SIEMENS

SIMATIC

S7
S7-1200 可编程控制器

系统手册

V4.4 11/2019
A5E02486685-AN

法律资讯

警告提示系统

为了您的人身安全以及避免财产损失，必须注意本手册中的提示。人身安全的提示用一个警告三角表示，仅与财产损失有关的提示不带警告三角。警告提示根据危险等级由高到低如下表示。

⚠ 危险
表示如果不采取相应的小心措施，**将会**导致死亡或者严重的人身伤害。

⚠ 警告
表示如果不采取相应的小心措施，**可能**导致死亡或者严重的人身伤害。

⚠ 小心
表示如果不采取相应的小心措施，可能导致轻微的人身伤害。

注意
表示如果不采取相应的小心措施，可能导致财产损失。

当出现多个危险等级的情况下，每次总是使用最高等级的警告提示。如果在某个警告提示中带有警告可能导致人身伤害的警告三角，则可能在该警告提示中另外还附带有可能导致财产损失的警告。

合格的专业人员

本文件所属的产品/系统只允许由符合各项工作要求的**合格人员**进行操作。其操作必须遵照各自附带的文件说明，特别是其中的安全及警告提示。 由于具备相关培训及经验，合格人员可以察觉本产品/系统的风险，并避免可能的危险。

按规定使用 Siemens 产品

请注意下列说明：

⚠ 警告
Siemens 产品只允许用于目录和相关技术文件中规定的使用情况。如果要使用其他公司的产品和组件，必须得到 Siemens 推荐和允许。正确的运输、储存、组装、装配、安装、调试、操作和维护是产品安全、正常运行的前提。必须保证允许的环境条件。必须注意相关文件中的提示。

商标

所有带有标记符号 ® 的都是 Siemens AG 的注册商标。本印刷品中的其他符号可能是一些其他商标。若第三方出于自身目的使用这些商标，将侵害其所有者的权利。

责任免除

我们已对印刷品中所述内容与硬件和软件的一致性作过检查。然而不排除存在偏差的可能性，因此我们不保证印刷品中所述内容与硬件和软件完全一致。印刷品中的数据都按规定经过检测，必要的修正值包含在下一版本中。

Siemens AG
Digital Industries
Postfach 48 48
90026 NÜRNBERG
德国

A5E02486685-AN
℗ 10/2019 本公司保留更改的权利

前言

手册用途

S7-1200 系列是一款可编程逻辑控制器 (PLC, Programmable Logic Controller)，可以控制各种自动化应用。S7-1200 设计紧凑、成本低廉且具有功能强大的指令集，这些特点使它成为控制各种应用的完美解决方案。S7-1200 型号和基于 Windows 的 STEP 7 编程工具 (页 39)提供了解决自动化问题时需要的灵活性。

本手册提供了有关 S7-1200 PLC 的安装和编程信息，其主要用户是具备可编程逻辑控制器基本知识的工程师、编程人员、安装人员和电工人员。

所需的基本知识

要理解本手册，需要具备自动化和可编程逻辑控制器的基本知识。

手册适用范围

本手册介绍了以下产品：

- STEP 7 Basic 和 Professional (页 39)
- S7-1200 CPU 固件版本 V4.4

有关本手册中所述 S7-1200 产品的完整列表，请参见技术规范 (页 1427)。

证书、CE 标签、C 标记和其它认证

请参见技术规范 (页 1427)以获取更多信息。

服务与支持

除了文档之外，西门子公司还在 Internet 和客户支持网站 (https://support.industry.siemens.com/cs/cn/zh)上提供了专业技术知识。

如需要回答任何技术问题、培训或订购 S7 产品方面的帮助，请与西门子经销商或销售部联系。 因为西门子销售代表都经过技术培训并掌握有关操作、过程和工业以及有关您使用的各种西门子产品的最具体的知识，所以他们能够最快最高效地回答您可能遇到的任何问题。

文档和信息

S7-1200 和 STEP 7 提供了各种文档和其它资源，供您查找所需的技术信息。

- S7-1200 可编程控制器系统手册提供有关整个 S7-1200 产品系列的操作、编程和规范的特定信息。
  系统手册有电子版 (PDF)。可通过西门子工业在线支持网站 (https://support.industry.siemens.com/cs/cn/zh)下载或浏览本电子手册和其它电子手册。每个 S7-1200 CPU 随附的文档磁盘中也提供了系统手册。
- 通过 STEP 7 的在线信息系统，可以直接访问概念性信息和具体说明，它们介绍了编程数据包的操作和功能以及 SIMATIC CPU 的基本操作。
- 西门子工业在线支持网站 (https://support.industry.siemens.com/cs/cn/zh)可用于访问电子版 (PDF) 的 SIMATIC 文档集，包括系统手册和 STEP 7 信息系统。现有文档可通过产品支持链接下载。借助此在线文档，您可以通过拖放不同文档中的主题来创建自己的自定义手册。西门子工业在线支持网站还提供以前发布的系统手册的更新版。
  您可以单击页面左侧的"mySupport"并从导航选项中选择“文档”(Documentation)，从而访问在线文档。要使用 mySupport 文档功能，您必须注册为正式用户。
- Siemens 工业在线支持网站还提供常见问题解答和有助于使用 S7-1200 和 STEP 7 的其它文档。
- 您还可以关注或加入服务与支持技术论坛 (https://support.industry.siemens.com/tf/ww/en/?Language=en&siteid=csius&treeLang=en&groupid=4000002&extranet=standard&viewreg=WW&nodeid0=34612486)关于产品的讨论。通过论坛，您可以与各领域的产品专家互动。
  - S7-1200 (https://support.industry.siemens.com/tf/ww/en/threads/237?title=simatic-s7-1200&skip=0&take=10&orderBy=LastPostDate+desc) 论坛
  - STEP 7 Basic (https://support.industry.siemens.com/tf/ww/en/threads/243?title=step-7-tia-portal&skip=0&take=10&orderBy=LastPostDate+desc) 论坛

安全性信息

Siemens 为其产品及解决方案提供了工业信息安全功能，以支持工厂、系统、机器和网络的安全运行。

为了防止工厂、系统、机器和网络受到网络攻击，需要实施并持续维护先进且全面的工业信息安全保护机制。Siemens 的产品和解决方案构成此类概念的其中一个要素。

客户负责防止其工厂、系统、机器和网络受到未经授权的访问。只有在有必要连接时并仅在采取适当安全措施（例如，防火墙和/或网络分段）的情况下，才能将该等系统、机器和组件连接到企业网络或 Internet。

关于可采取的工业信息安全措施的更多信息，请访问 (https://www.siemens.com/industrialsecurity)。

Siemens 不断对产品和解决方案进行开发和完善以提高安全性。Siemens 强烈建议您及时更新产品并始终使用最新产品版本。如果使用的产品版本不再受支持，或者未能应用最新的更新程序，客户遭受网络攻击的风险会增加。

要及时了解有关产品更新的信息，请订阅 Siemens 工业信息安全 RSS 源，网址为 (https://www.siemens.com/industrialsecurity)。

目录

产品概述 1

1.1 S7-1200 PLC 简介

S7-1200 控制器使用灵活、功能强大，可用于控制各种各样的设备以满足您的自动化需求。S7-1200 结构紧凑、组态灵活且具有功能强大的指令集，这些特点的组合使它成为控制各种应用的完美解决方案。

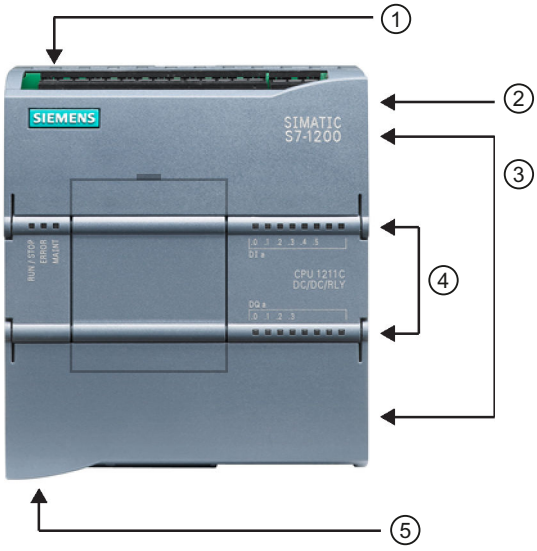
CPU 将以下元素和更多元素结合在一个紧凑的外壳中，创造出一款功能强大的控制器：

- 微处理器
- 集成的电源
- 输入和输出电路
- 内置 PROFINET
- 高速运动控制 I/O

在您下载用户程序后，CPU 将包含监控应用中的设备所需的逻辑。CPU 根据用户程序逻辑监视输入并更改输出，用户程序可以包含布尔逻辑、计数、定时、复杂数学运算、运动控制以及与其它智能设备的通信。

CPU 提供一个 PROFINET 端口用于通过 PROFINET 网络通信。还可使用附加模块基于如下网络和协议进行通信：

- PROFIBUS
- GPRS
- LTE
- 具有安全集成功能（防火墙、VPN）的 WAN
- RS485
- RS232
- RS422
- IEC 60870
- DNP3
- USS
- MODBUS



① 电源接口

② 存储卡插槽（上部保护盖下面）

③ 可拆卸用户接线连接器（保护盖下面）

④ 板载 I/O 的状态 LED

⑤ PROFINET 连接器（CPU 的底部）

有多种安全功能可用于保护对 CPU 和控制程序的访问：

- 每个 CPU 都提供密码保护 (页 198)功能，用户可以通过该功能组态对 CPU 功能的访问权限。
- 可以使用“专有技术保护” (页 201)隐藏特定块中的代码。
- 可以使用复制保护 (页 202)将程序绑定到特定存储卡或 CPU。

表格 1-1　　CPU 型号的比较

特征		CPU 1211C	CPU 1212C	CPU 1214C	CPU 1215C	CPU 1217C
物理尺寸 (mm)		90 x 100 x 75		110 x 100 x 75	130 x 100 x 75	150 x 100 x 75
用户存储器	工作	50 KB	75 KB	100 KB	125 KB	150 KB
	负载	1 MB	2 MB	4 MB		
	保持性	10 KB				
本地板载 I/O	数字量	6 个输入/ 4 个输出	8 个输入/ 6 个输出	14 个输入/ 10 个输出		
	模拟量	2 个输入			2 个输入/2 个输出	
过程映像大小	输入 (I)	1024 个字节				
	输出 (Q)	1024 个字节				
位存储器 (M)		4096 个字节		8192 个字节		
信号模块 (SM) 扩展		无	2	8		

特征		CPU 1211C	CPU 1212C	CPU 1214C	CPU 1215C	CPU 1217C
信号板 (SB)、电池板 (BB) 或通信板 (CB)		1				
通信模块 (CM) （左侧扩展）		3				
高速计数器	总计	最多可组态 6 个使用任意内置或 SB 输入的高速计数器				
	1 MHz	-				Ib.2 到 Ib.5
	100/[1]80 kHz	Ia.0 到 Ia.5				
	30/[1]20 kHz	--	Ia.6 到 Ia.7	Ia.6 到 Ib.5		Ia.6 到 Ib.1
	200 kHz[3]					
脉冲输出 [2]	总计	最多可组态 4 个使用任意内置或 SB 输出的脉冲输出				
	1 MHz	--				Qa.0 到 Qa.3
	100 kHz	Qa.0 到 Qa.3				Qa.4 到 Qb.1
	20 kHz	--	Qa.4 到 Qa.5	Qa.4 到 Qb.1		--
存储卡		SIMATIC 存储卡（选件）				
数据日志	数量	每次最多打开 8 个				
	大小	每个数据日志为 500 MB 或受最大可用装载存储器容量限制				
实时时钟保持时间		通常为 20 天，40°C 时最少为 12 天（免维护超级电容）				
PROFINET 以太网通信端口		1			2	
实数数学运算执行速度		2.3 μs/指令				
布尔运算执行速度		0.08 μs/指令				

[1] 将 HSC 组态为正交工作模式时，可应用较慢的速度。

[2] 对于具有继电器输出的 CPU 模块，必须安装数字量信号 (SB) 才能使用脉冲输出。

[3] 与 SB 1221 DI x 24 V DC 200 kHz 和 SB 1221 DI 4 x 5 V DC 200 kHz 一起使用时最高可达 200 kHz。

不同的 CPU 型号提供了各种各样的特征和功能，这些特征和功能可帮助用户针对不同的应用创建有效的解决方案。有关特定 CPU 的详细信息，请参见技术规范 (页 1427)。

表格 1-2　S7-1200 支持的块、定时器和计数器

元素			说明				
块	类型		OB、FB、FC、DB				
	大小	CPU 型号	CPU 1211C	CPU 1212C	CPU 1214C	CPU 1215 C	CPU 1217 C
		代码块	50 KB	64 KB	64 KB	64 KB	64 KB
		已链接 [1] 数据块	50 KB	75 KB	100 KB	125 KB	150 KB
		未链接 [2] 数据块	256 KB	256 KB	256 KB	256 KB	256 KB
	数量		最多可达 1024 个块 (OB + FB + FC + DB)				
	嵌套深度		16（从程序循环 OB 或启动 OB 开始）； 6（从任意中断事件 OB 开始）[3]				
	监视		可以同时监视 2 个代码块的状态				
OB	程序循环		多个				
	启动		多个				
	延时中断		4（每个事件 1 个）				
	循环中断		4（每个事件 1 个）				
	硬件中断		50（每个事件 1 个）				
	时间错误中断		1				
	诊断错误中断		1				
	拔出或插入模块		1				
	机架或站故障		1				
	日时钟		多个				
	状态		1				
	更新		1				
	配置文件		1				
定时器	类型		IEC				
	数量		仅受存储器大小限制				
	存储		DB 结构，每个定时器 16 个字节				

元素		说明
计数器	类型	IEC
	数量	仅受存储器大小限制
	存储	DB 结构，大小取决于计数类型 • SInt 和 USInt：3 个字节 • Int 和 UInt：6 个字节 • DInt 和 UDInt：12 个字节

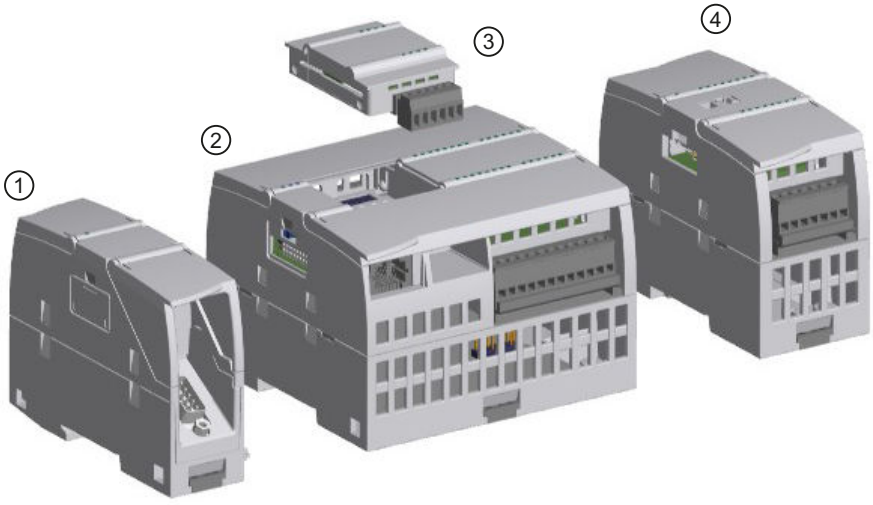
[1] 存储在工作存储器和装载存储器中。不能超过工作或装载存储器的剩余大小。

[2] 仅存储在装载存储器中

[3] 安全程序使用二级嵌套。因此，用户程序在安全程序中的嵌套深度为四。

1.2 CPU 的扩展功能

S7-1200 系列提供了各种模块和插入式板，用于通过附加 I/O 或其它通信协议来扩展 CPU 的功能。有关特定模块的详细信息，请参见技术规范 (页 1427)。



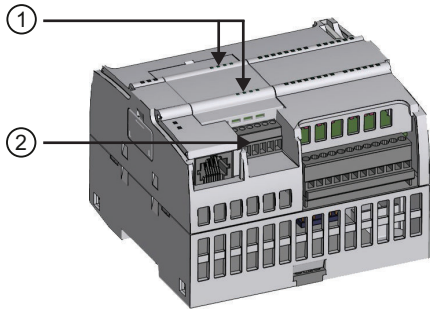
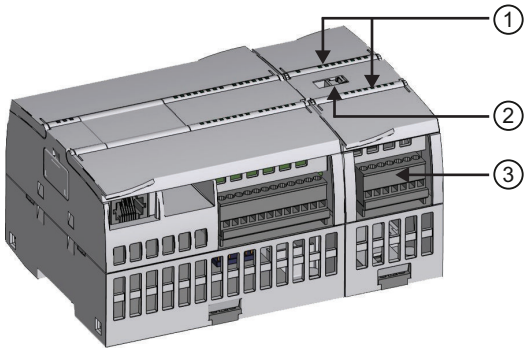
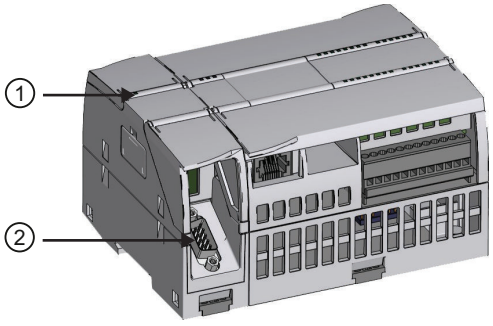
① 通信模块 (CM) 或通信处理器 (CP) (页 1622)

② CPU（CPU 1211C (页 1440)、CPU 1212C (页 1455)、CPU 1214C (页 1470)、CPU 1215C (页 1486)、CPU 1217C (页 1503)）

③ 信号板 (SB)（数字 SB (页 1588)、模拟 SB (页 1601)），通信板 (CB) (页 1633) 或电池板 (BB) CPU（CPU 1211C、CPU 1212C、CPU 1214C、CPU 1215C、CPU 1217C） (页 1620)

④ 信号模块 (SM)（数字 SM (页 1522)、模拟 SM (页 1544)、热电偶 SM (页 1559)、RTD SM (页 1566)、工艺 SM） (页 1573)

表格 1-3　S7-1200 扩展模块

模块类型	说明
CPU 支持一个插入式扩展板： ● 信号板 (SB) 可为 CPU 提供附加 I/O。SB 连接在 CPU 的前端。 ● 通信板 (CB) 可以为 CPU 增加其它通信端口。 ● 电池板 (BB) 可提供长期的实时时钟备份。	 ① SB 上的状态 LED ② 可拆卸用户接线连接器
信号模块 (SM) 可以为 CPU 增加其它功能。SM 连接在 CPU 右侧。 ● 数字量 I/O ● 模拟量 I/O ● RTD 和热电偶 ● SM 1278 IO-Link 主站 ● SM 1238 电能表 (https://support.industry.siemens.com/cs/ww/en/view/109483435)	 ① 状态 LED ② 总线连接器滑动接头 ③ 可拆卸用户接线连接器
通信模块 (CM) 和通信处理器 (CP) 将增加 CPU 的通信选项，例如 PROFIBUS 或 RS232/RS485 的连接性（适用于 PtP、Modbus 或 USS）或者 AS-i 主站。 CP 可以提供其它通信类型的功能，例如通过 GPRS、LTE、IEC、DNP3 或 WDC 网络连接到 CPU。 ● CPU 最多支持三个 CM 或 CP ● 各 CM 或 CP 连接在 CPU 的左侧（或连接到另一 CM 或 CP 的左侧）	 ① 状态 LED ② 通信连接器

1.3 HMI 基本型面板

SIMATIC HMI 基本型面板提供了触屏式设备，用于执行基本的操作员监控任务。所有面板的保护等级均为 IP65 并通过了 CE、UL、cULus 和 NEMA 4x 认证。

可用的基本型 HMI 面板 (页 1657)如下所述：

- KTP400 Basic：4" 触摸屏，带 4 个可组态按键，分辨率为 480 x 272，800 个变量
- KTP700 Basic：7" 触摸屏，带 8 个可组态按键，分辨率为 800 x 480，800 个变量
- KTP700 Basic DP：7" 触摸屏，带 8 个可组态按键，分辨率为 800 x 480，800 个变量
- KTP900 Basic：9" 触摸屏，带 8 个可组态按键，分辨率为 800 x 480，800 个变量
- KTP1200 Basic：12" 触摸屏，带 10 个可组态按键，分辨率为 800 x 480，800 个变量
- KTP 1200 Basic DP：12" 触摸屏，带 10 个可组态按键，分辨率为 800 x 400，800 个变量

新功能 2

以下为 V4.4 版本的新增功能：

- OPC UA 服务器 (页 1018) - S7-1200 CPU 支持一部分可能的 OPC UA 功能
- 开放式用户通信 (OUC) (页 828) 更新：
  - 支持通过 TMAIL_C 发送带有用户文件附件（配方和数据日志）的电子邮件
  - 通过 TMAIL_C 进行 DNS 名称解析
  - 支持“InterfaceId 为 0”，此时 CPU 会选择适当的 CPU 接口。（不包含 CP 模块）
  - 支持 DNS 名称解析以进行 TCP/UDP 通信
- 更新了指令：
  - SCATTER、SCATTER_BLK、GATHER 和 GATHER_BLK (页 273)
- 运动控制：MC_Reset（确认错误）指令可在用户程序启用轴之前确认排队的错误。
- Web 服务器
  - 连续协调 S7-1200 和 S7-1500 之间的标准网站
  - 支持已组态 PROFINET IO 设备或模块的固件更新
  - 数据日志下载/清除 Log_Download_Clear 可用于：
    - 查看 PLC 上所有数据日志的列表
    - 将数据日志从 PLC 下载到计算机
    - 从 PLC 中删除数据日志
    - 检索并清除 PLC 中的数据日志
  - 用户文件浏览器
- 两个新的信号数字模块提供漏型输出（输出负载的 M 开关）：
  - SM 1223 DI 16 x 24 V DC、DQ 16 x 24 V DC 漏型 (6ES7223-1BL32-1XB0) (页 1532)
  - SM 1222 DQ 16 x 24 V DC 漏型 (6ES7222-1BH32-1BX0) (页 1526)

用 V4.x.x CPU 更换 V3.0 CPU

将 S7-1200 V3.0 CPU 更换为 S7-1200 V4.x.x CPU 时，请注意两个版本间记录的差异 (页 1667)和所需的用户操作。

参见

PROFINET 连接的参数 (页 833)

S7-1200 功能安全手册 (https://support.industry.siemens.com/cs/cn/zh/view/104547552/en)

STEP 7 编程软件 3

STEP 7 软件提供了一个用户友好的环境，供用户开发、编辑和监视控制应用所需的逻辑，其中包括用于管理和组态项目中所有设备（例如控制器和 HMI 等设备）的工具。为了帮助用户查找需要的信息，STEP 7 提供了内容丰富的在线帮助系统。

STEP 7 提供了标准编程语言，用于方便高效地开发适合用户具体应用的控制程序。

- LAD（梯形图逻辑） (页 188)是一种图形编程语言。它使用基于电路图的表示法。
- FBD（函数块图） (页 189)是基于布尔代数中使用的图形逻辑符号的编程语言。
- SCL（结构化控制语言） (页 189)是一种基于文本的高级编程语言。

创建代码块时，应选择该块要使用的编程语言。用户程序可以使用由任意或所有编程语言创建的代码块。

说明

STEP 7 是 TIA Portal 中的编程和组态软件。除了包括 STEP 7 外，TIA Portal 中还包括设计和执行运行过程可视化的 WinCC，以及 WinCC 以及 STEP 7 的在线帮助。

S7-1200 V4.4 中的新功能要求使用 STEP 7 Professional V16。

3.1 系统要求

必须使用管理员权限来安装 STEP 7。

表格 3-1 系统要求

硬件/软件	要求
处理器类型	Intel® Core™ i3-6100U，2.30 GHz 或更高频率
RAM	8 GB
可用硬盘空间	系统驱动器 C:\ 上的 20 GB 空间

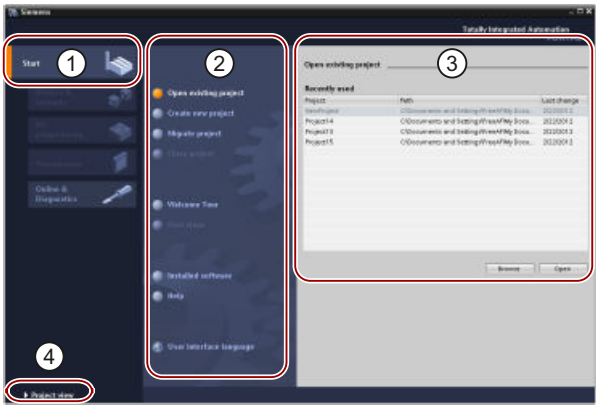
硬件/软件	要求
操作系统	STEP 7 适用于以下操作系统： ● Windows 7（64 位）： – Windows 7 Home Premium SP1 ** – Windows 7 Professional SP1 – Windows 7 Enterprise SP1 – Windows 7 Ultimate SP1 ● Windows 10（64 位）： – Windows 10 Home V1709 ** – Windows 10 Home V1803 ** – Windows 10 Professional V1709 – Windows 10 Professional V1803 – Windows 10 Enterprise V1709 – Windows 10 Enterprise V1803 – Windows 10 Enterprise 2016 LTSB – Windows 10 IoT Enterprise 2015 LTSB – Windows 10 IoT Enterprise 2016 LTSB ● Windows Server（64 位） – Windows Server 2012 R2 StdE（完全安装） – Windows Server 2016 Standard（完全安装）
图形卡	32 MB RAM 24 位颜色深度
屏幕分辨率	1024 x 768
网络	STEP 7 和 CPU 之间的通信采用 100 Mbit/s 以太网或更快网速

*包括所有适用的安全更新。有关操作系统的详细信息，请参见 Microsoft Windows 或 Microsoft 网站上的帮助。

** 仅适用于基本版

3.2 使工作更轻松的不同视图

STEP 7 提供了一个用户友好的环境，供用户开发控制器逻辑、组态 HMI 可视化和设置网络通信。为帮助用户提高生产率，STEP 7 提供了两种不同的项目视图：根据工具功能组织的面向任务的门户集（门户视图），或项目中各元素组成的面向项目的视图（项目视图）。请选择能让您的工作最高效的视图。只需通过单击就可以切换门户视图和项目视图。



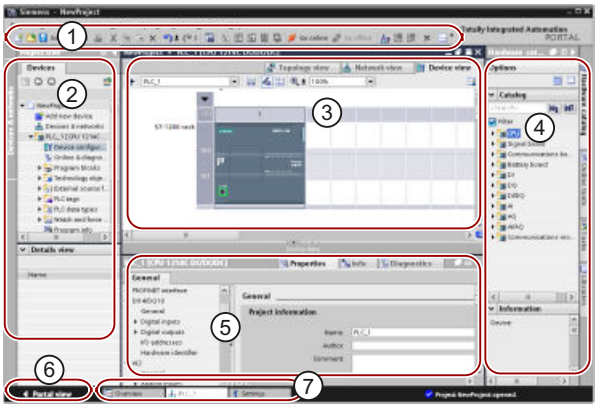
门户视图

① 不同任务的门户

② 所选门户的任务

③ 所选操作的选择面板

④ 切换到项目视图



项目视图

① 菜单和工具栏

② 项目浏览器

③ 工作区

④ 任务卡

⑤ 巡视窗口

⑥ 切换到门户视图

⑦ 编辑器栏

由于这些组件组织在一个视图中，所以您可以方便地访问项目的各个方面。工作区由三个选项卡形式的视图组成。

- 设备视图：显示已添加或已选择的设备及其相关模块
- 网络视图：显示网络中的 CPU 和网络连接
- 拓扑视图：显示网络的 PROFINET 拓扑，包括设备、无源组件、端口、互连及端口诊断

每个视图还可用于执行组态任务。巡视窗口显示用户在工作区中所选对象的属性和信息。当用户选择不同的对象时，巡视窗口会显示用户可组态的属性。巡视窗口包含用户可用于查看诊断信息和其它消息的选项卡。

编辑器栏会显示所有打开的编辑器，从而帮助用户更快速和高效地工作。要在打开的编辑器之间切换，只需单击不同的编辑器。还可以将两个编辑器垂直或水平排列在一起显示。通过该功能可以在编辑器之间进行拖放操作。

STEP 7 信息系统对 STEP 7 的所有组态、编程和监视工具都提供了内容丰富的在线帮助。对于本手册未包括的详细说明，您可以参考此系统。

3.3 易于使用的工具

3.3.1 将指令插入用户程序中

STEP 7 提供了包含各种程序指令的任务卡。 这些指令按功能分组。

要创建程序，可将指令从任务卡拖动到程序段中。

3.3.2 从“收藏夹”工具栏调用指令

STEP 7 提供了“收藏夹”(Favorites) 工具栏，可供用户快速访问常用的指令。 只需单击指令的图标即可将其插入程序段！

（要访问指令树中的“收藏夹”，请双击该图标。）

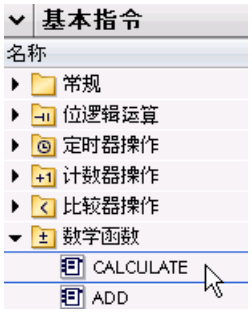
用户可以通过添加新指令方便地自定义“收藏夹”(Favorites)。

只需将指令拖放到“收藏夹”(Favorites)。

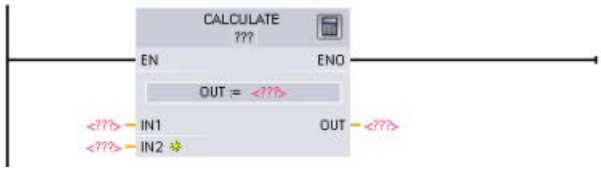
如此单击即可插入该指令！

3.3.3 使用简单指令创建复杂等式

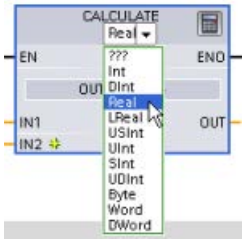
Calculate 指令 (页 240)可以根据定义的等式生成作用于多个输入参数的数学函数，从而生成结果。



在 Basic 指令树中，展开“数学函数”(Math functions) 文件夹。 双击 Calculate 指令以将该指令插入用户程序中。



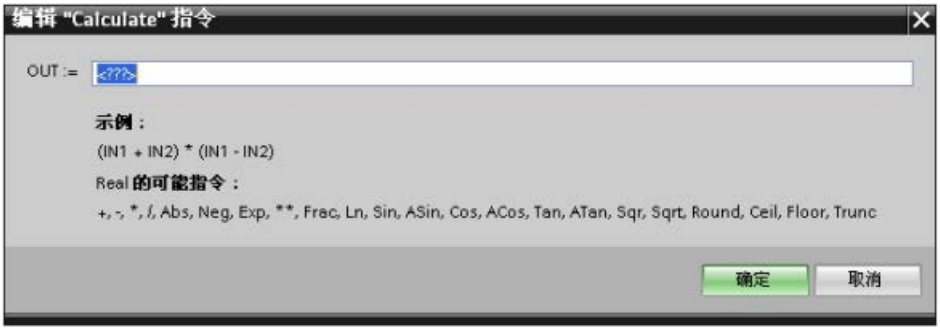
未组态的 Calculate 指令提供了两个输入参数和一个输出参数。



单击“???”并为输入参数和输出参数选择数据类型。（所有输入参数和输出参数的数据类型必须相同。）

对于本示例，请选择“Real”数据类型。

单击“编辑等式”(Edit equation) 图标以输入等式。



对于本示例，请输入以下等式来标定原有模拟值。（“In”和“Out”标识对应于 Calculate 指令的参数。）

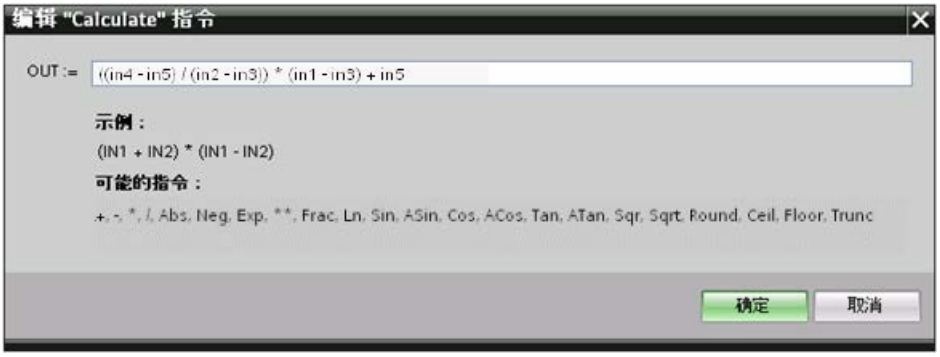
$Out_{value} = ((Out_{high} - Out_{low}) / (In_{high} - In_{low})) * (In_{value} - In_{low}) + Out_{low}$

Out = ((in4 - in5) / (in2 - in3)) * (in1 - in3) + in5

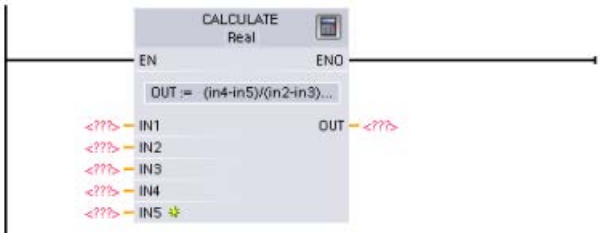
其中：	Out_{value}	(Out)	标定的输出值
	In_{value}	(in1)	模拟量输入值
	In_{high}	(in2)	标定输入值的上限
	In_{low}	(in3)	标定输入值的下限
	Out_{high}	(in4)	标定输出值的上限
	Out_{low}	(in5)	标定输出值的下限

在“编辑 Calculate”(Edit Calculate) 框中，输入带有参数名称的等式：

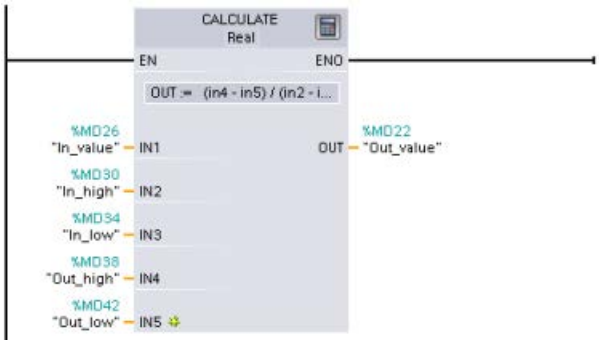
OUT = ((in4 - in5) / (in2 - in3)) * (in1 - in3) + in5



单击“确定”(OK) 后，Calculate 指令就会生成指令所需的输入。



输入与参数对应的值的变量名称。



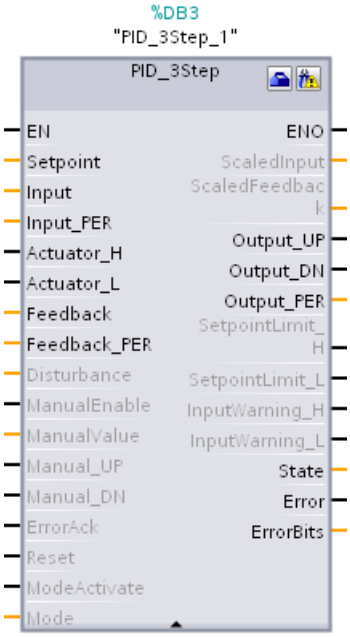
3.3.4 向 LAD 或 FBD 指令添加输入或输出

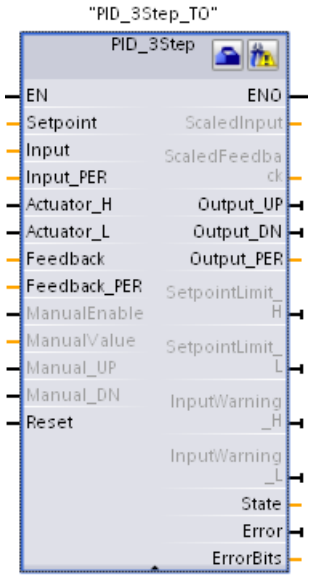
有些指令允许您另外创建输入或输出。

- 要添加输入或输出，请单击“创建”(Create) 图标，或在其中一个现有 IN 或 OUT 参数的输入短线处单击右键，并选择“插入输入”(Insert input) 命令。
- 要删除输入或输出，请在其中一个现有 IN 参数或 OUT 参数（原始输入多于两个时）的短线处单击右键，然后选择“删除”(Delete) 命令。

3.3.5 可扩展指令

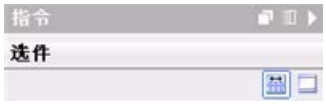
一些更为复杂的指令是可扩展的，只显示主要输入和输出。 要显示所有输入和输出，请单击指令底部的箭头。



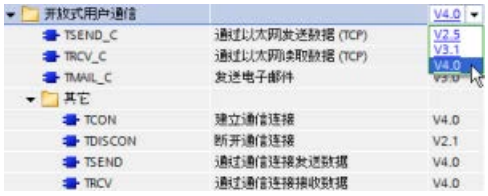


3.3.6 选择指令的版本

某些指令集（如 Modbus、PID 和运动指令集）经过多个开发和发布周期后形成了多种发布版本。 为了有助于确保与较早项目的兼容性以及对这些项目进行移植，STEP 7 允许您选择要插入用户程序中的指令版本。



单击指令树任务卡上的图标可启用指令树的标题和列。



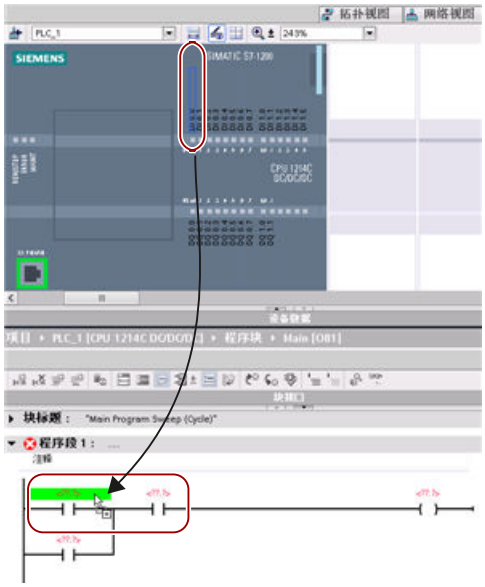
要更改指令版本，需从下拉列表中选择合适的版本。

3.3.7 修改 STEP 7 的外观和组态

用户可以选择不同的设置，例如界面的外观、语言或项目的保存目录。

在“选项”(Options) 菜单中选择“设置”(Settings) 命令更改这些设置。

3.3.8 在编辑器之间拖放



为帮助用户快速方便地执行任务，STEP 7 允许用户将元素从一个编辑器拖放到另一个编辑器中。 例如，可以将 CPU 的输入拖动到用户程序中指令的地址上。

必须放大至少 200% 才能选中 CPU 的输入或输出。

请注意，变量名称不仅会在 PLC 变量表中显示，还会在 CPU 上显示。

要一次显示两个编辑器，请使用“拆分编辑器”(Split editor) 菜单命令或工具栏中的相应按钮。

要在已打开的编辑器之间切换，请单击编辑器栏中的图标。

3.3.9 更改 CPU 的工作模式

该 CPU 没有用于更改工作模式（STOP 或 RUN）的物理开关。

请使用 “启动 CPU”(Start CPU) 和 “停止 CPU”(Stop CPU) 工具栏按钮更改 CPU 的工作模式。

在设备组态 (页 147)中组态 CPU 时，应组态 CPU 属性中的启动行为 (页 163)。

“在线和诊断”(Online and Diagnostics) 门户还提供了用于更改在线 CPU 工作模式的操作面板。要使用 CPU 操作员面板，必须在线连接到 CPU。 “在线工具”(Online tools) 任务卡显示的操作员面板显示了在线 CPU 的工作模式。也可以通过该操作员面板更改在线 CPU 的工作模式。

使用操作员面板上的按钮更改工作模式（STOP 或 RUN）。操作员面板还提供了用于复位存储器的 MRES 按钮。

RUN/STOP 指示器的颜色指示 CPU 当前的工作模式。黄色表示 STOP 模式，而绿色表示 RUN 模式。

通过 STEP 7 中的设备组态 (页 147)，还可以在 CPU 上电时组态默认运行模式 (页 84)。

说明

还可以通过 Web 服务器 (页 1025)模式或 SIMATIC 自动化工具 (https://support.industry.siemens.com/cs/cn/zh/view/98161300/en)模式更改 CPU 的操作模式。

3.3.10 更改 DB 的调用类型

STEP 7 允许您方便地创建或更改指令或 FB 的 DB 关联。

- 您可以在不同 DB 之间切换关联。
- 可以在单背景数据块与多背景数据块之间切换关联。
- 可以创建背景数据块（如果背景数据块丢失或不可用）。

可通过在程序编辑器中右键单击相关指令或 FB，或者通过选择“选项”(Options) 菜单中的“块调用”(Block call) 命令，来访问“更改调用类型”(Change call type) 命令。

通过“调用选项”(Call options) 对话框可选择单背景数据块或多背景数据块。还可以从可用 DB 的下拉列表中选择具体 DB。

3.3.11 暂时从网络中断开设备

从网络视图中，可断开各个网络设备与子网的连接。由于不会从项目中删除相关设备的组态，因此可轻松恢复与设备的连接。

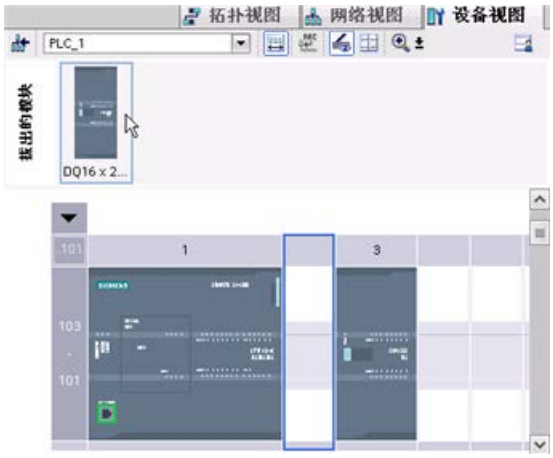
右键单击网络设备接口，然后从右键快捷菜单中选择“从子网断开”(Disconnect from subnet) 命令。

STEP 7 会重新组态网络连接，但不会从项目中删除断开的设备。删除该网络连接时，接口地址不会发生变化。

下载新的网络连接时，CPU 必须设置为 STOP 模式。

要重新连接设备，只需创建到设备端口的新网络连接。

3.3.12 从组态中虚拟拔出设备



STEP 7 为“拔出的”模块提供了一个存储区域。 用户可以从机架中拖出模块以保存该模块的组态。 这些拔出的模块会随项目一同保存，从而在将来不必重新组态参数即可再次插入相应模块。

此功能的其中一种用途是用于临时维护。 想想用户可能正等待一个替换模块，并计划临时使用一个不同的模块来短期替换相应模块。 此时，用户可以将组态的模块从机架拖动到“拔出的模块”(Unplugged modules) 区域，然后插入临时模块。

3.4 向后兼容性

STEP 7 V16 支持 S7-1200 V4.4 CPU 的组态和编程。

可以将 S7-1200 V4.x CPU 早期版本的项目从 STEP 7 V13 SP1 或更高版本下载到 S7-1200 V4.4 CPU 中。组态和编程将受到 S7-1200 CPU 早先版本以及 STEP 7 支持版本的功能及指令设置的限制。

这种向后兼容性可使用户运行在先前为旧版本设计和编程的 S7-1200 V4.4 CPU 型号上的程序。

⚠ 警告

从旧版本 STEP 7 复制粘贴程序逻辑时的风险

从旧版本 STEP 7 复制程序逻辑可能导致程序执行出现无法预测的行为或者编译失败。不同版本的 STEP 7 执行程序元素的方式不同。如果从旧版本粘贴内容到 STEP 7 V15 后进行了更改，编译器不会一直检测相关差异。如不修正程序，执行不可预测的程序逻辑可能导致严重的人员伤害甚至死亡。

使用旧版本 STEP 7 中的程序逻辑时，必须将整个项目升级到最新版本 STEP 7。随后，用户可以根据需要复制、剪切、粘贴和编辑程序逻辑。可以在 STEP 7 V16 中打开 STEP 7 V13 SP1 或更高版本的项目。随后，STEP 7 执行必要的兼容性转换并正确升级程序。为确保正确编译和执行程序，必须执行这类升级转换和修正。如果项目版本低于 STEP 7 V13 SP1，则必须将项目升级到 STEP 7 V16 (页 1667)。

无法将 V1.0、V2.0 或 V3.0 S7-1200 CPU 项目下载到 S7-1200 V4.x CPU。参见 设备更换和备件兼容性 (页 1667) 主题，了解如何将早期项目升级到可以下载的项目。

说明

S7-1200 V1.x CPU 版本的项目

在 STEP 7 V15.1 中，无法打开包含 S7-1200 V1.x CPU 的 STEP 7 项目。要使用当前项目，必须使用 STEP 7 V13 SP1（包含所有更新）打开项目并将 S7-1200 V1.x CPU 转换为 V2.0 或更高版本。之后，才可以用 STEP 7 V15.1 打开 CPU 经转换的已保存项目。

参见

新功能 (页 37)

安装 4

4.1 S7-1200 设备安装准则

S7-1200 设备设计得易于安装。可以将 S7-1200 安装在面板或标准导轨上，并且可以水平或垂直安装 S7-1200。S7-1200 尺寸较小，用户可以有效地利用空间。

电气设备标准将 SIMATIC S7-1200 系统分类为开放式设备。必须将 S7-1200 安装在外壳、控制柜或电控室内。仅限获得授权的人员能打开外壳、控制柜或进入电控室。

安装时应为 S7-1200 提供干燥的环境。可以考虑使用 SELV/PELV 电路在干燥位置处提供电击防护。

安装时应按照适用的电气和建筑规范，为特定位置类别的开放式设备提供经过批准的机械强度、可燃性保护以及稳定性防护。

由于灰尘、潮湿和大气污染引起的导电性污染会导致 PLC.中发生操作和电气故障。

如果将 PLC 放在可能存在导电性污染的区域，必须采用具有适当保护等级的外壳对 PLC 实施保护。IP54 是常用于脏乱环境中电气设备外壳的一种保护等级，可能适合您的应用环境。

⚠ 警告
S7-1200 安装不当会导致发生电气故障或出现意外的机械操作。 电气故障或意外的机械操作可能会导致死亡、人员重伤和/或财产损失。 必须遵守适当操作环境的所有安装和维护说明以确保设备安全运行。

将 S7-1200 设备与热辐射、高压和电噪声隔离开

作为布置系统中各种设备的基本规则，必须将产生高压和高电噪声的设备与 S7-1200 等低压逻辑型设备隔离开。

在面板上配置 S7-1200 的布局时，请考虑发热设备并将电子式设备布置在控制柜中较凉爽区域。少暴露在高温环境中会延长所有电子设备的使用寿命。

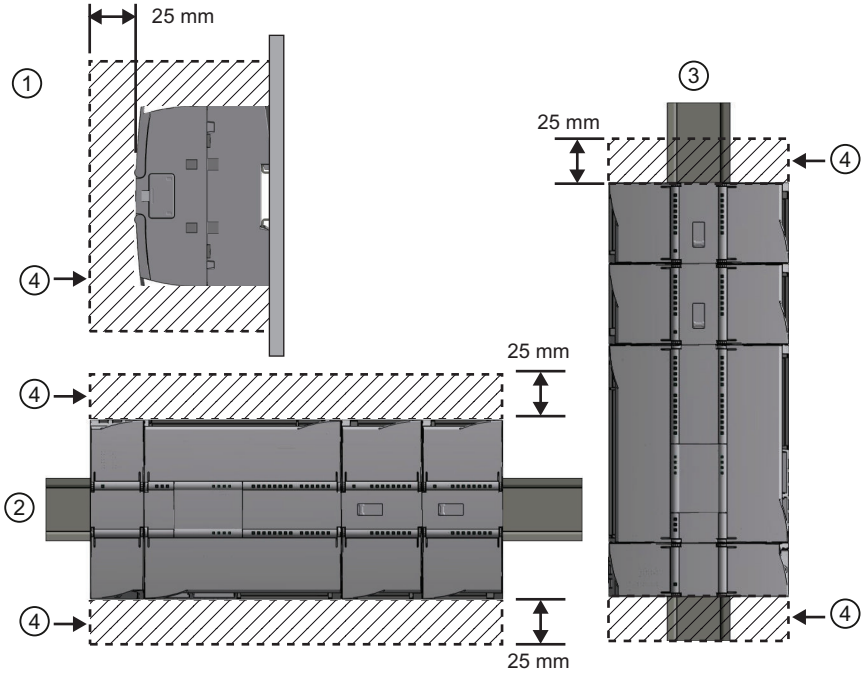
另外还要考虑面板中设备的布线。避免将低压信号线和通信电缆铺设在具有交流动力线和高能量快速开关直流线的槽中。

留出足够的空隙以便冷却和接线

S7-1200 被设计成通过自然对流冷却。为保证适当冷却，在设备上方和下方必须留出至少 25 mm 的空隙。此外，模块前端与机柜内壁间至少应留出 25 mm 的深度。

⚠ 小心
对于纵向安装，允许的最大环境温度将降低 10°C。 请按下图所示调整垂直安装的 S7-1200 系统的方位。 确保正确安装 S7-1200 系统。

规划 S7-1200 系统的布局时，应留出足够的空隙以方便接线和通信电缆连接。



① 侧视图　　③ 垂直安装

② 水平安装　　④ 空隙区域

4.2 功率预算

CPU 有一个内部电源，用于为 CPU、信号模块、信号板和通信模块供电，并可满足其它 24 V DC 用户的电源要求。

有关 CPU 所提供的 5 V DC 逻辑预算以及信号模块、信号板和通信模块的 5 V DC 功率要求的信息，请参考技术规范 (页 1427)。请参考“计算功率预算” (页 1647)来确定 CPU 可以为您的配置提供多少电能（或电流）。

CPU 提供 24 V DC 传感器电源，可以为输入点、信号模块上的继电器线圈电源或其它要求供给 24 V DC。如果您的 24 V DC 电源要求超出该传感器电源的预算，则必须给系统增加外部 24 V DC 电源。有关具体 CPU 的 24 V DC 传感器电源功率预算，请参考技术规范 (页 1427)。

如果需要外部 24 V DC 电源，请确保该电源不要与 CPU 的传感器电源并联。为提高电噪声防护能力，建议连接不同电源的公共端 (M)。

⚠ 警告

将外部 24 V DC 电源与 24 V DC 传感器电源并联会导致这两个电源之间有冲突，因为每个电源都试图建立自己首选的输出电压电平

该冲突可能使其中一个电源或两个电源的寿命缩短或立即出现故障，从而导致 PLC 系统的运行不确定。运行不确定可能导致死亡、人员重伤和/或财产损失。

DC 传感器电源和任何外部电源应分别给不同位置供电。

S7-1200 系统中的一些 24 V DC 电源输入端口是互连的，并且通过一个公共逻辑电路连接多个 M 端子。例如，在数据表中指定为“非隔离”时，以下电路是互连的：CPU 的 24 V DC 电源、SM 的继电器线圈的电源输入或非隔离模拟量输入的电源。所有非隔离的 M 端子必须连接到同一个外部参考电位。

⚠ 警告

将非隔离的 M 端子连接到不同参考电位将导致意外的电流，该电流可能导致 PLC 和任何连接设备损坏或运行不确定。

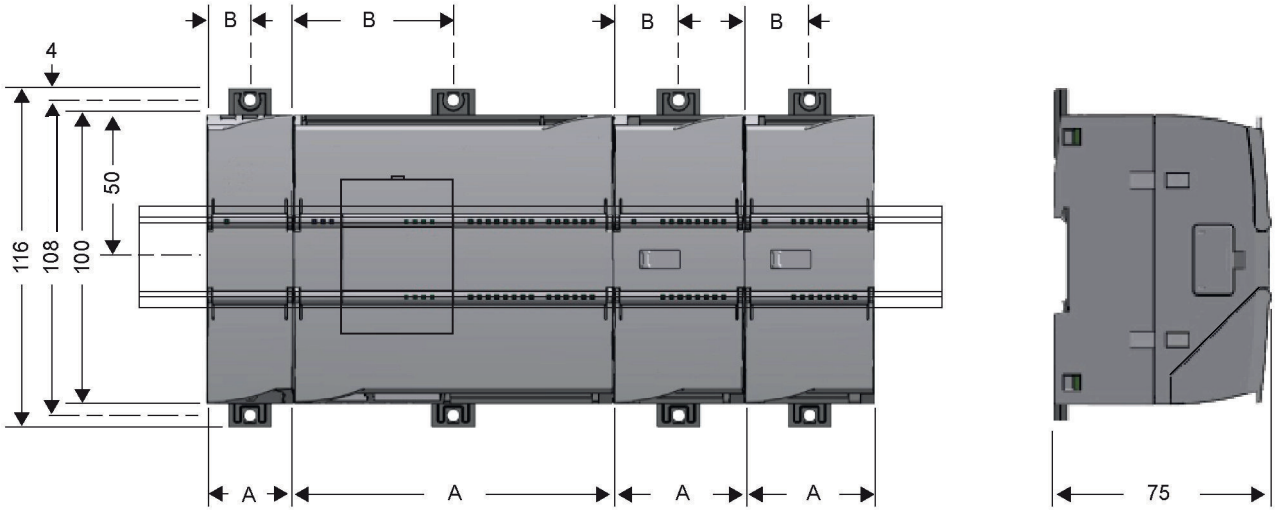
不遵守这些准则可能会导致设备损坏或运行不确定，而后者可能导致死亡、人员重伤和/或财产损失。

务必确保 S7-1200 系统中的所有非隔离 M 端子都连接到同一个参考电位。

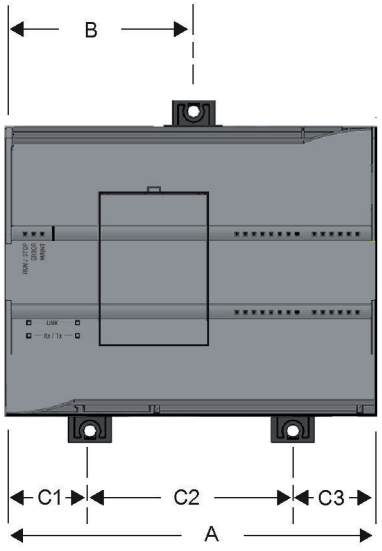
4.3 安装和拆卸步骤

4.3.1 S7-1200 设备的安装尺寸

CPU 1211C, CPU 1212C, CPU 1214C
(measurements in mm)



CPU 1215C, CPU 1217C
(measurements in mm)



表格 4-1　安装尺寸 (mm)

S7-1200 设备		宽度 A (mm)	宽度 B (mm)	宽度 C (mm)
CPU	CPU 1211C 和 CPU 1212C	90	45	--
	CPU 1214C	110	55	--
	CPU 1215C	130	65（顶部）	底部： C1：32.5 C2：65 C3：32.5
	CPU 1217C	150	75	底部： C1：37.5 C2：75 C3：37.5
信号模块	数字 8 和 16 点 模拟 2、4 和 8 点 热电偶 4 和 8 点 RTD 4 点 SM 1278 IO Link 主站	45	22.5	--
	数字量 DQ 8 x 继电器（切换）	70	35	--
	模拟 16 点 RTD 8 点	70	35	--
	SM 1238 电能表模块	45	22.5	--

S7-1200 设备		宽度 A (mm)	宽度 B (mm)	宽度 C (mm)
通信接口	CM 1241 RS232 和 CM 1241 RS422/485 CM 1243-5 PROFIBUS 主站和 CM 1242-5 PROFIBUS 从站 CM 1242-2 AS-i 主站 CP 1242-7 GPRS V2 CP 1243-7 LTE-US CP 1243-7 LTE-EU CP 1243-1 CP 1243-8 IRC RF120C	30	15	--
	TS（远程服务）Adapter IE Advanced [1] TS（远程服务）Adapter IE Basic[1] TS 适配器 TS 模块	 30 30	 15 15	 -- --

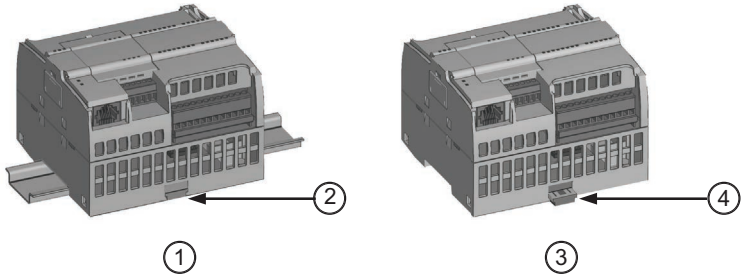
[1] 安装 TS（远程服务）Adapter IE Advanced 或 IE Basic 之前，必须先连接 TS 适配器和 TS 模块。总宽度（“宽度 A”）为 60 mm。

每个 CPU、SM、CM 和 CP 都支持安装在 DIN 导轨或面板上。使用模块上的 DIN 导轨卡夹将设备固定到导轨上。这些卡夹还能掰到一个伸出位置以提供将设备直接安装到面板上的螺钉安装位置。设备上 DIN 卡夹的安装孔内部尺寸是 4.3 mm。

必须在设备的上方和下方留出 25 mm 的发热区以便空气自由流通。

安装和拆卸 S7-1200 设备

CPU 可以很方便地安装到标准 DIN 导轨或面板上。 可使用 DIN 导轨卡夹将设备固定到 DIN 导轨上。 这些卡夹还能掰到一个伸出位置以提供设备面板安装时所用的螺钉安装位置。



① DIN 导轨安装

② DIN 导轨卡夹处于锁紧位置

③ 面板安装

④ 卡夹处于伸出位置用于面板安装

在安装或拆卸任何电气设备之前，请确保已关闭相应设备的电源。 同时，还要确保已关闭所有相关设备的电源。

> ⚠ **警告**
>
> **安装或拆卸已上电的 S7-1200 或相关设备可能会导致电击或意外设备操作。**
>
> 如果在安装或拆卸过程中没有断开 S7-1200 或相关设备的所有电源，则可能会由于电击或意外设备操作而导致死亡、人员重伤和/或财产损失。
>
> 务必遵守适当的安全预防措施，确保在尝试安装或拆卸 S7-1200 CPU 或相关设备前断开 S7-1200 的电源。

务必确保无论何时更换或安装 S7-1200 设备，都使用正确的模块或同等设备。

> ⚠ **警告**
>
> **S7-1200 模块安装不当可能导致 S7-1200 中的程序工作异常。**
>
> 如果不是用相同型号、方向或顺序来更换 S7-1200 设备，则可能会由于意外设备操作而导致死亡、人员重伤和/或财产损失。
>
> 请使用相同型号的设备来更换 S7-1200 设备，并确保设备的方向和位置放置正确。

> ⚠ **警告**
>
> **请勿在易燃或易爆环境中断开连接设备。**
>
> 在易燃或易爆环境中断开连接设备可能会引起火灾或爆炸，从而导致死亡、人员重伤和/或财产损失。
>
> 在易燃或易爆环境中使用时请务必遵守相应的安全预防措施。

说明

静电放电可能会损坏设备或 CPU 上的卡槽。

在拿放设备时，请与已接地的导电垫接触或使用接地腕带。

4.3.2 安装和拆卸 CPU

可以将 CPU 安装到 DIN 导轨或面板上。

说明

将全部通信模块连接到 CPU 上，然后将该组件作为一个单元来安装。 在安装 CPU 之后分别安装信号模块。

将该单元安装到 DIN 导轨或面板上时，应考虑以下几点：

- 若是 DIN 导轨安装，确保 CPU 和相连 CM 的上部 DIN 导轨卡夹处于锁紧（内部）位置而下部 DIN 导轨卡夹处于伸出位置。
- 将设备安装到 DIN 导轨上后，将下部 DIN 导轨卡夹推到锁紧位置以将设备锁定在 DIN 导轨上。
- 若是面板安装，确保将 DIN 导轨卡夹推到伸出位置。

要将 CPU 安装到面板上，请按以下步骤操作：

1. 按照安装尺寸 (mm) (页 56) 表中所示的尺寸，执行定位、钻孔和攻丝以准备安装孔 (M4)。
2. 确保 CPU 和所有 S7-1200 设备都与电源断开。
3. 从模块上掰出安装卡夹。 确保 CPU 上部和下部的 DIN 导轨卡夹都处于伸出位置。
4. 使用带弹簧和平垫圈的 Pan Head M4 螺钉将模块固定到面板上。 不要使用平头螺钉。

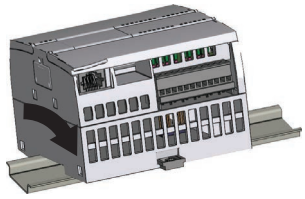
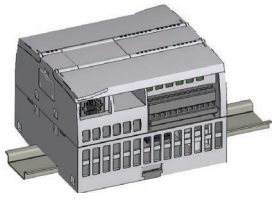
说明

螺钉类型将由安装时的材料决定。 应施加适当的扭矩，直到弹簧垫圈变平。 避免对安装螺钉施加过多扭矩。 不要使用平头螺钉。

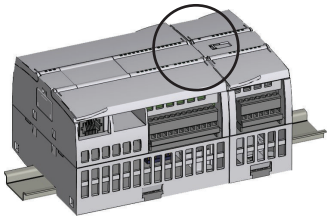
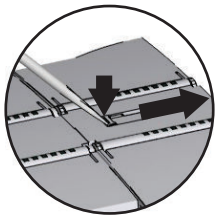
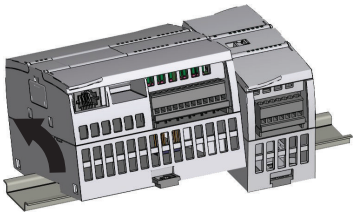
说明

当 CPU 的使用环境振动比较大或垂直安装时，使用 DIN 导轨挡块可能会有帮助。 在 DIN 导轨上使用端盖（8WA1808 或 8WA1805）以确保模块保持连接状态。 如果系统处于剧烈振动环境中，面板安装可给 CPU 提供较高的振动保护等级。

表格 4-2　将 CPU 安装在 DIN 导轨上

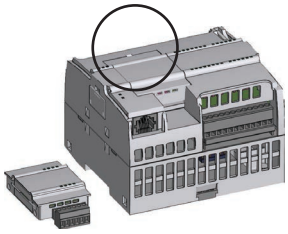
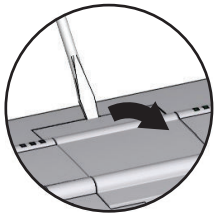
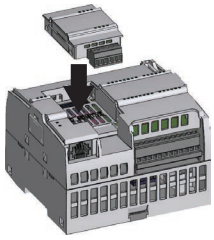
任务	步骤
	1. 安装 DIN 导轨。 每隔 75 mm 将导轨固定到安装板上。 2. 确保 CPU 和所有 S7-1200 设备都与电源断开。 3. 将 CPU 挂到 DIN 导轨上方。 4. 拉出 CPU 下方的 DIN 导轨卡夹以便能将 CPU 安装到导轨上。 5. 向下转动 CPU 使其在导轨上就位。 6. 推入卡夹将 CPU 锁定到导轨上。
	

表格 4-3　将 CPU 从 DIN 导轨上卸下

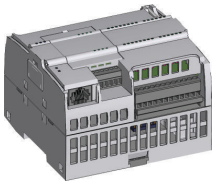
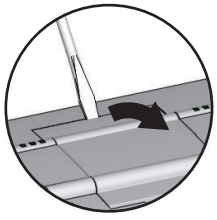
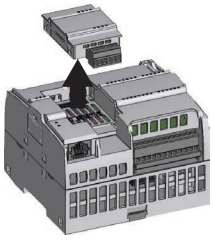
任务	步骤
  	1. 确保 CPU 和所有 S7-1200 设备都与电源断开。 2. 从 CPU (页 67) 断开 I/O 连接器、接线和电缆。 3. 将 CPU 和所有相连的通信模块作为一个完整单元拆卸。 所有信号模块应保持安装状态。 4. 如果 SM 已连接到 CPU，则需要缩回总线连接器： – 将螺丝刀放到信号模块上方的小接头旁。 – 向下按使连接器与 CPU 相分离。 – 将小接头完全滑到右侧。 5. 卸下 CPU： – 拉出 DIN 导轨卡夹从导轨上松开 CPU。 – 向上转动 CPU 使其脱离导轨，然后从系统中卸下 CPU。

4.3.3 安装和拆卸 SB、CB 或 BB

表格 4-4 安装 SB、CB 或 BB 1297

任务	步骤
  	1. 确保 CPU 和所有 S7-1200 设备都与电源断开。 2. 卸下 CPU 上部和下部的端子板盖板。 3. 将螺丝刀插入 CPU 上部接线盒盖背面的槽中。 4. 轻轻将盖直接撬起并从 CPU 上卸下。 5. 将模块直接向下放入 CPU 上部的安装位置中。 6. 用力将模块压入该位置直到卡入就位。 7. 重新装上端子板盖子。

表格 4-5 拆卸 SB、CB 或 BB 1297

任务	步骤
  	1. 确保 CPU 和所有 S7-1200 设备都与电源断开。 2. 卸下 CPU 上部和下部的端子板盖板。 3. 用螺丝刀轻轻分离以卸下信号板连接器（如已安装）。 4. 将螺丝刀插入模块上部的槽中。 5. 轻轻将模块撬起使其与 CPU 分离。 6. 不使用螺丝刀，将模块直接从 CPU 上部的安装位置中取出。 7. 将盖板重新装到 CPU 上。 8. 重新装上端子板盖子。

安装或更换 BB 1297 电池板中的电池

BB 1297 要求的电池型号为 CR1025。电池未随 BB 1297 一起提供，必须另行购买。要安装或更换电池，请执行以下步骤：

1. 在 BB 1297 中，将电池正极朝上，负极靠近印刷线路板来安装新电池。
2. BB 1297 已准备好安装到 CPU 中。确保 CPU 和所有 S7-1200 设备都与电源断开，同时按照上述安装指示安装 BB 1297。

更换 BB 1297 中的电池：

1. 确保 CPU 和所有 S7-1200 设备都与电源断开。按照上述拆卸指示将 BB 1297 从 CPU 中取出。
2. 使用小号螺丝刀小心地取下旧电池。将电池从卡夹下部推出。
3. 安装新的 CR1025 替换电池时，使电池正极朝上，负极靠近印刷线路板。
4. 按照上述安装指示重新安装 BB 1297 电池板。

⚠ **警告**

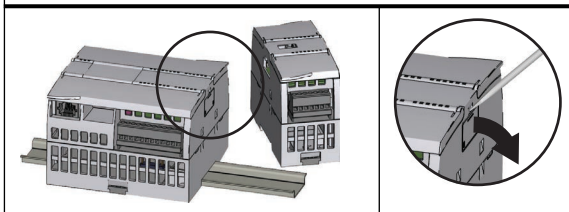
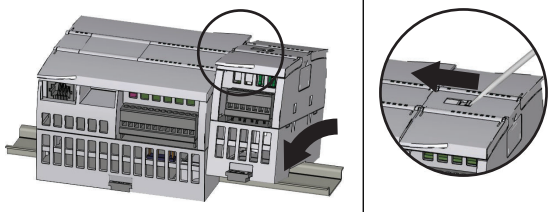
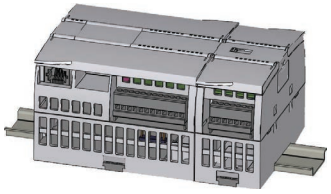
在 BB 1297 中安装未规定的电池或将未规定的电池连接到电路，可能会导致火灾或部件元件损坏以及不可预测的设备运行情况。

火灾或不可预测的设备运行状况可能导致死亡、严重人身伤害或财产损坏。

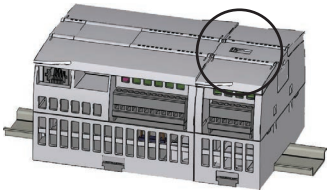
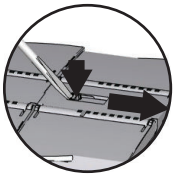
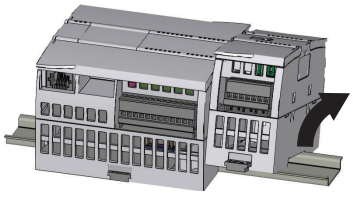
请仅使用规定的 CR1025 电池作为实时时钟的后备电源。

4.3.4 安装和拆卸 SM

表格 4-6 安装 SM

任务	步骤
	在安装 CPU 之后安装 SM。 1. 确保 CPU 和所有 S7-1200 设备都与电源断开。 2. 卸下 CPU 右侧的连接器盖： – 将螺丝刀插入盖上方的插槽中。 – 将其上方的盖轻轻撬出并卸下盖。 3. 收好盖以备再次使用。
	将 SM 连接到 CPU： 1. 将 SM 装在 CPU 旁边。 2. 将 SM 挂到 DIN 导轨上方。 3. 拉出下方的 DIN 导轨卡夹以便将 SM 安装到导轨上。 4. 向下转动 CPU 旁的 SM 使其就位并推入下方的卡夹将 SM 锁定到导轨上。
	伸出总线连接器即为 SM 建立了机械和电气连接。 1. 将螺丝刀放到 SM 上方的小接头旁。 2. 将小接头滑到最左侧，使总线连接器伸到 CPU 中。 要接着信号模块再安装信号模块，请按照相同的步骤操作。

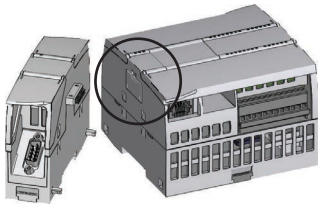
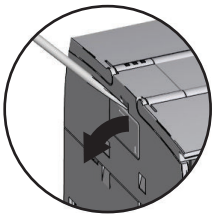
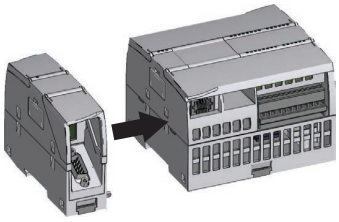
表格 4-7　　卸下 SM

任务	步骤
	可以在不卸下 CPU 或其它 SM 处于原位时卸下任何 SM。 1. 确保 CPU 和所有 S7-1200 设备都与电源断开。 2. 将 I/O 连接器和接线从 SM (页 67) 上卸下。 3. 缩回总线连接器。 – 将螺丝刀放到 SM 上方的小接头旁。 – 向下按使连接器与 CPU 相分离。 – 将小接头完全滑到右侧。 如果右侧还有 SM，则对该 SM 重复该步骤。
	
	卸下 SM： 1. 拉出下方的 DIN 导轨卡夹从导轨上松开 SM。 2. 向上转动 SM 使其脱离导轨。 从系统中卸下 SM。 3. 如有必要，用盖子盖上 CPU 的总线连接器以避免污染。 要拆除信号模块旁的信号模块，请按照相同的步骤操作。

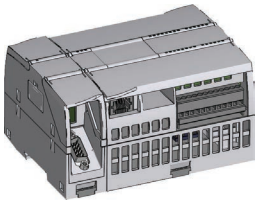
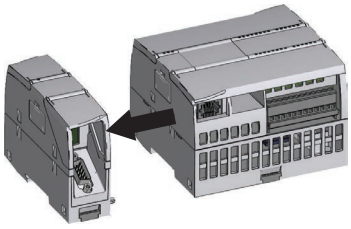
4.3.5 安装和拆卸 CM 或 CP

将全部通信模块连接到 CPU 上，然后将该组件作为一个单元来安装，如安装和拆卸 CPU (页 60) 中所示。

表格 4-8 安装 CM 或 CP

任务	步骤
  	1. 确保 CPU 和所有 S7-1200 设备都与电源断开。 2. 请首先将 CM 连接到 CPU 上，然后再将整个组件作为一个单元安装到 DIN 导轨或面板上。 3. 卸下 CPU 左侧的总线盖： – 将螺丝刀插入总线盖上方的插槽中。 – 轻轻撬出上方的盖。 4. 卸下总线盖。 收好盖以备再次使用。 5. 将 CM 或 CP 连接到 CPU 上： – 使 CM 的总线连接器和接线柱与 CPU 上的孔对齐。 – 用力将两个单元压在一起直到接线柱卡入到位。 6. 将 CPU 和 CP 安装到 DIN 导轨或面板上。

表格 4-9 拆卸 CM 或 CP

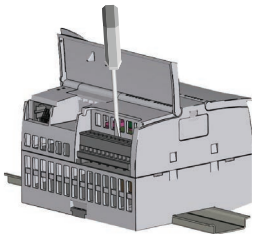
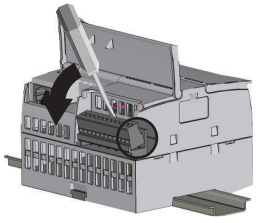
任务	步骤
 	将 CPU 和 CM 作为一个完整单元从 DIN 导轨或面板上卸下。 1. 确保 CPU 和所有 S7-1200 设备都与电源断开。 2. 拆除 CPU 和 CM 上的 I/O 连接器和所有接线及电缆。 3. 对于 DIN 导轨安装，将 CPU 和 CM 上的下部 DIN 导轨卡夹掰到伸出位置。 4. 从 DIN 导轨或面板上卸下 CPU 和 CM。 5. 用力抓住 CPU 和 CM，并将它们分开。

注意
不要使用工具分离模块。 请不要使用工具来分离模块，否则可能损坏设备。

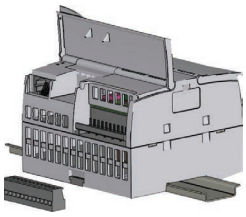
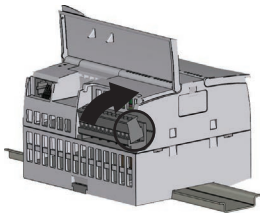
4.3.6 拆卸和重新安装 S7-1200 端子板连接器

CPU、SB 和 SM 模块提供了方便接线的可拆卸连接器。

表格 4-10 拆卸连接器

任务	步骤
	通过卸下 CPU 的电源并打开连接器上的盖子，准备从系统中拆卸端子板连接器。 1. 确保 CPU 和所有 S7-1200 设备都与电源断开。 2. 查看连接器的顶部并找到可插入螺丝刀头的槽。 3. 将螺丝刀插入槽中。 4. 轻轻撬起连接器顶部使其与 CPU 分离。 连接器从夹紧位置脱离。 5. 抓住连接器并将其从 CPU 上卸下。
	

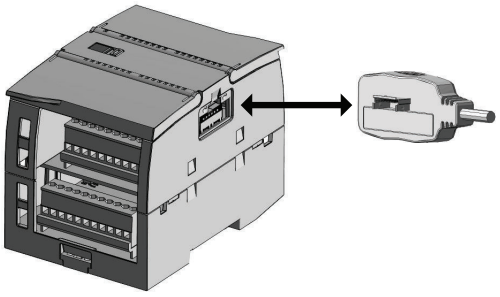
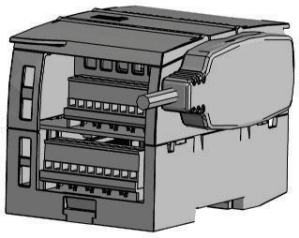
表格 4-11　安装连接器

任务	步骤
	通过断开 CPU 的电源并打开连接器的盖子，准备端子板安装的组件。 1. 确保 CPU 和所有 S7-1200 设备都与电源断开。 2. 使连接器与单元上的插针对齐。 3. 将连接器的接线边对准连接器座沿的内侧。 4. 用力按下并转动连接器直到卡入到位。 仔细检查以确保连接器已正确对齐并完全啮合。
	

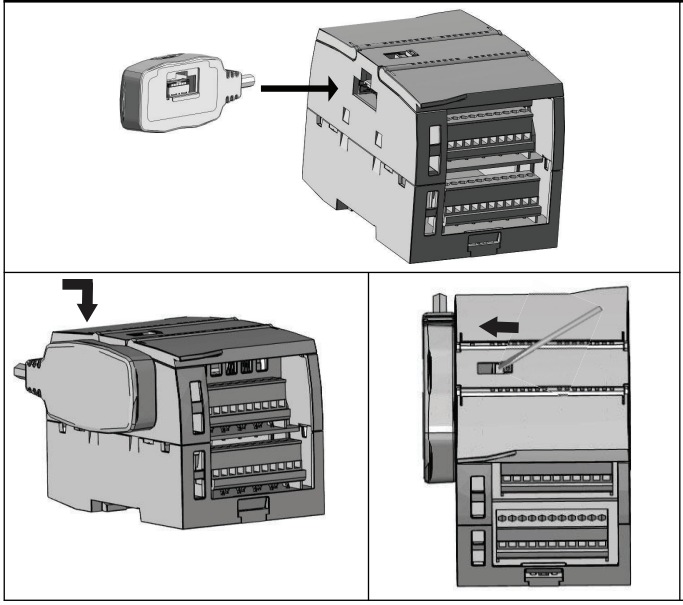
4.3.7 安装和卸下扩展电缆

S7-1200 扩展电缆可用来更灵活地组态 S7-1200 系统的布局。每个 CPU 系统只允许使用一条扩展电缆。可以将扩展电缆安装在 CPU 和第一个 SM 之间，或者安装在任意两个 SM 之间。

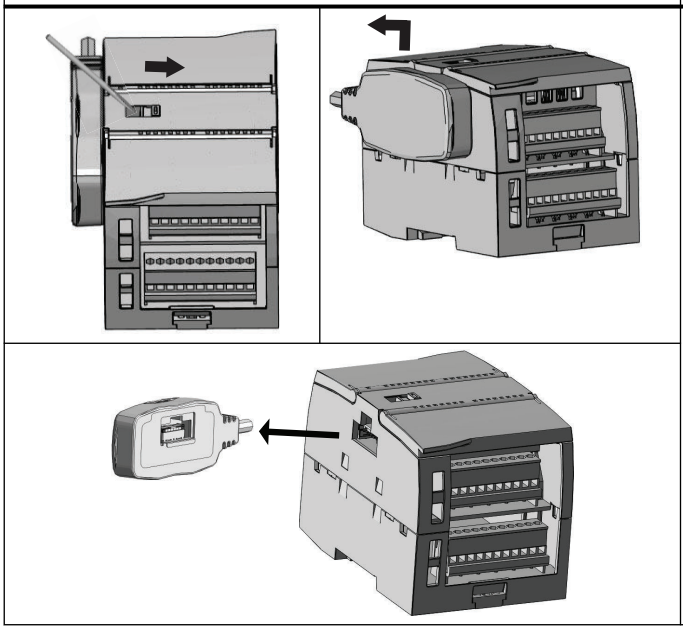
表格 4-12　安装和卸下扩展电缆的公连接器

任务	步骤
	要安装公连接器： 1. 确保 CPU 和所有 S7-1200 设备都与电源断开。 2. 将公连接器按压到信号模块或 CPU 右侧的总线连接器中。 要卸下公连接器： 1. 确保 CPU 和所有 S7-1200 设备都与电源断开。 2. 拔出公连接器，使其从信号模块或 CPU 上松开。
	

表格 4-13　安装扩展电缆的母连接器

任务	步骤
	1. 确保 CPU 和所有 S7-1200 设备都与电源断开。 2. 将母连接器放到信号模块左侧的总线连接器上。 3. 将母连接器的钩伸端滑入总线连接器处的外壳，并轻轻按下，使钩咬合。 4. 将连接器锁定到位： – 将螺丝刀放到信号模块上方的小接头旁。 – 将小接头完全滑到左侧。 要使连接器啮合，必须将连接器小接头一直向左滑动。必须将连接器小接头锁定到位。

表格 4-14　卸下扩展电缆的母连接器

任务	步骤
	1. 确保 CPU 和所有 S7-1200 设备都与电源断开。 2. 解除锁定连接器： – 将螺丝刀放到信号模块上方的小接头旁。 – 轻轻按下连接器，将小接头完全滑到右侧。 3. 轻轻向上提起连接器，使钩伸端分离。 4. 卸下母连接器。

说明

在振动环境中安装扩展电缆

如果将扩展电缆连接在移动或固定不牢的模块上，电缆插入端的摁扣连接可能会慢慢松动。

为了提供额外的应力消除作用，应使用电缆扎带将插入端电缆固定在 DIN 导轨（或其它位置）上。

安装期间拉拽电缆时应避免用力过猛。安装完成后，确保电缆与模块连接到位。

4.3.8 TS（远程服务）适配器

4.3.8.1 连接远程服务适配器

安装 TS（远程服务）Adapter IE Basic 或 TS（远程服务）Adapter IE Advanced 之前，必须先连接 TS 适配器和 TS 模块。

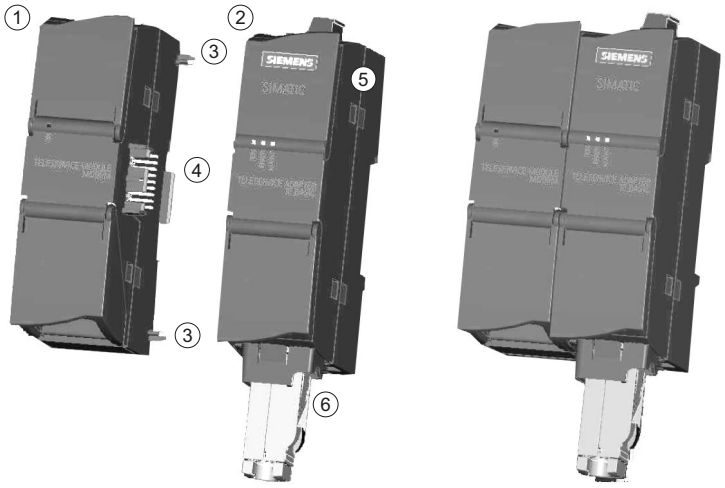
可用的 TS 模块：

- TS 模块 RS232
- TS 模块 Modem
- TS 模块 GSM
- TS 模块 ISDN

说明

如果接触 TS 模块的插头连接器 ④ 的触点，则可能损坏 TS 模块。

请遵守 ESD 准则，以免静电放电损坏 TS 模块。 在连接 TS 模块和 TS 适配器之前，请确保它们都处于空闲状态。



①	TS 模块	④	TS 模块的插头连接器
②	TS 适配器	⑤	无法打开
③	部件	⑥	以太网端口

说明

在连接 TS 模块和 TS 适配器基本单元之前，确保触针 ④ 没有弯曲。

连接时，确保公连接器和导销位置正确。

只能把一个 TS 模块连接到 TS 适配器中。 请勿将 TS 适配器强行连接到不同设备，如 S7-1200 CPU。 请勿更改连接器的机械构造，也不要卸下或损坏导销。

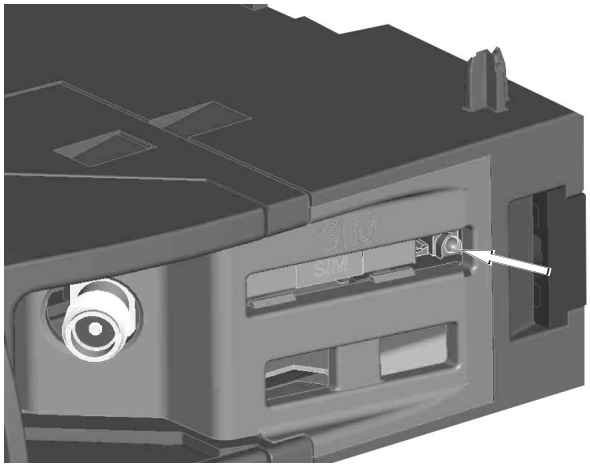
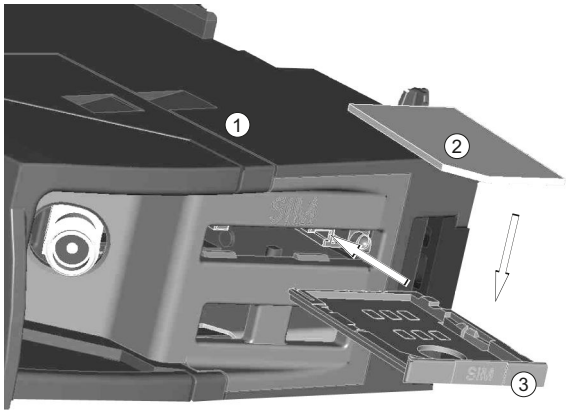
4.3.8.2 安装 SIM 卡

将 SIM 卡插槽置于 TS module GSM 下方。

说明

只有在 TS module GSM 断电情况下才能卸下或插入 SIM 卡。

表格 4-15　安装 SIM 卡

任务	步骤
	使用尖物按压 SIM 卡托上的弹出按钮（箭头方向），取出 SIM 卡托。
	如图所示将 SIM 卡放入 SIM 卡托，然后将 SIM 卡托放回卡槽中。 ①TS Module GSM ② SIM 卡 ③ SIM 卡托

说明

确保卡托中的 SIM 卡朝向正确。 否则，SIM 卡无法与模块连接，弹出按钮也可能无法弹出卡托。

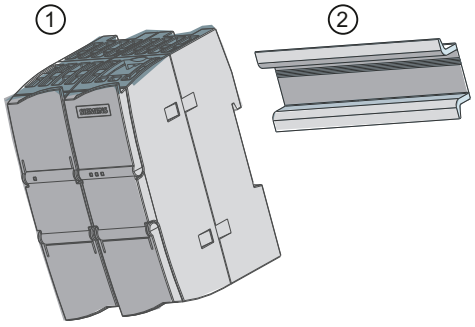
4.3.8.3 将 TS 适配器单元安装在 DIN 导轨上

先决条件： 必须已将 TS 适配器和 TS 模块连接在一起，且必须已安装 DIN 导轨。

说明

如果垂直安装 TS 单元或在剧烈振动环境中进行安装，TS 模块可能与 TS 适配器断开连接。在 DIN 导轨上使用端盖 8WA1808 以确保模块保持连接状态。

表格 4-16　安装和拆卸 TS 适配器

任务	步骤
	安装： 1. 将连有 TS 模块的 TS 适配器 ① 挂在 DIN 导轨 ② 上。 2. 向后旋转单元，直到咬合为止。 3. 推入每个模块上的 DIN 导轨卡夹，将各个模块固定在导轨上。
	拆卸： 1. 从 TS 适配器下方卸下模拟电缆和以太网电缆。 2. 断开 TS 适配器的电源。 3. 用螺丝刀松开两个模块上的导轨卡夹。 4. 向上旋转单元，将其从导轨上卸下。

警告

安装或拆卸 TS 适配器的安全要求。

断开单元的电源之前，先卸下模拟电缆和以太网电缆以断开 TS 适配器的接地连接。 不遵守本预防措施可引发意外的设备操作，进而导致死亡、人员重伤和/或财产损失。

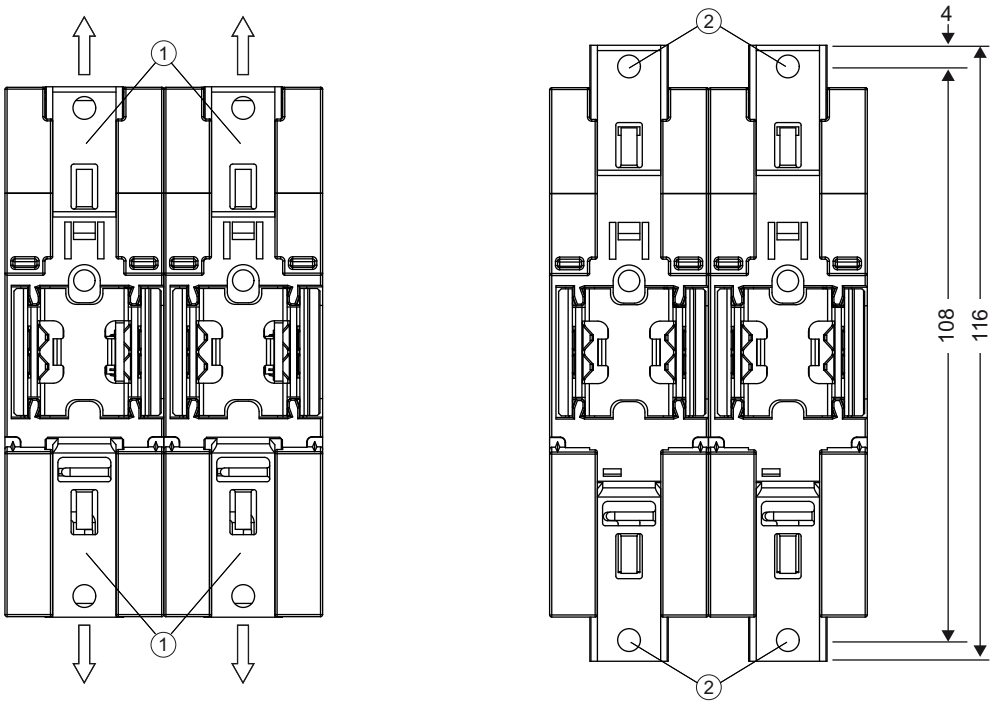
安装或拆卸 TS 适配器过程中请始终遵守上述要求。

4.3.8.4　将 TS 适配器安装到面板上

先决条件： 必须已连接 TS 适配器和 TS 模块。

1. 沿箭头方向将连接滑块 ① 朝 TS 适配器和 TS 模块的后方推，直至其咬合。
2. 用螺钉将 TS 适配器和 TS 模块固定到指定安装墙上标有 ② 的位置。

下图为 TS 适配器的后视图，在两个位置都有连接滑块 ①：



① 连接滑块

② 墙式安装用的钻孔

4.4 接线准则

所有电气设备的正确接地和接线非常重要，因为这有助于确保实现最佳系统运行以及为您的应用和 S7-1200 提供更好的电噪声防护。请参考技术规范 (页 1427)以查看 S7-1200 的接线图。

先决条件

在对任何电气设备进行接地或者接线之前，请确保设备的电源已经断开。 同时，还要确保已关闭所有相关设备的电源。

确保在对 S7-1200 和相关设备接线时遵守所有适用的电气规程。 请根据所有适用的国家和地方标准来安装和操作所有设备。 请联系当地的管理机构确定哪些规范和标准适用于您的具体情况。

⚠ 警告

安装已上电的 S7-1200 或相关设备或者为这些设备接线可能会导致电击或意外设备操作。

如果在安装或拆卸过程中没有断开 S7-1200 或相关设备的所有电源，则可能会由于电击或意外设备操作而导致死亡、人员重伤和/或财产损失。

务必遵守适当的安全预防措施，确保在尝试安装或拆卸 S7-1200 或相关设备前断开 S7-1200 的电源。

在您规划 S7-1200 系统的接地和接线时，务必考虑安全问题。 电子控制设备（如 S7-1200）可能会失灵和导致正在控制或监视的设备出现意外操作。 因此，应采取一些独立于 S7-1200 的安全措施以防止可能的人员受伤或设备损坏。

⚠ 警告

控制设备在不安全情况下运行时可能会出现故障，从而导致受控设备的意外运行。

这种意外操作可能会导致死亡、人员重伤和/或财产损失。

应使用紧急停止功能、机电超控功能或其它独立于 S7-1200 的冗余安全功能。

绝缘准则

S7-1200 交流电源和 I/O 与交流电路的边界经过设计，经验证可以在交流线路电压与低压电路之间实现安全隔离。根据各种适用的标准，这些边界包括双重或加强绝缘，或者基本绝缘加辅助绝缘。跨过这些边界的组件（例如，光耦合器、电容器、变压器和继电器）已通过安全隔离认证。仅采用交流线路电压的电路才与其它电路实现安全隔离。24 V DC 电路间的隔离边界仅起一定作用，不应依赖于这些边界提供安全性。

根据 EN 61131-2，集成有交流电源的 S7-1200 的传感器电源输出、通信电路和内部逻辑电路属于 SELV（安全超低电压）电路。

要维持 S7-1200 低压电路的安全特性，到通信端口、模拟电路以及所有 24 V DC 额定电源和 I/O 电路的外部连接必须由合格的电源供电，该电源必须满足各种标准对 SELV、PELV、2 类、限制电压或受限电源的要求。

⚠ 警告
若使用非隔离或单绝缘电源通过交流线路给低压电路供电，可能会导致本来应当可以安全触摸的电路上出现危险电压，例如，通信电路和低压传感器线路。 这种意外的高压可能会引起电击而导致死亡、人员重伤和/或财产损失。 只应当使用合格的高压转低压整流器作为可安全接触的限压电路的供电电源。

S7-1200 的接地准则

将应用设备接地的最佳方式是确保 S7-1200 和相关设备的所有公共端和接地连接在同一个点接地。 该点应该直接连接到系统的大地接地。

所有地线应尽可能地短且应使用大线径，例如，2 mm^2 (14 AWG)。

确定接地点时，应考虑安全接地要求和保护性中断装置的正常运行。

S7-1200 的接线准则

规划 S7-1200 的接线时，应提供一个可同时切断 S7-1200 CPU 电源、所有输入电路和所有输出电路电力供应的隔离开关。请提供过流保护（例如，熔断器或断路器）以限制电源线中的故障电流。考虑在各输出电路中安装熔断器或其它电流限制器提供额外保护。

为所有可能遭雷电冲击的线路安装合适的浪涌抑制设备。有关详细信息，请参阅“一般技术数据”部分中的浪涌抗扰性 (页 1427)。

避免将低压信号线和通信电缆铺设在同一个具有交流线和高能量快速开关直流线的线槽中。始终成对布线，中性线或公共线与火线或信号线成对。

使用尽可能短的电线并确保线径适合承载所需电流。

导线和电缆因具有高于 S7-1200 周围的环境温度 30 °C 的温度等级（例如，针对 55 °C 的环境温度，应采用温度等级至少为 85 °C 的电缆）。应从特定电路图额定值和安装环境来确定其它导线类型和材料要求。

使用屏蔽线以便最好地防止电噪声。通常在 S7-1200 端将屏蔽层接地能获得最佳效果。应使用与电缆屏蔽层相连的连接器将通信电缆屏蔽层接地至 S7-1200 通信连接器外壳，或将通信电缆屏蔽层与单独的接地端相连。应围绕屏蔽层使用夹子或铜带来提供较大的接地点连接表面，将其它电缆屏蔽层接地。

在给通过外部电源供电的输入电路接线时，应在电路中安装过流保护装置。由 S7-1200 的 24 V DC 传感器电源供电的电路不需要外部保护，因为该传感器电源的电流已经受到限制。

所有 S7-1200 模块都有供用户接线的可拆卸连接器。要防止连接器松动，请确保连接器固定牢靠并且导线被牢固地安装到连接器中。

为了有利于防止安装中出现意外的电流，S7-1200 在某些点提供绝缘边界。在您规划系统的接线时，应考虑这些绝缘边界。有关所提供的绝缘程度和绝缘边界位置的信息，请参见技术规范 (页 1503)。采用交流线路电压的电路与其它电路实现安全隔离。24 V DC 电路间的隔离边界仅起一定作用，不应依赖于这些边界提供安全性。

下表总结了 S7-1200 CPU、SM 和 SB 的接线规则。

表格 4-17　S7-1200 CPU、SM 和 SB 的接线规则

适用的接线规则...	CPU 与 SM 连接器		SB 连接器
连接技术	推入	螺钉	螺钉
软绞线的可连接导线横截面	2 mm^2 到 0.3 mm^2（14 AWG 到 22 AWG）		1.3 mm^2 至 0.3 mm^2（16 AWG 至 22 AWG）
每个连接的导线数	1 根或 2 根导线组合在双层套筒中，总横截面积最大可为 2 mm^2		1 根或 2 根导线，总横截面为 1.3 mm^2
剥线长度	使用套管进行牢固的电连接	6.4 mm	6.3 到 7 mm
拧紧扭矩*（最大值）	空	0.56 N·m（5 英寸 • 磅）	0.33 N·m（3 英寸 • 磅）
工具	2.5 到 3.0 mm 一字螺丝刀		

* 为避免损坏连接器，小心不要将螺丝拧得过紧。

说明

绞线上的端头或末端套管会降低散丝导致短路的风险。长度超过建议剥线长度的端头应包括一个绝缘环以避免因导线侧向移动而导致的短路。裸导线的横截面积限制也适用于端头。

参见

技术规范 (页 1427)

灯负载的使用准则

由于接通浪涌电流大，灯负载（包括 LED 灯负载）会损坏继电器触点。该浪涌电流通常是钨灯稳态电流的 10 到 15 倍。对于在应用期间将进行大量开关操作的灯负载，建议安装可更换的插入式继电器或浪涌限制器。

感性负载的使用准则

将抑制电路与感性负载配合使用，以在控制输出断开时限制电压升高。抑制电路可保护输出，防止通过感性负载中断电流时产生的高压瞬变导致其过早损坏。

此外，抑制电路还能限制开关感性负载时产生的电噪声。抑制能力差的感性负载产生的高频噪声会中断 PLC 的运行。配备一个外部抑制电路，使其从电路上跨接在负载两端并且在位置上接近负载，这样对降低电气噪声最有效。

S7-1200 的 DC 输出包括内部抑制电路，该电路足以满足大多数应用对感性负载的要求。由于 S7-1200 继电器输出触点可用于开关直流或交流负载，所以未提供内部保护。

一种良好的抑制解决方案是使用接触器或其它感性负载，制造商为这些感性负载提供了集成在负载设备中的抑制电路，或将抑制电路作为可选附件提供。但是，一些制造商提供的抑制电路可能不适合您的应用。为获得最佳的噪声消减和触点寿命，可能还需要额外的抑制电路。

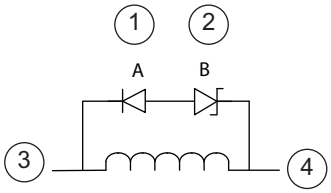
对于交流负载，可将金属氧化物变阻器 (MOV) 或其它电压钳制设备与并联 RC 电路配合使用，但不如单独使用有效。不带并联 RC 电路的 MOV 抑制器通常会导致出现高达钳位电压的显著高频噪声。

良好的受控关断瞬变的振铃频率不超过 10 kHz，最好小于 1 kHz。交流线路的峰值电压对地应在 +/- 1200 V 的范围内。使用 PLC 内部抑制的直流负载的负峰值电压比 24 V DC 电源电压低大约 40 V。外部抑制应将瞬变限制在 36 V 电源范围内，以卸载内部抑制。

说明

抑制电路的有效性取决于具体应用，必须验证其是否适合您的具体应用。确保所有组件的额定值均正确，并使用示波器观察关断瞬变。

用于开关 DC 感性负载的 DC 或继电器输出的典型抑制电路



① 1N4001 二极管或同等元件

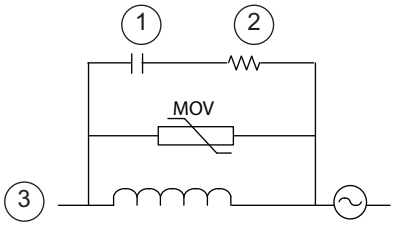
② 8.2 V 稳压二极管（直流输出），
36 V 稳压二极管（继电器输出）

③ 输出点

④ M，24 V 参考

在大多数应用中，在直流感性负载两端增加一个二极管 (A) 就可以了，但如果您的应用要求更快的关闭时间，则建议再增加一个稳压二极管 (B)。请确保正确选择稳压二极管，以适合输出电路中的电流量。

用于开关 AC 感性负载的继电器输出的典型抑制电路



① 关于 C 值，请参见表格

② 关于 R 值，请参见表格

③ 输出点

请确保金属氧化物变阻器 (MOV) 的工作电压至少比额定线电压高出 20%。

选择为脉冲应用推荐的脉冲级非感性电阻和电容（通常为金属薄膜型）。确认元件满足平均功率、峰值功率和峰值电压要求。

如果自行设计抑制电路，下表给出了一系列交流负载的建议电阻值和电容值。这些值是理想元件参数下的计算结果。表中的 I rms 指满载时负载的稳态电流。

表格 4-18　交流抑制电路电阻和电容值

感性负载			抑制值		
I rms	230 V AC	120 V AC	电阻		电容
A	VA	VA	Ω	W（功率额定值）	nF
0.02	4.6	2.4	15000	0.1	15
0.05	11.5	6	5600	0.25	470
0.1	23	12	2700	0.5	100
0.2	46	24	1500	1	150
0.5	115	60	560	2.5	470

感性负载			抑制值		
1	230	120	270	5	1000
2	460	240	150	10	1500

表中的值满足的条件：

最大关断瞬变阶跃 < 500 V
电阻峰值电压 < 500 V
电容峰值电压 < 1250 V
抑制电流 < 负载电流的 8% (50 Hz)
抑制电流 < 负载电流的 11% (60 Hz)
电容 dV/dt < 2 V/μs
电容脉冲功耗：$\int(dv/dt)^2\ dt < 10000\ V^2/\mu s$
谐振频率 < 300 Hz
电阻功率对应于 2 Hz 最大开关频率
假设典型感性负载的功率因数为 0.3

差分输入和输出准则

差分输入和输出与标准输入和输出不同。 每个差分输入和输出都有两个引脚。 要判断差分输入或输出是开启还是关闭，可测量这两个引脚之间的电压差。

请参见附录 A 中 CPU 1217C (页 1503)的详细规范。

PLC 概念 5

5.1 用户程序的执行

CPU 支持以下类型的代码块，使用它们可以创建有效的用户程序结构：

- 组织块 (OB) 定义程序的结构。 有些 OB 具有预定义的行为和启动事件，但用户也可以创建具有自定义启动事件的 OB。
- 功能 (FC) 和功能块 (FB) 包含与特定任务或参数组合相对应的程序代码。 每个 FC 或 FB 都提供一组输入和输出参数，用于与调用块共享数据。 FB 还使用相关联的数据块（称为背景数据块）来保存该 FB 调用实例的数据值。 可多次调用 FB，每次调用都采用唯一的背景数据块。 调用带有不同背景数据块的同一 FB 不会对其它任何背景数据块的数据值产生影响。
- 数据块 (DB) 存储程序块可以使用的数据。

用户程序的执行顺序是：从一个或多个在进入 RUN 模式时运行一次的可选启动组织块 (OB) 开始，然后执行一个或多个循环执行的程序循环 OB。还可以将 OB 与中断事件关联，该事件可以是标准事件或错误事件。当发生相应的标准或错误事件时，即会执行这些 OB。

功能 (FC) 或功能块 (FB) 是指可从 OB 或其它 FC/FB 调用的程序代码块，可下至以下嵌套深度：

- 16（从程序循环 OB 或启动 OB 开始）
- 6（从任意中断事件 OB 开始）
  注：安全程序使用二级嵌套。因此，用户程序在安全程序中的嵌套深度为四。

FC 不与任何特定数据块 (DB) 相关联。FB 与 DB 直接相关并使用该 DB 传递参数及存储中间值和结果。

用户程序、数据及组态的大小受 CPU 中可用装载存储器和工作存储器的限制。对各个 OB、FC、FB 和 DB 块的数目没有特殊限制。但是块的总数限制在 1024 之内。

每个周期都包括写入输出、读取输入、执行用户程序指令以及执行后台处理。该周期称为扫描周期或扫描。

S7-1200 自动化解决方案可由配备 S7-1200 CPU 和附加模块的中央机架组成。术语“中央机架”表示 CPU 和关联模块采用导轨或面板式安装。只有在通电时才会对模块（SM、SB、BB、CB、CM 或 CP）进行检测和记录。

- 不支持通电时在中央机架中插入或拔出模块（热插拔）。切勿在 CPU 通电时在中央机架中插入或拔出模块。

⚠ 警告
插入或拔出模块的安全要求 从中央机架插入或移除模块（SM、SB、BB、CD、CM 或 CP）之前，如果未禁用 CPU 的所有电源，可能会造成损坏或不可预测的行为，从而导致死亡或人员重伤和/或财产损失。 在中央机架中插入或拔出模块前，请务必切断 CPU 和中央机架的电源并遵守相应的安全预防措施。

- 可在 CPU 通电时插入或拔出 SIMATIC 存储卡。但在 CPU 处于 RUN 模式时插入或拔出存储卡会使 CPU 进入 STOP 模式。

注意
CPU 处于 RUN 模式时拔出存储卡的风险 在 CPU 处于 RUN 模式时插入或拔出存储卡会使 CPU 进入 STOP 模式，这可能导致受控的设备或过程受损。 只要插入或拔出存储卡，CPU 就立即进入 STOP 模式。在插入或拔出存储卡前，务必确保 CPU 当前未控制任何机器或过程。因此务必要为您的应用或过程安装急停电路。

- 如果在 CPU 处于 RUN 模式时在分布式 I/O 机架（AS-i、PROFINET 或 PROFIBUS）中插入或拔出模块，CPU 将在诊断缓冲区中生成一个条目，若存在拔出或插入模块 OB 则执行该 OB，并且默认保持在 RUN 模式。

过程映像更新与过程映像分区

CPU 伴随扫描周期使用内部存储区（即过程映像）对本地数字量和模拟量 I/O 点进行同步更新。过程映像包含物理输入和输出（CPU、信号板和信号模块上的物理 I/O 点）的快照。

可组态在每个扫描周期或发生特定事件中断时在过程映像中对 I/O 点进行更新。也可对 I/O 点进行组态使其排除在过程映像的更新之外。例如，当发生如硬件中断这类事件时，过程可能只需要特定的数据值。通过为这些 I/O 点组态映像过程更新，使其与分配给硬件中断 OB 的分区相关联，就可避免在过程不需要持续更新时，CPU 于每个扫描周期中执行不必要的数据值更新。

对于需要在每个扫描周期进行更新的 I/O，CPU 将在每个扫描周期期间执行以下任务：

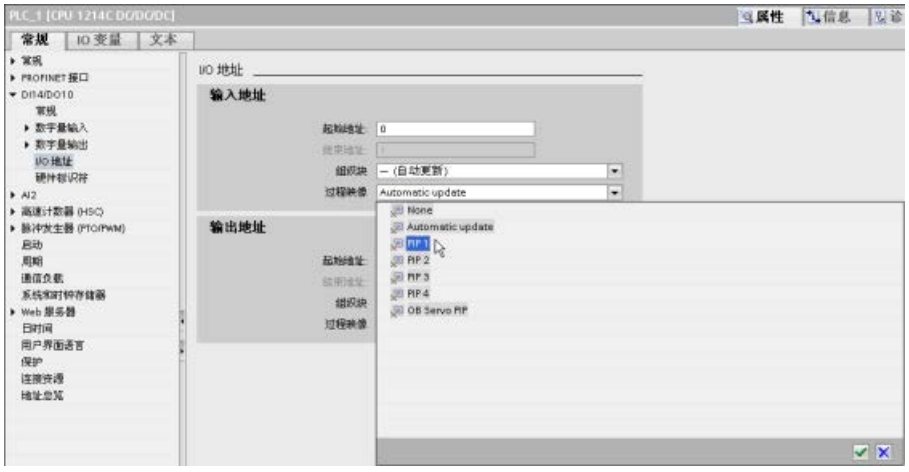
- CPU 将过程映像输出区中的输出值写入到物理输出。
- CPU 仅在用户程序执行前读取物理输入，并将输入值存储在过程映像输入区。这样一来，这些值便将在整个用户指令执行过程中保持一致。
- CPU 执行用户指令逻辑，并更新过程映像输出区中的输出值，而不是写入实际的物理输出。
  这一过程通过在给定周期内执行用户指令而提供一致的逻辑，并防止物理输出点可能在过程映像输出区中多次改变状态而出现抖动。

为控制在每个扫描周期或在事件触发时是否自动更新 I/O 点，S7-1200 提供了五个过程映像分区。第一个过程映像分区 PIP0 指定用于每个扫描周期都自动更新的 I/O，此为默认分配。其余四个分区 PIP1、PIP2、PIP3 和 PIP4 可用于将 I/O 过程映像更新分配给不同的中断事件。在设备组态中将 I/O 分配给过程映像分区，并在创建中断 OB (页 178) 或编辑 OB 属性 (页 178)时将过程映像分区分配给中断事件。

默认情况下，在设备视图中插入模块时，STEP 7 会将其 I/O 过程映像更新为“自动更新”(Automatic update)。对于组态为“自动更新”(Automatic update) 的 I/O，CPU 将在每个扫描周期自动处理模块和过程映像之间的数据交换。

要将数字量或模拟量点分配给过程映像分区，或将 I/O 点排除在过程映像更新之外，请按照以下步骤操作：

1. 在设备组态中查看相应设备的“属性”(Properties) 选项卡。
2. 根据需要在“常规 (General)”下展开选项，找出所需的 I/O 点。
3. 选择“I/O 地址”(I/O addresses)。
4. 也可以从“组织块”(Organization block) 下拉列表中选择一个特定的 OB。
5. 在“过程映像”(Process image) 下拉列表中将“自动更新”(Automatic update) 更改为“PIP1”、“PIP2”、“PIP3”、“PIP4”或“无”(None)。选择“无”(None) 表示只能通过立即指令对此 I/O 进行读写。要将这些点重新添加到过程映像自动更新中，请将该选项再次更改为“自动更新”(Automatic update)。



可以在指令执行时立即读取物理输入值和立即写入物理输出值。无论 I/O 点是否被组态为存储到过程映像中，立即读取功能都将访问物理输入的当前状态而不更新过程映像输入区。立即写入物理输出功能将同时更新过程映像输出区（如果相应 I/O 点组态为存储到过程映像中）和物理输出点。如果想要程序不使用过程映像，而是直接从物理点立即访问 I/O 数据，则在 I/O 地址后加后缀“:P”。

说明

使用过程映像分区

如果将 I/O 分配给过程映像分区 PIP1 - PIP4 中的其中一个，但未将 OB 分配给该分区，那么 CPU 决不会将 I/O 更新至过程映像，也不会通过过程映像更新 I/O。将 I/O 分配给未分配相应 OB 的 PIP，相当于将过程映像指定为“无”(None)。可使用直接读指令直接从物理 I/O 中读取 I/O，或使用直接写指令直接写入物理 I/O。CPU 不更新过程映像。

CPU 支持 PROFINET、PROFIBUS、以及 AS-Interface 网络 (页 805)的分布式 I/O。

5.1.1 CPU 的工作模式

CPU 有以下三种工作模式：STOP 模式、STARTUP 模式和 RUN 模式。CPU 前面的状态 LED 指示当前工作模式。

- 在 STOP 模式下，CPU 不执行程序。您可以下载项目。
- 在 STARTUP 模式下，执行一次启动 OB（如果存在）。在启动模式下，CPU 不会处理中断事件。
- 在 RUN 模式，程序循环 OB 重复执行。RUN 模式中的任意点处都可能发生中断事件，这会导致相应的中断事件 OB 执行。可在 RUN 模式下下载项目的某些部分 (页 1406)。

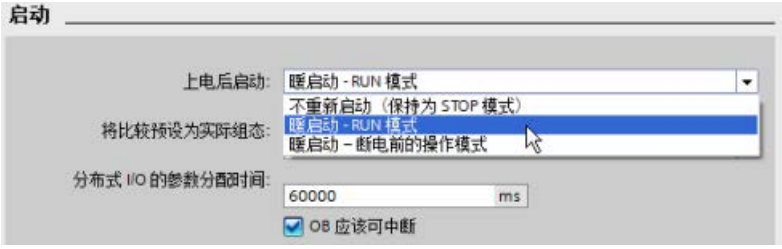
CPU 支持通过暖启动进入 RUN 模式。暖启动不包括储存器复位。执行暖启动时，CPU 会初始化所有的非保持性系统和用户数据，并保留所有保持性用户数据值。

存储器复位将清除所有工作存储器、保持性及非保持性存储区、将装载存储器复制到工作存储器并将输出设置为组态的“对 CPU STOP 的响应”(Reaction to CPU STOP)。存储器复位不会清除诊断缓冲区，也不会清除永久保存的 IP 地址值。

可组态 CPU 中“上电后启动”(startup after POWER ON) 设置。该组态项出现在 CPU“设备组态”(Device Configuration) 的“启动”(Startup) 下。通电后，CPU 将执行一系列上电诊断检查和系统初始化操作。在系统初始化过程中，CPU 将删除所有非保持性位 (M) 存储器，并将所有非保持性 DB 的内容复位为装载存储器的初始值。CPU 将保留保持性位 (M) 存储器

和保持性 DB 的内容，然后进入相应的工作模式。检测到的某些错误会阻止 CPU 进入 RUN 模式。CPU 支持以下组态选项：

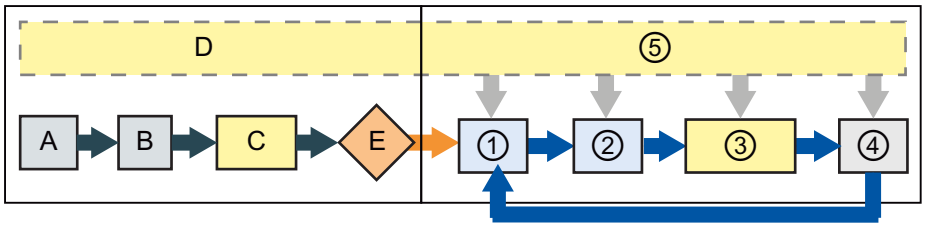
- 不重新启动（保持为 STOP 模式）
- 暖启动 - RUN 模式
- 暖启动 - 断电前的模式



注意
可修复故障可使 CPU 进入 STOP 模式。 CPU 可能因如下可修复故障进入 STOP 模式： ● 可替换信号模块故障 ● 临时故障，如电力线干扰或不稳定上电事件 这种情况可导致财产损失。 如果已将 CPU 组态为“暖启动 - 断电前的模式”(Warm restart - mode prior to POWER OFF)，CPU 则在掉电或发生故障前进入工作模式。如果在发生掉电或故障时，CPU 处于 STOP 模式，则 CPU 将在上电时进入 STOP 模式。CPU 保持 STOP 模式，直至 CPU 收到进入 RUN 模式的命令。如果在发生掉电或故障时，CPU 处于 RUN 模式，则 CPU 将在下次上电时进入 RUN 模式。在 CPU 未检测到可禁止其进入 RUN 模式的条件下，CPU 将进入 RUN 模式。 可将欲独立于 STEP 7 连接而运行的 CPU 组态为“暖启动 - RUN”(Warm restart - RUN)。此启动模式将 CPU 设置为在下一次循环上电时返回到 RUN 模式。

可以使用编程软件在线工具中的“STOP”或“RUN”命令 (页 1392)更改当前工作模式。也可在程序中包含 STP 指令 (页 314)，以使 CPU 切换到 STOP 模式。可通过该指令根据程序逻辑停止程序的执行。

- 在 STOP 模式下，CPU 处理所有通信请求（如果适用）并执行自诊断。CPU 不执行用户程序。过程映像也不会自动更新。
- 在 STARTUP 和 RUN 模式下，CPU 执行下图所示的任务：



STARTUP		RUN	
A	将物理输入的状态复制到 I 存储器	①	将 Q 存储器写入物理输出
B	将 Q 输出（映像）存储区初始化为零、上一个值或组态的替换值将 PB、PN 和 AS-i 输出设为零	②	将物理输入的状态复制到 I 存储器
C	将非保持性 M 存储器和数据块初始化为其初始值，并启用组态的循环中断事件和时钟事件。 执行启动 OB。	③	执行程序循环 OB
D	将所有中断事件存储到要在进入 RUN 模式后处理的队列中	④	执行自检诊断
E	启用 Q 存储器到物理输出的写入操作	⑤	在扫描周期的任何阶段处理中断和通信

说明

包括 HMI 通信在内的通信不能中断程序循环 OB 以外的其它 OB。

启动过程

只要工作模式从 STOP 切换到 RUN，CPU 就会清除过程映像输入、初始化过程映像输出并处理启动 OB。通过 “启动 OB”中的指令对过程映像输入进行任何的读访问，都只会读取零值，而不是读取当前物理输入值。因此，要在启动模式下读取物理输入的当前状态，必须执行立即读取操作。接着再执行启动 OB 以及任何相关的 FC 和 FB。如果存在多个启动 OB，则按照 OB 编号依次执行各 OB，编号最小的 OB 优先执行。

每个启动 OB 都包含帮助您确定保持性数据和时钟有效性的启动信息。可以在启动 OB 中编写指令，以检查这些启动值，从而采取适当的措施。启动 OB 支持以下启动位置：

表格 5-1　启动 OB 支持的启动位置

输入	数据类型	说明
LostRetentive	Bool	如果保持性数据存储区丢失，该位为真
LostRTC	Bool	如果时钟（实时时钟）丢失，该位为真

在启动过程中，CPU 还会执行以下任务：

- 在启动阶段，对中断进行排队但不加以处理
- 在启动阶段，不执行任何循环时间监视
- 在启动模式下，可以更改 HSC（High-Speed Counter，高速计数器）、PWM（Pulse-Width Modulation，脉冲宽度调制）以及 PtP（Point-to-Point communication，点对点通信）模块的组态
- 只有在 RUN 模式下才会真正运行 HSC、PWM 和点对点通信模块

执行完启动 OB 后，CPU 将进入 RUN 模式并在连续的扫描周期内处理控制任务。

5.1.2 在 RUN 模式下处理扫描周期

在每个扫描周期中，CPU 都会写入输出、读取输入、执行用户程序、更新通信模块以及响应用户中断事件和通信请求。 在扫描期间会定期处理通信请求。

以上操作（用户中断事件除外）按先后顺序定期进行处理。 对于已启用的用户中断事件，将根据优先级按其发生顺序进行处理。 对于中断事件，如果适用的话，CPU 将读取输入、执行 OB，然后使用关联的过程映像分区 (PIP) 写入输出。

系统要保证扫描周期在一定的时间段内（即最大循环时间）完成；否则将生成时间错误事件。

- 在每个扫描周期的开始，从过程映像重新获取数字量及模拟量输出的当前值，然后将其写入到 CPU、SB 和 SM 模块上组态为自动 I/O 更新（默认组态）的物理输出。 通过指令访问物理输出时，输出过程映像和物理输出本身都将被更新。
- 随后在该扫描周期中，将读取 CPU、SB 和 SM 模块上组态为自动 I/O 更新（默认组态）的数字量及模拟量输入的当前值，然后将这些值写入过程映像。 通过指令访问物理输入时，指令将访问物理输入的值，但输入过程映像不会更新。
- 读取输入后，系统将从第一条指令开始执行用户程序，一直执行到最后一条指令。 其中包括所有的程序循环 OB 及其所有关联的 FC 和 FB。 程序循环 OB 根据 OB 编号依次执行，OB 编号最小的先执行。

在扫描期间会定期处理通信请求，这可能会中断用户程序的执行。

自诊断检查包括定期检查系统和检查 I/O 模块的状态。

中断可能发生在扫描周期的任何阶段，并且由事件驱动。 事件发生时，CPU 将中断扫描循环，并调用被组态用于处理该事件的 OB。 OB 处理完该事件后，CPU 从中断点继续执行用户程序。

5.1.3 组织块 (OB)

OB 控制用户程序的执行。CPU 中的特定事件将触发组织块的执行。OB 无法互相调用。FC 或 FB 不能调用 OB。只有发生诊断中断或时间间隔这类事件才能启动 OB 的执行。CPU 按照 OB 对应的优先级对其进行处理，遵从高优先级在前低优先级在后的顺序执行 OB。最低优先等级为 1（对应主程序循环），最高优先等级为 26。

5.1.3.1 程序循环 OB

程序循环 OB 在 CPU 处于 RUN 模式时循环执行。主程序块是一种程序循环 OB。您可在此处放置控制程序的说明和调用其他用户块。您可以拥有多个程序循环 OB，CPU 将按编号顺序执行这些 OB。主 (OB 1) 是默认程序循环。

程序循环事件

程序循环事件在每个程序循环（扫描）期间发生一次在程序循环期间，CPU 写入输出、读取输入和执行程序循环 OB。程序循环事件是必需的，并且一直启用。可以不为程序循环事件选择任何程序循环 OB，也可以选择多个 OB。程序循环事件发生后，CPU 将执行编号最小的程序循环 OB（通常为“Main”OB 1）。在程序循环中，CPU 会依次（按编号顺序）执行其它程序循环 OB。程序循环执行，因此将在以下时刻发生程序循环事件：

- 上一个启动 OB 执行结束
- 上一个程序循环 OB 执行结束

表格 5-2　程序循环 OB 的起始信息

输入	数据类型	说明
Initial_Call	Bool	初始调用 OB 时为“True”
Remanence	Bool	保持性数据可用时为“True”

5.1.3.2 启动 OB

启动 OB 在 CPU 的操作模式从 STOP 切换到 RUN 时执行一次，包括处于 RUN 模式时和执行 STOP 到 RUN 切换命令时上电。 之后将开始执行主 “程序循环”OB。

启动事件

启动事件在从 STOP 切换到 RUN 模式时发生一次，并触发 CPU 执行启动 OB。 可为启动事件组态多个 OB。 启动 OB 按编号顺序执行。

表格 5-3　启动 OB 的起始信息

输入	数据类型	说明
LostRetentive	Bool	保持性数据丢失时为“True”
LostRTC	Bool	日期和时间丢失时为“True”

5.1.3.3 延时中断 OB

延时中断 OB 在组态的时延后执行。

延时中断事件

将延时中断事件组态为在经过一个指定的延时后发生。 延迟时间可通过 SRT_DINT 指令分配。 延时事件将中断程序循环以执行相应的延时中断 OB。只能将一个延时中断 OB 连接到一个延时事件。 CPU 支持四个延时事件。

表格 5-4　延时中断 OB 的启动信息

输入	数据类型	说明
Sign	Word	传递给 SRT_DINT 调用触发的标识符

5.1.3.4 循环中断 OB

循环中断 OB 以指定的时间间隔执行。最多可组态四个循环中断事件，每个循环中断事件对应一个 OB。

循环中断事件

用户可通过循环中断事件组态中断 OB 在组态的周期时间执行。创建循环中断 OB 时即可组态初始周期时间。循环事件负责中断程序循环并执行相应的循环中断 OB。请注意，循环中断事件的优先级比程序循环事件更高。

一个循环事件只可连接一个循环中断 OB。

可为每一个循环中断分配一个相移，从而使循环中断彼此错开一定的相移量执行。例如，如果有 5 ms 的循环事件和 10 ms 的循环事件，并且这两个事件每 10 毫秒同时发生一次。如果将 5 ms 的事件相移 1 到 4 ms，将 10 ms 的事件相移 0 ms，则这两个事件不再会同时发生。

默认相位偏移为 0。要更改初始相移，或更改循环事件的循环时间，请执行以下步骤：

1. 在项目树中右键单击循环中断 OB。
2. 从上下文菜单中选择“属性”(Properties)。
3. 单击“循环中断 [OB 30]”(Cyclic interrupt [OB 30]) 对话框中的“循环中断”(Cyclic interrupt)，然后输入新的初始值。

最大相移为 6000 ms（6 秒）或为最大循环时间，选择两者中的较小者。

还可以用 Query 循环中断 (QRY_CINT) 和 Set 循环中断 (SET_CINT) 指令在程序中查询并更改扫描时间和相移。SET_CINT 指令设置的扫描时间和相移不会在上电循环或切换到 STOP 模式的过程中保持不变；扫描时间和相移值会在上电循环或切换到 STOP 模式后重新变为初始值。CPU 共支持四个循环中断事件。

5.1.3.5 硬件中断 OB

硬件中断 OB 在发生相关硬件事件时执行。硬件中断 OB 将中断正常的循环程序执行来响应硬件事件信号。

硬件中断事件

硬件发生变化时将触发硬件中断事件，例如输入点上的上升沿/下降沿事件或者 HSC（High Speed Counter，高速计数器）事件。S7-1200 支持为每个硬件中断事件使用一个中断 OB。可在设备组态中启用硬件事件，并在设备组态中为事件分配 OB，也可在用户程序中通过 ATTACH 指令进行分配。CPU 支持多个硬件中断事件。具体的可用事件由 CPU 型号和输入点数决定。

硬件中断事件数具有以下限制：

沿：

- 上升沿事件：最多 16 条
- 下降沿事件：最多 16 条

HSC 事件：

- CV=PV：最多 6 个
- 方向更改：最多 6 条
- 外部复位：最多 6 条

表格 5-5　硬件中断 OB 的启动信息

输入	数据类型	说明
LADDR	HW_IO	触发硬件中断的模块的硬件标识符。
USI	WORD	用户结构标识符（16#0001 至 16#FFFF），保留供以后使用
IChannel	USINT	触发中断的通道的编号
EventType	BYTE	与触发中断的事件相关的模块特定事件类型的标识符，例如下降沿或上升沿。

EventType 中的位取决于如下触发模块：

模块/子模块	值	过程事件
CPU 或 SB 的 板载 I/O	16#0	上升沿
	16#1	下降沿

模块/子模块	值	过程事件
HSC	16#0	HSC CV=RV1
	16#1	HSC 方向已更改
	16#2	HSC 复位
	16#3	HSC CV=RV2

5.1.3.6 时间错误中断 OB

如已组态，那么当扫描周期超过最大周期时间或发生时间错误事件时，将执行时间错误中断 OB (OB 80)。如已触发，错误中断将中断正常的循环程序执行或其它任何事件 OB。

发生任何上述事件都将生成一个描述相应事件的诊断缓冲区条目。无论是否存在时间错误中断 OB，都将生成诊断缓冲区条目。

时间错误中断事件

出现几种不同时间错误情况中的任何一种都会引起时间错误事件：

- 扫描周期超过最大周期时间
  如果程序循环在指定的最大扫描周期时间内未完成，就会出现“超出最大周期时间”这种情况。有关最大周期时间条件、如何组态 CPU 属性中的最大扫描周期时间以及如何重置周期定时器的更多信息，请参见“监视和组态周期时间 (页 103)”部分。
- 由于在 CPU 结束执行第一次中断 OB 前又启动了第二次中断（循环或延时），因此 CPU 无法启动所请求的 OB。
- 发生队列溢出
  如果中断的出现频率超过 CPU 的处理频率，就会出现“发生队列溢出”这种情况。CPU 通过不同的队列对各种事件类型的未决（排队的）事件数量加以限制。如果相应队列已满时发生某一事件，那么 CPU 将生成一个时间错误事件。

所有时间错误事件都会触发时间错误中断 OB（如果存在）的执行。如果不存在时间错误中断 OB，则 CPU 更改为 STOP 模式。

通过执行 RE_TRIGR 指令 (页 313)重启周期时间监视，用户程序可将程序循环执行时间最多延长为所组态最大周期时间的十倍。但是，如果在同一程序循环中出现两次“超出最大周期时间”情况且没有复位循环定时器，则无论时间错误中断 OB 是否存在，CPU 都将切换到 STOP 模式。请参见“S7-1200 系统手册中的监视循环时间 (页 103)”部分。

时间错误中断 OB 包含的启动信息可帮助您确定生成时间错误的事件和 OB。可以在 OB 中编写指令，以检查这些启动值并采取适当的措施。

表格 5-6　时间错误 OB (OB 80) 的启动信息

输入	数据类型	说明
fault_id	BYTE	16#01 - 超出最大循环时间 16#02 - 请求的 OB 无法启动 16#07 和 16#09 - 发生队列溢出
csg_OBnr	OB_ANY	出错时正在执行的 OB 的编号
csg_prio	UINT	导致错误的 OB 的优先级

要在项目中包括时间错误中断 OB，请在树形结构的“程序块”(Program blocks) 下双击“添加新块”(Add new block)，然后依次选择“组织块”(Organization block)、“时间错误中断”(Time error interrupt)，将时间错误中断添加到项目中。

新 V4.0 CPU 的优先级为 22。如果将 V3.0 CPU 更换为 V4.0 CPU (页 1667)，则优先级为 26，即对 V3.0 有效的优先级。无论哪种情况，优先级字段都可以编辑，用户可以将优先级设置为 22 到 26 之间的任何值。

5.1.3.7 诊断错误中断 OB

当 CPU 检测到诊断错误，或者具有诊断功能的模块发现错误且为该模块启用了诊断错误中断时，将执行诊断错误中断 OB。诊断错误中断 OB 将中断正常的循环程序执行。如果希望 CPU 在收到诊断错误后进入 STOP 模式，可在诊断错误中断 OB 中包含一个 STP 指令，以使 CPU 进入 STOP 模式。

如果未在程序中包含诊断错误中断 OB，CPU 将忽略此类错误并保持 RUN 模式。

诊断错误事件

模拟（本地）、PROFINET、PROFIBUS 和其它一些数字（本地）设备都能够检测并报告诊断错误。发生或清除几种不同诊断错误情况中的任何一种都会引起诊断错误事件。所支持的诊断错误有以下几种：

- 无用户电源
- 超出上限
- 超出下限
- 断路
- 短路

如果存在诊断错误中断 OB (OB 82)，那么诊断错误事件将触发中断执行。如果不存在，CPU 将忽略该错误。

要在项目中包括诊断错误中断 OB，请在树形结构的“程序块”(Program blocks) 下双击“添加新块”(Add new block)，然后依次选择“组织块”(Organization block)、“诊断错误中断”(Diagnostic error interrupt)，将诊断错误中断添加到项目中。

说明

多通道本地模拟设备（I/O、RTD 和热电偶）的诊断错误

诊断错误中断 OB 一次只能处理一个通道的诊断错误。

如果多通道设备的两个通道出现错误，则第二个错误只会在以下情况下触发诊断错误中断 OB：第一个通道错误已清除，由第一个错误触发的诊断错误中断 OB 已执行完毕，并且第二个错误仍然存在。

诊断错误中断 OB 包含的启动信息可帮助您确定事件发生原因是错误的出现还是清除所致，以及确定报告错误的设备和通道。可以在诊断错误中断 OB 中编写指令，以检查这些启动值并采取适当的措施。

说明

如果没有未决诊断事件，诊断错误 OB 启动信息会将子模块作为一个整体来参考

在 V3.0 中，诊断错误离去事件的启动信息始终指示事件源。在 V4.0 中，如果离去事件离开子模块时无未决诊断，启动信息将完全参考子模块 (16#8000)，即使事件源为特定通道。

例如，如果断路触发了通道 2 上的诊断错误事件，纠正故障后清除诊断错误事件，启动信息将不参考通道 2，而是参考子模块 (16#8000)。

表格 5-7　诊断错误中断 OB 的启动信息

输入	数据类型	说明
IOstate	WORD	设备的 IO 状态： ● 如果组态正确，则位 0 = 1，如果组态不再正确，则 = 0。 ● 如果出现错误（如断线），则位 4 = 1。（如果没有错误，则位 4 = 0。） ● 如果组态不正确，则位 5 = 1，如果组态再次正确，则 = 0。 ● 如果发生了 I/O 访问错误，则位 7 = 1。有关存在访问错误的 I/O 的硬件标识符，请参见 LADDR。（如果没有错误，则位 6 = 0。）
LADDR	HW_ANY	报告错误的设备或功能单元的硬件标识符 [1]

输入	数据类型	说明
Channel	UINT	通道号
MultiError	BOOL	如果存在多个错误，参数值为 TRUE

[1] LADDR 输入包含返回错误的设备或功能单元的硬件标识符。硬件标识符是在设备或网络视图中插入组件时自动分配的，它出现在 PLC 变量的 “常量”(Constants) 选项卡中。还会自动为硬件标识符分配名称。不能更改这些 PLC 变量的 “常量”(Constants) 选项卡中的条目。

5.1.3.8 拔出或插入模块 OB

当已组态和非禁用分布式 I/O 模块或子模块（PROFIBUS、PROFINET、AS-i）生成插入或拔出模块相关事件时，系统将执行 “拔出或插入模块”OB。

拔出或插入模块事件

以下情况将产生拔出或插入模块事件：

- 有人拔出或插入一个已组态的模块
- 扩展机架中实际并没有所组态的模块
- 扩展机架中的不兼容模块与所组态的模块不相符
- 扩展机架中插入了与所组态模块兼容的模块，但组态不允许替换值
- 模块或子模块发生参数化错误

如果尚未对该 OB 进行编程，那么当已组态且未禁用的分布式 I/O 模块以上任意情况时，CPU 都将保持在 RUN 模式。

无论是否已对该 OB 进行编程，当中央机架中的模块以上任意情况时，CPU 都将切换到 STOP 模式。

表格 5-8 拔出或插入模块 OB 的启动信息

输入	数据类型	说明
LADDR	HW_IO	硬件标识符
Event_Class	Byte	16#38: 模块已插入 16#29： 模块已拔出
Fault_ID	Byte	故障标识符

5.1.3.9 机架或站故障 OB

当 CPU 检测到分布式机架或站出现故障或发生通信丢失时，将执行 “机架或站故障”OB。

机架或站故障事件

检测到以下任一情况时，CPU 将生成机架或站故障事件：

- DP 主站系统故障或 PROFINET IO 系统故障（进入或离开事件）
- DP 从站系统故障或 IO 设备故障（进入或离开事件）
- PROFINET I 设备的某些子模块发生故障

如果尚未对该 OB 进行编程，那么发生以上任意情况时，CPU 将保持在 RUN 模式。

表格 5-9　机架或站故障 OB 的启动信息

输入	数据类型	说明
LADDR	HW_IO	硬件标识符
Event_Class	Byte	16#38：离开事件 16#39：进入事件
Fault_ID	Byte	故障标识符

5.1.3.10 时钟 OB

时钟 OB 根据所组态的时钟时间条件执行。 CPU 支持两个时钟 OB。

时钟事件

可将时钟中断事件组态为在某个指定的日期或时间发生一次，或者按照以下周期之一循环发生：

- 每分钟： 每分钟发生中断。
- 每小时： 每小时发生中断。
- 每天： 在每天的指定时间（小时和分钟）发生中断。
- 每周： 在每周指定日期的指定时间（例如，每周二下午 4:30）发生中断。
- 每月： 在每月指定日期的指定时间发生中断。 日期编号必须介于 1 和 28 之间（包括 1 和 28）。
- 每个月末： 在每个月最后一天的指定时间发生中断。
- 每年： 在每年的指定日期（月和日）发生中断。 不能指定 2 月 29 日。

表格 5-10　时钟事件 OB 的启动信息

输入	数据类型	说明
CaughtUp	Bool	已向前设置时间，因此满足 OB 调用
SecondTimes	Bool	已向后设置时间，因此第二次启动 OB 调用

5.1.3.11 状态 OB

如果 DPV1 或 PNIO 从站触发状态中断，则执行状态 OB。 如果 DPV1 或 PNIO 从站的组件（模块或机架）更改了其工作模式（例如由 RUN 变为 STOP），则可能发生这种情况。

状态事件

有关可触发状态中断的事件的详细信息，请参见 DPV1 或 PNIO 从站的制造商文档。

表格 5-11　状态 OB 的启动信息

输入	数据类型	说明
LADDR	HW_IO	硬件标识符
Slot	UInt	插槽号
Specifier	Word	报警说明符

5.1.3.12 更新 OB

如果 DPV1 或 PNIO 从站触发更新中断，则执行更新 OB。

更新事件

有关可触发更新中断的事件的详细信息，请参见 DPV1 或 PNIO 从站的制造商文档。

表格 5-12　更新 OB 的启动信息

输入	数据类型	说明
LADDR	HW_IO	硬件标识符
Slot	UInt	插槽号
Specifier	Word	报警说明符

5.1.3.13 配置文件 OB

如果 DPV1 或 PNIO 从站触发配置文件特定的中断，则执行配置文件 OB。

配置文件事件

有关可触发配置文件中断的事件的详细信息，请参见 DPV1 或 PNIO 从站的制造商文档。

表格 5-13 配置文件 OB 的启动信息

输入	数据类型	说明
LADDR	HW_IO	硬件标识符
Slot	UInt	插槽号
Specifier	Word	报警说明符

5.1.3.14 MC 伺服和 MC 插补器 OB

在创建运动工艺对象并将驱动器接口设置为 “模拟驱动器接口”(Analog drive connection) 或 “PROFIDrive”时，STEP 7 会自动创建只读 MC 伺服和 MC 插补器 OB。用户无需编辑任何 OB 属性，也无需直接创建此 OB。CPU 将这些 OB 用于闭环控制。有关更多详细信息，请参见 STEP 7 信息系统。

5.1.3.15 MC-PreServo

可以对 MC-PreServo OB 进行编程，使其包含程序逻辑：在 MC-Servo OB 执行前直接执行 STEP 7 程序。

MC-PreServo 事件

MC-PreServo OB 使您可读取所组态的应用周期信息（单位为 μs）。

表格 5-14 MC-PreServo OB 的起始信息

输入	数据类型	说明
Initial_Call	BOOL	TRUE 表示从 STOP 转为 RUN 的过程中首次调用该 OB
PIP_Input	BOOL	TRUE 表示相关的过程映像输入为最新值。
PIP_Output	BOOL	TRUE 表示在最后一个周期后，CPU 将相关的过程映像输出适时传送到输出中。
IO_System	USINT	触发中断的分布式 I/O 系统的编号

输入	数据类型	说明
Event_Count	INT	n：丢失的循环数 -1：丢失的循环数未知（例如，由于更改了循环）
Synchronous	BOOL	预留
CycleTime	UDINT	显示为 MC-Servo OB 组态的应用周期，单位为 µs

5.1.3.16 MC-PostServo

可以对 MC-PreServo OB 进行编程，使其包含程序逻辑：在 MC-Servo OB 执行后直接执行 STEP 7 程序。

MC-PostServo 事件

MC-PreServo OB 使您可读取所组态的应用周期信息（单位为 µs）。

表格 5-15 MC-PostServo OB 的起始信息

输入	数据类型	说明
Initial_Call	BOOL	TRUE 表示从 STOP 转为 RUN 的过程中首次调用该 OB
PIP_Input	BOOL	TRUE 表示相关的过程映像输入为最新值。
PIP_Output	BOOL	TRUE 表示在最后一个周期后，CPU 将相关的过程映像输出适时传送到输出中。
IO_System	USINT	触发中断的分布式 I/O 系统的编号
Event_Count	INT	n：丢失的循环数 -1：丢失的循环数未知（例如，由于更改了循环）
Synchronous	BOOL	预留
CycleTime	UDINT	显示为 MC-Servo OB 组态的应用周期，单位为 µs

5.1.3.17 事件执行的优先级与排队

CPU 处理操作受事件控制。事件会触发要执行的中断 OB。可以在块的创建期间、设备配置期间或使用 ATTACH 或 DETACH 指令指定事件的中断 OB。有些事件定期发生，例如，程序循环或循环事件。而其它事件只发生一次，例如，启动事件和延时事件。还有一些事件则在硬件触发事件时发生，例如，输入点上的沿事件或高速计数器事件。诊断错误和时间错误等事件只在出现错误时发生。事件优先级和队列用于确定事件中断 OB 的处理顺序。

CPU 按照优先级顺序处理事件，1 为最低优先级，26 为最高优先级。在 S7-1200 CPU V4.0 之前的版本中，每种 OB 类型都有固定的优先级（1 到 26）。从 V4.0 开始，可为每个组态的 OB 分配优先级。优先级编号在 OB 属性的特性中进行配置。

可中断与不可中断执行模式

OB (页 88) 按照其触发事件的优先级顺序执行。在 CPU 设备组态的启动属性 (页 163)中，您可以将 OB 执行组态为可中断或不可中断。请注意，程序循环 OB 始终为可中断，但可将其它所有 OB 组态为可中断或不可中断。

如果设置了可中断模式，则在执行 OB 并且 OB 执行结束前发生了更高优先级的事件时，将中断正在运行的 OB，以允许更高优先级的事件 OB 运行。运行更高级别的事件直至结束后，才会继续执行之前中断的 OB。如果执行可中断 OB 时发生多个事件，CPU 将按照优先级顺序处理这些事件。

如果未设置可中断模式，则无论触发的 OB 在运行期间是否触发了其它任何事件，都将继续运行直至结束。

考虑以下两种情况，其中中断事件触发循环 OB 和延时 OB。在这两种情况中，延时 OB (OB 201) 没有过程映像分区分配 (页 81)并且以优先级 4 执行。循环 OB (OB 200) 分配了 PIP1 过程映像分区并且以优先级 2 执行。下图显示了不可中断执行模式与可中断执行模式下执行 OB 的区别：

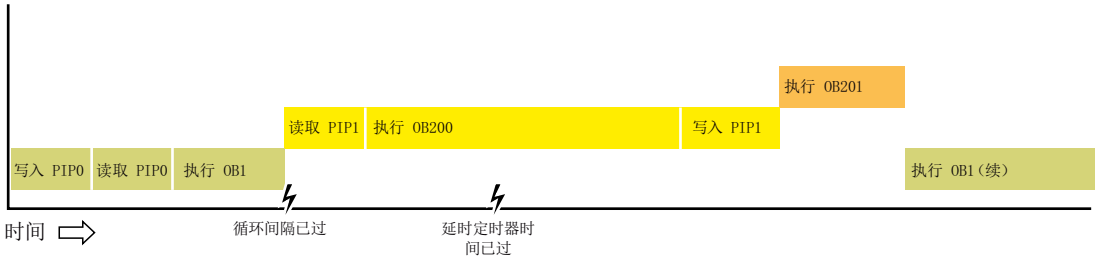


图 5-1　情况 1：不可中断 OB 执行

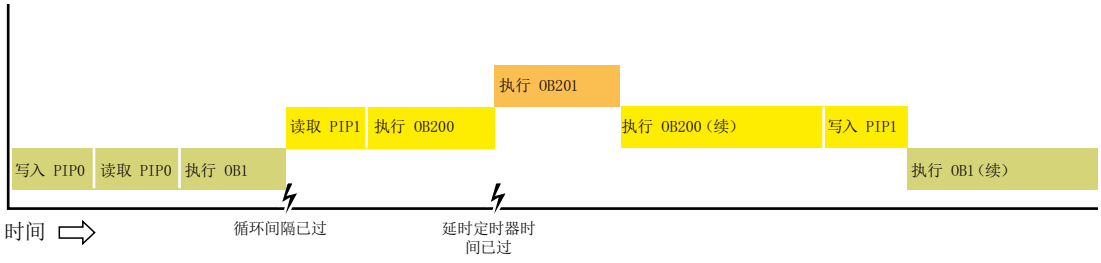


图 5-2　情况 2：可中断 OB 执行

说明

如果将 OB 执行模式组态为不可中断，则时间错误 OB 不能中断除程序循环 OB 以外的 OB。在 S7-1200 CPU V4.0 之前的版本中，时间错误 OB 可中断任何正在执行的 OB。而从 V4.0 开始，如果希望时间错误 OB（或其它任何优先级更高的 OB）能够中断除程序循环 OB 以外的执行中 OB，必须将 OB 执行组态为可中断。

了解事件执行的优先级与排队

CPU 通过各种事件类型的不同队列限制单一来源的未决（排队的）事件数量。达到给定事件类型的未决事件限值后，下一个事件将丢失。可以使用时间错误中断 OB (页 92) 响应队列溢出。

请注意，STEP 7 可用于组态循环中断 OB 和时间 OB 的一些特定事件队列参数。

有关 CPU 过载行为和事件排队的更多详细信息，请参见 STEP 7 信息系统。

每个 CPU 事件都具有相关优先级。通常，CPU 按优先级顺序处理事件（优先级最高的最先进行处理）。对于优先级相同的事件，CPU 按照“先到先得”的原则进行处理。

表格 5-16　OB 事件

事件	允许的数量	默认 OB 优先级
程序循环	1 个程序循环事件 允许多个 OB	1[1]
启动	1 个启动事件 [1] 允许多个 OB	1[1]
延时	最多 4 个时间事件 每个事件 1 个 OB	OB 20：3 OB 21：4 OB 22：5 OB 23：6 OB 123 至 OB 32767：3

事件	允许的数量	默认 OB 优先级
循环中断	最多 4 个事件 每个事件 1 个 OB	OB 30：8 OB 31：9 OB 32：10 OB 33：11 OB 34：12 OB 35：13 OB 36：14 OB 37：16 OB 38：17 OB 123 至 OB 32767：7
硬件中断	最多 50 个硬件中断事件 [2]	18
	每个事件 1 个 OB，但可对多个事件使用同一个 OB	18
时间错误	1 个事件（仅当组态时）[3]	22 或 26[4]
诊断错误	1 个事件（仅当组态时）	5
拔出或插入模块	1 个事件	6
机架或站故障	1 个事件	6
日时钟	最多 2 个事件	2
状态	1 个事件	4
更新	1 个事件	4
配置文件	1 个事件	4
MC 伺服	1 个事件	25
MC 插补器	1 个事件	24

[1] 启动事件和程序循环事件不会同时发生，因为启动事件运行结束后程序循环事件才启动。

[2] 如果使用 DETACH 和 ATTACH 指令，则可具有 50 个以上的硬件中断事件 OB。

[3] 可以将 CPU 组态为在超出最大扫描周期时间时保持 RUN 模式，也可使用 RE_TRIGR 指令复位周期时间。但是，如果同一个扫描周期第二次超出最大扫描周期时间，CPU 就会进入 STOP 模式。

[4] 新 V4.0 或 V4.1 CPU 的优先级为 22。如果是将 V3.0 CPU 更换为 V4.0 或 V4.1 CPU，则优先级为 26：即对 V3.0 有效的优先级。无论哪种情况，优先级字段都可以编辑，用户可以将优先级设置为 22 到 26 之间的任何值。

有关详细信息，请参见主题“用 V4.x CPU 更换 V3.0 CPU (页 1667)”。

另外，CPU 可识别出无关联 OB 的其它事件。下表介绍了这些事件和相应的 CPU 操作：

表格 5-17　附加事件

事件	说明	CPU 操作
I/O 访问错误	直接 I/O 读/写错误	CPU 将第一次错误记录在诊断缓冲区中并保持 RUN 模式。您可以使用 GET_ERROR_ID (页 314) 指令访问错误原因。
最大周期时间错误	CPU 超出组态的周期时间两次	CPU 将错误记录在诊断缓冲区中并切换为 STOP 模式。
外围设备访问错误	过程映像更新期间出现 I/O 错误	CPU 将第一次错误记录在诊断缓冲区中并保持 RUN 模式。
编程错误	程序执行错误	• 如果启用了错误处理，系统会在错误结构中输入错误原因。您可以使用 GET_ERROR_ID (页 314) 指令访问错误原因。 • 如果启用了全局错误处理，系统将在诊断缓冲区中输入访问错误启动事件，并保持 RUN 模式。

中断等待时间

如果中断事件发生时程序循环 OB 是唯一激活的事件服务例程，则中断事件等待时间（该时间是指从通知 CPU 发生了事件到 CPU 开始执行处理该事件的 OB 中的第一条指令）约为 175 µs。

5.1.4　监视和组态循环时间

循环时间是指 CPU 操作系统在 RUN 模式下执行循环阶段所需的时间。CPU 提供了两种监视循环时间的方法：

- 最大扫描周期时间
- 最小扫描周期时间

扫描周期监视在启动事件完成后开始。此功能的组态出现在 CPU“设备配置”(Device Configuration) 的 “循环时间”(Cycle time) 下。

CPU 监视扫描周期，并在扫描周期时间超出组态的最大扫描周期时间时做出响应。如果扫描周期时间超出组态的最大扫描周期时间，则 CPU 会生成错误并做出如下响应：

- 如果用户程序中包含时间错误中断 OB (页 92)，则 CPU 将执行该中断。
- 如果用户程序不包含时间错误中断 OB，则时间错误事件将生成一个诊断缓冲区条目。CPU 进入 STOP 模式。

RE_TRIGR 指令 (页 313)（重新触发周期时间监视）可用于复位记录周期时间的定时器。如果当前程序循环执行耗费的时间小于所组态最大扫描周期时间的十倍，则 RE_TRIGR 指令将重新触发周期时间监视并返回“ENO = TRUE”。否则 RE_TRIGR 指令将不会重新触发周期时间监视，并返回“ENO = FALSE”。

通常，扫描周期会尽快执行，当前扫描周期一完成，下一个扫描周期就会开始。视用户程序和通信任务而定，扫描周期的时间段在各次扫描中有所不同。为了消除这种差异，CPU 支持一种可选的最小扫描周期时间。如果启用此可选功能并提供以 ms 为单位的最小扫描周期时间，则在执行完程序循环 OB 后 CPU 会延时，直至经过最小扫描周期时间后才重复程序循环。

如果 CPU 完成正常扫描周期的时间小于指定的最小循环时间，则 CPU 将用额外的扫描周期时间执行运行诊断和/或处理通信请求。

如果 CPU 在指定的最小循环时间内未完成扫描周期，CPU 将正常完成扫描（包括通信处理），并且不会因超出最小扫描时间而引起任何系统响应。下表定义了循环时间监视功能的范围和默认值：

表格 5-18　循环时间的范围

循环时间	值范围 (ms)	默认值
最大扫描周期时间 [1]	1 到 6000	150 ms
最小扫描周期时间 [2]	1 到最大扫描周期时间	禁用

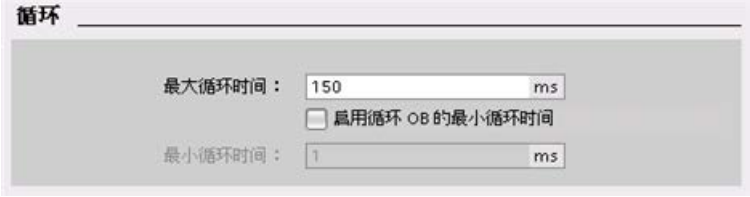
[1] 最大扫描周期时间始终启用。组态循环时间使其介于 1 ms 到 6000 ms 之间。默认值为 150 ms。

[2] 最小扫描周期时间为可选项，默认情况下被禁用。必要时，可组态一个 1 ms 到最大扫描周期时间之间的周期时间。

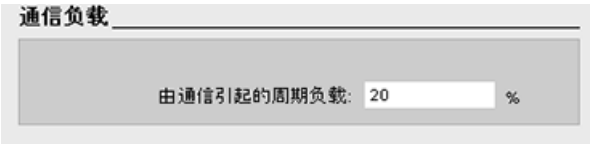
组态循环时间和通信负载

利用设备配置中的 CPU 属性可以组态以下参数：

- 周期：可输入最大扫描周期监视时间。也可启用并输入最小扫描周期时间。



- 通信负载：可以组态一个百分比时间，专门用于通信任务。



说明

通信优先级

通信任务的优先级为 1。因为 1 是最低优先级，所以其他 CPU 事件可以中断通信处理。其他事件导致的中断可能会对扫描周期的通信处理产生负面影响。可以调整“通信造成的周期负载”百分比，以增加专用于通信处理的扫描周期部分。

有关扫描周期的更多信息，请参见监视循环时间 (页 103)。

5.1.5 CPU 存储器

存储器管理

CPU 提供了以下用于存储用户程序、数据和组态的存储区：

- 装载存储器，用于非易失性地存储用户程序、数据和组态。将项目下载到 CPU 后，CPU 会先将程序存储在装载存储区中。该存储区位于存储卡（如存在）或 CPU 中。CPU 能够在断电后继续保持该非易失性存储区。存储卡支持的存储空间比 CPU 内置的存储空间更大。
- 工作存储器是易失性存储器，用于在执行用户程序时存储用户项目的某些内容。CPU 会将一些项目内容从装载存储器复制到工作存储器中。该易失性存储区将在断电后丢失，而在恢复供电时由 CPU 恢复。
- 保持性存储器，用于非易失性地存储限量的工作存储器值。断电过程中，CPU 使用保持性存储区存储所选用户存储单元的值。如果发生断电或掉电，CPU 将在上电时恢复这些保持性值。

要显示编译程序块的存储器使用情况，请右键单击 STEP 7 项目树中“程序块”(Program blocks) 文件夹中的块，然后从上下文菜单中选择“资源”(Resources)。“编译属性” (Compiliation properties) 显示了编译块的装载存储器和工作存储器。

要显示在线 CPU 的存储器使用情况，请双击 STEP 7 中的“在线和诊断”(Online and diagnostics)，展开“诊断”(Diagnostics)，然后选择“存储器”(Memory)。

保持性存储器

将某些数据标记为保持性数据可以避免发生电源故障后造成数据丢失。该 CPU 允许您将以下数据配置为保持性数据：

- 位存储器 (M)：可以在 PLC 变量表或分配列表中定义位存储器的保持性存储器的大小。保持性位存储器总是从 MB0 开始向上连续贯穿指定的字节数。通过 PLC 变量表或在分配列表中通过单击“保持性”(Retain) 工具栏图标指定该值。输入从 MB0 开始保留的 M 字节个数。
  注意：对于任何块，都可通过在“程序块”(Program blocks) 文件夹中选择块，然后选择**“工具 > 分配列表”(Tools > Assignment list)** 菜单命令来显示分配列表。
- 函数块 (FB) 的变量：如果 FB 为“优化块访问”(Optimized block access) 类型，则该 FB 的接口编辑器将包含“保持”(Retain) 列。在该列中，可以单独为每个变量选择“保持”(Retain)、“非保持”(Non-retain) 或“在 IDB 中设置”(Set in IDB)。将此类 FB 置于程序中时，和该 FB 对应的实例 DB 也将包含此“保持”(Retain) 列。在优化的 FB 中，如果在变量的“保持性”(Retain) 选项中选择“在 IDB 中设置”(Set in IDB)（在背景数据块中设置），那么只能更改背景 DB 接口编辑器中某个变量的保持性状态。
  如果 FB 非“优化块访问”(Optimized block access) 类型，则该 FB 的接口编辑器将不包含“保持”(Retain) 列。将此类 FB 置于程序中时，和该 FB 对应的实例 DB 仍将包含一个可进行编辑的“保持”(Retain) 列。如果是这种情况，在选择所有变量时为任意变量结果选择“保持”(Retain) 选项。同样，在取消选择所有变量时为任意变量结果取消选择该选项。要查看或修改 FB 是否已优化，打开 FB 属性然后选则属性。
- 全局数据块的变量：在保持性状态分配方面，全局 DB 与 FB 类似。根据块访问设置情况，用户可以定义全局数据块的单个变量或所有变量的保持性状态。
  - 如果在 DB 创建时选择“优化”(Optimized)，则可以设置每个单独变量的保持性状态。
  - 如果在创建 DB 时选择“标准 - 与 S7-300/400 兼容”(Standard - compatible with S7-300/400)，则该保持性状态的设置将适用于该 DB 的所有变量；即变量要么都具有保持性，要么都没有。

该 CPU 最多支持 10240 字节的保持性数据。要了解可用保持性字节数，请在 PLC 变量表或分配列表中单击“保持性”(Retain) 工具栏图标。尽管这里是为 M 存储器指定保持性范围的地方，但第二个箭头会指示可用于 M 和 DB 的总剩余存储空间。请注意，要保证该值的准确性，必须编译带有保持性变量的所有数据块。

说明

下载程序不会清除或更改保持性存储器中的现有值。如果要在下载之前清除保持性存储器，请在下载程序前将 CPU 复位为出厂设定。

5.1.5.1 系统和时钟存储器

使用 CPU 属性可启用“系统存储器”和“时钟存储器”的相应字节。 程序逻辑可通过这些函数的变量名称来引用它们的各个位。

- 可以将 M 存储器的一个字节分配给系统存储器。 该系统存储器字节提供了以下四个位，用户程序可通过以下变量名称引用这四个位：
  - 第一个周期：（变量名称“FirstScan”）在启动 OB 完成后的第一次扫描期间内，该位设置为 1。（执行了第一次扫描后，“首次扫描”位将设置为 0。）
  - 诊断状态变化：（变量名称：“DiagStatusUpdate”）在 CPU 记录了诊断事件后的一个扫描周期内设置为 1。 由于直到首次程序循环 OB 执行结束，CPU 才能置位“DiagStatusUpdate”位，因此用户程序无法检测在启动 OB 执行期间或首次程序循环 OB 执行期间是否发生过诊断更改。
  - 始终为 1（高）(Always 1 (high))：（变量名称“AlwaysTRUE”），该位始终设置为 1。
  - 始终为 0（低）(Always 0 (low))：（变量名称“AlwaysFALSE”），该位始终设置为 0。
- 可以将 M 存储器的一个字节分配给时钟存储器。 被组态为时钟存储器的字节中的每一位都可生成方波脉冲。 时钟存储器字节提供了 8 种不同的频率，其范围从 0.5 Hz（慢）到 10 Hz（快）。 这些位可作为控制位（尤其在与沿指令结合使用时），用于在用户程序中周期性触发动作。

CPU 在从 STOP 模式切换到 STARTUP 模式时初始化这些字节。 时钟存储器的位在 STARTUP 和 RUN 模式下会随 CPU 时钟同步变化。

> ⚠ 小心
>
> **覆盖系统存储器位或时钟存储器位时的风险**
>
> 改写系统存储器或时钟存储器的各个位可能会破坏这些功能中的数据，同时还可能导致用户程序错误运行，进而造成设备损坏和人员伤害。
>
> 因为时钟存储器和系统存储器都不是预留的 M 存储器，所以指令或通信可以写入这些单元并破坏其中的数据。
>
> 避免向这些单元写入数据以确保这些功能正常运行，并且应始终为过程或机器使用紧急停止电路。

系统存储器组态了一个字节，其中的各个位会在发生特定事件时启用（值 = 1）。

表格 5-19　系统存储器

7	6	5	4	3	2	1	0
保留 值 0				始终熄灭 值 0	常开 值 1	诊断状态指示 • 1: 变化 • 0: 无更改	首次扫描指示 • 1: 启动后首次扫描 • 0: 不是首次扫描

时钟存储器组态了一个字节，该字节的各个位分别按固定的时间间隔循环启用和禁用。 每个时钟位都会在相应的 M 存储器位产生一个方波脉冲。 这些位可作为控制位（尤其在与沿指令结合使用时），用于在用户代码中周期性触发动作。

表格 5-20　时钟存储器

位号	7	6	5	4	3	2	1	0
变量名称								
周期 (s)	2.0	1.6	1.0	0.8	0.5	0.4	0.2	0.1
频率 (Hz)	0.5	0.625	1	1.25	2	2.5	5	10

由于时钟存储器与 CPU 周期异步运行，因此，时钟存储器的状态可能会在一个长周期中发生多次改变。

5.1.6 诊断缓冲区

CPU 提供了一个诊断缓冲区，其中包含的每个条目对应一个诊断事件。每个条目都包含了事件发生的日期和时间、事件类别及事件描述。条目按时间顺序显示，最新发生的事件位于最上面。此日志最多可提供 50 个最近发生的事件。日志填满后，新事件将替换日志中最早的事件。掉电时，将保存事件。

诊断缓冲区中记录以下事件类型：

- 所有系统诊断事件；例如，CPU 错误和模块错误
- CPU 的每次状态切换（每次上电、每次切换到 STOP 模式、每次切换到 RUN 模式）

必须在线访问诊断缓冲区 (页 1393)。从 “在线和诊断”(Online & diagnostics) 视图中，在 “诊断 > 诊断缓冲区”(Diagnostics > Diagnostics buffer) 下查找诊断缓冲区。

减少安全诊断事件的数量

部分安全事件会在诊断缓冲区中生成重复条目。这些消息可能会堵塞诊断缓冲区，从而可能阻碍其它事件消息。您可以组态 PLC 限定安全事件的诊断消息数量。可以在 CPU 的设备组态（其中可以抑制循环消息）中基于时间间隔进行选择：

如果选择在时间间隔内总结安全事件，您可以将时间间隔的单位设置为秒、分钟或小时，数值范围设置为 1 ...255。

如果选择对安全事件进行限定，将限定以下几种类型的事件：

- 使用正确或错误的密码转至在线状态
- 检测被操控的通信数据
- 检测存储卡上被操控的数据
- 检测被操控的固件更新文件
- 更改后的保护等级（访问保护）下载到 CPU
- 限制或启用密码合法性（通过指令或 CPU 显示器）
- 由于超出允许的并行访问尝试次数，在线访问被拒绝
- 现有在线连接处于禁用状态的超时
- 使用正确或错误的密码登录到 Web 服务器
- 创建 CPU 的备份
- 恢复 CPU 组态

5.1.7 日时钟

CPU 支持日时钟。 在 CPU 断电期间，超级电容器提供时钟继续运行所需的电能。 超级电容器在 CPU 通电时充电。 在 CPU 通电至少 24 小时之后，超级电容器所具有的电量通常足以维持时钟运行 20 天。

STEP 7 将时钟设置为系统时间，它有一个初始的默认值或者遵循出厂值。 若要使用日时钟，必须进行设置。 诸如用于诊断缓冲区条目、数据日志文件和数据日志条目的时间戳都是基于系统时间。 从在线 CPU 的 “在线和诊断”(Online & diagnostics) 视图中的 “设置日时钟” 功能 (页 1389)下设置日时钟。 然后，STEP 7 从您设置的时间中加上或者减去 Windows 操作系统与 UTC（世界协调时间）的偏差来计算系统时间。 如果您的 Windows 操作系统的时区和夏令时的设置与您所处的区域相一致，则将日时钟设置为当前的本地时间会产生 UTC 的系统时间。

STEP 7 中包含读写系统时间（RD_SYS_T 和 WR_SYS_T）、读取本地时间 (RD_LOC_T) 和设置时区 (SET_TIMEZONE) 的指令 (页 340)。 RD_LOC_T 指令使用您在 CPU 的一般属性 (页 163)的 “日时钟”(Time of day) 组态中所设置的时区和夏令时偏移量来计算本地时间。 这些设置可以设置您本地时间的时区、选择性地设置夏令时并指定夏令时的开始时间和结束时间。 您也可以通过使用 SET_TIMEZONE 指令来设定这些设置。

5.1.8 组态从 RUN 切换到 STOP 时的输出

可以组态 CPU 处于 STOP 模式时数字量输出和模拟量输出的特性。 可以将 CPU、SB 或 SM 的任何输出设置为冻结值或使用替换值：

- 替换特定的输出值（默认）： 为 CPU、SB 或 SM 设备的每个输出（通道）分别输入替换值。
  数字输出通道的默认替换值为 OFF，而模拟输出通道的默认替换值为 0。
- 冻结输出以保持上一个状态： 工作模式从 RUN 切换到 STOP 时，输出将保留当前值。 上电后，输出被设置为默认的替换值。

可以在 “设备配置”(Device Configuration) 中组态输出的行为。 选择相应的设备，然后使用 “属性”(Properties) 选项卡组态每个设备的输出。

说明

某些分布式 I/O 模块提供了用于响应 CPU 停止模式的额外设置。 请从这些模块的设备配置中的选项列表中进行选择。

CPU 从 RUN 切换到 STOP 后，CPU 将保留过程映像，并根据组态写入相应的数字和模拟输出值。

5.2 数据存储、存储区、I/O 和寻址

5.2.1 访问 S7-1200 的数据

STEP 7 简化了符号编程。 用户为数据地址创建符号名称或“变量”，作为与存储器地址和 I/O 点相关的 PLC 变量或在代码块中使用的局部变量。 要在用户程序中使用这些变量，只需输入指令参数的变量名称。

为了更好地理解 CPU 的存储区结构及其寻址方式，以下段落将对 PLC 变量所引用的“绝对”寻址进行说明。 CPU 提供了以下几个选项，用于在执行用户程序期间存储数据：

- 全局储存器： CPU 提供了各种专用存储区，其中包括输入 (I)、输出 (Q) 和位存储器 (M)。 所有代码块可以无限制地访问该储存器。
- PLC 变量表： 在 STEP 7 PLC 变量表中，可以输入特定存储单元的符号名称。 这些变量在 STEP 7 程序中为全局变量，并允许用户使用应用程序中有具体含义的名称进行命名。

- 数据块 (DB)： 可在用户程序中加入 DB 以存储代码块的数据。 从相关代码块开始执行一直到结束，存储的数据始终存在。 “全局”DB 存储所有代码块均可使用的数据，而背景 DB 存储特定 FB 的数据并且由 FB 的参数进行构造。
- 临时存储器： 只要调用代码块，CPU 的操作系统就会分配要在执行块期间使用的临时或本地存储器 (L)。 代码块执行完成后，CPU 将重新分配本地存储器，以用于执行其它代码块。

每个存储单元都有唯一的地址。 用户程序利用这些地址访问存储单元中的信息。 对输入 (I) 或输出 (Q) 存储区（例如 I0.3 或 Q1.7）的引用会访问过程映像。 要立即访问物理输入或输出，请在引用后面添加“:P”（例如，I0.3:P、Q1.7:P 或 "Stop:P"）。

表格 5-21　存储区

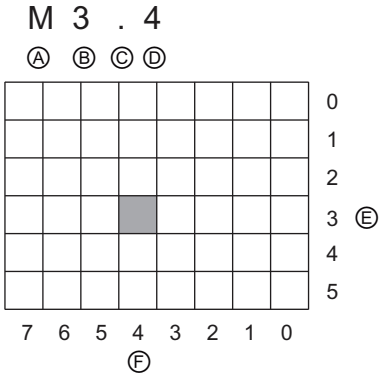
存储区	说明	强制	保持性
I 过程映像输入 I_:P[1] （物理输入）	在扫描周期开始时从物理输入复制	无	无
	立即读取 CPU、SB 和 SM 上的物理输入点	支持	无
Q 过程映像输出 Q_:P[1] （物理输出）	在扫描周期开始时复制到物理输出	无	无
	立即写入 CPU、SB 和 SM 上的物理输出点	支持	无
M 位存储器	控制和数据存储器	无	支持 （可选）
L 临时存储器	存储块的临时数据，这些数据仅在该块的本地范围内有效	无	无
DB 数据块	数据存储器，同时也是 FB 的参数存储器	无	是 （可选）

[1] 要立即访问（读取或写入）物理输入和物理输出，请在地址或变量后面添加“:P”（例如，I0.3:P、Q1.7:P 或“Stop:P”）。

每个存储单元都有唯一的地址。 用户程序利用这些地址访问存储单元中的信息。 绝对地址由以下元素组成：

- 存储区标识符（如 I、Q 或 M）
- 要访问的数据的大小（“B”表示 Byte、“W”表示 Word 或“D”表示 DWord）
- 数据的起始地址（如字节 3 或字 3）

访问布尔值地址中的位时，不要输入大小的助记符号。 仅需输入数据的存储区、字节位置和位位置（如 0.0、Q0.1 或 M3.4）。



A	存储区标识符	E	存储区的字节
B	字节地址： 字节 3	F	选定字节的位
C	分隔符（“字节.位”）		
D	位在字节中的位置（位 4，共 8 位）		

本示例中，存储区和字节地址（M 代表位存储区，3 代表 Byte 3）通过后面的句点（“.”）与位地址（位 4）分隔。

访问 CPU 存储区中的数据

STEP 7 简化了符号编程。通常，可在 PLC 变量表、数据块中创建变量，也可在 OB、FC 或 FB 的接口中创建变量。这些变量包括名称、数据类型、偏移量和注释。此外，在数据块中，还可设定起始值。在编程时，通过在指令参数中输入变量名称，可以使用这些变量。也可以选择在指令参数中输入绝对操作数（存储区、大小和偏移量）。以下各部分的实例介绍了如何输入绝对操作数。程序编辑器会自动在绝对操作数前面插入 % 字符。可以在程序编辑器中将视图切换到以下几种视图之一：符号、符号和绝对，或绝对。

I（过程映像输入）：CPU 仅在每个扫描周期的循环 OB 执行之前对外围（物理）输入点进行采样，并将这些值写入到输入过程映像。可以按位、字节、字或双字访问输入过程映像。允许对过程映像输入进行读写访问，但过程映像输入通常为只读。

表格 5-22　I 存储器的绝对地址

位	I[字节地址].[位地址]	I0.1
字节、字或双字	I[大小][起始字节地址]	IB4、IW5 或 ID12

通过在地址后面添加“:P”，可以立即读取 CPU、SB、SM 或分布式模块的数字量和模拟量输入。使用 I_:P 访问与使用 I 访问的区别是，前者直接从被访问点而非输入过程映像获得数据。

这种 I_:P 访问称为“立即读”访问，因为数据是直接从源而非副本获取的，这里的副本是指在上次更新输入过程映像时建立的副本。

因为物理输入点直接从与其连接的现场设备接收值，所以不允许对这些点进行写访问。即，与可读或可写的 I 访问不同的是，I_:P 访问为只读访问。

I_:P 访问也仅限于单个 CPU、SB 或 SM 所支持的输入大小（向上取整到最接近的字节）。例如，如果将 2 DI/2 DQ SB 的输入组态为从 I4.0 开始，则可按 I4.0:P 和 I4.1:P 或 IB4:P 的形式访问输入点。以 I4.7:P 形式访问 I4.2:P 不会被拒绝，但没有任何意义，因为不会使用这些点。但不允许 IW4:P 和 ID4:P 的访问形式，因为它们超出了与该 SB 相关的字节偏移量。

使用 I_:P 访问不会影响存储在输入过程映像中的相应值。

表格 5-23　I 存储器的绝对地址（立即）

位	I[字节地址].[位地址]:P	I0.1:P
字节、字或双字	I[大小][起始字节地址]:P	IB4:P、IW5:P 或 ID12:P

Q（过程映像输出）：CPU 将存储在输出过程映像中的值复制到物理输出点。可以按位、字节、字或双字访问输出过程映像。过程映像输出允许读访问和写访问。

表格 5-24　Q 存储器的绝对地址

位	Q[字节地址].[位地址]	Q1.1
字节、字或双字	Q[大小][起始字节地址]	QB5、QW10、QD40

通过在地址后面添加“:P”，可以立即写入 CPU、SB、SM 或分布式模块的物理数字量和模拟量输出。使用 Q_:P 访问与使用 Q 访问的区别是，前者除了将数据写入输出过程映像外还直接将数据写入被访问点（写入两个位置）。这种 Q_:P 访问有时称为“立即写”访问，因为数据是被直接发送到目标点；而目标点不必等待输出过程映像的下一次更新。

因为物理输出点直接控制与其连接的现场设备，所以不允许对这些点进行读访问。即，与可读或可写的 Q 访问不同的是，Q_:P 访问为只写访问。

Q_:P 访问也仅限于单个 CPU、SB 或 SM 所支持的输出大小（向上取整到最接近的字节）。例如，如果将 2 DI/2 DQ SB 组态为从 Q4.0 开始，则可按 Q4.0:P 和 Q4.1:P 或 QB4:P 的形式访问输出点。以 Q4.7:P 的形式访问 Q4.2:P 不会被拒绝，但没有任何意义，因为不会使用这些点。但不允许 QW4:P 和 QD4:P 的访问形式，因为它们超出了与该 SB 相关的字节偏移量。

使用 Q_:P 访问既影响物理输出，也影响存储在输出过程映像中的相应值。

表格 5-25　Q 存储器的绝对地址（立即）

位	Q[字节地址].[位地址]:P	Q1.1:P
字节、字或双字	Q[大小][起始字节地址]:P	QB5:P、QW10:P 或 QD40:P

M（位存储区）：针对控制继电器及数据的位存储区（M 存储器）用于存储操作的中间状态或其它控制信息。可以按位、字节、字或双字访问位存储区。M 存储器允许读访问和写访问。

表格 5-26　M 存储器的绝对地址

位	M[字节地址].[位地址]	M26.7
字节、字或双字	M[大小][起始字节地址]	MB20、MW30、MD50

临时（临时存储器）：CPU 根据需要分配临时存储器。启动代码块（对于 OB）或调用代码块（对于 FC 或 FB）时，CPU 将为代码块分配临时存储器并将存储单元初始化为 0。

临时存储器与 M 存储器类似，但有一个主要的区别：M 存储器在“全局”范围内有效，而临时存储器在“局部”范围内有效：

- M 存储器：任何 OB、FC 或 FB 都可以访问 M 存储器中的数据，也就是说这些数据可以全局性地用于用户程序中的所有元素。
- 临时存储器：CPU 限定只有创建或声明了临时存储单元的 OB、FC 或 FB 才可以访问临时存储器中的数据。临时存储单元是局部有效的，并且其它代码块不会共享临时存储器，即使在代码块调用其它代码块时也是如此。例如：当 OB 调用 FC 时，FC 无法访问对其进行调用的 OB 的临时存储器。

CPU 为每个 OB 优先级都提供了临时（本地）存储器：

- 16 KB 用于启动和程序循环（包括相关的 FB 和 FC）
- 6 KB 用于每次额外的中断事件线程，包括相关的 FB 和 FC

只能通过符号寻址的方式访问临时存储器。

可通过 STEP 7 中的调用结构查看程序中各块占用的临时（本地）存储器空间。从项目树中选择“程序信息”(Program info)，然后选择“调用结构”(Call structure) 选项卡。可以显示程序中的所有 OB，并且您可以进一步展开查看它们调用的块。对于每个块，都可以显示本地数据分配。用户也可以通过 STEP 7“工具 > 调用结构”(**Tools > Call structure**) 菜单命令来访问“调用结构”(Call structure) 显示。

DB（数据块）：DB 存储器用于存储各种类型的数据，其中包括操作的中间状态或 FB 的其它控制信息参数，以及许多指令（如定时器和计数器）所需的数据结构。可以按位、字节、字或双字访问数据块存储器。读/写数据块允许读访问和写访问。只读数据块只允许读访问。

表格 5-27　DB 存储器的绝对地址

位	DB[数据块编号].DBX[字节地址].[位地址]	DB1.DBX2.3
字节、字或双字	DB[数据块编号].DB [大小][起始字节地址]	DB1.DBB4、DB10.DBW2、DB20.DBD8

说明

在 LAD 或 FBD 中指定绝对地址时，STEP 7 会为此地址加上“%”字符前缀，以指示其为绝对地址。编程时，可以输入带或不带“%”字符的绝对地址（例如 %I0.0 或 I.0）。如果忽略，则 STEP 7 将加上“%”字符。

在 SCL 中，必须在地址前输入“%”来表示此地址为绝对地址。如果没有“%”，STEP 7 将在编译时生成未定义的变量错误

对 CPU 和 I/O 模块中的 I/O 进行组态

向设备组态添加 CPU 和 I/O 模块时，STEP 7 会自动分配 I 地址和 Q 地址。通过在设备组态中选择地址字段并输入新编号，可以更改默认寻址设置。

- 无论模块是否使用所有点，STEP 7 都按每组 8 点（1 字节）的方式分配数字量输入和输出。
- STEP 7 以 2 个为一组分配模拟量输入和输出，其中每个模拟点占用 2 个字节（16 位）。

图中显示的示例是配有两个 SM 及一个 SB 的 CPU 1214C。在该示例中，可以将 DI8 模块的地址更改为 2 而不是 8。用户可借助该工具更改大小有误或与其它地址冲突的地址范围。

5.3 模拟值的处理

模拟量信号模块可以提供输入信号，或等待表示电压范围或电流范围的输出值。这些范围是 ±10 V、±5 V、±2.5 V 或 0 - 20 mA。模块返回的值是整数值，其中，0 到 27648 表示电流的额定范围，-27648 到 27648 表示电压的额定范围。任何该范围之外的值即表示上溢或下溢。有关超出范围值的类型的详细信息，请参见模拟量输入表示法 (页 1556)和模拟量输出表示法 (页 1557)表格。

在控制程序中，很可能需要以工程单位使用这些值，例如表示体积、温度、重量或其它数量值。要以工程单位使用模拟量输入，必须首先将模拟值标准化为由 0.0 到 1.0 的实数（浮点）值。然后，必须将其标定为其表示的工程单位的最小值和最大值。对于要转换为模拟量输出值的以工程单位表示的值，应首先将以工程单位表示的值标准化为 0.0 和 1.0 之间的值，然后将其标定为 0 到 27648 之间或 -27648 到 27648 之间（取决于模拟模块的范围）的值。STEP 7 为此提供了 NORM_X 和 SCALE_X 指令 (页 300)。还可以使用 CALCULATE 指令 (页 240)来标定模拟值 (页 43)。

示例：模拟值处理

例如，假设模拟量输入的电流范围为 0 - 20 mA。模拟量输入模块返回的测量值介于 0 和 27648 之间。在此示例中，假设使用此模拟量输入值测量 50 °C 到 100 °C 的温度。几个采样值的含义如下：

模拟量输入值	工程单位
0	50 °C
6192	62.5 °C
12384	75 °C
18576	87.5 °C
27648	100 °C

在此示例中，通过模拟量输入值确定工程单位的计算方法如下：

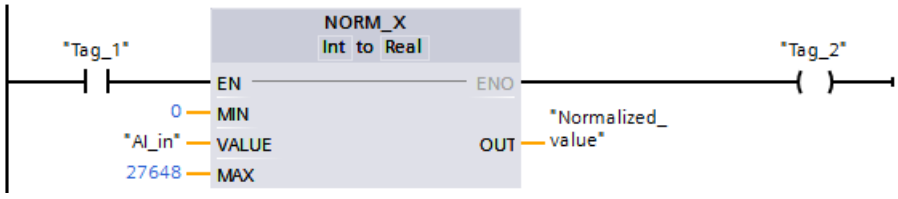
工程组态单位值 = 50 +（模拟量输入值）* (100 - 50) / (27648 - 0)

对于一般情况，公式为：

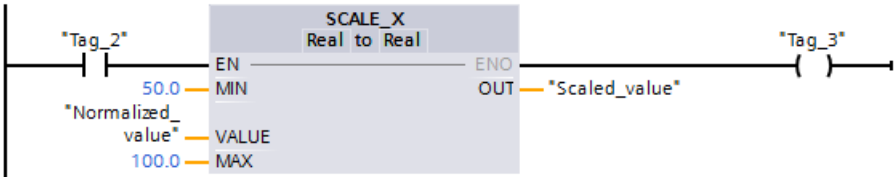
工程单位值 = (工程单位范围下限) +
(模拟量输入值) *
(工程单位范围上限 - 工程单位范围下限) /
(模拟量输入上限 - 模拟量输入下限)

在 PLC 应用中，典型的方法是将模拟量输入值标准化为 0.0 至 1.0 之间的浮点值。然后，需要将得到的值换算为工程单位范围内的浮点值。为简单起见，以下 LAD 指令使用常数值表示范围；实际上可能选择使用变量：

程序段 1



程序段 2



5.4 数据类型

数据类型用于指定数据元素的大小以及如何解释数据。每个指令参数至少支持一种数据类型，而有些参数支持多种数据类型。将光标停在指令的参数域上方，便可看到给定参数所支持的数据类型。

形参指的是指令上标记该指令要使用的数据位置的标识符（例如：ADD 指令的 IN1 输入）。实参指的是包含指令要使用的数据的存储单元（含“%”字符前缀）或常量（例如，%MD400 "Number_of_Widgets"）。用户指定的实参的数据类型必须与指令指定的形参所支持的数据类型之一匹配。

指定实参时，必须指定变量（符号）或者绝对（直接）存储器地址。变量将符号名（变量名）与数据类型、存储区、存储器偏移量和注释关联在一起，并且可以在 PLC 变量编辑器或块

（OB、FC、FB 和 DB）的接口编辑器中进行创建。如果输入一个没有关联变量的绝对地址，使用的地址大小必须与所支持的数据类型相匹配，而默认变量将在输入时创建。

除了 String、Struct、Array 和 DTL，其它所有数据类型都可以在 PLC 变量编辑器和块接口编辑器中使用。String、Struct、Array 和 DTL 只可在块接口编辑器中使用。还可以为许多输入参数输入常数值。

- 位和位序列 (页 121)：Bool（布尔或位值）、Byte（8 位字节值）、Word（16 位值）、DWord（32 位双字值）
- 整型 (页 122)
  - USInt（无符号 8 位整数）、SInt（有符号 8 位整数）、
  - UInt（无符号 16 位整数）、Int（有符号 16 位整数）
  - UDInt（无符号 32 位整数）、DInt（有符号 32 位整数）
- 浮点实数 (页 123)：Real（32 位实数或浮点值）、LReal（64 位实数或浮点值）
- 时间和日期 (页 124)：Time（32 位 IEC 时间值）、Date（16 位日期值）、TOD（32 位时间值）、DTL（12 字节日期和时间结构）
- 字符和字符串 (页 126)：Char（8 位单字符）、String（最长 254 个字符的可变长度字符串）
- 数组 (页 128)
- 数据结构 (页 129)：Struct
- PLC 数据类型 (页 130)
- Variant 数据类型 (页 130)

尽管以下 BCD 格式不能作为数据类型使用，转换指令支持以下 BCD 数字格式：

表格 5-28　BCD 格式的大小和范围

格式	大小（位）	数字范围	常量输入示例
BCD16	16	-999 到 999	123，-123
BCD32	32	-9999999 到 9999999	1234567，-1234567

5.4.1 Bool、Byte、Word 和 DWord 数据类型

表格 5-29 位和位序列数据类型

数据类型	位大小	数值类型	数值范围	常数示例	地址示例
Bool	1	布尔运算	FALSE 或 TRUE	TRUE	I1.0 Q0.1 M50.7 DB1.DBX2.3 Tag_name
		二进制	2#0 或 2#1	2#0	
		无符号整数	0 或 1	1	
		八进制	8#0 或 8#1	8#1	
		十六进制	16#0 或 16#1	16#1	
Byte	8	二进制	2#0 到 2#1111_1111	2#1000_1001	IB2 MB10 DB1.DBB4 Tag_name
		无符号整数	0 到 255	15	
		有符号整数	-128 到 127	-63	
		八进制	8#0 到 8#377	8#17	
		十六进制	B#16#0 到 B#16#FF，16#0 到 16#FF	B#16#F、16#F	
Word	16	二进制	2#0 到 2#1111_1111_1111_1111	2#1101_0010_1001_0110	MW10 DB1.DBW2 Tag_name
		无符号整数	0 到 65535	61680	
		有符号整数	-32768 到 32767	72	
		八进制	8#0 到 8#177_777	8#170_362	
		十六进制	W#16#0 到 W#16#FFFF、16#0 到 16#FFFF	W#16#F1C0、16#A67B	
DWord	32	二进制	2#0 到 2#1111_1111_1111_1111_1111_1111_1111_1111	2#1101_0100_1111_1110_1000_1100	MD10 DB1.DBD8 Tag_name
		无符号整数*	0 到 4_294_967_295	15_793_935	
		有符号整数*	-2_147_483_648 到 2_147_483_647	-400000	
		八进制	8#0 到 8#37_777_777_777	8#74_177_417	
		十六进制	DW#16#0000_0000 到 DW#16#FFFF_FFFF、16#0000_0000 到 16#FFFF_FFFF	DW#16#20_F30A、16#B_01F6	

* 下划线“_”是用于增加大于 8 位的数字可读性的千位分隔符。

5.4.2 整数数据类型

表格 5-30 整型数据类型（U = 无符号，S = 短，D = 双）

数据类型	位大小	数值范围	常数示例	地址示例
USInt	8	0 到 255	78, 2#01001110	MB0、DB1.DBB4、Tag_name
SInt	8	-128 到 127	+50, 16#50	
UInt	16	0 到 65,535	65295, 0	MW2、DB1.DBW2、Tag_name
Int	16	-32,768 到 32,767	30000, +30000	
UDInt	32	0 到 4,294,967,295	4042322160	MD6、DB1.DBD8、Tag_name
DInt	32	-2,147,483,648 到 2,147,483,647	-2131754992	

5.4.3 浮点型实数数据类型

如 ANSI/IEEE 754-1985 标准所述，实（或浮点）数以 32 位单精度数 (Real) 或 64 位双精度数 (LReal) 表示。 单精度浮点数的精度最高为 6 位有效数字，而双精度浮点数的精度最高为 15 位有效数字。 在输入浮点常数时，最多可以指定 6 位 (Real) 或 15 位 (LReal) 有效数字来保持精度。

表格 5-31 浮点型实数数据类型（L = 长浮点型）

数据类型	位大小	数值范围	常数示例	地址示例
Real	32	-3.402823e+38 到 -1.175 495e-38、 ±0、 +1.175 495e-38 到 +3.402823e+38	123.456, -3.4, 1.0e-5	MD100、DB1.DBD8、Tag_name
LReal	64	-1.7976931348623158e+308 到 -2.2250738585072014e-308、 ±0、 +2.2250738585072014e-308 到 +1.7976931348623158e+308	12345.123456789e40、1.2E+40	DB_name.var_name 规则： ● 不支持直接寻址 ● 可在 OB、FB 或 FC 块接口数组中进行分配

计算涉及到包含非常大和非常小数字的一长串数值时，计算结果可能不准确。 如果数字相差 10 的 x 次方，其中 x > 6 (Real) 或 15 (LReal)，则会发生上述情况。 例如 (Real)： 100 000 000 + 1 = 100 000 000.

5.4.4 时间和日期数据类型

表格 5-32 时间和日期数据类型

数据类型	大小	范围	常量输入示例
Time	32 位	T#-24d_20h_31m_23s_648ms 到 T#24d_20h_31m_23s_647ms 存储形式： -2,147,483,648 ms 到 +2,147,483,647 ms	T#5m_30s T#1d_2h_15m_30s_45ms TIME#10d20h30m20s630ms 500h10000ms 10d20h30m20s630ms
日期	16 位	D#1990-1-1 到 D#2168-12-31	D#2009-12-31 DATE#2009-12-31 2009-12-31
Time_of_Day	32 位	TOD#0:0:0.0 到 TOD#23:59:59.999	TOD#10:20:30.400 TIME_OF_DAY#10:20:30.400 23:10:1
DTL（长格式日期和时间）	12 个字节	最小： DTL#1970-01-01-00:00:00.0 最大： DTL#2262-04-11:23:47:16.854 775 807	DTL#2008-12-16-20:30:20.250

Time

TIME 数据作为有符号双整数存储，被解释为毫秒。 编辑器格式可以使用日期 (d)、小时 (h)、分钟 (m)、秒 (s) 和毫秒 (ms) 信息。

不需要指定全部时间单位。 例如，T#5h10s 和 500h 均有效。

所有指定单位值的组合值不能超过以毫秒表示的时间日期类型的上限或下限（-2,147,483,648 ms 到 +2,147,483,647 ms）。

日期

DATE 数据作为无符号整数值存储，被解释为添加到基础日期 1990 年 1 月 1 日的天数，用以获取指定日期。 编辑器格式必须指定年、月和日。

TOD

TOD (TIME_OF_DAY) 数据作为无符号双整数值存储，被解释为自指定日期的凌晨算起的毫秒数（凌晨 = 0 ms）。 必须指定小时（24 小时/天）、分钟和秒。 可以选择指定小数秒格式。

DTL

DTL（日期和时间长型）数据类型使用 12 个字节的结构保存日期和时间信息。 可以在块的临时存储器或者 DB 中定义 DTL 数据。 必须在 DB 编辑器的 “起始值”(Start value) 列为所有组件输入一个值。

表格 5-33　DTL 的大小和范围

长度（字节）	格式	值范围	值输入的示例
12	时钟和日历 年-月-日:时:分:秒.纳秒	最小： DTL#1970-01-01-00:00:00.0 最大： DTL#2554-12-31-23:59:59.999 999 999	DTL#2008-12-16-20:30:20.250

DTL 的每一部分均包含不同的数据类型和值范围。 指定值的数据类型必须与相应部分的数据类型相一致。

表格 5-34　DTL 结构的元素

Byte	组件	数据类型	值范围
0	年	UINT	1970 到 2554
1			
2	月	USINT	1 到 12
3	日	USINT	1 到 31
4	工作日 [1]	USINT	1（星期日）到 7（星期六）[1]
5	小时	USINT	0 到 23
6	分	USINT	0 到 59
7	秒	USINT	0 到 59

Byte	组件	数据类型	值范围
8	纳秒	UDINT	0 到 999 999 999
9			
10			
11			

[1] 年-月-日:时:分:
秒.纳秒格式中不包括星期。

5.4.5 字符和字符串数据类型

表格 5-35 字符和字符串数据类型

数据类型	大小	范围	常量输入示例
Char	8 位	16#00 到 16#FF	'A', 't', '@', 'ä', '∑'
WChar	16 位	16#0000 到 16#FFFF	'A', 't', '@', 'ä', '∑', 亚洲字符、西里尔字符以及其它字符
String	n+ 2 字节	n =（0 到 254 字节）	"ABC"
WString	n + 2 个字	n =（0 到 65534 个字）	"ä123@XYZ.COM"

Char 和 WChar

Char 在存储器中占一个字节，可以存储以 ASCII 格式（包括扩展 ASCII 字符代码）编码的单个字符。WChar 在存储器中占一个字的空间，可包含任意双字节字符表示形式。

编辑器语法在字符的前面和后面各使用一个单引号字符。可以使用可见字符和控制字符。

String 和 WString

CPU 支持使用 String 数据类型存储一串单字节字符。String 数据类型包含总字符数（字符串中的字符数）和当前字符数。String 类型提供了多达 256 个字节，用于在字符串中存储最大总字符数（1 个字节）、当前字符数（1 个字节）以及最多 254 个字节。String 数据类型中的每个字节都可以是从 16#00 到 16#FF 的任意值。

WString 数据类型支持单字（双字节）值的较长字符串。第一个字包含最大总字符数；下一个字包含总字符数，接下来的字符串可包含多达 65534 个字。WString 数据类型中的每个字可以是 16#0000 - 16#FFFF 之间的任意值。

可以对 IN 类型的指令参数使用带单引号的文字串（常量）。例如，‘ABC’是由三个字符组成的字符串，可用作 S_CONV 指令中 IN 参数的输入。还可通过在 OB、FC、FB 和 DB 的块接口编辑器中选择“String”或“WString”数据类型来创建字符串变量。无法在 PLC 变量编辑器中创建字符串。

可从数据类型下拉列表中选择一种数据类型，输入关键字“String”或“WString”，然后在方括号中以字节 (String) 或字 (WString) 为单位指定最大字符串大小。例如，“MyString String[10]”指定 MyString 的最大长度为 10 个字节。如果不包含带有最大长度的方括号,则假定字符串的最大长度为 254 并假定 WString 的最大长度为 65534。“MyWString WString[1000]”可指定一个 1000 字的 WString。

以下示例定义了一个最大字符计数为 10，当前字符计数为 3 的字符串。这意味着该字符串当前包含 3 个单字节字符，但可以对其进行扩展使其包含多达 10 个单字节字符。

表格 5-36　String 数据类型示例

总字符数	当前字符数	字符 1	字符 2	字符 3	...	字符 10
10	3	'C' (16#43)	'A' (16#41)	'T' (16#54)	...	-
字节 0	字节 1	字节 2	字节 3	字节 4	...	字节 11

以下示例定义了一个最大字符计数为 500，当前字符计数为 300 的 WString。这意味着该字符串当前包含 300 个单字字符，但可以对其进行扩展使其包含多达 500 个单字字符。

表格 5-37　WString 数据类型示例

总字符数	当前字符数	字符 1	字符 2 到 299	字符 300	...	字符 500
500	300	'ä' (16#0084)	ASCII 字符字	'M' (16#004D)	...	-
字 0	字 1	字 2	字 3 到 300	字 301	...	字 501

ASCII 控制字符可用于 Char、Wchar、String 和 WString 数据中。下表给出了控制字符语法的示例。

表格 5-38　有效的 ASCII 控制字符

控制字符	ASCII 十六进制值 (Char)	ASCII 十六进制值 (WChar)	控制功能	示例
$L 或 $l	16#0A	16#000A	换行	'$LText'、'$0AText'
$N 或 $n	16#0A 和 16#0D	16#000A 和 16#000D	线路中断 新行显示字符串中的两个字符。	'$NText'、'$0A $0DText'
$P 或 $p	16#0C	16#000C	换页	'$PText'、'$0CText'
$R 或 $r	16#0D	16#000D	回车 (CR)	'$RText'、'$0DText'
$T 或 $t	16#09	16#0009	制表符	'$TText'、'$09Text'
$$	16#24	16#0024	美元符号	'100$$', '100$24'
$'	16#27	16#0027	单引号	'$'Text$''、'$27Text$27'

5.4.6 数组数据类型

数组

可以创建包含多个相同数据类型元素的数组。 数组可以在 OB、FC、FB 和 DB 的块接口编辑器中创建。 无法在 PLC 变量编辑器中创建数组。

要在块接口编辑器中创建数组，请为数组命名并选择数据类型“Array [lo .. hi] of type”，然后根据如下说明编辑“lo”、“hi”和“type”：

- lo - 数组的起始（最低）下标
- hi - 数组的结束（最高）下标
- type - 数据类型之一，例如 BOOL、SINT、UDINT

表格 5-39 ARRAY 数据类型规则

数据类型	数组语法		
ARRAY	Name [index1_min..index1_max, index2_min..index2_max] of <数据类型> ● 全部数组元素必须是同一数据类型。 ● 索引可以为负，但下限必须小于或等于上限。 ● 数组可以是一维到六维数组。 ● 用逗点字符分隔多维索引的最小最大值声明。 ● 不允许使用嵌套数组或数组的数组。 ● 数组的存储器大小 = （一个元素的大小 * 数组中的元素的总数）		
	数组索引	有效索引数据类型	数组索引规则
	常量或变量	USInt, SInt, UInt, Int, UDInt, DInt	● 限值：-32768 到 +32767 ● 有效： 常量和变量混合 ● 有效： 常量表达式 ● 无效： 变量表达式

示例： 数组声明	ARRAY[1..20] of REAL	一维，20 个元素
	ARRAY[-5..5] of INT	一维，11 个元素
	ARRAY[1..2, 3..4] of CHAR	二维，4 个元素
示例： 数组地址	ARRAY1[0]	ARRAY1 元素 0
	ARRAY2[1,2]	ARRAY2 元素 [1,2]
	ARRAY3[i,j]	如果 i =3 且 j=4，则对 ARRAY3 的元素 [3, 4] 进行寻址

5.4.7 数据结构数据类型

可以用数据类型“Struct”来定义包含其它数据类型的数据结构。 Struct 数据类型可用来以单个数据单元方式处理一组相关过程数据。 在数据块编辑器或块接口编辑器中命名 Struct 数据类型并声明内部数据结构。

数组和结构还可以集中到更大结构中。 一套结构可嵌套八层。 例如，可以创建包含数组的多个结构组成的结构。

5.4.8 PLC 数据类型

PLC 数据类型可用来定义可以在程序中多次使用的数据结构。可以通过打开项目树的“PLC 数据类型”分支并双击“添加新数据类型”项来创建 PLC 数据类型。在新创建的 PLC 数据类型项上，两次单击可重命名默认名称，双击则会打开 PLC 数据类型编辑器。

可使用在数据块编辑器中的相同编辑方法创建自定义 PLC 数据类型结构。为任何必要的数据类型添加新的行，以创建所需数据结构。

如果创建新的 PLC 数据类型，则该新 PLC 类型名称将出现在 DB 编辑器和代码块接口编辑器的数据类型选择器下拉列表中。

您可以按照以下方式使用 PLC 数据类型：

- 作为代码块接口或数据块中的数据类型
- 作为创建使用同一数据结构的多个全局数据块的模板
- 作为 CPU I 和 Q 存储区中 PLC 变量声明的数据类型

例如，PLC 数据类型可能是混合颜色的配方。用户可以将该 PLC 数据类型分配给多个数据块。您可以在每个数据块中调整变量以创建特定颜色。

5.4.9 Variant 指针数据类型

Variant 数据类型可以指向不同数据类型的变量或参数。 Variant 指针可以指向结构和单独的结构元素。 Variant 指针不会占用存储器的任何空间。

表格 5-40 Variant 指针的属性

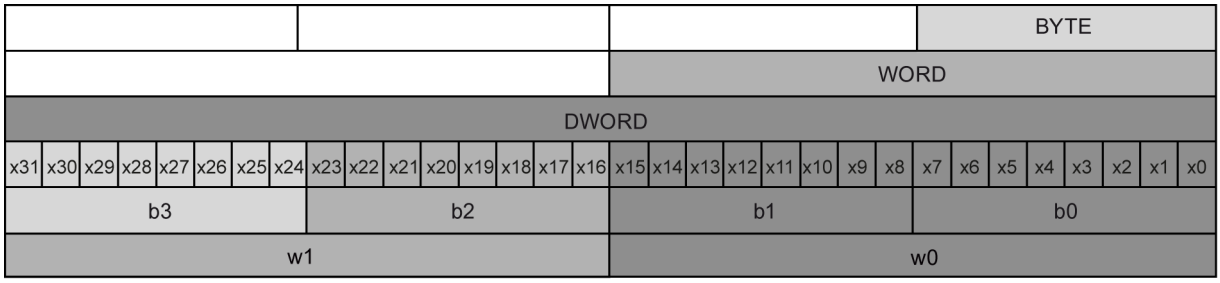
长度（字节）	表示方式	格式	示例输入
0	符号	操作数	MyTag
		DB_name.Struct_name.element_name	MyDB.Struct1.pressure1
	绝对	操作数	%MW10
		DB_number.Operand Type Length	P#DB10.DBX10.0 INT 12

5.4.10 访问一个变量数据类型的“片段”

可以根据大小按位、字节、或字级别访问 PLC 变量和数据块变量。 访问此类数据片段的语法如下所示：

- "<PLC 变量名称>".xn（按位访问）
- "<PLC 变量名称>".bn（按字节访问）
- "<PLC 变量名称>".wn（按字访问）
- "<数据块名称>".<变量名称>.xn（按访问）
- "<数据块名称>".<变量名称>.bn（按字节访问）
- "<数据块名称>".<变量名称>.wn（按字访问）

双字大小的变量可按位 0 - 31、字节 0 - 3 或字 0 - 1 访问。一个字大小的的变量可按位 0 - 15、字节 0 - 1 或字 0 访问。字节大小的变量则可按位 0 - 7 或字节 0 访问。当预期操作数为位、字节或字时，则可使用位、字节和字片段访问方式。



说明

可以按片段访问的有效数据类型有： Byte、Char、Conn_Any、Date、DInt、DWord、Event_Any、Event_Att、Hw_Any、Hw_Device、HW_Interface、Hw_Io、Hw_Pwm、Hw_SubModule、Int、OB_Any、OB_Att、OB_Cyclic、OB_Delay、OB_WHINT、OB_PCYCLE、OB_STARTUP、OB_TIMEERROR、OB_Tod、Port、Rtm、SInt、Time、Time_Of_Day、UDInt、UInt、USInt 和 Word。 Real 类型的 PLC 变量可以按片段访问，但 Real 类型的数据块变量则不行。

示例

在 PLC 变量表中，“DW”是一个声明为 DWORD 类型的变量。 在以下示例中，显示了按位、字节和字片段的访问方式：

	LAD	FBD	SCL
按位访问	"DW".x11	"DW".x11 &	IF "DW".x11 THEN ... END_IF;
按字节访问	"DW".b2 == Byte "DW".b3	"DW".b2 IN1 == Byte "DW".b3 IN2	IF "DW".b2 = "DW".b3 THEN ... END_IF;
按字访问	"DW".w0 IN1 EN ENO OUT AND Word "DW".w1 IN2	... EN "DW".w0 IN1 OUT AND Word "DW".w1 IN2 ENO	out:= "DW".w0 AND "DW".w1;

5.4.11 访问带有一个 AT 覆盖的变量

借助 AT 变量覆盖，可通过一个不同数据类型的覆盖声明访问已声明的块变量。例如，可以通过 Array of Bool 寻址数据类型为 Byte、Word 或 DWord 变量的各个位。AT 覆盖支持以下变量类型：

- 标准访问块中的变量
- 优化块中的保留变量

声明

要覆盖一个参数，可以在待覆盖的参数后直接声明一个附加参数，然后选择数据类型“AT”。编辑器随即创建该覆盖，然后选择将用于该覆盖的数据类型、结构或数组。

示例

在本例中，显示一个标准访问 FB 的输入参数。字节变量 B1 将由布尔数组覆盖：

B1		Byte	0.0
OV	AT"B1"	Array[0..7] of Bool	0.0
OV[0]		Bool	0.0
OV[1]		Bool	0.1
OV[2]		Bool	0.2
OV[3]		Bool	0.3
OV[4]		Bool	0.4
OV[5]		Bool	0.5
OV[6]		Bool	0.6
OV[7]		Bool	0.7

另一个示例是 DWord 变量由一个 Struct 覆盖。Struct 包括字、字节和两个布尔值：

DW1		DWord	2.0
DW1_Struct	AT"DW1"	Struct	2.0
W1		Word	0.0
B1		Byte	2.0
BO1		Bool	3.0
BO2		Bool	3.1

块接口的 “偏移量”(Offset) 列中显示与原始变量相关的被覆盖数据类型的位置。

可直接在程序逻辑中指定覆盖类型的地址：

LAD	FBD	SCL
#OV[1]	& #OV[1]	IF #OV[1] THEN ... END_IF;
#DW1_Struct.W1 == Word W#16#000C	== Word #DW1_Struct.W1 IN1 W#16#000C IN2	IF #DW1_Struct.W1 = W#16#000C THEN ... END_IF;
MOVE EN ENO #DW1_Struct.B1 IN OUT1	MOVE ... EN OUT1 <???> #DW1_Struct.B1 IN ENO	out1 := #DW1_Struct.B1;
#OV[4] #DW1_Struct.BO2	& #OV[4] #DW1_Struct.BO2	IF #OV[4] AND #DW1_Struct.BO2 THEN ... END_IF;

准则

- 在可进行标准（未优化）访问的 FB 和 FC 块中，可覆盖变量。
- 在优化的 FB 和 FC 块中，可覆盖任何保留变量。

- 可以覆盖所有类型和所有声明部分的变量。
- 可以同使用其它块参数一样使用覆盖后的参数。
- 不能覆盖 VARIANT 类型的参数。
- 覆盖参数的大小必须小于等于被覆盖的参数。
- 必须在覆盖变量并选择关键字“AT”作为初始数据类型后立即声明覆盖变量。

5.5 使用存储卡

说明

CPU 仅支持预格式化的 SIMATIC 存储卡 (页 1639)。
在将程序复制到格式化的存储卡之前，请删除存储卡中以前保存的所有程序。

可将存储器用作传送卡或程序卡。传送卡和程序卡包括所有代码块和数据块、所有工艺对象和设备组态。传送卡和程序卡不包含如强制表、监视表或 PLC 变量表等表格。

- 使用传送卡 (页 137)将程序复制到 CPU 的内部装载存储器中，而使用 STEP 7。
  在密码丢失或忘记密码时 (页 145)，可使用空传送卡访问受密码保护的 CPU。
- 将程序卡 (页 139)用作 CPU 的外部装载存储器。

下载固件更新 (页 143)时，也会使用存储卡。

5.5.1 在 CPU 中插入存储卡

注意
对存储卡和卡槽进行静电放电保护 静电放电可能会损坏存储卡或 CPU 上的卡槽。 在操控存储卡时，请先接触接地传导垫和/或佩戴接地腕带。将存储卡存放在导电容器内。

检查以确定存储卡没有写保护。滑动保护开关，使其离开“Lock”位置。

注意如果将写保护存储卡插入 CPU 中，STEP 7 会在下一次上电时显示诊断消息提醒您这一情况。CPU 将无故障上电，但如果存储卡为写保护，例如，包含配方或数据日志的指令将返回错误。

⚠ 警告

插入存储卡之前，请检查并确认 CPU 当前未执行任何操作。

如果将存储卡（无论组态为程序卡、传送卡还是固件更新卡）插入到正在运行的 CPU，CPU 将立即进入 STOP 模式，这可能引起过程中断，进而导致人员死亡或严重受伤。

在插入或拔出存储卡前，务必确保 CPU 当前未控制任何机器或过程。因此务必要为您的应用或过程安装急停电路。

说明

请勿将 V3.0 程序传送卡插入 S7-1200 V4.x CPU。

版本 3.0 程序传送卡与版本 S7-1200 V4.x CPU 不兼容。插入含有 V3.0 程序的存储卡会导致 CPU 错误。

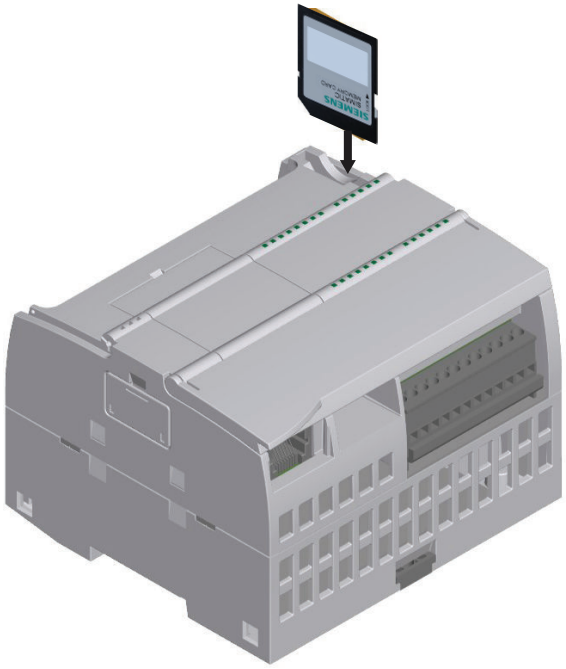
如果插入无效版本的程序传送卡 (页 137)，则请取出该卡，然后执行 STOP 到 RUN 切换、存储器复位 (MRES) 或循环上电。将 CPU 从错误状态恢复后，即可下载有效的 V4.x CPU 程序。

要将 V3.0 程序传送到 V4.x 程序，必须使用 TIA Portal 在硬件配置中更改设备。

说明

如果在 CPU 处于 STOP 模式时插入存储卡，则诊断缓冲区将显示一条消息提示存储卡评估已经启动。下次 CPU 切换到 RUN 模式、使用 MRES 复位 CPU 存储器或者 CPU 循环上电时，CPU 会评估存储卡。

表格 5-41　插入存储卡

	要插入存储卡，需打开 CPU 顶盖，然后将存储卡插入到插槽中。推弹式连接器可以轻松地插入和取出。 存储卡要求正确安装。

插入存储卡时的 CPU 行为

当在 CPU 中插入存储卡时，CPU 将执行以下步骤：

1. 切换到 STOP 模式（如果尚未在 STOP 模式）
2. 提示以下选项之一：
   – 循环上电
   – 切换到 RUN 模式。
   – 执行存储器复位
3. 评估存储卡

CPU 如何评估存储卡

如果不在设备组态的保护属性中 (页 201)组态 CPU“禁用从内部装载存储器到外部装载存储器的复制操作”，CPU 将确定您插入的存储卡为何中类型：

- **空存储卡**：空白存储卡不具备作业文件（S7_JOB.S7S）。如果插入空白存储卡，CPU 将添加一个程序作业文件。如果随后将内部装载存储器复制到外部装载存储器（存储卡中的程序文件）中并将内部装载存储器擦除。
- **空白程序卡**：空白程序卡具备一个空的程序作业文件。此时，CPU 将内部装载存储器复制到外部装载存储器（存储卡中的程序文件）中并将内部装载存储器擦除。

如果在设备组态的保护属性中组态 CPU“禁用从内部装载存储器到外部装载存储器的复制操作”，CPU 将执行以下操作：

- **空存储卡**：空白存储卡不具备作业文件（S7_JOB.S7S）。如果插入空白存储卡，CPU 将不执行任何操作。CPU 不会创建程序作业文件并不将内部装载存储器复制到外部装载存储器（存储卡中的程序文件）中。不擦除内部装载存储器。
- **空白程序卡**：空白程序卡具备一个空的程序作业文件。对于此情况，CPU 不执行操作。CPU 不会将内部装载存储器复制到外部装载存储器（存储卡中的程序文件）中。不擦除内部装载存储器。

如果将程序卡 (页 139)、传送卡 (页 137)或包含固件更新的存储卡 (页 143)插入 CPU 中，“禁用从内部装载存储器到外部装载存储器的复制操作”的组态设置对 CPU 如何评估存储卡没有影响。

5.5.2 将项目复制到存储卡之前组态 CPU 的启动参数

将程序复制到传送卡或程序卡时，程序中包含了 CPU 的启动参数。 将程序复制到传送卡之前，请始终确保组态了 CPU 在循环上电后的工作模式。 选择 CPU 是在 STOP 模式、RUN 模式还是上一个模式（通电周期之前）下启动。

5.5.3 将存储卡用作“传送”卡

注意
对存储卡和卡槽进行静电放电保护 静电放电可能会损坏存储卡或 CPU 上的卡槽。 通过以下一种或两种方法安全处理存储卡： ● 请与已接地的导电垫接触。 ● 在操控存储卡时，请先佩戴接地腕带。 将存储卡存放在导电容器内。

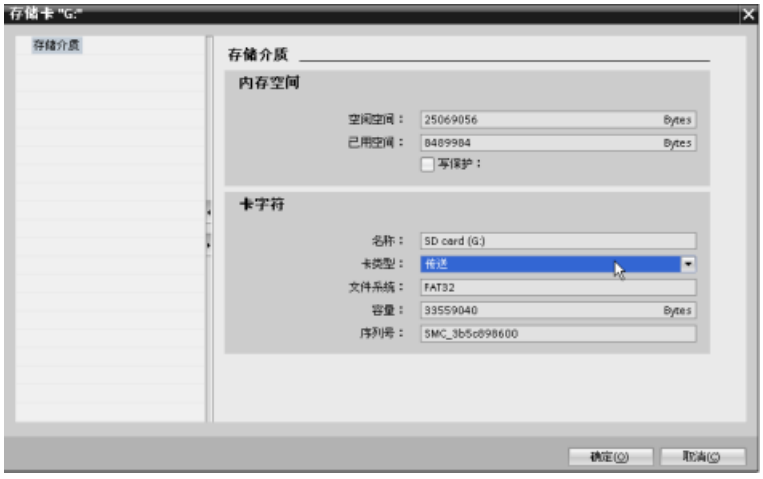
创建传送卡

请牢记在将程序复制到传送卡之前组态 CPU 的启动参数 (页 137)。要创建传送卡，请按以下步骤操作：

1. 将不受写保护的空白 SIMATIC 存储卡插入与计算机相连的 SD 卡读卡器/写卡器中。（如果卡处于写保护状态，则应滑动保护开关，使其离开“Lock”位置。）
   如果要重复使用包含用户程序、数据日志、配方或固件更新程序的 SIMATIC 存储卡，那么在重新使用该存储卡之前**必须**删除这些文件。使用 Windows 资源管理器显示存储卡的内容，删除“S7_JOB.S7S”文件以及任何现有文件夹（如“SIMATIC.S7S”、“FWUPDATE.S7S”、“DataLogs”和“Recipes”）。

注意
请勿删除存储卡上的“__LOG__”和“crdinfo.bin”隐藏文件。 存储卡必须包含“__LOG__”和“crdinfo.bin”文件。如果删除了这些文件，将无法在 CPU 中使用该存储卡。

2. 在项目树中（项目视图），展开“SIMATIC 卡读卡器”(SIMATIC Card Reader) 文件夹，然后选择读卡器。
3. 右键单击读卡器中存储卡对应的驱动器盘符，然后从右键快捷菜单中选择“属性”(Properties)，显示“存储卡”(Memory card) 对话框。
4. 在“存储卡”(Memory card) 对话框中，从“卡类型”(Card type) 下拉菜单中选择“传送”(Transfer)。
   此时，STEP 7 将创建空传送卡。如果要创建空传送卡以便在丢失 CPU 密码 (页 145)后恢复，请从读卡器中移除传送卡。



5. 通过在项目树中选择 CPU 设备（例如 PLC_1 [CPU 1214C DC/DC/DC]），将该 CPU 设备拖动到存储卡来添加程序。（另一种方法是复制 CPU 设备，并将其粘贴到存储卡中。）将 CPU 设备复制到存储卡时，“装载预览”(Load preview) 对话框会打开。
6. 在“装载预览”(Load preview) 对话框中，单击“装载”(Load) 按钮，以将 CPU 设备复制到存储卡。
7. 在对话框显示一条消息指示 CPU 设备（程序）已正确装载时，单击“完成”(Finish) 按钮。

使用传送卡

⚠ 警告

插入存储卡之前，请检查并确认 CPU 当前并未执行任何操作。

插入存储卡会使 CPU 切换到 STOP 模式，这可能会影响在线操作或机器的运行。意外的过程操作或机器操作可能会导致死亡、人身伤害和/或财产损失。

插入传送卡前，请务必确保 CPU 处于 STOP 模式且程序处于安全状态。

说明

不要将 V3.0 程序传送卡插入更高型号的 CPU 中。

版本 3.0 程序传送卡和更高型号 S7-1200 CPU 不兼容。插入含有 V3.0 程序的存储卡会导致 CPU 错误。

如果插入无效版本的程序传送卡，则请取出该卡，然后执行 STOP 到 RUN 切换、存储器复位 (MRES) 或循环上电。将 CPU 从错误条件中恢复后，可以下载有效的 CPU 程序。

要将程序传送到 CPU，请按以下步骤操作：

1. 将传送卡插入 CPU 中 (页 134)。如果 CPU 处于 RUN 模式，它将转至 STOP 模式。维护 (MAINT) LED 闪烁，表示需要对存储卡进行评估。此时，现有程序仍在 CPU 中。
2. 对 CPU 循环上电以评估存储卡。另一种重启 CPU 的办法是通过 STEP 7 执行 STOP-RUN 切换或存储器复位 (MRES)。
3. 重启后，CPU 会对存储卡进行评估并将程序复制到 CPU 的内部装载存储器。
   RUN/STOP LED 呈绿色和黄色交替闪烁，表示正在复制程序。当 RUN/STOP LED 呈黄色亮起（保持稳定）且 MAINT LED 闪烁黄色时，表示复制过程已完成。然后可以取出存储卡。
4. 重启 CPU（通过恢复供电或另一种重启方法），以评估传送到内部装载存储器的新程序。

CPU 随后进入您为项目组态的启动模式（RUN 或 STOP）。

说明

将 CPU 设置为 RUN 模式之前，必须先取出传送卡。

5.5.4 将存储卡用作“程序”卡

注意

静电放电可能会损坏存储卡或 CPU 上的卡槽。

在操控存储卡时，请先接触接地传导垫和/或佩戴接地腕带。将存储卡存放在导电容器内。

检查以确定存储卡没有写保护。滑动保护开关，使其离开“Lock”位置。

在将程序元素复制到程序卡之前，请删除存储卡中以前保存的所有程序。

创建程序卡

存储卡被用作程序卡时，它就是 CPU 的外部装载存储器。如果取出程序卡，CPU 的内部装载存储器会是空的。

说明

如果在 CPU 中插入空存储卡，然后通过对 CPU 循环上电、执行 STOP 到 RUN 切换或者执行存储器复位 (MRES) 来进行储存卡评估，则 CPU 内部装载存储器中的程序和强制值将复制到存储卡中。（此时存储卡就是程序卡。）复制完成后，将擦除 CPU 内部装载存储器中的程序。CPU 随后进入组态的启动模式（RUN 或 STOP）。

请务必牢记在将项目复制到程序卡之前组态 CPU 的启动参数 (页 137)。要创建程序卡，请按以下步骤操作：

1. 将不受写保护的空白 SIMATIC 存储卡插入与计算机相连的 SD 卡读卡器/写卡器中。（如果卡处于写保护状态，则应滑动保护开关，使其离开“Lock”位置。）
   如果要重复使用包含用户程序、数据日志、配方或固件更新程序的 SIMATIC 存储卡，那么在重新使用该存储卡之前必须删除这些文件。可以使用 Windows 资源管理器，显示存储卡中的内容并删除以下文件和文件夹（如其存在）：
   - S7_JOB.S7S
   - SIMATIC.S7S
   - FWUPDATE.S7S
   - DataLogs
   - Recipes

注意
请勿删除存储卡上的“__LOG__”和“crdinfo.bin”隐藏文件。 存储卡必须包含“__LOG__”和“crdinfo.bin”文件。如果删除了这些文件，将无法在 CPU 中使用该存储卡。

2. 在项目树中（项目视图），展开“读卡器/USB 存储器”(Card Reader/USB memory) 文件夹，然后选择读卡器。
3. 右键单击读卡器中存储卡对应的驱动器盘符，然后从右键快捷菜单中选择“属性”(Properties)，显示“存储卡”(Memory card) 对话框。

4. 在“存储卡”(Memory card) 对话框中，从快捷菜单中选择“程序”(Program)。

5. 通过在项目树中选择 CPU 设备（例如 PLC_1 [CPU 1214C DC/DC/DC]），将该 CPU 设备拖动到存储卡来添加程序。（另一种方法是复制 CPU 设备，并将其粘贴到存储卡中。）将 CPU 设备复制到存储卡时，“装载预览”(Load preview) 对话框会打开。
6. 在“装载预览”(Load preview) 对话框中，单击“装载”(Load) 按钮，以将 CPU 设备复制到存储卡。
7. 在对话框显示一条消息指示 CPU 设备（程序）已正确装载时，单击“完成”(Finish) 按钮。

将程序卡用作 CPU 的装载存储器

> ⚠ **警告**
>
> **与插入程序卡相关的风险**
>
> 插入存储卡之前，请检查并确认 CPU 当前并未执行任何操作。
>
> 插入存储卡会使 CPU 切换到 STOP 模式，这可能会影响在线操作或机器的运行。意外的过程操作或机器操作可能会导致死亡、人身伤害和/或财产损失。
>
> 在插入存储卡前，请务必确保 CPU 处于离线模式且处于安全状态。

要对 CPU 使用程序卡，请按以下步骤操作：

1. 将程序卡插入 CPU。如果 CPU 处于 RUN 模式，则它将切换到 STOP 模式。维护 (MAINT) LED 闪烁，表示需要对存储卡进行评估。
2. 对 CPU 循环上电以评估存储卡。另一种重启 CPU 的办法是通过 STEP 7 执行 STOP-RUN 切换或存储器复位 (MRES)。
3. CPU 重启并对程序卡进行评估后，将擦除其内部装载存储器。

CPU 随后进入您为 CPU 组态的启动模式（RUN 或 STOP）。

程序卡必须保留在 CPU 中。取出程序卡将导致 CPU 的内部装载存储器中不会留下任何程序。

⚠ 警告

与取出程序卡相关的风险

如果取出程序卡，CPU 将失去外部装载存储器，并生成一条错误消息。CPU 切换到 STOP 模式并且错误 LED 闪烁。

控制设备在不安全情况下运行时可能会出现故障，从而导致受控设备的意外运行。这种意外运行可能会导致人员死亡、重伤和/或设备损坏。

取出程序卡时，必须清楚您正在将程序从 CPU 中移除。

SIMATIC 存储卡的使用寿命

SIMATIC 存储卡的使用寿命取决于以下等因素：

- 每个存储器块的删除和写入操作次数
- 写入的字节数
- 环境温度等外部影响

说明

写入和删除操作对 SIMATIC 存储卡使用寿命的影响

写入或删除操作（尤其是重复的（循环）写入/删除操作）将缩短 SIMATIC 存储卡的使用寿命。

循环执行以下指令将缩短存储卡的使用寿命，具体情况取决于写入次数与数据量：

- CREATE_DB（通过属性“在装载存储器中创建 DB”）
- DataLogWrite
- RecipeExport
- RecipeImport（如果目标 DB 位于装载存储器中）
- WRIT_DBL
- SET_TIMEZONE

除了循环写入/删除操作之外，写入或删除大量数据也会对 SIMATIC 存储卡的使用寿命造成负面影响。

5.5.5 固件更新

您可以使用 SIMATIC 存储卡执行固件更新。

注意
对存储卡和卡槽进行静电放电保护 静电放电可能会损坏存储卡或 CPU 上的卡槽。 在操控存储卡时，请先接触接地传导垫和/或佩戴接地腕带。将存储卡存放在导电容器内。

从 Siemens 工业在线支持 (https://support.industry.siemens.com/cs/cn/zh)下载固件更新时，会使用 SIMATIC 存储卡。在该网站中，导航到“下载”(Downloads)。从此处搜索需要更新的特定类型模块。

此外，还可以直接访问 S7-1200 下载网页 (https://support.industry.siemens.com/cs/cn/zh/ps/13683/dl)。

说明

固件更新无法将 S7-1200 CPU V3.0 或之前的版本更新到 S7-1200 V4.0 或 V4.1。

还可以通过以下任一方法来执行固件更新：

- 使用 STEP 7 的在线和诊断工具 (页 1390)
- 使用 Web 服务器“模块信息”标准 Web 页面 (页 1045)
- 使用 SIMATIC 自动化工具 (https://support.industry.siemens.com/cs/cn/zh/view/98161300/en)

注意
请勿使用 Windows 的格式化程序或其它格式化程序来重新格式化存储卡。 如果使用 Microsoft Windows 的格式化程序重新格式化了 Siemens 存储卡，那么 S7-1200 CPU 将无法再使用该存储卡。

要将固件更新下载到存储卡中，请执行以下步骤：

1. 将不受写保护的空白 SIMATIC 存储卡插入与计算机相连的 SD 卡读卡器/写卡器中。（如果卡处于写保护状态，则应滑动保护开关，使其离开“Lock”位置。）
   您可重复使用包含用户程序或其它固件更新程序的 SIMATIC 存储卡，但您必须删除该存储卡上的一些文件。
   要重复使用存储卡，**必须**在下载固件更新前删除“S7_JOB.S7S”文件以及任何现有“数据日志”文件夹或任何文件夹（如“SIMATIC.S7S”或“FWUPDATE.S7S”）。可以使用 Windows 资源管理器，显示存储卡中的内容并删除相关文件和文件夹。

注意
请勿删除存储卡上的“__LOG__”和“crdinfo.bin”隐藏文件。 存储卡必须包含“__LOG__”和“crdinfo.bin”文件。如果删除了这些文件，将无法在 CPU 中使用该存储卡。

2. 选择该模块所对应的固件更新 zip 文件，然后将其下载到您的计算机中。双击该文件，将该文件的目标路径设置为 SIMATIC 存储卡的根目录，然后开始解压缩。解压缩完成之后，存储卡的根目录中将包含一个“FWUPDATE.S7S”目录和一个“S7_JOB.S7S”文件。
3. 从读卡器/写卡器中安全弹出卡。

要安装固件更新，请执行以下步骤：

⚠ 警告
在安装固件更新之前，请确定 CPU 当前未执行任何进程。 安装固件更新程序时 CPU 将切换到 STOP 模式，这可能会影响在线操作或机器的运行。意外的过程操作或机器操作可能会导致死亡、人身伤害和/或财产损失。 在插入存储卡前，请务必确保 CPU 处于离线模式且处于安全状态。

1. 将存储卡插入 CPU 中。如果 CPU 处于 RUN 模式，则 CPU 将切换到 STOP 模式。维护 (MAINT) LED 闪烁，表示需要对存储卡进行评估。
2. 对 CPU 进行通电以启动固件更新程序。另一种重启 CPU 的办法是通过 STEP 7 执行 STOP-RUN 切换或存储器复位 (MRES)。

   说明

   要完成模块的固件更新，必须确保模块的 24 V DC 电源保持接通。

   CPU 重启之后，将开始执行固件更新。RUN/STOP LED 呈绿色和黄色交替闪烁，表示正在复制更新程序。等到 RUN/STOP LED 为黄色常亮且 MAINT LED 闪烁时，表示复制过程已完成。然后必须取出存储卡。
3. 取出存储卡后，再次重新启动 CPU（通过重新通电或其它重新启动方法）以装载新固件程序。

用户程序和硬件配置将不受固件更新的影响。CPU 通电后，CPU 将进入组态后的启动状态。（如果 CPU 的启动模式已组态为“暖启动 - 断电前的模式”，CPU 将处于 STOP 模式，因为 CPU 的前一个状态为 STOP。）

说明

更新多个连接到 CPU 的模块

如果硬件配置包含多个与存储卡上单个固件更新文件相对应的模块，则 CPU 将按组态顺序（即按模块在 STEP 7 设备组态中的位置的升序）对所有适用模块（CM、SM 和 SB）应用更新。

如果已将多个模块的多个固件更新下载到存储卡，则 CPU 将按这些更新下载到存储卡的顺序应用更新。

5.6 丢失密码后恢复

如果用户丢失受密码保护的 CPU 的密码，则可使用空传送卡删除受密码保护的程序。 空传送卡将擦除 CPU 内部的装载存储器。 随后可以将新的用户程序从 STEP 7 下载到 CPU 中。

有关创建和使用空传送卡的信息，请参见传送卡 (页 137)部分。

⚠ 警告

插入存储卡之前，请检查并确认 CPU 当前并未执行任何操作

如果将传送卡插入正在运行的 CPU 中，CPU 将进入 STOP 模式。 控制设备在不安全情况下运行时可能会出现故障，从而导致受控设备的意外运行。 这种意外运行可能会导致人员死亡、重伤和/或设备损坏。

插入传送卡前，请务必确保 CPU 处于 STOP 模式且程序处于安全状态。

将 CPU 设置为 RUN 模式之前，必须先取出传送卡。

设备配置

6

通过向项目中添加 CPU 和其它模块，可以为 PLC 创建设备组态。

① 通信模块 (CM) 或通信处理器 (CP)： 最多 3 个，分别插在插槽 101、102 和 103 中

② CPU： 插槽 1

③ CPU 的 PROFINET 端口

④ 信号板 (SB)、通信板 (CB) 或电池板 (BB)： 最多 1 个，插在 CPU 中

⑤ 数字或模拟 I/O 的信号模块 (SM)： 最多 8 个，分别插在插槽 2 到 9 中（CPU 1214C、CPU 1215C 和 CPU 1217C 允许使用 8 个；CPU 1212C 允许使用 2 个；CPU 1211C 不允许使用任何信号模块）

组态控制

S7-1200 的设备组态还支持“组态控制 (页 152)”，在此可以为项目组态一个最大组态，包括实际操作中可能用不到的模块。 此功能（有时也称作“选件处理”）允许用户组态一个最大组态，可供多个应用中所安装模块的变量使用。

6.1 插入 CPU

可以通过 Portal 视图或 STEP 7 的项目视图将 CPU 插入到项目中。

- 在视图中，选择“设备和网络”(Devices & Networks) 并单击“添加新设备”(Add new device)。
- 在项目视图中的项目名称下，双击“添加新设备”(Add new device)。

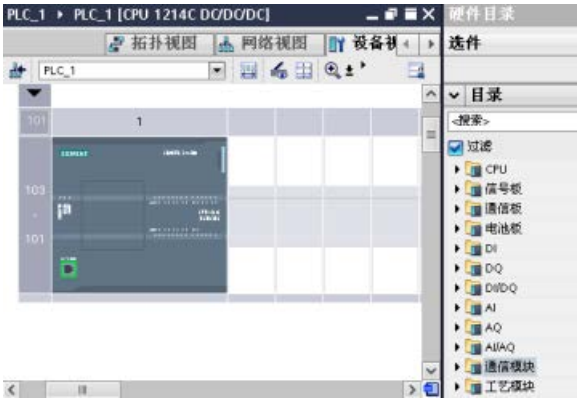
确保插入了列表中的正确型号和固件版本。通过从“添加新设备”(Add new device) 对话框中选择 CPU，可创建机架和 CPU。

说明

不能使用 STEP 7 V14 及更高版本将 V1.0 S7-1200 CPU 添加到项目中。

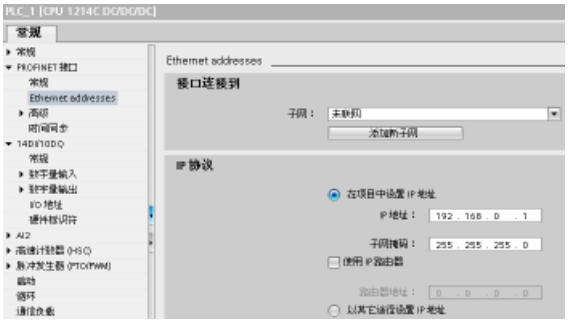
“添加新设备”对话框

硬件配置的设备视图



通过在设备视图中选择 CPU，可在巡视窗口中显示 CPU 属性。

CPU 不具有预组态的 IP 地址。设备配置期间必须为 CPU 手动分配 IP 地址。如果 CPU 连接到网络上的路由器，则也应输入路由器的 IP 地址。



6.2 上传已连接 CPU 的组态

STEP 7 提供两种上传已连接 CPU 的硬件配置的方法：

- 将已连接设备作为新站上传
- 组态未指定的 CPU 并检测已连接 CPU 的硬件配置

不过需要注意的是，第一种方法将同时上传已连接 CPU 的硬件配置和软件。

将设备作为新站上传

要将已连接设备作为新站上传，请按以下步骤操作：

1. 从项目树的“在线访问”(Online access) 节点中展开通信接口。
2. 双击“更新可访问的设备”(Update accessible devices)。
3. 从检测到的设备中选择 PLC。

4. 从 STEP 7 的“在线”(Online) 菜单中，选择“将设备作为新站上传（硬件和软件）”(Upload device as new station (hardware and software)) 菜单命令。

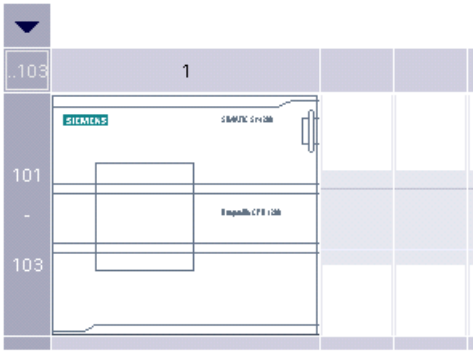
STEP 7 将同时上传硬件配置和程序块。

检测未指定 CPU 的硬件配置

如果已连接到 CPU，则可以将该 CPU（包括所有模块）的组态上传到用户项目中。 只需创建新项目并选择“未指定的 CPU”而不是选择特定的 CPU 即可。（也可通过从“新手上路”(First steps) 中选择“创建 PLC 程序”(Create a PLC program) 完全跳过设备组态。 STEP 7 即会自动创建一个未指定的 CPU。）

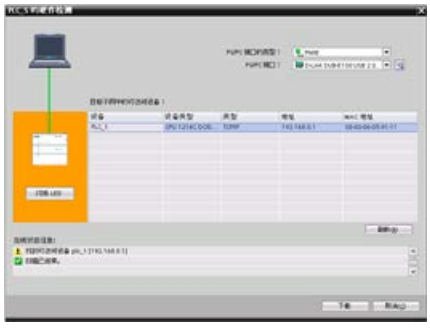
在程序编辑器中，从“在线”(Online) 菜单中选择“硬件检测”(Hardware detection) 命令。

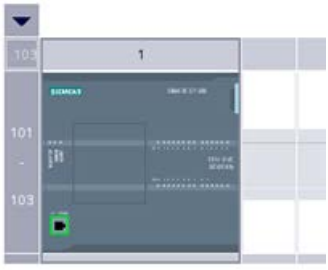
在设备组态编辑器中，选择用于检测所连设备组态的选项。



未指定该设备。
→ 请使用 硬件目录 指定 CPU.
→ 或 检测 相连设备的组态。

从在线对话框中选择 CPU 并单击 “加载”(Load) 按钮后，STEP 7 会上传 CPU 以及所有模块（SM、SB 或 CM）的硬件配置。 随后可以为 CPU 和模块 (页 163)组态参数。





6.3 将模块添加到组态

使用硬件目录将模块添加到 CPU：

- 信号模块 (SM) 提供附加的数字或模拟 I/O 点。 这些模块连接在 CPU 右侧。
- 信号板 (SB) 仅为 CPU 提供几个附加的 I/O 点。 SB 安装在 CPU 的前端。
- 电池板 1297 (BB) 可提供长期的实时时钟备份。 BB 安装在 CPU 的前端。
- 通信板 (CB) 提供附加的通信端口（如 RS485）。 CB 安装在 CPU 的前端。
- 通信模块 (CM) 和通信处理器 (CP) 提供附加的通信端口（如用于 PROFIBUS 或 GPRS）。 这些模块连接在 CPU 左侧。

要将模块插入到设备组态中，可在硬件目录中选择模块，然后双击该模块或将其拖到高亮显示的插槽中。 必须将模块添加到设备组态并将硬件配置下载到 CPU 中，模块才能正常工作。

表格 6-1　　将模块添加到设备组态中

模块	选择模块	插入模块	结果
SM			
SB、BB 或 CB			
CM 或 CP			

使用“组态控制”功能 (页 152)，用户可以添加信号模块和信号板到设备组态，虽然这样有可能与特定应用的实际硬件不符，但可用于共享通用用户程序、CPU 型号以及一些已组态模块的相关应用。

6.4　组态控制

6.4.1　组态控制的优点和应用

当您想创建一个要在多个不同安装中使用的自动化解决方案（机器）时，组态控制将发挥作用。

可加载 STEP 7 设备组态和用户程序到不同的已安装 PLC 组态。 仅需进行一些简单的调整，即可使 STEP 7 项目与实际安装对应。

6.4.2 组态集中安装和可选模块

使用 STEP 7 和 S7-1200 的组态控制功能，可以为标准机器组态一个最大组态，并可操作选用其中一部分组态的版本（选项）。《使用 STEP 7 组态 PROFINET》手册 (https://support.industry.siemens.com/cs/cn/zh/view/49948856)中将这些项目类型称为“标准机器项目”。

在启动程序块中编程的控制数据记录将通知 CPU 与组态相比实际安装中丢失了哪些模块，或是哪些模块位于与组态不同的插槽中。组态控制不会影响模块的参数分配。

只要用户能够从 STEP 7 的最大设备组态中获取实际组态，便可使用组态控制进行多种不同的灵活安装。

要激活组态控制并构建所需的控制数据记录，请按以下步骤操作：

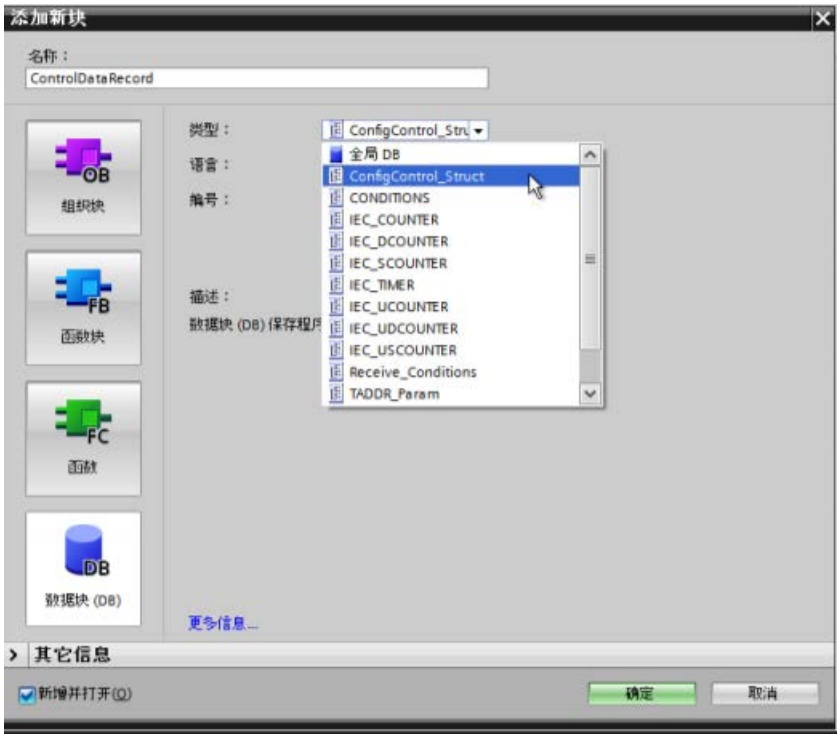
1. 也可以将 CPU 复位成出厂设置，以确保 CPU 中不存在不兼容的控制数据记录。
2. 在 STEP 7 的设备组态中选择 CPU。
3. 从 CPU 属性的“组态控制”(Configuration control) 节点中，选择“启用使用用户程序重新组态设备”(Enable reconfiguration of device with user program) 复选框。

4. 创建一个用于包含控制数据记录的 PLC 数据类型。将其组态为以下结构，包含 4 个用于存储组态控制信息的 USint，以及对应于 S7-1200 设备最大组态的插槽的其它 USint，操作如下：

ConfigControl_Struct

	名称	数据类型	默认值	注释
1	ConfigControl	Struct		
2	Block_length	USInt	16	Length of control data record, including header
3	Block_ID	USInt	196	Data record number
4	Version	USInt	5	
5	Subversion	USInt	0	
6	Slot_1	USInt	255	Assignment for CPU annex card/Actual annex card
7	Slot_2	USInt	255	Configured slot 2 / Assigned "real" slot
8	Slot_3	USInt	255	Configured slot 3 / Assigned "real" slot
9	Slot_4	USInt	255	Configured slot 4 / Assigned "real" slot
10	Slot_5	USInt	255	Configured slot 5 / Assigned "real" slot
11	Slot_6	USInt	255	Configured slot 6 / Assigned "real" slot
12	Slot_7	USInt	255	Configured slot 7 / Assigned "real" slot
13	Slot_8	USInt	255	Configured slot 8 / Assigned "real" slot
14	Slot_9	USInt	255	Configured slot 9 / Assigned "real" slot
15	Slot_101	USInt	255	Configured slot 101 / Assigned "real" slot
16	Slot_102	USInt	255	Configured slot 102 / Assigned "real" slot
17	Slot_103	USInt	255	Configured slot 103 / Assigned "real" slot

5. 为已创建的 PLC 数据类型创建一个数据块。



6. 在该数据块中，按如下所示组态 Block_length、Block_ID、版本以及次版本。根据是否存在插槽以及其在实际安装中的位置组态插槽的值：
   - 0：实际组态中不存在已组态的模块。（插槽为空。）
   - 1 到 9，101 到 103：已组态插槽的实际插槽位置
   - 255：STEP 7 设备组态在此插槽中不包含模块。

说明

组态控制不适用于信号板上的 HSC 和 PTO

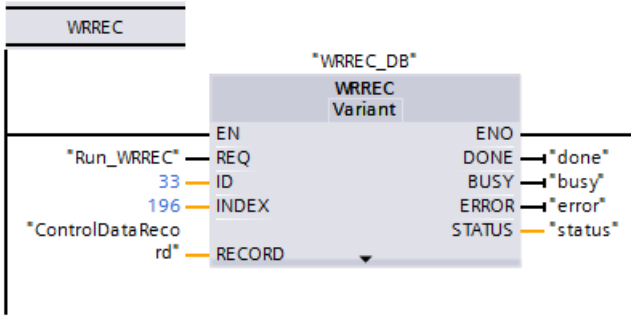
如果 CPU 中有一个信号板组态用于 HSC 或 PTO，则不得通过对组态控制数据记录中的 Slot_1 写“0”来将其禁用。如需使用组态控制，必须组态 CPU 的 HSC 和 PTO 设备。

ControlDataRecord

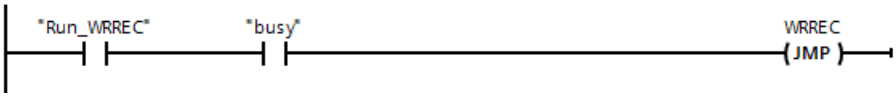
	名称	数据类型	启动值	注释
1	Static			
2	ConfigControl	Struct		
3	Block_length	USInt	16	Length of control data record, including header
4	Block_ID	USInt	196	Data record number
5	Version	USInt	5	
6	Subversion	USInt	0	
7	Slot_1	USInt	255	Assignment for CPU annex card/ Actual annex card
8	Slot_2	USInt	255	Configured slot 2 / Assigned "real" slot
9	Slot_3	USInt	255	Configured slot 3 / Assigned "real" slot
10	Slot_4	USInt	255	Configured slot 4 / Assigned "real" slot
11	Slot_5	USInt	255	Configured slot 5 / Assigned "real" slot
12	Slot_6	USInt	255	Configured slot 6 / Assigned "real" slot
13	Slot_7	USInt	255	Configured slot 7 / Assigned "real" slot
14	Slot_8	USInt	255	Configured slot 8 / Assigned "real" slot
15	Slot_9	USInt	255	Configured slot 9 / Assigned "real" slot
16	Slot_101	USInt	255	Configured slot 101 / Assigned "real" slot
17	Slot_102	USInt	255	Configured slot 102 / Assigned "real" slot
18	Slot_103	USInt	255	Configured slot 103 / Assigned "real" slot

有关如何分配插槽值的说明，请参见组态控制示例 (页 159)。

7. 在启动 OB 中，调用扩展的 WRREC（写入数据记录）指令，将创建的控制数据记录传送到硬件 ID 33 的索引 196。使用标签和 JMP（跳转）指令等待 WRREC 指令完成。
   程序段 1：



程序段 2：



说明

WRREC 指令在启动 OB 中传送完控制数据记录后组态控制才会生效。如果已启用组态控制但 CPU 不具有控制数据记录，则在退出 STARTUP 模式时会转到 STOP 模式。确保已将启动 OB 设置为传送控制数据记录。

模块排列

下表列出了插槽号分配情况：

插槽	模块
1	信号板或通信板（CPU 附件卡）
2 到 9	信号模块
101 到 103	通信模块

控制数据记录

控制数据记录 196 包含插槽分配并表示实际组态，如下所示：

Byte	元素	值	说明
0	块长度	16	Header
1	块 ID	196	
2	版本	5	
3	次版本	0	
4	CPU 附件卡的分配	实际附件卡，0 或 255*	控制元素 说明已将设备中的哪个实际插槽分配给每个单元中组态的插槽。
5	组态的插槽 2 的分配	实际插槽，0 或 255*	
...	...	...	
12	组态的插槽 9 的分配	实际插槽，0 或 255*	

Byte	元素	值	说明
13	组态的插槽 101 的分配	实际插槽或 255*	与信号模块不同，实际存在的通信模块的实际插槽必须与已组态的插槽相同。
14	组态的插槽 102 的分配	实际插槽或 255*	
15	组态的插槽 103 的分配	实际插槽或 255*	

***插槽值：**

0：实际组态中不存在已组态的模块。（插槽为空。）

1 到 9，101 到 103：已组态插槽的实际插槽位置

255：STEP 7 设备组态在此插槽中不包含模块。

说明

创建 PLC 变量类型的替代方法

作为创建自定义 PLC 变量类型的替代方法，您可以使用控制数据记录的所有结构元素来直接创建数据块。甚至可以在该数据块中组态多个结构以用作多个控制数据记录组态。两种实现方式都可在启动期间有效传输控制数据记录。

准则

请遵守以下准则：

- 组态控制不支持通信模块的位置更改。也不能使用组态控制来停用通信模块。插槽 101 到 103 的控制数据记录插槽位置必须与实际安装对应。如果未在设备组态中为插槽组态模块，在控制数据记录中为该插槽位置输入 255。如果已为插槽组态了模块，输入组态的插槽作为该位置的实际插槽。
- F-I/O 模块不支持组态控制。F-I/O 模块的控制数据记录插槽位置必须与 F-I/O 模块的已组态插槽位置相同。如果想要通过控制数据记录移动或删除组态的 F-I/O 模块，则所有实际安装的 F-I/O 模块都将发生“参数分配”错误，并且均不允许进行交换。
- 在已填充（已使用）的插槽之间不能有嵌入式空（未使用）插槽。例如，如果实际组态在插槽 4 中有一个模块，则实际组态在插槽 2 和 3 中也必须有模块。相应地，如果实际组态在插槽 102 中有一个通信模块，则实际组态在插槽 101 中也必须有一个模块。
- 如果已启用组态控制，却没有控制数据记录，则 CPU 仍未做好运行准备。如果启动 OB 未传送一个有效的控制数据记录，则 CPU 从启动模式返回到 STOP 模式。CPU 在这种情况下不会初始化集中式 I/O，并将在诊断缓冲区中输入转到 STOP 模式的原因。
- CPU 将成功传送的控制数据记录保存在保持性存储器中，也就是说，在不更改组态的情况下重启时无需重新写入控制数据记录 196。

- 每个实际插槽只能在控制数据记录中出现一次。
- 只能将一个实际插槽分配给一个已组态插槽。

说明

修改组态

使用已修改的组态写入控制数据记录将触发 CPU 的下述自动响应：存储器通过后续启动复位并采用已修改组态。

由于该响应，CPU 将删除原始的控制数据记录并保持性地保存新的控制数据记录。

运行期间的特性

对于在线显示以及诊断缓冲区中的显示（模块正常或模块故障），STEP 7 都将使用设备组态而不是不同的实际组态。

示例：一个模块输出诊断数据。该模块组态插入插槽 4，但实际却插入插槽 3。在线视图将指示已组态的插槽 4 存在故障。在实际组态中，插槽 3 中的模块通过 LED 显示屏指示错误。

如果已在控制数据记录中将模块组态为丢失（0 个条目），则自动化系统会按如下方式运行：

- 在控制数据记录中被标识为不存在的模块不会提供诊断并且它们的状态始终为正常。值状态正常。
- 对不存在的输出量的直接写访问或对不存在的输出量的过程映像的写访问将不产生任何影响；CPU 不会报告任何访问错误。
- 对不存在的输入量的直接读访问或对不存在的输入量的过程映像的读访问将为每个输入生成一个“0”值；CPU 不会报告任何访问错误。
- 向不存在的模块写入数据记录将不产生任何影响；CPU 不会报告任何错误。
- 尝试从不存在的模块读取数据记录将生成错误，因为 CPU 无法返回一个有效的数据记录。

错误消息

如果在写入控制数据记录期间发生错误，CPU 将返回下列错误消息：

错误代码	含义
16#80B1	非法长度；控制数据记录中的长度信息不正确。
16#80B5	未分配组态控制参数
16#80E2	数据记录在错误的 OB 中传送。该数据记录必须在启动 OB 中传送。

错误代码	含义
16#80B0	控制数据记录的块类型（字节 2）不等于 196。
16#80B8	参数错误；模块指示存在无效参数，例如： ● 控制数据记录试图修改通信模块或通信附件卡的组态。通信模块和通信附件卡的实际组态必须等于 STEP 7 组态。 ● 为 STEP 7 项目中未组态插槽分配的值不等于 255。 ● 为已组态插槽分配的值超出范围。 ● 分配的组态具有一个“内部”空闲插槽，例如，插槽 n 已分配而插槽 n-1 未分配。

6.4.3 组态控制示例

本示例介绍了由一个 CPU 和三个 I/O 模块组成的配置。在第一次实际安装中，插槽 3 处的模块并不存在，因此可使用组态控制将其“隐藏”。

第二次安装时，应用将包括最初隐藏的模块，但现在该模块位于最后一个插槽中。修改后的控制数据记录可提供有关模块插槽分配的信息。

示例：使用已组态但未使用模块的实际安装

设备组态包含实际安装中可能存在的所有模块（最大组态）。这种情况下，在设备组态中应位于插槽 3 中的模块在实际组态中不存在。

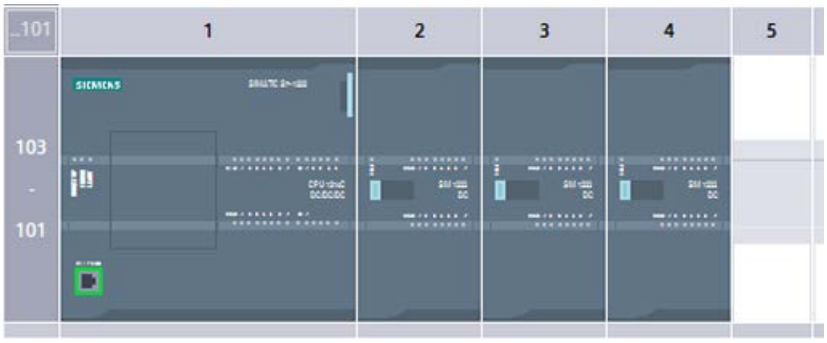


图 6-1　最大安装（即装有三个信号模块）的设备组态

图 6-2　插槽 3 中组态的模块不存在，以及插槽 4 的组态模块位于实际插槽 3 中时的实际安装

要指示丢失模块的不存在，必须在控制数据记录中使用 0 组态插槽 3。

ControlDataRecord

		名称	数据类型	启动值	注释
1		Static			
2		ConfigControl	Struct		
3		Block_length	USInt	16	Length of control data record, including header
4		Block_ID	USInt	196	Data record number
5		Version	USInt	5	
6		Subversion	USInt	0	
7		Slot_1	USInt	255	Assignment for CPU annex card/Actual annex card
8		Slot_2	USInt	2	Configured slot 2 / Assigned "real" slot
9		Slot_3	USInt	0	Configured slot 3 / Assigned "real" slot
10		Slot_4	USInt	3	Configured slot 4 / Assigned "real" slot
11		Slot_5	USInt	255	Configured slot 5 / Assigned "real" slot
12		Slot_6	USInt	255	Configured slot 6 / Assigned "real" slot
13		Slot_7	USInt	255	Configured slot 7 / Assigned "real" slot
14		Slot_8	USInt	255	Configured slot 8 / Assigned "real" slot
15		Slot_9	USInt	255	Configured slot 9 / Assigned "real" slot
16		Slot_101	USInt	255	Configured slot 101 / Assigned "real" slot
17		Slot_102	USInt	255	Configured slot 102 / Assigned "real" slot
18		Slot_103	USInt	255	Configured slot 103 / Assigned "real" slot

示例：随后将模块添加到不同插槽中的实际安装

在第二个示例中，设备组态中应位于插槽 3 中的模块在实际安装中存在，但位于插槽 4 中。

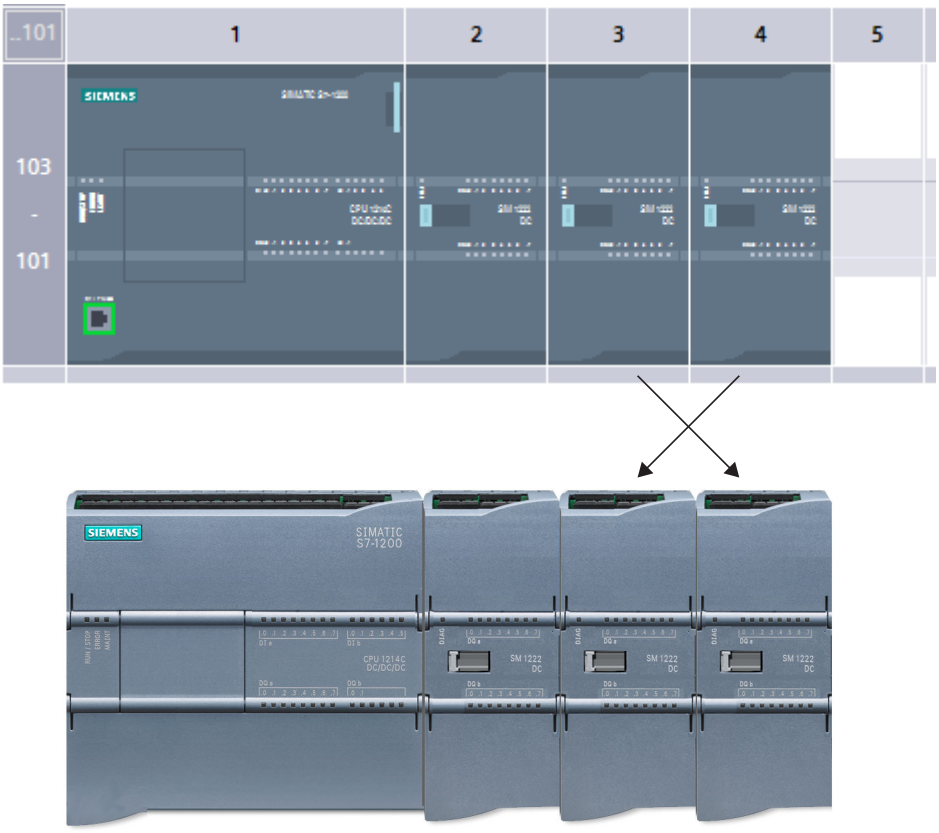


图 6-3　插槽 3 和 4 中的模块交换后，设备组态与实际安装的比较

要将设备组态与实际安装关联，可编辑控制数据记录，将模块分配到正确的插槽位置。

ControlDataRecord

	名称	数据类型	启动值	注释
1	Static			
2	ConfigControl	Struct		
3	Block_length	USInt	16	Length of control data record, including header
4	Block_ID	USInt	196	Data record number
5	Version	USInt	5	
6	Subversion	USInt	0	
7	Slot_1	USInt	255	Assignment for CPU annex card/Actual annex ...
8	Slot_2	USInt	2	Configured slot 2 / Assigned "real" slot
9	Slot_3	USInt	4	Configured slot 3 / Assigned "real" slot
10	Slot_4	USInt	3	Configured slot 4 / Assigned "real" slot
11	Slot_5	USInt	255	Configured slot 5 / Assigned "real" slot
12	Slot_6	USInt	255	Configured slot 6 / Assigned "real" slot
13	Slot_7	USInt	255	Configured slot 7 / Assigned "real" slot
14	Slot_8	USInt	255	Configured slot 8 / Assigned "real" slot
15	Slot_9	USInt	255	Configured slot 9 / Assigned "real" slot
16	Slot_101	USInt	255	Configured slot 101 / Assigned "real" slot
17	Slot_102	USInt	255	Configured slot 102 / Assigned "real" slot
18	Slot_103	USInt	255	Configured slot 103 / Assigned "real" slot

6.5 更改设备

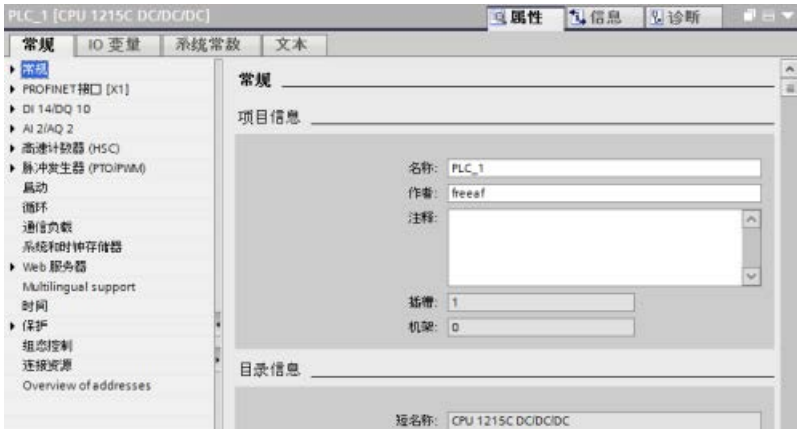
您可以更改已组态 CPU 或模块的设备类型。在设备组态中，右键单击设备并从上下文菜单中选择“更改设备”(Change device)。在随后出现的对话框中，导航到您想要更换的 CPU 或模块并选择。“更改设备”(Change device) 对话框将显示两个设备之间的兼容性信息。

有关在不同 CPU 版本之间更改设备的注意事项，请参见用 V4.x CPU 更换 V3.0 CPU (页 1667)。

6.6 组态 CPU 的运行

6.6.1 概述

要组态 CPU 的运行参数，在设备视图（整个 CPU 周围的蓝色轮廓）中选择 CPU，并使用巡视窗口的 “属性”(Properties) 选项卡。



表格 6-2 CPU 属性

属性	说明
PROFINET 接口	设置 CPU 的 IP 地址和时间同步
DI、DO 和 AI	组态本地（板载）数字量和模拟量 I/O 的特性（例如，数字量输入滤波时间和对 CPU 停止的数字量输出响应）。
高速计数器 (页 559)和脉冲发生器 (页 494)	启用并组态高速计数器 (HSC, High-Speed Counter) 以及用于脉冲串运行 (PTO, Pulse-Train Operation) 和脉冲宽度调制 (PWM, Pulse-Width Modulation) 的脉冲发生器 将 CPU 或信号板的输出组态为脉冲发生器时（与 PWM 或运动控制指令配合使用），会从 Q 存储器中移除相应的输出地址，并且这些地址在用户程序中不能用于其它用途。如果用户程序向用作脉冲发生器的输出写入值，则 CPU 不会将该值写入到物理输出。

属性	说明
启动 (页 84)	**上电后启动：**选择进行关到开转换之后 CPU 的特性，如在 STOP 模式下启动或在暖启动后转到 RUN 模式
	支持的硬件兼容性：组态所有系统组件（SM、SB、CM、CP 和 CPU）的替换策略： ● 允许可接受的替换 ● 允许任何替换（默认） 各模块内部均包含基于 I/O 数量、电气兼容性以及其它对应比较点的替换兼容性要求。例如，16 通道的 SM 是 8 通道 SM 的可接受替换设备，但 8 通道 SM 不是 16 通道 SM 的可接受替换设备。如果选择“允许可接受的替换”，则 STEP 7 会实施替换规则；否则，STEP 7 将允许任何替换。
	分布式 I/O 的参数分配时间：组态将分布式 I/O 切换到在线状态所允许的最长时间（默认值：60000 ms）。（在启动期间，CM 和 CP 会从 CPU 接收供电和通信参数。该分配时间是连接到 CM 或 CP 的 I/O 切换到在线状态所允许的时间。） 无论分配时间是多少，分布式 I/O 切换为在线状态后，CPU 会立即进入 RUN 模式。如果分布式 I/O 未在这一时间内切换为在线状态，则 CPU 仍会在没有分布式 I/O 的情况下进入 RUN 模式。 **注：**如果组态使用 CM 1243-5（PROFIBUS 主站），不要将此参数设置为低于 15 秒（15000 ms），以确保模块切换到在线状态。
	OB 应可中断：组态 CPU 中（所有 OB）的 OB 执行是否可中断 (页 99)
周期 (页 103)	定义最大循环时间或固定的最小循环时间
通信负载	分配专门用于通信任务的 CPU 时间百分比
系统和时钟存储器 (页 108)	启用一个字节用于“系统存储器”功能，并启用一个字节用于“时钟存储器”功能（其中每个位都按预定义频率打开和关闭）
Web 服务器 (页 1025)	启用和组态 Web 服务器功能。
时钟	选择时区并组态夏令时
多语言支持 (页 168)	针对每种 Web 服务器用户界面显示语言，为 Web 服务器分配一种项目语言，用于显示诊断缓冲区条目文本。
保护 (页 198)	设置用于访问 CPU 的读/写保护和密码
组态控制 (页 152)	启用为不同的实际设备组态组态可控制的主站设备组态
连接资源 (页 807)	提供可用于 CPU 的通信连接资源汇总以及已组态的连接资源数
地址总览	提供已为 CPU 组态的 I/O 地址的摘要

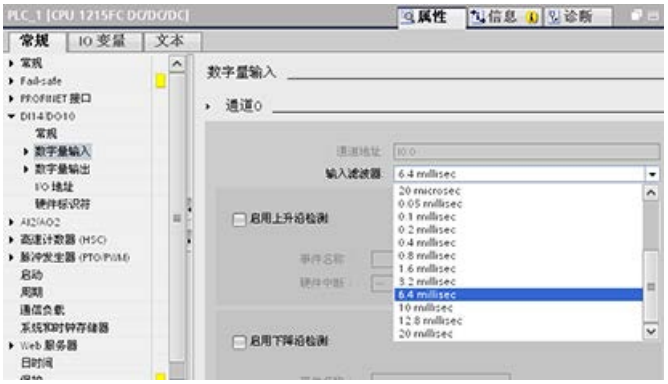
6.6.2 组态数字量输入滤波时间

数字量输入滤波器可防止程序响应输入信号中的意外快速变化，这些变化可能因开关触点跳跃或电气噪声产生。6.4 ms 的默认滤波时间能够阻止典型机械触点发生意外转换。应用中的不同点可能需要较短的滤波时间来检测和响应快速传感器的输入，或需要较长的滤波时间来阻止较慢的触点跳跃或较长的脉冲噪声。

6.4 ms 的输入滤波时间表示单个信号从“0”变为“1”，或从“1”变为“0”必须持续约 6.4 ms 才能够被检测到，而短于约 6.4 ms 的单个高脉冲或低脉冲不会被检测到。如果输入信号在“0”和“1”之间切换的时间短于滤波时间，则在旧值脉冲基础上新值脉冲的累积时间超过滤波时间时，用户程序中的输入点值可能会发生变化。

数字量输入滤波器的工作方式如下：

- 输入“1”时，滤波器进行加计数，达到滤波时间时停止。计数时间达到滤波时间时，映像寄存器的点将从“0”变为“1”。
- 输入“0”时，滤波器进行减计数，达到“0”时停止。计数达到“0”时，映像寄存器的点将从“1”变为“0”。
- 如果输入反复变化，计数器将交替进行加计数和减计数。当计数的净累积量达到滤波时间或“0”时，映像寄存器会发生变化。
- “0”比“1”多的快速变化信号最终将变为“0”，如果“1”比“0”多，映像寄存器最终将变为“1”。



每一个输入点都有一个适用于所有应用的滤波器组态：过程输入、中断、脉冲捕捉和 HSC 输入。要组态输入滤波时间，选择“数字量输入”(Digital Inputs)。

数字量输入的默认滤波时间为 6.4 ms。可以从输入滤波器下拉列表中选择滤波时间。有效滤波时间范围为 0.1 us 到 20.0 ms。

⚠ 警告
对数字量输入通道的滤波时间进行更改的风险 如果数字量输入通道的滤波时间更改自以前的设置，则新的“0”电平输入值可能需要保持长达 20.0 ms 的时间，然后滤波器才会完全响应新输入。在此期间，可能不会检测到持续时间少于 20.0 ms 的短“0”脉冲事件或对其计数。 滤波时间的这种更改会引发意外的机械或过程操作，这可能会导致人员死亡、重伤和/或设备损坏。 为了确保新的滤波时间立即生效，必须关闭 CPU 电源后再开启。

为用作 HSC 的数字量输入组态滤波时间

对于设置为高速计数器 (HSC) 的输入，需要将输入滤波时间设置为适合的值以避免计数遗漏。

Siemens 建议以下设置：

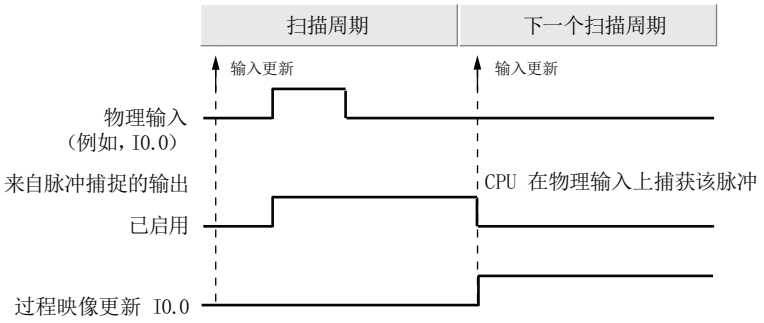
HSC 的类型	建议的输入滤波时间
1 MHz	0.1 微秒
100 kHz	0.8 微秒
30 kHz	3.2 微秒

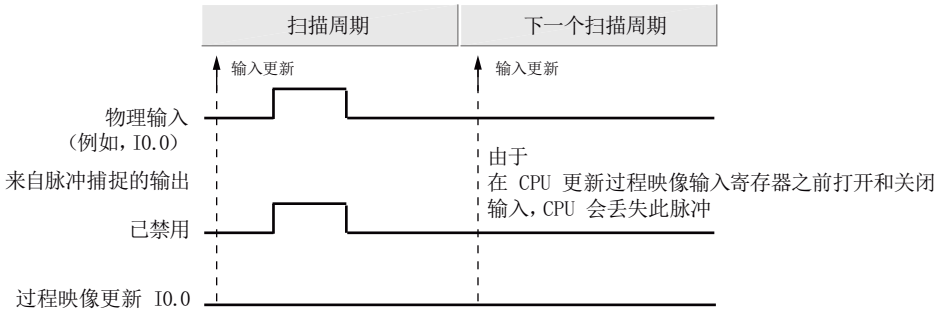
6.6.3 脉冲捕捉

S7-1200 CPU 为数字量输入点提供脉冲捕捉功能。通过脉冲捕捉功能可以捕捉高电平脉冲或低电平脉冲。此类脉冲出现的时间极短，CPU 在扫描周期开始读取数字量输入时，可能无法始终看到此类脉冲。

启用输入的脉冲捕捉时，将锁存并保持输入状态的更改，直至下一个输入周期更新。这可以确保捕捉并保持持续时间较短的脉冲，直至 CPU 读取输入。

下图所示为启用和禁用脉冲捕捉时 S7-1200 CPU 的基本操作：

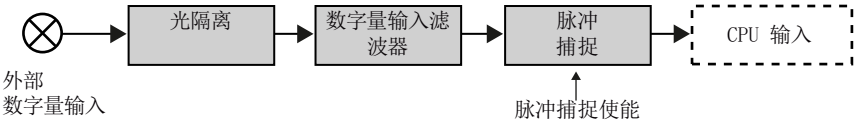




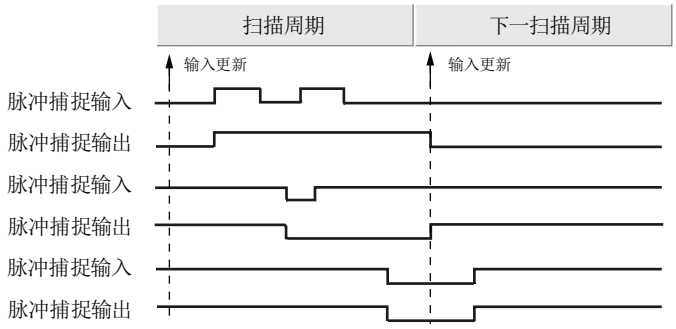
说明

由于脉冲捕捉功能在通过输入滤波器后会对输入进行操作，因此必须调整输入滤波器时间，以使滤波器不会消除脉冲。

下图显示数字量输入电路方框图：



下图显示启用脉冲捕捉功能时对各种不同输入条件的响应。如果在某一特定扫描中存在一个以上脉冲，仅读取第一个脉冲。如果在某一特定扫描中有多个脉冲，则应当使用上升/下降沿中断事件：



6.7 组态多语言支持

多语言支持设置可为 S7-1200 Web 服务器 (页 1025)的每种用户界面语言分配两类项目语言中的一种。您也可以不为用户界面语言组态项目语言。

什么是项目语言？

项目语言即为 TIA Portal 用来将用户自定义项目文本显示为程序段注释和块注释的语言。

在 TIA Portal 中，从项目树中所选项目的 **“工具 > 项目语言”(Tools > Project languages)** 菜单命令中选择项目语言。

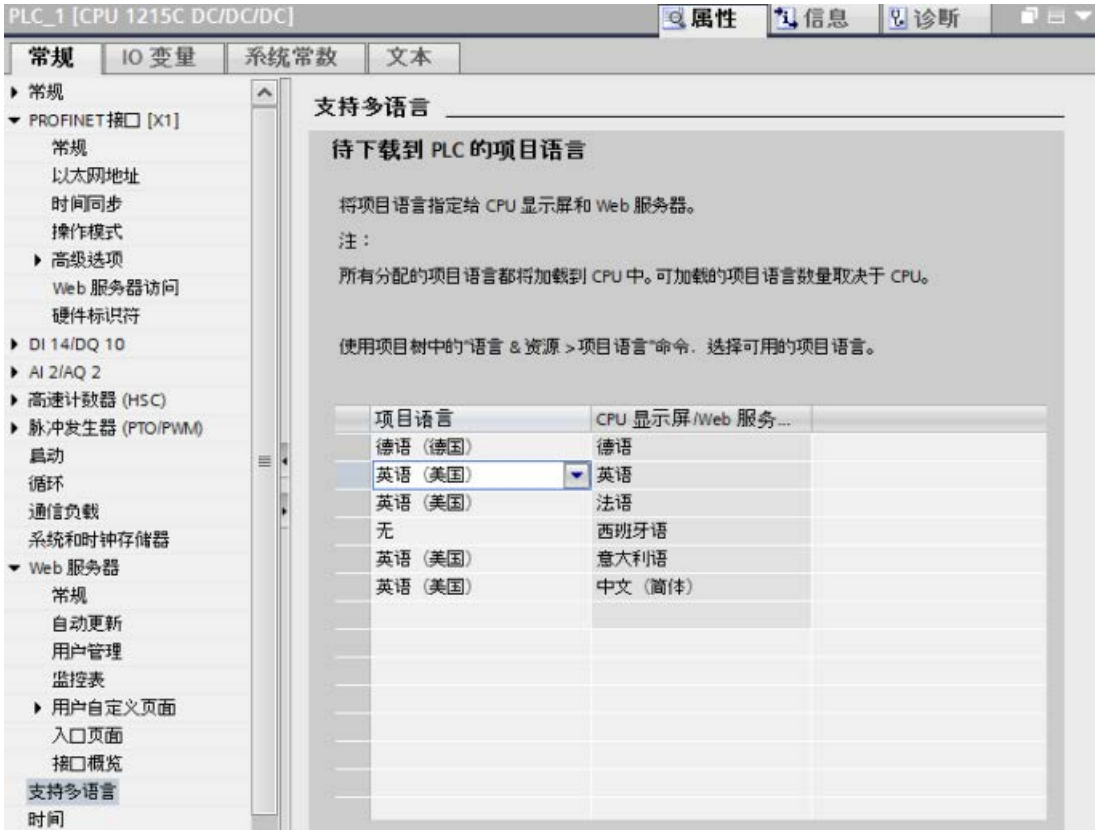
随后可以从 **“工具 > 项目文本”(Tools > Project texts)** 菜单命令用每种项目语言组态用户文本（例如程序段注释和块注释）。当更改 TIA Portal 用户界面语言时，程序段注释、块注释和其它多语言项目文本会用相应的项目语言显示。从 **“选项 > 设置”(Options > Settings)** 项目语言菜单命令中设置 TIA Portal 用户界面语言。

项目语言和项目文本也可从项目树的 **“语言和资源”(Languages & resources)** 节点组态。

Web 服务器可以使用一到两种 STEP 7 项目语言来显示诊断缓冲区消息。

项目语言与 Web 服务器用户界面语言的对应关系

Web 服务器支持与 TIA Portal 一样的用户界面语言；不过，它最多只支持两种项目语言。可根据 Web 服务器的用户界面语言为其组态两种项目语言中的一种，用于显示诊断缓冲器文本条目。这些设置可在 CPU 设备组态的“多语言支持”(Multilingual support) 属性中组态。（无法从 Web 服务器查看程序段注释、程序块注释以及其它多语言文本。）



在“多语言支持”(Multilingual support) 属性中，右侧的用户界面语言不可编辑。这些语言均为可用于 TIA Portal 和 Web 服务器用户界面的预定义语言。“分配项目语言”(Assign project language) 设置可组态，可选择已组态的两种项目语言之一，也可以选择“无”(None)。由于 S7-1200 CPU 仅支持两种项目语言，因此，在所有支持的用户界面语言范围内，组态的项目语言不能与用户界面语言相同。

在下述组态中，当 Web 服务器用户界面为德语时，Web 服务器用德语显示诊断缓冲区条目 (页 1045)；当 Web 服务器用户界面为西班牙语时，不显示任何诊断缓冲区事件的文本；对于所有其它语言，都用英语显示诊断缓冲区条目。

6.8 组态模块的参数

要组态模块的运行参数，请在设备视图中选择模块，并使用巡视窗口的“属性”(Properties) 选项卡组态模块的参数。

组态信号模块 (SM) 或信号板 (SB)

信号模块和信号板的设备组态可用于组态以下各项：

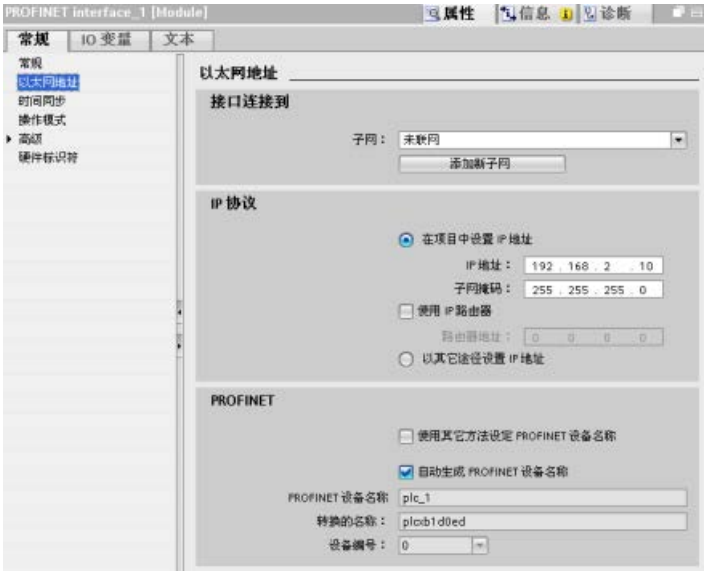
- 数字量 I/O： 可组态各个输入用于上升沿检测或下降沿检测（将每个检测分别与一个事件和硬件中断进行关联），或用于在输入过程映像的下一次更新期间进行“脉冲捕捉”（瞬时脉冲之后停留）。 输出可使用冻结值或替换值。
- 模拟量 I/O： 为各个输入组态参数，如测量类型（电压或电流）、范围和平滑化，也可启用下溢或上溢诊断。 模拟量输出提供诸如输出类型（电压或电流）之类的参数，也可用于诊断，例如，短路（针对电压输出）或上/下限诊断。 请勿在“属性”(Properties) 对话框中组态以工程单位表示的模拟量输入和模拟量输出的范围。 必须按照主题“模拟值的处理 (页 118)”的说明在程序逻辑中进行相应处理。
- I/O 地址： 组态用于设置模块的输入和输出的起始地址。 您还可以将输入和输出分配给过程映像分区（PIP0、PIP1、PIP2、PIP3、PIP4）或自动更新，或者不使用过程映像分区。 有关过程映像和过程映像分区的说明，请参见“执行用户程序 (页 81)”。

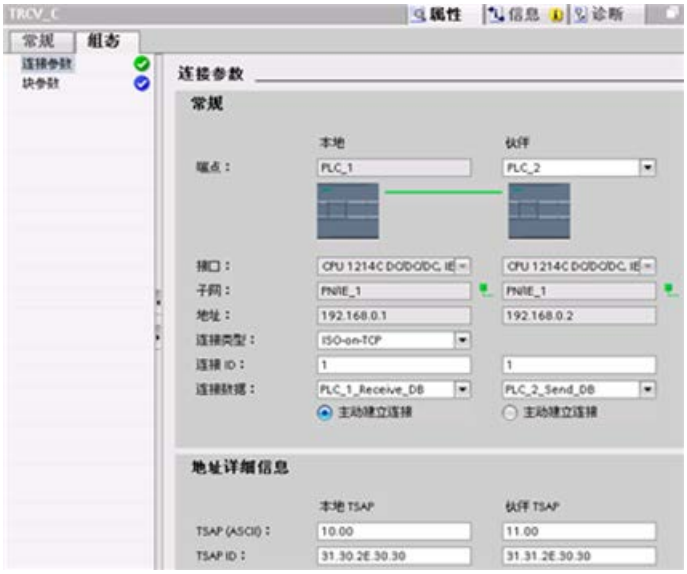
组态通信接口（CM、CP 或 CB）

根据通信接口的类型组态网络参数。

6.9 组态 CPU 以进行通信

S7-1200 的设计旨在解决您的通信和联网需求，不仅支持最简单的网络，而且支持更复杂的网络。 S7-1200 还提供允许您与其他设备通信的工具，例如，使用自身通信协议的打印机和秤。

	使用设备组态的“网络视图”(Network view) 可以在项目中的各个设备之间创建网络连接。创建网络连接之后，使用巡视窗口的“属性”(Properties) 选项卡可组态网络的参数。 更多相关信息，请参见“创建网络连接”(页 811)。
	在“属性”(Properties) 窗口中，选择“以太网地址”(Ethernet addresses) 组态条目。 STEP 7 会显示以太网地址组态对话框，该对话框可将软件项目与接收该项目的 CPU 的 IP 地址关联。 注：S7-1200 CPU 不具有预组态的 IP 地址。必须手动为 CPU 分配 IP 地址。 更多相关信息，请参见“分配 Internet 协议 (IP) 地址” (页 815)。

	对于 TCP、ISO-on-TCP 和 UDP 以太网协议，使用指令（TSEND_C、TRCV_C 或 TCON）的“属性”(Properties) 组态“本地/伙伴”连接。 该图显示了 ISO-on-TCP 连接“组态”(Configuration) 选项卡中的“连接属性”。 更多相关信息，请参见“组态本地/伙伴连接路径” (页 812)。
	完成组态后，将项目下载到 CPU。下载项目时会组态所有 IP 地址。 更多相关信息，请参见“测试 PROFINET 网络” (页 822)。

说明

要建立与 CPU 的连接，网络接口卡 (NIC) 和 CPU 的网络类别和子网必须相同。可以设置网络接口卡的 IP 地址使其与 CPU 的默认 IP 地址匹配，也可以更改 CPU 的 IP 地址，使其与网络接口卡的网络类别和子网匹配。

有关如何实现这一操作的信息，请参见“分配 Internet 协议 (IP) 地址” (页 815)。

6.10 时间同步

日时钟的时钟同步旨在使所有本地时钟与同一个主时钟同步。主时钟会在初始阶段同步本地时钟，而且还会定期重新执行时钟同步以免随时间发生偏差而受到影响。

对于 S7-1200 及其本地基本组件，只有 CPU 和部分 CP 模块的日时钟需要同步。可以组态 CPU 的日时钟以与外部主时钟同步。外部主时钟可使用 NTP 服务器或通过 S7-1200（与包含主时钟的 SCADA 系统相连）的本地机架中的 CP 提供日时钟。

有关所有支持时间同步功能的 S7-1200 CP (https://support.industry.siemens.com/cs/cn/zh/ps) 的详细信息，请参见西门子工业在线支持的 S7-1200 CP 产品支持部分。

设置日时钟

有以下四种方式设置 S7-1200 CPU 中的日时钟：

- 使用 NTP 服务器 (页 825)
- 使用 STEP 7
- 通过用户程序
- 使用 HMI 面板

通过选中“CPU 与设备模块同步”(CPU synchronizes the modules of the device) 复选框将 CP 模块组态为与 CPU 时钟时间同步，如下所示：

默认情况下，既不启用“通过 NTP 服务器设置时间同步”，也不启用“CP 时钟与 CPU 时钟时间同步”。

可以单独启用 CPU 时钟的时间同步和 CP 时钟的时间同步。这样一来，当通过任意上述方法设置 CPU 的时钟时，便可启用通过 CPU 设置 CP 时钟的时间同步。

可以使用 NTP 服务器选择更新时间间隔。NTP 服务器的更新时间间隔默认设为 10 秒。

在某个模块中激活时间同步后，如果未选中 CPU 的“时间同步”(Time synchronization) 对话框中的“CPU 与设备模块同步”(CPU synchronizes the modules of the device) 复选框，则

STEP 7 会提示用户进行勾选。如果组态了多个主时钟源用于时间同步（例如，在多个 CP 上激活时间同步，或者在 CPU 和模块上都激活了时间同步），STEP 7 也会提醒用户。

说明

在 CP 上激活时间同步会导致 CP 设置 CPU 的时钟。

如果在 CPU“时间同步”(Time synchronization) 对话框中选中“CPU 与设备模块同步”(CPU synchronizes the modules of the device)，则 CPU 为时间主站。随后 CP 模块将与 CPU 的时钟同步。

说明

只能为 CPU 组态一个时间源。CPU 从多个源中（例如，NTP 服务器或 CP 模块）接收时间同步会造成时间更新冲突。来自多个源的时间同步会对基于日时钟的指令和事件造成不利影响。

编程概念 7

7.1 设计 PLC 系统的指南

设计 PLC 系统时，可从若干方法和标准中进行选择。 下列常规指南可应用到许多设计项目中。 当然，必须遵守您自己公司程序的指令、自身培训以及当地已被接受的实践。

表格 7-1 设计 PLC 系统的指南

建议步骤	任务
对过程或机器进行分区	将过程或机器划分为彼此独立的部分。 这些分区会确定控制器之间的边界，并影响功能描述规范和资源的分配。
创建功能规范	写下过程或机器的每一部分（如 I/O 点）的操作说明、操作的功能描述、在允许进行每个执行器（如螺线管、电机或驱动器）的操作之前必须实现的状态、操作员界面的描述以及过程或机器其它部分的任何接口。
设计安全电路	出于安全考虑，标识任何可能需要硬接线逻辑的设备。 请记住，控制设备在不安全方式下可能会出现故障，可能会造成意外启动或机械运转变化。 其中意外或错误的机械运转可能会导致人员的身体伤害或重大的财产损失，请考虑实施机电替代装置（其独立于 PLC 运行）以防止不安全的运行。 安全电路的设计中应包含以下任务： ● 标识任何可能造成危险的不正确或意外的执行器操作。 ● 标识可确保操作不危险的条件，并确定如何独立于 PLC 检测这些条件。 ● 标识上电和断电时 PLC 如何影响过程，并标识检测错误的方式和时间。 此信息仅用于设计正常和预期的异常操作。 出于安全考虑，不应依赖此“最佳情况”方案。 ● 设计可独立于 PLC 来阻止危险运行的手动或机电安全替代装置。 ● 从独立于 PLC 的电路提供相应状态信息，以便程序和任何操作员界面具有必要的信息。 ● 标识针对过程安全运行的任何其它安全相关要求。
规划系统安全	确定访问相关过程所需的保护 (页 198)级别。 可以对 CPU 和程序块进行密码保护，以防受到未经授权的访问。
指定操作员站	根据功能规范的要求，创建以下操作员站的绘图： ● 显示与过程或机器相关的每个操作员站的位置的总览图。 ● 操作员站中设备的机械布局图，如显示屏、开关和灯。 ● 包含 PLC 和信号模块中相关 I/O 的电气图。

建议步骤	任务
创建组态图	根据功能规范的要求，创建控制设备的组态图： ● 显示与过程或机器相关的每个 PLC 位置的总览图。 ● 每个 PLC 和任何 I/O 模块的机械布局图，其中包括任何控制柜及其它设备。 ● 每个 PLC 和任何 I/O 模块的电气图，其中包括设备模型号、通信地址和 I/O 地址。
创建符号名称的列表	创建绝对地址的符号名称列表。 不仅包括物理 I/O 信号，也包括要在程序中使用的其它元素（如变量名）。

7.2 构建用户程序

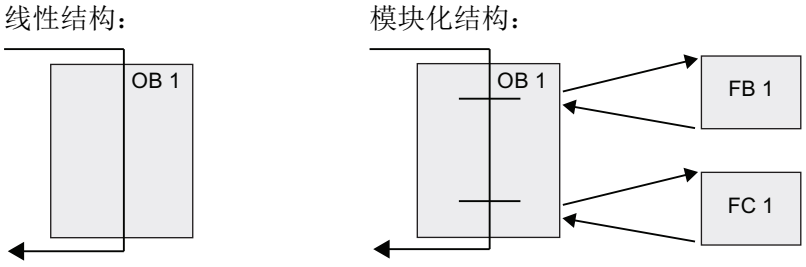
创建用于自动化任务的用户程序时，需要将程序的指令插入代码块中：

- 组织块 (OB) 对应于 CPU 中的特定事件，并可中断用户程序的执行。 用于循环执行用户程序的默认组织块 (OB 1) 为用户程序提供基本结构。 如果程序中包括其它 OB，这些 OB 会中断 OB 1 的执行。其它 OB 可执行特定功能，如用于启动任务、用于处理中断和错误或者用于按特定的时间间隔执行特定的程序代码。
- 功能块 (FB) 是从另一个代码块（OB、FB 或 FC）进行调用时执行的子例程。 调用块将参数传递到 FB，并标识可存储特定调用数据或该 FB 实例的特定数据块 (DB)。 更改背景 DB 可使通用 FB 控制一组设备的运行。 例如，借助包含每个泵或阀门的特定运行参数的不同背景数据块，一个 FB 可控制多个泵或阀。
- 功能 (FC) 是从另一个代码块（OB、FB 或 FC）进行调用时执行的子例程。 FC 不具有相关的背景 DB。 调用块将参数传递给 FC。 FC 中的输出值必须写入存储器地址或全局 DB 中。

为用户程序选择结构类型

根据实际应用要求，可选择线性结构或模块化结构用于创建用户程序：

- 线性程序按顺序逐条执行用于自动化任务的所有指令。 通常，线性程序将所有程序指令都放入用于循环执行程序的 OB (OB 1) 中。
- 模块化程序调用可执行特定任务的特定代码块。 要创建模块化结构，需要将复杂的自动化任务划分为与过程的工艺功能相对应的更小的次级任务。 每个代码块都为每个次级任务提供程序段。 通过从另一个块中调用其中一个代码块来构建程序。



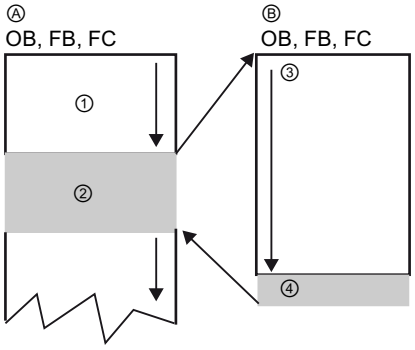
通过创建可在用户程序中重复使用的通用代码块，可简化用户程序的设计和实现。 使用通用代码块具有许多优点：

- 可为标准任务创建能够重复使用的代码块，如用于控制泵或电机。 也可以将这些通用代码块存储在可由不同的应用或解决方案使用的库中。
- 将用户程序构建到与功能任务相关的模块化组件中，可使程序的设计更易于理解和管理。 模块化组件不仅有助于标准化程序设计，也有助于使更新或修改程序代码更加快速和容易。
- 创建模块化组件可简化程序的调试。 通过将整个程序构建为一组模块化程序段，可在开发每个代码块时测试其功能。
- 创建与特定工艺功能相关的模块化组件，有助于简化对已完成应用程序的调试，并减少调试过程中所用的时间。

7.3 使用块来构建程序

通过设计 FB 和 FC 执行通用任务，可创建模块化代码块。然后可通过由其它代码块调用这些可重复使用的模块来构建程序。调用块将设备特定的参数传递给被调用块。

当一个代码块调用另一个代码块时，CPU 会执行被调用块中的程序代码。执行完被调用块后，CPU 会继续执行调用块。继续执行该块调用之后的指令。



A 调用块

B 被调用（或中断）块

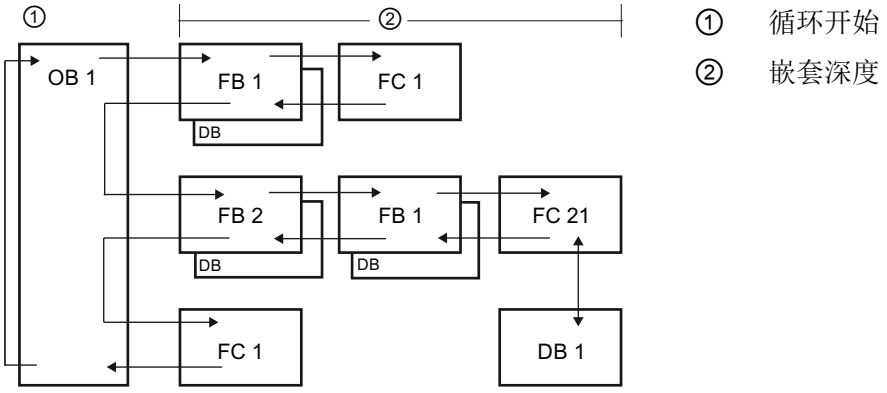
① 程序执行

② 用于触发其它块执行的指令或事件

③ 程序执行

④ 块结束（返回到调用块）

可嵌套块调用以实现更加模块化的结构。在以下示例中，嵌套深度为 3：程序循环 OB 加 3 层对代码块的调用。



注：最大嵌套深度为六。安全程序使用二级嵌套。因此，用户程序在安全程序中的嵌套深度为四。

7.3.1 组织块 (OB)

组织块为程序提供结构。 它们充当操作系统和用户程序之间的接口。 OB 是由事件驱动的。 事件（如诊断中断或时间间隔）会使 CPU 执行 OB。某些 OB 预定义了起始事件和行为。

程序循环 OB 包含用户主程序。 用户程序中可包含多个程序循环 OB。 RUN 模式期间，程序循环 OB 以最低优先级等级执行，可被其它事件类型中断。 启动 OB 不会中断程序循环 OB，因为 CPU 在进入 RUN 模式之前将先执行启动 OB。

完成程序循环 OB 的处理后，CPU 会立即重新执行程序循环 OB。 该循环处理是用于可编程逻辑控制器的“正常”处理类型。 对于许多应用来说，整个用户程序位于一个程序循环 OB 中。

可创建其它 OB 以执行特定的功能，如用于处理中断和错误或用于以特定的时间间隔执行特定程序代码。 这些 OB 会中断程序循环 OB 的执行。

使用 “添加新块”(Add new block) 对话框在用户程序中创建新的 OB。

总是由事件驱动中断处理。 发生此类事件时，CPU 会中断用户程序的执行并调用已组态用于处理该事件的 OB。 完成中断 OB 的执行后，CPU 会在中断点继续执行用户程序。

CPU 按优先级确定处理中断事件的顺序。 可为多个中断事件分配相同的优先级。 更多相关信息，请参见组织块 (页 88)和执行用户程序 (页 81)。

创建附加 OB

可为用户程序，甚至为程序循环和启动 OB 事件创建多个 OB。 使用 “添加新块”(Add new block) 对话框创建 OB 并为 OB 输入名称。

如果为用户程序创建了多个程序循环 OB，则 CPU 会按数字顺序从具有最小编号（例如 OB 1）的程序循环 OB 开始执行每个程序循环 OB。 例如： 当第一个程序循环 OB（例如 OB 1）完成后，CPU 将执行下一个编号更高的程序循环 OB。

组态 OB 的属性

可对 OB 的属性进行修改。例如，可组态 OB 编号或编程语言。

说明

请注意，您可将局部过程映像编号分配给对应于 PIP0、PIP1、PIP2、PIP3 或 PIP4 的 OB。如果您为局部过程映像编号输入编号，则 CPU 将创建该过程映像分区。 有关过程映像分区的说明，请参见主题“执行用户程序 (页 81)”。

7.3.2 功能 (FC)

功能 (FC) 是通常用于对一组输入值执行特定运算的代码块。 FC 将此运算结果存储在存储器位置。 例如，可使用 FC 执行标准运算和可重复使用的运算（例如数学计算）或者执行工艺功能（如使用位逻辑运算执行独立的控制）。 FC 也可以在程序中的不同位置多次调用。此重复使用简化了对经常重复发生的任务的编程。

FC 不具有相关的背景数据块 (DB)。 对于用于计算该运算的临时数据，FC 采用了局部数据堆栈。 不保存临时数据。 要长期存储数据，可将输出值赋给全局存储器位置，如 M 存储器或全局 DB。

7.3.3 功能块 (FB)

功能块 (FB) 是使用背景数据块保存其参数和静态数据的代码块。 FB 具有位于数据块 (DB) 或“背景”DB 中的变量存储器。 背景 DB 提供与 FB 的实例（或调用）关联的一块存储区并在 FB 完成后存储数据。 可将不同的背景 DB 与 FB 的不同调用进行关联。 通过背景 DB 可使用一个通用 FB 控制多个设备。 通过使一个代码块对 FB 和背景 DB 进行调用，来构建程序。然后，CPU 执行该 FB 中的程序代码，并将块参数和静态局部数据存储在背景 DB 中。 FB 执行完成后，CPU 会返回到调用该 FB 的代码块中。 背景 DB 保留该 FB 实例的值。 随后在同一扫描周期或其它扫描周期中调用该功能块时可使用这些值。

可重复使用的代码块和关联的存储区

用户通常使用 FB 控制在一个扫描周期内未完成其运行的任务或设备的运行。 要存储运行参数以便从一个扫描快速访问到下一个扫描，用户程序中的每一个 FB 都具有一个或多个背景 DB。 调用 FB 时，也需要指定包含块参数以及用于该调用或 FB “实例” 的静态局部数据的背景 DB。 FB 完成执行后，背景 DB 将保留这些值。

通过设计用于通用控制任务的 FB，可对多个设备重复使用 FB，方法是：为 FB 的不同调用选择不同的背景 DB。

FB 将 Input、Output 和 InOut 以及静态参数存储在背景数据块中。

您还可以在 RUN 模式下修改和下载函数块接口 (页 1411)。

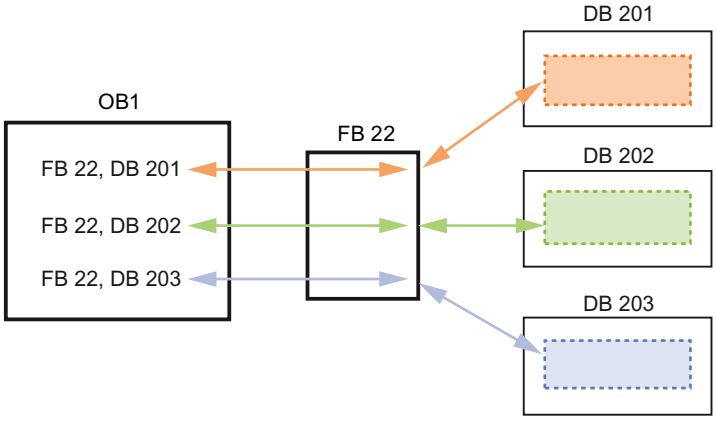
在背景数据块中分配起始值

背景数据块存储每个参数的默认值和起始值。 起始值提供在执行 FB 时使用的值。 然后可在用户程序执行期间修改起始值。

FB 接口还提供一个 “默认值”(Default value) 列，使您能够在编写程序代码时为参数分配新的起始值。 然后将 FB 中的这个默认值传给关联背景数据块中的起始值。 如果不在 FB 接口中为参数分配新的起始值，则将背景数据块中的默认值复制到起始值。

使用带多个 DB 的单个 FB

下图显示了三次调用同一个 FB 的 OB，方法是针对每次调用使用一个不同的数据块。 该结构使一个通用 FB 可以控制多个相似的设备（如电机），方法是在每次调用时为各设备分配不同的背景数据块。 每个背景 DB 存储单个设备的数据（如速度、加速时间和总运行时间）。



在此实例中，FB 22 控制三个独立的设备，其中 DB 201 用于存储第一个设备的运行数据，DB 202 用于存储第二个设备的运行数据，DB 203 用于存储第三个设备的运行数据。

7.3.4 数据块 (DB)

在用户程序中创建数据块 (DB) 以存储代码块的数据。用户程序中的所有程序块都可访问全局 DB 中的数据，而背景 DB 仅存储特定功能块 (FB) 的数据。

相关代码块执行完成后，DB 中存储的数据不会被删除。有两种类型的 DB：

- 全局 DB 存储程序中代码块的数据。任何 OB、FB 或 FC 都可访问全局 DB 中的数据。
- 背景 DB 存储特定 FB 的数据。背景 DB 中数据的结构反映了 FB 的参数（Input、Output 和 InOut）和静态数据。（FB 的临时存储器不存储在背景 DB 中。）

说明

尽管背景 DB 反映特定 FB 的数据，然而任何代码块都可访问背景 DB 中的数据。

您还可以在 RUN 模式下修改和下载数据块 (页 1411)。

只读数据块

可将 DB 组态为只读：

1. 在项目浏览器中右键单击相应 DB，然后在右键快捷菜单中选择“属性”(Properties)。
2. 在“属性”(Properties) 对话框中选择“特性”(Attributes)。
3. 选择“在设备中写保护数据块”(Data block write-protected in the device) 选项并单击“确定”(OK)。

已优化的数据块和标准数据块

您还可以将数据块组态为标准或已优化。标准 DB 与 STEP 7 Classic 编程工具以及经典的 S7-300 和 S7-400 CPU 兼容。可优化访问的数据块无固定的定义结构。数据元素在声明中仅包含一个符号名，在块中没有固定地址。CPU 会将元素自动存储到块的可用存储区中，以免在存储器中留下间隙。这样一来，便可最优化地利用存储器容量。

要设置对数据块的优化访问，请按以下步骤操作：

1. 在 STEP 7 项目树中展开程序块文件夹。
2. 右键单击数据块并从上下文菜单中选择“属性”(Properties)。
3. 为属性选择“优化块访问”(Optimized block access)。

请注意，默认情况下会为新数据块选中优化块访问。如果取消选择“优化块访问”(Optimized block access)，则块将采用标准访问。

说明

函数块及其背景数据块的块访问类型

请确保以下情况：如果函数块的设置是“Optimized block access”(优化的块访问)，则该函数块的背景数据块的设置也应该是“Optimized block access”(优化的块访问)。同样，如果没有为该函数块选择“Optimized block access”(优化的块访问)，从而该函数属于标准访问类型，则应确保背景数据块也为标准类型，而不是优化的块访问类型。

如果没有兼容的块访问类型，那么在函数块执行期间从人机界面对该函数块的 IN/OUT 参数值所做的更改可能会丢失。

7.3.5 创建可重复使用的代码块

使用项目浏览器中“程序块”(Program blocks) 下的“添加新块”(Add new block) 对话框创建 OB、FB、FC 和全局 DB。

创建代码块时，需要为块选择编程语言。 无需为 DB 选择语言，因为它仅用于存储数据。

选中“添加新对象并打开”(Add new and open) 复选框（默认），在项目视图中打开代码块。

可存储想要在库中重复使用的对象。 每个项目都有一个与之相连的项目库。 除项目库外，您还可以创建可在多个项目中使用的任意数量的全局库。 由于库彼此兼容，因此可以复制库要素并将其从一个库移动到另一个库。

库可用于创建块的模板：首先将块粘贴到项目库中，随后在其中进一步开发块。 最后，将块从项目库复制到全局库。 可将全局库共享给正在使用项目的其他同事。 他们可使用块并根据需要进一步调整块以满足各自的需求。

有关库操作的详细信息，请参见 STEP 7 在线帮助库主题。

7.3.6 向块传递参数

函数块 (FB) 和函数 (FC) 有三种不同接口类型：

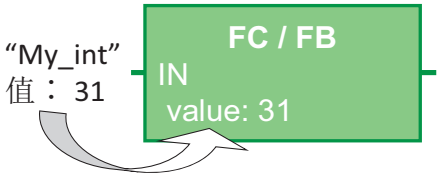
- IN
- IN/OUT
- OUT

函数块和函数通过 IN 和 IN/OUT 接口类型接收参数。 块对这些数据进行处理，此后，通过 IN/OUT 和 OUT 接口类型将返回值传回调用者。

用户程序采用以下两种方法中的某一种传递参数。

传值

用户程序以 “传值”(call-by-value) 方式将参数传递给某个函数时，用户程序会将实际参数值复制给块的 IN 接口类型的输入参数。 该操作期间，被复制值要求使用额外存储空间。



当用户程序调用该块时，会复制这些值。

传引用

用户程序以 “传引用”(call-by-reference) 方式向某个函数传递参数时，用户程序将引用 IN/OUT 接口类型的实参地址，不进行值复制操作。 该操作过程不需要额外的存储空间。

当用户程序调用该块时，会引用实际参数的地址。

说明

通常情况下，针对结构变量使用 IN/OUT 接口类型（例如，ARRAY、STRUCT 和 STRING），避免不必要地增大所需的数据存储器。

块优化和参数传递

对于简单数据类型（例如，INT、DINT 和 REAL 型），用户程序可以以“传值”方式传递函数块的参数。 传递复杂数据类型（例如，STRUCT、ARRAY 和 STRING）时，可以采用“传引用”方式。

用户程序传递的函数块参数通常在和该函数块相关的背景数据块 (DB) 中：

- 通过将参数复制给背景数据块，或者，复制位于背景数据中参数，用户程序可以以“传值”方式传递简单数据类型（例如，INT、DINT 和 REAL）的参数。
- 用户程序将复杂数据类型（例如，STRUCT、ARRAY 和 STRING）复制到用于 IN 和 OUT 参数类型的背景数据块中，或者，复制位于该背景数据块中的复杂数据类型。
- 对于 IN/OUT 接口类型，用户程序以“传引用”方式传递复杂数据类型。

数据块可以创建成“优化的”或“标准的”（未优化）数据块。 优化型数据块的体积小于非优化型数据块。 优化型数据块和非优化型数据块中的数据元素顺序不一样。 关于优化型数据块的更多说明，请参阅 S7-1200/1500、STEP 7 (TIA Portal) S7 编程指南（出版日期：2014 年 3 月） (https://support.industry.siemens.com/cs/cn/zh/view/81318674/en)中的章节“优化块”。

可以创建用来处理优化或非优化数据的函数块和函数。 可以选择复选框“优化块访问” (Optimized block access)，将其作为块的属性。 默认情况下，用户程序会优化程序块；程序块期望传递给该块的数据采用优化格式。

用户向某个函数传递复杂参数（例如，STRUCT 结构的参数）时，系统会检查包含该结构的数据块的优化设置和程序块的优化设置。 如果你同时优化该数据块和该函数，用户程序将以“传引用”方式传递该结构 (STRUCT)。 如果选择了不优化该数据块和该函数，也采用“传引用”方式传递该结构。

但是，如果函数和数据块采用不同优化设置（即，优化了一个块且没有优化另一个块），则必须将 STRUCT 转换成函数所期望的格式。 例如，如果选择了不优化该数据块但优化该函数，则数据块中的 STRUCT 须转换成优化格式后才能被该函数进行处理。 该转换过程由系统完成，其方法是：先制作该 STRUCT 的一个“副本”，接着，将它转换成该函数所期望的优化格式。

总而言之，当用户程序将某个复杂数据类型（例如，STRUCT）作为 IN/OUT 参数传递给某个函数时，该函数希望用户程序以“传引用”方式传递 STRUCT。

- 对于含该结构的数据块和该函数，如果都选择了优化或者不优化，用户程序将以“传引用”方式传递数据。
- 如果对数据块和函数没有配置相同的优化设置（优化其中一个且不优化另一个），系统必须先制作 STRUCT 的一个副本，再将其传递给函数。 由于系统必须制作该结构的副本，因此，该操作可以高效地将“传引用”转换成“传值”。

优化设置对用户程序的影响作用

如果 HMI 或中断组织块更改了结构中的元素，参数复制将可能导致用户程序出现问题。 例如，某个函数有一个 IN/OUT 参数（正常情况下以“传引用”方式传递），但是，数据块和该函数采用了不同的设置，则：

1. 用户程序准备调用该函数时，系统必须制作该结构的一个“副本”，以将该数据的格式转换成与该函数相匹配的格式。
2. 用户程序采用该结构的该“副本”的引用调用该函数。
3. 该函数运行期间出现了一个中断组织块，且该中断组织块更改了原结构中的某个值。
4. 该函数运行完毕。由于该结构是一个 IN/OUT 参数，因此，系统将该值以原来的格式复制回原结构。

采用制作结构副本的方式进行格式转换的后果：该中断组织块改写过的数据将被丢失。 对于采用 HMI 写入的值，情况也同样如此。 HMI 也可能中断用户程序的执行，并以和中断组织块相同的方式写入某个值。

解决该问题的方法有很多种：

- 最好的方法是：需要使用复杂数据类型（例如，STRUCT）时，对程序块和数据块采用相匹配的优化设置。 这种方法可以保证用户程序总是以“传引用”方式进行参数传递。
- 另一种方法是使中断组织块或 HMI 不直接修改该结构中的元素。 让组织块或 HMI 修改另一个变量，此后，用户在用户程序的某个特定位置将该变量复制到该结构中。

7.4 了解数据一致性

CPU 为所有基本数据类型（例如 Word 或 DWord）和所有系统定义的结构（例如 IEC_TIMERS 或 DTL）保持数据一致性。 值的读/写操作无法中断。 （例如，在读写四字节的 DWord 之前，CPU 会防止对该 DWord 值进行访问。）为确保程序循环 OB 和中断 OB 无法同时写入同一个存储单元，在程序循环 OB 中的读/写操作完成之前，CPU 不会执行中断 OB。

如果用户程序共享存储器中在程序循环 OB 和中断 OB 之间生成的多个值，用户程序还必须确保在修改或读取这些值时保持一致性。 可以在程序循环 OB 中使用 DIS_AIRT（禁用报警中断）和 EN_AIRT（启用报警中断）指令，以防止对共享值进行访问。

- 在代码块中插入 DIS_AIRT 指令，以确保在读/写操作期间无法执行中断 OB。
- 插入读/写能够被中断 OB 更改的值的指令。
- 在顺序结尾插入 EN_AIRT 指令，以取消 DIS_AIRT，并允许执行中断 OB。

HMI 设备或另一个 CPU 发出的通信请求也能够中断程序循环 OB 的执行。通信请求也会导致与数据一致性相关的问题。 CPU 确保基本数据类型始终由用户程序指令执行一致地读取和写入。 由于通信会周期性地中断用户程序，因而不能保证 HMI 能够同时更新 CPU 中的多个值。 例如，给定 HMI 画面上显示的值可能来自 CPU 的不同扫描周期。

PtP（Point-to-Point，点到点）指令、PROFINET 指令（例如，TSEND_C 和 TRCV_C）和 PROFINETS 分布式 I/O 指令 (页 385)和 PROFIBUS 分布式 I/O 指令 (页 385)可用于传送被中断的数据缓冲区。 通过避免对程序循环 OB 和中断 OB 中的缓冲区进行任何读/写操作，可以确保数据缓冲区的数据一致性。如果需要在中断 OB 中修改这些指令的缓冲区值，请使用 DIS_AIRT 指令延迟所有中断（中断 OB 或源自 HMI 或另一个 CPU 的通信中断），直到执行了 EN_AIRT 指令。

说明

使用 DIS_AIRT 指令延迟中断 OB 的处理，直到执行了 EN_AIRT 指令，以此影响用户程序的中断等待时间（从事件发生到执行中断 OB 的时间）。

7.5 编程语言

STEP 7 为 S7-1200 提供以下标准编程语言：

- LAD（梯形图逻辑）是一种图形编程语言。 它使用基于电路图 (页 188)的表示法。
- FBD（功能块图）是基于布尔代数 (页 189)中使用的图形逻辑符号的编程语言。
- SCL（结构化控制语言）是一种基于文本的高级编程语言 (页 189)。

创建代码块时，应选择该块要使用的编程语言。

用户程序可以使用由任意或所有编程语言创建的代码块。

7.5.1 梯形图 (LAD)

电路图的元件（如常闭触点、常开触点和线圈）相互连接构成程序段。

要创建复杂运算逻辑，可插入分支以创建并行电路的逻辑。 并行分支向下打开或直接连接到电源线。 用户可向上终止分支。

LAD 向多种功能（如数学、定时器、计数器和移动）提供“功能框”指令。

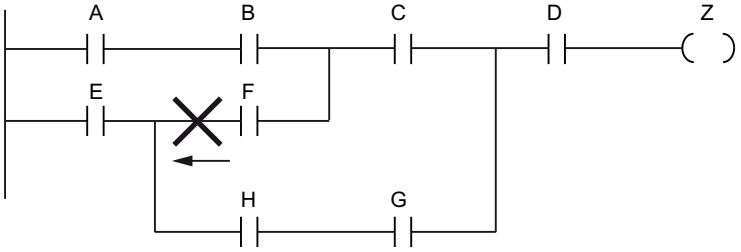
STEP 7 不限制 LAD 程序段中的指令（行和列）数。

说明

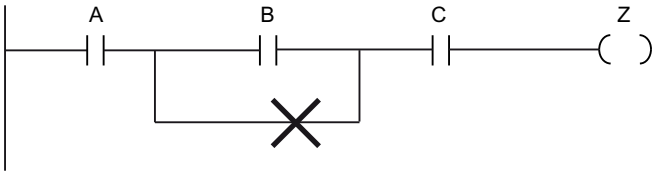
每个 LAD 程序段都必须使用线圈或功能框指令来终止。

创建 LAD 程序段时请注意以下规则：

- 不能创建可能导致反向能流的分支。

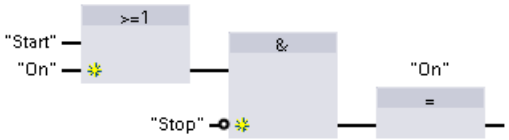


- 不能创建可能导致短路的分支。



7.5.2 功能块图 (FBD)

与 LAD 一样，FBD 也是一种图形编程语言。 逻辑表示法以布尔代数中使用的图形逻辑符号为基础。



要创建复杂运算的逻辑，在功能框之间插入并行分支。

算术功能和其它复杂功能可直接结合逻辑框表示。

STEP 7 不限制 FBD 程序段中的指令（行和列）数。

7.5.3 SCL

结构化控制语言 (SCL, Structured Control Language) 是用于 SIMATIC S7 CPU 的基于 PASCAL 的高级编程语言。 SCL 支持 STEP 7 的块结构 (页 177)。 可以使用以下三种编程语言之一将程序块包括到项目中：SCL、LAD 和 FBD。

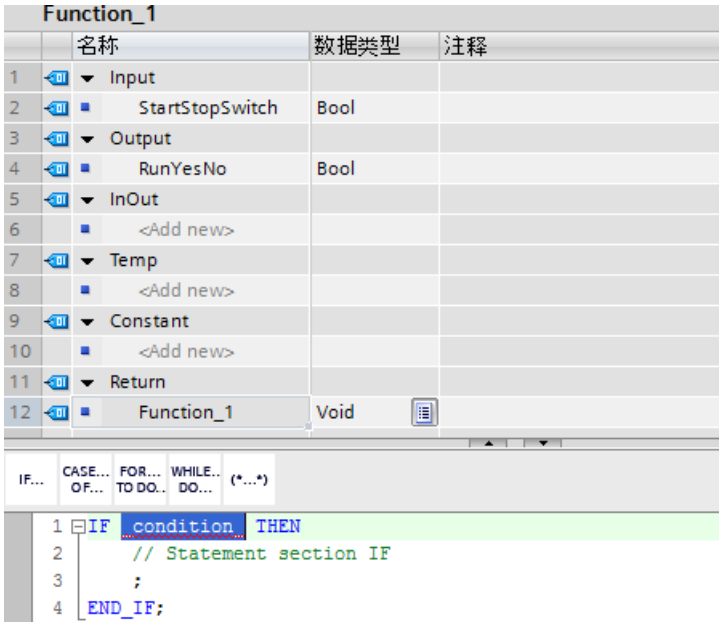
SCL 指令使用标准编程运算符，例如，用 (:=) 表示赋值，算术功能（+ 表示相加，- 表示相减，* 表示相乘，/ 表示相除）。 SCL 也使用标准的 PASCAL 程序控制操作，如 IF-THEN-ELSE、CASE、REPEAT-UNTIL、GOTO 和 RETURN。 SCL 编程语言中的语法元素还可以使用所有的 PASCAL 参考。 许多 SCL 的其它指令（如定时器和计数器）与 LAD 和 FBD 指令匹配。 有关特定指令的更多信息，请参见基本指令 (页 213)和扩展指令 (页 337)章节中的特定指令。

7.5.3.1 SCL 程序编辑器

可以在创建该块时指定任何块类型（OB、FB 或 FC）以便使用 SCL 编程语言。 STEP 7 提供包含以下元素的 SCL 程序编辑器：

- 用于定义代码块参数的接口部分
- 用于程序代码的代码部分
- 包含 CPU 支持的 SCL 指令的指令树

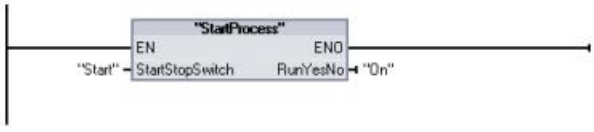
可以直接在代码部分输入指令的 SCL 代码。 编辑器包含用于通用代码结构和注释的按钮。 要了解更复杂的指令，只需从指令树拖动 SCL 指令并将其放入程序中。 也可以使用任意文本编辑器创建 SCL 程序，然后将相应文件导入 STEP 7 中。



在 SCL 代码块接口部分，可以声明下列类型的参数：

- Input、Output、InOut 和 Ret_Val： 这些参数定义代码块的输入变量、输出变量和返回值。 执行代码块期间局部使用此处输入的变量名称。 通常不会使用变量表中的全局变量名称。
- Static（仅适用于 FB，上述示例适用于 FC）： 代码块使用静态变量在背景数据块中存储静态中间结果。 块会一直保留静态数据，直到多个周期后被覆盖。 块的名称（此块将其作为多重背景调用）也存储在静态局部数据中。
- Temp：这些参数是执行代码块期间使用的临时变量。
- Constant：这些是为代码块指定的常数值。

如果从其它代码块调用 SCL 代码块，该 SCL 代码块的参数会显示为输入或输出。



本示例中，“Start”和“On”变量（来自项目变量表）相当于 SCL 程序声明表中的“StartStopSwitch”和“RunYesNo”。

7.5.3.2 SCL 表达式和运算

构造 SCL 表达式

SCL 表达式是用于计算值的公式。表达式由操作数和运算符（如 *、/、+ 或 -）组成。操作数可以是变量、常量或表达式。

表达式的计算按一定的顺序进行，具体由以下因素决定：

- 每个运算符均具有预定义的优先级，首先执行优先级最高的运算。
- 按从左至右的顺序处理优先级相同的运算符。
- 可使用圆括号指定要一起计算的一系列运算符。

表达式的结果可用于将值分配给程序使用的变量、用作由控制语句使用的条件、用作其它 SCL 指令的参数或者用于调用代码块。

表格 7-2 SCL 中的运算符

类型	操作	操作员	优先级
圆括号	（*表达式*）	(,)	1
数学	乘方	**	2
	符号（一元加号）	+	3
	符号（一元减号）	-	3
	倍增	*	4
	除法	/	4
	取模	MOD	4
	加法	+	5
	减法	-	5
比较	小于	<	6
	小于或等于	<=	6
	大于	>	6
	大于或等于	>=	6
	等于	=	7
	不等于	<>	7

类型	操作	操作员	优先级
位逻辑	取反（一元）	NOT	3
	AND 逻辑运算	AND 或 &	8
	异或逻辑运算	XOR	9
	OR 逻辑运算	OR	10
赋值	赋值	:=	11

作为一种高级编程语言，SCL 使用标准语句实现基本任务：

- 赋值语句：**:=**
- 算术功能：**+**、**-**、***** 和 **/**
- 全局变量的寻址：**"<**变量名称**>"**（变量名称或数据块名称括在双引号内）
- 局部变量的寻址：**#<**变量名称**>**（在变量名称前加“**#**”符号）

以下示例显示了用法不同的各种表达式：

`"C" := #A+#B;`	将两个局部变量之和赋值给一个变量
`"Data_block_1".Tag := #A;`	为数据块变量赋值
`IF #A > #B THEN "C" := #A;`	IF-THEN 语句的条件
`"C" := SQRT (SQR (#A) + SQR (#B));`	SQRT 指令的参数

算术运算符可以处理各种数值数据类型。结果的数据类型取决于最高有效操作数的数据类型。例如，使用 INT 操作数和 REAL 操作数的乘法运算会产生 REAL 结果值。

控制语句

控制语句是 SCL 表达式的一种专用类型，可用于执行以下任务：

- 程序分支
- 重复 SCL 编程代码的某些部分
- 跳转到 SCL 程序的其它部分
- 按条件执行

SCL 控制语句包括 IF-THEN、CASE-OF、FOR-TO-DO、WHILE-DO、REPEAT-UNTIL、CONTINUE、GOTO 和 RETURN。

一条语句通常占一行代码。可以在一行中输入多条语句，或者可将一条语句断开成多行代码以使代码易于阅读。分隔符（如制表符、换行符和多余空格）在语法检查期间会被忽略。END 语句可终止控制语句。

以下示例显示的是 FOR-TO-DO 控制语句。（两种形式的代码在语法上均有效。）

```
FOR x := 0 TO max DO sum := sum + value(x); END_FOR;
FOR x := 0 TO max DO
     sum := sum + value(x);
END_FOR;
```

还可以为控制语句提供标签。用语句前的逗点将标签隔开：

```
Label: <Statement>;
```

有关完整的 SCL 编程语言参考，请参见 STEP 7 在线帮助。

条件

条件是一个比较表达式或逻辑表达式，其结果为 BOOL 类型（值为 TRUE 或 FALSE）。以下示例显示了各种类型的条件：

`#Temperature > 50`
`#Counter <= 100`
`#CHAR1 < 'S'`
	关系表达式

`(#Alpha <> 12) AND NOT #Beta`	比较和逻辑表达式
`5 + #Alpha`	算术表达式

条件可以使用算术表达式：

- 如果结果是非零的任何值，则表达式的条件为 TRUE。
- 如果结果为零，则表达式的条件为 FALSE。

从 SCL 程序中调用其它代码块

要调用用户程序中的其它代码块，只需使用参数输入 FB 或 FC 的名称（或绝对地址）。对于 FB，还必须提供 FB 待调用的背景数据块。

`<DB 名称> (参数列表)`	作为单个背景调用
`<#背景名称> (参数列表)`	作为多重背景调用
`"MyDB"(MyInput:=10, MyInOut:="Tag1");`	
`<FC 名称> (参数列表)`	标准调用
`<操作数>:=<FC 名称> (参数列表)`	在表达式中调用
`"MyFC"(MyInput:=10, MyInOut:="Tag1");`	

还可将块从导航树中拖动到 SCL 程序编辑器中，然后完成参数分配。

将块注释添加到 SCL 代码

可通过在 (* 和 *) 之间加入注释文本，将块注释添加到 SCL 代码中。可在 (* 和 *) 之间添加任意数目的注释行。SCL 程序块可能包括多个块注释。为方便编程，SCL 编辑器包括一个块注释按钮和通用控制语句：

寻址

与 LAD 和 FBD 一样，SCL 允许用户在用户程序中使用变量地址（符号寻址）或绝对地址。SCL 还允许使用变量作为数组索引。

绝对寻址

%I0.0 **%MB100**	在绝对地址之前加上“%”符号。没有“%”，STEP 7 将在编译时生成未定义的变量错误。

符号寻址

"PLC_Tag_1"	PLC 变量表中的变量
"Data_block_1".Tag_1	数据块中的变量
"Data_block_1".MyArray[#i]	数据块数组中的数组元素

7.5.3.3 使用 PEEK 和 POKE 指令进行索引寻址

SCL 提供 PEEK 和 POKE 指令，可用来从数据块、I/O 或存储器中读取内容或是向其中写入内容。而您提供操作中具体字节偏移量或位偏移量的参数。

说明

与数据块一起使用 PEEK 和 POKE 指令时，必须使用标准（未优化的）数据块。同时需要注意 PEEK 和 POKE 指令仅用于传输数据。它们无法识别地址中的数据类型。

PEEK(area:=_in_, dbNumber:=_in_, byteOffset:=_in_);	读取引用数据块、I/O 或存储区中由 byteOffset 引用的字节。 引用数据块示例： **%MB100 := PEEK(area:=16#84, dbNumber:=1, byteOffset:=#i);** 引用 IB3 输入示例： **%MB100 := PEEK(area:=16#81, dbNumber:=0, byteOffset:=#i); // when #i = 3**
PEEK_WORD(area:=_in_, dbNumber:=_in_, byteOffset:=_in_);	读取引用数据块、I/O 或存储区中由 byteOffset 引用的字。 示例： **%MW200 := PEEK_WORD(area:=16#84, dbNumber:=1, byteOffset:=#i);**
PEEK_DWORD(area:=_in_, dbNumber:=_in_, byteOffset:=_in_);	读取引用数据块、I/O 或存储区中由 byteOffset 引用的双字。 示例： **%MD300 := PEEK_DWORD(area:=16#84, dbNumber:=1, byteOffset:=#i);**
PEEK_BOOL(area:=_in_, dbNumber:=_in_, byteOffset:=_in_, bitOffset:=_in_);	读取引用数据块、I/O 或存储区中由 bitOffset 和 byteOffset 引用的布尔值。 示例： **%MB100.0 := PEEK_BOOL(area:=16#84, dbNumber:=1, byteOffset:=#ii, bitOffset:=#j);**
POKE(area:=_in_, dbNumber:=_in_, byteOffset:=_in_, value:=_in_);	向引用数据块、I/O 或存储区中引用的 byteOffset 写入值（Byte、Word 或 DWord） 引用数据块示例： **POKE(area:=16#84, dbNumber:=2, byteOffset:=3, value:="Tag_1");** 引用 QB3 输出示例： **POKE(area:=16#82, dbNumber:=0, byteOffset:=3, value:="Tag_1");**

```
POKE_BOOL(area:=_in_,
          dbNumber:=_in_,
          byteOffset:=_in_,
          bitOffset:=_in_,
          value:=_in_);
```

向引用数据块、I/O 或存储区中引用的 bitOffset 和 byteOffset 写入布尔值

示例：

```
POKE_BOOL(area:=16#84, dbNumber:=2,
byteOffset:=3, bitOffset:=5,
value:=0);
```

```
POKE_BLK(area_src:=_in_,
          dbNumber_src:=_in_,
          byteOffset_src:=_in_,
          area_dest:=_in_,
          dbNumber_dest:=_in_,
          byteOffset_dest:=_in_,
          count:=_in_);
```

将引用源数据块、I/O 或存储区从引用字节偏移量开始的共“count”个字节写入引用目标数据块、I/O 或存储区中引用的 byteOffset 区域

示例：

```
POKE_BLK(area_src:=16#84,
dbNumber_src:=#src_db,
byteOffset_src:=#src_byte,
area_dest:=16#84,
dbNumber_dest:=#src_db,
byteOffset_dest:=#src_byte,
count:=10);
```

对于 PEEK 和 POKE 指令，“area”、“area_src”和“area_dest”参数可以使用以下值。对于数据块以外的其它区域，dbNumber 参数必须为 0。

16#81	I
16#82	Q
16#83	M
16#84	DB

7.5.4 LAD、FBD 和 SCL 的 EN 和 ENO

确定指令的“能流”（EN 和 ENO）

特定指令（如数学和移动指令）为 EN 和 ENO 提供参数。 这些参数与 LAD 或 FBD 中的能流有关并确定在该扫描期间是否执行指令。 SCL 还允许用户为代码块设置 ENO 参数。

- EN（使能输入）是布尔输入。 要执行功能框指令，能流 (EN = 1) 必须出现在此输入端。 如果 LAD 框的 EN 输入直接连接到左侧电源线，将始终执行该指令。
- ENO（使能输出）是布尔输出。 如果该功能框在 EN 输入端有能流且正确执行了其功能，则 ENO 输出会将能流 (ENO = 1) 传递到下一个元素。 如果执行功能框指令时检测到错误，则在产生该错误的功能框指令处终止该能流 (ENO = 0)。

表格 7-3　EN 和 ENO 的操作数

程序编辑器	输入/输出	操作数	数据类型
LAD	EN, ENO	能流	Bool
FBD	EN	I、I:P、Q、M、DB、Temp、能流	Bool
	ENO	能流	Bool
SCL	EN[1]	TRUE, FALSE	Bool
	ENO[2]	TRUE, FALSE	Bool

[1] EN 仅适用于 FB。

[2] 可以选择将 ENO 与 SCL 代码块一起使用。 代码块完成时，用户必须组态 SCL 编译器来设置 ENO。

通过组态 SCL 来设置 ENO

要组态 SCL 编译器以设置 ENO，请按以下步骤操作：

1. 从 “选项”(Options) 菜单中选择 “设置”(Settings) 命令。
2. 展开“PLC 编程”(PLC programming) 属性并选择“SCL（结构化控制语言）”(SCL (Structured Control Language))。
3. 选择 “自动设置 ENO”(Set ENO automatically) 选项。

在程序代码中使用 ENO

您还能够通过将 ENO 分配给 PLC 变量或在局部块中评估 ENO 等方式在程序代码中使用 ENO。

示例：

```
"MyFunction"
  ( IN1 := … ,
    IN2 := … ,
    OUT1 => #myOut,
    ENO => #statusFlag ); // PLC tag statusFlag holds the value o
f ENO

"MyFunction"
  ( IN1 := …
    IN2 := … ,
    OUT1 => #myOut,
    ENO => ENO );  // block status flag of "MyFunction"
                   // is stored in the local block

IF ENO = TRUE THEN
    // execute code only if MyFunction returns true ENO
```

Ret_Val 或 Status 参数对 ENO 的影响

某些指令（如通信指令或字符串转换指令）提供一个输出参数，其中包含有关指令处理的信息。 例如，某些指令提供通常为 Int 数据类型的 Ret_Val（返回值）参数，其中包含 -32768 到 +32767 范围内的状态信息。还有些指令提供通常为 Word 数据类型的 Status 参数，其中存储十六进制值 16#0000 到 16#FFFF 范围内的状态信息。 Ret_Val 或 Status 参数中存储的数字值确定该指令的 ENO 状态。

- Ret_Val：介于 0 至 32767 的值通常设置 ENO = 1（即 TRUE）。 介于 -32768 至 -1 的值通常设置 ENO = 0（即 FALSE）。 要评估 Ret_Val，将表示法更改为十六进制。
- Status：介于 16#0000 至 16#7FFF 的值通常设置 ENO = 1（即 TRUE）。 介于 16#8000 至 16#FFFF 的值通常设置 ENO = 0（即 FALSE）。

需要多次扫描才能执行的指令通常提供 Busy 参数 (Bool)，用于表示指令处于活动状态，但尚未完成执行。 此类指令通常还提供 Done 参数 (Bool) 和 Error 参数 (Bool)。Done 表示指令已完成且无错误，而 Error 表示指令已完成，但存在错误情况。

- Busy = 1（即 TRUE）时，ENO = 1（即 TRUE）。
- Done = 1（即 TRUE）时，ENO = 1（即 TRUE）。
- Error = 1（即 TRUE）时，ENO = 0（即 FALSE）。

参见

OK（检查有效性）和 NOT_OK（检查无效性） (页 237)

7.6 保护

7.6.1 CPU 的访问保护

CPU 提供了四个安全等级，用于限制对特定功能的访问。为 CPU 组态安全等级和密码时，可以对那些不输入密码就能访问的功能和存储区进行限制。

每个等级都允许在访问某些功能时不使用密码。CPU 的默认状态是没有任何限制，也没有密码保护。要限制 CPU 的访问，可以对 CPU 的属性进行组态并输入密码。

通过网络输入密码并不会使 CPU 的密码保护受到威胁。密码保护不适用于用户程序指令的执行，包括通信功能。输入正确的密码便可访问该级别的所有功能。

PLC 到 PLC 通信（使用代码块中的通信指令）不受 CPU 中安全等级的限制。

表格 7-4　　CPU 的安全级别

安全等级	访问限制
完全访问（无保护）	允许完全访问，没有密码保护。
读访问	允许 HMI 访问、比较离线/在线代码块和各种形式的 PLC 到 PLC 通信（无密码保护）。 修改（写入）CPU 需要密码。更改 CPU 模式 (RUN/STOP) 不需要密码。
HMI 访问	允许 HMI 访问和各种形式的 PLC 到 PLC 通信，没有密码保护。 读取 CPU 中的数据、比较离线/在线代码块、修改（写入）CPU 以及切换 CPU 模式 (RUN/STOP) 时需要密码。
无访问（完全保护）	不允许没有密码保护的访问。 进行 HMI 访问、读取 CPU 中的数据、比较离线/在线代码块和修改（写入）CPU 时需要密码。

请注意，可以为 CPU 设置任何安全级别的紧急（临时）IP 地址 (页 1017)。

密码区分大小写。要组态保护级别和密码，请按以下步骤操作：

1. 在“设备组态”(Device configuration) 中，选择 CPU。
2. 在巡视窗口中，选择“属性”(Properties) 选项卡。
3. 选择“保护”(Protection) 属性以选择保护等级和输入密码。

当您将此组态下载至 CPU 时，用户将具有 HMI 访问权限，可以在无密码的情况下访问 HMI 功能。要读取数据或比较离线/在线代码块，用户必须输入“读访问”(Read access) 的已组

态密码或“完全访问（无保护）”(Full access (no protection)) 的密码。要写入数据，用户必须输入“完全访问（无保护）”的已组态密码。

⚠ 警告

对受保护的 CPU 进行未授权访问

拥有 CPU 完全访问权限的用户有权限读写 PLC 变量。无论 CPU 访问级别是多少，Web 服务器用户都有权限读写 PLC 变量。未经授权访问 CPU 或将 PLC 变量更改为无效值可能会中断过程操作并可能导致死亡、严重人身伤害和/或财产损失。

授权用户可以执行共模模式更改、写入 PLC 数据以及进行固件更新。Siemens 建议您遵守以下安全实践：

- 使用强密码对 CPU 访问级别和 Web 服务器用户 ID (页 1029) 进行密码保护。强密码在长度上至少为十个字符，可以是字母、数字和特殊字符的组合，不能是可在字典上找到的词，并且不能是可从个人信息推断出的名字或标识符。保管好密码并经常更改密码。
- 仅使用 HTTPS 协议启用对 Web 服务器的访问。
- 不要扩展 Web 服务器“所有人”(Everybody) 用户的默认最低权限。
- 对程序逻辑中的变量执行错误检查和范围检查，因为 Web 页面用户可将 PLC 变量更改为无效值。

连接机制

要使用 PUT/GET 指令访问远程连接伙伴，用户还必须得到许可。

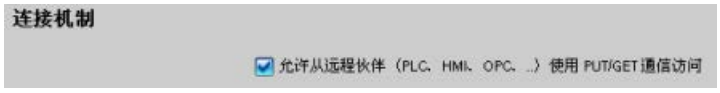
默认情况下，“允许使用 PUT/GET 通信进行访问”(Permit access with PUT/GET communication) 选项处于未启用状态。这时，只有需要对本地 CPU 和通信伙伴同时进行组态和编程的通信连接才能实现对 CPU 数据的读写访问。例如，可以通过 BSEND/BRCV 指令进行访问。

因此，本地 CPU 仅作为服务器的连接（也就是说，本地 CPU 中不存在带有通信伙伴的通信组态/编程）在 CPU 运行期间不可用，例如：

- 通过通信模块进行 PUT/GET、FETCH/WRITE 或 FTP 访问
- 从其它 S7 CPU 进行 PUT/GET 访问
- 通过 PUT/GET 通信进行 HMI 访问

如果您希望允许从客户端访问 CPU 数据，即您不希望限制 CPU 的通信服务，请按以下步骤操作：

1. 将保护访问级别组态为除“无访问（完全保护）”(No access (complete protection)) 外的任意级别。
2. 选择“允许使用 PUT/GET 通信进行访问”(Permit access with PUT/GET communication) 复选框。



