 0080F4012C71，默认 **IP** 地址是 85.16.44.113。

11.2 TwidoPort 的 Telnet 配置

概览

本章主题

本章介绍如何配置 ConneXium TwidoPort 模块。

本节包含了哪些内容?

本节包含了以下主题：

介绍 Telnet 配置

Telnet 配置概览

你能用远程登录的方式配置 TwidoPort （使用 VT100 兼容的远程登录客户端）以应对下述情况：Twido 配置缺失或 BootP 请求 2 分钟内不被响应 (导致使用默认 IP 地址)。

准备 Telnet 配置

注意：TwidoPort 的 Telenet 要求
用 Telnet 配置 TwidoPort 时，确信：
- TwidoPort 由 Twido 控制器通过串口供电。
- Telnet 的本地反馈设为关闭。

要使用 Telnet，必须添加 TwidoPort 默认 IP 地址 (或 TwidoPort 配置的 IP 地址) 到 PC 的路由表，使用如下命令：

```
C:\> route add 85.0.0.0 mask 255.0.0.0 local_IP_address_of_PC
```

示例：
如 PC 的 IP 地址是 192.168.10.30 并且 TwidoPort 的默认 IP 地址 （或配置的 IP 地址）是 85.16.44.113，则完整的命令如下：

```
C:\> route add 85.0.0.0 mask 255.0.0.0 192.168.10.30
```

Telnet 主菜单

启动 Telnet 主菜单

启动 Telnet （如：键入 telnet 85.16.44.113 在命令行或使用 Windows™ Hyperterminal™), Telnet 主菜单将显示：回车键：

```
        Telemecanique 499 TWD 01 100 Configuration and Diagnostics
                (c) 2004 Schneider Automation Inc

1) IP/Ethernet Settings
                    IP Source: DEFAULT
                  IP Address: 85.16.44.113
             Default Gateway: 85.16.44.113
                   Netmask: 0.0.0.0
        Ethernet Frame Type: ETHERNETII

2) Serial Configuration
        Baud Rate: 19200
        Data Bits: 8
           Parity:  NONE
        Stop Bits: 1
         Protocol:  RTU

3) Gateway Configuration
          Slave Address Source: UNIT_ID
             Gateway Mode: SLAVE
               MB Broadcasts: ENABLED

4) Security Configuration

5) Ethernet Statistics

6) Serial Statistics

Commands: D)efault settings, S)ave, F)irmware Upgrade, Q)uit without save
     Select Command or Parameter(1..6) to change:
```

IP/ 以太网设置

配置 IP/ 以太网设置

用以下指令改变 IP/ 以太网设置：

步骤	动作	注释
1	启动 Telnet。	用指令打开 Telnet 主菜单（见 *Telnet 主菜单， P.312*）。
2	选 （键入）1 改变 IP 来源到 STORED 然后按回车键	STORED 已是 IP 源。
3	手动设定期望 IP 参数。（见 TwidoPort 以太网设定 按下表）	其他参数包括： ● IP 地址 ● 默认网关 ● 掩码 ● 以太网帧类型
4	选择 R 按回车键	Telnet 主菜单出现。 (再按回车键能刷新屏幕。)

IP Source

选择的 IP 源 指示获得 IP 配置的地址：

- STORED- 从本地闪存。
- SERVED- 从 BootP 服务器。
- TWIDO- 从 Twido 控制器。

默认 IP address (默认值) 从 MAC 地址获得。（通过定义，默认设置不可选）

注意：Twido 控制器内的有效配置比用户选择有高优先权。

以太网设定举例

下图显示 TwidoPort 以太网设定举例：

```
Telemecanique 499 TWD 01 100 Configuration and Diagnost
        (c) 2004 Schneider Automation Inc

IP/Ethernet Settings
-------------------------------------------------
        1)IP Source: DEFAULT
        2)IP Address: 85.16.44.113
        3)Default Gateway: 85.16.44.113
        4)Netmask: 0.0.0.0
        5)Ethernet Frame Type: ETHERNET2
-------------------------------------------------
Commands: R)eturn to Main Menu
Select Command or Parameter(1..N) to change:
```

串行口参数设置

前言

注意：在通常情况下，不必配置 TwidoPort 串行口参数，因为模块支持自动波特率功能。

串行口参数配置

要配置 TwidoPort 串行口参数：

步骤	动作	注释
1	启动 Telnet。	用指令打开 Telnet 主菜单（见 *Telnet 主菜单，p.312*）。
2	选 （键入） 2 去改变串行口参数配置。	见下图。
3	校验或复位设置。	其他参数包括： ● 波特率 ● 数据位 ● 校验位 ● 停止位 ● 协议
4	选择 R 按回车键。	Telnet 主菜单出现。（能再次按回车键刷新屏幕。）

串行口参数配置举例

下图显示 TwidoPort 串行口参数配置：

```
Telemecanique 499 TWD 01 100 Configuration and Diagnostics
        (c) 2004 Schneider Automation Inc

 Serial Configuration
        1) Baud Rate:  19200
        2) Data Bits:  8
        3)    Parity:  NONE
        4) Stop Bits:  1
           Protocol:   RTU

Commands: R)eturn to Main Menu
Select Command or Parameter(1..N) to change:
```

配置网关

前言

注意：在通常情况下，不必配置 TwidoPort 网关。

配置参数网关

配置 TwidoPort 网关：

步骤	动作	注释	
1	启动 Telnet。	用指令打开 Telnet 主菜单（见 *Telnet 主菜单， p.312*）。	
2	选择 （键入） 3 改变网关参数。	见下图。	
3	以下网关参数可用：		
	(1) 子站地址源	固定的	如果子站地址源是固定的，设定地址位 Twido Modbus 地址。有效地址是 1 到 247。
		UNIT_ID	使用 Modbus/TCP 帧的单元 ID。
	(2) 网关模式	子站	仅此版本可选。
	(3) MB broadcasts	禁止	TwidoPort 的串行口上无广播消息发出。
		可用	Twido 的串行口上有广播消息发出 (见下注)
4	选择 R 按回车键。	Telnet 主菜单出现。(能再次按回车键刷新屏幕。)	
注意	**Twido 不支持广播 Modbus 消息。**		

网关设定举例

下图显示 TwidoPort 的网关设定举例：

```
Telemecanique 499 TWD 01 100 Configuration and Diagnostics
        (c) 2004 Schneider Automation Inc

Gateway Configuration
        1) Slave Address Source: UNIT_ID
        2) Slave Address: 20
        3) Gateway Mode: SLAVE
        4) MB Broadcasts: ENABLED

Commands: R)eturn to Main Menu
Select Command or Parameter(1..4) to change: _
```

安全配置

配置安全设定

用下面的指令改变默认的密码：

步骤	动作	注释
1	启动 Telnet。	用指令打开 Telnet 主菜单（见 *Telnet 主菜单，p. 312*）。
2	选 （键入） 4 按回车键	安全配置屏幕出现。
3	选择 C 按回车键	
4	输入旧密码。	被授权的用户知道旧密码 （默认是 **USERUSER**）。
5	输入新密码。	输入新密码 （见 *下注*）。
6	再次输入新密码。	见可接受的密码的注释。
7	选择 R 按回车键。	Telnet 主菜单出现。（能再次按回车键刷新屏幕。）
注意	密码细节： ● 最小长度：**4** 字符 ● 最大长度：**10** 字符 ● 可用字符：**0** 到 **9**， **a** 到 **z**， **A** 到 **Z** (不能用空格)	

以太网统计

显示以太网统计

显示 TwidoPort 的以太网统计：

步骤	动作	注释
1	启动 Telnet。	用指令打开 Telnet 主菜单（见 *Telnet 主菜单，p. 312*）。
2	选 （键入） 5 显示以太网模块统计屏幕。	见下图：
3	按回车键刷新屏幕。	
4	按 C 清除统计，按回车键。	所有的计数器复位到 0。
5	选择 R 按回车键。	Telnet 主菜单出现。（能再次按回车键刷新屏幕。）

以太网模块统计屏幕

TwidoPort担 以太网模块统计屏幕：

```
                Telemecanique 499 TWD 01 100 Configuration and Diagnostics
                        (c) 2004 Schneider Automation Inc
                        ETHERNET MODULE STATISTICS

Status: 0x9103                          IP Address: 192.168.1.141
System Log Entry: No                    Mac Address: 0:80:f4:0:4c:18
Transmit Speed: 100BASE-T               Subnet Mask: 255.255.0.0
Full/Half Duplex: Half Duplex           Gateway Address: 192.168.1.1

Transmit Statistics     Receive Statistics      Functioning Errors

Transmits: 63           Receives: 532             Missed Packets: 0
Transmit Retries: 0     Framing Errors: 0         Collision Errors: 0
Lost Carrier: 0         Overflow Errors: 0        Transmit Timeouts: 0
Late Collision: 0       CRC Errors: 0             Memory Errors: 0
Tx Buffer Errors: 0     Rx Buffer Errors: 0       Net Interface Restarts: 0
SILO Underflow: 0

Broadcast Packets Received: 37          Multicast Packets Received: 7

Commands: [Enter] to Refresh, C>lear Statistics, R>eturn to Main Menu
```

串行口统计

显示串行口统计

显示 TwidoPort 的串行口统计：

步骤	动作	注释
1	启动 Telnet。	用指令打开 Telnet 主菜单（见 *Telnet 主菜单，p. 312*）。
2	选 （键入 6）显示串行口统计屏幕，按回车键。	见下图：串行口统计被更新。
3	按 C 清除统计，按回车键。	所有的计数器复位到 0。
4	选择 R 按回车键。	Telnet 主菜单出现。(能再次按回车键刷新屏幕。)

串行口统计屏幕

TwidoPort担 串行口统计屏幕：

```
                Telemecanique 499 TWD 01 100 Configuration and Diagnostics
                        (c) 2004 Schneider Automation Inc

---------------------- SERIAL STATISTICS --------------------------------

Serial Bus Statistics
        Bus Message Count: 8284
        Bus Comm. Error Count: 0
Modbus Slave Statistics
        Slave Message Count: 4142
        Slave Exception Error Count: 3187
        Slave No Response Count: 0

-------------------------------------------------------------------------

Commands: [Enter] to Refresh, C)lear Statistics, R)eturn to Main Menu
```

保存配置

保存配置

要保存配置，键入配置密码：

步骤	动作	注释
1	启动 Telnet。	用指令打开 Telnet 主菜单 （见 *Telnet 主菜单， p. 312* ）。
2	选择 S 按回车键	
3	输入配置密码。	默认密码是 USERUSER)。 (见下注)。
注意	设置个性密码的细节参见安全配置，**p. 318**。	

保存配置确认屏幕

TwidoPort 扭保存配置确认屏幕：

```
                Telemecanique 499 TWD 01 100 Configuration and Diagnostics
                        (c) 2004 Schneider Automation Inc
                        SAVE CONFIGURATION
================================================================================

        Configuration successfully stored to Twido.
        Reboot your module for the new Configuration to be in effect.

        Rebooting in 5 Seconds. You will lose your telnet connection.

Connection to host lost.
```

恢复默认值

恢复默认值

恢复默认值：

步骤	动作	注释
1	启动 Telnet。	用指令打开 Telnet 主菜单（见 *Telnet 主菜单，p. 312*）。
2	选择 D 显示默认配置屏幕。	见下图：
3	按回车键。	按回车键显示主菜单
4	保存默认配置。	见以上保存配置 （见保存配置，p.321）。

默认配置屏幕

TwidoPort's 担默认配置屏幕：

升级 TwidoPort Firmware

前言

注意：
1. 在升级前必须获得新版本的 TwidoPort firmware。
2. 升级前停止其他操作。
3. 升级时 Modbus 通讯无效。

升级 Firmware

用获得的最新 firmware 版本升级现有 TwidoPort 的 firmware 按以下步骤操作：

步骤	动作	注释
1	启动 Telnet。	用指令打开 Telnet 主菜单（见 *Telnet 主菜单，p. 312*）。
2	选 F 启动 firmware 升级	5 秒后，TwidoPort 复位，Telnet 连接断开。
3	在命令行输入：ftp 和 TwidoPort 的 IP 地址。	举例：ftp 85.16.44.113
4	输入：ftptwd	在登录提示处。
5	输入：cd fw	用户进入 fw 目录
6	输入：put App.out。（见注 1 和 2。）	一条信息指明 ftp 完成。（见注 3。）
注 1	**文件是大小写有关的**	
注 2	**确信 App.out 在 ftp 客户端的当前工作目录下。**	
注 3	**一条信息指明 TwidoPort 在 ftp 完成后 5 秒内将重启动。**	

Firmware 升级过程中

下图显示 Firmware 升级过程 屏幕

```
        Telemecanique 499 TWD 01 100 Configuration and Diagnostics
        ----------------------------------------------------------

        FIRMWARE UPGRADE IN-PROGRESS...
        Module will reboot in 5 Seconds.
        After Reboot, Connect via FTP to download new Firmware.

        FTP Instructions:
                1) Connect via FTP:  ftp  192.168.2.160
                2) Change to /fw directory:  ftp>cd fw
                3) Download new fw:  ftp>put App.out

        After the FTP download is complete, the module will reboot automatically
.

        Rebooting now.  Goodbye.


Connection to host lost.
```

Kernel 模式

如有效的 firmware 不存在， TwidoPort 进入 Kernel 模式。此时用 Telnet 连接到 TwidoPort，下图将显示：

```
Telemecanique 499 TWD 01 100
----------------------------------------------
Kernel Version 90.02d

Download valid Exec.App.out, to leave kernel mode.

   To exit type 'quit' 'QUIT' or control D
```

忘记密码 /IP 配置?

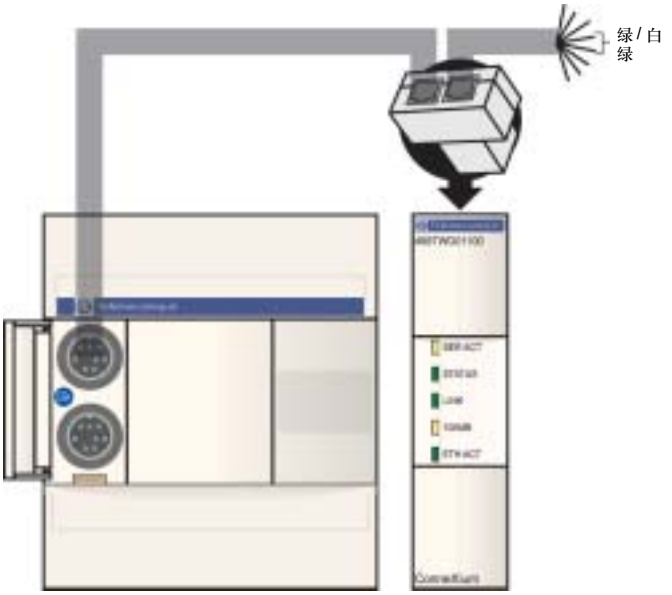
在备份模式连接

用指令在备份模式连接到 TwidoPort。

步骤	动作	注释
1	连接串行口的 3 和 6 脚 (地)。	用 Schneider 170 XTS 04 100 RJ-45 T-connector. （*见下图*）
2	通过 ftp 连接到 TwidoPort。（见注）	TwidoPort 用以下默认 IP 配置： ● IP 地址：192.168.2.102 ● 子网掩码：255.255.0.0 ● 网关地址：192.168.2.102 ● 帧类型：Ethernet II
3	得到文件 fw/Conf.dat。	得到 IP 配置和密码 Conf.dat 文件中。
4	用文本编辑器打开 Conf.dat。	
注	不需要密码。	

FTP 连接

下图显示通过 FTP 在备份模式下连接到 TwidoPort：



11.3 通信特征

概览

本节主题

本节描述了 ConneXium TwidoPort 以太网关支持的通信特征。

本节包含了哪些内容?

本节包含了以下主题：

以太网特征

导言

ConneXium TwidoPort 增加了对 Telemecanique Twido 产品线的连接。它是在单个 Twido Modbus/RTU (RS-485) 设备和 Modbus/TCP 网络从模式物理层之间的网关。TwidoPort 不要求一个单独的电源，因为它可以通过 Twido 控制器的串口取电。这个网关模块仅支持从模式。

以太网特征

TwidoPort 支持 Ethernet 特点：

- **自动 - 流通**
  TwidoPort 支持 10/100TX 自适应，它仅支持半双工模式。
- **Auto-MDI/MDI-X**
  TwidoPort 支持传送和接收的自动切换以与终端设备建立通信 (auto-MDI/MDI-X)。所以， TwidoPort，使用直连或交叉电缆很容易和其他支持 Ethernet 的终端设备互连。

Modbus/TCP 通信协议

关于 Modbus

Modbus 协议是一个主 / 从协议，它允许主站请求从站响应或者基于他们的请求采取行动。主站对各个从站寻址或者对所有从站发送广播报文。
从机对每一个单独发送给它们的查询返回讯息 （响应）。但对广播方式的查询不做响应。

关于 TCP Modbus/TCP 通信

TwidoPort 支持多达 8 个 Modbus/TCP 同时连接。试图使用大于 8 个连接会导致性能降低，因为 TwidoPort 关闭最先打开的连接以提供新的连接。

工作原理

Modbus/TCP 客户端可以通过 TwidoPort 桥连接 Twido，该桥处于 Twido 设备 (Modbus/RTU 通过 RS-485 串行连接) 与 Modbus/TCP 以太网设备之间。

注意：当在你的网络上使用 TwidoPort 时，你的系统设计要求必须考虑到串行连接内在的带宽限制。每秒 40 个 Modbus 包可得到最好的性能。一个请求命令请求多个寄存器要比一个请求命令请求一个寄存器效率高。

你不能通过 TwidoPort 接收来自 Twido 控制器的读或写请求。

本地支持的 Modbus 功能码

列表功能代码

TwidoPort 回应以下本地支持的 Modbus 功能代码、（本地支持的功能代码被 TwidoPort 直接回应，而 Twido 控制器不回应。）

Modbus 功能代码	**子功能代码**	**操作码**	**操作码**
8	0	无效	返回询问数据
8	10	无效	清除计数器
8	11	无效	返回总线信息计数
8	12	无效	返回总线通信错误计数
8	13	无效	返回总线异常错误计数
8	14	无效	返回从站报文计数
8	15	无效	返回从站无响应计数
8	21	3	得到以太网统计
8	21	4	清除以太网统计
43	14	无效	读取设备 ID（见*注释 1*。）
注 1	**TwidoPort 仅支持基本对象 ID，既可以用流又可以用单独访问来读设备标示功能代码。**		

注意：有关信息形式和处理的详细信息。见 Modbus 规格在 www.modbus.org

操作显示操作

12

概览

本章主题

本章提供了使用可选 Twido 操作显示的详细资料。

本章包含了哪些内容?

本章包含了以下主题：

操作显示

介绍

操作显示是一个 Twido 选件，用于显示和控制应用程序数据和一些控制器功能如运行状态和实时钟 （RTC）。该选件可以是一体型控制器的卡 （TWDXCPODC）或者模块型控制器的扩展模块 （TWDXCPODM）。
操作显示有两种工作模式：

- 显示模式：仅显示数据。
- 编辑模式：允许改变数据。

注意：操作显示在控制器扫描循环的特定时间间隔被更新。这将在解释专用输出 %PLS 或 %PWM 输出的显示时导致混乱。这些输出被采样时，它们的输出将总为零，并且显示这个值。

显示及功能

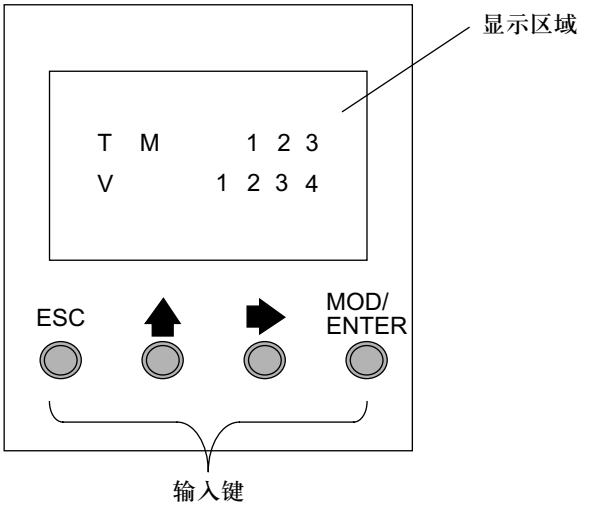
操作显示提供了下面各个显示，以及完成每个显示相应的功能。

- 控制器标识和状态信息：操作显示显示固件修订本及控制器状态。用运行，初始化，停止命令改变控制器状态。
- 系统对象和变量：数据显示
  由地址选择应用程序数据：%I， %Q，和基本控制器的其它软件对象。监控和改变所选软件数据对象的值。
- 串口设置：通信显示
  显示和修改通信串口设置。
- 日期时钟定时：时间 / 日期显示
  显示和配置当前日期和时间 （如果 RTC 已安装）。
- 实时修正：RTC 因子
  显示和修改选件 RTC 的 RTC 修正值。

注意：
1. TWDLCA • 40DRF 系列内置 RTC 卡。
2. 对其他控制器，只有安装了实时钟 （RTC）选项卡 （TWDXCPRTC），时钟和实时纠错功能才有效。

图例

下面图例显示了操作显示视图，由一个显示区域和四个可按按钮输入键组成。



显示区域

显示提供了一个可以显示两行字符的 LCD 显示器：

- 显示器的第一行有三个 13 段字符和四个 7 段字符。
- 第二行有一个 13 段字符，一个 3 段字符 （表示正 / 负号），和五个 7 段字符。

注意：如果在正常方式下，第一行显示对象名，而第二行显示其数值。
如在数据方式下，第一行显示 %SW68 数值，而第二行显示 %SW69 数值。

输入键

四个输入可按按钮的功能取决于操作显示模式。

关键	显示模式下	编辑模式下
ESC		放弃修改并返回到以前显示。
▲		进入被编辑对象的下一值。
▶	前进到下一显示。	进入编辑的下一对象类型。
MOD/ ENTER	进入编辑模式。	接受修改并返回到以前显示。

选择及操作显示

操作显示的初始显示屏幕显示控制器标识和状态信息。▶ 按钮顺序进入每个显示。如果控制器没有检测到选件 RTC 卡 (TWDXCPRTC)，将不会显示日期时钟时间或实时修正因子屏幕。

一个捷径是按 ESC 键返回到初始显示屏幕。对大部分屏幕，按下 ESC 键将返回到控制器标识和状态信息屏幕。只有当系统对象和变量编辑没有初始输入 （%I0.0.0）时，按下 ESC 将让您进行第一次或初始系统对象输入。

若是修改对象值，不是按 ▶ 按钮进入第一个值的数字，而是按 MOD/ENTER 键。

控制器标识和状态信息

介绍

Twido 可选操作显示的初始显示屏幕显示控制器标识和状态信息。

举例

固件版本被显示在显示区域的右上角，并且控制器状态显示在显示区域的左上角，如下所示：

控制器状态

控制器状态包含如下：

- **NCF：没被配置**
  控制器处于 NCF 状态直到应用程序被装载。应用程序装载之前不允许其它状态。可以通过修改系统位 %S8 来测试输入 / 输出 （见系统位 (%S), *p. 658*）。
- **STP：被停止**
  一旦应用程序存在于控制器中，状态改变到 STP 或被停止状态。在此状态，应用程序不运行。输入被更新且数据值保存它们最近的值。输出在此状态不被更新。
- **INI：初始**
  您能只能从 STP 状态选择切换控制器到 INI 或初始状态。应用程序不运行。控制器的输入不被更新且数据值设为它们的初始状态。此状态输出不被更新。
- **RUN：运行**
  当处于 RUN 或运行状态时应用程序运行。控制器的输入被更新且数据值根据应用程序被设置。这是输出被更新的唯一状态。
- **HLT：暂停 （用户程序错误）**
  如果控制器进入 ERR 或出错状态，应用程序被暂停。输入被更新且数据值保存它们最近的值。此状态输出不被更新。这个模式下，错误代码以一个无符号十进制值显示在操作显示的右下部分。
- **NEX：不可执行 （不可执行）**
  对用户逻辑程序进行在线修改。结果：应用程序不再被执行。它将不能返回状态直到引起不可执行状态的所有原因被解决。

显示和改变控制器状态

使用操作显示，您能改变 STP 状态到 INI 状态，或 STP 到 RUN，或 RUN 到 STP。改变控制器状态如下：

步骤	动作
1	按住 ➡ 键直到操作显示可见 （或按 ESC）。当前控制器状态显示在显示区域的左上角。
2	按 MOD/ENTER 键进入编辑模式。
3	按住 ⬆ 键选择一个控制器状态。
4	按 MOD/ENTER 键接受修改值，或按 ESC 键放弃任何修改。

系统对象和变量

介绍

可选操作显示为监控和调节应用程序数据提供了这些特性：

- 通过地址 （如 %I 或 %Q）选择应用程序数据。
- 监控所选软件对象 / 变量值。
- 修改当前显示的数据对象值 （包括强制输入和输出）。

系统对象和变量

系统对象和变量可以被操作显示显示和修改，按照被访问顺序，列表如下。

对象	变量 / 属性	描述	访问
输入	%Ix.y.z	值	读 / 强制
输出	%Qx.y.z	值	读 / 写 / 强制
定时器	%TMX.V %TMX.P %TMX.Q	当前值 预置值 完成	读 / 写 读 / 写 读
计数器	%Cx.V %Cx.P %Cx.D %Cx.E %Cx.F	当前值 预置值 完成 空 满	读 / 写 读 / 写 读 读 读
存储位	%Mx	值	读 / 写
存储字	%MWx	值	读 / 写
常量字	%KWx	值	读
系统位	%Sx	值	读 / 写
系统字	%SWx	值	读 / 写
模拟输入	%IWx.y.z	值	读
模拟输出	%QWx.y.z	值	读 / 写
高速计数器	%FCx.V %FCx.VD* %FCx.P %FCx.PD* %FCx.D	当前值 当前值 预设值 预设值 完成	读 读 读 / 写 读 / 写 读
超高速计数器	%VFCx.V %VFCx.VD* %VFCx.P %VFCx.PD* %VFCx.U %VFCx.C %VFCx.CD* %VFCx.S0 %VFCx.S0D* %VFCx.S1 %VFCx.S1D* %VFCx.F %VFCx.T %VFCx.R %VFCx.S	当前值 当前值 预设值 预设值 计数方向 捕获值 捕获值 阀值 0 阀值 0 阀值 1 阀值 1 溢出 时基 反射输出使能 反射输入使能	读 读 读 / 写 读 / 写 读 读 读 读 / 写 读 / 写 读 / 写 读 / 写 读 读 / 写 读 / 写 读 / 写
输入网络字	%INWx.z	值	读

对象	变量 / 属性	描述	访问
输出网络字	%QNWx.z	值	读 / 写
Grafcet	%Xx	步位	读
脉冲发生器	%PLS.N %PLS.ND* %PLS.P %PLS.D %PLS.Q	脉冲数 脉冲数 预设值 完成 当前输出	读 / 写 读 / 写 读 / 写 读 读
脉宽调节器	%PWM.R %PWM.P	比率 预置值	读 / 写 读 / 写
鼓形控制器	%DRx.S %DRx.F	当前步数满	读 读
步进计数器	%SCx.n	步进计数器位	读 / 写
寄存器	%Rx.I %Rx.O %Rx.E %Rx.F	输入 输出 空 满	读 / 写 读 / 写 读 读
移位寄 存器	%SBR.x.yy	寄存器位	读 / 写
消息	%MSGx.D %MSGx.E	完成 错误	读 读
AS-I 从设备输入	%IAx.y.z	值	读 / 强制
AS-I 模拟从设备输入	%IWAx.y.z	值	读
AS-I 从设备输出	%QAx.y.z	值	读 / 写 / 强制
AS-I 模拟从设备输出	%QWAx.y.z	值	读 / 写
CANopen 子站 PDO 输入	%IWCx.y.z	单字值	读
CANopen 子站 PDO 输出	%QWCx.y.z	单字值	读 / 写

注意：

1. (*) 表示 32- 位双字变量。除了 Twido TWDLC • A10DRF 控制器外。

2. 变量如果没有在程序中使用，将不会被显示，因为 Twido 使用动态存储分配。

3. 如果 %MW 的值大于 +32767 或小于 -32768，操作显示将连续闪烁。

4. 如果 %SW 的值大于 65535，操作显示将连续闪烁，除了 %SW0%SW11。如果输入值超出限制，其值将返回到配置值。
5. 如果 %PLS.P 的输入值超出限制，其值将写为饱和值。

对象和变量显示与修改

每个系统对象类型通过开始输入对象 (%I) 被访问，顺序经过消息对象 (%MSG)，最后环回到输入对象 （%I）。

显示一个系统对象：

步骤	动作
1	按住 ➡ 键直到数据显示屏幕出现。 输入对象 （“I”）将被显示在显示区域的左上角。字母 “I”（或者前面查看数据对象的名字）不闪烁。
2	按 MOD/ENTER 键进入编辑模式。 输入对象 “I”字符 （或者前面查看数据对象的名字）开始闪烁。
3	按下 ⬆ 键顺序显示对象列表。
4	按下 ➡ 键顺序显示对象类型并按下 ⬆ 键增加其值。用 ➡ 键 ⬆ 键操纵和修改显示对象的所有值。
5	重复步骤 3 和 4 直到编辑完成。
6	按下 MOD/ENTER 键接受修改值。 **注意：**对象的名字和地址在接受任何修改之前必须有效。即它们必须在使用操作显示之前就存在于控制器的配置中。 按下 ESC 放弃编辑模式下的任何修改。

数据值和显示格式

对象或变量的数据值通常以符号或非符号整数显示在显示区域的右下角。另外，所有字段禁止以零开头的显示值。操作显示中每个对象的地址以下面七种格式之一被显示：

- I/O 格式
- AS-I 从设备 I/O 格式
- CANopen 子站 I/O 格式
- 功能模块格式
- 简单格式
- 网络 I/O 格式
- 步进计数器格式
- 移位寄存器格式

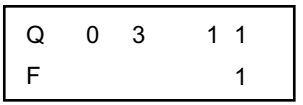
输入 / 输出格式

输入 / 输出对象 （%I，%Q，%IW 和 %QW）具有三部分地址（例如：%IX.Y.Z），被显示如下：

- 左上为对象类型和控制器地址
- 上中为扩展地址
- 右上为 I/O 通道

在简单输入 (%I) 和输出 (%Q) 的情况下，显示的左下部分包含一个字符，或者是“U”表示非强制或者是 “F”表示强制位。强制值显示在屏幕的右下部分。

输出对象 %Q0.3.11 在显示区域显示如下：



AS-I 从设备 I/O 格式

AS-I 从设备 I/O 对象 (%IA， %QA， %IWA 和 %QWA) 具有四部分地址 （例如：%IAx.y.z），被显示如下：

- 左上为对象类型
- 上左偏中为扩展总线上的 AS-I 主模块地址
- 上右偏中为 AS-I 总线上的从设备地址
- 右上为从设备 I/O 通道。

在简单输入 (%IA) 和输出 (%QA) 的情况下，显示的左下部分包含一个字符，或者是“U”表示非强制或者是 “F”表示强制位。强制值显示在屏幕的右下部分。

输出对象 %QA1.3A.2 在显示区域显示如下：

```
QA  1   3A      2
F               1
```

CANopen 子站 I/O 格式

CANopen 子站 PDO I/O 对象 (%IWC 和 %QWC) 有 4 部分地址 (如: %IWCx.y.z) ，被显示如下：

- 左上为对象类型
- CANopen 主模块地址显示在左上偏中
- CANopen 子站地址显示在右上偏中
- 子站 PDO I/O 通道显示在右上
- 下面为对象符号值

下例中， PDO 输出对象 %QWC1.3.2 包含符号数 +24680：

```
QWC 1   3       2
      +   24680
```

功能模块格式

功能模块 (%TM， %C， %FC， %VFC， %PLS， %PWM， %DR， %R，和 %MSGj) 具有两部分地址，包含一个对象号和一个变量或属性名字。它们被显示如下：

- 左上为功能模块名字
- 右上为功能模块号 （或例子）
- 左下为变量或属性
- 右下为属性值

下面示例中， 123 号定时器当前值被设为 1, 234。

简单格式

简单格式用于对象 %M，%MW，%KW，%MD，%KD，%MF，%KF，%S，%SW 和 %X：

- 右上为对象号
- 下面是对象的符号值

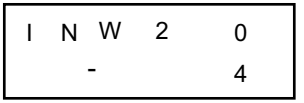
下面示例中， 67 号存储字包含值 +123。

网络输入 / 输出格式

网络输入 / 输出对象 (%INW 和 %QNW) 在显示区域显示如下：

- 左上为对象类型
- 上中为控制器地址
- 右上为对象号
- 下面为对象符号值

下面示例中，配置成远程地址 #2 的远程控制器的第一个输入网络字被设为值 -4。

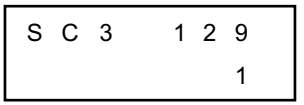


步进计数器格式

步进计数器 (%SC) 格式显示对象号和步进计数器位如下：

- 左上为对象名字和编号
- 右上为步进计数器位
- 下面是步进计数器位的值

下面示例中， 3 号步进计数器的第 129 位被置为 1。

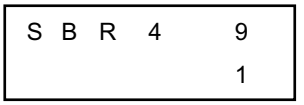


移位寄存器格式

移位寄存器 (%SBR) 在显示区域显示如下：

- 左上为对象名字和编号
- 右上为寄存器位编号
- 右下是寄存器位值

下面示例是 4 号移位寄存器的显示。



串口设置

介绍

操作显示允许您显示协议设置并改变所有通过 TwidoSoft 配置的串口的地址。串口的最大数目是两个。下面示例中，第一个端口配置成 Modbus 协议，地址为 123。第二个串口配置成远程连接，地址为 4。

串口设置显示和修改

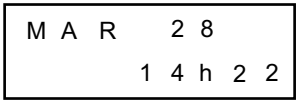
Twido 控制器最多可以支持两个串口。使用操作显示显示串口设置：

步骤	动作
1	按住 ➡ 键直到通信显示出现。第一个串口协议设置的一个字母 (“M”，“R”，或 “A”) 将显示在操作显示的左上角落。
2	按 MOD/ENTER 键进入编辑模式。
3	按住 ➡ 键直到进入您要修改的字段。
4	按住 ⬆ 键增加值。
5	继续 3 和 4 直到地址设置完成。
6	按 MOD/ENTER 键接受修改值或 ESC 放弃任何修改。
注意：地址是控制器配置的一部分。用操作显示改变它的值意味着不能使用原来的 Twido 应用程序连接。 TwidoSoft 将要求重新下载程序。	

实时时钟计时

介绍

如果您的 Twido 控制器安装了 RTC 选件卡 （TWDXCPRTC），您可以使用操作显示修改日期和时间。月被显示在 HMI 显示的左上方。月的地方将包含值 “RTC” 直到输入一个有效值。日显示在右上角。时刻用军事格式。小时和分显示在右下角并用字母 “h” 分开。下面示例显示 RTC 设置为三月 28 日下午 2：22。



注意：
1. TWDLCA • 40DRF 系列内置 RTC 卡。
2. 对其他控制器，只有安装了实时时钟 （RTC）选项卡 （TWDXCPRTC），时钟和实时纠错功能才有效。

日期时钟时间显示和修改

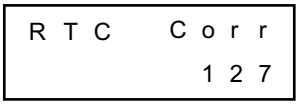
显示和修改日期时钟的时间：

步骤	动作
1	按住 ➡ 键直到时间 / 日期显示出现。月值 (“一月”，“二月”) 将显示在显示区域的左上角。如果月没有被初始化，左上角将显示值 “RTC”。
2	按 MOD/ENTER 键进入编辑模式。
3	按住 ➡ 键直到进入您要修改的字段。
4	按住 ⬆ 键增加值
5	继续 3 和 4 直到日期时间值完成。
6	按 MOD/ENTER 键接受修改值或 ESC 放弃任何修改。

实时修正因子

介绍

您能使用操作显示显示和修改实时修正因子。每个实时时钟 (RTC) 选件模块有一个 RTC 修正因子，用于修正 RTC 模块晶振的不准确。修正因子是一个无符号 3 位整数从 0 到 127，且显示在显示的右下角。
下面示例显示了修正因子 127。



RTC 修正显示和修改

显示和修改实时修正因子：

步骤	动作
1	按住 ➡ 键直到 RTC 因子显示出现。“RTC Corr”将显示在操作显示的上面一行。
2	按 MOD/ENTER 键进入编辑模式。
3	按住 ➡ 键直到进入您要修改的字段。
4	按住 ⬆ 键增加值。
5	继续 3 和 4 直到 RTC 修正值完成。
6	按 MOD/ENTER 键接受修改值或 ESC 放弃任何修改。

Twido 语言描述

III

概览

本部分的主题

本部分提供了使用梯形图，列表和 Grafcet 编程语言创建 Twido 可编程控制器控制程序的说明。

本部分包含了哪些内容?

本部分包含了以下章节：

梯形图语言

13

概览

本章主题

本章描述了怎样使用梯形图语言编程。

本章包含了哪些内容?

本章包含了以下主题：

梯形图介绍

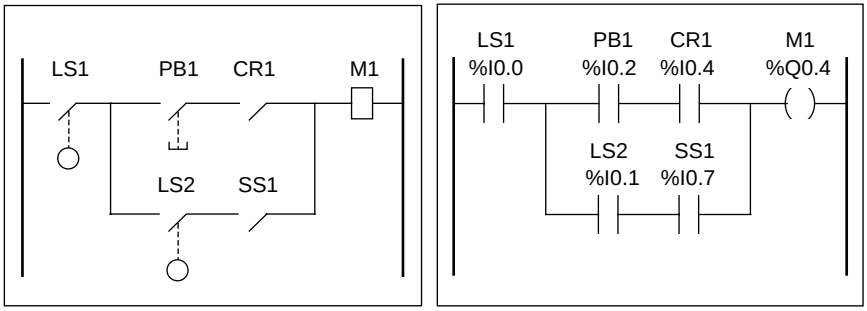
介绍

梯形图类似于用来描述继电器控制电路的继电器逻辑图。两者之间的主要区别是继电器逻辑图中没有的梯形图编程的下列特点：

- 所有的输入都可以用触点符号来表示 (⊣⊢).
- 锁有的输出都可以用线圈来表示 (⊣( )⊢).
- 梯形图指令集中包括数字运算。

梯形图和继电器电路等效

下面图例是一个继电器逻辑电路的简化接线图与其等效梯形图。



继电器逻辑电路　　梯形图

请注意上面图例中，梯形图中所有与继电器逻辑图中开关设备相关的输入都以触点形式表示。继电器逻辑图中的 M1 输出线圈在梯形图中用输出线圈符号表示。梯形图中每个触点 / 线圈符号上的地址标号都对应于与控制器相连的外部输入 / 输出的位置。

梯级

用梯形图编写的程序由梯级构成，梯级是指画在两条垂直电源栏里的图形指令集。梯级由控制器按顺序执行。

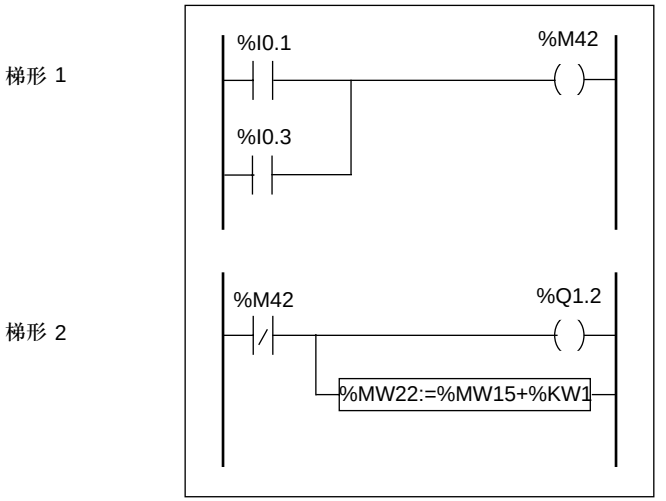
图形指令集有下述功能：

- 控制器的输入 / 输出 （按钮，传感器，继电器，指示灯，等等）
- 控制器的功能 （定时器，计数器，等等）
- 数学和逻辑运算 （加法，除法，与，或，等等）
- 比较运算和其它数字运算 （A<B， A=B，移位，循环，等等）
- 控制器的内部变量 （位，字，等等）

垂直和水平连接这些图形指令最终实现一个或多个输出与 / 或动作。一个梯级最多只能支持一组相关指令。

梯级示例

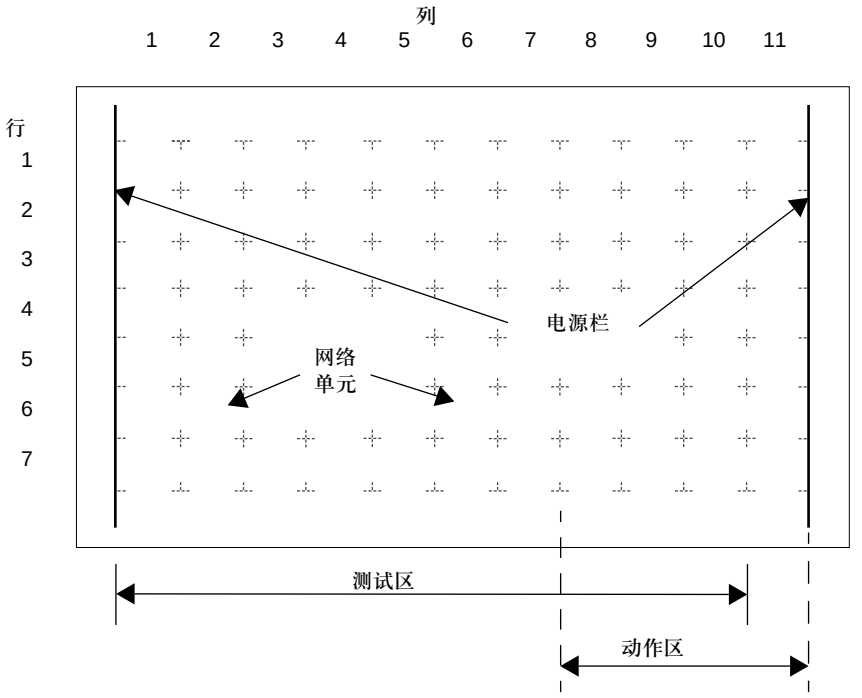
下图是一个由两个梯级组成的梯形图程序示例。



梯形图编程原则

编程网格

每个梯级由 7 行 11 列组成，形成两个区域，如下图所示。



网格区域

梯形图编程网格分为两个区：

- 测试区
  包括动作发生所必须具备的条件。由列 1-10 组成，包括触点，功能模块，和比较模块组成。
- 动作区
  包括测试区相关测试条件所引起的输出或操作。由列 8-11 组成并包括线圈和操作模块。

网格中指令输入

梯级提供了一个 7 行 11 列的编程网格，并从网格的最左上方单元开始。编程即向网格中的单元输入指令。测试指令，比较模块，和功能模块被输入到测试区域的单元并左对齐。测试逻辑为动作区提供了连贯性。在动作区里，线圈，数字运算，和程序流控制指令被输入并右对齐。
梯级在网格中从上到下从左到右地被解释或执行 （完成测试和赋值输出）。

梯级头

除了梯级以外，在它上方还有一个梯级头。用梯级头说明梯级的逻辑目的。梯级头可以包括下列信息：

- 梯级编号
- 标号 （%Li)
- 子程序声明 （SRi:）
- 梯级标题
- 梯级注释

关于使用梯级头说明程序的更详细情况，见*程序文档，p.371*。

梯形图模块

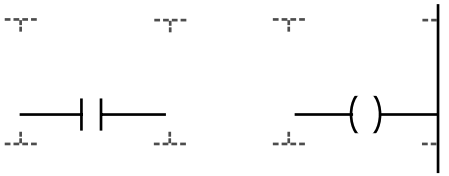
介绍

梯形图由如下所示表示程序流和功能的模块组成：

- 触点
- 线圈
- 程序流指令
- 功能模块
- 比较模块
- 操作模块

触点，线圈 和程序流

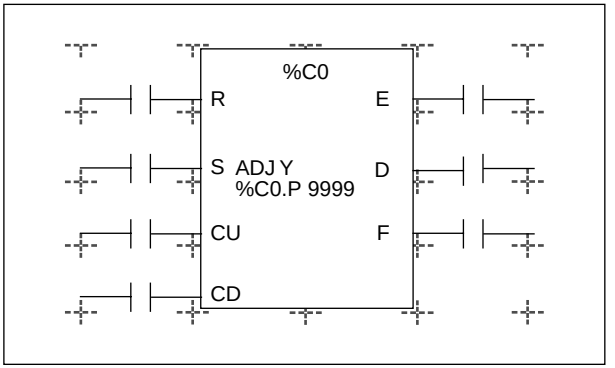
触点，线圈，和程序流 （跳转和调用）指令占据梯形图编程网格的一个单元。功能块，比较块，和操作块占据多个单元。
下面是触点和线圈示例。



触点　　线圈

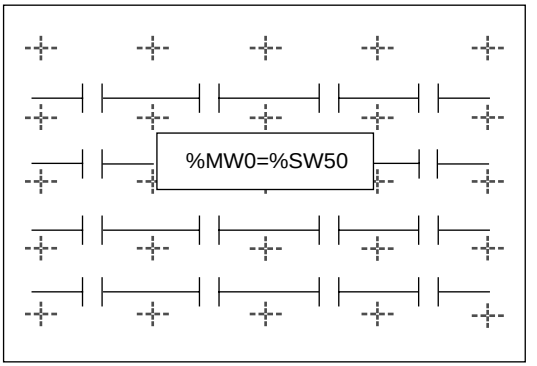
功能模块

功能块位于编程网格的测试区。该模块必须出项在第一行；它的上面和下面不允许出现梯形图指令或连续线。梯形图测试指令放在功能块的输入侧，动作指令放在功能块的输出侧。
功能模块垂直指向，并占据编程网格的四行两列。
下面是一个计数器模块示例。



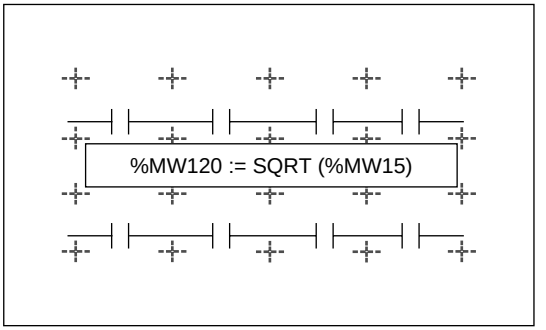
比较模块

比较模块位于网格中的测试区。它可以出现在测试区的任意行列，只要指令全长不超出测试区。
比较模块呈水平走向。它占据网格中的一行两列。
见下面比较模块示例。



操作模块

操作模块位于编程网格的动作区。该模块可以出现在动作区的任意一行。指令右对齐；它出现在右边直至最后一列。
操作模块呈水平走向且占据编程网格的一行四列。
下面是一个操作模块示例。



梯形图图形单元

介绍

梯形图指令由图形单元组成。

触点

触点图形单元用于测试区编程且占据一个单元 （一行一列）。

名称	图形单元	指令	功能
常开触点		LD	当控制位对象为状态 1 时通过触点。
常闭触点		LDN	当控制位对象为状态 0 时通过触点。
上升沿触点		LDR	上升沿：检测控制位对象从 0 变为 1。
下降沿触点		LDF	下降沿：检测控制位对象从 1 变为 0。

连接单元

图形连接单元用于连接测试和动作图形单元。

名称	图形单元	功能
水平连接		两个电源栏之间测试和动作图形单元的连续连接。
垂直连接		平行测试和动作图形单元连接。

线圈

线圈图形单元用于动作区编程且占据一个单元 （一行一列）。

名称	图形单元	指令	功能
直接线圈	—( )—	ST	相关位对象得到测试区结果值。
取反线圈	—(/)—	STN	相关位对象得到测试区结果的相反值。
置位线圈	—(S)—	S	当测试区结果为 1 时相关位对象被置为 1。
复位线圈	—(R)—	R	当测试区结果为 1 时相关位对象被置为 0。
跳转或子程序调用	->>%Li ->>%SRi	JMP SR	连接到一个标注指令，向上跳转或向下跳转。
转换条件线圈	—(#)—		Grafcet 语言。用于当与转换相关的条件满足时跳转到下一步。
子程序返回	<RET>	RET	位于子程序末端返回到主程序。
结束程序	<END>	END	程序结束定义。

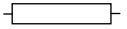
功能模块

功能模块的图形单元在测试区被调用，需要四行两列单元 （除了超高速计数器需要五行两列）。

名称	图形单元	功能
定时器，计数器，寄存器，等等。		每个功能模块使用输入和输出和其它图形单元连接。 **注意**：功能模块的输出不能互相连接 （垂直短接）。

操作和比较模块

比较模块在测试区被调用，操作模块在动作区被调用。

名称	图形单元	功能
比较模块		比较两个操作数，当结果为真时输出变为 1。 大小：一行两列
操作模块		完成算术和逻辑运算。 大小：一行四列

特殊梯形图指令 OPEN 和 SHORT

介绍

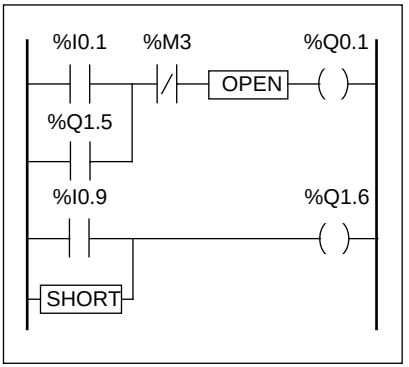
OPEN 和 SHORT 指令为梯形图程序的调试和故障排除提供了简便方法。这两个特殊指令靠短路或者开路梯级的连续性来改变一个梯级的逻辑，如下表所解释。

指令	描述	列表指令
OPEN	在梯级的连续性中创建一个断点，不管最近的逻辑运算结果。	AND 0
SHORT	允许梯级连贯地通过，不管最近的逻辑运算结果。	OR 1

列表编程中， OR 和 AND 用于创建 OPEN 和 SHORT 指令，分别使用立即值 0 和 1。

示例

下面是 OPEN 和 SHORT 指令使用示例。



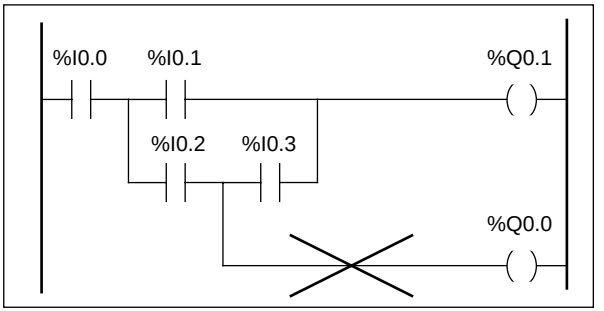

```
LD      %I0.1
OR      %Q1.5
ANDN    %M3
AND     0
ST      %Q0.1
LD      %I0.9
OR      1
ST      %Q1.6
```

编程建议

程序跳转处理	使用程序跳转应避免回路过长，否则会增加扫描时间。避免跳转到上游指令。（上游指令线位于程序跳转之前，下游指令线出现在程序跳转后。）。
输出编程	输出位，类似内部位，在程序中只能被修改一次。在输出位情形下，当输出被更新时只有最后的扫描值被考虑。
使用直接连线的紧急停止传感器	传感器直接用于控制器不能处理的紧急停止。它们必须直接连接到相关输出。
重新上电处理	手动重新上电时，装置的自动重启可能导致意外的设备操作 （使用系统位 %S0, %S1 和 %S13）。
时间和调度模块管理	应该检查系统位 %S51 的状态，它显示任何 RTC 故障。
语法和错误检查	当输入程序时， TwidoSoft 检查指令，操作数，和它们之间结合的语法。

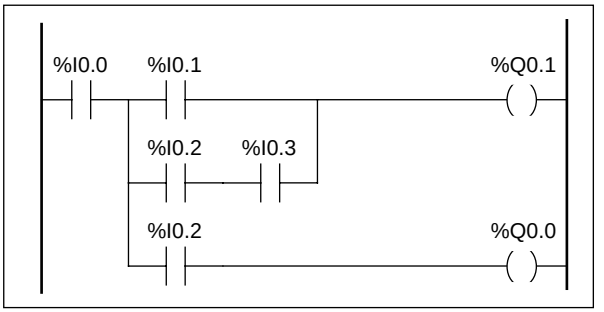
圆括号使用的其它注意事项

赋值操作不能位于圆括号内：



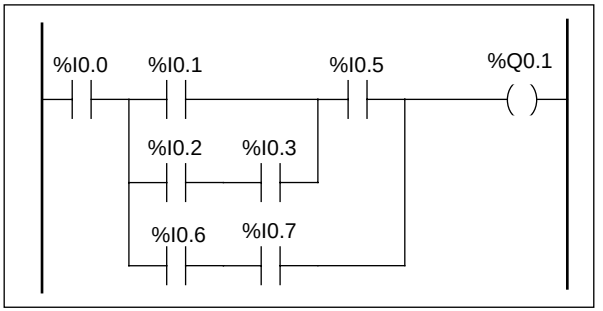

```
LD      %I0.0
AND     %I0.1
OR(     %I0.2
ST      %Q0.0
AND     %I0.3
)
ST      %Q0.1
```

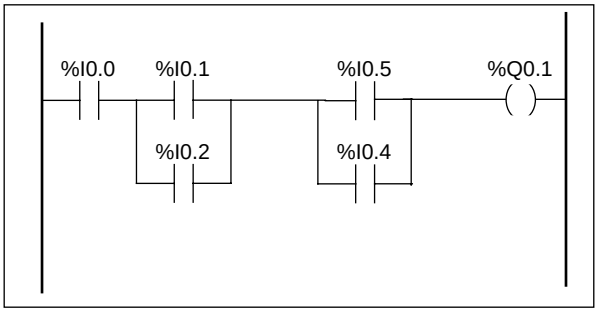
为了实现相同的功能，必须使用下面的等价编程：



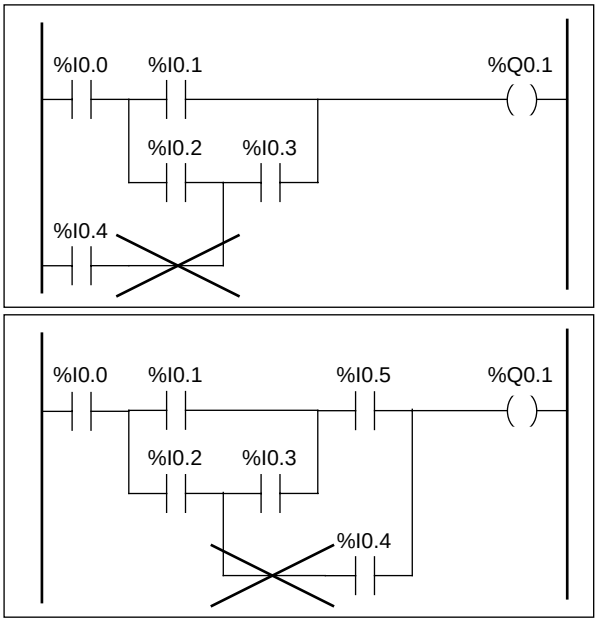

```
LD      %I0.0
MPS
AND(    %I0.1
OR(     %I0.2
AND     %I0.3
)
)
ST      %Q0.1
MPP
AND     %I0.2
ST      %Q0.0
```

如果几个触点平行，它们必须互相嵌套或完全分开：

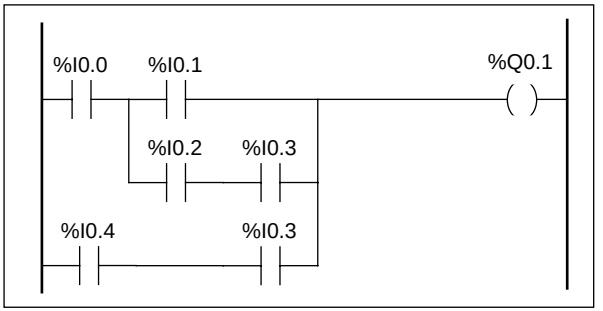




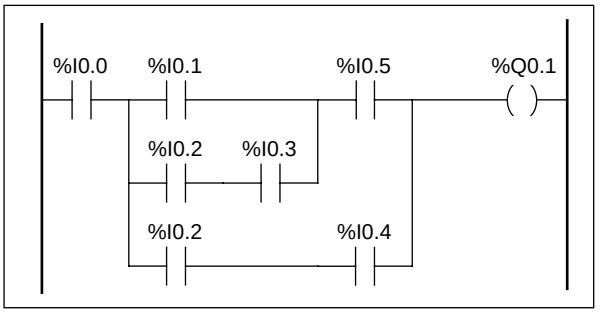
下面图表不能被编程：



为了执行它们的等价图表，它们必须修改如下：




```
LD      %I0.0
AND(    %I0.1
OR(     %I0.2
AND     %I0.3
)
)
OR(     %I0.4
AND     %I0.3
)
ST      %Q0.1
```




```
LD      %I0.0
AND(    %I0.1
OR(     %I0.2
AND     %I0.3
)
AND     %I0.5
OR(     %I0.2
AND     %I0.4
)
)
ST      %Q0.1
```

梯形图 / 列表可逆性

介绍

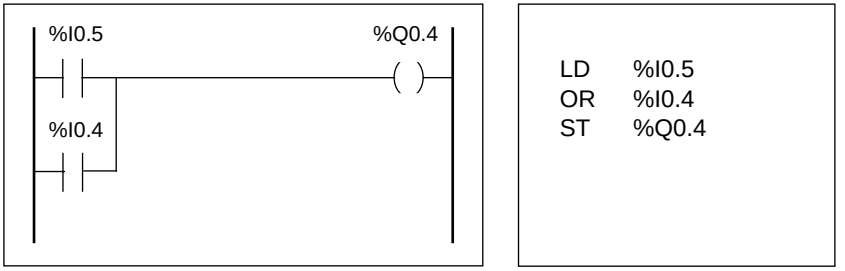
程序可逆是 TwidoSoft 编程软件的一个特点，指支持应用程序在梯形图和列表之间的相互转换。
使用 TwidoSoft 设置程序的默认显示：或者列表或者梯形图格式 （通过用户参数设置）。 TwidoSoft 也可以用来切换列表和梯形图视图。

可逆性理解

理解程序可逆特点的关键在于清楚梯级和相关指令列表之间的关系：

- **梯级：**梯形图指令集合组成的逻辑表达式。
- **列表序列：**列表编程指令集合，对应梯形图指令并表示相同的逻辑表达式。

下面图例显示了一个普通梯级和它的列表指令等效的逻辑表达程序。



应用程序内部都是以列表指令形式存储，不管程序是通过梯形图语言还是列表语言写成。TwidoSoft 利用这两种语言间的程序结构相似性，使用程序内部列表映像将程序显示在列表和梯形图视图和编辑器里，或者是一个列表程序（它的基本形式），或者图形化为一个梯形图，这取决于用户参数的选择。

可逆性保证

梯形图程序一般可以转换到列表。然而，一些列表逻辑不能转换到梯形图。为了保证从列表指令转换为梯形图，请遵照列表编程指令规定在*梯形图 / 列表可逆原则，p.369.*

梯形图 / 列表可逆原则

可逆所需指令

列表语言中可逆功能模块的结构需要使用下列指令：

- **BLK** 标志模块开始，并定义梯级的开始和模块输入部分的起始。
- **OUT_BLK** 标志模块输出部分的起始。
- **END_BLK** 标志模块和梯级的结束。

指令列表程序不是一定要使用可逆功能模块才能正常运行。因为一些指令可以用于列表编程但不可逆。对标准功能块不可逆列表编程的描述，见*标准功能模块编程原则，p.423*。

需要避免的不等价指令

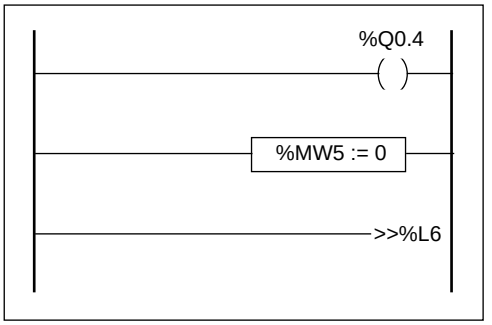
避免使用某些没有等价梯形图的列表指令，或指令与操作数的结合。例如，指令 N（布尔累加器值取反）没有等价梯形图指令。

下表列出了所有不能转换到梯形图的列表编程指令。

列表指令	操作数	描述
JMPCN	%Li	条件为非时跳转
N	没有	取反 （Not）
ENDCN	没有	条件为非时结束

无条件梯级

无条件梯级编程也需要遵守列表编程原则以保证列表和梯级之间的可逆性。无条件梯级没有测试或条件。输出或动作指令在任何情况下都能被激活或执行。
下图是无条件梯级示例及其等价列表序列。




```
LD    1
ST    %Q0.4
LD    1
[%MW5 := 0]
JMP   %L6
```

注意除了 JMP 指令外，上面每个无条件列表序列都以紧跟着数字 1 的装入指令开始。这个结合将布尔累加器置为 1，从而每次程序扫描时线圈 （存储指令）都置为 1， %MW5 都置为 0。无条件跳转列表指令 （JMP %L6）是例外，它的执行与累加器的值无关，不需要将累加器置为 1。

梯形列表栏

如果一个列表程序不完全可逆，则其可逆部分以梯形图显示，不可逆部分显示在梯形列表栏。
梯形列表栏功能类似一个小型列表编辑器，允许用户查看和修改梯形图程序的不可逆部分。

程序文档

标注用户程序

用户可以通过在列表或梯形图编辑器里输入注释来标注自己的程序：

- 在列表编辑器里输入列表注释来标注用户程序。这些注释可能出现在程序指令的同一行，或独立成行。
- 使用梯形图编辑器在梯级头标注用户程序，这些程序注释位于梯级上方。

TwidoSoft 编程软件使用这些注释来转换程序。当将一个程序从列表转换为梯形图时， TwidoSoft 使用其中部分的列表注释构造梯级头。这样，插入到列表序列之间的注释用做梯级头。

列表行注释示例

以下是一个带有列表行注释的列表程序示例。

```
---- ( * 这是梯级 0 头标题 * )
---- ( * 这是梯级 0 的第一个头注释 * )
---- ( * 这是梯级 0 的第二个头注释 * )
     0  LD % I0.0 （* 这是一行注释 *）
     1  OR %I0.1 （* 当转换为梯形图时，一行注释被忽略 *）
     2  ANDM %M10
     3  ST   M101
---- ( * 这是梯级 1 的头 * )
---- ( * 此梯级包含一个标签 * )
---- ( * 这是梯级 1 的第一个头注释 * )
---- ( * 这是梯级 1 的第二个头注释 * )
---- ( * 这是梯级 1 的第四个头注释 * )
     4  %L5：
     5  LD %M101
     6  [%MW20: =%KW2*16]
---- ( * 此梯级仅包含一个头标题 * )
     7  LD %Q0. 5
     8  OR %I0. 3
     9  ORR I0. 13
    10 ST %Q0.5
```

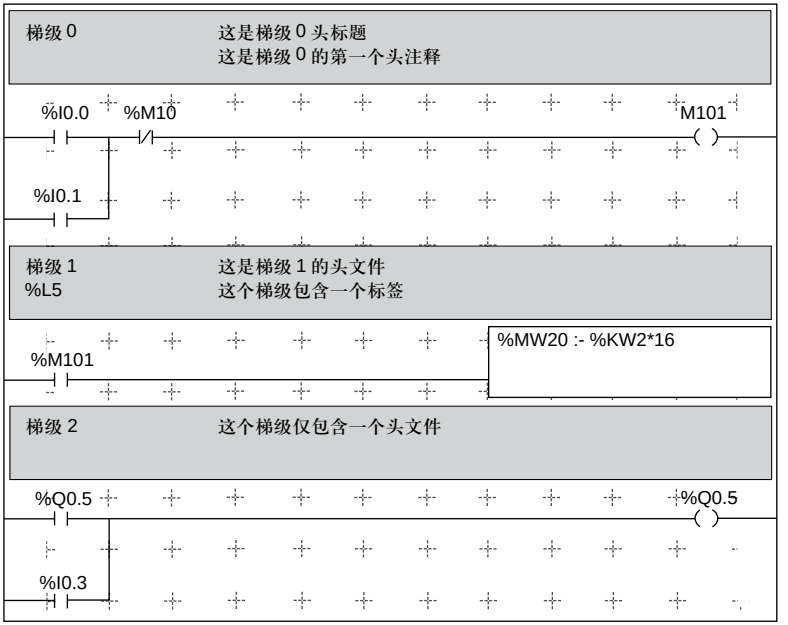
将列表注释转换为梯形图

当将列表指令转换为梯形图程序时，列表行注释依照如下规则显示在梯形图编辑器中：

- 一行的第一条注释指定为梯级头。
- 从第二条开始的任何注释作为梯级的主体部分。
- 一旦标题的主体部分行已经占满，在列表序列之间的其他行注释将被忽略，包括列表指令中的列表注释也同样。

梯级头注释举例

以下是一个带有梯级头注释的梯形图程序。



将梯形图注释转换为列表

当将一个梯形图程序转换为列表指令时，梯级头注释将依照如下规则显示在列表编辑器中：

- 任何梯级头注释将插入到相关列表序列之间。
- 任何标号 （%Li:）或子程序声明 （SRi:）将置于梯级头的下一行列表序列之前。
- 如果列表转换为梯形图，任何被忽略的注释将重新出现在列表编辑器中。

指令列表语言

14

概览

本章主题

本章描述了怎样使用指令列表语言编程。

本章包含了哪些内容?

本章包含了以下主题：

列表程序概述

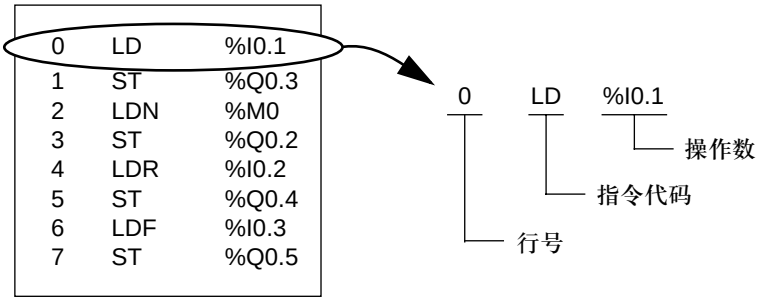
介绍

由一系列指令组成的列表语言写成的程序被控制器顺序执行。每个列表指令表示成一个程序行并由三个部分组成：

- 行号
- 指令代码
- 操作数

列表程序示例

下面是一个列表程序示例。



行号

行号在您输入一个指令时自动生成。空行和注释行没有行号。

指令代码

指令代码是一种操作符号，用于确定使用操作数进行何种操作。典型操作特指布尔和数字运算。

例如，在上面示例程序中， LD 是 LOAD 指令代码的缩写。 LOAD 指令放置 （装载）操作数 %I0.1 的值到一个内部寄存器，即累加器。

指令有两种基本形式：

- 测试指令
  它们对动作执行的必要条件进行设置或测试。例如， LOAD （LD）和 AND。
- 动作指令
  它们按照设置条件的结果执行动作。例如，赋值指令如 STORE （ST）和 RESET （R）。

操作数

操作数是数字，地址，或符号，表示程序在指令中可以处理的一个值。如，在上面的示例程序中，操作数 %I0.1 是一个地址，其值是控制器的一个输入。一个指令根据指令代码的类型，可以有 0 至 3 个操作数。

操作数可以表示：

- 控制器输入和输出如传感器，按钮，和继电器。
- 预置系统功能如定时器和计数器。
- 算术，逻辑，比较和数字运算。
- 控制器内部变量如位和字。

列表指令操作

介绍

列表指令只有一个显式操作数，另外是隐式操作数。隐式操作数是布尔累加器中的值。例如，指令 LD %I0.1 中， %I0.1 为显式操作数。一个隐式操作数存储于累加器中并将被 %I0.1 的值覆盖。

操作

列表指令对累加器和显式操作数的内容执行指定的操作，且其结果将替代累加器。例如，执行 AND %I1.2 指令表示将累加器的值和输入 1.2 进行逻辑与运算，并用运算结果替换累加器的内容。

所有布尔指令，除了 Load， Store，和 Not，都对两个操作数进行操作。两个操作数的值可以是 True 或 False，且指令的程序执行产生一个值：True 或 False。Load 指令将操作数的值装入累加器， Store 指令将累加器的值传递给操作数。Not 指令没有显式操作数，只是转换累加器的状态。

支持的列表指令

下面表格显式了列表指令语言的指令选集：

指令类型	示例	功能
位指令	LD %M10	读内部位 %M10
模块指令	IN %TM0	开始定时器 %TM0
字指令	[%MW10 := %MW50+100]	加运算
程序指令	SR5	调用子程序 #5
Grafcet 指令	-*-8	步 #8

列表语言指令

介绍

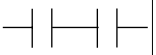
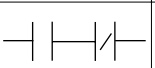
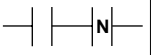
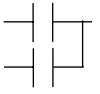
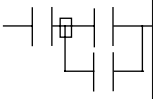
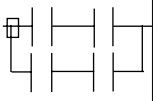
列表语言由下面几种语言组成：

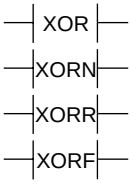
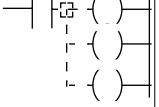
- 测试指令
- 动作指令
- 功能模块指令

本节辨别和描述了用于列表编程的 Twido 指令。

测试指令

下表描述了列表语言中的测试指令。

名称	等价梯形图元素	功能
LD		布尔运算结果与操作数状态相同。
LDN		布尔运算结果为操作数状态取反。
LDR		当检测到操作数 （上升沿）从 0 变为 1 时布尔运算结果变为 1。
LDF		当检测到操作数 （下降沿）从 1 变为 0 时布尔运算结果变为 1。
AND		布尔运算结果等于前面指令布尔运算结果和操作数状态的逻辑与结果。
ANDN		布尔运算结果等于前面指令布尔运算结果和操作数状态取反的逻辑与结果。
ANDR		布尔运算结果等于前面指令布尔运算结果和操作数上升沿（1= 上升沿）检测的逻辑与结果。
ANDF		布尔运算结果等于前面指令布尔运算结果和操作数下降沿（1= 下降沿）检测的逻辑与结果。
OR		布尔运算结果等于前面指令布尔运算结果和操作数状态的逻辑或结果。
AND(		逻辑与 （8 层嵌套）
OR(		逻辑或 （8 层嵌套）

名称	等价梯形 图元素	功能
XOR, XORN, XORR, XORF		异或
MPS MRD MPP		转换到线圈
N	-	取反 （NOT）

动作指令

下表描述了列表语言中的动作指令。

名称	等价梯形图元素	功能
ST	—( )—	相关操作数取值为测试区结果值。
STN	—(/)—	相关操作数取值为测试区结果值取反。
S	—(S)—	当测试区结果为 1 时相关操作数置为 1。
R	—(R)—	当测试区结果为 1 时相关操作数置为 0。
JMP	-	无条件向上或向下转移到一个标记序列。
SRn	->>%SRi	转移到子程序开始。
RET	<RET>	从子程序返回。
END	<END>	程序结束。
ENDC	<ENDC>	布尔运算结果为 1 时程序结束。
ENDCN	<ENDCN>	布尔运算结果为 0 时程序结束。

功能模块指令

下表描述了列表语言中的功能模块指令。

名称	等价梯形图元素	功能
定时器，计数器，寄存器，等等。		每个功能模块均有模块控制指令。一个结构化的格式直接用于硬连线模块的输入和输出。 **注意：**功能模块的输出不能互相连接 （垂直短接）。

圆括号使用

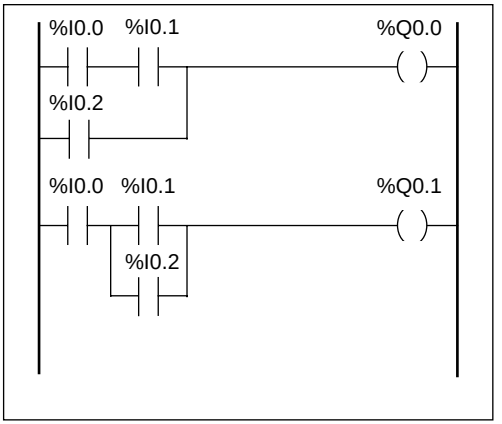
介绍

在逻辑与和逻辑或指令中，圆括号用于指定梯形图的并列部分。圆括号和指令关联如下：

- 左括号和指令 AND 或者 OR 相连。
- 右括号和每个左括号后的指令相连。

与指令使用示例

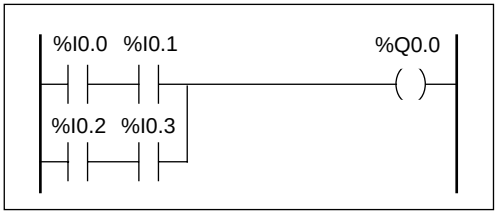
下图是圆括号和与指令使用示例：AND （...）。




```
LD      %I0.0
AND     %I0.1
OR      %I0.2
ST      %Q0.0

LD      %I0.0
AND(    %I0.1
OR      %I0.2
)
ST      %Q0.1
```

或指令使用示例

下图是圆括号和或指令使用示例：OR （...）。




```
LD      %I0.0
AND     %I0.1
OR(     %I0.2
AND     %I0.3
)
ST      %Q0.0
```

修饰符

下表列出了可用于圆括号的修饰符。

修饰符	功能	示例
N	取反	AND （N 或 OR （N
F	下降沿	AND （F 或 OR （F
R	上升沿	AND （R 或 OR （R
[	比较	见*比较指令*，p.453

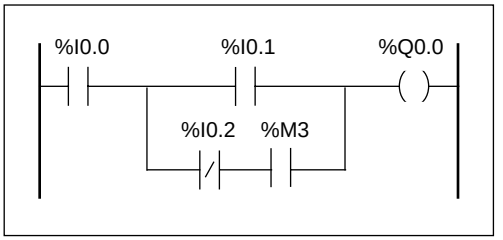
嵌套圆括号

最多可以嵌套八层圆括号。
嵌套圆括号时请遵守下列规则：

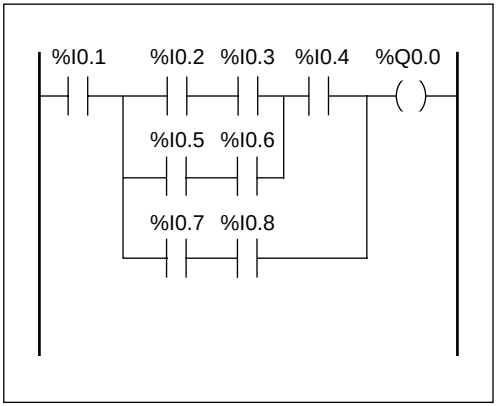
- 左右括号必须对应。
- 标号 （%Li:），子程序 （SRi:），跳转指令 （JMP），和功能模块指令不能位于括号内。
- 存储指令 ST， STN， S，和 R 不能位于括号内。
- 堆栈指令 MPS， MRD，和 MPP 不能位于括号内。

圆括号嵌套示例

下图提供了圆括号嵌套示例。




```
LD      %I0.0
AND(    %I0.1
OR(N    %I0.2
AND     %M3
)
)
ST      %Q0.0
```




```
LD      %I0.1
AND(    %I0.2
AND     %I0.3
OR(     %I0.5
AND     %I0.6
)
AND     %I0.4
OR(     %I0.7
AND     %I0.8
)
)
ST      %Q0.0
```

堆栈指令 （MPS， MRD， MPP）

介绍

堆栈指令处理与线圈相关的线路。 MPS， MRD，和 MPP 指令使用一个临时存储区即堆栈，可以存放最多八个布尔表达式。

注意：这些指令不能用于圆括号内的表达式。

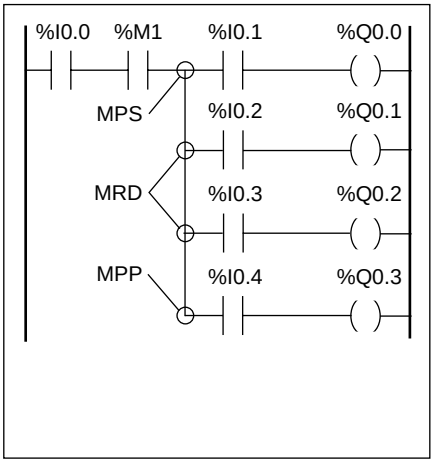
堆栈指令操作

下表描述了三个堆栈指令的操作。

指令	描述	功能
MPS	存储压入堆栈	存储最近一次逻辑指令的结果 （累加器的内容）到堆栈的顶部 （压入），并使堆栈中其它值向堆栈底部移动一格。
MRD	从堆栈读取存储	将堆栈顶部值读入累加器。
MPP	从堆栈取出存储	拷贝堆栈顶部值到累加器，并将堆栈内其它值向顶部移动一格。

堆栈指令示例

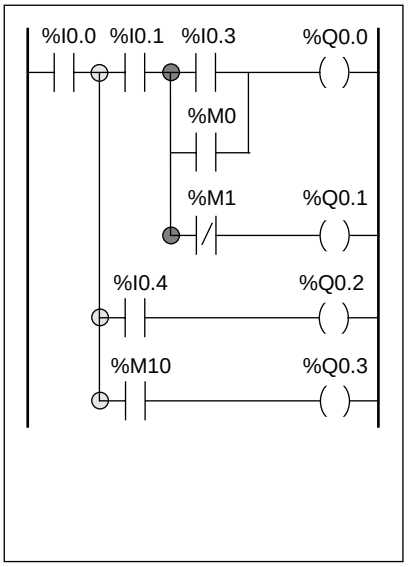
下图是堆栈指令使用示例。




```
LD      %I0.0
AND     %M1
MPS
AND     %I0.1
ST      %Q0.0
MRD
AND     %I0.2
ST      %Q0.1
MRD
AND     %I0.3
ST      %Q0.2
MPP
AND     %I0.4
ST      %Q0.3
```

堆栈操作示例

下图显式了堆栈指令怎样进行操作。




```
LD      %I0.0
MPS
AND     %I0.1
MPS
AND(    %I0.3
OR      %M0
)
ST      %Q0.0
MPP
ANDN    %M1
ST      %Q0.1
MRD
AND     %I0.4
ST      %Q0.2
MPP
AND     %M10
ST      %Q0.3
```

Grafcet

15

概览

本章主题

本章描述了怎样使用 Grafcet 语言编程。

本章包含了哪些内容?

本章包含了以下主题：

Grafcet 指令描述

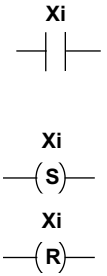
介绍

TwidoSoft 中 Grafcet 指令提供了翻译控制顺序的一个简单方法 （Grafcet 表）。Grafcet 的最大步数取决于 Twido 控制器的型号。任何时刻活动步的数目仅由步的总数目所限制。

对于 TWDLCAA10DRF 和 TWDLCAA16DRF，可使用步 1 到 62。步 0 和 63 保留为前处理和后处理。对所有其它控制器，可使用步 1 到 95。

Grafcet 指令

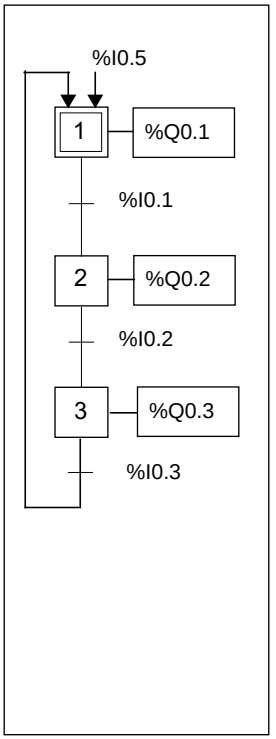
下表列出了 Grafcet 表编程所需的所有指令和对象：

图形表示 （1）	TwidoSoft 语言抄本	功能
图例：  	=*= i	开始初始步 （2）
	# i	在停止当前步后激活步 i
	-*- i	开始步 i 并使相关转换有效 （2）
	#	停止当前步并不激活其它任何步
	#Di	停止步 i 和当前步
	=*= POST	开始后处理并结束顺序处理
	%Xi	步 i 的相关位，可以被测试和被写 （步的最大数目取决于控制器）
	LD %Xi, LDN %Xi AND %Xi, ANDN %Xi, OR %Xi, ORN %Xi XOR %Xi, XORN %Xi	测试步 i 的活动性
	S %Xi	激活步 i
 	R %Xi	停止步 i

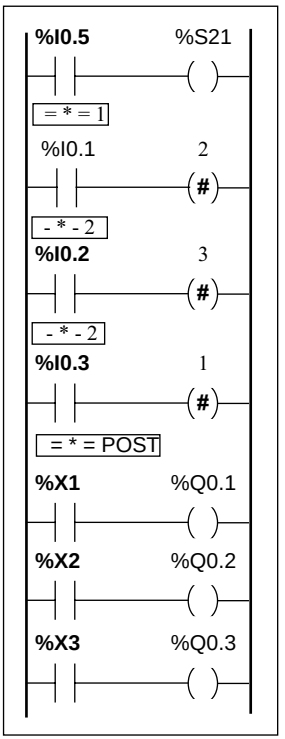
(1) 不考虑图形表示。
(2) 被写的第一步 =*=i 或 -*-i 表示顺序处理开始及预处理结束。

Grafcet 示例

线性顺序：



不支持

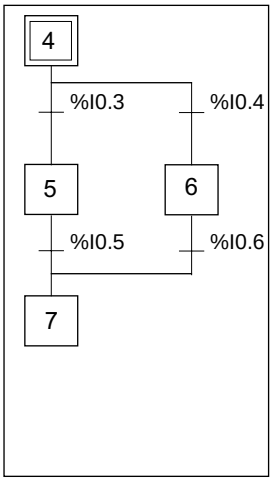


Twido 梯形图
语言程序

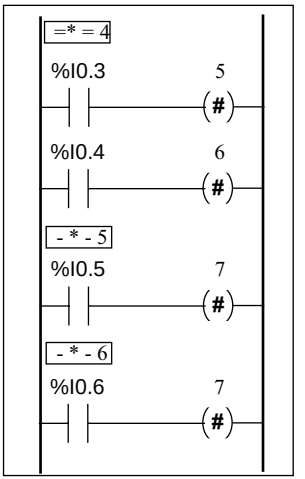
```
LD      %I0.5
ST      %S21
=*=     1
LD      %I0.1
 #      2
-*-     2
LD      %I0.2
 #      3
-*-     3
LD      %I0.3
 #      1
=*= POST
LD      %X1
ST      %Q0.1
LD      %X2
ST      %Q0.2
LD      %X3
ST      %Q0.3
```

Twido 指令
列表程序

并列顺序：



不支持

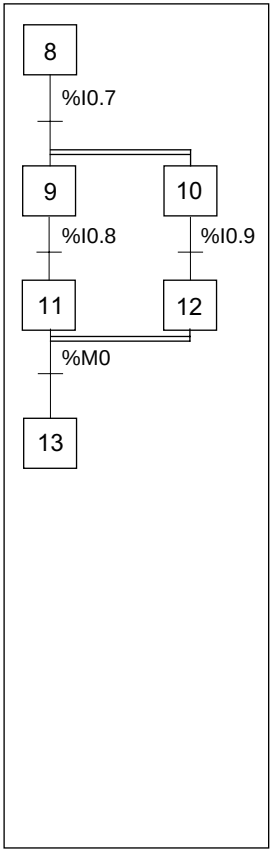


Twido 梯形图
语言程序

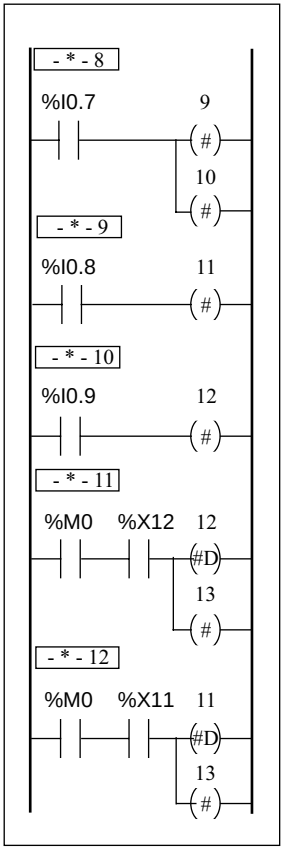
```
=*=     4
 LD     %I0.3
 #      5
 LD     %I0.4
 #      6

 -*-    5
 LD     %I0.5
 #      7

-*-     6
 LD     %I0.6
 #      7
```

Twido 指令
列表程序

同步顺序：



不支持



Twido 梯形图
语言程序

```
 -*-      8
 LD       %I0.7
 #        9
 #        10

-*-       9
 LD       %I0.8
 #        11

-*-       10
 LD       %I0.9
 #        12

-*-       11
 LD       %M0
 AND      %X12
 #D       12
 #        13

-*-       12
 LD       %M0
 AND      %X11
 #D       11
 #        13
```

Twido 指令
列表程序

注意：对于将要运行的 Grafcet 表，必须用 =*=i 指令（初始步）声明至少一个活动步或在处理过程中使用系统位 %S23 和指令 S %Xi 预置表。

Grafcet 程序结构描述

介绍

一个 TwidoSoft Grafcet 程序包含三部分：

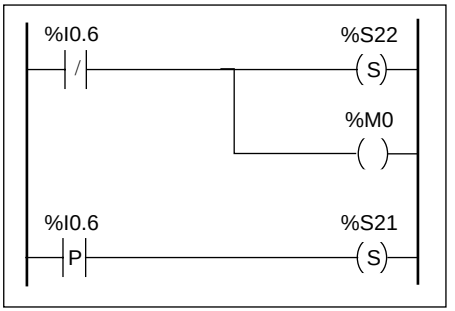
- 预处理
- 顺序处理
- 后处理

预处理

预处理由以下组成：

- 电源恢复
- 故障检查
- 修改工作模式
- 预置 Grafcet 步
- 输入逻辑

输入 %I0.6 的上升沿将位 %S21 置为 1。它停止活动步并激活非活动步。




```
000   LDN   %I0.6
001   S     %S22
002   ST    %M0
003   LDR   %I0.6
004   S     %S21
```

预处理开始于程序的第一行，结束于 "= * =" 或 "- * -" 指令的第一次出现。

三个系统位专用于 Grafcet 控制：%S21, %S22 和 %S23。它们通常在预处理时由程序置为 1 （如果需要）。预处理结束时系统执行其相应功能并将这些系统位复置到 0。

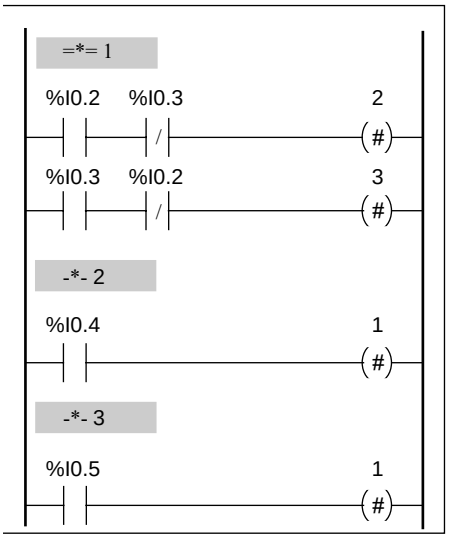
系统位	名字	描述
%S21	Grafcet 初始化	停止所有活动步并激活初始步。
%S22	Grafcet 重新初始化	停止所有步。
%S23	Grafcet 预置	如果对象 %Xi 在预处理时被应用程序明确所写，则该位必须被置为 1。如果该位由预处理保持为 1 且对象 %Xi 没有任何变化，则 Grafcet 被冻结 （不考虑更新）。

顺序处理

顺序处理在表中发生 （指令表示表）：

- 步
- 与步有关的动作
- 转换
- 转换条件

示例：




```
005    =*=     1
006    LD      %I0.2
007    ANDN    %I0.3
008    #       2
009    LD      %I0.3
010    ANDN    %I0.2
011    #       3
012    -*-     2
013    LD      %I0.4
014    #       1
015    -*-     3
016    LD      %I0.5
017    #       1
```

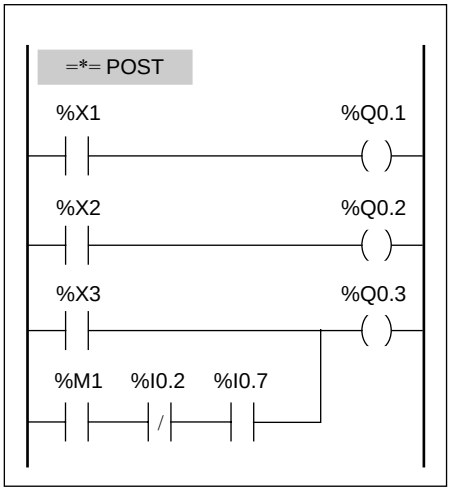
顺序处理结束于指令 "= * = POST" 的执行或程序结束。

后处理

后处理由如下组成：

- 来自顺序处理用于控制输出的命令
- 输出特定安全互锁

示例：




```
018   =*=    POST
019   LD     %X1
020   ST     %Q0.1
021   LD     %X2
022   ST     %Q0.2
023   LD     %X3
024   OR(    %M1
025   ANDN   %I0.2
026   AND    %I0.7
027   )
028   ST     %Q0.3
```

与 Grafcet 步相关的动作

介绍

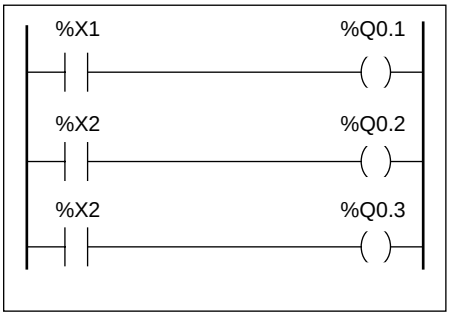
TwidoSoft Grafcet 程序提供了两种方法对与步相关的动作进行编程：

- 在后处理部分
- 在步自己的列表指令或梯级

在后处理中关联动作

如果存在安全或运行模式限制，最好在 Grafcet 应用程序的后处理部分对动作进行编程。您能通过 Set 和 Reset 列表指令或在梯形图程序中激活线圈来激活 Grafcet 步 （%Xi）。

示例：

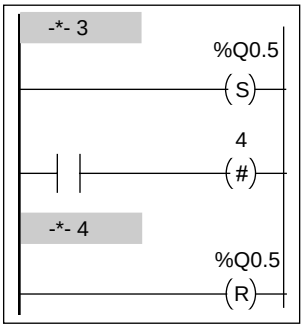



```
018   =*=    POST
019   LD     %X1
020   ST     %Q0.1
021   LD     %X2
022   ST     %Q0.2
023   LD     %X3
024   ST     %Q0.3
```

从应用程序关联动作

您能在列表指令或梯级中对与步相关的动作进行编程。在这种情况下，列表指令或梯级不被扫描除非步处于活动状态。这是更为有效，可读，和可维持的使用 Grafcet 的方式。

示例：




```
020   -*-     3
021   LD      1
022   S       %Q0.5
023   LD      %M10
024   #       4
025   -*-     4
026   LD      1
027   R       %Q0.5
028   ...
029   ...
```

指令和功能描述

IV

概览

本部分的主题

本部分对 Twido 语言的基本和高级指令，以及系统位和字进行了详细描述。

本部分包含了哪些内容？

本部分包含了以下章节：

基本指令

16

概览

本章的主题

本章提供了有关于创建 Twido 控制器基本控制程序的指令和功能模块的详细资料。

本章包含了那些内容？

本章包含了以下几节：

16.1 布尔运算

概览

本章的主题

本节提供了布尔运算介绍，包括布尔指令描述和编程指导。

本节包含了哪些内容?

本节包含了以下几题：

布尔指令

介绍

布尔指令可与梯形图语言元素相比较。这些指令归纳如下表所示。

条目	指令	示例	描述
测试元素	装载 （LD）指令等价于常开触点。	LD %I0.0	当位 %I0.0 为 1 时触点闭合。
动作元素	存储 (ST) 指令等价于线圈。	ST %Q0.0	相关位对象取位累加器的逻辑值 （前面逻辑运算的结果）。

测试元素的布尔运算结果应用于动作元素，如下面图例所示。

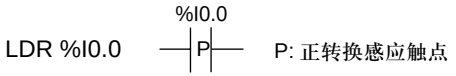
```
LD    %I0.0
AND   %I0.1
ST    %Q0.0
```

控制器输入测试

布尔测试指令可用于检测控制器输入的上升或下降沿。沿在 “第 n-1 次扫描”时的输入状态与当前 “第 n 次扫描时”的输入状态不同时被检测到。沿在当前扫描中保持检测值不变。

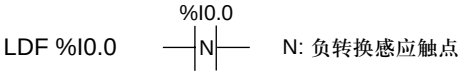
上升沿检测

LDR 指令 （装载上升沿）等价于上升沿检测触点。上升沿检测输入值从 0 到 1 的变化。
正变换传感触点用于检测上升沿，如下图所示。



下降沿检测

LDF 指令 （装载下降沿）等价于下降沿检测触点。下降沿检测输入值从 1 到 0 的变化。
负变换传感触点用于检测上升沿，如下图所示。



边沿检测

下表归纳了边沿检测的指令和时序：

边沿	测试指令	梯形图	时序图
上升沿	LDR %I0.0	%I0.0 P	上升沿 %I0.2 时间 T 布尔型结果 T=1 控制器扫描周期 时间
下降沿	LDF %I0.0	%I0.0 N	下降沿 %I0.2 时间 T 布尔型结果 T=1 控制器扫描周期 时间

注意：可将边沿指令应用于 %Mi 内部位。

布尔指令描述格式了解

介绍

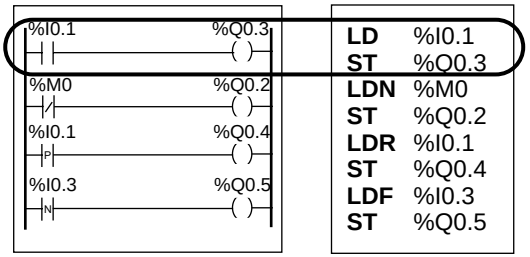
本节中的每个布尔指令通过下面几个方面的信息进行描述：

- 简述
- 指令示例及对应梯形图
- 允许操作数列表
- 时序图

下面解释对本节怎样描述布尔指令提供了更为详细的说明。

示例

下面图例显示了每个指令的示例是怎样给出的。



梯形图对等　　列表指令

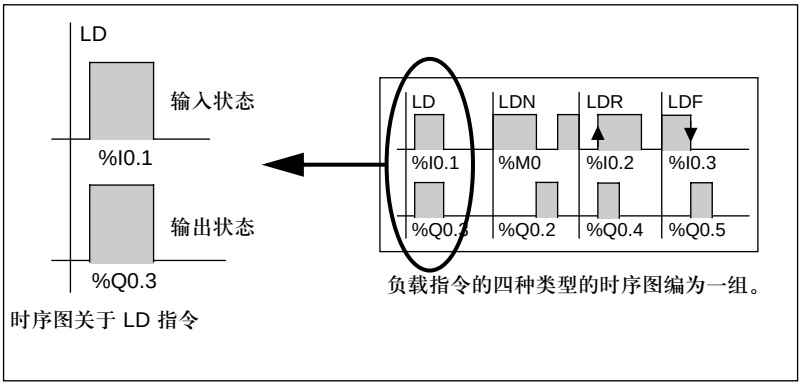
允许操作数

下表定义了用于布尔指令的允许操作数类型。

操作数	描述
0/1	立即值 0 或 1
%I	控制器输入 %Ii.j
%Q	控制器输出 %Qi.j
%M	内部位 %M
%S	系统位 %Si
%X	步位 %Xi
%BLK.x	功能模块位 （例如， %TMi.Q）
%•:Xk	字位 （例如， %MWi:Xk）
[	比较表达式 （例如， [%MWi<1000]）

时序图

下面图例显示了每个指令的时序图是怎样显示的。



时序图关于 LD 指令

负载指令的四种类型的时序图编为一组。

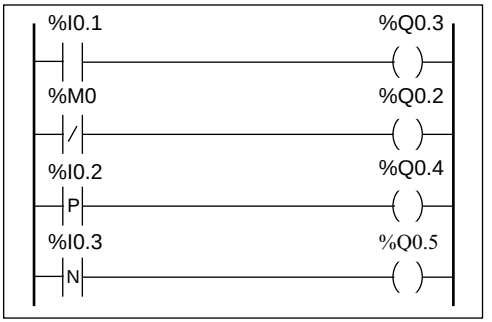
装载指令 （LD， LDN， LDR， LDF）

介绍

装载指令 LD， LDN， LDR 和 LDF 分别对应于常开，常闭，上升沿和下降沿触点（LDR 和 LDF 只能用于控制器输入和内部字， AS-I 和 PDO CANopen 从站输入）。

示例

下图是装载指令示例。




```
LD    %I0.1
ST    %Q0.3
LDN   %M0
ST    %Q0.2
LDR   %I0.2
ST    %Q0.4
LDF   %I0.3
ST    %Q0.5
```

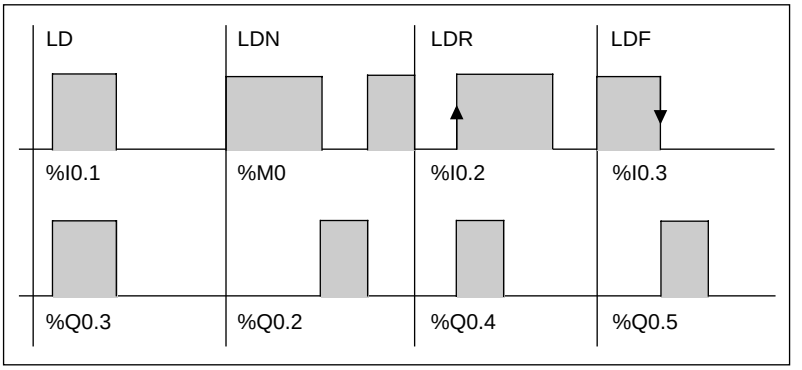
允许操作数

下表列出了装卸指令类型，等价梯形图及允许操作数。

列表指令	等价梯形图	允许操作数
LD		0/1, %I, %IA, %IWCx.y.z:Xk, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk,[
LDN		0/1, %I, %IA, %IWCx.y.z:Xk, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk,[
LDR		%I, %IA, %M
LDF		%I, %IA, %M

时序图

下图显示了装载指令的时序图。



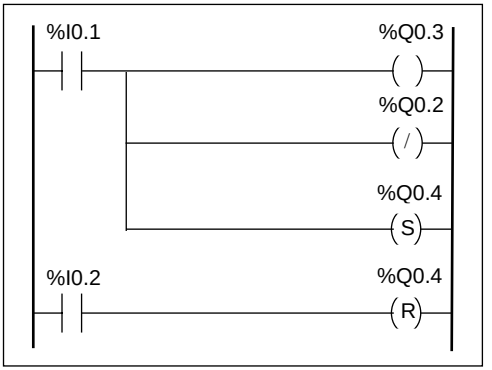
赋值指令 （ST， STN， R， S）

介绍

赋值指令 ST， STN， S 和 R 分别对应直接，反，置位，和复位线圈。

示例

下图是赋值指令示例。




```
LD      %I0.1
ST      %Q0.3

STN     %Q0.2
S       %Q0.4

LD      %I0.2
R       %Q0.4
```

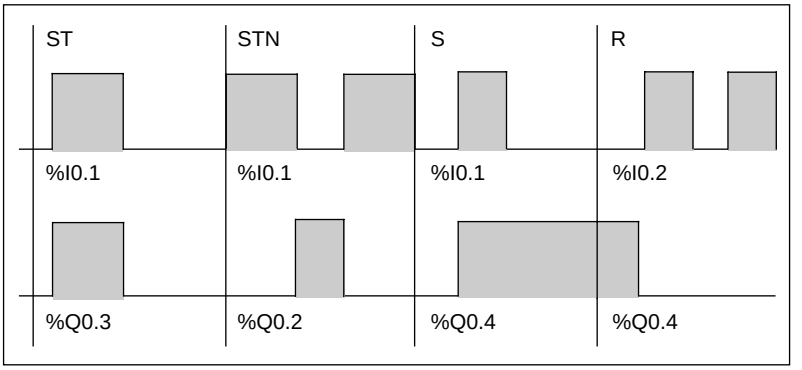
允许操作数

下表列出了赋值指令类型，等价梯形图及允许操作数。

列表指令	等价梯形图	允许操作数
ST	( )	%Q,%QA,%M,%S,%BLK.x,%•:Xk
STN	(/)	%Q,%QA%M,%S,%BLK.x,%•:Xk
S	(S)	%Q,%QA,%M,%S,%X,%BLK.x,%•:Xk
R	(R)	%Q,%QA,%M,%S,%X,%BLK.x,%•:Xk

时序图

下图显示了赋值指令的时序图。



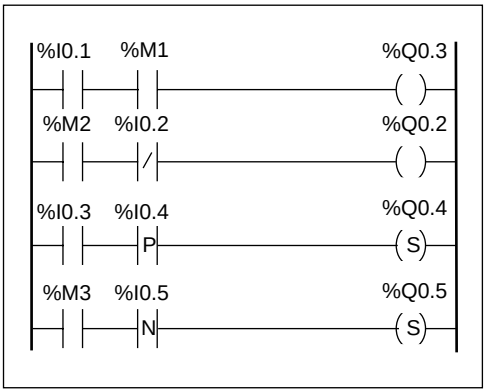
逻辑与指令 （AND， ANDN， ANDR， ANDF）

介绍

逻辑与指令执行操作数 （或它的反转，或上升沿，或下降沿）和前面指令的布尔运算结果间的逻辑与操作。

示例

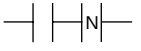
下图是逻辑与指令示例。




```
LD      %I0.1
AND     %M1
ST      %Q0.3
LD      %M2
ANDN    %I0.2
ST      %Q0.2
LD      %I0.3
ANDR    %I0.4
S       %Q0.4
LD      %M3
ANDF    %I0.5
S       %Q0.5
```

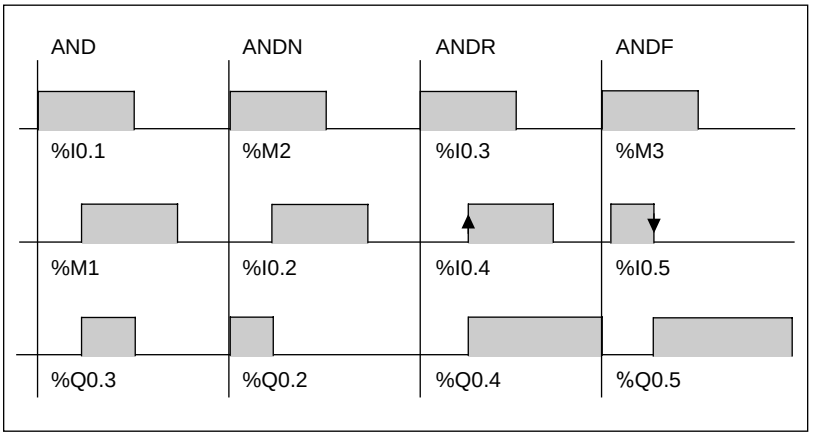
允许操作数

下表列出了逻辑与指令类型，等价梯形图及允许操作数。

列表指令	等价梯形图	允许操作数
AND		0/1, %I, %IA, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk, [
ANDN		0/1, %I, %IA, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk, [
ANDR		%I, %IA, %M
ANDF		%I, %IA, %M

时序图

下图显示了逻辑与指令的时序图。



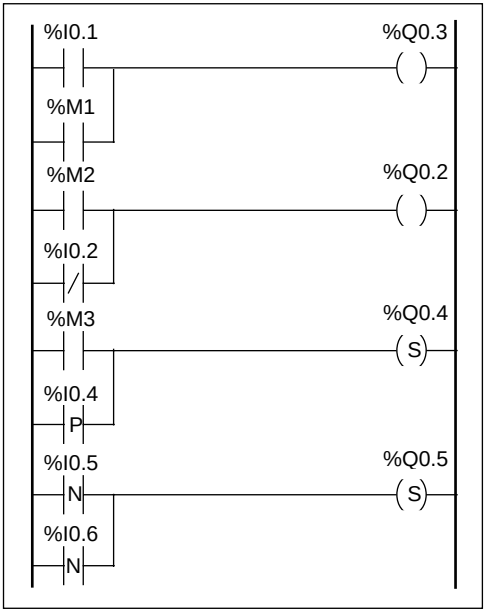
逻辑或指令 （OR， ORN， ORR， ORF）

介绍

逻辑或指令执行操作数 （或它的反转数，或上升沿，或下降沿）和前面指令的布尔运算结果间的逻辑或操作。

示例

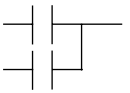
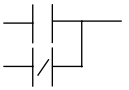
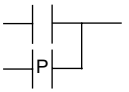
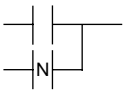
下图是逻辑与指令示例。




```
LD      %I0.1
OR      %M1
ST      %Q0.3

LD      %M2
ORN     %I0.2
ST      %Q0.2

LD      %M3
ORR     %I0.4
S       %Q0.4

LDF     %I0.5
ORF     %I0.6
S       %Q0.5
```

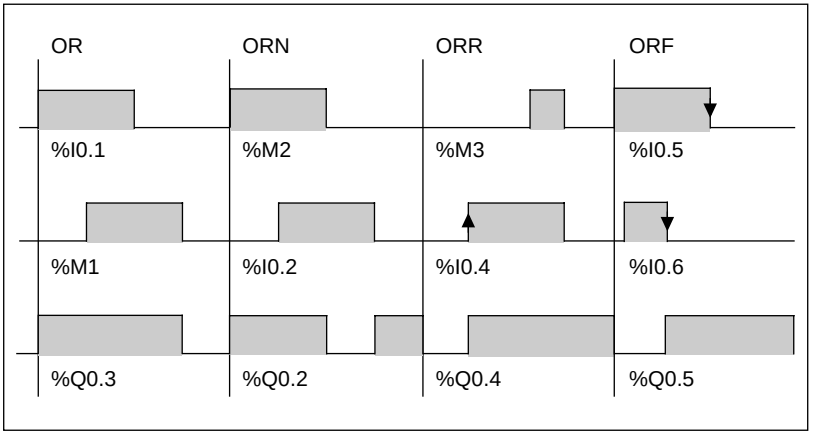
允许操作数

下表列出了逻辑或指令类型，等价梯形图及允许操作数。

列表指令	等价梯形图	允许操作数
OR		0/1, %I,%IA, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk
ORN		0/1, %I,%IA, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk
ORR		%I, %IA, %M
ORF		%I, %IA, %M

时序图

下图显示了逻辑或指令的时序图。



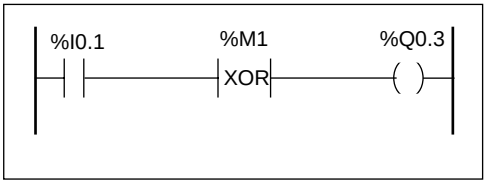
异或指令 （XOR， XORN， XORR， XORF）

介绍

异或指令执行操作数 （或它的反转数，或上升沿，或下降沿）和前面指令的布尔运算结果间的异或操作。

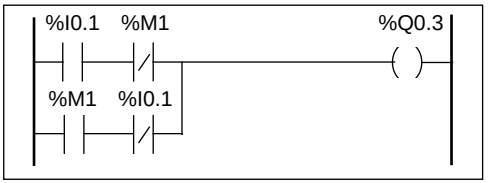
示例

下图是异或指令的使用示例。
使用 XOR 指令的示意图




```
LD      %I0.1
XOR     %M1
ST      %Q0.3
```

不使用 XOR 指令的示意图




```
LD      %I0.1
ANDN    %M1
OR(     %M1
ANDN    %I0.1
)
ST      %Q0.3
```

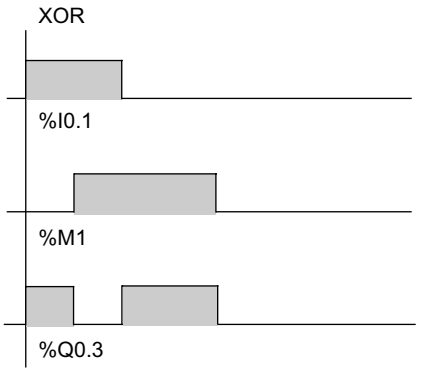
允许操作数

下表列出了异或指令类型及允许操作数。

列表指令	允许操作数
XOR	%I, %IA, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk
XORN	%I, %IA, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk
XORR	%I, %IA, %M
XORF	%I, %IA, %M

时序图

下图显示了异或指令的时序图。

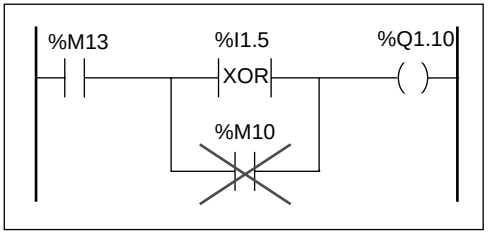


特殊情况

下面是对梯形图中使用异或指令的特别警告：

- 不要在梯级的第一个位置插入异或触点。
- 不要将异或触点与其它梯形图元素并行放置 （见下面示例）。

如下图所示，加入一个和异或触点相并行的元素将会产生确认错误。



取反指令 （N）

介绍

取反 （N）指令将前面指令的布尔运算结果取反。

示例

下图是取反指令使用示例。

```
LD      %I0.1
OR      %M2
ST      %Q0.2
N
AND     %M3
ST      %Q0.3
```

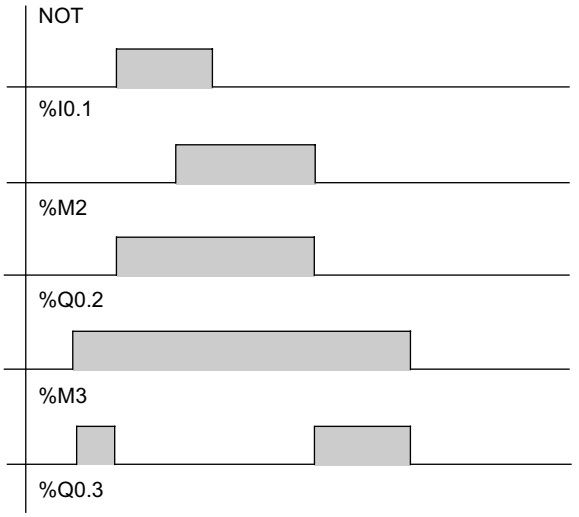
注意：取反指令不可逆。

允许操作数

不适用。

时序图

下图显示了取反指令的时序图。



16.2 基本功能模块

概览

本节的主题

本节提供了基本功能模块使用描述和编程指导。

本节包含了哪些内容?