当您将此组态下载至 CPU 时，CPU 将允许与远程伙伴进行 PUT/GET 通信

7.6.2 外部装载存储器

也可以防止从内部装载存储器备份到外部装载存储器（SIMATIC 存储卡）。要防止从内部装载存储器到外部装载存储器的复制操作，请按照以下步骤操作：

1. 在 STEP 7 中，从 CPU 设备组态的“常规”(General) 属性中选择“保护”(Protection)。
2. 在“外部装载存储器”(External Load Memory) 部分，选择“禁用从内部装载存储器到外部装载存储器的复制操作”(Disable copy from internal load memory to external load memory)。

有关该属性对 CPU 插入存储卡的影响，另请参见在 CPU 中插入存储卡 (页 134)主题。

7.6.3 专有技术保护

专有技术保护可防止程序中的一个或多个代码块（OB、FB、FC 或 DB）受到未经授权的访问。 用户创建密码以限制对代码块的访问。 密码保护会防止对代码块进行未授权的读取或修改。 如果没有密码，只能读取有关代码块的以下信息：

- 块标题、块注释和块属性
- 传送参数（IN、OUT、IN_OUT、Return）
- 程序的调用结构
- 交叉引用中的全局变量（不带使用时的信息），但局部变量已隐藏

将块组态为“专有技术”保护时，只有在输入密码后才能访问块内的代码。

使用代码块的“属性”(Properties) 任务卡组态该块的专有技术保护。 打开代码块后，从“属性”(Properties) 中选择“保护”(Protection)。

1. 在代码块的“属性”(Properties) 中，单击“保护”(Protection) 按钮显示“专有技术保护”(Know-how protection) 对话框。
2. 单击“定义”(Define) 按钮输入密码。

输入并确认密码后，单击“确定”(OK)。

7.6.4 复制保护

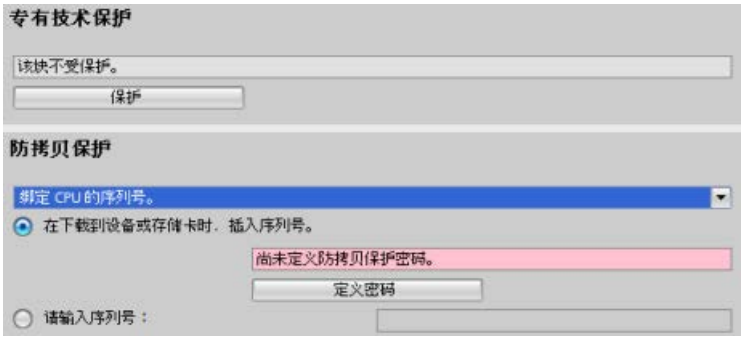
附加安全特性允许捆绑程序块，以用于特定存储卡或 CPU。该特性对于保护您的知识产权特别有用。当您将程序块与特定设备捆绑在一起时，就会将程序或代码块限制为仅用于特定存储卡或 CPU。此特性允许用户通过电子方式（例如通过 Internet 或电子邮件）或发送存储卡的方式分配程序或代码块。复制保护可用于 OB (页 178)、FB (页 180) 和 FC (页 180)。S7-1200 CPU 支持三种类型的块保护：

- 与 CPU 的序列号进行绑定
- 与存储卡的序列号进行绑定
- 与强制性密码动态绑定

使用代码块的“属性”(Properties) 任务卡将块捆绑到特定 CPU 或存储卡。

1. 打开代码块之后选择“保护”(Protection)。

2. 在“复制保护”(Copy protection) 任务下的下拉列表中，选择要使用的复制保护的类型。



3. 对于与 CPU 或存储卡序列号的绑定，可以在下载时插入序列号，也可以输入存储卡或 CPU 的序列号。

说明

序列号区分大小写。

对于与强制性密码的动态绑定，定义下载或复制块所必须使用的密码。
随后下载 (页 204)带有动态绑定的块时，必须输入可用于下载块的密码。请注意，复制保护密码和专有技术保护 (页 201)密码是两个不同的密码。

7.7 下载程序的元素

可将项目的元素从编程设备下载到 CPU。下载项目时，CPU 会存储内部装载存储器中的用户程序（OB、FC、FB 和 DB）；如果存在 SIMATIC 存储卡，CPU 也会存储该外部装载存储器（存储卡）中的用户程序。

可从以下任何位置将项目从编程设备下载到 CPU：

- 项目树：右键单击程序元素，然后单击上下文相关的“下载”(Download) 选择项。
- 在线菜单：单击“下载到设备”(Download to device) 选择项。
- 工具栏：单击“下载到设备”(Download to device) 图标。
- 设备组态：右键单击 CPU 并选择要下载的元素。

请注意，如果已将与强制性密码的动态绑定 (页 202)应用于任一程序块，则只有输入受保护块的密码才能下载该程序块。如果已为多个块组态了该类型的复制保护，则必须输入每个受保护块的密码才能下载这些块。

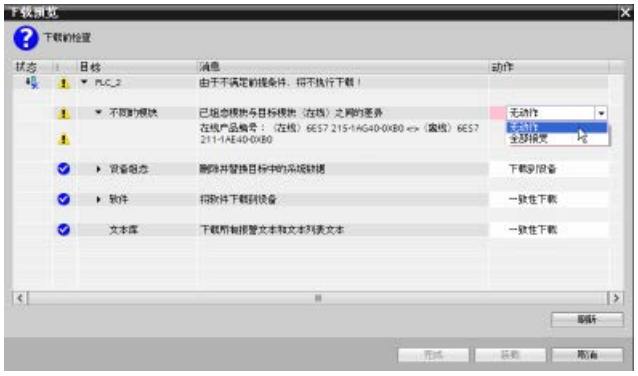
说明

下载程序不会清除或更改保持性存储器中的现有值。如果要在下载之前清除保持性存储器，请在下载程序前将 CPU 复位为出厂设定。

您还可以将 Basic HMI 面板的面板项目 (页 35)从 TIA Portal 下载到 S7-1200 CPU 的存储卡中。

组态的 CPU 与连接的 CPU 不同时的下载操作

在满足存储器要求和 I/O 兼容性的前提下，当连接的 CPU 容量足以存储从组态的 CPU 下载的内容时，STEP 7 和 S7-1200 允许下载操作。可以将某个 CPU 中的组态和程序下载到更大的 CPU 中，例如从 CPU 1211C DC/DC/DC 下载到 CPU 1215C DC/DC/DC，因为 I/O 兼容且存储器容量充足。在这种情况下，下载操作会显示一条警告：“组态的模块与目标模块（在线）不同”(Differences between configured and target modules (online))，并且“加载预览”(Load preview) 对话框中会显示相应的部件编号和固件版本。如果想要停止下载操作，必须选择“无操作”(No action)；如果想要继续下载操作，必须选择“全部接受”(Accept all)：



说明

在将组态的 CPU 下载到连接的不同 CPU 后进行在线连接 (页 1386)时，项目树中组态的 CPU 对应的项目将显示在线状态指示。但在线和诊断视图中显示实际连接的 CPU 模块类型。

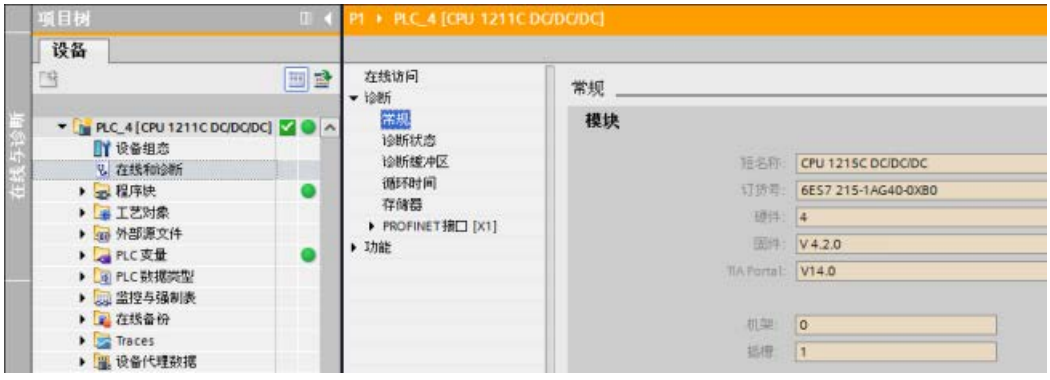


图 7-1　组态的 CPU 与连接的 CPU 不同时的在线视图

当然，可以在设备组态中更改设备 (页 162)，以便使组态的 CPU 与连接的 CPU 属于相同模块类型。当您尝试更改设备时，“更改设备”(Change device) 对话框中会提供完整的兼容性详细信息。

当连接的 CPU 容量不足以存储从组态的 CPU 下载的内容时，STEP 7 和 S7-1200 会禁止下载操作；例如，下列情况下无法下载硬件组态和程序：

- CPU 1215C DC/DC/DC 到 CPU 1212C DC/DC/DC，因为工作存储器容量不足
- CPU 1211C DC/DC/继电器 到 CPU 1211C DC/DC/DC，因为 I/O 不兼容
- CPU 1217C DC/DC/DC 到任意 CPU 1211C、CPU 1212C、CPU 1214C 或 CPU 1215C，因为 CPU 1217C 的输出为 1.5 V DC
- CPU 1214C V4.2.x 到 CPU 1214C V4.0，因为固件版本不向后兼容

在这类情况下，“加载预览”(Load preview) 对话框将显示如下错误：

下载失败后恢复

如果下载失败，巡视窗口的“信息”(Info) 选项卡中会显示原因。诊断缓冲区也会提供相关信息。下载失败后，需按照以下步骤操作才能顺利完成下载：

1. 按错误消息所述修正问题。
2. 再次尝试下载操作。

在极少情况下，下载成功但 CPU 的后续循环上电失败。在这种情况下，诊断缓冲区中会显示错误，例如：

- 16# 02:4175 -- CPU 错误：存储卡评估错误：CPU 组态的版本未知或不兼容 描述当前卡类型：无存储卡 功能已完成/中止，新启动禁止设置： ..- 存储卡丢失，类型错误，内容错误或受保护

如果出现该错误并且额外的下载尝试均失败，则必须清空内部装载存储器或外部装载存储器：

1. 如果使用的是内部装载存储器，则将 CPU 复位为出厂设置。
2. 如果使用的是 SIMATIC 存储卡，则将其拔出并删除存储卡的内容 (页 139)，然后再重新插入。
3. 下载硬件组态和软件。

参见

将在线 CPU 与离线项目同步 (页 207)

7.8 将在线 CPU 与离线项目同步

将项目块下载到 CPU 时，CPU 会检测在线 CPU 中的块或变量自上次下载后是否发生更改。在这类情况下，CPU 会为您提供同步这些更改的选项。这意味着，在将项目下载到 CPU 之前，您可以将在线 CPU 的更改上传到项目中。在线 CPU 中的更改可能由于以下各种因素导致：

- 在运行期间更改数据块变量的起始值，例如通过 WRIT_DBL 指令 (页 543)或加载配方
- 从存在下列其中一个或多个条件的“二级”项目（与上一次下载操作无关的项目）执行下载操作：
  - 在线 CPU 包含项目中没有的程序块。
  - 离线项目和在线 CPU 的数据块变量或块属性不同。
  - 在线 CPU 包含离线项目中没有的 PLC 变量。

说明

如果要编辑用于上一次下载操作的项目中的块或变量，不必进行任何有关同步的选择。STEP 7 和 CPU 会检测到离线项目更改比在线 CPU 更新，并继续执行正常的下载操作。

同步选项

向 CPU 下载项目时，如果 STEP 7 检测到在线 CPU 中的数据块或变量比项目值更新，则将显示同步对话框。例如，如果 STEP 7 程序已执行 WRIT_DBL 并更改 Data_block_1 中某个变量的起始值，则开始执行下载操作时会显示如下同步对话框：

该对话框中列有更改所在的程序块。该对话框中提供以下选项：

- “在线/离线比较”(Online/offline comparison)：单击此按钮时，STEP 7 会显示项目的程序块、系统块、工艺对象、PLC 变量和 PLC 数据类型与在线 CPU 比较 (页 1395)的结果。对于每个对象，可单击查看包含时间戳在内的差异的详细分析。可以使用该信息决定如何处理在线 CPU 与项目的差异。
- “同步”(Synchronize)：单击此按钮时，STEP 7 会将在线 CPU 的数据块、变量和其他对象上传到项目。之后可以继续下载程序，程序执行再次导致项目与 CPU 失去同步的情况除外。
- “无需同步继续 (Continue without synchronization)”：单击此按钮时，STEP 7 会将项目下载到 CPU。
- “取消”(Cancel)：单击此按钮时，将取消下载操作。

7.9 从在线 CPU 上传

还可以由在线 CPU 或连接到编程设备的存储卡复制程序块。

为复制的程序块准备离线项目：

1. 添加一个与在线 CPU 匹配的 CPU 设备。
2. 展开该 CPU 节点一次，以便“程序块”(Program blocks) 文件夹可见。

要从在线 CPU 向离线项目上传程序块，请按照以下步骤操作：

1. 在离线项目中，单击“程序块”(Program blocks) 文件夹。
2. 单击“转到在线”(Go online) 按钮。
3. 单击“上传”(Upload) 按钮。
4. 在“上传”(Upload) 对话框 (页 1386)中，确认所选项。

完成上传后，STEP 7 会显示项目中所有已上传的程序块。

7.9.1 将在线 CPU 与离线 CPU 进行比较

使用 STEP 7 中的“比较”编辑器 (页 1395)，可以查找在线和离线项目之间的差异。 此功能在从 CPU 进行上传之前非常有用。

7.10 调试和测试程序

7.10.1 监视和修改 CPU 中的数据

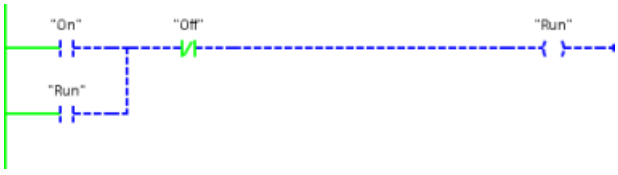
如下表所示，可以监视和修改在线 CPU 中的值。

表格 7-5　使用 STEP 7 监视和修改数据

编辑器	监视	修改	强制
监视表格	有	有	无
强制表格	有	无	有

编辑器	监视	修改	强制
程序编辑器	有	有	无
变量表	有	无	无
DB 编辑器	有	无	无

通过监视表格监视



通过 LAD 编辑器监视

有关监视和修改 CPU 中的数据 (页 1397)的更多信息，请参见“在线和诊断”一章。

7.10.2 监视表格和强制表格

使用“监视表格”监视和修改正在由在线 CPU 执行的用户程序的值。 可在项目中创建并保存不同的监视表格以支持各种测试环境。 这使得用户可以在调试期间或出于维修和维护目的重新进行测试。

通过监视表格，可监视 CPU 并与 CPU 交互，如同 CPU 执行用户程序一样。 不仅可以显示或更改代码块和数据块的变量值，还可以显示或更改 CPU 存储区的值，包括输入和输出（I 和 Q）、外围设备输入 (I:P)、位存储器 (M) 和数据块 (DB)。

通过监视表格，可在 STOP 模式下启用 CPU 的物理输出 (Q:P)。 例如，测试 CPU 的接线时可为输出端赋特定值。

STEP 7 还提供强制表格，用于将变量“强制”设为特定值。 有关强制的更多信息，请参见“在线和诊断”一章的 CPU 中的强制值 (页 1404)一节。

说明

强制值存储在 CPU 中，而不是监视表格中。

无法强制输入（或“I”地址）。 但是，可以强制外围设备输入。 要强制外围设备输入，请在地址后面添加一个“:P”（例如：“On:P”）。

同时，STEP 7 还提供根据触发条件跟踪并记录程序变量 (页 1415)的功能。

7.10.3 用于显示使用情况的交叉引用

巡视窗口可显示有关所选对象在整个项目中使用情况的交叉引用信息，例如用户程序、CPU 以及任何 HMI 设备。 “交叉引用”(Cross-reference) 选项卡显示使用了所选对象的实例和使用该对象的其它对象。 巡视窗口还包括交叉引用中仅在线可用的块。 要显示交叉引用，请选择 “显示交叉引用”(Show cross-references) 命令。 （在项目视图中，可在 “工具”(Tools) 菜单中找到交叉引用。）

说明

不必关闭编辑器即可看到交叉引用信息。

可以对交叉引用中的条目进行排序。 交叉引用列表提供用户程序中存储器地址和变量的使用概况。

- 创建和更改程序时，用户始终能够掌握所使用的操作数、变量和块调用情况。
- 从交叉引用可直接跳转到操作数和变量的使用位置。
- 在程序测试或故障排除期间，系统会通知您哪个块中的哪条命令在处理哪个存储单元、哪个画面在使用哪个变量，以及哪个块被其它哪个块调用。

表格 7-6 交叉引用的元素

列	说明
对象 (Object)	使用下级对象或被下级对象使用的对象的名称
数量	使用数量
使用位置	每个使用位置，例如，程序段
属性 (Property)	被引用对象的特定属性，例如，多重背景声明中的变量名称
作为 (as)	显示对象的更多相关信息，例如，背景数据块用作模板还是用作多重背景
访问 (Access)	访问类型，对操作数的访问是读访问 (R)、写访问 (W) 还是二者的组合。
地址	操作数的地址
类型	有关创建对象所使用的类型和语言的信息
路径 (Path)	对象在项目树中的路径

视安装的产品而定，交叉引用表可能显示额外的列或不同的列。

7.10.4 用于检查调用层级的调用结构

调用结构描述了用户程序中块的调用层级。 其提供了以下几个方面的概要信息：所用的块、对其它块的调用、各个块之间的关系、每个块的数据要求以及块的状态。 可从调用结构打开程序编辑器并对块进行编辑。

显示调用结构时会显示用户程序中使用的块的列表。 STEP 7 高亮显示调用结构的第一级，并显示未被程序中的其它任何块调用的所有块。 调用结构的第一级显示 OB 以及未被 OB 调用的所有 FC、FB 和 DB。如果某个代码块调用了其它块，则被调用块将以缩进方式显示在调用块的下方。 调用结构仅显示被代码块调用的那些块。

可以选择在调用结构中仅显示导致冲突的块。 下列情况会导致冲突：

- 块执行的任何调用具有更旧或更新代码时间戳
- 块所调用块的接口已更改
- 块所使用变量的地址和/或数据类型已更改
- 块未被 OB 直接或间接调用
- 块调用了不存在的块或缺失的块

可以将多个块调用和数据块分为一组。 可使用下拉列表来查看指向各个调用位置的链接。

还可执行一致性检查以显示时间戳冲突。 若在生成程序期间或之后更改块的时间戳，将导致时间戳冲突，而这又会导致调用块和被调用块间出现不一致。

- 通过重新编译代码块可纠正大多数时间戳和接口冲突。
- 如果通过编译无法解决不一致问题，可使用 “详细资料”(Details) 列中的链接转到程序编辑器中的问题源。 然后可手动消除任何不一致情况。
- 必须重新编译所有以红色标记的块。

基本指令 8

8.1 位逻辑运算

8.1.1 位逻辑指令

使用 LAD 和 FBD 处理布尔逻辑非常高效。SCL 不但非常适合处理复杂的数学计算和项目控制结构，而且也可以使用 SCL 处理布尔逻辑。

LAD 触点

表格 8-1 常开触点和常闭触点

LAD	SCL	说明

"IN"
⊣ ⊢
	IF in THEN
	Statement;
	ELSE
	Statement;
	END_IF;
		常开触点和常闭触点：可将触点相互连接并创建用户自己的组合逻辑。如果用户指定的输入位使用存储器标识符 I（输入）或 Q（输出），则从过程映像寄存器中读取位值。控制过程中的物理触点信号会连接到 PLC 上的 I 端子。CPU 扫描已连接的输入信号并持续更新过程映像输入寄存器中的相应状态值。
		通过在 I 偏移量后加上“:P”（例如：“%I3.4:P”），可执行立即读取物理输入。对于立即读取，直接从物理输入读取位数据值，而非从过程映像中读取。立即读取不会更新过程映像。

"IN"
⊣/⊢
	IF NOT (in) THEN
	Statement;
	ELSE
	Statement;
	END_IF;	

表格 8-2 参数的数据类型

参数	数据类型	说明
IN	Bool	分配位

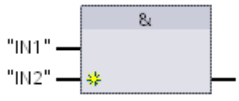
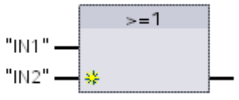
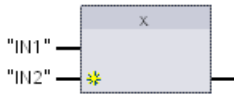
- 在赋的位值为 1 时，常开触点将闭合 (ON)。
- 在赋的位值为 0 时，常闭触点将闭合 (ON)。
- 以串联方式连接的触点创建 AND 逻辑程序段。
- 以并联方式连接的触点创建 OR 逻辑程序段。

FBD、AND、OR 和 XOR 功能框

在 FBD 编程中，LAD 触点程序段变为与 (&)、或 (>=1) 和异或 (x) 功能框程序段，可在其中为功能框输入和输出指定位值。也可以连接到其它逻辑框并创建用户自己的逻辑组合。在程序段中放置功能框后，可从“收藏夹”(Favorites) 工具栏或指令树中拖动“插入输入”(Insert input) 工具，然后将其放置在功能框的输入侧以添加更多输入。也可以右键单击功能框输入连接器并选择“插入输入”(Insert input)。

功能框输入和输出可连接到其它逻辑框，也可输入未连接输入的位地址或位符号名称。执行功能框指令时，当前输入状态会应用到二进制功能框逻辑，如果为真，功能框输出将为真。

表格 8-3　AND、OR 和 XOR 功能框

FBD	SCL[1]	说明
	`out := in1 AND in2;`	AND 功能框的所有输入必须都为“真”，输出才为“真”。
	`out := in1 OR in2;`	OR 功能框只要有一个输入为“真”，输出就为“真”。
	`out := in1 XOR in2;`	XOR 功能框必须有奇数个输入为“真”，输出才为“真”。

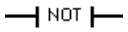
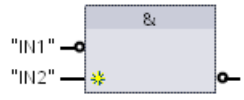
[1] 对于 SCL：必须将运算的结果赋给要用于其它语句的变量。

表格 8-4　参数的数据类型

参数	数据类型	说明
IN1, IN2	Bool	输入位

NOT 逻辑反相器

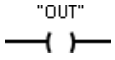
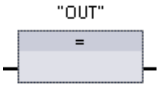
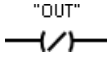
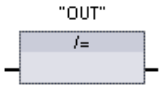
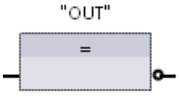
表格 8-5　取反 RLO（逻辑运算结果）

LAD	FBD	SCL	说明
	 	**NOT**	对于 FBD 编程，可从 “收藏夹”(Favorites) 工具栏或指令树中拖动 “取反逻辑运算结果”(Invert RLO) 工具，然后将其放置在输入或输出端以在该功能框连接器上创建逻辑反相器。 LAD NOT 触点取反能流输入的逻辑状态。 ● 如果没有能流流入 NOT 触点，则会有能流流出。 ● 如果有能流流入 NOT 触点，则没有能流流出。

输出线圈和赋值功能框

线圈输出指令写入输出位的值。如果用户指定的输出位使用存储器标识符 Q，则 CPU 接通或断开过程映像寄存器中的输出位，同时将指定的位设置为等于能流状态。控制执行器的输出信号连接到 CPU 的 Q 端子。在 RUN 模式下，CPU 系统将连续扫描输入信号，并根据程序逻辑处理输入状态，然后通过在过程映像输出寄存器中设置新的输出状态值进行响应。CPU 系统会将存储在过程映像寄存器中的新的输出状态响应传送到已连接的输出端子。

表格 8-6　赋值和赋值取反

LAD	FBD	SCL	说明
		**out := <布尔表达式>;**	在 FBD 编程中，LAD 线圈变为分配（= 和 /=）功能框，可在其中为功能框输出指定位地址。功能框输入和输出可连接到其它功能框逻辑，用户也可以输入位地址。 通过在 Q 偏移量后加上“:P”（例如：“%Q3.4:P”），可指定立即写入物理输出。对于立即写入，将位数据值写入过程映像输出并直接写入物理输出。
	 	**out := NOT <布尔表达式>;**	

表格 8-7　参数的数据类型

参数	数据类型	说明
OUT	Bool	分配位

- 如果有能流通过输出线圈或启用了 FBD“=”功能框，则输出位设置为 1。
- 如果没有能流通过输出线圈或未启用 FBD“=”赋值功能框，则输出位设置为 0。
- 如果有能流通过反向输出线圈或启用了 FBD“/=”功能框，则输出位设置为 0。
- 如果没有能流通过反向输出线圈或未启用 FBD“/=”功能框，则输出位设置为 1。

8.1.2 置位和复位指令

置位和复位 1 位

表格 8-8 S 和 R 指令

LAD	FBD	SCL	说明
"OUT" —(S)—	"OUT" S "IN"	不提供	置位输出： S（置位）激活时，OUT 地址处的数据值设置为 1。S 未激活时，OUT 不变。
"OUT" —(R)—	"OUT" R "IN"	不提供	复位输出： R（复位）激活时，OUT 地址处的数据值设置为 0。R 未激活时，OUT 不变。

[1] 对于 LAD 和 FBD： 这些指令可放置在程序段的任何位置。

[2] 对于 SCL： 必须在应用程序内编写代码来复制该函数。

表格 8-9 参数的数据类型

参数	数据类型	说明
IN（或连接到触点/门逻辑）	Bool	要监视位置的位变量
OUT	Bool	要置位或复位位置的位变量

置位和复位位域

表格 8-10　SET_BF 和 RESET_BF 指令

LAD[1]	FBD	SCL	说明
"OUT" —(SET_BF)— "n"	"OUT" SET_BF EN N	不提供	置位位域： SET_BF 激活时，为从寻址变量 OUT 处开始的“n”位分配数据值 1。 SET_BF 未激活时，OUT 不变。
"OUT" —(RESET_BF)— "n"	"OUT" RESET_BF EN N	不提供	复位位域： RESET_BF 为从寻址变量 OUT 处开始的“n”位写入数据值 0。 RESET_BF 未激活时，OUT 不变。

[1] 对于 LAD 和 FBD： 这些指令必须是分支中最右端的指令。

[2] 对于 SCL： 必须在应用程序内编写代码来复制该函数。

表格 8-11　参数的数据类型

参数	数据类型	说明
OUT	Bool	要置位或复位的位域的起始元素（例如： #MyArray[3])
n	常数 (UInt)	要写入的位数

置位优先和复位优先触发器

表格 8-12　RS 和 SR 指令

LAD/FBD	SCL	说明
"INOUT" RS R　Q S1	不提供	复位/置位触发器： RS 是置位优先锁存，其中置位优先。 如果置位 (S1) 和复位 (R) 信号都为真，则地址 INOUT 的值将为 1。
"INOUT" SR S　Q R1	不提供	置位/复位触发器： SR 是复位优先锁存，其中复位优先。 如果置位 (S) 和复位 (R1) 信号都为真，则地址 INOUT 的值将为 0。

[1] 对于 LAD 和 FBD： 这些指令必须是分支中最右端的指令。

[2] 对于 SCL： 必须在应用程序内编写代码来复制该函数。

表格 8-13　参数的数据类型

参数	数据类型	说明
S, S1	Bool	置位输入；1 表示优先
R, R1	Bool	复位输入；1 表示优先
INOUT	Bool	分配的位变量“INOUT”
Q	Bool	遵循“INOUT”位的状态

“INOUT”变量分配要置位或复位的位地址。 可选输出 Q 遵循“INOUT”地址的信号状态。

指令	S1	R	“INOUT”位
RS	0	0	先前状态
	0	1	0
	1	0	1
	1	1	1
	S	R1	
SR	0	0	先前状态
	0	1	0
	1	0	1
	1	1	0

8.1.3 上升沿和下降沿指令

表格 8-14 上升沿和下降沿跳变检测

LAD	FBD	SCL	说明
"IN" P "M_BIT"	"IN" P "M_BIT"	不可用 [1]	扫描操作数的信号上升沿。 LAD：在分配的“IN”位上检测到正跳变（断到通）时，该触点的状态为 TRUE。该触点逻辑状态随后与能流输入状态组合以设置能流输出状态。P 触点可以放置在程序段中除分支结尾外的任何位置。 FBD：在分配的输入位上检测到正跳变（关到开）时，输出逻辑状态为 TRUE。P 功能框只能放置在分支的开头。
"IN" N "M_BIT"	"IN" N "M_BIT"	不可用 [1]	扫描操作数的信号下降沿。 LAD：在分配的输入位上检测到负跳变（开到关）时，该触点的状态为 TRUE。该触点逻辑状态随后与能流输入状态组合以设置能流输出状态。N 触点可以放置在程序段中除分支结尾外的任何位置。 FBD：在分配的输入位上检测到负跳变（开到关）时，输出逻辑状态为 TRUE。N 功能框只能放置在分支的开头。
"OUT" P "M_BIT"	"OUT" P= "M_BIT"	不可用 [1]	在信号上升沿置位操作数。 LAD：在进入线圈的能流中检测到正跳变（关到开）时，分配的位“OUT”为 TRUE。能流输入状态总是通过线圈后变为能流输出状态。P 线圈可以放置在程序段中的任何位置。 FBD：在功能框输入连接的逻辑状态中或输入位赋值中（如果该功能框位于分支开头）检测到正跳变（关到开）时，分配的位“OUT”为 TRUE。输入逻辑状态总是通过功能框后变为输出逻辑状态。P= 功能框可以放置在分支中的任何位置。
"OUT" N "M_BIT"	"OUT" N= "M_BIT"	不可用 [1]	在信号下降沿置位操作数。 LAD：在进入线圈的能流中检测到负跳变（开到关）时，分配的位“OUT”为 TRUE。能流输入状态总是通过线圈后变为能流输出状态。N 线圈可以放置在程序段中的任何位置。 FBD：在功能框输入连接的逻辑状态中或在输入位赋值中（如果该功能框位于分支开头）检测到负跳变（通到断）时，分配的位“OUT”为 TRUE。输入逻辑状态总是通过功能框后变为输出逻辑状态。N= 功能框可以放置在分支中的任何位置。

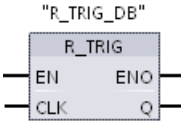
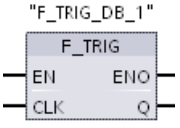
[1] 对于 SCL：必须在应用程序内编写代码来复制该函数。

表格 8-15　P_TRIG 和 N_TRIG

LAD/FBD	SCL	说明
	不可用 [1]	扫描 RLO（逻辑运算结果）的信号上升沿。 在 CLK 输入状态 (FBD) 或 CLK 能流输入 (LAD) 中检测到正跳变（断到通）时，Q 输出能流或逻辑状态为 TRUE。 在 LAD 中，P_TRIG 指令不能放置在程序段的开头或结尾。在 FBD 中，P_TRIG 指令可以放置在除分支结尾外的任何位置。
	不可用 [1]	扫描 RLO 的信号下降沿。 在 CLK 输入状态 (FBD) 或 CLK 能流输入 (LAD) 中检测到负跳变（通到断）时，Q 输出能流或逻辑状态为 TRUE。 在 LAD 中，N_TRIG 指令不能放置在程序段的开头或结尾。在 FBD 中，N_TRIG 指令可以放置在除分支结尾外的任何位置。

[1] 对于 SCL：必须在应用程序内编写代码来复制该函数。

表格 8-16　R_TRIG 和 F_TRIG 指令

LAD/FBD	SCL	说明
	**"R_TRIG_DB"(CLK:=_in_, Q=> _bool_out_);**	在信号上升沿置位变量。 分配的背景数据块用于存储 CLK 输入的前一状态。在 CLK 输入状态 (FBD) 或 CLK 能流输入 (LAD) 中检测到正跳变（断到通）时，Q 输出能流或逻辑状态为 TRUE。 在 LAD 中，R_TRIG 指令不能放置在程序段的开头或结尾。在 FBD 中，R_TRIG 指令可以放置在除分支结尾外的任何位置。
	"F_TRIG_DB"(CLK:=_in_, Q=> _bool_out_);	在信号下降沿置位变量。 分配的背景数据块用于存储 CLK 输入的前一状态。在 CLK 输入状态 (FBD) 或 CLK 能流输入 (LAD) 中检测到负跳变（通到断）时，Q 输出能流或逻辑状态为 TRUE。 在 LAD 中，F_TRIG 指令不能放置在程序段的开头或结尾。在 FBD 中，F_TRIG 指令可以放置在除分支结尾外的任何位置。

在程序中插入 R_TRIG 和 F_TRIG 指令时，将自动打开“调用选项”(Call options) 对话框。在此对话框中，您可以分配
沿存储器位将存储在其自身的数据块中（单个背景）还是作为局部变量（多重背景）存储在块接口中。如果创建了一个单独的数据块，则可以在项目树中“Program resources”文

件夹的
“程序块 > 系统块”(Program blocks > System blocks) 下找到。

表格 8-17　参数的数据类型（P 和 N 触点/线圈、P=、N= 和 P_TRIG and N_TRIG）

参数	数据类型	说明
M_BIT	Bool	保存输入的前一个状态的存储器位
IN	Bool	检测其跳变沿的输入位
OUT	Bool	指示检测到跳变沿的输出位
CLK	Bool	检测其跳变沿的能流或输入位
Q	Bool	指示检测到沿的输出

所有的边沿指令都采用存储位（M_BIT:P/N 触点/线圈，P_TRIG/N_TRIG）或（背景数据块位：R_TRIG, F_TRIG) 保存被监控输入信号的先前状态。通过将输入的状态与前一状态进行比较来检测沿。如果状态指示在关注的方向上有输入变化，则会在输出写入 TRUE 来报告沿。否则，输出会写入 FALSE。

说明

沿指令每次执行时都会对输入和存储器位值进行评估，包括第一次执行。在程序设计期间必须考虑输入和存储器位的初始状态，以允许或避免在第一次扫描时进行沿检测。

由于存储器位必须从一次执行保留到下一次执行，所以应该对每个沿指令都使用唯一的位，并且不应在程序中的任何其它位置使用该位。还应避免使用临时存储器和可受其它系统功能（例如 I/O 更新）影响的存储器。仅将 M、全局 DB 或静态存储器（在背景 DB 中）用于 M_BIT 存储器分配。

8.2 定时器运行

使用定时器指令可创建编程的时间延时。 用户程序中可以使用的定时器数仅受 CPU 存储器容量限制。 每个定时器均使用 16 字节的 IEC_Timer 数据类型的 DB 结构来存储功能框或线圈指令顶部指定的定时器数据。 STEP 7 会在插入指令时自动创建该 DB。

表格 8-18 定时器指令

LAD/FBD 功能框	LAD 线圈	SCL	说明
IEC_Timer_0 TP Time IN Q PT ET	TP_DB TP "PRESET_Tag"	"IEC_Timer_0_DB".TP(IN:=_bool_in_, PT:=_time_in_, Q=>_bool_out_, ET=>_time_out_);	TP 定时器可生成具有预设宽度时间的脉冲。
IEC_Timer_1 TON Time IN Q PT ET	TON_DB TON "PRESET_Tag"	"IEC_Timer_0_DB".TON (IN:=_bool_in_, PT:=_time_in_, Q=>_bool_out_, ET=>_time_out_);	TON 定时器在预设的延时过后将输出 Q 设置为 ON。
IEC_Timer_2 TOF Time IN Q PT ET	TOF_DB TOF "PRESET_Tag"	"IEC_Timer_0_DB".TOF (IN:=_bool_in_, PT:=_time_in_, Q=>_bool_out_, ET=>_time_out_);	TOF 定时器在预设的延时过后将输出 Q 重置为 OFF。
IEC_Timer_3 TONR Time IN Q R ET PT	TONR_DB TONR "PRESET_Tag"	"IEC_Timer_0_DB".TONR (IN:=_bool_in_, R:=_bool_in_, PT:=_time_in_, Q=>_bool_out_, ET=>_time_out_);	TONR 定时器在预设的延时过后将输出 Q 设置为 ON。 在使用 R 输入重置经过的时间之前，会跨越多个定时时段一直累加经过的时间。
仅 FBD： PT PT	TON_DB PT "PRESET_Tag"	PRESET_TIMER(PT:=_time_in_, TIMER:=_iec_timer_in_);	PT（预设定时器）线圈会在指定的 IEC_Timer 中装载新的 PRESET 时间值。
仅 FBD： RT	TON_DB RT	RESET_TIMER(_iec_timer_in_);	RT（复位定时器）线圈会复位指定的 IEC_Timer。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“IEC_Timer_0_DB”是背景 DB 的名称。

表格 8-19　参数的数据类型

参数	数据类型	说明
功能框： IN 线圈： 能流	Bool	TP、TON 和 TONR： 功能框： 0=禁用定时器，1=启用定时器 线圈： 无能流=禁用定时器，能流=启用定时器 TOF： 功能框： 0=启用定时器，1=禁用定时器 线圈： 无能流=启用定时器，能流=禁用定时器
R	Bool	仅 TONR 功能框： 0=不重置 1= 将经过的时间和 Q 位重置为 0
功能框： PT 线圈： "PRESET_Tag"	Time	定时器功能框或线圈： 预设的时间输入
功能框： Q 线圈： DBdata.Q	Bool	定时器功能框： Q 功能框输出或定时器 DB 数据中的 Q 位 定时器线圈： 仅可寻址定时器 DB 数据中的 Q 位
功能框： ET 线圈： DBdata.ET	Time	定时器功能框： ET（经历的时间）功能框输出或定时器 DB 数据中的 ET 时间值 定时器线圈： 仅可寻址定时器 DB 数据中的 ET 时间值。

表格 8-20　PT 和 IN 参数值变化的影响

定时器	PT 和 IN 功能框参数和相应线圈参数的变化
TP	• 定时器运行期间，更改 PT 没有任何影响。 • 定时器运行期间，更改 IN 没有任何影响。
TON	• 定时器运行期间，更改 PT 没有任何影响。 • 定时器运行期间，将 IN 更改为 FALSE 会复位并停止定时器。
TOF	• 定时器运行期间，更改 PT 没有任何影响。 • 定时器运行期间，将 IN 更改为 TRUE 会复位并停止定时器。
TONR	• 定时器运行期间更改 PT 没有任何影响，但对定时器中断后继续运行会有影响。 • 定时器运行期间将 IN 更改为 FALSE 会停止定时器但不会复位定时器。 将 IN 改回 TRUE 将使定时器从累积的时间值开始定时。

PT（预设时间）和 ET（经过的时间）值以表示毫秒时间的有符号双精度整数形式存储在指定的 IEC_TIMER DB 数据中。 TIME 数据使用 T# 标识符，可以简单时间单元（T#200ms 或 200）和复合时间单元（如 T#2s_200ms）的形式输入。

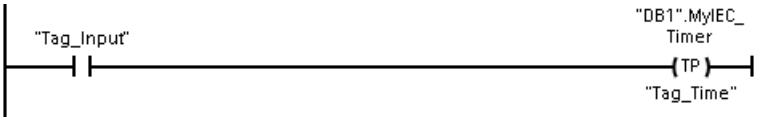
表格 8-21　TIME 数据类型的大小和范围

数据类型	大小	有效数值范围 [1]
TIME	32 位，以 DInt 数据的形式存储	T#-24d_20h_31m_23s_648ms 到 T#24d_20h_31m_23s_647ms 以 -2,147,483,648 ms 到 +2,147,483,647 ms 的形式存储

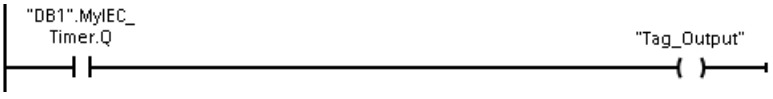
[1] 在定时器指令中，无法使用上面所示 TIME 数据类型的负数范围。 负的 PT（预设时间）值在定时器指令执行时被设置为 0。 ET（经过的时间）始终为正值。

定时器线圈示例

-(TP)-、-(TON)-、-(TOF)- 和 -(TONR)- 定时器线圈必须是 LAD 网络中的最后一个指令。 如定时器示例中所示，后面网络中的触点指令会求出定时器线圈 IEC_Timer DB 数据中的 Q 位值。 同样，如果要在程序中使用经过的时间值，必须访问 IEC_timer DB 数据中的 ELAPSED 元素。



当 Tag_Input 位的值由 0 转换为 1 时，脉冲定时器启动。 定时器开始运行并持续 Tag_Time 时间值指定的时间。



只要定时器运行，就存在 DB1.MyIEC_Timer.Q 状态=1 且 Tag_Output 值=1。当经过 Tag_Time 值后，DB1.MyIEC_Timer.Q=0 且 Tag_Output 值=0。

重置定时器 -(RT)- 和预设定时器 -(PT)- 线圈

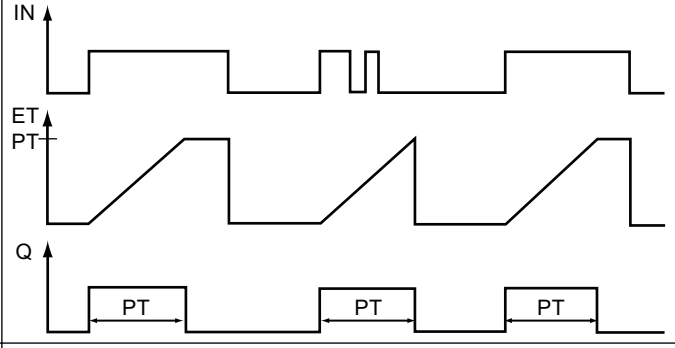
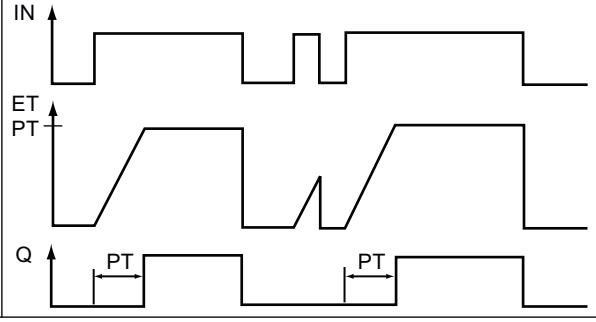
这些线圈指令可与功能框或线圈定时器一起使用并可放置在中间位置。 线圈输出能流状态始终与线圈输入状态相同。 若 -(RT)- 线圈激活，指定 IEC_Timer DB 数据中的 ELAPSED 时间元素将重置为 0。若 -(PT)- 线圈激活，使用所分配的时间间隔值加载指定 IEC_Timer DB 数据中的 PRESET 时间元素。

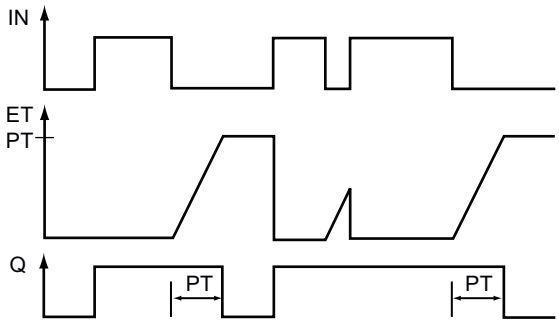
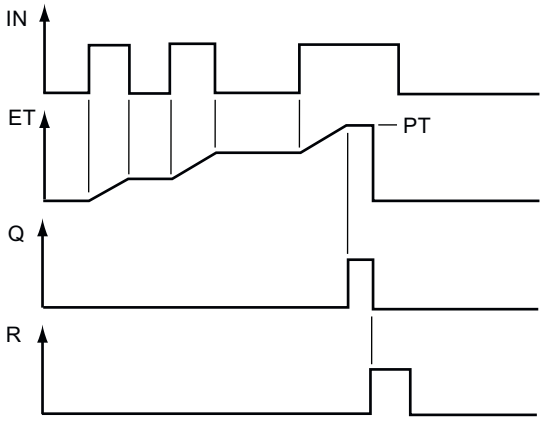
说明

在 FB 中放置定时器指令时，可以选择 “多重背景数据块”(Multi-instance data block) 选项。各定时器结构名称可以对应不同的数据结构，但定时器数据包含在同一个数据块中，无需为每个定时器都使用一个独立的数据块。 这样可减少处理定时器所需的处理时间和数据存储空间。 在共享的多重背景数据块中的定时器数据结构之间不存在交互作用。

定时器的运行

表格 8-22　IEC 定时器的类型

定时器	时序图
TP：生成脉冲 TP 定时器可生成具有预设宽度时间的脉冲。	
TON：接通延时 TON 定时器在预设的延时过后将输出 Q 设置为 ON。	

定时器	时序图
TOF： 关断延时 TOF 定时器在预设的延时过后将输出 Q 重置为 OFF。	
TONR： 时间累加器 TONR 定时器在预设的延时过后将输出 Q 设置为 ON。 在使用 R 输入重置经过的时间之前，会跨越多个定时时段一直累加经过的时间。	

说明

在 CPU 中，没有给任何特定的定时器指令分配专门的资源。 每个定时器使用 DB 存储器中其自身的结构和一个连续运行的内部 CPU 定时器来执行定时。

当由于 TP、TON、TOF 或 TONR 指令的输入上出现沿跳变而启动定时器时，连续运行的内部 CPU 定时器的值将被复制到为该定时器指令分配的 DB 结构的 START 成员中。 该起始值在定时器继续运行期间将保持不变，随后将在每次更新定时器时使用。 每次启动定时器时，都会从内部 CPU 定时器将一个新的起始值加载到定时器结构中。

更新定时器时，将从内部 CPU 定时器的当前值中减去上述起始值以确定经过的时间。 再将经过的时间与预设值进行比较以确定定时器 Q 位的状态。 然后在为该定时器分配的 DB 结构中，更新 ELAPSED 和 Q 成员。 注意，经过的时间将停留在预设值上（达到预设值后定时器便不会继续累加经过的时间）。

当且仅当满足以下条件时才会执行定时器更新：

- 已执行定时器指令（TP、TON、TOF 或 TONR）
- 某个指令直接引用 DB 中定时器结构的“ELAPSED”成员
- 某个指令直接引用 DB 中定时器结构的“Q”成员

定时器编程

规划和创建用户程序时应考虑以下定时器运行说明：

- 可在同一个扫描周期内多次更新定时器。 每次执行定时器指令（TP、TON、TOF、TONR）和每次将定时器结构的 ELAPSED 或 Q 成员用作其它已执行指令的参数时，都会更新定时器。 这在需要最新时间数据（本质上是立即读取定时器）时会是一项优点。 但是，如果希望在整个程序扫描周期内保持一致的值，则请将定时器指令放置在需要这些值的其它所有指令之前，并使用定时器指令的 Q 和 ET 输出中的变量而不是定时器 DB 结构的 ELAPSED 和 Q 成员。
- 扫描期间可以不执行定时器更新。 可以在函数中启动定时器，然后在一个或多个扫描周期内不再调用该函数。 如果没有执行引用定时器结构中 ELAPSED 或 Q 成员的其它指令，则不会更新定时器。 直到再次执行定时器指令或执行将定时器结构的 ELAPSED 或 Q 用作参数的其它指令时，才会再次更新定时器。
- 尽管并不常见，但可以将同一个 DB 定时器结构分配给多个定时器指令。 通常，为避免意外交互作用，应当使每个 DB 定时器结构仅对应一个定时器指令（TP、TON、TOF、TONR）。
- 自复位定时器适合用于触发需要周期性发生的动作。 通常，将引用定时器位的常闭触点放置在定时器指令前面可创建自复位定时器。 该定时器网络通常位于使用该定时器位来触发动作的一个或多个依赖型网络上面。 当定时器时间已到（经过的时间达到预设值）时，定时器位将在一个扫描周期内为 ON，因而可执行由该定时器位控制的依赖型网络逻辑。 下次执行定时器网络时，常闭触点将为 OFF，从而复位定时器并清除定时器位。 下次扫描期间，常闭触点将为 ON，因此将重启定时器。 创建此类自复位定时器时，请勿将定时器 DB 结构的“Q”成员用作该定时器指令前面常闭触点的参数。 而是要使用与该定时器指令的“Q”输出相连的变量。 如果访问定时器 DB 结构的 Q 成员，将导致定时器更新，且如果因常闭触点而更新定时器，该触点将立即复位该定时器。 定时器指令的 Q 输出将在一个扫描周期内不为 ON，并且依赖型网络不会执行。

RUN-STOP-RUN 切换或 CPU 循环上电后保留时间数据

如果从运行模式阶段切换到停止模式或 CPU 循环上电并启动了新运行模式阶段，则存储在之前运行模式阶段中的定时器数据将丢失，除非将定时器数据结构指定为具有保持性（TP、TON、TOF 和 TONR 定时器）。

将定时器指令放到程序编辑器中后，如果接受调用选项对话框中的默认设置，则将自动分配一个**无法实现具有保持性**的背景数据块。 要使定时器数据具有保持性，必须使用全局数据块或多重背景数据块。

指定全局数据块将定时器数据存储为保持性数据

无论将定时器放在什么位置（OB、FC 或 FB），该选项都有效。

1. 创建一个全局数据块：
   - 在项目树中双击“添加新块”(Add new block)。
   - 单击数据块 (DB) 图标
   - 对于“类型”(Type)，选择“全局数据块”(global DB)。
   - 如果希望能够将该数据块中各数据元素选择为具有保持性，则确保选中数据块类型“优化”(Optimized) 框。 另一个数据块类型选项“标准 - 与 S7-300/400 兼容”(Standard - compatible with S7-300/400) 仅允许将所有 DB 数据元素都设置为具有保持性或没有保持性。
   - 单击“确定”(OK)
2. 向该数据块中添加定时器结构：
   - 在新的全局数据块中，添加 IEC_Timer 数据类型的静态变量。
   - 在“保持性”(Retain) 列中，选中相应框以使该结构具有保持性。
   - 重复此过程为要存储在该数据块中的所有定时器创建结构。 可以将每个定时器结构放置在独立的全局数据块中，也可以将多个定时器结构放置在同一个全局数据块中。 除定时器外，还可以将其它静态变量放置在该全局数据块中。 将多个定时器结构放置在同一个全局数据块中可减少总的块数。
   - 可根据需要重命名定时器结构。
3. 打开程序块来选择保持性定时器的放置位置（OB、FC 或 FB）。
4. 将定时器指令放置在所需位置。
5. 在调用选项对话框出现后，单击“取消”按钮。
6. 在新的定时器指令上方，输入上面所创建全局数据块和定时器结构的名称（请勿使用助手浏览）（例如：“Data_block_3.Static_1”）。

指定多重背景数据块以将定时器数据存储为保持性数据

该选项仅对于将定时器放置在 FB 中有效。

该选项取决于 FB 属性是否指定“优化块访问”(Optimized block access)（仅允许符号访问）。 要检查现有 FB 访问属性的组态情况，请在项目树中右键单击该 FB，选择“属性”(Properties)，然后选择“特性”(Attributes)。

如果 FB 指定“优化块访问”(Optimized block access)（仅允许符号访问）：

1. 打开 FB 进行编辑。
2. 将定时器指令放在 FB 中的所需位置。
3. “调用选项”(Call options) 对话框出现后，单击“多重背景”(Multi instance) 图标。 仅在将该指令放置于 FB 中后，“多重背景”(Multi instance) 选项才可用。
4. 如有需要，请在“调用选项”(Call options) 对话框中重命名定时器。

5. 单击“确定”(OK)。 定时器指令将出现在编辑器中，而 IEC_TIMER 结构将出现在“FB 接口”(FB Interface) 的“静态”(Static) 下。
6. 如有必要，打开 FB 接口编辑器（可能需要单击小箭头以展开视图）。
7. 在“静态”(Static) 下，找到刚刚创建的定时器结构。
8. 在此定时器结构的“保持性”(Retain) 列中，改为选择“保持性”(Retain)。 此后只要从另一程序块调用此 FB，都将利用此接口定义（包含标有保持性的定时器结构）创建背景数据块。

如果 FB 未指定“优化块访问”(Optimized block access)，则块访问类型为标准访问，标准访问与 S7-300/400 传统组态兼容，且允许符号访问和直接访问。 要将多重背景分配给标准块访问 FB，请按以下步骤操作：

1. 打开 FB 进行编辑。
2. 将定时器指令放在 FB 中的所需位置。
3. “调用选项”(Call options) 对话框出现后，单击“多重背景”(Multi instance) 图标。 仅在将该指令放置于 FB 中后，“多重背景”(Multi instance) 选项才可用。
4. 如有需要，请在“调用选项”(Call options) 对话框中重命名定时器。
5. 单击“确定”(OK)。 定时器指令将出现在编辑器中，而 IEC_TIMER 结构将出现在“FB 接口”(FB Interface) 的“静态”(Static) 下。
6. 打开将使用此 FB 的块。
7. 将此 FB 置于所需的位置。 如此将为该 FB 创建一个背景数据块。
8. 打开将 FB 放入编辑器时创建的背景数据块。
9. 在“静态”(Static) 下，找到所需的定时器结构。 在此定时器结构的“保持性”(Retain) 列中，选中相应框使该结构具有保持性。

8.3 计数器操作

表格 8-23 计数器指令

LAD/FBD	SCL	说明
"Counter name" CTU Int CU Q R CV PV	"IEC_Counter_0_DB".CTU(CU:=_bool_in, R:=_bool_in, PV:=_in, Q=>_bool_out, CV=>_out);	可使用计数器指令对内部程序事件和外部过程事件进行计数。每个计数器都使用数据块中存储的结构来保存计数器数据。用户在编辑器中放置计数器指令时分配相应的数据块。 • CTU 是加计数器 • CTD 是减计数器 • CTUD 是加减计数器
"Counter name" CTD Int CD Q LD CV PV	"IEC_Counter_0_DB".CTD(CD:=_bool_in, LD:=_bool_in, PV:=_in, Q=>_bool_out, CV=>_out);	
"Counter name" CTUD Int CU QU CD QD R CV LD PV	"IEC_Counter_0_DB".CTUD(CU:=_bool_in, CD:=_bool_in, R:=_bool_in, LD:=_bool_in, PV:=_in_, QU=>_bool_out, QD=>_bool_out, CV=>_out_);	

[1] 对于 LAD 和 FBD：从指令名称下的下拉列表中选择计数值数据类型。

[2] STEP 7 会在插入指令时自动创建 DB。

[3] 在 SCL 示例中，“IEC_Counter_0_DB”是背景 DB 的名称。

表格 8-24 参数的数据类型

参数	数据类型 [1]	说明
CU, CD	Bool	加计数或减计数，按加或减一计数
R (CTU, CTUD)	Bool	将计数值重置为零
LD (CTD, CTUD)	Bool	预设值的装载控制
PV	SInt, Int, DInt, USInt, UInt, UDInt	预设计数值
Q, QU	Bool	CV >= PV 时为真

参数	数据类型 [1]	说明
QD	Bool	CV <= 0 时为真
CV	SInt, Int, DInt, USInt, UInt, UDInt	当前计数值

[1] 计数值的数值范围取决于所选的数据类型。如果计数值是无符号整型数，则可以减计数到零或加计数到范围限值。如果计数值是有符号整数，则可以减计数到负整数限值或加计数到正整数限值。

用户程序中可以使用的计数器数仅受 CPU 存储器容量限制。计数器占用以下存储器空间：

- 对于 SInt 或 USInt 数据类型，计数器指令占用 3 个字节。
- 对于 Int 或 UInt 数据类型，计数器指令占用 6 个字节。
- 对于 DInt 或 UDInt 数据类型，计数器指令占用 12 个字节。

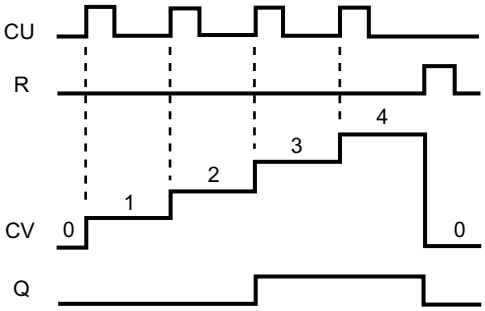
这些指令使用软件计数器，软件计数器的最大计数速率受其所在的 OB 的执行速率限制。指令所在的 OB 的执行频率必须足够高，以检测 CU 或 CD 输入的所有跳变。要了解更快的计数操作，请参见 CTRL_HSC 指令 (页 559)。

说明

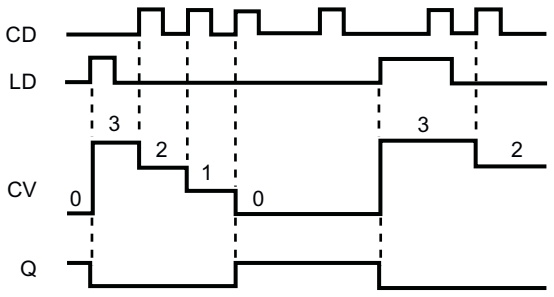
在 FB 中放置计数器指令后，可以选择多重背景数据块选项，各计数器结构名称可以对应不同的数据结构，但计数器数据包含在同一个数据块中，从而无需每个计数器都使用一个单独的数据块。这减少了计数器所需的处理时间和数据存储空间。在共享的多重背景数据块中的计数器数据结构之间不存在交互作用。

计数器的运行

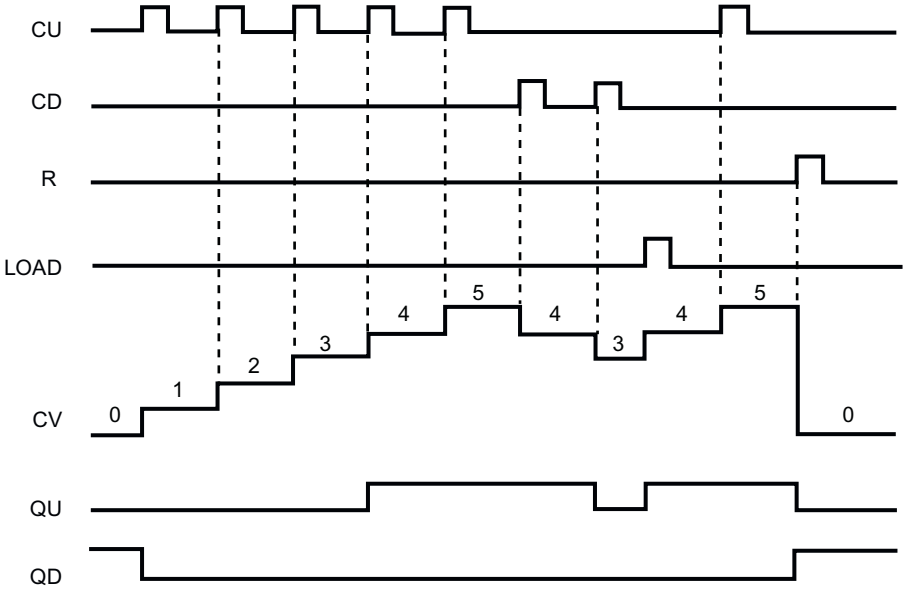
表格 8-25　CTU 运算（加计数）

计数器	运行
当参数 CU 的值从 0 变为 1 时，CTU 计数器会使计数值加 1。CTU 时序图显示了计数值为无符号整数时的运行（其中，PV = 3）。 • 如果参数 CV（当前计数值）的值大于或等于参数 PV（预设计数值）的值，则计数器输出参数 Q = 1。 • 如果复位参数 R 的值从 0 变为 1，则当前计数值重置为 0。	

表格 8-26　CTD 运算（减计数）

计数器	运行
当参数 CD 的值从 0 变为 1 时，CTD 计数器会使计数值减 1。CTD 时序图显示了计数值为无符号整数时的运行（其中，PV = 3）。 ● 如果参数 CV（当前计数值）的值等于或小于 0，则计数器输出参数 Q = 1。 ● 如果参数 LOAD 的值从 0 变为 1，则参数 PV（预设值）的值将作为新的 CV（当前计数值）装载到计数器。	

表格 8-27　CTUD 运算（加计数和减计数）

计数器	运行
当加计数 (CU) 输入或减计数 (CD) 输入从 0 转换为 1 时，CTUD 计数器将加 1 或减 1。CTUD 时序图显示了计数值为无符号整数时的运行（其中 PV = 4）。 ● 如果参数 CV 的值大于等于参数 PV 的值，则计数器输出参数 QU = 1。 ● 如果参数 CV 的值小于或等于零，则计数器输出参数 QD = 1。 ● 如果参数 LOAD 的值从 0 变为 1，则参数 PV 的值将作为新的 CV 装载到计数器。 ● 如果复位参数 R 的值从 0 变为 1，则当前计数值重置为 0。	

RUN-STOP-RUN 切换或 CPU 循环上电后保留计数器数据

如果从运行模式阶段切换到停止模式或 CPU 循环上电并启动了新运行模式阶段，则存储在之前运行模式阶段中的计数器数据将丢失，除非将定时器数据结构指定为具有保持性（CTU、CTD 和 CTUD 计数器）。

将计数器指令放到程序编辑器中后，如果接受调用选项对话框中的默认设置，则将自动分配一个**无法实现具有保持性**的背景数据块。 要使计数器数据具有保持性，必须使用全局数据块或多重背景数据块。

指定全局数据块将计数器数据存储为保持性数据

无论将计数器放在什么位置（OB、FC 或 FB），该选项都有效。

1. 创建一个全局数据块：
   - 在项目树中双击“添加新块”(Add new block)。
   - 单击数据块 (DB) 图标
   - 对于“类型”(Type)，选择“全局数据块”(global DB)。
   - 如果希望能够将该数据块中的各个项选择为具有保持性，则确保选中“仅符号访问”(symbolic-access-only) 框。
   - 单击“确定”(OK)
2. 向该数据块添加计数器结构：
   - 在新的全局数据块中，添加使用以下计数器数据类型之一的新静态变量。 务必要考虑到想要用于预设值和计数值的类型。
   - 在“保持性”(Retain) 列中，选中相应框以使该结构具有保持性。
   - 重复此过程为要存储在该数据块中的所有计数器创建结构。 可以将每个计数器结构放置在独立的全局数据块中，也可以将多个计数器结构放置在同一个全局数据块中。 除计数器外，还可以将其它静态变量放置在该全局数据块中。 将多个计数器结构放置在同一个全局数据块中可减少总的块数。
   - 可根据需要重命名计数器结构。
3. 打开程序块来选择保持性计数器的放置位置（OB、FC 或 FB）。
4. 将计数器指令放置在所需位置。
5. 在调用选项对话框出现后，单击“取消”按钮。 您现在应该看到新的计数器指令，在指令名称的上面和下面均显示“???”。
6. 在新的计数器指令上方，输入上面所创建全局数据块和计数器结构的名称（请勿使用助手浏览）（例如：“Data_block_3.Static_1”）。 这需要填入对应的预设值和计数值类型（例如：UInt 对应于 IEC_UCounter 结构）。

计数器数据类型	预设值和计数值的相应类型
IEC_Counter	INT
IEC_SCounter	SINT
IEC_DCounter	DINT
IEC_UCounter	UINT
IEC_USCounter	USINT
IEC_UDCounter	UDINT

指定多重背景数据块以将计数器数据存储为保持性数据

该选项仅对于将计数器放置在 FB 中有效。

该选项取决于 FB 属性是否指定“优化块访问”(Optimized block access)（仅允许符号访问）。 要检查现有 FB 访问属性的组态情况，请在项目树中右键单击该 FB，选择“属性”(Properties)，然后选择“特性”(Attributes)。

如果 FB 指定“优化块访问”(Optimized block access)（仅允许符号访问）：

1. 打开 FB 进行编辑。
2. 将计数器指令放在 FB 中的所需位置。
3. “调用选项”(Call options) 对话框出现后，单击“多重背景”(Multi instance) 图标。 仅在将该指令放置于 FB 中后，“多重背景”(Multi instance) 选项才可用。
4. 如有需要，请在“调用选项”(Call options) 对话框中重命名计数器。
5. 单击“确定”(OK)。 计数器指令将出现在编辑器中并且预设值和计数值的类型为 INT，而 IEC_COUNTER 结构将出现在“FB 接口”(FB Interface) 的“静态”(Static) 下。
6. 如有需要，请在计数器指令中将类型从 INT 更改为其它类型之一。 计数器结构将相应更改。
7. 如有必要，打开 FB 接口编辑器（可能需要单击小箭头以展开视图）。
8. 在“静态”(Static) 下，找到刚刚创建的计数器结构。
9. 在此计数器结构的“保持性”(Retain) 列中，改为选择“保持性”(Retain)。 此后只要从另一程序块调用此 FB，都将利用此接口定义（包含标有保持性的计数器结构）创建背景数据块。

如果 FB 未指定“优化块访问”(Optimized block access)，则块访问类型为标准访问，标准访问与 S7-300/400 传统组态兼容，且允许符号访问和直接访问。 要将多重背景分配给标准块访问 FB，请按以下步骤操作：

1. 打开 FB 进行编辑。
2. 将计数器指令放在 FB 中的所需位置。
3. “调用选项”(Call options) 对话框出现后，单击“多重背景”(Multi instance) 图标。 仅在将该指令放置于 FB 中后，“多重背景”(Multi instance) 选项才可用。
4. 如有需要，请在“调用选项”(Call options) 对话框中重命名计数器。
5. 单击“确定”(OK)。 计数器指令将出现在编辑器中并且预设值和计数值的类型为 INT，而 IEC_COUNTER 结构将出现在“FB 接口”(FB Interface) 的“静态”(Static) 下。
6. 如有需要，请在计数器指令中将类型从 INT 更改为其它类型之一。 计数器结构将相应更改。
7. 打开将使用此 FB 的块。
8. 将此 FB 置于所需的位置。 如此将为该 FB 创建一个背景数据块。
9. 打开将 FB 放入编辑器时创建的背景数据块。
10. 在“静态”(Static) 下，找到所需的计数器结构。 在此计数器结构的“保持性”(Retain) 列中，选中相应框使该结构具有保持性。

计数器指令中显示的类型（对于预设值和计数 FB 接口中显示的对应的结构类型值）

INT	IEC_Counter
SINT	IEC_SCounter
DINT	IEC_DCounter
UINT	IEC_UCounter
USINT	IEC_USCounter
UDINT	IEC_UDCounter

8.4 比较运算

8.4.1 比较值指令

表格 8-28 比较指令

LAD	FBD	SCL	说明

"IN1" == Byte "IN2"	== Byte "IN1" IN1 "IN2" IN2	`out := in1 = in2;`
		or
		`IF in1 = in2`
		`THEN out := 1;`
		`ELSE out := 0;`
		`END_IF;`
			比较数据类型相同的两个值。该 LAD 触点比较结果为 TRUE 时，则该触点会被激活。如果该 FBD 功能框比较结果为 TRUE，则功能框输出为 TRUE。

[1] 对于 LAD 和 FBD：单击指令名称（如“==”），以从下拉列表中更改比较类型。单击“???”并从下拉列表中选择数据类型。

表格 8-29 参数的数据类型

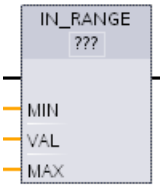
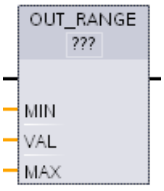
参数	数据类型	说明
IN1, IN2	Byte, Word, DWord, SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, String, WString, Char, Char, Time, Date, TOD, DTL, 常数	要比较的值

表格 8-30　比较说明

关系类型	满足以下条件时比较结果为真 ...
=	IN1 等于 IN2
<>	IN1 不等于 IN2
>=	IN1 大于或等于 IN2
<=	IN1 小于或等于 IN2
>	IN1 大于 IN2
<	IN1 小于 IN2

8.4.2　IN_Range（范围内值）和 OUT_Range（范围外值）

表格 8-31　范围内值和范围外值指令

LAD/FBD	SCL	说明
	`out := IN_RANGE(min, val, max);`	测试输入值是在指定的值范围之内还是之外。 如果比较结果为 TRUE，则功能框输出为 TRUE。
	`out := OUT_RANGE(min, val, max);`	

[1] 对于 LAD 和 FBD：单击“???”并从下拉列表中选择数据类型。

表格 8-32　参数的数据类型

参数	数据类型 [1]	说明
MIN, VAL, MAX	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, 常数	比较器输入

[1] 输入参数 MIN、VAL 和 MAX 的数据类型必须相同。

- 满足以下条件时 IN_RANGE 比较结果为真：MIN <= VAL <= MAX
- 满足以下条件时 OUT_RANGE 比较结果为真：VAL < MIN 或 VAL > MAX

8.4.3 OK（检查有效性）和 NOT_OK（检查无效性）

表格 8-33 OK（检查有效性）和 Not OK（检查无效性）指令

LAD	FBD	SCL	说明
"IN" OK	"IN" OK	不提供	测试输入数据参考是否为符合 IEEE 规范 754 的有效实数。
"IN" NOT_OK	"IN" NOT_OK	不提供	

[1] 对于 LAD 和 FBD：如果该 LAD 触点为 TRUE，则激活该触点并传递能流。如果该 FBD 功能框为 TRUE，则功能框输出为 TRUE。

表格 8-34 参数的数据类型

参数	数据类型	说明
IN	Real, LReal	输入数据

表格 8-35 操作

指令	满足以下条件时 REAL 数测试结果为 TRUE：
OK	输入值为有效实数 [1]
NOT_OK	输入值不是有效实数 [1]

[1] 如果 Real 或 LReal 类型的值为 +/- INF（无穷大）、NaN（不是数字）或者非标准化的值，则其无效。非标准化的值是非常接近于 0 的数字。CPU 在计算中用 0 替换非标准化的值。

8.4.4 变型和数组比较指令

8.4.4.1 相同和不同比较指令

S7-1200 CPU 提供了用于查询 Variant 操作数所指向的变量的数据类型是否与另一个操作数的数据类型相同的指令。

此外，S7-1200 CPU 还提供了用于查询数组元素的数据类型是否与另一个操作数的数据类型相同的指令。

在这些指令中，将 <Operand1> 与 <Operand2> 进行比较。 <Operand1> 的数据类型必须为 Variant。<Operand2> 可以是 PLC 数据类型的基本数据类型。在 LAD 和 FBD 中，<Operand1> 是指令上方的操作数。在 LAD 中，<Operand2> 是指令下方的操作数。

对于所有指令，如果通过相同或不同测试，则逻辑运算结果 (RLO) 为 1 (true)，否则为 0 (false)。

相同和不同类型比较指令如下所示：

- EQ_Type（比较数据类型与变量数据类型是否“相等”）
- NE_Type（比较数据类型与变量数据类型是否“不相等”）
- EQ_ElemType（比较 ARRAY 元素数据类型与变量数据类型是否“相等”）
- NE_ElemType（比较 ARRAY 元素数据类型与变量数据类型是否“不相等”）

表格 8-36　EQ 和 NE 指令

LAD	FBD	SCL	描述
#Operand1 EQ_Type "Operand2"	#Operand1 EQ_Type "Operand2" IN2 OUT	不可用	测试 Operand1 处的变型所指向的变量是否与 Operand2 处的变量具备相同的数据类型。
#Operand1 NE_Type "Operand2"	#Operand1 NE_Type "Operand2" IN2 OUT	不可用	测试 Operand1 处的变型所指向的变量是否与 Operand2 处的变量具备不同的数据类型。
#Operand1 EQ_ElemType "Operand2"	#Operand1 EQ_ElemType "Operand2" IN2 OUT	不可用	测试 Operand1 处的变型所指向的数组元素是否与 Operand2 处的变量具备相同的数据类型。
#Operand1 NE_ElemType "Operand2"	#Operand1 NE_ElemType "Operand2" IN2 OUT	不可用	测试 Operand1 处的变型所指向的数组元素是否与 Operand2 处的变量具备不同的数据类型。

表格 8-37　参数的数据类型

参数	数据类型	描述
Operand1	Variant	第一个操作数
Operand2	位字符串、整数、浮点数、定时器、日期和时间、字符串、ARRAY、PLC 数据类型	第二个操作数

8.4.4.2 空比较指令

可以使用指令 IS_NULL 和 NOT_NULL 来决定输入是否实际上指向对象。

对于两个指令来说，<Operand> 必须为 Variant 数据类型。

表格 8-38 IS_NULL（查询等于零的指针）和 NOT_NULL（查询等于零的指针）指令

LAD	FBD	SCL	说明
#Operand IS_NULL	#Operand IS_NULL OUT	不提供	测试 Operand 的 Variant 所指向的变量是否为空，即不指向任何对象。
#Operand NOT_NULL	#Operand NOT_NULL OUT	不提供	测试 Operand 的 Variant 所指向的变量是否不为空，即指向一个对象。

表格 8-39 参数的数据类型

参数	数据类型	说明
Operand	Variant	用于评估是否为空的操作数。

8.4.4.3 IS_ARRAY（检查数组）

可以使用“检查数组”指令来查询 Variant 是否指向 Array 数据类型的变量。

<操作数> 必须为 Variant 数据类型。

如果操作数是数组，则指令返回 1 (true)。

表格 8-40 IS_ARRAY（检查数组）

LAD	FBD	SCL	说明
#Operand IS_ARRAY	#Operand IS_ARRAY OUT	IS_ARRAY(_variant_in_)	测试 Operand 的 Variant 所指向的变量是否为数组。

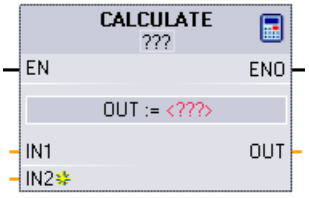
表格 8-41 参数的数据类型

参数	数据类型	说明
Operand	Variant	评估是否为数组的操作数。

8.5 数学函数

8.5.1 CALCULATE（计算）

表格 8-42 CALCULATE 指令

LAD/FBD	SCL	说明
	使用标准 SCL 数学表达式创建等式。	CALCULATE 指令可用于创建作用于多个输入上的数学函数（IN1，IN2，.. INn），并根据您定义的等式在 OUT 处生成结果。 ● 首先选择数据类型。 所有输入和输出的数据类型必须相同。 ● 要添加其它输入，请单击最后一个输入处的图标。

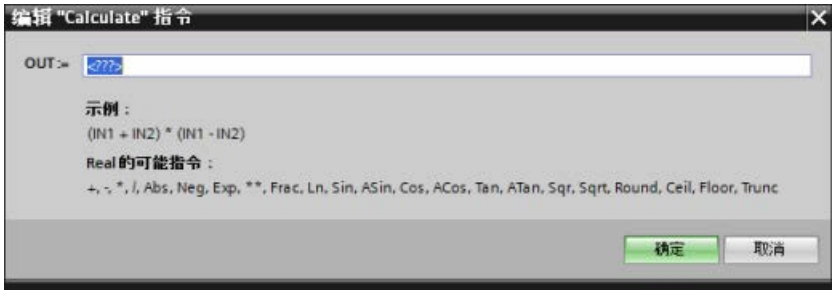
表格 8-43 参数的数据类型

参数	数据类型 [1]
IN1, IN2, ..INn	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord

[1] IN 和 OUT 参数必须具有相同的数据类型（通过对输入参数进行隐式转换）。 例如： 如果 OUT 是 INT 或 REAL，则 SINT 输入值将转换为 INT 或 REAL 值

单击计算器图标可打开对话框，在其中定义数学函数。 输入等式作为输入（如 IN1 和 IN2）和操作数。 单击 “确定”(OK) 保存函数时，对话框会自动生成 CALCULATE 指令的输入。

对话框显示一个示例，以及可根据 OUT 参数的数据类型加入的一列指令：



说明

还必须为函数中的任何常量生成输入。 然后会在指令 CALCULATE 的相关输入中输入该常量值。

通过输入常量作为输入，可将 CALCULATE 指令复制到用户程序的其它位置，从而无需更改函数。 之后，不需要修改函数，就可以更改指令输入的值或变量。

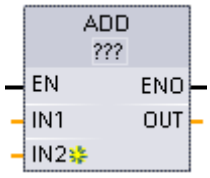
当执行 CALCULATE 并成功完成计算中的所有单个运算时，ENO = 1，否则 ENO = 0。

有关 CALCULATE 指令的示例，请参见“使用简单指令创建复杂等式 (页 43)”。

8.5.2 加法、减法、乘法和除法指令

表格 8-44 加法、减法、乘法和除法指令

LAD/FBD	SCL	说明

	`out := in1 + in2;`
	`out := in1 - in2;`
	`out := in1 * in2;`
	`out := in1 / in2;`
		• ADD：加法 (IN1 + IN2 = OUT)
		• SUB：减法 (IN1 - IN2 = OUT)
		• MUL：乘法 (IN1 * IN2 = OUT)
		• DIV：除法 (IN1 / IN2 = OUT)
		整数除法运算会截去商的小数部分以生成整数输出。

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

表格 8-45 参数的数据类型（LAD 和 FBD）

参数	数据类型 [1]	说明
IN1, IN2	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, 常数	数学运算输入
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	数学运算输出

[1] 参数 IN1、IN2 和 OUT 的数据类型必须相同。

要添加 ADD 或 MUL 输入，请单击“创建”(Create) 图标，或在其中一个现有 IN 参数的输入短线处单击右键，并选择“插入输入”(Insert input) 命令。

要删除输入，请在其中一个现有 IN 参数（多于两个原始输入时）的输入短线处单击右键，并选择“删除”(Delete) 命令。

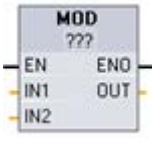
启用数学指令 (EN = 1) 后，指令会对输入值（IN1 和 IN2）执行指定的运算并将结果存储在通过输出参数 (OUT) 指定的存储器地址中。运算成功完成后，指令会设置 ENO = 1。

表格 8-46　ENO 状态

ENO	说明
1	无错误
0	数学运算结果值可能超出所选数据类型的有效数值范围。返回适合目标大小的结果的最低有效部分。
0	除数为 0 (IN2 = 0)：结果未定义，返回 0。
0	Real/LReal：如果其中一个输入值为 NaN（不是数字），则返回 NaN。
0	ADD Real/LReal：如果两个 IN 值均为 INF，但符号不同，则这是非法运算并返回 NaN。
0	SUB Real/LReal：如果两个 IN 值均为 INF，且符号相同，则这是非法运算并返回 NaN。
0	MUL Real/LReal：如果一个 IN 值为零而另一个为 INF，则这是非法运算并返回 NaN。
0	DIV Real/LReal：如果两个 IN 值均为零或 INF，则这是非法运算并返回 NaN。

8.5.3　MOD（返回除法的余数）

表格 8-47　求模（返回除法的余数）指令

LAD/FBD	SCL	说明
	`out := in1 MOD in2;`	可以使用 MOD 指令返回整数除法运算的余数。用输入 IN1 的值除以输入 IN2 的值，在输出 OUT 中返回余数。

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

表格 8-48　参数的数据类型

参数	数据类型 [1]	说明
IN1 和 IN2	SInt, Int, DInt, USInt, UInt, UDInt, 常数	求模输入
OUT	SInt, Int, DInt, USInt, UInt, UDInt	求模输出

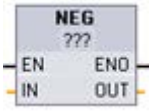
[1] 参数 IN1、IN2 和 OUT 的数据类型必须相同。

表格 8-49　ENO 值

ENO	说明
1	无错误
0	值 IN2 = 0，OUT 被赋以零值

8.5.4　NEG（取反）

表格 8-50　NEG（求二进制补码）指令

LAD/FBD	SCL	说明
	`-(in);`	使用 NEG 指令可将参数 IN 的值的算术符号取反并将结果存储在参数 OUT 中。

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

表格 8-51　参数的数据类型

参数	数据类型 [1]	说明
IN	SInt, Int, DInt, Real, LReal, Constant	数学运算输入
OUT	SInt, Int, DInt, Real, LReal	数学运算输出

[1] 参数 IN 和 OUT 的数据类型必须相同。

表格 8-52　ENO 状态

ENO	说明
1	无错误
0	结果值超出所选数据类型的有效数值范围。 以 SInt 为例：NEG (-128) 的结果为 +128，超出该数据类型的最大值。

8.5.5 INC（递增）和 DEC（递减）

表格 8-53 INC 和 DEC 指令

LAD/FBD	SCL	说明
	`in_out := in_out + 1;`	递增有符号或无符号整数值： IN_OUT 值 +1 = IN_OUT 值
	`in_out := in_out - 1;`	递减有符号或无符号整数值： IN_OUT 值 - 1 = IN_OUT 值

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

表格 8-54 参数的数据类型

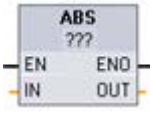
参数	数据类型	说明
IN/OUT	SInt, Int, DInt, USInt, UInt, UDInt	数学运算输入和输出

表格 8-55 ENO 状态

ENO	说明
1	无错误
0	结果值超出所选数据类型的有效数值范围。 SInt 示例：(+127) INC 的结果为 +128，超出该数据类型的最大值。

8.5.6 ABS（计算绝对值）

表格 8-56 ABS（绝对值）指令

LAD/FBD	SCL	说明
	`out := ABS(in);`	计算参数 IN 的有符号整数或实数的绝对值并将结果存储在参数 OUT 中。

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

表格 8-57　参数的数据类型

参数	数据类型 [1]	说明
IN	SInt, Int, DInt, Real, LReal	数学运算输入
OUT	SInt, Int, DInt, Real, LReal	数学运算输出

[1] 参数 IN 和 OUT 的数据类型必须相同。

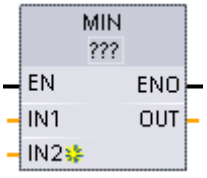
表格 8-58　ENO 状态

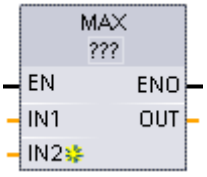
ENO	说明
1	无错误
0	数学运算结果值超出所选数据类型的有效数值范围。 SInt 示例：(-128) ABS 的结果为 +128，超出该数据类型的最大值。

8.5.7　MIN（获取最小值）和 MAX（获取最大值）

表格 8-59　MIN（获取最小值）和 MAX（获取最大值）指令

LAD/FBD	SCL	说明

	`out:= MIN(`
	`in1:=_variant_in_,`
	`in2:=_variant_in_`
	`[,...in32]);`
		MIN 指令用于比较两个参数 IN1 和 IN2 的值并将最小（较小）值分配给参数 OUT。

	`out:= MAX(`
	`in1:=_variant_in_,`
	`in2:=_variant_in_`
	`[,...in32]);`
		MAX 指令用于比较两个参数 IN1 和 IN2 的值并将最大（较大）值分配给参数 OUT。

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

表格 8-60　参数的数据类型

参数	数据类型 [1]	说明
IN1, IN2 [...IN32]	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Time, Date, TOD, 常数	数学运算输入（最多 32 个输入）
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Time, Date, TOD	数学运算输出

[1] IN1、IN2 和 OUT 参数的数据类型必须相同。

要添加输入，请单击“创建”(Create) 图标，或在其中一个现有 IN 参数的输入短线处单击右键，并选择“插入输入”(Insert input) 命令。

要删除输入，请在其中一个现有 IN 参数（多于两个原始输入时）的输入短线处单击右键，并选择“删除”(Delete) 命令。

表格 8-61　ENO 状态

ENO	说明
1	无错误
0	仅适用于 Real 数据类型： ● 至少一个输入不是实数 (NaN)。 ● 结果 OUT 为 +/- INF（无穷大）。

8.5.8　LIMIT（设置限值）

表格 8-62　LIMIT（设置限值）指令

LAD/FBD	SCL	说明

	`LIMIT(MN:=_variant_in_,`
	`IN:=_variant_in_,`
	`MX:=_variant_in_,`
	`OUT:=_variant_out_);`
		Limit 指令用于测试参数 IN 的值是否在参数 MIN 和 MAX and if not, clamps the value at MIN or MAX. 指定的值范围内

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

表格 8-63　参数的数据类型

参数	数据类型 [1]	说明
MN, IN 和 MX	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Time, Date, TOD·常数	数学运算输入
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Time, Date, TOD	数学运算输出

[1] 参数 MN、IN、MX 和 OUT 的数据类型必须相同。

如果参数 IN 的值在指定的范围内，则 IN 的值将存储在参数 OUT 中。如果参数 IN 的值超出指定的范围，则 OUT 值为参数 MIN 的值（如果 IN 值小于 MIN 值）或参数 MAX 的值（如果 IN 值大于 MAX 值）。

表格 8-64　ENO 状态

ENO	说明
1	无错误
0	Real：如果 MIN、IN 和 MAX 的一个或多个值是 NaN（不是数字），则返回 NaN。
0	如果 MIN 大于 MAX，则将值 IN 分配给 OUT。

SCL 示例：

- MyVal := LIMIT(MN:=10,IN:=53, MX:=40); //结果：MyVal = 40
- MyVal := LIMIT(MN:=10,IN:=37, MX:=40); //结果：MyVal = 37
- MyVal := LIMIT(MN:=10,IN:=8, MX:=40); //结果：MyVal = 10

8.5.9 指数、对数及三角函数指令

使用浮点指令可编写使用 Real 或 LReal 数据类型的数学运算程序：

- SQR：计算平方 ($IN^2 = OUT$)
- SQRT：计算平方根 ($\sqrt{IN} = OUT$)
- LN：计算自然对数 (LN(IN) = OUT)
- EXP：计算指数值 ($e^{IN} = OUT$)，其中底数 e = 2.71828182845904523536
- EXPT：取幂 ($IN1^{IN2} = OUT$)
  EXPT 参数 IN1 和 OUT 总是为同一数据类型，可以选定为 Real 或 LReal。可以从众多数据类型中为指数参数 IN2 选择数据类型。
- FRAC：提取小数（浮点数 IN 的小数部分 = OUT）

- SIN:计算正弦值 (sin(IN radians) = OUT)
- ASIN：计算反正弦值（arcsine(IN) = OUT 弧度），其中 sin(OUT 弧度) = IN
- COS：计算余弦（cos(IN 弧度) = OUT）
- ACOS：计算反余弦值 (arccos(IN) = OUT 弧度)，其中 cos(OUT 弧度) = IN
- TAN：计算正切值（tan(IN 弧度) = OUT）
- ATAN：计算反正切值 (arctan(IN) = OUT 弧度)，其中 tan(OUT 弧度) = IN

表格 8-65　浮点型数学运算指令示例

LAD/FBD	SCL	说明

	`out := SQR(in);`
	或
	`out := in * in;`
		平方：IN [2] = OUT
		例如：如果 IN = 9，则 OUT = 81。

	`out := in1 ** in2;`	综合指数：IN1 ^{IN2} = OUT 例如：如果 IN1 = 3 且 IN2 = 2，则 OUT = 9。

[1] 对于 LAD 和 FBD：单击“???”（按指令名称）并从下拉菜单中选择数据类型。

[2] 对于 SCL：还可以使用基本的 SCL 数学运算符来创建数学表达式。

表格 8-66　参数的数据类型

参数	数据类型	说明
IN, IN1	Real, LReal, Constant	输入
IN2	SInt, Int, DInt, USInt, UInt,UDInt, Real, LReal, Constant	EXPT 指数输入
OUT	Real, LReal	输出

表格 8-67　ENO 状态

ENO	指令	条件	结果 (OUT)
1	全部	无错误	有效结果
0	SQR	结果超出有效 Real/LReal 范围	+INF
		IN 为 +/- NaN（不是数字）	+NaN
	SQRT	IN 为负数	-NaN
		IN 为 +/- INF（无穷大）或 +/- NaN	+/- INF 或 +/- NaN
	LN	IN 为 0.0、负数、-INF 或 -NaN	-NaN
		IN 为 +INF 或 +NaN	+INF 或 +NaN
	EXP	结果超出有效 Real/LReal 范围	+INF
		IN 为 +/- NaN	+/- NaN
	SIN, COS, TAN	IN 为 +/- INF 或 +/- NaN	+/- INF 或 +/- NaN
	ASIN，ACOS	IN 超出 -1.0 到 +1.0 的有效范围	+NaN
		IN 为 +/- NaN	+/- NaN
	ATAN	IN 为 +/- NaN	+/- NaN
	FRAC	IN 为 +/- INF 或 +/- NaN	+NaN
	EXPT	IN1 为 +INF 且 IN2 不是 -INF	+INF
		IN1 为负数或 -INF	如果 IN2 为 Real/LReal，则为 +NaN， 否则为 -INF
		IN1 或 IN2 为 +/- NaN	+NaN
		IN1 为 0.0 且 IN2 为 Real/LReal（只能为 Real/LReal）	+NaN

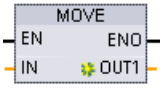
8.6 移动操作

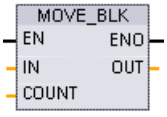
8.6.1 MOVE（移动值）、MOVE_BLK（移动块）、UMOVE_BLK（无中断移动块）和 MOVE_BLK_VARIANT（移动块）

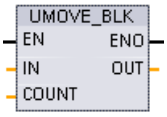
使用移动指令可将数据元素复制到新的存储器地址并从一种数据类型转换为另一种数据类型。移动过程不会更改源数据。

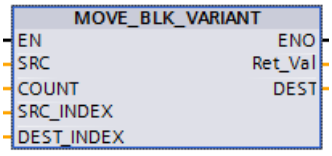
- MOVE 指令用于将单个数据元素从参数 IN 指定的源地址复制到参数 OUT 指定的目标地址。
- MOVE_BLK 和 UMOVE_BLK 指令具有附加的 COUNT 参数。COUNT 指定要复制的数据元素个数。每个被复制元素的字节数取决于 PLC 变量表中分配给 IN 和 OUT 参数变量名称的数据类型。

表格 8-68　MOVE、MOVE_BLK、UMOVE_BLK 和 MOVE_BLK_VARIANT 指令

LAD/FBD	SCL	说明
	`out1 := in;`	将存储在指定地址的数据元素复制到新地址或多个地址。[1]

	`MOVE_BLK(`
	`in:=_variant_in,`
	`count:=_uint_in,`
	`out=>_variant_out);`
		将数据元素块复制到新地址的可中断移动。

	`UMOVE_BLK(`
	`in:=_variant_in,`
	`count:=_uint_in,`
	`out=>_variant_out);`
		将数据元素块复制到新地址的不可中断移动。

	`MOVE_BLK(`
	`SRC:=_variant_in,`
	`COUNT:=_udint_in,`
	`SRC_INDEX:=_dint_in,`
	`DEST_INDEX:=_dint_in,`
	`DEST=>_variant_out);`
		将源存储区域的内容移动到目标存储区域。
		可以将一个完整的数组或数组中的元素复制到另一个具有相同数据类型的数组中。源数组和目标数组的大小（元素数量）可以不同。可以复制数组中的多个或单个元素。源数组和目标数组都可以用 Variant 数据类型来指代。

[1] MOVE 指令：要在 LAD 或 FBD 中添加其它输出，请单击输出参数旁的“创建”(Create) 图标。对于 SCL，请使用多个赋值语句。还可以使用任一循环结构。

表格 8-69　MOVE 指令的数据类型

参数	数据类型	说明
IN	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Char, WChar, Array, Struct, DTL, Time, Date, TOD, IEC 数据类型，PLC 数据类型	源地址
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Char, WChar, Array, Struct, DTL, Time, Date, TOD, IEC 数据类型，PLC 数据类型	目标地址

要添加 MOVE 输出，请单击“创建”(Create) 图标，或右键单击现有 OUT 参数之一的输出短线，并选择“插入输出”(Insert output) 命令。

要删除输出，请在其中一个现有 OUT 参数（多于两个原始输出时）的输出短线处单击右键，并选择“删除”(Delete) 命令。

表格 8-70　MOVE_BLK 和 UMOVE_BLK 指令的数据类型

参数	数据类型	说明
IN	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal Byte, Word, DWord, Time, Date, TOD, WChar	源起始地址
COUNT	UInt	要复制的数据元素数
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, WChar	目标起始地址

表格 8-71　MOVE_BLK_VARIANT 指令的数据类型

参数	数据类型	说明
SRC	Variant（指向数组或单独的数组元素）	欲进行复制操作的源块
COUNT	UDInt	要复制的数据元素数
SRC_INDEX	DInt	SRC 数组的零基索引
DEST_INDEX	DInt	DEST 数组的零基索引
RET_VAL	Int	错误信息
DEST	Variant（指向数组或单独的数组元素）	源块内容所要复制到的目标区域

说明

数据复制操作规则

- 要复制 Bool 数据类型，请使用 SET_BF、RESET_BF、R、S 或输出线圈 (LAD) (页 216)
- 要复制单个基本数据类型，请使用 MOVE
- 要复制基本数据类型数组，请使用 MOVE_BLK 或 UMOVE_BLK
- 要复制结构，请使用 MOVE
- 要复制字符串，请使用 S_MOVE (页 348)
- 要复制字符串中的单个字符，请使用 MOVE
- MOVE_BLK 和 UMOVE_BLK 指令不能用于将数组或结构复制到 I、Q 或 M 存储区。

MOVE_BLK 和 UMOVE_BLK 指令在处理中断的方式上有所不同：

- 在 MOVE_BLK 执行期间**排队并处理**中断事件。在中断 OB 子程序中未使用移动目标地址的数据时，或者虽然使用了该数据，但目标数据不必一致时，使用 MOVE_BLK 指令。如果 MOVE_BLK 操作被中断，则最后移动的一个数据元素在目标地址中是完整并且一致的。MOVE_BLK 操作会在中断 OB 执行完成后继续执行。
- 在 UMOVE_BLK 完成执行前**排队但不处理**中断事件。如果在执行中断 OB 子程序前移动操作必须完成且目标数据必须一致，则使用 UMOVE_BLK 指令。更多信息，请参阅数据一致性 (页 186)部分。

执行 MOVE 指令之后，ENO 始终为真。

表格 8-72 ENO 状态

ENO	条件	结果
1	无错误	成功复制了全部的 COUNT 个元素。
0	源 (IN) 范围或目标 (OUT) 范围超出可用存储区。	复制适当的元素。不复制部分元素。

表格 8-73 MOVE_BLK_VARIANT 指令的条件代码

RET_VAL (W#16#...)	说明
0000	无错误
80B4	数据类型不匹配。
8151	不能访问参数 SRC。
8152	SRC 参数中的操作数为无效类型。
8153	参数 SRC 生成代码时出错

RET_VAL (W#16#...)	说明
8154	参数 SRC 的操作数的数据类型为 Bool。
8281	参数 COUNT 的值无效。
8382	参数 SRC_INDEX 的值超出 Variant 限制范围。
8383	参数 SRC_INDEX 的值超出数组的上限。
8482	参数 DEST_INDEX 的值超出 Variant 限制。
8483	参数 DEST_INDEX 的值超出数组的上限。
8534	参数 DEST 受写保护。
8551	不能访问参数 DEST。
8552	DEST 参数的操作数为无效类型。
8553	参数 DEST 生成代码时出错
8554	参数 DEST 的操作数对应的数据类型为 Bool。
* 错误代码可在程序编辑器中显示为整数或十六进制值。	

8.6.2 Deserialize

可以使用“取消序列化”指令将 PLC 数据类型 (UDT) 块的顺序表示转换回 PLC 数据类型并填充所有内容。如果比较结果为 TRUE，则功能框输出为 TRUE。

按顺序表达的 PLC 数据类型所对应的存储区必须采用 Array of Byte 数据类型，并且必须为数据块声明标准的访问方式，而不是优化访问方式。转换前要确保有足够的存储空间。

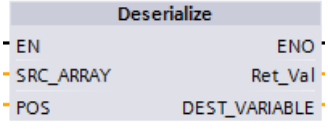
该指令可以将多个按顺序表示的已转换 PLC 数据类型重新转换回之前的原始数据类型。

说明

如果只想转换一个按顺序表达的 PLC 数据类型 (UDT)，也可以使用指令“TRCV：通过通信连接接收数据”。

表格 8-74 DESERIALIZE 指令

LAD/FBD	SCL	说明

	`ret_val := Deserialize(`
	`SRC_ARRAY:=_variant_in_,`
	`DEST_VARIABLE=>_variant_out_,`
	`POS:=_dint_inout_);`
		将按顺序表达的 PLC 数据类型 (UDT) 转换回 PLC 数据类型，并填充整个内容

表格 8-75　DESERIALIZE 指令的参数

参数	类型	数据类型	说明
SRC_ARRAY	IN	Variant	包含数据流的全局数据块
DEST_VARIABLE	INOUT	Variant	已转换的 PLC 数据类型 (UDT) 存储所在的变量
POS	INOUT	DInt	已转换的 PLC 数据类型所使用的字节数
RET_VAL	OUT	Int	错误信息

表格 8-76　RET_VAL 参数

RET_VAL* (W#16#...)	说明
0000	无错误
80B0	SRC_ARRAY 和 DEST_VARIABLE 参数的存储区重叠。
8136	DEST_VARIABLE 参数的数据块未采用标准访问类型。
8150	参数 SRC_ARRAY 的 Variant 数据类型不含任何值。
8151	参数 SRC_ARRAY 生成代码时出错。
8153	SRC_ARRAY 参数的存储空间不足。
8250	参数 DEST_VARIABLE 的 Variant 数据类型不含任何值。
8251	参数 DEST_VARIABLE 生成代码时出错。
8254	DEST_VARIABLE 参数的数据类型无效。
8382	参数 POS 的值超出数组的限制。
* 可以在程序编辑器中以整数或十六进制的形式查看错误代码。	

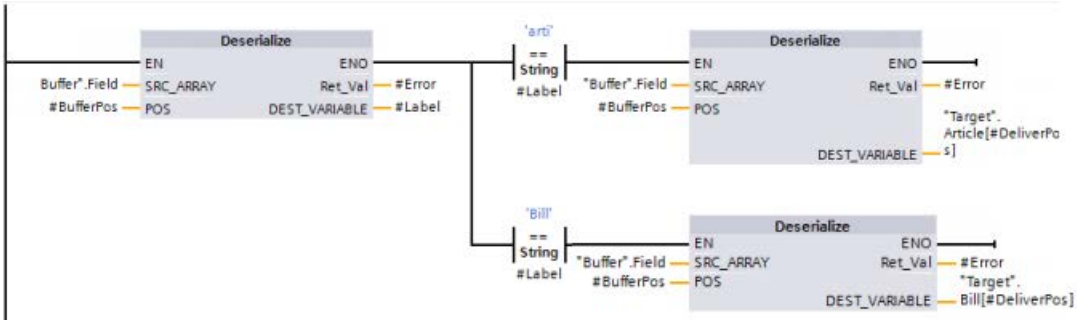
示例：Deserialize 指令

以下举例说明了该指令的工作原理：

程序段 1：

“MOVE”指令将值“0”移动到“#BufferPos”数据块变量。然后 Deserialize 指令将对“Buffer”数据块中按顺序表达的客户数据进行反序列化，并将其写入到“Target”数据块中。Deserialize 指令计算已转换的数据所占的字节数，并将其存储到“#BufferPos”数据块变量。

程序段 2：



“Deserialize”指令对“Buffer”所指向的按顺序表达的数据流进行反序列化，并将相应字符写入到“#Label”操作数中。逻辑将使用比较指令“arti”和“Bill”来对字符进行比较。如果“arti”的比较结果为 TRUE，则数据为部件数据，将进行反序列化并写入到“Target”数据块的“article”数据结构中。如果“Bill”的比较结果为 TRUE，则数据为计费数据，将进行反序列化并写入到“Target”数据块的“Bill”数据结构中。

函数块（或函数）接口：

		名称	数据类型
1		Input	
2		DeliverPos	Int
3		Output	
4		InOut	
5		Static	
6		Temp	
7		BufferPos	DInt
8		Error	Int
9		Label	String[4]

自定义 PLC 数据类型：

以下为两个 PLC 数据类型 (UDT) 的结构示例：

Article

		名称	数据类型
1		Number	DInt
2		Declaration	String
3		Colli	Int

Client

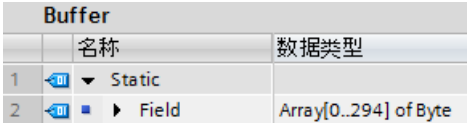
		名称	数据类型
1		Title	Int
2		Firstname	String[10]
3		Surname	String[10]

数据块：

两个数据块示例如下：

Target

	名称	数据类型
1	Static	
2	Client	"Client"
3	Article	Array[0..10] of "Article"
4	Bill	Array[0..10] of Int



Buffer

	名称	数据类型
1	Static	
2	Field	Array[0..294] of Byte

8.6.3 Serialize

可以使用“Serialize”指令将多个 PLC 数据类型 (UDT) 转换成按顺序表达的版本，并且不丢失结构。

可以使用此指令将程序中的多个结构化数据项暂时保存到缓冲区中（例如，保存到全局数据块中），并发送给另一 CPU。存储已转换的 PLC 数据类型的存储区必须采用 ARRAY of BYTE 数据类型，并且已声明为标准访问方式。转换前要确保有足够的存储空间。

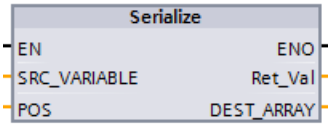
POS 参数包含有关已转换的 PLC 数据类型所占字节数的信息。

说明

如果只想发送一个 PLC 数据类型 (UDT)，可以使用指令“TSEND：通过通信连接发送数据”。

表格 8-77 SERIALIZE 指令

LAD/FBD	SCL	说明

	`ret_val := Serialize(`
	`SRC_VARIABLE=>_variant_in_,`
	`DEST_ARRAY:=_variant_out_,`
	`POS:=_dint_inout_);`
		将 PLC 数据类型 (UDT) 转换为按顺序表达的版本。

表格 8-78 SERIALIZE 指令的参数

参数	类型	数据类型	说明
SRC_VARIABLE	IN	Variant	待转换为按顺序表达版本的 PLC 数据类型 (UDT)
DEST_ARRAY	INOUT	Variant	作为所生成的数据流的存储目标的数据块

参数	类型	数据类型	说明
POS	INOUT	DInt	已转换的 PLC 数据类型所使用的字节数。计算出的 POS 参数是从零开始的。
RET_VAL	OUT	Int	错误信息

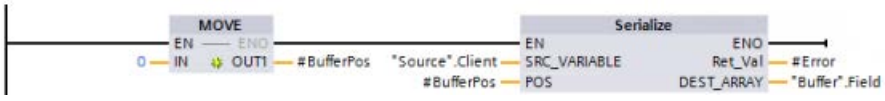
表格 8-79　RET_VAL 参数

RET_VAL* (W#16#...)	说明
0000	无错误
80B0	SRC_VARIABLE 和 DEST_ARRAY 参数的存储区重叠。
8150	参数 SRC_VARIABLE 的 Variant 数据类型不含任何值。
8152	参数 SRC_VARIABLE 生成代码时出错。
8236	DEST_ARRAY 参数的数据块未采用标准访问类型。
8250	参数 DEST_ARRAY 的 Variant 数据类型不含任何值。
8252	参数 DEST_ARRAY 生成代码时出错。
8253	DEST_ARRAY 参数的存储空间不足。
8254	DEST_VARIABLE 参数的数据类型无效。
8382	参数 POS 的值超出数组的限制。
* 可以在程序编辑器中以整数或十六进制的形式查看错误代码。	

示例：Serialize 指令

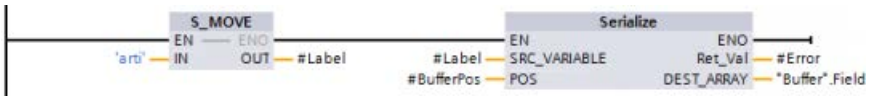
以下举例说明了该指令的工作原理：

程序段 1：



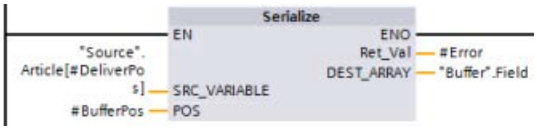
“MOVE”指令将值“0”移动到“#BufferPos”参数。Serialize 指令将对“Source”数据块中的客户数据进行序列化，并以按顺序表达的方式将其写入到“Buffer”数据块中。该指令会将按顺序表达的版本中所用的字节数存储到“#BufferPos”参数中。

程序段 2：



逻辑此时将插入一些分隔符文本，以便利以后对顺序表达版本进行反序列化。“S_MOVE”指令将文本字符串“arti”移动到“#Label”参数。“Serialize”指令将源客户数据后的这些字符写入到“Buffer”数据块中。此指令将把文本字符串“arti”所含的字节数累加到“#BufferPos”参数的已存数据中。

程序段 3：



“Serialize”指令将序列化“Source”数据块中特定部件的数据（该数据在运行期间计算），并以按顺序表达的方式写入到“Buffer”数据块的“arti”字符后面。

块接口：

		名称	数据类型
1		Input	
2		DeliverPos	Int
3		Output	
4		InOut	
5		Static	
6		Temp	
7		BufferPos	DInt
8		Error	Int
9		Label	String[4]

自定义 PLC 数据类型：

以下为两个 PLC 数据类型 (UDT) 的结构示例：

Article

		名称	数据类型
1		Number	DInt
2		Declaration	String
3		Colli	Int

Client

		名称	数据类型
1		Title	Int
2		Firstname	String[10]
3		Surname	String[10]

数据块：

两个数据块示例如下：

Source

		名称	数据类型
1		Static	
2		Client	"Client"
3		Article	Array[0..10] of "Article"

Buffer

		名称	数据类型
1		Static	
2		Field	Array[0..294] of Byte

8.6.4 FILL_BLK（填充块）和 UFILL_BLK（无中断填充块）

表格 8-80 FILL_BLK 和 UFILL_BLK 指令

LAD/FBD	SCL	说明

	`FILL_BLK(`
	`in:=_variant_in,`
	`count:=int,`
	`out=>_variant_out);`
		可中断填充指令：使用指定数据元素的副本填充地址范围

	`UFILL_BLK(`
	`in:=_variant_in,`
	`count:=int,`
	`out=>_variant_out);`
		不中断填充指令：使用指定数据元素的副本填充地址范围

表格 8-81 参数的数据类型

参数	数据类型	说明
IN	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	数据源地址
COUNT	UDint, USInt, UInt	要复制的数据元素数
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	数据目标地址

说明

数据填充操作规则

- 要使用 BOOL 数据类型填充，请使用 SET_BF、RESET_BF、R、S 或输出线圈 (LAD)
- 要使用单个基本数据类型填充，请使用 MOVE
- 要使用基本数据类型填充数组，请使用 FILL_BLK 或 UFILL_BLK
- 要填充字符串中的单个字符，请使用 MOVE
- FILL_BLK 和 UFILL_BLK 指令不能用于将数组填充到 I、Q 或 M 存储区。

FILL_BLK 和 UFILL_BLK 指令可将源数据元素 IN 复制到通过参数 OUT 指定初始地址的目标中。复制过程不断重复并填充相邻的一组地址，直到副本数等于 COUNT 参数。

FILL_BLK 和 UFILL_BLK 指令在处理中断的方式上有所不同：

- 在 FILL_BLK 执行期间**排队并处理**中断事件。在中断 OB 子程序中未使用移动目标地址的数据时，或者虽然使用了该数据，但目标数据不必一致时，使用 FILL_BLK 指令。
- 在 UFILL_BLK 完成执行前**排队但不处理**中断事件。如果在执行中断 OB 子程序前移动操作必须完成且目标数据必须一致，则使用 UFILL_BLK 指令。

表格 8-82　ENO 状态

ENO	条件	结果
1	无错误	IN 元素成功复制到全部的 COUNT 个目标中。
0	目标 (OUT) 范围超出可用存储区	复制适当的元素。不复制部分元素。

8.6.5　SWAP（交换字节）

表格 8-83　SWAP 指令

LAD/FBD	SCL	说明
	`out := SWAP(in);`	用于反转二字节和四字节数据元素的字节顺序。不改变每个字节中的位顺序。执行 SWAP 指令之后，ENO 始终为 TRUE。

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

表格 8-84　参数的数据类型

参数	数据类型	说明
IN	Word, DWord	有序数据字节 IN
OUT	Word, DWord	反转有序数据字节 OUT

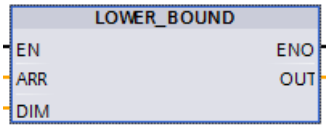
示例 1	参数 IN = MB0（执行前）		参数 OUT = MB4，（执行后）	
地址	MW0	MB1	MW4	MB5
W#16#1234	12	34	34	12
WORD	MSB	LSB	MSB	LSB

示例 2	参数 IN = MB0（执行前）				参数 OUT = MB4，（执行后）			
地址	MD0	MB1	MB2	MB3	MD4	MB5	MB6	MB7
DW#16#12345678	12	34	56	78	78	56	34	12
DWORD	MSB			LSB	MSB			LSB

8.6.6 LOWER_BOUND：（读取 ARRAY 下限）

表格 8-85 LOWER_BOUND 指令

LAD/FBD	SCL	说明

	`out := LOWER_BOUND(`
	`ARR:=_variant_in_,`
	`DIM:=_udint_in_);`
		在块接口中，可声明 ARRAY[*] 的变量。这些局部变量可读取 ARRAY 限值。此时，需要在 DIM 参数中指定维数。
		LOWER_BOUND（读取 ARRAY 下限）指令允许读取 ARRAY 的变量下限。

参数

下表列出了指令“LOWER_BOUND：读取 ARRAY 下限”：

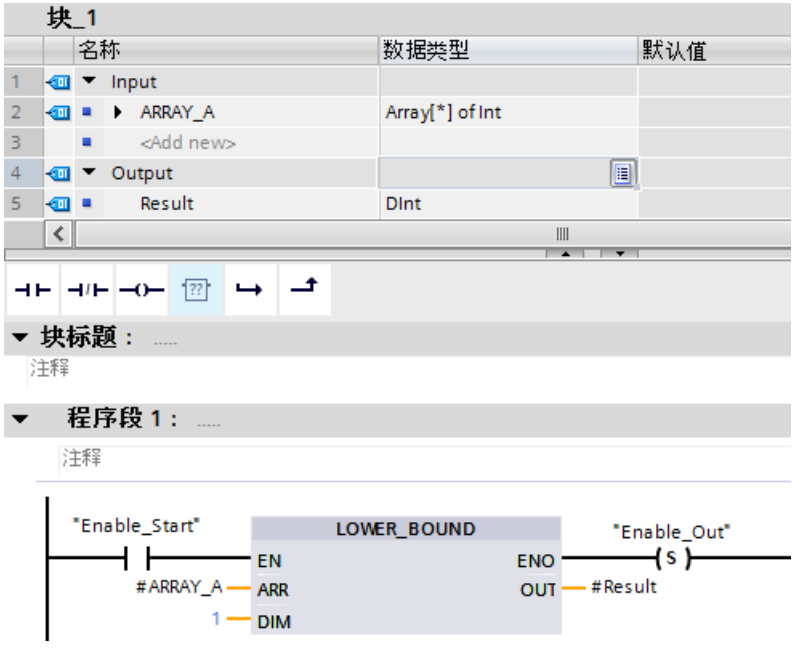
参数	声明	数据类型	存储区	说明
EN	Input	BOOL	I、Q、M、D、L	使能输入
ENO	Output	BOOL	I、Q、M、D、L	如果满足下列条件之一，则使能输出 ENO 的信号状态为“0”： ● 使能输入 EN 的信号状态为“0”。 ● 输入 DIM 处指定的维数不存在。
ARR	Input	ARRAY [*]	FB：InOut 部分 FC：Input 和 InOut 部分	待读取可变下限的 ARRAY。

参数	声明	数据类型	存储区	说明
DIM	Input	UDINT	I、Q、M、D、L 或常数	待读取可变下限的 ARRAY 维度。
OUT	Output	DINT	I、Q、M、D、L	结果

有关有效数据类型的更多信息，请参见 “数据类型 (页 119)”：

示例

在函数 (FC) 块界面中，输入参数 ARRAY_A 是具有可变维数的一维数组。

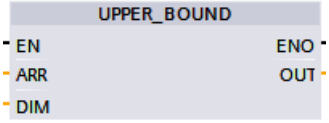


如果操作数“Enable_Start”返回信号状态“1”，则 CPU 执行 LOWER_BOUND 指令。该指令从一维数组中读取 ARRAY #ARRAY_A 的可变下限。如果指令执行未发生错误，该指令将操作数“Enable_Out”和操作数“Result”设置为数组下限。

8.6.7 UPPER_BOUND：（读取 ARRAY 上限）

表格 8-86 LOWER_BOUND 指令

LAD/FBD	SCL	说明

	`out := UPPER_BOUND(`
	`ARR:=_variant_in_,`
	`DIM:=_udint_in_);`
		在块接口中，可声明 ARRAY[*] 的变量。这些局部变量可读取 ARRAY 限值。此时，需要在 DIM 参数中指定维数。
		UPPER_BOUND（读取 ARRAY 上限）指令允许读取 ARRAY 的变量上限。

参数

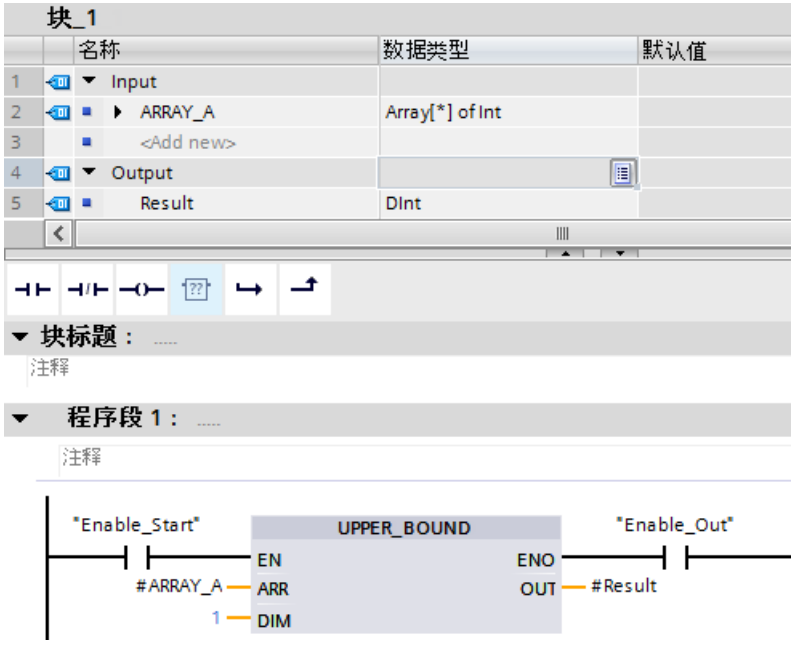
下表列出了指令“UPPER_BOUND：读取 ARRAY 上限”：

参数	声明	数据类型	存储区	说明
EN	Input	BOOL	I、Q、M、D、L	使能输入
ENO	Output	BOOL	I、Q、M、D、L	使能输出
ARR	Input	ARRAY [*]	FB：InOut 部分 FC：Input 和 InOut 部分	待读取可变上限的 ARRAY。
DIM	Input	UDINT	I、Q、M、D、L 或常数	待读取可变上限的 ARRAY 维度。
OUT	Output	DINT	I、Q、M、D、L	结果

有关有效数据类型的更多信息，请参见“数据类型 (页 119)”：

示例

在函数 (FC) 块界面中，输入参数 ARRAY_A 是具有可变维数的一维数组。



如果操作数“Enable_Start”返回信号状态“1”，则 CPU 执行指令。该指令从一维数组中读取 ARRAY #ARRAY_A 的可变上限。如果指令执行未发生错误，该指令将置位操作数“Enable_Out”和操作数“Result”。

8.6.8 读/写存储器指令

8.6.8.1 PEEK 和 POKE（仅 SCL）

SCL 提供 PEEK 和 POKE 指令，可用来从数据块、I/O 或存储器中读取内容或是向其中写入内容。而您提供操作中具体字节偏移量或位偏移量的参数。

说明

与数据块一起使用 PEEK 和 POKE 指令时，必须使用标准（未优化的）数据块。同时需要注意 PEEK 和 POKE 指令仅用于传输数据。它们无法识别地址中的数据类型。

PEEK(area:=_in_, dbNumber:=_in_, byteOffset:=_in_);	读取引用数据块、I/O 或存储区中由 byteOffset 引用的字节。 引用数据块示例： **%MB100 := PEEK(area:=16#84, dbNumber:=1, byteOffset:=#i);** 引用 IB3 输入示例： **%MB100 := PEEK(area:=16#81, dbNumber:=0, byteOffset:=#i); // when #i = 3**
PEEK_WORD(area:=_in_, dbNumber:=_in_, byteOffset:=_in_);	读取引用数据块、I/O 或存储区中由 byteOffset 引用的字。 示例： **%MW200 := PEEK_WORD(area:=16#84, dbNumber:=1, byteOffset:=#i);**
PEEK_DWORD(area:=_in_, dbNumber:=_in_, byteOffset:=_in_);	读取引用数据块、I/O 或存储区中由 byteOffset 引用的双字。 示例： **%MD300 := PEEK_DWORD(area:=16#84, dbNumber:=1, byteOffset:=#i);**
PEEK_BOOL(area:=_in_, dbNumber:=_in_, byteOffset:=_in_, bitOffset:=_in_);	读取引用数据块、I/O 或存储区中由 bitOffset 和 byteOffset 引用的布尔值。 示例： **%MB100.0 := PEEK_BOOL(area:=16#84, dbNumber:=1, byteOffset:=#ii, bitOffset:=#j);**
POKE(area:=_in_, dbNumber:=_in_, byteOffset:=_in_, value:=_in_);	向引用数据块、I/O 或存储区中引用的 byteOffset 写入值（Byte、Word 或 DWord） 引用数据块示例： **POKE(area:=16#84, dbNumber:=2, byteOffset:=3, value:="Tag_1");** 引用 QB3 输出示例： **POKE(area:=16#82, dbNumber:=0, byteOffset:=3, value:="Tag_1");**

POKE_BOOL(area:=_in_, dbNumber:=_in_, byteOffset:=_in_, bitOffset:=_in_, value:=_in_);	向引用数据块、I/O 或存储区中引用的 bitOffset 和 byteOffset 写入布尔值 示例： **POKE_BOOL(area:=16#84, dbNumber:=2, byteOffset:=3, bitOffset:=5, value:=0);**
POKE_BLK(area_src:=_in_, dbNumber_src:=_in_, byteOffset_src:=_in_, area_dest:=_in_, dbNumber_dest:=_in_, byteOffset_dest:=_in_, count:=_in_);	将引用源数据块、I/O 或存储区从引用字节偏移量开始的共“count”个字节写入引用目标数据块、I/O 或存储区中引用的 byteOffset 区域 示例： **POKE_BLK(area_src:=16#84, dbNumber_src:=#src_db, byteOffset_src:=#src_byte, area_dest:=16#84, dbNumber_dest:=#src_db, byteOffset_dest:=#src_byte, count:=10);**

对于 PEEK 和 POKE 指令，“area”、“area_src”和“area_dest”参数可以使用以下值。对于数据块以外的其它区域，dbNumber 参数必须为 0。

16#81	I
16#82	Q
16#83	M
16#84	DB

8.6.8.2 读取和写入大尾和小尾指令 (SCL)

S7-1200 CPU 提供用于以小尾格式和大尾格式读取和写入数据的 SCL 指令。小尾格式是指最低有效位所在的字节是存储器的最低地址。大尾格式是指最高有效位所在的字节是存储器的最低地址。

以小尾格式和大尾格式读取和写入数据的四个 SCL 指令如下所示：

- READ_LITTLE （以小尾格式读取数据）
- WRITE_LITTLE （以小尾格式写入数据）
- READ_BIG （以大尾格式读取数据）
- WRITE_BIG （以大尾格式写入数据）

表格 8-87　读取和写入大尾和小尾指令

LAD/FBD	SCL	说明

不提供	`READ_LITTLE(`
	`src_array:=_variant_in_,`
	`dest_Variable =>_out_,`
	`pos:=_dint_inout)`
		以小尾字节格式从存储区读取数据并写入到单个变量中。

不提供	`WRITE_LITTLE(`
	`src_variable:=_in_,`
	`dest_array =>_variant_inout_,`
	`pos:=_dint_inout)`
		以小尾字节形式将单个变量的数据写入到存储区。

不提供	`READ_BIG(`
	`src_array:=_variant_in_,`
	`dest_Variable =>_out_,`
	`pos:=_dint_inout)`
		以大尾字节格式从存储区读取数据并写入到单个变量中。

不提供	`WRITE_BIG(`
	`src_variable:=_in_,`
	`dest_array =>_variant_inout_,`
	`pos:=_dint_inout)`
		以大尾字节形式将单个变量的数据写入到存储区。

表格 8-88　READ_LITTLE and READ_BIG 指令的参数

参数	数据类型	说明
src_array	Array of Byte	欲进行数据读取的目标存储区
dest_Variable	位字符串、整数、浮点数、定时器、日期和时间、字符串	欲进行数据写入的目标变量
pos	DINT	从零开始算起，在 src_array 输入中开始读取数据的位置。

表格 8-89　WRITE_LITTLE and WRITE_BIG 指令的参数

参数	数据类型	说明
src_variable	位字符串、整数、浮点数、LDT, TOD, LTOD, DATA, Char, WChar	来自变量的源数据
dest_array	Array of Byte	数据写入的目标存储区
pos	DINT	从零开始算起，在 dest_array 输出中开始写入数据的位置。

表格 8-90　RET_VAL 参数

RET_VAL* (W#16#...)	说明
0000	无错误
80B4	SRC_ARRAY 或 DEST_ARRAY 不是 Array of Byte
8382	参数 POS 的值超出数组的限制。
8383	参数 POS 的值在数组的限制范围内，但是存储区的大小超出了数组的上限。
* 可以在程序编辑器中以整数或十六进制的形式查看错误代码。	

8.6.9　Variant 指令

8.6.9.1　VariantGet（读取 VARIANT 变量值）

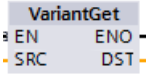
可以使用“读取 Variant 变量值”指令读取 SRC 参数的 Variant 所指向的变量，并将其写入到 DST 参数的变量中。

SRC 参数的数据类型为 Variant。除了 Variant 之外，所有数据类型都可为 DST 参数指定。

DST 参数的变量所用的数据类型必须与 Variant 所指向的数据类型相匹配。

表格 8-91　VariantGet 指令

LAD/FBD	SCL	说明

	`VariantGet(`
	`SRC:=_variant_in_,`
	`DST=>_variant_out_);`
		读取 SRC 参数所指向的变量，并将其写入到 DST 参数的变量中

说明

想要复制结构和数组，可以使用“MOVE_BLK_VARIANT：移动块”指令。

表格 8-92　VariantGet 指令的参数

参数	数据类型	说明
SRC	Variant	指向源数据的指针
DST	位字符串、整数、浮点数、定时器、日期和时间、字符串、ARRAY 元素、PLC 数据类型	将要写入数据的目标

表格 8-93　ENO 状态

ENO	条件	结果
1	无错误	指令会将 SRC 所指向的变量数据复制到 DST 变量中。
0	使能输入 EN 的信号状态为“0”，或数据类型不匹配。	指令不复制任何数据。

8.6.9.2　VariantPut（写入 VARIANT 变量值）

可以使用“写入 VARIANT 变量值”指令将 SRC 参数中变量的值写入到 VARIANT 所指向的 DST 参数的变量中。

DST 参数的数据类型为 VARIANT。除了 VARIANT 之外，所有数据类型都可为 SRC 参数指定。

SRC 参数的变量所用的数据类型必须与 VARIANT 所指向的数据类型相匹配。

表格 8-94　VariantPut 指令

LAD/FBD	SCL	说明

VariantPut
EN　ENO
SRC
DST
	`VariantPut(`
	`  SRC:=_variant_in_,`
	`  DST=>_variant_in_);`
		将 SRC 参数所引用的变量写入到 DST 参数所指向的变量中

说明

想要复制结构和数组，可以使用 “MOVE_BLK_VARIANT:移动块”指令。

表格 8-95　VariantPut 指令的参数

参数	数据类型	说明
SRC	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY elements, PLC data types	指向源数据的指针
DST	Variant	将要写入数据的目标

表格 8-96　ENO 状态

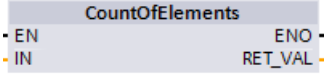
ENO	条件	结果
1	无错误	指令会将 SRC 的变量数据复制到 DST 变量中。
0	使能输入 EN 的信号状态为“0”，或数据类型不匹配。	指令不复制任何数据。

8.6.9.3　CountOfElements（获取 ARRAY 元素数目）

可以使用“获取 ARRAY 元素数目”指令来查询 Variant 指向的变量中所含有的 Array 元素数目。

如果是一维 ARRAY，指令将返回上限和下限间之差 +1。如果是多维 ARRAY，指令返回所有维度的结果。

表格 8-97　CountOfElements 指令

LAD/FBD	SCL	说明
	`Result := CountOfElements( _variant_in_);`	计算 IN 参数指向的数组中所含数组元素的数目。

说明

如果 Variant 指向 Array of Bool，指令的计数范围将包含填充元素（至最接近的字节边界）。例如，对 Array[0..1] of Bool 进行计数时，指令将返回 8。

表格 8-98　CountOfElements 指令的参数

参数	数据类型	说明
IN	Variant	待计算数组元素个数的变量
RET_VAL	UDint	指令结果

表格 8-99　ENO 状态

ENO	条件	结果
1	无错误	指令将返回数组元素的数目。
0	使用输入 EN 的信号状态为“0”或变量未指向数组。	指令返回 0。

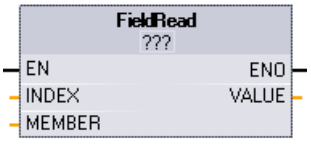
8.6.10　早期指令

8.6.10.1　FieldRead（读取域）和 FieldWrite（写入域）指令

说明

STEP 7 V10.5 不支持数组索引或多维数组形式的变量引用。 FieldRead 和 FieldWrite 指令曾用于为一维数组提供变量数组索引操作。 STEP 7 V11 和更高版本支持数组索引和多维数组形式的变量。 STEP 7 V11 和更高版本中包含了 FieldRead 和 FieldWrite，以便向后兼容使用了这些指令的程序。

表格 8-100　FieldRead 和 FieldWrite 指令

LAD/FBD	SCL	说明
	`value := member[index];`	FieldRead 用于从第一个元素由 MEMBER 参数指定的数组中读取索引值为 INDEX 的数组元素。 数组元素的值将传送到 VALUE 参数指定的位置。
	`member[index] := value;`	WriteField 用于将 VALUE 参数指定的位置上的值传送给第一个元素由 MEMBER 参数指定的数组。 该值将传送给由 INDEX 参数指定数组索引的数组元素。

[1]　对于 LAD 和 FBD： 单击“???”并从下拉菜单中选择数据类型。

表格 8-101 参数的数据类型

参数和类型		数据类型	说明
Index	输入	DInt	要读取或写入的数组元素的索引号
Member [1]	输入	二进制数、整数、浮点数、定时器、DATE、TOD 以及作为 ARRAY 变量元素的 CHAR 和 WCHAR	在全局数据块或块接口中定义的一维数组的第一个元素的位置。 例如： 如果将数组索引指定为 [-2..4]，则第一个元素的索引为 -2，而不是 0。
值 [1]	Out	二进制数、整数、浮点数、定时器、DATE、TOD、CHAR、WCHAR	将指定的数组元素复制到的位置 (FieldRead) 被复制到指定的数组元素的值的位置 (FieldWrite)

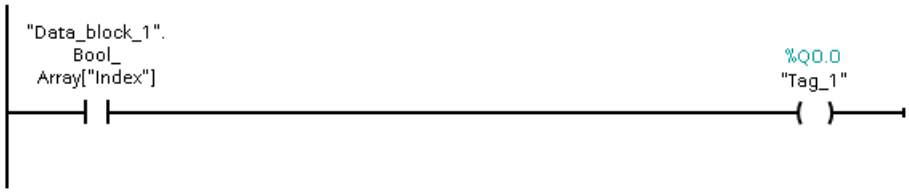
[1] MEMBER 参数和 VALUE 参数指定的数组元素的数据类型必须相同。

如果满足下列条件之一，则使能输出 ENO = 0：

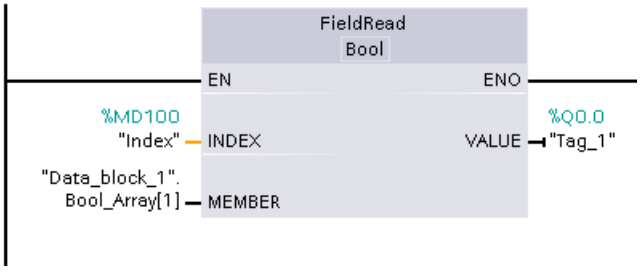
- EN 输入的信号状态为“0”
- 在 MEMBER 参数引用的数组中未定义 INDEX 参数指定的数组元素
- 处理过程中发生溢出之类的错误

示例： 通过数组索引访问数据

要通过变量访问数组中的元素，仅需在程序逻辑中将该变量用作数组索引即可。 例如，以下程序段中通过 PLC 变量“Index”引用的“Data_block_1”内布尔数组的布尔值来设置输出。



使用变量数组索引的逻辑结构与之前使用 FieldRead 指令的方法相同：



可以使用变量数组索引逻辑替换 FieldWrite 和 FieldRead 指令。

SCL 中没有 FieldRead 或 FieldWrite 指令，但支持通过变量对数组进行间接寻址：

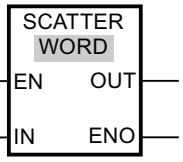
```
#Tag_1 := "Data_block_1".Bool_Array[#Index];
```

8.6.11 SCATTER

SCATTER：将位序列解析为单个位

指令“将位序列解析为单个位”用于将数据类型为 BYTE、WORD 或 DWORD 的变量解析为单个位，并保存在仅包含布尔型元素的 ARRAY of BOOL、匿名 STRUCT 或 PLC 数据类型中。

表格 8-102 SCATTER

LAD/FBD	SCL	说明
	`SCATTER (IN := #SourceWord,` `OUT => #DestinationArray};`	SCATTER：指令“将位序列解析为单个位”用于将数据类型为 BYTE、WORD 或 DWORD 的变量解析为单个位，并保存在仅包含布尔型元素的 ARRAY of BOOL、匿名 STRUCT 或 PLC 数据类型中。

说明

多维 ARRAY of BOOL

使用“将位序列解析为单个位”指令时，不支持多维 ARRAY of BOOL。

说明

ARRAY、STRUCT 或 PLC 数据类型的长度

ARRAY、匿名 STRUCT 或 PLC 数据类型中包含的元素数目必须恰好等于位序列指定的数量。例如，如果数据类型为 BYTE，则 ARRAY、STRUCT 或 PLC 数据类型中必须恰好包含 8 个元素（WORD = 16、DWORD = 32）。

说明

注意指令的可用性

该指令可用于固件版本为 V4.2 及以上版本的 S7-1200 系列 CPU 和固件版本为 V2.1 及以上版本的 S7-1500 系列 CPU。

例如，通过这种方式您可以解析状态字，并使用索引读取和改变单个位的状态。使用 GATHER，可再次将各个位组合为位序列。

如果满足下列条件之一，使能输出 ENO 将返回信号状态“0”：

- 使能输入 EN 的信号状态为“0”。
- ARRAY、STRUCT 或 PLC 数据类型中包含的 BOOL 元素数目不足。

SCATTER 指令的数据类型

下表列出了该指令的参数：

参数	声明	数据类型		存储区	说明
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I、Q、M、D、L 或常量	使能输入
ENO	Output	BOOL	BOOL	I、Q、M、D、L	使能输出
IN	Input	BYTE, WORD, DWORD	BYTE, WORD, DWORD	I、Q、M、D、L	所解析的位序列 这些值不得位于的 I/O 区域或工艺对象的 DB 内。
OUT	Output	ARRAY[*] of BOOL、STRUCT 或 PLC 数据类型 *：8、16、32 或 64 个元素	ARRAY[*] of BOOL、STRUCT 或 PLC 数据类型 *：8、16、32 或 64 个元素	I、Q、M、D、L	保存 ARRAY、STRUCT 或 PLC 数据类型中的各个位

有关有效数据类型的更多信息，请参见“另请参见”。

ARRAY 的使用示例

在块接口中创建以下变量：

变量	区域	数据类型
Enable	Input	BOOL
SourceWord		WORD

变量	区域	数据类型
EnableOut	Output	BOOL
DestinationArray		ARRAY[0..15] of BOOL

以下示例说明了该指令的工作原理：

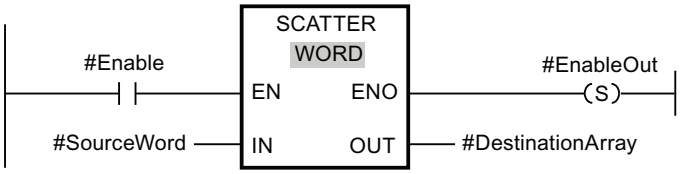


图 8-1　SCATTER 示例 2

下表将通过具体的操作数值对该指令的工作原理进行说明：

参数	操作数	数据类型
IN	SourceWord	WORD（16 位）
OUT	DestinationUDT	操作数“DestinationUDT”为 PLC 数据类型 (UDT)，其中包含 16 个元素，因此与待解析 WORD 的长度相同。

如果操作数 #Enable 在使能输入 EN 返回信号状态“1”，则将执行该指令。数据类型为 WORD 的操作数 #SourceWord 解析为单个位 (16)，并将其分配给操作数 #DestinationArray 的各个元素。如果在指令执行期间出现错误，操作数 #EnableOut 将在使能输出 ENO 处返回信号状态“0”。

有关上述示例的更多信息和程序代码，请参见“指令的示例库”。

PLC 数据类型 (UDT) 的使用示例

创建以下 PLC 数据类型“myBits”：

myBits

	Name	Data type
1	GeneralError	Bool
2	DeviceError	Bool
3	CommError	Bool
4	myError1	Bool
5	myError2	Bool
6	myError3	Bool
7	myError4	Bool
8	myError5	Bool
9	myError6	Bool
10	myError7	Bool
11	myError8	Bool
12	myError9	Bool
13	myError10	Bool
14	myError11	Bool
15	myError12	Bool
16	myError13	Bool

在块接口中创建以下变量：

变量	区域	数据类型
Enable	Input	BOOL
SourceWord		WORD
EnableOut	Output	BOOL
DestinationUDT		“myBits”

以下示例说明了该指令的工作原理：

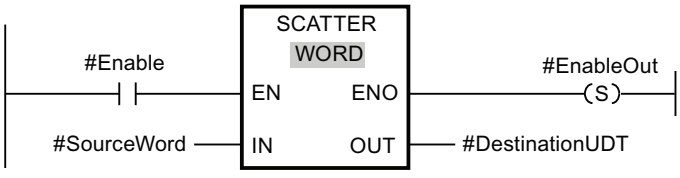


图 8-2　SCATTER_Bsp_example

下表将通过具体的操作数值对该指令的工作原理进行说明：

参数	操作数	数据类型
IN	SourceWord	WORD（16 位）
OUT	DestinationUDT	操作数“DestinationUDT”为 PLC 数据类型 (UDT)，其中包含 16 个元素，因此与待解析 WORD 的长度相同。

如果操作数 #Enable 在使能输入 EN 返回信号状态“1”，则将执行该指令。数据类型为 WORD 的操作数 #SourceWord 解析为单个位 (16)，并将其分配给操作数 #DestinationArray 的各个元素。如果在指令执行期间出现错误，操作数 #EnableOut 将在使能输出 ENO 处返回信号状态“0”。

参见

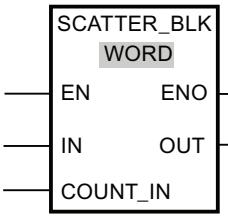
新功能 (页 37)

8.6.12 SCATTER_BLK

SCATTER_BLK：将位序列 ARRAY 的元素解析为单个位

“将位序列 ARRAY 的元素解析为单个位”指令用于将 ARRAY of BYTE、ARRAY of WORD 或 ARRAY of DWORD 的一个或多个元素解析为单个位，并保存在仅包含布尔型元素的 ARRAY of BOOL、匿名 STRUCT 或 PLC 数据类型中。在 COUNT_IN 参数中，可指定待解析源 ARRAY 中的元素数目。IN 参数中源 ARRAY 的元素数量可能多于 COUNT_IN 参数中的指定数量。要保存所解析位序列的各个位，ARRAY of BOOL、匿名 STRUCT 或 PLC 数据类型中必须包含足够的元素数量。但这可能会增加目标存储区空间。

表格 8-103 SCATTER_BLK

LAD/FBD	SCL	说明
	**SCATTER_BLK(IN:=_byte_in_, COUNT_IN:=_uint_in_, OUT=>_bool_out_);_IN:=_uint_in_,**	“将位序列 ARRAY 的元素解析为单个位”指令用于将 ARRAY of BYTE、ARRAY of WORD 或 ARRAY of DWORD 的一个或多个元素解析为单个位，并保存在仅包含布尔型元素的 ARRAY of BOOL、匿名 STRUCT 或 PLC 数据类型中。在 COUNT_IN 参数中，可指定待解析源 ARRAY 中的元素数目。IN 参数中源 ARRAY 的元素数量可能多于 COUNT_IN 参数中的指定数量。要保存所解析位序列的各个位，ARRAY of BOOL、匿名 STRUCT 或 PLC 数据类型中必须包含足够的元素数量。但这可能会增加目标存储区空间。

说明

ENO 为 false 时不写入任何数据

在 S7-1200 CPU 中，仅对于 SCATTER_BLK 指令，如果 ENO 为 FALSE，则不会将任何数据写入输出。

说明

多维 ARRAY of BOOL

如果 ARRAY 是一个多维 ARRAY of BOOL，即使未显式声明，也将对所包含维度的填充位进行计数。

示例 1：ARRAY[1..10,0..4,1..2] of BOOL 的处理方式与 ARRAY[1..10,0..4,1..8] of BOOL 或 ARRAY[0..399] of BOOL 类似。

示例 2：在 IN 参数处，将互连一个 ARRAY[0..5] of WORD (sourceArrayWord[2])。参数 COUNT_IN 的值为“3”。在 OUT 参数处，将互连一个 ARRAY[0..1,0..5,0..7] of BOOL (destinationArrayBool[0,0,0])。参数 IN 和参数 OUT 中两个数组的大小均为 96 位。ARRAY of WORD 将解析为 48 个单个位

说明

如果目标 ARRAY 的 ARRAY 下限不是“0”，请注意以下几点：

出于性能原因，索引必须始终以 BYTE、WORD 或 DWORD 限值开始。这就意味着索引必须从 ARRAY 的下限处开始计算。通过以下公式进行相应计算：

有效下标 = ARRAY 下限 + n（位序列数量）× 所需位序列位数

对于 ARRAY[-2..45] of BOOL 和位序列 WORD，计算如下所示：

- 有效索引 (-2) = -2 + 0 × 16
- 有效索引 (14) = -2 + 1 × 16
- 有效索引 (30) = -2 + 2 × 16

相关示例，请参见如下内容。

说明

该指令的可用性

该指令可用于固件版本为 V4.2 及以上版本的 S7-1200 系列 CPU 和固件版本为 V2.1 及以上版本的 S7-1500 系列 CPU。

例如，通过这种方式您可以解析状态字，并使用索引读取和改变单个位的状态。使用 GATHER，可再次将各个位组合为位序列。

如果满足下列条件之一，使能输出 ENO 将返回信号状态“0”：

- 使能输入 EN 的信号状态为“0”。
- 源 ARRAY 中的元素数量少于 COUNT_IN 参数中的指定数量。

- 目标 ARRAY 的索引不以 BYTE、WORD 或 DWORD 限值开始。在这种情况中，将不向 ARRAY of BOOL 中写入任何结果。
- ARRAY[*] of BOOL、STRUCT 或 PLC 数据类型未提供所需的元素数量。
  - S7-1500-CPU：此时，将尽可能多地解析位序列并写入 ARRAY of BOOL、匿名 STRUCT 或 PLC 数据类型中。其余位序列将不考虑在内。
  - S7-1200-CPU：不含复制过程。

SCATTER_BLK 指令的数据类型

下表列出了该指令的参数：

参数	声明	数据类型		存储区	说明
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I、Q、M、D、L 或常量	使能输入
ENO	Output	BOOL	BOOL	I、Q、M、D、L	使能输出
IN	Input	ARRAY[*] of <位序列> 的元素	ARRAY[*] of <位序列> 的元素	I、Q、M、D、L	所解析的 ARRAY of <位序列> 这些值不得位于的 I/O 区域或工艺对象的 DB 内。

参数	声明	数据类型		存储区	说明
		S7-1200	S7-1500		
COUNT_IN	Input	USINT, UINT, UDINT	USINT, UINT, UDINT	I、Q、M、D、L	对待解析源 ARRAY 中的元素数量进行计数。 该值不得位于的 I/O 区域或工艺对象的数据库内。
OUT	Output	ARRAY[*] of BOOL、STRUCT 或 PLC 数据类型的元素	ARRAY[*] of BOOL、STRUCT 或 PLC 数据类型的元素	I、Q、M、D、L	保存 ARRAY、STRUCT 或 PLC 数据类型中的各个位

可以从指令框的“???”下拉列表中选择所需位序列。

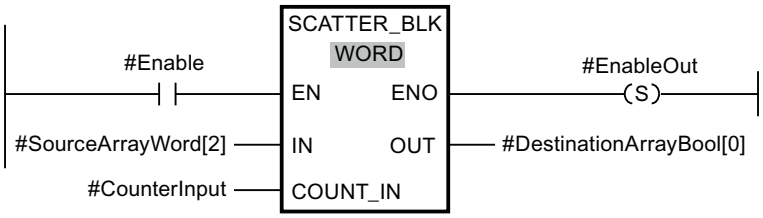
有关有效数据类型的更多信息，请参见“另请参见”。

下限为“0”的目标 ARRAY 示例

在块接口中创建以下变量：

变量	区域	数据类型
Enable	Input	BOOL
SourceArrayWord		ARRAY[0..5] of WORD
CounterInput		UDINT
EnableOut	Output	BOOL
DestinationArrayBool		ARRAY[0..95] of BOOL

以下示例说明了该指令的工作原理：



下表将通过具体的操作数值对该指令的工作原理进行说明：

参数	操作数	数据类型
IN	SourceArrayWord[2]	ARRAY[0..5] of WORD（可以解析 96 位。）
COUNT_IN	CounterInput = 3	UDINT3（将解析 3 个字或 48 位。这意味着目标 ARRAY 中至少需包含 48 个位。）
OUT	DestinationArrayBool[0]	操作数 “DestinationArrayBool”的数据类型为 ARRAY[0..95] of BOOL。这表示它可以提供 96 个 BOOL 元素。

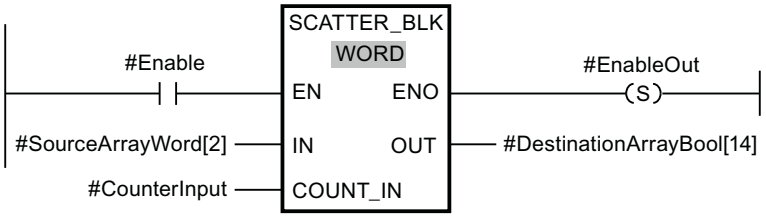
如果操作数 #Enable 在使能输入 EN 返回信号状态“1”，则将执行该指令。从分配给操作数 #DestinationArrayBool 各元素的第 1 个元素开始，将操作数 #SourceArrayWord 的第 3、第 4 和第 5 个 WORD 解析为单个的位 (48)。如果在指令执行期间出现错误，操作数 #EnableOut 将在使能输出 ENO 处返回信号状态“0”。

下限为“-2”的目标 ARRAY 示例

在块接口中创建以下变量：

变量	区域	数据类型
Enable	Input	BOOL
SourceArrayWord		ARRAY[0..5] of WORD
CounterInput		UDINT
EnableOut	Output	BOOL
DestinationArrayBool		ARRAY[-2..93] of BOOL

以下示例说明了该指令的工作原理：



下表将通过具体的操作数值对该指令的工作原理进行说明：

参数	操作数	数据类型
IN	SourceArrayWord[2]	ARRAY[0..5] of WORD（可以解析 96 位。）
COUNT_IN	CounterInput = 3	UDINT3（将解析 3 个字或 48 位。这意味着目标 ARRAY 中至少需包含 48 个位。）
OUT	DestinationArrayBool[14]	操作数“DestinationArrayBool”的数据类型为 ARRAY[-2..93] of BOOL。这表示它可以提供 96 个 BOOL 元素。

如果操作数 #Enable 在使能输入 EN 返回信号状态“1”，则将执行该指令。从分配给操作数 #DestinationArrayBool 的各元素的第 16 个元素开始，将操作数 #SourceArrayWord 的第 3、第 4 和第 5 个字解析为单个位 (48)。如果在指令执行期间出现错误，操作数 #EnableOut 将在使能输出 ENO 处返回信号状态“0”。其余 32 位不写入。

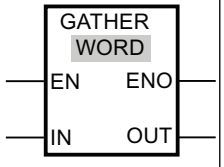
有关上述示例的更多信息和程序代码，请参见“指令的示例库”。

8.6.13 GATHER

GATHER

指令“将各个位组合为位序列”用于将仅包含布尔型元素的 ARRAY of BOOL、匿名 STRUCT 或 PLC 数据类型中的各个位组合为一个位序列。位序列保存在数据类型为 BYTE、WORD、DWORD 或 LWORD 的变量中。

表格 8-104 GATHER

LAD/FBD	SCL	说明
	`GATHER(IN := #SourceArray, OUT => #DestinationArray};`	GATHER：指令“将各个位组合为位序列”用于将仅包含布尔型元素的 ARRAY of BOOL、匿名 STRUCT 或 PLC 数据类型中的各个位组合为一个位序列。位序列保存在数据类型为 BYTE、WORD 或 DWORD 的变量中。

说明

多维 ARRAY of BOOL

使用“将各个位组合为位序列”指令时，不支持多维 ARRAY of BOOL。

说明

ARRAY、STRUCT 或 PLC 数据类型的长度

ARRAY、STRUCT 或 PLC 数据类型中包含的元素数目必须恰好等于位序列指定的数量。

例如，如果数据类型为 BYTE，则 ARRAY、匿名 STRUCT 或 PLC 数据类型中必须恰好包含 8 个元素（WORD = 16、DWORD = 32）。

说明

该指令的可用性

该指令可用于固件版本为 V4.2 及以上版本的 S7-1200 系列 CPU 和固件版本为 V2.1 及以上版本的 S7-1500 系列 CPU。

如果满足下列条件之一，使能输出 ENO 将返回信号状态“0”：

- 使能输入 EN 的信号状态为“0”。
- ARRAY、匿名 STRUCT 或 PLC 数据类型 (UDT) 中 BOOL 元素的数量少于或多于位序列所指定的数量。此时，系统不传送 BOOL 元素。
- 可用的元素数少于所需的位数量。

GATHER 指令的数据类型

下表列出了该指令的参数：

参数	声明	数据类型		存储区	说明
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I、Q、M、D、L 或常量	使能输入
ENO	Output	BOOL	BOOL	I、Q、M、D、L	使能输出

参数	声明	数据类型		存储区	说明
		S7-1200	S7-1500		
IN	Input	ARRAY[*] of BOOL、STRUCT 或 PLC 数据类型 *：8、16、32 或 64 个元素	ARRAY[*] of BOOL、STRUCT 或 PLC 数据类型 *：8、16、32 或 64 个元素	I、Q、M、D、L	ARRAY、STRUCT 或 PLC 数据类型，这些数据类型中的各个位将组合为一个位序列。 这些值不得位于的 I/O 区域或工艺对象的 DB 内。
OUT	Output	BYTE, WORD, DWORD	BYTE, WORD, DWORD	I、Q、M、D、L	组合后的位序列，保存在一个变量中

可以从指令框的“???”下拉列表中选择所需位序列。

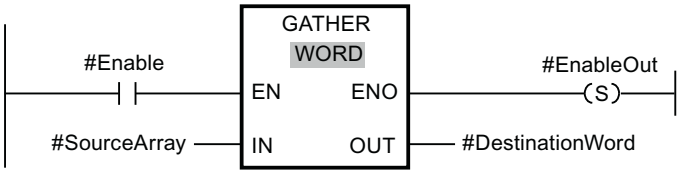
有关有效数据类型的更多信息，请参见“另请参见”。

ARRAY 的使用示例

在块接口中创建以下变量：

变量	区域	数据类型
Enable	Input	BOOL
SourceArray		ARRAY[0..15] of BOOL
EnableOut	Output	BOOL
DestinationWord		WORD

以下示例说明了该指令的工作原理：



下表将通过具体的操作数值对该指令的工作原理进行说明：

参数	操作数	数据类型
IN	SourceArray	操作数“SourceArray”的数据类型为 ARRAY[0..15] of BOOL。其中包含 16 个元素，因此与需要将位进行组合的字的长度相当。
OUT	DestinationWord	WORD（16 位）

如果操作数 #Enable 在使能输入 EN 返回信号状态“1”，则将执行该指令。操作数 #SourceArray 的各个位将组合为一个 WORD。如果在指令执行期间出现错误，操作数 #EnableOut 将在使能输出 ENO 处返回信号状态“0”。

有关上述示例的更多信息和程序代码，请参见“指令的示例库”。

PLC 数据类型 (UDT) 的使用示例

创建以下 PLC 数据类型“myBits”：

myBits

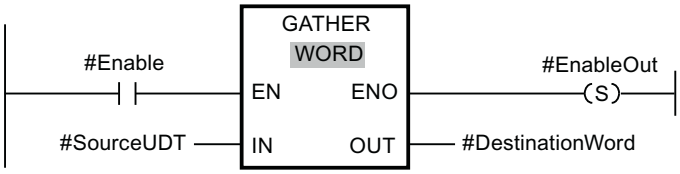
	Name	Data type
1	GeneralError	Bool
2	DeviceError	Bool
3	CommError	Bool
4	myError1	Bool
5	myError2	Bool
6	myError3	Bool
7	myError4	Bool
8	myError5	Bool
9	myError6	Bool
10	myError7	Bool
11	myError8	Bool
12	myError9	Bool
13	myError10	Bool
14	myError11	Bool
15	myError12	Bool
16	myError13	Bool

在块接口中创建以下变量：

变量	区域	数据类型
Enable	Input	BOOL
SourceUDT		“myBits”

变量	区域	数据类型
EnableOut	Output	BOOL
DestinationWord		WORD

以下示例说明了该指令的工作原理：



下表将通过具体的操作数值对该指令的工作原理进行说明：

参数	操作数	数据类型
IN	SourceUDT	操作数“SourceUDT”为 PLC 数据类型 (UDT)，其中包含 16 个元素，因此与需要将位进行组合的字的长度相当。
OUT	DestinationWord	WORD（16 位）

如果操作数 #Enable 在使能输入 EN 返回信号状态“1”，则将执行该指令。操作数 #SourceUDT 的各个位将组合为一个 WORD。如果在指令执行期间出现错误，操作数 #EnableOut 将在使能输出 ENO 处返回信号状态“0”。

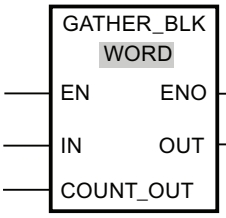
8.6.14 GATHER_BLK

说明

指令“将各个位组合为 ARRAY of <位序列> 的多个元素”用于将仅包含布尔型元素的 ARRAY of BOOL、匿名 STRUCT 或 PLC 数据类型中的各个位组合为 ARRAY of <位序列> 中的一个或多个元素。您可以在 COUNT_OUT 参数中指定要写入的目标 ARRAY 元素数量。在此步中，还可隐式指定 ARRAY of BOOL、匿名 STRUCT 或 PLC 数据类型中所需的元素数量。OUT 参数中目标 ARRAY 的元素数量可能多于 COUNT_OUT 参数中的指定数量。要保存待组合的各个位，ARRAY of <位序列> 中必须包含足够的元素数目。但目标 ARRAY 可能更大。

表格 8-105 GATHER

LAD/FBD	SCL	说明

	`GATHER_BLK(IN := #SourceArrayBool[0],`
	`COUNT_OUT := #CounterOutput,`
	`OUT => #DestinationArrayWord[2] );`
		指令“将各个位组合为 ARRAY of <位序列> 的多个元素”用于将仅包含布尔型元素的 ARRAY of BOOL、匿名 STRUCT 或 PLC 数据类型中的各个位组合为 ARRAY of <位序列> 中的一个或多个元素。您可以在 COUNT_OUT 参数中指定要写入的目标 ARRAY 元素数量。在此步中，还可隐式指定 ARRAY of BOOL、匿名 STRUCT 或 PLC 数据类型中所需的元素数量。OUT 参数中目标 ARRAY 的元素数量可能多于 COUNT_OUT 参数中的指定数量。要保存待组合的各个位，ARRAY of <位序列> 中必须包含足够的元素数目。但目标 ARRAY 可能更大。

说明

ENO 为 false 时不写入任何数据

在 S7-1200 CPU 中，仅对于 GATHER_BLK 指令，如果 ENO 为 FALSE，则不会将任何数据写入输出。

说明

多维 ARRAY of BOOL

如果 ARRAY 是一个多维 ARRAY of BOOL，即使未显式声明，也将对所包含维度的填充位进行计数。

示例 1：ARRAY[1..10,0..4,1..2] of BOOL 的处理方式与 ARRAY[1..10,0..4,1..8] of BOOL 或 ARRAY[0..399] of BOOL 类似。

示例 2：在 OUT 参数处，将互连一个 ARRAY[0..5] of WORD (sourceArrayWord[2])。参数 COUNT_IN 的值为“3”。在 IN 参数处，将互连一个 ARRAY[0..1,0..5,0..7] of BOOL (destinationArrayBool[0,0,0])。参数 IN 和参数 OUT 中两个数组的大小均为 96 位。48 个单个位将组合为 ARRAY of BOOL。

说明

如果源 ARRAY 的 ARRAY 下限不是“0”，请注意以下几点：

出于性能原因，索引必须始终以 BYTE、WORD 或 DWORD 限值开始。这就意味着索引必须从 ARRAY 的下限处开始计算。通过以下公式进行相应计算：

有效下标 = ARRAY 下限 + n（位序列数量）× 所需位序列位数

对于 ARRAY[-2..45] of BOOL 和位序列 WORD，计算如下所示：

- 有效索引 (-2) = -2 + 0 × 16
- 有效索引 (14) = -2 + 1 × 16
- 有效索引 (30) = -2 + 2 × 16

相关示例，请参见如下内容。

说明

该指令的可用性

该指令可用于固件版本为 V4.2 及以上版本的 S7-1200 系列 CPU 和固件版本为 V2.1 及以上版本的 S7-1500 系列 CPU。

如果满足下列条件之一，使能输出 ENO 将返回信号状态“0”：

- 使能输入 EN 的信号状态为“0”。
- 源 ARRAY 的索引不以 BYTE、WORD 或 DWORD 限值开始。此时，不会向 ARRAY of <位序列> 中写入任何结果。
- ARRAY[*] of <位序列> 中未提供所需的元素数量。
  - S7-1500-CPU：此时，将尽可能多地组合位序列并写入 ARRAY of <位序列> 中。不再考虑剩余的位。
  - S7-1200-CPU：不含复制过程。

GATHER_BLK 指令的数据类型

下表列出了该指令的参数：

参数	声明	数据类型		存储区	说明
		S7-1200	S7-1500		
EN	Input	BOOL	BOOL	I、Q、M、D、L 或常量	使能输入
ENO	Output	BOOL	BOOL	I、Q、M、D、L	使能输出
IN	Input	ARRAY[*] of BOOL、STRUCT 或 PLC 数据类型的元素	ARRAY[*] of BOOL、STRUCT 或 PLC 数据类型的元素	I、Q、M、D、L	ARRAY of BOOL、STRUCT 或 PLC 数据类型，其中各个位待组合（源 ARRAY） 这些值不得位于的 I/O 区域或工艺对象的 DB 内。
COUNT_OUT	Input	USINT, UINT, UDINT	USINT, UINT, UDINT	I、Q、M、D、L	计数目标 ARRAY 的元素数量。 该值不得位于的 I/O 区域或工艺对象的数据库内。
OUT	Output	ARRAY[*] of <位序列> 的元素	ARRAY[*] of <位序列> 的元素	I、Q、M、D、L	保存各个位的 ARRAY of <位序列>（目标 ARRAY）

可以从指令框的“???”下拉列表中选择所需位序列。

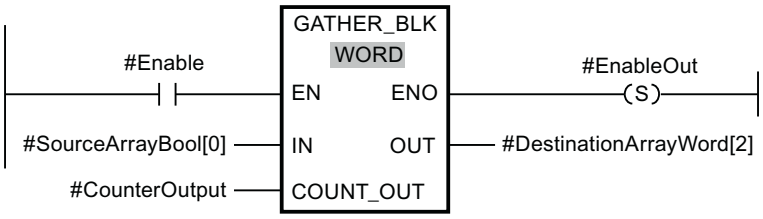
有关有效数据类型的更多信息，请参见“另请参见”。

下限为“0”的源 ARRAY 示例

在块接口中创建以下变量：

变量	区域	数据类型
Enable	Input	BOOL
SourceArrayBool		ARRAY[0..95] of BOOL
CounterOutput		UDINT
EnableOut	Output	BOOL
DestinationArrayWord		ARRAY[0..5] of WORD

以下示例说明了该指令的工作原理：



下表将通过具体的操作数值对该指令的工作原理进行说明：

参数	操作数	数据类型
IN	SourceArrayBool[0]	操作数“SourceArrayBool”的数据类型为 ARRAY[0..95] of BOOL。这意味着，它可提供 96 个 BOOL 元素，可再次组合成字。
COUNT_OUT	CounterOutput = 3	UDINT3（要写入 3 个字。这意味着源 ARRAY 中必须包含 48 个位。）
OUT	DestinationArrayWord[2]	操作数“DestinationArrayWord”的数据类型为 ARRAY[0..5] of WORD。这意味着包含 6 个 WORD 元素。

如果操作数 #Enable 在使能输入 EN 返回信号状态“1”，则将执行该指令。从操作数 #SourceArrayBool 的第 1 个元素开始，将 48 个位组合到操作数 #DestinationArrayWord 中。目标 ARRAY 中的起点是第三个元素。这意味着前 16 位将写入目标 ARRAY 的第三个字，第

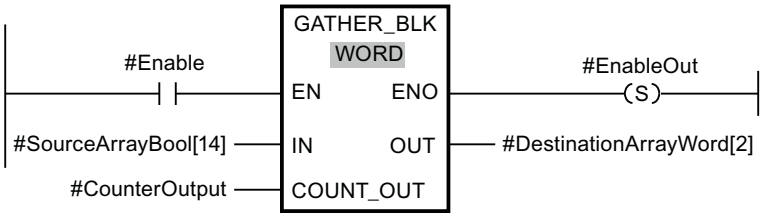
二组 16 位写入第四个字，第三组 16 位写入第五个字。如果在指令执行期间出现错误，操作数 #EnableOut 将在使能输出 ENO 处返回信号状态“0”。

下限为“-2”的源 ARRAY 示例

在块接口中创建以下变量：

变量	区域	数据类型
Enable	Input	BOOL
SourceArrayBool		ARRAY[-2..93] of BOOL
CounterOutput		UDINT
EnableOut	Output	BOOL
DestinationArrayWord		ARRAY[0..5] of WORD

以下示例说明了该指令的工作原理：



下表将通过具体的操作数值对该指令的工作原理进行说明：

参数	操作数	数据类型
IN	SourceArrayBool[14]	操作数“SourceArrayBool”的数据类型为 ARRAY[-2..93] of BOOL。由于起点是第 16 个元素，因此仅 80 个 BOOL 元素可再次组合到字中。
COUNT_OUT	CounterOutput = 3	UDINT3（要写入 3 个字。这意味着源 ARRAY 中必须包含 48 个位。）
OUT	DestinationArrayWord[2]	操作数“DestinationArrayWord”的数据类型为 ARRAY[0..5] of WORD。这意味着包含 6 个 WORD 元素。

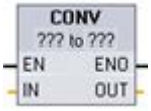
如果操作数 #Enable 在使能输入 EN 返回信号状态“1”，则将执行该指令。从操作数 #SourceArrayBool 的第 16 个元素开始，将 48 个位组合到操作数 #DestinationArrayWord 中。目标 ARRAY 中的起点是第三个元素。这意味着将忽略源 ARRAY 的前 16 位。第二组 16 位将写入目标 ARRAY 的第三个字，第三组 16 位写入第四个字，第四组 16 位写入第五个字。不考虑源 ARRAY 剩余的 64 个位。如果在指令执行期间出现错误，操作数 #EnableOut 将在使能输出 ENO 处返回信号状态“0”。

有关上述示例的更多信息和程序代码，请参见“指令的示例库”。

8.7 转换操作

8.7.1 CONV（转换值）

表格 8-106 转换 (CONV) 指令

LAD/FBD	SCL	说明
	`out := <data type in>_TO_<data type out>(in);`	将数据元素从一种数据类型转换为另一种数据类型。

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

[2] 对于 SCL：通过识别输入参数 (in) 和输出参数 (out) 的数据类型来构造转换指令。例如，DWORD_TO_REAL 将 DWord 值转换为 Real 值。

表格 8-107 参数的数据类型

参数	数据类型	说明
IN	位串 [1]，SInt, USInt, Int, UInt, DInt, UDInt, Real, LReal, BCD16, BCD32, Char, WChar	输入值
OUT	位串 [1]，SInt, USInt, Int, UInt, DInt, UDInt, Real, LReal, BCD16, BCD32, Char, WChar	转换为新数据类型的输入值

[1] 该指令不允许您选择位串（Byte、Word、DWord）。要为指令参数输入数据类型 Byte、Word 或 DWord 的操作数，选择位长度相同的无符号整型。例如 为 Byte 选择 USInt、为 Word 选择 UInt 或为 DWord 选择 UDInt。

选择（转换源）数据类型之后，（转换目标）下拉列表中将显示可能的转换项列表。与 BCD16 进行转换仅限于 Int 数据类型。与 BCD32 进行转换仅限于 DInt 数据类型。

表格 8-108 ENO 状态

ENO	说明	结果 (OUT)
1	无错误	有效结果
0	IN 为 +/- INF 或 +/- NaN	+/- INF 或 +/- NaN
0	结果超出 OUT 数据类型的有效范围	OUT 设置为 IN 值

8.7.2 SCL 的转换指令

SCL 的转换指令

表格 8-109 从 Bool、Byte、Word 或 DWord 进行转换

数据类型	指令	结果
Bool	**BOOL_TO_BYTE, BOOL_TO_WORD, BOOL_TO_DWORD, BOOL_TO_INT, BOOL_TO_DINT**	值被传送到目标数据类型的最低有效位。
Byte	**BYTE_TO_BOOL**	最低有效位被传送到目标数据类型。
	BYTE_TO_WORD, BYTE_TO_DWORD	值被传送到目标数据类型的最低有效字节。
	BYTE_TO_SINT, BYTE_TO_USINT	值被传送到目标数据类型。
	BYTE_TO_INT, BYTE_TO_UINT, BYTE_TO_DINT, BYTE_TO_UDINT	值被传送到目标数据类型的最低有效字节。
Word	**WORD_TO_BOOL**	最低有效位被传送到目标数据类型。
	WORD_TO_BYTE	源值的最低有效字节被传送到目标数据类型
	WORD_TO_DWORD	值被传送到目标数据类型的最低有效字。
	WORD_TO_SINT, WORD_TO_USINT	源值的最低有效字节被传送到目标数据类型。
	WORD_TO_INT, WORD_TO_UINT	值被传送到目标数据类型。
	WORD_TO_DINT, WORD_TO_UDINT	值被传送到目标数据类型的最低有效字。

数据类型	指令	结果
DWord	DWORD_TO_BOOL	最低有效位被传送到目标数据类型。
	DWORD_TO_BYTE, DWORD_TO_WORD, DWORD_TO_SINT	源值的最低有效字节被传送到目标数据类型。
	DWORD_TO_USINT, DWORD_TO_INT, DWORD_TO_UINT	源值的最低有效字被传送到目标数据类型。
	DWORD_TO_DINT, DWORD_TO_UDINT, DWORD_TO_REAL	值被传送到目标数据类型。

表格 8-110 从短整型（SInt 或 USInt）进行转换

数据类型	指令	结果
SInt	SINT_TO_BOOL	最低有效位被传送到目标数据类型。
	SINT_TO_BYTE	值被传送到目标数据类型
	SINT_TO_WORD, SINT_TO_DWORD	值被传送到目标数据类型的最低有效字节。
	SINT_TO_INT, SINT_TO_DINT, SINT_TO_USINT, SINT_TO_UINT, SINT_TO_UDINT, SINT_TO_REAL, SINT_TO_LREAL, SINT_TO_CHAR, SINT_TO_STRING	值被转换。
USInt	USINT_TO_BOOL	最低有效位被传送到目标数据类型。
	USINT_TO_BYTE	值被传送到目标数据类型
	USINT_TO_WORD, USINT_TO_DWORD, USINT_TO_INT, USINT_TO_UINT, USINT_TO_DINT, USINT_TO_UDINT	值被传送到目标数据类型的最低有效字节。
	USINT_TO_SINT, USINT_TO_REAL, USINT_TO_LREAL, USINT_TO_CHAR, USINT_TO_STRING	值被转换。

表格 8-111 从整型（Int 或 UInt）进行转换

数据类型	指令	结果
Int	INT_TO_BOOL	最低有效位被传送到目标数据类型。
	INT_TO_BYTE, INT_TO_DWORD, INT_TO_SINT, INT_TO_USINT, INT_TO_UINT, INT_TO_UDINT, INT_TO_REAL, INT_TO_LREAL, INT_TO_CHAR, INT_TO_STRING	值被转换。
	INT_TO_WORD	值被传送到目标数据类型。
	INT_TO_DINT	值被传送到目标数据类型的最低有效字节。

数据类型	指令	结果
UInt	UINT_TO_BOOL	最低有效位被传送到目标数据类型。
	UINT_TO_BYTE, UINT_TO_SINT, UINT_TO_USINT, UINT_TO_INT, UINT_TO_REAL, UINT_TO_LREAL, UINT_TO_CHAR, UINT_TO_STRING	值被转换。
	UINT_TO_WORD, UINT_TO_DATE	值被传送到目标数据类型。
	UINT_TO_DWORD, UINT_TO_DINT, UINT_TO_UDINT	值被传送到目标数据类型的最低有效字节。

表格 8-112 从双整型（Dint 或 UDInt）进行转换

数据类型	指令	结果
DInt	DINT_TO_BOOL	最低有效位被传送到目标数据类型。
	DINT_TO_BYTE, DINT_TO_WORD, DINT_TO_SINT, DINT_TO_USINT, DINT_TO_INT, DINT_TO_UINT, DINT_TO_UDINT, DINT_TO_REAL, DINT_TO_LREAL, DINT_TO_CHAR, DINT_TO_STRING	值被转换。
	DINT_TO_DWORD, DINT_TO_TIME	值被传送到目标数据类型。
UDInt	UDINT_TO_BOOL	最低有效位被传送到目标数据类型。
	UDINT_TO_BYTE, UDINT_TO_WORD, UDINT_TO_SINT, UDINT_TO_USINT, UDINT_TO_INT, UDINT_TO_UINT, UDINT_TO_DINT, UDINT_TO_REAL, UDINT_TO_LREAL, UDINT_TO_CHAR, UDINT_TO_STRING	值被转换。
	UDINT_TO_DWORD, UDINT_TO_TOD	值被传送到目标数据类型。

表格 8-113 从实数（Real 或 LReal）进行转换

数据类型	指令	结果
Real	REAL_TO_DWORD, REAL_TO_LREAL	值被传送到目标数据类型。
	REAL_TO_SINT, REAL_TO_USINT, REAL_TO_INT, REAL_TO_UINT, REAL_TO_DINT, REAL_TO_UDINT, REAL_TO_STRING	值被转换。
LReal	LREAL_TO_SINT, LREAL_TO_USINT, LREAL_TO_INT, LREAL_TO_UINT, LREAL_TO_DINT, LREAL_TO_UDINT, LREAL_TO_REAL, LREAL_TO_STRING	值被转换。

表格 8-114 从 Time、DTL、TOD 或 Date 进行转换

数据类型	指令	结果
Time	TIME_TO_DINT	值被传送到目标数据类型。
DTL	DTL_TO_DATE, DTL_TO_TOD	值被转换。
TOD	TOD_TO_UDINT	值被转换。
Date	DATE_TO_UINT	值被转换。

表格 8-115 从 Char 或 String 进行转换

数据类型	指令	结果
Char	CHAR_TO_SINT, CHAR_TO_USINT, CHAR_TO_INT, CHAR_TO_UINT, CHAR_TO_DINT, CHAR_TO_UDINT	值被转换。
	CHAR_TO_STRING	值被传送到字符串的第一个字符。
String	STRING_TO_SINT, STRING_TO_USINT, STRING_TO_INT, STRING_TO_UINT, STRING_TO_DINT, STRING_TO_UDINT, STRING_TO_REAL, STRING_TO_LREAL	值被转换。
	STRING_TO_CHAR	字符串的第一个字符被复制到 Char。

8.7.3 ROUND（取整）和 TRUNC（截尾取整）

表格 8-116 ROUND 和 TRUNC 指令

LAD/FBD	SCL	说明
	`out := ROUND (in);`	将实数转换为整数。对于 LAD/FBD，在指令框中单击“???”选择输出数据类型，例如“DInt”。 对于 SCL，ROUND 指令的默认输出数据类型为 DINT。要舍入为另一种输出数据类型，输入具有数据类型的显式名称的指令名称，例如：ROUND_REAL 或 ROUND_LREAL。 实数的小数部分舍入为最接近的整数值（IEEE - 取整为最接近值）。如果该数值刚好是两个连续整数的一半（例如，10.5），则将其取整为偶数。例如： • ROUND (10.5) = 10 • ROUND (11.5) = 12
	`out := TRUNC(in);`	TRUNC 用于将实数转换为整数。实数的小数部分被截成零（IEEE - 取整为零）。

[1] 对于 LAD 和 FBD：单击“???”（按指令名称）并从下拉菜单中选择数据类型。

表格 8-117 参数的数据类型

参数	数据类型	说明
IN	Real，LReal	浮点型输入
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	取整或截取后的输出

表格 8-118 ENO 状态

ENO	说明	结果 (OUT)
1	无错误	有效结果
0	IN 为 +/- INF 或 +/- NaN	+/- INF 或 +/- NaN

8.7.4 CEIL 和 FLOOR（浮点数向上和向下取整）

表格 8-119 CEIL 和 FLOOR 指令

LAD/FBD	SCL	说明
	`out := CEIL(in);`	将实数（Real 或 LReal）转换为大于或等于所选实数的最小整数（IEEE“向正无穷取整”）。
	`out := FLOOR(in);`	将实数（Real 或 LReal）转换为小于或等于所选实数的最大整数（IEEE“向负无穷取整”）。

[1] 对于 LAD 和 FBD：单击“???”（按指令名称）并从下拉菜单中选择数据类型。

表格 8-120 参数的数据类型

参数	数据类型	说明
IN	Real，LReal	浮点型输入
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	转换后的输出

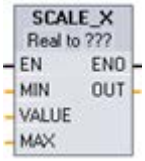
表格 8-121 ENO 状态

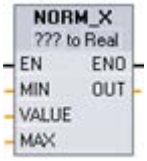
ENO	说明	结果 (OUT)
1	无错误	有效结果
0	IN 为 +/- INF 或 +/- NaN	+/- INF 或 +/- NaN

8.7.5 SCALE_X（标定）和 NORM_X（标准化）

表格 8-122 SCALE_X 和 NORM_X 指令

LAD/FBD	SCL	说明

	`out :=SCALE_X(min:=_in_,`
	`value:=_in_,`
	`max:=_in_);`
		按参数 MIN 和 MAX 所指定的数据类型和值范围对标准化的实参数 VALUE（其中，0.0 <= VALUE <= 1.0）进行标定：
		OUT = VALUE (MAX - MIN) + MIN

	`out :=NORM_X(min:=_in_,`
	`value:=_in_,`
	`max:=_in_);`
		标准化通过参数 MIN 和 MAX 指定的值范围内的参数 VALUE：
		OUT = (VALUE - MIN) / (MAX - MIN)，其中 (0.0 <= OUT <= 1.0)

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

表格 8-123 参数的数据类型

参数	数据类型 [1]	说明
MIN	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	输入范围的最小值
VALUE	SCALE_X：Real, LReal NORM_X：SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	要标定或标准化的输入值
MAX	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	输入范围的最大值
OUT	SCALE_X：SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal NORM_X：Real, LReal	标定或标准化后的输出值

[1] 对于 SCALE_X：参数 MIN、MAX 和 OUT 的数据类型必须相同。
对于 NORM_X：参数 MIN、VALUE 和 MAX 的数据类型必须相同。

说明

SCALE_X 参数 VALUE 应限制为 (0.0 <= VALUE <= 1.0)

如果参数 VALUE 小于 0.0 或大于 1.0：

- 线性标定运算会生成一些小于 MIN 参数值或大于 MAX 参数值的 OUT 值，作为 OUT 值，这些数值在 OUT 数据类型值范围内。此时，SCALE_X 执行会设置 ENO = TRUE。
- 还可能会生成一些不在 OUT 数据类型值范围内的标定数值。此时，OUT 参数值会被设置为一个中间值，该中间值等于被标定实数在最终转换为 OUT 数据类型之前的最低有效部分。在这种情况下，SCALE_X 执行会设置 ENO = FALSE。

NORM_X 参数 VALUE 应限制为 (MIN <= VALUE <= MAX)

如果参数 VALUE 小于 MIN 或大于 MAX，线性标定运算会生成小于 0.0 或大于 1.0 的标准化 OUT 值。在这种情况下，NORM_X 执行会设置 ENO = TRUE。

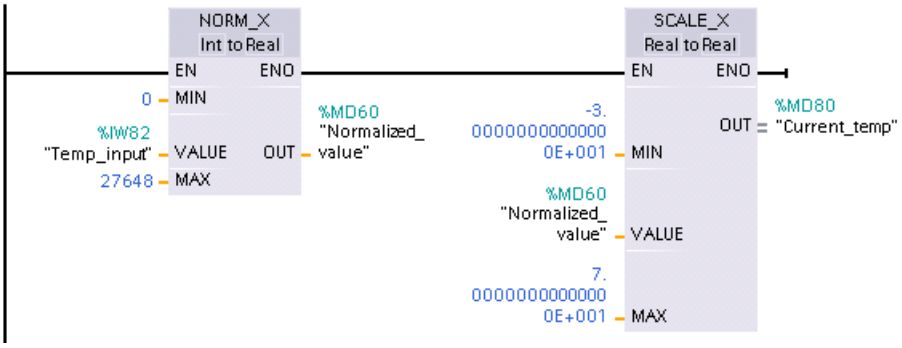
表格 8-124 ENO 状态

ENO	条件	结果 (OUT)
1	无错误	有效结果
0	结果超出 OUT 数据类型的有效范围	中间结果：实数在最终转换为 OUT 数据类型前的最低有效部分。
0	参数 MAX <= MIN	SCALE_X：用实数 VALUE 的最低有效部分填充 OUT 大小。 NORM_X：扩展 VALUE 数据类型中的 VALUE 来填充双字大小。
0	参数 VALUE = +/- INF 或 +/- NaN	将 VALUE 写入 OUT

示例 (LAD)：标准化和标定模拟量输入值

来自电流输入型模拟量信号模块或信号板的模拟量输入的有效值在 0 到 27648 范围内。假设模拟量输入代表温度，其中模拟量输入值 0 表示 -30.0 摄氏度，27648 表示 70.0 摄氏度。

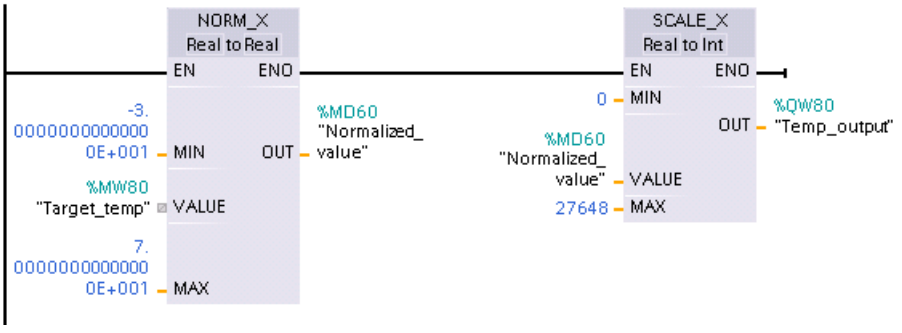
要将模拟值转换为对应的工程单位，应将输入标准化为 0.0 到 1.0 之间的值，然后再将其标定为 -30.0 到 70.0 之间的值。结果值是用模拟量输入（以摄氏度为单位）表示的温度：



请注意，如果模拟量输入来自电压型模拟量信号模块或信号板，则 NORM_X 指令的 MIN 值是 -27648，而不是 0。

示例 (LAD)：标准化和标定模拟量输出值

要在电流输出型模拟量信号模块或信号板中设置的模拟量输出的有效值必须在 0 到 27648 范围内。假设模拟量输出表示温度设置，其中模拟量输入值 0 表示 -30.0 摄氏度，27648 表示 70.0 摄氏度。要将存储器中的温度值（范围是 -30.0 到 70.0）转换为 0 到 27648 范围内的模拟量输出值，必须将以工程单位表示的值标准化为 0.0 到 1.0 之间的值，然后将其标定为 0 到 27648 范围内的模拟量输出值：



请注意，如果模拟量输出应用到电压型模拟量信号模块或信号板，则 SCALE_X 指令的 MIN 值是 -27648，而不是 0。

有关电压和电流形式的模拟量输出表示法 (页 1556)和模拟量输出表示法 (页 1557)的详细信息，请参见技术规范。

8.7.6 变量转换指令

8.7.6.1 VARIANT_TO_DB_ANY（将 VARIANT 转换为 DB_ANY）

可以使用“VARIANT to DB_ANY”指令读取 IN 参数处的操作数，然后将其转化为数据类型 DB_ANY。IN 参数属于 Variant 数据类型，并且代表实例数据块或者 ARRAY 数据块。创建程序时，不需要知道哪个数据块与 IN 参数相对应。指令在运行期间读取数据块编号，并将其写入到 RET_VAL 参数的操作数中。

表格 8-125 VARIANT_TO_DB_ANY 指令

LAD/FBD	SCL	说明
不提供	`RET_VAL := VARIANT_TO_DB_ANY( in := _variant_in_, err => _int_out_);`	从 Variant IN 参数读取操作数，并将其存储到函数结果中（采用 DB_ANY 类型）

表格 8-126 VARIANT_TO_DB_ANY 指令的参数

参数	数据类型	说明
IN	Variant	代表实例数据块或者数组数据块的变量
RET_VAL	DB_ANY	包含已转换数据块编号的 DB_ANY 数据类型输出
ERR	Int	错误信息

表格 8-127 ENO 状态

ENO	条件	结果
1	无错误	指令会对输入 Variant 进行转换，并将其存储到 DB_ANY 函数输出中
0	使能输入 EN 的信号状态为“0”或 IN 参数无效。	指令不起任何作用。

表格 8-128 VARIANT_TO_DB_ANY 参数的错误输出代码

Err (W#16#...)	说明
0000	无错误
252C	IN 参数中的 Variant 数据类型值为 0。CPU 切换至 STOP 模式。
8131	数据块不存在或过短（首次访问）。
8132	数据块过短并且不是 Array 数据块（第二次访问）。
8134	数据块处于写保护状态
8150	参数 IN 中的 Variant 数据类型的值为“0”。要接收此错误信息，必须激活 “在块内处理错误”(Handle errors within block) 块属性。否则，CPU 将切换到 STOP 模式，并发送错误代码 16#252C
8154	数据块的数据类型不正确。
* 错误代码可在程序编辑器中显示为整数或十六进制值。	

8.7.6.2 DB_ANY_TO_VARIANT（将 DB_ANY 转换为 VARIANT）

可以使用“DB_ANY to VARIANT”读取符合下列要求的数据块的编号。IN 参数中的操作数采用 DB_ANY 数据类型，这意味着，创建程序时不需要知道要读取哪个数据块。指令在运行期间读取数据块编号，并通过 VARIANT 指针将其写入到函数结果 RET_VAL 中。

表格 8-129 DB_ANY_TO_VARIANT 指令

LAD/FBD	SCL	说明
不提供	`RET_VAL := DB_ANY_TO_VARIANT( in := _db_any_in_, err => _int_out_);`	从 Variant IN 参数中读取数据块编号，并将其存储到函数结果中（采用类型 Variant）。

表格 8-130 DB_ANY_TO_VARIANT 指令的参数

参数	数据类型	说明
IN	DB_ANY	包含数据块编号的变量
RET_VAL	Variant	包含已转换数据块编号的 DB_ANY 数据类型输出
ERR	Int	错误信息

表格 8-131 ENO 状态

ENO	条件	结果
1	无错误	指令将转换变量中的数据块编号，并将其存储到 DB_ANY 函数输出中
0	使能输入 EN 的信号状态为“0”或 IN 参数无效。	指令不起任何作用。

表格 8-132 DB_ANY_TO_VARIANT 参数的错误输出代码

Err (W#16#...)	说明
0000	无错误
8130	数据块编号为 0。
8131	数据块不存在或过短。
8132	数据块过短并且不是 Array 数据块。
8134	数据块处于写保护状态。
8154	数据块的数据类型不正确。
8155	未知类型代码
* 错误代码可在程序编辑器中显示为整数或十六进制值。	

8.8 程序控制操作

8.8.1 JMP（RLO = 1 时跳转）、JMPN（RLO = 0 时跳转）和 Label（跳转标签）指令

表格 8-133 JMP、JMPN 和 LABEL 指令

LAD	FBD	SCL	说明
Label_name —{JMP}—|	Label_name JMP	请参见 GOTO (页 327) 语句。	RLO（逻辑运算结果）= 1 时跳转： 如果有能流通过 JMP 线圈 (LAD)，或者 JMP 功能框的输入为真 (FBD)，则程序将从指定标签后的第一条指令继续执行。
Label_name —{JMPN}—|	Label_name JMPN		RLO = 0 时跳转： 如果没有能流通过 JMPN 线圈 (LAD)，或者 JMPN 功能框的输入为假 (FBD)，则程序将从指定标签后的第一条指令继续执行。
Label_name	Label_name		JMP 或 JMPN 跳转指令的目标标签。

[1] 通过在 LABEL 指令中直接键入来创建标签名称。可以使用参数助手图标来选择 JMP 和 JMPN 标签名称字段可用的标签名称。也可在 JMP 或 JMPN 指令中直接键入标签名称。

表格 8-134 参数的数据类型

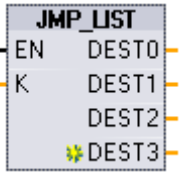
参数	数据类型	说明
Label_name	标签标识符	跳转指令以及相应跳转目标程序标签的标识符

- 各标签在代码块内必须唯一。
- 可以在代码块中进行跳转，但不能从一个代码块跳转到另一个代码块。
- 可以向前或向后跳转。
- 可以在同一代码块中从多个位置跳转到同一标签。

8.8.2 JMP_LIST（定义跳转列表）

表格 8-135 JMP_LIST 指令

LAD/FBD	SCL	说明

	`CASE k OF`
	`  0: GOTO dest0;`
	`  1: GOTO dest1;`
	`  2: GOTO dest2;`
	`  [n: GOTO destn;]`
	`END_CASE;`
		JMP_LIST 指令用作程序跳转分配器，控制程序段的执行。根据 K 输入的值跳转到相应的程序标签。程序从目标跳转标签后面的程序指令继续执行。如果 K 输入的值超过（标签数 - 1），则不进行跳转，继续处理下一程序段。

表格 8-136 参数的数据类型

参数	数据类型	说明
K	UInt	跳转分配器控制值
DEST0, DEST1, .., DESTn.	程序标签	与特定 K 参数值对应的跳转目标标签： 如果 K 的值等于 0，则跳转到分配给 DEST0 输出的程序标签。如果 K 的值等于 1，则跳转到分配给 DEST1 输出的程序标签，以此类推。如果 K 输入的值超过（标签数 - 1），则不进行跳转，继续处理下一程序段。

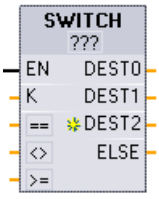
对于 LAD 和 FBD：在程序中第一次放置 JMP_LIST 功能框时，该功能框有两个跳转标签输出。可以添加或删除跳转目标。

单击功能框内的创建图标（位于最后一个 DEST 参数的左侧）可添加新的跳转标签输出。

- 右键单击输出短线，并选择“插入输出”(Insert ouput) 命令。
- 右键单击输出短线，并选择“删除”(Delete) 命令。

8.8.3 SWITCH（跳转分配器）

表格 8-137 SWITCH 指令

LAD/FBD	SCL	说明
	不提供	SWITCH 指令用作程序跳转分配器，控制程序段的执行。根据 K 输入的值与分配给指定比较输入的值的比较结果，跳转到与第一个为“真”的比较测试相对应的程序标签。如果比较结果都不为 TRUE，则跳转到分配给 ELSE 的标签。程序从目标跳转标签后面的程序指令继续执行。

[1] 对于 LAD 和 FBD：在功能框名称下方单击，并从下拉菜单中选择数据类型。

[2] 对于 SCL：使用 IF-THEN 语句进行比较。

表格 8-138 参数的数据类型

参数	数据类型 [1]	说明
K	UInt	常用比较值输入
==、<>、<、<=、>. >=	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, TOD, Date	分隔比较值输入，获得特定比较类型
DEST0, DEST1, .., DESTn, ELSE	程序标签	与特定比较对应的跳转目标标签： 首先处理 K 输入下面的第一个比较输入，如果 K 值与该输入的比较结果为“真”，则跳转到分配给 DEST0 的标签。下一比较测试使用接下来的下一个输入，如果比较结果“真”，则跳转到分配给 DEST1 的标签。依次对其它比较进行类似的处理，如果比较结果都不为“真”，则跳转到分配给 ELSE 输出的标签。

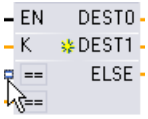
[1] K 输入和比较输入 (==, <>, <, <=, >, >=) 的数据类型必须相同。

添加输入、删除输入和指定比较类型

在程序中第一次放置 LAD 或 FBD SWITCH 功能框时，该功能框有两个比较输入。可以分配比较类型以及添加输入/跳转目标，如下所示。

单击功能框内的比较运算符，并从下拉列表中选择新运算符。

单击功能框中的创建图标（位于最后一个 DEST 参数的左侧）可添加新的比较目标参数。



- 右键单击输入短线，并选择“插入输入”(Insert input) 命令。
- 右键单击输入短线并选择“删除”(Delete) 命令。

表格 8-139　SWITCH 功能框数据类型选择和允许的比较运算

数据类型	比较	运算符语法
Byte、Word、DWord	等于	==
	不等于	<>
SInt、Int、DInt、USInt、UInt、UDInt、Real、LReal、Time、TOD、Date	等于	==
	不等于	<>
	大于或等于	>=
	小于或等于	<=
	大于	>
	小于	<

SWITCH 功能框放置规则

- 比较输入前可以不连接 LAD/FBD 指令。
- 由于没有 ENO 输出，因此，在一个程序段中只允许使用一条 SWITCH 指令，并且 SWITCH 指令必须是程序段中的最后一个运算。

8.8.4 RET（返回）

可选的 RET 指令用于终止当前块的执行。 当且仅当有能流通过 RET 线圈 (LAD)，或者当 RET 功能框的输入为真 (FBD) 时，则当前块的程序执行将在该点终止，并且不执行 RET 指令以后的指令。 如果当前块为 OB，则参数“Return_Value”将被忽略。 如果当前块为 FC 或 FB，则将参数“Return_Value”的值作为被调用功能框的 ENO 值传回到调用例程。

不要求用户将 RET 指令用作块中的最后一个指令；该操作是自动完成的。 一个块中可以有多个 RET 指令。

有关 SCL，请参见 RETURN (页 328) 语句。

表格 8-140 Return_Value (RET) 执行控制指令

LAD	FBD	SCL	说明
"Return_Value" —(RET)—|	"Return_Value" RET	RETURN;	终止当前块的执行

表格 8-141 参数的数据类型

参数	数据类型	说明
Return_Value	Bool	RET 指令的“Return_value”参数被分配给调用块中块调用功能框的 ENO 输出。

以下是在 FC 代码块中使用 RET 指令的示例步骤：

1. 创建新项目并添加 FC：
2. 编辑该 FC：
   – 从指令树添加指令。
   – 添加一个 RET 指令，包括参数“Return_Value”的以下值之一：
     TRUE、FALSE，或用于指定所需返回值的存储位置。
   – 添加更多的指令。
3. 从 MAIN [OB1] 调用 FC。

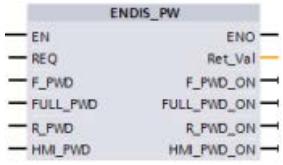
MAIN 代码块中 FC 功能框的 EN 输入必须为真，才能开始执行 FC。

执行了有能流通过 RET 指令的 FC 后，该 FC 的 RET 指令所指定的值将出现在 MAIN 代码块中 FC 功能框的 ENO 输出上。

8.8.5 ENDIS_PW（启用/禁用 CPU 密码）

表格 8-142 ENDIS_PW 指令

LAD/FBD	SCL	说明

	`ENDIS_PW(`
	`req:=_bool_in_,`
	`f_pwd:=_bool_in_,`
	`full_pwd:=_bool_in_,`
	`r_pwd:=_bool_in_,`
	`hmi_pwd:=_bool_in_,`
	`f_pwd_on=>_bool_out_,`
	`full_pwd_on=>_bool_out_,`
	`r_pwd_on=>_bool_out_,`
	`hmi_pwd_on=>_bool_out_);`
		即使客户端能够提供正确的密码，ENDIS_PW 指令也可以允许或禁止客户端连接到 S7-1200 CPU。
		此指令不会禁止 Web 服务器密码。

表格 8-143 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	如果 REQ=1，执行函数
F_PWD	IN	Bool	故障安全密码： 允许 (=1) 或禁止 (=0)
FULL_PWD	IN	Bool	完全访问密码： 允许 (=1) 或禁止 (=0) 完全访问密码
R_PWD	IN	Bool	读访问密码： 允许 (=1) 或禁止 (=0)
HMI_PWD	IN	Bool	HMI 密码： 允许 (=1) 或禁止 (=0)
F_PWD_ON	OUT	Bool	故障安全密码状态： 已允许 (=1) 或已禁止 (=0)
FULL_PWD_ON	OUT	Bool	完全访问密码状态： 已允许 (=1) 或已禁止 (=0)
R_PWD_ON	OUT	Bool	只读密码状态： 已允许 (=1) 或已禁止 (=0)
HMI_PWD_ON	OUT	Bool	HMI 密码状态： 已允许 (=1) 或已禁止 (=0)
Ret_Val	OUT	Word	函数结果

使用 REQ=1 调用 ENDIS_PW 会禁止相应密码输入参数为 FALSE 的密码类型。 可以单独允许或禁止每个密码类型。 例如，如果允许故障安全密码但是禁止所有其它密码，则可以限制 CPU 访问一小组员工。

程序扫描期间会同步执行 ENDIS_PW，并且密码输出参数始终显示允许密码的当前状态，与无论输入参数 REQ 无关。 设置为允许的所有密码必须可更改为禁用/允许。 否则会返回错误，并且执行 ENDIS_PW 前处于允许状态的所有密码都将恢复为允许。 也就是说，在标准 CPU

（未组态故障安全密码）中，F_PWD 必须始终设置为 1，以便生成返回值 0。在本例中，F_PWD_ON 始终为 1。

说明

- 如果 HMI 密码处于禁止状态，执行 ENDIS_PW 可以阻止 HMI 设备的访问。
- 执行 ENDIS_PW 后，先于 ENDIS_PW 获得授权的客户端会话保持不变。

上电后，CPU 访问会受到先前在常规 CPU 保护组态中所定义密码的限制。 必须执行新的 ENDIS_PW 以重新建立禁止有效密码的能力。 不过，如果立即执行 ENDIS_PW 并禁止所需密码，则可以锁定 TIA Portal 访问。 在密码禁止之前，您可以使用定时器指令延迟 ENDIS_PW 执行，以留出时间输入密码。

说明

恢复锁定 TIA Portal 通信的 CPU

有关如何使用存储卡擦除 PLC 内部装载存储器的详细信息，请参见主题“丢失密码后恢复 (页 145)”。

由于发生错误而将工作模式更改为 STOP 时，STP 执行或 STEP 7 不会取消保护。 在 CPU 循环上电前，保护始终有效。 请参见下表了解详细信息。

操作	工作模式	ENDIS_PW 密码控制
通过 STEP 7 复位存储器后	STOP	活动： 已禁止密码保持禁止状态。
上电或更换存储卡后	STOP	关： 未禁止任何密码。
在程序循环 OB 或启动 OB 中执行 ENDIS_PW 后	STARTUP、RUN	活动： 根据 ENDIS_PW 参数禁止密码
通过 RUN 或 STARTUP 更改工作模式后，或通过 STP 指令、错误或 STEP 7 将 STARTUP 更改为 STOP 后	STOP	活动： 已禁止密码保持禁止状态

说明

使用强密码对 CPU 访问级别进行密码保护。 强密码在长度上至少为十个字符，可以是字母、数字和特殊字符的组合，不能是可在字典上找到的词，并且不能是可从个人信息推断出的名字或标识符。 保管好密码并经常更改密码。

表格 8-144 条件代码

RET_VAL (W#16#...)	说明
0000	无错误
8090	不支持该指令。
80D0	未组态故障安全密码。
80D1	未组态读/写访问密码。
80D2	未组态读访问密码。
80D3	未组态 HMI 访问密码。

8.8.6 RE_TRIGR（重置周期监视时间）

表格 8-145 RE_TRIGR 指令

LAD/FBD	SCL	说明
RE_TRIGR EN ENO	`RE_TRIGR();`	RE_TRIGR（重新触发扫描时间监视狗）用于延长扫描循环监视狗定时器生成错误前允许的最大时间。

RE_TRIGR 指令用于在单个扫描循环期间重新启动扫描循环监视定时器。 结果是从最后一次执行 RE_TRIGR 功能开始，使允许的最大扫描周期延长一个最大循环时间段。

说明

对于 S7-1200 CPU 固件版本 2.2 之前的版本，RE_TRIGR 限制为从程序循环 OB 执行，并可能用于无限期地延长 PLC 扫描时间。 如果从启动 OB、中断 OB 或错误 OB 执行 RE_TRIGR，则不会复位监视狗定时器且 ENO = FALSE。

对于固件版本 2.2 及以上版本，可从任何 OB（包括启动、中断和错误 OB）执行 RE_TRIGR。 但是，PLC 扫描时间最长只能延长到已组态最大循环时间的 10 倍。

设置 PLC 最大循环时间

可以在设备配置的 “循环时间”(Cycle time) 下组态最大扫描循环时间值。

表格 8-146 循环时间值

循环时间监视	最小值	最大值	默认值
最大循环时间	1 ms	6000 ms	150 ms

监视狗超时

如果最大扫描循环定时器在扫描循环完成前达到预置时间，则会生成错误。 如果用户程序中包含时间错误中断 OB (OB 80)，则 CPU 将执行时间错误中断 OB，该中断可包含程序逻辑以创建具体响应。

如果用户程序不包含时间错误中断 OB，则忽略第一个超时条件并且 CPU 保持在 RUN 模式。 如果在同一程序扫描中第二次发生最大扫描时间超时（2 倍的最大循环时间值），则触发错误会导致切换到 STOP 模式。

在 STOP 模式下，用户程序停止执行，而 CPU 系统通信和系统诊断仍继续执行。

8.8.7 STP（退出程序）

表格 8-147 STP 指令

LAD/FBD	SCL	说明
	`STP();`	STP 可将 CPU 置于 STOP 模式。 CPU 处于 STOP 模式时，将停止程序执行并停止过程映像的物理更新。

有关详细信息，请参见： 组态从 RUN 切换到 STOP 时的输出 (页 112)。

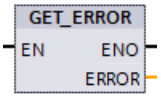
如果 EN = TRUE，CPU 将进入 STOP 模式，程序执行停止，并且 ENO 状态无意义。 否则，EN = ENO = 0。

8.8.8 GET_ERROR 和 GET_ERROR_ID（获取本地错误信息和获取本地错误 ID）指令

获取错误指令提供有关程序块执行错误的信息。 如果在代码块中添加了 GET_ERROR 或 GET_ERROR_ID 指令，便可在程序块中处理程序错误。

GET_ERROR

表格 8-148 GET_ERROR 指令

LAD/FBD	SCL	说明
	`GET_ERROR(_out_);`	指示发生本地程序块执行错误，并用详细错误信息填充预定义的错误数据结构。

表格 8-149 参数的数据类型

参数	数据类型	说明
ERROR	ErrorStruct	错误数据结构： 可以重命名该结构，但不能重命名结构中的成员。

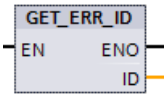
表格 8-150 ErrorStruct 数据结构的元素

结构组件		数据类型	说明
ERROR_ID		Word	错误 ID
FLAGS		Byte	显示块调用期间是否出错。 • 16#01: 块调用期间出错。 • 16#00: 块调用期间未出错。
REACTION		Byte	默认响应： • 0: 忽略（写入错误）， • 1: 以替代值“0”继续（读取错误）， • 2: 跳转指令（系统错误）
CODE_ADDRESS		CREF	有关块地址和类型的信息
	BLOCK_TYPE	Byte	出错块的类型： • 1: OB • 2: FC • 3: FB
	CB_NUMBER	UInt	代码块的编号
	OFFSET	UDInt	对内部存储器的引用

<table>
<tr><th>结构组件</th><th>数据类型</th><th>说明</th></tr>
<tr><td rowspan="2">MODE</td><td rowspan="2">Byte</td><td>访问模式： 根据具体的访问类型，可输出以下信息：
<table>
<tr><th>模式</th><th>(A)</th><th>(B)</th><th>(C)</th><th>(D)</th><th>(E)</th></tr>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>1</td><td></td><td></td><td></td><td></td><td>偏移</td></tr>
<tr><td>2</td><td></td><td></td><td>区域</td><td></td><td></td></tr>
<tr><td>3</td><td>位置</td><td>范围</td><td></td><td>编号</td><td></td></tr>
<tr><td>4</td><td></td><td></td><td>区域</td><td></td><td>偏移</td></tr>
<tr><td>5</td><td></td><td></td><td>区域</td><td>DB 编号</td><td>偏移</td></tr>
<tr><td>6</td><td>伙伴编号 /访问</td><td></td><td>区域</td><td>DB 编号</td><td>偏移</td></tr>
<tr><td>7</td><td>伙伴编号 /访问</td><td>插槽号/范围</td><td>区域</td><td>DB 编号</td><td>偏移</td></tr>
</table></td></tr>
<tr><td></td><td></td></tr>
<tr><td>OPERAND_NUMBER</td><td>UInt</td><td>机器命令的操作数</td></tr>
<tr><td>POINTER_NUMBER_LOCATION</td><td>UInt</td><td>(A) 内部指针</td></tr>
<tr><td>SLOT_NUMBER_SCOPE</td><td>UInt</td><td>(B) 内部存储器中的存储区</td></tr>
<tr><td colspan="2">DATA_ADDRESS</td><td>NREF</td><td>有关操作数地址的信息</td></tr>
<tr><td></td><td>AREA</td><td>Byte</td><td>(C) 存储区：
• L： 16#40 – 4E、86、87、8E、8F、C0 – CE
• I： 16#81
• Q： 16#82
• M： 16#83
• DB： 16#84、85、8A、8B</td></tr>
<tr><td></td><td>DB_NUMBER</td><td>UInt</td><td>(D) 数据块编号</td></tr>
<tr><td></td><td>OFFSET</td><td>UDInt</td><td>(E) 操作数的相对地址</td></tr>
</table>

GET_ERROR_ID

表格 8-151 GetErrorID 指令

LAD/FBD	SCL	说明
	`GET_ERR_ID();`	指示发生程序块执行错误，并报告错误的 ID（标识符代码）。

表格 8-152 参数的数据类型

参数	数据类型	说明
ID	Word	ErrorStruct ERROR_ID 成员的错误标识符值

表格 8-153 Error_ID 值

ERROR_ID 十六进制值	ERROR_ID 十进制值	程序块执行错误
0	0	无错误
2520	9504	损坏的字符串
2522	9506	操作数超出范围读取错误
2523	9507	操作数超出范围写入错误
2524	9508	无效区域读取错误
2525	9509	无效区域写入错误
2528	9512	数据分配读取错误（位赋值不正确）
2529	9513	数据分配写入错误（位赋值不正确）
252C	9516	未初始化指针错误
2530	9520	DB 受到写保护
2533	9523	使用了无效指针
2538	9528	访问错误：DB 不存在
2539	9529	访问错误：使用了错误 DB
253A	9530	全局 DB 不存在
253C	9532	版本错误或 FC 不存在
253D	9533	指令不存在
253E	9534	版本错误或 FB 不存在
253F	9535	指令不存在
2550	9552	访问错误：DB 不存在
2575	9589	程序嵌套深度错误
2576	9590	局部数据分配错误
2942	10562	物理输入点不存在
2943	10563	物理输出点不存在

运行

默认情况下，CPU 通过将错误记录到诊断缓冲区来响应块执行错误。 但是，如果在代码块中放置一个或多个 GET_ERROR 或 GET_ERROR_ID 指令，即将该块设置为在块内处理错误。 在这种情况下，CPU 不在诊断缓冲区中记录错误。 而是在 GET_ERROR 或 GET_ERROR_ID 指令的输出中报告错误信息。 可以使用 GET_ERROR 指令读取详细错误信息，或使用 GET_ERROR_ID 指令只读取错误标识符。 因为后续错误往往只是第一个错误的结果，所以第一个错误通常最重要。

在块内第一次执行 GET_ERROR 或 GET_ERROR_ID 指令将返回块执行期间检测到的第一个错误。 在块启动到执行 GET_ERROR 或 GET_ERROR_ID 期间随时都可能发生该错误。 随后执行 GET_ERROR 或 GET_ERROR_ID 将返回上次执行 GET_ERROR 或 GET_ERROR_ID 以来发生的第一个错误。 不保存错误历史，执行任一指令都将使 PLC 系统重新捕捉下一个错误。

可以在数据块编辑器和块接口编辑器中添加 GET_ERROR 指令所使用的 ErrorStruct 数据类型，这样程序逻辑便可以访问这些值。 从数据类型下拉列表中选择 ErrorStruct 以添加该结构。 您可以使用唯一的名称创建多个 ErrorStruct 元素。 不能重命名 ErrorStruct 的成员。

ENO 指示的错误条件

如果 EN = TRUE 且 GET_ERROR 或 GET_ERROR_ID 执行，则：

- ENO = TRUE 表示发生代码块执行错误并提供错误数据
- ENO = FALSE 表示未发生代码块执行错误

可以将错误响应程序逻辑连接到在发生错误后激活的 ENO。 如果存在错误，该输出参数会将错误数据存储在程序能够访问这些数据的位置。

GET_ERROR 和 GET_ERROR_ID 可用来将错误信息从当前执行块（被调用块）发送到调用块。 将该指令放置在被调用块程序的最后一个程序段中可以报告被调用块的最终执行状态。

8.8.9 RUNTIME（测量程序运行时间）

表格 8-154 RUNTIME 指令

LAD/FBD	SCL	说明
RUNTIME EN ENO MEM Ret_Val	`Ret_Val := RUNTIME(` `  _lread_inout_);`	测量整个程序、各个块或命令序列的运行时间。

如果要测量整个程序的运行时间，请在 OB1 中调用指令“测量程序运行时间”。运行时间的测量从第一次调用指令开始，输出 RET_VAL 将在第二次调用后返回程序的运行时间。测

得的运行时间包括程序执行期间可能发生的所有 CPU 进程，如，由更高级别的事件或通信所引发的中断。“测量程序运行时间”指令读取 CPU 的内部计数器并将值写入 IN-OUT 参数 MEM。该指令根据内部计数器频率计算当前程序运行时间并将其写入输出 RET_VAL。

如果要测量单个块或单个命令序列的运行时间，您需要三个单独的程序段。在程序内的单个程序段内分别调用指令“测量程序运行时间”。在首次调用该指令时设置运行时间测量的起点。然后在下一程序段中调用所要测量的程序块或命令序列。在另一个程序段中，第二次调用“测量程序运行时间”指令，然后如同在第一次调用该指令那样，为 IN-OUT 参数 MEM 分配相同的内存。在第三个程序段中，“测量程序运行时间”指令读取内部 CPU 计数器，然后根据内部计数器频率计算程序块或命令序列的当前运行时间并将其写入输出 RET_VAL。

“测量程序运行时间”指令使用内部高频计数器来计算时间。如果计数器溢出，该指令返回值 <= 0.0。请忽略此类运行时间值。

说明

CPU 不能准确确定某个命令序列的运行时间，因为在程序的优化编译期间，命令序列内的指令序列会发生变化。

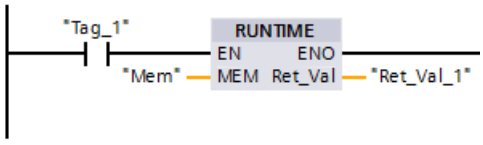
表格 8-155　参数的数据类型

参数	数据类型	说明
MEM	LReal	运行时间测量的起点
RET_VAL	LReal	测得的运行时间（以秒为单位）

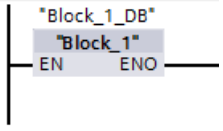
示例：RUNTIME 指令

以下示例显示如何使用 RUNTIME 指令来测量函数块的执行时间：

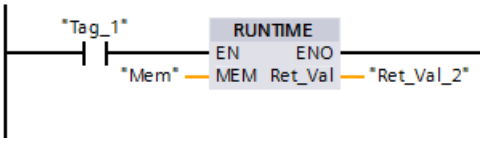
程序段 1：



程序段 2：



程序段 3：



当程序段 1 中的“Tag_1”操作数的信号状态为“1”时，RUNTIME 指令执行。在首次调用该指令时设置运行时间测量的起点，并作为第二次调用该指令的参考值缓冲到“Mem”操作数中。

函数块 FB1 在程序段 2 中执行。

当 FB1 程序块完成并且“Tag_1”操作数的信号状态为“1”时，程序段 3 中的 RUNTIME 指令执行。第二次调用该指令时将计算程序块的运行时间并将结果写入输出 RET_VAL_2。

8.8.10 SCL 程序控制语句

8.8.10.1 SCL 程序控制语句概述

结构化控制语言 (SCL, Structured Control Language) 提供三类用于结构化用户程序的程序控制语句：

- 选择语句： 选择语句可将程序执行转移到备选语句序列。
- 循环： 可以使用迭代语句控制循环执行。 迭代语句指定应根据某些条件重复执行的程序部分。
- 程序跳转： 程序跳转是指立刻跳转到特定的跳转目标，因而跳转到同一块内的其它语句。

这些程序控制语句都使用 PASCAL 编程语言的语法。

表格 8-156 SCL 程序控制语句类型

程序控制语句		说明
选择	IF-THEN 语句 (页 321)	用将程序执行转移到两个备选分支之一（取决于条件为 TRUE 还是 FALSE）
	CASE 语句 (页 322)	用于选择执行 *n* 个备选分支之一（取决于变量值）
循环	FOR 语句 (页 323)	只要控制变量在指定值范围内，就重复执行某一语句序列
	WHILE-DO 语句 (页 324)	只要仍满足执行条件，就重复执行某一语句序列
	REPEAT-UNTIL 语句 (页 325)	重复执行某一语句序列，直到满足终止条件为止

程序控制语句		说明
程序跳转	CONTINUE 语句 (页 326)	停止执行当前循环迭代
	EXIT 语句 (页 327)	无论是否满足终止条件，都会随时退出循环
	GOTO 语句 (页 327)	使程序立即跳转到指定标签
	RETURN 语句 (页 328)	使程序立刻退出正在执行的块，返回到调用块

8.8.10.2 IF-THEN 语句

IF-THEN 语句是条件语句，可控制程序流，根据逻辑表达式的 Bool 值的结果决定是否执行一组语句。您还可以使用括号嵌套或结构化多条 IF-THEN 语句的执行。

表格 8-157 IF-THEN 语句的元素

SCL	说明

`IF "condition" THEN`
`statement_A;`
`statement_B;`
`statement_C;`
`;`
	如果“condition”为 TRUE 或 1，则执行后面的语句，直到遇到 END_IF 语句为止。
	如果“condition”为 FALSE 或 0，则跳转到 END_IF 语句（除非程序包含可选的 ELSIF 或 ELSE 语句）。

`[ELSIF "condition-n" THEN`
`statement_N;`
`;]`
	可选的 ELSEIF[1] 语句提供其它要评估的条件。例如：如果 IF-THEN 语句中的“condition”为 FALSE，则程序将评估“condition-n”。如果“condition-n”为 TRUE，则执行“statement_N”。

`[ELSE`
`statement_X;`
`;]`
	可选的 ELSE 语句提供 IF-THEN 语句的“condition”为 FALSE 时将要执行的语句。

`END_IF;`	END_IF 语句用于终止 IF-THEN 指令。

[1] 可以在一条 IF-THEN 语句中包含多条 ELSIF 语句。

表格 8-158 IF-THEN 语句的变量

变量	说明
“condition”	必需。逻辑表达式为 TRUE (1) 或 FALSE (0)。
“statement_A”	可选。“condition”为 TRUE 时要执行的一条或多条语句。
“condition-n”	可选。可选 ELSIF 语句要评估的逻辑表达式。

变量	说明
“statement_N”	可选。ELSIF 语句的“condition-n”为 TRUE 时要执行的一条或多条语句。
“statement_X”	可选。IF-THEN 语句的“condition”为 FALSE 时要执行的一条或多条语句。

IF 语句按照下列规则执行：

- 执行第一个逻辑表达式为 TRUE 的语句序列。不执行其余语句序列。
- 如果无布尔型表达式为 TRUE，则执行 ELSE 引入的语句序列（或者，如果 ELSE 分支不存在，则不执行语句序列）。
- 不限制 ELSIF 语句的数量。

说明

与 IF 语句相比，使用一个或多个 ELSIF 分支存在一定的优势，就是不用再评估有效表达式后面的逻辑表达式。从而，可缩短程序的运行时间。

8.8.10.3 CASE 语句

表格 8-159 CASE 语句的元素

SCL	说明

`CASE "Test_Value" OF`
`  "ValueList":Statement[; Statement, ...]`
`  "ValueList":Statement[; Statement, ...]`
`[ELSE`
`Else-statement[; Else-statement, ...]]`
`END_CASE;`
	CASE 语句根据表达式的值来选择执行多组语句中的一组。

表格 8-160 参数

参数	说明
“Test_Value”	必需。任何 Int 数据类型的数字表达式
“ValueList”	必需。单个值、或逗号分隔的值或值范围的列表。（使用两个句点定义值范围：2..8）下例说明了不同变型的值列表： 1：Statement_A; 2, 4:Statement _B; 3, 5..7,9:Statement _C;
语句	必需。“Test_Value”与值列表中任何一个值匹配时执行的一条或多条语句
Else-statement	可选。与“ValueList”中的任何一个值都不匹配时执行的一条或多条语句

CASE 语句按照下列规则执行：

- Test_value 表达式必须返回一个 Int 类型的值。
- 处理 CASE 语句时，程序会检查 Test_value 表达式的值是否包含在指定的值列表中。如果找到匹配项，则执行分配给该列表的语句成分。
- 如果未找到匹配项，则执行 ELSE 后面的程序段，如果不存在 ELSE 分支，则不执行任何语句。

示例：嵌套 CASE 语句

CASE 语句可以嵌套使用。每个嵌套的 CASE 语句必须具有相关联的 END_CASE 语句。

```
CASE "var1" OF
       1 : #var2 := 'A';
       2 : #var2 := 'B';
ELSE
       CASE "var3" OF

          65..90: #var2 := 'UpperCase';
          97..122: #var2 := 'LowerCase';
       ELSE

          #var2:= 'SpecialCharacter';
       END_CASE;
END_CASE;
```

8.8.10.4 FOR 语句

表格 8-161 FOR 语句的元素

SCL	说明

`FOR "control_variable" := "begin" TO "end"`
`[BY "increment"] DO`
`  statement;`
`  ;`
`END_FOR;`
	FOR 语句用于在控制变量处于指定的值范围内时重复执行某一语句序列。使用 FOR 定义循环时需要指定初始值和最终值。这两个值的数据类型必须与控制变量的相同。
	可以嵌套使用 FOR 循环。END_FOR 语句与最后执行的 FOR 指令配对。

表格 8-162 参数

参数	说明
“control_variable”	必需。整型（Int 或 DInt），用作循环计数器
“begin”	必需。指定控制变量初始值的简单表达式
“end”	必需。确定控制变量最终值的简单表达式
“increment”	可选。每次循环后“control variable”的变化量。“increment”与“control variable”具有相同的数据类型。如果未指定“increment”的值，则每次循环之后，运行变量的值加 1。不能在执行 FOR 语句期间更改“increment”。

FOR 语句的执行方式如下：

- 循环开始时，控制变量设置为初始值（初始分配），每次重复进行循环时，控制变量会增加指定增量（正增量）或减少指定增量（负增量），直至达到最终值。
- 每次执行完循环之后，会检查该条件（达到最终值）以确定是否满足该条件。如果没有满足结束条件，则重新执行语句序列，否则循环将终止并继续执行循环后面的语句。

定义 FOR 语句的规则：

- 控制变量的数据类型只能是 Int 或 DInt。
- 可以省略语句 BY [increment]。如果未指定增量，则自动默认为 +1。

要结束循环而不考虑“condition”表达式的状态，请使用 EXIT 语句 (页 327)。EXIT 语句将执行紧随 END_FOR 语句之后的语句。

使用 CONTINUE 语句 (页 326) 可跳过某个 FOR 循环的后续语句，并继续执行循环，同时检查是否满足终止条件。

8.8.10.5 WHILE-DO 语句

表格 8-163 WHILE 语句

SCL	说明

`WHILE "condition" DO`
`  Statement;`
`  Statement;`
`  ...;`
`END_WHILE;`
	WHILE 语句执行一系列语句，直到给定条件为 TRUE。
	可以嵌套使用 WHILE 循环。END_WHILE 语句与最后执行的 WHILE 指令配对。

表格 8-164 参数

参数	说明
“condition”	必需。值为 TRUE 或 FALSE 的逻辑表达式。（“null”条件被视为 FALSE。）
Statement	可选。在条件值为 TRUE 之前执行的一条或多条语句。

说明

WHILE 语句先评估“condition”的状态，然后执行语句。要执行语句一次或多次而不考虑“condition”的状态，请使用 REPEAT 语句 (页 325)。

WHILE 语句按照下列规则执行：

- 每次循环执行循环体之前，评估执行条件。
- 只要执行条件的值为 TRUE，就重复执行 DO 后面的循环体。
- 一旦值变为 FALSE，则立即跳过循环，去执行循环后面的语句。

要结束循环而不考虑“condition”表达式的状态，请使用 EXIT 语句 (页 327)。EXIT 语句将执行紧随 END_WHILE 语句之后的语句。

使用 CONTINUE 语句可跳过 WHILE 循环后面的语句，并在检查是否满足终止条件后决定是否继续执行循环。

8.8.10.6 REPEAT-UNTIL 语句

表格 8-165 REPEAT 指令

SCL	说明
REPEAT **Statement;** **;** **UNTIL “条件”** **END_REPEAT;**	REPEAT 语句执行一组语句，直到给定条件为 TRUE。 可以嵌套使用 REPEAT 循环。END_REPEAT 语句始终与最后执行的 Repeat 指令配对。

表格 8-166 参数

参数	说明
Statement	可选。在条件值为 TRUE 之前执行的一条或多条语句。
“condition”	必需。一个或多个用以下两种方式表达的表达式：值为 TRUE 或 FALSE 的数字表达式或字符串表达式。“null”条件被视为 FALSE。

说明

在循环的首次迭代过程中，REPEAT 语句在执行相关语句（即使“condition”为 FALSE）后评估“condition”的状态。要在执行这些语句前查看“condition”的状态，请使用 WHILE 语句 (页 324)。

要结束循环而不考虑“condition”表达式的状态，请使用 EXIT 语句 (页 327)。EXIT 语句将执行紧随 END_REPEAT 语句之后的语句。

使用 CONTINUE 语句 (页 326) 可跳过 REPEAT 循环的后续语句，并继续执行循环，同时检查是否满足终止条件。

8.8.10.7 CONTINUE 语句

表格 8-167 CONTINUE 语句

SCL	说明

`CONTINUE`
`  Statement;`
`  ;`
	CONTINUE 语句跳过程序循环（FOR、WHILE、REPEAT）后面的语句，并在检查是否满足终止条件后决定是否继续执行循环。如果不满足，则继续执行循环。

CONTINUE 语句按照下列规则执行：

- 该语句立即终止循环体的执行。
- 根据是否满足重复执行循环的条件，决定是再次执行循环体还是退出迭代语句而去执行紧随其后的语句。
- 在 FOR 语句中，在执行 CONTINUE 语句后控制变量立即增加指定的增量。

只能在循环中使用 CONTINUE 语句。在嵌套循环中，CONTINUE 始终与直接包含它的循环相关。CONTINUE 通常与 IF 语句一起使用。

如果要退出循环而不考虑终止测试情况，请使用 EXIT 语句。

示例：CONTINUE 语句

下例说明了使用 CONTINUE 语句来避免计算值的百分数时发生被 0 除的错误：

```
FOR i := 0 TO 10 DO
IF 值[i] = 0 THEN CONTINUE; END_IF;
     p := part / value[i] * 100;
     s := INT_TO_STRING(p);
     percent := CONCAT(IN1:=s, IN2:="%");
END_FOR;
```

8.8.10.8 EXIT 语句

表格 8-168 EXIT 指令

SCL	说明
`EXIT;`	EXIT 语句用于随时退出循环（FOR、WHILE 或 REPEAT），而不考虑是否满足终止条件。

EXIT 语句按照下列规则执行：

- 该语句会立即退出该退出语句所处的重复语句。
- 继续执行该循环后面（例如 END_FOR 之后）的程序。

在循环中使用 EXIT 语句。在嵌套循环中，EXIT 语句将处理权返回到下一更高嵌套级。

示例：EXIT 语句

```
FOR i := 0 TO 10 DO
CASE value[i, 0] OF
        1..10: value [i, 1]:="A";
        11..40: value [i, 1]:="B";
        41..100: value [i, 1]:="C";
ELSE
EXIT;
END_CASE;
END_FOR;
```

8.8.10.9 GOTO 语句

表格 8-169 GOTO 语句

SCL	说明

`GOTO JumpLabel;`
`Statement;`
`... ;`
`JumpLabel: Statement;`
	GOTO 语句通过跳转到同一块中的某个标签来跳过语句。
	跳转标签（“JumpLabel”）和 GOTO 语句必须在同一个块中。跳转标签的名称只能在块中分配一次。每个跳转标签都可以是多条 GOTO 语句的跳转目标。

不能跳转到循环部分（FOR、WHILE 或 REPEAT）。可以在循环中进行跳转。

示例：GOTO 语句

在以下示例中：根据操作数“Tag_value”的值在对应跳转标签定义的位置继续执行程序。如果“Tag_value”等于 2，则会在跳转标签“MyLabel2”位置继续执行，并会跳过“MyLabel1”。

```
CASE "Tag_value" OF
1 : GOTO MyLabel1;
2 : GOTO MyLabel2;
ELSE GOTO MyLabel3;
END_CASE;
```

```
MyLabel1:"Tag_1" := 1;
MyLabel2:"Tag_2" := 1;
MyLabel3:"Tag_4" := 1;
```

8.8.10.10 RETURN 语句

表格 8-170 RETURN 指令

SCL	说明
`RETURN;`	Return 指令用于无条件退出正在执行的代码块。程序执行返回到调用块或操作系统（退出 OB 时）。

示例：RETURN 指令：

```
IF "错误" <> 0 THEN
RETURN;
END_IF;
```

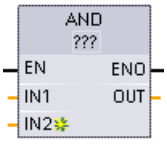
说明

执行最后一条指令后，代码块自动返回到调用块。不要在代码块末尾插入 RETURN 指令。

8.9 字逻辑指令

8.9.1 AND、OR 和 XOR 逻辑运算指令

表格 8-171 AND、OR 和 XOR 逻辑运算指令

LAD/FBD	SCL	说明
	`out := in1 AND in2;`	AND：逻辑 AND
	`out := in1 OR in2;`	OR：逻辑 OR
	`out := in1 XOR in2;`	XOR：逻辑异或

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

要添加输入，请单击 “创建”(Create) 图标，或在其中一个现有 IN 参数的输入短线处单击右键，并选择 “插入输入”(Insert input) 命令。

要删除输入，请在其中一个现有 IN 参数（多于两个原始输入时）的输入短线处单击右键，并选择“删除”(Delete) 命令。

表格 8-172 参数的数据类型

参数	数据类型	说明
IN1，IN2	Byte, Word, DWord	逻辑输入
OUT	Byte, Word, DWord	逻辑输出

[1] 所选数据类型将 IN1、IN2 和 OUT 设置为相同的数据类型。

IN1 和 IN2 的相应位值相互组合，在参数 OUT 中生成二进制逻辑结果。执行这些指令之后，ENO 总是为 TRUE。

8.9.2 INV（求反码）

表格 8-173 INV 指令

LAD/FBD	SCL	说明
	不提供	计算参数 IN 的二进制反码。通过对参数 IN 各位的值取反来计算反码（将每个 0 变为 1，每个 1 变为 0）。执行该指令后，ENO 总是为 TRUE。

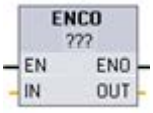
[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

表格 8-174 参数的数据类型

参数	数据类型	说明
IN	SInt, Int, DInt, USInt, UInt, UDInt, Byte, Word, DWord	要取反的数据元素
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Byte, Word, DWord	取反后的输出

8.9.3 DECO（解码）和 ENCO（编码）指令

表格 8-175 ENCO 和 DECO 指令

LAD/FBD	SCL	说明
	`out := ENCO(_in_);`	将位序列编码成二进制数 ENCO 指令将参数 IN 转换为与参数 IN 的最低有效设置位的位位置对应的二进制数，并将结果返回给参数 OUT。如果参数 IN 为 0000 0001 或 0000 0000，则将值 0 返回给参数 OUT。如果参数 IN 的值为 0000 0000，则 ENO 设置为 FALSE。
	`out := DECO(_in_);`	将二进制数解码成位序列 DECO 指令通过将参数 OUT 中的相应位位置设置为 1（其它所有位设置为 0）解码参数 IN 中的二进制数。执行 DECO 指令之后，ENO 始终为 TRUE。 注：DECO 指令的默认数据类型为 DWORD。在 SCL 中，将指令名称更改为 DECO_BYTE 或 DECO_WORD 可解码字节或字值，并分配到字节或字变量或地址。

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

表格 8-176 参数的数据类型

参数	数据类型	说明
IN	ENCO：Byte, Word, DWord DECO：UInt	ENCO：要编码的位序列 DECO：要解码的值
OUT	ENCO：Int DECO：Byte, Word, DWord	ENCO：编码后的值 DECO：解码后的位序列

表格 8-177 ENO 状态

ENO	条件	结果 (OUT)
1	无错误	有效位号
0	IN 为零	OUT 设置为零

DECO 参数 OUT 的数据类型选项（Byte、Word 或 DWord）限制参数 IN 的可用范围。如果参数 IN 的值超出可用范围，将执行求模运算，如下所示提取最低有效位。

DECO 参数 IN 的范围：

- 3 位（值 0-7）IN 用于设置 Byte OUT 中 1 的位位置
- 4 位（值 0-15）IN 用于设置 Word OUT 中 1 的位位置
- 5 位（值 0-31）IN 用于设置 DWord OUT 中 1 的位位置

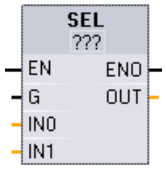
表格 8-178 示例

DECO IN 值			DECO OUT 值（解码单个位位置）
Byte OUT	最小 IN	0	00000001
8 位	最大 IN	7	10000000
Word OUT	最小 IN	0	0000000000000001
16 位	最大 IN	15	1000000000000000
DWord OUT	最小 IN	0	00000000000000000000000000000001
32 位	最大 IN	31	10000000000000000000000000000000

8.9.4 SEL（选择）、MUX（多路复用）和 DEMUX（多路分用）指令

表格 8-179 SEL（选择）指令

LAD/FBD	SCL	说明

	`out := SEL(`
	`g:=_bool_in,`
	`in0:-_variant_in,`
	`in1:=_variant_in);`
		SEL 根据参数 G 的值将两个输入值之一分配给参数 OUT。

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

表格 8-180 SEL 指令的数据类型

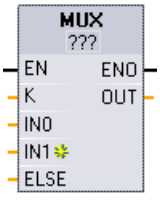
参数	数据类型 [1]	说明
G	Bool	• 0 选择 IN0 • 1 选择 IN1
IN0, IN1	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	输入
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	输出

[1] 输入变量和输出变量必须为相同的数据类型。

条件代码：执行 SEL 指令之后，ENO 始终为 TRUE。

表格 8-181 MUX（多路复用）指令

LAD/FBD	SCL	说明

	`out := MUX(`
	`k:=_unit_in,`
	`in1:=variant_in,`
	`in2:=variant_in,`
	
	`[...in32:=variant_in,]`
	`inelse:=variant_in);`
		MUX 根据参数 K 的值将多个输入值之一复制到参数 OUT。如果参数 K 的值大于 (IN*n* - 1)，则会将参数 ELSE 的值复制到参数 OUT。

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

要添加输入，请单击“创建”(Create) 图标，或在其中一个现有 IN 参数的输入短线处单击右键，并选择“插入输入”(Insert input) 命令。

要删除输入，请在其中一个现有 IN 参数（多于两个原始输入时）的输入短线处单击右键，并选择“删除”(Delete) 命令。

表格 8-182 MUX 指令的数据类型

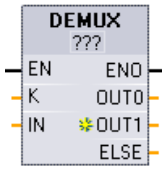
参数	数据类型	说明
K	UInt	• 0 选择 IN1 • 1 选择 IN2 • *n* 选择 IN*n*
IN0, IN1, .. INn	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	输入

参数	数据类型	说明
ELSE	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	输入替换值（可选）
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	输出

[1] 输入变量和输出变量必须为相同的数据类型。

表格 8-183 DEMUX（多路分用）指令

LAD/FBD	SCL	说明

	`DEMUX(`
	`k:=_unit_in,`
	`in:=variant_in,`
	`out1:=variant_in,`
	`out2:=variant_in,`
	
	`[...out32:=variant_in,]`
	
	`outelse:=variant_in);`
		DEMUX 将分配给参数 IN 的位置值复制到多个输出之一。参数 K 的值选择将哪一输出作为 IN 值的目标。如果 K 的值大于数值 (OUT*n* - 1)，则会将 IN 值复制到分配给 ELSE 参数的位置。

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

要添加输出，请单击 “创建”(Create) 图标，或在其中一个现有 OUT 参数的输出短线处单击右键，并选择 “插入输出”(Insert output) 命令。

要删除输出，请在其中一个现有 OUT 参数（多于两个原始输出时）的输出短线处单击右键，并选择 “删除”(Delete) 命令。

表格 8-184 DEMUX 指令的数据类型

参数	数据类型 [1]	说明
K	UInt	选择器的值： • 0 选择 OUT1 • 1 选择 OUT2 • n 选择 OUTn
IN	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	输入

参数	数据类型 [1]	说明
OUT0, OUT1, .. OUTn	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	输出
ELSE	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	K 大于 (OUTn - 1) 时的替换输出

[1] 输入变量和输出变量必须为相同的数据类型。

表格 8-185 MUX 和 DEMUX 指令的 ENO 状态

ENO	条件	结果 (OUT)
1	无错误	MUX：选择的 IN 值被复制到 OUT DEMUX：IN 值被复制到选定的 OUT
0	MUX：K 大于输入数 -1	• 不提供 ELSE：OUT 不变， • 提供 ELSE，将 ELSE 值分配给 OUT
	DEMUX： K 大于输出数 -1	• 不提供 ELSE：输出不变， • 提供 ELSE，将 IN 值复制到 ELSE

8.10 移位与循环移位

8.10.1 SHR（右移）和 SHL（左移）指令

表格 8-186 SHR 和 SHL 指令

LAD/FBD	SCL	说明

	`out := SHR(`
	`  in:=_variant_in_,`
	`  n:=_uint_in);`
	`out := SHL(`
	`  in:=_variant_in_,`
	`  n:=_uint_in);`
		使用移位指令（SHL 和 SHR）移动参数 IN 的位序列。结果将分配给参数 OUT。参数 N 指定移位的位数：
		• SHR：右移位序列
		• SHL：左移位序列

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

表格 8-187　参数的数据类型

参数	数据类型	说明
IN	整数	要移位的位序列
N	USInt, UDint	要移位的位数
OUT	整数	移位操作后的位序列

- 若 N=0，则不移位。将 IN 值分配给 OUT。
- 用 0 填充移位操作清空的位位置。
- 如果要移位的位数 (N) 超过目标值中的位数（Byte 为 8 位、Word 为 16 位、DWord 为 32 位），则所有原始位值将被移出并用 0 代替（将 0 分配给 OUT）。
- 对于移位操作，ENO 总是为 TRUE。

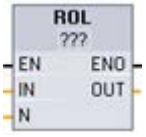
表格 8-188　示例：Word 数据的 SHL

自右插入零，使 Word 的位左移 (N = 1)			
IN	1110 0010 1010 1101	首次移位前的 OUT 值：	1110 0010 1010 1101
		首次左移后：	1100 0101 0101 1010
		第二次左移后：	1000 1010 1011 0100
		第三次左移后：	0001 0101 0110 1000

8.10.2　ROR（循环右移）和 ROL（循环左移）指令

表格 8-189　ROR 和 ROL 指令

LAD/FBD	SCL	说明

	`out := ROL(`
	`in:=_variant_in_,`
	`n:=_uint_in);`
	`out := ROR(`
	`in:=_variant_in_,`
	`n:=_uint_in);`
		循环指令（ROR 和 ROL）用于将参数 IN 的位序列循环移位。结果分配给参数 OUT。参数 N 定义循环移位的位数。
		• ROR：循环右移位序列
		• ROL：循环左移位序列

[1]　对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

表格 8-190 参数的数据类型

参数	数据类型	说明
IN	整数	要循环移位的位序列
N	USInt, UDint	要循环移位的位数
OUT	整数	循环移位操作后的位序列

- 若 N=0，则不循环移位。将 IN 值分配给 OUT。
- 从目标值一侧循环移出的位数据将循环移位到目标值的另一侧，因此原始位值不会丢失。
- 如果要循环移位的位数 (N) 超过目标值中的位数（Byte 为 8 位、Word 为 16 位、DWord 为 32 位），仍将执行循环移位。
- 执行循环指令之后，ENO 始终为 TRUE。

表格 8-191 示例：Word 数据的 ROR

将各个位从右侧循环移出到左侧 (N = 1)			
IN	0100 0000 0000 0001	首次循环移位前的 OUT 值：	0100 0000 0000 0001
		首次循环右移后：	1010 0000 0000 0000
		第二次循环右移后：	0101 0000 0000 0000

扩展指令 9

9.1 日期、时间和时钟功能

9.1.1 日期和时钟指令

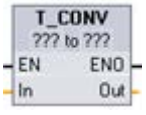
日期和时间指令用于日历和时间计算。

- T_CONV 将值在（日期和时间数据类型）以及（字节、字和双字大小数据类型）之间进行转换
- T_ADD 加上 Time 和 DTL 值：(Time + Time = Time) 或 (DTL + Time = DTL)
- T_SUB 减去 Time 和 DTL 值：(Time - Time = Time) 或 (DTL - Time = DTL)
- T_DIFF 提供两个 DTL 值的差值作为 Time 值：DTL - DTL = Time
- T_COMBINE 将 Date 值和 Time_and_Date 值组合在一起生成 DTL 值

有关 DTL 和 Time 数据的格式信息，请参见时间和日期数据类型 (页 124)部分。

表格 9-1 T_CONV （转换时间并提取）指令

LAD/FBD	SCL 示例	说明

	`out := DINT_TO_TIME(`
	`in:=_variant_in);`
	
	`out := TIME_TO_DINT(`
	`in:=_variant_in);`
		T_CONV 将值在（日期和时间数据类型）以及（字节、字和双字大小数据类型）之间进行转换。

[1] 对于 LAD 和 FBD 框：单击“???”并从下拉菜单中选择源/目标数据类型。

[2] 对于 SCL：将 T_CONV 从指令树拖放到程序编辑器中，然后选择源/目标数据类型。

表格 9-2　T_CONV 转换的有效数据类型

数据类型 IN（或 OUT）	数据类型 OUT（或 IN）
TIME （毫秒）	DInt, Int, SInt, UDInt, UInt, USInt, TOD 仅 SCL： Byte, Word, Dword
DATE （自 1990 年 1 月 1 日起的天数）	DInt, Int, SInt, UDInt, UInt, USInt, DTL 仅 SCL： Byte, Word, Dword
TOD （自午夜起至 24:00:00.000 的毫秒）	DInt, Int, SInt, UDInt, UInt, USInt, TIME, DTL 仅 SCL： Byte, Word, Dword

说明

使用 T_CONV 将较大的数据大小转换为较小的数据大小

将含较多字节的较大数据类型转换为含较少字节的较小数据类型时，可以截取数据值。如果发生该错误，会将 ENO 置 0。

转换为/转换自 DTL 数据类型

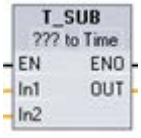
DTL（时间和日期长型）包括年、月、日和时间数据。DTL 数据可转换为/转换自 DATE 和 TOD 数据类型。
但是，使用 DATE 数据转换的 DTL 仅会影响年、月、日的值。使用 TOD 数据转换的 DTL 仅会影响时、分、秒的值。

将 T_CONV 转换为 DTL 时，DTL 格式中未受影响的数据元素将保持不变。

表格 9-3　T_ADD（时间相加）和 T_SUB（时间相减）指令

LAD/FBD	SCL	说明

	`out := T_ADD(`
	`in1:=_variant_in,`
	`in2:=_time_in);`
		T_ADD 将输入 IN1 的值（DTL 或 Time 数据类型）与输入 IN2 的 Time 值相加。参数 OUT 提供 DTL 或 Time 值结果。允许以下两种数据类型的运算：
		• Time + Time = Time
		• DTL + Time = DTL

	`out := T_SUB(`
	`in1:=_variant_in,`
	`in2:=_time_in);`
		T_SUB 从 IN1（DTL 或 Time 值）中减去 IN2 的 Time 值。参数 OUT 以 DTL 或 Time 数据类型提供差值。可进行两种数据类型操作。
		• Time - Time = Time
		• DTL - Time = DTL

[1] 对于 LAD 和 FBD：单击“???”并从下拉菜单中选择数据类型。

表格 9-4　T_ADD 和 T_SUB 参数的数据类型

参数和类型		数据类型	说明
IN1[1]	IN	DTL，Time	DTL 或 Time 值
IN2	IN	Time	要加上或减去的 Time 值
OUT	OUT	DTL，Time	DTL 或 Time 的和值或差值

[1] 从指令名称下方提供的下拉列表中选择 IN1 的数据类型。所选的 IN1 数据类型同时也会设置参数 OUT 的数据类型。

表格 9-5　T_DIFF（时差）指令

LAD/FBD	SCL	说明

	`out := T_DIFF(`
	`in1:=_DTL_in,`
	`in2:=_DTL_in);`
		T_DIFF 从 DTL 值 (IN1) 中减去 DTL 值 (IN2)。参数 OUT 以 Time 数据类型提供差值。
		• DTL - DTL = Time

表格 9-6　T_DIFF 参数的数据类型

参数和类型		数据类型	说明
IN1	IN	DTL	DTL 值
IN2	IN	DTL	要减去的 DTL 值
OUT	OUT	Time	Time 差

条件代码：ENO = 1 代表未发生错误。ENO = 0 和参数 OUT = 0 错误：

- DTL 值无效
- Time 值无效

表格 9-7　T_COMBINE （组合时间）指令

LAD/FBD	SCL	说明

	`out :=`
	`CONCAT_DATE_TOD(`
	`In1 := _date_in,`
	`In2 := _tod_in);`
		T_COMBINE 将 Date 值和 Time_of_Day 值组合在一起生成 DTL 值。

[1] 请注意，在扩展指令中，T_COMBINE 指令相当于 SCL 中的 CONCAT_DATE_TOD 函数。

表格 9-8　T_COMBINE 参数的数据类型

参数和类型		数据类型	说明
IN1	IN	Date	要组合的 Date 值必须在 DATE#1990-01-01 和 DATE#2089-12-31 之间
IN2	IN	Time_of_Day	要组合的 Time_of_Day 值
OUT	OUT	DTL	DTL 值

9.1.2　时钟功能

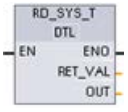
> ⚠ 警告
>
> **存在攻击者通过网络时间协议 (Network Time Protocol, NTP) 同步访问用户网络的风险**
>
> 如果攻击者能通过网络时间协议 (NTP) 同步访问用户网络，那么便可能通过改变 CPU 系统时间来中断过程控制。过程控制中断可能造成死亡、重伤或财产损失。
>
> 默认情况下，S7-1200 CPU 的 NTP 客户端功能处于禁用状态，启用该功能时，仅允许将已组态的 IP 地址用作 NTP 服务器。CPU 默认禁用此功能，必须组态此功能才能实现远程控制 CPU 系统时间修正。
>
> S7-1200 CPU 支持“日时钟”中断和时钟指令，这两个指令均依赖于精确的 CPU 系统时间。如果组态 NTP 并接受从服务器进行时间同步，那么必须确保服务器是可靠来源。否则会导致安全漏洞，从而使未知用户能够通过改变 CPU 系统时间来中断过程控制。
>
> 有关安全信息和建议，请参见 Siemens 服务与支持网站上的“工业安全操作准则 (http://www.industry.siemens.com/topics/global/en/industrial-security/Documents/operational_guidelines_industrial_security_en.pdf)”。

时钟指令用于设置和读取 CPU 系统时钟。使用数据类型 DTL (页 124) 提供日期和时间值。

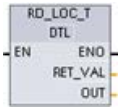
表格 9-9　系统时间指令

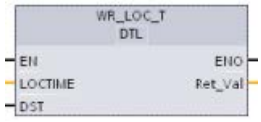
LAD/FBD	SCL	说明

	`ret_val := WR_SYS_T(`
	
	`in:=_DTL_in_);`
		WR_SYS_T（设置时钟）使用参数 IN 中的 DTL 值设置 CPU 时钟。该时间值不包括本地时区或夏令时偏移量。

	`ret_val := RD_SYS_T(`
	
	`out=>_DTL_out);`
		RD_SYS_T（读取时间）从 CPU 中读取当前系统时间。该时间值不包括本地时区或夏令时偏移量。

LAD/FBD	SCL	说明

	`ret_val := RD_LOC_T(`
	
	`out=>_DTL_out);`
		RD_LOC_T（读取本地时间）以 DTL 数据类型提供 CPU 的当前本地时间。该时间值反映了就夏令时（如果已经组态）进行过适当调整的本地时区。

	`ret_val := WR_LOC_T(`
	
	`LOCTIME:=DTL_in_,`
	`DST:_in_;`
		WR_LOC_T（写入本地时间）设置 CPU 时钟的日期与时间。您可使用 DTL 数据类型在 LOCTIME 中将日期和时间信息指定为本地时间。该指令使用 “TimeTransformationRule (页 343)”数据块结构计算系统时间。本地时间和系统时间的时间信息间隔特定于产品并且至少为一毫秒。如果 LOCTIME 参数的输入值小于 CPU 支持的输入值，则这些值在系统时间计算期间将进位。
		注：必须使用 CPU 设备组态设置 “时钟”(Time of day) 属性（时区、DST 激活、DST 启动和 DST 停止）。否则，WR_LOC_T 不能解释 DST 时间更改。

表格 9-10　参数的数据类型

参数和类型		数据类型	说明
IN	IN	DTL	要在 CPU 系统时钟内设置的时间
OUT	OUT	DTL	RD_SYS_T：当前 CPU 系统时间 RD_LOC_T：当前本地时间，包括任何对夏令时的调整（如组态）
LOCTIME	IN	DTL	WR_LOC_T：本地时间
DST	IN	BOOL	WR_LOC_T：Daylight Saving Time 仅在 “双重小时值” 期间时钟更改为夏令时才进行求值。 • TRUE = 夏令时（第一个小时） • FALSE = 标准时间（第二个小时）
RET_VAL	OUT	Int	执行条件代码

- 通过使用用户在设备组态常规选项卡 “时间”(Time of day) 参数中设置的时区和夏令时偏移量计算本地时间。
- 时区组态是相对于 UTC 或 GMT 时间的偏移量。
- 夏令时组态指定夏令时开始时的月份、星期、日期和小时。

- 标准时间组态也会指定标准时间开始时的月份、星期、日期和小时。
- 时区偏移量始终会应用到系统时间值。只有在夏令时有效时才会应用夏令时偏移量。

说明

夏令时和标准起始时间组态

CPU 设备组态的“夏令时开始”(Start for daylight saving time) 的“时间”(Time of day) 属性必须是本地时间。

条件代码：ENO = 1 表示未发生错误。ENO = 0 表示发生了执行错误，同时在 RET_VAL 输出中提供条件代码。

RET_VAL (W#16#....)	说明
0000	当前的本地时间为标准时间。
0001	夏令制时间已组态，当前的本地时间为夏令制时间。
8080	本地时间不可用或 LOCTIME 值无效。
8081	年份值非法或 LOCTIME 参数分配的时间值无效
8082	月份值非法（DTL 格式中的字节 2）
8083	日期值非法（DTL 格式中的字节 3）
8084	小时值非法（DTL 格式中的字节 5）
8085	分钟值非法（DTL 格式中的字节 6）
8086	秒数值非法（DTL 格式中的字节 7）
8087	纳秒值非法（DTL 格式中的字节 8 到 11）
8089	时间值不存在（转换为夏令时时，小时已过）
80B0	实时时钟发生了故障
80B1	尚未定义“TimeTransformationRule”结构。

9.1.3 TimeTransformationRule 数据结构

说明

标准时间与夏令时之间的转换规则在 TimeTransformationRule 结构中定义。 结构如下：

名称		数据类型	说明
TimeTransformationRule		STRUCT	
	Bias	INT	本地时间与 UTC 的时差 [min] 范围： -1439 到 1439
	DaylightBias	INT	夏令时与标准时间的时差 [min] 范围： 0 到 60
	DaylightStartMonth	USINT	转换为夏令时的月份 范围： 1 到 12
	DaylightStartWeek	USINT	转换为夏令时的星期 1 = 该月的第一周，...， 5 = 该月的最后一周
	DaylightStartWeekday	USINT	夏令时转换的周几： 1 = 星期日
	DaylightStartHour	USINT	夏令时转换的小时： 范围： 0 到 23
	DaylightStartMinute	USINT	夏令时转换的分钟 范围： 0 到 59
	StandardStartMonth	USINT	转换为标准时间的月份 范围： 1 到 12
	StandardStartWeek	USINT	转换为标准时间的星期 1 = 该月的第一周，...， 5 = 该月的最后一周
	StandardStartWeekday	USINT	标准时间转换的周几： 1 = 星期日
	StandardStartHour	USINT	标准时间转换的小时 范围： 0 到 23

名称		数据类型	说明
	StandardStartMinute	USINT	标准时间转换的分钟 范围： 0 到 59
	TimeZoneName	STRING[80]	时区名称：“(GMT+01:00) 柏林、伯尔尼、布鲁塞尔、罗马、斯德哥尔摩和越南”

9.1.4 SET_TIMEZONE（设置时区）

表格 9-11 SET_TIMEZONE 指令

LAD/FBD	SCL	说明
	"SET_TIMEZONE_DB"(REQ:=_bool_in, Timezone:=_struct_in, DONE=>_bool_out_, BUSY=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_);	设置本地时区和夏令时参数，以用于将 CPU 系统时间转换为本地时间。

[1] 在 SCL 示例中，“SET_TIMEZONE_DB”是背景 DB 的名称。

表格 9-12 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	REQ=1：执行功能
Timezone	IN	TimeTransformationRule	将系统时间转换为本地时间的规则
DONE	OUT	Bool	功能执行完毕
BUSY	OUT	Bool	功能忙
ERROR	OUT	Bool	检测到错误
STATUS	OUT	Word	功能结果/错误消息

要手动组态 CPU 的时区参数，请使用设备组态“常规”(General) 选项卡中的“时间”(Time of day) 属性。

使用 SET_TIMEZONE 指令设置本地时间组态。 “TimeTransformationRule (页 343)”结构的参数用于分配本地时区以及在标准时间和夏令时之间自动切换的时间。

说明

SET_TIMEZONE 指令对闪存的影响

SET_TIMEZONE 指令会在闪存（内部装载存储器或存储卡）内执行写入操作。为了避免缩短闪存的使用寿命，可以使用 SET_TIMEZONE 指令进行少量更新。

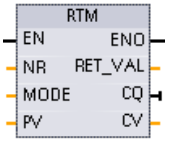
条件代码：ENO = 1 表示未发生错误。ENO = 0 表示发生了执行错误，同时在 STATUS 输出中提供条件代码。

STATUS (W#16#....)	说明
0	无错误
7000	无激活的作业处理
7001	开始处理作业。参数 BUSY = 1，DONE = 0
7002	中间调用（与 REQ 无关）：指令已激活，BUSY 的值为“1”。
808x	第 x 个组件出错：例如，8084 表明 DaylightStartWeekif 不是从 1 到 5 的值。

9.1.5 RTM（运行时间计时器）

表格 9-13 RTM 指令

LAD/FBD	SCL	说明

	`RTM(NR:=_uint_in_,`
	`MODE:=_byte_in_,`
	`PV:=_dint_in_,`
	`CQ=>_bool_out_,`
	`CV=>_dint_out_);`
		RTM（运行时间计时器）指令可以设置、启动、停止和读取 CPU 中的运行时间小时计时器。

表格 9-14　参数的数据类型

参数和类型		数据类型	说明
NR	IN	UInt	运行时间计时器编号：（取值范围：0..9）
MODE	IN	Byte	RTM 执行模式编号： ● 0 = 获取值（然后状态值写入 CQ，当前值写入 CV） ● 1 = 启动（从上一计数值开始） ● 2 = 停止 ● 4 = 设置（设为 PV 中指定的值） ● 5 = 设置（设为 PV 中指定的值），然后启动 ● 6 = 设置（设为 PV 中指定的值），然后停止 ● 7 = 将 CPU 中的所有 RTM 值保存到 MC（存储卡）
PV	IN	DInt	指定运行时间计时器的预设小时值
RET_VAL	OUT	Int	功能结果/错误消息
CQ	OUT	Bool	运行时间计时器的状态（1 = 正在运行）
CV	OUT	DInt	指定计时器的当前运行小时值

CPU 最多可运行 10 个运行小时计时器来跟踪关键控制子系统的运行小时数。必须对每个定时器执行一次 RTM 分别启动小时计时器。CPU 从运行模式切换为停止模式时，所有运行小时计时器都将停止。还可以使用 RTM 执行模式 2 停止各个的定时器。

CPU 从停止模式切换为运行模式时，必须对每个已启动的定时器执行一次 RTM 来重新启动小时计时器。运行时间计时器值大于 2147483647 小时后，将停止计时并发出“上溢”错误。必须为每个定时器执行一次 RTM 指令，以复位或修改定时器。

CPU 电源故障或循环上电会导致将当前运行时间计时器值保存在保持性存储器中的断电过程。在 CPU 上电时，所存储的运行时间计时器值将重新加载到定时器，之前的运行时间小时总数不会丢失。必须重启运行时间计时器才能累加额外的运行时间。

用户程序还可以使用 RTM 执行模式 7 将运行时间计时器值保存在存储卡中。执行 RTM 模式 7 时的所有定时器的状态将保存在存储卡中。由于小时定时器会在程序运行过程中或启动或停止，随着时间的推移，这些存储值就可能出错。因此，必须周期性更新存储卡值，以捕获重要的运行事件。在存储卡中保存 RTM 值的好处是，在替代 CPU 中插入存储卡时，就可以在其中使用程序和所保存的 RTM 值。如果未将 RTM 值保存在存储卡中，则会丢失定时器值（在替代 CPU 中）。

说明

避免过度调用执行存储卡写操作的程序

尽可能减少闪存卡写操作，以延长存储卡的使用寿命。

表格 9-15　条件代码

RET_VAL (W#16#....)	说明
0	无错误
8080	运行时间定时器编号错误
8081	负值已传递给参数 PV
8082	操作小时计数器溢出
8091	输入参数 MODE 包含非法值
80B1	无法将值保存到 MC (MODE=7)

9.2 字符串和字符

9.2.1 String 数据概述

字符串数据类型

String 数据被存储成 2 个字节的标头后跟最多 254 个 ASCII 码字符组成的字符字节。 String 标头包含两个长度。 第一个字节是初始化字符串时方括号中给出的最大长度，默认值为 254。 第二个标头字节是当前长度，即字符串中的有效字符数。 当前长度必须小于或等于最大长度。 String 格式占用的存储字节数比最大长度大 2 个字节。

初始化 String 数据

在执行任何字符串指令之前，必须将 String 输入和输出数据初始化为存储器中的有效字符串。

有效 String 数据

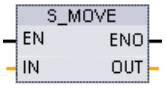
有效字符串的最大长度必须大于 0 但小于 255。当前长度必须小于等于最大长度。

字符串无法分配给 I 或 Q 存储区。

有关详细信息，请参见： String 数据类型的格式 (页 126)。

9.2.2 S_MOVE（移动字符串）

表格 9-16 字符串移动指令

LAD/FBD	SCL	说明
	`out := in;`	将源 IN 字符串复制到 OUT 位置。S_MOVE 的执行并不影响源字符串的内容。

表格 9-17 参数的数据类型

参数	数据类型	说明
IN	String	源字符串
OUT	String	目标地址

如果输入 IN 中字符串的实际长度超过输出 OUT 存储的字符串最大长度，则会复制 OUT 字符串能容纳的部分 IN 字符串。

9.2.3 字符串转换指令

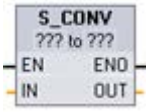
9.2.3.1 S_CONV、STRG_VAL 和 VAL_STRG（在字符串与数值之间转换）指令

可以使用以下指令将数字字符串转换为数值或将数值转换为数字字符串：

- S_CONV 用于将数字字符串转换成数值或将数值转换成数字字符串
- STRG_VAL 使用格式选项将数字字符串转换成数值
- VAL_STRG 使用格式选项将数值转换成数字字符串

S_CONV（转换字符串）

表格 9-18　字符串转换指令

LAD/FBD	SCL	说明
	`out := <Type>_TO_<Type>(in);`	将字符串转换成相应的值，或将值转换成相应的字符串。S_CONV 指令没有输出格式选项。因此，S_CONV 指令比 STRG_VAL 指令和 VAL_STRG 指令更简单，但灵活性更差。

[1] 对于 LAD/FBD：单击“???”并从下拉列表中选择数据类型。

[2] 对于 SCL：从扩展指令中选择 S_CONV，然后应答数据类型转换的提示信息。STEP 7 随后会显示相应的转换指令。

表格 9-19　数据类型（字符串到值）

参数和类型		数据类型	说明
IN	IN	String, WString	输入字符串
OUT	OUT	String, WString, Char, WChar, SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	输出数值

字符串参数 IN 的转换从首个字符开始，并一直进行到字符串的结尾，或者一直进行到遇到第一个不是“0”到“9”、“+”、“-”或“.”的字符为止。结果值将在参数 OUT 中指定的位置提供。如果输出数值不在 OUT 数据类型的范围内，则参数 OUT 设置为 0，并且 ENO 设置为 FALSE。否则，参数 OUT 将包含有效的结果，并且 ENO 设置为 TRUE。

输入 String 格式规则：

- 如果在 IN 字符串中使用小数点，则必须使用“.”字符。
- 允许使用逗点字符“,”作为小数点左侧的千位分隔符，并且逗点字符会被忽略。
- 忽略前导空格。

S_CONV（值到字符串的转换）

表格 9-20　数据类型（值到字符串）

参数和类型		数据类型	说明
IN	IN	String, WString, Char, WChar, SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	输入数值
OUT	OUT	String, WString	输出字符串

整数值、无符号整数值或浮点值 IN 在 OUT 中被转换为相应的字符串。在执行转换前，参数 OUT 必须引用有效字符串。有效字符串由第一个字节中的最大字符串长度、第二个字节中的当前字符串长度以及后面字节中的当前字符串字符组成。转换后的字符串将从第一个字符开始替换 OUT 字符串中的字符，并调整 OUT 字符串的当前长度字节。OUT 字符串的最大长度字节不变。

被替换的字符数取决于参数 IN 的数据类型和数值。被替换的字符数必须在参数 OUT 的字符串长度范围内。OUT 字符串的最大字符串长度（第一个字节）应大于或等于被转换字符的最大预期数目。下表显示了 S_CONV 值到字符串的转换示例：

输出 String 格式规则：

- 写入到参数 OUT 的值不使用前导“+”号。
- 使用定点表示法（不可使用指数表示法）。
- 参数 IN 为 Real 数据类型时，使用句点字符“.”表示小数点。
- 输出字符串中的值为右对齐并且值的前面有填有空字符位置的空格字符。

表格 9-21　每种数据类型的最大字符串长度

IN 数据类型	S_CONV 分配的字符位置	转换的字符串示例 [1]	包括最大及当前长度字节在内的总字符串长度
USInt	4	“x255”	6
SInt	4	“-128”	6
UInt	6	“x65535”	8
Int	6	“-32768”	8
UDInt	11	“x4294967295”	13
DInt	11	“-2147483648”	13

IN 数据类型	S_CONV 分配的字符位置	转换的字符串示例 [1]	包括最大及当前长度字节在内的总字符串长度
Real	14	“x-3.402823E+38” “x-1.175495E-38” “x+1.175495E-38” “x+3.402823E+38”	16
LReal	21	"-1.7976931348623E+308" "-2.2250738585072E-308" "+2.2250738585072E-308" "+1.7976931348623E+308"	23

[1] “x”字符代表用于填写分配给转换值的右对齐字段中空位置的空格字符。

STRG_VAL（将字符串转换为数值）

表格 9-22　字符串转换成值指令

LAD/FBD	SCL	说明

	`"STRG_VAL"(`
	`in:=_string_in,`
	`format:=_word_in,`
	`p:=uint_in,`
	`out=>_variant_out);`
		将数字字符串转换为相应的整型或浮点型表示法。

[1] 对于 LAD/FBD：单击“???”并从下拉列表中选择数据类型。

表格 9-23　STRG_VAL 指令的数据类型

参数和类型		数据类型	说明
IN	IN	String, WString	要转换的 ASCII 字符串
FORMAT	IN	Word	输出格式选项
P	IN	UInt, Byte, USInt	IN：指向要转换的第一个字符的索引（第一个字符 = 1）
OUT	OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	转换后的数值

转换从字符串 IN 中的字符偏移量 P 位置开始，并一直进行到字符串的结尾，或者一直进行到遇到第一个不是“+”、“-”、“.”、“,”、“e”、“E”或“0”到“9”的字符为止。结果放置在参数 OUT 中指定的位置。

必须在执行前将 String 数据初始化为存储器中的有效字符串。

以下定义了 STRG_VAL 指令的 FORMAT 参数。未使用的位位置必须设置为零。

表格 9-24　STRG_VAL 指令的格式

位 16							位 8	位 7							位 0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	f	r

f = 表示法格式　　1= 指数表示法
　　0 = 定点表示法

r = 小数点格式　　1 =“,”（逗号字符）
　　0 =“.”（周期字符）

表格 9-25　FORMAT 参数的值

FORMAT (W#16#)	表示法格式	小数点表示法
0000（默认）	定点	"."
0001		","
0002	指数	"."
0003		","
0004 到 FFFF	非法值	

STRG_VAL 转换的规则：

- 如果使用句点字符“.”作为小数点，则小数点左侧的逗点“,”将被解释为千位分隔符字符。允许使用逗点字符并且会将其忽略。
- 如果使用逗点字符“,”作为小数点，则小数点左侧的句点“.”将被解释为千位分隔符字符。允许使用句点字符并且会将其忽略。
- 忽略前导空格。

VAL_STRG（将数值转换为字符串）

表格 9-26　值转换成字符串的运算

LAD/FBD	SCL	说明

	`"VAL_STRG"(`
	`in:=_variant_in,`
	`size:=_usint_in,`
	`prec:=_usint_in,`
	`format:=_word_in,`
	`p:=uint_in,`
	`out=>_string_out);`
		将整数值、无符号整数值或浮点值转换为相应的字符串表示法。

[1] 对于 LAD/FBD：单击“???”并从下拉列表中选择数据类型。

表格 9-27　VAL_STRG 指令的数据类型

参数和类型		数据类型	说明
IN	IN	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	要转换的值
SIZE	IN	USInt	要写入 OUT 字符串的字符数
PREC	IN	USInt	小数部分的精度或大小。不包括小数点。
FORMAT	IN	Word	输出格式选项
P	IN	UInt, Byte, USInt	IN：指向要替换的第一个 OUT 字符串字符的索引（第一个字符 = 1）
OUT	OUT	String, WString	转换后的字符串

此指令用于将参数 IN 表示的值转换为参数 OUT 所引用的字符串。在执行转换前，参数 OUT 必须为有效字符串。

转换后的字符串从字符偏移量计数 P 位置开始替换 OUT 字符串中的字符，一直到参数 SIZE 指定的字符数。SIZE 中的字符数必须在 OUT 字符串长度范围内（从字符位置 P 开始计数）。如果 SIZE 参数为零，则字符将覆盖字符串 OUT 中 P 位置的字符，且没有任何限制。该指令对于将数字字符嵌入到文本字符串中很有用。例如，可以将数字“120”放入字符串“Pump pressure = 120 psi”中。

参数 PREC 用于指定字符串中小数部分的精度或位数。如果参数 IN 的值为整数，则 PREC 指定小数点的位置。例如，如果数据值为 123 且 PREC = 1，则结果为“12.3”。对于 Real 数据类型，支持的最大精度为 7 位。

如果参数 P 大于 OUT 字符串的当前大小，则会添加空格，一直到位置 P，并将该结果附加到字符串末尾。如果达到了最大 OUT 字符串长度，则转换结束。

以下定义了 VAL_STRG 指令的 FORMAT 参数。未使用的位位置必须设置为零。

表格 9-28 VAL_STRG 指令的格式

位 16							位 8	位 7							位 0
0	0	0	0	0	0	0	0	0	0	0	0	0	s	f	r

s = 数字符号字符　　1= 使用符号字符“+”和“-”
　　0 = 仅使用符号字符“-”

f = 表示法格式　　1= 指数表示法
　　0 = 定点表示法

r = 小数点格式　　1 =“,”（逗号字符）
　　0 =“.”（周期字符）

表格 9-29 FORMAT 参数的值

FORMAT (WORD)	数字符号字符	表示法格式	小数点表示法
W#16#0000	仅“-”	定点	"."
W#16#0001			","
W#16#0002		指数	"."
W#16#0003			","
W#16#0004	“+”和“-”	定点	"."
W#16#0005			","
W#16#0006		指数	"."
W#16#0007			","
W#16#0008 到 W#16#FFFF	非法值		

参数 OUT 字符串的格式规则如下：

- 如果转换后的字符串小于指定的大小，则会在字符串的最左侧添加前导空格字符。
- 如果 FORMAT 参数的符号位为 FALSE，则会将无符号和有符号整型值写入输出缓冲区，且不带前导“+”号。必要时会使用“-”号。
  <前导空格><无前导零的数字>'.'<PREC 数字>
- 如果符号位为 TRUE，则会将无符号和有符号整型值写入输出缓冲区，且总是带前导符号字符。
  <前导空格><符号><不带前导零的数字>'.'<PREC 数字>
- 如果 FORMAT 被设置为指数表示法，则会按以下方式将 Real 数据类型的值写入输出缓冲区：
  <前导空格><符号><数字> '.' <PREC 数字>'E' <符号><不带前导零的数字>
- 如果 FORMAT 被设置为定点表示法，则会按以下方式将整型、无符号整型和实型值写入输出缓冲区：
  <前导空格><符号><不带前导零的数字>'.'<PREC 数字>
- 小数点左侧的前导零会被隐藏，但与小数点相邻的数字除外。
- 小数点右侧的值被舍入为 PREC 参数所指定的小数点右侧的位数。
- 输出字符串的大小必须比小数点右侧的位数多至少三个字节。
- 输出字符串中的值为右对齐。

ENO 报告的条件

当转换操作遇到错误，指令返回下列结果：

- ENO 设置为 0。
- OUT 设置为 0，或者如字符串到值的转换示例中所示。
- OUT 不变，或者如 OUT 为字符串时的示例中所示。

表格 9-30　ENO 状态

ENO	说明
1	无错误
0	非法或无效参数；例如，访问一个不存在的 DB
0	非法字符串，要求该字符串的最大长度为 0 或 255
0	非法字符串，当前长度大于最大长度
0	转换后的数值对于指定的 OUT 数据类型而言过大。

ENO	说明
0	OUT 参数的最大字符串大小必须足够大，以接受参数 SIZE 所指定的字符数（从字符位置参数 P 开始）。
0	非法 P 值，P=0 或 P 大于当前字符串长度
0	参数 SIZE 必须大于参数 PREC。

表格 9-31　S_CONV 字符串到值的转换示例

IN 字符串	OUT 数据类型	OUT 值	ENO
“123”	Int 或 DInt	123	TRUE
“-00456”	Int 或 DInt	-456	TRUE
“123.45”	Int 或 DInt	123	TRUE
“+2345”	Int 或 DInt	2345	TRUE
"00123AB"	Int 或 DInt	123	TRUE
“123”	Real	123.0	TRUE
“123.45”	Real	123.45	TRUE
"1.23e-4"	Real	1.23	TRUE
"1.23E-4"	Real	1.23	TRUE
“12345.67”	Real	12345.67	TRUE
"3.4e39"	Real	3.4	TRUE
"-3.4e39"	Real	-3.4	TRUE
"1.17549e-38"	Real	1.17549	TRUE
“12345”	SInt	0	FALSE
"A123"	不适用	0	FALSE
""	不适用	0	FALSE
“++123”	不适用	0	FALSE
“+-123”	不适用	0	FALSE

表格 9-32　S_CONV 值到字符串的转换示例

数据类型	IN 值	OUT 字符串 [1]	ENO
UInt	123	“xxx123”	TRUE
UInt	0	“xxxxx0”	TRUE

数据类型	IN 值	OUT 字符串 [1]	ENO
UDInt	12345678	“xxx12345678”	TRUE
Real	+9123.456	“xx+9.123456E+3”	TRUE
LReal	+9123.4567890123	“xx +9.1234567890123 E+3”	TRUE
Real	-INF	“xxxxxxxxxxxxINF”	FALSE
Real	+INF	“xxxxxxxxxxxxINF”	FALSE
Real	NaN	“xxxxxxxxxxxxNaN”	FALSE

[1] “x”字符代表用于填写分配给转换值的右对齐字段中空位置的空格字符。

表格 9-33　示例：STRG_VAL 转换

IN 字符串	FORMAT (W#16#....)	OUT 数据类型	OUT 值	ENO
“123”	0000	Int 或 DInt	123	TRUE
“-00456”	0000	Int 或 DInt	-456	TRUE
“123.45”	0000	Int 或 DInt	123	TRUE
“+2345”	0000	Int 或 DInt	2345	TRUE
"00123AB"	0000	Int 或 DInt	123	TRUE
“123”	0000	Real	123.0	TRUE
“-00456”	0001	Real	-456.0	TRUE
“+00456”	0001	Real	456.0	TRUE
“123.45”	0000	Real	123.45	TRUE
“123.45”	0001	Real	12345.0	TRUE
“123.45”	0000	Real	12345.0	TRUE
“123.45”	0001	Real	123.45	TRUE
".00123AB"	0001	Real	123.0	TRUE
"1.23e-4"	0000	Real	1.23	TRUE
"1.23E-4"	0000	Real	1.23	TRUE
"1.23E-4"	0002	Real	1.23E-4	TRUE
“12345.67”	0000	Real	12345.67	TRUE
“12345.67”	0001	Real	12.345	TRUE

IN 字符串	FORMAT (W#16#....)	OUT 数据类型	OUT 值	ENO
"3.4e39"	0002	Real	+INF	TRUE
"-3.4e39"	0002	Real	-INF	TRUE
"1.1754943e-38"（及更小值）	0002	Real	0.0	TRUE
“12345”	不适用	SInt	0	FALSE
"A123"	不适用	不适用	0	FALSE
""	不适用	不适用	0	FALSE
“++123”	不适用	不适用	0	FALSE
“+-123”	不适用	不适用	0	FALSE

下面的 VAL_STRG 转换示例均基于按以下方式初始化的 OUT 字符串：

“`Current Temp = xxxxxxxxxx C`”
其中字符“`x`”表示为转换后的值分配的空格字符。

表格 9-34　示例：VAL_STRG 转换

数据类型	IN 值	P	SIZE	FORMAT (W#16#....)	PREC	OUT 字符串	ENO
UInt	123	16	10	0000	0	`Current Temp = xxxxxxx123 C`	TRUE
UInt	0	16	10	0000	2	`Current Temp = xxxxxx0.00 C`	TRUE
UDInt	12345678	16	10	0000	3	`Current Temp = x12345.678 C`	TRUE
UDInt	12345678	16	10	0001	3	`Current Temp = x12345,678 C`	TRUE
Int	123	16	10	0004	0	`Current Temp = xxxxxx+123 C`	TRUE
Int	-123	16	10	0004	0	`Current Temp = xxxxxx-123 C`	TRUE
Real	-0.00123	16	10	0004	4	`Current Temp = xxx-0.0012 C`	TRUE
Real	-0.00123	16	10	0006	4	`Current Temp = -1.2300E-3 C`	TRUE

数据类型	IN 值	P	SIZE	FORMAT (W#16#....)	PREC	OUT 字符串	ENO
Real	-INF	16	10	不适用	4	`Current Temp = xxxxxx-INF C`	FALSE
Real	+INF	16	10	不适用	4	`Current Temp = xxxxxx+INF C`	FALSE
Real	NaN	16	10	不适用	4	`Current Temp = xxxxxxxNaN C`	FALSE
UDInt	12345678	16	6	不适用	3	`Current Temp = xxxxxxxxxx C`	FALSE

9.2.3.2 Strg_TO_Chars 和 Chars_TO_Strg（在字符串与字符数组之间转换）指令

Strg_TO_Chars 将 ASCII 字符串复制到字符字节数组中。

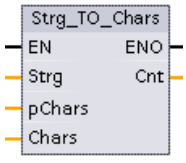
Chars_TO_Strg 将 ASCII 字符字节数组复制到字符串中。

说明

只允许将零基数组类型 (Array [0..n] of Char) 或 (Array [0..n] of Byte) 作为指令 Chars_TO_Strg 的输入参数 Chars，或作为指令 Strg_TO_Chars 的 IN_OUT 参数 Chars 。

表格 9-35 Strg_TO_Chars 指令

LAD/FBD	SCL	说明

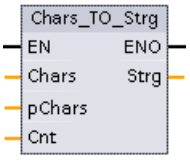
	`Strg_TO_Chars(`
	`Strg:=_string_in_,`
	`pChars:=_dint_in_,`
	`Cnt=>_uint_out_,`
	
	`Chars:=_variant_inout_);`
		将整个输入字符串 Strg 复制到 IN_OUT 参数 Chars. 的字符数组中。
		该操作会从 pChars 参数指定的数组元素编号开始覆盖字节。
		可以使用所有受支持的最大长度 (1..254) 的字符串。
		结束分隔符不会被写入；这由用户负责。要在最后写入的数组字符后面设置结束分隔符，应使用下一数组元素编号 [pChars+Cnt]。

表格 9-36　参数的数据类型 (Strg_TO_Chars)

参数和类型		数据类型	说明
Strg	IN	String, WString	源字符串
pChars	IN	DInt	写入目标数组的第一个字符串字符的数组元素编号
Chars	IN_OUT	Variant	Chars 参数是从输入字符串复制的零基字符数组 [0..n] 的指针。可以在 DB 中声明数组，也可以在块接口中将其声明为本地变量。 示例："DB1".MyArray 指向 DB1 中的 MyArray [0..10] of Char 元素值。
Cnt	OUT	UInt	已复制的字符数

表格 9-37　Chars_TO_Strg 指令

LAD/FBD	SCL	说明

	`Chars_TO_Strg(`
	`Chars:=_variant_in_,`
	`pChars:=_dint_in_,`
	`Cnt:=_uint_in_,`
	`Strg=>_string_out_);`
		将字符数组的全部或一部分复制到字符串。
		执行 Chars_TO_Strg 之前必须声明输出字符串。之后，Chars_TO_Strg 操作会覆盖该字符串。
		可以使用所有受支持的最大长度 (1..254) 的字符串。
		Chars_TO_Strg 操作不会更改字符串的最大长度值。达到最大字符串长度后，将停止从数组复制到字符串。
		字符数组中的 nul 字符“$00”或 16#00 值起分隔符的作用，用于结束向字符串复制字符的操作。

表格 9-38　参数的数据类型 (Chars_TO_Strg)

参数和类型		数据类型	说明
Chars	IN	Variant	Chars 参数是要转换为字符串的零基字符数组 [0..n] 的指针。可以在 DB 中声明数组，也可以在块接口中将其声明为本地变量。 示例："DB1".MyArray 指向 DB1 中的 MyArray [0..10] of Char 元素值。
pChars	IN	Dint	数组中要复制的第一个字符的元素编号。默认值为数组元素 [0]。
Cnt	IN	UInt	要复制的字符数：0 表示全部
Strg	OUT	String, WString	目标字符串

表格 9-39　ENO 状态

ENO	说明
1	无错误
0	Chars_TO_Strg： 尝试将多于字符串声明中最大长度字节允许的字符字节复制到输出字符串中。
0	Chars_TO_Strg： nul 字符 (16#00) 值出现在输入字符字节数组中。
0	Strg_TO_Chars：尝试将多于元素数量限定允许的字符字节复制到输出数组中。

9.2.3.3　ATH 和 HTA（在 ASCII 字符串与十六进制数之间转换）指令

使用 ATH（ASCII 到十六进制）和 HTA（十六进制到 ASCII）指令进行 ASCII 字符字节（仅字符 0 到 9 和大写 A 到 F）与相应的 4 位十六进制半字节之间的转换。

表格 9-40　ATH 指令

LAD/FBD	SCL	说明

	`ret_val := ATH(`
	`in:=_variant_in_,`
	`n:=_int_in_,`
	`out=>_variant_out_);`
		将 ASCII 字符转换为压缩的十六进制数字。

表格 9-41　ATH 指令的数据类型

参数类型		数据类型	说明
IN	IN	Variant	指向 ASCII 字符字节数组的指针
N	IN	UInt	要转换的 ASCII 字符字节数
RET_VAL	OUT	Word	执行条件代码
OUT	OUT	Variant	指向转换后的十六进制字节数组的指针

转换从参数 IN 指定的位置开始，并持续 N 个字节。结果放置在 OUT 指定的位置。只能转换有效的 ASCII 字符 0 到 9、小写 a 到 f 和大写 A 到 F。任何其它字符都将被转换为零。

8 位 ASCII 编码的字符将被转换为 4 位十六进制半字节。可将两个 ASCII 字符转换为一个包含两个 4 位十六进制半字节的字节。

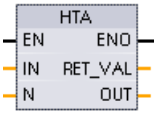
参数 IN 和 OUT 指定的是字节数组而不是十六进制 String 数据。ASCII 字符将被转换，并以其被读取的顺序放置在十六进制输出中。如果 ASCII 字符数为奇数，则在最后一个转换的十六进制数字的最右侧半字节中放置零。

表格 9-42　示例：ASCII 到十六进制 (ATH) 转换

IN 字符字节	N	OUT 值	ENO
'0a23'	4	W#16#0A23	TRUE
'123AFx1a23'	10	16#123AF01023	FALSE
'a23'	3	W#16#A230	TRUE

表格 9-43　HTA 指令

LAD/FBD	SCL	说明

	`ret_val := HTA(`
	`in:=_variant_in_,`
	`n:=_uint_in_,`
	`out=>_variant_out_);`
		将压缩的十六进制数字转换为相应的 ASCII 字符字节。

表格 9-44　HTA 指令的数据类型

参数和类型		数据类型	说明
IN	IN	Variant	指向输入字节数组的指针
N	IN	UInt	要转换的字节数（每个输入字节有两个 4 位半字节，并会生成 2N 个 ASCII 字符）
RET_VAL	OUT	Word	执行条件代码
OUT	OUT	Variant	指向 ASCII 字符字节数组的指针

转换从参数 IN 指定的位置开始，并持续 N 个字节。每个 4 位半字节都会转换为单个 8 位 ASCII 字符，并会生成 2N 个 ASCII 字符输出字节。全部 2N 个输出字节都会被写为 ASCII 字符 0 到 9 以及大写的 A 到 F。参数 OUT 指定一个字节数组，而不是字符串。

十六进制字节的每个半字节将按其读入的顺序转换为一个字符（首先转换十六进制数字最左侧的半字节，然后转换该字节最右侧的半字节）。

表格 9-45　示例：十六进制到 ASCII (HTA) 转换

IN 值	N	OUT 字符字节	ENO（执行 HTA 之后，ENO 始终为 TRUE）
W#16#0123	2	'0123'	TRUE
DW#16#123AF012	4	'123AF012'	TRUE

表格 9-46　ATH and HTA 条件代码

RET_VAL (W#16#....)	说明	ENO
0000	无错误	TRUE
0007	无效的 ATH 输入字符：发现不属于 ASCII 字符 0-9、小写 a 到 f 和大写 A 到 F 的字符	FALSE
8101	非法或无效的输入指针，例如，访问一个不存在的数据块。	FALSE
8120	输入字符串的格式无效，即，最大值 = 0、最大值 =255、当前值 > 最大值或允许的指针长度 < 最大值	FALSE
8182	输入缓冲区对于 N 来说过小	FALSE
8151	数据类型不允许用于输入缓冲区	FALSE
8301	非法或无效的输出指针，例如，访问一个不存在的数据块。	FALSE
8320	输出字符串的格式无效，即，最大值 = 0、最大值 =255、当前值 > 最大值或允许的指针长度 < 最大值	FALSE
8382	输出缓冲区对于 N 来说过小	FALSE
8351	数据类型不允许用于输出缓冲区	FALSE

9.2.4　字符串操作指令

控制程序可以使用以下字符串和字符指令为操作员显示和过程日志创建消息。

9.2.4.1 MAX_LEN（字符串的最大长度）

表格 9-47 最大长度指令

LAD/FBD	SCL	说明
	`out := MAX_LEN(in);`	MAX_LEN（字符串的最大长度）提供了在输出 OUT 中分配给字符串 IN 的最大长度值。如果处理指令期间出错，则输出空字符串长度。 String 和 WString 数据类型包含两个长度：第一个字节（或字）指定最大长度，第二个字节（或字）指定当前长度（当前有效字符的数量）。 ● 在方括号中指定每个 String 或 WString 声明的字符串最大长度。String 占用的字节数超过最大长度 2 个字节。WString 占用的字数超过最大长度 2 个字。 ● 当前长度表示实际使用的字符数。当前长度必须小于或等于最大长度。对于 String，当前长度以字节为单位，对于 WString，当前长度以字为单位。 使用 MAX_LEN i 指令获取字符串的最大长度，使用 LEN 指令获取字符串的当前长度。

表格 9-48 参数的数据类型

参数和类型		数据类型	说明
IN	IN	String, WString	输入字符串
OUT	OUT	DInt	IN 字符串允许的最大字符数

9.2.4.2 LEN（确定字符串的长度）

表格 9-49 长度指令

LAD/FBD	SCL	说明
	`out := LEN(in);`	LEN（长度）提供输出 OUT 处的字符串 IN 的当前长度。空字符串的长度为零。

表格 9-50　参数的数据类型

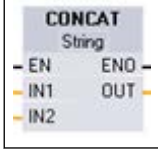
参数和类型		数据类型	说明
IN	IN	String, WString	输入字符串
OUT	OUT	Int, DInt, Real, LReal	IN 字符串的有效字符数

表格 9-51　ENO 状态

ENO	条件	OUT
1	没有无效字符串条件	有效字符串长度
0	IN 的当前长度超出 IN 的最大长度	当前长度被设置为 0
	IN 的最大长度不在分配的存储范围内	
	IN 的最大长度为 255（非法长度）	

9.2.4.3　CONCAT（合并字符串）

表格 9-52　连接字符串指令

LAD/FBD	SCL	说明
	`out := CONCAT(in1, in2);`	CONCAT（连接字符串）将字符串参数 IN1 和 IN2 连接成一个字符串，并在 OUT 输出。连接后，字符串 IN1 是组合字符串的左侧部分，而 IN2 是其右侧部分。

表格 9-53　参数的数据类型

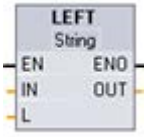
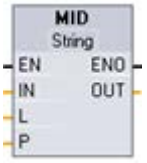
参数和类型		数据类型	说明
IN1	IN	String, WString	输入字符串 1
IN2	IN	String, WString	输入字符串 2
OUT	OUT	String, WString	组合字符串（字符串 1 + 字符串 2）

表格 9-54　ENO 状态

ENO	条件	OUT
1	未检测到错误	有效字符
0	连接后的结果字符串比 OUT 字符串的最大长度长	复制结果字符串字符直到达到 OUT 的最大长度为止
	IN1 的当前长度超出 IN1 的最大长度，IN2 的当前长度超出 IN2 的最大长度，或 OUT 的当前长度超出 OUT 的最大长度（无效字符串）	当前长度被设置为 0
	IN1、IN2 或 OUT 的最大长度不在分配的存储范围内	
	IN1 或 IN2 的最大长度为 255，或者 OUT 的最大长度为 0 或 255（String 数据类型）	
	IN1 或 IN2 的最大长度为 65534，或者 OUT 的最大长度为 0 或 65534（WString 数据类型）	

9.2.4.4　LEFT、RIGHT 和 MID（读取字符串中的子串）指令

表格 9-55　左侧、右侧和中间子串操作

LAD/FBD	SCL	说明
	`out := LEFT(in, L);`	LEFT（左侧子串）提供由字符串参数 IN 的前 L 个字符所组成的子串。 ● 如果 L 大于 IN 字符串的当前长度，则在 OUT 中返回整个 IN 字符串。 ● 如果输入是空字符串，则在 OUT 中返回空字符串。
	`out := MID(in, L, p);`	MID（中间子串）提供字符串的中间部分。中间子串为 L 个字符长，并从字符位置 P（包括 P）开始算起 如果 L 和 P 的和超出字符串参数 IN 的当前长度，则返回从字符位置 P 开始并一直到 IN 字符串结尾的子串。
	`out := RIGHT(in, L);`	RIGHT（右侧子串）提供字符串的最后 L 个字符。 ● 如果 L 大于 IN 字符串的当前长度，则在参数 OUT 中返回整个 IN 字符串。 ● 如果输入是空字符串，则在 OUT 中返回空字符串。

表格 9-56　参数的数据类型

参数和类型		数据类型	说明
IN	IN	String, WString	输入字符串
L	IN	Int	要创建的子串的长度： • LEFT 使用字符串最左侧的字符数 • RIGHT 使用字符串最右侧的字符数 • MID 使用字符串中从位置 P 开始的字符数
P	IN	Int	仅限 MID：要复制的第一个子串字符的位置 P= 1，表示 IN 字符串的起始字符位置
OUT	OUT	String, WString	输出字符串

表格 9-57　ENO 状态

ENO	条件	OUT
1	未检测到错误	有效字符
0	• L 或 P 小于或等于 0 • P 大于 IN 的最大长度 • IN 的当前长度超出 IN 的最大长度，或者 OUT 的当前长度超出 OUT 的最大长度 • IN 或 OUT 的最大长度不在分配的存储范围内 • IN 或 OUT 的最大长度为 0 或 255（String 数据类型），或者 0 或 65534（WString 数据类型）	当前长度被设置为 0
	要复制的子串长度 (L) 比 OUT 字符串的最大长度长。	复制字符，直到达到 OUT 的最大长度为止
	仅限 MID：L 或 P 小于或等于 0	当前长度被设置为 0
	仅限 MID：P 大于 IN 的最大长度	
	IN1 的当前长度超出 IN1 的最大长度，或者 IN2 的当前长度超出 IN2 的最大长度（无效字符串）	当前长度被设置为 0
	IN1、IN2 或 OUT 的最大长度不在分配的存储范围内	
	IN1、IN2 或 OUT 的最大长度为非法长度：0 或 255（String 数据类型），或者 0 或 65534（WString 数据类型）	

9.2.4.5 DELETE（删除字符串中的字符）

表格 9-58 删除子串指令

LAD/FBD	SCL	说明
	`out := DELETE(in, L, p);`	从字符串 IN 中删除 L 个字符。从字符位置 P（包括该位置）处开始删除字符，剩余字串在参数 OUT 中输出。 ● 如果 L 等于零，则在 OUT 中返回输入字符串。 ● 如果 L 与 P 的和大于输入字符串的长度，则一直删除到该字符串的末尾。

表格 9-59 参数的数据类型

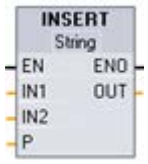
参数和类型		数据类型	说明
IN	IN	String, WString	输入字符串
L	IN	Int	要删除的字符数
P	IN	Int	要删除的第一个字符的位置：IN 字符串的第一个字符的位置编号为 1
OUT	OUT	String, WString	输出字符串

表格 9-60 ENO 状态

ENO	条件	OUT
1	未检测到错误	有效字符
0	P 大于 IN 的当前长度	将 IN 复制到 OUT 且不删除任何字符
	删除字符后的结果字符串比 OUT 字符串的最大长度长	复制结果字符串字符，直到达到 OUT 的最大长度为止
	L 小于 0，或者 P 小于或等于 0	当前长度被设置为 0
	IN 的当前长度超出 IN 的最大长度，或者 OUT 的当前长度超出 OUT 的最大长度	
	IN 或 OUT 的最大长度不在分配的存储范围内	
	IN 或 OUT 的最大长度为 0 或 255	

9.2.4.6 INSERT（在字符串中插入字符）

表格 9-61 插入子串指令

LAD/FBD	SCL	说明
	`out := INSERT(in1, in2, p);`	将字符串 IN2 插入字符串 IN1。在位置 P 的字符后开始插入。

表格 9-62 参数的数据类型

参数和类型		数据类型	说明
IN1	IN	String, WString	输入字符串 1
IN2	IN	String, WString	输入字符串 2
P	IN	Int	字符串 IN1 中字符串 IN2 插入点前的最后一个字符位置 字符串 IN1 的第一个字符的位置编号为 1。
OUT	OUT	String, WString	结果字符串

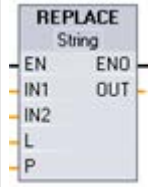
表格 9-63 ENO 状态

ENO	条件	OUT
1	未检测到错误	有效字符
0	P 大于 IN1 的长度	IN2 紧接最后一个 IN1 字符与 IN1 连接
	P 小于 0	当前长度被设置为 0
	插入后的结果字符串比 OUT 字符串的最大长度长	复制结果字符串字符，直到达到 OUT 的最大长度为止
	IN1 的当前长度超出 IN1 的最大长度，IN2 的当前长度超出 IN2 的最大长度，或 OUT 的当前长度超出 OUT 的最大长度（无效字符串）	当前长度被设置为 0
	IN1、IN2 或 OUT 的最大长度不在分配的存储范围内	
	IN1 或 IN2 的最大长度为 255，或者 OUT 的最大长度为 0 或 255（String 数据类型）	
	IN1 或 IN2 的最大长度为 65534，或者 OUT 的最大长度为 0 或 65534（WString 数据类型）	

9.2.4.7 REPLACE（替换字符串中的字符）

表格 9-64 替换子串指令

LAD/FBD	SCL	说明

	`out := REPLACE(`
	`in1:=_string_in_,`
	`in2:=_string_in_,`
	`L:=_int_in_,`
	`p:=_int_in);`
		替换字符串参数 IN1 中的 L 个字符。使用字符串参数 IN2 中的替换字符，从字符串 IN1 的字符位置 P（包括该位置）开始替换。

表格 9-65 参数的数据类型

参数和类型		数据类型	说明
IN1	IN	String, WString	输入字符串
IN2	IN	String, WString	替换字符的字符串
L	IN	Int	要替换的字符数
P	IN	Int	要替换的第一个字符的位置
OUT	OUT	String, WString	结果字符串

如果参数 L 等于零，则在字符串 IN1 的位置 P 处插入字符串 IN2，而不删除字符串 IN1 中的任何字符。

如果 P 等于 1，则用字符串 IN2 的字符替换字符串 IN1 的前 L 个字符。

表格 9-66　ENO 状态

ENO	条件	OUT
1	未检测到错误	有效字符
0	P 大于 IN1 的长度	IN2 紧接最后一个 IN1 字符与 IN1 连接
	P 在 IN1 范围内，但 IN1 中剩余的字符数小于 L	IN2 从位置 P 开始替换 IN1 的后端字符
	替换后的结果字符串比 OUT 字符串的最大长度长	复制结果字符串字符，直到达到 OUT 的最大长度为止
	IN1 的最大长度为 0	IN2 字符被复制到 OUT
	L 小于 0，或者 P 小于或等于 0	当前长度被设置为 0
	IN1 的当前长度超出 IN1 的最大长度，IN2 的当前长度超出 IN2 的最大长度，或 OUT 的当前长度超出 OUT 的最大长度	
	IN1、IN2 或 OUT 的最大长度不在分配的存储范围内	
	IN1 或 IN2 的最大长度为 255，或者 OUT 的最大长度为 0 或 255（String 数据类型）	
	IN1 或 IN2 的最大长度为 65534，或者 OUT 的最大长度为 0 或 65534（WString 数据类型）	

9.2.4.8　FIND（在字符串中查找字符）

表格 9-67　查找子串指令

LAD/FBD	SCL	说明

	`out := FIND(`
	`  in1:=_string_in_,`
	`  in2:=_string_in);`
		提供由 IN2 指定的子串在字符串 IN1 中的字符位置。从左侧开始搜索。在 OUT 中返回 IN2 字符串第一次出现的字符位置。如果在字符串 IN1 中没有找到字符串 IN2，则返回零。

表格 9-68　参数的数据类型

参数和类型		数据类型	说明
IN1	IN	String, WString	在该字符串内搜索
IN2	IN	String, WString	搜索该字符串
OUT	OUT	Int	字符串 IN1 中第一个搜索匹配项的字符位置

表格 9-69　ENO 状态

ENO	条件	OUT
1	未检测到错误	有效字符位置
0	IN2 大于 IN1	字符位置被设置为 0
	IN1 的当前长度超出 IN1 的最大长度，或者 IN2 的当前长度超出 IN2 的最大长度（无效字符串）	
	IN1 或 IN2 的最大长度不在分配的存储范围内	
	IN1 或 IN2 的最大长度为 255（String 数据类型）或 65535（WString 数据类型）	

9.2.5　运行系统信息

9.2.5.1　GetSymbolName（读取输入参数的变量）

表格 9-70　GetSymbolName 指令

LAD/FBD	SCL	说明

GetSymbolName
EN　ENO
variable　OUT
size
	`OUT := GetSymbolName(`
	`variable:=_parameter_in_,`
	`size:=_dint_in_);`
		GetSymbolName 指令返回对应来自块接口的变量名称的字符串。
		程序可以使用不同变量多次调用指令。与变量的过程值无关。
		指令返回和 OUT 参数处读取的名称相同的名称。

参数

下表列出了 GetSymbolName 指令的参数：

参数	声明	数据类型	存储区	说明
VARIABLE	Input	PARAMETER	Input、Output、InOut 参数区域	来自本地块接口、希望为其返回名称字符串值的变量
SIZE	Input	DINT	I、Q、M、D、L	OUT 参数处，输出字符数的限值： • SIZE > 0：GetSymbolName 返回名称的前 SIZE 个字符。 • SIZE = 0：GetSymbolName 返回整个名称。 • SIZE < 0：GetSymbolName 返回名称的最后 SIZE 个字符。
OUT	Return	WSTRING	I、Q、M、D、L	输入参数提供的变量名输出

在 VARIABLE 参数处指定块接口的输入参数。仅为该参数使用接口参数，并且没有 PLC 或数据块变量。

要限制读取变量名称的长度，可使用 SIZE 参数。如果该指令截断了名称，则通过名称末尾的字符“...”指示截断（Unicode 字符 16#2026）。请注意，该字符的长度为 1。

有关有效数据类型的更多信息，请参见 “数据类型 (页 119)”。

示例：SIZE 参数的含义

在以下示例中，说明了 SIZE 参数的含义。以下变量名称从块接口读取：“MyPLCTag”（左右双引号为名称的一部分。）

SIZE	GetSymbolName 返回	说明
1	'...'	• WSTRING 的第一个字符：' • 被截断名称的标识符：... • WSTRING 的最后一个字符：'
2	'"...'	• WSTRING 的第一个字符：' • 名称的第一个字符和被截断名称的标识符："... • WSTRING 的最后一个字符：'

SIZE	GetSymbolName 返回	说明
3	'"M...'	• WSTRING 的第一个字符：' • 名称的前两个字符和被截断名称的标识符："..."M... • WSTRING 的最后一个字符：'
6	'"MyPL...'	• WSTRING 的第一个字符：' • 名称的前五个字符和被截断名称的标识符："MyPL... • WSTRING 的最后一个字符：'
0	'"MyPLCTag"'	• WSTRING 的第一个字符：' • 该名称的所有字符："MyPLCTag" • WSTRING 的最后一个字符：'

示例：读取符号名称

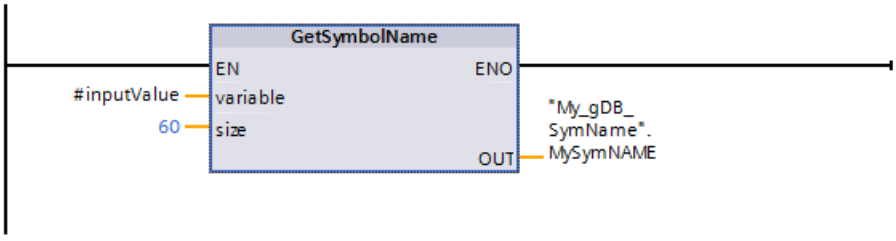
在以下示例中，通过块的输入参数读取互连的变量的名称。

在全局数据块中创建两个用于存储数据的变量。

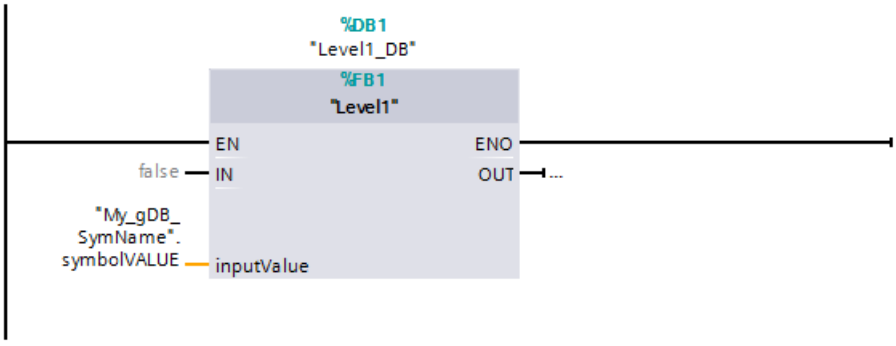
My_gDB_SymName

	名称	数据类型	启动值
1	Static		
2	MySymNAME	WString	WSTRING#''
3	symbolVALUE	Byte	16#42

在 Level1 块中，使用 BYTE 数据类型创建输入参数 inputValue。调用 Level1 块中的 GetSymbolName 指令。互连该指令的参数，如下所示：



如下所示互连 Level1 块的 inputValue 参数。



在 Level1 块中执行 GetSymbolName 指令。使用指令的输入参数 VARIABLE 检查 Level1 块的输入参数 inputValue 的互连情况。为执行此操作时，读取 symbolVALUE 变量并通过输出参数 OUT（“MySymNAME”）将其作为字符串输出。根据输入参数 SIZE 的值，字符串的长度被限制为 60 个字符。

My_gDB_SymName

	名称	数据类型	启动值	监视值
1	Static			
2	MySymNAME	WString	WSTRING#''	WSTRING#'"My_gDB_SymName".symbolVALUE'
3	symbolVALUE	Byte	16#42	16#42

9.2.5.2 GetSymbolPat（查询输入参数分配的复合全局名称）

表格 9-71 GetSymbolPath 指令

LAD/FBD	SCL	说明

GetSymbolPath
EN — ENO
variable — OUT
size
	`OUT := GetSymbolPath(`
	`variable:=_parameter_in_,`
	`size:=_dint_in_);`
		GetSymbolPath 指令用于读取块（FB 或 FC）本地接口处输入参数的复合全局名称。此名称包含存储路径与变量名。
		程序可以使用不同变量多次调用指令。与变量的过程值无关。
		指令返回和 OUT 参数处读取的名称相同的名称。

参数

下表列出了 GetSymbolPath 指令的参数：

参数	声明	数据类型	存储区	说明
VARIABLE	Input	PARAMETER	Input、Output、InOut 参数区域	选择用于读取输入参数来源全局名称的本地接口。
SIZE	Input	DINT	I、Q、M、D、L 或常数	OUT 参数处，输出字符数的限值。 • SIZE > 0：GetSymbolPath 返回名称的前 SIZE 个字符。 • SIZE = 0：GetSymbolPath 返回整个名称。 • SIZE < 0：GetSymbolPath 返回名称的最后 SIZE 个字符。
OUT	Output	WSTRING	I、Q、M、D、L	输出输入参数来源的变量名称。

有关有效数据类型的更多信息，请参见 “数据类型 (页 119)”。

用法

使用 GetSymbolPath 指令时请注意以下建议：

- 指定块接口，通过该块接口将在该指令的参数 VARIABLE 处读取输入变量名称：
  - 如果数据块变量提供输入参数，GetSymbolPath 输出数据块的名称、包含的结构与变量名称。
  - 如果 PLC 变量提供输入参数，GetSymbolPath 输出 PLC 变量的名称。
  - 如果某常量提供输入参数，GetSymbolPath 输出常量值。
- 要限制读取变量名称的长度，可使用 SIZE 参数。如果名称被截断，则在该名称的末尾处将标识为字符“...”（Unicode 字符 16#2026）。请注意，该字符的长度为 1。

示例：SIZE 参数的含义

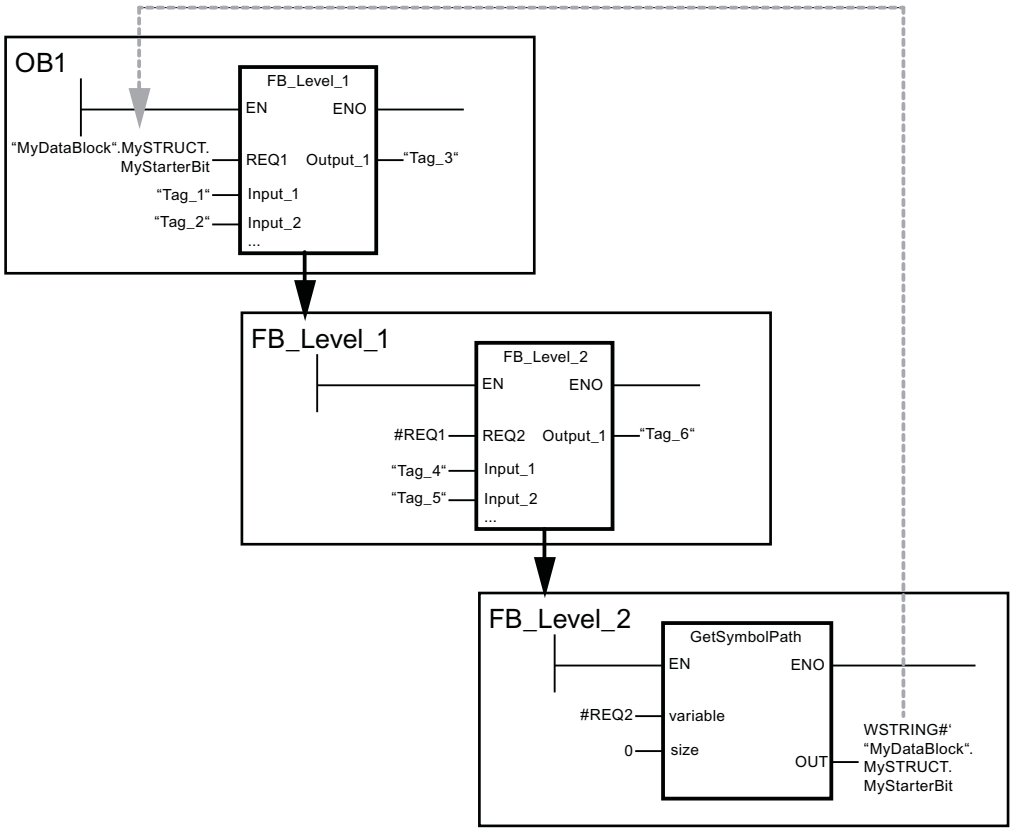
以下示例说明了 SIZE 参数的含义。GetSymbolPath 已从块接口读取了以下变量名称：“MyPLCTag”（左右双引号为名称的一部分。）

SIZE	GetSymbolPath 返回	说明
1	'...'	• WSTRING 的第一个字符：' • 被截断名称的标识符：... • WSTRING 的最后一个字符：'
2	'"...'	• WSTRING 的第一个字符：' • 名称的第一个字符和被截断名称的标识符："... • WSTRING 的最后一个字符：'
3	'"M...'	• WSTRING 的第一个字符：' • 名称的前两个字符和被截断名称的标识符："..."M... • WSTRING 的最后一个字符：'
6	'"MyPL...'	• WSTRING 的第一个字符：' • 名称的前五个字符和被截断名称的标识符："MyPL... • WSTRING 的最后一个字符：'
0	'"MyPLCTag"'	• WSTRING 的第一个字符：' • 该名称的所有字符："MyPLCTag" • WSTRING 的最后一个字符：'

示例：在多个块调用等级调用 GetSymbolPath

以下示例说明了如何在多个调用级别上使用 GetSymbolPath：

- 组织块 OB1 调用 FB_Level_1 块，继而调用 FB_Level_2 块。
- FB_Level_2 块执行 GetSymbolPath 以在 REQ2 接口处读取参数的路径。
- 由于 REQ1 接口提供 REQ2，因此指令可确定 REQ1 的输入参数路径。
- MyStarterBit 变量是 REQ1 输入参数。该位在 MyDatablock 数据块的 MySTRUCT 结构中。
  GetSymbolPath 读取该信息并在 OUT 参数处输出路径 ("MyDataBlock".MySTRUCT.MyStarterBit)。



9.2.5.3 GetInstanceName（读取块实例的名称）

表格 9-72 GetInstanceName 指令

LAD/FBD	SCL	说明
GetInstanceName - EN ENO - - size OUT -	`OUT := GetInstanceName(` `  size:=_dint_in_);`	可以使用 GetInstanceName 指令在函数块中读取实例数据块的名称。

参数

下表列出了 GetInstanceName 指令的参数：

参数	声明	数据类型	存储区	说明
SIZE	Input	DINT	I、Q、M、D、L 或常数	OUT 参数处，输出字符数的限值。 • SIZE > 0：GetInstanceName 返回名称的前 SIZE 个字符。 • SIZE = 0：GetInstanceName 返回整个名称。 • SIZE < 0：GetInstanceName 返回名称的最后 SIZE 个字符。
OUT	Output	WSTRING	I、Q、M、D、L	背景数据块的读取名称

有关有效数据类型的更多信息，请参见 “数据类型 (页 119)”。

示例：SIZE 参数的含义

要限制读实例名称的长度，可使用 SIZE 参数。如果指令已将名称截断，将由名称末尾处的字符“...”（Unicode 字符 16#2026）指示。请注意，该字符的长度为 1。

以下示例说明了 SIZE 参数的含义。GetInstanceName 已从块接口读取了以下实例名称：“Level1_DB”（左右双引号为名称的一部分。）

SIZE	GetSymbolPath 返回	说明
1	'...'	• WSTRING 的第一个字符：' • 被截断名称的标识符：... • WSTRING 的最后一个字符：'
2	'"...'	• WSTRING 的第一个字符：' • 名称的第一个字符和被截断名称的标识符："... • WSTRING 的最后一个字符：'
3	'"L...'	• WSTRING 的第一个字符：' • 名称的前两个字符和被截断名称的标识符："..."L... • WSTRING 的最后一个字符：'

SIZE	GetSymbolPath 返回	说明
6	'"Leve...'	• WSTRING 的第一个字符：' • 名称的前五个字符和被截断名称的标识符："Leve... • WSTRING 的最后一个字符：'
0	'"Level1_DB"'	• WSTRING 的第一个字符：' • 该名称的所有字符："Level1_DB" • WSTRING 的最后一个字符：'

GetInstanceName 将实例数据块的名称写入到 OUT 参数。如果实例数据块的名称比 WSTRING 的最大长度更长，指令将截断该名称。

示例：读取实例数据块的名称

以下示例显示如何读取实例数据块的名称。

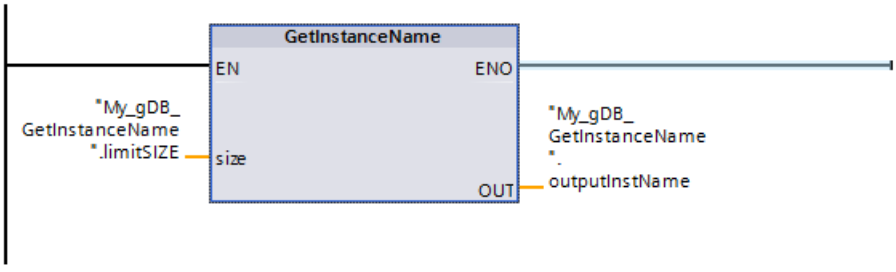
在全局数据块中创建两个用于存储数据的变量。

按如下所示定义指令参数。

My_gDB_GetInstanceName

		Name	Datentyp	Startwert
1		Static		
2		limitSIZE	DInt	0
3		outputInsName	WString	WSTRING#''

Level1_gin 块执行 GetInstanceName 指令，该指令确定 Level1_gin 块的关联的实例数据块，并在输出参数 OUT 处将名称作为字符串输出（outputInstName）。根据参数 SIZE (limitSIZE) 的值 0，字符串的长度不受限制。



My_gDB_GetInstanceName

		名称	数据类型	启动值	监视值
1		Static			
2		limitSIZE	DInt	0	0
3		outputInsName	WString	WSTRING#''	WSTRING#'"Level_1_DB"'

9.2.5.4 GetInstancePath（查询块实例的复合全局名称）

表格 9-73 GetInstancePath 指令

LAD/FBD	SCL	说明
GetInstancePath EN ENO size OUT	`OUT := GetInstancePath(` `  size:=_dint_in_);`	可以使用 GetInstancePath 指令在函数块中读取块实例的组合全局名称。当程序调用多个实例时，块实例的组合全局名称是完整调用层级的路径。

参数

下表列出了 GetInstancePath 指令的参数：

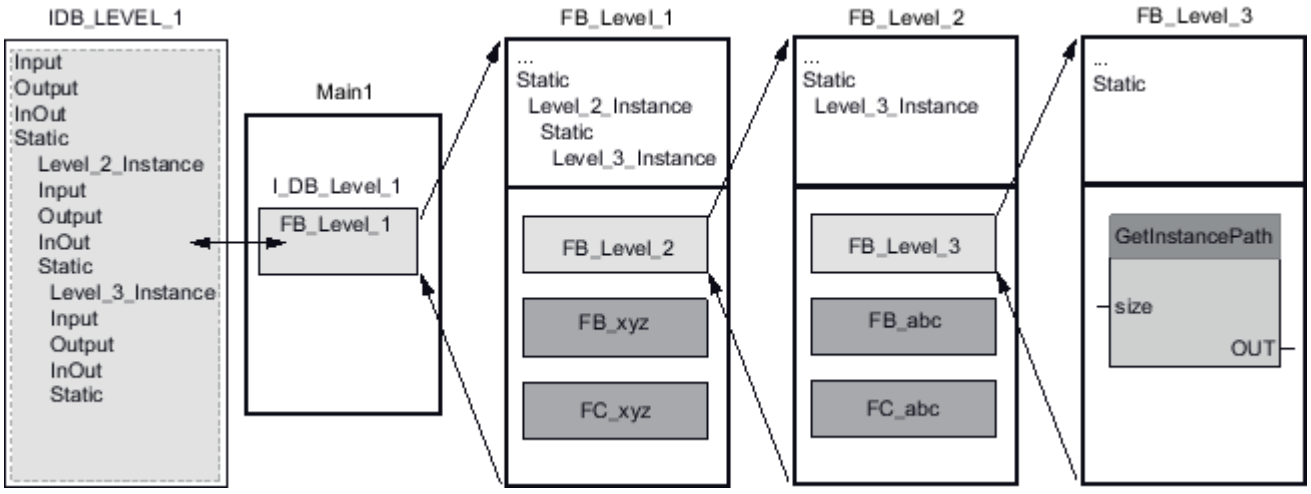
参数	声明	数据类型	存储区	说明
SIZE	Input	DINT	I、Q、M、D、L 或常数	OUT 参数处，输出字符数的限值。 • SIZE > 0：GetInstancePath 返回名称的前 SIZE 个字符。 • SIZE = 0：GeInstancePath 返回整个名称。 • SIZE < 0：GetInstancePath 返回名称的最后 SIZE 个字符。
OUT	Output	WSTRING	I、Q、M、D、L	读取块实例的全局名称。 如果块实例的全局名称比 WSTRING 的最大长度（254 个字符）长，GetInstancePath 将截断该名称。

有关有效数据类型的更多信息，请参见“数据类型 (页 119)”。

示例：调用 GetInstancePath 指令获取 FB 调用的多实例

在以下示例中，FB_Level_3 函数块调用 GetInstancePath 指令。

- FB_Level_3 函数块将其数据存储在 FB_Level_2 调用函数块中。
- FB_Level_2 函数块将其数据存储在 FB_Level_1 调用函数块中。
- FB_Level_1 函数块将其数据存储在 IDB_LEVEL_1 背景数据块中。通过使用多个实例，FB_Level_1 的背景数据块可包含三个函数块的所有数据。



在本示例中，根据 SIZE 参数的值，GetInstancePath 指令将返回以下值：

SIZE	GetInstancePath 返回	说明
1	'...'	• WSTRING 的第一个字符：' • 被截断名称的标识符：... • WSTRING 的最后一个字符：'
2	'"...'	• WSTRING 的第一个字符：' • 名称的第一个字符和被截断名称的标识符："... • WSTRING 的最后一个字符：'
3	'"I...'	• WSTRING 的第一个字符：' • 名称的前两个字符和被截断名称的标识符："..."I... • WSTRING 的最后一个字符：'
6	'"IDB_...'	• WSTRING 的第一个字符：' • 名称的前五个字符和被截断名称的标识符："IDB_... • WSTRING 的最后一个字符：'
0	'"IDB_LEVEL_1".Level_2_Instance.Level_3_Instance'	• WSTRING 的第一个字符：' • 该名称的所有字符："IDB_LEVEL_1".Level_2_Instance.Level_3_Instance • WSTRING 的最后一个字符：'

说明

将函数块中的 GetInstancePath 与单个实例一起使用

如果在其中调用 GetInstancePath 的函数块在其自身的实例数据块中保存数据，GetInstancePath 将单个实例的名称作为全局名称输出。在这种情况下，参数 OUT 中的结果对应于 GetInstanceName (页 378) 指令。

9.2.5.5 GetBlockName（读取块名称）

表格 9-74 GetBlockName 指令

LAD/FBD	SCL	说明
GetBlockName EN ENO SIZE RET_VAL	`RET_VAL := GetBlockName(` `size:=_dint_in_);`	可以使用 GetBlockName 指令读取在其中调用指令的块的名称。

参数

下表列出了 GetBlockName 指令的参数：

参数	声明	数据类型	存储区	说明
SIZE	Input	UINT	I、Q、M、D、L 或常数	RET_VAL 参数处，输出字符数的限值。 • SIZE > 0：GetBlockName 返回名称的前 SIZE 个字符。 • SIZE = 0：GetBlockName 返回整个名称。 • SIZE < 0：GetBlockName 返回名称的最后 SIZE 个字符。
RET_VAL	Output	WSTRING	I、Q、M、D、L	背景数据块的读取名称

有关有效数据类型的更多信息，请参见“数据类型 (页 119)”。

示例：SIZE 参数的含义

要将块名称的长度限制为特定字符数量，则可在参数 SIZE 处指定最大长度。如果 GetBlockName 将名称截断，将由名称末尾处的字符“...”（Unicode 字符 16#2026）指示截断。请注意，该字符的长度为 1。

以下示例说明了 SIZE 参数的含义。GetBlockName 已读取以下块名称：Level1_gbn"（左右双引号为名称的一部分。）

SIZE	GetBlockName 返回	说明
1	'...'	• WSTRING 的第一个字符：' • 被截断名称的标识符：... • WSTRING 的最后一个字符：'
2	'"...'	• WSTRING 的第一个字符：' • 名称的第一个字符和被截断名称的标识符："... • WSTRING 的最后一个字符：'
3	'"L...'	• WSTRING 的第一个字符：' • 名称的前两个字符和被截断名称的标识符："..."L... • WSTRING 的最后一个字符：'
6	'"Leve...'	• WSTRING 的第一个字符：' • 名称的前五个字符和被截断名称的标识符："Leve... • WSTRING 的最后一个字符：'
0	'"Level1_gbn"'	• WSTRING 的第一个字符：' • 该名称的所有字符："Level1_gbn" • WSTRING 的最后一个字符：'

GetBlockName 在参数 RET_VAL 处写入块名称。如果块名称比 WSTRING 的最大长度长，将截断该名称。

示例：读取块名称

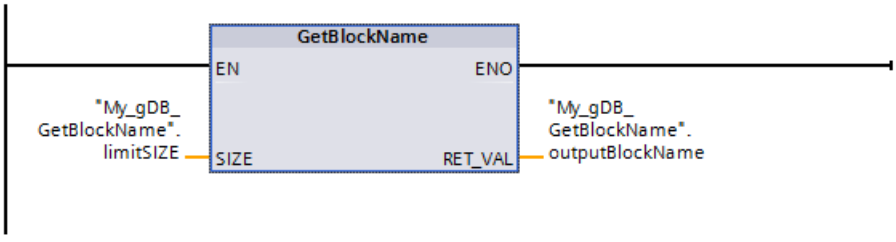
以下示例显示如何读取块名称。

1. 在全局数据块中创建两个用于存储数据的变量。

My_gDB_GetBlockName

	名称	数据类型	启动值
1	Static		
2	limitSIZE	DInt	0
3	outputBlockName	WString	WSTRING#''

2. 按如下所示定义指令参数：



Level1_gbn 块执行 GetBlockName 指令。GetBlockName 读取 Level1_gbn 块的名称，在输出参数 RET_VAL (outputBlockName) 处将该名称作为字符串输出。由于 SIZE 参数为 0 (limitSIZE)，字符串的长度不受限制。

My_gDB_GetBlockName

		名称	数据类型	启动值	监视值
1		Static			
2		limitSIZE	DInt	0	0
3		outputBlockName	WString	WSTRING#''	WSTRING#'"Level_1_gbn"

9.3 分布式 I/O（PROFINET、PROFIBUS 或 AS-Interface）

9.3.1 分布式 I/O 指令

可对 PROFINET、PROFIBUS 或 AS-i 使用以下分布式 I/O 指令：

- RDREC 指令 (页 387)：从模块或设备读取带有编号 INDEX 的数据记录。
- WRREC 指令 (页 387)：将带有编号 INDEX 的数据记录传输到由 ID 定义的模块或设备。
- GETIO 指令 (页 390)：一致性地读取 DP 标准从站/PROFINET IO 设备的所有输入。
- SETIO 指令 (页 391)：一致性地从 OUTPUTS 参数定义的源范围传输数据到被定址的 DP 标准从站/PROFINET IO 设备。
- GETIO_PART 指令 (页 392)：一致性地读取 IO 模块输入的相关部分。
- SETIO_PART 指令 (页 394)：一致性地从 OUTPUTS 参数覆盖的源区域写入数据到 IO 模块的输出中。
- RALRM 指令 (页 395)：允许您从模块或设备接收中断及所有相应信息，并将该信息提供给其输出参数。
- DPRD_DAT 指令 (页 409)：允许您通过 DPRD_DAT 指令从模块或设备读取大于 64 个字节的一致性数据区域。
- DPWR_DAT 指令 (页 409):允许您通过 DPWR_DAT 指令向模块或设备写入大于 64 个字节的一致性数据区域。

D_ACT_DP 指令 (页 399)允许您有针对性地禁用和启用组态的 PROFINET IO 设备。还可确定每个指定的 PROFINET IO 设备当前处于激活还是取消激活状态。

说明

注：只能将 D_ACT_DP 指令用于 PROFINET IO 设备。不能使用 PROFIBUS DP 从站的指令。

DPNRM_DG 指令 (页 418)允许您读取 DP 从站的当前诊断数据，格式根据 EN 50 170 Volume 2，PROFIBUS 中的规定。

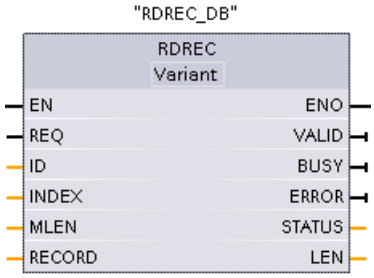
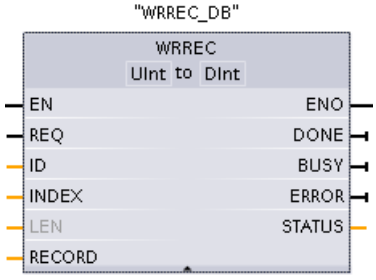
说明

只能将 DPNRM_DG 指令用于 PROFIBUS。

9.3.2 RDREC 和 WRREC（读/写数据记录）

可以对 PROFINET、PROFIBUS 和 AS-i 使用 RDREC（读取数据记录）和 WRREC（写入数据记录）指令。

表格 9-75 RDREC 和 WRREC 指令

LAD/FBD	SCL	说明
	"RDREC_DB"(req:=_bool_in_, ID:=_word_in_, index:=_dint_in_, mlen:=_uint_in_, valid=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_dword_out_, len=>_uint_out_, record:=_variant_inout_);	使用 RDREC 指令从通过 ID 寻址的组件（如中央机架或分布式组件（PROFIBUS DP 或 PROFINET IO））读取编号为 INDEX 的数据记录。在 MLEN 中分配要读取的最大字节数。目标区域 RECORD 的选定长度至少应该为 MLEN 个字节。
	"WRREC_DB"(req:=_bool_in_, ID:=_word_in_, index:=_dint_in_, len:=_uint_in_, done=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_dword_out_, record:=_variant_inout_);	使用 WRREC 指令将记录号为 INDEX 的数据 RECORD 传送到通过 ID 寻址的 DP 从站/PROFINET IO 设备组件，如中央机架上的模块或分布式组件（PROFIBUS DP 或 PROFINET IO）。 分配要传送的数据记录的字节长度。因此，源区域 RECORD 的选定长度至少应该为 LEN 个字节。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“RDREC_DB”和“WRREC_DB”是背景 DB 的名称。

表格 9-76　RDREC 和 WRREC 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	REQ = 1：传送数据记录
ID	IN	HW_IO (Word)	DP 从站/PROFINET IO 组件（模块或子模块）的逻辑地址： ● 对于输出模块，必须将位 15 置位（例如，对于地址 5：ID:=DW#16#8005）。 ● 对于组合模块，应指定两个地址中的较小者。 注：在 V3.0 中，可以通过以下两种方法之一来确定设备 ID： ● 通过选择下列“网络视图”(Network view) 选项： – “设备”（灰色框） – 设备的“属性” – “硬件标识符” 注：然而，并非所有设备都会显示硬件标识符。 ● 通过选择下列“项目树”(Project tree) 菜单选项： – PLC 变量 – 默认变量表 – “系统常量”选项卡 将显示所有已组态的设备硬件标识符。 注：在 V4.0 中，转到变量表并在“系统常量”(System Constants) 下找到“设备名称 [标头]”参数来确定接口模块的设备 ID（硬件标识符）。
INDEX	IN	Byte, Word, USInt, UInt, SInt, Int, DInt	数据记录号
MLEN	IN	Byte, USInt, UInt	要获取的数据记录信息的最大长度（字节）(RDREC)
VALID	OUT	Bool	新数据记录已接收并且有效 (RDREC)。上一请求已完成且没有出错后，VALID 位将保持为 TRUE 一个扫描周期时间。
DONE	OUT	Bool	已传送数据记录 (WRREC)。上一请求已完成且没有出错后，DONE 位将保持为 TRUE 一个扫描周期时间。
BUSY	OUT	Bool	● BUSY = 1：读取 (RDREC) 或写入 (WRREC) 过程尚未终止。 ● BUSY = 0：数据记录传送已完成。

参数和类型		数据类型	说明
ERROR	OUT	Bool	ERROR = 1：读取 (RDREC) 或写入 (WRREC) 出现错误。上一请求因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。STATUS 参数中的错误代码值仅在 ERROR = TRUE 的一个扫描周期内有效。
STATUS	OUT	DWord	块状态 (页 404)或错误信息 (页 557)
LEN	OUT (RDREC) IN (WRREC)	UInt	• 已获取的数据记录信息的长度 (RDREC) • 要传送的数据记录的最大长度（字节）(WRREC)
RECORD	IN_OUT	Variant	• 已获取的数据记录的目标区域 (RDREC) • 数据记录 (WRREC)

RDREC 和 WRREC 指令以异步方式运行，即，处理过程跨越多个指令调用。以 REQ = 1 调用 RDREC 或 WRREC 来启动作业。

通过输出参数 BUSY 和输出参数 STATUS 的两个中间字节显示作业状态。输出参数 BUSY 设置为 FALSE 时，说明数据记录的传送完成。

输出参数 VALID (RDREC) 或 DONE (WRREC) 的值为 TRUE 时（只持续一个扫描周期），表示数据记录已成功传送到目标区域 RECORD (RDREC) 或目标设备 (WRREC)。使用 RDREC 时，输出参数 LEN 包含所获取数据的长度（字节）。

输出参数 ERROR （只在 ERROR = TRUE 时持续一个扫描周期）表示发生数据记录传送错误。在这种情况下，输出参数 STATUS （只在 ERROR = TRUE 时持续一个扫描周期）包含错误信息。

由硬件设备制造商定义数据记录。有关数据记录的详细信息，请参见硬件设备制造商的设备文档。

最多可以同时使用四条 RDREC 指令和四条 WRREC 指令。

说明

如果已通过 GSD 文件（GSD 修订版 3 及更高版本）组态 DPV1 从站且 DP 主站的 DP 接口已设置为“S7 兼容”(S7 compatible)，则可能无法在用户程序中通过“RDREC”从 I/O 模块读取数据记录，也不能通过“WRREC”将记录写入 I/O 模块。这种情况下，DP 主站寻址错误的插槽（组态的插槽 + 3）。

解决方法：将 DP 主站的接口设置为“DPV1”。

说明

“RDREC”和“WRREC”指令的接口与“符合 IEC 61131-3 的 PROFIBUS 准则、PROFIBUS 通信和代理函数块”中定义的“RDREC”和“WRREC”FB 完全相同。

说明

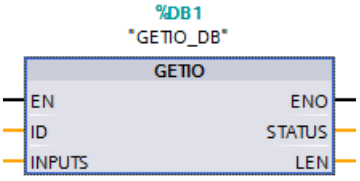
如果您使用“RDREC”或“WRREC”读取或写入 PROFINET IO 的数据记录，那么 CPU 会将参数 INDEX、MLEN 和 LEN 中的负值解释为无符号 16 位整型值。

9.3.3 GETIO（读取过程映像）

可以使用指令“GETIO”一致性地读取 DP 从站和 PROFINET IO 设备模块和子模块的输入。指令“GETIO”调用指令 “DPRD_DAT (页 409)”。如果在数据传输过程中未发送任何错误，则读取的数据将被输入到 INPUTS 指示的目标区域中。

表格 9-77 GETIO （读取过程映像）指令

LAD/FBD	SCL	说明

	`"GETIO_DB"(`
	`id:=_uint_in_,`
	`status=>_dword_out_,`
	`len=>_int_out_,`
	`inputs:=_variant_inout_);`
		使用指令“GETIO”一致性地读取 DP 标准从站/PROFINET IO 设备的所有输入。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“GETIO_DB”是背景 DB 的名称。

目标区域的长度必须大于或等于选定组件的长度。

如果从具有模块化组态或几个 DP 标识符的 DP 标准从站中读取，则每次“GETIO”调用，只访问组态起始地址处一个组件/DP 标识符的数据。

参数

下表列出了“GETIO”指令的参数：

参数	声明	数据类型	说明
ID	IN	HW_SUBMODULE	DP 标准从站/PROFINET IO 设备的硬件 ID。
STATUS[1]	OUT	DWord	存储 “DPRD_DAT (页 409)”的错误信息，格式为 DW#16#40xxxx00

参数	声明	数据类型	说明
LEN	OUT	Int	读取的数据量，单位[字节]
INPUTS	IN_OUT	Variant	读取数据所在的目标区域：目标区域的长度必须大于或等于选定 DP 标准从站/PROFINET IO 设备的长度。 可以使用以下数据类型： ● 系统数据类型和系统数据类型数组：BYTE、CHAR、SINT、USINT、WORD、INT、UINT、DWORD、DINT、UDINT、REAL、LREAL、LWORD、LINT ● 用户定义类型 (UDT) ● 结构 (STRUCT)，但仅在未经优化的数据块中 (DB)

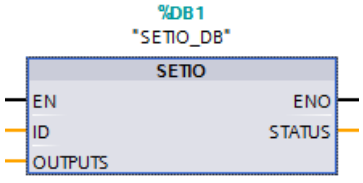
1 当显示“GETIO” 错误代码时，使用 DWord 数据类型。

9.3.4 SETIO（传送过程映像）

“SETIO”指令用于一致性地从 OUTPUTS 参数定义的源范围读取传输数据到 DP 从站和 PROFINET IO 设备的已寻址模块或子模块中。如果已将 DP 标准从站/PROFINET IO 设备的相关地址区域组态为过程映像中的一致性范围，则数据将被传输到过程映像。传输期间“SETIO”调用 “DPWR_DAT (页 409)”指令。

表格 9-78 SETIO （读取过程映像）指令

LAD/FBD	SCL	说明

	`"SETIO_DB"(`
	`id:=_uint_in_,`
	`status=>_dword_out_,`
	`outputs:=_variant_inout_);`
		“SETIO”指令用于一致性地从参数 OUTPUTS 定义的源范围传输数据到寻址的 DP 标准从站/PROFINET IO 设备中。

1 STEP 7 会在插入指令时自动创建 DB。

2 在 SCL 示例中，“SETIO_DB”是背景 DB 的名称。

源范围的长度必须大于或等于选定组件的长度。

对于模块化组态或具有几个 DP 标识符的 DP 标准从站/PROFINET IO 设备，每次“SETIO”调用，只能访问一个 DP 标识符/组件。

参数

下表列出了“SETIO”指令的参数：

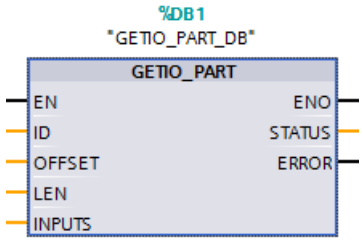
参数	声明	数据类型	说明
ID	IN	HW_SUBMODULE	DP 标准从站/PROFINET IO 设备的硬件 ID。
STATUS[1]	OUT	DWord	存储 “DPWR_DAT (页 409)”的错误信息，格式为 DW#16#40xxxx00
OUTPUTS	IN_OUT	Variant	要写入数据的源范围：源范围的长度必须大于或等于选定 DP 标准从站/PROFINET IO 设备的长度。 可以使用以下数据类型： ● 系统数据类型和系统数据类型数组：BYTE、CHAR、SINT、USINT、WORD、INT、UINT、DWORD、DINT、UDINT、REAL、LREAL、LWORD、LINT ● 用户定义类型 (UDT) ● 结构 (STRUCT)，但仅在未经优化的数据块中 (DB)

[1] 当显示“SETIO”错误代码时，使用 DWord 数据类型。

9.3.5 GETIO_PART（读取过程映像区域）

可以使用指令“GETIO_PART”一致性地读取 DP 从站和 PROFINET IO 设备模块和子模块输入的相关部分。GETIO_PART 调用指令 “DPRD_DAT (页 409)”。

表格 9-79 GETIO_PART （读取过程映像区域）指令

LAD/FBD	SCL	说明
	"GETIO_PART_DB"(id:=_uint_in_, offset:=_int_in_, len:=_int_in_, status=>_dword_out_, error=>_bool_out_, inputs:=_variant_inout_);	指令 GETIO_PART 用于一致性地读取 IO 模块输入的相关部分。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“GETIO_PART_DB”是背景 DB 的名称。

可使用 ID 输入参数，通过硬件 ID 选择 IO 模块。

可使用 OFFSET 和 LEN 参数指定过程映像区域中要读取的部分。如果 OFFSET 和 LEN 指定的输入区域没有被模块完全覆盖，则相应块将返回错误代码 DW#16#4080B700。

目标区域的长度必须大于或等于待读取的字节数：

- 如果数据传送过程中没有出现错误，则 ERROR 接收 FALSE 值。读取的数据将写入由参数 INPUTS 定义的目标区域内。
- 如果数据传送过程中没有出现错误，则 ERROR 接收 TRUE 值。STATUS 参数将从 DPRD_DAT 中接收错误信息。
- 如果目标区域大于 LEN，则指令将写入目标区域的前 LEN 个字节。ERROR 接收 FALSE 值。

参数

下表列出了 GETIO_PART 指令的参数：

参数	声明	数据类型	说明
ID	IN	HW_SUBMODULE	模块的硬件标识符
OFFSET	IN	Int	组件过程映像中要读取的第一个字节的编号（最小值：0）
LEN	IN	Int	要读取的字节数
STATUS[1]	OUT	DWord	如果 ERROR = TRUE，存储“DPRD_DAT (页 409)”的错误信息，格式为 DW#16#40xxxx00
ERROR	OUT	Bool	错误显示：ERROR = TRUE，如果调用 DPRD_DAT (页 409) 时出错
INPUTS	IN_OUT	Variant	读取数据的目标区域：如果目标区域大于 LEN，则指令将写入目标区域的前 LEN 个字节。 可以使用以下数据类型： • 系统数据类型和系统数据类型数组：BYTE、CHAR、SINT、USINT、WORD、INT、UINT、DWORD、DINT、UDINT、REAL、LREAL、LWORD、LINT • 用户定义类型 (UDT) • 结构 (STRUCT)，但仅在未经优化的数据块中 (DB)

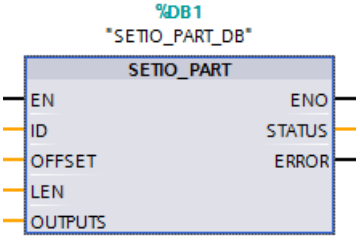
[1] 当显示 GETIO_PART 错误代码时，使用 DWord 数据类型。

9.3.6 SETIO_PART（传送过程映像区域）

使用指令“SETIO_PART”，可将数据从 OUTPUTS 覆盖的源区域一致性地写入 DP 从站和 PROFINET IO 设备的模块或子模块的输出中。SETIO_PART 调用指令 “DPWR_DAT (页 409)”。

表格 9-80 SETIO_PART （传送过程映像区域）指令

LAD/FBD	SCL	说明

	`"SETIO_PART_DB"(`
	`id:=_uint_in_,`
	`offset:=_int_in_,`
	`len:=_int_in_,`
	`status=>_dword_out_,`
	`error=>_bool_out_,`
	`outputs:=_variant_inout_);`
		可使用 SETIO_PART 指令，一致性地将数据从 OUTPUTS 覆盖的源区域写入到 IO 模块的输出中。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“SETIO_PART_DB”是背景 DB 的名称。

通过输入参数 ID,可根据所识别的硬件选择 I/O 模块。

通过参数 OFFSET 和 LEN,可为由 ID 寻址的组件指定待写入的过程映像区域部分。如果 OFFSET 和 LEN 指定的输出区域没有被模块完全覆盖，则相应块将返回错误代码 DW#16#4080B700。

目标区域的长度必须大于或等于待读取的字节数：

- 如果数据传送过程中没有出现错误，则 ERROR 接收 FALSE 值。
- 如果数据传送过程中出现错误，则 ERROR 接收 TRUE 值，STATUS 接收 DPWR_DAT 的错误信息。
- 如果源区域大于 LEN，指令将传输 OUTPUTS 的前 LEN 个字节。ERROR 接收 FALSE 值。