本节包含了以下主题：

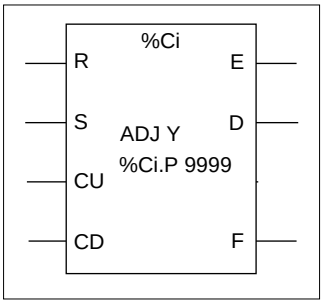
基本功能模块

介绍

功能模块是程序使用的位对象和特殊字的来源。基本功能模块提供简单功能，如定时器或加 / 减计数。

功能模块示例

下图是一个加 / 减计数器功能模块示例。



加 / 减计数器

位对象

位对象对应模块输出。布尔测试指令能用下面任一方法访问这些位：

- 直接方式 （例如， LD E），如果它们在可逆编程中与模块有线连接 （见标准功能模块编程原则 P.423）。
- 通过指定模块类型 （例如， LD %Ci.E）。

输入由指令表访问。

字对象

字对象对应指定的参数和值如下 :

- **模块配置参数：**一些参数能被程序访问 （如，预置参数），一些参数不能被程序访问 （如，时基）。
- **当前值：**例如， %Ci.V，当前计数值。

可访问位和字对象

下表描述了可被程序访问的基本功能模块位和字对象。

基本功能模块	符号	范围 (i)	对象类型	描述	地址	写访问
定时器	%TMi	0 - 127	字	当前值	%TMi.V	不可以
				预置值	%TMi.P	可以
			位	定时器输出	%TMi.Q	不可以
加 / 减计数器	%Ci	0 - 127	字	当前值	%Ci.V	不可以
				预置值	%Ci.P	可以
			位	下溢输出 （空）	%Ci.E	不可以
				预置输出达到	%Ci.D	不可以
				满溢输出 （满）	%Ci.F	不可以

标准功能模块编程原则

介绍

用下面方法之一对标准功能模块编程：

- 功能模块指令 （例如， BLK %TM2）：梯形图语言中这种可逆的编程方法使得功能模块操作可以在程序一个地方执行。
- 特殊指令 （例如， CU %Ci）：不可逆的编程方法使得功能模块输入操作可在程序几个地方执行 (例如，行 100 CU %C1, 行 174 CD %C1，行 209 LD %C1.D)。

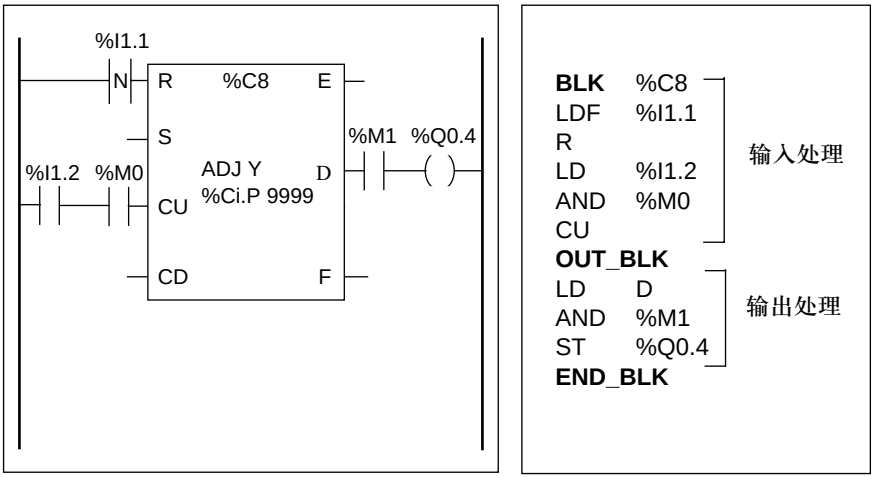
可逆编程

使用可逆编程指令 BLK， OUT_BLK 和 END_BLK：

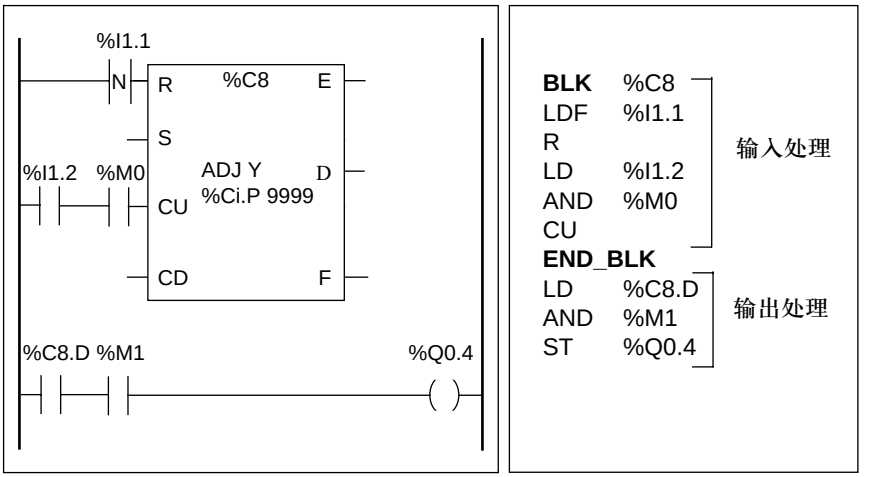
- **BLK:** 表示模块的开始。
- **OUT_BLK:** 用于与模块输出直接连线。
- **END_BLK:** 表示模块的结束。

输出连线示例

下面示例显示了带有输出连线的计数器功能模块的可逆编程。



输出不连线示例　　下面示例显示了不带有输出连线的计数器功能模块的可逆编程。



注意：只有相关模块中的测试和输入指令可以放在 BLK 和 OUT_BLK 指令之间（如果程序中没有 OUT_BLK 就放在 BLK 和 END_BLK 之间）。

定时器功能模块 （%TMi）

介绍

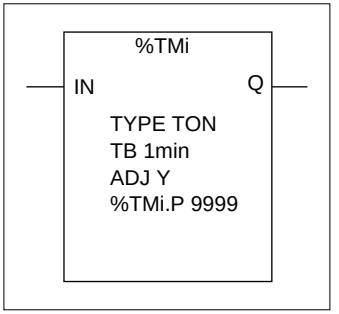
有三种定时器功能模块：

- TON （定时器导通－延时）：这种定时器用于控制导通－延时动作。
- TOF （定时器关断－延时）：这种定时器用于控制关断－延时动作。
- TP （定时器 - 脉冲）：这种定时器用于产生精确宽度的脉冲。

延时或脉冲周期可编程，并且可使用 TwidoSoft 进行修改。

图例

下面是定时器功能模块图例。



定时器功能模块

参数

定时器具有如下参数：

参数	标识	值
定时器编号	%TMi	0 到 63：TWDLCAA10DRF 和 TWDLCAA16DRF 0 到 127 对所有其它控制器。
类型	TON	• 定时器导通 – 延时 （默认）
	TOF	• 定时器关断 – 延时
	TP	• 脉冲 （单稳态）
时基	TB	1min （默认）， 1s， 100ms， 10ms， 1ms
当前值	%TMi.V	当定时器工作时，该字从 0 增加到 %TMi.P。可被程序读和测试，但不可写。 %TMi.V 可以通过活动表编辑器修改。
预置值	%TMi.P	0 - 9999. 该字可读，测试和被写，默认值是 9999。周期或产生的延时为 %TMi.P x TB。
动态监控表编辑器	Y/N	Y：Yes，预置 %TMi.P 值可以通过活动表编辑器修改。 N：No，预置 %TMi.P 值不能通过活动表编辑器被修改。
输入使能（或指令）	IN	上升沿 （TON 或 TP 类型）或下降沿 （TOF 类型）启动定时器。
定时器输出	Q	根据执行功能的类型，相关位 %TMi.Q 置为 1：TON，TOF 或 TP

注意：预置值越大，定时器的精度越高。

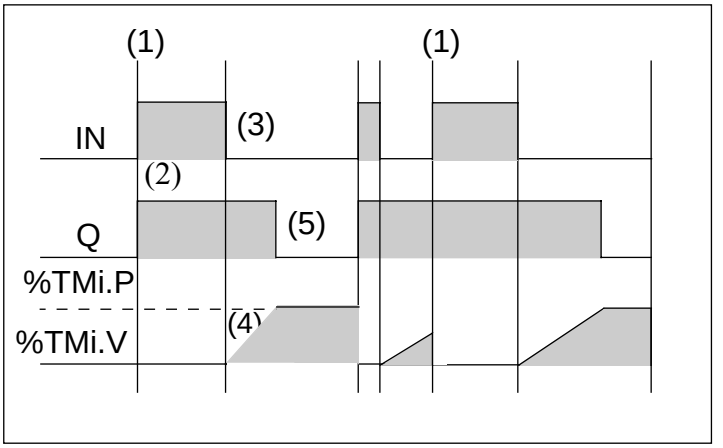
TOF 类型定时器

介绍

使用 TOF （定时器关断 – 延时）类型定时器控制关断 – 延时动作。延时可用 TwidoSoft 编程。

时序图

下面是 TOF 类型定时器操作时序图。



操作

下表描述了 TOF 类型定时器的操作。

阶段	描述
1	当前值 %TMi.V 也会被输入 IN 的上升沿置为 0，。
2	检测到输入 IN 的上升沿时， %TMi.Q 输出位置为 1。
3	定时器在输入 IN 的下降沿开始工作。
4	当前值 %TMi.V 以时基 TB 的每个脉冲为一个增量，增加到 %TMi.P。
5	当前值到达 %TMi.P 时， %TMi.Q 输出位置为 0。

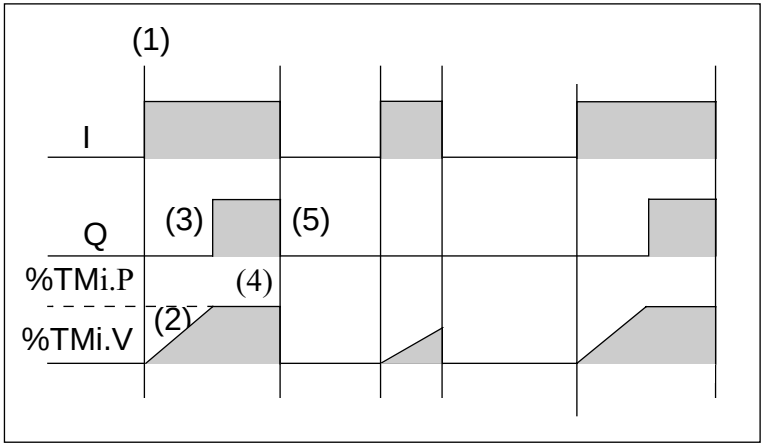
TON 类型定时器

介绍

使用 TON （定时器导通 – 延时）类型定时器控制导通 – 延时动作。延时可用 TwidoSoft 编程。

时序图

下面是 TON 类型定时器操作时序图。



操作

下表描述了 TON 类型定时器的操作。

阶段	描述
1	定时器在输入 IN 的上升沿开始工作。
2	当前值 %TMi.V 以时基 TB 的每个脉冲为一个单位增量，从 0 增加到 %TMi.P。
3	当前值到达 %TMi.P 时 %TMi.Q 输出位置为 1。
4	当输入 IN 为 1 时 %TMi.Q 输出位保持为 1。
5	当检测到输入 IN 的下降沿，即使定时器未到达 %TMi.P，定时器停止， %TMi.V 置为 0。

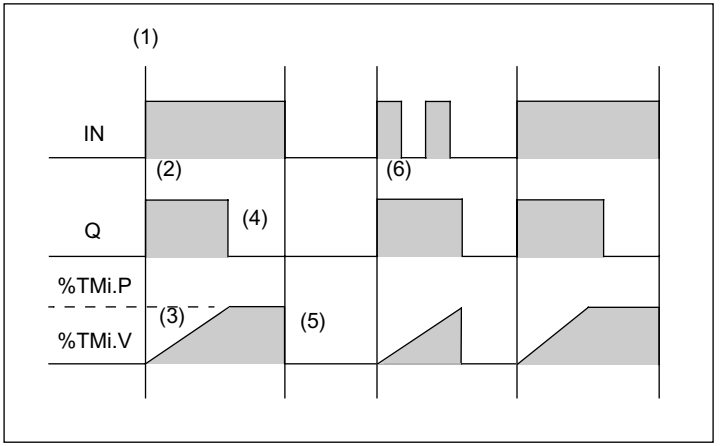
TP 类型定时器

介绍

使用 TP （定时器 – 脉冲）类型定时器用于产生具有精确宽度的脉冲。延时可用 TwidoSoft 编程。

时序图

下面是 TP 类型定时器操作时序图。



操作

下表描述了 TP 类型定时器的操作。

阶段	描述
1	定时器在输入 IN 的上升沿开始工作。如果定时器还没开始则当前值 %TMi.V 置为 0。
2	当定时器开始时 %TMi.Q 输出位置为 1。
3	当前值 %TMi.V 以时基 TB 的每个脉冲为单位增量，从 0 增加到 %TMi.P。
4	当前值到达 %TMi.P 时 %TMi.Q 输出位置为 0。
5	当 %TMi.V 等于 %TMi.P 且输入 IN 回到 0 时，当前值 %TMi.V 置为 0。
6	定时器不能被复位。一旦 %TMi.V 等于 %TMi.P 且输入 IN 为 0，则 %TMi.V 置为 0。

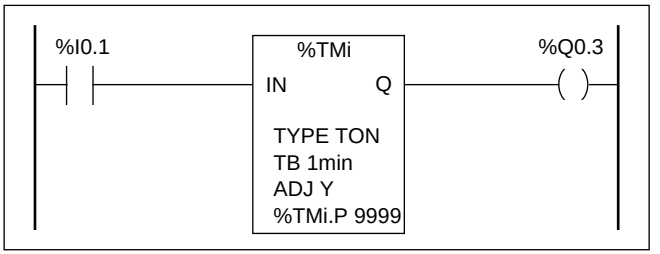
定时器编程和配置

介绍

不管定时器功能模块 (%TMi) 用途如何，它们的编程方法相同。定时器功能 (TON, TOF, 或 TP) 在配置中选定。

例子

下图是定时器功能模块可逆和不可逆编程示例。



可逆编程

```
BLK     %TM1
LD      %I0.1
IN
OUT_BLK
LD      Q
ST      %Q0.3
END_BLK
```

不可逆编程

```
LD      %I0.1
IN      %TM1
LD      %TM1.Q
ST      %Q0.3
```

配置

下面参数必须在配置中输入：

- 定时器类型：TON， TOF，或 TP
- 时基：1min， 1s， 100ms， 10ms 或 1ms
- 预置值 (%TMi.P)：0 到 9999
- 可调节：复选或不复选

特殊情况

下表包含了使用定时器功能块的一些特殊情况。

特殊情况	描述
冷启动 （%S0=1）的影响	强制当前值为 0。输出 %TMi.Q 置为 0。预置值复位到配置中的定义值。
热启动 （%S1=1）的影响	对定时器的当前值和预置值都没有影响。电源断电时当前值不变。
控制器停止的影响	停止控制器不会冻结当前值
程序跳转的影响	跳过定时器模块并不冻结该定时器。定时器将继续增加直到达到预置值 %TMi.P）。此时，定时器模块赋值给输出 Q 的完成位 （%TMi.Q）改变状态。然而，直接连线到模块输出的相关输出不被激活，并不被控制器扫描。
位 %TMi.Q （完成位）测试	程序中最好只进行一次位 %TMi.Q 测试。
修改预置值 %TMi.P 的影响	只有当定时器被重新激活时，通过指令修改或调节的预置值才起作用。

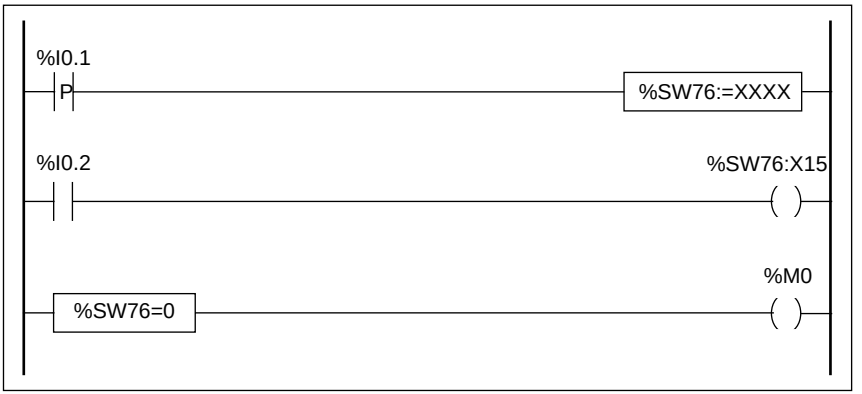
1 ms 时基定时器

1 ms 时基只适用于前五个定时器。四个系统字 %SW76， %SW77， %SW78，和 %SW79 可用作 “沙漏”。则系统对这四个字分别以毫秒为单位进行递减，如果它们具有正值。
通过连续装载这些字中的一个或测试中间值可以得到多重定时。如果这四个字中一个值小于 0，它将不会被修改。定时器可以通过设置对应的第 15 位为 1 “被冻结”，然后通过将其复位到 0 “解冻”。

编程示例

下面是定时器功能模块的编程示例。

```
LDR     %I0.1           （在 %I0.1 的上升沿启动定时器）
[%SW76:=XXXX]           （XXXX= 需要值）
LD      %I0.2           （冻结的可选管理，输入 I0.2 冻结）
ST      %SW76:X15
LD      [%SW76=0]       （定时器结束测试）
ST      %M0
.....
```



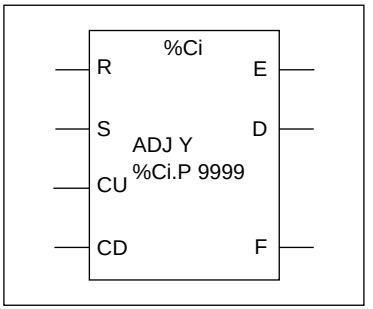
加 / 减计数器功能模块 （%Ci）

介绍

计数器功能模块 （%Ci）提供事件的加和减计数。这两种运算可以同时进行。

图例

下面是加 / 减计数器功能模块图例。



加减计数器功能模块

参数

寄存器功能模块具有如下参数：

参数	标识	值
计数器编号	%Ci	0 到 127
当前值	%Ci.V	字根据输入 （或指令） CU 和 CD 被增加或减少。可被程序读和测试，但不可写。使用数据编辑器修改 %Ci.V。
预置值	%Ci.P	0 ⩽ %Ci.P ⩽ 9999 能被读、测试和写（默认值：9999）。
用活动表编辑器编辑	ADJ	● Y：Yes，预置值可以通过活动表编辑器修改。 ● N：No，预置值不能使用活动表编辑器修改。
输入 （或指令）复位	R	状态为 1：%Ci.V = 0.
输入 （或指令）预置	S	状态为 1：%Ci.V = %Ci.P.
加运算输入（或指令）	CU	在上升沿增加 %Ci.V。
减运算输入（或指令）	CD	在上升沿减少 %Ci.V。
减运算溢出输出	E (Empty)	当减计数器 %Ci.V 从 0 变到 9999 时，相关 %Ci.E=1（当 %Ci.V 到达 9999 时置为 1，如果计数器继续减少则复位为 0）。
预置输出达到	D （完成）	当 %Ci.V=%Ci.P 时，相关位 %Ci.D=1。
加运算溢出输出	F (Full)	当 %Ci.V 从 9999 变到 0 时，相关位 %Ci.F=1（当 %Ci.V 到达 0 时置为 1，如果计数器继续增加则复位为 0）。

操作

下表描述了加 / 减计数器操作的主要过程。

操作	动作	结果
加计数	加计数输入 CU 出现上升沿（或指令 CU 被激活）。	当前值 %Ci.V 加 1。
	当前值 %Ci.V 等于预置值 %Ci.P。	“预置达到”输出位 %Ci.D 变为 1。
	当前值 %Ci.V 从 9999 变为 0。	输出位 %Ci.F （加计数溢出）变为 1。
	如果计数器继续增加。	输出位 %Ci.F （加计数溢出）复位到 0。
减计数	减计数输入 CD 出现上升沿（或指令 CD 被激活）。	当前值 %Ci.V 减 1。
	当前值 %Ci.V 从 0 变为 9999。	输出位 %Ci.E （减计数溢出）变为 1。
	如果计数器继续减少。	输出位 %Ci.E （减计数溢出）复位到 0。
加 / 减计数	要同时使用加计数和减计数功能 （或同时激活指令 CD 和 CU），必须同时控制两个对应的输入 CU 和 CD。这两个输入被连续扫描。如果它们都为 1，则当前值保持不变。	
复位	输入 R 置为状态 1 （或者指令 R 被激活）。	当前值 %Ci.V 被强制为 0。输出 %Ci.E。 %Ci.D 和 %Ci.F 置为 0。复位输入优先。
预置	如果输入 S 置为 1 （或指令 S 被激活）且复位输入为 0 （或指令 R 未被激活）。	当前值 %Ci.V 取 %Ci.P 值且输出 %Ci.D 置为 1。

特殊情况

下表显示了计数器特殊操作 / 配置情况列表。

特殊情况	描述
冷重启 （%S0=1）的影响	● 当前值 %Ci.V 置为 0。 ● 输出位 %Ci.E，%Ci.D，和 %Ci.F 置为 0。 ● 预置值根据配置定义值初始化。
控制器停止后热重启 （%S1=1）的影响	对计数器的当前值 （%Ci.V）没有影响。
修改预置 %Ci.P 的影响	当模块被应用程序处理 （一个输入被激活）时，通过指令或调节来修改预置值会产生影响。

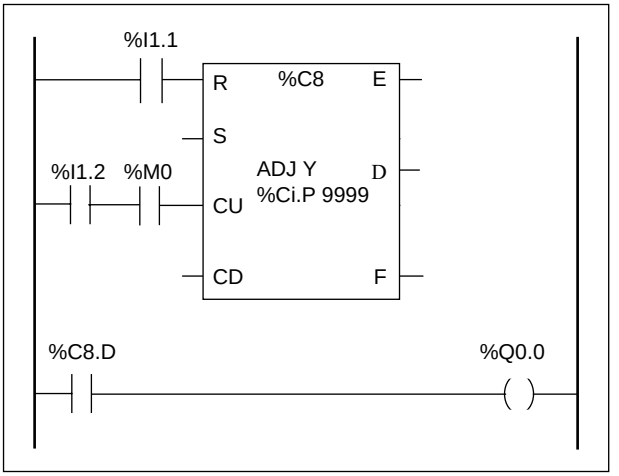
计数器编程和配置

介绍

下面示例是一个提供高达 5000 条计数的计数器。输入 %I1.2 的每个脉冲 （当内部位 %M0 置为 1 时）都使计数器 %C8 增加，直至达到它的预置值 （位 %C8.D=1）。计数器的值由输入 %I1.1 复位。

编程示例

下图是计数器功能模块可逆编程和不可逆编程示例。



梯形图

```
BLK    %C8
LD     %I1.1
R
LD     %I1.2
AND    %M0
CU
END_BLK
LD     %C8.D
ST     %Q0.0
```

可逆编程

```
LD     %I1.1
R      %C8
LD     %I1.2
AND    %M0
CU     %C8
LD     %C8.D
ST     %Q0.0
```

不可逆编程

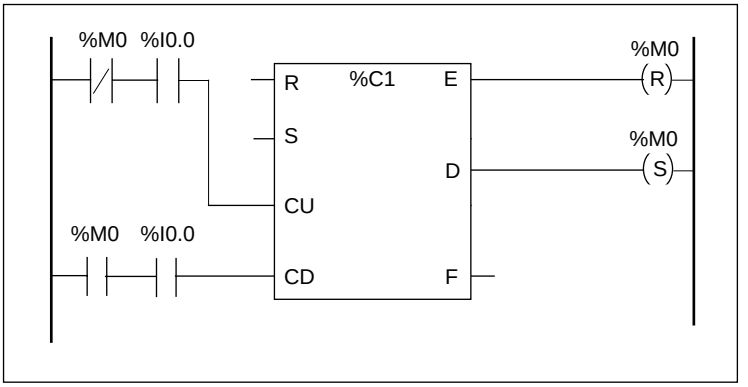
配置

下面参数必须在配置中输入：

- 预置值 (%Ci.P)：此例中设为 5000
- 可调节：是

加 / 减计数器示例

下图是一个加 / 减计数器功能模块示例。



梯形图

在此例中，若取 %C1.P = 4，%C1.V 当前值计数器将从 0 增加到 3，然后从 3 减至 0。每当 %I0.0 由 0 变为 1 时，%C1.V 也在 0 和 3 之间变化。

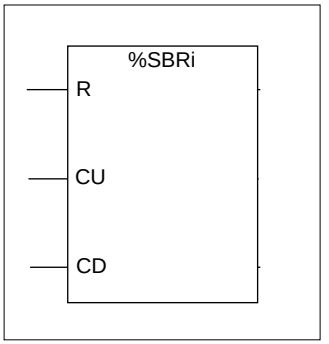
移位寄存器功能模块 （%SBRi）

介绍

移位寄存器功能模块 （%SBRi）提供了二进制数据位 （0 或 1）的左移或右移。

图例

下面是一个移位寄存器功能模块示例。



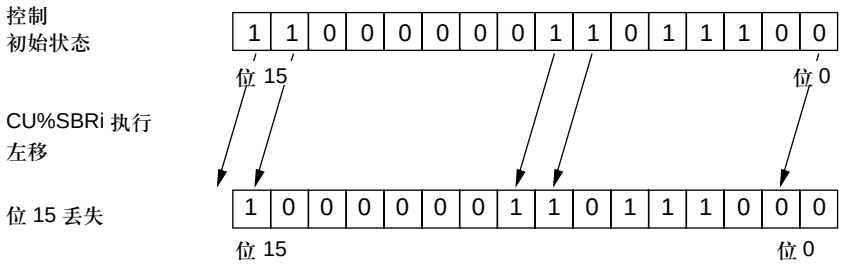
参数

移位寄存器功能模块具有下列参数。

参数	标识	值
寄存器编号	%SBRi	0 到 7
寄存器位	%SBRi.j	移位寄存器的位 0 到 15 （j = 0 到 15）可被测试指令测试，且由赋值指令写。
输入 （或指令）复位	R	当功能块参数 R 为 1 时，将设置寄存器位 0 到 15 %SBRi.j 为 0。
左移输入 （或指令）	CU	其上升沿将寄存器位左移一位。
右移输入 （或指令）	CD	其上升沿将寄存器位右移一位。

操作

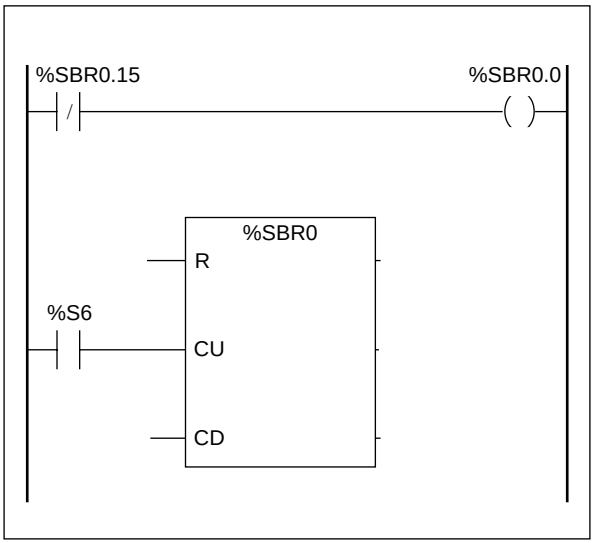
下图显示了移位操作前后的位形式。



使用指令 CD 右移一位（位 15 到位 0）也是如此。位 0 被丢失。如果一个 16 位寄存器不能满足要求，可以通过程序串连几个寄存器。

编程

下面示例中，每秒左移一位且假设位 0 与位 15 的状态相反。



可逆
编程

```
LDN    %SBR0.15
ST     %SBR0.0
BLK    %SBR0
LD     %S6
CU
END_BLK
```

不可逆
编程

```
LDN    %SBR0.15
ST     %SBR0.0
LD     %S6
CU     %SBR0
```

特殊情况

下表包含了移位寄存器功能模块编程的特殊情况列表。

特殊情况	描述
冷启动 （%S0=1）的影响	所有寄存器字的位被置为 0。
热启动 （%S1=1）的影响	对寄存器字的位没有影响。

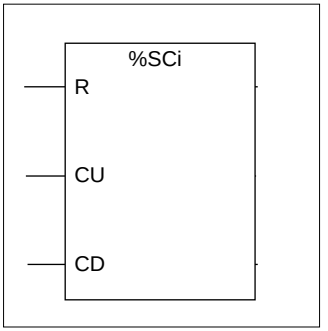
步进计数器功能模块 （%SCi）

介绍

步进计数器功能模块 （%SCi）提供了一系列的步，这些步可赋值给动作。从一个步移动到另一个步取决于外部或内部事件。每当一个步处于激活状态时，相关位被置为 1。步进计数器在一个时刻只能有一个步被激活。

图例

下面是一个步进计数器功能模块示例。



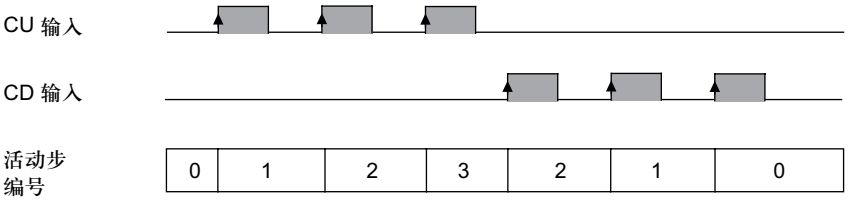
参数

步进计数器功能模块具有下列参数：

参数	标识	值
步进计数器编号	%SCi	0 - 7
步进计数器位	%SCi.j	步进计数器的位 0 到 225 （j = 0 到 225）可被装载逻辑测试，且由赋值指令写。
输入 （或指令）复位	R	当功能块参数 R 为 1 时，将复位步进计数器。
输入 （或指令）增加	CU	其上升沿将步进计数器增加一步。
输入 （或指令）减少	CD	其上升沿将步进计数器减少一步。

时序图

下面是步进计数器功能模块操作时序图。

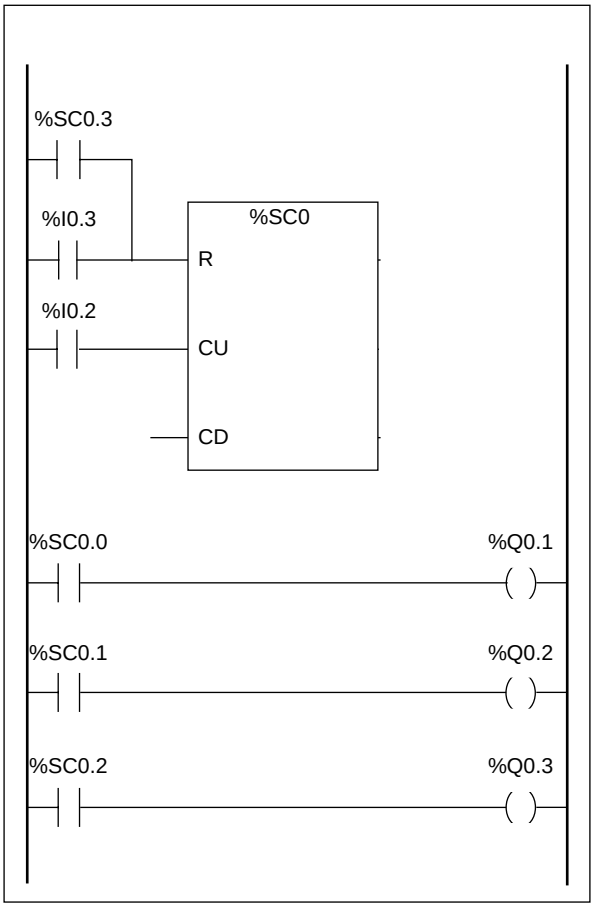


编程

下面是一个步进计数器功能模块示例。

- 步进计数器 0 由输入 %I0.2 增加。
- 步进计数器 0 由输入 %I0.3 或当它到达步 3 时复位到 0。
- 步 0 控制输出 %Q0.1，步 1 控制输出 %Q0.2，步 2 控制输出 %Q0.3。

下图显示了此例的可逆和不可逆编程。



可逆编程

```
BLK     %SC0
LD      %SC0.3
OR      %I0.3
R
LD      %I0.2
CU
END_BLK
LD      %SC0.0
ST      %Q0.1
LD      %SC0.1
ST      %Q0.2
LD      %SC0.2
ST      %Q0.3
```

不可逆编程

```
LD      %SC0.3
OR      %I0.3
R       %SC0
LD      %I0.2
CU      %SC0
LD      %SC0.0
ST      %Q0.1
LD      %SC0.1
ST      %Q0.2
LD      %SC0.2
ST      %Q0.3
```

特殊情况

下表包含了步进计数器功能模块操作的特殊情况列表。

特殊情况	描述
冷启动 （%S0=1）的影响	初始化步进计数器。
热启动 （%S1=1）的影响	对步进计数器没有影响。

16.3 数字运算

概览

本节的目标

本节提供了数字运算介绍，包括描述和编程指导。

本节包含了哪些内容?

本节包含了以下主题:

数字指令介绍

概览

数字指令一般用于 16 位字 （见*字对象*， *P.32*）和 32 位双字 （见*浮点和双字对象*，P.35）。它们写在方括号内。如果前面逻辑运算的结果为真 （布尔运算累加器 =1），则执行数字指令。如果前面逻辑运算的结果为假 （布尔运算累加器 =0），则不执行数字指令且操作数保持不变。

赋值指令

介绍

赋值指令用于把操作数 Op2 装入操作数 Op1。

赋值

赋值指令语法。

[Op1:=Op2] <=> Op2 -> Op1

赋值操作可用于：

- 位串
- 字
- 双字
- 浮点字
- 字表
- 双字表
- 浮点字表

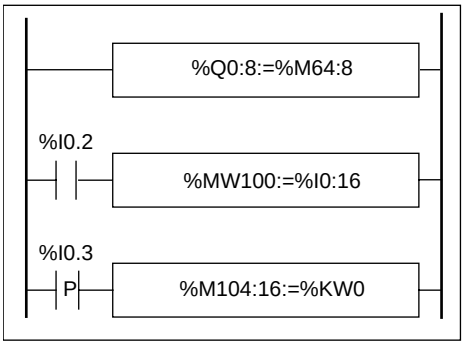
位串赋值

该操作可用于下列位串 （见*结构化对象* P.49）：

- 位串 -> 位串 （例 1）
- 位串 -> 字 （例 2）或双字 （索引）
- 字或双字 （索引） -> 位串 （例 3）
- 立即值 -> 位串

示例

位串赋值示例。




```
LD      1
[%Q0:8:=%M64:8]                    (Ex. 1)

LD      %I0.2
[%MW100:=%I0:16]                   (Ex. 2)

LDR     %I0.3
[%M104:16:=%KW0]                   (Ex. 3)
```

使用规则：

- 对位串 -> 字赋值：这些位串从右开始被传给字 （位串中的第一位赋值给字中的第 0 位），字中没有被传送的位 （字长≤ 16）被设为 0。
- 对字 -> 位串赋值：字中的位从右开始传送（字的第 0 位被传送到位串的第一位）。

位串赋值

位串赋值的语法。

运算符	语法	操作数 1 （Op1）	操作数 2 （Op2）
:=	[Op1: = Op2] 操作数 1(Op1) 为操作数 2 (Op2) 的值	%MWi,%QWi, %QWAi,%SWi %MWi[%MWi], %MDi, %MDi[%MWi] **%Mi:L, %Qi:L, %Si:L, %Xi:L**	立即值 , %MWi, %KWi, %IW,%IWAi, %INWi, %QWi, %QWAi %QNWi, %SWi, %BLK.x, %MWi[%MWi], %KWi[%MWi], %MDi[%MWi], %KDi[%MWi], **%Mi:L,%Qi:L, %Si:L, %Xi:L, %Ii:L**

注意：缩写 %BLK.x （例如， %C0.P）用来表示任意功能模块字。

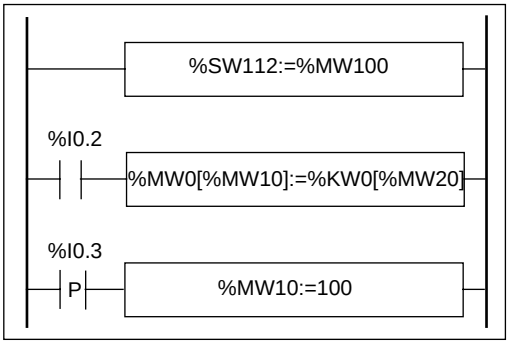
字赋值

赋值操作可用于下列字和双字：

- 字 （索引） -> 字 （例 2）或双字 （索引或非索引）
- 双字 （索引） -> 或双字 （索引或非索引）
- 立即全值 -> 字 （例 3）或双字 （索引或非索引）
- 位串 -> 字或双字
- 浮点 （索引或非索引） -> 浮点 （索引或非索引）
- 字或双字 -> 位串
- 立即浮点值 -> 浮点 （索引或非索引）

举例

字赋值示例。




```
LD      1
[%SW112:=%MW100]                    (Ex. 1)

LD      %I0.2
[%MW0[%MW10]:=                      (Ex. 2)
%KW0[%MW20]]

LDR     %I0.3                       (Ex. 3)
[%MW10:=100]
```

语法

字赋值语法。

运算符	语法
:=	[Op1: = Op2] 操作数 1 （Op1）为操作数 2 （Op2）的值

下表给出了详细操作数：

类型	操作数 1 (Op1)	操作数 2 （Op2）
字， 双字， 位串	**%BLK.x, %MWi, %QWi, %QWAi, %SWi %MWi[MWi**, %MDi, %MDi[%MWj]], %Mi:L, %Qi:L, %Si:L, %Xi:L	**Immediate value, %MWi, %KWi, %IW, %IWAi, %QWi, %QWAi, %SWi, %MWi[MWi], %KWi[MWi], %MDi, %MDi[%MWj], %KDi, %KDi[MWj]** , %INW, %Mi:L, %Qi:L, %QNW, %Si:L, %Xi:L, %Ii:L
浮点	**%MFi, %MFi[%MWj]**	立即浮点值， **%MFi, %MFi[%MWj], %KFi, %KFi[%MWj]**

注意：缩写 %BLK.x （例如， R3.I）用来表示任意功能模块字。对位串 %Mi:L，%Si:L，和 %Xi:L，位串的第一个基地址必须是 8 的倍数 （0, 8, 16, ..., 96, ...）。

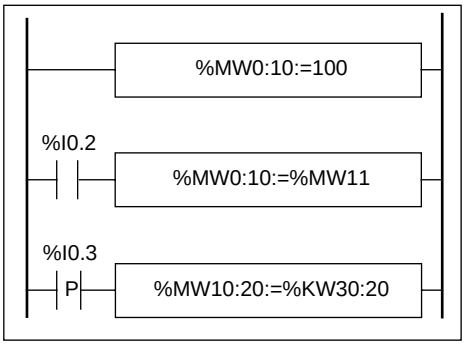
字，双字和浮点表赋值

赋值操作可用于下列对象表 （见字表 P.50）：

- 立即值 -> 字表 （例 1）或双字表
- 字 -> 字表 （例 2）
- 字表 -> 字表 （例 3）
  两个表的长度 （L）应该相同。
- 双字 -> 双字表
- 双字表 -> 双字表
  两个表的长度 （L）应该相同。
- 立即浮点值 -> 浮点表
- 浮点表 -> 浮点表
- 浮点表 -> 浮点表
  两个表的长度 （L）应该相同。

示例

字表赋值示例：




```
LD      1
[%MW0:10:=100]                    (Ex. 1)

LD      %I0.2
[%MW0:10:=%MW11]                  (Ex. 2)

LDR     %I0.3
[%MW10:20:=%KW30:20]              (Ex. 3)
```

语法

字，双字和浮点表赋值语法

运算符	语法
:=	[Op1: = Op2] 操作数 1 （Op1）为操作数 2 （Op2）的值

下表给出了详细操作数：

类型	操作数 1 （Op1）	操作数 2 （Op2）
字表	**%MWi:L, %SWi:L**	**%MWi:L, %SWi:L,** 立即全值， %MWi, %KWi, %IW, %QW, %IWA, %QWA, %SWi, %BLK.x
双字表	**%MDi:L**	立即全值，**%MDi, %KDi,%MDi:L,** **%KDi:L**
浮点字表	**%MFi:L]**	立即浮点值， **%MFi, %KFi, %MFi:L, %KFi:L**

注意：缩写 %BLK.x （例如， R3.I）用于描述任意功能模块字。

比较指令

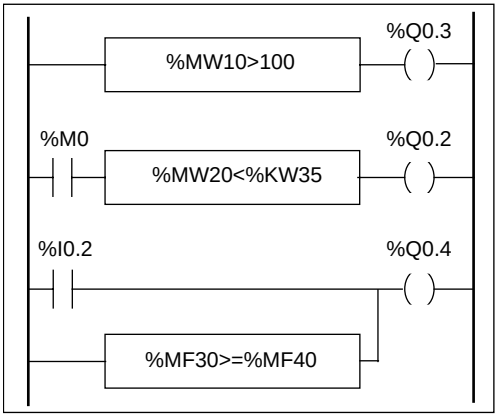
介绍

比较指令用于比较两个操作数。
下表列出了比较指令类型。

指令	功能
>	测试操作数 1 是否大于操作数 2
>=	测试操作数 1 是否大于或等于操作数 2
<	测试操作数 1 是否小于操作数 2
<=	测试操作数 1 是否小于或等于操作数 2
=	测试操作数 1 是否等于操作数 2
<>	测试操作数 1 是否不等于操作数 2

结构

比较指令要加方括号，跟在指令 LD， AND，和 OR 的后面。当被请求的比较为真时，结果为 1。
比较指令示例。




```
LD    [%MW10 > 100]
ST    %Q0.3

LD    %M0
AND   [%MW20 < %KW35]
ST    %Q0.2

LD    %I0.2
OR    [%MF30>=%MF40]
ST    %Q0.4
```

语法

比较指令语法：

运算符	语法
>, >=, <, <=, =, <>	LD [Op1 运算符 Op2] AND [Op1 运算符 Op2] OR [Op1 运算符 Op2]

操作数：

类型	操作数 1 (Op1)	操作数 2 (Op2)
字	%MWi, %KWi, %INWi, %IW, %IWAi, %QNWi, %QWi, %QWAi, %QNWi, %SWi, %BLK.x	立即值，%MWi, %KWi, %INWi, %IW, %IWAi, %QNWi, %QW, %QWAi, %SWi, %BLK.x, %MWi [%MWi], %KWi [%MWi]
双字	%MDi, %KDi	立即值，%MDi, %KDi, %MDi [%MWi], %KD [%MWi]
浮点字	%MFi, %KFi	立即浮点值，%MFi, %KFi, %MFi [%MWi], %KFi [%MWi]

注意：比较指令可用于圆括号内。

圆括号内比较指令使用示例：

```
LD      %M0
AND(    [%MF20 > 10.0]
OR      %I0.0
)
ST      %Q0.1
```

整数算术指令

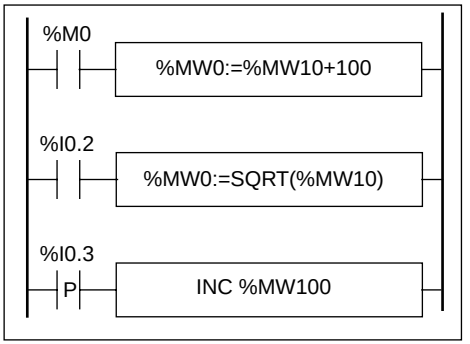
介绍

算术指令用于执行两个整数操作数之间或一个整数操作数上的算术运算。
下表列出了算术指令类型。

指令	功能
+	两个操作数相加
-	两个操作数相减
*	两个操作数相乘
/	两个操作数相除
REM	两个操作数相除的余数
SQRT	一个操作数的平方根
INC	一个操作数递增
DEC	一个操作数递减
ABS	一个操作数的绝对值

结构

算术运算如下所示：




```
LD      %M0
[%MW0:=%MW10 + 100]

LD      %I0.2
[%MW0:=SQRT(%MW10)]

LDR     %I0.3
[INC %MW100]
```

语法

语法取决于使用的运算符，如下表所示。

运算符	语法
+,-,*,/,REM	[Op1:= Op 2 运算符 Op3]
INC, DEC	[运算符 Op1]
SQRT (1)	[Op1: = SQRT(Op2)]
ABS (1)	[Op1: = ABS(Op2)]

操作数：

类型	操作数 1 (Op1)	操作数 2 和 3 (Op2 & 3) (1)
字	%MWi, %QWi, %QWAi, %SWi	立即值, %MWi, %KWi, %INW, %IW, %IWAi, %QNW, %QW, %QWAi, %SWi, %BLK.x
双字	%MDi	立即值, %MDi, %KDi

注意：（1）使用这个运算符时， Op2 不能是立即值。
ABS 函数只能用于对双字 （%MD 和 %KD）和浮点 （%MF 和 %KF）的操作。因此， OP1 和 OP2 必须为双字或浮点。

溢出和错误条件

加法

- 字运算中溢出
  如果结果超出范围，位 %S18 （溢出）被置为 1，且所得结果不正确 （见下页例 1）。用户程序管理位 %S18。

注意：

对双字，其范围是 -2147483648 和 21474836487。

乘法

- 运算时溢出
  如果结果超出字的范围，位 %S18 （溢出）被置为 1，且结果没有意义。

除法 / 取余

- 被 0 除
  如果除数是 0，则不能进行除法运算且系统位 %S18 被置为 1。结果不正确。
- 运算时溢出
  如果商超出字的范围，位 %S18 被置为 1。

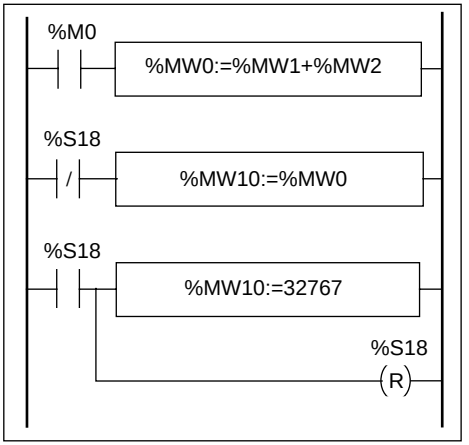
平方根开方

- 运算时溢出
  平方根开方只适用正值。这样，其结果总为正。如果平方根操作数为负，系统位 %S18 被置为 1 且结果不正确。

注意：用户程序用于管理系统位 %S17 和 %S18。它们被控制器置为 1，且必须由程序复位以便再次使用 （见前面示例页）。

示例

示例 1：加法溢出。




```
LD      %M0
[%MW0:=%MW1 + %MW2]

LDN     %S18
[%MW10:=%MW0]

LD      %S18
[%MW10:=32767]
R       %S18
```

如果 %MW1=23241 且 %MW2=21853，实际结果 （45094）不能由一个 16 位字表示，位 %S18 被置为 1 且获得的结果 （-20442）不正确。此例中结果大于 32767 时，其值都固定为 32767。

逻辑指令

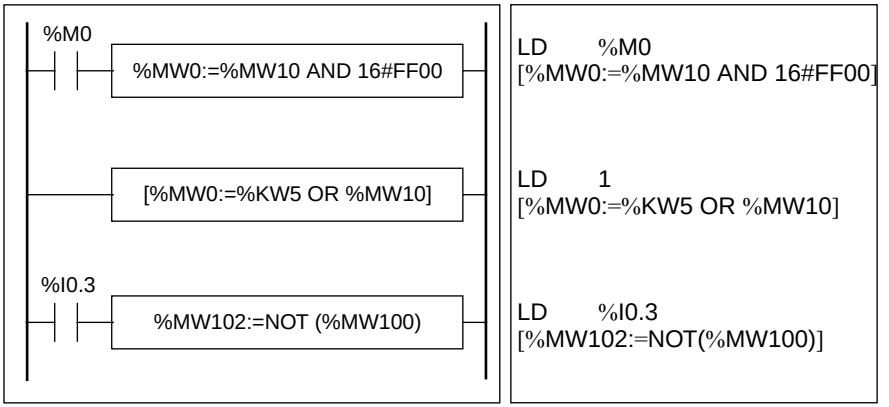
介绍

逻辑指令用于执行两个字操作数之间或一个字操作数的逻辑运算。
下表列出了逻辑指令类型。

指令	功能
AND	与 （位方式），用于两个操作数之间
OR	逻辑或 （位方式），用于两个操作数之间
XOR	异或 （位方式），用于两个操作数之间
NOT	逻辑反 （位方式），用于一个操作数

结构

逻辑运算执行如下：



语法

语法取决于使用的运算符：

运算符	语法	操作数 1 (Op1)	操作数 2 和 3 (Op2 & 3)
AND, OR, XOR	[Op1: = Op2 运算符	%MWi, %QWi, %QWAi, %SWi	立即值 (1), %MWi, %KWi, %IW, %IWAi, %QW, %QWAi, %SWi, %BLK.x
NOT	[Op1: =NOT(Op2)]		

注意：(1) 在 NOT 指令中， Op2 不能是立即值。

示例

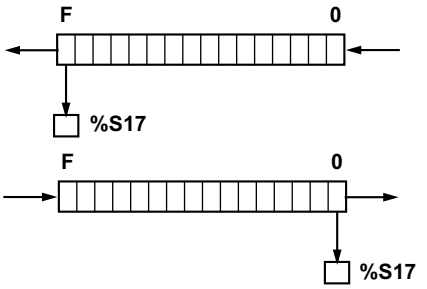
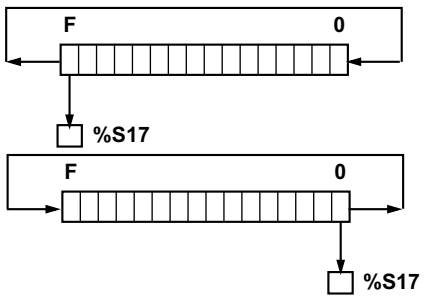
下面是一个逻辑与指令示例：

```
[ %MW15:=%MW32 AND %MW12]
```

移位指令

介绍

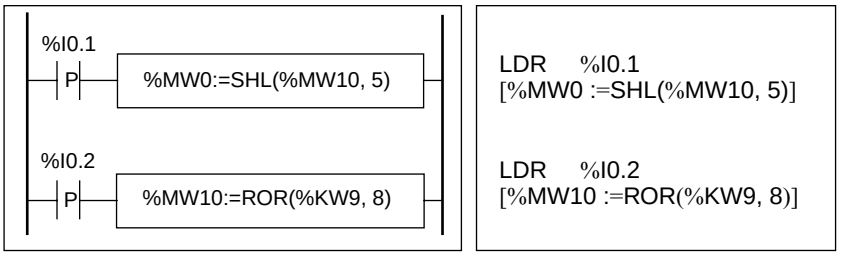
移位指令将操作数向右或向左移动若干位。
下表列出了移位指令类型。

指令	功能	
逻辑移位		
SHL(op2,i)	向左逻辑移动 i 位。	
SHR(op2,i)	向右逻辑移动 i 位。	
循环移位		
ROR(op2,i)	向左循环移动 i 位。	
ROL(op2,i)	向右循环移动 i 位	

注意：系统位 %S17 （见*系统位 （%S）,P.658*）用于容量超出状态。

结构

移位操作执行如下:



语法

语法取决于使用的运算符，如下表所示。

运算符	语法
SHL, SHR	[Op1: = Operator (Op2,i)]
ROL, ROR	

操作数：

类型	操作数 1 (Op1)	操作数 2 (Op2)
字	%MWi, %QWi, %QWAi, %SWi	%MWi, %KWi, %IW, %IWAi, %QW, %QWAi, %SWi, %BLK.x
双字	%MDi	%MDi, %KDi

转换指令

介绍

转换指令执行数字不同类型间的转换。
下表列出了转换指令类型。

指令	功能
BTI	BCD --> 二进制 转换
ITB	二进制 --> BCD 转换

BCD 码介绍

二进制编码的十进制 (BCD) 用四位二进制码表示表示一个十进制数 （0 到 9）。这样一个 16 位字对象可以用四个十进制数字表示 （0000 - 9999），一个 32 位双字对象可以包含 8 个十进制数字。
转换中，如果值不是 BCD 码，则位 %S18 置为 1。该位必须由程序测试和复位到 0。
十进制数的 BCD 码表示：

十进制	0	1	2	3	4	5	6	7	8	9
BCD	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

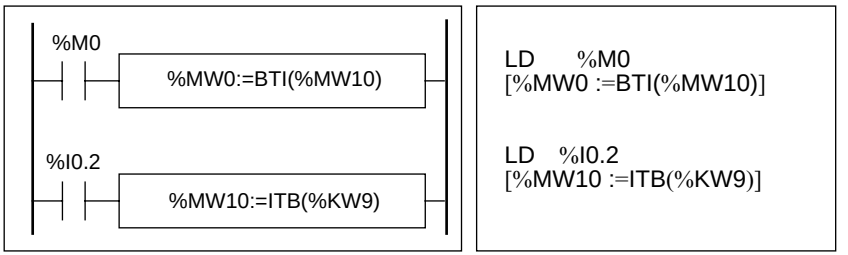
示例：

- 字 %MW5 的 BCD 值 "2450" 对应二进制值：0010 0100 0101 0000
- 字 %MW12 的十进制值 "2450" 对应二进制值：0000 1001 1001 0010

用指令 BTI 将字 %MW5 转换成字 %MW12。
用指令 ITB 将字 %MW12 转换成字 %MW5。

结构

转换操作执行如下:



语法

语法取决于使用的运算符，如下表所示

运算符	语法
BTI, ITB	[Op1: = Operator (Op2)]

操作数：

类型	操作数 1 (Op1)	操作数 2 (Op2)
字	%MWi, %QWi, %QWAi, %SWi	%MWi, %KWi, %IW, %IWAi, %QW, %QWAi, %SWi, %BLK.x
双字	%MDi	%MDi, %KDi

应用示例：

BTI 指令用来处理 BCD 码在控制器输入的设定值。
ITB 指令用来以 BCD 码显示数字值 （如，计算结果，功能模块的当前值）。

单 / 双字转换指令

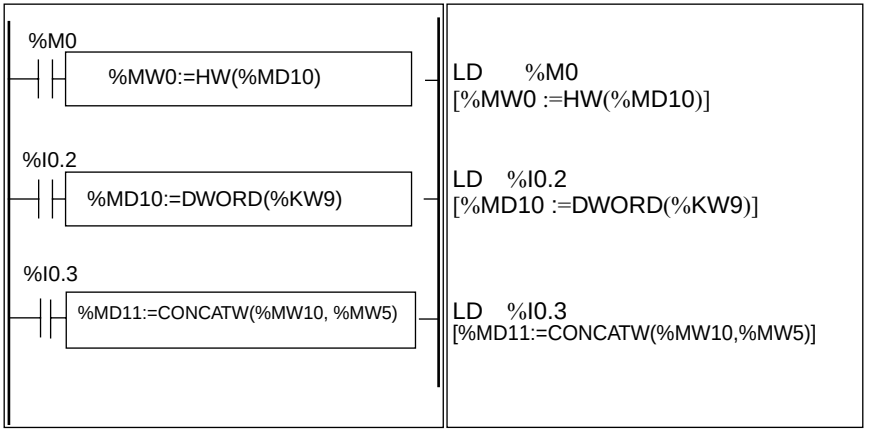
介绍

下表描述了用于执行单字和双字之间转换的指令：

指令	功能
LW	抽取双字的 LSB 到一个字。
HW	抽取双字的 MSB 到一个字。
CONCATW	合并两个字到一个双字。
DWORD	转换一个 16 位字到一个 32 位字。

结构

转换操作执行如下：



语法

语法取决于使用的运算符，如下表所示：I

运算符	语法	操作数 1 (Op1)	操作数 2 (Op2)	操作数 3 (Op3)
LW, HW	Op1 = 运算符 (Op2)	%MWi	%MDi, %KDi	[-]
CONCATW	Op1 = 运算符 (Op2, Op3))	%MDi	%MWi, %KWi, 自然数	%MWi, %KWi, 自然数
DWORD	Op1 = 运算符 (Op2)	%MDi	%MWi, %KWi	[-]

16.4 程序指令

概览

本节的主题

本节提供了程序指令介绍。

本节包含了哪些内容?

本节包含了以下主题：

END 指令

介绍

End 指令定义一个程序扫描执行的结束。

END, ENDC, 和 ENDCN

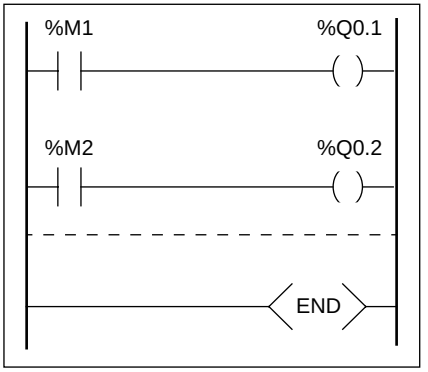
有三个不同的结束指令可用：

- END: 程序无条件结束
- ENDC: 如果前面测试指令布尔运算结果是 1，则程序结束
- ENDCN: 如果前面测试指令布尔运算结果是 0，则程序结束

默认 （正常模式）情况下，当程序结束被激活时，输出被更新且开始下一次扫描。如果是周期扫描，则当周期结束时输出被更新且开始下一次扫描。

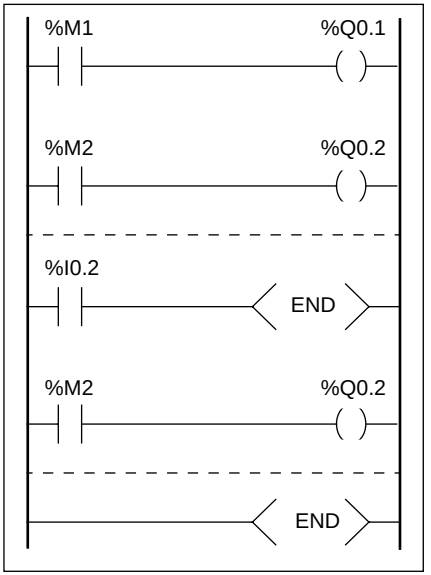
示例

无条件 END 指令示例。




```
LD      %M1
ST      %Q0.1
LD      %M2
ST      %Q0.2

.....

END
```

有条件 END 指令示例。




```
LD      %M1
ST      %Q0.1
LD      %M2
ST      %Q0.2

.....

LD      %I0.2
ENDC                  → 如果 %I0.2 = 1,
                        程序扫描结束 ning
LD      %M2
ST      %Q0.2

                      如果 %I0.2 = 0,
                      继续程序扫描，直到下
.....                 一个 END 指令

END
```

NOP 指令

NOP

NOP 指令不执行任何操作。用它在程序中 “保留” 行，以便您以后插入指令而无需修改行号。

跳转指令

介绍

跳转指令使程序执行立即中断并转入执行标号为 %Li （i = 1 到 16 对于一体型控制器， 1 到 63 对于其它控制器）的程序行。

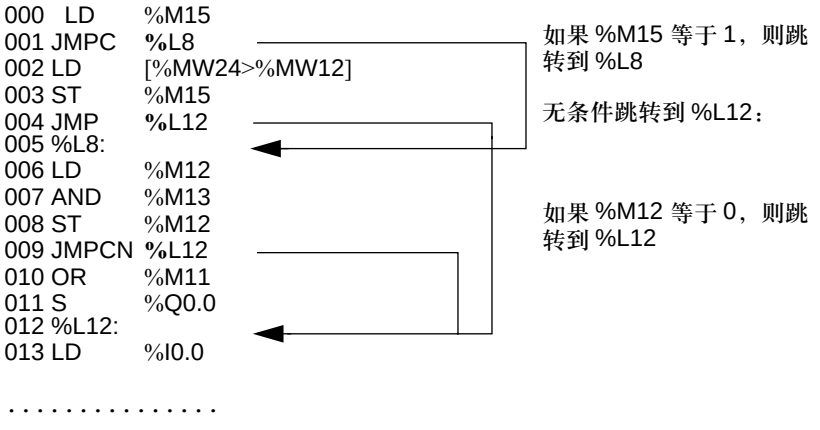
JMP, JMPC 和 JMPCN

有三个不同的跳转指令可用：

- JMP: 程序无条件跳转
- JMPC: 如果前面逻辑布尔运算结果为 1，则程序跳转
- JMPCN: 如果前面逻辑布尔运算结果为 0，则程序跳转

示例

跳转指令示例。



指导

- 跳转指令不允许用于圆括号内，且不能位于指令 AND（，OR（和右括号指令 ") " 之间。
- 标号只能位于指令 LD， LDN， LDR， LDF 或 BLK 之前。
- 标号 %Li 的编号在程序中只能被定义一次。
- 程序可以向下或向上跳转，当向上跳转时，必须注意程序扫描时间。延长扫描时间可能导致看门狗的触发。

子程序指令

介绍

子程序指令使程序执行一个子程序，然后返回到主程序。

SRn, SRn: 和 RET.

子程序由三步组成：

- 如果前面布尔指令运算结果是 1，此 **SRn** 指令调用标号为 **SRn** 的子程序。
- 子程序用标号 **SRn**：表示， n = 0 到 15 对于 TWDLCAA10DRF，TWDLCAA16DRF， 0 到 63 对于其它控制器。
- 位于子程序末端的 **RET** 指令将程序流返回到主程序。

示例

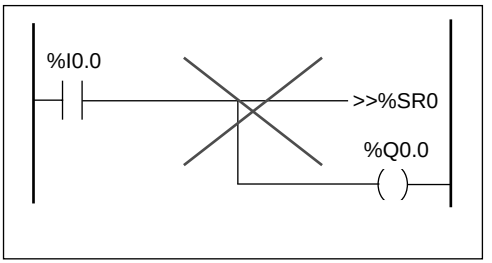
子程序指令示例。

```
000  LD     %M15
001  AND    %M5
002  ST     %Q0.0
003  LD     [%MW24>%MW12]
004  SR8                          跳转到子程序 SR8
005  LD     %I0.4
006  AND    M13
007  _
008  _
009
010  END

011  SR8:
012  LD     1
013  IN     %TM0
014  LD     %TM0.Q
015  ST     %M15
010  RET                          返回到主子程序
. . . . . . . . . . . . . . . . . . . .
```

指导

- 一个子程序不能调用另一个子程序。
- 子程序指令不允许用于圆括号内，且不能位于指令 AND （,OR （和右括号指令 "） " 之间。
- 标号只能位于指令 LD 或 BLK 之前，用于标识一个布尔等式 （或梯级）的开始。
- 赋值指令不能跟随在子程序调用之后。这是因为子程序可能改变布尔运算累加器的内容。这样返回时，它的值可能与调用前不同。见下面示例。

子程序编程示例。




```
LD      %I0.0
SR0
ST      %Q0.0

LD      %I0.0
ST      %Q0.0
SR0
```

高级指令

17

概览

本章的主题

本章提供了有关用于创建 Twido 可编程控制器高级控制程序的指令和功能模块的详细资料。

本章包含了哪些内容?

本章包含了以下几节：

17.1 高级功能模块

概览

本节的主题

本节提供了高级功能模块介绍，包括编程示例。

本节包含了哪些内容?

本章包含了以下主题：

与高级功能模块有关的位和字对象

介绍

高级功能模块使用与标准功能模块类似的专用字和位类型。高级功能模块包括：

- LIFO/FIFO 寄存器 (%R)
- 鼓控制器 (%DR)
- 高速计数器 (%FC)
- 超高速计数器 (%VFC)
- 脉宽调制输出 (%PWM)
- 脉冲发生器输出 (%PLS)
- 移位寄存器 (%SBR)
- 步进计数器 (%SC)
- 消息控制模块 (%MSG)

程序可访问的对象

下表包含了与各种高级功能模块相关的程序可访问字和位的概览。请注意下表中的写访问决定于配置中选择的 “可调节”设置。通过 TwidoSoft 或操作接口设置允许或拒绝对字或位的访问。

高级功能模块	相关字和位		地址	写访问
%R	字	寄存器输入	%Ri.I	可以
	字	寄存器输出	%Ri.O	可以
	位	寄存器输出满	%Ri.F	不可以
	位	寄存器输出空	%Ri.E	不可以
%DR	字	当前步号	%DRi.S	可以
	位	上一步等于当前步	%DRi.F	不可以
%FC	字	当前值	%FCi.V	可以
	字	预置值	%FCi.P	可以
	位	完成	%FCi.D	不可以
%VFC	字	当前值	%VFCi.V	不可以
	字	预置值	%VFCi.P	可以
	位	计数方向	%VFCi.U	不可以
	字	捕获值	%VFCi.C	不可以
	字	阈值 0	%VFCi.S0	可以
	字	阈值 1	%VFCi.S1	可以
	位	溢出	%VFCi.F	不可以
	位	映像输出 0 使能	%VFCi.R	可以
	位	映像输出 1 使能	%VFCi.S	可以
	位	阈值输出 0	%VFCi.TH0	不可以
	位	阈值输出 1	%VFCi.TH1	不可以
	位	频率测量时基	%VFCi.T	可以
%PWM	字	在 1 处与总周期相关的脉冲百分比	%PWMi.R	可以
	字	预置周期	%PWMi.P	可以
%PLS	字	脉冲数目	%PLSi.N	可以
	字	预置值	%PLSi.P	可以
	位	当前输出使能	%PLSi.Q	不可以
	位	发生完成	%PLSi.D	不可以
%SBR	位	寄存器位	%SBRi.J	不可以

高级功能模块	相关字和位		地址	写访问
%SC	位	步进计数器位	%SCi.j	可以
%MSG	位	完成	%MSGi.D	不可以
	位	错误	%MSGi.E	不可以

高级功能模块编程原则

概览

所有的 Twido 应用程序以列表程序的格式存储，即使在梯形图编辑器中写成，因此， Twido 控制器又被称为列表 “机器”。术语 “可逆性”指 TwidoSoft 能将列表程序用梯形图表示，反之亦然。默认情况下，所有梯形图程序都是可逆的。
和基本功能模块一样，高级功能模块也必须考虑可逆性规则。列表语言中可逆功能模块的结构需要用到下列指令：

- **BLK:** 标志模块的开始和功能模块的输入部分
- **OUT_BLK:** 标志功能模块输出部分的开始
- **END_BLK:** 标志功能模块的结束

注意：这些可逆功能模块指令的使用对一个正确功能的列表程序来说并不是强制性的。一些指令可用于列表语言编程但并不可逆。

专用输入和输出

高速计数器，超高速计数器，PLS，和 PWM 高级功能使用专用输入和输出，但这些位不被任何一个模块专用所保留。而且，您必须管理这些专用资源的使用。
当使用这些高级功能时，必须管理这些专用资源的使用，您必须管理怎样分配专用输入和输出。TwidoSoft 帮助配置这些资源，显示输入 / 输出配置详细信息，并在一个专用输入或输出已被配置的功能模块使用时提出警告。
下表是专用输入和输出及其特殊功能概括。
当用于计数功能时：

输入	用法
%I0.0.0	%VFC0: 加 / 减管理或相位 B
%I0.0.1	%VFC0: 脉冲输入或相位 A
%I0.0.2	%FC0: 脉冲输入或 %VFC0 预置输入
%I0.0.3	%FC1: 脉冲输入或 %VFC0 捕获输入
%I0.0.4	%FC2: 脉冲输入或 %VFC1 捕获输入
%I0.0.5	%VFC1 预置输入
%I0.0.6	%VFC1: 加 / 减管理或相位 B
%I0.0.7	%VFC1: 脉冲输入或相位 A

当用于计数或特殊功能时：

输出	用法
%Q0.0.0	%PLS0 或 PWM0 输出
%Q0.0.1	%PLS1 或 PWM1 输出
%Q0.0.2	%VFC0 映像输出
%Q0.0.3	
%Q0.0.4	%VFC1 映像输出
%Q0.0.5	

专用输入和输出使用

TwidoSoft 对专用输入和输出的使用规定了下面规则。

- 每个使用专用 I/O 的功能模块必须被配置，然后被应用程序引用。专用 I/O 只能在配置功能模块时被分配，程序引用时不能分配。
- 功能模块被配置后，它的专用输入和输出不能被应用程序或另一功能模块使用。例如，如果您配置 %PLS0，您不能在 %DR0 （鼓控制器）或应用程序逻辑 （如 ST %Q0.0.0）中使用 %Q0.0.0。
- 如果一个功能模块所需的专用输入或输出已被应用程序或另一功能模块使用，则这个功能模块不能被配置。

  例如，如果您配置 %FC0 作为一个加计数器，则您不能配置 %VFC0 使用 %I0.0.2 作为捕获输入。

注意：为了改变专用 I/O 的使用，可以通过设置对象的类型为 “未使用”来取消功能模块配置，然后删除应用程序中被调用的功能模块。

LIFO/FIFO 寄存器功能模块 （%Ri）

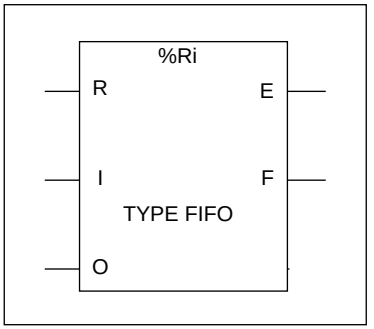
介绍

一个寄存器是一个内存块，可以存储 16 个 16 位的字，可用两种方式的其中一种：

- 队列方式 （先入先出）即 FIFO。
- 堆栈方式 （后进先出）即 LIFO。

图例

下面是一个寄存器功能模块图例。



寄存器功能模块

参数

寄存器功能模块具有如下参数:

参数	标识	值
寄存器编号	%Ri	0 到 3。
类型	FIFO 或 LIFO	队列方式或堆栈方式。
输入字	%Ri.I	寄存器输入字。可读取，测试，和写入。
输出字	%Ri.O	寄存器输出字。可读取，测试，和写入。
存储输入 （或指令）	I (In)	上升沿处将字 %Ri.I 的内容存入寄存器。
取回输入 （或指令）	O (Out)	上升沿处将一个数据字装入字 %Ri.O 内。
输入 （或指令）复位	R (Reset)	状态为 1 时，初始化寄存器。
空输出	E (Empty)	对应位 %Ri.E 表示寄存器空。 可调试。
满输出	F (Full)	对应位表 %Ri.F 示寄存器满。 可调试。

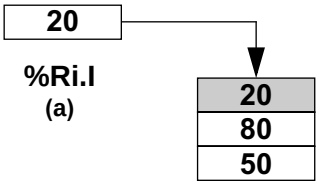
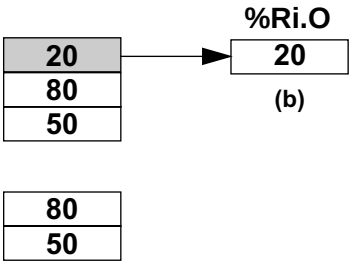
LIFO 操作

介绍

LIFO 操作 （后进先出）中，最后进入的数据项最先被取出。

操作

下表是 LIFO 操作描述。

步骤	描述	示例
1	当接收到一个存储请求 （输入 I 的上升沿或指令 I 被激活），输入字 %Ri.I 的内容 （已经被装入）被存放到堆栈的顶部 （图 a）。当堆栈已满 （输出 F=1）时，不能再存入数据。	存储%Ri.I的内容到堆栈的顶部。 
2	当接收到一个取回请求 （上升沿输入 O 或指令 O 被激活），堆栈顶部的数据字 （最后进入的字）被装入字 %Ri.O （图 b）。当寄存器为空 （输出 E=1）时，不能再取出数据。输出字 %Ri.O 不变且保持原值。	取出堆栈中最上面数据字。 
3	堆栈可在任何时候被复位 （输入 R 状态为 1 或指令 R 被激活）。指针指向堆栈的顶端元素。	

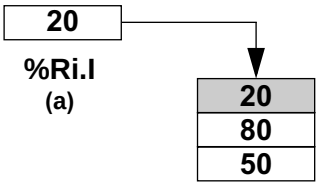
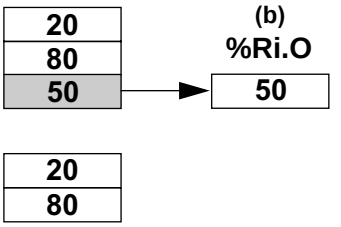
FIFO，操作

介绍

FIFO 操作 （先进先出）中，最先进入的数据项最先被取出。

操作

下表是 FIFO 操作描述。

步骤	描述	示例
1	当接收到一个存储请求 （输入 I 的上升沿或指令 I 被激活），输入字 %Ri.I 的内容 （已经被装入）被存放到队列的顶部（图 a）。当队列已满 （输出 F=1）时，不能再存入数据。	存储%Ri.I的内容到队列的顶部。 
2	当接收到一个取回请求 （输入 O 的上升沿或指令 O 被激活），队列最底部的数据字被装入字 %Ri.O，且寄存器队列中的数据字都往底部移动一格（图 b）。 当寄存器为空 （输出 E=1）时，不能再取出数据。输出字 %Ri.O 不变且保持原值。	取出第一个数据项装入 %Ri.O 中。 
3	队列可在任何时候被复位（输入 R 状态为 1 或指令 R 被激活）。	

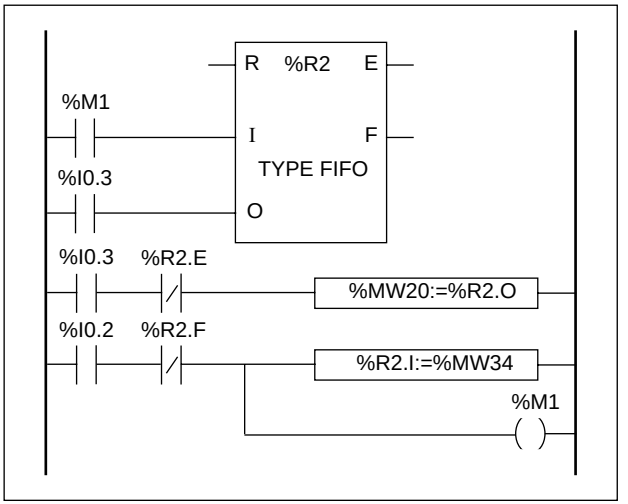
寄存器编程和配置

介绍

下面编程示例显示了寄存器 %R2 未满 （%R2.F = 0）时，使用存储请求 （%I0.2）将一个存储字 （%MW34）的内容装入寄存器 （%R2.I）。寄存器的存储请求由 %M1 实现。取出请求由输入 %I0.3 来实现，且如果寄存器不为空 （%R2.E = 0），则 %R2.O 的内容被装入 %MW20。

编程示例

下图是一个寄存器功能模块可逆和不可逆编程示例。



梯形图

```
BLK          %R2
LD           %M1
I
LD           %I0.3
O
END_BLK
LD           %I0.3
ANDN         %R2.E
[%MW20:=%R2.O]
LD           %I0.2
ANDN         %R2.F
[%R2.I:=%MW34]
ST           %M1
```

可逆编程

```
LD           %M1
I            %R2
LD           %I0.3
O            %R2
ANDN         %R2.E
[%MW20:=%R2.O]
LD           %I0.2
ANDN         %R2.F
[%R2.I:=%MW34]
ST           %M1
```

不可逆编程

配置

配置时必须输入寄存器类型这个参数：
- FIFO (默认), 或
- LIFO

特殊情况

下表列出了寄存器功能模块编程的特殊情况：

特殊情况	描述
冷启动 （%S0=1）的影响	初始化寄存器的内容。与输出 E 相关的输出位 %Ri.E 被置为 1。
控制器停止后热启动 （%S1=1）的影响	对寄存器的当前值和输出位的状态没有影响。

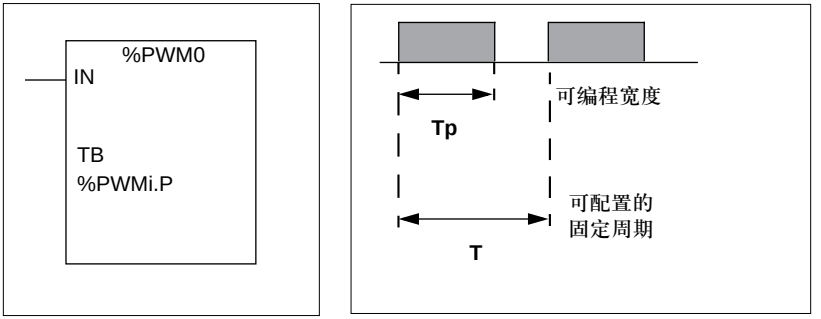
脉宽调制功能模块 （%PWM）

介绍

脉宽调制 （%PWM）功能模块在专用输出通道 %Q0.0.0 或 %Q0.0.1 生成一个方波信号，宽度可变，因此占空比也可变。由于频率限制，带有继电器输出的控制器的这两个通道不支持这个功能。
有两个 %PWM 模块可用。%PWM0 使用专用输出 %Q0.0.0，%PMW1 使用专用输出 %Q0.0.1。%PLS 功能模块也使用这两个输出，因此您必须在这两个功能之间做出选择。

图例

PWM 模块和时序图：



参数

下表列出了 PWM 功能模块参数。

参数	标识	描述
时基	TB	0.142 ms, 0.57 ms, 10 ms, 1 s （默认值）
预置周期	%PWMi.P	0 < %PWMi.P <= 32767 如果时基为 10 ms 或 1 s 0 < %PWMi.P <= 255 如果时基为 0.57 ms 或 0.142 ms 0 = 不使用
占空比	%PWMi.R	T 表示状态为 1 的信号在周期中的百分比。宽度 Tp 等于： Tp= T* （%PWMi.R/100）用户应用程序写 %PWMi.R 值。这个字控制周期的占空比。对于 T 定义，见下面 “周期范围”。默认值为 0，当值大于 100 时视为等于 100。
脉冲发生器输入	IN	状态为 1 时，脉宽调制信号在输出通道发生。状态为 0 时，输出通道被置为 0。

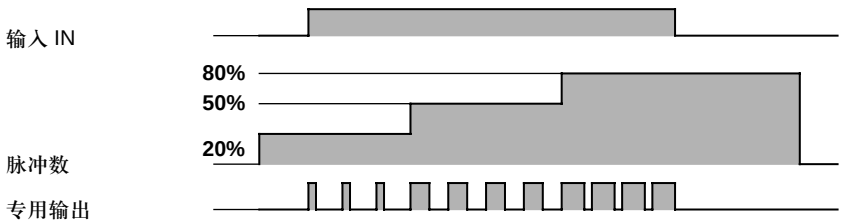
周期范围

预置值和时基可在配置中修改。它们用于确定信号周期 T=%PWMi.P* TB。获得的比率越小，则选择的 %PWMi.P 越大。可用的周期范围：

- 时基 0.142 ms 时， 0.142 ms 到 36.5 ms （27.4Hz 到 7kHz）
- 时基 0.57 ms 时， 0.57 ms 到 146 ms （6.84Hz 到 1.75kHz）
- 时基 10 ms 时， 10 ms 到 5.45 mins
- 时基 1 sec 时， 1 sec 到 9.1 hours

操作

输出信号的频率在配置时通过选择时基 TB 和预置 %PWMi.P 来设置。在程序中修改占空比 %PWMi.R 来调制信号的宽度。下面是 PWM 功能模块占空比改变时的脉冲图示。



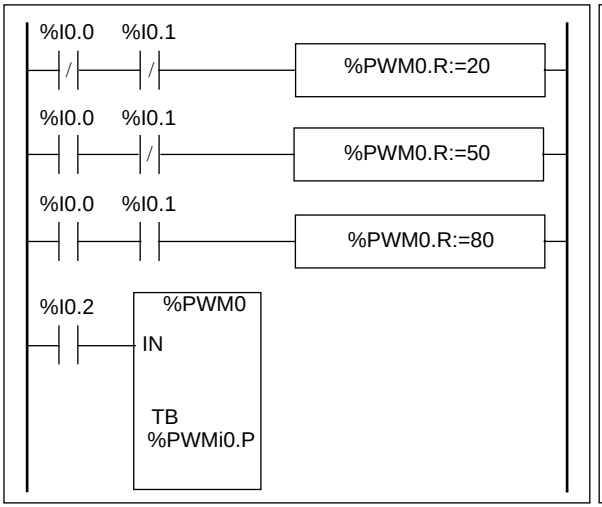
编程和配置

在这个示例中，信号宽度由程序根据控制器输入 %I0.0.0 和 %I0.0.1 的状态来修改。如果 %I0.0.1 和 %I0.0.2 置为 0， %PWM0.R 比率置为 20%，则信号在状态 1 的持续时间为：20 % × 500 ms = 100 ms。

如果 %I0.0.0 置为 0， %I0.0.1 置为 1， %PWM0.R 比率置为 50%（持续时间为 250 ms）。

如果 %I0.0.0 和 %I0.0.1 置为 1， %PWM0.R 比率置为 80%（持续时间为 400 ms）。

编程示例：




```
LDN       %I0.0
ANDN      %I0.1
[%PWM0.R:=20]
LD        %I0.0
ANDN      %I0.1
[%PWM0.R:=50]
LD        %I0.0
AND       %I0.1
[%PWM0.R:=80]
BLK       %PWM0
LD        %I0.2
IN
END_BLK
```

特殊情况

下表显示了 PWM 功能模块特殊操作列表。

特殊情况	描述
冷启动 （%S0=1）的影响	设置 %PWMi.R 比率为 0。另外， %PWMi.P 复位到配置预置值，且这将取代活动表编辑器或可选操作显示中完成的任何变化。
热启动 （%S1=1）的影响	没有影响。
输出专用给 %PWM 模块的影响	使用一个编程设备强制输出 %Q0.0.0 或 %Q0.0.1 不会停止信号的生成。

脉冲发生器输出功能模块 （%PLS）

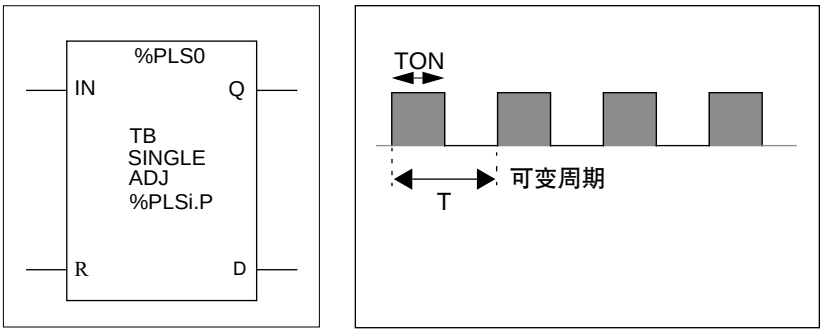
介绍

%PLS 功能模块用于生成方波信号。有两个 %PLS 功能模块可用专用输出通道 %Q0.0.0 或 %Q0.0.1。 %PLS 功能模块只允许一个信号宽度，或 50% 的占空比。当脉冲序列执行时，您可以选择限制脉冲的数目或周期。这可由配置的时间决定，和 / 或被用户应用程序更新。

注意：若控制器的这两个通道为继电器输出，则不支持 %PLS 功能。

示例

脉冲发生器功能块例子，以单字模式：



- TON=T/2 对于 0.142ms 和 0.57ms 时基
  = （%PLSi.P*TB） /2
- TON=[整数部分 （%PLSi.P） /2]*TB 对于 10ms 到 1s 时基。

说明

下表为 PLS 功能模块特性：

功能	对象	描述
时基	TB	0.142 ms, 0.57 ms, 10 ms, 1 sec
预置周期	%PLSi.P	当以 0.142 ms， 0.57 ms 为时基时，即使 %PLS1.N 或 %PLS1.ND* 到达，脉冲输出 %PLS1 也不会停止。 ● 1 < %PLSi.P <= 32767 对于时基 10 ms 或 1 s ● 0 < %PLSi.P <= 255 对于时基 0.57 ms 或 0.142 ms ● 0 = 功能未被使用。 要从时基为 10ms 或 1s 的占空比获得好的精确级别，则推荐使 %PLSi >= 100，如果 P 是奇数。
脉冲数目	%PLSi.N %PLSi.ND *	以周期 T 产生的脉冲数被限制为 0 <= %PLSi.N <= 32767 标准模式或 0 <= %PLSi.ND <= 4294967295 双字模式。默认值为 0。 要产生无限的脉冲数，可以设置 %PLSi.N 或 %PLSi.ND 为 0。不管可调节的设置，脉冲数目总可以被改变。
可调节	Y/N	如果置为 Y，则可以通过 HMI 或活动表编辑器修改预置值 %PLSi.P。如果置为 N 则不能访问这个预置值。
脉冲发生器输入	IN	状态为 1 时，脉冲生成发生在专用输出通道。状态为 0 时，输出通道被置为 0。
输入复位	R	状态为 1 时，输出 %PLSi.Q 和 %PLSi.D 被置为 0。周期 T 内生成的脉冲数目被置为 0。
当前脉冲生成输出	%PLSi.Q	状态为 1 时，表示脉冲信号在配置的专用输出通道生成。
脉冲生成完成输出	%PLSi.D	状态为 1 时，信号生成完成。达到期望的脉冲数目。

注意：（*）指双字变量。

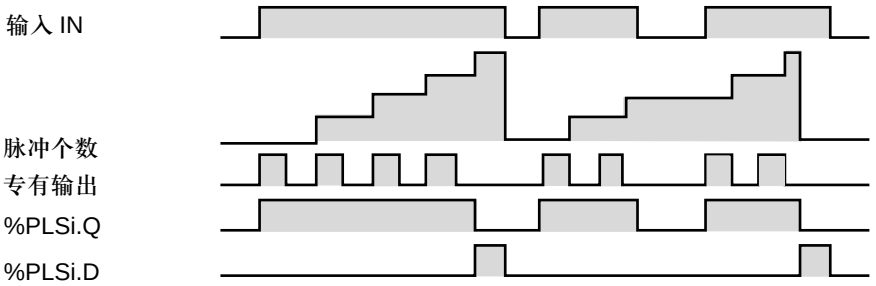
周期范围

预置值和时基可在配置中修改。它们用于确定信号周期 T=%PLSi.P * TB。
可用的周期范围：

- 时基 0.142 ms 时， 0.142 ms 到 36.5 ms （27.4Hz 到 7kHz）
- 时基 0.57 ms 时， 0.57 ms 到 146 ms （6.84 Hz 到 1.75 kHz）
- 时基 10 ms 时， 20 ms 到 5.45 mins
- 时基 1 sec 时， 2sec 到 9.1 hours

操作

下面是 %PLS 功能模块图例。



特殊情况

特殊情况	描述
冷启动 （%S0=1）的影响	设置 %PLSi.P 到配置定义值
热启动 （%S1=1）的影响	没有影响
修改预置值 （%PLSi.P）的影响	立即生效
输出专用给 %PLS 模块的影响	使用一个编程设备强制输出 %Q0.0.0 或 %Q0.0.1 不会停止信号的生成。

注意：%PLSx.D 当达到期望的脉冲数目时被置位。可以设置输入 IN 或 R 到 1 将其复位。

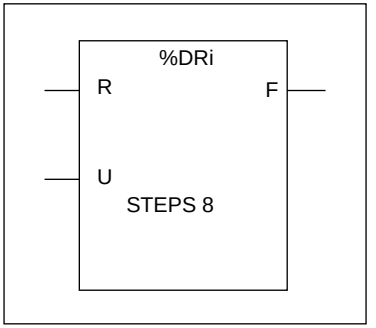
鼓控制器功能模块 （%DR）

介绍

鼓控制器的工作原理与机电类鼓控制器类似，即根据外部事件改变步序。在每一步，凸轮的高点给出一个命令让控制器执行。鼓控制器中，每一步的高点用状态 1 来表示，且作为控制位被赋给输出位 %Qi.j 或内部位 %Mi。

图例

下面是鼓控制器功能模块图例。



鼓控制器功能模块

参数

鼓控制器功能模块具有如下参数：

参数	标识	值
编号	%DRi	0 到 3 对一体型控制器，0 到 7 对模块型控制器
当前步号	%DRi.S	0<%DRi.S<7。该字可被读和写。被写值必须是十进制立即值。被写后，在功能模块下次执行时生效。
步数		1 到 8 （默认值）
回到 0 步输入（或指令）	R (Reset)	状态为 1 时，将磁鼓控制器置为步 0。
前进输入 （或指令）	U (Upper)	上升沿使鼓控制器前进一步并更新控制位。
输出	F (Full)	表示当前步等于定义的最后一步。相关位 %DRi.F 可被测试 （例如，如果 %DRi.S= 配置的步数 -1，则 %DRi.F=1）。
控制位		与步 （16 位控制位）相关的输出或内部位，在配置编辑器中被定义。

鼓控制器功能模块 %DRi 操作

介绍

鼓控制器包括：

- 由 8 步和 16 个数据位 （步的状态）组成的常数 （凸轮）矩阵，数据列编号为 0 到 F。
- 一个控制位列表，它与配置输出 （%Qi.j.k）或存储字 （%Mi）相对应。在当前步，控制位为该步定义的二进制状态。

下表中的示例概括了鼓控制器的主要特性。

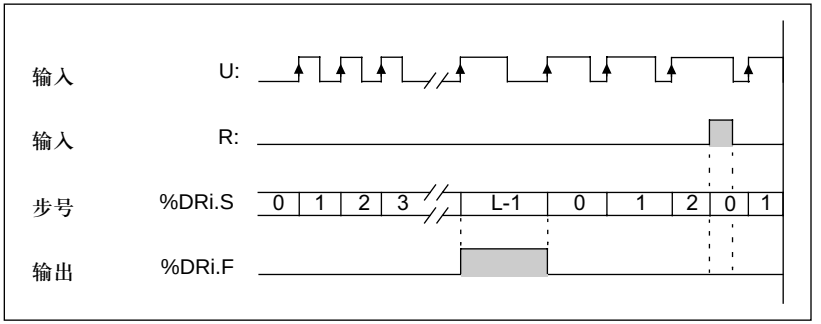
列	0	1	2		D	O	F
控制位	%Q0.1	%Q0.3	%Q1.5		%Q0.6	%Q0.5	%Q1.0
0 步	0	0	1		1	1	0
1 步	1	0	1		1	0	0
5 步	1	1	1		0	0	0
6 步	0	1	1		0	1	0
7 步	1	1	1		1	0	0

操作

上面示例中，步 5 是当前步，控制位 %Q0.1， %Q0.3，和 %Q1.5 被置为状态 1；控制位 %Q0.6， %Q0.5，和 %Q1.0 被置为状态 0。当前步号在输入 U 的每个上升沿处 （或指令 U 被激活时）增加。当前步可通过程序修改。

时序图

下图是鼓控制器操作说明。



特殊情况

下表包含了鼓控制器操作的特殊情况列表。

特殊情况	描述
冷启动 （%S0=1）的影响	复位鼓控制器到步 0 （控制位更新）。
冷启动 （%S0=1）的影响	当前步后更新控制位。
程序跳转的影响	鼓控制器不再被扫描，意味着控制位不被复位。
控制位更新	只在步发生变化或冷热启动时发生。

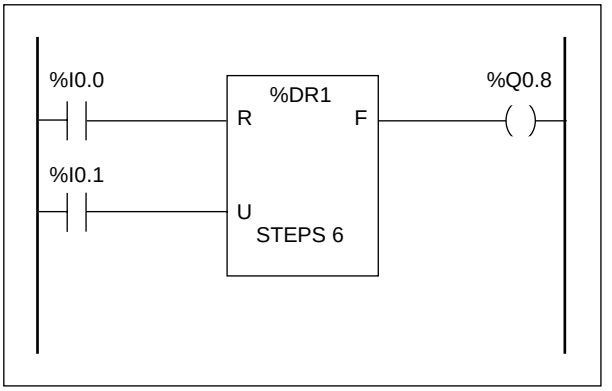
鼓控制器编程和配置

介绍

下面是一个鼓控制器编程和配置示例。每次输入 %I0.1 被置为 1，则前面六个输出 %Q0.0 到 %Q0.5 被激活。输入 I0.0 将输出复位到 0。

编程示例

下面是一个鼓控制器可逆和不可逆编程示例。



梯形图

```
BLK     %DR1
LD      %I0.0
R
LD      %I0.1
U
OUT_BLK
LD      F
ST      %Q0.8
END_BLK
```

配置

下面信息在配置中定义：

- 步数：6
- 鼓控制器每步的输出状态 （控制位）。

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
步骤 1:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
步骤 2:	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
步骤 3:	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
步骤 4:	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
步骤 5:	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
步骤 6:	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

- 控制位分配。

1:	%Q0.0	4:	%Q0.1
2:	%Q0.2	5:	%Q0.3
3:	%Q0.4	6:	%Q0.5

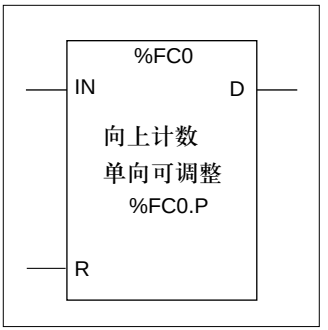
高速计数器功能模块 （%FC）

介绍

高速计数器功能模块 （%FC）可用作加计数器或减计数器。它对数字量输入的上升沿进行计数，频率可到 5kHz ，可用单字或双字模式。因为高速计数器由特殊硬件中断管理，所以保持的最高频率采样率可根据您的特殊应用和硬件配置来改变。TWDLCA • 40DRF 一体型控制器可以配置 4 路高速计数器，然而其他所有的一体型控制器最多可以配置 3 路高速计数器，模块型控制器最多可以使用 2 路。高速计数器功能块 %FC0， %FC1， %FC2，和 %FC3 分别使用特定的输入 %I0.0.2，%I0.0.3， %I0.0.4 和 %I0.0.5。这些位不保留为专用。其它功能模块使用这些专用资源时必须考虑它们的分配。

图例

以下为以单字模式的高速计数器功能块例子。



参数

下表列出了高速计数器功能模块的参数。

参数	标识	描述
功能	类型	配置中设置，可设置为加计数或减计数。
预置值	%FCi.P %FCi.PD	初始值可设为： -> 在 1 和 65635 之间标准模式， -> 在 1 和 4294967295 双字模式，
可调节	Y/N	如果设为 Y，可通过操作器显示模块和动态表编辑器修改预设值 %FCi.P 或 %FCi.PD 和当前值 %FCi.V 或 %FCi.VD。如果设为 N，则不能访问预置值。
当前值	%FCi.V %FCi.VD	当前值根据选择的加或减计数功能增加或减少。对加计数，当前计数值可以被更新而且在标准模式达到 65535 （%FCi.V）和双字模式 4294967295 （%FCi.VD）。对减计数，当前值可以为预设值 %FCi.P 或 %FCi.PD 而且可以计数减到 0。
输入使能	IN	状态为 1 时，当前值根据应用到物理输入的脉冲更新。状态为 0 时，当前值保持上次值不变。
复位	%FCi.R	用于初始化模块。状态为 1 时，如果配置为加计数，当前值被复位为 0，如果配置为减计数可置为 %FCi.P 或 %FCi.PD。完成位 %FCi.D 被置回到它的默认值。
完成	%FCi.D	当配置为加计数且 %FCi.V 或 %FCi.VD 达到 %FCi.P 或 %FCi.PD 时，或者配置为减计数且 %FCi.V 或 %FCi.VD 达到 0 时，该位置 1。该位只读，只能通过置 %FCi.R 为 1 将其复位。

特别注意

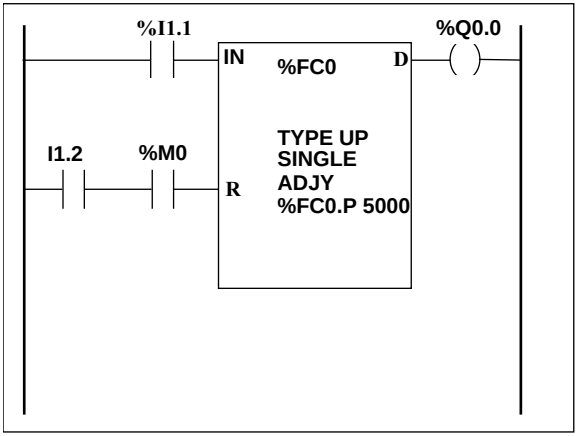
如果配置为可调节，那么在任何时候应用程序可以改预设值 %FCi.P 或 %FCi.PD 和当前值 %FCi.V 或 %FCi.VD。但是，新值仅在输入复位激活时或在输出 %FCi.D 的上升沿生效。这使得不同的计数得到延续而不丢失脉冲。

参数

如果配置为加计数，当专用输入出现一个上升沿时，当前值增加 1。当达到预设值 %FCi.P 或 %FCi.PD 时，完成输出位 %FCi.D 置 1。
如果配置为减计数，当专用输入出现一个上升沿时，当前值减 1。当前值为 0 时，完成输出位 %FCi.D 置 1 且预设值被装入当前值 %FCi.V 或 %FCi.VD。

配置和编程

此例中，当 %I1.1 被置为 1 时，应用程序对高达 5000 条的项目进行计数。 %FC0 的输入是专用输入 %I0.0.2。当到达预置值时， %FC0.D 被置为 1 且保持相同值直至 %FC0.R 得到由 %I1.2 和 %M0 逻辑 “与”的结果给出的命令。




```
BLK       %FC0
LD        %I1.1
IN
LD        %I1.2
AND       %M0
R
OUT_BLK
LD D
ST %Q0.0
END_BLK
```

特殊情况

下表包含了 %FC 功能模块的特殊操作情况列表：

特殊情况	描述
冷启动 （%S0=1）的影响	将所有被用户或用户应用程序配置的 %FC 属性值复位。
热启动 （%S1=1）的影响	没有影响。
控制器停止的影响	控制器被停止时， %FC 继续计数且可以进行参数设置。

超高速计数器功能模块 （%VFC）

介绍

超高速计数器功能模块 （%VFC）可由 TwidoSoft 配置成执行下面功能之一：

- 加 / 减计数器
- 加 / 减 2 相计数器
- 单加计数器
- 单减计数器
- 频率计

%VFC 支持数字量输入在单字或双字模式下频率高达 20kHz。 TWDLCA?40DRF 一体型控制器能配置两路超高速计数器，然而其他所有一体型控制器只能配置一路超高速计数器 （%VFC）。模块型控制器能配置高达两路超高速计数器 （%VFC）。

专用 I/O 分配

超高速计数器功能模块 （%VFC）使用专用输入及辅助输入和输出。这些输入和输出不保留为专用。其它功能模块使用这些专用资源时必须考虑它们的分配。下面列表概括了这些分配：

		主要输入		辅助输入		映像输出	
%VFC0	选择使用	输入 IA	输入 IB	预设	捕捉	输出 0	输出 1
	加 / 减计数器	%I0.0.1	%I0.0.0 (UP=0/DO=1)	%I0.0.2 (1)	%I0.0.3 (1)	%Q0.0.2 (1)	%Q0.0.3 (1)
	加 / 减 2 相计数器	%I0.0.1	%I0.0.0 （脉冲 ）	%I0.0.2 (1)	%I0.0.3 (1)	%Q0.0.2 (1)	%Q0.0.3 (1)
	单加计数器	%I0.0.1	(2)	%I0.0.2 (1)	%I0.0.3 (1)	%Q0.0.2 (1)	%Q0.0.3 (1)
	单减计数器	%I0.0.1	(2)	%I0.0.2 (1)	%I0.0.3 (1)	%Q0.0.2 (1)	%Q0.0.3 (1)
	频率计	%I0.0.1	(2)	(2)	(2)	(2)	(2)
%VFC1	选择使用	输入 IA	输入 IB	预设	捕捉	输出 0	输出 1
	加 / 减计数器	%I0.0.7	%I0.0.6 (UP = 0/DO = 1)	%I0.0.5 (1)	%I0.0.4 (1)	%Q0.0.4 (1)	%Q0.0.5 (1)
	加 / 减 2 相计数器	%I0.0.7	%I0.0.6 (脉冲)	%I0.0.5 (1)	%I0.0.4 (1)	%Q0.0.4 (1)	%Q0.0.5 (1)
	单加计数器	%I0.0.7	(2)	%I0.0.5 (1)	%I0.0.4 (1)	%Q0.0.4 (1)	%Q0.0.5 (1)
	单减计数器	%I0.0.7	(2)	%I0.0.5 (1)	%I0.0.4 (1)	%Q0.0.4 (1)	%Q0.0.5 (1)
	频率计	%I0.0.7	(2)	(2)	(2)	(2)	(2)

注释：

(1) = 可选择
(2) = 不使用
Ipres = 预制输入
Ica= 扑捉输入
输入 **IA** = 脉冲输入
输入 **IB** = 脉冲或加 / 减
UP/DO = 加 / 减计数

在不使用时，输入或输出保持为正常的数字 I/O，并可按主循环中的应用管理。

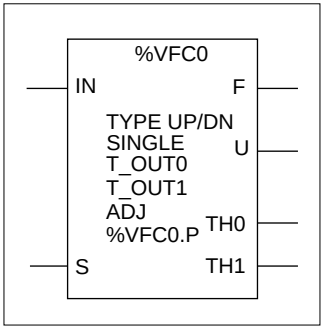
如果 %I0.0.2 用于 %FC0，则不可用。

如果 %I0.0.3 用于 %FC2，则不可用。

如果 %I0.0.4 用于 %FC3，则不可用。

图例

这是一个单字模式的超高速计数块 （%VFC）：



特性　　下面表格列出了超高速计数器功能模块的特性。

功能	描述	值	%VFC 使用	实时访问
当前值 (%VFCi.V) (%VFCi.VD*)	当前值根据物理输入和功能选择增加或减少。该值可使用预置输入 （%VFCi.S）预置或复位。	%VFCi.V: 0 -> 65535 %VFCi.VD: 0 -> 4294967295	CM	读
预设值 (%VFCi.P) (%VFCi.PD*)	仅为加 / 减计数功能和单加或单减计数使用。	%VFCi.P: 0 -> 65535 %VFCi.PD: 0 -> 4294967295	CM	读与写 (1)
捕捉值 (%VFCi.C) (%VFCi.CD*)	仅为加 / 减计数功能和单加或单减计数使用。	%VFCi.C: 0 -> 65535 %VFCi.CD: 0 -> 4294967295	CM	读
计数方向 (%VFCi.U)	被系统设置，该位用于加 / 减计数功能表示计数方向： 对于单相加或减计数器， %I0.0.0 决定 %VFC0 的方向，%I0.0.6 决定 %VFC1。 对于两相加 / 减计数器，两个信号的相的不同决定方向。对 %VFC0， %I0.0 为 IB 专用， %I0.1 为 IA 专用。对 %VFC1， %I0.6 为 IB 专用， %I0.7 为 IA 专用。	0 (减计数) 1 (加计数)	CM	读
映像输出 0 使能 (%VFCi.R)	使映像输出 0 有效	0 (不能) 1 (使能)	CM	读和写 (2)
映像输出 1 使能 (%VFCi.S)	使映像输出 1 有效	0 (不能) 1 (使能)	CM	读和写 (2)
阈值 (%VFCi.S0) (%VFCi.S0D*)	该字包含阈值 0 的值。其意义在功能模块的配置中定义。注意：该值必须小于 %VFCi.S1。	%VFCi.S0: 0 -> 65535 %VFCi.S0D: 0 -> 4294967295	CM	读和写 (1)
阈值 S1 (%VFCi.S1) (%VFCi.S1D*)	该字包含阈值 1 的值。其意义在功能模块的配置中定义。注意：该值必须大于 %VFCi.S0。	%VFCi.S1: 0 -> 65535 %VFCi.S1D: 0 -> 4294967295	CM	读和写 (1)

功能	描述	值	%VFC 使用	实时访问
频率测量时基 (%VFCi.T)	配置项为 100 或 1000 ms 时基。	1000 或 100	FM	读和写 (1)
可调节 (Y/N)	配置项，当被选时，允许用户在运行时修改预置值，阈值，和频率测量时基值。	N (不可以) Y (可以)	CM 或 FM	不可以
输入使能 (IN)	用于使当前功能有效或失效。	0 (不可以)	CM 或 FM	读和写 (3)
预置输入 (S)	根据配置，状态为 1 时： ● 加 / 减或减计数：用预置值复位当前值。 ● 单加计数：将当前值复位到 0。 另外，它也初始化阈值输出的操作，及使用户通过操作显示或用户程序对阈值所作的任何修改生效。	0 或 1	CM 或 FM	读和写
输出溢出 (F)	0 到 65535 或从 65535 到 0 标准模式 0 到 4294967295 或从 4294967295 到 0 双字模式	0 或 1	CM	读
阈值 位 0 (%VFCi.TH0)	当前值大于或等于阈值 %VFCi.S0 时置为 1。建议在程序中对该位只测试一次，因为它将被实时更新。用户应用程序对该值使用时的有效性负责。	0 或 1	CM	读
阈值 位 1 (%VFCi.TH1)	当前值大于或等于阈值 %VFCi.S1 时置为 1。建议在程序中对该位只测试一次，因为它将被实时更新。用户应用程序对该值使用时的有效性负责。	0 或 1	CM	读

（*）表示一个 32- 位双字变量。，除了 Twido TWDLC・A10DRF 控制器外。
（1）只有调节设置为 1 时才可写。
（2）只有配置后才可以访问。
（3）只能通过应用程序而不是操作显示或活动表编辑器进行读和写访问。
CM = 计数模式
FM = 频率计模式

计数功能描述

超高速计数器 （%VFC）最大工作频率可达 20 kHz，范围从 0 到 65535 标准模式和 0 到 4294967295。通过以下方式对脉冲计数：
表：

功能	描述	%VFC0		%VFC1	
		IA	IB	IA	IB
加 / 减计数器	脉冲应用于物理输入，当前操作 （加计数 / 减计数）由物理输入 IB 的状态给出。	%I0.0.1	%I0.0.0	%I0.0.7	%I0.0.6
加 / 减 2 相计数器	编码器的两相应用于物理输入 IA 和 IB。	%I0.0.1	%I0.0.0	%I0.0.7	%I0.0.6
单加计数器	脉冲应用于物理输入 IA。（IB 未被使用）	%I0.0.1	ND	%I0.0.7	ND
单减计数器	脉冲应用于物理输入 IA。（IB 未被使用）	%I0.0.1	ND	%I0.0.7	ND

与功能模块有关的注意

加计数或减计数操作在脉冲的上升沿完成，且计数模块必须为使能允许。
有两个可选输入用于计数模式：ICa 和 IPres。 ICa 用于捕捉当前值 （%VFCi.V 或 %VFCi.VD）并存储在 %VFCi.C 或 %VFCi.CD 里。如果用到的话，捕捉输入对 %VFC0 被配置为 %I0.0.3， %VFC1 为 %I0.0.4。
当输入 IPres 被激活时，当前值在以下几个方面受到影响：

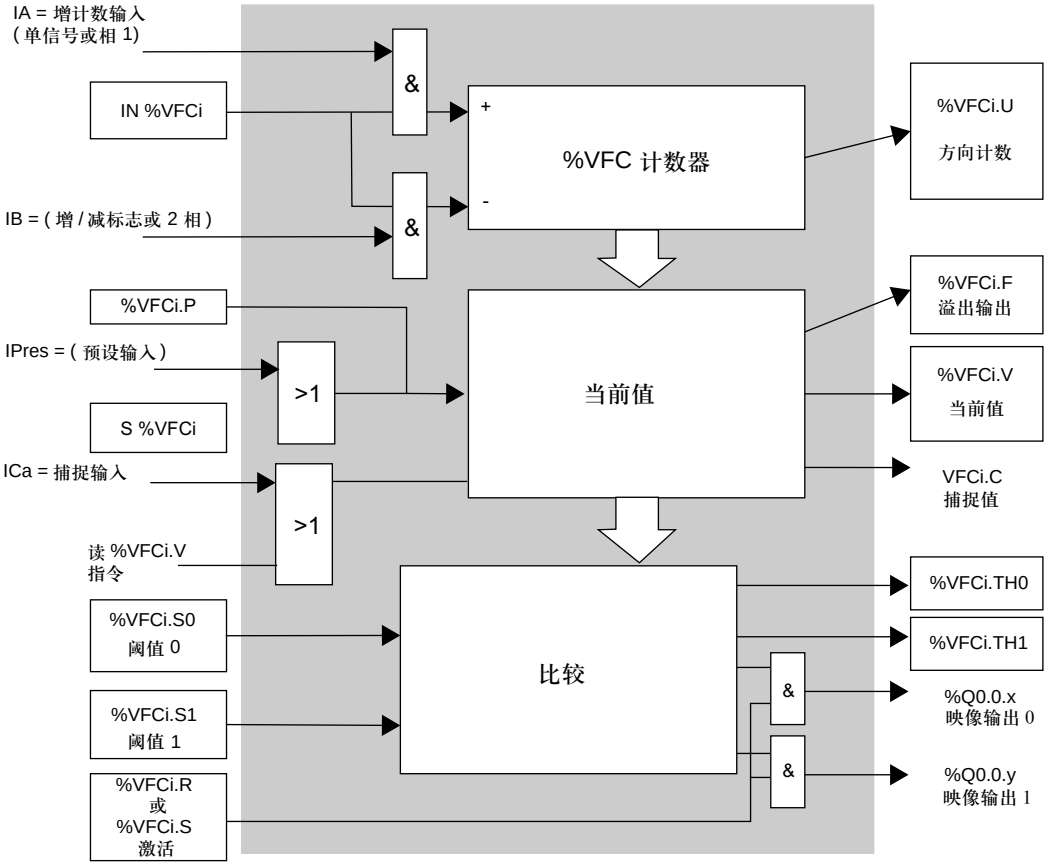
- 对加计数， %VFCi.V 或 %VFCi.VD 被设为 0
- 对减计数， %VFCi.P 或 %VFCi.PD 的内容分别写入 %VFCi.V 或 %VFCi.VD。
- 对频率计， %VFCi.V 或 %VFCi.PD 被设为 0

警告：%VFCi.F 也被置为 0。如果用到话，输入 IPres 对 %VFC0 指定为 %I0.0.2，对 %VFC1 指定为 %I0.0.5。

与功能模块输出有关的注意

对所有功能，当前值与两个值比较 （%VFCi.S0 或 %VFCi.S0D 和 %VFCi.S1 或 %VFCi.S1D）。根据比较结果，如果当前值大于或等于对应阈值，则两个位对象（%VFCi.TH0 和 %VFCi.TH1）被置为 1，否则复位到 0。映像输出 （如果被配置）与这些比较一致被置为 1。注意：可以不配置，或配置 1 个或 2 个输出。
%VFC.U 是 FB 的一个输出，它提供相关计数器变化 （1 对于加， 0 对于减）的方向。

计数功能图 以下是一个标准模式的计数功能图 （对双字模式，你将使用双字功能变量）：



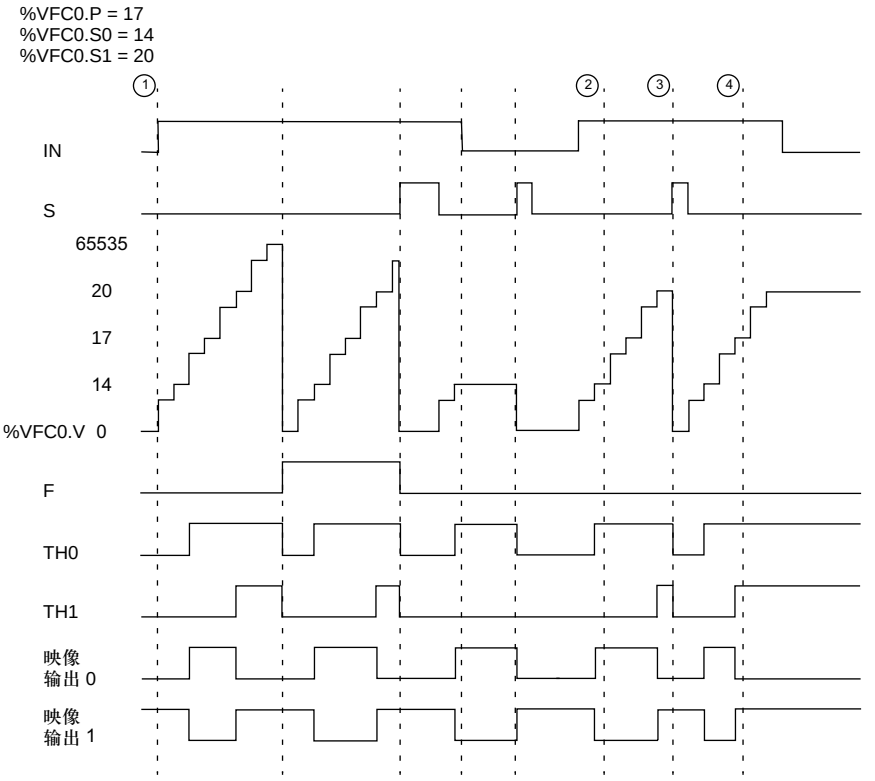
注意：输出管理与控制器周期时间无关。响应时间从 0 变化到 1ms。

单加计数器操作

下面是一个 %VFC 单加计数器模式使用示例。此例中下列配置元素已被配置：%VFC0.P 预置值是 17，低阈值 %VFC0.S0 是 14，高阈值 %VFC0.S1 是 20。

映象输出	值 < %VFC.S0	%VFC0.S0 <= 值 < %VFC0.S1	值 > = %VFC0.S1
%Q0.0.2		X	
%Q0.0.3	X		X

时序图如下：



①：%VFC0.U = 1，因为 %VFC 是一个加计数器

②：把 %VFC0.S1 改为 17

③：S 输入激活使阈值 S1 新值在下一次计数时有效

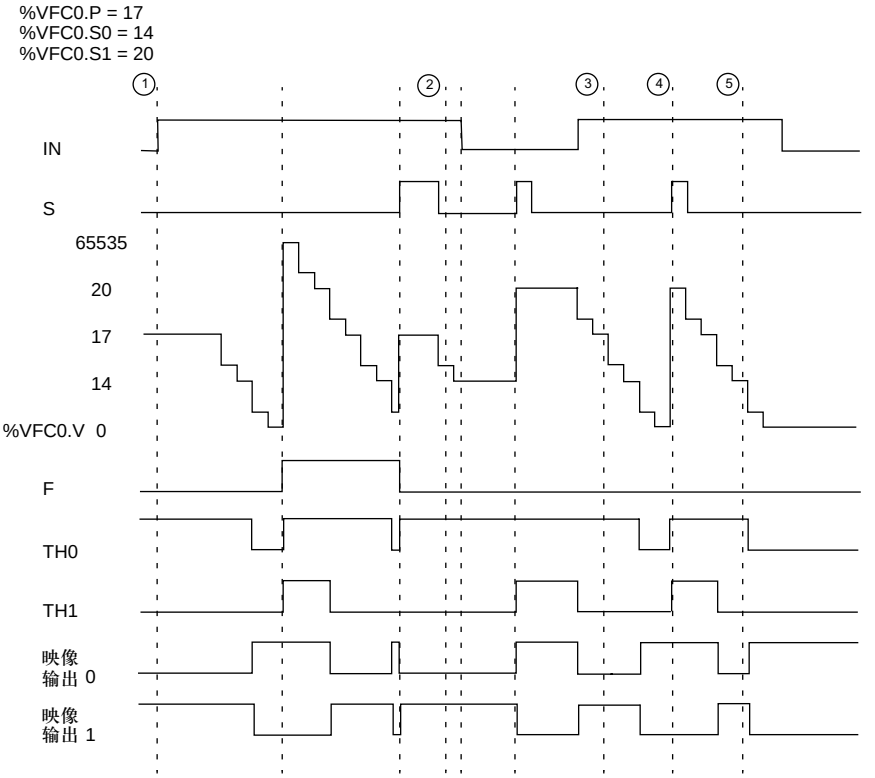
④：捕捉到了当前值，因此 %VFC0.C = 17

单减计数器操作

下面是一个 %VFC 单减计数器模式使用示例。此例中下列配置元素已被配置：%VFC0.P 预置值是 17，低阈值 %VFC0.S0 是 14，高阈值 %VFC0.S1 是 20。