参数

下表列出了 SETIO_PART 指令的参数：

参数	声明	数据类型	说明
ID	IN	HW_SUBMODULE	IO 模块的硬件标识符
OFFSET	IN	Int	组件过程映像中要写入的第一个字节的编号（最小值：0）

参数	声明	数据类型	说明
LEN	IN	Int	要写入的字节数
STATUS[1]	OUT	DWord	如果 ERROR = TRUE，存储 “DPWR_DAT (页 409)”的错误信息，格式为 DW#16#40xxxx00
ERROR	OUT	Bool	错误显示：ERROR = TRUE，如果调用 “DPWR_DAT (页 409)”时出错。
OUTPUTS	IN_OUT	Variant	要写入数据的源范围：如果源区域大于 LEN，则传送 OUTPUTS 的前 LEN 个字节。 可以使用以下数据类型： • 系统数据类型和系统数据类型数组：BYTE、CHAR、SINT、USINT、WORD、INT、UINT、DWORD、DINT、UDINT、REAL、LREAL、LWORD、LINT • 用户定义类型 (UDT) • 结构 (STRUCT)，但仅在未经优化的数据块中 (DB)

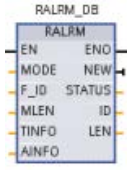
[1] 当显示 SETIO_PART 错误代码时，使用 DWord 数据类型。

9.3.7 RALRM（接收中断）

可以对 PROFINET 和 PROFIBUS 使用 RALRM（读取报警）指令。

表格 9-81 RALRM 指令

LAD/FBD	SCL	说明

	`"RALRM_DB"(`
	`mode:=_int_in_,`
	`f_ID:=_word_in_,`
	`mlen:=_uint_in_,`
	`new=>_bool_out_,`
	`status=>_dword_out_,`
	`ID=>_word_out_,`
	`len=>_uint_out_,`
	`tinfo:=_variant_inout_,`
	`ainfo:=_variant_inout_);`
		使用 RALRM（读取报警）指令从 PROFIBUS 或 PROFINET I/O 模块/设备读取诊断中断信息。
		输出参数中的信息包含被调用 OB 的启动信息以及中断源的信息。
		在中断 OB 中调用 RALRM，可返回导致中断的事件的相关信息。在 S7-1200 中，支持以下诊断 OB 中断：状态、更新、配置文件、诊断错误中断、拔出或插入模块、机架或站故障。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“RALRM_DB”是背景 DB 的名称。

表格 9-82　参数的数据类型

参数和类型		数据类型	说明
MODE	IN	Byte, USInt, SInt, Int	工作模式
F_ID	IN	HW_IO (Word)	组件（模块）的逻辑起始地址，将从该地址接收中断 注：可以通过以下两种方法之一来确定设备 ID： ● 通过选择下列“网络视图”(Network view) 选项： – “设备”（灰色框） – 设备的“属性” – “硬件标识符” 注：并非所有设备都会显示硬件标识符。 ● 通过选择下列“项目树”(Project tree) 菜单选项： – PLC 变量 – 默认变量表 – “系统常量”选项卡 – 将显示所有已组态的设备硬件标识符。
MLEN	IN	Byte, USInt, UInt	要接收的数据中断信息的最大长度（字节）。如果 MLEN 为 0，则允许接收的数据中断信息量与 AINFO 目标区域中提供的数据中断信息量相同。
NEW	OUT	Bool	已接收新中断。
STATUS	OUT	DWord	RALRM 指令的状态。有关详细信息，请参见“RDREC、WRREC 和 RALRM 的 STATUS 参数” (页 404)。
ID	OUT	HW_IO (Word)	导致诊断中断的 I/O 模块的硬件标识符 注：有关如何确定设备 ID 的说明，请参见参数 F_ID 。
LEN	OUT	DWord, UInt, UDInt, DInt, Real, LReal	已接收的 AINFO 中断信息的长度
TINFO	IN_OUT	Variant	任务信息：OB 启动和管理信息的目标范围。TINFO 的长度始终为 32 个字节。
AINFO	IN_OUT	Variant	中断信息：头信息和附加中断信息的目标区域。对于 AINFO，如果 MLEN 大于 0，将提供至少 MLEN 个字节的长度。AINFO 长度为变量。

说明

如果在启动事件不是 I/O 中断的 OB 中调用“RALRM”，该指令在输出中提供的信息会相应减少。

在不同的 OB 中调用“RALRM”时，务必要使用不同的背景数据块。如果评估来自相关中断 OB 外“RALRM”调用的数据，则需为每个 OB 启动事件指定一个单独的实例 DB。

说明

“RALRM”指令的接口与“符合 IEC 61131-3 的 PROFIBUS 准则、PROFIBUS 通信和代理函数块”中定义的“RALRM”FB 完全相同。

调用 RALRM

可以在三种不同的操作模式 (MODE) 下调用 RALRM 指令。

表格 9-83 RALRM 指令操作模式

MODE	说明
0	• ID 包含触发中断的 I/O 模块的硬件标识符。 • 输出参数 NEW 被设置为 TRUE。 • LEN 产生 0 输出。 • AINFO 和 TINFO 不使用任何信息进行更新。
1	• ID 包含触发中断的 I/O 模块的硬件标识符。 • 输出参数 NEW 被设置为 TRUE。 • LEN 产生输出，数量为返回的 AINFO 数据的字节数。 • AINFO 和 TINFO 使用中断相关信息进行更新。
2	如果分配给输入参数 F_ID 的硬件标识符触发了中断，则： • ID 包含触发中断的 I/O 模块的硬件标识符。应与 F_ID 处的值相同。 • 输出参数 NEW 被设置为 TRUE。 • LEN 产生输出，数量为返回的 AINFO 数据的字节数。 • AINFO 和 TINFO 使用中断相关信息进行更新。

说明

如果为过短的 TINFO 或 AINFO 指定目标区域，则 RALRM 无法返回完整信息。

MLEN 可限制所返回 AINFO 数据的量。

有关如何解释 TINFO 和 AINFO 数据的信息，请参见 STEP 7 在线信息系统的 AINFO 参数和 TINFO 参数。

TInfo 组织块数据

下表显示了如何为 RALRM 指令安排 TInfo 数据：

与 OB 相同：状态、更新、配置文件、诊断错误中断、拔出或插入模块以及机架或站故障	0	SI_Format	OB_Class	OB_Nr	
	4	LADDR			
TI_Submodule - OB：状态、更新和配置文件	4			插槽	
	8	区分符		0	
TI_DiagnosticInterrupt - OB：诊断错误中断	4			IO_State	
	8	通道		MultiError	0
TI_PlugPullModule - OB：拔出或插入模块	4			Event_Class	Fault_ID
	8	0		0	
TI_StationFailure - OB：机架或站故障	4			Event_Class	Fault_ID
	8	0		0	
与 OB 相同：状态、更新、配置文件、诊断错误中断、拔出或插入模块以及机架或站故障	12	0			
	16				
	20	地址		slv_prfl	intr_type
	24	flags1	flags2	id	
	28 [1]	制造商		背景	

[1] 字节 28 - 31（制造商和背景）不能与 PROFIBUS 配合使用。

说明

有关 TINFO 数据的详细信息，请参见 STEP 7 的在线信息系统。

9.3.8 D_ACT_DP（启用/禁用 PROFINET IO 设备）

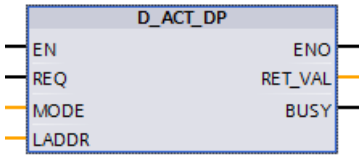
通过指令“D_ACT_DP”，可根据需要禁用和启用组态的 PROFINET IO 设备。此外，还可确定每个指定的 PROFINET IO 设备当前处于激活还是取消激活状态。

说明

只能将 D_ACT_DP 指令用于 PROFINET IO 设备。不能使用 PROFIBUS DP 从站的指令。

表格 9-84 D_ACT_DP 指令

LAD/FBD	SCL	说明

	`"D_ACT_DP_DB"(`
	`req:=_bool_in_,`
	`mode:=_usint_in_,`
	`laddr:=_uint_in_,`
	`ret_val=>_int_out_,`
	`busy=>_bool_out_);`
		使用 D_ACT_DP 指令禁用和启用组态的 PROFINET IO 设备并确定每个指定的 PROFINET IO 设备当前处于激活还是取消激活状态。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“D_ACT_DP_SFB_DB”是背景 DB 的名称。

无法通过指令 D_ACT_DP 禁用/启用 IE/PB Link PN IO 类型的网关。但如果在指定网关中使用 D_ACT_DP，则 CPU 将返回值 W#16#8093（对于 LADDR 中指定的地址，没有可激活或取消激活的硬件对象）。

说明

取消激活或禁用操作需要通过循环控制点的多个运行。因此，需要等待编程回路中这种作业结束。

功能描述

D_ACT_DP 是异步指令，这意味着作业处理需要多次执行 D_ACT_DP 指令来完成。使 REQ = 1，调用“D_ACT_DP”，将启动作业。

输出参数 RET_VAL 和 BUSY 用于指示作业状态。

应用

如果在实际不存在或当前不需要的 CPU 中组态 PROFINET IO 设备，CPU 将不会按规定时间间隔继续访问这些 PROFINET IO 设备。取消激活设备后，将停止进一步的 CPU 访问。从而避免错误事件的发生。

示例

从机器 OEM 的角度看，这可提供大量的系列生产选项。但每一交付的机器都只包括一种所选选项组合。

制造商将每个可能的机器选项组态为 PROFINET IO 设备。制造商执行该操作从而可创建和维护拥有所有可能选项的通用用户程序。在机器启动时，可使用 D_ACT_DP 取消激活所有不存在的 PROFINET IO 设备。

与此类似的情况是机床，可以使用大量加工选件，但实际上常常用到的仅是其中的一小部分。这些工具即作为 PROFINET IO 设备执行。使用 D_ACT_DP，用户程序可激活当前需要的工具，取消激活那些稍后才用到的工具。

作业标识

如果已启动一个取消激活或激活作业，并在作业完成之前，再次调用了 D_ACT_DP，则指令的行为取决于新的调用是否会涉及同一作业。如果输入参数 LADDR 匹配，则该调用将作为跟随调用解释。

取消激活 PROFINET IO 设备

如果使用 D_ACT_DP 取消激活 PROFINET IO 设备时，则其过程输出会置为组态的替代值或“0”（安全状态）。分配的 PROFINET IO 控制器不再继续寻址该组件。PROFINET IO 控制器或 CPU 上的错误 LED 不会将取消激活的 PROFINET IO 设备识别为故障或丢失。

CPU 将 PROFINET IO 设备的过程映像输入更新为“0”。因此，CPU 将取消激活的 PROFINET IO 设备作为故障 PROFINET IO 设备对待。

如果从程序直接访问之前取消激活的 PROFINET IO 设备，则系统行为取决于块错误处理选择：

- 如果启用了全局错误处理，系统将在诊断缓冲区中输入访问错误启动事件，并保持 RUN 模式。
- 如果启用了错误处理，系统会在错误结构中输入错误原因。您可以使用 GET_ERROR_ID (页 314) 指令访问错误原因。
  读访问错误返回“0”。有关错误处理的更多信息，请参见“事件执行的优先级与排队” (页 99)。

如果尝试访问通过指令（如“RD_REC (页 387)”）取消激活的 PROFINET IO 设备，则在 RET_VAL 中会收到和不可用 PROFINET IO 设备相同的错误信息。

如果在使用 D_ACT_DP 进行取消激活操作后，PROFINET IO 站出现故障，操作系统不会检测该故障。

激活 PROFINET IO 设备

使用 D_ACT_DP 重新激活 PROFINET IO 设备后，即由相关的 PROFINET IO 控制器组态部件并分配参数（如同返回故障 PROFINET IO 站一样）。当组件能够传送用户数据时，启用即完成。

如果尝试通过 D_ACT_DP 指令激活无法访问的 PROFINET IO 设备（如，因物理断开总线而导致无法访问），则在分布式 I/O 所组态的参数分配时间结束后，该指令将返回错误代码 W#16#80A7。PROFINET IO 设备将激活，但实际上所激活的 PROFINET IO 设备无法访问并显示相应的系统诊断信息。

如果随后可再次访问 PROFINET IO 设备，就会引起标准的系统行为。

说明

对 PROFINET IO 设备进行激活可能较为耗时。如果要取消当前正在运行的激活作业，为 LADDR 和 MODE 输入相同值 2 并运行 D_ACT_DP 指令。为 MODE 输入值 2，重复调用 D_ACT_DP，直至显示 RET_VAL 等于 0，指示激活的作业已成功取消。

参数

下表列出了 D_ACT_DP 指令的参数：

参数	声明	数据类型	说明
REQ	IN	Bool	电平触发控制参数 REQ = 1：启用或禁用运行
MODE	IN	USInt	作业标识符 可能值： • 0：请求寻址组件的状态信息（激活/取消激活）（使用 RET_VAL 参数进行输出） • 1：激活 PROFINET IO 设备 • 2：取消激活 PROFINET IO 设备

参数	声明	数据类型	说明
LADDR	IN	HW_DEVICE	PROFINET IO 设备的硬件标识符 (HW_Device) 可以从网络视图中的 PROFINET IO 设备属性或标准变量表的 “系统常量”(System constants) 选项卡中获得编号。 如果该处同时指定了设备诊断的标识符以及操作状态转换的 ID，则必须使用设备诊断的代码。
RET_VAL	OUT	Int	在指令执行过程中如果发生错误，则返回值将包含错误代码。
BUSY	OUT	Bool	有效代码： ● BUSY = 1：作业仍处于激活状态。 ● BUSY = 0：作业已终止。

参数 RET_VAL

错误代码* (W#16#...)	说明
0000	作业已经成功完成。
0001	PROFINET IO 设备处于启用状态（该错误代码仅适用于 MODE = 0 时的情况）。
0002	PROFINET IO 设备处于取消激活状态（该错误代码仅适用于 MODE = 0 时的情况）。
7000	首次调用时，REQ = 0：在 LADDR 中指定的作业未激活，BUSY 的值为“0”。
7001	首次调用时，REQ = 1。程序触发 LADDR 中指定的作业，BUSY 的值为“0”。
7002	中间调用（与 REQ 无关）。激活的作业仍处于激活状态；BUSY = 1。
8090	● 没有使用 LADDR 指定的地址组态模块。 ● 将 CPU 作为智能从站/智能设备运行，并在 LADDR 中指定了该智能从站/智能设备的地址。
8092	取消激活当前寻址的 PROFINET IO 设备（MODE =2）不能通过激活取消（MODE =1）。在稍后时间启用组件。
8093	LADDR 中指定的地址不属于任何可激活或取消激活的 PROFINET IO 设备，或 MODE 参数未知。
8094	已尝试激活作为工具更换端口潜在通信伙伴的装置。但此时，已有其它装置在该端口启用。激活的装置仍保持激活状态。
80A0	CPU 和 IO 控制器的通信期间产生错误。

错误代码* (W#16#...)	说明
80A1	无法为寻址的组件分配参数。（仅当 MODE = 1 时，才可能出现该错误代码。） 注：如果所激活的设备进行参数分配时，该组件再次发生故障，则 D_ACT_DP 指令将返回该错误信息。如果某个模块参数分配失败，则 D_ACT_DP 将返回错误信息 W#16#0000。
80A3	相关 PROFINET IO 控制器不支持该功能。
80A4	对于外部 PROFINET IO 控制器，CPU 不支持该功能。
80A6	PROFINET IO 设备中的插槽错误；无法访问所有用户数据（只有在 MODE = 1 时，才会出现该错误代码）。 注：只有在分配参数后及 D_ACT_DP 指令执行结束之前激活的组件再次故障，D_ACT_DP 才返回该错误信息。如果仅一个单个模块不可用，D_ACT_DP 返回错误信息 W#16#0000。
80A7	在激活过程中发生超时错误：远程设备无法访问，或为集中式和分布式 I/O 设置的参数分配时间过短。远程设备的状态为“已激活”(activated)，但无法访问。
80AA	激活时 PROFINET IO 设备中有错误：组态差异
80AB	激活时 PROFINET IO 设备中有错误：参数分配错误
80AC	激活时 PROFINET IO 设备中有错误：需要维护
80C1	D_ACT_DP 已启动且一直以另一地址运行（当 MODE = 1 和 MODE = 2 时，可能会产生该错误代码）。
80C3	• 临时资源错误：CPU 当前正在处理最大可能数量的激活和禁用作业 (8)。（仅当 MODE = 1 且 MODE = 2 时，才可能出现该错误代码。） • CPU 正忙于接收修改的组态。当前无法启用/禁用 PROFINET IO 设备。
80C6	PROFINET：重启时会放弃用户未收集的作业。
常见错误信息	有关如何获取错误信息，请参见 GET_ERROR_ID (页 314) 指令。
* 在程序编辑器中，错误代码可显示为整数或十六进制值。	

9.3.9 RDREC、WRREC 和 RALRM 的 STATUS 参数

输出参数 STATUS 包含被解释为 ARRAY[1...4] OF BYTE 的错误信息，其结构如下：

表格 9-85 STATUS 输出数组

数组元素	名称	说明
STATUS[1]	Function_Num	• B#16#00（如果无错误） • DPV1-PDU 的功能 ID： 如果发生错误，会对 , B#16#80 执行“逻辑或”运算（对于读取数据记录： B#16#DE；对于写入数据记录：B#16#DF）。如果未使用 DPV1 协议元素，则输出 B#16#C0 。
STATUS[2]	Error_Decode	错误 ID 的位置
STATUS[3]	Error_Code_1	错误 ID
STATUS[4]	Error_Code_2	制造商特定的错误 ID 扩展

表格 9-86 STATUS[2] 值

Error_decode (B#16#....)	源型	说明
00 到 7F	CPU	无错误或无警告
80	DPV1	因不符合 IEC 61158-6 而出错
81 到 8F	CPU	B#16#8x 表示在指令的第“x”个调用参数中存在错误。
FE、FF	DP 配置文件	配置文件特定错误

表格 9-87 STATUS[3] 值

Error_decode (B#16#....)	Error_code_1 (B#16#....)	解释 (DVP1)	说明
00	00		无错误，无警告
70	00	保留，拒绝	初始调用；未传送活动数据记录
	01	保留，拒绝	初始调用；已开始传送数据记录
	02	保留，拒绝	中间调用；已激活数据记录传送

Error_decode (B#16#....)	Error_code_1 (B#16#....)	解释 (DVP1)	说明
80	90	保留，通过	逻辑起始地址无效
	92	保留，通过	Variant 指针的类型非法
	93	保留，通过	通过 ID 或 F_ID 寻址的 DP 组件未组态。
	96		“RALRM (页 395)”不能提供 OB 启动信息、管理信息、文件头信息或其它中断信息。 对于以下 OB，可以使用 “DPNRM_DG (页 418)”指令异步读取相关 DP 从站的当前诊断消息帧（OB 启动信息中的地址信息）： ● 硬件中断 (页 91) ● 状态 (页 97)、更新 (页 97)或配置文件 (页 98) ● 诊断错误中断 (页 93) ● 拔出或插入模块 (页 95)
	A0	读取错误	读取模块时得到否定确认。
	A1	写错误	写入模块时得到否定确认
	A2	模块故障	第 2 层出现 DP 协议错误（例如，从站故障或总线故障）
	A3	保留，通过	● PROFIBUS DP： 直接数据链路映射器或用户接口/用户出现 DP 协议错误 ● PROFINET IO： 常规 CM 错误
	A4	保留，通过	通信总线上的通信中断
	A5	保留，通过	-
	A7	保留，通过	DP 从站或模块已被占用（临时错误）。
	A8	版本冲突	DP 从站或模块报告出现不兼容版本。
	A9	特性不受支持	特性不受 DP 从站或模块支持
	AA 到 AF	用户特定	DP 从站或模块报告其应用中出现制造商特定错误。请检查 DP 从站或模块的制造商提供的文档。
	B0	索引无效	模块中出现未知数据记录；非法数据记录编号 ≥ 256
	B1	写长度错误	RECORD 参数中的长度信息不正确。 ● 对于“RALRM”： AINFO 中的长度错误 注： 要立即访问有关如何解释“AINFO”返回缓冲区的信息，请参见 STEP 7 的在线信息系统。 ● 对于 “RDREC (页 387)”和 “WRREC (页 387)”： “MLEN”中的长度错误
	B2	无效插槽	组态的插槽未被占用。

Error_decode (B#16#....)	Error_code_1 (B#16#....)	解释 (DVP1)	说明
	B3	类型冲突	实际模块类型与指定的模块类型不匹配。
	B4	无效区域	DP 从站或模块报告对无效区域的访问。
	B5	状态冲突	DP 从站或模块未就绪
	B6	访问被拒绝	DP 从站或模块拒绝访问。
	B7	无效范围	DP 从站或模块报告参数或值的范围无效。
	B8	无效参数	DP 从站或模块报告参数无效。
	B9	无效类型	DP 从站或模块报告类型无效： • 对于“RDREC (页 387)”： 缓冲区过小（无法读取子网） • 对于“WRREC (页 387)”： 缓冲区过小（无法写入子网）
	BA 到 BF	用户特定	DP 从站或模块在访问时报告制造商特定错误。 请检查 DP 从站或模块的制造商提供的文档。
	C0	读限制冲突	• 对于“WRREC (页 387)”： 仅当 CPU 处于 STOP 模式时才能写入数据。 **注意**： 这意味着无法通过用户程序写入数据。 只能使用 PG/PC 在线写入数据。 • 对于“RDREC (页 387)”： 模块可发送数据记录，但没有数据，或仅当 CPU 处于 STOP 模式时才能读取数据。 **注意**： 如果仅当 CPU 处于 STOP 模式时才能读取数据，则用户程序无法进行评估。 在这种情况下，只能使用 PG/PC 在线读取数据。
	C1	写限制冲突	针对相同数据记录向模块发送的上一次写请求的数据尚未被该模块处理。
	C2	资源忙	模块正在处理 CPU 允许的最多作业数。
	C3	资源不可用	所需操作资源当前已被占用。
	C4		内部临时错误。 无法执行作业。 重复作业。 如果此错误经常发生，请检查安装的电气干扰源。
	C5		DP 从站或模块不可用
	C6		由于取消优先级而使数据记录传送被取消。
	C7		作业由于 DP 主站的暖启动或冷启动而中止。
	C8 到 CF		DP 从站或模块报告制造商特定资源错误。 请检查 DP 从站或模块的制造商提供的文档。

Error_decode (B#16#....)	Error_code_1 (B#16#....)	解释 (DVP1)	说明
	Dx	用户特定	DP 从站特定。 参见 DP 从站的说明。
81	00 到 FF		初始调用参数错误（对于“RALRM (页 395)”：MODE)
	00		非法工作模式
82	00 到 FF		第二个调用参数错误
88	00 到 FF		第八个调用参数错误（对于“RALRM (页 395)”：TINFO) **注：** 要立即访问有关如何解释“TINFO”返回缓冲区的信息，请参见 STEP 7 的在线信息系统。
	01		语法 ID 错误
	23		超出数量结构或目标区域过小
	24		范围 ID 错误
	32		DB/DI 号超出用户范围
	3A		区域 ID DB/DI 的 DB/DI 号为 NULL，或指定的 DB/DI 不存在。
89	00 到 FF		第九个调用参数错误（对于“RALRM (页 395)”：AINFO) **注：** 要立即访问有关如何解释“AINFO”返回缓冲区的信息，请参见 STEP 7 的在线信息系统。
	01		语法 ID 错误
	23		超出数量结构或目标区域过小
	24		范围 ID 错误
	32		DB/DI 号超出用户范围
	3A		区域 ID DB/DI 的 DB/DI 号为 NULL，或指定的 DB/DI 不存在。
8A	00 到 FF		第 10 个调用参数错误
8F	00 到 FF		第 15 个调用参数错误
FE、FF	00 到 FF		配置文件特定错误

数组元素 STATUS[4]

出现 DPV1 错误时，DP 主机会将 STATUS[4] 传递给 CPU 和指令。 如果没有 DPV1 错误，该值将被设置为 0，但对于 RDREC 有以下例外情况：

- 如果 MLEN > RECORD 中的目标区域长度，则 STATUS[4] 包含 RECORD 中的目标区域长度。
- 如果实际数据记录长度 < MLEN < RECORD 中的目标区域长度，则 STATUS[4] = MLEN。
- 如果必须设置 STATUS[4] > 255，则 STATUS[4] = 0

在 PROFINET IO 中，STATUS[4] 的值为 0。

9.3.10 其它

9.3.10.1 DPRD_DAT 和 DPWR_DAT（读/写一致性数据）

使用 DPRD_DAT（读取一致性数据）指令一致性地读取一个或多个字节的数据，使用 DPWR_DAT（写入一致性的数据）指令一致性地传送一个或多个字节的数据。可将 DPRD_DAT 和 DPWR_DAT 指令用于 PROFINET 和 PROFIBUS。

表格 9-88 DPRD_DAT 和 DPWR_DAT 指令

LAD/FBD	SCL	说明

	`ret_val := DPRD_DAT(`
	`laddr:=_word_in_,`
	`record=>_variant_out_);`
		使用 DPRD_DAT 指令从以下其中一个位置的模块或子模块中读取一个或多个字节的数据：
		• 本地基本 I/O
		• DP 从站
		• PROFINET I/O 设备
		CPU 传送一致读取的数据。如果数据传送过程中未出错，则 CPU 会将读取的数据输入到通过 RECORD 参数设置的目标区域中。目标区域的长度必须与通过 STEP 7 为所选模块组态的长度相同。执行 DPRD_DAT 指令时，只能访问一个模块或子模块的数据。传送开始于组态的起始地址。

	`ret_val := DPWR_DAT(`
	`laddr:=_word_in_,`
	`record:=_variant_in_);`
		使用 DPWR_DAT 指令可将 RECORD 中的数据一致性地传送到以下位置：
		• 本地基站中的已寻址模块或子模块
		• DP 标准从站
		• PROFINET I/O 设备
		源区域的长度必须与通过 STEP 7 为所选模块或子模块组态的长度相同。

- S7-1200 CPU 支持一致性外围 I/O 读取或写入 1、2 或 4 字节长度的数据。使用 DPRD_DAT 指令和 DPWR_DAT 指令分别一致性地读取和写入长度为 1、2 或 4 字节的数据。
- 您可以使用这些指令访问大小为 1 个或多个字节的数据区域。如果访问被拒绝，将出现错误代码 W#16#8090。
- PROFINET 支持最多 1024 字节的一致性数据。无需使用这些指令在 S7-1200 与 PROFINET 设备之间进行一致传送。

说明

如果正在对一致性数据使用 DPRD_DAT 和 DPWR_DAT 指令，则必须从过程映像自动更新中删除该一致性数据。有关详细信息，请参见“PLC 概念：用户程序的执行” (页 81)。

表格 9-89 参数

参数	声明	数据类型	说明
LADDR	IN	HW_IO (Word)	将要从中读取数据的模块的硬件 ID。 (DPRD_DAT) 将写入数据的模块的硬件 ID。(DPWR_DAT) 该硬件 ID 位于设备视图或系统常量的模块属性中。
RECORD	OUT	Variant	已读取的用户数据的目标区域 (DPRD_DAT) 或要写入的用户数据的源区域 (DPWR_DAT)。此区域的大小必须与通过 STEP 7 为所选模块组态的区域大小完全相同。
RET_VAL	OUT	Int	如果在此函数已激活情况下发生错误，则返回值中将包含一个错误代码。

DPRD_DAT 操作

可使用 LADDR 参数选择 DP 标准从站/PROFINET IO 设备的模块。如果寻址的模块上出现访问错误，则输出错误代码 W#16#8090。

可使用 RECORD 参数定义读取数据的目标范围：

- 目标范围长度至少应与所选模块的输入长度相同。仅传送输入；不考虑其它字节。如果从带有模块化组态或多个 DP 标识符的 DP 标准从站中读取，则在每次调用 DPRD_DAT 时，只能访问具有组态硬件标识符的模块的数据。如果所选的目标范围过小，则会在 RET_VAL 参数输出错误代码 W#16#80B1。
- 可以使用以下数据类型：Byte、Char、Word、DWord、Int、UInt、USInt、SInt、DInt、UDInt。在类型为 ARRAY 或 STRUCT 的用户定义类型 (UDT) 数据结构中也可使用这些数据类型。
- 不支持 STRING 数据类型。
- 如果数据传输期间未发生任何错误，则已读取的数据将输入到由 RECORD 参数定义的目标范围中。

DPWR_DAT 操作

可使用 LADDR 参数选择 DP 标准从站/PROFINET IO 设备的模块。如果寻址的模块上出现访问错误，则输出错误代码 W#16#8090。

可使用 RECORD 参数定义要被写入的数据的源范围：

- 源范围长度至少应与所选模块的输出长度相同。仅传送输出；不考虑其它字节。如果参数 RECORD 指定的源范围长度大于所组态模块的输出长度，那么最多只能传送最大输出长度的数据。如果参数 RECORD 中的源范围短于所组态模块的输出，则 RET_VAL 参数会输出错误代码 W#16#80B1。
- 可以使用以下数据类型：Byte、Char、Word、DWord、Int、UInt、USInt、SInt、DInt、UDInt。在类型为 ARRAY 或 STRUCT 的用户定义类型 (UDT) 数据结构中也可使用这些数据类型。
- 不支持 STRING 数据类型。
- 数据以同步方式传送，即，指令完成时写入过程即完成。

错误代码

表格 9-90　DPRD_DAT 和 DPWR_DAT 错误代码

错误代码 [1]	说明
0000	未出错
8090	适用于下列情况之一： ● 没有为指定的逻辑基址组态模块。 ● 忽略了有关一致性数据长度的限制。 ● 没有以十六进制格式在 LADDR 参数中输入起始地址。
8092	RECORD 参数支持如下数据类型：Byte, Char, Word, DWord, Int, UInt, USInt, SInt, DInt, UDInt, and arrays of these types。
8093	LADDR 中指定的逻辑地址处不存在可以从其中读取 (DPRD_DAT) 一致性数据或向其写入 (DPWR_DAT) 一致性数据的 DP 模块/PROFINET IO 设备。
80A0	访问 I/O 设备时检测到访问错误 (DPRD_DAT)。
80B1	指定的目标区域的长度 (DPRD_DAT) 或源区域的长度 (DPWR_DAT) 与通过 STEP 7 Basic 组态的用户数据的长度不同。
80B2	外部 DP 接口模块出现系统错误 (DPRD_DAT) 和 (DPWR_DAT)

[1] 当显示 DPRD_DAT 和 DPWR_DAT 错误代码时，使用数据类型 Word。

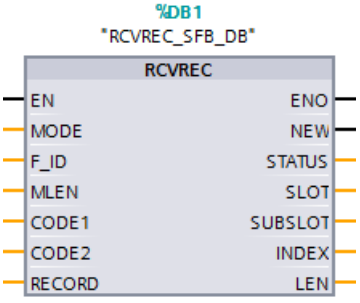
说明

如果访问 DPV1 从站，这些从站的错误信息可从 DP 主站转发到指令。

9.3.10.2 RCVREC（智能设备/智能从站接收数据记录）

智能设备可以从更高等级控制器接收数据记录。使用指令 RCVREC，可以在用户程序中进行接收（接收数据记录）。

表格 9-91 RCVREC 指令

LAD/FBD	SCL	说明
	"RCVREC_SFB_DB"(mode:=_int_in_, F_ID:=_uint_in_, mlen:=_uint_in_, code1:=_byte_in_, code2:=_byte_in_, new=>_bool_out_, status=>_dword_out_, slot=>_uint_out_, subslot=>_uint_out_, index=>_uint_out_ len=>_uint_out_, record:=_variant_inout_);	使用 RCVREC 指令从更高等级控制器接收数据记录。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“RCVREC_SFB_DB”是背景 DB 的名称。

该指令具有以下操作模式：

- 检查智能设备是否有数据记录接收请求
- 将数据记录提供给输出参数
- 给上一级控制器发送响应

使用输入参数 MODE，可以确定该指令执行的操作模式。

智能设备必须处于 RUN 或 STARTUP 模式。

使用 MLEN，可以指定要接收的最大字节数。目标范围 RECORD 的所选长度应至少为 MLEN 个字节。

如果已接收到数据记录（MODE = 1 或 MODE = 2），则输出参数 NEW 表示数据记录已存储在 RECORD 中。注意 RECORD 应有足够的长度。输出参数 LEN 存储所接收数据记录的实际长度，单位为[字节]。

将 CODE1 和 CODE2 置为 0，以给上一级控制器发送肯定响应。如果要拒绝接收的数据记录，可通过 CODE1 的错误代码 1 和 CODE2 的 错误代码 2，给上位控制器发送否定响应。

说明

如果智能设备已接收到数据记录接收请求，则必须确认该请求在特定时间内完成。确认后，必须在该时间段内，向上一级控制器发送响应。否则，智能设备会检测到超时错误，造成智能设备的操作系统向上一级控制器发送否定响应。有关该时间段的取值信息，请参见 CPU 技术规范。

发生错误后，输出参数 STATUS 接收到错误信息。

操作模式

使用输入参数 MODE，可以确定指令 RCVREC 的操作模式。此步骤的具体解释见下表：

MODE	含义
0	检查是否存在数据记录接收请求。 如果智能设备中存在上一级控制器的数据记录，则该指令将只写入 NEW、SLOT、SUBSLOT、INDEX 和 LEN 输出参数。如果在 MODE = 0 条件下多次调用该指令，则输出参数只引用唯一且相同的请求。
1	从智能设备的任何子插槽接收数据记录。 如果在智能设备中存在用于智能设备任何子插槽的上一级控制器的数据记录，则该指令只写入输出参数，并将数据记录传送到参数 RECORD。
2	从智能设备的特定子插槽接收数据记录 如果在智能设备中存在用于智能设备特定子插槽的上一级控制器的数据记录，则该指令只写入输出参数，并将数据记录传送到参数 RECORD。
3	向上一级控制器发送肯定响应 该指令检查上一级控制器的请求，以接收数据记录，接受现有数据记录，并向上一级控制器发送肯定响应。
4	向上一级控制器发送否定响应 该指令检查上一级控制器的请求，以接收数据记录，拒绝现有数据记录，并向上一级控制器发送否定响应。将拒绝原因输入到输入参数 CODE1 和 CODE2。

说明

接收到数据记录后（NEW = 1），必须调用 RCVREC 指令两次，以确保完全处理。必须按下面顺序操作：

- 首次调用时，MODE = 1 或 MODE = 2
- 第二次调用时，MODE = 3 或 MODE = 4

参数

下表列出了 RCVREC 指令的参数：

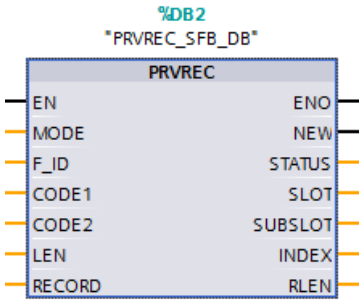
参数	声明	数据类型	说明
MODE	IN	Int	模式
F_ID	IN	HW_SUBMODULE	用于要接收数据记录的智能设备传送区域中的子插槽（仅对 MODE = 2）。高位字始终置为 0。
MLEN	IN	Int	要接收数据记录的最大长度，单位[字节]
CODE1	IN	Byte	零（针对 MODE = 3）和/或错误代码 1（针对 MODE = 4）
CODE2	IN	Byte	零（针对 MODE = 3）和/或错误代码 2（针对 MODE = 4）
NEW	OUT	Bool	• MODE = 0：收到新的数据记录 • MODE = 1 或 2：数据记录已传送到 RECORD
STATUS	OUT	DWord	错误信息。有关详细信息，请参见“状态参数” (页 404)。
SLOT	OUT	HW_SUBMODULE	与 F_ID 相同
SUBSLOT	OUT	HW_SUBMODULE	与 F_ID 相同
INDEX	OUT	UInt	接收的数据记录数
LEN	OUT	UInt	接收的数据记录长度
RECORD	IN_OUT	Variant	接收的数据记录所在的目标范围

9.3.10.3 PRVREC（智能设备/智能从站使数据记录可用）

智能设备可以从上一级控制器接收请求，以使数据记录可用。使用指令 PRVREC，智能设备可使数据记录在用户程序中可用（使数据记录可用）。

表格 9-92 PRVREC 指令

LAD/FBD	SCL	说明

	`"PRVREC_SFB_DB"(`
	`mode:=_int_in_,`
	`F_ID:=_uint_in_,`
	`code1:=_byte_in_,`
	`code2:=_byte_in_,`
	`len:=_uint_in_,`
	`new=>_bool_out_,`
	`status=>_dword_out_,`
	`slot=>_uint_out_,`
	`subslot=>_uint_out_,`
	`index=>_uint_out_`
	`rlen=>_uint_out_,`
	`record:=_variant_inout_);`
		使用 PRVREC 指令从上一级控制器接收请求，以使数据记录可用。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“PRVREC_SFB_DB”是背景 DB 的名称。

该指令具有以下操作模式：

- 检查智能设备是否有使数据记录可用的请求
- 将请求的数据记录传送到上一级控制器
- 给上一级控制器发送响应

使用输入参数 MODE，可以确定该指令执行的操作模式。

智能设备必须处于 RUN 或 STARTUP 模式。

通过 LEN 输入要发送数据记录应具有的最大字节数。目标范围 RECORD 的所选长度应至少为 LEN 个字节。

如果存在使数据记录可用请求（MODE = 0），则输出参数 NEW 被设置为 TRUE。

如果已接受使数据记录可用请求，则将肯定响应的 RECORD 写入具有请求数据记录的上一级控制器，并将 0 写入 CODE1 和 CODE2。如果要拒绝使数据记录可用的请求，可通过 CODE1 的错误代码 1 和 CODE2 的 错误代码 2，给上一级控制器发送否定响应。

说明

如果智能设备已接收到使数据记录可用请求，则必须确认该请求在特定时间内完成。确认后，必须在该时间段内，向上一级控制器发送响应。否则，智能设备会检测到超时错误，造成智能设备的操作系统向上一级控制器发送否定响应。有关该时间段的取值信息，请参见 CPU 技术规范。

发生错误后，输出参数 STATUS 接收到错误信息。

操作模式

使用输入参数 MODE，可以确定指令 PRVREC 的操作模式。此步骤的具体解释见下表：

MODE	含义
0	检查是否存在使数据记录可用请求。 如果智能设备中存在来自上一级控制器使数据记录可用的请求，则该指令将只写入 NEW、SLOT、SUBSLOT、INDEX 和 RLEN 输出参数。如果在 MODE = 0 条件下多次调用该指令，则输出参数只引用唯一且相同的请求。
1	从智能设备的任何子插槽接收使数据记录可用请求。 如果在智能设备中存在用于智能设备任何子插槽、且来自上一级控制器的此类请求，则该指令只写入输出参数。
2	从智能设备的特定子插槽接收使数据记录可用请求。 如果在智能设备中存在用于智能设备特定子插槽、且来自上一级控制器的此类请求，则该指令只写入输出参数。
3	使数据记录可用，并向上一级控制器发送肯定响应。 该指令检查上一级控制器的请求，以使数据记录可用，将请求的数据记录提供给 RECORD，并向上一级控制器发送肯定响应。
4	向上一级控制器发送否定响应 该指令检查上一级控制器的请求，以使数据记录可用，拒绝该请求，并向上一级控制器发送否定响应。将拒绝原因输入到输入参数 CODE1 和 CODE2。

说明

接收到请求后（NEW = 1），必须调用 PRVREC 指令两次，以确保完全处理。必须按下面顺序操作：

- 首次调用时，MODE = 1 或 MODE = 2
- 第二次调用时，MODE = 3 或 MODE = 4

参数

下表列出了 PRVREC 指令的参数：

参数	声明	数据类型	说明
MODE	IN	Int	模式
F_ID	IN	HW_SUBMODULE	用于要发送数据记录的智能设备传送区域中的子插槽（仅对 MODE = 2）。高位字始终置为 0。
CODE1	IN	Byte	零（针对 MODE = 3）和/或错误代码 1（针对 MODE = 4）
CODE2	IN	Byte	零（针对 MODE = 3）和/或错误代码 2（针对 MODE = 4）
LEN	IN	UInt	要发送数据记录的最大长度，单位[字节]
NEW	OUT	Bool	上一级控制器已请求新的数据记录。
STATUS	OUT	DWord	错误信息。有关详细信息，请参见“状态参数” (页 404)。
SLOT	OUT	HW_SUBMODULE	与 F_ID 相同
SUBSLOT	OUT	HW_SUBMODULE	与 F_ID 相同
INDEX	OUT	UInt	要发送数据记录的数量
RLEN	OUT	UInt	要发送数据记录的长度
RECORD	IN_OUT	Variant	数据记录可用

9.3.10.4 DPNRM_DG（读取 PROFIBUS DP 从站的诊断数据）

可以对 PROFIBUS 使用 DPNRM_DG（读取诊断数据）指令。

表格 9-93 DPNRM_DG 指令

LAD/FBD	SCL	说明

	`ret_val := DPNRM_DG(`
	`  req:=_bool_in_,`
	`  laddr:=_word_in_,`
	`  record=>_variant_out_,`
	`  busy=>_bool_out_);`
		使用 DPNRM_DG 指令将以“EN 50 170 第 2 卷，PROFIBUS”所指定的格式来读取 DP 从站的当前诊断数据。 在顺利完成数据传输后，已读取的数据被输入到由 RECORD 指定的目标区域。

表格 9-94 DPNRM_DG 指令的参数数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	REQ=1：读取请求
LADDR	IN	HW_DPSLAVE	所组态的 DP 从站诊断地址： 必须是该站点的地址，而不是 I/O 设备的地址。 在“网络”(Network) 视图的“设备组态”(Device configuration) 中，选择相应站（不是设备的图像）来确定诊断地址。 输入十六进制格式的地址。 例如，诊断地址 1022 表示为 LADDR:=W#16#3FE。
RET_VAL	OUT	Int	如果在此函数已激活情况下发生错误，则返回值中将包含一个错误代码。 如果没有错误，则将实际传输的数据长度输入 RET_VAL。
RECORD	OUT	Variant	已读取的诊断数据的目标区域。 要读取的数据记录（或目标区域）的最小长度为 6 字节。 要发送的数据记录的最大长度为 240 字节。 标准从站可提供大于 240 字节的诊断数据，最高可达 244 字节。 在这种情况下，将前 240 字节传送到目标区域，并在数据中设置溢出位。
BUSY	OUT	Bool	BUSY=1：读取作业还未完成

通过在调用 DPNRM_DG 指令时将值 1 赋给输入参数 REQ 启动读取作业。 读取作业异步执行，即需要多次调用 DPNRM_DG 指令。 此作业的状态由输出参数 RET_VAL 和 BUSY 指示。

表格 9-95　从站诊断数据结构

字节	说明
0	站状态 1
1	站状态 2
2	站状态 3
3	主站号
4	供应商 ID（高字节）
5	供应商 ID（低字节）
6 ...	附加从站诊断信息

表格 9-96　DPNRM_DG 指令的错误代码

错误代码	说明	限制
0000	无错误	-
7000	首次调用时 REQ = 0： 没有激活的数据传输；BUSY 的值为 0。	-
7001	首次调用时 REQ = 1： 没有激活的数据传输；BUSY 的值为 1。	分布式 I/O
7002	临时调用（与 REQ 无关）： 数据传送已经激活；BUSY 的值为 1。	分布式 I/O
8090	指定的逻辑基址无效： 无基址。	-
8092	RECORD 参数支持以下数据类型： Byte, Char, Word, DWord, Int, UInt, USInt, SInt, DInt, UDInt, and arrays of these types.	-
8093	• 通过 LADDR 指定的模块不允许使用此指令（适用于 S7-1200 的 S7-DP 模块允许使用）。 • LADDR 指定 I/O 设备，而不是站。 在 “网络”(Network) 视图的 “设备组态”(Device configuration) 中，选择相应站（不是设备的图像）来确定 LADDR 的诊断地址。	-
80A2	• 第 2 层出现 DP 协议错误（例如，从站故障或总线故障） • 对于 ET200S，无法在 DPV0 模式下读取数据记录。	分布式 I/O
80A3	用户接口/用户中出现 DP 协议错误	分布式 I/O
80A4	通信总线上出现通信故障	CPU 与外部 DP 接口模块之间发生错误。

错误代码	说明	限制
80B0	• 相应模块类型不支持此指令。 • 该模块不能识别数据记录。 • 不允许使用数据目录号 241。	-
80B1	在 RECORD 参数中指定的长度不正确。	指定长度 > 记录长度
80B2	组态的插槽未被占用。	-
80B3	实际模块类型与要求的模块类型不匹配。	-
80C0	无诊断信息。	-
80C1	模块尚未处理其中同一数据记录前一次写入作业的数据。	-
80C2	模块正在处理 CPU 允许的最多作业数。	-
80C3	所需资源（存储器等）当前被占用。	-
80C4	内部临时错误。 作业无法处理。 请重复该作业。 如果此错误频繁发生，请检查系统是否存在电干扰源。	-
80C5	分布式 I/O 不可用。	分布式 I/O
80C6	数据记录传送因优先等级中止（重启或后台）而停止	分布式 I/O
8xyy[1]	常规错误代码	

有关常规错误代码的更多信息，请参见“扩展指令，分布式 I/O： RDREC、WRREC 和 RALRM 的错误信息” (页 404)。

9.4 PROFIenergy

PROFIenergy 是一个使用 PROFINET 进行能源管理且与制造商和设备无关的配置文件。要降低生产间歇期和意外停产过程中的能源损耗，可使用 PROFIenergy 统一协同地关断相应设备。

PROFINET IO 控制器通过用户程序中的特殊命令关闭 PROFINET 设备/电源模块。无需附加硬件。PROFINET 设备可直接解译 PROFIenergy 命令。

S7-1200 CPU 不支持 PE 控制器功能。S7-1200 CPU 只能作为 PROFIenergy 实体（具有智能设备功能）。

PROFIenergy 控制器（PE 控制器）

PE 控制器为较高级别的 CPU（例如，S7-1500），可激活或禁用较低级别设备的空闲状态。PE 控制器使用用户程序，禁用或重新启用特定生产组件或整个生产线。下位空闲设备通过相应指令（函数块）接收来自用户程序的命令。

用户程序使用 PROFINET 通信协议发送命令。PI 命令可以是将 PE 实体切换为节能模式的控制命令，也可以是读取状态或测量值的命令。

可以使用 PE_I_DEV 指令请求模块中的数据。用户程序必须确定 PE 控制器正在请求的信息，并通过数据记录从能源模块中进行检索。模块本身不直接支持 PE 命令。模块将能源测量信息存储于共享区域，较低级别的 CPU（例如，S7-1200）会触发 PE_I_DEV 指令，以将其返回至 PE 控制器。

PROFIenergy 实体（PE 实体）

PE 实体（例如，S7-1200）接收 PE 控制器（例如，S7-1500）的 PROFIenergy 命令，然后相应地执行这些命令（例如，通过返回一个测量值或激活节能模式）。在具有 PROFIenergy 功能的设备中实现 PE 实体这一过程是特定于设备和制造商的。

参考信息

有关 PROFIenergy 的更多信息，请参见 TIA Portal STEP 7 在线帮助。有关使用 PROFIenergy 指令的示例，请参见工业在线支持的条目“PROFIenergy - 通过 SIMATIC S7 实现节能 (http://support.automation.siemens.com/CN/view/zh/41986454)”。

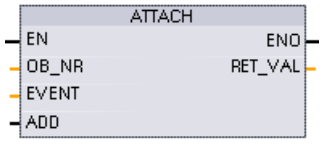
9.5 中断

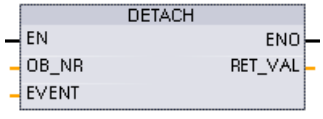
9.5.1 ATTACH 和 DETACH（附加/分离 OB 和中断事件）指令

使用 ATTACH 和 DETACH 指令可激活和禁用由中断事件驱动的子程序。

表格 9-97 ATTACH 和 DETACH 指令

LAD/FBD	SCL	说明

	`ret_val := ATTACH(`
	`ob_nr:=_int_in_,`
	`event:=_event_att_in_,`
	`add:=_bool_in_);`
		ATTACH 启用响应硬件中断事件的中断 OB 子程序执行。

	`ret_val := DETACH(`
	`ob_nr:=_int_in_,`
	`event:=_event_att_ in);`
		DETACH 禁用响应硬件中断事件的中断 OB 子程序执行。

表格 9-98　参数的数据类型

参数和类型		数据类型	说明
OB_NR	IN	OB_ATT	组织块标识符：从使用 “添加新块”(Add new block) 功能创建的可用硬件中断 OB 中进行选择。双击该参数域，然后单击助手图标可查看可用的 OB。
EVENT	IN	EVENT_ATT	事件标识符：从在 PLC 设备组态中为数字输入或高速计数器启用的可用硬件中断事件中进行选择。双击该参数域，然后单击助手图标可查看这些可用事件。
ADD （仅限 ATTACH）	IN	Bool	• ADD = 0（默认值）：该事件将取代先前为此 OB 附加的所有事件。 • ADD = 1：该事件将添加到先前为此 OB 附加的事件中。
RET_VAL	OUT	Int	执行条件代码

硬件中断事件

CPU 支持以下硬件中断事件：

- 上升沿事件：前 12 个内置 CPU 数字量输入（DIa.0 到 DIb.3）以及所有 SB 数字量输入
  - 数字输入从 OFF 切换为 ON 时会出现上升沿，以响应连接到输入的现场设备的信号变化。
- 下降沿事件：前 12 个内置 CPU 数字量输入（DIa.0 到 DIb.3）以及所有 SB 数字量输入
  - 数字输入从 ON 切换为 OFF 时会出现下降沿。
- 高速计数器 (HSC) 当前值 = 参考值 (CV = RV) 事件（HSC 1 至 6）
  - 当前计数值从相邻值变为与先前设置的参考值完全匹配时，会生成 HSC 的 CV = RV 中断。
- HSC 方向变化事件（HSC 1 至 6）
  - 当检测到 HSC 从增大变为减小或从减小变为增大时，会发生方向变化事件。
- HSC 外部复位事件（HSC 1 至 6）
  - 某些 HSC 模式允许分配一个数字输入作为外部复位端，用于将 HSC 的计数值重置为零。当该输入从 OFF 切换为 ON 时，会发生此类 HSC 的外部复位事件。

在设备组态期间启用硬件中断事件

必须在设备组态中启用硬件中断。如果要在组态或运行期间附加此事件，则必须在设备组态中为数字输入通道或 HSC 选中启用事件框。

PLC 设备组态中的复选框选项：

- 数字量输入
  - 启用上升沿检测
  - 启用下降沿检测
- 高速计数器 (HSC)
  - 启用此高速计数器
  - 生成计数器值等于参考计数值的中断
  - 生成外部复位事件的中断
  - 生成方向变化事件的中断

向用户程序添加新硬件中断 OB 代码块

默认情况下，第一次启用事件时，没有任何 OB 附加到该事件。“HW 中断：”(HW interrupt:) 设备组态“<未连接>”(<not connected>) 标签会对此加以指示。只有硬件中断 OB 能附加到硬件中断事件。所有现有的硬件中断 OB 都会出现在“HW 中断：”(HW interrupt:) 下拉列表中。如果未列出任何 OB，则必须按下列步骤创建类型为“硬件中断”的 OB。在项目树的“程序块”(Program blocks) 分支下：

1. 双击“添加新块”(Add new block)，选择“组织块 (OB)”(Organization block (OB))，然后选择“硬件中断”(Hardware interrupt)。
2. 也可以重命名 OB、选择编程语言（LAD、FBD 或 SCL）以及选择块编号（切换为手动并选择与建议块编号不同的块编号）。
3. 编辑该 OB，添加事件发生时要执行的已编程响应。可以从此 OB 调用嵌套最深的 FC 和 FB。安全程序的最大嵌套深度为四。对于其它程序，最大嵌套深度为六。

OB_NR 参数

所有现有的硬件中断 OB 名称都会出现在设备组态“HW 中断：”(HW interrupt:) 下拉列表和 ATTACH /DETACH 参数 OB_NR 下拉列表中。

EVENT 参数

启用某个硬件中断事件时，将为该事件分配一个唯一的默认事件名称。可以通过编辑“事件名称：”(Event name:) 编辑框更改该事件的名称，但该名称必须唯一。这些事件名称将成为“常量”(Constants) 变量表中的变量名称，并显示在 ATTACH 和 DETACH 指令框的 EVENT 参数下拉列表中。变量的值是用于标识事件的内部编号。

常规操作

每个硬件事件都可附加到一个硬件中断 OB 中，在发生该硬件中断事件时将排队执行该硬件中断 OB。在组态或运行期间可附加 OB 事件。

用户可以在组态时将 OB 附加到已启用的事件或使其与该事件分离。要在组态过程中向事件附加一个 OB，必须使用“HW 中断：”(HW interrupt:) 下拉列表（单击右侧的向下箭头），然后从可用的硬件中断 OB 列表中选择一个 OB。从该列表中选择相应的 OB 名称，或者选择“<未连接>”(<not connected>) 以删除该附加关系。

也可以在运行期间附加或分离已启用的硬件中断事件。在运行期间使用 ATTACH 或 DETACH 程序指令（如有必要可多次使用）将已启用的中断事件附加到相应的 OB 或与其分离。如果当前未附加到任何 OB（选择了设备组态中的“<未连接>”(<not connected>) 选项或由于执行了 DETACH 指令），则将忽略已启用的硬件中断事件。

DETACH 操作

使用 DETACH 指令将特定事件或所有事件与特定 OB 分离。如果指定了 EVENT，则仅将该事件与指定的 OB_NR 分离；当前附加到此 OB_NR 的任何其它事件仍保持附加状态。如果未指定 EVENT，则分离当前附加到 OB_NR 的所有事件。

条件代码

表格 9-99　条件代码

RET_VAL (W#16#....)	ENO	说明
0000	1	无错误
0001	1	没有要分离的事件（仅 DETACH）
8090	0	OB 不存在
8091	0	OB 类型错误
8093	0	事件不存在

9.5.2 循环中断

9.5.2.1 SET_CINT（设置循环中断参数）

表格 9-100 SET_CINT（设置循环中断参数）

LAD/FBD	SCL	说明

	`ret_val := SET_CINT(`
	`ob_nr:=_int_in_,`
	`cycle:=_udint_in_,`
	`phase:=_udint_in_);`
		设置特定的中断 OB 以开始循环中断程序扫描过程。

表格 9-101 参数的数据类型

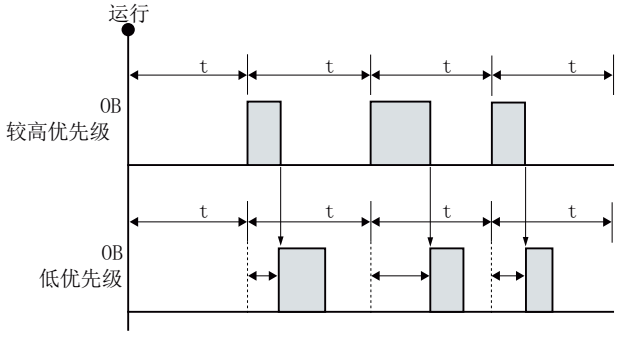
参数和类型		数据类型	说明
OB_NR	IN	OB_CYCLIC	OB 号（允许使用符号名称）
CYCLE	IN	UDInt	时间间隔（微秒）
PHASE	IN	UDInt	相移（微秒）
RET_VAL	OUT	Int	执行条件代码

示例：时间参数

- 如果 CYCLE 时间 = 100 us，则由 OB_NR 引用的中断 OB 将每隔 100 us 中断一次循环程序扫描。中断 OB 在执行后会将执行控制交回程序扫描过程，从而继续从中断位置开始扫描。
- 如果 CYCLE 时间 = 0，则中断事件被禁用，并且不会执行中断 OB。
- PHASE（相移）时间是 CYCLE 时间间隔开始前的指定延迟时间。可使用相移来控制优先级较低的 OB 的执行时间。

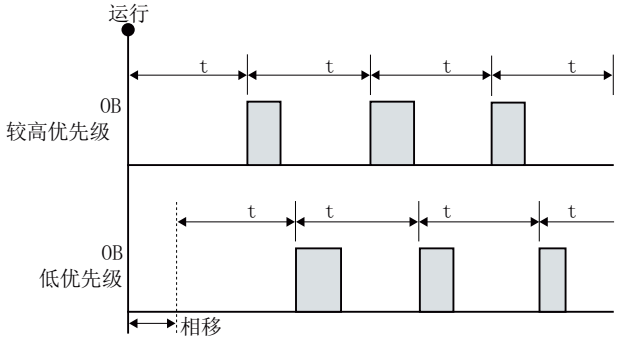
如果以相同的时间间隔调用优先级较高和优先级较低的 OB，则只有在优先级较高的 OB 完成处理后才会调用优先级较低的 OB。低优先级 OB 的执行起始时间会根据优先级较高的 OB 的处理时间来延迟。

没有相移的 OB 调用



如果希望以固定的时间周期来执行优先级较低的 OB，则相移时间应大于优先级较高的 OB 的处理时间。

有相移的 OB 调用



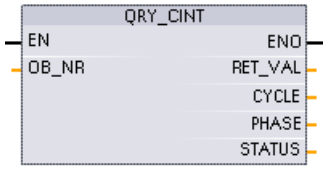
表格 9-102 条件代码

RET_VAL (W#16#....)	说明
0000	无错误
8090	OB 不存在或类型错误
8091	无效周期时间
8092	无效相移时间
80B2	OB 未附加事件

9.5.2.2 QRY_CINT（查询循环中断参数）

表格 9-103 QRY_CINT（查询循环中断）

LAD/FBD	SCL	说明

	`ret_val := QRY_CINT(`
	`ob_nr:=_int_in_,`
	`cycle=>_udint_out_,`
	`phase=>_udint_out__,`
	`status=>_word_out_);`
		获取循环中断 OB 的参数和执行状态。返回的值早在执行 QRY_CINT 时便已存在。

表格 9-104 参数的数据类型

参数和类型		数据类型	说明
OB_NR	IN	OB_CYCLIC	OB 号（允许使用类似 OB_MyOBName 的符号名称）
RET_VAL	OUT	Int	执行条件代码
CYCLE	OUT	UDInt	时间间隔（微秒）
PHASE	OUT	UDInt	相移（微秒）
STATUS	OUT	Word	循环中断状态代码： • 位 0 到 4，请参见下面的 STATUS 表 • 其它位，始终为 0

表格 9-105 STATUS 参数

位	值	说明
0	0	CPU RUN 期间
	1	启动过程中
1	0	中断已启用。
	1	中断已通过 DIS_IRT 指令禁用。
2	0	中断未激活或已过期。
	1	中断已激活。
4	0	通过 OB_NR 标识的 OB 不存在。
	1	通过 OB_NR 标识的 OB 存在。
其它位		始终为 0

如果发生错误，RET_VAL 显示相应的错误代买，并且参数 STATUS = 0。

表格 9-106 RET_VAL 参数

RET_VAL (W#16#....)	说明
0000	无错误
8090	OB 不存在或类型错误。
80B2	OB 未附加事件。

9.5.3 时钟中断

⚠ 警告

存在攻击者通过网络时间协议 (Network Time Protocol, NTP) 同步访问用户网络的风险

如果攻击者能通过网络时间协议 (NTP) 同步访问用户网络，那么便可能通过改变 CPU 系统时间来中断过程控制。过程控制中断可能造成死亡、重伤或财产损失。

默认情况下，S7-1200 CPU 的 NTP 客户端功能处于禁用状态，启用该功能时，仅允许将已组态的 IP 地址用作 NTP 服务器。CPU 默认禁用此功能，必须组态此功能才能实现远程控制 CPU 系统时间修正。

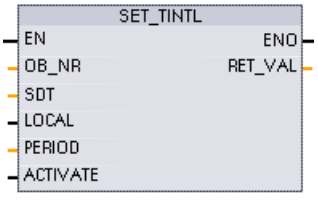
S7-1200 CPU 支持“日时钟”中断和时钟指令，这两个指令均依赖于精确的 CPU 系统时间。如果组态 NTP 并接受从服务器进行时间同步，那么必须确保服务器是可靠来源。否则会导致安全漏洞，从而使未知用户能够通过改变 CPU 系统时间来有限地控制您的过程。

有关安全信息和建议，请参见西门子服务与支持网站上的“工业安全操作准则 (http://www.industry.siemens.com/topics/global/en/industrial-security/Documents/operational_guidelines_industrial_security_en.pdf)”。

9.5.3.1 SET_TINTL（设置时钟中断）

表格 9-107 SET_TINTL（使用 DTL 数据类型设置日期和时钟中断）

LAD/FBD	SCL	说明

	`ret_val := SET_TINTL(`
	`OB_NR:=_int_in_,`
	`SDT:=_dtl_in_,`
	`LOCAL:=_bool_in_`
	`PERIOD:=_word_in_`
	`ACTIVATE:=_bool_in_);`
		设置日期和时钟中断。程序中断 OB 可以设置为执行一次，或者在分配的时间段内多次执行。

表格 9-108 参数的数据类型

参数和类型		数据类型	说明
OB_NR	IN	OB_TOD (INT)	OB 号（允许使用符号名称）
SDT	IN	DTL	启动日期和时间：将忽略秒和毫秒，并且可设置为 0。
LOCAL	IN	Bool	0 = 使用系统时间 1 = 使用本地时间（条件是 CPU 组态为本地时间，否则使用系统时间）
PERIOD	IN	Word	从起始日期和时间到再次发生中断事件的时段。 • W#16#0000 = 一次 • W#16#0201 = 每分钟 • W#16#0401 = 每小时 • W#16#1001 = 每天 • W#16#1201 = 每周 • W#16#1401 = 每月 • W#16#1801 = 每年 • W#16#2001 = 月末
ACTIVATE	IN	Bool	0 = 必须执行 ACT_TINT 才能激活中断事件。 1 = 中断事件已激活。
RET_VAL	OUT	Int	执行条件代码

程序可以使用 SET_TINTL 设置将执行分配的中断 OB 的日期和时钟中断事件。起始日期和时间由参数 SDT 设置，再次发生中断的时间段（如，每天或每周）由参数 PERIOD 设置。如果将重复周期设置为每月，则必须将起始日期设置为 1 号到 28 号中的一天。由于二月份没有 29 号到 31 号，因此不能使用这些值。如果希望在每月末发生中断事件，则将月末用于参数 PERIOD。

忽略参数 SDT 中的 DTL 数据工作日值。从在线 CPU 的 “在线和诊断”(Online & diagnostics) 视图中，使用 “设置日时钟”(Set time of day) 功能设置 CPU 的当前日期和时间。必须设置年、月、日。STEP 7 根据 CPU 的日期和时间时钟计算中断的时间间隔。

说明

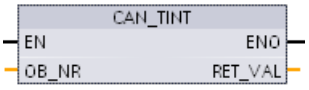
从夏天更改为冬天（夏令时）时，当天的第一个小时不存在。使用的起始时间应该从第二个小时开始，或者在第一个小时内使用附加的延时中断。

表格 9-109 条件代码

RET_VAL (W#16#....)	说明
0000	无错误
8090	无效的 OB_NR 参数
8091	无效的 SDT 起始时间参数： （例如，夏令时开始时跳过的小时内的起始时间）
8092	无效的 PERIOD 参数
80A1	该起始时间已过。 （仅在 PERIOD = W #16#0000 时发生该错误代码。）

9.5.3.2 CAN_TINT（取消时钟中断）

表格 9-110 CAN_TINT（取消日期和时钟中断）

LAD/FBD	SCL	说明
	`ret_val:=CAN_TINT(_int_in);`	为指定的中断 OB 取消起始日期和时钟中断事件。

表格 9-111 参数的数据类型

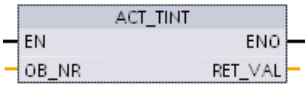
参数和类型		数据类型	说明
OB_NR	IN	OB_TOD (INT)	OB 号（允许使用符号名称）
RET_VAL	OUT	Int	执行条件代码

表格 9-112 条件代码

RET_VAL (W#16#....)	说明
0000	无错误
8090	无效的 OB_NR 参数
80A0	无起始日期/为中断 OB 设置的时间

9.5.3.3 ACT_TINT（激活时钟中断）

表格 9-113 ACT_TINT（激活日期和时钟中断）

LAD/FBD	SCL	说明
	`ret_val:=ACT_TINT(_int_in_);`	为指定的中断 OB 激活起始日期和时钟中断事件。

表格 9-114 参数的数据类型

参数和类型		数据类型	说明
OB_NR	IN	OB_TOD (INT)	OB 号（允许使用符号名称）
RET_VAL	OUT	Int	执行条件代码

表格 9-115 条件代码

RET_VAL (W#16#....)	说明
0000	无错误
8090	无效的 OB_NR 参数
80A0	没有为相关的时钟中断 OB 设置启动日期和时钟。
80A1	激活的时间已过。仅当设置为执行一次中断 OB 时发生该错误。

9.5.3.4 QRY_TINT（查询时钟中断状态）

表格 9-116 QRY_TINT（查询日期和时钟中断）

LAD/FBD	SCL	说明

	`ret_val:=QRY_TINT(`
	`  OB_NR:=_int_in_,`
	`  STATUS=>_word_out_);`
		为指定的中断 OB 查询日期和时钟中断状态。

表格 9-117 参数的数据类型

参数和类型		数据类型	说明
OB_NR	IN	OB_TOD (INT)	要查询的中断 OB 的 OB 号（允许使用符号名称）
RET_VAL	OUT	Int	执行条件代码
STATUS	OUT	Word	指定的中断 OB 的状态

表格 9-118 STATUS 参数

位	值	说明
0	0	运行中
	1	在启动过程中
1	0	中断已启用。
	1	中断已禁用。
2	0	中断未激活或已过期。
	1	中断已激活。
4	0	分配的 OB_NR 不存在。
	1	存在具有分配的 OB_NR 的 OB。
6	1	日期和时钟中断使用本地时间。
	0	日期和时钟中断使用系统时间。
其它		始终为 0

表格 9-119 条件代码

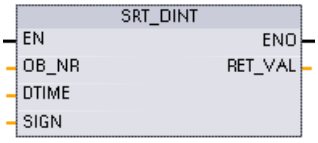
RET_VAL (W#16#....)	说明
0000	无错误
8090	无效的 OB_NR 参数

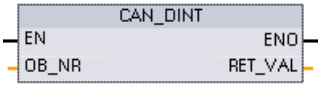
9.5.4 延时中断

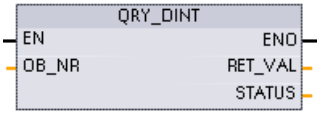
可使用 SRT_DINT 和 CAN_DINT 指令启动和取消延时中断处理过程，或使用 QRY_DINT 指令查询中断状态。每个延时中断都是一个在指定的延迟时间过后发生的一次性事件。如果在延迟时间到期前取消延时事件，则不会发生程序中断。

表格 9-120 SRT_DINT、CAN_DINT 和 QRY_DINT 指令

LAD/FBD	SCL	说明

	`ret_val := SRT_DINT(`
	`ob_nr:=_int_in_,`
	`dtime:=_time_in_,`
	`sign:=_word_in_);`
		SRT_DINT 启动延时中断，在参数 DTIME 指定的延迟过后执行 OB。

	`ret_val := CAN_DINT(` `ob_nr:=_int_in_);`	CAN_DINT 取消已启动的延时中断。在这种情况下，将不执行延时中断 OB。

	`ret_val := QRY_DINT(`
	`ob_nr:=_int_in_,`
	`status=>_word_out_);`
		QRY_DINT 查询通过 OB_NR 参数指定的延时中断的状态。

表格 9-121 参数的数据类型

参数和类型		数据类型	说明
OB_NR	IN	OB_DELAY	将在延迟时间过后启动的组织块 (OB)：从使用 “添加新块”(Add new block) 项目树功能创建的可用延时中断 OB 中进行选择。双击该参数域，然后单击助手图标可查看可用的 OB。
DTIME [1]	IN	Time	延迟时间值（1 到 60000 ms）
SIGN [1]	IN	Word	S7-1200 不使用：接受任何值。为避免发生错误，必须指定一个值。
RET_VAL	OUT	Int	执行条件代码
STATUS	OUT	Word	QRY_DINT 指令：所指定延时中断 OB 的状态，请参见下文表格

[1] 仅限 SRT_DINT

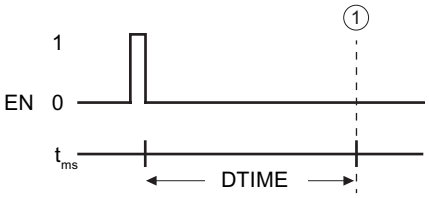
操作

当 EN=1 时，SRT_DINT 指令启动内部时间延时定时器 (DTIME)。延迟时间过去后，CPU 将生成可触发相关延时中断 OB 执行的程序中断。在指定的延时发生之前执行 CAN_DINT 指令可取消进行中的延时中断。激活延时中断事件的总次数不得超过四次。

说明

当 EN=1 时，SRT_DINT 会在每次扫描时开启时间延时定时器。断言 EN=1 作为单触发而不是设置 EN=1 开始延时。

SRT_DINT 指令的时序图：



① 延时中断执行

在项目中添加延时中断 OB

只能将延时中断 OB 分配给 SRT_DINT 和 CAN_DINT 指令。新项目中不存在延时中断 OB。必须将延时中断 OB 添加到项目中。要创建延时中断 OB，请按以下步骤操作：

1. 在项目树的“程序块”(Program blocks) 分支中双击“添加新块”(Add new block)，选择“组织块 (OB)”(Organization block (OB))，然后选择“延时中断”(Time delay interrupt)。
2. 可以重命名 OB、选择编程语言或选择块编号。如果要分配与自动分配的编号不同的块编号，请切换到手动编号模式。
3. 编辑延时中断 OB 子程序，并创建要在发生延时超时事件时执行的已编程响应。可从延时中断 OB 调用其它的 FC 和 FB 代码块。安全程序的最大嵌套深度为四。对于其它程序，最大嵌套深度为六。
4. 编辑 SRT_DINT 和 CAN_DINT 指令的 OB_NR 参数时，将可以使用新分配的延时中断 OB 名称。

QRY_DINT 参数 STATUS

表格 9-122　如果存在错误 (REL_VAL <> 0)，则 STATUS = 0。

位	值	说明
0	0	处于 RUN 状态
	1	在启动过程中

位	值	说明
1	0	中断已启用。
	1	中断已禁用。
2	0	中断未激活或已过期。
	1	中断已激活。
4	0	不存在具有 OB_NR 中所指定 OB 号的 OB。
	1	存在具有 OB_NR 中所指定 OB 号的 OB。
其它位		始终为 0

条件代码

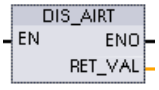
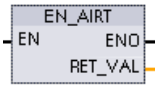
表格 9-123 SRT_DINT、CAN_DINT 和 QRY_DINT 的条件代码

RET_VAL (W#16#...)	说明
0000	未出错
8090	不正确的参数 OB_NR
8091	不正确的参数 DTIME
80A0	未启动延时中断。

9.5.5 DIS_AIRT 和 EN_AIRT（延迟/启用较高优先级的中断和异步错误事件）指令

使用 DIS_AIRT 和 EN_AIRT 指令可禁用和启用报警中断处理过程。

表格 9-124 DIS_AIRT 和 EN_AIRT 指令

LAD/FBD	SCL	说明
	`DIS_AIRT();`	DIS_AIRT 可延迟新中断事件的处理。可在 OB 中多次执行 DIS_AIRT。
	`EN_AIRT();`	对先前使用 DIS_AIRT 指令禁用的中断事件处理，可使用 EN_AIRT 来启用。每一次 DIS_AIRT 执行都必须通过一次 EN_AIRT 执行来取消。 必须在同一个 OB 中或从同一个 OB 调用的任意 FC 或 FB 中完成 EN_AIRT 执行后，才能再次启用此 OB 的中断。

表格 9-125 参数的数据类型

参数和类型		数据类型	说明
RET_VAL	OUT	Int	延迟次数 = 队列中的 DIS_AIRT 执行次数。

由操作系统会统计 DIS_AIRT 执行的次数。在特别通过 EN_AIRT 指令再次取消之前或者在已完成处理当前 OB 之前，这些执行中的每一个都保持有效。例如：如果通过五次 DIS_AIRT 执行禁用中断五次，则在再次启用中断前，必须通过五次 EN_AIRT 执行来取消禁用。

再次启用中断事件后，将处理 DIS_AIRT 生效期间发生的中断；或者在完成执行当前 OB 后，立即处理中断。

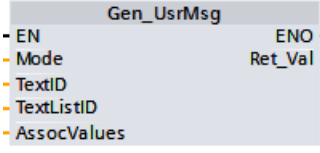
参数 RET_VAL 表示禁用中断处理的次数，即已排队的 DIS_AIRT 执行的个数。只有当参数 RET_VAL = 0 时，才会再次启用中断处理。

9.6 报警

9.6.1 Gen_UsrMsg（生成用户诊断报警）

表格 9-126 Gen_UsrMsg 指令

LAD/FBD	SCL	说明

	`ret_val :=Gen_UsrMsg(`
	`Mode:=_uint_in_,`
	`TextID:=_uint_in_,`
	`TextListID:=_uint_in_,`
	`AssocValues:=_struct_inout_);`
		使用“Gen_UsrMsg”指令生成用户诊断报警，可以是到达的报警也可以是离去的报警。通过用户诊断报警，可以将用户条目写入诊断缓冲区并发送相应报警。
		条目在诊断缓冲区中同时创建，而报警却将进行异步传送。
		如果指令在执行过程中出错，则将在参数 RET_VAL 处输出该错误。

报警的内容

文本列表定义该报警的内容：

- 通过参数 TextListID 定义要使用的文本列表。为此，需在项目导航中打开对话框“文本列表”(Text lists)。在对话框“文本列表”(Text lists) 中，将显示列“ID”。在参数 TextListID 中应用该 ID。
- 使用参数 TextID 选择要写入诊断缓冲区的文本列表条目。为此，可通过应用参数 TextID 中“起始范围/终止范围”(Range from / range to) 列的数字值，从“文本列表条目”(Text lists entries) 中选择一个条目。在文本列表条目中，“起始范围”和“终止范围”列的值必须相同。

有关文本列表的详细信息，请参见 STEP 7 信息系统。

定义关联值

文本列表条目可定义待添加到报警的新相关值：

- 添加以下信息到文本列表条目定义相关值：
  @<关联值的数量><元素类型><格式规范>@
- 使用系统数据类型 AssocValues 指定在生成报警时要添加的相关值。

有关相关值结构的详细信息，请参见 STEP 7 信息系统。

参数

下表列出了“Gen_UsrMsg”指令的参数：

参数	声明	数据类型	存储区	说明
Mode	Input	UInt	I、Q、M、D、L 或常数	用于选择报警状态的参数： • 1：到达的报警 • 2：离去的报警
TextID	Input	UInt	I、Q、M、D、L 或常数	将用于报警文本的文本列表条目 ID。
TextListID	Input	UInt	I、Q、M、D、L 或常数	包含文本列表条目的文本列表 ID。

参数	声明	数据类型	存储区	说明
Ret_Val	Return	Int	I、Q、M、D、L	指令的错误代码。
AssocValues	InOut	VARIANT	D、L	指向允许定义相关值的系统数据类型 AssocValues 的指针。

有关有效数据类型的更多信息，请参见 “数据类型 (页 119)”。

参数 AssocValues

通过系统数据类型 AssocValues，可定义待发送的相关值。所允许的最大相关值为 8。可通过将数据块的数据类型设置为“AssocValues”创建结构。

并通过在参数 Value[x] 中输入相关值的值选择相关值。注意事项：

- Gen_UsrMsg 指令将 TextID 和 TextListID 的值作为待发送的相关值。因此，分配“1”和“2”作为寻址关联值的数字。不得使用数字“1”或“2”寻址关联值。
- 通过参数 Value [1] 以数字“3”形式寻址相关值，通过参数 Value [2] 以数字“4”形式寻址相关值，以此类推。

字节	参数	数据类型	起始值	说明	关联值的编号
0..1	Value[1]	UINT	0	报警的第一个相关值。	3
2..3	Value[2]	UINT	0	报警的第二个相关值。	4
4..5	Value[3]	UINT	0	...	5
6..7	Value[4]	UINT	0	...	6
8..9	Value[5]	UINT	0	...	7
10..11	Value[6]	UINT	0	...	8
12..13	Value[7]	UINT	0	...	9
14..15	Value[8]	UINT	0	报警的第八个相关值。	10

参数 RET_VAL

下表定义了 RET_VAL 参数的输出值。另请参见“扩展指令的常见错误代码 (页 557)”。

错误代码* (W#16#...)	说明
0000	无错误
8080	不支持 MODE 参数中的值。
80C1	并行调用过多，导致资源瓶颈。
8528	参数 5 (AssocValues) 不是整字节。
853A	参数 5 (AssocValues) 引用无效的点。
* 在程序编辑器中，错误代码可显示为整数或十六进制。	

9.7 诊断（PROFINET 或 PROFIBUS）

9.7.1 诊断指令

以下诊断指令适用于 PROFINET 或 PROFIBUS：

- RD_SINFO 指令 (页 440)：读取当前 OB 启动信息
- LED 指令 (页 451)：读取分布式 I/O 设备的 LED 状态。
- Get_IM_Data 指令 (页 452)：检查指定模块或子模块的标识和维护 (I&M) 数据。
- Get_Name 指令 (页 454)：读取 PROFINET IO 设备、PROFIBUS 从站或 AS-i slave 的名称。
- GetStationInfo 指令 (页 460)：读取位于本地 IO 系统中 PROFINET IO 设备或下级 IO 系统中 PROFINET IO 设备的 IP 或 MAC 地址（使用 CP/CM 模块连接）。
- DeviceStates 指令 (页 468)：获取 I/O 子系统中分布式 I/O 设备的运行状态。
- ModuleStates 指令 (页 474)：获取分布式 I/O 设备中各模块的运行状态。
- GET_DIAG 指令 (页 480)：从指定的设备读取诊断信息。

说明

只能将 GetStationInfo 指令用于 PROFINET IO 设备。不能使用 PROFIBUS DP 从站的指令。

9.7.2 RD_SINFO（读取当前 OB 启动信息）

说明

表格 9-127 RD_SINFO 指令

LAD/FBD	SCL	说明

	`ret_val := RD_SINFO(`
	`TOP_SI=>_variant_out_,`
	
	`START_UP_SI=>_variant_out_)`
	`;`
		使用指令“RD_SINFO”读取下列 OB 的启动信息。
		● 上一次调用的但尚未执行完成的 OB
		● 上一次 CPU 启动的启动 OB
		两种情况下都没有时间戳。如果在 OB 100、OB 101 或 OB 102 中进行调用，则将返回两个相同的启动信息。

参数

下表列出了“RD_SINFO”指令的参数：

参数	声明	数据类型	存储区	说明
RET_VAL	Return	INT	I、Q、M、D、L	错误信息
TOP_SI	Output	VARIANT	D、L	当前 OB 的启动信息
START_UP_SI	Output	VARIANT	D、L	最后启动的启动 OB 的启动信息

有关有效数据类型的更多详细信息，请参见 “数据类型 (页 119)”。

参数 TOP_SI 的 SDT

下表列出了参数 TOP_SI 的 SDT：

组织块 (OB)	系统数据类型 (SDT)	系统数据类型编号
任意	SI_classic*	592*
	SI_none	593
ProgramCycleOB	SI_ProgramCycle	594
TimeOfDayOB	SI_TimeOfDay	595
TimeDelayOB	SI_Delay	596

组织块 (OB)	系统数据类型 (SDT)	系统数据类型编号
CyclicOB	SI_Cyclic	597
ProcessEventOB	SI_HWInterrupt	598
ProfileEventOB StatusEventOB UpdateEventOB	SI_Submodule	601
SynchronousCycleOB	SI_SynchCycle	602
IOredundancyErrorOB	SI_IORedundancyError	604
CPUredundancyErrorOB	SI_CPURedundancyError	605
TimeErrorOB	SI_TimeError	606
DiagnosticErrorOB	SI_DiagnosticInterrupt	607
PullPlugEventOB	SI_PlugPullModule	608
PeripheralAccessErrorOB	SI_AccessError	609
RackStationFailureOB	SI_StationFailure	610
ServoOB	SI_Servo	611
IpoOB	SI_Ipo	612
StartupOB	SI_Startup	613
ProgrammingErrorOB IOaccessErrorOB	SI_ProgIOAccessError	614

*The SI_classic SDT 不适用于 S7-1200。如果 TOP_SI 参数的类型为 SI_classic，那么 S7-1200 CPU 会返回一个 #16#8081 的 RET_VAL。

参数 START_UP_SI 的 SDT

下表列出了参数 START_UP_SI 的 SDT：

系统数据类型 (SDT)	系统数据类型编号
SI_classic*	592
SI_none	593
SI_Startup	613

*The SI_classic SDT 不适用于 S7-1200。如果 START_UP_SI 参数的类型为 SI_classic，那么 S7-1200 CPU 会返回一个 #16#8083 的 RET_VAL。

结构

下表定义了各结构的结构元素：

表格 9-128 SI_classic 结构

结构元素	数据类型	说明
EV_CLASS	BYTE	• 位 0 至 3：事件 ID • 位 4 至 7：事件类别
EV_NUM	BYTE	事件编号
PRIORITY	BYTE	优先级编号 （B#16#FE 的含义：OB 不可用或已禁用，或无法在当前操作模式中启动）
NUM	BYTE	OB 编号
TYP2_3	BYTE	数据 ID 2_3： 标识在 ZI2_3 中输入的信息
TYP1	BYTE	数据 ID 1： 标识在 ZI1 中输入的信息
ZI1	WORD	附加信息 1
ZI2_3	DWORD	附加信息 2_3

表格 9-129 SI_none 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）

表格 9-130 SI_ProgramCycle 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 1	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）

结构元素	数据类型	说明
Initial_Call	BOOL	OB_Class = 1、30、52、61、65
Remanence	BOOL	OB_Class = 1

表格 9-131 SI_TimeOfDay 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 10	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
CaughtUp	BOOL	OB_Class = 10
SecondTime	BOOL	OB_Class = 10

表格 9-132 SI_Delay 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 20	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
Sign	WORD	OB_Class = 20

表格 9-133 SI_Cyclic 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 30	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
Initial_Call	BOOL	OB_Class = 1、30、52、61、65
Event_Count	INT	OB_Class = 30、51、52、61、65、91、92

表格 9-134 SI_HWInterrupt 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 40	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
LADDR	HW_IO	OB_Class = 40、51、55、56、57、70、82、83、85、86、91、92
USI	WORD	OB_Class = 40
IChannel	USINT	OB_Class = 40
EventType	BYTE	OB_Class = 40

表格 9-135 SI_Submodule 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
LADDR	HW_IO	OB_Class = 40、51、55、56、57、70、82、83、85、86、91、92
Slot	UINT	OB_Class = 55、56、57
Specifier	WORD	OB_Class = 55、56、57

表格 9-136 SI_SynchCycle 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 61	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
Initial_Call	BOOL	OB_Class = 1、30、52、61、65
PIP_Input	BOOL	OB_Class = 61、91、92
PIP_Output	BOOL	OB_Class = 61、91、92
IO_System	USINT	OB_Class = 61、91、92

结构元素	数据类型	说明
Event_Count	INT	OB_Class = 30、51、52、61、65、91、92
SyncCycleTime	LTIME	计算得到的循环时间

表格 9-137 SI_IORedundancyError 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 70	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
LADDR	HW_ANY	OB_Class = 40、51、55、56、57、70、82、83、85、86、91、92
Event_Class	BYTE	OB_Class = 70、83、85、86
Fault_ID	BYTE	OB_Class = 70、80、83、85、86

表格 9-138 SI_CPURedundancyError 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 72	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
Switch_Over	BOOL	OB_Class = 72

表格 9-139 SI_TimeError 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 80	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
Fault_ID	BYTE	OB_Class = 70、80、83、85、86
Csg_OBnr	OB_ANY	OB_Class = 80
Csg_Prio	UINT	OB_Class = 80

表格 9-140 SI_DiagnosticInterrupt 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 82	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
IO_State	WORD	OB_Class = 82
LADDR	HW_ANY	OB_Class = 40、51、55、56、57、70、82、83、85、86、91、92
Channel	UINT	OB_Class = 82
MultiError	BOOL	OB_Class = 82

表格 9-141 SI_PlugPullModule 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 83	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
LADDR	HW_IO	OB_Class = 40、51、55、56、57、70、82、83、85、86、91、92
Event_Class	BYTE	OB_Class = 70、83、85、86
Fault_ID	BYTE	OB_Class = 70、80、83、85、86

表格 9-142 SI_AccessError 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 85	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
LADDR	HW_IO	OB_Class = 40、51、55、56、57、70、82、83、85、86、91、92
Event_Class	BYTE	OB_Class = 70、83、85、86
Fault_ID	BYTE	OB_Class = 70、80、83、85、86
IO_Addr	UINT	OB_Class = 85
IO_LEN	UINT	OB_Class = 85

表格 9-143 SI_StationFailure 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 86	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
LADDR	HW_IO	OB_Class = 40、51、55、56、57、70、82、83、85、86、91、92
Event_Class	BYTE	OB_Class = 70、83、85、86
Fault_ID	BYTE	OB_Class = 70、80、83、85、86

表格 9-144 SI_Servo 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 91	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
Initial_Call	BOOL	OB_Class = 1、30、52、61、65
PIP_Input	BOOL	OB_Class = 61、91、92
PIP_Output	BOOL	OB_Class = 61、91、92
IO_System	USINT	OB_Class = 61、91、92
Event_Count	INT	OB_Class = 30、51、52、61、65、91、92
Synchronous	BOOL	

表格 9-145 SI_Ipo 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 92	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
Initial_Call	BOOL	OB_Class = 1、30、52、61、65
PIP_Input	BOOL	OB_Class = 61、91、92
PIP_Output	BOOL	OB_Class = 61、91、92

结构元素	数据类型	说明
IO_System	USINT	OB_Class = 61、91、92
Event_Count	INT	OB_Class = 30、51、52、61、65、91、92
Reduction	UINT	OB_Class = 92

表格 9-146 SI_Startup 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT := 100	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
LostRetentive	BOOL	OB_Class = 100
LostRTC	BOOL	OB_Class = 100

表格 9-147 SI_ProgIOAcessError 结构

结构元素	数据类型	说明
SI_Format	USINT	• 16#FF = 无信息 • 16#FE = 优化启动信息
OB_Class	USINT	“无信息”或“优化启动信息”的 OB 类别
OB_Nr	UINT	OB 编号（1 到 32767）
BlockNr	UINT	OB_Class = 121、122
Reaction	USINT	OB_Class = 121、122
Fault_ID	BYTE	OB_Class = 121、122
BlockType	USINT	OB_Class = 121、122
Area	USINT	OB_Class = 121、122
DBNr	DB_ANY	OB_Class = 121、122
Csg_OBNr	OB_ANY	OB_Class = 121、122
Csg_Prio	USINT	OB_Class = 121、122
Width	USINT	OB_Class = 121、122

说明

如果创建的块属性为“Standard”，则 SI_classic 结构中指定的结构元素内容将与 OB 临时变量的内容相同。

但请注意，各 OB 的临时变量可具有不同名称和数据类型。另请注意，每个 OB 的调用接口都包含有关 OB 请求的日期与时间的附加信息。

结构元素 EV_CLASS 的位 4 至 7 包括事件类别。可以是下面的值：

- 1：来自标准 OB 的启动事件
- 2：来自同步错误 OB 的启动事件
- 3：来自异步错误 OB 的启动事件

结构元素 PRIORITY 提供属于当前 OB 的优先级。

除这两个元素之外，NUM 也很重要。NUM 包含当前 OB 或最后启动的启动 OB 的编号。

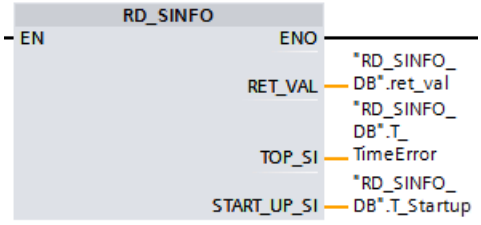
RET_VAL 参数

下表列出了 RET_VAL 参数值的含义：

错误代码* (W#16#...)	说明
8081	当前 OB 的启动信息与指定的系统数据类型不匹配。
8083	上一次启动中启动 OB 的启动信息与指定的系统数据类型不匹配。
* 在程序编辑器中，错误代码可显示为整数或十六进制值。	

示例

时间错误中断 OB (OB 80) 为最后调用且未完全处理的 OB。启动 OB (OB 100) 为最后开始的启动 OB。指令调用读取如下启动信息。RD_SINFO_DB 是包含 OB 类型的 SDT 变量的数据块：



下表说明了指令“RD_SINFO”的参数 TOP_SI 的结构元素与 OB 80 的相关本地变量之间的分配关系。

TOP_SI 结构元素	数据类型	OB 80 - 相关本地变量	数据类型
EV_CLASS	BYTE	OB80_EV_CLASS	BYTE
EV_NUM	BYTE	OB80_FLT_ID	BYTE
PRIORITY	BYTE	OB80_PRIORITY	BYTE
NUM	BYTE	OB80_OB_NUMBR	BYTE
TYP2_3	BYTE	OB80_RESERVED_1	BYTE
TYP1	BYTE	OB80_RESERVED_2	BYTE
ZI1	WORD	OB80_ERROR_INFO	WORD
ZI2_3	DWORD	OB80_ERR_EV_CLASS	BYTE
		OB80_ERR_EV_NUM	BYTE
		OB80_OB_PRIORITY	BYTE
		OB80_OB_NUM	BYTE

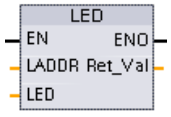
下表说明了指令“RD_SINFO”的参数 START_UP_SI 的结构元素与 OB 100 的相关本地变量之间的分配关系。

START_UP_SI 结构元素	数据类型	OB 100 - 本地变量	数据类型
EV_CLASS	BYTE	OB100_EV_CLASS	BYTE
EV_NUM	BYTE	OB100_STRTUP	BYTE
PRIORITY	BYTE	OB100_PRIORITY	BYTE
NUM	BYTE	OB100_OB_NUMBR	BYTE
TYP2_3	BYTE	OB100_RESERVED_1	BYTE
TYP1	BYTE	OB100_RESERVED_2	BYTE
ZI1	WORD	OB100_STOP	WORD
ZI2_3	DWORD	OB100_STRT_INFO	DWORD

9.7.3 LED（读取 LED 状态）

表格 9-148 LED 指令

LAD/FBD	SCL	说明

	`ret_val := LED(`
	`laddr:=_word_in_,`
	`LED:=_uint_in_);`
		使用 LED 指令可读取 CPU 上 LED 的状态 (页 1383)。通过 RET_VAL 输出返回指定 LED 的状态。

表格 9-149 参数的数据类型

参数和类型		数据类型	说明		
LADDR	IN	HW_IO	CPU 的标识符 [1]		
LED	IN	UInt	LED 标识号		
			1	RUN/STOP	颜色 1 = 绿色，颜色 2 = 黄色
			2	出错	颜色 1 = 红色
			3	维护	颜色 1 = 黄色
RET_VAL	OUT	Int	LED 的状态		

[1] 对于连接的 CPU 的标识符，请从参数下拉列表中选择 Local~Common。

表格 9-150 RET_VAL 的状态

RET_VAL (W#16#...)	说明	
0 到 9 LED 状态	0	LED 不存在
	1	灭
	2	颜色 1 常亮
	3	颜色 2 常亮
	4	颜色 1 以 2 Hz 的频率闪烁
	5	颜色 2 以 2 Hz 的频率闪烁
	6	颜色 1 和 2 以 2 Hz 的频率交替闪烁
	9	LED 状态不可用
8091	由 LADDR 标识的设备不存在	
8092	由 LADDR 标识的设备不支持 LED	

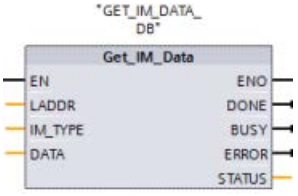
RET_VAL (W#16#...)	说明
8093	LED 标识符未定义
80Bx	由 LADDR 标识的 CPU 不支持 LED 指令

9.7.4 Get_IM_Data（读取标识和维护数据）

可使用 Get_IM_Data 指令检查指定模块或子模块的标识和维护 (I&M) 数据。

表格 9-151 Get_IM_Data 指令

LAD/FBD	SCL	说明

	`"GET_IM_DATA_DB"(LADDR:=16#0,`
	`IM_TYPE:=0,`
	`DONE=>_bool_out_,`
	`BUSY=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`DATA:=_variant_inout_);`
		可使用 Get_IM_Data 指令检查指定模块或子模块的标识和维护 (I&M) 数据。

表格 9-152 参数的数据类型

参数和类型		数据类型	说明
LADDR	Input	HW_IO	模块标识符
IM_TYPE	Input	UInt	标识和维护 (I&M) 数据编号： • 0：I&M0（MLFB、序列号、版本及其它信息） • 1：I&M1（标识） • 2：I&M2（安装日期） • 3：I&M3（描述符） • 4：I&M4（签名）
RET_VAL	Output	Int	状态（条件代码）
DATA	InOut	Variant	I&M 数据（STRING 或 BYTE 数组）；建议在 IM_TYPE = 0 时，使用 SDT“IM0_Data”。

标识和维护 (I&M) 数据有助于您检查系统组态、检测硬件变更或查看维护数据。模块标识数据（I 数据）为只读数据。模块维护数据（M 数据）取决于安装日期等系统信息。在维护规划期间创建 M 数据并将其写入模块中：

- 如果参数 DATA 所使用的数据类型为字符串，则根据 I&M 数据的长度设置字符串的当前长度。
- 如果参数 DATA 所使用的数据类型为 Byte 或 Char 数组，则按照字节顺序复制 I&M 数据。
- 如果参数 DATA 所使用的数据类型是一个结构，则按照字节顺序复制 I&M 数据。
- 如果 DATA 中给定的字节/字符数组比请求的 I&M 数据长，则附加字节值 16#00。
- 不支持其它数据类型，否则将返回错误 8093。

表格 9-153　条件代码

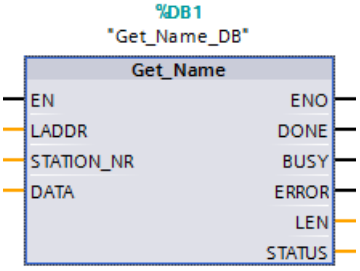
RET_VAL (W#16#...)	说明
0	无错误
8091	LADDR 不存在
8092	LADDR 未寻址到支持 I&M 数据的 HW 对象
8093	不支持参数 DATA 给定的数据类型
80B1	CPU 不支持在此 LADDR 中使用 DATA 指令
80B2	CPU 不支持 IM_TYPE
8452	完整的 I&M 信息不适合 DATA 参数给定的变量。最长可以返回一个与变量字节长度相等的部分结果。

9.7.5 Get_Name（读取 PROFINET IO 设备的名称）

“Get_Name”指令读取 PROFINET IO 设备、PROFIBUS 从站或 AS-i slave 的名称。设备名称将显示在网络视图和 IO 设备的属性中。

表格 9-154 Get_Name 指令

LAD/FBD	SCL	说明

	`"Get_Name_DB"(`
	`LADDR:=_uint_in_,`
	`STATION_NR:=_uint_in_,`
	`DONE=>_bool_out_,`
	`BUSY=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`LEN=>_dint_out_,`
	`STATUS=>_word_out_,`
	`DATA:=_variant_inout_);`
		使用 Get_Name 指令读取 PROFINET IO 设备或 PROFIBUS 从站的名称。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“Get_Name_DB”是背景 DB 的名称。

通过使用分布式 IO 系统的硬件标识符（LADDR 参数处）和 PROFINET IO 设备的设备编号或 PROFIBUS 从站的 PROFIBUS 地址（STATION_NR 参数）可选择 IO 设备。

执行该指令后，程序将在由 DATA 参数寻址的区域中写入 IO 设备名称。

所读取的名称取决于 IO 设备的类型：

- DP 从站或 IO 设备：前端模块名称
- 智能从站或智能设备：接口模块名称
- HMI 面板：接口名称
- PC 站：接口模块名称
- GSD 设备：显示设备接入点 (DAP) 的名称（接口或前端模块的名称）

指令通过 LEN 参数写入该名称的长度。如果该名称的长度大于 DATA 参数指定的区域，则程序将只写入寻址区域的最大长度的部分名称。

该名称的最大长度为 128 个字符。

说明

读取的 CPU 的名称 (V 1.1)

如果参数 LADDR 和 STATION_NR, 的值均为“0”，则该指令将写入 CPU 的名称。

参数

下表列出了 Get_Name 指令的参数：

参数	声明	数据类型	说明
LADDR	IN	HW_IOSYSTEM	分布式 I/O 系统的硬件标识符 (HW-IoSystem)。该值为系统常量或 IO 系统属性。
STATION_NR	IN	UInt	• PROFINET IO 设备：该设备编号将显示在“以太网地址”(Ethernet addresses) 下方 IO 设备属性内的网络视图 (Network view) 中。 • PROFIBUS 从站：该 PROFIBUS 地址将显示在“PROFIBUS 地址”(PROFIBUS address) 下方 PROFIBUS 从站属性内的网络视图 (Network view) 中。
DATA	IN_OUT	Variant	指向名称要写入的目标区域的指针。
DONE	OUT	Bool	指令执行成功。传送到 DATA 参数处指定区域的模块名称。
BUSY	OUT	Bool	状态参数： • 0：指令执行完成。 • 1：指令的执行尚未完成。
ERROR	OUT	Bool	状态参数： • 0：无错误 • 1：指令执行期间出现错误。 参数 STATUS 详细信息。
LEN	OUT	DInt	IO 设备名称的长度（字符数）。
STATUS	OUT	Word	状态参数：该参数设置仅维持一次调用所持续的时间。为了显示该状态，需将 STATUS 复制至一个空闲数据区。

STATUS 参数

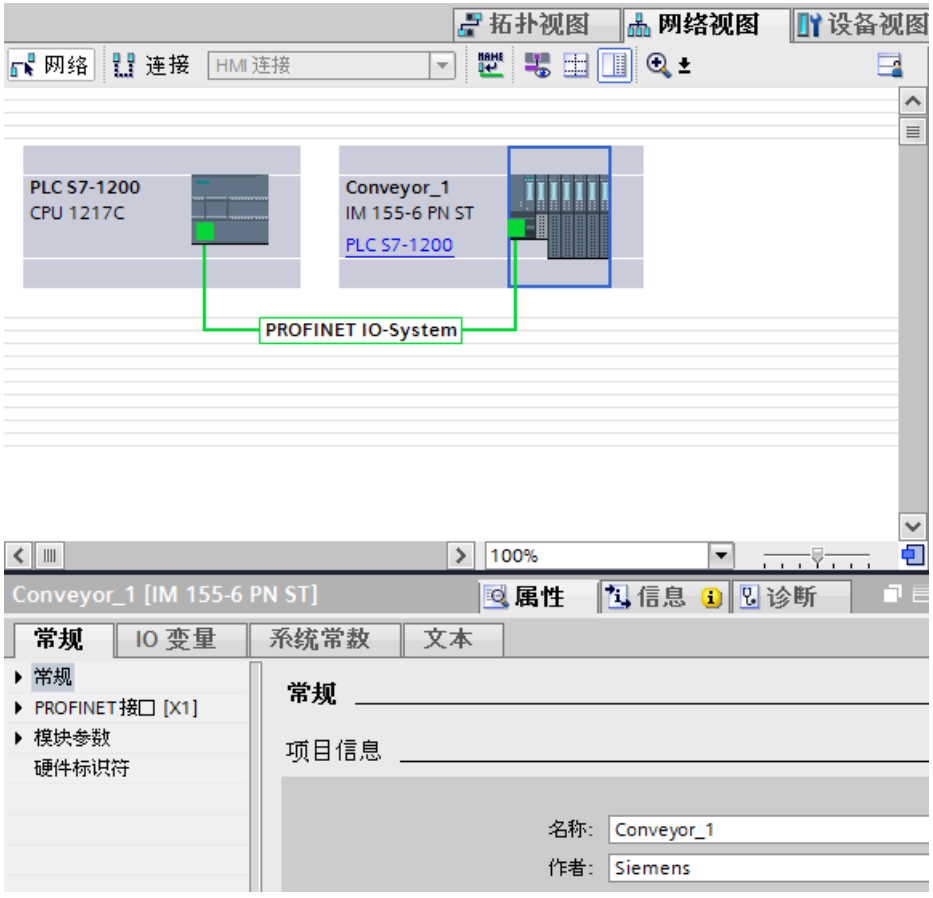
错误代码* (W#16#...)	说明
0	无错误
7000	没有作业正在处理
7001	第一次调用 Get_Name 异步指令。指令的执行尚未完成 (BUSY = 1，DONE = 0)。
7002	另一次调用 Get_Name 异步指令。指令的执行尚未完成 (BUSY = 1，DONE = 0)。

错误代码* (W#16#...)	说明
8090	在 LADDR 参数中指定的硬件标识符在项目中不存在。
8092	LADDR 参数的值无法寻址 PROFINET IO 系统。
8093	指令不支持 DATA 参数中的数据类型。
8095	所选的 PROFINET IO 系统中没有该设备编号（STATION_NR 参数），或者找不到 IO 设备。
80B1	所用 CPU 不支持该指令。
80C3	临时资源错误：CPU 当前正在处理的同步块调用的最大数量。仅当至少一个块调用执行完成后，才能执行 Get_Name。
8852	DATA 参数处指定的区域过短，无法写入 IO 设备的完整名称。只能写入最大允许长度的部分名称。 要读取完整名称，需在 DATA 参数处指定一个较大的数据区域。此区域的大小必须至少与 LEN 参数处指定的字符数相一致。
* 在程序编辑器中，错误代码可显示为整数或十六进制值。	

示例

以下举例说明了如何读取 ET 200SP PROFINET IO 设备的站名称。

1. 组态 ET 200SP：
   - 在网络视图中创建站名称为“Conveyor_1”的 ET 200SP，并将其分配给相同 CPU 的 PROFINET IO 系统。
   - 将 CPU 作为 ET 200SP IO 控制器分配。
   - 在位于 “以太网地址”(Ethernet addresses) 的默认属性中，使用编号为“1”的默认设备。



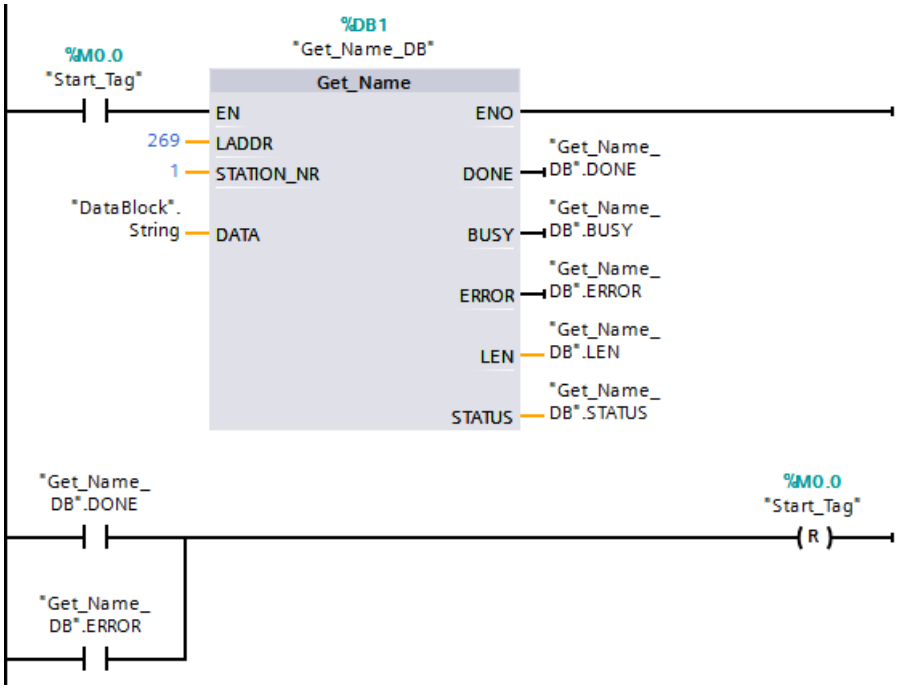
2. 分配 Get_Name 指令的参数：
   - 在参数 LADDR 中输入 IO 系统的硬件 ID。在本例中，硬件 ID 为“269”。您可以在如下位置找到硬件 ID：
     “PLC 变量 > 显示所有变量 > 系统常量选项卡 > 本地 PROFINET_IO 系统”(PLC tags > Show all tags > System constants tab > Local-PROFINET_IO-System)
   - 在 STATION_NR 参数上输入 ET 200SP 的设备编号。在本例中，设备编号为“1”。

– 在参数 DATA 处，将变量与数据类型为 STRING 的数据块相关联。

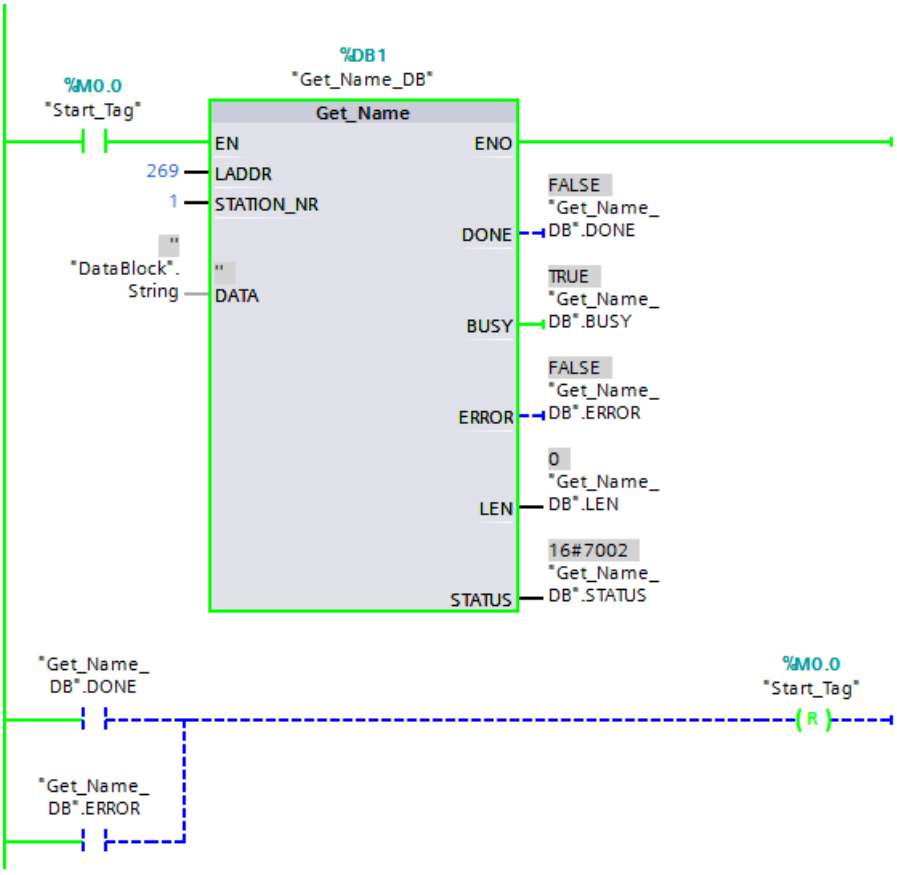
说明

当使用下拉列表选择组态变量 DATA 参数，选择 DB（在本例中，“Datablock”）以及变量（在本例中，“String[ ]”）。要读取整个 “字符串”(String) 数据类型，必须删除括号，因此最终结果为：“Datablock”.String

– 并为该指令的输出参数定义 PLC 变量（存储区、标记）。

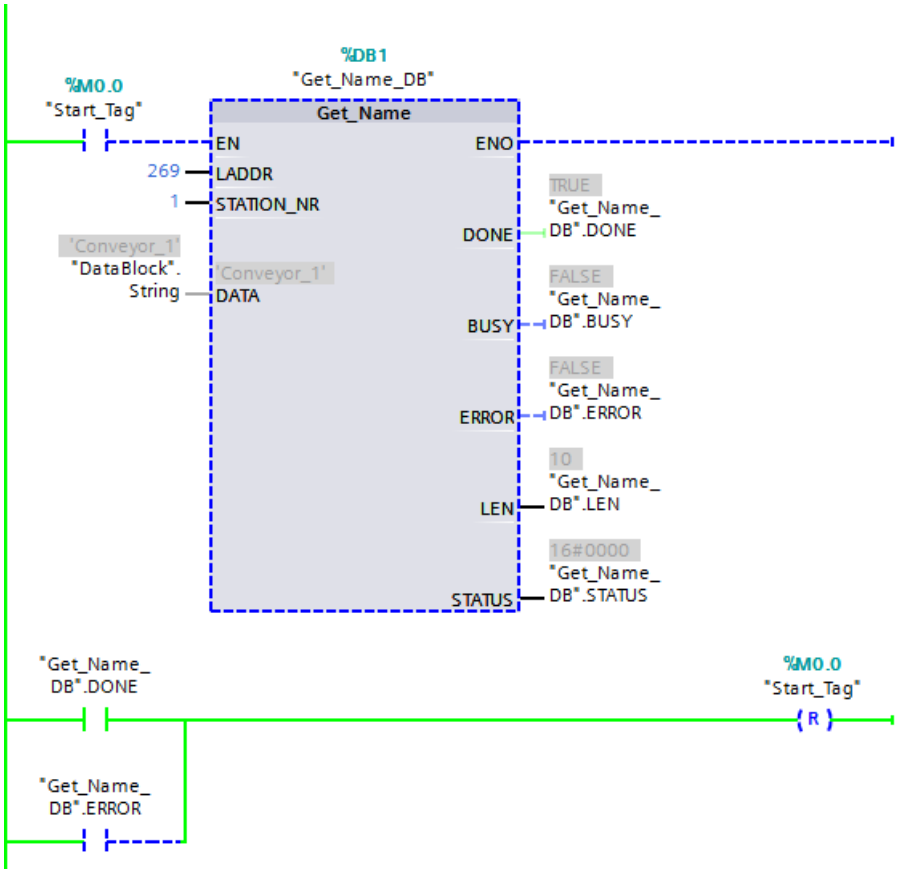


3. 正在执行 Get_Name 指令：
   – 随着指令执行，BUSY 输出参数可能会设置为“1”，然后 DONE 参数设置为“0”。
   – 并在输出参数 STATUS 处，显示错误代码信息。



4. 完成 Get_Name 指令的执行：
   – 执行该指令之后，程序将 ET 200SP 的站名称“Conveyor_1”写入参数 DATA 中的数据块内。

– 程序将站名称中的字符数“10”写入 LEN 参数。



9.7.6 GetStationInfo（读取 PROFINET IO 设备的 IP 或 MAC 地址）

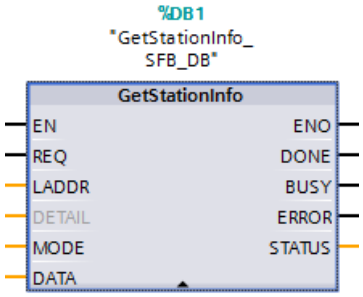
“GetStationInfo”指令读取位于本地 IO 系统中 PROFINET IO 设备或下级 IO 系统中 PROFINET IO 设备的 IP 或 MAC 地址（使用 CP/CM 模块连接）。

说明

只能使用 PROFINET IO 设备的 GetStationInfo 指令。不能使用 PROFIBUS DP 从站的指令。

表格 9-155 GetStationInfo 指令

LAD/FBD	SCL	说明

	`"GetStationInfo_SFB_DB"(`
	`REQ:=_bool_in_,`
	`LADDR:=_uint_in_,`
	`DETAIL:=_uint_in_,`
	`MODE:=_uint_in_,`
	`DONE=>_bool_out_,`
	`BUSY=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`DATA:=_variant_inout_);`
		使用 GetStationInfo 指令读取 PROFINET IO 设备的 IP 或 MAC 地址。通过该指令，还可以读取下级 IO 系统中 IO 设备的 IP 或 MAC 地址（使用 CP/CM 模块连接）。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“GetStationInfo_SFB_DB”是背景 DB 的名称。

在 LADDR 参数处，使用站的硬件标识符寻址 IO 设备。您可以在如下位置找到硬件 ID：“PLC 变量 > 显示所有变量 > 系统常量选项卡”(PLC tags > Show all tags > System constants tab)。在 “名称”(Name) 列中搜索“IODevice”，然后在 “数据类型”(Data type) 列中搜索“Hw_Device”。

通过 MODE 参数，选择要读取的信息。

在 DATA 参数处，可分配指令写入的所读取地址数据的数据区。使用“IF_CONF_v4”结构存储 IP 地址。使用“IF_CONF_MAC”结构存储 MAC 地址。

使用 REQ 控制参数启用地址数据读取。这需要能够访问 IO 设备。

指令通过 BUSY、DONE、 ERROR 输出参数和 STATUS 输出参数显示了读取作业的执行状态。

说明

仅使用站的硬件标识符寻址 IO 设备

站、IO 设备和 PROFINET 接口都具有各自的硬件标识符。对于 GetStationInfo 指令，仅使用站的硬件标识符。

例如，如果通过参数 LADDR 寻址 PROFINET 接口，则不读取该地址数据，且 CPU 将会生成 "8092" 错误代码。

要读取集成 PROFINET 接口或集中组态中 CM/CP 模块的地址数据，使用“RDREC”指令。

参数

下表列出了 GetStationInfo 指令的参数：

参数	声明	数据类型	说明
REQ	IN	Bool	控制参数请求 使用 REQ = “1”启动信息读取操作。
LADDR	IN	HW_DEVICE	IO 设备站的硬件标识符 该编号源自网络视图中的站属性，或源自默认变量表的“系统常量”(System constants) 选项卡。
DETAIL	IN	HW_SUBMODULE	DETAIL 参数未使用。保持不连接参数。
MODE	IN	UNIT	选择要读取的地址数据： • MODE = 1：地址参数符合 IPv4（S7-1200 CPU，固件版本 V4.2 及以上版本） • MODE = 2：MAC 地址（S7-1200 CPU，固件版本：V4.2）
DATA	IN_OUT	Variant	指向程序写入 IO 设备地址区域的区域指针。MODE = 1 时使用“IF_CONF_v4”结构，MODE = 2 时使用“IF_CONF_MAC”结构。
DONE	OUT	Bool	程序成功执行指令。程序将地址数据传送到 DATA 参数。
BUSY	OUT	Bool	STATUS 参数： • 0：指令执行完成。 • 1：指令的执行尚未完成。
ERROR	OUT	Bool	STATUS 参数： • 0：无错误。 • 1：指令执行期间出现错误。 详细信息将在 STATUS 参数中输出。
STATUS	OUT	Word	STATUS 参数： 该参数设置仅维持一次调用所持续的时间。为了显示该状态，需将 STATUS 复制至一个空闲数据区。

DATA 参数

- 在 DATA 参数处使用“IF_CONF_v4”结构来根据 IPv4 保存地址参数：

字节	参数	数据类型	起始值	说明
0 … 1	Id	UINT	30	“IF_CONF_v4”结构的 ID
2 … 3	Length	UNIT	18	在 BYTE 中读取的数据的长度。
4 ... 5	Mode	UNIT	0	与“GetStationInfo”指令无关（保留为“0”）
6 ... 9	InterfaceAddress	ARRAY [1..4] of BYTE	-	IP_V4 格式的 IO 设备的 IP 地址（例如 192.168.3.10）： • addr[1] = 192 • addr[2] = 168 • addr[3] = 3 • addr[4] = 10
10 … 13	SubnetMask	ARRAY [1..4] of BYTE	-	IP_V4 格式的 IO 设备的子网掩码（如 255.255.255.0）： • addr[1] = 255 • addr[2] = 255 • addr[3] = 255 • addr[4] = 0
14 ... 17	DefaultRouter	ARRAY [1..4] of BYTE	-	IP_V4 格式的路由器的 IP 地址（例如 192.168.3.1）： • addr[1] = 192 • addr[2] = 168 • addr[3] = 3 • addr[4] = 1

- 在参数“DATA”中使用“IF_CONF_MAC”结构存储 MAC 地址：

字节	参数	数据类型	起始值	说明
0 … 1	Id	UINT	3	“IF_CONF_MAC”结构的 ID
2 … 3	Length	UNIT	12	在 BYTE 中读取的数据的长度。

字节	参数	数据类型	起始值	说明
4 ... 5	Mode	UNIT	0	与“GetStationInfo”指令无关（保留为“0”）
6 ... 11	MACAddress	ARRAY [1..6] of BYTE	-	IO 设备的 MAC 地址（例如 08-00-06-12-34-56）： • Mac[1] = 8 • Mac[2] = 0 • Mac[3] = 6 • Mac[4] = 12 • Mac[5] = 34 • Mac[6] = 56

STATUS 参数

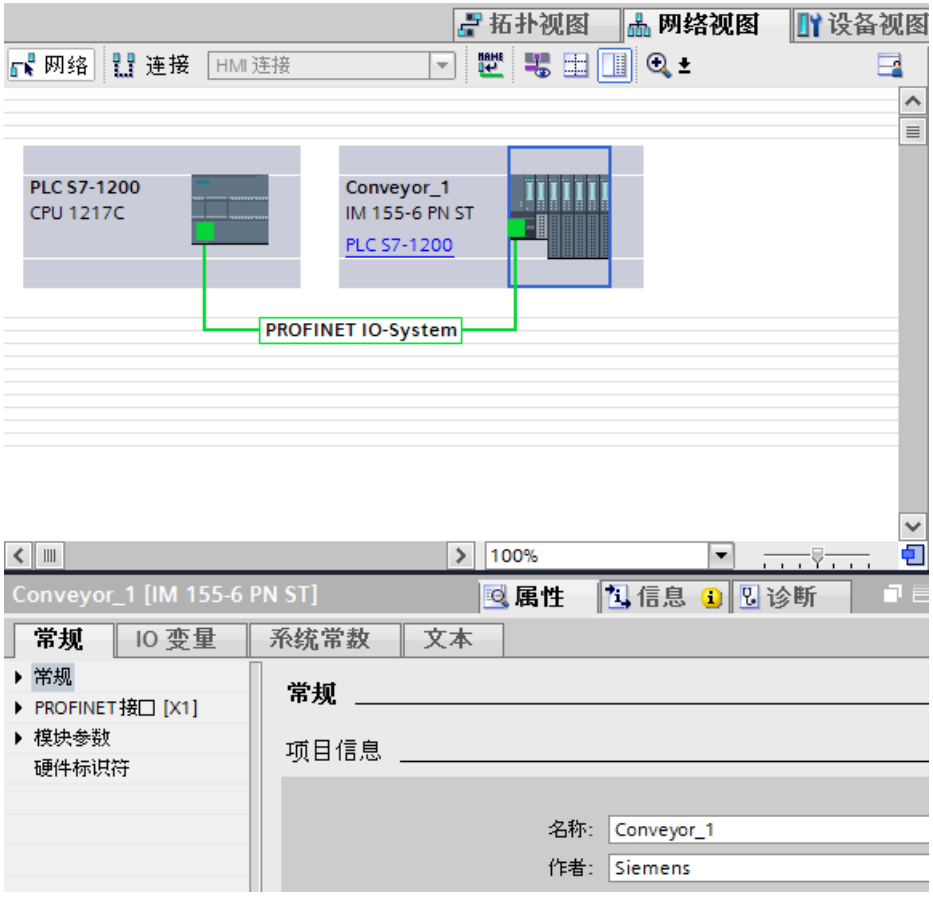
错误代码* (W#16#...)	说明
0	无错误
7000	没有作业正在处理
7001	第一次调用异步指令 GetStationInfo。指令的执行尚未完成 (BUSY = 1，DONE = 0)。
7002	另一次调用异步指令 GetStationInfo。指令的执行尚未完成 (BUSY = 1，DONE = 0)。
8080	不支持 MODE 参数中的值。
8090	未组态 LADDR 参数指定的硬件标识符。
8092	LADDR 参数不会寻址 PROFINET IO 设备。
8093	DATA 参数中的数据类型无效。
80A0	不读取请求的信息。
80C0	不可访问寻址的 IO 设备。
80C3	已达到了 GetStationInfo 指令允许的最大同时调用数（10 个实例）。
* 在程序编辑器中，错误代码将显示为整数或十六进制值。	

示例

在下面的示例中，使用 GetStationInfo 指令读取 IO 设备的 IP 地址数据并将信息写入数据块。IP 地址数据包括 IP 地址、子网掩码和（如果已使用）路由器的地址数据。

IO 控制器执行 GetStationInfo 指令，该指令读取下级 IO 设备的 IP 地址信息（例如，ET200MP）。

1. 组态 ET 200SP：
   – 在网络视图中创建站名称为“Conveyor_1”的 ET 200SP，并将其分配给相同 CPU 的 PROFINET IO 系统。
   – 将 CPU 作为 ET 200SP IO 控制器分配。



2. 分配 GetStationInfo 指令的参数：

– 在全局数据块中创建 5 个变量和 1 个结构（数据类型为 IF_CONF_v4），用于存储该 IP 地址数据。可以为该结构指定任意名称。（在此例中，结构名称为“IP_Address”。）

GetStationInfo_Global_DB

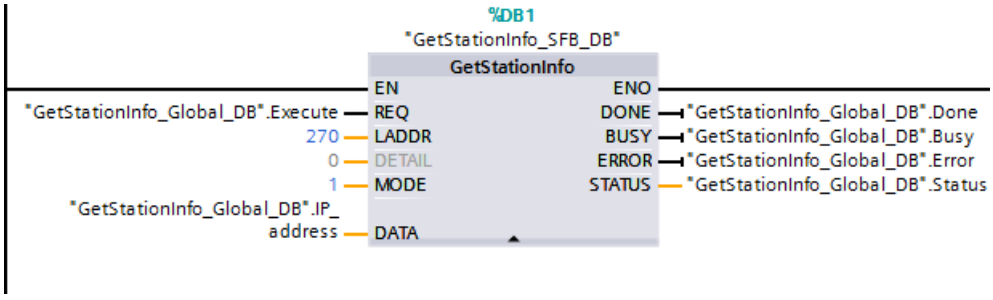
	名称	数据类型	启动值
1	Static		
2	Execute	Bool	false
3	IP_address	IF_CONF_v4	
4	Id	UInt	30
5	Length	UInt	18
6	Mode	UInt	0
7	InterfaceAddress	IP_V4	
8	ADDR	Array[1..4] of Byte	
9	ADDR[1]	Byte	16#0
10	ADDR[2]	Byte	16#0
11	ADDR[3]	Byte	16#0
12	ADDR[4]	Byte	16#0
13	SubnetMask	IP_V4	
14	DefaultRouter	IP_V4	
15	Done	Bool	false
16	Busy	Bool	false
17	Error	Bool	false
18	Status	Word	16#0

3. 分配 GetStationInfo 指令的参数：
   – 在参数 LADDR 中输入 IO 设备的硬件 ID。硬件标识符唯一地标识产品。在本例中，硬件 ID 为“270”。您可以在如下位置找到硬件 ID：“PLC 变量 > 显示所有变量 > 系统常量选项卡”(PLC tags > Show all tags > System constants tab)
   在“名称”(Name) 列中搜索 IO 设备，然后在“数据类型”(Data type) 列中搜索“Hw_Device”。关联值是您在 LADDR 参数中输入的硬件 ID 标识符。
   – 为 MODE 参数选择“1”（根据 IPv4 读取地址参数）。
   – 在 DATA 参数处连接 IF_CONF_v4 结构。

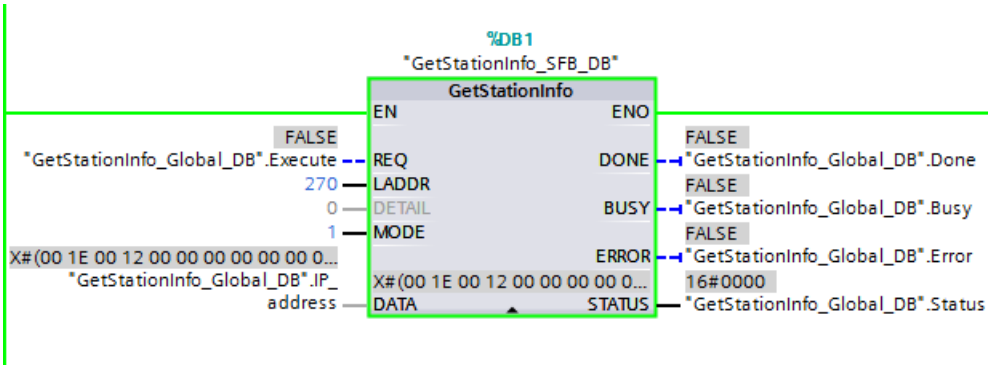
说明

当使用下拉列表选择组态变量 DATA 参数，选择 DB（在本例中，“GetStationInfo_Global_DB”）以及变量（在本例中，“IP address”）。要读取内部 IF_CONF_v4 数据类型，必须删除“IP 地址”后的显示时间，因此最终结果为：“GetStationInfo_Global_DB”.IP 地址

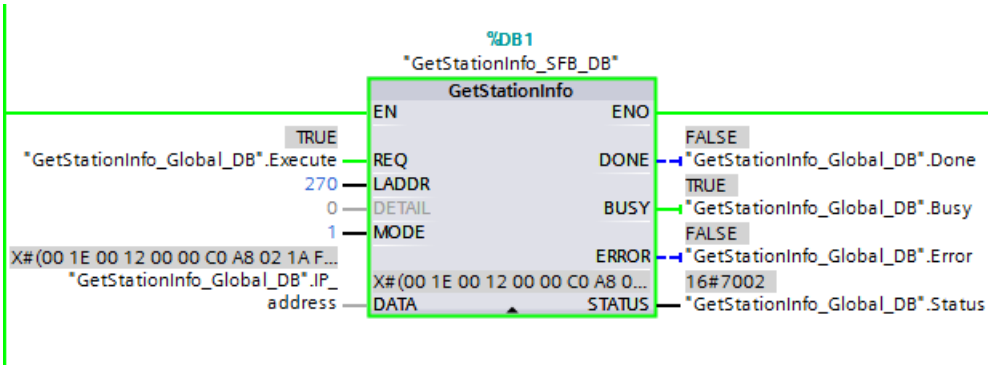
   – 从全局 DB 中为该指令的输出参数定义 PLC 变量（存储区、标记）。



4. 正在执行 GetStationInfo 指令：
   – 当 REQ 输出 = 1（FALSE）时，指令显示在 DATA 输入/输出参数处无 IP 地址信息或在 STATUS 输出参数处无错误代码信息。



5. 完成 GetStationInfo 指令的执行：
   – 当 REQ 输入 = 1 时 (TRUE)，程序执行该指令并将 IP 址写入数据块。程序将 IP 地址 “C0 A8 02 1A”（“192.168.2.26”十进制数的值）写入 DATA 输入/输出参数。



GetStationInfo_Global_DB

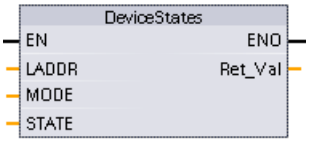
	名称	数据类型	启动值	监视值
1	Static			
2	Execute	Bool	false	TRUE
3	IP_address	IF_CONF_v4		
4	Id	UInt	30	30
5	Length	UInt	18	18
6	Mode	UInt	0	0
7	InterfaceAddress	IP_V4		
8	ADDR	Array[1..4] of Byte		
9	ADDR[1]	Byte	16#0	16#C0
10	ADDR[2]	Byte	16#0	16#A8
11	ADDR[3]	Byte	16#0	16#02
12	ADDR[4]	Byte	16#0	16#1A
13	SubnetMask	IP_V4		
14	DefaultRouter	IP_V4		
15	Done	Bool	false	TRUE
16	Busy	Bool	false	FALSE
17	Error	Bool	false	FALSE
18	Status	Word	16#0	16#0000

9.7.7 DeviceStates 指令

可以使用 DeviceStates 指令返回连接到指定的分布式 I/O 主站的所有分布式 I/O 从站设备的状态。

表格 9-156 DeviceStates 指令

LAD/FBD	SCL	说明

	`ret_val := DeviceStates(`
	`laddr:=_word_in_,`
	`mode:=_uint_in_,`
	`state:=_variant_inout_);`
		DeviceStates 获取 I/O 子系统的 I/O 设备运行状态。指令执行后，STATE 参数将以位列表形式包含各个 I/O 设备的错误状态（针对分配的 LADDR 和 MODE）。此信息与在 STEP 7 诊断视图中看到的设备状态一致。
		DeviceStates 的 LADDR 输入使用分布式 I/O 接口的硬件标识符。在 TIA 门户中，PLC 的硬件标识符可以通过在 PLC 变量表的 “系统常量”(System constants) 选项卡中查找“HW_IOSYSTEM”数据类型找到。

表格 9-157 参数的数据类型

参数和类型		数据类型	说明
LADDR	IN	HW_IOSYSTEM	逻辑地址：（I/O 系统的标识符）
MODE	IN	UInt	支持以下五种工作模式。MODE 输入用于确定返回哪条数据作为指定的 STATE 信息。这些模式如下： • 1：设备组态处于激活状态 • 2：设备故障 • 3：设备已禁用 • 4：设备存在 • 5：设备中存在问题

参数和类型		数据类型	说明
RET_VAL	OUT	Int	执行条件代码
STATE[1]	InOut	Variant	接收每个设备的错误状态的缓冲区：为 STATE 参数选择的数据类型可以是任何位类型（Bool、Byte、Word 或 DWord），也可以是位类型的数组 • 返回的 STATE 数据的第一个字节的 0 位是摘要位。该位设置为 TRUE 时，表示其它数据可用。 • STATE 参数返回的数据展现了位位置与分布式 I/O 地址之间一对一的关系。此设备寻址对于 PROFIBUS 和 PROFINET 为 TRUE。例如，第一个字节的 4 位与 PROFIBUS 地址 4 或 PROFINET 设备号 4 对应。

[1] 对于 PROFIBUS-DP，状态信息的长度为 128 位。对于 PROFINET I/O，长度为 1024 位。

指令执行后，STATE 参数将以位列表形式包含各个 I/O 设备的错误状态（针对分配的 LADDR 和 MODE）。

表格 9-158 条件代码

RET_VAL (W#16#...)	说明
0	无错误
8091	LADDR 不存在。
8092	LADDR 未寻址 I/O 系统。
8093	为 STATE 参数分配的数据类型无效：有效数据类型为（Bool、Byte、Word 或 Dword）或者（Bool、Byte、Word 或 Dword）的数组
80Bx	CPU 不支持在此 LADDR 中使用 DeviceStates 指令。
8452	完整的状态数据对于分配的 STATE 参数来说过大。STATE 缓冲区包含部分结果。

9.7.7.1 DeviceStates 组态示例

PROFIBUS 示例

PROFIBUS 示例的构成如下：

- 16 个 PROFIBUS 设备，名称为“DPSlave_10”至“DPSlave_25”
- 这 16 个 PROFIBUS 设备分别使用 PROFIBUS 地址 10 至 25。
- 每个从站设备都使用多个 I/O 模块组态。
- 显示返回的 STATE 参数信息的前四个字节。

MODE	示例 1： 正常运行没有错误	示例 2： PROFIBUS 从站设备 DPSlave_12 有一个模块拔出	示例 3： PROFIBUS 从站设备 DPSlave_12 断开连接
1: 设备组态处于激活状态	0x01FC_FF03	0x01FC_FF03	0x01FC_FF03
2: 设备故障	0x0000_0000	0x0110_0000	0x0110_0000
3: 设备已禁用	0x0000_0000	0x0000_0000	0x0000_0000
4: 设备存在	0x01FC_FF03	0x01FC_FF03	0x01EC_FF03
5: 设备中存在问题	0x0000_0000	0x0110_0000	0x0110_0000

以下四个表格显示了当前分析的四个字节数据的二进制明细：

表格 9-159 示例 1： 无错误： 对于 MODE 1（设备组态处于激活状态），返回 0x01FC_FF03。

字节和对应的值	位序列和对应的值	注意
字节 1 0x01	位 7 0000-0001 位 0	0 位为真；数据可用。
字节 2 0xFC	位 15 1111-1100 位 8	
字节 3 0xFF	位 23 1111-1111 位 16	
字节 4 0x03	位 31 0000-0011 位 24	

使用地址 10（位 10）至 25（位 25）组态设备。
不使用地址 1 至 9 组态设备。
MODE 4（设备存在）数据与 MODE 1（设备组态处于激活状态）匹配，因此组态的设备与现有设备相匹配。

表格 9-160 示例 2： 已从 PROFIBUS 从站设备“DPSlave_12”拔出一个模块。 对于 MODE 2（设备故障），返回 0x0110_0000。

字节和对应的值	位序列和对应的值	注意
字节 1 0x01	位 7 0000-0001 位 0	0 位为真；数据可用。
字节 2 0x10	位 15 0001-0000 位 8	
字节 3 0x00	位 23 0000-0000 位 16	
字节 4 0x00	位 31 0000-0000 位 24	

设备 12（位 12）已标记为故障。
MODE 5（设备中存在问题）返回的信息与 MODE 2（设备故障）一样。

表格 9-161 示例 2（续）： 已从 PROFIBUS 从站设备“DPSlave_12”拔出一个模块。 对于 MODE 4（设备存在），返回 0x01FC_FF03。

字节和对应的值	位序列和对应的值	注意
字节 1 0x01	位 7 0000-0001 位 0	0 位为真；数据可用。
字节 2 0xFC	位 15 1111-1100 位 8	
字节 3 0xFF	位 23 1111-1111 位 16	
字节 4 0x03	位 31 0000-0011 位 24	

即使设备 12（位 12）存在如以上 MODE 2 中所示的错误，但该设备在网络中仍可以正常运行，导致 MODE 4（设备存在）将该设备显示为“现有设备”。

表格 9-162 示例 3： PROFIBUS 从站设备“DPSlave_12”与 PROFIBUS 网络断开连接（电缆断开或断电）。 “DPSlave_12”仍检测为故障设备以及设备出错。 不同点是，“DPSlave_12”不再检测为存在的设备。 对于 MODE 4（设备存在），返回 0x01EC_FF03。

字节和对应的值	位序列和对应的值	注意
字节 1 0x01	位 7 0000-0001 位 0	0 位为真；数据可用。
字节 2 0xEC	位 15 1110-1100 位 8	
字节 3 0xFF	位 23 1111-1111 位 16	
字节 4 0x03	位 31 0000-0011 位 24	

设备 12（位 12）已标记为不存在。 除此之外，设备 10 至 25 仍报告为存在。

PROFINET 示例

PROFINET 示例的构成如下：

- 16 个 PROFINET 从站设备，名称为“et200s_1”至“et200s_16”
- 这 16 个 PROFINET 设备分别使用 PROFINET 设备号 1 至 16。
- 每个从站设备都使用多个 I/O 模块组态。
- 显示返回的 STATE 参数信息的前四个字节。

MODE	示例 1： 正常运行没有错误	示例 2： PROFINET 从站 et200s_1 模块已拔出	示例 3： PROFINET 从站 et200s_1 已断开连接
1: 设备组态处于激活状态	0xFFFF_0100	0xFFFF_0100	0xFFFF_0100
2 - 设备故障	0x0000_0000	0x0300_0000	0x0300_0000
3 - 设备已禁用	0x0000_0000	0x0000_0000	0x0000_0000
4 - 设备存在	0xFFFF_0100	0xFFFF_0100	0xFDFF_0100
5 - 设备中存在问题	0x0000_0000	0x0300_0000	0x0300_0000

以下四个表格显示了当前分析的四个字节数据的二进制明细：

表格 9-163 示例 1： 无错误： 对于 MODE 1（设备组态处于激活状态），返回 0xFFFF_0100。

字节和对应的值	位序列和对应的值	注意
字节 1 0xFF	位 7 1111-1111 位 0	0 位为真；数据可用。
字节 2 0xFF	位 15 1111-1111 位 8	
字节 3 0x01	位 23 0000-0001 位 16	
字节 4 0x00	位 31 0000-0000 位 24	

使用地址 1（位 1）至 16（位 16）组态设备。
不使用地址 1 至 9 组态设备。
MODE 4（设备存在）数据与 MODE 1（设备组态处于激活状态）匹配，因此组态的设备与现有设备相匹配。

表格 9-164 示例 2： 已从 PROFINET 从站设备“et200s_1”拔出一个模块。 对于 MODE 2（设备故障），返回 0x0300_0000。

字节和对应的值	位序列和对应的值	注意
字节 1 0x03	位 7 0000-0011 位 0	0 位为真；数据可用。
字节 2 0x00	位 15 0000-0000 位 8	
字节 3 0x00	位 23 0000-0000 位 16	
字节 4 0x00	位 31 0000-0000 位 24	

设备 1（位 1）已标记为故障。 因为设备仍然存在，MODE 4（设备存在）显示的数据与正常工作状态的一样。
MODE 5（设备中存在问题）返回的信息与 MODE 2（设备故障）一样。

表格 9-165 示例 2（续）： 已从 PROFIBUS 从站设备“et200s_1”拔出一个模块。 对于 MODE 4（设备存在），返回 0xFFFF_0100。

字节和对应的值	位序列和对应的值	注意
字节 1 0xFF	位 7 1111-1111 位 0	0 位为真；数据可用。
字节 2 0xFF	位 15 1111-1111 位 8	
字节 3 0x01	位 23 0000-0001 位 16	
字节 4 0x00	位 31 0000-0000 位 24	

即使设备 1（位 1）存在如以上 MODE 2 中所示的错误，但该设备在网络中仍可以正常运行，导致 MODE 4（设备存在）将该设备显示为“现有设备”。

表格 9-166 示例 3： PROFINET 从站设备“et200s_1”与 PROFINET 网络断开连接（电缆断开或断电）。 对于 MODE 4（设备存在），返回 0xFDFF_0100。

字节和对应的值	位序列和对应的值	注意
字节 1 0xFD	位 7 1111-1101 位 0	0 位为真；数据可用。
字节 2 0xFF	位 15 1111-1111 位 8	
字节 3 0x01	位 23 0000-0001 位 16	
字节 4 0x00	位 31 0000-0000 位 24	

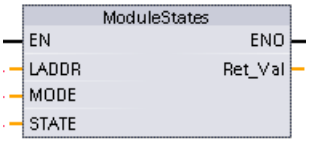
设备 1（位 1）不存在。 设备 2（位 2）至 16（位 16）存在。

9.7.8 ModuleStates 指令

可以使用 ModuleStates 指令返回 PROFIBUS 或 PROFINET 站中所有模块的状态。

表格 9-167 ModuleStates 指令

LAD/FBD	SCL	说明

	`ret_val := ModuleStates(`
	`laddr:=_word_in_,`
	`mode:=_uint_in,`
	`state:=_variant_inout);`
		ModuleStates 获取 I/O 模块的运行状态。指令执行后，STATE 参数将以位列表形式包含各个 I/O 模块的错误状态（针对分配的 LADDR 和 MODE）。此信息与在 STEP 7 诊断视图中看到的模块状态一致。
		ModuleStates 的 LADDR 输入使用的是分布式 I/O 站的硬件标识符而非前端模块本身的硬件标识符。查找硬件标识符的方法为：选择网络视图中的整个站，然后在属性下的硬件标识符部分进行查找。还可以通过在 PLC 变量表的“系统常量”(system constants) 选项卡中查找“Hw_Device”和“Hw_DpSlave”数据类型。

表格 9-168 参数的数据类型

参数和类型		数据类型	说明
LADDR	IN	HW_DEVICE	逻辑地址（I/O 模块的标识符）
MODE	IN	UInt	支持以下五种工作模式。 MODE 输入用于确定返回哪条数据作为指定的 STATE 信息。这些模式如下： • 1：模块组态处于激活状态 • 2：模块有故障 • 3：模块已禁用 • 4：模块存在 • 5：模块中存在问题

参数和类型		数据类型	说明
RET_VAL	OUT	Int	状态（条件代码）
STATE[1]	InOut	Variant	接收每个模块的错误状态的缓冲区：用于 STATE 参数的数据类型可以是任何位类型（Bool、Byte、Word 或 DWord），也可以是位类型的数组。 • 返回的 STATE 数据的第一个字节的 0 位是摘要位。该位设置为 TRUE 时，表示其它数据可用。 • STATE 参数返回的数据展现了位位置与模块位置之间一对一的关系。此插槽寻址对于 PROFIBUS 和 PROFINET 为 TRUE。例如，对于具有前端模块、电源模块和一对 I/O 模块的 ET 200SP，第一个字节的位 1 对应于前端模块，位 2 对应于电源模块，位 3 和 4 分别对应于两个 I/O 模块。

[1] 最多可分配 128 位。所需位数取决于 I/O 模块的使用情况。

表格 9-169 条件代码

RET_VAL (W#16#...)	说明
0	无错误
8091	由 LADDR 标识的模块不存在。
8092	由 LADDR 标识的模块未寻址 I/O 设备。
8093	STATE 参数的数据类型无效：有效数据类型为（Bool、Byte、Word 或 Dword）或者（Bool、Byte、Word 或 Dword）的数组。
80Bx	该 CPU 不支持在此 LADDR 中使用 ModuleStates 指令。
8452	完整的状态数据对于分配的 STATE 参数来说过大。STATE 缓冲区包含部分结果。

9.7.8.1 DeviceStates 组态示例

PROFIBUS 示例

PROFIBUS 示例的构成如下：

- 16 个 PROFIBUS 设备，名称为“DPSlave_10”至“DPSlave_25”
- 这 16 个 PROFIBUS 设备分别使用 PROFIBUS 地址 10 至 25。
- 每个从站设备都使用多个 I/O 模块组态。

- 该示例使用 PROFIBUS 从站“DPSlave_12”的 LADDR 参数，该从站含有一个前端模块、一个电源模块和两个 I/O 模块。
- 显示返回的 STATE 参数信息的前四个字节。

MODE	示例 1： 正常运行没有错误	示例 2： PROFIBUS 从站设备 DPSlave_12 模块已拔出	示例 3： PROFIBUS 从站设备 DPSlave_12 断开连接
1: 模块组态处于激活状态	0x1F00_0000	0x1F00_0000	0x1F00_0000
2: 模块有故障	0x0000_0000	0x0900_0000	0x1F00_0000
3: 模块已禁用	0x0000_0000	0x0000_0000	0x0000_0000
4: 模块存在	0x1F00_0000	0x1700_0000	0x0000_0000
5: 模块中存在问题	0x0000_0000	0x0900_0000	0x1F00_0000

以下四个表格显示了当前分析的四个字节数据的二进制明细：

表格 9-170　示例 1：无错误：对于 MODE 1（模块组态处于激活状态），返回 0x1F00_0000。

字节和对应的值	位序列和对应的值	注意
字节 1 0x1F	位 7 0001-1111 位 0	0 位为真；数据可用。
字节 2 0x00	位 15 0000-0000 位 8	
字节 3 0x00	位 23 0000-0000 位 16	
字节 4 0x00	位 31 0000-0000 位 24	

插槽 1（位 1）至 4（位 4）含有模块。插槽 5（位 5）及以上的插槽都不含模块。
MODE 4（模块存在）数据与 MODE 1（模块组态处于激活状态）匹配，因此组态的模块与现有模块相匹配。

表格 9-171　示例 2：已从 PROFIBUS 从站设备“DPSlave_12”拔出一个模块。对于 MODE 2（模块故障），返回 0x0900_0000。

字节和对应的值	位序列和对应的值	注意
字节 1 0x09	位 7 0000-1001 位 0	0 位为真；数据可用。
字节 2 0x00	位 15 0000-0000 位 8	
字节 3 0x00	位 23 0000-0000 位 16	
字节 4 0x00	位 31 0000-0000 位 24	

只有模块 3（位 3）已标记为故障。所有其它模块都可正常工作。

表格 9-172 示例 2（续）： 已从 PROFIBUS 从站设备“DPSlave_12”拔出一个模块。 对于 MODE 4（模块存在），返回 0x1700_0000。

字节和对应的值	位序列和对应的值	注意
字节 1 0x17	位 7 0001-0111 位 0	0 位为真；数据可用。
字节 2 0x00	位 15 0000-0000 位 8	
字节 3 0x00	位 23 0000-0000 位 16	
字节 4 0x00	位 31 0000-0000 位 24	

模块 3（位 3）显示为不存在。 模块 1、2 和 4（位 1、2 和 4）显示为存在。

表格 9-173 示例 3： PROFIBUS 从站设备“DPSlave_12”与 PROFIBUS 网络断开连接（电缆断开或断电）。 对于 MODE 2（模块故障），返回 0x1F00_0000。

字节和对应的值	位序列和对应的值	注意
字节 1 0x1F	位 7 0001-1111 位 0	0 位为真；数据可用。
字节 2 0x00	位 15 0000-0000 位 8	
字节 3 0x00	位 23 0000-0000 位 16	
字节 4 0x00	位 31 0000-0000 位 24	

因为设备不存在，插槽 1 至 4（位 1 至 4）中的模块全部标记为故障。
MODE 5（模块中存在问题）显示的信息与 MODE 2（模块故障）一样。

PROFINET 示例

PROFINET 示例的构成如下：

- 16 个 PROFINET 从站设备，名称为“et200s_1”至“et200s_16”
- 这 16 个 PROFINET 设备分别使用 PROFINET 设备号 1 至 16。
- 每个从站设备都使用多个 I/O 模块组态。
- 该示例使用 PROFINET 从站“et200s_1”，该从站含有一个前端模块、一个电源模块和 18 个 I/O 模块。
- 显示返回的 STATE 参数信息的前四个字节。

MODE	示例 1： 正常运行没有错误	示例 2： PROFINET et200s_1 从站模块 已拔出	示例 3： PROFINET et200s_1 从站已断 开连接
1: 模块组态处于激活状态	0xFFFF_1F00	0xFFFF_1F00	0xFFFF_1F00
2: 模块有故障	0x0000_0000	0x0180_0000	0xFFFF_1F00
3: 模块已禁用	0x0000_0000	0x0000_0000	0x0000_0000
4: 模块存在	0xFFFF_1F00	0xFF7F_1F00	0x0000_0000
5: 模块中存在问题	0x0000_0000	0x0180_0000	0xFFFF_1F00

以下四个表格显示了当前分析的四个字节数据的二进制明细：

表格 9-174 示例 1： 无错误： 对于 MODE 1（模块组态处于激活状态），返回 0xFFFF_1F00。

字节和对应的值	位序列和对应的值	注意
字节 1 0xFF	位 7 1111-1111 位 0	0 位为真；数据可用。
字节 2 0xFF	位 15 1111-1111 位 8	
字节 3 0x1F	位 23 0001-1111 位 16	
字节 4 0x00	位 31 0000-0000 位 24	

插槽 1（位 1）至 20（位 20）含有模块。 插槽 21（位 21）及以上的插槽都不含模块。
MODE 4（模块存在）数据与 MODE 1（模块组态处于激活状态）匹配，因此组态的模块与现有模块相匹配。

表格 9-175 示例 2： 已从 PROFINET 从站设备“et200s_1”拔出一个模块。 对于 MODE 2（模块故障），返回 0x0180_0000。

字节和对应的值	位序列和对应的值	注意
字节 1 0x01	位 7 0000-0001 位 0	0 位为真；数据可用。
字节 2 0x80	位 15 1000-0000 位 8	
字节 3 0x00	位 23 0000-0000 位 16	
字节 4 0x00	位 31 0000-0000 位 24	

只有模块 15（位 15）已标记为故障。 所有其它模块都可正常工作。

表格 9-176 示例 2（续）： 已从 PROFIBUS 从站设备“et200s_1”拔出一个模块。 对于 MODE 4（模块存在），返回 0xFF7F_1F00。

字节和对应的值	位序列和对应的值	注意
字节 1 0xFF	位 7 1111-1111 位 0	0 位为真；数据可用。
字节 2 0x7F	位 15 0111-1111 位 8	
字节 3 0x1F	位 23 0001-1111 位 16	
字节 4 0x00	位 31 0000-0000 位 24	

模块 15（位 15）显示为不存在。 模块 1 至 14（位 1 至 14）和 16 至 20（位 16 至 20）显示为存在。

表格 9-177 示例 3： PROFINET 从站设备“et200s_1”与 PROFINET 网络断开连接（电缆断开或断电）。 对于 MODE 2（模块故障），返回 0xFFFF_1F00。

字节和对应的值	位序列和对应的值	注意
字节 1 0xFF	位 7 1111-1111 位 0	0 位为真；数据可用。
字节 2 0xFF	位 15 1111-1111 位 8	
字节 3 0x1F	位 23 0001-1111 位 16	
字节 4 0x00	位 31 0000-0000 位 24	

因为设备不存在，插槽 1 至 20（位 1 至 20）中的模块全部标记为故障。
MODE 5（模块中存在问题）显示的信息与 MODE 2（模块故障）一样。

9.7.9 GET_DIAG（读取诊断信息）

说明

可以使用“GET_DIAG”指令读出硬件设备的诊断信息。硬件设备通过 LADDR 参数进行选择。使用 MODE 参数选择要读出的诊断信息。

表格 9-178 GET_DIAG 指令

LAD/FBD	SCL	说明

	`ret_val := GET_DIAG(`
	`mode:=_uint_in_,`
	`laddr:=_word_in_,`
	`cnt_diag=>_uint_out_,`
	`diag:=_variant_inout_,`
	`detail:=_variant_inout_);`
		从分配的硬件设备读取诊断信息。

参数

下表列出了“GET_DIAG”指令的参数：

表格 9-179 参数的数据类型

参数和类型		数据类型	说明
MODE	IN	UInt	使用 MODE 参数选择要输出的诊断数据。
LADDR	IN	HW_ANY (Word)	设备的硬件 ID
RET_VAL	OUT	Int	指令的状态
CNT_DIAG	OUT	UInt	输出诊断详细信息的数量
DIAG	InOut	Variant	指向用于存储所选模式的诊断信息的数据区。
DETAILS	InOut	Variant	指向用于存储与所选模式一致的诊断详细信息的数据区。

MODE 参数

根据 MODE 参数的值，在 DIAG、CNT_DIAG 和 DETAILS 输出参数中输出不同的诊断数据。

表格 9-180 MODE 参数

MODE	说明	DIAG	CNT_DIAG	DETAILS
0	以 DWord 格式输出模块的所有支持诊断信息，其中位 X=1 表示支持模式 X。	所支持模式的位字符串（DWord 格式），其中位 X=1 表示支持模式 X。 当 MODE 参数为 0 时，S7-1200 CPU 会忽略 LADDR 参数。	0	-
1	输出已寻址硬件对象的固有状态。	诊断状态：与 DIS 结构一致的输出。（注意：请参见下面的“DIS 结构”信息以及本部分结尾处的 GET_DIAG 指令示例。）	0	-
2	输出已寻址硬件对象所有从属模块的状态。	输出与 DNN 结构一致的诊断数据。（注意：请参见下面的“DNN 结构”信息以及本部分结尾处的 GET_DIAG 指令示例。）	0	-

DIS 结构

在 MODE 参数 =1 的情况下，诊断信息的输出与 DIS 结构一致。下表列出了各个参数值的含义：

表格 9-181 诊断信息源 (DIS) 的结构

参数	数据类型	值	说明
MaintenanceState	DWord	枚举	
		0	不需要维护
		1	模块或设备已禁用。
		2	-
		3	-
		4	-
		5	需要维护
		6	要求维护
		7	错误
		8	附属模块中的状态未知/错误
		9	-
		10	输入/输出不可用。

参数	数据类型	值	说明
Componentstate Detail	DWord	位数组	模块子模块的状态： ● 位 0 到位 15：模块的状态消息 ● 位 16 到位 31：CPU 的状态消息
		0 到 2（枚举）	附加信息： ● 位 0：无附加信息 ● 位 1：不允许传送
		3	位 3 = 1：至少一个通道支持诊断限定符。
		4	位 4 = 1：至少一个通道或一个组件需要维护
		5	位 5 = 1：至少一个通道或一个组件要求维护
		6	位 6 = 1：至少一个通道或一个组件有错误
		7 到 10	保留（始终为 0）
		11 到 14	位 11 = 1：PNIO - 子模块正确 位 12 = 1：PNIO - 更换模块 位 13 = 1：PNIO - 错误模块 位 14 = 1：PNIO - 模块已断开
		15	保留（始终为 0）
		16 到 31	CPU 生成的模块状态信息： 位 16 = 1：模块已禁用 位 17 = 1：CiR 操作激活 位 18 = 1：输入不可用 位 19 = 1：输出不可用 位 20 = 1：溢出诊断缓冲区 位 21 = 1：诊断不可用 位 22 - 31：保留（始终为 0）

参数	数据类型	值	说明
OwnState	Uint16	枚举	OwnState 参数的值描述了模块的维护状态。
		0	无故障
		1	模块或设备已禁用。
		2	需要维护
		3	要求维护
		4	错误
		5	无法从 CPU 访问模块或设备（对于 CPU 下的模块和设备有效）。
		6	输入/输出不可用。
		7	-
IO State	Uint16	位数组	模块的 I/O 状态
		0	位 0 = 1：不需要维护
		1	位 1 = 1：模块或设备已禁用。
		2	位 2 = 1：需要维护
		3	位 3 = 1：要求维护
		4	位 4 = 1：错误
		5	位 5 = 1：无法从 CPU 访问模块或设备（对于 CPU 下的模块和设备有效）。
		6	限定符；如果位 0、2 或 3 置位，则位 7 = 1
		7	输入/输出不可用。
		8 到 15	保留（始终为 0）

参数	数据类型	值	说明
OperatingState	UInt16	枚举	
		0	-
		1	处于 STOP 状态/固件更新
		2	处于 STOP 状态/复位存储器
		3	处于 STOP 状态/自启动
		4	处于 STOP 状态
		5	存储器复位
		6	处于 START 状态
		7	处于 RUN 状态
		8	-
		9	处于 HOLD 状态
		10	-
		11	-
		12	模块有故障
		13	-
		14	无电源
		15	CiR
		16	处于 STOP 状态/无 DIS
		17	In
		18	
		19	
		20	

DNN 结构

在 MODE 参数=2 的情况下，诊断详细信息的输出与 DNN 结构一致。下表列出了各个参数值的含义：

表格 9-182 诊断导航节点 (DNN) 的结构

参数	数据类型	值	说明
SubordinateState	UINT	Enum	从属模块的状态（请参见 DIS 结构的参数 OwnState）
SubordinateIOState	WORD	Bitarray	从属模块的输入和输出状态（请参见 DIS 结构的参数 IO State）
DNNmode	WORD	Bitarray	• 位 0 = 0：诊断已启用 • 位 0 = 1：诊断已禁用 • 位 1 到位 15：保留

RET_VAL 参数

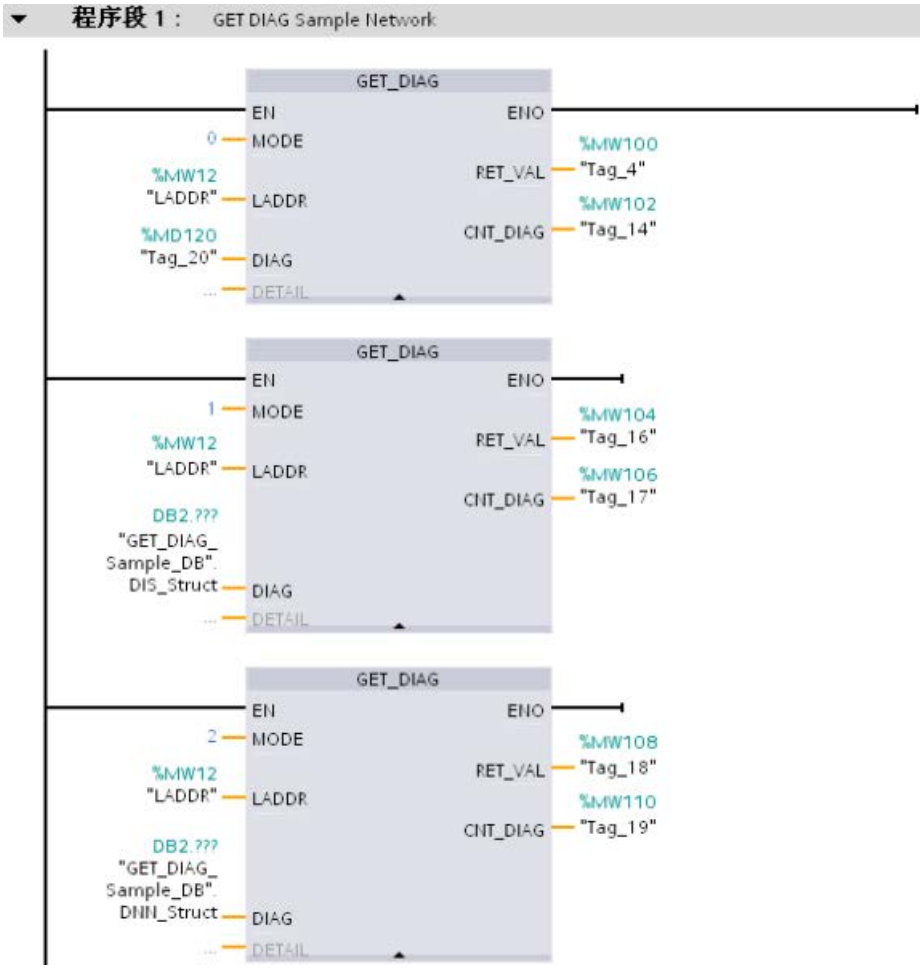
表格 9-183 RET_VAL 参数的错误代码

错误代码 (W#16#...)	说明
0	无错误
8080	不支持 MODE 参数中的值。
8081	所选模式（参数 MODE）不支持 DIAG 参数中的类型。
8082	所选模式（参数 MODE）不支持 DETAILS 参数中的类型。
8090	LADDR 不存在。
8091	CHANNEL 参数中的所选通道不存在。
80C1	并行执行的资源不足

示例

下面的梯形逻辑程序段和 DB 显示了如何使用三种结构的三种模式：

- DIS
- DNN



① DNN

② DIS

说明

在 DB 中，必须手动键入数据类型，以访问三种结构中的每一种；没有下拉列表选择。如下所示，准确键入数据类型：

- DNN
- DIS

9.7.10 分布式 I/O 的诊断事件

说明

对于 PROFIBUS IO 系统，除非硬件兼容性设置为允许可接受替换模块 (页 163)，并且有一个或多个模块丢失或者不是已组态模块的可接受替换模块，否则 CPU 在下载或循环上电后将转到 RUN 模式。

如下表所示，CPU 支持对分布式 I/O 系统中的组件组态的诊断。 只要发生下面提到的错误，诊断缓冲区就会生成一个日志条目。

表格 9-184 PROFINET 和 PROFIBUS 诊断事件的处理

错误类型	成为站的诊断信息？	在诊断缓冲区中生成条目？	CPU 的操作模式
诊断错误	是	是	保持 RUN 模式
机架或站故障	是	是	保持 RUN 模式
I/O 访问错误 [1]	否	是	保持 RUN 模式

错误类型	成为站的诊断信息？	在诊断缓冲区中生成条目？	CPU 的操作模式
外围设备访问错误 [2]	否	是	保持 RUN 模式
插/拔事件	是	是	保持 RUN 模式

[1] I/O 访问错误示例原因： 已移除的一个模块。

[2] 外围设备访问错误示例原因： 非周期性地与没有通信的子模块进行通信。

可对每个站使用 GET_DIAG 指令 (页 480)来获取相应的诊断信息。 用户借此可通过编程来处理设备错误，并根据需要将 CPU 切换为 STOP 模式。 采用此方法时，您需要指定从哪个硬件设备读取状态信息。

GET_DIAG 指令使用站的“L 地址”(LADDR) 来获取整个站点的健康状况。 此 L 地址可在 “网络组态”(Network Configuration) 视图中找到，或者也可选择整个站机架（整个灰色区域）并在站的 “属性”(Properties) 选项卡中获取。 对于各模块的 LADDR，既可在该模块的属性中查看（在设备组态中），也可在 CPU 的默认变量表中查看。

9.8 脉冲

9.8.1 CTRL_PWM（脉宽调制）

表格 9-185 CTRL_PWM（脉宽调制）指令

LAD/FBD	SCL	说明

	`"CTRL_PWM_DB"(`
	`PWM:=_uint_in_,`
	`ENABLE:=_bool_in_,`
	`BUSY=>_bool_out_,`
	`STATUS=>_word_out_);`
		提供占空比可变的固定循环时间输出。PWM 输出以指定频率（循环时间）启动之后将连续运行。脉冲宽度会根据需要进行变化以影响所需的控制。

[1] 插入该指令后，STEP 7 显示用于创建相关数据块的 “调用选项” (Call Options) 对话框。

[2] 在 SCL 示例中，“CTRL_PWM_DB”是背景 DB 的名称。

表格 9-186 参数的数据类型

参数和类型		数据类型	说明
PWM	IN	HW_PWM (Word)	PWM 标识符：已启用的脉冲发生器的名称将变为“常量” (constant) 变量表中的变量，并可用作 PWM 参数。（默认值：0）
ENABLE	IN	Bool	1 = 启动脉冲发生器 0 = 停止脉冲发生器
BUSY	OUT	Bool	功能忙（默认值：0）
STATUS	OUT	Word	执行条件代码（默认值：0）

CTRL_PWM 指令将参数信息存储在 DB 中。数据块参数不是由用户单独更改的，而是由 CTRL_PWM 指令进行控制。

通过将其变量名称用于 PWM 参数，指定要使用的已启用脉冲发生器。

EN 输入为 TRUE 时，PWM_CTRL 指令根据 ENABLE 输入的值启动或停止所标识的 PWM。脉冲宽度由相关 Q 字输出地址中的值指定。

由于 CPU 在 CTRL_PWM 指令执行后处理请求，所以参数 BUSY 总是报告 FALSE。如果检测到错误，则 ENO 设置为 FALSE 且参数 STATUS 包含条件代码。

CPU 第一次进入 RUN 模式时，脉冲宽度将设置为在设备组态中组态的初始值。根据需要将值写入设备配置中指定的 Q 字位置（“输出地址”/“起始地址：”）以更改脉冲宽度。使用指令（如移动、转换、数学）或 PID 功能框将所需脉冲宽度写入相应的 Q 字。必须使用 Q 字值的有效范围（百分数、千分数、万分数或 S7 模拟格式）。

说明

无法强制分配给 PWM 和 PTO 的数字量 I/O 点

在设备组态期间分配脉冲宽度调制 (PWM, Pulse-Width Modulation) 和脉冲串输出 (PTO, Pulse-Train Output) 设备使用的数字量 I/O 点。将数字 I/O 点分配给这些设备之后，无法通过监视表格强制功能修改所分配的 I/O 点的地址值。

表格 9-187 STATUS 参数的值

STATUS	说明
0	无错误
80A1	PWM 标识符未寻址到有效的 PWM。

9.8.2 CTRL_PTO（脉冲串输出）

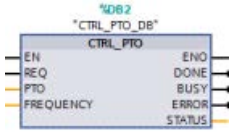
PTO 指令以指定频率提供 50% 占空比输出的方波。可以使用 CTRL_PTO 指令分配无工艺对象 (TO) 轴数据块 (DB) 的频率。

此指令要求脉冲发生器。必须在硬件配置中激活脉冲发生器并选中信号类型。如需了解更多信息，请参见“为 PWM 或 PTO 组态脉冲通道” (页 496)。

可以访问“扩展任务卡”(Task Cards, Extended) 指令中的 CTRL_PTO 指令。

表格 9-188 CTRL_PTO （脉冲串输出）指令

LAD / FBD[1]	SCL[2]	说明

	`"CTRL_PTO_DB"(`
	`REQ:=_bool_in_,`
	`PTO:=_uint_in_,`
	`FREQUENCY:=_udint_in_,`
	`DONE=>_bool_out_,`
	`BUSY=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		PTO 指令允许用户控制方波（50% 占空比）输出的频率。

[1] 插入该指令后，STEP 7 显示用于创建相关数据块的 “调用选项” (Call Options) 对话框。

[2] 在 SCL 示例中，“CTRL_PTO_DB”是背景 DB 的名称。

表格 9-189 参数的数据类型

参数和类型		数据类型	说明
EN	IN	Bool	1 = 指令已激活 0 = 指令已禁用
REQ	IN	Bool	1 = 将 PTO 输出频率设置为 FREQUENCY 中的输出值 0 = PTO 无修改
PTO	IN	HW_PTO (Word)	PTO 标识符：脉冲发生器的硬件 ID： ● 已启用的脉冲发生器的名称将变为 “常量”(constant) 变量表中的变量，并可用作 PTO 参数。（默认值 = 0） ● 该硬件 ID 位于 “设备视图”(Device view) 的 “脉冲发生器属性”(Properties of the pulse generator) 中。脉冲发生器的硬件 ID 同时位于系统常量中。（默认值 = 0）
FREQUENCY	IN	UDInt	PTO 所需频率（赫兹）。此值仅适用于当 REQ = 1 时（默认值为 0 Hz）
DONE	OUT	Bool	函数已成功执行，未发生任何错误（默认值：0）
BUSY	OUT	Bool	功能忙（默认值：0）

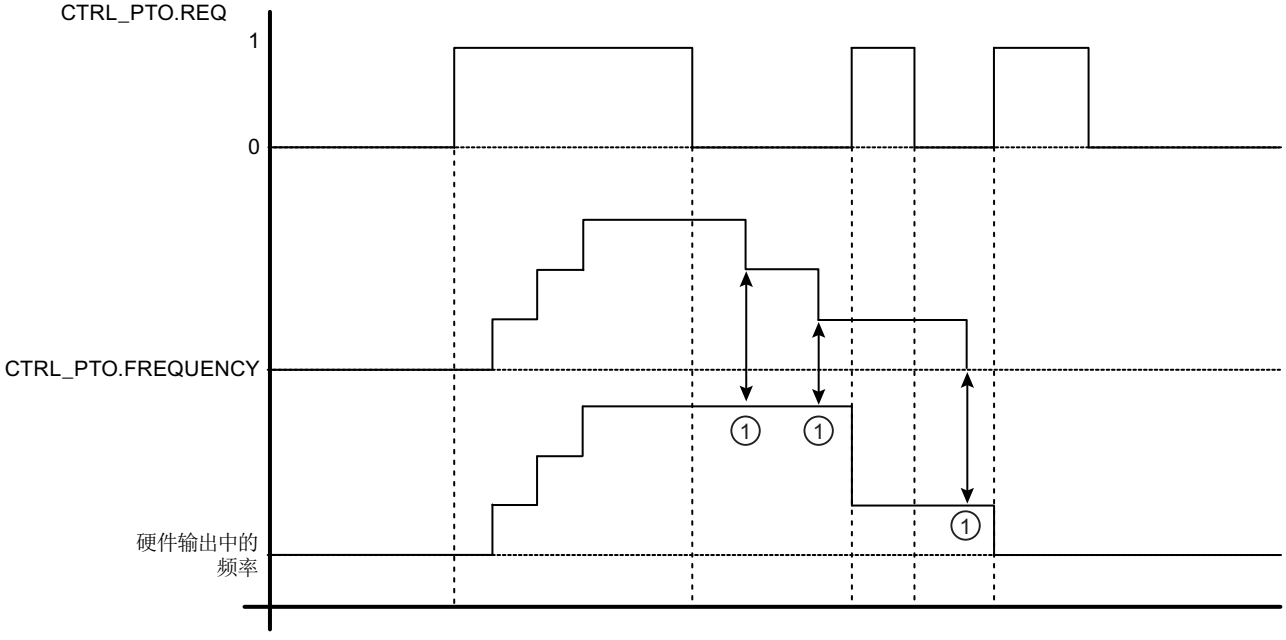
参数和类型		数据类型	说明
ERROR	OUT	Word	检测到错误（默认值：0）
STATUS	OUT	Word	执行条件代码（默认值：0）

CTRL_PTO 指令将参数信息存储在 DB 中。数据块参数不是由用户单独更改的，而是由 CTRL_PTO 指令进行控制。

通过将其变量名称或硬件标识符用于 PTO 参数，指定要使用的已启用脉冲发生器。

当 EN 输入为 TRUE 时，CTRL_PTO 指令启动或停止所标识的 PTO。当 EN 输入为 FALSE 时，不执行 CTRL_PTO 指令且 PTO 保留其当前状态。

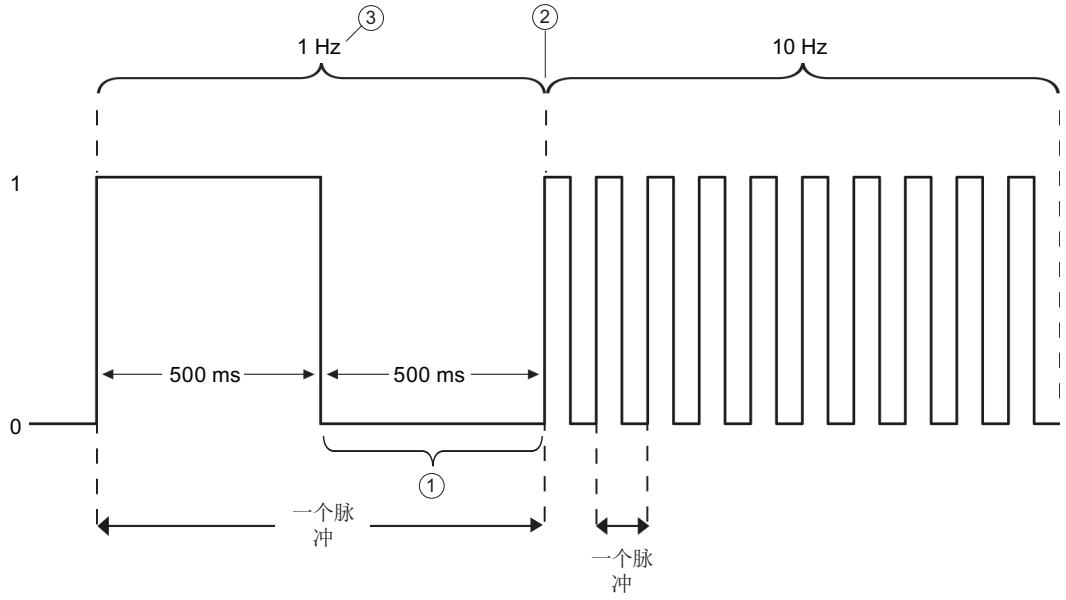
当将 REQ 输入设置为 TRUE 时，FREQUENCY 值生效。如果 REQ 为 FALSE，则无法修改 PTO 的输出频率，且 PTO 继续输出脉冲。



① 当 REQ = 0 时，输出频率无更改

由于 CTRL_PTO 指令只启动 PTO，CTRL_PTO 指令立即结束。因此，始终不要开启 BUSY 输出。只要不发生错误，将会进行 DONE 输出。如果检测到错误，则 ERROR 参数设置为 TRUE，且 STATUS 参数包含条件代码。

当用户用给定的频率激活 CTRL_PTO 指令，S7-1200 以给定的频率输出脉冲串。用户可随时更改所需频率。在修改频率时，S7-1200 会在修改为新的所需频率前结束当前脉冲。例如，如果所需频率为 1 Hz（用时 1000 ms 完成）并且在 500 ms 后用户将频率修改为 10 Hz，频率将会在 1000 ms 时间周期结束时被修改。



① 在 500 ms 后用户将频率修改为 10 Hz。

② 1 Hz 脉冲必须在频率修改为新的 10 Hz 频率前结束。

③ 1 Hz 对应 1000 ms

脉冲发生器硬件对象具有以下限制：仅一个指令可以将脉冲发生器作为 PTO 使用，且硬件组态编辑器对脉冲发生器的使用进行管理。尝试访问 PTO 的其它指令返回了一个错误：“0x8090”（具有指定硬件 ID 的脉冲发生器正在使用中。）

说明

无法强制分配给 PWM 和 PTO 的数字量 I/O 点

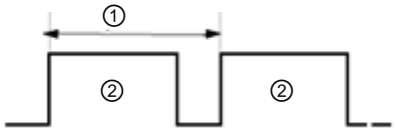
在设备组态期间分配脉冲宽度调制 (PWM) 和脉冲串输出 (PTO) 设备使用的数字量 I/O 点。将数字 I/O 点分配给这些设备之后，无法通过监视表格强制功能修改所分配的 I/O 点的地址值。

表格 9-190 STATUS 参数错误代码值

错误代码 (W#16#...)	说明
0	无错误
0x8090	具有指定硬件 ID 的脉冲发生器正在使用中。
0x8091	频率超出范围。所需频率超出所选脉冲输出的最大频率。
0x80A1	PTO 标识符（硬件 ID）未寻址到有效的 PTO。

错误代码 (W#16#...)	说明
0x80D0	具有指定硬件 ID 的脉冲发生器未激活。在 CPU 属性的“脉冲发生器 (PTO/PWM)”(Pulse generators (PTO/PWM)) 中，激活该脉冲发生器。
0x80D1	具有指定硬件 ID 的脉冲发生器无 PTO 选择。在“硬件配置”(Hardware Configuration) 中选择 PTO。

9.8.3 脉冲输出的作用



① 循环时间

② 脉冲宽度

脉冲宽度可表示为循环时间的百分数（0 到 100）、千分数（0 到 1000）、万分数（0 到 10000）或 S7 模拟格式。

脉冲宽度可从 0（无脉冲，始终关闭）到满刻度（无脉冲，始终打开）变化。

由于 PWM 输出可从 0 到满刻度变化，因此可提供在许多方面都与模拟输出相同的数字输出。例如，PWM 输出可用于控制电机的速度，速度范围可以是从停止到全速；也可用于控制阀的位置，位置范围可以是从闭合到完全打开。

在硬件配置中组态频率 (页 496)。从用户程序中控制脉冲宽度。

有四种脉冲发生器可用于控制高速脉冲输出功能：PWM 和脉冲串输出 (PTO, Pulse train output)。PTO 由运动控制指令使用。可将每个脉冲发生器指定为 PWM 或 PTO，但不能指定为既是 PWM 又是 PTO。

可以使用板载 CPU 输出，也可以使用可选的信号板输出。下表列出了输出点编号（假定使用默认输出组态）。如果更改了输出点编号，则输出点编号将为用户指定的编号。请注意，PWM 仅需要一个输出，而 PTO 每个通道可选择使用两个输出。如果脉冲功能不需要输出，则相应的输出可用于其它用途。有关 I/O 分配的情况，请参见下表。

下表显示了默认的 I/O 分配；但是，可将这四种脉冲发生器组态为任意内置 CPU 或 SB 数字量输出。不同的输出点支持不同的电压与速度，因此分配 PWM/PTO 位置时要将该因素考虑在内。

说明

用户程序中的其它指令无法使用脉冲串输出。

将 CPU 或信号板的输出组态为脉冲发生器时（与 PWM 或运动控制 PTO 指令配合使用），会从 Q 存储器中移除相应的输出地址，并且这些地址在用户程序中不能用于其它用途。如果用户程序向用作脉冲发生器的输出写入值，则 CPU 不会将该值写入到物理输出。

说明

可以释放 PTO 方向输出以在程序中的其它位置使用。

每个 PTO 需要分配两个输出：一个作为脉冲输出，一个作为方向输出。可以只使用脉冲输出而不使用方向输出。随后可以释放方向输出以用于用户程序中的其它用途。

表格 9-191 脉冲发生器的默认输出分配 [3]

说明	脉冲	方向
PTO1		
内置 I/O	Q0.0	Q0.1
SB I/O	Q4.0	Q4.1
PWM1		
内置输出	Q0.0	-
SB 输出	Q4.0	-
PTO2		
内置 I/O	Q0.2	Q0.3
SB I/O	Q4.2	Q4.3
PWM2		
内置输出	Q0.2	-
SB 输出	Q4.2	-
PTO3		
内置 I/O	Q0.4[1]	Q0.5[1]
SB I/O	Q4.0	Q4.1
PWM3		
内置输出	Q0.4[1]	-
SB 输出	Q4.1	-
PTO4		
内置 I/O	Q0.6[2]	Q0.7[2]
SB I/O	Q4.2	Q4.3
PWM4		

说明	脉冲	方向
内置输出	Q0.6[2]	-
SB 输出	Q4.3	-

[1] CPU 1211C 没有输出 Q0.4、Q0.5、Q0.6 或 Q0.7。因此，这些输出不能在 CPU 1211C 中使用。

[2] CPU 1212C 没有输出 Q0.6 或 Q0.7。因此，这些输出不能在 CPU 1212C 中使用。

[3] 该表适用于 CPU 1211C、CPU 1212C、CPU 1214C、CPU 1215C 以及 CPU 1217C PTO/PWM 功能。

9.8.4 为 PWM 或 PTO 组态脉冲通道

要准备 PWM 或 PTO 操作，首先通过选择 CPU 在设备组态中组态脉冲通道，然后选择脉冲发生器 (PTO/PWM)，并选择 PWM1/PTO1 到 PWM4/PTO4。启用脉冲发生器（复选框）。如果启用一个脉冲发生器，将为该特定脉冲发生器分配一个唯一的默认名称。可通过在“名称：”(Name:) 编辑框编辑名称来更改它，但是名称必须是唯一的。已启用的脉冲发生器的名称将变为“常量”(constant) 变量表中的变量，并可作为以下参数使用：

- CTRL_PWM 指令的 PWM 参数
- CTRL_PTO 指令的 PTO 参数

也可在“注释：”(Comment:) 编辑框中写入有关此特定脉冲发生器的注释。

表格 9-192 CPU 输出：最大频率 (PTO) 和最小循环时间 (PWM)

CPU	CPU 输出通道	PTO 最大频率	PWM 最小循环时间
1211C	Qa.0 到 Qa.3	100 kHz	10 µs
1212C	Qa.0 到 Qa.3	100 kHz	10 µs
	Qa.4、Qa.5	20 kHz	50 µs
1214C 和 1215C	Qa.0 到 Qa.3	100kHz	10 µs
	Qa.4 到 Qb.1	20 kHz	50 µs
1217C	DQa.0 到 DQa.3（.0+，.0- 到 .3+，.3-）	1 MHz	1 µs
	DQa.4 到 DQb.1	100 kHz	10 µs

表格 9-193 SB 信号板输出：最大频率 (PTO) 和最小循环时间 (PWM)

SB 信号板	SB 输出通道	PTO 最大频率	PWM 最小循环时间
SB 1222，200 kHz	DQe.0 到 DQe.3	200kHz	5 µs
SB 1223，200 kHz	DQe.0，DQe.1	200kHz	5 µs
SB 1223	DQe.0，DQe.1	20 kHz	50 µs

说明

上表中给出了每个 CPU 和信号板输出的最小循环时间。但是，当所组态 PWM 脉冲发生器的循环时间小于此硬件的最短循环时间时，TIA Portal 并不会提醒用户。您的应用可能因此会出现问题，因而请务必确保不会超出硬件限制。

说明

当您设置 PWM 信号的脉宽时，如果时基为“毫秒”，实际脉宽（脉冲为高电平的时间）必须大于或等于 1 毫秒。如果时基为“微秒”，实际脉宽必须大于或等于 1 微秒。如果脉宽小于 1 倍“时基”，输出将关断。

例如，周期时间为 10 微秒时，百分之 5 的脉冲持续时间可得到 0.5 微秒的脉宽。因为该值小于 1 微秒，PWM 信号关闭。

参数分配

在参数分配部分中，用户可以组态输出脉冲的参数。根据选择 PWM 或 PTO，以下选项可供使用：

- 信号类型：将脉冲输出组态为 PWM 或 PTO。有关 PTO 选择的更多信息，请参见“相位调整” (页 664)：
  - PWM
  - PTO（脉冲 A 和方向 B）
  - PTO（向上脉冲 A 和向下脉冲 B）
  - PTO（A/B 相移）
  - PTO（A/B 相移 - 四相）
- 时间基准（仅适用于 PWM）：请选择使用的时间单位：
  - 毫秒
  - 微秒

- 脉冲宽度格式（仅适用于 PWM）：分配脉冲持续时间（宽度）的精度：
  - 百分数（0 到 100）
  - 千分数（0 到 1000）
  - 万分数（0 到 10000）
  - S7 模拟格式（0 到 27648）
- 循环时间（仅适用于 PWM）：分配完成一次脉冲需要的持续时间（循环时间是高脉冲时间与低脉冲时间的和）。可以通过选中复选框“允许在运行时修改循环时间”(Allow runtime modification of the cycle time)，在运行时更改循环时间。有关详细信息，参见下文中“I/O 地址”部分。范围是 1 到 16,777,215 个时间单位。
- 初始脉冲持续时间（仅适用于 PWM）：分配第一次脉冲的脉冲持续时间。可通过使用 I/O 地址中组态的 Q 字地址，在运行系统中更改初始脉冲持续时间值。范围取决于脉冲持续时间格式。
- 允许在运行时修改循环时间（仅适用于 PWM）：如果选择该选项，您的程序便能在程序处于运行状态时，修改 PWM 信号的循环时间。有关详细信息，参见下文中“I/O 地址”部分。

说明

设置 PWM 信号的脉冲持续时间时，请务必考虑附录 A 中规定的输出通道开关延迟。输出端测得的实际脉冲持续时间可能会大于选择的脉冲持续时间。脉冲持续时间的增加对于小脉冲持续时间和高频的影响更为突出。请务必检查输出端测得的脉冲持续时间是否与用户要求匹配。

确定脉冲持续时间值

“初始脉冲持续时间”乘以“循环时间”可得出“脉冲持续时间”。选择“时基”、“脉冲持续时间格式”、“循环时间”和“初始脉冲持续时间”时，请谨记：整个“脉冲持续时间”不能为小数值。如果生成的“脉冲持续时间”是一个小数值，则应调整“初始脉冲持续时间”或更改时基，从而生成一个整数值。

以下是两个示例：

- 示例 1：如果选择以下值：
  - 时基 = 毫秒 (ms)
  - 脉冲持续时间格式 = 百分数（0 到 100）
  - 循环时间 = 3 ms
  - 初始脉冲持续时间 = 75

  生成的“脉冲持续时间”= 0.75 x 3 ms = 2.25 ms

  此“脉冲持续时间”值为小数值时，会造成操作 CTRL_PWM 指令时出错。“脉冲持续时间”值必须为整数值。
- 示例 2：如果选择以下值：
  - 时基 = 微秒 (µs)
  - 脉冲持续时间格式 = 百分数（0 到 100）
  - 循环时间 = 3000 µs
  - 初始脉冲持续时间 = 75

  生成的“脉冲持续时间”= 0.75 x 3000 µs = 2250 µs

  此“脉冲持续时间”值为整数值，CTRL_PWM 指令可使用该值正常操作。

硬件输出

在硬件输出部分，从下拉菜单中选择输出通道。基于组态，可选择一个或两个输出。如果确实为脉冲发生器分配输出通道，那么此输出通道不可被另一个脉冲发生器、HSC、或过程映像寄存器所使用。

说明

用户程序中的其它指令无法使用脉冲发生器输出

将 CPU 或信号板的输出组态为脉冲发生器时（与 PWM、PTO 或运动控制指令配合使用），会从 Q 存储器中移除相应的输出地址，且这些地址在程序中不能用于其它用途。如果您的程序向用作脉冲发生器的输出写入某个值，则 CPU 不会将该值写入到物理输出。

I/O 地址

PWM 为“脉冲持续时间”(Pulse duration) 指定了 Q 存储器的两个字节。PWM 运行时，可以在分配的 Q 存储器中修改该值以及更改“脉冲持续时间”(Pulse duration)。

在“I/O 地址”(I/O Address) 部分，在要用于存储“脉冲持续时间”(Pulse duration) 值的位置输入 Q 字地址。

PWM“脉冲持续时间”(Pulse duration) 值的默认地址如下所示：

- PWM1：QW1000
- PWM2：QW1002
- PWM3：QW1004
- PWM4：QW1006

对于 PWM，每次 CPU 从 STOP 模式转换为 RUN 模式时，此处值将控制脉冲持续时间并被初始化为“初始脉冲持续时间：”(Initial pulse duration:) 值（如以上分配）。可在运行系统中通过更改 Q 字值来更改脉冲持续时间。脉冲宽度值的范围取决于参数分配下组态的脉冲持续时间格式。

您还可以为 PWM 信号的“循环时间”(Cycle time) 额外分配 Q 存储器的 4 个字节。关于 PWM 信号图，请参见“脉冲输出的操作” (页 494)。选中“允许在运行时修改循环时间”(Allow runtime modification of the cycle time) 复选框后，前两个字节用于保持“脉冲持续时间” (Pulse duration) 值，后四个字节用于保持“循环时间”(Cycle time) 值。

PWM 运行时，您可以修改分配给 PWM 的 Q 存储器结尾的双字值。这会改变 PWM 信号的“循环时间”(Cycle time)。例如，启用该选项后，CPU 会为 PWM1 分配六个字节，并由您确定使用 QB1008 到 QB1013。下载程序并启动 PWM 后，可以使用 QW1008 修改“脉冲持续时间”(Pulse duration)，以及使用 QD1010 修改“循环时间”(Cycle time)。

CPU 每次从 STOP 切换为 RUN 模式时，CPU 均会将 Q 存储器中的“循环时间”(Cycle time) 值初始化为上述“参数分配”部分中分配的“循环时间”(Cycle time) 值。Q 存储器中“循环时间”(Cycle time) 值的单位和取值范围与“参数分配”部分中组态的相同。

选中“允许在运行时修改循环时间”(Allow runtime modification of the cycle time) 复选框后，TIA Portal 会自动为输出地址选择新的地址。新的输出地址不能与脉冲发生器的默认地址相同。TIA Portal 将使用六个连续字节的下一个可用块。如果在搜索到 Q 存储器末尾前未找到 Q 存储器的可用块，则会从 Q 存储器的地址“0”开始继续搜索可用块。

针对 PTO 组态的脉冲发生器不使用 Q 字地址。

9.9 配方和数据日志

9.9.1 配方

9.9.1.1 配方概述

配方数据存储

- 在项目中创建的配方数据块必须存储在 CPU 装载存储器中。 可以使用内部 CPU 存储器或外部存储器“程序”卡。
- 另一个必须创建的 DB 是活动配方数据块。 此 DB 必须在工作存储器中，其中使用程序逻辑读取或写入一个活动配方记录。

配方数据管理

配方数据块使用一个产品配方记录数组。 配方数组的每个元素代表一种不同的配方形式，各个配方以一组共同的成分为基础。

- 创建 PLC 数据类型或结构，以定义一个配方记录中的所有成分。 此数据类型模板重复使用于所有配方记录。 根据分配给配方成分的起始值而产生不同的产品配方。
- 使用 READ_DBL 指令，可以随时将配方从配方数据块（装载存储器中的所有配方）传送到活动配方数据块（工作存储器中的一个配方）。 配方记录移动到工作存储器后，程序逻辑便可读取成分值并开始生产运行。 此过程将配方数据需要的 CPU 工作存储器使用量降到最低。
- 如果在生产运行期间使用 HMI 设备调整活动配方成分值，可以使用 WRIT_DBL 指令将修改的值写入配方数据块。

配方导出（从配方数据块到 CSV 文件）

可以使用 RecipeExport 指令将完整的配方记录集生成为一个 CSV 文件。 未使用的配方记录也被导出。

配方导入（从 CSV 文件到配方数据块）

完成配方导出操作后，即可将生成的 CSV 文件用作数据结构模板。

1. 使用 CPU web 服务器中的文件浏览器页面将现有配方 CSV 文件从 CPU 下载到 PC
2. 使用 ASCII 文本编辑器修改配方 CSV。 可以修改分配给成分的起始值，但不能修改数据类型或数据结构
3. 将修改的 CSV 文件从 PC 再次上传到 CPU。 但是，在 CPU Web 服务器允许上传操作之前，必须删除或重命名 CPU 装载存储器中的旧 CSV 文件（具有相同名称）。
4. 将修改的 CSV 文件上传到 CPU 后，便可以使用 RecipeImport 指令将新的起始值从修改的 CSV 文件（在 CPU 装载存储器中）传送到配方数据块（在 CPU 装载存储器中）。

9.9.1.2 配方示例

配方实例

下表显示如何准备用于配方数据块的配方信息。该实例配方数据块存储 5 条记录，其中三条已使用。第四条和第五条记录留空以供将来扩展。表中的每行表示一条记录，存储配方名称、成分数据类型和成分值。

productname	water	barley	wheat	hops	yeast	waterTmp	mashTmp	mashTime	QTest
Pils	10	9	3	280	39	40	30	100	0
Lager	10	9	3	150	33	50	30	120	0
BlackBeer	10	9	3	410	47	60	30	90	1
Not_used	0	0	0	0	0	0	0	0	0
Not_used	0	0	0	0	0	0	0	0	0

创建配方数据块

说明

配方数据块的规则

- 配方数据块必须包含一维数组，结构是 PLC 数据类型或结构。本配方实例显示如何使用 PLC 数据类型创建配方数据块。
- 在实例中，成分的数据类型都是 UINT 数据类型。成分数据类型也可以是除结构以外的任何混合数据类型。在配方数据块数组元素中，不允许 PLC 数据类型中存在结构，也不允许结构中嵌套结构。

首先，创建新的 PLC 数据类型

添加新的 PLC 数据类型，名称作为配方类型。在下图中，“Beer_Recipe”是新的复合 PLC 数据类型，它存储一系列简单数据类型。“Beer_Recipe”PLC 数据类型是一个数据模板，它在每个配方数据块记录以及活动配方数据块中重复使用。输入所有实例配方共用的成分名称和数据类型。各成分值以后在配方数据块中添加。

Beer_Recipe

		Name	Data type	Default value
1		productname	String[20]	'Beer_Recipe'
2		water	UInt	0
3		barley	UInt	0
4		wheat	UInt	0
5		hops	UInt	0
6		yeast	UInt	0
7		waterTmp	UInt	0
8		mashTmp	UInt	0
9		mashTime	UInt	0
10		QTest	UInt	0

第二步，创建配方数据块

- 将配方数据块创建为全局数据块，并启用数据块属性“仅存储在装载存储器中”(Only store in load memory)。
- 配方数据块的名称用作相应 CSV 文件的文件名。数据块名称中使用的字符必须遵守 Windows 文件系统命名限制。字符 \ / : * ? " < > | 及空格字符均不允许使用。
- 配方数组分配是 "Products" 作为 Array [1.. 5] of "Beer_Recipe"。数组大小 5 是可创建的配方风味的最大数目。
- 配方成分值添加为数据块起始值。

在下图中，展开的“BlackBeer”配方显示了配方记录的所有成分。

Recipe_DB

	Name	Data type	Offset	Start value
1	Static			
2	Products	Array [1 .. 5] of "Beer_Recipe"	...	
3	Products[1]	"Beer_Recipe"	...	
4	Products[2]	"Beer_Recipe"	...	
5	Products[3]	"Beer_Recipe"	...	
6	productname	String[20]	...	'BlackBeer'
7	water	UInt	...	10
8	barley	UInt	...	9
9	wheat	UInt	...	3
10	hops	UInt	...	410
11	yeast	UInt	...	47
12	waterTmp	UInt	...	60
13	mashTmp	UInt	...	30
14	mashTime	UInt	...	90
15	QTest	UInt	...	1
16	Products[4]	"Beer_Recipe"	...	
17	Products[5]	"Beer_Recipe"	...	

配方导出（从配方数据块到 CSV 文件）

执行“RecipeExport (页 506)”，可将配方数据块数据传送到 CSV 文件，如下面的文本文件所示。

```
Recipe_DB.csv

index,productname,water,barley,wheat,hops,yeast,waterTmp,
mashTmp,mashTime,QTest
1,”Pils”,10,9,3,280,39,40,30,100,0
2,”Lager”,10,9,3,150,33,50,30,120,0
3,”BlackBeer”,10,9,3,410,47,60,30,90,1
4 ”Not_used”,0,0,0,0,0,0,0,0,0
5 ”Not_used”,0,0,0,0,0,0,0,0,0
```

配方导入（从 CSV 文件到配方数据块）

1. 使用 Web 服务器中的文件浏览器页面 (页 1065)将现有配方 CSV 文件从 CPU 装载存储器下载到 PC
2. 使用 ASCII 文本编辑器修改配方 CSV。可以修改分配给成分的起始值，但不能修改数据类型或数据结构
3. 将修改的 CSV 文件从 PC 回传到 CPU。但是，在 Web 服务器允许上传操作之前，必须删除或重命名 CPU 装载存储器中的旧 CSV 文件（具有相同名称）。
4. 将修改的 CSV 文件上传到 CPU 后，便可以使用 RecipeImport 指令将新的起始值从修改的 CSV 文件（在 CPU 装载存储器中）传送到配方数据块（在 CPU 装载存储器中）。

CSV 文件必须精确匹配对应的配方数据块结构

- 可以更改 CSV 文件中的值，但不允许更改结构。RecipeImport 指令要求记录和成分的数量与目标配方数据块结构完全匹配。否则 RecipeImport 的执行会失败。例如，如果在配方数据块中定义了 10 个配方但实际使用的只有 6 个配方，则 CSV 文件的第 7 至第 10 行也传送到数据块。必须检查该数据是否有效。例如，对于未使用配方记录中的产品名称，可以赋予一个变量“Not_used”。
- 如果向文本文件添加数据记录并导入修改的文件，请确保您分配的配方数据块数组限制可以有足够的元素用于所有配方记录。
- 导出到 CSV 文件期间会自动生成索引编号。如果创建附加数据记录，请相应添加连续的索引编号。
- 执行 RecipeImport 会检查 CSV 文件数据，判断结构是否正确以及值是否与相关配方数据块中分配的数据类型匹配。例如，Bool 数据类型不能存储整数值，否则 RecipeImport 的执行会失败。

在 Excel 中显示 CSV 配方数据

可以在 Excel 中打开 CSV 文件，以便于阅读和编辑。如果未将逗号识别为十进制分隔符，请使用 Excel 导入功能以结构化形式输出该数据。

	A	B	C	D	E	F	G	H	I	J	K
1	index	product	water	barley	wheat	hops	yeast	waterTmp	mashTmp	mashTime	QTest
2	1	"Pils"	10	9	3	280	39	40	30	100	0
3	2	"Lager"	10	9	3	150	33	50	30	120	0
4	3	"BlackBeer"	10	9	3	410	47	60	30	90	1
5	4	"Not_used"	0	0	0	0	0	0	0	0	0
6	5	"Not_used"	0	0	0	0	0	0	0	0	0

说明

PLC 数据类型元素名称字段中的逗号

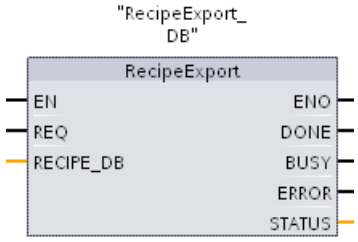
请勿在配方中所用 PLC 数据类型元素的名称字段中使用逗号。如果在名称字段中使用逗号，Excel 会在显示 .csv 文件时插入额外的列。当您编辑配方记录文件起始值时，这些额外的列可能会引入错误。

9.9.1.3 传送配方数据的程序指令

RecipeExport（导出配方）

表格 9-194 RecipeExport 指令

LAD/FBD	SCL	说明

	`"RecipeExport_DB"(`
	`req:=_bool_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	
	`Recipe_DB:=_variant_inout_);`
		“RecipeExport”指令将所有配方记录从配方数据块导出到 CSV 文件格式。CSV 文件包含产品名称、成分名称和起始值。CSV 文件存储在内部装载存储器中；如果安装了可选的外部“程序”存储器卡，则 CSV 文件也可以存储在外部装载存储器中。
		导出操作由“REQ”参数触发。BUSY 参数在导出处理期间会设置为“1”。RecipeExport 的执行停止后，BUSY 复位为“0”，并且在 DONE 参数中用“1”表示操作完成。如果执行期间发生错误，则参数 ERROR 和 STATUS 会指示结果。

在配方可以导出之前，必须创建配方数据块。配方数据块的名称用作新 CSV 文件的文件名。如果具有相同名称的 CSV 文件已经存在，则在导出操作期间会被覆盖。

可以使用 CPU 的内置 Web 服务器的文件浏览器页面 (页 1065)来访问配方 CSV 文件。该文件被置于 CPU 装载存储器根目录的配方文件夹中。

表格 9-195 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	控制参数 REQUEST：在上升沿激活导出。
RECIPE_DB	输入/输出	Variant	指向配方数据块的指针。有关详细信息，请参见“配方数据块实例 (页 502)”。数据块名称中的字符必须遵守 Windows 文件系统命名限制。\ / : * ? " < > | 及空格字符均不允许使用。
DONE	OUT	Bool	上一请求已完成且没有出错后，DONE 位将保持为 TRUE 一个扫描周期时间。（默认值：False）
BUSY	OUT	Bool	RecipeExport 的执行 • 0：没有操作正在进行 • 1：有操作正在进行

参数和类型		数据类型	说明
ERROR	OUT	Bool	上一请求因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。STATUS 参数中的错误代码值仅在 ERROR = TRUE 的一个扫描周期内有效。 • 0：没有警告或错误 • 1：发生错误。STATUS 参数提供错误类型的信息。
STATUS	OUT	Word	执行条件代码

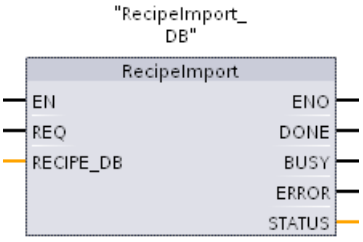
表格 9-196 ERROR 和 STATUS 的值

ERROR	STATUS (W#16#....)	说明
0	0000	无错误
0	7000	无 REQ 沿时调用：BUSY = 0，DONE = 0
0	7001	有 REQ 沿时首次调用（工作）：BUSY = 1，DONE = 0
0	7002	第 N 次调用（工作）：BUSY = 1，DONE = 0
1	8070	所有实例存储器都在使用。
1	8090	文件名称包含无效字符
1	8091	无法处理使用 RECIPE_DB 引用的数据结构。
1	8092	RECIPE_DB 中指定的数据结构超过 5000 字节
1	80B3	MC 或内部装载存储器中没有足够的空间
1	80B4	MC 受写保护
1	80B6	未启用配方数据块属性“仅存储在装载存储器中”(Only store in load memory)。
1	80C0	CSV 文件被临时锁定
1	80C1	DB 被临时锁定

RecipeImport（导入配方）

表格 9-197　RecipeImport 指令

LAD/FBD	SCL	说明

	`"RecipeImport_DB"(`
	`req:=_bool_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	
	`Recipe_DB:=_variant_inout_);`
		“RecipeImport”指令将配方数据从 CPU 装载存储器中的 CSV 文件导入到 RECIPE_DB 参数引用的配方数据块中。导入过程中，配方数据块中的起始值被覆盖。导入操作由“REQ”参数触发。BUSY 参数在导入处理期间会设置为“1”。RecipeImport 的执行停止后，BUSY 复位为“0”，并且在 DONE 参数中用“1”表示操作完成。如果执行期间发生错误，则参数 ERROR 和 STATUS 会指示结果。

表格 9-198　参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	控制参数 REQUEST：在上升沿激活导入。
RECIPE_DB	输入/输出	Variant	指向配方数据块的指针。有关详细信息，请参见 “配方数据块实例 (页 502)”。数据块名称中的字符必须遵守 Windows 文件系统命名限制。\ / : * ? " < > | 及空格字符均不允许使用。
DONE	OUT	Bool	上一请求已完成且没有出错后，DONE 位将保持为 TRUE 一个扫描周期时间。（默认值：False）
BUSY	OUT	Bool	• 0 - 无操作正在进行 • 1 - 操作正在进行
ERROR	OUT	Bool	上一请求因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。STATUS 参数中的错误代码值仅在 ERROR = TRUE 的一个扫描周期内有效。
STATUS	OUT	Word	执行条件代码（默认值：0）

只有配方数据块中包含一个与 CSV 文件数据结构一致的结构，才能执行配方导入操作。

CSV 文件规则：

- CSV 文件必须位于内部装载存储器根目录的“Recipes”文件夹中；如果安装了可选的外部“程序”存储器卡，则 CSV 文件也可以位于外部装载存储器的相应文件夹中。
- CSV 文件的名称必须与 RECIPE_DB 参数中的数据块名称相匹配。
- CSV 文件的第一行（标题）包含配方成分的名称。导入期间会忽略第一行。导入过程期间，不会检查 CSV 文件和数据块中配方成分的名称是否一致。
- 在每种情况下，CSV 文件每一行的第一个值都作为配方的索引编号。各个配方按索引的顺序导入。因此，CSV 文件中的索引必须按升序排列并且不能间断（否则，STATUS 参数中会输出错误消息 80B0）。
- CSV 文件中包含的配方数据记录数不能超过配方数据块中提供的数量。数据记录的最大值由数据块中的数组限值指出。

表格 9-199 ERROR 和 STATUS 的值

ERROR	STATUS (W#16#....)	说明
0	0000	无错误
0	7000	无 REQ 沿时调用：BUSY = 0，DONE = 0
0	7001	有 REQ 沿时首次调用（工作）：BUSY = 1，DONE = 0
0	7002	第 N 次调用（工作）：BUSY = 1，DONE = 0
1	8070	所有实例存储器都在使用。
1	8090	文件名称包含无效字符。
1	8092	没有匹配的 CSV 文件可导入。可能原因：CSV 文件的名称与配方 DB 的名称不匹配。
1	80C0	CSV 文件被临时锁定。
1	80C1	数据块被临时锁定。
1	80B0	CSV 文件中的索引编号不连续、未按升序排列或超过数据块中的最大数（数组限值）。
1	80B1	配方数据块的结构与 CSV 文件不匹配：CSV 文件包含过多的字段。
1	80B2	配方数据块的结构与 CSV 文件不匹配：CSV 文件包含过少的字段。
1	80B6	未启用配方数据块属性“仅存储在装载存储器中”(Only store in load memory)。
1	80D0 +n	配方数据块的结构与 CSV 文件不匹配：字段 n 中的数据类型不匹配 (n<=46)。
1	80FF	配方数据块的结构与 CSV 文件不匹配：字段 n 中的数据类型不匹配 (n>46)。

9.9.1.4 配方实例程序

配方示例程序的先决条件

下面列出了配方示例程序的先决条件：

- 一个存储所有配方记录的配方数据块。 配方数据块存储在装载存储器中。
- 在工作存储器中存储一个配方副本的活动配方数据块。

有关配方数据块和相应 CSV 文件的详细信息，请参见“配方数据块实例 (页 502)”。

创建活动配方数据块

在“添加新块”(Add new block) 窗口中：

- 在“添加新块”(Add new block) 窗口中，选择“数据块”(Data block) 按钮
- 在“类型”(Type) 下拉菜单中，选择您先前创建的“Beer_recipe”PLC 数据类型。

不需要起始值。 在将一个配方从配方数据块传送到活动配方数据块时，数据块数据值将置位。 在本实例中，活动配方数据块是 READ_DBL 的目标数据并为 WRITE_DBL 提供源数据。 下图显示 Active_Recipe 数据块。

Active_Recipe

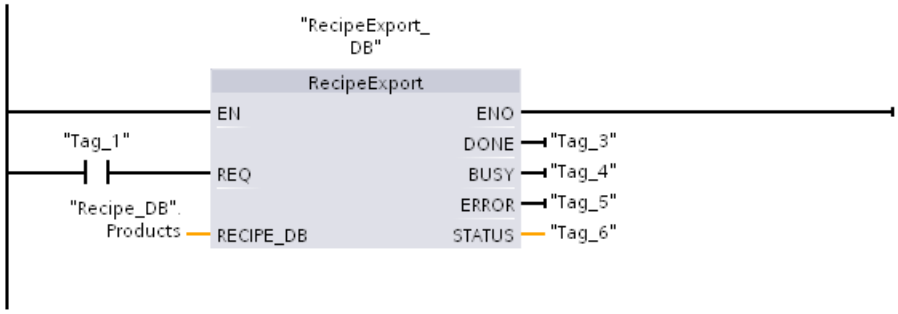
	Name	Data type	Start value
1	Static		
2	productname	String[20]	'Beer_Recipe'
3	water	UInt	0
4	barley	UInt	0
5	wheat	UInt	0
6	hops	UInt	0
7	yeast	UInt	0
8	waterTmp	UInt	0
9	mashTmp	UInt	0
10	mashTime	UInt	0
11	QTest	UInt	0

背景数据块

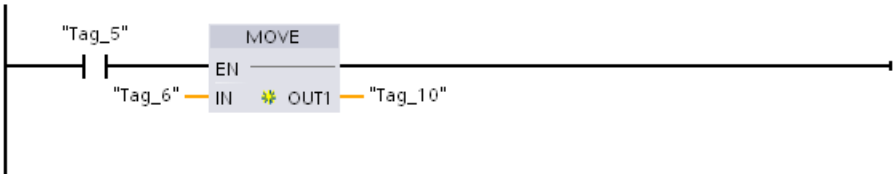
指令 RecipeExport ("RecipeExport_DB") 和 RecipeImport ("RecipeImport_DB") 使用的背景数据块是在将指令置于程序中时自动创建的。 背景数据块用于控制指令的执行，不在程序逻辑中引用。

示例配方程序

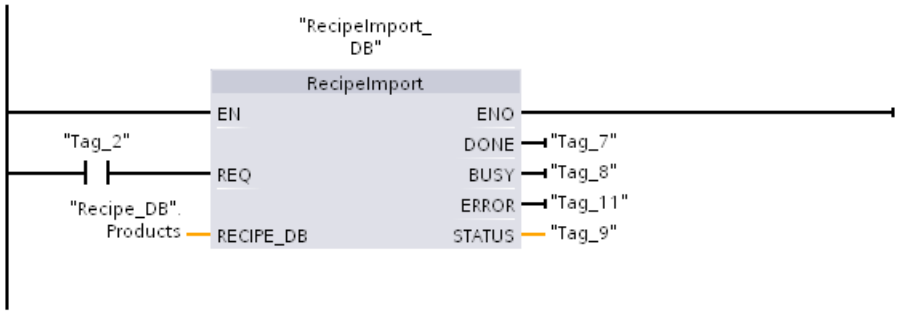
程序段 1 REQ 上升沿启动导出过程。 CSV 文件由配方数据块数据生成并被置于 CPU 存储器配方文件夹。



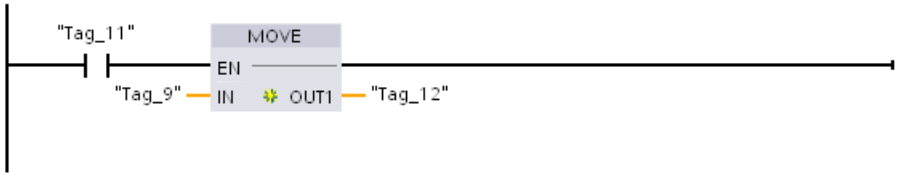
程序段 2 捕获 RecipeExport 执行的 STATUS 输出，考虑到该指令仅在一个扫描周期内有效。



程序段 3 REQ 上升沿启动导入过程。 现有配方数据块载入读取自 CPU 存储器配方文件夹的相应 CSV 文件中的所有配方数据。

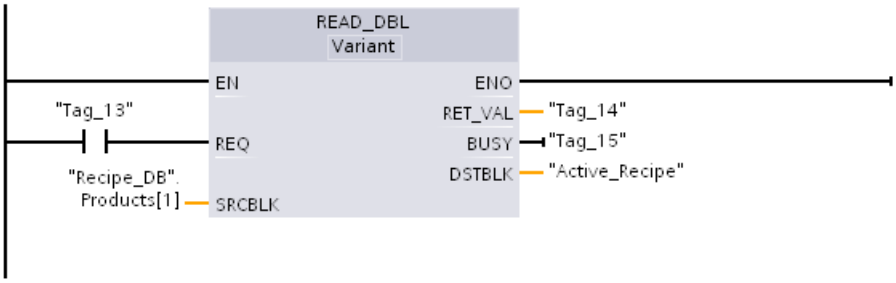


程序段 4 捕获 RecipeImport 执行的 STATUS 输出，考虑到该指令仅在一个扫描周期内有效。

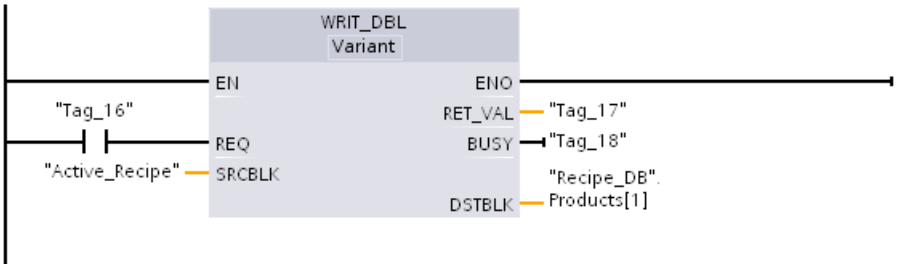


程序 5 READ_DBL 从配方“Recipe_DB”中复制起始值。 结果值[1]（在 CPU 的装载存储器中）至 Active_Recipe DB 的当前值（在 CPU 的工作内存中）。 READ_DBL 执行后，程序逻辑通过在 Active_Recipe 数据块中寻址可以访问配方成分值。 例如，符号地址

("Active_Recipe".productname) 和 ("Active_Recipe.water) 为程序逻辑提供当前配方名称和用水量。



程序段 6 运行期间，HMI 设备可以修改 Active_Recipe 数据块中存储的成分值。 可以通过执行 WRIT_DBL 存储改进的配方数据。 本例中， Recipe_DB 的全部起始值都用于这一个配方 "Recipe_DB"。 结果值 [1] 被 "Active_Recipe" 数据块中的当前值覆盖。



9.9.2 数据日志

控制程序可以使用 Data log 指令将运行数据值存储在永久性日志文件中。CPU 将数据日志文件以标准 CSV （逗号分割值）格式存储在闪存（CPU 或存储卡）中。CPU 按大小预定的循环日志文件形式组织数据记录。

在程序中，可以使用 Data log 指令创建、打开和关闭日志文件，还可以向日志文件写入记录。用户通过创建定义单个日志记录的数据缓冲区来确定需要记录的程序值。CPU 使用该数据缓冲区临时存储新的日志记录。控制程序在运行时会将新的当前值移到缓冲区中。程序将所有当前数据值更新后，可执行 DataLogWrite 指令，将缓冲区数据传送到数据日志记录中。

可以从 Web 服务器的“文件浏览器”页面打开、编辑、保存、重命名或删除数据日志文件。必须有读取权限才能查看文件浏览器，必须有修改权限才能编辑、删除或重命名数据日志文件。

9.9.2.1 数据日志记录结构

DataLogCreate 指令的 DATA 和 HEADER 参数分配日志记录的所有数据元素的数据类型和列标题说明。

DataLogCreate 指令的 DATA 参数

DATA 参数指向用作新日志记录临时缓冲区的存储器，必须将其分配给 M 或 DB 位置。

可以分配整个 DB（源自创建 DB 时分配的 PLC 数据类型），也可分配部分 DB（指定的 DB 元素可以是任何数据类型、数据类型结构、PLC 数据类型或数据数组）。

DataLogCreate 指令将结构数据类型限制为单嵌套级。在这种情况下，字符串数组不视为单嵌套级。DataLogCreate 指令当前不会返回错误。该指令仅处理字符串数组中的第一个字符串。所声明数据元素的总数应与标题参数中指定的列数相对应。可以分配的最大数据元素个数为 253（带时间戳）或 255（不带时间戳）。此限制可保证记录始终处于 Excel 工作表的 256 列限制范围内。

DATA 参数可在“标准”（与 S7-300/400 兼容）或“优化”DB 类型中分配保持性数据元素或非保持性数据元素。

要写入数据日志记录，首先必须用新过程值装载临时 DATA 记录，然后执行 DataLogWrite 指令将新记录值保存到 Datalog 文件中。

DataLogCreate 指令的 HEADER 参数

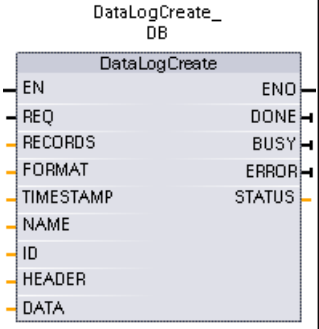
HEADER 参数指向 CSV 文件中编码的数据矩阵的第一行的列标题名称。HEADER 数据必须位于 DB 或 M 存储器，且字符必须遵守标准 CSV 格式规则，各列名称用逗号分隔。数据类型可以是字符串、字节数组或字符数组。字符/字节数组的大小可以增加，其中字符串被限制为最多 255 个字节。HEADER 参数是可选参数。如果未分配 HEADER 参数，则不会在数据日志文件中创建标题行。

9.9.2.2 控制数据日志的程序指令

DataLogCreate（创建数据日志）

表格 9-200 DataLogCreate 指令

LAD/FBD	SCL	说明

	`"DataLogCreate_DB"(`
	`req:=_bool_in_,`
	`records:=_udint_in_,`
	`format:=_uint_in_,`
	`timestamp:=_uint_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`name:=_string_inout_,`
	`ID:=_dword_inout_,`
	`header:=_variant_inout_,`
	`data:=_variant_inout_);`
		创建和初始化数据日志文件。CPU 在 \DataLogs 文件夹中使用 NAME 参数中的名称创建文件，并且以隐式打开以便执行写操作。在程序中，可使用 Data log 指令将运行系统过程数据存入 CPU 的闪存或存储卡中。
		STEP 7 会在插入指令时自动创建关联的背景数据块。

[1] 在 SCL 示例中，“DataLogCreate_DB”是背景数据块的名称。

表格 9-201 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	通过由低到高的（上升沿）信号启动操作。（默认值：False）
RECORDS	IN	UDint	覆盖最旧条目前循环数据日志可存储的最大数据记录数： 不包括标题记录。PLC 装载存储器的可用空间必须足够大，以确保成功创建数据日志。（默认值 - 1）
FORMAT	IN	UInt	数据日志格式： ● 0 - 内部格式（不支持） ● 1 - 逗号分隔值“csv-eng”（默认值）
TIMESTAMP	IN	UInt	数据时间戳格式：日期和时间字段的列标题是可选的。时间戳可使用系统时间（世界协调时间 - UTC）或本地时间。 ● 0 - 无时间戳 ● 1 - 日期和时间戳，系统时间（默认值） ● 2 - 日期和时间戳，本地时间

参数和类型		数据类型	说明
NAME	IN	Variant	数据日志名称：用户可在此提供名称。此变量仅支持 String 数据类型，且必须位于 DB 或本地存储器中。（默认值：''） 该字符串引用为数据日志文件名。使用 ASCII 字符集中的字符，除了字符 \ / : * ? " < > | 及空格字符。
ID	IN/OUT	DWord	数据日志数字标识符：存储该生成值以便与其它数据日志指令配合使用。ID 参数仅用作 DataLogCreate 指令的输出。（默认值：0） 该参数不支持符号名称访问。
HEADER	IN/OUT	Variant	指向 CSV 文件中编码的数据矩阵的第一行的数据日志列标题名称。（默认值：null）。 HEADER 数据必须位于 DB 或 M 存储器。 字符必须遵守标准 CSV 格式规则，各列名称用逗号分隔。数据类型可以是字符串、字节数组或字符数组。字符/字节数组的大小可以增加，其中字符串被限制为最多 255 个字节。 HEADER 参数是可选参数。如果未设置 HEADER 参数，则不会在数据日志文件中创建标题行。
DATA	IN/OUT	Variant	指向记录数据结构、用户自定义类型 (UDT) 或数组。记录数据必须位于 DB 或 M 存储器。 DATA 参数指定数据日志记录的各个数据元素（列）及其数据类型。DataLogCreate 指令将结构数据类型限制为单嵌套级。在这种情况下，字符串数组不视为单嵌套级。DataLogCreate 指令当前不会返回错误。该指令仅处理字符串数组中的第一个字符串。所声明数据元素的个数应与标题参数中指定的列数相对应。可以分配的最大数据元素个数为 253（带时间戳）或 255（不带时间戳）。此限制可保证记录始终处于 Excel 工作表的 256 列限制范围内。
DONE	OUT	Bool	上一请求已完成且没有出错后，DONE 位将保持为 TRUE 一个扫描周期时间。（默认值：False）
BUSY	OUT	Bool	• 0 - 无操作正在进行 • 1 - 操作正在进行

参数和类型		数据类型	说明
ERROR	OUT	Bool	上一请求因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。STATUS 参数中的错误代码值仅在 ERROR = TRUE 的一个扫描周期内有效。
STATUS	OUT	Word	执行条件代码（默认值：0）

CPU 根据 RECORDS 和 DATA 参数按预定的固定大小创建数据日志文件并按循环日志文件形式组织数据记录。当指令返回 DONE = TRUE 时，DataLogCreate 指令为整个数据日志分配持久 CPU 内存。由于存在文件系统管理和相关的值，CPU 中所需的内存大于文件的大小。数据日志的持久性内存保持分配状态，直至 CPU 以下列方式之一取消分配内存：

- 用户程序调用 DataLogDelete 指令
- Web 服务器用户从 Web 服务器中删除数据日志
- SIMATIC Automation Tool 用户从 SIMATIC Automation Tool 中删除数据日志

通过其它方式（例如使用读卡器）删除数据日志文件，不会为数据日志取消分配 CPU 持久内存。

DataLogWrite 指令不断将新记录添加到数据日志文件中，直到已存储 RECORDS 参数指定的最大记录数为止。写入的下一条记录将覆盖最早的记录。另一 DataLogWrite 操作将覆盖下一条最早的数据记录，依此类推。

存储器资源使用情况：

- 数据日志仅占用装载存储器。
- 组合起来的所有数据日志的大小受装载存储器可用资源的限制。仅可同时打开八个数据日志。可通过文件浏览器 (页 1065)标准 Web 页面管理这些数据日志。请参见此标准 Web 页面的描述，获取可同时维护的数据日志数量的相关指南。
- RECORDS 参数的最大可能数值是 UDint 数的限值 (4,294,967,295)。RECORD 参数的实际限值取决于单个记录的大小、其它数据日志的大小及装载存储器的可用资源。此外，Excel 对 Excel 工作表中允许的行数也有一定限制。

说明

在开始数据日志写入操作前，必须完成执行数据日志创建

- DataLogCreate 和 DataLogNewFile 日志文件创建操作可能持续多个程序扫描周期。创建日志文件所需的实际时间取决于记录结构和记录数。程序逻辑必须监视并捕捉到 DONE 位转换为 TRUE 状态后，才表示日志文件创建完成。如果用户程序在数据日志创建操作完成之前执行 DataLogWrite 指令，写操作将无法按要求写入新的数据日志记录。
- 在非常快的程序扫描运行的特定情况下，数据日志创建过程可能需要较长时间。如果这种长时间的创建过程过慢，应确保已激活“启用循环 OB 的最小循环时间”复选框，并将最小循环时间设置为 1 ms 或更大的值。更多信息，请参见组态循环时间和通信负载 (页 103)。

说明

DataLogNewFile 指令可复制现有数据日志的记录结构

如果要防止覆盖任何数据记录，则可在当前数据日志已存储最大记录数后，使用 DataLogNewFile 指令基于当前数据日志创建新数据日志。新数据记录将存储到新数据日志文件中。旧数据日志文件及记录数据仍保存在闪存中。

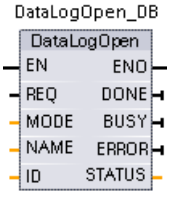
表格 9-202 ERROR 和 STATUS 的值

ERROR	STATUS (W#16#....)	说明
0	0000	无错误
0	7000	无 REQ 沿时调用：BUSY = 0，DONE = 0
0	7001	有 REQ 沿时首次调用（工作）：BUSY = 1，DONE = 0
0	7002	第 N 次调用（工作）：BUSY = 1，DONE = 0
1	8070	所有内部实例存储器都在使用。
1	807F	内部错误
1	8090	文件名无效
1	8091	名称参数不是字符串引用。
1	8093	已存在具有该名称的数据记录。使用其它名称，确保现有数据日志的 .csv 文件未打开，然后使用文件浏览器 (页 1065) Web 服务器页面删除现有数据日志。
1	8097	请求的文件长度超出文件系统最大值。
1	80B2	不在源 ID 范围内 注：删除一些现有数据日志或减少数据记录结构中列的数量可避免此错误。
1	80B3	可用装载存储器空间不足。
1	80B4	MC（存储卡）为写保护。

ERROR	STATUS (W#16#....)	说明
1	80C0	归档文件已锁定
1	80C1	打开的文件过多：最多只允许同时打开 8 个数据日志文件。
1	8253	记录计数无效
1	8353	格式选择无效
1	8453	时间戳选择无效
1	8B24	HEADER 区域分配无效：例如，指向本地存储器
1	8B51	HEADER 参数数据类型无效
1	8B52	HEADER 参数数据元素过多
1	8C24	DATA 区域分配无效：例如，指向本地存储器
1	8C51	DATA 参数数据类型无效
1	8C52	DATA 参数数据元素过多

DataLogOpen（打开数据日志）

表格 9-203　DataLogOpen 指令

LAD/FBD	SCL	说明
	"DataLogOpen_DB"(req:=_bool_in_, mode:=_uint_in_, done=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, name:=_string_inout_, ID:=_dword_inout_);	打开已有数据日志文件。必须先打开数据日志，才能向该日志写入 (页 520)新记录。可单独打开或关闭各个数据日志。最多可同时打开八个数据日志。 STEP 7 会在插入指令时自动创建关联的背景数据块。

[2] 在 SCL 示例中，“DataLogOpen_DB”是背景 DB 的名称。

表格 9-204 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	通过由低到高的（上升沿）信号启动操作。（默认值：False）
MODE	IN	UInt	工作模式： ● 0 - 附加到现有数据（默认值） ● 1 - 清除所有现有记录
NAME	IN	Variant	现有数据日志的名称：此变量仅支持 String 数据类型，且只可位于本地、DB 或 M 存储器。（默认值：''）
ID	输入/输出	DWord	数据日志的数字标识符。（默认值：0） **注：**该参数不支持符号名称访问。
DONE	OUT	Bool	上一请求已完成且没有出错后，DONE 位将保持为 TRUE 一个扫描周期时间。（默认值：False）
BUSY	OUT	Bool	● 0 - 无操作正在进行 ● 1 - 操作正在进行
ERROR	OUT	Bool	上一请求因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。STATUS 参数中的错误代码值仅在 ERROR = TRUE 的一个扫描周期内有效。
STATUS	OUT	Word	执行条件代码（默认值：0）

可提供已有数据日志的 NAME 或 ID（ID 参数作为输入）。如果同时提供这两个参数，但有效的 ID 与 NAME 数据日志不对应，则使用 ID，而忽略 NAME。

NAME 必须是 DataLogCreate 指令创建的数据日志的名称。如果只提供 NAME 且 NAME 指定一个有效数据日志，将返回对应的 ID（ID 参数作为输出）。

说明

数据日志文件的一般用法

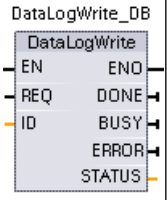
- 执行 DataLogCreate 和 DataLogNewFile 操作后会自动打开数据日志文件。
- PLC 执行 RUN-STOP 切换或 PLC 循环上电后会自动关闭数据日志文件。
- 必须打开了数据日志文件，才能执行新的 DataLogWrite 操作。
- 最多可同时打开八个数据日志文件。可能存在八个以上数据日志文件，但必须关闭一些数据日志文件，使打开的文件数不超过八个。

表格 9-205 ERROR 和 STATUS 的值

ERROR	STATUS (W#16#)	说明
0	0000	无错误
0	0002	警告：数据日志文件已通过该应用程序打开
0	7000	无 REQ 沿时调用：BUSY = 0，DONE = 0
0	7001	有 REQ 沿时首次调用（工作）：BUSY = 1，DONE = 0
0	7002	第 N 次调用（工作）：BUSY = 1，DONE = 0
1	8070	所有内部实例存储器都在使用。
1	8090	数据日志定义与现有数据日志文件不一致。
1	8091	名称参数不是字符串引用。
1	8092	数据日志不存在。
1	80C0	数据日志文件被锁定。
1	80C1	打开的文件过多：最多只允许同时打开 8 个数据日志文件。

DataLogWrite（写入数据日志）

表格 9-206 DataLogWrite 指令

LAD/FBD	SCL	说明
	"DataLogWrite_DB"(req:=_bool_in_, done=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, ID:=_dword_inout_);	将数据记录写入指定的数据日志。必须先打开 (页 518)已有目标数据日志，才能使用 DataLogWrite 指令对其执行写入操作。 STEP 7 会在插入指令时自动创建关联的背景数据块。

[2] 在 SCL 示例中，“DataLogWrite_DB”是背景 DB 的名称。

表格 9-207 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	通过由低到高的（上升沿）信号启动操作。（默认值：False）
ID	In/Out	DWord	数据日志数字标识符。仅用作 DataLogWrite 指令的输入。（默认值：0） **注：**该参数不支持符号名称访问。

参数和类型		数据类型	说明
DONE	OUT	Bool	上一请求已完成且没有出错后，DONE 位将保持为 TRUE 一个扫描周期时间。
BUSY	OUT	Bool	• 0 - 无操作正在进行 • 1 - 操作正在进行
ERROR	OUT	Bool	上一请求因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。STATUS 参数中的错误代码值仅在 ERROR = TRUE 的一个扫描周期内有效。
STATUS	OUT	Word	执行条件代码（默认值：0）

DataLogCreate 指令的 DATA 参数定义记录缓冲区的存储器地址和数据结构。控制程序必须使用当前运行系统过程值装载记录缓冲区，然后执行 DataLogWrite 指令将新记录数据从缓冲区移动到数据日志。

ID 参数用于标识数据日志和数据记录组态。DataLogCreate 指令将生成 ID 编号。

如果循环数据日志文件中存在空记录，则 DataLogWrite 指令将写入下一条可用的空记录。如果所有记录均不为空，则 DataLogWrite 指令将覆盖最早的记录。

注意

在开始数据日志写入操作前，必须完成数据日志创建操作

DataLogCreate 和 DataLogNewFile 日志文件创建操作可能持续多个程序扫描周期。创建日志文件所需的实际时间取决于记录结构和记录数。程序逻辑必须监视并捕捉到 DONE 位转换为 TRUE 状态后，才表示日志文件创建完成。如果在数据日志创建操作完成之前执行 DataLogWrite 指令，写操作将无法写入新的数据日志记录。

说明

数据日志对 CPU 内存储器的影响

每写一个数据日志均占用至少 2 KB 存储空间。如果程序频繁地写少量数据，则每一次写操作至少消耗 2 KB 内存。采用某个数据块 (DB) 存放这些小数据量数据项，然后，以较小频次将该数据块写入数据变量不失为一种更好的实现方法。

如果程序需要非常频繁地写大量数据变量条目，则应该考虑采用可以更换的 SD 存储卡。

注意

CPU 电源故障时数据日志数据丢失的可能性

如果未完成 DataLogWrite 操作时发生电源故障，则当前正向数据日志传送的数据记录可能会丢失。

表格 9-208 ERROR 和 STATUS 的值

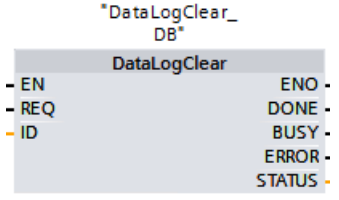
ERROR	STATUS (W#16#)	说明
0	0000	无错误
0	0001	表明数据日志已满：创建的各数据日志只能存储指定的最大记录数。如果写入最后一条记录后达到最大记录数，则下一写操作将覆盖最早的记录。
0	7000	无 REQ 沿时调用：BUSY = 0，DONE = 0
0	7001	有 REQ 沿时首次调用（工作）：BUSY = 1，DONE = 0
0	7002	第 N 次调用（工作）：BUSY = 1，DONE = 0
1	8070	所有内部实例存储器都在使用。
1	8092	数据日志不存在。
1	80B0	数据日志文件未打开（仅针对显式打开模式）。

DataLogClear（清空数据日志）

说明

表格 9-209 DataLogClear 指令

LAD/FBD	SCL	说明

	`"DataLogClear_DB"(`
	`REQ:=_bool_in_,`
	`DONE=>_bool_out_,`
	`BUSY=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`ID:=_dword_inout_);`
		指令“DataLogClear”可删除现有数据记录中的所有数据记录。该指令不会删除 CSV 文件的可选标题（请参见指令“DataLogCreate (页 514)”的 HEADER 参数说明）。
		可以使用参数 ID 选择要删除数据记录的数据记录。

“DataLogClear_DB”是背景数据块的名称。

要求

删除数据记录前，必须打开数据记录。使用 DataLogOpen (页 518) 指令打开数据日志。

参数

下表列出了“DataLogClear”指令的参数：

参数	声明	数据类型	存储区	说明
REQ	Input	BOOL	I、Q、M、L、D、T、C 或常量 （对于 S7-1500，T 和 C 只能在 LAD 和 FBD 中使用）	在上升沿执行指令。
ID	InOut	DWORD	I、Q、M、D、L	数字数据日志标识符
DONE	Output	BOOL	I、Q、M、D、L	指令已成功执行。
BUSY	Output	BOOL	I、Q、M、D、L	指令的执行尚未完成。
ERROR	Output	BOOL	I、Q、M、D、L	• 0：无错误。 • 1：指令执行期间出现错误。 详细信息将在 STATUS 参数中输出。
STATUS	Output	WORD	I、Q、M、D、L	状态参数 该参数设置仅维持一次调用所持续的时间。因此，要显示其状态，应将 STATUS 参数复制到可用数据区域。

有关有效数据类型的更多信息，请参见 “数据类型 (页 119)”。

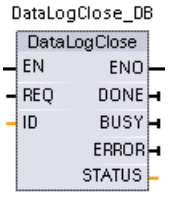
参数 STATUS

错误代码* (W#16#...)	说明
0000	无错误。
7000	未激活任何作业处理。
7001	启动作业处理。参数 BUSY = 1，DONE = 0
7002	中间调用（与 REQ 无关）：已激活指令；BUSY 的值为“1”。
8080	使用 ID 参数选择的数据记录文件无法使用“DataLogClear”指令处理。
8092	数据日志不存在。
80A2	文件系统返回写入错误。
80B0	数据记录未打开。

错误代码* (W#16#...)	说明
80B4	存储卡受到写保护。
* 在程序编辑器中，错误代码可显示为整数或十六进制值。有关切换显示格式的信息，请参见“另请参见”。	

DataLogClose（关闭数据日志）

表格 9-210 DataLogClose 指令

LAD/FBD	SCL	说明
	"DataLogClose_DB"(req:=_bool_in_, done=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, ID:=_dword_inout_);	关闭打开的数据日志文件。对已关闭的数据日志执行 DataLogWrite 操作将导致错误。再次执行 DataLogOpen 操作之前，禁止对此数据日志执行写操作。 切换到 STOP 模式时将关闭所有已打开的数据日志文件。 STEP 7 会在插入指令时自动创建关联的背景数据块。

[2] 在 SCL 示例中，“DataLogClose_DB”是背景 DB 的名称。

表格 9-211 参数的数据类型

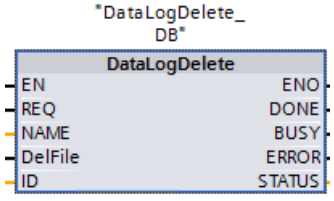
参数和类型		数据类型	说明
REQ	IN	Bool	通过由低到高的（上升沿）信号启动操作。（默认值：False）
ID	输入/输出	DWord	数据日志的数字标识符。仅用作 DataLogClose 指令的输入。（默认值：0） **注：**该参数不支持符号名称访问。
DONE	OUT	Bool	上一请求已完成且没有出错后，DONE 位将保持为 TRUE 一个扫描周期时间。
BUSY	OUT	Bool	• 0 - 无操作正在进行 • 1 - 操作正在进行
ERROR	OUT	Bool	上一请求因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。STATUS 参数中的错误代码值仅在 ERROR = TRUE 的一个扫描周期内有效。
STATUS	OUT	Word	执行条件代码（默认值：0）

表格 9-212 ERROR 和 STATUS 的值

ERROR	STATUS (W#16#)	说明
0	0000	无错误
0	0001	数据日志未打开
0	7000	无 REQ 沿时调用：BUSY = 0，DONE = 0
0	7001	有 REQ 沿时首次调用（工作）：BUSY = 1，DONE = 0
0	7002	第 N 次调用（工作）：BUSY = 1，DONE = 0
1	8092	数据日志不存在。

DataLogDelete（删除数据日志）

表格 9-213 DataLogDelete 指令

LAD/FBD	SCL	说明
	"DataLogDelete_DB"(REQ:=_bool_in_, NAME:=_variant_in_, DelFile:=_bool_in_, DONE=>_bool_out_, BUSY=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_, ID:=_dword_inout_);	使用“DataLogDelete”指令可删除数据日志文件。仅当通过指令“DataLogCreate”或“DataLogNewFile”创建数据记录的情况下才能删除该日志及其所含数据记录。

“DataLogDelete_DB”是背景数据块的名称。

参数

下表列出了“DataLogDelete”指令的参数：

参数	声明	数据类型	存储区	说明
REQ	Input	BOOL	I、Q、M、L、D、T、C 或常量 （对于 S7-1500，T 和 C 只能在 LAD 和 FBD 中使用）	在上升沿执行指令。
NAME	Input	VARIANT	L、D	数据记录的文件名
DELFILE	Input	BOOL	I、Q、M、D、L 或常量	• 0：将保留数据记录。 • 1：将删除数据记录。

参数	声明	数据类型	存储区	说明
ID	InOut	DWORD	I、Q、M、D、L	数字数据日志标识符
DONE	Output	BOOL	I、Q、M、D、L	指令已成功执行。
BUSY	Output	BOOL	I、Q、M、D、L	尚未完成数据记录的删除。
ERROR	Output	BOOL	I、Q、M、D、L	• 0：无错误。 • 1：指令执行期间出现错误。 详细信息将在 STATUS 参数中输出。
STATUS	Output	WORD	I、Q、M、D、L	状态参数 该参数设置仅维持一次调用所持续的时间。因此，要显示其状态，应将 STATUS 参数复制到可用数据区域。

有关有效数据类型的更多信息，请参见“数据类型 (页 119)”。

参数 NAME 和 ID

使用 NAME 和 ID 参数选择要删除的数据记录。首先，对 ID 参数求值。如果存在相关 ID 的数据记录，则将不会再判断 NAME 参数。如果 ID 参数中使用值“0”，则 NAME 参数中必须使用数据类型为 STRING 的值。

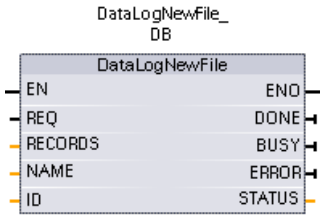
参数 RET_VAL

错误代码* (W#16#...)	说明
0	无错误。
7000	未激活任何作业处理。
7001	启动作业处理。参数 BUSY = 1，DONE = 0
7002	中间调用（与 REQ 无关）：已激活指令；BUSY 的值为“1”。
8091	NAME 参数使用的数据类型不是 STRING。
8092	数据日志不存在。
80A2	文件系统返回写入错误。
80B4	存储卡受到写保护。
* 在程序编辑器中，错误代码可显示为整数或十六进制值。有关切换显示格式的信息，请参见“另请参见”。	

DataLogNewFile（新文件中的数据日志）

表格 9-214　DataLogNewFile 指令

LAD/FBD	SCL	说明

	`"DataLogNewFile_DB"(`
	`req:=_bool_in_,`
	`records=:_udint_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`name=:_DataLog_out_,`
	`ID:=_dword_inout_);`
		允许程序根据现有数据日志文件创建新的数据日志文件。
		STEP 7 会在插入指令时自动创建关联的背景数据块。

[2]　在 SCL 示例中，“DataLogNewFile_DB”是背景 DB 的名称。

表格 9-215　参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	通过由低到高的（上升沿）信号启动操作。（默认值：False）
RECORDS	IN	UDInt	覆盖最旧条目前循环数据日志可存储的最大数据记录数。（默认值：1) 不包括标题记录。CPU 装载存储器的可用空间必须足够大，以确保成功创建数据日志。
NAME	IN	Variant	数据日志名称：用户可在此提供名称。此变量仅支持 String 数据类型，且只可位于本地、DB 或 M 存储器。（默认值：''） 该字符串引用还用作数据日志文件名。名称中的字符必须遵守 Windows 文件系统命名限制。字符 \ / : * ? " < > | 及空格字符均不允许使用。）
ID	输入/输出	DWord	数字数据日志标识符（默认值：0): ● 执行时，ID 输入标识有效数据日志。将从该数据日志复制新数据日志组态。 ● 执行后，ID 参数成为返回新建数据日志文件的 ID 的输出。 **注：**该参数不支持符号名称访问。
DONE	OUT	Bool	上一请求已完成且没有出错后，DONE 位将保持为 TRUE 一个扫描周期时间。
BUSY	OUT	Bool	● 0 - 无操作正在进行 ● 1 - 操作正在进行

参数和类型		数据类型	说明
ERROR	OUT	Bool	上一请求因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。STATUS 参数中的错误代码值仅在 ERROR = TRUE 的一个扫描周期内有效。
STATUS	OUT	Word	执行条件代码（默认值：0）

如果数据日志已满或被视为已完成，但您不想丢失数据日志中存储的任何数据，则可执行 DataLogNewFile 指令。可以根据此已写满的数据日志文件的结构创建一个新的空数据日志文件。将从原始数据日志复制标题记录以及原始数据日志属性（DATA 记录缓冲区、数据格式和时间戳设置）。隐式关闭原始数据日志文件并隐式打开新数据日志文件。

DataLogWrite 参数触发器：程序必须监视每个 DataLogWrite 操作的 ERROR 和 STATUS 参数。在写入最终记录且数据日志写满时，DataLogWrite ERROR 位 = 1 且 DataLogWrite STATUS 字 = 1。此 ERROR 值和 STATUS 值只可扫描一次，所以监视逻辑必须使用 ERROR = 1 作为时间门来捕获 STATUS 值，然后测试 STATUS = 1 （数据日志写满）是否成立。

DataLogNewFile 操作：当程序逻辑获得数据日志已满信号时，此状态用于激活 DataLogNewFile 操作。必须使用现有（通常已满）的已打开数据日志的 ID 执行 DataLogNewFile，但要使用新的唯一 NAME 参数。DataLogNewFile 操作完成后，将返回新的数据日志 ID 值（作为输出参数），该值与新的数据日志名称相对应。新数据日志文件隐式打开并可存储新记录。针对新数据日志文件的新的 DataLogWrite 操作必须使用 DataLogNewFile 操作返回的 ID 值。

注意
在开始数据日志写入操作前，必须完成数据日志创建操作 DataLogCreate 和 DataLogNewFile 日志文件创建操作可能持续多个程序扫描周期。创建日志文件所需的实际时间取决于记录结构和记录数。程序逻辑必须监视并捕捉到 DONE 位转换为 TRUE 状态后，才表示日志文件创建完成。如果在数据日志创建操作完成之前执行 DataLogWrite 指令，写操作将无法按要求写入新的数据日志记录。

表格 9-216 ERROR 和 STATUS 的值

ERROR	STATUS (W#16#)	说明
0	0000	无错误
0	7000	无 REQ 沿时调用：BUSY = 0，DONE = 0
0	7001	有 REQ 沿时首次调用（工作）：BUSY = 1，DONE = 0
0	7002	第 N 次调用（工作）：BUSY = 1，DONE = 0
1	8070	所有内部实例存储器都在使用。

ERROR	STATUS (W#16#)	说明
1	8090	文件名无效
1	8091	名称参数不是字符串引用。
1	8092	数据日志不存在。
1	8093	数据日志已存在。
1	8097	请求的文件长度超出文件系统最大值。
1	80B2	不在源 ID 范围内 注：删除一些现有数据日志来为新数据日记创建资源。
1	80B3	可用装载存储器空间不足。
1	80B4	MC 受写保护。
1	80C1	打开的文件过多。

9.9.2.3 使用数据日志

数据日志文件以逗号分隔值格式 (*.csv) 存储在永久性闪存中。 可以使用 PLC Web 服务器功能或通过取出 PLC 存储卡并将其插入标准 PC 读卡器中来查看数据日志。

使用 PLC Web 服务器功能查看数据日志

如果 PLC PROFINET 端口和 PC 连接到网络，则可使用 PC Web 浏览器（如 Microsoft Internet Explorer 或 Mozilla Firefox）访问内置 PLC Web 服务器。 运行 PLC Web 服务器时，PLC 可以处于运行模式或停止模式。 如果 PLC 处于运行模式，则当 PLC Web 服务器通过网络传送日志数据时，控制程序会继续执行。

Web 服务器访问：

1. 在目标 CPU 的设备配置中启用 Web 服务器 (页 1028)。
2. 通过 PROFINET 网络将 PC 连接到 PLC (页 1030)。
3. 通过内置 Web 服务器访问 CPU (页 1036)。
4. 使用“文件浏览器”标准 Web 页面 (页 1065)下载、编辑和删除数据日志文件。
5. 使用类似 Microsoft Excel 等电子表格应用程序打开 .csv 文件。

说明

数据日志管理

在文件系统中，可保留不超过 1000 个数据日志。 超过此数目时，Web 服务器就没有用于显示数据日志的足够空间。

如果您发现“文件浏览器”Web 页面无法显示数据日志，则必须将 CPU 置于 STOP 模式，以便显示并删除数据日志。

管理您的数据日志以确保仅保留需要维护的数目，且不会超过 1000 个数据日志。

查看 PLC 存储卡中的数据日志

如果 S7-1200 CPU 中插入了“程序”型 S7-1200 存储卡，则可以取出该存储卡，然后将该卡插入 PC 或 PG 上的标准 SD（安全数码卡）卡槽或 MMC （多媒体卡）卡槽中。 取出存储卡后，PLC 将处于停止模式，因此不执行控制程序。

在 Windows 资源管理器中导航至存储卡中的 \DataLog 目录。 所有 *.csv 数据日志文件都位于该目录下。

复制数据日志文件，然后将副本放到 PC 的本地驱动器中。 接着，可以使用 Excel 打开 *.csv 文件的本地副本，而不是存储在存储卡中的原始文件。

注意

可以借助 PC 读卡器复制 S7-1200 存储卡中的数据日志文件但不要修改或删除这些文件

推荐使用标准 Web 服务器“文件浏览器”页面工具来查看、下载（复制）和删除数据日志文件。

如果直接通过 Windows 资源管理器浏览存储卡文件系统，则可能意外删除/修改数据日志或其它系统文件，这样一来可能会损坏文件或使存储卡无法使用。

注意

数据日志对存储卡的影响

为确保系统的整体性能和稳定性，请将数据日志的记录频率限定为不得超过每 200 ms 一次。

9.9.2.4 数据日志文件大小的限制

数据日志文件与程序、程序数据、组态数据、用户定义的 Web 页面和 PLC 系统数据共享 PLC 装载存储器空间。使用内部装载存储器的大型程序需要的装载存储器空间相应更大。数据日志文件的自由空间可能不足。在这种情况下，可以使用“程序卡” (页 139)来增加装载存储器的容量。S7-1200 CPU 既可以使用内部装载存储器也可以使用外部装载存储器，但不能同时使用。

数据日志文件的最大大小规则

单个数据日志文件的最大大小不可超过可用装载存储器大小或 500 MB（二者之中的较小值）。在这种情况下，500 MB 大小表示兆字节的十进制定义，因此数据日志文件的最大大小为 500,000,000 字节或 500 x 1000^2 字节。

表格 9-217 装载存储器大小

数据区	CPU 1211C	CPU 1212C	CPU 1214C	CPU 1215C、CPU 1217C	数据存储
内部装载存储器 闪存	1 MB	2 MB	4 MB	4 MB	用户程序和程序数据、组态数据、数据日志、用户定义的 Web 页面以及 PLC 系统数据
外部装载存储器 可选“程序卡”闪存	4 MB、12 MB、24 MB、256 MB、2 GB 或 32 GB，取决于 SD 卡大小				

确定装载存储器可用空间

正常运行期间操作系统会使用和释放存储空间，因此装载存储器可用空间量会随之变化。使用下列步骤查看装载存储器大小。

1. 建立 STEP 7 和目标 S7-1200 PLC 之间的在线连接。
2. 下载用于控制您的数据日志操作的程序。
3. 根据需要创建任何可选的用户定义 Web 页面。用于访问数据日志的标准 Web 页面存储在 PLC 固件中，不占用装载存储器的空间。
4. 请使用“在线工具和诊断工具” (页 1383)(Online and diagnostic tools) 或“网页服务器诊断页” (页 1042)(Web server Diagnostics page) 来查看总装载存储器大小和自由空间。

计算数据日志文件的大小（所有数据记录）

在数据日志文件创建时，CPU 会分配最大存储器大小。除了所有数据记录需要的大小，您还必须包括用于数据日志标头（如果使用）、时间戳标头（如果使用）和记录索引标头的存储空间以及用于存储器分配的最小块大小。

使用下列公式确定数据日志文件的大小，确保不违反最大大小规则。

数据日志数据字节 = （（一个记录中的数据字节 + 时间戳字节 + 12 字节）* 记录数）

标头

数据日志标头字节 = 标头字符字节 + 2 字节

标头字符字节

- 无数据标头和无时间戳 = 7 字节
- 无数据标头，有时间戳（有时间戳标头） = 21 字节
- 有数据标头，无时间戳 = 所有列标题文本包括分隔符逗号的字符字节数
- 有数据标头和时间戳（有时间戳标头） = 所有列标题文本包括分隔符逗号的字符字节数 + 21 字节

数据

数据日志数据字节 =（（一个记录中的数据字节 + 时间戳字节 + 12 字节）* 记录数）

一个数据记录中的数据字节

DataLogCreate DATA 参数指向一个结构，该结构用于为一条数据日志记录分配数据字段数和各数据字段的数据类型。

将给出的数据类型的出现次数乘以该数据类型所需的字节数。对一条记录中的每个数据类型重复该过程，并对所有数据类型求和得到一条记录中所有数据元素的总计字节。

各数据元素的大小

日志数据以 CSV（逗号分隔值）文件格式存储为若干字符字节。下表给出了存储各数据元素所需的字节数。

数据类型	字节数（包括数据加上一个逗号字节）
Bool	2
Byte	5
Word	7
DWord	12
Char	4

String	示例 1：MyString String[10] 最大字符串大小分配为 10 个字符。 文本字符 + 自动填充的空格字符 = 10 个字节 成对的双引号 + 逗号字符 = 3 个字节 10 + 3 = 13 个字节（总和） 示例 2：Mystring2 String 如果未使用方括号指定大小，则默认分配 254 个字节。 文本字符 + 自动填充的空格字符 = 254 个字节 成对的双引号 + 逗号字符 = 3 个字节 254 + 3 = 257 个字节（总和）
USInt	5
UInt	7
UDInt	12
SInt	5
Int	7
DInt	12
Real	16
LReal	25
Time	15
DTL	24

数据日志文件中的记录数

DataLogCreate 指令的 RECORDS 参数用于设置数据日志文件中可存储的最大记录数。

一个数据记录中的时间戳字节

- 无时间戳 = 0 字节
- 有时间戳 = 20 字节

9.9.2.5 数据日志示例程序

该数据日志示例程序未显示从动态过程获取采样值必需的所有程序逻辑，但显示了数据日志指令的重要操作。所使用的日志文件的结构和数目取决于过程控制要求。

说明

数据日志文件的一般用法

- 执行 DataLogCreate 和 DataLogNew 文件操作后会自动打开数据日志文件。
- PLC 执行 RUN-STOP 切换或 PLC 循环上电后会自动关闭数据日志文件。
- 必须打开了数据日志文件，才能执行 DataLogWrite 操作。
- 最多可同时打开八个数据日志文件。可能存在八个以上数据日志文件，但必须关闭一些数据日志文件，使打开的文件数不超过八个。

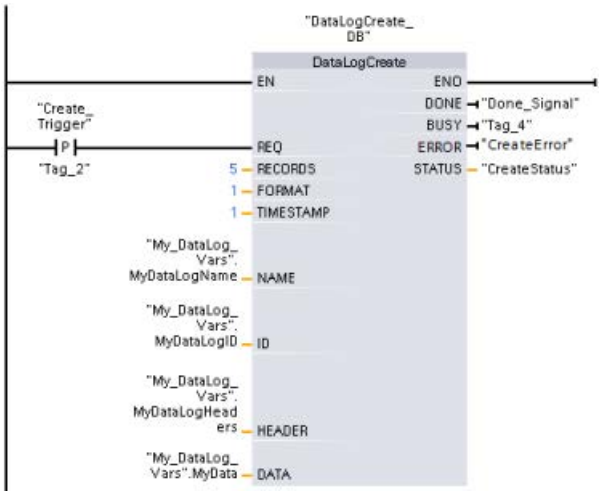
示例数据日志程序

在数据块中创建示例数据日志名称、标题文本和 MyData 结构。三个 MyData 变量临时存储新的采样值。通过执行 DataLogWrite 指令将这些 DB 位置的过程采样值传送到数据日志文件。

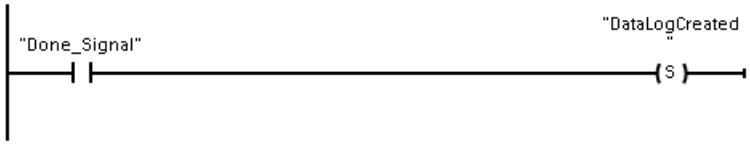
My_Datalog_Vars

	名称	数据类型	启动值
1	Static		
2	MyNEWDataLogName	String	'MyNEWDataLog'
3	MyDataLogName	String	'MyDataLog'
4	MyDataLogID	DWord	0
5	MyDataLogHeaders	String	'Count,Temperature,Pressure'
6	MyData	Struct	
7	MyCount	Int	0
8	MyTemperature	Real	0.0
9	MyPressure	Real	0.0

程序段 1 REQ 上升沿启动数据日志创建过程。



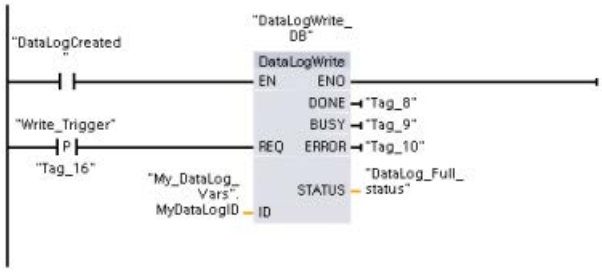
程序段 2 捕获 DataLogCreate 的 DONE 输出，考虑到该指令仅在一个扫描周期内有效。



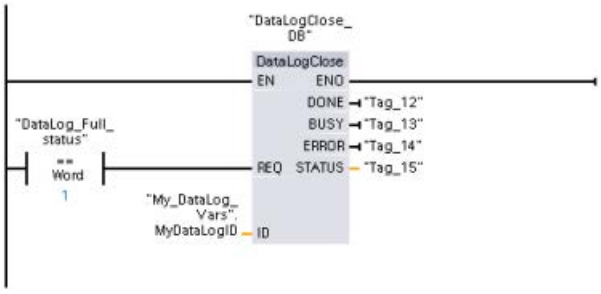
程序段 3 如果存在错误，则保存状态输出

程序段 4 上升沿信号触发何时将新过程值存储在 MyData 结构中。

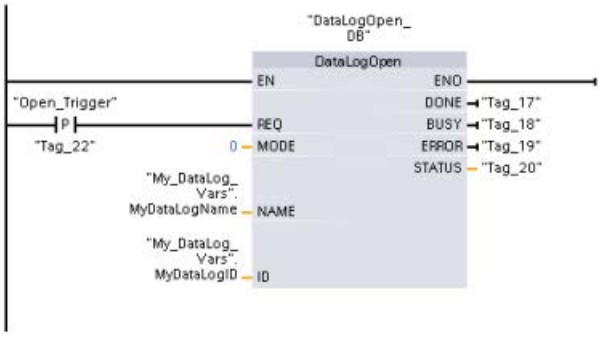
程序段 5 EN 输入状态取决于何时完成 DataLogCreate 操作。创建操作将跨越多个扫描周期，并且必须在执行写入操作之前完成。REQ 输入的上升沿信号是触发已启用写入操作的事件。



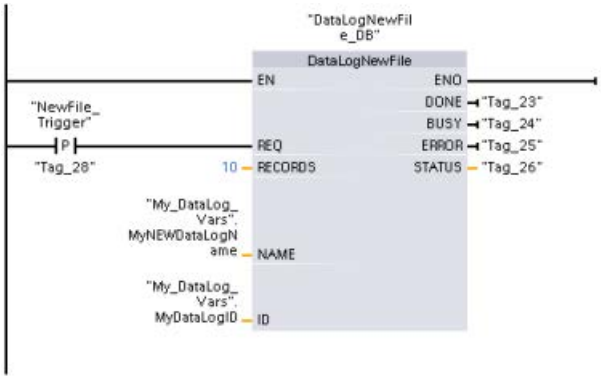
程序段 6 在写入最后一条记录后立即关闭数据日志。执行写入最后一条记录的 DataLogWrite 操作后，将通过 DataLogWrite STATUS 输出为 1 来指示日志文件已写满状态。



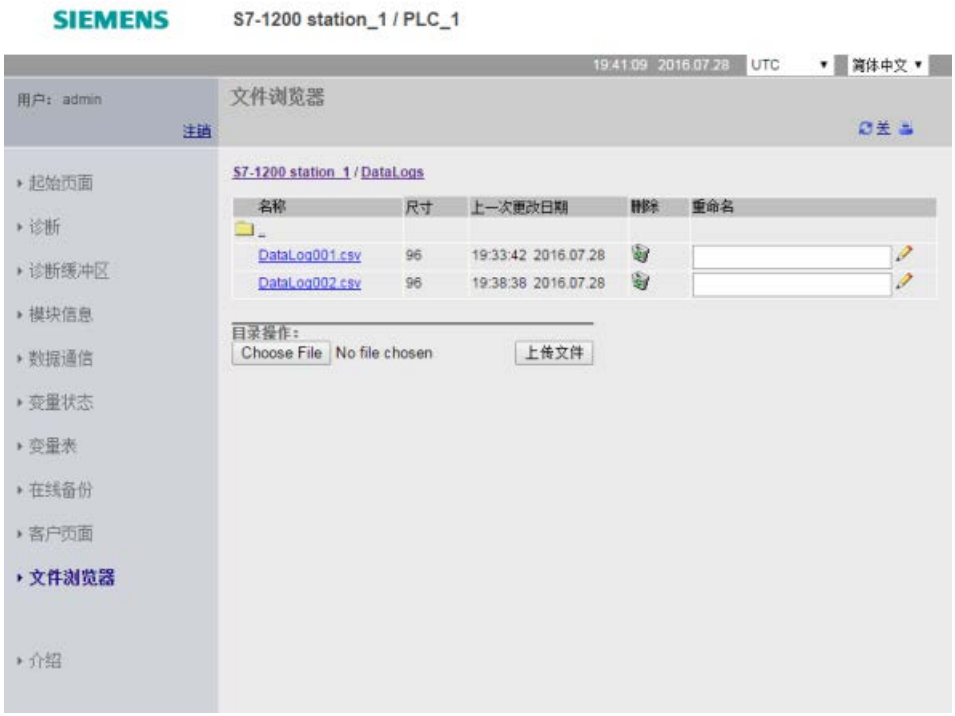
程序段 7 DataLogOpen REQ 输入的上升信号沿会模拟用户按下 HMI 上的按钮打开数据日志文件的操作。如果打开所有记录都填满了过程数据的数据日志文件，则下一个 DataLogWrite 操作会覆盖最早的记录。您也许想保存之前的数据日志并创建新数据日志，如程序段 7 中所示。



程序段 8 ID 参数是 IN/OUT 类型。首先应现有数据日志的 ID 值，以便能复制其结构。完成 DataLogNewFile 操作后，会将新数据记录的唯一 ID 新值写回 ID 参考位置。未显示所需的 DONE 位 = TRUE 捕获，有关 DONE 位逻辑的示例，请参见程序段 1、2 和 4。



通过 S7-1200 CPU Web 服务器看到的由示例程序创建的数据日志文件



① 如果未以修改权限登录，则“删除”(Delete) 选项不可用。

② 如果未以修改权限登录，则“重命名”(Rename) 选项不可用。

表格 9-218 用 Excel 查看的已下载 .csv 文件示例

在最多五条记录的文件中写入了两条记录	

	A	B	C	D	E	F
1	Record	Date	UTC Time	Count	Temperature	Pressure
2	1	9/29/2010	21:01:46	5	5.00E+00	5.00E+00
3	2	9/29/2010	21:01:47	5	5.00E+00	5.00E+00
4						
5						

最多五条记录的数据日志文件中有五条记录	

	A	B	C	D	E	F
1	Record	Date	UTC Time	Count	Temperature	Pressure
2	1	9/30/2010	20:26:56	1	9.86E+01	3.52E+01
3	2	9/30/2010	20:28:43	2	1.00E+02	3.73E+01
4	3	9/30/2010	20:29:03	3	9.99E+01	3.68E+01
5	4	9/30/2010	20:29:21	4	9.95E+01	3.64E+01
6	5	9/30/2010	20:30:19	5	9.92E+01	3.74E+01
7						

将又一条记录写入上述已写满的文件后，第六次写入操作会用第六条数据覆盖最早的第一条记录。另一次写入操作会用第七条记录覆盖第二条记录，依此类推。	

	A	B	C	D	E	F
1	Record	Date	UTC Time	Count	Temperature	Pressure
2	6	9/30/2010	20:32:03	6	9.86E+01	3.58E+01
3	2	9/30/2010	20:28:43	2	1.00E+02	3.73E+01
4	3	9/30/2010	20:29:03	3	9.99E+01	3.68E+01
5	4	9/30/2010	20:29:21	4	9.95E+01	3.64E+01
6	5	9/30/2010	20:30:19	5	9.92E+01	3.74E+01
7						

说明

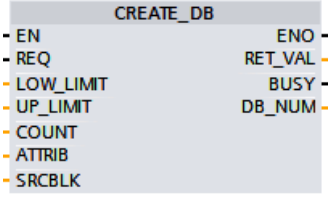
数据日志不再采用 //END 标志对未满的数据日志文件末尾进行标记。在 S7-1200 CPU V4.1 之前的版本中，未满的数据日志都包含一个 //END 标志。

9.10 数据块控制

9.10.1 CREATE_DB（创建数据块）

表格 9-219 CREATE_DB 指令

LAD/FBD	SCL	说明

	`ret_val := CREATE_DB(`
	`REQ:=_bool_in_,`
	`LOW_LIMIT:=_uint_in_,`
	`UP_LIMIT:=_uint_in_,`
	`COUNT:=_udint_in_,`
	`ATTRIB:=_byte_in_,`
	`BUSY=>_bool_out_,`
	`DB_NUM=>_uint_out_);`
		使用指令“CREATE_DB”在装载存储器和/或工作存储器中创建新的数据块。
		指令“CREATE_DB”不会更改用户程序的校验和。
		仅在工作存储器中生成的数据模块具有如下属性：
		● 在存储器复位或电源断开/接通后此数据块不再存在。
		● 当下载时或当从停止模式切换到运行模式时，其内容保持不变。

数据块编号

从参数 LOW_LIMIT（下限）和 UP_LIMIT（上限）所定义的范围中分配一个编号给所创建的数据块。“CREATE_DB”可将指定范围中的最小编号分配给数据块。不能分配用户程序中已有数据块的编号。

如果要创建具有特定编号的数据块，请针对指定范围的上限和下限输入同一编号。如果工作存储器和/或装载存储器中已存在具有相同编号的数据块，或者该数据块作为复制的数据块存在，则将中断此指令，并在 RET_VAL 参数生成错误消息。

数据块的起始值

SRCBLK 参数用来定义将创建数据块的起始值。SRCBLK 参数是指向数据块或数据块区域的指针，在该数据块或数据块区域应用起始值。SRCBLK 参数指向的数据块必须已通过标准访问权限生成（“优化块访问”属性已禁用）。

- 如果 SRCBLK 参数指定的区域大于生成的数据块，则直至所生成数据块长度的所有值将应用为起始值。
- 如果通过 SRCBLK 参数指定的区域小于生成的数据块，则剩余值将以“0”填充。

为了确保数据一致性，正在执行“CREATE_DB”时（这表明只要参数 BUSY = TRUE），不得更改此数据区域。

功能描述

“CREATE_DB”为异步执行指令，可以跨多个调用执行。调用“CREATE_DB”时，REQ = 1 可启动该作业。

输出参数 RET_VAL 和 BUSY 用于指示作业状态。

另请参见 “DELETE_DB（删除数据块） (页 549)”

参数

下表列出了指令“CREATE_DB”的参数：

参数	声明	数据类型	存储区	说明
REQ	Input	BOOL	I、Q、M、D、L 或常量	电平触发控制参数“request to activate” REQ = 1：请求创建数据块。
LOW_LIMIT	Input	UINT	I、Q、M、D、L 或常量	数据块编号分配范围下限可能的最小编号为 60000。
UP_LIMIT	Input	UINT	I、Q、M、D、L 或常量	“CREATE_DB”用于向数据块分配编号的区域的上限（可能的最大数据块编号：60999)
COUNT	Input	UDINT	I、Q、M、D、L 或常量	计数值指定需要为所生成数据块预留的字节数。该字节数必须为偶数。最大长度为 65534 个字节。

<table>
<tr><th>参数</th><th>声明</th><th>数据类型</th><th>存储区</th><th>说明</th></tr>
<tr><td rowspan="6">ATTRIB</td><td rowspan="6">Input</td><td rowspan="6">BYTE</td><td rowspan="6">I、Q、M、D、L 或常量</td><td>使用 ATTRIB 参数中字节的前 4 位定义数据块的属性 *：</td></tr>
<tr><td>• 第 0 位 = 0：未设置属性“仅存储在装载内存中”(Only store in load memory)。
• 位 0 = 1：已设置属性“仅存储在装载内存中”(Only store in load memory)。使用此设置时，数据块在工作存储器中不占用空间，并且不包括在程序中。也不能使用位命令访问数据块。当位 0 = 1 时，与位 2 的选择不相关。
要保证与 STEP 7 V5.x 兼容，位 0 和位 3 必须综合考虑（见下面）。</td></tr>
<tr><td>• 位 1 = 0：未设置属性“在设备中写保护数据块”(Data block write-protected in the device)。
• 位 1 = 1：已设置属性“在设备中写保护数据块”(Data block write-protected in the device)。</td></tr>
<tr><td>• 位 2 = 0：数据块为保持型数据块（仅适用于装载存储器以及工作存储器中生成的数据块）。如果至少一个值设置为保持型，数据块将被视为保持型。
• 位 2 = 1：数据块不是保持型数据块
保持性数据块不支持仅存储在装载存储器以及工作存储器中的数据块。如果使用“保持性且仅装载存储器”(retentive and only load memory) 或“保持性且仅工作存储器”(retentive and only work memory) 两个组合中的一个调用“CREATE_DB”指令，则要生成的数据块将不会标记为具有保持性。</td></tr>
<tr><td>• 位 3 = 0：在装载存储器或工作存储器中创建数据库（使用 0 位选择，请参见上文）
• 位 3 = 1：在装载存储器和工作存储器中创建数据库（位 0 不相关）
要保证与 STEP 7 V5.x 兼容，位 0 和位 3 必须组合使用。当位 3 = 1 时，位 0 不受影响。
<table>
<tr><th>位 0</th><th>位 3</th><th>数据块生成</th></tr>
<tr><td>0</td><td>0</td><td>仅在工作存储器中</td></tr>
<tr><td>1</td><td>0</td><td>仅在装载存储器中</td></tr>
<tr><td>不相关</td><td>1</td><td>工作存储器和装载存储器</td></tr>
</table></td></tr>
<tr><td>• 位 4 = 0 - 未指定起始值（将忽略 SRCBLK 参数中的输入值）。</td></tr>
</table>

参数	声明	数据类型	存储区	说明
				● 位 4 = 1 - 指定起始值（与 SRCBLK 参数指向的数据块一致的值）。
SRCBLK	Input	VARIANT	D	指向数据块的指针，该数据块的值将用于初始化将要生成的数据块。
RET_VAL	Return	INT	I、Q、M、D、L	错误信息
BUSY	Output	BOOL	I、Q、M、D、L	BUSY = 1：该过程尚未完成。
DB_NUM	Output	DB_DYN (UINT)	I、Q、M、D、L	已创建数据块的编号。
* 此处选择的属性对应于数据块的属性中的特性。				

有关有效数据类型的更多信息，请参见“数据类型 (页 119)”。

参数 RET_VAL

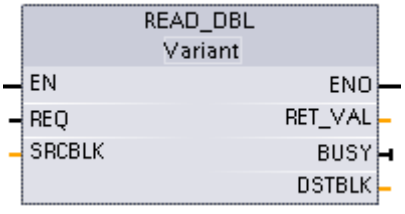
错误代码* (W#16#...)	说明
0000	无错误
0081	目标区域大于源区域。 将源区域完全写入目标区域。而目标区域的其余字节则保持不变。
7000	首次调用时，REQ = 0：未激活数据传送；BUSY 的值为“0”。
7001	首次调用时，REQ = 1：已触发数据传送；BUSY 的值为“1”。
7002	中间调用（与 REQ 无关）：已激活数据传送；BUSY 的值为“1”。
8081	源区域大于目标区域。 将写入整个目标区域，并忽略源区域中剩余的字节。
8092	“创建数据块”功能当前不可用，原因是： ● “压缩用户存储器”功能当前处于活动状态。 ● 已达到 CPU 上的最大数据块数目。
8093	没有为参数 SRCBLK 指定数据块，或指定的数据块不在工作存储器中。
8094	为 ATTRIB 参数指定的值无效。
80A1	数据块编号错误： ● 编号为“0” ● 编号超出 CPU 数据块数量的特定上限。 ● 下限 > 上限

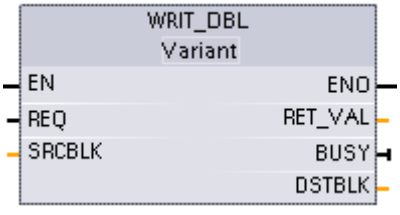
错误代码* (W#16#...)	说明
80A2	数据块长度错误： ● 长度为“0” ● 长度值为奇数 ● 长度大于 CPU 的允许值
80A3	SRCBLK 参数中的数据块不是通过标准访问权限创建的。
80B1	没有可用的数据块编号。
80B2	工作存储器空间不足。
80B4	存储卡受到写保护。
80BB	装载存储器空间不足。
80C3	已达到“CREATE_DB”指令可同时激活的最大数目。
常见错误信息	另请参见 “扩展指令的常见错误代码 (页 557)”
* 在程序编辑器中，错误代码可显示为整数或十六进制值。	

9.10.2 READ_DBL 和 WRIT_DBL（读取/写入装载存储器中的数据块）指令

表格 9-220 READ_DBL 和 WRIT_DBL 指令

LAD/FBD	SCL	说明

	`READ_DBL(`
	`req:=_bool_in_,`
	`srcblk:=_variant_in_,`
	`busy=>_bool_out_,`
	`dstblk=>_variant_out_);`
		将 DB 的全部或部分起始值从装载存储器复制到工作存储器的目标 DB 中。
		在复制期间，装载存储器的内容不变。

	`WRIT_DBL(`
	`req:=_bool_in_,`
	`srcblk:=_variant_in_,`
	`busy=>_bool_out_,`
	`dstblk=>_variant_out_);`
		将 DB 全部当前值或部分值从工作存储器复制到装载存储器的目标 DB 中。
		在复制期间，工作存储器的内容不变。

表格 9-221 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	BOOL	如果 BUSY = 0，则高电平信号会启动操作。
SRCBLK	IN	VARIANT	READ_DBL：指向装载存储器中源数据块的指针 WRIT_DBL：指向工作存储器中源数据块的指针
RET_VAL	OUT	INT	执行条件代码
BUSY	OUT	BOOL	BUSY = 1 表示读取/写入过程尚未完成。
DSTBLK	OUT	VARIANT	READ_DBL：指向工作存储器中目标数据块的指针 WRIT_DBL：指向装载存储器中目标数据块的指针

通常，DB 存储在装载存储器（闪存）和工作存储器 (RAM) 中。起始值（初始值）始终存储在装载存储器中，当前值始终存储在工作存储器中。READ_DBL 可用于将一组起始值从装载存储器复制到工作存储器中程序引用的 DB 的当前值。可使用 WRIT_DBL 将存储在内部装载存储器或存储卡中的起始值更新为工作存储器中的当前值。

说明

WRIT_DBL 和 READ_DBL 指令对闪存的影响

WRIT_DBL 指令会在闪存（内部装载存储器或存储卡）内执行写入操作。为了避免影响闪存的使用寿命，可以采用 WRIT_DBL 指令进行更新，例如，记录对某个生产工艺的更改。出于同样的考虑，请避免频繁地调用读操作指令 READ_DBL。

在 STEP 7 程序中，调用 READ_DBL 和 WRIT_DBL 指令前，必须为这些指令创建数据块。如果源数据块被创建成“标准”类型，则目标数据块也必须为“标准”类型。如果源数据块被创建成“优化”类型，则目标数据块也必须为“优化”类型。

如果 DB 为标准 DB，则可指定一个变量名称或 P# 值。P# 值允许指定和复制指定大小（字节、字或双字）的任意数量的元素。因此，可复制 DB 的全部或部分内容。如果 DB 是优化 DB，则只能指定一个变量名称；无法使用 P# 操作符。如果为标准或优化数据块（或者其它工作存储器类型）指定变量名称，则指令会复制此变量引用的数据。可以是用户定义类型、数组或基本元素。如果 DB 是标准 DB 而不是优化 DB，则这些指令只能使用数据类型结构。

如果它是优化存储器中的结构，则必须使用用户定义类型 (UDT)。仅用户定义类型可确保源结构和目标结构的“数据类型”完全相同。

说明

在“优化”DB 中使用结构（数据类型 Struct）

在“优化”DB 中使用 Struct 数据类型时，首先必须为 Struct 创建一个用户自定义数据类型 (UDT)。然后通过 UDT 组态源 DB 和目标 DB。UDT 确保针对两个 DB Struct 内的数据类型保持一致。

对于“标准”DB,使用 Struct 时无需创建 UDT。

READ_DBL 和 WRIT_DBL 相对于循环程序扫描异步执行。处理期间需要多次调用 READ_DBL 和 WRIT_DBL 指令。通过 REQ = 1 来调用指令启动 DB 传输作业，然后监视 BUSY 和 RET_VAL 输出以确定数据传输的完成时间以及是否正确。

说明

WRIT_DBL 和 READ_DBL 指令对通信负荷的影响

WRIT_DBL 或 READ_DBL 指令持续启用时，可能会消耗大量通信资源，使 STEP 7 无法与 CPU 进行通信。因此，对于 REQ 参数，请使用上升沿输入 (页 219)，而不使用常开或常闭型输入 (页 213)。后者在多次扫描期间会一直保持接通状态（即，信号为高电平）。

为确保数据的一致性，请勿在 READ_DBL 处理过程中修改目标区域或在 WRIT_DBL 处理过程中修改来源区域（即，不要在 BUSY 参数为 TRUE 时修改）。

SRCBLK 和 DSTBLK 参数限制：

- 数据块必须先创建，然后才可引用。
- BOOL 类型的 VARIANT 指针长度必须可被 8 整除。
- 源指针和目标指针中 STRING 类型的 VARIANT 指针的长度必须相同。

配方和机器设置信息

可使用 READ_DBL 和 WRIT_DBL 指令来管理配方或机器设置信息。虽然可以限制写入的次数以防止闪存损耗，但以上方法实际上是另一种归档那些值不经常更改的保持性数据的方法。这样即可在提供给常规掉电保持性数据的容量的基础上有效增加保持性存储器的容量，至少可满足不经常更改值的需要。可使用 WRIT_DBL 指令将配方信息或机器设置信息从工作存储器保存到装载存储器，并使用 READ_DBL 指令将这些信息从装载存储器提取到工作存储器。

表格 9-222 条件代码

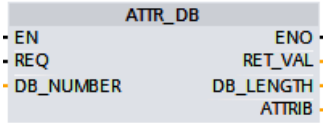
RET_VAL (W#16#...)	说明
0000	无错误
0081	警告：源区域小于目标区域。源数据已完全复制，目标区域中的额外字节未更改。
7000	REQ = 0 时调用：BUSY = 0
7001	REQ = 1 时首次调用（工作）：BUSY = 1
7002	第 N 次调用（工作）：BUSY = 1
8051	数据块类型错误
8081	源区域大于目标区域。目标区域已满，源数据中的剩余字节被忽略。
8251	源数据块类型错误
82B1	缺少源数据块
82C0	源 DB 正在被其它语句或通信功能编辑。
8551	目标数据块类型错误
85B1	缺少目标数据块
85C0	目标 DB 正在被其它语句或通信功能编辑。
80C3	当前已有超过 50 个的 READ_DBL 或 WRIT_DBL 语句排队等候执行。

另请参见配方 (页 501)

9.10.3 ATTR_DB（读取数据块属性）

表格 9-223 ATTR_DB 指令

LAD/FBD	SCL	说明

	`ret_val := ATTR_DB(`
	`REQ:=_bool_in_,`
	`DB_NUMBER:=_uint_in_,`
	`DB_LENGTH=>_udint_out_,`
	`ATTRIB=>_byte_out_);`
		可使用指令“ATTR_DB”获取有关 CPU 的工作存储器中某个数据块 (DB) 的信息。该指令可决定所选 DB 的 ATTRIB 参数中的属性集。
		对于优化访问类型的数据块和仅位于装载存储器中的数据块，其长度无法读取。此时，参数 DB_LENGH 的值为“0”。
		请勿将 ATTR_DB 应用于具有优化访问且激活了预留存储空间的数据块。
		请勿通过“ATTR_DB”指令读取运动控制的数据块。因此，将输出错误代码 80B2。

参数

下表列出了“ATTR_DB”指令的参数：

参数	声明	数据类型	存储区	说明
REQ	Input	BOOL	I、Q、M、D、L 或常量	REQ = 1：块属性读取请求
DB_NUMBER	Input	DB_ANY	I、Q、M、D、L 或常量	要测试的 DB 的编号
RET_VAL	Output	INT	I、Q、M、D、L	错误信息
DB_LENGTH	Output	UDINT	I、Q、M、D、L	• 所选数据块中包含的数据字节数。 • “0”表示优化访问的数据块和仅位于装载存储器中的数据块。

参数	声明	数据类型	存储区	说明
ATTRIB	Output	BYTE	I、Q、M、D、L	DB 属性：
				● 第 0* 位 = 0：未设置属性“仅存储在装载内存中”(Only store in load memory)。 ● 第 0 位* = 1：已设置属性“仅存储在装载内存中”(Only store in load memory)。
				● 位 1 = 0：未设置属性“在设备中写保护数据块”(Data block write-protected in the device)。 ● 位 1 = 1：已设置属性“在设备中写保护数据块”(Data block write-protected in the device)。
				如果第 0 位 = 1，则第 2 位不受影响，并且值为 1。 ● 位 2 = 0：保持性 - 如果至少一个值设置为保持性，则数据块将被视为保持性。 ● 位 2 = 1：非保持性 - 整个 DB 都不是保持性。
				● 位 3 * = 0：该 DB 在装载存储器（第 0 位 = 1）或工作存储器（第 0 位 = 0）中。 ● 位 3 * = 1：装载存储器和工作存储器中均会生成该 DB
* 将在指令“CREATE_DB（创建数据块） (页 539)”的参数中说明位 0 和位 3 之间的关系。				

有关有效数据类型的更多信息，请参见“数据类型 (页 119)”。

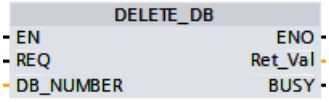
参数 RET_VAL

错误代码* (W#16#...)	说明
0000	未发生错误。
80A1	输入参数 DB_NUMBER 中有错误：所选的实际参数 ● 等于“0” ● 大于所用 CPU 允许的最大 DB 编号。
80B1	CPU 上不存在具有指定编号的 DB。
80B2	无法使用“ATTR_DB”指令读取运动控制工艺对象的数据块。
常见错误信息	另请参见“扩展指令的常见错误代码 (页 557)”
* 在程序编辑器中，错误代码可显示为整数或十六进制值。	

9.10.4 DELETE_DB（删除数据块）

表格 9-224 DELETE_DB 指令

LAD/FBD	SCL	说明

	`ret_val := DELETE_DB(`
	`REQ := _bool_in_,`
	`DB_NUMBER := _uint_in_,`
	`BUSY => _bool_out_);`
		“DELETE_DB”指令用于删除通过调用“CREATE_DB (页 539)”指令由用户程序创建的数据块 (DB)。
		如果数据块不是通过“CREATE_DB”创建的，DELETE_DB 将通过参数 RET_VAL 返回错误代码 W#16#80B5。
		DELETE_DB 调用不会立即删除选定的数据块，而是在执行循环 OB 后的循环控制点处删除。

功能描述

“DELETE_DB”指令将异步执行。即，可通过多次调用执行这一指令。在 REQ = 1 时调用该指令，将开始中断传送。

输出参数 BUSY 和输出参数 RET_VAL 的第 2 个和第 3 个字节用于显示作业状态。

当输出参数 BUSY 的值为 FALSE 时，数据块的删除即完成。

参数

下表列出了指令“DELETE_DB”的参数：

参数	声明	数据类型	存储区	说明
REQ	Input	BOOL	I、Q、M、D、L 或常量	REQ =1：请求删除在参数 DB_NUMBER 中指定编号的 DB
DB_NUMBER	Input	UINT	I、Q、M、D、L 或常量	要删除的 DB 的编号
RET_VAL	Output	INT	I、Q、M、D、L	错误信息（请参见“RET_VAL 参数”）
BUSY	Output	BOOL	I、Q、M、D、L	BUSY= 1：该过程尚未完成。

有关有效数据类型的更多信息，请参见 “数据类型 (页 119)”。

参数 RET_VAL

错误代码* (W#16#...)	说明
0000	未发生错误。
7000	首次调用时，REQ = 0：未激活数据传送；BUSY 的值为“0”。
7001	首次调用时，REQ = 1：已触发数据传送；BUSY 的值为“1”。
7002	中间调用（与 REQ 无关）：已激活数据传送；BUSY 的值为“1”。
80A1	输入参数 DB_NUMBER 中有错误： ● 参数的值为“0”。 ● 参数值大于所用 CPU 允许的最大 DB 编号。
80B1	CPU 上不存在具有指定编号的 DB。
80B4	无法删除该 DB，原因是 CPU 存储卡受到写保护。
80B5	未使用“CREATE_DB”创建该 DB。
80BB	装载存储器空间不足。
80C3	由于存在临时资源限制，此时不能执行 “删除 DB”功能。
常见错误信息	另请参见 “扩展指令的常见错误代码 (页 557)”
* 在程序编辑器中，错误代码可显示为整数或十六进制值。	

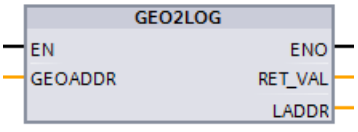
9.11 处理地址

9.11.1 GEO2LOG（根据插槽确定硬件标识符）

可使用 GEO2LOG 指令根据插槽信息确定硬件标识符。

表格 9-225 GEO2LOG 指令

LAD/FBD	SCL	描述

	`ret_val := GEO2LOG(`
	`GEOADDR:=_variant_in_out_,`
	`laddr:=_word_out_);`
		可使用 GEO2LOG 指令根据插槽信息确定硬件标识符。

GEO2LOG 指令根据您使用 GEOADDR 系统数据类型定义的插槽信息来确定硬件标识符：

根据在 HWTYPE 参数处定义的硬件的类型，可通过其它 GEOADDR 参数评估以下信息：

- HWTYPE = 1 时（PROFINET IO 系统）：
  - 仅评估 IOSYSTEM。不考虑 GEOADDR 的其它参数。
  - 输出 PROFINET IO 系统的硬件标识符。
- HWTYPE = 2 时（PROFINET IO 设备）：
  - 评估 IOSYSTEM 和 STATION。不考虑 GEOADDR 的其它参数。
  - 输出 PROFINET IO 设备的硬件标识符。
- HWTYPE = 3 时（机架）：
  - 仅评估 IOSYSTEM 和 STATION。不考虑 GEOADDR 的其它参数。
  - 输出机架的硬件标识符。
- HWTYPE = 4 时（模块）：
  - 评估 IOSYSTEM、STATION,以及 SLOT。不考虑 GEOADDR 的 SUBSLOT 参数。
  - 输出模块的硬件标识符。
- HWTYPE = 5 时（子模块）：
  - 评估 GEOADDR 的所有参数。
  - 输出子模块的硬件标识符。

未评估 GEOADDR 系统数据类型的 AREA 参数。

表格 9-226　参数的数据类型

参数和类型		数据类型	说明
GEOADDR	IN/OUT or IN ?	Variant	指向 GEOADDR 系统数据类型结构的指针。GEOADDR 系统数据类型包含可用于确定硬件 ID 的插槽信息。 更多信息，请参见“GEOADDR 系统数据类型” (页 556)。
RET_VAL	OUT or RETURN ?	Int	错误信息输出。
LADDR	OUT	HW_ANY	组件或模块的硬件标识符。 此编号为自动分配，存储在硬件配置的属性中。

有关有效数据类型的更多信息，请参见 STEP 7 在线帮助中的“有效数据类型概述”。

表格 9-227 条件代码

RET_VAL* (W#16#...)	说明
0	未出错。
8091	GEOADDR 中 HWTYPE 的值无效。
8094	GEOADDR 中 IOSYSTEM 的值无效。
8095	GEOADDR 中 STATION 的值无效。
8096	GEOADDR 中 SLOT 的值无效。
8097	GEOADDR 中 SUBSLOT 的值无效。
* 错误代码可能在程序编辑器中显示为整数或十六进制值。	

9.11.2 LOG2GEO（根据硬件标识符确定插槽）

使用 LOG2GEO 指令从逻辑地址中确定属于硬件标识符的地理地址（模块插槽）。

表格 9-228 LOG2GEO 指令

LAD/FBD	SCL	说明

	`ret_val := LOG2GEO(`
	`  laddr:=_word_in_,`
	
	`GEOADDR:=_variant_in_out_);`
		可使用 LOG2GEO 指令确定属于硬件标识符的模块插槽。

LOG2GEO 指令根据硬件标识符来确定逻辑地址的地理地址：

- 使用 LADDR 参数根据硬件标识符选择逻辑地址。
- GEOADDR 中包含 LADDR 输入所给定的逻辑地址的地理地址。

说明

在 HW 类型不支持组件的情况下，将返回模块 0 的子插槽号。
如果 LADDR 输入未寻址到 HW 对象，则发生错误。

表格 9-229　参数的数据类型

参数和类型		数据类型	说明
LADDR	IN	HW_ANY	IO 系统或模块的硬件标识符。此编号为自动分配，将存储在 CPU 属性或硬件配置的接口中。
RET_VAL	OUT	Int	指令的错误代码
GEOADDR	IN_OUT	Variant	指向 GEOADDR 系统数据类型的指针。GEOADDR 系统数据类型包含插槽信息。 更多信息，请参见“GEOADDR 系统数据类型” (页 556)。

有关有效数据类型的更多信息，请参见 STEP 7 在线帮助中的“有效数据类型概述”。

表格 9-230　条件代码

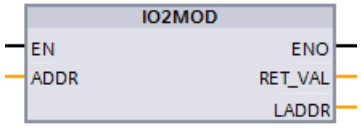
RET_VAL (W#16#...)	说明
0000	无错误
8090	LADDR 参数指定的地址无效。
* 错误代码可能在程序编辑器中显示为整数或十六进制值。	

9.11.3　IO2MOD（根据 I/O 地址确定硬件标识符）

可使用 IO2MOD 指令根据（子）模块的 I/O 地址确定该模块的硬件标识符。

表格 9-231　IO2MOD 指令

LAD/FBD	SCL	说明

	`ret_val := IO2MOD(`
	`ADDR:=_word_in_,`
	`LADDR:=_word_out_);`
		可使用 IO2MOD 指令确定属于硬件标识符的模块插槽。

IO2MOD 指令根据（子）模块的 I/O 地址（I、Q、PI、PQ）确定该模块的硬件标识符。

在 ADDR 参数中输入 IO 地址。如果在此参数中使用了一系列 IO 地址，仅通过评估第一个地址来确定硬件标识符。如果正确指定了第一个地址，则在 ADDR 处指定的地址长度没有任何意义。如果使用了包含多个模块或未使用地址的地址区域，则还可以确定第一个模块的硬件标识符。

如果在 ADDR 参数中未指定（子）模块的 IO 地址，则会在 RET_VAL 参数处输出错误代码“8090”。

说明

SCL 中的 IO 地址输入

在 SCL 中，无法通过 IO 地址 ID“%QWx:P”进行编程。这种情况下，可使用过程映像中的符号变量名称或绝对地址。

表格 9-232 参数的数据类型

参数	声明	数据类型	存储区	说明
ADDR	IN or IN/ OUT ?	Variant	I、Q、M、D、L	（子）模块内的 IO 地址（I、Q、PI、PQ）。 确保片段访问未用于参数 ADDR。如果使用了片段访问，将会在 LADDR 参数处输出不正确的值。
RET_VAL	OUT or RETURN ?	Int	I、Q、M、D、L	指令的错误代码。
LADDR	OUT	HW_IO	I、Q、M、D、L	IO（子）模块的确定的硬件标识符（逻辑地址）。

有关有效数据类型的更多信息，请参见 STEP 7 在线帮助中的“有效数据类型概述”。

表格 9-233 条件代码

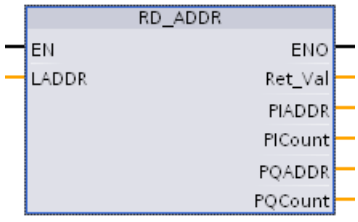
RET_VAL* (W#16#...)	说明
0	未出错。
8090	在 ADDR 参数处指定的 IO 地址未被任何硬件组件使用。
* 错误代码可能在程序编辑器中显示为整数或十六进制值。	

9.11.4 RD_ADDR（根据硬件标识符确定 IO 地址）

使用 RD_ADDR 指令获取子模块的 I/O 地址。

表格 9-234 RD_ADDR 指令

LAD/FBD	SCL	说明

	`ret_val := RD_ADDR(`
	`laddr:=_word_in_,`
	`PIADDR=>_udint_out_,`
	`PICount=>_uint_out_,`
	`PQADDR=>_udint_out_,`
	`PQCount=>_uint_out_,);`
		使用 RD_ADDR 指令获取子模块的 I/O 地址。

RD_ADDR 指令根据子模块的硬件标识符确定输入或输出的长度和起始地址：

- 使用 LADDR 参数根据硬件标识符选择输入或输出模块。
- 以下输出参数根据其为输入模块还是输出模块加以使用：
  - 如果是输入模块，则在 PIADDR 和 PICOUNT 参数中输出确定值。
  - 如果是输出模块，则在 PQADDR 和 PQCOUNT 参数中输出确定值。
- PIADDR 和 PQADDR 参数各自包含模块 I/O 地址的起始地址。
- PICOUNT 和 PQCOUNT 参数各自包含输入或输出的字节数（8 位输入/输出对应 1 个字节，16 位输入/输出对应 2 个字节）。

表格 9-235 参数的数据类型

参数和类型		数据类型	说明
LADDR	IN	HW_IO	（子）模块的硬件标识符
RET_VAL	OUT	Int	指令的错误代码
PIADDR	OUT	UDInt	输入模块的起始地址
PICOUNT	OUT	UInt	输入的字节数
PQADDR	OUT	UDInt	输出模块的起始地址
PQCOUNT	OUT	UInt	输出的字节数

有关有效数据类型的更多信息，请参见 STEP 7 在线帮助中的“有效数据类型概述”。

表格 9-236 条件代码

RET_VAL (W#16#...)	说明
0000	无错误
8090	LADDR 参数中的模块硬件标识符无效。
* 错误代码可能在程序编辑器中显示为整数或十六进制值。	

9.11.5 GEOADDR 系统数据类型

地理地址

系统数据类型 GEOADDR 包含模块地理地址（或插槽信息）。

- PROFINET IO 的地理地址：
  对于 PROFINET IO，地理地址由 PROFINET IO 系统 ID、设备号、插槽号和子模块（如果使用子模块）组成。
- PROFIBUS DP 的地理地址：
  对于 PROFIBUS DP，地理地址由 DP 主站系统的 ID、站号和插槽号组成。

可在每个模块的硬件配置中找到模块的插槽信息。

GEOADDR 系统数据类型的结构

如果在数据块中输入“GEOADDR”作为数据类型，将自动创建结构 GEOADDR。

参数名称		数据类型	描述
GEOADDR		STRUCT	
	HWTYPE	UINT	硬件类型： • 1：IO 系统 (PROFINET/PROFIBUS) • 2：IO 设备/DP 从站 • 3：机架 • 4：模块 • 5：子模块 如果指令不支持某种硬件类型，则输出 HWTYPE“0”。
	AREA	UINT	区域 ID： • 0 = CPU • 1 = PROFINET IO • 2 = PROFIBUS DP • 3 = AS-i
	IOSYSTEM	UINT	PROFINET IO 系统（0 = 机架中的中央单元）
	STATION	UINT	• 区域标识符 AREA = 0 时表示机架号（中央模块）。 • 区域标识符 AREA > 0 时表示站号。
	SLOT	UINT	插槽号
	SUBSLOT	UINT	子模块编号。如果无子模块可用或无法插入任何子模块，则此参数的值为“0”。

9.12 扩展指令的常见错误代码

表格 9-237 扩展指令的常见错误代码

条件代码 (W#16#....)[1]	说明
8x22[2]	存储区对于输入太小
8x23	存储区对于输出太小
8x24	输入区非法
8x25	输出区非法
8x28	输入位赋值非法

条件代码 (W#16#....)[1]	说明
8x29	输出位赋值非法
8x30	输出区是只读 DB。
8x3A	DB 不存在。

[1] 如果执行代码块时出现其中一个错误，则 CPU 保持在 RUN（默认）或组态为 STOP。 也可以在该代码块中使用 GetError 或 GetErorID 指令在本地处理错误（CPU 保持在 RUN 状态），并编写程序来响应错误。

[2] “x”表示错误的参数编号。 参数编号从 1 开始。

工艺指令 10

10.1 计数（高速计数器）

“计数器操作” (页 230)中所述的基本计数器指令限于发生在低于 S7-1200 CPU 扫描周期速率的计数事件。高速计数器 (HSC) 功能提供了发生在高于 PLC 扫描周期速率的计数脉冲。此外，还可以组态 HSC 以测量或设置脉冲发生的频率和周期，如运动控制可以通过 HSC 读取电机编码器信号。

要使用 HSC 功能，首先必须使用“设备组态”(Device Configuration) 画面中的 CPU“属性” (Properties) 选项卡启用并组态 HSC。初次组态 HSC，请参见“组态高速计数器” (页 575)。

在下载硬件组态后，HSC 可以计数脉冲或测量频率而不需要任何调用指令。当 HSC 处于“计数”(Count) 或“周期”(Period) 模式，计数值在每个扫描周期的过程映像（I 存储器）中被自动捕获并更新。如果 HSC 处于频率模式，过程映像值为频率 (Hz)。

除计数和测量外，HSC 还可以生成硬件中断事件，进行取决于物理输入点的状态的操作，并根据指定的计数器事件生成一个输出脉冲（仅 V4.2 或以上版本的 CPU）。工艺指令 CTRL_HSC_EXT 允许用户程序以编程的方式控制 HSC。CTRL_HSC_EXT 更新 HSC 参数并在执行后返回最近更新值。当 HSC 处于“计数”(DB)、“周期”(DB)、“频率”(DB) 模式时，可以使用 CTRL_HSC_EXT 指令。

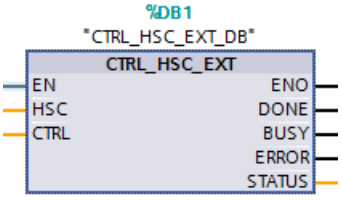
说明

CTRL_HSC_EXT 指令代替了针对 V4.2 或更新版本 CPU 项目的早期 CTRL_HSC 指令。所有 CTRL_HSC 指令功能及多个附加功能可用于 CTRL_HSC_EXT 指令。早期 CTRL_HSC 指令仅能够与早期 S7-1200 项目兼容且不应在新项目中使用。

10.1.1 CTRL_HSC_EXT（控制高速计数器）指令

10.1.1.1 指令概述

表格 10-1 CTRL_HSC_EXT 指令

LAD/FBD	SCL	描述
	"CTRL_HSC_1_DB" (hsc:=_hw_hsc_in_, done:=_done_out_, busy:=_busy_ out_, error:=_error_out_, status:=_status_out_, ctrl:=_variant_in_);	全部 CTRL_HSC_EXT （控制高速计数器（扩展））指令都使用系统定义的数据结构（存储在用户自定义的全局背景数据块中）存储计数器数据。将 HSC_Count、HSC_Period 或 HSC_Frequency 数据类型作为输入参数分配到 CTRL_HSC_EXT 指令。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“CTRL_HSC_1_DB”是背景 DB 的名称。

表格 10-2 参数的数据类型

参数	声明	数据类型	描述
HSC	IN	HW_HSC	HSC 标识符
CTRL	IN_OUT	Variant	SFB 输入和返回数据。 注：更多相关信息，请参见“CTRL_HSC_EXT 指令系统数据类型 (SDT) (页 565)”。
DONE	OUT	Bool	1= 表示 SFB 已完成。始终为 1，因为 SFB 为同步模式
BUSY	OUT	Bool	始终为 0，因为功能从未处于繁忙状态
ERROR	OUT	Bool	1 = 表示错误
STATUS	OUT	Word	执行条件代码 注：如需了解更多信息，请参见下方的“执行条件代码”表。

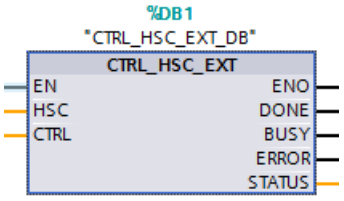
表格 10-3　执行条件代码

STATUS (W#16#)	描述
0	无错误
80A1	HSC 标识符没有对 HSC 寻址
80B1	新方向中的非法值
80B4	新周期中的非法值
80B5	新 Op 模式行为中的非法值
80B6	新限制行为中的非法值
80D0	SFB 124 不可用

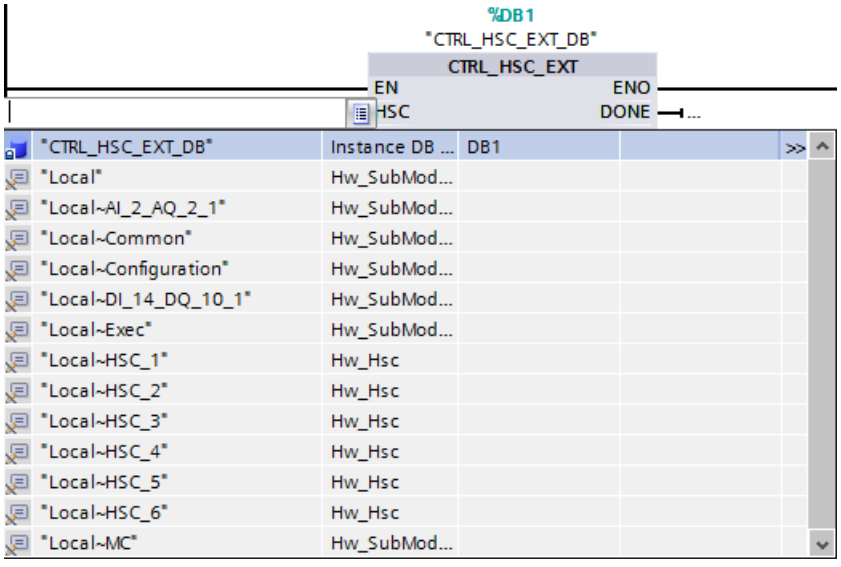
10.1.1.2 示例

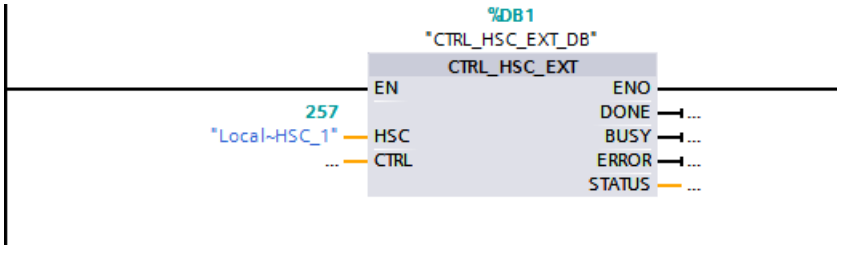
要使用 CTRL_HSC_EXT 指令，请按照下列步骤操作：

1. 将 CTRL_HSC_EXT 指令放在梯形图程序段中，同时也生成下列背景数据块：“CTRL_HSC_EXT_DB”：



2. 将在 HSC 属性中找到的 HSC 的硬件标识符连接到梯形图指令的“HSC”引脚。也可以从此输入引脚的下拉菜单中选择 6 个“Hw_Hsc”对象中的 1 个。HSC1 的默认变量名称为 "Local~HSC_1"：



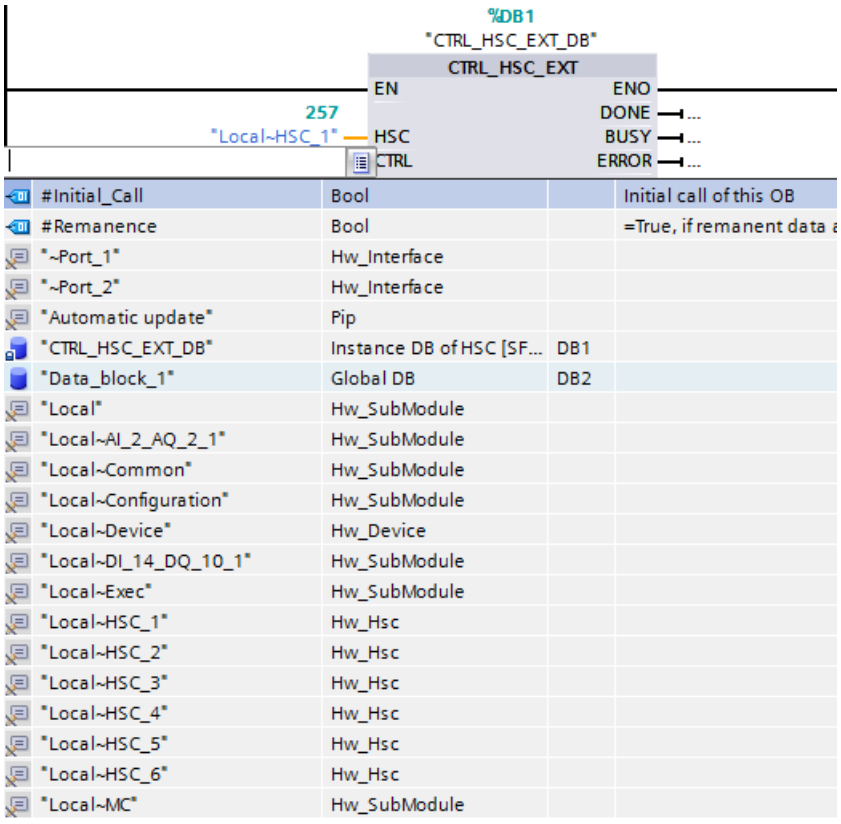


3. 生成名为“Data_block_1”的全局数据块（也可以使用现有的全局数据块。）：
   – 在“Data_block_1”中，找到一个空行，并添加一个名称为“MyHSC”的变量。
   – 在 “数据类型”(Data type) 列中，添加以下系统数据类型 (SDT) 之一。选择与 HSC 组态的计数类型对应的 SDT。稍后，可以在本部分中找到更多 HSC SDT 的相关信息。下拉菜单不包含这些类型，因此确保准确键入如下所示的 SDT 名称：HSC_Count、HSC_Period 或 HSC_Frequency
   – 输入数据类型后，可以扩展“MyHSC”变量以查看所有包含在数据结构中的字段。在此可以找到每个字段的数据类型并修改默认的起始值：

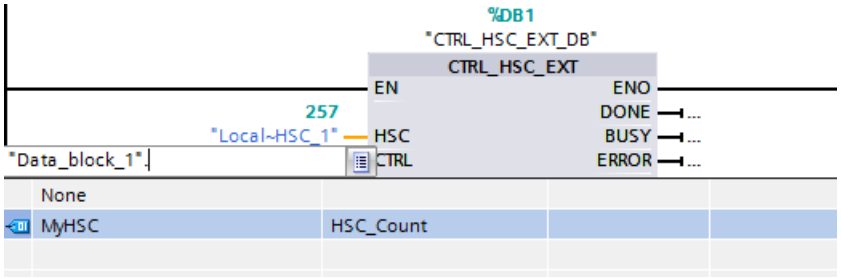
Data_block_1

	名称	数据类型	启动值
1	Static		
2	MyHSC	HSC_Count	
3	CurrentCount	DInt	0
4	CapturedCount	DInt	0
5	SyncActive	Bool	false
6	DirChange	Bool	false
7	CmpResult_1	Bool	false
8	CmpResult_2	Bool	false
9	OverflowNeg	Bool	false
10	OverflowPos	Bool	false
11	EnHSC	Bool	false
12	EnCapture	Bool	false
13	EnSync	Bool	false
14	EnDir	Bool	false
15	EnCV	Bool	false
16	EnSV	Bool	false
17	EnReference1	Bool	false
18	EnReference2	Bool	false
19	EnUpperLmt	Bool	false
20	EnLowerLmt	Bool	false
21	EnOpMode	Bool	false
22	EnLmtBehavior	Bool	false
23	EnSyncBehavior	Bool	false
24	NewDirection	Int	0
25	NewOpModeBeha...	Int	0
26	NewLimitBehavior	Int	0
27	NewSyncBehavior	Int	0
28	NewCurrentCount	DInt	0
29	NewStartValue	DInt	0
30	NewReference1	DInt	0
31	NewReference2	DInt	0
32	NewUpperLimit	DInt	0
33	New_Lower_Limit	DInt	0

4. 将变量“Data_block_1'.MyHSC”赋值到 CTRL_HSC_EXT 指令 CTRL 输入针脚。
   – 选择“Data_Block_1”。



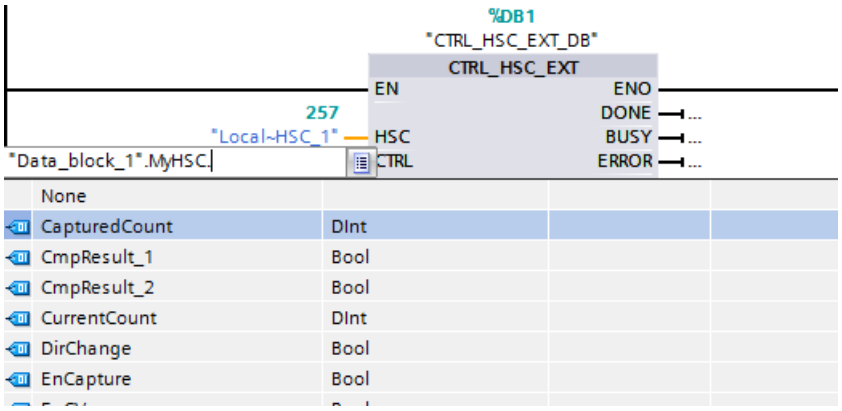
– 选择“MyHSC”。



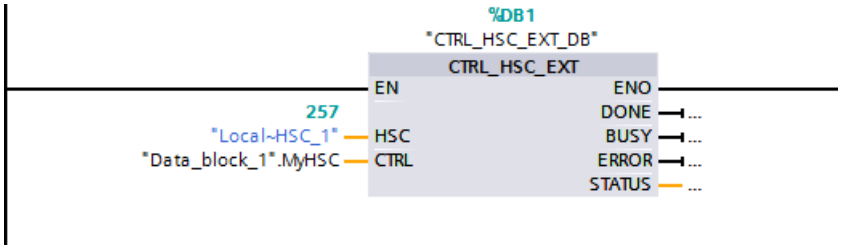
– 删除“'Data_Block_1'.MyHSC”后的句点 (".")。然后，单击框外或按一次 ESC 键，然后按 Enter 键。

说明

删除“'Data_Block_1'.MyHSC”后的句点 (".") 后，不要只按 Enter 键。此动作会将句点 (".") 重新放入框中。



– 完整 CTRL 输入如下所示。



在 PLC 中组态 HSC 后，可以执行 CTRL_HSC_EXT 指令。如果发生错误，则 ENO 将设置为 “0”，且 STATUS 输出将指示条件代码。

参见

CTRL_HSC_EXT 指令系统数据类型 (SDT) (页 565)

10.1.1.3 CTRL_HSC_EXT 指令系统数据类型 (SDT)

下列系统数据类型 (SDT) 仅用于 CTRL_HSC_EXT 指令的 CTRL 引脚。要想使用这些系统数据类型，需要创建用户数据块并添加与 HSC 组态模式（计数类型）对应的 SDT 数据类型的对象。STEP 7 不会在下拉菜单中显示这些数据类型。如下所示输入确切的 SDT 名称。

HSC 的 SDT 输入用前缀“En”或“New”来表示。带有前缀“En”或“New”的输入启用 HSC 功能或更新相应参数。前缀“New”表示更新值。要使新值生效，相应“En”位必须为真且“New”值必须有效。当执行 CTRL_HSC_EXT 指令时，程序用于输入修改并通过附加的相应 SDT 参考更新输出。

SDT：HSC_Count

“HSC_Count”数据类型与用于为“计数”模式组态的 HSC 对应。在计数模式提供以下功能：

- 访问当前脉冲计数
- 在输入事件上锁存当前脉冲计数
- 在输入事件上将当前脉冲计数复位为起始值
- 访问状态位，说明发生特定 HSC 事件
- 使用软件或硬件输入禁用 HSC
- 使用软件或硬件输入更改计数方向
- 更改当前脉冲计数
- 更改起始值（当 CPU 切换到 RUN 状态或触发同步函数时使用）
- 更改用于比较的两个独立参考（或预置）值
- 更改计数上限和下限
- 当脉冲计数达到这些限制，更改 HSC 运行方式
- 在当前脉冲计数达到参考（预设）值时，生成硬件中断事件
- 当同步（复位）输入激活时，生成硬件中断事件
- 当计数方向随着外部输入发生改变时，生成硬件中断事件
- 在指定计数事件上生成单输出脉冲

当事件发生且 CTRL_HSC_EXT 指令执行时，指令会设置状态位。在执行如下 CTRL_HSC_EXT 指令时，指令会清除状态位，除非在指令执行前事件再次发生。

表格 10-4　HSC_Count 结构

结构元素	声明	数据类型	描述
CurrentCount	输出	Dint	返回 HSC 的当前计数值
CapturedCount	输出	Dint	返回在指定输入事件上捕获的计数值
SyncActive	输出	Bool	状态位：同步输入已激活
DirChange	输出	Bool	状态位：计数方向已更改
CmpResult1	输出	Bool	状态位：CurrentCount 等于发生的 Reference1 事件
CmpResult2	输出	Bool	状态位：CurrentCount 等于发生的 Reference2 事件
OverflowNeg	输出	Bool	状态位：CurrentCount 达到最低下限值
OverflowPos	输出	Bool	状态位：CurrentCount 达到最高上限值

结构元素	声明	数据类型	描述
EnHSC	输入	Bool	当为真时，启用 HSC 进行计数脉冲；当为假时，禁用计数功能。
EnCapture	输入	Bool	当为真时，启用捕获输入；当为假时，捕获输入无效。
EnSync	输入	Bool	当为真时，启用同步输入，当为假时，同步输入无效。
EnDir	输入	Bool	启用 NewDirection 值生效
EnCV	输入	Bool	启用 NewCurrentCount 值生效
EnSV	输入	Bool	启用 NewStartValue 值生效
EnReference1	输入	Bool	启用 NewReference1 值生效
EnReference2	输入	Bool	启用 NewReference2 值生效
EnUpperLmt	输入	Bool	启用 NewUpperLimit 值生效
EnLowerLmt	输入	Bool	启用 New_Lower_Limit 值生效
EnOpMode	输入	Bool	启用 NewOpModeBehavior 值生效
EnLmtBehavior	输入	Bool	启用 NewLimitBehavior 值生效
EnSyncBehavior	输入	Bool	不使用此值。
NewDirection	输入	Int	计数方向：1 = 加计数；-1 = 减计数；所有其它值保留。
NewOpModeBehavior	输入	Int	正在溢出的 HSC 的：1 = HSC 停止计数（HSC 必须禁用并重新启用才能继续计数）；2 = HSC 继续操作；所有其它值保留。
NewLimitBehavior	输入	Int	正在溢出的 CurrentCount 值的结果：1 = 将 CurrentCount 设置为相反限值；2 = 将 CurrentCount 设置为开始值；所有其它值保留。
NewSyncBehavior	输入	Int	不使用此值。
NewCurrentCount	输入	Dint	CurrentCount 值
NewStartValue	输入	Dint	StartValue：HSC 初始值
NewReference1	输入	Dint	Reference1 值
NewReference2	输入	Dint	Reference2 值

结构元素	声明	数据类型	描述
NewUpperLimit	输入	Dint	计数上限值
New_Lower_Limit	输入	Dint	计数下限值

SDT：HSC_Period

“HSC_Period”数据类型与用于为“周期”模式组态的 HSC 对应。利用 CTRL_HSC_EXT 指令，程序可以按指定测量间隔访问输入脉冲数量。此指令允许用高纳秒精度计算输入脉冲之间的时间间隔。

表格 10-5　HSC_Period 结构

结构元素	声明	数据类型	描述
ElapsedTime	OUT	UDInt	参见以下描述。
EdgeCount	OUT	UDInt	参见以下描述。
EnHSC	IN	Bool	为真时，启用周期测量的 HSC；为假时，则禁用周期测量。
EnPeriod	IN	Bool	启用 NewPeriod 值生效。
NewPeriod	IN	Int	指定测量间隔时间（毫秒）。允许的值只有 10、100 或 1000 ms。

ElapsedTime 返回连续测量间隔最后一个计数事件之间的时间（单位：纳秒）。若在测量间隔内无计数事件发生，则 ElapsedTime 返回自最后一个计数事件算起的累计时间。ElapsedTime 的范围为 0 至 4,294,967,280 纳秒（0x0000 0000 至 0xFFFF FFF0）。返回值 4,294,967,295 (0xFFFF FFFF) 表示发生溢出的周期。溢出表示在脉冲边缘之间的时间大于 4.295 秒且周期无法用此指令进行计算。自 0xFFFF FFF1 至 0xFFFF FFFE 的值为保留值。

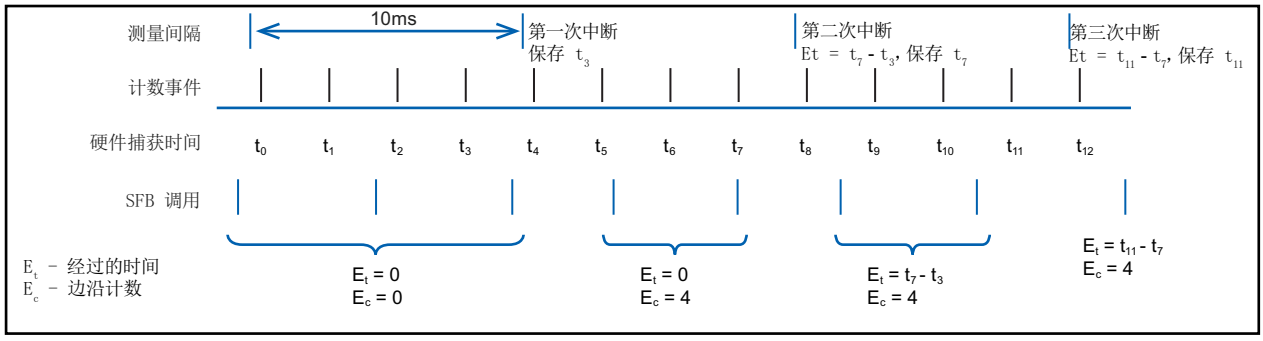
EdgeCount 返回测量间隔内计数事件的数量。只有在 EdgeCount 值大于 0 时才能计算周期。如果 ElapsedTime 为“0”（没有收到输入脉冲）或 0xFFFF FFFF（出现周期溢出），则 EdgeCount 中的值无效。

当 EdgeCount 有效，使用下列公式计算周期（纳秒）：周期 = ElapsedTime/EdgeCount

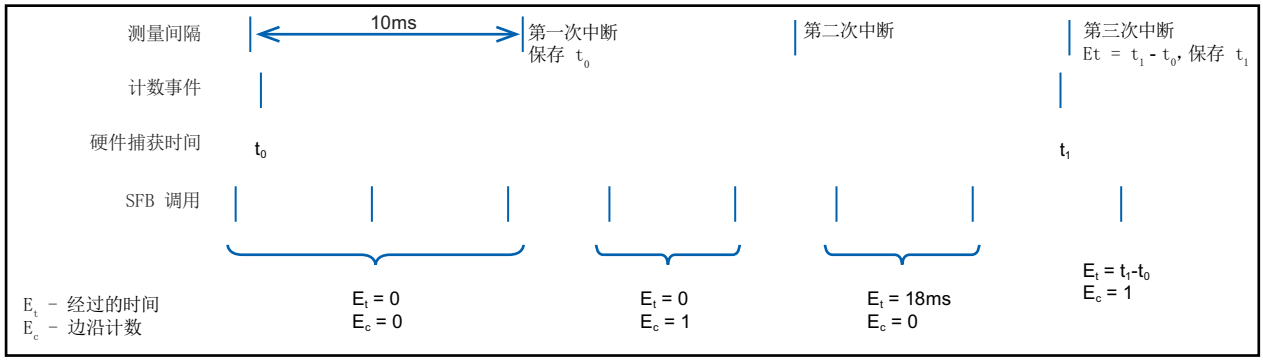
计算的时间周期值为测量间隔内发生的所有脉冲的平均时间周期。如果输入脉冲周期大于测量间隔（10、100 或 1000 ms），那么周期计算需要多个测量间隔。

以下示例描述如何采用该指令进行周期测量：

示例 1：测量间隔中的多个计数事件



示例 2：多个测量间隔中零个和一个计数事件



规则：
1. 如果 $E1_t$ =0，则周期无效
2. 否则，周期 = E_t / E_c

SDT：HSC_Frequency

“HSC_Frequency”数据类型与用于为“频率”模式组态的 HSC 对应。利用 CTRL_HSC_EXT 指令，程序可以按指定时间周期访问指定高速计数器的输入脉冲数量。

在频率模式下通过 CTRL_HSC_EXT 指令可以使用以下功能：

表格 10-6 HSC_Frequency structure

结构元素	声明	数据类型	说明
频率	输出	DInt	返回以“Hz”为单位的频率值，涵盖测量间隔时间。HSC 减计数时，指令会返回一个负频率值。
EnHSC	输入	Bool	为 True 时，启用频率测量的 HSC；若为 False 时，则禁用频率测量。
EnPeriod	IN	Bool	启用 NewPeriod 值生效。
NewPeriod	输入	Int	指定测量间隔时间（毫秒）。其值只能为 10、100 或 1000 毫秒。

CTRL_HSC_EXT 指令通过与周期模式相同的测量方法测量频率，从而找到 ElapsedTime 和 EdgeCount。这个指令用以下公式计算有符号的整数 Hz 频率：频率 = EdgeCount/ ElapsedTime

频率值若为浮点数，可用上面的公式在 HSC 为周期模式时计算频率。注意：在周期模式下，返回的 ElapsedTime 值以纳秒为单位，可能需要对该值进行比例缩放。

10.1.2 使用高速计数器。

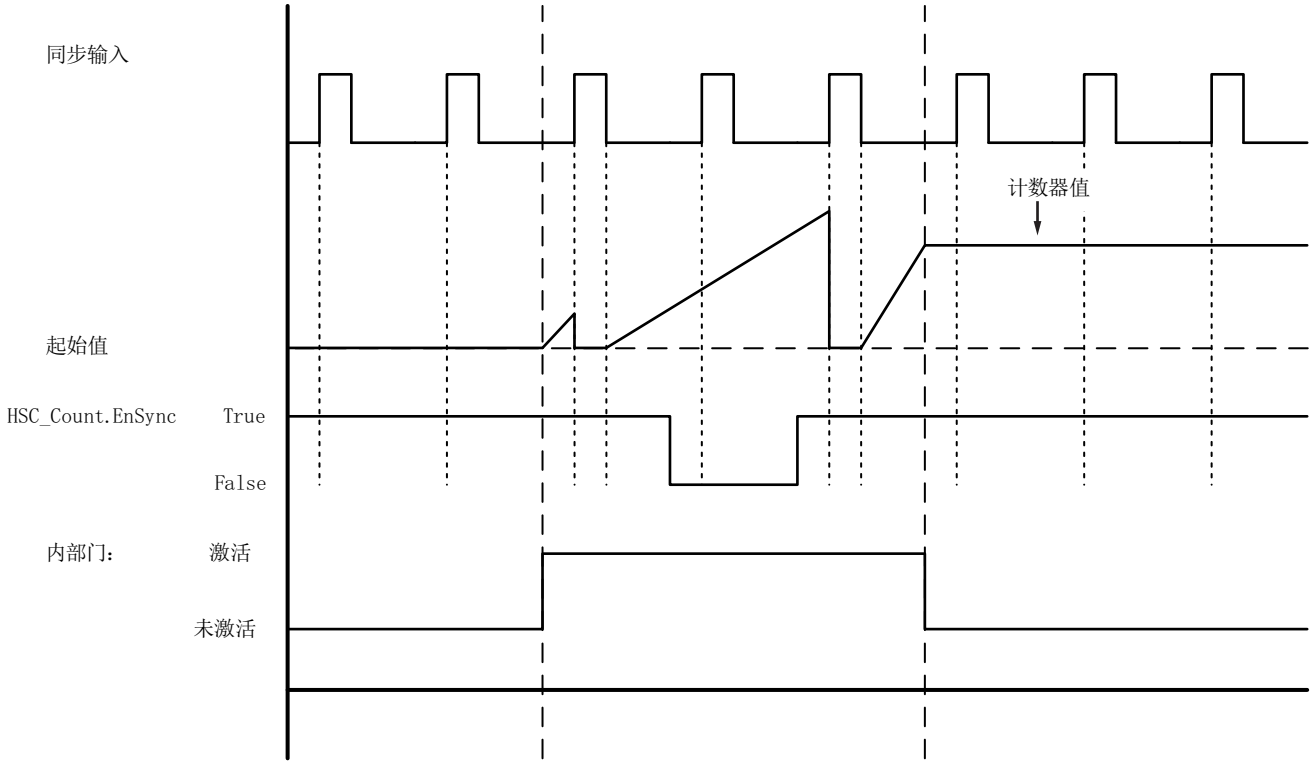
10.1.2.1 同步功能

同步功能可通过外部输入信号给计时器设置起始刻度值。也可通过执行 CTRL_HSC_EXT 指令对起始刻度值进行更改。这样，用户可以将当前计数值与所需的外部输入信号出现值同步。

同步始终以输入信号出现值为准，且无论内部门状态如何，同步始终有效。必须将“HSC_Count.EnSync”位设为 true 才能启用同步功能。

同步完成后，CTRL_HSC_EXT 指令会将 HSC_Count.SyncActive 状态位设置为 true。但如果在上次指令执行时未进行同步，CTRL_HSC_EXT 指令则会将 HSC_Count.SyncActive 状态位设置为 false。

下图为组态活跃等级高的输入信号时的同步示例：



说明

组态输入过滤器会延迟数字量输入的控制信号。
此输入点功能仅可用在组态计数模式的 HSC 时使用。

有关如何组态同步功能的信息，请参见输入功能 (页 582)。

10.1.2.2 门功能

许多应用需要根据其他事件的情况来开启或关闭计数程序。出现这类情况时，便会通过内部门功能来开启或关闭计数。每个 HSC 通道有两个门：软件门和硬件门。这些门的状态将决定内部门的状态。请参见下表。

如果软件门和硬件门都处于打开状态或尚未进行组态，则内部门会打开。如果内部门打开，则开始计数。如果内部门关闭，则会忽略其他所有计数脉冲，且停止计数。

表格 10-7 门功能状态

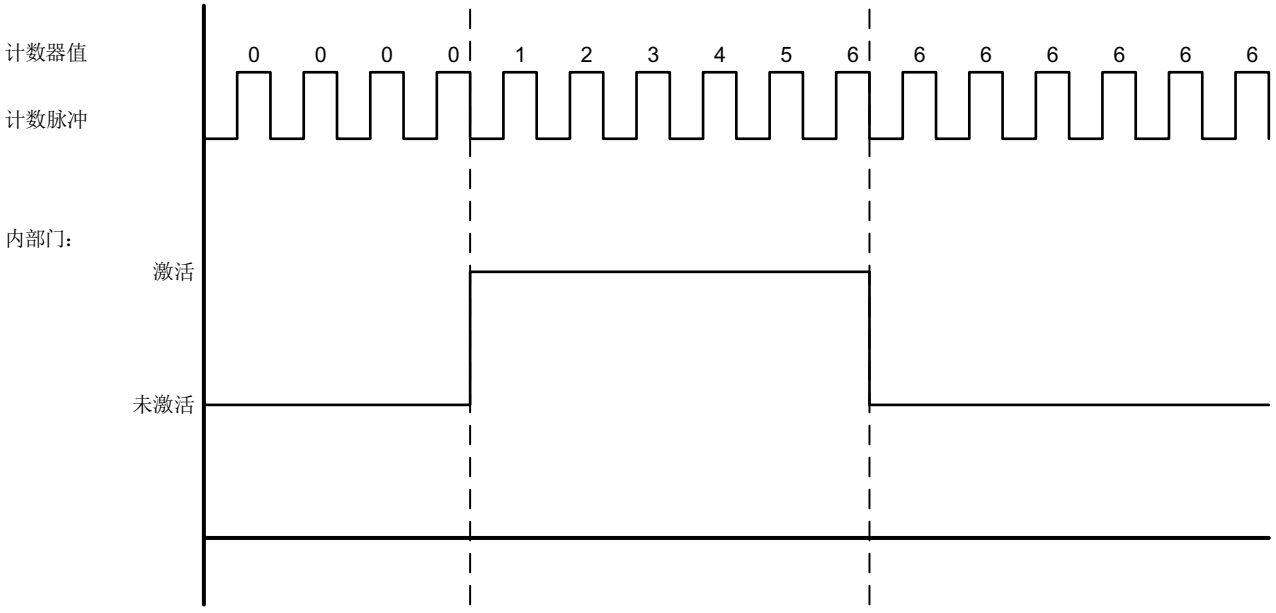
硬件门	软件门	内部门
打开/未组态	打开	打开
打开/未组态	已关闭	已关闭
已关闭	打开	已关闭
已关闭	已关闭	已关闭

术语“打开”用于表示门处于的活动状态。同理，术语“已关闭”用于表示门处于的静止状态。

使用与 CTRL_HSC_EXT 指令关联的 SDT 中的“HSC_Count.EnHSC”使能位可对控制软件门进行控制。开启软件门时，将“HSC_Count.EnHSC”位设置为 true，关闭软件门时，将“HSC_Count.EnHSC”位设置为 false。执行 CTRL_HSC_EXT 指令可以更新软件门的状态。

硬件门为备选件，可以在 HSC 属性区启用或禁用硬件门。仅通过硬件门控制计数过程时，软件门需要保持打开状态。如果不对硬件门组态，则硬件门将始终视为打开且内部门的状态会与软件门的状态相同。

下图显示用数字量输入来打开或关闭硬件门的实例。组态高活跃等级的数字量输入：



说明

组态输入过滤器会延迟数字量输入的控制信号。

硬件门功能仅可用在组态计数模式的 HSC 时使用。在“周期”和“频率”模式下，内部门的状态与软件门的状态相同。

在周期模式下，通过“HSC_Period.EnHSC”控制软件门。

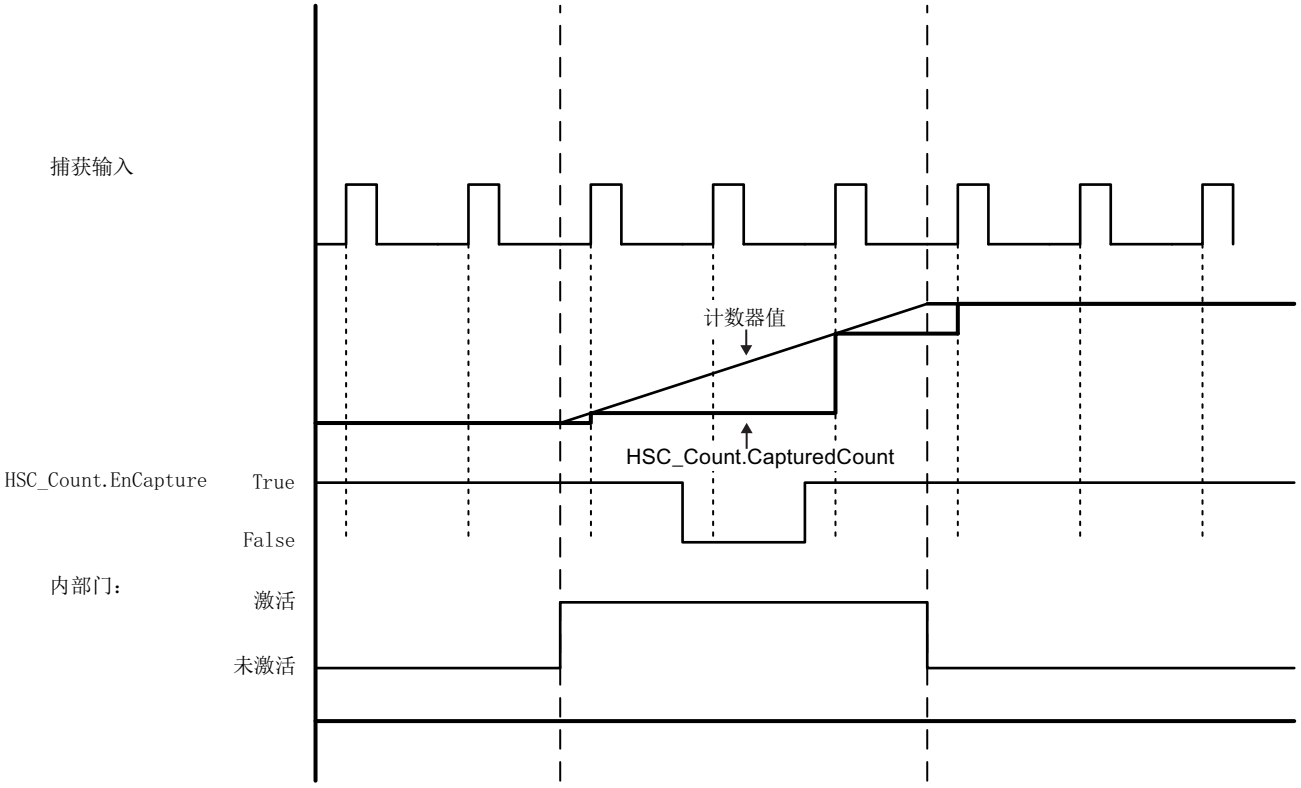
在频率模式下，通过“HSC_Frequency.EnHSC”控制软件门。

有关如何组态门功能的信息，请参见输入功能 (页 582)。

10.1.2.3 捕获功能

可使用“捕获”功能通过外部参照信号来保存当前计数值。通过“HSC_Count.EnCapture”位组态并启用捕获功能后，捕获功能会在外部输入沿出现的位置捕获当前计数。无论内部门的状态如何捕获功能始终有效。程序会在门关闭后保存未更改的计数器值。执行 CTRL_HSC_EXT 指令后，程序会在“HSC_Count.CapturedCount”存储捕获值。

下图显示了组态捕获功能在上升沿上进行捕获的示例。当通过 CTRL_HSC_EXT 指令将“HSC_Count.EnCapture”位设置为 false 时，捕获输入不会触发捕获当前计数。