映象输出	值 < %VFC.S0	%VFC0.S0 <= 值 < %VFC0.S1	值 > = %VFC0.S1
%Q0.0.2	X		X
%Q0.0.3		X	

示例：



① : %VFC0.U = 0 因为%VFC 是一个减计数器

② : 把 %VFC0.P 改为 20

③ : 把 %VFC0.S1 改为 17

④ : S 输入激活使得阈值 S1 新值在下一计数中有效

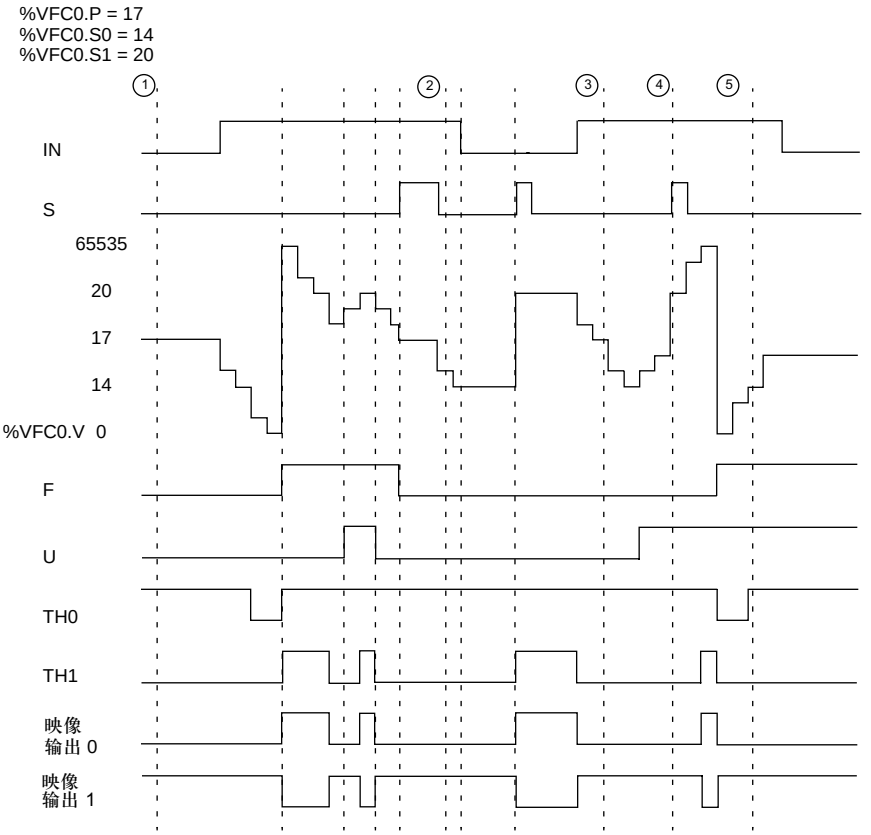
⑤ : 当前值被捕捉到，因此 %VFC0.C = 17

加 - 减计数器操作

下面是一个 %VFC 加 - 减计数器模式使用示例。此例中下列配置元素已被配置：%VFC0.P 预置值是 17，低阈值 %VFC0.S0 是 14，高阈值 %VFC0.S1 是 20。

映象输出	值 < %VFC.S0	%VFC0.S0 <= 值 < %VFC0.S1	值 >= %VFC0.S1
%Q0.0.2			X
%Q0.0.3	X	X	

示例：



① : 输入 IN 被置为 1 且输入 S 被置为 1

② : 把 %VFC0.P 改为 20

③ : 把 %VFC0.S1 改为 17

④ : S 输入激活使得阈值 S1 新值在下一计数中有效

⑤ : 当前值被捕捉到，因此 %VFC0.C = 17

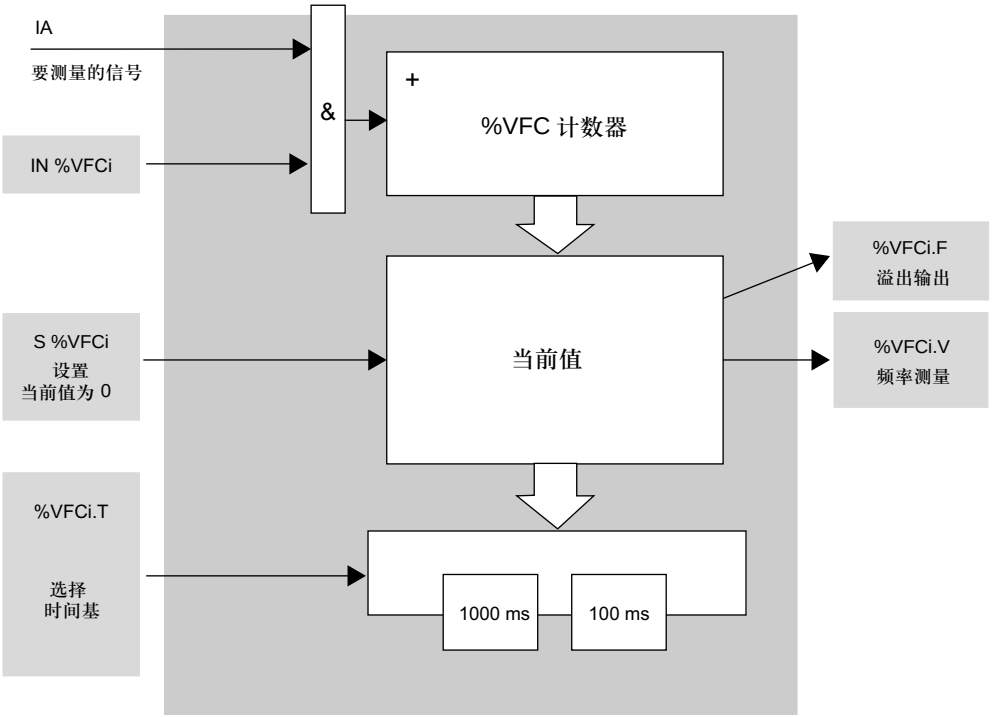
频率计功能描述

%VFC 的频率计功能用于测量输入 IA 上的周期信号的频率，单位为 Hz。可测量的频率范围是 10 到 20kHz。用户可通过一个新的对象 %VFC.T （时基）在 2 个时基之间选择一个。一个是值 100，表示 100 ms 的时基，一个是值 1000，表示 1 秒的时基。

时基	测量范围	精度	更新
100 ms	100 Hz 到 20 kHz	0.05 % 对于 20 kHz, 10 % 对于 100 Hz	每秒 10 次
1 s	10 Hz 到 20 kHz	0.005 % 对于 20 kHz, 10 % 对于 10 Hz	每秒 1 次

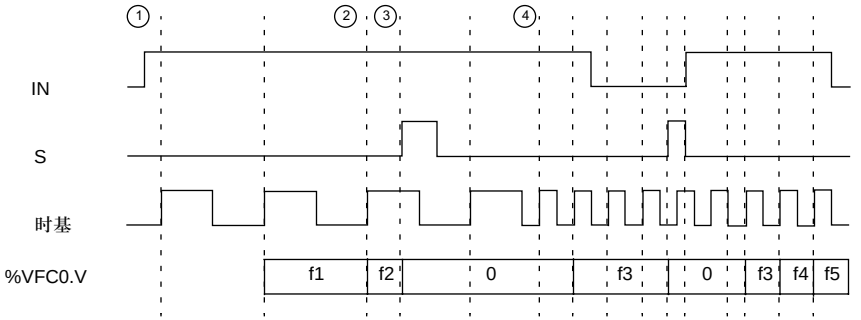
频率计功能图

下面是频率计功能图：



频率计操作

下面是 %VFC 频率计模式使用的时序图示例。



①：第一个频率测量由此开始。

②：当前频率值被更新。

③：输入 IN 为 1，且输入 S 为 1

④：把 %VFC0.T 改为 100 ms: 这个改变取消当前测量，并开始了另一个。

特殊情况

下表包含了 %VFC 功能模块的特殊操作情况列表。

特殊情况	描述
冷启动 （%S0=1）的影响	将所有被用户或用户应用程序配置的 %VFC 属性值复位。
热启动 （%S1=1）的影响	没有影响
控制器停止的影响	%VFC 停止其功能且输出保持在当前状态。

发送 / 接收消息 - 交换指令 （EXCH）

介绍

一个 Twido 控制器配置后可与 Modbus 从设备通信，或以字符模式 （ASCII）发送和 / 或接收消息。
TwidoSoft 为这些通信提供了下列功能：

- EXCH 指令用于发送 / 接收消息
- 交换控制功能模块 （%MSG）用于控制数据交换

Twido 控制器在处理 EXCH 指令时使用指定端口的配置协议。每个通信端口可被分配一个不同的协议。通过添加端口号到 EXCH 或 %MSG 功能 （EXCH1，EXCH2， %MSG1， %MSG2）可以访问通信端口。
另外，TWDLCAE40DRF 系列控制器可使用 EXCH3 指令和 %MSG3 功能在以太网上实现 Modbus TCP 通讯。

EXCH 指令

EXCH 指令允许 Twido 控制器发送和 / 或接收信息到 / 从 ASCII 设备。用户定义一个含有发送和 / 或接收 （高达 250 个数据字节传送和 / 或接收）数据的字表（%MWi:L）。字表的格式在与每个协议有关的段落被定义。消息交换通过 EXCH 指令完成。

Syntax

下面是 EXCH 指令的格式：
[EXCHx%MWi:L]
这里：x = 串行口号 （1 或 2）； x = 以太网端口 （3；：L = 字表总字数 （最大 121）。内部字表 %MWi:L 的值为 i+L <= 255。
Twido 控制器必须在第二个交换指令开始之前通过第一个 EXCHx 指令完成交换。发送几个消息时必须使用 %MSG 功能模块。

> **注意：**要得到关于 Modbus TCP 通信指令 EXCH3 的更多信息，请参阅 *TCP Modbus 消息， p.199*。

交换控制模块 （%MSGx）

介绍

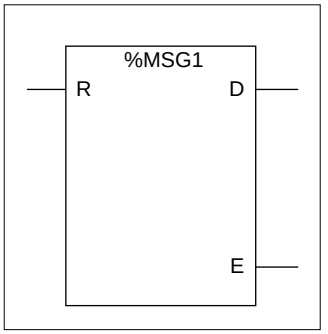
注意：%MSGx 里的 “x” 表示控制器端口：“x = 1 或 2”
- x = 1 或 2，分别表示控制器串口 1 或 2 ；
- x = 3，表示控制器的以太网端口 （仅适用于 TWDLCAE40DRF 控制器）。关于 %MSG3 功能的更多信息，请参阅 *TCP Modbus 消息， p.199 页*。

%MSGx 功能模块管理数据交换且具有三个功能：
- 通信错误校验：
  错误校验核实 EXCH 指令编程的模块长度 （字表）足够包含将被发送的消息长度 （与字表中的第一个字的低位字节的编程长度比较）。
- 多消息协调：
  为了确保多消息发送时的协调性， %MSGx 功能模块提供决定前一条消息何时完成所必需的信息。
- 优先消息发送：
  %MSGx 功能模块允许当前消息的发送被停止，以保证紧急消息的立即发送。

%MSGx 功能模块的编程不是必需的。

图解

下面是一个 %MSGx 功能模块示例。



参数

下表列出了 %MSGx 功能模块的参数。

参数	标识	值
输入 （或指令）复位	R	置为 1 时，通信重新初始化：%MSGx.E = 0 和 %MSGx.D = 1。
通信完成输出	%MSGx.D	状态 1 表示通信在下列情况完成： ● 发送结束 （如果是发送） ● 接收结束 （收到结束字符） ● 错误 ● 模块重启 状态 0 表示请求在处理过程中。
故障 （出错）输出	%MSGx.E	状态 1 表示通信在下列情况完成： ● 命令错误 ● 表配置错误 ● 收到不正确的字符 （速率，奇偶，等等） ● 接收表满 （未更新） 状态 0 表示消息长度和连接都正确。

如果使用一个 EXCH 指令时出错，则 %MSGx.D 和 %MSGx.E 被置为 1，且系统字 %SW63 包含端口 1 的错误代码， %SW64 包含端口 2 的错误代码。见 *系统字（%SW），p.667*。

输入复位 （R）

当输入复位置为 1 时
- 处于发送状态的消息被停止。
- 故障 （出错）输出被置为 0。
- 完成位被置为 1。

现在可以发送一条新的消息。

故障 （出错）输出 （%MSGx.E）

因为通信编程出错，或者是因为消息传送出错时出错输出被置为 1。如果与 EXCH 指令相关的数据模块定义的字节数 （字 1 的低位字节）大于 128 （十六进 +80），出错输出被置为 1。
如果发送一个 Modbus 消息到一个 Modbus 设备中存在问题也将使出错输出被置为 1。这种情况下，用户应该检查连线和目的设备对 Modbus 通信的支持问题。

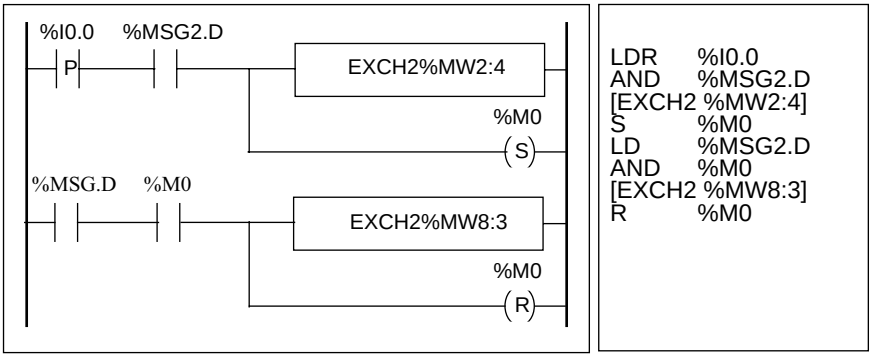
通信完成输出 （%MSGx.D）

当完成输出被置为 1 时， Twido 控制器准备发送另一个消息。当发送多消息时推荐使用 %MSGx.D 位。如果不使用，消息可能被丢失。

多条连续消息的发送

EXCH 指令的执行激活应用程序中的消息模块。消息模块未被激活 （%MSGx.D = 1）时消息被发送。如果同一循环中发送多条消息，则只有第一条消息被发送。用户需要使用程序来管理多消息的发送。

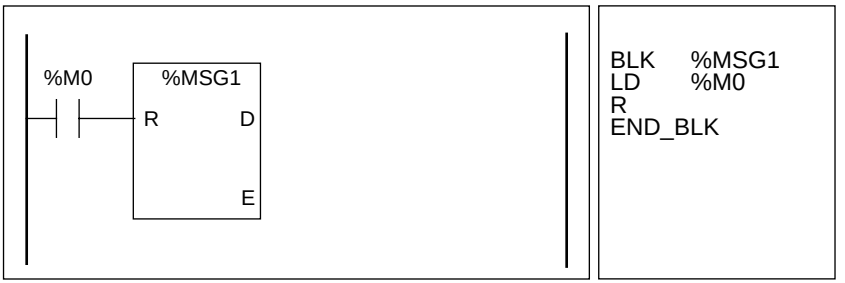
在端口 2 上连续发送两条消息的示例：



交换重新初始化

通过激活输入 （或指令） R 可以取消一个交换。这个输入初始化通信，将输出 %MSGx.E 复位到 0，将输出 %MSGx.D 复位到 1。检测到故障时可以对交换重新初始化。

交换重新初始化示例：



特殊情况

下表是 %MSGx 功能模块的特殊操作情况。

特殊情况	描述
冷启动 （%S0=1）的影响	强制通信重新初始化。
热启动 （%S1=1）的影响	没有影响。
控制器停止的影响	如果消息正在发送，则控制器停止其发送并重新初始化输出 %MSGx.D 和 %MSGx.E。

17.2 时钟功能

概览

本章目的

本节描述了 Twido 控制器的时间管理功能。

本节包含了哪些内容?

本节包含了以下主题：

时钟功能

介绍

带有实时时钟选件 （RTC）的 Twido 控制器具有日历时钟功能，并提供以下功能：

- **调度模块**用于控制预定时间或计划时间执行的动作。
- **时间 / 日期标记**用于给事件分配时间和日期并测量事件的持续时间。

访问 Twido 的日历功能可通过选择调度模块此模块位于 TwidoSoft 软件菜单。另外，日历功能可通过程序来设置。如果控制器关断前电池被连续充电 6 小时以上，则控制器关断后时钟设置可继续工作，最多可达 30 天。

日历时钟为 24 小时格式，并且可以考虑闰年情况。

RTC 修正值

RTC 修正值对 RTC 的正确操作是必需的。每个 RTC 单元都写有它自己的修正值。这个值的配置可在 TwidoSoft 中使用 **RTC** 配置选项，该选项位于控制器操作对话框。

调度模块

介绍

调度模块用于控制在预定的月，日，时间执行的动作。最多可使用 16 个调度模块且不需要任何程序输入。

注意：检查系统位 %S51 和系统字 %SW118 以确定实时时钟（RTC）选件已经安装，见*系统位 （%S）， p.658*。使用调度模块需要 RTC 选件。

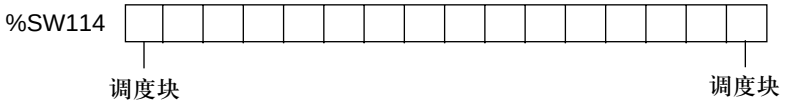
参数

下表列出了调度模块的参数：

参数	格式	功能 / 范围
调度模块编号	n	n = 0 到 15
配置	确认框	选择这个框配置所选的调度模块编号。
输出位	%Qx.y.z	分配的输出被调度模块激活：%Mi 或 %Qj.k。 当当前日期和时间介于活动周期的开始设置和结束设置之间时，输出被置为 1。
开始月	一月到十二月	调度模块的开始月。
结束月	一月到十二月	调度模块的结束月。
开始日期	1 - 31	调度模块的开始日期。
结束日期	1 - 31	调度模块的结束日期。
开始时间	hh:mm	调度模块的开始时间，小时 （0 到 23）和分 （0 到 59）。
结束时间	hh:mm	调度模块的结束时间，小时 （0 到 23）和分 （0 到 59）。
星期几	星期一到星期日	确认框识别激活的调度模块处于星期几。

调度模块使能

系统字 %SW114 的位使能 （位置为 1）或禁止 （位置为 0） 16 个调度模块中的每个模块操作。
调度模块在 %SW114 中的分配：



默认情况下 （或冷启动之后）系统字的所有位被置为 1。程序可以选择使用这些位。

调度模块的输出

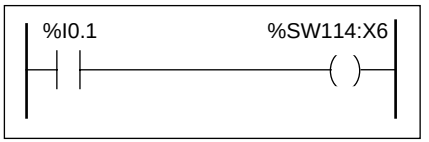
如果同一输出 （%Mi 或 %Qj.k）被分配给几个模块，则最后赋值给这个对象的是每个模块的逻辑或结果 （同一输入可以有几个 “操作范围”）。

示例

下表显示了用于夏季喷洒程序示例的参数：

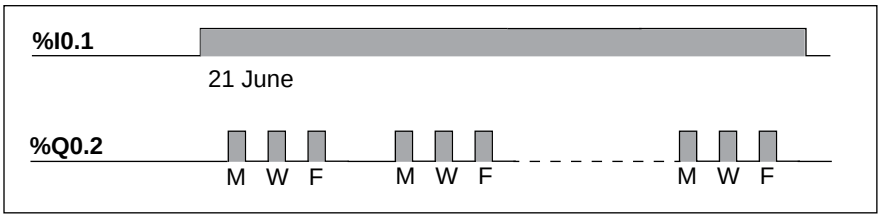
参数	值	描述
调度模块	6	调度模块编号 6
输出位	%Q0.2	激活输出 %Q0.2
开始月	六月	六月开始动作
结束月	九月	九月结束活动
开始日期	21	六月 21 日开始动作
结束日期	21	九月 21 日停止动作
星期几	星期一，星期三，星期五	星期一，星期三和星期五运行动作
开始时间	21:00	21:00 开始动作
结束时间	22:00	22:00 停止动作

使用以下程序，可以通过一个开关或一个湿度监测器连线到输入%I0.1 禁止该调度模块。




```
LD    %I0.1
ST    %SW114:X6
```

下面时序图是输出 %Q0.2 激活显示。



程序中的时间和日期标记

日期和时间的得到都可通过系统字 %SW50 到 %SW53 （见*系统字 (%SW)*，*p.667*）。所以可以通过在当前日期和时间与包含设定值的立即值和字 %MWi （或 %KWi）之间进行比较，实现在控制器程序里对时间和日期标记。

时间 / 日期标记

介绍

系统字 %SW49 到 %SW53 包含 BCD 格式的当前日期和时间 （见 *BCD 码检查，p.463*，这对在外围设备上的显示或发送到外围设备是有用的。这些系统字用来存储事件的时间和日期。（见 *系统字 （%SW），p.667*）。

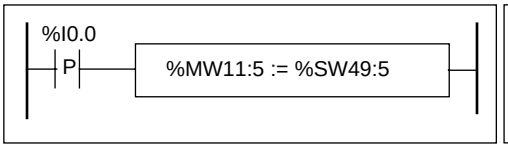
注意：日期和时间也可以通过可选择的操作器显示来设置 （见 *实时时钟计时 p.346*）。

事件日期标记

为了标记事件日期，使用赋值操作，将系统字的内容发送到内部字中，然后处理这些内部字即可 （例如，通过 EXCH 指令发送到显示单元）。

编程示例

下面示例显示了怎样用输入 %I0.1 的上升沿来标记日期。




```
LDR     %I0.0
[%MW11:5 := %SW49:5]
```

一旦检测到一个事件，字表包含：

编码	高字节	低字节
%MW11		星期 1
%MW12	00	秒
%MW13	小时	分钟
%MW14	月	日
%MW15	世纪	年

注意：（1） 1 = 星期一，2 = 星期二，3 = 星期三，4 = 星期四，5 = 星期五，6 = 星期六，7 = 星期日。

字表示例

2002 年 4 月 19 日，星期一， 13:40:30 数据示例：

字	值 （十六进制）	意义
%MW11	0001	星期一
%MW12	0030	30 秒
%MW13	1340	13 时， 40 分
%MW14	0419	04 = 4 月 19 日
%MW15	2002	2002

上次停止的日期和时间

系统字 %SW54 到 %SW57 包含上次停止的日期和时间，且字 %SW58 包含显示上次停止原因的代码，均为 BCD 格式 （见*系统字 （%SW）， p.667*）。

日期和时间设置

介绍

您可以使用下面方法之一更新日期和时间设置：

- TwidoSoft

  使用时间设置对话框。此对话框的获得经由控制器操作对话框。它的显示通过选择控制器操作或控制器菜单。

- 系统字

  使用系统字 %SW49 到 %SW53 或系统字 %SW59。

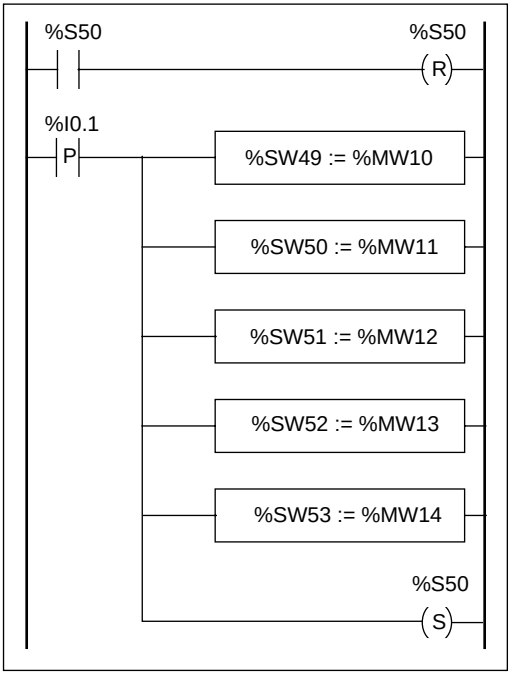
日期和时间设置只有 RTC 选件 （TWDXCPRTC）已经安装在控制器上才可以更新。注意 TWDLCA • 40DRF 系列一体型控制器已经有内置 RTC。

%SW49 到 %SW53 的使用

为了使用系统字 %SW49 到 %SW53 设置日期和时间，位 %S50 必须被置为 1。结果如下：

- 通过内部时钟取消字 %SW49 到 %SW53 的更新。
- 发送字 %SW49 到 %SW53 被写值到内部时钟。

编程示例：




```
LD      %S50
R       %S50

LDR     %I0.1
[%SW49 := %MW10]
[%SW50 := %MW11]
[%SW51 := %MW12]
[%SW52 := %MW13]
[%SW53 := %MW14]
S       %S50
```

字 %MW10 到 %MW14 将包含新的日期和时间，它们通过 BCD 格式表示 （见 *BCD 码视图，p.463*）而且对应于字 %SW49 到 %SW53。

字表必须包含新的日期和时间：

编码	高字节	低字节
%MW10		星期 1
%MW11		秒
%MW12	小时	分钟
%MW13	月	日
%MW14	世纪	年

注意：（1）1 = 星期一，2 = 星期二，3 = 星期三，4 = 星期四，5 = 星期五，6= 星期六，7 = 星期日。

2002 年 4 月 19 日，星期一数据示例：

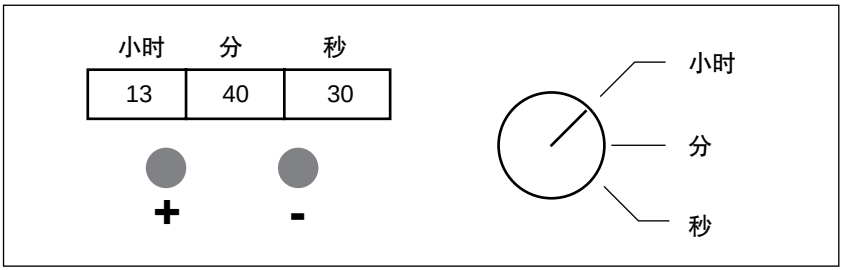
字	值（十六进制）	意义
%MW10	0001	星期一
%MW11	0030	30 秒
%MW12	1340	13 时，40 分
%MW13	0419	04 = 4 月 19 日
%MW14	2002	2002

使用 SW59

更新日期和时间的另一方法是使用系统位 %S59 和日期调节系统字 %SW59。设置位 %S59 为 1 并通过字 %SW59 可以调节当前日期和时间（见*系统字（%SW）*，*p.667*）。%SW59 每个日期和时间单元的增量或减量取决于上升沿。

应用示例

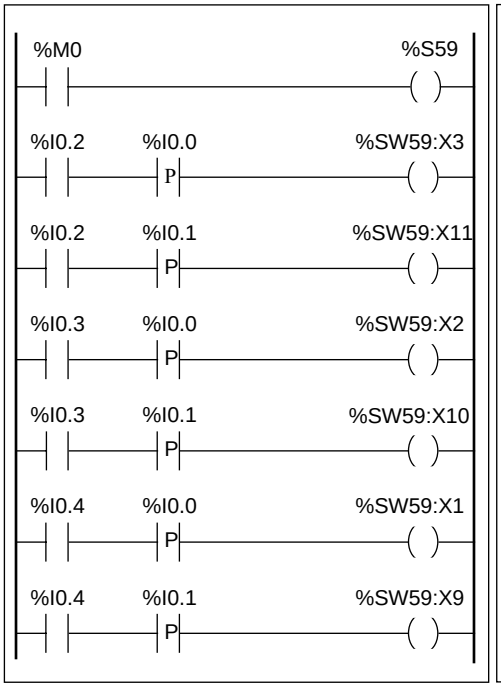
用下面面板来修改内部时钟的小时，分，和秒。



命令描述：

- 分别使用输入 %I0.2，%I0.3，和 %I0.4 使小时 / 分 / 秒开关选择需要修改的时间显示。
- 使用输入 %I0.0 按按钮 “+” 增加所选的时间显示。
- 使用输入 %I0.1 按按钮 “-” 减少所选的时间显示。

下面程序从面板读取输入并设置内部时钟。




```
LD    %M0
ST    %S59
LD    %I0.2          (小时)
ANDR  %I0.0
ST    %SW59:X3
LD    %I0.2
ANDR  %I0.1
ST    %SW59:X11      (分)
LD    %I0.3
ANDR  %I0.0
ST    %SW59:X2
LD    %I0.3
ANDR  %I0.1
ST    %SW59:X10      (秒)
LD    %I0.4
ANDR  %I0.0
ST    %SW59:X1
LD    %I0.4
ANDR  %I0.1
ST    %SW59:X9
```

17.3 Twido PID 快速启动指南 （PID 快速上手指南）

概览

介绍

本节包括 Twido 控制器上提供的开始 PID 控制和自整定功能的信息。

本节包含了哪些内容?

本节包含了以下主题：

文档目的

介绍

通过实例来说明所有正确配置的步骤和设置你的 Twido 控制器的 PID 控制功能，可以引导你快速入门。

注意：在 Twido 上执行 PID 功能不要求特殊知识，只需要一些严密的配置，你就能在尽可能短的时间里得到最佳结果。

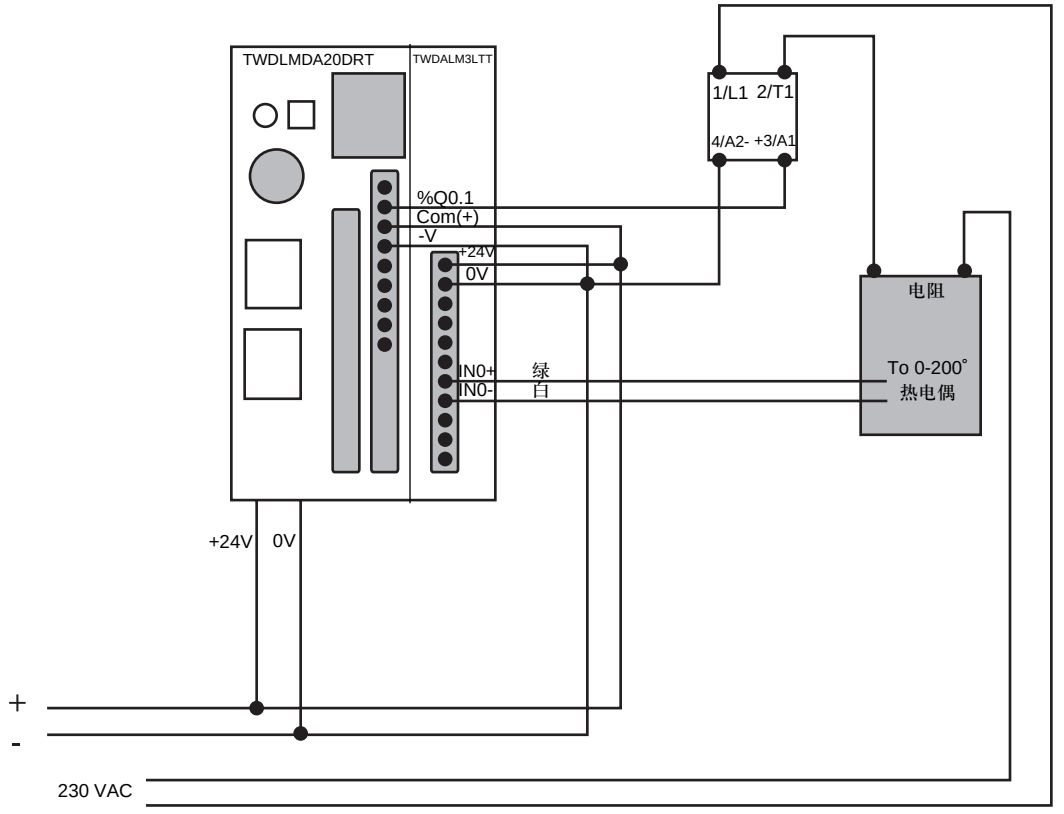
本文档包括：

本文档解释了以下步骤：

步骤	描述
1	用于控制的模拟量通道配置
2	PID 配置先决条件
3	PID 配置
4	控制初始化设置
5	AT + PID 控制设置
6	调试和调节

关于该指南的例子

在这个例子里，我们选择了一个 K 型热电偶 （0-200° ）。
我们将直接通过 PID 控制器控制控制器本体上的晶体管输出的 PWM 功能 （见*步骤 3 - 配置 PID， p.543*）
下图显示了例子中的试验设置：



步骤 1 - 控制用的模拟量通道配置

介绍

一般地，PID 控制器使用模拟量反馈信号 （叫做 “过程值”）去测量要调节的值。这个值可能是个液位，温度，距离，或另外应用的其他值。

模拟量测量信号例子

让我们举一个温度测量的例子。
使用传感器返回一个模拟测量值到控制器，该值决定于被测量的值。对于象 PT100 或热电偶这样的温度传感器，所测量的信号随着目前温度的增加而增加。

如何增加一个模拟量扩展模块

在离线模式，选择本体控制器，增加模拟量模块作为本体扩展。扩展模块编号取决于配置的位置。

如何配置模拟量输入通道

下表描述了配置扩展模块模拟量通道的过程：

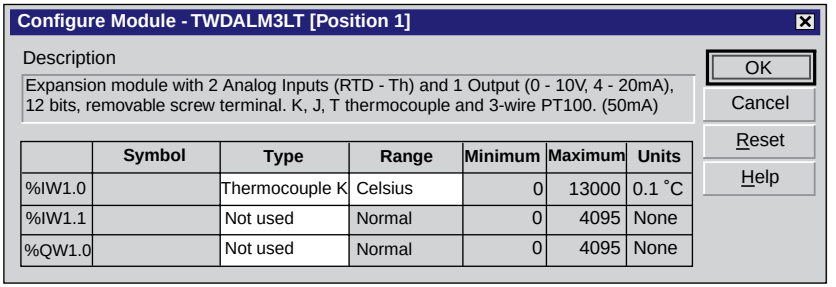
步骤	动作
1	右击扩展总线→增加模块。
2	从列表选择想要的增加的模块。例如，TWDALM3LT 用于使用 PT100 或热电偶测量温度。
3	点击增加，然后点击完成。
4	右击你增加的模块，然后选择配置。
5	在类型栏，根据传感器类型选择相应的输入类型 （K 型热电偶，如果传感器是这个类型的话）。
6	在范围栏，选择传感器测量单位。对温度单位常选择摄氏温度，因为这会作为计算值，通过模拟量模块送回一个真实测量的温度值。
7	给已配置模拟量模块的输入符号提供一个地址。将用于完成 PID 域的填充 （在这个例子中是 %IW1.0，）。
8	如果模拟量输出用于驱动外部控制，对模拟量输出做以上同样的操作。

模拟量通道配置例子

根据使用的测量类型，可能有几种配置，如下所示：

- 对在本文档里使用的应用例子，我们选择了一个 K 型热电偶（0-200° ）。读到的过程值是非常容易理解的 （2000 乘上单位 0.1，计算得出 200° ）。
- 对另外的测量类型，你如国在类型栏选择 0-10V 或 4 - 20 mA，或在范围栏自定义。那么调节值范围 （输入 0 在最小栏和 10000 在最大栏）直接对应过程值（10 V 相当于值 10000）。

下例显示了 K 型热电偶模拟量通道配置：



	Symbol	Type	Range	Minimum	Maximum	Units
%IW1.0		Thermocouple K	Celsius	0	13000	0.1 ˚C
%IW1.1		Not used	Normal	0	4095	None
%QW1.0		Not used	Normal	0	4095	None

步骤 2 - PID 配置的先决条件

介绍

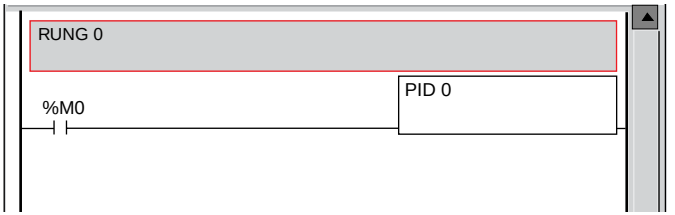
在配置 PID 前，保证下面的阶段已经执行：

阶段	描述
1	在程序里 PID 已使能。
2	扫描周期已经配置

使能 PID 在程序里

PID 控制器必须通过指令在程序里使能。这个指令可以是一个输入位或内部位。在以下例子里， PID 通过指令 %M0 使能：

- 在梯形图：



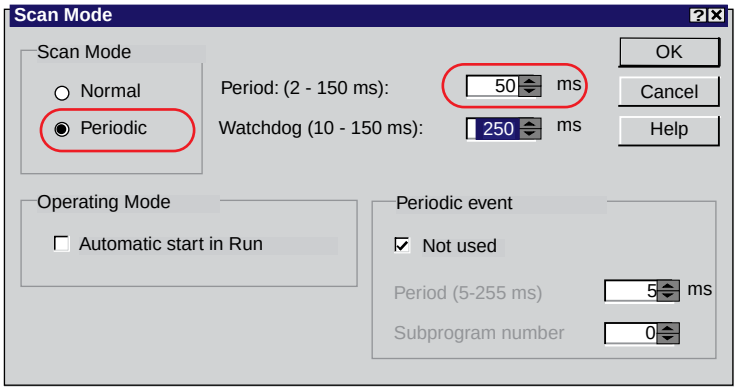
- 在指令列表：

```
----
  0 LD    %M0
  1 [ PID 0 ]
```

注意：保证你使用正确的语法：
在 “PID” 和 PID 序号间有一个空格 （例如 PID< 空格 >0）。

扫描周期配置

当使用 PID 控制功能时，强烈建议把 PLC 扫描模式配置为周期性，下表描述了配置扫描模式的步骤。

步骤	动作
1	从 TwidoSoft 菜单条，选择**程序→编辑扫描模式**。
2	选择**周期框**。
3	如下所示设置周期时间：：  **注意：**周期时间应该调节为程序和期望性能的大小。（50ms 时间是一个较好的选择）。

步骤 3 - 配置 PID

介绍

在这个例子里，我们选择了执行大部分的 Twido PID 控制功能。当然，一些选择不是必需的并且可以简化。

自整定 （AT）

PID 控制器有一个自整定功能以简化调节环的参数设置 （这个功能可在本文档的其他部分作为 AT 查阅到）。

工作模式

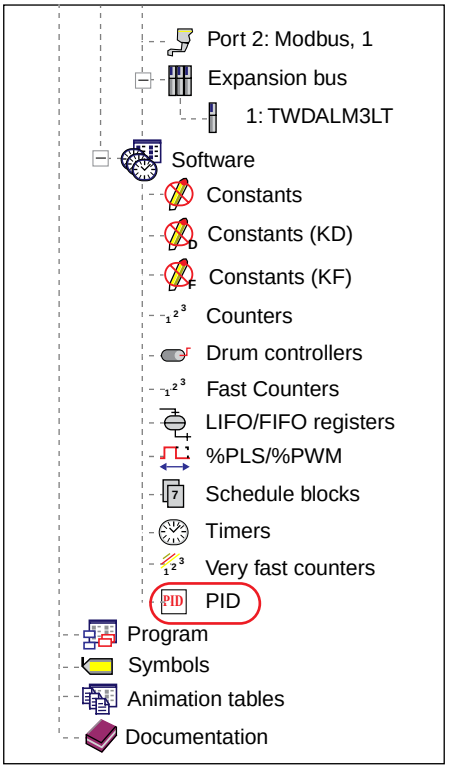
Twido PLC 的 PID 控制功能提供四个独特的工作模式，在 PID 对话框的概要显示页可以选择：

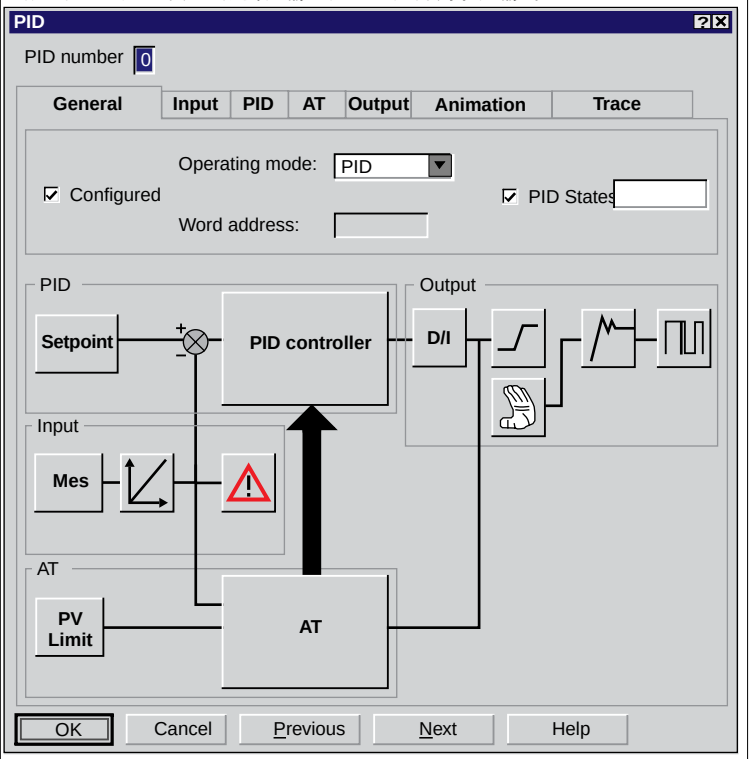
- **PID** = 简单 PID 控制器。
- **AT + PID** = 当 PID 启动后，自整定功能被激活并自动得到参数值：**Kp**， **Ti**， **Td** （**PID** 显示页）和 PID 动作类型 （输出显示页）。自整定结束后，控制器就切换到 PID 模式，并使用 AT 得到的参数来调整设定点。
- **AT** = 当 PID 启动后，自整定功能被激活并且自动得到参数值：**Kp**，**Ti**，**Td**（**PID** 显示页）和 PID 动作类型 （输出显示页）。在 PID 停止后等待。增益值 **Kp**，**Ti**， **Td** （**PID** 显示页）和 PID 动作类型 （输出显示页）将被输入。
- **Word address** （字地址） = PID 工作模式的选择可以被程序控制，通过分配期望值到与选择关联的字地址：
  - %MWxx=1: 控制器工作在简单的 PID 模式。
  - %MWxx=2: 控制器工作在 AT + PID。
  - %MWxx=3: 控制器仅工作在 AT 模式。

  通过字地址配置类型，使用户能够管理 PID 工作模式。通过应用程序，可以达到所需要求。

PID 对话框

下表显示了 PID 对话框和访问不同 PID 设定配置表的过程：

步骤	动作
1	在 TwidoSoft 窗口左边的配置浏览器里双击 **PID** 项，如下图所示： 

步骤	动作
2	PID 对话框出现，可用于输入不同的控制参数设置，如下图所示。在离线模式，出现以下几种表：**概要，输入， PID，自调节，输出：**  **重要的：**表必须按次序输入，他们出现在 PID 对话框：首先概要，输入 PID，自调节然后输出。 **注意：**在在线模式下，屏幕显示另外增加两个表，动态监控和跟踪，分别用于诊断和控制器工作显示。

参数动态修改

对 PID 参数的动态修改 （工作和在线模式），建议在相关的区域输入内存地址，从而避免总是把工作状态切换到离线模式后再改变值。

概要表设置

以下显示了如何在 PID 对话框中设置概要表：

步骤	动作
1	在概要表，选择配置框以激活 PID 并设置以下表。
2	在操作模式的下拉列表，选择期望的工作类型 （见*操作模式，p.543*）。 **在例子里：**我们将选择内存地址模式并输入字 %MW17 在相关区域。 PID 工作模式将取决于 %MW17 的值。

输入表设置

下表显示了如何在 PID 对话框设置输入表：

步骤	动作
1	在输入表，在相关区域输入模拟量地址用于测量。 **本例中：**我们已经选择 **%IW1.0** 作为温度测量。
2	必要的话，确认检查框并在相关区域填入值，以设定低限位和高限位测量报警。 **注意：**输入值可以是个定值 （在相关区域输入）或可修改值 （在相关区域填入内存地址：%MWx）。

PID 表设定

下表显示了如何在 PID 对话框设置 **PID** 表：

步骤	动作
1	在 **PID** 表，输入要使用的值以定义控制器的设定点。一般情况，这个值是一个内存地址或模拟量输入地址。 **在这个例子里：**我们输入 %MW0，将用作设定点字。
2	设定自整定的 **Kp**， **Ti**， **Td** 参数。 **重要的：**如果 **AT** 或 **AT+PID** 模式被选择， Kp， Ti 和 Td 区域使用内存地址来完成是必需的，以使自整定功能自动填入所找到的值。 **在例子里：**我们输入 %MW10 为 Kp， %MW11 为 Ti 和 %MW12 为 Td。 **注意：**原则上，对于一个还没有建立的应用，确定 Kp， Ti 和 Td 的最佳调节值是相当困难的。所以，我们强烈建议你在这些区域里输入一个内存地址，可以使得在线模式输入这些值，从而避免切换到离线模式下才能改变值。
3	输入 PID 采样时间。这个值被控制器使用以获得测量和更新输出。 **在这个例子里：**我们设置 PID 采样周期为 100，即 1s。假设被调节的系统有一个几分钟的时间常数，这个采样周期值也是正确的。 **重要的：**我们建议你设置采样周期为控制器扫描周期的整数倍，并和被调节的系统一致。

表设定为 AT

下表显示了如何在 PID 对话框设置 **AT** 表：

步骤	动作
1	在 **AT** 表，如果你要使用 AT，就确认授权 （Manual mode）框。
2	输入测量限制值。在 AT 期间测量值不能超过这个限制值。
3	输入输出设定点值，该值被发送给控制器输出，以产生 AT。
特别注意	对于关于设定这些值的进一步细节参考 **PID** 功能的自整定章节 **p.582**。
建议	我们强烈建议你在这些区域里输入内存字地址，为使能在线模式输入这些值，从而避免必须切换到离线模式才能改变值。

输出表设定

下表显示了如何在 PID 对话框设定输出表：

	**警告** **系统过载危险** 提醒你手动模式会直接作用于控制器输出。所以，发送一个手动设定点（输出区域）直接作用在开环控制系统时，在这个工作模式应该小心处理。 **如果不注意这个警告将会导致 严重伤害或设备损坏。**

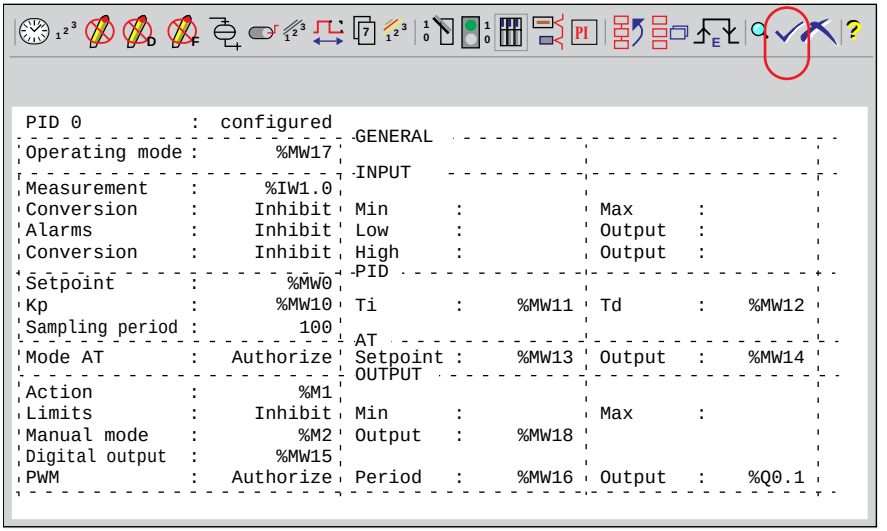
步骤	动作
1	在输出表，从动作下拉列表选择。 这个选择决定于被配置的系统： ● **正向作用：**当变量 （设定点 - 测量值）增加 （冷却控制器）控制器输出减小。 ● **反向作用：**当变量 （设定点 - 测量值）增加 （加热控制器）控制器输出增加。 **重要的：**当使用 AT 功能，列表自动选择位地址。工作模式由 AT 功能决定，在这种情况下在相关区域输入位。
2	必要的话，输入控制器输出的警报门限值。在某些需要管理过程超限报警的应用里，这个功能是必要的。
3	设定手动模式。 下拉列表提供几种选择： ● **限制** = 无手动模式。 ● **自动化** = 控制器仅在手动模式工作。 ● **位地址** = 位值用于改变手动工作模式 （位置 0 = 自动模式，位置 1 = 手动模式）。 **在例子里：**这里我们选择 %M2 激活手动模式选择，并用 %MW18 调节手动设定点的值。
4	可调节数字输出字。这个字被控制器用于发送控制设定点。 可能直接是一个模拟量输出通道地址 （%QWx.y）或一个内存字 （%MWxx）。 **重要的：**当使用 PWM，在这个区域里输入一个内存地址 （%MWxx）。

步骤	动作
5	如果系统需要的话，设置 PWM 输出： 1. 检查自动化框如果你想通过 PWM 发生器器控制系统。 2. 在相关区域输入 PWM 控制周期。 3. 输入输出用于控制 PWM 发生器。我们建议你使用本体控制器上的晶体管输出来实现这个功能 （例如，对 TWDLMDA20DRT 本体控制器为 %Q0.0 或 %Q0.1）。
6	通过点击在屏幕左边的 **OK** 按钮，确认控制器配置。
7	为了配置几个 PID 控制器，点击下一个以增加要设定的 PID 序号。

PID 配置编辑器

一旦你配置了 PID，你必须确认 PID 配置编辑器，编辑器用来概述每个已配置 PID 的所有参数。

为了确认配置编辑器屏幕，点击接收图标在工具栏，如下所示：



步骤 4 - 控制设定初始化

安装先决条件

安装前，你必须按照以下步骤：

步骤	动作
1	连接 PC 到控制器并传送应用程序。
2	切换控制器为运行模式。

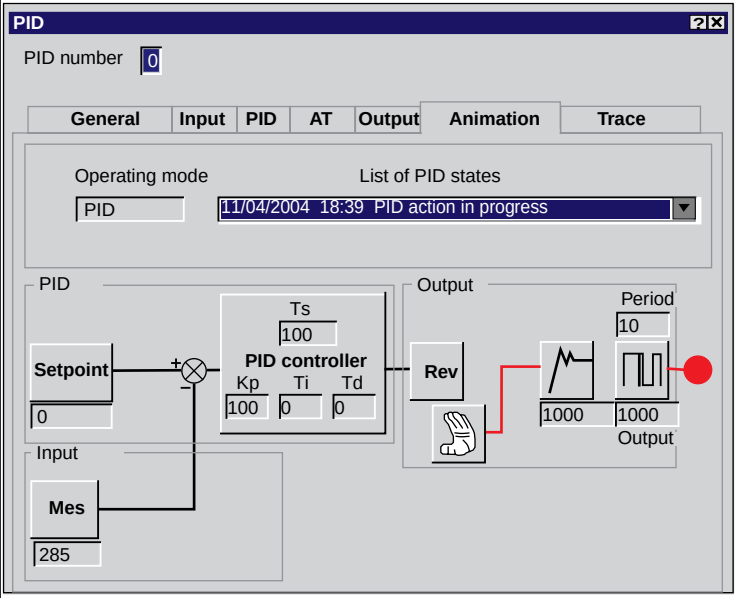
注意：切换控制器为运行模式前，确认机器的工作状态允许进入运行。

程序

必须按以下步骤初始化控制器设定：

步骤	动作
1	建立一个包含需要诊断主要对象的动态数据表。 **在这个例子里：** ● %MW0：设定值， ● %IW1.0：测量值， ● %M0：PID 控制器使能， ● %M1：PID 控制器动作类型 （由 AT 功能设置）， ● %M2：自动或手动模式选择， ● %MW10 到 %MW12：PID 控制器系数， ● %MW13：AT 模式下不能超出的测量限制， ● %MW14：AT 模式下控制器输出设定点， ● %MW15：PID 控制器离散输出 （由控制器输入）， ● %MW16：设置 PWM 周期， ● %MW17：PID 控制器工作模式选择， ● %MW18：与 %M2 位选择相关的手动设定点。
2	检查 %IW1.0 里被测量值的一致性。 **在这个例子里：** **1.** 当系统稳定并冷却时，获得测量值 248。 **2.** 这似乎是一致的，我们在温度和所读的值之间有一个乘法系数 10。我们也可以通过外部来影响测量以确保所读的值是一致的 （在探针附近增加温度，检测测量值也增加）。 **注意：**这个测试非常重要，因为控制器工作依赖于测量精度。 **3.** 如果你怀疑测量精度的话，把控制器设置为停止模式并检查模拟量模块的输入接线 （电压计或电流计对应输入 0-10V / 4-20mA，电阻计对应 PT100 （100 Ω在 20° ）或热电偶 （几十Ω）： ● 首先从模拟量模块端子上断开探针。 ● 检查有没有把线接反 （连接到输入线， PT100 补偿电缆的颜色）。 **警告：**IN0 和 IN1 输入通道有一个共用的端子 （-）。 ● 检查模拟量模块有 24VDC 供电。 ● 检查 4-20 mA 输入传感器已经得到供电。Twido 模拟量输入模块不共用一个电流源。

步骤	动作
3	启动 PID 控制器为手动模式，逐步增加 AT 功能需要的限制值。 设定控制器为手动模式： **1.** 切换控制器为运行模式。 **2.** 输入动态表里以下值的内存地址： ● %M2：手动模式选择 = 1， （M2=1 => 手动模式， M2=0 => 自动模式）， ● %MW16：PWM 周期设定 = 10， ● %MW17：PID 控制器工作模式选择 = 1 （仅用于 PID）， ● %MW18：相关 %M2 位选择手动设定点 = 1000。 这个设定点值可能会选择多次，如果系统重新回到初始状态。 在这个例子里：我们选择了值 1000，对应一个增加的平均温度值（2000 相对应于 200°）。当冷却时，系统以 250 开始。
4	确认控制器在运行模式。 （%M0：控制器确认 = 1，在动态表中输入）。
5	在配置浏览器里双击 **PID** 项。

步骤	动作
6	对要监控的 PID 激活动态监控表，并在以下屏幕检查动态匹配：  **注意：**在 PID 控制器被使能的情况下，屏幕才会刷新 （同时应用也要设置为运行）。
7	激活跟踪表对想要的 PID 号，然后： **1.** 设置滚动时间，从下拉列表选择 15 分钟，以观察测量信号过程的轨迹。 **2.** 检查测量值保持在系统可以接收的范围内。测量值的增加可以在跟踪表内被监控到。当它稳定后，可以读到相应的值 （例如， 350 计算对应 35° ，或与初始温度相比增加了 10° ）。

步骤	动作
8	如果我们看到执行器没有受到控制，检查输出回路： ● 对模拟量输出，从模拟量模块检查输出电压或电流。 ● 对 PWM 输出，检查： 　● 有关的输出 LED 是亮的 （如 %Q0.1，在这个例子里）， 　● TWDLMDA20DRT 本体输出的电源和 0V 回路接线， 　● 执行器的供电。
9	在动态监控表里输入以下值来关掉 PID 显示屏幕并停止手动模式： ● %M0：使能控制器 = 0 （停止 PID 控制器） ● %M2：自动或手动模式选择 = 0 （停止手动模式） ● %MW17：PID 控制器工作模式选择 = 0 ● %MW18：与 %M2 位选择相关的手动设定点 = 0。