说明

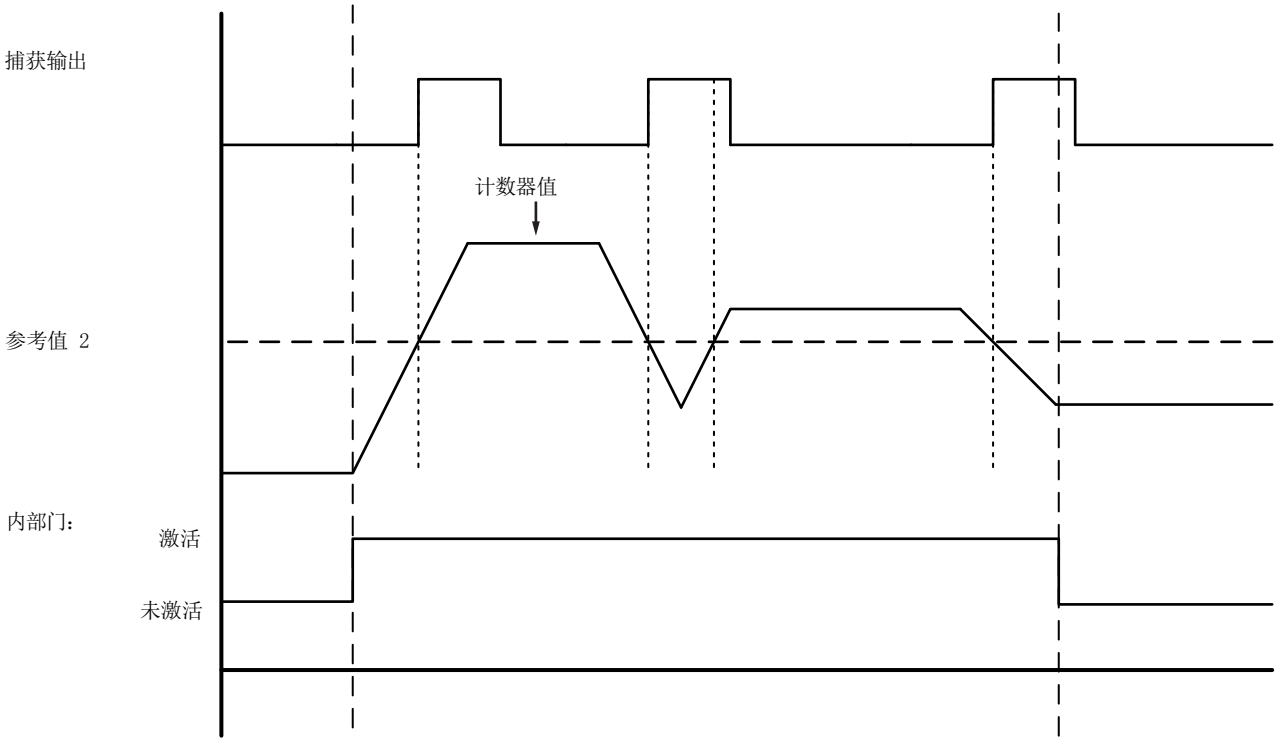
组态输入过滤器会延迟数字量输入的控制信号。

此输入点功能仅可用在组态计数模式的 HSC 时使用。

有关如何组态“捕获”功能的信息，请参见输入功能 (页 582)。

10.1.2.4 比较功能

启用“比较”输出值功能会生成一个可组态脉冲，每次发生组态的事件时便会产生脉冲。这些事件将包括与其中一个参照值或计数机溢出相等的计数。如果正在脉冲且又发生了组态的事件，则该事件不会产生脉冲。



说明

此输出功能仅可在组态计数模式的 HSC 时使用。

有关如何组态“比较”功能的信息，请参见输出功能 (页 582)。

10.1.2.5 应用

较为典型的应用是利用 HSC 监控增量轴编码器发出的返回信息。轴编码器提供指定的每转计数，可作为输入到 HSC 的时钟发生器输入值使用。另外，每转产生一次的复位脉冲，可作为输入到 HSC 的同步输入值使用。

启动时，程序将第一个参考值加载至 HSC，并将输出值设置为初始状态。这些输出值会在当前计数小于参考值期间保持初始状态。在当前计数等于参考值、发生同步事件（复位）以及方向改变时，HSC 将中断程序。

在每个计数都等于参考值时，则会发生中断事件，并且程序会将新的参考值加载至 HSC，并将输出值设置为下一个状态。在发生同步中断事件时，程序会设置第一个参考值和第一个输出状态，并重复此循环。

由于中断的频率远低于 HSC 的计数速率，因此能够在对 CPU 扫描周期影响相对较小的情况下实现对高速操作的精确控制。通过提供中断，可以在独立的中断例程中执行每次的新预设值装载操作以实现简单的状态控制。也可以在一个中断程序中处理所有中断事件。

由用户程序或外部输入信号触发的“门”功能能够中止编码器脉冲的计数。通过取消激活门可以忽略轴任何的移动。这样，当编码器继续将脉冲发送至 HSC 时，计数值会保持在门停止移动前产生的最后一个值。在门移动时，计数会从门停止移动前产生的最后一个值开始回复计数。

启用“捕捉”功能会在外部输入出现的地方捕获当前计数。过程（比如，校准例程）可使用该功能确定在事件之间产生的脉冲。

启用“比较输出”功能会生成一个可组态脉冲，每次当前计数达到其中一个参考值或溢出值（超过计数限值）时便会产生脉冲。当某个 HSC 事件发生时，可使用该脉冲作为信号启动另一个过程。

通过用户程序或外部输入信号控制计数方向。

可组态“频率”模式的 HSC 来实现旋转轴的速度。该功能可提供有符号的整数值（以 HZ 为单位）。因为复位信号每转产生一次，所以测量复位信号频率则提供了轴转速的快速提示信息（按每秒转数计算）。

如果需要该频率的浮点值，则需组态“周期”模式的 HSC。可通过“周期”返回的 ElapsedTime 与 EdgeCount 值计算频率。

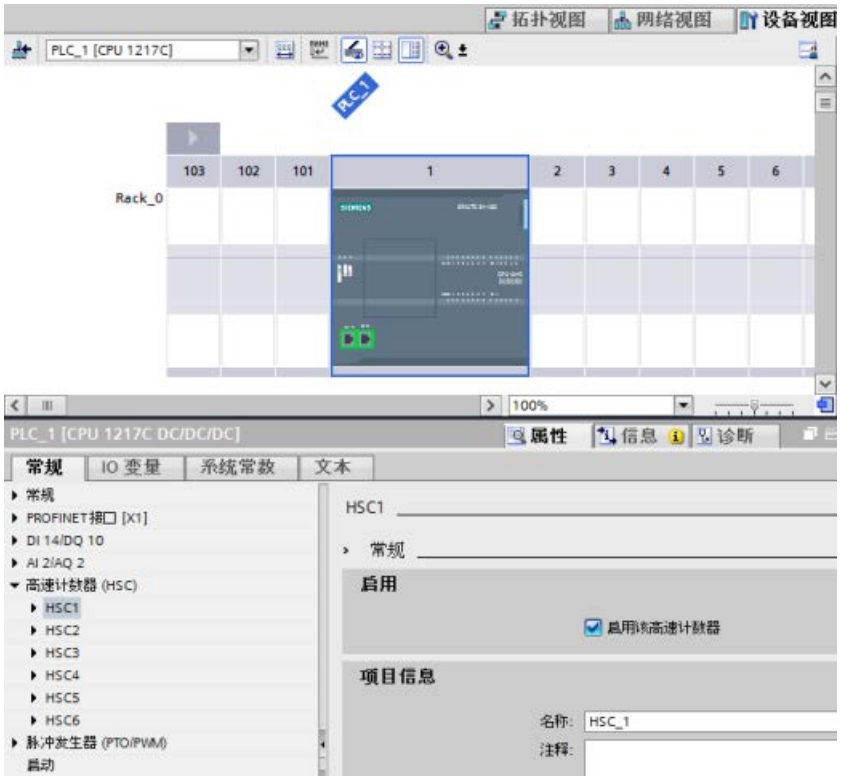
10.1.3 组态高速计算器

设置告诉计数器 (HSC)：

- 选择项目浏览器中的“设备组态”(Device Configuration)。
- 选择要组态的 CPU。
- 单击位于巡视窗口中的“属性”(Properties) 选项卡（请参见下图）。
- 在“常规”(General) 选项卡所示列表下选择要启用的 HSC（请参见下图）。

最多可组态六个高速计数器（HSC1 至 HSC6）。通过选择“启用此高速计数器”(Enable this high speed counter) 选项启用 HSC。启用后，STEP 7 随即为该 HSC 指定一个唯一的默认名称。可通过在“名称”(Name) 编辑字段编辑默认名称来更改它，但是名称必须是唯一的。已启用的 HSC 名称将变为“系统常量”(System constant) 变量表中具有“HW_Hsc”数据类型

的变量，并可用作 CTRL_HSC_EXT 指令的“HSC”参数。更多相关信息，请参见“组态 CPU 的运行” (页 163)：



启用 HSC 后，STEP 7 会将单相位计数设置为默认组态。HSC 时钟发生器输入的数字输入过滤器设置好之后，可将程序下载至 PLC，然后 CPU 会做好计数的准备。若要更改 HSC 的组态，请进行下一部分“计数类型”的操作。

下表简要列出了每个组态可用的输入与输出：

表格 10-8　HSC 的计数模式

类型	输入 1	输入 2	输入 3	输入 4	输入 5	输出 1	功能
具有内部方向控制的单相	时钟	-	-	-	-	-	计数、频率或周期
			同步	门	捕获	比较	计数
具有外部方向控制的单相	时钟	方向	-	-	-	-	计数、频率或周期
			同步	门	捕获	比较	计数

类型	输入 1	输入 2	输入 3	输入 4	输入 5	输出 1	功能
两阶段	加时钟	减时钟	-	-	-	-	计数、频率或周期
			同步	门	捕获	比较	计数
A/B 计数器	A 相	B 相	-	-	-	-	计数、频率或周期
			同步 [1]	门	捕获	比较	计数
A/B 计数器的四相	A 相	B 相	-	-	-	-	计数、频率或周期
			同步 [1]	门	捕获	比较	计数

[1] 对于编码器：Z 相，归位

10.1.3.1 HSC 的类型

计数或模式的类型共有四种。当更改模式时，可用于 HSC 组态的选项也会更改：

- 计数：计算脉冲次数并根据方向控制的状态递增或递减计数值。外部 I/O 可在指定事件上重置计数、取消计数、启动当前值捕获及产生单相。输出值为当前计数值且该计数值在发生捕获事件时产生。
- 周期：会在指定的时间周期内计算输入脉冲的次数。返回脉冲的计数及持续时间（单位为：纳秒）。会在频率测量周期指定的时间周期结束后，捕获并计算值。“周期”(Period) 模式可用于 CTRL_HSC_EXT 指令，但不适用于 CTRL_HSC 指令。
- 频率：测量输入脉冲和持续时间，然后计算出脉冲的频率。程序会返回一个有符号的双精度整数的频率（单位为 Hz）。如果计数方向向下，该值为负。会在频率测量周期指定的时间周期结束时，捕获并计算值。
- 运动控制：用于运动控制工艺对象，不适用于 HSC 指令。有关详细信息，请参见“运动控制 (页 657)”。

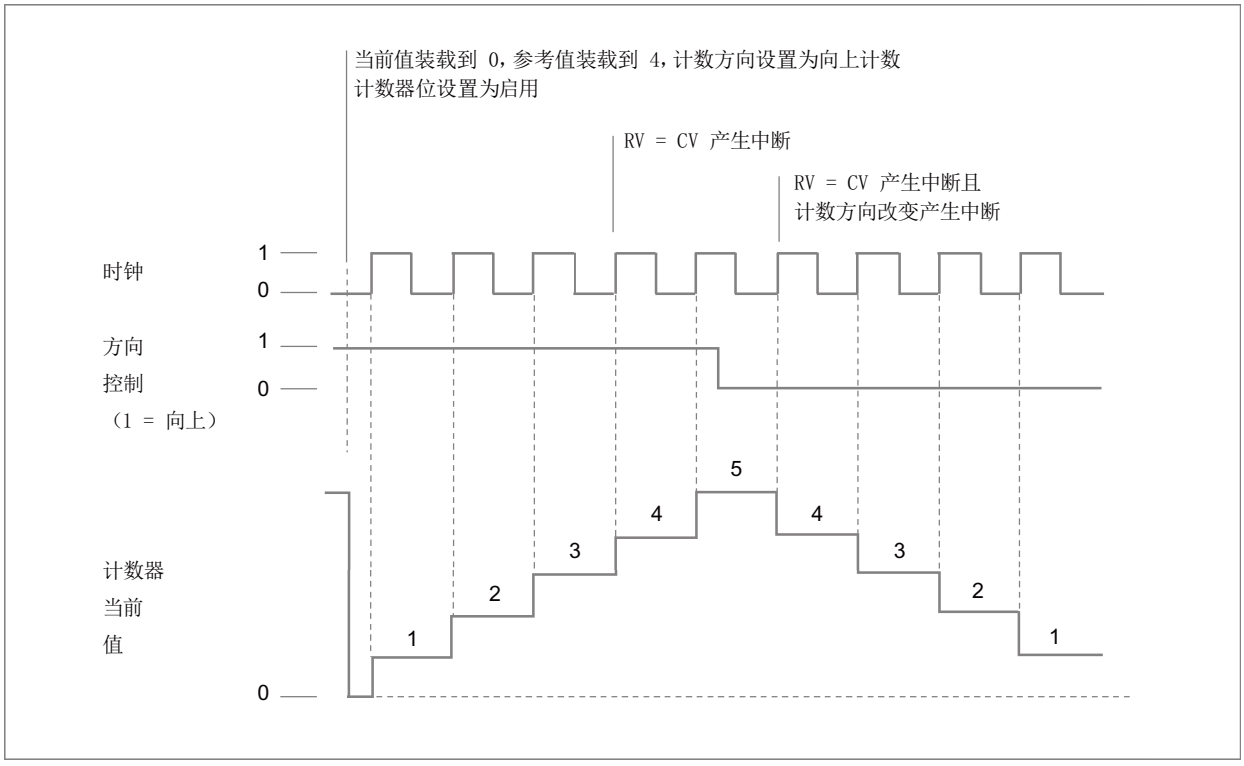
10.1.3.2 运行阶段

选择所需要的 HSC 运行阶段。在更改计数值、出现当期值 (CV) 等于参考值 (RV) 事件及出现方向改变的事件时，将会显示以下四个值。

单相

单相（不适用于运动控制）计数脉冲：

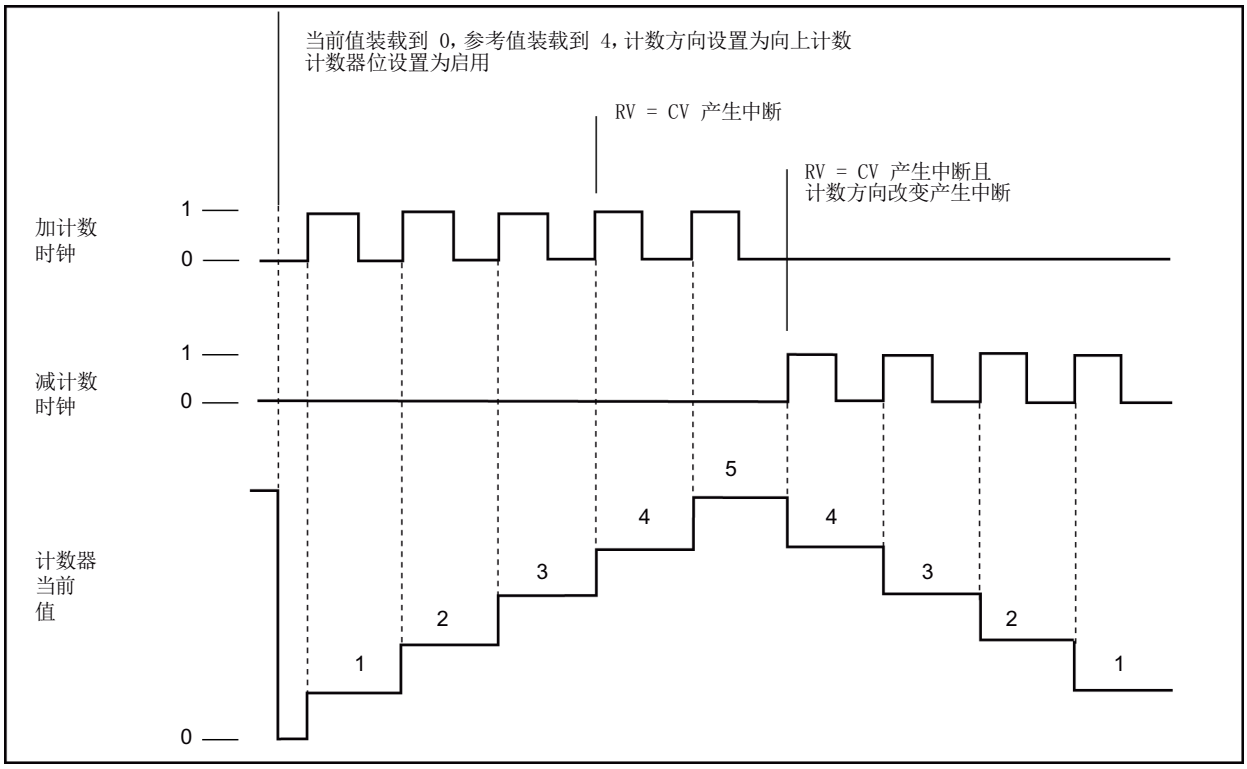
- 用户程序（内部方向控制）：
  - “1”为向上
  - “-1”为向下
- 硬件输入（外部方向控制）：
  - “高级”为向上。
  - “低级”为向下。



两个相位

两个相位计数：

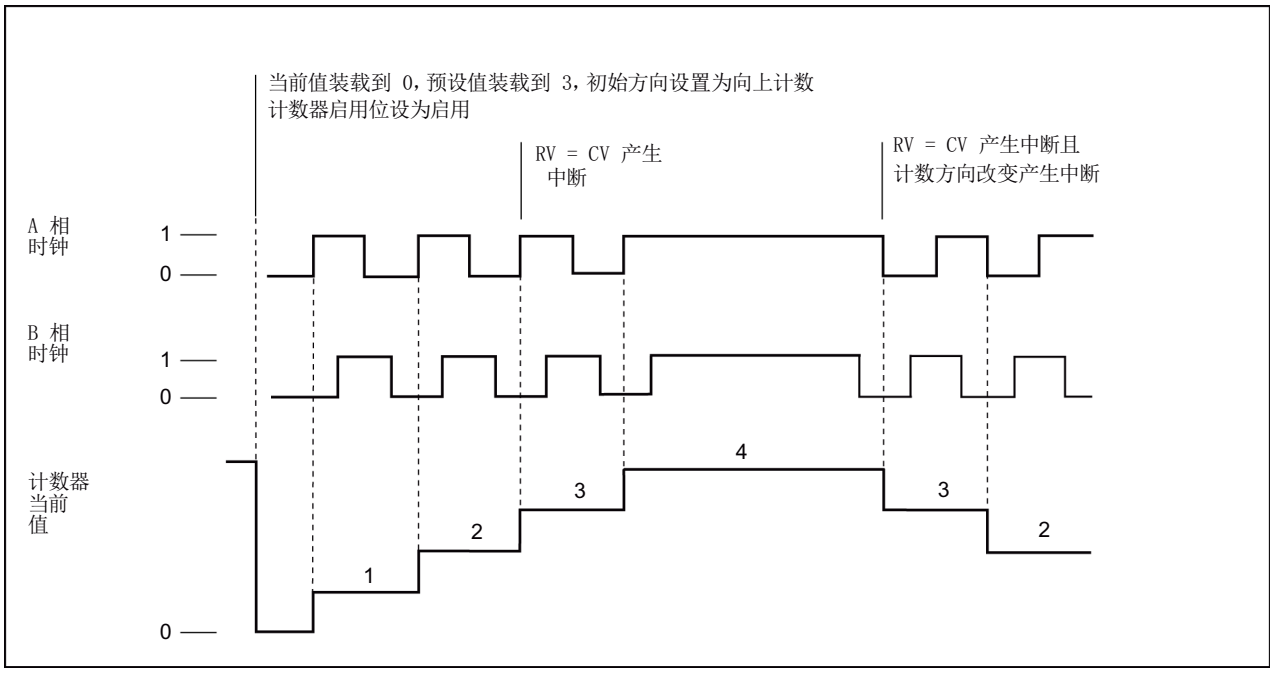
- 加时钟输入向上
- 减时钟输入向下



A/B 计数器

A/B 相正交计数：

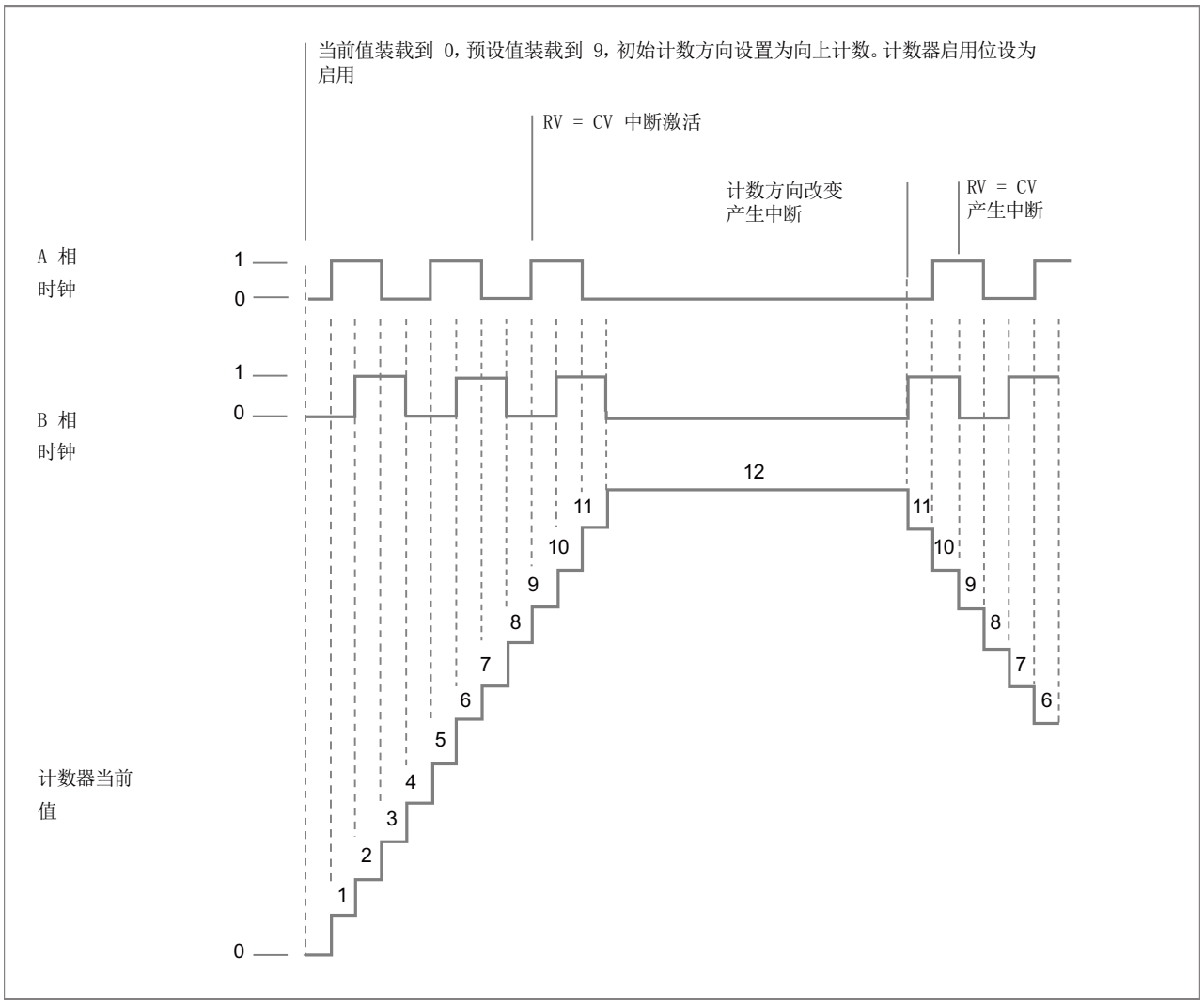
- 计时钟 B 输入值低时，计时钟 A 输入值的上升沿向上
- 计时钟 B 输入值低时，计时钟 A 输入值的下降沿向下



A/B 计数器的四相

A/B 相正交四相计数：

- 计时钟 B 输入值低时，计时钟 A 输入值的上升沿向上
- 计时钟 B 输入值高时，计时钟 A 输入值的下降沿向上
- 计时钟 A 输入值高时，计时钟 B 输入值的上升沿向上
- 计时钟 A 输入值低时，计时钟 B 输入值的下降沿向上
- 计时钟 A 输入值低时，计时钟 B 输入值的上升沿向下
- 计时钟 A 输入值高时，计时钟 B 输入值的下降沿向下
- 计时钟 B 输入值高时，计时钟 A 输入值的上升沿向下
- 计时钟 B 输入值低时，计时钟 A 输入值的下降沿向下



10.1.3.3 初始值

CPU 每次开始运行时会加载初始值。初始值仅在计数模式中使用：

- 初始计数器值：CPU 从 STOP 模式转变为 RUN 模式或程序触发同步输入时，程序会将当前计数值设置为初始值。
- 初始参考值：在当前计数到达参考值时，如果已设置好了相关的功能，程序将会生成一个中断信号和/或一个输出脉冲。
- 初始参考值 2：在当前值达到参考值 2 时，如果已设置好了功能，程序将会生成一个输出脉冲。
- 初始上限值：最大计数值。最大默认值为 +2,147,483,647 次脉冲。
- 初始下线值：最小计数值。最小默认值为 -2,147,483,648 次脉冲。

上述值和计数器达到限值时的行为仅可以在“计数”模式下使用。可以通过 HSC_Count SDT 的 CTRL_HSC_EXT 指令调整这些值和行为。

10.1.3.4 输入功能

计时钟和方向的输入值可根据运行阶段来确定计数的事件与方向。计数模式下，仅可使用“同步”、“捕获”及门输入值，并且能够启用和组态各种触发类型的输入。

同步输入

同步输入可将当前计数值设置为起始值（或初始计数器）。通常可用同步输入将计数器重置为“0”。当输入引脚为以下其中一种状态时，可触发同步：

- 高
- 低
- 由低升高
- 由高降低
- 由高降低，或由低升高

捕获输入

捕获输入可将捕获到的计数值设置到触发捕获输入时保存的计数值。当输入引脚为以下其中一种状态时，可触发捕获：

- 由低升高
- 由高降低
- 由高降低，或由低升高

门输入

门输入可停止 HSC 计数。当输入引脚为以下其中一种状态时，可触发门：

- 高
- 低

10.1.3.5 输出功能

比较输出功能是 HSC 的唯一输出，且仅在“计数”模式下可用。

比较输出

在出现以下事件时，可通过组态比较输出生成一个脉冲：

- 计数器值等于参考值（计数方向：向上）
- 计数器值等于参考值（计数方向：向下）
- 计数器值等于参考值（计数方向：向上或向下）
- 计数器值等于参考值 2（计数方向：向上）
- 计数器值等于参考值 2（计数方向：向下）
- 计数器值等于参考值 2（计数方向：向上）
- 上升沿溢出
- 下降沿溢出

可在 1 至 500 ms 循环周期范围内组态输出脉冲；默认循环时间为 10 ms。可在 1 至 100% 范围内任意设置脉宽或占空比；默认脉宽为 50%。

如果有多个比较输出事件在指定的循环周期内出现，则会因为当前脉冲尚未完成循环而导致这些事件产生的输出脉冲丢失。当前脉冲完成后（组态的循环时间已过），脉冲发生器便会生成新的脉冲。

10.1.3.6 中断事件

事件组态区下，可通过下拉菜单（或创建新的 OB）选择硬件中断 OB，然后将其连接到 HSC 事件。中断的优先级取值范围在 2 至 26 之间，其中 2 为最低优先级，26 为最高优先级。根据 HSC 组态的情况，可使用以下事件：

- 计数器值等于参考值事件：HSC 的计数值达到参考值时，便会发生计数器值等于参考值事件组态期间在初始参考值区下对参考值进行设置，或通过 CTRL_HSC_EXT 指令更新 “NewReference1”进行设置。有关详细信息，请参阅 “运行阶段 (页 577)”部分。
- 同步事件：只要启用并触发同步输入便会发生同步。
- 更改方向事件：计数方向发生变化时，发生更改方向事件有关详细信息，请参阅 “运行阶段 (页 577)”部分。

10.1.3.7 硬件输入针脚分配

每次启用 HSC 输入时，在 CPU 或可选信号板上选择需要的输入点，（通信与信号模块不支持 HSC 输入）。选择输入点时，STEP 7 在选项旁显示最大频率值。数字量输入滤波器的设置可能需要调整，以便所有有效信号频率都可以通过滤波器。关于设置 HSC 输入滤波器，请参见“组态数字量输入滤波器时间 (页 165)”。

说明

CPU 和 SB 输入通道（V4 或更高版本的固件）具有可组态的输入滤波时间

早期固件版本具有无法更改的固定 HSC 输入通道和固定滤波时间。

V4 或更高版本可以分配输入通道和滤波时间。默认输入滤波设置为 6.4 ms，将最大计数速率限制为 78 Hz。根据您系统设计的情况，可更改滤波器设置来计数更高或更低的频率。

⚠ 警告

为数字量输入通道更改滤波时间设置的风险

如果数字量输入通道的滤波时间更改自以前的设置，则新的“0”电平输入值可能需要保持长达 20.0 ms 的累积时间，然后滤波器才会完全响应新输入。在此期间，可能不会检测到持续时间少于 20.0 ms 的短“0”脉冲事件或对其计数。

更改滤波时间会引发意外的机械或过程操作，这可能会导致人员死亡、重伤和/或设备损坏。

对 CPU 循环上电以确保新的滤波时间立即生效。

使用以下表格并确保连接的 CPU 和 SB 输入通道可以支持过程信号中的最大脉冲速率：

表格 10-9 CPU 输入：最大频率

CPU	CPU 输入通道	运行阶段：单相或两个相位	运行阶段：A/B 计数器或 A/B 计数器的四相
1211C	Ia.0 到 Ia.5	100 kHz	80 kHz
1212C	Ia.0 到 Ia.5	100 kHz	80 kHz
	Ia.6，Ia.7	30 kHz	20 kHz
1214C 和 1215C	Ia.0 到 Ia.5	100kHz	80kHz
	Ia.6 到 Ib.5	30 kHz	20 kHz
1217C	Ia.0 到 Ia.5	100 kHz	80 kHz
	Ia.6 到 Ib.1	30 kHz	20 kHz
	Ib.2 到 Ib.5 （.2+，.2- 到 .5+，.5-）	1 MHz	1 MHz

表格 10-10 SB 信号板输入：最大频率（可选信号板）

SB 信号板	SB 输入通道	运行阶段：单相或两个相位	运行阶段：A/B 计数器或 A/B 计数器的四相
SB 1221，200 kHz	Ie.0 到 Ie.3	200kHz	160 kHz
SB 1223，200 kHz	Ie.0，Ie.1	200kHz	160 kHz
SB 1223	Ie.0，Ie.1	30 kHz	20 kHz

将输入点分配至 HSC 功能时，可将相同的输入点分配给多个 HSC 功能。例如，将 I0.3 分配给 HSC1 同步输入和 HSC2 同步输入来同步相同时间内的 HSC 计数为有效的组态，但是会生成编辑器警告。

尽可能避免将同一个 HSC 的多个输入功能分配至相同输入点。例如，可以有效组态为将 I0.3 分配至同步输入和 HSC1 门输入来同步计数并同时禁用计数。进行这样的组态可能会出现意外的结果。

警告

多个功能分配至单个数字输入通道的风险

将相同 HSC 的多个输入功能分配给一个公共输入点会导致意想不到的结果。多个功能分配给一个触发器产生触发时，无法掌握 PLC 执行功能的顺序。这种情况称为紊乱情况，并且会出现意外情况。

这种紊乱情况会引发意外的机械或过程操作，这可能会导致人员死亡、重伤和/或设备损坏。

要避免紊乱情况，请不要将同一个 HSC 的两个以上的输入功能分配给同一个输入引脚。如果 HSC 将两个输入功能分配给同一个引脚，则需将触发器设置为不同时触发。切记，下降沿与低电平在同一情况下发生，上升沿与高电平在同一情况下发生。

说明

CPU 设备组态时可分配用于高速计数器 (HSC) 设备上的数字量输入点和输出点。在针对 HSC 设备分配输入和输出点时，无法在监视表中通过强制功能更改这些点的值。HSC 能够完全控制这些输入点和输出点。

10.1.3.8 硬件输出引脚的分配

启用比较输出时，请选择可用的输出点。组态 HSC（或其他技术对象，比如，脉冲发生器）使用的输出点时，该输出点只能用于该对象。其他组件无法使用该输出点，并且这个输出点也无法强制设为某个值。为多个 HSC 组态单个输出通道或组态用于 HSC 和脉冲输出的单个输出通道时，程序会生成编辑错误。

10.1.3.9 HSC 输入存储器地址

每个 HSC 都使用保存当前计数的 I-memory 双字部分。如果组态频率的 HSC，则会将频率保存在输入存储器的位置。输入地址的可用范围：I0.0 至 I1023.7（最大起始地址为：I1020.0）。HSC 的输入地址不能与映射到其他组件上的输入地址重叠。有关过程映像的详细信息，请参见“执行用户程序 (页 81)”。

下表列出了为各 HSC 的值分配的默认地址：

表格 10-11 HSC 默认地址

高速计数器 (HSC)	当前值数据类型	当前值默认地址
HSC1	DInt	ID 1000
HSC2	DInt	ID 1004
HSC3	DInt	ID 1008
HSC4	DInt	ID 1012
HSC5	DInt	ID 1016
HSC6	DInt	ID 1020

10.1.3.10 硬件标识符

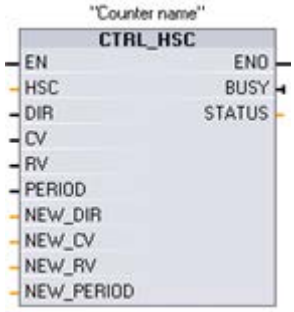
各 HSC 都有一个唯一的硬件标识符，用于 HSC_CTRL 及 HSC_CTRL_EXT 指令。在“系统常量”中能够找到硬件标识符的 PLC 变量。名称为“HSC_1”的 HSC 变量为：“Local~HSC_1”、数据类型为：“Hw_Hsc”。选定 CTRL_HSC_EXT 指令的 HSC 输入值时，HSC 的变量也会在下拉菜单中显示。

10.1.4 早期的 CTRL_HSC（控制高速计数器）指令

10.1.4.1 指令概述

表格 10-12 CTRL_HSC 指令（针对通用计数）

LAD/FBD	SCL	说明

	`"CTRL_HSC_1_DB" (`
	`hsc:=W#16#0,`
	`dir:=False,`
	`cv:=False,`
	`rv:=False,`
	`period:=False,`
	`new_dir:=0,`
	`new_cv:=L#0,`
	`new_rv:=L#0,`
	`new_period:=0,`
	`busy=>_bool_out_,`
	`status=>_word_out_);`
		每个 CTRL_HSC （控制高速计数器）指令都使用 DB 中存储的结构来保存计数器数据。在编辑器中放置 CTRL_HSC 指令后分配 DB。

[1] 插入该指令后，STEP 7 显示用于创建相关数据块的 “调用选项” (Call Options) 对话框。

[2] 在 SCL 示例中，“CTRL_HSC_1_DB”是背景 DB 的名称。

表格 10-13 参数的数据类型

参数	声明	数据类型	说明
HSC	IN	HW_HSC	HSC 标识符
DIR[1, 2]	IN	Bool	1 = 请求新方向
CV[1]	IN	Bool	1 = 请求设置新的计数器值
RV[1]	IN	Bool	1 = 请求设置新的参考值
PERIOD[1]	IN	Bool	1 = 请求设置新的周期值 （仅限频率测量模式）
NEW_DIR	IN	Int	新方向：1= 向上，-1= 向下
NEW_CV	IN	DInt	新计数器值
NEW_RV	IN	DInt	新参考值

参数	声明	数据类型	说明
NEW_PERIOD	IN	Int	以毫秒为单位的新周期值（仅限频率测量模式）。其值只能为 10、100 或 1000 毫秒： 1000 = 1 秒 100 = 0.1 秒 10 = 0.01 秒
BUSY[3]	OUT	Bool	功能忙
STATUS	OUT	Word	执行条件代码

1 如果不请求更新参数值，则将忽略相应的输入值。

2 仅当组态的计数方向设置为“用户程序（内部方向控制）”(User program (internal direction control)) 时，DIR 参数才有效。用户在 HSC 设备组态中确定如何使用该参数。

3 对于 CPU 或 SB 上的 HSC，BUSY 参数的值始终为 0。

您可以在 CPU 的设备组态中为各 HSC 的计数/频率功能、复位选项、中断事件组态、硬件 I/O 以及计数值地址对相应参数进行组态。

可以通过用户程序来修改某些 HSC 参数，从而对计数过程提供程序控制：

- 将计数方向设置为 NEW_DIR 值
- 将当前计数值设置为 NEW_CV 值
- 将参考值设置为 NEW_RV 值
- 将周期值（仅限频率测量模式）设置为 NEW_PERIOD 值

如果执行 CTRL_HSC 指令后以下布尔标记值置位为 1，则相应的 NEW_xxx 值将装载到计数器。CTRL_HSC 指令执行一次可处理多个请求（同时设置多个标记）。

- DIR = 1 是装载 NEW_DIR 值的请求，0 = 无变化
- CV = 1 是装载 NEW_CV 值的请求，0 = 无变化
- RV = 1 是装载 NEW_RV 值的请求，0 = 无变化
- PERIOD = 1 是装载 NEW_PERIOD 值的请求，0 = 无变化

如果出现错误，则 ENO 将设置为“0”，且 STATUS 输出将指示条件代码：

表格 10-14 Execution condition codes

STATUS (W#16#)	说明
0	无错误
80A1	HSC 标识符没有对 HSC 寻址

STATUS (W#16#)	说明
80B1	NEW_DIR 的值非法
80B2	NEW_CV 的值非法
80B3	NEW_RV 的值非法
80B4	NEW_PERIOD 的值非法
80C0	多路访问高速计数器 如果计数类型 (页 577)设置为 “周期”(Period) 或 “运动控制”(Motion control)，则会发生此错误。这些类型不适用于 CTRL_HSC 指令，并且仅受 CTRL_HSC_EXT 指令支持。
80D0	CPU 硬件配置中未启用高速计数器 (HSC)

10.1.4.2 使用 CTRL_HSC

CTRL_HSC 指令通常放置在触发计数器硬件中断事件时执行的硬件中断 OB 中。例如，如果 CV=RV 事件触发计数器中断，则硬件中断 OB 代码块执行 CTRL_HSC 指令并且可通过装载 NEW_RV 值更改参考值。

在 CTRL_HSC 参数中没有提供当前计数值。在高速计数器硬件的组态期间分配存储当前计数值的过程映像地址。可以使用程序逻辑直接读取计数值。返回给程序的值将是读取计数器瞬间的正确计数。但计数器仍将继续对高速事件计数。因此，程序使用旧的计数值完成处理前，实际计数值可能会更改。

10.1.4.3 HSC 当前计数值

CPU 将各 HSC 的当前值存储在输入 (I) 地址中。下表列出了为各 HSC 的当前值分配的默认地址。可通过修改设备组态中的 CPU 属性来更改当前值的 I 地址。

高速计数器使用 DInt 值存储当前计数值。DInt 计数值的取值范围是：-2147483648 至 +2147483647。自 CPU 固件 V4.2 起，可以组态范围值。有关详细信息，请参见 “初始值 (页 581)”。

进行加计数时，计数器从最大正值翻转到最大负值；进行减计数时，计数器从最大负值翻转到最大正值。频率返回值以赫兹为单位（比如：123.4 Hz 的返回值为 123）。

表格 10-15 HSC 默认地址

HSC	当前值数据类型	当前值默认地址
HSC1	DInt	ID1000
HSC2	DInt	ID1004

HSC	当前值数据类型	当前值默认地址
HSC3	DInt	ID1008
HSC4	DInt	ID1012
HSC5	DInt	ID1016
HSC6	DInt	ID1020

10.2 PID 控制

STEP 7 为 S7-1200 CPU 提供以下 PID 指令：

- PID_Compact 指令用于通过连续输入变量和输出变量控制工艺过程。
- PID_3Step 指令用于控制电机驱动的设备，如需要通过离散信号实现打开和关闭动作的阀门。
- PID_Temp 指令提供一个通用的 PID 控制器，可用于处理温度控制的特定需求。

说明

只有 CPU 从 STOP 切换到 RUN 模式后，在 RUN 模式下对 PID 组态和下载进行的更改才会生效。而在“PID 参数”(PID parameters) 对话框中使用 “起始值控制”(Start value control) 进行的更改立即生效。

全部三个 PID 指令（PID_Compact、PID_3Step 和 PID_Temp）都可以计算启动期间的 P 分量、I 分量以及 D 分量（如果组态为“预调节”）。还可以将指令组态为“精确调节”，从而可对参数进行优化。用户无需手动确定参数。

说明

以恒定的采样时间间隔执行 PID 指令（最好在循环 OB 中）。

由于 PID 回路需要一段时间来响应控制值的变化，因此请勿在每个循环中都计算输出值。请勿在主程序循环 OB（如 OB 1）中执行 PID 指令。

PID 算法的采样时间表示两次输出值（控制值）计算之间的时间。在自调节期间计算输出值，并取整为循环时间的倍数。每次调用时都会执行 PID 指令的所有其它函数。

PID 算法

PID（比例/积分/微分）控制器会测量两次调用之间的时间间隔并评估监视采样时间的结果。每次进行模式切换时以及初始启动期间都会生成采样时间的平均值。该值用作监视功能的参考并用于计算。监视包括两次调用之间的当前测量时间和定义的控制器采样时间的平均值。

PID 控制器的输出值由三个分量组成：

- P（比例）：如果通过“P”分量计算，则输出值与设定值和过程值（输入值）之差成比例。
- I（积分）：如果通过“I”分量计算，则输出值与设定值和过程值（输入值）之差的持续时间成比例增加，以最终校正该差值。
- D（微分）：如果通过“D”分量计算，输出值与设定值和过程值（输入值）之差的变化率成函数关系，并随该差值的变化加快而增大。从而根据设定值尽快矫正输出值。

PID 控制器使用以下公式来计算 PID_Compact 指令的输出值。

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s}(w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1}(c \cdot w - x) \right]$$

y	输出值	x	过程值
w	设定值	s	拉普拉斯算子
K_p	比例增益（P 分量）	a	微分延迟系数（D 分量）
T_1	积分作用时间（I 分量）	b	比例作用加权（P 分量）
T_D	微分作用时间（D 分量）	c	微分作用加权（D 分量）

PID 控制器使用以下公式来计算 PID_3Step 指令的输出值。

$$\Delta y = K_p \cdot s \cdot \left[(b \cdot w - x) + \frac{1}{T_I \cdot s}(w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1}(c \cdot w - x) \right]$$

y	输出值	x	过程值
w	设定值	s	拉普拉斯算子
K_p	比例增益（P 分量）	a	微分延迟系数（D 分量）
T_1	积分作用时间（I 分量）	b	比例作用加权（P 分量）
T_D	微分作用时间（D 分量）	c	微分作用加权（D 分量）

10.2.1 插入 PID 指令和工艺对象

STEP 7 提供了两个 PID 控制指令：

- PID_Compact 指令及其相关工艺对象提供具有调节功能的通用 PID 控制器。 工艺对象中包含控制环的所有设置。
- PID_3Step 指令及其相关工艺对象为通过电机驱动的阀门提供具有特定设置的 PID 控制器。 工艺对象中包含控制环的所有设置。 PID_3Step 控制器提供两个附加的布尔型输出。

创建工艺对象之后，必须组态参数 (页 625)。 还应调整自动调节参数（启动期间的“预调节”或手动“精确调节”），以调试 PID 控制器的操作 (页 643)。

表格 10-16 插入 PID 指令和工艺对象

将 PID 指令插入用户程序时，STEP 7 会自动为指令创建工艺对象和背景数据块。 背景数据块包含 PID 指令要使用的所有参数。 每个 PID 指令必须具有自身的唯一背景数据块才能正确工作。 插入 PID 指令并创建工艺对象和背景数据块之后，需组态工艺对象的参数 (页 625)。	

表格 10-17 （可选）通过项目浏览器创建工艺对象

还可以在插入 PID 指令**之前**为项目创建工艺对象。 如果在将 PID 指令插入用户程序之前创建工艺对象，用户便可以在插入 PID 指令时选择工艺对象。	
要创建工艺对象，请在项目浏览器中双击“添加新对象”(Add new object) 图标。	
单击“控制”(Control) 图标并选择适用于该 PID 控制器类型（PID_Compact 或 PID_3Step）的工艺对象。 可以为工艺对象创建可选名称。 单击“确定”(OK) 创建工艺对象。	

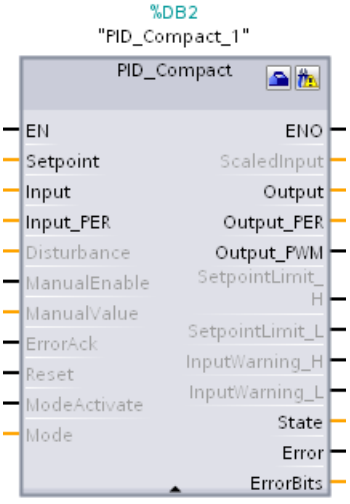
10.2.2 PID_Compact

10.2.2.1 PID_Compact 指令

PID_Compact 指令提供自动和手动模式下具有集成自我调节功能的通用 PID 控制器。

表格 10-18 PID_Compact 指令

LAD/FBD	SCL	说明

	`"PID_Compact_1"(`
	`Setpoint:=_real_in_,`
	`Input:=_real_in_,`
	`Input_PER:=_word_in_,`
	`Disturbance:=_real_in_,`
	`ManualEnable:=_bool_in_,`
	`ManualValue:=_real_in_,`
	`ErrorAck:=_bool_in_,`
	`Reset:=_bool_in_,`
	`ModeActivate:=_bool_in_,`
	`Mode:=_int_in_,`
	`ScaledInput=>_real_out_,`
	`Output=>_real_out_,`
	`Output_PER=>_word_out_,`
	`Output_PWM=>_bool_out_,`
	
	`SetpointLimit_H=>_bool_out_,`
	
	`SetpointLimit_L=>_bool_out_,`
	`InputWarning_H=>_bool_out_,`
	`InputWarning_L=>_bool_out_,`
	`State=>_int_out_,`
	`Error=>_bool_out_,`
	`ErrorBits=>_dword_out_);`
		PID_Compact 提供可在自动模式和手动模式下自我调节的 PID 控制器。PID_Compact 是具有抗积分饱和功能且对 P 分量和 D 分量加权的 PID T1 控制器。

[1] STEP 7 会在插入指令时自动创建工艺对象和背景数据块。该背景数据块包含工艺对象的参数。

[2] 在 SCL 示例中，“PID_Compact_1”是背景 DB 的名称。

表格 10-19 参数的数据类型

参数和类型		数据类型	说明
Setpoint	IN	Real	PID 控制器在自动模式下的设定值。（默认值：0.0）
Input	IN	Real	用户程序的变量用作过程值的源。（默认值：0.0） 如果正在使用 Input 参数，则必须设置 Config.InputPerOn = FALSE。

参数和类型		数据类型	说明
Input_PER	IN	Word	模拟量输入用作过程值的源。（默认值：W#16#0） 如果正在使用 Input_PER 参数，则必须设置 Config.InputPerOn = TRUE。
Disturbance	IN	Real	干扰变量或预控制值
ManualEnable	IN	Bool	启用或禁用手动操作模式。（默认值：FALSE）： • FALSE 至 TRUE 沿激活“手动模式”，同时 State = 4，Mode 保持不变。 ManualEnable = TRUE 时，无法利用 ModeActivate 的上升沿或使用调试对话框更改工作模式。 • TRUE 至 FALSE 沿激活 Mode 分配的工作模式。 注：建议您只使用 ModeActivate 更改工作模式。
ManualValue	IN	Real	手动操作的输出值。（默认值：0.0） 可以使用从 Config.OutputLowerLimit 到 Config.OutputUpperLimit 的值。
ErrorAck	IN	Bool	复位 ErrorBits 和警告输出。FALSE 至 TRUE 沿
Reset	IN	Bool	重新启动控制器。（默认值：FALSE）： • FALSE 至 TRUE 沿： – 切换到“未激活”模式 – 复位 ErrorBits 和警告输出 – 清除积分作用 – 保持 PID 参数 • 只要 Reset = TRUE，则 PID_Compact 便会保持在“未激活”模式 (State = 0)。 • TRUE 至 FALSE 沿： – PID_Compact 切换到保存在 Mode 参数中的工作模式。
ModeActivate	IN	Bool	PID_Compact 切换到保存在 Mode 参数中的工作模式。FALSE 至 TRUE 沿：
Mode	IN	Int	期望的 PID 模式；在 Mode Activate 输入的上升沿激活。
ScaledInput	OUT	Real	标定的过程值。（默认值：0.0）
Output[1]	OUT	Real	REAL 格式的输出值。（默认值：0.0）
Output_PER[1]	OUT	Word	模拟量输出值。（默认值：W#16#0）
Output_PWM[1]	OUT	Bool	脉冲宽度调制的输出值。（默认值：FALSE） 开关时间构成输出值。

参数和类型		数据类型	说明
SetpointLimit_H	OUT	Bool	设定值上限。（默认值：FALSE） 如果 SetpointLimit_H = TRUE，则说明达到设定值的绝对上限 (Setpoint ≥ Config.SetpointUpperLimit)。 设定值限制为 Config.SetpointUpperLimit。
SetpointLimit_L	OUT	Bool	设定值下限。（默认值：FALSE） 如果 SetpointLimit_L = TRUE，则说明达到设定值的绝对下限 (Setpoint ≥ Config.SetpointLowerLimit)。 设定值限制为 Config.SetpointLowerLimit。
InputWarning_H	OUT	Bool	如果 InputWarning_H = TRUE，则说明过程值已达到或超出警告上限。（默认值：FALSE）
InputWarning_L	OUT	Bool	如果 InputWarning_L = TRUE，则说明过程值已达到或低于警告下限。（默认值：FALSE）
State	OUT	Int	PID 控制器的当前操作模式。（默认值：0） 可以使用 Mode 输入参数和 ModeActivate 的上升沿更改工作模式： • State = 0：未激活 • State = 1：预调节 • State = 2：手动精确调节 • State = 3：自动模式 • State = 4：手动模式 • State = 5：通过错误监视替换输出值
Error	OUT	Bool	如果 Error = TRUE，则该周期内至少有一条错误消息未决。（默认值：FALSE） 注：V1.x PID 中的 Error 参数是包含错误代码的 ErrorBits 字段。它现在是一个布尔标记，说明有错误发生。
ErrorBits	OUT	DWord	PID_Compact 指令 ErrorBits 参数表 (页 599)定义未决的错误消息。（默认值：DW#16#0000（无错误））。ErrorBits 具有保持性并在 Reset 或 ErrorAck 的上升沿复位。 注：在 V1.x 中，ErrorBits 参数定义为 Error 参数并且不存在。

[1] 您可以并行使用 Output、Output_PER 和 Output_PWM 参数的输出。

PID_Compact 控制器的操作

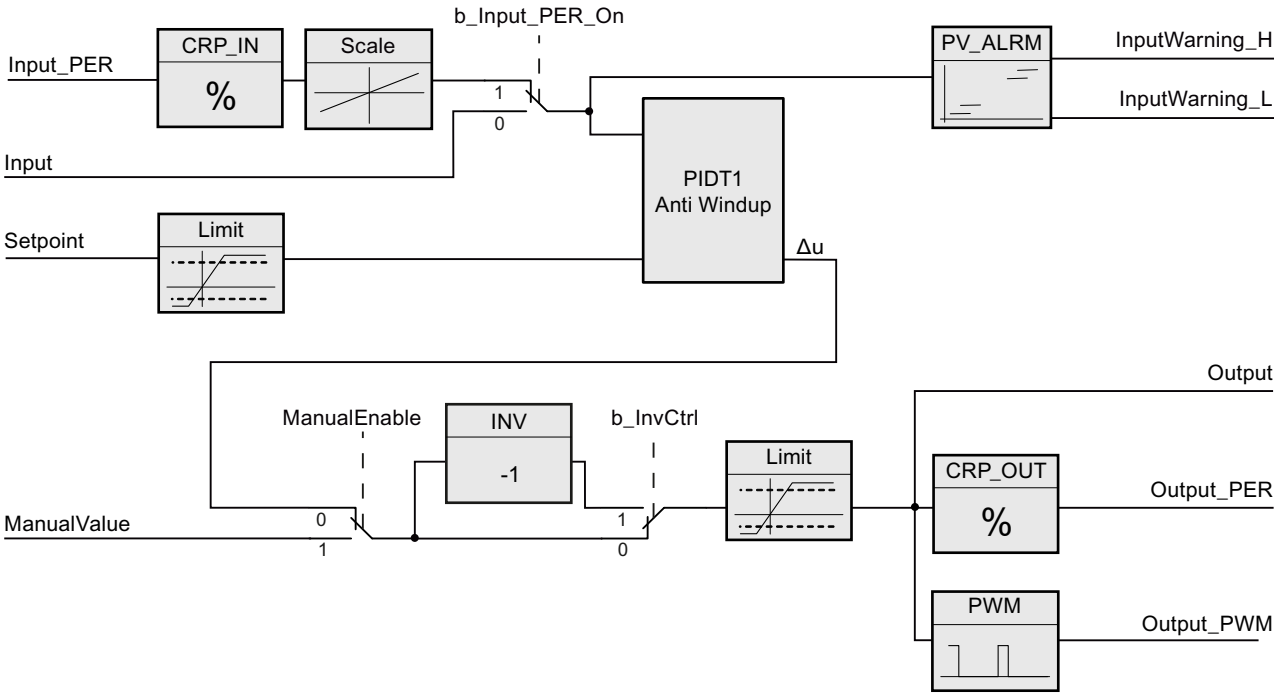


图 10-1　PID_Compact 控制器的操作

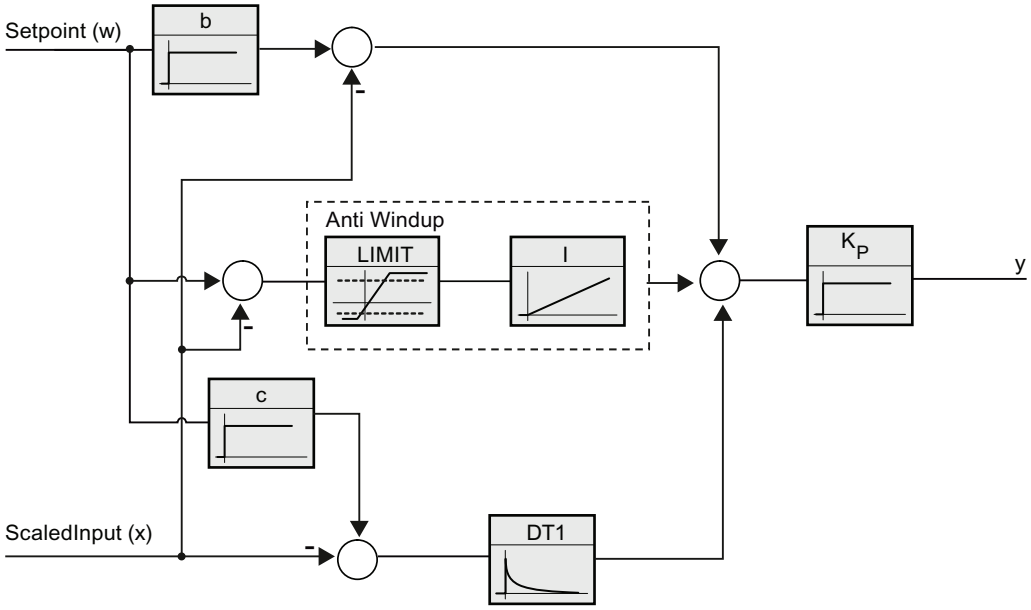


图 10-2　PID_Compact 控制器作为具有抗积分饱和功能的 PIDT1 控制器时的操作

10.2.2.2　PID_Compact 指令过程值限制

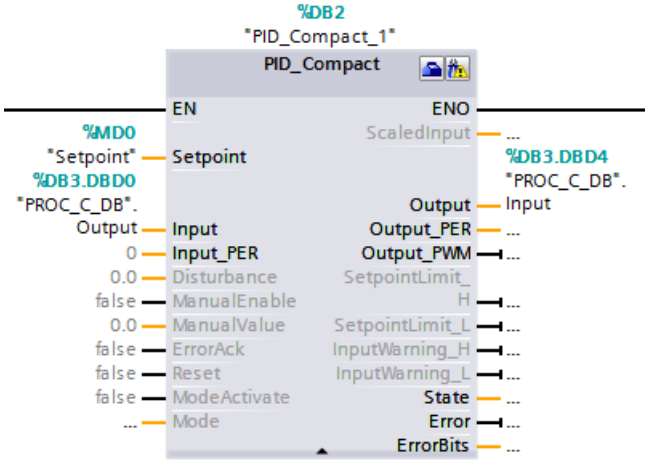
“过程值限制”通常与模拟量输入组合使用，不过，它们也可以用于其它用途。

过程值限制组态有两个功能：

- 为 PID 块上的设定值上限/下限报警输出设定上限和下限
- 设置限值，确保无论设定值如何，过程变量都不会超出或低于这些限值。该组态为过程定义固定限值。

可在 PID 块上使用四个报警输出：

- SetpointLimit_H：设定值超出上限。
- SetpointLimit_L：设定值超出下限。
- InputWarning_H：过程变量超出输入上限。
- InputWarning_L：过程变量超出输入下限。



设置过程值限值时，触发设定值上限和下限报警输出变为 True 的点需设为相同值。例如，使用上述设置时，如果输入的设定值大于 120%，则“SetpointLimit_H”输出会变为 True。此动作对于下限值也一样。如果输入的设定值小于 0%，则“SetpointLimit_L”会变为 True。这时会向程序提供指示，输入的设定值已超出范围。此报警会提示用户重新输入设定值。

如果输入的设定值超出范围，Compact_ID 会自动将过程变量限制在已组态的范围内。例如，过程值上限设为 120%（如上图所示）时，用户仍可输入一个高于 120% 的设定值。过程变

量接近 120% 时，PID 会减少输出并将过程控制在 120% 上限。同理，设定值小于过程值下限时，也会出现同样的操作。PID 不允许过程变量低于下限。凭借此特性，用户可定义正常自动 PID 控制过程中可接受的过程运算。不过，除非 PID 处于自动模式，否则此特性不适用于启动和关断。如果 PID 处于自动模式且设定值和过程变量均小于下限，那么 PID 会尝试将过程控制在已组态的下限处。

10.2.2.3 PID_Compact 指令 ErrorBit 参数

如果存在多个错误未决，则错误代码的值将通过二进制加法显示。例如，显示错误代码 0003 表示错误 0001 和 0002 未决。

表格 10-20 PID_Compact 指令 ErrorBit 参数

ErrorBit (DW#16#...)	说明
0000	无错误
0001 [1, 2]	参数 Input 超出了过程值限值的范围。 Input > Config.InputUpperLimit Input < Config.InputLowerLimit
0002 [2, 3]	参数 Input_PER 的值无效。请检查模拟量输入是否有错误尚未解决。
0004 [4]	精确调节期间出错。无法保持过程值的振荡。
0008 [4]	预调节开始时出错。过程值过于接近设定值。开始精确调节。
0010 [4]	调节期间设定值发生更改。 注意：可在 CancelTuningLevel 变量中设置允许的设定值波动。
0020	精确调节期间不允许预调节。 注意：如果在错误发生前 ActivateRecoverMode = TRUE，则 PID_Compact 保持在精确调节模式。
0080 [4]	预调节期间出错。输出值限值的组态不正确。 检查是否已正确组态输出值的限值以及该限值是否与控制逻辑匹配。
0100 [4]	精确调节期间由于无效参数导致出错。
0200 [2, 3]	参数 Input 的值无效：值的数字格式无效。
0400 [2, 3]	输出值计算失败。检查 PID 参数。
0800 [1, 2]	采样时间错误：循环中断 OB 的采样时间内没有调用 PID_Compact。
1000 [2, 3]	参数 Setpoint 的值无效：值的数字格式无效。

ErrorBit (DW#16#...)	说明
10000	参数 ManualValue 的值无效：值的数字格式无效。 注意：如果在错误发生前 ActivateRecoverMode = TRUE，则 PID_Compact 使用 SubstituteOutput 作为输出值。只要在 ManualValue 参数中分配有效值，PID_Compact 便会将其用作输出值。
20000	变量 SubstituteValue 的值无效：值的数字格式无效。 PID_Compact 使用输出值下限作为输出值。 注意：如果在错误发生之前自动模式已激活，ActivateRecoverMode = TRUE 且错误不再处于未决状态，则 PID_Compact 切换回自动模式。
40000	参数 Disturbance 的值无效：值的数字格式无效。 注意：如果在错误发生前自动模式已激活且 ActivateRecoverMode = FALSE，则 Disturbance 将设置为零。PID_Compact 保持自动模式。 注意：如果在错误发生前预调节或精确调节已激活且 ActivateRecoverMode = TRUE，则 PID_Compact 切换到 Mode 参数中保存的工作模式。如果当前阶段中的干扰对输出值无影响，则不会取消调节。

[1] 注：如果在错误发生前自动模式已激活且 ActivateRecoverMode = TRUE，则 PID_Compact 保持自动模式。

[2] 注：如果在错误发生前预调节或精确调节已激活且 ActivateRecoverMode = TRUE，则 PID_Compact 切换到 Mode 参数中保存的工作模式。

[3] 注：如果在错误发生前自动模式已激活且 ActivateRecoverMode = TRUE，则 PID_Compact 输出组态的替换输出值。当错误不再处于未决状态时，PID_Compact 切换回自动模式。

[4] 注：如果在错误发生前 ActivateRecoverMode = TRUE，则 PID_Compact 取消调节并切换到 Mode 参数中保存的工作模式。

10.2.2.4 PID_Compact 指令的警告参数

PID 控制器有多个未决警告时，将会采用二进制加法显示错误代码的值。 例如，显示错误代码 0003，表示错误 0001 和 0002 处于待决状态。

表格 10-21 PID_Compact 指令 Warning 参数

警告 (DW#16#...)	说明
0000	无未决警告。
0001 [1]	预调整期间未找到拐点。
0002	“运行中调整”期间，强制启动了振动。 （该“警告（Warning）”参数抑制了该警告，仅在用于诊断目的的“内部警告（WarningInternal）”参数中才可见。）
0004 [1]	该整定值被限制为不得大于已经组态的极限值。
0008 [1]	对于选定的这种计算方法，没有必要设置控制器系统的全部属性。 而是采用 TIR.TuneRuleHeat / TIR.TuneRuleCool = 3 计算 PID 的这些参数。
0010	由于 Reset = TRUE 或 ManualEnable = TRUE，因此，运行模式不能更改。
0020	调用 OB 的循环时间决定 PID 算法的采样时间。 采用更短 OB 循环时间改善结果值。
0040 [1]	过程值超过了其中某个警告限值。
0080	“模式（Mode）”中的无效值。 运行模式未切换。
0100 [1]	手动设置值被限制为不得大于控制器输出的极限值。
0200	不支持指定的调整规则。 未计算任何 PID 参数。
1000	输出替代值超过了输出值的极限值，因此，无法达到该输出替代值。

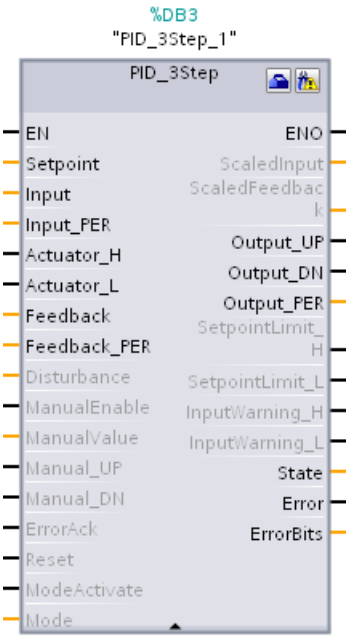
[1] 注： 排除警告原因或者采用有效参数重新执行用户操作后，PID 控制器立即自动删除以下警告： 0001、0004、0008、0040 和 0100。

10.2.3 PID_3Step

10.2.3.1 PID_3Step 指令

PID_3Step 指令用于组态具有自调节功能的 PID 控制器，这样的控制器已针对通过电机控制的阀门和执行器进行过优化。

表格 10-22 PID_3Step 指令

LAD/FBD	SCL	说明
	"PID_3Step_1"(SetpoInt:=_real_in_, Input:=_real_in_, ManualValue:=_real_in_, Feedback:=_real_in_, InputPer:=_word_in_, FeedbackPer:=_word_in_, Disturbance:=_real_in_, ManualEnable:=_bool_in_, ManualUP:=_bool_in_, ManualDN:=_bool_in_, ActuatorH:=_bool_in_, ActuatorL:=_bool_in_, ErrorAck:=_bool_in_, Reset:=_bool_in_, ModeActivate:=_bool_in_, Mode:=_int_in_, ScaledInput=>_real_out_, ScaledFeedback=>_real_out_, ErrorBits=>_dword_out_, OutputPer=>_word_out_, State=>_int_out_, OutputUP=>_bool_out_, OutputDN=>_bool_out_, SetpoIntLimitH=>_bool_out_, SetpoIntLimitL=>_bool_out_, InputWarningH=>_bool_out_, InputWarningL=>_bool_out_, Error=>_bool_out_, ErrorBits=>_dword_out_);	PID_3Step 用于组态具有自调节功能的 PID 控制器，这样的控制器已针对通过电机控制的阀门和执行器进行过优化。它提供两个布尔型输出。 PID_3Step 是具有抗积分饱和功能且对 P 分量和 D 分量加权的 PID T1 控制器。

[1] STEP 7 会在插入指令时自动创建工艺对象和背景数据块。该背景数据块包含工艺对象的参数。

[2] 在 SCL 示例中，“PID_3Step_1”是背景 DB 的名称。

表格 10-23 参数的数据类型

参数和类型		数据类型	说明
Setpoint	IN	Real	PID 控制器在自动模式下的设定值。（默认值：0.0）
Input	IN	Real	用户程序的变量用作过程值的源。（默认值：0.0） 如果正在使用 Input 参数，则必须设置 Config.InputPerOn = FALSE。
Input_PER	IN	Word	模拟量输入用作过程值的源。（默认值：W#16#0） 如果正在使用 Input_PER 参数，则必须设置 Config.InputPerOn = TRUE。
Actuator_H	IN	Bool	上端停止位阀门的数字位置反馈 如果 Actuator_H = TRUE，则阀门处于上端停止位，且不再向此方向移动。（默认值：FALSE）
Actuator_L	IN	Bool	下端停止位阀门的数字位置反馈 如果 Actuator_L = TRUE，则阀门处于下端停止位，且不再向此方向移动。（默认值：FALSE）
Feedback	IN	Real	阀门的位置反馈。（默认值：0.0） 如果正在使用 Feedback 参数，则必须设置 Config.FeedbackPerOn = FALSE。
Feedback_PER	IN	Int	阀门位置的模拟反馈。（默认值：W#16#0） 如果正在使用 Feedback_PER 参数，则必须设置 Config.FeedbackPerOn = TRUE. Feedback_PER 根据以下变量标定： • Config.FeedbackScaling.LowerPointIn • Config.FeedbackScaling.UpperPointIn • Config.FeedbackScaling.LowerPointOut • Config.FeedbackScaling.UpperPointOut
Disturbance	IN	Real	干扰变量或预控制值
ManualEnable	IN	Bool	启用或禁用手动操作模式。（默认值：FALSE）： • FALSE 至 TRUE 沿激活“手动模式”，同时 State = 4，Mode 保持不变。 ManualEnable = TRUE 时，无法利用 ModeActivate 的上升沿或使用调试对话框更改工作模式。 • TRUE 至 FALSE 沿激活 Mode 分配的工作模式。 注：建议您只使用 ModeActivate 更改工作模式。

参数和类型		数据类型	说明
ManualValue	IN	Real	手动操作的过程值。（默认值：0.0） 在手动模式下，可指定阀门的绝对位置。仅当正使用 OutputPer 或者位置反馈可用时，才评估 ManualValue。
ManualUP	IN	Bool	• Manual_UP = TRUE： – 即使使用 Output_PER 或位置反馈，阀门也会打开。如果未到达上端停止位，阀门将不再移动。 – 另请参见 Config.VirtualActuatorLimit • Manual_UP = FALSE： – 使用 Output_PER 或位置反馈会使阀门移动到 ManualValue。否则阀门将不再移动。 注：Manual_UP 和 Manual_DN 同时设为 TRUE 时，阀门不再移动。
ManualDN	IN	Bool	• Manual_DN = TRUE： – 即使使用 Output_PER 或位置反馈，阀门也会打开。如果未到达上端停止位，阀门将不再移动。 – 另请参见 Config.VirtualActuatorLimit • Manual_DN = FALSE： – 使用 Output_PER 或位置反馈会使阀门移动到 ManualValue。否则阀门将不再移动。
ErrorAck	IN	Bool	复位 ErrorBits 和警告输出。FALSE 至 TRUE 沿
Reset	IN	Bool	重新启动控制器。（默认值：FALSE）： • FALSE 至 TRUE 沿： – 切换到“未激活”模式 – 复位 ErrorBits 和警告输出 – 清除积分作用 – 保持 PID 参数 • 只要 Reset = TRUE，则 PID_3Step 便会保持在“未激活”模式 (State = 0)。 • TRUE 至 FALSE 沿： – PID_3Step 切换到保存在 Mode 参数中的工作模式。
ModeActivate	IN	Bool	PID_3Step 切换到保存在 Mode 参数中的模式。FALSE 至 TRUE 沿：
Mode	IN	Int	期望的 PID 模式；在 Mode Activate 输入的上升沿激活。
ScaledInput	OUT	Real	标定的过程值

参数和类型		数据类型	说明
ScaledFeedback	OUT	Real	标定的阀门位置反馈 注：对于无位置反馈的执行器，其位置由 ScaledFeedback 指示并且非常不精确。在这种情况下，ScaledFeedback 只能用于粗略估计当前位置。
Output_UP	OUT	Bool	用于打开阀门的数字输出值。（默认值：FALSE） 如果 Config.OutputPerOn = FALSE，则会使用 Output_UP 参数。
Output_DN	OUT	Bool	用于关闭阀门的数字输出值。（默认值：FALSE） 如果 Config.OutputPerOn = FALSE，则会使用 Output_DN 参数。
Output_PER	OUT	Word	模拟量输出值。 如果 Config.OutputPerOn = TRUE，则会使用 Output_PER 参数。
SetpointLimitH	OUT	Bool	设定值上限。（默认值：FALSE） 如果 SetpointLimitH = TRUE，则说明达到设定值的绝对上限 (Setpoint ≥ Config.SetpointUpperLimit)。 注：设定值限制为 (Setpoint ≥ Config.SetpointUpperLimit)。
SetpointLimitL	OUT	Bool	设定值下限。（默认值：FALSE） 如果 SetpointLimitL = TRUE，则说明达到设定值的绝对下限 (Setpoint ≥ Config.SetpointLowerLimit)。 注：设定值限制为 (Setpoint ≥ Config.SetpointLowerLimit)。
InputWarningH	OUT	Bool	如果 InputWarningH = TRUE，则说明输入值已达到或超出警告上限。（默认值：FALSE）
InputWarningL	OUT	Bool	如果 InputWarningL = TRUE，则说明输入值已达到或超出警告下限。（默认值：FALSE)

参数和类型		数据类型	说明
State	OUT	Int	PID 控制器的当前操作模式。（默认值：0） 可以使用 Mode 输入参数和 ModeActivate: 的上升沿更改工作模式 • State = 0：未激活 • State = 1：预调节 • State = 2：手动精确调节 • State = 3：自动模式 • State = 4：手动模式 • State = 5：替换输出值方式 • State = 6：切换时间测量 • State = 7：错误监视 • State = 8：出现错误监控时，替换输出值的方式\ • State = 10：无停止位信号的手动模式
Error	OUT	Bool	如果 Error = TRUE，则至少存在一个错误消息未决。（默认值：FALSE） 注：V1.x PID 中的 Error 参数是包含错误代码的 ErrorBits 字段。它现在是一个布尔标记，说明有错误发生。
ErrorBits	OUT	DWord	PID_3Step 指令 ErrorBits 参数表 (页 609)定义未决的错误消息。（默认值：DW#16#0000（无错误））。ErrorBits 具有保持性并在 Reset 或 ErrorAck 的上升沿复位。 注：在 V1.x 中，ErrorBits 参数定义为 Error 参数并且不存在。

PID_3Step 控制器的操作

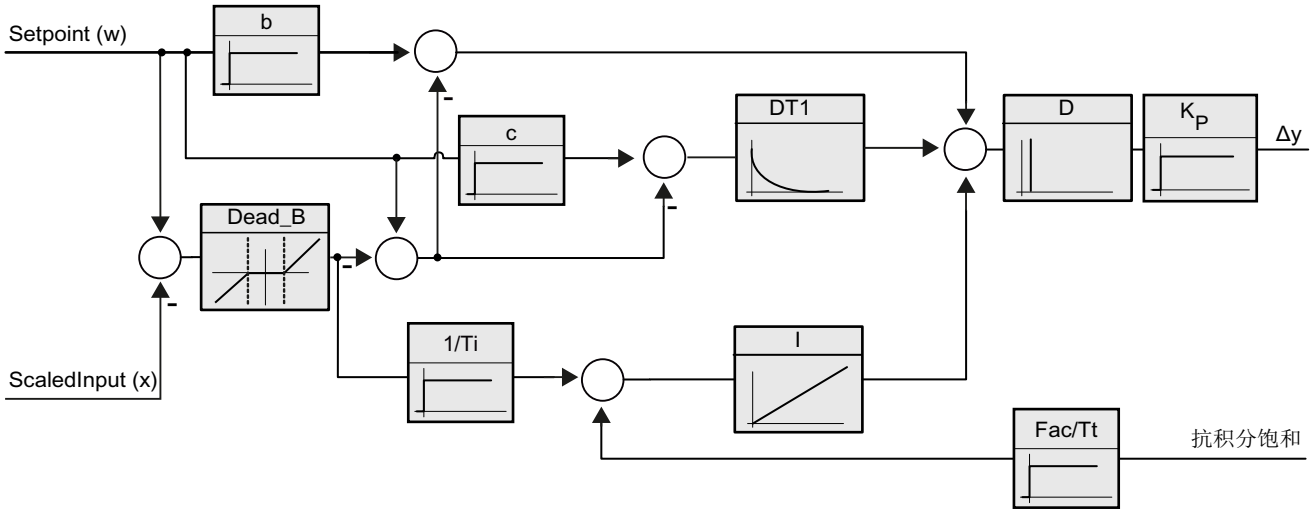


图 10-3　PID_3Step 控制器作为具有抗积分饱和功能的 PID T1 控制器时的操作

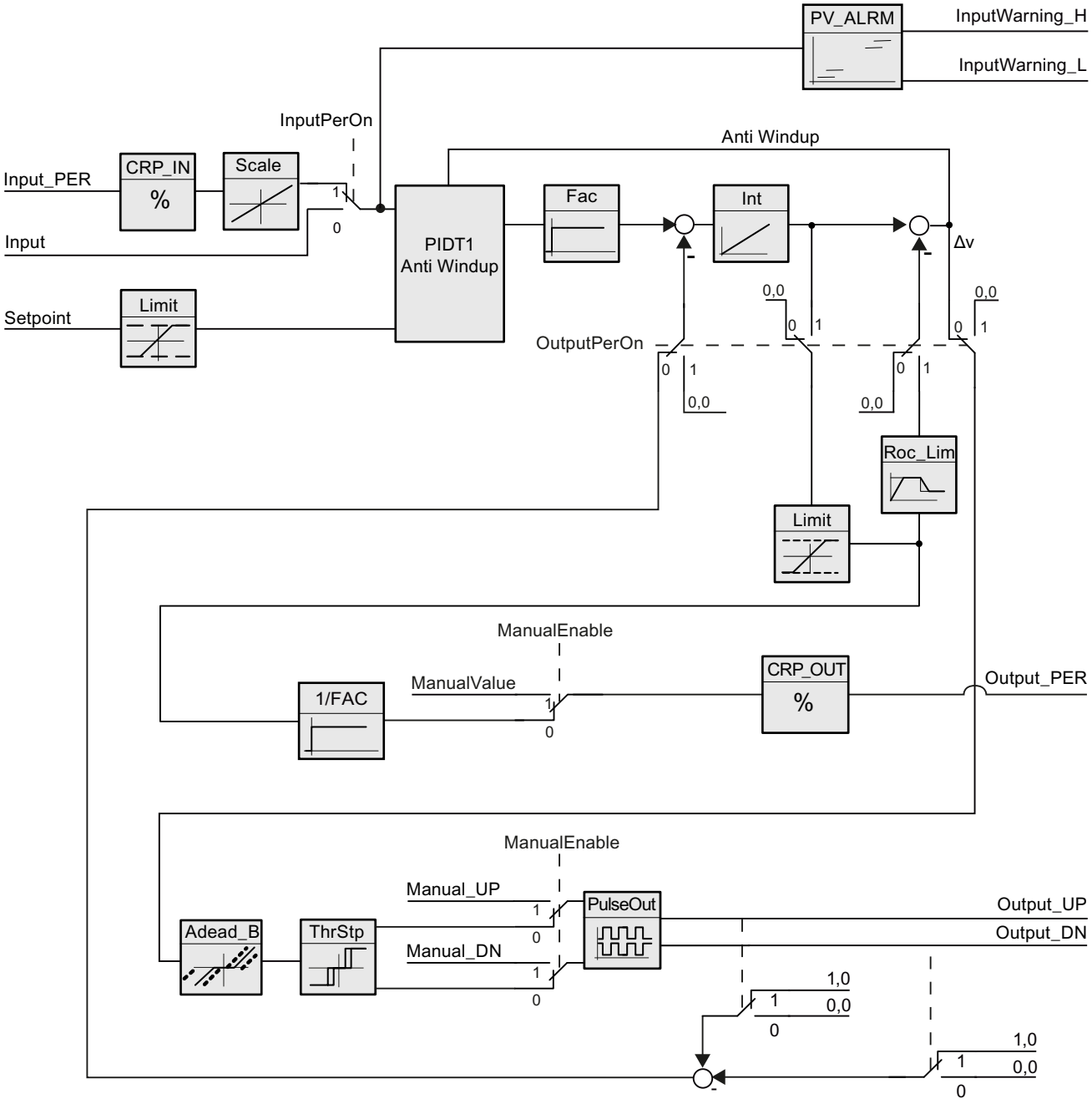


图 10-4　无位置反馈的 PID_3Step 控制器的操作

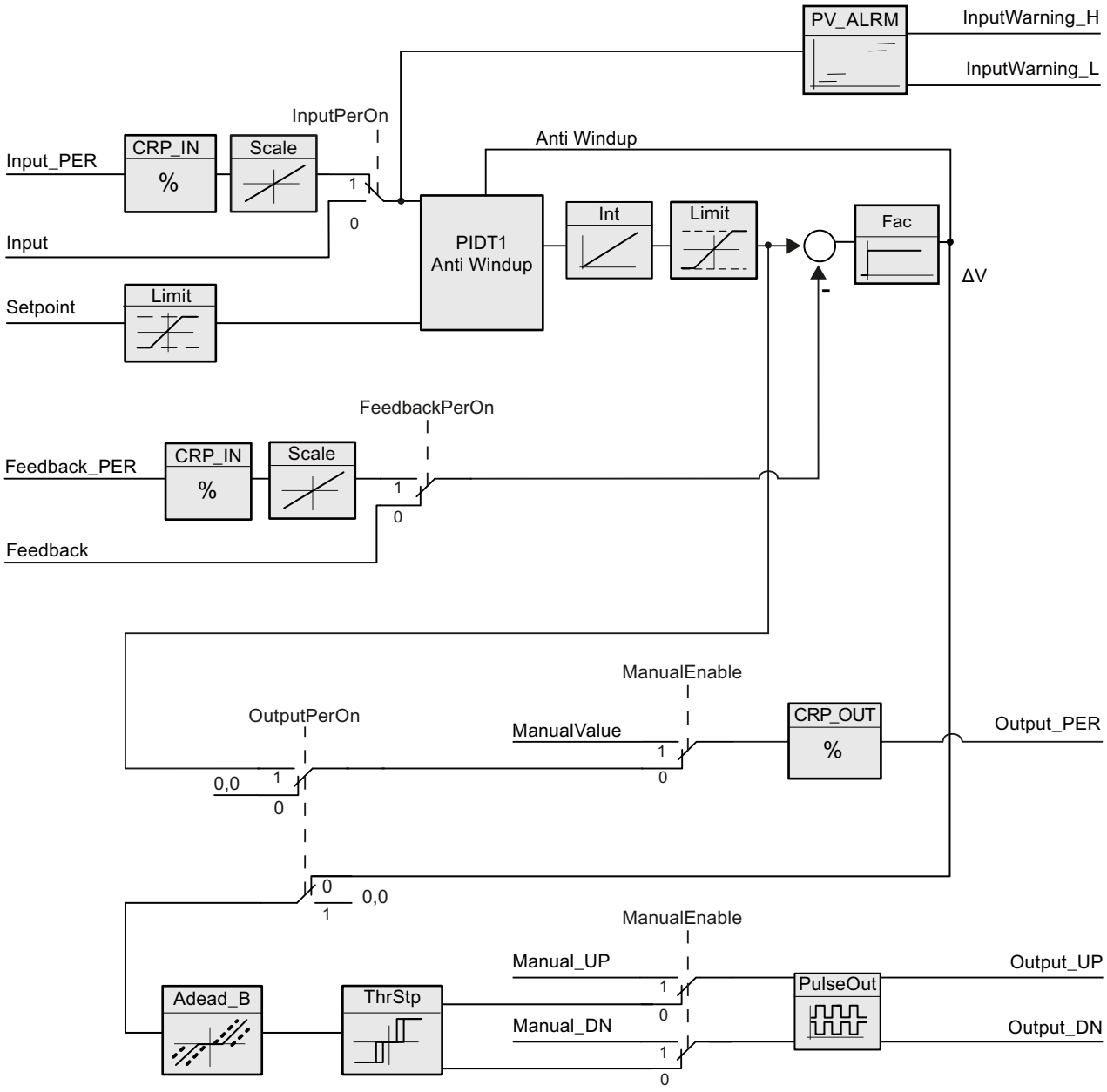


图 10-5　启用了位置反馈的 PID_3Step 控制器的操作

10.2.3.2 PID_3Step 指令的 ErrorBit 参数

如果存在多个错误未决，则错误代码的值将通过二进制加法显示。例如，显示错误代码 0003 表示错误 0001 和 0002 未决。

表格 10-24 PID_3STEP 指令的 ErrorBit 参数

ErrorBit (DW#16#...)	说明
0000	无错误
0001 [1, 2]	参数 Input 超出了过程值限值的范围。 Input > Config.InputUpperLimit Input < Config.InputLowerLimit
0002 [2, 3]	参数 Input_PER 的值无效。请检查模拟量输入是否有错误尚未解决。
0004 [4]	精确调节期间出错。无法保持过程值的振荡。
0010 [4]	调节期间设定值发生更改。 注意：可在 CancelTuningLevel 变量中设置允许的设定值波动。
0020	精确调节期间不允许预调节。 注意：如果在错误发生前 ActivateRecoverMode = TRUE，则 PID_3Step 保持在精确调节模式。
0080 [4]	预调节期间出错。输出值限值的组态不正确。 检查是否已正确组态输出值的限值以及该限值是否与控制逻辑匹配。
0100 [4]	精确调节期间由于无效参数导致出错。
0200 [2, 3]	参数 Input 的值无效：值的数字格式无效。
0400 [2, 3]	计算输出值失败。检查 PID 参数。
0800 [1, 2]	采样时间错误：循环中断 OB 的采样时间内没有调用 PID_3Step。
1000 [2, 3]	参数 Setpoint 的值无效：值的数字格式无效。
2000 [1, 2, 5]	参数 Feedback_PER 的值无效。 请检查模拟量输入是否有错误尚未解决。
4000 [1, 2, 5]	参数 Feedback 的值无效：值的数字格式无效。

ErrorBit (DW#16#...)	说明
8000 [1, 2]	数字位置反馈期间出错。Actuator_H = TRUE 且 Actuator_L = TRUE。 执行器无法移动到替代输出值，并且将保持在当前位置。在该状态下不能使用手动模式。 为了从此状态移动执行器，必须取消激活“执行器停止位” (Config.ActuatorEndStopOn = FALSE) 或者切换到无停止位信号的手动模式 (Mode = 10)。
10000	参数 ManualValue 的值无效：值的数字格式无效。 执行器无法移动到手动值，并且将保持当前位置。 在 ManualValue 中分配一个有效值或者在手动模式下通过 Manual_UP 和 Manual_DN 移动执行器。
20000	变量 SavePosition 的值无效：值的数字格式无效。 执行器无法移动到替代输出值，并且将保持在当前位置。
40000	参数 Disturbance 的值无效：值的数字格式无效。 注意：如果在错误发生前自动模式已激活且 ActivateRecoverMode = FALSE，则 Disturbance 将设置为零。PID_3Step 保持自动模式。 注意：如果在错误发生前预调节或精确调节已激活且 ActivateRecoverMode = TRUE，则 PID_3Step 切换到 Mode 参数中保存的工作模式。如果当前阶段中的干扰对输出值无影响，则不会取消调节。 转换时间测量期间错误没有影响。

1 注：如果在错误发生前自动模式已激活且 ActivateRecoverMode = TRUE，则 PID_3Step 保持自动模式。

2 注：如果在错误发生前已激活预调节、精确调节或转换时间测量模式，并且 ActivateRecoverMode = TRUE，则 PID_3Step 将切换到 Mode 参数中保存的工作模式。

3 注：如果在错误发生之前自动模式已激活并且 ActivateRecoverMode = TRUE，则 PID_3Step 将切换到“在监视错误的同时逼近替代输出值”或“错误监视”模式。当错误不再处于未决状态时，PID_3Step 切换回自动模式。

4 注：如果在错误发生前 ActivateRecoverMode = TRUE，则 PID_3Step 取消调节并切换到 Mode 参数中保存的工作模式。

5 执行器无法移动到替代输出值，并且将保持当前位置。在手动模式下，仅可通过 Manual_UP 和 Manual_DN 更改执行器的位置，而不可通过 ManualValue 更改。

10.2.3.3 PID_3Step 指令的警告参数

PID 控制器有多个未决警告时，将会采用二进制加法显示错误代码的值。 例如，显示错误代码 0003，表示错误 0001 和 0002 处于待决状态。

表格 10-25 PID_Compact 指令 Warning 参数

警告 (DW#16#...)	说明
0000	无未决警告。
0001 [1]	预调整期间未找到拐点。
0002	“运行中调整”期间，强制启动了振动。 （该“警告（Warning）”参数抑制了该警告，仅在用于诊断目的的“内部警告（WarningInternal）”参数中才可见。）
0004 [1]	该整定值被限制为不得大于已经组态的极限值。
0008 [1]	对于选定的这种计算方法，没有必要设置控制器系统的全部属性。 而是采用 TIR.TuneRuleHeat / TIR.TuneRuleCool = 3 计算 PID 的这些参数。
0010	由于 Reset = TRUE 或 ManualEnable = TRUE，因此，运行模式不能更改。
0020	调用 OB 的循环时间决定 PID 算法的采样时间。 采用更短 OB 循环时间改善结果值。
0040 [1]	过程值超过了其中某个警告限值。
0080	“模式（Mode）”中的无效值。 运行模式未切换。
0100 [1]	手动设置值被限制为不得大于控制器输出的极限值。
0200	不支持指定的调整规则。 未计算任何 PID 参数。
1000	输出替代值超过了输出值的极限值，因此，无法达到该输出替代值。

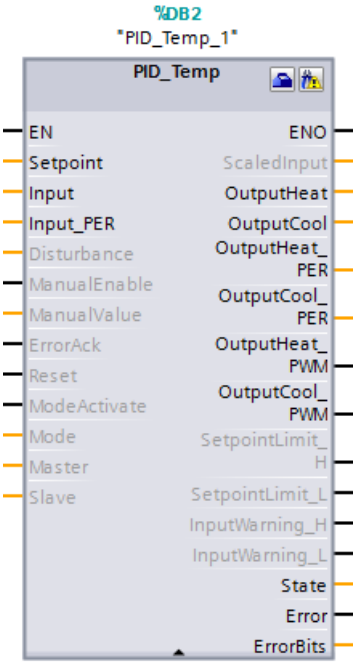
[1] 注： 排除警告原因或者采用有效参数重新执行用户操作后，PID 控制器立即自动删除以下警告： 0001、0004、0008、0040 和 0100。

10.2.4 PID_Temp

10.2.4.1 PID_Temp 指令

PID_Temp 指令提供一个通用的 PID 控制器，可用于处理温度控制的特定需求。

表格 10-26 PID_Temp 指令

LAD/FBD	SCL	说明
	"PID_Temp_1"(Setpoint:=_real_in_, Input:=_real_in_, Input_PER:=_int_in_, Disturbance:=_real_in_, ManualEnable:=_bool_in_, ManualValue:=_real_in_, ErrorAck:=_bool_in_, Reset:=_bool_in_, ModeActivate:=_bool_in_, Mode:=_int_in_, Master:=_dword_in Save:=_dword_in ScaledInput=>_real_out_, OutputHeat=>_real_out_, OutputCool=>_real_out_, OutputHeat_PER=>_int_out_, OutputCool_PER=>_int_out_, OutputHeat_PWM=>_bool_out_, OutputCool_PWM=>_bool_out_, SetpointLimit_H=>_bool_out_, SetpointLimit_L=>_bool_out_, InputWarning_H=>_bool_out_, InputWarning_L=>_bool_out_, State=>_int_out_, Error=>_bool_out_, ErrorBits=>_dword_out_);	PID_Temp 具有以下功能： ● 使用不同执行器加热或冷却此过程 ● 用于处理温度过程的集成式自动调节功能 ● 级联处理取决于同一执行器的多个温度

[1] STEP 7 会在插入指令时自动创建工艺对象和背景数据块。该背景数据块包含工艺对象的参数。

[2] 在 SCL 示例中，“PID_Temp_1”是背景 DB 的名称。

表格 10-27 参数的数据类型

参数和类型		数据类型	说明
Setpoint	IN	Real	PID 控制器在自动模式下的设定值。（默认值：0.0）
Input	IN	Real	用户程序的变量用作过程值的源。（默认值：0.0） 如果正在使用 Input 参数，则必须设置 Config.InputPerOn = FALSE。
Input_PER	IN	Int	模拟量输入用作过程值的源。（默认值：0） 如果正在使用 Input_PER 参数，则必须设置 Config.InputPerOn = TRUE。
Disturbance	IN	Real	干扰变量或预控制值
ManualEnable	IN	Bool	启用或禁用手动操作模式。（默认值：FALSE）： • FALSE 至 TRUE 沿激活“手动模式”，State = 4，Mode 保持不变。 ManualEnable = TRUE 时，无法利用 ModeActivate 的上升沿或使用调试对话框更改工作模式。 • TRUE 至 FALSE 沿激活 Mode 分配的工作模式。 注：建议您只使用 ModeActivate 更改工作模式。
ManualValue	IN	Real	手动操作的输出值。（默认值：0.0） 可以使用从 Config.OutputLowerLimit 到 Config.OutputUpperLimit 的值。
ErrorAck	IN	Bool	使用 FALSE 至 TRUE 沿复位 ErrorBits 和警告输出。（默认值：FALSE）
Reset	IN	Bool	重新启动控制器。（默认值：FALSE）： • FALSE 至 TRUE 沿： – 切换到“未激活”模式 – 复位 ErrorBits 和警告输出 – 清除积分作用 – 保持 PID 参数 • 只要 Reset = TRUE，则 PID_Temp 便会保持在“未激活”模式 (State = 0)。 • TRUE 至 FALSE 沿： – PID_Temp 切换到保存在 Mode 参数中的工作模式。
ModeActivate	IN	Bool	通过 FALSE 至 TRUE 沿，PID_Temp 切换到保存在 Mode 参数中的操作模式。（默认值：FALSE）

参数和类型		数据类型	说明
Mode	IN/OUT	Int	在 Mode Activate 输入的上升沿激活。 操作模式选择（默认值：0.0）： • Mode = 0：未激活 • Mode = 1：预调节 • Mode = 2：精确调节 • Mode = 3：自动模式 • Mode = 4：手动模式 “带错误监视的替代输出值”(State = 5)。这无法由用户来激活；只是一种自动错误响应。
Master	IN/OUT	DWord	到主站的级联连接（AntiWindUp 和调节条件）。（默认值：DW#16#0000） • 位 0 - 15：未在 PID_Temp 指令中使用 • 位 16 - 23：限值计数器：如果此值达到了限制，从站会递增该值。针对抗积分饱和功能处理的界限内的从站数（参见 Config.Cascade.AntiWindUpMode 参数）。 • 位 24：IsAutomatic：如果此控制器的所有从站均处于自动模式且被处理为用于检查级联中的调节条件，则此位设置为“1”。此位与 AllSlaveAutomaticState 参数的作用相同。 • 位 25：“IsReplacement 设定值”：如果此控制器一个从站的“替换设定值”已激活且被处理为用于检查级联中的调节条件，则此位设置为“1”。反向值存储在 NoSlaveReplacementSetpoint 参数中。
Slave	IN/OUT	DWord	
ScaledInput	OUT	Real	标定的过程值。（默认值：0.0）
OutputHeat[1]	OUT	Real	REAL 格式的加热输出值。（默认值：0.0） 此输出值采用 Config.Output.Heat.Select 参数进行计算，与输出选择无关。
OutputCool[1]	OUT	Real	REAL 格式的冷却输出值。（默认值：0.0） 此输出值采用 Config.Output.Cool.Select 参数进行计算，与输出选择无关。
OutputHeat_PER[1]	OUT	Int	外设值格式的加热输出值（默认值：0） 只有使用 Config.Output.Heat.Select = 2 参数选择此输出值时，才会计算此输出值。如未选择，此输出始终为“0”。
OutputCool_PER[1]	OUT	Int	外设值格式的冷却输出值（默认值：0） 只有使用 Config.Output.Cool.Select = 2 参数选择此输出值时，才会计算此输出值。如未选择，此输出始终为“0”。

参数和类型		数据类型	说明
OutputHeat_PWM[1]	OUT	Bool	加热过程的脉宽调制输出值。（默认值：FALSE） 只有使用 Config.Output.Heat.Select = 1（默认值）参数选择此输出值时，才会计算此输出值。如未选择，此输出始终为 FALSE。
OutputCool_PWM[1]	OUT	Bool	冷却过程的脉宽调制输出值。（默认值：FALSE） 只有使用 Config.Output.Cool.Select = 1（默认值）参数选择此输出值时，才会计算此输出值。如未选择，此输出始终为 FALSE。
SetpointLimit_H	OUT	Bool	设定值上限。（默认值：FALSE） 如果 SetpointLimit_H = TRUE，则说明达到设定值的绝对上限 (Setpoint ≥ Config.SetpointUpperLimit)。 设定值限制为 Config.SetpointUpperLimit。
SetpointLimit_L	OUT	Bool	设定值下限。（默认值：FALSE） 如果 SetpointLimit_L = TRUE，则说明达到设定值的绝对下限 (Setpoint ≥ Config.SetpointLowerLimit)。 设定值限制为 Config.SetpointLowerLimit。
InputWarning_H	OUT	Bool	如果 InputWarning_H = TRUE，则说明过程值已达到或超出警告上限。（默认值：FALSE）
InputWarning_L	OUT	Bool	如果 InputWarning_L = TRUE，则说明过程值已达到或低于警告下限。（默认值：FALSE）
State	OUT	Int	PID 控制器的当前操作模式。（默认值：0） 可以使用 Mode 输入参数和 ModeActivate 的上升沿更改工作模式： • State = 0：未激活 • State = 1：预调节 • State = 2：精确调节 • State = 3：自动模式 • State = 4：手动模式 • State = 5：通过错误监视替换输出值
Error	OUT	Bool	如果 Error = TRUE，则该周期内至少有一条错误消息未决。（默认值：FALSE） 注：V1.x PID 中的 Error 参数是包含错误代码的 ErrorBits 字段。它现在是一个布尔标记，说明有错误发生。

参数和类型		数据类型	说明
ErrorBits	OUT	DWord	PID_Temp 指令，ErrorBits 参数表 (页 622)定义未决的错误消息。（默认值：DW#16#0000（无错误））。ErrorBits 具有保持性并在 Reset 或 ErrorAck 的上升沿复位。 注：在 V1.x 中，ErrorBits 参数定义为 Error 参数并且不存在。
Warning	OUT	DWord	PID_Temp 指令，Warning 参数表 (页 624)定义未决的用户相关警告消息。（默认值：DW#16#0000（无警告））。
WarningInternal	OUT	DWord	PID_Temp 指令，WarningInternal 参数表定义未决的内部警告消息（包括所有警告）。（默认值：DW#16#0000（无内部警告））。

[1] 您可以并行使用 Output、Output_PER 和 Output_PWM 参数的输出。

PID_Temp 控制器的操作

选择加热和/或冷却控件

用户必须首先选择除参数“ActivateCooling”中的加热输出外，是否还需要冷却设备。 然后必须定义是要在参数“AdvancedCooling”中使用两个 PID 参数集（高级模式）还是仅使用一个 PID 参数集和一个额外的加热/冷却系数。

使用 CoolFactor

如果希望应用加热/冷却系数，必须手动定义该值。 必须根据应用程序中的技术数据（执行器的比例增益比率（例如执行器的最大加热和冷却功率的比率））确定该值，并将其分配给参数“CoolFactor”。 加热/冷却系数 2.0 表示加热设备的影响力是冷却设备的两倍。 如果使用冷却系数，PID_Temp 将计算输出信号，并根据其符号，将输出信号乘以加热/冷却系数（当符号为负时）或不乘以加热/冷却系数（符号为正时）。

使用两个 PID 参数集

在调试期间，可以自动检测用于加热和冷却的不同 PID 参数集。 与使用加热/冷却系数相比，这样可以提高控制性能，因为除不同的比例增益外，还可以考虑两个参数集的不同延时时间。 但缺点是这要花费更多时间来进行调节。 如果激活了 PID 参数切换 (Config.AdvancedCooling = TRUE)，PID_Temp 控制器将以“自动模式”检测（控制已激活），如果这时需要加热或冷却，将使用 PID 参数集进行控制。

ControlZone

使用 PID_Temp 控制器，可以在参数“ControlZone”中为每个参数集定义一个控制区。 如果控制偏差（设定值 – 输入）在控制区内，PID_Temp 将使用 PID 算法来计算输出信号。 但如果控制偏差超出了定义的范围，输出将设置为最大加热或最大冷却输出值（冷却输出被激

活）/最大加热输出值（冷却输出被禁用）。 用户可以使用此功能更快地达到所需的设定值，特别是对于温度变化较慢的初始加热过程。

DeadZone

通过“DeadZone”参数，可以定义 PID 算法忽略的加热和冷却控制偏差的宽度。 这意味着此范围内的控制偏差将被抑制，PID_Temp 控制器将类似于设定值，并且过程值相同。 因此，可以减少控制器对设定值的不必要干预，并节约执行器。 如果要应用 DeadZone，则必须手动定义该值。 自动调节功能不会自动设置 DeadZone 值。 对于不制冷的加热控制器或使用 CoolFactor 的加热/冷却控制器，DeadZone 是对称的（在 -Retain.CtrlParams.Heat.DeadZone 和 +Retain.CtrlParams.Heat.DeadZone 之间）。 对于使用两个 PID 参数集的加热/冷却控制器，DeadZone 可以是对称的（在 -Retain.CtrlParams.Cool.DeadZone 和 +Retain.CtrlParams.Heat.DeadZone 之间）。

PID_Temp 控制器操作

以下方框图说明了 PID_Temp 指令的标准和级联操作：

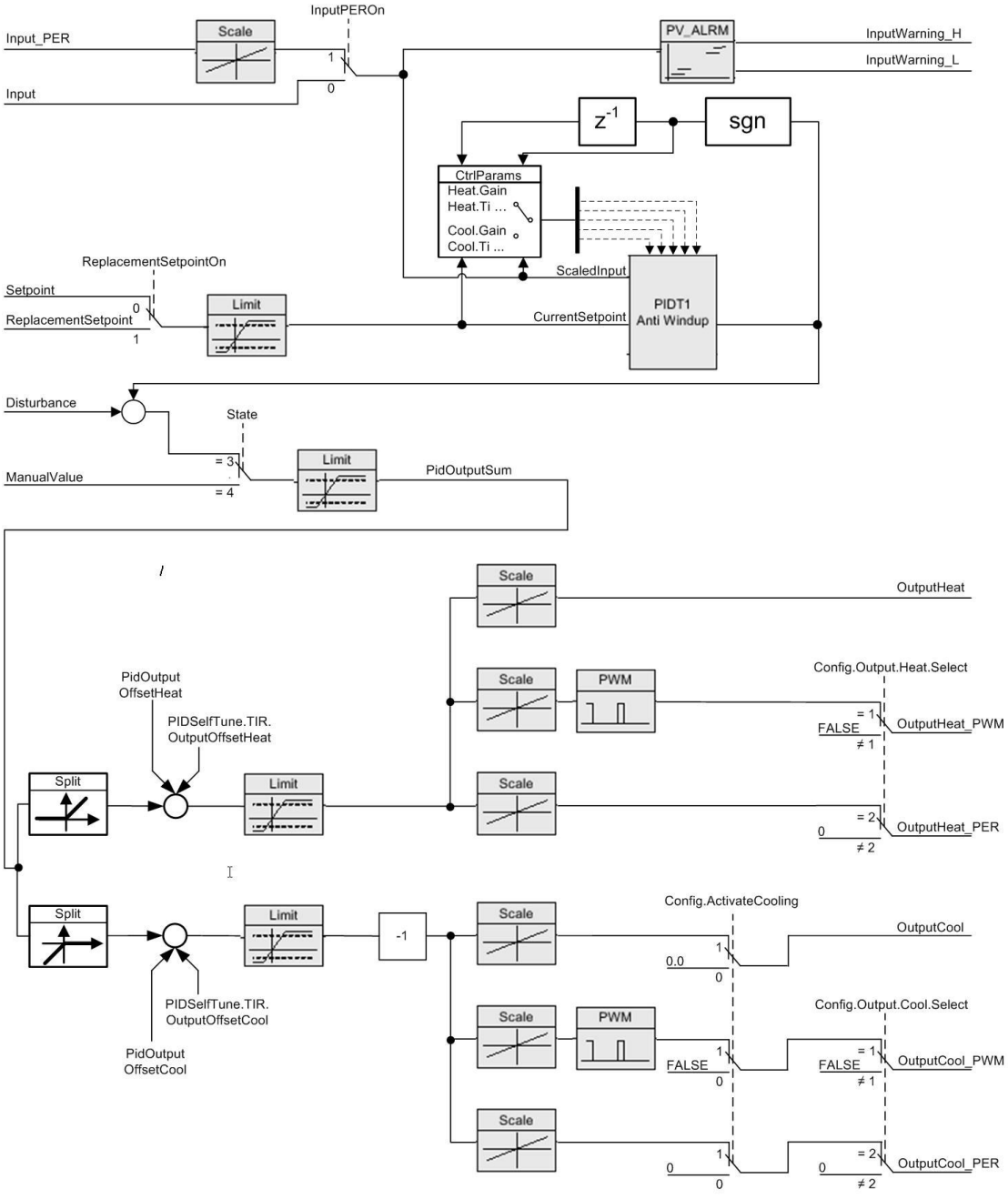


图 10-6　PID_Temp_Operation_Block_Diagram

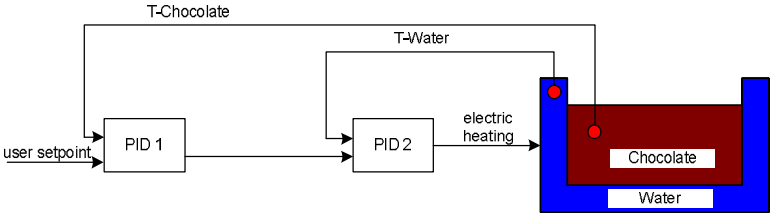


图 10-7 PID_Temp_Cascade_Operation_Block_Diagram

级联控制器

可以级联温度 PID 控制器来处理多个依赖相同执行器的温度。

调用顺序

必须在同一个 OB 周期内调用级联的 PID 控制器。首先必须调用主站，然后调用控制信号流中的下一个从站，最后调用级联中的最后一个从站。PID_Temp 指令不自动检查调用顺序。

通信连接

在级联控制器时，必须连接主站和从站，使其能够互相共享信息。必须沿信号流方向将从站的“Master”IN/OUT 参数与其主站的“Slave”IN/OUT 参数相连接。

这样会在级联中显示一个 PID_Temp 控制器连接，并且此级联具有两个子级联：“PID_Temp1”提供设定值。组态将“PID_Temp2”、“PID_Temp3”、“PID_Temp5”、“PID_Temp6”和“PID_Temp8”的输出连接到过程：

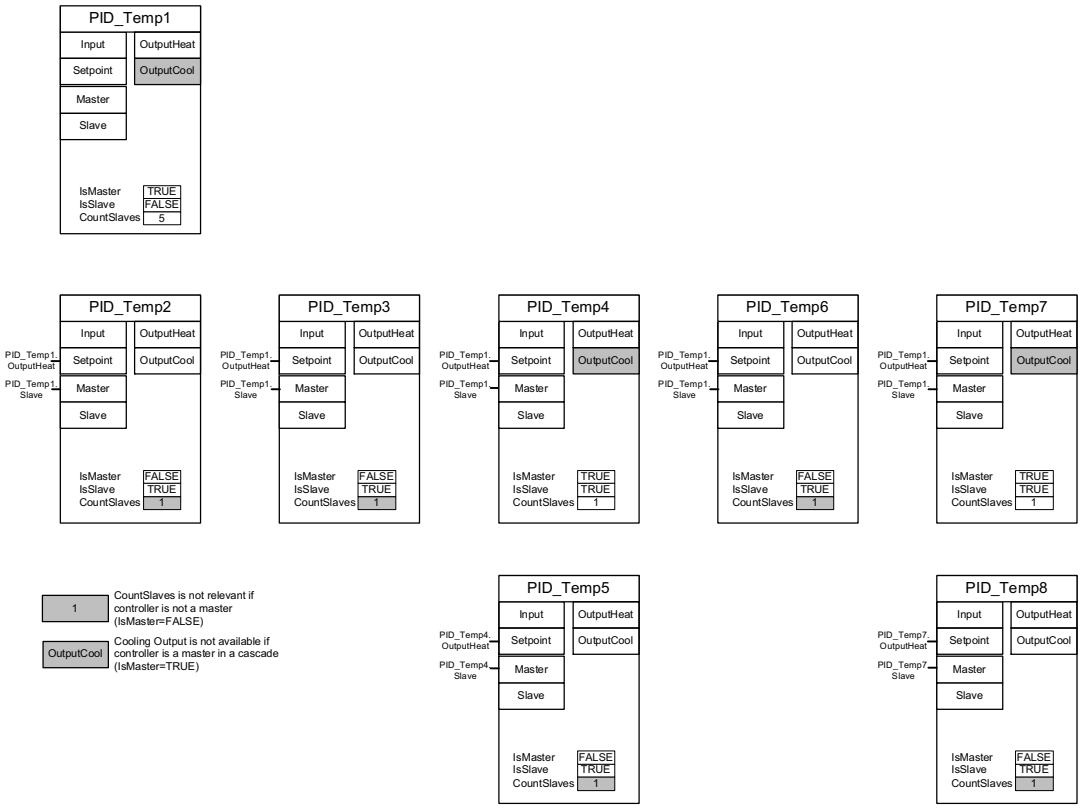


图 10-8 PID_Temp_Cascading_communication_connection

替换设定值

PID_Temp 指令在“ReplacementSetpoint”参数中提供另一个设定值输入，用户可以通过设置参数 “ReplacementSetpointOn” = TRUE 将此参数激活。在调试或调节从站控制器期间，可以使用“ReplacementSetpoint”作为设定值输入，而不必断开主站与从站之间的输出-到-设定值连接。此连接对于级联的正常运行非常重要。

通过这种方式，不必更改程序并下载即可将从站与其主站暂时分开。用户只需激活“ReplacementSetpoint”并在完成后将其重新禁用。如果可以在“CurrentSetpoint”参数中看到设定值，则该值对 PID 算法生效。.

自动调节

级联主站控制器的自动调节必须符合以下要求：

- 从其内部从站到第一个主站都已进行过调试。
- 主站的所有从站必须处于“自动模式”。
- 主站的输出必须是从站的设定值。

PID_Temp 指令将为级联中的自动调节提供以下支持：

- 如果开始自动调节主站控制器，主站将检查是否所有从站均处于“自动模式”，并检查是否为所有从站禁用了“替换设定值”功能 (“ReplacementSetpointOn” = FALSE)。如果不符合这些条件，则无法自动调节主站。主站取消调节，进入“未激活”模式（如果“ActivateRecoverMode” = FALSE），或者恢复到 “Mode” 参数中存储的模式（如果“ActivateRecoverMode” = TRUE）。主站显示错误消息 200000hex（“级联中的主站发生错误。从站不处于自动模式或者启用了替换设定值，阻止了主站的调节。”）。
- 当所有从站均处于“自动模式”时，系统将设置参数 “AllSlaveAutomaticState” = TRUE。用户可以在自己的程序中应用此参数，或者本地化错误 200000hex 的原因。
- 当为所有从站禁用“ReplacementSetpoint”时，系统将设置参数 “NoSlaveReplacementSetpoint” = TRUE。用户可以在自己的程序中应用此参数，或者本地化错误 200000hex 的原因。

在使用 PID_Temp 指令调试对话框时，用户可以得到进一步的级联调节支持 (页 645)。

操作模式与错误处理

PID_Temp 控制器不允许其主站或从站切换操作模式。这意味着当从站发生错误时，级联内的主站仍保持其当前模式。如果有两个或更多并行从站使用此主站控制器运行，这就是一个优点，一条链中发生错误不会导致并行链关闭。

同样，如果主站发生错误，级联内的从站仍保持其当前操作模式。但从站的进一步操作取决于主站的组态，因为从站的设定值是主站的输出。这意味着如果使用 “ActivateRecoverMode” = TRUE 组态主站并且发生错误，主站将输出上一个有效值或一个替换输出值作为从站的设定值。如果使用 “ActivateRecoverMode” = FALSE 组态主站，主站将切换到“未激活模式”并将所有输出设置为“0.0”，让从站使用“0.0”作为其设定值。

因为只有从站控制器能直接访问执行器，并且在主站发生错误时从站仍保持其操作模式，所以能够避免过程受损。例如，对于塑料加工设备，从站停止工作，关闭执行器，并允许塑料在设备内部单独硬化是非常致命的错误，因为主站控制器有错误。

抗积分饱和

级联中的从站从其主站的输出获取其设定值。如果从站达到自己的输出限制，同时主站仍看到控制偏差（设定值 – 输入），主站将冻结或减少其积分贡献以防止所谓的“积分饱和”。在发生“积分饱和”时，主站将其积分贡献增加到了一个非常大的值，要使控制器能够重新正常响应，必须首先降低该值。此类“积分饱和”会对控制的动态性产生负面影响。PID_Temp 通过组态主站控制器的参数“Config.Cascade.AntiWindUpMode”，提供了在级联中防止这种影响的方法：

值	说明
0	禁用抗积分饱和功能。
1	将主站控制器的积分贡献比率从“界限内的从站数”减少到“现有从站数”（参数“CountSlaves”）。
2	在一个从站到达其限制后，立即冻结主站的积分贡献。仅当 "Config.Cascade.IsMaster" = TRUE 时适用。

10.2.4.2 PID_温度错误位参数

如果 PID 控制器存在多个未决警告，则错误代码的值将通过二进制加法显示。例如，显示错误代码 0003 表示错误 0001 和 0002 未决。

表格 10-28 PID_Temp 指令 ErrorBit 参数

ErrorBit (DW#16#...)	说明
0000	无错误
0001 [1, 2]	参数 Input 超出了过程值限值的范围。 Input > Config.InputUpperLimit Input < Config.InputLowerLimit
0002 [2, 3]	参数 Input_PER 的值无效。请检查模拟量输入是否有错误尚未解决。
0004 [4]	精确调节期间出错。无法保持过程值的振荡。
0008 [4]	预调节开始时出错。过程值过于接近设定值。开始精确调节。
0010 [4]	调节期间设定值发生更改。 注：可在 CancelTuningLevel 变量中设置允许的设定值波动。
0020	精确调节期间不允许预调节。 注：如果在错误发生前 ActivateRecoverMode = TRUE，则 PID_Temp 保持在精确调节模式。

ErrorBit (DW#16#...)	说明
0040 [4]	预调节期间出错。冷却不会减少过程值。
0080 [4]	预调节期间出错。输出值限值的组态不正确。 检查是否已正确组态输出值的限值以及该限值是否与控制逻辑匹配。
0100 [4]	精确调节期间由于无效参数导致出错。
0200 [2, 3]	参数 Input 的值无效：值的数字格式无效。
0400 [2, 3]	输出值计算失败。检查 PID 参数。
0800 [1, 2]	采样时间错误：循环中断 OB 的采样时间内没有调用 PID_Temp。
1000 [2, 3]	参数 Setpoint 的值无效：值的数字格式无效。
10000	参数 ManualValue 的值无效：值的数字格式无效。 注：如果在错误发生前 ActivateRecoverMode = TRUE，则 PID_Temp 使用 SubstituteOutput 作为输出值。只要在 ManualValue 参数中分配有效值，PID_Temp 便会将其用作输出值。
20000	变量 SubstituteValue 的值无效：值的数字格式无效。 PID_Temp 使用输出值下限作为输出值。 注：如果在错误发生之前自动模式已激活，ActivateRecoverMode = TRUE 且错误不再处于未决状态，则 PID_Temp 切换回自动模式。
40000	参数 Disturbance 的值无效：值的数字格式无效。 注：如果在错误发生前自动模式已激活且 ActivateRecoverMode = FALSE，则 Disturbance 将设置为零。PID_Temp 保持自动模式。 注：如果在错误发生前预调节或精确调节已激活且 ActivateRecoverMode = TRUE，则 PID_Temp 切换到 Mode 参数中保存的工作模式。如果当前阶段中的干扰对输出值无影响，则不会取消调节。
200000	级联中的主站发生错误。从站不处于自动模式或者启用了替换设定值，阻止了主站的调节。
400000	当冷却正在进行时，PID 控制器不允许执行加热预调节。
800000	要开始进行冷却预调节，过程值必须接近设定值。
1000000	启动调节时出错。“Heat.EnableTuning”和“Cool.EnableTuning”未设置或与组态不匹配。
2000000	冷却预调节要求成功的加热预调节。

ErrorBit (DW#16#...)	说明
4000000	启动精调时出错。“Heat.EnableTuning”和“Cool.EnableTuning”不能同时设置。
8000000	PID 参数计算期间发生的错误导致参数无效（例如，负增益；当前 PID 参数保持不变并且调节无影响）。

[1] 注：如果在错误发生前自动模式已激活且 ActivateRecoverMode = TRUE，则 PID_Temp 保持自动模式。

[2] 注：如果在错误发生前预调节或精确调节已激活且 ActivateRecoverMode = TRUE，则 PID_Temp 切换到 Mode 参数中保存的工作模式。

[3] 注：如果在错误发生前自动模式已激活且 ActivateRecoverMode = TRUE，则 PID_Compact 输出组态的替换输出值。当错误不再处于未决状态时，PID_Temp 切换回自动模式。

[4] 注：如果在错误发生前 ActivateRecoverMode = TRUE，则 PID_Temp 取消调节并切换到 Mode 参数中保存的工作模式。

10.2.4.3 PID_温度警告参数

PID 控制器有多个未决警告时，将会采用二进制加法显示错误代码的值。 例如，显示错误代码 0003，表示错误 0001 和 0002 处于待决状态。

表格 10-29 PID_Temp 指令 Warning 参数

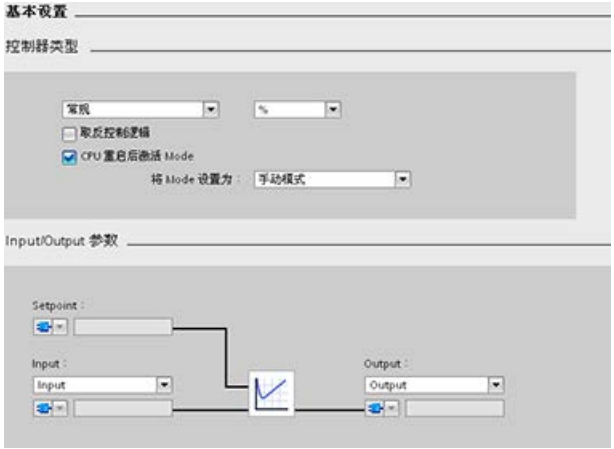
警告 (DW#16#...)	说明
0000	无未决警告。
0001 [1]	预调整期间未找到拐点。
0002	“运行中调整”期间，强制启动了振动。（该“警告（Warning）”参数抑制了该警告，仅在用于诊断目的的“内部警告（WarningInternal）”参数中才可见。）
0004 [1]	该整定值被限制为不得大于已经组态的极限值。
0008 [1]	对于选定的这种计算方法，没有必要设置控制器系统的全部属性。而是采用 TIR.TuneRuleHeat / TIR.TuneRuleCool = 3 计算 PID 的这些参数。
0010	由于 Reset = TRUE 或 ManualEnable = TRUE，因此，运行模式不能更改。
0020	调用 OB 的循环时间决定 PID 算法的采样时间。 采用更短 OB 循环时间改善结果值。

警告 (DW#16#...)	说明
0040 [1]	过程值超过了其中某个警告限值。
0080	“模式（Mode）”中的无效值。 运行模式未切换。
0100 [1]	手动设置值被限制为不得大于控制器输出的极限值。
0200	不支持指定的调整规则。 未计算任何 PID 参数。
1000	输出替代值超过了输出值的极限值，因此，无法达到该输出替代值。
4000	指定用于加热和/或冷却的输出选择不支持。 仅 OutputHeat 和 OutputCool 激活。
8000	PIDSelfTune.SUT.AdaptDelayTime 参数的设定值不支持，因此，使用默认值 "0" 。
10000	PIDSelfTune.SUT.CoolingMode 参数的设定值不支持，因此，使用默认值 "0" 。

[1] 注： 排除警告原因或者采用有效参数重新执行用户操作后，PID 控制器立即自动删除以下警告： 0001、0004、0008、0040 和 0100。

10.2.5 组态 PID_Compact 和 PID_3Step 控制器

工艺对象的参数可决定 PID 控制器的操作。 使用该图标可打开组态编辑器。 



表格 10-30 PID_Compact 指令的组态设置示例

设置		说明
基础	控制器类型	选择工程单元。
	反转控制逻辑 (Invert the control logic)	允许选择反作用 PID 回路。 ● 如果未选择该选项，则 PID 回路处于直接作用模式，在输入值小于设定值时，PID 回路的输出会增大。 ● 如果选择了该选项，则在输入值大于设定值时，PID 回路的输出会增大。
	CPU 重启后启用上一模式 (Enable last mode after CPU restart)	在复位 PID 回路之后，或在超出输入限值后回到有效范围时，重新启动 PID 回路。
	输入 (Input)	为过程值选择 Input 参数或 Input_PER 参数（用于模拟量）。Input_PER 可直接来自模拟量输入模块。
	输出	为输出值选择 Output 参数或 Output_PER 参数（用于模拟量）。Output_PER 可直接进入模拟量输出模块。
过程值	标定过程值的范围和限值。 如果过程值低于下限或高出上限，则 PID 回路进入未激活模式，并将输出值设置为 0。 要使用 Input_PER，必须标定模拟过程值（输入值）。	

表格 10-31 PID_3Step 指令的组态设置示例

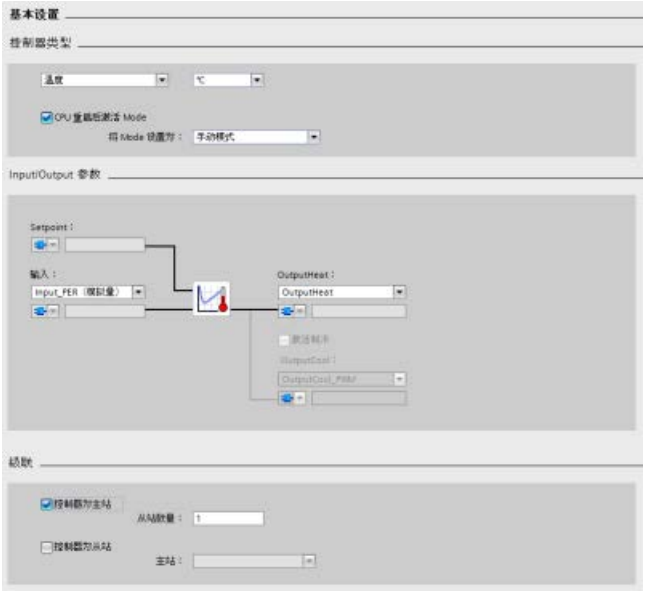
设置		说明
基础	控制器类型	选择工程单元。
	反转控制逻辑	允许选择反作用 PID 回路。 ● 如果未选择该选项，则 PID 回路处于直接作用模式，在输入值小于设定值时，PID 回路的输出会增大。 ● 如果选择了该选项，则在输入值大于设定值时，PID 回路的输出会增大。
	CPU 重启后激活模式	在复位 PID 回路之后，或在超出输入限值后回到有效范围时，重新启动 PID 回路。 将模式设置为： 定义重新启动后用户想要 PID 跳转到的模式。
	Input	为过程值选择 Input 参数或 Input_PER 参数（用于模拟量）。Input_PER 可直接来自模拟量输入模块。
	输出	选择为输出值使用数字量输出（Output_UP 和 Output_DN）或使用模拟量输出 (Output_PER)。
	反馈	选择返回到 PID 回路的设备状态的类型： ● 无反馈（默认） ● 反馈 ● Feedback_PER
过程值	标定过程值的范围和限值。 如果过程值低于下限或高出上限，则 PID 回路进入未激活模式，并将输出值设置为 0。 要使用 Input_PER，必须标定模拟过程值（输入值）。	

设置		说明
执行器	电机切换时间 (Motor transition time)	设置阀门从打开到关闭的时间。（可在阀门的数据表或面板上找到该值。）
	最短打开时间 (Minimum ON time)	设置阀门的最短运动时间。（可在阀门的数据表或面板上找到该值。）
	最短关闭时间 (Minimum OFF time)	设置阀门的最短暂停时间。（可在阀门的数据表或面板上找到该值。）
	对错误的响应	定义检查到错误或复位 PID 回路时的阀门行为。 如果选择使用替换位置，请输入“安全位置”(Safety position)。 对于模拟反馈或模拟输出，应为输出选择在上限和下限之间的值。 对于数字输出，只能选择 0%（关闭）或 100%（打开）。
	刻度位置反馈 [1] (Scale Position Feedback)	• “上端停止位”(High end stop) 和“下端停止位”(Lower end stop) 定义最大正向位置（完全打开）和最大反向位置（完全关闭）。“上端停止位”(High end stop) 必须大于“下端停止位”(Lower end stop)。 • “过程值上限”(High limit process value) 和“过程值下限”(Low limit process value) 定义调节模式和自动模式中阀门的上限位置和下限位置。 • “FeedbackPER”（“下限”和“上限”）定义阀门位置的模拟反馈。“FeedbackPER 上限”必须大于“FeedbackPER 下限”。
高级	监视过程值	为过程值设置警告上限和下限。
	PID 参数	如果用户需要，可以在此窗口中输入自己的 PID 调节参数。 必须选中“启用手动输入”(Enable Manual Entry) 复选框来执行此操作。

[1] 只有在“基本”(Basic) 设置中启用了“反馈”(Feedback) 时，才能编辑“刻度位置反馈”(Scale Position Feedback)。

10.2.6 组态 PID_Temp 控制器

工艺对象的参数可决定 PID 控制器的操作。 使用该图标可打开组态编辑器。 



表格 10-32 PID_Temp 指令的组态设置示例

设置		说明
基础	控制器类型	选择工程单元。
	CPU 重启后激活模式	在复位 PID 回路之后，或在超出输入限值后回到有效范围时，重新启动 PID 回路。 将模式设置为： 定义重新启动后用户想要 PID 跳转到的模式。
	Input	为过程值选择 Input 参数或 Input_PER 参数（用于模拟量）。Input_PER 可直接来自模拟量输入模块。
	OutputHeat	选择为输出值使用数字量输出（OutputHeat 和 OutputHeat_PWM）或使用模拟量输出 (OutputHeat_PER (analog))。
	OutputCool	选择为输出值使用数字量输出（OutputCool 和 OutputCool_PWM）或使用模拟量输出 (OutputCool_PER (analog))。
过程值	标定过程值的范围和限值。 如果过程值低于下限或高出上限，则 PID 回路进入未激活模式，并将输出值设置为 0。 要使用 Input_PER，必须标定模拟过程值（输入值）。	
级联	控制器为主站	将控制器设置为主站并选择从站数量。
	控制器为从站	将控制器设置为从站并选择主站数量。

控制器类型

设置	TO-DB 参数	数据类型	取值范围	说明
实际数量	“PhysicalQuantity”	Int (Enum)	• 常规 • 温度（=默认值）	预选择实际单位值 无多值控制，并且在功能视图的在线模式下无法编辑。
计量单位	“PhysicalUnit”	Int (Enum)	• 常规： 单位 = % • 温度： 单位（可能的选项） = – °C（默认值） – °F – K	如果更改实际数量，选择的用户单位将设置回“0”。
CPU 重启后激活模式	“RunModeByStartup”	Bool	复选框	如果设置为 TRUE（默认值），在重启（电源打开，关闭，再打开）后或 PLC 从 STOP 模式转换到 RUN 模式后，控制器将切换到“Mode”变量中存储的状态”。 否则，PID_Temp 将继续处于“未激活”模式。
将模式设置为	“模式”	Int (Enum)	模式（可能的选项）： • 0: 未激活 • 1: 预调节 • 2: 精确调节 • 3: “自动”模式 • 4: “手动”模式（默认值）	工程站 (ES) 根据用户的选择设置“Mode”变量的起始值。“Mode”的默认值（存储在 TO-DB 中）为“Manual Mode”。

输入/输出参数

设置	TO-DB 参数	数据类型	取值范围	说明
设定值	设定值	Real)	Real	只能在属性页面中访问 功能视图的在线模式下无多值控制。
选择输入	“Config.InputPerOn”	Bool (Enum)	Bool	选择要使用的输入类型。 可选择： • FALSE：“Input” (Real) • TRUE：“Input_PER (analog)”
Input	Input 或 Input_PER	Real 或 Int	Real 或 Int	只能在属性页面中访问。 功能视图的在线模式下无多值控制。
选择输出（加热）	“Config.Output.Heat.Select”	Int (Enum)	2 >= Config.Output.Heat.Select >= 0	选择要用于加热的输出类型。 可选择： • “OutputHeat” (Real) • “OutputHeat_PWM” (Bool)（默认值） • “OutputHeat_PER (analog)” (Word) 如果用户激活了“级联”(Cascade) 部分的“此控制器为主站” (This controller is a master) 复选框，则设置为“OutputHeat”一次。

设置	TO-DB 参数	数据类型	取值范围	说明
输出（加热）	OutputHeat、OutputHeat_PER 或 OutputHeat_PWM	Real、Int 或 Bool	实型、整型或布尔型	只能在属性页面中访问。 功能视图的在线模式下无多值控制。
激活输出（冷却）	“Config.ActivateCooling”	Bool	Bool	选中此复选框： ● 设置 “Config.Output.Heat.PidLowerLimit = 0.0 一次。 ● 设置 “Config.ActivateCooling”参数为 TRUE，而不是 FALSE（未选中时的默认值）。 ● 激活所有其他“输出（冷却）”控件（在“基本设置” (Basic settings) 和其它视图中）。 ● 将从 PID 符号到控件的线从灰色更改为黑色。 ● 禁用了“级联” (Cascade) 部分的“此控制器为主站” (This controller is a master) 复选框。 注： 仅当不将控制器组态为级联的主站时才可用（禁用了“级联”(Cascade) 部分的“此控制器为主站” (This controller is a master) 复选框；"Config.Cascade.IsMaster" = FALSE）。

设置	TO-DB 参数	数据类型	取值范围	说明
选择输出（冷却）	“Config.Output.Cool.Select”	Int (Enum)	2 >= Config.Output.Heat.Select >= 0	选择要用于冷却的输出类型。 可选择： • “OutputCool” (Real) • “OutputCool_PWM” (Bool)（默认值） • “OutputCool_PER (analog)” (Word) 仅当选中“激活输出（冷却）”(Activate output (cooling)) 时才可用；(Config.ActivateCooling = TRUE)。
输出（冷却）	OutputCool、OutputCool_PER 或 OutputCool_PWM	Real、Int 或 Bool	实型、整型或布尔型	只能在属性页面中访问。 功能视图的在线模式下无多值控制。

级联参数

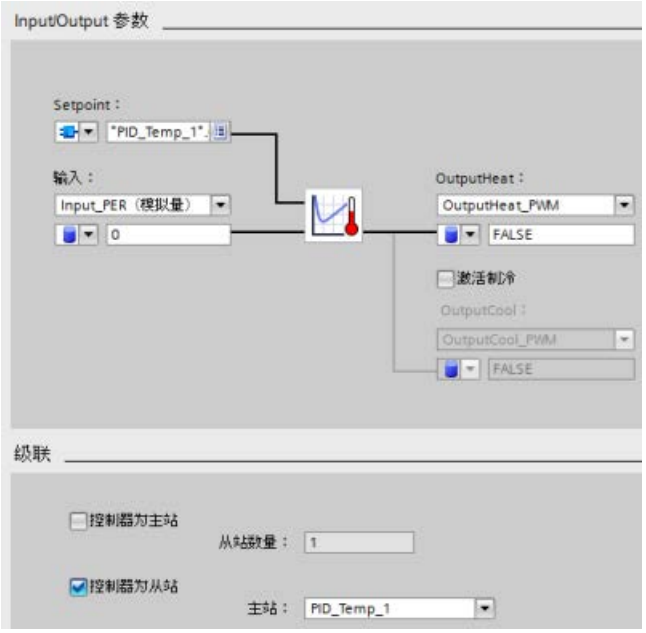
通过以下参数，可选择控制器作为主站或从站，并确定直接从主站控制器接收其设定值的从站控制器的数量：

设置	TO-DB 参数	数据类型	取值范围	说明
此控制器为主站	“Config.Cascade.IsMaster”	Bool	Bool	显示此控制器是否为级联中的主站。 在选中此复选框时，执行以下操作： ● 将参数“Config.Cascade.IsMaster”设置为 TRUE，而不是 FALSE（未选中时的默认值）。 ● 将“输入/输出参数”(Input / output parameters) 部分的“选择输出（加热）”(Selection Output (heating))”设置为“OutputHeat”一次（Config.Output.Heat.Select = 0）。 ● 启用“从站数目”(Number of Slaves) 输入域。 ● 禁用“输入/输出参数”(Input / output parameters) 部分的“激活输出（冷

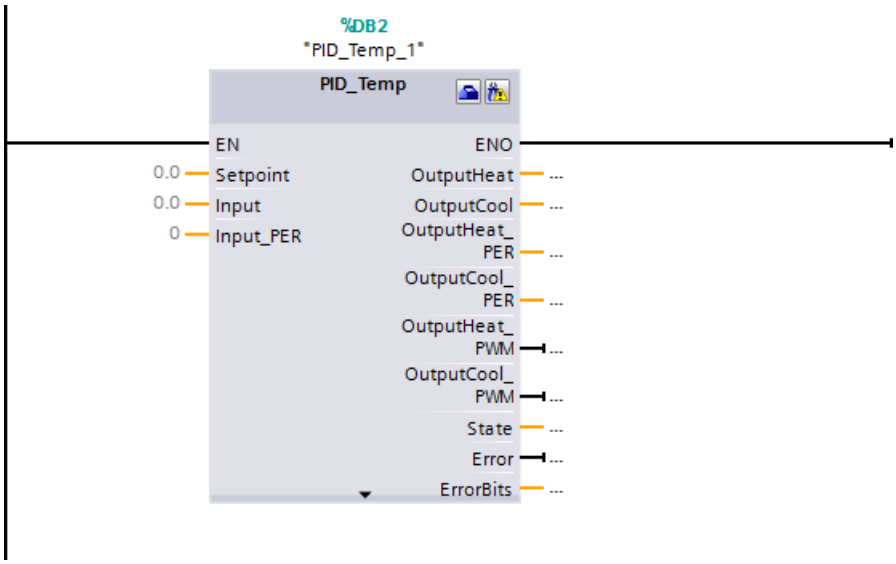
设置	TO-DB 参数	数据类型	取值范围	说明
				却）”(Activate output (cooling))。 注： 仅当禁用了此控制器的冷却输出时才可用（禁用“输入/输出参数”(Input / output parameters) 部分的“激活输出（冷却）”(Activate output (cooling)) 复选框 (Config.ActivateCooling = FALSE)）。
从站数目	“Config.Cascade.CountSlaves”	Int	255 >= Config.Cascade.CountSlaves >= 1	直接从该主站控制器获取其设定值的从站控制器的数目。 PID_Temp 指令将该值与其它值一起处理以用于抗积分饱和处理。仅当激活了“此控制器为主站”(This controller is a master) 复选框 (Config.Cascade.IsMaster = TRUE) 时，“从站数目”(Number of slaves) 才可用。
此控制器为从站	“Config.Cascade.IsSlave”	Bool	Bool	显示此控制器是否为级联中的从站。 选中此复选框时，将参数“Config.Cascade.IsSlave”设置为 TRUE，而不是 FALSE（未选中时的默认值）。 必须在属性页面中选中此复选框以启用“SelectionMaster”下拉列表。

示例： 级联控制器

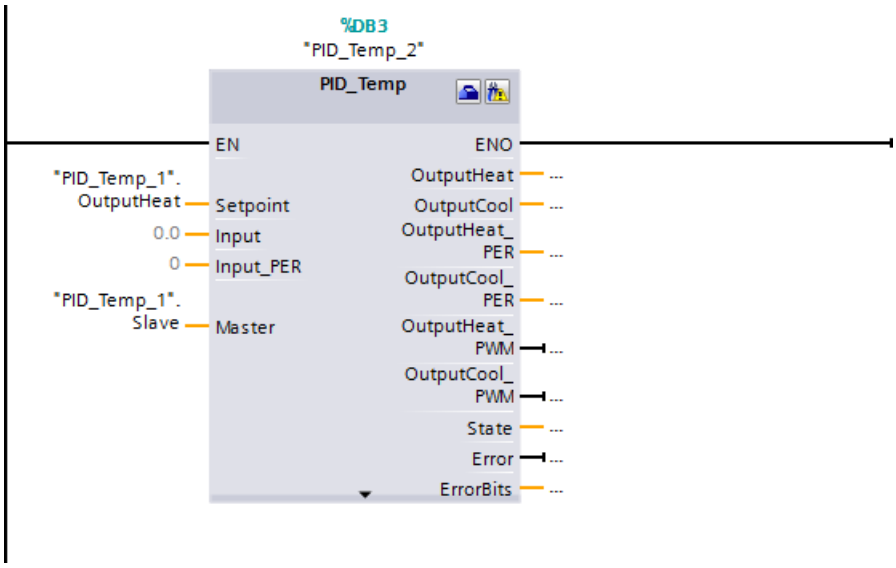
选择“PID_Temp_1”作为主站后，在 “基本设置”(Basic settings) 对话框下面，可以看到从站控制器“PID_Temp_2”的 “输入/输出参数”(Input / output parameters) 部分和 “级联” (Cascade) 部分。 主站和从站控制器之间已建立了连接：



程序段 1： 在这些程序段中，通过编程编辑器在“PID_Temp_1”主站和“PID_Temp_2”从站之间建立了连接：



程序段 2： 已在“PID_Temp_1”主站的“OutputHeat"和“Slave”参数与“PID_Temp_2”从站的“Setpoint”和“Master”参数之间分别建立了连接：



温度自动调节过程

PID_Temp 指令提供两种自动调节模式：

- “预调节”（参数“Mode”= 1）
- “精确调节”（参数“Mode”= 2）

根据控制器组态，提供这些调节方法的不同版本：

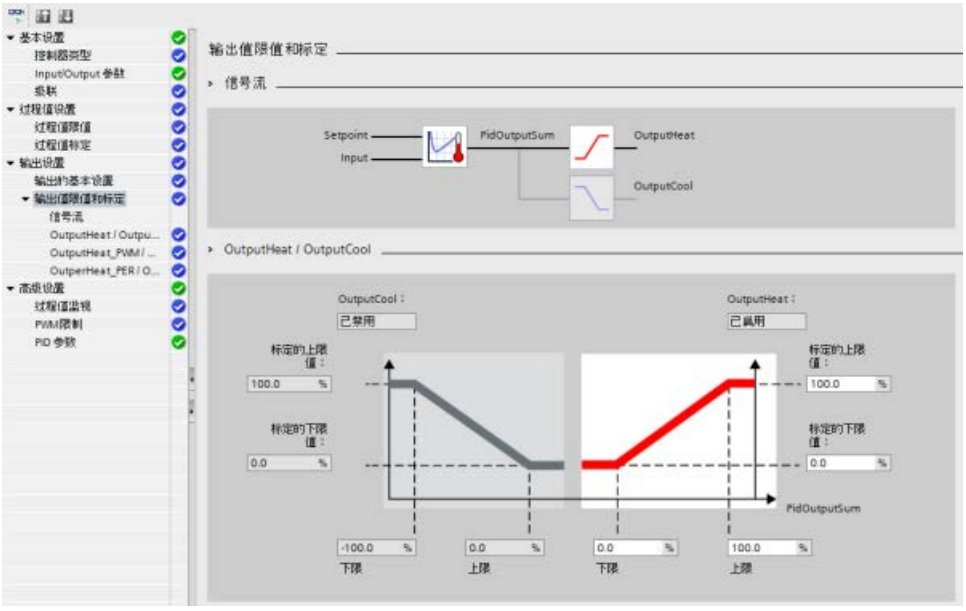
组态	具有加热输出的控制器	具有加热和冷却输出且使用冷却系数的控制器	具有加热和冷却输出且使用两个 PID 参数集的控制器
相关 TO-DB 值	• Config.ActivateCooling = FALSE • Config.AdvancedCooling = irrelevant	• Config.ActivateCooling = TRUE • Config.AdvancedCooling = FALSE	• Config.ActivateCooling = TRUE • Config.AdvancedCooling = TRUE
可用调节方法	• “预调节加热” • “精确调节加热”（不能使用冷却偏移量）	• “预调节加热” • “精确调节加热”（可以使用冷却偏移量）	• “预调节加热和冷却” • “预调节加热” • “预调节冷却” • “精确调节加热”（可以使用冷却偏移量） • “精确调节冷却”（可以使用加热偏移量）

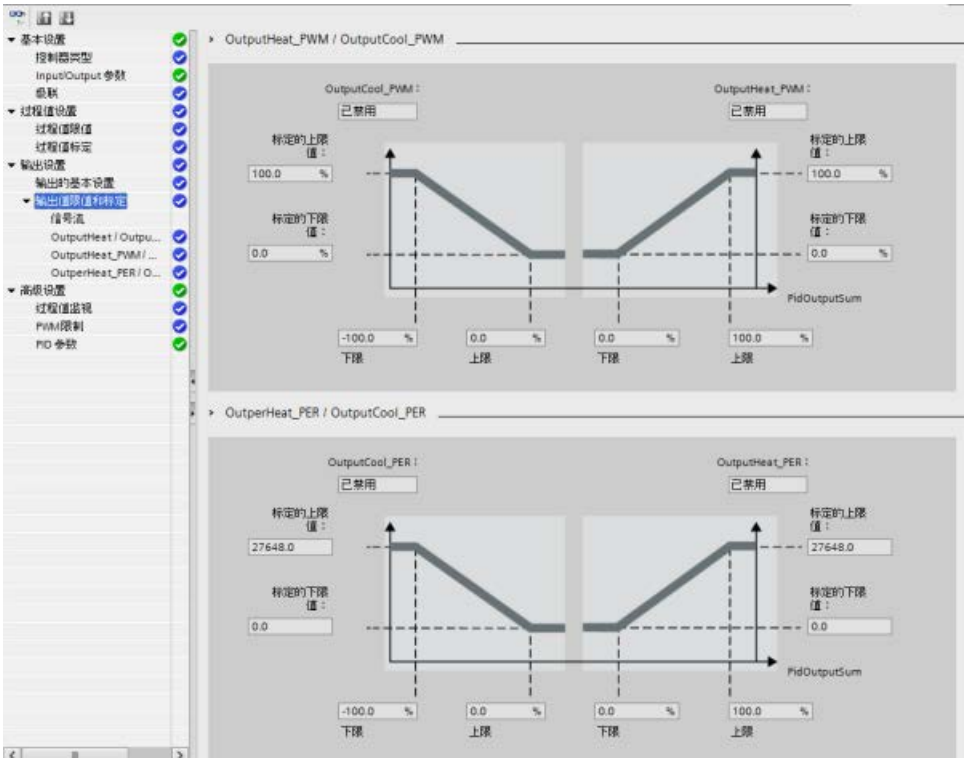
输出值限值和标定

禁用冷却激活

如果组态 PID_Temp 指令作为级联主站，“基本设置”(Basic settings) 视图中的“激活输出（冷却）”(Activate output (cooling)) 复选框将不选中并被禁用，“输出设置”(Output settings) 视图中所有依赖于冷却激活的设置也会禁用。

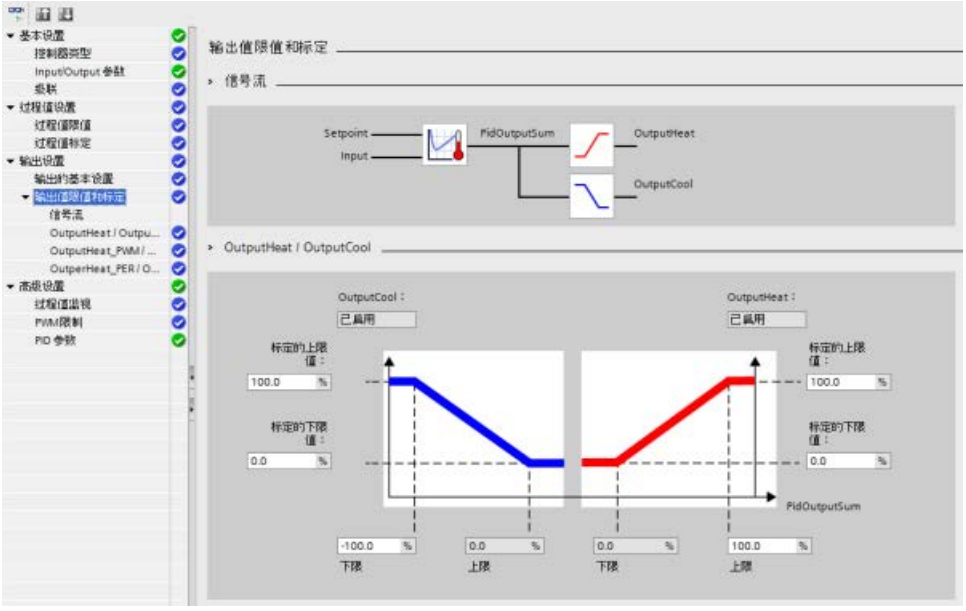
下图显示了禁用冷却时“输出设置”(Output settings) 视图中的“输出值限值和标定”(Output value limits and scaling) 部分（在“输入/输出参数”(Input / output parameters) 视图中选择了 OutputHeat_PWM 并始终启用 OutputHeat）：

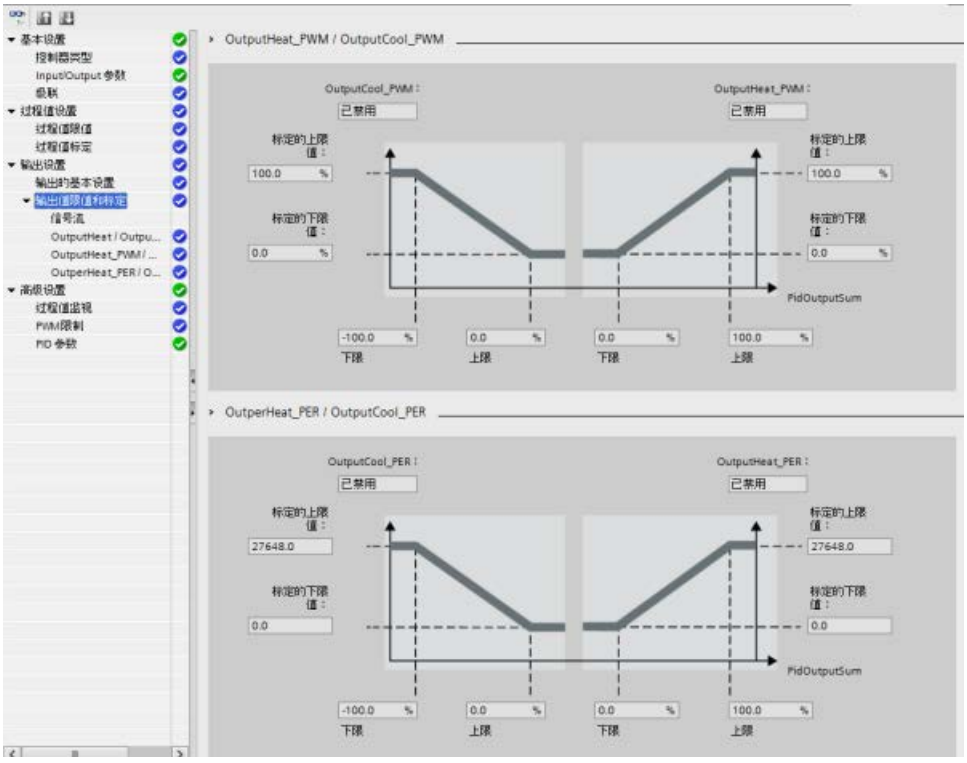




启用冷却激活

下图显示了激活冷却时“输出设置”(Output settings) 视图中的“输出值限值和标定”(Output value limits and scaling) 部分（在“输入/输出参数”(Input / output parameters) 视图中选择了 OutputCool_PER 和 OutputHeat_PWM 并始终启用 OutputCool 和 OutputHeat）：





操作模式

要手动更改操作模式，用户需要设置控制器的“Mode”输入-输出参数，并通过将“ModeActivate”从 FALSE 更改为 TRUET 将其激活（触发上升沿）。 在下次模式更改之前，必须复位“ModeActivate”，它不会自动复位。

输出参数“State”显示当前操作模式，并设置为请求的“Mode”（如果可行）。 “State”参数无法直接更改；它只能由控制器通过“Mode”参数或自动操作模式更改进行更改。

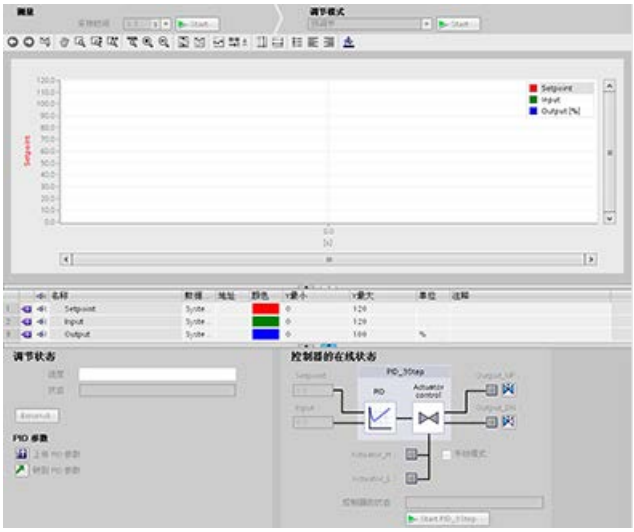
“Mode”/ “State”	名称	说明
0	未激活	PID_Temp 指令： ● 禁用 PID 算法和脉宽调制 ● 所有控制器输出（OutputHeat、OutputCool、OutputHeat_PWM、OutputCool_PWM、OutputHeat_PER、OutputCool_PER）设置为“0” (FALSE)，而不考虑组态的输出限值或偏移量。 可通过设置 “Mode” = 0、“Reset” = TRUE 或通过发生错误进入此模式。
1	预调节（开始调节/ SUT）	此模式在第一次启动控制器时确定参数。 与 PID_Compact 不同，对于 PID_Temp，必须使用“Heat.EnableTuning”和“Cool.EnableTuning”参数选择是否需要启用加热调节和/或冷却调节。 用户可以从“未激活”、“自动”模式或“手动”模式激活“预调节”。 如果调节成功，PID_Temp 将切换到“自动”模式。 如果调节失败，操作模式的切换将取决于“ActivateRecoverMode”。.
2	精确调节（在运行时调节/TIR）	此模式通过设定值确定 PID 控制器的最佳参数设置。 与 PID_Compact 不同，对于 PID_Temp，必须使用“Heat.EnableTuning”和“Cool.EnableTuning”参数选择是否需要启用加热调节或冷却调节。 用户可以从“未激活”、“自动”模式或“手动”模式激活“精确调节”。 如果调节成功，PID_Temp 将切换到“自动”模式。 如果调节失败，操作模式的切换将取决于“ActivateRecoverMode”。.
3	“自动”模式	在“自动”模式（标准 PID 控制模式）下，PID 算法的结果确定输出值。 如果发生错误，PID_Temp 将切换到“未激活”，并且 “ActivateRecoverMode” = FALSE。 如果发生错误并且 “ActivateRecoverMode” = TRUE，操作模式的切换将取决于错误。 有关更多信息，请参见 PID_Temp 指令 ErrorBit 参数 (页 622)。

“Mode”/ “State”	名称	说明
4	“手动”模式	在这种模式下，PID 控制器将参数“ManualValue”的值标定、限制并传送到输出。 PID 控制器在 PID 算法的标定中分配“ManualValue”（类似于“PidOutputSum”），所以它的值决定它对加热或冷却输出是否有效。 可通过设置 “Mode” = 4 或 “ManualEnable”= TRUE 进入此模式。
5	通过错误监视替换输出值（“恢复”模式）	可以通过设置 "Mode" = 5. 激活此模式。如果在错误发生时“自动”模式处于激活状态，此模式将是控制器的自动错误响应： ● SetSubstituteOutput = FALSE（上一个有效输出值） ● SetSubstituteOutput = TRUE（存储在参数“SubstituteOutput”中的值） 当 PID_Temp 处于“自动”模式并且 “ActivateRecoverMode” = TRUE 时，PID_Temp 将 在出现以下错误时更改为此模式： ● ““Input_PER”参数的值无效。 请检查模拟量输入是否出错（例如，断线）。” (ErrorBits = DW#16#0002) ● ““Input”参数的值无效。 值不是一个数字。” (ErrorBits = DW#16#0200) ● “输出值计算失败。 请检查 PID 参数。” (ErrorBits = DW#16#0400) ● ““Setpoint”参数的值无效。 值不是一个数字。” (ErrorBits = DW#16#1000) 当错误不再处于未决状态时，PID_Temp 将自动切换回“自动”模式。

10.2.7 调试 PID_Compact 和 PID_3Step 控制器

使用调试编辑器可组态 PID 控制器，使其在启动时和操作过程中可自动调节。要打开调试编辑器，请单击指令或项目浏览器上的图标。

表格 10-33 调试画面示例 (PID_3Step)

	• 测量：要在实时趋势中显示设定值、过程值（输入值）和输出值，请输入采样时间并单击“开始”(Start) 按钮。 • 调节模式：要调节 PID 循环，请选择“预调节”(Pretuning) 或“精确调节”(Fine tuning)（手动）并单击“开始”(Start) 按钮。PID 控制器会运行多个阶段，以计算系统响应时间和更新时间。通过这些值可计算相应的调节参数。 完成调节过程之后，可以单击调试编辑器的“PID 参数”(PID Parameters) 部分中的“上传 PID 参数”(Upload PID parameters) 按钮来存储新参数。 如果调节期间未发生错误，则 PID 的输出值变为 0。PID 模式将设置为“未激活”模式。状态可指示错误。

PID 起始值控制

您可以编辑 PID 组态参数的实际值，以便可以通过在线模式优化 PID 控制器的特性。

打开 PID 控制器的“工艺对象”(Technology objects) 及其“组态”(Configuration) 对象。要访问起始值控制，单击此对话框左上角的“眼镜图标”：

现在可以更改 PID 控制器组态参数的任何值，如下图所示。

可以将实际值与每个参数的项目（离线）起始值和 PLC（在线）起始值进行比较。这对于比较工艺对象数据块 (TO-DB) 的在线/离线差异以及了解在 PLC 下一次“停止到开始”转换时哪些值将用作当前值很有必要。此外，比较图标还会通过视觉指示帮助您轻松确定在线/离线差异：

上图展示了带有比较图标的 PID 参数画面，其中显示出在线和离线项目之间有哪些值存在差异。绿色图标表示值相同，蓝色/橙色图标表示值不同。

另外，单击带有向下箭头的参数按钮，可打开一个显示每个参数的项目（离线）起始值和 PLC（在线）起始值的小窗口：

10.2.8 调试 PID_Temp 控制器

使用调试编辑器可组态 PID 控制器，使其在启动时和操作过程中可自动调节。要打开调试编辑器，请单击指令或项目浏览器上的图标。

表格 10-34　调试画面示例 (PID_Temp)

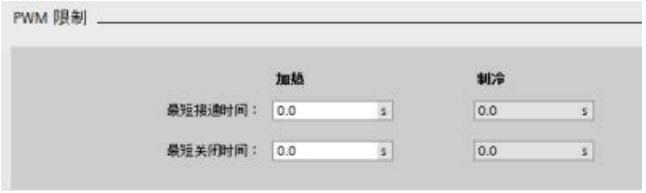
	测量：要在实时趋势中显示设定值、过程值（输入值）和输出值，请输入采样时间并单击“开始”(Start) 按钮。 调节模式：要调节 PID_Temp 循环，请选择“预调节”(Pretuning) 或“精确调节”(Fine tuning)（手动）并单击“开始”(Start) 按钮。PID 控制器会运行多个阶段，以计算系统响应时间和更新时间。通过这些值可计算相应的调节参数。 完成调节过程之后，可以单击调试编辑器的“PID 参数”(PID Parameters) 部分中的“上传 PID 参数”(Upload PID parameters) 按钮来存储新参数。 如果调节期间未发生错误，则 PID 的输出值变为“0”。PID 模式将设置为“未激活”模式。状态可指示错误。

PWM 限值

使用 PID_Temp 的软件 PWM 功能控制的执行器可能需要保护，以免出现太短的脉冲持续时间（例如，可控硅继电器需要开启 20 ms 以上才能正常反应）；用户指定了一个最短时间。执行器还可以忽略比较短的脉冲并因此影响控制质量。需要设置一个最短的关断时间（例如，防止过热）。

要显示 PWM 限值视图，必须在工艺对象 (TO) 组态中打开功能视图，并从导航树中的“高级设置”(Advanced settings) 节点选择“PWM 限值”(PWM limits)。

如果在功能视图中打开“PWM 限值”(PWM limits) 视图并激活监视功能（“眼镜”按钮），所有控件都会显示来自 TO-DB 的在线监视值（橙色背景色）和多值控件，用户可以编辑这些值（前提是满足组态条件；请参见下表）。



设置	TO-DB 参数	数据类型	取值范围	说明
最短时间（加热）[1,2]	“Config.Output.Heat. MinimumOnTime”	Real	100000.0 >= "Config.Output. Heat. MinimumOnTime >= 0.0	“OutputHeat_PWM ”中的脉冲永不会短于该值。
最短关断时间（加热）[1,2]	“Config.Output.Heat. MinimumOffTime”	Real	100000.0 >= "Config.Output. Heat. MinimumOffTime >= 0.0	“OutputHeat_PWM ”中的中断永不会短于该值。

设置	TO-DB 参数	数据类型	取值范围	说明
最短时间（冷却）[1,3,4]	“Config.Output.Cool.MinimumOnTime”	Real	100000.0 >= Config.Output.Cool.MinimumOnTime >= 0.0	“OutputCool_PWM”中的脉冲永不会短于该值。
最短关断时间（冷却）[1,3,4]	“Config.Output.Cool.MinimumOffTime”	Real	100000.0 >= Config.Output.Cool.MinimumOffTime >= 0.0	“OutputCool_PWM”中的中断永不会短于该值。

1 该域显示“s”（秒）作为时间单位。

2 如果“基本设置”(Basic settings) 视图中的选择输出（加热）不是“OutputHeat_PWM” (Config.Output.Heat.Select = TRUE)，则应将该值设为“0.0”。

3 如果“基本设置”(Basic settings) 视图中的选择输出（冷却）不是“OutputCool_PWM” (Config.Output.Cool.Select = TRUE)，则应将该值设为“0.0”。

4 仅当选中“基本设置”(Basic settings) 视图中的“激活输出（冷却）”(Activate output (cooling)) (Config.ActivateCooling = TRUE) 时才可用。

PID 参数

下面显示了“高级设置”(Advanced settings) 视图的“PID 参数”(PID Parameters) 部分（禁用了冷却功能和/或“PID 参数切换”(PID parameterswitchover) 功能）。

设置	TO-DB 参数	数据类型	取值范围	说明
启用手动输入	“Retain.CtrlParams. SetByUser”	Bool	Bool	必须选中此复选框才能手动输入 PID 参数。
比例增益（加热）[2]	“Retain.CtrlParams. Heat.Gain”	Real	Gain >= 0.0	PID 加热比例增益
积分作用时间（加热）[1,2]	“Retain.CtrlParams. Heat.Ti”	Real	100000.0 >= Ti >= 0.0	PID 加热积分作用时间。
微分作用时间（加热）[1,2]	“Retain.CtrlParams. Heat.Td”	Real	100000.0 >= Td >= 0.0	PID 加热微分作用时间。
微分延迟系数（加热）[2]	“Retain.CtrlParams. Heat.TdFiltRatio”	Real	TdFiltRatio >= 0.0	PID 加热微分延迟系数，定义微分滞后时间作为 PID 微分时间中的系数。
比例作用加权（加热）[2]	“Retain.CtrlParams. Heat.PWeighting”	Real	1.0 >=PWeighting >= 0.0	PID 加热比例增益的加权，采用直接或环路控制路径。

设置	TO-DB 参数	数据类型	取值范围	说明
微分作用加权（加热）[2]	“Retain.CtrlParams. Heat.DWeighting”	Real	1.0 >=DWeighting >= 0.0	PID 加热微分部分的加权，采用直接或环路控制路径。
PID 算法采样时间（加热）[1,2]	“Retain.CtrlParams. Heat.Cycle”	Real	100000.0 >=Cycle > 0.0	PID 控制器用于加热的内部调用周期。 舍入为 FB 调用周期时间的整数倍。
死区宽度（加热）[2,3]	“Retain.CtrlParams. Heat.DeadZone”	Real	DeadZone>= 0.0	加热控制偏差的死区宽度。
控制区（加热）[2,3]	“Retain.CtrlParams. Heat.ControlZone”	Real	ControlZone> 0.0	在 PID 控制处于激活状态时用于加热的控制偏差区的宽度。如果控制偏差超出此范围，输出将切换到最大输出值。 默认值为“MaxReal”，所以只要不执行自动调节，控制区就会被禁用。 禁止对控制区使用值“0.0”；使用值“0.0”时，PID_Temp 类似于双位置控制器，始终以满功率加热或冷却。

设置	TO-DB 参数	数据类型	取值范围	说明
控制器结构（加热）	“PIDSelfTune.SUT.TuneRuleHeat”、“PIDSelfTune.TIR.TuneRuleHeat”	Int	"PIDSelfTune.SUT.TuneRuleHeat" = 0..2，"PIDSelfTune.TIR.TuneRuleHeat" = 0..5	用户可以选择加热调节算法。 可选择： • PID (Temperature)（默认值）("PIDSelfTune.SUT.TuneRuleHeat" = 2) ("PIDSelfTune.TIR.TuneRuleHeat" = 0) • PID ("PIDSelfTune.SUT.TuneRuleHeat" = 0) ("PIDSelfTune.TIR.TuneRuleHeat" = 0) • PI ("PIDSelfTune.SUT.TuneRuleHeat" = 1) ("PIDSelfTune.TIR.

设置	TO-DB 参数	数据类型	取值范围	说明
				TuneRuleHeat" = 4) 任何其它组合都会显示“用户自定义”，但“用户自定义”并非默认提供。 “PID (Temperature)”是 PID_Temp 的一种新算法，可使用特定的预调节 (SUT) 方法处理温度。
比例增益（冷却）[4]	“Retain.CtrlParams. Cool.Gain”	Real	Gain >= 0.0	PID 冷却比例增益
积分作用时间（冷却）[1,4]	“Retain.CtrlParams. Cool.Ti”	Real	100000.0 >=Ti >= 0.0	PID 冷却积分作用时间
微分作用时间（冷却）[1,4]	“Retain.CtrlParams. Cool.Td”	Real	100000.0 >=Td >= 0.0	PID 冷却微分作用时间
微分延迟系数（冷却）[4]	“Retain.CtrlParams. Cool.TdFiltRatio”	Real	TdFiltRatio>= 0.0	PID 冷却微分延迟系数，定义微分滞后时间作为 PID 微分时间中的系数。
比例作用加权（冷却）[4]	“Retain.CtrlParams. Cool.PWeighting”	Real	1.0 >=PWeighting >= 0.0	PID 冷却比例增益的加权，采用直接或环路控制路径。
微分作用加权（冷却）[4]	“Retain.CtrlParams. Cool.DWeighting”	Real	1.0 >=DWeighting >= 0.0	PID 冷却微分部分的加权，采用直接或环路控制路径。
PID 算法采样时间（冷却）[1,4]	“Retain.CtrlParams. Cool.Cycle”	Real	100000.0 >=Cycle > 0.0	PID 控制器用于冷却的内部调用周期。 舍入为 FB 调用周期时间的整数倍。
死区宽度（冷却）[3,4]	“Retain.CtrlParams. Cool.DeadZone”	Real	DeadZone>= 0.0	冷却控制偏差的死区宽度

设置	TO-DB 参数	数据类型	取值范围	说明
控制区（冷却）[3,4]	“Retain.CtrlParams.Cool.ControlZone”	Real	ControlZone> 0.0	在 PID 控制处于激活状态时用于冷却的控制偏差区的宽度。如果控制偏差超出此范围，输出将切换到最大输出值。 默认值为“MaxReal”，所以只要不执行自动调节，控制区就会被禁用。 禁止对控制区使用值“0.0”；使用值“0.0”时，PID_Temp 类似于双位置控制器，始终以满功率加热或冷却。

设置	TO-DB 参数	数据类型	取值范围	说明
控制器结构（冷却）	“PIDSelfTune.SUT.TuneRuleCool”、“PIDSelfTune.TIR.TuneRuleCool”	Int	"PIDSelfTune.SUT.TuneRuleHeat" = 0..2，"PIDSelfTune.TIR.TuneRuleHeat" = 0..5	用户可以选择冷却调节算法。 可选择： • PID (Temperature)（默认值）("PIDSelfTune.SUT.TuneRuleCool" = 2) ("PIDSelfTune.TIR.TuneRuleCool = 0) • PID ("PIDSelfTune.SUT.TuneRuleCool" = 0) ("PIDSelfTune.TIR.TuneRuleCool" = 0) • PI ("PIDSelfTune.SUT.TuneRuleCool" = 1) ("PIDSelfTune.TIR.

设置	TO-DB 参数	数据类型	取值范围	说明
				TuneRuleCool" = 4) 任何其它组合都会显示“用户自定义”，但“用户自定义”并非默认提供。 “PID (Temperature)”是 PID_Temp 的一种新算法，可使用特定的预调节 (SUT) 方法处理温度。 仅当选中/选择以下项目时可用：“基本设置”(Basic settings) 视图中的“激活输出（冷却）”(Activate output (cooling)) ("Config.ActivateCooling" = TRUE) 和“输出设置”(Output settings) 视图中的“PID 参数切换”(PID parameter switchover) (Config.AdvancedCooling = TRUE)。

1 该域显示“s”（秒）作为时间单位。

2 仅当选中了 PID 参数中的“启用手动输入”(Enable manual entry) ("Retain.CtrlParams.SetByUser" = TRUE) 时才可用。

3 根据“基本设置”(Basic settings) 视图中的选择，计量单位显示在域的末尾。

4 仅当选中/选择以下项目时可用：PID 参数中的“启用手动输入”(Enable manual entry) ("Retain.CtrlParams.SetByUser" = TRUE)，“基本设置”(Basic settings) 视图中的“激活输出（冷却）”(Activate output (cooling)) ("Config.ActivateCooling" = TRUE) 和“输出设置”(Output settings) 视图中的“PID 参数切换”(PID parameter switchover) (Config.AdvancedCooling = TRUE)。

PID 起始值控制

您可以编辑 PID 组态参数的实际值，以便可以通过在线模式优化 PID 控制器的特性。

打开 PID 控制器的“工艺对象”(Technology objects) 及其“组态”(Configuration) 对象。要访问起始值控制，单击此对话框左上角的“眼镜图标”：

现在可以更改 PID 控制器组态参数的任何值，如下图所示。

可以将实际值与每个参数的项目（离线）起始值和 PLC（在线）起始值进行比较。这对于比较工艺对象数据块 (TO-DB) 的在线/离线差异以及了解在 PLC 下一次“停止到开始”转换时哪些值将用作当前值很有必要。此外，比较图标还会通过视觉指示帮助您轻松确定在线/离线差异：

上图展示了带有比较图标的 PID 参数画面，其中显示出在线和离线项目之间有哪些值存在差异。绿色图标表示值相同，蓝色/橙色图标表示值不同。

另外，单击带有向下箭头的参数按钮，可打开一个显示每个参数的项目（离线）起始值和 PLC（在线）起始值的小窗口：

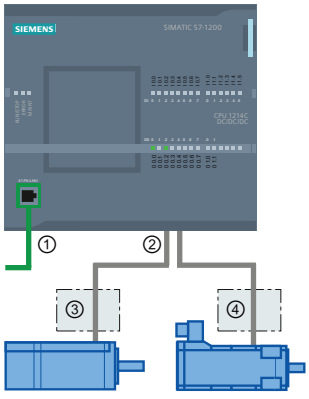
10.3 运动控制

CPU 提供运动控制功能以通过脉冲接口运行步进电机和伺服电机。运动控制功能负责对驱动器进行监控。

- “轴”工艺对象用于组态机械驱动器的数据、驱动器的接口、动态参数以及其它驱动器属性。
- 通过对 CPU 的脉冲输出和方向输出进行组态来控制驱动器。
- 用户程序使用运动控制指令来控制轴并启动运动任务。
- PROFINET 接口用于在 CPU 与编程设备之间建立在线连接。除了 CPU 的在线功能外，附加的调试和诊断功能也可用于运动控制

说明

仅当 CPU 从 STOP 切换为 RUN 模式时，RUN 模式下对运动控制配置和下载的更改才会生效。



① PROFINET

② 脉冲和方向输出

③ 步进电机的电源部分

④ 伺服电机的电源部分

DC/DC/DC 型 CPU S7-1200 上配备有用于直接控制驱动器的板载输出。继电器型 CPU 需要具有用来控制驱动器的 DC 输出的信号板。

信号板 (SB, Signal Board) 将板载 I/O 扩展为包含多个附加 I/O 点。具有两个数字量输出的 SB 可用作控制一台电机的脉冲输出和方向输出。具有四个数字量输出的 SB 可用作控制两台电机的脉冲输出和方向输出。不能将内置继电器输出用作控制电机的脉冲输出。不论是使用板载 I/O、SB I/O 还是二者的组合，最多可以拥有四个脉冲发生器。

这四个脉冲发生器具有默认的 I/O 分配，但是，它们可组态为 CPU 或 SB 上的任意数字量输出。不能将 CPU 上的脉冲发生器分配至 SM 或分布式 I/O。

说明

用户程序中的其它指令无法使用脉冲串输出

将 CPU 或信号板的输出组态为脉冲发生器时（与 PWM 或运动控制指令配合使用），相应的输出地址不再控制输出。如果用户程序向用作脉冲发生器的输出写入值，则 CPU 不会将该值写入到物理输出。

表格 10-35　可控制驱动器的最大数目

CPU 型号		板载 I/O；未安装任何 SB		带 SB（2 x DC 输出）		带 SB（4 x DC 输出）	
		带方向	不带方向	带方向	不带方向	带方向	不带方向
CPU 1211C	DC/DC/DC	2	4	3	4	4	4
	AC/DC/继电器	0	0	1	2	2	4
	DC/DC/继电器	0	0	1	2	2	4
CPU 1212C	DC/DC/DC	3	4	3	4	4	4
	AC/DC/继电器	0	0	1	2	2	4
	DC/DC/继电器	0	0	1	2	2	4
CPU 1214C	DC/DC/DC	4	4	4	4	4	4
	AC/DC/继电器	0	0	1	2	2	4
	DC/DC/继电器	0	0	1	2	2	4
CPU 1215C	DC/DC/DC	4	4	4	4	4	4
	AC/DC/继电器	0	0	1	2	2	4
	DC/DC/继电器	0	0	1	2	2	4
CPU 1217C	DC/DC/DC	4	4	4	4	4	4

说明

最多四个脉冲发生器。

不论是使用板载 I/O、SB I/O 还是二者的组合，最多可以拥有四个脉冲发生器。

表格 10-36 CPU 输出：最大频率

CPU	CPU 输出通道	脉冲和方向输出	A/B，正交，上/下和脉冲/方向
1211C	Qa.0 到 Qa.3	100 kHz	100 kHz
1212C	Qa.0 到 Qa.3	100 kHz	100 kHz
	Qa.4、Qa.5	20 kHz	20 kHz
1214C 和 1215C	Qa.0 到 Qa.3	100kHz	100kHz
	Qa.4 到 Qb.1	20 kHz	20 kHz
1217C	DQa.0 到 DQa.3（.0+，.0- 到 .3+，.3-）	1 MHz	1 MHz
	DQa.4 到 DQb.1	100 kHz	100 kHz

表格 10-37 SB 信号板输出：最大频率（可选信号板）

SB 信号板	SB 输出通道	脉冲和方向输出	A/B，正交，上/下和脉冲/方向
SB 1222，200 kHz	DQe.0 到 DQe.3	200kHz	200 kHz
SB 1223，200 kHz	DQe.0，DQe.1	200kHz	200 kHz
SB 1223	DQe.0，DQe.1	20 kHz	20 kHz

表格 10-38 脉冲输出的频率范围

脉冲输出	频率
板载	4 PTO：2 Hz ≤ f ≤ 1 MHz，4 PTO：2 Hz ≤ f ≤ 100 kHz，或用于 4 个 PTO 的这些值的任意组合。[1] [2]
标准 SB	2 Hz ≤ f ≤ 20 kHz
高速 SB	2 Hz ≤ f ≤ 200 kHz

[1] 请参见下表，了解 CPU 1217C 的四种可能输出速度组合。

[2] 请参见下表，了解 CPU 1211C、CPU 1212C、CPU 1214C 或 CPU 1215C 的四种可能输出速度组合。

示例：CPU 1217C 脉冲输出速度组态

说明

使用板载差分输出，CPU 1217C 可生成高达 1 MHz 的脉冲输出。

以下示例显示了四种可能的输出速度组合：

- 示例 1：4 - 1 MHz PTO，不带方向输出
- 示例 2：1 - 1 MHz、2 - 100 kHz 和 1 - 20 kHz PTO，全部带方向输出
- 示例 3：4 - 200 kHz PTO，不带方向输出
- 示例 4：2 - 100 kHz PTO 和 2 - 200 kHz PTO，全部带方向输出

P = 脉冲 D = 方向	CPU 板载输出										高速 SB 输出				标准 SB 输出	
	1 MHz 输出 (Q)				100 kHz 输出 (Q)						200 kHz 输出 (Q)				20 kHz 输出 (Q)	
	0.0 +	0.1 +	0.2+	0.3 +	0.4	0.5	0.6	0.7	1.0	1.1	4.0	4.1	4.2	4.3	4.0	4.1
	0.0-	0.1-	0.2-	0.3-												

P = 脉冲 D = 方向		CPU 板载输出										高速 SB 输出				标准 SB 输出	
示例 1：4 - 1 MHz（无方向输出）	PTO1	P															
	PTO2		P														
	PTO3			P													
	PTO4				P												
示例 2：1 - 1 MHz、2 - 100 KHz 和 1 - 20 kHz（全部带方向输出）	PTO1	P	D														
	PTO2					P	D										
	PTO3							P	D								
	PTO4															P	D
示例 3: 4 - 200 kHz（无方向输出）	PTO1											P					
	PTO2												P				
	PTO3													P			
	PTO4														P		
示例 4: 2 - 100 kHz; 2 - 200 kHz（全部带有方向输出）	PTO1					P	D										
	PTO2							P	D								
	PTO3											P	D				
	PTO4													P	D		

示例：CPU 1211C、CPU 1212C、CPU 1214C 和 CPU 1215C 脉冲输出速度组态

以下示例显示了四种可能的输出速度组合：

- 示例 1：4 - 100 kHz PTO，不带方向输出
- 示例 2：2 - 100 kHz PTO 和 2 - 20 kHz PTO，全部带方向输出
- 示例 3：4 - 200 kHz PTO，不带方向输出
- 示例 4：2 - 100 kHz PTO 和 2 - 200 kHz PTO，全部带方向输出

P = 脉冲 D = 方向		CPU 板载输出										高速 SB 输出				低速 SB 输出	
		100 kHz 输出 (Q)				20 kHz 输出 (Q)						200 kHz 输出 (Q)				20 kHz 输出 (Q)	
		0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	1.0	1.1	4.0	4.1	4.2	4.3	4.0	4.1
		CPU 1211C															
		CPU 1212C				CPU 1212C											
		CPU 1214C				CPU 1214C		CPU 1214C									
		CPU 1215C				CPU 1215C		CPU 1215C									
示例 1：4 - 100 kHz（无方向输出）	PTO1	P															
	PTO2		P														
	PTO3			P													
	PTO4				P												
示例 2：2 - 100 KHz；2 - 20 KHz（全部带方向输出）	PTO1	P	D														
	PTO2			P	D												
	PTO3					P	D										
	PTO4							P	D								

P = 脉冲 D = 方向		CPU 板载输出										高速 SB 输出				低速 SB 输出	
示例 3: 4 - 200 kHz（无方向输出）	PTO1											P					
	PTO2												P				
	PTO3													P			
	PTO4														P		
示例 4: 2 - 100 kHz; 2 - 200 kHz（全部带有方向输出）	PTO1	P	D														
	PTO2			P	D												
	PTO3											P	D				
	PTO4													P	D		

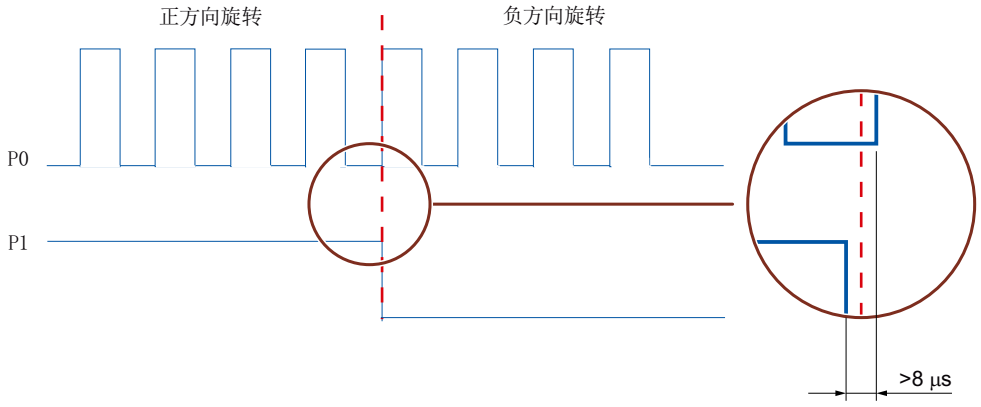
说明

有关详细信息，请参见 *TIA Portal V15 中的 SIMATIC STEP 7 S7-1200 运动控制 V6.0*。

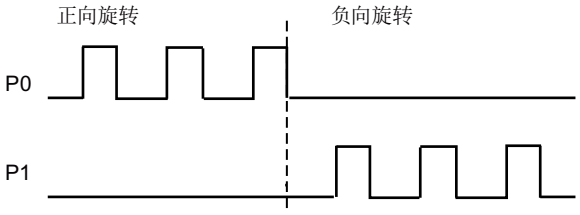
10.3.1 “定相”(Phasing)

步进/伺服驱动器的“定相”(Phasing) 接口有 4 个选项。选项如下：

- PTO（脉冲 A 和方向 B）：如果选择 PTO（脉冲 A 和方向 B）选项，则一个输出 (P0) 控制脉冲，另一输出 (P1) 控制方向。如果脉冲处于正向，则 P1 为高电平（激活）。如果脉冲处于负向，则 P1 为低电平（未激活）：

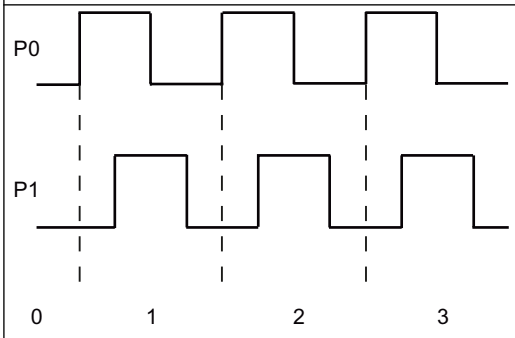
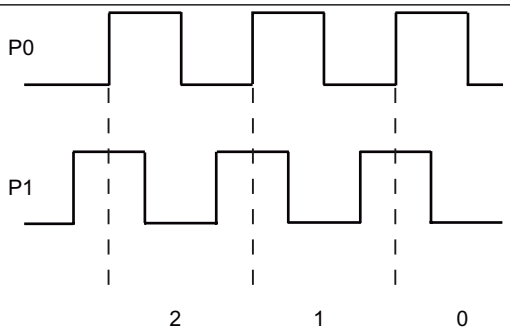


- PTO（向上脉冲 A 和向下脉冲 B）：如果选择 PTO（向上脉冲 A 和向下脉冲 B）选项，则一个输出 (P0) 脉冲控制正方向，另一个输出 (P1) 脉冲控制负方向：

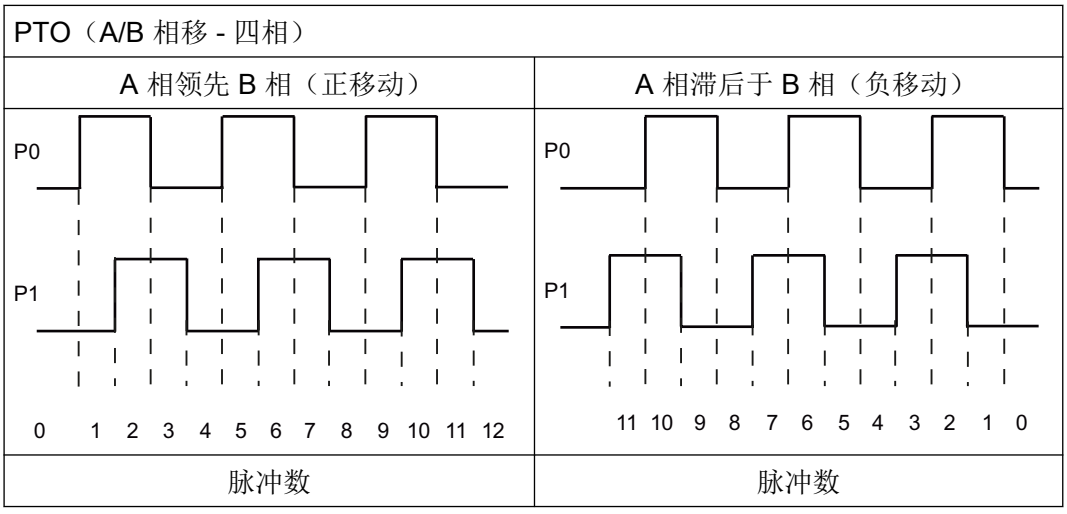


- PTO（A/B 相移）：如果选择 PTO（A/B 相移）选项，则两个输出均以指定速度产生脉冲，但相位相差 90 度。它是一种 1X 组态，表示一个脉冲是 P0 的两次正向转换之间的时间量。这种情况下，方向由先变为高电平的输出转换决定。P0 领先 P1 表示正向。P1 领先 P0 表示负向。

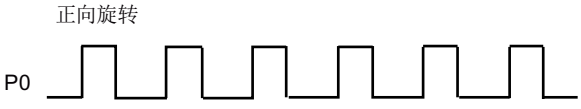
  生成的脉冲数取决于 A 相的 0 到 1 的转换次数。相位关系决定了移动方向：

PTO（A/B 相移）	
A 相领先 B 相（正移动）	A 相滞后于 B 相（负移动）
	
脉冲数	脉冲数

- PTO（A/B 相移 - 四相）：如果选择 PTO（A/B 相移 - 四相）选项，则两个输出均以指定速度产生脉冲，但相位相差 90 度。四相是一种 4X 组态，表示一个脉冲是每个输出（正向和负向）的转换。这种情况下，方向由先变为高电平的输出转换决定。P0 领先 P1 表示正向。P1 领先 P0 表示负向。
  四相取决于 A 相和 B 相的正向和负向转换。您可以组态转换次数。相位关系（A 领先 B 或 B 领先 A）决定了移动方向。



- PTO（脉冲和方向（已取消选择方向））：如果在 PTO（脉冲和方向（已取消选择方向））中取消方向输出，则输出 (P0) 控制脉冲。未使用输出 P1，输出 P1 可供其它程序使用。在此模式下 CPU 只接受正向运动命令。选择此模式时，运动控制限制进行非法负向组态。如果运动应用仅在一个方向进行，则可保存输出。单相（一个输出）如下图所示（假设极性为正）：



10.3.2 组态脉冲发生器

1. 添加工艺对象：
   - 在项目树中，展开节点“工艺对象”(Technology Objects)，然后选择“添加新对象”(Add new object）。
   - 选择“轴”(Axis) 图标（必要时可以重命名），然后单击“确定”(OK) 打开轴对象的组态编辑器。

     说明

     为确保项目的一致性，重命名工艺对象后，请在 CPU 处于 STOP 模式时将项目下载到 CPU 中。当删除一个工艺对象，并用一个新名称和数据块编号创建新工艺对象时，名称将会变化。

   - 显示“基本参数”(Basic parameters) 下的“为轴控制选择 PTO”(Select PTO for Axis Control) 属性，然后选择所需脉冲。

     说明

     如果以前从未在“CPU 属性”(CPU Properties) 中组态 PTO，则将 PTO 组态为使用其中一个板载输出。

     如果使用了输出信号板，则选择“设备组态”(Device configuration) 按钮以转到“CPU 属性”(CPU Properties)。在“参数分配”(Parameter assignment) 下的“脉冲选项”(Pulse options) 中，将数据源组态为信号板输出。

   - 对其余的基本参数和扩展参数进行组态。
2. 对应用进行编程：将 MC_Power 指令插入代码块。
   - 对于“轴”输入，请选择已创建并组态的轴工艺对象。
   - 将 Enable 输入设置为 TRUE 可以使其它运动指令起作用。
   - 将 Enable 输入设置为 FALSE 会取消其它运动指令。

     说明

     每个轴只包括一个 MC_Power 指令。

3. 插入其它运动指令，以生成所需的运动。

说明

将脉冲发生器组态为信号板输出：选择 CPU 的“脉冲发生器 (PTO/PWM)”(Pulse generators (PTO/PWM)) 属性（在“设备组态”(Device configuration) 中）并启用脉冲发生器。每个 S7 - 1200 CPU V1.0、V2.0、V2.1 及 V2.2 都可使用两个脉冲发生器。S7 - 1200 CPU V3.0 及 V4.0 CPU 可使用四个脉冲发生器。在相同组态区域的“脉冲选项”(Pulse options) 下，选择用作以下用途的脉冲发生器：“PTO”。

说明

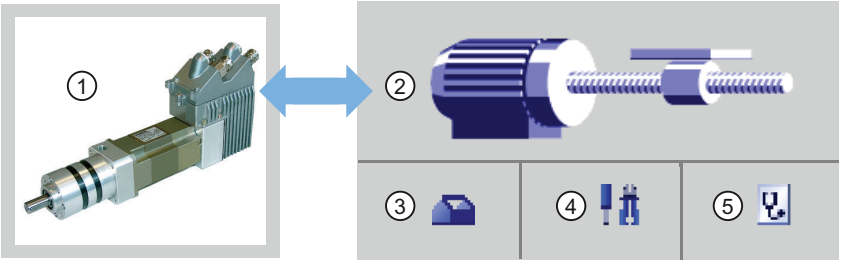
CPU 以 10 ms 为“时间片”或时间段计算运动任务。执行一个时间片时，下一时间片会在队列中等待执行。如果中断某个轴上的运动任务（通过执行该轴的其它新运动任务实现），新运动任务要等待长达 20 ms（当前时间片的剩余时间加上排队的时间片）后才可执行。

10.3.3 开环运动控制

10.3.3.1 组态轴

通过 PTO（脉冲串输出）在 PLC 和驱动器上连接开环轴。对于使用 PTO 的运动控制应用，CPU 需要使用板载或信号板 (SB) 数字量 I/O。这会限制小型 PLC 可使用的轴的最大数量。

STEP 7 为“轴”工艺对象提供组态工具、调试工具和诊断工具。



① 驱动器
② 工艺对象
③ 组态
④ 调试
⑤ 诊断

说明

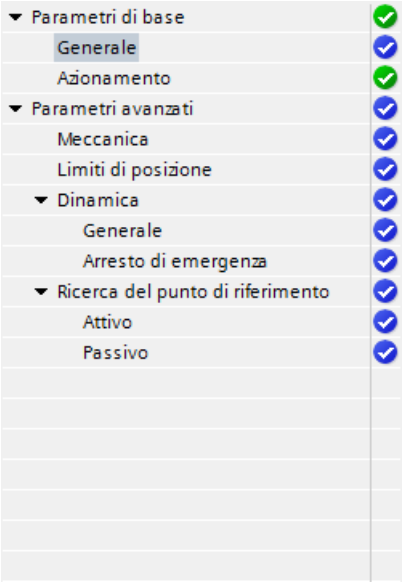
对于固件版本 V2.2 以及更早版本的 CPU，PTO 需使用高速计数器 (HSC) 的内部功能。也就是说，在其它地方无法使用相应的 HSC。

PTO 和 HSC 间的分配是固定的。如果激活 PTO1，则它将与 HSC1 连接。如果激活 PTO2，则它将与 HSC2 连接。生成脉冲时，不能监视当前值（例如，在 ID1000 中）。

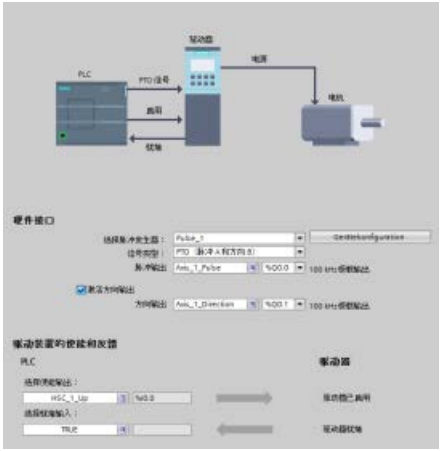
S7-1200 V3.0 以及更高版本的 CPU 则无此限制；当在这些 CPU 中组态脉冲输出时，所有 HSC 仍可供程序使用。

表格 10-39 用于运动控制的 STEP 7 工具

工具	说明
组态	组态“轴”工艺对象的下列属性： ● 要用的 PTO 的选择以及驱动器接口的组态 ● 机械的属性和驱动器（机器或系统）的传动比参数 ● 位置限制属性、动态属性和归位属性 在工艺对象的数据块中保存组态数据。
调试	无需创建用户程序即可测试轴的功能。启动该工具时，将显示控制面板。控制面板上提供了下列命令： ● 启用和禁用轴 ● 在点动模式下移动轴 ● 以绝对和相对方式定位轴 ● 使轴归位 ● 确认错误信息 可以为运动命令指定速度以及加速度/减速度。控制面板中还将显示当前的轴状态。
诊断	监视轴和驱动器的当前状态和错误信息。



PTO 轴的树选择器不包括编码器、模数、位置监视和控制回路组态菜单。

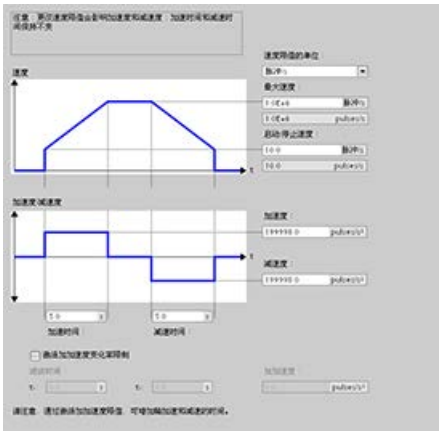


为轴创建工艺对象之后，通过定义基本参数（如 PTO 和驱动器接口的组态）来组态该轴。还可以组态轴的其它属性，例如位置限制属性、动态属性和归位属性。

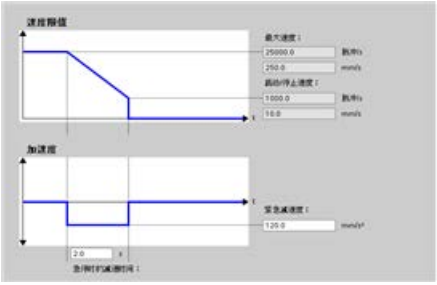
说明

在用户程序中可以根据新量纲单位调整运动控制指令的输入参数值。

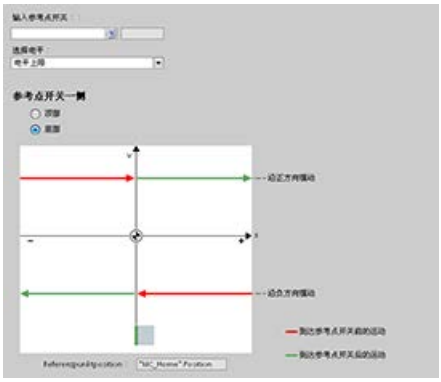
组态驱动器信号、驱动器机械装置和位置监视（硬件和软件限位开关）的属性。

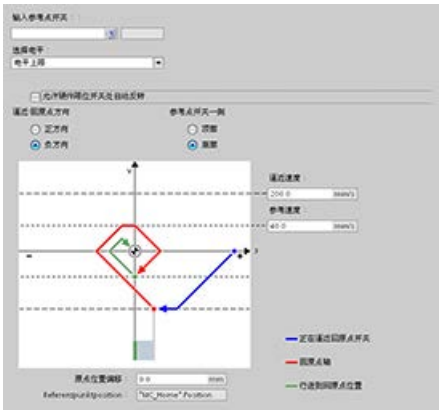


组态急停命令的运动动态和行为。



还可以组态回原点行为（被动和主动）。





使用“调试”(Commissioning) 控制面板独立于用户程序对功能进行测试。

单击“启动”(Startup) 图标对轴进行调试。

控制面板会显示轴的当前状态。不仅可以启用和禁用轴，还可以测试轴定位（以绝对和相对方式）以及指定速度、加速度和减速度。还可以测试归位和点动任务。控制面板还可用于确认错误。

10.3.3.2 调试

“状态和错误位”诊断功能

诊断功能“状态和错误位”(Status and error bits) 用于监视轴的最重要状态和错误消息。当轴激活时，可以在“手动控制”模式和“自动控制”模式下在线显示诊断功能：

表格 10-40 轴的状态

状态	说明
启用	轴已启用且准备就绪，可通过运动控制任务进行控制。 （工艺对象的变量：<轴名称>.StatusBits.Enable）
已回原点	轴已回原点，可执行运动控制指令“MC_MoveAbsolute”的绝对定位任务。对于相对回原点而言，轴不必回原点。特殊情况： ● 主动回原点期间，该状态为 FALSE。 ● 如果回原点的轴经受被动回原点，则在被动回原点期间该状态设置为 TURE。 （工艺对象的变量：<轴名称>.StatusBits.HomingDone）
错误	“轴”工艺对象发生错误。有关错误的更多信息，请参见自动控制模式下运动控制指令的 ErrorID 和 ErrorInfo 参数。在手动模式下，控制面板中的“上一错误”(Last error) 字段显示更多错误原因信息。 （工艺对象的变量：<轴名称>.StatusBits Error）
控制面板激活	在控制面板中启用了“手动控制”模式。控制面板对“轴”工艺对象具有优先控制权。不能通过用户程序来控制轴。 （工艺对象的变量：<轴名称>.StatusBits ControlPanelActive）

表格 10-41 驱动器状态

状态	说明
驱动器准备就绪	驱动器准备好运行。 （工艺对象的变量：<轴名称>.StatusBits.DriveReady）
错误	驱动器在其准备就绪信号故障后报告了错误。 （工艺对象的变量：<轴名称>.ErrorBits.DriveFault）

表格 10-42 轴运动的状态

状态	说明
停止	轴处于停止状态。 （工艺对象的变量：<轴名称>.StatusBits.StandStill）
加速	轴在加速。 （工艺对象的变量：<轴名称>.StatusBits.Acceleration）
恒速	轴在恒速运转。 （工艺对象的变量：<轴名称>.StatusBits.ConstantVelocity）
减速	轴在减速（速度下降）。 （工艺对象的变量：<轴名称>.StatusBits.Deceleration）

表格 10-43 运动模式的状态

状态	说明
定位	轴会执行运动控制指令“MC_MoveAbsolute”或“MC_MoveRelative”或者控制面板的定位任务。 （工艺对象的变量：<轴名称>.StatusBits.PositioningCommand）
速度命令	轴会以设定速度执行运动控制指令“MC_MoveVelocity”或“MC_MoveJog”或者控制面板的任务。 （工艺对象的变量：<轴名称>.StatusBits.SpeedCommand）
回原点	轴会执行运动控制指令“MC_Home”或者控制面板的回原点任务。 （工艺对象的变量：<轴名称>.StatusBits.Homing）

表格 10-44 错误位

错误	说明
到达最小软件限位	已到达下限软件限位开关。 （工艺对象的变量：<轴名称>.ErrorBits.SwLimitMinReached）
超出最小软件限位	已超出下限软件限位开关。 （工艺对象的变量：<轴名称>.ErrorBits.SwLimitMinExceeded）
到达最大软件限位	已到达上限软件限位开关。 （工艺对象的变量：<轴名称>.ErrorBits.SwLimitMaxReached）

错误	说明
超出最大软件限位	已超出上限软件限位开关。 （工艺对象的变量：<轴名称>.ErrorBits.SwLimitMaxExceeded）
负硬件限位	已逼近下限硬件限位开关。 （工艺对象的变量：<轴名称>.ErrorBits.HwLimitMin）
正硬件限位	已逼近上限硬件限位开关。 （工艺对象的变量：<轴名称>.ErrorBits.HwLimitMax）
PTO 已使用	另一个轴正在使用相同的 PTO 并且已使用“MC_Power”启用该轴。 （工艺对象的变量：<轴名称>.ErrorBits.HwUsed）
组态错误	错误地组态了“轴”工艺对象，或者在用户程序运行期间错误地修改了可编辑的组态数据。 （工艺对象的变量：<轴名称>.ErrorBits.ConfigFault）
常规错误	发生内部错误。 （工艺对象的变量：<轴名称>.ErrorBits.SystemFault）

“运动状态”诊断功能

诊断功能“运动状态”(Motion status) 用于监视轴的运动状态。可在下表中找到对各运动状态的描述：

表格 10-45 运动状态

状态	说明
实际位置	在“实际位置”(Actual position) 框中，将指示测量的轴位置。如果轴未回原点，则该值是相对于轴启用位置的位置值。 （工艺对象的变量：<轴名称>.ActualPosition）
实际速度	在“实际速度”(Actual velocity) 框中，将指示测量的轴速度。 （工艺对象的变量：<轴名称>.ActualVelocity）
位置设定值	在“位置设定值”(Position setpoint) 框中，将指示测量的轴位置设定值。如果轴未回原点，则该值是相对于轴启用位置的位置值。 （工艺对象的变量：<轴名称>.Position）
速度设定值	在“速度设定值”(Velocity setpoint) 框中，将指示计算的轴速度设定值。 （工艺对象的变量：<轴名称>.Velocity）

状态	说明
目标位置	在“目标位置”(Target position) 框中，将指示活动定位命令或轴命令表的当前目标位置。“目标位置”(Target position) 的值仅在定位命令执行期间有效。 （工艺对象的变量：<轴名称>.StatusPositioning.TargetPosition）
剩余行程距离	在“剩余行程距离”(Remaining travel distance) 框中，将指示活动定位命令或轴命令表的当前剩余行程距离。“剩余行程距离”(Remaining travel distance) 值仅在定位命令执行期间有效。 （工艺对象的变量：<轴名称>.StatusPositioning.Distance）

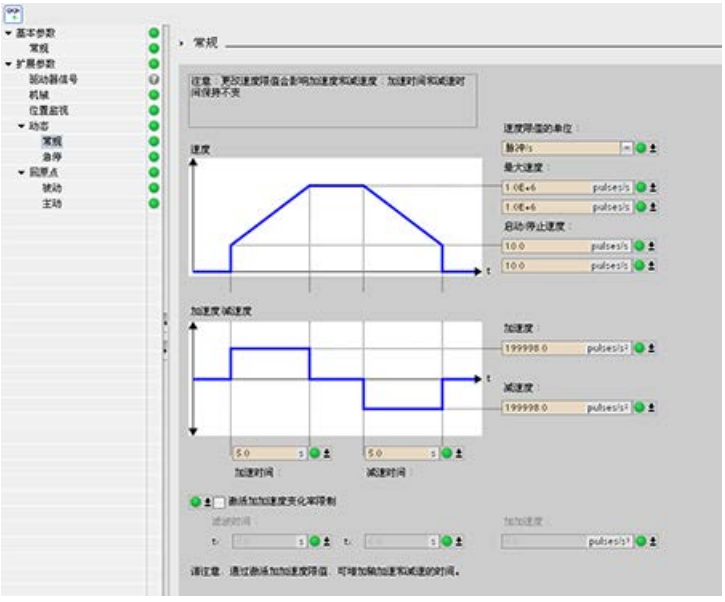
表格 10-46 动态限制

动态限制	说明
速度	“速度”(Velocity) 字段指示组态的最大轴速度。 （工艺对象的变量：<轴名称>.Config.DynamicLimits.MaxVelocity）
加速度	“加速度”(Acceleration) 字段指示当前组态的轴的加速度。 （工艺对象的变量：<轴名称>.Config.DynamicDefaults.Acceleration）
减速度	“减速度”(Deceleration) 字段指示当前组态的轴的减速度。 （工艺对象的变量：<轴名称>.Config.DynamicDefaults.Deceleration）

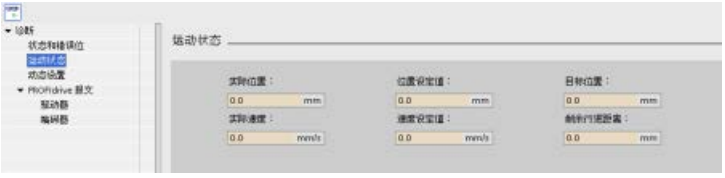
运动起始值控制

您可以编辑运动组态参数的实际值，以便可以在在线模式下优化过程的特性。

打开运动控制的“工艺对象”(Technology objects) 及其“组态”(Configuration) 对象。要访问起始值控制，单击此对话框左上角的“眼镜图标”：

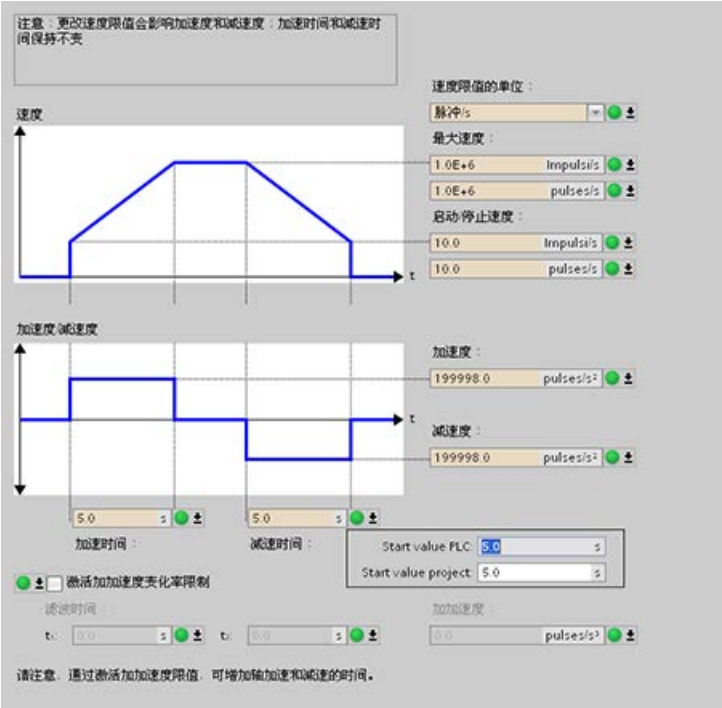


使用“运动状态”(Motion status) 诊断功能可监视轴的运动状态。当轴激活时，可以在“手动控制”模式和“自动控制”模式下在线显示诊断功能。



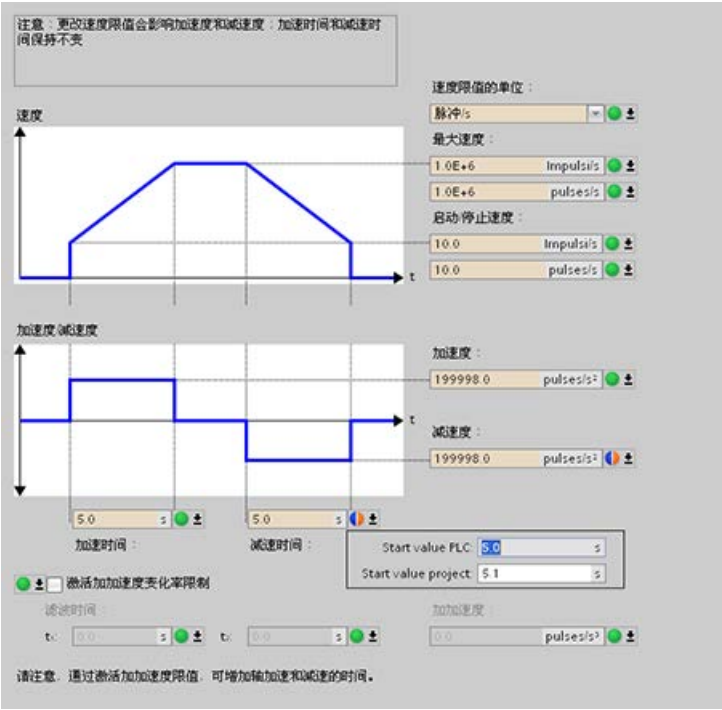
现在可以更改任意运动控制组态参数的值，如下图所示。

可以将实际值与每个参数的项目（离线）起始值和 PLC（在线）起始值进行比较。这对于比较工艺对象数据块 (TO-DB) 的在线/离线差异以及了解在 PLC 下一次“停止到开始”转换时哪些值将用作当前值很有必要。此外，比较图标还会通过视觉指示帮助您轻松确定在线/离线差异：



上图展示了带有比较图标的运动参数画面，其中显示了在线和离线项目之间有哪些值存在差异。绿色图标表示值相同，蓝色/橙色图标表示值不同。

另外，单击带有向下箭头的参数按钮，可打开一个显示每个参数的项目（离线）起始值和 PLC（在线）起始值的小窗口。



10.3.4 闭环运动控制

10.3.4.1 组态轴

通过以下两个连接其中之一连接 PLC 上的闭环轴和驱动器：

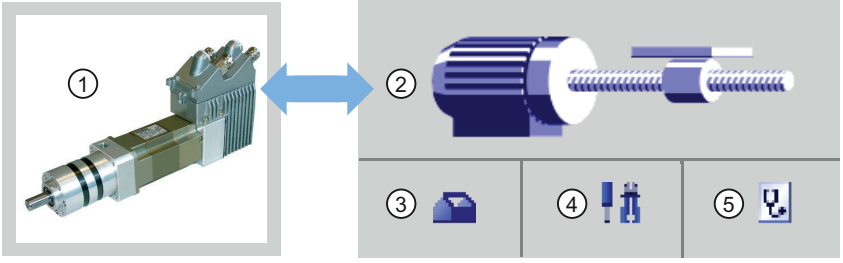
- 模拟驱动器：此连接可使用板载、SB 或 信号模块 (SM) 模拟量 I/O；无需使用 PTO。以下模拟量 I/O 分辨率可用于应用：
  - 板载 I/O：10 位（最低 I/O 分辨率）
  - 信号板 (SB) I/O：12 位
  - SM I/O：14 位（最高 I/O 分辨率）
- PROFIdrive：此连接为网络解决方案，无需使用 PTO。

闭环轴也需要编码器。可将编码器连接到：

- 驱动器上的编码器接口
- 高速计数器 (HSC)
- 工艺模块 (TM)
- PROFINET/PROFIBUS 上的 PROFIdrive 编码器

对于 PROFIdrive 或模拟驱动器连接，最多可连接八个驱动器（或轴）。

STEP 7 为“轴”工艺对象提供组态工具、调试工具和诊断工具。



① 驱动器
② 工艺对象
③ 组态
④ 调试
⑤ 诊断

表格 10-47　用于闭环运动控制的 STEP 7 工具

工具	说明
组态	组态“轴”工艺对象的下列属性： ● 要使用的模拟驱动器接口或 PROFIdrive 的选择以及驱动器和编码器接口的组态 ● 机械的属性和驱动器与编码器（机器或系统）的传动比参数 ● 位置限制属性、动态属性和归位属性 在工艺对象的数据块中保存组态数据。
调试	无需创建用户程序即可测试轴的功能。启动该工具时，将显示控制面板。控制面板上提供了下列命令： ● 启用和禁用轴 ● 在点动模式下移动轴 ● 以绝对和相对方式定位轴 ● 使轴归位 ● 确认错误信息 可以为运动命令指定速度以及加速度/减速度。控制面板中还将显示当前的轴状态。
诊断	监视轴和驱动器的当前状态和错误信息。

说明

在用户程序中可以根据新量纲单位调整运动控制指令的输入参数值。

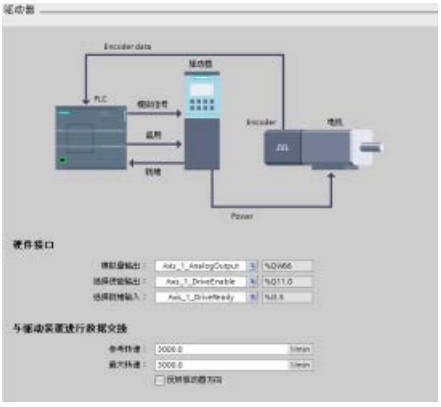
为轴创建工艺对象之后，通过定义基本参数（模拟驱动器或 PROFIdrive 连接和驱动器及编码器的组态）来组态该轴。

模拟驱动器或 PROFIdrive 连接的树选择器包括编码器、模数、位置监视和控制回路组态菜单。

模拟驱动器接口组态

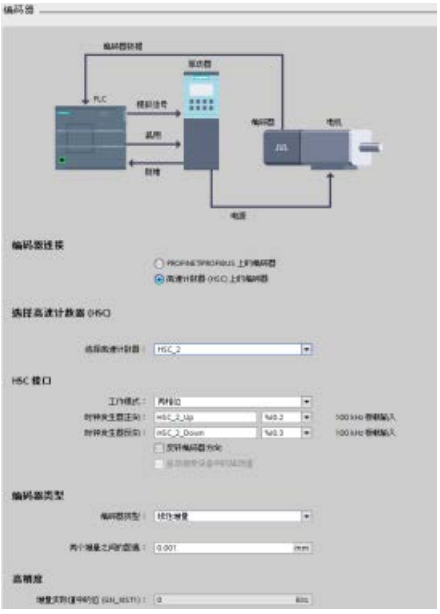
在“常规”(General) 组态对话框中，选择下列参数：

- “模拟驱动器接口”(Analog drive connection) 单选按钮
- 计量单位



在“驱动器”(Drive) 组态对话框中，选择下列参数：

- 模拟驱动器硬件输出
- 数据交换驱动器速度
  注：最大速度必须大于或等于基准（标称）速度。



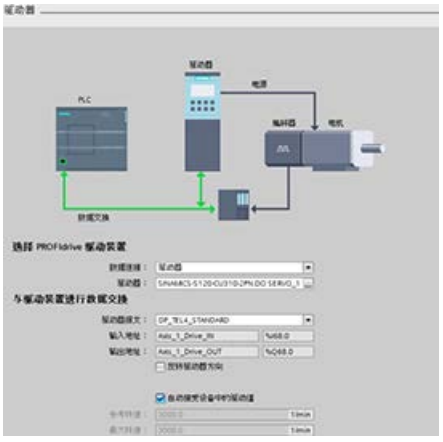
在“编码器”(Encoder) 组态对话框中，选择下列参数：

- 模拟驱动器编码器耦合（例如，高速计数器 (HSC)）
- HSC 接口
- 编码器类型
- 高精度

PROFIdrive 组态

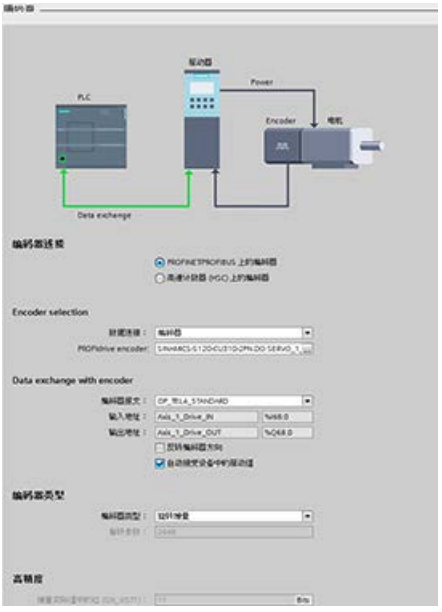
在“常规”(General) 组态对话框中，选择下列参数：

- “PROFIdrive”单选按钮
- 计量单位



在“驱动器”(Drive) 组态对话框中，选择下列参数：

- PROFIdrive 驱动器
- 与驱动器之间的数据交换
  注：最大速度必须大于或等于基准（标称）速度。



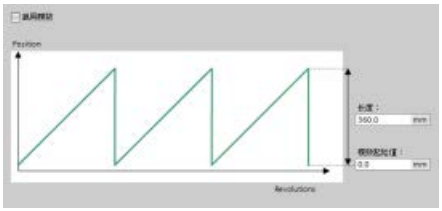
在“编码器”(Encoder) 组态对话框中，选择下列参数：

- PROFIdrive 编码器耦合（例如，PROFINET 上的 PROFIdrive 编码器）
- PROFIdrive 编码器
- 与编码器之间的数据交换
- 编码器类型
- 高精度

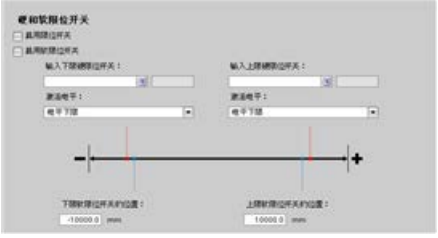
扩展参数

还可以组态闭环轴的以下属性：

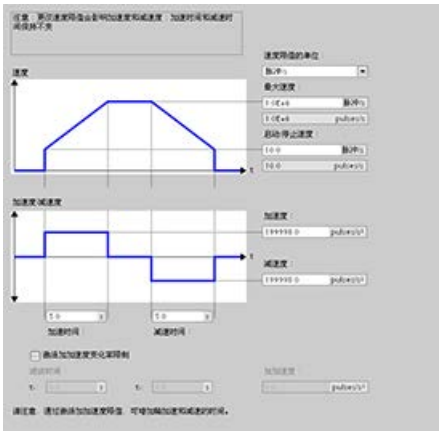
- 模数
- 位置限制
- 动态
- 归位
- 位置监视
- 跟随误差
- 停止信号
- 控制回路



模数：可以组态“模数”轴在循环区域中移动负载，该区域有一个起始值/起始位置和一个给定的长度。如果负载位置达到此区域的终点，则会自动重新设置为起始值。在选中“启用模数”(Enable Modulo) 复选框时，启用“长度”(Length) 和“模数起始值”(Modulo start value) 字段。

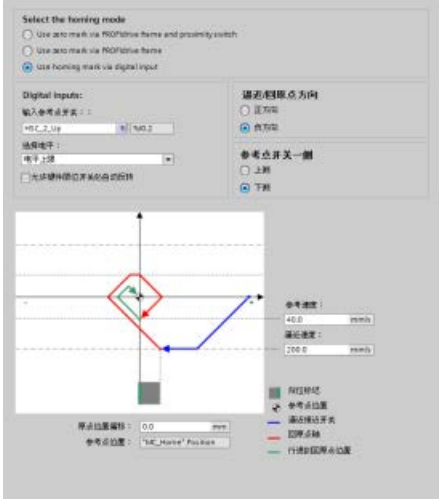


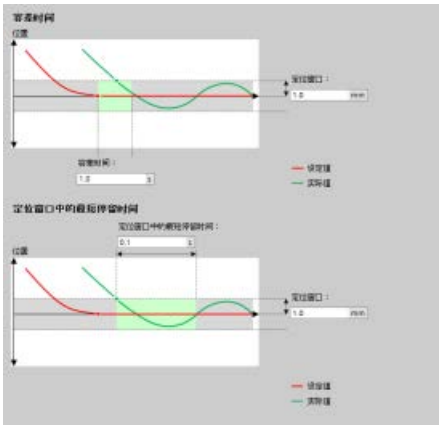
位置限制：可以组态驱动器信号、驱动器机械装置和位置监视（硬限位开关和软限位开关）的属性。



动态：可以组态急停命令的运动动态和行为。

归位：还可以组态归位行为（被动和主动）。



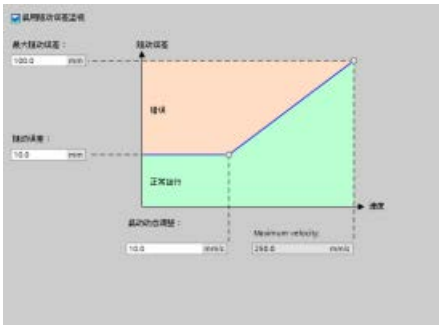


“位置监视”：可以为定位窗口组态容差时间以及最短停留时间。

系统将以下三个参数直接与轴 TO-DB 相连：

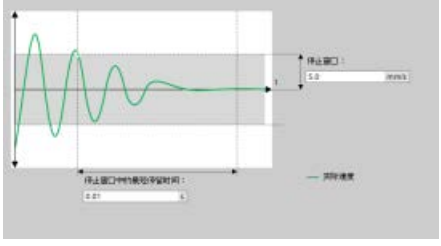
- 定位窗口
- 容差时间
- 在定位窗口停留的最短时间

注：“位置窗口”(Positioning window) 字段的最小值为“0.001”，最大值为“1 E+12”。



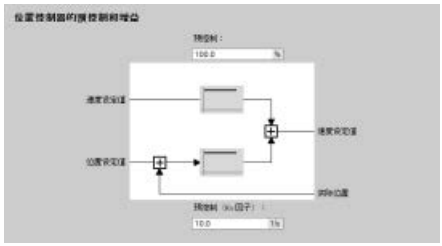
“跟随误差”：可以组态在特定速度范围内允许的距离误差。可选中“启用跟随误差监控”复选框来激活跟随误差。用户可以组态下列参数：

- 最大跟随误差
- 跟随误差
- 启动动态调整
- 最大速度



“停止信号”：用户可以组态下列参数：

- 在停止窗口停留的最短时间
- 停止窗口。



“控制回路”：可以组态被称为“预控制（Kv 因子）”的速度增益。

使用“调试”(Commissioning) 控制面板独立于用户程序对功能进行测试。

单击“启动”(Startup) 图标对轴进行调试。

控制面板会显示轴的当前状态。不仅可以启用和禁用轴，还可以测试轴定位（以绝对和相对方式）以及指定速度、加速度和减速度。还可以测试归位和点动任务。控制面板还可用于确认错误。

10.3.4.2 ServoOB

创建用于 S7-1200 运动控制的工艺对象时，CPU 将自动创建用于处理工艺对象的组织块 "MC-Servo (OB 91)"。工艺对象的运动控制功能可创建自己的优先级别，并且 SIMATIC S7-1200 执行系统将按照应用循环调用 OB。

MC 伺服 OB 受到写保护。无法更改内容。

而在 CPU 上为运动控制组态的所有工艺对象的位置控制算法将在 MC-Servo OB 中进行计算。

可以根据控制质量和系统负载等方面的需求，设定组织块的应用循环和优先级。可将多个 ServoOB 与“CyclicServoEvent”连接。MC-Servo OB 的属性页便会显示两个 CyclicServoEvent 可更改值：

- 优先级编号
- 循环时间

MC-PreServo OB 和 MC-PostServo OB 为 ServoOB 的实例，都属于 ServoOB 类型。这些 OB 为可选项，仅在 MC-Servo OB 存在时才可供使用，并包含用户代码。三个 OB（MC-PreServo、MC-Servo 和 MC-PostServo）都需在同一运行级别运行，并由同一 CyclicServoEvent 触发。PLC 固件会基于 OB 块编号依次执行 OB：

OB 实例	OB 编号	自动化系统对象模型 (ASOM) 事件	工程系统对象模型 (ESOM) 事件
MC-PreServo	67	ServoOB	-
MC 伺服电机	91	ServoOB	ServoOB
MC-PostServo	95	ServoOB	-

与 ASOM 相比，工程系统对象模型 (ESOM) 仅将事件与 MC-Servo OB 连接。ESOM 不会将事件与 MC-PreServo OB 或 MC-PostServo OB 连接。

通过应用周期 MC-Servo (OB 91) 组态轴

应用周期 MC-Servo (OB 91)

可以在组织块的属性中设置应用周期，以在其中调用 MC 伺服 OB：

- 与总线同步：MC-Servo OB 可通过总线系统同时调用。可以在所选总线系统的属性中设置发送时钟。
- 循环：可通过分配的应用周期周期性地调用 MC-Servo OB。

为避免在 CPU 上执行程序时的扰动，请按如下说明根据所用轴数设置应用周期：

应用周期 = 轴数 x 2 ms

轴数	应用周期
1	2 ms
2	4 ms
4	8 ms
8	16 ms

SINAMICS G120 驱动器每 4 ms 更新一次驱动过程映像。为改善控制，请将 MC-Servo (OB 91) 的应用周期设置为 4 ms 或 4 ms 的倍数。

MC-Servo (OB 91) 溢出

所选的应用循环时间必须足够长，才能在一个循环中处理运动控制的所有工艺对象。如果未遵循应用循环，将会发生上溢。

MC-Servo (OB 91) 溢出时，CPU 不会进入 STOP 模式。（TIA Portal 在线帮助中有关 MC-Servo (OB 91) 溢出时会进入 STOP 模式的说法是错误的。）

必要时，可通过时间错误 OB (OB 80) 将 CPU 设置为在 MC-Servo (OB 91) 溢出时进入 STOP 模式。

过程映像分区“OB 伺服 PIP”

如需对运动控制所用的全部 I/O 模块（例如，硬限位开关）实现最优控制，请将这些模块分配给过程映像分区“OB 伺服 PIP”。这样分配后，I/O 模块便可与工艺对象同时进行处理。

运动控制使用高速计数器 (HSC) 时，运动控制会自动将 HSC 分配给过程映像分区“OB 伺服 PIP”。

10.3.4.3 速度控制操作

凭借速度控制操作，用户可以通过“速度控制”移动定位轴。用户可以使用 MC_Power 指令启用轴。利用 MC_MoveVelocity 和 MC_MoveJog 指令，用户可以按速度设定值移动轴。即使传感器出错，没有有效的实际值，也仍可移动轴。“速度控制”模式设定了以下条件：

- 禁用轴的定位控制器
- 直接为驱动器设置速度设定值
- 将轴位置设定值设为零
- 传感器值有效时更新轴的实际位置
- 不定义跟随误差和控制器误差，将其设为零

用户可使用以下三条指令及相关参数来激活和禁用“速度控制”模式：

- MC_Power.StartMode (Int)
- MC_MoveVelocity.PositionControlled (Bool)
- MC_MoveJog.PositionControlled (Bool)

MC_Power

使用 MC_Power 指令，用户可以在“速度控制”模式下启用轴。如果没有有效的传感器值或轴不能切换到“位置控制”模式，则可执行此操作。

只能使用 StartMode“0”和“1”。其它值显示错误：

StartMode	PTO 轴	伺服轴
0	忽略	速度控制
1	忽略	位置控制
其它	无效模式 MC_Power.ErrorID = 0x8412 MC_Power.ErrorID = 0x0011	无效模式 MC_Power.ErrorID = 0x8412 MC_Power.ErrorID = 0x0011

通过“MC_Power.Enable = FALSE”禁用 TO 轴期间，可获取操作模式。根据操作模式，轴会作出不同的响应：

StartMode	模式：位置控制	模式：速度控制
0：急停	轴将根据 ActualPosition 和 ActualVelocity 采用“DynamicDefaults. EmergencyDeceleration”进行位置控制的减速。	轴将根据 ActualVelocity（如果可用）采用“DynamicDefaults. EmergencyDeceleration”进行速度控制的减速。
1：立即停止	驱动器将以所连驱动器内的“AUS3”斜坡停止。	驱动器将以所连驱动器内的“AUS3”斜坡停止。
2：带有加速度变化率控制的紧急停止	轴将根据设定值位置以组态的紧急减速度进行位置控制的制动。如果激活了冲击控制，则不考虑组态的冲击。	轴将根据设定值速度以组态的紧急减速度进行速度控制的制动。如果激活了冲击控制，则不考虑组态的冲击。

驱动器关闭期间不能更改操作模式，因为在驱动器停止前不接受任何新的运动控制命令。驱动器静止后，可以再次启用轴。

MC_MoveVelocity/MC_MoveJog

无论操作模式为何（速度控制/位置控制），块都会将轴切换到已组态的模式。这种情况会在闭环运动控制或停止状态下发生。

MC_Halt

MC_Halt 指令不会更改操作模式。在速度控制模式下，计算得到的减速斜率取决于设定值速度和组态的减速度。

如果到达停止窗口，则该命令完成并显示“Done = TRUE”。

其它运动控制命令

速度控制模式将保持激活状态，直到下列其中一条命令激活为止：

- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity (PositionControlled = TRUE)

- MC_MoveJog (PositionControlled = TRUE)
- MC_Home：
  - 主动回原点（模式 3）
  - 其它模式（被动、直接、调整绝对值传感器）均被拒绝，并提供 ErrorId：
    ErrorId 8207：“命令被拒绝”
    ErrorInfo 006B：“调用速度控制模式被拒绝”。

轴 TODB

如果在速度控制操作模式下执行 MC_Power、MC_MoveVelocity 或 MC_MoveJog 这三条运动控制指令之一，轴 TODB 会显示操作模式。

- Axis.Statusbit.NonPositionControlled = TRUE
- Axis.Position = 0.0

计算新设定值位置

在运动控制期间，模式从速度控制切换到闭环控制后，必须重新计算设定值位置：

- 进行预控制 (kpc > 0) 时：位置 = 实际位置 + 实际速度 * vtc
- 不进行预控制 (kpc = 0) 时：位置 = 实际位置 + 实际速度 / kv（vtc = 代替预控制时间常量；kv = 位置控制增益）

软件限位开关

在速度控制模式下，软件限位开关处于未激活状态。

硬件限位开关

在速度控制模式下，支持硬件限位开关。

10.3.4.4 消息帧 4 支持

PROFdrive 消息帧 4 包含执行器的值和来自两个不同编码器的值。第一个传感器值来自电机编码器。第二个传感器值由机器上的附加编码器提供。

机械编码器与 SINAMICS CU 直接相连，CU 在消息帧 4 中提供这两个传感器值。

轴“驱动器”组态对话框

在硬件组态中组态消息帧 4 后，即可在轴“驱动器”(Drive) 组态对话框中选择使用。

轴“编码器”组态对话框

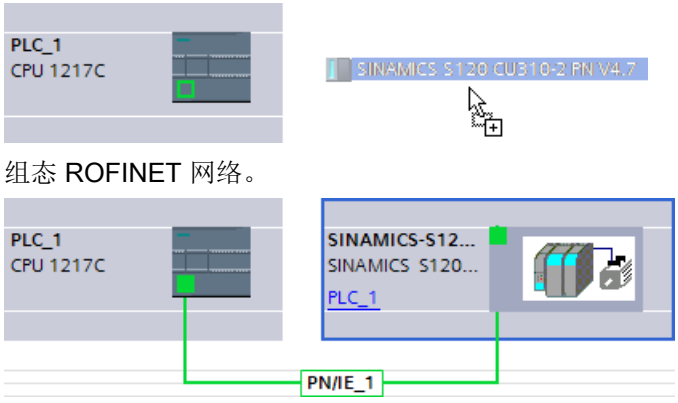
在轴“编码器”(Encoder) 组态对话框中，有两个选项：

- “PROFINET/PROFIBUS 上的编码器”(Encoder on PROFINET/PROFIBUS)
- “高速计数器 (HSC) 上的编码器”(Encoder on high-speed counter (HSC))

“PROFINET/PROFIBUS 上的编码器”(Encoder on high-speed counter (HSC)) 为默认选项，不过两个选项都可选用。

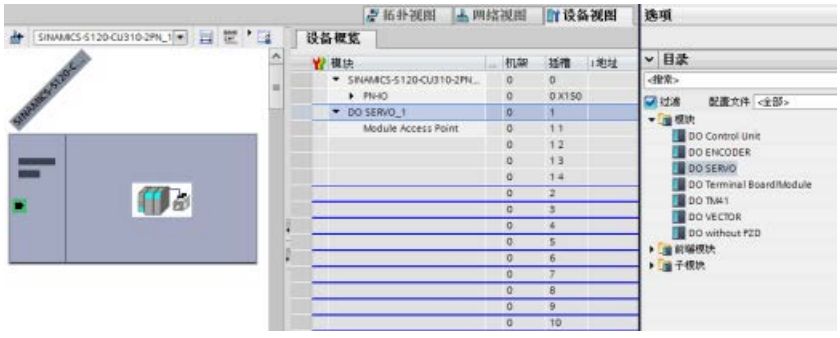
示例：通过消息帧 4 组态 SINAMICS S120 和编码器

1. 选择 SINAMICS 驱动器：
   使用硬件目录添加 SINAMICS S120 CU310-2 PN V4.7 驱动器。为此，请展开以下容器：
   – 其它现场设备
   – PROFINET IO
   – 驱动器
   – SIEMENS AG
   – SINAMICS
   如下图所示插入驱动器：

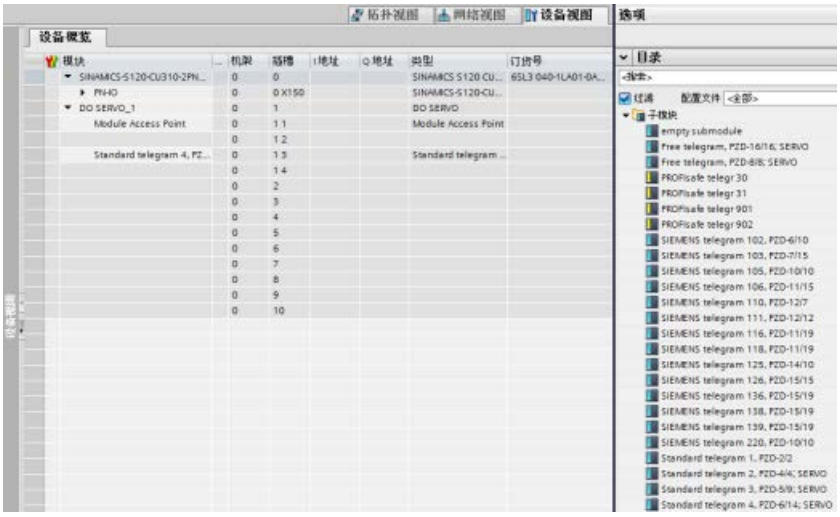


   组态 ROFINET 网络。

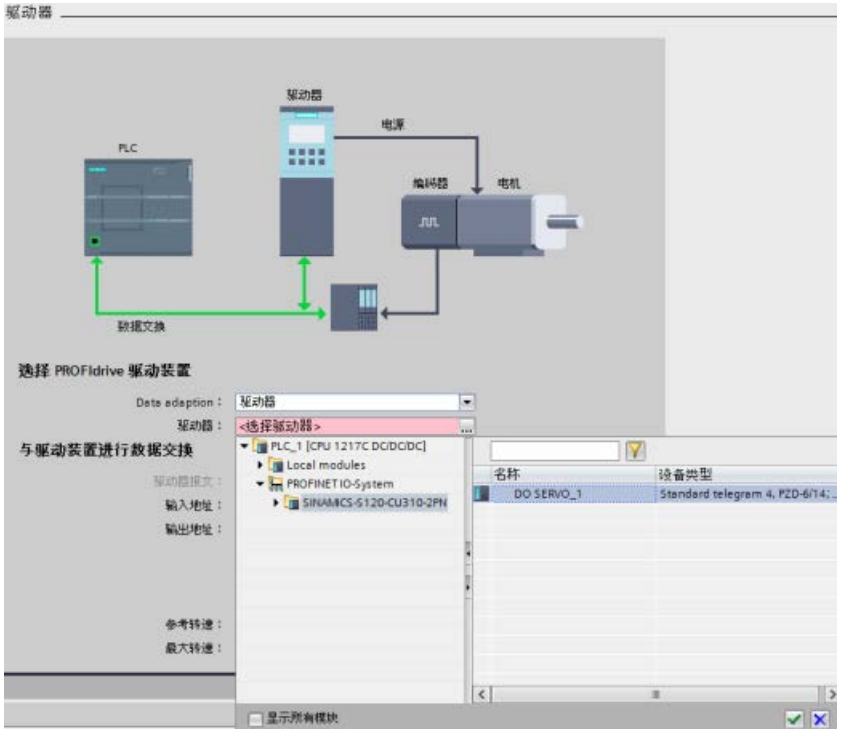
2. 选择 DO SERVO:
   – 从“网络”(Network) 视图中，双击 SINAMICS S120 CU310-2 PN V4.7 驱动器。
   – 打开设备总览。
   – 在硬件目录中，展开“模块”(Module) 容器。
   – 双击或拖动 DO SERVO 驱动器对象，将其插入第一个空白行：



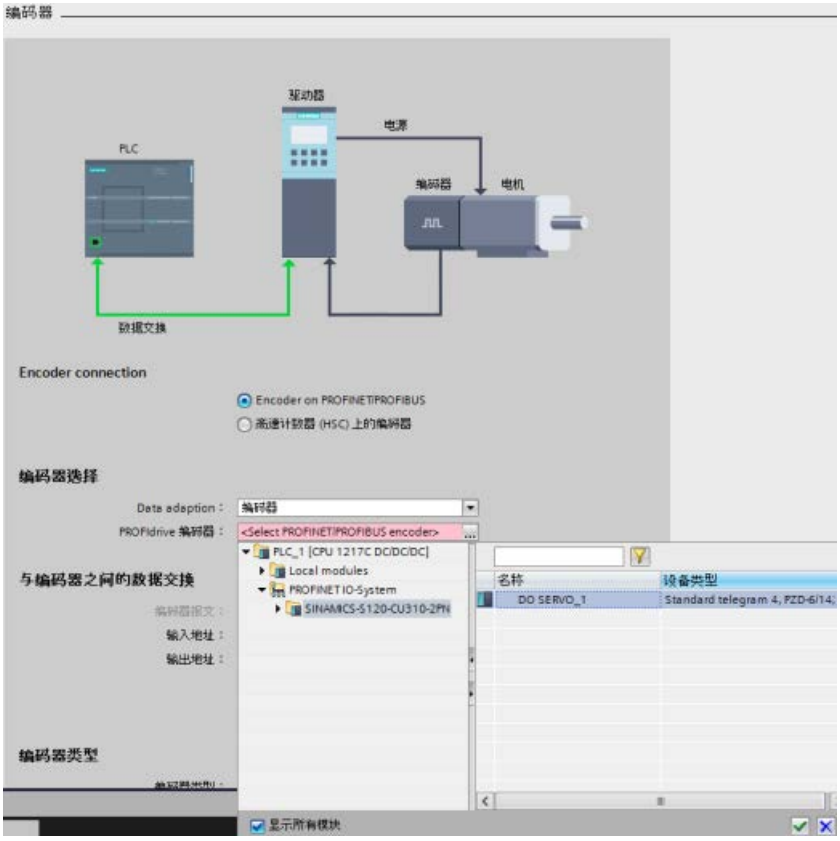
3. 选择消息帧 4：
   - 在硬件目录中，展开“子模块”(Submodules) 容器。
   - 双击或拖动“标准消息帧 4，PZD-6/15;SERVO”(Standard telegram 4, PZD-6/15;SERVO)，将其插入第二个空白行。
   - 必须跳过一个空白行后插入消息帧 4，如下图所示：



4. 在“驱动器”(Drive) 组态对话框中选择编码器：
   - 导航到轴组态对话框，以及“基本参数”(Basic parameters)、“驱动器”(Drive) 组态对话框。
   - 移动到“选择 PROFIdrive 驱动器，驱动器：”(Select PROFIdrive drive, Drive:)字段。
   - 单击省略号按钮。
   - 双击“PROFINET IO 系统”(PROFINET IO-System") 以打开其文件夹。
   - 单击“SINAMICS-S120-CU310-2PN”。
   - 右窗格中将显示“DO SERVO_1：标准消息帧 4，PZD-6/14；SERVO”(DO SERVO_1: Standard telegam 4, PZD-6/14; SERVO)。
   - 单击绿色复选按钮输入组态。



5. 组态编码器：
   - 在轴“驱动器”(Drive) 组态对话框中选择消息帧 4 后，“编码器”(Encoder) 驱动器组态对话框会在导航树中提供一个新的“消息帧中的编码器”(Encoder in telegram) 条目。选择“消息帧中的编码器”(Encoder in telegram) 条目后，右侧窗格会显示两个条目，编码器 1 和编码器 2，及其编码器值。
   - 运动控制分配一个编码器作为驱动器编码器，分配另一个编码器作为机器编码器。选择编码器 1 或编码器 2 时，即确定分配哪个编码器作为驱动器编码器。未选择的编码器被分配作为机器编码器。
   - 选择驱动器编码器后，单击绿色复选按钮输入组态：



10.3.4.5 仿真轴

要想在未连接驱动器的 PLC 上使用 PROFIdrive 或模拟驱动器轴时，可使用仿真模式。

想要执行下列操作时，需要进入仿真模式：

- 在不使用驱动器的条件下调试程序序列
- 在不移动轴的条件下测试用户程序
- 在无实际运动的条件下使用过程模型仿真轴行为。
- 驱动器和编码器不得相连：测试还能够在未连接驱动器的情况下进行；后续将添加并组态驱动器。

组态仿真模式

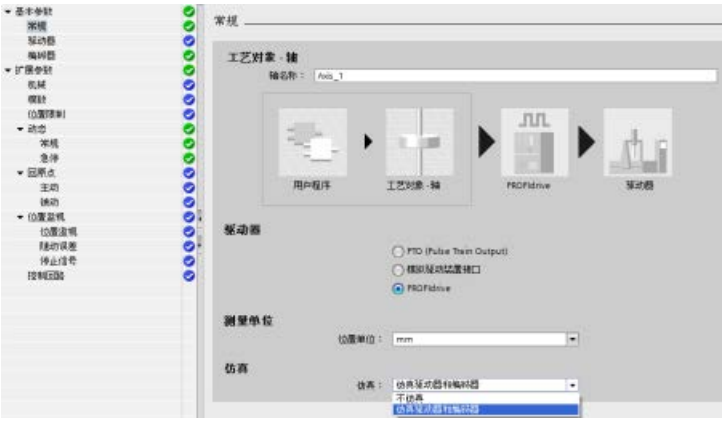
下列选项可用：

TODB 值	对话框条目	注释
0	无仿真	无仿真
1	仿真驱动器和编码器	硬件组态中有/无正确轴和组态 I/O 地址时的仿真

仿真模式只能用于伺服轴，因此，只能使用 PROFIdrive 和模拟驱动器。对于 PTO 轴，仿真模式为“0”。

当轴类型更改为 PTO 时，TIA Portal 需要补充仿真模式的其余部分。另外，对于仿真模式为“0”的 PTO，应选中“数据调整”(Data adaption)，否则，会自动设置。

用户可在“基本参数”(Basic parameters) 对话框 >“常规”(General) 部分 >“仿真”(Simulation) 字段中选择不同仿真模式，如下图所示：



模式 1：“仿真驱动器和编码器”(Simulate drive and encoder)：

- 在此模式下，程序中的轴在 PLC 中执行操作，但并未实际连接硬件，例如 PROFIdrive 驱动器和传感器。
- 用户无需使用工艺对象数据块 (TODB) 中的逻辑 I/O 地址。这意味着，用户不需要在 TO 和硬件组态中组态传感器和驱动器以及其它数字信号（硬件限位开关和参考点开关）。轴不会向传感器和驱动器的逻辑地址提供数据。
- 仿真 TODB 实际速度。
- 对于 PROFIdrive，不使用 PROFIdrive 消息帧。
- 对于 PROFIdrive，PLC 显示了一个关于已组态硬件（外设或驱动器）丢失的诊断错误，但这对轴的应用并无影响。
- 在所有回原点模式下直接创建回原点报告。设置相应的位置和状态。不检测硬件输入。

- 所有连接均可为空。
- 轴控制面板\调节面板按预期运行。
- 支持速度控制轴。
- 不支持虚拟轴（例如，SMC 定义的轴）；不过，可以使用无硬件连接的仿真轴代替虚拟轴。

下载：

PLC 在 RUN 模式下时，可在更改仿真模式后下载轴。在这种情况下，由 PLC 设置 StatusBit RestartRequired。重新启动轴后，PLC 会将相关变更传送到工作存储器。下表列出了与硬件组态（执行器、传感器和 PositionLimit）关联和仿真时必选/可选的所有轴参数：

TODB 参数	模式 1：仿真 PROFIdrive
Actor.Interface.AddressIn	可选
Actor.Interface.AddressOut	可选
Actor.Interface.EnableDriveOutput	可选
Actor.Interface.DriveReadyInput	可选
Sensor.Interface.AddressIn	可选
Sensor.Interface.AddressOut	可选
Sensor.ActiveHoming.DigitalInputAddress	可选
Sensor.PassiveHoming.DigitalInputAddres s	可选
PositionLimits_HW.MinSwitchAddress	可选
PositionLimits_HW.MaxSwitchAddress	可选

10.3.4.6 数据调整

概述

对于可从驱动器或传感器模块中读取，并且已在 PLC 和驱动器/传感器设备中进行相同组态的执行器和传感器数据，用户可以进行调整。

组态 Adaptation RT

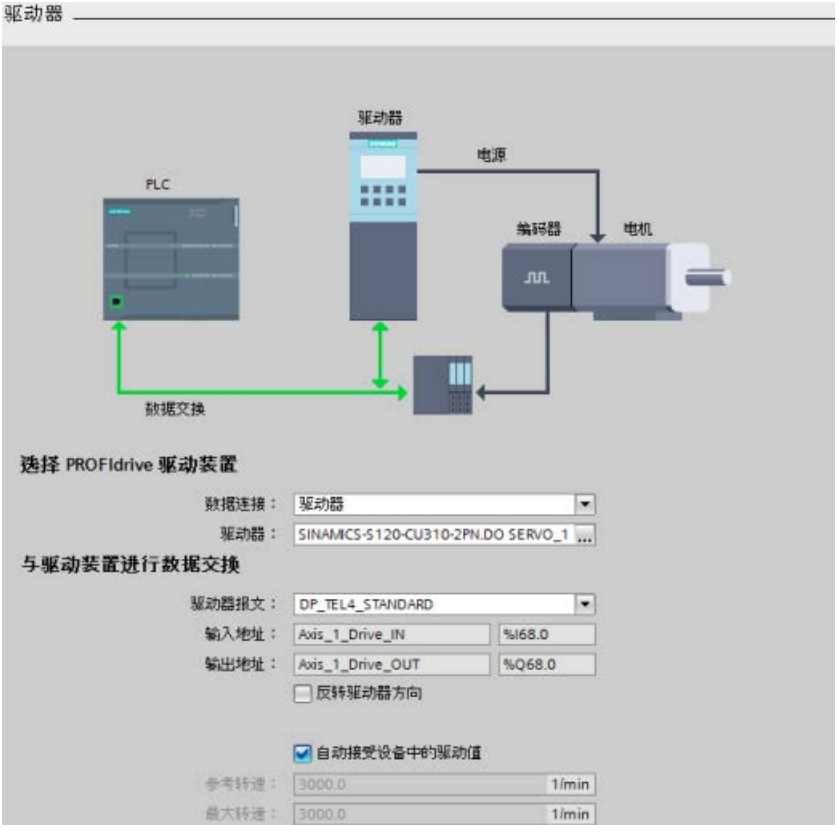
CPU 在 TO-DB 中为指定执行器和传感器组态 Adaptation RT：

- <axis>.Sensor[i].DataAdaptation：DINT [0:NO | 1:YES]
- <axis>.Actor.DataAdaptation：DINT [0:NO | 1:YES]

运行期间，可通过轴组态对话框、“基本参数”(Basic parameters)、“驱动器”(Drive) 和“编码器”(Encoder) 组态对话框更改 Adaptation RT 组态。调整操作在 TO 启动或重启，或者与驱动器通信丢失的情况下均有效。

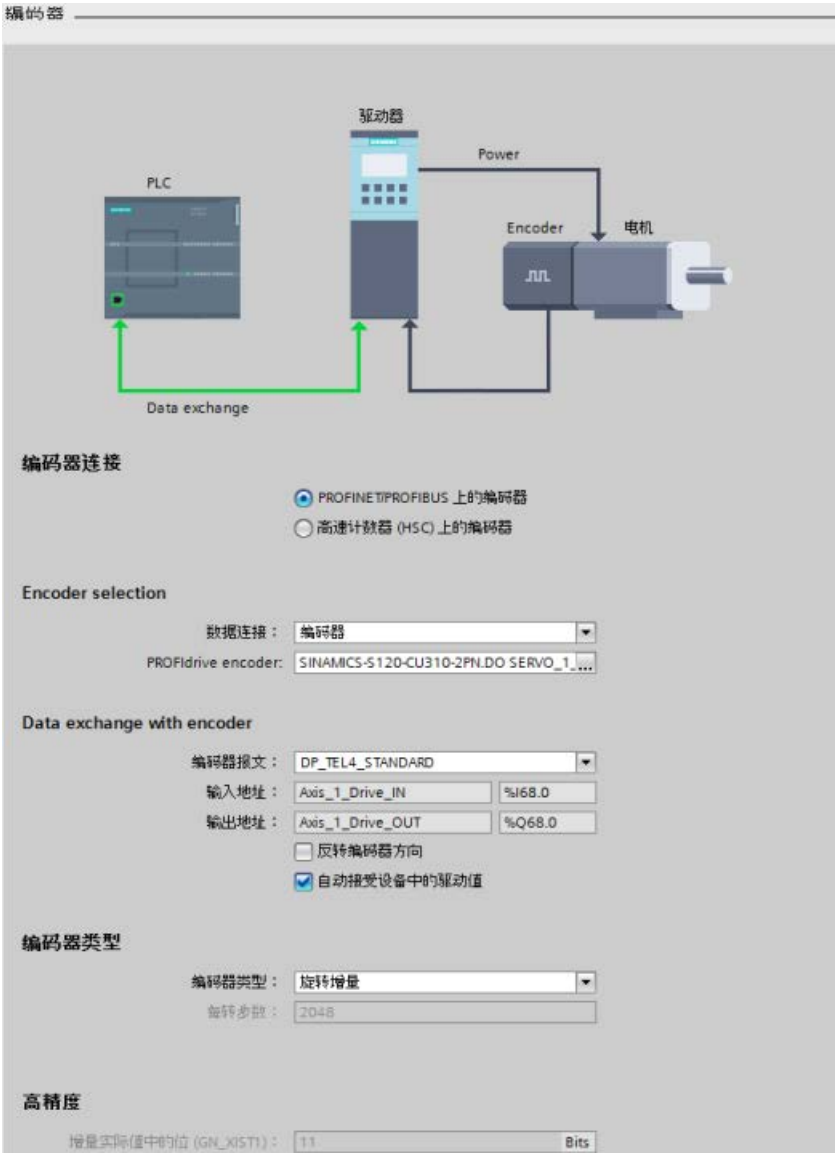
在 TIA Portal 中，使用轴组态对话框、“基本参数”(Basic parameters)、“驱动器”(Drive) 和“编码器”(Encoder) 对话框中的“自动接受设备中的驱动器值”(Automatic acceptance of drive values in the device) 复选框即可选择“数据调整”(Data adaptation)。

下图显示了轴组态“驱动器”(Drive) 对话框：



- 如果已连接 SINAMICS 驱动器：
  - 默认情况下，选中该复选框。
  - “参考速度”(Reference speed) 和 “最大速度”(Maximum speed) 复选框后的控件呈灰显。
  - 执行器数据的 Adaptation RT 已激活。
  - 如果手动取消选中该复选框，则可以更改这两个复选框后的控件值。
- 如果未连接 SINAMICS 驱动器或其它驱动器：
  - 默认情况下，该复选框取消选中并呈灰显。
  - “参考速度”(Reference speed) 和 “最大速度”(Maximum speed) 复选框后的控件激活并且可编辑。
  - 执行器数据的 Adaptation RT 未激活。

下图显示了轴组态 “编码器”(Encoder) 对话框：



- 如果已连接 PROFIdrive 编码器：
  - 默认情况下，选中该复选框。
  - “编码器类型”(Encoder type)（仅限“每转步进数”(Steps per revolution)）和“高精度”(Fine resolution) 复选框后的控件呈灰显。
  - 编码器数据的 Adaptation RT 已激活。
  - 如果手动取消选中该复选框，则可以更改这两个复选框后的控件值。
- 如果未连接 PROFIdrive 编码器（HSC 或 TM 模块）：
  - 默认情况下，该复选框取消选中并呈灰显。
  - “编码器类型”(Encoder type) 和“高精度”(Fine resolution) 复选框后的控件激活并且可编辑。
  - 执行器数据的 Adaptation RT 未激活。

执行 Adaptation RT

激活 Adaptation RT 后即可执行调整：

- 启动工艺对象 (TO)（或 PLC 启动）或创建 TO（下载 TO-DB 时）期间
- 驱动器或编码器断电（或通信丢失）后再次启动
- 重启 TO（每个重启数据经过调整后）

在 Adaptation RT 有效通信期间，运动控制会拒绝用户程序中关于此驱动器的通信命令，并显示用户程序中的资源错误。

在 Adaptation RT 通信过程中，驱动器断开会中断调整。当驱动器再次响应时，便会再次启动调整。

如果设备选择负向确认，则运动控制不会覆盖 TO 轴组态。如果选择了 Adaptation RT 但系统不能正常运行调整，运动控制会显示错误并且设备不能启用。

显示调整状态和错误

运动控制会显示 Adaptation RT 状态和参数错误以及 ErrorIDs/ErrorInfos：

- 运动控制显示轴的调整状态：如果不能运行调整或运行时出错，运动控制会显示轴错误以及 MC_Power 指令的 ErrorID/ErrorInfo。
- 常规调整参数：制造商、设备、版本和日期均可从 P964[7] 参数读出。P964 参数为 PROFIdrive 参数，可从执行器和编码器获取。
- 性能：运动控制在启动或重启后执行 Adaptation RT。如果之后不更改组态，考虑到系统运行性能，需要禁用数据调整。共有两种解决方案：
  - 调整数值后通过在 RT 中应用来保存数值，然后禁用调整。
  - 调整数值后将其上传，将这些值传输到项目中后禁用调整。
- 调整行为：
  - 启动/重启期间，所有待调整的已组态设备（执行器和传感器）均得到调整。
  - 如果设备在此步骤中不可调整，运动控制会显示错误并且状态会变为“ADAPTATION_ERROR”。
  - 当某个编码器出错时，仍可在速度控制模式下启用轴，因为该模式通常不需要使用编码器。

关于数据调整 ErrorID 和 ErrorInfo 的列表，请参见“运动控制 ErrorID 和 ErrorInfo”。

调整执行器数据

执行器数据的调整特定于设备，仅受 SINAMICS 驱动器支持。SINAMICS 驱动器支持数据调整，并在 Adaptation RT 激活时显示一个错误。

单位

调整执行器数据时需考虑驱动器中的组态单位。数值和单位取决于 DO 类型、DO 功能模块和单位系统。

执行器数据调整仅支持旋转驱动器（非线性驱动器）和 1/min 单位。

读取并评估以下参数：

- p107（DO 类型）
- p108（DO 功能模块）
- p505（SI 单位或 US 单位）

从驱动器选择的当前数据记录

SINAMICS 驱动器支持不同的数据集用于编码器和执行器。SINAMICS 驱动器在启动调整时会调整当前数据记录。因此，会读出“p51”（驱动器的当前数据记录）：

- 参考速度等于所有数据记录：“p2000”（参考速度）与数据记录无关。
- 最大速度与数据记录无关：“p1082”
- 如果当前数据记录中的控制值一致，SINAMICS 驱动器不会检查不同数据记录的最大值是否不同。例如，对其它数据记录进行数据调整后会出现这种情况。

电机类型

SINAMICS 支持两种电机类型：

- 线性电机
- 默认电机（旋转电机）

采用基本运动控制 (BMC) 时，SINAMICS 仅支持旋转电机。

如果 SINAMICS 驱动器组态了一个线性电机（参数“r108，位 12”），运动控制会在进行一致性检查或数据调整中止后显示一个 ErrorID。

参数

运动控制支持“DO-Servo”和“DO-Vector”。调整驱动器的以下参数：

TO-DB 执行器参数	SINAMICS 参数
Actor.DriveParameter.ReferenceSpeed	p2000
Actor.DriveParameter.MaxSpeed	p1082

不会调整执行器的三个 TO DB 参数，但会检查其正确性：

TO-DB 传感器参数	SINAMICS/PROFIdrive 参数
Actor.type	r108，位 12
Actor.Interface.AddressIn.RID	p922 或 p2079
Actor.Interface.AddressOut.RID	p922 或 p2079

下表列出了每个参数的值：

TO-DB 执行器参数	值
Actor.type	• 0 = 模拟 • 1 = PROFIdrive • 2 = PTO 注：TIA Portal 仅支持旋转驱动器。这意味着，只有值“1”有效。
Actor.Interface.AddressIn.RID	• 0208_0708 = 消息帧 1 • 0208_070A = 消息帧 2 • 0208_070C = 消息帧 3 • 0208_0720 = 消息帧 4 • 0208_070E = 消息帧 81 • 0208_0710 = 消息帧 83
Actor.Interface.AddressOut.RID	• 0208_0709 = 消息帧 1 • 0208_070B = 消息帧 2 • 0208_070D = 消息帧 3 • 0208_0721 = 消息帧 4 • 0208_070F = 消息帧 81 • 0208_0711 = 消息帧 83

SINAMICS/PROFIdrive 参数	值
r108，位 12	• 0 = 线性驱动器 • 1 = 旋转驱动器 注：TIA Portal 仅支持旋转驱动器。
p922 或 p2079	• 1 = 消息帧 1 • 2 = 消息帧 2 • 3 = 消息帧 3 • 4 = 消息帧 4 • 81 = 消息帧 81 • 83 = 消息帧 83

检查最大速度

TIA Portal 会检查 TO-DB“DynamicLimits.MaxVelocity”参数的有效性。不过，通过数据调整，只有在系统处于 RUN 模式并且不执行 TIA Portal 验证时才能进行检查。verification.

一致性检查

运动控制会在首次启动或重启 TO 轴期间执行一致性检查。如果执行器的数据调整已激活，运动控制还会显示一个 ErrorID。一致性检查涉及到消息帧、电机类型和最大速度：

- “p922”或“p2079”中的消息帧：如果 TO 和驱动器组态不一致，运动控制会显示错误。
- DO 伺服的“r108，位 12”中的电机类型：如果 TO 和驱动器组态不一致，运动控制会显示错误。
- 使用组态的轴参数不能达到最大速度，此时运动控制会显示错误。（注：如果未更改轴组态，该错误经确认后运动控制不会再显示。）

特例：

- 最大速度 (p1082) > 2 x 参考速度 (p2000)
- 运动控制会在内部将最大速度降至“2x”参考速度。轴组态中的条目不受限制。不过，运动控制会调整“p1082”的值并显示一个错误。

调整传感器数据

仅支持对有效的编码器进行数据调整 (p0979)。

已调整的数据量

运动控制会调整“p0979”的实际值描述（包括旋转编码器或线性编码器组态），并检查 TIA Portal 编码器类型参数“增量或绝对值”。运动控制不会调整或评估消息帧 83 中的 NIST 参考值。

将消息帧中的编码器与 PLC 和驱动器中的编码器关联

- 运动控制在 PLC 中将编码器的映射组态为两个 TO-DB 参数的 VREF.RID 中消息帧中的实际值（实际值 1 或实际值 2）：
  - <axis>.Sensor[i].Interface.AddressIn
  - <axis>.Sensor[i].Interface.AddressOut
- 驱动器中的映射使用“p979”中的索引设置进行（SINAMICS 中的编码器建模）。

参数

运动控制会调整以下参数：

TO-DB 传感器参数	SINAMICS/PROFIdrive 参数
Sensor[i].System	• P979.[1] 或 P979.[11] • 位 0 = 0：旋转编码器 • 位 0 = 1：线性编码器
增量旋转传感器	
Sensor[i].Parameter.StepsPerRevolution	P979.[2] 或 P979.[12]
Sensor[i].Parameter.FineResolutionXist1	P979.[3] 或 P979.[13]
增量线性传感器	
Sensor[i].Parameter.Resolution	P979.[2] 或 P979.[12]
Sensor[i].Parameter.FineResolutionXist1	P979.[3] 或 P979.[13]
绝对值旋转传感器	
Sensor[i].Parameter.StepsPerRevolution	P979.[2] 或 P979.[12]
Sensor[i].Parameter.FineResolutionXist1	P979.[3] 或 P979.[13]
Sensor[i].Parameter.DeterminableRevolutions	P979.[5] 或 P979.[15]
Sensor[i].Parameter.FineResolutionXist2	P979.[4]，bzw.P979.[14]
绝对值线性传感器	
Sensor[i].Parameter.Resolution	P979.[2] 或 P979.[12]

TO-DB 传感器参数	SINAMICS/PROFIdrive 参数
Sensor[i].Parameter.FineResolutionXist1	P979.[3] 或 P979.[13]
Sensor[i].Parameter.FineResolutionXist2	P979.[4]，bzw.P979.[14]

运动控制不会调整一个 TO-DB 传感器参数，但会检查其一致性：

TO-DB 传感器参数	SINAMICS/PROFIdrive 参数
Sensor[i].Type	• P979.[5]，bzw.P979.[15] • 位 0 = 0：增量编码器 • 位 0 > 1：绝对值编码器
Sensor[i].Interface.AddressIn Sensor[i].Interface.AddressOut	p922 或 p2079

绝对值编码器可用作增量编码器，但增量编码器不能用作绝对值编码器。运动控制显示 TO 编码器类型的不一致情况及其在 PROFIdrive 消息帧中的相关实际值错误。注：使用 SINAMICS FW V2.6 时，PROFIdrive 接口有一个传感器零位标记。

一致性检查

运动控制会在首次启动或重启 TO 轴期间执行一致性检查。如果传感器的数据调整已激活，运动控制还会显示一个 ErrorID。一致性检查涉及到消息帧和传感器类型：

- “p922”或“p2079”中的消息帧：如果 TO 和传感器组态不一致，运动控制会显示一个错误。
- “P979.[5]”或“P979.[15]”中的传感器类型：绝对值编码器可用作增量编码器，但增量编码器不能用作绝对值编码器。如果存在不一致，运动控制会显示一个错误。

必须上传的参数

下表列出了必须从驱动器上传到 TIA portal 中的所有参数。要管理这种数据调整，必须扩展 TO-DB。可使用 TO-DB 的以下结构来执行此次扩展：

执行器	类型	默认值	可更改	注释
Actor.DataAdaptation	DINT	0	R（通过重启）	调整激活： • 0：否 • 1：是

StatusDrive	类型	注释
StatusDrive.AdaptationState	DINT	调整状态： • 0：NOT_ADAPTED：无法接收数据。 • 1：IN_ADAPTATION：数据调整刚开始。 • 2：ADAPTED：数据已调整 • 3:NOT_APPLICABLE：未选择调整或者该操作不可用于此驱动器 • 4:ADAPTATION_ERROR：调整期间出错： – 轴无法启用。 – 运动控制显示一个组态错误。

传感器	类型	默认值	可更改	注释
Sensor.DataAdaptation	DINT	0	R（通过重启）	调整激活： • 0：否 • 1：是

StatusSensor	类型	注释
StatusSensor.AdaptationState	DINT	调整状态： • 0：NOT_ADAPTED：无法接收数据。 • 1：IN_ADAPTATION：数据调整刚开始。 • 2：ADAPTED：数据已调整。 • 3：NOT_APPLICABLE：未选择调整或者该操作不可用于此驱动器。 • 4：ADAPTATION_ERROR：调整期间出错： – 轴无法启用。 – 运动控制显示一个组态错误。

ErrorWord	类型	注释
...	Bool	
位 15：调整错误	Bool	ErrorID：
...	Bool	

10.3.4.7 使用 TM 脉冲模块进行的轴控制

TM 脉冲模块是 ET 200SP 双通道脉冲输出模块，可与阀和电机配合使用。该模块可支持两个电流为 2 A 的 24 V DC 通道或一个电流为 4 A 的 24 V DC 通道。

可使用 TM 脉冲 2x24V 输出模块的“直流电机”操作模式，通过双极性 PWM 输出以双向驱动电机。可以为电机分配一个数字量输入作为“外部停止”信号。

TM 脉冲模块可执行以下功能来支持运动控制：

- 可编程输出对 CPU/主站 STOP 条件的响应
- 故障检测和诊断：
  - L+ 电源电压缺失或处于欠压状态
  - 数字量输出短路/过载
  - 传感器电源短路/欠压
  - 过温故障
  - 参数化故障
  - 模块/固件错误

示例

某些应用需要能够轻松移动到指定位置，但无需绝对精确和遵守特定规约。可通过 ET 200SP TM 脉冲模块结合使用公共直流电机和闭环伺服控制，以便对电机进行控制。该应用示例涉及 ET 200SP TM 脉冲模块的以下模式：“带直流电机的 PWM”。关于其它 ET 200SP TM 脉冲模式的更多信息，请参见《*ET 200SP 工艺模块 TM 脉冲 2x24V 手册*》。

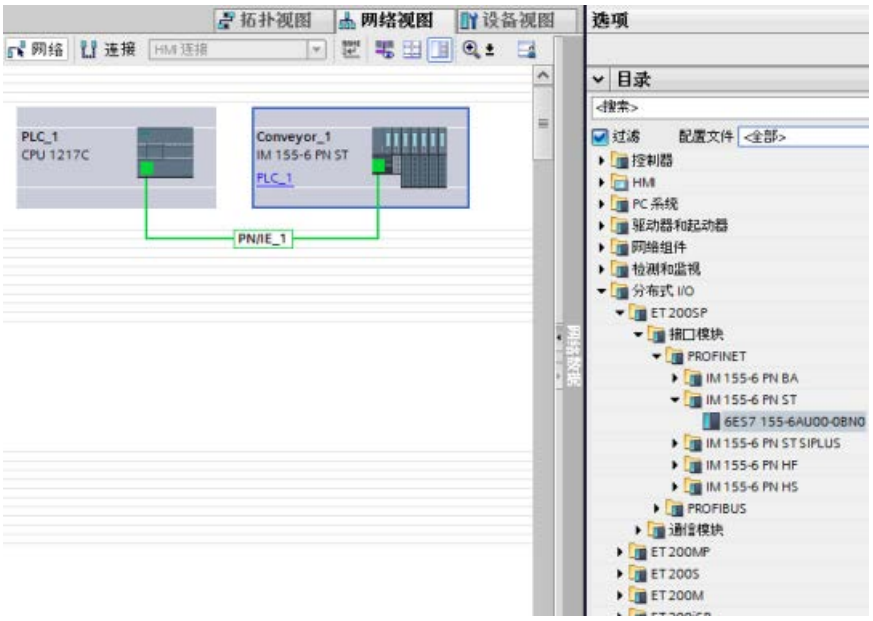
要使用实际的闭环控制，需要采用一种位置反馈方法。必须将编码器连接至电机，以向控制系统提供反馈。该示例提供了三种实现方法：

- 内置于 S7-1200 中的高速计数器 (HSC)
- ET 200SP TM 计数模块
- ET 200SP TM PosInput 模块

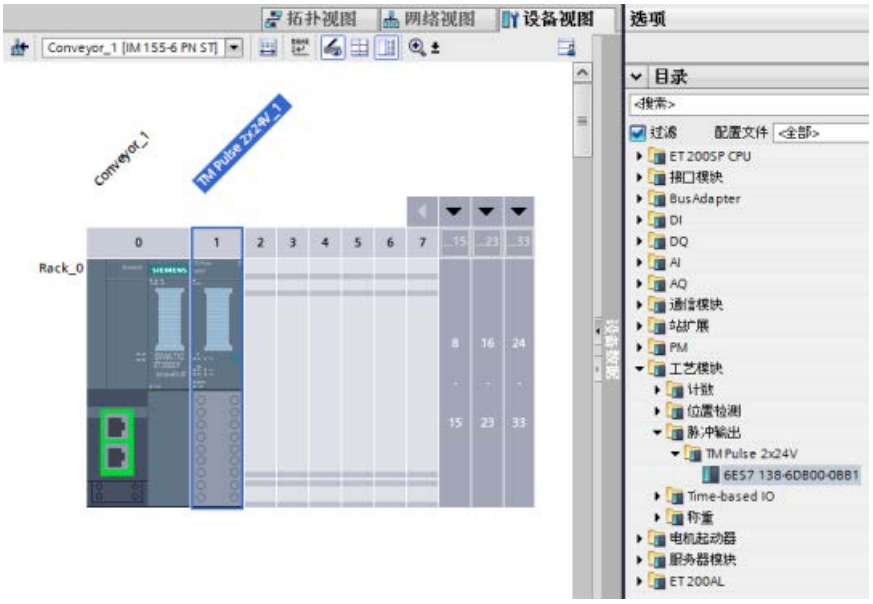
在某些要求以多种速度进行移动的情况下，闭环控制可以工作在速度控制模式下。如果使用此模式，则无需位置反馈。

按照以下步骤组态 TM 脉冲模块：

1. 组态 S7-1200 CPU。
2. 选择所需 ET 200SP 接口模块并将其置于“设备视图”(Device View) 中：



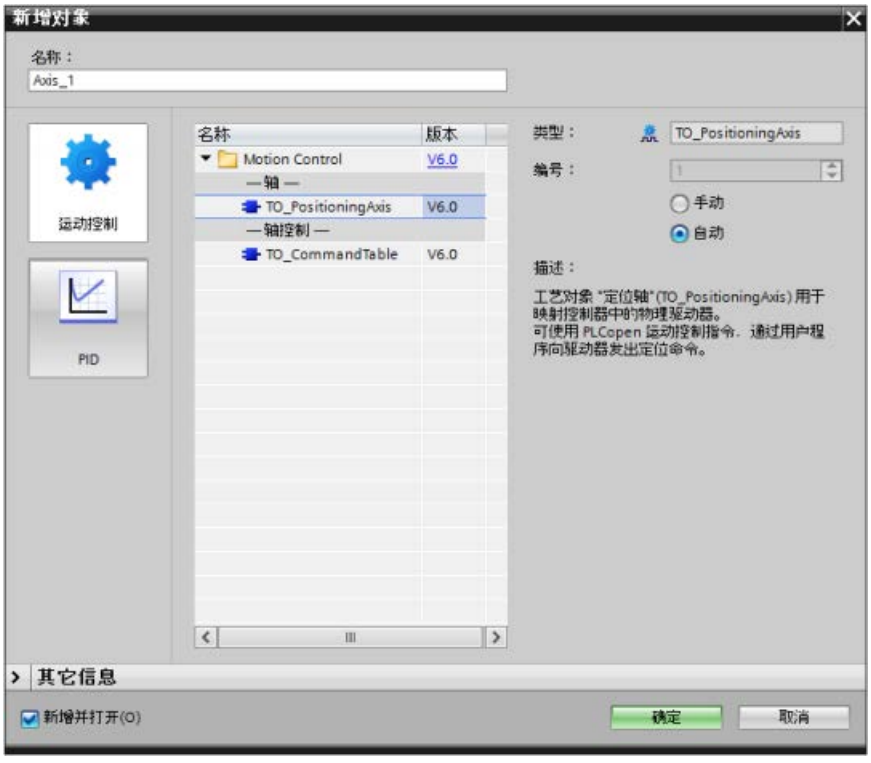
3. 添加 ET 200SP TM 脉冲模块：



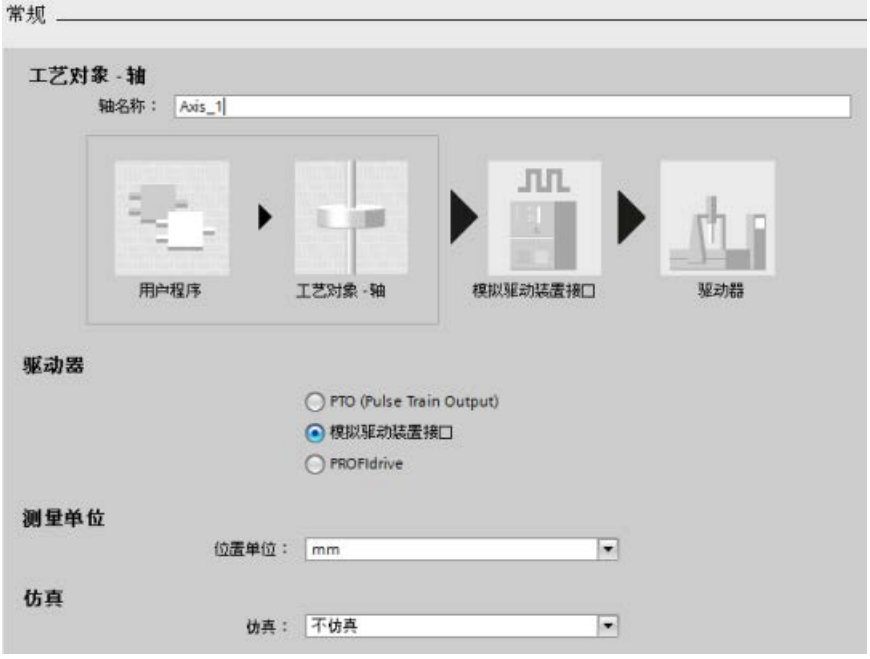
4. 在“通道”(Channel) 组态中，选择“双通道 (2A)”(2 channels (2A))。
5. 将操作模式设置为“带直流电机的 PWM”(PWM with DC Motor)。
6. 根据要求设置诊断和通道参数。

按照以下步骤组态具有位置反馈的运动轴：

1. 通过闭环运动控制系统组态接口时，应使用模拟量控制，而非 PROFIdrive 或 PTO。可以按照将模拟量输出用作伺服驱动器输入的轴的组态方式组态要控制的轴。按下图所示添加“TO PositioningAxis”：

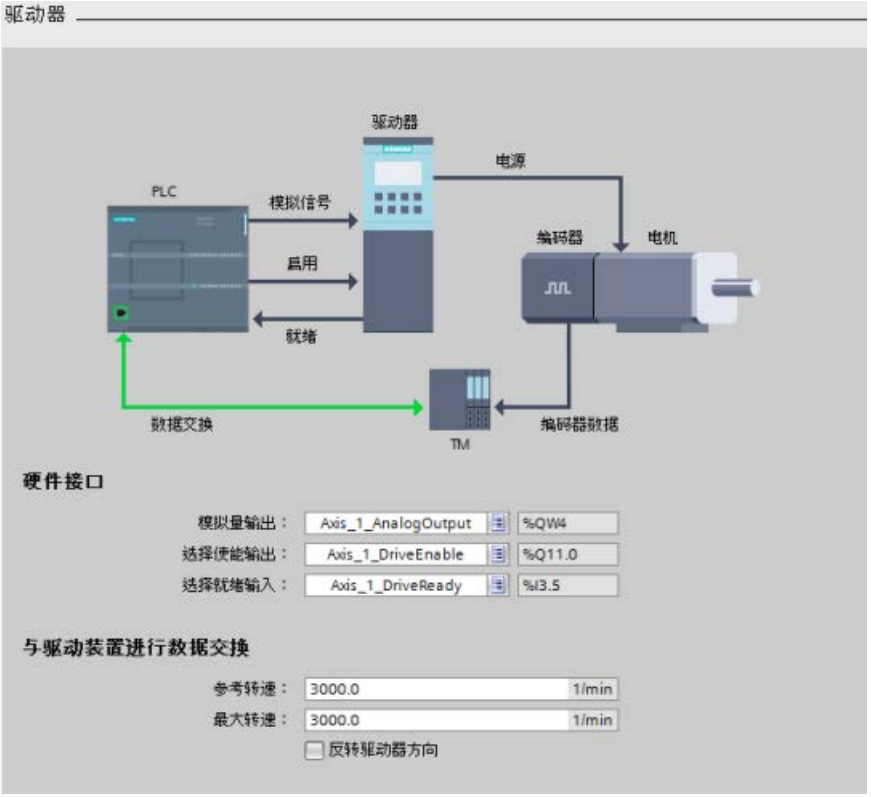


2. 在轴组态的“常规”(General) 组态对话框中，选择“模拟驱动器连接”(Analog drive connection)：



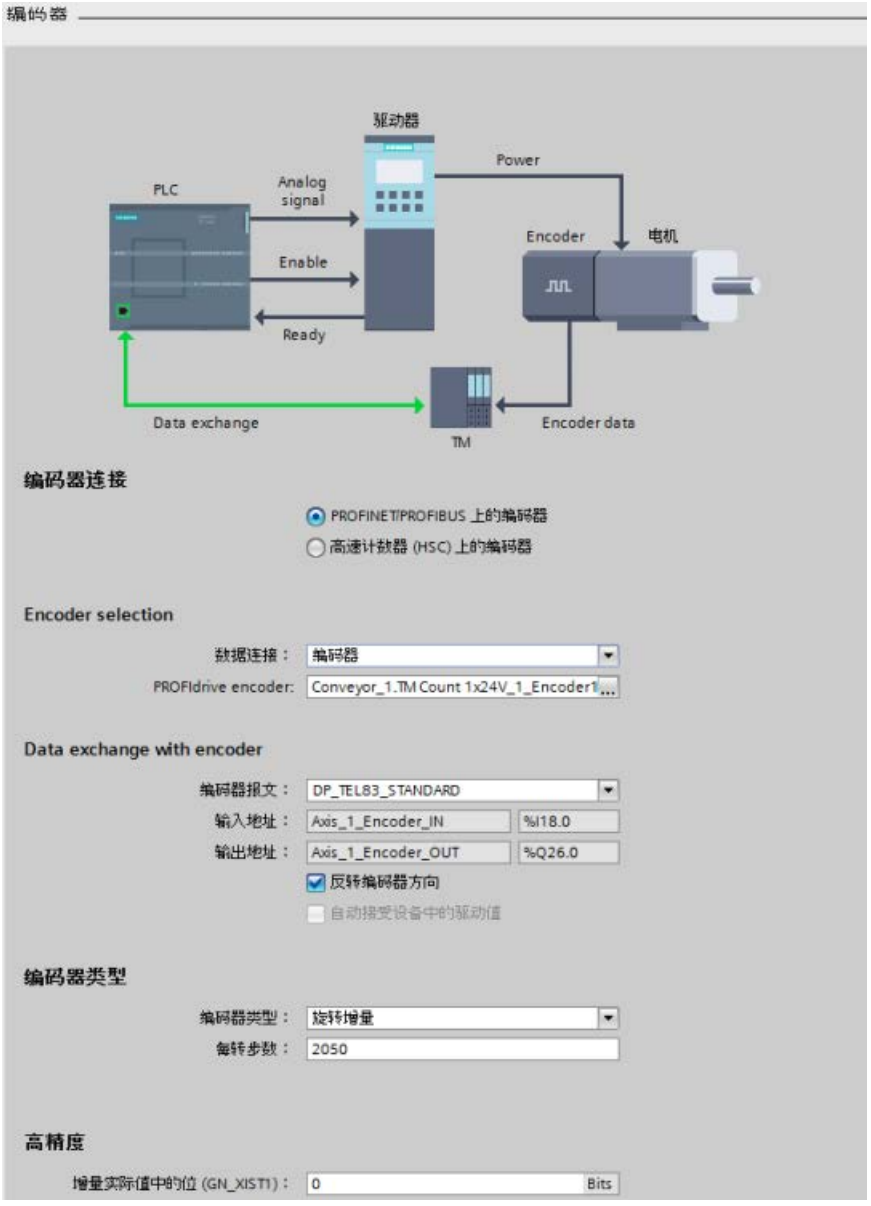
3. 现在，需要多个模拟量输出和一个驱动器使能信号，以用于驱动器组态。转至 ET 200SP 接口模块的“设备视图”(Device View) 和 TM 脉冲模块的“设备总览”(Device Overview) 条目。使用 TM 脉冲模块的起始 Q 地址和《*SIMATIC ET 200SP 工艺模块 TM 脉冲 2x24V (6ES7138-6DB00-0BB1)》手册*中的控制接口和反馈接口表，可以确定驱动器所需的以下模拟量输出和驱动器使能信号：
   – 模拟量输出
   – 选择使能输出
   – 选择就绪输入

   在轴组态的“驱动器”(Drive) 组态对话框中，选择硬件接口 I/O 和数据交换值：



4. 在轴组态的“编码器”(Encoder) 组态对话框中，选择以下任一编码器以完成组态：

- TM 计数模块
- TM PosInput 模块
- 高速计数器 (HSC)

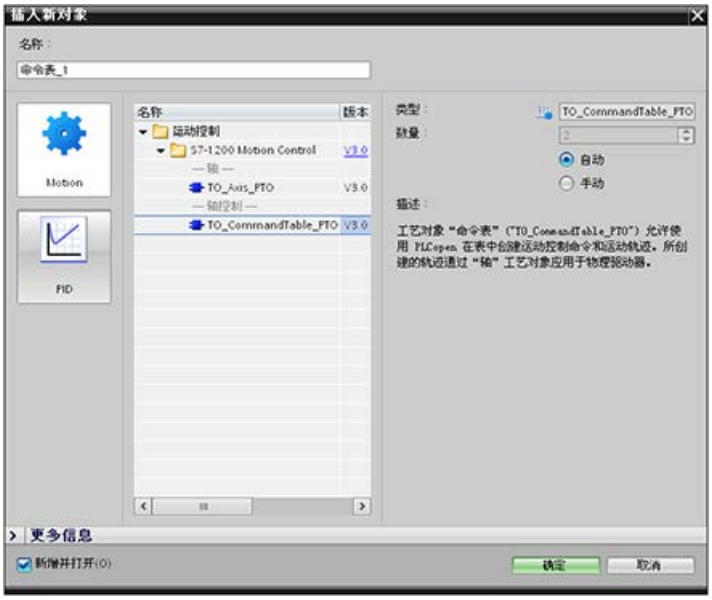


10.3.5 组态 TO_CommandTable_PTO

可以使用工艺对象组态 MC_CommandTable 指令。 以下示例将演示如何操作。

添加工艺对象

1. 在项目树中，展开节点“工艺对象”(Technology Objects)，然后选择“添加新对象”(Add new object）。
2. 选择“CommandTable”图标（必要时可以重命名），然后单击“确定”(OK) 打开 CommandTable 对象的组态编辑器。



为应用规划步

可在“命令表”(Command Table) 组态窗口中创建所需的运动序列，并根据趋势图中的图形视图来检查结果。

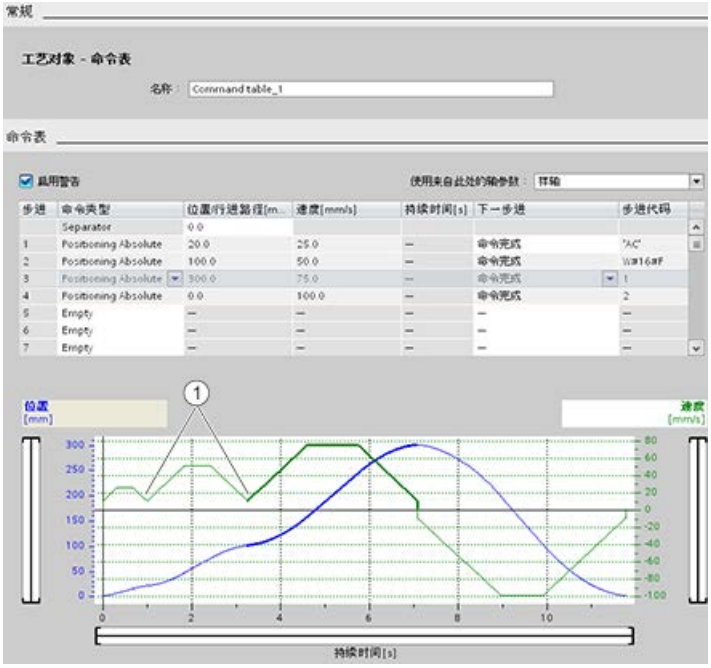
可选择要用于处理命令表的命令类型。 最多可输入 32 个步。 按顺序处理命令，轻松生成复杂运动轨迹。

表格 10-48　MC_CommandTable 命令类型

命令类型	说明
Empty	空白用作占位符，以便添加任意命令。 在处理命令表时，忽略空白条目。
Halt	暂停轴。 注： 该命令仅在“Velocity setpoint”命令之后使用。
Positioning Relative	根据距离定位轴。 该命令将按给定的距离和速度移动轴。
Positioning Absolute	根据位置定位轴。 该命令以指定的速度将轴移到给定位置。
Velocity setpoint	按给定速度移动轴。

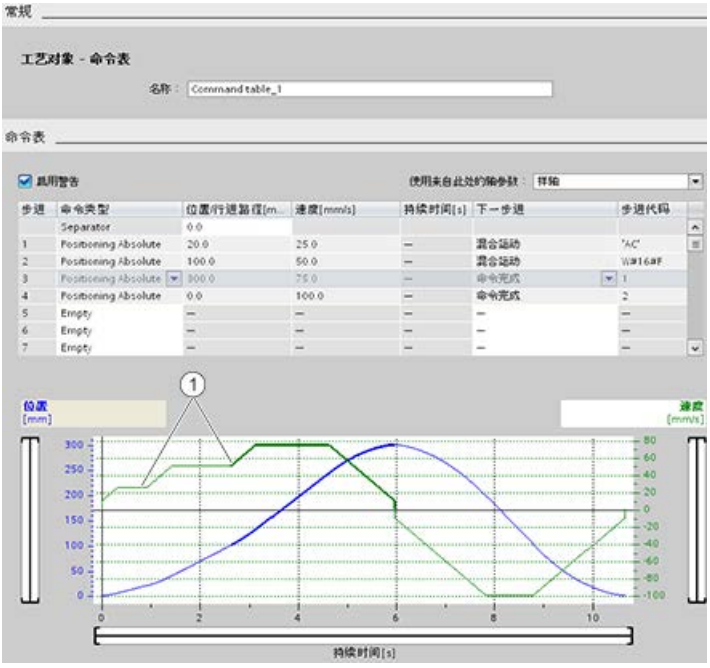
命令类型	说明
Wait	等待给定期间结束。 “Wait”不会停止已激活的行进运动。
Separator	在选定行上方添加“分隔”线。 利用分隔线，可在单个命令表中定义多个轨迹。

在下图中，“命令完成”(Command complete) 用作到下一步的切换。 该类切换允许设备减速到启动/停止速度，然后在下一步开始时重新加速。



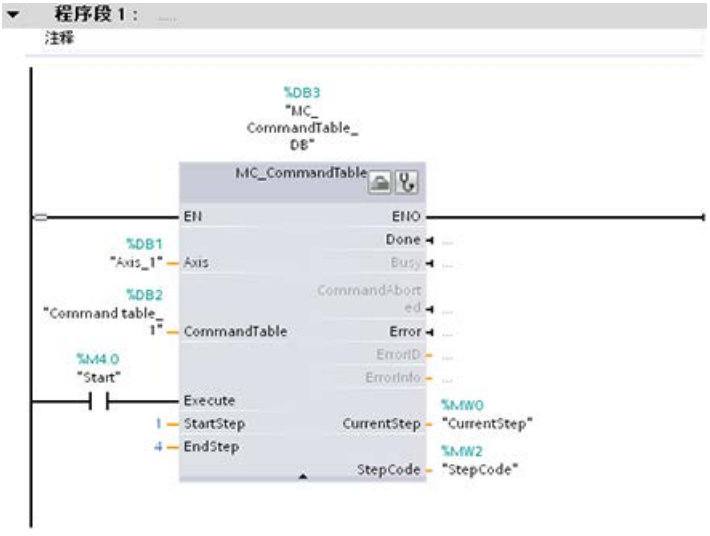
① 轴在两步之间减速到启动/停止速度。

在下图中，“混合运动”(Blending motion) 用作到下一步的切换。 该类切换允许设备保持现有速度进入下一步，从而使设备平稳地从一步切换到下一步。 采用混合方式可缩短完全执行某轨迹所需的总时间。 如果不采用混合方式，运行该示例将需要七秒。 如果采用混合方式，则执行时间将减少一秒，因此总时间为六秒。



① 轴继续移动，并加速或减速到下一步速度，这会节省时间和减少机械磨损。

CommandTable 的运行受 MC_CommandTable 指令控制，如下所示：



10.3.6 S7-1200 的运动控制的操作

10.3.6.1 用于运动控制的 CPU 输出

CPU 提供四个脉冲输出发生器。每个脉冲输出发生器提供一个脉冲输出和一个方向输出，用于通过脉冲接口对步进电机驱动器或伺服电机驱动器进行控制。脉冲输出为驱动器提供电机运动所需的脉冲。方向输出则用于控制驱动器的行进方向。

PTO 输出生成频率可变的方波输出。脉冲发生由通过 H/W 组态和/或 SFC/SFB 提供的组态和执行信息来控制。

在 CPU 处于 RUN 模式下时，根据用户的选择，将由存储在图像寄存器中的值或者脉冲发生器的输出来驱动数字量输出。在 STOP 模式下，PTO 发生器不控制输出。

板载 CPU 输出和信号板的输出可用作脉冲和方向输出。在设备组态期间，可以在“属性” (Properties) 选项卡的脉冲发生器 (PTO/PWM) 中，选择板载 CPU 输出或信号板输出。只有 PTO (Pulse Train Output) 适用于运动控制。

下表显示了默认 I/O 分配；但是，可将这四个脉冲发生器组态为任意数字量输出。

说明

用户程序中的其它指令无法使用脉冲串输出。

将 CPU 或信号板的输出组态为脉冲发生器时（与 PWM 或运动控制指令配合使用），相应的输出地址不再控制输出。如果用户程序向用作脉冲发生器的输出写入值，则 CPU 不会将该值写入到物理输出。

说明

可以释放 PTO 方向输出以在程序中的其它位置使用。

每个 PTO 需要分配两个输出：一个作为脉冲输出，一个作为方向输出。可以只使用脉冲输出而不使用方向输出。随后可以释放方向输出以用于用户程序中的其它用途。不能将输出同时用于 PTO 方向输出和用户程序。

表格 10-49 脉冲和方向输出的默认地址分配

使用运动控制的输出		
	脉冲	方向
PTO1		
内置 I/O	Q0.0	Q0.1
SB I/O	Q4.0	Q4.1
PTO2		

使用运动控制的输出		
内置 I/O	Q0.2	Q0.3
SB I/O	Q4.2[1]	Q4.3[1]
PTO3		
内置 I/O	Q0.4[2]	Q0.5[2]
SB I/O	Q4.0	Q4.1
PTO4		
内置 I/O	Q0.6[3]	Q0.7[3]
SB I/O	Q4.2	Q4.3

1 输出 Q4.2 和 Q4.3 仅适用于 SB 1222 DQ4。

2 CPU 1211C 没有输出 Q0.4、Q0.5、Q0.6 或 Q0.7。因此，这些输出不能在 CPU 1211C 中使用。

3 CPU 1212C 没有输出 Q0.6 或 Q0.7。因此，这些输出不能在 CPU 1212C 中使用。

4 该表适用于 CPU 1211C、CPU 1212C、CPU 1214C、CPU 1215C 以及 CPU 1217C PTO 功能。

驱动器接口

对于运动控制，可以选择将驱动器接口组态为“驱动器启用”或“驱动器准备就绪”。使用驱动器接口时，针对“驱动器启用”可选择数字量输出，针对“驱动器准备就绪”可选择数字量输入。

说明

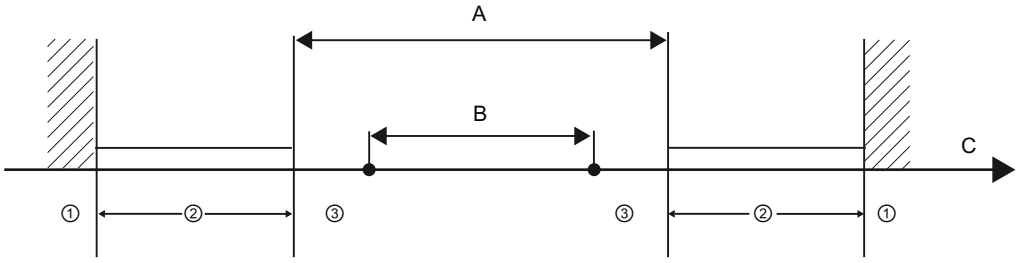
如果已选择 PTO (Pulse Train Output) 并将其分配给某个轴，固件将通过相应的脉冲和方向输出接管控制。

在实现上述控制功能接管后，将断开过程映像和 I/O 输出间的连接。虽然用户可通过用户程序或监视表格写入脉冲输出和方向输出的过程映像，但所写入的内容不会传送到 I/O 输出。因此通过用户程序或监视表格无法监视 I/O 输出。读取的信息只反映过程映像中的值，与 I/O 输出的实际状态并不完全一致。

对于 CPU 固件非永久使用的其它所有 CPU 输出，通常可以通过过程映像监控 I/O 输出的状态。

10.3.6.2 用于运动控制的硬件和软件限位开关

硬件和软件限位开关用于限制轴的“允许行程范围”和“工作范围”。



①	机械停止块	A	允许的轴行程范围
②	硬件下限和上限	B	轴的工作范围
③	软件下限和上限	C	距离

在组态中或用户程序中使用硬件和软件限位开关之前，必须事先将其激活。只有在轴回原点之后，才可以激活软件限位开关。

硬件限位开关

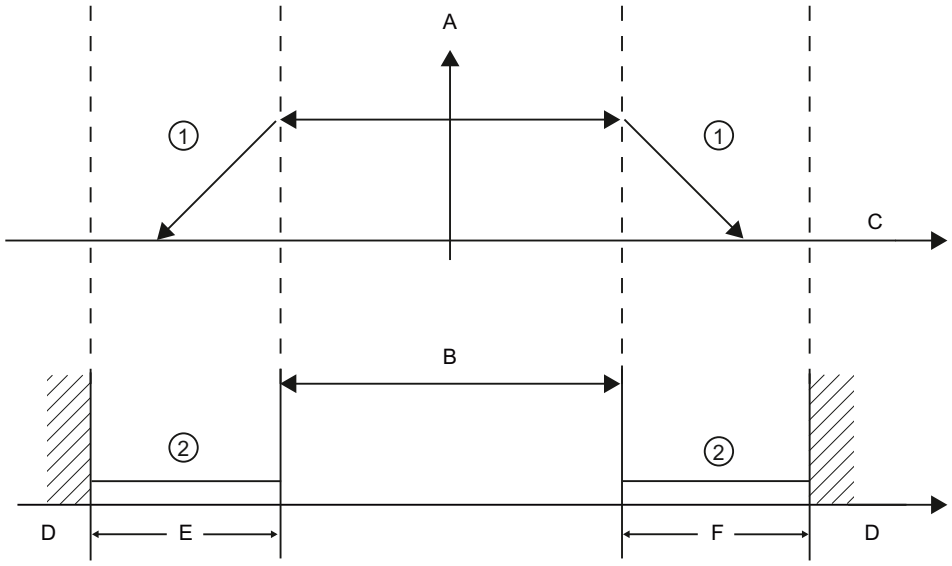
硬件限位开关确定轴的最大行程范围。硬件限位开关是物理开关元件，必须与 CPU 中具有中断功能的输入相连接。仅使用逼近后始终保持切换的硬件限位开关。只有在返回到允许的行程范围后，该切换状态才会被撤消。

表格 10-50 硬件限值的可用输入

说明	CPU	RPS、LIM- 和 LIM+ [1]
内置 I/O	CPU 1211C	I0.0 - I0.5
	CPU 1212C	I0.0 - I0.7
	CPU 1214C、CPU 1215C 和 CPU 1217C	I0.0 - I01.3
SB I/O	所有 S7-1200 CPU	I4.0 - I4.3

[1] 参考点切换输入 (RPS)，反向限位输入 (LIM-) 和正向限位输入 (LIM+)

逼近硬件限位开关时，轴将以所组态的紧急减速度制动直到停止。指定的紧急减速度必须足够大，才能确保在机械停止块前使轴停止。下图显示了轴逼近硬件限位开关后的轴行为。



① 轴将以所组态的紧急减速度制动直到停止。

② 硬件限位开关产生“已逼近”状态信号的范围。

A [速度]

B 允许的行程范围

C 距离

D 机械停止块

E 下限硬件限位开关

F 上限硬件限位开关

警告

对数字量输入通道的滤波时间进行更改的风险

如果数字量输入通道的滤波时间更改自以前的设置，则新的“0”电平输入值需要保持长达 20.0 ms 的累积时间，然后滤波器才会完全响应新输入。在此期间，无法检测到持续时间短于 20.0 ms 的“0”脉冲事件或对其进行计数。

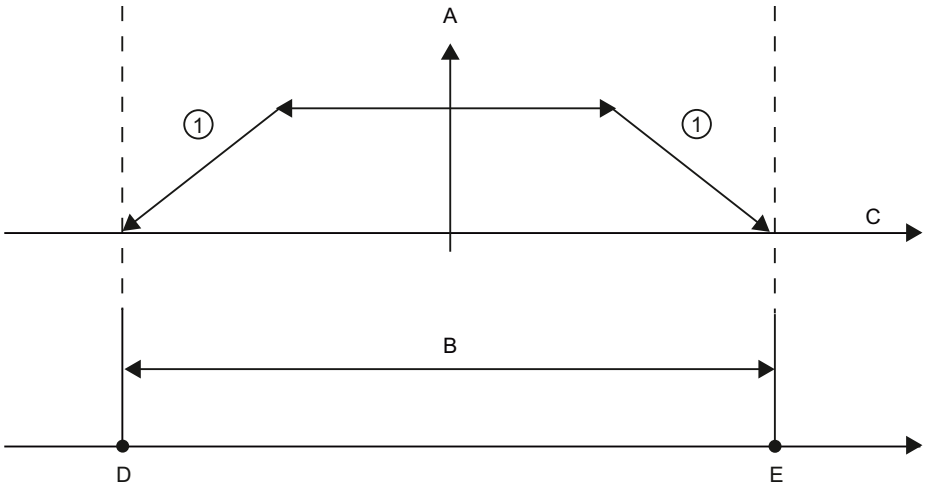
滤波时间的这种更改会引发意外的机器或过程操作，这可能会导致人员死亡、重伤和/或设备损坏。

为了确保新的滤波时间立即生效，必须关闭 CPU 电源后再开启。

软件限位开关

软件限位开关将限制轴的“工作范围”。它们必须位于行程范围的相关硬件限位开关内。由于软件限位开关的位置可以灵活设置，所以可以根据当前的运行轨迹和具体要求来限定轴的工作范围。与硬件限位开关不同，软件限位开关只通过软件来实现，而无需借助自身的开关元件。

如果软件限位开关激活，则在软件限位开关所在的位置将停止当前的运动。轴将以所组态的减速度制动。下图显示了轴到达软件限位开关前的行为。



① 轴将以所组态的减速度制动直到停止。

A [速度]

B 工作范围

C 距离

D 下限软件限位开关

E 上限软件限位开关

如果机械停止块位于软件限位开关的后面并且有发生机械损坏的风险，则需要使用附加的硬件限位开关。

地址变化时的边沿检测组态

如果在 TO PositionAxis 中将限位开关或输入回原点开关组态到某个输入地址，运动控制会自动为其组态边沿中断。随后，如果将限位开关或输入回原点开关切换到另一个地址，旧地址的边沿检测组态保持有效。

在轴组态中的“扩展参数”(Extended parameters) 的“位置限制”(Position limits) 对话框中，可添加硬件限位开关和软件限位开关。添加硬件限位开关输入时，会自动激活边沿检测。随后，如果决定更改输入地址，系统会显示一个边沿检测对话框，其中提供以下选项：

- “是”(Yes)：切换到新地址、激活新地址的边沿检测并禁用旧地址的边沿检测（默认选项）
- “否”(No)：切换到新地址、激活新地址的边沿检测并保持旧地址的边沿检测
- “取消”(Cancel)：不切换到新地址，并且保持当前的边沿检测状态

边沿检测对话框选项	切换到新地址	激活新地址的边沿检测	禁用旧地址的边沿检测
是 (Yes)（默认）	√	√	√
否 (No)	√	√	×
取消 (Cancel)	×	不适用	×

说明

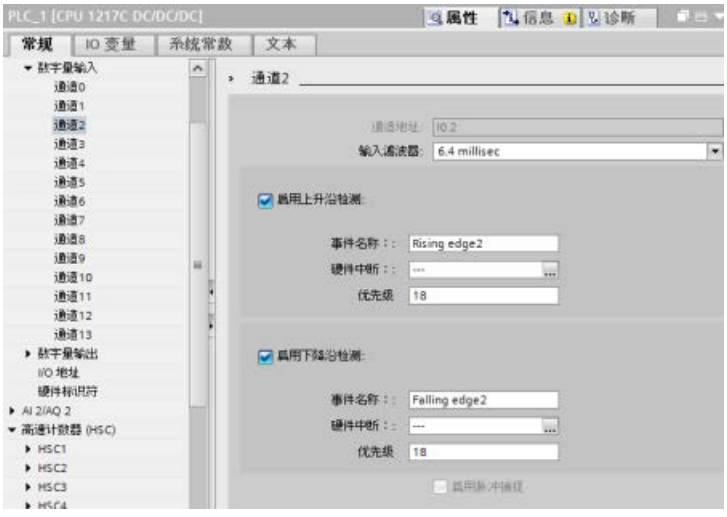
对于不需要边沿检测的伺服组态，不会显示对话框。

说明

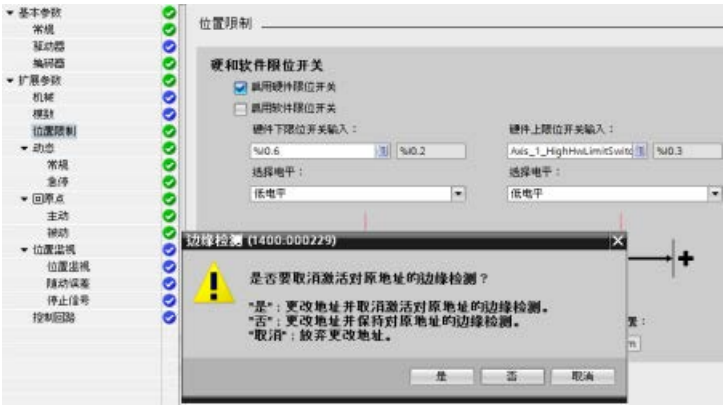
如果有一个 OB 链接到输入，运动控制不会执行上述边沿检测操作。

示例：切换到新的硬件限位开关地址，激活新地址的边沿检测，禁用旧地址的边沿检测

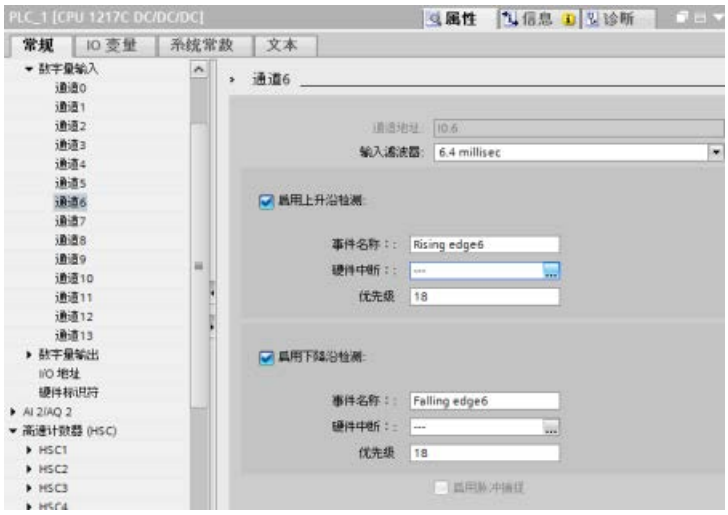
1. 当前状态：已将“输入低电平硬件限位开关”(Input low HW limit switch) 连接到 I0.2。该组态会自动启用 I0.2 的边沿检测。该组态还会显示在“CPU 属性”(CPU Properties) 的“数字量输入”(Digital Inputs) 中：

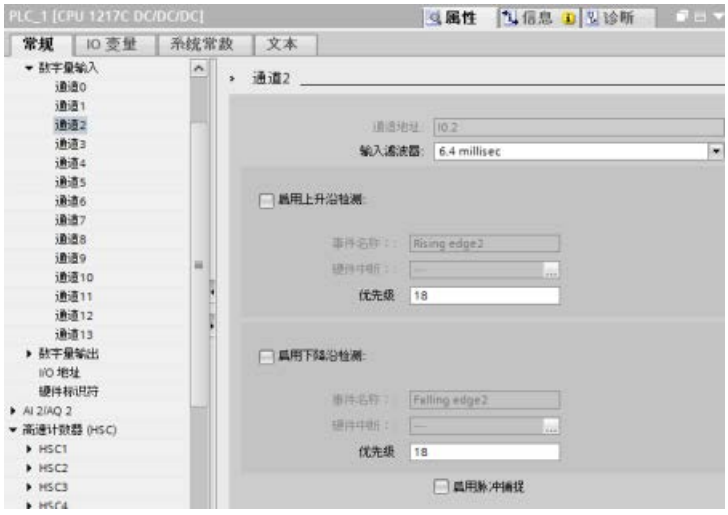


2. 已将“输入低电平硬件限位开关”(Input low HW limit switch) 切换到 I0.6 并确认。随即显示边沿检测对话框。选择：“是：切换地址并禁用旧地址的边沿检测。”(Yes: Change address and deactivate edge detection on old address.)：



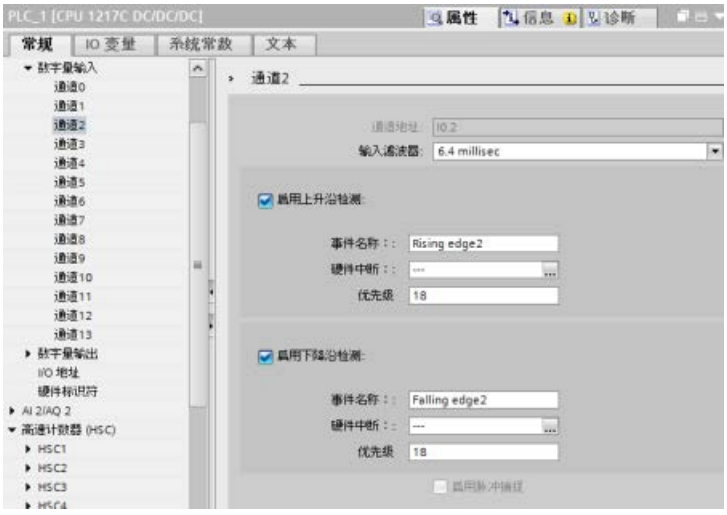
3. CPU 会接受新的 I0.6 地址并激活 I0.6 的边沿检测。“CPU 属性”(CPU Properties) 的“数字量输入”(Digital Inputs) 中的 I0.2 边沿检测自动被禁用：


PLC_1 [CPU 1217C DC/DC/DC]
属性
信息
诊断
常规
IO 变量
系统常数
文本
数字量输入
通道0
通道1
通道2
通道3
通道4
通道5
通道6
通道7
通道8
通道9
通道10
通道11
通道12
通道13
数字量输出
I/O 地址
硬件标识符
AI 2/AQ 2
高速计数器 (HSC)
HSC1
HSC2
HSC3
HSC4
通道6
通道地址: I0.6
输入滤波器: 6.4 millisec
启用上升沿检测:
事件名称 : : Rising edge6
硬件中断 : : ---
优先级 18
启用下降沿检测:
事件名称 : : Falling edge6
硬件中断 : : ---
优先级 18
启用脉冲捕捉

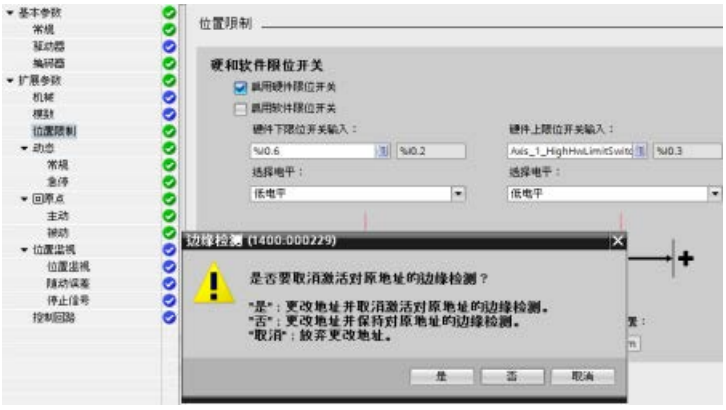

PLC_1 [CPU 1217C DC/DC/DC]
属性
信息
诊断
常规
IO 变量
系统常数
文本
数字量输入
通道0
通道1
通道2
通道3
通道4
通道5
通道6
通道7
通道8
通道9
通道10
通道11
通道12
通道13
数字量输出
I/O 地址
硬件标识符
AI 2/AQ 2
高速计数器 (HSC)
HSC1
HSC2
HSC3
HSC4
通道2
通道地址: I0.2
输入滤波器: 6.4 millisec
启用上升沿检测:
事件名称 : : Rising edge2
硬件中断 : : ---
优先级 18
启用下降沿检测:
事件名称 : : Falling edge2
硬件中断 : : ---
优先级 18
启用脉冲捕捉

示例：切换到新的硬件限位开关地址，激活新地址的边沿检测，保持旧地址的边沿检测

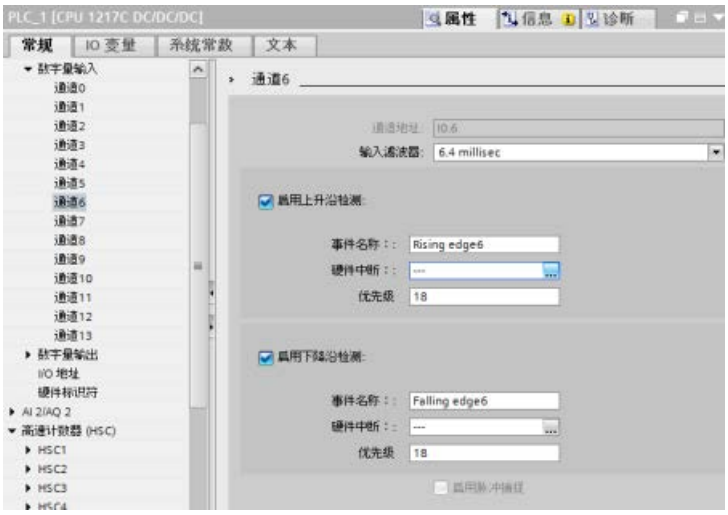
1. 当前状态：已将“输入低电平硬件限位开关”(Input low HW limit switch) 连接到 I0.2。该组态会自动启用 I0.2 的边沿检测。该组态还会显示在“CPU 属性”(CPU Properties) 的“数字量输入”(Digital Inputs) 中：



2. 已将“输入低电平硬件限位开关”(Input low HW limit switch) 切换到 I0.6 并确认。随即显示边沿检测对话框。选择：“否：切换地址并保持旧地址的边沿检测。”(No: Change address and keep edge detection on old address.)：

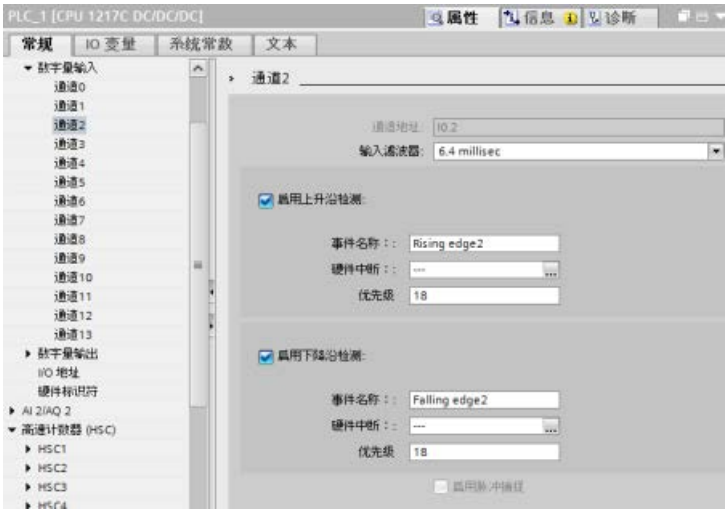


3. CPU 会接受新的 I0.6 地址并激活 I0.6 的边沿检测。“CPU 属性”(CPU Properties) 的“数字量输入”(Digital Inputs) 中的 I0.2 边沿检测保持激活状态：


PLC_1 [CPU 1217C DC/DC/DC]
属性
信息
诊断
常规
IO 变量
系统常数
文本
数字量输入
通道0
通道1
通道2
通道3
通道4
通道5
通道6
通道7
通道8
通道9
通道10
通道11
通道12
通道13
数字量输出
I/O 地址
硬件标识符
AI 2/AQ 2
高速计数器 (HSC)
HSC1
HSC2
HSC3
通道6
通道地址:
I0.6
输入滤波器:
6.4 millisec
启用上升沿检测:
事件名称 : :
Rising edge6
硬件中断 : :

优先级
18
启用下降沿检测:
事件名称 : :
Falling edge6
硬件中断 : :

优先级
18

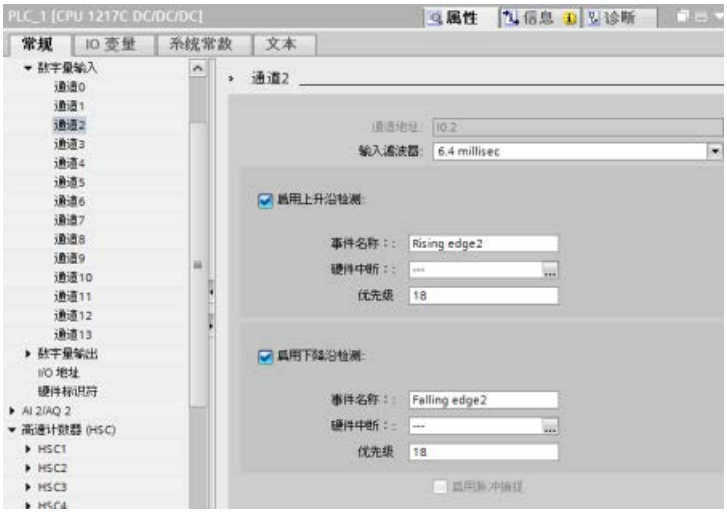

PLC_1 [CPU 1217C DC/DC/DC]
属性
信息
诊断
常规
IO 变量
系统常数
文本
数字量输入
通道0
通道1
通道2
通道3
通道4
通道5
通道6
通道7
通道8
通道9
通道10
通道11
通道12
通道13
数字量输出
I/O 地址
硬件标识符
AI 2/AQ 2
高速计数器 (HSC)
HSC1
HSC2
HSC3
通道2
通道地址:
I0.2
输入滤波器:
6.4 millisec
启用上升沿检测:
事件名称 : :
Rising edge2
硬件中断 : :

优先级
18
启用下降沿检测:
事件名称 : :
Falling edge2
硬件中断 : :

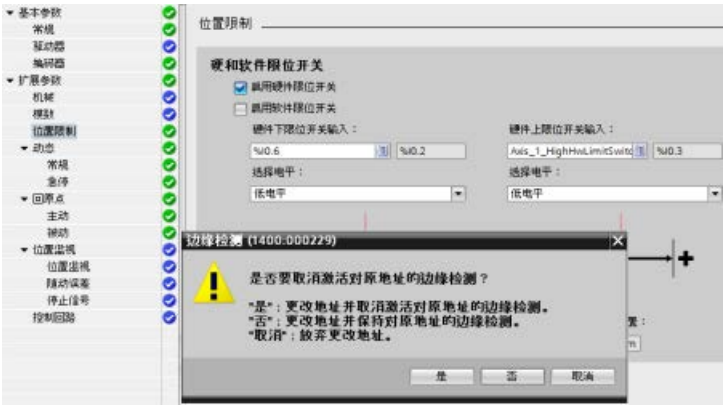
优先级
18

示例：取消对新的硬件限位开关地址的边沿检测的更改

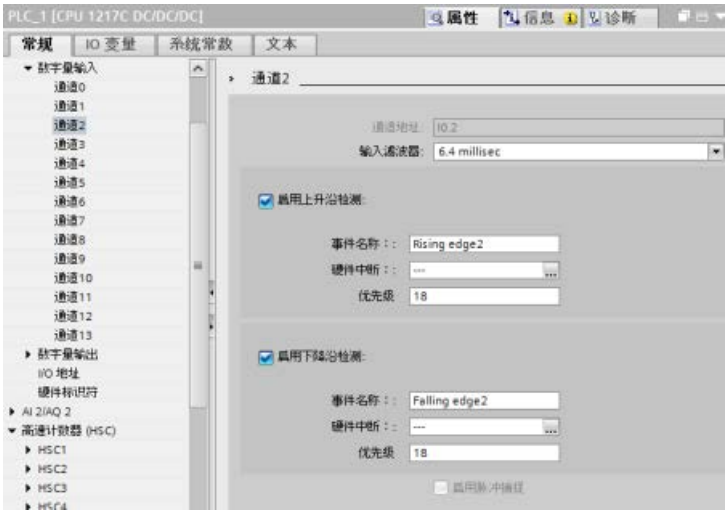
1. 当前状态：已将“输入低电平硬件限位开关”(Input low HW limit switch) 连接到 I0.2。该组态会自动启用 I0.2 的边沿检测。该组态还会显示在“CPU 属性”(CPU Properties) 的“数字量输入”(Digital Inputs) 中：



2. 已将“输入低电平硬件限位开关”(Input low HW limit switch) 切换到 I0.6 并确认。随即显示边沿检测对话框。选择：“取消”(Cancel)。



3. “输入低电平硬件限位开关”(Input low HW limit switch) 保持组态为 I0.2，I0.2 的边沿检测保持激活状态：



更多信息

用户程序可通过启用或禁用硬件和软件的限位功能来消除硬件或软件限位。可通过轴 DB 进行选择。

- 要启用或禁用硬件限位功能，请访问 DB 路径“<轴名称>/Config/**PositonLimits_HW**”中的“Active”变量 (Bool)。“Active”变量的状态可启用或禁用硬件限位的使用。
- 要启用或禁用软件限位功能，请访问 DB 路径“<轴名称>/Config/**Position Limits_SW**”中的“Active”变量 (Bool)。此“Active”变量的状态可启用或禁用软件限位。

还可以利用用户程序修改软件限位（例如，提高机器设置的灵活性或缩短机器转换时间）。用户程序可以将新值写入 DB 路径“<轴名称>/Config/**PositionLimits_SW**”的“ MinPosition”和“ MaxPosition”变量中（采用 Real 格式的工程单位）。

10.3.6.3 回原点

回原点是指轴坐标与实际的物理驱动器位置匹配。（如果驱动器当前处于位置 x，则轴将被调整到位置 x。）对于位置控制的轴，位置的输入与显示完全参考轴的坐标。

说明

轴坐标必需与实际情形相一致。如果要确保通过驱动器也能准确到达轴的绝对目标位置，上述步骤必不可缺。

MC_Home 指令可启动轴的回原点操作。

有 4 种不同的回原点功能。前两种功能允许用户设置轴的当前位置，后两种功能可相对于回原点参考传感器放置轴。

- 模式 0 - 绝对式直接参考：指令执行时，此模式将告知轴它的确切位置。该模式将内部位置变量设置为回原点指令的 Position 输入的值。此模式用于机器校准和设置。
  轴位置的设置与参考点开关无关。也不会终止当前的行进运动。MC_Home 指令的 Position 输入参数的值将被立即设置为轴的参考点。要将参考点分配给具体的机械位置，在执行回原点操作时轴必须停止在该位置。
- 模式 1 - 相对式直接参考：指令执行时，该模式将使用内部位置变量并加上回原点指令的 Position 输入的值。考虑到机器偏移时通常使用此模式。
  轴位置的设置与参考点开关无关。也不会终止当前的行进运动。以下语句适用于回到原点后轴的定位：新轴位置 = 当前轴位置 + MC_Home 指令的 Position 参数的值。
- 模式 2 - 被动参考：当轴在移动的过程中经过参考点开关时，当前位置将设置为回原点位置。此功能有助于应对正常的机器磨损和齿轮间隙，从而无需对磨损进行手动补偿。如前所述，回原点指令的 Position 输入将添加到参考点开关指示的位置，从而可轻松补偿回原点位置。
  在被动回原点期间，指令 MC_Home 不会执行任何回原点运动。用户必须通过其它运动控制指令来执行该步骤所需的行进运动。检测到参考点开关时，将根据组态使轴回到原点。被动回原点启动时，不会中止当前的行进运动。
- 模式 3 - 主动参考：此模式是最精确的使轴回原点方法。运动的初始方向和速度在工艺对象组态扩展参数 Homing 中进行组态。这取决于机器的配置。还可以确定参考点开关信号的上升沿或下降沿是否是回原点位置。几乎所有传感器都具有一个有效范围；如果“稳态开启”位置用作回原点信号，则回原点位置可能会出现错误，因为“开启”信号有效范围将覆盖距离范围。利用该信号的上升沿或下降沿，可得到更加精确的回原点位置。与其它所有模式一样，回原点指令的 Position 输入的值将被添加到硬件参考位置。
  在主动回原点模式下，MC_Home 指令执行所需的参考点逼近。检测到参考点开关时，将根据组态使轴回到原点。同时终止当前的行进运动。

模式 0 和模式 1 不需要移动轴。这两种模式通常用在设置和校准中。模式 2 和模式 3 需要轴运动并经过在“轴”工艺对象中组态为参考点开关的传感器。参考点可放在轴的工作区内或放在常规工作区外、运动范围内。

回原点参数的组态

在“回原点”(Homing) 组态窗口中，组态主动和被动回原点参数。可以使用运动控制指令中的“Mode”输入参数设置回原点方法。其中，Mode = 2 表示被动回原点，Mode = 3 表示主动回原点。

说明

采用以下措施之一可确保机器在发生反向时不会行进到机械停止块：

- 保持较低的逼近速度
- 增大组态的加速度/减速度
- 增大硬件限位开关和机械停止块间的距离

表格 10-51 使轴回原点的参数组态

参数	说明
输入参考点开关 （主动和被动回原点）	从下拉列表框中为参考点开关选择数字量输入。输入必须具有中断功能。板载 CPU 输入和所插入信号板输入都可以选作参考点开关的输入。 数字量输入的默认滤波时间为 6.4 ms。采用默认时间的数字量输入用作参考点开关的输入时，可能引起意外减速，从而导致出现误差。根据回原点速度和参考点开关的范围，可能检测不到参考点。可以在数字量输入的设备组态的“输入滤波器”(Input filter) 中设置滤波时间。 指定的滤波时间必须小于参考点开关的输入信号的持续时间。
参考点开关 （主动和被动回原点）	• 主动回原点：选择在参考点开关的下端还是上端引用轴。根据轴的起始位置和组态的回原参数，参考点的逼近顺序可能与组态窗口中图示的顺序不同。 • 被动回原点：对于被动回原点，必须由用户通过运动命令来执行回原点的行进运动。回原点发生在参考点开关的哪一端取决于以下因素： – “参考点开关”组态 – 被动回原点过程中当前的行进方向

参数	说明
到达硬件限位开关后自动反转 （仅限主动回原点）	激活该复选框，可将硬件限位开关用作指示参考点逼近的反向凸轮。必须组态硬件限位开关并激活反向功能。 如果在主动回原点期间到达硬件限位开关，轴将以组态的减速度减速（不是以紧急减速度），然后反向。然后反向检测参考点开关。 如果未激活反向功能且在主动回原点期间轴到达硬件限位开关，将因错误取消参考点逼近并按紧急减速度使轴制动。
逼近方向 （主动和被动回原点）	通过方向选择，可以决定主动回原点期间用于搜索参考点开关的“逼近方向”以及回原点的方向。回原点方向将指定执行回原点操作时轴用于逼近组态的参考点开关侧的行进方向。
逼近速度 （仅限主动回原点）	指定参考点逼近期间搜索参考点开关的速度。 限值（与所选的用户单位无关）： 启动/停止速度 ≤ 逼近速度 ≤ 最大速度
回原点速度 （仅限主动回原点）	指定轴逼近回原点参考点开关的速度。 限值（与所选的用户单位无关）： 启动/停止速度 ≤ 回原点速度 ≤ 最大速度
回原点位置偏移 （仅限主动回原点）	如果期望的参考点与参考点开关的位置有偏移，则可在该字段中指定回原点位置偏移。 如果值不等于 0，轴回到参考点开关位置后将执行以下动作： 1. 以回原点速度使轴移动回原点位置偏移指定的一段距离。 2. 到达回原点位置偏移的位置后，将该轴位置设置为绝对参考位置。并通过运动控制指令“MC_Home”的参数“Position”，指定该绝对参考位置。 限值（与所选的用户单位无关）： -1.0e12 ≤ 回原点位置偏移 ≤ 1.0e12

表格 10-52　影响回原点的因素

影响因素：			结果：
组态 逼近方向	组态 参考点开关	当前的行进方向	回原点发生在 参考点开关
正方向	“下（负）侧”	正方向	下侧
		负方向	上侧
正方向	“上（正）侧”	正方向	上侧
		负方向	下侧

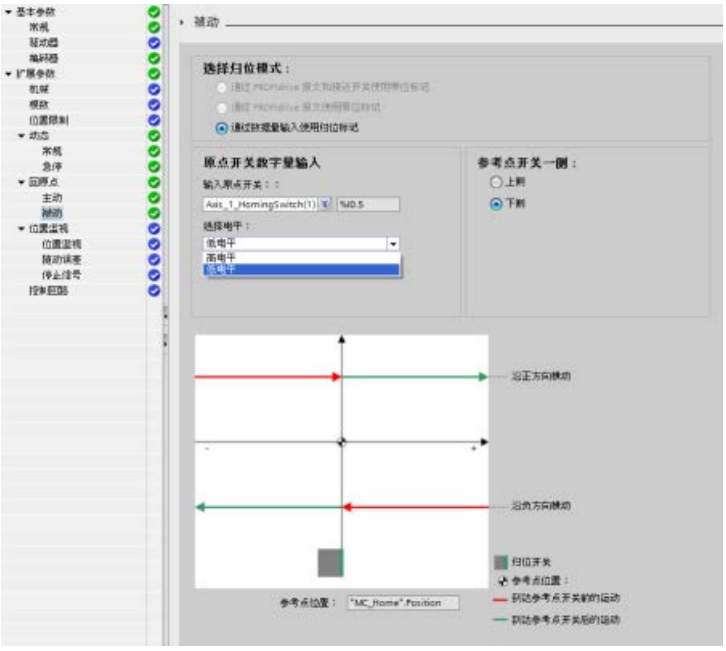
影响因素：			结果：
组态 逼近方向	组态 参考点开关	当前的行进方向	回原点发生在 参考点开关
负方向	“下（负）侧”	正方向	上侧
		负方向	下侧
负方向	“上（正）侧”	正方向	下侧
		负方向	上侧

回原点参考点开关电平

在轴组态“回原点”(Homing) 对话框中，可将“数字量输入回原点开关”(Digital input homing switch) 组态为主动或被动回原点。作为该组态的一部分，用户还可以更改闭环轴（PROFIdrive 和模拟）参考点开关的电平（高或低）。默认值为高电平。

示例： 选择被动参考点开关电平

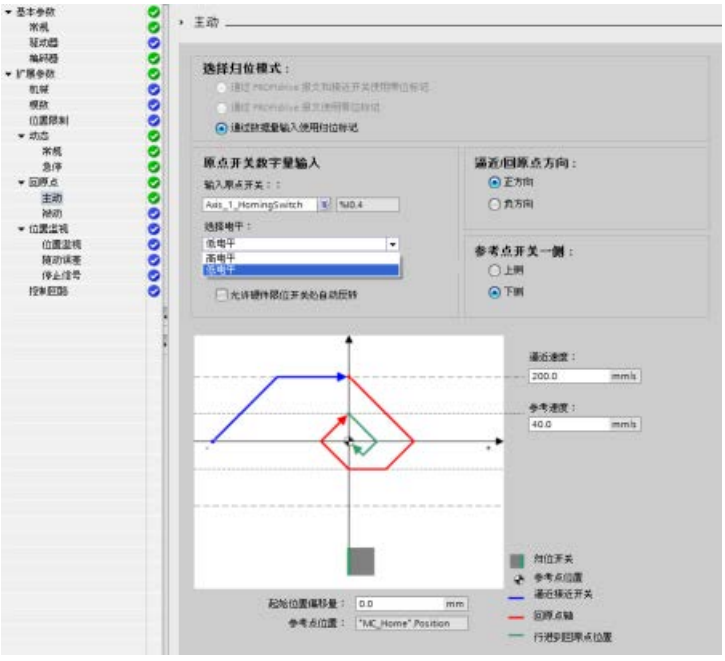
1. 已使用被动回原点组态一个含模拟/PROFIdrive 轴的 S7-1200 项目。根据具体应用，可选择“高电平”或“低电平”作为被动回原点开关电平：



2. 程序执行被动回原点。
3. 完成被动回原点后，轴即回到原点。

示例：选择主动参考点开关电平

1. 已使用主动回原点组态一个含模拟/PROFIdrive 轴的 S7-1200 项目。根据具体应用，可选择“高电平”或“低电平”作为主动回原点开关电平：



2. 程序执行主动回原点。
3. 完成主动回原点后，轴即回到原点。

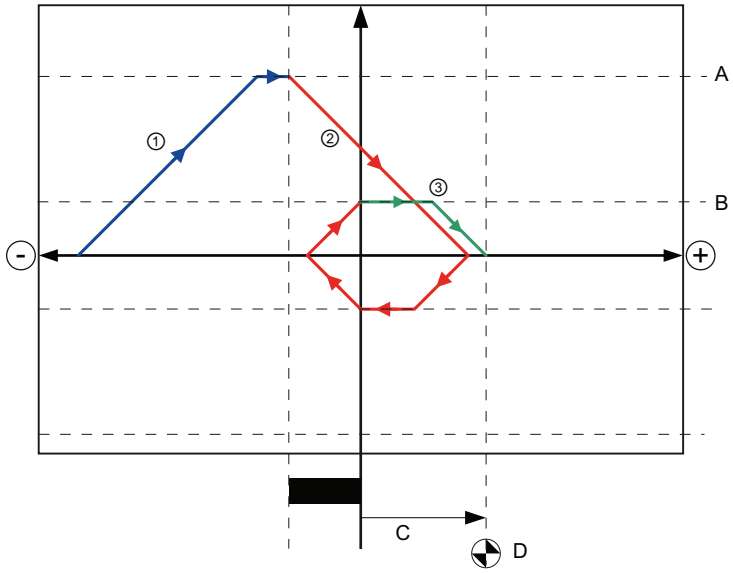
主动回原点的顺序

使用运动控制指令“MC_Home”（输入参数 Mode = 3），可以启动主动回原点。在这种情况下，可以通过输入参数“Position”来指定绝对参考点的坐标。也可以在控制面板上启动主动回原点，以便进行测试。

下图举例说明了使用以下组态参数时主动参考点逼近的特征曲线：

- “逼近方向”=“正方向逼近”
- “参考点开关”=“上（正）侧”
- “回原点位置偏移”值 > 0

表格 10-53 MC 归位的速度特性曲线

操作	注意	
	A	逼近速度
	B	回原点速度
	C	归位位置坐标
	D	回原点位置偏移

①	搜索阶段（蓝色曲线段）：主动回原点开始时，轴加速到组态的“逼近速度”并以该速度搜索参考点开关。
②	参考点逼近（红色曲线段）：检测到参考点开关时，本示例中的轴将制动并反向，以“回原点速度”在组态的参考点开关侧回原点。
③	行进到参考点位置（绿色曲线段）：轴回原点到参考点开关位置后，轴将以“回原点速度”行进到“参考点坐标”。到达“参考点坐标”时，轴将立即停止在指令 MC_Home 的 Position 输入参数中指定的位置值处。

说明

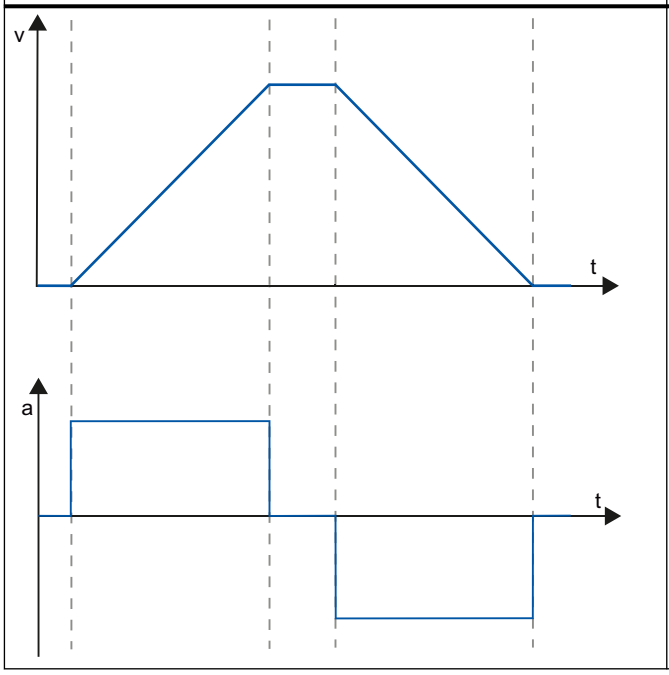
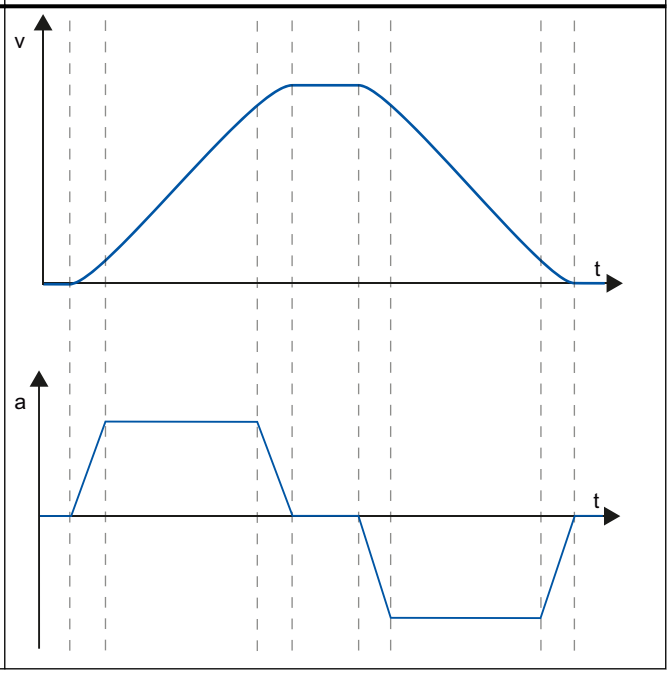
如果回原点搜索没有按照预期的那样运行，请检查分配给硬件限位或参考点的输入。可能已经在设备配置中禁用了这些输入的沿中断。

请检查相关轴工艺对象的组态数据，以查看为“HW Low Limit Switch Input”、“HW High Limit Switch Input”和“Input reference point switch”分配了哪些输入（如果有）。然后打开 CPU 的设备配置，检查所分配的每个输入。确认是否选择了“启用上升沿检测”(Enable rising edge detection) 和“启用下降沿检测”(Enable falling edge detection)。如果未选择这些属性，请删除轴组态中指定的输入，然后再次选择这些属性。

10.3.6.4 冲击限制

利用冲击限制可在加速和减速斜坡期间减小机械装置上的应力。 步进限制器处于激活状态时，加速度和减速度的值不会突然发生变化；该值会在转换阶段进行调整。 下图显示了不使用冲击限制和使用冲击限制时的速度和加速度曲线。

表格 10-54 加加速度限制

不使用步进限制器时的曲线	使用步进限制器时的曲线
	

冲击限制使轴运动的速度曲线变得“平滑”。 例如，这可以确保传送带实现软启动和软制动。

10.3.7 运动控制指令

10.3.7.1 MC 指令概述

运动控制指令使用相关工艺数据块和 CPU 的专用 PTO（脉冲串输出）来控制轴上的运动。

- MC_Power (页 734) 可启用和禁用运动控制轴。
- MC_Reset (页 737) 可复位所有运动控制错误。所有可确认的运动控制错误都会被确认。
- MC_Home (页 738) 可建立轴控制程序与轴机械定位系统之间的关系。
- MC_Halt (页 741) 可取消所有运动过程并使轴停止运动。停止位置未定义。

- MC_MoveAbsolute (页 743) 可启动到某个绝对位置的运动。达到目标位置后该作业结束。
- MC_MoveRelative (页 746) 可启动相对于起始位置的定位运动。
- MC_MoveVelocity (页 748) 可使轴以指定的速度行进。
- MC_MoveJog (页 751) 可执行用于测试和启动目的的点动模式。
- MC_CommandTable (页 753) 用于将轴命令作为一个运动序列运行。
- MC_ChangeDynamic (页 756) 用于更改轴的动态设置。
- MC_WriteParam (页 758) 用于写入选定数量的参数来通过用户程序更改轴功能。
- MC_ReadParam (页 760) 用于读取选定数量的参数，以指示在轴输入中定义的轴的当前位置、速度等。

CPU 固件级别

如果具有固件版本为 V4.1 或更高版本的 S7-1200 CPU，则为每个运动指令选择 V5.0 的版本。

如果具有固件版本为 V4.0 或更早版本的 S7-1200 CPU，则为每个运动指令选择适用的 V4.0、V3.0、V2.0 或 V1.0 版本。

说明

运动控制 V1.0 到 V3.0 中的指令主动控制指令的 ENO 输出。当块内发生错误时，ENO 输出将切换为关闭状态。通过块上的 ERROR、ErrorID 和 ErrorInfo 输出指示错误。利用 ENO 输出，可以评估指令的状态并以连续方式执行随后的指令。

若是运动控制 V4.0 和 V5.0 中的指令，ENO 输出在指令执行期间一直保持为“真”，即使出现错误状态也如此。对于使用 V3.0 或更早版本的运动控制的程序，若是依赖 ENO 状态，则会导致程序出错。为更正这种情况，使用运动控制 V4.0 或更高版本时，应通过 DONE 和 ERROR 输出（而非 ENO 输出）来评估指令状态。

说明

CPU 以 10 ms 为“时间片”或时间段计算运动任务。执行一个时间片时，下一时间片会在队列中等待执行。如果中断某个轴上的运动任务（通过执行该轴的其它新运动任务实现），新运动任务要等待长达 20 ms（当前时间片的剩余时间加上排队的时间片）后才可执行。

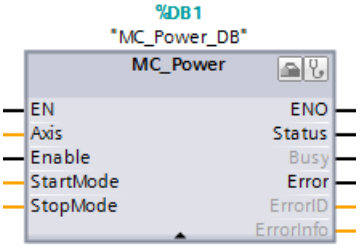
10.3.7.2 MC_Power（释放/阻止轴）

说明

如果由于错误而将轴关闭，则在消除并确认错误后会自动再次将其启用。这要求输入参数 Enable 的值在该过程中保持为 TRUE。

表格 10-55 MC_Power 指令

LAD/FBD	SCL	说明

	`"MC_Power_DB"(`
	`Axis:=_multi_fb_in_,`
	`Enable:=_bool_in_,`
	`StartMode:=_int_in_,`
	`StopMode:=_int_in_,`
	`Status=>_bool_out_,`
	`Busy=>_bool_out_,`
	`Error=>_bool_out_,`
	`ErrorID=>_word_out_,`
	`ErrorInfo=>_word_out_);`
		MC_Power 运动控制指令可启用或禁用轴。在启用或禁用轴之前，应确保以下条件：
		● 已正确组态工艺对象。
		● 没有未决的启用-禁止错误。
		运动控制任务无法中止 MC_Power 的执行。禁用轴（输入参数 Enable = FALSE）将中止相关工艺对象的所有运动控制任务。

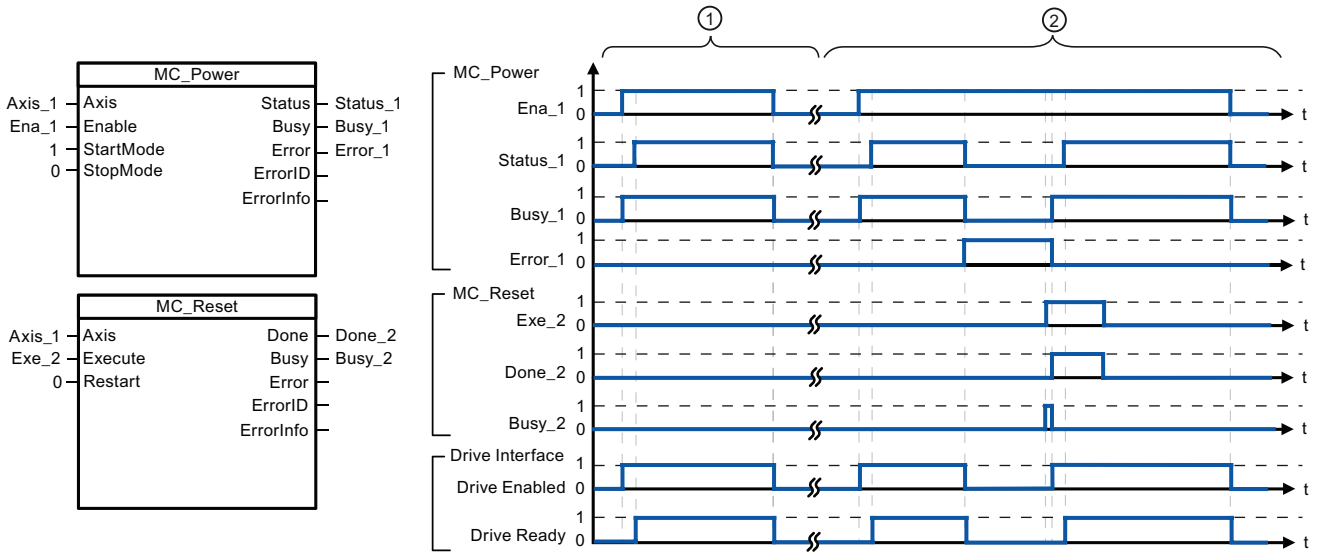
[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“MC_Power_DB”是背景 DB 的名称。

表格 10-56 MC_Power 指令的参数

参数和类型		数据类型	说明
Axis	IN_OUT	TO_Axis	轴工艺对象
Enable	IN	Bool	● FALSE（默认）：所有激活的任务都将按照参数化的“StopMode”而中止，并且轴也会停止。 ● TRUE：运动控制尝试启用轴。
StartMode	IN	Int	● 0：速度控制 注：只有在信号检测（False 变为 True）期间才会评估 StartMode 参数。 ● 1：位置控制（默认）

参数和类型		数据类型	说明
StopMode	IN	Int	• 0：急停：如果禁用轴的请求未决，则轴将以组态的紧急减速度制动。轴在达到停止后被禁用。 • 1：立即停止：如果禁用轴的请求未决，该轴将在不减速的情况下被禁用。脉冲输出立即停止。 • 2：通过冲击控制进行急停：如果禁用轴的请求未决，则轴将以组态的急停减速度制动。如果激活了冲击控制，则不考虑组态的冲击。轴在达到停止后被禁用。
Status	OUT	Bool	轴使能的状态： • FALSE：轴已禁用： – 轴不会执行运动控制任务并且不接受任何新任务（例外：MC_Reset 任务）。 – 轴未回原点。 – 禁用时，直到轴达到停止状态，状态才会更改为 FALSE。 • TRUE：轴已启用： – 轴已准备好执行运动控制任务。 – 轴启用时，直到信号“驱动器就绪”(Drive ready) 进入未决，状态才会更改为 TRUE。如果在轴组态中未组态“驱动器就绪”(Drive ready) 驱动器接口，状态会立即更改为 TRUE。
Busy	OUT	Bool	FALSE：MC_Power 无效。 TRUE：MC_Power 已生效。
Error	OUT	Bool	FALSE：无错误 TRUE：运动控制指令“MC_Power”或相关工艺发生错误。出错原因可在“ErrorID”和“ErrorInfo”参数中找到。
ErrorID	OUT	Word	参数“Error""的错误 ID
ErrorInfo	OUT	Word	参数“ErrorID”的错误信息 ID



① 启用轴，然后再次禁用轴。驱动器向 CPU 返回 “驱动器就绪”(Drive ready) 信号后，可通过“Status_1”读出成功启用信息。

② 启用轴后，出现了导致轴被禁用的错误。该错误被消除并通过“MC_Reset”进行确认。然后再次启用该轴。

要启用组态了驱动器接口的轴，请按以下步骤操作：

1. 检查上文所述的要求。
2. 使用所需值初始化输入参数“StopMode”。将输入参数“Enable”设置为 TRUE。
   “驱动器已启用”(Drive enabled) 的使能输出更改为 TRUE 以启用驱动器的电源。CPU 等待驱动器的 “驱动器就绪”(Drive ready) 信号。
   当 “驱动器就绪”(Drive ready) 信号出现在 CPU 的已组态就绪输入中时，轴将变为启用状态。输出参数“Status”和工艺对象变量 <轴名称>.StatusBits.Enable 指示值 TRUE。

要启用未组态驱动器接口的轴，请按以下步骤操作：

1. 检查上文所述的要求。
2. 使用所需值初始化输入参数“StopMode”。将输入参数“Enable”设置为 TRUE。轴已启用。输出参数“Status”和工艺对象变量 <轴名称>.StatusBits.Enable 指示值 TRUE。

要禁用轴，请按以下步骤操作：

1. 将轴切换到停止状态。
   可在工艺对象变量 <轴名称>.StatusBits.StandStill 中确定轴何时处于停止状态。
2. 达到停止状态后将输入参数“Enable”设置为 FALSE。
3. 如果输出参数“Busy”和“Status”以及工艺对象变量 <轴名称>.StatusBits.Enable 指示值 FALSE，则禁用轴的操作已完成。

10.3.7.3 MC_Reset（确认错误）

表格 10-57 MC_Reset 指令

LAD/FBD	SCL	说明
	"MC_Reset_DB"(Axis:=_multi_fb_in_, Execute:=_bool_in_, Restart:=_bool_in_, Done=>_bool_out_, Busy=>_bool_out_, Error=>_bool_out_, ErrorID=>_word_out_, ErrorInfo=>_word_out_);	使用 MC_Reset 指令可确认“导致轴停止的运行错误”和“组态错误”。需要确认的错误可在“解决方法”下的“ErrorIDs 和 ErrorInfos 的列表”中找到。 使用 MC_Reset 指令前，必须已将需要确认的未决组态错误的原因消除（例如，通过将“轴”工艺对象中的无效加速度值更改为有效值）。 自 V3.0 及更高版本起，在 RUN 操作模式下，Restart 命令可将轴组态下载至工作存储器。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“MC_Reset_DB”是背景 DB 的名称。

MC_Reset 任务无法被任何其它运动控制任务中止。新的 MC_Reset 任务不会中止任何其它已激活的运动控制任务。

表格 10-58 MC_Reset 指令的参数

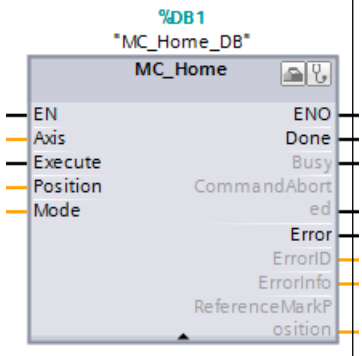
参数和类型		数据类型	说明
Axis	IN	TO_Axis_1	轴工艺对象
Execute	IN	Bool	出现上升沿时开始任务
Restart	IN	Bool	TRUE = 从装载存储器将轴组态下载至工作存储器。只有轴处于禁用状态时才能执行该命令。
			FALSE = 确认未决错误
Done	OUT	Bool	TRUE = 错误已确认。
Busy	OUT	Bool	TRUE = 正在执行任务。
Error	OUT	Bool	TRUE = 任务执行期间出错。出错原因可在“ErrorID”和“ErrorInfo”参数中找到。
ErrorID	OUTP	Word	参数“Error""的错误 ID
ErrorInfo	OUT	Word	参数“ErrorID”的错误信息 ID

要使用 MC_Reset 确认错误，请按以下步骤操作：

1. 检查上文所述的要求。
2. 在 Execute 输入参数出现上升沿时开始确认错误。
3. 当 Done 等于 TRUE 并且工艺对象变量 <轴名称>.StatusBits.Error 等于 FALSE 时，错误已被确认。
4. 自运动控制库 V7.0 及更高版本起，MC_Reset 可以在用户程序启用轴之前确认排队的错误。如果与参与者和编码器的通信正常，MC_Reset 将确认排队的错误。

10.3.7.4 MC_Home（使轴归位）

表格 10-59 MC_Home 指令

LAD/FBD	SCL	说明
	"MC_Home_DB"(Axis:=_multi_fb_in_, Execute:=_bool_in_, Position:=_real_in_, Mode:=_int_in_, Done=>_bool_out_, Busy=>_bool_out_, CommandAborted=>_bool_out_, Error=>_bool_out_, ErrorID=>_word_out_, ErrorInfo=>_word_out_, ReferenceMarkPosition=>_real_out_);	使用 MC_Home 指令可将轴坐标与实际物理驱动器位置匹配。轴的绝对定位需要回原点： 为了使用 MC_Home 指令，必须先启用轴。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“MC_Home_DB”是背景 DB 的名称。

可使用以下类型的回原点：

- 绝对式直接回原点 (Mode = 0)：当前轴位置被设置为参数“Position”的值。
- 相对式直接回原点 (Mode = 1)：当前轴位置的偏移量为参数“Position”的值。
- 被动回原点 (Mode = 2)：在被动回原点期间，指令 MC_Home 不会执行任何回原点运动。用户必须通过其它运动控制指令来执行该步骤所需的行进运动。检测到参考点开关时，轴将回到原点。
- 主动回原点 (Mode = 3)：自动执行回原点步骤。
- 绝对编码器调节（相对）(Mode = 6)：将当前位置位移参数“MC_Home.Position”的值。
- 绝对编码器调节（绝对）(Mode = 7)：将当前位置设置为参数“MC_Home.Position”的值。

表格 10-60 MC_Home 指令的参数

参数和类型		数据类型	说明
Axis	IN_OUT	TO_Axis	轴工艺对象
Execute	IN	Bool	出现上升沿时开始任务
Position	IN	Real	• Mode = 0、2 和 3（完成回原点操作后轴的绝对位置） • Mode = 1（当前轴位置的校正值） • Mode = 6（当前位置位移参数“MC_Home.Position”的值。） • Mode = 7（当前位置设置为参数“MC_Home.Position”的值。） 限值：$-1.0e^{12} \leq$ Position $\leq 1.0e^{12}$
Mode	IN	Int	归位模式： • 0：绝对式直接回原点 新的轴位置为参数“Position”的位置值。 • 1：相对式直接回原点 新的轴位置为当前轴位置 + 参数“Position”的位置值。 • 2：被动回原点 根据轴组态回原点。回原点后，参数“Position”的值被设置为新的轴位置。 • 3：主动回原点 按照轴组态进行参考点逼近。回原点后，参数“Position”的值被设置为新的轴位置。 • 6：将当前位置位移参数“MC_Home.Position”的值。计算出的绝对值偏移值始终存储在 CPU 内。(<Axis name>.StatusSensor.AbsEncoderOffset) • 7：将当前位置设置为参数“MC_Home.Position”的值。计算出的绝对值偏移值始终存储在 CPU 内。(<Axis name>.StatusSensor.AbsEncoderOffset)
Done	OUT	Bool	TRUE = 任务完成
Busy	OUT	Bool	TRUE = 正在执行任务。
CommandAborted	OUT	Bool	TRUE = 任务在执行过程中被另一任务中止。
Error	OUT	Bool	TRUE = 任务执行期间出错。出错原因可在“ErrorID”和“ErrorInfo”参数中找到。
ErrorID	OUT	Word	参数“Error"”的错误 ID
ErrorInfo	OUT	Word	参数“ErrorID”的错误信息 ID
ReferenceMarkPosition	OUT	Real	之前坐标系中参考标记处的轴位置。

“ReferenceMarkPosition”输出参数：回原点功能会在执行前保存当前位置，并通过 ReferenceMarkPosition 输出参数提供。该输出显示了不同回原点模式下的以下值：

- 主动/被动回原点：显示主动/被动回原点期间及之后，在之前坐标系中零位标记/参考标记处的轴位置。
- 绝对值编码器调整：显示绝对值编码器调整期间及调整后的上一个轴位置。
- 直接回原点：显示直接回原点期间及回原后的上一个轴位置。

回原点期间，运动控制会将轴位置设置为“MC_Home.Position”输入的新值。“MC_Home.ReferenceMarkPosition”的值在“MC_Home.Done”= TRUE 时有效。

说明

在下列情况下，轴回原点会失败：

- 通过 MC_Power 指令禁用轴
- 在自动控制和手动控制之间切换
- 主动回原点开始时（成功完成回原点操作后，可再次进行轴回原点操作。）
- 对 CPU 循环上电后
- CPU 重新启动后（RUN-to-STOP 或 STOP-to-RUN）

要使轴回原点，请按以下步骤操作：

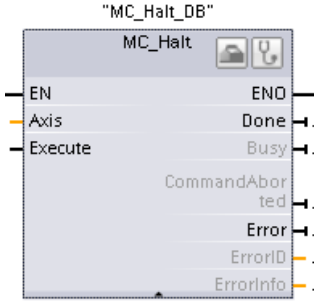
1. 检查上文所述的要求。
2. 使用相应的值初始化必要的输入参数，然后在输入参数“Execute”出现上升沿时开始回原点操作。
3. 如果输出参数“Done”和工艺对象变量 <轴名称>.StatusBits.HomingDone 指示值 TRUE，则回原点操作完成。

表格 10-61　超驰响应

模式	说明	
0 或 1	MC_Home 任务无法被任何其它运动控制任务中止。新的 MC_Home 任务不会中止任何已激活的运动控制任务。位置相关的运动任务在回原点后将根据新的原点位置（Position 输入参数中的值）恢复。	
2	MC_Home 任务可被下列运动控制任务中止： MC_Home 任务 Mode = 2、3：新 MC_Home 任务可中止以下已激活的运动控制任务。 MC_Home 任务 Mode = 2：位置相关的运动任务在回原点后将根据新的原点位置（Position 输入参数中的值）恢复。	
3	MC_Home 任务可被下列运动控制任务中止： • MC_Home Mode = 3 • MC_Halt • MC_MoveAbsolute • MC_MoveRelative • MC_MoveVelocity • MC_MoveJog	新 MC_Home 任务可中止下列激活的运动控制任务： • MC_Home 模式 = 2、3 • MC_Halt • MC_MoveAbsolute • MC_MoveRelative • MC_MoveVelocity • MC_MoveJog

10.3.7.5　MC_Halt（暂停轴）

表格 10-62　MC_Halt 指令

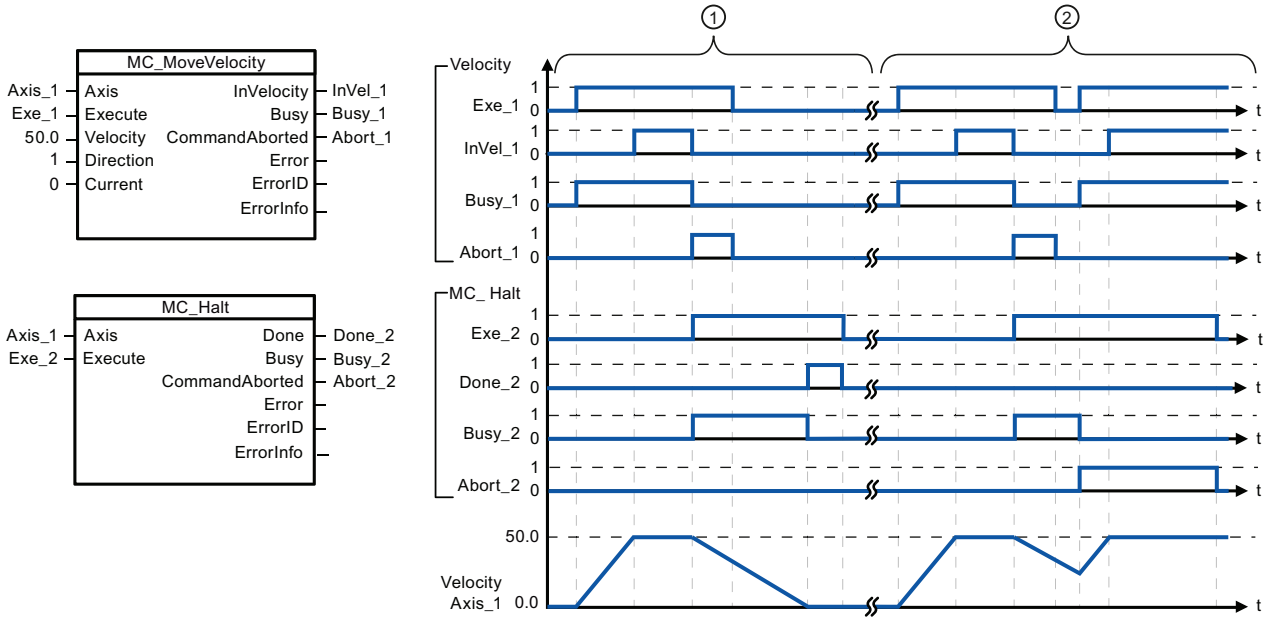
LAD/FBD	SCL	说明
	"MC_Halt_DB"(Axis:=_multi_fb_in_, Execute:=_bool_in_, Done=>_bool_out_, Busy=>_bool_out_, CommandAborted=>_bool_out_, Error=>_bool_out_, ErrorID=>_word_out_, ErrorInfo=>_word_out_);	使用 MC_Halt 指令可停止所有运动并将轴切换到停止状态。停止位置未定义。 为了使用 MC_Halt 指令，必须先启用轴。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“MC_Halt_DB”是背景 DB 的名称。

表格 10-63 MC_Halt 指令的参数

参数和类型		数据类型	说明
Axis	IN	TO_Axis_1	轴工艺对象
Execute	IN	Bool	出现上升沿时开始任务
Done	OUT	Bool	TRUE = 速度达到零
Busy	OUT	Bool	TRUE = 正在执行任务。
CommandAborted	OUT	Bool	TRUE = 任务在执行期间被另一任务中止。
Error	OUT	Bool	TRUE = 任务执行期间出错。出错原因可在“ErrorID”和“ErrorInfo”参数中找到。
ErrorID	OUT	Word	参数“Error”的错误 ID
ErrorInfo	OUT	Word	参数“ErrorID”的错误信息 ID



下面的值已在“动态 > 常规”(Dynamics > General) 组态窗口中组态：加速度 = 10.0，减速度 = 5.0

① 轴由 MC_Halt 任务进行制动，直到进入停止状态。轴的停止状态通过“Done_2”来指示。

② 当 MC_Halt 任务对轴进行制动处理时，另一个运动任务会中止该任务。该中止通过“Abort_2”来标识。

超驰响应

MC_Halt 任务可被下列运动控制任务中止：

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

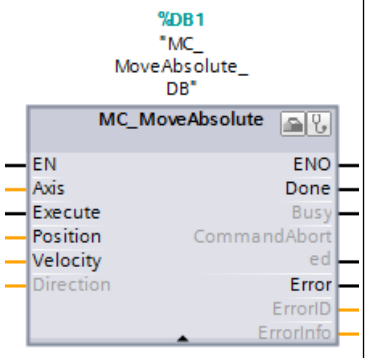
新 MC_Halt 任务可中止下列激活的运动控制任务：

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

10.3.7.6 MC_MoveAbsolute（以绝对方式定位轴）

表格 10-64 MC_MoveAbsolute 指令

LAD/FBD	SCL	说明

	`"MC_MoveAbsolute_DB"(`
	`Axis:=_multi_fb_in_,`
	`Execute:=_bool_in_,`
	`Position:=_real_in_,`
	`Velocity:=_real_in_,`
	`Direction:=_int_in_,`
	`Done=>_bool_out_,`
	`Busy=>_bool_out_,`
	`CommandAborted=>_bool_out_,`
	`Error=>_bool_out_,`
	`ErrorID=>_word_out_,`
	`ErrorInfo=>_word_out_);`
		使用 MC_MoveAbsolute 指令可启动轴到绝对位置的定位运动。
		为了使用 MC_MoveAbsolute 指令，必须先启用轴，同时必须使其回原点。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“MC_MoveAbsolute_DB”是背景 DB 的名称。

表格 10-65 MC_MoveAbsolute 指令的参数

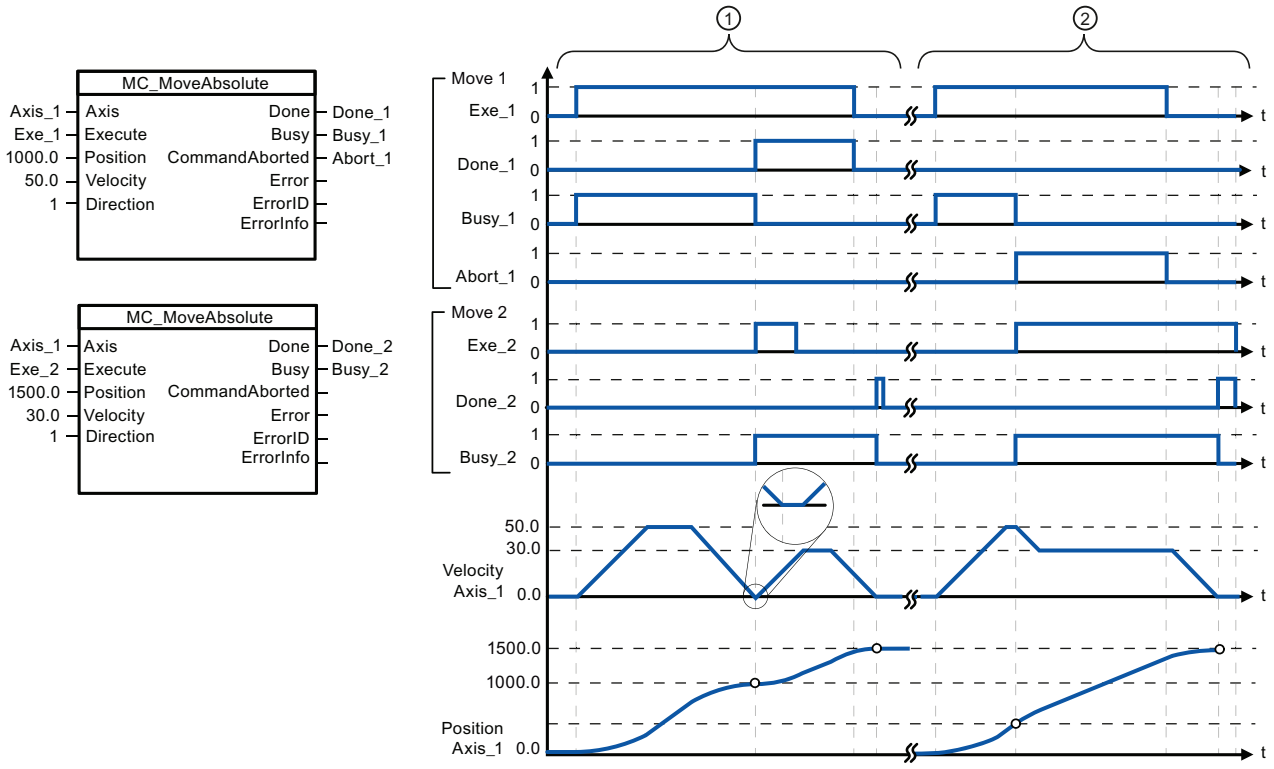
参数和类型		数据类型	说明
Axis	IN	TO_Axis_1	轴工艺对象
Execute	IN	Bool	出现上升沿时开始任务（默认值：False）
Position	IN	Real	绝对目标位置（默认值：0.0） 限值：$-1.0e^{12} \leq$ Position $\leq 1.0e^{12}$

参数和类型		数据类型	说明
Velocity	IN	Real	轴的速度（默认值：10.0） 由于组态的加速度和减速度以及要逼近的目标位置的原因，并不总是能达到此速度。 限值：启动/停止速度 ≤ Velocity ≤ 最大速度
Direction	IN	Int	旋转方向（默认值：0）
Done	OUT	Bool	TRUE = 已达到绝对目标位置
Busy	OUT	Bool	TRUE = 正在执行任务。
CommandAborted	OUT	Bool	TRUE = 任务在执行期间被另一任务中止。
Error	OUT	Bool	TRUE = 任务执行期间出错。出错原因可在“ErrorID”和“ErrorInfo”参数中找到。
ErrorID	OUT	Word	参数“Error”的错误 ID（默认值：0000）
ErrorInfo	OUT	Word	参数“ErrorID”的错误信息 ID（默认值：0000）

用户可以组态位置轴作为模数轴。使用模数轴时，可使用“方向”(Direction) 输入参数选择运动控制方向。如果轴不是模数轴，那么运动控制会忽略“方向”(Direction) 输入。

下表显示了“方向”(Direction) 输入的有效值：

值	枚举	注释
0	SIGN_OF_VELOCITY	速度符号定义运动控制方向
1	正	正向速度运动控制
2	负	负向速度运动控制
3	SHORTEST_WAY	距离目标最短的运动控制



下面的值已在“动态 > 常规”(Dynamics > General) 组态窗口中组态：加速度 = 10.0，减速度 = 10.0

① 轴在 MC_MoveAbsolute 任务的驱动下移动到绝对位置 1000.0 处。轴到达目标位置时，通过“Done_1”对此情况进行标识。“Done_1”= TRUE 时，将启动另一个目标位置为 1500.0 的 MC_MoveAbsolute 任务。由于存在响应时间（例如，用户程序的循环时间等），轴会暂时进入停止状态（请参见放大的细节图）。轴到达新的目标位置时，通过“Done_2”对此情况进行标识。

② 当前 MC_MoveAbsolute 任务将由另一个 MC_MoveAbsolute 任务中止。该中止通过“Abort_1”来标识。轴随后以新的加速度移动到新的目标位置 1500.0 处。到达新的目标位置时，通过“Done_2”对此情况进行标识。

超驰响应

MC_MoveAbsolute 任务可被下列运动控制任务中止：

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

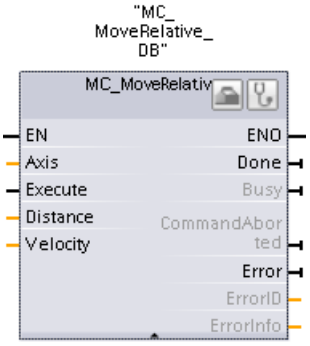
新 MC_MoveAbsolute 任务可中止下列激活的运动控制任务：

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

10.3.7.7 MC_MoveRelative（以相对方式定位轴）

表格 10-66 MC_MoveRelative 指令

LAD/FBD	SCL	说明

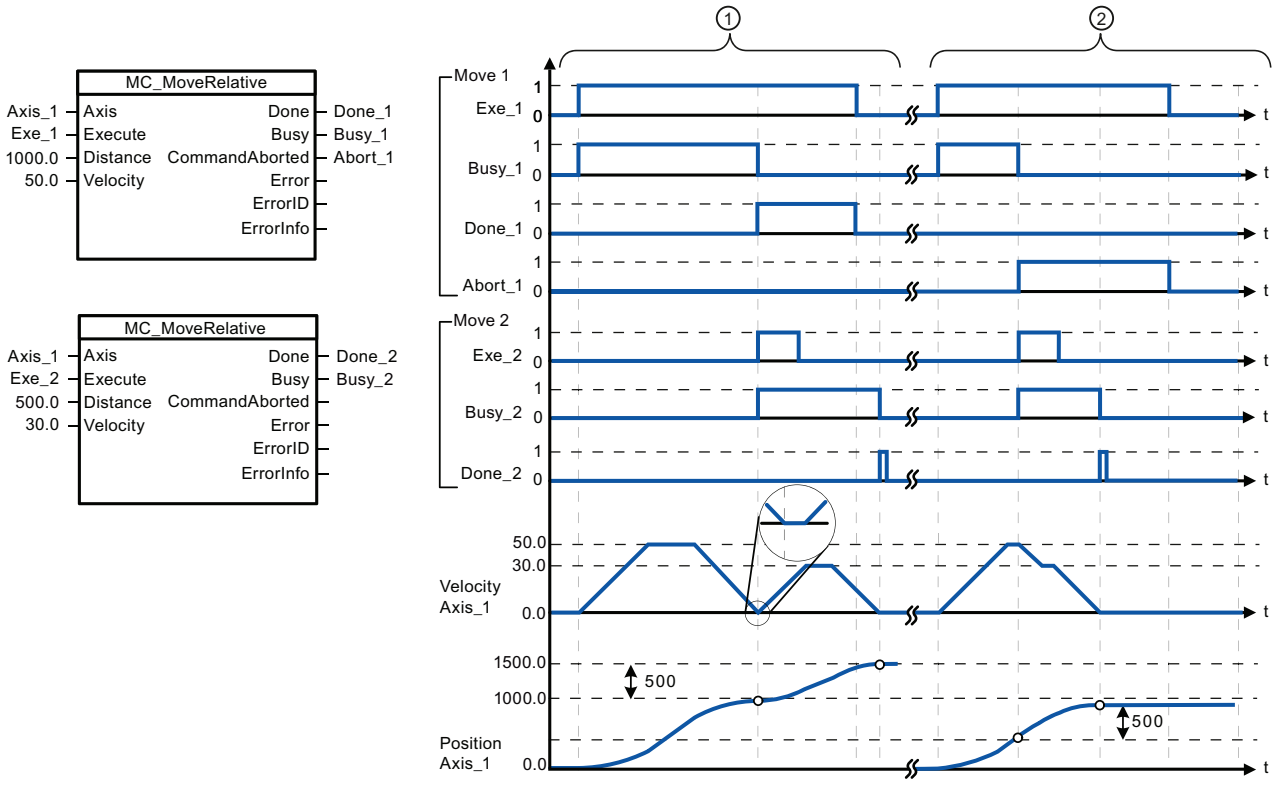
	`"MC_MoveRelative_DB"(`
	`Axis:=_multi_fb_in_,`
	`Execute:=_bool_in_,`
	`Distance:=_real_in_,`
	`Velocity:=_real_in_,`
	`Done=>_bool_out_,`
	`Busy=>_bool_out_,`
	`CommandAborted=>_bool_out_,`
	`Error=>_bool_out_,`
	`ErrorID=>_word_out_,`
	`ErrorInfo=>_word_out_);`
		使用 MC_MoveRelative 指令可启动相对于起始位置的定位运动。
		为了使用 MC_MoveRelative 指令，必须先启用轴。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“MC_MoveRelative_DB”是背景 DB 的名称。

表格 10-67 MC_MoveRelative 指令的参数

参数和类型		数据类型	说明
Axis	IN	TO_Axis_1	轴工艺对象
Execute	IN	Bool	出现上升沿时开始任务（默认值：False）
Distance	IN	Real	定位操作的行进距离（默认值：0.0） 限值：$-1.0e^{12} \leq$ Distance $\leq 1.0e^{12}$
Velocity	IN	Real	轴的速度（默认值：10.0） 由于组态的加速度和减速度以及要行进的距离的原因，并不总是能达到此速度。 限值：启动/停止速度 ≤ Velocity ≤ 最大速度
Done	OUT	Bool	TRUE = 已达到目标位置
Busy	OUT	Bool	TRUE = 正在执行任务。
CommandAborted	OUT	Bool	TRUE = 任务在执行期间被另一任务中止。
Error	OUT	Bool	TRUE = 任务执行期间出错。出错原因可在“ErrorID”和“ErrorInfo”参数中找到。
ErrorID	OUT	Word	参数“Error”的错误 ID（默认值：0000）
ErrorInfo	OUT	Word	参数“ErrorID”的错误信息 ID（默认值：0000）



下面的值已在“动态 > 常规”(Dynamics > General) 组态窗口中组态：加速度 = 10.0，减速度 = 10.0

① 轴在 MC_MoveRelative 任务的驱动下移动 1000.0 的距离（“Distance”）。轴到达目标位置时，通过“Done_1”对此情况进行标识。“Done_1”= TRUE 时，将启动另一个行进距离为 500.0 的 MC_MoveRelative 任务。由于存在响应时间（例如，用户程序的循环时间），轴会暂时进入停止状态（请参见放大的细节图）。轴到达新的目标位置时，通过“Done_2”对此情况进行标识。

② 当前 MC_MoveRelative 任务将由另一个 MC_MoveRelative 任务中止。该中止通过“Abort_1”来标识。轴随后以新的加速度移动一段新的距离（“Distance”）500.0。到达新的目标位置时，通过“Done_2”对此情况进行标识。

超驰响应

MC_MoveRelative 任务可被下列运动控制任务中止：

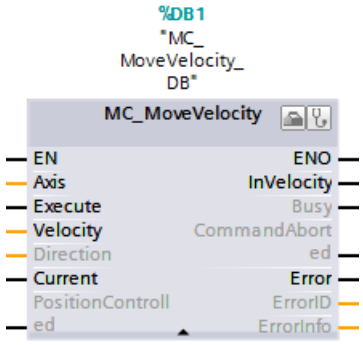
- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

新 MC_MoveRelative 任务可中止下列激活的运动控制任务：

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

10.3.7.8 MC_MoveVelocity（以预定义速度移动轴）

表格 10-68 MC_MoveVelocity 指令

LAD/FBD	SCL	说明
	"MC_MoveVelocity_DB"(Axis:=_multi_fb_in_, Execute:=_bool_in_, Velocity:=_real_in_, Direction:=_int_in_, Current:=_bool_in_, PositionControlled:=_bool_in_, InVelocity=>_bool_out_, Busy=>_bool_out_, CommandAborted=>_bool_out_, Error=>_bool_out_, ErrorID=>_word_out_, ErrorInfo=>_word_out_);	使用 MC_MoveVelocity 指令以指定的速度持续移动轴。 为了使用 MC_MoveVelocity 指令，必须先启用轴。

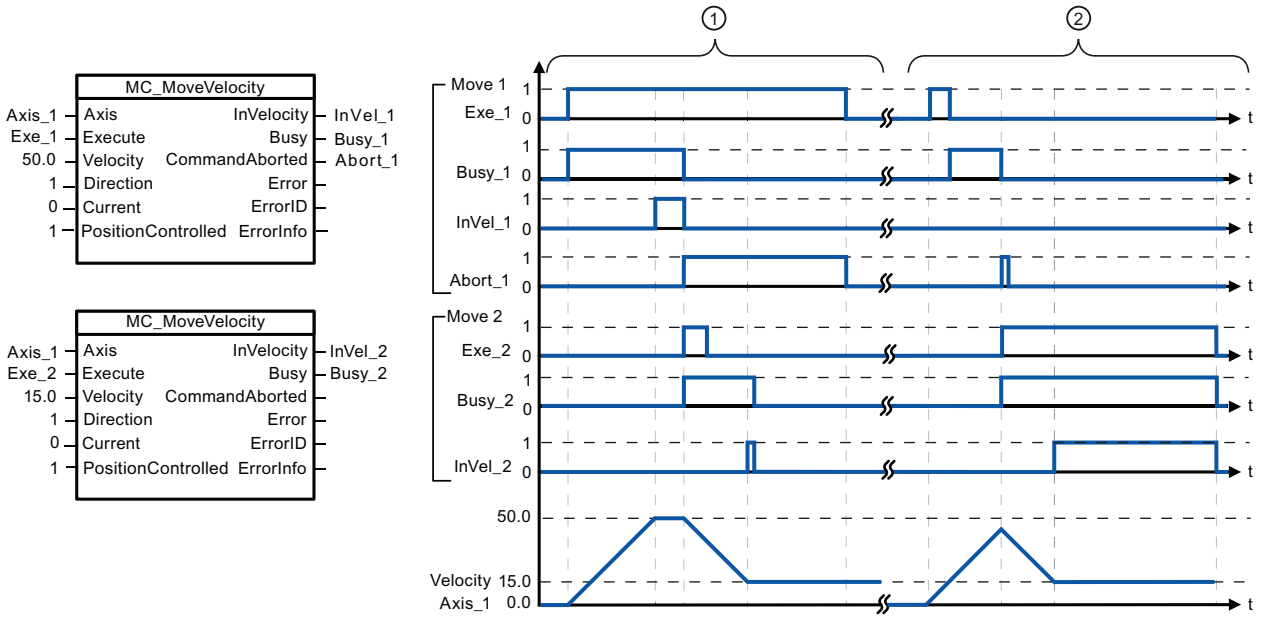
[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“MC_MoveVelocity_DB”是背景 DB 的名称。

表格 10-69 MC_MoveVelocity 指令的参数

参数和类型		数据类型	说明
Axis	IN	TO_SpeedAxis	轴工艺对象
Execute	IN	Bool	出现上升沿时开始任务（默认值：False）
Velocity	IN	Real	指定轴运动的速度（默认值：10.0）100.0） 限值：启动/停止速度 ≤ |Velocity| ≤ 最大速度 （允许 Velocity = 0.0）
Direction	IN	Int	指定方向： • 0：旋转方向与参数“Velocity”中的值符号一致（默认值） • 1：正旋转方向（参数“Velocity”的值符号被忽略。） • 2：负旋转方向（参数“Velocity”的值符号被忽略。）

参数和类型		数据类型	说明
Current	IN	Bool	保持当前速度： ● FALSE：禁用“保持当前速度”。使用参数“Velocity”和“Direction”的值。（默认值） ● TRUE：激活“保持当前速度”。不考虑参数“Velocity”和“Direction”的值。 当轴继续以当前速度运动时，参数 "InVelocity" 返回值 TRUE.
PositionControlled	IN	Bool	● 0：速度控制 ● 1：位置控制（默认值：True）
InVelocity	OUT	Bool	TRUE： ● 如果“Current”= FALSE：已达到参数“Velocity"”中指定的速度。 ● 如果 "Current" = TRUE：轴在启动时以当前速度运动。
Busy	OUT	Bool	TRUE = 正在执行任务。
CommandAborted	OUT	Bool	TRUE = 任务在执行期间被另一任务中止。
Error	OUT	Bool	TRUE = 任务执行期间出错。出错原因可在“ErrorID”和“ErrorInfo”参数中找到。
ErrorID	OUT	Word	参数“Error”的错误 ID（默认值：0000）
ErrorInfo	OUT	Word	参数“ErrorID”的错误信息 ID（默认值：0000）



下面的值已在“动态 > 常规”(Dynamics > General) 组态窗口中组态：加速度 = 10.0，减速度 = 10.0

① 当前 MC_MoveVelocity 任务通过“InVel_1”来指示已达到目标速度。该任务随后会被另一个 MC_MoveVelocity 任务中止。该中止通过“Abort_1”来标识。达到新的目标速度 15.0 时，将通过“InVel_2”对此情况进行指示。轴随后以新的恒定加速度继续移动。

② 在达到目标速度之前，当前 MC_MoveVelocity 任务会由另一个 MC_MoveVelocity 任务中止。该中止通过“Abort_1”来标识。达到新的目标速度 15.0 时，将通过“InVel_2”对此情况进行指示。轴随后以新的恒定加速度继续移动。

超驰响应

MC_MoveVelocity 任务可被下列运动控制任务中止：

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

新 MC_MoveVelocity 任务可中止下列激活的运动控制任务：

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

说明

速度设置为零 (Velocity = 0.0) 时的行为

“Velocity”= 0.0 的 MC_MoveVelocity 任务（如 MC_Halt 任务）可中止激活的运动任务并利用组态的减速度停止轴运动。轴停止运动后，输出参数“InVelocity”将指示 TRUE 并至少持续一个程序循环的时间。

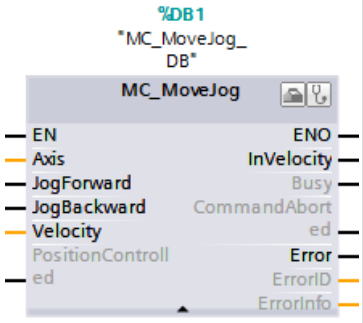
“Busy”的值在减速过程中为 TRUE，并随“InVelocity”一起变为 FALSE。如果设置了参数“Execute”= TRUE，则锁存“InVelocity”和“Busy”。

启动 MC_MoveVelocity 任务时，将设置工艺对象的状态位“SpeedCommand”。轴停止运动后，将立即设置状态位“ConstantVelocity”。启动新运动任务时，这两个位均会适应的新情况。

10.3.7.9 MC_MoveJog（在点动模式下移动轴）

表格 10-70 MC_MoveJog 指令

LAD/FBD	SCL	说明

	`"MC_MoveJog_DB"(`
	`Axis:=_multi_fb_in_,`
	`JogForward:=_bool_in_,`
	`JogBackward:=_bool_in_,`
	`Velocity:=_real_in_,`
	
	`PositionControlled:=_bool_in_,`
	`InVelocity=>_bool_out_,`
	`Busy=>_bool_out_,`
	`CommandAborted=>_bool_out_,`
	`Error=>_bool_out_,`
	`ErrorID=>_word_out_,`
	`ErrorInfo=>_word_out_);`
		使用 MC_MoveJog 指令以指定的速度在点动模式下持续移动轴。该指令通常用于测试和调试。
		为了使用 MC_MoveJog 指令，必须先启用轴。

[1] STEP 7 会在插入指令时自动创建 DB。

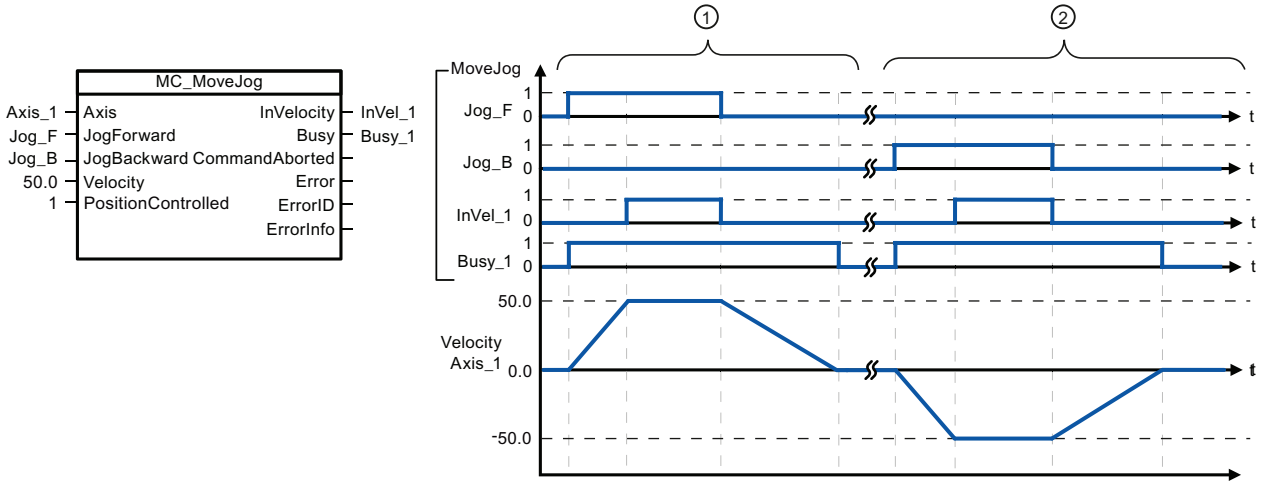
[2] 在 SCL 示例中，“MC_MoveJog_DB”是背景 DB 的名称。

表格 10-71 MC_MoveJog 指令的参数

参数和类型		数据类型	说明
Axis	IN	TO_SpeedAxis	轴工艺对象
JogForward[1]	IN	Bool	只要此参数为 TRUE，轴就会以参数“Velocity”中指定的速度沿正向移动。参数“Velocity”的值符号被忽略。（默认值：False）

参数和类型		数据类型	说明
JogBackward[1]	IN	Bool	只要此参数为 TRUE，轴就会以参数“Velocity”中指定的速度沿负向移动。参数“Velocity”的值符号被忽略。（默认值：False）
Velocity	IN	Real	点动模式的预设速度（默认值：10.0）100.0） 限值：启动/停止速度 ≤ |Velocity| ≤ 最大速度
PositionControlled	IN	Bool	• 0：速度控制 • 1：位置控制（默认值：True）
InVelocity	OUT	Bool	TRUE = 已达到参数“Velocity”中指定的速度。
Busy	OUT	Bool	TRUE = 正在执行任务。
CommandAborted	OUT	Bool	TRUE = 任务在执行期间被另一任务中止。
Error	OUT	Bool	TRUE = 任务执行期间出错。出错原因可在“ErrorID”和“ErrorInfo”参数中找到。
ErrorID	OUT	Word	参数“Error”的错误 ID（默认值：0000）
ErrorInfo	OUT	Word	参数“ErrorID”的错误信息 ID（默认值：0000）

[1] 如果 JogForward 和 JogBackward 参数同时为 TRUE，则轴将以组态后的减速度停止运动。将通过参数“Error”、“ErrorID”和“ErrorInfo”指示错误。



下面的值已在“动态 > 常规”(Dynamics > General) 组态窗口中组态：加速度 = 10.0，减速度 = 5.0

① 通过“Jog_F”在点动模式下沿正方向移动轴。达到目标速度 50.0 时，将通过“InVelo_1”对此情况进行指示。轴会在 Jog_F 复位后再次制动直到停止。

② 通过“Jog_B”在点动模式下沿负方向移动轴。达到目标速度 50.0 时，将通过“InVelo_1”对此情况进行指示。轴会在 Jog_B 复位后再次制动直到停止。

超驰响应

MC_MoveJog 任务可被下列运动控制任务中止：

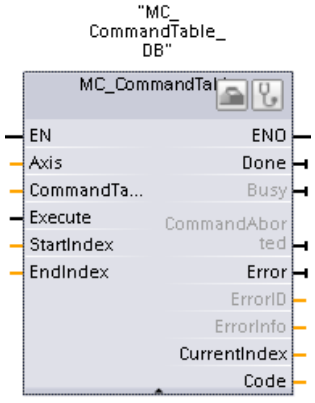
- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

新 MC_MoveJog 任务可中止下列激活的运动控制任务：

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

10.3.7.10 MC_CommandTable（按运动顺序运行轴命令）

表格 10-72 MC_CommandTable 指令

LAD/FBD	SCL	说明
	"MC_CommandTable_DB"(Axis:=_multi_fb_in_, CommandTable:=_multi_fb_in_, Execute:=_bool_in_, StartIndex:=_uint_in_, EndIndex:=_uint_in_, Done=>_bool_out_, Busy=>_bool_out_, CommandAborted=>_bool_out_, Error=>_bool_out_, ErrorID=>_word_out_, ErrorInfo=>_word_out_, CurrentIndex=>_uint_out_, Code=>_word_out_);	针对电机控制轴执行一系列单个运动，这些运动可组合成一个运动序列。 在脉冲串输出的工艺对象命令表 (TO_CommandTable_PTO) 中，可以组态这些单个的运动。

[1] STEP 7 会在插入指令时自动创建 DB。

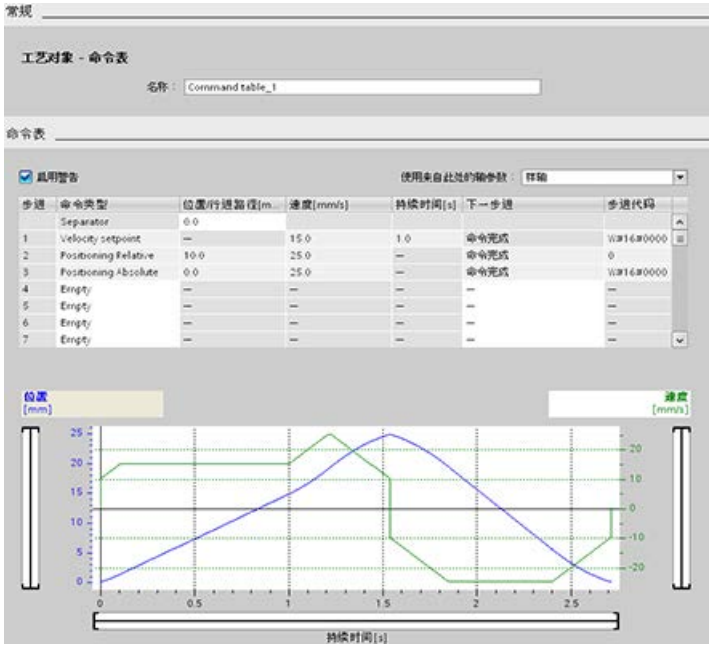
[2] 在 SCL 示例中，“MC_CommandTable_DB”是背景 DB 的名称。

表格 10-73 MC_CommandTable 指令的参数

参数和类型		数据类型	初始值	说明
Axis	IN	TO_Axis_1	-	轴工艺对象
Table	IN	TO_CommandTable_1	-	命令表工艺对象
Execute	IN	Bool	FALSE	使用上升沿启动作业

参数和类型		数据类型	初始值	说明
StartIndex	IN	Int	1	从此步骤开始命令表处理 限制：1 ≤ **StartIndex** ≤ EndIndex
EndIndex	IN	Int	32	从此步骤结束命令表处理 限制：StartIndex ≤ **EndIndex** ≤ 32
Done	OUT	Bool	FALSE	MC_CommandTable 处理已成功完成
Busy	OUT	Bool	FALSE	正在运行
CommandAborted	OUT	Bool	FALSE	该任务在处理期间被另一任务中止。
Error	OUT	Bool	FALSE	处理时出错。出错原因会通过参数 ErrorID 和 ErrorInfo. 指出。
ErrorID	OUT	Word	16#0000	错误标识符
ErrorInfo	OUT	Word	16#0000	错误信息
Step	OUT	Int	0	当前在处理的步骤
Code	OUT	Word	16#0000	当前处理步骤的用户定义标识符

可在“命令表”(Command Table) 组态窗口中创建所需的运动序列，并根据趋势图中的图形视图来检查结果。



可选择要用于处理命令表的命令类型。最多可输入 32 项作业。将按顺序处理命令。

表格 10-74 MC_CommandTable 命令类型

命令类型	说明
Empty	空白用作占位符，以便添加任意命令。在处理命令表时，忽略空白条目。
Halt	暂停轴。 注：该命令仅在“Velocity setpoint”命令之后使用。
Positioning Relative	根据距离定位轴。该命令将按给定的距离和速度移动轴。
Positioning Absolute	根据位置定位轴。该命令以指定的速度将轴移到给定位置。
Velocity setpoint	按给定速度移动轴。
Wait	等待给定期间结束。“Wait”不会停止已激活的行进运动。
Separator	在选定行上方添加“分隔”线。利用分隔线，可在单个命令表中定义多个轨迹。

执行 MC_CommandTable 的先决条件：

- 工艺对象 TO_Axis_PTO V2.0 必须已正确组态。
- 工艺对象 TO_CommandTable_PTO 必须已正确组态。
- 必须释放轴。

超驰响应

MC_CommandTable 任务可被下列运动控制任务中止：

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog
- MC_CommandTable

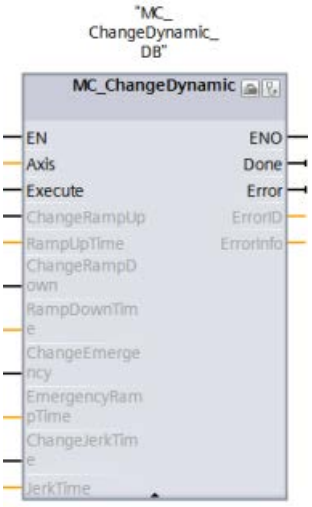
新 MC_CommandTable 任务可中止下列激活的运动控制任务：

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog
- MC_CommandTable
- 启动第一个“Positioning Relative”、“Positioning Absolute”、“Velocity setpoint”或“Halt”命令时的当前运动控制作业

10.3.7.11 MC_ChangeDynamic（更改轴的动态设置）

表格 10-75 MC_ChangeDynamic 指令

LAD/FBD	SCL	说明

	`"MC_ChangeDynamic_DB"(`
	`Execute:=_bool_in_,`
	`ChangeRampUp:=_bool_in_,`
	`RampUpTime:=_real_in_,`
	`ChangeRampDown:=_bool_in_,`
	`RampDownTime:=_real_in_,`
	`ChangeEmergency:=_bool_in_,`
	`EmergencyRampTime:=_real_in_,`
	`ChangeJerkTime:=_bool_in_,`
	`JerkTime:=_real_in_,`
	`Done=>_bool_out_,`
	`Error=>_bool_out_,`
	`ErrorID=>_word_out_,`
	`ErrorInfo=>_word_out_);`
		更改运动控制轴的动态设置：
		● 更改加速时间（加速度）值
		● 更改减速时间（减速度）值
		● 更改急停减速时间（急停减速度）值
		● 更改平滑时间（冲击）值

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“MC_ChangeDynamic_DB”是背景 DB 的名称。

表格 10-76 MC_ChangeDynamic 指令的参数

参数和类型		数据类型	说明
Axis	IN	TO_Axis_1	轴工艺对象
Execute	IN	Bool	出现上升沿时开始命令。默认值：FALSE
ChangeRampUp	IN	Bool	TRUE = 根据输入参数“RampUpTime”更改加速时间。默认值：FALSE
RampUpTime	IN	Real	在没有冲击限制的情况下，从静止状态加速到组态的最大速度的时间（以秒为单位）。默认值：5.00 更改将会影响变量 <轴名称>.Config.DynamicDefaults.Acceleration。该变量的说明中便会显示更改所产生的影响。
ChangeRampDown	IN	Bool	TRUE = 按照输入参数"RampDownTime"更改减速时间。默认值：FALSE

参数和类型		数据类型	说明
RampDownTime	IN	Real	在没有冲击限制的情况下，轴从组态的最大速度减速到静止状态的时间（以秒为单位）。默认值：5.00 更改将会影响变量 <轴名称>. Config.DynamicDefaults.Deceleration。该变量的说明中便会显示更改所产生的影响。
ChangeEmergency	IN	Bool	TRUE = 按照输入参数"EmergencyRampTime"更改急停减速时间。 默认值：FALSE
EmergencyRampTime	IN	Real	在没有冲击限制的情况下，在急停模式下，轴从组态的最大速度减速到静止状态的时间（以秒为单位）。默认值：2.00 更改将会影响变量 <轴名称>. Config.DynamicDefaults.EmergencyDeceleration。该变量的说明中便会显示更改所产生的影响。
ChangeJerkTime	IN	Bool	TRUE = 根据输入参数“JerkTime”更改平滑时间。默认值：FALSE
JerkTime	IN	Real	用于轴加速度和减速度的平滑时间（以秒为单位）。默认值：0.25 更改将会影响变量 <轴名称>. Config.DynamicDefaults.Jerk。该变量的说明中便会显示更改所产生的影响。
Done	OUT	Bool	TRUE = 更改的值已写入工艺数据块。在更改生效时将显示变量的描述。默认值：FALSE
Error	OUT	Bool	TRUE = 命令执行期间出错。出错原因可在“ErrorID”和“ErrorInfo”参数中找到。默认值：FALSE
ErrorID	OUT	Word	错误标识符。默认值：16#0000
ErrorInfo	IN	Word	错误信息。默认值：16#0000

执行 MC_ ChangeDynamic 的先决条件：

- 工艺对象 TO_Axis_PTO V2.0 必须已正确组态。
- 必须释放轴。

说明

只能使用 MC_ChangeDynamic 指令通过 PTO（脉冲串输出）连接驱动器。

超驰响应

MC_ChangeDynamic 命令无法被其它任何运动控制命令中止。

新的 MC_ChangeDynamic 命令不会中止任何已激活的运动控制作业。

说明

为输入参数“RampUpTime”、“RampDownTime”、“EmergencyRampTime”和“RoundingOffTime”指定的值可以使生成的轴参数“加速度”(acceleration)、“延时”(delay)、“急停延时”(emergency stop-delay) 和“冲击”(jerk) 超出允许限值。

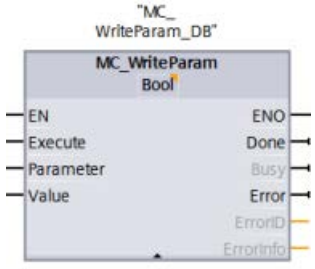
请确保将 MC_ChangeDynamic 参数保持在轴工艺对象的动态组态设置的限制范围内。

10.3.7.12 MC_WriteParam（写入工艺对象的参数）

使用 MC_WriteParam 指令可写入选定数量的参数来通过用户程序更改轴功能。

表格 10-77 MC_WriteParam 指令

LAD/FBD	SCL	说明

	`"MC_WriteParam_DB"(`
	`Parameter:=_variant_in_,`
	`Value:=_variant_in_,`
	`Execute:=_bool_in_,`
	`Done:=_bool_out_,`
	`Error:=_real_out_,`
	`ErrorID:=_word_out_,`
	`ErrorInfo:=_word_out_);`
		使用 MC_WriteParam 指令可写入公共参数（例如，加速度值和用户 DB 值）。

[1] STEP 7 会在插入指令时自动创建 DB。

[2] 在 SCL 示例中，“MC_WriteParam_DB”是背景 DB 的名称。

可以写入公共参数。不能写入“MotionStatus”和“StatusBits”。下表列出了有效参数：

可写入参数的名称	可写入参数的名称
Actor.InverseDirection	DynamicDefaults.Acceleration
Actor.DirectionMode	DynamicDefaults.Deceleration
Actor.DriveParameter.PulsesPerDriveRevolution	DynamicDefaults.Jerk
Sensor[1].ActiveHoming.Mode	DynamicDefaults.EmergencyDeceleration
Sensor[1].ActiveHoming.SideInput	PositionLimitsHW.Active
Sensor[1].ActiveHoming.Offset	PositionLimitsHW.MaxSwitchedLevel

可写入参数的名称	可写入参数的名称
Sensor[1].ActiveHoming.SwitchedLevel	PositionLimitsHW.MinSwitchedLevel
Sensor[1].PassiveHoming.Mode	PositionLimitsSW.Active
Sensor[1].PassiveHoming.SideInput	PositionLimitsSW.MinPosition
Sensor[1].PassiveHoming.SwitchedLevel	PositionLimitsSW.MaxPosition
Units.LengthUnit	Homing.AutoReversal
Mechanics.LeadScrew	Homing.ApproachDirection
DynamicLimits.MinVelocity	Homing.ApproachVelocity
DynamicLimits.MaxVelocity	Homing.ReferencingVelocity

表格 10-78 MC_WriteParam 指令的参数

参数和类型		数据类型	说明
PARAMNAME	IN	Variant	在其中写入值的参数名称
VALUE	IN	Variant	写入到所分配参数的值
EXECUTE	IN	Bool	启动指令。默认值：FALSE
DONE	OUT	Bool	已写入值。默认值：FALSE
BUSY	OUT	Bool	如果为 TRUE，则正在执行指令。默认值：FALSE
ERROR	OUT	Real	如果为 TRUE，则发生了错误。默认值：FALSE
ERRORID	OUT	Word	错误 ID
ERRORINFO	OUT	Word	ERRORID 的相关信息

表格 10-79 ERRORID 和 ERRORINFO 的条件代码

ERRORID (W#16#...)	ERRORINFO (W#16#...)	说明
0	0	成功更改轴 TO-DB 参数
$8410_{[1]}$	$0028_{[1]}$	设置了无效参数（长度不正确的轴 TO-DB 参数）
$8410_{[1]}$	$0029_{[1]}$	设置了无效参数（无轴 TO-DB 参数）
$8410_{[1]}$	$002B_{[1]}$	设置了无效参数（只读轴 TO-DB 参数）
$8410_{[1]}$	$002C_{[1]}$	设置了有效参数，但未禁用轴

ERRORID (W#16#...)	ERRORINFO (W#16#...)	说明
Config Error[2]	Config Error[2]	设置了超出范围的有效参数（公共只读轴 TO-DB 参数）
Config Error[3]	Config Error[3]	设置了超出范围的有效参数（公共轴 TO-DB 参数）

[1] MC_WriteParam 出错

[2] MC_Power 出错

[3] MC_Power 和 MC_MoveXXX 或 MC_CommandTable 出错

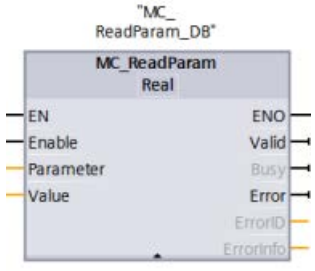
说明

对于通过 PROFIdrive/模拟量输出实现的驱动器连接，无法使用 MC_WriteParam 写入参数，这需要重启工艺对象。TIA Portal 在线帮助中有关该指令说明是错误的。

10.3.7.13 MC_ReadParam 指令（读取工艺对象的参数）

使用 MC_ReadParam 指令可读取选定数量的参数，以指示轴输入过程中定义的轴的当前位置、速度等。

表格 10-80 MC_ReadParam 指令

LAD/FBD	SCL	说明
	"MC_ReadParam_DB"(Enable:=_bool_in_, Parameter:=_variant_in_, Value:=_variant_in_out_, Valid:=_bool_out_, Busy:=_bool_out_, Error:=_real_out_, ErrorID:=_word_out_, ErrorInfo:=_word_out_);	使用 MC_ReadParam 指令可读取单个状态值，与周期控制点无关。

1 STEP 7 会在插入指令时自动创建 DB。

2 在 SCL 示例中，“MC_ReadParam_DB”是背景 DB 的名称。

MC_ReadParam 指令通过启用来生效。只要输入“启用”为真，指令就会将指定的“参数”读取至“值”存储位置。

每个周期控制点 (CCP) 的“MotionStatus”和“Position”值根据当前 HSC 值进行更新。

“MotionStatus”的“Velocity”值是当前时间段（更新周期 ~10ms）结束时的命令速度。MC_ReadParam 同样可以读取该值。

如果发生错误，指令将切换到错误状态，只有“启用”输入的新上升沿才能将其复位。

表格 10-81 MC_ReadParam 指令的参数

参数和类型		数据类型	说明
ENABLE	IN	Bool	启动指令。默认值：FALSE
PARAMETER	IN	Variant	指向要读取的 TO 参数的指针
VALID	OUT	Bool	如果为 TRUE，则已读取该值。默认值：FALSE
BUSY	OUT	Bool	如果为 TRUE，则正在执行指令。默认值：FALSE
ERROR	OUT	Real	如果为 TRUE，则发生了错误。默认值：FALSE
ERRORID	OUT	Word	错误 ID。默认值：0
ERRORINFO	OUT	Word	与 ERRORID. 默认值相关的信息：0
VALUE	INOUT	Variant	指向存储该读取值位置的指针

表格 10-82 ERRORID 和 ERRORINFO 的条件代码

ERRORID (W#16#...)	ERRORINFO (W#16#...)	说明
0	0	成功读取参数
8410	0028	无效参数（长度不正确）
8410	0029	无效参数（无 TO-DB）
8410	0030	无效参数（不可读）
8411	0032	无效参数（值错误）

TO 参数

轴“MotionStatus”由四个值组成。可在程序运行时读取这些值，来监视这些值的变化：

变量名称	数据类型	通过 MC_ReadParam 读取
MotionStatus：	结构	-
• Position	REAL	√

变量名称	数据类型	通过 MC_ReadParam 读取
• Velocity	REAL	√
• Distance	REAL	√
• TargetPosition	REAL	√

10.3.8 监视激活的命令

10.3.8.1 监视具有输出参数“Done”的 MC 指令

具有输出参数“Done”的运动控制指令通过输入参数“Execute”启动，并且具有明确的结论（例如，对于运动控制指令“MC_Home”：回原点已成功）。任务完成后，轴处于停止状态。

- 如果任务已功完成，则输出参数“Done”的值为 TRUE。
- 输出参数“Busy”、“CommandAborted”和“Error”发出信号，指示任务仍在处理、已中止或有未决的错误。运动控制指令“MC_Reset”无法中止，所以没有输出参数“CommandAborted”。
  - 在运动控制任务处理期间，输出参数“Busy”的值为 TRUE。如果任务已完成、中止或因错误停止，则输出参数“Busy”的值将变为 FALSE。无论输入参数“Execute”的信号状态是什么，都会发生这种变化。
  - 输出参数“Done”、“CommandAborted”和“Error”的值至少在一个周期内都为 TRUE。当输入参数“Execute”设置为 TRUE 时，将锁存这些状态消息。

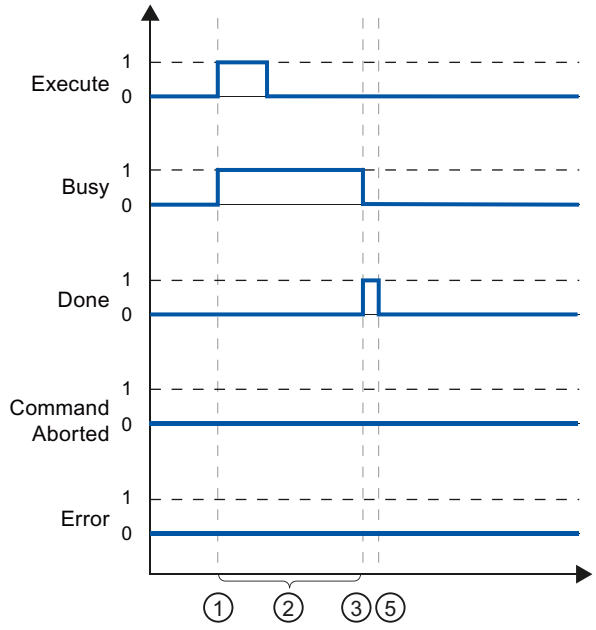
以下运动控制指令的任务具有明确的结论：

- MC_Reset
- MC_Home
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative

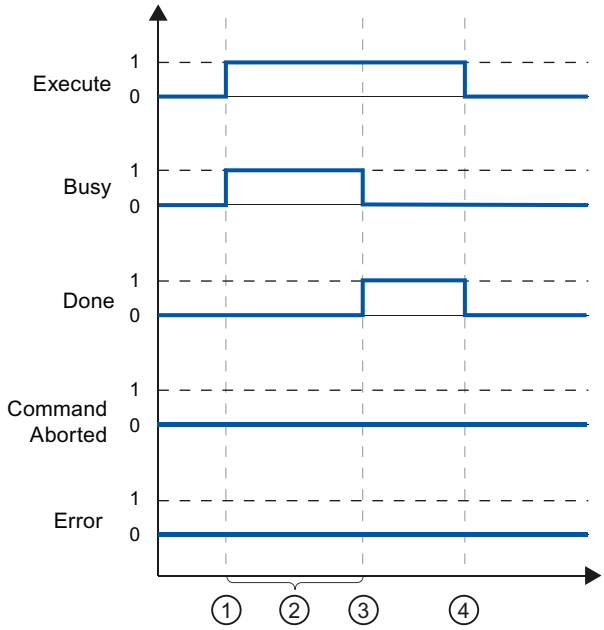
下图针对各种示例情况显示了状态位的操作。

- 第一个示例显示了已完成的任务的轴行为。如果运动控制任务已在对其下结论前完全执行，则通过输出参数“Done”的 TURE 值对此进行指示。输入参数“Execute”的信号状态影响输出参数“Done”中的显示持续时间。
- 第二个示例显示了已中止的任务的轴行为。如果运动控制任务在执行期间中止，则通过输出参数“CommandAborted”的 TURE 值对此进行指示。输入参数“Execute”的信号状态影响输出参数“CommandAborted”中的显示持续时间。
- 第三个示例显示了出现错误时的轴行为。如果在运动控制任务执行期间出错，则通过输出参数“Error”的 TURE 值对此进行指示。输入参数“Execute”的信号状态影响输出参数“Error”中的显示持续时间。

表格 10-83　示例 1 - 任务完成执行



如果在处理任务期间“Execute”= FALSE



如果在完成任务后“Execute”= FALSE

① 输入参数“Execute”的上升沿时启动任务。根据编程情况，“Execute”在任务执行期间仍然可能重置为 FALSE 值，或者保持为 TURE 值，直到任务完成为止。

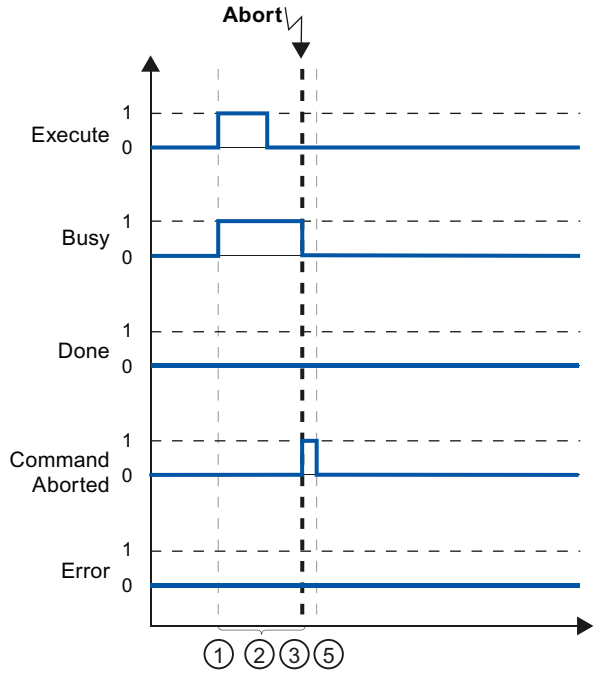
② 任务激活期间，输出参数“Busy”的值为 TRUE。

③ 任务结束后（例如，对于运动控制指令“MC_Home”：回原点已成功），输出参数“Busy”的值变为 FALSE，并且“Done”的值变为 TRUE。

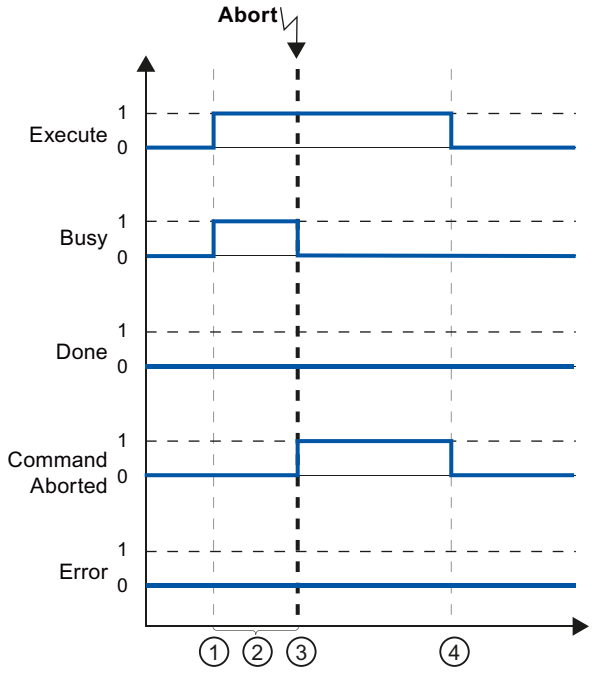
④ 如果“Execute”的值在任务完成之前保持为 TRUE，则“Done”的值也将保持为 TRUE 并且其值随“Execute”一起变为 FALSE。

⑤ 如果“Execute”已在任务完成之前设置为 FALSE，则“Done”的值仅在一个执行周期内为 TRUE。

表格 10-84　示例 2 - 中止任务



如果任务中止之前“Execute”= FALSE



如果任务中止之后“Execute”= FALSE

① 输入参数“Execute”的上升沿时启动任务。根据编程情况，“Execute”在任务执行期间仍然可能重置为 FALSE 值，或者保持为 TURE 值，直到任务完成为止。

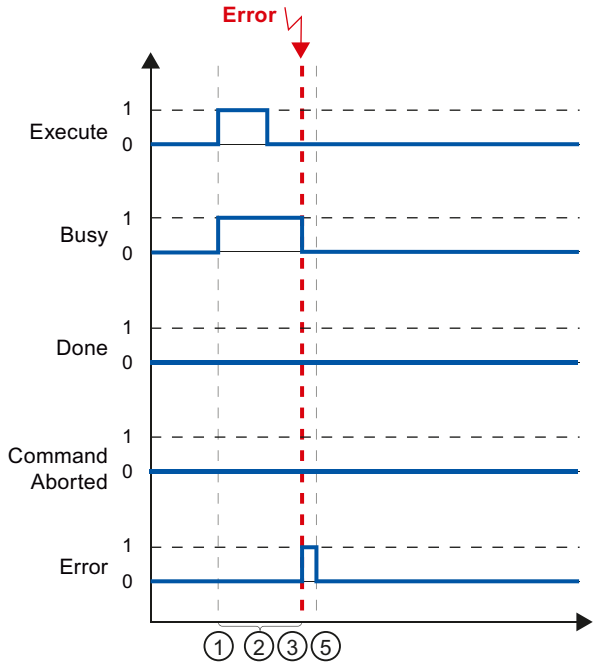
② 任务激活期间，输出参数“Busy”的值为 TRUE。

③ 在任务执行期间，任务被其它运动控制任务中止。如果任务中止，输出参数“Busy”的值将变为 FALSE 且“CommandAborted”的值将变为 TRUE。

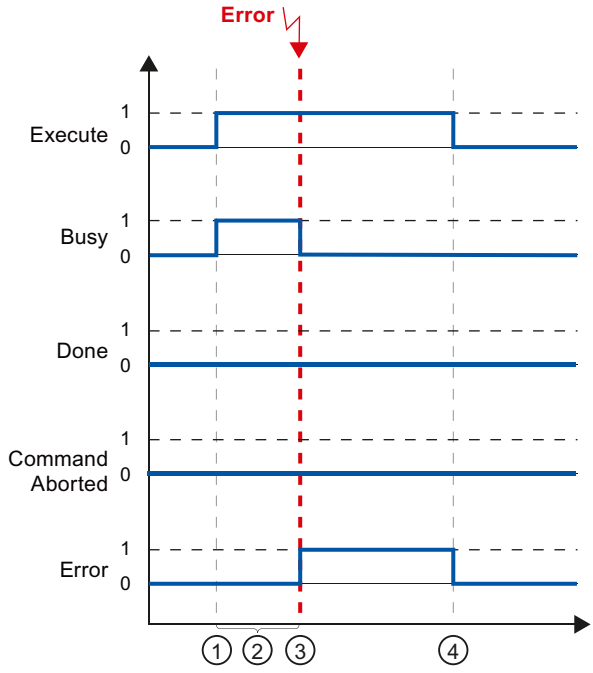
④ 如果“Execute”的值在任务中止之前保持为 TRUE，则“CommandAborted”的值也将保持为 TRUE 并且其值随“Execute”一起变为 FALSE。

⑤ 如果“Execute”已在任务中止之前设置为 FALSE，则“CommandAborted”的值仅在一个执行周期内为 TRUE。

表格 10-85　示例 3 - 任务执行期间出错



如果出现错误之前“Execute”= FALSE



如果出现错误之后“Execute”= FALSE

① 输入参数“Execute”的上升沿时启动任务。根据编程情况，“Execute”在任务执行期间仍然可能重置为 FALSE 值，或者保持为 TURE 值，直到任务完成为止。

② 任务激活期间，输出参数“Busy”的值为 TRUE。

③ 任务执行期间出错。出错时，输出参数“Busy”的值将变为 FALSE 且“Error”的值将变为 TRUE。

④ 如果“Execute”的值在出错之前保持为 TRUE，则“Error”的值也将保持为 TRUE 并且其值仅随“Execute”一起变为 FALSE。

⑤ 如果“Execute”已在出错之前设置为 FALSE，则“Error”的值仅在一个执行周期内为 TRUE。

10.3.8.2 监控 MC_Velocity

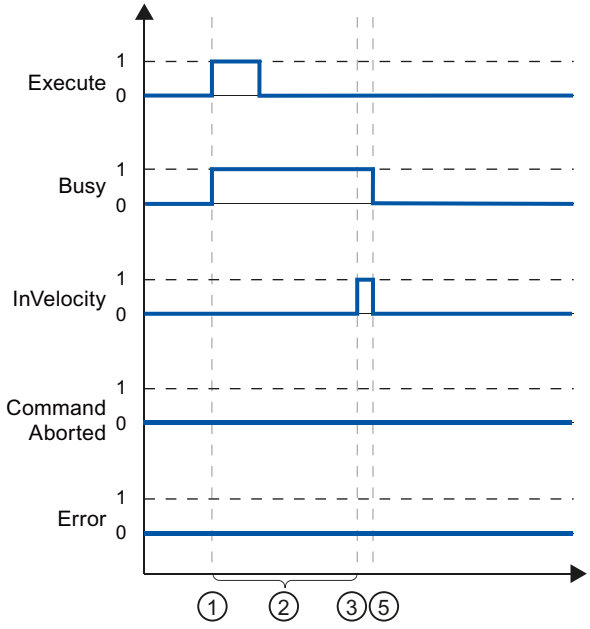
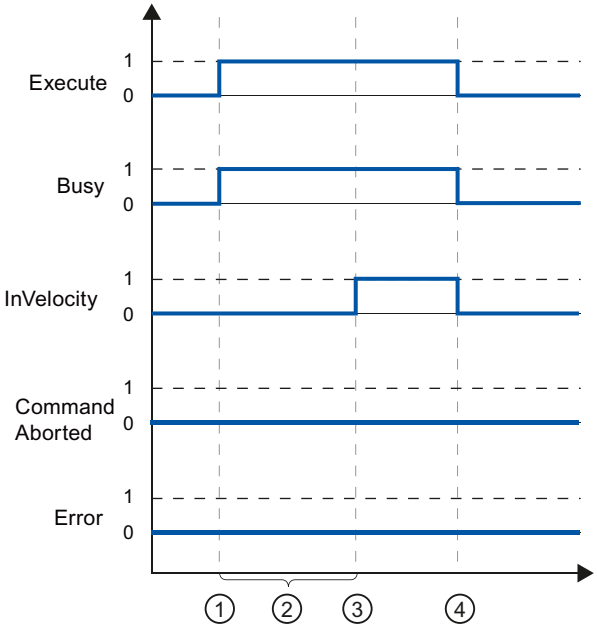
运动控制指令“MC_MoveVelocity”的任务以指定的速度执行移动：

- 未明确定义运动控制指令“MC_MoveVelocity”任务的结束。第一次达到设置的速度且轴恒速运转时，就实现了任务目标。如果达到设置的速度，则通过输出参数“InVelocity”的 TURE 值对此进行指示。
- 已达到设置的速度且输入参数“Execute”的值已设置为 FALSE 时，任务完成。然而，在任务完成时轴运动尚未完成。例如，可以使用运动控制任务“MC_Halt”停止轴运动。
- 输出参数“Busy”、“CommandAborted”和“Error”发出信号，指示任务仍在处理、已中止或有未决的错误。
  - 在运动控制任务执行期间，输出参数“Busy”的值为 TRUE。如果任务已完成、中止或因错误停止，则输出参数“Busy”的值将变为 FALSE。无论输入参数“Execute”的信号状态是什么，都会发生这种变化。
  - 当输出参数“InVelocity”、“CommandAborted”和“Error”的条件满足时，它们的值至少在一个周期内都为 TRUE。当输入参数“Execute”设置为 TRUE 时，将锁存这些状态消息。

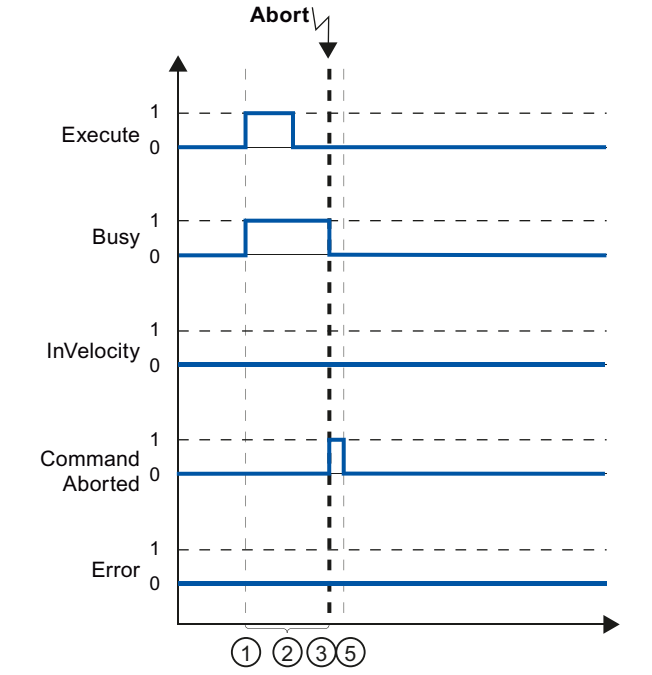
下图针对各种示例情况显示了状态位的操作：

- 第一个示例显示了轴达到设置的速度时的行为。如果运动控制任务已在达到设置的速度前完成执行，则通过输出参数“InVelocity”的 TURE 值对此进行指示。输入参数“Execute”的信号状态影响输出参数“InVelocity”中的显示持续时间。
- 第二个示例显示了在达到设置的速度之前中止任务时的轴行为。如果运动控制任务在达到设置速度前中止，则通过输出参数“CommandAborted”的 TURE 值对此进行指示。输入参数“Execute”的信号状态影响输出参数“CommandAborted”中的显示持续时间。
- 第三个示例显示了在达到设置的速度之前出错时的轴行为。如果在运动控制任务执行期间，达到设置速度之前出错，则通过输出参数“Error”的 TURE 值对此进行指示。输入参数“Execute”的信号状态影响输出参数“Error”中的显示持续时间。

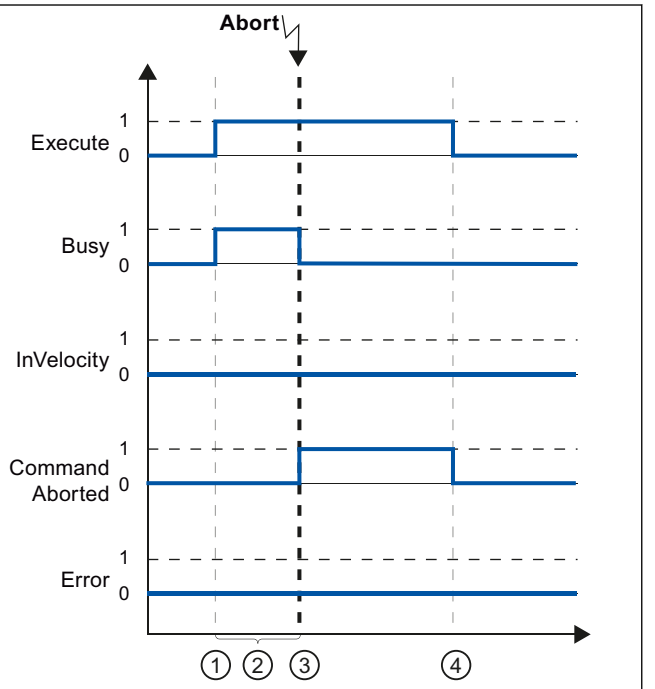
表格 10-86　示例 1 - 如果达到设置的速度

 如果达到组态的速度之前“Execute”= FALSE	 如果达到组态的速度之后“Execute”= FALSE
① 输入参数“Execute”的上升沿时启动任务。根据编程情况，“Execute”的值可在达到设置的速度前重置为 FALSE，也可在达到设置的速度后重置为 FALSE。 ② 任务激活期间，输出参数“Busy”的值为 TRUE。 ③ 达到设置的速度时，输出参数“InVelocity”的值变为 TRUE。 ④ 如果“Execute”的值在达到设置的速度后仍保持为 TRUE，则任务将保持激活状态。“InVelocity”和“Busy”的值保持为 TRUE 并且其状态仅随“Execute”变为 FALSE。 ⑤ 如果“Execute”已在达到设置的速度前重置为 FALSE，则任务将在达到设置的速度时完成。“InVelocity”的值仅在一个执行周期内为 TRUE，并且随“Busy”一起变为 FALSE。	

表格 10-87　示例 2 - 如果任务在达到设置速度前中止



如果任务中止之前“Execute”= FALSE



如果任务中止之后“Execute”= FALSE

① 输入参数“Execute”的上升沿时启动任务。根据编程情况，“Execute”在任务执行期间仍然可以被重置为 FALSE 值，或者保持为 TURE 值，直到任务中止为止。

② 任务激活期间，输出参数“Busy”的值为 TRUE。

③ 在任务执行期间，任务被其它运动控制任务中止。如果任务中止，输出参数“Busy”的值将变为 FALSE 且“CommandAborted”的值将变为 TRUE。

④ 如果“Execute”的值在任务中止之前保持为 TRUE，则“"CommandAborted”的值也将保持为 TRUE 并且其状态随“Execute”一起变为 FALSE。

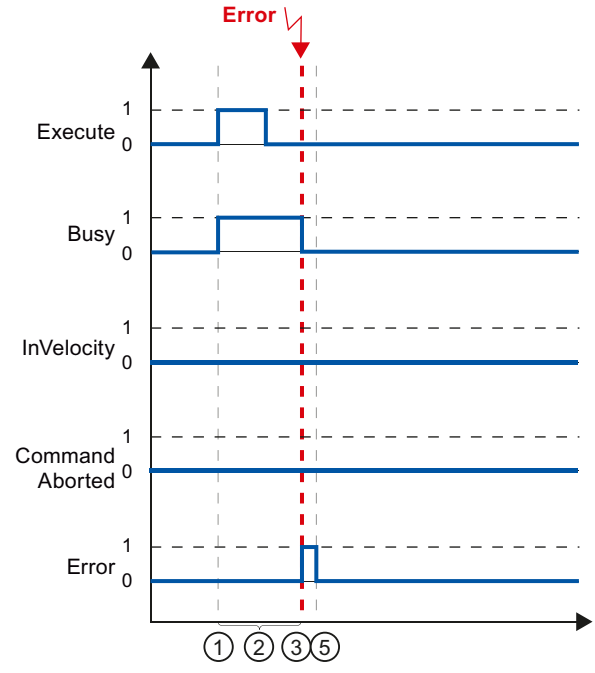
⑤ 如果“Execute”已在任务中止之前重置为 FALSE，则“CommandAborted”的值仅在一个执行周期内为 TRUE。

说明

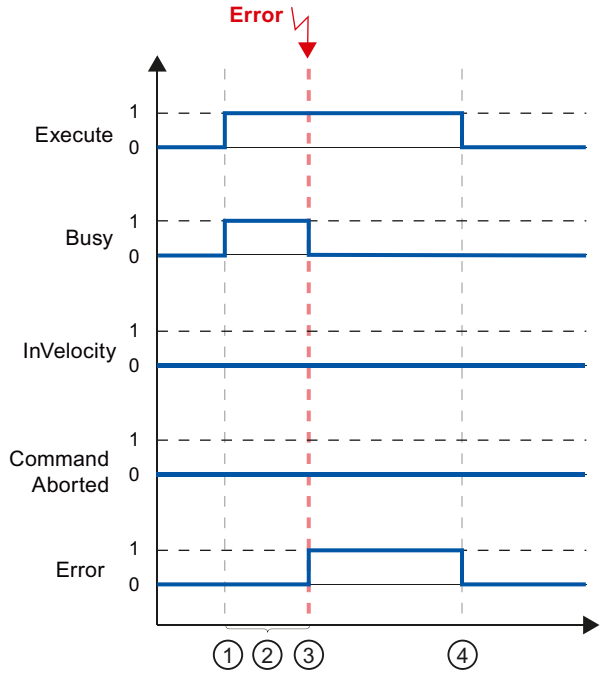
在以下条件下，输出参数“CommandAborted”不指示出现中止：

- 已达到设置的速度，输入参数“Execute”的值为 FALSE，并且已启动一个新的运动控制任务。
- 达到设置的速度并且输入参数“Execute”的值为 FALSE 时，任务完成。所以不会将新任务的启动指示为中止。

表格 10-88　示例 3 - 如果在达到设置的速度之前出错



如果出现错误之前“Execute”= FALSE



如果出现错误之后“Execute”= FALSE

① 输入参数“Execute”的上升沿时启动任务。根据编程情况，“Execute”在任务执行期间仍然可以被重置为 FALSE 值，或者保持为 TURE 值，直到出现错误为止。

② 任务激活期间，输出参数“Busy”的值为 TRUE。

③ 任务执行期间出错。出错时，输出参数“Busy”的值将变为 FALSE 且“Error”的值将变为 TRUE。

④ 如果“Execute”的值在出错之前保持为 TRUE，则“Error”的值也将保持为 TRUE 并且其状态仅随“Execute”一起变为 FALSE。

⑤ 如果“Execute”已在出错之前重置为 FALSE，则“Error”的值仅在一个执行周期内为 TRUE。

说明

在以下条件下，输出参数“Error”不指示出现错误：

- 已达到设置的速度，输入参数“Execute”的值为 FALSE，并且发生轴错误（例如，逼近软件限位开关）。
- 达到设置的速度并且输入参数“Execute”的值为 FALSE 时，任务完成。任务完成后，轴错误仅在运动控制指令“MC_Power”中指示。

10.3.8.3 监控 MC_MoveJog

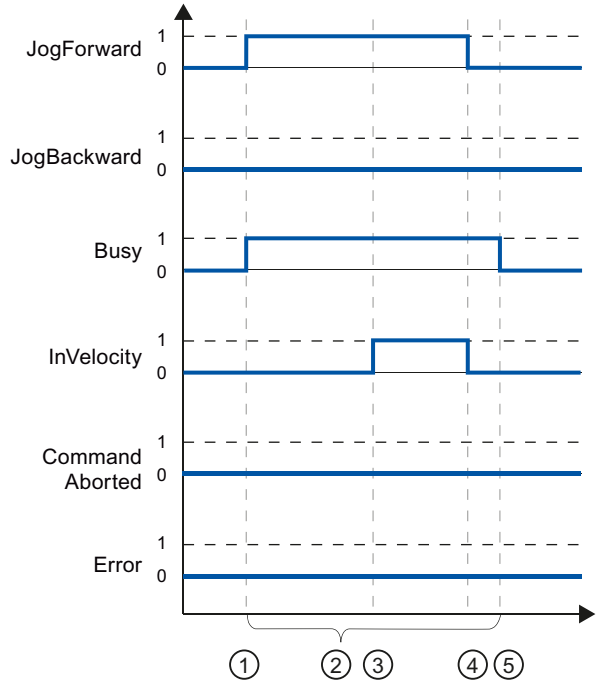
运动控制指令“MC_MoveJog”的任务是实现点动操作。

- 未明确定义运动控制任务“MC_MoveJog”的结束。第一次达到设置的速度且轴恒速运转时，就实现了任务目标。如果达到设置的速度，则通过输出参数“InVelocity”的 TURE 值对此进行指示。
- 输入参数“JogForward”或“JogBackward”的值已设置为 FALSE 并且轴已停止时，命令完成。
- 输出参数“Busy”、“CommandAborted”和“Error”发出信号，指示任务仍在处理、已中止或有未决的错误。
  - 在运动控制任务处理期间，输出参数“Busy”的值为 TRUE。如果任务已完成、中止或因错误停止，则输出参数“Busy”的值将变为 FALSE。
  - 只要轴在以设置的速度运转，输出参数“InVelocity”的值就为 TRUE。输出参数“CommandAborted”和“Error”保持该状态至少一个周期。只要输入参数“JogForward”或“JogBackward”设置为 TRUE，就锁存这些状态消息。

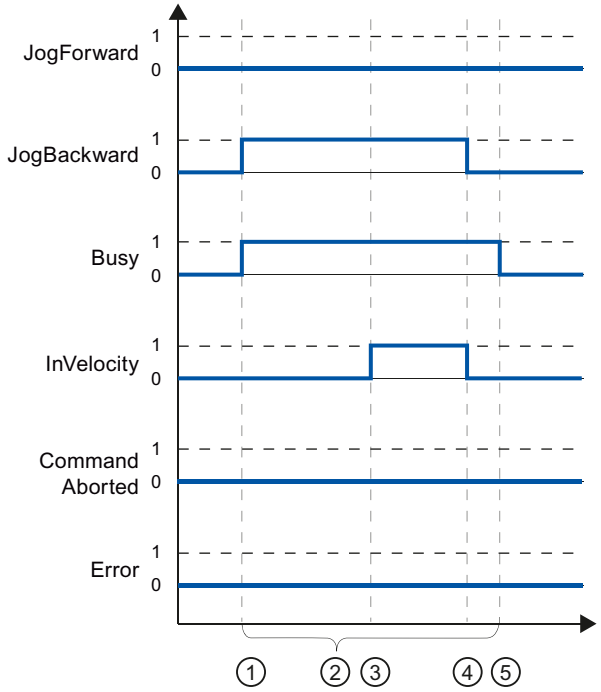
下图针对各种示例情况显示了状态位的操作。

- 第一个示例显示了达到并保持设置的速度时的轴行为。如果运动控制任务已在达到设置的速度前完成执行，则通过输出参数“InVelocity”的 TURE 值对此进行指示。
- 第二个示例显示了任务中止时的轴行为。如果运动控制任务在执行期间中止，则通过输出参数“CommandAborted”的 TURE 值对此进行指示。该行为与是否达到设置的速度无关。
- 第三个示例显示了出现错误时的轴行为。如果在运动控制任务执行期间出错，则通过输出参数“Error”的 TURE 值对此进行指示。该行为与是否达到设置的速度无关。

表格 10-89　示例 1 - 如果达到并保持设置的速度



JogForward



JogBackward

① 输入参数“JogForward”或“JogBackward”的上升沿时启动任务。

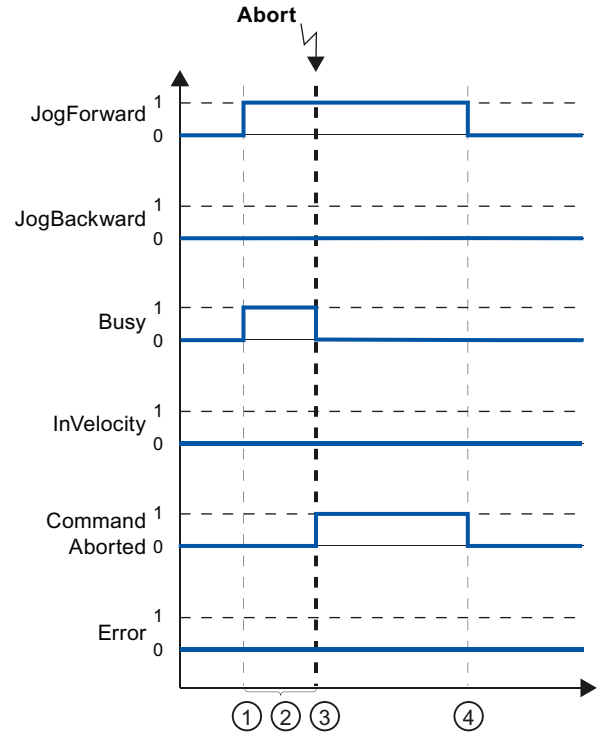
② 任务激活期间，输出参数“Busy”的值为 TRUE。

③ 达到设置的速度时，输出参数“InVelocity”的值变为 TRUE。

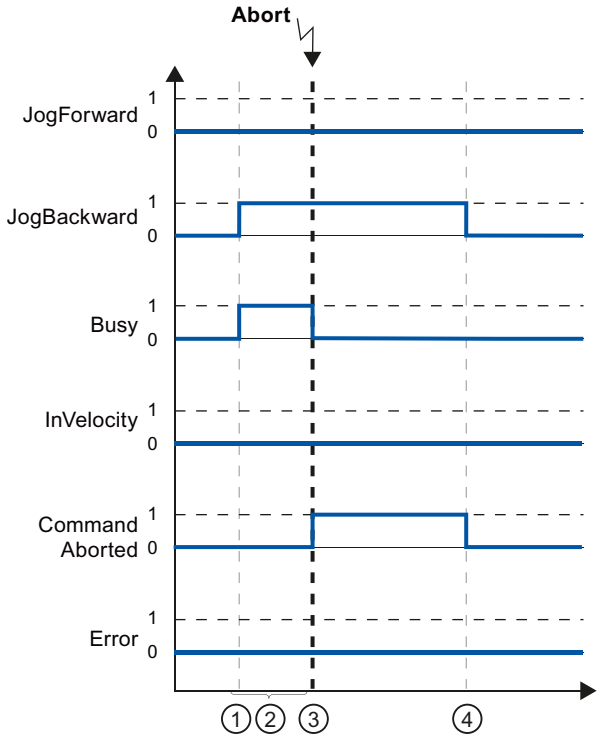
④ 输入参数“JogForward”或“JogBackward”的值重置为 FALSE 时，轴运动结束。轴开始减速。结果，轴不再恒速运转并且输出参数“InVelocity”的状态变为 FALSE。

⑤ 如果轴已停止，则运动控制任务完成并且输出参数“Busy”的值将变为 FALSE。

表格 10-90　示例 2 - 如果任务在执行期间中止



JogForward



JogBackward

① 输入参数“JogForward”或“JogBackward”的上升沿时启动任务。

② 任务激活期间，输出参数“Busy”的值为 TRUE。

③ 在任务执行期间，任务被其它运动控制任务中止。如果任务中止，输出参数“Busy”的值将变为 FALSE 且“CommandAborted”的值将变为 TRUE。

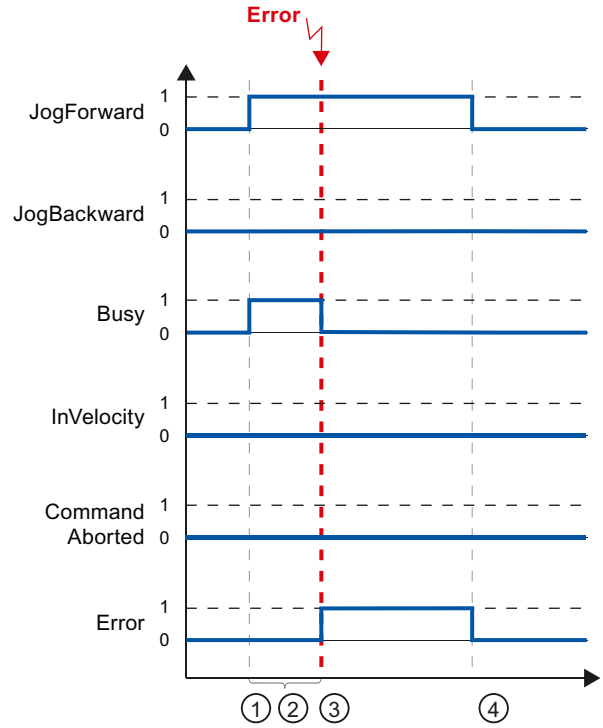
④ 输入参数“JogForward”或“JogBackward”的值重置为 FALSE 时，输出参数“CommandAborted”的值将变为 FALSE。

说明

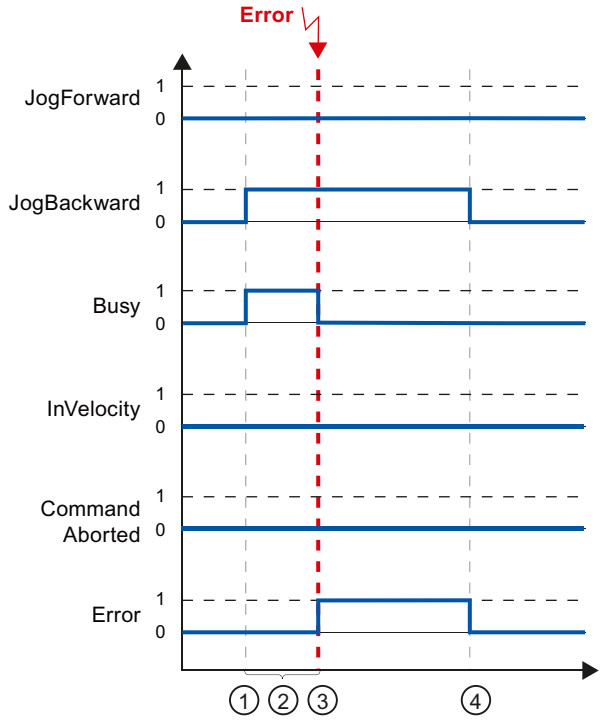
如果满足以下所有条件，则在输出参数“CommandAborted”中指示任务中止并且仅持续一个执行周期：

输入参数“JogForward”和“JogBackward”的值为 FALSE（但轴仍在减速），并且新的运动控制任务已启动。

表格 10-91　示例 3 - 如果在任务执行期间出错



JogForward



JogBackward

① 输入参数“JogForward”或“JogBackward”的上升沿时启动任务。

② 任务激活期间，输出参数“Busy”的值为 TRUE。

③ 任务执行期间出错。出错时，输出参数“Busy”的值将变为 FALSE 且“Error”的值将变为 TRUE。

④ 输入参数“JogForward”或“JogBackward”的值重置为 FALSE 时，输出参数“Error”的值将变为 FALSE。

说明

如果满足以下所有条件，则在输出参数“Error”中指示出错并且仅持续一个执行周期：

输入参数“JogForward”和“JogBackward”的值为 FALSE（但轴仍在减速），并且发生新错误（例如，逼近软件限位开关）。

10.3.9　运动控制的 ErrorID 和 ErrorInfo

下表列出了运动控制指令和数据调整中指示的所有 ErrorID 和 ErrorInfo。除了错误原因，还列出了消除错误的补救措施。

根据错误响应，轴在停止时如果发生操作错误，将会停止轴。可能会出现以下错误响应：

- **取消启用**
  输出设定点 0，并取消启用。轴将根据驱动器中的组态进行制动，并转入停止状态。
- **通过急停斜坡功能进行停止**
  将中止处于激活状态的运动命令。轴将通过在“工艺对象 > 扩展参数 > 动态 > 急停斜坡功能”(Technology object > Extended parameters > Dynamics > Emergency stop ramp)中所组态的急停减速度功能进行制动（没有任何加加速度的限制），并转入停止状态。

伴随轴停止的运行错误

ErrorID	ErrorInfo	说明	解决方法	错误反应
16#8000		**驱动器错误，丢失“驱动器就绪”信号**		-
	16#0001	-	使用“MC_Reset”指令确认错误；根据需要提供驱动器信号，重新启动命令	
16#8001		**已触发下限软限位开关**		通过急停斜坡功能进行停止。
	16#000E	已经以当前组态的减速度到达下限软限位开关的位置	使用“MC_Reset”指令确认错误；使用运动命令使轴沿正向移动，超出软限位开关的范围	
	16#000F	已经以急停减速度到达下限软限位开关的位置		
	16#0010	已经以急停减速度超出下限软限位开关的位置		
16#8002		**已触发上限软限位开关**		通过急停斜坡功能进行停止。
	16#000E	已经以当前组态的减速度到达上限软限位开关的位置	使用“MC_Reset”指令确认错误；使用运动命令使轴沿负向移动，超出软限位开关的范围	
	16#000F	已经以急停减速度到达上限软限位开关的位置		
	16#0010	已经以急停减速度超出上限软限位开关的位置		

ErrorID	ErrorInfo	说明	解决方法	错误反应
16#8003		已到达下限硬限位开关		通过 PTO (Pulse Train Output) 的驱动器连接： 通过急停斜坡功能进行停止。 通过 PROFIdrive/模拟量输出的驱动器连接： 取消启用。
	16#000E	已到达下限硬限位开关。轴以急停减速度停止。 （主动归位期间，未找到归位开关）	使用指令“MC_Reset”确认已启用轴的错误；使用运动命令使轴沿正向移动，超出硬限位开关的范围。	
16#8004		已到达上限硬限位开关		通过 PTO (Pulse Train Output) 的驱动器连接： 通过急停斜坡功能进行停止。 通过 PROFIdrive/模拟量输出的驱动器连接： 取消启用。
	16#000E	已到达上限硬限位开关。轴以急停减速度停止。 （主动归位期间，未找到归位开关）	使用指令“MC_Reset”确认已启用轴的错误；使用运动命令使轴沿负向移动，超出硬限位开关的范围。	
16#8005		PTO/HSC 已被另一个轴占用		-
	16#0001	-	**轴未正确组态：** 更正 PTO (Pulse Train Output) / HSC (High Speed Counter) 的组态并将其下载到控制器	
			多个轴将通过一个 PTO 运行： 另一个轴正在使用 PTO / HSC。如果要控制当前轴，则必须使用“MC_Power”(Enable = FALSE) 禁用另一个轴。	

ErrorID	ErrorInfo	说明	解决方法	错误反应
16#8006		控制面板中发生通信错误		取消启用。
	16#0012	已超时	检查电缆连接，然后再次按下“手动控制”(Manual control) 按钮。	
16#8007		该轴无法启用		-
	16#0025	正在重新启动	请等待，直到轴重新启动完成。	
	16#0026	正在 RUN 模式下执行加载过程	请等待，直到加载过程完成。	
16#8008		无效的运动方向		-
	16#002E	不允许所选的运动方向。	• 调整运动方向，然后重新启动该命令。 • 在工艺对象组态的“扩展参数 > 机械”(Extended parameters > Mechanics) 下可调整允许的旋转方向。重新启动命令。	
	16#002F	不能将所选的运动方向反转。		
16#8009		未找到参考开关/编码器零位标记		通过急停斜坡功能进行停止。
	16#0033	组态、编码器的硬件或安装、归位开关中出现错误。	• 连接合适的设备。 • 检查设备 (I/O)。 • 比较 HW Config 中的组态和工艺对象。	

ErrorID	ErrorInfo	说明	解决方法	错误反应
16#800A		编码器发出报警消息		取消启用。
	16#0001	-	检查设备的功能、连接和 I/O。	
	16#0034	编码器中的硬件错误		
	16#0035	编码器变脏		
	16#0036	读取编码器绝对值期间出错	将驱动器中的编码器类型或编码器参数 P979 与工艺对象的组态数据进行比较。	
	16#0037	监视编码器的零位标记	编码器报告零位标记监视错误（Gx_XIST2 中的错误代码 0x0002，请参见 PROFIdrive 配置文件）。 检查设备的电磁兼容性 (EMC)。	
	16#0038	编码器处于“停止”(Parking) 状态	• 有关错误原因，请查看所连接的驱动器或编码器。 • 检查错误消息是否可能由驱动器或编码器的调试操作触发。	
	16#0040	PROFIdrive：总线上的编码器故障（站故障）。	检查设备的功能、连接和 I/O。	
	16#0041	PROFIdrive：编码器设备状况错误		
16#800B		位置超出范围		取消启用。
	16#0039	正方向超出范围	使轴回到有效的实际值范围。	
	16#003A	负方向超出范围		
	16#003B	位置控制时钟周期中实际位置的改变大于模数长度。	调整所用编码器的模数长度。	

ErrorID	ErrorInfo	说明	解决方法	错误反应
16#800C		驱动器发出报警消息		取消启用。
	16#0001	-	检查设备的功能、连接和 I/O。	
	16#003C	PROFIdrive：驱动装置信号“请求的控制”故障		
	16#003D	PROFIdrive：驱动装置已关闭		
	16#003E	PROFIdrive：总线上的驱动装置故障（站故障）		
	16#003F	PROFIdrive：驱动装置设备状况错误	• 检查设备的功能、连接和 I/O。 • 将 HW Config 的时钟参数（PROFIBUS 线路、驱动器或编码器的从 OM）和执行系统的时钟参数进行比较。Tmapc 和伺服必须以相同的时钟周期时间组态。	
16#800D		超出了允许跟随误差		取消启用。
	16#0001	-	• 检查控制回路的组态。 • 检查编码器的信号方向。 • 检查跟随误差监控的组态。	
16#800E		硬限位开关处发生错误		取消启用。
	16#0042	激活硬限位开关的任意行进方向非法	由于激活了硬限位开关，将禁用编程的运动方向。 向相反方向缩回轴。	
	16#0043	硬限位开关极性反转，轴无法释放	检查硬限位开关的机械组态。	
	16#0044	两个硬限位开关均激活，轴无法释放		

ErrorID	ErrorInfo	说明	解决方法	错误反应
16#800F		目标范围发生错误		取消启用。
	16#0045	未达到目标范围	在定位容差时间内未达到目标范围。 ● 检查定位监控的组态。 ● 检查控制回路的组态。	
	16#0046	再次离开目标范围	在最短停留时间内已离开目标范围。 ● 检查定位监控的组态。 ● 检查控制回路的组态。	
16#8010		不是模数轴时，下限软限位开关的位置值大于上限软限位开关的位置值		取消启用。
	16#0001	-	更改软限位开关的位置。	

不伴随轴停止的运行错误

ErrorID	ErrorInfo	说明	解决方法
16#8200		轴未启用	
	16#0001	-	启用轴；重新启动命令。
	16#003D	如果关闭了采用模拟驱动器连接的驱动器，则显示 16#003D	启用轴；重新启动命令。
16#8201		轴已由另一个“MC_Power”实例启用	
	16#0001	-	仅通过一个“MC_Power”实例启用轴。
16#8202		同时执行最大数量的运动控制命令（对于通过 PTO (Pulse Train Output) 的驱动器连接，最多同时执行 200 条命令，对于通过 PROFIdrive/模拟量输出的驱动器连接，最多同时执行 100 条命令）	
	16#0001	-	减少并行激活命令数；重新启动命令 如果运动控制指令的参数“Busy” = TRUE，则说明命令已激活。
16#8203		轴当前在“手动控制”(Manual control) 模式（轴控制面板）下运行	
	16#0001	-	退出“手动控制”；重新启动命令。
16#8204		轴未归位	

ErrorID	ErrorInfo	说明	解决方法
	16#0001	-	使用指令“MC_Home”使轴归位；重新启动命令。
16#8205		**轴当前由用户程序控制（该错误仅显示在轴控制面板中）**	
	16#0013	轴已在用户程序中启用	使用指令“MC_Power”禁用轴或者在轴控制面板中再次选择“手动控制”(Manual control)
16#8206		**工艺对象尚未激活**	
	16#0001	-	使用指令“MC_Power" Enable = TRUE 启用轴或者在轴控制面板中启用轴。
16#8207		**命令被拒绝**	
	16#0016	正在执行主动归位；无法启动另一种归位方法。	等到主动归位完成或通过运动命令（例如，“MC_Halt”）中止主动归位。
	16#0018	当轴正在主动或被动归位时，不能使用命令表进行移动。	一直等到直接地或被动归位完成。
	16#0019	正在处理命令表时，轴不能主动或被动归位。	等到命令表完成或通过运动命令（例如，“MC_Halt”）中止命令表。
	16#0052	指定的位置超出了数值限制。	在运动控制指令中输入有效的位置值。
	16#0053	轴正在沿斜坡上升。	等到轴准备就绪，可运行。
	16#0054	实际值无效	要执行“MC_Home”命令，实际值必须有效。 请检查实际值的状态。工艺对象 <轴名称>. “StatusSensor.State”变量的值必须为 2（有效值）。
16#8208		**最大速度和启动/停止速度的差值无效**	
	16#0002	值不是有效数字	更正该值；重新启动命令。
	16#000A	值小于或等于 0。	
16#8209		**对工艺对象“轴”的加速度无效**	
	16#0002	值不是有效数字	更正该值；重新启动命令。
	16#000A	值小于或等于 0。	

ErrorID	ErrorInfo	说明	解决方法
16#820A		无法重新启动轴	
	16#0013	轴已在用户程序中启用	使用“MC_Power”指令禁用轴；再次重新启动。
	16#0027	轴当前在“手动控制”模式（轴控制面板）下运行	退出“手动控制”；再次重新启动。
	16#002C	轴未禁用。	禁用轴；重新启动命令。
	16#0047	工艺对象未准备好重新启动。	重新下载项目。
	16#0048	不满足工艺对象的重新启动条件。	禁用工艺对象。
16#820B		无法执行命令表	
	16#0026	正在 RUN 模式下执行加载过程	请等待，直到加载过程完成。
16#820C		无组态可用	
	16#0001	-	内部错误 联系服务热线。

块参数错误

ErrorID	ErrorInfo	说明	解决方法
16#8400		运动控制指令的“Position”参数值无效	
	16#0002	值不是有效数字	更正该值；重新启动命令。
	16#0005	该值超出值范围（大于 1E+12）	
	16#0006	该值超出值范围（小于 1E+12）	
16#8401		运动控制指令的“Distance”参数值无效	

ErrorID	ErrorInfo	说明	解决方法
	16#0002	值不是有效数字	更正该值；重新启动命令。
	16#0005	该值超出值范围（大于 1E+12）	
	16#0006	该值超出值范围（小于 1E+12）	
16#8402		**运动控制指令的“Velocity”参数值无效**	
	16#0002	值不是有效数字	更正该值；重新启动命令。
	16#0008	值大于组态的最大速度	
	16#0009	值小于组态的启动/停止速度	
	16#0024	值小于 0	
16#8403		**运动控制指令的“Direction”参数值无效**	
	16#0011	所选值无效	更正所选值；重新启动命令。
16#8404		**运动控制指令的“Mode”参数值无效**	
	16#0011	所选值无效	更正所选值；重新启动命令。
	16#0015	未组态主动/被动归位	更正组态并将其下载到控制器；启用轴并重新启动命令。
	16#0017	虽然硬限位开关已禁用，但还是激活了在硬限位开关处反向	• 使用变量 <轴名称>.PositionLimitsHW.Active = TRUE 激活硬限位开关，并重新启动命令。 • 更正组态并将其下载到控制器；启用轴并重新启动命令。
	16#0055	增量编码器的模式无效	需使用参数"Mode" = 0, 1, 2, 3 启动增量编码器的归位操作。
	16#0056	绝对值编码器的模式无效	绝对值编码器无法执行被动和主动归位操作（“Mode”= 2、3）。 需使用参数“Mode”= 0、1 启动绝对值编码器的归位操作。

ErrorID	ErrorInfo	说明	解决方法
16#8405		运动控制指令的“StopMode”参数值无效	
	16#0011	所选值无效	更正所选值；再次启用轴。
16#8406		不允许同时正向和反向点动	
	16#0001	-	采取措施确保参数“JogForward”和“JogBackward”的信号状态不会同时为 TRUE；重新启动命令。
16#8407		只有禁用激活的轴后，才允许使用指令“MC_Power”切换到另一个轴。	
	16#0001	-	禁用激活的轴；然后可以切换到其它轴并启用该轴。
16#8408		运动控制指令的“Axis”参数值无效	
	16#001A	指定的值与所需的工艺对象版本不匹配	更正该值；重新启动命令。
	16#001B	指定的值与所需的工艺对象类型不匹配	
	16#001C	指定的值不是运动控制工艺数据块	
16#8409		运动控制指令的“CommandTable”参数值无效	
	16#001A	指定的值与所需的工艺对象版本不匹配	更正该值；重新启动命令。
	16#001B	指定的值与所需的工艺对象类型不匹配	
	16#001C	指定的值不是运动控制工艺数据块	
16#840A		运动控制指令的“StartStep”参数值无效	
	16#000A	值小于或等于 0。	更正该值；重新启动命令。
	16#001D	开始步进大于结束步进	
	16#001E	值大于 32	
16#840B		运动控制指令的“EndStep”参数值无效	

ErrorID	ErrorInfo	说明	解决方法
	16#000A	值小于或等于 0。	更正该值；重新启动命令。
	16#001E	值大于 32	
16#840C		**运动控制指令的“RampUpTime”参数值无效**	
	16#0002	值不是有效数字	更正该值；重新启动命令。
	16#000A	值小于或等于 0。	
16#840D		**运动控制指令的“RampDownTime”参数值无效**	
	16#0002	值不是有效数字	更正该值；重新启动命令。
	16#000A	值小于或等于 0。	
16#840E		**运动控制指令的“EmergencyRampTime”参数值无效**	
	16#0002	值不是有效数字	更正该值；重新启动命令。
	16#000A	值小于或等于 0。	
16#840F		**运动控制指令的“JerkTime”参数值无效**	
	16#0002	值不是有效数字	更正该值；重新启动命令。
	16#000A	值小于或等于 0。	
16#8410		**运动控制指令的“Parameter”参数值无效**	

ErrorID	ErrorInfo	说明	解决方法
	16#0002	值不是有效数字	更正该值；重新启动命令。
	16#000B	地址无效	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
	16#0028	VARIANT 指针“参数”和“值”的数据类型不匹配。	使用适当的数据类型；重新启动命令。
	16#0029	VARIANT 指针“参数”并未指向工艺对象的数据块。	更正 VARIANT 指针；重新启动命令。
	16#002A	无法读取 VARIANT 指针“参数”值。	更正 VARIANT 指针；重新启动命令。
	16#002B	无法写入 VARIANT 指针“参数”值。	更正 VARIANT 指针或值；重新启动命令。
	16#002C	轴未禁用。	禁用轴；重新启动命令。
16#8411		运动控制指令的“Value”参数值无效	
	16#0002	值不是有效数字。	更正该值；重新启动命令。
	16#0005	该值超出值范围（大于 1E+12）。	
	16#0006	该值超出值范围（小于 1E+12）。	

轴的组态错误

ErrorID	ErrorInfo	说明	解决方法
16#8600		脉冲发生器 (PTO) 的参数分配无效	
	16#000B	地址无效	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
	16#0014	所选硬件由另一应用程序使用	
16#8601		高速计数器 (HSC) 的参数分配无效	

ErrorID	ErrorInfo	说明	解决方法
	16#000B	地址无效	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
	16#0014	所选硬件由另一应用程序使用	
16#8602		“使能输出”(Enable output) 的参数分配无效	
	16#000B	地址无效	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
16#8603		“输入就绪”(Ready input) 的参数分配无效	
	16#000B	地址无效	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
16#8604		“电机每转的脉冲数”(Pulses per motor revolution) 值无效	
	16#000A	值小于或等于零	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
16#8605		“每转的距离”(Distance per revolution) 值无效	
	16#0002	值不是有效数字	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
	16#0005	该值超出值范围（大于 1E+12）	
	16#000A	值小于或等于零	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8606		“启动/停止速度”(Start / stop velocity) 值无效	
	16#0002	值不是有效数字	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
	16#0003	值高于硬件上限值	
	16#0004	值小于硬件下限值	
	16#0007	启动/停止速度大于最大速度	

ErrorID	ErrorInfo	说明	解决方法
16#8607		“最大速度”(maximum velocity) 值无效	
	16#0002	值不是有效数字	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
	16#0003	值高于硬件上限值	
	16#0004	值小于硬件下限值	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8608		“加速度”(Acceleration) 值无效	
	16#0002	值不是有效数字	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
	16#0003	值高于硬件上限值	
	16#0004	值小于硬件下限值	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8609		“减速度”(Deceleration) 值无效	

ErrorID	ErrorInfo	说明	解决方法
	16#0002	值不是有效数字	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
	16#0003	值高于硬件上限值	
	16#0004	值小于硬件下限值	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#860A		“急停减速度”(Emergency stop deceleration) 值无效	
	16#0002	值不是有效数字	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
	16#0003	值高于硬件上限值	
	16#0004	值小于硬件下限值	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#860B		下限软限位开关的位置值无效	
	16#0002	值不是有效数字	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
	16#0005	该值超出值范围（大于 1E+12）	
	16#0006	该值超出值范围（小于 1E+12）	
	16#0030	下限软限位开关的位置值大于上限软限位开关的位置值	
16#860C		上限软限位开关的位置值无效	

ErrorID	ErrorInfo	说明	解决方法
	16#0002	值不是有效数字	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
	16#0005	该值超出值范围（大于 1E+12）	
	16#0006	该值超出值范围（小于 1E+12）	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#860D		下限硬限位开关的地址无效	
	16#000B	地址无效	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
	16#000C	下降沿的地址无效	
	16#000D	上升沿的地址无效	
16#860E		上限硬限位开关的地址无效	
	16#000B	地址无效	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
	16#000C	下降沿的地址无效	
	16#000D	上升沿的地址无效	
16#860F		“起始位置偏移值”(home position offset) 无效	

ErrorID	ErrorInfo	说明	解决方法
	16#0002	值不是有效数字	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
	16#0005	该值超出值范围（大于 1E+12）	
	16#0006	该值超出值范围（小于 1E+12）	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8610		“逼近速度”(approach velocity) 值无效	
	16#0002	值不是有效数字	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
	16#0008	速度大于最大速度	
	16#0009	速度小于启动/停止速度	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8611		“归位速度”(Homing velocity) 值无效	
	16#0002	值不是有效数字	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
	16#0008	速度大于最大速度	
	16#0009	速度小于启动/停止速度	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。

ErrorID	ErrorInfo	说明	解决方法
16#8612		**归位开关的地址无效**	
	16#000B	地址无效	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
	16#000C	下降沿的地址无效	
	16#000D	上升沿的地址无效	
16#8613		**虽然未组态硬限位开关，但还是在主动归位过程中，激活了在该硬限位开关处反向**	
	16#0001	-	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8614		**“冲击”(Jerk) 值无效**	
	16#0002	值不是有效数字	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
	16#001F	值大于最大冲击值	
	16#0020	值小于最小冲击值	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8615		**“测量单位”的值无效**	
	16#0011	所选值无效	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
16#8616		**归位开关的地址无效（自版本 V4 起的被动归位）**	

ErrorID	ErrorInfo	说明	解决方法
	16#0011	所选值无效	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8617		变量 <轴名称>.Sensor.Sensor[1].ActiveHoming.Mode 的值无效	
	16#0011	所选值无效 （有效值：2 = 通过数字量输入归位）	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8618		变量 <轴名称>.Sensor.Sensor[1].PassiveHoming.Mode 的值无效	
	16#0011	所选值无效 （有效值：2 = 通过数字量输入归位）	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8619		变量 <轴名称>.Actor.Type 的值无效	
	16#0011	所选值无效 （有效值：2 = 通过脉冲接口连接）	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#861A		“允许的旋转方向”(Permitted direction of rotation) 值无效	
	16#0011	所选值无效	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
	16#002D	禁用方向输出时不允许“双向”(Both directions) 运动	
16#861B		负载齿轮因数错误	
	16#0031	数值无效。	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。

ErrorID	ErrorInfo	说明	解决方法
16#861C		通过增量编码器归位的数据组合非法	
	16#0031	数值无效。	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#861D		设置的编码器安装方式无效。<轴名称>.Sensor.Sensor[1].MountingMode 中的值无效	
	16#0011	所选值无效	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#861E		测量编码器轮周长的组态无效。<轴名称>.Sensor.Sensor[1].Parameter.DistancePerRevolution 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#861F		组态的线性编码器精度错误。<轴名称>.Sensor.Sensor[1].Parameter.Resolution 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8620		设置的 Gn_XIST1 高精度值无效。<轴名称>.Sensor.Sensor[1].Parameter.FineResolutionXist1 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8621		在 <轴名称>.Sensor.Sensor[1].Parameter.FineResolutionXist1 中设置的 Gn_XIST1 高精度值与 PROFIdrive 参数 P979 中的不一致	

ErrorID	ErrorInfo	说明	解决方法
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8622		组态日期 <轴名称>.Actor.Interface.AddressIn 或 <轴名称>.Actor.Interface.AddressOut 的值无效	
	16#0011	所选值无效	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
16#8623		变量 <轴名称>.Sensor.Sensor[1].Type 中设置的值无效。	
	16#0011	所选值无效	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
16#8624		设置的编码器系统无效。<轴名称>.Sensor.Sensor[1].System 中的值无效	
	16#0011	所选值无效	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8625		定位监控的参数错误。<轴名称>.PositioningMonitoring.MinDwellTime 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8626		定位监控的参数错误。<轴名称>.PositioningMonitoring.Window 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8627		PROFIdrive 接口组态的实际值错误。<轴名称>.Sensor.Sensor[1].Interface.AddressIn 或 <轴名称>.Sensor.Sensor[1].Interface.AddressOut 中的值无效	
	16#0011	所选值无效	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
16#8628		控制器数错误	

ErrorID	ErrorInfo	说明	解决方法
	16#0030	值的数字格式不正确，或者超出有效的数字范围	控制回路的增益或预控制值错误。 ● 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 ● 在线更正不正确的值；使用指令“MC_Reset”并视需要重新启动命令（<轴名称>.PositionControl.Kv、<轴名称>.PositionControl.Kpc）
16#8629		停止信号的限值错误。<轴名称>.StandStillSignal.VelocityThreshold 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	● 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 ● 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#862A		定位监控的参数错误。<轴名称>.PositioningMonitoring.ToleranceTime 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	● 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 ● 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#862B		PROFIBUS 参数化不一致；Ti 和 To 的总和大于一个 DP 周期	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
16#862C		停止监视的参数错误。<轴名称>.StandStillSignal.MinDwellTime 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	● 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 ● 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#862D		跟随误差监控的参数错误。<轴名称>.FollowingError.MinValue 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	● 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 ● 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。

ErrorID	ErrorInfo	说明	解决方法
16#862E		组态日期 <轴名称>.Modulo.Length 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#862F		组态日期 <轴名称>.Modulo.StartValue 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8630		组态日期 <轴名称>.Actor.DriveParameter.ReferenceSpeed 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8631		设置的 Gn_XIST2 高精度值无效。<轴名称>.Sensor.Sensor[1].Parameter.FineResolutionXist2 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8632		可确定的编码器精度值无效。<轴名称>.Sensor.Sensor[1].Parameter.DeterminableRevolutions 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8633		指定的被动归位的归位开关逼近方向无效。<轴名称>.Sensor.Sensor[1].PassiveHoming.Direction 中的值无效	
16#8634		跟随误差监控的参数错误。<轴名称>.FollowingError.MaxValue 中的值无效	

ErrorID	ErrorInfo	说明	解决方法
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8635		跟随误差监控的参数错误。<轴名称>.FollowingError.MinVelocity 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8636		控制器因数不正确。<轴名称>.PositionControl.Kpc 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
16#8637		组态日期 <轴名称>.Sensor.Sensor[1].Interface.Type 中的值无效	
	16#0011	所选值无效	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
16#8638		组态日期 <轴名称>.Sensor.Sensor[1].Interface.HSC 中的值无效	
	16#0011	所选值无效	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
16#8639		驱动器上有错误	
	16#0049	设备的组态错误	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
	16#004A	工艺需要一个更小的伺服时钟。	内部系统错误。 检查项目的一致性并将项目重新加载到控制器中。
	16#004B	斜坡上升过程中未初始化设备驱动程序。	要启用工艺对象，必须完成执行器驱动程序的初始化。 请稍后再执行该命令。
16#863A		与驱动器的通信错误	

ErrorID	ErrorInfo	说明	解决方法
	16#004C	设备的组态错误	• 连接合适的设备。 • 检查设备 (I/O)。 • 比较 HW Config 中的组态和工艺对象。
	16#004D	设备驱动程序需要一个更小的伺服时钟。	• 连接合适的设备。 • 检查设备 (I/O)。 • 比较 HW Config 中的组态和工艺对象。
	16#004E	与设备进行内部通信时发生错误	检查项目的一致性并将项目重新加载到控制器中。
16#863B		**编码器中的错误**	
	16#0049	设备的组态错误	将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。
	16#004A	工艺需要一个更小的伺服时钟。	内部系统错误。 检查项目的一致性并将项目重新加载到控制器中。
	16#004B	斜坡上升过程中未初始化设备驱动程序。	要启用工艺对象，必须完成执行器驱动程序的初始化。 请稍后再执行该命令。
16#863C		**与编码器的通信错误**	
	16#004C	设备的组态错误	• 连接合适的设备。 • 检查设备 (I/O)。 • 比较 HW Config 中的组态和工艺对象。
	16#004D	设备驱动程序需要一个更小的伺服时钟。	• 连接合适的设备。 • 检查设备 (I/O)。 • 比较 HW Config 中的组态和工艺对象。
	16#004E	与设备进行内部通信时发生错误	检查项目的一致性并将项目重新加载到控制器中。
16#863D		**与设备（驱动器或编码器）的通信错误**	

ErrorID	ErrorInfo	说明	解决方法
	16#0030	值的数字格式不正确，或者超出有效的数字范围	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值；使用指令“MC_Reset”确认错误并视需要重新启动命令。
	16#0055	请求的逻辑地址无效。	• 连接合适的设备。 • 检查设备 (I/O)。 • 检查 HW Config 中的拓扑组态。 • 比较 HW Config 中的组态和工艺对象。
	16#0056	请求的逻辑输出地址无效。	
	16#0057	请求的逻辑输出地址无效。	
16#863E		变量“ControlPanel.Input.TimeOut”的值 无效（轴控制面板）	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	更正工艺对象 <轴名称>.ControlPanel.Input.TimeOut 的变量值。 以毫秒为单位指定该值。
16#863F		组态日期 <轴名称>.Actor.DriveParameter.MaxSpeed 中的值无效	
	16#0030	值的数字格式不正确，或者超出有效的数字范围	将驱动器和工艺对象组态中的参考值更正为 Actuator.MaxSpeed/2。 对于模拟驱动器接口，将驱动器和工艺对象组态中的参考值更正为 Actuator.MaxSpeed/1.17。

数据调整错误

ErrorID	ErrorInfo	说明	解决方法	错误反应
16#8640		调整执行器组态时出错		
	16#0030	值的数字格式不正确，或者超出有效的数字范围。	重新启动	调整错误
	16#0059	该设备未分配给任何 SINAMICS 驱动器装置或不支持调整所需的服务。	重新启动	调整错误
	16#005A	因资源不足取消调整。	重新启动	调整错误

ErrorID	ErrorInfo	说明	解决方法	错误反应
	16#005B	只有在设备直连 IO 区域时才能进行调整。	重新启动	调整错误
	16#005C	最大速度 (p1082)：参数不存在，或者参数值无法读取或超出允许的限值范围。硬件指示错误，取消读参数操作。	重新启动	调整错误
	16#005D	最大力矩/力 (p1520)：参数不存在，或者参数值无法读取或超出允许的限值范围。硬件指示错误，取消读参数操作。	重新启动	调整错误
	16#005E	最大力矩/力 (p1521)：参数不存在，或者参数值无法读取或超出允许的限值范围。硬件指示错误，取消读参数操作。	重新启动	调整错误
	16#005F	高精度力矩/力限制 (p1544)：参数不存在，或者参数值无法读取或超出允许的限值范围。硬件指示错误，取消读参数操作。	重新启动	调整错误
	16#0060	基本速度/额定速度 (p2000)：参数不存在，或者参数值无法读取或超出允许的限值范围。硬件指示错误，取消读参数操作。	重新启动	调整错误
	16#0061	额定力矩/额定力 (p2003)：参数不存在，或者参数值无法读取或超出允许的限值范围。硬件指示错误，取消读参数操作。	重新启动	调整错误
16#8041		**调整编码器组态时出错**		
	16#0030	值的数字格式不正确，或者超出有效的数字范围。	重新启动	调整错误
	16#005A	因资源不足取消调整。	重新启动	调整错误
	16#005B	只有在设备直连 IO 区域时才能进行调整。	重新启动	调整错误

ErrorID	ErrorInfo	说明	解决方法	错误反应
	16#0059	该设备未分配给任何 SINAMICS 驱动器装置或不支持调整所需的服务。	重新启动	调整错误
	16#0062	编码器系统 (r0979[1/11].0)：参数不存在，或者参数值无法读取或超出允许的限值范围。硬件指示错误，取消读参数操作。	重新启动	调整错误
	16#0063	编码器精度 (r0979[2/12])：参数不存在，或者参数值无法读取或超出允许的限值范围。硬件指示错误，取消读参数操作。	重新启动	调整错误
	16#0064	编码器精度 Gx_XIST1 (r0979[3/13])：参数不存在，或者参数值无法读取或超出允许的限值范围。硬件指示错误，取消读参数操作。	重新启动	调整错误
	16#0065	编码器精度 Gx_XIST2 (r0979[4/14])：参数不存在，或者参数值无法读取或超出允许的限值范围。硬件指示错误，取消读参数操作。	重新启动	调整错误
	16#0066	可解析的编码器转数 (r0979[5/15])：参数不存在，或者参数值无法读取或超出允许的限值范围。硬件指示错误，取消读参数操作。	重新启动	调整错误
16#8642		**在内部调整组态**		
	16#0067	1：Actor.MaxSpeed 值无效（Actor.MaxSpeed 大于 2*Actor.ReferenceSpeed）；补救措施：例如，在驱动器中，设置 P2000 = P1082。	复位	组态错误
16#8643		**TO 和驱动器组态不一致**		
	16#0068	组态的消息帧类型与设备的消息帧类型（P922 或 P2079）不兼容。	复位	组态错误

ErrorID	ErrorInfo	说明	解决方法	错误反应
	16#0069	力矩分辨率不兼容。	复位	组态错误
	16#006A	主应用周期的基本周期时间与伺服时钟周期的周期时间不相等。	复位	组态错误
	16#006B	工艺对象的处理周期与驱动器的应用周期不相等。	复位	组态错误
	16#006C	在驱动器中，设置用于线性电机的功能模块。	复位	组态错误
16#8644		**TO 和编码器组态不一致**		
	16#0068	组态的消息帧类型与设备的消息帧类型（P922 或 P2079）不兼容。	复位	组态错误
	16#006A	主应用周期的基本周期时间与伺服时钟周期的周期时间不相等。	复位	组态错误
	16#006B	工艺对象的处理周期与驱动器的应用周期不相等。	复位	组态错误
	16#006D	驱动器上的编码器不属于绝对值编码器 (P979)。	复位	组态错误
16#8645		**通过设置的驱动器和轴参数，无法达到最大速度**		
	16#0001	常规	复位	组态错误

命令表的组态错误

ErrorID	ErrorInfo	说明	解决方法
16#8700		**命令表中“命令类型”(Command type) 的值无效**	
	16#0001	-	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值，并视需要重新启动命令。
16#8701		**命令表中“位置/行进路径”(Position/travel path) 的值无效**	

ErrorID	ErrorInfo	说明	解决方法
	16#0002	值不是有效数字	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值，并视需要重新启动命令。
	16#0005	该值超出值范围（大于 1E+12）	
	16#0006	该值超出值范围（小于 1E+12）	
16#8702		**命令表中“速度”(Velocity) 的值无效**	
	16#0002	值不是有效数字	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值，并视需要重新启动命令。
	16#0008	值大于组态的最大速度	
	16#0009	值小于组态的启动/停止速度	
16#8703		**命令表中“持续时间”(Duration) 的值无效**	
	16#0002	值不是有效数字	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值，并视需要重新启动命令。
	16#0021	值大于 64800 s	
	16#0022	值小于 0.001 s	
16#8704		**命令表中“下一步”(Next step) 的值无效**	
	16#0011	所选值无效	• 将无错误组态下载到控制器；使用指令“MC_Power”再次启用轴。 • 在线更正不正确的值，并视需要重新启动命令。
	16#0023	该命令不允许命令转换	

内部错误

ErrorID	ErrorInfo	说明	解决方法
16#8FFF		内部错误	
	16#F0**	-	CPU 的断电和上电 如果这样做无效，则联系客户支持。准备好以下信息： • ErrorID • ErrorInfo • 诊断缓冲区条目

通信 11

S7-1200 可实现 CPU 与编程设备、HMI 和其它 CPU 之间的多种通信。

> ⚠ 警告
>
> **如果攻击者能以物理方式访问您的网络，那么便可能读写数据。**
>
> TIA Portal、CPU 和 HMI（使用 GET/PUT 的 HMI 除外）均采用安全通信，可防止重放攻击和“中间人”攻击。 启用这种通信后，将以纯文本形式交换签名消息，这种方式允许攻击者读取数据，但可避免未经授权的数据写入操作。 TIA Portal（而非通信过程）将对受专有技术保护的块中的数据进行加密。
>
> 所有其它形式的通信（通过 PROFIBUS、PROFINET、AS-i 或其它 I/O 总线、GET/PUT、传输块 (T-block) 和通信模块 (CM) 进行的 I/O 交换）均没有安全功能。 必须通过限制物理访问来保护这些形式的通信。 如果攻击者能利用这些形式的通信以物理方式访问您的网络，那么便可能读写数据。
>
> 有关安全信息和建议，请参见 Siemens 服务与支持网站上的“工业安全操作准则 (http://www.industry.siemens.com/topics/global/en/industrial-security/Documents/operational_guidelines_industrial_security_en.pdf)”。

PROFINET

PROFINET 用于使用用户程序通过以太网与其它通信伙伴交换数据：

- 在 S7-1200 中，PROFINET 支持 16 个最多具有 256 个子模块的 IO 设备，PROFIBUS 允许使用 3 个独立的 PROFIBUS DP 主站，每个 DP 主站支持 32 个从站，每个 DP 主站最多具有 512 个模块。
- S7 通信
- 用户数据报协议 (UDP)
- ISO on TCP (RFC 1006)
- 传输控制协议 (TCP)

PROFINET IO 控制器

作为采用 PROFINET IO 的 IO 控制器，CPU 可与本地 PN 网络上或通过 PN/PN 耦合器（连接器）连接的最多 16 台 PN 设备通信。 有关详细信息，请参见 PROFIBUS 和 PROFINET International (PI) (www.profinet.com)。

PROFIBUS

PROFIBUS 用于使用用户程序通过 PROFIBUS 网络与其它通信伙伴交换数据：

- 借助 CM 1242-5，CPU 作为 PROFIBUS DP 从站运行。
- 借助 CM 1243-5，CPU 作为 1 类 PROFIBUS DP 主站运行。
- PROFIBUS DP 从站、PROFIBUS DP 主站和 AS-i（左侧 3 个通信模块）以及 PROFINET 均采用单独的通信网络，不会相互制约。

AS-i

通过 S7-1200 CM 1243-2 AS-i 主站可将 AS-i 网络连接到 S7-1200 CPU。

CPU 至 CPU S7 通信

您可以创建与伙伴站的通信连接并使用 GET 和 PUT 指令与 S7 CPU 进行通信。

TeleService 通信

在通过 GPRS 的 TeleService 中，安装了 STEP 7 的工程师站通过 GSM 网络 Internet 和与具有 CP 1242-7 的 SIMATIC S7-1200 站进行通信。该连接通过用作中介并连接到 Internet 的远程控制服务器运行。

IO-Link

利用 S7-1200 SM 1278 4xIO-Link 主站，可将 IO-Link 设备与 S7-1200 CPU 相连。

11.1 异步通信连接

通信服务概述

此 CPU 支持以下通信服务：

通信服务	功能	使用 PROFIBUS DP		使用以太网
		CM 1243-5 DP 主站模块	CM 1242-5 DP 从站模块	
PG 通信	调试、测试、诊断	√	×	√
HMI 通信	操作员控制和监视	√	×	√
S7 通信	使用已组态连接交换数据	√	×	√
路由 PG 功能	例如，跨网络边界进行测试和诊断	×	×	×
PROFIBUS DP	在主站与从站之间交换数据	√	√	×
PROFINET IO	I/O 控制器和 I/O 设备之间的数据交换	×	×	√
Web 服务器	诊断	×	×	√
SNMP [1]（简单网络管理协议）	用于网络诊断和参数化的标准协议	×	×	√
S7 路由	通过路由表，即使设备位于不同 S7 子网上，通信伙伴也可以和每个设备进行通信。	×	×	√
通过 TCP/IP 的开放式通信	使用 TCP/IP 协议通过工业以太网交换数据（使用可装载 FB）	×	×	√
通过 ISO on TCP 的开放式通信	使用 ISO on TCP 协议通过工业以太网交换数据（使用可加载 FB）	×	×	√
通过 UDP 的开放式通信	使用 UDP 协议通过工业以太网交换数据（使用可装载 FB）	×	×	√

[1] CPU 支持无陷阱 SNMP V 1。

可用连接

对于 PROFINET 和 PROFIBUS，CPU 最多可支持下列数量的并发异步通信连接。分配给每个类别的最大连接资源数为固定值；您无法更改这些值。但可组态六个“可用自由连接”以按照应用要求增加任意类别的连接数。

连接资源

	站资源			模块资源
	预留		动态 !	PLC_2 [CPU 1217C DC/DC/...
最大资源数：		62	6	68
	最大	已组态	已组态	已组态
PG 通信：	4	-	-	-
HMI 通信：	12	0	0	0
S7 通信：	8	0	0	0
开放式用户通信：	8	0	0	0
Web 通信：	30	-	-	-
其它通信:	-	-	0	0
使用的总资源：		0	0	0
可用资源：		62	6	68

说明

在添加 CM/CP 模块时，S7-1200 通信连接的总数不增加。

说明

OPC UA 连接

OPC UA 连接占用六个“可用自由连接”中的资源。OPC UA 连接在“其它通信：”(Other communication:) 列和“动态”(Dynamic) 列中可见。确保应用程序有足够的可用连接。

根据已分配的连接资源，每个设备的可用连接数如下：

	编程 终端 (PG)	人机 界面 (HMI)	GET/PUT 客户端/服务器	开放式用户 通信	Web 浏览器
连接资源的最大数量	4 （保证支持 1 个 PG 设备）	12 （保证支持 4 个 HMI 设备）	8	8	30 （保证支持 3 个 Web 浏览器）

例如，CPU 有四个可用的 PG 连接资源。根据当前使用的 PG 功能，该 PG 实际可能使用其可用连接资源中的一个、两个、三个或四个。可以始终使用一个 PG。

另一个示例为 HMI 数，如下图所示。HMI 具有 12 个可用连接资源。根据拥有的 HMI 类型或型号以及使用的 HMI 功能，每个 HMI 实际可能使用其可用连接资源中的一个、两个或三

个。考虑到正在使用的可用连接资源数，可以同时使用四个以上的 HMI。但是，要始终确保至少有四个 HMI。HMI 可利用其可用连接资源（每个一个，共三个）实现下列功能：

- 读取
- 写入
- 报警和诊断

这仅仅是个示例。使用连接的实际数量会根据 HMI 类型和版本而不同。

示例	HMI 1	HMI 2	HMI 3	HMI 4	HMI 5	总的可用连接资源
使用的连接资源	2	2	2	3	3	12

说明

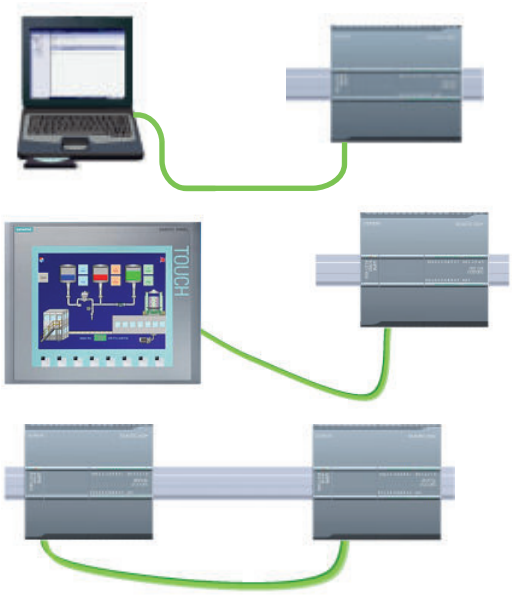
Web 服务器 (HTTP) 连接：CPU 提供用于多个 Web 浏览器的连接。此 CPU 可同时支持的浏览器数取决于给定 Web 浏览器请求/使用的连接数。

说明

开放式用户通信、S7 连接、HMI、编程设备以及 Web 服务器 (HTTP) 通信连接可以根据当前使用的功能使用多个连接资源。

11.2 PROFINET

CPU 可使用标准 TCP 通信协议与其它 CPU、编程设备、HMI 设备和非 Siemens 设备通信。



CPU 连接到编程设备

CPU 连接到 HMI

CPU 连接到另一个 CPU

以太网交换

CPU 1211C、1212C、和 1214C 拥有独立以太网接口并不包含集成以太网交换机。编程设备或 HMI 与 CPU 之间的直接连接不需要以太网交换机。不过，含有两个以上的 CPU 或 HMI 设备的网络需要以太网交换机。

① CPU 1215C

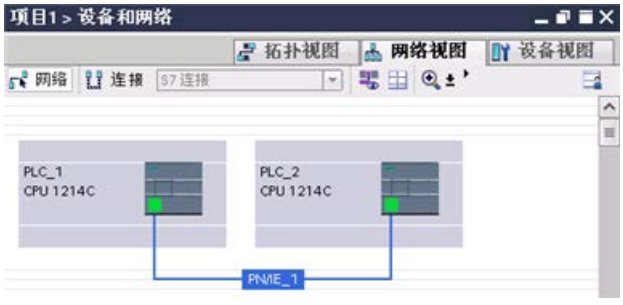
② CSM1277 以太网交换机

CPU 1215C 和 CPU 1217C 具有内置的双端口以太网交换机。您可使用具有 CPU 1215C 和另两个 S7-1200 CPU 的网络。也可以使用安装在机架上的 CSM1277 4 端口以太网交换机来连接多个 CPU 和 HMI 设备。

11.2.1 创建网络连接

使用设备配置的“网络视图”(Network view) 在项目中的各个设备之间创建网络连接。 创建网络连接之后，使用巡视窗口的“属性”(Properties) 选项卡组态网络的参数。

表格 11-1 创建网络连接

操作	结果
选择“网络视图”(Network view) 以显示要连接的设备。	
选择一个设备上的端口，然后将连接拖到第二个设备上的端口处。	
释放鼠标按钮以创建网络连接。	

11.2.2 组态本地/伙伴连接路径

本地/伙伴（远程）连接定义两个通信伙伴的逻辑分配以建立通信服务。连接定义了以下内容：

- 涉及的通信伙伴（一个主动，一个被动）
- 连接类型（例如，PLC、HMI 或设备连接）
- 连接路径

通信伙伴执行指令来设置和建立通信连接。用户使用参数指定主动和被动通信端点伙伴。设置并建立连接后，CPU 会自动保持和监视该连接。

如果连接终止（例如，因断线），主动伙伴将尝试重新建立组态的连接。不必再次执行通信指令。

连接路径

将 TSEND_C、TRCV_C 或 TCON 指令插入用户程序后，只要选中指令的任意部分，巡视窗口都会显示连接的属性。在通信指令“属性”(Properties) 的“组态”(Configuration) 选项卡中指定通信参数。

表格 11-2　组态连接路径（使用指令的属性）

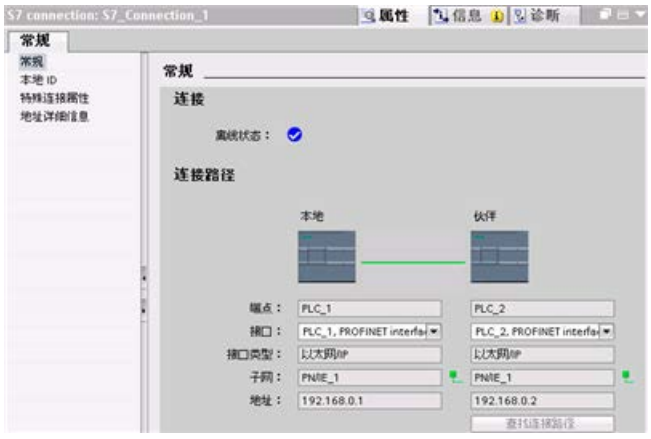
TCP、ISO-on-TCP 和 UDP	连接属性
对于 TCP、ISO-on-TCP 和 UDP 以太网协议，使用指令（TSEND_C、TRCV_C 或 TCON）的“属性”(Properties) 组态“本地/伙伴”连接。 右图显示了 ISO-on-TCP 连接的“组态”(Configuration) 选项卡中的“连接属性”(Connection properties)。	

说明

组态其中一个 CPU 的连接属性时，STEP 7 允许您选择伙伴 CPU 中的特定连接 DB（如果存在），或为伙伴 CPU 创建连接 DB。必须已为该项目创建伙伴 CPU，且不能是“未指定的”CPU。

还必须将 TSEND_C、TRCV_C 或 TCON 指令插入伙伴 CPU 的用户程序中。插入指令时，应选择由组态创建的连接 DB。

表格 11-3　为 S7 通信组态连接路径（设备组态）

S7 通信（GET 和 PUT）	连接属性
对于 S7 通信，请使用网络的“设备和网络”编辑器组态本地/伙伴连接。可以单击“突出显示：连接”(Highlighted: Connection) 按钮访问“属性”(Properties)。 “常规”(General) 选项卡中提供有多个属性： ● “常规”(General)（已显示） ● “本地 ID”(Local ID) ● “特殊连接属性”(Special connection properties) ● “地址详细信息”(Address details)（已显示）	 

要获取更多信息以及可用通信指令的列表，请参见“PROFINET”部分的“协议” (页 828)，或“S7 通信”部分的“创建 S7 连接” (页 1007)。

表格 11-4　多 CPU 连接的参数

参数		定义
地址		分配的 IP 地址
常规	端点	分配给伙伴（接收）CPU 的名称
	接口	分配给接口的名称
	子网	分配给子网的名称
	接口类型	*仅 S7 通信*：接口类型
	连接类型	以太网协议的类型
	连接 ID	ID 号
	连接数据	本地和伙伴 CPU 的数据存储位置
	建立主动连接	用于选择本地或伙伴 CPU 作为主动连接方的单选按钮
地址详细信息	端点	*仅 S7 通信*：分配给伙伴（接收）CPU 的名称
	机架/插槽	*仅 S7 通信*：机架和插槽位置：
	连接资源	*仅 S7 通信*：组态与 S7-300 或 S7-400 CPU 通信的 S7 连接时使用的 TSAP 组件
	端口（十进制）	TCP 和 UPD：十进制格式的伙伴 CPU 端口
	TSAP[1] 和子网 ID：	ISO on TCP (RFC 1006) 和 S7 通信：ASCII 格式和十六进制格式的本地和伙伴 CPU TSAP

[1] 组态与 S7-1200 CPU 的 ISO-on-TCP 连接时，请在被动通信伙伴的 TSAP 扩展中仅使用 ASCII 字符。

传输服务访问点 (TSAP)

通过 TSAP、ISO on TCP 协议和 S7 通信，允许有多个连接访问单个 IP 地址。TSAP 可唯一标识连接到同一个 IP 地址的这些通信端点连接。

在“连接参数”(Connection Parameters) 对话框的“地址详细信息”(Address Details) 部分，定义要使用的 TSAP。在“本地 TSAP”(Local TSAP) 域中输入 CPU 中连接的 TSAP。在“伙伴 TSAP”(Partner TSAP) 域下输入为伙伴 CPU 中的连接分配的 TSAP。

端口号

使用 TCP 和 UDP 协议时，本地（主动）连接 CPU 的连接参数组态必须指定远程伙伴（被动）连接 CPU 的 IP 地址和端口号。

在“连接参数”(Connection Parameters) 对话框的“地址详细信息”(Address Details) 部分，定义要使用的端口。在“本地端口”(Local Port) 域中输入 CPU 中连接的端口。在“伙伴端口”(Partner Port) 域下输入为伙伴 CPU 中的连接分配的端口。

11.2.3 分配 Internet 协议 (IP) 地址

11.2.3.1 为编程设备和网络设备分配 IP 地址

如果编程设备使用连接到工厂 LAN 的板载适配器卡，那么编程设备和 CPU 必须存在同一子网上。设备的 IP 地址和子网掩码的组合即可指定设备的子网。相关帮助，请联系您的当地网络管理员。

网络 ID 是 IP 地址的前三个八位位组（例如，**211.154.184**.16）。该网络 ID 是用户 IP 网络的唯一标识。子网掩码的值通常为 **255.255.255.0**。然而，由于用户的计算机处于工厂 LAN 中，子网掩码可能有不同的值（例如，**255.255.254.0**）以设置唯一的子网。在子网掩码以 AND 逻辑操作方式与设备 IP 地址组合时，可定义 IP 子网的边界。

说明

在万维网环境下，编程设备、网络设备和 IP 路由器可与全世界通信，但必须分配唯一的 IP 地址以避免与其它网络用户冲突。请联系公司 IT 部门熟悉工厂网络的人员分配 IP 地址。

⚠ 警告

通过 Web 服务器对 CPU 进行未经授权的访问

未经授权访问 CPU 或将 PLC 变量更改为无效值可能会中断过程操作并可能导致死亡、严重人身伤害和/或财产损失。

启用 Web 服务器可让授权用户执行工作模式更改、写入 PLC 数据以及进行固件更新，Siemens 建议遵照以下安全实践：

- 仅使用 HTTPS 协议启用对 Web 服务器的访问。
- 使用可靠的密码对 Web 服务器用户 ID 进行密码保护 (页 1029)。强密码在长度上至少为十个字符，可以是字母、数字和特殊字符的组合，不能是可在字典上找到的词，并且不能是可从个人信息推断出的名字或标识符。保管好密码并经常更改密码。
- 不要扩展“所有人”(Everybody) 用户的默认最低权限。
- 对程序逻辑中的变量执行错误检查和范围检查，因为 Web 页面用户可将 PLC 变量更改为无效值。

说明

当不想将 CPU 连入公司 LAN 时，非常适合使用次级网络适配器卡。在首次测试或调试测试期间，这种安排尤其实用。

使用桌面上的“网上邻居”(My Network Places) 分配或检查编程设备的 IP 地址

要分配或检查编程设备的 IP 地址，请按以下步骤操作：

1. 打开“启动”(Start) 菜单的“控制面板”(Control Panel)。
2. 打开“网络和共享中心”(Network and Sharing Center) 并选择连接到 CPU 的网络适配器的“本地连接”(Local Area Connection)
3. 单击“本地连接状态”(Local Area Connection Status) 对话框上的“属性”(Properties)。
4. 在“本地连接属性”(Local Area Connection Properties) 对话框中，为“此连接使用下列项目：”(This connection uses the following items:) 字段选择“网络协议版本 4 (TCP/IPv4)” (Internet Protocol Version 4 (TCP/IPv4))。
5. 单击“属性”(Properties) 按钮。
6. 选择“自动获取 IP 地址”(Obtain an IP address automatically) 或选择“使用以下 IP 地址”(Use the following IP address) 以输入静态 IP 地址。
7. 如果选择“使用以下 IP 地址”(Use the following IP address)，请设置 IP 地址和子网掩码：
   - 将 IP 地址设置为与 CPU 使用相同的网络 ID 和子网。例如，如果 CPU IP 地址是 **192.168.0**.1，则可以将 IP 地址设置为 **192.168.0**.200。
   - 选择子网掩码 **255.255.255**.0。
   - 保持默认网关空白。

现在可连接到 CPU。

说明

网络接口卡和 CPU 必须在相同子网上以使 STEP 7 可找到并与此 CPU 进行通信。

请咨询 IT 支持人员来帮助您设置网络组态，进而连接到 S7-1200 CPU。

11.2.3.2 检查编程设备的 IP 地址

可以使用以下菜单选项来检查编程设备的 MAC 地址和 IP 地址：

1. 在“项目树”(Project tree) 中展开“在线访问”(Online access)。
2. 右键单击所需网络并选择“属性”(Properties)。
3. 在网络对话框中展开“组态”(Configurations) 并选择“工业以太网”(Industrial Ethernet)。

将显示编程设备的 MAC 地址和 IP 地址。

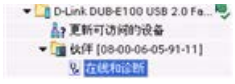
11.2.3.3 在线给 CPU 分配 IP 地址

可以在线为网络设备分配 IP 地址。这在进行初始设备配置时尤其有用。

1.在“项目树”(Project tree) 中，确认 CPU 不具有预组态的 IP 地址。展开“在线访问”(Online access) 显示设备所在网络的适配器卡，然后双击“更新可访问的设备”(Update accessible devices)。

如果 STEP 7 显示 MAC 地址，而非 IP 地址，表示未分配 IP 地址。

2.在所需可访问设备下双击“在线和诊断”(Online & diagnostics)。



3.“在线与诊断”(Online & diagnostics) 对话框中，选择“功能 > 分配 IP 地址”(Functions > Assign IP address)。

4.在“IP 地址”(IP address) 区域输入新 IP 地址，然后单击“分配 IP 地址”(Assign IP address) 按钮。

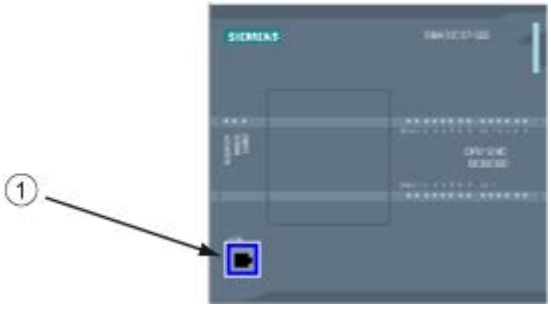
5.在“项目树”(Project tree) 中，确保 STEP 7 已将新 IP 地址分配给 CPU。

双击“更新可访问的设备”(Update accessible devices)，显示组态的 IP 地址。

11.2.3.4 为项目中的 CPU 组态 IP 地址

组态 PROFINET 接口

要为 PROFINET 接口组态参数，请选择 CPU 上的绿色 PROFINET 框。巡视窗口中的“属性”(Properties) 选项卡会显示 PROFINET 端口。



① PROFINET 端口

组态 IP 地址

以太网 (MAC) 地址：在 PROFINET 网络中，制造商会为每个设备都分配一个“介质访问控制”地址（MAC 地址）以进行标识。MAC 地址由六组数字组成，每组两个十六进制数，这些数字用连字符 (-) 或冒号 (:) 分隔并按传输顺序排列（例如 01-23-45-67-89-AB 或 01:23:45:67:89:AB）。

IP 地址：每个设备也都必须具有一个 Internet 协议 (IP) 地址。该地址使设备可以在更加复杂的路由网络中传送数据。

每个 IP 地址分为四段，每段占 8 位，并以点分十进制格式表示（例如，211.154.184.16）。IP 地址的第一部分用于表示网络 ID（您正位于什么网络中？），地址的第二部分表示主机 ID（对于网络中的每个设备都是唯一的）。IP 地址 192.168.x.y 是一个标准名称，视为未在 Internet 上路由的专用网的一部分。

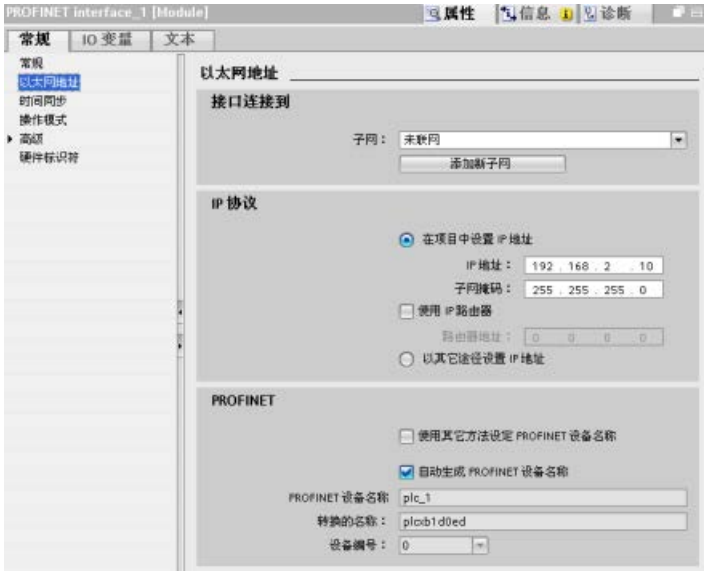
子网掩码：子网是已连接的网络设备的逻辑分组。在局域网 (LAN, Local Area Network) 中，子网中的节点往往彼此之间的物理位置相对接近。掩码（称为子网掩码或网络掩码）定义 IP 子网的边界。

子网掩码 255.255.255.0 通常适用于小型本地网络。这就意味着此网络中的所有 IP 地址的前 3 个八位位组应该是相同的，该网络中的各个设备由最后一个八位位组（8 位域）来标识。举例来说，在小型本地网络中，为设备分配子网掩码 255.255.255.0 和 IP 地址 192.168.2.0 到 192.168.2.255。

不同子网间的唯一连接通过路由器实现。如果使用子网，则必须部署 IP 路由器。

IP 路由器：路由器是 LAN 之间的链接。通过使用路由器，LAN 中的计算机可向其它任何网络发送消息，这些网络可能还隐含着其它 LAN。如果数据的目的地不在 LAN 内，路由器会将数据转发给可将数据传送到其目的地的另一个网络或网络组。

路由器依靠 IP 地址来传送和接收数据包。



IP 地址属性：在“属性”(Properties) 窗口中，选择“以太网地址”(Ethernet addresses) 组态条目。STEP 7 会显示以太网地址组态对话框，该对话框可将软件项目与接收该项目的 CPU 的 IP 地址相关联。

表格 11-5　IP 地址的参数

参数		说明
子网	连接到设备的子网的名称。单击“添加新子网”(Add new subnet) 按钮以创建新的子网。默认为“未连接”(Not connected)。可以有两种连接类型： ● 默认情况下“未连接”(Not connected) 提供本地连接。 ● 网络具有两个或多个设备时，需要子网。	
IP 协议	IP 地址	为 CPU 分配的 IP 地址
	子网掩码	分配的子网掩码
	使用 IP 路由器	单击该复选框以指示 IP 路由器的使用
	路由器地址	为路由器分配的 IP 地址（如果适用）

说明

下载项目时会组态所有 IP 地址。如果 CPU 不具有预组态的 IP 地址，必须将该项目与目标设备的 MAC 地址相关联。如果 CPU 连接到网络中的路由器，则也必须输入路由器的 IP 地址。

“使用其它方法设置 IP 地址”(Set IP address using a different method) 单选按钮允许用户在线更改 IP 地址，或在下载程序之后通过“T_CONFIG (页 919)”指令进行更改。这种 IP 地址分配方法仅适用于 CPU。

⚠ 警告

下载具有“使用其它方法设置 IP 地址”(Set IP address using different method) 的硬件配置

下载启用了“使用其它方法设置 IP 地址”(Set IP address using a different method) 选项的硬件配置后，不能将 CPU 操作模式从 RUN 切换到 STOP，或者从 STOP 切换到 RUN。

在这些情况下，用户设备将继续运行，如果未采取适当的预防措施，则可能导致意外的机器或过程操作，从而导致死亡、严重人身伤害或财产损失。

确保先设置 CPU IP 地址，然后在实际的自动化环境中使用 CPU。这可以通过将 STEP 7 编程包、SIMATIC 自动化工具或连接的 HMI 设备与 T_CONFIG 指令配合使用来完成。

⚠ 警告

PROFINET 网络可能停止的情况

在线更改 CPU 的 IP 地址或通过用户程序更改时，可能会出现 PROFINET 网络停止的情况。

如果 CPU 的 IP 地址更改为子网外的 IP 地址，PROFINET 网络将失去通信，并会停止所有数据交换。可将用户设备组态为在这些情况下保持运行。如果未采取适当的预防措施，丢失 PROFINET 通信可能会导致意外的机器或过程操作，从而导致死亡、严重人身伤害或财产损失。

如果必须手动更改 IP 地址，则应确保新 IP 地址在子网范围内。

组态 PROFINET 端口

在默认情况下，CPU 会为自动协商组态 PROFINET 接口的端口。要使自动协商正常运行，必须将两个站都组态到自动协商。如果其中一个站为固定组态（例如，在 100 Mbps 处为全双工）且另一个站被设置为自动协商，那么自动协商将失效，导致使用半双工进行操作。

要克服自动协商的这个限制，可使用 S7-1200 提供的选项禁用自动协商。在禁用自动协商时，S7-1200 会自动为全双工操作在 100 Mbps 处组态。

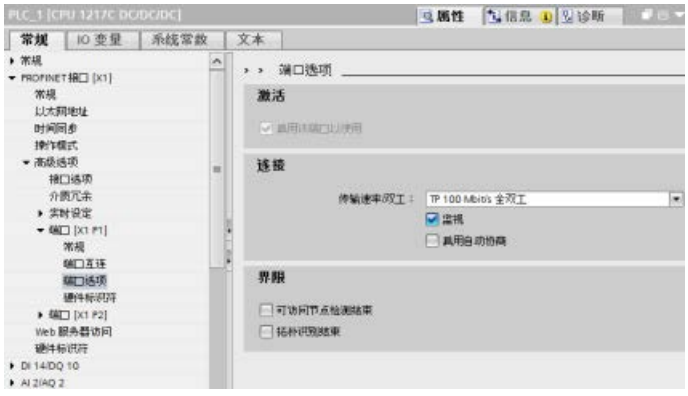
可为每个端口设置固定传输率和双工：

1. 选择“高级”(Advanced) 选项以及需要组态的端口。然后，选择“端口”(Port) 选项。
2. 在“连接”(Connection)，“传输率”(Transmission rate)/双工字段，选择以下一个选项：
   - 自动：CPU 和对等设备可通过自动协商决定端口的传输率和双工。
   - TP 100 Mbps 全双工：如果禁用自动协商，端口在 100 Mbps 的半双工状态下运行。如果启用自动协商，此端口可在 100 Mbps 的全双工状态下运行或在另一个 CPU 与对等设备自动协商的传输率/双工状态下运行（如果选择“监视”(Monitor)，此对等设备会在诊断缓冲区中显示如下信息）。

3. 监视：在选择复选框时，如果在此端口出现以下情况，系统将会在诊断缓冲区中显示消息：
   – 不能在端口中建立链接
   – 建立链接失败
   – 请选择“TP 100 Mbps 全双工”(TP 100 Mbps full-duplex) 作为传输率/双工，CPU 会使用自动协商建立链接，且协商传输率不等于 100 Mbps 或协商双工等于半双工。
4. 启用自动协商：在 100 Mbps，一旦将传输率/双工字段设置到全双工，便可以禁用自动协商。清除 “启用自动协商”(Enable autonegotiation) 复选框来禁用自动协商。

说明

如果不禁用自动协商，CPU 和对等设备会协商此端口的传输率和双工。



11.2.4 测试 PROFINET 网络

在完成组态后，下载项目 (页 204)到 CPU 中。 下载项目时会组态所有 IP 地址。

在线为设备分配 IP 地址

S7-1200 CPU 不具有预组态的 IP 地址。 必须手动为 CPU 分配 IP 地址：

- 要在线为设备分配 IP 地址，请参见“设备组态： 在线为 CPU 分配 IP 地址” (页 817)了解此逐步操作过程。
- 要在项目中分配 IP 地址，必须在设备配置中组态 IP 地址，保存配置并将其下载到 PLC。 请参见“设备组态： 为项目中的 CPU 组态 IP 地址” (页 818)，获取更多信息。

使用“扩展的下载到设备”(Extended download to device) 对话框测试所连接的网络设备

S7-1200 CPU“下载到设备”(Download to device) 功能及其“扩展的下载到设备”(Extended download to device) 对话框可以显示所有可访问的网络设备，以及是否为所有设备都分配了唯一的 IP 地址。 要显示全部可访问和可用的设备以及为其分配的 MAC 地址或 IP 地址，请选中“显示所有可访问设备”(Show all accessible devices) 复选框。

如果所需网络设备不在此列表中，则说明由于某种原因而中断了与该设备的通信。 必须检查设备和网络是否有硬件和/或组态错误。

11.2.5 查找 CPU 上的以太网 (MAC) 地址

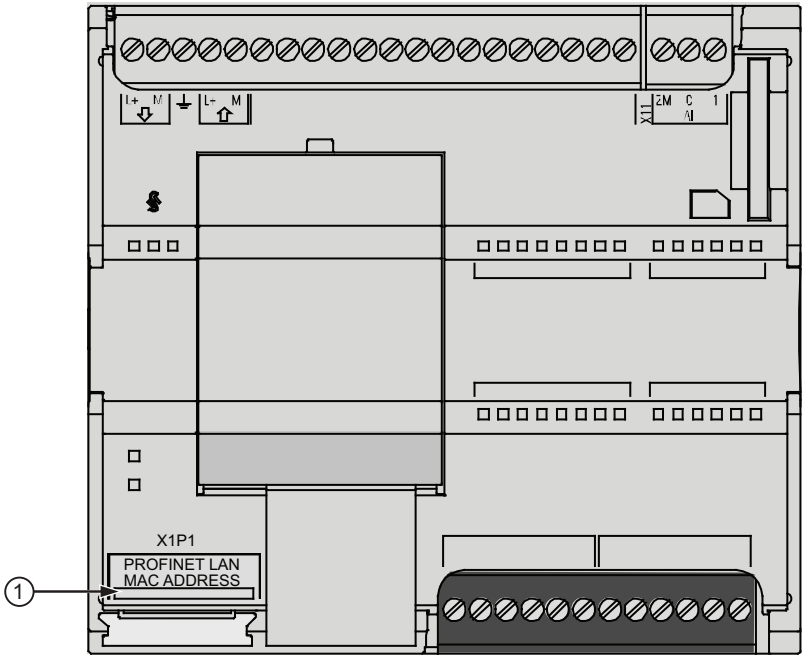
在 PROFINET 网络中，“介质访问控制”地址（MAC 地址）是制造商为了标识网络接口而分配的标识符。 MAC 地址通常用制造商的注册标识号进行编码。

外观良好、按标准 (IEEE 802.3) 格式印制的 MAC 地址由六组数字组成，每组两个十六进制数，这些数字组用连字符 (-) 或冒号 (:) 分隔并按传输顺序排列（例如 01-23-45-67-89-ab 或 01:23:45:67:89:ab）。

说明

每个 CPU 在出厂时都已装载了一个永久、唯一的 MAC 地址。 您无法更改 CPU 的 MAC 地址。

MAC 地址印在 CPU 正面左下角位置。 请注意，必须提起下面的门才能看到 MAC 地址信息。



① MAC 地址

最初，CPU 没有 IP 地址，只有工厂安装的 MAC 地址。 PROFINET 通信要求为所有设备都分配唯一的 IP 地址。

可以使用 CPU“下载到设备” (Download to device) 功能及其“扩展的下载到设备”(Extended download to device) 对话框，显示所有可访问的网络设备以确保已经为所有设备分配了唯一的 IP 地址。 此对话框可显示所有可访问和可用的设备以及所分配的 MAC 地址或 IP 地址。 在识别缺少所需唯一 IP 地址的设备时，MAC 地址就十分重要。

11.2.6 组态网络时间协议 (NTP) 同步

⚠ 警告

存在攻击者通过网络时间协议 (Network Time Protocol, NTP) 同步访问用户网络的风险

如果攻击者能通过网络时间协议 (NTP) 同步访问用户网络，那么便可能通过改变 CPU 系统时间来中断过程控制。过程控制中断可能造成死亡、重伤或财产损失。

默认情况下，S7-1200 CPU 禁用 NTP 客户端功能。启用 NTP 功能时，只有用户组态的 IP 地址可以用作 NTP 服务器。必须组态 NTP 功能以允许通过远程服务器进行 CPU 系统时间更正。

S7-1200 CPU 支持“日时钟”中断和时钟指令，这两个指令均依赖于精确的 CPU 系统时间。如果组态 NTP 并接受从服务器进行时间同步，那么必须确保服务器是可靠来源。否则会导致安全漏洞，从而使未知用户能够通过改变 CPU 系统时间来破坏对过程的控制。

有关安全信息和建议，请参见 Siemens 服务与支持网站上的“工业安全操作准则 (http://www.industry.siemens.com/topics/global/en/industrial-security/Documents/operational_guidelines_industrial_security_en.pdf)”。

网络时间协议 (NTP, Network Time Protocol) 被广泛用于使计算机系统的时钟与 Internet 时间服务器同步。在 NTP 模式中，CPU 按固定时间间隔将日时钟查询（客户机模式中）发送到子网 (LAN) 的 NTP 服务器。根据服务器的响应，来计算最可靠、最准确的时间，并同步工作站的日时钟。

这种模式的优点是可以跨子网同步时间。

最多可以组态四个 NTP 服务器的 IP 地址。更新时间间隔定义各时间查询之间的时间间隔（单位为秒）。时间间隔的值范围在 10 秒到一天之间。

在 NTP 模式下，服务器通常会传送 UTC（协调世界时）；与 GMT（格林威治标准时间）相对应。

在 CPU 设备组态的 “属性”(Properties) 窗口中，选择 “时间同步”(Time synchronization) 组态条目。STEP 7 显示 “时间同步”(Time synchronization) 组态对话框：

说明

下载项目时 CPU 会收到所有 IP 地址。

表格 11-6　时间同步参数

参数	定义
通过 NTP 服务器启用时间同步	选择复选框以通过 NTP 服务器启用时间同步。
服务器 1 (Server 1)	为网络时间服务器 1 分配的 IP 地址
服务器 2 (Server 2)	为网络时间服务器 2 分配的 IP 地址
服务器 3 (Server 3)	为网络时间服务器 3 分配的 IP 地址
服务器 4 (Server 4)	为网络时间服务器 4 分配的 IP 地址
时间同步更新间隔	时间间隔值（秒）
CPU 同步设备的模块。	选择复选框，使 CP 时钟与 CPU 时钟同步。

11.2.7 PROFINET 设备启动时间、命名和地址分配

PROFINET IO 可以延长系统的启动时间（可组态超时）。 设备较多和设备较慢都会影响切换到 RUN 模式的时间。

在 V4.0 及更高版本中，S7-1200 PROFINET 网络上最多支持 16 个 PROFINET IO 设备。

每个站（或 IO 设备）会在启动时单独启动，这会影响总的 CPU 启动时间。 如果将可组态的超时值设定过低，就可能没有足够的总 CPU 启动时间让所有站完成启动。 若发生这种情况，会导致假的站错误。

在 “启动”(Startup) 下的“CPU 属性”(CPU Properties) 中，可以找到 “分布式 I/O 的参数分配时间”(Parameter assignment time for distributed I/O)（超时）。 默认的可组态超时为 60,000 ms（1 分钟）；用户可以组态该时间。

STEP 7 中的 PROFINET 设备命名及寻址

所有 PROFINET 设备必须都具有设备名称和 IP 地址。 使用 STEP 7 定义设备名称并组态 IP 地址。 使用 PROFINET DCP（Discovery and Configuration Protocol，发现和组态协议）将设备名称下载到 IO 设备。

系统启动时的 PROFINET 地址分配

控制器会向网络广播设备名称，设备会以其 MAC 地址进行响应。 然后，控制器会使用 PROFINET DCP 协议为设备分配 IP 地址：

- 如果 MAC 地址具有已组态的 IP 地址，则相应的站执行启动。
- 如果 MAC 地址不具有组态的 IP 地址，则 STEP 7 会分配项目中组态的地址，之后，相应的站会执行启动。
- 如果这一过程出现问题，则会产生站错误，且不会进行启动。 这种情况会导致超出可组态的超时值。

11.2.8 开放式用户通信

11.2.8.1 协议

CPU 的集成 PROFINET 端口支持多种以太网网络上的通信标准：

- 传输控制协议 (TCP)
- ISO on TCP (RFC 1006)
- 用户数据报协议 (UDP)

表格 11-7 协议以及用于每种协议的通信指令

协议	用途示例	在接收区输入数据	通信指令	寻址类型
TCP	CPU 与 CPU 通信 帧传输	特殊模式	仅 TRCV_C 和 TRCV	将端口号分配给本地（主动）和伙伴（被动）设备
		指定长度的数据接收	TSEND_C、TRCV_C、TCON、TDISCON、TSEND 和 TRCV	
ISO on TCP	CPU 与 CPU 通信 消息的分割和重组	特殊模式	仅 TRCV_C 和 TRCV	将 TSAP 分配给本地（主动）和伙伴（被动）设备
		协议控制	TSEND_C、TRCV_C、TCON、TDISCON、TSEND 和 TRCV	
UDP	CPU 与 CPU 通信 用户程序通信	用户数据报协议	TUSEND 和 TURCV	将端口号分配给本地（主动）和伙伴（被动）设备，但不是专用连接
S7 通信	CPU 与 CPU 通信 从 CPU 读取数据/向 CPU 写入数据	指定长度的数据传输和接收	GET 和 PUT	将 TSAP 分配给本地（主动）和伙伴（被动）设备
PROFINET IO	CPU 与 PROFINET IO 设备通信	指定长度的数据传输和接收	内置	内置

参见

新功能 (页 37)

11.2.8.2 TCP 和 ISO on TCP

传输控制协议 (TCP) 是由 RFC 793 描述的一种标准协议： 传输控制协议。 TCP 的主要用途是在过程对之间提供可靠、安全的连接服务。 该协议有以下特点：

- 由于它与硬件紧密相关，因此它是一种高效的通信协议
- 它适合用于中等大小或较大的数据量（最多 8192 字节）
- 它为应用带来了更多的便利，特别是对于错误恢复、流控制和可靠性。
- 它是一种面向连接的协议
- 它可以非常灵活地用于只支持 TCP 的第三方系统
- 有路由功能
- 只能应用静态数据长度。
- 消息会被确认。
- 使用端口号对应用程序寻址。
- 大多数用户应用协议（例如 TELNET 和 FTP）都使用 TCP。
- 由于使用 SEND/RECEIVE 编程接口的缘故，需要编程来进行数据管理。

基于传输控制协议 (TCP) 的国际标准组织 (ISO) (RFC 1006) (ISO on TCP) 是一种能够将 ISO 应用移植到 TCP/IP 网络的机制。 该协议有以下特点：

- 它是与硬件关系紧密的高效通信协议
- 它适合用于中等大小或较大的数据量（最多 8192 字节）
- 与 TCP 相比，它的消息提供了数据结束标识符并且它是面向消息的。
- 具有路由功能；可用于 WAN
- 可用于实现动态数据长度。
- 由于使用 SEND/RECEIVE 编程接口的缘故，需要编程来进行数据管理。

通过传输服务访问点 (TSAP, Transport Service Access Point)，TCP 协议允许有多个连接访问单个 IP 地址（最多 64K 个连接）。 借助 RFC 1006，TSAP 可唯一标识连接到同一个 IP 地址的这些通信端点连接。

11.2.8.3 通信服务和使用的端口号

S7-1200 CPU 支持下表中列出的协议。 对于每种协议，CPU 指定了地址参数、各自的通信层以及通信角色和通信方向。

通过这些信息，可以将自动化系统的安全保护措施与所用的协议进行匹配（例如防火墙）。 仅以太网或 PROFINET 网络才有信息安全措施。 PROFIBUS 没有任何安全措施，因此，该表不包括任何 PROFIBUS 协议。

该表列出了 CPU 所使用的不同协议层和协议：

协议	端口号	(2) 链路层 (4) 传输层	功能	说明
PROFINET 协议				
DCP （发现和组态协议）	不相关	(2) 以太网 II 和 IEEE 802.1Q 及以太网类型 0x8892 (PROFINET)	可访问设备 PROFINET 发现和配置	PROFINET 采用 DCP 协议发现设备并提供基本设置。 DCP 使用特定的组播 MAC 地址： xx-xx-xx-01-0E-CF, xx-xx-xx = 组织唯一标识符
LLDP （Link Layer Discovery Protocol，链路层发现协议）	不相关	(2) 以太网 II 和 IEEE 802.1Q 及以太网类型 0x88CC (PROFINET)	PROFINET 链路层发现协议	PROFINET 使用 LLDP 来发现和管理 PROFINET 设备间的邻近关系。 LLDP 使用特定的组播 MAC 地址： 01-80-C2-00-00-0E

11.2.8.4 特殊模式

通常，TCP 和 ISO-on-TCP 接收指定长度的数据包（1 到 8192 字节）。 但 TRCV_C 和 TRCV 通信指令还提供“特殊”通信模式，可接收可变长度的数据包（1 到 1472 字节）。

说明

如果将数据存储在“优化”DB（仅符号访问）中，则只能接收数据类型为 Byte、Char、USInt 和 SInt 的数组中的数据。

要针对特殊模式组态 TRCV_C 或 TRCV 指令，请置位 ADHOC 指令输入参数。

如果在特殊模式下并未频繁调用 TRCV_C 或 TRCV 指令，则可在一次调用中接收多个数据包。 例如： 如果要通过一次调用接收五个 100 字节的数据包，TCP 可将这五个数据包打包成一个 500 字节的数据包一起传送，而 ISO-on-TCP 则可将该数据包重组成五个 100 字节的数据包。

11.2.8.5 开放式用户通信指令的连接 ID

将 TSEND_C、TRCV_C 或 TCON PROFINET 指令插入到用户程序中时，STEP 7 会创建一个背景数据块，以组态设备之间的通信通道（或连接）。使用指令的“属性”(Properties) (页 812) 组态连接的参数。这些参数中有该连接的连接 ID。

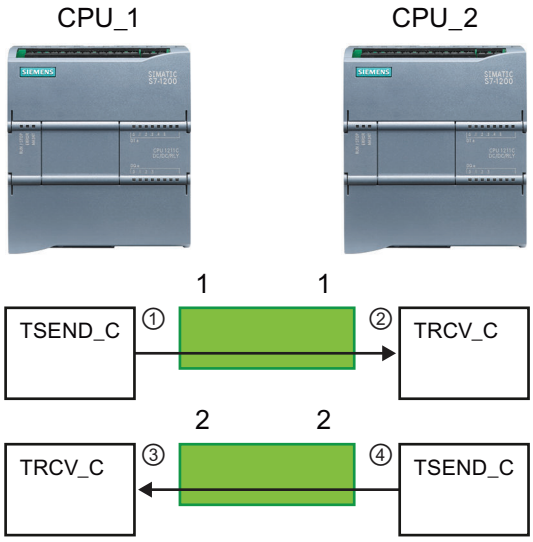
- 连接 ID 对于 CPU 必须是唯一的。创建的每个连接必须具有不同的 DB 和连接 ID。
- 本地 CPU 和伙伴 CPU 都可以对同一连接使用相同的连接 ID 编号，但连接 ID 编号不需要匹配。连接 ID 编号只与各 CPU 用户程序中的 PROFINET 指令相关。
- CPU 的连接 ID 可以使用任何数字。但是，从“1”开始按顺序组态连接 ID 可以很容易地跟踪特定 CPU 使用的连接数。

  说明

  用户程序中的每个 TSEND_C、TRCV_C 或 TCON 指令都创建一个新连接。为每个连接使用正确的连接 ID 非常重要。

以下示例显示了两个 CPU 之间的通信，这两个 CPU 使用 2 个单独的连接来发送和接收数据。

- CPU_1 中的 TSEND_C 指令通过第一个连接（CPU_1 和 CPU_2 上的“连接 ID 1”）与 CPU_2 中的 TRCV_C 链接。
- CPU_1 中的 TRCV_C 指令通过第二个连接（CPU_1 和 CPU_2 上的“连接 ID 2”）与 CPU_2 中的 TSEND_C 链接。



① CPU_1 上的 TSEND_C 创建一个连接并为该连接分配一个连接 ID（CPU_1 的连接 ID 1）。

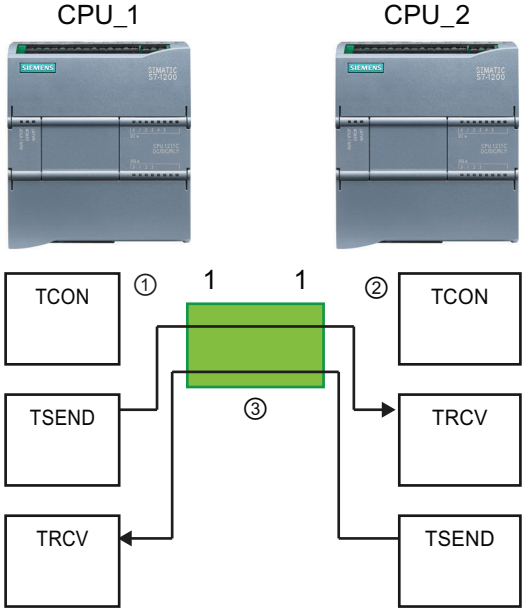
② CPU_2 上的 TRCV_C 为 CPU_2 创建连接并分配连接 ID（CPU_2 的连接 ID 1）。

③ CPU_1 上的 TRCV_C 为 CPU_1 创建第二个连接并为该连接分配不同的连接 ID（CPU_1 的连接 ID 2）。

④ CPU_2 上的 TSEND_C 创建第二个连接并为该连接分配不同的连接 ID（CPU_2 的连接 ID 2）。

以下示例显示了两个 CPU 之间的通信，这两个 CPU 使用 1 个连接来发送和接收数据。

- 每个 CPU 都使用 TCON 指令来组态两个 CPU 之间的连接。
- CPU_1 中的 TSEND 指令通过由 CPU_1 中的 TCON 指令组态的连接 ID（“连接 ID 1”）链接到 CPU_2 中的 TRCV 指令。CPU_2 中的 TRCV 指令通过由 CPU_2 中的 TCON 指令组态的连接 ID（“连接 ID 1”）链接到 CPU_1 中的 TSEND 指令。
- CPU_2 中的 TSEND 指令通过由 CPU_2 中的 TCON 指令组态的连接 ID（“连接 ID 1”）链接到 CPU_1 中的 TRCV 指令。CPU_1 中的 TRCV 指令通过由 CPU_1 中的 TCON 指令组态的连接 ID（“连接 ID 1”）链接到 CPU_2 中的 TSEND 指令。



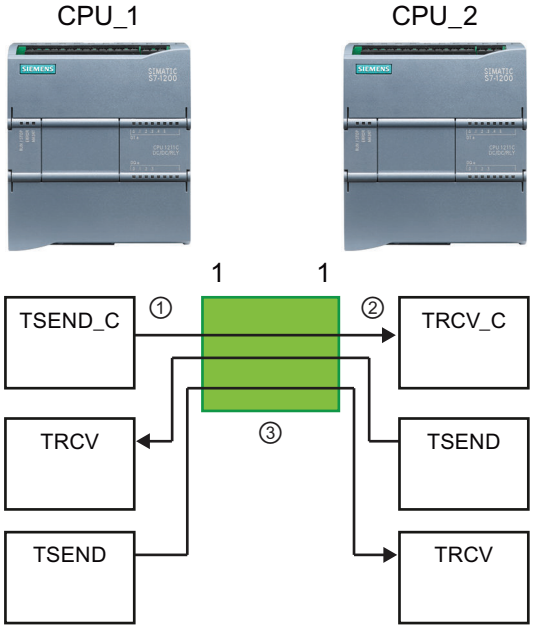
① CPU_1 上的 TCON 创建一个连接并在 CPU_1 上为该连接分配连接 ID (ID=1)。

② CPU_2 上的 TCON 创建一个连接并在 CPU_2 上为该连接分配连接 ID (ID=1)。

③ CPU_1 上的 TSEND 和 TRCV 使用 CPU_1 上的 TCON 创建的连接 ID (ID=1)。

CPU_2 上的 TSEND 和 TRCV 使用 CPU_2 上的 TCON 创建的连接 ID (ID=1)。

如以下示例所示，还可以使用单个 TSEND 和 TRCV 指令通过由 TSEND_C 或 TRCV_C 指令创建的连接进行通信。TSEND 和 TRCV 指令本身不会创建新连接，因此必须使用由 TSEND_C、TRCV_C 或 TCON 指令创建的 DB 和连接 ID。



① CPU_1 上的 TSEND_C 创建一个连接并为该连接分配连接 ID (ID=1)。

② CPU_2 上的 TRCV_C 创建一个连接并在 CPU_2 上为该连接分配连接 ID (ID=1)。

③ CPU_1 上的 TSEND 和 TRCV 使用 CPU_1 上的 TSEND_C 创建的连接 ID (ID=1)。

CPU_2 上的 TSEND 和 TRCV 使用 CPU_2 上的 TRCV_C 创建的连接 ID (ID=1)。

11.2.8.6 PROFINET 连接的参数

TSEND_C、TRCV_C 和 TCON 指令需要与连接相关的参数，才能连接到伙伴设备。TCON_Param 结构为 TCP、ISO-on-TCP 和 UDP 协议分配这些参数。通常使用指令的 “属性”(Properties) 中的 “组态”(Configuration) (页 812) 选项卡来指定这些参数。如果无法访问 “组态”(Configuration) 选项卡，则必须在指令参数中提供 TCON_Param 结构。

在 V4.1 或更高版本中，TCON_IP_V4 结构会为 TCP 协议分配参数，TCON_IP_RFC 结构会为 ISO-on-TCP 协议分配参数。

在 V4.3 或更高版本中，TCON_IP_V4_SEC 结构会为 TCP 协议分配更多参数。为了在两个 S7-1200 CPU 之间建立安全的 TCP 通信，必须创建系统数据类型为 TCON_IP_V4_SEC 的数据块或在各个 CPU 中创建数据块，执行参数分配并直接在指令中进行调用。TCON、TSEND_C 和 TRCV_C 指令支持 TCON_IP_V4_SEC 系统数据类型。

在 V4.4 或更高版本中，使用 TCON_QDN 和 TCON_QDN_SEC 结构通过完全限定域名组态 TCP 和 UDP 的通信连接，并使用 TCON_QDN_SEC 结构通过具有安全通信的完全限定域名组态 TCP 的通信连接。

TCON_Param

表格 11-8　连接描述的结构 (TCON_Param)

Byte	参数和数据类型		说明
0 到 1	block_length	UInt	长度：64 个字节（固定）
2 至 3	id	CONN_OUC (Word)	对该连接的引用：值范围：1（默认值）到 4095。在 ID 下，为 TSEND_C、TRCV_C 或 TCON 指令指定该参数的值。
4	connection_type	USInt	连接类型： ● 17：TCP（默认） ● 18：ISO-on-TCP ● 19：UDP
5	active_est	Bool	连接类型的 ID： ● TCP 和 ISO-on-TC： – FALSE：被动连接 – TRUE：主动连接（默认） ● UDP：FALSE
6	local_device_id	USInt	本地 PROFINET 或工业以太网接口的 ID：1（默认值）
7	local_tsap_id_len	USInt	所用 local_tsap_id 参数的长度（以字节表示）；可能值： ● TCP：0（主动，默认值）或 2（被动） ● ISO-on-TCP：2 到 16 ● UDP：2
8	rem_subnet_id_len	USInt	该参数未使用。
9	rem_staddr_len	USInt	伙伴端点地址的长度（以字节表示）： ● 0：未指定（参数 rem_staddr 不相关） ● 4（默认值）：参数 rem_staddr 中的 IP 地址有效（仅对于 TCP 和 ISO-on-TCP）
10	rem_tsap_id_len	USInt	所用 rem_tsap_id 参数的长度（以字节表示）；可能值： ● TCP：0（被动）或 2（主动，默认值） ● ISO-on-TCP：2 到 16 ● UDP：0
11	next_staddr_len	USInt	该参数未使用。

Byte	参数和数据类型		说明
12 到 27	local_tsap_id	Array [1..16] of Byte	连接的本地地址部分： ● TCP 和 ISO-on-TC：本地端口号（可能值：1 到 49151；推荐值：2000...5000）： – local_tsap_id[1] = 十六进制表示的端口号的高位字节； – local_tsap_id[2] = 十六进制表示的端口号的低位字节； – local_tsap_id[3-16] = 不相关 ● ISO-on-TCP：本地 TSAP-ID： – local_tsap_id[1] = B#16#E0; – local_tsap_id[2] = 本地端点的机架和插槽（位 0 到 4：插槽号，位 5 到 7：机架号）； – local_tsap_id[3-16] = TSAP 扩展，可选 ● UDP：该参数未使用。 注：请确保 local_tsap_id 的每个值在 CPU 中都是唯一的。
28 到 33	rem_subnet_id	Array [1..6] of USInt	该参数未使用。
34 到 39	rem_staddr	Array [1..6] of USInt	仅 TCP 和 ISO-on-TCP：伙伴端点的 IP 地址。（与被动连接不相关。）例如，IP 地址 192.168.002.003 存储在数组的下列元素中： rem_staddr[1] = 192 rem_staddr[2] = 168 rem_staddr[3] = 002 rem_staddr[4] = 003 rem_staddr[5-6] = 不相关

Byte	参数和数据类型		说明
40 到 55	rem_tsap_id	Array [1..16] of Byte	连接的伙伴地址部分 ● TCP：伙伴端口号。范围：1 到 49151；推荐值：2000 到 5000）： – rem_tsap_id[1] = 十六进制表示的端口号的高位字节 – rem_tsap_id[2] = 十六进制表示的端口号的低位字节； – rem_tsap_id[3-16] = 不相关 ● ISO-on-TCP：伙伴 TSAP-ID： – rem_tsap_id[1] = B#16#E0 – rem_tsap_id[2] = 伙伴端点的机架和插槽（位 0 到 4：插槽号，位 5 到 7：机架号） – rem_tsap_id[3-16] = TSAP 扩展，可选 ● UDP：该参数未使用。
56 到 61	next_staddr	Array [1..6] of Byte	该参数未使用。
62 到 63	spare	Word	保留：W#16#0000

TCON_IP_V4

表格 11-9　连接描述的结构 (TCON_IP_V4)：与 TCP 一起使用

Byte	参数和数据类型		说明
0 到 1	InterfaceId	HW_ANY	IE 接口子模块的硬件标识符
2 至 3	ID	CONN_OUC (Word)	对该连接的引用：值范围：1（默认值）到 4095。在 ID 下，为 TSEND_C、TRCV_C 或 TCON 指令指定该参数的值。
4	ConnectionType	Byte	连接类型： ● 11：TCP/IP（默认） ● 17：TCP/IP（为了兼容老系统，包含该连接类型。推荐使用“11：TCP/IP（默认）”。） ● 19：UDP
5	ActiveEstablished	Bool	主动/被动建立连接： ● TRUE：主动连接（默认） ● FALSE：被动连接
	V4 IP 地址		
6	ADDR[1]	Byte	八位位组 1

Byte	参数和数据类型		说明
7	ADDR[1]	Byte	八位位组 2
8	ADDR[1]	Byte	八位位组 3
9	ADDR[1]	Byte	八位位组 4
10 到 11	RemotePort	UInt	远程 UDP/TCP 端口号
12 至 13	LocalPort	UInt	本地 UDP/TCP 端口号

TCON_IP_V4_SEC

表格 11-10 连接描述的结构 (TCON_IP_V4_SEC)：与 TCP 一起使用

Byte	参数和数据类型		说明
0 至 15	ConnPara	TCON_IP_v4	连接参数的 SDT 有关 interface_id 的信息： • 如果将 interface_id 的值保留为预设值 0，则 CPU 的操作系统将评估远程 IP 地址和本地现有的 IP 路由，然后指定用于建立 OUC 安全连接的 CPU 工业以太网接口。此时，系统始终会将诊断数据分配给 CPU 的第一个工业以太网接口。 • 如果将 CPU 或 CP 的工业以太网接口的硬件标识符指定为 interface_id，则可使用相关的工业以太网接口建立 OUC 安全连接。
16	ActivateSecure-Conn	Bool	激活该连接的安全通信 如果该参数的值为 FALSE（默认值），则将忽略后续的安全参数。即，该连接为非安全连接。此时，可建立一个非安全的 TCP 或 UDP 连接。
17	TLSServerReq-ClientCert	Bool	仅用于服务器端：从 TLS 客户端请求 X.509-V3 证书。 FALSE（默认值）
18 至 19	ExtTLSCapabilities	Word	• 第 0 位：仅用于客户端。置位意味着客户端将验证服务器 X.509-V3 证书中证书主体的替代名称 (subjectAlternateName)，从而对该服务器的身份进行检查。在建立连接后对证书进行检查。16#0（默认值） • 位 1 到 15：预留，用于后期升级

Byte	参数和数据类型		说明
20 至 23	TLSServerCertRef	UDInt	• 服务器端：自身 X.509-V3 证书的 ID • 客户端：X.509-V3 证书（通常为 CA 证书）的 ID，TLS 客户端使用该 ID 验证 TLS 服务器的身份。如果该参数为 0，则 TLS 客户端将使用客户端证书存储区中当前加载的所有 (CA) 证书对服务器的身份进行验证。0（默认值）
24 到 27	TLSClientCertRef	UDInt	• 客户端：自身 X.509-V3 证书的 ID • 服务器端：X.509-V3 证书（或 X.509-V3 证书组）的 ID，TLS 服务器使用它验证 TLS 客户端。如果该参数为 0，则 TLS 服务器将使用服务器证书存储区中当前加载的所有 (CA) 证书对客户端的身份进行验证。0（默认值）

TCON、TSEND_C 和 TRCV_C 指令的背景数据块中的 CONNECT 连接参数包含对所用数据块的引用。

说明

可以建立基于 IPv4 的非安全 TCP 或 UDP 连接。

此外，可使用 SDT TCON_IP_V4_SEC 组态基于 IPv4 的非安全 TCP 或 UDP 连接。

TCON_IP_RFC

表格 11-11 连接描述的结构 (TCON_IP_RFC)：与 ISO on TCP 一起使用

Byte	参数和数据类型		说明
0 到 1	InterfaceId	HW_ANY	IE 接口子模块的硬件标识符
2 至 3	ID	CONN_OUC (Word)	对该连接的引用：值范围：1（默认值）到 4095。在 ID 下，为 TSEND_C、TRCV_C 或 TCON 指令指定该参数的值。
4	ConnectionType	Byte	连接类型： • 12：ISO-on-TCP（默认） • 17：ISO-on-TCP （为了兼容早期系统，包含该连接类型。推荐使用“12：ISO-on-TCP（默认）”。）
5	ActiveEstablished	Bool	主动/被动建立连接： • TRUE：主动连接（默认） • FALSE：被动连接
6 至 7	备用		未使用
	V4 IP 地址		

Byte	参数和数据类型		说明
8	ADDR[1]	Byte	八位位组 1
9	ADDR[1]	Byte	八位位组 2
10	ADDR[1]	Byte	八位位组 3
11	ADDR[1]	Byte	八位位组 4
	远程传输选择器		
12 至 13	TSelLength	UInt	TSelector 的长度
14 到 45	TSel	array [1..32] of Byte	TSAP 名称的字符数组
	本地传输选择器		
46 到 47	TSelLength	UInt	TSelector 的长度
48 到 79	TSel	array [1..32] of Byte	TSAP 名称的字符数组

TCON_QDN

表格 11-12 基于 TCON_QDN 的连接描述结构

Byte	参数和数据类型		说明
0 到 1	InterfaceId	HW_ANY	S7-1200-CPU（自固件 V4.4 起）： ● 已插入 CP 的 InterfaceId： – 基于 S7-1200（自固件 V3.2 起）的 CP – 基于 ET 200SP（自固件 V2. 起）的 CP V4.4 之前版本的 S7-1200-CPU： ● 参数无关
2 至 3	ID	CONN_O UC	引用该连接（值范围：1 到 4095）。对于 TCON 指令，在 ID 中指定该参数的值。
4	ConnectionType	BYTE	连接类型： ● 11：TCP（十进制 11 = 十六进制 0x0B） ● 19：UDP（十进制 19 = 十六进制 0x13）
5	ActiveEstablished	BOOL	连接建立类型的标识符： ● FALSE：被动连接建立 ● TRUE：主动连接建立

Byte	参数和数据类型		说明
6 到 261	RemoteQDN	Array of STRING [1..254]	伙伴端点的完全限定域名，必须以“.”结尾。 请注意，在 SIMATIC 网络中，该名称（含句点）不得超过 254 个字符。
262 到 263	RemotePort	UINT	远程连接伙伴的端口地址
264 到 265	LocalPort	UINT	本地连接伙伴的端口地址

TCON_QDN_SEC

表格 11-13 基于 TCON_QDN_SEC 的连接描述结构

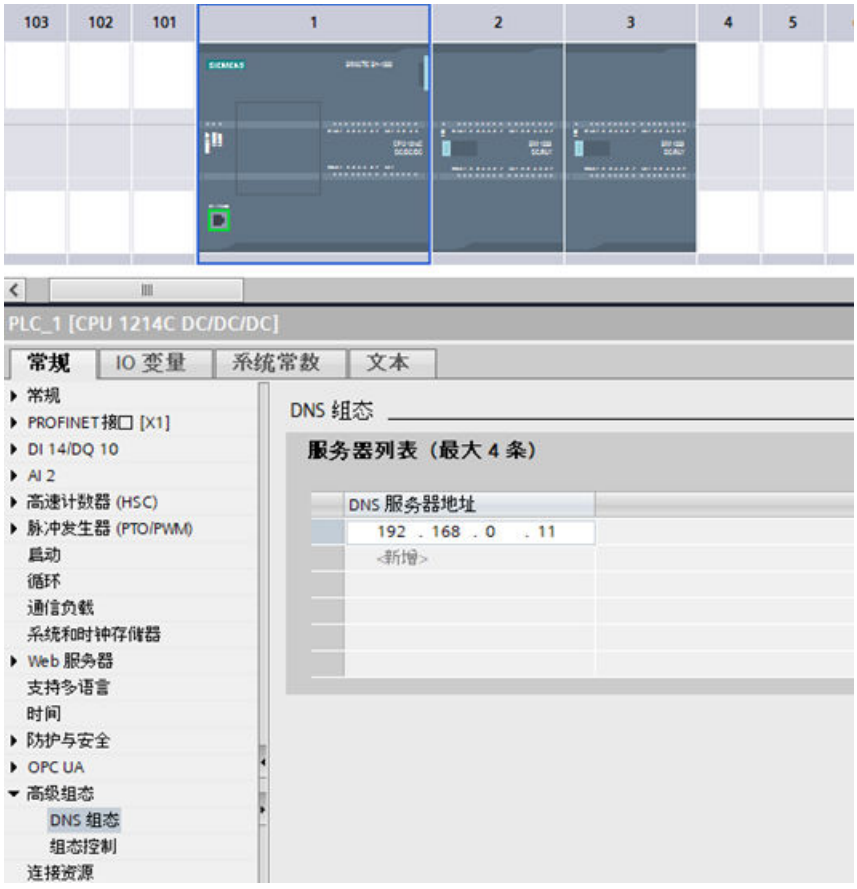
Byte	参数和数据类型		说明
0 到 271	ConnPara	TCON_QDN	连接参数
272	ActivateSecureConn	BOOL	激活该连接的安全通信。 如果该参数的值为 FALSE，则将忽略后面的安全参数。此时，可建立一个非安全的 TCP 或 UDP 连接。
273	TLSServerReqClientCert	BOOL	仅用于服务器端：TLS 客户端需要具有 X.509-V3 证书
274 到 275	ExtTLSCapabilities	WORD	• 第 0 位：仅用于客户端。置位位表示客户端将验证服务器 X.509-V3 证书中的 subjectAlternate- 名称，从而对该服务器的身份进行检查。在建立连接后对证书进行检查。 • 位 1 到 15：预留，用于后期升级
276 到 279	TLSServerCertRef	UDINT	• 服务器端：自身 X.509-V3 证书的 ID • 客户端：X.509-V3 证书（通常为 CA 证书）的 ID，TLS 客户端使用该 ID 验证 TLS 服务器的身份。如果该参数为 0，则 TLS 客户端将使用客户端证书中心当前加载的所有 (CA) 证书对服务器的身份进行验证。
280 到 283	TLSClientCertRef	UDINT	• 客户端：自身 X.509-V3 证书的 ID • 服务器端：X.509-V3 证书（或 X.509-V3 证书组）的 ID，TLS 服务器使用该 ID 验证 TLS 客户端的身份。如果该参数为 0，则 TLS 服务器将使用服务器证书存储区中当前加载的所有 (CA) 证书对客户端的身份进行验证。

11.2.8.7 组态 DNS

必须组态域名系统 (DNS) 才能使用安全 OUC。网络中必须至少存在一个 DNS 服务器，并且必须为 S7-1200 CPU 组态至少一个 DNS 服务器。

按以下步骤组态 DNS 服务器：

1. 导航到 S7-1200 CPU 的设备视图。
2. 转到 “属性”(Properties) 页面和 “常规”(General) 选项卡。
3. 单击 DNS 组态以显示组态页面。
4. 在 “服务器列表”(Server list) 表格中 DNS 服务器地址下的第一行，单击“<添加新地址>”(<Add new>) 并输入 DNS 服务器的 IP 地址。



11.2.8.8 TSEND_C 和 TRCV_C 指令

从 S7-1200 CPU V4.1 或更高版本开始，与 STEP 7 V13 SP1 或更高版本一起使用时，CPU 可以扩展 TSEND_C 和 TRCV_C 指令的功能，以便使用结构符合 TCON_IP_V4 和 TCON_IP_RFC 的连接参数。

从 S7-1200 CPU V4.3 或更高版本开始，与 STEP 7 V15.1 或更高版本一起使用时，CPU 可以扩展 TSEND_C 和 TRCV_C 指令的功能，以便使用结构符合 TCON_IP_V4、TCON_IP_V4_SEC 和 TCON_IP_RFC 的连接参数。

从 S7-1200 CPU V4.4 或更高版本开始，与 STEP 7 V16 或更高版本一起使用时，CPU 可以扩展 TSEND_C 和 TRCV_C 指令的功能，以便使用结构符合 TCON_IP_V4、TCON_IP_V4_SEC、TCON_IP_RFC、TCON_QDN 和 TCON_QDN_SEC 的连接参数。

因此，S7-1200 支持两组 TSEND_C 和 TRCV_C 指令：

- 早期 TSEND_C 和 TRCV_C 指令 (页 854)：这些 TSEND_C 和 TRCV_C 指令在 S7-1200 V4.0 之前的版本中便已存在，只能使用结构符合 TCON_Param 的连接参数。
- TSEND_C 和 TRCV_C 指令 (页 843)：这些 TSEND_C 和 TRCV_C 指令具备早期指令的所有功能，而且还能够使用结构符合 TCON_IP_V4、TCON_IP_V4_SEC、TCON_IP_RFC、TCON_QDN, and TCON_QDN_SEC 的连接参数。

选择 TSEND_C 和 TRCV_C 指令的版本

在 STEP 7 中提供了两种版本的 TSEND_C 和 TRCV_C 指令：

- V2.5 和 V3.1 可用于 STEP 7 Basic/Professional V13 或更早版本。
- 版本 4.0 在 STEP 7 Basic/Professional V13 SP1 或新版本中提供。

可以从兼容性和移植便利性方面考虑，选择将相应的指令版本插入用户程序中。

不要在同一个 CPU 程序中使用不同版本的指令。

单击指令树任务卡上的图标可启用指令树的标题和列。

要更改 TSEND_C 和 TRCV_C 指令的版本，请从下拉列表中选择相应版本。可以选择一组指令或分别选择各个指令。

使用指令树将 TSEND_C 或 TRCV_C 指令放入程序时，将根据所选的 TSEND_C 或 TRCV_C 指令在项目树中创建新的 FB 或 FC 实例。在项目树的“PLC_x > 程序块 > 系统块 > 程序资源”(PLC_x > Program blocks > System blocks > Program resources) 下可看到新的 FB 或 FC 实例。

要检验程序中 TSEND_C 或 TRCV_C 指令的版本，必须检查项目树的属性而不是程序编辑器中所显示框的属性。选择项目树的 TSEND_C 或 TRCV_C FB 或 FC 实例，单击右键，选择“属性”(Properties)，然后选择“信息”(Information) 页查看 TSEND_C 或 TRCV_C 指令的版本号。

TSEND_C 和 TRCV_C（使用以太网发送和接收数据）

TSEND_C 指令兼具 TCON、TDISCON 和 TSEND 指令的功能。TRCV_C 指令兼具 TCON、TDISCON 和 TRCV 指令的功能。（有关这些指令的详细信息，请参见“TCON、TDISCON、TSEND 和 TRCV (页 862)”。）

最少可传送 (TSEND_C) 或接收 (TRCV_C) 一个字节的数据，最多 8192 字节。TSEND_C 不支持传送布尔位置的数据，TRCV_C 也不会在布尔位置中接收数据。有关使用这些指令传送数据的信息，请参见数据一致性 (页 186)部分。

说明

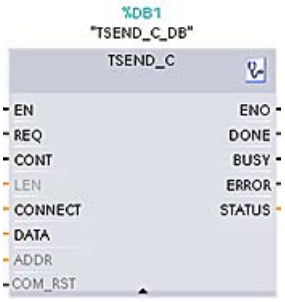
初始化通信参数

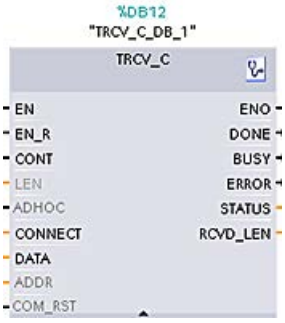
插入 TSEND_C 或 TRCV_C 指令之后，可使用该指令 (页 812)的“属性”(Properties) 来组态通信参数 (页 833)。在巡视窗口为通信伙伴输入参数时，STEP 7 会在指令的背景数据块中输入相应数据。

如果要使用多重背景数据块，必须在两个 CPU 上手动组态该 DB。

表格 11-14 TSEND_C 和 TRCV_C 指令

LAD/FBD	SCL	说明

	`"TSEND_C_DB"(`
	`req:=_bool_in_,`
	`cont:=_bool_in_,`
	`len:=_uint_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`connect:=_struct_inout_,`
	`data:=_variant_inout_,`
	`com_rst:=_bool_inout_);`
		TSEND_C 可与伙伴站建立 TCP 或 ISO on TCP 通信连接、发送数据，并且可以终止该连接。设置并建立连接后，CPU 会自动保持和监视该连接。

	`"TRCV_C_DB"(`
	`en_r:=_bool_in_,`
	`cont:=_bool_in_,`
	`len:=_uint_in_,`
	`adhoc:=_bool_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`rcvd_len=>_uint_out_,`
	`connect:=_struct_inout_,`
	`data:=_variant_inout_,`
	`com_rst:=_bool_inout_);`
		TRCV_C 可与伙伴 CPU 建立 TCP 或 ISO on TCP 通信连接，可接收数据，并且可以终止该连接。设置并建立连接后，CPU 会自动保持和监视该连接。

[1] STEP 7 会在插入指令时自动创建 DB。

表格 11-15 TSEND_C 和 TRCV_C 参数的数据类型

参数和类型		数据类型	说明
REQ (TSEND_C)	IN	Bool	在上升沿启动发送作业
EN_R (TRCV_C)	IN	Bool	启用接收
CONT	IN	Bool	控制通信连接： • 0：数据发送完成后断开通信连接。 • 1：建立并保持通信连接。 发送数据 (TSEND_C)（在参数 REQ 的上升沿）或接收数据 (TRCV_C) （在参数 EN_R 的上升沿）时，参数 CONT 的值必须为 TRUE，才能建立或保持连接。

参数和类型		数据类型	说明
LEN	IN	UDInt	可选参数（隐藏） 通过作业发送 (TSEND_C) 或接收 (TRCV_C) 的最大字节数。如果在 DATA 参数中使用纯符号值，则 LEN 参数的值必须为“0”。
ADHOC (TRCV_C)	IN	Bool	可选参数（隐藏） TCP 连接类型的特殊模式请求。
CONNECT	IN_OUT	Variant	指向连接说明的指针： ● 对于 TCP 或 UDP，使用结构 TCON_IP_v4 或 TCON_QDN。 相关说明，请参见：“PROFINET 连接参数 (页 833)”。 ● 对于使用安全通信的 TCP，使用结构 TCON_IP_V4_SEC 或 TCON_QDN_SEC。 相关说明，请参见：“PROFINET 连接参数 (页 833)”。 ● 对于 ISO-on-TCP，使用结构 TCON_IP_RFC。 相关说明，请参见：“PROFINET 连接参数 (页 833)”。 ● 对于 CP 1543-1/CP 1545-1 的 ISO 连接，使用结构 TCON_ISOnative。 相关说明，请参见 TIA Portal 在线帮助：“基于 TCON_ISOnative 的连接描述结构”。 ● 连接 SMS 客户端时，可使用 TCON_PHONE 系统数据类型。 相关说明，请参见 TIA Portal 在线帮助：“TCON_Phone 的连接参数”。 ● 使用 CM 1542-5 进行 FDL 连接时，需使用系统数据类型 TCON_FDL；请参见 TIA Portal 在线帮助：“TCON_FDL 的连接参数”。
DATA	IN_OUT	Variant	指向包含以下内容的发送区的指针： ● 待发送数据的地址和长度 (TSEND_C) ● 所接收数据的地址和最大长度 (TRCV_C)
ADDR	IN_OUT	Variant	可选参数（隐藏） 指向连接类型为 UDP 的接收方地址的指针。地址信息会映射在结构 TADDR_Param ### 中。

参数和类型		数据类型	说明
COM_RST	IN_OUT	Bool	可选参数（隐藏） 重新启动该指令： ● 0：不相关 ● 1：完全重新启动该指令；根据 CONT，现有连接或者会终止，或者会先复位然后再重新建立。 通过 TSEND_C 或 TRCV_C 指令评估后，COM_RST 参数会复位，因此不应对其进行静态切换。
DONE	OUT	Bool	状态参数，可具有以下值： ● 0：发送作业尚未启动或仍在执行。 ● 1：发送作业已正确无误地执行。此状态仅显示一个周期的时间。
BUSY	OUT	Bool	状态参数，可具有以下值： ● 0：发送作业尚未开始或已完成。 ● 1：发送作业尚未完成。无法启动新的发送作业。
ERROR	OUT	Bool	状态参数，可具有以下值： ● 0：无错误 ● 1：建立连接、传输数据或终止连接过程中发生错误。
STATUS	OUT	Word	指令状态（请参见 ERROR 和 STATUS 参数描述）。
RCVD_LEN (TRCV_C)	OUT	Int	实际接收到的数据量（字节）。

说明

TSEND_C 指令需要通过 REQ 输入参数的上升沿来启动发送作业。然后，BUSY 参数在处理期间会设置为 1。发送作业完成时，将通过 DONE 或 ERROR 参数被设置为 1 并持续一个扫描周期进行指示。在此期间，将忽略 REQ 输入参数的上升沿。

说明

LEN 参数的默认设置 (LEN = 0) 使用 DATA 参数来确定要传送的数据的长度。建议由 TSEND_C 指令传送的数据与 TRCV_C 指令的 DATA 参数大小相同。

如果使用 LEN 参数的默认设置且发送的句段数据必须小于 DATA 参数大小，请遵循以下原则。如果 TSEND_C 传输的数据大小不等于 TRCV_C DATA 参数大小，那么 TRCV_C 会保持在忙碌状态（状态代码：7006），直到从 TSEND_C 传输数据全部大小等于 TRCV_C DATA 参数大小。

在数据大小等于 DATA 参数缓冲区大小之前，TRCV_C DATA 参数缓冲区不会显示已接收的新数据。

TSEND_C 操作

TSEND_C 指令异步执行且依次实现以下功能：

1. 设置并建立通信连接：
   如果在参数 REQ 检测到上升沿并且尚不存在任何通信连接，则 TSEND_C 会设置并建立通信连接。设置并建立连接后，CPU 会自动保持和监视该连接。参数 CONNECT 中指定的连接描述用于设置通信连接。可以使用下列连接类型：
   – TCP、ISO-on-TCP 和 UDP 协议的 TCON_Param 结构
   – 对于 V4.1 或更高版本，TCP/UDP：使用结构 TCON_IP_V4 在参数 CONNECT 进行的连接描述
   – 对于 V4.1 或更高版本，ISO-on-TCP：使用结构 TCON_IP_RFC 在参数 CONNECT 进行的连接描述
   – 对于 V4.3 或更高版本，TCP：使用结构 TCON_IP_V4_SEC 在参数 CONNECT 进行的连接描述
   – 对于 V4.4 或更高版本，TCP：使用结构 TCON_QDN 和 TCON_QDN_SEC 在参数 CONNECT 进行的连接描述

   CPU 进入 STOP 模式时，将终止现有连接并移除所设置的相应连接。要再次设置并建立该连接，必须再次执行 TSEND_C。有关可能的通信连接数的信息，请参见 CPU 的技术规格。
2. 通过现有的通信连接发送数据：
   在参数 REQ 中检测到上升沿时执行发送作业。如上文所述，首先将建立通信连接。用户使用参数 DATA 指定发送区。这包括要发送数据的地址和长度。请勿在 DATA 参数中使用数据类型为 BOOL 或 Array of BOOL 的数据区。使用参数 LEN 可指定通过一个发送作业发送的最大字节数。如果在 DATA 参数中使用符号名称，则 LEN 参数的值应为“0”。
   在发送作业完成前不允许编辑要发送的数据。
3. 终止通信连接：
   如果 REQ 参数处于上升沿时 CONT 参数的值为“0”，则发送完数据后将终止通信连接。否则，将保持通信连接。

如果发送作业成功执行，则参数 DONE 将设置为“1”。在此之前，通信连接可能会终止（请参见以上与 CONT 参数相关的说明）。参数 DONE 的信号状态为“1”并不能确认通信伙伴已读取发送数据。

参数 COM_RST 设置为“1”时，将复位 TSEND_C。如果此时传输数据，则数据可能会丢失。

根据 CONT 参数，可能出现以下几种情况：

- CONT = “0”：
  建立现有通信连接。
- CONT =“1”且已建立通信连接：
  复位现有通信连接并再次建立通信连接。
- CONT =“1”且未建立通信连接。
  不建立通信连接。

通过指令 T_SEND 进行评估后，COM_RST 参数将复位。要在执行 (DONE = 1) 后再次启用 TSEND_C，则通过 REQ = 0 调用一次该指令

TRCV_C 操作

TRCV_C 指令异步执行且依次实现以下功能：

1. 设置并建立通信连接：
   如果 EN_R 参数 =“1”并且不存在通信连接，则 TRCV_C 会设置并建立通信连接。设置并建立连接后，CPU 会自动保持和监视该连接。
   参数 CONNECT 中指定的连接描述用于设置通信连接。可以使用下列连接类型：
   – TCP、ISO-on-TCP 和 UDP 协议的 TCON_Param 结构
   – 对于 V4.1 和更高版本，TCP / UDP：通过结构 TCON_IP_V4 在参数 CONNECT 进行的连接描述
   – 对于 V4.1 和更高版本，ISO-on-TCP：通过结构 TCON_IP_RFC 在参数 CONNECT 进行的连接描述
   – 对于 V4.3 和更高版本，TCP：使用结构 TCON_IP_V4_SEC 在参数 CONNECT 进行的连接描述
   – 对于 V4.4 或更高版本，TCP：使用结构 TCON_QDN 和 TCON_QDN_SEC 进行的连接描述。
   CPU 进入 STOP 模式时，将终止现有连接并移除所设置的相应连接。要再次设置并建立该连接，必须使用 EN_R =“1”再次执行 TRCV_C。
   如果在建立通信连接前将 EN_R 设置为“0”，则即使 CONT =“0”仍将建立并保持该连接。但是，不会接收任何数据（DONE 将保持为“0”）。
   有关可能的通信连接数的信息，请参见 CPU 的技术规格。
2. 通过现有的通信连接接收数据：
   参数 EN_R 设置为值“1”时，启用数据接收。如上文所述，首先将建立通信连接。接收到的数据将输入到接收区中。根据所用的协议选项，通过参数 LEN（如果 LEN <> 0）或者通过参数 DATA（如果 LEN = 0）的长度信息指定接收区长度。如果在 DATA 参数中使用纯符号值，则 LEN 参数的值必须为“0”。
   如果在首次接收数据前将 EN_R 设置为“0”，即使 CONT = 0 仍将保持该通信连接。但是，不会接收任何数据（DONE 将保持为“0”）。
3. 终止通信连接：
   如果启动所建立的连接时 CONT 参数的值为“0”，则数据接收完成后将终止通信连接。否则，将保持通信连接。

如果接收作业成功执行，则参数 DONE 将设置为“1”。在此之前，通信连接可能会终止（请参见以上与 CONT 参数相关的说明）。

置位参数 COM_RST 时，TRCV_C 将复位。如果再次执行该指令时正在接收数据，可能会导致数据丢失。根据 CONT 参数，可能出现以下几种情况：

- CONT = “0”：
  建立现有通信连接。
- CONT =“1”且已建立通信连接：
  复位现有通信连接并再次建立通信连接。
- CONT =“1”且未建立通信连接：
  不建立通信连接。

通过指令“TRCV_”进行评估后，COM_RST 参数将复位。

TRCV_C 处理与 TRCV 指令相同的接收模式。下表说明了在接收区输入数据的方法：

协议选项	接收区中数据的可用性	连接描述的参数 Connection_type	LEN 参数	RCVD_LEN 参数
TCP（特殊模式）	数据立即可用。	B#16#11	通过 TRCV_C 指令 ADHOC 输入选择	1 到 1472
TCP（指定长度的数据接收）	完全接收到参数 LEN 指定的数据长度之后，数据就立即可用。	B#16#11	1 到 8192	与参数 LEN 的值相同
ISO on TCP（协议控制的数据传输）	完全接收到参数 LEN 指定的数据长度之后，数据就立即可用。	B#16#12	1 到 8192	与参数 LEN 的值相同

说明

特殊模式

“特殊模式”仅在使用 TCP 协议选项时才可用。要针对特殊模式组态 TRCV_C 指令，请置位 ADHOC 指令输入参数。接收区长度由参数 DATA 中的指针定义。实际接收的数据长度通过 RCVD_LEN 参数输出。最多可接收 1460 个字节。

说明

将包含“特殊模式”的 S7-300/400 STEP 7 项目导入 S7-1200 中

在 S7-300/400 STEP 7 项目中，通过将“0”分配给 LEN 参数来选择“特殊模式”。在 S7-1200 中，可通过置位 ADHOC 指令输入参数为特殊模式组态 TRCV_C 指令。

如果将包含“特殊模式”的 S7-300/400 STEP 7 项目导入 S7-1200 中，则必须将 LEN 参数更改为“65535”。

说明

TCP（指定长度的数据接收）

使用参数 LEN 的值指定数据接收的长度。完全接收到参数 LEN 中指定的长度之后，参数 DATA 中指定的数据在接收区中就立即可用。

说明

ISO on TCP（协议控制的数据传输）

使用 ISO on TCP 协议时，将以协议控制的方式传输数据。接收区由参数 LEN 和 DATA 定义。

BUSY、DONE 和 ERROR 参数

说明

由于 TSEND_C 采用异步处理，所以在 DONE 参数值或 ERROR 参数值为 TRUE 前，必须保持发送方区域中的数据一致。

对于 TSEND_C，参数 DONE 状态为 TRUE 表示数据已成功发送。但并不表示连接伙伴 CPU 实际读取了接收缓冲区。

由于 TRCV_C 采用异步处理，因此仅当参数 DONE = 1 时，接收方区域中的数据才一致。

表格 11-16 TSEND_C 和 TRCV_C 指令的 BUSY、DONE 和 ERROR 参数

BUSY	DONE	ERROR	说明
1	0	0	正在处理发送作业。
0	1	0	发送作业已成功完成。
0	0	1	连接建立或发送作业已完成，但存在一个错误。出错原因在参数 STATUS 中指定。
0	0	0	未分配新的发送作业。

可使用 BUSY、DONE、ERROR 和 STATUS 参数检查执行状态。参数 BUSY 表示作业正在执行。使用参数 DONE，可以检查发送作业是否已成功执行完毕。如果执行 TSEND_C 或 TRCV_C 过程中出错，则将置位 ERROR 参数。错误信息通过参数 STATUS 输出。

Error 和 Status 参数

表格 11-17 TSEND_C 和 TRCV_C 指令的 ERROR 和 STATUS 条件代码

ERROR	STATUS * (W#16#...)	说明
0	0000	发送 (TSEND_C) 或接收 (TRCV_C) 作业已正确无误地执行。
0	0001	通信连接已建立。
0	0003	通信连接已关闭。
0	7000	没有激活的发送作业执行；未建立通信连接。
0	7001	• 启动发送 (TSEND_C) 或接收 (TRCV_C) 作业执行。 • 建立连接。 • 等待连接伙伴。

ERROR	STATUS * (W#16#...)	说明
0	7002	作业正在执行（与 REQ 无关）
0	7003	指令正在终止通信连接。
0	7004	已建立通信连接并对其进行监视；没有激活的发送 (TSEND_C) 或接收 (TRCV_C) 作业执行。
0	7005	TSEND_C：正在进行数据传输。
0	7006	TRCV_C：指令正在接收数据。
1	8085	• 参数 LEN 大于所允许的最大值。 • 参数 LEN 或 DATA 的值在第一次调用指令后发生改变。
1	8086	参数 CONNECT 中的参数 ID 超出了允许范围。
1	8087	已达到最大连接数；无法建立更多连接。
1	8088	参数 LEN 的值与参数 DATA 中设置的接收区不匹配。
1	8089	• CONNECT 参数未指向数据块。 • CONNECT 参数未指向连接描述。 • 对于选定的连接类型，手动创建的连接描述结构错误。
1	8091	超出最大嵌套深度。
1	809A	CONNECT 参数所指向的字段与连接描述的长度不匹配。
1	809B	连接描述中的 InterfaceId 与 CPU 或 CP 不匹配。
1	80A1	• 正在使用连接或端口。 • 通信错误： – 尚未建立指定的连接。 – 正在终止指定的连接。无法通过此连接进行传送。 – 正在重新初始化接口。
1	80A2	本地端口或远程端口已被系统使用。有关更多信息，请参见“TCON 和 TDISCON 指令” (页 862)、“ERROR 和 STATUS 条件代码”。
1	80A3	• 正在尝试重新建立现有连接。 • 正在尝试终止不存在的连接。 • 嵌套的 T_DIAG 指令报告指令已将连接关闭。
1	80A4	该连接远程端点的 IP 地址无效，即此地址与本地伙伴的 IP 地址重复。
1	80A7	通信错误：在发送作业完成之前，通过 COM_RST = 1 调用了该指令。
1	80AA	另一块已使用相同的连接 ID 建立连接。在参数 REQ 的新上升沿重复作业。

ERROR	STATUS * (W#16#...)	说明
1	80B3	• 使用协议类型 UDP 时，ADDR 参数不包含任何数据。 • 连接描述错误 • 另一连接描述已使用本地端口。
1	80B4	使用 ISO-on-TCP 协议选项 (ConnectionType = B#16#12) 建立被动连接 (ActiveEstablished = FALSE) 时，违反了以下一个或两个条件： • local_tsap_id_len >= B#16#02 • local_tsap_id[1] = B#16#E0
1	80B5	连接类型 13 = UDP 仅支持建立被动连接。
1	80B6	连接描述数据块的 ConnectionType 参数存在参数分配错误。
1	80B7	• 对于系统数据类型 TCON_Param： 连接描述数据块的以下参数之一出错：block_length、local_tsap_id_len、rem_subnet_id_len、rem_staddr_len、rem_tsap_id_len、next_staddr_len。 • 对于系统数据类型 TCON_IP_V4、TCON_IP_RFC, TCON_IP_V4_SEC, TCON_QDN, 和 TCON_QDN_SEC ： 指令将伙伴端点的 IP 地址设为 0.0.0.0。
1	80C3	• 所有连接资源均已使用。 • 具有该 ID 的块已在一个具有不同优先级的组中处理。
1	80C4	临时通信错误： • 指令此时无法建立连接。 • 由于连接路径中防火墙的所需端口未打开，因此指令无法建立连接。 • 接口正在接收新参数或指令正在建立连接。 • “TDISCON (页 862)”指令正在删除已组态的连接。 • 使用 COM_RST =1 调用的指令正在终止所用的连接。 • 连接伙伴处暂时无可用的接收资源。连接伙伴尚未就绪，无法接收。
1	80C5	• 通信伙伴终止连接。 • 远程连接伙伴未释放 LSAP.
1	80C6	网络错误： • 本地设备无法访问远程伙伴。 • PROFIBUS 物理断开
1	8722	参数 CONNECT 出错：无效源区域（数据块中未声明的区域）。
1	873A	参数 CONNECT 出错：无法访问连接描述（不能访问数据块）。
1	877F	参数 CONNECT 出错：内部错误
1	8822	TSEND_C： DATA 参数：源区域无效，DB 中不存在该区域。

ERROR	STATUS * (W#16#...)	说明
1	8824	TSEND_C： DATA 参数：指针 VARIANT 存在区域错误。
1	8832	TSEND_C： DATA 参数：DB 编号过大。
1	883A	TSEND_C： CONNECT 参数：无法访问指定的连接数据（例如，由于 DB 不存在）。
1	887F	TSEND_C： DATA 参数：内部错误（例如，VARIANT 引用无效）
1	893A	TSEND_C： DATA 参数：无法访问发送区域（例如，由于 DB 不存在）。
1	8922	TRCV_C： DATA 参数：目标区域无效；DB 中不存在该区域。
1	8924	TRCV_C： DATA 参数：指针 VARIANT 存在区域错误。
1	8932	TRCV_C： DATA 参数：DB 编号过大。
1	893A	TRCV_C： CONNECT 参数：无法访问指定的连接数据（例如，由于 DB 不存在）。
1	897F	TRCV_C： DATA 参数：内部错误（例如，VARIANT 引用无效）。
1	8A3A	TRCV_C： DATA 参数：无法访问该数据区（例如，由于数据块不存在）。
* 在程序编辑器中，错误代码可显示为整数或十六进制值。		

说明

指令 TCON、TSEND、TRCV 和 TDISCON 的错误消息

内部使用时，TSEND_C 指令使用 TCON、TSEND 和 TDISCON 指令；TRCV_C 指令使用 TCON、TRCV 和 TDISCON 指令。有关这些指令错误消息的详细信息，请参见“TCON、TDISCON、TSEND 和 TRCV (页 862)”。

以太网连接协议

每个 CPU 都集成了一个支持标准 PROFINET 通信的 PROFINET 端口。TSEND_C、TRCV_C、TSEND 和 TRCV 指令均支持 TCP 和 ISO on TCP 以太网协议。

更多相关信息，请参见“设备配置：组态本地/伙伴连接路径 (页 812)”。

参见

开放式用户通信指令的连接 ID (页 831)

11.2.8.9 早期 TSEND_C 和 TRCV_C 指令

在 STEP 7 V13 SP1 和 S7-1200 V4.1 CPU 之前的版本中，TSEND_C 和 TRCV_C 指令只能使用结构符合“TCON_Param”的连接参数。 一般概念适用于两个指令集。 关于编程信息，请参见各个早期 TSEND_C 和 TRCV_C 指令。

选择 TSEND_C 和 TRCV_C 指令的版本

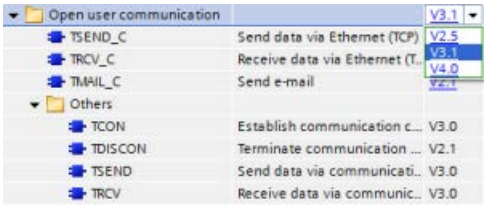
在 STEP 7 中提供了两种版本的 TSEND_C 和 TRCV_C 指令：

- V2.5 和 V3.1 可用于 STEP 7 Basic/Professional V13 或更早版本。
- 版本 4.0 在 STEP 7 Basic/Professional V13 SP1 或新版本中提供。

可以从兼容性和移植便利性方面考虑，选择将相应的指令版本插入用户程序中。

不要在同一个 CPU 程序中使用不同版本的指令。

单击指令树任务卡上的图标可启用指令树的标题和列。



要更改 TSEND_C 和 TRCV_C 指令的版本，请从下拉列表中选择相应版本。可以选择一组指令或分别选择各个指令。

使用指令树将 TSEND_C 或 TRCV_C 指令放入程序时，将根据所选的 TSEND_C 或 TRCV_C 指令在项目树中创建新的 FB 或 FC 实例。在项目树的“PLC_x > 程序块 > 系统块 > 程序资源”(PLC_x > Program blocks > System blocks > Program resources) 下可看到新的 FB 或 FC 实例。

要检验程序中 TSEND_C 或 TRCV_C 指令的版本，必须检查项目树的属性而不是程序编辑器中所显示框的属性。选择项目树的 TSEND_C 或 TRCV_C FB 或 FC 实例，单击右键，选择“属性”(Properties)，然后选择“信息”(Information) 页查看 TSEND_C 或 TRCV_C 指令的版本号。

早期 TSEND_C 和 TRCV_C（通过以太网发送和接收数据）

早期 TSEND_C 指令兼具早期 TCON、TDISCON 和 TSEND 指令的功能。TRCV_C 指令兼具 TCON、TDISCON 和 TRCV 指令的功能。（有关这些指令的详细信息，请参见“早期 TCON、TDISCON、TSEND 和 TRCV（TCP 通信）指令 (页 874)”。）

最少可传送 (TSEND_C) 或接收 (TRCV_C) 一个字节的数据，最多 8192 字节。TSEND_C 不支持传送布尔位置的数据，TRCV_C 也不会在布尔位置中接收数据。有关使用这些指令传送数据的信息，请参见数据一致性 (页 186)部分。

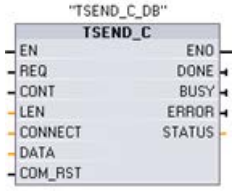
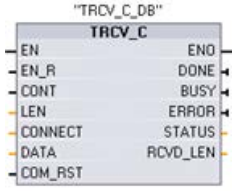
说明

初始化通信参数

插入 TSEND_C 或 TRCV_C 指令之后，可使用该指令 (页 812)的 “属性”(Properties) 来组态通信参数 (页 833)。在巡视窗口为通信伙伴输入参数时，STEP 7 会在指令的背景数据块中输入相应数据。

如果要使用多重背景数据块，必须在两个 CPU 上手动组态该 DB。

表格 11-18 TSEND_C 和 TRCV_C 指令

LAD/FBD	SCL	说明
	"TSEND_C_DB"(req:=_bool_in_, cont:=_bool_in_, len:=_uint_in_, done=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, connect:=_struct_inout_, data:=_variant_inout_, com_rst:=_bool_inout_);	TSEND_C 可与伙伴站建立 TCP 或 ISO on TCP 通信连接、发送数据，并且可以终止该连接。设置并建立连接后，CPU 会自动保持和监视该连接。
	"TRCV_C_DB"(en_r:=_bool_in_, cont:=_bool_in_, len:=_uint_in_, adhoc:=_bool_in_, done=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, rcvd_len=>_uint_out_, connect:=_struct_inout_, data:=_variant_inout_, com_rst:=_bool_inout_);	TRCV_C 可与伙伴 CPU 建立 TCP 或 ISO on TCP 通信连接，可接收数据，并且可以终止该连接。设置并建立连接后，CPU 会自动保持和监视该连接。

[1] STEP 7 会在插入指令时自动创建 DB。

表格 11-19 TSEND_C 和 TRCV_C 参数的数据类型

参数和类型		数据类型	说明
REQ (TSEND_C)	IN	Bool	REQ = 1：在上升沿，利用参数 CONNECT 给出的连接，启动 TSEND_C 发送作业。（创建和保持通信连接，也要求 CONT = 1。）
EN_R (TRCV_C)	IN	Bool	EN_R = 1 时，TRCV_C 准备接收。处理接收作业。（创建和保持通信连接，也要求 CONT = 1。）
CONT	IN	Bool	控制通信连接： • 0：断开通信连接 • 1：建立并保持通信连接 发送数据（TSEND_C）（在参数 REQ 的上升沿）时，参数 CONT 的值必须为 TRUE，才能建立或保持连接。 接收数据（TRCV_C）（在参数 EN_R 的上升沿）时，参数 CONT 的值必须为 TRUE，才能建立或保持连接。
LEN	IN	UInt	要发送 (TSEND_C) 或接收 (TRCV_C) 的最大字节数： • 默认 = 0：DATA 参数确定要发送 (TSEND_C) 或接收 (TRCV_C) 的数据长度。 • 特殊模式 = 65535：设置可变长度的数据接收 (TRCV_C)。
CONNECT	IN_OUT	TCON_Param	指向连接描述 (页 833)的指针
DATA	IN_OUT	Variant	• 包含要发送数据 (TSEND_C) 的地址和长度 • 包含接收数据 (TRCV_C) 的起始地址和最大长度。
COM_RST	IN_OUT	Bool	允许重新启动指令： • 0：不相关 • 1：完成函数块的重新启动，现有连接将终止。
DONE	OUT	Bool	• 0：作业尚未开始或仍在运行。 • 1：作业无错完成。
BUSY	OUT	Bool	• 0：作业完成。 • 1：作业尚未完成。无法触发新作业。
ERROR	OUT	Bool	状态参数，可具有以下值： • 0：无错误 • 1：处理期间出错。STATUS 提供错误类型的详细信息。

参数和类型		数据类型	说明
STATUS	OUT	Word	包括错误信息的状态信息。（请参见下表中的“错误和状态参数”。）
RCVD_LEN (TRCV_C)	OUT	Int	实际接收到的数据量（字节）

说明

TSEND_C 指令需要通过 REQ 输入参数的上升沿来启动发送作业。然后，BUSY 参数在处理期间会设置为 1。发送作业完成时，将通过 DONE 或 ERROR 参数被设置为 1 并持续一个扫描周期进行指示。在此期间，将忽略 REQ 输入参数的上升沿。

说明

LEN 参数的默认设置 (LEN = 0) 使用 DATA 参数来确定要传送的数据的长度。建议由 TSEND_C 指令传送的数据与 TRCV_C 指令的 DATA 参数大小相同。

如果使用 LEN 参数的默认设置且发送的句段数据必须小于 DATA 参数大小，请遵循以下原则。如果 TSEND_C 传输的数据大小不等于 TRCV_C DATA 参数大小，那么 TRCV_C 会保持在忙碌状态（状态代码：7006），直到从 TSEND_C 传输的数据全部大小等于 TRCV_C DATA 参数大小。

在数据大小等于 DATA 参数缓冲区大小之前，TRCV_C DATA 参数缓冲区不会显示已接收的新数据。

TSEND_C 操作

下列功能说明了 TSEND_C 指令的操作：

- 要建立连接，请在 CONT = 1 时执行 TSEND_C。
- 成功建立连接后，TSEND_C 便会置位 DONE 参数一个周期。
- 要终止通信连接，请在 TSEND_C = 0 时执行 CONT。连接将立即中止。这还会影响接收站。将在接收站关闭该连接，并且接收缓冲区内的数据可能会丢失。
- 要通过建立的连接发送数据，请在 REQ 的上升沿执行 TSEND_C。发送操作成功执行后，TSEND_C 便会置位 DONE 参数一个周期。
- 要建立连接并发送数据，请在 CONT =1 且 REQ = 1 时执行 TSEND_C。发送操作成功执行后，TSEND_C 便会置位 DONE 参数一个周期。

TRCV_C 操作

下列功能说明了 TRCV_C 指令的操作：

- 要建立连接，请在参数 CONT = 1 时执行 TRCV_C。
- 要接收数据，请在参数 EN_R = 1 时执行 TRCV_C。参数 EN_R = 1 且 CONT = 1 时，TRCV_C 连续接收数据。
- 要终止连接，请在参数 CONT = 0 时执行 TRCV_C。连接将立即中止，且数据可能丢失。

TRCV_C 处理与 TRCV 指令相同的接收模式。下表说明了在接收区输入数据的方法：

表格 11-20 将数据输入接收区

协议选项	在接收区输入数据	参数 “connection_type”	LEN 参数的值	RCVD_LEN 参数的值（字节）
TCP	特殊模式	B#16#11	65535	1 到 1472
TCP	指定长度的数据接收	B#16#11	0（推荐）或 1 到 8192，65535 除外	1 到 8192
ISO on TCP	特殊模式	B#16#12	65535	1 到 1472
ISO on TCP	协议控制	B#16#12	0（推荐）或 1 到 8192，65535 除外	1 到 8192

说明

特殊模式

使用 TCP 或 ISO on TCP 协议时可以存在“特殊模式”。用户通过将“65535”分配给 LEN 参数来设置特殊模式。接收区与 DATA 构成的区域相同。已接收数据的长度将输出到参数 RCVD_LEN 中。

如果将数据存储在“优化”DB（仅符号访问）中，则只能接收数据类型为 Byte、Char、USInt 和 SInt 的数组中的数据。

说明

将包含“特殊模式”的 S7-300/400 STEP 7 项目导入 S7-1200 中

在 S7-300/400 STEP 7 项目中，通过将“0”分配给 LEN 参数来选择“特殊模式”。在 S7-1200 中，用户通过将“65535”分配给 LEN 参数来设置特殊模式。

如果将包含“特殊模式”的 S7-300/400 STEP 7 项目导入 S7-1200 中，则必须将 LEN 参数更改为“65535”。

说明

在 DONE 参数值或 ERROR 参数值为 TRUE 前，必须保持发送方区域中的数据一致。

由于 TSEND_C 采用异步处理，所以在 DONE 参数值或 ERROR 参数值为 TRUE 前，必须保持发送方区域中的数据一致。

对于 TSEND_C，参数 DONE 状态为 TRUE 表示数据已成功发送。但并不表示连接伙伴 CPU 实际读取了接收缓冲区。

由于 TRCV_C 采用异步处理，因此仅当参数 DONE = 1 时，接收方区域中的数据才一致。

表格 11-21 TSEND_C 和 TRCV_C 指令的 BUSY、DONE 和 ERROR 参数

BUSY	DONE	ERROR	说明
TRUE	不相关	不相关	正在处理作业。
FALSE	TRUE	FALSE	作业已成功完成。
FALSE	FALSE	TRUE	作业因错结束。出错原因可在 STATUS 参数中找到。
FALSE	FALSE	FALSE	未分配新作业。

Error 和 Status 参数

表格 11-22 TSEND_C 和 TRCV_C 指令的 ERROR 和 STATUS 条件代码

ERROR	STATUS	说明
0	0000	作业已无错执行
0	7000	无激活的作业处理
0	7001	启动作业处理，正在建立连接，正在等待连接伙伴
0	7002	正在发送或接收数据
0	7003	正终止连接
0	7004	连接已建立并受到监视，无激活的作业处理
1	8085	LEN 参数的值大于允许的最大值。
1	8086	CONNECT 参数超出允许范围。
1	8087	已达到最大连接数；无法建立更多连接。
1	8088	LEN 参数对于在 DATA 中指定的存储区无效。
1	8089	CONNECT 参数未指向数据块。
1	8091	超出最大嵌套深度。
1	809A	CONNECT 参数指向的字段与连接描述的长度不匹配。

ERROR	STATUS	说明
1	809B	连接描述中的 local_device_id 与 CPU 的不匹配。
1	80A1	通信错误： ● 尚未建立指定的连接 ● 当前正在终止指定的连接；无法通过该连接传输 ● 正在重新初始化接口
1	80A3	正在尝试终止不存在的连接
1	80A4	远程伙伴连接的 IP 地址无效。例如，远程伙伴的 IP 地址与本地伙伴的 IP 地址相同。
1	80A5	连接 ID (页 831) 已被使用。
1	80A7	通信错误：在 TSEND_C 完成前调用了 TDISCON。
1	80B2	参数 CONNECT 指向使用关键字 UNLINKED 生成的数据块。
1	80B3	不一致的参数： ● 连接描述错误 ● 本地端口（参数 local_tsap_id）已在另一个连接描述中存在。 ● 连接描述中的 ID 与作为参数指定的 ID 不同
1	80B4	使用 ISO on TCP (connection_type = B#16#12) 建立被动连接时，条件代码 80B4 提示您输入的 TSAP 不符合下列某一项地址要求： ● 如果本地 TSAP 长度为 2 个字节且首字节的 TSAP ID 值为 E0 或 E1（十六进制），则第二字节必须为 00 或 01。 ● 如果本地 TSAP 长度为 3 个或更多字节，且首字节的 TSAP ID 值为 E0 或 E1（十六进制），则第二字节必须为 00 或 01，且所有其它字节必须为有效的 ASCII 字符。 ● 如果本地 TSAP 长度为 3 个或更多字节，且首字节的 TSAP ID 值既不为 E0 也不为 E1（十六进制），则 TSAP ID 的所有字节都必须为有效的 ASCII 字符。 有效 ASCII 字符的字节值为 20 到 7E（十六进制）。
1	80B7	所传送数据的数据类型和/或长度与伙伴 CPU 上用于写入该数据的区域不相符。
1	80C3	所有连接资源均已使用。
1	80C4	临时通信错误： ● 此时无法建立连接 ● 接口正在接收新参数 ● TDISCON 当前正在删除已组态连接。
1	8722	CONNECT 参数：源区域无效：DB 中不存在该区域。
1	873A	CONNECT 参数：无法访问连接描述（例如，DB 不可用）

ERROR	STATUS	说明
1	877F	CONNECT 参数：内部错误，如无效的 ANY 引用
1	893A	参数包含未装载的 DB 的编号。

以太网连接协议

每个 CPU 都集成了一个支持标准 PROFINET 通信的 PROFINET 端口。TSEND_C、TRCV_C、TSEND 和 TRCV 指令均支持 TCP 和 ISO on TCP 以太网协议。

更多相关信息，请参见“设备配置：组态本地/伙伴连接路径 (页 812)”。

11.2.8.10 TCON、TDISCON、TSEND 和 TRCV 指令

从 S7-1200 CPU V4.1 或更高版本开始，与 STEP 7 V13 SP1 或更高版本一起使用时，S7-1200 CPU 可以扩展 TCON 指令的功能，以便使用结构符合 TCON_IP_V4 和 TCON_IP_RFC 的连接参数。S7-1200 CPU 还可以扩展 TSEND 和 TRCV 指令的功能，以便使用结构符合 TCON_IP_V4 和 TCON_IP_RFC 的连接参数。

从 S7-1200 CPU V4.3 或更高版本开始，与 STEP 7 V15.1 或更高版本一起使用时，S7-1200 CPU 可以扩展 TCON 指令的功能，以便使用结构符合 TCON_IP_V4、TCON_IP_V4_SEC 和 TCON_IP_RFC 的连接参数。

从 S7-1200 CPU V4.4 或更高版本开始，与 STEP 7 V16 或更高版本一起使用时，CPU 可以扩展 TCON 指令的功能，以便使用结构符合 TCON_IP_V4、TCON_IP_V4_SEC、TCON_IP_RFC、TCON_QDN 和 TCON_QDN_SEC 的连接参数。

因此，S7-1200 支持两组 TCON、TDISCON、TSEND 和 TRCV 指令：

- 早期 TCON、TDISCON、TSEND 和 TRCV 指令 (页 874)：这些 TCON、TDISCON、TSEND 和 TRCV 指令在 S7-1200 V4.0 之前的版本中便已存在，只能使用结构符合 TCON_Param 的连接参数。
- TCON、TDISCON、TSEND 和 TRCV 指令 (页 862)：这些 TCON、TDISCON、TSEND 和 TRCV 指令具备早期指令的所有功能，而且还能够使用结构符合 TCON_IP_V4、TCON_IP_V4_SEC、TCON_IP_RFC, TCON_QDN 和 TCON_QDN_SEC 的连接参数。

选择 TCON、TDISCON、TSEND 和 TRCV 指令的版本

STEP 7 中提供两个版本的 TCON、TDISCON、TSEND 或 TRCV 指令：

- V2.5 和 V3.1 可用于 STEP 7 Basic/Professional V13 或更早版本。
- 版本 4.0 在 STEP 7 Basic/Professional V13 SP1 或新版本中提供。

可以从兼容性和移植便利性方面考虑，选择将相应的指令版本插入用户程序中。

不要在同一个 CPU 程序中使用不同版本的指令。

单击指令树任务卡上的图标可启用指令树的标题和列。

要更改 TCON、TDISCON、TSEND 或 TRCV 指令的版本，请从下拉列表中选择版本。可以选择一组指令或分别选择各个指令。

使用指令树将 TCON、TDISCON、TSEND 或 TRCV 指令放入程序时，将根据所选的 TCON、TDISCON、TSEND 或 TRCV 指令在项目树中创建新的 FB 或 FC 实例。在项目树的“PLC_x > 程序块 > 系统块 > 程序资源”(PLC_x > Program blocks > System blocks > Program resources) 下可看到新的 FB 或 FC 实例。

要检验程序中 TCON、TDISCON、TSEND 或 TRCV 指令的版本，必须检查项目树的属性而不是程序编辑器中所显示框的属性。选择项目树的 TCON、TDISCON、TSEND 或 TRCV FB 或 FC 实例，单击右键，选择“属性”(Properties)，然后选择“信息”(Information) 页查看 TCON、TDISCON、TSEND 或 TRCV 指令的版本号。

TCON、TDISCON、TSEND 和 TRCV（TCP 通信）指令

使用 TCP 和 ISO on TCP 协议的以太网通信

说明

TSEND_C 和 TRCV_C 指令

为帮助简化 PROFINET/以太网通信的编程，TSEND_C 指令和 TRCV_C 指令兼具 TCON、TDISCON、TSEND 和 TRCV 指令的功能：

- TSEND_C 兼具 TCON、TDISCON 和 TSEND 指令的功能。
- TRCV_C 兼具 TCON、TDISCON 和 TRCV 指令的功能。

以下指令控制通信过程：

- TCON 在客户机与服务器 (CPU) PC 之间建立 TCP/IP 连接。
- TSEND 和 TRCV 发送和接收数据。
- TDISCON 断开连接。

最少可传送 (TSEND) 或接收 (TRCV) 一个字节的数据，最多 8192 字节。TSEND 不支持传送布尔位置的数据，TRCV 也不会在布尔位置中接收数据。有关使用这些指令传送数据的信息，请参阅数据一致性 (页 186)部分。

TCON、TDISCON、TSEND 和 TRCV 异步运行，即，作业处理需要多次执行指令来完成。例如，执行参数 REQ = 1 的 TCON 指令来启动用于设置和建立连接的作业。然后，另外执行 TCON 来监视作业进度并使用参数 DONE 来测试作业是否完成。

下表给出了 BUSY、DONE 和 ERROR 之间的关系。使用该表可以确定当前作业状态：

表格 11-23 BUSY、DONE 和 ERROR 参数之间的交互作用

BUSY	DONE	ERROR	说明
1	0	0	正在处理作业。
0	1	0	作业已成功完成。
0	0	1	作业以出错而结束。错误原因通过参数 STATUS 输出。
0	0	0	未分配新作业。

TCON 和 TDISCON

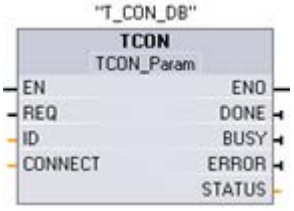
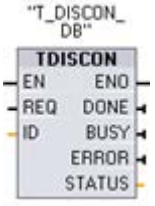
说明

初始化通信参数

插入 TCON 指令之后，可使用该指令 (页 812)的 “属性”(Properties) 来组态通信参数 (页 833)。在巡视窗口为通信伙伴输入参数时，STEP 7 会在指令的背景数据块中输入相应数据。

如果要使用多重背景数据块，必须在两个 CPU 上手动组态该 DB。

表格 11-24　TCON 和 TDISCON 指令

LAD/FBD		说明
	"TCON_DB"(req:=_bool_in_, ID:=_undef_in_, done=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, connect:=_struct_inout_);	TCP 和 ISO on TCP：TCON 启动从 CPU 到通信伙伴的通信连接。
	"TDISCON_DB"(req:=_bool_in_, ID:=_word_in_, done=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_);	TCP 和 ISO on TCP：TDISCON 终止从 CPU 到通信伙伴的通信连接。

[1]　STEP 7 会在插入指令时自动创建 DB。

表格 11-25　TCON 和 TDISCON 参数的数据类型

参数	声明	数据类型	说明
REQ	IN	Bool	在上升沿时，启动相应作业以建立 ID 所指定的连接。
ID	IN	CONN_OUC (Word)	引用已分配的连接。 值范围：W#16#0001 到 W#16#0FFF

参数	声明	数据类型	说明
CONNECT (TCON)	IN_OUT	VARIANT	指向连接说明的指针： • 对于 TCP 或 UDP，使用结构 TCON_IP_v4 或 TCON_QDN。 相关说明，请参见：“PROFINET 连接参数 (页 833)”。 • 对于使用安全通信的 TCP，使用结构 TCON_IP_V4_SEC 或 TCON_QDN_SEC。 相关说明，请参见：“PROFINET (页 833) 连接参数 (页 833)”。 • 对于 ISO-on-TCP，使用结构 TCON_IP_RFC。 相关说明，请参见：“PROFINET 连接参数 (页 833)”。 • 对于 CP 1543-1/CP 1545-1 的 ISO 连接，使用结构 TCON_ISOnative。 相关说明，请参见：TIA Portal 在线帮助：“基于 TCON_ISOnative 的连接描述结构”。 • 连接 SMS 客户端时，可使用 TCON_PHONE 系统数据类型。 相关说明，请参见 TIA Portal 在线帮助：“TCON_Phone 的连接参数”。 • 使用 CM 1542-5 进行 FDL 连接时，需使用系统数据类型 TCON_FDL；请参见 TIA Portal 在线帮助：“TCON_FDL 的连接参数”。
DONE	OUT	Bool	状态参数，可具有以下值： • 0：作业尚未启动，或仍在执行过程中。 • 1：作业已执行，且无任何错误。
BUSY	OUT	Bool	状态参数，可具有以下值： • 0：作业尚未启动或已完成。 • 1：作业尚未完成。无法启动新作业。
ERROR	OUT	Bool	状态参数 ERROR： • 0：无错误 • 1：已出错
STATUS	OUT	Word	指令的状态

两个通信伙伴都执行 TCON 指令来设置和建立通信连接。用户使用参数指定主动和被动通信端点伙伴。设置并建立连接后，CPU 会自动保持和监视该连接。

如果连接终止（例如，因断线或远程通信伙伴原因），主动伙伴将尝试重新建立组态的连接。不必再次执行 TCON。

执行 TDISCON 指令或 CPU 切换到 STOP 模式后，会终止现有连接并删除所设置的连接。要设置和重新建立连接，必须再次执行 TCON。

表格 11-26 ERROR 和 STATUS 指令的 TCON 和 TDISCON 条件代码

ERROR	STATUS * (W#16#...)	说明
0	0000	连接已成功建立。
0	7000	无激活的作业处理
0	7001	启动作业执行；建立连接 (TCON) 或终止连接 (TDISCON)。
0	7002	指令正在建立连接（与 REQ 无关）；建立连接 (TCON) 或终止连接 (TDISCON)。
1	8085	TCON：正在使用连接 ID。
1	8086	TCON：ID 参数超出了有效范围。
1	8087	TCON：已达到最大连接数；无法建立更多连接
1	8089	TCON：参数 CONNECT 未指向连接描述，或者连接描述是手动创建的。
1	809A	TCON：指令不支持参数 CONNECT 中的结构或者长度无效。
1	809B	TCON： ● 连接描述中的 InterfaceId 元素与 CPU 或 CP 不匹配，或者值为“0”。 ● TCON_xxx 结构中的 InterfaceId 元素未引用 CPU 或 CM/CP 接口的硬件标识符。
1	80A1	TCON：TCP/UDP：连接或端口处于使用状态。
1	80A2	TCON: 系统正在使用本地或远程端口。有关更多信息，请参见“指令的通用参数”(页 932)、“被动 ISO 和 TCP 通信的受限 TSAP 和端口号”。
1	80A3	TCON：用户程序创建的连接 (TCON) 正在使用 ID 参数的值。连接使用相同的 ID，且参数 CONNECT 中的连接设置相同。
1	80A4	TCON：该连接远程端点的 IP 地址无效，或者与本地伙伴的 IP 地址重复。
1	80A7	TCON：通信错误：在“TCON”完成前执行了“TDISCON”。
1	80B3	参数分配不一致
1	80B4	TCON：仅适用于 TCON_IP_RFC：出现以下情况之一： ● 指令未分配本地 T 选择器。 ● 第一个字节不含值 0x0E。 ● 本地 T 选择器以“SIMATIC-”开头。
1	80B5	TCON：指令只允许为连接类型 13 = UDP 建立被动连接（结构 TCON_xxx 的 ActiveEstablished 参数值为 TRUE）。

ERROR	STATUS * (W#16#...)	说明
1	80B6	TCON：数据块的 ConnectionType 参数中存在连接描述参数分配错误： ● 仅对 TCON_IP_V4, TCON_IP_V4_SEC, TCON_QDN, TCON_QDN_SEC 有效：0x11、0x0B 和 0x13 ● 仅对 TCON_IP_RFC 有效：0x0C 和 0x12
1	80B7	TCON：对于 TCON_IP_V4, TCON_IP_V4_SEC, TCON_QDN, TCON_QDN_SEC： ● TCP（主动建立连接）：远程端口为“0”。 ● TCP（被动建立连接）：本地端口为“0”。 ● UDP：本地端口为“0”。 ● 指令将伙伴端点的 IP 地址设为 0.0.0.0。 TCON：对于 TCON_IP_RFC： ● 指令指定的本地 (local_tselector) 或远程 (remote_tselector) T 选择器的长度超过 32 字节。 ● 对于 T 选择器（本地或远程）的 TSelLength，指令指定的长度大于 32 字节。 ● 特定连接伙伴的 IP 地址长度出错 ● 指令将伙伴端点的 IP 地址设为 0.0.0.0。
1	80B8	TCON：本地连接描述（参数 CONNECT 中的结构）的参数 ID 和指令的参数 ID 不同。
1	80C3	TCON：所有连接资源均已使用。
1	80C4	临时通信错误： ● 指令此时无法建立连接 (TCON)。 ● 由于连接路径中防火墙的所需端口未打开，因此指令无法建立连接 (TCON)。 ● 接口正在接收新参数（TCON 和 TDISCON）。 ● “TDISCON”指令正在删除已组态的连接 (TCON)。
1	80C5	TCON：远程伙伴执行以下操作之一： ● 拒绝建立连接 ● 终止连接 ● 主动结束连接
1	80C6	TCON：指令无法访问远程伙伴（网络错误）。
1	80C7	TCON：执行超时
1	80C8	TCON：用户程序创建的连接 (TCON) 正在使用 ID 参数的值。连接使用相同的 ID，但参数 CONNECT 中的连接设置不同。
1	80C9	TCON：远程伙伴验证失败。想要建立连接的远程伙伴与参数 CONNECT 的结构中定义的伙伴不匹配。

ERROR	STATUS * (W#16#...)	说明
1	80CE	TCON：本地接口的 IP 地址为 0.0.0.0。
1	80E0	TCON：指令接收了不适用或错误的消息。
1	80E1	TCON：通信握手过程中发生错误。可能的原因： ● 用户中止该过程 ● 安全等级不足 ● 指令不支持重新协商。 ● 指令不支持 SSL/TLS 版本。 ● 主机名称验证失败。
1	80E2	证书不受支持/证书无效/无证书 可能的原因：对于相关模块，CPU 未设置时钟或同步模块。 示例：模块默认的日期设置为 1/1/2012，且 CPU 在调试过程中未设置日期。证书的有效期从 2016 年 8 月 20 日开始，到 2024 年 8 月 20 日结束。在这种情况下，模块的日期超出证书的有效期；证书对于模块无效。
1	80E3	证书已丢弃。
1	80E4	未找到有效的证书颁发机构。
1	80E5	证书已过期。
1	80E6	传输层安全协议中发生完整性错误
1	80E7	不支持 X.509-V3 证书扩展
1	80E9	指令不支持无服务器证书的 TLS 服务器。
1	80EA	指令不支持 DTLS (UDP) 协议。
1	80EB	客户端无法请求客户端证书。
1	80EC	服务器无法基于 subjectAlternateName 进行验证（仅适用于客户端）。
1	80ED	TLSServerCertRef_m-ID 无效
* 在程序编辑器中，错误代码可显示为整数或十六进制值。		

TSEND 和 TRCV

说明

使用 PROFINET 开放式用户通信协议时，如果执行 TSEND 指令但不在远程设备上执行相应的 TRCV 指令，则 TSEND 指令可能无限期处于“繁忙状态”，等待 TRCV 指令接收数据。在这种状态下，TSEND 指令“繁忙”输出将置位，“状态”输出的值为“0x7002”。传输的数据大于 4096 字节时可能会出现这种情况。在下次执行 TRCV 指令时会解决这一问题。

表格 11-27 TSEND 和 TRCV 指令

LAD/FBD	SCL	说明
	"TSEND_DB"(req:=_bool_in_, ID:=_word_in_, len:=_udint_in_, done=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, data:=_variant_inout_);	TCP 和 ISO on TCP：TSEND 通过从 CPU 到伙伴站的通信连接发送数据。
	"TRCV_DB"(en_r:=_bool_in_, ID:=_word_in_, len:=_udint_in_, adhoc:=_bool_in_, ndr=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, rcvd_len=>_udint_out_, data:=_variant_inout_);	TCP 和 ISO on TCP：TRCV 通过从伙伴站到 CPU 的通信连接接收数据。

[1] STEP 7 会在插入指令时自动创建 DB。

表格 11-28 TSEND 和 TRCV 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	TSEND：在上升沿启动发送作业。传送通过 DATA 和 LEN 指定的区域中的数据。
EN_R	IN	Bool	TRCV：允许 CPU 进行接收；EN_R = 1 时，TRCV 准备接收。处理接收作业。
ID	IN	CONN_OUC (Word)	指向相关连接的引用。ID 必须与本地连接描述信息内的相关参数 ID 相同。 值范围：W#16#0001 到 W#16#0FFF
LEN	IN	UDInt	要发送 (TSEND) 或接收 (TRCV) 的最大字节数： ● 默认 = 0：DATA 参数确定要发送 (TSEND) 或接收 (TRCV) 的数据长度。 ● 特殊模式 = 65535：设置可变长度的数据接收 (TRCV)。
ADHOC	IN	Bool	TRCV：可选参数（隐藏） TCP 连接类型的特殊模式请求。

参数和类型		数据类型	说明
DATA	IN_OUT	Variant	指向发送 (TSEND) 或接收 (TRCV) 数据区的指针；数据区包含地址和长度。该地址引用 I 存储器、Q 存储器、M 存储器或 DB。
DONE	OUT	Bool	TSEND： ● 0：作业尚未开始或仍在运行。 ● 1：无错执行作业。
NDR	OUT	Bool	TRCV： ● NDR = 0：作业尚未开始或仍在运行。 ● NDR = 1：作业已成功完成。
BUSY	OUT	Bool	● BUSY = 1：作业尚未完成。无法触发新作业。 ● BUSY = 0：作业已完成。
ERROR	OUT	Bool	ERROR = 1：处理期间出错。STATUS 提供错误类型的详细信息
STATUS	OUT	Word	包括错误信息的状态信息。（请参见下表中的错误和状态条件代码。）
RCVD_LEN	OUT	UDInt	TRCV：实际接收到的数据量（以字节为单位）

说明

TSEND 指令需要通过 REQ 输入参数的上升沿来启动发送作业。然后，BUSY 参数在处理期间会设置为 1。发送作业完成时，将通过 DONE 或 ERROR 参数被设置为 1 并持续一个扫描周期进行指示。在此期间，将忽略 REQ 输入参数的上升沿。

TRCV 操作

TRCV 指令将收到的数据写入到通过以下两个变量指定的接收区：

- 指向区域起始位置的指针
- 如果不为 0 则为区域长度或 LEN 上提供的值

说明

LEN 参数的默认设置 (LEN = 0) 使用 DATA 参数来确定要传送的数据的长度。建议由 TSEND 指令传送的数据与 TRCV 指令的 DATA 参数大小相同。

如果使用 LEN 参数的默认设置且发送的句段数据必须小于 DATA 参数大小，请遵循以下原则。建议持续高 EN_R 位直到相应 TSEND 传送适当量的数据来填充 TRCV DATA 参数。如果 TSEND 传输的数据大小不等于 TRCV DATA 参数大小，那么 TRCV 会保持在忙碌状态（状态代码：7002），然而 EN_R 位为高直到从 TSEND 传输的数据全部大小等于 TRCVDATA 参数大小。如果 TRCV 的 EN_R 位为脉冲，则它需要执行与 TSEND 次数相同的脉冲来接收数据。

在数据大小等于 DATA 参数缓冲区大小之前，TRCV DATA 参数缓冲区不会显示已接收的新数据。

接收所有作业数据后，TRCV 会立即将其传送到接收区并将 NDR 设置为 1。

表格 11-29 将数据输入接收区

协议选项	在接收区输入数据	参数 “connection_type”	LEN 参数的值	RCVD_LEN 参数的值（字节）
TCP	特殊模式	B#16#11	通过 TRCV 指令 ADHOC 输入选择	1 到 1472
TCP	指定长度的数据接收	B#16#11	0（推荐）或 1 到 8192，65535 除外	1 到 8192
ISO on TCP	特殊模式	B#16#12	65535	1 到 1472
ISO on TCP	协议控制	B#16#12	0（推荐）或 1 到 8192，65535 除外	1 到 8192

说明

特殊模式

使用 TCP 或 ISO on TCP 协议时可以存在“特殊模式”。要针对特殊模式组态 TRCV 指令，请置位 ADHOC 指令输入参数。接收区与 DATA 构成的区域相同。已接收数据的长度将输出到参数 RCVD_LEN 中。接收数据块后，TRCV 会立即将数据写入接收区并将 NDR 设置为 1。

如果将数据存储在“优化”DB（仅符号访问）中，则只能接收数据类型为 Byte、Char、USInt 和 SInt 的数组中的数据。

说明

将包含“特殊模式”的 S7-300/400 STEP 7 项目导入 S7-1200 中

在 S7-300/400 STEP 7 项目中，通过将“0”分配给 LEN 参数来选择“特殊模式”。在 S7-1200 中，可通过置位 TRCV 指令输入参数为特殊模式组态 ADHOC 指令。

如果将包含“特殊模式”的 S7-300/400 STEP 7 项目导入 S7-1200 中，则必须将 LEN 参数更改为“65535”。

表格 11-30 TSEND 和 TRCV 指令的 ERROR 和 STATUS 条件代码

ERROR	STATUS	说明
0	0000	• 发送作业无错完成 (TSEND) • 已接受新数据：在 RCVD_LEN 中显示已接收数据的当前长度 (TRCV)。
0	7000	• 无激活的作业处理 (TSEND) • 块未准备好接收 (TRCV)
0	7001	• 启动作业处理，正在发送数据：在执行此处理期间，操作系统访问 DATA 发送区中的数据 (TSEND)。 • 块准备接收，接收作业已激活 (TRCV)。
0	7002	• 后续指令执行（与 REQ 无关），正在处理作业：在执行此处理期间，操作系统访问 DATA 发送区中的数据 (TSEND)。 • 后续指令执行，正在处理接收作业：数据在执行此处理期间写入接收区。因此，错误可能导致接收区中的数据不一致 (TRCV)。
1	8085	• LEN 参数的值大于允许的最大值 (TSEND) 和 (TRCV)。 • 自第一次指令执行 (TRCV) 以来，LEN 或 DATA 参数发生变化。
1	8086	ID 参数不在允许的地址范围内。
1	8088	LEN 参数大于 DATA 中指定的存储区。

ERROR	STATUS	说明
1	80A1	通信错误： ● 尚未建立指定的连接（TSEND 和 TRCV）。 ● 当前正在终止指定的连接。无法通过该连接执行传送或接收作业（TSEND 和 TRCV）。 ● 正在重新初始化接口 (TSEND)。 ● 接口正在接收新参数 (TRCV)。
1	80C3	内部缺乏连接 (页 831)资源：具有该 ID 的块正在一个具有不同优先级的组中处理。
1	80C4	临时通信错误： ● 此时无法建立与通信伙伴的连接。 ● 接口正在接收新参数设置或当前正在建立连接。

以太网连接协议

每个 CPU 都集成了一个支持标准 PROFINET 通信的 PROFINET 端口。TSEND_C、TRCV_C、TSEND 和 TRCV 指令均支持 TCP 和 ISO-on-TCP 以太网协议。

更多相关信息，请参见“设备配置：组态本地/伙伴连接路径 (页 812)”。

11.2.8.11 早期 TCON、TDISCON、TSEND 和 TRCV 指令

在 STEP 7 V13 SP1 和 S7-1200 V4.1 CPU 之前的版本中，TCON、TDISCON、TSEND 和 TRCV 指令只能使用结构符合“TCON_Param”的连接参数。 一般概念适用于两个指令集。 相关的编程信息，请参见各个早期 TCON、TDISCON、TSEND 和 TRCV 指令。

选择 TCON、TDISCON、TSEND 和 TRCV 指令的版本

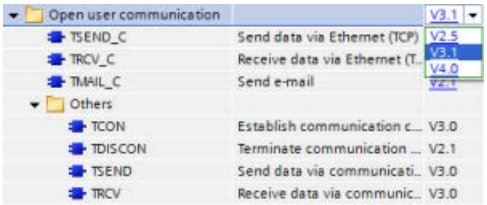
STEP 7 中提供两个版本的 TCON、TDISCON、TSEND 或 TRCV 指令：

- V2.5 和 V3.1 可用于 STEP 7 Basic/Professional V13 或更早版本。
- 版本 4.0 在 STEP 7 Basic/Professional V13 SP1 或新版本中提供。

可以从兼容性和移植便利性方面考虑，选择将相应的指令版本插入用户程序中。

不要在同一个 CPU 程序中使用不同版本的指令。

单击指令树任务卡上的图标可启用指令树的标题和列。



要更改 TCON、TDISCON、TSEND 或 TRCV 指令的版本，请从下拉列表中选择版本。可以选择一组指令或分别选择各个指令。

使用指令树将 TCON、TDISCON、TSEND 或 TRCV 指令放入程序时，将根据所选的 TCON、TDISCON、TSEND 或 TRCV 指令在项目树中创建新的 FB 或 FC 实例。在项目树的“PLC_x > 程序块 > 系统块 > 程序资源”(PLC_x > Program blocks > System blocks > Program resources) 下可看到新的 FB 或 FC 实例。

要检验程序中 TCON、TDISCON、TSEND 或 TRCV 指令的版本，必须检查项目树的属性而不是程序编辑器中所显示框的属性。选择项目树的 TCON、TDISCON、TSEND 或 TRCV FB 或 FC 实例，单击右键，选择 “属性”(Properties)，然后选择 “信息”(Information) 页查看 TCON、TDISCON、TSEND 或 TRCV 指令的版本号。

早期 TCON、TDISCON、TSEND 和 TRCV（TCP 通信）指令

使用 TCP 和 ISO on TCP 协议的以太网通信

说明

TSEND_C 和 TRCV_C 指令

为帮助简化 PROFINET/以太网通信的编程，TSEND_C 指令和 TRCV_C 指令兼具 TCON、TDISCON、TSEND 和 TRCV 指令的功能：

- TSEND_C 兼具 TCON、TDISCON 和 TSEND 指令的功能。
- TRCV_C 兼具 TCON、TDISCON 和 TRCV 指令的功能。

以下指令控制通信过程：

- TCON 在客户机与服务器 (CPU) PC 之间建立 TCP/IP 连接。
- TSEND 和 TRCV 发送和接收数据。
- TDISCON 断开连接。

最少可传送 (TSEND) 或接收 (TRCV) 一个字节的数据，最多 8192 字节。TSEND 不支持传送布尔位置的数据，TRCV 也不会在布尔位置中接收数据。有关使用这些指令传送数据的信息，请参阅数据一致性 (页 186)部分。

TCON、TDISCON、TSEND 和 TRCV 异步运行，即，作业处理需要多次执行指令来完成。例如，执行参数 REQ = 1 的 TCON 指令来启动用于设置和建立连接的作业。然后，另外执行 TCON 来监视作业进度并使用参数 DONE 来测试作业是否完成。

下表给出了 BUSY、DONE 和 ERROR 之间的关系。使用该表可以确定当前作业状态：

表格 11-31 BUSY、DONE 和 ERROR 参数之间的交互作用

BUSY	DONE	ERROR	说明
TRUE	不相关	不相关	正在处理作业。
FALSE	TRUE	FALSE	作业已成功完成。
FALSE	FALSE	TRUE	作业因错结束。出错原因可在 STATUS 参数中找到。
FALSE	FALSE	FALSE	未分配新作业。

TCON 和 TDISCON

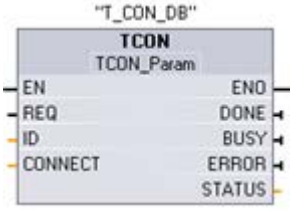
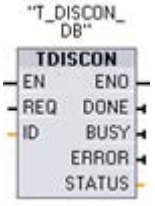
说明

初始化通信参数

插入 TCON 指令之后，可使用该指令 (页 812)的 “属性”(Properties) 来组态通信参数 (页 833)。在巡视窗口为通信伙伴输入参数时，STEP 7 会在指令的背景数据块中输入相应数据。

如果要使用多重背景数据块，必须在两个 CPU 上手动组态该 DB。

表格 11-32 TCON 和 TDISCON 指令

LAD/FBD		说明
	"TCON_DB"(req:=_bool_in_, ID:=_undef_in_, done=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, connect:=_struct_inout_);	TCP 和 ISO on TCP：TCON 启动从 CPU 到通信伙伴的通信连接。
	"TDISCON_DB"(req:=_bool_in_, ID:=_word_in_, done=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_);	TCP 和 ISO on TCP：TDISCON 终止从 CPU 到通信伙伴的通信连接。

[1] STEP 7 会在插入指令时自动创建 DB。

表格 11-33 TCON 和 TDISCON 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	控制参数 REQ 启动用于建立通过 ID 指定的连接的作业。在上升沿时启动该作业。
ID	IN	CONN_OUC (Word)	引用要建立的 (TCON) 或终止的 (TDISCON)连接、到远程伙伴的连接或用户程序与操作系统通信层之间的连接。ID 必须与本地连接描述中的相关参数 ID 相同。 值范围：W#16#0001 到 W#16#0FFF
CONNECT (TCON)	IN_OUT	TCON_Param	指向连接描述 (页 833)的指针
DONE	OUT	Bool	• 0：作业尚未开始或仍在运行。 • 1：作业无错完成。
BUSY	OUT	Bool	• 0：作业完成。 • 1：作业尚未完成。无法触发新作业。

参数和类型		数据类型	说明
ERROR	OUT	Bool	状态参数，可具有以下值： ● 0：无错误 ● 1：处理期间出错。STATUS 提供错误类型的详细信息。
STATUS	OUT	Word	包括错误信息的状态信息。（请参见下表中的错误和状态条件代码。）

两个通信伙伴都执行 TCON 指令来设置和建立通信连接。用户使用参数指定主动和被动通信端点伙伴。设置并建立连接后，CPU 会自动保持和监视该连接。

如果连接终止（例如，因断线或远程通信伙伴原因），主动伙伴将尝试重新建立组态的连接。不必再次执行 TCON。

执行 TDISCON 指令或 CPU 切换到 STOP 模式后，会终止现有连接并删除所设置的连接。要设置和重新建立连接，必须再次执行 TCON。

表格 11-34 ERROR 和 STATUS 指令的 TCON 和 TDISCON 的条件代码

ERROR	STATUS	说明
0	0000	连接已成功建立。
0	7000	无激活的作业处理
0	7001	启动作业处理；正在建立连接 (TCON) 或正在终止连接 (TDISCON)
0	7002	后续调用（与 REQ 无关）；正在建立连接 (TCON) 或正在终止连接 (TDISCON)
1	8086	参数 ID 超出允许的地址范围。
1	8087	TCON：已达到最大连接数；无法建立更多连接。
1	809B	TCON：连接描述中的 local_device_id 与 CPU 的不匹配。
1	80A1	TCON：连接或端口已被用户占用。
1	80A2	TCON：本地端口或远程端口已被系统占用。
1	80A3	正在尝试重新建立现有连接 (TCON) 或终止不存在的连接 (TDISCON)。
1	80A4	TCON：远程连接端点的 IP 地址无效；可能与本地通信伙伴的 IP 地址匹配。
1	80A5	TCON：连接 ID (页 831) 已被使用。
1	80A7	TCON：通信错误：在 TDISCON 完成前执行了 TCON。TDISCON 必须先完全终止 ID 引用的连接。
1	80B2	TCON：CONNECT 参数指向通过属性“仅存储在装载存储器中”生成的某个数据块。

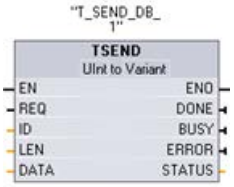
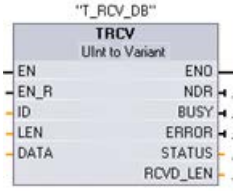
ERROR	STATUS	说明
1	80B4	TCON：使用 ISO on TCP (connection_type = B#16#12) 建立被动连接时，条件代码 80B4 提示您输入的 TSAP 不符合下列某一项地址要求： ● 如果本地 TSAP 长度为 2 个字节且首字节的 TSAP ID 值为 E0 或 E1（十六进制），则第二字节必须为 00 或 01。 ● 如果本地 TSAP 长度为 3 个或更多字节，且首字节的 TSAP ID 值为 E0 或 E1（十六进制），则第二字节必须为 00 或 01，且所有其它字节必须为有效的 ASCII 字符。 ● 如果本地 TSAP 长度为 3 个或更多字节，且首字节的 TSAP ID 值既不为 E0 也不为 E1（十六进制），则 TSAP ID 的所有字节都必须为有效的 ASCII 字符。 有效 ASCII 字符的字节值为 20 到 7E（十六进制）。
1	80B5	TCON：连接类型 "13 = UDP" 只允许创建被动连接。
1	80B6	TCON：SDT TCON_Param 的 CONNECTION_TYPE 参数存在参数分配错误。
1	80B7	TCON：连接描述数据块的以下参数之一出错： ● block_length ● local_tsap_id_len ● rem_subnet_id_len ● rem_staddr_len ● rem_tsap_id_len ● next_staddr_len 注：在 TCP 被动模式下执行 TCON 时，LOCAL_TSAP_ID_LEN 必须为“2”，且 REM_TSAP_ID_LEN 必须为“0”。
1	80B8	TCON：本地连接描述中的参数与参数 ID 不同。
1	80C3	TCON：所有连接资源都在使用。
1	80C4	临时通信错误： ● 此时无法建立连接 (TCON)。 ● TDISCON (TCON) 当前正在删除已组态连接。 ● 当前正在建立连接 (TDISCON)。 ● 接口正在接收新参数（TCON 和 TDISCON）。

TSEND 和 TRCV

说明

使用 PROFINET 开放式用户通信协议时，如果执行 TSEND 指令但不在远程设备上执行相应的 TRCV 指令，则 TSEND 指令可能无限期处于“繁忙状态”，等待 TRCV 指令接收数据。在这种状态下，TSEND 指令“繁忙”输出将置位，“状态”输出的值为“0x7002”。传输的数据大于 4096 字节时可能会出现这种情况。在下次执行 TRCV 指令时会解决这一问题。

表格 11-35 TSEND 和 TRCV 指令

LAD/FBD	SCL	说明
	"TSEND_DB"(req:=_bool_in_, ID:=_word_in_, len:=_udint_in_, done=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, data:=_variant_inout_);	TCP 和 ISO on TCP：TSEND 通过从 CPU 到伙伴站的通信连接发送数据。
	"TRCV_DB"(en_r:=_bool_in_, ID:=_word_in_, len:=_udint_in_, ndr=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, rcvd_len=>_udint_out_, data:=_variant_inout_);	TCP 和 ISO on TCP：TRCV 通过从伙伴站到 CPU 的通信连接接收数据。

[1] STEP 7 会在插入指令时自动创建 DB。

表格 11-36 TSEND 和 TRCV 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	TSEND：在上升沿启动发送作业。传送通过 DATA 和 LEN 指定的区域中的数据。
EN_R	IN	Bool	TRCV：允许 CPU 进行接收；EN_R = 1 时，TRCV 准备接收。处理接收作业。

参数和类型		数据类型	说明
ID	IN	CONN_OUC (Word)	指向相关连接的引用。ID 必须与本地连接描述信息内的相关参数 ID 相同。 值范围：W#16#0001 到 W#16#0FFF
LEN	IN	UInt	要发送 (TSEND) 或接收 (TRCV) 的最大字节数： ● 默认 = 0：DATA 参数确定要发送 (TSEND) 或接收 (TRCV) 的数据长度。 ● 特殊模式 = 65535：设置可变长度的数据接收 (TRCV)。
DATA	IN_OUT	Variant	指向发送 (TSEND) 或接收 (TRCV) 数据区的指针；数据区包含地址和长度。该地址引用 I 存储器、Q 存储器、M 存储器或 DB。
DONE	OUT	Bool	TSEND： ● 0：作业尚未开始或仍在运行。 ● 1：无错执行作业。
NDR	OUT	Bool	TRCV： ● NDR = 0：作业尚未开始或仍在运行。 ● NDR = 1：作业已成功完成。
BUSY	OUT	Bool	● BUSY = 1：作业尚未完成。无法触发新作业。 ● BUSY = 0：作业已完成。
ERROR	OUT	Bool	ERROR = 1：处理期间出错。STATUS 提供错误类型的详细信息
STATUS	OUT	Word	包括错误信息的状态信息。（请参见下表中的错误和状态条件代码。）
RCVD_LEN	OUT	Int	TRCV：实际接收到的数据量（以字节为单位）

说明

TSEND 指令需要通过 REQ 输入参数的上升沿来启动发送作业。然后，BUSY 参数在处理期间会设置为 1。发送作业完成时，将通过 DONE 或 ERROR 参数被设置为 1 并持续一个扫描周期进行指示。在此期间，将忽略 REQ 输入参数的上升沿。

TRCV 操作

TRCV 指令将收到的数据写入到通过以下两个变量指定的接收区：

- 指向区域起始位置的指针
- 如果不为 0 则为区域长度或 LEN 上提供的值

说明

LEN 参数的默认设置 (LEN = 0) 使用 DATA 参数来确定要传送的数据的长度。建议由 TSEND 指令传送的数据与 TRCV 指令的 DATA 参数大小相同。

如果使用 LEN 参数的默认设置且发送的句段数据必须小于 DATA 参数大小，请遵循以下原则。建议持续高 EN_R 位直到相应 TSEND 传送适当量的数据来填充 TRCV DATA 参数。如果 TSEND 传输的数据大小不等于 TRCV DATA 参数大小，那么 TRCV 会保持在忙碌状态（状态代码：7002），然而 EN_R 位为高直到从 TSEND 传输的数据全部大小等于 TRCVDATA 参数大小。如果 TRCV 的 EN_R 位为脉冲，则它需要执行与 TSEND 次数相同的脉冲来接收数据。

在数据大小等于 DATA 参数缓冲区大小之前，TRCV DATA 参数缓冲区不会显示已接收的新数据。

接收所有作业数据后，TRCV 会立即将其传送到接收区并将 NDR 设置为 1。

表格 11-37　将数据输入接收区

协议选项	将数据输入接收区	参数 “connection_type”	LEN 参数的值	RCVD_LEN 参数的值（字节）
TCP	特殊模式	B#16#11	65535	1 到 1472
TCP	指定长度的数据接收	B#16#11	0（推荐）或 1 到 8192，65535 除外	1 到 8192
ISO on TCP	特殊模式	B#16#12	65535	1 到 1472
ISO on TCP	协议控制	B#16#12	0（推荐）或 1 到 8192，65535 除外	1 到 8192

说明

特殊模式

使用 TCP 或 ISO on TCP 协议时可以存在“特殊模式”。用户通过将“65535”分配给 LEN 参数来设置特殊模式。接收区与 DATA 构成的区域相同。接收数据的长度将输出到参数 RCVD_LEN 中。接收数据块后，TRCV 会立即将数据写入接收区并将 NDR 设置为 1。

如果将数据存储在“优化”DB（仅符号访问）中，则只能接收数据类型为 Byte、Char、USInt 和 SInt 的数组中的数据。

说明

将包含“特殊模式”的 S7-300/400 STEP 7 项目导入 S7-1200 中

在 S7-300/400 STEP 7 项目中，通过将“0”分配给 LEN 参数来选择“特殊模式”。在 S7-1200 中，用户通过将“65535”分配给 LEN 参数来设置特殊模式。

如果将包含“特殊模式”的 S7-300/400 STEP 7 项目导入 S7-1200 中，则必须将 LEN 参数更改为“65535”。

TSEND 和 TRCV Error 和 Status 条件代码

ERROR	STATUS	说明
0	0000	• 发送作业无错完成 (TSEND) • 已接受新数据：在 RCVD_LEN 中显示已接收数据的当前长度 (TRCV)。
0	7000	• 无激活的作业处理 (TSEND) • 块未准备好接收 (TRCV)
0	7001	• 启动作业处理，正在发送数据：在执行此处理期间，操作系统访问 DATA 发送区中的数据 (TSEND)。 • 块准备接收，接收作业已激活 (TRCV)。
0	7002	• 后续指令执行（与 REQ 无关），正在处理作业：在执行此处理期间，操作系统访问 DATA 发送区中的数据 (TSEND)。 • 后续指令执行，正在处理接收作业：数据在执行此处理期间写入接收区。因此，错误可能导致接收区中的数据不一致 (TRCV)。
1	8085	• LEN 参数的值大于允许的最大值 (TSEND) 和 (TRCV)。 • 自第一次指令执行 (TRCV) 以来，LEN 或 DATA 参数发生变化。
1	8086	ID 参数不在允许的地址范围内。
1	8088	LEN 参数大于 DATA 中指定的存储区。
1	80A1	通信错误： • 尚未建立指定的连接（TSEND 和 TRCV）。 • 当前正在终止指定的连接。无法通过该连接执行传送或接收作业（TSEND 和 TRCV）。 • 正在重新初始化接口 (TSEND)。 • 接口正在接收新参数 (TRCV)。
1	80C3	内部缺乏资源：具有该 ID 的块正在一个具有不同优先级的组中处理。
1	80C4	临时通信错误： • 此时无法建立与通信伙伴的连接。 • 接口正在接收新参数设置或当前正在建立连接。

以太网连接协议

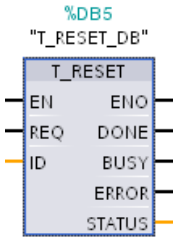
每个 CPU 都集成了一个支持标准 PROFINET 通信的 PROFINET 端口。TSEND_C、TRCV_C、TSEND 和 TRCV 指令均支持 TCP 和 ISO-on-TCP 以太网协议。

更多相关信息，请参见“设备配置：组态本地/伙伴连接路径 (页 812)”。

11.2.8.12 T_RESET（终止和重新建立现有连接）指令

使用指令“T_RESET”可终止并重新建立现有连接：

表格 11-38 T_RESET 指令

LAD/FBD	SCL	说明
	"T_RESET_DB"(req:=_bool_in_, id:=_word_in_, done=>_bool_out_, error=>_bool_out_, status=>_word_out_);	使用 T_RESET 指令终止并重新建立现有连接。

将保留连接的本地端点。如果符合以下条件，即自动生成本地端点：

- 连接已组态并装载到 CPU。
- 连接已由用户程序生成，例如通过调用指令“TCON (页 862)”。

无论连接使用的是 CPU 本地接口还是 CM/CP 接口，所有连接类型都可以执行“T_RESET”指令。例外情况是在使用 TCP 的特殊模式下进行数据传输的连接，因为此类连接无法使用连接 ID 引用。

使用 REQ 参数调用“T_RESET”指令后，通过参数 ID 指定的连接将终止，并且必要时，数据发送和接收缓冲区会清空。取消连接的同时会取消所有正在进行的数据传输。因此，如果正在传输数据，便存在数据丢失的风险。随后，定义为主动连接伙伴的 CPU 将自动尝试恢复中断的通信连接。因此，无需调用指令“TCON (页 862)”重新建立通信连接。

输出参数 DONE、BUSY 和 STATUS 指示作业的状态。

参数的数据类型

下表列出了“T_RESET”指令的参数：

参数	声明	数据类型	存储区	说明
REQ	Input	BOOL	I、Q、M、D、L、T、C 或常数	控制参数 REQUEST 启动用于终止 ID 所指定的连接的作业。在上升沿启动作业。
ID	Input	CONN_OUC (WORD)	L、D 或常数	对将终止的被动方连接的引用。ID 必须与本地连接描述中的相应参数 ID 相同。 值范围：W#16#0001 到 W#16#0FFF
DONE	Output	BOOL	I、Q、M、D、L	状态参数 DONE ● 0：作业未启动，或者仍在执行之中。 ● 1：已成功执行作业。
BUSY	Output	BOOL	I、Q、M、D、L	状态参数 BUSY ● 0：作业已完成。 ● 1：作业尚未完成。
ERROR	Output	BOOL	I、Q、M、D、L	状态参数 ERROR ● 0：未出错。 ● 1：处理时出错。STATUS 参数提供错误类型的详细信息
STATUS	Output	WORD	I、Q、M、D、L	状态参数 STATUS 错误信息（请参见“STATUS 参数” 表）。

STATUS 参数

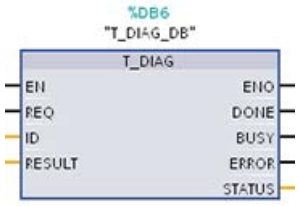
错误位	STATUS * (W#16#...)	说明
0	0000	无错误。
0	0001	尚未建立连接。
0	7001	已启动连接终止。
0	7002	正在终止连接。
1	8081	ID 参数中指定了未知连接。

11.2.8.13 T_DIAG（检查连接状态和读取信息）指令

“T_DIAG”指令可检查连接的状态并读取有关该连接本地端点的更多信息：

表格 11-39 T_DIAG 指令

LAD/FBD	SCL	说明

	`"T_DIAG_DB"(`
	`req:=_bool_in_,`
	`id:=_word_in_,`
	`done=>_bool_out_,`
	`error=>_bool_out_,`
	
	`status=>_dword_out_);`
		使用“T_DIAG”指令检查连接的状态并读取有关该连接本地端点的更多信息。

“T_DIAG”指令的工作方式如下：

- 连接由 ID 参数引用。可以同时读取连接编辑器中组态的连接端点和已编程的连接端点（例如，使用“TCON”指令）。
  由于此过程中不生成任何连接 ID，因此无法诊断临时连接端点（例如连接到工程师站时创建的端点）。
- 读取的连接信息存储在参数 RESULT 引用的结构中。
- 输出参数 STATUS 指示是否可以读取该连接信息。参数 RESULT 中的结构的连接信息仅在“T_DIAG”指令完成且 STATUS = W#16#0000、ERROR = FALSE 时有效。
  如果发生错误，将无法评估连接信息。

可能的连接信息

“TDiag_Status”结构可用于读取参数 RESULT 中的连接信息。TDiag_Status 结构中仅包含有关连接端点的最重要信息（例如，所使用的协议、连接状态以及发送和接收的数据字节数）。

下面介绍 TDiag_Status 结构的结构和参数（请参见“TDIAG_Status 结构”表）。

参数的数据类型

下表列出了“T_DIAG”指令的相关参数：

参数	声明	数据类型	存储区	说明
REQ	Input	BOOL	I、Q、M、D、L、T、C 或常数	存在上升沿时启动指令，检查参数 ID 中指定的连接。
ID	Input	CONN_OUC (WORD)	L、D 或常数	引用已分配的连接。 取值范围：W#16#0001 到 W#16#0FFF
RESULT	InOut	VARIANT	D	指向存储连接信息的结构的指针。可以在参数 RESULT 中使用结构 TDiag_Status（有关说明，请参见“TDIAG_Status 结构”表）。
DONE	Output	BOOL	I、Q、M、D、L	状态参数： • 0：指令尚未开始或仍在执行。 • 1：指令已无错执行。
BUSY	Output	BOOL	I、Q、M、D、L	状态参数： • 0：指令尚未开始或已完成。 • 1：指令尚未完成。无法启动新作业。
ERROR	Output	BOOL	I、Q、M、D、L	状态参数： • 0：无错误。 • 1：出现错误。
STATUS	Output	WORD	I、Q、M、D、L	指令的状态

参数 BUSY、DONE 和 ERROR

可以使用 BUSY、DONE、ERROR 和 STATUS 参数检查“T_DIAG”指令的执行状态。参数 BUSY 表示作业正在执行。可使用 DONE 参数检查是否已成功执行指令。如果执行“T_DIAG”过程中出错，将置位参数 ERROR。

下表列出了参数 BUSY、DONE 和 ERROR 之间的关系：

BUSY	DONE	ERROR	说明
1	-	-	正在处理指令。
0	1	0	指令已成功执行。仅在这种情况下，RESULT 引用结构中的数据才有效。
0	0	1	指令完成，但存在错误。错误原因通过参数 STATUS 输出。
0	0	0	尚未分配新指令。

STATUS 参数

下表列出了 STATUS 参数值的含义：

错误位	STATUS * (W#16#...)	说明
0	0000	指令“T_DIAG”已成功执行。可对 RESULT 参数所引用结构中的数据进行评估。
0	7000	未激活任何指令处理。
0	7001	已启动指令处理。
0	7002	正在读取连接信息（REQ 参数不相关）。
1	8086	ID 参数值超出有效范围（W#16#0001 到 W#16#0FFF）。
1	8089	参数 RESULT 指向无效数据类型（仅限结构 TDIAG_Status 和 TDIAG_StatusExt）。
1	80A3	参数 ID 引用了不存在的连接端点。通过编程的连接，调用“TDISCON”指令后仍可能发生此错误。
1	80C4	内部错误。连接端点暂时不可访问。

TDIAG_Status 结构

下表详细介绍了 TDIAG_Status 结构的形式。仅当指令已执行且没有错误时，各个元素的值才有效。如果发生错误，参数的内容不会改变：

名称	数据类型	说明
TDIAG_Status 结构包含下列参数：		
InterfaceID	HW_ANY	CPU 或 CM/CP 的接口 ID (LADDR)。
ID	CONN_OUC	诊断的连接 ID。成功调用后，此元素的值与“T_DIAG”指令的参数 ID 相同。

名称	数据类型	说明
ConnectionType	BYTE	用于连接的协议类型： • 0x01：未使用。 • ... • 0x0B：TCP 协议 (IP_v4) • 0x0C：ISO-on-TCP 协议 (RFC1006) • 0x0D：TCP 协议 (DNS) • 0x0E：拨入协议 • 0x0F：WDC 协议 • 0x10：SMTP 协议 • 0x11：TCP 协议 • 0x12：TCP 和 ISO-on-TCP 协议 (RFC1006) • 0x13：UDP 协议 • 0x14：保留 • 0x15：PROFIBUS 总线访问协议 (FDL) • 0x16：ISO 8073 传输协议（ISO 原生） • ... • 0x20：SMTP 或 SMTPS 协议 - 基于 IPv4 • 0x21：SMTP 或 SMTPS 协议 - 基于 IPv6 • 0x22：SMTP 或 SMTPS 协议 - 基于 FQDN (Fully Qualified Domain Name) • ... • 0x70：S7 连接 • 其它：保留
ActiveEstablished	BOOL	• FALSE：本地，被动连接端点 • TRUE：本地，主动连接端点
State	BYTE	连接端点的当前状态 • 0x00：未使用。 • 0x01：连接终止。临时状态，例如调用“T_RESET”指令后的状态。系统随后将自动尝试重新建立连接。 • 0x02：主动连接端点正在尝试与远程通信伙伴建立连接。 • 0x03：被动连接端点正在等待与远程通信伙伴建立连接。 • 0x04：连接已建立。 • 0x05：正在终止连接。原因可能是已调用 "T_RESET" 或 "T_DISCON" 指令。也可能是因为协议错误或线路中断。 • 0x06..0xFF：未使用。

名称	数据类型	说明
Kind	BYTE	连接端点的模式： • 0x00：未使用。 • 0x01：已组态并装载到 CPU 中的静态连接。 • 0x02：已组态并装载到 CPU 中的动态连接（当前不支持）。 • 0x03：使用 "TCON" 指令在用户程序中生成的已编程连接。调用 “TDISCON”或转换为 CPU STOP 状态导致连接端点损坏。 • 0x04：例如，由工程师站 (ES) 或操作员站 (OS) 建立的临时动态连接（由于无 ID，因此目前无法诊断该连接类型）。 • 0x05..0xFF：未使用。
SentBytes	UDINT	发送的数据字节数。
ReceivedBytes	UDINT	接收的数据字节数。

11.2.8.14 TMAIL_C（通过 CPU 的以太网接口发送电子邮件）指令

概述

使用指令 TMAIL_C 通过 S7-1200 CPU 的以太网接口发送电子邮件。

指令 TMAIL_C 有两个功能：

- 通过 CPU 接口发送电子邮件（仅使用无 SSL 的 SMTP 协议）
- 通过 CP 接口发送电子邮件（使用无 SSL 的 SMTP 协议或带 SSL 的 SMTP 协议）。如果想使用 SSL 功能，必须将 TMAIL_C 输入参数 CERTINDEX 设置为 1 且使用 CP 接口。此外，CP 的证书存储位置必须存储有有效证书。

如果使用 TMail_V4 系统数据类型 (SDT)，则只能通过 S7-1200 CPU 的集成以太网端口在不安全的条件下发送电子邮件。通过 S7-1200 CPU 集成以太网端口发送电子邮件时，不支持使用 TMail_V6 SDT。

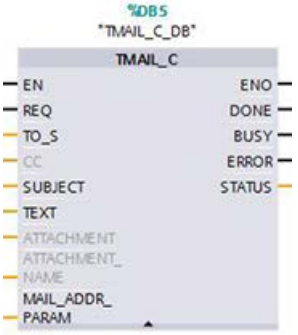
如果使用 TMail_FQDN SDT，则只能通过通信处理器 (CP) 发送电子邮件。不能将 TMail_FQDN SDT 用于通过集成的 S7-1200 CPU 以太网端口发送电子邮件。

从 TMAIL_C V6.0 或更高版本以及 S7-1200 CPU 固件版本 V4.4 或更高版本开始，可以使用 TMAIL_C 指令通过 S7-1200 CPU 的集成以太网端口基于安全通信发送电子邮件。使用 TMail_V4_SEC 或 TMail_QDN_SEC SDT 通过 MAIL_ADDR_PARAM 参数定义发送过程所需的数据。

只有在组态了硬件后且网络架构允许与邮件服务器建立通信连接时，才可使用 TMAIL_C 指令。

表格 11-40 TMAIL_C 指令

LAD/FBD	SCL	说明

	`"TMAIL_C_DB"(`
	`req:=_bool_in_,`
	`to_s:=_string_in_,`
	`cc:=_string_in_,`
	`subject:=_string_in_,`
	`text:=_string_in_,`
	`attachment:=_variant_in_,`
	`attachment_name:=_string_in_,`
	`mail_addr_param:=_string_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_);`
		TMAIL_C 指令通过 S7-1200 CPU 的以太网接口发送电子邮件。

[1] STEP 7 会在插入指令时自动创建 DB。

使用以下参数定义电子邮件内容和连接数据：

- 使用参数 TO_S 和 CC 定义收件人地址。
- 使用参数 SUBJECT 和 TEXT 定义电子邮件的内容。
- 在 ATTACHMENT 和 ATTACHMENT_NAME 参数中使用 VARIANT 指针定义附件。
- TMAIL_C 指令使用以下 SDT 通过 MAIL_ADDR_PARAM 参数定义连接数据，并执行寻址以及对邮件服务器进行验证：
  - TMail_V4
  - TMail_V4_SEC
  - TMail_QDN_SEC

  如果使用固件版本为 V4.3 或以下版本的 S7-1200 CPU 的接口，则必须使用 TMail_V4 SDT。此时，只能使用 SMTP 发送电子邮件。
- 参数 REQ 出现“0”至“1”的上升沿时，将启动电子邮件的发送任务。
- TMAIL_C 指令通过“BUSY”、“DONE”、“ERROR”和“STATUS”输出参数指示作业状态。

不可使用 TMAIL_C 指令直接发送 SMS。邮件服务器是否可将电子邮件以 SMS 形式转发取决于电信提供商。

指令的操作

TMAIL_C 指令将异步执行。即，可通过多次调用执行这一指令。调用“TMAIL_C”指令时必须指定实例。

在下列情况下，与邮件服务器的连接将丢失：

- CPU 在 TMAIL_C 指令处于激活状态时切换到 STOP 模式。
- 工业以太网总线出现通信问题。这种情况下，电子邮件的传输将中断，并且不会到达其接收方。

成功执行指令并发送电子邮件后连接也将被取消。

注意
更改用户程序 仅在下列情况下，才能更改直接影响 TMAIL_C 调用的用户程序部分： • CPU 处于“STOP”模式。 • 未发送任何电子邮件（REQ = 0 且 BUSY = 0）。 这具体是指删除和替换含有 TMAIL_C 调用或者 TMAIL_C 实例调用的程序块。 如果忽略这一限制，可能导致占用连接资源。通过工业以太网使用 TPC/IP 通信功能时，自动化系统可能切换到某种不确定的状态。 传输更改之后，需要对 CPU 执行一次暖启动或冷启动。

数据一致性

TMAIL_C 指令在运行时会使用 TO_S、CC、SUBJECT、TEXT、ATTACHMENT 和 MAIL_ADDR_PARAM 参数，也就是说只有在作业完成后 (BUSY = 0) 才能更改这些参数。

SMTP 验证

此处，验证指身份核实程序，例如，使用密码查询。

如果使用的是 S7-1200 CPU 接口，指令 TMAIL_C 将支持大部分邮件服务器所需的 SMTP 授权程序 AUTH-LOGIN。有关邮件服务器授权程序的信息，请参见邮件服务器的使用手册或者 Internet 服务提供商的网站。

- 在能够使用授权程序 AUTH-LOGIN 前，需要为指令 TMAIL_C 提供登录邮件服务器所需的用户名。该用户名相当于用户在邮件服务器上建立邮箱帐号时所使用的用户名。其通过参数 UserName 传送到参数 MAIL_ADDR_PARAM 结构中。
  如果未在参数 MAIL_ADDR_PARAM 中指定用户名，则不使用 AUTH-LOGIN 验证程序。此时，电子邮件将以无授权方式发送。
- 若要登录，指令 TMAIL_C 还需要相关密码。该密码相当于建立邮箱帐号时指定的密码。其通过参数 PassWord 传送到参数 MAIL_ADDR_PARAM 结构中。

参数的数据类型

下表列出了 TMAIL_C 指令的参数：

参数	声明	数据类型	存储区	说明
REQ	Input	BOOL	I、Q、M、D、L、T、C 或常数	控制参数 REQUEST：上升沿时激活电子邮件的发送任务。
TO_S (页 904)	Input	STRING	D	收件人地址 最大长度为 180 个字符（字节）的 STRING。 有关电子邮件地址格式，请参见参数说明中的示例。
CC (页 904)	Input	STRING	D	CC 收件人地址（可选） 最大长度为 180 个字符（字节）的 STRING。 与 TO_S 参数的电子邮件地址格式相同。如果在此处分配一个空字符串，则该指令不会将电子邮件发送给抄送收件人。
SUBJECT	Input	STRING	D	电子邮件的主题 最大长度为 180 个字符（字节）的 STRING。

参数	声明	数据类型	存储区	说明
TEXT	Input	STRING	D	电子邮件的文本（可选） 最大长度为 180 个字符（字节）的 STRING。如果为此参数分配一个空字符串，则该指令将发送不带文本的电子邮件。
ATTACHMENT	Input	VARIANT	D	电子邮件附件（可选） 对最大长度为 64 KB 的字符/字节/字/双字/字符串字段（ArrayOfChar, ArrayOfByte、ArrayOfWord、ArrayOfDWord, or String）的引用。 注：如果未对 ATTACHMENT 参数赋值或为其分配了空字符串，则该指令将发送不带附件的电子邮件。
ATTACHMENT_NAME	Input	VARIANT	D	电子邮件附件名称（可选） 引用最大长度为 50 个字符（字节）的字符串来定义附件的文件名。如果在该参数中分配空字符串，将发送文件名为“attachment.bin”的电子邮件附件。 使用 AttachmentName 参数，可以指定通信伙伴接收到电子邮件时显示的附件名称。 TMail_FileReference SDT 自动将 FileName 参数用于 AttachmentName 参数。 如果使用 TMail_FileReference SDT，AttachmentName 参数则不适用。将其留空。 使用 TMail_FileReference SDT 时，如果在 AttachmentName 参数中输入数据，则 TMAIL_C 指令会生成错误。有关更多信息，请参见“错误条件代码，背景数据块的 SFB_STATUS 参数”。

参数	声明	数据类型	存储区	说明
MAIL_ADDR_PARAM (页 894)	Input	VARIANT	D	电子邮件服务器的连接参数和地址 使用 TMail_V4、TMail_FQDN、TMail_V4_SEC 或 TMail_QDN_SEC SDT 可定义连接参数（见参数说明）。
DONE (页 905)	Output	BOOL	I、Q、M、D、L	状态参数 • DONE = 0：作业未启动，或者仍在执行之中。 • DONE = 1：作业已正确完成。
BUSY (页 905)	Output	BOOL	I、Q、M、D、L	状态参数 • BUSY=0：TMAIL_C 的处理已停止。 • BUSY = 1：电子邮件传输尚未完成。
ERROR (页 905)	Output	BOOL	I、Q、M、D、L	状态参数 • ERROR = 0：未发生错误。 • ERROR = 1：执行过程中发生错误。有关错误类型的详细信息，请参见 STATUS。
STATUS (页 908)	Output	WORD	I、Q、M、D、L	状态参数 指令 TMAIL_C 的返回值或错误信息（见参数说明）。

说明

可选参数

仅当可选参数 CC、TEXT 和 ATTACHMENT 包含长度大于 0 的字符串时，才会通过电子邮件发送相应的参数。

MAIL_ADDR_PARAM 参数

在 MAIL_ADDR_PARAM 参数中，可使用 TMail_v4、TMail_FQDN、TMail_V4_SEC 或 TMail_QDN_SEC 结构定义发送电子邮件的连接，并保存电子邮件服务器地址和登录详情。

在 MAIL_ADDR_PARAM 参数中使用的系统数据类型 (SDT) 取决于电子邮件服务器的寻址格式：

- TMail_V4：根据 IP 地址 (IPv4) 寻址
- TMail_FQDN：根据完全限定域名 (FQDN) 寻址
- TMail_V4_SEC：根据 IP 地址 (IPv4) 进行安全寻址
- TMail_QDN_SEC：根据完全限定域名 (FQDN) 进行安全寻址

可使用的 SDT 取决于通过 InterfaceId 参数寻址的接口。如果要将 TMAIL_C 指令用于 CPU 的内部接口，必须在 MAIL_ADDR_PARAM 参数中使用 TMail_V4、 TMail_V4_SEC 或 TMail_QDN_SEC SDT。

如果使用 TMail_FQDN SDT，则只能通过通信处理器 (CP) 发送电子邮件。不能将 TMail_FQDN SDT 用于通过集成的 S7-1200 CPU 以太网端口发送电子邮件。

表格 11-41 **TMail_V4：通过 IP 地址 (IPv4) 寻址邮件服务器**

参数	数据类型	说明
TMail_v4	Struct	
InterfaceId	LADDR	以太网接口的硬件标识符
ID	CONN_OUC	连接 ID
ConnectionType	BYTE	连接类型。选择 16#20 作为 IPv4 的连接类型。
ActiveEstablished	BOOL	建立主动/被动连接。CPU 始终为 SMTP 客户端。
CertIndex	BYTE	=0：使用了 SMTP (**S**imple **M**ail **T**ransfer **P**rotocol)。如果是通过 S7-1200 CPU 发送电子邮件，则必须使用 SMTP。
WatchDogTime	TIME	执行看门狗。使用该参数定义发送操作的最长执行时间。该值用于确定 TMAIL_C 指令保持执行状态的时长，超过该时间后将超时并且 TMAIL_C 指令执行结束。 从 TMAIL_C 版本 V6.0 开始，WatchDogTime 值现在可以为零，这表明 TMAIL_C 指令将在执行期间禁用定时器。仍可为 WatchDogTime 输入非零值以提高 TMAIL_C 指令执行的精确度。 注：如果连接速度较慢，则连接的建立时间可能比较长（约 1 分钟）。指定参数 WatchDogTime 时，必须为连接的建立预留足够的时间。
MailServerAddress	IP_v4	邮件服务器的 IP 地址。IPv4 的格式如下所示：XXX.XXX.XXX.XXX（十进制）。 示例：192.142.131.237
UserName	STRING[254]	邮件服务器登录名
PassWord	STRING[254]	邮件服务器密码
From	EMAIL_ADDR	电子邮件的发送方地址，由以下两个 STRING 参数定义。例如，“myname@mymailserver.com”。

参数		数据类型	说明
	LocalPartPlusAtSign	STRING[64]	发送方地址的本地部分，包括 @ 符号。示例：“myname@”。
	FullQualifiedDomainName	STRING[254]	邮件服务器的 Fully Qualified Domain Name（缩写为 FQDN）。示例：“mymailserver.com”。

表格 11-42 **TMail_FQDN *：通过 FQDN** 对邮件服务器寻址

参数		数据类型	说明
TMail_FQDN		Struct	
	InterfaceId	LADDR	以太网接口的硬件标识符
	ID	CONN_OUC	连接 ID
	ConnectionType	BYTE	连接类型。选择 16#22 作为 FQDN 的连接类型。
	ActiveEstablished	BOOL	建立主动/被动连接。通信处理器 (CP) 始终是 SMTP 客户端。
	CertIndex	BYTE	=0：使用 SMTP（**S**imple **M**ail **T**ransfer **P**rotocol，简单邮件传输协议）。
	WatchDogTime	TIME	执行看门狗。使用该参数定义发送操作的最长执行时间。 注：如果连接速度较慢，则连接的建立时间可能比较长（约 1 分钟）。指定参数 WatchDogTime 时，必须为连接的建立预留足够的时间。
	MailServerAddress	STRING[254]	邮件服务器的 FQDN（**F**ully **Q**ualified **D**omain **N**ame，完全限定域名）。使用 FQDN 对邮件服务器进行寻址。 示例：“www.mymailserver.com.”
	UserName	STRING[254]	邮件服务器登录名
	PassWord	STRING[254]	邮件服务器密码
	From	Struct	电子邮件的发送方地址，由以下两个 STRING 参数定义。例如，“myname@mymailserver.com”。

参数		数据类型	说明
	LocalPartPlusAtSign	STRING[64]	发送方地址的本地部分，包括 @ 符号。示例：“myname@”。
	FullQualifiedDomainName	STRING[254]	邮件服务器的完全限定域名（缩写为 FQDN）。例如：“mymailserver.com”。

* 如果使用 TMail_FQDN SDT，则只能通过通信处理器 (CP) 发送电子邮件。不能将 TMail_FQDN SDT 用于通过集成的 S7-1200 CPU 以太网端口发送电子邮件。

表格 11-43 **TMail_V4_SEC：通过 IP 地址 (IPv4) 寻址邮件服务器**

参数	数据类型	说明
TMail_V4_SEC	Struct	

参数	数据类型	说明
InterfaceId	LADDR	以太网接口值硬件标识符的取值范围： ● 0（新）：操作系统自行选择一个适合的集成端口。 ● 要使用的集成端口的硬件标识符。
ID	CONN_OUC	引用该连接：值范围： ● 0（新）：操作系统从内部范围选择一个自由连接 ID。 ● 1 到 4095：将使用连接 ID
ConnectionType	BYTE	连接类型。选择 16#20 作为 IPv4 的连接类型。
ActiveEstablishment	BOOL	建立主动/被动连接。CPU 始终为 SMTP 客户端。
WatchDogTime	TIME	执行看门狗。使用该参数定义发送操作的最长执行时间。该值用于确定 TMAIL_C 指令保持执行状态的时长，超过该时间后将超时并且 TMAIL_C 指令执行结束。 从 TMAIL_C 版本 V6.0 开始，WatchDogTime 值现在可以为零，这表明 TMAIL_C 指令将在执行期间禁用定时器。仍可为 WatchDogTime 输入非零值以提高 TMAIL_C 指令执行的精确度。 注：如果连接速度较慢，则连接的建立时间可能比较长（约 1 分钟）。指定参数 WatchDogTime 时，必须为连接的建立预留足够的时间。
MailServerAddress	IP_V4	电子邮件服务器的 IP 地址（IPv4 格式）：XXX.XXX.XXX.XXX（小数位） 示例：192142131237
UserName	STRING[254]	用户名 当用户访问电子邮件收件箱时，需向电子邮件提供商提供“用户名”来验证用户身份，以证明该收件箱归其所有。
PassWord	STRING[254]	用户密码 当用户访问电子邮件收件箱时，需向电子邮件提供商提供“密码”来验证用户身份，以证明该收件箱归其所有。
From	EMAIL_ADDR	电子邮件的发送方地址，由以下两个 STRING 参数定义。例如：“myname@mymailserver.com”

参数		数据类型	说明
	LocalPartPlusAtSign	STRING[64]	发送方地址的本地部分，包括 @ 符号。示例：“myname@”
	FullQualifiedDomainName	STRING[254]	邮件服务器的完全限定域名（缩写为 FQDN）。例如：“mymailserver.com”
RemotePort		UINT	邮件服务器的 TCP 端口
ActivateSecureConn		BOOL	• 1：安全 SMTP 连接 • 0：SMTP 连接（非安全连接）。此时，以下参数不相关。
ExtTLSCapabilities		BYTE	当前未使用
TLSServerCertRef		UDINT	对电子邮件服务器中 X.509 V3 (CA) 证书的引用，用于 TLS 客户端对 TLS 服务器进行认证

表格 11-44 **TMail_QDN_SEC：通过 FQDN** 对邮件服务器寻址

参数	数据类型	说明
TMail_QDN_SEC	Struct	

参数	数据类型	说明
InterfaceId	LADDR	以太网接口值硬件标识符的取值范围： ● 0（新）：操作系统自行选择一个适合的集成端口。 ● 要使用的集成端口的硬件标识符。
ID	CONN_OUC	引用该连接：值范围： ● 0（新）：操作系统从内部范围选择一个自由连接 ID。 ● 1 到 4095：要使用的连接 ID
ConnectionType	BYTE	连接类型。选择 16#22 作为 FQDN 的连接类型。
ActiveEstablishment	BOOL	建立主动/被动连接。CPU 始终为 SMTP 客户端。
WatchDogTime	TIME	执行看门狗。使用该参数定义发送操作的最长执行时间。该值用于确定 TMAIL_C 指令保持执行状态的时长，超过该时间后将超时并且 TMAIL_C 指令执行结束。 从 TMAIL_C 版本 V6.0 开始，WatchDogTime 值现在可以为零，这表明 TMAIL_C 指令将在执行期间禁用定时器。仍可为 WatchDogTime 输入非零值以提高 TMAIL_C 指令执行的精确度。 注：如果连接速度较慢，则连接的建立时间可能比较长（约 1 分钟）。指定参数 WatchDogTime 时，必须为连接的建立预留足够的时间。
MailServerQDN	STRING[254]	邮件服务器的 FQDN（Fully Qualified Domain Name，完全限定域名）。使用完全限定域名对邮件服务器进行寻址，该域名必须以“.”结尾。 示例：“www.mymailserver.com.”
UserName	STRING[254]	用户名 当用户访问电子邮件收件箱时，需向电子邮件提供商提供“用户名”来验证用户身份，以证明该收件箱归其所有。
PassWord	STRING[254]	用户密码 当用户访问电子邮件收件箱时，需向电子邮件提供商提供“密码”来验证用户身份，以证明该收件箱归其所有。
From	EMAIL_ADDR	电子邮件的发送方地址，由以下两个 STRING 参数定义。例如：“myname@mymailserver.com”

参数		数据类型	说明
	LocalPartPlusAtSign	STRING[64]	发送方地址的本地部分，包括 @ 符号。示例：“myname@”
	FullQualifiedDomainName	STRING[254]	邮件服务器的完全限定域名（缩写为 FQDN）。例如：“mymailserver.com”
RemotePort		UINT	邮件服务器的 TCP 端口
ActivateSecureConn		BOOL	• 1：安全 SMTP 连接 • 0：SMTP 连接（非安全连接）。此时，以下参数不相关。
ExtTLSCapabilities		BYTE	当前未使用
TLSServerCertRef		UDINT	对电子邮件服务器中 X.509 V3 (CA) 证书的引用，用于 TLS 客户端对 TLS 服务器进行认证

TO_S 和 CC 参数

对于 TMAIL_C 指令 V6.0 之前的版本和 S7-1200 CPU 固件 V4.4，输入 TO_S 和 CC 参数时，以下规则适用：

- 在各地址前输入空格和开尖括号“<”。
- 在各地址后输入闭尖括号“>”。
- 在 TO 和 CC 地址之间输入逗号。

以下为此类 TO_S 和 CC 参数字符串示例：

- <wenna@mydomain.com>, <ruby@mydomain.com>
- <admin@mydomain.com>, <judy@mydomain.com>

从 TMAIL_C 指令版本 V6.0 或更高版本和 S7-1200 CPU 固件 V4.4 或更高版本开始，输入参数时，仅以下规则适用：

- 在 TO 和 CC 地址之间输入逗号或分号。

以下为此类 TO_S 和 CC 参数字符串示例：

- wenna@mydomain.com, ruby@mydomain.com
- admin@mydomain.com, judy@mydomain.com

由于运行时间和存储空间方面的原因，TMAIL_C 指令不对 TO_S 或 CC 参数执行任何语法检查。

参数 DONE、BUSY 和 ERROR

如果输出参数 BUSY 的状态由“1”变为“0”，则输出参数 DONE、BUSY 和 ERROR 均仅显示一个周期。

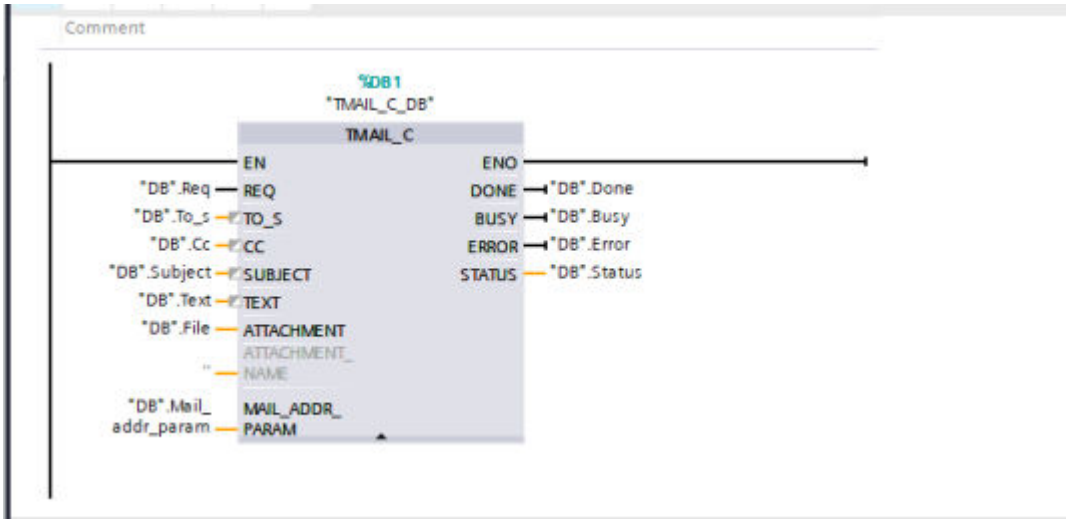
下表列出了 DONE、BUSY 和 ERROR 之间的关系。使用该表，可以确定指令“TMAIL_C”的当前状态，以及电子邮件发送完成的时间。

DONE	BUSY	ERROR	说明
0	1	0	正在处理作业。
1	0	0	作业已成功完成。
0	0	1	作业以出错而结束。出错原因可在 STATUS (页 908) 参数中找到。
0	0	0	没有为 TMAIL_C 指令分配（新）作业。

使用电子邮件附件发送数据日志、配方和用户文件

从 TMAIL_C V6.0 或更高版本以及 S7-1200 CPU 固件版本 V4.4 或更高版本开始，可以使用 TMAIL_C 指令的 Attachment 参数添加和访问 TMail_FileReference SDT。然后可以在 SIMATIC 存储卡 (SMC) 上寻址文件路径。如果没有存储卡，仍然可以访问 PLC 内部装载存储器上的配方和数据日志目录。

TMail_FileReference SDT 自动将 FileName 参数用于 AttachmentName 参数。



TMail_FileReference SDT

TMail_FileReference SDT 由两个参数组成，这两个参数都是 SIMATIC 字符串：

- 在 DirectoryPath 参数中，可以寻址目标文件所在的目录。
- FileName 参数表示希望在目录中访问的文件名和文件扩展名（如适用），该参数由之前的参数指定。

寻址 DirectoryPath 参数时，TMAIL_C 指令仅限于访问 DataLogs、Recipes 或 UserFiles 目录。也可以在这些目录中寻址子目录。

除了上述基本目录限制以外，下面的“文件寻址规则”部分还描述了寻址子目录和文件名时必须遵守的规则。

TMAIL_C 指令不会限制可发送文件的大小。构建程序时请务必注意这一点。

文件寻址规则

通过 TMAIL_C 指令使用 TMail_FileReference SDT 寻址文件时，必须遵守特定规则以确保精确寻址。以下各小节分别描述了 DirectoryPath 和 FileName 的特性。通常，以下内容适用于 TMail_FileReference SDT 中的两个参数，如果未遵守这些规则，将导致 TMAIL_C 指令产生错误状态：

- 不能将空字符串用作子目录或文件名。
- 不能在参数字符串中使用任何 ASCII 控制字符（十六进制范围：0x00 到 0x1F）。
- 不能在任何参数字符串中使用以下保留字符：
  - <（小于）
  - >（大于）
  - :（冒号）
  - "（双引号）
  - /（斜杠）（允许在 DirectoryPath 中使用该字符作为分隔符）
  - \（反斜杠）
  - |（竖线或管线）
  - ?（问号）
  - *（星号）
- 任何子目录或文件名都不能以空格或句点结尾。

DirectoryPath

在 SDT 的 DirectoryPath 参数中输入所需目录时，请牢记以下几点。根目录是由 PLC 的固件逻辑推断出的，用户不必了解。可以选择输入前斜杠和后斜杠 (/)。如果用户未输入任何斜杠，固件将自动在路径中添加斜杠。因此，以下 DirectoryPath 条目均有效：

- /DataLogs/
- /DataLogs
- DataLogs/
- DataLogs

也可以使用以下格式访问比基目录更深层的路径："/DataLogs/dir1/"，其中 (/) 表示新目录。最大深度为八层，包括根目录。

除了"文件寻址规则"部分所述的规则之外，还必须注意，严禁使用相对路径。因此，"/DataLogs/"是非法的路径名。此外，还禁止在任何子目录部分使用句点来表示当前目录（例如"/DataLogs/"）。

FileName

使用 SDT 的 FileName 参数时，请牢记"文件寻址规则"部分所述的规则。除此之外，还必须注意，PLC 的操作系统将文件名限制为少于 60 个字符。如果尝试寻址大于或等于 60 个字符的文件名，TMAIL_C 指令将中止操作并产生错误。

除了这些例外情况，可以附加任何大小或采用任何扩展名的文件。寻址的文件可能包含也可能不包含文件扩展名。

错误条件代码

STATUS 参数

下表列出了 STATUS 参数处“TMAIL_C”的返回值：

返回值 STATUS* (W#16#...)：	说明	注意
0000	TMAIL_C 已成功执行完毕。	TMAIL_C 无错完成并不表示发送的电子邮件一定能到达目标地址。 收件人地址不正确并不会导致 TMAIL_C 指令生成状态错误。这种情况下，不能保证电子邮件能发送至其它收件人，即使这些收件人地址正确无误也是如此。
7001	TMAIL_C 处于激活状态 (BUSY = 1)。	首次调用：作业已触发。
7002	TMAIL_C 处于激活状态 (BUSY = 1)。	中间调用：作业已激活。
8xxx	TMAIL_C 的执行已完成，且存在一个内部调用通信指令的错误代码。	相关详细信息，请参见 TCON、TDISCON、TSEND 和 TRCV (页 862)通信指令的 STATUS 参数的描述。
8009	内部函数错误	内部函数返回一个错误。有关详细信息，请参见背景数据块的 SFB_STATUS 参数。下面列出了可能的值。
8010	连接建立期间出错	在实例数据块的参数 SFB_STATUS 中可以找到有关评估的更多信息。参数 SFB_STATUS 中显示的错误代码将在 TCON (页 862) 指令的 STATUS 参数说明中进行解释。
8011	发送数据时出错	在实例数据块的参数 SFB_STATUS 中可以找到有关评估的更多信息。参数 SFB_STATUS 中显示的错误代码在 TSEND (页 862) 指令的参数 STATUS 描述中有相应说明。

返回值 STATUS* (W#16#...)：	说明	注意
8012	接收数据时出错	在实例数据块的参数 SFB_STATUS 中可以找到有关评估的更多信息。参数 SFB_STATUS 中显示的错误代码将在 TRCV (页 862) 指令的 STATUS 参数说明中进行解释。
8013	连接建立期间出错	在实例数据块的参数 SFB_STATUS 中可以找到有关评估的更多信息。参数 SFB_STATUS 中显示的错误代码将在 TCON (页 862) 和 TDISCON (页 862) 指令的 STATUS 参数说明中进行解释。
8014	无法建立连接。	输入的邮件服务器 IP 地址（MailServerAddress (页 894)）可能不正确，或者连接建立时间间隔（WatchDogTime (页 894)）过短。也有可能是因为 CPU 没有网络连接，或者 CPU 组态不正确。
8015	MAIL_ADDR_PARAM 的数据类型不正确	有效数据类型只有系统数据类型（结构）Tmail_v4 和 TMail_FQDN。
8016	参数 ATTACHMENT 的数据类型不正确	以下列表列出了有效的数据类型： • ArrayOfChar • ArrayOfByte • ArrayOfWord • ArrayOfDWord • String 注： ArrayOfChar 和 String 数据类型仅适用于 TMAIL_C 指令版本 V5.0 或更高版本。
8017	参数 ATTACHMENT 的数据长度不正确	数据长度必须 <= 65534 字节。