步骤 5 - 控制设定 AT + PID

介绍

在本节，我们将要看到如何配置控制器以启动 AT+PID 工作模式。在这个工作模式下，控制器将自动调节控制器系数 Kp， Ti， Td。

注意：按照顺序，系统不可以受到任何由于外部变化将影响最终调节的任何干扰。在运行 AT 之前，也要确保系统是稳定的。

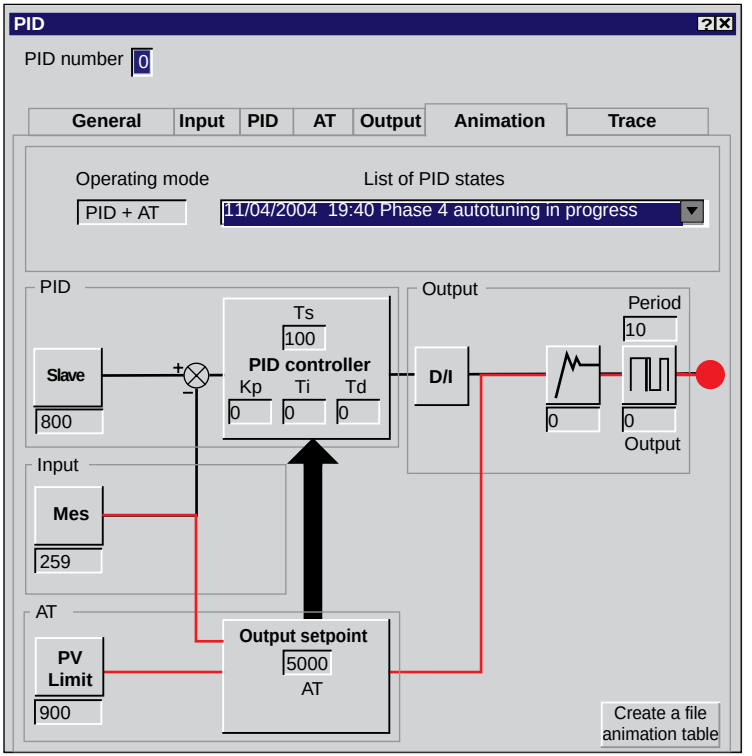
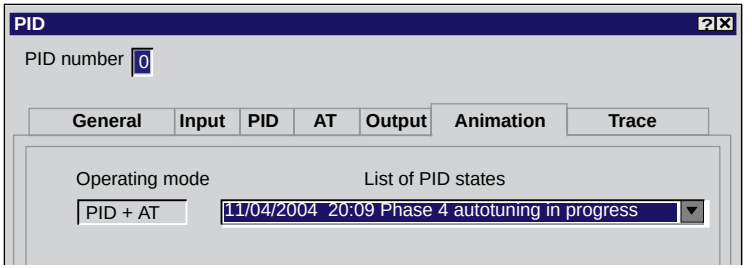
Kp， Ti 和 Td 设置提醒

对可能在 AT+PID 模式下的工作，必须符合以下两种条件：

- 此 Kp， Ti， Td 系数必须配置为内存地址 （%MWxx）。
- 此动作类型在输出表必须设置到一个内存位地址 （%Mxx）。

为了设置控制器为 AT+PID 模式，如下处理：

步骤	动作
1	输入或检查动态表里有以下内存变量地址： ● %M2：自动或手动模式选择 = 0， ● %MW0：PID 控制器设定点 = 600 （在这个例子里，在自整定后，这个值将被控制器用来维持为 60°）， ● %MW10 到 %MW12：PID 控制器系数 （开始为 0，自整定后会有相应数据填入）， ● %MW13：在 AT 模式不能超过的测量限制 = 900 （在这个例子里，如果超过 90° 将会产生一个错误）， ● %MW14：在 AT 模式控制器输出设定点 = 2000 （从手动模式下的测试）。这是改变应用到过程输出的步骤。在 AT 模式，输出设定点直接作用于控制器输出。 该值可以是一个内部字 （%MW0 到 %MW2999），一个内部常量 （%KW0 到 %KW255）或一个立即值。因此该值必须在 0 和 10,000 之间。 **注意：**输出自整定设定点必须总是比上次作用于过程的输出要大。 ● %MW15：PID 控制器离散输出 （由控制器输入）， ● %MW16：PWM 周期设定 （为 10，之前已设定）， ● %MW17：PID 控制器工作模式选择 = 2 （AT + PID）， ● %MW18：与 %M2 位相关选择的设定点 = 0。
2	配置 Twido 控制器的扫描为周期扫描模式 （周期模式）。
3	设定自整定的 Twido 控制器扫描周期时间，这个时间是 PID 控制器的采样周期（Ts）值的一个严格整数倍。 **注意：**关于如何确定采样周期的进一步细节，见*自整定的必要条件 p.602 和确定采样周期 （Ts）的方法 p.603*。
4	确认控制器在运行模式。
5	输入内存位 %M0。 %M0：控制器确认 = 1 （在动态表里）。
6	在配置浏览器里双击 **PID** 项。

步骤	动作
7	对要检测的 PID 号激活动态监控表，并检查屏幕下的动态匹配：  **注意：**PID 控制器的屏幕只有在控制器使能下才会刷新 （并且 API 设为运行）。
8	点击跟踪表并等待系统启动 AT。  **注意：**在 AT 过程变化之前，等待时间可能会持续 10-20 分钟。

计算出来的 Kp，Ti 和 Td 系数的存储

一旦自整定系列完成，被分配给 Kp，Ti 和 Td 系数的内存字会用计算出的值来填入。只要应用程序有效（断电小于 30 天）并且没有执行冷启动（%S0），这些值被写入 RAM 内存并存入控制器。

注意：如果系统被外部波动的影响所干扰，这个值可能很难写入 PID 控制器的设置并且控制器只能切换到 PID 模式。

AT 循环

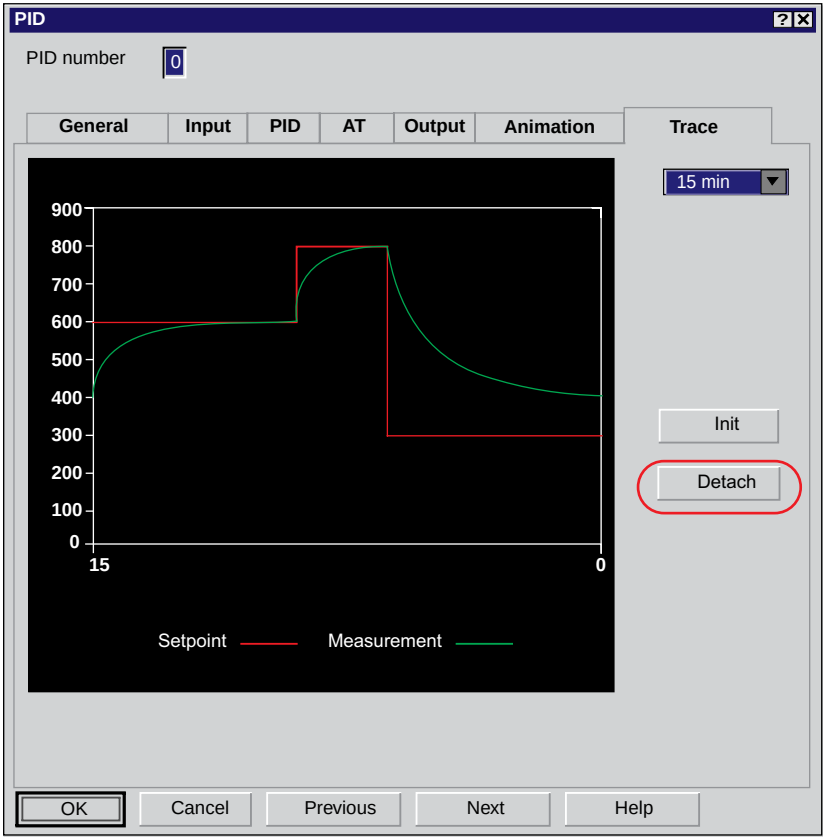
自整定在每次切换到运行或冷启动（%S0）时重复。
所以你应该在程序重启时使用诊断字来进行测试。

步骤 6 - 调试调节

访问动态表

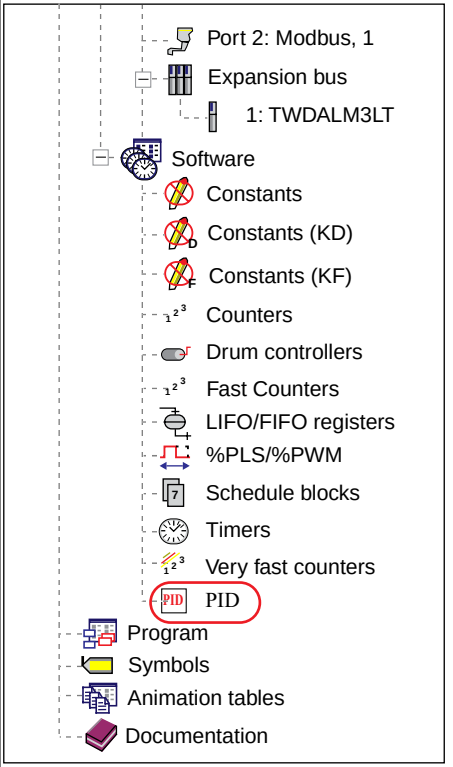
为便于调试系统，当 PID 控制器屏幕在前台时，任何时候动态表都可以被访问。

注意：
当只观察设定点和过程值时，在跟踪表 （见下面窗口的跟踪表），按下分离按钮，动态表可以通过菜单窗口→动态表编辑 - 动态 ... 被访问。



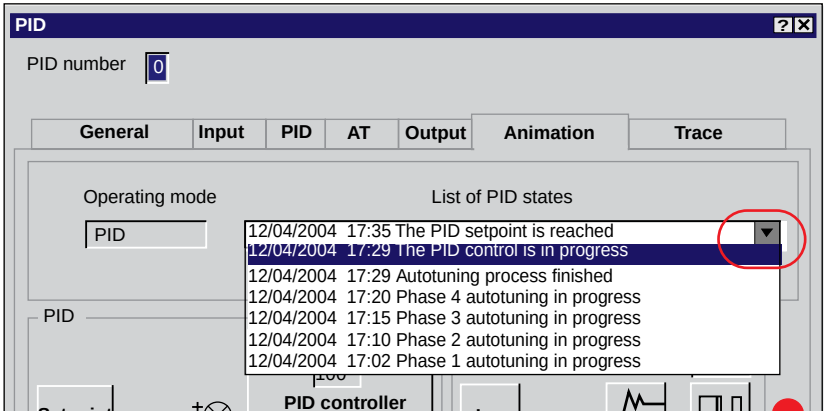
返回到 PID 屏幕

为了返回到 PID 控制器屏幕而不丢失趋势图的历史轨迹，如下过程：

步骤	动作
1	在 TwidoSoft 屏幕左方的浏览器里 （见下面的浏览器窗口）双击 PID 项目： 
2	当 PID 控制器窗口出现时，在概要页中选择想要的 PID 号。

PID 状态历史

对 PID 控制器的动态监控表，你可以从下拉列表里选择，从而访问当前控制器的最新 15 个状态，如下所示：



注意：当 PC 和 TwidoSoft PID 为在线模式时，状态会被存储。

17.4 PID 功能

概览

本节目的

本节描述了 PID 功能的动作，功能和运行情况。

注意：要想快速获得有关 PID 功能的信息，诸如 PID 自整定，请参照 *TwidoPID 快速启动指南，p.536*）。

本节包含了哪些内容?

本节包含了以下主题：

总的介绍

总述

PID 调节功能是一个 TwidoSoft 编程语言功能。
它允许在 TwidoSoft2.0 或更高版本的控制器上进行 PID 调节的编程。
该功能特别适用于：

- 响应需要辅助调节功能的连续处理的需要 （例如：磨光机，压力，等）
- 响应简单调节处理的需要 （例如：金属炉、陶瓷炉、小型制冷组合系统等）

它的安装非常容易因为它集成在：

- 配置
- 和调试

屏幕中与此相关的程序行 （梯形图中的操作模块或指令列表中的 PID 调用）指示所使用的 PID 编号。
梯形表语言中的程序线示例：

注意：任何 Twido 自动控制应用中，可以配置的最大 PID 回路数是 14。

重要特性

重要特性如下：

- 模拟输入，
- 配置的测量线性转换，
- 可配置的上限或下限输入警报，
- 模拟或 PWM 输出，
- 配置的输出，
- 可配置的直接或反向动作。

主要调节环

概览

调节环的工作有三个明显的阶段：

- 获得数据：
  - 通过过程传感器 （模拟量，编码器）得到测量值
  - 设定定值，一般从控制器的内部变量或 TwidoSoft 动态监控表中获得
- 执行 PID 调节算法
- 通过离散量 （PWM）或模拟输出发送相应命令到它们驱动的执行器

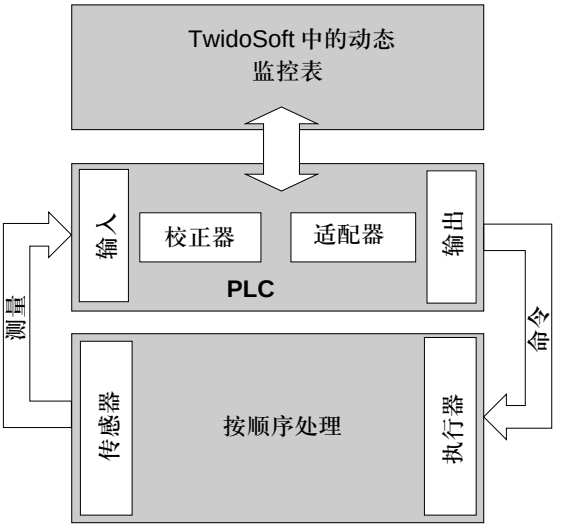
PID 算法由以下生成命令信号：

- 输入模块的采样测量值
- 由操作员或程序确定的设定值
- 不同的修正参数值

修正信号或者被连接到执行器的控制器模拟输出模块直接处理，或者通过控制器的离散输出的 PWM 调节进行处理。

图解

下面是调节环的系统原理图。

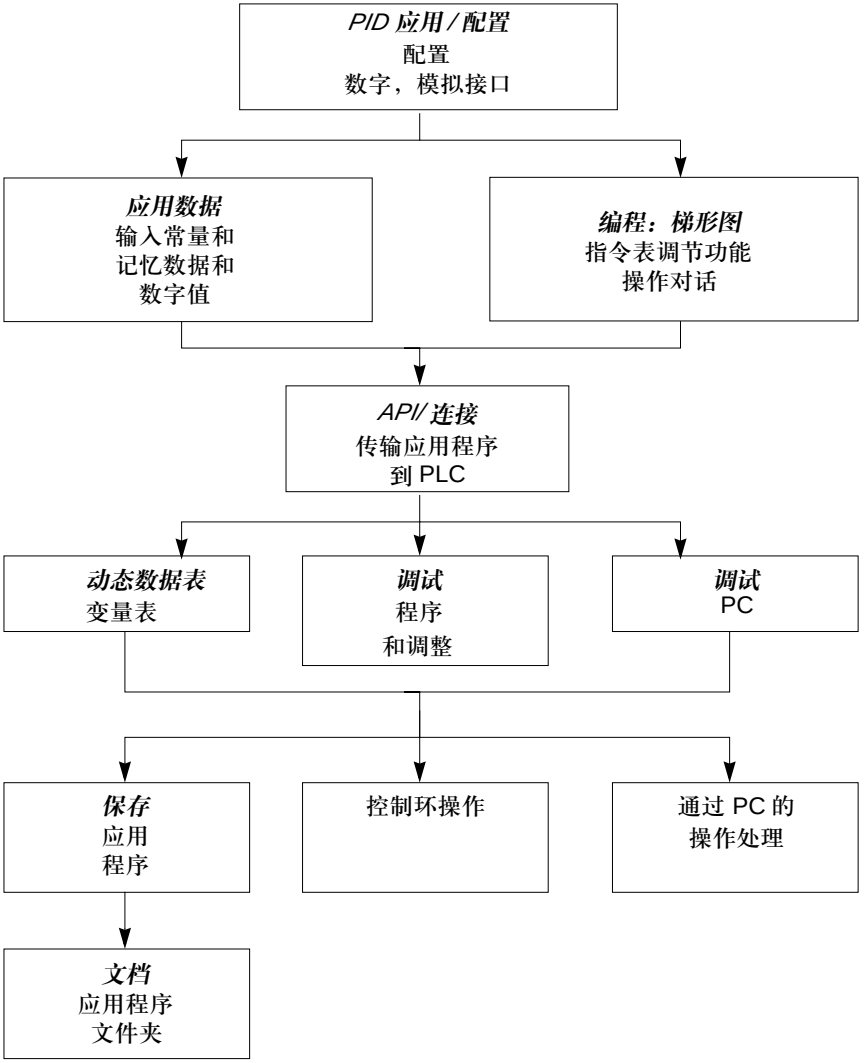


调节应用的开发方法

调节应用的开发方法

下图描述了创建和调试一个调节应用时所执行的全部任务。

注意：顺序定义与您自己的工作方法有关，它仅为一个示例。



兼容和性能

概览

Twido PID 功能适用于 TwidoSoft2.0 和更高版本，这样它的使用要求一些硬件和软件具有后面段落所描述的兼容性。
另外，该功能需要的资源在性能段落中提出。

兼容性

Twido PID 功能适用于版本 2.0 或更高版本软件的 Twido。
如果您的 Twido 使用更早版本的软件，您可以更新它的 firmware，以便使用 PID 功能。

注意：版本 1.0 的模拟输入 / 输出模块不需要更新就可以用作 PID 输入 / 输出模块。

为了在不同的硬件版本上配置和编辑 PID 功能，您必须使用版本 2.0 或更高的 TwidoSoft software。

性能

PID 调节环具有如下性能：

描述	时间
回路执行时间	0.4 ms

PID 功能的细节特性

概要

PID 功能通过模拟测量和默认 [0-10000] 格式的设定值，提供与此相同格式的模拟命令或脉宽调制 （PWM）的数字输出，来完成 PID 修正。
所有的 PID 参数在配置它们的窗口中给予解释。这里，我们将简要概括可用功能，说明测量值，并描述它们在功能流程图中怎样与 PID 结合。

注意：为了使用全部量程 （最佳分辨率），您可以将连到 PID 的模拟输入配置成 0-10000 格式。不过，如果您使用默认配置 （0-4095），控制器也将正常运行。

注意：为使操作正确规范，务必保证 Twido PLC 处于周期扫描模式。从而 PID 函数在每个循环周期都被执行， PID 输入数据采样的周期设置在配置中 （见下表）。

可用功能细节

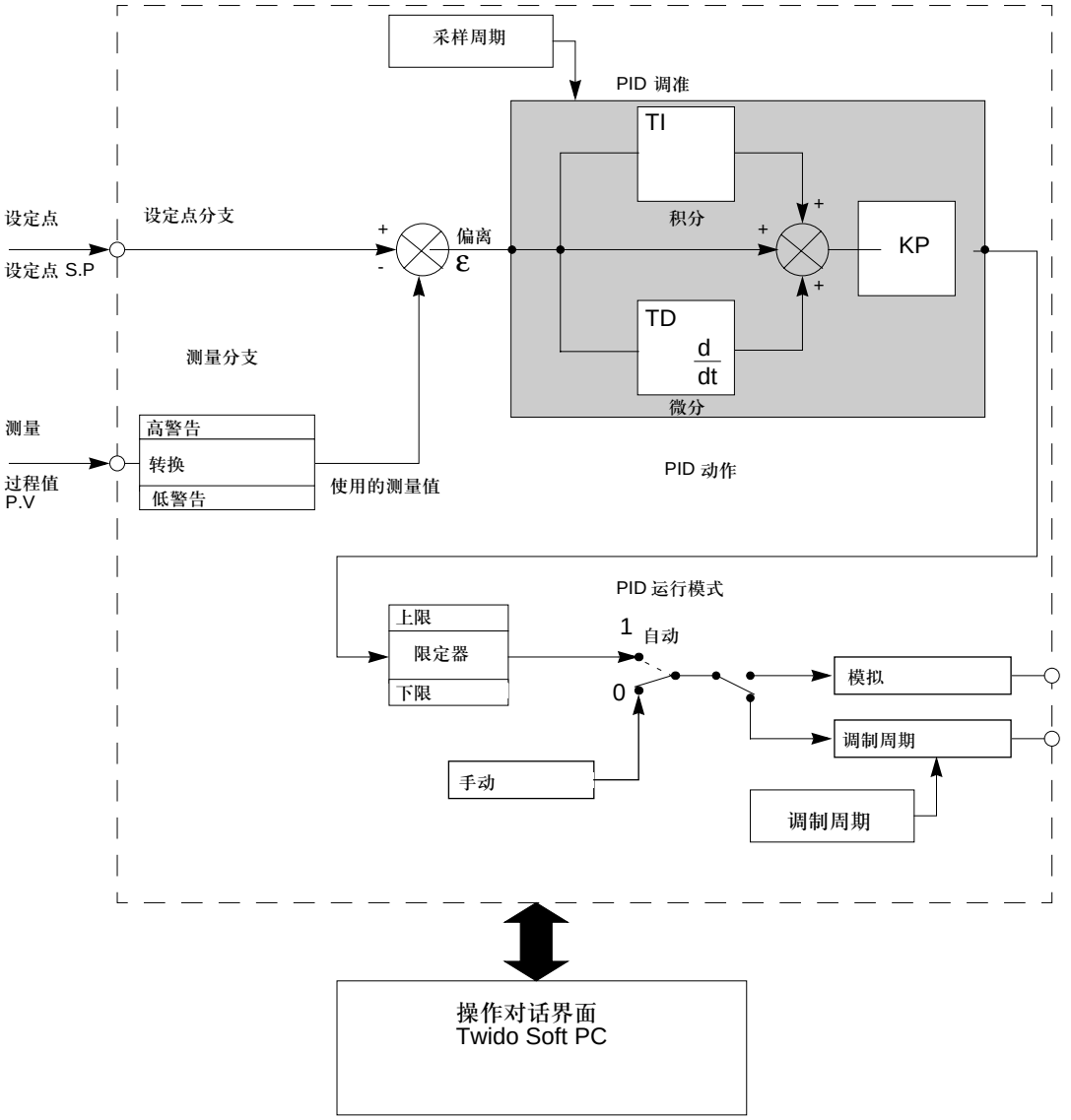
下表指出了各种可用功能及其范围：

功能	范围和注释
输入线性转换	允许您将一个 0 到 10000 格式 （模拟输入模块分辨率）的值转换为一个 -32768 和 32767 之间的值
比例增益	其系数是 100，值介于 1 和 10000 之间。因此它对应一个在 0.01 和 100 之间变化的增益值。 **注意：**如果输入一个非法的增益值 （负数或 0 增益）， TwidoSoft 将忽略该用户设定，并将设为默认值 100
积分时间	使用 0.1 秒的时基，其值介于 0 和 20000 之间。它对应一个介于 0 到 2000.0 秒之间的一个积分时间。
微分时间	使用 0.1 秒的时基，其值介于 0 和 10000 之间。它对应一个介于 0 和 1000.0 秒之间的一个微分时间。
采样时间	使用 0.01 秒的时基，其值介于 1 和 10000 之间。它对应一个介于 0.01 和 100 秒之间的一个采样周期。
PWM 输出	使用 0.1 秒的时基，其值介于 1 和 500 之间。它对应一个介于 0.1 和 50 秒之间的一个调制周期。
模拟输出	值介于 0 和 +10000 之间
过程变量的高限警报	转换值之后设置该警报。如果转换被激活，它可以设为一个介于 -32768 和 32767 之间的值，否则值介于 0 和 10000 之间。
过程变量的低限警报	转换值之后设置该警报。如果转换被激活，它可以设为一个介于 -32768 和 32767 之间的值，否则值介于 0 和 10000 之间。
输出的上限设定值	对一个模拟输出值此限定值介于 0 和 10000 之间。当 PWM 处于激活状态时，此限定值与调制周期的百分比对应。 0% 对应 0，100% 对应 10000。

功能	范围和注释
输出的下限设定值	对一个模拟输出值此限定值介于 0 和 10000 之间。当 PWM 处于激活状态时，此限定值与调制周期的百分比对应。 0% 对应 0，100% 对应 10000。
手动模式	当手动模式被激活时，输出被赋给一个用户设置的确定值。这个输出值介于 0 和 10000 之间 （PWM 输出为 0 到 100%）。
正向或反向动作	正向或反向都可以设定，且直接对输出动作。
自整定 （AT）	该功能提供了自动整定比例增益 （Kp），积分时间 （Ti），微分时间 （Td）及正 / 反向动作参数的功能，以达到对控制过程的优化控制。

注意：对于上表描述的每个功能怎样工作的更为深入的解释，请参见下图。

工作原理　　下图呈现了 PID 功能的工作原理。



注意：参数的描述见上面的表格和配置页面。

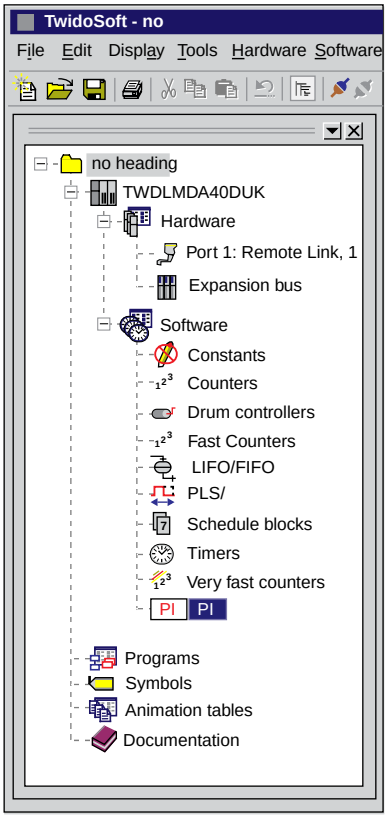
怎样访问 PID 配置

概览

下面图描述了怎样访问 TWIDO 控制器的 PID 配置。

过程

下表描述了访问 PID 配置的过程：

步骤	动作
1	确认处于离线模式。
2	打开浏览器 **结果：** 
3	双击 PID。 **结果：**PID 配置画面显示出来，并处于默认的 General 表 （见 *PID 功能的 General 表，p.574*）。 **注意：**您也可以右键点击 PID 并选择编辑。选项或从菜单中选择软件 → PID 或使用程序 → 配置编辑器→ PID 图标菜单，如果使用后一种方法，选择 PID 并点击放大镜图标来选择一个 PID。

PID 功能总表

概览

当您从浏览器打开 PID，您打开的是 PID 配置窗口。这个窗口允许您：

- 配置每个 PID，
- 调试每个 PID，

当您打开这个屏幕时，根据情况：

- 如果是离线模式：您将进入 General 表且可以进行参数配置，
- 如果是在线模式：您将进入动态监控表且可以进行参数调试和调节。

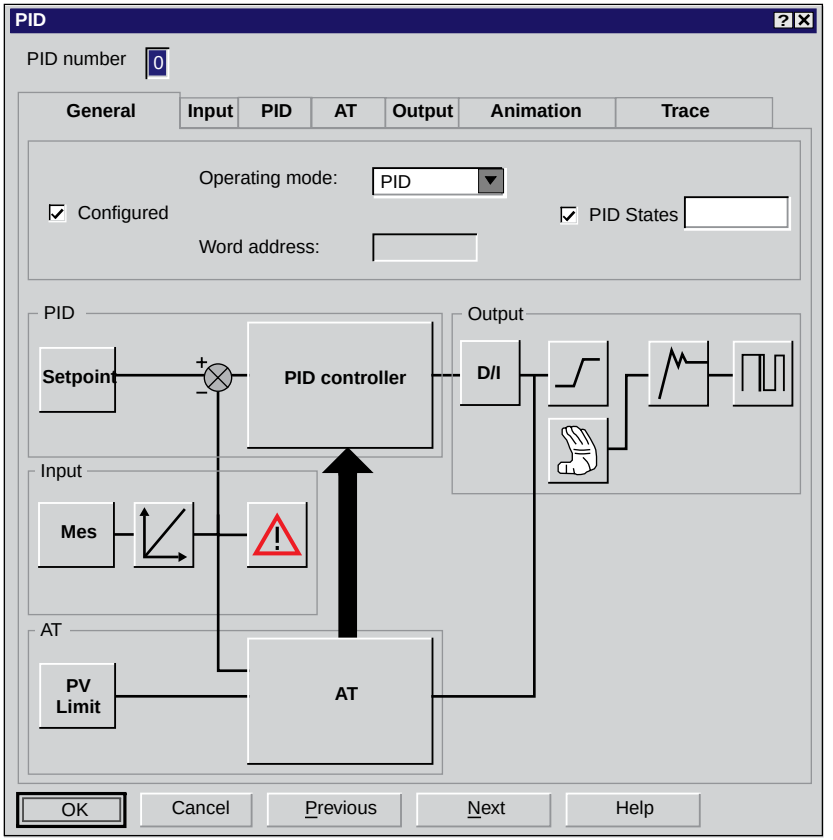
注意：在某些情况下，因为以下两种原因，灰色部分的表项可能不能打开：

- 当前激活的操作模式（离线或在线）不允许进行这些参数的修改。
- 选择了“PID only”（仅选 PID 功能），那么将禁止设定 AT 表的相关参数，因为它们已经无效。

下面段落描述了 General 表。

PID 功能的 General 表

下面屏幕用于输入总的 PID 参数。



说明

下表是对您可以定义的设置的描述。

区域	描述
PID 编号	在这里指定您想配置的 PID 编号。 值介于 0 到 13 之间，每个应用最多 14 个 PID。
配置	必须选中此框才能配置 PID。否则在此屏不能执行任何动作，且即使应用程序中存在 PID，也不能使用。
操作模式	在这里指定希望使用的运行模式，可以选择如下所示的三种模式和一个字地址： ● PID ● AT ● AT+PID ● Word address
Word address（字地址）	可以在这儿指定一个内部字地址 （%MW0 到 %MW2999），用来设定运行模式。该字内可以根据希望的运行模式，写入三个不同数值： ● %MWx = 1 （设为 PID only） ● %MWx = 2 （设为 AT + PID） ● %MWx = 3 （设为 AT only）
PID States （**PID** 状态）	如果选中并激活此功能，可在此对话框中写入一个内存字地址 （%MW0 到 %MW2999），用它来存储 PID 运行和自整定时状态 （更多信息，请参照 *PID 状态和错误代码*）。
图	这个图允许您观看您配置的各种可能的 PID。

PID 的输入表

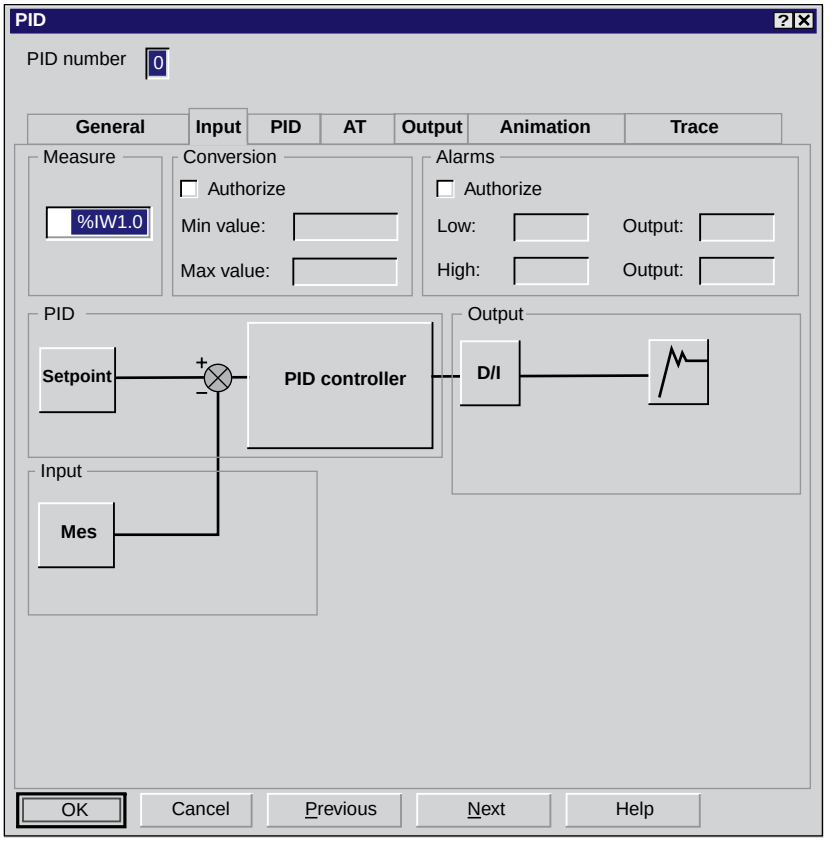
概览

该表用于输入 PID 输入参数。

注意：它在离线模式下被访问。

PID 功能输入表

下面屏幕用于输入 PID 输入参数。



说明

下表是对您可以定义的设置的描述。

说明	描述
PID 编号	在这里指定您想配置的 PID 编号。 值介于 0 到 13 之间，每个应用最多 14 个 PID。
测量值	在这里指定变量，变量包含被控制的过程值。 默认范围是 0 到 10000。您可以指定一个内部字 （%MW0 到 %MW2999）或一个模拟输入 （%Iwx.y）。
转换	如果要对指定为 PID 输入的过程变量进行转换，就要选中此框。 如果选中此框，则最小值和最大值区域可以被访问。 转换是线性的将 0 到 10,000 转换为 -32768 到 +32767。
最小值 **最大值**	指定转换范围的最小和最大值。过程变量将自动在 [最小值到最大值] 范围内进行转换。 **注意：**此最小值必须总是小于最大值。 最小值或最大值可以是内部字 （%MW0 到 %MW2999），内部常量（%KW0 到 %KW255）或介于 -32768 和 +32767 之间的值。
警报	如果要激活输入变量的报警，就要选中此框。 **注意：**报警值应该根据过程变量转换后的相关值进行确定。当转换被激活时，它们必须介于最小值和最大值之间。否则，值将在 0 和 10000。
低 **输出**	测量值小于低限报警值。 该值可以是一个内部字 （%MW0 到 %MW2999），一个内部常量（%KW0 到 %KW255）或一个直接值。输出到达低限设定值时被置为 1。输出可以是一个内部位 （%M0 到 %M255）或一个输出 （%Qx.0 到 %Qx.32）。
高 **输出**	测量值大于高限报警值。 该值可以是一个内部字 （%MW0 到 %MW2999），一个内部常量（%KW0 到 %KW255）或一个直接值。输出到达高限设定值时被置为 1。输出是一个位址，输出可以是一个内部位 （%M0 到 %M255）或一个输出 （%Qx.0 到 %Qx.32）。
图	这个图允许您观看您配置的各种可能的 PID。

PID 功能 PID 表

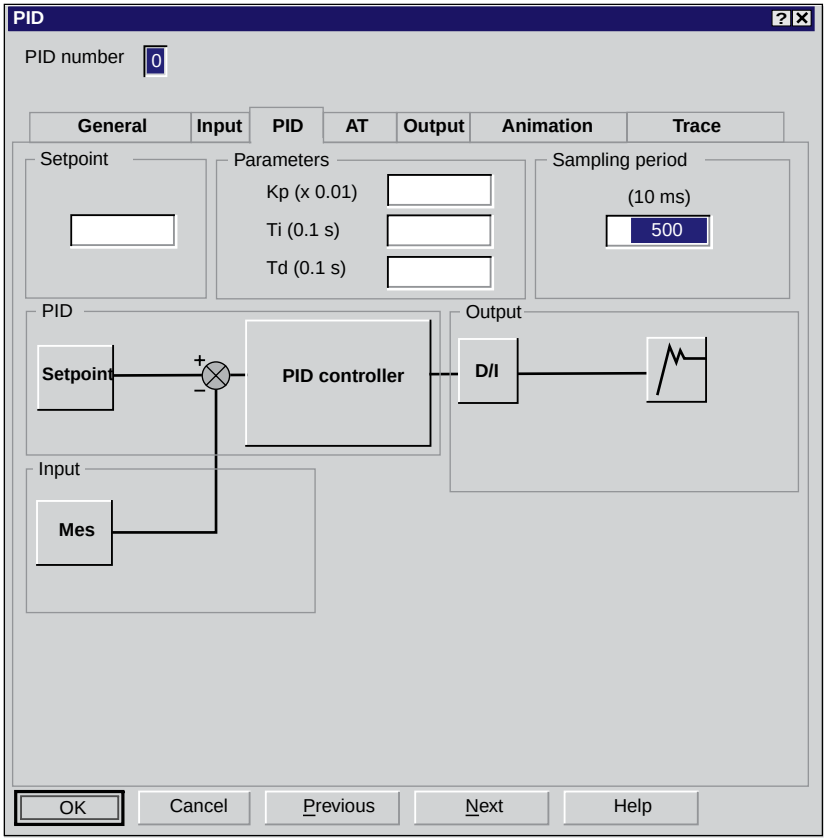
概览

该表用于输入 PID 内部参数。

注意：它在离线模式下被访问。

PID 功能 PID 表

下屏用于输入 PID 参数。



说明

下表是对您可以定义的设置的描述。

区域	描述
PID 编号	在这里指定您想配置的 PID 编号。 值介于 0 到 13 之间，每个应用最多 14 个 PID。
设定值	在这里指定 PID 设定值。该值可以是一个内部字 （%MW0 到 %MW2999），一个内部常量 （%KW0 到 %KW255）或一个立即值。当转换被禁止时该值介于 0 和 10000 之间。否则它必须介于转换的最小值和最大值之间。
Kp * 100	在这里指定乘上 100 后的 PID 比例系数。 该值可以是一个内部字 （%MW0 到 %MW2999），一个内部常量（%KW0 到 %KW255）或一个立即值。参数 Kp 正确的范围是：0 < Kp < 10000。 **注意：**如 Kp 设为 0 （Kp ⩽ 0 是无效）默认值 Kp=100 被自动设定。
TI (0.1 sec)	在这里指定时基为 0.1 秒的积分系数。 该值可以是一个内部字 （%MW0 到 %MW2999），一个内部常量（%KW0 到 %KW255）或一个立即值。 它必须介于 0 和 20000。 **注意：**要取消 PID 的积分功能，设定该系数为 0。
Td (0.1 sec)	在这里指定时基为 0.1 秒的微分系数。 该值可以是一个内部字 （%MW0 到 %MW2999），一个内部常量（%KW0 到 %KW255）或一个立即值。 它必须介于 0 和 10000 之间。 **注意：**要取消 PID 功能的微分功能，设定此系数为 0。
采样时间	在这里指定 PID 采样周期，对应的时基为 10-2 秒 （10 ms）。 该值可以是一个内部字 （%MW0 到 %MW2999），一个内部常量（%KW0 到 %KW255）或一个立即值。 它必须介于 1 （0.01 s）到 10000 （100 s）。
图	这个图允许您观看您配置的各种可能的 PID。

注意：当 AT 功能激活后，Kp，Ti 和 Td 这些参数就不能由用户来设定，它们将由 AT 算法自动进行设定。此时，在这些区域必须输入内存字 （%MW0 到 %MW2999）。
注意：当 AT 功能激活后，不要输入内部常数或立即值，否则，在运行 PID 功能时它将导致产生错误。

PID 的自整定表

概览

要正确设定一个 PID 的各项参数是很麻烦的，很耗时且容易出错。所有这些都使过程控制不是很容易凭经验设定，即使用户很专业。所以，优化整定 PID 的设计有时就很难获得。

PID 的自整定功能，是设计用来自动完成下面四个 PID 项：

- 增益值，
- 积分值，
- 微分值，和
- 正 / 反向动作选择。

所以自整定功能就可以为过程控制提供快速和优化的整定。

自整定功能的要求

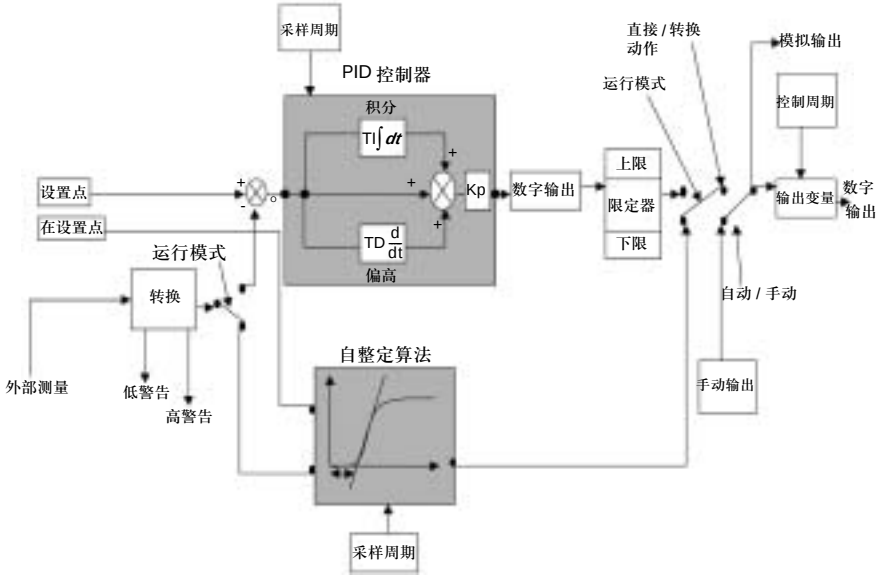
PID 自整定功能通常用来进行温度控制。

一般情况下，应用自整定功能的过程控制需要满足以下要求：

- 整个过程的工作范围都是线性的，
- 过程变化响应反映到模拟输出遵循暂态逼近曲线，并且
- 过程变量会有小幅振动。（在温度控制过程中，这意味着过程和环境间不会有反常的高热交换速率）。

自整定功能的工作原理

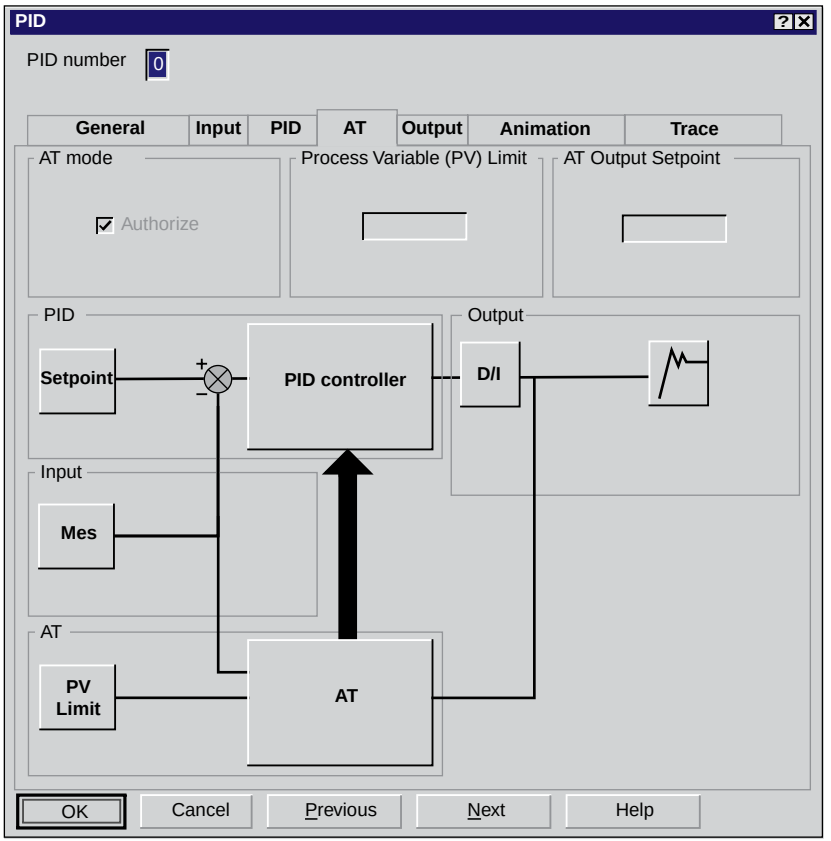
下图说明了自整定功能的工作原理，以及它与 PID 回路的相互影响：



PID功能中的AT表

下面的图片用来显示激活 / 关闭自整定功能，以及如何输入自整定参数。

注意：它仅能在离线模式下可访问。



说明

	**警告** **过程变量 （PV）门限和输出设定值必须谨慎设定。** PID 自整定功能是一个开环过程，它不加任何调整就直接作用于过程控制，除了过程变量 （PV）门限和输出设定值。所以，这两个值必须在过程允许的范围内谨慎设定，以防可能发生的过程过载。 **如果不注意这个警告将会导致 严重伤害或设备损坏。**

下表是对您可以定义的设置的描述。

区域	描述
授权	如果要激活自整定功能，就要选中此框。 有两种方法可以选中此框，这要看 PID 功能 General （总表）表中的工作方式设定，是手动设定还是通过一个字地址。 ● 如果设定操作模式为 PID+AT 或 AT 或总表（见 *PID 功能总表, p.574*），于是批准项被自动选中，并变灰 （它不能被取消）。 ● 如果通过 %MWx（%MWx = 2：PID+AT ； %MWx = 3：AT）设定工作方式，这时就要手动选中批准框，才能允许输入自整定参数。 **结果：**以上两种情况下， AT 表中的所有配置项都处于激活状态，必须按照要求输入设定值和输出值。
过程变量的限制 （PV）	指定自整定过程中可测量的过程变量不能超过的门限值，它可以是一个内部字（%MW0 到 %MW2999，这取决于系统内存字的多少），一个常数 （%KW0 到 %KW255），或者一个立即值。 当转换被禁止时该值介于 0 和 10000 之间。 否则，它该值须处于转换的最小值和最大值之间。

区域	描述
AT 输出设定值	在这里指定自整定的输出值。该值是应用到过程中的阶变值。 该值可以是一个内部字 （%MW0 到 %MW2999），一个内部常量 （%KW0 到 %KW255）或一个立即值。 该值介于 0 和 10000 之间。 **注意：**自整定中的输出设定值总是比作用于过程的最后一个输出值大。

注意：当自整定功能激活后，在以下 PID 功能中的参数将不能使用常数（%KWx）和自然数：

- **Kp**， **Ti** 和 **Td** 这些参数必须设为内存字 （%MWx）在 PID 表内；
- 动作域自动设为 “地址位”在 OUT 表内；
- 位框中必须填入适当的内存位 （%Mx）在 OUT 表内。

计算 Kp，Ti，Td 系数

一旦自整定完成，计算出的 Kp， Ti 和 Td PID 系数：

- 分别存储到内存字 （%MWx）中，并且
- 可以通过动态监控表查看，仅在 TwidoSoft 在线模式时。

PID 的输出表

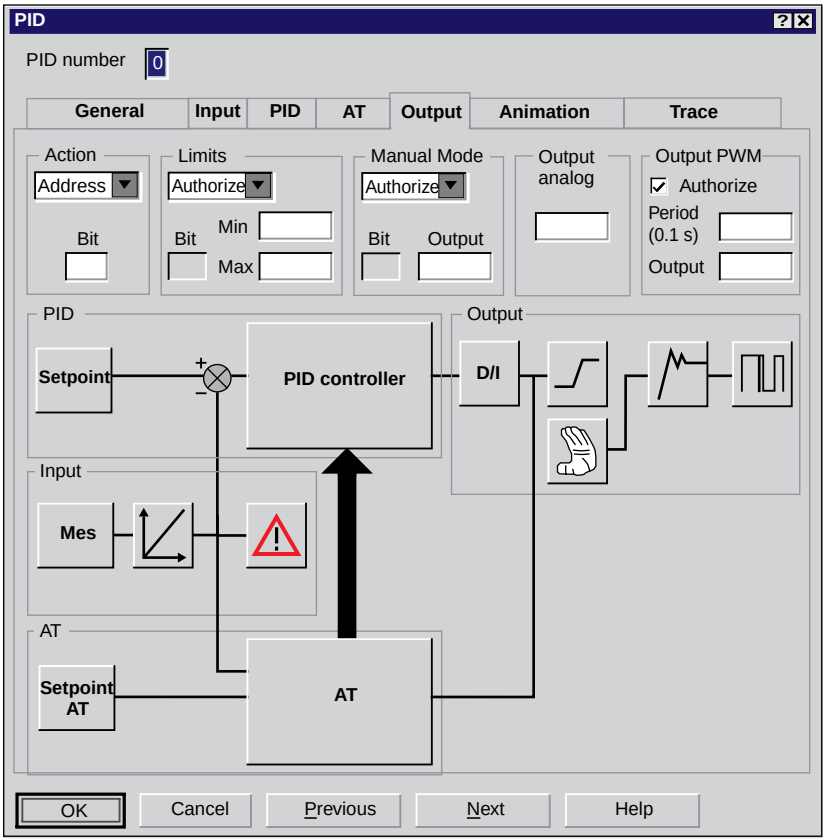
概览

该表用于输入 PID 输出参数。

注意：它在离线模式下被访问。

PID 的输出表

下屏用于输入 PID 输出参数。



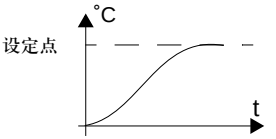
说明

下表是对您可以定义的设置的描述。

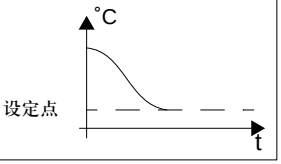
区域	描述
PID 编号	在这里指定您想配置的 PID 编号。 值介于 0 到 13 之间，每个应用最多 14 个 PID。
动作	在这里指定 PID 动作方向类型。有三个选项可选：反向，正向或位址。 如果您选择了位址，您可以通过程序使用相关位修改这个类型，这个相关位是一个内部位 （%M0 到 %M255）或一个输入 （%Ix.0 到 %Ix.32）。 当该位被置为 1 时动作是正向的，否则是反向的。 **注意：**当激活自整定功能后，自整定算法将自行决定动作是正向还是反向。这时，从该下拉菜单中只有一种选项可用：位址。必须在位文本框内填入一个内部位 （%M0 到 %M255）。不要输入内部常数或立即值到位文本框内，否则将导致运行错误。
门限位	在这里指定您是否为 PID 输出设置限定值。有三个选项可选：有效，无效或位址。 如果您选择了位址，您可以通过程序修改相关位对有效 （位为 1）或无效 （位为 0）门限进行管理，这个相关位是一个内部位 （%M0 到 %M255）或一个输入 （%Ix.0 到 %Ix.32）。
最小 **最大**	在这里设定 PID 输出的上限和下限值。 **注意：**此最小必须总是小于最大。 最小或最大可以是内部字 （%MW0 到 %MW2999），内部常量 （%KW0 到 %KW255）或介于 1 和 10000 之间的值。
手动模式位 **输出**	在这里指定您是否需要切换 PID 到手动模式。有三个选项可选：有效，无效或位址。 如果您选择了位址，您可以通过程序修改相关位，以切换到手动模式 （位为 1）或切换到自动模式 （位为 0）， 这个相关位是一个内部位 （%M0 到 %M255）或一个输入 （%Ix.0 到 %Ix.32）。 此输出必须包含要赋给模拟输出的值，当 PID 处于手动模式时。 这个输出 （%MW0 到 %MW2999）或一个立即值 [0-10000]。
模拟输出	在自动模式指定 PID 输出。 这个模拟输出可以是 %MW- 型 （%MW0 到 %MW2999）或 %QW- 型 （%QWx.0）。

区域	描述
PWM 输出使能 **时基（0.1s）** **输出**	如果您想使用 PID 的 PWM 功能，请选中此框。 指定调制周期，时基（**0.1s**）。这个周期必须介于 1 和 500 之间，且可以是一个内部字（%MW0 到 %MW2999）或内部常量（%KW0 到 %KW255）。 指定 PWM 输出位作为输出。这可以是一个内部位（%M0 到 %M255）或一个输出（%Qx.0 到 %Qx.32）。
图	这个图允许您观看您配置的各种可能的 PID。

注意：动作域的 Reverse 功能用于达到高设定点（如：加热）



动作域的 Direct 功能用于达到低设定点（如：制冷）



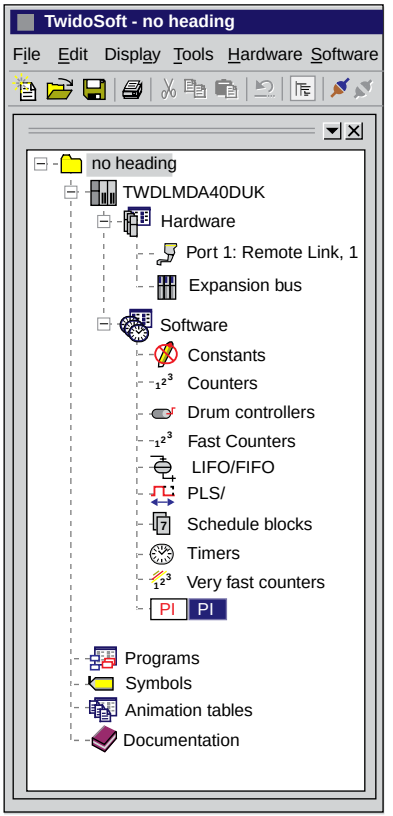
怎样访问 PID 调试

概览

下面段落描述了怎样访问 TWIDO 控制器的 PID 调试屏幕。

过程

下表描述了访问 PID 调试屏幕的过程：

步骤	动作
1	确认处于在线模式。
2	打开浏览器。 **结果：** 
3	双击 **PID**。 **结果：**PID 配置画面显示出来，并处于默认的动态监控 （见 *PID 功能功态监控表， P.592*）。 **注意：**您也可以右键点击 **PID**，并选择编辑选项；或从菜单中选择软件→ **PID** 或使用程序→配置编辑器→ **PID Icon** 菜单，如果使用后一种方法，选择 PID 并点击放大镜图标来选择一个 PID。

PID 功能动态监控表

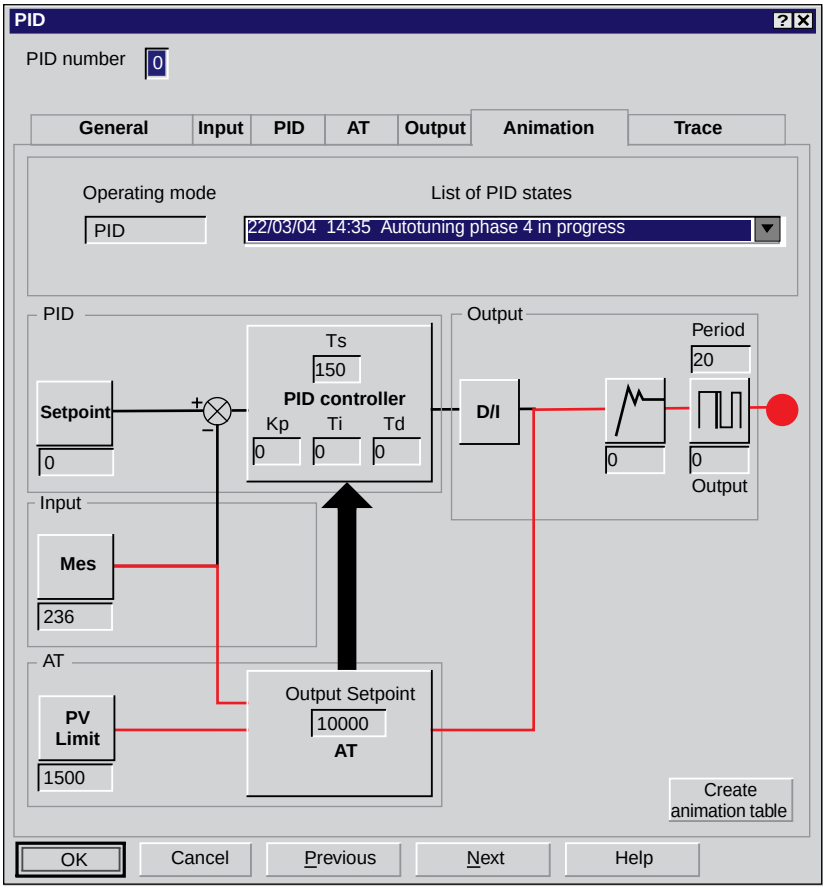
概览

该表用于调试 PID。
该图表取决于您已经创建的 PID 控制类型。只有已配置元件可被显示。
该显示是动态的。活动连接显示为红色，非活动连接显示为黑色。

注意：它在在线模式下被访问。

PID 动态监控表

下面屏幕用于 PID 显示和调试。



描述

下表描述了窗口的各个区域。

区域	描述
PID 编号	在这里指定您想调试的 PID 编号。 值介于 0 到 13 之间，每个应用最多 14 个 PID。
操作模式	这个区域显示当前 PID 工作模式。
PID 状态列表	该下拉列表允许实时查看最后 15 个 PID 状态。该列表在每个状态变化时被更新，并指示变化的日期和时间以及当前状态。
创建动态监控表	点击创建动态监控表，创建一个文件，包含图表显示的所有变量，使得可以在线修改它们和调试 PID。