返回值 STATUS* (W#16#...)：	说明	注意
82xx, 84xx, 或 85xx	邮件服务器产生的错误消息对应于 SMTP 协议的错误编号（“8”除外）。 以下行列出了可能出现的几个错误代码。	关于 SMTP 错误代码和 SMTP 协议中其它错误代码的更多详细信息，请参见 Internet 或者邮件服务器的错误信息文档。还可通过邮件服务器查看背景数据块的 BUFFER1 参数中的最新错误消息。可在背景数据块的 DATEN 下找到 TMAIL_C 指令发送的上一数据。
8450	活动未执行：邮箱不可用/无法访问	请稍后重试。
8451	活动已中止：本地处理出错	请稍后重试。
8500	语法错误：未知错误。这还包括命令字符串过长所致的错误。电子邮件服务器不支持 LOGIN 授权程序时，也会出现此类错误。	请检查 TMAIL_C 的参数。尝试发送无需验证的电子邮件。为此，可以用空字符串代替参数 UserName 的内容。如果没有指定用户名，则不使用 LOGIN 验证程序。
8501	语法错误：参数的输入不正确	可能的原因：TO_S 或 CC 参数处的地址错误（另请参见：TO_S 和 CC 参数 (页 904)）。
8502	命令无法识别或者不能执行	检查您的输入项，尤其是参数 FROM。参数可能不完整，可能忘记输入“@”或“.”（另请参见：TO_S 和 CC 参数 (页 904)）。
8535	SMTP 验证不完整	输入的用户名或者密码可能不正确。
8550	无法访问邮件服务器。您没有访问权限。	输入的用户名或者密码可能不正确，或者邮件服务器不支持您的登录。错误的另一个原因可能是 TO_S 或 CC 参数处“@”后的域名不正确（另请参见：TO_S 和 CC 参数 (页 904)）。
8552	活动已中止：超过了所分配的存储容量	请稍后重试。
8554	传送失败	请稍后重试。
* 错误代码可在程序编辑器中显示为整数或十六进制值。		

背景数据块的 SFB_STATUS 参数

对于 S7-1200 CPU 固件版本 V4.4 或更高版本，以及 TMAIL_C 指令版本 V6.0 或更高版本，背景 DB 的 SFB_STATUS 参数可能返回以下值：

背景 DB 的 SFB_STATUS 参数的返回值 (W#16#...)	说明
8085	连接 ID（ID 参数）已经被已组态的连接使用。
8086	ID 参数超出了有效范围。
8087	已达到最大连接数；无法建立更多连接
8088 *	文件不存在或者当前不可用。
8089 *	无法打开文件，因为同时打开的文件数已超过系统的限制。在 S7-1200 上，每个文件系统最多允许打开 26 个文件。
808A *	DirectoryPath 包含除 DataLogs、Recipes 或 UserFiles 以外的目录，或者其中一个寻址的子目录违反了前述寻址规则。有关详细信息，请参见 DirectoryPath (页 905)。
808B *	FileName 包含非法字符序列或已留空。有关详细信息，请参见 FileName (页 905)。
808C *	寻址作为附件的文件路径时，AttachmentName 参数必须留空。
8092	TO_S 和 CC 参数为空，或者 From 子参数为空或不完整。
8093	MAIL_ADDR_PARAM 参数要求将连接升级为安全连接，但邮件服务器不支持 STARTTLS 命令。
8095	邮件服务器响应无效。邮件服务器可能不符合 RFC。
809A	集成接口不支持参数 MAIL_ADDR_PARAM 的 SDT 结构。
809B	MAIL_ADDR_PARAM 参数中 SDT 的 ID 接口无效
80A1	指定的连接或远程端口正在使用中。
80A3	ID 已由用户程序创建的连接使用。
80A4	远程连接端点的 IP 地址无效，或者与本地伙伴的 IP 地址重复。
80A7	通信错误：在 TMAIL_C 完成前执行了 TDISCON。
80B7	远程端口为 0，或伙伴端点的 IP 地址已设置为 0.0.0.0。

背景 DB 的 SFB_STATUS 参数的返回值 (W#16#...)	说明
80C3	CPU 中资源不足
80C4	临时通信错误： ● 此时无法建立连接。 ● 由于连接路径中防火墙的指定端口未打开，无法建立连接。 ● 接口当前正在接收新参数。 ● TDISCON 指令当前正在删除已组态的连接。
80C5	邮件服务器拒绝建立连接，已终止或主动结束该连接。
80C6	无法访问连接伙伴（网络错误）。
80C7	执行超时
80C8	正在尝试重新建立现有连接。
80C9	连接伙伴验证失败。邮件服务器不对应于 MailServerAddress 参数中定义的伙伴。
80CE	本地接口的 IP 地址为 0.0.0.0。
80D0	在尝试使用 DNS 时，MailServerAddress 参数包含一个空字符串。
80D1	MailServerAddress 参数不是全限定域名。结尾处的句点可能缺失。
80D2	未组态 DNS 服务器地址。
80D3	无法解析完全限定域名 (FQDN)。可能的原因： ● DNS 服务器无法访问（例如，DNS 服务器已关闭或远程端口不可访问）。 ● 与 DNS 服务器进行通信时出错。 ● DNS 服务器返回一个有效的 DNS 响应；但该响应中未包含 IPv4 地址。
80E0	由于消息出错，与邮件服务器之间的通信失败。可能的原因： ● 消息认证代码无效 ● 消息解码失败。 ● 消息解压出错 ● 内部容量溢出

背景 DB 的 SFB_STATUS 参数的返回值 (W#16#...)	说明
80E1	通信握手过程中发生错误。可能的原因： ● 用户中止该过程 ● 安全等级不足 ● 不支持重新协商 ● 不支持该 SSL/TLS 版本。 ● 说明错误
80E2	证书不支持/无效 可能的原因：所连接模块的时间日期未设置，或模块未同步。 示例：模块默认的日期设置为 1/1/2012 且日期不在调试过程中设置。证书的有效期从 2016 年 8 月 20 日开始，到 2024 年 8 月 20 日结束。在这种情况下，模块的日期超出证书的有效期；证书对于模块无效。
80E3	邮件服务器证书已丢弃。
80E4	未找到邮件服务器证书的有效证书颁发机构。
80E5	邮件服务器证书过期。
80E6	传输层安全协议出现完整性错误
80E7	不支持邮件服务器证书扩展
80E9	不支持不含服务器证书的 TLS 服务器

* 这些错误代码已添加到 TMAIL_C 指令中，用于辅助诊断不适当的文件路径寻址。

11.2.8.15 UDP

UDP 是由 RFC 768 描述的一种标准协议：用户数据报协议。UDP 提供了一种一个应用程序向另一个应用程序发送数据报可采用的机制；但是，数据的传输得不到保证。该协议有以下特点：

- 快速通信协议
- 适合用于小数据量到中等数据量（最多 1472 字节）
- UDP 是比 TCP 更加简单的传输控制协议，其薄层占用资源非常少
- 可以非常灵活地与许多第三方系统一起使用
- 有路由功能
- 使用端口号指引数据报

- 不确认消息：需要负责错误恢复和安全性的应用程序
- 由于使用 SEND/RECEIVE 编程接口的缘故，需要编程来进行数据管理

UDP 支持广播通信。要使用广播，必须组态 ADDR 组态的 IP 地址部分。例如：IP 地址为 192.168.2.10、子网掩码为 255.255.255.0 的 CPU 将使用广播地址 192.168.2.255。

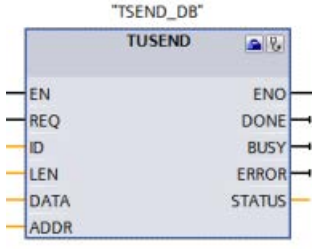
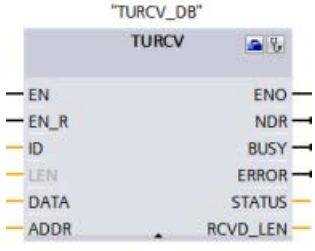
11.2.8.16 TUSEND 和 TURCV

以下指令控制 UDP 通信过程：

- TCON 在客户机与服务器 (CPU) PC 之间建立通信连接。
- TUSEND 和 TURCV 发送和接收数据。
- TDISCON 断开客户机与服务器之间的通信。

有关 TCON 和 TDISCON 通信指令的更多信息，请参见“TCP 和 ISO-on-TCP”部分中的 TCON、TDISCON、TSEND 和 TRCV (页 862)。

表格 11-45 TUSEND 和 TURCV 指令

LAD/FBD	SCL	说明
	"TUSEND_DB"(req:=_bool_in_, ID:=_word_in_, len:=_udint_in_, done=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, data:=_variant_inout_);	TUSEND 指令通过 UDP 将数据发送到参数 ADDR 指定的远程伙伴。 要启动用于发送数据的作业，请调用 REQ = 1 的 TUSEND 指令。
	"TURCV_DB"(en_r:=_bool_in_, ID:=_word_in_, len:=_udint_in_, ndr=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, rcvd_len=>_udint_out_, data:=_variant_inout_);	TURCV 指令通过 UDP 接收数据。 参数 ADDR 显示发送方地址。 TURCV 成功完成后，参数 ADDR 将包含远程伙伴（发送方）的地址。 TURCV 不支持特殊模式。 要启动用于接收数据的作业，请调用 EN_R = 1 的 TURCV 指令。

[1] STEP 7 会在插入指令时自动创建 DB。

TCON、TDISCON、TUSEND 和 TURCV 异步运行，即，作业处理需要多次执行指令来完成。

表格 11-46 TUSEND 和 TURCV 参数的数据类型

参数和类型		数据类型	说明
REQ (TUSEND)	IN	Bool	在上升沿启动发送作业。 传送通过 DATA 和 LEN 指定的区域中的数据。
EN_R (TURCV)	IN	Bool	• 0: CPU 无法接收。 • 1: 允许 CPU 进行接收。 TURCV 指令准备接收，并处理接收作业。
ID	IN	Word	引用用户程序与操作系统通信层之间的相关连接。ID 必须与本地连接描述中的相关参数 ID 相同。 值范围： W#16#0001 到 W#16#0FFF。
LEN	IN	UDInt	要发送 (TUSEND) 或接收 (TURCV) 的字节数。 • 默认值为 0。DATA 参数确定要发送或接收的数据长度。 • 否则为值范围： 1 到 1472
DONE (TUSEND)	IN	Bool	状态参数 DONE (TUSEND)： • 0: 作业尚未开始或仍在运行。 • 1: 作业无错完成。
NDR (TURCV)	OUT	Bool	状态参数 NDR (TURCV)： • 0: 作业尚未开始或仍在运行。 • 1: 作业已成功完成。
BUSY	OUT	Bool	• 1: 作业尚未完成。 无法触发新作业。 • 0: 作业已完成。
ERROR	OUT	Bool	状态参数，可具有以下值： • 0: 无错误 • 1: 处理时出错。 STATUS 提供错误类型的详细信息。
STATUS	OUT	Word	包括错误信息的状态信息。（请参见下表中的错误和状态条件代码。）
RCVD_LEN	OUT	UDInt	接收的字节数 (TURCV)

参数和类型		数据类型	说明
DATA	IN_OUT	Variant	发送区 (TUSEND) 或接收区 (TURCV) 的地址： ● 过程映像输入表 ● 过程映像输出表 ● 存储器位 ● 数据块
ADDR	IN_OUT	Variant	指向接收方（对于 TUSEND）或发送方（对于 TURCV）的地址的指针（例如，P#DB100.DBX0.0 byte 8）。 该指针可指向任何存储区。 需要 8 字节的结构，具体如下： ● 前 4 个字节包含远程 IP 地址。 ● 接下来的 2 个字节指定远程端口号。 ● 最后 2 个字节保留。

作业状态由输出参数 BUSY 和 STATUS 指示。STATUS 与以异步方式工作的指令的 RET_VAL 输出参数一致。

下表给出了 BUSY、DONE (TUSEND)、NDR (TURCV) 和 ERROR 之间的关系。 通过该表格，用户可以确定指令（TUSEND 或 TURCV）的当前状态或者发送（传送）/接收过程完成的时间。

表格 11-47 BUSY、DONE (TUSEND)/NDR (TURCV) 和 ERROR 参数的状态

BUSY	DONE / NDR	ERROR	说明
TRUE	不相关	不相关	正在处理作业。
FALSE	TRUE	FALSE	作业已成功完成。
FALSE	FALSE	TRUE	作业因错结束。 出错原因可在 STATUS 参数中找到。
FALSE	FALSE	FALSE	未给该指令分配（新）作业。

[1] 由于指令以异步方式工作： 对于 TUSEND，在 DONE 参数值或 ERROR 参数值为 TRUE 前，必须保持发送方区域中的数据一致。 对于 TURCV，仅当 NDR 参数值为 TRUE 时，接收方区域中的数据才一致。

表格 11-48 TUSEND 和 TURCV 指令的 ERROR 和 STATUS 的条件代码

ERROR	STATUS	说明
0	0000	● 发送作业无错完成 (TUSEND)。 ● 接受了新数据。 在 RCVD_LEN 中显示已接收数据的当前长度 (TURCV)。
0	7000	● 无激活的作业处理 (TUSEND) ● 块未准备好接收 (TURCV)

ERROR	STATUS	说明
0	7001	• 启动作业处理，正在发送数据 (TUSEND)： 在执行此处理期间，操作系统访问 DATA 发送区中的数据。 • 块准备接收，接收作业已激活 (TURCV)。
0	7002	• 后续指令执行（与 REQ 无关），正在处理作业 (TUSEND)： 在执行此处理期间，操作系统访问 DATA 发送区中的数据。 • 后续指令执行，正在处理作业： 在执行此处理期间，TURCV 指令将数据写入接收区。 因此，错误可能导致接收区中的数据不一致。
1	8085	LEN 参数值大于最大允许值，其值为 0 (TUSEND)，或者自第一次执行指令 (TURCV) 以来更改了 LEN 或 DATA 参数的值。
1	8086	ID 参数不在允许的地址范围内。
1	8088	• LEN 参数大于 DATA 中指定的存储区 (TUSEND) 或接收区 (TURCV)。 • 接收区过小 (TURCV)。
1	8089	ADDR 参数未指向数据块。
1	80A1	通信错误： • 尚未建立用户程序和操作系统通信层之间的指定连接。 • 当前正在终止用户程序和操作系统通信层之间的指定连接。 无法通过该连接执行传送 (TUSEND) 或接收作业 (TURCV)。 • 正在重新初始化接口。
1	80A4	远程连接端点的 IP 地址无效；可能与本地 IP 地址匹配 (TUSEND)。
1	80B3	• 设置的协议（连接说明中的 connection_type 参数）不是 UDP。 请使用 TSEND 或 TRCV 指令。 • ADDR 参数： 端口号的设置无效 (TUSEND)
1	80C3	• 具有该 ID 的块正在一个具有不同优先级的组中处理。 • 内部缺乏资源
1	80C4	临时通信错误： • 此时无法建立用户程序和操作系统通信层之间的连接 (TUSEND)。 • 接口正在接收新参数 (TUSEND)。 • 当前正在重新启动连接 (TURCV)。

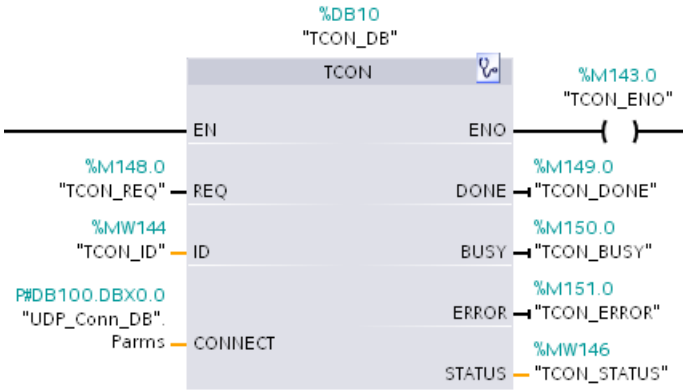
以太网连接协议

每个 CPU 都集成了一个支持标准 PROFINET 通信的 PROFINET 端口。 TUSEND 和 TURCV 指令支持 UDP 以太网协议。

更多相关信息，请参见“设备配置”一章中的“组态本地/伙伴连接路径” (页 812)。

操作

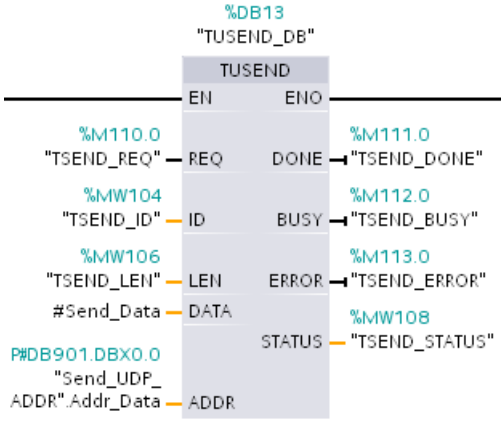
两个伙伴在 UDP 通信中均为被动方。 下图给出“TCON_Param”数据类型的典型参数起始值。 端口号 (LOCAL_TSAP_ID) 以 2 字节形式表示。 允许使用除 161、34962、34963 和 34964 外的所有端口。



UDP_Conn_DB

	名称	数据类型	偏移量	启动值	注释
1	Static				
2	Parms	TCON_Param	0.0		
3	BLOCK_LENGTH	UInt	0.0	64	byte length of SDT
4	ID	CONN_OUC	2.0	1	reference to the connection
5	CONNECTION_TYPE	USInt	4.0	19	17: TCP/IP, 18: ISO on TCP
6	ACTIVE_EST	Bool	5.0	false	active/passive connection establishment
7	LOCAL_DEVICE_ID	USInt	6.0	1	1: local IE interface
8	LOCAL_TSAP_ID_LEN	USInt	7.0	2	byte length of local TSAP id/port number
9	REM_SUBNET_ID_LEN	USInt	8.0	0	byte length of remote subnet id
10	REM_STADDR_LEN	USInt	9.0	0	byte length of remote IP address
11	REM_TSAP_ID_LEN	USInt	10.0	0	byte length of remote port/TSAP id
12	NEXT_STADDR_LEN	USInt	11.0	0	byte length of next station address
13	LOCAL_TSAP_ID	Array[1..16] of Byte	12.0		TSAP id/local port number
14	LOCAL_TSAP_ID[1]	Byte		B#16#07	
15	LOCAL_TSAP_ID[2]	Byte		B#16#D0	

TUSEND 指令通过 UDP 将数据发送到“TADDR_Param”数据类型中指定的远程伙伴。 TURCV 指令通过 UDP 接收数据。 如下图所示，成功执行 TURCV 指令之后，“TADDR_Param”数据类型会显示远程伙伴（发送方）的地址。



Send_UDP_ADDR

	名称	数据类型	偏移量	启动值	注释
1	Static				
2	Addr_Data	TADDR_Param	0.0		
3	REM_IP_ADDR	Array[1..4] of USint	0.0		remote station address
4	REM_IP_ADDR[1]	USint		0	
5	REM_IP_ADDR[2]	USint		0	
6	REM_IP_ADDR[3]	USint		0	
7	REM_IP_ADDR[4]	USint		0	
8	REM_PORT_NR	UInt	4.0	0	remote port number
9	RESERVED	Word	6.0	0	unused; has to be 0

11.2.8.17 T_CONFIG

指令“T_CONFIG”可以更改以太网地址、PROFINET 设备名称或 NTP 服务器的 IP 地址，从而在用户程序中进行时间同步。可以永久或临时调整以下特征：

- IP 地址
- 子网掩码
- 路由器地址

- 站名称
- 最多四个 NTP 服务器的 IP 地址

说明

位于“以太网地址”(Ethernet address) 页面的 CPU“属性”(Properties) 的“在设备上直接设置 IP 地址” (页 928)(IP address is set directly at the device) 单选按钮允许在下载程序之后在线或使用“T_CONFIG”指令更改 IP 地址。

位于“以太网地址”(Ethernet address) 页面的 CPU“属性”(Properties) 的“在设备上直接设置 PROFINET 设备名称” (页 930)(PROFINET device name is set directly at the device) 单选按钮允许在下载程序之后在线或使用“T_CONFIG”指令更改 PROFINET 设备名称。

位于“时间同步”(Time synchronization) 页面的 CPU“属性”(Properties) 的“通过 NTP 服务器启用时间同步” (页 931)(Enable time synchronization via NTP serve) 框允许更改最多 4 个 NTP 服务器的 IP 地址。

说明

不能一次执行多个 T_CONFIG 指令。

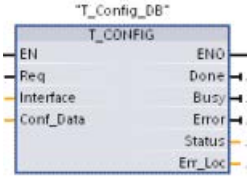
说明

可以永久性或临时性更改 IP 地址或 CPU 站名称。只能临时性更改 NTP 服务器的 IP 地址。

- 永久性更改表示该更改具有保持性，意味着在电源故障时更改仍然存在。
- 临时更改表示更改具有易失性并且会在停电后返回原始值。

表格 11-49 T_CONFIG 指令

LAD/FBD	SCL	说明

	`"T_CONFIG_DB"(`
	`Req:=_bool_in_,`
	`Interface:=_uint_in_,`
	`Conf_Data:=_variant_in_,`
	`Done=>_bool_out_,`
	`Busy=>_bool_out_,`
	`Error=>_bool_out_,`
	`Status=>_dword_out_,`
	`Err_Loc=>_dword_out_);`
		在用户程序中使用 T_CONFIG 指令可更改 IP 组态参数。
		T_CONFIG 异步运行。执行作业时需要多次调用指令。

表格 11-50 参数的 T_CONFIG 数据类型

参数和类型		数据类型	说明
REQ	Input	Bool	在上升沿时启动该指令。
INTERFACE	Input	HW_Interface	网络接口的 ID

参数和类型		数据类型	说明
CONF_DATA	Input	Variant	参考组态数据结构；CONF_DATA 由最多包含 4 个系统数据类型的结构定义(SDT)。
DONE	Output	Bool	• 0：作业尚未启动或仍在运行。 • 1：作业已无错执行。
BUSY	Output	Bool	• 0：作业已完成。 • 1：作业尚未完成。无法触发新作业。
ERROR	Output	Bool	状态参数，可具有以下值： • 0：无错误 • 1：处理期间出错。STATUS 提供错误类型的详细信息。
STATUS	Output	DWord	包括错误信息的状态信息。（请参见下表中的错误和状态条件代码。）
ERR_LOC	Output	DWord	故障位置（CONF_DATA 结构中的 ID 字段和子字段位置）

IP 组态信息与上面所述参数 CONF_DATA 中的 Variant 指针一起存储在 CONF_DATA 数据块中。 T_CONFIG 指令的成功执行以 IP 组态数据传送到网络接口宣告结束。

指令“T_CONFIG”的状态和错误消息通过参数“STATUS”和“ERR_LOC”输出。

- 错误原因通过参数 STATUS 输出。
- 错误位置通过参数 ERR_LOC 输出。提供有下列选项：
  - 16#0000_0000：无错误或指令调用错误（例如，指令参数赋值错误或 PROFINET 接口通信错误）。
  - 16#0001_0000：系统数据类型 IF_CONF_HEADER 的参数中出现的组态数据错误。
  - 16#0001_000x：系统数据类型 IF_CONF_V4 或 IF_CONF_NOS 或 IF_CONF_NTP 的组态数据存在错误（x 为 T_CONFIG 结构中错误子块的位置）。例如，如果 T_CONFIG 结构中包含一个指定 IP 地址的子块和一个指定站名称的子块，且错误位于指定站名称的子块内，则 ERR_LOC 的值为 0001_0002。）

下表显示了参数 STATUS 和 ERR_LOC 的可能取值：

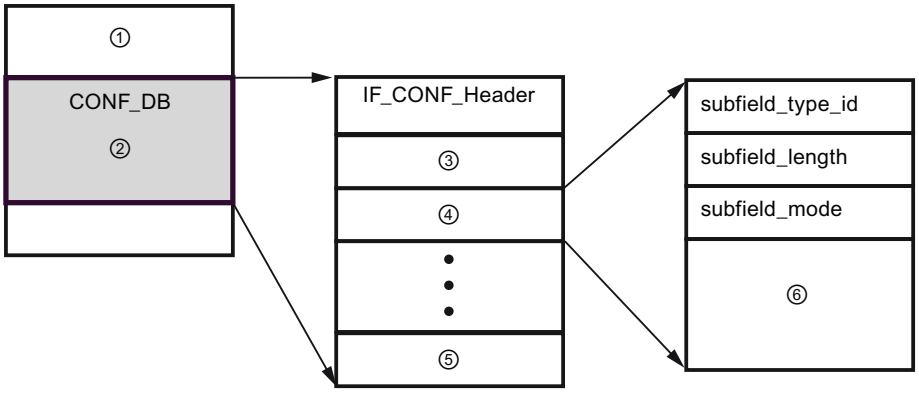
STATUS*	ERR_LOC*	说明
0000_0000	0000_0000	订单处理成功完成。
0070_0000	0000_0000	未激活任何作业处理。

STATUS*	ERR_LOC*	说明
0070_0100	0000_0000	启动订单处理。
0070_0200	0000_0000	中间调用（与 REQ 无关）。
C08x_yy00	0000_0000	常见错误信息。
C080_8000	0000_0000	指令调用错误： 参数 Interface 中的硬件 ID 无效。
C080_8100	0000_0000	指令调用错误： 参数 Interface 中的硬件 ID 无法寻址 PROFINET 接口。
C080_8700	0000_0000	指令调用错误： 参数 CONF_DATA 中的数据块的长度不正确。
C080_8800	0001_0000	系统数据类型 IF_CONF_HEADER 出错： 参数 FieldType 的值无效。FieldType 取值为“0”。
C080_8900	0001_0000	系统数据类型 IF_CONF_HEADER 出错： 参数 FieldId 含有无效值或多次使用。FieldId 取值为“0”。
C080_8A00	0001_0000	系统数据类型 IF_CONF_HEADER 出错： 参数 SubfieldCount 中的数量不正确。输入所用系统数据类型 IF_CONF_V4、IF_CONF_NOS 或 IF_CONF_NTP 的正确数量。
C080_8B00	0001_000x	系统数据类型 IF_CONF_V4、 IF_CONF_NOS 或 IF_CONF_NTP 中存在错误： 参数 Id 的值无效。IF_CONF_V4 为“30”；IF_CONF_NOS 为“40”；IF_CONF_NTP 为“17”。
C080_8C00	0001_000x	系统数据类型 IF_CONF_V4、 IF_CONF_NOS 或 IF_CONF_NTP 中存在错误： 所使用的系统数据类型不正确，订单错误或多次使用一个系统数据类型。
C080_8D00	0001_000x	系统数据类型 IF_CONF_V4、 IF_CONF_NOS 或 IF_CONF_NTP 中存在错误： 参数 Length 的值不正确或无效。
C080_8E00	0001_000x	系统数据类型 IF_CONF_V4、 IF_CONF_NOS 或 IF_CONF_NTP 中存在错误： 参数 Mode 的值不正确或无效。 ● 对于 IF_CONF_V4 和 IF_CONF_NOS， 的值只能为“1”（永久）或“2”（临时）。 ● 对于 IF_CONF_NTP，值只能为“2”（临时）。

STATUS*	ERR_LOC*	说明
C080_9000	0001_000x	系统数据类型 IF_CONF_V4、 IF_CONF_NOS 或 IF_CONF_NTP 中存在错误： 不能应用组态数据。可能的原因： • IF_CONF_V4：在硬件配置中，“在设备上设置 IP 地址”(Set IP address on the device) 设置未选中。 • IF_CONF_NOS：在硬件配置中，“在设备上设置 PROFINET 设备名称”(Set PROFINET device name on the device) 设置未选中。 • IF_CONF_NTP：在硬件配置中，“通过 NTP 服务器启用时间同步”(Enable time synchronization via NTP server) 设置未选择，而且 NTP 服务器的 IP 地址未设置。
C080_9400	0001_000x	系统数据类型 IF_CONF_V4、 IF_CONF_NOS 或 IF_CONF_NTP 中存在错误： 参数值未定义或无效。
C080_9500	0001_000x	系统数据类型 IF_CONF_V4、 IF_CONF_NOS 或 IF_CONF_NTP 中存在错误： 两个参数的取值不一致。
C080_C200	0000_0000	指令调用错误： 无法传送组态数据。可能的原因：无法访问 PROFINET 接口。
C080_C300	0000_0000	指令调用错误： 资源不足（例如，使用不同参数多次调用“T_CONFIG”）。
C080_C400	0000_0000	指令调用错误： 临时通信错误。转换为夏令时时间的时间规范。
C080_D200	0000_0000	指令调用错误： 无法调用。所选 PROFINET 接口不支持该指令。

CONF_DATA 数据块

下图显示了待传送的组态数据在组态 DB 中的存储情况。



①	组态 DB	④	子字段 2
②	组态数据	⑤	子字段 *n*
③	子字段 1	⑥	子字段特定的参数

CONF_DB 的组态数据由一个包含字段头 (IF_CONF_Header) 的字段和多个子字段构成。IF_CONF_Header 提供以下元素：

- field_type_id（数据类型 UInt）：零
- field_id（数据类型 UInt）：零
- subfield_cnt（数据类型 UInt）：子字段数

各子字段又由字段头（subfield_type_id、subfield_length、subfield_mode）和子字段特定的参数组成。各子字段必须由偶数个字节组成。subfield_mode 可以支持 1 或 2 的值。请参见下表：

说明

目前仅允许一个字段 (IF_CONF_Header)。其参数 field_type_id 和 field_id 的值必须为零。其它具有不同 field_type_id 和 field_id 值的字段用于将来扩展。

表格 11-51　支持的子字段

subfield_type_id	数据类型	说明
30	IF_CONF_V4	IP 参数：IP 地址、子网掩码、路由器地址
40	IF_CONF_NOS	PROFINET IO 设备名称 (Name of station)
17	IF_CONF_NTP	网络时间协议 (NTP)

表格 11-52 IF_CONF_V4 数据类型的元素

名称		数据类型	起始值	说明
Id		UInt	30	subfield_type_id
Length		UInt	18	subfield_length
Mode		UInt	0	subfield_mode（1：永久或 2：暂时）
InterfaceAddres s		IP_V4	-	接口地址
ADDR		Array [1..4] of Byte		
	ADDR[1]	Byte	0	IP 地址高位字节：200
	ADDR[2]	Byte	0	IP 地址高位字节：12
	ADDR[3]	Byte	0	IP 地址低位字节：1
	ADDR[4]	Byte	0	IP 地址低位字节：144
SubnetMask		IP_V4	-	子网掩码
ADDR		Array [1..4] of Byte		
	ADDR[1]	Byte	0	子网掩码高位字节：255
	ADDR[2]	Byte	0	子网掩码高位字节：255
	ADDR[3]	Byte	0	子网掩码低位字节：255
	ADDR[4]	Byte	0	子网掩码低位字节：0
DefaultRouter		IP_V4	-	默认路由器
ADDR		Array [1..4] of Byte		
	ADDR[1]	Byte	0	路由器高位字节：200
	ADDR[2]	Byte	0	路由器高位字节：12
	ADDR[3]	Byte	0	路由器低位字节：1
	ADDR[4]	Byte	0	路由器低位字节：1

表格 11-53 IF_CONF_NOS 数据类型的元素

名称	数据类型	起始值	说明
Id	UInt	40	subfield_type_id
Length	UInt	246	subfield_length

名称	数据类型	起始值	说明
Mode	UInt	0	subfield_mode（1：永久或 2：暂时）
NOS (Name of station)	Array[1..240] of Byte	0	站名称：必须从第一个字节开始填充 ARRAY。如果 ARRAY 比要指定的站名称长，则必须在实际站名称后输入零字节（符合 IEC 61158-6-10）。否则，将拒绝 NOS，并且 “T_CONFIG (页 919)”指令将在 STATUS 中输入错误代码 DW#16#C0809400。如果用零填充第一个字节，则将删除站名称。

站名称有以下限制：

- 站名称中的名称部分，即两个点之间的字符串，不得超过 63 个字符。
- 不允许使用变音、括号、下划线、斜线、空格等特殊字符。允许使用的特殊字符仅为破折号。
- 站名称不得以“-”字符开始或结尾。
- 站名称不得以数字开头。
- 不允许站名称形式 n.n.n.n (n = 0 ... 999)。
- 站名称不得以字符串“port-xyz”或“port-xyz-abcde”（a、b、c、d、e、x、y、z = 0 ... 9）开头。

说明

可以创建一个 ARRAY 类型的“NOS”，其长度在 2 到 240 个字节之间。此时，必须相应地对“Length”（子域的长度）进行调整。

表格 11-54 IF_CONF_NTP 数据类型的元素

名称	数据类型	起始值	说明
Id	UInt	17	subfield_type_id
Length	UInt	22	subfield_length
Mode	UInt	0	subfield_mode（2：暂时）
NTP_IP	Array[1...4] of IP_V4	-	NTP 服务器的 IP 地址
NTP_IP[1]	IP_V4		NTP 服务器 1 的 IP 地址
ADDR	Array[1...4] of Byte	0	
ADDR[1]	Byte	0	IP 地址的高位
ADDR[2]	Byte	0	IP 地址的高位

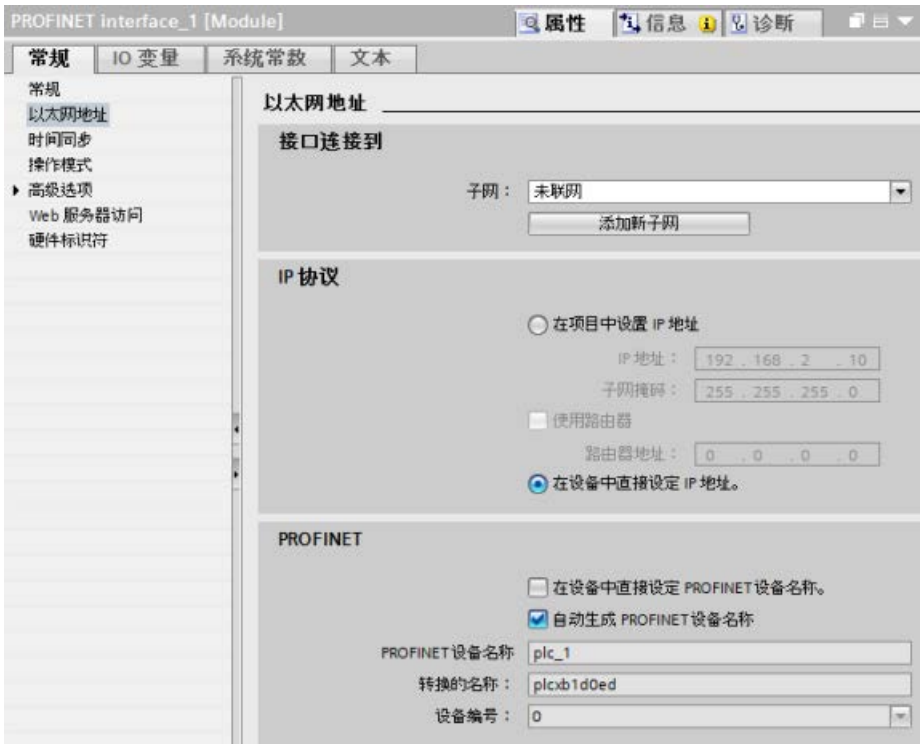
名称	数据类型	起始值	说明
ADDR[3]	Byte	0	IP 地址的低位
ADDR[4]	Byte	0	IP 地址的低位
NTP_IP[2]	IP_V4		NTP 服务器 2 的 IP 地址
ADDR	Array[1...4] of Byte	0	
ADDR[1]	Byte	0	IP 地址的高位
ADDR[2]	Byte	0	IP 地址的高位
ADDR[3]	Byte	0	IP 地址的低位
ADDR[4]	Byte	0	IP 地址的低位
NTP_IP[3]	IP_V4		NTP 服务器 3 的 IP 地址
ADDR	Array[1...4] of Byte	0	
ADDR[1]	Byte	0	IP 地址的高位
ADDR[2]	Byte	0	IP 地址的高位
ADDR[3]	Byte	0	IP 地址的低位
ADDR[4]	Byte	0	IP 地址的低位
NTP_IP[4]	IP_V4		NTP 服务器 4 的 IP 地址
ADDR	Array[1...4] of Byte	0	
ADDR[1]	Byte	0	IP 地址的高位
ADDR[2]	Byte	0	IP 地址的高位
ADDR[3]	Byte	0	IP 地址的低位
ADDR[4]	Byte	0	IP 地址的低位

示例： 使用 T_CONFIG 指令更改 IP 参数

在以下示例中，更改了“addr”子字段下的 "InterfaceAddress"（IP 地址）、"SubnetMask" 和 "DefaultRouter"（IP 路由器）。要在下载程序之后使用“T_CONFIG”指令更改 IP 参数，必须在 “以太网地址”(Ethernet address) 页面的 CPU“属性”(Properties) 中选中 “在设备上直接设置 IP 地址”(IP address is set directly at the device) 单选按钮。

CONF_DATA_1

	名称	数据类型	启动值
1	Static		
2	Conf_data	Struct	
3	header	IF_CONF_Header	
4	FieldType	UInt	0
5	FieldId	UInt	0
6	SubfieldCount	UInt	1
7	addr	IF_CONF_v4	
8	Id	UInt	30
9	Length	UInt	18
10	Mode	UInt	1
11	InterfaceAddress	IP_V4	
12	ADDR	array [1..4] of Byte	
13	ADDR[1]	Byte	192
14	ADDR[2]	Byte	168
15	ADDR[3]	Byte	2
16	ADDR[4]	Byte	30
17	SubnetMask	IP_V4	
18	ADDR	array [1..4] of Byte	
19	ADDR[1]	Byte	255
20	ADDR[2]	Byte	255
21	ADDR[3]	Byte	255
22	ADDR[4]	Byte	0
23	DefaultRouter	IP_V4	
24	ADDR	array [1..4] of Byte	
25	ADDR[1]	Byte	192
26	ADDR[2]	Byte	168
27	ADDR[3]	Byte	2
28	ADDR[4]	Byte	1

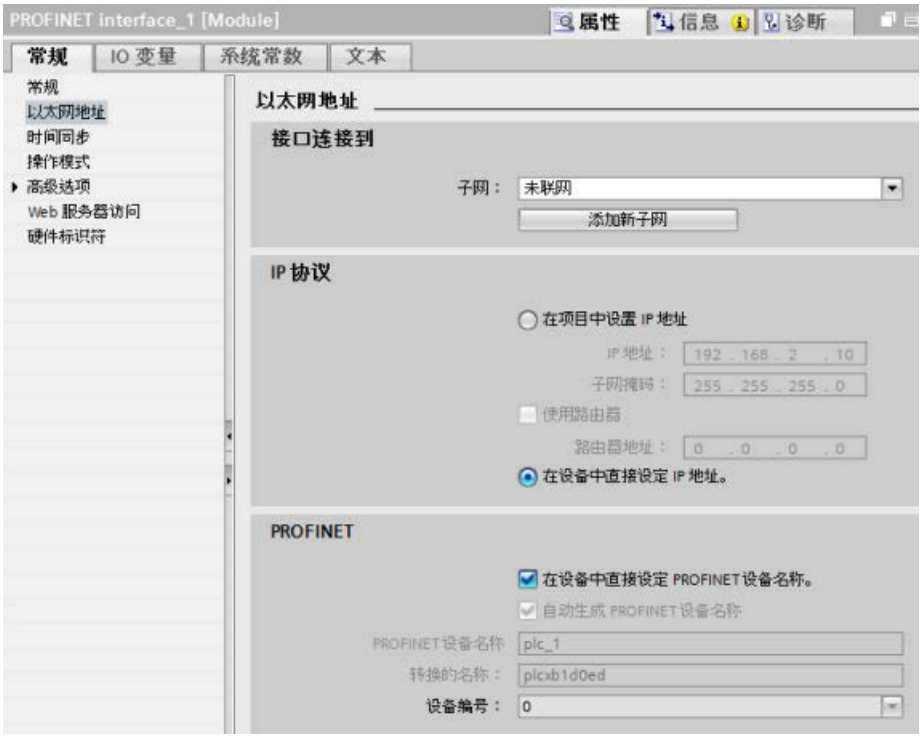

PROFINET interface_1 [Module]
属性
信息
诊断
常规
IO 变量
系统常数
文本
常规
以太网地址
时间同步
操作模式
高级选项
Web 服务器访问
硬件标识符
以太网地址
接口连接到
子网：
未联网
添加新子网
IP 协议
在项目中设置 IP 地址
IP 地址：
192 . 168 . 2 . 10
子网掩码：
255 . 255 . 255 . 0
使用路由器
路由器地址：
0 . 0 . 0 . 0
在设备中直接设定 IP 地址。
PROFINET
在设备中直接设定 PROFINET 设备名称。
自动生成 PROFINET 设备名称
PROFINET 设备名称
plc_1
转换的名称：
plcxb1d0ed
设备编号：
0

示例： 使用 T_CONFIG 指令更改 IP 参数和 PROFINET IO 设备名称

在以下示例中，更改了“addr”和“nos”(Name of station) 这两个子字段。要在下载程序之后使用“T_CONFIG”指令更改 PROFINET 设备名称，必须在“以太网地址”(Ethernet address) 页面的 CPU“属性”(Properties) 中选中“在设备上直接设置 PROFINET 设备名称”(PROFINET device name is set directly at the device) 复选框。

CONF_DATA_2

	名称	数据类型	启动值
1	Static		
2	Conf_data	Struct	
3	header	IF_CONF_Header	
4	FieldType	UInt	0
5	FieldId	UInt	0
6	SubfieldCount	UInt	2
7	addr	IF_CONF_v4	
8	Id	UInt	30
9	Length	UInt	18
10	Mode	UInt	1
11	InterfaceAddress	IF_V4	
12	ADDR	array [1..4] of Byte	
13	SubnetMask	IF_V4	
14	ADDR	array [1..4] of Byte	
15	DefaultRouter	IF_V4	
16	ADDR	array [1..4] of Byte	
17	nos	IF_CONF_NOS	
18	Id	UInt	40
19	Length	UInt	246
20	Mode	UInt	1
21	NOS	array [1..240] of Byte	



示例：使用 T_CONFIG 指令更改 NTP 服务器中的 IP 地址

在下列示例中，在“ntp”（网络时间协议 (NTP) 服务器）子字段中，T_CONFIG 指令最多修改四个 NTP 服务器的 IP 地址。

在 CPU 属性、PROFINET 接口 [X1]、时间同步页面中，通过选中“通过 NTP 服务器激活时间同步”复选框来组态 NTP 同步，如下图所示。然后可以在程序下载之后使用“T_CONFIG”指令更改 NTP 服务器中的 IP 地址。

CONF_DATA_3

	名称	数据类型	启动值
1	Static		
2	Conf_Data	Struct	
3	header	IF_CONF_Header	
4	FieldType	UInt	0
5	FieldId	UInt	0
6	SubfieldCount	UInt	1
7	ntp	IF_CONF_NTP	
8	Id	UInt	17
9	Length	UInt	22
10	Mode	UInt	2
11	NTP_IP	Array[1..4] of IP_V4	
12	NTP_IP[1]	IP_V4	
13	ADDR	Array[1..4] of Byte	
14	ADDR[1]	Byte	192
15	ADDR[2]	Byte	168
16	ADDR[3]	Byte	2
17	ADDR[4]	Byte	5
18	NTP_IP[2]	IP_V4	
19	ADDR	Array[1..4] of Byte	
20	ADDR[1]	Byte	192
21	ADDR[2]	Byte	168
22	ADDR[3]	Byte	2
23	ADDR[4]	Byte	6
24	NTP_IP[3]	IP_V4	
25	ADDR	Array[1..4] of Byte	
26	ADDR[1]	Byte	192
27	ADDR[2]	Byte	168
28	ADDR[3]	Byte	2
29	ADDR[4]	Byte	7
30	NTP_IP[4]	IP_V4	
31	ADDR	Array[1..4] of Byte	
32	ADDR[1]	Byte	192
33	ADDR[2]	Byte	168
34	ADDR[3]	Byte	2
35	ADDR[4]	Byte	8

11.2.8.18 指令的公共参数

REQ 输入参数

许多开放式用户通信指令使用 REQ 输入在由低电平向高电平切换时启动操作。 REQ 输入在指令执行一次的时间内必须为高电平 (TRUE)，不过 REQ 输入可以在所需时间内一直保持为 TRUE。 在 REQ 输入为 FALSE 时执行指令以便能复位 REQ 输入的历史状态之前，该指令不会启动其它操作。 只有这样，指令才能检测低电平到高电平的跳变以启动下一个操作。

在程序中放置这些指令之一后，STEP 7 会提示用户指定背景数据块。 对每个指令调用使用一个唯一的背景数据块。 这样可确保每个指令都能正确地处理诸如 REQ 等输入。

ID 输入参数

这是对 STEP 7 中“设备和网络”(Devices and networks) 的“网络视图”(Network view) 中的“本地 ID（十六进制）”(Local ID (hex)) 的引用，并且是要用于该通信块的网络的 ID。 ID 必须与本地连接描述中的相关参数 ID 相同。

DONE、NDR、ERROR 和 STATUS 输出参数

这些指令提供说明完成状态的输出：

表格 11-55 开放式用户通信指令输出参数

参数	数据类型	默认值	说明
DONE	Bool	FALSE	设置为 TRUE 并持续执行一次所需的时间，以表明上一请求已经完成且没有出现错误；否则为 FALSE。
NDR	Bool	FALSE	设置为 TRUE 并持续执行一次所需的时间，以表明请求的动作已经完成且没有出现错误并已接收新的数据；否则为 FALSE。
BUSY	Bool	FALSE	激活时设置为 TRUE 以表明： ● 作业尚未完成。 ● 无法触发新作业。 作业完成时设置为 FALSE。

参数	数据类型	默认值	说明
ERROR	Bool	FALSE	设置为 TRUE 并持续执行一次所需的时间，以表明上一请求已经完成但出现了错误，相应的错误代码在 STATUS 中；否则为 FALSE。
STATUS	Word	0	结果状态： ● 如果设置了 DONE 或 NDR 位，则 STATUS 被设置为 0 或信息代码。 ● 如果设置了 ERROR 位，则 STATUS 被设置为一个错误代码。 ● 如果没有设置以上任何一位，则指令会返回说明功能当前状态的状态结果。 STATUS 在该功能执行期间一直保持其值。

说明

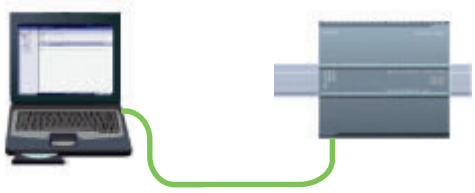
请注意，DONE、NDR 和 ERROR 仅置位一个执行周期的时间。

被动 ISO 和 TCP 通信的 TSAP 和端口号限制

如果使用“TCON”指令设置并建立被动通信连接，则下列端口地址将受到限制，不应该使用：

- ISO TSAP（被动）：
  - 01.00, 01.01, 02.00, 02.01, 03.00, 03.01
  - 10.00, 10.01, 11.00, 11.01, ... BF.00、BF.01
- TCP 端口（被动）： 5001, 102, 123, 20, 21, 25, 34962, 34963, 34964, 80
- UDP 端口（被动）： 161, 34962, 34963, 34964

11.2.9 与编程设备通信



CPU 可以与网络上的 STEP 7 编程设备进行通信。

在 CPU 和编程设备之间建立通信时请考虑以下几点：

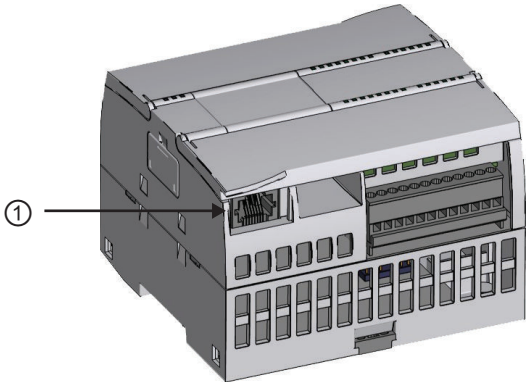
- 组态/设置： 需要进行硬件配置。
- 一对一通信不需要以太网交换机；网络中有两个以上的设备时需要以太网交换机。

11.2.9.1 建立硬件通信连接

PROFINET 接口可在编程设备和 CPU 之间建立物理连接。 由于 CPU 内置了自动跨接功能，所以对该接口既可以使用标准以太网电缆，又可以使用跨接以太网电缆。 将编程设备直接连接到 CPU 时不需要以太网交换机。

要在编程设备和 CPU 之间创建硬件连接，请按以下步骤操作：

1. 安装 CPU (页 60)。
2. 将以太网电缆插入下图所示的 PROFINET 端口中。
3. 将以太网电缆连接到编程设备上。



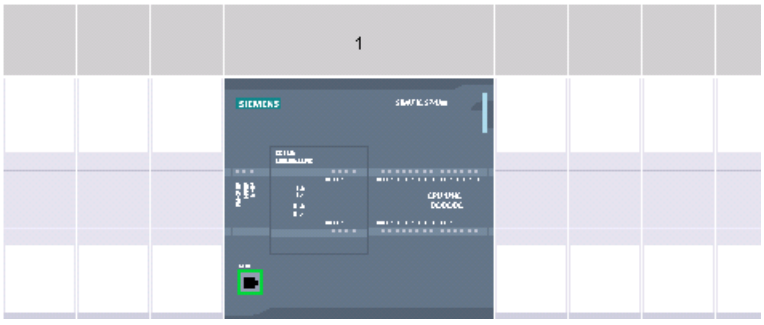
① PROFINET 端口

可选配张力消除装置以加固 PROFINET 连接。 关于订货信息，请参见“备件与其它硬件” (页 1657)。

11.2.9.2 配置设备

如果已使用 CPU 创建项目，则在 STEP 7 中打开项目。

如果没有，请创建项目并在机架中插入 CPU (页 148)。 在下面的项目中，“设备视图” (Device View) 中显示了 CPU。



11.2.9.3 分配 Internet 协议 (IP) 地址

分配 IP 地址

在 PROFINET 网络中，每个设备还必须具有一个 Internet 协议 (IP) 地址。该地址使设备可以在更加复杂的路由网络中传送数据：

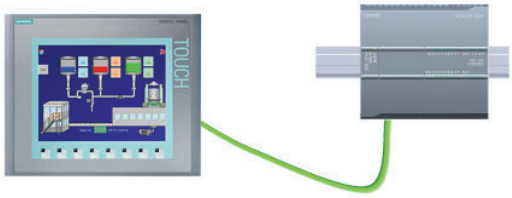
- 如果编程或其它网络设备使用连接到工厂 LAN 的板载适配器卡或连接到隔离网络的以太网到 USB 适配器卡，则必须为它们分配 IP 地址。更多相关信息，请参见“为编程设备和网络设备分配 IP 地址” (页 815)。
- 还可以在线为 CPU 或网络设备分配 IP 地址。这在进行初始设备配置时尤其有用。更多相关信息，请参见“在线为 CPU 分配 IP 地址” (页 815)。
- 组态项目中的 CPU 或网络设备后，可以组态 PROFINET 接口的参数及其 IP 地址。更多相关信息，请参见“为项目中的 CPU 组态 IP 地址” (页 817)。

11.2.9.4 测试 PROFINET 网络

完成组态后，必须将项目下载到 CPU 中。 下载项目时会组态所有 IP 地址。

CPU“下载到设备”(Download to device) 功能及其“扩展的下载到设备”(Extended download to device) 对话框可以显示所有可访问的网络设备，以及是否为所有设备都分配了唯一的 IP 地址。 更多相关信息，请参见“测试 PROFINET 网络” (页 822)。

11.2.10 HMI 到 PLC 通信



CPU 支持通过 PROFINET 端口与 HMI (页 35) 通信。 设置 CPU 和 HMI 之间的通信时必须考虑以下要求：

组态/设置：

- 必须组态 CPU 的 PROFINET 端口与 HMI 连接。
- 必须已设置和组态 HMI。
- HMI 组态信息是 CPU 项目的一部分，可以在项目内部进行组态和下载。
- 一对一通信不需要以太网交换机；网络中有两个以上的设备时需要以太网交换机。

说明

机架安装的 CSM1277 4 端口以太网交换机可用于连接 CPU 和 HMI 设备。 CPU 上的 PROFINET 端口不包含以太网交换设备。

支持的功能：

- HMI 可以对 CPU 读/写数据。
- 可基于从 CPU 重新获取的信息触发消息。
- 系统诊断

表格 11-56 组态 HMI 与 CPU 之间的通信时所需的步骤

步骤	任务
1	建立硬件通信连接 通过 PROFINET 接口建立 HMI 和 CPU 之间的物理连接。 由于 CPU 内置了自动跨接功能，所以对该接口既可以使用标准以太网电缆，又可以使用跨接以太网电缆。 连接一个 HMI 和一个 CPU 不需要以太网交换机。 更多相关信息，请参见“与编程设备通信： 建立硬件通信连接” (页 934)。
2	配置设备 更多相关信息，请参见“与编程设备通信： 组态设备” (页 934)。
3	组态 HMI 与 CPU 之间的逻辑网络连接 更多相关信息，请参见“HMI 与 PLC 通信： 组态两个设备之间的逻辑网络连接” (页 937)。

步骤	任务
4	在项目中组态 IP 地址 使用相同的组态过程；但必须为 HMI 和 CPU 组态 IP 地址。 更多相关信息，请参见“设备配置： 为项目中的 CPU 组态 IP 地址” (页 818)。
5	测试 PROFINET 网络 必须为每个 CPU 和 HMI 设备都下载相应的组态。 更多相关信息，请参见“设备配置： 测试 PROFINET 网络” (页 822)。

11.2.10.1 组态两个设备之间的逻辑网络连接

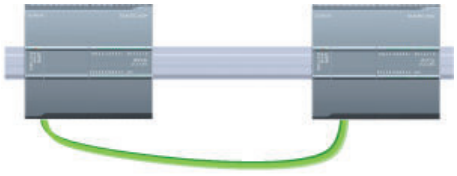
使用 CPU 配置机架后，您即准备好组态网络连接。

在“设备和网络”(Devices and Networks) 门户中，使用“网络视图”(Network view) 创建项目中各设备之间的网络连接。 首先，请单击“连接”(Connections) 选项卡，然后使用右侧的下拉框选择连接类型（例如 ISO-on-TCP 连接）。

要创建 PROFINET 连接，单击第一个设备上的绿色 (PROFINET) 框，然后拖出一条线连接到第二个设备上的 PROFINET 框。 松开鼠标按钮，即可创建 PROFINET 连接。

有关详细信息，请参见“设备配置： 创建网络连接” (页 811)。

11.2.11 PLC 到 PLC 通信



通过使用 TSEND_C 和 TRCV_C 指令，一个 CPU 可与网络中的另一个 CPU 进行通信。

设置两个 CPU 之间的通信时必须考虑以下事宜：

- 组态/设置： 需要进行硬件配置。
- 支持的功能： 向对等 CPU 读/写数据
- 一对一通信不需要以太网交换机；网络中有两个以上的设备时需要以太网交换机。

表格 11-57 组态两个 CPU 之间的通信时所需的步骤

步骤	任务
1	建立硬件通信连接 通过 PROFINET 接口建立两个 CPU 之间的物理连接。 由于 CPU 内置了自动跨接功能，所以对该接口既可以使用标准以太网电缆，又可以使用跨接以太网电缆。 连接两个 CPU 时不需要以太网交换机。 更多相关信息，请参见“与编程设备通信： 建立硬件通信连接” (页 934)。
2	配置设备 必须组态项目中的两个 CPU。 更多相关信息，请参见“与编程设备通信： 组态设备” (页 934)。
3	组态两个 CPU 之间的逻辑网络连接 更多相关信息，请参见“PLC 与 PLC 通信： 组态两个设备之间的逻辑网络连接” (页 938)。
4	在项目中组态 IP 地址 使用相同的组态过程；但必须为两个 CPU（例如，PLC_1 和 PLC_2）组态 IP 地址。 更多相关信息，请参见“设备配置： 为项目中的 CPU 组态 IP 地址” (页 818)。
5	组态传送（发送）和接收参数 必须在两个 CPU 中均组态 TSEND_C 和 TRCV_C 指令，才能实现两个 CPU 之间的通信。 更多相关信息，请参见“组态两个 CPU 之间的通信： 组态传送（发送）和接收参数” (页 939)。
6	测试 PROFINET 网络 必须为每个 CPU 都下载相应的组态。 更多相关信息，请参见“设备配置： 测试 PROFINET 网络” (页 822)。

11.2.11.1 组态两个设备之间的逻辑网络连接

使用 CPU 配置机架后，您即准备好组态网络连接。

在“设备和网络”(Devices and Networks) 门户中，使用“网络视图”(Network view) 创建项目中各设备之间的网络连接。 首先，请单击“连接”(Connections) 选项卡，然后使用右侧的下拉框选择连接类型（例如 ISO-on-TCP 连接）。

要创建 PROFINET 连接，单击第一个设备上的绿色 (PROFINET) 框，然后拖出一条线连接到第二个设备上的 PROFINET 框。 松开鼠标按钮，即可创建 PROFINET 连接。

有关详细信息，请参见“设备配置： 创建网络连接” (页 811)。

11.2.11.2 组态两台设备间的本地/伙伴连接路径

组态常规参数

在通信指令的“属性”(Properties) 组态对话框中指定通信参数。 只要选中了该指令的任何一部分，此对话框就会出现在页面底部附近。

更多相关信息，请参见“设备配置： 组态本地/伙伴连接路径 (页 812)”。

在“连接参数”(Connection parameters) 对话框的“地址详细信息”(Address Details) 部分，定义要使用的 TSAP 或端口。 在“本地 TSAP”(Local TSAP) 字段中输入 CPU 中连接的 TSAP 或端口。 在“伙伴 TSAP”(Partner TSAP) 字段下输入为伙伴 CPU 中的连接分配的 TSAP 或端口。

11.2.11.3 组态传送（发送）和接收参数

通信块（例如 TSEND_C 和 TRCV_C）用于建立两个 CPU 之间的连接。 在 CPU 可进行 PROFINET 通信前，必须组态传送（或发送）消息和接收消息的参数。 这些参数决定了在向目标设备传送消息或从目标设备接收消息时的通信工作方式。

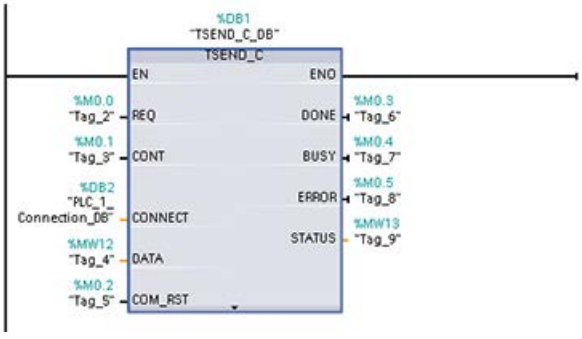
组态 TSEND_C 指令传送（发送）参数

TSEND_C 指令

TSEND_C 指令 (页 843)可创建与伙伴站的通信连接。 通过该指令可设置和建立连接，并会在通过指令断开连接前一直自动监视该连接。 TSEND_C 指令兼具 TCON、TDISCON 和 TSEND 指令的功能。

通过 STEP 7 中的设备配置，可以组态 TSEND_C 指令传送数据的方式。 首先，从“通信”(Communications) 文件夹的“指令”(Instructions) 任务卡中将该指令插入程序中。
TSEND_C 指令将与“调用选项”(Call options) 对话框一起显示，在该对话框中可以分配用于存储该指令参数的 DB。

可以为输入和输出分配变量存储位置，如下图所示：



组态常规参数

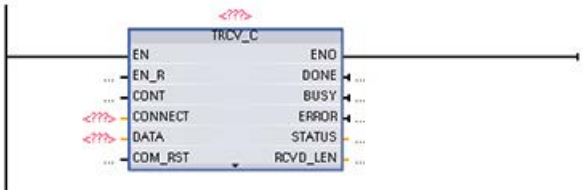
在 TSEND_C 指令的 “属性组态”(Properties configuration) 对话框中指定通信参数。 只要选中了 TSEND_C 指令的任何一部分，此对话框就会出现在页面底部附近。

组态 TRCV_C 指令接收参数

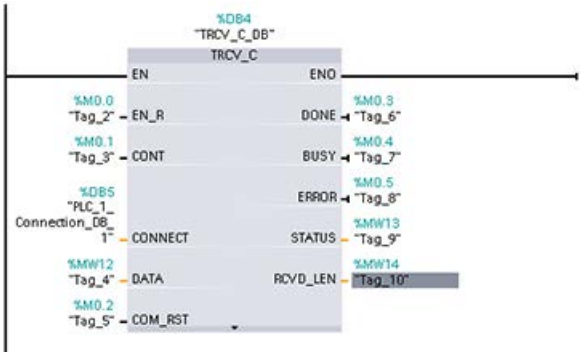
TRCV_C 指令

TRCV_C 指令 (页 843)可创建与伙伴站的通信连接。 通过该指令可设置和建立连接，并会在通过指令断开连接前一直自动监视该连接。 TRCV_C 指令兼具 TCON、TDISCON 和 TRCV 指令的功能。

通过 STEP 7 中的 CPU 组态，可以组态 TRCV_C 指令接收数据的方式。 首先，从 “通信”(Communications) 文件夹的 “指令”(Instructions) 任务卡中将该指令插入程序中。 TRCV_C 指令将与 “调用选项”(Call options) 对话框一起显示，在该对话框中可以分配用于存储该指令参数的 DB。



可以为输入和输出分配变量存储位置，如下图所示：



组态常规参数

在 TRCV_C 指令的 “属性组态”(Properties configuration) 对话框中指定通信参数。 只要选中了 TRCV_C 指令的任何一部分，此对话框就会出现在页面底部附近。

11.2.12 配置 CPU 和 PROFINET IO 设备

11.2.12.1 添加 PROFINET IO 设备

添加 PROFINET IO 设备

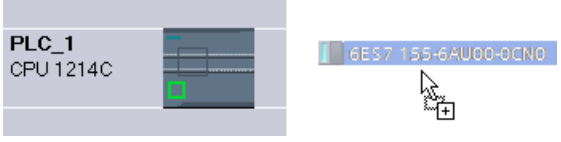
在“设备和网络”(Devices and Networks) 门户中，使用硬件目录添加 PROFINET IO 设备。

说明

要添加一个 PROFINET IO 设备，可以使用 STEP 7 Professional/Basic，V11 或更高版本。

例如，在硬件目录中展开下列容器以添加 ET 200SP IO 设备：分布式 I/O、ET 200SP、接口模块和 PROFINET。此时可从 ET 200SP 设备列表中选择接口模块（按零件号排序）并添加 ET 200SP IO 设备。

表格 11-58 将 ET 200SP IO 设备添加到设备组态

插入 IO 设备	结果
	

现在可将 PROFINET IO 设备连接到 CPU：

1. 右键单击设备上的“未分配”(Not assigned) 链接，然后从上下文菜单中选择“分配新的 IO 控制器”(Assign new IO controller) 以显示“选择 IO 控制器”(Select IO controller) 对话框。
2. 从项目的 IO 控制器列表中选择 S7-1200 CPU（本例中为“PLC_1”）。
3. 单击“确定”(OK) 创建网络连接。

也可以转到“设备和网络”(Devices and Networks) 门户并使用“网络视图”(Network view) 创建项目中各设备之间的网络连接：

1. 要创建 PROFINET 连接，单击第一个设备上的绿色 (PROFINET) 框，然后拖出一条线连接到第二个设备上的 PROFINET 框。
2. 松开鼠标按钮，即可创建 PROFINET 连接。

更多相关信息，请参见“设备配置：组态通信 CPU” (页 171)。

11.2.12.2 分配 CPU 和设备名称

分配 CPU 和设备名称

设备之间的网络连接还会将 PROFINET IO 设备分配给 CPU，从而 CPU 能够控制相应设备。要更改该分配，请单击 PROFINET IO 设备上显示的“PLC 名称”(PLC Name)。将打开一个对话框，允许用户从当前 CPU 上断开 PROFINET IO 设备以重新分配设备，或根据需要保持不分配状态。

PROFINET 网络中的设备在分配名称后才可与 CPU 连接。如果 PROFINET 设备尚未分配名称，或要更改该设备的名称，则可使用“网络视图”(Network view) 为 PROFINET 设备分配名称。可通过右键单击 PROFINET IO 设备并选择“分配设备名称”(Assign device name) 来实现。

对于各 PROFINET IO 设备，必须在 STEP 7 项目和 PROFINET 网络的 PROFINET IO 设备中为该设备分配相同的名称。（可以使用 PROFINET 网络中的 STEP 7“在线和诊断” (Online & diagnostics) 工具或 PRONETA 调试、组态及诊断工具分配设备名称。）如果名称缺失或两个位置中的名称不匹配，则 PROFINET IO 数据交换模式将不会运行。更多相关信息，请参见“在线和诊断工具：在线为 PROFINET 设备分配名称 (页 1387)”。

11.2.12.3 分配 Internet 协议 (IP) 地址

分配 IP 地址

在 PROFINET 网络中，每个设备还必须具有一个 Internet 协议 (IP) 地址。该地址使设备可以在更加复杂的路由网络中传送数据：

- 如果编程或其它网络设备使用连接到工厂 LAN 的板载适配器卡或连接到隔离网络的以太网到 USB 适配器卡，则必须为它们分配 IP 地址。更多相关信息，请参见“为编程设备和网络设备分配 IP 地址” (页 815)。
- 还可以在线为 CPU 或网络设备分配 IP 地址。这在进行初始设备配置时尤其有用。更多相关信息，请参见“在线为 CPU 分配 IP 地址” (页 817)。
- 组态项目中的 CPU 或网络设备后，可以组态 PROFINET 接口的参数及其 IP 地址。更多相关信息，请参见“为项目中的 CPU 组态 IP 地址” (页 818)。

11.2.12.4 组态 IO 循环时间

组态 IO 循环时间

CPU 会在“IO 循环”期间为 PROFINET IO 设备提供新数据。可以单独组态每台设备的更新时间，更新时间可确定在 CPU 和设备之间交换数据的时间间隔。

在 PROFINET 网络上每台设备的默认设置中，由 STEP 7 根据要交换的数据量和分配给控制器的设备数自动计算“IO 循环”更新时间。如果不希望自动计算更新时间，则可以更改此设置。

在 PROFINET IO 设备的“属性”(Properties) 组态对话框中指定“IO 循环”(IO cycle) 参数。只要选中了该指令的任何一部分，此对话框就会出现在页面底部附近。

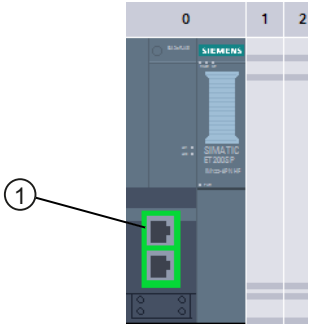
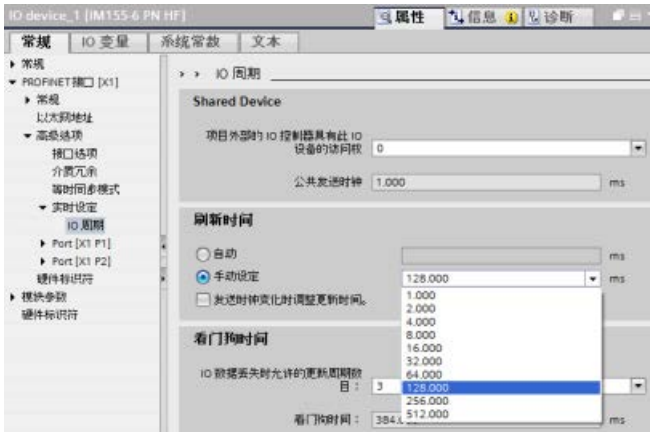
在 PROFINET IO 设备的“设备视图”(Device view) 中单击 PROFINET 端口。在“PROFINET 接口”(PROFINET Interface) 对话框中，通过以下菜单选项访问“IO 循环”(IO cycle) 参数：

- “高级选项”(Advanced options)
- “实时设置”(Realtime settings)
- “IO 循环”(IO cycle)

通过以下选项定义 IO 循环“更新时间”(Update time)：

- 要自动计算合适的更新时间，请选择“自动”(Automatic)。
- 要亲自设置更新时间，请选择“可进行设置”(Can be set) 并输入所需更新时间（单位为 ms）。

表格 11-59 组态 ET 200SP PROFINET IO 循环时间

ET 200SP PROFINET IO 设备	ET 200SP PROFINET IO 循环对话框
	

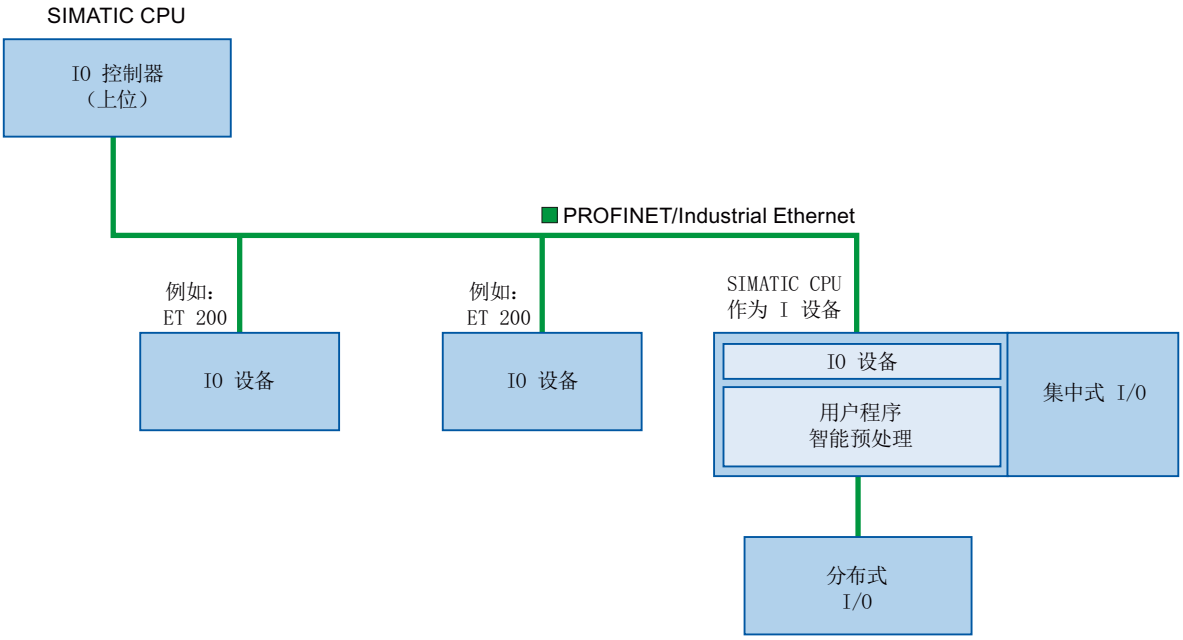
① PROFINET 端口

11.2.13 组态 CPU 和 PROFINET 智能设备

11.2.13.1 智能设备功能

CPU 的“智能设备”（智能 IO 设备）功能简化了与 IO 控制器的数据交换以及 CPU 操作（例如，用作子过程的智能预处理单元）。智能设备作为 IO 设备链接到“上位”IO 控制器。

预处理由 CPU 上的用户程序完成。 集中式或分布式（PROFINET IO 或 PROFIBUS DP）I/O 中采集的过程值由用户程序进行预处理，并通过 PROFINET IO 接口提供给上位站的 CPU。



“智能设备”的命名约定

在本说明的其余部分，将把具有智能设备功能的 CPU 或通信处理器简称为“智能设备”。

11.2.13.2 智能设备的性能和优势

应用范围

智能设备的应用领域：

- 分布式处理：
  可以将复杂自动化任务划分为较小的单元/子过程。这样便可对各个过程进行管理，从而简化了子任务。
- 分隔子过程：
  通过使用智能设备，可以将分布广泛的大量复杂过程划分为具有可管理的接口的多个子过程。如有必要，这些子过程可以存储在各个 STEP 7 项目中，随后可将这些项目合并成一个主站项目。
- 专有知识保护：
  对于智能设备的接口描述，只能通过 GSD 文件提供组件，而不能通过 STEP 7 项目来提供。由于不再需要发布，因此用户可以保护其程序。

性能

智能设备的性能：

- 取消 STEP 7 项目的链接：
  智能设备的创建者和用户可具有完全独立的 STEP 7 自动化项目。GSD 文件构成 STEP 7 项目间的接口。这样便可通过标准化接口链接到标准 IO 控制器。
- 实时通信：
  通过 PROFINET IO 接口为智能设备提供确定性的 PROFINET IO 系统。

优点

智能设备具有以下优势：

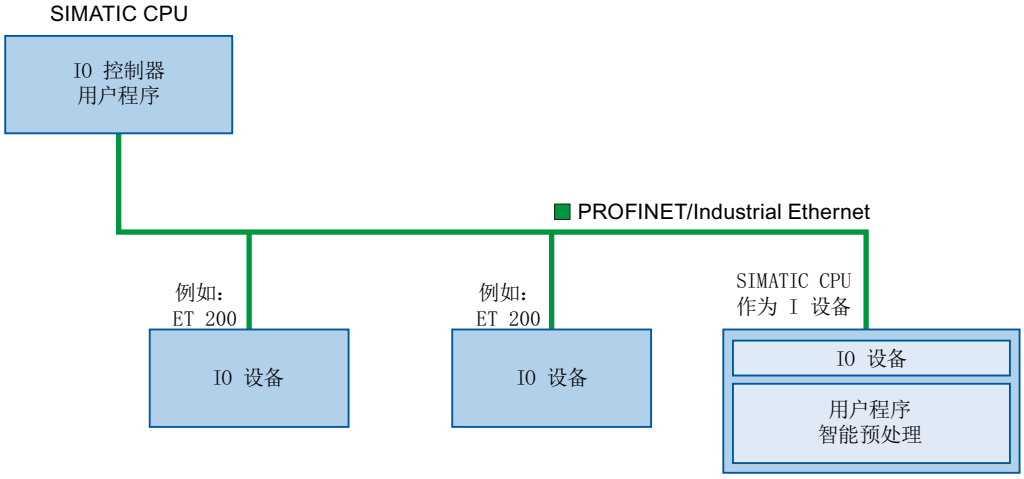
- 轻松链接 IO 控制器
- 在 IO 控制器间进行实时通信
- 将计算量分配给智能设备，减轻 IO 控制器的负担。
- 由于在局部处理过程数据，通信负载降低。
- 易于管理，原因是可以在单独的 STEP 7 项目中处理子任务。

11.2.13.3 智能设备的特性

智能设备就像标准 IO 设备那样被集成到 IO 系统中。

不带下级 PROFINET IO 系统的智能设备

智能设备没有自己的分布式 I/O。充当 IO 设备角色的智能设备的组态和参数分配对于分布式 I/O 系统而言是相同的（例如 ET 200）。



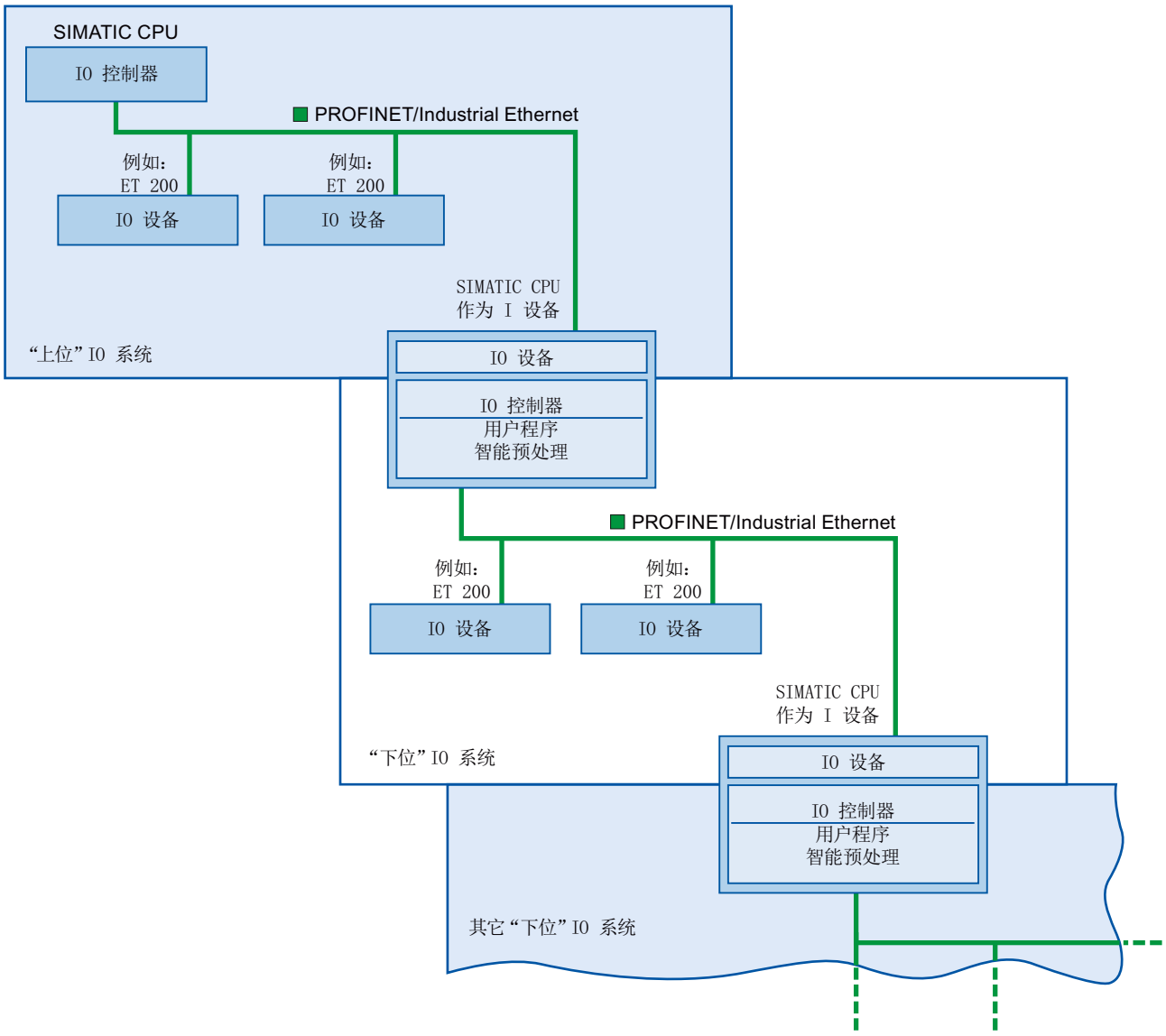
不带下级 PROFINET IO 系统的智能设备

根据组态的不同，智能设备除了可以作为 IO 设备之外，也可以用作 PROFINET 接口上的 IO 控制器。

这意味着，智能设备可通过其 PROFINET 接口成为上层 IO 系统的一部分，并可作为 IO 控制器来支持自身的下层 IO 系统。

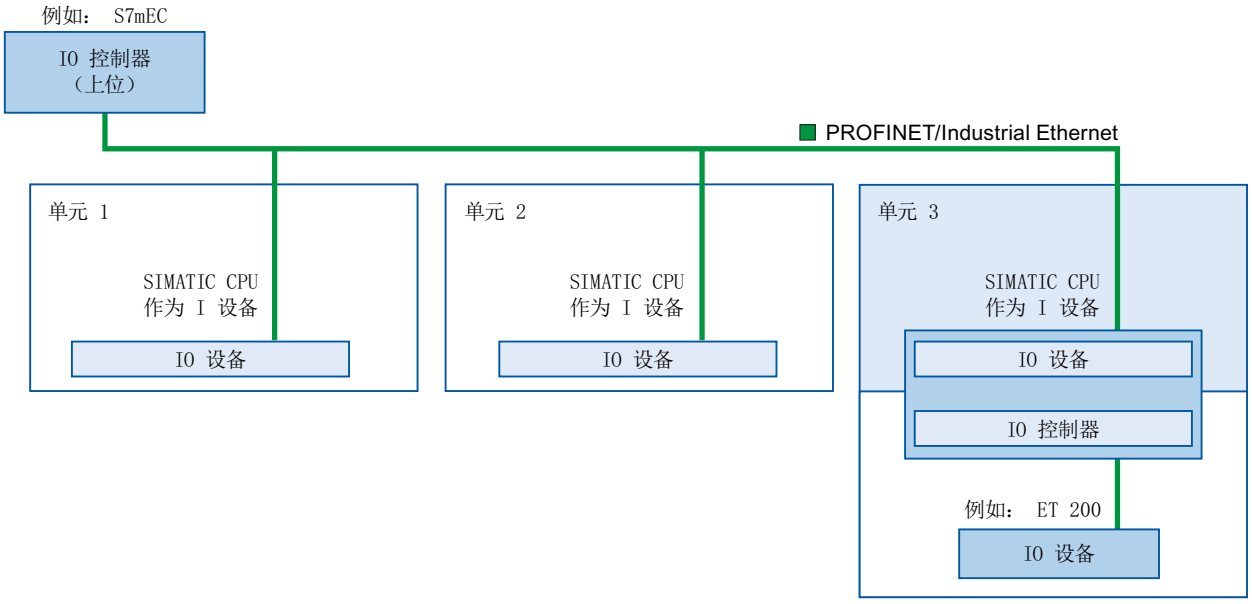
反过来，下位 IO 系统又可以包含智能设备（见下图）。 这样就可实现分层的 IO 系统结构。

除用作 IO 控制器外，智能设备还可通过 PROFIBUS 接口用作下位 PROFIBUS 系统的 DP 主站。


SIMATIC CPU
IO 控制器
PROFINET/Industrial Ethernet
例如：
ET 200
IO 设备
例如：
ET 200
IO 设备
SIMATIC CPU
作为 I 设备
“上位”IO 系统
IO 设备
IO 控制器
用户程序
智能预处理
PROFINET/Industrial Ethernet
例如：
ET 200
IO 设备
例如：
ET 200
IO 设备
SIMATIC CPU
作为 I 设备
“下位”IO 系统
IO 设备
IO 控制器
用户程序
智能预处理
其它“下位”IO 系统

示例： 作为 IO 设备和 IO 控制器的智能设备

我们以印刷过程为例来介绍作为 IO 设备和 IO 控制器的智能设备。 智能设备可控制一个单元（一个子过程）。 例如，可通过一个单元在印刷好的材料包装中插入其它纸张（如活页或小册子）。



单元 1 和单元 2 各有一个带集中式 I/O 的智能设备。智能设备与分布式 I/O 系统（如 ET 200）一起构成单元 3。

智能设备上的用户程序负责对过程数据进行与处理。 对于此任务来说，智能设备的用户程序需要来自上位 IO 控制器的默认设置（例如，控制数据）。 智能设备为上位 IO 控制器提供结果（例如，子任务的状态）。

11.2.13.4 上位 IO 系统与下位 IO 系统之间的数据交换

传送区是与智能设备 CPU 的用户程序之间的接口。用户程序对输入进行处理并输出处理结果。

传送区提供用于 IO 控制器与智能设备之间通信的数据。传送区包含一个可在 IO 控制器与智能设备之间不断进行交换的信息单元。有关传送区的组态与使用的详细信息，请参见“组态智能设备” (页 953)部分。

当控制器与智能设备之间的网络连接断开时，输入传送区存在不同的处理方式。

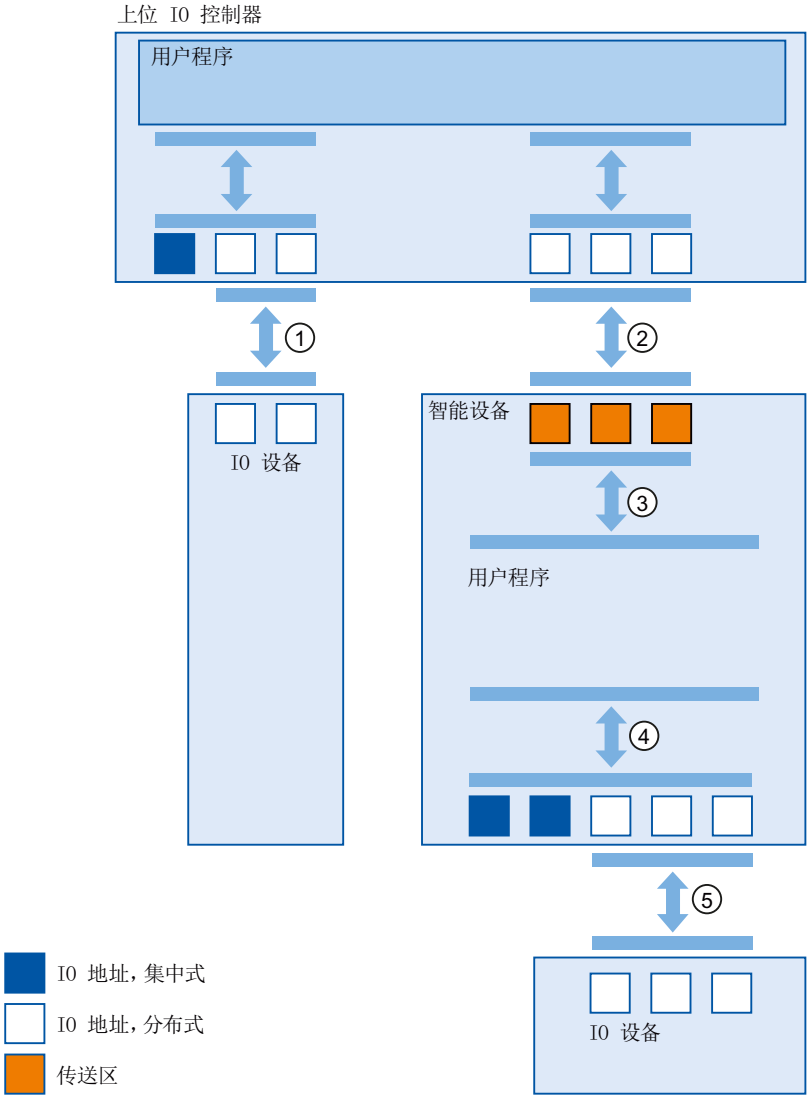
当网络连接断开时，CPU 将零写入控制器上的输入传送区。在智能设备上，输入传送区保持其最后的值。

您可以组态系统以避免在常规智能设备（非共享智能设备）上出现这种情形。为此，针对进入事件，清除“机架或站故障 OB”中智能设备的输入传送区。请按以下步骤操作：

1. 在您的项目中添加一个“机架或站故障 OB”(Rack or Station Failure OB)。（该 OB 编号默认为 OB 86）。
2. 向 OB 添加逻辑，从而当 LADDR 的启动变量指示智能设备硬件 ID 的值并且 Event_Class 的启动变量指示一个“进入”事件时，将智能设备的输入值写为零：
   – 您可以在“系统常量”(System constants) 选项卡的默认变量表中找到智能设备硬件 ID。硬件 ID 为“HW_Device”类型，变量名指示其为智能设备（例如，“Local~PROFINET_interface_1~IODevice”）。
   – Event_Class 中的“16#39”值表示一个“进入”事件。如果“Event_Class”输入变量包含“16#39”值，这表示“机架或站故障 OB”当前处于激活状态（与清除状态相反）。

数据交换流

下图显示了上位 IO 系统和下位 IO 系统之间的数据交换。下面内容根据编号介绍了各种通信关系：



① **上位 IO 控制器与普通 IO 设备之间的数据交换**

在这种方式中，IO 控制器和 IO 设备可通过 PROFINET 来交换数据。

② **上位 IO 控制器与智能设备之间的数据交换**

在这种方式中，IO 控制器和智能设备可通过 PROFINET 来交换数据。

上位 IO 控制器与智能设备之间的数据交换基于常规 IO 控制器与 IO 设备之间的关系。

对于上位 IO 控制器，智能设备的传送区代表某个预组态站的子模块。

IO 控制器的输出数据是智能设备的输入数据。与此类似，IO 控制器的输入数据是智能设备的输出数据。

③ 用户程序与传送区之间的传输关系

在这种方式中，用户程序与传送区交换输入和输出数据。

④ 用户程序与智能设备的 I/O 之间的数据交换

在这种方式中，用户程序与集中式/分布式 I/O 交换输入和输出数据。

⑤ 智能设备与下位 IO 设备之间的数据交换

在这种方式中，智能设备与其 IO 设备交换数据。数据通过 PROFINET 传送。

11.2.13.5 组态智能设备

主要有两种组态方法：

- 组态某一项目的智能设备
- 组态另一项目或工程系统中使用的智能设备。

使用 STEP 7，可以通过将已组态的智能设备导出到 GSD 文件，为其它项目或工程组态系统组态一个智能设备。导入其它项目或工程系统中 GSD 文件的方法与导入其它 GSD 文件的方法相同。用于数据交换的传送区域与其它数据都存储在该 GSD 文件中。

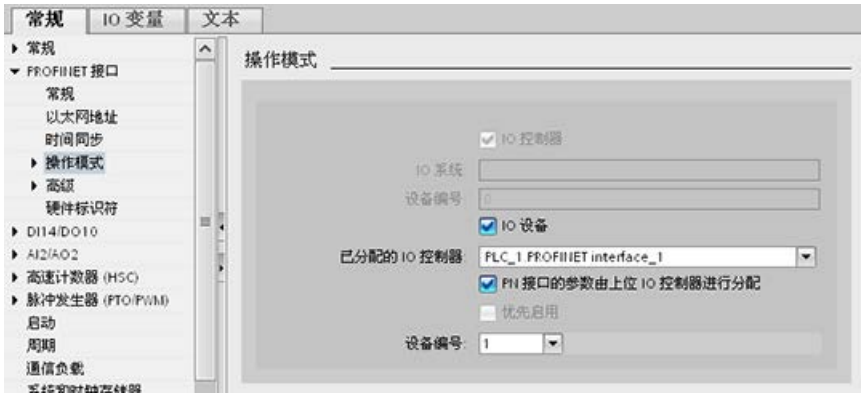
说明

当使用 S7-1200 作为共享智能设备和控制器，确保增加 PROFINET 智能设备和 PROFINET IO 更新时间以便清除通信性能的影响。当选择 2 ms 作为单一 PROFINET 智能设备时间的更新时间并选择 2 ms 作为单一 PROFINET IO 时间的更新时间，系统会非常稳定且运行良好。

组态某一项目的智能设备

1. 将 PROFINET CPU 从硬件目录拖放到网络视图中。
2. 将同样可组态为 IO 设备的 PROFINET CPU 从硬件目录拖放到网络视图中。此设备已组态为智能设备（例如 CPU 1215C）。
3. 为该智能设备选择 PROFINET 接口。
4. 在区域导航的巡视窗口中选择“工作模式”(Operating mode)，然后勾选“IO 设备”(IO device) 复选框。

5. 此时在“分配的 IO 控制器”(Assigned IO controller) 下拉列表中选择 IO 控制器。
   选择 IO 控制器后，网络视图中将显示两个设备间的网络和 IO 系统。



6. 通过“由上位 IO 控制器分配 PN 接口参数”(Parameter assignment of PN interface by higher-level IO controller) 复选框，指定接口参数由智能设备本身分配还是由上位 IO 控制器分配。如果操作带有下位 IO 系统的智能设备，则无法由上位 IO 控制器来分配智能设备 PROFINET 接口参数（如端口参数）。

7. 组态传送区。传送区位于区域导航“智能设备通信”(I-device communication) 部分：
   – 单击“传送区”(Transfer area) 列的第一个字段。STEP 7 会分配一个可随时更改的默认名称。
   – 选择通信关系类型：当前仅可选择 CD 或 F-CD。
   – 地址会自动预置；必要时可更正地址并确定将一致传输的传送区长度。

8. 在区域导航中，将为每个传送区创建一个单独的条目。选择其中一个条目后，便可以调整、更正传送区的详细信息并向其中添加注释。

说明

如果将 S7-1200 作为智能设备组态，传送区的最大大小为 1024 输入或输出字节。制约因素有可能取决于控制设备上的本地 I/O 以及地址空间限制。

使用 GSD 文件组态智能设备

如果在另一个项目或工程系统中使用智能设备，请按上述步骤组态上位 IO 控制器和智能设备。

但请在传送区组态完成后单击“导出”(Export) 按钮，为智能设备中新建一个 GSD 文件。此 GSD 文件代表其它项目组态的智能设备。

“导出”(Export) 按钮位于巡视窗口的“智能设备通信”(I-device communication) 部分。

随即会编译硬件组态并打开导出对话框。

在显示的字段中为智能设备代理分配名称以及描述。单击“导出”(Export) 按钮完成此过程。

最后，导入 GSD 文件（例如导入到另一项目中）。

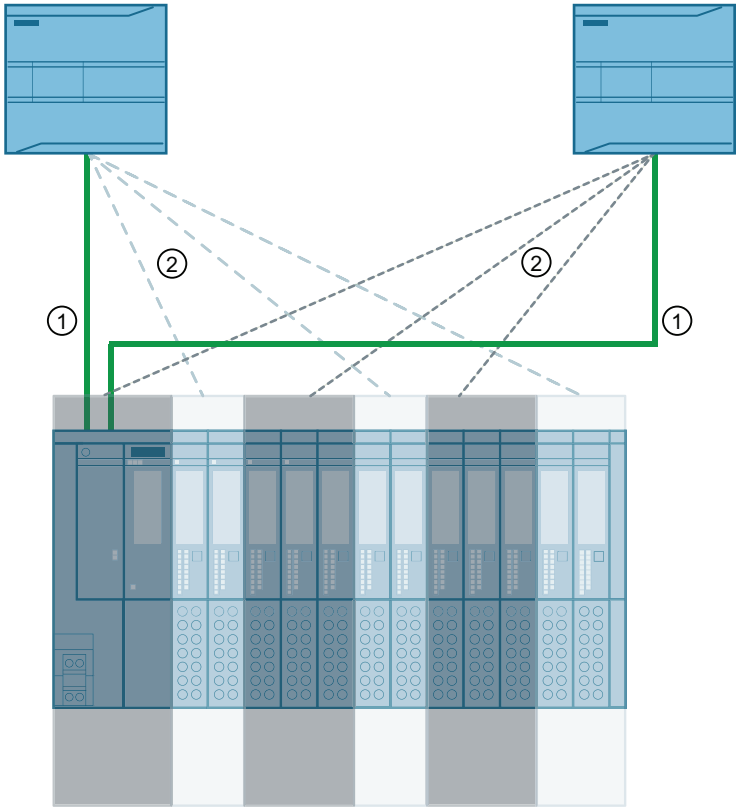
11.2.14 共享设备

11.2.14.1 共享设备的功能

大型或分布较广的分布式系统经常使用数量众多 IO 控制器。

不使用“共享设备”功能，I/O 设备的每个 IO 模块都会分配给同一个 IO 控制器。如果互相之间物理封闭的传感器必须向不同的 IO 控制器提供数据，则需要多个 IO 设备。

“共享设备”功能允许 IO 设备的模块或子模块在不同的 IO 控制器中进行划分。这充分体现了灵活的自动化理念。例如，可以将多个邻近的 I/O 模块组合到一个 IO 设备中。



① PROFINET

② 逻辑分配

原理

对共享设备子模块的访问将在各个 IO 控制器之间分配。每个共享设备子模块以独占方式分配给一个 IO 控制器。

要求（GSD 组态）

- STEP 7 V12 + SP 1 或更高版本
- S7-1200 CPU 固件 V4.1 或更高版本作为 IO 控制器
- IO 设备支持共享设备功能，例如接口模块 IM 155-5 PN ST
- 已安装用于组态 IO 设备的 GSD 文件
- 组态为智能设备的 S7-1200 CPU 支持共享设备功能。必须从 STEP 7（V5.5 及以上版本）导出智能设备的 PROFINET GSD 文件，然后将其导入到 STEP 7 (TIA Portal)。

组态访问权

IO 设备必须存在于多个项目中，IO 设备的模块或子模块才能分配给不同的 IO 控制器。每个 IO 控制器需要一个单独的项目。

使用接口模块的“共享设备”(Shared device) 参数确定 IO 控制器有权访问的模块或子模块：

- 如果本地 IO 控制器有权访问组态的模块，则从列表中选择 IO 控制器的名称。
- 如果 IO 控制器来自另一个项目，而不是有权访问已组态模块的本地 IO 控制器，则选择条目“---”。

如果一个项目中的每个模块或子模块正好分配给一个 IO 控制器，则访问的组态一致。

模块或子模块分配给另一个 IO 控制器

下图描述了本地 IO 控制器的“共享设备”(Shared device) 参数的“---”设置的后果。

在本例中，本地 IO 控制器无法访问通过这种方法组态的模块。这表明：

- 模块或子模块没有数据交换
- 没有收到报警或诊断，这意味着在线视图中未显示诊断状态
- 模块或子模块没有参数分配

设置实时属性

STEP 7 计算通信负载，然后计算产生的更新时间。必须在项目中输入项目外部 IO 控制器的编号，在该项目中共享设备的 PROFINET 接口分配给 IO 控制器，以便可使用共享设备组态进行计算。

共享设备可能的最大 IO 控制器数目取决于设备。此数目存储在共享设备的 GSD 文件中。

可以通过 S7-1200 CPU 设置非常短的发送时钟（最短 为 1 ms）作为 IO 控制器。此发送时钟可以短于共享设备支持的最短发送时钟。在这种情况下，IO 控制器使用它支持的发送时钟来运行共享设备（发送时钟调整）。

示例：一个 CPU 支持的最短的发送时钟为 1 ms。一个组态的 IO 设备支持最短 1.25 ms 的发送时钟；另一个 IO 设备支持最短 1 ms 的发送时钟。此时，可将 CPU 的短发送时钟设置为 1 ms。CPU 使用 1.25 ms 的发送时钟运行“慢速”IO 设备。

组态规则

- 使用共享设备的 IO 控制器在不同的项目中创建。在每个项目中，必须注意应在每个站中对共享设备进行相同组态。只有一个 IO 控制器可以永远访问子模块。组态不一致会导致共享设备发生故障。
- 仅当模块或子模块分配给同一项目中的 I/O 控制器时，才能编辑模块或子模块的 I/O 地址。
- 共享设备在每个项目中必须具有相同的 IP 参数和相同的设备名称。
- 对于有权访问共享设备的所有 IO 控制器，发送时钟必须相同。
- 连接共享设备子网的 S7 子网 ID 在所有项目中必须相同。
- 仅当共享设备的 PROFINET 接口分配给本地 IO 控制器时，以下功能才可用：
  - 优先化启动
  - 端口属性的参数分配

限制条件

因为共享设备组态分布在多个项目中，所以有以下限制条件：

- 未分配给此 IO 控制器的模块或子模块的地址在有权访问共享设备的每个 IO 控制器的地址总览中都不显示。
- 在进行一致性检查时，共享设备的组态限制计算不考虑未分配的模块或子模块。因此，用户必须自行验证未超过子模块的最大数量和共享设备循环 IO 数据的最大数量。关于这些数量的最大值，参见所用设备的文档。
- STEP 7 中不检测一个模块或子模块分配给多个 IO 控制器之类的组态错误。
- 加载共享设备组态的 CPU 没有任何关于 IO 设备是否为共享设备的信息。因此加载的组态中会缺少分配给其它 IO 控制器和其它 CPU 的模块或子模块。所以这些模块或子模块既不会显示在 CPU Web 服务器中，也不会显示在 CPU 显示屏中。

11.2.14.2 示例： 组态共享设备（GSD 组态）

此示例介绍如何使用 STEP 7 V13 SP1 或更高版本将分布式 I/O 系统组态为共享设备。

对于不同的 IO 控制器，可使用不同工程组态工具进行“分布式”组态。下述步骤基于 STEP 7 V13 SP1 或更高版本，并且仅限于组态 S7-1200 系列的两个 IO 控制器，这两个控制器共享一个共享设备。

该示例创建各含有一个 IO 控制器的两个项目：

- Controller1
- Controller2

必须在两个项目中都创建共享设备，即便 IO 设备在物理上是同一个。

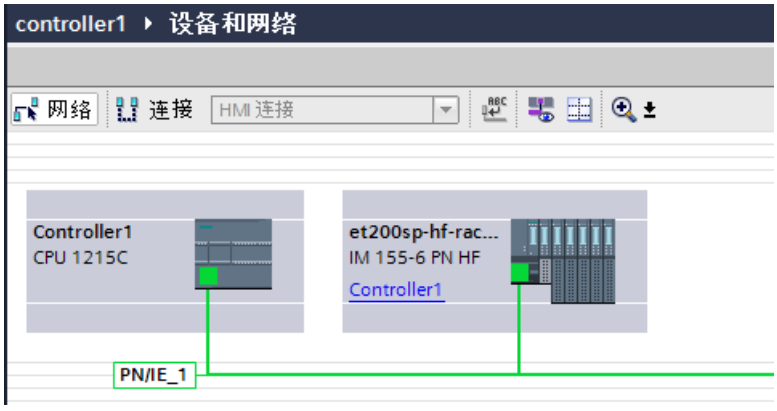
要求

- STEP 7 V13 SP1 或更高版本
- IO 设备支持共享设备功能（例如，ET 200SP IM 155-6 PN HF V3.1）。
- 已安装用于将 IO 设备组态为共享设备的 GSD 文件。

操作步骤：创建项目 1

要使用共享设备创建第一个项目，请按以下步骤操作：

1. 启动 STEP 7。
2. 创建名为“Controller1”的新项目。
3. 从网络视图的硬件目录中插入一个 CPU 1215C。将其命名为“Controller1”。
4. 从硬件目录插入具有“共享设备”功能的 IO 设备（例如 ET 200SP）（硬件目录：其它现场设备 > PROFINET IO > I/O）。
5. 将 IO 控制器“Controller1”分配给 IO 设备。



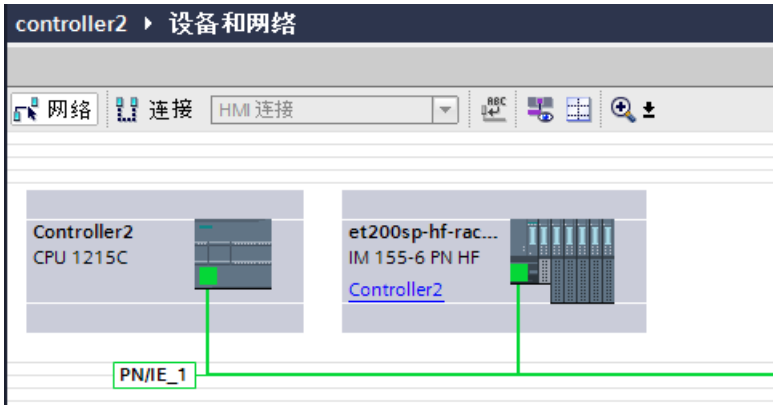
6. 双击 IO 设备并将硬件目录中的所有必需模块和子模块插入到设备总览表中。
7. 分配模块参数。
8. 保存项目。

操作步骤：创建项目 2

要使用共享设备创建第二个项目，请按以下步骤操作：

1. 再次启动 STEP 7。
   将打开 STEP 7 的一个新实例。
2. 在新实例中，创建一个名为“Controller2”的新项目。
3. 在网络视图中插入 CPU 1215C。将其命名为“Controller2”。

4. 复制项目“Controller1”中的 IO 设备，并将其插入到项目“Controller2”的网络视图中。
5. 将 IO 控制器“Controller2”分配给 IO 设备。



6. 保存项目。

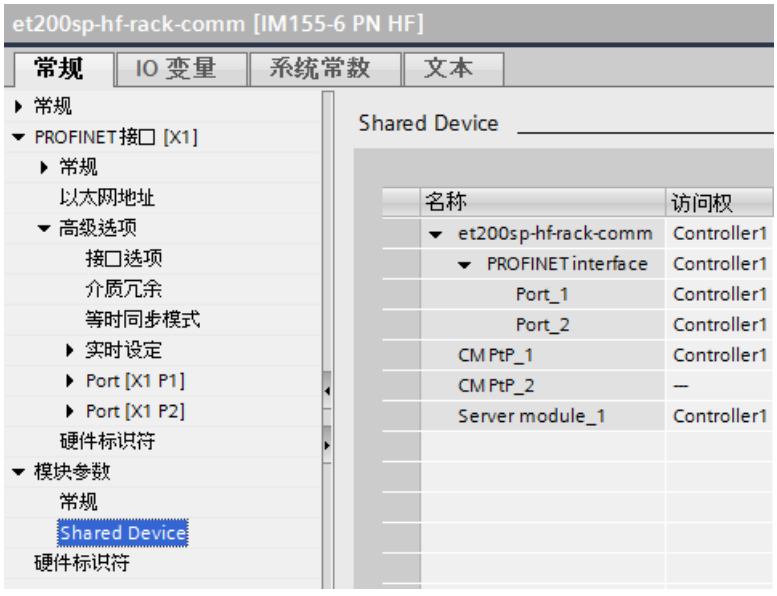
两个项目现在有结构相同的 IO 设备，必须在下一步中为不同类型的 IO 控制器访问组态该 IO 设备。

操作步骤：组态对共享设备的访问

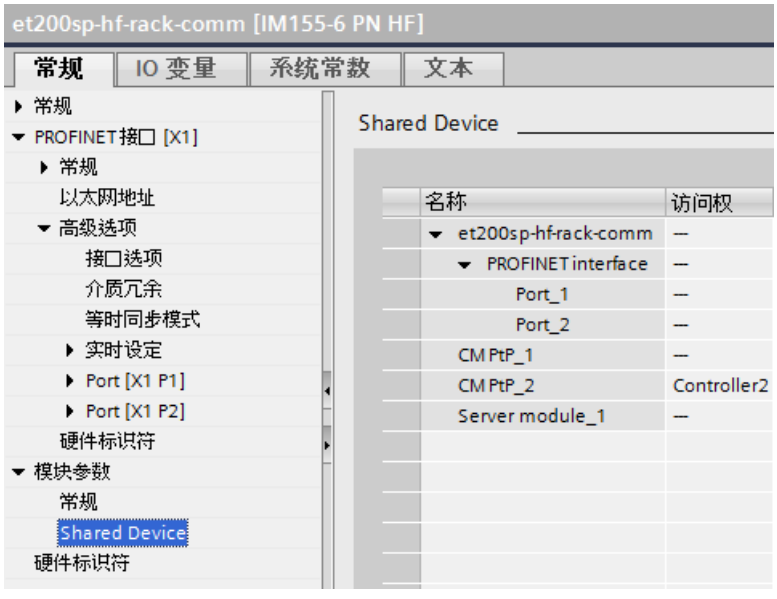
插入到共享设备中的模块和子模块会自动分配到本地 CPU。要更改分配情况，请按以下步骤操作：

1. 选择项目“Controller1”的网络或设备视图中的接口模块。
2. 在巡视窗口中选择 “共享设备”(Shared device) 区域。
   将会出现一张表，显示有权访问所有已组态模块的各个模块或子模块的 CPU。默认设置是本地 CPU 有权访问所有模块和子模块。

3. 保持仍保留在本地 CPU 的地址范围内的所有模块和子模块的“Controller1”设置。
从“Controller2”项目 (Controller2) 中，为将位于 CPU 地址范围内的所有模块和子模块选择设置“---”。这意味着项目外的 IO 控制器将有权访问模块或子模块。



4. 选择项目“Controller2”的网络或设备视图中的接口模块。
5. 在巡视窗口中选择 “共享设备”(Shared device) 区域。
将会出现一张表，显示有权访问所有已组态模块的各个模块或子模块的 CPU。
6. 从“Controller1”项目 (Controller1) 中，为将位于 CPU 地址范围内的所有模块和子模块选择设置“---”。



7. 最后，检查两个项目中每个模块或子模块的访问权设置是否“完整”。这意味着如果本地 CPU 在一个项目中有访问权，那么必须在另一个项目中设置选项“---”，反之亦然。
   注：PROFINET 接口和端口的选项“---”使相关的参数为只读，无法更改。PROFINET 接口的参数和端口参数只能在其 PROFINET 接口分配给本地 CPU 的项目中编辑。无论怎样，两个项目中的端口都可以互连。
8. 检查是否为所有项目中的共享设备设置了相同的 IP 地址参数和设备名称。
   检查是否在所有项目中为连接共享设备的子网设置了相同的 S7 子网 ID（子网属性，巡视窗口中的“常规”(General) 区域）。

说明

如果更改共享设备：请在共享设备上的每个项目中进行同样的更改。确保仅一个 IO 控制器有权访问模块或子模块。

操作步骤：调整实时设置

为确保所有 IO 控制器和共享设备使用适当的发送时钟运行，并确保根据通信负载正确计算更新时间，必须调整并检查以下设置：

1. 选择其 IO 控制器有权访问 PROFINET 接口和共享设备端口的项目。
2. 在网络视图中选择共享设备的接口模块。
3. 在巡视窗口中，导航至“PROFINET 接口 > 高级选项 > 实时设置 > IO 周期”(PROFINET interface > Advanced options > Real time settings > IO cycle) 区域。
4. 在“共享设备”(Shared device) 区域中，设置项目外部 IO 控制器的数目。最大数目取决于 IO 设备（在 GSD 文件中指定）。
5. 必须为每个有权访问共享设备的模块或子模块的 IO 控制器设置相同的发送时钟：

- 如果使用 STEP 7 (TIA Portal) 组态 IO 控制器：
  - 打开相应的项目。
  - 选择 IO 控制器的 PROFINET 接口。
  - 在巡视窗口中选择“高级选项 > 实时设置 > IO 通信”(Advanced options > Real-time settings > IO communication) 区域，并设置共享的发送时钟。
- 如果使用其它工程组态工具组态 IO 控制器：
  - 在 STEP 7 (TIA Portal) 中选择共享设备的 PROFINET 接口，并在共享设备上读出发送时钟（“高级选项 > 实时设置”(Advanced options > Real-time settings) 区域）。
  - 在工程组态工具中输入读取发送时钟。

说明

如果在 STEP 7 (TIA Portal) 中组态有权访问共享设备的所有 IO 控制器，则可以在 IO 控制器上设置比共享设备支持的发送时钟更短的发送时钟（发送时钟调整）。

编译和加载

必须编译不同 IO 控制器的组态，并将其一个接一个地加载到 CPU。

由于对单独项目进行分布式组态，在访问参数分配错误时，STEP 7 不输出一致性错误。以下是错误分配访问参数的示例：

- 多个 IO 控制器可以访问同一个模块
- IP 地址参数或发送时钟不同

控制器操作之前不会显示这些错误，且这些错误将输出为组态错误。

11.2.14.3 示例：将智能设备组态为共享设备

该示例描述了如何使用 STEP 7 V13 SP1 及以上版本将 S7-1200 组态为智能设备，然后将其作为共享设备在两个项目中使用。

对于不同的 IO 控制器，可使用不同工程组态工具进行“分布式”组态。以下描述的步骤基于 STEP 7 V13 SP1，并且仅限用于组态两个 S7-1200 系列 IO 控制器，这两个 IO 控制器共享智能设备的传送区作为共享设备。智能设备本身属于 CPU 1215C。

该示例创建各含有一个 IO 控制器的三个项目：

- S7-1200-I-Device
- Controller1
- Controller2

使用 S7-1200-I-Device 项目来组态智能设备。为了分配各自上位 IO 控制器中的传输区域，可在 Controller1 和 Controller2 项目中使用 S7-1200-I-Device 的 PROFINET GSD 变量。

共享智能设备概念

共享智能设备概念需要最少三个独立项目：

- 智能设备项目：对智能设备进行组态和编程，以执行特定自动化任务。可将传输区域定义为上位控制器的 I/O 接口，并将这些传输区域分配给不同 IO 控制器。为了连接到上位 IO 控制器，需提供一个 PROFINET GSD 文件并使用传输区域来访问智能设备。
- 共享智能设备的控制器（两个项目）：在组态 PROFINET IO 系统时，使用智能设备作为 PROFINET GSD 变量，并指定 IO 控制器用于访问传送区的 I/O 地址。

说明

如果将 S7-1200 作为智能设备组态，传送区的最大大小为 1024 输入或输出字节。制约因素有可能取决于控制设备上的本地 I/O 以及地址空间限制。

智能设备

向用作智能设备的 S7-1200 CPU 分配以下参数：

- 集中式和分布式 I/O
- 所需的传输区域
- 可访问此智能设备的 IO 控制器的数量（对于共享设备，始终大于 1）

说明

可以组态不带上位 IO 控制器的智能设备。因此，在创建用户程序以从传送区编辑地址时，只能使用传送区的本地 I/O 地址（与“智能设备中的地址”一致）。除了与上一级 IO 控制器的连接外，已完全组态的智能设备将装载到 S7-1200 CPU。

用户可以从智能设备组态导出 PROFINET GSD 文件。

共享智能设备的控制器

必须在使用该共享智能设备组态 PROFINET IO 系统时所使用的所有工程组态系统中，安装从智能设备组态创建的 PROFINET GSD 文件。如果在使用该智能设备的地方都使用 STEP 7 V13 SP1 组态，那么在 STEP 7 中安装 GSD 文件即可。

可以在相关项目的 PROFINET IO 系统中将智能设备组态为一个 GSD 变量。在 STEP 7 V13 SP1 中，安装后该智能设备位于“其它现场设备 > PROFINET IO > PLC 与 CP”(Other field devices > PROFINET IO > PLCs & CPs) 下。

在每个相关项目中，指定哪些传送区被专门分配给上位 IO 控制器（默认设置：全部）。可以将其它传送区设置为“---”（不分配）。在此情况下，本地 IO 控制器无法访问此传输区域，并且您可将该传输区域分配该其它项目中的其它 IO 控制器。

要求

- STEP 7 V13 SP1 或更高版本
- IO 设备支持共享设备功能（例如，ET 200SP IM 155-6 PN HF V3.1）。
- 已安装用于将 IO 设备组态为共享设备的 GSD 文件。

操作步骤：创建 S7-1200-I-device 项目

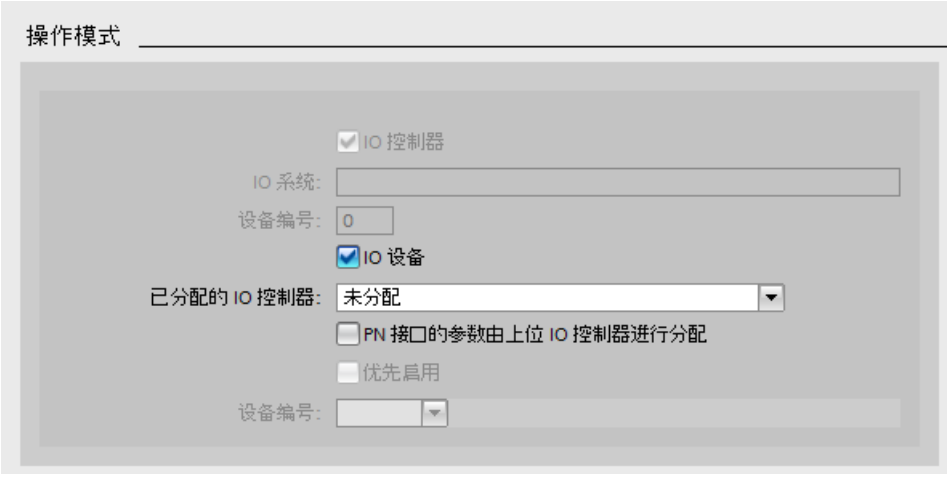
要使用共享智能设备创建项目，请按以下步骤操作：

1. 启动 STEP 7。
2. 创建一个名为“S7-1200-I-device”的新项目。

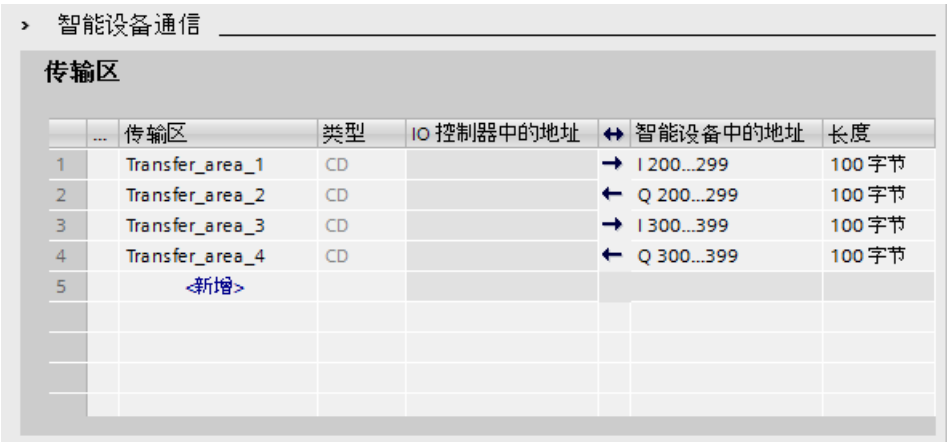
3. 从网络视图的硬件目录中插入一个 CPU 1215C。指定名称“S7-1200-I-device”。

4. 双击 IO 设备并组态所有必需的模块及子模块。

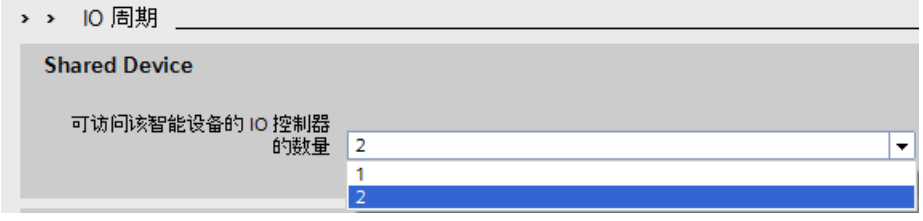
5. 分配模块参数。特别是，必须在 PROFINET 接口 [X1] 的区域中对 CPU 进行以下设置：
   – 在“操作模式”(Operating mode) 区域中启用“IO 设备”(IO device) 选项。



   – 在“操作模式”>“智能设备组态”("Operating mode" > "I-device configuration") 区域中组态传送区。“IO 控制器中的地址”(Address in IO controller) 列仍为空，这是因为未分配 IO 控制器。



   注：要将输入区改为输出区或者反之，则必须导航至相应传送区的区域。
   – 选择将在运行期间访问共享智能设备的 IO 控制器的数量（至少为 2）（“操作模式 > 实时设置”(“Operating Mode > Real time settings”) 区域，“共享设备”(Shared Device) 区域）。



6. 保存项目。
7. 单击“导出”(Export) 按钮（“模式”>“智能设备组态”("Mode" > "I-device configuration") 区域，“导出通用站描述文件 (GSD)”(Export general station description file (GSD)） 部分）。如果您在“Export”(导出) 对话框中不更改名称，则 GSD 文件使用指定格式的名称（例如，“GSDML-V2.31-#Siemens-PreConf_S7-1200-I-Device-20130925-123456”）。

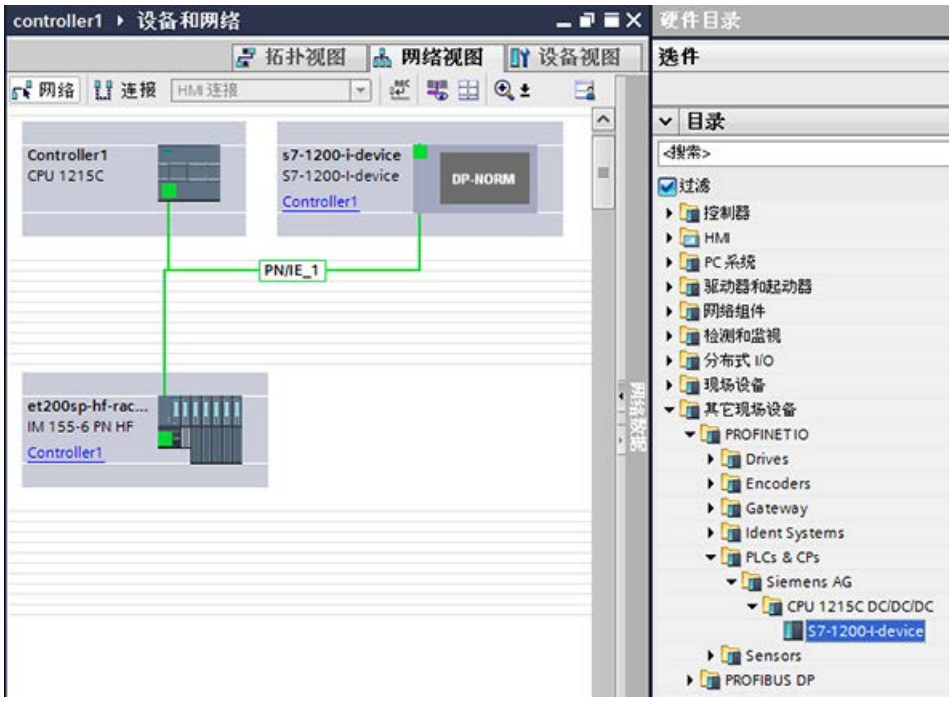
操作步骤：创建 Controller1 项目

要使用共享智能设备创建第一个项目，请按以下步骤操作：

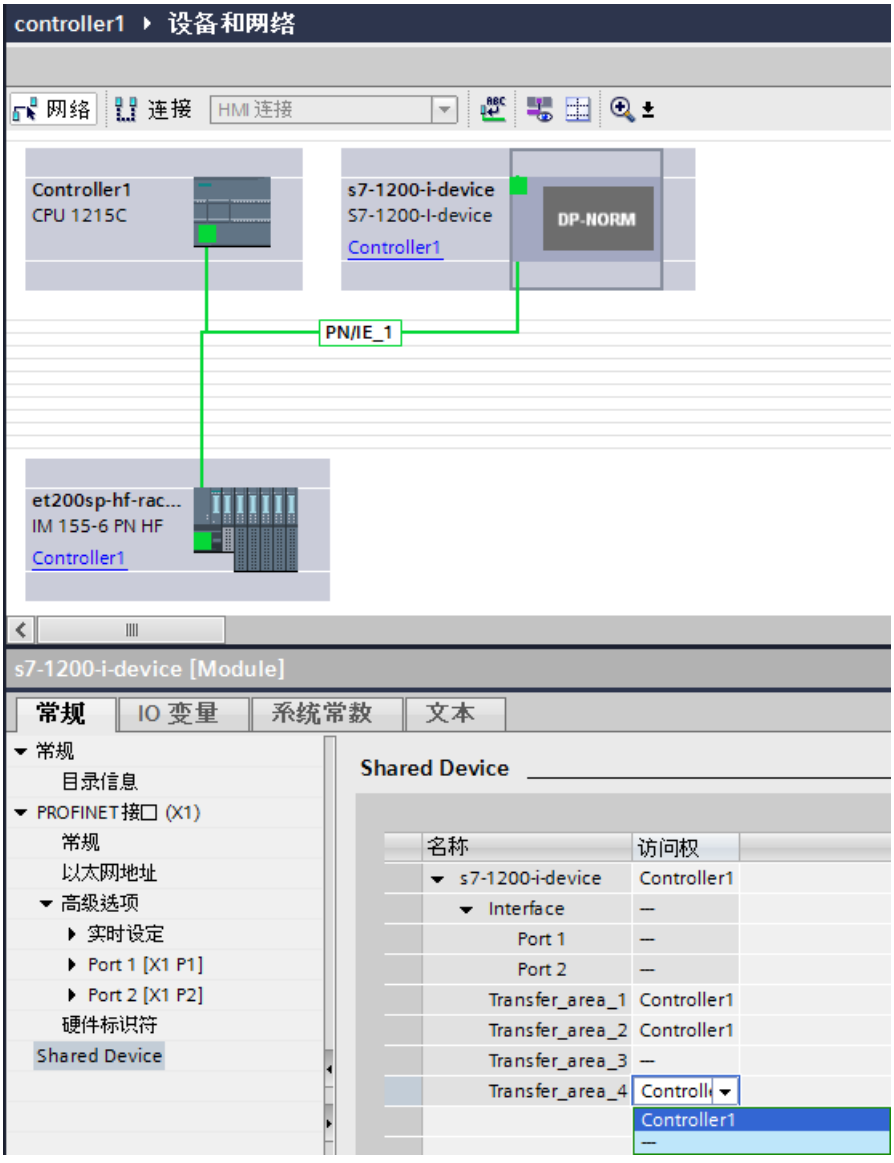
1. 启动 STEP 7。
2. 通过导出智能设备 CPU (S7-1200-I-Device) 安装 PROFINET GSD 文件。

3. 创建名为“Controller1”的新项目。
4. 在网络视图中插入 CPU 1215C。该 CPU 的名称应为“Controller1”。
5. 从硬件目录插入智能设备（硬件目录：其它现场设备 > PROFINET IO > PLC 与 CP (Other field devices > PROFINET IO > PLCs & CPs)）。

6. 将 IO 控制器“Controller1”分配给智能设备。



7. 在智能设备的属性中选择“共享设备”(Shared device) 区域：
   – 在表中，所有传送区和 PROFINET 接口都分配给本地 IO 控制器 (Controller1)。
   – 定义 Controller1 CPU CPU 不应访问的传送区。为这些区域选择“---”条目。这些传送区是为 Controller2 提供的。



8. 用户可以通过设备总览中 IO 控制器的设备视图来调整地址。要打开设备总览，请双击智能设备。

9. 保存项目。

步骤 – 创建 Controller2 项目

要使用共享设备创建第二个项目，请按以下步骤操作：

1. 再次启动 STEP 7。
   将打开 STEP 7 的一个新实例。
2. 在新实例中，创建一个名为“Controller2”的新项目。
3. 在网络视图中插入 CPU 1215C。分配名称“Controller2”。
4. 从硬件目录插入智能设备（硬件目录：其它现场设备 > PROFINET IO > PLC 与 CP (Other field devices > PROFINET IO > PLCs & CPs)）。
5. 将 IO 控制器“Controller2”分配给智能设备。
6. 就如同在 Controller1 项目中一样，调整对传送区的访问权。确保没有重复的分配结果。
7. 调整子网和 PROFINET 接口的参数。因为共享智能设备涉及到不同项目中的相同设备，所以这些数据必须匹配。
8. 保存项目。

两个项目现在有同样组态的共享智能设备。在下一步中，仍然应在不同的项目中检查 IO 控制器访问权和 PROFINET 接口的参数。

总结 - 为访问共享设备分配参数

传送区可以自动分配给本地 IO 控制器。要更改分配情况，请按以下步骤操作：

1. 单击“Controller1”项目的网络视图中的“S7-1200-I-Device”设备，并选择 “共享设备”(Shared device) 区域。
2. 将会出现一张表，显示有权访问每个已组态传送区的 CPU。默认设置是本地 CPU 有权访问所有模块和子模块。
3. 保持仍保留在本地 CPU 地址范围内的所有传送区的设置“Controller1”。
   从“Controller2”项目中，为将位于“Controller2”CPU 地址范围内的所有传送区选择设置“---”。
   这意味着项目外的 IO 控制器将有权访问传送区。
4. 对于其余项目，使用相同的步骤。

5. 最后，检查两个项目中每个模块或子模块的访问权设置是否“完整”。这意味着如果本地 CPU 在一个项目中有访问权，那么必须在另一个项目中设置选项“---”，反之亦然。
   注：PROFINET 接口和端口的选项“---”使相关的参数为只读，无法更改。PROFINET 接口的参数和端口参数只能在其 PROFINET 接口分配给本地 CPU 的项目中编辑。无论怎样，两个项目中的端口都可以互连。
6. 检查是否为所有项目中的共享设备设置了相同的 IP 地址参数和设备名称。
   检查是否在所有项目中为连接共享设备的子网设置了相同的 S7 子网 ID（子网属性，巡视窗口中的“常规”(General) 区域）。

说明

如果对智能设备进行更改（例如，更改传输区域的数量或长度），请再次以 GSD 文件的形式导出该智能设备。在每个使用智能设备作为共享设备的项目中重新安装 GSD 文件。确保仅一个 IO 控制器有权访问传送区。

说明

当使用 S7-1200 作为共享智能设备和控制器，确保增加 PROFINET 智能设备和 PROFINET IO 更新时间以便清除通信性能的影响。当选择 2 ms 作为单一 PROFINET 智能设备时间的更新时间并选择 2 ms 作为单一 PROFINET IO 时间的更新时间，系统会非常稳定且运行良好。

在 PROFINET 智能设备或 IO 的“属性组态”(Properties configuration) 对话框中指定“IO 周期”参数。更多相关信息，请参见“组态 IO 循环时间” (页 944)。

操作步骤 - 调整实时设置

为确保所有 IO 控制器和共享设备使用适当的发送时钟运行，并确保根据通信负载正确计算更新时间，必须调整并检查以下设置：

1. 必须为每个有权访问共享设备的模块或子模块的 IO 控制器设置相同的发送时钟：

- 如果使用 STEP 7 (TIA Portal) 组态 IO 控制器，执行以下步骤：
  - 打开相应的项目。
  - 选择 IO 控制器的 PROFINET 接口。
  - 在巡视窗口中选择“高级选项 > 实时设置 > IO 通信”(Advanced options > Real-time settings > IO communication) 区域，并设置共享的发送时钟。
- 如果使用其它工程组态工具组态 IO 控制器，执行以下步骤：
  - 在 STEP 7 (TIA Portal) 中选择共享设备的 PROFINET 接口，并在共享设备上读出发送时钟（“高级选项 > 实时设置”(Advanced options > Real-time settings) 区域）
  - 在工程组态工具中输入读取发送时钟。

说明

如果在 STEP 7 (TIA Portal) 中组态有权访问共享智能设备的**所有** IO 控制器，则可以在 IO 控制器上设置比共享设备支持的发送时钟更短的发送时钟（发送时钟调整）。

编译和下载

必须编译不同 IO 控制器的组态，并将其一个接一个地下载到 CPU。

由于对单独项目进行分布式组态，在访问参数分配错误时，STEP 7 不输出一致性错误。以下是错误分配访问参数的示例：

- 多个 IO 控制器可以访问同一个模块。
- IP 地址参数或发送时钟不同。

控制器操作之前不会显示这些错误，且这些错误将输出为组态错误。

11.2.15 介质冗余协议 (MRP)

以下 V4.2.x 及更高版本的 S7-1200 CPU 支持作为 MRP 客户端的操作，但不满足 MRP 管理器角色要求：

- CPU 1215C
- CPU 1217C
- CPU 1215FC

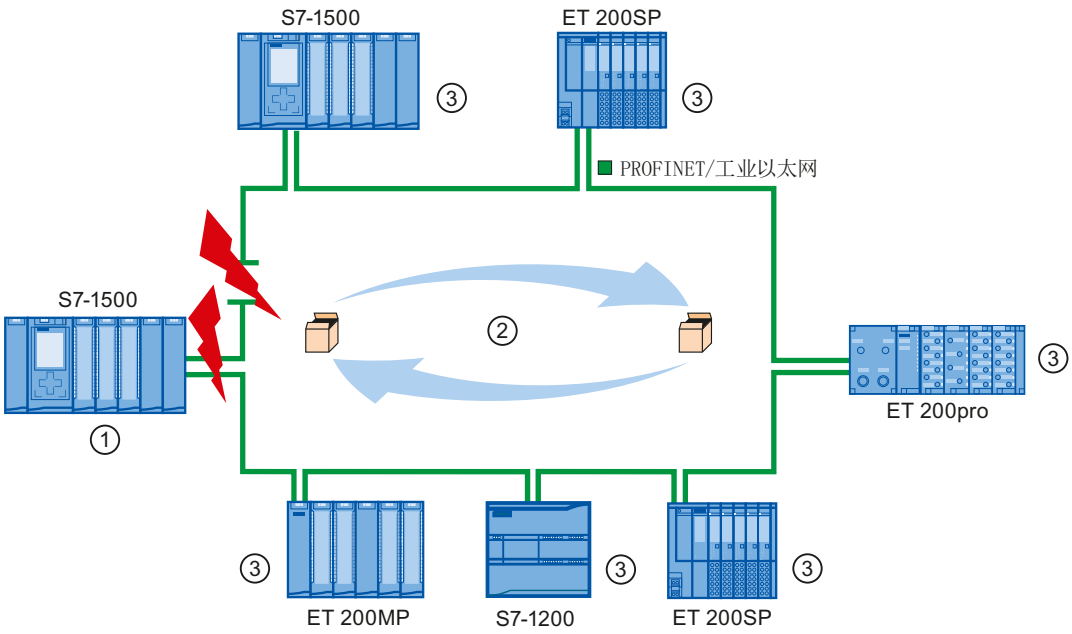
三个 S7-1200 CPU 都具备两个 PN 端口，为用于初始化 MRP 客户端操作的 MRP 协议和组态参数提供支持时需要这两个端口。

11.2.15.1 环形拓扑的介质冗余

为了提高具有光纤或电气线形总线型拓扑结构的工业以太网的网络可用性，可以通过将终端设备连接在一起，将线性总线形拓扑转换为环形拓扑。

环形拓扑结构中的设备可以是 IO 设备、IO 控制器、外部交换机和/或通信模块的集成交换机。

若要建立具有介质冗余性的环形拓扑结构，需要在一个设备中将线形总线型拓扑结构的两个自由端接在一起。将线形总线型拓扑结构闭合以形成一个环型网络可通过环网中某个设备的两个端口（环网端口）来完成。生成的环网中的一个设备将承担 MRP 管理器的角色。环网中的所有其它设备均为 MRP 客户端。



① MRP 管理器

② 测试帧

③ MRP 客户端

设备的环形端口用于建立与环形拓扑结构中两个相邻设备的连接。可在相关设备的组态中来选择和设置环网端口（如果可能，也可以预设）。

在环形拓扑中如何实现介质冗余

如果环形中任何一点断开，将自动重新组态各设备之间的数据路径。重新组态之后，设备可以再次使用。

在 MRP 管理器中，两个环网端口之一将被阻止为正常通信而进行的不间断网络运行，这样就不会将数据帧循环。对于数据传输而言，该环型拓扑就是一种线形总线型拓扑。MRP 管理器监视环网中有无中断。为此，它会从环网端口 1 和环网端口 2 发送测试帧。测试帧在环网的两个方向上传输，直到到达 MRP 管理器的另一个环网端口。

两个设备之间的连接断开或环网中的某个设备发生故障，都会引起环网中断。

如果 MRP 管理器的测试帧在环网中断期间不再能到达另一个环网端口，MRP 管理器就会连接它的两个环网端口。这个替代路径以线形总线型拓扑结构的形式再次恢复所有其余设备之间的正常连接。

从环网中断到恢复正常运行的线形总线型拓扑结构的时间称为重新组态时间。

一旦消除了中断，就会再次建立原来的传输路径，MRP 管理器的两个环网端口断开，MRP 客户端得到该变化的通知。随后，MRP 客户端将再次使用通向其它设备的原始路径。

介质冗余方法

SIMATIC 中的标准介质冗余方法是 (MRP)，其典型重新组态时间为 200 ms。每个环网最多可有 50 个设备。

11.2.15.2 使用介质冗余协议 (MRP)

“MRP”进程符合 IEC 61158 类型 10“PROFINET”中指定的介质冗余协议 (MRP)。

要求

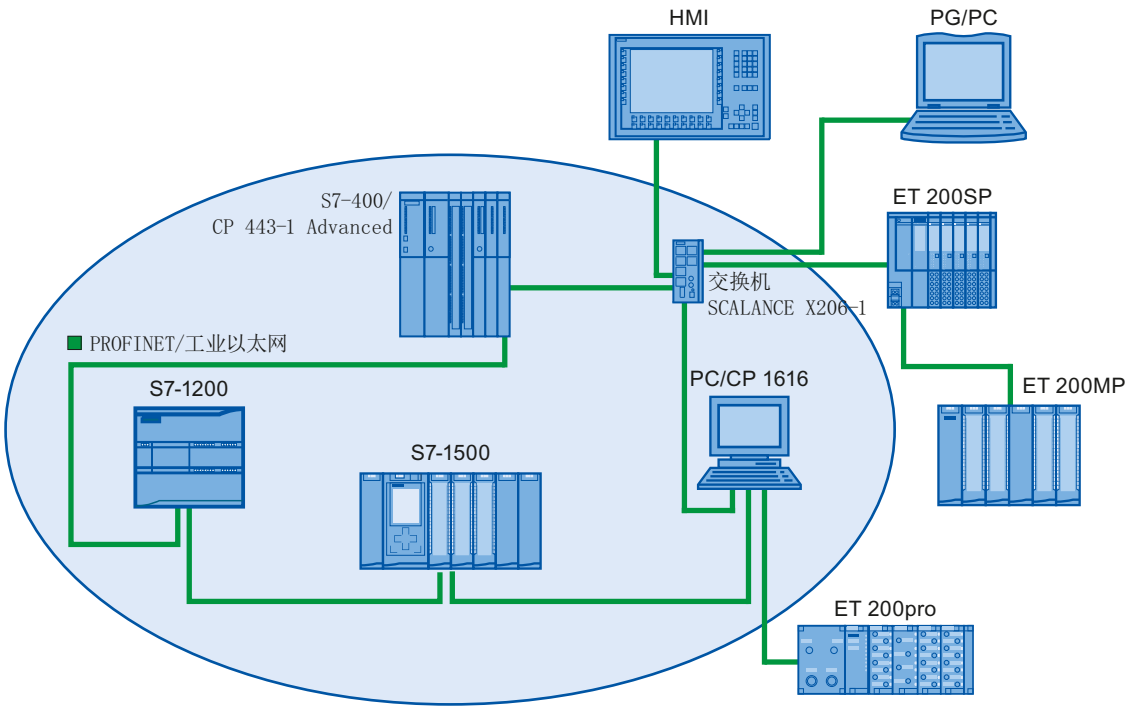
如果要使用 MRP 实现无故障运行，必须满足以下要求：

- 需要使用 MRP 的环网只能包括支持此功能的设备。
- 必须为环网中的所有设备启用“MRP”。
- 所有设备必须使用其环网端口进行互连。
- 必须有至少一个 MRP 管理器（角色“管理器（自动）”）可用。
- 环网必须包含不超过 50 台设备。否则，重新组态时间将可能超过或等于 200 ms。
- 环网中的所有伙伴端口必须具有相同的设置。

拓扑

下面的示意图显示了应用 MRP 的环网中可能的设备拓扑结构。阴影显示的椭圆内的设备位于冗余域中。

下图为应用 MRP 的环网拓扑结构示例：



以下规则适用于使用 MRP 的带介质冗余的环网拓扑结构：

- 环网中的所有设备属于同一冗余域。
- 环网中的一个设备将承担 MRP 管理器角色。
- 环网中的所有其它设备均为 MRP 客户端。

您可以通过未组态为环网端口的端口，将不符合 MRP 的设备连接到网络只能对具备多于两个端口的设备执行此操作（例如，SCALANCE X 交换机或具备 CP1616 的 PC）。

限制条件

您可以进行以下类型的通信：

- 可使用 MRP 来实现 MRP 和 RT: RT 操作。

  说明

  如果环网的重新组态时间大于 IO 设备的选定看门狗时间，则 RT 通信中断（站故障）。IO 设备所选的看门狗时间必须大于 200 ms。有关详细信息，参见“看门狗时间”部分。

- MRP 和 TCP/IP（TSEND、HTTP...）：可实现使用 MRP 的 TCP/IP 通信，这是因为可重新发送丢失的数据包（如果合适）。

- MRP 和有限启动：
  - 如果在环网中组态 MRP，则无法在相关设备上的 PROFINET 应用程序中使用“优先启动”(prioritized startup) 功能。
  - 如果要使用“优先启动”(prioritized startup) 功能，则必须在组态中禁用 MRP（该设备不能是环网的一部分）。
- 具有两个以上端口的 PROFINET 设备上的 MRP：如果在环网中运行了一个具有两个以上端口的 PROFINET 设备，则必须对不在环网中的端口设置同步边界。通过设置同步边界，可定义同步域的边界。您无法转发传输用来同步同步域内设备的同步帧。

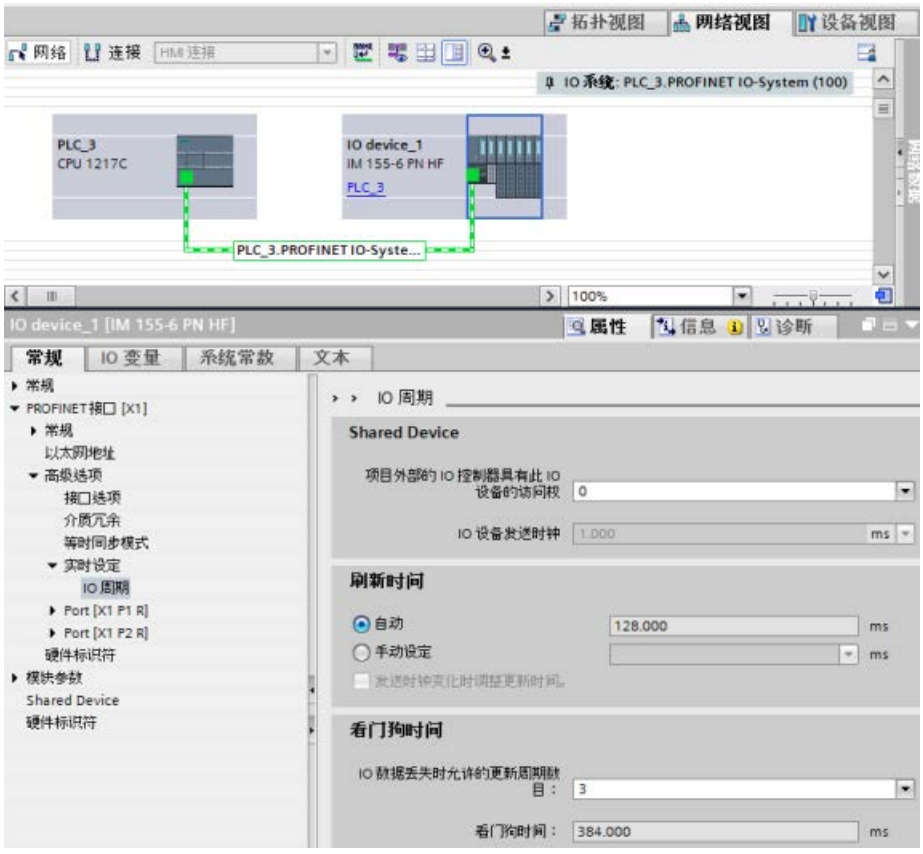
看门狗时间

看门狗时间是 IO 控制器或 IO 设备允许的且不含接收 IO 数据时间的时间间隔。如果在看门狗时间内 IO 控制器没有为 IO 设备提供数据，IO 设备将检测丢失的帧并输出替换值。这种情况将作为站故障在 IO 控制器内进行报告。

您可以为 PROFINET IO 设备组态看门狗时间。不要直接输入看门狗时间，而是通过“当 IO 数据丢失时可接受的更新周期次数”(Accepted number of update cycles when IO data is missing) 进行设置。最终的看门狗时间由更新周期次数自动计算得来。

要分配看门狗时间，请按以下步骤操作：

1. 在“网络”(Network) 视图或“设备”(Device) 视图中选择 IO 设备的 PROFINET 接口。
2. 在接口属性中，导航到：“高级选项 > 实时设置 > IO 周期”(Advanced options > Realtime settings > IO cycle)。
3. 从下拉列表中选择所需周期数。



11.2.15.3 组态介质冗余

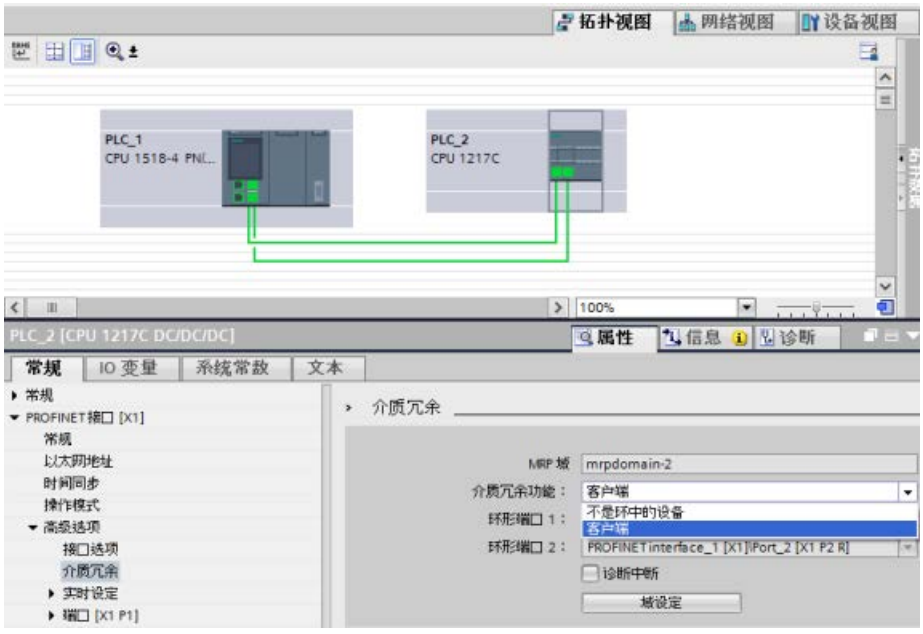
应用程序中的所有部件必须支持介质冗余协议 (MRP)。

操作步骤

要组态介质冗余，请按以下步骤操作：

1. 通过相应端口互连建立一个环网（如，在拓扑视图中）。
2. 选择一个要为其组态介质冗余的 PROFINET 设备。

3. 在巡视窗口中，浏览到“PROFINET 接口 [X1] > 高级选项 > 介质冗余”(PROFINET interface [X1] > Advanced options > Media redundancy)。



4. 在“介质冗余角色”(Media redundancy role) 下，为设备分配“管理器（自动）”(Manager (Auto))、“客户端”(Client) 或“环网中无设备”(Not device in the ring) 角色。
当在“TIA Portal 拓扑”(TIA Portal Topology) 视图中组态环网时，TIA Portal 将自动为您设置介质冗余角色。如果设备是管理器，TIA Portal 设置介质冗余角色为“管理器（自动）”(Manager (Auto))。对于 S7-1200，介质冗余角色被自动设置为“客户端”(Client)。

说明

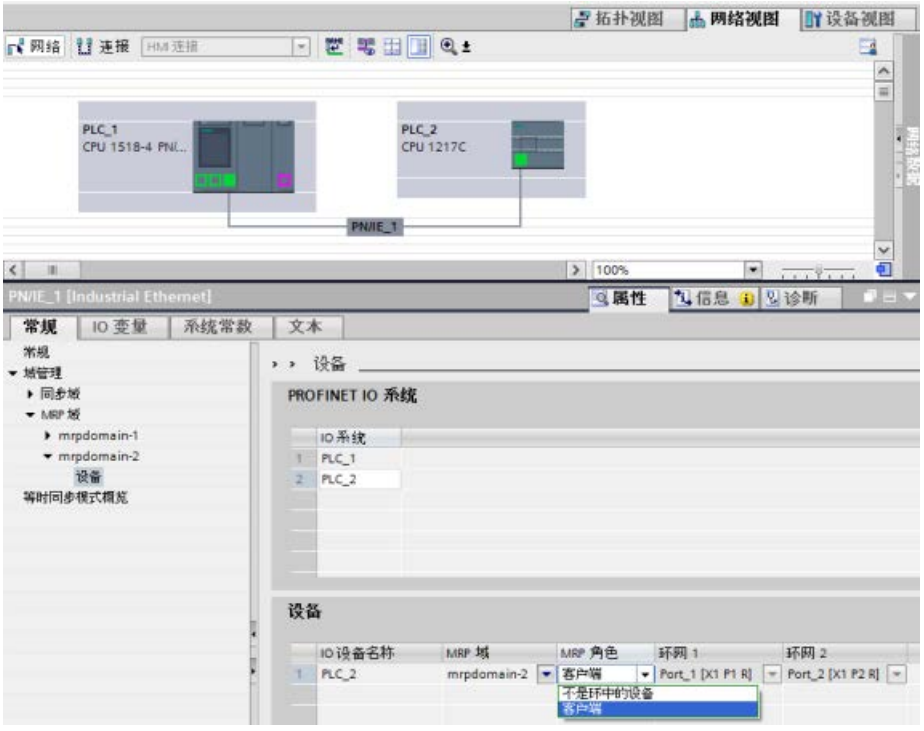
您不能将“管理器（自动）”(Manager (Auto)) 介质冗余角色分配给 S7-1200 CPU。

5. 针对环网中的所有 PROFINET 设备，重复步骤 2 到 4。

或者：

1. 在网络视图中突出显示 PROFINET IO 系统。
2. 单击 PROFINET IO 系统。

3. 在巡视窗口中，浏览至所需的 MRP 域的设备。



4. 对于 PROFINET 设备，设置“管理器（自动）”(Manager (Auto))、“客户端”(Client) 或“环网中无设备”(Not device in the ring) 角色。

说明

您不能将“管理器（自动）”(Manager (Auto)) MRP 角色分配给 S7-1200 CPU。

“介质冗余”(Media redundancy) 设置选项 MRP 角色

根据使用的设备，可使用“管理器”(Manager)、“管理器（自动）”(Manager (Auto))、“客户端”(Client) 和“环网中无设备”(Not device in the ring) 角色。

规则：

- 环网上只能有一个“管理器”(Manager) 角色的设备。不再允许有其它设备具有“管理器”(Manager) 或“管理器（自动）”(Manager (Auto)) 角色。环网中的所有其它设备只能具有“客户端”(Client) 角色。不在环网中的设备可以具备“客户端”(Client) 角色。
- 如果环网中没有设备具有“管理器”(Manager) 角色，则环网中必须至少有一个设备具有“管理器（自动）”(Manager (Auto)) 角色。环网中可以有任何数量的设备具有“客户端”(Client) 和“管理器（自动）”(Manager (Auto)) 角色。

说明

您不能将“管理器”(Manager) 或“管理器（自动）”(Manager (Auto)) MRP 角色分配给 S7-1200 CPU。

“介质冗余”(Media redundancy) 设置选项环网端口 1 和环网端口 2

一次选择一个要组态为环网端口 1 或环网端口 2 的端口。下拉列表框显示每种设备类型的可能端口选择。如果在出厂前设置了端口，此域将不可用。

说明

由于 S7-1200 CPU 只有两个端口，因此 S7-1200 中无需组态环网端口。

诊断中断

如果 MRP 状态的诊断中断将在本地 CPU 中输出，请选中“诊断中断”(Diagnostic interrupts) 复选框。可组态以下诊断中断：

- 布线或端口错误：
  环网端口出现如下错误时，CPU 会生成诊断中断：
  - 相邻的环网端口不支持 MRP。
  - 环网端口连接到非环网端口。
  - 环网端口连接到其它 MRP 域的环网端口。
- 中断/恢复（仅 MRP 管理器）
  如果环网中断后再恢复原始组态，CPU 生成诊断中断。如果在 0.2 秒内发生了这两种中断，则表明环网中断。

可通过对诊断错误中断 OB (OB 82) 中的适当响应进行编程，以响应用户程序中的这些事件。

说明

第三方设备作为 MRP 管理器

为确保在第三方设备用作环网中的 MRP 管理器时无错运行，必须在闭合环网前对环网中的所有其它设备分配固定角色“客户端”(Client)。否则，将产生循环数据帧和发生网络故障。

11.2.16 S7 路由

在 STEP 7 网络视图中，您可以通过连接不同 S7 子网中的设备创建复杂通信拓扑结构。可以连接经典 S7-300/S7-400 CPU 和 CP 以及最新 S7 CPU 和 CP，还可以纳入 HMI 和 PC 站，如 OPC 服务器。

决定哪些设备必须使用 STEP 7 进行通信和建立必要连接后，工程组态系统 (ES) 可以下载对应的路由表到不同 S7 路由器中作为硬件组态的一部分。下载路由表到不同 S7 路由器后，即使设备位于不同 S7 子网上，ES 或其它通信伙伴也可以和每个设备进行通信。之所有能够实现此类通信是因为 CPU 和/或 CP 在中间作为 S7 路由器使用。CPU 和/或 CP 转发收到的连接请求给下一个 S7 路由器直到连接请求到达目标设备且设备建立好了 S7 连接。

CPU 通过写入记录机制传送本地基站中 CP 设备所需的路由表。发出连接请求时，路由表将建立从一个设备到另一个设备的路由，其中包括远程 S7 Subnet_ID。收到连接请求的设备会询问路由表，查找到目标 S7 子网路径上的下一个站点，并转发连接请求。最终，连接请求到达目标设备，响应通过连接路径反向传送。

S7-1200 CPU 具有单个 PN 接口和多达三个连接本地通信总线的 CP。因此，当在 S7-1200 站内进行路由时您有两个选项：

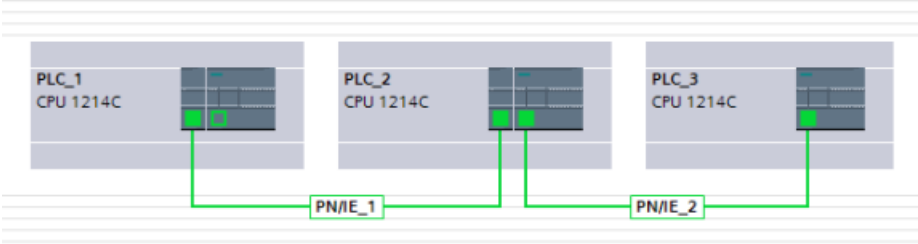
- 在 CPU 和 CP 之间路由。
- 从一个 CP 路由到另一个 CP。

有关所有支持 S7 路由功能的 S7-1200 CP 的详细信息，请参见西门子工业在线支持的 S7-1200 CP 产品支持部分。CP 1243-1 (https://support.industry.siemens.com/cs/cn/zh/view/584459) 为支持 S7 路由能力的 CP 模块示例。

11.2.16.1 CPU 和 CP 接口之间的 S7 路由

由于 S7-1200 CPU 限定为单个 PN 接口，独立 CPU 无法作为路由器使用。永远无法将独立 CPU 同时连接多个 S7 子网。当在 CPU 的本地基站安装 CP 模块时，可以连接到多 S7 子网并使用路由。

在以下示例系统中，为了使 PLC_1 和 PLC_3 进行通信，工程组态系统 (ES) 必须通过 PLC_2 路由消息。ES 必须为 PLC_2 下载路由表，PLC_2 必须为其本地基站中的 CP 模块提供路由表。路由表就位后，即使未直接连接，PLC_1 仍可以和 PLC_3 进行通信。

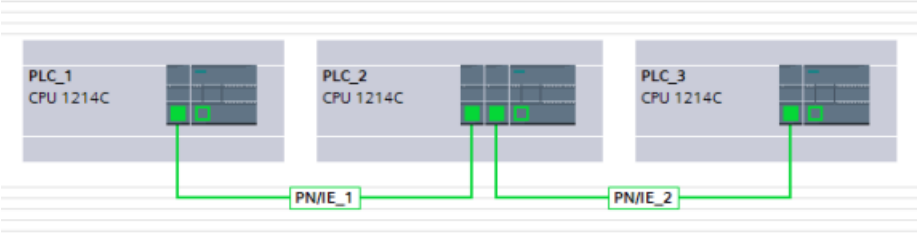


为了检查从任意 S7 子网到其他 S7 子网的路由，PLC_1 必须建立和 PLC_3 之间的传输连接，PLC_3 必须建立到 PLC_1 的连接。这样做使得从 PLC PN/IE 接口到一个 CP 模块以及从 CP 模块到 PLC 的 PN/IE 接口的路由成为可能。

11.2.16.2 两个 CP 接口之间的 S7 路由

由于 S7-1200 CPU 支持多达三个 CP 模块，因此您可以将所有三个模块连接到不同的 S7 子网上。当在 CPU 的本地基站安装至少两个 CP 模块并连接到不同 S7 子网时，您可以使用路由功能。

在以下示例系统中，为了使 PLC_1 和 PLC_3 进行通信，工程组态系统 (ES) 必须通过 PLC_2 将消息从本地基站上一个 CP 模块路由到另一个 CP 模块。ES 必须为 PLC_2 下载路由表，PLC_2 必须为两个 CP 模块提供路由表。路由表就位后，即使未直接连接，PLC_1 仍可以和 PLC_3 进行通信。同时应该注意到在两个 CP 模块之间进行路由时，将不通过 PLC_2 的 PN/IE 接口发送消息。



11.2.17 禁用 SNMP

简单网络管理协议 (SNMP) 是用来收集和组织关于 IP 网络上受管设备信息，以及修改该信息从而更改设备行为的互联网标准协议。支持 SNMP 的设备通常包括路由器、交换机、服务器、工作站、打印机、数据机柜等。

SNMP 协议广泛应用于网络管理系统中，监视连接网络的设备，从而保证对设备状况的关注。SNMP 采用多种服务和工具对网络拓扑机构进行检测和诊断。有关具有 SNMP 功能设备的属性信息保存在管理信息库 (MIB) 文件中，用户需要具有相应权限才能访问。SNMP 在受管系统上以变量形式显示描述系统组态的管理数据。可以通过管理应用程序对这些变量进行查询（或有时可进行设置）。

SNMP 使用 UDP 传输协议且具有两个网络组件：

- SNMP 管理器：监视网络节点
- SNMP 客户端：收集各个网络节点中各种网络指定信息，并以结构化的形式存储在管理信息库 (MIB) 中。基于这些收集到的数据，可对网络进行详细诊断。

部分条件下，应用程序可能需要您禁用 SNMP。这些条件包括以下：

- 网络中的安全设置不允许使用 SNMP。
- 使用自己的 SNMP 解决方案（例如通过自己的通信指令）。

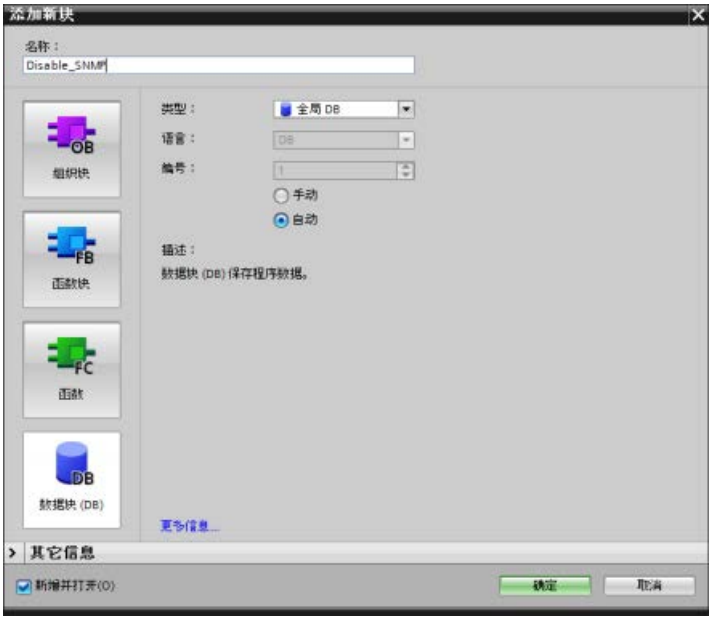
如果禁用了设备 SNMP 功能，则不再具备部分选项以对网络拓扑结构进行诊断（如，PRONETA 工具或 CPU 中的 Web 服务器）。

11.2.17.1 禁用 SNMP

禁用 SNMP

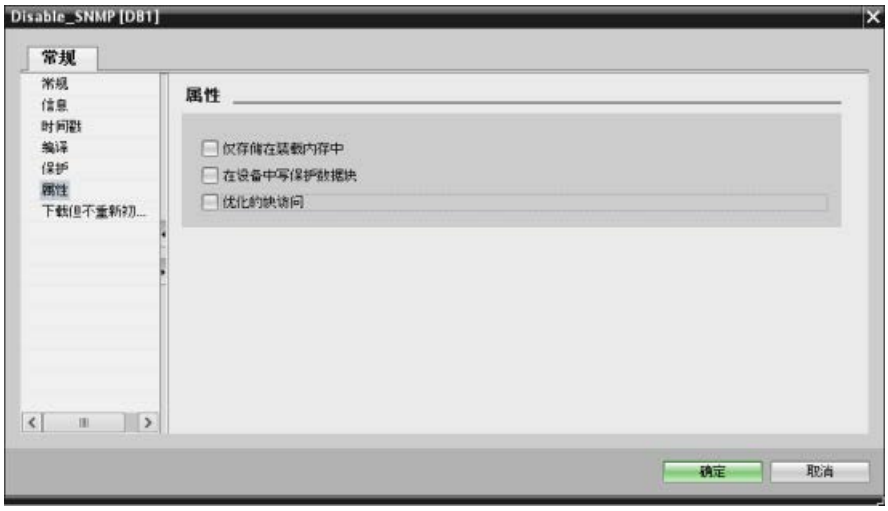
请按如下步骤操作，在 S7-1200 CPU 中禁用 SNMP：

1. 创建典型数据块 (DB)：



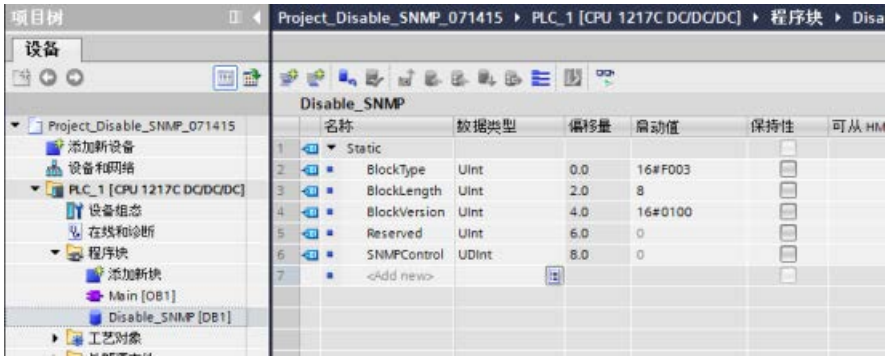
2. 选择新建 DB 的属性。

3. 选择“属性”(Attributes) 选项卡。取消选中“优化块访问”(Optimized block access) 复选框：

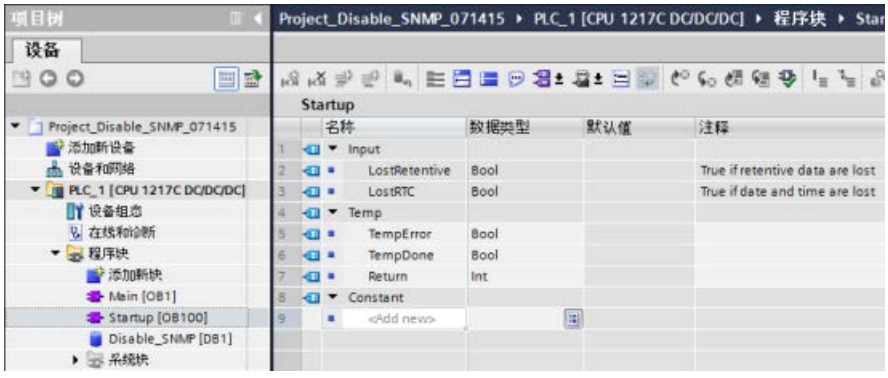


4. 单击“确定”(OK) 按钮。
将显示一则消息，建议您重新编译程序。此时重新编译程序。

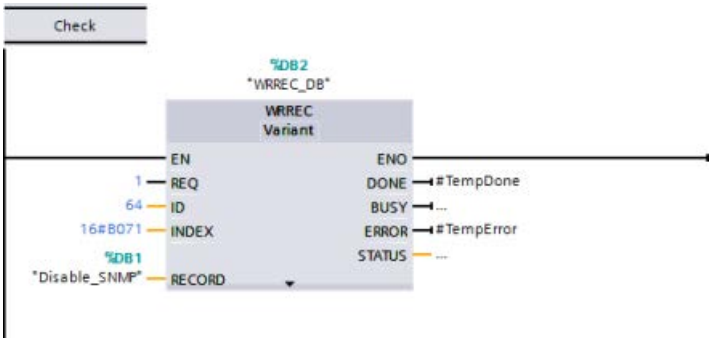
5. 在典型 DB 块接口，使用所示值创建下列静态变量。将在程序中使用这些变量禁用内部 SNMP 实现：



6. 在启动 OB (OB100) 中，添加如下所示临时变量：



7. 使用 LAD 编辑器，在启动 OB (OB100) 的程序段 1 中，插入标签（跳转标签）（在下面的示例中，标签名为“Check”）和具有所示输入和输出的 WRRC （写入记录）指令：



8. 插入下列回路和具有跳转至标签 (JMP) 输出的 Check 代码。该代码确保完成调用，并在退出启动 OB 前禁用 SNMP：

11.2.18 诊断

有关利用组织块 (OB) 诊断这些通信网络的信息，请参见“组织块 (OB)” (页 88)。

11.2.19 分布式 I/O 指令

请参见“分布式 I/O（PROFINET、PROFIBUS 或 AS-i）” (页 385)，以了解有关如何将分布式 I/O 指令用于这些通信网络的信息。

11.2.20 诊断指令

请参见“诊断（PROFINET 或 PROFIBUS）”：“诊断指令” (页 439)，以了解有关如何将诊断指令用于这些通信网络的信息。

11.2.21 分布式 I/O 的诊断事件

请参见“诊断（PROFINET 或 PROFIBUS）”：“分布式 I/O 的诊断事件” (页 488)，以了解有关如何将该诊断信息用于这些通信网络的信息。

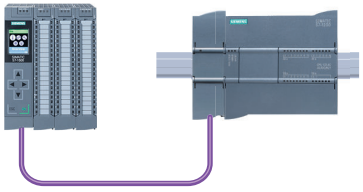
11.3 PROFIBUS

PROFIBUS 系统使用总线主站来轮询 RS485 串行总线上以多点方式分布的从站设备。PROFIBUS 从站可以是任何处理信息并将其输出发送到主站的外围设备（I/O 传感器、阀门、电机驱动器或其它测量设备）。该从站构成网络上的被动站，因为它没有总线访问权限，只能确认接收到的消息或根据请求将响应消息发送给主站。所有 PROFIBUS 从站具有相同的优先级，并且所有网络通信都源于主站。

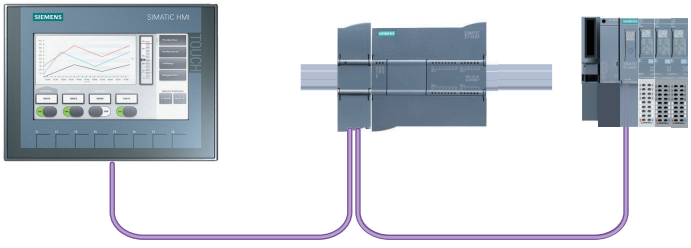
PROFIBUS 主站构成网络的“主动站”。PROFIBUS DP 定义两类主站。第 1 类主站（通常是中央可编程控制器 (PLC) 或运行特殊软件的 PC）处理与分配给它的从站之间的常规通信或数据交换。第 2 类主站（通常是组态设备，如用于调试、维护或诊断的膝上型计算机或编程控制台）是主要用于调试从站和诊断的特殊设备。

S7-1200 可通过 CM 1242-5 通信模块作为从站连接到 PROFIBUS 网络。CM 1242-5（DP 从站）模块可以是 DP V0/V1 主站的通信伙伴。如果想在第三方系统中组态模块，可使用适合 CM 1242-5（DP 从站）的 GSD 文件，模块随附的 CD 或 Internet 上 Siemens 自动化客户支持 (https://support.industry.siemens.com/cs/cn/zh/ps/6GK7242-5DX30-0XE0)页面中提供了该文件。

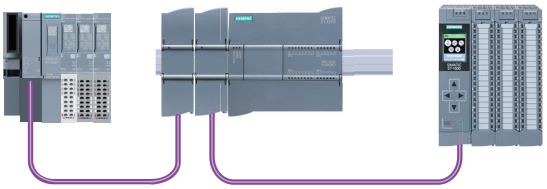
在下图中，S7-1200 是 S7-300 控制器的 DP 从站：



S7-1200 也可通过 CM 1243-5 通信模块作为主站连接到 PROFIBUS 网络。CM 1243-5（DP 主站）模块可以是 DP V0/V1 从站的通信伙伴。在下图中，S7-1200 是控制 ET 200SP DP 从站的主站：



如果同时安装了 CM 1242-5 和 CM 1243-5，则 S7-1200 既可充当上位 DP 主站系统的从站，又可充当下位 DP 从站系统的主站：



对于 V3.0 及以上版本，最多可以在一个工作站上组态三个 PROFIBUS CM，可以是 DP 主 CM 或 DP 从属 CM 的任意组合。采用 V3.0 或更高版本的 CPU 固件时，每个 DP 主站最多可控制 32 个从站。

PROFIBUS CM 的组态数据存储在本地 CPU 中。这样就可以在必要时方便地替换这些通信模块。

要将 PROFIBUS 与 S7-1200 V4.0 或更高版本的 CPU 配合使用，必须至少将 PROFIBUS 主站 CM 固件升级至 V1.3。

说明

始终将 PROFIBUS CM 固件更新为可用的最新版本 (http://support.automation.siemens.com/CN/view/zh/42131407)。可通过以下任一方法执行固件更新：

- 使用 STEP 7 的在线和诊断工具 (页 1390)
- 使用 SIMATIC 存储卡 (页 143)
- 使用 Web 服务器“模块信息”标准 Web 页面 (页 1045)
- 使用 SIMATIC 自动化工具 (https://support.industry.siemens.com/cs/cn/zh/view/98161300/en)

11.3.1 PROFIBUS CM 的通信服务

PROFIBUS CM 使用 PROFIBUS DP-V1 协议。

DP-V1 支持的通信类型

可通过 DP-V1 实现以下类型的通信：

- 周期性通信（CM 1242-5 和 CM 1243-5）
  两个 PROFIBUS 模块支持周期性通信，因而可在 DP 从站和 DP 主站之间传送过程数据。周期性通信由 CPU 的操作系统进行处理。 此时不需要软件块。 直接在 CPU 的过程映像中读取或写入 I/O 数据。
- 非周期性通信（仅限 CM 1243-5）
  DP 主站模块还支持使用软件块进行非周期性通信：
  – “RALRM”指令可用于处理中断。
  – “RDREC”和“WRREC”指令可用于传送组态和诊断数据。

CM 1243-5 不支持的功能： SYNC/FREEZE 和 Get_Master_Diag

CM 1243-5 的其它通信服务

CM 1243-5 DP 主站模块另外还支持以下通信服务：

- S7 通信
  – PUT/GET 服务
    DP 主站起客户机和服务器的作用，可通过 PROFIBUS 对其它 S7 控制器或 PC 进行查询。
  – PG/OP 通信
    通过 PG 功能，可以从 PG 下载组态数据和用户程序，以及将诊断数据传送到 PG。进行 OP 通信时，可用的通信伙伴有 HMI 面板、装有 WinCC flexible 的 SIMATIC 面板 PC 或者支持 S7 通信的 SCADA 系统。

11.3.2 PROFIBUS CM 用户手册参考资料

更多信息

有关 PROFIBUS CM 的详细信息，请参见设备手册。 您可以在 Internet 的 Siemens 工业自动化客户支持页面上找到这些手册，相应的条目 ID 如下：

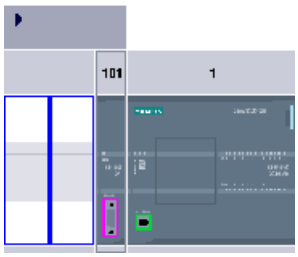
- CM 1242-5 (https://support.industry.siemens.com/cs/cn/zh/ps/15667)
- CM 1243-5 (https://support.industry.siemens.com/cs/cn/zh/ps/15669)

11.3.3 配置 DP 主站和从站设备

11.3.3.1 添加 CM 1243-5（DP 主站）模块和 DP 从站

在“设备和网络”(Devices and networks) 门户中，使用硬件目录向 CPU 添加 PROFIBUS 模块。这些模块连接在 CPU 左侧。要将模块插入到硬件组态中，可在硬件目录中选择模块，然后双击该模块或将其拖到高亮显示的插槽中。

表格 11-60 将 PROFIBUS CM 1243-5（DP 主站）模块添加到设备组态

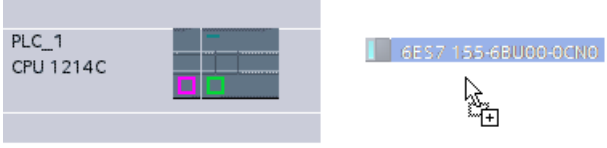
模块	选择模块	插入模块	结果
CM 1243-5（DP 主站）			

同样也使用硬件目录添加 DP 从站。例如，要添加 ET 200SP DP 从站，可在硬件目录中展开下列容器：

- 分布式 I/O
- ET 200SP
- 接口模块
- PROFIBUS

下一步，从零件号列表中选择“6ES7 155-6BU00-0CN0”(IM155-6 DP HF)，并按下图所示添加 ET 200SP DP 从站。

表格 11-61 将 ET 200SP DP 从站添加至设备组态

插入 DP 从站	结果
	

11.3.3.2 组态两台 PROFIBUS 设备之间的逻辑网络连接

组态 CM 1243-5（DP 主站）模块后，便可以组态网络连接。

在“设备和网络”(Devices and Networks) 门户中，使用“网络视图”(Network view) 创建项目中各设备之间的网络连接。 要创建 PROFIBUS 连接，请选择第一台设备上的紫色 (PROFIBUS) 框。 拖出一条线连接到第二台设备上的 PROFIBUS 框。 释放鼠标按钮，即可创建 PROFIBUS 连接。

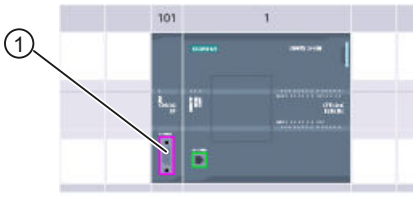
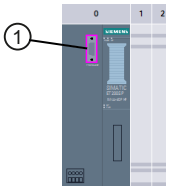
有关详细信息，请参见“设备配置：创建网络连接” (页 811)。

11.3.3.3 给 CM 1243-5 模块和 DP 从站分配 PROFIBUS 地址

组态 PROFIBUS 接口

组态两台 PROFIBUS 设备之间的逻辑网络连接后，便可以组态 PROFIBUS 接口的参数。为此，请单击 CM 1243-5 模块上的紫色 PROFIBUS 框，PROFIBUS 接口即显示在巡视窗口的“属性”(Properties) 选项卡中。以相同的方式组态 DP 从站 PROFIBUS 接口。

表格 11-62 组态 CM 1243-5（DP 主站）模块和 ET 200SP DP 从站 PROFIBUS 接口

CM 1243-5（DP 主站）模块	ET 200SP DP 从站
	

① PROFIBUS 端口

分配 PROFIBUS 地址

在 PROFIBUS 网络中，为每台设备分配了一个 PROFIBUS 地址。这个地址可以在 0 到 127 的范围内，但下列情况除外：

- 地址 0：为网络组态和/或连接到总线的编程工具保留
- 地址 1：Siemens 保留给第一个主站使用
- 地址 126：为不具有开关设置且必须通过网络重新寻址的出厂设备保留
- 地址 127：为给网络上所有设备广播消息保留，不可以分配给运转设备

因此，可用于 PROFIBUS 运转设备的地址的范围是 2 到 125。

在“属性”(Properties) 窗口中，选择“PROFIBUS 地址”(PROFIBUS address) 组态条目。STEP 7 将显示 PROFIBUS 地址组态对话框，该对话框用于分配设备的 PROFIBUS 地址。

表格 11-63 PROFIBUS 地址的参数

参数		说明
子网	连接到设备的子网的名称。单击“添加新子网”(Add new subnet) 按钮以创建新的子网。默认为“未连接”(Not connected)。可以有两种连接类型： ● 默认情况下“未连接”(Not connected) 提供本地连接。 ● 网络具有两个或多个设备时，需要子网。	
参数	地址	分配给设备的 PROFIBUS 地址
	最高地址	最高 PROFIBUS 地址基于 PROFIBUS 上的主动站（例如 DP 主站）。被动 DP 从站单独具有范围是 1 到 125 的 PROFIBUS 地址，即使最高 PROFIBUS 地址被设置为（例如）15。最高 PROFIBUS 地址与令牌传递有关（发送权限传递），并且令牌只传递给主动站。指定最高 PROFIBUS 地址可优化总线。
	传输率	组态的 PROFIBUS 网络的传输率：PROFIBUS 传输率的范围是 9.6 Kbps 到 12 Mbps。传输率设置取决于所使用的 PROFIBUS 节点的属性。传输率不应大于最慢节点所支持的传输率。 通常需要为 PROFIBUS 网络上的主站设置传输率，而所有 DP 从站都将自动使用该传输率（自动波特）。

11.3.4 分布式 I/O 指令

请参见“分布式 I/O（PROFINET、PROFIBUS 或 AS-i）” (页 385)，以了解有关如何将分布式 I/O 指令用于这些通信网络的信息。

11.3.5 诊断指令

请参见“诊断（PROFINET 或 PROFIBUS）”：“诊断指令” (页 439)，以了解有关如何将诊断指令用于这些通信网络的信息。

11.3.6 分布式的诊断事件

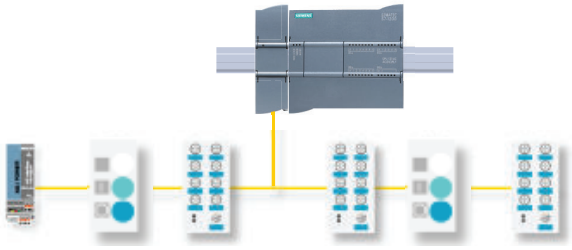
请参见“诊断（PROFINET 或 PROFIBUS）”：“分布式 I/O 的诊断事件”(页 488)，以了解有关如何将该诊断信息用于这些通信网络的信息。

11.4 AS-i

通过 S7-1200 AS-i 主站 CM 1243-2 可将 AS-i 网络连接到 S7-1200 CPU。

执行器/传感器接口（或者说 AS-i）是自动化系统中最低级别的单一主站网络连接系统。 CM 1243-2 作为网络中的 AS-i 主站。 仅需一条 AS-i 电缆，即可将传感器和执行器（AS-i 从站设备）经由 CM 1243-2 连接到 CPU。CM 1243-2 可处理所有 AS-i 网络协调事务，并通过为其分配的 I/O 地址中继传输从执行器和传感器到 CPU 的数据和状态信息。根据从站类型，可以访问二进制值或模拟值。 AS-i 从站是 AS-i 系统的输入和输出通道，并且只有在由 CM 1243-2 调用时才会激活。

在下图中，S7-1200 是控制 AS-i 数字量/模拟量 I/O 模块从站设备的 AS-i 主站。



要将 AS-i 与 S7-1200 V4.0 CPU 配合使用，必须将 AS-i 主站 CM 的固件升级为 V1.1。

可通过 Web 服务器或 SIMATIC 存储卡进行此升级操作。

说明

对于 V4.0 S7-1200 CPU，如果使用 Web 服务器或 SIMATIC 存储卡将 AS-i 固件从 V1.0 升级至 V1.1，则必须按照下列步骤在 AS-i Master CM 1243-2 中更新 AS-i 固件：

1. 将固件升级下载至 AS-i Master CM 1243-2 中。
2. 下载完成后，对 S7-1200 CPU 循环上电以在 AS-i Master CM 1243-2 中完成固件升级过程。
3. 对每个附加的 AS-i 主站 CM 1243-2 重复步骤 1 和步骤 2。S7-1200 PLC 最多支持三个 AS-i 主站 CM 1243-2。

说明

建议始终将 AS-i CM 固件更新至最新可用版本 (http://support.automation.siemens.com/CN/view/zh/43416171)，相应版本可从 Siemens 服务和支持网站获取。

11.4.1 组态 AS-i 主站和从站设备

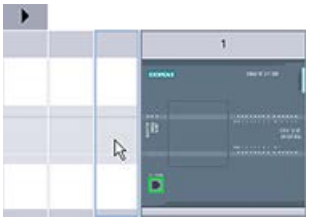
AS-i 主站 CM 1243-2 作为通信模块集成到 S7-1200 自动化系统中。

有关 AS-i 主站 CM 1243-2 的详细信息，请参见“SIMATIC S7-1200 的 AS-i 主站 CM 1243-2 和 AS-i 数据解耦装置 DCM 1271”手册 (https://support.industry.siemens.com/cs/cn/zh/ps/15750/man)。

11.4.1.1 添加 AS-i 主站 CM 1243-2 和 AS-i 从站

使用硬件目录将 AS-i 主站 CM1243-2 模块添加到 CPU。 这些模块连接到 CPU 的左侧，并且最多可使用三个 AS-i 主站 CM1243-2 模块。 要将模块插入到硬件组态中，可在硬件目录中选择模块，然后双击该模块或将其拖到高亮显示的插槽中。

表格 11-64 向设备组态添加 AS-i 主站 CM1243-2 模块

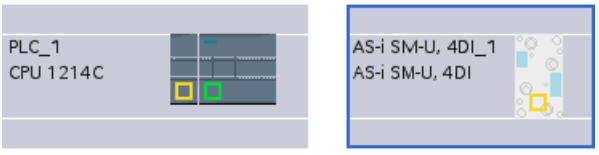
模块	选择模块	插入模块	结果
CM 1243-2 AS-i 主站			

同样也使用硬件目录添加 AS-i 从站。 例如，要添加“紧凑型数字量输入 I/O 模块”从站，请在硬件目录中展开下列容器：

- 现场设备
- AS-interface 接口从站

接下来，从零件号列表中选择“3RG9 001-0AA00”（AS-i SM-U，4DI），并按下图所示添加“紧凑型数字量输入 I/O 模块”从站。

表格 11-65 向设备组态添加 AS-i 从站

插入 AS-i 从站	结果
	

11.4.1.2 组态两个 AS-i 设备之间的逻辑网络连接

组态 AS-i 主站 CM1243-2 后，便可以组态网络连接。

在“设备和网络”(Devices and Networks) 门户中，使用“网络视图”(Network view) 创建项目中各设备之间的网络连接。 要创建 AS-i 连接，在第一个设备上选择黄色的 (AS-i) 框。 拖出一条线连接到第二个设备上的 AS-i 框。 松开鼠标按钮，即可创建 AS-i 连接。

更多相关信息，请参见“设备组态： 创建网络连接” (页 811)。

11.4.1.3 组态 AS-i 主站 CM1243-2 的属性

要组态 AS-i 接口的参数，请单击 AS-i 主站 CM1243-2 模块上的黄色 AS-i 框，巡视窗口的“属性”(Properties) 选项卡将显示该 AS-i 接口。

在 STEP 7 巡视窗口中，可以查看、组态以及更改常规信息、地址和操作参数：

表格 11-66 AS-i 主站 CM1243-2 模块属性

属性	说明
常规	AS-i 主站 CM 1243-2 的名称
操作参数	AS-i 主站的响应参数
I/O 地址	从站 I/O 地址的地址区域
AS-i 接口 (X1)	分配的 AS-i 网络

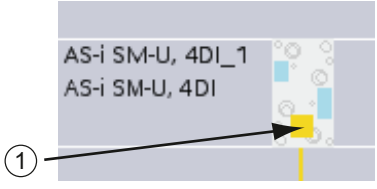
说明

“AS-i 组态故障的诊断中断”(Diagnostic interrupt for faults in the AS-i configuration) 和“自动地址编程”(Automatic address programming) 始终处于激活状态，因此呈灰显。

11.4.1.4 为 AS-i 从站分配 AS-i 地址

组态 AS-i 从站接口

要组态 AS-i 接口的参数，请单击 AS-i 从站上的黄色 AS-i 框，巡视窗口的 “属性”(Properties) 选项卡将显示该 AS-i 接口。



① AS-i 端口

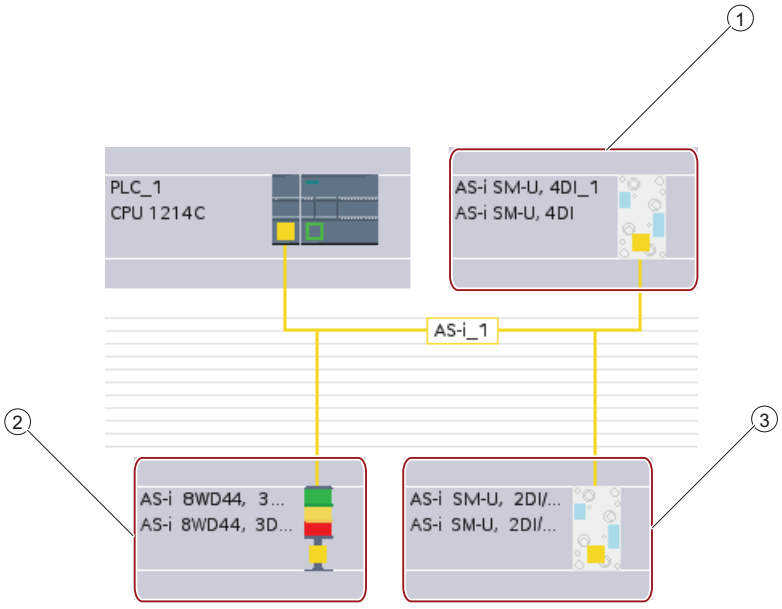
分配 AS-i 从站地址

在 AS-i 网络中，每台设备都分配有一个 AS-i 从站地址。 此地址的范围可从 0 到 31；但是，地址 0 只预留给新从站设备。 从站地址从 1（A 或 B）一直到 31（A 或 B），总计最多 62 台从站设备。

“标准”AS-i 设备使用完整地址，其数字地址不带 A 或 B 标识。 “A/B 节点”AS-i 设备的每个地址都有 A 或 B，这样 31 个地址全都可以使用两次。 地址空间范围为 1A 到 31A 再加 1B 到 31B。

1 - 31 范围内的任何地址都可分配给 AS-i 从站设备；即，无论是从站从地址 21 开始，还是为第一个从站分配地址 1，都无关紧要。

在下面的示例中，三个 AS-i 设备的地址分别为“1”（标准类型设备）、“2A”（A/B 节点类型设备）和“3”（标准类型设备）：

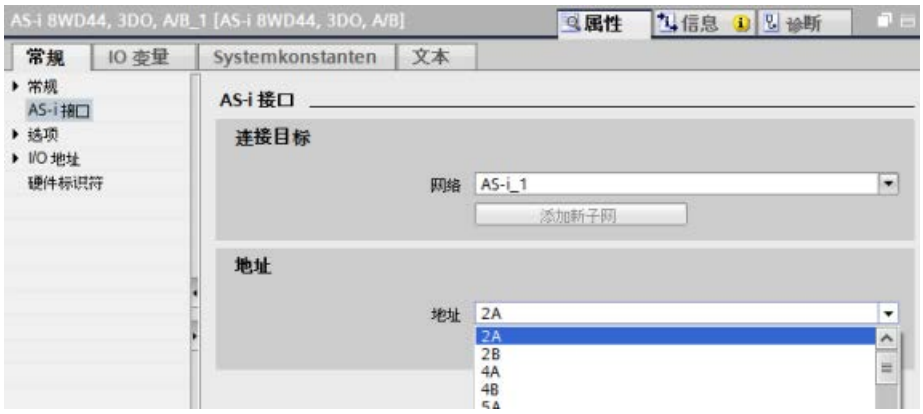


① AS-i 从站地址 1；设备： AS-i SM-U，4DI；订货号： 3RG9 001-0AA00

② AS-i 从站地址 2A；设备： AS-i 8WD44，3DO，A/B；订货号： 8WD4 428-0BD

③ AS-i 从站地址 3；设备： AS-i SM-U，2DI/2DO；订货号： 3RG9 001-0AC00

在此处输入 AS-i 从站地址：



表格 11-67 AS-i 接口的参数

参数	说明
网络	设备所连接到的网络的名称
地址	为从站设备分配的 AS-i 地址范围是从 1（A 或 B）到 31（A 或 B），总计最多 62 台从站设备

11.4.2 在用户程序和 AS-i 从站之间交换数据

11.4.2.1 STEP 7 基本组态

AS-i 主站在 CPU 的 I/O 区域中预留一个 62 字节的数据区。 在此将按照字节访问数字量数据；对于每个从站，都有一个字节的输入数据和一个字节的输出数据。

并在 AS-i 主站 CM 1243-2 的巡视窗口中，指示 AS-i 数字量从站到所分配字节数据位的 AS-i 连接分配。

可以通过相应位逻辑运算（如“AND”）的显示 I/O 地址或位分配，访问用户程序中 AS-i 从站的数据。

说明

如果未使用 STEP 7 组态 AS-i 从站，则自动激活 “系统分配”(System assignment)。

如果不组态任何从站，则必须使用在线功能“ACTUAL > EXPECTED”通知 AS-i 主站 CM1243-2 有关实际总线组态的信息。

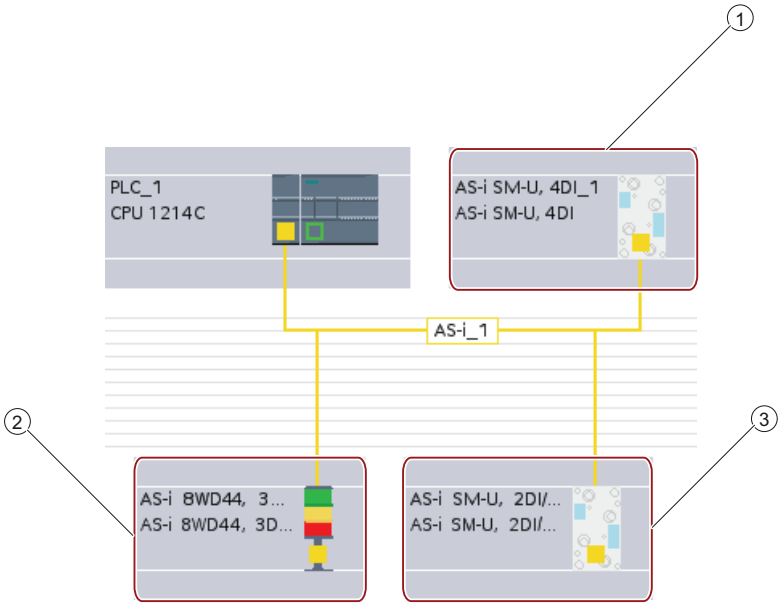
更多信息

有关 AS-i 主站 CM 1243-2 的详细信息，请参见“SIMATIC S7-1200 的 AS-i 主站 CM 1243-2 和 AS-i 数据解耦装置 DCM 1271”手册 (https://support.industry.siemens.com/cs/cn/zh/ps/15750/man)。

11.4.2.2 使用 STEP 7 组态从站

传输 AS-i 数字值

在循环操作中，CPU 通过 AS-i 主站 CM1243-2 访问 AS-i 从站的数字量输入和输出。 可以通过 I/O 地址或数据记录传输访问数据。



① AS-i 从站地址 1

② AS-i 从站地址 2A

③ AS-i 从站地址 3

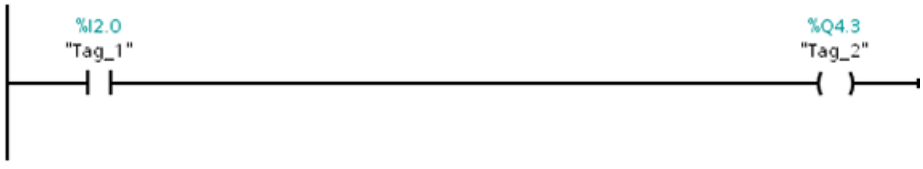
在此将按照字节访问数字量数据（即，每个 AS-i 数字量从站都对应一个字节）。 在 STEP 7 中组态 AS-i 从站时，将在相应 AS-i 的巡视窗口中显示访问用户程序中数据的 IO 地址。

上述 AS-i 网络中的数字量输入模块（AS-i SM-U、4DI）已分配了从站地址 1。单击该数字量输入模块，设备 “属性”(Properties) 的“AS 接口”(AS interface) 选项卡将显示从站地址，如下所示：

上述 AS-i 网络中的数字量输入模块（AS-i SM-U、4DI）已分配了 I/O 地址 2。单击该数字量输入模块，设备 “属性”(Properties) 的“I/O 地址”(I/O addresses) 选项卡将显示 I/O 地址，如下所示：

可以通过对 I/O 地址进行相应位逻辑运算（如“AND”）或位分配，来访问用户程序中 AS-i 从站的数据。 以下一段小程序举例说明了如何进行分配：

在本程序中将轮询输入 I2.0。 在 AS-i 系统中，该输入属于 从站 1（第 2 个输入字节，第 0 位）。 随后设置的输出 Q4.3 对应于 AS-i 从站 3（第 4 个输出字节，第 3 位）



传输 AS-i 模拟值

如果在 STEP 7 中已将该 AS-i 从站组态为模拟量从站，那么就可以通过 CPU 的过程映像访问 AS-i 从站的模拟量数据。

如果没有在 STEP 7 中组态模拟量从站，那么只能通过非周期性函数（数据记录接口）访问 AS-i 从站的数据。 在 CPU 的用户程序中，可以使用 RDREC（读取数据记录）和 WRREC（写入数据记录）分布式 I/O 指令读取和写入 AS-i 调用。

说明

在 S7 站的启动过程中，可以通过 AS-i 主站 CM1243-2 上的 CPU 传输通过 STEP 7 指定并下载到 S7 站中的 AS-i 从站的组态信息。 并会覆盖由“系统分配”在线功能 (页 997) ("ACTUAL -> EXPECTED") 确定的所有现有组态信息。

更多信息

有关 AS-i 主站 CM 1243-2 的详细信息，请参见“SIMATIC S7-1200 的 AS-i 主站 CM 1243-2 和 AS-i 数据解耦装置 DCM 1271”手册 (http://support.automation.siemens.com/WW/view/en/50414115/133300)。

11.4.3 分布式 I/O 指令

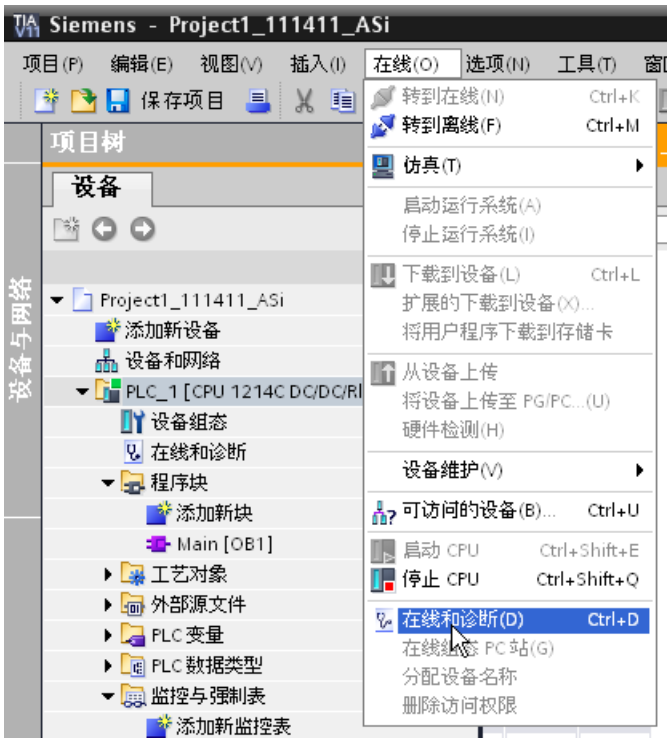
请参见“分布式 I/O（PROFINET、PROFIBUS 或 AS-i）” (页 385)，以了解有关如何将分布式 I/O 指令用于这些通信网络的信息。

11.4.4 使用 AS-i 在线工具

在线更改 AS-i 操作模式

必须在线查看和更改 AS-i 的操作模式。

要转到在线模式，必须先在“设备组态”(Device Configuration) 中选择 AS-i 主站 CM1243-2 模块，然后单击工具栏中的“转到在线”(Go online) 按钮。 然后，从“在线”(Online) 菜单中选择“在线和诊断”(Online and diagnostics) 命令。



有 2 种 AS-i 操作模式：

- 保护模式：
  - 不能更改 AS-i 从站设备和 CPU I/O 的地址。
  - 绿色“CM” LED 熄灭。
- 组态模式：
  - 可以在 AS-i 从站设备和 CPU I/O 地址进行相应更改。
  - 绿色“CM” LED 亮起。

在“设置 AS-i 地址”(Set AS-i address) 字段中，可以更改 AS-i 从站地址。 对于尚未分配地址的新从站，其地址始终为 0。在为其分配地址之前，它会被主站检测为尚未分配地址的新从站，并且不包括在常规通信范围之内。

组态错误

黄色“CER” LED 亮起时，表示 AS-i 从站设备组态中出现错误。 选择“ACTUAL > EXPECTED”按钮，用 AS-i 现场网络从站设备的组态覆盖 AS-i 主站 CM1243-2 模块从站设备的组态。

11.5 S7 通信

11.5.1 GET 和 PUT（从远程 CPU 读取和写入）

可以使用 GET 和 PUT 指令通过 PROFINET 和 PROFIBUS 连接与 S7 CPU 通信。仅当在本地 CPU 属性的“保护”(Protection) 属性中为伙伴 CPU 激活了“允许使用 PUT/GET 通信进行访问”(Permit access with PUT/GET communication) 功能后，才可进行此操作：

- 访问远程 CPU 中的数据：S7-1200 CPU 在 ADDR_x 输入字段中只能使用绝对地址对远程 CPU (S7-200/300/400/1200) 的变量寻址。
- 访问标准 DB 中的数据：S7-1200 CPU 在 ADDR_x 输入字段中只能使用绝对地址对远程 S7 CPU 标准 DB 中的 DB 变量寻址。
- 访问优化 DB 中的数据：S7-1200 CPU 不能访问远程 S7-1200 CPU 的优化 DB 中的 DB 变量。
- 访问本地 CPU 中的数据：S7-1200 CPU 可使用绝对地址或符号地址分别作为 GET 或 PUT 指令的 RD_x 或 SD_x 输入字段的输入。

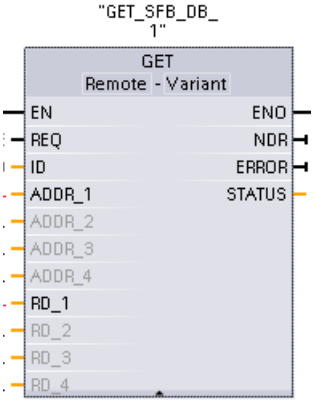
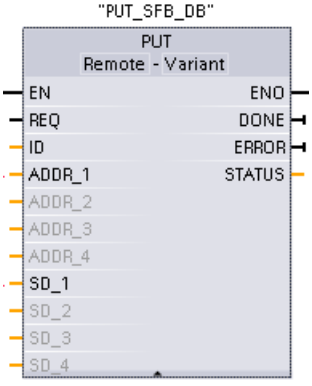
说明

V4.0 CPU 程序 GET/PUT 操作不会自动启用

V3.0 CPU 程序 GET/PUT 操作在 V4.0 CPU 中会自动启用。

不过，V4.0 CPU 中 V4.0 CPU 程序 GET/PUT 操作不会自动启用。要启用 GET/PUT 访问 (页 198)，必须转到 CPU“设备组态”(Device configuration)，打开巡视窗口，选择“属性” (Properties) 选项卡下的“保护”(Protection) 属性。

表格 11-68 GET 和 PUT 指令

LAD/FBD	SCL	说明
	"GET_DB"(req:=_bool_in_, ID:=_word_in_, ndr=>_bool_out_, error=>_bool_out_, status=>_word_out_, addr_1:=_remote_inout_, [...addr_4:=_remote_inout_,] rd_1:=_variant_inout_ [,...rd_4:=_variant_inout_]);	使用 GET 指令从远程 S7 CPU 中读取数据。远程 CPU 可处于 RUN 或 STOP 模式下。 STEP 7 会在插入指令时自动创建该 DB。
	"PUT_DB"(req:=_bool_in_, ID:=_word_in_, done=>_bool_out_, error=>_bool_out_, status=>_word_out_, addr_1:=_remote_inout_, [...addr_4:=_remote_inout_,] sd_1:=_variant_inout_, [....sd_4:=_variant_inout_]);	使用 PUT 指令将数据写入远程 S7 CPU。远程 CPU 可处于 RUN 或 STOP 模式下。 STEP 7 会在插入指令时自动创建该 DB。

表格 11-69 参数的数据类型

参数和类型		数据类型	说明
REQ	Input	Bool	通过由低到高的（上升沿）信号启动操作。
ID	Input	CONN_PRG (Word)	S7 连接 ID（十六进制）
NDR (GET)	Output	Bool	新数据就绪： • 0：请求尚未启动或仍在运行 • 1：已成功完成任务
DONE (PUT)	Output	Bool	DONE: • 0：请求尚未启动或仍在运行 • 1：已成功完成任务

参数和类型		数据类型	说明
ERROR STATUS	Output Output	Bool Word	• ERROR=0 STATUS 值： – 0000H：既没有警告也没有错误 – <> 0000H：警告，STATUS 提供详细信息 • ERROR = 1 出现错误。STATUS 提供有关错误性质的详细信息。
ADDR_1	InOut	远程	指向远程 CPU 中存储待读取 (GET) 或待发送 (PUT) 数据的存储区。
ADDR_2	InOut	远程	
ADDR_3	InOut	远程	
ADDR_4	InOut	远程	
RD_1 (GET) SD_1 (PUT)	InOut	Variant	指向本地 CPU 中存储待读取 (GET) 或待发送 (PUT) 数据的存储区。 允许的数据类型：Bool（只允许单个位）、Byte、Char、Word、Int、DWord、DInt 或 Real。 注：如果该指针访问 DB，则必须指定绝对地址，如： P# DB10.DBX5.0 Byte 10 在此情况下，10 代表 GET 或 PUT 的字节数。
RD_2 (GET) SD_2 (PUT)	InOut	Variant	
RD_3 (GET) SD_3 (PUT)	InOut	Variant	
RD_4 (GET) SD_4 (PUT)	InOut	Variant	

必须确保 ADDR_x（远程 CPU）与 RD_x 或 SD_x （本地 CPU）参数的长度（字节数）和数据类型相匹配。标识符“Byte”之后的数字是 ADDR_x、RD_x 或 SD_x 参数引用的字节数。

说明

通过 GET 指令可接收的字节总数或者通过 PUT 指令可发送的字节总数有一定的限制。具体限制取决于使用了四个可用地址和存储区中的多少：

- 如果仅使用 ADDR_1 和 RD_1/SD_1，则一个 GET 指令可获取 222 个字节，一个 PUT 指令可发送 212 个字节。
- 如果使用 ADDR_1、RD_1/SD_1、ADDR_2 和 RD_2/SD_2，则一个 GET 指令总共可获取 218 个字节，一个 PUT 指令总共可发送 196 个字节。
- 如果使用 ADDR_1、RD_1/SD_1、ADDR_2、RD_2/SD_2、ADDR_3 和 RD_3/SD_3，则一个 GET 指令总共可获取 214 个字节，一个 PUT 指令总共可获取 180 个字节。
- 如果使用 ADDR_1、RD_1/SD_1、ADDR_2、RD_2/SD_2、ADDR_3、RD_3/SD_3、ADDR_4、RD_4/SD_4，则一个 GET 指令总共可获取 210 个字节，一个 PUT 指令总共可发送 164 个字节。

各个地址和存储区参数的字节数之和必须小于等于定义的限值。如果超出这些限值，则 GET 或 PUT 指令将返回错误。

在 REQ 参数的上升沿出现时，读操作 (GET) 或写操作 (PUT) 将装载 ID、ADDR_1 和 RD_1 (GET) 或 SD_1 (PUT) 参数。

- 对于 GET：从下次扫描开始，远程 CPU 会将请求的数据返回接收区 (RD_x)。成功完成读取操作后，NDR 参数将置 1。新操作只有在之前的操作完成后才能开始。
- 对于 PUT：本地 CPU 开始将数据发送 (SD_x) 到远程 CPU 中的存储位置 (ADDR_x)。写操作顺利完成后，远程 CPU 返回执行确认。PUT 指令的 DONE 参数被设置为 1。新写入操作只有在之前操作完成后才能开始。

说明

为确保数据的一致性，应始终在访问数据或启动另一读/写操作前评估已经完成的操作（对于 GET 评估 NDR = 1；对于 PUT 评估 DONE = 1）。

ERROR 和 STATUS 参数提供有关读 (GET) 或写 (PUT) 操作的状态信息。

表格 11-70 错误信息

ERROR	STATUS（十进制）	说明
0	11	• 由于前一个作业还没有结束，所以不能执行新作业。 • 正在以较低优先级处理此作业。
0	25	通信已启动。正在处理作业。
1	1	通讯故障，如： • 未装载连接描述（本地或远程） • 连接被中断（例如：电缆断线、CPU 关闭或 CM/CB/CP 处于 STOP 模式） • 没有建立到通信伙伴的连接
1	2	来自伙伴设备的否定应答。无法执行任务。
1	4	发送区指针（GET 的 RD_x，或 PUT 的 SD_x）出错，包括数据长度或数据类型。
1	8	在伙伴 CPU 上发生访问错误
1	10	无法访问本地用户存储器（例如，尝试访问已经删除的数据块）
1	12	调用 SFB 时： • 指定了不属于 GET 或 PUT 的背景数据块 • 未指定背景数据块，而是指定了一个共享数据块 • 未发现背景数据块（装载新的背景数据块）

ERROR	STATUS（十进制）	说明
1	20	• 超出并行作业/实例的最大数量 • 当 CPU 处于 RUN 模式时，实例过载 首次执行 GET 或 PUT 指令时可能出现此状态
1	27	CPU 中没有相应的 GET 或 PUT 指令。

11.5.2 创建 S7 连接

连接机制

要使用 PUT/GET 指令访问远程连接伙伴，用户还必须得到许可。

默认情况下，“允许使用 PUT/GET 通信进行访问”(Permit access with PUT/GET communication) 选项处于未启用状态。 这时，只有需要对本地 CPU 和通信伙伴同时进行组态和编程的通信连接才能实现对 CPU 数据的读写访问。 例如，可以通过 BSEND/BRCV 指令进行访问。

因此，本地 CPU 仅作为服务器的连接（也就是说，本地 CPU 中不存在带有通信伙伴的通信组态/编程）在 CPU 运行期间不可用，例如：

- 通过通信模块进行 PUT/GET、FETCH/WRITE 或 FTP 访问
- 从其它 S7 CPU 进行 PUT/GET 访问
- 通过 PUT/GET 通信进行 HMI 访问

如果希望允许从客户端访问 CPU 数据，即不希望限制 CPU 的通信服务，要实现此级别的安全性，请参见“S7-1200 CPU 的访问保护 (页 198)”。

连接类型

所选的连接类型用于创建与伙伴站的通信连接。 控制器将设置、建立并自动监视该连接。

在“设备和网络”(Devices and Networks) 门户中，使用“网络视图”(Network view) 创建项目中各设备之间的网络连接。 首先，请单击“连接”(Connections) 选项卡，然后使用右侧的下拉框选择连接类型（例如 S7 连接）。 单击第一个设备上的绿色 (PROFINET) 框，然后拖出一条线连接到第二个设备上的 PROFINET 框。 松开鼠标按钮，即可创建 PROFINET 连接。

更多相关信息，请参见“创建网络连接” (页 811)。

单击“突出显示： 连接”(Highlighted: Connection) 按钮访问通信指令的“属性”(Properties) 组态对话框。

11.5.3 组态两台设备间的本地/伙伴连接路径

组态常规参数

在通信指令的“属性”(Properties) 组态对话框中指定通信参数。 只要选中了该指令的任何一部分，此对话框就会出现在页面底部附近。

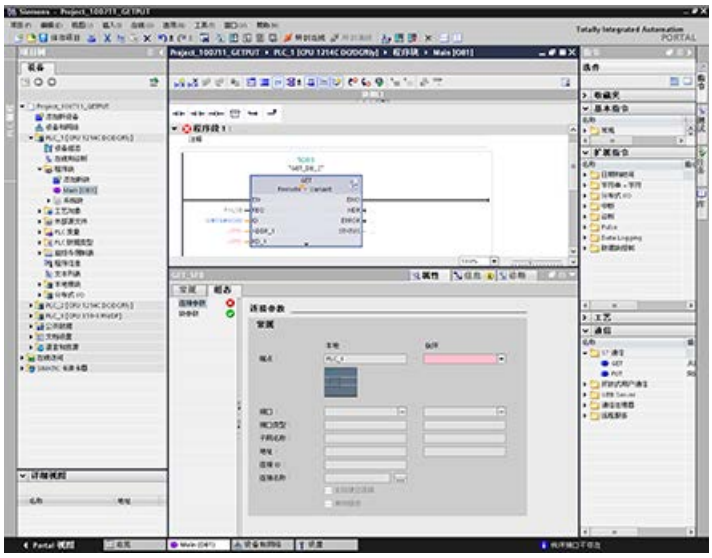
更多相关信息，请参见“设备配置： 组态本地/伙伴连接路径 (页 812)”。

在“连接参数”(Connection parameters) 对话框的“地址详细信息”(Address Details) 部分，定义要使用的 TSAP 或端口。 在“本地 TSAP”(Local TSAP) 字段中输入 CPU 中连接的 TSAP 或端口。 在“伙伴 TSAP”(Partner TSAP) 字段下输入为伙伴 CPU 中的连接分配的 TSAP 或端口。

11.5.4 GET/PUT 连接参数分配

GET/PUT 指令连接参数分配是一项用于 CPU 间 S7 通信连接组态的用户辅助功能。

插入 GET 或 PUT 块后，STEP 7 显示 GET/PUT 指令的连接参数分配对话框：



每次选择指令的任何一部分，巡视窗口都会显示连接的属性。可以在通信指令“属性”(Properties) 的“组态”(Configuration) 选项卡中组态通信参数。

说明

V4.1 及以上版本 CPU 程序 GET/PUT 操作不会自动启用

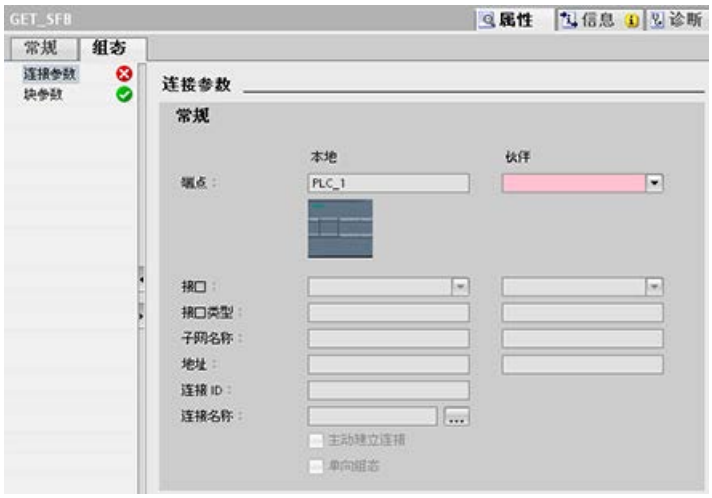
V3.0 CPU 程序 GET/PUT 操作在 V4.1 及以上版本 CPU 中会自动启用。

不过，V4.1 及以上版本 CPU 程序 GET/PUT 操作在 V4.1 及以上版本 CPU 中不会自动启用。要启用 GET/PUT 访问 (页 198)，必须转到 CPU“设备组态”(Device configuration)，打开巡视窗口，选择“属性”(Properties) 选项卡下的“保护”(Protection) 属性。

11.5.4.1 连接参数

在“连接参数”(Connection parameters) 页面中，可以组态必要的 S7 连接，以及组态由 GET/PUT 块参数“ID”引用的参数“连接 ID”(Connection ID)。 页面内容包括有关本地端点的信息，用户可在页面中定义本地接口。 您还可定义伙伴端点。

通过“块参数”(Block parameters) 页面可组态其它块参数。



表格 11-71 连接参数： 常规定义

参数		定义
连接参数：常规	端点	“本地端点”： 分配给本地 CPU 的名称 “伙伴端点”： 分配给伙伴（远程）CPU 的名称 注： 在“伙伴端点”(Partner End point) 下拉列表中，系统将显示当前项目中所有可能的 S7 连接伙伴以及选项“未指定”(unspecified)。 未指定伙伴是指当前不在 STEP 7 项目中的通信伙伴（例如，第三方设备通信伙伴）。
	接口	分配给接口的名称 注： 您可通过更改本地和伙伴接口来更改连接
	接口类型	接口类型
	子网名称	分配给子网的名称
	地址	分配的 IP 地址 注： 您可为“未指定”通信伙伴指定一个第三方设备远程地址。
	连接 ID	ID 号： 由 GET/PUT 连接参数分配自动生成
	连接名称	本地和伙伴 CPU 的数据存储位置： 由 GET/PUT 连接参数分配自动生成
	主动连接建立	用于选择本地 CPU 作为主动连接的复选框
	单向	指定单向或双向连接的复选框；只读 注： 在 PROFINET GET/PUT 连接中，本地与伙伴设备都可以作为服务器或客户端。 这样就可以进行双向连接，并取消选中“单向”(One-way) 复选框。 在某些情况下，PROFIBUS GET/PUT 连接中的伙伴设备只能作为服务器（例如 S7-300），并选中“单向”(One-way) 复选框。

连接 ID 参数

共有三种更改系统定义连接 ID 的方法：

1. 用户可在 GET/PUT 块中直接更改当前 ID。 如果新 ID 属于已存在的连接，则连接将更改。
2. 用户可在 GET/PUT 块中直接更改当前 ID，但不能有新 ID。 系统已创建新 S7 连接。
3. 可通过 “连接概况”(Connection overview) 对话框更改当前的 ID： 用户的输入与相应 GET/PUT 块中的 ID 参数同步。

说明

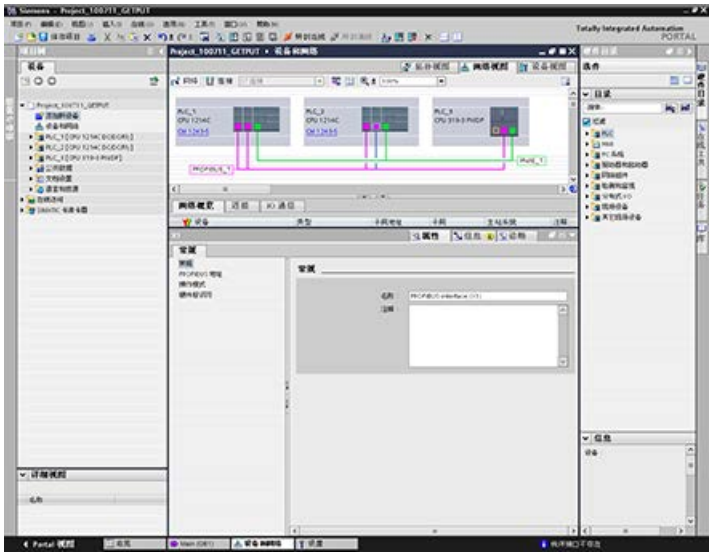
GET/PUT 块的参数“ID”不是连接名称，而是一个数字表达式，其写法类似于以下示例：W#16#1

连接名称参数

可以通过特殊用户控件（ “连接概况”(Connection overview) 对话框）编辑连接名称。 该对话框提供所有可用 S7 连接，可以选择这些连接作为当前 GET/PUT 通信的备选方式。 用户可在此表中创建全新的连接。 单击 “连接名称”(Connection name) 字段右侧的按钮，可启动 “连接概况”(Connection overview) 对话框。

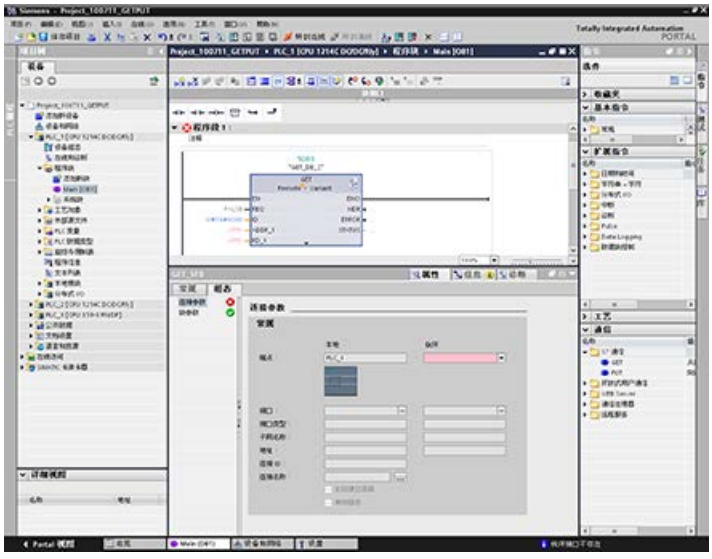
11.5.4.2 组态 CPU 间的 S7 连接

假设 PLC_1、PLC_2 和 PLC_3 的组态如下图所示，为“PLC_1”插入 GET 或 PUT 块。



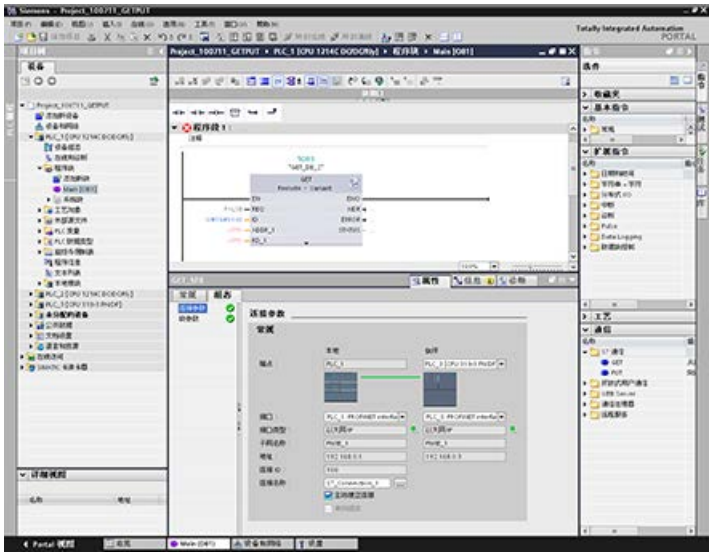
对于 GET 或 PUT 指令，将在巡视窗口中自动显示“属性”(Properties) 选项卡，且包含以下菜单选项：

- “组态”(Configuration)
- “连接参数”(Connection parameters)



组态 PROFINET S7 连接

对于“伙伴端点”，请选择“PLC_3”。



系统将进行以下更改以对此进行响应：

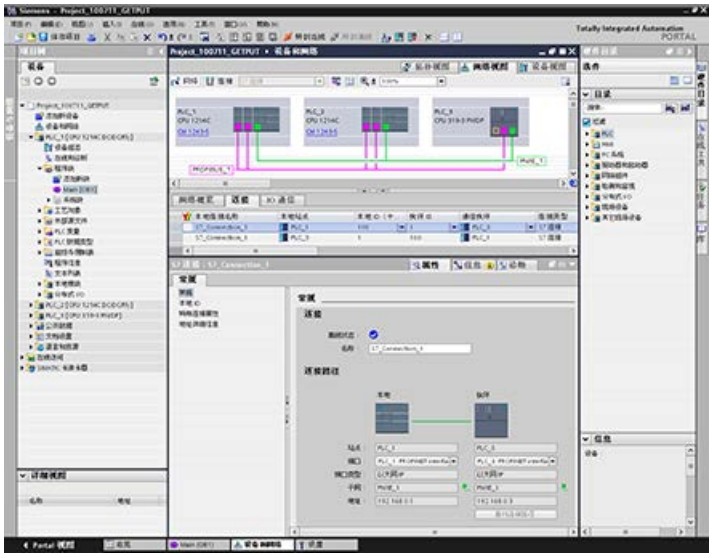
表格 11-72 连接参数：常规值

参数		定义
连接参数：常规	端点	“本地端点”中为只读的“PLC_1”。 “伙伴端点”字段中为“PLC_3[CPU319-3PN/DP]”： ● 颜色从红色变为白色 ● 显示“伙伴”设备图像。 ● PLC_1 和 PLC_3 设备图像之间有一条连接线（绿色以太网线）。
	接口	“本地接口”中为“CPU1214C DC/DC/DC、PROFINET interface (R0/S1)”。 “伙伴接口”中为：“CPU319-3PN/DP、PROFINET interface (R0/S2)”。
	接口类型	“本地接口类型”中为“Ethernet/IP”；控件为只读。 “伙伴接口类型”中为“Ethernet/IP”；控件为只读。 本地和伙伴“接口类型”（绿色以太网图标）旁显示接口类型的图像。
	子网名称	“本地子网名称”中为“PN/IE_1”；控件为只读。 “伙伴子网名称”中为“PN/IE_1”；控件为只读。
	地址	“本地地址”中为本地 IP 地址；控件为只读。 “伙伴地址”中为伙伴 IP 地址；控件为只读。
	连接 ID	“连接 ID”中为“100”。 在程序编辑器中，Main [OB1] 中的 GET/PUT 块“连接 ID”值也为“100”。
	连接名称	“连接名称”中为默认的连接名称（例如，“S7_Connection_1”）；控件已启用。
	主动连接建立	选中并启用，以选择本地 CPU 作为主动连接。
	单向	只读且取消选中。 注：“PLC_1”（S7-1200 CPU 1214CDC/DC/继电器）和“PLC_3” (S7-300 CPU 319-3PN/DP) 在 PROFINET GET/PUT 连接中都可以作为服务器和客户端，实现双向连接。

属性视图树中的 GET/PUT 图标也将从红色变为绿色。

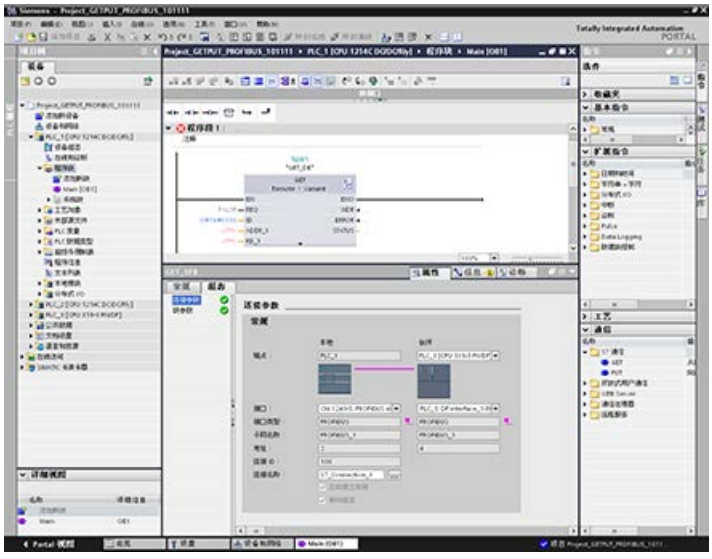
完成 PROFINET S7 连接

在“网络视图”(Network view) 中，将在“PLC_1”和“PLC_3”之间的“连接”(Connections) 表中显示 S7 双向连接。



组态 PROFIBUS S7 连接

对于“伙伴端点”，请选择“PLC_3”。



系统将进行以下更改以对此进行响应：

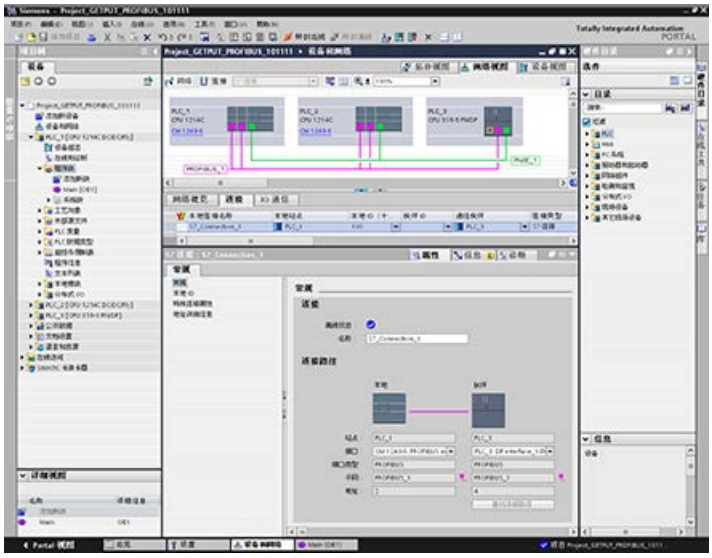
表格 11-73 连接参数： 常规值

参数		定义
连接参数：常规	端点	“本地端点”中为只读的“PLC_1”。 “伙伴端点”字段中为“PLC_3[CPU319-3PN/DP]”： ● 颜色从红色变为白色 ● 显示“伙伴”设备图像。 ● PLC_1 和 PLC_3 设备图像之间有一条连接线（紫色的 PROFIBUS 线）。
	接口	“本地接口”中为“CPU1214C DC/DC/DC、PROFIBUS interface (R0/S1)”。 “伙伴接口”中为： “CPU319-3PN/DP、PROFIBUS interface (R0/S2)”。
	接口类型	“本地接口类型”中为“PROFIBUS”；控件为只读。 “伙伴接口类型”中为“PROFIBUS”；控件为只读。 本地和伙伴“接口类型”（紫色的 PROFIBUS 图标）旁显示接口类型的图像。
	子网名称	“本地子网名称”中为“PROFIBUS _1”；控件为只读。 “伙伴子网名称”中为“PROFIBUS _1”；控件为只读。
	地址	“本地地址”中为本地 IP 地址；控件为只读。 “伙伴地址”中为伙伴 IP 地址；控件为只读。
	连接 ID	“连接 ID”中为“100”。 在程序编辑器中，Main [OB1] 中的 GET/PUT 块“连接 ID”值也为“100”。
	连接名称	“连接名称”中为默认的连接名称（例如，“S7_Connection_1”）；控件已启用。
	主动连接建立	只读，选中并启用，以选择本地 CPU 作为主动连接。
	单向	只读且选中。 注： “PLC_3” (S7-300 CPU319-3PN/DP) 在 PROFIBUS GET/PUT 连接中只能作为服务器（无法同时作为客户端），只能进行单向连接。

属性视图树中的 GET/PUT 图标也将从红色变为绿色。

完成 PROFIBUS S7 连接

在 “网络视图”(Network view) 中，将在“PLC_1”和“PLC_3”之间的 “连接”(Connections) 表中显示 S7 单向连接。



11.6 无法通过 IP 地址访问 CPU 时的做法

如果无法通过 IP 地址访问 CPU，可以为 CPU 设置紧急（临时）IP 地址。通过紧急 IP 地址，可以重新建立与 CPU 的通信，以便使用有效 IP 地址下载设备组态。

可能需要紧急 IP 地址的理由

如果有人在下载某个项目时遇到以下问题之一，则 CPU 可能无法访问：

- CPU 的 PROFINET 接口的 IP 地址与网络上的其它设备重复。
- CPU 的子网不正确。
- 子网掩码使 CPU 无法访问。

在上述情况下，无法再从 STEP 7 访问 CPU。

分配紧急 IP 地址

可以在以下条件下分配紧急 IP 地址：

- STEP 7 中的设备组态在 IP 协议中有 “在项目中设置 IP 地址”(Set IP address in the project)。
- CPU 处于 STOP 模式。

在上述情况下，可以使用 DCP 工具将设备的 IP 地址设置为紧急 IP 地址。例如，SIMATIC 自动化工具具有“DCP 设置 IP 地址”(DCP Set IP address) 命令。可以设置紧急 IP 地址，而不受 CPU 的保护等级 (页 198)限制。使用 DCP 工具设置临时 IP 地址后，CPU 上的维护 LED 将亮起。诊断缓冲区还包括一个指示用户已启用以太网接口的紧急地址的条目。

分配紧急 IP 地址后恢复 IP 地址

诊断缓冲区会在启用或禁用紧急 IP 地址时通知用户。可以通过关闭和打开 CPU 来重置紧急 IP 地址。

在分配了紧急 IP 地址后，可以使用 CPU 的有效 IP 地址下载 STEP 7 项目。下载项目后，重新启动 CPU。

11.7 OPC UA 服务器

固件版本为 4.4 或更高版本的 S7-1200 CPU 支持 OPC UA 通信。该 CPU 支持组态为 OPC UA 服务器，从而允许数据访问。

OPC 基金会 (https://opcfoundation.org/)提供 OPC UA 通信核心功能的规范。S7-1200 CPU 支持一部分可能的 OPC UA 功能。以下各节描述了一部分 OPC UA 功能。

参见

新功能 (页 37)

11.7.1 OPC UA 服务器组态

在 TIA Portal 的 CPU 硬件属性中组态 OPC UA 服务器。

11.7.1.1 激活 OPC UA 服务器

默认情况下，OPC UA 服务器处于未激活状态。必须在 CPU 的硬件属性中选中“激活 OPC UA 服务器”(Activate OPC UA server) 复选框，来显式激活 OPC UA 服务器。

11.7.1.2 OPC UA 服务器的设置

下表列出了为 OPC UA 服务器组态的设置。该表也描述了 S7-1200 V4.4 固件的服务器限制：

服务器设置	最小值	最大值	默认值	是否可由用户调节？
通信端口	1024	49151	4840	√
最大会话超时	1s	600,000s	30s	√
并发会话	1	5	2	√
服务器接口上的节点 [1]		1000		×
最大已注册节点 [2]	100	1000	200	√
总命名空间 [3]		18		×
订阅限制				
最短采样时间间隔	100ms、250ms、500ms、1000ms、5000ms、10000ms		1000ms	√
最短发布时间间隔	200ms	20,000,000ms	1000ms	√
并发订阅		5		×
最大受监视节点数	100	500	200	√

[1] 这是用户定义节点的最大数量，不包括在服务器接口上隐式定义的节点。

[2] 组态了“已注册”节点以进行快速查找。服务器为已注册节点表保留空间。该保留区的大小取决于用户定义的“最大注册节点数”。

[3] 这是为所有服务器接口定义的命名空间和引用命名空间的总数。

11.7.2 OPC UA 服务器安全

OPC UA 服务器需要证书来激活。激活服务器时，TIA Portal 会自动生成证书。可以在 PLC 属性中更改证书。

11.7.2.1 支持的安全策略

可以选择服务器支持的安全策略。OPC UA 客户端在运行时选择的安全策略用于确定客户端和服务器之间的通信安全性。

S7-1200 支持以下安全策略：

安全策略	默认为启用？
无安全	√
Basic128Rsa15 – 签名	×
Basic128Rsa15 – 签名和加密	×
Basic256 – 签名	×
Basic256 – 签名和加密	×
Basic256Sha256 – 签名	√
Basic256Sha256 – 签名和加密	√

11.7.2.2 受信客户端

默认情况下，OPC UA 服务器允许来自任何客户端的连接。

或者，可以定义受信客户端的列表。受信客户端由其证书来标识。选中后，只允许出示受信证书的客户端连接到服务器。

要指定受信客户端，请在 TIA Portal 中将这些客户端的证书添加到“硬件属性 > OPC UA 安全 > 安全通道 > 受信客户端”(Hardware properties > OPC UA security > Secure channel > Trusted clients) 下的“受信客户端”(Trusted clients) 列表中。

11.7.2.3 用户认证

S7-1200 支持访客认证和用户名/密码认证两种方式。默认为访客认证。

启用访客认证后，连接时，客户端不会要求用户提供用户名和密码。

禁用访客认证后，仅允许提供了之前指定的用户名和密码的客户端进行连接。

可以通过两种方式为 OPC UA 服务器定义有效用户：

- 在 CPU 的硬件属性中定义有效用户。
- 如果已为项目启用了全局安全，则会通过为各个 TIA Portal 项目用户指定的角色来识别有效用户。最多可组态 21 个用户。

11.7.3 OPC UA 服务器接口

S7-1200 OPC UA 服务器不支持标准 SIMATIC 服务器接口。而用户必须在 TIA Portal 中定义服务器接口的结构和内容，然后将其下载到 PLC。

要添加服务器接口，请按以下步骤操作：

- 在项目树中，单击 PLC 名称
- 选择“OPC UA 通信”(OPC UA communication)
- 选择 “服务器接口”(Server interfaces)
- 选择 “添加新服务器接口”(Add new server interface)
- 输入服务器相关信息
- 将服务器信息下载到 PLC

11.7.3.1 支持的数据类型

OPC 基金会 (https://opcfoundation.org/)定义了一组支持的数据类型，这些数据类型用于描述变量及其变量类型的 Value 属性的结构。S7-1200 V4.4 支持其中一部分数据类型，以及其它源自受支持数据类型的定义类型。

下表列出了 S7-1200 V4.4 支持的数据类型：

SIMATIC 类型	OPC UA 类型名称	节点标识符
Bool	Boolean	i=1
SInt	SByte	i=2
USInt	Byte	i=3
Int	Int16	i=4
UInt	UInt16	i=5
DInt	Int32	i=6
UDInt	UInt32	i=7
Real	浮点型	i=10
LReal	双精度浮点型	i=11
WString	字符串	i=12
DWord	StatusCode	i=19

请注意，该列表表示受支持的基本节点类型，而不表示受支持节点类型的完整列表，因为许多 SIMATIC 数据类型都映射到基本节点类型。映射到基本节点类型的任何 SIMATIC 数据类型也是受支持的节点类型。

S7-1200 服务器接口支持读取和写入简单数据类型的节点。不支持以下各项：

- 结构化数据类型
- 数组
- 联合
- 服务器方法

S7-1200 接受类型不受支持的服务器的下载操作。但是，如果客户端尝试读取或写入类型不受支持的节点，PLC 将返回错误。

11.7.3.2 PLC 表示

OPC UA 服务器接口提供表示 PLC 的节点，这些节点的属性用于描述 PLC。只要激活 OPC UA 服务器，即会提供这些节点。

服务器接口中提供以下信息：

属性	值		
NodeId	Ns=SI;s=PLC		
BrowseName	SI:<PLC> 其中 <PLC> 是用户分配给其 TIA Portal 项目中 PLC 的名称		
References	**NodeClass**	**BrowseName**	**注释**
DeviceSet 对象的组件			
按对象文件夹组织			
HasTypeDefinition	ObjectType	SimaticDeviceType	源自设备类型
HasProperty	变量	DeviceManual	
HasProperty	变量	DeviceRevision	
HasProperty	变量	EngineeringRevision	
HasProperty	变量	HardwareRevision	
HasProperty	变量	图标	
HasProperty	变量	制造商	
HasProperty	变量	型号	
HasProperty	变量	OperatingMode	
HasProperty	变量	OrderNumber	
HasProperty	变量	RevisionCounter	

属性	值		
HasProperty	变量	SerialNumber	
HasProperty	变量	SoftwareRevision	

11.7.3.3 可下载的服务器接口

在 TIA Portal 中创建和编辑 OPC UA 服务器接口的组件。可以通过两种方式定义 OPC UA 服务器接口：

- 这些接口可能基于外部创建的 XML 文件（称为配套规范）
- 可以根据程序中包含的数据块元素和全局变量在 TIA Portal 中直接定义这些接口

下载后，这些接口定义了对 OPC UA 客户端可见的数据接口。

需要运行系统许可证

运行 S7-1200 CPU 的 OPC UA 服务器需要使用运行系统许可证。许可类型包含有：

- SIMATIC OPC UA S7-1200 Basic DVD 6ES7823-0BA00-2BA0
- SIMATIC OPC UA S7-1200 Basic DL 6ES7823-0BE00-2BA0

所需许可证类型显示在“属性 > 常规 > 运行系统许可证 > OPC-UA > 所需许可证类型”(Properties > General > Runtime licenses > OPC-UA > Type of required license) 下。若要确认购买所需许可证，请按照以下步骤进行操作：

1. 单击 CPU 属性中的“运行许可证 > OPC UA”(Runtime licenses > OPC UA)。
2. 在“购买的许可证类型”(Type of purchased license) 下拉列表中，选择所需的许可证。

Web 服务器

12

借助 S7-1200 的 Web 服务器，用户可经由 Web 页面来访问 CPU 相关数据以及过程数据。

可通过 PC 或移动设备访问 S7-1200 Web 页面。对于小屏幕设备，Web 服务器支持一系列基本页面 (页 1037)。

使用 Web 浏览器通过 CUP 建立连接可访问 S7-1200 CPU 的 IP 地址，或访问本地机架中 已启用 Web 服务器的 CP（通信处理器）模块 (页 1033)的 IP 地址。S7-1200 支持多个并发连接。

说明

Web 服务器多个并发连接

S7-1200 Web 服务器允许 30 个并发连接。每个打开的浏览器实例可以占用 2 到 8 个连接。Web 服务器最多允许 7 个登录用户，但西门子建议尽可能地减少并发用户的数量。在平均工作负载下，通常平均一次登录 7 个用户是正常的。

标准 Web 页面

S7-1200 包括标准 Web 页面 (页 1036)，您可以从 PC 的 Web 浏览器 (页 1030)或一台移动设备 (页 1032)进行访问：

- 介绍 (页 1041) - 标准 Web 页面的进入点
- 起始页面 (页 1041) - 有关 CPU 的常规信息
- 诊断 (页 1042) - 有关 CPU 的详细信息，包括序列号、订单号、版本号、程序保护和存储器使用情况
- 诊断缓冲区 - 诊断缓冲区
- 模块信息 (页 1045) - 有关本地模块和远程模块以及更新本地模块固件的能力
- 通信 (页 1049) - 有关网络地址、通信接口物理属性、统计、参数的信息，以及连接概要和诊断信息
- 变量状态 (页 1052) - CPU 变量和 I/O，可通过地址或 PLC 变量名访问

- 监控表 (页 1054) - 在 STEP 7 中组态的监控表
- 在线备份 (页 1056) - 能够备份在线 CPU 或恢复之前进行的在线备份
- 数据日志 (页 1058)- 可用于查看 PLC 上所有数据日志的列表，将数据日志从 PLC 下载到计算机，从 PLC 中删除数据日志，以及检索并清除 PLC 中的数据日志
- 用户文件 (页 1061)- 可用于查看 PLC 上用户文件的列表，将用户文件从 PLC 下载到计算机，将用户文件从计算机上传到 PLC，以及删除 PLC 上的用户文件
- 用户定义的页面 (页 1066)- 创建用户定义的 Web 页面以访问 CPU 数据
- 文件浏览器 (页 1065) - 用于浏览存储在 CPU 或存储卡内部的文件（如数据日志和配方）的浏览器
- 登录 (页 1037) - 以其他用户身份登录，或注销。

S7-1200 CPU 中含有这些页面，提供英语、德语、法语、西班牙语、意大利语和简体中文等版本。要浏览除“简介”(Introduction) 和“开始”(Start) 页面外的所有页面，需要额外在 STEP 7 中组态用户权限 (页 1029)。

说明

Web 服务器启用其它语言

可以选择其它语言显示标准 Web 页面。从设备组态的“属性”(Properties) 选项卡中选择“用户接口语言”(User interface languages)，将 STEP 7 项目语言分配给所支持的六种语言之一。下载设备组态后，标准 Web 页面将显示用户界面语言选择器。

用户定义的 Web 页面

S7-1200 还支持您创建可访问 CPU 数据的用户定义的 Web 页面。可以使用所选的 HTML 创作软件来开发这类页面，并且可将预定义的“AWP”（Automation Web Programming，自动化 Web 编程）命令包含在 HTML 代码中以访问 CPU 数据。有关开发用户定义 Web 页面以及在 STEP 7 中进行相关组态和编程的具体信息，请参见用户定义的 Web 页面 (页 1066) 一章。

您可以通过标准或基本 Web 页面从 PC 或移动设备访问用户自定义页面。您还可以为 Web 服务器组态其中一个用户自定义 Web 页面为入口页面 (页 1085)。

Web 浏览器要求

Siemens 已经对 Web 服务器标准页面进行测试，并验证了对以下 Web 浏览器的支持情况：

- Internet Explorer 8 到 11
- Microsoft Edge

- Mozilla Firefox V22 到 V32，V42 到 V47
- Google Chrome V33 到 V38，V46 到 V47
- IOS 9 设备的 Mobile Safari 和 Mobile Chrome
- 以下系统版本的安卓浏览器：
  - Jellybean v4.3
  - Kitkat v4.4
  - Lollipop V5.0 到 v5.1
  - Marshmellow v6.0
- Google Android 的 Mobile Chrome

当在 WinCC 项目中使用 HTML 浏览器控制时，Web 服务器支持以下 Siemens HMI 面板访问标准页面：

- 精简系列面板
  - Gen 2 KTP400 至 KTP1200
- 精智面板
  - TP700 至 TP2200
  - KP400 至 KP1500
  - KTP400
  - TP700 Comfort Outdoor
- 移动面板
  - Gen 2 KTP700[F]、KTP900[F]

有关可干扰标准或用户定义 Web 页面显示的浏览器相关限制，请参见限制 (页 1108)部分。

Web 服务器性能

许多因素都会影响 Web 服务器的性能。S7-1200 CPU 和编程设备必须与消耗资源和处理时间的其它任务共享时间。如果 Web 服务器的性能较差，请尝试进行以下调整，以提高 Web 服务器性能：

- 将 PLC 上的通信负载 (页 103)从 20% 增加到 50%。
- 组态最短扫描时间 (页 103)。设置最短扫描时间可增加 S7-1200 CPU 与编程设备之间的通信时间。
- 使用 S7-1200 CPU 的以太网接口，而非 CP 模块 (页 1033)，以访问 Web 服务器。

参见

Diagnostic Buffer (页 1045)

12.1 启用 Web 服务器

在 STEP 7 中，通过“设备组态”(Device Configuration) 为要连接的 CPU 启用 Web 服务器。

要启用 Web 服务器，请按以下步骤操作：

1. 在设备组态视图中选择 CPU。
2. 在巡视窗口中，从 CPU 属性中选择“Web 服务器”(Web server)。
3. 选中“激活此设备所有模块上的 Web 服务器”(Activate web server on all modules of this device) 复选框。
4. 出于安全考虑，为了对 Web 服务器进行安全访问，确保选中“仅允许使用 HTTPS 访问”(Permit access only with HTTPS)。
5. 如果选择了“自动更新”(Automatic update) 的“启用自动更新”(Enable automatic update)，则标准 Web 页面将默认每十秒刷新一次。您也可以在“更新间隔”(Update interval) 字段中输入自定义刷新时间周期，单位为秒。

⚠ 警告

通过 Web 服务器对 CPU 进行未经授权的访问

未经授权访问 CPU 或将 PLC 变量更改为无效值可能会中断过程操作并可能导致死亡、严重人身伤害和/或财产损失。

由于启用 Web 服务器后授权用户可执行操作模式更改、写入 PLC 数据以及固件更新，Siemens 建议遵照以下安全实践：

- 仅使用 HTTPS 协议启用对 Web 服务器的访问。
- 使用可靠的密码对 Web 服务器用户 ID 进行密码保护 (页 1029)。强密码在长度上至少为十个字符，可以是字母、数字和特殊字符的组合，不能是可在字典上找到的词，并且不能是可从个人信息推断出的名字或标识符。保管好密码并经常更改密码。
- 不要扩展“所有人”(Everybody) 用户的默认最低权限。
- 对程序逻辑中的变量执行错误检查和范围检查，因为 Web 页面用户可将 PLC 变量更改为无效值。
- 如果您不在受保护的网络范围内，请使用安全的虚拟专用网络 (VPN) 连接到 S7-1200 PLC Web 服务器。

下载设备组态后，可使用标准 Web 页面访问 CPU 的简介和开始页面。要访问其它页面，用户必须组态一个或多个 Web 服务器用户 (页 1029)。

如果创建并启用了用户自定义 Web 页面 (页 1066)，则可通过标准 Web 页面或基本 Web 页面的导航菜单访问这些页面。

说明

设备更换：使用 V4.x CPU 替换 V3.0 CPU

如果您使用 V4.x CPU 替换现有的 V3.0 CPU (页 1667) 并将您的 V3.0 项目转换为 V4.x 项目，请注意，STEP 7 和 V4.x CPU 将为以下两项保持 Web 服务器设置

- “激活此设备所有模块上的 Web 服务器”(Activate web server on all modules of this device)
- 仅允许使用 HTTPS 访问 (Permit access only with HTTPS)

说明

如果正在“在 RUN 模式下进行下载” (页 1406)，那么在下载完成之前，标准和用户定义的 Web 页面不会更新数据值，也不允许写入任何数据值。下载期间，Web 服务器会放弃写入数据值的任何尝试。

12.2 组态 Web 服务器用户

您可为用户组态通过 Web 服务器访问 CPU 的各种权限级别。

要组态 Web 服务器用户及其相关权限，请按以下步骤操作：

1. 在设备组态视图中选择 CPU。
2. 在巡视窗口的 CPU 属性中选择“Web 服务器”(Web server)，启用 Web 服务器 (页 1028)。
3. 在 Web 服务器属性中选择 “用户管理”(User management)。
4. 为想要使用的用户登录输入用户名、访问级别和密码。

将组态下载到 CPU 后，只有授权用户才能以相应权限访问 Web 服务器功能。

Web 服务器访问级别

STEP 7 提供的默认用户名称为 “所有人”(Everybody)，没有密码。默认情况下，此用户没有任何附加权限，只能查看开始 (页 1041)和简介 (页 1041)两个标准的 Web 页面。不过，可以为 “所有人”(Everybody) 用户和其他用户组态附加权限：

- 查询诊断
- 读取变量
- 写入变量
- 读取变量状态
- 写入变量状态

- 打开“用户自定义 Web”(user-defined web) 页面
- 在用户自定义 Web 页面中进行写操作
- 读取文件
- 写入/删除文件
- 更改工作模式
- 闪烁 LED
- 执行固件更新
- 备份 CPU
- 恢复 CPU
- 更改系统参数
- 更改应用程序参数

如果为 Web 服务器设置用户自定义 Web 页面为入口页面 (页 1085)，则“所有人” (Everybody) 用户必须具备“打开用户自定义 Web 页面”的权限。

警告

访问 Web 服务器

授予“所有人”(Everybody) 用户相应权限，即可在没有密码的情况下登录 Web 服务器。未经授权访问 CPU 或将 PLC 变量更改为无效值可能会中断过程操作并可能导致死亡、严重人身伤害和/或财产损失。

由于具有足够权限的“所有人”(Everybody) 用户能够在没有密码的情况下执行工作模式更改、写入 PLC 数据以及进行固件更新，Siemens 建议遵照以下安全实践

- 仅使用 HTTPS 协议启用对 Web 服务器的访问。
- 使用可靠的密码对 Web 服务器用户 ID 进行密码保护。强密码在长度上至少为十个字符，可以是字母、数字和特殊字符的组合，不能是可在字典上找到的词，并且不能是可从个人信息推断出的名字或标识符。保管好密码并经常更改密码。
- 不要扩展“所有人”(Everybody) 用户的默认最低权限。
- 对程序逻辑中的变量执行错误检查和范围检查，因为 Web 页面用户可将 PLC 变量更改为无效值。
- 如果您不在受保护的网络范围内，请使用安全的虚拟专用网络 (VPN) 连接到 S7-1200 PLC Web 服务器。

12.3 通过 PC 访问 Web 页面

可以通过 S7-1200 CPU 或本地机架中任意已启用 Web 服务器的 CP (页 1033) 的 IP 地址从 PC 或从移动设备访问 S7-1200 的标准 Web 页面。

要通过 PC 访问 S7-1200 的标准 Web 页面，请按以下步骤操作：

1. 请确保 S7-1200 和 PC 位于同一个以太网中，或二者之间直接使用标准以太网电缆进行连接。
2. 打开 Web 浏览器，输入 URL“https://ww.xx.yy.zz”，其中“ww.xx.yy.zz”与 S7-1200 CPU 或本地机架中 CP 的 IP 地址对应。

Web 浏览器打开“简介”(Introduction) 标准 Web 页面 (页 1041)，或者如果将用户自定义 Web 页面组态为入口页面 (页 1085)，将打开其默认 HTML 页面。

说明

如果您不在受保护的网络范围内，请使用安全的虚拟专用网络 (VPN) 连接到 S7-1200 PLC Web 服务器。另外，还要注意 Web 环境或操作系统可能造成的任何限制 (页 1108)。

通过输入页面 URL 访问标准网页

可以通过页面的 URL 访问特定的标准网页。为此，请以“https://ww.xx.yy.zz/<page>.html”的方式输入 URL，其中“ww.xx.yy.zz”与 S7-1200 CPU 或本地机架中 CP 的 IP 地址对应：

- https://ww.xx.yy.zz/start.html - 显示有关 CPU 常规信息的起始页面 (页 1041)
- https://ww.xx.yy.zz/identification.html - 显示有关 CPU 的标识 (页 1042)信息，包括序列号、订单号和版本号，现在称作“诊断”(Diagnostics) 页面
- https://ww.xx.yy.zz/module.html - 有关本地机架中的模块和固件更新能力 (页 1045)的信息
- https://ww.xx.yy.zz/communication.html - 有关网络地址、通信接口的物理属性和通信统计的通信 (页 1049)信息
- https://ww.xx.yy.zz/diagnostic.html - 诊断缓冲区 (页 1045)
- https://ww.xx.yy.zz/variable.html - CPU 变量（变量）和 I/O (页 1052)，可以通过地址、PLC 变量名或数据块变量名进行访问
- https://ww.xx.yy.zz/watch.html - 监控表 (页 1054)
- https://ww.xx.yy.zz/filebrowser.html - 用于访问存储在 CPU 内部或存储卡中的数据日志文件或配方文件的浏览器 (页 1065)
- https://ww.xx.yy.zz/index.html - 进入标准 Web 页面的简介页面 (页 1041)
- https://ww.xx.yy.zz/login.html - 用户当前未进行登录时的登录 (页 1037)页面，否则，页面为空。

例如，如果输入“https://ww.xx.yy.zz/communication.html”，浏览器将显示通信页面。

说明

请注意任何以上没有明确列出的 Web 页面（例如，“在线备份”(Online backup) 页面 (页 1056)）都不能直接访问 URL。

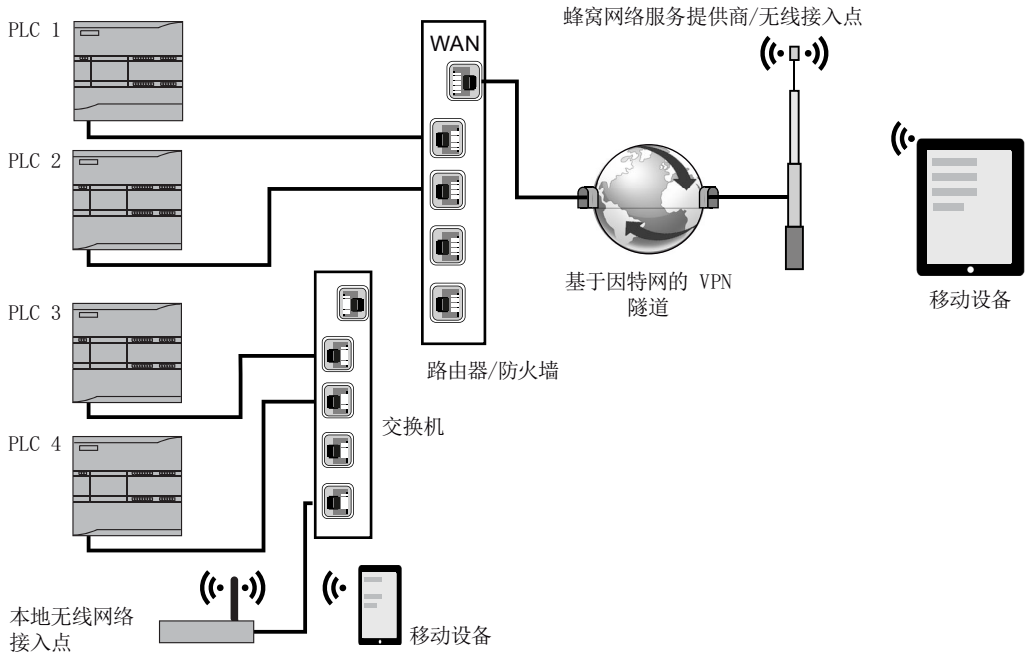
安全访问

如果您不在受保护的网络范围内，请使用安全的虚拟专用网络 (VPN) 连接到 S7-1200 PLC Web 服务器。要求使用 https://（而不是 http://）对标准 Web 页面进行安全访问 (页 1028)。使用 https:// 连接到 S7-1200 时，网站将通过数字证书对会话进行加密。Web 服务器将安全地发送数据，而不会被任何人看到。通常，您会收到安全警告，可以按“是”(Yes) 继续浏览标准 Web 页面。要避免每次安全访问时都出现安全警告，可以将 Siemens 软件证书导入 Web 浏览器 (页 1034)。

12.4 通过移动设备访问 Web 页面

要通过移动设备访问 S7-1200，必须将 PLC 连接到与 Internet 或本地无线接入点相连的网络。使用安全的虚拟专用网络 (VPN) 将移动设备连接到 S7-1200 PLC Web 服务器。可以使用无线路由器中的端口转发功能将 PLC 的 IP 地址映射到移动设备可通过 Internet 进行访问的地址。要组态端口转发功能，请按路由器软件组态的说明进行操作。路由器支持多少 PLC 和开关设备，您就可以连接多少。

没有端口转发时，您可以连接到 PLC，但只能在无线信号的范围内进行本地连接。



在此示例中，处于本地无线接入点范围内的移动设备可根据相应的 IP 地址连接到 PLC 3 和 PLC 4。移动设备可以通过本地无线范围外的 Internet，使用各 PLC 的端口转发地址连接到 PLC 1 和 PLC 2。

要访问标准 Web 页面，必须能够访问蜂窝服务或无线接入点。要通过 Internet 访问 PLC，应在要访问 PLC 的移动设备的 Web 浏览器中输入端口转发地址，例如 http://ww.xx.yy.zz:pppp 或 https://ww.xx.yy.zz:pppp，其中 ww.xx.yy.zz 是路由器地址，pppp 是特定 PLC 的端口分配。

要通过本地无线接入点进行本地访问，请输入 S7-1200 CPU 或本地机架中已启用 Web 服务器的 CP (页 1033) 的 IP 地址：

- http://ww.xx.yy.zz 或 https://ww.xx.yy.zz 访问标准 Web 页面 (页 1036)
- http://ww.xx.yy.zz/basic 或 https://ww.xx.yy.zz/basic 访问基础 Web 页面 (页 1037)

为获得更高安全性，请将 Web 服务器配置为只能通过安全访问 (HTTPS) (页 1028) 来访问。

12.5 通过 CP 模块访问 Web 页面

不论是从 PC 还是从移动设备访问 Web 服务器，如果已在 STEP 7 中组态了以下 CP 模块之一并将其安装在具有 S7-1200 CPU 的本地机架中，就可以通过它连接到 Web 页面：

- CP 1242-7 GPRS V2
- CP 1243-1

- CP 1243-7 LTE-EU
- CP 1243-7 LTE-US
- CP 1243-8 IRC

可以使用“起始”标准 Web 页面 (页 1041)通过这些 CP 模块访问 Web 页面。“起始”页面将显示您的本地机架中拥有的所有已组态和已安装的 CP 模块，但只能从以上所列的模块访问 Web 页面。

说明

已启用 Web 服务器的 CP 位于本地机架中时对标准 Web 页面的访问

如果已启用 Web 服务器的 CP 位于本地机架，则在连接到 S7-1200 标准 Web 页面时，可能会观察到一或两分钟的延迟。如果页面不可用，或出现错误时，只需等待一或两分钟，然后刷新页面。

12.6 下载和安装安全证书

可以将默认 Siemens 安全证书下载到 Internet 选项中。

借助该证书，在 Web 浏览器中输入 https://ww.xx.yy.zz（其中“ww.xx.yy.zz”是设备的 IP 地址）时，就不必进行安全验证。如果使用 http:// URL，而不是 https:// URL，则不需要下载并安装该证书。

在支持 S7-1200 V4.3 CPU 的 STEP 7 V15 SP1 或更高版本中，可以在 S7-1200 CPU 的设备组态中创建证书。此功能位于设备的“保护和安全 > 证书管理器”(Protection & Security > Certificate manager) 一般设置下。有关证书管理器以及如何创建全局和本地 CPU 专用证书的说明，请参见 STEP 7 信息系统。

此外，在支持 S7-1200 V4.3 CPU 的 STEP 7 V15 SP1 或更高版本中，还可以为 S7-1200 CPU 的 Web 服务器创建证书。

下载证书

使用简介页面 (页 1041)中的“下载证书”(download certificate) 链接将 Siemens 安全证书下载到 PC 中。下载和导入程序会因所用 Web 浏览器而异。

将证书导入 Internet Explorer

1. 单击“简介”(Introduction) 页面中的“下载证书”(download certificate) 链接。
2. 在下一个对话框中，单击“打开”(Open) 打开文件。

3. 在“证书”(Certificate) 对话框中，单击“安装证书”(Install Certificate) 按钮以启动“证书导入向导”(Certificate Import Wizard)。
4. 单击“证书导入向导”(Certificate Import Wizard) 对话框中的“下一步”(Next) 以设置证书存储位置。
5. 选择“将所有证书存储在以下位置”(Place all certificates in the following store) 并单击“浏览”(Browse) 按钮。
6. 从“选择证书存储位置”(elect Certificate Store) 对话框中，选择“第三方根证书颁发机构”(Third-Party Root Certification Authorities)，然后单击“确定”(OK)。
7. 单击“下一步”(Next)，然后单击“完成”(Finish) 以完成“证书导入向导”(Certificate Import Wizard)。

将证书导入 Mozilla Firefox

1. 单击 Intro 页面中的“download certificate”链接。
2. 出现提示时，单击“确定”(OK) 以信任 S7-1200 Controller Family。

在旧版本的 Mozilla Firefox 上，单击“下载证书”(download certificate) 后，必须保存文件并执行向导：

1. 单击打开证书的对话框中的“保存文件”(Save file)。将出现“Downloads”对话框。
2. 在“下载”(Downloads) 对话框中，双击“MiniWebCA_Cer.crt”或所创建证书的名称。如果已多次尝试下载，则会显示多个副本。只需双击其中一个重复的证书条目即可。
3. 如果提示打开可执行文件，请单击“OK”。
4. 如果出现“Open File - Security Warning”对话框，请单击“Open”。将出现“Certificate”对话框。
5. 单击“Certificate”对话框中的“Install Certificate”按钮。
6. 按照“Certificate Import Wizard”对话框的提示导入证书，并使操作系统自动选择证书存储位置。
7. 如果出现“Security Warning”对话框，请单击“Yes”确认安装证书。

其它浏览器

导入和安装 Siemens 证书的过程与 Web 浏览器相同。

在 Web 浏览器的 Internet 选项中安装 Siemens 安全证书“S7-1200 Controller Family”之后，以 https:// ww.xx.yy.zz 访问 Web 服务器时不会再出现安全验证提示。

说明

CPU 重启后，安全证书保持不变。如果更改设备的 IP 地址，并且使用 Internet 浏览器或 Mozilla Firefox 以外的浏览器，则必须下载新证书。

12.7 标准 Web 页面

12.7.1 标准 Web 页面的布局

每个 S7-1200 标准 Web 页面的布局均相同，都具有导航链接和页面控件。无论您在 PC 还是在移动设备上查看页面，每个页面都具有相同的内容区域，但布局和导航控件会根据屏幕大小和设备分辨率发生变化。在标准 PC 或大型移动设备上，标准 Web 页面的布局如下：

① Web 服务器标题，其中包括显示 PLC 本地时间或 UTC 时间的选择器以及显示语言 (页 163)的选择器

② 登录或注销

③ 包含您正在查看的页面的名称的标准 Web 页面标题。本示例为 CPU 的 Identification 页面。有些标准 Web 页面（如模块信息页面）还会在此处显示导航路径，前提是您可以访问该类型的多个画面。

④ 刷新图标：对于具有自动更新功能的页面，可用来启用或禁用自动更新功能；对于不具有自动更新功能的页面，可以使页面用当前数据进行更新

⑤ 打印图标：准备并显示所显示页面提供的信息的可打印版本

⑥ 用来切换到其它页面的导航区

⑦ 正在查看的特定标准 Web 页面的内容区域。此处以“诊断”(Diagnostics) 页面为例。

说明

CP 模块标准 Web 页面

部分 CP 模块 (页 1033)提供和 S7-1200 CPU 标准 Web 页面外观和功能类似的 Web 页面。有关 CP 标准 Web 页面的说明请参见 CP 文档。

12.7.2 基本页面

Web 服务器提供旨在用于移动设备的基本页面。您可以通过设备 IP 地址和 URL 附加的 “basic”访问基本页面：http://ww.xx.yy.zz/basic 或 https://ww.xx.yy.zz/basic

基本页面和标准页面的外观类似，但仍有所不同。该页面会省略导航区域、登录区域和标题区域，但包含用于在 Web 页面中前进和后退的按钮。基本页面还包括 “主页”(Home page) 按钮，帮助导航到 “导航”(Navigation) 页面。还可以使用移动设备随附的导航控件进行导航。例如，基本 “诊断”(Diagnostics) 页面会按如下所示的进行垂直显示：

基本页面显示的最小分辨率为 240 x 240 像素。

请注意，本章中的标准 Web 页面图代表标准 PC Web 页面的外观。大多数标准 Web 页面具有相等的基本页面。

12.7.3 登录和用户权限

每个 PC 标准 Web 页面都会在导航窗格上方提供登录窗口。考虑到空间因素，基本 Web 页面提供单独的登录页面。S7-1200 为不同的用户登录提供不同的访问级别（权限）：

- 查询诊断
- 读取变量

- 写入变量
- 读取变量状态
- 写入变量状态
- 打开用户自定义 Web 页面
- 在用户自定义 Web 页面中进行写操作
- 读取文件
- 写入/删除文件
- 更改工作模式
- 闪烁 LED
- 执行固件更新
- 创建 PLC 备份
- 通过备份固件还原 PLC

在 CPU 的 STEP 7 设备组态的 Web 服务器用户管理属性中，组态用户角色、相应访问级别（权限）和密码 (页 1029)。

登录

STEP 7 提供的默认用户名称为“所有人”(Everybody)，没有密码。默认情况下，此用户没有任何附加权限，只能查看起始 (页 1041)和介绍 (页 1041)两个标准 Web 页面。不过，可以为“所有人”(Everybody) 用户和组态的其他用户授予附加权限：

⚠ 警告

访问 Web 服务器

授予“所有人”(Everybody) 用户相应权限，即可在没有密码的情况下登录 Web 服务器。未经授权访问 CPU 或将 PLC 变量更改为无效值可能会中断过程操作并可能导致死亡、严重人身伤害和/或财产损失。

由于具有足够权限的“所有人”(Everybody) 用户能够在没有密码的情况下执行工作模式更改、写入 PLC 数据以及进行固件更新，Siemens 建议遵照以下安全实践

- 仅使用 HTTPS 协议启用对 Web 服务器的访问。
- 使用可靠的密码对 Web 服务器用户 ID 进行密码保护 (页 1029)。强密码在长度上至少为十个字符，可以是字母、数字和特殊字符的组合，不能是可在字典上找到的词，并且不能是可从个人信息推断出的名字或标识符。保管好密码并经常更改密码。
- 不要扩展“所有人”(Everybody) 用户的默认最低权限。
- 对程序逻辑中的变量执行错误检查和范围检查，因为 Web 页面用户可将 PLC 变量更改为无效值。
- 如果您不在受保护的网络范围内，请使用安全的虚拟专用网络 (VPN) 连接到 S7-1200 PLC Web 服务器。

要执行特定操作（如更改控制器的操作模式或向存储器写入值以及更新 CPU 固件），必须具有所需权限。请注意，如果您已经将 CPU 的保护等级 (页 198)设置为“完全保护（无访问）”(Complete protection (no access))，则无论 Web 服务器用户权限设置如何，“所有人”(Everybody) 用户都不具备 Web 服务器的访问权限。

在 PC 或宽大的移动设备上显示时，登录框位于每个标准 Web 页面的左上角附近。

在显示基本页面的小型移动设备上，登录页面为单独页面。可从主页进行选择。

要登录，请按以下步骤操作：

1. 在“用户名”(Username) 字段中输入用户名称。
2. 在“密码”(Password) 字段中输入用户密码。

如果持续三十分钟没有操作，则会话过期。如果当前加载的页面不断刷新，则登录会话超时将重置，从而防止会话过期。

说明

如果在登录时遇到任何问题，请从“简介”(Introduction) 页面 (页 1041)下载 Siemens 安全证书 (页 1034)。随后便可成功登录。

注销

从 PC 或较宽的移动设备查看时，只需在任意页面中单击“注销”(Logout) 链接即可注销。

从基本页面开始，从主页导航至登录/注销页面，然后点击“注销”(Logout) 按钮。

退出后，您只能使用“Everybody”用户权限来访问和查看标准 Web 页面。每个标准 Web 页面描述都定义了该页面所需的权限。

说明

关闭 Web 服务器前注销

如果您已登录到 Web 服务器，请确保在关闭 Web 浏览器前先注销。Web 服务器最多支持 7 个并发登录。

12.7.4 简介

“简介”(Introduction) 页面是进入 S7-1200 标准 Web 页面的欢迎画面。

在该页面中单击“Enter”可访问 S7-1200 标准 Web 页面。 屏幕上方是有用的 Siemens Web 网站的链接以及下载 Siemens 安全证书 (页 1034)的链接。 您还可以选择在将来访问该 Web 服务器时跳过介绍页面。

12.7.5 Start

起始页面显示所连接 CPU 或 CP 的图示，并列出设备的常规信息以及将项目下载到 CPU 时使用的 TIA Portal 版本。对于 CPU，如果您以 “更改工作模式”(change operating mode) 权限 (页 1029)登录 (页 1037)，则可以使用按钮更改工作模式以及点亮 LED。

如果已在具有 S7-1200 CPU 的本地机架中组态和安装了已启用 Web 服务器的 CP 模块 (页 1033)，则可以在屏幕底部看见这些模块。将鼠标指针悬停在已启用 Web 服务器的 CP 模块上，单击即可访问标准 Web 网页。有关 CP 模块 Web 页面的信息，请参见 CP 模块文档。当您将鼠标悬停在 CP 模块上时，可以看到该模块的名称。

Web 服务器还显示本地机架中的任何其它 CM 和 CP 模块，但如果这些模块不包含 Web 页面，则无法单击它们。这些 CM 和 CP 模块的外观以浅灰色（亮度已降低）显示，表示这些模块只作显示，无法单击。

请注意，S7-1200 故障安全 CPU 会在此页面上显示与功能安全相关的额外数据。

12.7.6 诊断

“诊断”(Diagnostics) 页面显示了 CPU 的标识特征、专有技术保护的组态设置、装载存储器的使用情况、工作存储器和保持性存储器：

该页面包含三个选项卡：

- 标识：模块和设备的标识特征以及 STEP 7 的位置信息
- 程序保护：专有技术保护和 CPU 绑定的状态，可用于对备件进行规划以及对 STEP 7 进行组态设置，从而允许或阻止将内部装载存储器复制到外部装载存储器（SIMATIC 存储卡）。
- 存储器：装载存储器、工作存储器和保持存储器的使用

对于 F-CPU，还配有“故障 - 安全”(Fail-safe) 选项卡。

查看“标识”(Identification) 页面需要具备“查询诊断”(query diagnostics) 权限 (页 1029)。

“标识”(Identification) 选项卡

“程序保护”(Program protection) 选项卡

“程序保护”(Program protection) 选项卡包括以下信息：

- 专有技术保护 (页 201)：显示是否为 STEP 7 中的任何项目块组态了专有技术保护
- 绑定 (页 202)：显示是否已经将程序绑定了 CPU 或 SIMATIC 存储卡
- 复制程序到存储卡 (页 201)：显示是否启用了将程序从内部装载存储器复制到外部装载存储器（SIMATIC 存储卡）中的功能

“内存”(Memory) 选项卡

“故障安全”(Fail-safe) 选项卡

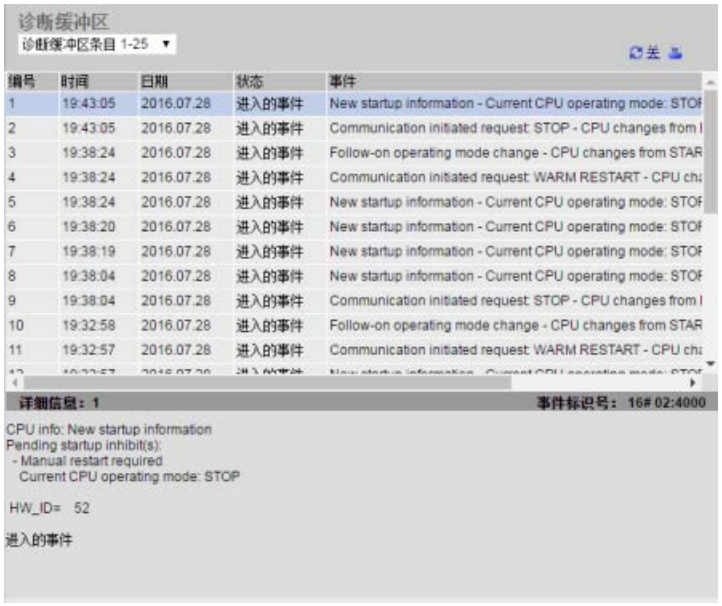
有关“诊断”(Diagnostics) 页面“故障安全”(Fail-safe) 选项卡的详细信息，请参见 S7-1200 功能安全手册 (https://support.industry.siemens.com/cs/cn/zh/view/104547552/en)。

12.7.7 Diagnostic Buffer

Diagnostic Buffer 页面会显示诊断事件。最新的事件是顶部的编号 1。最早的事件是编号 50。从左侧选择器中选择要显示的诊断缓冲区条目范围，可以是 1-25、1-26 或 1-50。从右侧选择器中选择是显示 UTC 时间还是显示 PLC 逻辑时间。事件发生时，页面顶部会显示包含时间和日期的诊断条目。

可以选择任何单独的诊断条目，以在页面底部显示有关该条目的详细信息。请注意，诊断缓冲区条目的显示语言取决于设备组态设置中的多语言支持 (页 168)。

“诊断缓冲区”(Diagnostic Buffer) 页面还包含按钮 ，该按钮用于将诊断缓冲区保存到 CSV 文件。默认情况下，Web 服务器将采用逗号分隔格式的文件保存到“下载”(Downloads) 文件夹中的 ASLog.csv。该文件包含保存操作时的完整诊断缓冲区。用户可以选择多次保存该文件，并保留多个文件。可以使用 Microsoft Excel 或任何文本编辑器打开诊断缓冲区文件。



查看“诊断缓冲区”(Diagnostic Buffer) 页面需要“查询诊断”权限 (页 1029)。

12.7.8 模块信息

Module Information 页面提供有关本地机架中所有模块的信息。屏幕顶部显示了 STEP 7 中基于设备组态的模块概述，底部显示了基于对应连接模块的选定模块的状态、标识和固件信息。模块信息页面还提供执行固件更新的功能。

查看“模块信息”(Module Information) 页面需要“查询诊断”权限 (页 1029)。

模块信息：“状态”(Status) 选项卡

Module Information 页面底部的 Status 选项卡显示顶部所选模块的当前状态的说明。如果本部分为空，则模块不具备待决诊断状态。

模块的状态图标

对于每个模块，顶部状态列显示一个图标指示模块的状态：

图标	含义
	无故障
	禁用
	需要维护
	要求维护
	错误
	CPU 无法访问模块或设备（除 CPU 外的设备）
	CPU 已和设备建立连接，但模块状态未知（除 CPU 外的设备）
	由于子模块阻塞了 I/O 通道，无法获得输入数据和输出数据（除 CPU 外的设备）

深入展开

可以选择顶部的链接来深入展开特定模块的模块信息。具有子模块的模块包含每个子模块的链接。显示的信息类型会根据所选模块的不同而异。例如，模块信息对话框最初会显示 S7-1200 站的名称、状态指示灯和注释。如果深入展开至 CPU，模块信息将显示 CPU 型号提供的数字量和模拟量输入输出的名称、I/O 地址信息、状态指示灯、插槽号和注释。

插槽	状态	名称		订货号	I 地址:	Q 地址:	注释
1.1	✔	DI14/DQ10_1	详细信息	6ES7 214-1AG40-0XB0	0	0	
1.2	✔	AI2_1	详细信息	6ES7 214-1AG40-0XB0	64	---	
1.16	✔	HSC_1	详细信息	6ES7 214-1AG40-0XB0	1000	---	
1.17	✔	HSC_2	详细信息	6ES7 214-1AG40-0XB0	1004	---	
1.18	✔	HSC_3	详细信息	6ES7 214-1AG40-0XB0	1008	---	
1.19	✔	HSC_4	详细信息	6ES7 214-1AG40-0XB0	1012	---	
1.20	✔	HSC_5	详细信息	6ES7 214-1AG40-0XB0	1016	---	
1.21	✔	HSC_6	详细信息	6ES7 214-1AG40-0XB0	1020	---	
1.32	✔	Pulse_1	详细信息	6ES7 214-1AG40-0XB0	---	1000	

在深入展开的同时，模块信息页面会显示访问的路径。可以单击该路径中的任意链接返回到更高级别。

模块信息：“标识”(Identification) 选项卡

“标识”(identification) 选项卡显示选定模块的标识和维护 (I&M) 信息。

请注意，如果单击上方的 F-I/O 模块，则底部会显示“安全”(Safety) 选项卡。在此选项卡中，会显示 S7-1200 功能安全手册 (https://support.industry.siemens.com/cs/cn/zh/view/104547552/en)中所述的与所选模块相关的具体数据。

模块信息：“固件”(Firmware) 选项卡

模块信息页面的“固件”(Firmware) 选项卡显示所选模块固件的相关信息。如果具有“执行固件更新”权限 (页 1029)，还可以执行本地机架中支持固件更新的 CPU 或其它模块或者 PROFINET I/O 模块的固件更新。对于远程模块，您可以查看固件信息但无法执行固件更新。

说明

要更新 CPU 固件，只能更新 3.0 及更高版本的 S7-1200 CPU。

执行固件更新

CPU 必须处于 STOP 模式下才能执行固件更新。当 CPU 处于 STOP 模式时，单击“浏览”(Browse) 按钮导航至并选择固件文件。Siemens 工业在线支持网站 (https://support.industry.siemens.com/cs/cn/zh)上提供固件更新程序。

在更新期间，该页面会显示消息，表明更新正在进行。更新完成后，页面将显示已更新固件的订货号和版本号。如果您更新过 CPU 或信号板的固件，Web 服务器将重新启动 CPU。

还可以通过以下任一方法来执行固件更新：

- 使用 STEP 7 的在线和诊断工具 (页 1390)
- 使用 SIMATIC 存储卡 (页 143)
- 使用 SIMATIC 自动化工具 (https://support.industry.siemens.com/cs/cn/zh/view/98161300/en)

说明

通过 Web 服务器执行固件更新时可能遇到的问题

如果在通过 Web 服务器进行固件更新的过程中通信中断，则 Web 浏览器会显示一条消息，询问您希望留在当前页面还是离开。为避免潜在的问题，请选择留在当前页面的选项。

如果您在通过 Web 服务器执行固件更新过程中关闭 Web 浏览器，您将无法将 CPU 的工作模式更改为 RUN 模式。如果发生这种情况，您必须对 CPU 循环上电才能将 CPU 更改为 RUN 模式。

12.7.9 Communication

通信页面显示连接的 CPU 的参数、通信统计、资源和连接信息。

查看“通信”(Communication) 页面需要具备“查询诊断”(query diagnostics) 权限。

“参数”(Parameter) 选项卡

“参数”(Parameter) 选项卡显示 CPU 的 MAC 地址、CPU 的 IP 地址和 IP 设置以及物理属性：

“统计”(Statistics) 选项卡

“统计”(Statistics) 选项卡显示了发送和接收的通信统计：

“连接资源”(Connection resources) 选项卡

“资源”(Resources) 选项卡显示了连接资源总数以及如何为不同类型通信分配连接资源的相关信息：

“连接状态”(Connection status) 选项卡

“连接”(Connections) 选项卡显示 CPU 的连接和选定连接的连接详情。

12.7.10 变量状态

“变量状态”页面允许您查看任何 CPU 中 I/O 或存储器数据。您可以输入直接地址（如 %I0.0）、PLC 变量名或指定数据块的变量。对于数据块变量，应使用双引号将数据块名称括起来。可以为每个监视值选择数据的显示格式。可以继续输入和指定值，只要所需值的数量不超过页面的限制。将自动显示监视值。可在任何时候单击“刷新”(Refresh) 按钮刷新所有监视值。如果已启用了 STEP 7 中的自动更新 (页 1028)，可以单击页面右上角的“关闭”(Off) 图标禁用该功能。如果已禁用自动更新，则可单击“开启”(On) 重新启用自动更新。

查看“变量状态”页面需要具备“读取变量状态”权限。

如果您以具有“写入变量状态”权限 (页 1037)的用户身份登录，则还可以修改数据值。在相应的“修改值”(Modify Value) 字段中输入任何希望设置的值。单击值旁边的“Go”按钮将该值写入 CPU。还可以输入多个值，然后单击“应用”(Apply) 将所有值写入 CPU 中。仅当您具有“写入变量状态”权限时，才会显示用于修改的按钮和列标签。

如果离开“变量状态”页面然后返回，则“变量状态”页面不会保留您的输入内容。可以为页面加书签，然后返回该书签，这样便可看到相同的条目。如果不为页面加书签，则必须重新输入变量。

对于经常监视或修改的值，考虑使用监视表 (页 1054)进行替换。

说明

变量使用标准“变量状态”页面时，请注意以下问题：

- 用单引号将所有字符串修改括起来。
- “变量状态”可以监视和修改包含以下字符的变量：&、<、(、+、 、(comma)、.、[,]、$ 或 %，假定您用双引号将变量名引起来，例如“Clock_2.5Hz”。
- 要仅监视或修改 DTL 变量的一个字段，可在地址中包括此字段，例如，"Data_block_1".DTL_tag.Year。根据 DTL 指定字段的数据类型输入整数值作为修改值。例如，Year 字段的数据类型为 UInt。
- 每页可输入的最大变量数为 50。
- 如果某个变量名称因包含特殊字符而被拒绝作为“变量状态”页面上的条目，则可用双引号将该变量名括起来。大多数情况下，该页面随后便能识别此变量名称。

另请参见

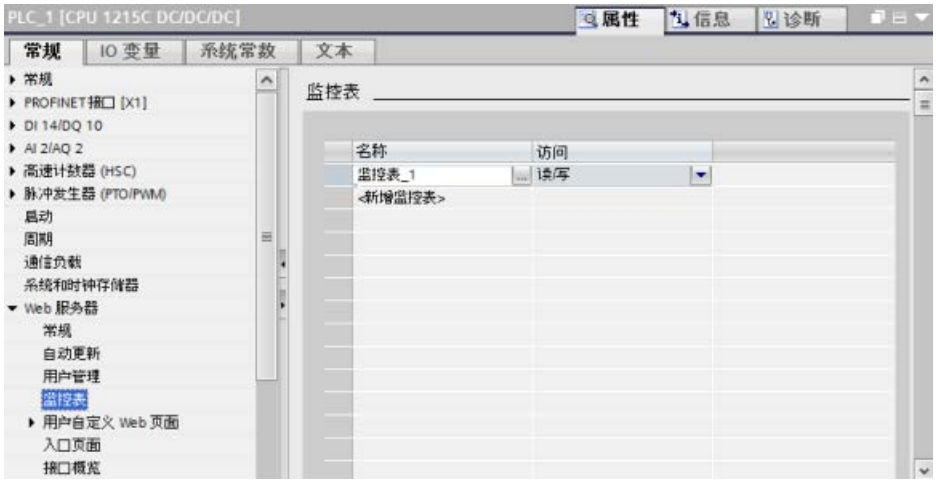
变量名称和值的输入规则 (页 1110)

12.7.11 监控表

Web 服务器允许您访问已在 STEP 7 中组态并下载到 CPU 中的监控表。条目数不高于 50 的监控表可在 Web 服务器中实现最佳性能。

用于为 Web 服务器选择监控表的 STEP 7 组态

在 STEP 7 中，可通过 CPU 的设备组态添加要在 Web 服务器中显示的监控表。对于从现有监控表列表中选择的每个监控表，您可以为其选择读或读/写权限。将监控表下载到 CPU 后，如果选择的是读权限，则只能查看监控表；如果选择的是读/写权限，则可以查看并修改监控表变量。



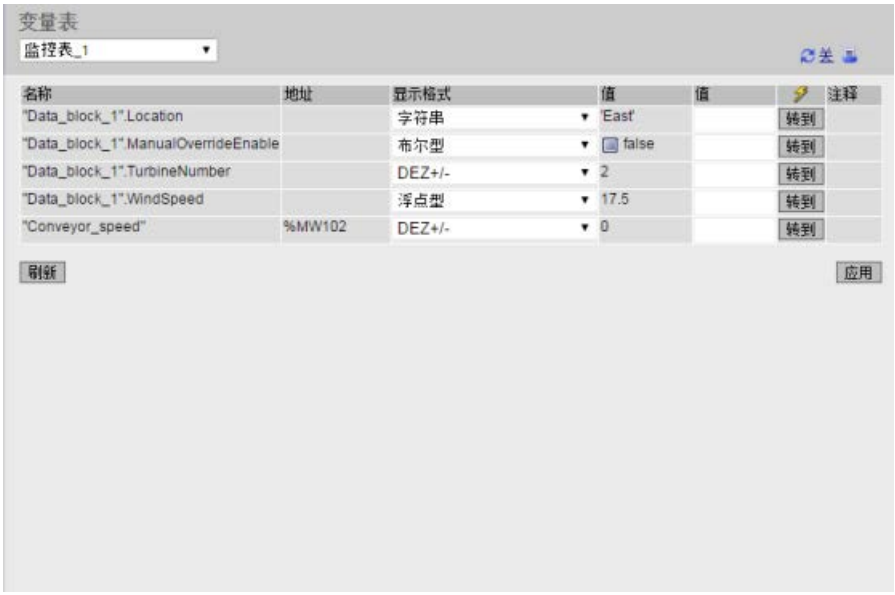
在设备组态的 Web 服务器部分完成监控表组态后，将硬件组态下载到 CPU 中。

通过 Web 服务器查看监控表

在 Web 服务器中，如果您拥有“读变量”权限 (页 1029)，则可以通过从导航菜单中选择“监控表”来访问已组态并下载到 CPU 中的监控表。如果您下载了多个监控表，则可以从下拉列表中选择想要显示的监控表。Web 服务器会显示您在 STEP 7 中创建的监控表，并按照相应的显示格式显示表中的当前值。您可以选择更改显示格式，但返回至监控表页面时，Web 服务器会默认采用 STEP 7 监控表中的显示格式。

通过 Web 服务器修改监控表变量

如果您下载了一个“读/写”访问级别的监控表，则当您登录 Web 服务器且拥有“写变量”权限 (页 1029)时，可以像在 STEP 7 中那样修改监控表中的变量值。您可以只修改单个变量值（修改单个变量值，然后单击“确定”(Go)），也可以同时修改多个变量值（输入多个值，然后单击“应用”(Apply)）。



说明

通过监控表修改变量的优势

如要通过监控表修改 CPU 中的变量和数据块变量，必须在 STEP 7 设备组态的 Web 服务器属性中组态监控表，并且必须为其选择读/写访问权限。这样一来，便可以对拥有“写变量”权限的用户施加限制，只允许他们修改已组态的 Web 服务器监控表中的变量。

另一方面，“变量状态” (页 1052)(Tag Status) 页面允许任何拥有“写变量状态”权限的用户修改 CPU 中的任意变量或数据块变量。

组态 Web 服务器用户管理权限 (页 1029)时务必小心，以便为 PLC 数据访问提供安全保护。

另请参见

变量名称和值的输入规则 (页 1110)

12.7.12 在线备份

“在线备份”(Online backup) 标准 Web 页面允许您为在线 PLC 创建 STEP 7 项目备份，以及恢复之前创建的 PLC 备份。创建或恢复备份之前，请将 PLC 置于 STOP 模式并停止与 PLC 之间的全部通信（例如，HMI 访问和 Web 服务器访问）。如果 CPU 未处于 STOP 模式，则备份和恢复功能会在继续执行之前提示您将 CPU 置于 STOP 模式。

如果您通过其中一个支持 Web 的 CP 模块访问 “在线备份”(Online backup) 页面，则可以执行备份操作，但不能执行恢复操作。

说明

您也可以通过 STEP 7 执行备份和恢复操作 (页 1420)。有关可备份和恢复的数据的完整说明，请参见这些主题。SIMATIC 自动化工具 (SAT) 也提供备份和恢复功能。

通过 Web 服务器备份文件时，PC 或设备会将备份文件保存到默认文件夹中，以供下载。通过 STEP 7 备份文件时，STEP 7 会将文件保存到 STEP 7 项目中。无法通过 Web 服务器恢复 STEP 7 备份文件，也不能通过 STEP 7 恢复 Web 服务器备份文件。但是，可以将 STEP 7 备份文件直接保存到 PC 或设备的下载文件夹中。这样一来，便可以通过 Web 服务器恢复这些文件。

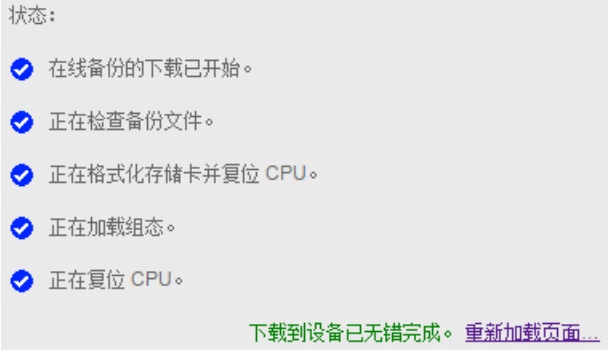
备份 PLC

在该页面的“备份 PLC”(Backup PLC) 部分，单击“创建在线备份”(Create online backup) 按钮为 PLC 中当前存储的项目创建备份。该功能需要“备份 CPU”用户权限 (页 1029)。如果必须将处于 RUN 模式的 CPU 切换到 STOP 模式，则还需要“更改工作模式”权限。PC 或设备会将备份文件存储在默认位置，以供下载。根据浏览器和设备设置的不同，可能会提示您保存文件。

恢复 PLC

在该页面的“恢复 PLC”(Restore PLC) 部分，输入 Web 服务器用户密码，然后单击“浏览” (Browse) 或“选择文件”(Choose File) 按钮（具体取决于浏览器）选择之前保存的备份文件。单击“加载在线备份”(Load online backup) 按钮并确认提示以将该文件加载到连接的 PLC 中。该页面需要“恢复 CPU”用户权限 (页 1029)。如果必须将处于 RUN 模式的 CPU 切换到 STOP 模式，则还需要“更改工作模式”权限。

在恢复操作的执行过程中，会显示一系列过程消息，用户必须重新进入用户登录界面输入密码。每个过程步骤顺利完成后，都会显示以下完成指示以及用于重载页面的链接：



警告

恢复内容未知的备份

如果恢复内容未知的备份，则在发生故障或程序错误时，可能导致重大财产损失或人员严重受伤。

此外，如果恢复的备份未在 CPU 的设备组态中启用 Web 服务器，则无法通过 Web 服务器访问 CPU。

确保备份包含内容已知的组态。

说明

恢复 CPU IP 地址不一致的备份

如果试图恢复的备份中的 CPU IP 地址与当前 CPU 的 IP 地址不一致，Web 服务器将无法显示恢复操作完成的消息。当“复位 CPU”消息显示超过五分钟时，请输入与备份文件中的地址对应的新 IP 地址。此时，CPU 的地址变为这一新 IP 地址，您可以继续执行 Web 服务器访问。

12.7.13 数据日志页面

通过“数据日志”(Data Logs) 页面可以与 STEP 7 程序中的数据日志交互。用户可以：

- 查看 PLC 上所有数据日志的列表
- 将数据日志从 PLC 下载到计算机
- 从 PLC 中删除数据日志
- 成功检索并清除数据日志

数据日志以不区分大小写的字母升序显示。数据日志列表以 50 个数据日志为增量分页。

说明

数据日志管理

在文件系统中，可保留不超过 1000 个数据日志。超过此数目时，Web 服务器就没有用于显示数据日志的足够空间。

如果发现“数据日志”Web 页面无法显示数据日志，则必须将 CPU 置于 STOP 模式，以便显示并删除数据日志。

管理您的数据日志以确保仅保留需要维护的数目，且不会超过 1000 个数据日志。

在 Excel 中使用数据日志

数据日志文件采用 USA/UK 的逗号分隔值格式 (CSV)。要在非 USA/UK 系统上的 Excel 中打开该文件，必须采用特定设置将其导入 Excel 中。

活动状态

当与文件关联的 CPU 上存在数据日志控制块时，“数据日志”(Data Logs) 页面上的“活动”(Active) 列会显示“是”(Yes)，没有数据日志控制块时则显示“否”(No)。即使 STEP 7 程序当前未与文件交互，也可以将活动状态设置为“是”(Yes)。

如果 STEP 7 程序打开了一个数据日志或正在写入数据日志，则 Web 服务器将无法删除、下载或检索并清除数据日志文件。此外，当 Web 服务器通过下载操作或检索并清除操作执行数据日志下载时，将无法执行其它数据日志操作，直至下载完成或取消。Web 服务器显示“应用程序忙”(Application Busy) 错误消息。

下载数据日志文件

单击文件名称即可下载相关的数据日志文件。如果文件不存在或另一个下载操作正在进行，Web 服务器将显示一条错误消息。该错误消息将保留在页面上，直至启动重新加载“数据日志”(Data Logs) 页面的操作。出现以下操作时，Web 服务器将重新加载“数据日志”(Data Logs) 页面：

- 刷新“数据日志”(Data Logs) 页面或导航离开并返回“数据日志”(Data Logs) 页面
- 更改当前的数据日志分页选择
- 成功删除数据日志
- 成功检索并清除数据日志

说明

数据日志错误消息

Web 服务器的自动刷新功能不会从页面中删除错误消息。

在以下情况下，“数据日志”(Data Logs) 页面上将生成错误消息：

操作	故障条件	错误消息
下载 检索/清除	• 文件不存在 • 文件名无效 • HTTP 请求方法无效 • SMC 受到写保护（仅检索/清除）	下载文件时出错
删除	• HTTP 请求方法无效 • 文件名无效 • 文件不存在	删除文件时出错
删除	• SMC 受到写保护	删除文件时出错：存储卡受到写保护

操作	故障条件	错误消息
下载 删除 检索/清除	• 没有所需权限 • HTTP 请求方法不是 POST 或 GET • URL 中的 ACTION 参数无效	禁止文件操作
下载 删除 检索/清除	• HTTP 引用源字段无效或缺失	禁止文件操作：无引用源
下载 删除 检索/清除	• 用户程序保持数据日志文件打开 • 当前下载数据日志	应用程序繁忙
下载 删除 检索/清除	• 意外的内部 PLC 错误	内部错误

删除数据日志

针对特定数据日志单击“删除”(Delete) 列中的 图标可删除数据日志。要删除数据日志文件，请从“删除”(Delete) 对话框中确认删除操作。

检索并清除数据日志

要打开数据日志并删除所有条目，请单击检索并清除 图标。要检索数据日志文件并清空内容，请从“检索并清除”(Retrieve and Clear) 对话框中确认操作。

确认操作后，可通过 Web 服务器下载数据日志文件的内容。“保存文件”(Save File) 对话框可用于选择是否保存数据日志。选择后，Web 服务器会清空数据日志文件的内容，但不删除文件。无法从“保存文件”(Save File) 对话框中取消数据日志的清除。只能从“检索并清除”(Retrieve and Clear) 对话框的初始确认中取消操作。

如果在检索并清除过程中发生错误，则不会从数据日志中删除任何数据，并且会显示一条错误消息。如果数据日志的“活动”(Active) 列中列出“否”(No)，则检索并清除操作不会清除内容，因为 STEP 7 程序没有对该文件的引用。之后，必须手动删除文件以删除数据。

参见

Web 服务器 (页 1025)

12.7.14 用户文件

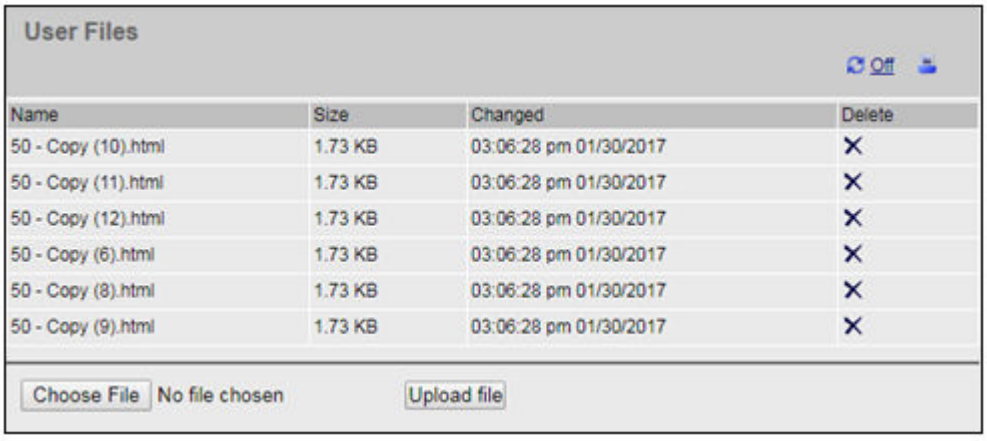
处理用户文件

可以使用“用户文件”(User Files) 页面访问 CPU 内部装载存储器或存储卡（外部装载存储器）上的文件。

对用户文件的访问权限类型取决于所具有的用户权限。任何具有“读取文件”权限的用户都可以通过“用户文件”(User Files) 页面查看文件和文件夹。如果具有“写入/删除文件”权限，则还可以：

- 查看 PLC 上所有用户文件的列表
- 将用户文件从 PLC 下载到计算机
- 将用户文件从计算机上传到 PLC
- 从 PLC 中删除用户文件

要查看用户文件，请从主导航页面中单击“用户文件”(User Files)



文件列表包括文件的当前大小和最后修改日期。文件列表是从 SMC 根目录中的“用户文件”(User Files) 目录填充的。

说明

用户文件管理

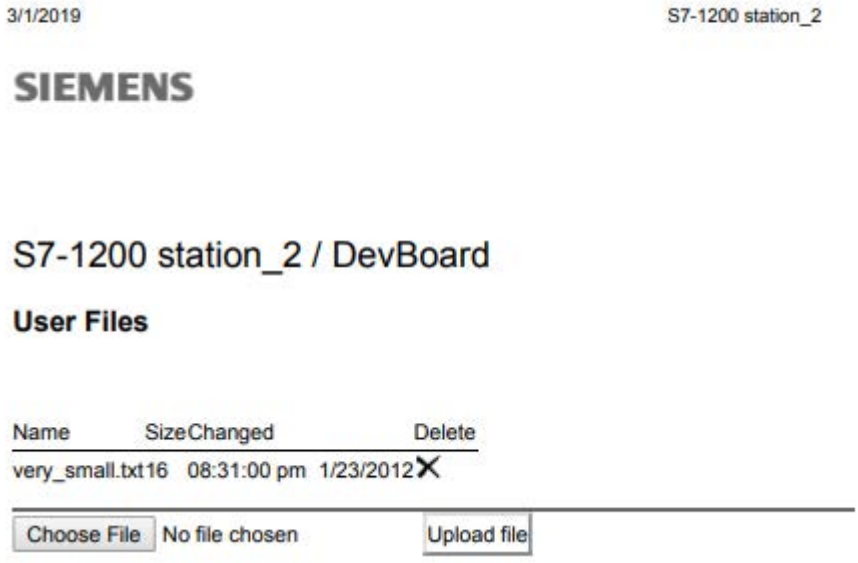
在 SMC 中，保留不超过 1000 个用户文件。如果存在更多文件，则 Web 服务器仅允许查看前 1,000 个文件。文件以不区分大小写的字母升序显示。

用户文件列表分页

用户文件列表以 50 个条目为增量分页。可通过下拉框选择要查看的条目范围。

打印用户文件列表

可以使用 PLC Web 页面上的 “打印”(Print) 图标打印 PLC 上的用户文件列表：



用户文件错误

如果操作未能成功完成，“用户文件”(User Files) 页面将显示一条错误消息：

该错误消息将保留在页面上，直至启动重新加载“用户文件”(User Files) 页面的操作。出现以下操作时，Web 服务器将重新加载“用户文件”(User Files) 页面：

- 刷新“用户文件”(User Files) 页面或导航离开并返回“用户文件”(User Files) 页面
- 更改当前用户文件分页选择
- 成功删除用户文件
- 成功上传用户文件

说明

用户文件错误消息

Web 服务器的自动刷新功能不会从“用户文件”(User Files) 页面中删除错误消息。

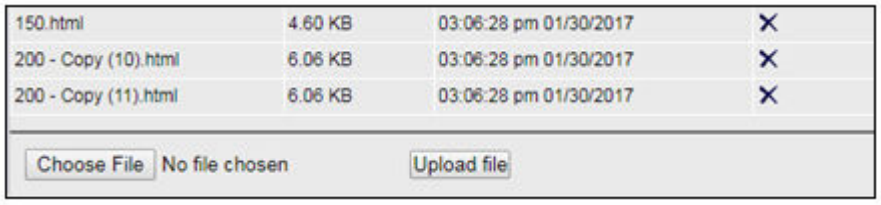
在以下情况中，“用户文件”(User Files) 页面上将生成错误消息：

操作	故障条件	错误消息
下载用户文件	• 文件不存在 • 内部 PLC 错误	下载文件时出错
下载用户文件 上传用户文件 删除用户文件	• 权限不足，无法执行请求的操作 • HTTP 请求方法或操作参数无效	禁止文件操作
下载用户文件 上传用户文件 删除用户文件	• HTTP 引用源字段无效或缺失	禁止文件操作 - 没有引用源
删除用户文件	• 存储卡受到写保护	禁止文件操作

操作	故障条件	错误消息
下载 删除 检索/清除	• 文件不存在 • 内部 PLC 错误	删除文件时出错 - 存储卡受到写保护
上传用户文件	• 文件名丢失或无效 • 内部 PLC 错误	上传文件时出错
上传用户文件	• PLC 中已存在该文件名	上传文件时出错 - 名称已存在
上传用户文件	• SMC 已满	上传文件时出错 - 存储卡已满
上传用户文件	• 文件名无效	上传文件时出错 - 文件名中的字符无效
上传用户文件	• 文件过大，无法存储在 SM 文件系统中	上传文件时出错 - 文件过大
上传用户文件	• SMC 受到写保护	上传文件时出错 - 存储卡受到写保护

上传用户文件

可以从用户文件上传表单中选择一个文件来上传用户文件：



要上传用户文件，请选择文件，然后单击 “上传文件”(Upload file) 以开始将文件从计算机传输到 PLC。在开始传输之前，必须满足以下前提条件：

要求	消息
必须选择一个文件	请输入一个文件名
文件名必须有效	允许使用字符 a 到 z、0 到 9、(){}[]$!=~（空格）
文件的大小必须小于 2 GB（文件系统限制）。	上传文件时出错 - 文件过大

参见

Web 服务器 (页 1025)

12.7.15 数据日志 UserFiles API

数据日志 UserFiles API

存在可用于 S7-1200 用户文件和数据日志的数据日志用户文件 API 功能。有关详细信息，请参见 *S7-1500、ET200SP、ET200pro Web 服务器使用手册*。

12.7.16 文件浏览器

可以使用“文件浏览器”(File Browser) 页面访问 CPU 内部装载存储器或存储卡（外部装载存储器）上的文件。文件浏览器页面最初显示装载存储器的根文件夹（其中包含“Recipes”文件夹），但如果使用存储卡，还会显示可能已创建的其它文件夹。

对文件和文件夹具有的文件访问类型取决于您的用户权限 (页 1029)。任何具有“读取文件”权限的用户都可以通过文件浏览器查看文件和文件夹。无论用户的登录权限如何，都不能删除 Recipes 文件夹，但是如果已在存储卡上建立了自定义文件夹，则可在以具有“写入/删除文件”权限的用户身份登录时删除这些文件夹。

单击文件夹可访问该文件夹中的各个文件。

配方文件

配方文件夹会显示装载存储器中存在的任何配方文件。配方文件也采用 CSV 格式，您可在 Microsoft Excel 或其它程序中将其打开。必须具有修改权限才能删除、修改和保存、重命名或上传配方文件。

上传文件和自动页面刷新

开始上传文件后，只要停留在文件浏览器 Web 页面上，上传操作便会继续进行。如果启用了自动更新以每 10 秒刷新一次 Web 服务器页面，则只要页面刷新，您就能看到文件上传操作的递增进度。例如，如果上传一个 2 MB 的文件，那么在文件上传期间，您将看到文件大小从 2500 到 5000、10000、15000 再到 20000（以字节为单位）这样的变化过程。

如果在上载完成之前离开文件浏览器页面，Web 服务器将删除不完整的文件。

更多信息

说明

文件名称约定

为使 Web 服务器能够使用数据日志和配方文件，文件名必须使用 ASCII 字符集中的字符，字符 \ / : * ? " < > | 及空格字符除外。

如果文件不符合此命名约定，则 Web 服务器可能在进行文件加载、删除或重命名等操作时出现错误。在这种情况下，可能需要使用读卡器和 Windows 文件管理器来重命名位于外部装载存储器的文件。

有关使用数据日志指令进行编程以及导入 (页 508)和导出 (页 506)配方的信息，请参见“配方和数据日志 (页 501)”一章。

参见

将 CSV 格式的数据日志导入非 USA/UK 版本的 Microsoft Excel 中 (页 1110)

12.8 用户定义的 Web 页面

S7-1200 Web 服务器还提供了一些方法，供您创建可融入 PLC 数据的应用特定的 HTML 页面。

⚠ 警告

通过用户定义的 Web 页面对 CPU 进行未经授权的访问

通过用户定义的 Web 页面对 CPU 进行未经授权的访问可能会中断过程操作，从而导致死亡、严重人身伤害和/或财产损失。

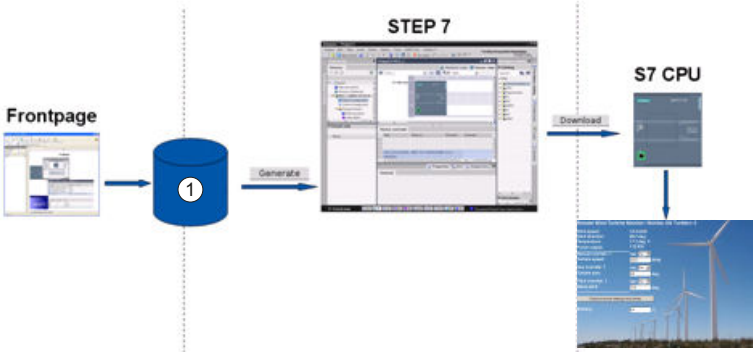
用户定义的 Web 页面中的不安全代码会引入跨站脚本 (XSS)、代码注入等安全漏洞。

按照“操作准则”中介绍的安全方式安装 S7-1200 CPU 可防止未经授权的访问，可在工业安全网站 (http://www.siemens.com/industrialsecurity)上找到这一准则。

可以选择相应的 HTML 编辑器来创建用户定义的 Web 页面，然后从可通过标准 Web 页面菜单访问的位置将这些页面下载到 CPU。 该过程涉及到以下几项任务：

- 使用 HTML 编辑器（如 Microsoft Frontpage）创建 HTML 页面。 (页 1067)
- 将 AWP 命令包含在 HTML 代码的 HTML 注释中 (页 1068)：AWP 命令是 Siemens 提供用于访问 CPU 信息的固定命令集。
- 将 STEP 7 组态为读取和处理 HTML 页面 (页 1083)
- 基于 HTML 页面生成块 (页 1083)
- 对 STEP 7 进行编程设计，以控制 HTML 页面的使用 (页 1085)
- 编译程序块并将其下载到 CPU (页 1087)
- 通过 PC 访问用户定义的 Web 页面 (页 1088)

该过程的图示如下：



① 具有嵌入式 AWP 命令的 HTML 文件

参见

Web 服务器 (页 1025)

12.8.1 创建 HTML 页面

用户可以使用选择的软件包来创建自己的 HTML 页面，以便与 Web 服务器一起使用。确保 HTML 代码符合 W3C（万维网联盟）的 HTML 标准。STEP 7 不会对 HTML 语法进行任何验证。

可使用能够以所见即所得或设计版式模式进行设计的软件包，但是必须能够在纯 HTML 表单中编辑 HTML 代码。大部分 Web 编写工具可以提供这种类型的编辑；否则，您始终可以

使用简单文本编辑器来编辑 HTML 代码。将以下代码行包含在 HTML 页面中，以将页面的字符集设置为 UTF-8：

```
<meta http-equiv="content-type" content="text/html;
charset=utf-8">
```

还要确保以 UTF-8 字符编码的格式保存编辑器中的文件。

使用 STEP 7 将 HTML 页面中的所有内容编译到 STEP 7 数据块中。这些数据块由一个管理 Web 页面显示的控制数据块、一个或多个包含已编译 Web 页面的片段数据块组成。请注意，如果有大量的 HTML 网页，尤其是那些具有很多图像的页面，它们的片段 DB 需要非常大的装载存储器空间 (页 1088)。如果 CPU 的内部装载存储器不足以容纳用户定义的 Web 页面，请使用存储卡 (页 134)来提供外部装载存储器。

要编写 HTML 代码程序以使用 S7-1200 中的数据，应将 AWP 命令 (页 1068)以 HTML 注释的形式包含在代码中。完成之后，将 HTML 页面保存到 PC，并记下保存这些页面的文件夹路径。

说明

包含 AWP 命令的 HTML 文件的文件大小限制为 64 KB。必须保持文件大小低于此限制，以便 STEP 7 能够成功编译页面。

刷新用户定义的 Web 页面

用户定义的 Web 页面不会自动刷新。可以根据需要选择是否编写用来刷新页面的 HTML 程序。对于显示 PLC 数据的页面，定期刷新可使数据保持最新。对于用作数据输入格式的 HTML 页面，刷新可能会干扰用户输入数据。如果希望整个页面自动进行刷新，可将以下命令行添加到 HTML 头文件，其中，“10”表示两次刷新间隔的时间（以秒为单位）：

```
<meta http-equiv="Refresh" content="10">
```

还可以使用 JavaScript 或其它 HTML 方法来控制页面或数据的刷新。相关信息，请参考 HTML 和 JavaScript 文档。

12.8.2 S7-1200 Web 服务器支持的 AWP 命令

S7-1200 Web 服务器提供了以 HTML 注释形式嵌入用户定义的 Web 页面中的 AWP 命令，这些命令具有以下用途：

- 读取变量 (页 1070)
- 写入变量 (页 1071)
- 读取特殊变量 (页 1073)

- 写入特殊变量 (页 1075)
- 定义枚举类型 (页 1077)
- 为枚举类型分配变量 (页 1078)
- 创建片段数据块 (页 1079)

一般语法

除读取变量的命令之外，AWP 命令的语法如下：

```
<!-- AWP_ <command name and parameters> -->
```

AWP 命令与典型的 HTML 表单命令一起使用时，可将变量写入 CPU。

接下来的各页面在介绍 AWP 命令时采用如下惯例：

- 方括号 [ ] 中包含的项为可选项。
- 尖括号 < > 中包含的项是要指定的参数值。
- 引号是命令的文字部分。 它们必须按所示的形式出现。
- 根据具体用法，变量或数据块名称中的特殊字符必须进行转义或用引号括号来 (页 1081)。

使用文本编辑器或 HTML 编辑模式可将 AWP 命令插入页面中。

说明

AWP 命令所需的语法

AWP 命令公式中“<!--”之后的空格和“-->”之前的空格，对于命令的正确编译至关重要。 疏漏空格字符可能导致编译器无法生成正确代码。 这种情况下，编译器不会显示错误。

AWP 命令汇总

接下来的各主题将详细介绍每个 AWP 命令的用法，但此处先对这些命令进行简单汇总：

读取变量

```
:=<Varname>:
```

写入变量

```
<!-- AWP_In_Variable Name='<Varname1>' [Use='<Varname2>'] ... -->
```

该 AWP 命令只是声明 Name 子句中的变量可写入。 HTML 代码将按 HTML 表单中 <input>、<select> 或其它 HTML 语句中的名称写入变量。

读取特殊变量

```
<!-- AWP_Out_Variable Name='<Type>:<Name>' [Use='<Varname>'] -->
```

写入特殊变量

```
<!-- AWP_In_Variable Name='<Type>:<Name>' [Use='<Varname>']-->
```

定义枚举类型

```
<!--
 AWP_Enum_Def Name='<Enum type name>' Values='<Value>, <Value>,..
. ' -->
```

引用枚举类型

```
<!-- AWP_In_Variable Name='<Varname>' Enum="<Enum 类型名称>" -->
<!-- AWP_Out_Variable Name='<Varname>' Enum="<Enum 类型名称>" -->
```

创建片段

```
<!-- AWP_Start_Fragment Name='<Name>' [Type=<Type>][ID=<id>] -->
```

导入片段

```
<!-- AWP_Import_Fragment Name='<名称>' -->
```

12.8.2.1 读取变量

用户定义的 Web 页面可读取 CPU 中的变量（PLC 变量）和数据块变量，前提是已将变量组态为可通过 HMI 访问。

语法

```
:=<Varname>:
```

参数

<Varname>	要读取的变量，可以是 STEP 7 程序中的 PLC 变量名称、数据块变量、I/O 或可寻址存储器。对于存储器、I/O 地址或别名 (页 1081)，请勿使用引号将变量名称括起来。对于 PLC 变量，请使用双引号将变量名称括起来。对于数据块变量，只用双引号将块名称括起来。变量名称位于引号外。请注意，应使用数据块名称，而不是数据块编号。使用数组元素语法引用数组元素。

示例

```
:="Conveyor_speed":
:="My_Data_Block".flag1:
:=I0.0:
:=MW100:
:="My_Data_Block".Array_Dim1[0]:
:="My_Data_Block".Array_Dim2[0,0]:
```

读取具有别名的变量的示例

```
<!-- AWP_Out_Variable Name='flag1' Use='"My_Data_Block".flag1' --
>
:=flag1:
```

说明

对变量引用使用别名 (页 1076)主题中介绍了如何定义 PLC 变量和数据块变量的别名。

如果变量名称或数据块名称包含特殊字符，则必须使用附加引号或转义字符，如处理包含特殊字符的变量名称 (页 1081)主题中所述。

12.8.2.2 写入变量

用户定义的页面可以将数据写入 CPU。这是通过 AWP 命令识别 CPU 中可从 HTML 页面写入的变量来实现的。该变量必须通过 PLC 变量名称或块变量名称指定。可以在一条语句中声明多个变量名称。要将数据写入 CPU，应使用标准 HTTP POST 命令。

典型用法是：使用与可写入的 CPU 变量对应的文本输入域或选择列表选项来在 HTML 页面中设计表单。与所有用户定义的页面相同，之后将通过 STEP 7 生成块，从而在 STEP 7 程序中包含这些块。具有修改变量权限的用户随后访问该页面并在输入字段中输入数据或从选择列表中选择选项时，Web 服务器会将输入转换为变量的相应数据类型，并将该值写入 CPU 中的变量。请注意，HTML 输入域和 HTML 选择列表的 name 子句所使用的语法通常也用于 AWP_In_Variable 命令的 name 子句。通常用单引号将名称括起来，但如果引用数据块，则应将数据块名称用双引号括起来。

有关表单管理的详细信息，请参考 HTML 文档。

语法

```
<!-- AWP_In_Variable Name='<Varname1>' [Use='<Varname2>'] ... -->
```

参数

<Varname1>	如果未提供 Use 子句，则 Varname1 表示要写入的变量。它可以是 STEP 7 程序中的 PLC 变量名称，或者是特定数据块中的变量。 如果提供了 Use 子句，则 Varname1 为 <Varname2> 中引用的变量的备用名称 (页 1076)。它是 HTML 页面中的本地名称。
<Varname2>	如果提供了 Use 子句，则 Varname2 表示要写入的变量。它可以是 STEP 7 程序中的 PLC 变量名称，或者是特定数据块中的变量。

对于 Name 子句和 Use 子句，必须用单引号将整个名称括起来。在单引号中，用双引号将 PLC 变量括起来，并用双引号将数据块名称括起来。数据块名称括在双引号之内，但数据块变量名称不在双引号之内。请注意，对于数据块变量，应使用块名称，而不是数据块编号。使用数组元素语法引用数组元素。

如果使用 AWP_In_Variable 命令使一个数据块可写入，则该数据块中的每一个变量均可写入。

HTML 输入域的用法示例

```
<!-- AWP_In_Variable Name='"Target_Level"' -->
<form method="post">
<p>Input Target Level:<input name='"Target_Level"' type="text" />
</p>
</form>

<!-- AWP_In_Variable Name='"Data_block_1".Braking' -->
<form method="post">
<p>Braking:<input name='"Data_block_1".Braking' type="text" />
%</p>
</form>

<!-- AWP_In_Variable Name='"Data_block_1".Array_Dim2' -->
<form method="post">
<p>二维数组值 [2,1]: <input name='"Data_block_1".Array_Dim2[2,1]'
type="text" /> %</p>
</form>
```

使用 Use 子句的示例

```
<!-- AWP_In_Variable Name='"Braking"'
Use='"Data_block_1".Braking' -->
<form method="post">
<p>Braking:<input name='"Braking"' type="text" /> %</p>
</form>
```

使用可写入数据块的示例

```
<!-- AWP_In_Variable Name='"Data_block_1"' -->
<form method="post">
<p>Braking:<input name='"Data_block_1".Braking' type="text" /> %
</p>
<p>Turbine Speed:<input name='"Data_block_1".TurbineSpeed'
size="10" value='"Data_block_1".TurbineSpeed' type="text" />
</p>
</form>
```

HTML 选择列表的用法示例

```
<!-- AWP_In_Variable Name='"Data_block_1".ManualOverrideEnable'--
>
```

```
<form method="post">
<select name='"Data_block_1".ManualOverrideEnable'>
<option value=:"Data_block_1".ManualOverrideEnable:> </option>
<option value=1>√</option>
<option value=0>×</option>
</select><input type="submit" value="Submit setting" /></form>
```

说明

只有具有“写入用户自定义页面”权限 (页 1029)的用户才能将数据写入 CPU。如果用户没有修改权限，Web 服务器会忽略这些命令。

如果变量名称或数据块名称包含特殊字符，则必须使用附加引号或转义字符，如主题“处理包含特殊字符的变量名称 (页 1081)”中所述。

12.8.2.3 读取特殊变量

Web 服务器可以从 PLC 中读取要存储在 HTTP 响应标题的特殊变量中的值。例如，您可能要从 PLC 变量中读取路径名，以使用 HEADER:Location 特殊变量将 URL 重新定向到其它位置。

语法

```
<!-- AWP_Out_Variable Name='<Type>:<Name>' [Use='<Varname>'] -->
```

参数

<Type>	特殊变量的类型包括： HEADER COOKIE_VALUE COOKIE_EXPIRES
<Name>	要获取所有 HEADER 变量名称的列表，请参考 HTTP 文档。下面给出了几个示例： Status：响应代码 Location：重定向的路径 Retry-After：预计服务在多长时间内对请求客户机不可用 对于 COOKIE_VALUE 和 COOKIE_EXPIRES 类型，<Name> 是特定 cookie 的名称。 COOKIE_VALUE:name：指定的 cookie 的值 COOKIE_EXPIRES:name：指定的 cookie 的到期时间（以秒为单位） 必须用单引号或双引号将 Name 子句括起来。 如果未指定 Use 子句，则特殊变量名称与 PLC 变量名称相对应。用单引号将完整的 Name 子句括起来，用双引号将 PLC 变量括起来。特殊变量名称和 PLC 变量名称必须完全匹配。
<Varname>	要读取的变量对应的 PLC 变量或数据块变量的名称 必须用单引号将 Varname 括起来。在单引号中，用双引号将 PLC 变量或数据块名称括起来。数据块名称括在双引号之内，但数据块变量名称不在双引号之内。请注意，对于数据块变量，应使用块名称，而不是数据块编号。

如果变量名称或数据块名称包含特殊字符，则必须使用附加引号或转义字符，如主题处理包含特殊字符的变量名称 (页 1081)中所述。

示例：读取没有 Use 子句的特殊变量

```
<!-- AWP_Out_Variable Name='"HEADER:Status"' -->
```

本示例中，HTTP 特殊变量“HEADER:Status”会接收 PLC 变量“HEADER:Status”的值。如果未指定 Use 子句，则 PLC 变量表中的名称必须与特殊变量的名称完全匹配。

示例：读取具有 Use 子句的特殊变量

```
<!-- AWP_Out_Variable Name='HEADER:Status' Use='"Status"' -->
```

本示例中，特殊变量“HEADER:Status”会接收 PLC 变量“Status”的值。

12.8.2.4 写入特殊变量

Web 服务器可以将 HTTP 请求标题中特殊变量的值写入 CPU。 例如，可以将与用户自定义网页相关的 cookie 信息、正在访问页面的用户信息或标题信息存储在 STEP 7 中。 Web 服务器可用于访问指定的特殊变量，以具有修改变量权限的用户身份登录时可将这些变量写入 CPU。

语法

```
<!-- AWP_In_Variable Name='<Type>:<Name>' [Use='<Varname>']-->
```

参数

<Type>	特殊变量的类型有： HEADER SERVER COOKIE_VALUE
<Name>	上面定义的类型中的特殊变量，如以下示例中所示： HEADER:Accept：可接受的内容类型 HEADER:User-Agent：有关发起请求的用户代理的信息。 SERVER:current_user_id：当前用户的 id；如果没有用户登录，则该变量为 0 SERVER:current_user_name：当前用户的名称 COOKIE_VALUE:<name>：指定的 cookie 的值 用单引号将 Name 子句括起来。 如果未指定 Use 子句，则特殊变量名称与 PLC 变量名称相对应。 用单引号将完整的 Name 子句括起来，用双引号将 PLC 变量括起来。 特殊变量名称和 PLC 变量名称必须完全匹配。 要获取所有 HEADER 变量名称的列表，请参考 HTTP 文档。
<Varname>	要写入特殊变量的 STEP 7 程序中的变量名，它可以是 PLC 变量名称或数据块变量。 必须用单引号将 Varname 括起来。 在单引号中，用双引号将 PLC 变量或数据块名称括起来。 数据块名称括在双引号之内，但数据块变量名称不在双引号之内。 请注意，对于数据块变量，应使用块名称，而不是数据块编号。

示例

```
<!-- AWP_In_Variable Name='"SERVER:current_user_id"' -->
```

在本示例中，Web 页面将 HTTP 特殊变量“SERVER:current_user_id”的值写入名称为“SERVER:current_user_id”的 PLC 变量中。

```
<!-- AWP_In_Variable Name=SERVER:current_user_id'
Use='"my_userid"' -->
```

在本示例中，Web 页面将 HTTP 特殊变量“SERVER:current_user_id”的值写入名称为“my_userid”的 PLC 变量中。

说明

只有具备修改变量权限的用户才能将数据写入 CPU。 如果用户没有修改权限，Web 服务器会忽略这些命令。

如果变量名称或数据块名称包含特殊字符，则必须使用附加引号或转义字符，如主题“处理包含特殊字符的变量名称 (页 1081)”中所述。

12.8.2.5 对变量引用使用别名

可以在用户定义的 Web 页面中为 In_Variable 或 Out_Variable 使用别名。 例如，您可以在 HTML 页面中使用不同于 CPU 中的符号名称，也可以使 CPU 中的变量与特殊变量等同。 AWP 的 Use 子句就可以实现这一功能。

语法

```
<-- AWP_In_Variable Name='<Varname1>' Use='<Varname2>' -->
<-- AWP_Out_Variable Name='<Varname1>' Use='<Varname2>' -->
```

参数

<Varname1>	必须将别名或特殊变量名称 Varname1 用单引号或双引号括起来。
<Varname2>	要为其分配别名的 PLC 变量的名称。 该变量可以是 PLC 变量、数据块变量或特殊变量。 必须用单引号将 Varname2 括起来。 在单引号中，用双引号将 PLC 变量、特殊变量或数据块名称括起来。 数据块名称括在双引号之内，但数据块变量名称不在双引号之内。 请注意，对于数据块变量，应使用块名称，而不是数据块编号。

示例

```
<-- AWP_In_Variable Name='SERVER:current_user_id'
Use='"Data_Block_10".server_user' -->
```

本示例中，特殊变量 SERVER:current_user_id 将被写入数据块“Data_Block_10”中的变量“server_user”。

```
<-- AWP_Out_Variable Name='Weight'
Use='"Data_Block_10".Tank_data.Weight'  -->
```

本示例中，可以在用户定义的 Web 页面的其余部分仅由“Weight”引用数据块结构成员 Data_Block_10.Tank_data.Weight 中的值。

```
<-- AWP_Out_Variable Name='Weight' Use='"Raw_Milk_Tank_Weight"'
-->
```

在本示例中，可以在用户定义的 Web 页面的其余部分仅由“Weight”引用 PLC 变量“Raw_Milk_Tank_Weight”中的值。

如果变量名称或数据块名称包含特殊字符，则必须使用附加引号或转义字符，如处理包含特殊字符的变量名称 (页 1081)主题中所述。

12.8.2.6 定义枚举类型

可以使用 AWP 命令在用户定义的页面中定义枚举类型并分配元素。

语法

```
<!-- AWP_Enum_Def Name='<Enum type name>' Values='<Value>,
<Value>,... ' -->
```

参数

<Enum type name>	枚举类型的名称，用单引号或双引号括起来。
<Value>	<constant>:<name> constant 表示枚举类型分配的数字值。 总数不受限制。 name 是分配给枚举元素的值。

请注意，应将整个枚举值分配字符串用单引号括起来，而将每个单独的枚举类型元素分配用双引号括起来。 对于用户定义的 Web 页面，枚举类型定义在全局范围内有效。 如果已在语言文件夹 (页 1101)中创建用户定义的 Web 页面，则对于语言文件夹中的所有页面，枚举类型定义在全局范围内有效。

示例

```
<!-- AWP_Enum_Def Name='AlarmEnum' Values='0:"No alarms", 1:"Tank
is full", 2:"Tank is empty"' -->
```

12.8.2.7 通过枚举类型引用 CPU 变量

可以将 CPU 中的变量分配给枚举类型。 在执行读操作 (页 1070)或写操作 (页 1071)时，可在用户定义的 Web 页面中的其它位置使用该变量。 在读操作中，Web 服务器将用相应的枚举文本值替换从 CPU 中读取的数字值。 在写操作中，Web 服务器将在向 CPU 写入值之前用与文本对应的枚举整型值来替换文本值。

语法

```
<!-- AWP_In_Variable Name='<Varname>' Enum="<EnumType>" -->
<!-- AWP_Out_Variable Name='<Varname>' Enum="<EnumType>" -->
```

参数

<Varname>	与枚举类型相关联的 PLC 变量或数据块变量的名称，或 PLC 变量的别名 (页 1076)（如已声明）。 必须用单引号将 Varname 括起来。 在单引号中，用双引号将 PLC 变量或数据块名称括起来。 请注意，对于数据块变量，应使用块名称，而不是数据块编号。 数据块名称括在双引号之内，但数据块变量名称不在双引号之内。
<EnumType>	枚举类型的名称，必须用单引号或双引号括起来。

枚举类型引用的适用范围为当前片段。

变量读取的用法示例

```
<!-- AWP_Out_Variable Name='"Alarm"' Enum="AlarmEnum" -->...
<p>The current value of "Alarm" is :="Alarm":</p>
```

如果 CPU 中“Alarm”的值为 2，则 HTML 页面会显示“The current value of "Alarm" is Tank is empty”，这是因为枚举类型定义 (页 1077)将文本字符串“Tank is empty”分配给了数字值 2。

变量写入的用法示例

```
<!-- AWP_Enum_Def Name='AlarmEnum' Values='0:"No alarms", 1:"Tank
is full", 2:"Tank is empty"' -->
<!-- AWP_In_Variable Name='"Alarm"' Enum='AlarmEnum' -->...
<form method="POST">
<p><input type="hidden" name='"Alarm"' value="Tank is full" /></p>
<p><input type="submit" value='Set Tank is full' /><p>
```

```
</form>
```

由于枚举类型定义 (页 1077)将“Tank is full”分配给了数字值 1，因此，值 1 被写入 CPU 中的 PLC 变量“Alarm”。

请注意，AWP_In_Variable 声明中的 Enum 子句必须与 AWP_Enum_Def 声明中的 Name 子句完全对应。

在变量写入过程中使用别名的用法示例

```
<!-- AWP_Enum_Def Name='AlarmEnum' Values='0:"No alarms", 1:"Tank
is full", 2:"Tank is empty"' -->
<!-- AWP_In_Variable Name='"Alarm"' Enum='AlarmEnum'
Use='"Data_block_4".Motor1.Alarm'-->...
<form method="POST">
<p><input type="hidden" name='"Alarm"' value="Tank is full" /></p>
<p><input type="submit" value='Set Tank is full' /><p>
</form>
```

由于枚举类型定义 (页 1077)将“Tank is full”分配给了数字值 1，因此，值 1 被写入 CPU 中与数据块“Data_Block_4”的 PLC 变量“Motor1.Alarm”对应的别名“Alarm”。

如果变量名称或数据块名称包含特殊字符，则必须使用附加引号或转义字符，如主题处理包含特殊字符的变量名称 (页 1081)中所述。

说明

上一版本要求具有独立的 AWP_Enum_Ref 声明才能够将变量与已定义的枚举类型相关联。STEP 7 和 S7-1200 支持包含 AWP_Enum_Ref 声明的现有代码；但现在已不再需要此命令。

12.8.2.8 创建片段

单击 Web 服务器的“CPU 属性”(CPU Properties) 中的 “生成块”(Generate blocks) 时，STEP 7 会将用户定义的 Web 页面转换和存储为控制 DB 和片段 DB。 可以为特定页面或特定页面部分建立特殊片段。 可以使用 AWP 命令“Start_Fragment”通过名称和编号来标识这些片段。 执行 AWP_Start_Fragment 命令之后，页面中的所有内容都属于该片段，直到再发出 AWP_Start_Command 或到达文件结尾。

语法

```
<!-- AWP_Start_Fragment Name='<Name>' [Type=<Type>][ID=<id>]
[Mode=<Mode>] -->
```

参数

<Name>	文本字符串： 片段 DB 的名称 片段名称必须以字母或下划线开头，并由字母、数字和下划线组成。 片段名称是以下形式的正则表达式： [a-zA-Z_][a-zA-Z_0-9]*
<Type>	“manual”或“automatic” manual： STEP 7 程序必须请求该片段并可相应地做出响应。 必须使用 STEP 7 和控制 DB 变量来控制片段的操作。 automatic： Web 服务器自动处理片段。 如果未指定类型参数，则默认设置为“automatic”。
<id>	整型标识号。 如果未指定 ID 参数，则 Web 服务器会默认分配一个数字。 对于手动片段，应将 ID 设为较小的数字。 ID 是 STEP 7 程序控制手动片段的方法。
<模式>	“可见” 或 “隐藏” 可见： 片段的内容将显示在用户定义的 Web 页面上。 隐藏： 片段的内容将不会显示在用户定义的 Web 页面上。 如果未指定类型参数，则默认设置为 “可见”(visible)。

手动片段

如果为用户定义的 Web 页面或页面部分创建手动片段，则 STEP 7 程序必须控制片段的发送时间。 手动控制时，STEP 7 程序必须在控制 DB 中为用户定义的页面设置相应的参数，然后针对修改的控制 DB 调用 WWW 指令。 要了解控制 DB 的结构以及如何处理各个页面和片段，请参见主题 高级用户定义 Web 页面控制 (页 1104)。

12.8.2.9 导入片段

可以根据部分 HTML 代码创建指定的片段，然后将该片段导入用户定义的 Web 页面集的其它位置。 例如，有一个包含起始页面的用户定义的 Web 页面集，还有几个可通过该起始页面中的链接访问的其它 HTML 页面。 假定每个单独的页面都要在页面上显示公司徽标。 那么，您可以创建片段 (页 1079)，用来加载公司徽标的图像。 每个单独的 HTML 页面都可以导入该片段以显示公司徽标。 您可以使用 AWP Import_Fragment 命令来执行该操作。 片段的 HTML 代码只存在于一个片段中，但您可以根据需要将该片段 DB 多次导入所选的众多 Web 页面中。

语法

```
<!-- AWP_Import_Fragment Name='<名称>' -->
```

参数

<Name>	文本字符串： 要导入的片段 DB 的名称

示例

用来创建显示图像的片段的 HTML 代码摘录：

```
<!-- AWP_Start_Fragment Name='My_company_logo' --><p><img
src="company_logo.jpg"></p>
```

导入了显示徽标图像的片段的其它 .html 文件中的 HTML 代码摘录：

```
<!-- AWP_Import_Fragment Name='My_company_logo' -->
```

这两个 .html 文件（一个文件创建片段，另一个文件导入片段）都位于您在 STEP 7 中组态用户定义的页面 (页 1083)时所定义的文件夹结构中。

12.8.2.10 组合定义

声明要在用户定义的 Web 页面中使用的变量时，可以将变量声明与变量的别名 (页 1076)组合在一起。还可以在一条语句中声明多个 In_Variable 以及多个 Out_Variable。

示例

```
<!-- AWP_In_Variable Name='"Level"' Name='"Weight"' Name='"Temp"'
-->
<!-- AWP_Out_Variable Name='HEADER:Status' Use='"Status"'
         Name='HEADER:Location' Use="Location"
         Name='COOKIE_VALUE:name' Use="my_cookie" -->
<!-- AWP_In_Variable Name='Alarm' Use='"Data_block_10".Alarm' -->
```

12.8.2.11 处理包含特殊字符的变量名称

在用户定义的 Web 页面中指定变量名称时，如果变量名称中包含具有特殊含义的字符，则需要特别注意。

读取变量

可以使用以下语法读取变量 (页 1070)：

```
:=<Varname>:
```

在读取变量时下列规则适用：

- 对于 PLC 变量表中的变量名称，应将变量名称用双引号括起来。
- 对于作为数据块变量的变量名称，应将该数据块名称用双引号括起来。变量在引号外。
- 对于直接 I/O 地址、存储器地址或别名等变量名称，请勿使用引号将读取变量括起来。
- 对于包含反斜杠的变量名称或数据块变量名称，应在反斜杠前面再加一个反斜杠。
- 如果变量名称或数据块变量名称包含冒号、小于号、大于号或 & 号，则为读取的变量定义不含特殊字符的别名，并通过该别名读取变量。在 Use 子句的变量名称中，在冒号前加上反斜杠。

表格 12-1　读取变量示例

数据块名称	变量名称	读取命令
不适用	ABC:DEF	`<!--AWP_Out_Variable Name='special_tag' Use ='"ABC:DEF"' -->` `:=special_tag:`
不适用	T\	`:="T\\":`
不适用	A \B 'C :D	`<!--AWP_Out_Variable Name='another_special_tag' Use='"A \\B \'C :D"' -->` `:=another_special_tag:`
不适用	a<b	`<!--AWP_Out_Variable Name='a_less_than_b' Use='"a<b"' -->` `:=a_less_than_b:`
Data_block_1	Tag_1	`:="Data_block_1".Tag_1:`
Data_block_1	ABC:DEF	`<!-- AWP_Out_Variable Name='special_tag' Use='"Data_block_1".ABC\:DEF'-->` `:=special_tag:`
DB A' B C D$ E	Tag	`:="DB A' B C D$ E".Tag:`
DB:DB	Tag:Tag	`<!--AWP_Out_Variable Name='my_tag' Use ='"DB:DB".Tag\:Tag' -->` `:=my_tag:`

Name 和 Use 子句

AWP 命令 AWP_In_Variable、AWP_Out_Variable、AWP_Enum_Def、AWP_Enum_Ref、AWP_Start_Fragment 和 AWP_Import_Fragment 含有 Name 子句。<input> 和 <select> 等 HTML 表单命令也含有 name 子句。AWP_In_Variable 和 AWP_Out_Variable 还可以含有 Use 子句。对于所有命令，在特殊字符处理方面 Name 和 Use 子句的语法相同：

- 为 Name 或 Use 子句提供的文本必须用单引号括起来。如果括起来的名称是 PLC 变量或数据块名称，则应使用单引号将整个子句括起来。
- 在 Name 或 Use 子句中，数据块名称和 PLC 变量名称必须用双引号括起来。
- 如果变量名称或数据块名称包含单引号字符或反斜杠，应使用反斜杠将该字符转义。反斜杠在 AWP 命令编译器中是转义字符。

表格 12-2　Name 子句示例

数据块名称	变量名称	Name 子句可选形式
不适用	ABC'DEF	`Name='"ABC\'DEF"'`
不适用	A \B 'C :D	`Name='"A \\B \'C :D"'`
Data_block_1	Tag_1	`Name='"Data_block_1".Tag_1'`
Data_block_1	ABC'DEF	`Name='"Data_block_1".ABC\'DEF'`
Data_block_1	A \B 'C :D	`Name='"Data_block_1".A \\B \'C :D'`
DB A' B C D$ E	Tag	`Name='"DB A\' B C D$ E".Tag'`

Use 子句遵从 Name 子句的约定。

说明

无论在 HTML 页面中使用什么字符，都应将 HTML 页面的字符集设置为 UTF-8 并在编辑器中使用 UTF-8 字符编码对其进行保存。

12.8.3 组态用户定义 Web 页面的使用

要通过 STEP 7 组态用户定义的 Web 页面，请执行以下步骤：

1. 在设备组态视图中选择 CPU。
2. 在巡视窗口中显示该 CPU 的“Web server”属性。
3. 如果尚未选择，则选中“激活此模块上的 Web 服务器”(Activate Web server on this module) 复选框。

4. 选择“仅允许使用 HTTPS 访问”(Permit access only with HTTPS) 的默认设置，确保 Web 服务器使用加密的通信，并提高 Web 可访问 CPU 的安全性。
5. 输入或浏览到 PC 上保存 HTML 默认页面（起始页面）的文件夹的名称。
6. 输入该默认页面的名称。
7. 为应用程序提供名称（可选）。Web 服务器使用应用程序名称对 web 页面进行进一步分类或分组。当您提供应用程序名称时，Web 服务器会用以下格式为用户定义页面创建 URL：http[s]://ww.xx.yy.zz/awp/<应用程序名称>/<页面名称>.html。如果未提供应用程序名称，则 URL 为 http[s]://ww.xx.yy.zz/awp/<pagename>.html。
   应用程序名称中避免使用特殊字符。有些字符可能会导致 Web 服务器无法显示用户定义的页面。

8. 在“高级”(Advanced) 部分中，输入包含 AWP 命令的文件的扩展名。默认情况下，STEP 7 分析扩展名为 .htm、.html 或 .js 的文件。如果还有其它文件扩展名，请将其附上。要节省处理资源，如果没有该类型的文件包含 AWP 命令，请不要输入文件扩展名。
9. 保留默认的 Web DB 编号，或输入适合的编号。这是控制 Web 页面显示的控制 DB 的 DB 编号。
10. 保留默认的片段 DB 起始编号，或输入适合的编号。这是首个包含 Web 页面的片段 DB。

生成程序块

单击“生成块”(Generate blocks) 按钮时，STEP 7 会基于指定的 HTML 源目录下的 HTML 页面生成数据块，以及一个用于 Web 页面操作的控制数据块。可以根据应用的需要设置这些属性 (页 1085)。STEP 7 还会生成一组片段数据块，以保存所有 HTML 页面的显示。在您生成数据块时，STEP 7 会更新属性，以显示控制数据块编号和首个片段数据块编号。生成该数据块之后，用户定义的 Web 页面就会成为 STEP 7 程序的一部分。与这些页面对应的块会出现在项目导航树中“程序块”(Program blocks) 下“系统块”(System blocks) 文件夹中的“Web 服务器”(Web server) 文件夹中。

删除程序块

要删除先前生成的数据块，请单击“Delete data blocks”按钮。STEP 7 将从项目中删除与用户定义的 Web 页面相对应的控制数据块和所有片段数据块。

12.8.4 组态入口页

在 CPU 的“设备组态”(Device Configuration) 中，可以将用户定义的 Web 页面指定为通过 PC 或移动设备访问 Web 服务器时的入口页。否则，入口页为“简介 (页 1041)” (Introduction) 标准 Web 页面。

要选择用户定义的 Web 页面作为入口页，请按以下步骤操作：

1. 在设备组态视图中选择 CPU。
2. 在巡视窗口的 CPU 属性中选择“Web 服务器”(Web server)，启用 Web 服务器 (页 1028)。
3. 在 Web 服务器属性中选择“入口页”(Entry page)。
4. 从下拉列表中选择“UP1”，将 Web 服务器组态为被访问时显示用户定义的页面。（另一个选项，即“Intro 页面”，可将 Web 服务器设置为被访问时显示标准的“简介”Web 页面。）

另外，必须将“所有人”(Everybody) 用户组态为拥有“打开用户定义的 Web 页面”权限，(页 1029)并在程序中包含对 WWW 指令 (页 1085)的调用。

完成组态并将项目下载到 CPU 后，Web 服务器便可以使用将用户定义的 Web 页面 (页 1083) 组态为入口页时所选择的“默认 HTML 页面”。

说明

CPU 必须处于 RUN 模式才能显示用户定义的入口页。

12.8.5 针对用户定义 Web 页面编写 WWW 指令

STEP 7 用户程序必须包含并执行 WWW 指令，以便能够通过标准 Web 页面访问用户定义 Web 页面。该控制数据块是 WWW 指令的输入参数，指定如片段数据块中所表示的页面内容、状态以及控制信息。在组态用户定义 Web 页面 (页 1083)时单击“Create blocks”按钮后，STEP 7 便会创建控制数据块。

编写 WWW 指令

要通过标准 Web 页面访问用户定义的 Web 页面，STEP 7 程序必须执行 WWW 指令。您可能希望用户定义的 Web 页面仅在应用程序要求和首选项指定的一些情况下可用。在这种情况下，程序逻辑可控制何时调用 WWW 指令。

表格 12-3　WWW 指令

LAD/FBD	SCL	说明
	`ret_val := WWW(` `ctrl_db:=_uint_in_);`	提供从标准 Web 页面访问用户定义的 Web 页面的权限

必须提供控制数据块输入参数 (CTRL_DB)，该参数对应于控制 DB 的整数 DB 编号。为用户定义的 Web 页面创建块后，可在 CPU 的“Web 服务器”(Web Server) 属性中找到此控制 DB 块编号（称为 Web DB 编号）。输入整数 DB 编号作为 WWW 指令的 CTRL_DB 参数。返回值 (RET_VAL) 包含函数结果。请注意，WWW 指令异步执行，RET_VAL 输出的初始值可能为 0，但这不能说明后来不会发生错误。程序会检查控制 DB 的状态，以确定应用程序是否已成功启动，或者通过随后对 WWW 的调用来检查 RET_VAL。

表格 12-4　返回值

RET_VAL	说明
0	无错误
16#00yx	x：相关位声称的请求处于等待状态： x=1：请求 0 x=2：请求 1 x=4：请求 2 x=8：请求 3 可以对 x 值进行逻辑或运算，以说明多个请求处于等待状态。例如，如果 x = 6，则说明请求 1 和请求 2 处于等待状态。 y：0：没有错误；1：存在错误，并且已在控制 DB 中置位“last_error” (页 1104)
16#803a	未装载控制 DB。
16#8081	控制 DB 的类型、格式或版本错误。
16#80C1	没有资源可用于初始化 Web 应用程序。

控制 DB 的使用

单击“生成块”(Generate blocks) 后，STEP 7 将创建控制数据块，并在用户定义的 Web 页面属性中显示控制 DB 编号。您可以在项目导航树中的“程序块”(Program blocks) 文件夹下找到该控制 DB。

通常情况下，STEP 7 程序会直接使用“生成块”(Generate blocks) 过程中创建的控制 DB，而不进行任何处理。但是，STEP 7 用户程序可以在控制 DB 中设置全局命令，以禁用 Web 服务器或随后重新启用 Web 服务器。对于创建为手动片段 DB (页 1083) 的用户定义页面，STEP 7 用户程序必须通过控制 DB 中的请求表来控制这些页面的特性。有关这些高级任务的信息，请参见主题高级用户定义 Web 页面控制 (页 1104)。

12.8.6 将程序块下载到 CPU

生成用户定义的 Web 页面块后，它们与所有其它程序块一样成为 STEP 7 程序的一部分。可以按照正常过程将这些程序块下载到 CPU。 注意，只能在 CPU 处于 STOP 模式时下载用户定义的网页程序块。

12.8.7 访问用户定义的 Web 页面

用户可以通过标准 Web 页面 (页 1030)访问用户定义 Web 页面。标准 Web 页面在左侧导航菜单中显示“用户定义的页面”的链接。基本页面导航也会提供“用户定义的页面”的链接。单击“用户定义的页面”链接时，Web 服务器将跳转到提供默认页面链接的页面。在用户定义的页面中，导航视特定页面的设计情况而定。

说明

您也可以针对 Web 服务器将用户定义的页面定义为入口页 (页 1085)。

12.8.8 特定于用户定义 Web 页面的限制

标准 Web 页面的限制 (页 1108)也适用于用户定义 Web 页面。 此外，用户定义 Web 页面会有一些特殊考虑。

装载存储器空间

单击“生成块”(Generate blocks) 之后，用户定义的 Web 页面就会成为数据块，这一过程需要用到装载存储器空间。 如果安装了存储卡，用户定义 Web 页面的外部装载存储器空间最大容量即为存储卡的容量。

如果未安装存储卡，这些块就会占用内部装载存储器空间，根据 CPU 型号的不同而会存在限制。

可以在 STEP 7 中通过“在线和诊断”工具检查已用装载存储器空间量和可用装载存储器空间量。还可以查看 STEP 7 基于用户定义 Web 页面生成的各个块的属性，并查看装载存储器使用量。

说明

如果需要减少用户定义 Web 页面所需空间，则减少图片的使用（如果适用）。

文本字符串中的引号

在用户自定义 Web 页面中，避免在用于任意目的的数据块变量中使用含有嵌入式单引号或双引号的文本字符串。 因为在 HTML 语法中，经常将单引号或双引号用作分隔符，文本字符串内的引号会破坏用户自定义 Web 页面的显示。

对于在用户自定义 Web 页面中使用的 String 型数据块变量，应遵守以下准则：

- 不要在 STEP 7 中为数据块变量字符串值输入单引号或双引号。
- 不要让用户程序将含有引号的字符串分配给这些数据块变量。

12.8.9 用户定义 Web 页面示例

12.8.9.1 用于监控风力发电机的 Web 页面

可以考虑将用于远程监控风力发电机的 Web 页面作为用户定义 Web 页面示例：

说明

在此应用中，风电厂中的每台风力发电机都配备一个 S7-1200 用于控制涡轮机。 在 STEP 7 程序中，各风力发电机都有一个数据块，其中包含该风力发电机的特定数据。

用户定义 Web 页面可用于通过 PC 远程访问涡轮机。 用户可以连接到特定风力发电机的 CPU 的标准 Web 页面并访问用户定义“Remote Wind Turbine Monitor”Web 页面，以查看该涡轮机的数据。 有权修改变量的用户还可以通过该 Web 页面将涡轮机置于手动模式，控制涡轮机速度、偏航和桨距变量。 此外，无论涡轮机处于手动还是自动控制模式，有权修改变量的用户都可设置制动值。

STEP 7 程序会检查用于替代自动控制的布尔值，如果为真，则涡轮机速度、偏航和桨距将使用用户输入值。 否则，程序将忽略这些值。

使用的文件

此用户定义 Web 页面示例包含三个文件：

- **Wind_turbine.html**： 这是实现上面所示显示画面的 HTML 页面，该页面使用 AWP 命令访问控制器数据。
- **Wind_turbine.css**： 这是包含该 HTML 页面格式样式的级联样式表。 可以选择是否使用级联样式表，但使用它可以简化 HTML 页面的开发。
- **Wind_turbine.jpg**：这是 HTML 页面使用的背景图片。 当然，可以选择是否在用户定义 Web 页面中使用图片，使用图片会额外占用 CPU 的存储空间。

这些文件没有随安装程序一起提供，但在这里将其作为示例来介绍。

实现

HTML 页面使用 AWP 命令从 PLC 读取值 (页 1070)（用于显示字段），并将值写入 PLC (页 1071)（用于用户输入的数据）。 此页面还使用 AWP 命令进行枚举类型定义 (页 1077) 和引用 (页 1078)以处理 ON/OFF 设置。

页面的第一部分显示标题行，其中包含风力发电机的编号。

页面的下一部分显示风力发电机所处的环境条件。 由涡轮机现场的 I/O 提供风速、风向和当前温度。

接下来，页面显示从 S7-1200 读取的涡轮机的功率输出。

以下部分介绍了涡轮机的手动控制，手动控制会替代由 S7-1200 负责的标准的自动控制。各类型如下所述：

- 手动替代： 启用对涡轮机设置的手动替代。 只有手动替代设置为真时，STEP 7 用户程序才会对涡轮机速度、偏航或桨距使用手动设置。

- 偏航替代： 启用对偏航设置的手动替代，对偏航使用手动设置。 手动替代和偏航替代都为真时，STEP 7 用户程序才会应用偏航设置。

- 桨距替代： 启用对叶片桨距的手动替代。 手动替代和桨距替代都为真时，STEP 7 用户程序才会应用叶片桨距设置。

HTML 页面包含一个用于将替代设置传送给控制器的提交按钮。

制动用户输入字段提供手动制动设置百分数。无需启用手动替代，STEP 7 用户程序便会接受制动值。

此外，HTML 页面还使用 AWP 命令来写入特殊变量 (页 1075)（其包含正在访问该页面的用户的用户 ID）到 PLC 变量表中的一个变量。

12.8.9.2 读取和显示控制器数据

“Remote Wind Turbine Monitor”HTML 页面使用多个 AWP 命令从控制器读取数据 (页 1070)，并将其显示在该页面中。 例如，考虑一下用于显示该示例 Web 页面的下面部分所示功率输出的 HTML 代码：

HTML 代码示例

“Remote Wind Turbine Monitor”HTML 页面的以下节选内容在一个表格行的左侧单元格内显示文本“Power Output:”，读取用于该功率输出的变量，并连同“kilowatts”的文本缩写“kW”显示在该表格行的右侧单元格中。

AWP 命令 :="Data_block_1".PowerOutput: 执行读取操作。 请注意，数据块通过名称而不是通过数据块编号来引用（即，通过“Data_block_1”而不是“DB1”）。

```
<tr style="height:2%;">
<td>
<p>功率输出：</p>
</td>
<td>
<p style="margin-bottom:5px;"> :="Data_block_1".PowerOutput:
kW</p>
</td>
</tr>
```

12.8.9.3 使用枚举类型

“Remote Wind Turbine Monitor”HTML 页面在三种情况下使用枚举类型：HTML 页面显示代表布尔值的“ON”或“OFF”和用户设置布尔值。 “ON”枚举类型使值为 1，“OFF”枚举类型使值为 0。例如，考虑一下用于在 "Data_block_1".ManualOverrideEnable 值中使用枚举类型读取和写入 Manual Override Enable 设置的 HTML 代码：

HTML 代码示例

从“Remote Wind Turbine Monitor”HTML 页面中摘录的以下部分显示如何声明值为“Off”和“On”（分别代表 0 和 1）的“OverrideStatus”枚举类型，以及随后设置对数据块（名为“Data_block_1”）中 ManualOverrideEnable 布尔变量的 OverrideStatus 的枚举类型引用。

```
<!-- AWP_In_Variable Name='"Data_block_1".ManualOverrideEnable'
Enum="OverrideStatus" -->

<!-- AWP_Enum_Def Name="OverrideStatus" Values='0:"Off",1:"On"'
-->
```

其中，HTML 页面在表格单元中包含一个用于显示 ManualOverrideEnable 当前状态的显示字段，该页面通常仅使用标准读取变量命令，但由于使用了先前声明和引用的枚举类型，所以该页面显示“Off”或“On”，而不是 0 或 1。

```
<td style="width:24%; border-top-style: Solid; border-top-width:
2px; border-top-color: #ffffff;">
<p>手动替代：  :="Data_block_1".ManualOverrideEnable:</p>
</td>
```

该 HTML 页面包含一个下拉选择列表，供用户更改 ManualOverrideEnable 的值。 该选择列表在选择列表中显示文本“Yes”和“No”。 使用枚举类型时，“Yes”关联到枚举类型的值“On”，“No”关联到值“Off”。 选项为空时保持 ManualOverrideEnable 的值不变。

```
<select name='"Data_block_1".ManualOverrideEnable'>
<option value=':"Data_block_1".ManualOverrideEnable:'> </option>
<option value="On">Yes</option>
<option selected value="Off">No</option>
</select>
```

选择列表包含在 HTML 页面上的表单中。 当用户单击提交按钮时，该页面将发布表单，如果用户已选择“Yes”，则会将值“1”写入 Data_block_1 中的布尔型 ManualOverrideEnable；如果用户选择的是“No”，则写入值“0”。

12.8.9.4 将用户输入写入控制器

“Remote Wind Turbine Monitor”HTML 页面包含多个用于将数据写入控制器的 AWP 命令 (页 1071)。 HTML 页面将 AWP_In_Variables 声明为布尔变量，以便有权修改变量的用户可以将风力发电机置于手动控制模式，并且可以启用涡轮机速动的手动替代、偏航替代和/或叶片桨距替代。 该页面还使用 AWP_In_Variables 来让有权修改变量的用户随后为涡轮机速度、偏航、桨距和制动百分数设置浮点值。 该页面使用 HTTP 表单发布命令将 AWP_In_Variables 写入控制器。

例如，考虑一下用于手动设置制动值的 HTML 代码：

HTML 代码示例

从“Remote Wind Turbine Monitor”HTML 页面中摘录的以下部分首先为 AWP_In_Variable 声明“Data_block_1”，该变量使 HTML 页面可以写入到数据块“Data_block_1”中的任何变量。 该页面在一个表格行的左侧单元格中显示文本“Braking:”。 表格行的右侧单元是一个字段，可接受用户为“Data_block_1”的“Braking”变量输入的值。 此用户输入值在 HTML 表单中，该表单使用 HTTP 方法“POST”将输入的文本数据传送到 CPU。 然后，页面会从控制器读取实际制动值，并在数据输入字段中显示该值。

有权修改变量的用户随后可以通过此页面将制动值写入制动控制 CPU 的数据块中。

```
<!-- AWP_In_Variable Name='"Data_block_1"' -->
...
<tr style="vertical-align: top; height: 2%;">
<td style="width: 22%;"><p>制动：</p></td>
<td>
<form method="POST">
<p><input name='"Data_block_1".Braking' size="10" type="text">
%</p>
</form>
</td>
</tr>
```

说明

请注意，如果用户定义的页面具有用于字符串数据类型可写入数据块变量的数据输入字段，则在该字段中输入字符串值时，用户必须用单引号将字符串括起来。

说明

请注意，如果在 AWP_In_Variable 声明中声明整个数据块，例如 <!-- AWP_In_Variable Name="'Data_block_1'" -->，则可从用户自定义 Web 页面写入该数据块内的所有变量。 如果要使数据块中的所有变量均为可写入变量，请使用此方法。 而如果仅想从用户自定义 Web 页面写入特定的数据块变量，则应在声明中对其进行具体声明，例如 <!-- AWP_In_Variable Name="'Data_block_1".Braking' -->

12.8.9.5 写入特殊变量

假如用户具有修改权限，“Remote Wind Turbine Monitor”Web 页面会将特殊变量 SERVER:current_user_id 写入到 CPU 中的 PLC 变量。 这种情况下，PLC 变量值包含正在访问“Remote Wind Turbine Monitor”Web 页面的用户的用户 ID。

Web 页面将特殊变量写入到 PLC 并且不需要用户界面。

HTML 代码示例

```
<!-- AWP_In_Variable Name="SERVER:current_user_id"
Use="User_ID"-->
```

12.8.9.6 引用： 远程风力发电机监视 Web 页面的 HTML listing

Wind_turbine.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<!--
该测试程序将仿真一个 Web 页面，用于监视和控制一个风力发电机
STEP 7 中所需的 PLC 变量和数据块变量：

PLC 变量：
User_ID:  Int
数据块：
Data_block_1
Data_Block_1 中的变量：

涡轮机编号： Int
风速： Real
风向： Real
温度： Real
功率输出： Real
启用手动替代： Bool
涡轮机转速： Real
偏航替代： Bool
偏航： Real
桨距替代： Bool
```

桨距： Real
制动： Real
用户自定义的网页中将显示 PLC 数据的当前值，并提供一个选择列表，以便使用枚举类型分配设置 3 个布尔值。 “提交”按钮将提交所选择的布尔值以及用于涡轮机速度、偏航和桨距的数据条目字段。 无需点击“提交”按钮即可设置制动值。

使用该页面时，无需运行实际的 STEP 7 程序。 理论上来说，如果设置了相关布尔值，则 STEP 7 程序将只计算涡轮机速度、偏航和桨距的值。 对 STEP 7 的唯一要求是，通过该页面所生成数据块的 DB 编号调用 WWW 指令。

```
-->
<!-- AWP_In_Variable Name='"Data_block_1"' -->
<!-- AWP_In_Variable Name='"Data_block_1".ManualOverrideEnable'
Enum="OverrideStatus" -->
<!-- AWP_In_Variable Name='"Data_block_1".PitchOverride'
Enum="OverrideStatus" -->
<!-- AWP_In_Variable Name='"Data_block_1".YawOverride'
Enum="OverrideStatus" -->
<!-- AWP_In_Variable Name="SERVER:current_user_id"
Use="User_ID"-->
<!-- AWP_Enum_Def Name="OverrideStatus" Values='0:"关",1:"开"' -->

<html>
<head>
<meta http-equiv="content-type" content="text/html;
charset=utf-8"><link rel="stylesheet" href="Wind_turbine.css">
<title>Remote Wind Turbine Monitor 页面</title>
</head>
<body>
<table cellpadding="0" cellspacing="2">
<tr style="height: 2%;">
<td colspan="2">
<h2>远程风力发电机监控： 涡轮机 #:="Data_block_1".TurbineNumber:</h2>
</td>

<tr style="height: 2%;"><td style="width: 25%;"><p>风速：</p></td>
<td><p> :="Data_block_1".风速： km/h</p></td>
</tr>

<tr style="height: 2%;">
<td style="width: 25%;"><p>风向：</p></td>
<td><p> :="Data_block_1".风向： 度</p></td>
</tr>

<tr style="height: 2%;"><td style="width: 25%;"><p>温度：</p></td>
<td><p> :="Data_block_1".温度： 摄氏度</p></td>
</tr>

<tr style="height: 2%;">
<td style="width: 25%;"><p>功率输出：</p></td>
<td><p style="margin-bottom:5px;"> :="Data_block_1".功率输出： kW</
p>
</td>
```

```
</tr>

<form method="POST" action="">
<tr style="height: 2%;" >
<td style="width=25%; border-top-style: Solid; border-top-width:
2px; border-top-color: #ffffff;">
<p>手动替代:  :="Data_block_1".ManualOverrideEnable:</p>
</td>
<td class="Text">设置:

<select name='"Data_block_1".ManualOverrideEnable'>
<option value=':="Data_block_1".启用手动替代: '> </option>
<option value="开">是</option>
<option value="关">否</option>
</select>

</td>
</tr>

<tr style="vertical-align: top; height: 2%;"><td style="width:
25%;"><p>涡轮机转速: </p></td>
<td>
<p style="margin-bottom:5px;"><input
name='"Data_block_1".TurbineSpeed' size="10"
value=':="Data_block_1".涡轮机转速: ' type="text"> RPM</p>
</td>
</tr>

<tr style="vertical-align: top; height: 2%;">
<td style="width: 25%;">
<p>偏航替代:  :="Data_block_1".偏航替代:  </p>
</td>
<td class="Text">设置:

<select name='"Data_block_1".YawOverride'>
<option value=':="Data_block_1".偏航替代: '> </option>
<option value="开">是</option>
<option value="关">否</option>
</select>

</td>
</tr>

<tr style="vertical-align: top; height: 2%;">
<td style="width: 25%;">
<p>涡轮机偏航: </p>
</td>
<td>
<p style="margin-bottom:5px;"><input name='"Data_block_1".Yaw'
size="10" value=':="Data_block_1".Yaw:' type="text"> 度</p>
</td>
</tr>
```

```
<tr style="vertical-align: top; height: 2%;">
<td style="width: 25%;">
<p>桨距替代:  :="Data_block_1".PitchOverride: </p>
</td>
<td class="Text">设置:

<select name='"Data_block_1".PitchOverride'>
<option value=':="Data_block_1".桨距替代: '> </option>
<option value="开">是</option>
<option value="关">否</option>
</select>

</td>
</tr>

<tr style="vertical-align: top; height: 2%;">
<td style="width=25%; border-bottom-style: Solid; border-bottom-
width: 2px; border-bottom-color: #ffffff;">
<p>叶片桨距:</p>
</td>
<td>
<p style="margin-bottom:5px;"><input name='"Data_block_1".Pitch'
size="10" value=':="Data_block_1".桨距: ' type="text"> 度</p>
</td>

</tr>
<tr style="height: 2%;">
<td colspan="2">
<input type="submit" value="提交替代设置和值">
</td>
</tr>
</form>

<tr style="vertical-align: top; height: 2%;">
<td style="width: 25%;"><p>制动:</p></td>
<td>
<form method="POST" action="">
<p> <input name='"Data_block_1".Braking' size="10"
value=':="Data_block_1".Braking:' type="text"> %</p>
</form>
</td>
</tr>
<tr><td></td></tr>

</table>
</body>
</html>
```

Wind_turbine.css

```
BODY {
```

```
    background-image: url('./Wind_turbine.jpg');
    background-position: 0% 0%;
    background-repeat: no-repeat;
    background-size: cover;
}
H2 {
    font-family: Arial;
    font-weight: bold;
    font-size: 14.0pt;
    color: #FFFFFF;
    margin-top:0px;
    margin-bottom:10px;
}
P {
    font-family: Arial;
    font-weight: bold;
    color: #FFFFFF;
    font-size: 12.0pt;
    margin-top:0px;
    margin-bottom:0px;
}
TD.Text {
    font-family: Arial;
    font-weight: bold;
    color: #FFFFFF;
    font-size: 12.0pt;
    margin-top:0px;
    margin-bottom:0px;
}
```

12.8.9.7 STEP 7 中示例 Web 页面的组态

要包括“Remote Wind Turbine Monitor”HTML 页面并将其作为 S7-1200 的用户定义 Web 页面，需要在 STEP 7 中对 HTML 页面的相关数据进行组态并基于该 HTML 页面创建数据块。

访问控制风力发电机的 S7-1200 的“CPU 属性”(CPU Properties)，并在 Web 服务器的用户定义的页面属性中输入组态信息：

组态字段

- “HTML 目录”(HTML directory)：此字段指定计算机上保存默认页面（主页或起始页面）的文件夹的完全限定路径名。使用“...”按钮可浏览至所需文件夹。
- “默认 HTML 页面”(Default HTML page)：此字段指定默认页面的文件名或 HTML 应用的主页。使用“...”按钮可选择所需文件。在此示例中，WindTurbine.html 是默认 HTML 页面。Remote Wind Turbine Monitor 示例仅包含一个页面，但在其它用户定义应用中，默认页面可以通过默认页面中的链接调用其它页面。在 HTML 代码中，默认页面必须引用与 HTML 源文件夹有关的其它页面。
- “应用名称”(Application name)：此可选字段包含一个名称，Web 浏览器在显示该页面时会将其显示在地址字段中。在本示例中，名称为“Remote Wind Turbine Monitor”，但您可以使用任何名称。

其它字段无需组态。

最终步骤

要按照所组态的方式使用 Remote Wind Turbine Monitor，需要生成块、以生成的控制 DB 的编号为输入参数来编写 WWW 指令 (页 1085)、下载程序块并将 CPU 置于运行模式。

当操作员以后访问控制风力发电机的 S7-1200 的标准 Web 页面时，便可通过导航条上的 “用户定义的页面”(User-defined pages) 链接访问“Remote Wind Turbine Monitor”Web 页面。此时，可通过此页面监控风力发电机。

12.8.10 创建多语言用户定义 Web 页面

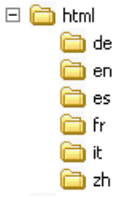
Web 服务器提供了一些用于提供以下语言形式的用户定义 Web 页面的方法：

- 德语 (de)
- 英语 (en)
- 西班牙语 (es)
- 法语 (fr)
- 意大利语 (it)
- 简体中文 (zh)

可通过在与各语言对应的文件夹结构 (页 1101)中创建 HTML 页面并在页面中设置名为“siemens_automation_language”的特定 cookie (页 1101) 来实现。 Web 服务器会响应此 cookie，并切换到相应语言文件夹中的默认页面。

12.8.10.1 创建文件夹结构

要提供多语言用户定义 Web 页面，请在 HTML 目录下创建一个文件夹结构。 文件夹名称应为特定的两个字母，必须按下面的方式命名：



de：德语
en： 英语
es： 西班牙语
fr： 法语
it： 意大利语
zh：简体中文

还可在该目录下创建页面所需的任何其它文件夹，例如，图片文件夹或脚本文件夹。

可以包括语言文件夹的任何一种子集。 不必包括所有六种语言。 在语言文件夹中，创建并设置相应语言形式的 HTML 页面。

12.8.10.2 设置语言切换

Web 服务器使用名为“siemens_automation_language”的 cookie 来执行语言切换。 此 cookie 在 HTML 页面中定义和设置，Web 服务器解释该 cookie 并相应地使用同名的语言文件夹中的相应语言来显示页面。 HTML 页面必须包含 JavaScript，这样才可将此 cookie 设置为预定义语言标识符之一： “de”、“ed”、“es”、“fr”、“it”或“zh”。

例如，如果 HTML 页面将 cookie 设置为“de”，则 Web 服务器将切换到“de”文件夹并显示具有 STEP 7 组态 (页 1103)过程中定义的默认 HTML 页面名称的页面。

示例

下面的示例使用各语言文件夹中名为“langswitch.html”的默认 HTML 页面。 此外，HTML 目录下有一个名为 "script" 的文件夹。 script 文件夹包含一个名为“lang.js”的 JavaScript 文件。 各 langswitch.html 页面使用此 JavaScript 来设置语言 cookie “siemens_automation_language”。

“en”文件夹中的“langswitch.html”的 HTML

HTML 页面的标题将语言设置为英语，将字符集设置为 UTF-8，并设置 JavaScript 文件 lang.js 的路径。

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Language" content="en">
```

```
<meta http-equiv="Content-Type" content="text/html;
charset=utf-8">
<title>Language switching english page</title>
<script type="text/javascript" src="script/lang.js" ></script>
```

文件正文使用选择列表供用户在德语和英语之间进行选择。 英语（“en”）是预选的语言。 用户更改语言时，该页面将调用具有所选选项值的 DoLocalLanguageChange() JavaScript 函数。

```
<!-- Language Selection -->
<table>
   <tr>
      <td align="right" valign="top" nowrap>
       <!-- change language immediately on selection change -->
          <select name="Language"
                    onchange="DoLocalLanguageChange(this)"
                    size="1">
            <option value="de" >German</option>
            <option value="en" selected >English</option>
           </select>
      </td>
   </tr>
</table><!-- Language Selection End-->
```

“de”文件夹中的“langswitch.html”的 HTML

除语言被设置为德语外，德语 langswitch.html 页面的标题与英语的相同。

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Language" content="de"><meta http-
equiv="Content-Type" content="text/html; charset=utf-8">
<title>Sprachumschaltung Deutsche Seite</title>
<script type="text/javascript" src="script/lang.js" ></script>
</head>
```

除所选语言的默认值为德语（“de”）外，德语页面中的 HTML 与英语页面中的完全相同。

```
<!-- Language Selection -->
<table>
   <tr>
     <td align="right" valign="top" nowrap>
      <!-- change language immediately on change of the selection
-->
       <select name="Language"
               onchange="DoLocalLanguageChange(this)"
               <size="1">
            <option value="de" selected >Deutsch</option>
            <option value="en" >Englisch</option>
       </select>
      </td>
   </tr>
</table><!-- Language Selection End-->
```

“script”文件夹中的 JavaScript“lang.js”

函数“DoLocalLanguageChange()”在 lang.js 文件中。 该函数调用“SetLangCookie()”函数，然后重新加载显示该 HTML 页面的窗口。

函数“SetLangCookie()”构造一个分配，将选择列表中的值分配给文档的“siemens_automation_language”cookie。 该函数还会设置应用程序的路径，以便被切换页面（非请求页面）接收该 cookie 的值。

可以选择在注释部分中，该页面对 cookie 的到期值进行设置。