PID 功能跟踪表

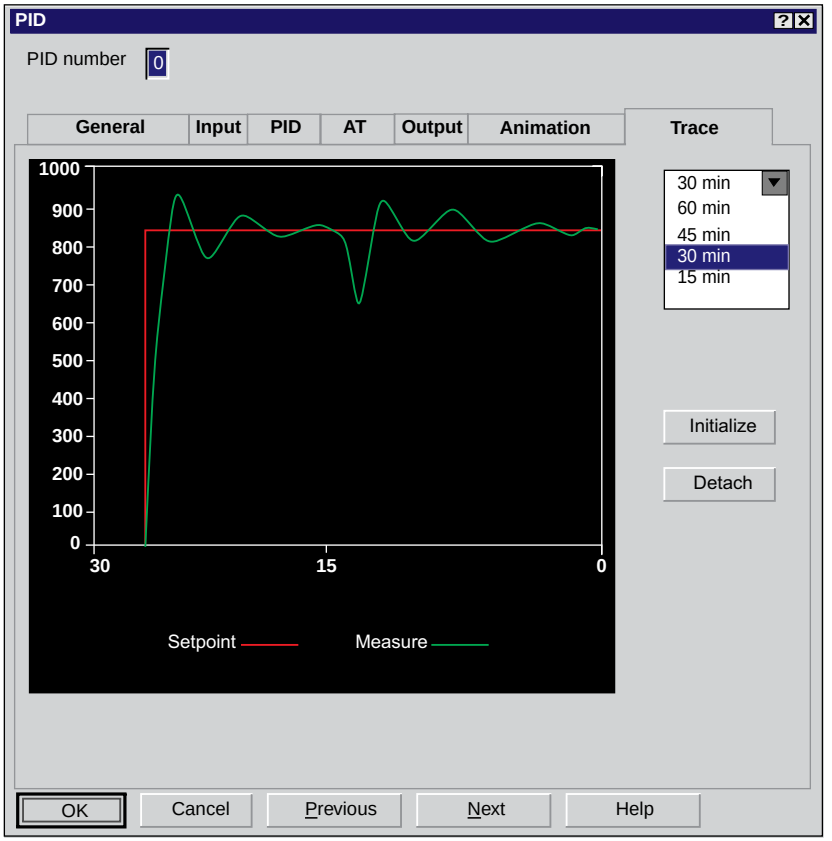
概览

该表允许您查看 PID 运行，并对其行为方式进行调节。
一旦显示调试窗口，则该趋势图就开始工作。

注意：它在在线模式下被访问。

PID 运行趋势图跟踪表

下面屏幕用于查看 PID 控制。



描述

下表描述了窗口的各个区域。

区域	描述
PID 编号	在这里指定您想查看的 PID 编号。 值介于 0 到 13 之间，每个应用最多 14 个 PID。
图表	这个区域显示了设定值和过程值图表。 横轴坐标标度 （X）通过窗口右上角菜单决定。 纵轴坐标标度通过 PID 输入配置值 （带或不带转换）决定。它自动被优化以获得图形的最佳显示。
横轴刻度菜单	这个菜单允许您修改横轴的标度。您可以从 4 个值中选择：15，30，45 或 60 分。
初始化	这个按钮清除图表并重新开始趋势图。

PID 功能状态和错误代码

概览

作为 **PID** 状态列表获得经由动态监控对话框 （见 *PID 功能动态监控表，p.592*），它允许查看并在最近 15 个 PID 状态间切换，Twido PID 控制器还能够在用户自定义的内存字中记录 PID 控制器及自整定过程的当前状态。
如何激活并配置 **PID** 状态内存字 （%MWi）请参阅 *PID 功能总表，p.574*。

PID 状态内存字

PID 状态内存字可以记录如下三种 PID 信息中的任意一个：
- PID 控制器的当前状态 （PID 状态）
- 自整定过程的当前状态 （AT 状态）
- PID 和自整定错误代码

注意：PID 状态内存字只可读。

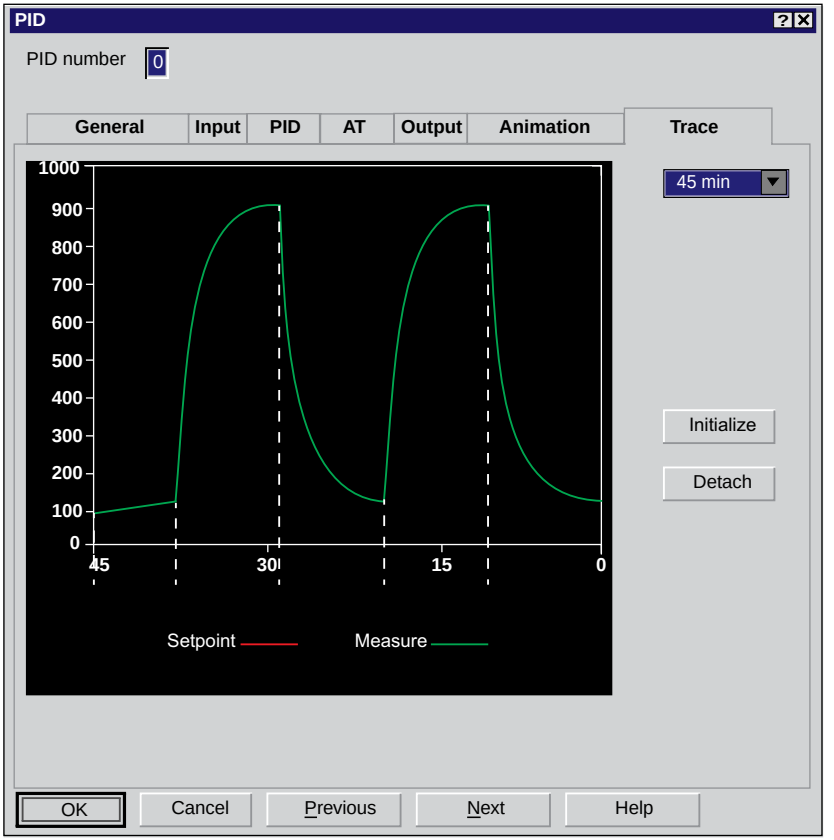
PID 状态内存字

下表是 PID 自整定状态与内存字中的十六进制数对照表：

PID 状态表示为十六进制数	**描述**
0000h	PID 控制没有激活
2000h	PID 控制进行中
4000h	已达到 PID 设定值

AT 状态说明

自整定过程分成四个连续的阶段，每个阶段必须确保完成，以此确保自整定的成功结束。下面的对应曲线及表格描述了 PID 自整定的四个阶段：



下表描述了 PID 自整定的各阶段：

AT 阶段	描述
1	阶段 **1** 是稳定阶段。它从用户启动自整定过程开始。在此阶段， Twido 自整定要检测并确认过程变量处于恒稳态。 **注意：**在自整定之前最后一次作用于过程的输出量，将被用作自整定的起始点及释放点。
2	阶段 **2** 把第一步改变作用于过程，并由它产生如上图所示的过程的第一步响应。

AT 阶段	描述
3	阶段 **3** 是释放阶段，它在第一步响应稳定后开始执行。 **注意：**释放趋于的稳定值，取决于自整定开如前作用于过程的输出量。
4	阶段 **4** 按上面第二阶段描述相同的数量和方式把第二步改变作用于过程。自整定过程此时完成并在第 4 阶段结束前将计算出的自整定参数分别存储于内存字中。 **注意：**此阶段完成后，过程变量就恢复到自整定开始前作用于过程的输出量的水平。

AT 状态内存字

下表是 PID 自整定状态与内存字中的十六进制数对照表：

AT 状态表示为十六进制数	描述
0100h	自整定处于第 1 阶段
0200h	自整定处于第 2 阶段
0400h	自整定处于第 3 阶段
0800h	自整定处于第 4 阶段
1000h	自整定过程完成

PID 和自整定错误代码

下表是在 PID 控制及自整定过程中可能遇到的错误：

PID/AT 过程	错误代码（十六进制）	描述
PID 错误	8001h	工作方式数值超出范围
	8002h	线性转换最小值等于最大值
	8003h	数字输出的上限小于下限值
	8004h	过程变量门限值超出线性转换范围
	8005h	过程变量门限小于 0 或大于 10000
	8006h	输出值超出线性转换范围
	8007h	设定值小于 0 或大于 10000
	8008h	控制动作与自整定开始时决定的动作方式不同
自整定错误	8009h	自整定错误：（PV）已达到门限值
	800Ah	自整定错误：由于过度采样或输出设定值太低
	800Bh	自整定错误：Kp 为 0
	800Ch	自整定错误：时间常数是一个负数
	800Dh	自整定错误：延时为负数
	800Eh	自整定错误：Kp 计算错误
	800Fh	自整定错误：时间常数超过延时时间比例 >20
	8010h	自整定错误：时间常数超过延时时间比例 <2
	8011h	自整定错误：Kp 超过门限值
	8012h	自整定错误：Ti 超过门限值
	8013h	自整定错误：Td 超过门限值

PID 自整定功能 （AT）

PID 整定总述

PID 控制依赖于用户自定义的以下三个参数：Kp， Ti 和 Td。 PID 整定的目的是为了准确的确定这些过程参数，以便为过程提供优化的控制。

自整定的作用

Twido PLC 的 PID 自整定功能特别适用于热控制过程，由于一个控制过程的 PID 的参数值可能与另一个的差别很大，所以自整定功能 Twido PLC 可以为准确的确定参数值提供帮助，而且会比猜测值更准确，更有效。

自整定的要求

在使用自整定功能前，请先确认 Twido PLC 满足以下四个要求：

- 控制过程必须是一个开环，稳定系统。
- 在自整定开始时，控制过程必须处于稳态，输入为 0 （例：烤箱或熔炉必须与环境温度相同。）
- 在自整定过程中，确保不要有扰动输入到过程中，否则要么计算的参数值不正确，要么自整定失败 （例：烤箱的门不要打开，即使一小会儿。）
- 把 Twido PLC 配置为周期扫描模式。一旦为自整定确定了正确的采样周期 （Ts），扫描周期必须配置，并且该采样周期 （Ts）必须是 Twido PLC 扫描周期的整数倍。

注意：为了确保 PID 和自整定功能正确运行，一定要将 Twido PLC 配置为周期扫描模式 （不是循环扫描模式）。（不象循环扫描模式时，每个扫描都在前一个结束后立即开始，这使得采样周期在每个扫描周期时不均衡。）

自整定工作模式

自整定既可以独立使用 （AT 模式）也可以与 PID 联合使用 （AT + PID）：

- **AT 模式：**在自整定过程成功完成，即确定了 Kp， Ti 和 Td （或者检测到一个自整定算法错误），自整定的数字输出被设为 0，并将如下信息显示在 **PID** 状态列表中 “自整定已完成”。
- **AT + PID 模式：**首先 AT 启动，成功完成自整定后，PID 调节回路启动（Kp，TI 和 Td 这些参数使用自整定值）。

  AT+ PID 注解：如果自整定算法发生错误：
  - 没有计算出 PID 参数；
  - 自整定的数字输出设定为自整定开始前作用于过程的输出值；
  - 在 PID 状态下拉列表中出现错误信息
  - PID 控制已取消。

> **注意：**无冲击转移
> 当处于 **AT+PID** 模式，从 AT 到 PID 的传送是无冲击的。

确定采样周期（Ts）的方法

正如后面两节将要介绍的 （见*附录 1：PID 基本原理， P619 和附录 2：一阶延时模型，p.621*），采样周期 （**Ts**）在 PID 控制中是一个重要参数。
采样周期可以从自整定 AT 时间常数 （τ）中推导得出。
通过自整定功能，有两种方法可以用来估测正确的采样周期 （Ts），它们在后续两节中有说明。

- 过程响应曲线法
- 错误跟踪法

这两种方法在下面两个子节中进行说明。

过程响应曲线法介绍

这种方法是在过程控制的输入设定一个阶变值，记录过程输出及时间。
过程响应曲线法产生如下假设：

- 作为一个一阶延时模型，过程控制可以通过下面的传送函数充分说明：

$$\frac{S}{U} = \frac{k}{1+\tau p} \cdot e^{-\theta p}$$

（更多信息，参阅附录 2：一阶延时模型）

使用过程响应曲线法

使用过程响应曲线法去确定采样周期 （Ts），要遵循以下步骤：

步骤	动作
1	假设已经将 PID 配置中的概要，输入， PID， AT 和输出项进行配置。
2	从应用浏览器选择 **PID** > 输出页。
3	从手动模式 （**Manual mode**）下拉列表选择授权或地址位，以激活手动输出并将输出域设置高电平 （介于 [5000-10000] 之间）。
4	选择 **PLC** > 传送 **PC** => **PLC**... 下载程序到 Twido PLC.
5	在 PID 配置窗口，切换到跟踪模式。
6	运行 PID，并查看响应曲线上升。
7	当曲线达到一个稳定的状态时，停止 PID 测量。 **注意：**PID 轨迹窗口保持活动状态。
8	利用下述图解法确定控制过程的时间常数 （τ）： **1.** 计算过程变量输出值的 63% （$S_{[63\%]}$），并利用下面的公式： $S_{[63\%]} = S_{[初始]} + （S_{[结束]} - S_{[初始]}） \times 63\%$ **2.** 从图上查找时间横坐标 （$t_{[63\%]}$），对应于 S （63%）。 **3.** 从图上查找到初始化时间 （$t_{[初始]}$），对应于过程响应上升的起始点。 **4.** 计算过程控制的时间常数 （τ）通过下面的关系式：$\tau = t_{[63\%]} - t_{[初始]}$
9	基于上一步所得的值 （τ）计算采样周期 （Ts），公式是：Ts = τ/75 **注意：**采样周期的时间单位是 10ms。所认，应将计算所得的值 （Ts）四舍五入为 10ms 的最小倍数。
10	从菜单栏中，选择程序 > 扫描模式编辑，并进行如下操作： **1.** 设定自整定的扫描模式为周期。 **2.** 设定自整定的扫描周期 （Ts）应该是扫描周期的整数倍，遵循以下原则： 扫描周期 = Ts / n, 这里 “n” 是一个正整数。 **注意：**“n” 值，并使扫描周期是一个正整数，并介于 [2 — 150 ms] 之间。

过程响应曲线应用举例

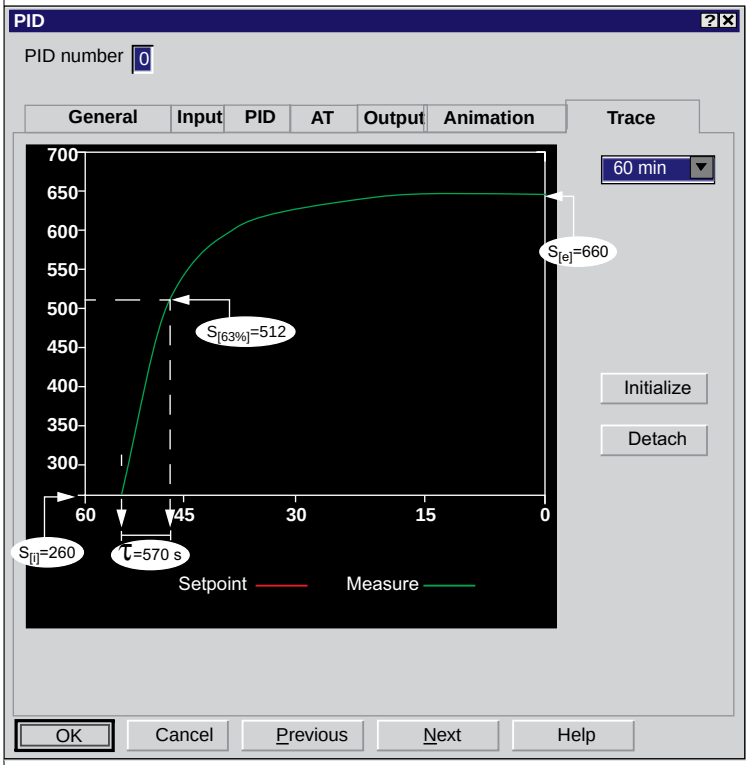
本例说明了如何利用上节所说的过程响应曲线法计算简单热过程中时间常数（τ）。

时间常数测量实验性设定如下所示：

- 本控制过程是以热灯泡为热源的烘箱。
- 温度采集由 Twido PLC 通过一个 Pt100 传感器获取，并且温度值记录为℃
- TwidoPLC 通过 PID 的 PWM 开关量输出来驱动加热灯泡。

实验过程如下：

步骤	动作
1	PID 输出表是从 PID 配置屏幕中选择。
2	手动模式是从输出表中选择。
3	手动模式的输出设定为 10000。
4	PID 已经从 PID 的跟踪表启动。
5	PID 运行在烘箱温度达到稳定状态后停止。

步骤	动作
6	以下信息从响应曲线中直接分析得到，并如下表所示：  这里 ● $S_{[i]}$ = 过程变量初始值 = 260 ● $S_{[e]}$ = 过程变量结束值 = 660 ● $S_{[63\%]}$ = 过程变量的 63% = $S_{[i]}$ + （$S_{[i]}$ - $S_{[e]}$）× 63% = 260+ （660-260）× 63% = 512 ● τ = 时间常数 = 过程变量从开始到达到 $S_{[63\%]}$ = 9 min 30 s = 570 s
7	采样周期 （Ts）利用下述关系得出： Ts = τ /75 = 570/75 = 7.6 s （7600 ms）

步骤	动作
8	在程序 > 扫描模式编辑对话框，扫描周期必须设定以便采样周期 （Ts）是扫描周期的一个精确整数倍，如下例所示：扫描周期 = Ts/76 = 7600/76 = 100 ms（它能满足条件：2 ms ≤扫描周期≤ 150 ms。）

错误跟踪法

错误跟踪法用于通过猜测为自整定提供采样周期，直到自整定算法成功得出用户认为正确的 Kp， Ti 和 Td 值。

注意：不同于过程响应曲线法，错误跟踪法不是基于任何过程响应的近似原理，而且，它的优势在于，能够向着采样周期的真实值方向逼近。

通过错误跟踪法估算自整定参数，按以下步骤：

步骤	动作
1	从 PID 配置窗口中选择自整定表。
2	设定自整定的输出门限值为 10000。
3	从 PID 配置窗口中选择 PID 表。
4	提供第一个或第 n^{th} 采样周期的猜测值。 **注意：**如果无法知道第一个采样周期可能的范围，将它设定为可能的最小值：1（10 ms 为一个单位）。
5	选择 PLC > 传送 PC => PLC…下载程序到 Twido PLC.
6	启动自整定。
7	从 PID 配置屏幕中选择动态监控表。
8	等待，直到自整定过程结束。
9	可能出现以下两种情况： ● 自整定已成功完成：可以继续第 9 步。 ● 自整定失败：这意味着刚才猜测的采样周期是不正确的。重新猜测一个新值，并重复以上 3 到 8 步，直到自整定过程最终完成。 按照以下原则提供新的猜测值： ● 自整定最后给出的错误信息是：“计算的时间常数是负数！”：这意味着采样周期 Ts 太大，应该降低采样周期值来提供新的猜测值。 ● 自整定最后给出的错误信息是：“采样错误！”：这意味着采样周期 Ts 太小。可以通过增加采样周期来提供新的猜测值。
10	这时可以在动态监控表中查看 PID 控制参数 （Kp， Ti 和 Td），如果需要，并可以在 PID 配置窗口中 PID 表内调整它们。 **注意：**如果通过它取得的控制参数不能让所有的 PID 调整都满意，那么可以重复上面的错误跟踪法，直到得到正确的控制参数 Kp， Ti 和 Td。

PID 参数调整

要改进通过自整定获得的 PID 参数 （Kp， Ti， Td）的过程调节，仍然可以通过手动调整这些参数，直接从 PID 配置窗口中的 PID 表修改，或者通过与它们对应的内存字 （%MW）。

使用自整定及 PID 控制的限制

此自整定特别适用于这样的过程控制，它们的时间常数 （τ）和延时时间 （θ）满足以下要求：（τ+θ）< 2700 s（例如：45 min）此 PID 控制特别适用于满足以下两个条件的过程调节：2 < （τ/θ） < 20，这里 （τ）是过程时间常数，（θ）是延时时间。

注意：按照这个比例 （τ / θ）：
- （t/θ） < 2：PID 调节已经达到其极限；在这种情况下需要更高级的调节技术。
- （t/θ） > 20：这样， on/off （或两步）可以取代 PID 控制器。

自整定功能错误及解决

下表给出了自整定错误信息，并说明了可能产生的原因及解决办法：

错误信息	可能原因	解释 / 可能的解决办法
自整定错误：（PV）已达到门限值	过程变量已经到达允许的最大值。	这是一个系统保险。 因为自整定是开环过程，过程变量 （PV）门限值是一个上限。
自整定错误：或是由于过度取样，或输出设置点太低。	两种可能原因中的一个： ● 采样周期太小。 ● 自整定输出太低。	增大采样周期或增加自整定输出设定值。
自整定错误：时间常数是一个负数	采样周期可能太大。	更多信息，请参阅 *PID 自整定功能 （AT）*，*p.602*。
自整定错误：Kp 计算错误	自整定失败 （未完成）。	检查 PID 和自整定参数并予以调整，这有利于提高成功完成整定。 同时检查，是否过程变量受到扰动因素影响。
自整定错误：时间常数与延时时间比例 > 20	$\tau / \theta > 20$	PID 调节已经没有保证。 更多信息，请参阅 *PID 自整定功能 （AT）*，*p.602*。
自整定错误：时间常数与延时时间比例 < 2	$\tau / \theta < 2$	PID 调节已经没有保证。 更多信息，请参阅 *PID 自整定功能 （AT） p.602*。
自整定错误：Kp 超过门限值	计算出的静态增益 （Kp）大于 10000。	某些应用变量的测量敏感度可能太低，应用测量范围必须调整为 [0-10000]。
自整定错误：Ti 超过门限值	计算所得的积分时间常数 （Ti）大于 20000。	达到计算限定值。
自整定错误：Td 超过门限值	计算所得的微分时间常 （Td）大于 10000。	达到计算限定值。

PID 参数调整方法

介绍

存在大量的调整 PID 参数的方法，推荐使用 Ziegler 和 Nichols 方法，各含有两个参数：

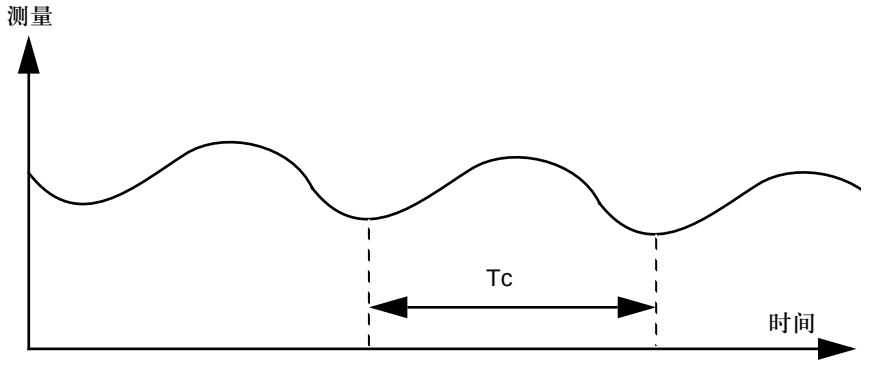
- 闭环调整，
- 开环调整。

在使用任何一种方法之前，必须首先设定 PID 动作方向：

- 如果输出 （OUT）的增加使得 PV 度量值增加，令 PID 反向动作 （KP> 0），
- 反之，如果它导致 PV 值减小，令 PID 正向动作 （KP<0）。

闭环调整

它主要是只用比例命令 （Ti = 0， Td = 0）来启动一个过程，通过增加输出并造成振荡，而使 PID 的设定值校正参数达到一定水平。所有这些都要提高临界输出水平（Kpc），这会造成无阻尼振荡，并且通过振荡周期 （Tc）减小来得到最优的调整。



根据 （PID 或 PI）调节的种类，调整系数的值如下：

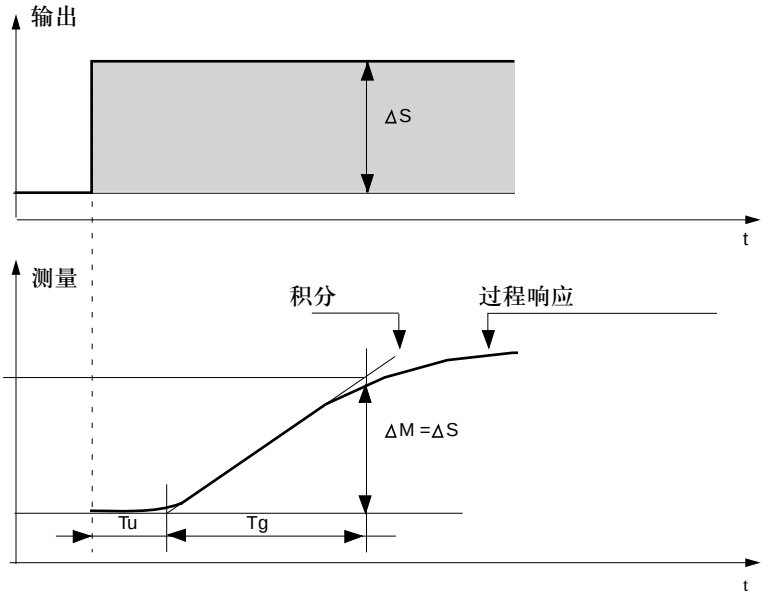
-	**Kp**	**Ti**	**Td**
PID	Kpc/1.7	Tc/2	Tc/8
PI	Kpc/2.22	0.83 x Tc	-

这里 Kp = 比例增益， Ti = 积分时间， TD = 微分时间。

注意：这种方法给出了一种非常动态的命令，可以通过设定值意外变化时的过冲表达自己，在这种情况下，降低输出值直至获得所需动作。

开环调整

由于调节器处于手动模式，使得输出处于一定等级，并令过程响应初始状态与带有纯延迟时间的调节器的状态相同。



右手边的交叉点代表对时间轴的积分，它决定时间 Tu。下一步， Tg 时间由控制器变量 （测量值）决定，该变量与调节器输出有同样的大小变化 （% 百分比）。
根据 （PID 或 PI）调节的种类，调整系数的值如下：

-	Kp	Ti	Td
PID	-1.2 Tg/Tu	2 x Tu	0.5 x Tu
PI	-0.9 Tg/Tu	3.3 x Tu	-

这里 Kp = 比例增益， Ti = 积分时间， TD = 微分时间。

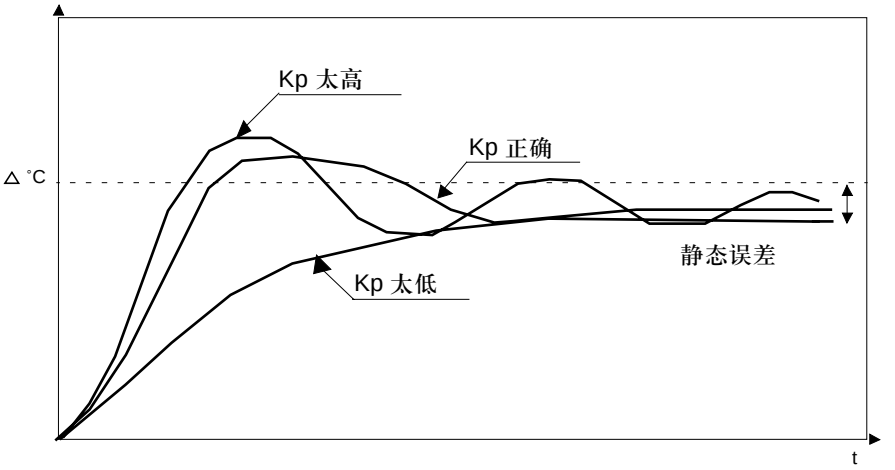
注意：注意单位。如果通过 TwidoSoft 进行调整，对于 KP 则将获得值乘以 100。

这种方法给出了一种非常动态的命令，可以通过设定值意外变化时的过冲表达自己。在这种情况下，降低输出值直至获得所需动作。这种方法是有趣的，因为它不需要任何有关环境和过程顺序的假定。可以像运行真正集成过程一样运行这个稳定的过程。如果是缓慢过程 （玻璃工业，…）则更加有趣，因为用户只需要在响应的开始来校准参数 Kp， Ti 和 Td。

PID 参数的任务和影响

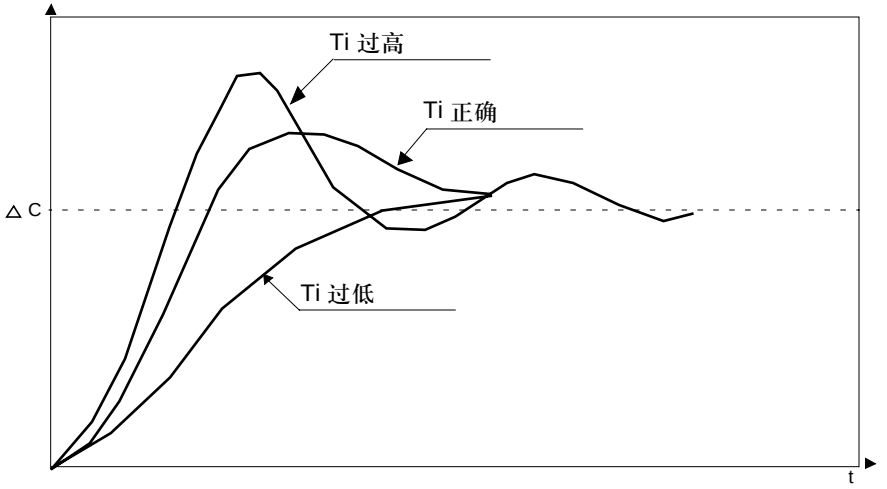
比例作用的影响

比例增益的作用是使过程响应的速度加快。增益越大，响应越快，静态误差 （正比例）越高，因此稳定性越差。必须找到速度和稳定之间的折衷。积分对于过程响应的影响如下：



积分作用的影响

积分的作用是消除静态误差，（过程值和设定值间的差）。积分作用标准越高 （Ti 低），响应越快，稳定性越差。必须找到合适的速度和稳定之间的折衷。积分作用对于过程响应的影响如下：

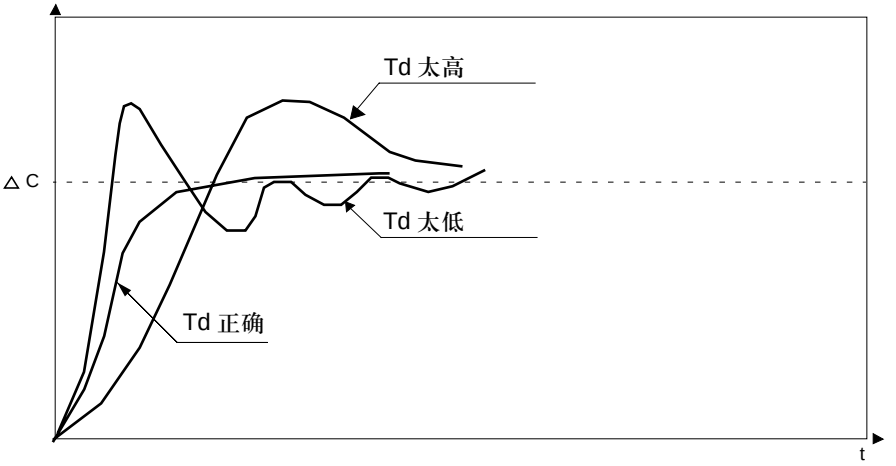


注意：T 低意味着积分作用标准高。

这里 Kp = 比例增益， Ti = 积分时间， Td = 微分时间。

微分作用的影响

微分作用是对趋势的预测。在实际中，它增加偏差中一个度量偏差速度的项，由此通过偏差增加来加速响应时间，偏差减少减慢响应时间来预测改变。微分作用的级别越高 （Td 高），响应越快。必须找到速度和稳定之间的折衷。



PID 控制回路的限制

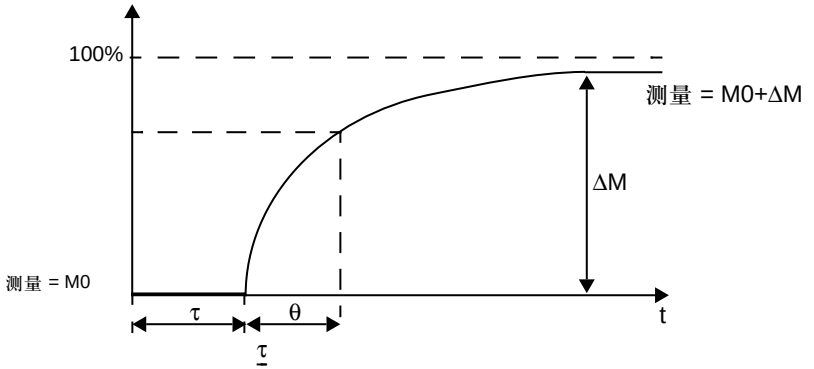
如果此过程由一个纯延迟一阶转移函数吸收：

$$(H(p)) = K\frac{(e^{(-\tau)p})}{(1+\theta p)}$$

这里：

τ = 模型延迟，

θ = 模型时间常数，



过程控制性能决定于比例 $\frac{\tau}{\theta}$

合适的 PID 过程控制，符合如下域：$2-\frac{\tau}{\theta}-20$

如果 $\frac{\tau}{\theta}$ 换句话即快速控制回路 （低 θ），或者过程有较大延时 （高 t），该 PID 过程控制就不合适了。在这种情况下，需要更复杂的运算法则。

如果 $\frac{\tau}{\theta}$ >20, 一个过程控制使用一个阈值加滞后就足够了。

附录 1：PID 基本原理

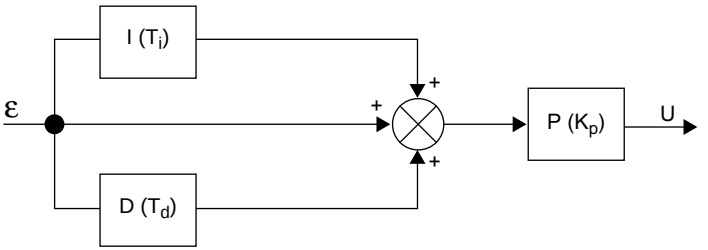
介绍

Twido 控制器上内置的 PID 控制功能，为简单过程控制提供了一个有效的控制方法，包含一个系统促进因素 （参看本章的设定值）和一个测量值 （参看测量值或过程变量）。

PID 控制器模型

Twido PID 控制器采用混合 （串 - 并） PID 校正 （见下面的 PID 模型图），通过模拟量测量，设定值为 [0-10000]，提供给模拟输出的的执行命令也是与之相同的格式。

PID 控制器模型混合形式如下图所示：



这里：

- I = 积分动作 （独立动作并与微分并行），
- D = 微分动作 （独立动作并与积分并行），
- P = 比例增益动作 （与积分及微分的合并输出串行动作），
- U = PID 控制器的输出 （最近的反馈，作为控制过程的输入。）

PID 控制规则

PID 控制器由控制比例增益 （Kp），和积分 （Ti）以及微分 （Td）时间的多项混合式 （串行 - 并行）组成。所以 Twido 控制器的 PID 规则遵循如下公式 *（Eq.1）*：

$$u(i) = K_P \cdot \left\{ \varepsilon(i) + \frac{T_s}{T_i} \sum_{j=1}^{i} \varepsilon(j) + \frac{T_d}{T_s} [\varepsilon(i) - \varepsilon(i-1)] \right\}$$

这里

- Kp = 控制比例增益，
- Ti = 积分时间常数，
- Td = 微分时间常数，
- Ts = 采样时间，
- ε （i） = 误差 （ε （i） = 设定值 - 过程变量。）

注意：根据积分时间常数 （Ti）有两种不同的算法：

- Ti ≠ 0: 在此情况下，应用加法。
- Ti = 0: 这是没有积分过程的情况，此时，应用定位算法，连同应用到 PID 输出变量的 +5000 偏移一起。

Kp， Ti 和 Td 的详细说明，请参照 *PID 功能 PID 表*。

正如可以从 （equ.1），PID 调整的主要参数是采样周期 （Ts）。

采样周期取决于与之相关的时间常数 （τ），一个过程内在的， PID 作为控制目标的参数。（见*附录 2：时间延时一阶模型 .p.621*。）

附录 2：时间延时一阶模型

介绍

本节介绍了时间延时一阶模型，用于说明简单但却重要的工业过程变化，包括热过程。

时间延时一阶模型

普遍假定简单 （一元）的热过程可以用一附延时模型来近似模拟。
这种一阶、开环过程的转移函数具有如下的拉斯域 （Laplace domain）*（equ.2）*：

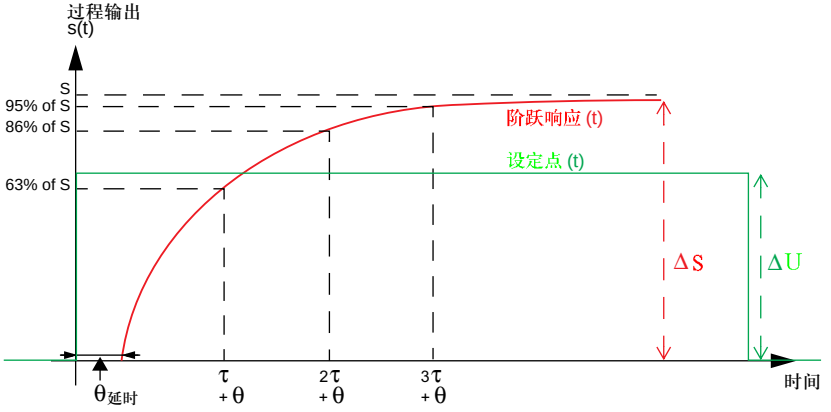
$$\frac{S}{U} = \frac{k}{1+\tau p} \cdot e^{-\theta p}$$

这里

- k = 静态增益，
- τ = 时间常数，
- θ = 延时，
- U = 过程输入 （这是 PID 控制器的输出），
- S = 过程输出。

过程时间常数

过程响应法则的主要参数 （equ.2）是时间常数 τ。它是过程控制内在的参数。
时间常数 （τ）在一阶系统中定义为一个时间 （单位 sec），它是系统输出变量从开始到等于最终输出的 63% 时所用的时间 u （t）。
下图描述了典型的一阶过程对一阶元素的响应：



这里

- k = 静态增益，由比率 ΔS/ΔU 得出，
- τ = 达到 63% 上升值时的时间 = 时间常数，
- 2τ = 达到 86% 上升值时时间，
- 3τ = 达到 95% 上升值时时间。

注意：当应用自整定功能时，采样时间 （Ts）必须在如下范围内选择：[τ/125 <Ts < τ/25]。理想化的，应该用 [Ts= τ/75]。（见 *PID 自整定功能 （AT）p.602*。）

17.5 浮点指令

概览

本节目的

本节描述了 TwidoSoft 语言中的浮点数指令。（*参见浮点和双字对象，p.35*）
比较和赋值指令描述见*数字运算，p.446*

本节包含了哪些内容?

本节包含了以下主题：

主题	页码
浮点算术指令	624
三角指令	629
转换指令	632
整数 <-> 浮点数转换指令	634

浮点算术指令

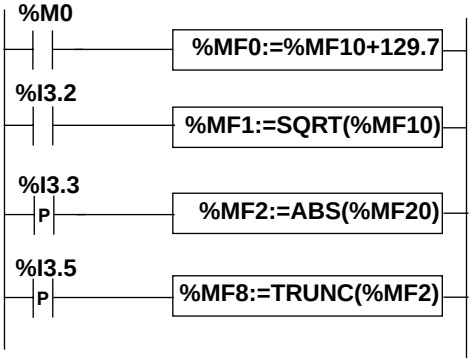
总述

这些指令用于执行两个操作数或一个操作数的算术运算。

+	两个操作数相加	**SQRT**	一个操作数的平方根
-	两个操作数相减	**ABS**	一个操作数的绝对值
*****	两个操作数相乘	**TRUNC**	浮点值的整数部分
/	两个操作数相除	**EXP**	自然指数
LOG	以 10 为底的对数	**EXPT**	整数的实数次幂
LN	自然对数		

结构

梯形图语言



指令列表语言

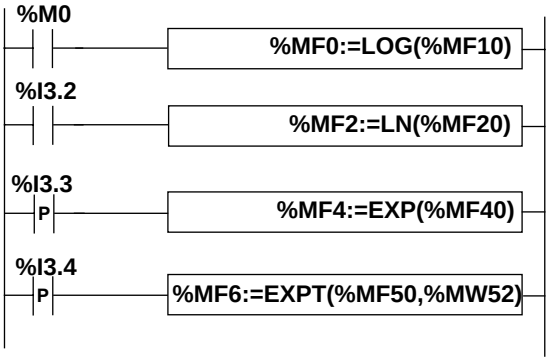
LD %M0
[%MF0:=%MF10+129.7]

LD %I3.2
[%MF1:=SQRT (%MF10)]

LDR %I3.3
[%MF2:=ABS (%MF20)]

LDR %I3.5
[%MF8:=TRUNC (%MF2)]

梯形图语言



指令列表语言

LD %M0
[%MF0: =LOG(%MF10]

LD %I3.2
[%MF2: =LN(%MF20)]

LDR %I3.3
[%MF4: =EXP(%MF40)]

LDR %I3.4
[%MF6: =EXPT(%MF50,%MW52)]

语法

浮点算术指令运算符和语法

运算符	语法
+, - *, /	Op1：= Op2 运算符 Op3
SQRT, ABS, TRUNC, LOG, EXP, LN	Op1：= 运算符 （Op2）
EXPT	Op1：= 运算符 （Op2， Op3）

注意：当您执行两个浮点数的加法或减法运算时，这两个操作数必须满足条件：Op1> Op2 × 2^{-24}，这里 Op1>Op2. 如果这个条件不能被满足，结果将等于操作数 1 （Op1）。尽管在单独运算的时候运算错误非常低 （2^{-24}），但因为其不断的重得计算，将产生不可预料的后果。
例如这种情况指令。 %MF2：= %MF2 + %MF0 被不确定重复。如果初始条件是 %MF0 = 1.0 和 %MF2= 0，值 %MF2 将被锁在 16777216。
因此我们提醒您在对重复计算编程时要特别小心。如果您相对这类计算进行编程，则客户应用程序要对截断误差进行管理。

浮点算术指令的操作数：

运算符	操作数 1 （**Op1**）	操作数 2 （**Op2**）	操作数 3 （**Op3**）
+, - *, /	%MFi	%MFi， %KFi，立即值	%MFi， %KFi，立即值
SQRT, ABS, LOG, EXP, LN	%MFi	%MFi, %KFi	[-]
TRUNC	%MFi	%MFi, %KFi	[-]
EXPT	%MFi	%MFi, %KFi	%MWi， %KWi，立即值

使用规则

- 浮点和整数值不能直接混合运算。转换操作 （见*整数 <-> 浮点转换指令，p.634*）将把它们进行相互转换。）
- 系统位 %S18 的管理方法相同于整数运算 （见*整数算术指令， p.455*），字 %SW17 （参见系统字 （*%SW*）， p.667）指示错误原因。
- 当函数的操作数是无效数 （例如：负数的对数），它将产生一个不确定或无限大的结果，并将位 %S18 置 1，系统字 %SW17 指示错误原因。

三角指令

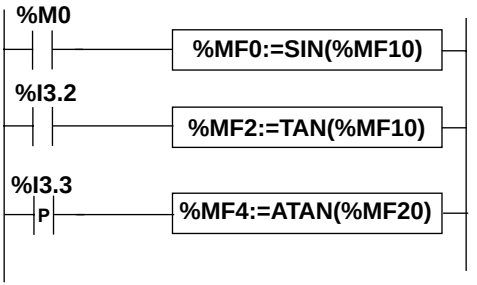
总述

这些指令使得用户可以执行三角函数运算。

SIN	一个角度 （弧度）的正弦，	**ASIN**	反正弦 （结果介于 $-\frac{\pi}{2}$ 和 $\frac{\pi}{2}$ ）
COS	一个角度 （弧度）的余弦，	**ACOS**	反余弦 （结果介于 0 和π）
TAN	一个角度 （弧度）的正切，	**ATAN**	反正切 （结果介于 $-\frac{\pi}{2}$ 和 $\frac{\pi}{2}$ ）

结构

梯形图语言



指令列表语言

LD %M0
[%MF0:=SIN(%MF10)]

LD %I3.2
[%MF2:=TAN(%MF10)]

LDR %I3.3
[%MF4:=ATAN(%MF20)]

语法

三角运算指令的运算符，操作数和语法

运算符	语法	操作数 1 (Op1)	操作数 2(Op2)
SIN, COS, TAN, ASIN, ACOS, ATAN	Op1：= 运算符（Op2）	%MFi	%MFi, %KFi

使用规则

- 当函数的操作数是无效数（例如：反余弦函数的操作数大于 1），它将产生一个不确定的或无穷大的结果，并且将位 %S18 置 1，字 %SW17（见*系统字（%SW）*，*p.667*）指示错误原因。
- SIN/COS/TAN 函数允许角度介于–4096π和4096π作为参数，但其精度随着角度超出 –2π 和+2π越大而越低，因为这些参数在以2π 为模型而带来的不精确在进行运算前就已产生。

转换指令

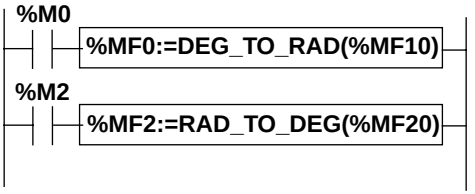
总述

这些指令用于执行转换操作。

DEG_TO_RAD	将角度转换为弧度，结果值介于 0 和 2π
RAD_TO_DEG	将弧度转换为角度，结果值介于 0 和 360 度之间

结构

梯形图语言



指令列表语言

LD %M0
[%MF0:=DEG_TO_RAD(%MF10)]

LD %M2
[%MF2:=RAD_TO_DEG(%MF20)]

语法

转换指令的运算符，操作数和语法

运算符	语法	操作数 1 **(Op1)**	操作数 2 **(Op2)**
DEG_TO_RAD **RAD_TO_DEG**	Op1：= 运算符 （Op2）	%MFi	%MFi, %KFi

使用规则

角度转换值必须介于 -737280.0 和 +737280.0 之间 （对于 DEG_TO_RAD 转换），或介于 –4096π 和 4096π 之间 （对于 RAD_TO_DEG 转换）。

对于这些范围之外的值，结果将显示为 +1.#NAN，位 %S18 和 %SW17：X0 将被置为 1。

整数 <-> 浮点转换指令

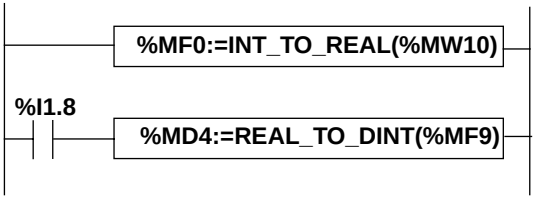
概要

提供了四个转换指令。
整数 <-> 浮点转换指令列表：

INT_TO_REAL	转换一个整数字 --> 浮点
DINT_TO_REAL	转换一个整数双字 --> 浮点
REAL_TO_INT	浮点转换 --> 整数字 （结果为最接近的代数值）
REAL_TO_DINT	浮点转换 --> 整数双字 （结果为最接近的代数值）

结构

梯形图语言



指令列表语言
LD 1
[%MF0:=INT_TO_REAL(%MW10)]

LD I1.8
[%MD4:=REAL_TO_DINT(%MF9)]

语法

运算符和语法 （转换一个整数字 --> 浮点）：

运算符	语法
INT_TO_REAL	Op1=INT_TO_REAL(Op2)

操作数 （转换一个整数字 --> 浮点）：

操作数 1 （Op1）	操作数 2 （Op2）
%MFi	%MWi,%KWi

示例：整数字 --> 浮点：147 --> 1.47e+02

运算符和语法 （转换双整数字 --> 浮点）：

运算符	语法
DINT_TO_REAL	Op1=DINT_TO_REAL(Op2)

操作数 （转换双字整数 --> 浮点）：

操作数 1 （Op1）	操作数 2 （Op2）
%MFi	%MDi,%KDi

示例：双字整数转换 --> 浮点：68905000 --> 6.8905e+07

运算符和语法 （转换浮点 --> 整数字或双字整数）：

运算符	语法
REAL_TO_INT	Op1= 操作数 （Op2）
REAL_TO_DINT	

操作数 （转换浮点 --> 整数字或双字整数）：

类型	操作数 1 （Op1）	操作数 2 （Op2）
字	%MWi	%MFi, %KFi
双字	%MDi	%MFi, %KFi

示例：

浮点转换 --> 整数字：5978.6 --> 5979

浮点转换 --> 双字整数：-1235978.6 --> -1235979

注意：如果实数转换到整数 （或实数转换到双字整数）时浮点值超出字 （或双字）的限制，位 %S18 将被置为 1。

取整精度

IEEE 754 标准为浮点运算定义了 4 种取整模式。
上面指令采用的是”最接近值取整”模式：
“如果可取的最接近值与理论值距离相等，则取其低有效位为 0 的那个值。”
某些情况下，取整的结果可以是省略值或超额值。
例如：
取整 10.5 -> 10
取整 11.5 -> 12

17.6 对象表指令

概览

本节目标

本节描述了有关表功能的特殊指令：

- 双字，
- 浮点对象。

表的赋值指令的描述位于章节”基本指令”（见字，*双字和浮点表赋值，p.451*）

本节包含了哪些内容?

本节包含了以下主题：

表求和功能

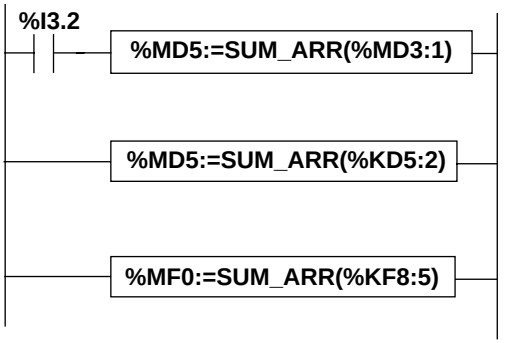
概要

SUM_ARR 功能将一个对象表的所有元素加在一起：

- 如果表由双字组成，则结果以双字格式给出
- 如果表由浮点字组成，则结果以浮点字格式给出

结构

梯形图语言



指令列表语言

LD %I3.2
[%MD5:=SUM_ARR(%MD3:1)]
LD1
[%MD5:=SUM_ARR(%KD5:2)]
LD1
[%MF0:=SUM_ARR(%KF8:5)]

语法

表求和指令语法：

```
Res:=SUM_ARR(Tab)
```

表求和指令参数

类型	结果 （res）	表 （Tab）
双字表	%MDi	%MDi:L,%KDi:L
浮点字表	%MFi	%MFi:L,%KFi:L

注意：当对表操作后的结果不在有效双字格式范围内时，系统位 %S18 将被置为 1。

示例

%MD5:=SUM(%MD30:4)

在这里 %MD30=10, %MD31=20, %MD32=30, %MD33=40

%MD5=10+20+30+40=100

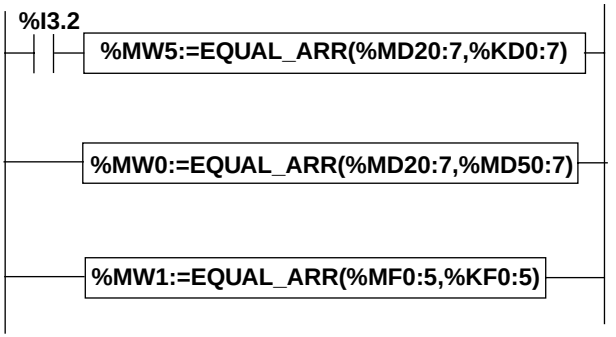
表比较指令

概要

EQUAL_ARR 功能对两个表的元素逐个进行比较。
如果找到不同，则第一个不同元素的序列号以一个字的形式返回，否则返回值等于 -1。
对整个表执行比较操作。

结构

梯形图语言



指令列表语言

LD %I3.2
[%MW5:=EQUAL_ARR(%MD20:7,KD0:7)]

LD1
[%MW0:=EQUAL_ARR(%MD20:7,%MD50:7)]

LD1
[%MW1:=EQUAL_ARR(%MF0:5,%KF0:5)]

语法

表比较指令语法：

```
Res:=EQUAL_ARR(Tab1,Tab2)
```

表比较指令参数：

类型	结果 （**Res**）	表 （**Tab 1** 和 **Tab 2**）
双字表	%MWi	%MDi:L,%KDi:L
浮点字表	%MWi	%MFi:L,%KFi:L

注意：
- 表的长度和类型必须相同。

举例

%MW5:=EQUAL_ARR(%MD30:4,%KD0:4)

2 个表的比较：

序列号	字表	常字表	区别
0	%MD30=10	%KD0=10	=
1	%MD31=20	%KD1=20	=
2	%MD32=30	%KD2=60	区别
3	%MD33=40	%KD3=40	=

字 %MW5 的值是 2 （第一个不同元素的序列号）

表查找指令

概要

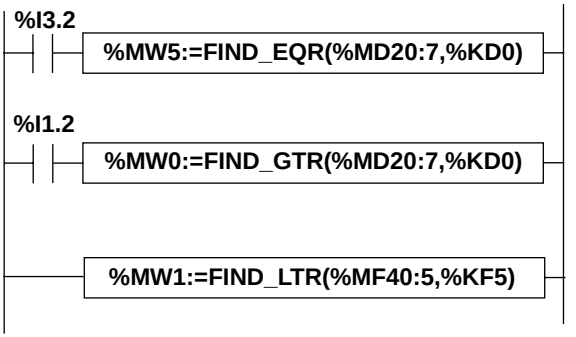
有 3 个查找函数：

- **FIND_EQR**：在双字或浮点字表中查找与给定值相等的第一个元素的位置
- **FIND_GTR**：在双字或浮点字表中查找比给定值大的第一个元素的位置
- **FIND_LTR**：在双字或浮点字表中查找比给定值小的第一个元素的位置

如果找到符合条件的第一个元素，则指令的结果等于它的序列号，否则等于 -1。

结构

梯形图语言



指令列表语言

LD %I3.2
[%MW5:=FIND_EQR(%MD20:7,KD0)]
LD %I1.2
[%MW0:=FIND_GTR(%MD20:7,%KD0)]
LD1
%MW1:=FIND_LTR(%MF40:5,%KF5)

语法

表查找指令语法：

功能	语法
FIND_EQR	Res：= 函数 （Tab,Val）
FIND_GTR	
FIND_LTR	

浮点字和双字表查找指令参数：

类型	结果 （Res）	表 （Tab）	值 （val）
浮点字表	%MWi	%MFi:L,%KFi:L	%MFi,%KFi
双字表	%MWi	%MDi:L,%KDi:L	%MDi,%KDi

举例

%MW5: =FIND_EQR(%MD30: 4,%KD0)
查找表中 =%KD0=30 的第一个双字的位置：

序列号	字表	结果
0	%MD30=10	-
1	%MD31=20	-
2	%MD32=30	值 （val），序列号
3	%MD33=40	-

表最大值和最小值查找功能

概要

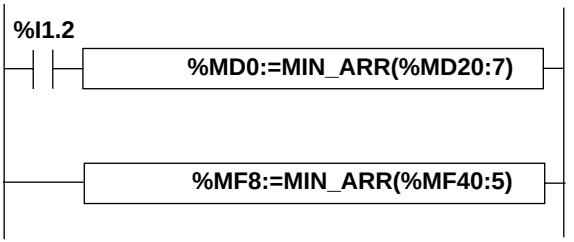
有 2 个查找函数：

- **MAX_ARR**：查找双字或浮点字表中的最大值
- **MIN_ARR**：查找双字或浮点字表中的最小值

这些指令的结果等于表中发现的最大值 （或最小值）。

结构

梯形图语言



指令列表语言

LD %I1.2
[%MD0:=MIN_ARR(%MD20:7)]
LD1
[%MF8:=MIN_ARR(%MF40:5)]

语法

表最大值和最小值查找指令语法：

功能	语法
MAX_ARR	Res：= 函数 （Tab）
MIN_ARR	

表最大值和最小值查找指令参数：

类型	结果 （**Res**）	表 （**Tab**）
双字表	%MDi	%MDi:L,%KDi:L
浮点字表	%MFi	%MFi:L,%KFi:L

表中某个值的出现次数

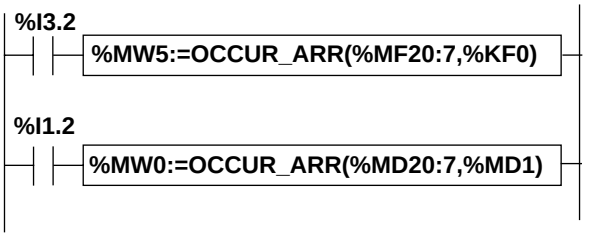
概述

这个查找函数：

- **OCCUR_ARR**：查找双字或浮点字表中与给定值相等的元素的个数

结构

梯形图语言



指令列表语言

LD %I3.2
[%MW5:=OCCUR_ARR(%MF20:7,%KF0)]
LD %I1.2
[%MW0:=OCCUR_ARR(%MD20:7,%MD1)

语法

表值频次统计指令语法：

功能	语法
OCCUR_ARR	Res：= 函数 （Tab,Val）

表值频次统计指令参数：

类型	结果 （**Res**）	表 （**Tab**）	值 （**Val**）
双字表	%MWi	%MDi:L,%KDi:L	%MDi,%KDi
浮点字表	%MWi	%MFi:L,%KFi:L	%MFi,%KFi

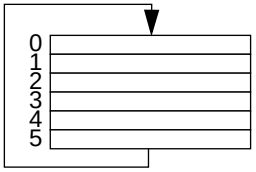
表循环移动功能

概述

有 2 个表移动函数：

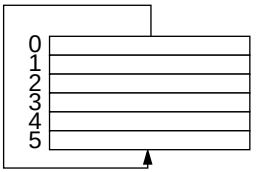
- **ROL_ARR**：在浮点字表中从上到下循环移动 n 个位置

ROL_ARR 功能图解



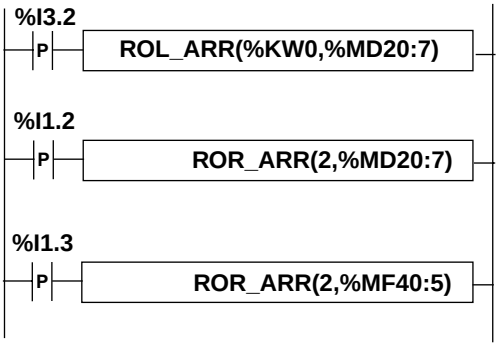
- **ROR_ARR**：在浮点字表中从下到上循环移动 n 个位置

ROR_ARR 功能图解



结构

梯形图语言



指令列表语言

LDR %I3.2
[ROL_ARR(%KW0,%MD20:7)]
LDR %I1.2
[ROR_ARR(2,%MD20:7)]
LDR %I1.3
[ROR_ARR(2,%MF40:5)]

语法

浮点数或双字表循环移动指令 ROL_ARR 和 ROR_ARR

功能	语法
ROL_ARR	功能 （n,Tab）
ROR_ARR	

浮点字表循环移动指令 ROL_ARR 和 ROR_ARR：

类型	位置数 （n）	表 （Tab）
浮点字表	%MWi，立即值	%MFi:L
双字表	%MWi，立即值	%MDi:L

注意：如果 n 值为负或零，则不执行移动。

表排序功能

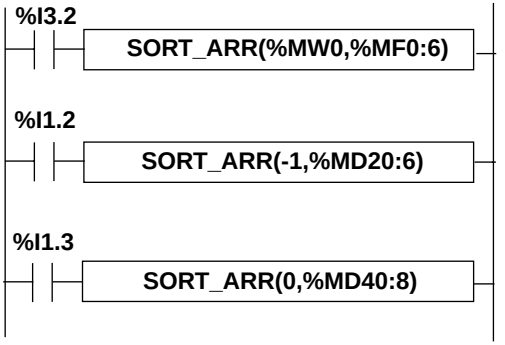
概述

有下面的排序函数可用：

- **SORT_ARR**：对一个双字或浮点字表的元素按照升序或降序进行排序并将结果存储在相同的表中。

结构

梯形图语言



指令列表语言

LD %I3.2
[SORT_ARR(%MW20,%MF0:6)]
LD %I1.2
[SORT_ARR(-1,%MD20:6)]
LD %I1.3
[SORT_ARR(0,%MF40:8)]

语法

表排序函数语法：

功能	语法
SORT_ARR	函数 （方向，表 （Tab））

- “方向”参数给出排序顺序：方向 > 0 表示升序；方向 < 0，表示降序，方向 = 0 表示不进行排序。
- 结果 （已排序表）返回到表 （Tab）参数 （要排序的表）。

表排序函数参数：

类型	排序顺序	表 （**Tab**）
双字表	%MWi，立即值	%MDi:L
浮点字表	%MWi，立即值	%MFi:L

浮点表插值功能

概要

此 LKUP 功能是用于对一个给定 X 值插入一系列 X 对 Y 的浮点数据。

插入原则

LKUP 功能按照线性内插值原则，如下公式所示：

（公式 1：） $$Y = Y_i + \left[\frac{(Y_{i+1} - Y_i)}{(X_{i+1} - X_i)} \cdot (X - X_i)\right]$$

因为 $X_i \le X \le X_{i+1}$ ，这里 $i = 1 \ldots (m-1)$；

假定 X_i 按升序排列：$X_1 \le X_2 \le \ldots X \ldots \le X_{m-1} \le X_m$.

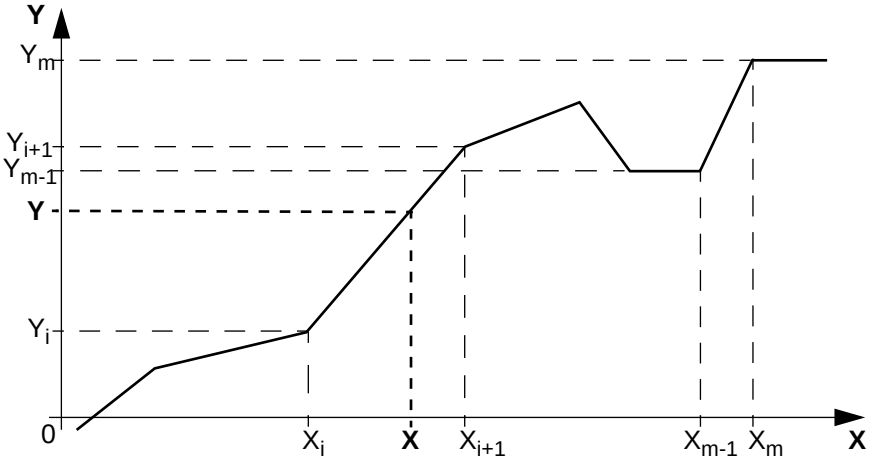
注意：如果任意两个连续的 Xi 值相等 （$X_i = X_{i+1} = X$），公式 （1）得到一个非法结果。在此情况时，与此结果相对应，将应用下面的运算法则来取代公式 （1）：

（公式 2：） $$Y = \left[\frac{(Y_{i+1} - Y_i)}{2}\right]$$

因为 $X_i = X_{i+1} = X$ ，这里 $i = 1 \ldots (m-1)$.

用图示法来表示的线性内插法原则

下面的图例描述了如上所述的线性内插法原则：



LKUP 功能的语法

LKUP 功能应用三个操作数，其中两个是函数参数，如下表所示：

语法	操作数 1（Op1） 输出变量	操作数 2（Op2） 用户定义（X）值	操作数 3（Op3） 用户定义 （Xi,Yi）变量数组
[Op1: = LKUP(Op2,Op3)]	%MWi	%MF0	整数值，%MWi 或 %KWi

Op1 定义

Op1 是保存插值功能输出结果的内存字。
通过 Op1 的值，用户可以知道插值是成功或失败，和是什么引起失败，
如下表概要所述：

Op1 (%Mwi)	描述
0	插值成功
1	插值错误：错误的数组 $X_m < X_{m-1}$
2	插值错误：Op2 超限，$X < X_1$
4	插值错误：Op2 超限， $X > X_m$
8	数组范围非法： ● Op3 设定为寄数，或 ● Op3 < 6。

注意：Op1 不包含计算的插补值 （Y）。对给定的 （X）值，插值结果 （Y）储存在 Op3 数组 %MF2 中 （见以下 *Op3 定义*）。

Op2 定义

Op2 是浮点变量 （Op3 浮点数组中的 %MF0），储存着用于计算插值 （Y）的用户自定义值 （X）：

- Op2 值的有效范围如下： $X_1 \leq Op2 \leq X_m$.

Op3 定义

Op3 （Op3/2），该数组内保存着 （X_i, Y_i）数据对。
Xi 和 Yi 存放浮点数对象的偶数索引，%MF4 开始 （注意，%MF0 和 %MF2 浮点数分别用于设定点的 X 值和插入值 Y）。
给定一个 （m）对数据构成的数组 （X_i, Y_i），浮点数组 （%MFu）的最大索引 （u）按照如下关系设定：

- *（公式 3：）* Op3=2 · m ;
- *（公式 4：）* u=2 · (Op3-1)

浮点数组 Op3 （%MFi）具有和下例相似的结构 （这里 Op3=8）：

(X)		(X_1)		(X_2)		(X_3)	
%MF0		**%MF4**		**%MF8**		**%MF12**	
	%MF2		**%MF6**		**%MF10**		**%MF14**
	(Y)		(Y_1)		(Y_2)		(Y_3)
							(Op3=8)

注意：作为上述浮点数组结构的结果， Op3 必须满足以下两个要求，否则将产生 LKUP 功能的错误：

- Op3 是一个偶数，和
- Op3 ⩾ 6 （至少两个数据点允许线性插补）。

结构

插值操作如下编程：


```
LD      %I3.2
[%MF20:=LKUP(%MF0,%KW1)]

LD      %I1.2
[%MF22:=LKUP(%MF0,10)]
```

举例

下面是使用 LKUP 插值功能的举例：
[%MW20:=LKUP(%MF0,10)]
在这个例子中：

- %MW20 是 Op1（输出变量）。
- %MF0 是用户定义的（X）值，与之相对应的（Y）值必须通过线性内插值进行计算来得到。
- %MF2 存储着线性内插计算所得的结果（Y）值。
- 10 是 Op3（通过上面的公式 3 给定）。它设置浮点数组的大小。最高项 %MFu（其中 u=18）通过上面的公式 4 给定。

这儿有四对数存储于浮点数组 Op3 [%MF4..%MF18] 内：

- %MF4 存储 X_1，%MF6 存储 Y_1。
- %MF8 存储 X_2，%MF10 存储 Y_2。
- %MF12 存储 X_3，%MF14 存储 Y_3。
- %MF16 存储 X_4，%MF18 存储 Y_4。

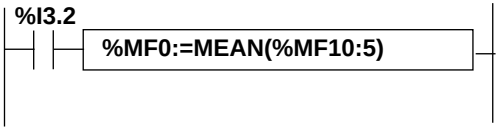
浮点表求均值功能

概述

此 MEAN 函数用于计算一个浮点表中给定数目的值的均值。

结构

梯形图语言



指令列表语言
LD %I3.2
[%MF0:=MEAN(%MF10:5)]

语法

浮点表均值计算函数语法：

功能	语法
MEAN	结果 = 函数 （Op1）

浮点表给定 L 个数的值的计算函数的参数：

操作数 （**Op1**）	结果 （**Res**）
%MFi:L, %KFi:L	%MFi

系统位和系统字

18

概览

本章的主题

本章提供了用于创建 Twido 控制器控制程序的系统位和系统字的概述。

本章包含了哪些内容?

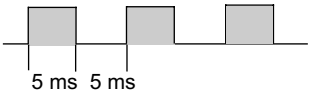
本章包含了以下主题：

系统位 （%S）

介绍

下面部分提供了有关系统位功能及怎样控制它们的详细信息。

详细描述 下面表格提供了有关系统位及怎样控制它们的概述：

系统位	功能	描述	初始状态	控制
%S0	冷启动处理	一般置为 0，下面将其置为 1： ● 电源恢复且数据丢失 （电池故障）， ● 用户程序或动态监控表编辑器， ● 操作显示器。 该位在第一次扫描时被置为 1，在下一次扫描前被系统置为 0。	0	S 或 U->S
%S1	热启动	一般置为 0，下面将其置为 1： ● 电源恢复且数据保留， ● 用户程序或动态监控表编辑器， ● 操作显示器。 该位在第一次扫描结束时被系统置为 0。	0	S 或 U->S
%S4 %S5 %S6 %S7	时基：10 ms 时基：100 ms 时基：1 s 时基：10 min	状态变化频率由内部时钟测量。它们与控制器扫描不同步。 示例：%S4  	-	S
%S8	连线测试	初始置为 1，该位用于控制器 “非配置” 状态测试连线：要修改此位的值，利用操作显示单元改变所需输出状态： ● 置为 1，输出复位， ● 置为 0，连线测试被允许。	1	U
%S9	复位输出	一般置为 0。它可以被程序或终端 （通过动态监控表编辑器）置为 1： ● 状态为 1 时，若控制器处于运行模式则输出被强制到 0， ● 状态为 0 时，输出被正常更新。	0	U
%S10	I/O 故障	一般置为 1。当检测到 I/O 故障时该位被系统置为 0。	1	S
%S11	看门狗溢出	一般置为 0。当程序执行时间 （扫描时间）超过最大扫描时间 （软件看门狗）时该位被系统置为 1。看门狗溢出导致控制器进入暂停状态。	0	S

系统位	功能	描述	初始状态	控制
%S12	PLC 处于运行模式	该位表示控制器处于运行状态。系统在控制器运行时将该位置为 1。在停止，初始化，或任何其它状态时置为 0。	0	S
%S13	运行的第一个循环	一般置为 0，在控制器变为运行状态后的第一个扫描过程中被系统置为 1。	1	S
%S17	容量超出	一般置为 0，它在下列情况被系统置为 1： ● 在循环或移动操作时。系统把此输出位转换为 1。它必须由用户程序在每次可能产生溢出的操作之后测试，溢出发生后由用户复位到 0。	0	S->U
%S18	算术溢出或错误	一般置为 0。它在进行运算时出现溢出的情况下被置为 1，它们是： ● 单字长度下结果大于 + 32 767 或小于 - 32 768， ● 双字长度下结果大于+ 2 147 483 647或小于- 2 147 483 648， ● 浮点结果大于 + 3.402824E+38 或小于 - 3.402824E+38， ● 被 0 除， ● 对负数求平方根， ● BTI 或 ITB 转换无意义：BCD 值超出限制。 它必须由用户程序在每次可能产生溢出的操作之后测试，溢出发生后由用户复位到 0。	0	S->U
%S19	扫描周期溢出 （周期扫描）	一般置为 0，该位在扫描周期溢出 （扫描时间大于用户在配置中定义或在 %SW0 中编程的周期）的情况下被系统置为 1。 该位由用户复位到 0。	0	S->U
%S20	索引溢出	一般置为 0，它在索引对象的地址小于 0 或大于对象的最大地址范围时被置为 1。 它必须由用户程序在每次可能产生溢出的操作之后测试，然后在溢出发生后复位到 0。	0	S->U

系统位	功能	描述	初始状态	控制
%S21	GRAFCET 初始化	一般置为 0，下面将其置为 1： ● 冷重启，%S0=1， ● 用户程序，且只能在预处理程序部分，使用 Set 指令 （S %S21）或设置线圈 -(S)- %S21， ● 终端。 状态为 1 时，它导致 GRAFCET 初始化。已激活步被停止且激活初始步。 它在 GRAFCET 初始化之后被系统复位到 0。	0	U->S
%S22	GRAFCET 复位	一般置为 0，能且只能被程序预处理时置为 1。 状态为 1 时它导致全部 GRAFCET 的活动步停止。 它在顺控程序开始执行时由系统复位到 0。	0	U->S
%S23	GRAFCET 预置和冻结	一般置为 0，它只能在预处理程序模块由程序置为 1。置为 1 时，它使 GRAFCET 的预置生效。维持该位在 1 将冻结 GRAFCET （冻结图表）。它在顺控程序开始执行时由系统复位到 0 以保证 GRAFCET 表从冻结状态变为活动状态。	0	U->S
%S24	操作显示	一般置为 0，该位可被用户置为 1。 ● 状态为 0 时，操作显示正常工作， ● 状态为 1 时，操作显示被冻结，保持当前显示不变，不能闪烁，且停止输入键处理。	0	U->S
%S25	选择操作显示器的显示模式	有两种显示模式：数据模式和正常模式。 ● %S25=0，正常模式有效。 在的一行，能输入对象名 （系统字，内存字，系统位）。 第二行显示当前值。 ● %S25=1，数据模式有效。 在第一行显示 %SW68。 在第二行显示 %SW69。 %S25=1，键盘操作无效。 **注意：**Firmware 版本 V3.0 或更高。	0	U

系统位	功能	描述	初始状态	控制
%S26	选择显示一有符号或无符号数在操作显示器上	两种类型可选：有符号或无符号。 ● %S26=0，有符号数显示有效。（-32768 to 32767） +/- 符号在每行的开头处。 ● %S26=1，无符号数显示有效。（0 to 65535） %S26 仅当 %S25=1 时被用。 注意：Firmware 版本 3.0 或更高。	0	U
%S31	事件标志	一般为 1。 ● 状态为 0 时，事件不能被执行且排队等待。 ● 状态为 1 时，事件可被执行， 该位能被用户或系统设为初始状态 1（冷启动）	1	U->S
%S38	允许事件进入事件队列	一般为 1。 ● 置为 0 时，事件不能进入事件队列。 ● 置为 l 时，一旦检测到事件就将 它们放置到事件队列， 该位能被用户或系统设为初始状态 1（冷启动）。	1	U->S
%S39	事件队列饱和	一般为 0。 ● 置为 0 时，所有事件都被报告， ● 置为 1 时，至少一个事件被丢失。 该位可由用户和系统 （在冷重启情况下）置为 0。	0	U->S
%S50	使用字 %SW49 到 %SW53 更新日期和时间	一般置为 0，该位可被程序或操作显示置为 1。 ● 置为 0 时，日期和时间均只可读， ● 置为 1 时，日期和时间可被更新。 控制器内部 RTC 在 %S50 下降沿被刷新。	0	U->S

系统位	功能	描述	初始状态	控制
%S51	日历时钟状态	一般置为 0，该位可被程序或操作显示置为 1。 ● 置为 0 时，日期和时间是不可变的， ● 置为 1 时，日期和时间必须由用户初始化。 当该位置为 1 时，日期时钟的时间数据无效。日期和时间可能从未配置过，电池可能电压低，或控制器 RTC 修正量不正确 （未配置，修正值和保存值不同，或超出范围）。状态 1 到状态 0 的转变强制写入修正常量到 RTC。	0	U->S
%S52	RTC 错误	由系统管理的此位表示 RTC 修正值还未输入，且时间和日期是错误的。 ● 置为 0 时，日期和时间是不可变的， ● 置为 1 时，日期和时间必须被初始化。	0	S
%S59	使用字 %SW59 更新日期和时间	一般置为 0，该位可被程序或操作显示置为 1。 ● 置为 0 时，不能管理系统字 %SW59， ● 置为1时，日期和时间根据 %SW59 设置的控制位的上升沿增加或减少。	0	U
%S66	BAT LED （电池指示灯）显示激活 / 关闭（仅有支持外部电池的控制器型号：TWDLCA・40DRF）	该系统位可由用户设定，它允许用户点亮或关掉 BAT LED （电池指示灯）： ● 设为 0 时，BAT LED 被激活（在上电时，被系统复位到 0）， ● 设为 1 时，BAT LED 被关闭（这时即使外部电池电压低或没有外部电池， LED 也不被点亮）。	0	S 或 U->S
%S69	用户 STAT LED 显示	置为 0 时， STAT LED 关断。 置为 1 时， STAT LED 打开。	0	U
%S75	外部电池状态 （仅有支持外部电池的控制器型号：TWDLCA・40DRF.）	该系统位由系统设定，它指示外部电池的状态，可由用户读取： ● 设定为 0 时，外部电池工作正常， ● 设定为 1 时，外部电池电量低，或没装外部电池。	0	S

系统位	功能	描述	初始状态	控制
%S95	恢复存储字	当前面存储内存字到内部 EEPROM 时，可以设置该位。完成后系统将该位置回 0 且恢复的内存字数置于 %SW97	0	U
%S96	备份程序完成	该位可在任何时刻被读取 （被程序读或调整时读），特别是在冷启动或热重启之后。 ● 置为 0 时，备份程序无效。 ● 置为 1 时，备份程序有效。	0	S
%S97	保存 %MW 完成	该位可在任何时刻被读取 （被程序读或调整时读），特别是在冷启动或热重启之后。 ● 置为 0 时，保存 %MW 无效。 ● 置为 1 时，保存 %MW 有效。	0	S
%S100	TwidoSoft 通信电缆连接	显示 TwidoSoft 通信电缆是否已连接。 ● 置为 1 时，没有连接 TwidoSoft 通信电缆或是 TwidoSoft。 ● 置为 0 时， TwidoSoft 通信电缆已连接。	-	S
%S101	端口地址 （Modbus 协议）改变	用系统字 %SW101 （端口 1） %SW102 和 （端口 2）来改变端口地址。为改变端口地址， %S101 必须置为 1。 ● 置为 0, 地址不能被改变。%SW101 和 %SW102 的值与当前端口地址相匹配， ● 置为 1，通过改变 %SW101（端口 1）和 %SW102（端口 2）的值可修改其地址。系统字修改完毕后， %S101 必须被置 0。	0	U

系统位	功能	描述	初始状态	控制
%S103 %S104	使用 ASCII 协议	准许在 Comm 1 （%S103）或 Comm 2 （%S104）上使用 ASCII 协议。 ASCII 协议通过系统字进行配置， %SW103 和 %SW105 配置 Comm 1， %SW104 和 %SW106 配置 Comm 2。 ● 设定为 0 时，它执行 TwidoSoft 中配置的协议， ● 置为 1， ASCII 协议用于 Comm 1 （%S103）或 Comm 2 （%S104）， %SW103 和 %SW105 必须提前配置好，且用于 Comm 1， %SW104 和 %SW106 用于 Comm 2。	0	U
%S110	远程连接交换	由程序或终端将此位复位为 0。 ● 对主机，置为 1 表示所有的远程连接交换 （仅远程 I/O）完成。 ● 对从机，置为 1 表示和主机的交换完成。	0	S->U
%S111	单一远程连接交换	● 对主机，置为 0 表示单一远程连接交换完成。 ● 对主机，置为 1 表示单一远程连接交换处于进行中。	0	S
%S112	连接远程连接	● 对主机，置为 0 表示远程连接处于激活状态。 ● 对主机，置为 1 表示远程连接处于非活动状态。	0	U
%S113	远程连接配置 / 操作	● 对主机或从机，置为 0 表示远程连接配置 / 操作完成。 ● 对主机，置为 1 表示其远程连接配置 / 操作出错。 ● 对从机，置为 1 表示其远程连接配置 / 操作出错。	0	S->U
%S118	远程 I/O 出错	一般置为 1。当远程连接检测到 I/O 故障时该位被置为 0。	1	S
%S119	本地 I/O 出错	一般置为 1。当检测到 I/O 故障时该位被置为 0。 %SW118 决定故障种类。 当故障消除时复位到 1。	1	S

表缩写描述

缩写表：

缩写	描述
S	系统控制
U	用户控制
U->S	由用户置为 1，由系统复位到 0
S->U	由系统置为 1，由用户复位到 0

系统字 （%SW）

介绍

下面部分提供了有关系统字功能及怎样控制它们的详细信息。

详细描述 下面表格提供了有关系统字功能及怎样控制它们的详细信息。

系统字	功能	描述	控制
%SW0	控制器扫描周期 （周期任务）	通过用户程序在动态监控表编辑器中修改配置中定义的控制器扫描周期。	U
%SW1	保存周期事件的周期	修改周期事件 [5-255 ms]，不丢失在扫描模式菜单中的周期设定值。 允许恢复扫描模式菜单中的周期设定值： ● 冷启动或 ● 写入 %SW1 的值在 [5-255] 范围之外。 %SW1 能在每个扫描结束时修改，用程序或动态数据表，不必停止程序。周期时间能被监控。	U
%SW6	控制器状态	控制器状态： 0 = 没有配置 2 = 停止 3 = 运行 4 = 暂停	S

系统字	功能	描述	控制

%SW7	控制器状态	● 位 [0]：备份 / 恢复处理：
		  ● 置为 1 表示备份 / 恢复在处理中，
		  ● 置为 0 表示备份 / 恢复已完成或不能进行。
		● 位 [1]：控制器的配置完成：
		  ● 置为 1 表示配置完成。
		● 位 [3..2] EEPROM 状态位：
		  ● 00 = 没有卡
		  ● 01 = 32 Kb EEPROM 卡
		  ● 10 = 64 Kb EEPROM 卡
		  ● 11 = 保留给将来使用
		● 位 [4]：RAM 中应用程序与 EEPROM 不同：
		  ● 置为 1 表示 RAM 应用程序与 EEPROM 不同。
		● 位 [5]：RAM 应用程序与备份卡不同：
		  ● 置为 1 表示 RAM 应用程序与备份卡不同。
		● 位 [6] 不被使用 （状态为 0）
		● 位 [7]：控制器保留：
		  ● 置为 1 表示保留。
		● 位 [8]：应用程序处于写模式：
		  ● 置为 1 表示应用程序受保护。
		● 位 [9] 不被使用 （状态为 0）
		● 位 [10]：第二个串口已安装：
		  ● 置为 1 表示已安装。
		● 位 [11]：第二个串口类型：(0 = EIA RS-232, 1 = EIA RS-485)：
		  ● 置为 0 = EIA RS-232
		  ● 置为 1 = EIA RS-485
		● 位 [12]：内部存储中应用程序有效：
		  ● 置为 1 表示应用程序有效。
		● 位 [13] 备份卡中应用程序有效：
		  ● 置为 1 表示应用程序有效。
		● 位 [14] RAM 中应用程序有效：
		  ● 置为 1 表示应用程序有效。
		● 位 [15]：准备执行：
		  ● 置为 1 表示已准备执行。
			S

%SW11	软件看门狗值	包含看门狗的最大值。值 （10 到 500 ms）由配置定义。	U
%SW14	商业版本， Vxx.yy	举例，如 %SW14=0232： ● 8 MSB=02 （十六进制）， xx=2 （十进制） ● 8 LSB=32 （十六进制）， yy=50 （十进制） Firmware 是 V2.50。 **注意：**Firmware 2.5 或更高。	S

系统字	功能	描述	控制
%SW15	Firmware 补丁， Pzz	举例，如 %SW15=0005： ● 8 MSB 不用 ● 8 LSB=05 （十六进制）， zz=5 （十进制） Firmware 补丁是 P05。 **注意：**Firmware 2.5 或更高。	S
%SW16	Firmware 版本， Vxx.yy	举例，如 %SW16=0232： ● 8 MSB=02 十六进制， xx=2 十进制 ● 8 LSB=32 十六进制， yy=50 十进制 Firmware 是 V2.50。 **注意：**Firmware 版本 2.5 或更高。	S
%SW17	浮点运算默认状态	当浮点算术运算检测到出错时，位 %S18 被置为 1 且 %SW17 的缺省状态根据下面代码得到更新： ● 位 [0]：无效运算，结果不是一个数 （1.#NAN 或 -1.#NAN）， ● 位 1：保留， ● 位 2：被 0 除，结果为无穷大 （-1.#INF 或 1.#INF）， ● 位 3：结果绝对值大于 +3.402824e+38，结果为无穷大 （-1.#INF 或 1.#INF）。	S 和 U
%SW18- %SW19	100 ms 绝对定时 计数器	计数器工作使用这两个字： ● %SW18 表示低位有效字， ● %SW19 表示高位有效字。	S 和 U
%SW20 到 %SW27	为 CANopen 地址 为 1-16 号从站提供状态指示	更详细信息，请参照 *CANopen 从站预留特定系统字， p.296*。	S
%SW30	上一次扫描时间	显示上一次控制器扫描时间 （ms）。 **注意：**这个时间对应一个扫描循环从开始 （输入请求）到结束 （输出更新）的时间。	S

系统字	功能	描述	控制
%SW31	最大扫描时间	显示自上一次冷启动以来最长的控制器扫描时间 （ms）。 **注意：** ● 这个时间对应一个扫描循环从开始 （输入请求）到结束 （输出更新）的时间。 ● 当选择输入锁存时，允许检测脉冲信号，脉冲宽度 （T_{ON}）和周期 （T_{pulse}）需满足以下条件： ● $T_{ON} \geq 1$ ms ● 其脉宽(Tpulse 必须大于在系统字%SW31中记录的最长扫描时间的两倍，即如下所示： $T_{pulse} \geq 2$ x %SW31. **注意：**如此条件不满足，一些脉冲可能丢失。	S
%SW32	最小扫描时间	显示自上一次冷启动以来最短的控制器扫描时间 （ms）。 **注意：**这个时间对应一个扫描循环从开始 （输入请求）到结束 （输出更新）的时间。	S
%SW48	事件数	显示自上一次冷启动以来执行了多少个事件。（周期时间除外）。 **注意：**初始设置为 0 （在应用程序装载和冷启动之后），每个事件的执行后，其计数增加。	S

系统字	功能	描述		控制
%SW49 %SW50 %SW51 %SW52 %SW53	实时时钟 （RTC）	RTC 功能：包含当前日期和时间值的系统字 （BCD 码格式）：		S 和 U
		%SW49	每星期中第几天 （N=1 表示星期一）	
		%SW50	00SS 秒	
		%SW51	HHMM 时和分	
		%SW52	MMDD 月和日	
		%SW53	CCYY 世纪和年	
		当位 %S50 被置 0 时，这些字由系统控制。当位 %S50 被置为 1 时，这些字可由用户程序或终端写入。		
%SW54 %SW55 %SW56 %SW57	上一次停止的日期和时间	系统字包含上一次电源故障或控制器停止的日期和时间 （BCD 格式）：		S
		%SW54	SS 秒	
		%SW55	HHMM 时和分	
		%SW56	MMDD 月和日	
		%SW57	CCYY 世纪和年	
%SW58	上一次停止的代码	显示上一次停止的原因代码		S
		1 =	运行 / 停止输入沿	
		2 =	因软件故障停止 （控制器扫描溢出）	
		3 =	停止命令	
		4 =	电源中断	
		5 =	因硬件故障停止	

系统字	功能	描述			控制
%SW59	调节当前日期	调节当前日期。 包含两组 8 位的数据，用来调节当前日期的设置。 该操作在位的上升沿执行。该字可用与否决定于位 %S59。			U
		增加	**减少**	**参数**	
		第 0 位	第 8 位	星期几	
		第 1 位	第 9 位	秒	
		第 2 位	第 10 位	分	
		第 3 位	第 11 位	时	
		第 4 位	第 12 位	天	
		第 5 位	第 13 位	月	
		第 6 位	第 14 位	年	
		第 7 位	第 15 位	世纪	
%SW60	RTC 修正	RTC 修正值			U
%SW63	EXCH1 模块错误代码	EXCH1 错误代码： 0 - 操作成功 1 - 传输字节数太大 （> 250） 2 - 发送表太小 3 - 字表太小 4 - 接收表溢出 5 - 超时 6 - 发送 7 - 表中含有错误命令 8 - 所选通讯口未配置 / 不可用 9 - 接收错误 10 - 接收时不能使用 %KW 11 - 发送偏移大于发送表 12 - 接收偏移大于接收表 13 - 控制器停止 EXCH 进程			S
%SW64	EXCH2 模块错误代码	EXCH2 错误代码：见 %SW63。			S

系统字	功能	描述	控制
%SW65	EXCH3 错误代码	EXCH3 错误代码仅在具有 Ethernet 功能的 TWDLCAE40DRF Twido 控制器上被执行。 1-4， 6-13：见 %SW63 （注意：错误代码 5 在此无效，而被 Ethernet 特定错误代码 109 和 122 所代替，见如下所述。） 以下是 Ethernet 的错误代码： 101 - 无此 IP 地址 102 - TCP 连接中断 103 - 无 socket 可用 （所有连接信道都处于繁忙状态） 104 - 网络断开 105 - 网络不可达 106 - 重启时网络中断连接 107 - 由对等设备中断了连接 108 - 由对等设备重置连接 109 - 连接超时 110 - 拒绝连接请求 111 - 主机关闭 120 - 未知索引 （远程设备在配置表中无索引） 121 - 重大故障 （MAC ；芯片；重复 IP） 122 - 数据发送后接收超时 123 - 以太网正在初始化	S
%SW67	控制器功能和类型	包含下列信息： ● 控制器类型位 [0 -11] ● 8B0 = TWDLC • 10DRF ● 8B1 = TWDLC • 16DRF ● 8B2 = TWDLMDA20DUK/DTK ● 8B3 = TWDLC • 24DRF ● 8B4 = TWDLMDA40DUK/DTK ● 8B6 = TWDLMDA20DRT ● 8B8 = TWDLCAA40DRF ● 8B9 = TWDLCAE40DRF ● 位 12, 13, 14, 15 不被使用 = 0	S

系统字	功能	描述	控制
%SW68 和 %SW69	能同时显示在操作显示器上	如 %S25=1，数据显示模式有效。键盘操作无效。 %SW68 和 %SW69 能同时显示在操作显示器上： ● %SW68 在第一行， ● %SW69 在第二行。 **注意：**Firmware 需 V3.0 和更高。	U

系统字	功能	描述	控制
%SW73 和 %SW74	AS-I 系统状态	● 位 [0]：置为 1 表示配置完成。 ● 位 [1]：置为 1 表示数据交换被激活。 ● 位 [2]：置为 1 表示模块处于离线模式。 ● 位 [3]：置为 1 表示 ASI_CMD 指令结束。 ● 位 [4]：置为 1 表示 ASI_CMD 指令正在处理。	S 和 U
%SW76 到 %SW79	减计数器 1-4	这 4 个字用作 1 ms 定时器。如果它们为一个正值则它们分别被系统减计数。这等于提供了 4 个以毫秒为单位的减计数器，工作范围为 1 ms 到 32767 ms。位 15 置 1 可以停止减计数。	S 和 U
%SW80	本体 I/O 状态	位 [0] 通道处于正常操作 （对于通道所有操作） 位 [1] 模块初始化 （或所有通道的信息初始化） 位 [2] 硬件故障 （外部电源故障，所有通道的普通故障） 位 [3] 模块配置错误 位 [4] 通道 0 输入数据转换处理中 位 [5] 通道 1 输入数据转换处理中 位 [6] 通道 0 输入热电偶没有配置 位 [7] 通道 1 输入热电偶没有配置 位 [8] 不被使用 位 [9] 没有使用 位 [10] 通道 0 模拟输入数据超出范围 位 [11] 通道 1 模拟输入数据超出范围 位 [12] 连线错误 （通道 0 模拟输入数据低于电流范围，电流回路开路） 位 [13] 连线错误 （通道 1 模拟输入数据低于电流范围，电流回路开路） 位 [14] 没有使用 位 [15] 输出通道不可用	S
%SW81	● 扩展 I/O 模块 1 状态：定义与 %SW80 相同 ● CANopen 主模块位于第一扩展位置时的状态： ● 位 [0] 配置状态 （1 = 配置完成； 0 = 配置错误） 位 [1] 运行状态 （1 = PDO 交换打开； 0 = PDO 交换关闭） 位 [2] 初始化状态 （1 = 初始化状态 ON ； 0 = 初始化状态 OFF） 位 [3] CAN_CMD 指令完成 （1 = 完成； 0 = 进行中） 位 [4] CAN_CMD 指令错误 （1 = 错误； 0 = OK） 位 [5] 初始化错误 （1 = 错误； 0 = OK） 位 [6] 信息丢失，电源错误 （1 = 错误， 0 = OK）		S
%SW82	扩展 I/O 模块 2 状态：定义同 %SW80 CANopen 主模块位于第二扩展位置时的状态同 %SW81		S

系统字	功能	描述	控制
%SW83	扩展 I/O 模块 3 状态：定义同 %SW80 CANopen 主模块位于第三扩展位置时的状态定义同 %SW81		S
%SW84	扩展 I/O 模块 4 状态：定义同 %SW80 CANopen 主模块位于第四扩展位置时的状态定义同 %SW81		S
%SW85	扩展 I/O 模块 5 状态：定义同 %SW80 CANopen 主模块位于第五扩展位置时的状态定义同 %SW81		S
%SW86	扩展 I/O 模块 6 状态：定义同 %SW80 CANopen 主模块位于第六扩展位置时的状态定义同 %SW81		S
%SW87	扩展 I/O 模块 7 状态：定义同 %SW80 CANopen 主模块位于第七扩展位置时的状态定义同 %SW81		S
%SW94	应用程序签名	当程序改变时，根据配置或编程数据，签名 （所有和校验的总和）会相应改变。 %SW94=91F3 （十六进制），签名是 91F3。 **注意：**Firmware 版本 V2.5 或更高。	S

%SW96	应用程序和 %MW 存储 / 恢复功能的命令和 / 或诊断。	● 位 [0]：表示 %MW 存储字需要存在 EEPROM 中：
		  ● 如果需要备份，置为 1，
		  ● 若备份未完成，置为 0。
		● 位 [1]：该位由 firmware 设置何时保存完成：
		  ● 备份完成该位置为 1，
		  ● S 若有新的备份请求该位置为 0。
		● 位 [2]：备份错误，参考位 8， 9， 10 和 14 以获取进一步信息。
		  ● 错误出现，置为 1，
		  ● 若有新的备份请求该位置为 0。
		● 位 [6]：若控制器内是空应用程序，该位置为 1。
		● 位 [8]：表明在 %SW97 中指定的 %MWs 个数大于程序中配置的 %MWs 个数：
		  ● 检测到错误，置为 1，
		● 位 [9]：表明在 %SW97 中指定的 %MWs 个数大于在 TwidoSoft 中应用程序指定的 %MWs 最多个数。
		  ● 检测到错误，置为 1，
		● 位 [10]：内部 RAM 和内部 EEPROM 内容有区别 （1 = 是）。
		  ● 若有区别，置为 1。
		● 位 [14]：表明 EEPROM 写错误发生：
		  ● 检测到错误，置为 1，
			S 和 U

系统字	功能	描述	控制
%SW97	存储 / 恢复功能的命令和诊断	当要备份内存字时，此值表示将要储存到内部 EEPROM 的 %MW 物理个数。当恢复存储字时，此值被上载到 RAM 的存储字个数更新。对于存储操作，当该值被置为 0 时，存储字不会被保存。用户必须定义用户逻辑程序。否则，该字在控制器应用程序中被置 0，除了下述情况：冷启动时，该字置为 -1 表示内部 Flash EEPROM 没有被保存的存储字 **%MW** 文件。冷启动时，如果内部闪存 EEPROM 包含存储字 **%MW**，文件中的存储字个数值必须设定在系统字 **%SW97**。	S 和 U

系统字	功能	描述	控制
%SW101 %SW102	通讯口 Modbus 地址	当 %S101 设定为 1，可以修改口 1 或口 2 的 Modbus 地址。口 1 的地址是 %SW101，口 2 的地址是 %SW102。	S

<table>
<tr><th>系统字</th><th>功能</th><th>描述</th><th>控制</th></tr>
<tr><td>%SW103
%SW104</td><td>ASCII 协议使用的配置 I</td><td>当位 %S103 （Comm 1）或 %S104 （Comm 2）置为 1 时，使用 ASCII 协议。系统字 %SW103 （Comm 1）或 %SW104 （Comm 2）的设定必须按照如下原理：
<table>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="8">字符串结束</td><td>数据位</td><td>停止位</td><td colspan="2">奇偶校验</td><td>RTS / CTS</td><td colspan="3">波特率</td></tr>
</table>

波特率：

0：1200 波特，
1：2400 波特，
2：4800 波特，
3：9600 波特，
4：19200 波特，
5：38400 波特。

RTS/CTS：

0：未激活，
1：激活，

奇偶校验：

00: 无，
10: 奇，
11: 偶 .

停止位：

0：1 个停止位，
1：2 个停止位。

数据位：

0：7 个数据位，
1：8 个数据位。

</td><td>S</td></tr>
<tr><td>%SW105
%SW106</td><td>ASCII 协议使用的配置</td><td>当位 %S103 （Comm 1）或 %S104 （Comm 2）置为 1 时，使用 ASCII 协议。系统字 %SW105 （Comm 1）或 %SW106 （Comm 2）必须按照如下原理设定：
<table>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="8">超时帧，单位 ms</td><td colspan="8">响应超时整倍 100 ms</td></tr>
</table></td><td>S</td></tr>
<tr><td>%SW111</td><td>远程连接状态</td><td>说明：位 0 对应远程控制器 1，位 1 对应远程控制器 2，如此等等。
位 [0] 到 [6]：

置为 0 = 远程控制器 1-7 缺失
置为 1 = 远程控制器 1-7 未缺失
位 [8] 到位 [14]：

置为 0 = 远程控制器 1-7 上检测到远程 I/O
置为 1 = 远程控制器 1-7 上检测到扩展控制器
</td><td>S</td></tr>
</table>

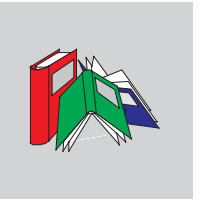
系统字	功能	描述	控制
%SW112	远程连接配置 / 操作错误码	00：操作成功 01：检测到超时 （从） 02：检测到校验和错误 （从） 03：配置不当 （从） 该字由系统设置为 1 且必须由用户重置。	S
%SW113	远程连接配置	说明：位 0 对应远程控制器 1，位 1 对应远程控制器 2，如此等等。 位 [0] 到 [6]： ● 置为 0 = 远程控制器 1-7 未配置 ● 置为 1 = 远程控制器 1-7 配置正常 位 [8] 到位 [14]： ● 置为 0 = 远程控制器 1-7 配置为远程 I/O ● 置为 1 = 远程控制器 1-7 配置为对等控制器	S
%SW114	调度模块启动	用户程序或操作显示来启动或关闭调度模块。 位 0：1 = 启动调度模块 #0 ... 位 15：1 = 启动调度模块 #15 初始时所有调度模块被启动。 如果调度模块被配置，默认值为 FFFF 如果调度模块未被配置，默认值为 0。	S 和 U
%SW118	控制器本体状态字	检测到的错误。 位 9：0 = 外部或通信错误。 位 12：0 = RTC 未安装 位 13：0 = 配置出错 （配置了 I/O 扩展但空缺或故障）。 本字的所有其他位置 1 且保留。对于无故障的控制器，本字的值为 FFFFh。	S
%SW120	扩展 I/O 模块状态	每个模块一位。 地址 0 = 位 0 1 = 有问题 0 = 没问题	S

表缩写描述

缩写表：

缩写	描述
S	系统控制
U	用户控制

术语



!

% 前缀，表示控制器中内部存储器地址，用于存储程序变量值，常量值， I/O 值，等等。

A

地址 控制器的内部寄存器，用来存储程序变量、常量、 I/O 值等等。地址用一个百分符号 （%）前缀来标识，例如：%I0.1 表示控制器 RAM 内存中的一个地址，用以存储第一个输入的值。

模拟电位器 一个应用电压，可由应用程序调节和转换到数字值。

编译程序 用来编译程序并检查程序错误的命令：语法和结构错误，符号与地址不对应，程序使用资源不存在，以及程序与可用控制器存储器不匹配。错误被显示在程序错误浏览器中。

动态监控表 在程序编写窗口或运行画面中生成的表格，当 PC 连接到控制器时，提供控制器变量查看并允许调试时输入强制值。可以存为以 .tat 为后缀的一个独立文件。

动态监控表编辑器	一个 TwidoSoft 应用程序中的专用窗口，用于查看和创建动态监控表。
应用程序	TwidoSoft 应用程序由程序，配置数据，符号，和文档组成。
应用程序浏览器	TwidoSoft 中的一个专用窗口，以树状结构显示应用程序，并能方便地配置及查看应用程序。
应用程序文件	Twido 应用程序，文件后缀为 .twd.
ASCII	美国信息交换标准码。一种通信协议，使用七位表示文字数字式字符，包括字母，数字，和一些图形及控制字符。
行自动验证	当在指令表中插入或修改指令时，该可选设定允许程序自行验证每行中的错误及未定义变量。所以在退出该行前必须更正每个元素。是否选择在参数对话框中设定。
自动装载	当控制器 RAM 中的程序丢失或紊乱时，该功能允许将外部存储器中的应用程序自动传送到控制器的 RAM 中。上电时，控制器将目前控制器 RAM 中的应用程序与可选备份存储卡 （如果安装的话）中的应用程序进行比较，如果存在不同，则将备份卡中的程序复制到控制器和内部 EEPROM。如果备份卡没有安装，那么在内部 EEPROM 的应用程序被复制到控制器。

B

备份	一个命令，复制控制器 RAM 中的应用程序到控制器内部 EEPROM 或可选备份存储卡 （如果安装的话）。
BootP	一个以 UDP/IP 为基础的协议 （引导程序协议），它允许引导主机动态配置自身而不用用户监督。引导程序协议提供一种方式通知主机分配给它的 IP 地址。

C

CAN 控制器区域网络：一种现场总线，最初用于汽车生产，现在应用于很多领域，从工业到其他行业。

CiA 自动化中的 CAN：应用 CAN 技术产品的用户及生产商的国际组织。

客户端 响应来自其他计算机过程的请求服务的计算机过程。

COB 通信对象：CAN 总线传送单元，每个 COB 都由一个唯一的标识符来区分，该标识符有 11 位编码，[0，2047]。每个 COB 最多包含 8 个字节数据，COB 的发送优先级由其标识符决定—标示码越小，相关的 COB 的权限就越大。

线圈 一个梯形图元素，表示控制器的一个输出。

冷启动或重启 控制器每次起动，都会将所有数据恢复到初始值，并且程序会在将所有变量清除后开始运行。所有的软件和硬件被初始化。装载一个新的应用程序到控制器 RAM 可能导致冷重启。任何一个不带电池备份的控制器总是在上电时冷启动。

注释行 在指令表程序中，注释可以与指令在不同行输入。注释行不含有行号，且必须插入到圆括号和星号内，例如：（*COMMENTS GO HERE*）。

注释 注释是用来对程序进行必要解释的文本。对梯形图程序，在梯级标题处输入最多三行文本来描述本梯级的目的。每行由 1 到 64 个字符组成。对指令表程序，在未编号的程序行输入文本，且必须插入到圆括号和星号内，例如：（*COMMENTS GO HERE*）。

一体型控制器 Twido 控制器的一个类型，简单，全内置配置，扩展受限。模块型是 Twido 控制器的其它类型。

配置编辑器 专用 TwidoSoft 窗口，用于管理硬件和软件配置。

常量 一个配置值，程序执行时不能修改。

触点 梯形图的一个元件，代表控制器的一个输入点。

计数器	一个功能模块，用于事件计数 （加或减计数）。
交叉索引	生成应用程序中已经使用的操作数，符号，行 / 梯级号和运算符的列表，用于简化应用程序的创建和管理。
交叉索引浏览器	TwidoSoft 应用程序中的一个专用窗口，用于查看交叉索引。

D

数据变量	见变量。
日期 / 时钟功能	用于对事件进行基于月、日、时的控制。见调度模块。
默认网关	网络或主机的 IP 地址，它所有的数据包寻址于未知网络或主机发送。通常默认网关都是一个路由器或其他设备。
鼓控制器	一个功能模块，其工作类似于机电式鼓型 （凸轮）控制器，步的改变与外部事件有关。

E

EDS	电子数据表：每个 CAN 设备的说明文件 （由生产商提供）。
EEPROM	可电擦除且可编程的只读存储器。 Twido 有一个内部 EEPROM 和一个可选外部 EEPROM 存储卡。
擦除	该命令可以将控制器中的应用程序擦除，并提供两种选择： ● 删除控制器 RAM、控制器内部 EEPROM，和可选备份卡的内容。 ● 只删除可选备份卡的内容。
执行装载器	32 位 Windows 应用程序，用来下载 Firmware 到 Twido 控制器。
扩展总线	扩展 I/O 模块通过这些总线连接到基本控制器。

扩展 I/O 模块	可选 I/O 扩展模块，通过它们可以增加 Twido 控制器的 I/O 点数。（不是所有控制器型号都允许扩展）

F

快速计数器	一个能提供比普通计数器模块计数频率更快的加、减计数的功能块，最高频率可达 5 KHz。
FIFO	先进先出。用于队列操作的功能块。
Firmware 执行系统	Firmware 执行系统是操作系统，执行您的应用程序和管理控制器工作。
强制	设定输入或输出值为 0 或 1，即使其真实值与此不同。在动态监控程序时用于调试。
帧	组成离散信息块的一组二进制位。帧包含网络控制信息或数据。帧的大小和组成取决于使用的网络结构。
帧类型	两种常见的帧类型是 Ethernet II 和 IEEE 802.3。
功能模块	一个程序单位，输入和变量基于一个定义的功能组织起来计算输出值，如定时器或计数器。

G

网关	连接不同网络结构的网络设备，工作在应用层。这个术语可参考路由器。
Grafcet	Grafcet 是用结构框图的方式表示顺序控制功能。 它是一个分析方法，将顺序控制系统分解为一系列的步，以及有关的动作，转换，和条件。

H

主机 （Host） 网络上的一个节点。

集线器 连接一系列柔性的中央模块以构建网络的设备。

I

初始状态 当 TwidoSoft 开始或还未打开应用程序时 TwidoSoft 的工作状态，显示在状态栏。

初始化 一个命令，能将全部数据设定为初始值。控制器必须在停止或错误模式。

序号 在程序中隶属特定功能块的一个唯一对象，例如，在定时器 %TMi 中， i. 就代表其序号。

指令表语言 应用指令表语言 （IL）写成的程序包含一系列指令，并由控制器顺序执行。每个指令由行号，指令码，和操作数组成。

Internet 基于 TCP/IP 的全球计算机互联网络。

IP Internet 协议，一个通用的网络层协议。通用网络层协议 IP 通常总是和 TCP 同时使用的。

IP 地址 因特网协议地址。使用 TCP/IP 分配给主机一个 32- 位地址。

L

梯形图语言编辑器 TwidoSoft 专用窗口，用于编辑一个梯形图程序。

梯形图语言 用梯形图语言编写的程序，由代表控制器程序指令的图形的一系列梯级组成，并被顺序执行，这些图形指令包括触点，线圈和模块符号。

梯形图中的指令表梯级	在梯形图中显示的指令表程序，不能被转化为梯形图部分。
输入锁存	脉冲输入被捕获且被记录，以便应用程序以后检查。
LIFO	后入先出，功能块用作堆栈。用于堆栈操作的功能块。
指令表编辑器	简单的程序编辑器，用于创建和编辑指令表程序。

M

MAC 地址	访问控制地址。一个设备的硬件地址。在工厂里一个 MAC 地址被分配给一个以太网 TCP/IP 模块。
主控制器	远程连接网络中配置成主机的 Twido 控制器。
MBAP	Modbus Application Protocol （Modbus 应用协议）。
存储卡	可选内存备份卡，可以用来备份和装载一个应用程序 （程序和配置数据）。存在两种容量：32 和 64 Kb。
内存使用指示器	一种棒图，在 TwidoSoft 主窗口中显示应用程序占用控制器所有内存的比例，当内存不够时提供警告。
Modbus	一个主 - 从通信协议，允许一个主机请求多个从机响应。
模块型控制器	Twido 控制器的一种型号，提供灵活的扩展配置。一体型控制器是其另一型号。
监控状态	当一个 PC 以非写模式连到一个控制器时 TwidoSoft 的工作状态，显示在状态栏。

N

网络	共享共同的数据路径和通信协议的互连设备。

节点	通讯网络中可以编设备。

O

离线操作	当 PC 机未与控制器相连，或当 PC 机内存中的应用程序与控制器内存中的不同时，TwidoSoft 的一种工作模式。可以在离线模式下开发一个应用程序。
离线状态	当 PC 与控制器断开时， TwidoSoft 的操作状态，显示在状态栏中。
在线操作	当 PC 与控制器相连且 PC 中的应用程序与控制器中的相同时， TwidoSoft 的一种工作模式。在线操作模式可以用来调试应用程序。
在线状态	当一个 PC 连接到一个控制器时 TwidoSoft 的工作状态，显示在状态栏。
操作数	程序可以在指令中用来计算并代表一定值的一个数，地址，或变量名。
工作状态	指示 TwidoSoft 的状态，显示在状态栏。有四个工作状态：初始化，离线，在线，和监视。
运算符	指令中表示运算类型的符号或代码。

P

数据包 （Packet）	通过网络发送的数据单元。
PC	个人计算机。
对等控制器	在远程网络连接中配置为从站的控制器。应用程序可在对等控制器存储器中执行且程序可以访问本地和扩展 I/O 数据，但是 I/O 数据不能直接传递给主控制器。运行在对等控制器的程序通过使用网络字 （%INW 和 %QNW）传送信息给主控制器。

控制器	Twido 可编程序控制器。有两类控制器：一体型和模块型。
脉冲发生器（PLS）	脉冲发生器。该功能块用来产生占空比为 50% 的方波。
参数选择	一个对话框，通过可选项设置指令表和梯形图程序编辑器。
程序错误浏览器	TwidoSoft 专用窗口，用于查看程序错误和警告。
可编程控制器	Twido 控制器，有两类控制器：一体型和模块型。
保护	提供两种应用程序的保护方式：密码保护，它可以控制存取；控制器应用保护，可以防止对应用程序的读、写操作。
协议	用以说明信息格式，并为两台或多台设备间通讯设定规则。
PWM	脉冲宽度调制。一个功能模块，生成一个矩形波，其占空比可通过程序设置来改变。

R

RAM	随机存取存储器 Twido 应用程序被下载到内部 RAM 中并运行。
实时时钟	可选件，即使控制器一定时间内没有上电也能保持时间。
映像输出	在计数模式，超高速计数器的当前值 （%VFC.V）通过与配置阈值比较以决定这些专用输出的状态。
寄存器	控制器内部专用寄存器，专用于 LIFO/FIFO 功能模块。
远程控制器	在远程连接网络中，一个 Twido 控制器配置为与一个主控制器通讯。

远程连接	一种高速主从总线，用于小数据量通讯，可以配置为一个主站与最多 7 个远程站（从站）通讯。可配置两种远程控制器传送数据到主控制器：对等控制器可以传送应用程序数据，远程 I/O 控制器则只能传送 I/O 数据。一个远程连接网络可能包括两种类型的组合。
资源管理器	TwidoSoft 的一个部件，通过跟踪应用程序编写及配置过程中相关对象的引用，监视内存使用情况。一个对象在应用程序中如果在指令表或梯级指令中被用作一个操作数，那即视为被引用。它能显示总的存储器被使用的百分比，以及在存储容量不足时提供警告。见内存使用指示。
可逆指令	一个编程方法，允许指令可被交替查看，或者是列表指令，或者是梯形图。
路由器（Router）	一种连接两个或更多网段的设备，并允许信息在它们中间流通。每个路由器会检测它收到的每个数据包，并决定是否冻结或发送它。路由器总是力求按效率最高的路径发送数据包。
RTC	见实时时钟。
RTU	远程终端单元。一种使用 8 个位的协议，用于在控制器和 PC 间通信。
运行	一个命令，使控制器运行一个应用程序。
梯级	一个梯级位于两个隐藏条之间，包含一组竖向、横向相互连接的梯形图元件。一个梯级最大的尺寸为 7 行 11 列。
梯级标题	位于梯级上方，用于说明梯级的目的。

S

扫描	控制器扫描一个程序并执行三个基本功能。首先，它读取输入并将这些值放到存储器；其次，它一次执行应用程序的一条指令并存储结果到存储器；最后，将结果刷新到输出。

扫描模式	指定控制器如何扫描一个程序，有两种扫描模式：常规 （循环），控制器连续扫描：周期扫描，控制器的扫描根据在配置中定义好的时间 （2-150 ms）周期性地执行。
调度模块	通过编辑日期及时间功能来控制事件的功能块，需要实时时钟选件支持。
服务器	为客户端提供服务的计算机过程。这个术语也可以指服务所基于的计算机过程。
步骤	Grafcet 步指定了顺序操作的状态。
停止	一个命令，使控制器停止运行一个应用程序。
子网 （Subnet）	在 IP 网络中的一个物理上的逻辑网络，它与该网络中的其它部分共享一个网络地址。
子网掩码	位掩码，用来确定 IP 地址中的哪些位对应于网络地址，哪些位对应于地址的子网部分。子网掩码是网络地址加上确定子网的保留位。
交换机	一个连接两个或多个分离的网络部件并允许其互相通信的网络设备。转换器根据帧的目的地址决定应将帧拦截或发出。
符号	符号是由最多 32 个字符组成的字符串，它们的第一个字符必须是字母。它能使应用程序个性化，以更具有可读性。
符号表	符号表应用于应用程序中，在符号编辑器中显示。

T

TCP	传输控制协议。
TCP/IP	一个包含传输控制协议和因特网协议的协议集合；英特网即建立在本通信协议集合之上。
阀值输出	一种输出线圈，由超高速计数器 （%VFC）在配置表中设置并直接控制。

定时器	一个功能模块，用于选择一个时间周期控制事件。
Twido	Schneider Electric 控制器系列，包含两类控制器 （一体型和模块型），通过扩展模块增加 I/O 点，还有一些选件如实时时钟，通信，操作显示，和备份存储卡。
TwidoSoft	基于 Windows 的 32 位图形开发软件，用它来配置 Twido 控制器并为其编程。

U

UDP	一个通信协议 （用户数据报协议），TCP/IP 协议组的一部分，用于应用程序传输数据包。 UDP 也是 TCP/IP 协议中负责端口地址的部分。
未定义符号	符号没有变量地址。

V

变量	存储器单元，可由程序寻址和修改。
超高速计数器	一个比计数器和高速计数器更快计数的功能模块，提供比常规计数器及高速计数器更高的计数频率，超高速计数器计数频率最高可达 20 KHz。

W

热重启	没有更改应用程序时断电后再上电即执行热启动。控制器恢复到断电前的状态并完成处理中的扫描。所有应用程序数据被保存。

索引

符号

A

B

C

D

E

F

G

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

X

施耐德电气公司
Schneider Electric China
www.schneider-electric.com.cn

北京市朝阳区将台路 2 号
和乔丽晶中心施耐德大厦
邮编：100016
电话：(010) 8434 6699
传真：(010) 8450 1130

Schneider Building, Chateau Regency,
No.2 Jiangtai Road, Chaoyang District
Beijing 100016 China
Tel：(010) 8434 6699
Fax: (010) 8450 1130

SC DOC 591-LEC

2005.11