```
function DoLocalLanguageChange(oSelect) {
        SetLangCookie(oSelect.value);
        top.window.location.reload();
    }
function SetLangCookie(value) {
        var strval = "siemens_automation_language=";
        // 这是 cookie，Web 服务器据此
        // 检测所需的语言。
        // Web 服务器需要此名称。
        strval = strval + value;
        strval = strval + "; path=/ ;";
        // 设置应用程序路径，否则
        // 该路径将被设置为请求页面，
        // 此页面将得不到 cookie。
        /* OPTIONAL
           如果此 cookie 的有效期应长于
           当前的浏览器会话，则使用期限设置：
           var now      = new Date();
           var endttime = new Date(now.getTime() + expiration);
           strval = strval + "; expires=" +
                    endttime.toGMTString() + ";";
        */
        document.cookie = strval;
    }
```

说明

如果所实现的用户定义 Web 页面包含特定语言文件夹（如“en”、“de”）内的 HTML 文件，同时还包含不在特定语言文件夹内的 HTML 文件，那么请注意，您无法使用这两个位置处的文件中的 AWP_Enum_Def 命令定义枚举类型。 如果要使用枚举类型，则必须在特定语言文件夹内的文件中或在特定语言文件夹之外的文件中定义枚举类型。 您无法在两个位置处的文件中进行枚举类型声明。

12.8.10.3 组态 STEP 7 以使用多语言页面结构

组态多语言用户定义 Web 页面的过程与组态用户定义 Web 页面 (页 1083)的常规过程类似。 为各语言创建文件夹后，将 HTML 目录设置设置为包含各语言文件夹的文件夹。 而不要将 HTML 目录设置为某一个语言文件夹。

选择默认 HTML 页面时，导航到相应的语言文件夹并选择将作为起始页面的 HTML 页面。随后生成块并将这些块下载到 CPU 时，Web 服务器将显示所组态的语言文件夹中的起始页面。

例如，如果此处所显示的文件夹结构为 C:\，则 HTML 目录设置为 C:\html，如果要选择英语作为初始页面显示语言，则导航到作为默认 HTML 页面设置的 en\langswitch.html。

12.8.11 高级用户定义 Web 页面控制

为用户定义 Web 页面生成数据块时，STEP 7 会创建一个控制 DB，该控制 DB 用于控制用户定义页面的显示和与该页面的交互。STEP 7 还会创建一组分别代表各页面的片段 DB。在正常情况下，不需要知道控制 DB 的结构或处理控制 DB 的方法。

例如，如果要开关 Web 应用程序或处理各手动片段，可使用控制 DB 变量和 WWW 指令来实现。

控制 DB 的结构

控制 DB 是一种全面的数据结构，可以在编写 STEP 7 用户程序时访问。这里仅介绍了一部分控制数据块变量。

Commandstate 结构

“Commandstate”是包含 Web 服务器的全局命令和全局状态的结构。

“Commandstate”结构中的全局命令

全局命令通常应用于 Web 服务器。可通过控制 DB 参数取消激活或重启 Web 服务器。

块变量	数据类型	说明
init	BOOL	评估控制 DB 并初始化 Web 应用程序
deactivate	BOOL	取消激活 Web 应用程序

Commandstate 结构中的全局状态

全局状态通常应用于 Web 服务器，其包含 Web 应用程序的状态信息。

块变量	数据类型	说明
initializing	BOOL	Web 应用程序正在读取控制 DB
error	BOOL	无法初始化 Web 应用程序
deactivating	BOOL	Web 应用程序正在终止
deactivated	BOOL	Web 应用程序已终止
initialized	BOOL	Web 应用程序已初始化
last_error	INT	WWW 的返回代码为 16#0010 时， 从 WWW 指令调用 (页 1085)中返回的最后一个错误： 16#0001：片段 DB 结构不一致 16#0002：应用程序名称已存在 16#0003：没有资源（存储器） 16#0004：控制 DB 结构不一致 16#0005：片段 DB 不可用 16#0006：片段 DB 不能用于 AWP 16#0007：枚举数据不一致 16#000D：控制 DB 大小冲突

请求表

请求表是包含应用于各片段 DB 的命令和状态的结构数组。如果通过“手动”类型 AWP_Start_Fragment (页 1079) 命令创建片段，则 STEP 7 用户程序必须通过控制 DB 控制这些页面。请求状态为只读，提供当前片段的相关信息。使用请求命令可控制当前片段。

块变量	数据类型	说明
requesttab	ARRAY [1 .. 4] OF STRUCT	用于各片段 DB 控制的结构数组。 Web 服务器一次最多可处理四个片段。 Web 服务器处理多个片段或通过多个浏览器会话处理片段时，特定片段的数组索引是任意的。

requesttab 结构的结构成员

块变量	数据类型	说明
page_index	UINT	当前 Web 页面的编号
fragment_index	UINT	当前片段的编号 - 可以设置为其它片段
// 请求命令		
continue	BOOL	启用发送当前页面/片段，从下一片段开始继续发送
repeat	BOOL	启用重新发送当前页面/片段，从同一片段开始继续发送
abort	BOOL	关闭 http 连接并且不进行发送
finish	BOOL	发送此片段；页面已完成 - 不处理任何其它片段
// 请求状态		请求状态为只读
idle	BOOL	空闲，但处于激活状态
waiting	BOOL	片段等待被启用
sending	BOOL	片段正在发送
aborting	BOOL	用户已终止当前请求

运行

只要程序对控制 DB 进行了更改，就必须调用 WWW 指令并将已修改的控制 DB 编号作为该指令的参数。当 STEP 7 用户程序执行 WWW 指令 (页 1085)时，全局命令和请求命令生效。

STEP 7 用户程序可明确地设置 fragment_index，从而使 Web 服务器通过请求命令处理指定片段。否则，在 WWW 指令执行时，Web 服务器将处理当前页面的当前片段。

使用 fragment_index 的可能方法包括：

- 处理当前片段：保持 fragment_index 不变并设置 continue 命令。
- 跳过当前片段：将 fragment_index 设置为 0 并设置 continue 命令。
- 将当前片段替换为其它片段：将 fragment_index 设置为新片段 ID 并设置 continue 命令。

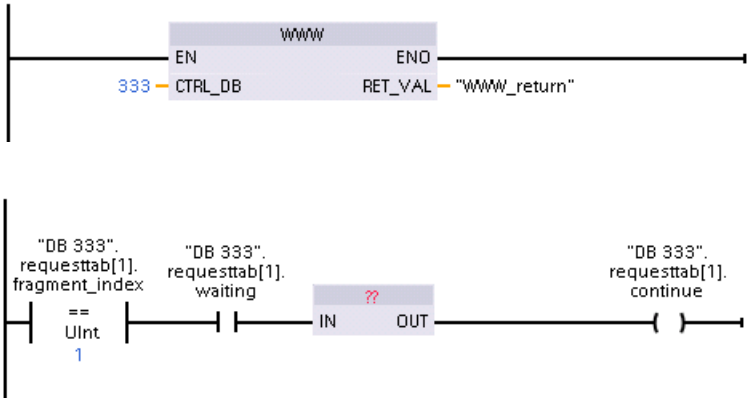
要检查全局状态或请求状态是否发生变化，STEP 7 用户程序必须调用 WWW 指令来评估这些状态的当前值。典型的方法是定期调用 WWW 指令，一直到出现特定状态。

说明

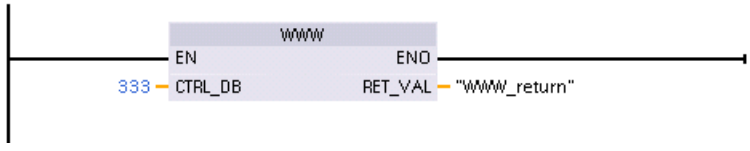
如果 STEP 7 用户程序设置了多条请求命令，则 WWW 指令仅会按优先级执行某一条命令，具体的顺序如下：abort、finish、repeat、continue。处理结束后，WWW 指令将清除所有请求命令。

示例

下面的示例显示了一个 STEP 7 用户程序，该程序检查在调用 WWW 指令后，ID 为 1 的片段是否进入等待状态。程序也可以等待其它应用程序特定条件的出现。然后执行该片段所需的任何操作，例如，设置数据块变量、执行计算或其它应用程序特定任务。然后，设置 continue 变量，以便 Web 服务器执行此片段。



当程序调用使用这一修改控制 DB 的 WWW 指令时，Web 浏览器将显示具有此片段的用户定义 Web 页面。



请注意，这是一个简化示例，要检查的片段可以是数组中的四个 requesttab 结构的任意一个。用户的程序必须处理所有四个 requesttab 结构。

12.9 限制

下列 IT 因素可能会影响 Web 服务器的使用：

- 通常，您必须使用 CPU 的 IP 地址或具有端口号的无线路由器的 IP 地址访问标准 Web 页面或用户自定义的 Web 页面。如果 Web 浏览器不允许直接连接到 IP 地址，请咨询 IT 管理员。如果本地策略支持 DNS，您可以通过指向该地址的 DNS 条目连接到 IP 地址。
- 防火墙、代理设置和其它网站特定的限制也会限制对 CPU 的访问。请咨询 IT 管理员来解决这些问题。
- 标准 Web 页面采用 JavaScript 和 cookie。如果 Web 浏览器设置禁用了 JavaScript 或 cookie，请将其启用。如果无法启用，某些功能将受限 (页 1109)。可根据需要来选择是否在用户自定义 Web 页面中使用 JavaScript 和 cookie。如果使用，则必须在浏览器中将其启用。
- Web 服务器支持安全套接层 (SSL)。可通过 URL http://ww.xx.yy.zz 或 https://ww.xx.yy.zz 来访问标准 Web 页面和用户定义的 Web 页面，其中，“ww.xx.yy.zz”表示 CPU 的 IP 地址。
- 西门子提供了用于对 Web 服务器进行安全访问的安全证书。可以从标准 Web 页面简介 (页 1041)下载安全证书，并将证书导入 Web 浏览器的 Internet 选项中 (页 1034)。如果选择不导入证书，则每次以 https:// 形式访问 Web 服务器时都会出现安全验证提示。

连接数目

Web 服务器最多支持 30 个活动连接。可以各种方式使用这 30 个连接，具体取决于您所使用的 Web 浏览器以及每页不同对象（.css 文件、图像、其它 .html 文件）的数目。在显示页面时某些连接仍然存在；其它连接在初始连接之后便不再存在。

例如，如果使用最多支持六个持续连接的特定版本 Mozilla Firefox，则在 Web 服务器开始丢弃连接前可使用五个浏览器或浏览器选项卡实例。如果页面未使用所有六个连接，则可使用更多浏览器或浏览器选项卡实例。

还要注意的是，活动连接的数目会影响页面性能。

说明

关闭 Web 服务器前注销

如果您已登录到 Web 服务器，请确保在关闭 Web 浏览器前先注销。Web 服务器最多支持 7 个并发登录。

注销失败会导致多个连接断开，具体取决于所用浏览器。通过多次打开和关闭 Web 服务器浏览器窗口而不注销，可以使用全部 30 个连接。如果使用全部连接，则用户尝试登录时，将接收到“登录无效”(Invalid login) 消息。用户需要等待 30 分钟，直至 Web 服务器释放出足够的连接，然后才能再次登录。为避免此问题，如果已登录，请始终在关闭 Web 服务器之前先注销。

12.9.1 使用 JavaScript

标准 Web 页面采用 HTML、JavaScript 和 cookie。如果站点限制使用 JavaScript 和 cookie，请将其启用，以使页面正常运行。如果无法为 Web 浏览器启用 JavaScript，将无法运行标准 Web 页面。可以考虑使用基本页面，该页面不使用 JavaScript。

参见

标准 Web 页面的布局 (页 1036)

12.9.2 Internet 选项不允许使用 cookie 时的功能限制

如果 Web 浏览器中禁用了 cookie，则会具有以下限制：

- 无法登录。
- 不能更改语言设置。
- 不能将 UTC 时间切换到 PLC 时间。 如果没有 cookie，所有时间都将采用 UTC 时间。

12.9.3 变量名称和值的输入规则

使用变量状态 (页 1052)和监控表 (页 1054)标准页面时，请注意以下约定：

- 如果修改整个 DTL 变量值（例如“Data_block_1_.DTL_tag”），则可使用以下 DTL 语法修改数值：DTL#YYYY-MM-DD-HH-MM-SS[.sssssssss]
- 使用指数计数法输入 Real 或 LReal 数据类型的值时：
  - 要输入具有正指数的实数值（如 +3.402823e+25）（Real 或 LReal），请按以下格式之一输入值：
    +3.402823e25
    +3.402823e+25
  - 要输入具有负指数的实数值（如 +3.402823e-25）（Real 或 LReal），请按如下形式输入值：
    +3.402823e-25
  - 确保采用指数计数法的实数值的尾数部分包含一个小数点。如果不包含小数点，则会导致值被改为意外整数值。例如，输入 -1.0e8，而不是 -1e8。
- LReal 值只能为 15 位（小数点位置不限）。输入 15 位以上的值会导致舍入错误。

“变量状态”(Tag status) 和 “监控表”(Watch Table) 页面的限制：

- URL 字符数最多为 2083 个。可以在浏览器的地址栏中查看表示当前页面的 URL。
- 对于字符显示格式，如果实际 CPU 值不是浏览器所解析的有效 ASCII 字符，则页面会显示前缀为美元符号 $ 的字符。

12.9.4 将 CSV 格式的数据日志导入非 USA/UK 版本的 Microsoft Excel 中

数据日志文件采用逗号分隔值 (CSV) 文件格式。 如果系统正在运行 USA 或 UK 版本的 Excel，则可从 “数据日志”(Data Logs) 页面直接用 Excel 打开这些文件。 但是，在其它国家/地区，由于逗号经常出现在数字记数法中，因此并未广泛使用这种格式。

要打开已保存的数据日志文件，请对非 USA/UK 版本的 Excel 执行以下步骤：

1. 打开 Excel 并创建空白工作簿。
2. 从 “数据 > 导入外部数据”(Data > Import External Data) 菜单中选择 “导入数据”(Import Data) 命令。
3. 导航到要打开的数据日志文件并将其选中。 将启动 “文本导入向导”(Text Import Wizard)。
4. 在 “文本导入向导”(Text Import Wizard) 中，将 “原始数据类型”(Original data type) 的默认选项从 “固定宽度”(Fixed width) 改为 “带分隔符”(Delimited)。
5. 单击 “下一步”(Next) 按钮。
6. 在 “步骤 2”(Step 2) 对话框中，选中 “逗号”(Comma) 复选框，以将分隔符类型从 “制表符”(Tab) 改为 “逗号”(Comma)。
7. 单击 “下一步”(Next) 按钮。

8. 在“步骤 3”(Step 3) 对话框中，可根据需要将日期格式从“MDY（月/日/年）”(MDY (month/day/year)) 更改为其它格式。
9. 完成“文本导入向导”(Text Import Wizard) 的其余步骤，以导入文件。

通信处理器和 Modbus TCP 13

13.1 使用串行通信接口

以下两个通信模块 (CM, Communication Module) 和一个通信板 (CB, Communication Board) 提供了用于 PtP 通信的接口：

- CM 1241 RS232 (页 1636)
- CM 1241 RS422/485 (页 1637)
- CB 1241 RS485 (页 1633)

最多可以连接三个 CM（类型不限）外加一个 CB，因而总共可提供四个通信接口。将 CM 安装到 CPU 或另一个 CM 的左侧。将 CB 安装在 CPU 的前端。有关模块安装与拆卸的信息，请参见安装指南 (页 66)。

串行通信接口具有以下特征：

- 具有隔离的端口
- 支持点对点协议
- 通过点对点通信处理器指令进行组态和编程
- 通过 LED 显示传送和接收活动
- 显示诊断 LED（仅限 CM）
- 均由 CPU 供电：不必连接外部电源。

请参见通信接口 (页 1622)的技术规范。

LED 指示灯

通信模块有三个 LED 指示灯：

- 诊断 LED (DIAG)：在 CPU 找到通信模块前，诊断 LED 将一直以红色闪烁。CPU 在上电后会检查 CM，并对其进行寻址。诊断 LED 开始以绿色闪烁。这表示 CPU 寻址到 CM，但尚未为其提供组态。将程序下载到 CPU 后，CPU 会将组态下载到组态的 CM。执行下载到 CPU 操作后，通信模块上的诊断 LED 应为绿色常亮。
- 发送 LED (Tx)：从通信端口向外传送数据时，发送 LED 将点亮。
- 接收 LED (Rx)：通信端口接收数据时，该 LED 将点亮。

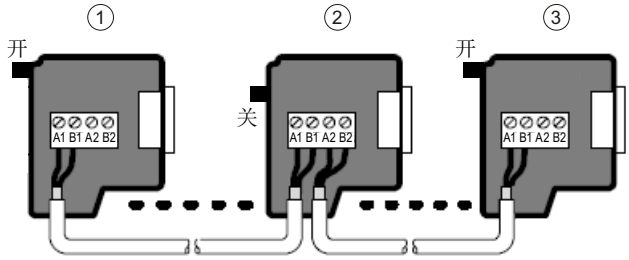
通信板具有发送 LED (TxD) 和接收 LED (RxD)。但没有诊断 LED。

13.2 偏置和端接 RS485 网络连接器

Siemens 提供了一个 RS485 网络连接器 (页 1654)，可用来方便地将多台设备连接到 RS485 网络。 该连接器带有两组端子，分别用于连接输入和输出网络电缆。 连接器还包括用于选择性地偏置和端接网络的开关。

说明

但只能端接和偏置 RS485 网络的两端。 不会端接或偏置这两个终端设备之间的设备。 无电缆屏蔽： 所有位置的金属导线必须接触大约 12 mm (1/2 in)。

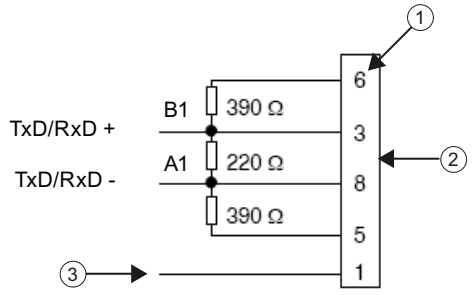
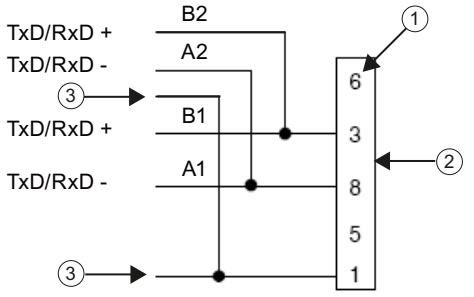


① 开关位置 = 开 (On)： 端接且偏置

② 开关位置 = 关 (Off)： 无端接或偏置

③ 开关位置 = 开 (On)： 端接且偏置

表格 13-1 RS485 连接器的端接和偏置

端接设备（偏置 ON）	非端接设备（偏置 OFF）
	

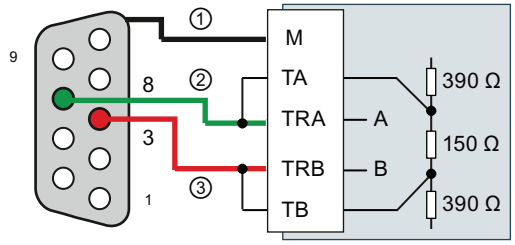
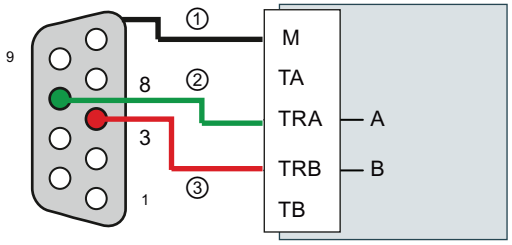
① 引脚编号

② 网络连接器

③ 电缆屏蔽

CB 1241 提供了用于端接和偏置网络的内部电阻。 要终止或偏置连接，应将 TRA 连接到 TA，将 TRB 连接到 TB，以便将内部电阻接到电路中。 CB 1241 没有 9 针连接器。 下表列出了与通信伙伴上的 9 针连接器之间的连接。

表格 13-2　CB 1241 的端接和偏置

端接设备（偏置 ON）	非端接设备（偏置 OFF）
	

① 将 M 连接到电缆屏蔽

② A = TxD/RxD -（绿色线/针 8）

③ B = TxD/RxD +（红色线/针 3）

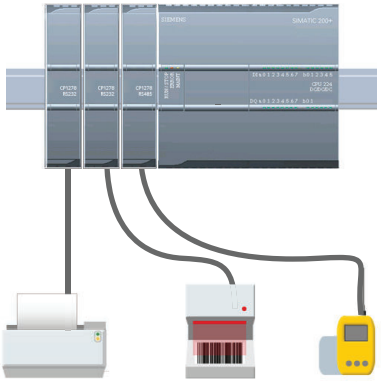
13.3 点对点 (PtP) 通信

CPU 支持下列基于字符的串行协议的点对点通信 (PtP)：

- PtP，自由口 (页 1116)
- PtP，3964(R) (页 1118)
- USS (页 1173)
- Modbus (页 1192)

13.3.1 PtP，自由口通信

支持自由口（即自由构建）协议的 PtP 可提供最大的自由度和灵活性，但需要在用户程序中包含大量的实现。



PtP 可用于实现多种可能性：

- 能够将信息直接发送到外部设备，例如，打印机
- 能够从其它设备（例如，条码阅读器、RFID 阅读器、第三方照相机或视觉系统以及许多其它类型的设备）接收信息
- 能够与其它设备（例如，GPS 设备、第三方照相机或视觉系统、无线调制解调器以及更多其它设备）交换信息（发送和接收数据）

这种类型的 PtP 通信属于串行通信，它使用标准 UART 来支持多种波特率和奇偶校验选项。RS232 和 RS422/485 通信模块 (CM 1241) 以及 RS485 通信板 (CB 1241) 提供了用于执行 PtP 通信的电气接口。

通过 PROFIBUS 或 PROFINET 的 PtP

PtP 允许您使用 PROFINET 或 PROFIBUS 分布式 I/O 机架与各类设备（RFID 阅读器、GPS 设备等）进行通信：

- PROFINET (页 810)：可以将 S7-1200 CPU 的以太网接口连接至 PROFINET 接口模块。可通过机架中 PtP 通信模块以接口模块实现与 PtP 设备的串行通信。
- PROFIBUS (页 986)：在 S7-1200 CPU 机架左边插入 PROFIBUS 通信模块。将 PROFIBUS 通信模块连接至 PROFIBUS 接口模块的机架。可通过机架中 PtP 通信模块以接口模块实现与 PtP 设备的串行通信。

出于这个原因，S7-1200 支持两组 PtP 指令：

- 早期点对点指令 (页 1294)：这些指令适用于 V4.0 版之前的 S7-1200，并且只能通过 CM 1241 通信模块或 CB 1241 通信板进行串行通信。
- 点对点指令 (页 1135)：这些指令具备早期指令的所有功能，并且增添了对使用 PROFINET 和 PROFIBUS 分布式 I/O 的 PtP 通信模块的支持。这些点对点指令允许您通过分布式 I/O 机架访问通信模块。
  要使用这些点对点指令，S7-1200 CM 1241 模块的固件版本不得低于 V2.1。这些模块只能连接至 CPU 左侧的本地机架。也可以使用 CB 1241 的点对点指令。
  通过分布式 I/O 的通信使用以下模块：

站	模块	产品编号	接口
ET 200MP	CM PtP RS232 BA	6ES7540-1AD00-0AA0	RS232
	CM PtP RS232 HF	6ES7541-1AD00-0AB0	RS232
	CM PtP RS422/485 BA	6ES7540-1AB00-0AA0	RS422/RS485
	CM PtP RS422/485 HF	6ES7541-1AB00-0AB0	RS422/RS485
ET 200SP	CM PtP	6ES7137-6AA00-0BA0	RS232 和 RS422/RS485

说明

可以使用点对点指令访问通信板、本地（或左侧）串口模块、PROFINET 串口模块和 PROFIBUS 串口模块。STEP 7 提供早期点对点指令的目的仅是为了支持现有程序。不过，早期命令仍适用于当前的 S7-1200 CPU。无须对之前程序的指令进行转换。

说明

时间同步和 PtP 通信的 CM 模块固件版本要求

如果已经为设备组态中的 Profinet 接口启用了“时间同步 (页 172)”(Time synchronization) 属性中的“CPU 同步设备模块”(CPU synchronizes the modules of the device)，则会将所连接通信模块的固件版本更新到最新可用的版本。针对旧固件版本的通信模块启用模块时间同步时，可能会引起通信问题或错误。

13.3.2 3964(R) 通信

S7-1200 CPU 支持 3964(R) 协议，允许 CM 1241 RS232 模块或 CM 1241 (RS422/485) 模块与采用 3964(R) 协议的通信伙伴进行通信。与上述 PtP 通信中定义消息的特定发送或接收特性有所不同，3964(R) 协议使用以下控制字符禁止严格协议：

- STX 文本起始字符
  要发送的字符串的起始字符
- DLE 数据链路转义字符
  数据传输转换
- ETX 文本结束字符
  要发送的字符串的结束字符
- BBC 块检查字符
- NAK 否定应答

有关协议的完整描述，请参见 S7-300 CP 341 点对点通信、安装和参数分配手册。 (https://support.industry.siemens.com/cs/cn/zh/view/1117397) 手册中描述串行数据传输原理的相关章节。

组态通信模块

要采用 3964(R) 协议与伙伴通信，必须在 STEP 7 的设备组态中添加以下通信模块之一：

- CM 1241 (RS232)
- CM 1241 (RS422/485)

CM 模块的固件版本必须为 V2.2.0 或更高版本。

然后，需要为通信模块组态通信端口 (页 1119)、优先级、和协议参数 (页 1133)。

采用 3964(R) 协议与伙伴通信

将 CM 组态为采用 3964(R) 协议后，使用标准点对点发送和接收指令在 CPU 与其通信伙伴之间传输数据。

CM 将发送指令的 BUFFER 参数中的数据嵌入 3964(R) 协议，然后将数据发送给通信伙伴。

CM 通过 3964(R) 协议从通信伙伴接收数据，删除其中的协议信息，然后将接收指令的 BUFFER 参数中的数据返回。

请参见以下点对点指令：

- Send_P2P（传输发送缓冲区数据） (页 1150)
- Receive_P2P（启用消息接收） (页 1155)

也可以使用早期的点对点发送和接收指令：

- SEND_PTP（传输发送缓冲区数据） (页 1303)
- RCV_PTP（启用消息接收） (页 1305)

13.3.3 组态 PtP 自由口通信

可以使用以下各种方法组态通信接口以进行 PtP 自由口通信：

- 使用 STEP 7 中的设备组态组态端口参数（波特率和奇偶校验）、发送参数和接收参数。CPU 存储设备组态设置，并在循环上电和从 RUN 模式切换到 STOP 模式后应用这些设置。
- 使用 Port_Config (页 1138)、Send_Config (页 1141)和 Receive_Config (页 1143) 指令设置参数。这些指令设置的端口设置在 CPU 处于 RUN 模式期间有效。在切换到 STOP 模式或循环上电后，这些端口设置会恢复为设备组态设置。

组态硬件设备 (页 147)之后，通过选择机架上的某个 CM 或 CB（如果已组态）来组态通信接口的参数。

巡视窗口中的“属性”(Properties) 选项卡显示所选 CM 或 CB 的参数。选择“端口组态”(Port configuration) 以编辑以下参数：

- 波特率
- 奇偶校验
- 每个字符的数据位数
- 停止位的数目
- 流控制（仅限 RS232）
- 等待时间

对于 CM 1241 RS232 和 CB RS485（除仅 CM 1241 RS232 支持的流控制 (页 1121)外），无论是组态 RS232 或 RS485 通信模块还是 RS485 通信板，端口组态参数都是相同的。但是，参数值可以不同。

对于 CM 1241 RS422/485，您还具有下列所示的额外端口组态选项。CM 1241 RS422/485 模块的 422 模式还支持软件流控制。

选择“端口组态”(Port configuration) 以编辑以下 RS422/485 参数：

- “工作模式”(Operating mode)：
  - 全双工 (RS422) 四线制模式（点对点连接）
  - 全双工 (RS422) 四线制模式（多点主站）
  - 全双工 (RS422) 四线制模式（多点从站）
  - 半双工 (RS485) 两线制模式
- “接收线路初始状态”(Receive line initial state)：
  - 无
  - 正向偏置（信号 R(A) 0V、信号 R(B) 5V）

STEP 7 用户程序还可通过 Port_Config 指令 (页 1138)组态端口或更改现有组态。指令主题提供更多关于工作模式和初始线路状态以及其它参数的详细信息。

参数	定义
波特率	波特率的默认值为 9.6 Kbps。有效选项有：300 波特、600 波特、1.2 Kb、2.4 Kb、4.8 Kb、9.6 Kb、19.2 Kb、38.4 Kb、57.6 Kb、76.8 Kb 和 115.2 Kb。
奇偶校验	奇偶校验的默认值是无奇偶校验。有效选项有：无奇偶校验、偶校验、奇校验、传号（奇偶校验位始终设为 1）和空号（奇偶校验位始终设为 0）。
每个字符的数据位数	字符中的数据位数。有效选择为 7 或 8。
停止位的数目	停止位的数目可以是 1 或 2。默认值是 1。
流控制	对于 RS232 通信模块，可以选择硬件或软件流控制 (页 1121)。如果选择硬件流控制，则可以选择是 RTS 信号始终激活还是切换 RTS。如果选择软件流控制，则可以定义 XON 和 XOFF 字符。 RS485 通信接口不支持流控制。CM 1241 RS422/485 模块的 422 模式支持软件流控制。
等待时间	等待时间是指 CM 或 CB 在断言 RTS 后等待接收 CTS 的时间，或者在接收 XOFF 后等待接收 XON 的时间，具体取决于流控制类型。如果在通信接口收到预期的 CTS 或 XON 之前超过了等待时间，CM 或 CB 将中止传送操作并向用户程序返回错误。指定等待时间，以毫秒表示。范围是 0 到 65535 毫秒。
工作模式	选择工作模式 RS422 或 RS485 以及网络组态。
接收线路初始状态	选择偏置选项。有效值为无、正向偏置和反向偏置。反向偏置用于检测电缆断线。

13.3.3.1 管理流控制

流控制是指为了不丢失数据而用来平衡数据发送和接收的一种机制。 流控制可确保传送设备发送的信息量不会超出接收设备所能处理的信息量。 流控制可以通过硬件或软件来实现。 RS232 CM 支持硬件及软件流控制。 RS485 CM 和 CB 不支持流控制。 CM 1241 RS422/485 模块的 422 模式支持软件流控制。 可在组态端口 (页 1119)时或使用 PORT_CFG 指令 (页 1294)指定流控制类型。

硬件流控制通过请求发送 (RTS, Request To Send) 和允许发送 (CTS, Clear To Send) 通信信号来实现。 对于 RS232 CM，RTS 信号从引脚 7 输出，而 CTS 信号通过引脚 8 接收。 RS232 CM 是 DTE（Data Terminal Equipment，数据终端设备）设备，其将 RTS 断言为输出并将 CTS 作为输入来监视。

硬件流控制： RTS 切换

如果为 RS232 CM 启用 RTS 切换的硬件流控制，则模块会将 RTS 信号设置为激活状态以发送数据。 它还会监视 CTS 信号以确定接收设备是否能接收数据。 CTS 信号激活后，只要 CTS 信号保持激活状态，模块便可发送数据。 如果 CTS 信号变为非激活状态，则传送必须停止。

CTS 信号变为激活状态时，传送会继续执行。 如果 CTS 信号在组态的等待时间内未激活，则模块会中止传送并向用户程序返回错误。 在端口组态 (页 1119)中指定等待时间。

对于需要“传送已激活”信号的设备，适合使用 RTS 切换流控制。 例如，无线调制解调器使用 RTS 作为“键”信号来激励无线发送器。 RTS 切换流控制对于标准电话调制解调器不起作用。 对电话调制解调器使用“RTS 始终激活”选项。

硬件流控制： RTS 始终激活

在“RTS 始终激活”模式下，CM 1241 默认情况下将 RTS 设置为激活状态。 设备（如电话调制解调器等）监视来自 CM 的 RTS 信号，并将该信号用作允许发送信号。 调制解调器仅在 RTS 处于激活状态时才向 CM 传送数据，即，电话调制解调器在见到激活的 CTS 信号后发送数据。如果 RTS 处于非激活状态，电话调制解调器不向 CM 传送数据。

要使调制解调器随时都能向 CM 发送数据，请组态“RTS 始终激活”硬件流控制。 CM 因此会将 RTS 信号设置为始终激活。 即使模块无法接受字符，CM 也不会将 RTS 设置为非激活状态。 传送设备必须确保不会使 CM 的接收缓冲区超负荷运行。

利用数据终端就绪 (DTR) 和数据设备就绪 (DSR) 信号

对于这两种硬件流控制类型的任何一种，CM 都会将 DTR 设置为激活状态。 只有当 DSR 信号变为激活状态时，模块才会进行传送。 仅在发送操作开始时评估 DSR 的状态。 如果 DSR 在传送操作开始后变为非激活状态，将不能暂停传送操作。

软件流控制

软件流控制使用消息中的特殊字符来实现流控制。 将组态表示 XON 和 XOFF 的十六进制字符。

XOFF 指示传送必须停止。 XON 指示传送可以继续。 XOFF 和 XON 不得是相同的字符。

传送设备从接收设备收到 XOFF 字符时，将停止传送。 传送设备收到 XON 字符时，传送又继续进行。 如果 CM 在通过端口组态 (页 1119)指定的等待时间内没有收到 XON 字符，它将中止传送并向用户程序返回错误。

软件流控制需要全双工通信，因为在传送过程中接收伙伴必须能够将 XOFF 发送到传送伙伴。 软件流控制只能用于仅包含 ASCII 字符的消息。 二进制协议无法使用软件流控制。

在 CPU 可进行 PtP 自由口通信前，必须组态传送（或发送）消息和接收消息的参数。这些参数决定了在向目标设备传送消息或从目标设备接收消息时的通信工作方式。

13.3.3.2 组态传送（发送）参数

在 CPU 的设备组态中，通过设置所选接口的“传送消息组态”(Transmit message configuration) 属性，来组态通信接口传送数据的方式。

还可以使用 Send_Config (页 1141)指令，从用户程序动态组态或更改传送消息参数。

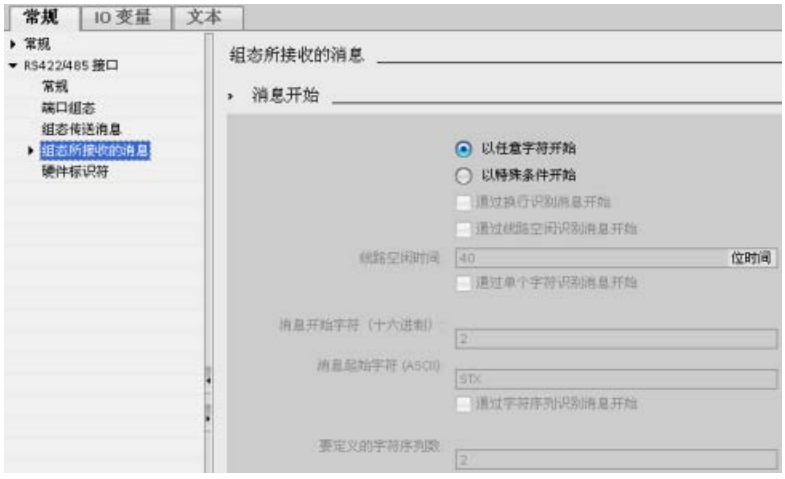
说明

在用户程序中通过 Send_Config 指令设置的参数值会覆盖“传送消息组态”(Transmit message configuration) 属性。 请注意，发生掉电时，CPU 不会保留通过 Send_Config 指令设置的参数。

参数	定义
RTS 接通延时 (RTS On delay)	指定在 RTS 激活后传送启动前需等待的时间。 范围是 0 到 65535 ms，默认值为 0。仅当端口组态 (页 1119)指定的是硬件流控制时，该参数才有效。 在经过 RTS 接通延迟时间后才会评估 CTS。 该参数仅适用于 RS232 模块。
RTS 关断延时 (RTS Off delay)	指定传送完成后 RTS 禁用前需等待的时间。 范围是 0 到 65535 ms，默认值为 0。仅当端口组态 (页 1119)指定的是硬件流控制时，该参数才有效。 该参数仅适用于 RS232 模块。
在消息开始时发送中断 (Send break at message start) 中断期间的位时间数 (Number of bit times in a break)	指定在每条消息开始时，在 RTS 接通延时（如果已组态）已到且 CTS 已激活的情况下先发送中断。 用户指定多少个位的时间构成一个中断，线路在中断期间保持空号状态。 默认值为 12，最大值为 65535，即最长 8 秒的限制。
发送中断后线路空闲信号 (Send idle line after a break) 中断后线路空闲 (Idle line after a break)	指定在消息开始前发送线路空闲信号。 如果组态了中断，则将在中断后发送。“中断后线路空闲”(Idle line after a break) 参数指定多少个位时间构成一次线路空闲，线路在空闲期间保持传号状态。 默认值为 12，最大值为 65535，即最长 8 秒的限制。

13.3.3.3 组态接收参数

在 CPU 的设备组态中，可以组态通信接口接收数据以及识别消息开始和结束的方式。在所选接口的 “接收消息组态”(Receive message configuration) 属性中设置这些参数。



还可以在用户程序中使用 Receive_Config 指令 (页 1143)来动态组态或更改接收消息参数。

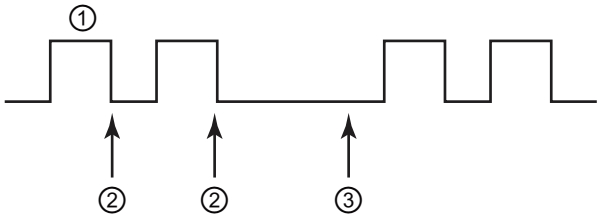
说明

在用户程序中通过 Receive_Config 指令设置的参数值会覆盖“接收消息组态”(Receive message configuration) 属性。请注意，发生掉电或转为 STOP 状态时，CPU 不会保留通过 RCV_CFG 指令设置的参数。

消息开始条件

用户可以决定通信接口识别消息开始的方式。在满足所组态的结束条件之前，开始字符以及组成消息的字符会一直进入接收缓冲区。

可以指定多个开始条件。如果指定多个开始条件，则只有在满足所有开始条件后才认为消息开始。例如，如果用户组态了线路空闲时间和特定开始字符，CM 或 CB 将首先查找要满足的线路空闲时间要求，然后 CM 将查找指定的开始字符。如果收到其它某个字符而不是指定的开始字符，CM 或 CB 将通过再次查找线路空闲时间来重新启动消息开始条件搜索。

参数	定义
以任意字符开始	“任意字符”条件指定，成功接收任何字符都将表示消息开始。该字符是消息中的第一个字符。
线路中断	“线路中断”条件指定在接收中断字符后开始消息接收操作。
线路空闲	“线路空闲”条件指定在接收线路空闲或平静了指定位时间后开始消息接收操作。一旦出现该条件，即启动消息接收。  ① 字符 ② 重启线路空闲定时器 ③ 检测到线路空闲并启动消息接收操作

参数	定义
特殊条件： 通过单个字符识别消息开始	指定通过特殊字符指示消息开始。然后，该字符便成为消息中的第一个字符。在该特定字符前接收到的任何字符都将被丢弃。默认字符是 STX。
特殊条件： 通过字符序列识别消息开始 (Recognize message start with a character sequence)	指定通过最多四个组态序列中的一个特殊字符序列来指示消息开始。可以为每个序列最多指定 5 个字符。对于每个字符位置，可以指定一个特定的十六进制字符，或者指定在序列匹配时忽略该字符（通配符字符）。字符序列中最后一个特定字符终止该开始条件序列。 程序根据组态的开始条件对进入序列进行评估，直到满足开始条件为止。只要满足了开始序列，就会开始评估结束条件。 最多可组态四个特定字符序列。如果几个不同的字符序列都指示消息开始，则使用多序列开始条件。如果与其中一个字符序列相匹配，消息就会开始。

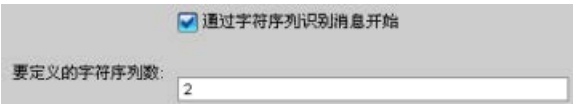
检查开始条件的顺序是：

- 线路空闲
- 线路中断
- 字符或字符序列

检查多个开始条件时，如果有一个条件没有满足，则 CM 或 CB 将从第一个所需的条件开始重新启动检查。CM 或 CB 确定已满足启动条件后，将开始评估结束条件。

示例组态：消息在两个字符序列出现一个时开始

请注意以下消息开始条件组态：




5 字符消息的起始序列
消息开始序列 1
检查字符 1
字符值（十六进制）：: 6A
字符值 (ASCII)：: j
检查字符 2
字符值（十六进制）：: 0
字符值 (ASCII)：: ANY
检查字符 3
字符值（十六进制）：: 0
字符值 (ASCII)：: ANY
检查字符 4
字符值（十六进制）：: 0
字符值 (ASCII)：: ANY
检查字符 5
字符值（十六进制）：: 1C
字符值 (ASCII)：: FS


消息开始序列 2
检查字符 1
字符值（十六进制）：: 0
字符值 (ASCII)：: ANY
检查字符 2
字符值（十六进制）：: 6A
字符值 (ASCII)：: j
检查字符 3
字符值（十六进制）：: 6A
字符值 (ASCII)：: j
检查字符 4
字符值（十六进制）：: 0
字符值 (ASCII)：: ANY
检查字符 5
字符值（十六进制）：: 0
字符值 (ASCII)：: ANY

对于该组态，只要出现其中一个序列，即会满足开始条件：

- 接到一个由五个字符构成的序列，且其第一个字符是 0x6A 而第五个字符是 0x1C 时。对于该组态，位置 2、3 和 4 的字符可以是任意字符。在接到第五个字符后，将开始评估结束条件。
- 接到两个连续的 0x6A 字符（前面为任意字符）时。在这种情况下，会在接到第二个 0x6A 后开始评估结束条件（3 个字符）。第一个 0x6A 前面的字符包含在开始条件中。

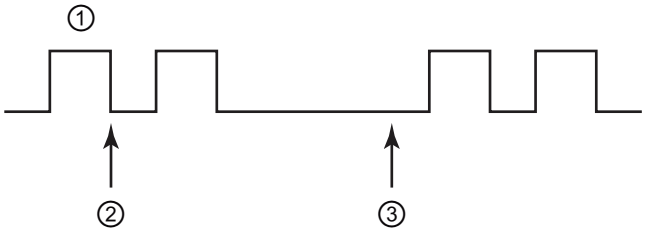
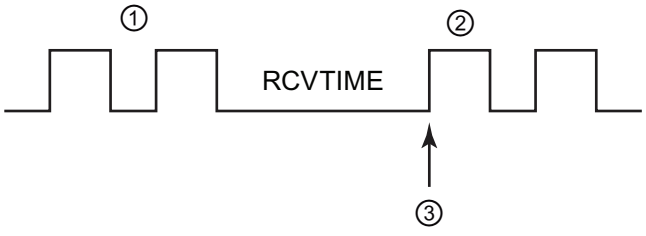
满足该开始条件的实例序列有：

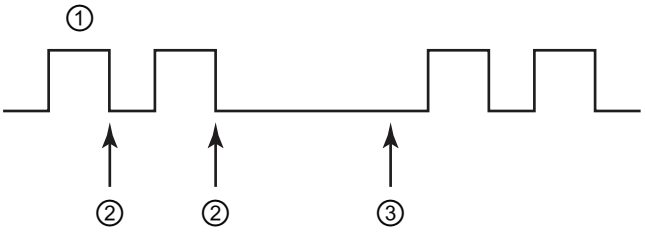
- <任意字符> 6A 6A
- 6A 12 14 18 1C
- 6A 44 A5 D2 1C

消息结束条件

用户还可以组态通信接口识别消息结束的方式。可以组态多个消息结束条件。如果出现组态条件中的任何一个，消息就会结束。

例如，可以采用消息超时 300 ms、字符间超时 40 个位的时间以及最大长度 50 个字节作为消息结束的结束条件。如果接收消息的时间超过 300 ms、任意两个字符间的间隔超过 40 个位的时间或接收到 50 个字节，消息即会结束。

参数	定义
通过消息超时识别消息结束 (Recognize message end by message timeout)	经过了组态的消息结束等待时间后，视为消息结束。消息超时时间从满足开始条件开始计算。默认值是 200 ms，有效范围是 0 到 65535 ms。  ① 接收的字符 ② 满足消息开始条件：消息定时器启动 ③ 消息定时器时间已到并终止消息
通过响应超时识别消息结束 (Recognize message end by response timeout)	如果在接收到有效的开始序列之前超过了组态的响应等待时间，视为消息结束。响应超时时间从传送结束和 CM 或 CB 开始接收操作时开始计算。默认响应超时时间是 200 ms，范围为 0 到 65535 ms。如果在响应时间段 RCVTIME 内没有接收到字符，则会向相应的 RCV_PTP 指令返回错误。响应超时不定义具体结束条件。它仅指定必须在指定时间内成功接收字符。用户必须另组态一个结束条件来指示实际的消息结束。  ① 传送的字符 ② 接收的字符 ③ 必须在该时间之前成功接收到第一个字符。

参数	定义
通过字符间隙识别消息结束 (Recognize message end by inter-character gap)	经过了组态的消息中两个连续字符间的最大超时后，视为消息结束。字符间隙的默认值是 12 个位的时间，最大值是 65535 个位的时间，即最长 8 秒。  ① 接收的字符 ② 重启字符间定时器 ③ 字符间定时器时间已到并终止消息。
通过接收固定数目的字符识别消息结束	在接收到指定数量的字符后，视为消息结束。固定长度的有效范围是 1 到 4096。 请注意，对于 S7-1200，该结束条件仅对 V4.0 或更高版本的 CPU 有效。
通过最大长度识别消息结束 (Recognize message end by max length)	在接收到组态的最大字符数后，视为消息结束。最大长度的有效范围是 1 到 1024。 使用该条件可以防止消息缓冲区超负荷运行错误。如果将该结束条件与超时结束条件结合使用，在出现超时条件时，即使未达到最大长度也会提供所有有效的已接收字符。仅当最大长度已知时，该条件才支持长度可变的协议。
从消息读取消息长度 (Read message length from message)	消息本身指定消息长度。在接收到指定长度的消息后，视为消息结束。以下说明了用于指定和解释消息长度的方法。
通过字符识别消息结束 (Recognize message end with a character)	在接收到指定的字符后，视为消息结束。
通过字符序列识别消息结束 (Recognize message end with a character sequence)	在接收到指定的字符序列后，视为消息结束。可以指定最多由 5 个字符组成的序列。对于每个字符位置，可以指定一个具体的十六进制字符，或者指定在序列匹配时忽略该字符。 结束条件不包括被忽略的前导字符。结束条件包括被忽略的尾随字符。

示例组态：通过字符序列结束消息

请注意以下消息结束条件组态：

在这种情况下，当接收到两个连续的 0x6A 字符（后跟任意两个字符）时，即满足结束条件。0x6A 0x6A 序列前面的字符不是结束字符序列的组成部分。终止结束字符序列时需要在 0x6A 0x6A 序列后面加两个字符。字符位置 4 和 5 中接收的值不相关，但必须接收它们才能满足结束条件。

说明

如果想用字符序列来指示消息的结束，应将该序列放置在最后一个字符位置。在上面的示例中，如果想用 0x6A 0x6A 结束不带任何尾随字符的消息，应在字符位置 4 和 5 中组态 0x6A。

在消息中指定消息长度

选择在消息中包括消息长度这一特殊条件时，必须提供三个用于定义消息长度相关信息的参数。

实际消息结构会因所用的协议而变化。三个参数如下所示：

- n：消息中出现长度说明符的字符位置（从 1 开始）
- 长度大小：长度说明符的字节数（1、2 或 4）
- 长度 m：跟在长度说明符后、不包括在长度计数范围内的字符数

结束字符可不连续。“长度 m”值可用于指定大小不包含在长度字段中的校验和字段的长度。

这些字段位于设备属性的接收消息组态中：

示例 1：假设某条消息是根据以下协议构造的：

STX	Len (n)	长度计数包括字符 3 到 14											
		ADR	PKE		INDEX		PWD		STW		HSW		BCC
1	2	3	4	5	6	7	8	9	10	11	12	13	14
STX	0x0C	xx	xxxx		xxxx		xxxx		xxxx		xxxx		xx

请按以下说明组态该消息的接收消息长度参数：

- n = 2（消息长度从字节 2 开始。）
- 长度大小 = 1（消息长度在一个字节中定义。）
- 长度 m = 0（长度说明符后没有不包括在长度计数中的字符。长度说明符后有 12 个字符。）

在本例中，从 3 到 14（包括 3 和 14）的字符都是 Len (n) 计数的字符。

示例 2：假设另一条消息是根据以下协议构造的：

SD1	Len (n)	Len (n)	SD2	长度计数包括字符 5 到 10						FCS	ED
				DA	SA	FA	数据单元 = 3 个字节				
1	2	3	4	5	6	7	8	9	10	11	12
xx	0x06	0x06	xx	xx	xx	xx	xx	xx	xx	xx	xx

请按以下说明组态该消息的接收消息长度参数：

- n = 3（消息长度从字节 3 开始。）
- 长度大小 = 1（消息长度在一个字节中定义。）
- 长度 m = 3（长度说明符后有 3 个字符不包括在长度计数中。在本实例的协议中，字符 SD2、FCS 和 ED 不包括在长度计数中。其它 6 个字符均包括在长度计数中；因此，长度说明符后总共有 9 个字符。）

在本例中，从 5 到 10（包括 5 和 10）的字符都是 Len (n) 计数的字符。

13.3.4 组态 3964(R) 通信

13.3.4.1 组态 3964(R) 通信端口

您可以使用以下各种方法组态通信接口以进行 3964(R) 通信：

- 使用 STEP 7 中的设备组态组态端口参数。CPU 存储设备组态设置，并在循环上电后应用这些设置。
- 使用 Port_Config (页 1138) 指令设置端口参数。这些指令设置的端口设置在 CPU 处于 RUN 模式期间有效。在循环上电后，这些端口设置会恢复为设备组态设置。

添加通信接口 (页 151)至设备组态后，通过选择机架上的其中一个 CM 来组态通信接口的参数。

巡视窗口中的 “属性”(Properties) 选项卡显示所选 CM 的参数。选择 “端口组态” (Port configuration) 以编辑以下参数：

- “协议”(Protocol)：3964(R)
- “工作模式”(Operating mode)
  （仅 CM 1241 (RS422/485) 模块）
- “接收线路初始状态”(Receive line initial state)
  （仅 CM 1241 (RS422/485) 模块）
- “断线”(Wire break)
  （仅 CM 1241 (RS422/485) 模块）
- 波特率
- 奇偶校验
- 数据位
- 停止位

参数	定义
协议	3964R 或自由口。选择 3964R 以将端口组态为用于 3964(R) 通信
工作模式*	全双工 (RS422) 四线制工作模式（点对点）。（已启用）
接收线路初始状态*	启用下列选项之一： • “无”(None) • “有偏置，R(A)>R(B)>=0V”(Bias with R(A)>R(B)>=0V) • “有偏置，R(B)>R(A)>=0V”(Bias with R(B)>R(A)>=0V)

参数	定义
断线*	启用下列选项之一： ● “无断线检查”(No wire-break check) ● “启用断线检查”(Enable wire-break check)
波特率	波特率的默认值为 9.6 Kbps。有效选项有：300 波特、600 波特、1.2 Kb、2.4 Kb、4.8 Kb、9.6 Kb、19.2 Kb、38.4 Kb、57.6 Kb、76.8 Kb 和 115.2 Kb。
奇偶校验	奇偶校验的默认值是无奇偶校验。有效选项有：无奇偶校验、偶校验、奇校验、传号（奇偶校验位始终设为 1）、空号（奇偶校验位始终设为 0）和任意奇偶校验（发送时奇偶校验位设为 0；接收时忽略奇偶校验错误）。
每个字符的数据位数	字符中的数据位数。有效选择为 7 或 8。
停止位的数目	停止位的数目可以是 1 或 2。默认值是 1。

* 仅 CM 1241 (RS422/485) 模块

13.3.4.2 组态 3964(R) 优先级和协议参数

您可以使用以下各种方法组态通信接口以进行 3964(R) 通信：

- 在通信接口的设备组态中，单击“3964(R) 组态”(3964(R) configuration) 设置优先级并组态协议参数。CPU 存储设备组态设置，并在循环上电后应用这些设置。
- 使用 P3964_Config (页 1148) 指令设置优先级和协议组态参数。这些指令设置的值在 CPU 处于 RUN 模式期间有效。在循环上电后，这些值会恢复为设备组态设置。

巡视窗口中的 “属性”(Properties) 选项卡显示所选 CM 的参数。选择“3964(R) 组态”(3964(R) configuration) 以编辑以下参数：

- “优先级”(Priority)（高或低）
- “协议参数”(Protocol parameters)
  - “带有块检查 (3964R)”(With block check (3964R))
  - “使用默认值”(Use default values)
    “连接尝试次数”(Connection attempts)
    “传输尝试次数”(Transmission attempts)
    “字符延迟时间”(Character delay time)
    “应答延时”(Acknowledgement delay)

参数	定义
优先级	高或低：CM 与通信伙伴必须一个设为高优先级，另一个设为低优先级。
带有块检查 (3964)	如果选中该参数，3964(R) 通信将启用传输安全性，即包含块检查字符 (BCC)。如果取消选中该参数，传输安全性不包含块检查字符。
“使用默认值”(Use default values)	如果选中该参数，3964(R) 的以下协议参数使用默认值： ● “连接尝试次数”(Connection attempts) ● “传输尝试次数”(Transmission attempts) ● “字符延迟时间”(Character delay time) ● “应答延时”(Acknowledgement delay) 如果取消选中该参数，则可以组态这些参数的值。
“连接尝试次数” (Connection attempts)	连接尝试的次数（默认值：6 次连接尝试） 1 到 255
“传输尝试次数” (Transmission attempts)	传输尝试的次数（默认值：6 次连接尝试） 1 到 255
“字符延迟时间” (Character delay time)	字符延迟时间设置（取决于设定的传输速率）（默认值：220 ms）1 ms 到 65535 ms
“应答延时” (Acknowledgement delay)	确认延迟时间设置（取决于设定的传输速率）（默认值：启用块检查时为 2000 ms；禁用块检查时为 550 ms）1 ms 到 65535 ms

说明

除了优先级之外，CM 模块与通信伙伴的协议设置必须相同。

13.3.5 点对点指令

13.3.5.1 点对点指令的公共参数

表格 13-3 PTP 指令的常见输入参数

参数	说明
REQ	许多 PtP 指令使用 REQ 输入在由低电平向高电平切换时启动操作。 REQ 输入在指令执行一次的时间内必须为高电平 (TRUE)，不过 REQ 输入可以在所需时间内一直保持为 TRUE。 在 REQ 输入为 FALSE 时调用指令以便能复位 REQ 输入的历史状态之前，指令不会启动其它操作。 只有这样，指令才能检测低电平到高电平的跳变以启动下一个操作。 将 PtP 指令放入程序时，STEP 7 会提示用户指定背景数据块。 对每个 PtP 指令调用使用一个唯一的背景数据块。 这样可确保每个指令都能正确地处理诸如 REQ 等输入。
PORT	在通信设备组态过程中分配端口地址。 组态后，可以从参数帮助下拉列表中选择默认端口的符号名称。 分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。 端口符号名称在 PLC 变量表的“常量”(Constants) 选项卡中分配。
位时间精度	有几个参数以位时间（通过组态的波特率确定）为单位指定的。 以位时间为单位指定参数可以使参数与波特率无关。 所有以位时间为单位的参数都可以被指定为最多 65535 个位。但 CM 或 CB 可测量的最长时间是 8 秒。

PtP 指令的输出参数 DONE、NDR、ERROR 和 STATUS 可提供 PtP 操作的执行完成状态。

表格 13-4 DONE、NDR、ERROR 和 STATUS 输出参数

参数	数据类型	默认值	说明
DONE	Bool	FALSE	设置为 TRUE 并持续执行一次所需的时间，以表明上一请求已经完成且没有出现错误；否则为 FALSE。
NDR	Bool	FALSE	设置为 TRUE 并持续执行一次所需的时间，以表明请求的动作已经完成且没有出现错误并已接收新的数据；否则为 FALSE。

参数	数据类型	默认值	说明
ERROR	Bool	FALSE	设置为 TRUE 并持续执行一次所需的时间，以表明上一请求已经完成但出现了错误，相应的错误代码在 STATUS 中；否则为 FALSE。
STATUS	Word	0	结果状态： ● 如果设置了 DONE 或 NDR 位，则 STATUS 被设置为 0 或信息代码。 ● 如果设置了 ERROR 位，则 STATUS 被设置为一个错误代码。 ● 如果没有设置以上任何一位，则指令会返回说明功能当前状态的状态结果。 STATUS 在该功能执行期间一直保持其值。

说明

DONE、NDR 和 ERROR 参数仅置位一个执行周期的时间。 程序逻辑必须将临时输出状态值保存在数据锁存器中，以便能检测到后续程序扫描中的状态变化。

表格 13-5　公共条件代码

STATUS (W#16#....)	说明
0000	无错误
7000	功能不忙
7001	功能忙于处理第一个调用。
7002	功能忙于处理后续调用（第一个调用后的轮询）。
8x3A	参数 x 中的指针非法
8070	所有内部实例存储器都被占用，正在执行的并发指令过多
8080	端口号非法。
8081	超时、模块错误或其它内部错误
8082	由于正在后台进行参数化，参数化失败。
8083	缓冲区溢出： CM 或 CB 返回一条接收到的消息，该消息的长度大于长度参数所允许的值。
8090	内部错误：错误的消息长度、错误的子模块或非法消息 请联系客户支持。

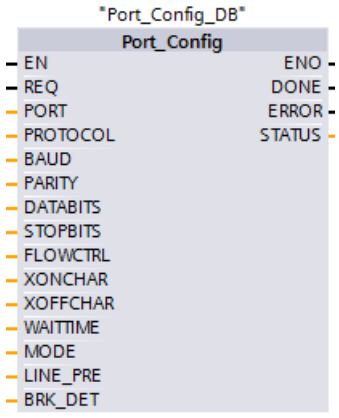
STATUS (W#16#....)	说明
8091	内部错误：参数化消息中的版本错误 请联系客户支持。
8092	内部错误：参数化消息中的记录长度错误 请联系客户支持。

表格 13-6　常见的错误类别

类别说明	错误类别	说明
端口组态	16#81Ax	用于定义常见端口组态错误
传送组态	16#81Bx	用于定义常见传送组态错误
接收组态	16#81Cx 16#82Cx	用于定义常见接收组态错误
传送运行时	16#81Dx	用于定义常见传送运行时错误
接收运行时	16#81Ex	用于定义常见接收运行时错误
信号处理	16#81Fx	用于定义与所有信号处理相关的常见错误
指针错误	16#8p01 到 16#8p51	用 ANY 指针错误，其中“p”是指令的参数编号
嵌入式协议错误	16#848x 16#858x	用于嵌入式协议错误

13.3.5.2 Port_Config（动态组态通信参数）

表格 13-7 Port_Config（端口组态）指令

LAD/FBD	SCL	说明
	"Port_Config_DB"(REQ:=_bool_in_, PORT:=_word_in_, PROTOCOL:=_uint_in_, BAUD:=_uint_in_, PARITY:=_uint_in_, DATABITS:=_uint_in_, STOPBITS:=_uint_in_, FLOWCTRL:=_uint_in_, XONCHAR:=_char_in_, XOFFCHAR:=_char_in_, WAITTIME:=_uint_in_, MODE:=_uint_in_, LINE_PRE:=_uint_in_, BRK_DET:=_uint_in_, DONE=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_);	Port_Config 允许您从用户程序更改端口参数，如波特率。 可以在设备配置属性中设置端口的初始静态组态，或者仅使用默认值。可以在用户程序中执行 Port_Config 指令来更改组态。

[1] STEP 7 会在插入指令时自动创建 DB。

CPU 不永久存储使用 Port_Config 指令设置的值。CPU 从 RUN 模式切换到 STOP 模式和循环上电后，将恢复设备配置中组态的参数。更多信息，请参见组态通信端口 (页 1119)和管理流控制 (页 1121)。

表格 13-8 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	在该输入的上升沿激活组态更改。（默认值：False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。（默认值：0）
PROTOCOL	IN	UInt	0 - 自由口协议（默认值） 1.3964(R) 协议
BAUD	IN	UInt	端口波特率（默认值：6）： 1 = 300 波特，2 = 600 波特，3 = 1200 波特，4 = 2400 波特， 5 = 4800 波特，6 = 9600 波特，7 = 19200 波特，8 = 38400 波特， 9 = 57600 波特，10 = 76800 波特，11 = 115200 波特

参数和类型		数据类型	说明
PARITY	IN	UInt	端口奇偶校验（默认值：1）： 1 = 无奇偶校验，2 = 偶校验，3 = 奇校验，4 = 传号校验，5 = 空号校验
DATABITS	IN	UInt	位/字符（默认值：1）： 1 = 8 个数据位，2 = 7 个数据位
STOPBITS	IN	UInt	停止位（默认值：1）： 1 = 1 个停止位，2 = 2 个停止位
FLOWCTRL*	IN	UInt	流控制（默认值：1）： 1 = 无流控制，2 = XON/XOFF，3 = 硬件 RTS 始终激活，4 = 硬件 RTS 切换
XONCHAR[1]	IN	Char	指定用作 XON 字符的字符。这通常是 DC1 字符 (16#11)。只有启用流控制时，才会评估该参数。（默认值：16#11）
XOFFCHAR[1]	IN	Char	指定用作 XOFF 字符的字符。这通常是 DC3 字符 (16#13)。只有启用流控制时，才会评估该参数。（默认值：16#13）
WAITTIME[1]	IN	UInt	指定在接收 XOFF 字符后等待 XON 字符的时间，或者指定在启用 RTS 后等待 CTS 信号的时间（0 到 65535 ms）。只有启用流控制时，才会评估该参数。（默认值：2000）
MODE[2]	IN	UInt	指定对模块工作模式的选择。 • 0 = 全双工 (RS232) • 1 = 全双工 (RS422) 四线制模式（点对点）；始终启用发送器 • 2 = 全双工 (RS422) 四线制模式（多点主站）；始终启用发送器 • 3 = 全双工 (RS422) 四线制模式（多点从站）；发送时启用发送器 • 4 = 半双工 (RS485) 双线制模式
LINE_PRE	IN	UInt	指定线路未激活（空闲）的条件。对于 RS422 和 RS485 模块，通过向 R(A) 和 R(B) 信号施加偏置电压来指定线路空闲的条件。可以进行下列选择： • 0 = 不偏置（无预置）（默认值） • 1 = 偏置，R(A) > R(B) ≥ 0V；仅限 RS422 • 2 = 偏置，R(B) > R(A) ≥ 0V；RS422 和 RS485

参数和类型		数据类型	说明
BRK_DET	IN	UInt	启用/禁用通信电缆断线检测。启用电缆断线检测功能可在通信电缆未连接到模块时，使模块指示故障。 在 RS422 点对点模式下，只有在施加偏置使 R(A) > R(B) ≥ 0V 并使用“接收线路预置”后，才能进行电缆断线检测。 ● 0 = 无任何电缆断线检测（默认值） ● 1 = 启用电缆断线检测
DONE	OUT	Bool	上一请求已完成且没有出错后，保持为 TRUE 一个执行周期时间
ERROR	OUT	Bool	上一请求已完成但出现错误后，保持为 TRUE 一个执行周期时间
STATUS	OUT	Word	执行条件代码（默认值：0）

[1] Protocol=1（3964(R) 协议）时不适用

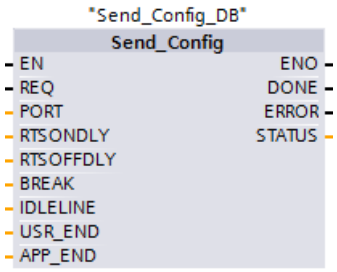
[2] Protocol=1（3964(R) 协议）时仅模式 0 和模式 1 有效，具体取决于 CM 模块是 RS232 模块还是 RS422 模块。

表格 13-9 条件代码

STATUS (W#16#....)	说明
81A0	特定协议不存在。
81A1	特定波特率不存在。
81A2	特定奇偶校验选项不存在。
81A3	特定数据位数不存在。
81A4	特定停止位数不存在。
80A5	特定流控制类型不存在。
81A6	等待时间为 0 且流控制启用
81A7	XON 和 XOFF 是非法值（例如，同一个值）
81A8	块标题中出现错误（例如，块类型错误或块长度错误）
81A9	重新组态被拒绝，因为一个组态正在进行
81AA	RS422/RS485 工作模式无效
81AB	用于断线检测的接收线路预置无效
81AC	RS232 断线处理无效
8280	读取模块时得到否定确认
8281	写入模块时得到否定确认
8282	DP 从站或模块不可用

13.3.5.3 Send_Config（动态组态串行传输参数）

表格 13-10 Send_Config（发送组态）指令

LAD/FBD	SCL	说明
	"Send_Config_DB"(REQ:=_bool_in_, PORT:=_word_in_, RTSONDLY:=_uint_in_, RTSOFFDLY:=_uint_in_, BREAK:=_uint_in_, IDLELINE:=_uint_in_, USR_END:=_string_in_, APP_END:=_string_in_, DONE=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_);	Send_Config 可用于动态组态 PtP 通信端口的串行传输参数。 执行 Send_Config 时，将放弃 CM 或 CB 内所有排队的消息。

[1] STEP 7 会在插入指令时自动创建 DB。

可以在设备配置属性中设置端口的初始静态组态，或者仅使用默认值。 可以在用户程序中执行 Send_Config 指令来更改组态。

CPU 不永久存储使用 Send_Config 指令设置的值。 CPU 从 RUN 模式切换到 STOP 模式和循环上电后，将恢复设备配置中组态的参数。 请参见组态传送（发送）参数 (页 1122)。

表格 13-11 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	在该输入的上升沿激活组态更改。 （默认值： False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。 分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。 端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。 （默认值： 0)
RTSONDLY	IN	UInt	启用 RTS 后执行任何 Tx 数据传输前要等待的毫秒数。 只有启用硬件流控制时，该参数才有效。 有效范围是 0 - 65535 ms。值 0 表示禁用此功能。 （默认值： 0)
RTSOFFDLY	IN	UInt	执行 Tx 数据传输后禁用 RTS 前要等待的毫秒数： 只有启用硬件流控制时，该参数才有效。 有效范围是 0 - 65535 ms。值 0 表示禁用此功能。 （默认值： 0)
BREAK	IN	UInt	该参数指定在各消息开始时将发送指定位时间的中断。 最大值是 65535 个位时间，最多为 8 秒。 值 0 表示禁用该功能。 （默认值： 12)

参数和类型		数据类型	说明
IDLELINE	IN	UInt	该参数指定在各消息开始前线路将保持空闲指定的位时间。 最大值是 65535 个位时间，最多为 8 秒。 值 0 表示禁用该功能。 （默认值： 0)
USR_END*	IN	STRING[2]	指定结束分隔符中的字符和数量。 结束分隔符将结合到发送缓冲区（仅字符），并且为所传送的消息标记结束位置（字符传输在遇到结束分隔符后停止）。 结束分隔符附加在消息的结束位置之后。 • STRING[2,0,xx,yy] – 结束分隔符未使用（默认） • STRING[2,1,xx,yy] – 结束分隔符为单字符 • STRING[2,2,xx,yy] – 结束分隔符为双字符 USR_END 和 APP_END 之一必须长度为零。
APP_END*	IN	STRING[5]	指定要附加到所传送信息上的字符和数量（只有字符会被附加上）。 STRING[5,0,aa,bb,cc,dd,ee] – 结束字符未使用（默认） • STRING[5,1,aa,bb,cc,dd,ee] – 传送一个结束字符 • STRING[5,2,aa,bb,cc,dd,ee] – 传送两个结束字符 • STRING[5,3,aa,bb,cc,dd,ee]– 传送三个结束字符 • STRING[5,4,aa,bb,cc,dd,ee] – 传送四个结束字符 • STRING[5,5,aa,bb,cc,dd,ee] – 传送五个结束字符
DONE	OUT	Bool	上一请求已完成且没有出错后，保持为 TRUE 一个执行周期时间
ERROR	OUT	Bool	上一请求已完成但出现错误后，保持为 TRUE 一个执行周期时间
STATUS	OUT	Word	执行条件代码（默认值： 0）

* 不支持 CM 和 CB 1241；参数必须使用空字符 ("")。

表格 13-12 条件代码

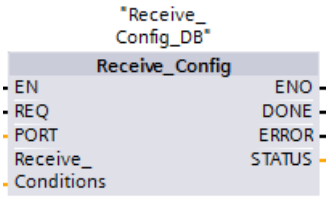
STATUS (W#16#....)	说明
81B0	不允许传送中断组态。 请联系客户支持。
81B1	中断时间大于允许的最大值。
81B2	空闲时间大于允许的最大值。
81B3	块标题错误，例如，块类型错误或块长度错误
81B4	重新组态被拒绝，因为一个组态正在进行
81B5	指定的结束分隔符的数量大于两个，或者结束字符的数量大于五个
81B6	当对固件嵌入式协议进行组态时，发送组态被拒绝。
8280	读取模块时得到否定确认

STATUS (W#16#....)	说明
8281	写入模块时得到否定确认
8282	DP 从站或模块不可用

13.3.5.4 Receive_Config（动态组态串行接收参数）

表格 13-13 Receive_Config（接收组态）指令

LAD/FBD	SCL	说明

	`"Receive_Config_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_uint_in_,`
	
	`Receive_Conditions:=_struct_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		Receive_Config 可用于动态组态 PtP 通信端口的串行接收方参数。该指令可组态表示接收消息开始和结束的条件。执行 Receive_Config 时，将放弃 CM 或 CB 内所有排队的消息。

[1] STEP 7 会在插入指令时自动创建 DB。

可以在设备配置属性中设置通信端口的初始静态组态，或者索性使用默认值。可以在用户程序中执行 Receive_Config 指令来更改组态。

CPU 不永久存储使用 Receive_Config 指令设置的值。CPU 从 RUN 模式切换到 STOP 模式和循环上电后，将恢复设备配置中组态的参数。有关详细信息，请参见主题“组态接收参数 (页 1123)”。

表格 13-14 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	在该输入的上升沿激活组态更改。（默认值：False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。（默认值：0）
CONDITIONS	IN	CONDITIONS	如下文所述，条件数据结构指定消息开始和结束条件。
DONE	OUT	Bool	上一请求已完成且没有出错后，保持为 TRUE 一个扫描周期时间

参数和类型		数据类型	说明
ERROR	OUT	Bool	上一请求已完成但出现错误后，保持为 TRUE 一个扫描周期时间
STATUS	OUT	Word	执行条件代码（默认值：0）

Receive_P2P 指令的开始条件

Receive_P2P 指令使用 Receive_Config 指令指定的组态来确定点对点通信消息的开始和结束。消息开始由开始条件确定。消息开始可以由一个开始条件或开始条件的组合来确定。如果指定多个开始条件，则只有满足所有条件后才能使消息开始。

有关消息开始条件的说明，请参见主题“组态接收参数 (页 1123)”。

参数 CONDITIONS 数据类型结构的第 1 部分（开始条件）

表格 13-15 START 条件的 CONDITIONS 结构

参数和类型		数据类型	说明
STARTCOND	IN	UInt	指定开始条件（默认值：1） • 01H - 开始字符 • 02H - 任意字符 • 04H - 线路中断 • 08H - 线路空闲 • 10H - 序列 1 • 20H - 序列 2 • 40H - 序列 3 • 80H - 序列 4
IDLETIME	IN	UInt	线路空闲超时所需的位时间数。（默认值：40）。仅与线路空闲条件一起使用。0 到 65535
STARTCHAR	IN	Byte	用于开始字符条件的开始字符。（默认值：B#16#2）

参数和类型		数据类型	说明
STRSEQ1CTL	IN	Byte	针对每个字符执行的序列 1 忽略/比较控制：（默认值：B#16#0） 它们是为开始序列中各字符启用的位。 • 01H - 字符 1 • 02H - 字符 2 • 04H - 字符 3 • 08H - 字符 4 • 10H - 字符 5 禁用与某个字符关联的位意味着该序列位置中的任意字符均符合条件。
STRSEQ1	IN	Char[5]	序列 1 开始字符（5 个字符）。默认值：0
STRSEQ2CTL	IN	Byte	针对每个字符执行的序列 2 忽略/比较控制。（默认值：B#16#0）
STRSEQ2	IN	Char[5]	序列 2 开始字符（5 个字符）。默认值：0
STRSEQ3CTL	IN	Byte	针对每个字符执行的序列 3 忽略/比较控制。默认值：B#16#0
STRSEQ3	IN	Char[5]	序列 3 开始字符（5 个字符）。默认值：0
STRSEQ4CTL	IN	Byte	针对每个字符执行的序列 4 忽略/比较控制。默认值：B#16#0
STRSEQ4	IN	Char[5]	序列 4 开始字符（5 个字符），默认值：0

示例

请注意以下所接收的十六进制编码消息：“68 10 aa 68 bb 10 aa 16”以及下表中列出的已组态开始序列。在成功接收到第一个 68H 字符时，开始评估开始序列。在成功接收到第四个字符（第二个 68H）时，开始条件 1 得到满足。只要满足了开始条件，就会开始评估结束条件。

开始序列处理会因各种奇偶校验、成帧或字符间时间错误而终止。由于不再满足开始条件，因而这些错误将导致不会有接收消息。

表格 13-16　开始条件

开始条件	第一个字符	第一个字符 +1	第一个字符 +2	第一个字符 +3	第一个字符 +4
1	68H	xx	xx	68H	xx
2	10H	aaH	xx	xx	xx
3	dcH	aaH	xx	xx	xx
4	e5H	xx	xx	xx	xx

Receive_P2P 指令的结束条件

消息结束由指定的结束条件确定。消息结束由第一次出现的一个或多个已组态结束条件来确定。主题“组态接收参数 (页 1123)”中“消息结束条件”部分介绍了可以在 Receive_Config 指令中组态的结束条件。

可以在设备配置的通信接口的属性中组态结束条件，或者通过 Receive_Config 指令组态结束条件。只要 CPU 从 STOP 模式切换到 RUN 模式，接收参数（开始条件和结束条件）就将恢复为设备配置设置。如果 STEP 7 用户程序执行 Receive_Config，则这些设置将更改为 Receive_Config 的条件。

参数 CONDITIONS 数据类型结构的第 2 部分（结束条件）

表格 13-17 END 条件的 CONDITIONS 结构

参数	参数类型	数据类型	说明
ENDCOND	IN	UInt 0	该参数指定消息结束条件： • 01H - 响应时间 • 02H - 消息时间 • 04H - 字符间隙 • 08H - 最大长度 • 10H - N + LEN + M • 20H - 序列
MAXLEN	IN	UInt 1	最大消息长度：仅当选择最大长度结束条件时使用。1 到 1024 个字节
N	IN	UInt 0	长度域在消息中的字节位置。仅与 N + LEN + M 结束条件一起使用。1 到 1022 个字节
LENGTHSIZE	IN	UInt 0	字节域的大小（1、2 或 4 个字节）。仅与 N + LEN + M 结束条件一起使用。
LENGTHM	IN	UInt 0	指定跟在长度域后、不包含在长度域值内的字符数。该参数仅与 N + LEN + M 结束条件一起使用。0 到 255 个字节

参数	参数类型	数据类型	说明
RCVTIME	IN	UInt 200	指定接收第一个字符所需的等待时间。如果在指定时间内没有成功接收到字符，接收操作将被终止且包含错误。该参数仅与响应时间条件一起使用。（0 到 65535 个位时间，最多 8 秒） 此参数不是消息结束条件，因为在接收到第一个响应字符时评估即终止。由于在预期有响应时却接收不到响应，因此仅就其能够终止接收方操作而言，它又是一个结束条件。必须选择一个单独的结束条件。
MSGTIME	IN	UInt 200	指定在接收到第一个字符后完成接收整条消息所需的等待时间。只有选择了消息超时条件时，才会使用该参数。（0 到 65535 毫秒）
CHARGAP	IN	UInt 12	指定字符间的位时间数。如果字符间的位时间数超出指定值，则结束条件得到满足。该参数仅与字符间隙条件一起使用。（0 到 65535 个位时间，最多 8 秒）
ENDSEQ1CTL	IN	Byte B#16#0	针对每个字符执行的序列 1 忽略/比较控制： 它们是为结束序列中各字符启用的位。字符 1 是位 0，字符 2 是位 1，依此类推，字符 5 是位 4。禁用与某个字符关联的位意味着该序列位置中的任意字符均符合条件。
ENDSEQ1	IN	Char[5] 0	序列 1 开始字符（5 个字符）

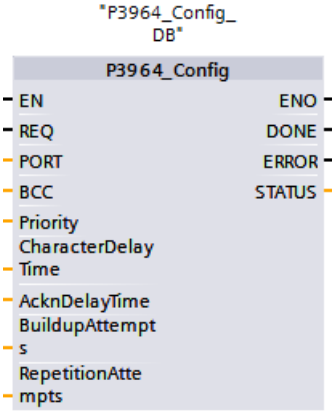
表格 13-18　条件代码

STATUS (W#16#....)	说明
81C0	所选开始条件非法
81C1	所选结束条件非法；未选择结束条件
81C2	启用了接收中断，但不允许此操作。
81C3	启用了最大长度结束条件，最大长度是 0 或大于 1024。
81C4	启用了计算长度，但 N >= 1023。
81C5	启用了计算长度，但长度不是 1、2 或 4。
81C6	启用了计算长度，但 M 值大于 255。

STATUS (W#16#....)	说明
81C7	启用了计算长度，但计算长度大于 1024。
81C8	启用了响应超时，但响应超时为零。
81C9	启用了字符间隙超时，但该字符间隙超时为零。
81CA	启用了线路空闲超时，但该线路空闲超时为零。
81CB	启用了结束序列，但所有字符均“不相关”。
81CC	启用了开始序列（4 个中的任何一个），但所有字符均“不相关”。
81CD	关于接收消息覆盖保护选择无效的错误
81CE	STOP 至 RUN 转换的接收消息缓冲区处理选择无效错误
81CF	块标题错误，例如，块类型错误或块长度错误
8281	写入模块时得到否定确认
8282	DP 从站或模块不可用
82C0	重新组态被拒绝，因为一个组态正在进行
82C1	为模块可缓冲消息数量指定的值大于最大允许值。
82C2	当对固件嵌入式协议进行组态时，接收组态被拒绝。
8351	数据类型不允许用于此 Variant 指针

13.3.5.5 P3964_Config（组态 3964(R) 协议）

表格 13-19 P3964_Config（组态 3964(R) 协议）指令

LAD/FBD	SCL	说明
	"P3964_Config_DB"(REQ:=_bool_in_, PORT:=_uint_in_, BCC:=_usint_in, Priority:= _usint_in, CharacterDelayTime:=_uint_in, AcknDelayTime:= _uint_in, BuildupAttempts:=_usint_in_, RepetitionAttempts:=_usint_in_, DONE=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_);	P3964_Config 允许您在运行期间更改优先级和协议参数。 可以在设备配置属性中设置端口的初始静态组态，或者仅使用默认值。可以在用户程序中执行 P3964_Config 指令来更改组态。

[1] STEP 7 会在插入指令时自动创建 DB。

CPU 不永久存储使用 P3964_Config 指令设置的值。CPU 循环上电后，将恢复设备组态中组态的参数。详细信息请参见组态 3964(R) 通信优先级和协议参数 (页 1133)。

表格 13-20 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	在该输入的上升沿激活组态更改。（默认值：False）
PORT	IN	UInt	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。（默认值：0）
BCC	IN	USInt	激活/取消激活块使用 • 0 = 不带块检查 • 1 = 带有块检查
Priority	IN	UInt	优先级选择 • 0 = 低优先级 • 1 = 高优先级 CM 的优先级必须和通信伙伴相反。
CharacterDelayTime	IN	UInt	字符延迟时间设置（取决于设定的传输速率）（默认值：220 ms） 1 ms 到 65535 ms
AcknDelayTime	IN	UInt	确认延迟时间设置（取决于设定的传输速率）（默认值：2000 ms） 1 ms 到 65535 ms
BuildupAttempts	IN	UInt	连接尝试的次数（默认值：6 次连接尝试） 1 到 255
RepetitionAttempts	IN	UInt	传输尝试的次数（默认值：6 次连接尝试） 1 到 255
DONE	OUT	Bool	上一请求已完成且没有出错后，保持为 TRUE 一个执行周期时间
ERROR	OUT	Bool	上一请求已完成但出现错误后，保持为 TRUE 一个执行周期时间
STATUS	OUT	Word	执行条件代码（默认值：0）

表格 13-21 条件代码

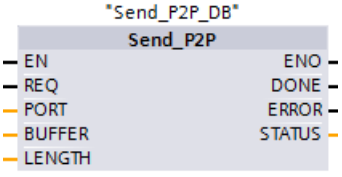
STATUS (W#16#....)	说明
16#8380	参数分配错误：“字符延迟时间”(Character delay time) 的值无效。
16#8381	参数分配错误：“响应超时”(Response timeout) 的值无效。

STATUS (W#16#....)	说明
16#8382	参数分配错误：“优先级”(Priority) 的值无效。
16#8383	参数分配错误：“块检查”(Block check) 的值无效
16#8384	参数分配错误：“连接尝试”(Connection attempts) 的值无效。
16#8385	参数分配错误：“传输尝试”(Transmission attempts) 的值无效。
16#8386	运行错误：连接尝试的次数超出限定
16#8387	运行错误：传输尝试的次数超出限定
16#8388	运行错误：“块检查字符”(Block check character) 错误。 块检查字符的内部计算值和通信伙伴在连接端点处接收到的块检查字符不符。
16#8389	运行错误：等待空闲接收缓冲区时接收到无效字符
16#838A	运行错误：接收时发生逻辑错误。 接收到 DLE 后，又接收到随机字符（除了 DLE 或 ETX）。
16#838B	运行错误：字符延迟时间超出
16#838C	运行错误：空闲接收缓冲区的等待时间已开始。
16#838D	运行错误：帧重复未在 NAK 后 4 秒内开始
16#838E	运行错误：在空闲模式下，接收到一个或多个字符（非 NAK 或 STX）。
16#838F	运行错误：初始化冲突 - 两个通信伙伴均设置了高优先级
16#8391	参数分配错误：由于设置了自由口，拒绝 3964 组态数据

13.3.5.6 Send_P2P（传输发送缓冲区数据）

表格 13-22 Send_P2P（发送点对点数据）指令

LAD/FBD	SCL	描述

	`"Send_P2P_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_word_in_,`
	`BUFFER:=_variant_in_,`
	`LENGTH:=_uint_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		Send_P2P 用于启动数据传输，并将分配的缓冲区传送到通信接口。在 CM 或 CB 块以指定波特率发送数据的同时，CPU 程序会继续执行。仅一个发送操作可以在某一给定时间处于未决状态。如果在 CM 或 CB 已经开始传送消息时执行第二个 Send_P2P，CM 或 CB 将返回错误。

[1] STEP 7 会在插入指令时自动创建 DB。

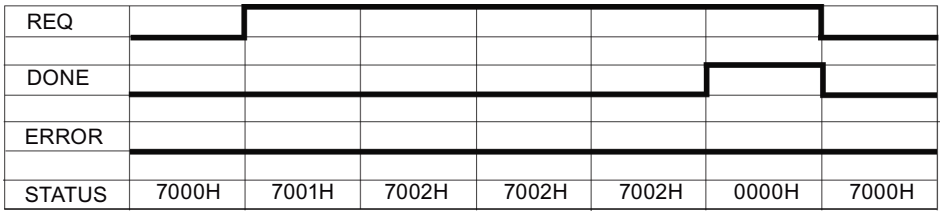
表格 13-23 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	在该传送使能输入的上升沿激活所请求的传送。这会启动将缓冲区数据传送到点对点通信接口。（默认值：False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量” (System constants) 选项卡中分配。（默认值：0）
BUFFER	IN	Variant	该参数指向传送缓冲区的起始位置。（默认值：0） **注：**不支持布尔数据或布尔数组。
LENGTH	IN	UInt	传输的帧长度（字节）（默认值：0） 传输复杂结构时，始终使用长度 0。当长度为 0 时，指令传送整个帧。
DONE	OUT	Bool	上一请求已完成且没有出错后，保持为 TRUE 一个扫描周期时间
ERROR	OUT	Bool	上一请求已完成但出现错误后，保持为 TRUE 一个扫描周期时间
STATUS	OUT	Word	执行条件代码（默认值：0）

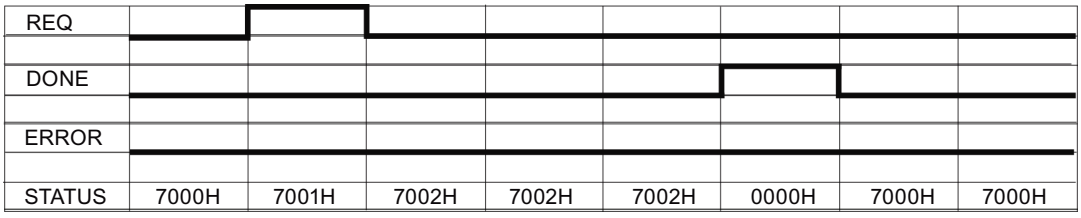
传送操作进行期间，DONE 和 ERROR 输出均为 FALSE。传送操作完成后，DONE 或 ERROR 输出将被设置为 TRUE 以显示传送操作的状态。当 DONE 或 ERROR 为 TRUE 时，STATUS 输出有效。

如果通信接口接受传送数据，则该指令将返回状态值 16#7001。如果 CM 或 CB 仍然忙于传输，则后续的 Send_P2P 执行将返回 16#7002。传送操作完成后，CM 或 CB 将返回传送操作状态 16#0000（如果未出错）。后续执行 REQ 为低电平的 Send_P2P 时，将返回状态 16#7000（不忙）。

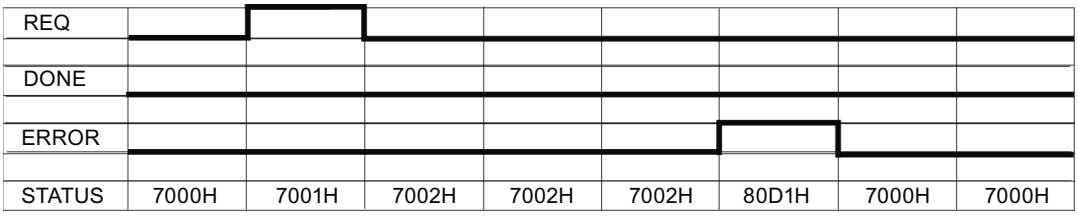
下图显示了输出值与 REQ 的关系。假设定期调用该指令以检查传送过程的状态。在下图中，假设每次扫描都调用该指令（用 STATUS 值表示）。



下图显示通过 REQ 线路脉冲（持续一个扫描周期）启动传送操作时，DONE 和 STATUS 参数是如何仅在一个扫描周期内有效。



下图显示了出错时 DONE、ERROR 和 STATUS 参数之间的关系。



只有 Send_P2P 再次使用相同的背景数据块执行之前，DONE、ERROR 和 STATUS 值才有效。

表格 13-24 条件代码

STATUS (W#16#....)	描述
81D0	传送方激活期间发出新请求
81D1	由于在等待时间内没有 CTS 信号，传送中止
81D2	由于没有来自 DCE 设备的 DSR，传送中止
81D3	由于队列溢出（传送 1024 个字节以上），传送中止
81D5	反向偏置信号（断线检测）
81D6	传输请求被拒绝，因为在传输缓冲区中未找到结束分隔符。
81D7	内部错误/FB 和 CM 同步错误
81D8	因为端口未组态，传输尝试被拒绝
81DF	因下列原因之一，CM 已复位 FB 的接口 ● 模块已重启（循环上电） ● CPU 已达到断点 ● 模块已进行参数设置 在每种情况下，模块都会在状态参数中表示此代码。在收到 SEND_P2P 的第一条记录后，模块会将 Status 和 Error 重置为零。
8281	写入模块时得到否定确认
8282	DP 从站或模块不可用
8301	ANY 指针中存在非法语法 ID

STATUS (W#16#....)	描述
8322	读参数时出现范围长度错误
8324	读参数时发生范围错误
8328	读取参数时发生对齐错误
8332	参数包含大于最大允许编号的 DB 编号（DB 编号错误）。
833A	BUFFER 参数的 DB 不存在。

说明

设置 Profibus 通信的最大记录长度

在使用 CM1243-5 Profibus 主站模块控制使用 RS232、RS422 或 RS485 点对点模块的 ET 200SP 或 ET 200MP Profibus 设备时，需要按如下规定将“max_record_len”数据块变量明确设置为 240：

运行 Port_Config、Send_Config 或 Receive_Config 等组态指令后，在背景数据块中（例如，"Send_P2P_DB".max_record_len）将“max_record_len”设为 240。

只有 Profibus 通信需要明确分配 max_record_len ；Profinet 通信已经使用有效的 max_record_len 值。

LENGTH 和 BUFFER 参数的交互作用

SEND_P2P 指令可以传送的最小数据单位是一个字节。BUFFER 参数决定要传送的数据的大小。BUFFER 参数不接受 Bool 数据类型或 Bool 数组。

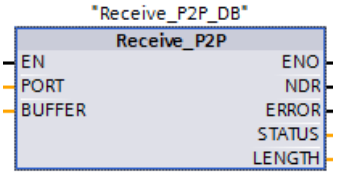
可以将 LENGTH 参数始终设置为 0，从而确保 SEND_P2P 发送 BUFFER 参数表示的整个数据结构。如果仅要传送 BUFFER 参数中的部分数据结构，则可对 LENGTH 进行以下设置：

表格 13-25 LENGTH 和 BUFFER 参数

LENGTH	BUFFER	说明
= 0	未使用	发送 BUFFER 参数中定义的全部数据。当 LENGTH = 0 时，用户无须指定传送字节数。
> 0	基本数据类型	LENGTH 值必须包含此数据类型的字节计数。例如，对于 Word 值，LENGTH 值必须为二。对于 Dword 或 Real，LENGTH 值必须为四。否则，不会传送任何数据并返回错误 8088H。
	结构	LENGTH 值所含字节数可以小于结构的完整字节长度，在这种情况下，指令将只发送结构的头 n 个字节，且该结构来自 BUFFER,n = LENGTH。由于结构的内部字节组织不总是确定不变的，所以可能得到无法预料的结果。在这种情况下，使用值为 0 的 LENGTH 来发送整个结构。
	数组	LENGTH 值必须包含小于或等于数组完整字节长度的字节数，而且还必须为数据元素字节计数的倍数。例如，对于 Word 数组，LENGTH 参数值必须为二的倍数；对于 Real 数组，必须为四的倍数。若指定了 LENGTH，则该指令传输与 LENGTH 值（字节）对应的数组元素数目。例如，如果 BUFFER 包含由 15 个 Dword 构成的数组（总共 60 个字节），LENGTH 指定为 20，则将传送数组中的前五个 Dword。 LENGTH 值必须为数据元素字节数的倍数。否则，STATUS = 8088H，ERROR = 1，且不进行任何传送。
	String	参数 LENGTH 包含要传送的字符数。只传送 String 中相应数量的字符。而不会传送 String 的最大长度和实际长度的字节数。

13.3.5.7 Receive_P2P（启用消息接收）

表格 13-26 Receive_P2P（接收点对点）指令

LAD/FBD	SCL	描述
	"Receive_P2P_DB"(PORT:=_word_in_, BUFFER:=_variant_in_, NDR=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_, LENGTH=>_uint_out_);	Receive_P2P 用于检查 CM 或 CB 中已接收的消息。如果有消息，则会将其从 CM 或 CB 传送到 CPU。如果发生错误，则会返回相应的 STATUS 值。

[1] STEP 7 会在插入指令时自动创建 DB。

表格 13-27 参数的数据类型

参数和类型		数据类型	描述
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。（默认值：0）0)
BUFFER	IN	Variant	该参数指向接收缓冲区的起始位置。该缓冲区应该足够大，可以接收最大长度消息。 不支持布尔数据或布尔数组。（默认值：0)
NDR	OUT	Bool	新数据就绪且操作无错完成后，保持为 TRUE 一个执行周期时间。
ERROR	OUT	Bool	操作已完成但出现错误后，保持为 TRUE 一个执行周期时间。
STATUS	OUT	Word	执行条件代码（默认值：0）0)
LENGTH	OUT	UInt	返回消息的长度（字节）（默认值：0）0)

NDR 或 ERROR 为 TRUE 时，STATUS 值有效。STATUS 值提供 CM 或 CB 中的接收操作终止的原因。它通常是正值，表示接收操作成功且接收过程正常终止。如果 STATUS 值为负数（十六进制值的最高有效位置位），则表示接收操作因错误条件终止，例如，奇偶校验、组帧或超限错误。

每个 PtP 通信接口最多可缓冲 1024 字节。这可以是一个大消息或几个较小的消息。如果 CM 或 CB 中存在多个消息，则 Receive_P2P 指令将返回最早的可用消息。随后执行 Receive_P2P 指令将返回下一个最早的可用消息。

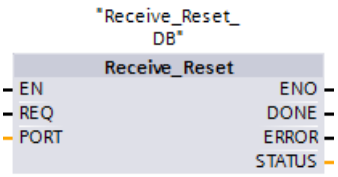
表格 13-28 条件代码

STATUS (W#16#...)	说明
0000	没有提供缓冲区
0094	因接收到最大字符长度，消息被终止
0095	因消息超时，消息被终止
0096	消息因字符间超时而终止
0097	消息因响应超时而终止
0098	因已满足“N+LEN+M”长度条件，消息被终止
0099	因已满足结束序列，消息被终止
8085	LENGTH 参数的值为 0 或大于 1 KB。
8088	LENGTH 参数或收到的长度大于 BUFFER 中指定的范围。
8090	不正确组态信息，错误信息长度，错误子模块，非法信息
81E0	因接收缓冲区已满，消息被终止
81E1	因出现奇偶校验错误，消息被终止
81E2	因组帧错误，消息被终止
81E3	因出现超限错误，消息被终止
81E4	因计算长度超出缓冲区大小，消息被终止
81E5	反向偏置信号（断线检测）
81E6	消息队列已满。报告此错误时将不提供数据。如果发生此情况，模块在无错误数据传送和此错误之间切换。
81E7	内部错误，指令和 CM 之间的同步错误：当检测到顺序错误时置位
81E8	消息被终止，字符间超时在尚未满足消息结束条件时就已过期
81E9	已检出 Modbus CRC 错误（仅限用于支持 Modbus 协议 CRC 生成/校验的模块）
81EA	Modbus 报文过短（仅限用于支持 Modbus 协议 CRC 生成/校验的模块）
81EB	消息被终止，已超过最大信息长度
8201	ANY 指针中存在非法语法 ID
8223	写参数时出现范围长度错误。参数整体或部分位于地址范围之外，或者使用 ANY 指针时位范围长度不为 8 的倍数。

STATUS (W#16#...)	说明
8225	写参数时发生范围错误。参数位于系统函数的非法范围内。
8229	写参数时发生地址对齐错误。引用的参数位于不等于 0 的位地址。
8230	参数位于一个只读的全局 DB 中
8231	参数位于只读背景数据块
8232	参数包含大于最大允许块编号的 DB 编号（DB 编号错误）。
823A	BUFFER 参数的 DB 不存在。
8280	读取模块时得到否定确认
8282	DP 从站或模块不可用

13.3.5.8 Receive_Reset（删除接收缓冲区）

表格 13-29 Receive_Reset（接收方复位）指令

LAD/FBD	SCL	描述
	"Receive_Reset_DB"(REQ:=_bool_in_, PORT:=_word_in_, DONE=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_);	Receive_Reset 可清空 CM 或 CB 中的接收缓冲区。

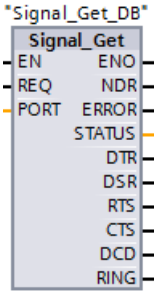
[1] STEP 7 会在插入指令时自动创建 DB。

表格 13-30 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	该使能输入出现上升沿时将激活接收方复位（默认值：False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。 分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。 端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。 （默认值：0） 0)
DONE	OUT	Bool	在一个扫描周期内为 TRUE 时，表示上一个请求已完成且没有错误。
ERROR	OUT	Bool	为 TRUE 时，表示上一个请求已完成但有错误。 此外，该输出为 TRUE 时，STATUS 输出还会包含相关错误代码。
STATUS	OUT	Word	错误代码（默认值：0）

13.3.5.9 Signal_Get（查询 RS-232 信号）

表格 13-31 Signal_Get（获取 RS232 信号）指令

LAD/FBD	SCL	描述
	"Signal_Get_DB"(REQ:=_bool_in_, PORT:=_uint_in_, NDR=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_, DTR=>_bool_out_, DSR=>_bool_out_, RTS=>_bool_out_, CTS=>_bool_out_, DCD=>_bool_out_, RING=>_bool_out_);	Signal_Get 用于读取 RS232 通信信号的当前状态。 该功能仅对 RS232 CM 有效。

[1] STEP 7 会在插入指令时自动创建 DB。

表格 13-32 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	在该输入的上升沿获取 RS232 信号状态值（默认值： False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。 分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。 端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。
NDR	OUT	Bool	新数据就绪且操作无错误地完成时，在一个扫描周期内为 TRUE
ERROR	OUT	Bool	操作已完成但出现错误后，保持为 TRUE 一个扫描周期时间
STATUS	OUT	Word	执行条件代码（默认值：0） 0)
DTR	OUT	Bool	数据终端就绪，模块就绪（输出）。 默认值： False
DSR	OUT	Bool	数据设备就绪，通信伙伴就绪（输入）。 默认值： False
RTS	OUT	Bool	请求发送，模块已做好发送准备（输出）。 默认值： False
CTS	OUT	Bool	允许发送，通信伙伴可以接收数据（输入）。 默认值： False
DCD	OUT	Bool	数据载波检测，接收信号电平（始终为 False，不支持）
RING	OUT	Bool	响铃指示器，来电指示（始终为 False，不支持）

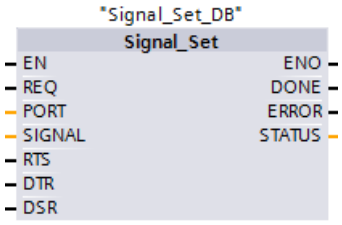
表格 13-33 条件代码

STATUS (W#16#....)	描述
81F0	CM 或 CB 是 RS485 且没有信号可用
81F4	块标题错误，例如，块类型错误或块长度错误
8280	读取模块时得到否定确认
8282	DP 从站或模块不可用

13.3.5.10 Signal_Set（设置 RS-232 信号）

表格 13-34 Signal_Set（设置 RS232 信号）指令

LAD/FBD	SCL	描述

	`"Signal_Set_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_word_in_,`
	`SIGNAL:=_byte_in_,`
	`RTS:=_bool_in_,`
	`DTR:=_bool_in_,`
	`DSR:=_bool_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		Signal_Set 用于设置 RS232 通信信号的状态。
		该功能仅对 RS232 CM 有效。

[1] STEP 7 会在插入指令时自动创建 DB。

表格 13-35 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	在该输入的上升沿启动设置 RS232 信号的操作（默认值： False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。 分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。 端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。 （默认值：0） 0)
SIGNAL	IN	Byte	选择要设置的信号： （允许多个）。 默认值： 0 • 01H = 设置 RTS • 02H = 设置 DTR • 04H = 设置 DSR

参数和类型		数据类型	说明
RTS	IN	Bool	请求发送，模块准备好将值发送到设备（真或假），默认值：False
DTR	IN	Bool	数据终端就绪，模块准备好将值发送到设备（真或假）。 默认值： False
DSR	IN	Bool	数据设备就绪（仅适用于 DCE 型接口），不使用。
DONE	OUT	Bool	上一请求已完成且没有出错后，保持为 TRUE 一个执行周期时间
ERROR	OUT	Bool	上一请求已完成但出现错误后，保持为 TRUE 一个执行周期时间
STATUS	OUT	Word	执行条件代码（默认值：0） 0)

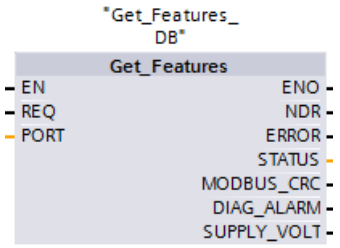
表格 13-36 条件代码

STATUS (W#16#....)	描述
81F0	CM 或 CB 是 RS485 且无法设置任何信号
81F1	因硬件流控制的原因而无法设置信号
81F2	因模块是 DTE 而无法设置 DSR
81F3	因模块是 DCE 而无法设置 DTR
81F4	块标题错误，例如，块类型错误或块长度错误
8280	读取模块时得到否定确认
8281	写入模块时得到否定确认
8282	DP 从站或模块不可用

13.3.5.11 Get_Features

表格 13-37 Get_Features（获取高级功能）指令

LAD/FBD	SCL	描述

	`"Get_Features_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_word_in_,`
	`NDR:=_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`MODBUS_CRC=>_bool_out_,`
	`DIAG_ALARM=>_bool_out_,`
	`SUPPLY_VOLT=>_bool_out);`
		Get_Features 读取模块的高级功能。

[1] STEP 7 会在插入指令时自动创建 DB。

表格 13-38 参数的数据类型

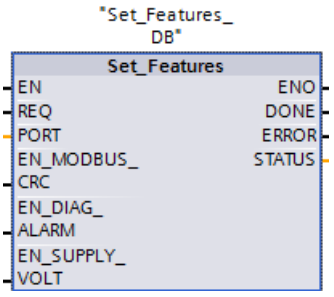
参数和类型		数据类型	描述
REQ	IN	Bool	在该输入的上升沿激活组态更改。（默认值：False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。（默认值：0）
NDR	OUT	Bool	指示新数据已就绪。
ERROR	OUT	Bool	上一请求已完成但出现错误后，保持为 TRUE 一个扫描周期时间
STATUS	OUT	Word	执行条件代码（默认值：0）
MODBUS_CRC *	OUT	Bool	MODBUS CRC 生成和检查
DIAG_ALARM*	OUT	Bool	诊断报警生成
SUPPLY_VOLT *	OUT	Bool	对缺失电源电压 L+ 的诊断可用

* Get_Features 在功能可用时将返回 TRUE (1)，在功能不可用时，返回 FALSE (0)

13.3.5.12 Set_Features

表格 13-39 Set_Features（设置高级功能）指令

LAD/FBD	SCL	说明

	`"Set_Features_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_word_in_,`
	`EN_MODBUS_CRC:=_bool_in_,`
	`EN_DIAG_ALARM:=_bool_in_,`
	
	`EN_SUPPLY_VOLT:=_bool_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		Set_Features 设置模块支持的高级功能。

[1] STEP 7 会在插入指令时自动创建 DB。

表格 13-40 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	在该输入的上升沿激活组态更改。（默认值：False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。（默认值：0）
EN_MODBUS_CRC	IN	Bool	启用 MODBUS CRC 生成和检查： ● 0：CRC 计算调节关闭（默认） ● 1：CRC 计算调节打开 注：仅 V2.1 版 CM、V4.1 版 CPU（带有 CB）和适用于分布式 I/O 的 CM PtP 模块支持此参数。
EN_DIAG_ALARM	IN	Bool	启用诊断报警生成： ● 0：诊断报警关闭 ● 1：诊断报警开启（默认）
EN_SUPPLY_VOLT	IN	Bool	启用对缺失电源电压 L+ 的诊断： ● 0：电源电压诊断已禁用（默认） ● 1：电源电压诊断已启用
DONE	OUT	Bool	指示设置的功能已完成

参数和类型		数据类型	说明
ERROR	OUT	Bool	上一请求已完成但出现错误后，保持为 TRUE 一个扫描周期时间
STATUS	OUT	Word	执行条件代码（默认值：0）

13.3.6 设计 PtP 通信

STEP 7 提供了一些扩展指令，使得用户程序能够使用程序中设计和实现的协议来执行点对点通信。 这些指令分为两类：

- 组态指令
- 通信指令

组态指令

必须先组态通信接口端口以及用于发送数据和接收数据的参数，然后才能通过用户程序执行 PtP 通信。

可以通过设备配置或用户程序中的如下指令，对各个 CM 或 CB 执行端口组态和消息组态：

- Port_Config (页 1138)
- Send_Config (页 1141)
- Receive_Config (页 1143)

通信指令

PtP 通信指令使用户程序能够与通信接口交换消息。 有关使用这些指令传送数据的信息，请参见数据一致性 (页 186)部分。

所有 PtP 功能都是异步运行的。 用户程序可以使用轮询架构来确定传送和接收的状态。 Send_P2P 和 Receive_P2P 可以同时执行。 通信模块和通信板根据需要对传送和接收消息进行缓冲，最大缓冲区大小为 1024 字节。

CM 和 CB 与实际的点对点设备交换消息。 消息协议位于一个缓冲区中，该缓冲区与特定通信端口交换信息。 缓冲区和端口是发送和接收指令的参数：

- Send_P2P (页 1150)
- Receive_P2P (页 1155)

以下指令可用于复位接收缓冲区，以及获取和设置特定的 RS232 信号：

- Receive_Reset (页 1157)
- Signal_Get (页 1158)
- Signal_Set (页 1159)

13.3.6.1 轮询架构

STEP 7 用户程序必须循环/周期性调用 S7-1200 点对点指令以检查收到的消息。 发送轮训可在发送结束时刻即告知用户程序。

轮询架构： 主站

主站的典型轮询顺序如下：

1. Send_P2P (页 1150)指令启动到 CM 或 CB 的传送。
2. 后续扫描期间会执行 Send_P2P 指令以轮询传送完成状态。
3. 当 Send_P2P 指令指示传送完成时，用户代码可以准备接收响应。
4. Receive_P2P (页 1155) 指令反复执行以检查响应。 在 CM 或 CB 收到响应消息后，Receive_P2P 指令将响应复制到 CPU 并指示已接收到新数据。
5. 用户程序随即可处理响应。
6. 转到第 1 步并重复该循环。

轮询架构： 从站

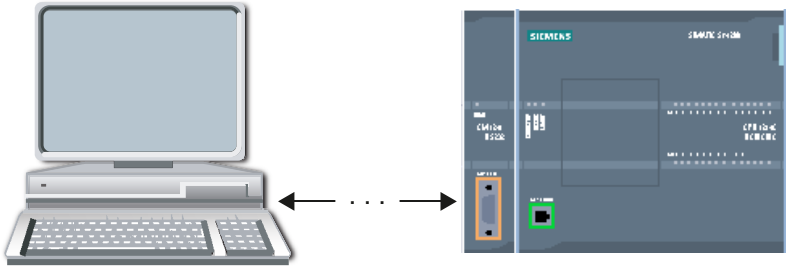
从站的典型轮询顺序如下：

1. 每次扫描用户程序都会执行 Receive_P2P 指令。
2. CM 或 CB 收到请求后，Receive_P2P 指令将指示新数据准备就绪并将请求复制到 CPU 中。
3. 用户程序随即处理请求并生成响应。
4. 使用 Send_P2P 指令将该响应往回发送给主站。
5. 反复执行 Send_P2P 以确保执行传送。
6. 转到第 1 步并重复该循环。

从站在等待响应期间，必须尽量频繁地调用 Receive_P2P ，以便能够在主站超时之前接到来自主站的传送。 要完成该任务，用户程序可以从循环 OB 调用 RCV_PTP，且循环时间应足够大，以便能在超时时间用完之前接到来自主站的传送。 如果将 OB 循环时间设置为可在主站的超时时间内执行该指令两次，则用户程序便可接到主站的传送，而不会错过任何传送。

13.3.7 示例： 点对点通信

在此示例中，S7-1200 CPU 通过 CM 1241 RS232 模块与装有终端仿真器的 PC 通信。 此示例中的点对点组态和 STEP 7 程序说明了 CPU 如何从 PC 接收消息和将该消息回送到 PC。



必须将 CM 1241 RS232 模块的通信接口连接到 PC 的 RS232 接口（通常为 COM1）。 由于这两个端口都是数据终端设备 (DTE)，所以在连接这两个端口时必须交换接收和发送引脚（引脚 2 和 3），可通过以下任何一种方法实现交换：

- 使用 NULL 调制解调器适配器和标准 RS232 电缆交换引脚 2 和 3。
- 使用已交换引脚 2 和 3 的 NULL 调制解调器电缆。 通常可以将电缆两端是否带有两个 9 针 D 型母头连接器作为识别 NULL 调制解调器电缆的依据。

13.3.7.1 组态通信模块

可通过 STEP 7 中的设备组态或通过用户程序指令来组态 CM 1241。 此示例使用设备组态方法。

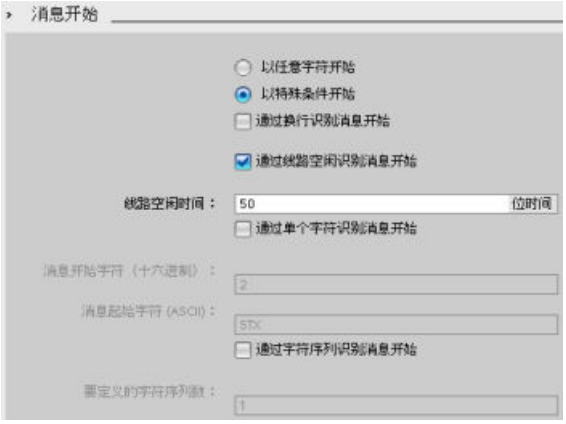
- 端口组态： 在“设备组态”(Device configuration) 中单击 CM 模块的通信端口，然后如下所示组态该端口：

说明

“操作模式”和“接收线路初始状态”的组态设置，只适用于 CM 1241 (RS422/RS485) 模块。 其它 CM 1241 模块没有这些端口组态设置。 请参见组态 RS422 和 RS485 (页 1168)。

- 传送消息组态： 接受传送消息组态的默认值。 在消息开始时将不发送中断信号。

- 接收消息开始组态： 将 CM 1241 组态为在通信线路处于非激活状态至少 50 个位时间（在 9600 波特时约为 5 毫秒 = 50 * 1/9600）时开始接收消息：



- 接收消息结束组态： 将 CM 1241 组态为在最多接收到 100 个字节或换行字符（十进制数 10 或十六进制数 a）时结束消息。 结束序列最多允许序列中具有五个结束字符。 该序列中的第五个字符是换行字符。 前面四个结束序列字符均是“不相关”字符或不选择的字符。 CM 1241 不评估“不相关”字符，但会在零或更多“不相关”字符后面寻找指示消息结束的换行字符。

13.3.7.2 RS422 和 RS485 工作模式

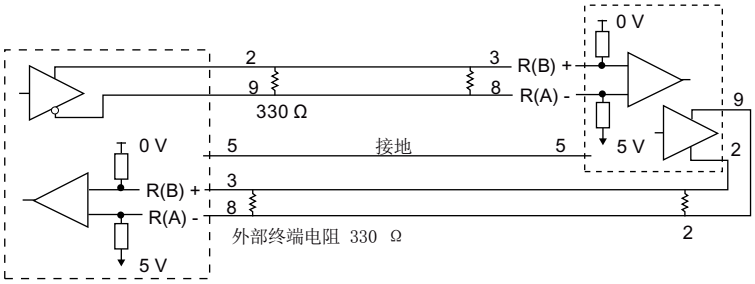
组态 RS422

对于 RS422 模式，有三种工作模式，具体取决于网络组态。 根据网络中的设备选择其中一种工作模式。 接收线路初始状态的不同选择参考了如下所示的详细情况。

- 全双工 (RS422) 四线制模式（点对点连接）： 在网络中有两台设备时选择此选项。 在接收线路初始状态中：
  - 在提供偏置和终端时（第 3 种情况），选择无。
  - 选择正向偏置以使用内部偏置和终端（第 2 种情况）。
  - 选择反向偏置以使用内部偏置和终端，并为两台设备启用电缆断线检测（第 1 种情况）。
- 全双工 (RS422) 四线制模式（多点主站）： 当网络具有一个主站和多个从站时，为主站选择此选项。 在接收线路初始状态中：
  - 在提供偏置和终端时（第 3 种情况），选择无。
  - 选择正向偏置以使用内部偏置和终端（第 2 种情况）。
  - 在此模式下，不能进行电缆断线检测。
- 全双工 (RS422) 四线制模式（多点从站）： 当网络具有一个主站和多个从站时，为所有从站选择此选项。 在接收线路初始状态中：
  - 在提供偏置和终端时（第 3 种情况），选择无。
  - 选择正向偏置以使用内部偏置和终端（第 2 种情况）。
  - 选择反向偏置以使用内部偏置和终端，并为从站启用电缆断线检测（第 1 种情况）。

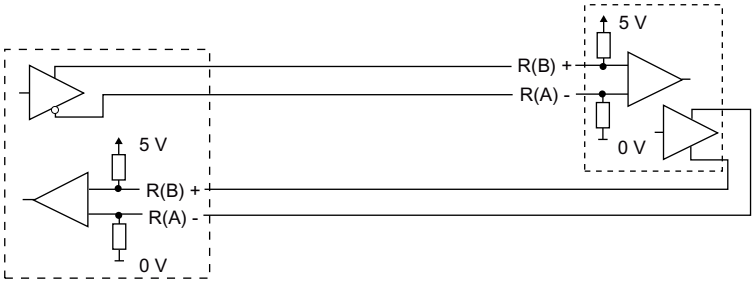
第 1 种情况： RS422，带电缆断线检测

- 工作模式： RS422
- 接收线路初始状态： 反向偏置（有偏置，R(A) > R(B) > 0V）
- 电缆断线： 启用电缆断线检测（发送器始终处于激活状态）



第 2 种情况： RS422，不带电缆断线检测，正向偏置

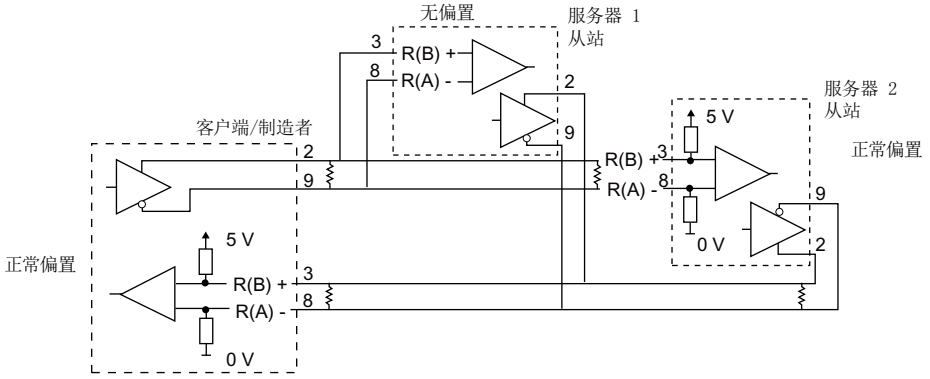
- 工作模式： RS422
- 接收线路初始状态：正向偏置（有偏置，R(B) > R(A) > 0 V）
- 电缆断线： 无电缆断线检测（发送器仅在发送时才启用）



第 3 种情况： RS422： 不带电缆断线检测，无偏置

- 工作模式： RS422
- 接收线路初始状态： 无偏置
- 电缆断线： 无电缆断线检测（发送器仅在发送时才启用）

偏置和终端由用户在网络末端节点处添加。



组态 RS485

对于 RS485 模式，只有一种工作模式。 接收线路初始状态的不同选择参考了如下所示的详细情况。

- 半双工 (RS485) 两线制模式。 在接收线路初始状态中：
  - 在提供偏置和终端时（第 5 种情况），选择无。
  - 选择正向偏置以使用内部偏置和终端（第 4 种情况）。

第 4 种情况： RS485： 正向偏置

- 工作模式： RS485
- 接收线路初始状态：正向偏置（有偏置，R(B) > R(A) > 0 V）

第 5 种情况： RS485： 无偏置（外部偏置）

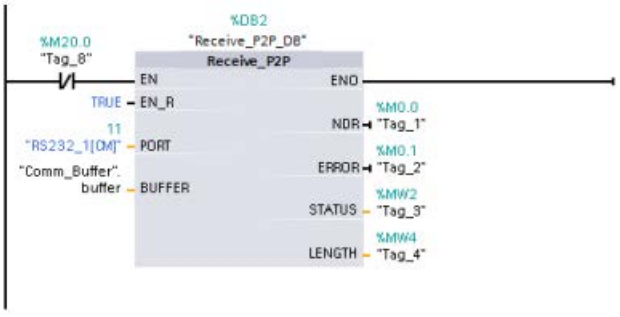
- 工作模式： RS485
- 接收线路初始状态： 无偏置（需要外部偏置）

13.3.7.3 编写 STEP 7 程序

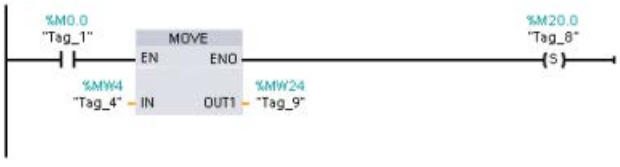
此示例程序使用全局数据块作为通信缓冲区，使用 RCV_PTP 指令 (页 1305)从终端仿真器接收数据，使用 SEND_PTP 指令 (页 1303)向终端仿真器回送缓冲数据。 要对该示例编程，需要添加数据块组态和主程序块 OB1，如下所述。

全局数据块“Comm_Buffer”： 创建一个全局数据块 (DB) 并将其命名为“Comm_Buffer”。 在该数据块中创建一个名为“buffer”，数据类型为 “字节数组 [0 .. 99]”的值。

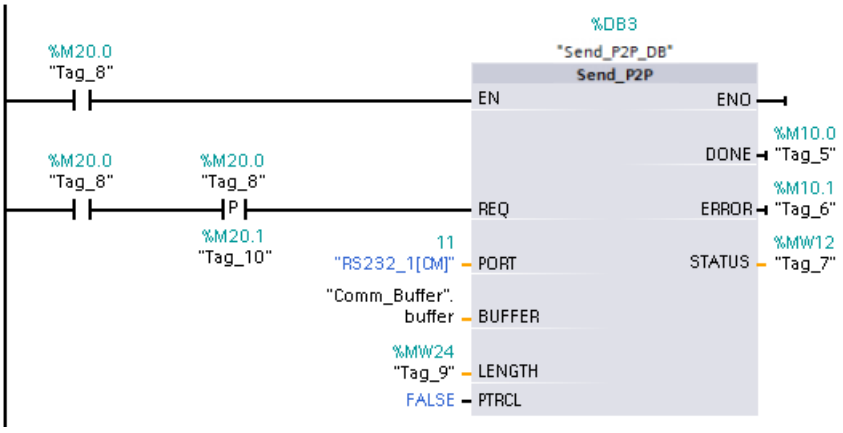
程序段 1： 只要 SEND_PTP 未激活，就启用 RCV_PTP 指令。 在程序段 4 中，MW20.0 中的 Tag_8 在发送操作完成时进行指示，因此是在通信模块相应地准备好接收消息时进行指示。



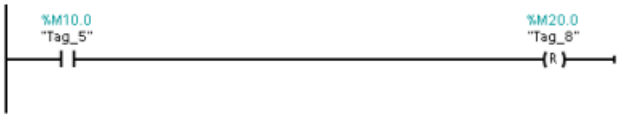
程序段 2： 使用由 RCV_PTP 指令设置的 NDR 值（M0.0 中的 Tag_1）来复制接收到的字节数，并使一个标记（M20.0 中的 Tag_8）置位以触发 SEND_PTP 指令。



程序段 3： M20.0 标记置位时启用 SEND_PTP 指令。 同时还使用此标记将 REQ 输入设置为 TRUE 一个扫描周期时间。 REQ 输入会通知 SEND_PTP 指令要传送新请求。 REQ 输入必须仅在 SEND_PTP 的一个执行周期内设置为 TRUE。 每个扫描周期都会执行 SEND_PTP 指令，直到传送操作完成。 CM 1241 传送完消息的最后一个字节时，传送操作完成。传送操作完成后，DONE 输出（M10.0 中的 Tag_5）将被置位为 TRUE 并持续 SEND_PTP 的一个执行周期。



程序段 4： 监视 SEND_PTP 的 DONE 输出并在传送操作完成时复位传送标记（M20.0 中的 Tag_8）。 传送标记复位后，程序段 1 中的 RCV_PTP 指令可以接收下一条消息。



13.3.7.4 组态终端仿真器

必须设置终端仿真器以支持此示例程序。 几乎可以在 PC 上使用任何终端仿真器，例如，超级终端。 确定终端仿真器处于断开模式后，如下所述编辑各设置：

1. 将终端仿真器设置为使用 PC 上的 RS232 端口（通常为 COM1）。
2. 将端口组态为 9600 波特、8 个数据位、无奇偶校验（无）、1 个停止位和无流控制。
3. 更改终端仿真器设置使其仿真 ANSI 终端。
4. 组态终端仿真器 ASCII 设置，使其在每行后（用户按下 Enter 键后）发送换行信号。
5. 本地回送字符，以便终端仿真器显示输入的内容。

13.3.7.5 运行示例程序

要运行示例程序，请执行以下步骤：

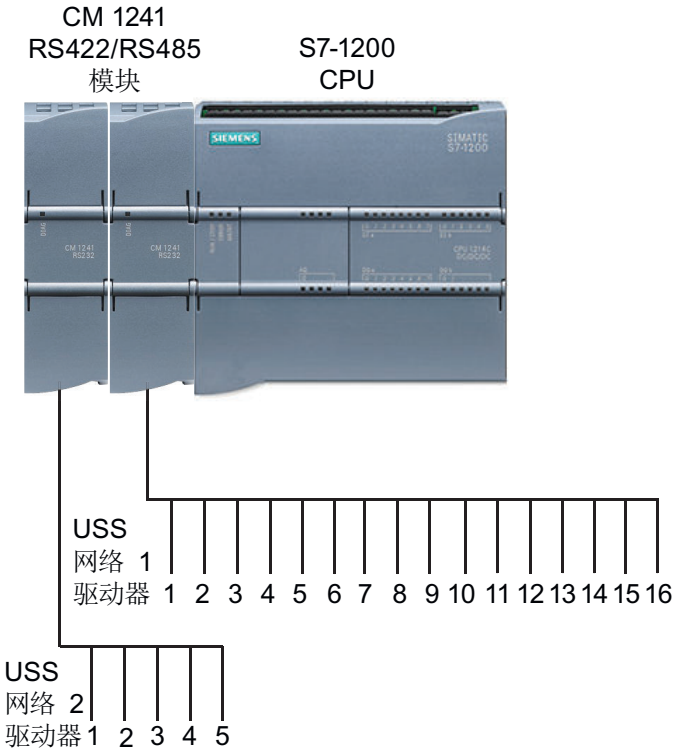
1. 将 STEP 7 程序下载到 CPU 并确保其处于 RUN 模式。
2. 单击终端仿真器上的 “连接”(connect) 按钮以应用组态更改并启动与 CM 1241 的终端会话。
3. 在 PC 中键入字符并按 Enter 键。

终端仿真器会将输入的字符发送到 CM 1241 和 CPU。 然后，CPU 程序将这些字符回送到终端仿真器。

13.4 通用串行接口 (USS) 通信

USS 指令可控制支持通用串行接口 (USS) 的电机驱动器的运行。可以使用 USS 指令通过与 CM 1241 RS485 通信模块或 CB 1241 RS485 通信板的 RS485 连接与多个驱动器通信。一个 S7-1200 CPU 中最多可安装三个 CM 1241 RS422/RS485 模块和一个 CB 1241 RS485 板。每个 RS485 端口最多操作十六台驱动器。

USS 协议使用主从网络通过串行总线进行通信。主站使用地址参数向所选从站发送消息。如果未收到传送请求，从站本身不会执行传送操作。各从站之间无法进行直接消息传送。USS 通信以半双工模式执行。以下 USS 图示显示了一个驱动器应用示例的网络图。



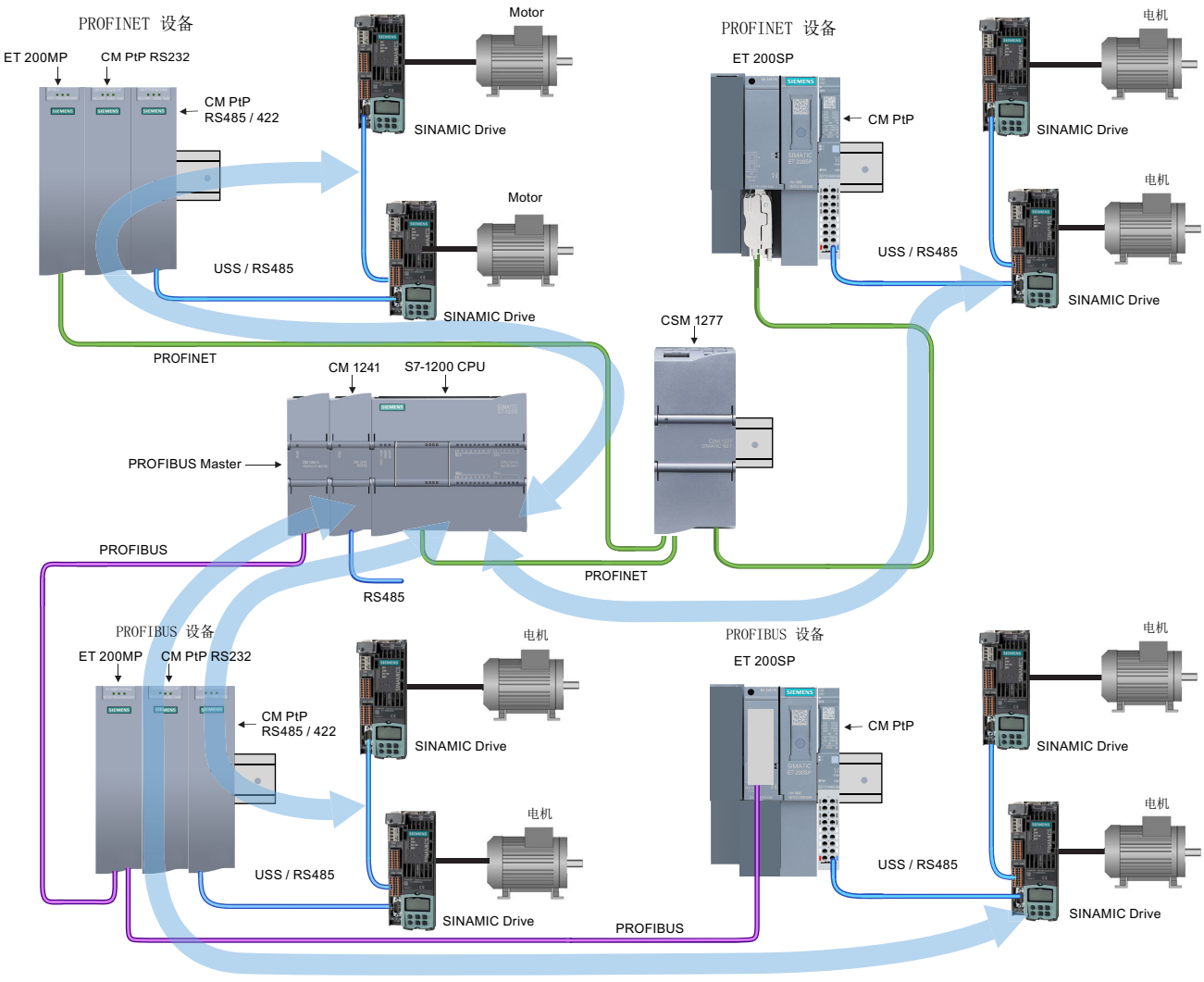
基于 PROFIBUS 或 PROFINET 的 USS 通信

对于 V4.1 版本及以上的 S7-1200 CPU 和 STEP 7 V13 SP1，该 CPU 扩展了 USS 的功能，可以使用 PROFINET 或 PROFIBUS 分布式 I/O 机架与各类设备（RFID 阅读器、GPS 设备和其它）进行通信：

- PROFINET (页 810)：可以将 S7-1200 CPU 的以太网接口连接至 PROFINET 接口模块。可通过机架中 PtP 通信模块以接口模块实现与 PtP 设备的串行通信。
- PROFIBUS (页 986)：在 S7-1200 CPU 机架左边插入 PROFIBUS 通信模块。将 PROFIBUS 通信模块连接至 PROFIBUS 接口模块的机架。可通过机架中 PtP 通信模块以接口模块实现与 PtP 设备的串行通信。

出于这个原因，S7-1200 支持两组 PtP 指令：

- 早期 USS 指令 (页 1314)：这些 USS 指令存在于 S7-1200 的 V4.0 版本之前，并且仅可通过 CM 1241 通信模块或 CB 1241 通信板进行串行通信。
- USS 指令 (页 1179)：这些 USS 指令具备早期指令的所有功能，并且增添了连接 PROFINET 和 PROFIBUS 分布式 I/O 的功能。这些 USS 指令可用于组态分布式 I/O 机架中 PtP 通信模块与 PtP 设备之间的通信。要使用这些 USS 指令，S7-1200 CM 1241 模块的固件版本不得低于 V2.1。



蓝色箭头表示设备间的双向通信流。

说明

用于 S7-1200 的 V4.1 版本时，可以对所有类型的点对点通信使用点对点指令：串行通信、基于 PROFINET 的串行通信和基于 PROFIBUS 的串行通信。STEP 7 提供早期点对点指令的目的仅是为了支持现有程序。早期命令仍适用于所有 S7-1200 CPU。无须对之前程序的指令进行转换。

13.4.1 选择 USS 指令的版本

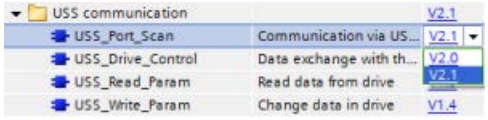
在 STEP 7 中可使用两个版本的 USS 指令：

- 版本 2.0（早期指令）最初在 STEP 7 Basic/Professional V13 中提供。
- 版本 2.1 在 STEP 7 Basic/Professional V13 SP1 或新版本中提供。

可以从兼容性和移植便利性方面考虑，选择将相应的指令版本插入用户程序中。

不能将两个版本的指令用于同一模块，但不同的模块可以使用不同版本的指令。

单击指令树任务卡上的图标可启用指令树的标题和列。



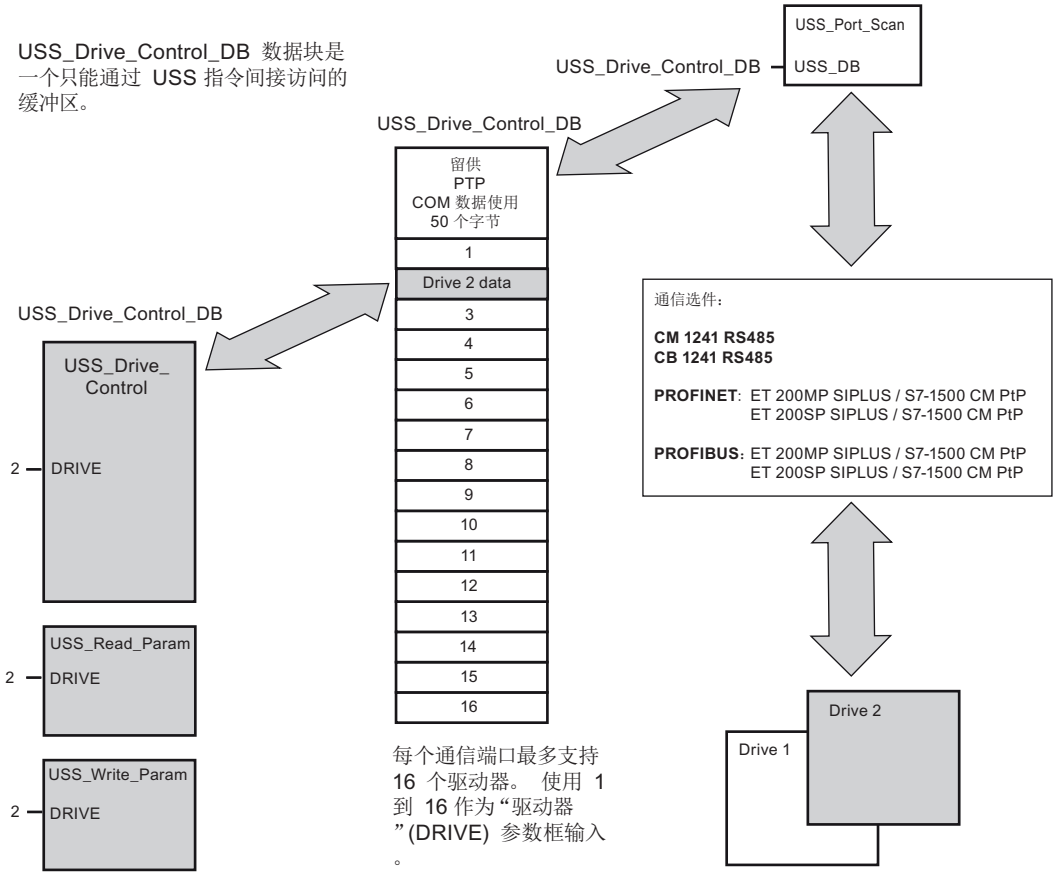
要更改 USS 指令的版本，请从下拉列表中选择相应版本。可以选择一组指令或分别选择各个指令。

使用指令树将 USS 指令放入程序时，将根据所选的 USS 指令在项目树中创建新的 FB 或 FC 实例。在项目树的“PLC_x > 程序块 > 系统块 > 程序资源”(PLC_x > Program blocks > System blocks > Program resources) 下可看到新的 FB 或 FC 实例。

要确认程序中 USS 指令的版本，必须检查项目树的属性而不是程序编辑器中显示的框的属性。选择项目树的 USS FB 或 FC 实例，单击右键，选择“属性”(Properties)，然后选择“信息”(Information) 页查看 USS 指令的版本号。

13.4.2 使用 USS 协议的要求

四条 USS 指令使用两个 FB 和两个 FC 来支持 USS 协议。 一个 USS 网络使用一个 USS_Port_Scan 背景数据块 (DB)。 USS_Port_Scan 背景数据块包含供该 USS 网络中所有驱动器使用的临时存储区和缓冲区。 各 USS 指令共享此数据块中的信息。



连接到一个 RS485 端口的所有驱动器（最多 16 个）是同一 USS 网络的一部分。 连接到另一 RS485 端口的所有驱动器是另一 USS 网络的一部分。 各 USS 网络通过单独的数据块进行管理。 与各 USS 网络相关的所有指令必须共享该数据块。 这包括用于控制各 USS 网络上的所有控制器的所有 USS_Drive_Control、USS_Port_Scan、USS_Read_Param 和 USS_Write_Param 指令。

USS_Drive_Control 指令是一个函数块 (FB)。 在程序编辑器中放置 USS_Drive_Control 指令时，系统将通过“调用选项”(Call options) 对话框提示您为该 FB 分配 DB。 如果对于该 USS 网络而言，它是该程序中的第一条 USS_Drive_Control 指令，则可以接受默认的 DB 分配（或根据需要更改名称），将相应地创建一个新 DB。 但是，如果对于该通道它不是第一条 USS_Drive_Control 指令，则必须使用“调用选项”(Call options) 对话框中的下拉列表选择先前为该 USS 网络分配的 DB 名称。

USS_Drive_Control 指令是一个函数块 (FB)，并且其通过点对点 (PtP) RS485 通信端口处理 CPU 和驱动器之间的实际通信。 每次调用此 FB 可处理与一个驱动器的一次通信。 用户程序必须尽快调用此 FB 以防止与驱动器通信超时。 可在主程序循环 OB 或任何中断 OB 中调用此 FB。

USS_Read_Param 和 USS_Write_Param 指令都是函数 (FC)。 在编辑器中放置这些 FC 时不分配 DB。 而您必须给这些指令的“USS_DB”输入分配合适的 DB 引用。 双击该参数字段，然后单击参数助手图标可查看可用的 DB 名称。

通常，应在循环中断 OB 中调用 USS_Port_Scan FB。该循环中断 OB 的循环时间应设置为最小调用间隔的一半左右（例如，1200 波特的通信应使用 350 ms 或更短的循环时间）。

用户程序通过 USS_Drive_Contro FB 可访问 USS 网络上指定的驱动器。 其输入和输出是驱动器的状态和控制。 如果网络上有 16 个驱动器，则用户程序必须具有至少 16 个 USS_Drive_Control 调用，每个驱动器一个调用。 应该以控制驱动器工作所需的速率调用这些块。

只能在主程序循环 OB 中调用 USS_Drive_Control FB。

⚠ 小心

从 OB 调用 USS 指令时的考虑事项

只能在主程序循环 OB 中调用 USS_Drive_Control、USS_Read_Param 和 USS_Write_Param。可在任何 OB 中调用 USS_Port_Scan FB，通常是在循环中断 OB 中调用。

不要在优先级比 USS_Port_Scan 指令所在 OB 的优先级高的 OB 中使用 USS_Drive_Control、USS_Read_Param 和 USS_Write_Param 指令。 例如，不要将 USS_Port_Scan 放置在主程序循环 OB 中，而将 USS_Read_Param 放置在循环中断 OB 中。如果未能防止 USS_Port_Scan 执行的中断，则会产生意外错误，进而导致人身伤害。

USS_Read_Param 和 USS_Write_Param FC 可读取和写入远程驱动器工作参数。 这些参数控制驱动器的内部运行。 有关这些参数的定义，请参见驱动器手册。 用户程序可包含尽可能多的这些功能，但在任何特定时刻，每个驱动器只能激活一个读或写请求。 只能在主程序循环 OB 中调用 USS_Read_Param 和 USS_Write_Param FC。

计算与驱动器通信所需的时间

与驱动器进行的通信与 S7-1200 扫描周期不同步。 在完成一个驱动器通信事务之前，S7-1200 通常完成了多个扫描。

USS_Port_Scan 间隔是一个驱动器事务所需的时间。 下表列出了各个通信波特率下的最小 USS_Port_Scan 时间间隔。 比 USS_Port_Scan 间隔更频繁地调用 USS_Port_Scan FB 不

会增加事务数。 如果通信错误导致尝试 3 次才能完成事务，则驱动器超时间隔是处理该事务可能花费的时间。 默认情况下，USS 协议库对每个事务最多自动进行 2 次重试。

表格 13-41 计算时间要求

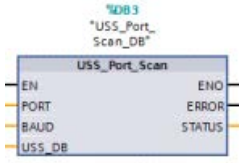
波特率	计算最小 USS_Port_Scan 调用间隔（毫秒）	每个驱动器的驱动器消息间隔超时（毫秒）
1200	790	2370
2400	405	1215
4800	212.5	638
9600	116.3	349
19200	68.2	205
38400	44.1	133
57600	36.1	109
115200	28.1	85

13.4.3 USS 指令

13.4.3.1 USS_Port_Scan（使用 USS 网络编辑通信）

表格 13-42 USS_Port_Scan 指令

LAD/FBD	SCL	说明

	`USS_Port_Scan(`
	`  PORT:=_uint_in_,`
	`  BAUD:=_dint_in_,`
	`  ERROR=>_bool_out_,`
	`  STATUS=>_word_out_,`
	`  USS_DB:=_fbtref_inout_);`
		USS_Port_Scan 指令用于处理 USS 网络上的通信。

表格 13-43 参数的数据类型

参数和类型		数据类型	说明
PORT	IN	Port	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。
BAUD	IN	DInt	用于 USS 通信的波特率。
USS_DB	INOUT	USS_BASE	将 USS_Drive_Control 指令放入程序时创建并初始化的背景数据块的名称。
ERROR	OUT	Bool	该输出为真时，表示发生错误，且 STATUS 输出有效。
STATUS	OUT	Word	请求的状态值指示扫描或初始化的结果。对于有些状态代码，还在“USS_Extended_Error”变量中提供了更多信息。

通常，程序中每个 PtP 通信端口只一个 USS_Port_Scan 指令，且每次调用该函数块 (FB) 都会处理与单个驱动器的通信。与同一个 USS 网络和 PtP 通信端口相关的所有 USS 功能都必须使用同一个背景数据块。

用户程序执行 USS_Port_Scan 指令的次数必须足够多，以防止驱动器超时。通常从循环中断 OB 调用 USS_Port_Scan 以防止驱动器超时并确保 USS_Drive_Control 调用可使用最新的 USS 数据更新内容。

说明

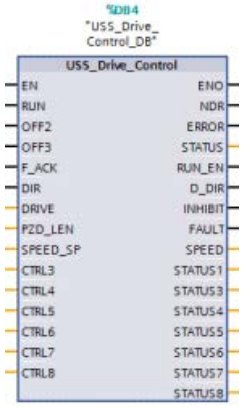
对 CB 1241 使用 USS 协议库和 USS_Port_Scan 指令时，必须将 LINE_PRE 数据块变量的值设置为 0（无初始状态）。LINE_PRE 数据块变量的默认值 2 导致 USS_Port_Scan 指令返回错误值 16#81AB。LINE_PRE 数据块变量在与 USS_Port_Scan 指令相关的数据块（通常名为 USS_Port_Scan_DB）中。

确保将 LINE_PRE 的起始值更改为 0（零）。

13.4.3.2 USS_Drive_Control（与驱动器交换数据）

表格 13-44 USS_Drive_Control 指令

LAD/FBD	SCL	说明

	`"USS_Drive_Control_DB"(`
	`RUN:=_bool_in_,`
	`OFF2:=_bool_in_,`
	`OFF3:=_bool_in_,`
	`F_ACK:=_bool_in_,`
	`DIR:=_bool_in_,`
	`DRIVE:=_usint_in_,`
	`PZD_LEN:=_usint_in_,`
	`SPEED_SP:=_real_in_,`
	`CTRL3:=_word_in_,`
	`CTRL4:=_word_in_,`
	`CTRL5:=_word_in_,`
	`CTRL6:=_word_in_,`
	`CTRL7:=_word_in_,`
	`CTRL8:=_word_in_,`
	`NDR=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`RUN_EN=>_bool_out_,`
	`D_DIR=>_bool_out_,`
	`INHIBIT=>_bool_out_,`
	`FAULT=>_bool_out_,`
	`SPEED=>_real_out_,`
	`STATUS1=>_word_out_,`
	`STATUS3=>_word_out_,`
	`STATUS4=>_word_out_,`
	`STATUS5=>_word_out_,`
	`STATUS6=>_word_out_,`
	`STATUS7=>_word_out_,`
	`STATUS8=>_word_out_);`
		USS_Drive_Control 指令通过创建请求消息和解释驱动器响应消息与驱动器交换数据。每个驱动器应使用一个单独的函数块，但与一个 USS 网络和 PtP 通信端口相关的所有 USS 函数必须使用同一个背景数据块。必须在放置第一个 USS_Drive_Control 指令时创建 DB 名称，然后引用初次指令使用时创建的 DB。
		STEP 7 会在插入指令时自动创建该 DB。

[1] LAD 和 FBD：通过单击功能框的底部，可展开该功能框，以显示所有参数。灰显的参数引脚可选，不需要进行参数分配。

表格 13-45 参数的数据类型

参数和类型		数据类型	说明
RUN	IN	Bool	驱动器起始位：该输入为真时，将使驱动器以预设速度运行。如果在驱动器运行时 RUN 变为假，电机将减速直至停止。这种行为不同于切断电源 (OFF2) 或对电机进行制动 (OFF3)。
OFF2	IN	Bool	电气停止位：该位为假时，将使驱动器在无制动的情况下自然停止。

参数和类型		数据类型	说明
OFF3	IN	Bool	快速停止位：该位为假时，将通过制动的方式使驱动器快速停止，而不只是使驱动器逐渐自然停止。
F_ACK	IN	Bool	故障确认位：设置该位以复位驱动器上的故障位。清除故障后会设置该位，以告知驱动器不再需要指示前一个故障。
DIR	IN	Bool	驱动器方向控制：设置该位以指示方向为向前（对于正 SPEED_SP）。
DRIVE	IN	USInt	驱动器地址：该输入是 USS 驱动器的地址。有效范围是驱动器 1 到驱动器 16。
PZD_LEN	IN	USInt	字长度：这是 PZD 数据的字数。有效值为 2、4、6 或 8 个字。默认值为 2。
SPEED_SP	IN	Real	速度设定值：这是以组态频率的百分比表示的驱动器速度。正值表示方向向前（DIR 为真时）。有效范围是 200.00 到 -200.00。
CTRL3	IN	Word	控制字 3：写入驱动器上用户可组态参数的值。必须在驱动器上组态该参数。（可选参数）
CTRL4	IN	Word	控制字 4：写入驱动器上用户可组态参数的值。必须在驱动器上组态该参数。（可选参数）
CTRL5	IN	Word	控制字 5：写入驱动器上用户可组态参数的值。必须在驱动器上组态该参数。（可选参数）
CTRL6	IN	Word	控制字 6：写入驱动器上用户可组态参数的值。必须在驱动器上组态该参数。（可选参数）
CTRL7	IN	Word	控制字 7：写入驱动器上用户可组态参数的值。必须在驱动器上组态该参数。（可选参数）
CTRL8	IN	Word	控制字 8：写入驱动器上用户可组态参数的值。必须在驱动器上组态该参数。（可选参数）
NDR	OUT	Bool	新数据就绪：该位为真时，表示输出包含新通信请求数据。
ERROR	OUT	Bool	出现错误：此参数为真时，表示发生错误，STATUS 输出有效。其它所有输出在出错时均设置为零。仅在 USS_Port_Scan 指令的 ERROR 和 STATUS 输出中报告通信错误。
STATUS	OUT	Word	请求的状态值指示扫描的结果。这不是从驱动器返回的状态字。
RUN_EN	OUT	Bool	运行已启用：该位指示驱动器是否在运行。
D_DIR	OUT	Bool	驱动器方向：该位指示驱动器是否正在向前运行。
INHIBIT	OUT	Bool	驱动器已禁止：该位指示驱动器上禁止位的状态。
FAULT	OUT	Bool	驱动器故障：该位指示驱动器已注册故障。用户必须解决问题，并且在该位被置位时，设置 F_ACK 位以清除此位。

参数和类型		数据类型	说明
SPEED	OUT	Real	驱动器当前速度（驱动器状态字 2 的标定值）：以组态速度百分数形式表示的驱动器速度值。
STATUS1	OUT	Word	驱动器状态字 1：该值包含驱动器的固定状态位。
STATUS3	OUT	Word	驱动器状态字 3：该值包含驱动器上用户可组态的状态字。
STATUS4	OUT	Word	驱动器状态字 4：该值包含驱动器上用户可组态的状态字。
STATUS5	OUT	Word	驱动器状态字 5：该值包含驱动器上用户可组态的状态字。
STATUS6	OUT	Word	驱动器状态字 6：该值包含驱动器上用户可组态的状态字。
STATUS7	OUT	Word	驱动器状态字 7：该值包含驱动器上用户可组态的状态字。
STATUS8	OUT	Word	驱动器状态字 8：该值包含驱动器上用户可组态的状态字。

首次执行 USS_Drive_Control 时，将在背景数据块中初始化由 USS 地址（参数 DRIVE）指示的驱动器。完成初始化后，随后执行 USS_Port_Scan 即可开始与具有此驱动器编号的驱动器通信。

更改驱动器编号操作将要求 CPU 从 STOP 模式切换到 RUN 模式以初始化相应的背景数据块。将输入参数组态到 USS TX 消息缓冲区中，并从“前一个”有效响应缓冲区（如果存在）读取输出。USS_Drive_Control 执行期间不进行数据传送。驱动器在 USS_Port_Scan 执行时通信。USS_Drive_Control 仅组态要发送的消息并解释已从前一个请求中接收的数据。

用户可以使用 DIR 输入 (Bool) 或使用符号（正或负）和 SPEED_SP 输入 (Real) 控制驱动器旋转方向。下表假定电机按正向旋转接线，说明这些输入如何一起决定驱动器旋转方向。

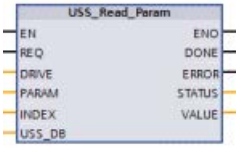
表格 13-46 SPEED_SP 和 DIR 参数的交互作用

SPEED_SP	DIR	驱动器旋转方向
数值 > 0	0	反转
数值 > 0	1	正转
数值 < 0	0	正转
数值 < 0	1	反转

13.4.3.3 USS_Read_Param（从驱动器读取参数）

表格 13-47 USS_Read_Param 指令

LAD/FBD	SCL	说明

	`USS_Read_Param(REQ:=_bool_in_,`
	`DRIVE:=_usint_in_,`
	`PARAM:=_uint_in_,`
	`INDEX:=_uint_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`VALUE=>_variant_out_,`
	`USS_DB:=_fbtref_inout_);`
		USS_Read_Param 指令用于从驱动器读取参数。与同一个 USS 网络和 PtP 通信端口相关的所有 USS 功能都必须使用同一个数据块。必须从主程序循环 OB 调用 USS_Read_Param。

表格 13-48 参数的数据类型

参数类型		数据类型	说明
REQ	IN	Bool	发送请求：REQ 为真时，表示需要新的读请求。如果该参数的请求已处于待决状态，将忽略新请求。
DRIVE	IN	USInt	驱动器地址：DRIVE 是 USS 驱动器的地址。有效范围是驱动器 1 到驱动器 16。
PARAM	IN	UInt	参数编号：PARAM 指示要写入的驱动器参数。该参数的范围为 0 到 2047。在部分驱动器上，最重要的字节可以访问值大于 2047 的 PARAM。有关如何访问扩展范围的详细信息，请参见驱动器手册。
INDEX	IN	UInt	参数索引：INDEX 指示要写入的驱动器参数索引。索引为一个 16 位值，其中最低有效字节是实际索引值，其范围是 0 到 255。最高有效字节也可供驱动器使用，且取决于具体的驱动器。有关详细信息，请参见驱动器手册。
USS_DB	INOUT	USS_BASE	将 USS_Drive_Control 指令放入程序时创建并初始化的背景数据块的名称。
VALUE	IN	Word, Int, UInt, DWord, DInt, UDInt, Real	这是已读取的参数的值，仅当 DONE 位为真时才有效。

参数类型		数据类型	说明
DONE[1]	OUT	Bool	该参数为真时，表示 VALUE 输出包含先前请求的读取参数值。USS_Drive_Control 发现来自驱动器的读响应数据时会设置该位。满足以下条件之一时复位该位：用户通过另一个 USS_Read_Param 轮询请求响应数据，或在执行接下来两个 USS_Drive_Control 调用的第二个时请求
ERROR	OUT	Bool	出现错误：ERROR 为真时，表示发生错误，并且 STATUS 输出有效。其它所有输出在出错时均设置为零。仅在 USS_Port_Scan 指令的 ERROR 和 STATUS 输出中报告通信错误。
STATUS	OUT	Word	STATUS 表示读请求的结果。对于有些状态代码，还在“USS_Extended_Error”变量中提供了更多信息。

[1] DONE 位表示已从参考电机驱动器读取有效数据并已将其传送给 CPU。它不表示 USS 库能够立即读取另一参数。必须将空的 PKW 请求发送到电机驱动器并由指令确认，才能使用特定驱动器的参数通道。立即调用指定电机驱动器的 USS_Read_Param 或 USS_Write_Param FC 将导致“0x818A”错误。

13.4.3.4 USS_Write_Param（修改驱动器中的参数）

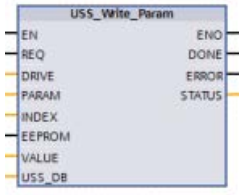
说明

EEPROM 写操作（用于 USS 驱动器内部的 EEPROM）

请勿过多使用 EEPROM 永久写操作。请尽可能减少 EEPROM 写操作次数以延长 EEPROM 的寿命。

表格 13-49 USS_Write_Param 指令

LAD/FBD	SCL	说明

	`USS_Write_Param(REQ:=_bool_in_,`
	`DRIVE:=_usint_in_,`
	`PARAM:=_uint_in_,`
	`INDEX:=_uint_in_,`
	`EEPROM:=_bool_in_,`
	`VALUE:=_variant_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`USS_DB:=_fbtref_inout_);`
		USS_Write_Param 指令用于修改驱动器中的参数。与同一个 USS 网络和 PtP 通信端口相关的所有 USS 功能都必须使用同一个数据块。
		必须从主程序循环 OB 中调用 USS_Write_Param。

表格 13-50 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	发送请求：REQ 为真时，表示需要新的写请求。如果该参数的请求已处于待决状态，将忽略新请求。
DRIVE	IN	USInt	驱动器地址：DRIVE 是 USS 驱动器的地址。有效范围是驱动器 1 到驱动器 16。
PARAM	IN	UInt	参数编号：PARAM 指示要写入的驱动器参数。该参数的范围为 0 到 2047。在部分驱动器上，最重要的字节可以访问值大于 2047 的 PARAM。有关如何访问扩展范围的详细信息，请参见驱动器手册。
INDEX	IN	UInt	参数索引：INDEX 指示要写入的驱动器参数索引。索引为一个 16 位值，其中最低有效字节是实际索引值，其范围是 0 到 255。最高有效字节也可供驱动器使用，且取决于具体的驱动器。有关详细信息，请参见驱动器手册。
EEPROM	IN	Bool	存储到驱动器 EEPROM：该参数为真时，写驱动器参数事务将存储在驱动器 EEPROM 中。如果为假，则写操作是临时的，在驱动器循环上电后不会保留。
VALUE	IN	Word, Int, UInt, DWord, DInt, UDInt, Real	要写入的参数值。切换为 REQ 时该值必须有效。
USS_DB	INOUT	USS_BASE	将 USS_Drive_Control 指令放入程序时创建并初始化的背景数据块的名称。
DONE[1]	OUT	Bool	DONE 为真时，表示输入 VALUE 已写入驱动器。 USS_Drive_Control 发现来自驱动器的写响应数据时会设置该位。如果用户通过另一个 USS_Drive_Control 轮询请求响应数据，或在执行接下来两个 USS_Drive_Control 调用的第二个时请求响应数据，则复位该位。
ERROR	OUT	Bool	ERROR 为真时，表示发生错误，并且 STATUS 输出有效。其它所有输出在出错时均设置为零。仅在 USS_Port_Scan 指令的 ERROR 和 STATUS 输出中报告通信错误。
STATUS	OUT	Word	STATUS 表示写请求的结果。对于有些状态代码，还在“USS_Extended_Error”变量中提供了更多信息。

[1] DONE 位表示已从参考电机驱动器读取有效数据并已将其传送给 CPU。它不表示 USS 库能够立即读取另一参数。必须将空的 PKW 请求发送到电机驱动器并由指令确认，才能使用特定驱动器的参数通道。立即调用指定电机驱动器的 USS_Read_Param 或 USS_Write_Param FC 将导致“0x818A”错误。

13.4.4 USS 状态代码

在 USS 功能的 STATUS 输出端返回 USS 指令状态代码。

表格 13-51 STATUS 代码 [1]

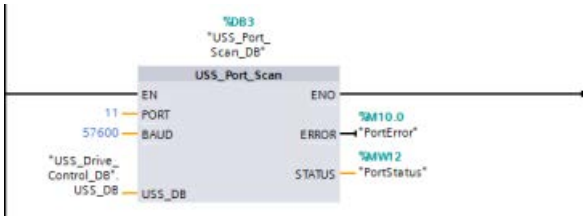
STATUS (W#16#....)	说明
0000	无错误
8180	驱动器响应的长度与从驱动器收到的字符数不匹配。 出错的驱动器编号在“USS_Extended_Error”变量中返回。 请参见本表格下方的扩展错误描述。
8181	VALUE 参数不是 Word、Real 或 DWord 数据类型。
8182	用户提供了 Word 参数值，但从驱动器响应中收到 DWord 或 Real 值。
8183	用户提供了 DWord 或 Real 参数值，但从驱动器响应中收到 Word 值。
8184	驱动器响应报文的校验和有错误。 出错的驱动器编号在“USS_Extended_Error”变量中返回。 请参见本表格下方的扩展错误描述。
8185	非法的驱动器地址（有效驱动器地址范围： 1 到 16）
8186	速度设定值超出有效范围（有效速度 SP 范围： -200% 到 200%）。
8187	对已发送的请求响应了错误的驱动器编号。 出错的驱动器编号在“USS_Extended_Error”变量中返回。 请参见本表格下方的扩展错误描述。
8188	指定的 PZD 字长度非法（有效范围 = 2、4、6 或 8 个字）
8189	指定了非法的波特率。
818A	参数请求通道正在由该驱动器的另一个请求使用。
818B	驱动器尚未对请求和重试做出响应。 出错的驱动器编号在“USS_Extended_Error”变量中返回。 请参见本表格下方的扩展错误描述。
818C	驱动器返回了有关参数请求操作的扩展错误。 请参见本表格下方的扩展错误描述。
818D	驱动器返回了有关参数请求操作的非法访问错误。 有关可能限制参数访问的原因信息，请参见驱动器手册。
818E	驱动器尚未初始化。 若从未调用过该驱动器的 USS_Drive_Control，则该错误代码将返回到 USS_Read_Param 或 USS_Write_Param。 这会防止首次扫描 USS_Drive_Control 的初始化过程覆盖未决的参数读/写请求，因为它会将驱动器初始化为新条目。 要修复该错误，请针对此驱动器编号调用 USS_Drive_Control。
80Ax-80Fx	从 USS 库调用的 PtP 通信 FB 返回的特定错误 - 这些错误代码值不会被 USS 库修改且在 PtP 指令说明中定义。

[1] 除了上述列出的 USS 指令错误，还可能返回底层 PtP 通信指令 (页 1135)的错误信息。

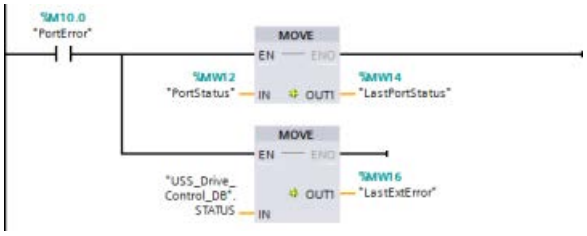
对于一些 STATUS 代码，在 USS_Drive_Control 背景数据块的“USS_Extended_Error”变量中提供更多信息。 对于 STATUS 代码 8180、8184、8187 和 818B（十六进制），USS_Extended_Error 包含出现通信错误的驱动器编号。 对于 STATUS 代码 818C（十六进制），USS_Extended_Error 包含使用 USS_Read_Param 或 USS_Write_Param 指令时从驱动器返回的驱动器错误代码。

示例： 通信错误报告

仅报告有关 USS_Port_Scan 指令（而非 USS_Drive_Control 指令）的通信错误 (STATUS = 16#818B)。 例如，如果没有正确地终止程序段，则驱动器可能切换到 RUN 模式，但 USS_Drive_Control 指令将为相关输出参数全部显示“0”。 在这种情况下，只能检测有关 USS_Port_Scan 指令的通信错误。 由于该错误仅在一个扫描周期内可见，所以需要添加一些捕获逻辑，如下面的示例所示。 在本例中，当 USS_Port_Scan 指令的错误位为 TRUE 时，STATUS 和 USS_Extended_Error 值将保存到 M 存储器中。 当 STATUS 代码值是十六进制的 8180、8184、8187 或 818B 时，驱动器编号将放在 USS_Extended_Error 变量中。



程序段 1"PortStatus"端口状态和 "USS_Drive_Control_DB".USS_Extended_Error
扩展错误代码值仅在
一个程序扫描周期内有效。 必须捕获这些值
以便后期处理。



程序段 2“PortError”触点触发将 “PortStatus”值存储在 “LastPortStatus”中以及将 "USS_Drive_Control_DB".USS_Extended_Error
值存储在“LastExtError”中。

对驱动器的内部参数进行读写访问

USS 驱动器支持对驱动器的内部参数进行读写访问。 通过该功能可进行驱动器的远程控制和组态。 由于发生类似值超出范围或驱动器当前模式的请求非法等错误，驱动器参数访问操作可能会失败。 驱动器会生成在“USS_Extended_Error”变量中返回的错误代码值。 该错误代码值仅对 USS_Read_Param 或 USS_Write_Param 指令的最后一次执行有效。 当 STATUS code 值为十六进制的 818C 时，驱动器错误代码将放入 USS_Extended_Error 变量中。 USS_Extended_Error 的错误代码值取决于驱动器型号。 有关读写参数操作的扩展错误代码的描述，请参见驱动器手册。

13.4.5 USS 常规驱动器设置要求

USS 常规驱动器设置要求包括以下几点：

- 驱动器必须设置为使用 4 个 PKW 字。
- 驱动器可组态为使用 2、4、6 或 8 个 PZD 字。
- 驱动器中 PZD 字的数量必须与该驱动器的 USS_DRV 指令的 USS_Drive_Control 输入相匹配。
- 所有驱动器的波特率必须与 USS_Port_Scan 指令的 BAUD 输入相匹配。
- 驱动器必须设置为可进行远程控制。
- 驱动器必须设置为使用适合通信链路上 USS 的频率设定值。
- 驱动器地址必须设置为 1 到 16，并且与 USS_Drive_Control 块上对应该驱动器的 DRIVE 输入相匹配。
- 驱动器的方向控制必须设置为使用驱动器设定值的极性。
- 必须正确终止 RS485 网络。

13.4.6 示例： USS 常规驱动器连接和设置

连接 MicroMaster 驱动器

本部分以 SIEMENS MicroMaster 驱动器为例提供相关信息。对于其它驱动器，请参见驱动器手册查看相关设置说明。

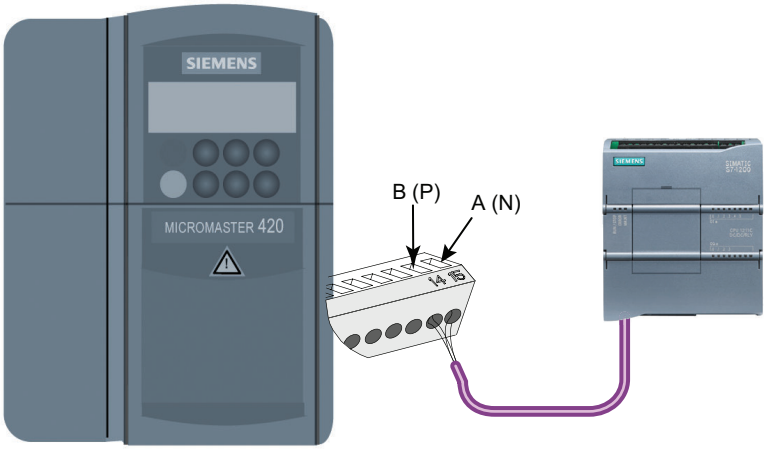
要建立与 MicroMaster 系列 4 (MM4) 驱动器的连接，请将 RS485 电缆的两端插入两个用于 USS 操作的笼式夹持无螺丝端子中。可使用标准 PROFIBUS 电缆和连接器连接 S7-1200。

⚠ 小心

参考电位不同的互连设备可导致互连电缆中有不想要的电流流过

这些不想要的电流可引发通信错误或损坏设备。确保要使用通信电缆连接的所有设备都共用一个公共电路参考点，或者进行隔离以防止出现不想要的电流。屏蔽层必须与外壳地或 9 针连接器的引脚 1 连接。建议将 MicroMaster 驱动器上的接线端子 2-0 V 与外壳地连接。

RS485 电缆另一端的两根线必须插入 MM4 驱动器的接线板中。要实现 MM4 驱动器上的电缆连接，请去掉驱动器的盖，露出接线板。有关如何去掉特定驱动器的盖的详细信息，请参见 MM4 用户手册。



接线板连接标有数字。使用 S7-1200 侧的 PROFIBUS 连接器，将电缆的 A 端子连接到驱动器的端子 15（对于 MM420）或端子 30 (MM440)。将 B (P) A (N) 电缆连接器的 B 端子连接到端子 14 (MM420) 或端子 29 (MM440)。

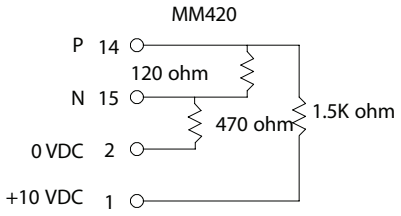
如果 S7-1200 是网络中的终止节点，或者如果是点对点连接，则需要使用连接器的端子 A1 和 B1（不是 A2 和 B2），这是因为通过这两个端子可进行终端设置（例如，使用 6ES7972-0BA40-0X40 型号的 DP 连接器）。

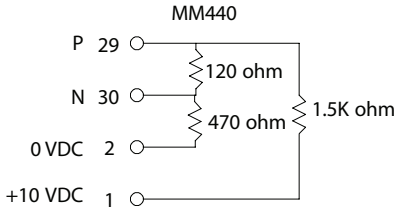
⚠ 小心

供电前正确更换驱动器盖

确保给设备通电之前已正确装上了驱动器盖。

如果驱动器被组态为网络中的终止节点，则还必须将终端电阻和偏压电阻连接到适当的终端连接。此图显示了终止和偏压所需的 MM4 驱动器连接示例。





设置 MM4 驱动器

将驱动器与 S7-1200 连接之前，必须确保驱动器具有下列系统参数。使用驱动器上的小键盘设置参数：

1.恢复驱动器的出厂设置（可选）。	P0010=30 P0970=1
如果跳过步骤 1，则请确保这些参数被设置成所指示的值。	USS PZD 长度 = P2012 索引 0 =（2、4、6 或 8） USS PKW 长度 = P2013 索引 0 = 4
2.启用对所有参数的读/写访问（专家模式）。	P0003=3
3.检查驱动器的电机设置。所使用的电机不同，设置也会不同。 要设置参数 P304、P305、P307、P310 和 P311，首先必须将参数 P010 设置为 1（快速调试模式）。这些参数设置完毕后，将参数 P010 设置为 0。参数 P304、P305、P307、P310 和 P311 只能在快速调试模式下更改。	P0304 = 电机额定电压 (V) P0305 = 电机额定电流 (A) P0307 = 电机额定功率 (W) P0310 = 电机额定频率 (Hz) P0311 = 电机额定速度
4.设置本地/远程控制模式。	P0700 索引 0=5
5.根据通信链路的 USS 选择频率设定值。	P1000 索引 0=5
6.加速时间（可选） 这是电机加速到最大频率所需的时间（秒）。	P1120 =（0 到 650.00）
7.减速时间（可选） 这是电机减速到完全停止所需的时间（秒）。	P1121 =（0 到 650.00）
8.设置串行链路的基准频率：	P2000 =（1 到 650 Hz）
9.设置 USS 标准化：	P2009 索引 0=0

10.设置 RS485 串行接口的波特率：	P2010 索引 0= 4（2400 波特） 5（4800 波特） 6（9600 波特） 7（19200 波特） 8（38400 波特） 9（57600 波特） 12（115200 波特）
11.输入从站地址。 可通过总线操作每台驱动器（最多 31 台）。	P2011 索引 0 =（0 到 31）
12.设置串行链路超时。 这是两份传入数据报文之间所允许的最长时间段。此功能用于在出现通信故障时关闭反相器。收到有效的数据报文后开始计时。如果在指定的时间段内未收到其它数据报文，则反相器脱扣并显示故障代码 F0070。将该值设置为零将关闭控制。	P2014 索引 0 =（0 到 65,535 ms） （0 = 禁用超时）
13.将数据从 RAM 传送到 EEPROM：	P0971=1（启动传送）将参数设置更改内容保存到 EEPROM

13.5 Modbus 通信

13.5.1 Modbus RTU 和 Modbus TCP 通信概述

Modbus 功能代码

- CPU 作为 Modbus RTU 主站（或 Modbus TCP 客户端）运行时，可在远程 Modbus RTU 从站（或 Modbus TCP 服务器）中读/写数据和 I/O 状态。可在程序逻辑中读取并处理远程数据。
- CPU 作为 Modbus RTU 从站（或 Modbus TCP 服务器）运行时，监控设备可在 CPU 存储器中读/写数据和 I/O 状态。RTU 主站（或 Modbus TCP 客户端）可以将新值写入从站/服务器 CPU 存储器，以供用户程序逻辑使用。

⚠ 警告

如果攻击者能以物理方式访问您的网络，那么便可能读写数据。

TIA Portal、CPU 和 HMI（使用 GET/PUT 的 HMI 除外）均采用安全通信，可防止重放攻击和“中间人”攻击。启用这种通信后，将以纯文本形式交换签名消息，这种方式允许攻击者读取数据，但可避免未经授权的数据写入操作。TIA Portal（而非通信过程）将对受专有技术保护的块中的数据进行加密。

所有其它形式的通信（通过 PROFIBUS、PROFINET、AS-i 或其它 I/O 总线、GET/PUT、传输块 (T-block) 和通信模块 (CM) 进行的 I/O 交换）均没有安全功能。必须通过限制物理访问来保护这些形式的通信。如果攻击者能利用这些形式的通信以物理方式访问您的网络，那么便可能读写数据。

相关安全信息和建议，请参见“工业安全操作准则” (http://www.industry.siemens.com/topics/global/en/industrial-security/Documents/operational_guidelines_industrial_security_en.pdf)。

表格 13-52　读取数据功能：读取远程 I/O 及程序数据

Modbus 功能代码	读取从站（服务器）功能 - 标准寻址
01	读取输出位：每个请求 1 到 2000 个位
02	读取输入位：每个请求 1 到 2000 个位
03	读取保持寄存器：每个请求 1 到 125 个字
04	读取输入字：每个请求 1 到 125 个字

表格 13-53　写入数据功能：写入远程 I/O 及修改程序数据

Modbus 功能代码	写入从站（服务器）功能 - 标准寻址
05	写入一个输出位：每个请求 1 位
06	写入一个保持寄存器：每个请求 1 个字
15	写入一个或多个输出位：每个请求 1 到 1968 个位
16	写入一个或多个保持寄存器：每个请求 1 到 123 个字

- Modbus 功能代码 08 和 11 提供从站设备通信诊断信息。
- Modbus 功能代码 0 将消息广播到所有从站（无从站响应）。广播功能不能用于 Modbus TCP，因为通信是以连接为基础的。

表格 13-54 Modbus 网络站地址

站		地址
RTU 站	标准站地址	1 到 247
	扩展站地址	1 到 65535
TCP 站	站地址	IP 地址和端口号

Modbus 存储区地址

实际可用的 Modbus 存储区地址数取决于 CPU 型号、存在多少工作存储器以及其他程序数据占用多少 CPU 存储区。下表给出地址范围的额定值。

表格 13-55 Modbus 存储区地址

站		地址范围
RTU 站	标准存储区地址	10K
	扩展存储区地址	64K
TCP 站	标准存储区地址	10K

Modbus RTU 通信

Modbus RTU（远程终端单元）是一个标准的网络通信协议，它使用 RS232 或 RS485 电气连接在 Modbus 网络设备之间传输串行数据。可在带有一个 RS232 或 RS485 CM 或一个 RS485 CB 的 CPU 上添加 PtP（点对点）网络端口。

Modbus RTU 使用主/从网络，单个主设备启动所有通信，而从设备只能响应主设备的请求。主设备向从一个从设备地址发送请求，然后该从设备地址对命令做出响应。

Modbus TCP 通信

Modbus TCP（传输控制协议）是一个标准的网络通信协议，它使用 CPU 上的 PROFINET 连接器进行 TCP/IP 通信。不需要额外的通信硬件模块。

Modbus TCP 使用开放式用户通信 (OUC, Open User Communication) 连接作为 Modbus 通信路径。除了 STEP 7 和 CPU 之间的连接外，还可能存在多个客户端-服务器连接。支持的混合客户端和服务器连接数最大为 CPU 型号所允许的最大连接数 (页 807)。

每个 MB_SERVER 连接必须使用一个唯一的背景数据块和 IP 端口号。每个 IP 端口只能用于 1 个连接。必须为每个连接单独执行各 MB_SERVER（带有其唯一的背景数据块和 IP 端口）。

Modbus TCP 客户端（主站）必须通过 DISCONNECT 参数控制客户端-服务器连接。基本的 Modbus 客户端操作如下所示。

1. 连接到特定服务器（从站）IP 地址和 IP 端口号
2. 启动 Modbus 消息的客户端传输，并接收服务器响应
3. 根据需要断开客户端和服务器的连接，以便与其它服务器连接。

程序中的 Modbus RTU 指令

- Modbus_Comm_Load：通过执行一次 Modbus_Comm_Load ，设置 PtP 端口参数，如波特率、奇偶校验和流控制。为 Modbus RTU 协议组态 CPU 端口后，该端口只能由 Modbus_Master Modbus_Slave 指令使用。
- Modbus_Master：Modbus_Master 指令使 CPU 充当 Modbus RTU 主设备，并与一个或多个 Modbus 从设备进行通信。
- Modbus_Slave：Modbus_Slave 指令使 CPU 充当 Modbus RTU 从设备，并与一个 Modbus 主设备进行通信。

程序中的 Modbus TCP 指令

- MB_CLIENT：进行客户端-服务器 TCP 连接、发送命令消息、接收响应，以及控制服务器断开
- MB_SERVER：根据要求连接至 Modbus TCP 客户端、接收 Modbus 信息及发送响应

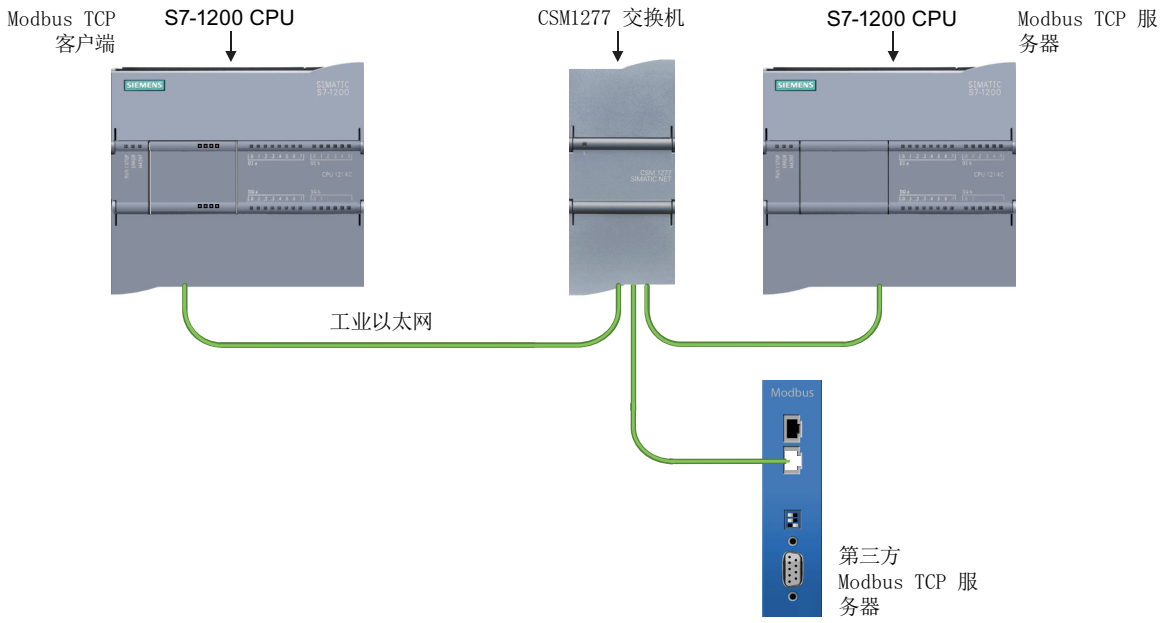
13.5.2 Modbus TCP

13.5.2.1 概述

对于 V4.1 及以上版本的 S7-1200 CPU 和 STEP 7 V13 SP1，该 CPU 扩展了 Modbus TCP 的功能，可使用增强型 T 块指令。

出于这个原因，S7-1200 支持两组 PtP 指令：

- 早期 Modbus TCP 指令 (页 1325)：这些 Modbus RTU 指令存在于 S7-1200 的 V4.0 版本之前。
- Modbus TCP 指令 (页 1197)：这些 Modbus TCP 指令提供了早期指令的全部功能。



13.5.2.2 选择 Modbus TCP 指令的版本

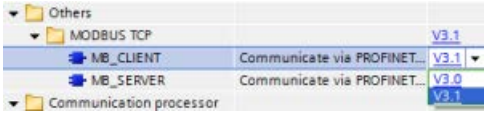
在 STEP 7 中可使用以下版本的 Modbus TCP 指令：

- 历史版本 2.1：兼容所有 CPU 和 CM 版本
- 历史版本 3.1：兼容所有 CPU 和 CM 版本
- 版本 4.2：兼容版本 V4.0 及以上的 CPU 和 V2.1 及以上的 CM
- 版本 5.1：兼容 V4.2 及以上版本的 CPU 和 V2.1 及以上版本的 CM

可以从兼容性和移植便利性方面考虑，选择将相应的指令版本插入用户程序中。

在指令任务卡中，在 “通信”(Communication) 组中的 “其它”(Others) 下显示 MODBUS TCP 指令。

要更改 Modbus TCP 指令的版本，请从下拉列表中选择相应版本。可以选择一组指令或分别选择各个指令。



使用指令树将 Modbus TCP 指令放入程序时，将在项目树中创建新的 FB 实例。在项目树的 “PLC_x > 程序块 > 系统块 > 程序资源”(PLC_x > Program blocks > System blocks > Program resources) 下可看到新的 FB 实例。

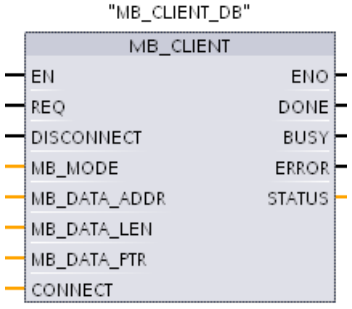
要确认程序中 Modbus TCP 指令的版本，必须检查项目树的属性而不是程序编辑器中显示的框的属性。选择项目树的 Modbus TCP FB 实例，单击右键，选择“属性”(Properties)，然后选择“信息”(Information) 页查看 Modbus TCP 指令的版本号。

13.5.2.3 Modbus TCP 指令

MB_CLIENT（作为 Modbus TCP 客户端使用 PROFINET 进行通信）指令

表格 13-56 MB_CLIENT 指令

LAD/FBD	SCL	说明

	`"MB_CLIENT_DB"(`
	`REQ:=_bool_in_,`
	`DISCONNECT:=_bool_in_,`
	`MB_MODE:=_usint_in_,`
	`MB_DATA_ADDR:=_udint_in_,`
	`MB_DATA_LEN:=_uint_in_,`
	`DONE=>_bool_out_,`
	`BUSY=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`MB_DATA_PTR:=_variant_inout_,`
	`CONNECT:=_variant_inout_);`
		MB_CLIENT 作为 Modbus TCP 客户端，通过 S7-1200 CPU 上的 PROFINET 端口进行通信。不需要额外的通信硬件模块。
		MB_CLIENT 可进行客户端-服务器连接、发送 Modbus 功能请求、接收响应，以及控制 Modbus TCP 服务器的断开。

表格 13-57 参数的数据类型

参数和类型		数据类型	说明
REQ	In	Bool	FALSE = 无 Modbus 通信请求 TRUE = 请求与 Modbus TCP 服务器通信
DISCONNECT	IN	Bool	DISCONNECT 参数允许程序控制与 Modbus 服务器设备的连接和断开。 如果 DISCONNECT = 0 且不存在连接，则 MB_CLIENT 尝试连接到分配的 IP 地址和端口号。 如果 DISCONNECT = 1 且存在连接，则尝试断开连接操作。每当启用此输入时，无法尝试其它操作。
MB_MODE	IN	USInt	模式选择：分配请求类型（读、写或诊断）。请参见下面的 Modbus 功能表了解详细信息。
MB_DATA_ADDR	IN	UDInt	Modbus 起始地址：分配 MB_CLIENT 访问的数据的起始地址。有效地址的相关信息，请参见下面的 Modbus 功能表。

参数和类型		数据类型	说明
MB_DATA_LEN	IN	UInt	Modbus 数据长度：分配此请求中要访问的位数或字数。有效长度的相关信息，请参见下面的 Modbus 功能表。
MB_DATA_PTR	IN_OUT	Variant	指向 Modbus 数据寄存器的指针：寄存器缓冲进出 Modbus 服务器的数据。指针必须分配一个未进行优化的全局 DB 或 M 存储器地址。
CONNECT	IN_OUT	Variant	引用包含系统数据类型为“TCON_IP_v4”的连接参数的数据块结构。
DONE	OUT	Bool	上一请求已完成且没有出错后，DONE 位将保持为 TRUE 一个扫描周期时间。
BUSY	OUT	Bool	• 0 - 无 MB_CLIENT 操作正在进行 • 1 - MB_CLIENT 操作正在进行
ERROR	OUT	Bool	MB_CLIENT 执行因错误而结束后，ERROR 位将在一个扫描周期时间内保持为 TRUE。STATUS 参数中的错误代码仅在 ERROR = TRUE 的一个循环周期内有效。
STATUS	OUT	Word	执行条件代码

说明

CPU 固件版本要求

本手册中所述的 Modbus TCP 指令要求固件版本为 V4.1 或更高版本。

REQ 参数

FALSE = 无 Modbus 通信请求
TRUE = 请求与 Modbus TCP 服务器通信

如果 MB_CLIENT 的实例没有激活且参数 DISCONNECT=0，当 REQ=1 时，将启动一个新的 Modbus 请求。如果尚未建立连接，则建立一个新的连接。

如果在当前请求完成前 DISCONNECT=0 且 REQ=1，从而再次执行 MB_CLIENT 的同一个实例，则不会进行后续 Modbus 传送。但是，一旦完成当前请求，如果通过 REQ=1 执行 MB_CLIENT，可处理新的请求。

完成当前 MB_CLIENT 通信请求后，DONE 位将在一个周期内保持为 TRUE。DONE 位可用作定时门，对多个 MB_CLIENT 请求进行排序。

说明

MB_CLIENT 处理期间输入数据的一致性

Modbus 客户端启动 Modbus 操作后，将在内部保存所有输入状态，然后在每次后续调用时进行比较。比较用于确定此特定调用是否是活动客户端请求的发起者。可使用一个公用背景数据块执行多个 MB_CLIENT 调用。

在主动处理 MB_CLIENT 操作期间应不改变输入，这一点很重要。若不遵循此规则，则 MB_CLIENT 无法确定活动实例。

MB_MODE 和 MB_DATA_ADDR 参数用于选择 Modbus 通信功能

MB_CLIENT 指令使用 MB_MODE 输入而非功能代码。MB_DATA_ADDR 分配远程数据的起始 Modbus 地址。

MB_MODE 和 MB_DATA_ADDR 一起确定实际 Modbus 消息中使用的功能代码。下表列出了 MB_MODE、MB_DATA_ADDR 和 Modbus 功能之间的对应关系：

表格 13-58 Modbus 功能

MB_MODE	Modbus 功能	数据长度	操作和数据	MB_DATA_ADDR
0	01	1 到 2000	读取输出位： 每个请求 1 到 2000 个位	1 到 9999
101	01	1 到 2000	读取输出位： 每个请求 1 到 2000 个位	00000 到 65535
0	02	1 到 2000	读取输入位： 每个请求 1 到 2000 个位	10001 到 19999
102	02	1 到 2000	读取输入位： 每个请求 1 到 2000 个位	00000 到 65535
0	03	1 到 125	读取保持寄存器： 每个请求 1 到 125 个字	40001 到 49999 或 400001 到 465535
103	03	1 到 125	读取保持寄存器： 每个请求 1 到 125 个字	00000 到 65535

MB_MODE	Modbus 功能	数据长度	操作和数据	MB_DATA_ADDR
0	04	1 到 125	读取输入字： 每个请求 1 到 125 个字	30001 到 39999
104	04	1 到 125	读取输入字： 每个请求 1 到 125 个字	00000 到 65535
1	05	1	写入一个输出位： 每个请求一位	1 到 9999
105	05	1	写入一个输出位： 每个请求一位	00000 到 65535
1	06	1	写入一个保持寄存器： 每个请求 1 个字	40001 到 49999 或 400001 到 465535
106	06	1	写入一个保持寄存器： 每个请求 1 个字	00000 到 65535
1	15	2 到 1968	写入多个输出位： 每个请求 2 到 1968 个位	1 到 9999
1	16	2 到 123	写入多个保持寄存器： 每个请求 2 到 123 个字	40001 到 49999 或 400001 到 465535
2	15	1 到 1968	写入一个或多个输出位： 每个请求 1 到 1968 个位	1 到 9999
2	15	1 到 1968	写入一个或多个输出位： 每个请求 1 到 1968 个位	1 到 9999
2	15	1 到 1968	写入一个或多个输出位： 每个请求 1 到 1968 个位	1 到 9999
115	15	1 到 1968	写入一个或多个输出位： 每个请求 1 到 1968 个位	00000 到 65535
2	16	1 到 123	写入一个或多个保持寄存器： 每个请求 1 到 123 个字	40001 到 49999 或 400001 到 465535
116	16	1 到 123	写入一个或多个保持寄存器： 每个请求 1 到 123 个字	00000 到 65535

MB_MODE	Modbus 功能	数据长度	操作和数据	MB_DATA_ADDR
11	11	0	读取服务器通信状态字和事件计数器。状态字指示忙闲情况（0 = 不忙，0xFFFF = 忙）。每成功完成一条消息，事件计数器的计数值递增。 对于该功能，MB_CLIENT 的 MB_DATA_ADDR 和 MB_DATA_LEN 参数都将被忽略。	
80	08	1	利用诊断代码 0x0000 检查服务器状态（回送测试、服务器回送请求） 每个请求 1 个字	
81	08	1	利用诊断代码 0x000A 重新设置服务器事件计数器 每个请求 1 个字	
3 到 10、12 到 79、82 到 100、107 到 114、117 到 255			保留	

说明

MB_DATA_PTR 分配一个缓冲区来存储从 Modbus TCP 服务器读取或写入到该服务器的数据

数据缓冲区可位于未进行优化的全局 DB 或 M 存储区地址中。

对于 M 存储器中的缓存器，使用 Any 指针格式。具体格式为 P#“位地址”“数据类型”“长度”，例如 P#M1000.0 WORD 500。

MB_DATA_PTR 参数指定一个通信缓冲区

- MB_CLIENT 通信功能：
  - 从 Modbus 服务器地址（00001 到 09999）读写 1 位数据
  - 从 Modbus 服务器地址（10001 到 19999）读取 1 位数据
  - 从 Modbus 服务器地址（30001 到 39999）和（40001 到 49999）读取 16 位字数据
  - 向 Modbus 服务器地址（40001 到 49999）写入 16 位字数据
- 向/从 MB_DATA_PTR 分配的 DB 或 M 储存器缓冲区传输字或位大小的数据。
- 如果通过 MB_DATA_PTR 分配 DB 为缓冲区，必须为所有 DB 数据元素分配数据类型。
  - 1 位 Bool 数据类型代表一个 Modbus 位地址
  - 16 位单字数据类型（如 WORD、UInt 和 Int）代表一个 Modbus 字地址
  - 32 位双字数据类型（如 DWORD、DInt 和 Real）代表两个 Modbus 字地址
- 可以通过 MB_DATA_PTR 分配复杂的 DB 元素，例如
  - 数组
  - 指定的结构，其中每个元素都是唯一的。
  - 指定的复杂结构，其中每个元素都具有唯一的名称以及 16 或 32 位数据类型。
- 不要求 MB_DATA_PTR 数据区位于同一个全局数据块（或 M 存储区）中。可分配一个数据块供 Modbus 读取，分配另一个数据块供 Modbus 写入，或分配一个数据块用于各个 MB_CLIENT。

CONNECT 参数分配用于建立 PROFINET 连接的数据

必须使用全局数据块并存储所需的连接数据，然后才能在 CONNECT 参数中引用此 DB。

1. 创建新的全局 DB 或使用现有全局 DB 来存储 CONNECT 数据。可使用一个 DB 存储多个 TCON_IP_v4 数据结构。每个 Modbus TCP 客户端或服务器连接使用一个 TCON_IP_v4 数据结构。可在 CONNECT 参数中引用连接数据。
2. 使用有帮助的名称对 DB 和静态变量进行命名。例如，将数据块命名为“Modbus 连接”，将静态变量命名为“TCPactive_1”（针对 Modbus TCP 客户端连接 1）。
3. 在 DB 编辑器的 “数据类型”(Data Type) 列中为示例静态变量“TCPactive_1”分配系统数据类型“TCON_IP_v4”。
4. 扩展 TCON_IP_v4 结构，从而可以修改连接参数，如下图所示。
5. 修改 MB_CLIENT 连接的 TCON_IP_v4 结构数据。
6. 输入 MB_CLIENT CONNECT 参数的 DB 结构引用。本示例中应为“Modbus 连接”.TCPactive_1。

Modbus connections

	名称	数据类型	启动值	注释
1	Static			
2	TCPactive_1	TCON_IP_v4		
3	InterfaceId	HW_ANY	64	HW-identifier of IE-interface submodule
4	ID	CONN_OUC	1	connection reference / identifier
5	ConnectionType	Byte	16#0B	type of connetion: 11=TCP/IP, 19=UDP (17=TC...
6	ActiveEstablished	Bool	True	active/passive connection establishment
7	RemoteAddress	IP_V4		remote IP address (IPv4)
8	ADDR	array [1..4] of Byte		IPv4 address
9	ADDR[1]	Byte	192	
10	ADDR[2]	Byte	168	
11	ADDR[3]	Byte	2	
12	ADDR[4]	Byte	241	
13	RemotePort	UInt	502	remote UDP/TCP port number
14	LocalPort	UInt	0	local UDP/TCP port number

修改各 MB_CLIENT 连接的 TCON_IP_V4 DB 数据

- **InterfaceID**：在设备组态窗口中单击 CPU PROFINET 端口图像。然后单击“常规” (General) 属性选项卡并使用该处显示的硬件标识符。
- **ID**：输入一个介于 1 到 4095 之间的连接 ID 编号。使用底层 TCON、TDISCON、TSEND 和 TRCV 指令建立 Modbus TCP 通信，用于 OUC（开放式用户通信）。
- **ConnectionType**：对于 TCP/IP，使用默认值 16#0B（十进制数 = 11）。
- **ActiveEstablished**：该值必须为 1 或 TRUE。主动连接，由 MB_CLIENT 启动 Modbus 通信。
- **RemoteAddress**：将目标 Modbus TCP 服务器的 IP 地址输入到四个 ADDR 数组单元中。例如，如上图所示输入 192.168.2.241。
- **RemotePort**：默认值为 502。该编号为 MB_CLIENT 试图连接和通信的 Modbus 服务器的 IP 端口号。一些第三方 Modbus 服务器要求使用其它端口号。
- **LocalPort**：对于 MB_CLIENT 连接，该值必须为 0。

多个客户端连接

Modbus TCP 客户端支持的并发连接数最多为 PLC 允许的开放式用户通信最大连接数。PLC 的连接总数（包括 Modbus TCP 客户端和服务器）不得超过支持的开放式用户通信最大连接数 (页 807)。

单独的并发客户端连接必须遵循以下规则：

- 各 MB_CLIENT 连接必须使用一个唯一的背景 DB
- 必须为各 MB_CLIENT 连接分配一个唯一的服务器 IP 地址
- 各 MB_CLIENT 连接分配一个唯一的连接 ID
- 是否需要唯一的 IP 端口号取决于服务器组态

各个背景 DB 必须使用不同的连接 ID。总之，背景 DB 和连接 ID 成对使用，且对每个连接必须是唯一的。

表格 13-59 MB_CLIENT 背景数据块：用户可访问静态变量

变量	数据类型	默认值	说明
Blocked_Proc_Timeout	Real	3.0	在 Modbus 客户端实例受阻后，移除该激活的实例前需等待的时间（秒）。例如，当已发出客户端请求，但应用程序在彻底完成该请求前停止执行该客户端功能时，就会出现这种情况。最大 S7-1200 限值是 55 秒。
MB_Unit_ID	Word	255	Modbus 设备标识符 Modbus TCP 服务器通过其 IP 地址寻址。因此 MB_UNIT_ID 参数不用于 Modbus TCP 寻址。 MB_UNIT_ID 参数与 Modbus RTU 协议中的从站地址相对应。如果 Modbus TCP 服务器用于采用 Modbus RTU 协议的网关，MB_UNIT_ID 可用于标识在串行网络上连接的从站设备。MB_UNIT_ID 将用于将请求转发给正确的 Modbus RTU 从站地址。 某些 Modbus TCP 设备可能要求 MB_UNIT_ID 参数保持受限范围内。
RCV_TIMEOUT	Real	2.0	MB_CLIENT 等待服务器响应请求的时间（秒）。
已连接	Bool	0	指示与所分配服务器的连接是已接通还是已断开：1 = 接通，0 = 断开

表格 13-60 MB_CLIENT 协议错误

STATUS (W#16#)	发送到 Modbus 客户端的响应代码 (B#16#)	Modbus 协议错误
8381	01	不支持此功能代码
8382	03	数据长度错误
8383	02	数据地址错误或访问的数据超出 MB_HOLD_REG 地址区的界限
8384	03	数据值错误
8385	03	不支持该数据诊断代码（功能代码 08）

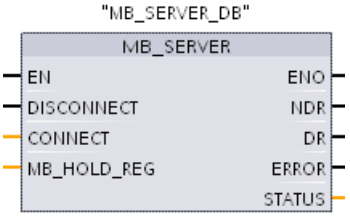
表格 13-61 MB_CLIENT 执行条件代码 [1]

STATUS (W#16#)	MB_CLIENT 参数错误
7001	MB_CLIENT 正在等待 Modbus 服务器响应指定 TCP 端口处的连接或断开连接请求。仅在第一次执行连接或断开操作时才返回此代码。
7002	MB_CLIENT 正在等待 Modbus 服务器响应指定 TCP 端口处的连接或断开连接请求。等待连接或断开操作完成时，将针对任何后续执行返回此代码。
7003	断开操作已成功完成（仅在一个 PLC 扫描周期内有效）。
80C8	服务器在指定的时间内未响应。MB_CLIENT 必须在分配的时间内使用最初传送的事务 ID 接收响应，否则将返回此错误。检查与 Modbus 服务器设备的连接。 尝试过重试操作（若适用）后，才返回此错误。
8188	模式无效
8189	数据地址无效
818A	数据长度无效
818B	指向 DATA_PTR 区的指针无效。可以是 MB_DATA_ADDRESS 与 MB_DATA_LEN 的组合。
818C	指针 DATA_PTR 指向未经优化的 DB 区（必须是未经优化的 DB 区或 M 存储区）
8200	端口正忙于处理现有的 Modbus 请求。
8380	接收到的 Modbus 帧不正确或接收到的字节太少。
8387	分配的连接 ID 参数和用于先前请求的 ID 不同。只能有一个单个连接 ID 与每个 MB_CLIENT 背景数据块配合使用。 如果从一个服务器接收到的 Modbus TCP 协议 ID 不是 0，该代码也可作为内部错误返回。
8388	Modbus 服务器返回一些和请求内容不同的数据。该代码仅适用于 Modbus 功能 15 或 16。

[1] 除了上面列出的 MB_CLIENT 错误外，也可以从底层传输块通信指令（TCON、TDISCON、TSEND 和 TRCV）返回错误。

MB_SERVER（作为 Modbus TCP 服务器通过 PROFINET 进行通信）指令

表格 13-62 MB_SERVER 指令

LAD/FBD	SCL	说明
	"MB_SERVER_DB"(DISCONNECT:=_bool_in_, CONNECT:=_variant_in_, NDR=>_bool_out_, DR=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_, MB_HOLD_REG:=_variant_inout_);	MB_SERVER 作为 Modbus TCP 服务器，通过 S7-1200 CPU 上的 PROFINET 端口进行通信。不需要额外的通信硬件模块。 MB_SERVER 可接收与 Modbus TCP 客户端的连接请求、接收 Modbus 功能请求并发送响应消息。

表格 13-63 参数的数据类型

参数和类型		数据类型	说明
DISCONNECT	IN	Bool	MB_SERVER 尝试与伙伴设备进行“被动”连接。也就是说，服务器被动地侦听来自任何请求 IP 地址的 TCP 连接请求。 如果 DISCONNECT = 0 且不存在连接，则可以启动被动连接。 如果 DISCONNECT = 1 且存在连接，则启动断开操作。该参数允许程序控制何时接受连接。每当启用此输入时，无法尝试其它操作。
CONNECT	IN	Variant	引用包含系统数据类型为“TCON_IP_v4”的连接参数的数据块结构。
MB_HOLD_REG	IN_OUT	Variant	指向 MB_SERVER Modbus 保持寄存器的指针：保持寄存器必须是一个未经优化的全局 DB 或 M 存储区地址。储存区用于保存允许 Modbus 客户端使用 Modbus 寄存器功能 3（读）、6（写）、16（写）和 23（写/读）访问的数据。
NDR	OUT	Bool	新数据就绪：0 = 没有新数据，1 = 表示 Modbus 客户端已写入新数据
DR	OUT	Bool	数据读取：0 = 没有读取数据，1 = 表示 Modbus 客户端已读取该数据。
ERROR	OUT	Bool	MB_SERVER 执行因错误而结束后，ERROR 位将在一个扫描周期时间内保持为 TRUE。STATUS 参数中的错误代码仅在 ERROR = TRUE 的一个循环周期内有效。
STATUS	OUT	Word	执行条件代码

说明

CPU 固件版本要求

本手册中所述的 Modbus TCP 指令要求固件版本为 V4.1 或更高版本。

说明

通过 MB_SERVER 指令使用功能 23

MB_SERVER 指令支持使用功能代码 23 在单个请求中写入和读取保持寄存器；然而 MB_CLIENT 指令并不支持该功能并返回错误代码。还应注意虽然请求中包含读和写，但是指令在读之前先处理写。

CONNECT 参数分配用于建立 PROFINET 连接的数据

必须使用全局数据块并存储所需的连接数据，然后才能在 CONNECT 参数中引用此 DB。

1. 创建新的全局 DB 或使用现有全局 DB 来存储 CONNECT 数据。可使用一个 DB 存储多个 TCON_IP_v4 数据结构。每个 Modbus TCP 客户端或服务器连接使用一个 TCON_IP_v4 数据结构。可在 CONNECT 参数中引用连接数据。
2. 使用有帮助的名称对 DB 和静态变量进行命名。例如，将数据块命名为“Modbus 连接”，将静态变量命名为“TCPpassive_1”（针对 Modbus TCP 服务器连接 1）。
3. 在 DB 编辑器的 “数据类型”(Data Type) 列中为示例静态变量“TCPactive_1”分配系统数据类型 “TCON_IP_v4”。
4. 扩展 TCON_IP_v4 结构，从而可以修改连接参数，如下图所示。
5. 修改 MB_SERVER 连接的 TCON_IP_v4 结构数据。
6. 输入 MB_SEVER CONNECT 参数的 DB 结构引用。本示例中应为“Modbus 连接”.TCPpassive_1。

Modbus connections

	名称	数据类型	启动值	注释
1	Static			
2	TCPpassive_1	TCON_IP_v4		
3	InterfaceId	HW_ANY	64	HW-identifier of IE-interface submodule
4	ID	CONN_OUC	1	connection reference / identifier
5	ConnectionType	Byte	16#0B	type of connetion: 11=TCP/IP, 19=UDP (17=TC...
6	ActiveEstablished	Bool	False	active/passive connection establishment
7	RemoteAddress	IP_V4		remote IP address (IPv4)
8	ADDR	array [1..4] of Byte		IPv4 address
9	ADDR[1]	Byte	192	
10	ADDR[2]	Byte	168	
11	ADDR[3]	Byte	2	
12	ADDR[4]	Byte	241	
13	RemotePort	UInt	0	remote UDP/TCP port number
14	LocalPort	UInt	502	local UDP/TCP port number

修改各 MB_SERVER 连接的 TCON_IP_V4 DB 数据

- InterfaceID：在设备组态窗口中单击 CPU PROFINET 端口图像。然后单击“常规” (General) 属性选项卡并使用该处显示的硬件标识符。
- ID：为该连接输入一个介于 1 和 4095 之间的唯一编号。使用底层 TCON、TDISCON、TSEND 和 TRCV 指令建立 Modbus TCP 通信，用于 OUC（开放式用户通信）。最多允许八个同步 OUC 连接。
- ConnectionType：对于 TCP/IP，使用默认值 16#0B（十进制值 = 11）。
- ActiveEstablished：该值必须为 0 或 FALSE。被动连接，MB_SERVER 正在等待 Modbus 客户端的通信请求。
- RemoteAddress：有两个选项。
  - 使用 0.0.0.0，则 MB_CLIENT 将响应来自任何 TCP 客户端的 Modbus 请求。
  - 输入目标 Modbus TCP 客户端的 IP 地址，则 MB_CLIENT 仅响应来自该客户端 IP 地址的请求。例如，如上图所示输入 192.168.2.241。
- RemotePort：对于 MB_SERVER 连接，该值必须为 0。
- LocalPort：默认值为 502。该编号为 MB_SERVER 试图连接和通信的 Modbus 客户端的 IP 端口号。一些第三方 Modbus 客户端要求使用其它端口号。

Modbus 和过程映像地址

MB_SERVER 允许进入的 Modbus 功能代码（1、2、4、5 和 15）在输入/输出过程映像中直接对位/字进行读/写。对于数据传输功能代码（3、6 和 16），MB_HOLD_REG 参数必须定义为大于一个字节的数据类型。下表显示了 Modbus 地址到 CPU 中过程映像的映射。

表格 13-64 Modbus 地址到过程映像的映射

Modbus 功能						S7-1200	
代码	功能	数据区	地址范围			数据区	CPU 地址
01	读位	输出	1	到	8192	输出过程映像	Q0.0 到 Q1023.7
02	读位	输入	10001	到	18192	输入过程映像	I0.0 到 I1023.7
04	读字	输入	30001	到	30512	输入过程映像	IW0 到 IW1022
05	写位	输出	1	到	8192	输出过程映像	Q0.0 到 Q1023.7
15	写位	输出	1	到	8192	输出过程映像	Q0.0 到 Q1023.7

进入的 Modbus 消息功能代码（3、6 和 16）在 Modbus 保持寄存器中读取/写入字，该寄存器可以在 M 存储区或数据块中。保持寄存器的类型由 MB_HOLD_REG 参数指定。

说明

MB_HOLD_REG 参数分配

定义为字数组、整数、宽字符、无符号整数、字节、短整数、无符号短整数、字符、双字、双整数、无符号双整数或实数的 Modbus 保持寄存器可以存放在任何存储区中。

定义为结构的 Modbus 保持寄存器必须存放在未经优化的 DB 中。

对于 M 存储器中的 Modbus 保持寄存器，使用 Any 指针格式。其格式为 P#“位地址” “数据类型” “长度” 。例如 P#M1000.0 WORD 500。

下表给出了 Modbus 地址到保持寄存器的映射示例，这种映射用于 Modbus 功能代码 03（读取字）、06（写入字）和 16（写入字）。DB 地址的实际上限取决于每种 CPU 型号的最大工作存储器限值和 M 存储器限值。

表格 13-65 Modbus 地址到 CPU 存储器地址的映射示例

Modbus 地址	MB_HOLD_REG 参数示例		
	P#M100.0 Word 5	P#DB10.DBx0.0 Word 5	"Recipe".ingredient
40001	MW100	DB10.DBW0	"Recipe".ingredient[1]
40002	MW102	DB10.DBW2	"Recipe".ingredient[2]
40003	MW104	DB10.DBW4	"Recipe".ingredient[3]
40004	MW106	DB10.DBW6	"Recipe".ingredient[4]
40005	MW108	DB10.DBW8	"Recipe".ingredient[5]

Modbus 应用协议标头

Modbus 应用协议标头是每个 Modbus TCP 消息的前 7 个字节。此标头包含事务标识符、协议标识符、长度和设备标识符。MB_SERVER 指令响应消息包含与 Modbus 请求消息中接收的事务标识符、协议标识符和设备标识符相同的值。 “长度”(Length) 字段由 MB_SERVER 指令算得。

多个服务器连接

可以创建多个服务器连接。单个 PLC 可与多个 Modbus TCP 客户端建立并发连接。

Modbus TCP 服务器支持的并发连接数最多为 PLC 允许的开放式用户通信最大连接数。PLC 的连接总数（包括 Modbus TCP 客户端和服务器）不得超过支持的开放式用户通信最大连接数 (页 807)。可在客户端和服务器类型的连接之间共享 Modbus TCP 连接。

单独的并发服务器连接必须遵循以下规则：

- 各 MB_SERVER 连接必须使用一个唯一的背景数据块。
- 必须为各 MB_SERVER 连接分配一个唯一的 IP 端口号。每个端口只能用于 1 个连接。
- 必须为各 MB_SERVER 连接分配一个唯一的连接 ID。
- 必须为每个连接（带有各自的背景数据块）单独调用 MB_SERVER。

连接 ID 对于每个单独的连接必须是唯一的。每个单独的背景 DB 必须使用单一的连接 ID。背景 DB 和连接 ID 成对使用，且对每个连接必须是唯一的。

表格 13-66 Modbus 诊断功能代码

MB_SERVER Modbus 诊断功能		
代码	子功能	说明
08	0x0000	返回查询数据回送测试：MB_SERVER 将向 Modbus 客户端回送接收到的数据字。
08	0x000A	清除通信事件计数器：MB_SEVER 将清除用于 Modbus 功能 11 的通信事件计数器。
11		获取通信事件计数器：MB_SERVER 使用内部通信事件计数器来记录发送到 Modbus 服务器的 Modbus 成功读取和写入请求次数。该计数器不会因任何功能 8、功能 11 请求或任何导致通信错误的请求而增加。 广播功能不能用于 Modbus TCP，因为在任何时刻仅存在一个客户端-服务器连接。

MB_SERVER 指令数据块 (DB) 变量

下表给出了存储在 MB_SERVER 背景数据块中的公共静态变量（可在用户程序中使用）。

表格 13-67 MB_SERVER 公共静态变量

变量	数据类型	默认值	说明
HR_Start_Offset	Word	0	指定 Modbus 保持寄存器的起始地址
Request_Count	Word	0	该服务器接收到的所有请求的数量。
Server_Message_Count	Word	0	该特定服务器接收到的请求的数量。
Xmt_Rcv_Count	Word	0	出现错误的传输或接收的数量。此外，如果接收到一条无效的 Modbus 消息，该值加 1。
Exception_Count	Word	0	需要返回例外的 Modbus 特定错误数
Success_Count	Word	0	该特定服务器接收到的且无协议错误的请求数。

变量	数据类型	默认值	说明
Connected	Bool	0	指示与所分配客户端的连接是已接通还是已断开：1 = 接通，0 = 断开
QB_Start	UInt	0	CPU 可写入的输出字节的起始地址（QB0 至 QB65535）
QB_Count	UInt	65535	远程设备可以写入的字节数。如果 QB_Count = 0，则远程设备无法写入输出。 示例：要想只允许 QB10 到 QB17 可写入，则 QB_Start = 10 且 QB_Count = 8。
QB_Read_Start	UInt	0	CPU 可读取的输出字节的起始地址（QB0 至 QB65535）
QB_Read_Count	UInt	65535	远程设备可以读取的输出字节数。如果 QB_Count = 0，则远程设备无法读取输出。示例：要想只允许 QB10 到 QB17 可读取，则 QB_Start = 10 且 QB_Count = 8。
IB_Read_Start	UInt	0	CPU 可读取的输入字节的起始地址（IB0 至 IB65535）
IB_Read_Count	UInt	65535	远程设备可以读取的输入字节数。如果 IB_Count = 0，则远程设备无法读取输入。示例：要想只允许 IB10 到 IB17 可读取，则 IB_Start = 10 且 IB_Count = 8。
NDR_immediate	Bool	FALSE	与参数 NDR（新数据就绪）含义相同。MB_SERVER 在处理 Modbus TCP 写请求的同一调用中更新“NDR_immediate”。
DR_immediate	Bool	FALSE	与参数 DR（数据读取）含义相同。MB_SERVER 在处理 Modbus TCP 写请求的同一调用中更新“DR_immediate”。

用户程序可以将数据写入控制 Modbus 服务器操作和以下变量中：

- HR_Start_Offset
- QB_Start
- QB_Count
- QB_Read_Start
- QB_Read_Count
- IB_Read_Start
- IB_Read_Count

MB_SERVER 指令数据块 (DB) 变量可用性的版本要求如下：

表格 13-68 MB_SERVER 指令数据块 (DB) 变量可用性的版本要求：指令、TIA Portal 和 S7-1200 CPU

MB_SERVER 指令版本	TIA Portal 的版本	S7-1200 CPU 固件 (FW) 版本	数据块变量
4.2	V14 SP1	CPU 固件 V4.0 或更高版本	QB_Start
			QB_Count
5.0 或以上版本	V15 或以上版本	CPU 固件 V4.2 或更高版本	QB_Start
			QB_Count
			QB_Read_Start
			QB_Read_Count
			IB_Read_Start
			IB_Read_Count
			NDR_immediate
			DR_immediate

HR_Start_Offset

Modbus 保持寄存器地址从 40001 开始。这些地址与保持寄存器的 PLC 存储器起始地址对应。不过，可以使用“HR_Start_Offset”变量将 Modbus 保持寄存器的起始地址定义为除 40001 外的其它数字。

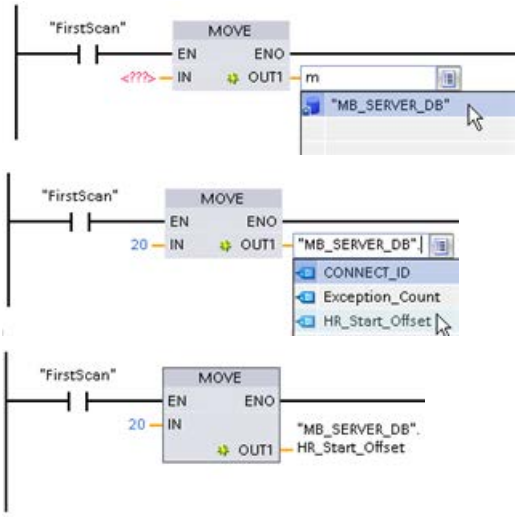
例如，如果保持寄存器起始于 MW100 且长度为 100 个字，偏移量 20 可指定保持寄存器的起始地址为 40021 而不是 40001。小于 40021 或大于 40119 的任何地址都将导致寻址错误。

表格 13-69 Modbus 保持寄存器寻址示例

HR_Start_Offset	地址	最小值	最大值
0	Modbus 地址（字）	40001	40099
	S7-1200 地址	MW100	MW298
20	Modbus 地址（字）	40021	40119
	S7-1200 地址	MW100	MW298

HR_Start_Offset 是 MB_SERVER 背景数据块中的一个字数据，用于分配 Modbus 保持寄存器的起始地址。将 MB_SERVER 放入程序后，可利用参数助手下拉列表设置该公共静态变量。

例如，将 MB_SERVER 放入 LAD 程序段后，可以切换到上一程序段，并分配 HR_Start_Offset。必须在执行 MB_SERVER 前分配起始地址。



使用默认 DB 名称

输入 Modbus 服务器变量：

1. 将光标放在参数字段中，然后输入 m 字符。
2. 从 DB 名称下拉列表中选择 “MB_SERVER_DB”。
3. 从 DB 变量下拉列表中选择 “MB_SERVER_DB.HR_Start_Offset”。

访问数据块 (DB) 中的数据区域，而不是直接访问 Modbus 地址

自 MB_SERVER 指令版本 V5.0 以及 S7-1200 CPU 固件 (FW) 版本 V4.2 起，用户可访问数据块中的数据区域，而不用直接访问过程映像和保持寄存器。为此，在全局 DB“属性” (Attributes) 属性页中，必须取消选中 “仅存储在装载存储器中”(Only store in load memory) 和 “优化块访问”(Optimized block access) 复选框。

如果 Modbus 请求到达时尚未为相应功能代码的 Modbus 数据类型定义数据区域，则 MB_SERVER 指令会按之前的指令版本处理请求：直接访问过程映像和保持寄存器。

如果已为功能代码的 Modbus 数据类型定义了数据区域，则 MB_SERVER 指令会对该数据区域进行读写操作。具体是读操作还是写操作取决于作业类型。

说明

如果组态了数据区域，则 MB_SERVER 指令会忽略与数据区域的 data_type 对应的背景数据块中，静态变量组态的偏移或范围。这些偏移和范围仅适用于 MB_HOLD_REG 所引用的过程映像或存储器。数据区域的启动和长度参数会提供自身定义偏移和范围的方法

单个 Modbus 请求只能对一个数据区域进行读写操作。如果要读取覆盖多个数据区域的保持寄存器，则需要多个 Modbus 请求。

数据区域的定义规则如下：

- 用户最多可在不同数据块中定义八个数据区域，每个数据块只能包含一个数据区域。单个 MODBUS 请求只能对恰好一个数据区域进行读写操作。每个数据区域对应于一个 MODBUS 地址区域。可以在实例数据块的“Data_Area_Array”静态变量中定义数据区域。
- 如果要使用的数据区域不到八个，则所需数据区域必须紧密相连，没有间隙。在处理过程中，数据区域中的第一个空白条目会终止数据区域搜索。例如，如果已定义字段元素 1、2、4 和 5，由于字段元素 3 留空，则“Data_Area_Array”只会识别字段元素 1 和 2。

- Data_Area_Array 字段包含八个元素：Data_Area_Array[1] 到 Data_Area_Array[8]
- 每个字段元素 Data_Area_Array[x]（其中 1 <= x <= 8）都是 MB_DataArea 类型的 UDT，结构如下：

参数	数据类型	含义
data_ type	UInt	映射到此数据区域的 MODBUS 数据类型的标识符： • 0：空字段元素或未使用数据区域的标识符。此时，数据块、起始和长度的值不相关。 • 1：过程映像输出（与功能代码 1、5 和 15 一起使用） • 2：过程映像输入（与功能代码 2 一起使用） • 3：保持寄存器（与功能代码 3、6 和 16 一起使用） • 4：输入寄存器（与功能代码 4 一起使用） 注：如果已定义 MODBUS 数据类型的数据区域，则指令 MB_SERVER 不能再直接访问此 MODBUS 数据类型。如果该数据类型的 MODBUS 请求地址与定义的数据区域不对应，则 STATUS 中会返回值 W#16#8383。
db	UInt	MODBUS 寄存器或后续定义的位所映射的目标数据块的编号 数据块编号在数据区域中必须是唯一的。不得在多个数据区域中定义相同的数据块编号。 在全局 DB“属性”(Attributes) 属性页中，必须取消选中“仅存储在装载存储器中”(Only store in load memory) 和“优化块访问”(Optimized block access) 复选框。 数据区域也是从数据块的字节地址 0 开始。 允许值：1 到 60999
起始	UInt	映射到数据块中的首个 MODBUS 地址（从地址 0.0 开始） 允许值：0 到 65535
长度	UInt	位数（对于 data_type 的值 1 和 2）或寄存器数量（对于 data_type 的值 3 和 4）。 相同 MODBUS 数据类型的 MODBUS 地址区域不得重叠。 允许值：1 到 65535

数据区域定义示例：

- 第一个示例：data_type = 3，db = 1，start = 10，length = 6
  CPU 将保持寄存器 (data_type = 3) 映射到数据块 1 (db = 1)，将 Modbus 地址 10 (start = 10) 置于数据字 0，将最后一个有效 Modbus 地址 15 (length = 6) 置于数据字 5。
- 第二个示例：data_type = 2，db = 15，start = 1700，length = 112
  CPU 将输入 (data_type = 2) 映射到数据块 15 (db = 15)，将 Modbus 地址 1700 (start = 1700) 置于数据字 0，将最后一个有效 Modbus 地址 1811 (length = 112) 置于数据字 111。

条件代码

表格 13-70 MB_SERVER 执行条件代码 [1]

STATUS (W#16#)	发送到 Modbus 服务器的响应代码 (B#16#)	Modbus 协议错误
7001		MB_SERVER 正在等待 Modbus 客户端连接到指定的 TCP 端口。仅在第一次执行连接或断开操作时才返回此代码。
7002		MB_SERVER 正在等待 Modbus 客户端连接到指定的 TCP 端口。等待完成连接或断开操作时，将针对任何后续执行返回此代码。
7003		断开操作已成功完成（仅在一个 PLC 扫描周期内有效）。
8187		MB_HOLD_REG 无效，可能指向优化的 DB 或小于 2 个字节的区域。
818C		指针 MB_HOLD_REG 指向未经优化的 DB 区（必须是未经优化的全局 DB 区或 M 存储区）或受阻过程的超时时间超过 55 秒限值。（仅适用于 S7-1200）
8381	01	不支持此功能代码
8382	03	数据长度错误
8383	02	数据地址错误或访问的数据超出 MB_HOLD_REG 地址区的界限
8384	03	数据值错误

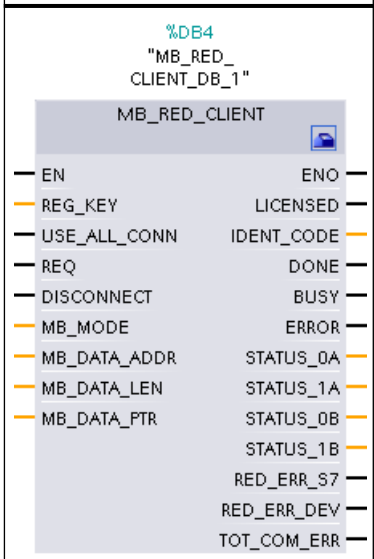
STATUS (W#16#)	发送到 Modbus 服务器的响应代码 (B#16#)	Modbus 协议错误
8385	03	不支持该数据诊断代码（功能代码 08）
8389		数据区域定义无效： ● data_type 值无效 ● 数据块编号无效或不存在： – db 值无效 – 数据块编号不存在 – 数据块编号已由另一个数据区域使用 – 数据块具有优化访问权限 – 数据块不在工作存储器中 ● 长度值无效 ● 属于同一 MODBUS 数据类型的 MODBUS 地址范围重叠

[1] 除了上面列出的 MB_SERVER 错误外，也可以从底层传输块通信指令（TCON、TDISCON、TSEND 和 TRCV）返回错误。

MB_RED_CLIENT（作为 Modbus TCP 客户端通过 PROFINET 进行冗余通信）

使用该指令可在 S7-1200 CPU 与支持 Modbus TCP 协议的设备之间建立连接。

表格 13-71 MB_RED_CLIENT 指令

LAD/FBD	SCL	说明
	"MB_RED_CLIENT_DB"(REG_KEY:=_string_in_, USE_ALL_CONN:=_bool_in_ REQ:=_bool_in_, DISCONNECT:=_bool_in_, MB_MODE:=_usint_in_, MB_DATA_ADDR:=_udint_in_, MB_DATA_LEN:=_uint_in_, LICENSED=>_bool_out_ IDENT_CODE=>_string_out_ DONE=>_bool_out_, BUSY=>_bool_out_, ERROR=>_bool_out_, STATUS_0A=>_word_out_, STATUS_1A=>_word_out_, STATUS_0B=>_word_out_, STATUS_1B=>_word_out_, RED_ERR_S7=>_bool_out_, RED_ERR_DEV=>_bool_out_, TOT_COM_ERR=>_bool_out_, MB_DATA_PTR:=_variant_inout_);	MB_RED_CLIENT 指令作为 Modbus TCP 客户端通过 PROFINET 连接进行通信。 可使用指令 MB_RED_CLIENT 在客户端和服务器之间建立冗余连接、发送 Modbus 请求、接收响应并通过 Modbus TCP 客户端控制连接终止。

表格 13-72 参数的数据类型

参数和类型		数据类型	说明
REG_KEY [1]	IN	STRING[17]	授权注册码 必须在每个 CPU 上分别对 MB_RED_CLIENT 指令进行授权。
USE_ALL_CONN	IN	Bool	指定用于发送帧的组态连接数目： • 0：通过一个连接发送帧，只有在发生错误的情况下才切换到下一个连接 • 1：通过所有组态连接发送帧

参数和类型		数据类型	说明
REQ	IN	Bool	针对 Modbus TCP 服务器的 Modbus 查询 REQ 为电平控制参数。这意味着只要输入置位 (REQ = TRUE)，指令就会发送通信请求。如果连接尚未建立，此时便会建立，且随后立即发送 Modbus 帧。 在服务器响应或输出错误消息之前，对输入参数的更改不会生效。 如果在 Modbus 请求期间再次设置了参数 REQ，此后将不会进行任何其它传输。
DISCONNECT	IN	Bool	借助该参数，可以控制 Modbus 服务器连接的建立和终止： ● 0：与通过 CONNECT 参数组态的连接伙伴（请参见 CONNECT 参数）建立通信连接。 ● 1：断开通信连接。在终止连接的过程中，不执行任何其它功能。成功终止连接后，STATUS_x 参数将输出值 0003。
MB_MODE [2]	IN	USInt	选择 Modbus 请求模式（读取、写入或诊断）或直接选择 Modbus 功能
MB_DATA_ADDR [2]	IN	UDInt	取决于 MB_MODE 的 Modbus 地址
MB_DATA_LEN	IN	UInt	数据长度：数据访问的位数或寄存器数
MB_DATA_PTR [2]	IN_OUT	Variant	指向待从 Modbus 服务器接收的数据或待发送到 Modbus 服务器的数据所在数据缓冲区的指针。
LICENSED [1]	OUT	Bool	● 0：指令未获授权 ● 1：指令已获授权
IDENT_CODE [1]	OUT	STRING[18]	授权标识。使用此字符串申请 REG_KEY 注册码。
DONE	OUT	Bool	如果激活的 Modbus 作业在至少一个连接上成功完成，则输出参数 DONE 中的该位将立即置“1”。
BUSY	OUT	Bool	● 0：无正在进行的 Modbus 请求 ● 1：正在处理 Modbus 请求 在建立和终止连接期间，输出参数 BUSY 不会置位。
ERROR	OUT	Bool	● 0：无错误 ● 1：激活的 Modbus 作业在任何组态的连接上都不能成功传输。出错原因由参数 STATUS_x 指示。
STATUS_0A [3]	OUT	Word	连接 0A 上指令的详细状态信息。
STATUS_1A [3]	OUT	Word	连接 1A 上指令的详细状态信息。
STATUS_0B [3]	OUT	Word	连接 0B 上指令的详细状态信息。

参数和类型		数据类型	说明
STATUS_1B [3]	OUT	Word	连接 1B 上指令的详细状态信息。
RED_ERR_S7 [3]	OUT	Bool	• 0：SIMATIC 中无冗余错误 • 1：SIMATIC 中存在冗余错误
RED_ERR_S7 [3]	OUT	Bool	• 0：链接伙伴侧无冗余错误 • 1：链接伙伴侧存在冗余错误
RED_ERR_S7 [3]	OUT	Bool	• 0：至少已建立 1 个组态连接 • 1：完全丢失通信，所有组态连接均终止

[1] 有关详细信息，参见下文中“授权”部分。

[2] 有关详细信息，参见下文中“输入参数：MB_MODE、MB_DATA_ADDR、MB_DATA_LEN 和 MB_DATA_PTR”部分。

[3] 有关详细信息，参见下文中“输出参数：STATUS_x、RED_ERR_S7、RED_ERR_DEV 和 TOT_COM_ERR”部分。

说明

MB_RED_CLIENT 调用过程中输入数据保持一致

调用 Modbus 客户端指令时，输入参数的值将存储在内部。在处理帧时，这些值不能更改。

说明

CPU 固件版本要求

手册本部分所述的 Modbus TCP 指令要求固件版本为 V4.2 或更高版本。
使用该指令时，无需其它任何硬件模块。

多个客户端连接

CPU 可处理多个 Modbus TCP 客户端连接。连接的最大数目取决于所使用的 CPU，具体请参见相关 CPU 的技术规范。一个 CPU 的总连接数，包括 Modbus TCP 客户端和服务器的连接数，不能超过所支持的最大连接数。

使用各客户端连接时，请记住以下规则：

- 每个 MB_RED_CLIENT 连接都必须使用唯一的背景数据块。
- 对于每个 MB_RED_CLIENT 连接，必须指定唯一的服务器 IP 地址。
- 每个 MB_RED_CLIENT 连接都需要一个唯一的连接 ID。连接 ID 在整个 CPU 范围内必须唯一。

运行和冗余

通信节点可设计为独立式或者冗余式。如果其中一个伙伴采用独立式设计，则将其称为单侧冗余。如果两个伙伴均为冗余式设计，则将其称为双侧冗余：

- 单侧冗余：
  - 说明：必须为通信伙伴之间的每个连接组态一个连接。**SIMATIC S7** 的连接点称为 **0** 和 **1**；**通信伙伴**的连接点称为 **A** 和 **B**。R-CPU 或 H-CPU 1 是指连接点 0，R-CPU 或 H-CPU 2 为连接点 1。
  - 组态：如果 S7 采用冗余设计，则为 S7 连接点 0 到链接伙伴的节点 A 创建一个连接（从 S7 连接点 **0** 到伙伴/节点 **A** 的连接 => 连接 **0A**），一个从 S7 连接点 1 到链接伙伴的节点 A 的连接（从 S7 连接点 **1** 到伙伴/节点 **A** 的连接 => 连接 **1A**）。下图给出了连接名称：

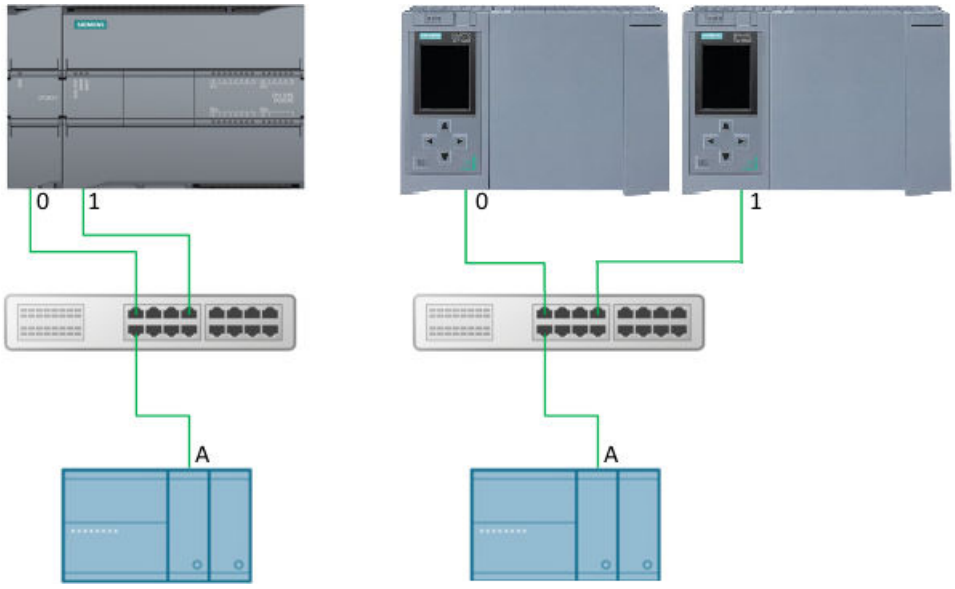


图 13-1　单侧冗余 S7

– 如果 S7 采用独立式设计，链接伙伴采用冗余设计，则为 S7 连接点 0 到链接伙伴的节点 A 创建一个连接（从 S7 连接点 **0** 到伙伴/节点 **A** 的连接 => 连接 **0A**），一个从 S7 连接点 0 到链接伙伴的节点 B 的连接（从 S7 连接点 **0** 到伙伴/节点 **B** 的连接 => 连接 **0B**）。下图给出了连接名称：

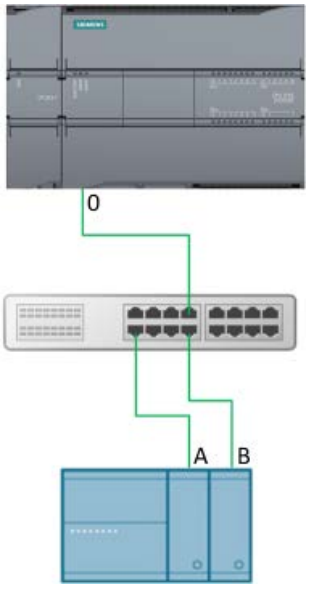


图 13-2　单侧冗余伙伴

- 双侧冗余：
  – 说明：必须为通信伙伴之间的每个连接组态一个连接。SIMATIC S7 的连接点称为 **0** 和 **1**；**通信伙伴**的连接点称为 **A** 和 **B**。
  R-CPU 或 H-CPU 1 是指连接点 0，R-CPU 或 H-CPU 2 为连接点 1。

– 组态：采用双侧冗余时，分别从 S7 连接点 0 创建两个连接（从 S7 连接点 **0** 到伙伴/节点 **A** 的连接 => 连接 **0A**，从 S7 连接点 **0** 到伙伴/节点 **B** 的连接 => 连接 **0B**），从连接点 1 创建两个连接到链接伙伴的节点 A 和 B（从 S7 连接点 **1** 到伙伴/节点 **A** 的连接 => 连接 **1A**，从 S7 连接点 **1** 到伙伴/节点 **B** 的连接 => 连接 **1B**）。下图给出了连接名称：

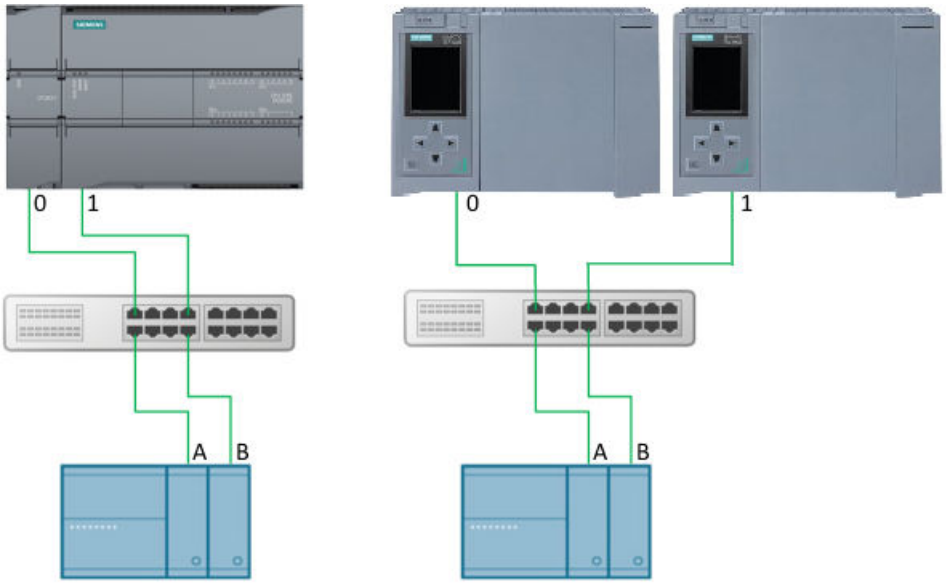


图 13-3　双侧冗余

- 帧处理：可通过一个或所有组态连接发送帧：
  - 通过一个连接发送帧：如果设置 USE_ALL_CONN = FALSE，将通过一个当前有效的连接发送 MODBUS 帧。发生超时（服务器无响应）或者连接故障时，会尝试通过其它（最多 4 个）组态连接发送帧。顺序为 0A、1A、0B 和 1B。如果已通过一个连接成功发送帧，则将此连接标记为“有效”，其它帧通信也通过此连接进行。有效连接发生连接故障时，会再次尝试通过所有组态连接发送帧。如果所有发送尝试均失败，ERROR 和 STATUS_x 会相应地置位。
    如果接收到响应帧，会执行真实性检查。如果检查成功，则执行所需操作，成功执行作业后，输出 DONE 置位。如果在检查期间检测到错误，则作业正常结束，位 ERROR 置位，并在 STATUS_x 中显示错误编号。在这种情况下，不会再尝试通过下一个组态连接发送帧。只有在检测到连接故障或者未接收到响应时，才会切换到其它组态连接。
  - 通过所有连接发送帧：如果设置 USE_ALL_CONN = TRUE，则将通过所有组态的既有连接发送 MODBUS 帧。当其中一个连接上接收到响应帧后，将执行有效性检查。如果检查成功，将执行所需操作。如果至少有一个连接上接收到有效响应帧，则输出 DONE 置位。
- 冗余输出 RED_ERR_S7、RED_ERR_DEV 和 TOT_COM_ERR：

– 冗余位 RED_ERR_S7、RED_ERR_DEV 和 TOT_COM_ERR 会根据状态输出的状态置位：

Number of faulty connections	STATUS_0A	STATUS_0B	STATUS_1A	STATUS_1B	RED_ERR_S7	RED_ERR_DEV	TOT_COM_ERR
0	okay	okay	okay	okay	FALSE	FALSE	FALSE
1	okay	okay	okay	Error	FALSE	FALSE	FALSE
	okay	okay	Error	okay	FALSE	FALSE	FALSE
	okay	Error	okay	okay	FALSE	FALSE	FALSE
	Error	okay	okay	okay	FALSE	FALSE	FALSE
2	okay	okay	Error	Error	TRUE	FALSE	FALSE
	okay	Error	okay	Error	FALSE	TRUE	FALSE
	Error	okay	okay	Error	FALSE	FALSE	FALSE
	okay	Error	Error	okay	FALSE	FALSE	FALSE
	Error	okay	Error	okay	FALSE	TRUE	FALSE
	Error	Error	okay	okay	TRUE	FALSE	FALSE
3	Error	Error	Error	okay	TRUE	TRUE	FALSE
	Error	Error	okay	Error	TRUE	TRUE	FALSE
	Error	okay	Error	Error	TRUE	TRUE	FALSE
	okay	Error	Error	Error	TRUE	TRUE	FALSE
4	Error	Error	Error	Error	TRUE	TRUE	TRUE

图 13-4　两侧冗余设置的中断位显示

Number of faulty connections	STATUS_0A	STATUS_0B	STATUS_1A	STATUS_1B	RED_ERR_S7	RED_ERR_DEV	TOT_COM_ERR
0	okay	0AFF	okay	0AFF	FALSE	FALSE	FALSE
1	okay	0AFF	Error	0AFF	TRUE	TRUE	FALSE
	Error	0AFF	okay	0AFF	TRUE	TRUE	FALSE
2	Error	0AFF	Error	0AFF	TRUE	TRUE	TRUE

图 13-5　一侧冗余设置的中断位显示

说明

客户端和服务器的端口号

Modbus 客户端使用的端口号从 2000 开始。通常通过端口号 502 对 Modbus 服务器进行寻址。

参数分配

对于 S7-1200，可以使用指令 MB_RED_CLIENT **V1.0** 和 **V1.1**。CPU 通过 CPU 或 CM/CP 的本地接口建立连接。CPU 使用 TCON_IP_V4 结构组态和建立连接。

MB_RED_CLIENT 的组态： 使用 MB_RED_CLIENT 指令的组态对话框进行以下设置：

- 连接 0A、1A、0B 和 1B 的连接参数（有关冗余组态的更多信息，请参见上文的“运行和冗余”。）
- 内部参数（可选）

通过 MB_RED_CLIENT 指令或通过工艺对象，可以打开组态对话框。

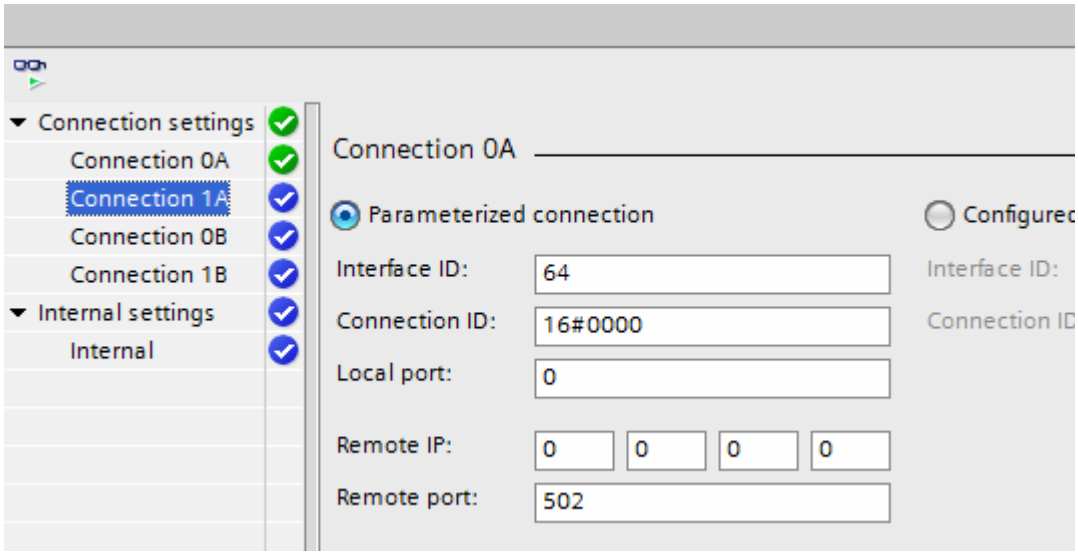


图 13-6　参数化的客户端连接

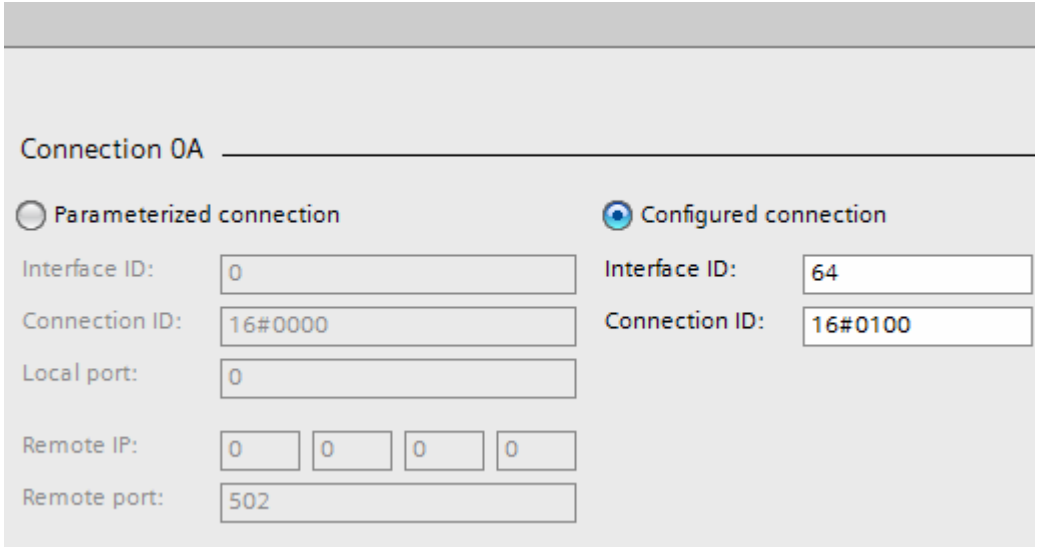


图 13-7　组态的客户端连接

变量	起始值	说明
组态的连接		
Interface ID	64	所用 PN 接口的硬件标识符
Connection ID	16#0000	所用连接的连接 ID 这些连接 ID 在整个 CPU 范围内必须唯一。
Local port	0	客户端的本地端口号。默认情况下不为客户端输入端口号。
Remote IP	0.0.0.0	服务器的远程 IP 地址
Remote port	502	服务器的远程端口号 Modbus/TCP 服务器的默认端口为 502。
组态的连接		
Interface ID	64	所用 PN 接口的硬件标识符
Connection ID	16#0000	所用连接的连接 ID 在网络视图中对这些连接进行组态。

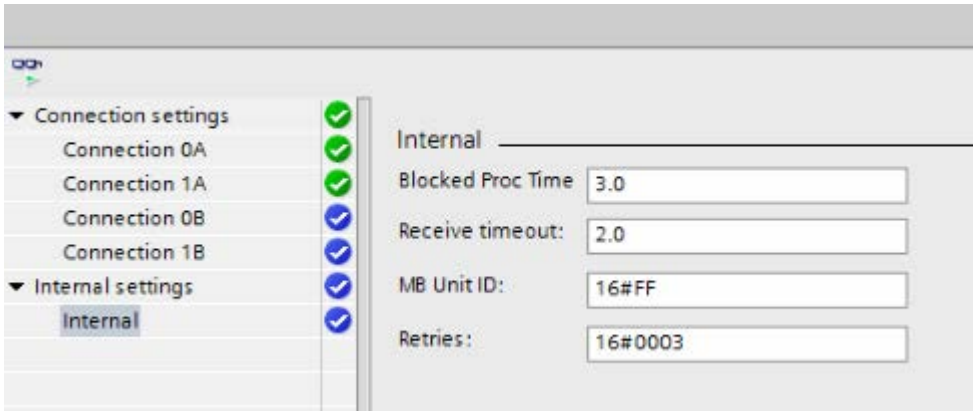


图 13-8　内部参数（可选）

变量	数据类型	起始值	说明
Blocked Proc Time	REAL	3.0	如果存在被阻止的 Modbus 实例，则为复位静态变量 ACTIVE 前的等待时间（单位为秒）。例如，如果输出了一个客户端请求，而且在该请求完全执行之前中止执行客户端功能，则可能发生这种情况。等待时间必须介于 0.5 s 到 55 s 之间。
Receive timeout	REAL	2.0	“MB_RED_CLIENT”指令等待服务器响应的时间间隔（单位为秒）。该值必须介于 0.5 s 到 55 s 之间。
MB_Unit_ID	BYTE	255	Modbus 设备检测： Modbus TCP 服务器使用其 IP 地址寻址。因此，寻址 Modbus TCP 时不会使用 MB_UNIT_ID 参数。 MB_UNIT_ID 参数对应于 Modbus RTU 协议的从站地址域。如果 Modbus/TCP 服务器用作 Modbus RTU 协议的网关，则可以使用 MB_UNIT_ID 标识串行网络中的从站设备。这种情况下，MB_UNIT_ID 参数会将请求转发给正确的 Modbus RTU 从站地址。 请注意，某些 Modbus/TCP 设备可能需要 MB_UNIT_ID 参数，以在有限的值范围内进行初始化。
Retries	WORD	3	指令 MB_RED_CLIENT 返回错误 W#16#80C8 之前尝试发送的次数。

说明

变量 MB_Transaction_ID

如果 Modbus TCP 服务器应答中的事务 ID 与 MB_RED_CLIENT 作业中的事务 ID 不一致，则指令 MB_RED_CLIENT 将等待一段时间 (RCV_TIMEOUT * RETRIES)，等待事务 ID 正确的 Modbus TCP 服务器进行应答；超出该时间后，指令将返回错误 W#16#80C8。

授权

必须在每个 CPU 上分别对 MB_RED_CLIENT 指令进行授权，且该指令需要付费。授权分为两个步骤：

- 显示许可证 IDENT_CODE
- 输入 REG_KEY 注册密钥：必须在每个 MB_RED_CLIENT 指令中分配 REG_KEY 注册密钥。将 REG_KEY 保存在共享数据块中，所有 MB_RED_CLIENT 指令均可通过该数据块接收所需的注册密钥。

显示许可证 IDENT_CODE 的程序：

1. 根据循环 OB 中的要求，为 MB_RED_CLIENT 指令分配参数。将程序下载到 CPU，并将 CPU 设置为 RUN。
2. 打开 Modbus 指令的背景数据块，然后单击“监视全部”(Monitor all) 按钮。
3. 背景数据块将在 IDENT_CODE 输出中显示一个 18 位的字符串。

	Name	Data type	Start value	Monitor value
1	Input			
2	REG_KEY	String[17]	''	''
3	USE_ALL_CONN	Bool	false	FALSE
4	REQ	Bool	false	FALSE
5	DISCONNECT	Bool	false	FALSE
6	MB_MODE	USInt	0	0
7	MB_DATA_ADDR	UDInt	0	0
8	MB_DATA_LEN	UInt	0	0
9	Output			
10	LICENSED	Bool	false	FALSE
11	IDENT_CODE	String[18]	''	'RTPCFIGDCDIIHJHAH4'
12	DONE	Bool	false	FALSE
13	BUSY	Bool	false	FALSE
14	ERROR	Bool	false	FALSE

图 13-9　许可证

4. 使用复制/粘帖功能，从数据块中复制该字符串，并粘贴到表格中（在订购产品后以电子邮件形式发送给客户的表格，或包含在 CD 光盘中）。
5. 使用服务请求将表格发送至客户支持 (https://support.industry.siemens.com/my/ww/en/requests/#createRequest)。随后用户将收到 CPU 的注册密钥。

输入注册密钥 REG_KEY 的程序：

1. 通过“添加新块...”(Add new block…) 插入具有唯一符号名的全新共享数据块，例如“License_DB”。
2. 在该块中创建数据类型为 STRING[17] 的 REG_KEY 参数。

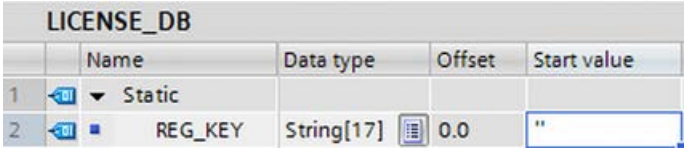


图 13-10　REG KEY

3. 使用复制/粘帖将已发送的 17 位注册密钥复制到“起始值”(Start value) 列。
4. 在循环 OB 中，在 MB_RED_CLIENT 指令的参数 REG_KEY 中，输入许可证数据块的名称和字符串名称（例如 License_DB.REG_KEY）。
5. 将修改后的块下载到 CPU。可以在运行时间输入注册密钥；不需要从 STOP 更改 RUN。
6. 现在，该 CPU 已获得使用 MB_RED_CLIENT 指令进行 Modbus/TCP 通信的授权，LICENSED 输出位为 TRUE。

授权缺失或不正确的纠正程序：

- 如果输入的注册密钥错误或未输入注册密钥，则 CPU 的 ERROR LED 指示灯闪烁。此外，S7-1200 CPU 会在诊断缓冲区中生成有关许可证缺失的循环条目。

图 13-11　诊断缓冲区

- 如果注册密钥缺失或不正确，CPU 仍将处理 Modbus/TCP 通信，但始终会在 STATUS_x 输出中显示“W#16#0A90”（无适用于功能性包装的有效许可证密钥）。输出位 LICENSED 为 FALSE。

输入参数：MB_MODE、MB_DATA_ADDR、MB_DATA_LEN 和 MB_DATA_PTR

MB_MODE、MB_DATA_ADDR 和 MB_DATA_LEN 参数的组合定义了当前 Modbus 消息中所使用的功能代码。

- **MB_MODE** 中包含有关是否进行读写操作的信息：
  读取：MB_MODE = 0、101、102、103 和 104
  写入：MB_MODE = 1、2、105、106、115 和 116（注意：MB_MODE = 2 时，Modbus 功能 15 和 05 或 Modbus 功能 16 和 06 无区别。）
- **MB_DATA_ADDR** 中包含有关待读取/写入的目标信息，以及 MB_RED_CLIENT 指令用于计算远程地址的地址信息。
- **MB_DATA_LEN** 包含待读取/写入的值的数量。

下表显示了“MB_RED_CLIENT”指令的输入参数 MB_MODE、MB_DATA_ADDR、MB_DATA_LEN 与 Modbus 功能之间的关系：

MB_MODE	MB_DATA_ADDR	MB_DATA_LEN	Modbus 功能	功能和数据类型
0	1 到 9999	1 到 2000	01	在远程地址 0 到 9998 处，读取 1 到 2000 个输出位
0	10001 到 19999	1 到 2000	02	在远程地址 0 到 9998 处，读取 1 到 2000 个输入位
0	• 40001 到 49999 • 400001 到 465535	1 到 125	03	• 在远程地址 0 到 9998 处，读取 1 到 125 个保持性寄存器 • 在远程地址 0 到 65534 处，读取 1 到 125 个保持性寄存器
0	30001 到 39999	1 到 125	04	在远程地址 0 到 9998 处，读取 1 到 125 个输入字
1	1 到 9999	1	05	在远程地址 0 到 9998 处，写入 1 个输出位
1	• 40001 到 49999 • 400001 到 465535	1	06	• 在远程地址 0 到 9998 处，写入 1 个保持性寄存器 • 在远程地址 0 到 65534 处，写入 1 个保持性寄存器
1	1 到 9999	2 到 1968	15	在远程地址 0 到 9998 处，写入 2 到 1968 个输出位

MB_MODE	MB_DATA_ADDR	MB_DATA_LEN	Modbus 功能	功能和数据类型
1	• 40001 到 49999 • 400001 到 465535	2 到 123	16	• 在远程地址 0 到 9998 处，写入 2 到 123 个保持性寄存器 • 在远程地址 0 到 65534 处，写入 2 到 123 个保持性寄存器
2	1 到 9999	1 到 1968	15	在远程地址 0 到 9998 处，写入 1 到 1968 个输出位
2	• 40001 到 49999 • 400001 到 465535	1 到 123	16	• 在远程地址 0 到 9998 处，写入 1 到 123 个保持性寄存器 • 在远程地址 0 到 65534 处，写入 1 到 123 个保持性寄存器
11	执行该功能时，指令不会评估 MB_DATA_ADDR 和 MB_DATA_LEN 参数。		11	读取服务器的状态字和事件计数器： • 状态字反映了处理状态（0 - 未处理，0xFFFF - 正在处理）。 • Modbus 请求成功执行时，事件计数器将递增。如果执行 Modbus 功能时出错，则服务器将发送消息，但不会递增事件计数器。
80	-	1	08	通过诊断代码 0x0000 检查服务器状态（返回循环测试 - 服务器发回请求）： 每次调用 1 个 WORD
81	-	1	08	通过诊断代码 0x000A 复位服务器的事件计数器： 每次调用 1 个 WORD
101	0 到 65535	1 到 2000	01	在远程地址 0 到 65535 处，读取 1 到 2000 个输出位
102	0 到 65535	1 到 2000	02	在远程地址 0 到 65535 处，读取 1 到 2000 个输入位
103	0 到 65535	1 到 125	03	在远程地址 0 到 65535 处，读取 1 到 125 个保持性寄存器
104	0 到 65535	1 到 125	04	在远程地址 0 到 65535 处，读取 1 到 125 个输入字
105	0 到 65535	1	05	在远程地址 0 到 65535 处，写入 1 个输出位
106	0 到 65535	1	06	在远程地址 0 到 65535 处，写入 1 个保持性寄存器

MB_MODE	MB_DATA_ADDR	MB_DATA_LEN	Modbus 功能	功能和数据类型
115	0 到 65535	1 到 1968	15	在远程地址 0 到 65535 处，写入 1 到 1968 个输出位
116	0 到 65535	1 到 123	16	在远程地址 0 到 65535 处，写入 1 到 123 个保持性寄存器
3 到 10、12 到 79、82 到 100、107 到 114、117 到 255				保留

示例：

变量	含义
MB_MODE = 1 MB_DATA_ADDR = 1 MB_DATA_LEN = 1	功能代码是 5 时，将从远程地址 0 开始写入 1 个输出位。
MB_MODE = 1 MB_DATA_ADDR = 1 MB_DATA_LEN = 2	功能代码是 15 时，将从远程地址 0 开始写入 2 个输出位。
MB_MODE = 104 MB_DATA_ADDR = 17834 MB_DATA_LEN = 125	功能代码是 4 时，将从远程地址 17.834 开始读取 125 个输入字。

MB_DATA_PTR：

MB_DATA_PTR 参数是一个指针，指向待从 Modbus 服务器接收的数据或待发送到 Modbus 服务器的数据所在的数据缓冲区。可使用全局数据块或存储区域 (M) 作为数据缓冲区。

对于存储区域 (M) 中的缓冲区，可通过以下方式使用 ANY 格式的指针：“P#位地址” “数据类型” “长度” （例如：P#M1000.0 WORD 500）。

MB_DATA_PTR 可引用不同的数据结构，具体取决于数据缓冲区所在的存储区域：

- 使用具有优化访问权限的全局数据块时，MB_DATA_PTR 可引用基本数据类型的变量或数组。支持以下数据类型：

数据类型	长度（位）
Bool	1
Byte、SInt、USInt、Char	8
Word、Int、WChar、UInt	16
DWord、DInt、UDInt、Real	32

  可使用所有 Modbus 功能支持的所有数据类型。例如，MB_RED_CLIENT 可将接收到的字节类型变量中的某个位写入一个特定地址内，而不改变该字节中的其它位。因此，在执行位操作功能时，无需使用位数组。

- 如果使用位存储器地址区或具有标准访问的全局数据块作为存储区，则 MB_DATA_PTR 对基本数据类型的使用无任何限制；此时，MB_DATA_PTR 还可引用诸如 PLC 数据类型 (UDT) 和系统数据类型 (SDT) 等复杂数据类型。

说明

将位存储器地址区域用作数据缓冲区

如果将位存储器地址区域用作 MB_DATA_PTR 数据缓冲区，则需注意该变量的值。对于 S7-1200 CPU，为 8 KB。

输出参数：STATUS_x、RED_ERR_S7、RED_ERR_DEV 和 TOT_COM_ERR

CPU 将错误消息显示在 MB_RED_CLIENT 指令的状态输出中：

说明

在程序编辑器中，错误状态代码可显示为整数或十六进制值：

1. 在程序编辑器中打开所需的块。
2. 单击“启用/禁用监视”(Monitor on/off) 切换程序状态的显示格式。（如果尚未建立在线连接，则打开“转至在线”(Go online) 对话框。在此对话框中，可以建立在线连接。）
3. 选择待监视的变量，然后在快捷菜单的“显示格式”(Display format) 中，选择所需的显示格式。

- STATUS_x 参数（常规状态信息）

STATUS (W#16#)	说明
0000	指令已执行，且无任何错误。
0001	连接已建立。
0003	连接已终止。
0A90	MB_RED_CLIENT 指令未授权。有关详细信息，参见上文中“授权”部分。
0AFF	连接未组态且未使用。必须组态“0A”连接。
7000	未激活任何作业且未建立任何连接（REQ=0，DISCONNECT=1）。
7001	已触发连接建立操作。
7002	中间调用。正在建立连接。
7003	正在终止连接。
7004	连接已建立且处于受监视状态。未激活任何作业处理。
7005	正在发送数据。
7006	正在接收数据。

- STATUS_x 参数（协议错误）

STATUS (W#16#)	说明
80C8	在指定时间段内，服务器无响应。检查与 Modbus 服务器的连接。仅当尝试了所组态的可重复尝试次数后，才报告该错误。 如果“MB_RED_CLIENT”指令在指定时间内没有收到最初传输事务 ID（请参见静态变量 MB_TRANSACTION_ID）的应答，则输出该错误代码。
8380	接收到的 Modbus 帧格式错误或字节数过少。
8382	• 帧头中的 Modbus 帧长度与接收到的字节数不一致。 • 字节数目与实际传送的字节数不一致（仅功能 1 到 4）。 • 收到的帧的起始地址与已经保存的起始地址不一致（功能 5、6、15 和 16）。 • 字数与实际传送的字数不一致（功能 15 和 16）。
8383	读/写数据错误或访问了 MB_DATA_PTR 地址外的区域。有关详细信息，请参见上文“MB_DATA_PTR”部分。
8384	• 接收到无效的异常代码。 • 接收到的数据值与最初由客户端发送的不同（功能 5、6 和 8）。 • 接收到无效状态值（功能 11）

STATUS (W#16#)	说明
8385	• 诊断代码不支持。 • 接收到的子功能代码与最初由客户端发送的不同（功能 8）。
8386	接收到的功能代码与最初发送的代码不一致。
8387	服务器接收到的 Modbus TCP 帧协议 ID 不为“0”。
8388	Modbus 服务器发送的数据长度与所处理的不同。只有使用 Modbus 功能 5、6、15 或 16 时，才会发生该错误。

- STATUS_x 参数（参数错误）

STATUS (W#16#)	说明
80B6	无效连接类型；仅支持 TCP 连接。
80BB	参数 ActiveEstablished 的值无效。只允许对客户端建立主动连接 (ActiveEstablished = TRUE)。
8188	参数 MB_MODE 的值无效。
8189	参数 MB_DATA_ADDR 中的数据地址无效。
818A	参数 MB_DATA_LEN 中的数据长度无效。
818B	参数 MB_DATA_PTR 的指针无效。请检查 MB_DATA_ADDR 和 MB_DATA_LEN 参数的值。有关“MB_DATA_ADDR”的更多信息，请参见上文“MB_DATA_ADDR”部分。
818C	参数 BLOCKED_PROC_TIMEOUT 或 RCV_TIMEOUT 超时（参见指令的静态变量）。BLOCKED_PROC_TIMEOUT 和 RCV_TIMEOUT 必须介于 0.5 s 和 55.0 s 之间。
8200	• CPU 当前正在通过该端口处理另一个 Modbus 请求。 • MB_RED_CLIENT 具有相同连接参数的另一实例正在处理现有 Modbus 请求。

说明

内部使用的通信指令的错误代码

对于 MB_RED_CLIENT 指令，除了会发生表中列出的错误外，还会发生由该指令所使用的通信指令（TCON、TDISCON、TSEND、TRCV、T_DIAG 和 TRESET）所引起的错误。

CPU 通过 MB_RED_CLIENT 指令的背景数据块来指定错误代码。CPU 将相应指令的错误代码显示在 "Static" 部分中的 STATUS 下。

有关错误代码的含义，请参见相应通信指令文档。

说明

发送或接收数据时发生通信错误

如果发送或接收数据时发生通信错误（80C4（临时通信错误。指定的连接临时终止。）、80C5（远程伙伴主动终止连接。）和 80A1（指定的连接中断或尚未建立。）），CPU 现有连接会终止。

即，可查看连接终止时所有返回的 STATUS 值，以及连接中止时仅输出造成连接终止原因的 STATUS 代码。

示例：如果接收数据时发生临时通信错误，则先输出 STATUS 7003 (ERROR=false)，然后再输出 80C4 (ERROR=true)。

MB_RED_SERVER（作为 Modbus TCP 服务器通过 PROFINET 进行通信）

使用该指令可在 S7-1200 CPU 与支持 Modbus TCP 协议的设备之间建立连接。

表格 13-73　MB_RED_SERVER 指令

LAD/FBD	SCL	说明
%DB5 "MB_RED_ SERVER_DB" MB_RED_SERVER EN / ENO REG_KEY / LICENSED DISCONNECT / IDENT_CODE MB_HOLD_REG / DR_NDR_0A ERROR_0A STATUS_0A DR_NDR_1A ERROR_1A STATUS_1A DR_NDR_0B ERROR_0B STATUS_0B DR_NDR_1B ERROR_1B STATUS_1B RED_ERR_S7 RED_ERR_DEV TOT_COM_ERR	"MB_RED_SERVER_DB"(DISCONNECT:=_bool_in_, LICENSED=>_bool_out_ IDENT_CODE=>_string_out_ DR_NDR_0A=>_bool_out, ERROR_0A=>_bool_out, STATUS_0A=>_word_out_, DR_NDR_1A=>_bool_out, ERROR_1A=>_bool_out, STATUS_1A=>_word_out_, DR_NDR_0B=>_bool_out, ERROR_0B=>_bool_out, STATUS_0B=>_word_out_, DR_NDR_1B=>_bool_out, ERROR_1B=>_bool_out, STATUS_1B=>_word_out_, RED_ERR_S7=>_bool_out_, RED_ERR_DEV=>_bool_out_, TOT_COM_ERR=>_bool_out_, MB_HOLD_REG:=_variant_inout_);	MB_RED_SERVER 指令作为 Modbus TCP 服务器通过 PROFINET 连接进行通信。 MB_RED_SERVER 指令将处理 Modbus TCP 客户端的连接请求、接收并处理 Modbus 请求并发送响应。

表格 13-74 参数的数据类型

参数和类型		数据类型	说明
REG_KEY [1]	IN	STRING[17]	授权注册码 必须在每个 CPU 上分别对 MB_RED_SERVER 指令进行授权。
DISCONNECT	IN	Bool	使用 MB_RED_SERVER 指令建立与一个伙伴模块的被动连接。服务器会对连接描述中给出的指定或未指定 IP 地址的连接请求作出响应。 接受一个连接请求后，可以使用该参数进行控制： ● 0：CPU 在无通信连接时建立被动连接。 ● 1：终止连接初始化。如果设置了输入，则 CPU 不会处理其它客户端请求，并且启动终止连接。成功终止连接后，STATUS_x 参数将输出值 0003。
MB_HOLD_REG [2]	IN_OUT	Variant	指向 MB_RED_SERVER 指令中 Modbus 保持性寄存器的指针 MB_HOLD_REG 引用的存储区必须大于两个字节。 保持性寄存器中包含 Modbus 客户端通过 Modbus 功能 3（读取）、6（写入）、16（多次写入）和 23（在一个作业中读写）可访问的值。
LICENSED [1]	OUT	Bool	● 0：指令未获授权 ● 1：指令已获授权
IDENT_CODE [1]	OUT	STRING[18]	授权标识。使用此字符串申请 REG_KEY 注册码。.
DR_NDR_0A	OUT	Bool	“Data Read”或“New Data Ready”至连接 0A： ● 0：无新数据 ● 1：Modbus 客户端读取或写入的新数据
ERROR_0A	OUT	Bool	如果在调用 MB_RED_SERVER 指令到连接 0A 的过程中出错，则将 ERROR_0A 参数的输出设置为“1”。有关错误原因的详细信息，将由 STATUS_0A 参数指定。
STATUS_0A [3]	OUT	Word	连接 0A 上指令的详细状态信息。
DR_NDR_1A	OUT	Bool	“Data Read”或“New Data Ready”至连接 1A： ● 0：无新数据 ● 1：Modbus 客户端读取或写入的新数据
ERROR_1A	OUT	Bool	如果在调用 MB_RED_SERVER 指令到连接 1A 的过程中出错，则将 ERROR_1A 参数的输出设置为“1”。有关错误原因的详细信息，将由 STATUS_1A 参数指定。
STATUS_1A [3]	OUT	Word	连接 1A 上指令的详细状态信息。

参数和类型		数据类型	说明
DR_NDR_0B	OUT	Bool	“Data Read”或“New Data Ready”至连接 0B： • 0：无新数据 • 1：Modbus 客户端读取或写入的新数据
ERROR_0B	OUT	Bool	如果在调用 MB_RED_SERVER 指令到连接 0B 的过程中出错，则将 ERROR_0B 参数的输出设置为“1”。有关错误原因的详细信息，将由 STATUS_0B 参数指定。
STATUS_0B [3]	OUT	Word	连接 0B 上指令的详细状态信息。
DR_NDR_1B	OUT	Bool	“Data Read”或“New Data Ready”至连接 1B： • 0：无新数据 • 1：Modbus 客户端读取或写入的新数据
ERROR_1B	OUT	Bool	如果在调用 MB_RED_SERVER 指令到连接 1B 的过程中出错，则将 ERROR_1B 参数的输出设置为“1”。有关错误原因的详细信息，将由 STATUS_1B 参数指定。
STATUS_1B [3]	OUT	Word	连接 1B 上指令的详细状态信息。
RED_ERR_S7 [3]	OUT	Bool	• 0：SIMATIC 中无冗余错误 • 1：SIMATIC 中存在冗余错误
RED_ERR_S7 [3]	OUT	Bool	• 0：链接伙伴侧无冗余错误 • 1：链接伙伴侧存在冗余错误
RED_ERR_S7 [3]	OUT	Bool	• 0：至少已建立 1 个组态连接 • 1：完全丢失通信，所有组态连接均终止

[1] 有关详细信息，参见下文中“授权”部分。

[2] 有关详细信息，参见下文中“MB_HOLD_REG 输入参数”部分。

[3] 有关详细信息，参见下文中“输出参数：ERROR_x、RED_ERR_S7、RED_ERR_DEV 和 TOT_COM_ERR”部分。

说明

安全信息

请注意，网络的每个客户端都会被授予对过程映像输入和输出以及数据块或 Modbus 保持性寄存器定义的位存储区的读写访问权。同时还提供有选项，用于限制 IP 地址访问，以防未授权读写操作。但请注意，共享地址也可能会受到未授权访问。

说明

CPU 固件版本要求

手册本部分所述的 Modbus TCP 指令要求固件版本为 V4.2 或更高版本。
使用该指令时，无需其它任何硬件模块。

多个服务器连接

CPU 可以用于：

- 处理多个服务器连接
- 在同一个服务器端口同时接受多个来自不同的客户端的多个连接

连接的最大数目取决于所使用的 CPU，具体请参见相关 CPU 的技术规范。一个 CPU 的总连接数，包括 Modbus TCP 客户端和服务器的连接数，不能超过所支持的最大连接数。

连接服务器时，请记住以下规则：

- 每个 MB_RED_SERVER 连接都必须使用唯一的背景数据块。
- 要连接到服务器端口的每个客户端都需要一个唯一的连接/连接 ID。
- 连接 ID 在整个 CPU 范围内必须唯一。

将 Modbus 地址映射到过程映像

MB_RED_SERVER 指令允许到达的 Modbus 功能（1、2、4、5 和 15）直接读取和写入访问 CPU 的过程映像输入和输出（使用数据类型 BOOL 和 WORD）。

对于 S7-1200-CPU，输入和输出过程映像的地址空间为 1 KB。

下表列出了上述 Modbus 功能的地址空间：

Modbus 功能					
功能代码	功能	数据区	地址空间		
01	读取：位	Output	0	到	65.535
02	读取：位	Input	0	到	65.535
04	读取：WORD	Input	0	到	65.535
05	写入：位	Output	0	到	65.535
15	写入：位	Output	0	到	65.535

通过功能代码 3、6、16 和 23 将到达的 Modbus 请求写入 Modbus 保持性寄存器或从寄存器中读取（可通过 MB_HOLD_REG 参数或 Data_Area_Array 指定保持性寄存器）。

Modbus 功能

下表列出了 MB_RED_SERVER 指令支持的所有 Modbus 功能：

功能代码	说明
01	读取输出位
02	读取输入位
03	读取保持性寄存器
04	读取输入字
05	写入输出位
06	写入保持性寄存器
08	诊断功能： ● 回送测试（子功能 0x0000）：MB_RED_SERVER 指令接收数据字并按原样返回 Modbus 客户端。 ● 复位事件计数器（子功能 0x000A）：使用指令 MB_RED_SERVER，可复位以下事件计数器：“Success_Count”、“Xmt_Rcv_Count”、“Exception_Count”、“Server_Message_Count”和“Request_Count”。
11	诊断功能：获取通信的事件计数器 MB_RED_SERVER 指令使用一个通信的内部事件计数器，记录发送到 Modbus 服务器上成功执行的读写请求数。 执行功能 8 或 11 时，事件计数器不会递增。这种情况同样适用于会导致通信错误的请求。例如，发生协议错误（如，不支持所接收 Modbus 请求中的功能代码）。
15	写入多个输出位
16	写入保持性寄存器
23	通过请求写入和读取保持性寄存器

运行和冗余

通信节点可设计为独立式或者冗余式。如果其中一个伙伴采用独立式设计，则将其称为单侧冗余。如果两个伙伴均为冗余式设计，可将其称为双侧冗余。

- 单侧冗余：
  - 必须为通信伙伴之间的每个连接组态一个连接。**SIMATIC S7** 的连接点称为 **0** 和 **1**；**通信伙伴**的连接点称为 **A** 和 **B**。
    R-CPU 或 H-CPU 1 是指连接点 0，R-CPU 或 H-CPU 2 为连接点 1。
  - 组态：如果 S7 采用冗余设计，则为 S7 连接点 0 到链接伙伴的节点 A 创建一个连接（从 S7 连接点 **0** 到伙伴/节点 **A** 的连接 **=>** 连接 **0A**），一个从 S7 连接点 1 到链接伙伴的节点 A 的连接（从 S7 连接点 **1** 到伙伴/节点 **A** 的连接 **=>** 连接 **1A**）。下图给出了连接名称：

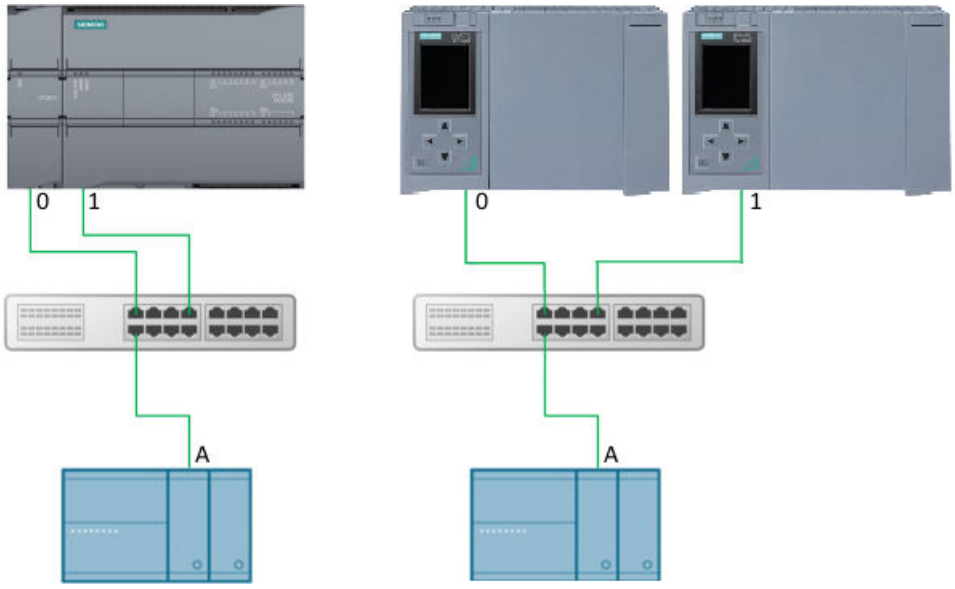


图 13-12　单侧冗余 S7

– 如果 S7 采用独立式设计，链接伙伴采用冗余设计，则为 S7 连接点 0 到链接伙伴的节点 A 创建一个连接（从 S7 连接点 **0** 到伙伴/节点 **A** 的连接 => 连接 **0A**），一个从 S7 连接点 0 到链接伙伴的节点 B 的连接（从 S7 连接点 **0** 到伙伴/节点 **B** 的连接 => 连接 **0B**）。下图给出了连接名称：

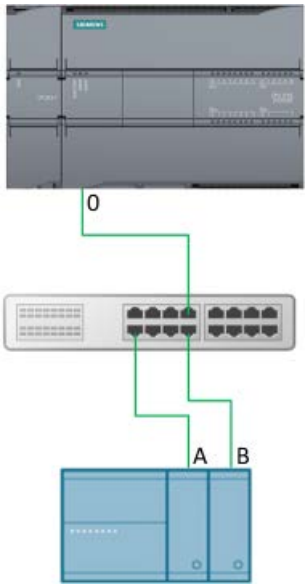


图 13-13　单侧冗余伙伴

- 双侧冗余：
  – 说明：必须为通信伙伴之间的每个连接组态一个连接。SIMATIC S7 的连接点称为 **0** 和 **1**；**通信伙伴**的连接点称为 **A** 和 **B**。
    R-CPU 或 H-CPU 1 是指连接点 0，R-CPU 或 H-CPU 2 为连接点 1。

– 组态：采用双侧冗余时，分别从 S7 连接点 0 创建两个连接（从 S7 连接点 **0** 到伙伴/节点 **A** 的连接 => 连接 **0A**，从 S7 连接点 **0** 到伙伴/节点 **B** 的连接 => 连接 **0B**），从连接点 1 创建两个连接到链接伙伴的节点 A 和 B（从 S7 连接点 **1** 到伙伴/节点 **A** 的连接 => 连接 **1A**，从 S7 连接点 **1** 到伙伴/节点 **B** 的连接 => 连接 **1B**）。下图给出了连接名称：

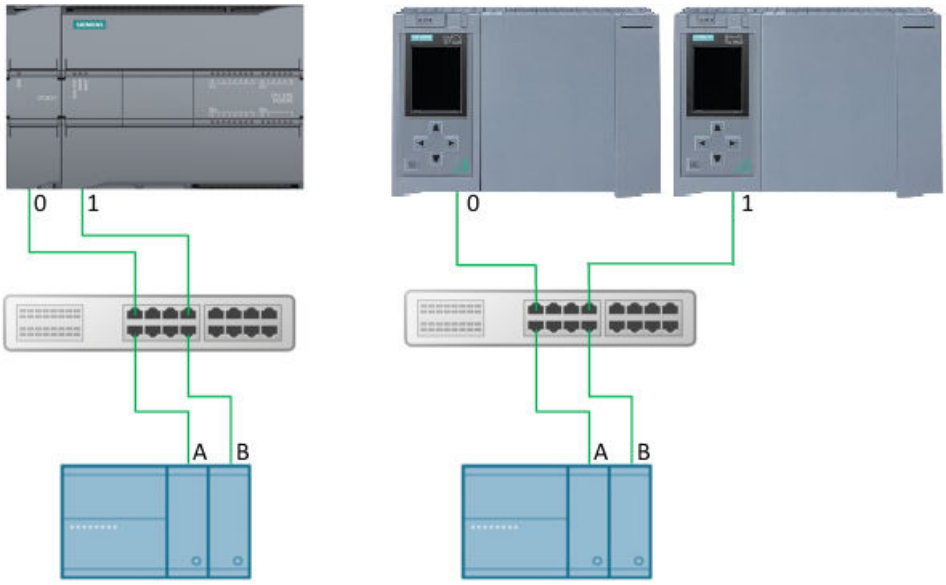


图 13-14　双侧冗余

- 帧处理：可通过所有组态连接接收帧。客户端可通过一个连接或通过所有连接发送帧。如果某个连接上接收到帧，则 CPU 将在相应的输出 DR_NDR_x 或 ERROR_x 中显示状态。每个连接独立运行，且对其它连接的显示没有影响。
- 冗余输出 RED_ERR_S7、RED_ERR_DEV 和 TOT_COM_ERR：
  - 冗余位 RED_ERR_S7、RED_ERR_DEV 和 TOT_COM_ERR 会根据状态输出的状态置位：

Number of faulty connections	STATUS_0A	STATUS_0B	STATUS_1A	STATUS_1B	RED_ERR_S7	RED_ERR_DEV	TOT_COM_ERR
0	okay	okay	okay	okay	FALSE	FALSE	FALSE
1	okay	okay	okay	Error	FALSE	FALSE	FALSE
	okay	okay	Error	okay	FALSE	FALSE	FALSE
	okay	Error	okay	okay	FALSE	FALSE	FALSE
	Error	okay	okay	okay	FALSE	FALSE	FALSE
2	okay	okay	Error	Error	TRUE	FALSE	FALSE
	okay	Error	okay	Error	FALSE	TRUE	FALSE
	Error	okay	okay	Error	FALSE	FALSE	FALSE
	okay	Error	Error	okay	FALSE	FALSE	FALSE
	Error	okay	Error	okay	FALSE	TRUE	FALSE
	Error	Error	okay	okay	TRUE	FALSE	FALSE
3	Error	Error	Error	okay	TRUE	TRUE	FALSE
	Error	Error	okay	Error	TRUE	TRUE	FALSE
	Error	okay	Error	Error	TRUE	TRUE	FALSE
	okay	Error	Error	Error	TRUE	TRUE	FALSE
4	Error	Error	Error	Error	TRUE	TRUE	TRUE

图 13-15　两侧冗余设置的中断位显示

Number of faulty connections	STATUS_0A	STATUS_0B	STATUS_1A	STATUS_1B	RED_ERR_S7	RED_ERR_DEV	TOT_COM_ERR
0	okay	0AFF	okay	0AFF	FALSE	FALSE	FALSE
1	okay	0AFF	Error	0AFF	TRUE	TRUE	FALSE
	Error	0AFF	okay	0AFF	TRUE	TRUE	FALSE
2	Error	0AFF	Error	0AFF	TRUE	TRUE	TRUE

图 13-16　一侧冗余设置的中断位显示

说明

客户端和服务器的端口号

Modbus 客户端使用的端口号从 2000 开始。通常通过端口号 502 对 Modbus 服务器进行寻址。可以为多个连接组态端口 502（多端口），具体取决于 CPU。如果为本地端口 502 组态至少两个连接，则在未指定连接的情况下，将发出请求的客户端随机分配给现有的服务器连接。不会自动将想要连接到 "MB_RED_SERVER" 指令的第一个客户端分配给连接 0A。将客户端请求分配给服务器连接后，在连接终止前，分配在交换帧期间保持不变。

参数分配

对于 S7-1200，可以使用 MB_RED_SERVER 指令 **V1.0** 和 **V1.1**。CPU 通过 CPU 或 CM/CP 的本地接口建立连接。CPU 使用 TCON_IP_V4 结构组态和建立连接。

MB_RED_SERVER 的组态： 使用 MB_RED_SERVER 指令的组态对话框进行以下设置：

- 连接 0A、1A、0B 和 1B 的连接参数（有关冗余组态的更多信息，请参见上文的“运行和冗余”。）
- 内部参数（可选）

通过 MB_RED_SERVER 指令或通过工艺对象，可以打开组态对话框。

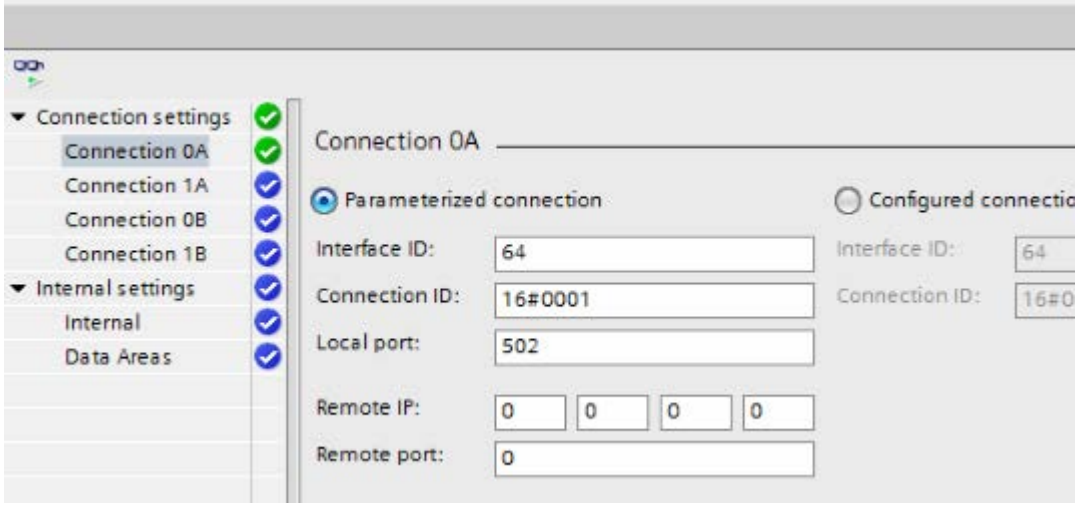


图 13-17　参数化的服务器连接

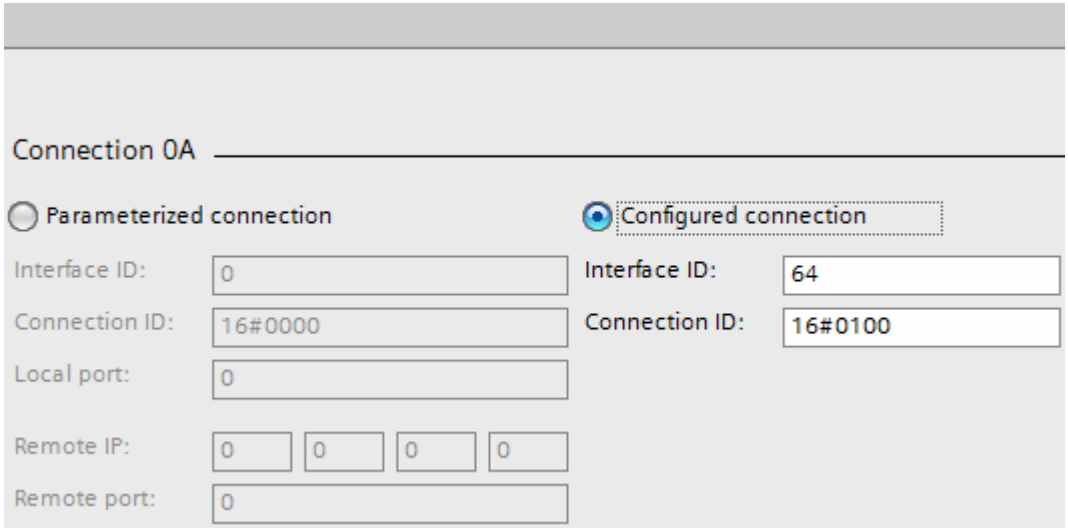


图 13-18　已组态的服务器连接

变量	起始值	说明
组态的连接		
Interface ID	64	所用 PN 接口的硬件标识符
Connection ID	16#0000	所用连接的连接 ID。这些连接 ID 在整个 CPU 范围内必须唯一。
Local port	502	服务器块的本地端口号。Modbus/TCP 服务器的默认端口为 502。
Remote IP	0.0.0.0	客户端的远程 IP 地址。默认情况下不为客户端输入 IP 地址。
Remote port	0	客户端的远程端口号。默认情况下不为客户端输入端口号。
组态的连接		
Interface ID	64	所用 PN 接口的硬件标识符
Connection ID	16#0000	所用连接的连接 ID。在网络视图中对这些连接进行组态。

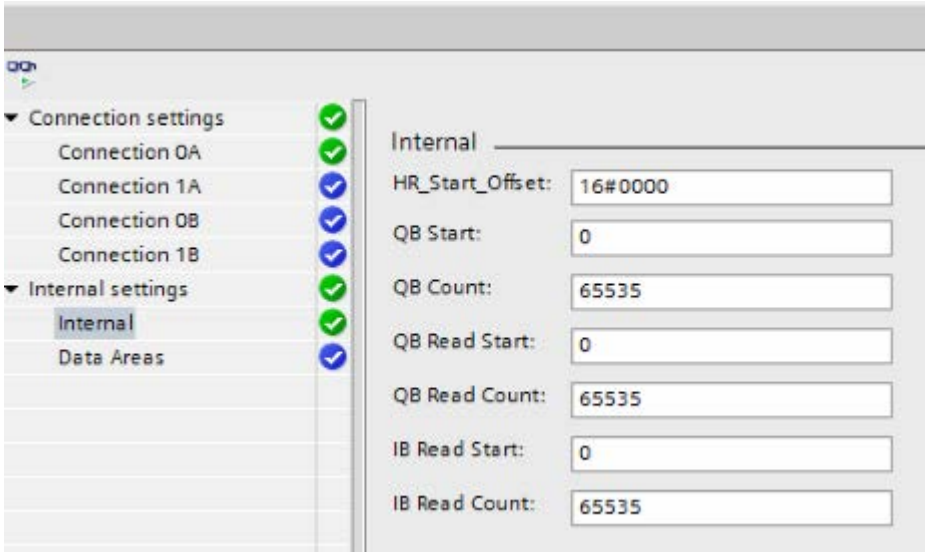


图 13-19　内部参数（可选）

变量	数据类型	起始值	说明
HR_Start_Offset	WORD	0	指定 Modbus 保持性寄存器的起始地址。
QB_Start	UINT	0	Modbus 主站可写入输出的寻址范围的起始地址（0 到 65535 个字节）
QB_Count	UINT	0	Modbus 主站可写入的输出字节数。 *示例：* • QB_Start=0 和 QB_Count=10：Modbus 主站可写入的输出字节数为 0 到 9。 • QB_Count=0：Modbus 主站无法写入任何输出字节。
QB_Read_Start	UINT	0	Modbus 主站可读取输出的寻址范围的起始地址（0 到 65535 个字节）
QB_Read_Count	UINT	0	Modbus 主站可读取的输出字节数。 *示例：* • QB_Read_Start=0 和 QB_Read_Count=10：Modbus 主站可读取的输出字节数为 0 to 9。 • QB_Read_Count=0：Modbus 主站无法读取任何输出字节。
IB_Read_Start	UINT	0	Modbus 主站可读取输入的寻址范围的起始地址（0 到 65535 个字节）

变量	数据类型	起始值	说明
IB_Read_Count	UINT	0	Modbus 主站可读取的输入字节数。 *示例：* IB_Read_Start=0 和 IB_Read_Count=10：Modbus 主站可读取的输入字节数为 0 to 9。 IB_Read_Count=0：Modbus 主站无法读取任何输入字节。
Data_Area_Array	ARRAY [1..8]		
data_type	UINT	0	数据类型：0 到 4
db	UINT	0	数据块编号
start	UINT	0	数据块中的第一个 Modbus 地址
length	UINT	0	数据块中 Modbus 值的数量

使用 HR_Start_Offset 静态变量寻址

Modbus 保持性寄存器的起始地址为 0。

*示例：*保持性寄存器的起始地址为 MW100，长度为 100 字。

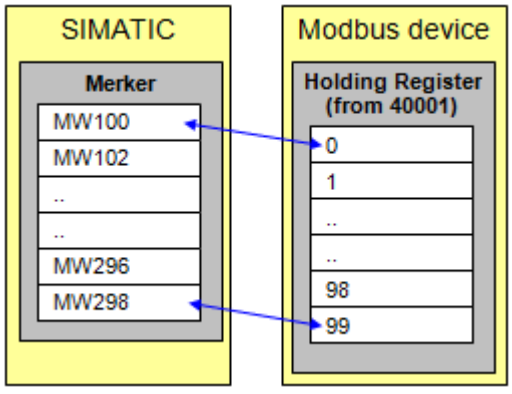


图 13-20　HR_Start_Offset_0

可通过定义 HR_Start_Offset 变量，将 Modbus 保持性寄存器的起始地址设置为不为 0。

*示例：*HR_Start_Offset 参数中偏移值为 20 表示，保持性寄存器的起始地址从 0 变为 20。在低于 20 或高于 119 的地址对保持性寄存器进行寻址时，将导致错误发生。

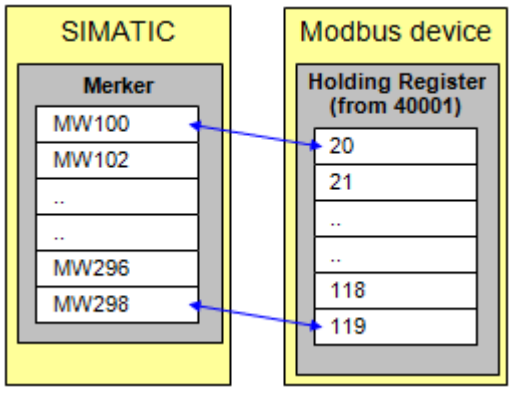


图 13-21　HR_Start_Offset_20

Data_Area_Array [1..8]

SIMATIC S7 存储器中有八个可以用于映射 MODBUS 地址的数据区。如果使用“保持性寄存器”数据类型定义数据区，则不对 MB_HOLD_REG 参数进行评估。而是根据作业类型，Modbus 主站将 Modbus 寄存器和位写入数据块或从数据块中读取。CPU 可在随后的程序执行中进一步处理这些值。

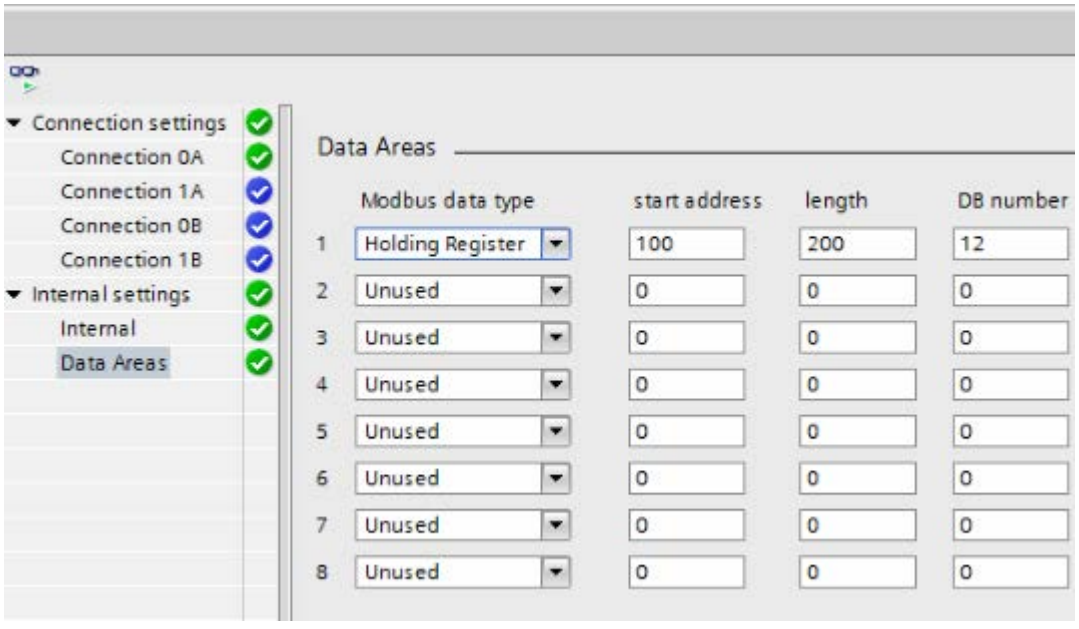


图 13-22　服务器数据区

任何作业都只能从一个数据块读取数据或向一个数据块写入数据。访问寄存器或位于多个数据块中的位值时，即使编号连续无间隔，也将分为两个作业。组态时请务必注意。一个数据块中可以映射的 Modbus 区（寄存器或位值）数目比 Modbus 主站通过一个帧可以处理的数目多。

data_type

data_type 参数指定 Modbus 主站在该数据块中映射的 MODBUS 数据类型。如果在 data_type 中输入值“0”，则 Modbus 主站不使用相应的数据区域。如果 Modbus 主站要使用多个 Data_Area，则必须依次定义这些区域。Modbus 主站将不会处理 data_type = 0 之后的任何条目。

标识符	数据类型	说明
0	未使用区域	
1	输出位（线圈）	位
2	输入位（输入）	位
3	保持性寄存器	字
4	输入字（输入寄存器）	字

db

db 参数指定映射 MODBUS 寄存器或下面定义的位值的数据块。数据块编号 0 为系统保留，CPU 不允许使用。

start, length

start 指定 Modbus 主站在数据块的数据字 0 中映射的第一个 Modbus 地址。length 参数定义了 Modbus 主站在数据块中映射的 MODBUS 地址的数量和长度。定义的数据区不得重叠。length 参数不得等于 0。

示例：使用 Data_Area_Array 映射地址。

数据区 1	data_type	3：保持性寄存器
	db	11
	起始	0
	长度	500
数据区 2	data_type	3：保持性寄存器
	db	12
	起始	720
	长度	181
数据区 3	data_type	4：输入字
	db	13
	起始	720
	长度	281

数据区 4	data_type	1：输出位
	db	14
	起始	640
	长度	611
数据区 5	data_type	2：输入位
	db	15
	起始	1700
	长度	601
数据区 6	data_type	1：输出位
	db	16
	起始	1700
	长度	601
数据区 7	data_type	未使用
	db	0
	起始	0
	长度	0
数据区 8	data_type	未使用
	db	0
	起始	0
	长度	0

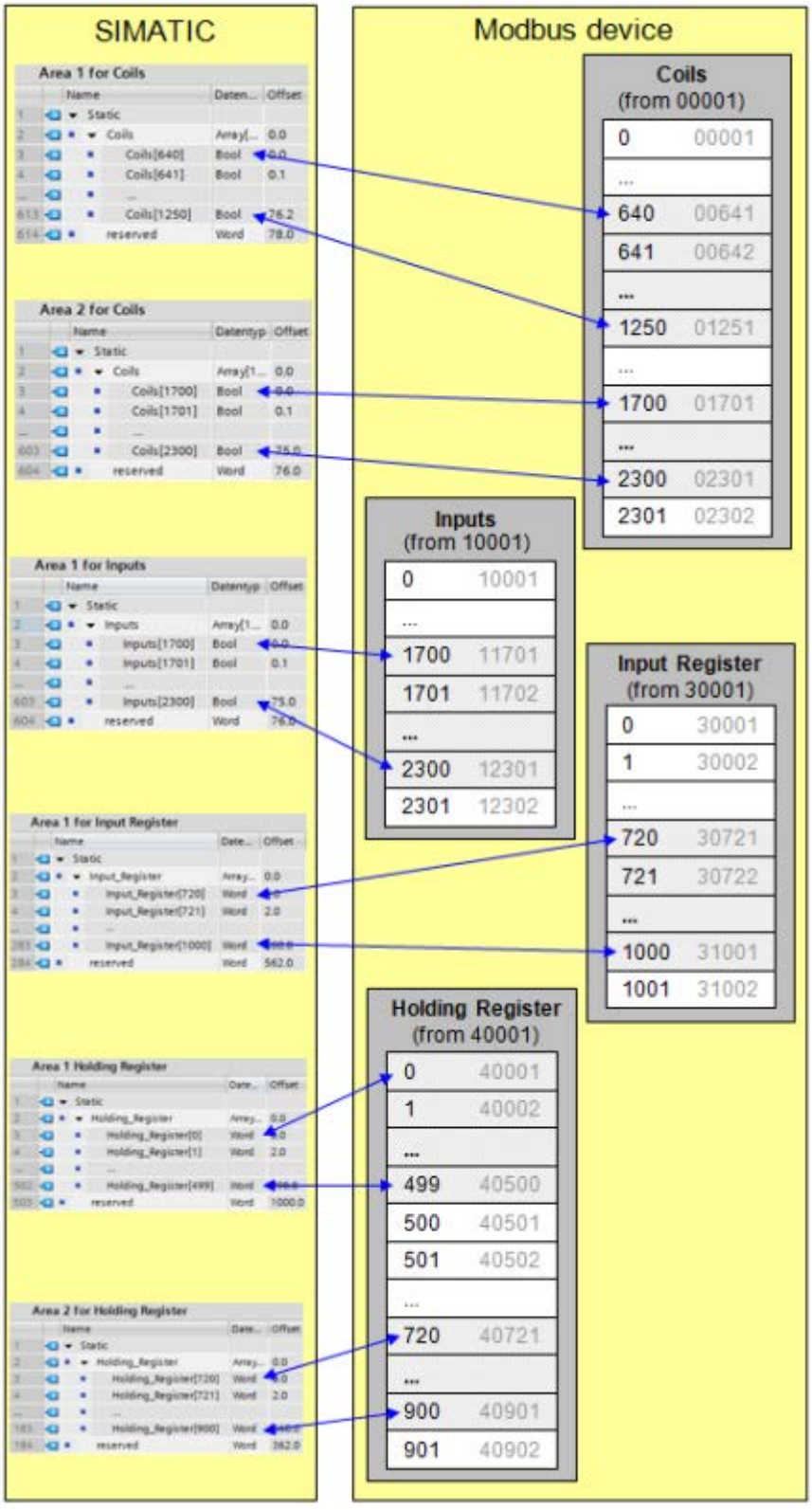

SIMATIC
Modbus device
Area 1 for Coils
Area 2 for Coils
Area 1 for Inputs
Area 1 for Input Register
Area 1 Holding Register
Area 2 for Holding Register
Coils
(from 00001)
0 00001
640 00641
641 00642
1250 01251
1700 01701
2300 02301
2301 02302
Inputs
(from 10001)
0 10001
1700 11701
1701 11702
2300 12301
2301 12302
Input Register
(from 30001)
0 30001
1 30002
720 30721
721 30722
1000 31001
1001 31002
Holding Register
(from 40001)
0 40001
1 40002
499 40500
500 40501
501 40502
720 40721
900 40901
901 40902

图 13-23　地址构建

授权

必须在每个 CPU 上分别对 MB_RED_SERVER 指令进行授权，且该指令需要付费。授权分为两个步骤：

- 显示许可证 IDENT_CODE
- 输入 REG_KEY 注册密钥：必须在每个 MB_RED_SERVER 指令中分配 REG_KEY 注册密钥。将 REG_KEY 保存在共享数据块中，所有 MB_RED_SERVER 指令均可通过该数据块接收所需的注册密钥。

显示许可证 IDENT_CODE 的程序：

1. 根据循环 OB 中的要求，为 MB_RED_SERVER 指令分配参数。将程序下载到 CPU，并将 CPU 设置为 RUN。
2. 打开 Modbus 指令的背景数据块，然后单击 “监视全部”(Monitor all) 按钮。
3. 背景数据块将在 IDENT_CODE 输出中显示一个 18 位的字符串。

		Name	Data type	Start value	Monitor value
1		Input			
2		REG_KEY	String[17]	''	''
3		USE_ALL_CONN	Bool	false	FALSE
4		REQ	Bool	false	FALSE
5		DISCONNECT	Bool	false	FALSE
6		MB_MODE	USInt	0	0
7		MB_DATA_ADDR	UDInt	0	0
8		MB_DATA_LEN	UInt	0	0
9		Output			
10		LICENSED	Bool	false	FALSE
11		IDENT_CODE	String[18]	''	'RTPCFIGDCDIIHJHAH4'
12		DONE	Bool	false	FALSE
13		BUSY	Bool	false	FALSE
14		ERROR	Bool	false	FALSE

图 13-24　许可证

4. 使用复制/粘帖功能，从数据块中复制该字符串，并粘贴到表格中（在订购产品后以电子邮件形式发送给客户的表格，或包含在 CD 光盘中）。
5. 使用服务请求将表格发送至客户支持 (https://support.industry.siemens.com/my/ww/en/requests/#createRequest)。随后用户将收到 CPU 的注册密钥。

输入注册密钥 REG_KEY 的程序：

1. 通过“添加新块...”(Add new block…) 插入具有唯一符号名的全新共享数据块，例如“License_DB”。
2. 在该块中创建数据类型为 STRING[17] 的 REG_KEY 参数。

LICENSE_DB

		Name	Data type	Offset	Start value
1		Static			
2		REG_KEY	String[17]	0.0	''

图 13-25　REG_KEY

3. 使用复制/粘帖将已发送的 17 位注册密钥复制到“起始值”(Start value) 列。
4. 在循环 OB 中，在 MB_RED_SERVER 指令的 REG_KEY 参数中，输入许可证数据块的名称和字符串名称（例如 License_DB.REG_KEY）。
5. 将修改后的块下载到 CPU。可以在运行时间输入注册密钥；不需要从 STOP 更改 RUN。
6. 现在，该 CPU 已获得使用 MB_RED_SERVER 指令进行 Modbus/TCP 通信的授权；LICENSED 输出位为 TRUE。

授权缺失或不正确的纠正程序：

- 如果输入的注册密钥错误或未输入注册密钥，则 CPU 的 ERROR LED 指示灯闪烁。此外，S7-1200 CPU 会在诊断缓冲区中生成有关许可证缺失的循环条目。

诊断缓冲区

事件

以PG/PC本地时间显示CPU事件时间戳

编号	日期和时间	事件
1	2018/12/20 12:35:15.782	FB 1086 中的区域长度错误 - 继续处理（无 OB 处理）
2	2018/12/20 12:35:15.781	该功能包无有效的许可密钥
3	2018/12/20 12:35:09.776	FB 1086 中的区域长度错误 - 继续处理（无 OB 处理）
4	2018/12/20 12:35:09.775	该功能包无有效的许可密钥
5	2018/12/20 12:35:03.770	FB 1086 中的区域长度错误 - 继续处理（无 OB 处理）
6	2018/12/20 12:35:03.769	该功能包无有效的许可密钥
7	2018/12/20 12:34:57.763	FB 1086 中的区域长度错误 - 继续处理（无 OB 处理）
8	2018/12/20 12:34:57.763	该功能包无有效的许可密钥
9	2018/12/20 12:34:51.756	FB 1086 中的区域长度错误 - 继续处理（无 OB 处理）

冻结显示

图 13-26　Diagnostic_buffer

- 如果注册密钥缺失或不正确，CPU 仍将处理 Modbus/TCP 通信，但始终会在 STATUS_x 输出中显示“W#16#0A90”（无适用于功能性包装的有效许可证密钥）。输出位 LICENSED 为 FALSE。

MB_HOLD_REG 输入参数

MB_HOLD_REG 参数是一个指向数据缓冲区的指针，该缓冲区用于存储 Modbus 客户端对其具有读写访问权限的数据。可使用全局数据块 (D) 或位存储器 (M) 作为存储区：

- 数据块 (D) 中地址的最大数量取决于 CPU 的最大数据块大小。
- 位存储器 (M) 中地址的最大数量取决于 CPU 的最大位存储区。

下图说明了针对 Modbus 功能 3（读取多个 WORD）、6（写入一个 WORD）、16（写入多个 WORD）和 23（读写多个 WORD）将 Modbus 地址映射到保持性寄存器的情况。

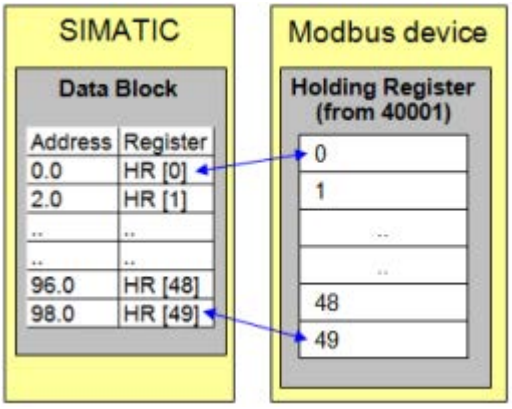


图 13-27　MB_HOLD_REG：偏移量为 0 的数据块

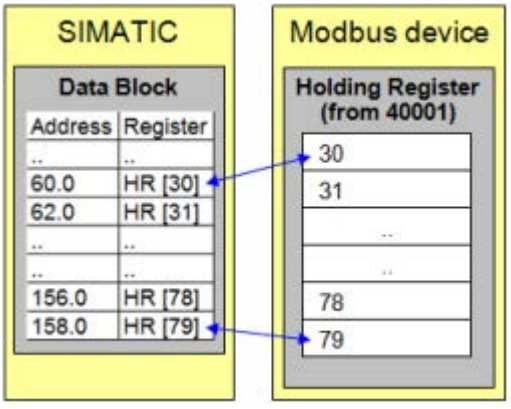


图 13-28　MB_HOLD_REG：偏移量为 60 的数据块

Data_Area_Array [1..8]：要使用可选参数 Data_Area_Array [1..8]， 请参见上文“参数分配”部分。

输出参数 ERROR_x、STATUS_x、RED_ERR_S7、RED_ERR_DEV 和 TOT_COM_ERR

CPU 将错误消息显示在 MB_RED_SERVER 指令的状态输出中：

说明

在程序编辑器中，错误状态代码可显示为整数或十六进制值：

1. 在程序编辑器中打开所需的块。
2. 单击“启用/禁用监视”(Monitor on/off) 切换程序状态的显示格式。（如果尚未建立在线连接，则打开“转至在线”(Go online) 对话框。在此对话框中，可以建立在线连接。）
3. 选择待监视的变量，然后在快捷菜单的“显示格式”(Display format) 中，选择所需的显示格式。

STATUS_x 参数（常规状态信息）

STATUS (W#16#)	说明
0000	指令已执行，且无任何错误。
0001	连接已建立。
0003	连接已终止。
0A90	MB_RED_SERVER 指令未授权。有关详细信息，参见上文中“授权”部分。
0AFF	连接未组态且未使用。必须组态 0A 连接。
7000	未激活任何调用且未建立任何连接 (REQ=0, DISCONNECT=1)。
7001	首次调用。已触发连接建立操作。
7002	中间调用。正在建立连接。
7003	正在终止连接。
7005	正在发送数据。
7006	正在接收数据。

STATUS_x 参数（协议错误）

STATUS (W#16#)	来自 MB_RED_SERVER 的错误消息中的错误代码 (B#16#)	说明
8380	-	接收到的 Modbus 帧格式错误或字节数过少。
8381	01	功能代码不受支持。

STATUS (W#16#)	来自 MB_RED_SERVER 的错误消息中的错误代码 (B#16#)	说明
8382	03	数据长度错误： • 接收到的 Modbus 帧长度无效 • Modbus 帧头中输入的帧长度与实际接收到的字节数不一致。 • Modbus 帧头中输入的字节数与实际接收到的字节数不一致（功能 15 和 16）。
8383	02	数据地址错误或访问了保持性寄存器（MB_HOLD_REG 参数）地址以外的区域。有关详细信息，请参见上文“MB_HOLD_REG”部分。
8384	03	数据值错误（功能 05）。
8385	03	诊断代码不受支持（仅限功能 08）。

STATUS_x 参数（参数错误）

STATUS (W#16#)	说明
80BB	参数 ActiveEstablished 的值无效 只允许对服务器建立被动连接 (active_established = FALSE)。
8187	参数 MB_HOLD_REG 的指针无效。数据区过小。
8389	数据区域定义无效： • data_type 值无效 • 数据块编号无效或不存在： – db 值无效 – 数据块编号不存在 – 数据块编号已由另一个数据区域使用 – 数据块具有优化访问权限 – 数据块不在工作存储器中 • length 值无效 • 属于同一 MODBUS 数据类型的 MODBUS 地址范围重叠

说明

内部使用的通信指令的错误代码

对于 MB_RED_SERVER 指令，除了会发生表中列出的错误外，还会发生由该指令所使用的通信指令（“TCON”、“TDISCON”、“TSEND”、“TRCV”、“T_DIAG”和“T_RESET”）所引起的错误。

错误代码通过 MB_RED_SERVER 指令的背景数据块来指定。相应指令的错误代码会显示在各个实例 "Static" 部分中的 STATUS 下。

有关错误代码的含义，请参见相应通信指令文档。

说明

发送或接收数据时发生通信错误

如果发送或接收数据时发生通信错误（80C4（临时通信错误。指定的连接临时终止。）、80C5（远程伙伴主动终止连接。）、80A1（指定的连接中断或尚未建立。）），CPU 现有连接会终止。

即，可查看连接终止时所有返回的 STATUS 值，以及连接中止时仅输出造成连接终止原因的 STATUS 代码。

示例：如果接收数据时发生临时通信错误，则先输出 STATUS 7003 (ERROR=false)，然后再输出 80C4 (ERROR=true)。

13.5.2.4 Modbus TCP 示例

示例： MB_SERVE 多 TCP 连接

可以拥有多个 Modbus TCP 服务器连接。为此，必须为每个连接单独执行 MB_SERVER。每个连接必须使用单独的背景数据块、连接 ID 和 IP 端口。S7-1200 仅允许每个 IP 端口进行一个连接。

为了达到最佳性能，应在每个程序周期为各个连接执行 MB_SERVER。

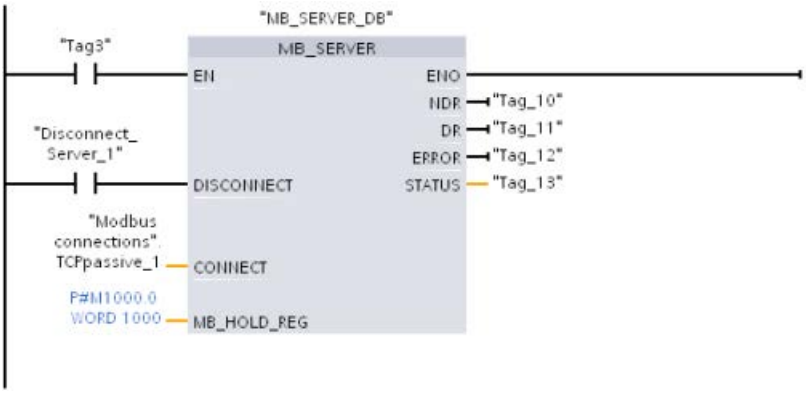
CONNECT 参数使用系统数据类型 TCON_IP_V4。这些数据结构的相关示例位于名为“Modbus 连接”的数据块中。“Modbus 连接”数据块包含两个 TCON_IP_V4 结构：“TCPpassive_1”（针对连接 1）和“TCP_passive_2”（针对连接 2）。程序段注释中描述的连接属性 ID 和 LocalPort 为存储在 CONNECT 数据结构中的数据元素。

TCON_IP_V4 CONNECT 数据同时包含 RemoteAddress ADDR 数组中的 IP 地址。TCPpassive_1 和 TCP_passive_2 内的 IP 地址分配对建立 TCP 服务器连接没有影响，但是会决定哪些 Modbus TCP 客户端可通过与各 MB_SERVER 连接进行通信。MB_SERVER 被动侦听 modbus 客户端消息，并将进入消息的 IP 地址与存储在相应 RemoteAddress ADDR 数组中的 IP 地址进行比较。

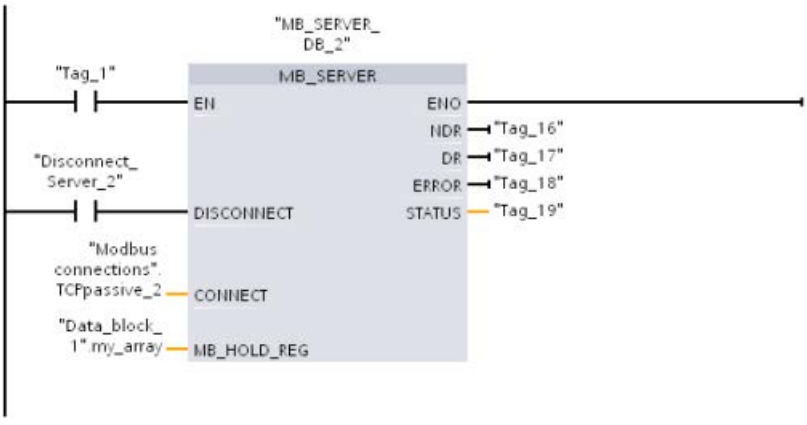
两个 MB_SERVER 指令可使用以下三种 MB_SERVER IP 地址变量：

- **IP 地址 = 0.0.0.0**
  各 MB_SERVER 将响应使用任意 IP 地址的所有 Modbus TCP 客户端。
- **IP 地址 = TCPpassive_1 和 TCPpassive_2 中的 IP 地址相同**
  两个 MB_SERVER 连接仅响应来自该 IP 地址的 Modbus 客户端。
- **IP 地址 = TCP_passive_1 和 TCP_passive_2 中的 IP 号不同**
  各 MB_SERVER 仅响应来自其 TCON_IP_V4 数据中存储的 IP 地址的 Modbus 客户端。

程序段 1：连接 #1，背景 DB =“MB_SERVER_DB”、“Modbus connections.TCPpassive_1”内（ID = 1 且 LocalPort = 502）



程序段 2：连接 #2，背景 DB =“MB_SERVER_DB_1”、“Modbus connections.TCPpassive_2”内（ID = 2 且 LocalPort = 503）



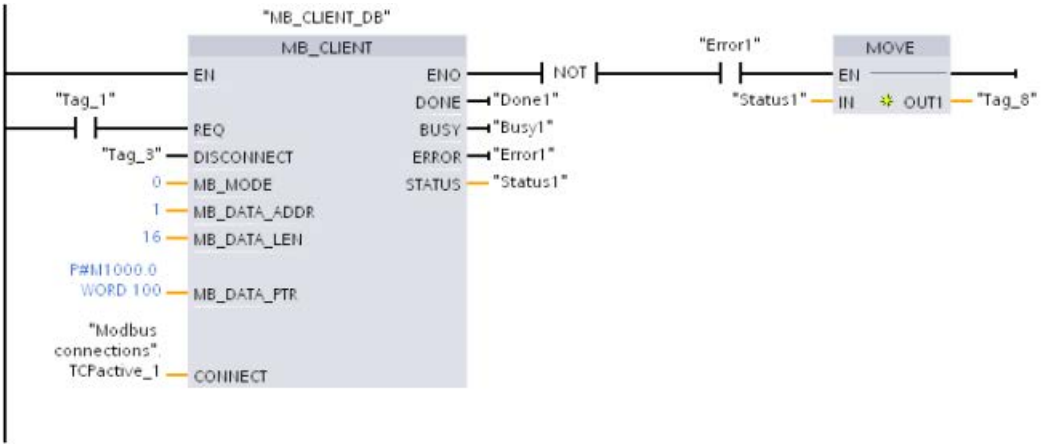
示例： MB_CLIENT 1： 通过公共 TCP 连接发送多个请求

多个 Modbus 客户端请求可通过同一连接发送。为此，必须使用相同的背景数据块、连接 ID 和端口号。

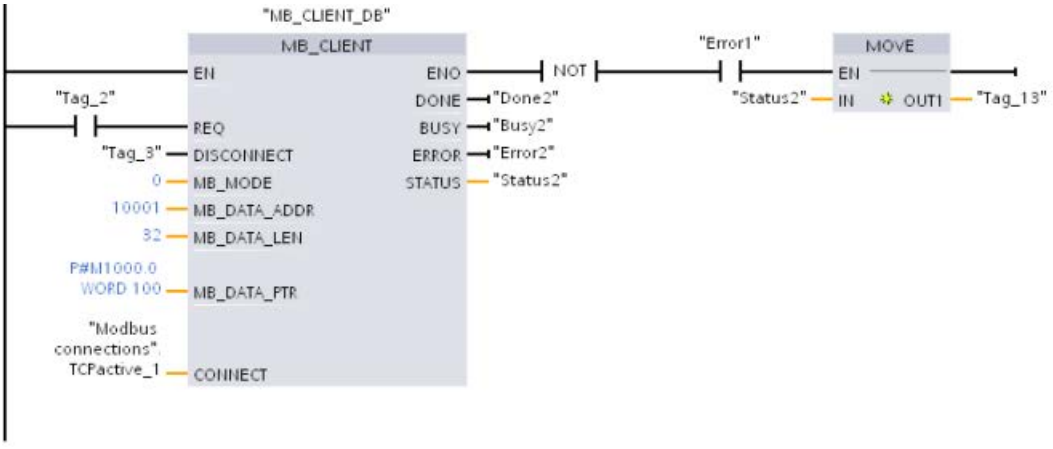
因为两个 MB_CLIENT 框使用相同的 CONNECT 参数 TCON_IP_v4 数据结构 ("Modbus_connections".TCPactive_1)，因此连接 ID、端口号和 IP 地址均相同。CONNECT IP 地址数据分配目标 Modbus TCP 服务器的 IP 地址。

在任意给定时间内，只能有一个 MB_CLIENT 处于激活状态。一个客户端完成执行后，下一个客户端才能开始执行。由程序逻辑负责执行顺序逻辑。本示例所示为两个客户端从单个 Modbus 客户端读取远程数据并将数据传送至 Modbus 客户端 CPU（从 M1000.0 起始的 M 存储器）。并捕获返回的错误（可选）。

程序段 1：Modbus 功能 1 - 从使用“Modbus 连接”.TCPactive_1 中所分配的 IP 地址的 Modbus TCP 服务器中读取 16 位输出位。



程序段 2：Modbus 功能 2 - 从使用“Modbus 连接”.TCPactive_1 中所分配的 IP 地址的 Modbus TCP 服务器中读取 32 位输入位。



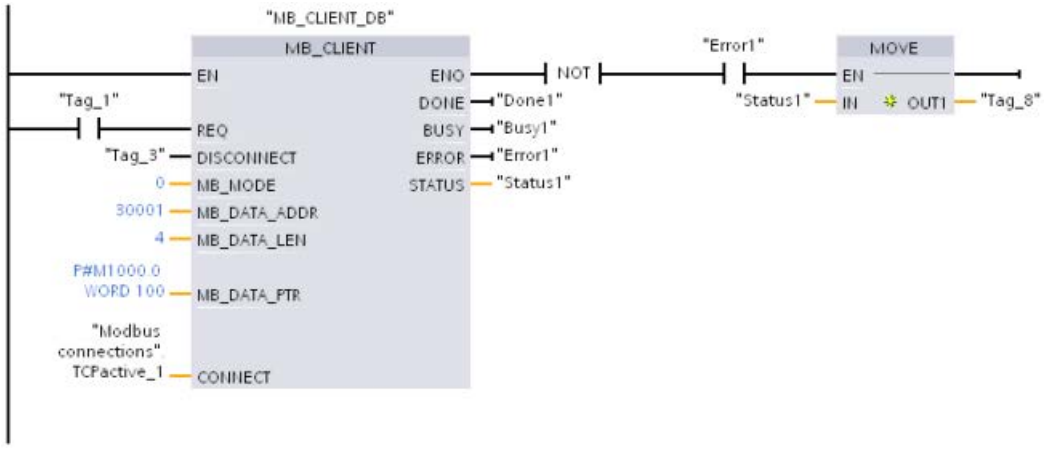
示例： MB_CLIENT 2： 通过不同的 TCP 连接发送多个请求

Modbus TCP 客户端的请求可通过各种不同连接来发送。 为此，必须使用不同的背景 DB 和连接 ID。

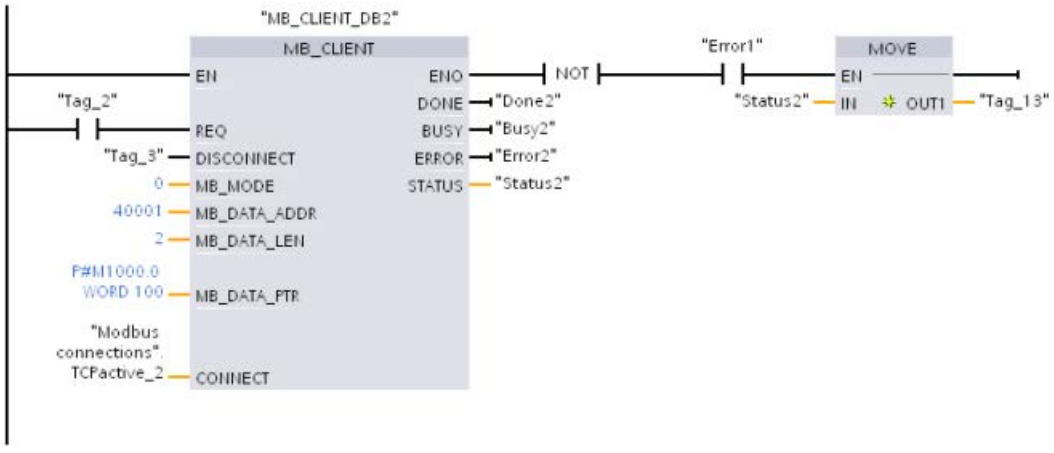
如要与同一 Modbus 服务器建立连接，则 RemotePort（IP 端口）号必须不同。 如果与不同服务器建立连接，则对 IP 端口号没有限制。

本示例所示为两个 Modbus TCP 户端将来自两个不同 Modbus TCP 服务器的远程数据传送至同一本地 CPU 存储区（起始地址为 M1000.0），并捕获返回的错误（可选）。

程序段 1： Modbus 功能 4 - 从 Modbus TCP 服务器读取输入过程映像字
CONNECT 参数 =“Modbus 连接”.TCPactive_1： 连接 **ID** = 1，**RemoteAddress** = 192.168.2.241，**RemotePort** = 502



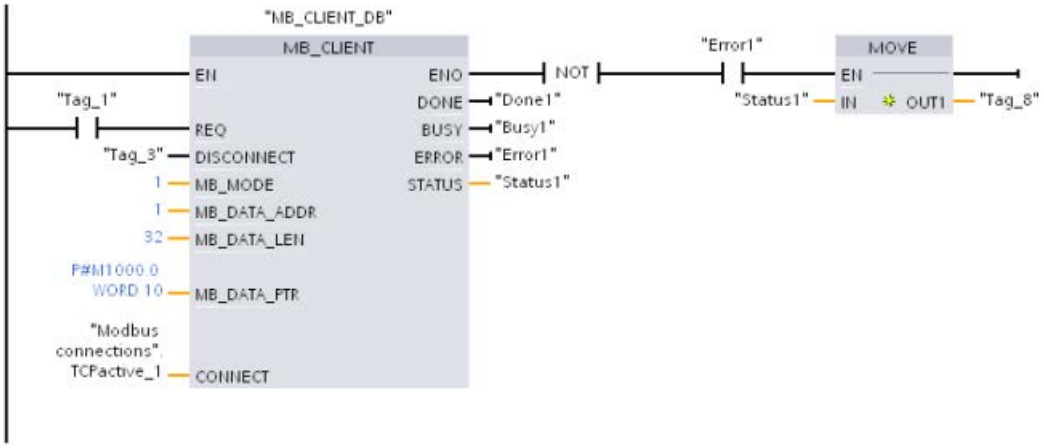
程序段 2： Modbus 功能 3 - 从 Modbus TCP 服务器读取保持寄存器字
CONNECT 参数 =“Modbus 连接”.TCPactive_2： 连接 **ID** = 2，**RemoteAddress** = 192.168.2.242，**RemotePort** = 502



示例： MB_CLIENT 3： 输出映像写入请求

本示例所示为 Modbus 客户端请求将位数据从本地 CPU 存储区（起始地址为 M1000.0）传送至远程 Modbus TCP 服务器。

程序段 1： Modbus 功能 15 - 在 Modbus 服务器中写入输出位

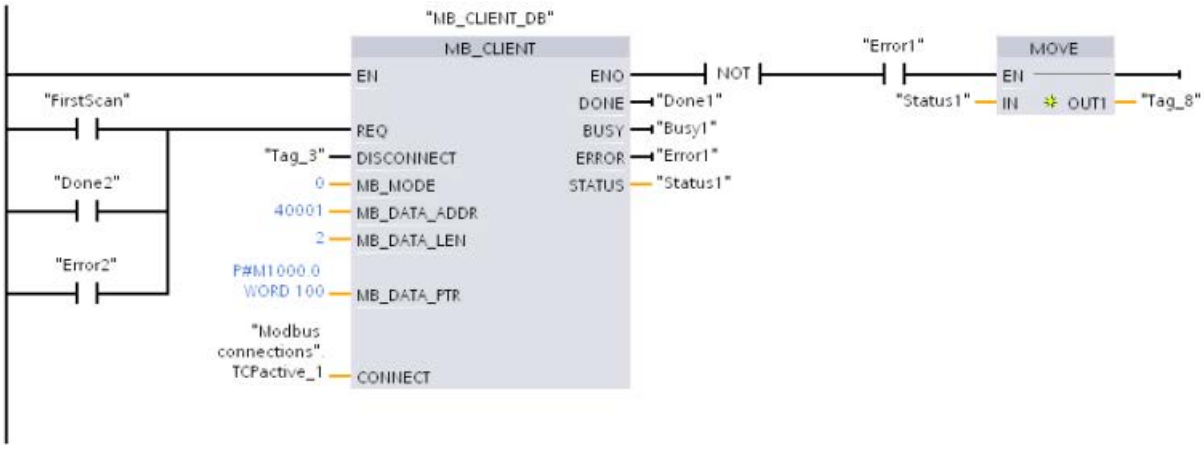


示例： MB_CLIENT 4： 协调多个请求

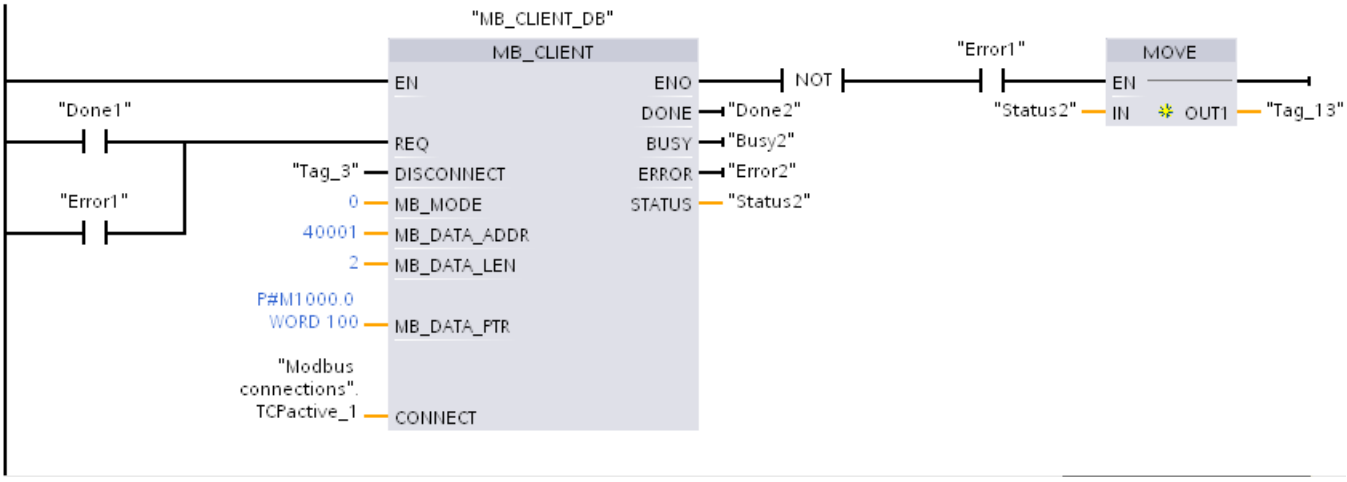
必须确保各个 Modbus TCP 请求都完成执行。 必须由程序逻辑来控制执行顺序。 下面的示例显示了首个和第二个客户端请求输出如何控制执行顺序。

该示例所示为两个客户端使用同一 CONNECT 连接数据（不同时使用）。 客户端将保持寄存器数据从同一远程 Modbus TCP 服务器传送至同一本地 CPU 存储区 M 地址。 此外，还捕获了返回的错误，这是可选的。

程序段 1： Modbus 功能 3 - 读取 Modbus TCP 服务器保持寄存器字



程序段 2： Modbus 功能 3 - 读取 Modbus TCP 服务器保持寄存器字



13.5.3 Modbus RTU

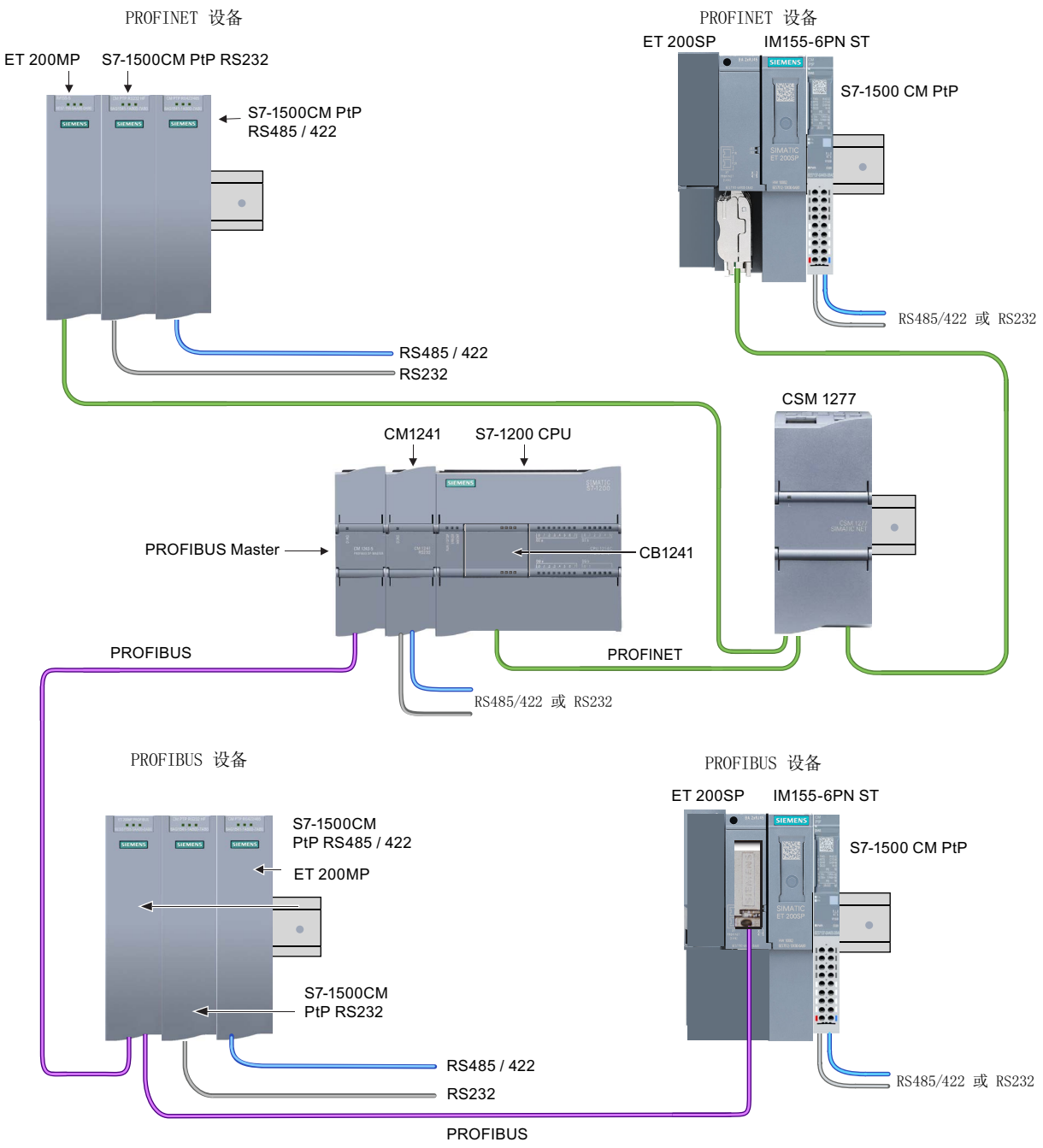
13.5.3.1 概述

对于 V4.1 版本及以上的 S7-1200 CPU 和 STEP 7 V13 SP1，该 CPU 扩展了 Modbus RTU 的功能，可以使用 PROFINET 或 PROFIBUS 分布式 I/O 机架与各类设备（RFID 阅读器、GPS 设备和其它）进行通信：

- PROFINET (页 810)：可以将 S7-1200 CPU 的以太网接口连接至 PROFINET 接口模块。可通过机架中 PtP 通信模块以接口模块实现与 PtP 设备的串行通信。
- PROFIBUS (页 986)：在 S7-1200 CPU 机架左边插入 PROFIBUS 通信模块。将 PROFIBUS 通信模块连接至 PROFIBUS 接口模块的机架。可通过机架中 PtP 通信模块以接口模块实现与 PtP 设备的串行通信。

出于这个原因，S7-1200 支持两组 PtP 指令：

- 早期 Modbus RTU 指令 (页 1344)：这些 Modbus RTU 指令存在于 S7-1200 的 V4.0 版本之前，并且仅可通过 CM 1241 通信模块或 CB 1241 通信板进行串行通信。
- Modbus RTU 指令 (页 1267)：这些 Modbus RTU 指令具备早期指令的所有功能，并且增添了连接 PROFINET 和 PROFIBUS 分布式 I/O 的功能。借助 Modbus RTU 指令，您可组态分布式 I/O 机架中 PtP 通信模块与 PtP 设备的之间通信。要使用这些 Modbus RTU 指令，S7-1200 CM 1241 模块的固件版本不得低于 V2.1。



说明

用于 S7-1200 的 V4.1 版本时，可以对所有类型的点对点通信使用点对点指令：串行通信、基于 PROFINET 的串行通信和基于 PROFIBUS 的串行通信。STEP 7 提供早期点对点指令的目的仅是为了支持现有程序。无论对于 V4.1 CPU 或 V4.0 还是更早版本的 CPU，早期指令仍然有效。无须对之前程序的指令进行转换。

13.5.3.2 选择 Modbus RTU 指令的版本

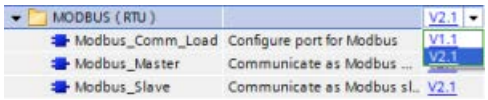
在 STEP 7 中可使用以下版本的 Modbus RTU 指令：

- 版本 3.4：兼容版本 V4.0 及以上的 CPU 和 V2.1 及以上的 CM
- 版本 4.3：兼容 V4.2 及以上版本的 CPU 和 V2.1 及以上版本的 CM

可以从兼容性和移植便利性方面考虑，选择将相应的指令版本插入用户程序中。此外，S7-1200 CPU 还继续支持历史版本的 Modbus RTU 指令 (页 1344)。

不能将两个版本的指令用于同一模块，但不同的模块可以使用不同版本的指令。

在指令任务卡中，在“通信”(Communication) 处理器组中显示 MODBUS (RTU) 指令。



要更改 Modbus RTU 指令的版本，请从下拉列表中选择相应版本。可以选择一组指令或分别选择各个指令。

使用指令树将 Modbus RTU 指令放入程序时，将在项目树中创建新的 FB 实例。在项目树的“PLC_x > 程序块 > 系统块 > 程序资源”(PLC_x > Program blocks > System blocks > Program resources) 下可看到新的 FB 实例。

要确认程序中 Modbus RTU 指令的版本，必须检查项目树的属性而不是程序编辑器中显示的框的属性。选择项目树的 Modbus RTU FB 实例，单击右键，选择“属性”(Properties)，然后选择“信息”(Information) 页查看 Modbus RTU 指令的版本号。

13.5.3.3 最多支持的 Modbus 从站数量

Modbus 寻址支持最多 247 个从站（从站编号 1 到 247）。每个 Modbus 网段最多可以有 32 个设备，具体取决于 RS485 接口的负载和驱动能力。当达到 32 个设备的限制时，必须使用中继器来扩展到下一个网段。需要七个中继器才能将 247 个从站连接到同一个主站的 RS485 接口。

Siemens 中继器仅支持 PROFIBUS；其功能为监视 PROFIBUS 令牌传递。Siemens 中继器不支持其它协议。因此，需要第三方 Modbus 中继器。

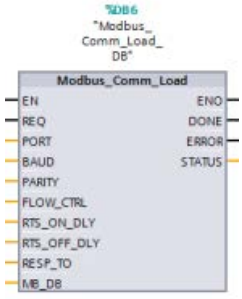
Modbus 超时默认较长；使用多个中继器不会产生延时问题。Modbus 主站不关心从站是否响应慢或者多个中继器是否延迟了响应。

13.5.3.4 Modbus RTU 指令

Modbus_Comm_Load（组态 Modbus RTU 的 PtP 模块上的 SIPLUS I/O 或端口）指令

表格 13-75 Modbus_Comm_Load 指令

LAD/FBD	SCL	说明

	`"Modbus_Comm_Load_DB"(`
	`REQ:=_bool_in,`
	`PORT:=_uint_in_,`
	`BAUD:=_udint_in_,`
	`PARITY:=_uint_in_,`
	`FLOW_CTRL:=_uint_in_,`
	`RTS_ON_DLY:=_uint_in_,`
	`RTS_OFF_DLY:=_uint_in_,`
	`RESP_TO:=_uint_in_,`
	`DONE=>_bool_out,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`MB_DB:=_fbtref_inout_);`
		Modbus_Comm_Load 指令可组态用于 Modbus RTU 协议通信的 SIPLUS I/O 或 PtP 端口。
		Modbus RTU 端口硬件选项：最多安装三个 CM（RS485 或 RS232），及一个 CB (R4845)。
		Modbus RTU SIPLUS I/O 选项：安装 ET 200MP S7-1500CM PtP（RS485/422 或 RS232）或 ET 200SP S7-1500 CM PtP（RS485/422 或 RS232）
		将 Modbus_Comm_Load 指令放入程序时自动分配背景数据块。

表格 13-76 参数的数据类型

参数和类型		数据类型	说明
EN	IN	Bool	注：Modbus RTU、Modbus_Comm_Load 指令使用 RDREC 和 WRREC 指令初始化 PTP 模块。但 RDREC/WRREC 指令异步运行，这意味着需要几次扫描才能完成指令运行。因此，您必须保持 Modbus_Comm_Load 指令的 EN 参数为真，直到 RDREC/WRREC 指令运行结束。
REQ	IN	Bool	通过由低到高的（上升沿）信号启动操作。 （仅版本 2.0）
PORT	IN	Port	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。
BAUD	IN	UDInt	波特率选择： 300、600、1200、2400、4800、9600、19200、38400、57600、76800、115200，其它所有值均无效

参数和类型		数据类型	说明
PARITY	IN	UInt	奇偶校验选择： • 0 – 无 • 1 – 奇校验 • 2 – 偶校验
FLOW_CTRL [1]	IN	UInt	流控制选择： • 0 –（默认）无流控制 • 1 – RTS 始终为 ON 的硬件流控制（不适用于 RS485 端口） • 2 - 带 RTS 切换的硬件流控制
RTS_ON_DLY [1]	IN	UInt	RTS 接通延时选择： • 0 –（默认）从 RTS 激活一直到传送消息的第一个字符之前无延时 • 1 到 65535 – 从 RTS 激活一直到传送消息的第一个字符之前以毫秒表示的延时（不适用于 RS485 端口）。不管 FLOW_CTRL 选择为何，都将应用 RTS 延时。
RTS_OFF_DLY [1]	IN	UInt	RTS 关断延时选择： • 0 –（默认）从传送最后一个字符一直到 RTS 转入非活动状态之前无延时 • 1 到 65535 – 从传送最后一个字符一直到 RTS 转入非活动状态之前以毫秒表示的延时（不适用于 RS485 端口）。不管 FLOW_CTRL 选择为何，都将应用 RTS 延时。
RESP_TO [1]	IN	UInt	响应超时： Modbus_Master 允许用于从站响应的时间（以毫秒为单位）。如果从站在此时间段内未响应，Modbus_Master 将重试请求，或者在发送指定次数的重试请求后终止请求并提示错误。 5 ms 到 65535 ms（默认值 = 1000 ms）。
MB_DB	IN	Variant	对 Modbus_Master 或 Modbus_Slave 指令所使用的背景数据块的引用。在用户的程序中放置 Modbus_Master 或 Modbus_Slave 后，该 DB 标识符将出现在 MB_DB 功能框连接的参数助手下拉列表中。
DONE	OUT	Bool	上一请求已完成且没有出错后，DONE 位将保持为 TRUE 一个扫描周期时间。（仅版本 2.0）
ERROR	OUT	Bool	上一请求因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。STATUS 参数中的错误代码值仅在 ERROR = TRUE 的一个扫描周期内有效。
STATUS	OUT	Word	执行条件代码

[1] Modbus_Comm_Load（V 2.x 或更高版本）的可选参数。单击 LAD/FBD 框底部的箭头，展开框并包含这些参数。

可执行 Modbus_Comm_Load 来组态端口以使用 Modbus RTU 协议。为使用 Modbus RTU 协议组态端口后，该端口只能由 Modbus_Master 或 Modbus_Slave 指令使用。

对用于 Modbus 通信的每个通信端口，都必须执行一次 Modbus_Comm_Load 来组态。为要使用的每个端口分配一个唯一的 Modbus_Comm_Load 背景数据块。最多可在 CPU 中安装三个通信模块（RS232 或 RS485）和一个通信板 (RS485)。从启动 OB 调用 Modbus_Comm_Load 并执行它一次，或使用第一个扫描系统标记 (页 106)发起调用以执行它一次。只有在必须更改波特率或奇偶校验等通信参数时，才再次执行 Modbus_Comm_Load。

如果将 Modbus 库与分布式机架中的模块结合使用，则必须在一个循环中断例程中执行 Modbus_Comm_Load 指令（例如，每秒或每隔 10 秒执行一次）。如果分布式机架的电源中断或者卸下了模块，则在模块恢复运行时，仅向 PtP 模块发送 HWConfig 参数组。由 Modbus_Master 启动的所有请求都会超时，并且 Modbus_Slave 转入静默状态（对任何消息均无响应）。循环执行 Modbus_Comm_Load 解决了这些问题。

将 Modbus_Master 或 Modbus_Slave 指令放入用户程序中时，将为其分配背景数据块。指定 Modbus_Comm_Load 指令的 MB_DB 参数时将引用该背景数据块。

Modbus_Comm_Load 实例数据块 (DB) 变量

下表显示了可在程序中使用的 Modbus_Comm_Load 的背景数据块中的公共静态变量。

表格 13-77 Modbus_Comm_Load 背景数据块静态变量

变量	数据类型	默认值	说明
ICHAR_GAP	Word	0	字符间最大字符延迟时间。该参数以毫秒为单位指定，用于增加接收字符间的预期时间。与此参数对应的位时间个数加到 Modbus 默认值的 35 个位时间（3.5 个字符时间）。
RETRIES	Word	2	在返回错误代码 0x80C8“无响应”之前主站进行的重复尝试次数。
EN_SUPPLY_VOLT	Bool	0	启用对缺失电源电压 L+ 的诊断。

变量	数据类型	默认值	说明
MODE	USInt	0	工作模式 有效工作模式如下： ● 0 = 全双工 (RS232) ● 1 = 全双工 (RS422) 四线制模式（点对点） ● 2 = 全双工 (RS422) 四线制模式（多主站，CM PtP (ET 200SP)） ● 3 = 全双工 (RS422) 四线制模式（多从站，CM PtP (ET 200SP)） ● 4 = 半双工 (RS485) 双线模式（参见下面的注释）
LINE_PRE	USInt	0	接收线路初始状态 有效的初始状态如下： ● 0 =“无”初始状态（参见下面的注释）。 ● 1 = 信号 R(A) = 5 V DC，信号 R(B) = 0 V DC （断路检测）：通过该初始状态可进行断路检测。 仅可配合如下选项使用：“全双工 (RS422) 四线制模式（点对点连接）”和“全双工 (RS422) 四线制模式（多点从站）”。 ● 2 = 信号 R(A) = 0 V DC，信号 R(B) = 5 V DC： 该默认设置对应空闲状态（无激活的发送操作）。无法通过该初始状态进行断路检测。
BRK_DET	USInt	0	断路检测 以下选择有效： ● 0 = 禁止断路检测 ● 1 = 激活断路检测
EN_DIAG_ALARM	Bool	0	激活诊断中断： ● 0 = 未激活 ● 1 = 已激活
STOP_BITS	USInt	1	停止位的数目： ● 1 = 1 个停止位 ● 2 = 2 个停止位 ● 0，3 到 255 = 保留

说明

使用 PROFIBUS 电缆连接 CM 1241 的 RS485 接口时需要此设置

表格 13-78 Modbus_Comm_Load 执行条件代码 [1]

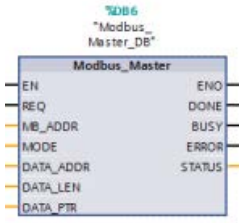
STATUS (W#16#)	说明
0000	无错误
8180	端口 ID 值无效（通信模块的端口/硬件标识符错误）
8181	波特率值无效
8182	奇偶校验值无效
8183	流控制值无效
8184	响应超时值无效（响应超时小于最小值 5 ms）
8185	MB_DB 参数不是 Modbus_Master 或 Modbus_Slave 指令的背景数据块。

[1] 除了上述列出的 Modbus_Comm_Load 错误，还可能返回底层 PtP 通信指令的错误。

Modbus_Master（作为 Modbus RTU 主站通过 SIPLUS I/O 或 PtP 端口通信）指令

表格 13-79 Modbus_Master 指令

LAD/FBD	SCL	说明

	`"Modbus_Master_DB"(`
	`REQ:=_bool_in_,`
	`MB_ADDR:=_uint_in_,`
	`MODE:=_usint_in_,`
	`DATA_ADDR:=_udint_in_,`
	`DATA_LEN:=_uint_in_,`
	`DONE=>_bool_out_,`
	`BUSY=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`DATA_PTR:=_variant_inout_);`
		Modbus_Master 指令作为 Modbus 主站利用之前执行 Modbus_Comm_Load 指令组态的端口进行通信。将 Modbus_Master 指令放入程序时自动分配背景数据块。指定 Modbus_Comm_Load 指令的 MB_DB 参数时将使用该 Modbus_Master 背景数据块。

表格 13-80 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	0 = 无请求 1 = 请求将数据传送到 Modbus 从站
MB_ADDR	IN	V1.0: USInt V2.0: UInt	Modbus RTU 站地址： 标准寻址范围（1 到 247） 扩展寻址范围（1 到 65535） 值 0 被保留用于将消息广播到所有 Modbus 从站。只有 Modbus 功能代码 05、06、15 和 16 是可用于广播的功能代码。

参数和类型		数据类型	说明
MODE	IN	USInt	模式选择：指定请求类型（读、写或诊断）。请参见下面的 Modbus 功能表了解详细信息。
DATA_ADDR	IN	UDInt	从站中的起始地址：指定要在 Modbus 从站中访问的数据的起始地址。请参见下面的 Modbus 功能表了解有效地址信息。
DATA_LEN	IN	UInt	数据长度：指定此请求中要访问的位数或字数。请参见下面的 Modbus 功能表了解有效长度信息。
DATA_PTR	IN_OUT	Variant	数据指针：指向要写入或读取的数据的 M 或 DB 地址（未经优化的 DB 类型）。
DONE	OUT	Bool	上一请求已完成且没有出错后，DONE 位将保持为 TRUE 一个扫描周期时间。
BUSY	OUT	Bool	• 0 - 无 Modbus_Master 操作正在进行 • 1 - Modbus_Master 操作正在进行
ERROR	OUT	Bool	上一请求因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。STATUS 参数中的错误代码值仅在 ERROR = TRUE 的一个扫描周期内有效。
STATUS	OUT	Word	执行条件代码

Modbus_Master 通信规则

- 必须先执行 Modbus_Comm_Load 组态端口，然后 Modbus_Master 指令才能与该端口通信。
- 如果要将某个端口用于初始化 Modbus 主站请求，则 Modbus_Slave 不应使用该端口。Modbus_Master 执行的一个或多个实例可使用该端口，但是对于该端口，所有 Modbus_Master 执行都必须使用同一个 Modbus_Master 背景数据块。
- Modbus 指令不使用通信中断事件来控制通信过程。用户程序必须轮询 Modbus_Master 指令以了解传送和接收的完成情况。

- 对于给定的端口，从程序循环 OB 中调用所有 Modbus_Master 执行。Modbus_Master 指令只能在一个程序循环或循环/延时执行等级执行。它们不能同时在两种执行优先级中执行。如果一个 Modbus_Master 指令被另一个执行优先级更高的 Modbus_Master 取代，将导致不正确的操作。Modbus_Master 指令不能在启动、诊断或时间错误执行优先级执行。
- Modbus_Master 指令启动传输后，必须连续执行已启用 EN 输入的该实例，直到返回状态 DONE=1 或状态 ERROR=1 为止。在这两个事件其中之一发生前，一个特殊的 Modbus_Master 实例被视为已激活。原始实例激活后，调用已启用 REQ 输入的其它任何实例都将导致错误。如果原始实例的连续执行过程停止，则请求状态保持激活一段时间，该时间由静态变量“Blocked_Proc_Timeout”指定。一旦超出该时间段，则下一个使用激活的 REQ 输入调用的 Modbus_Master 指令成为激活实例。这可以防止单个 Modbus_Master 指令独占或锁定对端口的访问。如果在由静态变量“Blocked_Proc_Timeout”指定的时间段内没有启用原始激活的实例，则下次执行此实例（未设置 REQ）时将清除激活状态。如果设置了 REQ，则此次执行将启动新的 Modbus_Master 请求，如同其它实例未曾激活一样。

REQ 参数

0 = 无请求；1 = 请求将数据传送到 Modbus 从站

可使用电平或边沿触发的触点控制此输入。只要此输入启用，状态机便会启动，以确保在当前请求完成前不允许使用同一背景数据块的任何其它 Modbus_Master 发出请求。在当前请求执行期间，将捕获所有其它输入状态并内部保存，直到接收到响应或检测到错误。

如果在当前请求完成前 REQ 输入 = 1，从而再次执行 Modbus_Master 的同一实例，则不会进行任何后续传送。但是，如果当前请求已完成，因为 REQ 输入 = 1 而再次执行 Modbus_Master 时，便会发出新请求。

DATA_ADDR 和 MODE 参数用于选择 Modbus 功能类型

DATA_ADDR（从站中的 Modbus 起始地址）：指定要在 Modbus 从站中访问的数据的起始地址。

Modbus_Master 指令使用 MODE 输入而非功能代码输入。MODE 和 Modbus 地址一起确定实际 Modbus 消息中使用的功能代码。下表列出了 MODE 参数、Modbus 功能代码和 Modbus 地址范围之间的对应关系。

表格 13-81 Modbus 功能

MODE	Modbus 功能	数据长度	操作和数据	Modbus 地址
0	01	1 到 2000 1 到 1992 [1]	读取输出位： 每个请求 1 到 1992 或 2000 个位	1 到 9999
0	02	1 到 2000 1 到 1992 [1]	读取输入位： 每个请求 1 到 1992 或 2000 个位	10001 到 19999
0	03	1 到 125 1 到 124 [1]	读取保持寄存器： 每个请求 1 到 124 或 125 个字	40001 到 49999 或 400001 到 465535
0	04	1 到 125 1 到 124 [1]	读取输入字： 每个请求 1 到 124 或 125 个字	30001 到 39999
104	04	1 到 125 1 到 124 1	读取输入字： 每个请求 1 到 124 或 125 个字	00000 到 65535
1	05	1	写入一个输出位： 每个请求一位	1 到 9999
1	06	1	写入一个保持寄存器： 每个请求 1 个字	40001 到 49999 或 400001 到 465535
1	15	2 到 1968 2 到 1960 [1]	写入多个输出位： 每个请求 2 到 1960 或 1968 个位	1 到 9999
1	16	2 到 123 2 到 122 [1]	写入多个保持寄存器： 每个请求 2 到 122 或 123 个字	40001 到 49999 或 400001 到 465535
2	15	1 到 1968 2 到 1960 [1]	写入一个或多个输出位： 每个请求 1 到 1960 或 1968 个位	1 到 9999
2	16	1 到 123 1 到 122 [1]	写入一个或多个保持寄存器： 每个请求 1 到 122 或 123 个字	40001 到 49999 或 400001 到 465535
11	11	0	读取从站通信状态字和事件计数器。状态字指示忙闲情况（0 - 不忙，0xFFFF - 忙）。每成功完成一条消息，事件计数器的计数值递增。 对于该功能，Modbus_Master 的 DATA_ADDR 和 DATA_LEN 操作数都将被忽略。	

MODE	Modbus 功能	数据长度	操作和数据	Modbus 地址
80	08	1	利用数据诊断代码 0x0000 检查从站状态（回送测试 - 从站回送请求） 每个请求 1 个字	
81	08	1	利用数据诊断代码 0x000A 重新设置从站事件计数器 每个请求 1 个字	
3 到 10、12 到 79、82 到 255			保留	

[1] 对于“扩展寻址”模式，根据功能所使用的数据类型，数据的最大长度将减小 1 个字节或 1 个字。

DATA_PTR 参数

DATA_PTR 参数指向要写入或读取的 DB 或 M 地址。如果使用数据块，则必须创建一个全局数据块为读写 Modbus 从站提供数据存储位置。

说明

DATA_PTR 数据块类型必须允许直接寻址

该数据块必须允许直接（绝对）寻址和符号寻址。创建该数据块时，必须选择“标准” (Standard) 访问属性。

自 Modbus_Master 指令版本 V4.0 或更高版本起，可以启用数据块属性“优化块访问” (Optimized block access)。只能在具有以下数据类型的优化存储器中使用单个元素或元素数组：Bool、Byte、Char、Word、Int、DWord、Dint、Real、USInt、UInt、UDInt、SInt 或 WChar。

DATA_PTR 参数的数据块结构

- 这些数据类型对 Modbus 地址 30001 到 39999、40001 到 49999 和 400001 到 465536 的**字读取**有效，对 Modbus 地址 40001 到 49999 和 400001 到 465536 的**字写入**也有效。
  - WORD、UINT 或 INT 数据类型的标准数组
  - 指定的 WORD、UINT 或 INT 结构，其中每个元素都具有唯一的名称和 16 位数据类型。
  - 指定的复杂结构，其中每个元素都具有唯一的名称以及 16 或 32 位数据类型。
- 用于 Modbus 地址 00001 到 09999 的**位读取**和写入和 10001 到 19999 的位读取。
  - 布尔数据类型的标准数组。
  - 唯一命名的布尔变量的已命名布尔结构。
- 尽管不是必需的，但还是建议每个 Modbus_Master 指令都具有各自的单独存储区。此建议的原因在于，如果多个 Modbus_Master 指令读取和写入同一个存储区，发生数据损坏的可能性会更大。
- 不要求 DATA_PTR 数据区位于同一个全局数据块中。可创建一个具有多个区域的数据块供 Modbus 读取、一个数据块供 Modbus 写入或一个数据块用于各个从站。

Modbus_Master 指令数据块 (DB) 变量

下表显示了可在程序中使用的 Modbus_Master 的背景数据块中的公共静态变量。

表格 13-82 Modbus_Master 背景数据块静态变量

Tag	数据类型	默认值	说明
Blocked_Proc_Timeout	Real	3.0	在 Modbus_Master 实例受阻后，移除该激活的实例前需等待的时间（秒）。例如，当已发出 Modbus_Master 请求，但程序在彻底完成该请求前停止调用该 Modbus_Master 功能时，就会出现这种情况。时间值必须大于 0 且小于 55 秒，否则发生错误。
Extended_Addressing	Bool	FALSE	组态单字节或双字节从站寻址： • FALSE = 单字节地址；0 到 247 • TRUE = 双字节地址（相当于扩展寻址）；0 到 65535
MB_DB	MB_BASE	-	Modbus_Comm_Load 指令的 MB_DB 参数必须连接 Modbus_Master 指令的 MB_DB 参数。

用户程序可以将值写入 Blocked_Proc_Timeout 和 Extended_Addressing 变量，以控制 Modbus_Master 操作。有关如何在程序编辑器中使用这些变量的示例以及有关 Modbus 扩展寻址的详细信息，请参见 HR_Start_Offset (页 1279) 和 Extended_Addressing (页 1279) 的 Modbus_Slave 主题说明。

条件代码

表格 13-83 Modbus_Master 执行条件代码（通信和组态错误）[1]

STATUS (W#16#)	说明
0000	无错误
80C8	从站超时。指定从站在指定时间内没有响应。请检查从站设备的波特率、奇偶性和接线。尝试过所有组态的重试操作后，才警告此错误。
80C9	Modbus_Master 指令因以下原因发生超时： ● 该指令正在等待来自模块的响应，而该模块正用于通信。 ● Blocked_Proc_Timeout 值设置得太小。 如果 PROFIBUS 或 PROFINET 分布式 I/O 设备从以下状况之一返回，就会报告该错误： ● 电源或通信中断 ● 通信模块插/拔事件 在这些情况下，将重新加载 PLC 的硬件组态，并且必须再次执行 Modbus_Comm_Load 才能正确组态通信模块。
80D1	接收方发出了暂停主动传输的流控制请求并且在指定的等待时间内未重新激活该传输。 在硬件流控制期间，如果接收方在指定的等待时间内没有声明 CTS，也会产生该错误。
80D2	传送请求中止，因为没有从 DCE 收到任何 DSR 信号。
80E0	因接收缓冲区已满，消息被终止。
80E1	因出现奇偶校验错误，消息被终止。
80E2	因组帧错误，消息被终止。
80E3	因出现超限错误，消息被终止。
80E4	因指定长度超出总缓冲区大小，消息被终止。
8180	无效端口 ID 值或 Modbus_Comm_Load 指令出错
8186	Modbus 站地址无效
8188	指定给广播请求的模式无效
8189	数据地址值无效
818A	数据长度值无效
818B	指向本地数据源/目标的指针无效：大小不正确

STATUS (W#16#)	说明
818C	DATA_PTR 的指针无效或 Blocked_Proc_Timeout 无效。数据区域必须是以下之一： ● 典型数据块 ● 符号或保持数据块中基本数据类型的数组 ● M 存储器
8200	端口正忙于处理传送请求。
8280	读取模块时否定确认。检查 PORT 参数处的输入。这种情况的可能原因是 PROFIBUS 或 PROFINET 分布式 I/O 模块断开，可由电源中断或拉动模块引起。
8281	写入模块时否定确认。检查 PORT 参数处的输入。这种情况的可能原因是 PROFIBUS 或 PROFINET 分布式 I/O 模块断开，可由电源中断或拉动模块引起。

表格 13-84 Modbus_Master 执行条件代码（Modbus 协议错误）[1]

STATUS (W#16#)	从站的响应代码	Modbus 协议错误
8380	-	CRC 错误
8381	01	不支持此功能代码
8382	03	数据长度错误
8383	02	数据地址错误或地址超出 DATA_PTR 区的有效范围
8384	大于 03	数据值错误
8385	03	不支持此数据诊断代码值（功能代码 08）
8386	-	响应中的功能代码与请求中的代码不匹配。
8387	-	响应的从站错误
8388	-	从站对写请求的响应不正确。从站返回的写请求与主站实际发送的写请求不匹配。

[1] 除了上述列出的 Modbus_Master 错误，还可能返回底层 PtP 通信指令的错误。

说明

设置 Profibus 通信的最大记录长度

在使用 CM1243-5 Profibus 主站模块控制使用 RS232、RS422 或 RS485 点对点模块的 ET 200SP 或 ET 200MP Profibus 设备时，需要按如下规定将“max_record_len”数据块变量明确设置为 240：

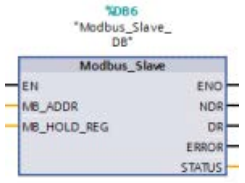
运行 Modbus_Comm_Load 后，在背景数据块的 Send_P2P 部分（例如，"Modbus_Master_DB".Send_P2P.max_record_len）将“max_record_len”设为 240。

只有 Profibus 通信需要明确分配 max_record_len ；Profinet 通信已经使用有效的 max_record_len 值。

Modbus_Slave（作为 Modbus RTU 从站通过 SIPLUS I/O 或 PtP 端口进行通信）指令

表格 13-85 Modbus_Slave 指令

LAD/FBD	SCL	说明

	`"Modbus_Slave_DB"(`
	`MB_ADDR:=_uint_in_,`
	`NDR=>_bool_out_,`
	`DR=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`MB_HOLD_REG:=_variant_inout_);`
		Modbus_Slave 指令允许用户程序用以下两种方式进行通信：
		● 作为 Modbus RTU 从站通过 CM（RS485 或 RS232）和 CB (RS485) 上的 PtP 端口进行通信
		● 作为 Modbus RTU 从站通过 Modbus RTU SIPLUS I/O 选项进行通信：
		– 安装 ET 200MP S7-1500CM PtP（RS485/422 或 RS232）。
		– 安装 ET 200SP S7-1500 CM PtP（RS485/422 或 RS232）。
		远程 Modbus RTU 主站发出请求时，用户程序会通过执行 Modbus_Slave 进行响应。STEP 7 在插入指令时自动创建背景数据块。在为 Modbus_Comm_Load 指令指定 MB_DB 参数时使用此 Modbus_Slave_DB 名称。

表格 13-86 参数的数据类型

参数和类型		数据类型	说明
MB_ADDR	IN	V1.0: USInt V2.0: UInt	Modbus 从站的站地址： 标准寻址范围（1 到 247） 扩展寻址范围（0 到 65535）
MB_HOLD_REG	IN_OUT	Variant	指向 Modbus 保持寄存器 DB 的指针：Modbus 保持寄存器可以是 M 存储器或数据块。
NDR	OUT	Bool	新数据就绪： • 0 – 无新数据 • 1 – 表示 Modbus 主站已写入新数据
DR	OUT	Bool	数据读取： • 0 – 无数据读取 • 1 – 表示 Modbus 主站已读取数据
ERROR	OUT	Bool	上一请求因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。如果执行因错误而终止，则 STATUS 参数的错误代码值仅在 ERROR = TRUE 的一个扫描周期内有效。
STATUS	OUT	Word	执行错误代码

Modbus 通信功能代码（1、2、4、5 和 15）可以在 CPU 的输入过程映像及输出过程映像中直接读写位和字。对于这些功能代码，MB_HOLD_REG 参数必须定义为大于一个字节的数据类型。下表给出了 Modbus 地址与 CPU 过程映像的映射示例。

表格 13-87 Modbus 地址到过程映像的映射

Modbus 功能						S7-1200	
代码	功能	数据区	地址范围			数据区	CPU 地址
01	读位	输出	1	到	8192	输出过程映像	Q0.0 到 Q1023.7
02	读位	输入	10001	到	18192	输入过程映像	I0.0 到 I1023.7
04	读字	输入	30001	到	30512	输入过程映像	IW0 到 IW1022
05	写位	输出	1	到	8192	输出过程映像	Q0.0 到 Q1023.7
15	写位	输出	1	到	8192	输出过程映像	Q0.0 到 Q1023.7

Modbus 通信功能代码（3、6、16）使用 Modbus 保持寄存器，该寄存器可以是 M 存储区地址范围或数据块。保持寄存器的类型由 Modbus_Slave 指令的 MB_HOLD_REG 参数指定。

说明

MB_HOLD_REG 数据块类型

Modbus 保持寄存器数据块必须允许直接（绝对）寻址和符号寻址。创建该数据块时，必须选择“标准”(Standard) 访问属性。

自 Modbus_Slave 指令版本 V4.0 或更高版本起，可以启用数据块属性“优化块访问”(Optimized block access)。只能在具有以下数据类型的优化存储器中使用单个元素或元素数组：Bool、Byte、Char、Word、Int、DWord、Dint、Real、USInt、UInt、UDInt、SInt 或 WChar。

下表给出了 Modbus 地址到保持寄存器的映射示例，这种映射用于 Modbus 功能代码 03（读取字）、06（写入字）和 16（读取字）。DB 地址的实际上限取决于每种 CPU 型号的最大工作存储器限值和 M 存储器限值。

表格 13-88 Modbus 地址到 CPU 存储器的映射

Modbus 主站地址	MB_HOLD_REG 参数示例				
	MW100	DB10.DBw0	MW120	DB10.DBW50	"Recipe".ingredient
40001	MW100	DB10.DBW0	MW120	DB10.DBW50	"Recipe".ingredient[1]
40002	MW102	DB10.DBW2	MW122	DB10.DBW52	"Recipe".ingredient[2]
40003	MW104	DB10.DBW4	MW124	DB10.DBW54	"Recipe".ingredient[3]
40004	MW106	DB10.DBW6	MW126	DB10.DBW56	"Recipe".ingredient[4]
40005	MW108	DB10.DBW8	MW128	DB10.DBW58	"Recipe".ingredient[5]

表格 13-89 诊断功能

S7-1200 Modbus_Slave Modbus 诊断功能		
代码	子功能	说明
08	0000H	返回查询数据回送测试： • 在 STEP 7 V15.1 版本之前，Modbus_Slave 将向 Modbus 主站回送接收到的数据字。 • 自 STEP 7 V15.1 或更高版本起，Modbus_Slave 指令 V4.1 或更高版本会回送接收到的一个或多个数据字。

S7-1200 Modbus_Slave Modbus 诊断功能		
08	000AH	清除通信事件计数器：Modbus_Slave 将清除用于 Modbus 功能 11 的通信事件计数器。
11		获取通信事件计数器：Modbus_Slave 使用内部通信事件计数器来记录发送到 Modbus_Slave 的 Modbus 成功读取和写入请求次数。该计数器不会因功能 8、功能 11 或广播请求而增加。同样也不会因任何导致通信错误（例如，奇偶校验错误或 CRC 错误）的请求而增加。

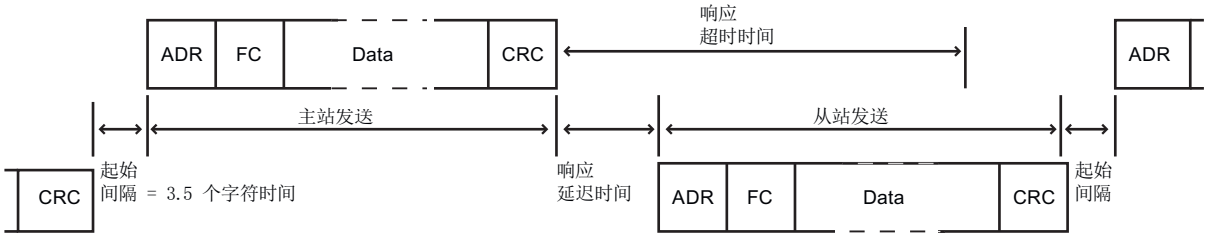
Modbus_Slave 指令支持来自任何 Modbus 主站的广播写请求，只要该请求是用于访问有效地址的请求即可。对于广播不支持的功能代码，Modbus_Slave 将生成错误代码“0x8188”。

Modbus_Slave 通信规则

- 必须先执行 Modbus_Comm_Load 组态端口，然后 Modbus_Slave 指令才能通过该端口通信。
- 如果某个端口作为从站响应 Modbus_Master，则请勿使用 Modbus_Master 指令对该端口进行编程。
- 对于给定端口，只能使用一个 Modbus_Slave 实例，否则将出现不确定的行为。
- Modbus 指令不使用通信中断事件来控制通信过程。用户程序必须通过轮询 Modbus_Slave 指令以了解传送和接收的完成情况来控制通信过程。
- Modbus_Slave 指令必须以一定的速率定期执行，以便能够及时响应来自 Modbus_Master 的进入请求。建议每次扫描时都从程序循环 OB 执行 Modbus_Slave。也可以从循环中断 OB 执行 Modbus_Slave，但并不建议这么做，因为中断例程的延时过长可能会暂时阻止其它中断例程的执行。

Modbus 定时信号

必须周期性执行 Modbus_Slave，才能接收来自 Modbus_Master 的每个请求并随之按要求响应。Modbus_Slave 的执行频率取决于 Modbus_Master 的响应超时时间。下图对此进行了说明。



响应超时时间 RESP_TO 是 Modbus_Master 等待 Modbus_Slave 开始响应的时间。该时间段不是由 Modbus 协议定义的，而是属于每个 Modbus_Master 的一个参数。必须基于用户 Modbus_Master 的具体参数确定 Modbus_Slave 的执行频率（相邻两次执行之间的时间）。在 Modbus_Master 的响应超时时间内至少应执行两次 Modbus_Slave。

Modbus_Slave 指令数据块 (DB) 变量

下表显示了可在程序中使用的 Modbus_Slave 的背景数据块中的公共静态变量。

表格 13-90 Modbus_Slave 背景数据块静态变量

变量	数据类型	默认值	说明
HR_Start_Offset	Word	0	分配 Modbus 保持寄存器的起始地址（默认值 = 0）
Extended_Addressing	Bool	FALSE	组态单字节或双字节从站寻址： • FALSE = 单字节地址 • TRUE = 双字节地址
Request_Count	Word	0	该从站接收到的所有请求的数量
Slave_Message_Count	Word	0	该特定从站接收到的请求的数量
Bad_CRC_Count	Word	0	接收到的具有 CRC 错误的请求的数量
Broadcast_Count	Word	0	接收到的广播请求的数量
Exception_Count	Word	0	需要通过向主站返回异常来确认的 Modbus 特定错误
Success_Count	Word	0	该特定从站接收到的没有协议错误的请求数量
MB_DB	MB_BASE	-	Modbus_Comm_Load 指令的 MB_DB 参数必须连接 Modbus_Slave 指令的 MB_DB 参数。
QB_Start	UInt	0	CPU 可写入的输出字节的起始地址（QB0 至 QB65535）
QB_Count	UInt	65535	远程设备可以写入的字节数。如果 QB_Count = 0，则远程设备无法写入输出。 示例：要想只允许 QB10 到 QB17 可写入，则 QB_Start = 10 且 QB_Count = 8。
QB_Read_Start	UInt	0	CPU 可读取的输出字节的起始地址（QB0 至 QB65535）
QB_Read_Count	UInt	65535	远程设备可以读取的输出字节数。如果 QB_Count = 0，则远程设备无法读取输出。示例：要想只允许 QB10 到 QB17 可读取，则 QB_Start = 10 且 QB_Count = 8。

变量	数据类型	默认值	说明
IB_Read_Start	UInt	0	CPU 可读取的输入字节的起始地址（IB0 至 IB65535）
IB_Read_Count	UInt	65535	远程设备可以读取的输入字节数。如果 IB_Count = 0，则远程设备无法读取输入。示例：要想只允许 IB10 到 IB17 可读取，则 IB_Start = 10 且 IB_Count = 8。

用户程序可以将数据写入控制 Modbus 服务器操作和以下变量中：

- HR_Start_Offset
- Extended_Addressing
- QB_Start
- QB_Count
- QB_Read_Start
- QB_Read_Count
- IB_Read_Start
- IB_Read_Count

Modbus_Slave 指令数据块 (DB) 变量可用性的版本要求如下：

表格 13-91 Modbus_Slave 指令数据块 (DB) 变量可用性的版本要求：指令、TIA Portal 和 S7-1200 CPU

Modbus_Slave 指令版本	TIA Portal 的版本	S7-1200 CPU 固件 (FW) 版本	数据块变量
3.0	V14 SP1	CPU 固件 V4.0 或更高版本	QB_Start
			QB_Count
4.0 或以上版本	V15 或以上版本	CPU 固件 V4.2 或更高版本	QB_Start
			QB_Count
			QB_Read_Start
			QB_Read_Count
			IB_Read_Start
			IB_Read_Count

HR_Start_Offset

Modbus 保持寄存器地址从 40001 或 400001 开始。这些地址与保持寄存器的 PLC 存储器起始地址对应。不过，可以组态“HR_Start_Offset”变量，将 Modbus 保持寄存器的起始地址定义为除 40001 或 400001 之外的其它值。

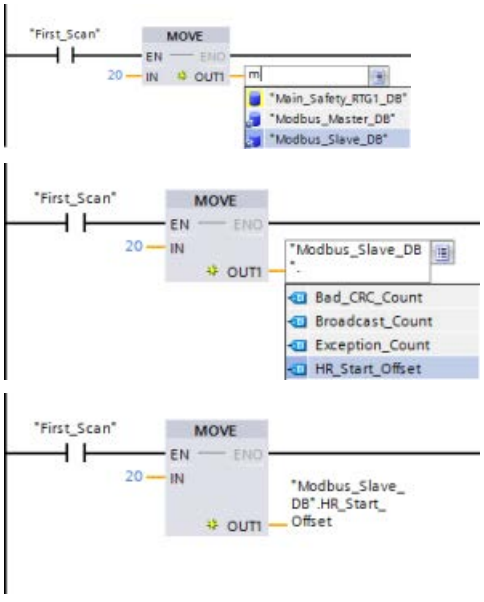
例如，如果保持寄存器被组态为起始于 MW100 并且长度为 100 个字。偏移量 20 可指定保持寄存器的起始地址为 40021 而不是 40001。低于 40021 和高于 400119 的任何地址都将导致寻址错误。

表格 13-92 Modbus 保持寄存器寻址示例

HR_Start_Offset	地址	最小值	最大值
0	Modbus 地址（字）	40001	40099
	S7-1200 地址	MW100	MW298
20	Modbus 地址（字）	40021	40119
	S7-1200 地址	MW100	MW298

HR_Start_Offset 是一个字值，用于指定 Modbus 保持寄存器的起始地址，存储在 Modbus_Slave 背景数据块中。将 Modbus_Slave 放入程序后，可利用参数助手下拉列表设置该公共静态变量值。

例如，将 Modbus_Slave 放入 LAD 程序段后，可以切换到先前的程序段，分配 HR_Start_Offset 值。该值必须在执行 Modbus_Slave 前分配。



使用默认 DB 名称输入 Modbus 从站变量：

1. 将光标放在参数字段中，然后输入 m 字符。
2. 从下拉列表中选择“Modbus_Slave_DB”。
3. 将光标放在 DB 名称的右侧（引号字符的后面），然后输入句点字符。
4. 从下拉列表中选择“Modbus_Slave_DB.HR_Start_Offset”。

Extended_Addressing

Extended_Addressing 变量的访问方式与上述的 HR_Start_Offset 参考相似，只是 Extended_Addressing 变量是布尔值。布尔值必须通过输出线圈（而非 MOVE 块）写入。

Modbus 从站寻址可组态为单字节（Modbus 标准方式）或双字节。扩展寻址用于对单一网络内超过 247 台设备进行寻址。选择扩展寻址后，最多可以对 64000 个地址进行寻址。下面以 Modbus 功能 1 的帧为例进行显示。

表格 13-93 单字节从站地址（字节 0）

功能 1	字节 0	字节 1	字节 2	字节 3	字节 4	字节 5	
请求	从站地址	F 代码	起始地址		线圈长度		
有效响应	从站地址	F 代码	长度	线圈数据			
错误响应	从站地址	0x81	E 代码				

表格 13-94 双字节从站地址（字节 0 和字节 1）

	字节 0	字节 1	字节 2	字节 3	字节 4	字节 5	字节 6
请求	从站地址		F 代码	起始地址		线圈长度	
有效响应	从站地址		F 代码	长度	线圈数据		
错误响应	从站地址		0x81	E 代码			

访问数据块 (DB) 中的数据区域，而不是直接访问 Modbus 地址

自 Modbus_Slave 指令版本 V4.0 以及 S7-1200 CPU 固件 (FW) 版本 V4.2 起，用户可访问数据块中的数据区域，而不用直接访问过程映像和保持寄存器。为此，在全局 DB“属性”(Attributes) 属性页中，必须取消选中“仅存储在装载存储器中”(Only store in load memory) 和“优化块访问”(Optimized block access) 复选框。

如果 Modbus 请求到达时尚未为相应功能代码的 Modbus 数据类型定义数据区域，则 Modbus_Slave 指令会按之前的指令版本处理请求：直接访问过程映像和保持寄存器。

如果已为功能代码的 Modbus 数据类型定义了数据区域，则 Modbus_Slave 指令会对该数据区域进行读写操作。具体是读操作还是写操作取决于作业类型。

说明

如果组态了数据区域，则 Modbus_Slave 指令会忽略与数据区域的 data_type 对应的背景数据块中，静态变量组态的偏移或范围。这些偏移和范围仅适用于 MB_HOLD_REG 所引用的过程映像或存储器。数据区域的启动和长度参数会提供自身定义偏移和范围的方法

单个 Modbus 请求只能对一个数据区域进行读写操作。如果要读取覆盖多个数据区域的保持寄存器，则需要多个 Modbus 请求。

数据区域的定义规则如下：

- 用户最多可在不同数据块中定义八个数据区域，每个数据块只能包含一个数据区域。单个 MODBUS 请求只能对恰好一个数据区域进行读写操作。每个数据区域对应于一个 MODBUS 地址区域。可以在实例数据块的“Data_Area_Array”静态变量中定义数据区域。
- 如果要使用的数据区域不到八个，则所需数据区域必须紧密相连，没有间隙。在处理过程中，数据区域中的第一个空白条目会终止数据区域搜索。例如，如果已定义字段元素 1、2、4 和 5，由于字段元素 3 留空，则“Data_Area_Array”只会识别字段元素 1 和 2。

- Data_Area_Array 字段包含八个元素：Data_Area_Array[1] 到 Data_Area_Array[8]
- 每个字段元素 Data_Area_Array[x]（其中 1 <= x <= 8）都是 MB_DataArea 类型的 UDT，结构如下：

参数	数据类型	含义
data_ type	UInt	映射到此数据区域的 MODBUS 数据类型的标识符： • 0：空字段元素或未使用数据区域的标识符。此时，数据块、起始和长度的值不相关。 • 1：过程映像输出（与功能代码 1、5 和 15 一起使用） • 2：过程映像输入（与功能代码 2 一起使用） • 3：保持寄存器（与功能代码 3、6 和 16 一起使用） • 4：输入寄存器（与功能代码 4 一起使用） 注：如果已定义 MODBUS 数据类型的数据区域，则指令 Modbus_Slave 不能再直接访问此 MODBUS 数据类型。如果该数据类型的 MODBUS 请求地址与定义的数据区域不对应，则 STATUS 中会返回值 W#16#8383。
db	UInt	MODBUS 寄存器或后续定义的位所映射的目标数据块的编号 数据块编号在数据区域中必须是唯一的。不得在多个数据区域中定义相同的数据块编号。 在全局 DB“属性”(Attributes) 属性页中，必须取消选中“仅存储在装载存储器中”(Only store in load memory) 和“优化块访问”(Optimized block access) 复选框。 数据区域也是从数据块的字节地址 0 开始。 允许值：1 到 60999
起始	UInt	映射到数据块中的首个 MODBUS 地址（从地址 0.0 开始） 允许值：0 到 65535
长度	UInt	位数（对于 data_type 的值 1 和 2）或寄存器数量（对于 data_type 的值 3 和 4）。 相同 MODBUS 数据类型的 MODBUS 地址区域不得重叠。 允许值：1 到 65535

数据区域定义示例：

- 第一个示例：data_type = 3，db = 1，start = 10，length = 6
  CPU 将保持寄存器 (data_type = 3) 映射到数据块 1 (db = 1)，将 Modbus 地址 10 (start = 10) 置于数据字 0，将最后一个有效 Modbus 地址 15 (length = 6) 置于数据字 5。
- 第二个示例：data_type = 2，db = 15，start = 1700，length = 112
  CPU 将输入 (data_type = 2) 映射到数据块 15 (db = 15)，将 Modbus 地址 1700 (start = 1700) 置于数据字 0，将最后一个有效 Modbus 地址 1811 (length = 112) 置于数据字 111。

条件代码

表格 13-95 Modbus_Slave 执行条件代码（通信和组态错误）[1]

STATUS (W#16#)	说明
80D1	接收方发出了暂停主动传输的流控制请求并且在指定的等待时间内未重新激活该传输。 在硬件流控制期间，如果接收方在指定的等待时间内没有声明 CTS，也会产生该错误。
80D2	传送请求中止，因为没有从 DCE 收到任何 DSR 信号。
80E0	因接收缓冲区已满，消息被终止。
80E1	因出现奇偶校验错误，消息被终止。
80E2	因组帧错误，消息被终止。
80E3	因出现超限错误，消息被终止。
80E4	因指定长度超出总缓冲区大小，消息被终止。
8180	无效端口 ID 值或 Modbus_Comm_Load 指令出错
8186	Modbus 站地址无效
8187	指向 MB_HOLD_REG DB 的指针无效：区域太小
818C	MB_HOLD_REG 指针无效。数据区域必须是以下之一： ● 典型数据块 ● 符号或保持数据块中基本数据类型的数组 ● M 存储器

表格 13-96 Modbus_Slave 执行条件代码（Modbus 协议错误）[1]

STATUS (W#16#)	从站的响应代码	Modbus 协议错误
8380	无响应	CRC 错误
8381	01	不支持功能代码或在广播内不支持
8382	03	数据长度错误
8383	02	数据地址错误或地址超出 DATA_PTR 区的有效范围
8384	03	数据值错误
8385	03	不支持此数据诊断代码值（功能代码 08）
8389		数据区域定义无效： • data_type 值无效 • 数据块编号无效或不存在： – db 值无效 – 数据块编号不存在 – 数据块编号已由另一个数据区域使用 – 数据块具有优化访问权限 – 数据块不在工作存储器中 • 长度值无效 • 属于同一 MODBUS 数据类型的 MODBUS 地址范围重叠

[1] 除了上述列出的 Modbus_Slave 错误，还可能返回底层 PtP 通信指令的错误。

说明

设置 PROFIBUS 通信的最大记录长度

在使用 CM1243-5 PROFIBUS 主站模块控制使用 RS232、RS422 或 RS485 点对点模块的 ET 200SP 或 ET 200MP PROFIBUS 设备时，需要按如下规定将“max_record_len”数据块变量明确设置为 240：

运行 Modbus_Comm_Load 后，在背景数据块的 Send_P2P 部分（例如，"Modbus_Slave_DB".Send_P2P.max_record_len）将“max_record_len”设为 240。

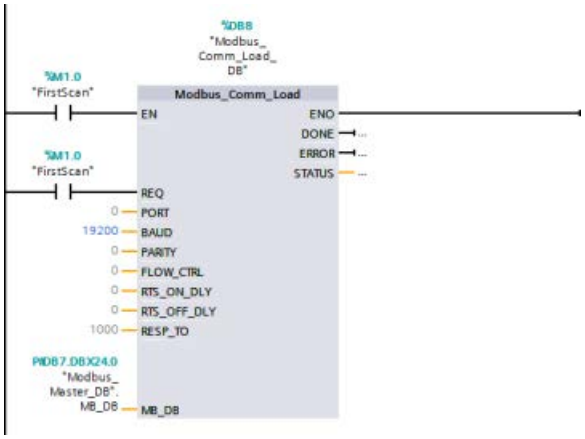
只有 PROFIBUS 通信需要明确分配 max_record_len ；Profinet 通信已经使用有效的 max_record_len 值。

13.5.3.5 Modbus RTU 示例

示例： Modbus RTU 主站程序

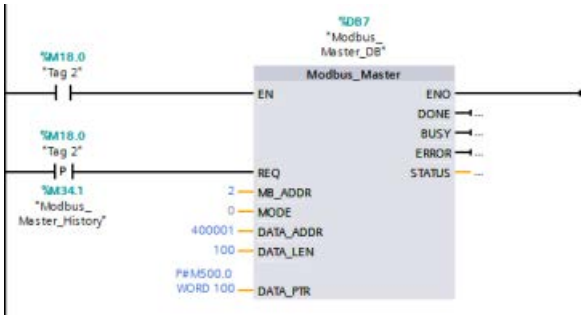
启动期间通过第一个扫描标志启用 Modbus_Comm_Load。通过此方式执行 Modbus_Comm_Load 时，必须保证串口组态在运行时不会更改。

程序段 1：仅在第一次扫描期间对 RS485 模块通信端口进行一次组态/初始化。

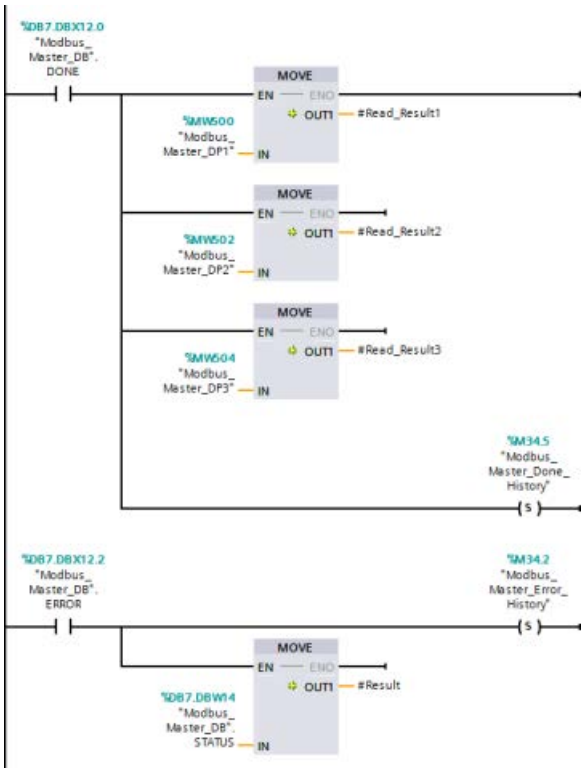


在程序循环 OB 中使用一个 Modbus_Master 指令，以与单个从站进行通信。要与其它从站通信，可在程序循环 OB 中使用另外的 Modbus_Master 指令，也可以重新使用一个 Modbus_Master FB。

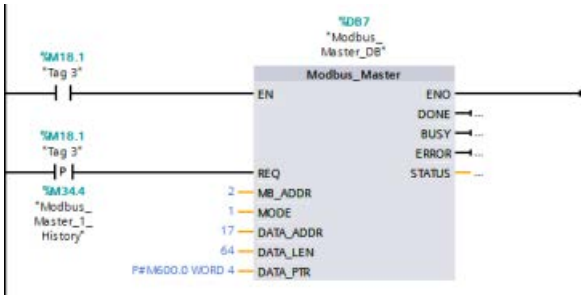
程序段 2：从从站 #2 上的位置 400001 读取 100 个字保持寄存器数据到存储器位置 MW500-MW698。



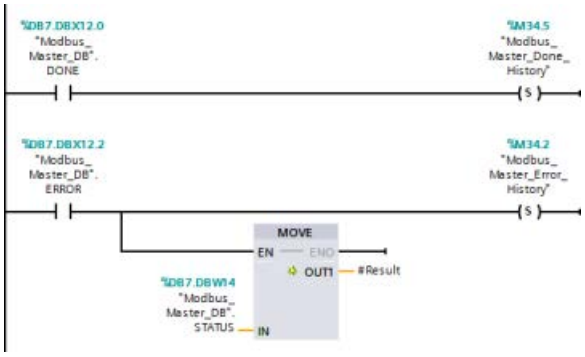
程序段 3：移动读取到其他位置的保持寄存器数据的前三个字，然后置位 DONE 历史位。一旦发生错误，该程序段还将置位错误历史位并将状态字保存到另一个位置。



程序段 4：将 MW600-MW607 的 64 位数据写入从站 #2 上的 00017 到 00081 输出位位置。



程序段 5：写入完成后置位 DONE 历史位。一旦发生错误，该程序段将置位错误历史位并保存状态代码。

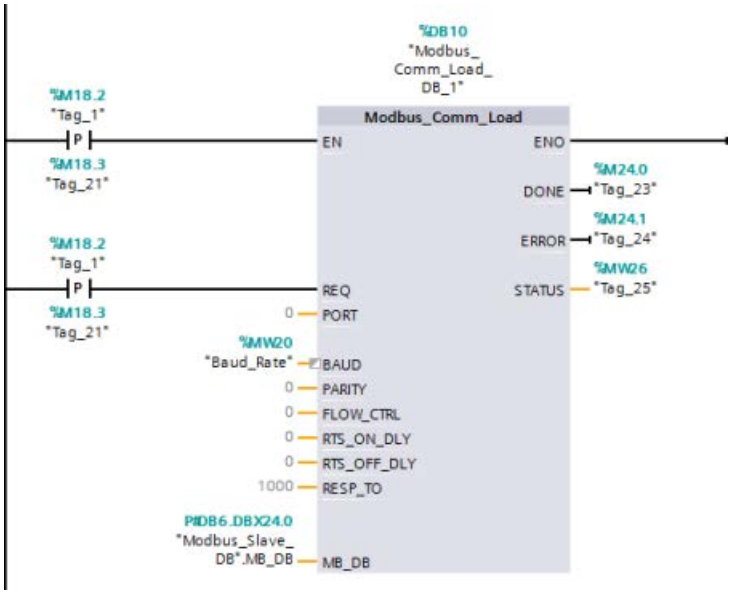


示例： Modbus RTU 从站程序

每次启用“Tag_1”启用时，初始化下面显示的 MB_COMM_LOAD。

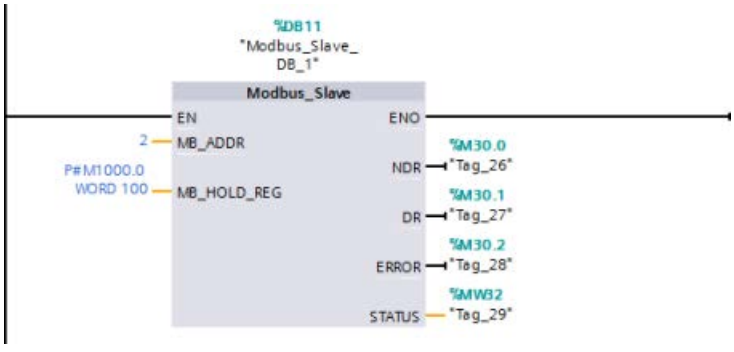
通过此方式执行 MB_COMM_LOAD 时，必须保证串口组态在运行时会根据 HMI 配置进行更改。

程序段 1： 每次 HMI 设备更改 RS485 模块参数时，都会初始化该参数。



下面显示的 MB_SLAVE 置于每 10 ms 执行一次的循环 OB 中。尽管这样不会使从站的绝对响应速度达到最快，但却可使短消息（在请求中占 20 字节或更低）达到 9600 波特的良好性能。

程序段 2： 每次扫描期间检查 Modbus 主站请求。 Modbus 保持寄存器被组态为 100 个字（从 MW1000 开始）。



13.6 早期 PtP 通信（仅限 CM/CB 1241）

在 STEP 7 V13 SP1 和 S7-1200 V4.1 CPU 之前的版本中，点对点通信指令使用不同的名称，而且在部分接口上也略有不同。 点对点通信 (页 1115)、端口 (页 1119)和参数组态 (页 1135)的一般概念均适用于这两个常规指令集。 相关的编程信息，请参见具体的早期点对点指令。

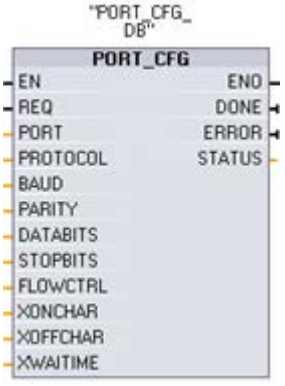
表格 13-97 常见的错误类别

类别说明	错误类别	说明
端口组态	80Ax	用于定义常见端口组态错误
传送组态	80Bx	用于定义常见传送组态错误
接收组态	80Cx	用于定义常见接收组态错误
传送运行时	80Dx	用于定义常见传送运行时错误
接收运行时	80Ex	用于定义常见接收运行时错误
信号处理	80Fx	用于定义与所有信号处理相关的常见错误

13.6.1 早期点对点指令

13.6.1.1 PORT_CFG（动态组态通信参数）

表格 13-98 PORT_CFG（端口组态）指令

LAD/FBD	SCL	说明
	"PORT_CFG_DB"(REQ:=_bool_in_, PORT:=_uint_in_, PROTOCOL:=_uint_in_, BAUD:=_uint_in_, PARITY:=_uint_in_, DATABITS:=_uint_in_, STOPBITS:=_uint_in_, FLOWCTRL:=_uint_in_, XONCHAR:=_char_in_, XOFFCHAR:=_char_in_, WAITTIME:=_uint_in_, DONE=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_);	PORT_CFG 可用于通过用户程序更改端口参数，如波特率等参数。 可以在设备配置属性中设置端口的初始静态组态，或者仅使用默认值。可以在用户程序中执行 PORT_CFG 指令来更改组态。

[1] STEP 7 会在插入指令时自动创建 DB。

PORT_CFG 组态更改不会永久存储在 CPU 中。CPU 从 RUN 模式切换到 STOP 模式和循环上电后将恢复设备配置中组态的参数。更多信息，请参见组态通信端口 (页 1119)和管理流控制 (页 1121)。

表格 13-99 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	在该输入的上升沿激活组态更改。（默认值：False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。（默认值：0）
PROTOCOL	IN	UInt	0 - 点对点通信协议（默认值） 1..n - 用于在将来定义特定的协议
BAUD	IN	UInt	端口波特率（默认值：6)： 1 = 300 波特，2 = 600 波特，3 = 1200 波特，4 = 2400 波特， 5 = 4800 波特，6 = 9600 波特，7 = 19200 波特，8 = 38400 波特， 9 = 57600 波特，10 = 76800 波特，11 = 115200 波特
PARITY	IN	UInt	端口奇偶校验（默认值：1)： 1 = 无奇偶校验，2 = 偶校验，3 = 奇校验，4 = 传号校验，5 = 空号校验
DATABITS	IN	UInt	每个字符的位数（默认值：1）： 1 = 8 个数据位、2 = 7 个数据位
STOPBITS	IN	UInt	停止位（默认值：1)： 1 = 1 个停止位，2 = 2 个停止位
FLOWCTRL	IN	UInt	流控制（默认值：1)： 1 = 无流控制，2 = XON/XOFF，3 = 硬件 RTS 始终激活，4 = 硬件 RTS 切换
XONCHAR	IN	Char	指定用作 XON 字符的字符。这通常是 DC1 字符 (16#11)。只有启用流控制时，才会评估该参数。（默认值：16#11）
XOFFCHAR	IN	Char	指定用作 XOFF 字符的字符。这通常是 DC3 字符 (116#3)。只有启用流控制时，才会评估该参数。（默认值：16#13）
XWAITIME	IN	UInt	指定在接收 XOFF 字符后等待 XON 字符的时间，或者指定在启用 RTS 后等待 CTS 信号的时间（0 到 65535 ms）。只有启用流控制时，才会评估该参数。（默认值：2000）2000）
DONE	OUT	Bool	上一请求已完成且没有出错后，保持为 TRUE 一个执行周期时间

参数和类型		数据类型	说明
ERROR	OUT	Bool	上一请求已完成但出现错误后，保持为 TRUE 一个执行周期时间
STATUS	OUT	Word	执行条件代码（默认值：0）0）

表格 13-100 条件代码

STATUS (W#16#....)	说明
80A0	特定协议不存在。
80A1	特定波特率不存在。
80A2	特定奇偶校验选项不存在。
80A3	特定数据位数不存在。
80A4	特定停止位数不存在。
80A5	特定流控制类型不存在。
80A6	等待时间为 0 且流控制启用
80A7	XON 和 XOFF 是非法值（例如，同一个值）

13.6.1.2 SEND_CFG（动态组态串行传输参数）

表格 13-101 SEND_CFG（发送组态）指令

LAD/FBD	SCL	说明

	`"SEND_CFG_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_uint_in_,`
	`RTSONDLY:=_uint_in_,`
	`RTSOFFDLY:=_uint_in_,`
	`BREAK:=_uint_in_,`
	`IDLELINE:=_uint_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		SEND_CFG 可用于动态组态 PtP 通信端口的串行传输参数。执行 SEND_CFG 时，将放弃 CM 或 CB 内所有排队的消息。

[1] STEP 7 会在插入指令时自动创建 DB。

可以在设备配置属性中设置端口的初始静态组态，或者仅使用默认值。可以在用户程序中执行 SEND_CFG 指令来更改组态。

SEND_CFG 组态更改不会永久存储在 CPU 中。CPU 从 RUN 模式切换到 STOP 模式和循环上电后将恢复设备配置中组态的参数。请参见组态传送（发送）参数。

表格 13-102 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	在该输入的上升沿激活组态更改。（默认值：False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。（默认值：0）
RTSONDLY	IN	UInt	启用 RTS 后执行任何 Tx 数据传输前要等待的毫秒数。只有启用硬件流控制时，该参数才有效。有效范围是 0 到 65535 ms。值 0 表示禁用该功能。（默认值：0）
RTSOFFDLY	IN	UInt	执行 Tx 数据传输后禁用 RTS 前要等待的毫秒数：只有启用硬件流控制时，该参数才有效。有效范围是 0 到 65535 ms。值 0 表示禁用该功能。（默认值：0）
BREAK	IN	UInt	该参数指定在各消息开始时将发送指定位时间的中断。最大值是 65535 个位时间，最多为 8 秒。值 0 表示禁用该功能。（默认值：12）
IDLELINE	IN	UInt	该参数指定在各消息开始前线路将保持空闲指定的位时间。最大值是 65535 个位时间，最多为 8 秒。值 0 表示禁用该功能。（默认值：12）
DONE	OUT	Bool	上一请求已完成且没有出错后，保持为 TRUE 一个执行周期时间
ERROR	OUT	Bool	上一请求已完成但出现错误后，保持为 TRUE 一个执行周期时间
STATUS	OUT	Word	执行条件代码（默认值：0）

表格 13-103 条件代码

STATUS (W#16#....)	说明
80B0	不允许传送中断组态。
80B1	中断时间大于允许的最大值。
80B2	空闲时间大于允许的最大值。

13.6.1.3 RCV_CFG（动态组态串行接收参数）

表格 13-104 RCV_CFG（接收组态）指令

LAD/FBD	SCL	说明

	`"RCV_CFG_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_uint_in_,`
	`CONDITIONS:=_struct_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		RCV_CFG 可用于动态组态 PtP 通信端口的串行接收方参数。该指令可组态表示接收消息开始和结束的条件。执行 RCV_CFG 时，将放弃 CM 或 CB 内所有排队的消息。

[1] STEP 7 会在插入指令时自动创建 DB。

可以在设备配置属性中设置通信端口的初始静态组态，或者索性使用默认值。可以在用户程序中执行 RCV_CFG 指令来更改组态。

RCV_CFG 组态更改不会永久存储在 CPU 中。CPU 从 RUN 模式切换到 STOP 模式和循环上电后将恢复设备配置中组态的参数。有关详细信息，请参见组态接收参数 (页 1123)。

表格 13-105 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	在该输入的上升沿激活组态更改。（默认值：False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。（默认值：0）
CONDITIONS	IN	CONDITIONS	如下文所述，条件数据结构指定消息开始和结束条件。
DONE	OUT	Bool	上一请求已完成且没有出错后，保持为 TRUE 一个扫描周期时间
ERROR	OUT	Bool	上一请求已完成但出现错误后，保持为 TRUE 一个扫描周期时间
STATUS	OUT	Word	执行条件代码（默认值：0）

RCV_PTP 指令的开始条件

RCV_PTP 指令使用 RCV_CFG 指令指定的组态来确定点对点通信消息的开始和结束。消息开始由开始条件确定。消息开始可以由一个开始条件或开始条件的组合来确定。如果指定多个开始条件，则只有满足所有条件后才能使消息开始。

有关消息开始条件的说明，请参见主题“组态接收参数 (页 1123)”。

参数 CONDITIONS 数据类型结构的第 1 部分（开始条件）

表格 13-106 START 条件的 CONDITIONS 结构

参数和类型		数据类型	说明
STARTCOND	IN	UInt	指定开始条件（默认值：1） • 01H - 开始字符 • 02H - 任意字符 • 04H - 线路中断 • 08H - 线路空闲 • 10H - 序列 1 • 20H - 序列 2 • 40H - 序列 3 • 80H - 序列 4
IDLETIME	IN	UInt	线路空闲超时所需的位时间数。（默认值：40）。仅与线路空闲条件一起使用。0 到 65535
STARTCHAR	IN	Byte	用于开始字符条件的开始字符。（默认值：B#16#2）
SEQ[1].CTL	IN	Byte	针对每个字符执行的序列 1 忽略/比较控制：（默认值：B#16#0） 它们是为开始序列中各字符启用的位。 • 01H - 字符 1 • 02H - 字符 2 • 04H - 字符 3 • 08H - 字符 4 • 10H - 字符 5 禁用与某个字符关联的位意味着该序列位置中的任意字符均符合条件。
SEQ[1].STR	IN	Char[5]	序列 1 开始字符（5 个字符）。默认值：0
SEQ[2].CTL	IN	Byte	针对每个字符执行的序列 2 忽略/比较控制。（默认值：B#16#0）
SEQ[2].STR	IN	Char[5]	序列 2 开始字符（5 个字符）。默认值：0
SEQ[3].CTL	IN	Byte	针对每个字符执行的序列 3 忽略/比较控制。默认值：B#16#0
SEQ[3].STR	IN	Char[5]	序列 3 开始字符（5 个字符）。默认值：0

参数和类型		数据类型	说明
SEQ[4].CTL	IN	Byte	针对每个字符执行的序列 4 忽略/比较控制。默认值：B#16#0
SEQ[4].STR	IN	Char[5]	序列 4 开始字符（5 个字符），默认值：0

示例

请注意以下所接收的十六进制编码消息：“**68** 10 aa **68** bb 10 aa 16”以及下表中列出的已组态开始序列。在成功接收到第一个 68H 字符时，开始评估开始序列。在成功接收到第四个字符（第二个 68H）时，开始条件 1 得到满足。只要满足了开始条件，就会开始评估结束条件。

开始序列处理会因各种奇偶校验、成帧或字符间时间错误而终止。由于不再满足开始条件，因而这些错误将导致不会有接收消息。

表格 13-107 开始条件

开始条件	第一个字符	第一个字符 +1	第一个字符 +2	第一个字符 +3	第一个字符 +4
1	**68**H	xx	xx	**68**H	xx
2	10H	aaH	xx	xx	xx
3	dcH	aaH	xx	xx	xx
4	e5H	xx	xx	xx	xx

RCV_PTP 指令的结束条件

消息结束由指定的结束条件确定。消息结束由第一次出现的一个或多个已组态结束条件来确定。主题“组态接收参数 (页 1123)”中“消息结束条件”部分介绍了可以在 RCV_CFG 指令中组态的结束条件。

可以在设备配置的通信接口的属性中组态结束条件，或者通过 RCV_CFG 指令组态结束条件。只要 CPU 从 STOP 模式切换到 RUN 模式，接收参数（开始条件和结束条件）就将恢复为设备配置设置。如果 STEP 7 用户程序执行 RCV_CFG，则这些设置将更改为 RCV_CFG 的条件。

参数 CONDITIONS 数据类型结构的第 2 部分（结束条件）

表格 13-108 END 条件的 CONDITIONS 结构

参数	参数类型	数据类型	说明
ENDCOND	IN	UInt 0	该参数指定消息结束条件： • 01H - 响应时间 • 02H - 消息时间 • 04H - 字符间隙 • 08H - 最大长度 • 10H - N + LEN + M • 20H - 序列
MAXLEN	IN	UInt 1	最大消息长度：仅当选择最大长度结束条件时使用。1 到 1024 个字节
N	IN	UInt 0	长度域在消息中的字节位置。仅与 N + LEN + M 结束条件一起使用。1 到 1022 个字节
LENGTHSIZE	IN	UInt 0	长度字段的大小（1、2 或 4 个字节）。仅与 N + LEN + M 结束条件一起使用。
LENGTHM	IN	UInt 0	指定跟在长度域后、不包含在长度域值内的字符数。该参数仅与 N + LEN + M 结束条件一起使用。0 到 255 个字节
RCVTIME	IN	UInt 200	指定接收第一个字符所需的等待时间。如果在指定时间内没有成功接收到字符，接收操作将被终止且包含错误。该参数仅与响应时间条件一起使用。（0 到 65535 个位时间，最多 8 秒） 此参数不是消息结束条件，因为在接收到第一个响应字符时评估即终止。由于在预期有响应时却接收不到响应，因此仅就其能够终止接收方操作而言，它又是一个结束条件。必须选择一个单独的结束条件。
MSGTIME	IN	UInt 200	指定在接收到第一个字符后完成接收整条消息所需的等待时间。只有选择了消息超时条件时，才会使用该参数。（0 到 65535 毫秒）
CHARGAP	IN	UInt 12	指定字符间的位时间数。如果字符间的位时间数超出指定值，则结束条件得到满足。该参数仅与字符间隙条件一起使用。（0 到 65535 个位时间，最多 8 秒）

参数	参数类型	数据类型	说明
SEQ.CTL	IN	Byte B#16#0	针对每个字符执行的序列 1 忽略/比较控制： 它们是为结束序列中各字符启用的位。字符 1 是位 0，字符 2 是位 1，依此类推，字符 5 是位 4。禁用与某个字符关联的位意味着该序列位置中的任意字符均符合条件。
SEQ.STR	IN	Char[5] 0	序列 1 开始字符（5 个字符）

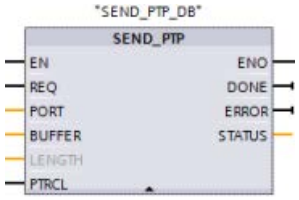
表格 13-109 条件代码

STATUS (W#16#....)	说明
80C0	所选开始条件非法
80C1	所选结束条件非法；未选择结束条件
80C2	启用了接收中断，但不允许此操作。
80C3	启用了最大长度结束条件，最大长度是 0 或大于 1024。
80C4	启用了计算长度，但 N >= 1023。
80C5	启用了计算长度，但长度不是 1、2 或 4。
80C6	启用了计算长度，但 M 值大于 255。
80C7	启用了计算长度，但计算长度大于 1024。
80C8	启用了响应超时，但响应超时为零。
80C9	启用了字符间隙超时，但该字符间隙超时为零。
80CA	启用了线路空闲超时，但该线路空闲超时为零。
80CB	启用了结束序列，但所有字符均“不相关”。
80CC	启用了开始序列（4 个中的任何一个），但所有字符均“不相关”。

13.6.1.4 SEND_PTP（传输发送缓冲区数据）

表格 13-110 SEND_PTP（发送点对点数据）指令

LAD/FBD	SCL	说明

	`"SEND_PTP_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_uint_in_,`
	`BUFFER:=_variant_in_,`
	`LENGTH:=_uint_in_,`
	`PTRCL:=_bool_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		SEND_PTP 用于启动数据传输，并将分配的缓冲区传送到通信接口。在 CM 或 CB 块以指定波特率发送数据的同时，CPU 程序会继续执行。仅一个发送操作可以在某一给定时间处于未决状态。如果在 CM 或 CB 已经开始传送消息时执行第二个 SEND_PTP，CM 或 CB 将返回错误。

[1] STEP 7 会在插入指令时自动创建 DB。

表格 13-111 参数的数据类型

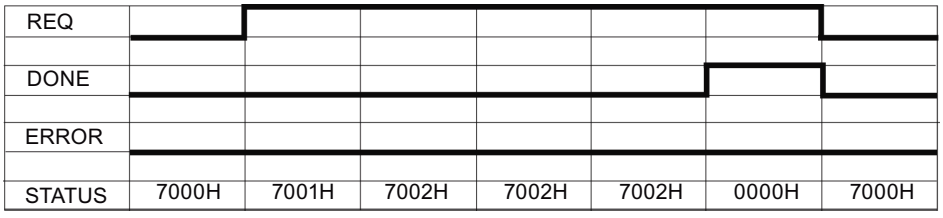
参数和类型		数据类型	说明
REQ	IN	Bool	在该传送使能输入的上升沿激活所请求的传送。这会启动将缓冲区数据传送到点对点通信接口。（默认值：False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。（默认值：0）
BUFFER	IN	Variant	该参数指向传送缓冲区的起始位置。（默认值：0） **注：**不支持布尔数据或布尔数组。
LENGTH [1]	IN	UInt	传输的帧长度（字节）（默认值：0） 传输复杂结构时，始终使用长度 0。
PTRCL	IN	Bool	保留供以后使用
DONE	OUT	Bool	上一请求已完成且没有出错后，保持为 TRUE 一个扫描周期时间
ERROR	OUT	Bool	上一请求已完成但出现错误后，保持为 TRUE 一个扫描周期时间
STATUS	OUT	Word	执行条件代码（默认值：0）

[1] 可选参数：单击 LAD/FBD 框底部的箭头，展开框并包含此参数。

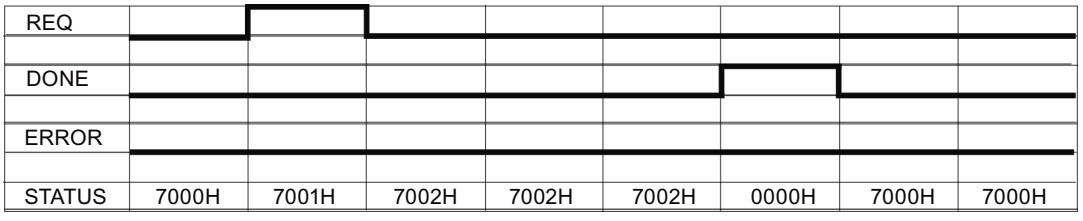
传送操作进行期间，DONE 和 ERROR 输出均为 FALSE。传送操作完成后，DONE 或 ERROR 输出将被设置为 TRUE 以显示传送操作的状态。当 DONE 或 ERROR 为 TRUE 时，STATUS 输出有效。

如果通信接口接受传送数据，则该指令将返回状态值 16#7001。如果 CM 或 CB 仍在忙于传送，则后续的 SEND_PTP 执行将返回 16#7002。传送操作完成后，CM 或 CB 将返回传送操作状态 16#0000（如果未出错）。后续执行 REQ 为低电平的 SEND_PTP 时，将返回状态 16#7000（不忙）。

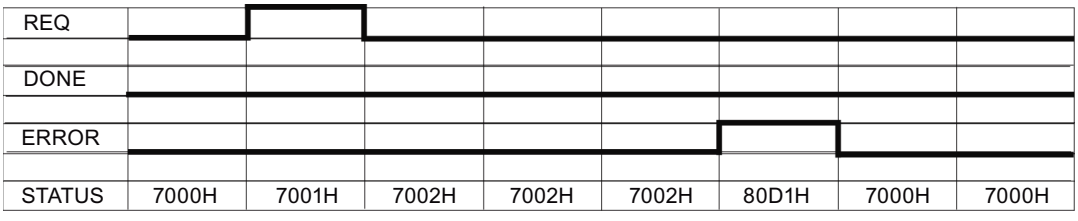
下图显示了输出值与 REQ 的关系。假设定期调用该指令以检查传送过程的状态。在下图中，假设每次扫描都调用该指令（用 STATUS 值表示）。



下图显示通过 REQ 线路脉冲（持续一个扫描周期）启动传送操作时，DONE 和 STATUS 参数是如何仅在一个扫描周期内有效。



下图显示了出错时 DONE、ERROR 和 STATUS 参数之间的关系。



只有 SEND_PTP 再次使用相同的背景 DB 执行后，DONE、ERROR 和 STATUS 值才有效。

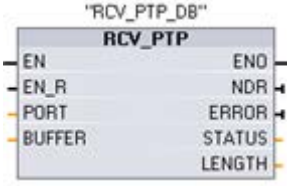
表格 13-112 条件代码

STATUS (W#16#....)	说明
80D0	传送方激活期间发出新请求
80D1	由于在等待时间内没有 CTS 信号，传送中止
80D2	由于没有来自 DCE 设备的 DSR，传送中止
80D3	由于队列溢出（传送 1024 个字节以上），传送中止

STATUS (W#16#....)	说明
80D5	反向偏置信号（断线检测）
833A	BUFFER 参数的 DB 不存在。

13.6.1.5 RCV_PTP（启用消息接收）

表格 13-113 RCV_PTP（接收点对点）指令

LAD/FBD	SCL	说明
	"RCV_PTP_DB"(EN_R:=_bool_in_, PORT:=_uint_in_, BUFFER:=_variant_in_, NDR=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_, LENGTH=>_uint_out_);	RCV_PTP 用于检查 CM 或 CB 中已接收的消息。如果有消息，则会将其从 CM 或 CB 传送到 CPU。如果发生错误，则会返回相应的 STATUS 值。

[1] STEP 7 会在插入指令时自动创建 DB。

表格 13-114 参数的数据类型

参数和类型		数据类型	说明
EN_R	IN	Bool	该输入为 TRUE 并且有消息时，会将消息从 CM 或 CB 传送到 BUFFER。EN_R 为 FALSE 时，将检查 CM 或 CB 是否收到消息并更新 NDR、ERROR 和 STATUS 输出，但不会将消息传送到 BUFFER。（默认值：0）0）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量” (System constants) 选项卡中分配。（默认值：0）0）
BUFFER	IN	Variant	该参数指向接收缓冲区的起始位置。该缓冲区应该足够大，可以接收最大长度消息。 不支持布尔数据或布尔数组。（默认值：0）0）
NDR	OUT	Bool	新数据就绪且操作无错完成后，保持为 TRUE 一个执行周期时间。
ERROR	OUT	Bool	操作已完成但出现错误后，保持为 TRUE 一个执行周期时间。
STATUS	OUT	Word	执行条件代码（默认值：0）0）
LENGTH	OUT	UInt	返回消息的长度（字节）（默认值：0）0）

注意，EN_R 输入与 RCV_PTP 指令的消息缓存区之间的以下关系：

输入 EN_R 控制着是否将接收到的消息复制到 BUFFER。

当 EN_R 输入为 TRUE 并且有消息时，CPU 将消息从 CM 或 CB 传送到 BUFFER 并更新 NDR、ERROR、STATUS,和 LENGTH 输出。

当 EN_R 为 FALSE 时，CPU 将检查 CM 或 CB 是否有消息并更新 NDR、ERROR,和 STATUS 输出，但不会将消息传送到 BUFFER。（注意，EN_R 的默认值为 FALSE。）

建议将 EN_R 设置为 TRUE 并通过 EN 输入控制 RCV_PTP 指令的执行。

NDR 或 ERROR 为 TRUE 时，STATUS 值有效。STATUS 值提供 CM 或 CB 中的接收操作终止的原因。它通常是正值，表示接收操作成功且接收过程正常终止。如果 STATUS 值为负数（十六进制值的最高有效位置位），则表示接收操作因错误条件终止，例如，奇偶校验、组帧或超限错误。

每个 PtP 通信接口最多可缓冲 1024 字节。这可以是一个大消息或几个较小的消息。如果 CM 或 CB 中存在多个消息，则 RCV_PTP 指令将返回最早的可用消息。随后执行 RCV_PTP 指令将返回下一个最早的可用消息。

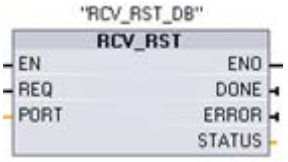
表格 13-115 条件代码

STATUS (W#16#...)	说明
0000	没有提供缓冲区
0094	因接收到最大字符长度，消息被终止
0095	因消息超时，消息被终止
0096	消息因字符间超时而终止
0097	消息因响应超时而终止
0098	因已满足“N+LEN+M”长度条件，消息被终止
0099	因已满足结束序列，消息被终止
80E0	因接收缓冲区已满，消息被终止
80E1	因出现奇偶校验错误，消息被终止
80E2	因组帧错误，消息被终止
80E3	因出现超限错误，消息被终止
80E4	因计算长度超出缓冲区大小，消息被终止
80E5	反向偏置信号（断线检测）
833A	BUFFER 参数的 DB 不存在。

13.6.1.6 RCV_RST（删除接收缓冲区）

表格 13-116 RCV_RST（接收方复位）指令

LAD/FBD	SCL	说明

	`"RCV_RST_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_uint_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		RCV_RST 可清空 CM 或 CB 中的接收缓冲区。

[1] STEP 7 会在插入指令时自动创建 DB。

表格 13-117 参数的数据类型

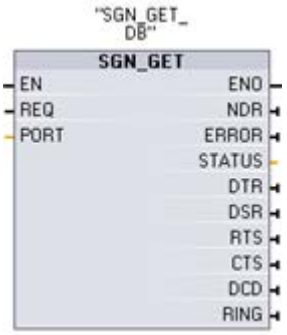
参数和类型		数据类型	说明
REQ	IN	Bool	在该使能输入的上升沿激活接收方重置（默认值：False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量” (System constants) 选项卡中分配。（默认值：0）。0）
DONE	OUT	Bool	在一个扫描周期内为 TRUE 时，表示上一个请求已完成且没有错误。
ERROR	OUT	Bool	为 TRUE 时，表示上一个请求已完成但有错误。此外，该输出为 TRUE 时，STATUS 输出还会包含相关错误代码。
STATUS	OUT	Word	错误代码（默认值：0）0） 有关通信状态代码，请参见点对点指令的公共参数 (页 1135)。

说明

您可能希望使用 RCV_RST 指令以确保在出现通信错误或更改波特率等通信参数后清除消息缓冲区。执行 RCV_RST 会导致模块清除所有内部消息缓冲区。清除消息缓冲区后，可确保程序执行后续接收指令时返回的是新消息，而不是 RCV_RST 调用之前的消息。

13.6.1.7 SGN_GET（查询 RS-232 信号）

表格 13-118 SGN_GET（获取 RS232 信号）指令

LAD/FBD	SCL	说明
	"SGN_GET_DB"(REQ:=_bool_in_, PORT:=_uint_in_, NDR=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_, DTR=>_bool_out_, DSR=>_bool_out_, RTS=>_bool_out_, CTS=>_bool_out_, DCD=>_bool_out_, RING=>_bool_out_);	SGN_GET 用于读取 RS232 通信信号的当前状态。 该功能仅对 RS232 CM 有效。

[1] STEP 7 会在插入指令时自动创建 DB。

表格 13-119 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	在该输入的上升沿获取 RS232 信号状态值（默认值：False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。
NDR	OUT	Bool	新数据就绪且操作无错误地完成时，在一个扫描周期内为 TRUE
ERROR	OUT	Bool	操作已完成但出现错误后，保持为 TRUE 一个扫描周期时间
STATUS	OUT	Word	执行条件代码（默认值：0）0）
DTR	OUT	Bool	数据终端就绪，模块就绪（输出）。默认值：False
DSR	OUT	Bool	数据设备就绪，通信伙伴就绪（输入）。默认值：False
RTS	OUT	Bool	请求发送，模块已做好发送准备（输出）。默认值：False
CTS	OUT	Bool	允许发送，通信伙伴可以接收数据（输入）。默认值：False
DCD	OUT	Bool	数据载波检测，接收信号电平（始终为 False，不支持）
RING	OUT	Bool	响铃指示器，来电指示（始终为 False，不支持）

表格 13-120 条件代码

STATUS (W#16#....)	说明
80F0	CM 或 CB 是 RS485 且没有信号可用

13.6.1.8 SGN_SET（设置 RS-232 信号）

表格 13-121 SGN_SET（设置 RS232 信号）指令

LAD/FBD	SCL	说明
	"SGN_SET_DB"(REQ:=_bool_in_, PORT:=_uint_in_, SIGNAL:=_byte_in_, RTS:=_bool_in_, DTR:=_bool_in_, DSR:=_bool_in_, DONE=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_);	SGN_SET 用于设置 RS232 通信信号的状态。 该功能仅对 RS232 CM 有效。

[1] STEP 7 会在插入指令时自动创建 DB。

表格 13-122 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	在该输入的上升沿启动设置 RS232 信号的操作（默认值：False）
PORT	IN	PORT	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。（默认值：0）
SIGNAL	IN	Byte	选择要设置哪些信号：（允许选择多个）。默认值：0 • 01H = 设置 RTS • 02H = 设置 DTR • 04H = 设置 DSR
RTS	IN	Bool	请求发送，模块准备好将值发送到设备（真或假），默认值：False
DTR	IN	Bool	数据终端就绪，模块准备好将值发送到设备（真或假）。默认值：False

参数和类型		数据类型	说明
DSR	IN	Bool	数据设备就绪（仅适用于 DCE 型接口），不使用。
DONE	OUT	Bool	上一请求已完成且没有出错后，保持为 TRUE 一个执行周期时间
ERROR	OUT	Bool	上一请求已完成但出现错误后，保持为 TRUE 一个执行周期时间
STATUS	OUT	Word	执行条件代码（默认值：0）0）

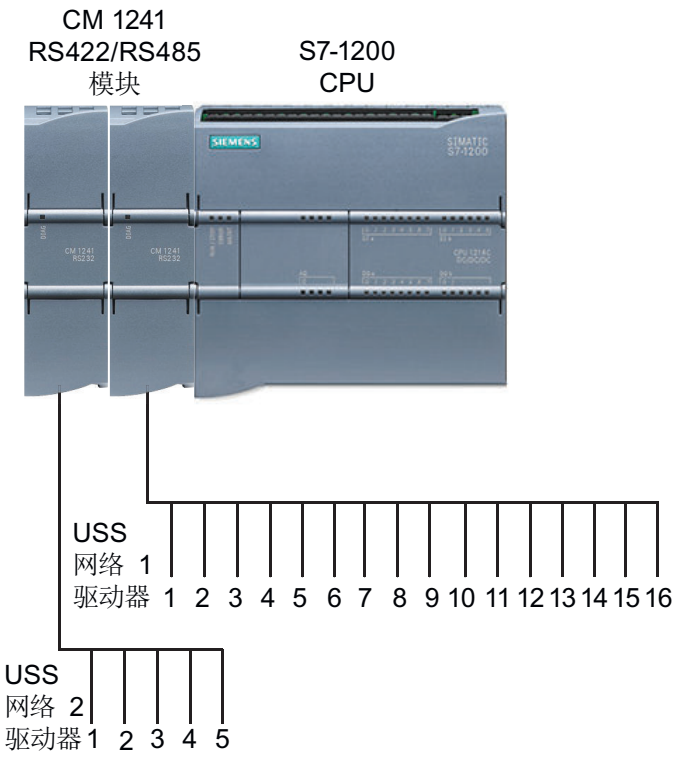
表格 13-123 条件代码

STATUS (W#16#....)	说明
80F0	CM 或 CB 是 RS485 且无法设置任何信号
80F1	因硬件流控制的原因而无法设置信号
80F2	因模块是 DTE 而无法设置 DSR
80F3	因模块是 DCE 而无法设置 DTR

13.7 早期 USS 通信（仅 CM/CB 1241）

USS 指令可控制支持通用串行接口 (USS) 的电机驱动器的运行。 可以使用 USS 指令通过与 CM 1241 RS485 通信模块或 CB 1241 RS485 通信板的 RS485 连接与多个驱动器通信。 一个 S7-1200 CPU 中最多可安装三个 CM 1241 RS422/RS485 模块和一个 CB 1241 RS485 板。 每个 RS485 端口最多操作十六台驱动器。

USS 协议使用主从网络通过串行总线进行通信。主站使用地址参数向所选从站发送消息。 如果未收到传送请求，从站本身不会执行传送操作。 各从站之间无法进行直接消息传送。 USS 通信以半双工模式执行。 以下 USS 图示显示了一个驱动器应用示例的网络图。



在 STEP 7 V13 SP1 和 S7-1200 V4.1 CPU 之前的版本中，USS 通信指令以不同的名称存在，在某些情况下，接口也略有不同。 一般概念适用于两个指令集。 关于编程信息，请参见各个早期 USS 指令。

13.7.1 选择 USS 指令的版本

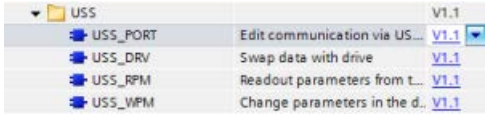
在 STEP 7 中可使用两个版本的 USS 指令：

- 版本 2.0 最初在 STEP 7 Basic/Professional V13 中提供。
- 版本 2.1 在 STEP 7 Basic/Professional V13 SP1 或新版本中提供。

可以从兼容性和移植便利性方面考虑，选择将相应的指令版本插入用户程序中。

不能将两个版本的指令用于同一模块，但不同的模块可以使用不同版本的指令。

单击指令树任务卡上的图标可启用指令树的标题和列。



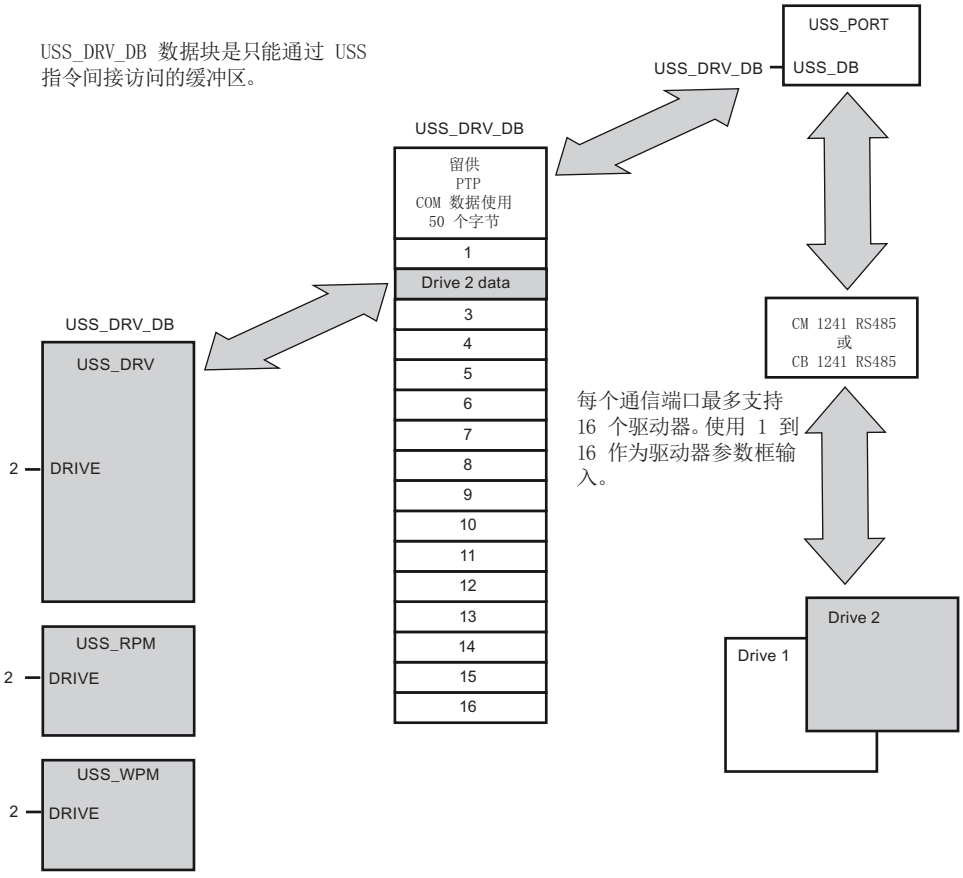
要更改 USS 指令的版本，请从下拉列表中选择相应版本。可以选择一组指令或分别选择各个指令。

使用指令树将 USS 指令放入程序时，将根据所选的 USS 指令在项目树中创建新的 FB 或 FC 实例。在项目树的“PLC_x > 程序块 > 系统块 > 程序资源”(PLC_x > Program blocks > System blocks > Program resources) 下可看到新的 FB 或 FC 实例。

要确认程序中 USS 指令的版本，必须检查项目树的属性而不是程序编辑器中显示的框的属性。选择项目树的 USS FB 或 FC 实例，单击右键，选择 “属性”(Properties)，然后选择 “信息” (Information) 页查看 USS 指令的版本号。

13.7.2 使用 USS 协议的要求

四条 USS 指令使用 1 个 FB 和 3 个 FC 来支持 USS 协议。 一个 USS 网络使用一个 USS_PORT 背景数据块 (DB)。 USS_PORT 背景数据块包含供该 USS 网络中所有驱动器使用的临时存储区和缓冲区。 各 USS 指令共享此数据块中的信息。



连接到一个 RS485 端口的所有驱动器（最多 16 个）是同一 USS 网络的一部分。 连接到另一 RS485 端口的所有驱动器是另一 USS 网络的一部分。 各 USS 网络通过单独的数据块进行管理。 与各 USS 网络相关的所有指令必须共享该数据块。 这包括用于控制各 USS 网络上的所有控制器的所有 USS_DRV、USS_PORT、USS_RPM 和 USS_WPM 指令。

USS_DRV 指令是函数块 (FB)。 在程序编辑器中放置 USS_DRV 指令时，系统将通过“调用选项”(Call options) 对话框提示您为该 FB 分配 DB。 如果对于该 USS 网络而言，它是该程序中的第一条 USS_DRV 指令，则可以接受默认的 DB 分配（或根据需要更改名称），将相应地创建一个新 DB。 但是，如果对于该通道它不是第一条 USS_DRV 指令，则必须使用“调用选项”(Call options) 对话框中的下拉列表选择先前为该 USS 网络分配的 DB 名称。

指令 USS_PORT、USS_RPM 和 USS_WPM 全部都是函数 (FC)。 在编辑器中放置这些 FC 时不分配 DB。 而您必须给这些指令的“USS_DB”输入分配合适的 DB 引用。 双击该参数字段，然后单击参数助手图标可查看可用的 DB 名称。

USS_PORT 函数通过点对点 (PtP) RS485 通信端口处理 CPU 和驱动器之间的实际通信。 每次调用此功能可处理与一个驱动器的一次通信。 用户程序必须尽快调用此功能以防止与驱动器通信超时。 可在主程序循环 OB 或任何中断 OB 中调用此函数。

通常，应在循环中断 OB 中调用 USS_PORT 函数。该循环中断 OB 的循环时间应设置为最小调用间隔的一半左右（例如，1200 波特的通信应使用 350 ms 或更短的循环时间）。

用户程序通过 USS_DRV 函数块可访问 USS 网络上指定的驱动器。 其输入和输出是驱动器的状态和控制。 如果网络上有 16 个驱动器，则用户程序必须具有至少 16 个 USS_DRV 调用，每个驱动器一个调用。 应该以控制驱动器工作所需的速率调用这些块。

只能在主程序循环 OB 中调用 USS_DRV 函数块。

⚠ 小心

从 OB 调用 USS 指令时的考虑事项

只能在主程序循环 OB 中调用 USS_DRV、USS_RPM 和 USS_WPM。可在任何 OB 中调用 USS_PORT 函数，通常是在循环中断 OB 中调用。

不要在优先级比 USS_PORT 指令所在 OB 的优先级高的 OB 中使用指令 USS_DRV、USS_RPM 或 USS_WPM。 例如，不要将 USS_PORT 放置在主程序循环 OB 中，而将 USS_RPM 放置在循环中断 OB 中。如果未能防止 USS_PORT 执行的中断，则会产生意外错误，进而导致人身伤害。

USS_RPM 和 USS_WPM 功能可读取和写入远程驱动器工作参数。 这些参数控制驱动器的内部运行。 有关这些参数的定义，请参见驱动器手册。 用户程序可包含尽可能多的这些功能，但在任何特定时刻，每个驱动器只能激活一个读或写请求。 只能在主程序循环 OB 中调用 USS_RPM 和 USS_WPM 函数。

计算与驱动器通信所需的时间

与驱动器进行的通信与 S7-1200 扫描周期不同步。 在完成一个驱动器通信事务之前，S7-1200 通常完成了多个扫描。

USS_PORT 间隔是一个驱动器事务所需的时间。 下表列出了各个通信波特率下的最小 USS_PORT 时间间隔。 比 USS_PORT 间隔更频繁地调用 USS_PORT 功能不会增加事务数。 如果通信错误导致尝试 3 次才能完成事务，则驱动器超时间隔是处理该事务可能花费的时间。 默认情况下，USS 协议库对每个事务最多自动进行 2 次重试。

表格 13-124 计算时间要求

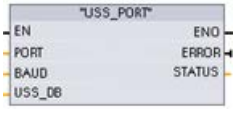
波特率	计算的最小 USS_PORT 调用间隔（毫秒）	每个驱动器的驱动器消息间隔超时（毫秒）
1200	790	2370
2400	405	1215
4800	212.5	638
9600	116.3	349
19200	68.2	205
38400	44.1	133
57600	36.1	109
115200	28.1	85

13.7.3 早期 USS 指令

13.7.3.1 USS_PORT（使用 USS 网络编辑通信）指令

表格 13-125 USS_PORT 指令

LAD/FBD	SCL	说明

	`USS_PORT(`
	`PORT:=_uint_in_,`
	`BAUD:=_dint_in_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`USS_DB:=_fbtref_inout_);`
		USS_PORT 指令用于处理 USS 网络上的通信。

表格 13-126 参数的数据类型

参数和类型		数据类型	说明
PORT	IN	Port	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。
BAUD	IN	DInt	用于 USS 通信的波特率。
USS_DB	INOUT	USS_BASE	将 USS_DRV 指令放入程序时创建并初始化的背景数据块的名称。
ERROR	OUT	Bool	该输出为真时，表示发生错误，且 STATUS 输出有效。
STATUS	OUT	Word	请求的状态值指示扫描或初始化的结果。对于有些状态代码，还在“USS_Extended_Error”变量中提供了更多信息。

通常，程序中每个 PtP 通信端口只一个 USS_PORT 指令，且每次调用该功能都会处理与单个驱动器的通信。与同一个 USS 网络和 PtP 通信端口相关的所有 USS 功能都必须使用同一个背景数据块。

用户程序执行 USS_PORT 指令的次数必须足够多，以防止驱动器超时。通常从循环中断 OB 调用 USS_PORT 以防止驱动器超时并确保 USS_DRV 调用可使用最新的 USS 数据更新内容。

13.7.3.2 USS_DRV（与驱动器交换数据）指令

表格 13-127 USS_DRV 指令

LAD/FBD	SCL	说明
默认视图  展开后的视图  	"USS_DRV_DB"(RUN:=_bool_in_, OFF2:=_bool_in_, OFF3:=_bool_in_, F_ACK:=_bool_in_, DIR:=_bool_in_, DRIVE:=_usint_in_, PZD_LEN:=_usint_in_, SPEED_SP:=_real_in_, CTRL3:=_word_in_, CTRL4:=_word_in_, CTRL5:=_word_in_, CTRL6:=_word_in_, CTRL7:=_word_in_, CTRL8:=_word_in_, NDR=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_, RUN_EN=>_bool_out_, D_DIR=>_bool_out_, INHIBIT=>_bool_out_, FAULT=>_bool_out_, SPEED=>_real_out_, STATUS1=>_word_out_, STATUS3=>_word_out_, STATUS4=>_word_out_, STATUS5=>_word_out_, STATUS6=>_word_out_, STATUS7=>_word_out_, STATUS8=>_word_out_);	USS_DRV 指令通过创建请求消息和解释驱动器响应消息与驱动器交换数据。每个驱动器应使用一个单独的函数块，但与一个 USS 网络和 PtP 通信端口相关的所有 USS 函数必须使用同一个背景数据块。必须在放置第一个 USS_DRV 指令时创建 DB 名称，然后引用初次指令使用时创建的 DB。 STEP 7 会在插入指令时自动创建该 DB。

[1] LAD 和 FBD：通过单击功能框的底部，可展开该功能框，以显示所有参数。灰显的参数引脚可选，不需要进行参数分配。

表格 13-128 参数的数据类型

参数和类型		数据类型	说明
RUN	IN	Bool	驱动器起始位：该输入为真时，将使驱动器以预设速度运行。如果在驱动器运行时 RUN 变为假，电机将减速直至停止。这种行为不同于切断电源 (OFF2) 或对电机进行制动 (OFF3)。
OFF2	IN	Bool	电气停止位：该位为假时，将使驱动器在无制动的情况下自然停止。

参数和类型		数据类型	说明
OFF3	IN	Bool	快速停止位：该位为假时，将通过制动的方式使驱动器快速停止，而不只是使驱动器逐渐自然停止。
F_ACK	IN	Bool	故障确认位：设置该位以复位驱动器上的故障位。清除故障后会设置该位，以告知驱动器不再需要指示前一个故障。
DIR	IN	Bool	驱动器方向控制：设置该位以指示方向为向前（对于正 SPEED_SP）。
DRIVE	IN	USInt	驱动器地址：该输入是 USS 驱动器的地址。有效范围是驱动器 1 到驱动器 16。
PZD_LEN	IN	USInt	字长度：这是 PZD 数据的字数。有效值为 2、4、6 或 8 个字。默认值为 2。
SPEED_SP	IN	Real	速度设定值：这是以组态频率的百分比表示的驱动器速度。正值表示方向向前（DIR 为真时）。有效范围是 200.00 到 -200.00。
CTRL3	IN	Word	控制字 3：写入驱动器上用户可组态参数的值。必须在驱动器上组态该参数。（可选参数）
CTRL4	IN	Word	控制字 4：写入驱动器上用户可组态参数的值。必须在驱动器上组态该参数。（可选参数）
CTRL5	IN	Word	控制字 5：写入驱动器上用户可组态参数的值。必须在驱动器上组态该参数。（可选参数）
CTRL6	IN	Word	控制字 6：写入驱动器上用户可组态参数的值。必须在驱动器上组态该参数。（可选参数）
CTRL7	IN	Word	控制字 7：写入驱动器上用户可组态参数的值。必须在驱动器上组态该参数。（可选参数）
CTRL8	IN	Word	控制字 8：写入驱动器上用户可组态参数的值。必须在驱动器上组态该参数。（可选参数）
NDR	OUT	Bool	新数据就绪：该位为真时，表示输出包含新通信请求数据。
ERROR	OUT	Bool	出现错误：此参数为真时，表示发生错误，STATUS 输出有效。其它所有输出在出错时均设置为零。仅在 USS_PORT 指令的 ERROR 和 STATUS 输出中报告通信错误。
STATUS	OUT	Word	请求的状态值指示扫描的结果。这不是从驱动器返回的状态字。
RUN_EN	OUT	Bool	运行已启用：该位指示驱动器是否在运行。
D_DIR	OUT	Bool	驱动器方向：该位指示驱动器是否正在向前运行。
INHIBIT	OUT	Bool	驱动器已禁止：该位指示驱动器上禁止位的状态。
FAULT	OUT	Bool	驱动器故障：该位指示驱动器已注册故障。用户必须解决问题，并且在该位被置位时，设置 F_ACK 位以清除此位。

参数和类型		数据类型	说明
SPEED	OUT	Real	驱动器当前速度（驱动器状态字 2 的标定值）：以组态速度百分数形式表示的驱动器速度值。
STATUS1	OUT	Word	驱动器状态字 1：该值包含驱动器的固定状态位。
STATUS3	OUT	Word	驱动器状态字 3：该值包含驱动器上用户可组态的状态字。
STATUS4	OUT	Word	驱动器状态字 4：该值包含驱动器上用户可组态的状态字。
STATUS5	OUT	Word	驱动器状态字 5：该值包含驱动器上用户可组态的状态字。
STATUS6	OUT	Word	驱动器状态字 6：该值包含驱动器上用户可组态的状态字。
STATUS7	OUT	Word	驱动器状态字 7：该值包含驱动器上用户可组态的状态字。
STATUS8	OUT	Word	驱动器状态字 8：该值包含驱动器上用户可组态的状态字。

首次执行 USS_DRV 时，将在背景数据块中初始化由 USS 地址（参数 DRIVE）指示的驱动器。完成初始化后，随后执行 USS_PORT 即可开始与具有此驱动器编号的驱动器通信。

更改驱动器编号操作将要求 CPU 从 STOP 模式切换到 RUN 模式以初始化相应的背景数据块。将输入参数组态到 USS TX 消息缓冲区中，并从“前一个”有效响应缓冲区（如果存在）读取输出。USS_DRV 执行期间不进行数据传送。驱动器在 USS_PORT 执行时通信。USS_DRV 仅组态要发送的消息并解释已从前一个请求中接收的数据。

用户可以使用 DIR 输入 (Bool) 或使用符号（正或负）和 SPEED_SP 输入 (Real) 控制驱动器旋转方向。下表假定电机按正向旋转接线，说明这些输入如何一起决定驱动器旋转方向。

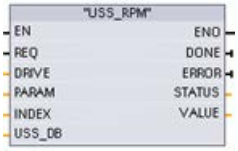
表格 13-129 SPEED_SP 和 DIR 参数的交互作用

SPEED_SP	DIR	驱动器旋转方向
数值 > 0	0	反转
数值 > 0	1	正转
数值 < 0	0	正转
数值 < 0	1	反转

13.7.3.3 USS_RPM（从驱动器读取参数）指令

表格 13-130 USS_RPM 指令

LAD/FBD	SCL	说明

	`USS_RPM(REQ:=_bool_in_,`
	`DRIVE:=_usint_in_,`
	`PARAM:=_uint_in_,`
	`INDEX:=_uint_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`VALUE=>_variant_out_,`
	`USS_DB:=_fbtref_inout_);`
		USS_RPM 指令用于从驱动器读取参数。与同一个 USS 网络和 PtP 通信端口相关的所有 USS 功能都必须使用同一个数据块。必须从主程序循环 OB 调用 USS_RPM。

表格 13-131 参数的数据类型

参数类型		数据类型	说明
REQ	IN	Bool	发送请求：REQ 为真时，表示需要新的读请求。如果该参数的请求已处于待决状态，将忽略新请求。
DRIVE	IN	USInt	驱动器地址：DRIVE 是 USS 驱动器的地址。有效范围是驱动器 1 到驱动器 16。
PARAM	IN	UInt	参数编号：PARAM 指示要写入的驱动器参数。该参数的范围为 0 到 2047。在部分驱动器上，最重要的字节可以访问大于 2047 的 PARAM 值。有关如何访问扩展范围的详细信息，请参见驱动器手册。
INDEX	IN	UInt	参数索引：INDEX 指示要写入的驱动器参数索引。索引为一个 16 位值，其中最低有效字节是实际索引值，其范围是 0 到 255。最高有效字节也可供驱动器使用，且取决于具体的驱动器。有关详细信息，请参见驱动器手册。
USS_DB	INOUT	USS_BASE	将 USS_DRV 指令放入程序时创建并初始化的背景数据块的名称。
VALUE	IN	Word, Int, UInt, DWord, DInt, UDInt, Real	这是已读取的参数的值，仅当 DONE 位为真时才有效。
DONE[1]	OUT	Bool	该参数为真时，表示 VALUE 输出包含先前请求的读取参数值。USS_DRV 发现来自驱动器的读响应数据时会设置该位。满足以下条件之一时复位该位：用户通过另一个 USS_WPM 轮询请求响应数据，或在执行接下来两个 USS_DRV 调用的第二个时请求

参数类型		数据类型	说明
ERROR	OUT	Bool	出现错误：ERROR 为真时，表示发生错误，并且 STATUS 输出有效。其它所有输出在出错时均设置为零。仅在 USS_PORT 指令的 ERROR 和 STATUS 输出中报告通信错误。
STATUS	OUT	Word	STATUS 表示读请求的结果。对于有些状态代码，还在“USS_Extended_Error”变量中提供了更多信息。

[1] DONE 位表示已从参考电机驱动器读取有效数据并已将其传送给 CPU。它不表示 USS 库能够立即读取另一参数。必须将空的 PKW 请求发送到电机驱动器并由指令确认，才能使用特定驱动器的参数通道。立即调用指定电机驱动器的 USS_RPM 或 USS_WPM FC 将导致 0x818A 错误。

13.7.3.4 USS_WPM（更改驱动器中的参数）指令

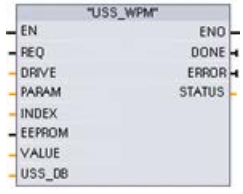
说明

EEPROM 写操作（用于 USS 驱动器内部的 EEPROM）

请勿过多使用 EEPROM 永久写操作。请尽可能减少 EEPROM 写操作次数以延长 EEPROM 的寿命。

表格 13-132 USS_WPM 指令

LAD/FBD	SCL	说明

	`USS_WPM(REQ:=_bool_in_,`
	`DRIVE:=_usint_in_,`
	`PARAM:=_uint_in_,`
	`INDEX:=_uint_in_,`
	`EEPROM:=_bool_in_,`
	`VALUE:=_variant_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`USS_DB:=_fbtref_inout_);`
		USS_WPM 指令用于修改驱动器中的参数。与同一个 USS 网络和 PtP 通信端口相关的所有 USS 功能都必须使用同一个数据块。
		必须从主程序循环 OB 中调用 USS_WPM。

表格 13-133 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	发送请求：REQ 为真时，表示需要新的写请求。如果该参数的请求已处于待决状态，将忽略新请求。
DRIVE	IN	USInt	驱动器地址：DRIVE 是 USS 驱动器的地址。有效范围是驱动器 1 到驱动器 16。
PARAM	IN	UInt	参数编号：PARAM 指示要写入的驱动器参数。该参数的范围为 0 到 2047。在部分驱动器上，最重要的字节可以访问大于 2047 的 PARAM 值。有关如何访问扩展范围的详细信息，请参见驱动器手册。
INDEX	IN	UInt	参数索引：INDEX 指示要写入的驱动器参数索引。索引为一个 16 位值，其中最低有效字节是实际索引值，其范围是 0 到 255。最高有效字节也可供驱动器使用，且取决于具体的驱动器。有关详细信息，请参见驱动器手册。
EEPROM	IN	Bool	存储到驱动器 EEPROM：该参数为真时，写驱动器参数事务将存储在驱动器 EEPROM 中。如果为假，则写操作是临时的，在驱动器循环上电后不会保留。
VALUE	IN	Word, Int, UInt, DWord, DInt, UDInt, Real	要写入的参数值。它必须在 REQ 切换时有效。
USS_DB	INOUT	USS_BASE	将 USS_DRV 指令放入程序时创建并初始化的背景数据块的名称。
DONE[1]	OUT	Bool	DONE 为真时，表示输入 VALUE 已写入驱动器。USS_DRV 发现来自驱动器的写响应数据时会设置该位。如果用户通过另一个 USS_WPM 轮询请求响应数据，或在执行接下来两个 USS_DRV 调用的第二个时请求响应数据，则复位该位。
ERROR	OUT	Bool	ERROR 为真时，表示发生错误，并且 STATUS 输出有效。其它所有输出在出错时均设置为零。仅在 USS_PORT 指令的 ERROR 和 STATUS 输出中报告通信错误。
STATUS	OUT	Word	STATUS 表示写请求的结果。对于有些状态代码，还在“USS_Extended_Error”变量中提供了更多信息。

[1] DONE 位表示已从参考电机驱动器读取有效数据并已将其传送给 CPU。它不表示 USS 库能够立即读取另一参数。必须将空的 PKW 请求发送到电机驱动器并由指令确认，才能使用特定驱动器的参数通道。立即调用指定电机驱动器的 USS_RPM 或 USS_WPM FC 将导致 0x818A 错误。

13.7.4 旧 USS 状态码

在 USS 功能的 STATUS 输出端返回 USS 指令状态代码。

表格 13-134 STATUS 代码 [1]

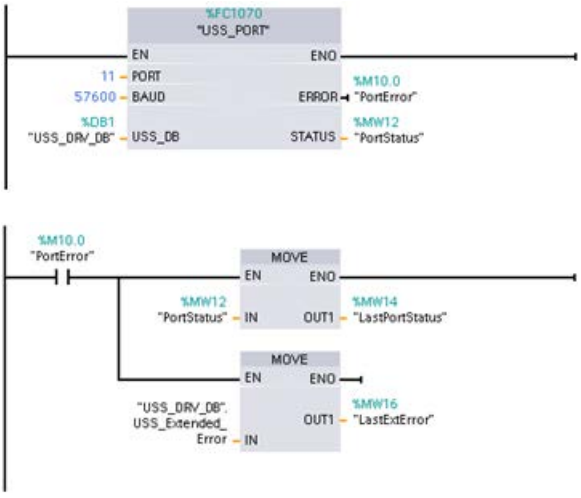
STATUS (W#16#....)	说明
0000	无错误
8180	驱动器响应的长度与从驱动器收到的字符数不匹配。 出错的驱动器编号在“USS_Extended_Error”变量中返回。 请参见本表格下方的扩展错误描述。
8181	VALUE 参数不是 Word、Real 或 DWord 数据类型。
8182	用户提供了 Word 参数值，但从驱动器响应中收到 DWord 或 Real 值。
8183	用户提供了 DWord 或 Real 参数值，但从驱动器响应中收到 Word 值。
8184	驱动器响应报文的校验和有错误。 出错的驱动器编号在“USS_Extended_Error”变量中返回。 请参见本表格下方的扩展错误描述。
8185	非法的驱动器地址（有效驱动器地址范围： 1 到 16）
8186	速度设定值超出有效范围（有效速度 SP 范围： -200% 到 200%）。
8187	对已发送的请求响应了错误的驱动器编号。 出错的驱动器编号在“USS_Extended_Error”变量中返回。 请参见本表格下方的扩展错误描述。
8188	指定的 PZD 字长度非法（有效范围 = 2、4、6 或 8 个字）
8189	指定了非法的波特率。
818A	参数请求通道正在由该驱动器的另一个请求使用。
818B	驱动器尚未对请求和重试做出响应。 出错的驱动器编号在“USS_Extended_Error”变量中返回。 请参见本表格下方的扩展错误描述。
818C	驱动器返回了有关参数请求操作的扩展错误。 请参见本表格下方的扩展错误描述。
818D	驱动器返回了有关参数请求操作的非法访问错误。 有关可能限制参数访问的原因信息，请参见驱动器手册。
818E	驱动器尚未初始化。 若从未调用过该驱动器的 USS_DRV，则该错误代码将返回到 USS_RPM 或 USS_WPM。 这会防止首次扫描 USS_DRV 的初始化过程覆盖未决的参数读/写请求，因为它会将驱动器初始化为新条目。 要修复该错误，请针对此驱动器编号调用 USS_DRV。
80Ax-80Fx	从 USS 库调用的 PtP 通信 FB 返回的特定错误 - 这些错误代码值不会被 USS 库修改且在 PtP 指令说明中定义。

[1] 除了上述列出的 USS 指令错误，还可能返回底层 PtP 通信指令的错误信息。

对于一些 STATUS 代码，在 USS_DRV 背景数据块的“USS_Extended_Error”变量中提供更多信息。 对于 STATUS 代码 8180、8184、8187 和 818B（十六进制），USS_Extended_Error 包含出现通信错误的驱动器编号。 对于 STATUS 代码 818C（十六进制），USS_Extended_Error 包含使用 USS_RPM 或 USS_WPM 指令时从驱动器返回的驱动器错误代码。

示例： 通信错误报告

仅报告有关 USS_PORT 指令（而非 USS_DRV 指令）的通信错误 (STATUS = 16#818B)。 例如，如果没有正确地终止网络，则驱动器可能切换到 RUN 模式，但 USS_DRV 指令将为相关输出参数全部显示 0。 在这种情况下，只能检测有关 USS_PORT 指令的通信错误。 由于该错误仅在一个扫描周期内可见，所以需要添加一些捕获逻辑，如下面的示例所示。 在本例中，当 USS_PORT 指令的错误位为 TRUE 时，STATUS 和 USS_Extended_Error 值将保存到 M 存储器中。 当 STATUS 代码值是十六进制的 8180、8184、8187 或 818B 时，驱动器编号将放在 USS_Extended_Error 变量中。



程序段 1"PortStatus"端口状态和 "USS_DRV_DB".USS_Extended_Error 扩展错误代码值仅在一个程序扫描周期内有效。 必须捕获这些值以便后期处理。

程序段 2“PortError”触点触发将“PortStatus”值存储在“LastPortStatus”中以及将 "USS_DRV_DB".USS_Extended_Error 值存储在“LastExtError”中。

对驱动器内部参数的读写访问

USS 驱动器支持对驱动器的内部参数进行读写访问。 通过该功能可进行驱动器的远程控制和组态。 由于发生类似值超出范围或驱动器当前模式的请求非法等错误，驱动器参数访问操作可能会失败。 驱动器会生成在“USS_Extended_Error”变量中返回的错误代码值。 该错误代码值仅对 USS_RPM 或 USS_WPM 指令的最后一次执行有效。 当 STATUS code 值为十六进制的 818C 时，驱动器错误代码将放入 USS_Extended_Error 变量中。 “USS_Extended_Error”的错误代码值取决于驱动器型号。 有关读写参数操作的扩展错误代码的描述，请参见驱动器手册。

13.7.5 早期 USS 常规驱动器设置要求

早期 USS 常规驱动器设置要求包括以下几点：

- 驱动器必须设置为使用 4 个 PKW 字。
- 驱动器可组态为使用 2、4、6 或 8 个 PZD 字。
- 驱动器中 PZD 字的数量必须与该驱动器的 USS_DRV 指令的 USS_DRV 输入相匹配。
- 所有驱动器的波特率必须与 USS_PORT 指令的 BAUD 输入相匹配。
- 驱动器必须设置为可进行远程控制。
- 驱动器必须设置为使用适合通信链路上 USS 的频率设定值。
- 驱动器地址必须设置为 1 到 16，并且与 USS_DRV 块上对应该驱动器的 DRIVE 输入相匹配。
- 驱动器的方向控制必须设置为使用驱动器设定值的极性。
- 必须正确终止 RS485 网络。

USS 常规驱动器连接和设置对于 USS 指令 (V4.1) 和早期 USS 指令（V4.0 及更早版本）来说是相同的。 详细信息，请参见“示例： USS 常规驱动器连接和设置” (页 1189)。

13.8 早期 Modbus TCP 通信

13.8.1 概述

在 STEP 7 V13 SP1 和 S7-1200 V4.1 CPU 之前的版本中，Modbus TCP 通信指令以不同的名称存在，在某些情况下，接口也略有不同。 一般概念适用于两个指令集。 关于编程信息，请参见各个早期 Modbus TCP 指令。

13.8.2 选择 Modbus TCP 指令的版本

在 STEP 7 中可使用以下版本的 Modbus TCP 指令：

- 历史版本 2.1：兼容所有 CPU 和 CM 版本
- 历史版本 3.1：兼容所有 CPU 和 CM 版本
- 版本 4.2：兼容版本 V4.0 及以上的 CPU 和 V2.1 及以上的 CM
- 版本 5.1：兼容 V4.2 及以上版本的 CPU 和 V2.1 及以上版本的 CM

可以从兼容性和移植便利性方面考虑，选择将相应的指令版本插入用户程序中。

在指令任务卡中，在“通信”(Communication) 组中的“其它”(Others) 下显示 MODBUS TCP 指令。

要更改 Modbus TCP 指令的版本，请从下拉列表中选择相应版本。可以选择一组指令或分别选择各个指令。

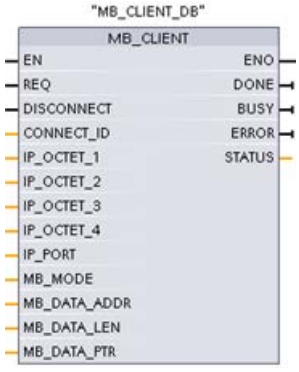
使用指令树将 Modbus TCP 指令放入程序时，将在项目树中创建新的 FB 实例。在项目树的“PLC_x > 程序块 > 系统块 > 程序资源”(PLC_x > Program blocks > System blocks > Program resources) 下可看到新的 FB 实例。

要确认程序中 Modbus TCP 指令的版本，必须检查项目树的属性而不是程序编辑器中显示的框的属性。选择项目树的 Modbus TCP FB 实例，单击右键，选择“属性”(Properties)，然后选择“信息”(Information) 页查看 Modbus TCP 指令的版本号。

13.8.3 早期 Modbus TCP 指令

13.8.3.1 MB_CLIENT（将 PROFINET 用作 Modbus TCP 客户端进行通信）

表格 13-135 MB_CLIENT 指令

LAD/FBD	SCL	说明
	"MB_CLIENT_DB"(REQ:=_bool_in_, DISCONNECT:=_bool_in_, CONNECT_ID=_uint_in_, IP_OCTET_1:=_byte_in_, IP_OCTET_2:=_byte_in_, IP_OCTET_3:=_byte_in_, IP_OCTET_4:=_byte_in_, IP_PORT:=_uint_in_, MB_MODE:=_usint_in_, MB_DATA_ADDR:=_udint_in_, MB_DATA_LEN:=_uint_in_, DONE=>_bool_out_, BUSY=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_, MB_DATA_PTR:=_variant_inout_);	MB_CLIENT 作为 Modbus TCP 客户端，通过 S7-1200 CPU 上的 PROFINET 连接器进行通信。不需要额外的通信硬件模块。 MB_CLIENT 可进行客户端-服务器连接、发送 Modbus 功能请求、接收响应，以及控制 Modbus TCP 服务器的断开。

表格 13-136 参数的数据类型

参数和类型		数据类型	说明
REQ	In	Bool	FALSE = 无 Modbus 通信请求 TRUE = 请求与 Modbus TCP 服务器通信
DISCONNECT	IN	Bool	DISCONNECT 参数允许程序控制与 Modbus 服务器设备的连接和断开。 如果 DISCONNECT = 0 且不存在连接，则 MB_CLIENT 尝试连接到分配的 IP 地址和端口号。 如果 DISCONNECT = 1 且存在连接，则尝试断开连接操作。每当启用此输入时，无法尝试其它操作。
CONNECT_ID	IN	UInt	CONNECT_ID 参数必须唯一标识 PLC 中的每个连接。MB_CLIENT 或 MB_SERVER 指令的各唯一实例必须含有一个唯一的 CONNECT_ID 参数。
IP_OCTET_1	IN	USInt	Modbus TCP 服务器 IP 地址：八位位组 1 Modbus TCP 服务器（客户端将通过 Modbus TCP 协议与其进行连接及通信）的 32 位 IPv4 IP 地址中的 8 位部分。
IP_OCTET_2	IN	USInt	Modbus TCP 服务器 IP 地址：八位位组 2
IP_OCTET_3	IN	USInt	Modbus TCP 服务器 IP 地址：八位位组 3
IP_OCTET_4	IN	USInt	Modbus TCP 服务器 IP 地址：八位位组 4
IP_PORT	IN	UInt	默认值 = 502：服务器（客户端尝试通过 TCP/IP 协议与其连接并最终通信）的 IP 端口号。
MB_MODE	IN	USInt	模式选择：分配请求类型（读、写或诊断）。请参见下面的 Modbus 功能表了解详细信息。
MB_DATA_ADDR	IN	UDInt	Modbus 起始地址：分配 MB_CLIENT 访问的数据的起始地址。请参见下面的 Modbus 功能表了解有效地址信息。
MB_DATA_LEN	IN	UInt	Modbus 数据长度：分配此请求中要访问的位数或字数。请参见下面的 Modbus 功能表了解有效长度信息。
MB_DATA_PTR	IN_OUT	Variant	指向 Modbus 数据寄存器的指针：寄存器缓冲数据进入 Modbus 服务器或来自 Modbus 服务器。指针必须分配一个未进行优化的全局 DB 或 M 存储器地址。
DONE	OUT	Bool	上一请求已完成且没有出错后，DONE 位将保持为 TRUE 一个扫描周期时间。
BUSY	OUT	Bool	• 0 - 无 MB_CLIENT 操作正在进行 • 1 - MB_CLIENT 操作正在进行

参数和类型		数据类型	说明
ERROR	OUT	Bool	MB_CLIENT 执行因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。STATUS 参数中的错误代码值仅在 ERROR = TRUE 的一个循环周期内有效。
STATUS	OUT	Word	执行条件代码

REQ 参数

FALSE = 无 Modbus 通信请求
TRUE = 请求与 Modbus TCP 服务器通信

如果 MB_CLIENT 的实例没有激活且参数 DISCONNECT=0，当 REQ=1 时，将启动一个新的 Modbus 请求。如果尚未建立连接，则建立一个新的连接。

如果在当前请求完成前 DISCONNECT=0 且 REQ=1，从而再次执行 MB_CLIENT 的同一个实例，则不会进行后续 Modbus 传送。但是，一旦完成当前请求，如果通过 REQ=1 执行 MB_CLIENT，可处理新的请求。

完成当前 MB_CLIENT 通信请求后，DONE 位将在一个周期内保持为 TRUE。DONE 位可用作定时门，对多个 MB_CLIENT 请求进行排序。

说明

MB_CLIENT 处理期间输入数据的一致性

Modbus 客户端启动 Modbus 操作后，将在内部保存所有输入状态，然后在每次后续调用时进行比较。比较用于确定此特定调用是否是活动客户端请求的发起者。可使用一个公用背景数据块执行多个 MB_CLIENT 调用。

因此，在主动处理 MB_CLIENT 操作期间应不改变输入，这一点很重要。若不遵循此规则，MB_CLIENT 无法确定其为活动实例。

MB_MODE 和 MB_DATA_ADDR 参数用于选择 Modbus 通信功能

MB_DATA_ADDR 分配要访问数据的起始 Modbus 地址。MB_CLIENT 指令使用 MB_MODE 输入而非功能代码输入。

MB_MODE 和 MB_DATA_ADDR 值一起确定实际 Modbus 消息中使用的功能代码。下表列出了 MB_MODE 参数、Modbus 功能和 Modbus 地址范围之间的对应关系。

表格 13-137 Modbus 功能

MB_MODE	Modbus 功能	数据长度	操作和数据	MB_DATA_ADDR
0	01	1 到 2000	读取输出位： 每个请求 1 到 2000 个位	1 到 9999
0	02	1 到 2000	读取输入位： 每个请求 1 到 2000 个位	10001 到 19999
0	03	1 到 125	读取保持寄存器： 每个请求 1 到 125 个字	40001 到 49999 或 400001 到 465535
0	04	1 到 125	读取输入字： 每个请求 1 到 125 个字	30001 到 39999
1	05	1	写入一个输出位： 每个请求一位	1 到 9999
1	06	1	写入一个保持寄存器： 每个请求 1 个字	40001 到 49999 或 400001 到 465535
1	15	2 到 1968	写入多个输出位： 每个请求 2 到 1968 个位	1 到 9999
1	16	2 到 123	写入多个保持寄存器： 每个请求 2 到 123 个字	40001 到 49999 或 400001 到 465535
2	15	1 到 1968	写入一个或多个输出位： 每个请求 1 到 1968 个位	1 到 9999
2	16	1 到 123	写入一个或多个保持寄存器： 每个请求 1 到 123 个字	40001 到 49999 或 400001 到 465535
11	11	0	读取服务器通信状态字和事件计数器。状态字指示忙闲情况（0 - 不忙，0xFFFF - 忙）。每成功完成一条消息，事件计数器的计数值递增。 对于该功能，MB_CLIENT 的 MB_DATA_ADDR 和 MB_DATA_LEN 参数都将被忽略。	
80	08	1	利用数据诊断代码 0x0000 检查服务器状态（回送测试 - 服务器回送请求） 每个请求 1 个字	

MB_MODE	Modbus 功能	数据长度	操作和数据	MB_DATA_ADDR
81	08	1	利用数据诊断代码 0x000A 重新设置服务器事件计数器 每个请求 1 个字	
3 到 10、12 到 79、82 到 255			保留	

说明

MB_DATA_PTR 分配一个缓冲区来存储从 Modbus TCP 服务器读取或写入到该服务器的数据

数据缓冲区可以是未进行优化的全局 DB 或 M 存储器地址。

对于 M 存储器中的缓冲区，使用标准的 Any 指针格式。具体格式为 P#“位地址” “数据类型” “长度”，例如 P#M1000.0 WORD 500。

MB_DATA_PTR 分配一个通信缓冲区

- MB_CLIENT 通信功能：
  - 从 Modbus 服务器地址（00001 到 09999）读写 1 位数据
  - 从 Modbus 服务器地址（10001 到 19999）读取 1 位数据
  - 从 Modbus 服务器地址（30001 到 39999）和（40001 到 49999）读取 16 位字数据
  - 向 Modbus 服务器地址（40001 到 49999）写入 16 位字数据
- 向/从 MB_DATA_PTR 分配的 DB 或 M 储存器缓冲区传输字或位大小的数据。
- 如果通过 MB_DATA_PTR 分配 DB 为缓冲区，必须为所有 DB 数据元素分配数据类型。
  - 1 位 Bool 数据类型代表一个 Modbus 位地址
  - 16 位单字数据类型（如 WORD、UInt 和 Int）代表一个 Modbus 字地址
  - 32 位双字数据类型（如 DWORD、DInt 和 Real）代表两个 Modbus 字地址

- 可以通过 MB_DATA_PTR 分配复杂的 DB 元素，例如
  - 标准数组
  - 指定的结构，其中每个元素都是唯一的。
  - 指定的复杂结构，其中每个元素都具有唯一的名称以及 16 或 32 位数据类型。
- 不要求 MB_DATA_PTR 数据区位于同一个全局数据块（或 M 存储器区）中。可分配一个数据块供 Modbus 读取，分配另一个数据块供 Modbus 写入，或分配一个数据块用于各个 MB_CLIENT 站。

多个客户端连接

Modbus TCP 客户端支持的并发连接数最多为 PLC 允许的开放式用户通信最大连接数。PLC 的连接总数（包括 Modbus TCP 客户端和服务器）不得超过支持的开放式用户通信最大连接数 (页 807)。可以在客户端和/或服务器类型的连接间共享 Modbus TCP 连接。

单独的客户端连接必须遵循以下规则：

- 每个 MB_CLIENT 连接必须使用一个不同的背景数据块
- 每个 MB_CLIENT 连接必须指定一个唯一的服务器 IP 地址
- 每个 MB_CLIENT 连接必须指定一个唯一的连接 ID
- 是否需要唯一的 IP 端口号取决于服务器组态

连接 ID 对于每个单独的连接必须是唯一的。这意味着单个的唯一连接 ID 只能与每个单独的背景数据块配合使用。总之，背景数据块和连接 ID 成对使用，且对每个连接必须是唯一的。

表格 13-138 MB_CLIENT 实例数据块用户可访问静态变量

变量	数据类型	默认值	说明
Blocked_Proc_Timeout	Real	3.0	在 Modbus 客户端实例受阻后，移除该激活的实例前需等待的时间（秒）。例如，当已发出客户端请求，但应用程序在彻底完成该请求前停止执行该客户端功能时，就会出现这种情况。最大 S7-1200 限值是 55 秒。
MB_Unit_ID	Word	255	Modbus 设备标识符 Modbus TCP 服务器通过其 IP 地址寻址。因此 MB_UNIT_ID 参数不用于 Modbus TCP 寻址。 MB_UNIT_ID 参数与 Modbus RTU 协议中的从站地址相对应。如果 Modbus TCP 服务器用于采用 Modbus RTU 协议的网关，MB_UNIT_ID 可用于标识在串行网络上连接的从站设备。MB_UNIT_ID 将用于将请求转发给正确的 Modbus RTU 从站地址。 某些 Modbus TCP 设备可能需要在受限的值范围内初始化 MB_UNIT_ID 参数。
RCV_TIMEOUT	Real	2.0	MB_CLIENT 等待服务器响应请求的时间（秒）。
已连接	Bool	0	指示与所分配服务器的连接是已接通还是已断开：1 = 接通，0 = 断开

表格 13-139 MB_CLIENT 协议错误

STATUS (W#16#)	发送到 Modbus 客户端的响应代码 (B#16#)	Modbus 协议错误
8381	01	不支持此功能代码
8382	03	数据长度错误
8383	02	数据地址错误或访问的数据超出 MB_HOLD_REG 地址区的界限
8384	03	数据值错误
8385	03	不支持此数据诊断代码值（功能代码 08）

表格 13-140 MB_CLIENT 执行条件代码 [1]

STATUS (W#16#)	MB_CLIENT 参数错误
7001	MB_CLIENT 正在等待 Modbus 服务器响应指定 TCP 端口处的连接或断开连接请求。仅在第一次执行连接或断开操作时才报告此代码。
7002	MB_CLIENT 正在等待 Modbus 服务器响应指定 TCP 端口处的连接或断开连接请求。等待连接或断开操作完成时，将针对任何后续执行报告此代码。
7003	断开操作已成功完成（仅在一个 PLC 扫描周期内有效）。
80C8	服务器在分配的时间内无响应。MB_CLIENT 必须在分配的时间内使用最初传送的事务 ID 接收响应，否则将返回此错误。检查与 Modbus 服务器设备的连接。 尝试过任何组态的重试操作（若适用）后，才报告此错误。
8188	模式值无效
8189	数据地址值无效
818A	数据长度值无效
818B	指向 DATA_PTR 区的指针无效。可以是 MB_DATA_ADDRESS 与 MB_DATA_LEN 的组合。
818C	指向优化的 DATA_PTR 区的指针（必须是未经优化的 DB 区或 M 存储区）
8200	端口正忙于处理现有的 Modbus 请求。
8380	接收到的 Modbus 帧有缺陷或接收到的字节太少。
8387	分配的连接 ID 参数和用于先前请求的 ID 不同。只能有一个单个连接 ID 与每个 MB_CLIENT 背景数据块配合使用。 如果从一个服务器接收到的 Modbus TCP 协议 ID 不是 0，也可作为内部错误使用。
8388	Modbus 服务器返回一些和请求内容不同的数据。这只适用于 Modbus 功能 15 或 16。

[1] 除了上面列出的 MB_CLIENT 错误外，也可以从底层传输块通信指令（TCON、TDISCON、TSEND 和 TRCV (页 862)）返回错误。

13.8.3.2 MB_SERVER（将 PROFINET 用作 Modbus TCP 客户端进行通信）

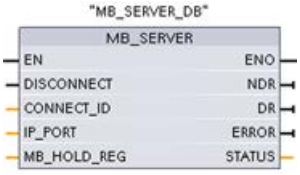
“MB_SERVER”指令作为 Modbus TCP 服务器，通过 S7-1200 CPU 上的 PROFINET 连接器进行通信。“MB_SERVER”指令用于处理 Modbus TCP 客户端的连接请求，接收并处理 Modbus 请求以及发送响应。

使用该指令时，无需使用额外的硬件模块。

注意
安全性信息 请注意，网络中的每个客户端对过程映像输入和输出以及 Modbus 保持寄存器定义的数据块或位存储区域都具有读写访问权限。 可以选择限制对某个 IP 地址的访问，从而阻止未经授权的读写操作。但请注意，共享地址也可用于未经授权的访问。

表格 13-141 MB_SERVER 指令

LAD/FBD	SCL	说明

	`"MB_SERVER_DB"(`
	`DISCONNECT:=_bool_in_,`
	`CONNECT_ID:=_uint_in_,`
	`IP_PORT:=_uint_in_,`
	`NDR=>_bool_out_,`
	`DR=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`MB_HOLD_REG:=_variant_inout_);`
		MB_SERVER 作为 Modbus TCP 服务器，通过 S7-1200 CPU 上的 PROFINET 连接器进行通信。不需要额外的通信硬件模块。
		MB_SERVER 可接收与 Modbus TCP 客户端的连接请求、接收 Modbus 功能请求并发送响应消息。

表格 13-142 参数的数据类型

参数和类型		数据类型	说明
DISCONNECT	IN	Bool	MB_SERVER 尝试与伙伴设备进行“被动”连接。也就是说，服务器被动地侦听来自任何请求 IP 地址的 TCP 连接请求。 如果 DISCONNECT = 0 且不存在连接，则可以启动被动连接。 如果 DISCONNECT = 1 且存在连接，则启动断开操作。这允许程序控制何时接受连接。每当启用此输入时，无法尝试其它操作。
CONNECT_ID	IN	UInt	CONNECT_ID 唯一标识 PLC 中的每个连接。MB_CLIENT 或 MB_SERVER 指令的各唯一实例必须含有一个唯一的 CONNECT_ID 参数。
IP_PORT	IN	UInt	默认值 = 502：用来标识 IP 端口的 IP 端口号，将监视该端口是否有来自 Modbus 客端的连接请求。 以下 TCP 端口号不允许用于 MB_SERVER 被动连接：20、21、25、80、102、123、5001、34962、34963 和 34964。

参数和类型		数据类型	说明
MB_HOLD_REG	IN_OUT	Variant	指向 MB_SERVER Modbus 保持寄存器的指针：保持寄存器必须是一个未经优化的全局 DB 或 M 存储器地址。储存区用于保存值，允许 Modbus 客户端使用 Modbus 寄存器功能 3（读）、6（写）和 16（写）访问这些值。
NDR	OUT	Bool	新数据就绪：0 = 没有新数据，1 = 表示 Modbus 客户端已写入新数据
DR	OUT	Bool	数据读取：0 = 没有读取数据，1 = 表示 Modbus 客户端已读取该数据。
ERROR	OUT	Bool	MB_SERVER 执行因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。STATUS 参数中的错误代码值仅在 ERROR = TRUE 的一个循环周期内有效。
STATUS	OUT	Word	执行条件代码

MB_SERVER 允许进入的 Modbus 功能代码（1、2、4、5 和 15）在 S7-1200 CPU 的输入过程映像及输出过程映像中直接读或写位和字。对于数据传输功能代码（3、6 和 16），MB_HOLD_REG 参数必须定义为大于一个字节的数据类型。下表显示了 Modbus 地址到 CPU 中过程映像的映射。

表格 13-143 Modbus 地址到过程映像的映射

Modbus 功能						S7-1200	
代码	功能	数据区	地址范围			数据区	CPU 地址
01	读位	输出	1	到	8192	输出过程映像	Q0.0 到 Q1023.7
02	读位	输入	10001	到	18192	输入过程映像	I0.0 到 I1023.7
04	读字	输入	30001	到	30512	输入过程映像	IW0 到 IW1022
05	写位	输出	1	到	8192	输出过程映像	Q0.0 到 Q1023.7
15	写位	输出	1	到	8192	输出过程映像	Q0.0 到 Q1023.7

进入的 Modbus 消息功能代码（3、6 和 16）在 Modbus 保持寄存器中读取或写入字，该寄存器可以是 M 存储区地址范围或数据块。保持寄存器的类型由 MB_HOLD_REG 参数指定。

说明

MB_HOLD_REG 参数分配

Modbus 保持寄存器可以位于未经优化的全局 DB 或 M 存储区地址中。

对于 M 存储区地址中的 Modbus 保持寄存器，使用标准的 Any 指针格式。其格式为 P#“位地址”“数据类型”“长度”。例如 P#M1000.0 WORD 500

下表给出了 Modbus 地址到保持寄存器的映射示例，这种映射用于 Modbus 功能代码 03（读取字）、06（写入字）和 16（写入字）。DB 地址的实际上限取决于每种 CPU 型号的最大工作存储器限值和 M 存储器限值。

表格 13-144 Modbus 地址到 CPU 存储器地址的映射示例

Modbus 地址	MB_HOLD_REG 参数示例		
	P#M100.0 Word 5	P#DB10.DBx0.0 Word 5	"Recipe".ingredient
40001	MW100	DB10.DBW0	"Recipe".ingredient[1]
40002	MW102	DB10.DBW2	"Recipe".ingredient[2]
40003	MW104	DB10.DBW4	"Recipe".ingredient[3]
40004	MW106	DB10.DBW6	"Recipe".ingredient[4]
40005	MW108	DB10.DBW8	"Recipe".ingredient[5]

多个服务器连接

可以创建多个服务器连接。这允许单个 PLC 建立与多个 Modbus TCP 客户端的并发连接。

Modbus TCP 服务器支持的并发连接数最多为 PLC 允许的开放式用户通信最大连接数。PLC 的连接总数（包括 Modbus TCP 客户端和服务器）不得超过支持的开放式用户通信最大连接数 (页 807)。可以在客户端和/或服务器类型的连接间共享 Modbus TCP 连接。

单独的服务器连接必须遵循以下规则：

- 每个 MB_SERVER 连接必须使用一个不同的背景数据块。
- 必须通过一个唯一的 IP 端口号建立每个 MB_SERVER 连接。每个端口只能用于 1 个连接。
- 每个 MB_SERVER 连接必须使用一个唯一的连接 ID。
- 必须为每个连接（带有各自的背景数据块）单独调用 MB_SERVER。

连接 ID 对于每个单独的连接必须是唯一的。这意味着单个的唯一连接 ID 只能与每个单独的背景数据块配合使用。总之，背景数据块和连接 ID 成对使用，且对每个连接必须是唯一的。

表格 13-145 Modbus 诊断功能代码

MB_SERVER Modbus 诊断功能		
代码	子功能	说明
08	0x0000	返回查询数据回送测试：MB_SERVER 将向 Modbus 客户端回送接收到的数据字。

MB_SERVER Modbus 诊断功能		
08	0x000A	清除通信事件计数器：MB_SEVER 将清除用于 Modbus 功能 11 的通信事件计数器。
11		获取通信事件计数器：MB_SERVER 使用内部通信事件计数器来记录发送到 Modbus 服务器的 Modbus 成功读取和写入请求次数。该计数器不会因功能 8 或功能 11 请求而增加。同时也不会因导致通信错误的任何请求而增加。 广播功能不能用于 Modbus TCP，因为在任何时刻仅存在一个客户端-服务器连接。

MB_SERVER 变量

下表给出了存储在 MB_SERVER 背景数据块中的公共静态变量（可在用户程序中使用）。

表格 13-146 MB_SERVER 公共静态变量

变量	数据类型	默认值	说明
HR_Start_Offset	Word	0	指定 Modbus 保持寄存器的起始地址
Request_Count	Word	0	该服务器接收到的所有请求的数量。
Server_Message_Count	Word	0	该特定服务器接收到的请求的数量。
Xmt_Rcv_Count	Word	0	出现错误的传输或接收的数量。此外，如果接收到一条无效的 Modbus 消息，该值加 1。
Exception_Count	Word	0	需要返回例外的 Modbus 特定错误数
Success_Count	Word	0	该特定服务器接收到的且无协议错误的请求数。
已连接	Bool	0	指示与所分配客户端的连接是已接通还是已断开：1 = 接通，0 = 断开

用户程序可以将值写入 HR_Start_Offset ，以控制 Modbus 服务器操作。可读取其它变量以监视 Modbus 的状态。

HR_Start_Offset

Modbus 保持寄存器地址从 40001 开始。这些地址与保持寄存器的 PLC 存储器起始地址对应。不过，可以组态“HR_Start_Offset”变量，将 Modbus 保持寄存器的起始地址定义为除 40001 之外的其它值。

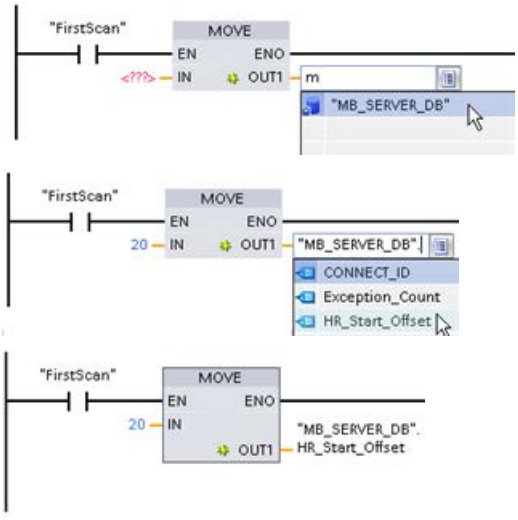
例如，如果保持寄存器被组态为起始于 MW100 并且长度为 100 个字。偏移量 20 可指定保持寄存器的起始地址为 40021 而不是 40001。低于 40021 和高于 40119 的任何地址都将导致寻址错误。

表格 13-147 Modbus 保持寄存器寻址示例

HR_Start_Offset	地址	最小值	最大值
0	Modbus 地址（字）	40001	40099
	S7-1200 地址	MW100	MW298
20	Modbus 地址（字）	40021	40119
	S7-1200 地址	MW100	MW298

HR_Start_Offset 是一个字值，用于指定 Modbus 保持寄存器的起始地址，存储在 MB_SERVER 背景数据块中。将 MB_SERVER 放入程序后，可利用参数助手下拉列表设置该公共静态变量值。

例如，将 MB_SERVER 放入 LAD 网络后，可以切换到先前的网络，并分配 HR_Start_Offset 值。该值必须在执行 MB_SERVER 前分配。



使用默认 DB 名称

输入 Modbus 服务器变量：

1. 将光标放在参数字段中，然后输入 m 字符。
2. 从 DB 名称的下拉列表中选择 “MB_SERVER_DB”。
3. 从 DB 变量的下拉列表中选择 “MB_SERVER_DB.HR_Start_Offset”。

表格 13-148 MB_SERVER 执行条件代码 [1]

STATUS (W#16#)	发送到 Modbus 服务器的响应代码 (B#16#)	Modbus 协议错误
7001		MB_SERVER 正在等待 Modbus 客户端连接到指定的 TCP 端口。仅在第一次执行连接或断开操作时才报告此代码。
7002		MB_SERVER 正在等待 Modbus 客户端连接到指定的 TCP 端口。等待连接或断开操作完成时，将针对任何后续执行报告此代码。
7003		断开操作已成功完成（仅在一个 PLC 扫描周期内有效）。
8187		指向 MB_HOLD_REG 的指针无效：区域太小
818C		指向优化的 MB_HOLD_REG 区（必须是未经优化的 DB 区或 M 储存区）的指针或受阻的过程超时超过 55 秒的限值。（仅适用于 S7-1200）
8381	01	不支持此功能代码
8382	03	数据长度错误
8383	02	数据地址错误或访问的数据超出 MB_HOLD_REG 地址区的界限
8384	03	数据值错误
8385	03	不支持此数据诊断代码值（功能代码 08）

[1] 除了上面列出的 MB_SERVER 错误外，也可以从底层传输块通信指令（TCON、TDISCON、TSEND 和 TRCV (页 862)）返回错误。

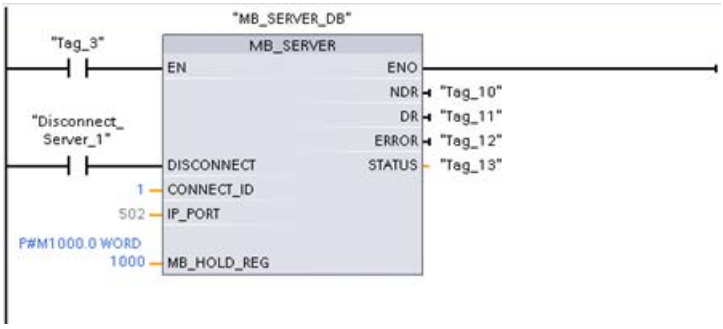
13.8.4 早期 Modbus TCP 示例

13.8.4.1 示例： 早期 MB_SERVER 多个 TCP 连接

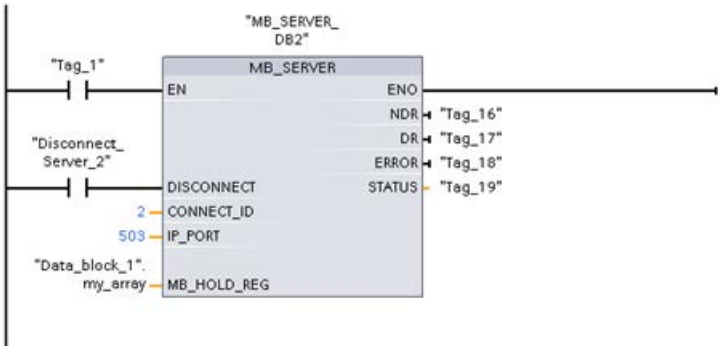
可以拥有多个 Modbus TCP 服务器连接。为此，必须为每个连接单独执行 MB_SERVER。每个连接必须使用单独的背景数据块、连接 ID 和 IP 端口。S7-1200 仅允许每个 IP 端口进行一个连接。

为了达到最佳性能，应在每个程序周期为各个连接执行 MB_SERVER。

程序段 1：带有独立 IP_PORT、连接 ID 和背景数据块的 1 号连接



程序段 2：带有独立 IP_PORT、连接 ID 和背景数据块的 2 号连接



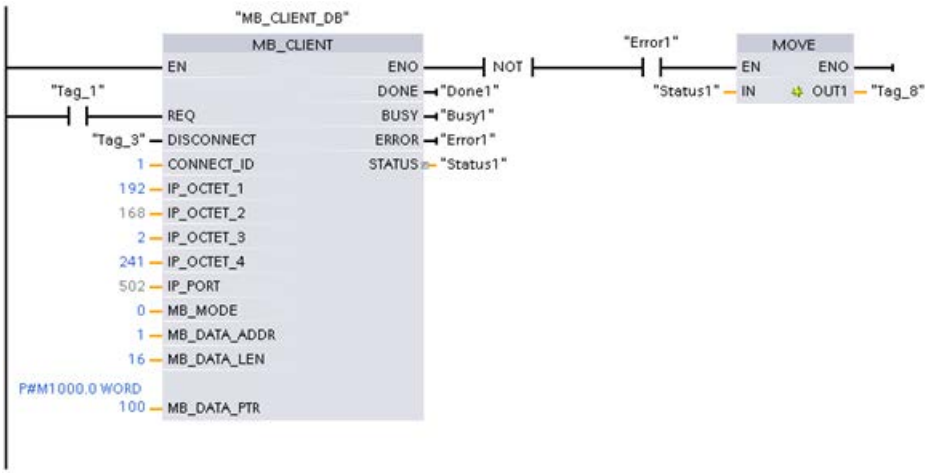
13.8.4.2 示例： 早期 MB_CLIENT 1： 通过公共 TCP 连接发送多个请求

多个 Modbus 客户端请求可通过同一连接发送。为此，必须使用相同的背景数据块、连接 ID 和端口号。

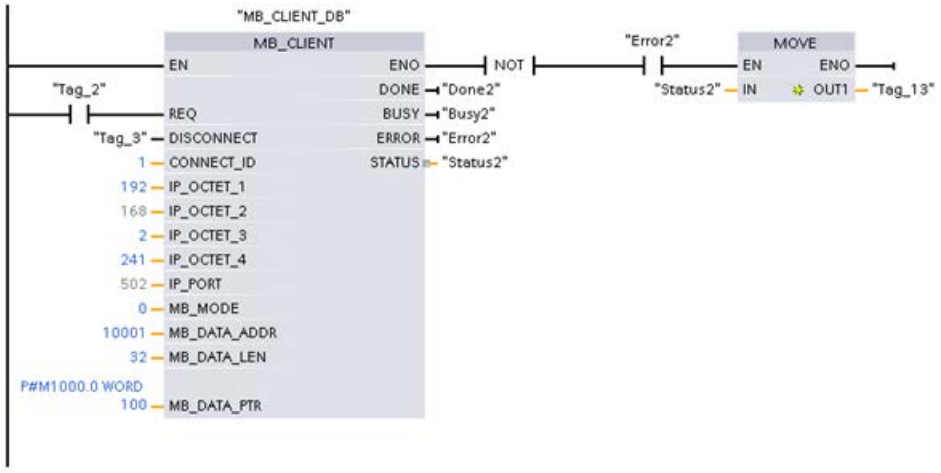
在任意给定时间，只能有一个客户端处于激活状态。一个客户端完成执行后，下一个客户端再开始执行。执行顺序由您的程序负责指定。

本示例所示为对同一存储区执行写操作的两个客户端。此外，还捕获了返回的错误，这是可选的。

程序段 1：Modbus 功能 1 - 读取 16 个输出映像位



程序段 2：Modbus 功能 2 - 读取 32 个输入映像位



13.8.4.3 示例： 早期 MB_CLIENT 2： 通过不同的 TCP 连接发送多个请求

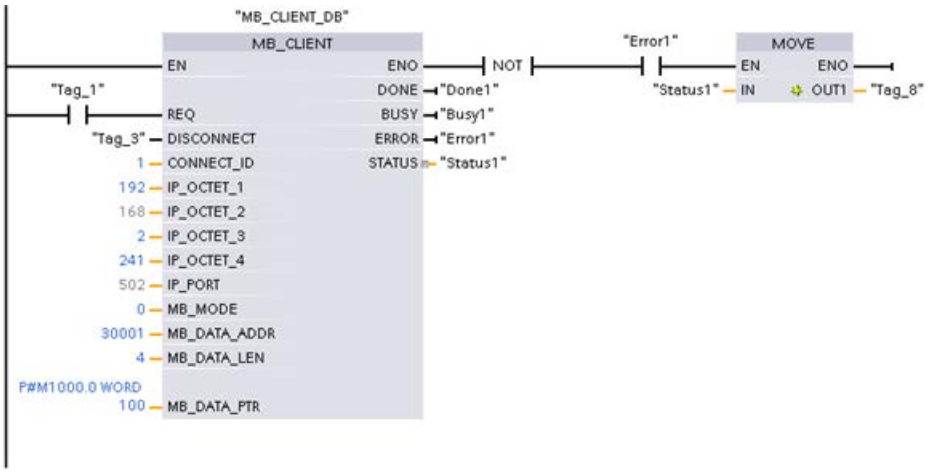
Modbus 客户端请求可通过不同连接来发送。为此，必须使用不同的背景数据块、IP 地址和连接 ID。

如要与同一 Modbus 服务器建立连接，端口号必须不同。如果与不同的服务器建立连接，则端口号方面没有限制。

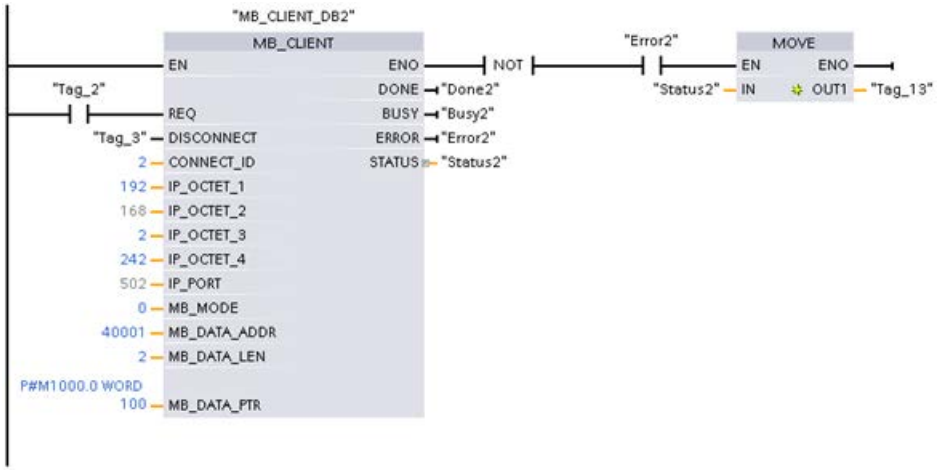
本示例所示为对同一存储区执行写操作的两个客户端。此外，还捕获了返回的错误，这是可选的。

程序段 1：

Modbus 功能 4 - 读取（S7-1200 存储器中的）输入字



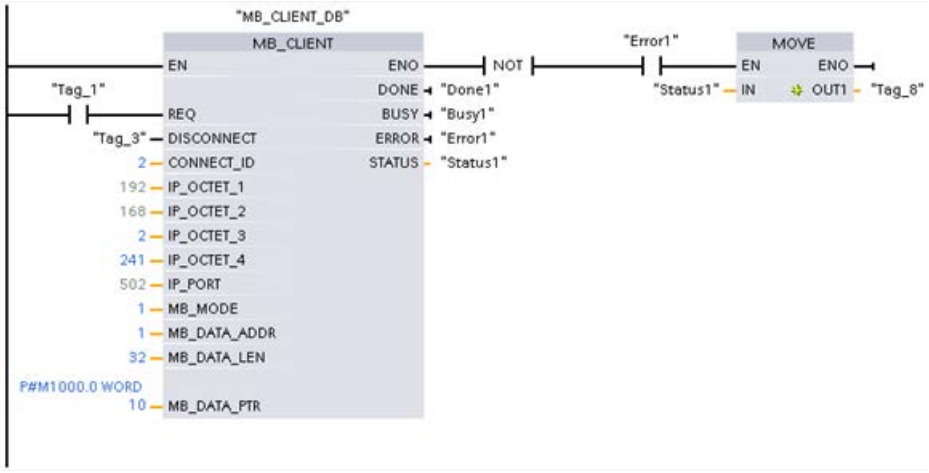
程序段 2：Modbus 功能 3 - 从 Modbus TCP 服务器读取保持寄存器字



13.8.4.4 示例： 早期 MB_CLIENT 3： 输出映像写入请求

本示例所示为 Modbus 客户端请求写入 S7-1200 输出映像。

程序段 1：Modbus 功能 15 - 写入 S7-1200 输出映像位

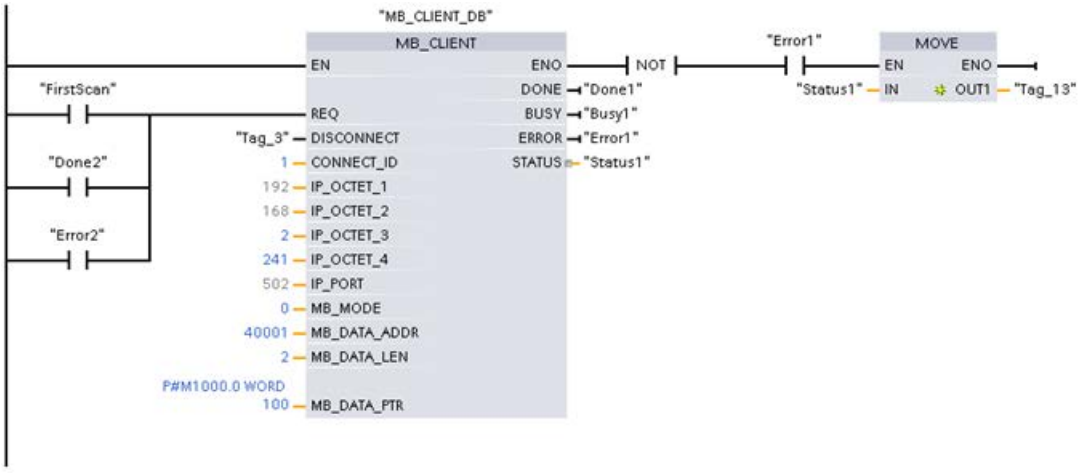


13.8.4.5 示例： 早期 MB_CLIENT 4： 协调多个请求

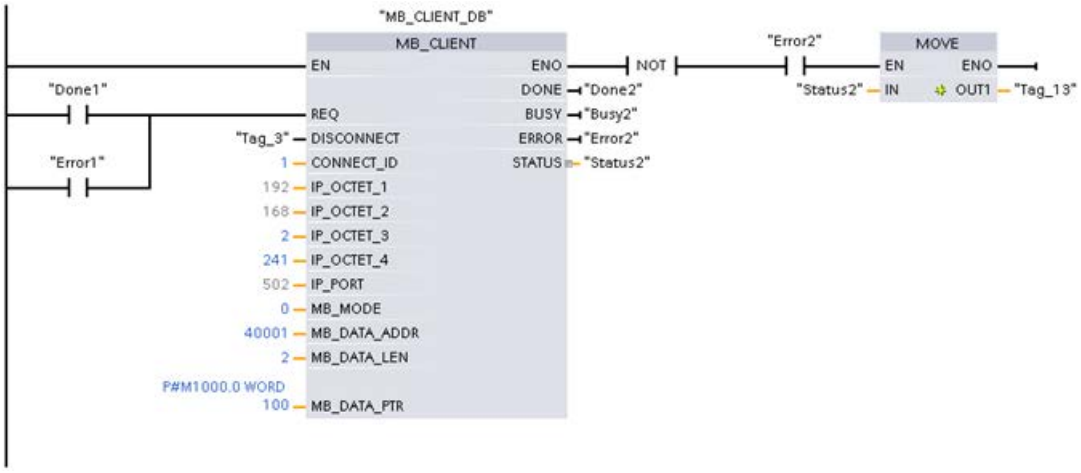
必须确保各个 Modbus TCP 请求都完成执行。此协调必须由程序提供。下面示例显示了首个和第二个客户端请求的输出如何用于协调执行。

本示例所示为对同一存储区执行写操作的两个客户端。此外，还捕获了返回的错误，这是可选的。

程序段 1：Modbus 功能 3 - 读取保持寄存器字



程序段 2：Modbus 功能 3 - 读取保持寄存器字



13.9 早期 Modbus RTU 通信（仅 CM/CB 1241）

13.9.1 概述

在 STEP 7 V13 SP1 和 S7-1200 V4.1 CPU 之前的版本中，Modbus RTU 通信指令以不同的名称存在，在某些情况下，接口也略有不同。 一般概念适用于两个指令集。 关于编程信息，请参见各个早期 Modbus RTU 指令。

13.9.2 选择 Modbus RTU 指令的版本

在 STEP 7 中可使用两个版本的早期 Modbus RTU 指令：

- 历史版本 1.3：兼容所有 CPU 和 CP 版本
- 历史版本 2.2：兼容所有 CPU 和 CP 版本
  （注：版本 2.2 将参数 REQ 和 DONE 添加到 MB_COMM_LOAD。在 V2.2 中，MB_MASTER 和 MB_SLAVE 的 MB_ADDR 参数还提供 UInt 值以进行扩展寻址。）

可以从兼容性和移植便利性方面考虑，选择将相应的指令版本插入用户程序中。

在指令任务卡中，在“通信”(Communication) 处理器组中显示 MODBUS 指令。

要更改 Modbus 指令的版本，请从下拉列表中选择相应版本。可以选择一组指令或分别选择各个指令。

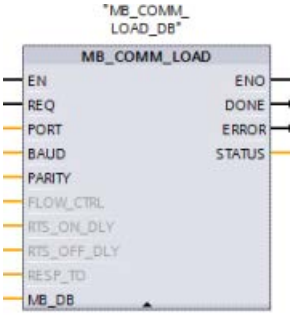
使用指令树将 Modbus 指令放入程序时，将在项目树中创建新的 FB 实例。在项目树的 “PLC_x > 程序块 > 系统块 > 程序资源”(PLC_x > Program blocks > System blocks > Program resources) 下可看到新的 FB 实例。

要确认程序中 Modbus 指令的版本，必须检查项目树的属性而不是程序编辑器中显示的框的属性。选择项目树的 Modbus FB 实例，单击右键，选择 “属性”(Properties)，然后选择 “信息”(Information) 页查看 Modbus 指令的版本号。

13.9.3 早期 Modbus RTU 指令

13.9.3.1 MB_COMM_LOAD（针对 Modbus RTU 组态 PtP 模块上的端口）

表格 13-149 MB_COMM_LOAD 指令

LAD/FBD	SCL	说明
	"MB_COMM_LOAD_DB"(REQ:=_bool_in, PORT:=_uint_in_, BAUD:=_udint_in_, PARITY:=_uint_in_, FLOW_CTRL:=_uint_in_, RTS_ON_DLY:=_uint_in_, RTS_OFF_DLY:=_uint_in_, RESP_TO:=_uint_in_, DONE=>_bool_out, ERROR=>_bool_out_, STATUS=>_word_out_, MB_DB:=_fbtref_inout_);	MB_COMM_LOAD 指令可组态用于 Modbus RTU 协议通信的 PtP 端口。Modbus 端口硬件选项：最多安装三个 CM（RS485 或 RS232），及一个 CB (R4845)。将 MB_COMM_LOAD 指令放入程序时自动分配背景数据块。

表格 13-150 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	通过由低到高的（上升沿）信号启动操作。 （仅版本 2.0）
PORT	IN	Port	安装并组态 CM 或 CB 通信设备之后，端口标识符将出现在 PORT 功能框连接的参数助手下拉列表中。分配的 CM 或 CB 端口值为设备配置属性“硬件标识符”。端口符号名称在 PLC 变量表的“系统常量”(System constants) 选项卡中分配。

参数和类型		数据类型	说明
BAUD	IN	UDInt	波特率选择： 300、600、1200、2400、4800、9600、19200、38400、57600、76800、115200，其它所有值均无效
PARITY	IN	UInt	奇偶校验选择： • 0 – 无 • 1 – 奇校验 • 2 – 偶校验
FLOW_CTRL [1]	IN	UInt	流控制选择： • 0 –（默认）无流控制 • 1 – RTS 始终为 ON 的硬件流控制（不适用于 RS485 端口） • 2 - 带 RTS 切换的硬件流控制
RTS_ON_DLY [1]	IN	UInt	RTS 接通延时选择： • 0 –（默认）从 RTS 激活一直到传送消息的第一个字符之前无延时 • 1 到 65535 – 从 RTS 激活一直到传送消息的第一个字符之前以毫秒表示的延时（不适用于 RS485 端口）。不管 FLOW_CTRL 选择为何，都将应用 RTS 延时。
RTS_OFF_DLY [1]	IN	UInt	RTS 关断延时选择： • 0 –（默认）从传送最后一个字符一直到 RTS 转入非活动状态之前无延时 • 1 到 65535 – 从传送最后一个字符一直到 RTS 转入非活动状态之前以毫秒表示的延时（不适用于 RS485 端口）。不管 FLOW_CTRL 选择为何，都将应用 RTS 延时。
RESP_TO [1]	IN	UInt	响应超时： MB_MASTER 允许用于从站响应的时间（以毫秒为单位）。如果从站在此时间段内未响应，MB_MASTER 将重试请求，或者在发送指定次数的重试请求后终止请求并提示错误。 5 ms 到 65535 ms（默认值 = 1000 ms）。
MB_DB	IN	Variant	对 MB_MASTER 或 MB_SLAVE 指令所使用的背景数据块的引用。在用户的程序中放置 MB_SLAVE 或 MB_MASTER 后，该 DB 标识符将出现在 MB_DB 功能框连接的参数助手下拉列表中。
DONE	OUT	Bool	上一请求已完成且没有出错后，DONE 位将保持为 TRUE 一个扫描周期时间。（仅版本 2.0）

参数和类型		数据类型	说明
ERROR	OUT	Bool	上一请求因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。STATUS 参数中的错误代码值仅在 ERROR = TRUE 的一个扫描周期内有效。
STATUS	OUT	Word	执行条件代码

[1] MB_COMM_LOAD（V 2.x 或更高版本）的可选参数。单击 LAD/FBD 框底部的箭头，展开框并包含这些参数。

可执行 MB_COMM_LOAD 来组态端口以使用 Modbus RTU 协议。为使用 Modbus RTU 协议组态端口后，该端口只能由 MB_MASTER 或 MB_SLAVE 指令使用。

对用于 Modbus 通信的每个通信端口，都必须执行一次 MB_COMM_LOAD 来组态。为要使用的每个端口分配一个唯一的 MB_COMM_LOAD 背景数据块。最多可在 CPU 中安装三个通信模块（RS232 或 RS485）和一个通信板 (RS485)。从启动 OB 调用 MB_COMM_LOAD 并执行它一次，或使用第一个扫描系统标记 (页 106)发起调用以执行它一次。只有在必须更改波特率或奇偶校验等通信参数时，才再次执行 MB_COMM_LOAD。

将 MB_MASTER 或 MB_SLAVE 指令放入用户程序中时，将为其分配背景数据块。指定 MB_COMM_LOAD 指令的 MB_DB 参数时将引用该背景数据块。

MB_COMM_LOAD 数据块变量

下表给出存储在 MB_COMM_LOAD 的背景数据块中的公共静态变量（可在用户程序中使用）。

表格 13-151 背景数据块中的静态变量

变量	数据类型	说明
ICHAR_GAP	UInt	字符间隙延时。此参数以毫秒为单位指定，用于增加预期的接收字符间的时间量。与此参数对应的位时间个数加到 Modbus 默认的 35 个位时间（3.5 个字符时间）。
RETRIES	UInt	主站在返回无响应错误代码 0x80C8 之前的重试次数。
STOP_BITS	USInt	每个字符组帧的停止位数目。有效值为 1 和 2。

表格 13-152 MB_COMM_LOAD 执行条件代码 [1]

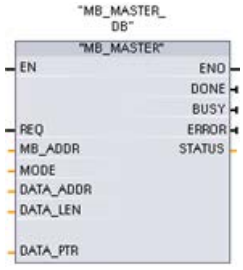
STATUS (W#16#)	说明
0000	无错误
8180	端口 ID 值无效（通信模块的端口/硬件标识符错误）
8181	波特率值无效

STATUS (W#16#)	说明
8182	奇偶校验值无效
8183	流控制值无效
8184	响应超时值无效（响应超时小于最小值 5 ms）
8185	MB_DB 参数不是 MB_MASTER 或 MB_SLAVE 指令的背景数据块。

[1] 除了上述列出的 MB_COMM_LOAD 错误，还可能返回底层 PtP 通信指令的错误。

13.9.3.2 MB_MASTER（作为 Modbus RTU 主站使用 PtP 端口通信）

表格 13-153 MB_MASTER 指令

LAD/FBD	SCL	说明
	"MB_MASTER_DB"(REQ:=_bool_in_, MB_ADDR:=_uint_in_, MODE:=_usint_in_, DATA_ADDR:=_udint_in_, DATA_LEN:=_uint_in_, DONE=>_bool_out_, BUSY=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_, DATA_PTR:=_variant_inout_);	MB_MASTER 指令作为 Modbus 主站利用之前执行 MB_COMM_LOAD 指令组态的端口进行通信。将 MB_MASTER 指令放入程序时自动分配背景数据块。指定 MB_COMM_LOAD 指令的 MB_DB 参数时将使用该 MB_MASTER 背景数据块。

表格 13-154 参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	0 = 无请求 1 = 请求将数据传送到 Modbus 从站
MB_ADDR	IN	V1.0: USInt V2.0: UInt	Modbus RTU 站地址： 标准寻址范围（1 到 247） 扩展寻址范围（1 到 65535） 值 0 被保留用于将消息广播到所有 Modbus 从站。只有 Modbus 功能代码 05、06、15 和 16 是可用于广播的功能代码。
MODE	IN	USInt	模式选择：指定请求类型（读、写或诊断）。请参见下面的 Modbus 功能表了解详细信息。
DATA_ADDR	IN	UDInt	从站中的起始地址：指定要在 Modbus 从站中访问的数据的起始地址。请参见下面的 Modbus 功能表了解有效地址信息。

参数和类型		数据类型	说明
DATA_LEN	IN	UInt	数据长度：指定此请求中要访问的位数或字数。请参见下面的 Modbus 功能表了解有效长度信息。
DATA_PTR	IN	Variant	数据指针：指向要写入或读取的数据的 M 或 DB 地址（未经优化的 DB 类型）。
DONE	OUT	Bool	上一请求已完成且没有出错后，DONE 位将保持为 TRUE 一个扫描周期时间。
BUSY	OUT	Bool	• 0 – 无正在进行的 MB_MASTER 操作 • 1 – MB_MASTER 操作正在进行
ERROR	OUT	Bool	上一请求因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。STATUS 参数中的错误代码值仅在 ERROR = TRUE 的一个扫描周期内有效。
STATUS	OUT	Word	执行条件代码

Modbus 主站通信规则

- 必须先执行 MB_COMM_LOAD 组态端口，然后 MB_MASTER 指令才能与该端口通信。
- 如果要将某个端口用于初始化 Modbus 主站请求，则 MB_SLAVE 不应使用该端口。MB_MASTER 执行的一个或多个实例可使用该端口，但是对于该端口，所有 MB_MASTER 执行都必须使用同一个 MB_MASTER 背景数据块。
- Modbus 指令不使用通信中断事件来控制通信过程。用户程序必须轮询 MB_MASTER 指令以了解传送和接收的完成情况。
- 对于给定的端口，从程序循环 OB 中调用所有 MB_MASTER 执行。Modbus 主站指令只能在一个程序循环或循环/延时执行等级执行。它们不能同时在两种执行优先级中执行。如果一个 Modbus 主站指令被另一个执行优先级更高的 Modbus 主站取代，将导致不正确的操作。Modbus 主站指令不能在启动、诊断或时间错误执行优先级执行。
- 主站指令启动传输后，必须连续执行已启用 EN 输入的该实例直到返回状态 DONE=1 或状态 ERROR=1 为止。在这两个事件其中之一发生前，一个特殊的 MB_MASTER 实例被视为已激活。原始实例激活后，调用已启用 REQ 输入的其它任何实例都将导致错误。如果原始实例的连续执行过程停止，则请求状态保持激活一段时间，该时间由静态变量 Blocked_Proc_Timeout 指定。一旦超出该时间段，则下一个使用激活的 REQ 输入调用的主站指令成为激活实例。这可以防止单个 Modbus 主站指令独占或锁定对端口的访问。如果在由静态变量“Blocked_Proc_Timeout”指定的时间段内没有启用原始激活的实例，则下次执行此实例（未设置 REQ）时将清除激活状态。如果设置了 REQ，则此次执行将启动新的主站请求，如同其它实例未曾激活一样。

REQ 参数

0 = 无请求；1 = 请求将数据传送到 Modbus 从站

可使用电平或边沿触发的触点控制此输入。只要此输入启用，状态机便会启动，以确保在当前请求完成前不允许使用同一背景数据块的任何其它 MB_MASTER 发出请求。在当前请求执行期间，将捕获所有其它输入状态并内部保存，直到接收到响应或检测到错误。

如果在当前请求完成前 REQ 输入 = 1，从而再次执行 MB_MASTER 的同一实例，则不会进行任何后续传送。但是，如果当前请求已完成，因为 REQ 输入 = 1 而再次执行 MB_MASTER 时，便会发出新请求。

DATA_ADDR 和 MODE 参数用于选择 Modbus 功能类型

DATA_ADDR（从站中的 Modbus 起始地址）：指定要在 Modbus 从站中访问的数据的起始地址。

MB_MASTER 指令使用 MODE 输入而非功能代码输入。MODE 和 Modbus 地址一起确定实际 Modbus 消息中使用的功能代码。下表列出了 MODE 参数、Modbus 功能代码和 Modbus 地址范围之间的对应关系。

表格 13-155 Modbus 功能

MODE	Modbus 功能	数据长度	操作和数据	Modbus 地址
0	01	1 到 2000 1 到 1992 [1]	读取输出位： 每个请求 1 到 1992 或 2000 个位	1 到 9999
0	02	1 到 2000 1 到 1992 [1]	读取输入位： 每个请求 1 到 1992 或 2000 个位	10001 到 19999
0	03	1 到 125 1 到 124 [1]	读取保持寄存器： 每个请求 1 到 124 或 125 个字	40001 到 49999 或 400001 到 465535
0	04	1 到 125 1 到 124 [1]	读取输入字： 每个请求 1 到 124 或 125 个字	30001 到 39999
1	05	1	写入一个输出位： 每个请求一位	1 到 9999
1	06	1	写入一个保持寄存器： 每个请求 1 个字	40001 到 49999 或 400001 到 465535
1	15	2 到 1968 2 到 1960 [1]	写入多个输出位： 每个请求 2 到 1960 或 1968 个位	1 到 9999

MODE	Modbus 功能	数据长度	操作和数据	Modbus 地址
1	16	2 到 123 2 到 122 [1]	写入多个保持寄存器： 每个请求 2 到 122 或 123 个字	40001 到 49999 或 400001 到 465535
2	15	1 到 1968 2 到 1960 [1]	写入一个或多个输出位： 每个请求 1 到 1960 或 1968 个位	1 到 9999
2	16	1 到 123 1 到 122 [1]	写入一个或多个保持寄存器： 每个请求 1 到 122 或 123 个字	40001 到 49999 或 400001 到 465535
11	11	0	读取从站通信状态字和事件计数器。状态字指示忙闲情况（0 - 不忙，0xFFFF - 忙）。每成功完成一条消息，事件计数器的计数值递增。 对于该功能，MB_MASTER 的 DATA_ADDR 和 DATA_LEN 操作数都将被忽略。	
80	08	1	利用数据诊断代码 0x0000 检查从站状态（回送测试 - 从站回送请求） 每个请求 1 个字	
81	08	1	利用数据诊断代码 0x000A 重新设置从站事件计数器 每个请求 1 个字	
3 到 10、12 到 79、82 到 255			保留	

[1] 对于“扩展寻址”模式，根据功能所使用的数据类型，数据的最大长度将减小 1 个字节或 1 个字。

DATA_PTR 参数

DATA_PTR 参数指向要写入或读取的 DB 或 M 地址。如果使用数据块，则必须创建一个全局数据块为读写 Modbus 从站提供数据存储位置。

说明

DATA_PTR 数据块类型必须允许直接寻址

该数据块必须允许直接（绝对）寻址和符号寻址。创建该数据块时，必须选择“标准” (Standard) 访问属性。

DATA_PTR 参数的数据块结构

- 这些数据类型对 Modbus 地址 30001 到 39999、40001 到 49999 和 400001 到 465536 的**字读取**有效，对 Modbus 地址 40001 到 49999 和 400001 到 465536 的**字写入**也有效。
  - WORD、UINT 或 INT 数据类型的标准数组
  - 指定的 WORD、UINT 或 INT 结构，其中每个元素都具有唯一的名称和 16 位数据类型。
  - 指定的复杂结构，其中每个元素都具有唯一的名称以及 16 或 32 位数据类型。
- 用于 Modbus 地址 00001 到 09999 的**位读取**和写入和 10001 到 19999 的位读取。
  - 布尔数据类型的标准数组。
  - 唯一命名的布尔变量的已命名布尔结构。
- 尽管不是必需的，但还是建议每个 MB_MASTER 指令都具有各自的单独存储区。此建议的原因在于，如果多个 MB_MASTER 指令读取和写入同一个存储区，发生数据损坏的可能性会更大。
- 不要求 DATA_PTR 数据区位于同一个全局数据块中。可创建一个具有多个区域的数据块供 Modbus 读取、一个数据块供 Modbus 写入或一个数据块用于各个从站。

Modbus 主站数据块变量

下表给出存储在 MB_MASTER 的背景数据块中的公共静态变量（可在用户程序中使用）。

表格 13-156 背景数据块中的静态变量

变量	数据类型	初始值	说明
Blocked_Proc_Timeout	Real	3.0	在 Modbus 主站实例受阻后，移除该激活的实例前需等待的时间（秒）。例如，当已发出主站请求，但程序在彻底完成该请求前停止调用该主站功能时，就会出现这种情况。时间值必须大于 0 且小于 55 秒，否则发生错误。默认值为 .5 秒。
Extended_Addressing	Bool	False	组态单字节或双字节从站寻址。默认值 = 0。 （0= 单字节地址、1= 双字节地址）

用户程序可以将值写入 Blocked_Proc_Timeout 和 Extended_Addressing 变量，以控制 Modbus 主站操作。有关如何在程序编辑器中使用这些变量的示例以及有关 Modbus 扩展寻址的详细信息，请参见 HR_Start_Offset 和 Extended_Addressing 的 MB_SLAVE 主题说明 (页 1353)。

条件代码

表格 13-157 MB_MASTER 执行条件代码（通信和组态错误）[1]

STATUS (W#16#)	说明
0000	无错误
80C8	从站超时。检查波特率、奇偶校验和从站的接线。
80D1	接收方发出了暂停主动传输的流控制请求并且在指定的等待时间内未重新激活该传输。在硬件流控制期间，如果接收方在指定的等待时间内没有声明 CTS，也会产生该错误。
80D2	传送请求中止，因为没有从 DCE 收到任何 DSR 信号。
80E0	因接收缓冲区已满，消息被终止。
80E1	因出现奇偶校验错误，消息被终止。
80E2	因组帧错误，消息被终止。
80E3	因出现超限错误，消息被终止。
80E4	因指定长度超出总缓冲区大小，消息被终止。
8180	无效端口 ID 值或 MB_COMM_LOAD 指令出错
8186	Modbus 站地址无效
8188	指定给广播请求的模式无效
8189	数据地址值无效
818A	数据长度值无效
818B	指向本地数据源/目标的指针无效：大小不正确
818C	DATA_PTR 的指针无效或 Blocked_Proc_Timeout 无效：数据区必须是 DB（允许符号访问和直接访问）或 M 存储区。
8200	端口正忙于处理传送请求。

表格 13-158 MB_MASTER 执行条件代码（Modbus 协议错误）[1]

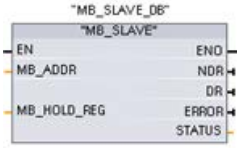
STATUS (W#16#)	从站的响应代码	Modbus 协议错误
8380	-	CRC 错误
8381	01	不支持此功能代码
8382	03	数据长度错误
8383	02	数据地址错误或地址超出 DATA_PTR 区的有效范围
8384	大于 03	数据值错误
8385	03	不支持此数据诊断代码值（功能代码 08）

STATUS (W#16#)	从站的响应代码	Modbus 协议错误
8386	-	响应中的功能代码与请求中的代码不匹配。
8387	-	响应的从站错误
8388	-	从站对写请求的响应不正确。从站返回的写请求与主站实际发送的写请求不匹配。

[1] 除了上述列出的 MB_MASTER 错误，还可能返回底层 PtP 通信指令的错误。

13.9.3.3 MB_SLAVE（作为 Modbus RTU 从站使用 PtP 端口通信）

表格 13-159 MB_SLAVE 指令

LAD/FBD	SCL	说明
	"MB_SLAVE_DB"(MB_ADDR:=_uint_in_, NDR=>_bool_out_, DR=>_bool_out_, ERROR=>_bool_out_, STATUS=>_word_out_, MB_HOLD_REG:=_variant_inout_);	MB_SLAVE 指令允许用户程序作为 Modbus 从站通过 CM（RS485 或 RS232）和 CB (RS485) 上的 PtP 端口进行通信。远程 Modbus RTU 主站发出请求时，用户程序会通过执行 MB_SLAVE 进行响应。STEP 7 在插入指令时自动创建背景数据块。在为 MB_COMM_LOAD 指令指定 MB_DB 参数时使用此 MB_SLAVE_DB 名称。

表格 13-160 参数的数据类型

参数和类型		数据类型	说明
MB_ADDR	IN	V1.0: USInt V2.0: UInt	Modbus 从站的站地址： 标准寻址范围（1 到 247） 扩展寻址范围（0 到 65535）
MB_HOLD_REG	IN	Variant	指向 Modbus 保持寄存器 DB 的指针：Modbus 保持寄存器可以是 M 存储器或数据块。
NDR	OUT	Bool	新数据就绪： • 0 – 无新数据 • 1 – 表示 Modbus 主站已写入新数据
DR	OUT	Bool	数据读取： • 0 – 无数据读取 • 1 – 表示 Modbus 主站已读取数据

参数和类型		数据类型	说明
ERROR	OUT	Bool	上一请求因错误而终止后，ERROR 位将保持为 TRUE 一个扫描周期时间。如果执行因错误而终止，则 STATUS 参数的错误代码值仅在 ERROR = TRUE 的一个扫描周期内有效。
STATUS	OUT	Word	执行错误代码

Modbus 通信功能代码（1、2、4、5 和 15）可以在 CPU 的输入过程映像及输出过程映像中直接读写位和字。对于这些功能代码，MB_HOLD_REG 参数必须定义为大于一个字节的数据类型。下表给出了 Modbus 地址与 CPU 过程映像的映射示例。

表格 13-161 Modbus 地址到过程映像的映射

Modbus 功能						S7-1200	
代码	功能	数据区	地址范围			数据区	CPU 地址
01	读位	输出	1	到	8192	输出过程映像	Q0.0 到 Q1023.7
02	读位	输入	10001	到	18192	输入过程映像	I0.0 到 I1023.7
04	读字	输入	30001	到	30512	输入过程映像	IW0 到 IW1022
05	写位	输出	1	到	8192	输出过程映像	Q0.0 到 Q1023.7
15	写位	输出	1	到	8192	输出过程映像	Q0.0 到 Q1023.7

Modbus 通信功能代码（3、6、16）使用 Modbus 保持寄存器，该寄存器可以是 M 存储区地址范围或数据块。保持寄存器的类型由 MB_SLAVE 指令的 MB_HOLD_REG 参数指定。

说明

MB_HOLD_REG 数据块类型

Modbus 保持寄存器数据块必须允许直接（绝对）寻址和符号寻址。创建该数据块时，必须选择“标准”(Standard) 访问属性。

下表给出了 Modbus 地址到保持寄存器的映射示例，这种映射用于 Modbus 功能代码 03（读取字）、06（写入字）和 16（读取字）。DB 地址的实际上限取决于每种 CPU 型号的最大工作存储器限值和 M 存储器限值。

表格 13-162 Modbus 地址到 CPU 存储器的映射

Modbus 主站地址	MB_HOLD_REG 参数示例				
	MW100	DB10.DBw0	MW120	DB10.DBW50	"Recipe".ingredient
40001	MW100	DB10.DBW0	MW120	DB10.DBW50	"Recipe".ingredient[1]
40002	MW102	DB10.DBW2	MW122	DB10.DBW52	"Recipe".ingredient[2]

Modbus 主站地址	MB_HOLD_REG 参数示例				
	MW100	DB10.DBw0	MW120	DB10.DBW50	"Recipe".ingredient
40003	MW104	DB10.DBW4	MW124	DB10.DBW54	"Recipe".ingredient[3]
40004	MW106	DB10.DBW6	MW126	DB10.DBW56	"Recipe".ingredient[4]
40005	MW108	DB10.DBW8	MW128	DB10.DBW58	"Recipe".ingredient[5]

表格 13-163 诊断功能

S7-1200 MB_SLAVE Modbus 诊断功能		
代码	子功能	说明
08	0000H	返回查询数据回送测试：MB_SLAVE 将向 Modbus 主站回送接收到的数据字。
08	000AH	清除通信事件计数器：MB_SLAVE 将清除用于 Modbus 功能 11 的通信事件计数器。
11		获取通信事件计数器：MB_SLAVE 使用内部通信事件计数器来记录发送到 Modbus 从站的 Modbus 成功读取和写入请求次数。该计数器不会因功能 8、功能 11 或广播请求而增加。同样也不会因任何导致通信错误（例如，奇偶校验错误或 CRC 错误）的请求而增加。

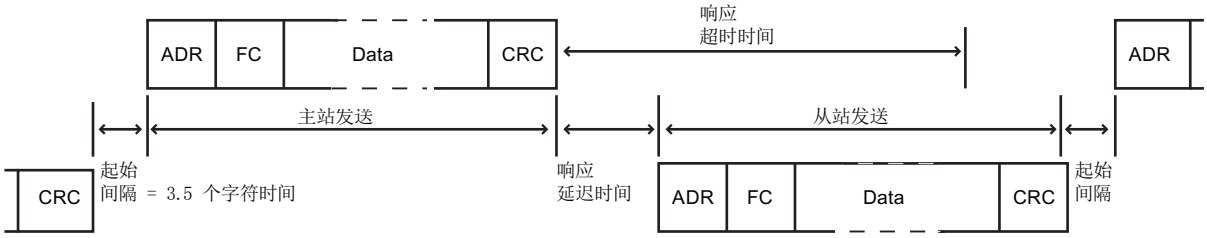
MB_SLAVE 指令支持来自任何 Modbus 主站的广播写请求，只要该请求是用于访问有效地址的请求即可。对于广播不支持的功能代码，MB_SLAVE 将生成错误代码 0x8188。

Modbus 从站通信规则

- 必须先执行 MB_COMM_LOAD 组态端口，然后 MB_SLAVE 指令才能通过该端口通信。
- 如果某个端口作为从站响应 Modbus 主站，则请勿使用 MB_MASTER 指令对该端口进行编程。
- 对于给定端口，只能使用一个 MB_SLAVE 实例，否则将出现不确定的行为。
- Modbus 指令不使用通信中断事件来控制通信过程。用户程序必须通过轮询 MB_SLAVE 指令以了解传送和接收的完成情况来控制通信过程。
- MB_SLAVE 指令必须以一定的速率定期执行，以便能够及时响应来自 Modbus 主站的进入请求。建议每次扫描时都从程序循环 OB 执行 MB_SLAVE。也可以从循环中断 OB 执行 MB_SLAVE，但并不建议这么做，因为中断例程的延时过长可能会暂时阻止其它中断例程的执行。

Modbus 定时信号

必须周期性执行 MB_SLAVE，才能接收来自 Modbus 主站的每个请求并随之按要求响应。MB_SLAVE 的执行频率取决于 Modbus 主站的响应超时时间。下图对此进行了说明。



响应超时时间 RESP_TO 是 Modbus 主站等待 Modbus 从站开始响应的时间。该时间段不是由 Modbus 协议定义的，而是属于每个 Modbus 主站的一个参数。必须基于用户 Modbus 主站的具体参数确定 MB_SLAVE 的执行频率（相邻两次执行之间的时间）。在 Modbus 主站的响应超时时间内至少应执行两次 MB_SLAVE。

Modbus 从站变量

下表给出了存储在 MB_SLAVE 背景数据块（可在用户程序中使用）中的公共静态变量。

表格 13-164 Modbus 从站变量

变量	数据类型	说明
Request_Count	Word	该从站接收到的所有请求的数量
Slave_Message_Count	Word	该特定从站接收到的请求的数量
Bad_CRC_Count	Word	接收到的具有 CRC 错误的请求的数量
Broadcast_Count	Word	接收到的广播请求的数量
Exception_Count	Word	需要返回例外的 Modbus 特定错误数
Success_Count	Word	该特定从站接收到的没有协议错误的请求数量
HR_Start_Offset	Word	指定 Modbus 保持寄存器的起始地址（默认值 = 0）
Extended_Addressing	Bool	组态单字节或双字节从站寻址 （0= 单字节地址、1= 双字节地址、默认 = 0）

程序可以将值写入 HR_Start_Offset 和 Extended_Addressing 变量以控制 Modbus 从站操作。可读取其它变量以监视 Modbus 的状态。

HR_Start_Offset

Modbus 保持寄存器地址从 40001 或 400001 开始。这些地址与保持寄存器的 PLC 存储器起始地址对应。不过，可以组态“HR_Start_Offset”变量，将 Modbus 保持寄存器的起始地址定义为除 40001 或 400001 之外的其它值。

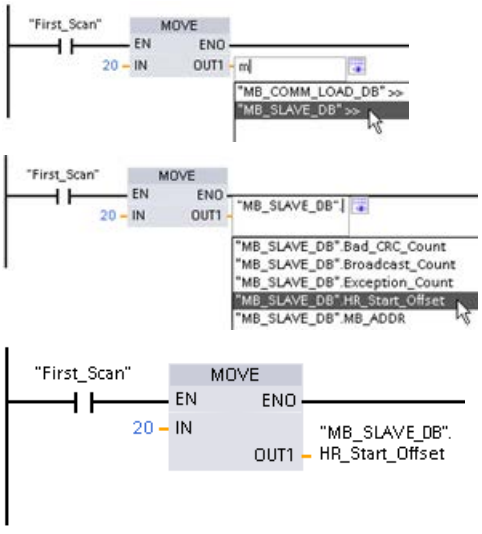
例如，如果保持寄存器被组态为起始于 MW100 并且长度为 100 个字。偏移量 20 可指定保持寄存器的起始地址为 40021 而不是 40001。低于 40021 和高于 400119 的任何地址都将导致寻址错误。

表格 13-165 Modbus 保持寄存器寻址示例

HR_Start_Offset	地址	最小值	最大值
0	Modbus 地址（字）	40001	40099
	S7-1200 地址	MW100	MW298
20	Modbus 地址（字）	40021	40119
	S7-1200 地址	MW100	MW298

HR_Start_Offset 是一个字值，用于指定 Modbus 保持寄存器的起始地址，存储在 MB_SLAVE 背景数据块中。将 MB_SLAVE 放入程序后，可利用参数助手下拉列表设置该公共静态变量值。

例如，将 MB_SLAVE 放入 LAD 程序段后，可以切换到先前的程序段，分配 HR_Start_Offset 值。该值必须在执行 MB_SLAVE 前分配。



使用默认 DB 名称输入 Modbus 从站变量：

1. 将光标放在参数字段中，然后输入 m 字符。
2. 从下拉列表中选择“MB_SLAVE_DB”。
3. 将光标放在 DB 名称的右侧（引号字符的后面），然后输入句点字符。
4. 从下拉列表中选择“MB_SLAVE_DB.HR_Start_Offset”。

Extended_Addressing

Extended_Addressing 变量的访问方式与上述的 HR_Start_Offset 参考相似，只是 Extended_Addressing 变量是布尔值。布尔值必须通过输出线圈（而非 MOVE 块）写入。

Modbus 从站寻址可组态为单字节（Modbus 标准方式）或双字节。扩展寻址用于对单一网络内超过 247 台设备进行寻址。选择扩展寻址后，最多可以对 64000 个地址进行寻址。下面以 Modbus 功能 1 的帧为例进行显示。

表格 13-166 单字节从站地址（字节 0）

功能 1	字节 0	字节 1	字节 2	字节 3	字节 4	字节 5	
请求	从站地址	F 代码	起始地址		线圈长度		
有效响应	从站地址	F 代码	长度	线圈数据			
错误响应	从站地址	0x81	E 代码				

表格 13-167 双字节从站地址（字节 0 和字节 1）

	字节 0	字节 1	字节 2	字节 3	字节 4	字节 5	字节 6
请求	从站地址		F 代码	起始地址		线圈长度	
有效响应	从站地址		F 代码	长度	线圈数据		
错误响应	从站地址		0x81	E 代码			

条件代码

表格 13-168 MB_SLAVE 执行条件代码（通信和组态错误）[1]

STATUS (W#16#)	说明
80D1	接收方发出了暂停主动传输的流控制请求并且在指定的等待时间内未重新激活该传输。 在硬件流控制期间，如果接收方在指定的等待时间内没有声明 CTS，也会产生该错误。
80D2	传送请求中止，因为没有从 DCE 收到任何 DSR 信号。
80E0	因接收缓冲区已满，消息被终止。
80E1	因出现奇偶校验错误，消息被终止。
80E2	因组帧错误，消息被终止。
80E3	因出现超限错误，消息被终止。

STATUS (W#16#)	说明
80E4	因指定长度超出总缓冲区大小，消息被终止。
8180	无效端口 ID 值或 MB_COMM_LOAD 指令出错
8186	Modbus 站地址无效
8187	指向 MB_HOLD_REG DB 的指针无效：区域太小
818C	指向 M 存储器或 DB（DB 区域必须允许符号地址和直接地址）的 MB_HOLD_REG 指针无效

表格 13-169 MB_SLAVE 执行条件代码（Modbus 协议错误）[1]

STATUS (W#16#)	从站的响应代码	Modbus 协议错误
8380	无响应	CRC 错误
8381	01	不支持功能代码或在广播内不支持
8382	03	数据长度错误
8383	02	数据地址错误或地址超出 DATA_PTR 区的有效范围
8384	03	数据值错误
8385	03	不支持此数据诊断代码值（功能代码 08）

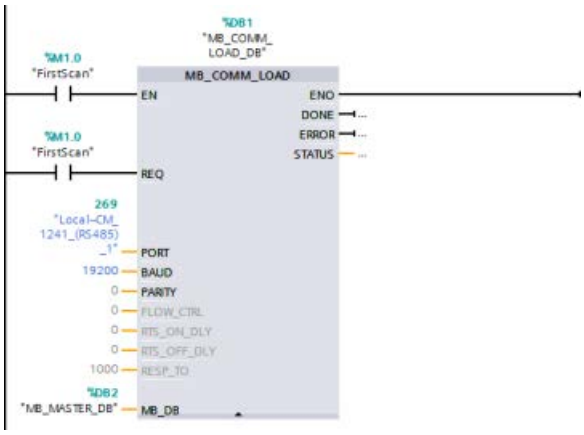
[1] 除了上述列出的 MB_SLAVE 错误，还可能返回底层 PtP 通信指令的错误。

13.9.4 早期 Modbus RTU 示例

13.9.4.1 示例： 早期 Modbus RTU 主站程序

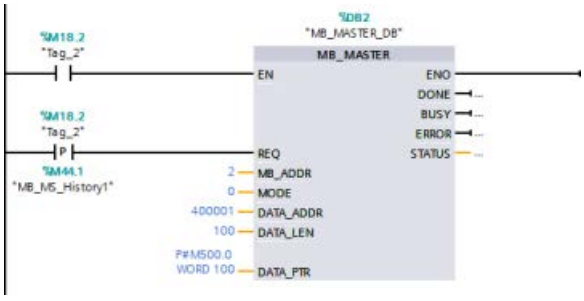
启动期间通过第一个扫描标志初始化 MB_COMM_LOAD。通过此方式执行 MB_COMM_LOAD 时，必须保证串口组态在运行时不会更改。

程序段 1：仅在第一次扫描期间对 RS485 模块通信端口进行一次组态/初始化。

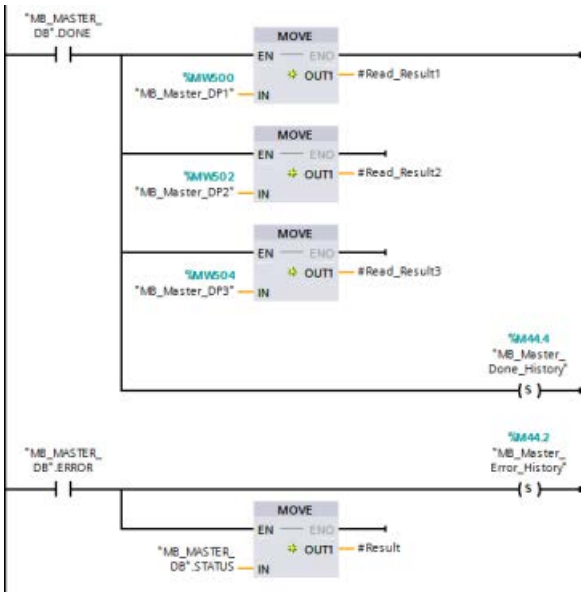


在程序循环 OB 中使用一个 MB_MASTER 指令，以与单个从站进行通信。要与其它从站通信，可在程序循环 OB 中使用额外的 MB_MASTER 指令，也可以重新使用一个 MB_MASTER FB。

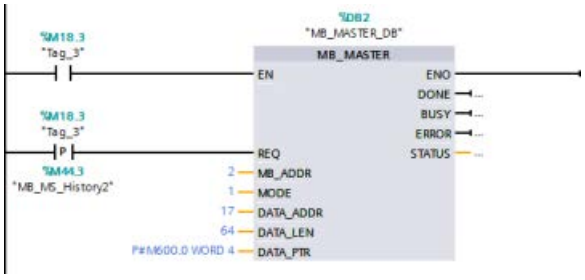
程序段 2：从从站 #2 上的位置 400001 读取 100 个字保持寄存器数据到存储器位置 MW500-MW698。



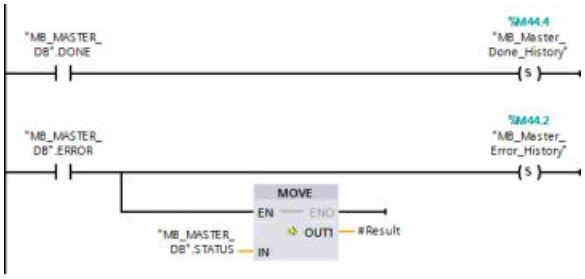
程序段 3：移动读取到其他位置的保持寄存器数据的前三个字，然后置位 DONE 历史位。一旦发生错误，该程序段还将置位错误历史位并将状态字保存到另一个位置。



程序段 4：将 MW600-MW607 的 64 位数据写入从站 #2 上的 00017 到 00081 输出位位置。



程序段 5：写入完成后置位 DONE 历史位。一旦发生错误，该程序段将置位错误历史位并保存状态代码。

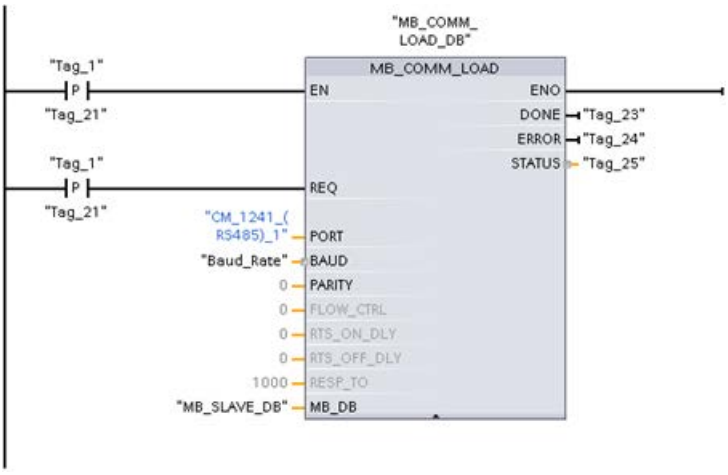


13.9.4.2 示例： 早期 Modbus RTU 从站程序

每次启用“Tag_1”启用时，初始化下面显示的 MB_COMM_LOAD。

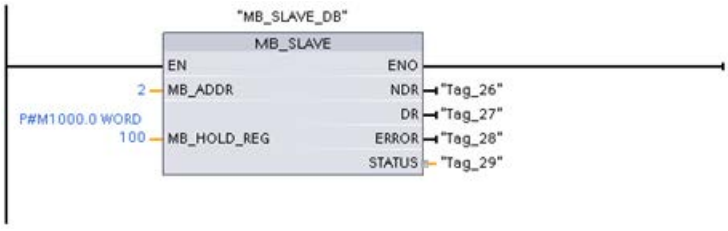
通过此方式执行 MB_COMM_LOAD 时，必须保证串口组态在运行时会根据 HMI 配置进行更改。

程序段 1：每次 HMI 设备更改 RS485 模块参数时，都会初始化该参数。



下面显示的 MB_SLAVE 置于每 10 ms 执行一次的循环 OB 中。尽管这样不会使从站的绝对响应速度达到最快，但却可使短消息（在请求中占 20 字节或更低）达到 9600 波特的良好性能。

程序段 2：每次扫描期间检查 Modbus 主站请求。Modbus 保持寄存器被组态为 100 个字（从 MW1000 开始）。



13.10 工业远程通信 (IRC)

13.10.1 远程控制通信处理器概述

工业远程通信提供对分布广泛的机器、设备和尺寸不尽相同的应用程序的安全、经济的访问权限。工业远程通信包括通过 CP 模块的通信方式：

- TeleControl：遥控用于将分布在广泛地理区域的各个过程站（远程终端单元/RTU）连接至一个或多个集中过程控制系统，以便进行监视和控制。远程网络产品频谱中的各种传输组件支持通过一系列公共和专用网络进行远程通信。特殊远程通信协议执行事件驱动或周期性交换过程数据，从而实现对总体过程的有效控制。
- TeleService：远程服务包括和远距离技术系统（机器、设备、计算机等）进行数据交换，从而实现发现错误、诊断、维护、修理或优化的目的。
- 远程通信的附加应用，如监视、智能电网应用和条件监视。

用于 S7-1200 的远程控制通信处理器

对于 TeleControl 应用，提供以下通信处理器，这些处理器大部分提供对 S7-1200 Web 服务器 (页 1033)的访问权限：

- CP 1243-1：
  - 部件号： 6GK7 243-1BX30-0XE0
  - 用于将 SIMATIC S7-1200 连接到控制中心（采用公共基础设施，例如 DSL，并通过 TeleControl Server Basic，TCSB 版本 V3）的通信处理器
  - 借助于 VPN 技术和防火墙，可通过通信处理器以受保护的方式访问 S7-1200。
  - 可将通信处理器用作 CPU 进行 S7 通信的附加以太网接口。
  - 通信处理器与 CPU 之间的通信通过可访问 PLC 变量的可组态数据点进行。
- CP 1243-1 DNP3：
  - 部件号： 6GK7 243-1JX30-0XE0
  - 用于通过 DNP3 协议将 SIMATIC S7-1200 连接到控制中心的通信处理器
  - 通信处理器与 CPU 之间的通信通过可访问 PLC 变量的可组态数据点进行。
- CP 1243-1 IEC：
  - 部件号： 6GK7 243-1PX30-0XE0
  - 用于通过 IEC 60870-5 协议将 SIMATIC S7-1200 连接到控制中心的通信处理器
  - 通信处理器与 CPU 之间的通信通过可访问 PLC 变量的可组态数据点进行。

- CP 1243-1 PCC：
  - 订货号： 6GK7 243-1HX30-0XE0
  - 用于通过设备云通信 (PCC) 将 SIMATIC S7-1200 连接到控制中心的通信处理器
  - 通信处理器与 CPU 之间的通信通过可访问 PLC 变量的可组态数据点进行。
- CP 1242-7：
  - 订货号： 6GK7 242-7KX31-0XE0
  - 用于通过移动无线 (GPRS) 和公共基础设施 (DSL) 以及 TeleControl Server Basic 将 SIMATIC S7-1200 连接到控制中心的通信处理器
- CP 1242-7 GPRS V2：
  - 部件号： 6GK7 242-7KX31-0XE0
  - 用于通过移动无线 (GPRS) 和公共基础设施 (DSL) 以及 TeleControl Server Basic (TCSB V3) 将 SIMATIC S7-1200 连接到控制中心的通信处理器
  - 借助于 VPN 技术和防火墙，可通过通信处理器以受保护的方式访问 S7-1200。
  - 可将通信处理器用作 CPU 进行 S7 通信的附加以太网接口。
  - 通信处理器与 CPU 之间的通信通过可访问 PLC 变量的可组态数据点进行。
- CP 1243-7 LTE-xx：
  - 用于通过移动无线 (GPRS) 和公共基础设施 (DSL) 以及 TeleControl Server Basic (TCSB V3) 将 SIMATIC S7-1200 连接到控制中心的通信处理器
  - 支持以下移动无线规范： GSM/GPRS, UMTS (G3), LTE
  - 为了覆盖采用不同移动无线规范的国家/地区，此通信处理器分两个型号提供：
    - CP 1243-7 LTE-US：
      - 北美标准
      - 部件号： 6GK7 243-7SX30-0XE0
    - CP 1243-7 LTE-EU：
      - 西欧标准
      - 部件号： 6GK7 243-7KX30-0XE0
  - 借助于 VPN 技术和防火墙，可通过通信处理器以受保护的方式访问 S7-1200。
  - 可将通信处理器用作 CPU 进行 S7 通信的附加以太网接口。
  - 通信处理器与 CPU 之间的通信通过可访问 PLC 变量的可组态数据点进行。
- CP 1243-8 IRC：
  - 订货号： 6GK7 242-8RX30-0XE0
  - 用于将 SIMATIC S7-1200 连接 ST7 网络、数据点组态和 VPN 的通信处理器

说明

必须将 TeleControl Server Basic 软件用于除 CP 1243-1 之外 CP 的 TeleControl 应用程序。

安全通信

经过良好验证的 SINAUT ST7 协议或标准化的 DNP3 或 IEC 60870-5 协议增加了工业远程通信的安全性 (http://w3app.siemens.com/mcms/infocenter/dokumentencenter/sc/ic/InfocenterLanguagePacks/Netzwerksicherheit/6ZB5530-1AP02-0BA4_BR_Network_Security_en_112015.pdf)。TeleControl 解决方案提供综合措施，防止数据被篡改或丢失。每个传输模块具备较大存储区用于存储数千数据帧，从而提供在传输链接中桥接停机时间的能力。专门的 VPN 解决方案保护特殊的基于 IP 的网络。

CP 1243-1 通信处理器将 SIMATIC S7-1200 控制器和以太网网络安全地进行连接。通过其综合性防火墙（状态检测）和 VPN 协议 (IPsec) 安全功能，通信处理器帮助保护 S7-1200 站和较低等级的网络免受未经许可的访问，并通过加密技术保护数据传输不受操控和窥探。此外，可以使用 CP，通过基于 IP 的远程网络，将 S7-1200 站集成到 TeleControl Server Basic 控制中心软件中。

13.10.2 连接到 GSM 网络

通过 GPRS 且基于 IP 的 WAN 通信

可以使用 CP 1242-7 通信处理器将 S7-1200 连接到 GSM 网络。 使用 CP 1242-7 可以实现远程站与控制中心的 WAN 通信，以及站间通信。

站间通信只能通过 GSM 网络实现。 要在远程站和控制室之间进行通信，控制中心必须具备可以访问 Internet 的 PC。

CP 1242-7 支持通过 GSM 网络的以下通信服务：

- GPRS（General Packet Radio Service，通用分组无线服务）
  通过 GSM 网络处理用于数据传输“GPRS”且面向数据包的服务。
- SMS（Short Message Service，短消息服务）
  CP 1242-7 可以接收和发送 SMS 消息。 通信伙伴可以是移动电话或 S7-1200。

CP 1242-7 适合在世界各地的工业领域使用，并且支持以下频段：

- 850 MHz
- 900 MHz
- 1,800 MHz
- 1,900 MHz

要求

站或控制中心使用的设备取决于具体的应用。

- 要与中央控制室通信或者要通过中央控制室进行通信，控制中心需要具备可以访问 Internet 的 PC。
- 除站设备之外，具有 CP 1242-7 的远程 S7-1200 站必须满足以下要求才能通过 GSM 网络进行通信：
  - 与相应的 GSM 网络供应商签订了合约
    如果使用 GPRS，合约中必须允许使用 GPRS 服务。
    如果存在仅通过 GSM 网络的站间直接通信，则 GSM 网络供应商必须为 CP 分配固定的 IP 地址。 在这种情况下，站间通信不经过控制中心。
  - 合约中包含 SIM 卡
    SIM 卡已插入 CP 1242-7 中。
  - 可在站范围内本地使用 GSM 网络

13.10.3 CP 1242-7 的应用

CP 1242-7 可用于以下应用：

遥控应用

- 通过 SMS 发送消息
  通过 CP 1242-7，远程 S7-1200 站的 CPU 可以接收来自 GSM 网络的 SMS 消息，或者通过 SMS 向已组态的移动电话或 S7-1200 发送消息。
- 与控制中心的通信
  远程 S7-1200 站通过 GSM 网络和 Internet 与主站中的遥控服务器进行通信。 为了能够使用 GPRS 传输数据，需在主站的遥控服务器上安装“TELECONTROL SERVER BASIC”应用程序。 遥控服务器使用集成的 OPC 服务器功能与更高层级的中央控制系统通信。
- S7-1200 站间通过 GSM 网络进行的通信
  可通过两种不同的方式处理配有 CP 1242-7 的远程站之间的通信：
  - 通过主站进行站间通信
    在此组态的主站中，将要在各个 S7-1200 站之间建立一个永久的安全连接，便于各个站之间相互通信以及与遥控服务器之间进行通信。 站间通信通过遥控服务器进行。
    CP 1242-7 在“Telecontrol”模式下运行。
  - 站间直接通信
    对于不经过主站的站间直接通信，使用带有固定 IP 地址的 SIM 卡，以便各站直接互相寻址。 可能的通信服务和安全功能（例如 VPN）取决于网络供应商所提供的服务。
    CP 1242-7 在“GPRS 直接”模式下运行。

通过 GPRS 的 TeleService

可通过 GSM 网络和 Internet 在配有 STEP 7 的工程师站与配有 CP 1242-7 的远程 S7-1200 站之间建立 TeleService 连接。 该连接从工程师站通过遥控服务器或 TeleService 网关（用作中介转发帧并建立相应授权）运行。 这些 PC 使用“TELECONTROL SERVER BASIC”应用程序的功能。

可以将 TeleService 连接用于以下方面：

- 将组态或程序数据由 STEP 7 项目下载到工作站
- 查询工作站中的诊断数据

13.10.4 CP 1242-7 的其它属性

CP 1242-7 的其它服务和功能

- 通过 Internet 执行 CP 的日时钟同步
  可以按照以下方法设置 CP 的时间：
  – 在“Telecontrol”模式下，由遥控服务器传送时间。CP 使用该时间来设置其自身的时间。
  – 在“GPRS 直接” 模式下，CP 可以使用 SNTP 请求时间。
  为同步 CPU 时间，可以用块从 CP 读出当前时间。
- 在存在连接问题时临时缓冲要发送的消息
- 由于可以连接到备用遥控服务器，提高了可用性
- 记录数据量
  记录已传送的数据量，并根据特定需要进行评估。

13.10.5 更多信息

CP 手册、相关文档和产品信息文档提供更多详细信息：

- CP 1242-7 (http://support.automation.siemens.com/WW/view/zh/45605894)
- CP 1243-7 LTE (https://support.industry.siemens.com/cs/cn/zh/ps/15924)
- CP 1243-1 DNP3 (https://support.industry.siemens.com/cs/cn/zh/ps/15938)
- CP 1243-8 IRC (https://support.industry.siemens.com/cs/cn/zh/ps/21162)
- CP 1243-1 IEC (https://support.industry.siemens.com/cs/cn/zh/ps/15942)
- 可进行的固件更新 (https://support.industry.siemens.com/cs/cn/zh/view/109482530/en)

13.10.6 附件

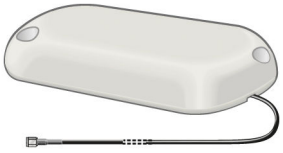
ANT794-4MR GSM/GPRS 天线

下列天线可用于 GSM/GPRS 网络，且安装在室内和室外均可：

- 四频天线 ANT794-4MR (http://support.automation.siemens.com/WW/view/zh/23119005)

简称	订货号	说明
ANT794-4MR	6NH9 860-1AA00	四频天线（900，1800/1900 MHz，UMTS）；防水；适合室内和室外使用；永久连接到天线的 5 m 连接电缆；SMA 连接器，包括安装支架、螺钉、墙用插座

- 平头天线 ANT794-3M



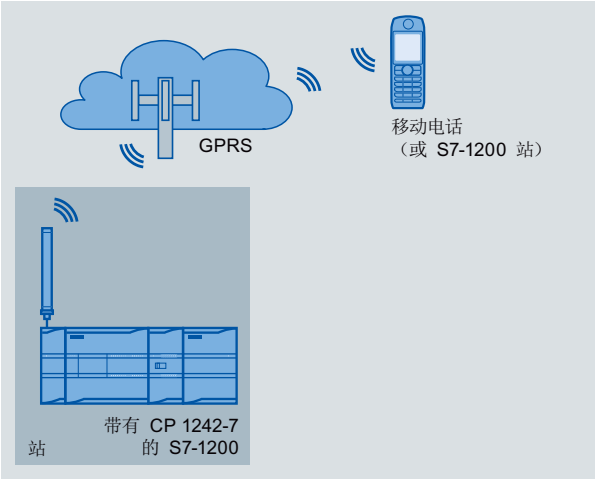
简称	订货号	说明
ANT794-3M	6NH9 870-1AA00	平头天线（900，1800/1900 MHz）；防水；适合室内和室外使用；永久连接到天线的 1.2 m 连接电缆；SMA 连接器，包括粘胶垫、可采用螺钉安装

必须单独订购天线。

13.10.7 遥控组态示例

下文针对配有 CP 1242-7 的站提供了几个组态示例。

通过 SMS 发送消息



配有 CP 1242-7 的 SIMATIC S7-1200 可以通过 SMS 向移动电话或已组态的 S7-1200 站发送消息。

通过控制中心进行遥控

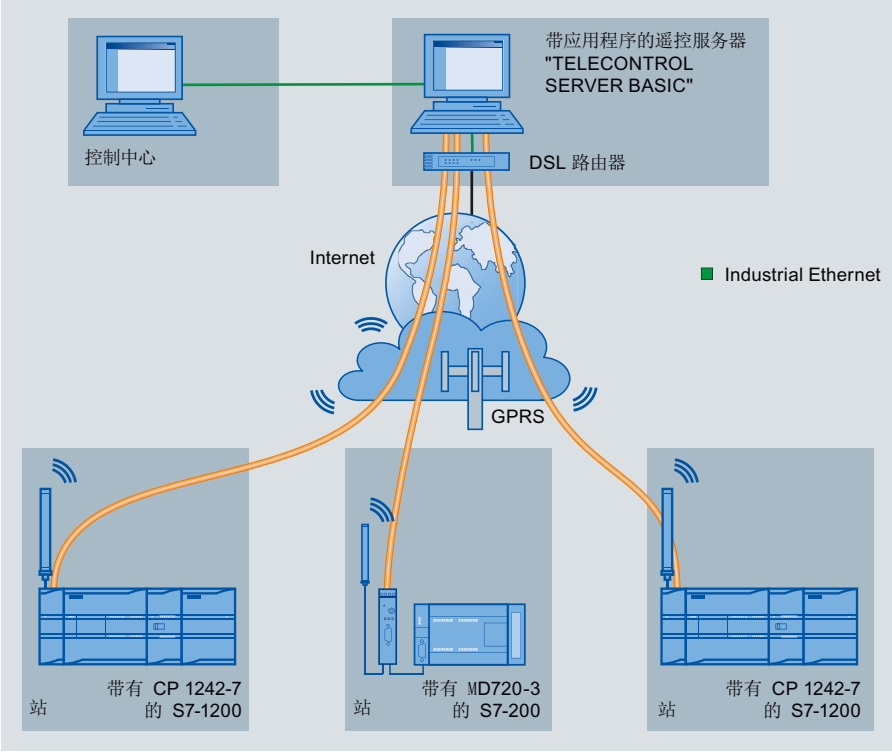


图 13-29　S7-1200 站与控制中心之间的通信

在遥控应用中，配有 CP 1242-7 的 SIMATIC S7-1200 通过 GSM 网络和 Internet 与控制中心通信。 “TELECONTROL SERVER BASIC”(TCSB) 应用程序安装在主站的遥控服务器上。 如此可实现以下应用：

- 工作站和控制中心之间的遥控通信
  在该应用中，由工作站通过 GSM 网络和 Internet 将现场数据发送至主站的遥控服务器。遥控服务器用于监控远程站。
- 工作站和装有 OPC 客户端的控制室之间的通信
  与第一种应用类似，工作站与遥控服务器进行通信。 使用集成的 OPC 服务器，遥控服务器与控制室中的 OPC 客户端交换数据。
  OPC 客户端和遥控服务器可位于一台计算机上，例如，当装有 WinCC 的控制中心计算机上安装了 TCSB 时，OPC 客户端和遥控服务器便可同时位于该计算机上。
- 通过控制中心进行站间通信
  站间通信可通过配有 CP 1242-7 的 S7 站实现。
  为了允许进行站间通信，遥控服务器会将发送站的消息转发到接收站。

站间直接通信

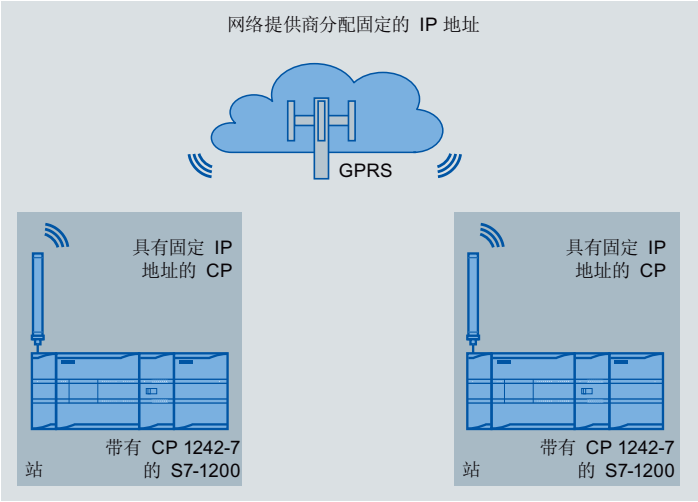


图 13-30　两个 S7-1200 站间直接通信

在此组态中，两个 SIMATIC S7-1200 站使用 CP 1242-7 通过 GSM 网络互相直接通信。 每个 CP 1242-7 都有一个固定 IP 地址。 GSM 网络供应商的相关服务必须允许这项规定。

通过 GPRS 的 TeleService

在通过 GPRS 的 TeleService 中，安装有 STEP 7 的工程师站通过 GSM 网络和 Internet 与 S7-1200 中的 CP 1242-7 进行通信。

由于防火墙对外部连接请求处于常闭状态，因此远程站和工程师站之间需要一个开关站。 此开关站可以是一个遥控服务器，如果组态中没有遥控服务器，此开关站也可以是 TeleService 网关。

使用遥控服务器的 TeleService

连接通过遥控服务器运行。

- 工程师站和遥控服务器通过 Intranet (LAN) 或 Internet 进行连接。
- 遥控服务器和远程站通过 Intranet 或 GSM 网络进行连接。

工程师站和遥控服务器也可以是同一台计算机；换言之，也可将 STEP 7 和 TCSB 安装在同一台计算机上。

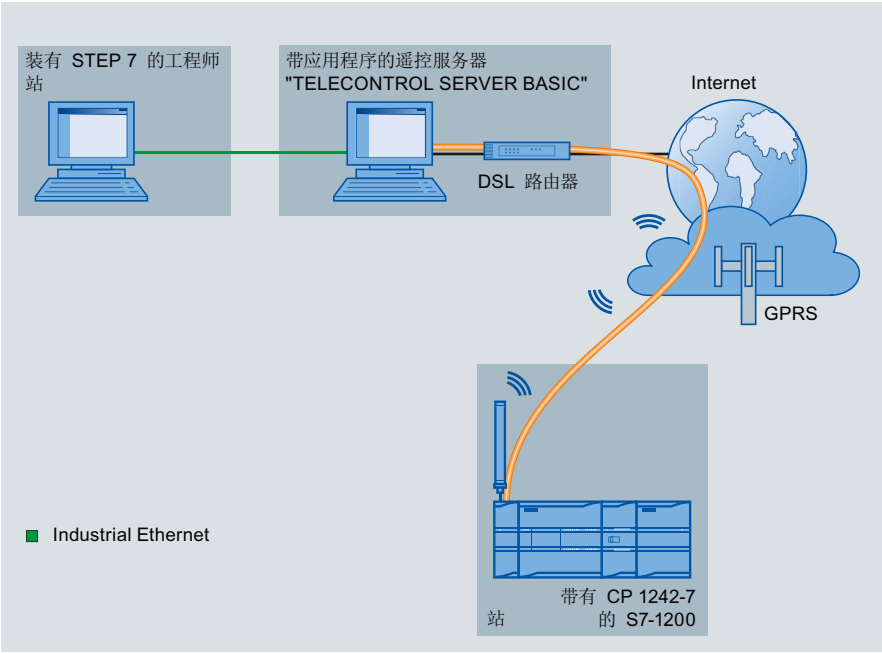


图 13-31　在使用遥控服务器的组态中通过 GPRS 的 TeleService

不使用遥控服务器的 TeleService

连接通过 TeleService 网关运行。

工程师站和 TeleService 网关间的连接可以是通过 LAN 或 Internet 运行的本地连接。

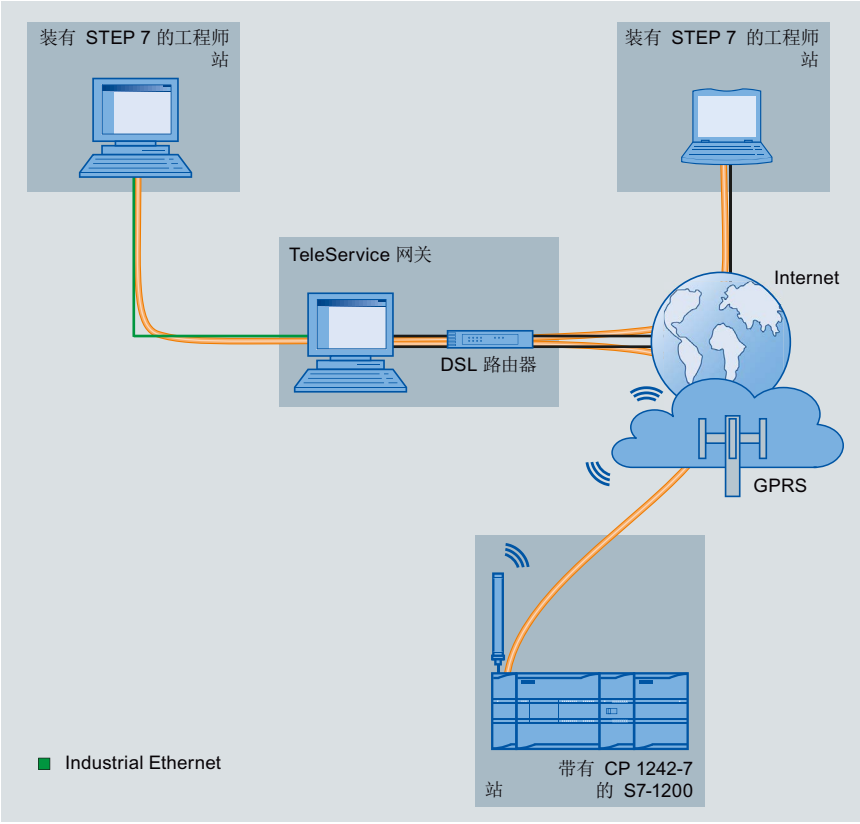
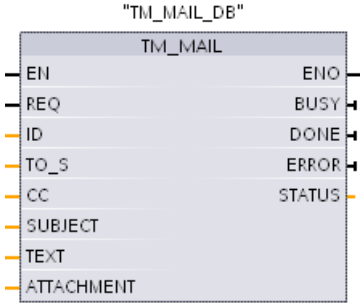


图 13-32　在使用 TeleService 网关的组态中通过 GPRS 的 TeleService

TeleService 通信（SMTP 电子邮件） 14

14.1 TM_Mail（发送电子邮件）指令

表格 14-1 TM_MAIL 指令

LAD/FBD	SCL	说明
	**"TM_MAIL_DB"(** **REQ:=_bool_in_,** **ID:=_int_in_,** **TO_S:=_string_in_,** **CC:=_string_in_,** **SUBJECT:=_string_in_,** **TEXT:= _string_in_,** **ATTACHMENT:=_variant_in_,** **BUSY=>_bool_out_,** **DONE=>_bool_out_,** **ERROR=>_bool_out_,** **STATUS=>_word_out_,);**	TM_MAIL 指令通过 CPU 工业以太网连接使用 TCP/IP 上的 SMTP（Simple Mail Transfer Protocol, 简单邮件传输协议）发送电子邮件消息。其中基于以太网的 Internet 连接性不可用，可选的 TeleService 适配器可用于与电话陆线的连接。TM_MAIL 会异步执行，并且该作业会持续多次 TM_MAIL 调用。调用 TM_MAIL 时，必须分配背景数据块。**绝不可设置该背景数据块的保持性属性**。如此可以确保背景数据块在 CPU 由 STOP 模式切换到 RUN 模式时初始化，以及可以触发新的 TM_MAIL 操作。

[1] STEP 7 会在插入指令时自动创建背景 DB。

当输入参数 REQ 出现上升沿（从 0 变为 1）时，开始发送电子邮件。下表给出了 BUSY、DONE 和 ERROR 之间的关系。可在连续调用期间评估这些参数来监视 TM_MAIL 执行的进度和检查完成情况。

当输出参数 BUSY 的状态由 1 变为 0 时，输出参数 DONE、ERROR、 STATUS 和 SFC_STATUS 只在一个循环周期内有效。程序逻辑必须保存临时输出状态值，以便能检测到后续程序执行周期中的状态变化。

说明

TM_MAIL 使用 CPU 的以太网接口通过 TCP/IP 发送邮件消息。要通过 CP 接口（具备或不具备 SSL）发送邮件消息，请使用指令 TMAIL_C（通过 CPU 的以太网接口发送电子邮件）指令 (页 889)。

表格 14-2 Done、Busy 和 Error 参数之间的交互作用

DONE	BUSY	ERROR	说明
不相关	1	不相关	正在处理作业。
1	0	0	作业已成功完成。
0	0	1	作业因出错而终止。有关错误原因的信息，请参见 STATUS 参数。
0	0	0	没有作业正在处理

如果 CPU 在 TM_MAIL 激活期间切换到 STOP 模式，则将终止与电子邮件服务器之间的通信连接。如果通过工业以太网总线进行 CPU 通信时出现问题，那么还将丢失与电子邮件服务器的通信连接。发生这些情况时，将暂停发送过程，同时接收方也收不到电子邮件。

注意

修改用户程序

程序块的删除和替换、对 TM_MAIL 的调用或者对 TM_MAIL 背景数据块的调用都会中断程序块的链接。如果未能保持已链接的程序块，则 TPC/IP 通信功能将进入不确定状态，进而可能导致财产损失。传送修改后的程序块之后，必须执行 CPU 重启（热启动）或冷启动。

为避免中断程序块的链接，仅在下列情况下更改用户程序中直接影响 TM_MAIL 调用的部分：

- CPU 处于 STOP 模式
- 未发送任何电子邮件（REQ 和 BUSY = 0）

数据一致性

在启动操作时会读取输入参数 ADDR_MAIL_SERVER。只有在当前操作完成并且启动新的 TM_MAIL 操作后，新值才会生效。

相反，参数 WATCH_DOG_TIME、TO_S、CC、FROM、SUBJECT、TEXT、ATTACHMENT、USERNAME 和 PASSWORD 将在执行 TM_MAIL 时被读取，并且仅在完成作业 (BUSY = 0) 后才可更改。

拨号连接：组态 TS 适配器的 IE 参数。

必须组态离开调用的 TeleService 适配器 IE 参数，以便与 Internet 服务提供商的拨号服务器相连接。如果设置了调用的“按需”属性，则仅在发送电子邮件时建立连接。对于模拟调制解调器连接，连接过程需要更多的时间（大约多出一分钟）。必须将额外的时间包括到 WATCH_DOG_TIME 值中。

表格 14-3　参数的数据类型

参数和类型		数据类型	说明
REQ	IN	Bool	通过由低到高的（上升沿）信号启动操作。
ID	IN	Int	连接标识符：请参见指令 TCON、TDISCON、TSEND 和 TRCV 的 ID 参数。 必须使用未在用户程序中用于该指令的任何其它实例的编号。
TO_S	IN	String	收件人地址：最大长度为 240 个字符的 STRING 数据
CC	IN	String	抄送收件人地址（可选）：最大长度为 240 个字符的 STRING 数据
SUBJECT	IN	String	电子邮件的主题名：最大长度为 240 个字符的 STRING 数据。
TEXT	IN	String	电子邮件的文本消息（可选）：最大长度为 240 个字符的 STRING 数据。 如果此参数是空字符串，则发送的电子邮件将不含任何消息文本。
ATTACHMENT	IN	Variant	指向电子邮件附件数据的指针：最大长度为 65534 字节的字节、字或双字数据。 如果未分配任何值，则发送的电子邮件不含附件。
DONE	OUT	Bool	• 0 - 作业尚未启动或仍在执行。 • 1 - 作业已执行，未出现错误。
BUSY	OUT	Bool	• 0 - 无操作正在进行 • 1 - 操作正在进行
ERROR	OUT	Bool	上一请求因错误而终止后，ERROR 位将在一个扫描周期的时间内保持为 1。STATUS 输出中的错误代码值仅在 ERROR = 1 的一个扫描周期内有效。
STATUS	OUT	Word	TM_MAIL 指令的返回值或错误信息。

参数和类型		数据类型	说明
ADDR_MAIL_SERVER	[1] Static	DWord	邮件服务器的 IP 地址：必须将每个 IP 地址片段分配为两个 4 位十六进制字符组成的 8 位位组。如果 IP 地址片段 = 等于十六进制值 A 的十进制值 10，则必须为该 8 位位组输入“0A”。 例如：IP 地址 = 192.168.0.10 ADDR_MAIL_SERVER = DW#16#C0A8000A，其中： • 192 = 16#C0， • 168 =16#A8 • 0 = 16#00 • 10 = 16#0A
WATCH_DOG_TIME	[1] Static	Time	TM_MAIL 完成整个 SMTP 过程（从连接到 SMTP 开始到完成 SMTP 传输）所允许的最长时间。如果超出该时间，TM_MAIL 结束执行并报告错误。 在 TM_MAIL 结束并报告错误之前的实际时间延时可能超过 WATCH_DOG_TIME，这是因为断开操作需要更多的时间。 开始时，应设置 2 分钟时间。对于 ISDN 电话连接，该时间可以短很多。
USERNAME	[1] Static	String	邮件帐户的用户名：最大长度为 180 个字符的 STRING 数据。
PASSWORD	[1] Static	String	邮件服务器密码：最大长度为 180 个字符的 STRING 数据。
FROM	[1] Static	String	发送方地址：最大长度为 240 个字符的 STRING
SFC_STATUS	[1] Static	Word	被调用通信块的执行条件代码

[1] 每次调用 TM_MAIL 时都不会修改这些参数的值。值分配在 TM_MAIL 实例数据块中，并且在首次调用 TM_MAIL 时，并不只被引用一次，

SMTP 验证

TM_MAIL 支持 SMTP AUTH LOGIN 验证方法。有关该验证方法的信息，请参见邮件服务器手册或 Internet 服务提供商的网站。

AUTH LOGIN 验证方法使用 TM_MAIL、USERNAME 和 PASSWORD 参数连接邮件服务器。以前必须在邮件服务器上设置电子邮件帐户的用户名和密码。

如果没有为 USERNAME 参数分配任何值，则不会使用 AUTH LOGIN 验证方法，并且电子邮件将在没有验证的情况下发送。

TO_S:、CC: 和 FROM:参数

参数 TO_S:、CC: 和 FROM: 是字符串，如下面的示例所示：

TO: <wenna@mydomain.com>, <ruby@mydomain.com>,

CC: <admin@mydomain.com>, <judy@mydomain.com>,

FROM: <admin@mydomain.com>

输入这些字符串时必须遵守以下规则：

- 必须输入“TO:”、“CC:”和“FROM:”字符（包括冒号）。
- 在每个地址前必须输入空格字符和起始尖括号“<”。例如，在“TO:”和 <电子邮件地址> 之间必须有空格字符。
- 在每个地址后必须输入结束尖括号“>”。
- 在 TO_S: 和 CC: 地址中的每个电子邮件地址后必须输入逗号字符“,”。例如，单个电子邮件地址后的逗号在“TO: <email address>,”中是必填项。
- FROM: 条目只能使用一个电子邮件地址，并且末尾不能有逗号。

考虑到运行模式和存储器的使用，不会对 TM_MAIL 的 TO_S:、CC: 和 FROM: 数据执行语法检查。如果未严格遵照上述格式规则。SMTP 电子邮件服务器事务将会失败。

STATUS 和 SFC_STATUS 参数

TM_MAIL 返回的执行条件代码可分为以下几类：

- W#16#0000：TM_MAIL 操作已成功完成
- W#16#7xxx：TM_MAIL 操作状态
- W#16#8xxx：内部调用通信设备或邮件服务器时出错

下表显示了 TM_MAIL 的执行条件代码，但不包括内部调用通信模块时生成的错误代码。

说明

电子邮件服务器要求

TM_MAIL 只能通过端口 25 与使用 SMTP 的电子邮件服务器通信。分配的端口号不能更改。

大多数 IT 部门和外部电子邮件服务器现在都禁用了端口 25 以防止 PC 受病毒感染而变为欺诈电子邮件生成器。

您可通过 SMTP 连接内部邮件服务器，并让内部服务器管理当前安全强化，该安全强化是通过 Internet 将电子邮件转发到外部邮件服务器所必需的功能。

示例：内部邮件服务器组态

如果将 Microsoft Exchange 用作内部邮件服务器，则可以配置服务器以使 SMTP 通过分配了 S7-1200 PLC 的 IP 地址访问。配置交换管理控制台：“服务器组态”(Server configuration) >“集线器传输”(Hub transport) >“接收连接器”(Receive connectors) >“IP 转发”(IP relay)。在“网络”(Network) 选项卡上，有名为“从具有这些 IP 地址的远程服务器接收邮件”(Receive mail from remote servers that have these IP addresses) 的框。您可在此处输入执行 TM_MAIL 指令的 PLC 的 IP 地址。该类使用内部 Microsoft Exchange 服务器的连接无需验证。

电子邮件服务器配置

TM_MAIL 只能使用允许端口 25 通信、SMTP 和 AUTH LOGIN 验证（可选）的电子邮件服务器。

组态兼容的电子邮件服务器帐户以接受远程 SMTP 登录。然后编辑 TM_MAIL 的实例 DB 以输入 TM_MAIL USERNAME 和 PASSWORD 字符串，这些字符串用于验证与您的电子邮件帐户的连接。

表格 14-4　条件代码

STATUS (W#16#...)：	SFC_STATUS (W#16#...)：	说明
0000	-	TM_MAIL 操作已完成，且未发生错误。这个零 STATUS 代码不能保证电子邮件确实已发送（请参见此表后的第一条注释）。
7001	-	TM_MAIL 处于激活状态 (BUSY = 1)。
7002	7002	TM_MAIL 处于激活状态 (BUSY = 1)。

STATUS (W#16#...)：	SFC_STATUS (W#16#...)：	说明
8xxx	xxxx	TM_MAIL 操作已完成，但内部调用通信指令时出错。有关 SFC_STATUS 参数的详细信息，请参见底层 PROFINET 开放式用户通信指令的 STATUS 参数说明。
8010	xxxx	连接失败：有关 SFC_STATUS 参数的详细信息，请参见 TCON 指令的 STATUS 参数说明。
8011	xxxx	发送数据时出错：有关 SFC_STATUS 参数的详细信息，请参见 TSEND 指令的 STATUS 参数说明。
8012	xxxx	接收数据时出错：有关 SFC_STATUS 参数的详细信息，请参见 TRCV 指令的 STATUS 参数说明。
8013	xxxx	连接失败：有关评估 SFC_STATUS 参数的详细信息，请参见 TCON 和 TDISCON 指令的 STATUS 参数说明。
8014	-	连接失败：可能输入了错误的邮件服务器 IP 地址 (ADDR_MAIL_SERVER) 或过短的连接时间 (WATCH_DOG_TIME)。也可能是 CPU 未与网络连接或 CPU 组态不正确。
8015	-	ATTACHMENT 参数的指针无效：使用具有数据类型和长度分配的 variant 指针。例如，“P#DB.DBX0.0”不正确，“P#DB.DBX0.0 byte 256”正确。
82xx, 84xx, 85xx	-	错误消息来自邮件服务器且对应于 SMTP 协议的错误编号“8”。请参见此表后的第二条注释。
8450	-	操作未执行：邮箱不可用；请稍后重试。
8451	-	操作已中止：处理过程中出现本地错误，请稍后重试。
8500	-	命令语法错误：原因可能是电子邮件服务器不支持 LOGIN 验证过程。请检查 TM_MAIL 的参数。尝试发送无需验证的电子邮件。尝试用空字符串替换参数 USERNAME。
8501	-	语法错误：参数不正确；可能在 TO_S 或 CC 参数中输入了错误地址。
8502	-	未知的命令或命令未执行：请检查输入的内容，尤其是参数 FROM。可能是输入不完整，漏掉了“@”或“.”字符。
8535	-	SMTP 验证不完整。输入的用户名或密码可能不正确。
8550	-	无法访问邮件服务器，或您没有访问权限。输入的用户名或密码可能不正确，或者邮件服务器不支持登录访问。该错误的另一个原因可能是在 TO_S 或 CC 参数中字符“@”后面输入的域名不正确。
8552	-	操作已中止：超出分配的存储器大小；请稍后重试。
8554	-	传输失败：请稍后重试。

说明

可能未报告的电子邮件传输错误

- 收件人地址输入不正确不会令 TM_MAIL 产生 STATUS 错误。在这种情况下，无法保证其他具有正确电子邮件地址的收件人能收到电子邮件。
- 有关 SMTP 错误代码的详细信息，请访问 Internet 或参见邮件服务器的错误文档。也可以从邮件服务器读取最后一条错误消息。该错误消息存储在 TM_MAIL 背景数据块的参数 buffer1 中。

在线和诊断工具 15

15.1 状态 LED

CPU 和 I/O 模块使用 LED 提供有关模块或 I/O 的运行状态的信息。

CPU 上的状态 LED

CPU 提供以下状态指示灯：

- STOP/RUN
  - 黄色常亮指示 STOP 模式
  - 纯绿色指示 RUN 模式
  - 闪烁（绿色和黄色交替）指示 CPU 处于 STARTUP 运行状态
- ERROR
  - 呈红色闪烁即表示出错，例如 CPU 内部出错、存储卡出错或组态出错（不匹配模块）
  - 闪烁红色三秒表示当前错误未持续。例如，实时时钟 (RTC) 会在断电时重置为默认时间。
  - 故障状态：
    - 纯红色指示硬件出现故障
    - 如果固件检测到故障，则所有 LED 闪烁
- MAINT（维护）在每次插入存储卡时闪烁。重新启动 CPU。然后 CPU 切换到 STOP 模式。在 CPU 切换到 STOP 模式后，执行以下操作之一以启动存储卡评估：
  - 将 CPU 切换到 RUN 模式
  - 执行存储器复位 (MRES)
  - CPU 循环上电

也可使用 LED 指令 (页 451)来确定 LED 的状态。

表格 15-1 CPU 上的状态 LED

说明	STOP/RUN 黄色/绿色	ERROR 红色	MAINT 黄色
断电	灭	灭	灭
启动、自检或固件更新	闪烁 （黄色和绿色交替）	-	灭
停止模式	亮（黄色）	-	-

说明	STOP/RUN 黄色/绿色	ERROR 红色	MAINT 黄色
运行模式	亮（绿色）	-	-
取出存储卡	亮（黄色）	-	闪烁
错误	亮（黄色或绿色）	闪烁	-
请求维护 ● 强制 I/O ● 需要更换电池（如果安装了电池板）	亮（黄色或绿色）	-	亮
硬件出现故障	亮（黄色）	亮	灭
LED 测试或 CPU 固件出现故障	闪烁 （黄色和绿色交替）	闪烁	闪烁
CPU 组态版本未知或不兼容	亮（黄色）	闪烁	闪烁

说明

“CPU 组态版本未知或不兼容”错误

试图将 S7-1200 V3.0 程序下载到 S7-1200 V4.0 CPU 中会导致 CPU 错误，CPU 将在诊断缓冲区显示相应错误消息。如果是因使用了无效版本的程序传送卡 (页 137)所致，则请取出该卡，然后执行 STOP 到 RUN 切换、存储器复位 (MRES) 或循环上电。如果是因下载了无效程序所致，则请将 CPU 复位为其出厂设置 (页 1389)。将 CPU 从错误状态恢复后，即可下载有效的 V4.3 CPU 程序。

CPU 还提供了两个可指示 PROFINET 通信状态的 LED。打开底部端子块的盖子可以看到 PROFINET LED。

- Link （绿色）点亮指示连接成功
- Rx/Tx （黄色）点亮指示传输活动

CPU 和各数字量信号模块 (SM) 为每个数字量输入和输出提供了 I/O Channel LED。I/O Channel （绿色）通过点亮或熄灭来指示各输入或输出的状态。

出现致命错误之后的 S7-1200 特性

CPU 固件在检测到致命错误时会尝试故障模式重新启动，如果重新启动成功，CPU 会通过持续闪烁 STOP/RUN、ERROR 和 MAINT LED 发出信号来指示故障模式。不能在故障模式重新启动后装载用户程序和硬件配置。

如果 CPU 成功完成故障模式重启，CPU 将执行以下操作:

- 将 CPU 和信号板输出置为 0
- 将中央机架信号模块的输出和分布式 I/O 设置为模块数字量输出的设备组态中“对 CPU STOP 的响应”(Reaction to CPU STOP) 选项

如果故障模式重新启动失败（例如，由于硬件故障），则 STOP 和 ERROR LED 亮起，MAINT LED 熄灭。

⚠ 警告
故障状态下无法保证正常运行 控制设备在不安全情况下运行时可能会出现故障，从而导致受控设备的意外运行。这种意外运行可能会导致人员死亡、重伤和/或设备损坏。 应使用紧急停止功能、机电超控功能或其它独立于 PLC 的冗余安全功能。

信号模块 (SM) 上的状态 LED

此外，各数字量 SM 还提供了指示模块状态的 DIAG LED：

- 绿色指示模块处于运行状态
- 红色指示模块有故障或处于非运行状态

各模拟量 SM 为各路模拟量输入和输出提供了 I/O Channel LED。

- 绿色指示通道已组态且处于激活状态
- 红色指示个别模拟量输入或输出处于错误状态

此外，各模拟量 SM 还提供有指示模块状态的 DIAG LED：

- 绿色指示模块处于运行状态
- 红色指示模块有故障或处于非运行状态

SM 可检测模块的通断电情况（必要时，还可检测现场侧电源）。

表格 15-2　信号模块 (SM) 上的状态 LED

说明	DIAG （红色/绿色）	I/O Channel （红色/绿色）
现场侧电源关闭 *	呈红色闪烁	呈红色闪烁
没有组态或更新在进行中	呈绿色闪烁	灭
模块已组态且没有错误	亮（绿色）	亮（绿色）

说明	DIAG （红色/绿色）	I/O Channel （红色/绿色）
错误状态	呈红色闪烁	-
I/O 错误（启用诊断时）	-	呈红色闪烁
I/O 错误（禁用诊断时）	-	亮（绿色）

* 状态仅在模拟信号模块上支持。

信号板 (SB) 上的状态 LED

各模拟量 SB 为各路模拟量输入和输出提供了 I/O Channel LED。

表格 15-3　信号板 (SB) 上的状态 LED

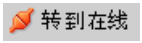
说明	I/O Channel （红色/绿色）
没有组态或更新在进行中	灭
信号板已组态且没有错误	亮（绿色）
I/O 错误（启用诊断时）	呈红色闪烁
I/O 错误（禁用诊断时）	亮（绿色）

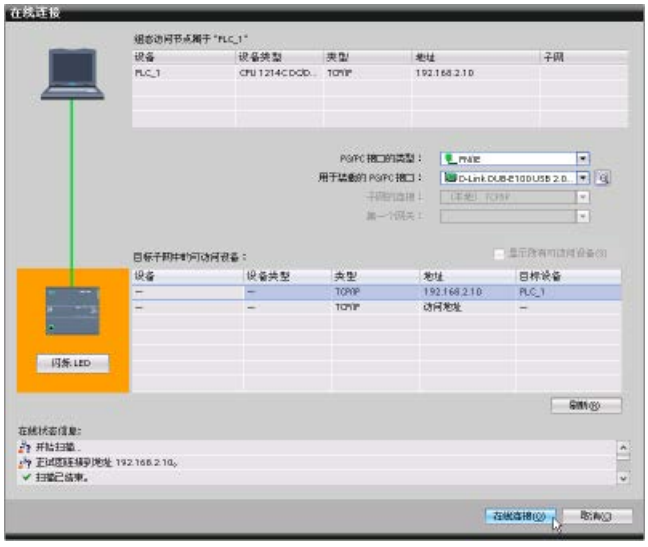
15.2 转到在线并连接到 CPU

装载程序和项目工程数据以及执行下列活动时，必须在编程设备和 CPU 之间建立在线连接：

- 测试用户程序
- 显示和改变 CPU 的工作模式 (页 1392)
- 显示和设置 CPU 的日期和日时钟 (页 1389)
- 显示模块信息
- 比较和同步 (页 1395)离线与在线程序块
- 上传和下载程序块
- 显示诊断和诊断缓冲区 (页 1393)
- 通过使用监视表格 (页 1400)监视并修改值来测试用户程序
- 使用强制表格强制 CPU 中的值 (页 1403)

要与组态的 CPU 建立在线连接，请单击“项目导航”树中的 CPU，并在“项目”(Project) 视图中单击“转到在线模式”(Go online) 按钮：





如果这是该 CPU 首次转到在线模式，则必须从“转到在线模式”(Go Online) 对话框中选择 PG/PC 接口的类型以及特定的 PG/PC 接口，然后才能在与该接口中发现的 CPU 建立在线连接。

现在，编程设备已连接到 CPU。 橙色单元指示存在在线连接。 现在，您就可以使用“项目树”和“在线工具任务卡”中的“在线和诊断”(Online & diagnostics) 工具。

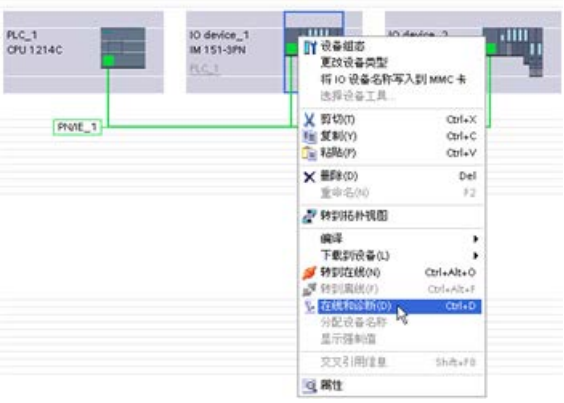
15.3 在线为 PROFINET IO 设备分配名称

PROFINET 网络中的设备在分配名称后才可与 CPU 连接。如果 PROFINET 设备尚未分配名称，或要更改该设备的名称，则可使用“设备和网络”(Devices & networks) 编辑器为该设备分配名称。

对于各 PROFINET IO 设备，必须在 STEP 7 项目（使用“在线和诊断”(Online & diagnostics) 工具）和 PROFINET IO 设备组态存储器（例如 ET200 S 接口模块组态存储器）

中为该设备分配相同的名称。如果名称缺失或两个位置中的名称不匹配，则 PROFINET IO 数据交换模式将不会运行。

1.在“设备和网络”(Devices & networks) 编辑器中，右键单击所需的 PROFINET IO 设备，并选择“在线和诊断”(Online & diagnostics)。



2.在“在线和诊断”(Online & diagnostics) 对话框中，选择以下菜单项：

- “功能”(Functions)
- “指定 PROFINET 设备名称”

单击“更新列表”(Update list) 按钮显示所有网络上的 PROFINET IO 设备。

3.在显示的列表中，单击所需的 PROFINET IO 设备，然后单击“分配名称”(Assign name) 按钮将该名称写入 PROFINET IO 设备组态存储器。

15.4 设置 IP 地址和日时钟

可以设置在线 CPU 中的 IP 地址 (页 817)和时间。 访问在线 CPU“项目树”中的“在线和诊断”(Online & diagnostics) 之后，可以显示或更改 IP 地址。 还可以显示或设置在线 CPU 的时间和日期参数。

说明

该特性仅对只具有 MAC 地址（还未分配 IP 地址）或已恢复出厂设置的 CPU 可用。

15.5 复位为出厂设置

可在以下情形下将 S7-1200 复位为原始出厂设置：

- CPU 有在线连接。
- CPU 处于 STOP 模式。

说明

如果 CPU 处于 RUN 模式，而用户要启动复位操作，则可在接受确认提示后将其切换到 STOP 模式。

步骤

要将 CPU 复位为出厂设置，请按以下步骤操作：

1. 打开 CPU 的“在线和诊断”(Online and Diagnostics) 视图。
2. 从“功能”(Functions) 文件夹中选择“复位为出厂设置”(Reset to factory settings)。
3. 如果要保留 IP 地址，请选中“保留 IP 地址”(Retain IP address) 复选框；如果要删除 IP 地址，则选中“删除 IP 地址”(Delete IP address) 复选框。

4. 单击“复位”(Reset) 按钮。
5. 单击“确定”(OK) 接受确认提示。

结果

模块会根据需要切换到 STOP 模式，并复位为出厂设置。CPU 执行了以下操作：

CPU 中安装了存储卡	CPU 中未安装存储卡
• 清空诊断缓冲区 • 复位时间 • 从存储卡恢复工作存储器 • 将所有操作数区域设置为组态的初始值 • 将所有参数设置为其组态的值 • 根据您所做的选择，保留或删除 IP 地址。（MAC 地址是固定的，始终不变。）[1] • 如果存在控制数据记录 (页 153)，则将其删除	• 清空诊断缓冲区 • 复位时间 • 清空工作存储器和内部装载内存 • 将所有操作数区域设置为组态的初始值 • 将所有参数设置为其组态的值 • 根据您所做的选择，保留或删除 IP 地址。（MAC 地址是固定的，始终不变。）[1] • 如果存在控制数据记录，则将其删除

[1] 如果选择了“保留 IP 地址”(Retain IP address)，CPU 将把 IP 地址、子网掩码和路由器地址（如果使用）设为硬件配置中的设置，除非已通过用户程序或其它工具修改了这些值，这种情况下 CPU 将恢复修改后的值。

15.6 更新固件

您可以使用以下两种方法，通过 STEP 7 在线和诊断工具更新连接的 CPU：

- 更新项目中 CPU
- 更新项目树中可访问的设备

更新项目中 CPU 的固件

要执行固件更新，请执行以下步骤：

1. 打开项目树中对应连接的 CPU 的 CPU。
2. 打开所连接 CPU 的“在线和诊断”(Online and Diagnostics) 视图。
3. 从“功能”(Function) 目录中选择“固件更新”(Firmware update)。
4. 在“固件加载程序”(Firmware loader) 区域中，单击“浏览”(Browse) 按钮导航到包含固件更新文件的位置。该位置可能是硬盘上从 Siemens 工业在线支持网站 (https://support.industry.siemens.com/cs/cn/zh)上下载的 S7-1200 (http://support.automation.siemens.com/WW/view/zh/34612486/133100) 固件更新文件所在的位置。

5. 选择与模块兼容的文件。表中会显示与所选文件兼容的模块。
6. 单击“运行更新”(Run update) 按钮。如有必要，根据对话框更改 CPU 的工作模式。

STEP 7 在加载固件更新时会显示进程对话框。完成后，对话框会提示您使用新固件启动模块。

说明

如果没有选择使用新固件启动模块，则在通过循环上电等操作复位模块前，先前的模块将保持激活状态。只有复位模块后，新固件才能激活。

更新可访问设备的固件

要对一个或多个可访问的设备执行固件更新，请按下列步骤操作：

1. 在项目树中打开“在线访问”(Online access)。
2. 打开 CPU 连接的通信接口。
3. 双击“更新可访问设备”(Update accessible devices) 然后等待 STEP 7 显示在线设备。
4. 展开要更新的 CPU 然后双击“在线和诊断”(Online & diagnostics)。
5. 展开“功能”(Functions) 目录中的“固件更新”(Firmware update)。将看到 PLC 以及该 PLC 的本地模块。选择“PLC”或“本地模块”(Local modules) 选项，您可以按照上述的“固件加载程序”(Firmware loader) 部分继续进行固件更新。

还可以通过以下任一方法来执行固件更新：

- 使用 SIMATIC 存储卡 (页 143)
- 使用 Web 服务器“模块信息”标准 Web 页面 (页 1045)
- 使用 SIMATIC 自动化工具 (https://support.industry.siemens.com/cs/cn/zh/view/98161300/en)

15.7 通过 STEP 7 格式化 SIMATIC 存储卡

您可以通过 STEP 7 在线和诊断工具格式化连接的 CPU 中的存储卡。为此，请按下列步骤操作：

1. 确保 CPU 处于 STOP 模式。注意如果 CPU 处于 RUN 模式时启动了格式化操作，STEP 7 将提示您允许 STEP 7 将 CPU 置于 STOP 模式下。
2. 将存储卡插入到连接的 CPU
3. 通过项目中的 CPU 或项目树中在线访问中的可访问设备打开连接的 CPU 的“在线和诊断”(Online & diagnostics)。
4. 如果 CPU 在线，选择所连 CPU 的“转到在线”(Go online) 选项。
5. 选择“功能”(Functions) 菜单中的“格式化存储卡”(Format memory card)。

6. 单击“格式化”(Format)。
7. 单击“是”(Yes) 确认提示。

然后 STEP 7 格式化存储卡，完成后在信息窗口中显示一条消息。完成后 CPU 处于 STOP 模式，STOP 和 MAINT 灯闪烁。此时还不能切换到 RUN 模式，必须执行以下任一步骤：

- 移除存储卡并重启 CPU。如果 CPU 的内部装载存储器含有程序，CPU 将带程序启动。
- 不移除存储卡重启 CPU：如果 CPU 的内部装载存储器含有程序，CPU 将该程序复制到存储卡然后带程序重启。如果内部装载存储器不含程序，CPU 将更改存储卡为程序卡 (页 139)并等待下载。

说明

格式化存储卡对内部装载存储器没有影响。

当插入存储卡（在插入存储卡和执行格式化操作中间未重启 CPU）时 CPU 正在使用内部装载存储器，CPU 将保持内部装载存储器的内容。

15.8 在线 CPU 的 CPU 操作员面板

“CPU 操作员面板”(CPU operator panel) 显示在线 CPU 的工作模式（STOP 或 RUN）。 该面板还显示 CPU 是否有错误或值是否处于强制状态。

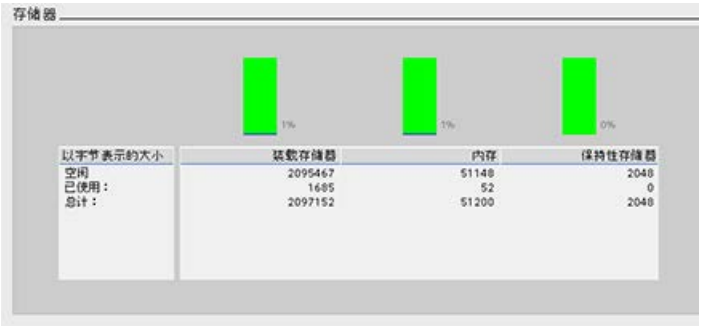
使用“在线工具”(Online Tools) 任务卡的 CPU 操作面板可更改在线 CPU 的工作模式。 只要 CPU 处于在线模式，便可访问“在线工具”(Online Tools) 任务卡。

15.9 监视循环时间和存储器使用情况

可以监视在线 CPU 的循环时间和存储器使用情况。

连接到在线 CPU 后，打开“在线工具”(Online tools) 任务卡查看以下测量值：

- 循环时间
- 存储器使用情况



15.10 显示 CPU 中的诊断事件

使用诊断缓冲区可以查看 CPU 的近期活动。 可通过“在线和诊断”(Online & Diagnostics) 访问项目树中在线 CPU 的诊断缓冲区。 它包含以下条目：

- 诊断事件
- CPU 工作模式改变（切换到 STOP 或 RUN 模式）

第一个条目包含最新的事件。 诊断缓冲区中的各条目均包含记录事件的日期和时间以及一段说明。

最大条目数由 CPU 决定。 最多支持 50 个条目。

仅永久存储诊断缓冲区中 10 个最新的事件。 将 CPU 复位为工厂设置会通过删除条目的方式复位诊断缓冲区。

还可以使用 GET_DIAG 指令 (页 480)来采集诊断信息。

15.11 比较离线 CPU 与在线 CPU

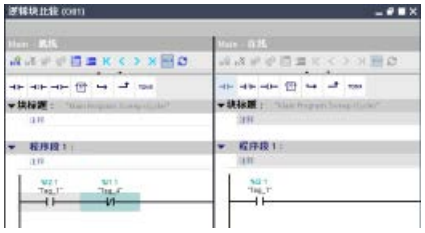
可以将在线 CPU 中的代码块与项目中的代码块进行比较。如果项目中的代码块与在线 CPU 的代码块不匹配，则可通过“比较”编辑器使项目与在线 CPU 同步，具体方法可以是将项目的代码块下载到 CPU 中，或者从项目中删除在线 CPU 中不存在的块。

在项目中选择 CPU。

使用“比较离线/在线”(Compare Offline/online) 命令打开“比较”(Compare) 编辑器。（通过“工具”(Tools) 菜单或右键单击项目中 CPU 的方式访问该命令。）

单击某对象的“操作”(Action) 列，在删除对象、不执行任何操作或将该对象下载到设备这三项中进行选择。

单击“同步”(Synchronize) 按钮装载代码块。



在“比较目标”(Compare to) 列中右键单击一个对象，并选择“开始详细比较”(Start detailed comparison) 按钮可并排显示代码块。

详细比较功能会高亮显示在线 CPU 代码块与项目中 CPU 代码块之间的差异。

说明

针对离线/在线比较操作受保护 CPU 所需的读访问

对于 STEP 7 V14 或更高版本，“HMI 访问”(HMI access) 安全等级不足以执行离线/在线比较操作。要进行离线/在线比较操作，必须具有“读访问”(Read access) 或“完全访问”(Full access) 权限。

另请参见 CPU 的访问保护 (页 198)

15.12 比较在线/离线拓扑

在 STEP 7 拓扑概览中，可将组态的离线拓扑与实际在线拓扑进行比较。

步骤

要找出已组态拓扑和实际拓扑之间的差异，请执行以下步骤：

1. 显示拓扑视图的概览表。
2. 在拓扑概览的工具栏中，单击“离线/在线比较”(offline/onlinecomparison) 按钮：

结果

在拓扑概览表中，STEP 7 移除了“伙伴站”(Partner station)、“伙伴接口”(Partner interface) 和“电缆数据”(Cable data) 列并插入了“状态”(Status) 和“操作”(Action) 比较列。对于拓扑概览中的每个设备或端口，“状态”(Status) 列按如下所示显示比较状态：

图标	含义
	在至少一个低级组件中存在不同的拓扑
	拓扑相同
	拓扑信息仅在离线状态下可用或设备已禁用
	拓扑信息仅在在线状态下可用
	拓扑不同
	设备不支持拓扑功能

对于每一个进行比较的接口或设备，“操作”(Action) 列提供以下选择：

图标	含义
=	无可执行的操作
←	采用在线连接

要重复进行比较，可单击拓扑概览中的 工具栏按钮。

有关拓扑视图、拓扑概览和在线/离线拓扑比较的更多信息，请参见 STEP 7 信息系统。还可以在 通过 STEP 7 V13 组态 PROFINET 功能手册 (https://support.industry.siemens.com/cs/cn/zh/view/49948856)中找到更多信息。

15.13 监视和修改 CPU 中的值

STEP 7 提供了用于监视 CPU 的在线工具：

- 您可以显示或监视变量的当前值。 监视功能不会改变程序顺序。 它为用户提供有关 CPU 中程序的顺序以及数据信息。
- 还可以使用其它功能控制用户程序的顺序和数据：
  - 可以修改在线 CPU 中变量的值，了解用户程序如何响应。
  - 可以将外围设备输出（如 Q0.1:P 或“Start”:P）强制为特定值。
  - 可以在 STOP 模式下启用输出。

说明

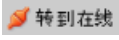
使用控制功能时必须始终小心谨慎。 这些功能可能会严重影响用户/系统程序的执行。

表格 15-4 STEP 7 编辑器的在线功能

编辑器	监视	修改	强制
监视表格	有	有	无
强制表格	有	无	有
程序编辑器	有	有	无
变量表	有	无	无
DB 编辑器	有	无	无

15.13.1 转到在线模式监视 CPU 中的值

要监视变量，必须在线连接到 CPU。 只需单击工具栏中的 “转到在线”(Go online) 按钮。



连接到 CPU 后，STEP 7 将工作区的标题变为橙色。

项目树显示离线项目和在线 CPU 的比较结果。 绿色圆点表示 CPU 与项目同步，即二者都具有相同的组态和用户程序。

变量表会显示变量。 监控表也可以显示变量以及直接地址。

要监视用户程序的执行并显示变量的值，请单击工具栏中的 “全部监视”(Monitor all) 按钮。

“监视值”(Monitor value) 字段中将显示每个变量的值。

15.13.2 显示程序编辑器中的状态

可在 LAD 和 FBD 程序编辑器中监控多达 50 个变量的状态。 使用编辑器栏显示 LAD 编辑器。 使用编辑器栏，可以在打开的编辑器之间切换视图，而无需打开或关闭编辑器。

在程序编辑器的工具栏中，单击 “接通/断开监视”(Monitoring on/off) 按钮，以显示用户程序的状态。

程序编辑器中的网络以绿色显示能流。

还可以右键单击指令或参数，以修改指令值。

15.13.3 捕获 DB 在线值快照用于恢复值操作

您可以从在线 CPU 捕获数据块变量的实际值快照，方便稍后使用。

注意以下前提条件：

- 必须能够在线连接到 CPU。
- 必须已在 STEP 7 中打开 DB。

捕获快照

要捕获快照，请按以下步骤操作：

1. 在 DB 编辑器中单击“监视所有变量”(Monitor all tags) 按钮：“监视值”(Monitor value) 列会显示实际数据值。
2. 单击 按钮捕获实际值快照并将其显示在“快照”(Snapshot) 列中。

您可以稍后使用该快照更新 CPU 实际值或替换起始值。

将快照值复制到 CPU 中

要将快照值复制到 CPU 中数据块变量的实际值中，单击以下按钮：

在线 CPU 将快照值加载到实际值中。“监视值”(Monitor value) 列显示了 CPU 中的实际值。扫描周期可能会在随后修改 CPU 中来自快照值的值，但在进行复制操作时，CPU 会对连续下载中的快照值进行加载操作。

说明

请注意如果快照包含状态信息、计时器值或计算信息，CPU 会在进行快照时恢复这些值。

将快照值复制到起始值

要将快照值复制到数据块变量的起始值中，单击以下按钮：

编译并将 DB 下载到 CPU 中后，DB 会在 CPU 进入 RUN 模式之后使用新起始值。

将单个快照或监视值复制到起始值

数据块编辑器还允许复制单个值并将其粘贴到起始值。只需右键单击“值”(value) 列中的任意值，然后选择“复制”(Copy) 将其放置在 Windows 剪贴板中。然后，可以单击任意起始值，然后选择“粘贴”(paste) 用剪贴板中的值替换该值。

编译并将 DB 下载到 CPU 中后，DB 会在 CPU 进入 RUN 模式之后使用新起始值。

15.13.4 使用监视表格来监视和修改 CPU 中的值

通过监视表格可以在 CPU 执行用户程序时对数据点执行监视和控制功能。 根据监视或控制功能的不同，这些数据点可以是过程映像（I 或 Q）、M、DB 或物理输入 (I_:P)。 由于监视功能只能显示从 Q 存储器写入的最后一个值，并且不会从物理输出读取实际值，因此无法准确监视物理输出 (Q_:P)。

监视功能不会改变程序顺序。 它为用户提供有关 CPU 中程序的顺序以及数据信息。

控制功能允许用户控制程序的顺序和数据。 使用控制功能时必须小心谨慎。 这些功能可能会严重影响用户/系统程序的执行。 三种控制功能是修改、强制和在 STOP 模式下启用输出。

使用监视表格可以执行以下在线功能：

- 监视变量的状态
- 修改个别变量的值

选择监视或修改变量的时间：

- 扫描循环开始时： 在该扫描循环开始时读取或写入值
- 扫描循环结束时： 在该扫描循环结束时读取或写入值
- 切换到停止

要创建监视表格：

1. 双击“添加新监视表格”(Add new watch table) 打开新监视表格。
2. 输入变量名称将变量添加到监视表格。

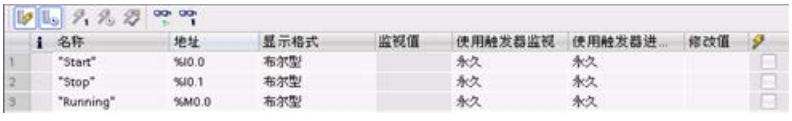
可使用以下选项监视变量：

- “监视全部”(Monitor all)： 该命令用于启动对激活的监视表格中的可见变量进行监视。
- “立即监视”(Monitor now)： 该命令用于启动对激活的监视表格中的可见变量进行监视。 监视表格仅立即监视变量一次。

可使用以下选项修改变量：

- “修改为 0”(Modify to 0) 将所选地址的值设置为“0”。
- “修改为 1”(Modify to 1) 将所选地址的值设置为“1”。
- “立即修改”(Modify now) 立即修改所选地址的值一个扫描周期。
- “使用触发器修改”(Modify with trigger) 修改所选地址的值。
  该功能不提供反馈来指示实际上是否修改了所选地址。 如果需要修改反馈，则使用“立即修改”(Modify now) 功能。
- “启用外围设备输出”(Enable peripheral outputs) 禁用输出禁用命令并且仅在 CPU 处于 STOP 模式时可用。

要监视变量，必须在线连接到 CPU。



	名称	地址	显示格式	监视值	使用触发器监视	使用触发器进...	修改值
1	"Start"	%I0.0	布尔型		永久	永久	
2	"Stop"	%I0.1	布尔型		永久	永久	
3	"Running"	%M0.0	布尔型		永久	永久	

使用监视表顶部的按钮选择不同的功能。

输入要监视的变量名称并从该下拉选择项中选择一种显示格式。 在线连接到 CPU 时，单击“监视”(Monitor) 按钮将在“监视值”(Monitor value) 字段中显示数据点的实际值。

15.13.4.1 监视或修改 PLC 变量时使用触发器

触发决定将在扫描周期中的哪个点监视或修改所选地址。

表格 15-5 触发器类型

触发器	说明
永久	连续采集数据
扫描周期开始时	永久： CPU 读取输入后，在扫描周期开始时连续采集数据
	一次： CPU 读取输入后，在扫描周期开始时采集一次数据
扫描周期结束时	永久： CPU 写入输出前，在扫描周期结束时连续采集数据
	一次： CPU 写入输出前，在扫描周期结束时采集一次数据
切换到 STOP 时	永久： CPU 切换到 STOP 时连续采集数据
	一次： CPU 切换到 STOP 后采集一次数据

要在给定触发点修改 PLC 变量，请选择周期开始或结束。

- 修改输出： 触发修改输出事件的最佳时机是在扫描周期结束且 CPU 马上要写入输出之前的时间。
  在扫描周期开始时监视输出的值以确定写入到物理输出中的值。 此外，在 CPU 将值写入到物理输出前监视输出以检查程序逻辑并与实际 I/O 行为进行比较。
- 修改输入： 触发修改输入事件的最佳时机是在周期开始、CPU 刚读取输入且用户程序要使用输入值之前的时间。
  如果怀疑值在扫描期间发生变化，您可能想在扫描周期结束时监视输入值，以确保扫描周期结束时的输入值与扫描周期开始时相同。 如果值不同，则用户程序可能会错误地写入到输入。

要诊断 CPU 转到 STOP 的可能原因，请使用 “切换到 STOP”(Transition to STOP) 触发器捕捉上一个过程值。

15.13.4.2 在 STOP 模式下启用输出

监视表格允许用户在 CPU 处于 STOP 模式时写入输出。 通过该功能可以检查输出的接线并检验连接到输出引脚的电线是将高电平信号还是低电平信号引入与其相连的过程设备端子。

> ⚠ 警告
>
> **在 STOP 模式下写入物理输出的风险**
>
> 即使在 CPU 处于 STOP 模式时，启用物理输出也可激活相连的过程点，进而可能导致意外的设备操作。 意外的设备操作可导致死亡或严重的人身伤害。
>
> 从监视表中写入到输出之前，请确保更改物理输出不会导致意外的设备操作。 请始终遵守过程设备的安全预防措施。

输出启用时，可以在 STOP 模式下修改输出的状态。 如果输出禁用，则无法在 STOP 模式下修改输出。 要在 STOP 模式下从监视表启用输出的修改，请按以下步骤操作：

1. 从“在线”(Online) 菜单中选择“扩展模式”(Expanded mode) 菜单命令。
2. 选择“在线”(Online) 菜单中“修改”(Modify) 命令的“启用外围设备输出”(Enable peripheral outputs) 选项，或者右键单击监视表行后从上下文菜单中选择。

如果已组态分布式 I/O，则无法在 STOP 模式下启用输出。如果尝试此操作，将返回错误。

将 CPU 设置为 RUN 模式会禁用“启用外围设备输出”(Enable peripheral outputs) 选项。

如果任何输入或输出被强制，则处于 STOP 模式时不允许 CPU 启用输出。 必须先取消强制功能。

15.13.5 CPU 中的强制值

15.13.5.1 使用强制表格

强制表格提供了“强制”功能，能够将与外围设备输入或外围设备输出地址对应的输入或输出点的值改写成特定的值。 CPU 在执行用户程序前将此强制值应用到输入过程映像并在将输出写入到模块前将其应用到输出过程映像。

> **说明**
>
> 强制值存储在 CPU 中，而不是强制表格中。
>
> 不能强制输入（或“I”地址）或输出（或“Q”地址）。 但是，可以强制外围设备输入或外围设备输出。 强制表格将自动在地址后面添加一个“:P”（例如： "On":P 或 "Run":P）。

	名称	地址	显示格式	监视值	强制值	F
1	"On":P	%I0.0:P	布尔型		TRUE	☑
2	"Off":P	%I0.1:P	布尔型			☐
3	"Run":P	%Q0.1:P	布尔型			☐

在“强制值”(Force value) 单元格中，输入要强制的输入值或输出值。 然后可以使用“强制” (Force) 列中的复选框启用对输入或输出的强制功能。

使用“启动或替换强制”(Start or replace forcing) 按钮强制设置强制表格中的变量值。 单击“停止强制”(Stop forcing) 按钮重置变量值。

在强制表格中，可以监视输入的强制值的状态。 但是不能监视输出的强制值。

还可以在程序编辑器中查看强制值的状态。

说明

在强制表格中强制输入或输出时，强制操作将变成项目组态的一部分。 如果关闭 STEP 7，被强制元素仍会在 CPU 程序中保持激活状态，直至这些元素被清除。 要清除这些被强制元素，必须使用 STEP 7 连接到在线 CPU，然后使用强制表格断开或停止对这些元素的强制功能。

15.13.5.2 强制功能的操作

CPU 允许用户在强制表格中指定物理输入或输出地址（I_:P 或 Q_:P）然后启动强制功能，以此来强制输入和输出点。

在程序中，物理输入的读取值被强制值覆盖。 程序在处理过程中使用该强制值。 程序写入物理输出时，输出值被强制值覆盖。 强制值出现在物理输出端并被过程使用。

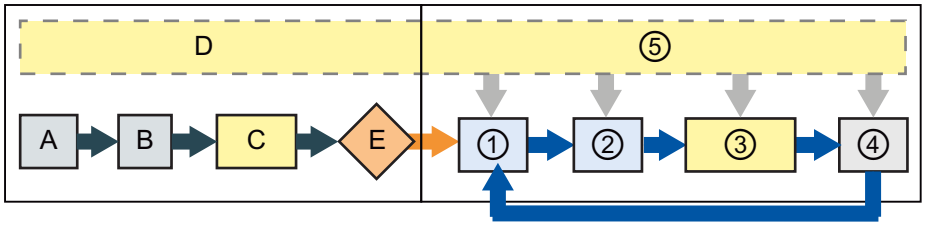
在强制表格中强制输入或输出时，强制操作将变成用户程序的一部分。 即使编程软件已关闭，强制选项在运行的 CPU 程序中仍保持激活，直到在线连接到编程软件并停止强制功能将其清除为止。 含有通过存储卡装载到另一个 CPU 的强制点的程序将继续强制程序中选择的点。

如果 CPU 正在执行写保护存储卡上的用户程序，则无法通过监控表初始化或更改对 I/O 的强制，因为用户无法改写写保护用户程序中的值。 强制写保护值的任何尝试都将生成错误。 如果使用存储卡传送用户程序，则该存储卡上的所有被强制元素都将被传送到 CPU。

说明

无法强制分配给 HSC、PWM 和 PTO 的数字 I/O 点

在设备配置期间分配高速计数器 (HSC)、脉冲宽度调制 (PWM) 和脉冲串输出 (PTO) 设备使用的数字 I/O 点。 将数字量 I/O 点的地址分配给这些设备之后，无法通过强制表的强制功能修改所分配的 I/O 点的地址值。



启动

A 强制功能不影响 I 存储区的清除。

B 强制功能不影响输出值的初始化。

C 启动 OB 执行期间，CPU 在用户程序访问物理输入时应用强制值。

D 不影响将中断事件存储到队列。

E 不影响写入到输出的启用。

RUN

① 将 Q 存储器写入到物理输出时，CPU 在更新输出时应用强制值。

② 读取物理输入时，CPU 仅在将这些输入复制到 I 存储器前应用强制值。

③ 用户程序（程序循环 OB）执行期间，CPU 在用户程序访问物理输入或写入物理输出时应用强制值。

④ 强制功能不影响通信请求和自检诊断的处理。

⑤ 不影响在扫描周期的任何时段内处理中断。

15.14 在 RUN 模式下下载

该 CPU 支持“在 RUN 模式下下载”(Download in RUN mode)。 此功能是为了让您以对过程干扰最小的方式对控制该过程的程序进行小幅改动。 但是，执行此功能也可以对程序进行重大更改，这可能会导致损坏甚至危险情况。

> ⚠ 警告
>
> **在 RUN 模式下下载的风险**
>
> 在 RUN 模式下向 CPU 中下载更改时，这些更改将立即影响过程操作。 在 RUN 模式下更改程序可能会引起意外的系统操作，进而导致人员死亡、重伤和/或设备损坏。
>
> 在 RUN 模式下执行下载的人员必须经过授权，并清楚 RUN 模式下的更改对系统运行的影响。

利用“在 RUN 模式下下载”功能，可在不切换为 STOP 模式的情况下对程序进行更改，并将其下载到 CPU 中:

- 可以在不停机的情况下对当前过程进行少量更改（例如，更改一个参数值）。
- 可利用此功能更快速地调试程序（例如，插入一段常开或常闭开关逻辑）

可在 RUN 模式下进行下列程序块和变量更改，并将其下载到 CPU 中：

- 创建、覆盖和删除函数 (FC)、函数块 (FB) 和变量表。
- 创建、删除以及覆盖数据块 (DB) 和函数块 (FB) 的背景数据块。 可添加到数据块结构并在 RUN 模式下下载它们。 根据组态设置 (页 1411)，CPU 可维持现有块变量的值并将新的数据块变量初始化为各自的初始值，或者 CPU 将所有数据块变量设置为初始值。 无法在 RUN 模式下下载 Web 服务器 DB（控件或片段）。
- 覆盖组织块 (OB)；但是，不能创建或删除 OB。

在 RUN 模式下，您一次最多可下载二十个块。 如果要下载的块多于二十个，必须将 CPU 置于 STOP 模式。

如果将更改下载到实际过程（相对仿真过程而言，程序调试期间可能会进行仿真），在下载前必须全面考虑可能会对机器操作员和机器造成的安全后果，这一点非常重要。

说明

如果 CPU 处于 RUN 模式，且进行了程序更改，则 STEP 7 始终会尝试先在 RUN 模式下下载。 如果不希望出现这种情况，则必须将 CPU 置于 STOP 模式。

如果“在 RUN 模式下下载”不支持所做的更改，那么 STEP 7 将提示用户 CPU 必须转到 STOP 模式。

15.14.1 “在 RUN 模式下下载”的先决条件

要向 RUN 模式下的 CPU 中下载程序更改，必须满足以下先决条件：

- CPU 版本为 V3.0 或更高版本

  说明

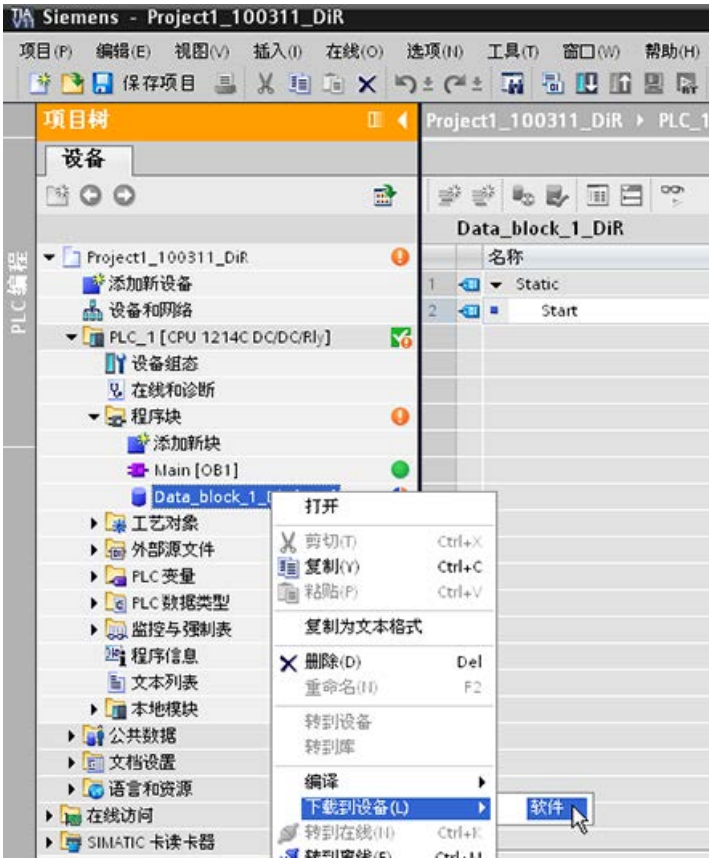
  要在 RUN 模式下修改扩展块接口 (页 1411)，CPU 必须为 V4.0 或以上版本。

- 程序必须编译成功。
- 必须已在运行 STEP 7 的编程设备和 CPU 之间成功建立通信。

15.14.2 在 RUN 模式下更改程序

要在 RUN 模式下更改程序，必须先确保 CPU 和程序满足先决条件 (页 1407)，然后再执行以下步骤：

1. 如果要在 RUN 模式下下载程序，请选择以下某种方法：
   – 从“在线”(Online) 菜单中选择“下载到设备”(Download to device) 命令。
   – 单击工具栏中的“下载到设备”(Download to device) 按钮。
   – 在“项目树”中，右键单击“程序块”并选择“下载到设备 > 软件”(Download to device > Software) 命令。



   如果程序已编译成功，STEP 7 会开始将该程序下载到 CPU 中。

2. STEP 7 将提示您加载程序或取消操作时，单击“加载”(Load) 将程序下载到 CPU。

15.14.3 下载所选块

在“程序块”(Program blocks) 文件夹中，可以选择单个块或选择要下载的块。

如果选择下载单个块，则“操作”(Action) 列中将只显示“统一下载”(Consistent download) 一个选项。
可以展开类别行，以确保选择要加载的块。 在本例中，仅对离线块进行少量更改，无需加载其它块。

在本例中，需要下载多个块。

说明

在 RUN 模式下，您一次最多可下载二十个块。 如果要下载的块多于二十个，必须将 CPU 置于 STOP 模式。

如果尝试在 RUN 模式下下载，但系统在实际下载前检测出无法执行该操作，则该对话框中将显示“停止模块”(Stop modules) 类别行。

单击“加载”(Load) 按钮，将显示“加载结果”(Load results) 对话框。 单击“完成”(Finish) 按钮完成下载。

15.14.4 其它块中存在编译错误时下载选定的单个块

如果当其它块中存在编译错误时尝试执行统一下载，则该对话框中将显示错误信息，并禁用加载按钮。

您必须更正其它块中的编译错误。 之后，才会激活“加载”(Load) 按钮。

15.14.5 在 RUN 模式下修改和下载现有块

利用“在 RUN 模式下下载”功能，您可以在数据块和函数块中添加和修改变量，然后在 RUN 模式下将更改的块下载到 CPU。

下载而不重新初始化

每个数据块和函数块都有一定大小的预留存储器，可用来向随后在 RUN 模式下下载的块中添加变量。默认情况下，存储器预留区域的初始大小为 100 字节。您可以向数据中添加其它变量，直至达到存储器预留区域的大小，并在 RUN 模式下将扩展块下载到 CPU。如果需要在块中为附加变量提供更多存储空间，也可以增大存储器预留区域。如果添加的变量超过了已分配的存储空间，则无法在 RUN 模式下将扩展块下载到 CPU 中。

利用“下载而不重新初始化”功能，您可以通过添加更多的数据块变量来扩展数据块并在 RUN 模式下下载扩展数据块。这样，您便可向数据块中添加变量并下载该数据块而不重新初始化程序。CPU 将保留现有数据块变量的值并将新添加的变量初始化为其起始值。

要为 CPU 处于 RUN 模式的在线项目启用该功能，请按照以下步骤操作：

1. 在 STEP 7 项目树的“程序块”(Program blocks) 文件夹中，打开块。
2. 单击块编辑器中的“下载而不重新初始化”(Download without reinitialization) 切换按钮启用该功能。（启用后图标周围将会被方框包围：）
3. 单击提示中的“确定”(OK) 以确认选择。
4. 向块接口添加变量并在 RUN 模式下下载该块。存储器预留区域允许多少新变量，您就可以添加并下载多少新变量。

如果向块中添加的字节数超过为存储器预留区域组态的字节数，则尝试在 RUN 模式下下载块时，STEP 7 会显示错误。您必须编辑块属性，增大存储空间。在启用“下载而不重新初始化”功能时，不能删除现有条目或修改块的“存储器预留区域”。要禁用“下载而不重新初始化”功能，请按照以下步骤操作：

1. 单击块编辑器中的“下载而不重新初始化”(Download without reinitialization) 切换按钮禁用该功能。（禁用后图标周围无方框包围：）
2. 单击提示中的“确定”(OK) 以确认选择。
3. 下载该块。在下载对话框中，必须选择“重新初始化”(reinitialize) 以下载该扩展块。

下载过程随即将所有的现有块变量和新块变量重新初始化为其起始值。

下载保持性块变量

在 RUN 模式下下载保持性块变量需要分配保持性存储器预留区域。要组态该保持性存储器预留区域，请按照以下步骤操作：

1. 在 STEP 7 项目树的“程序块”(Program blocks) 文件夹中，右键单击该块并在上下文菜单中选择“属性”(Properties)。
2. 选择“下载而不重新初始化”(Download without reinitialization) 属性。
3. 选中“启用下载而不重新初始化保持性变量”(Enable download without reinitialization for retentive tags) 复选框。
4. 组态为保持性存储器预留区域提供的字节数。
5. 单击“确定”(OK) 保存更改。
6. 向数据块中添加保持性数据块变量并在 RUN 模式下下载该数据块。保持性存储器预留区域允许多少新保持性数据块变量，您就可以添加并下载多少新保持性数据块变量。

如果向块中添加的保持性字节数超过为保持性存储器预留区域组态的字节数，则尝试在 RUN 模式下下载块时，STEP 7 会显示错误。您向保持性存储器预留区域中添加的保持性块变量不能超过区域大小，这样才能在 RUN 模式下下载这些变量。

下载扩展的保持性块变量时，变量将包含其当前值。

为新块组态保留存储空间大小

新数据块的默认存储器预留区域的大小为 100 字节。创建新块时，预留区域提供 100 个字节。如果要更改新块的存储器预留区域大小，则可在 PLC 编程设置中更改设置：

1. 在 STEP 7 中选择 **“选项 > 设置”(Options > Settings)** 菜单命令。
2. 在“设置”(Settings) 对话框中，展开“PLC 编程”(PLC programming) 并选择“常规”(General)。
3. 在“下载而不重新初始化”(Download without reinitialization) 部分，输入存储器预留区域的字节数。

创建新块时，STEP 7 使用为新块输入的存储器预留区域组态。

限制

在 RUN 模式下编辑和下载块时，以下限制适用：

- 通过添加新变量扩展块接口并在 RUN 模式下下载仅适用于优化块 (页 182)。
- 如果不重新初始化，则无法在 RUN 模式下更改块结构并下载已更改的块。将新成员添加到 Struct (页 129) 变量、更改变量名称、数组大小、数据类型或保持性状态都需要重新初始化该块才能在 RUN 模式下下载该块。对于现有块变量，可以执行并且在 RUN 模式下下载而不重新初始化的唯一修改是对起始值（数据块）、默认值（函数块）或注释的更改。
- 在 RUN 模式下下载的新块变量数不能超过存储器预留区域可容纳的数目。
- 在 RUN 模式下下载的新的保持性块变量数不能超过保持性存储器预留区域可容纳的数目。

15.14.6 下载失败时的系统响应

执行“在 RUN 模式下下载”的过程中，如果出现网络连接故障，则 STEP 7 将显示以下“加载预览”(Load preview) 对话框：

15.14.7 在 RUN 模式下下载的考虑事项

在 RUN 模式下下载程序之前，如果发生以下情况，则需考虑 RUN 模式下进行修改对 CPU 运行的影响：

- 如果删除一个输出的控制逻辑，则在下一次上电循环或切换到 STOP 模式之前，CPU 将始终保持该输出的最终状态。
- 如果删除了正在运行的高速计数器或脉冲输出函数，则该高速计数器或脉冲输出将继续运行，直至下一次上电循环或切换到 STOP 模式。
- 在下一次上电循环或者从 STOP 切换到 RUN 模式之前，任何以首次扫描位状态为条件的逻辑都不会执行。首次扫描位只会因切换到 RUN 模式而置位，不受 RUN 模式下下载的影响。
- 不能覆盖数据块 (DB) 的当前值和/或变量。

说明

CPU 必须支持在 RUN 模式下进行更改，程序的编译必须没有错误，CPU 必须能与 STEP 7 通信，并且 CPU 必须无错误，这样才能在 RUN 模式下下载程序。

可在 RUN 模式下对程序块和变量进行以下更改，并将其下载到 CPU 中：

- 创建、覆盖和删除函数 (FC)、函数块 (FB) 和变量表。
- 创建和删除数据块 (DB)；但是，不会覆盖 DB 的结构更改。只能覆盖 DB 初始值。无法在 RUN 模式下下载 Web 服务器 DB（控件或片段）。
- 覆盖组织块 (OB)；但是，不能创建或删除 OB。

在 RUN 模式下，您一次最多可下载二十个块。如果要下载的块多于二十个，必须将 CPU 置于 STOP 模式。

下载一旦启动，在其完成前将无法在 STEP 7 中执行其它任务。

由于“在 RUN 模式下下载”，可能导致出错的指令

CPU 中激活了“在 RUN 模式下下载”后，以下指令可能会发生临时错误。如果 CPU 正准备激活已下载的更改，那么初始化指令时将出现错误。在此过程中，CPU 将暂停用户程序访问加载存储器的初始化过程，同时完成正在进行的用户程序对加载存储器的访问。完成后，将统一激活所下载的更改。

指令	暂停激活时的响应
DataLogCreate	STATUS = W#16#80C0, ERROR = TRUE
DataLogOpen	STATUS = W#16#80C0, ERROR = TRUE
DataLogWrite	STATUS = W#16#80C0, ERROR = TRUE
DataLogClose	STATUS = W#16#80C0, ERROR = TRUE
DataLogNewFile	STATUS = W#16#80C0, ERROR = TRUE

指令	暂停激活时的响应
DataLogClear	STATUS = W#16#80C0, ERROR = TRUE
DataLogDelete	STATUS = W#16#80C0, ERROR = TRUE
READ_DBL	RET_VAL = W#16#82C0
WRIT_DBL	RET_VAL = W#16#82C0
Create_DB	RET_VAL = W#16#80C0
Delete_DB	RET_VAL = W#16#80C0
RTM	RET_VAL = 0x80C0

无论何种情况，只要发生错误，指令的 RLO 输出都将失败。该错误是临时错误。如果出现错误，则需稍后重试该指令。

说明

而不能在执行 OB 的过程中重试该操作。

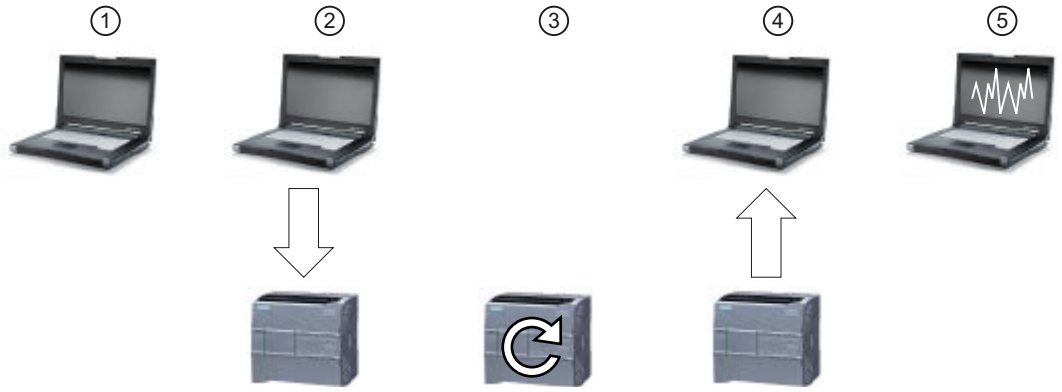
15.15 根据触发条件跟踪并记录 CPU 数据

STEP 7 提供了轨迹和逻辑分析器功能，可用于组态 PLC 要跟踪和记录的变量。随后可将记录的跟踪测量数据上传到编程设备并使用 STEP 7 工具分析、管理并以图形方式显示这些数据。使用 STEP 7 项目树中的 “轨迹”(Traces) 文件夹创建和管理轨迹。

说明

跟踪测量数据仅在 STEP 7 项目中可用，通过其它工具进行处理时则不可用。

下图显示了轨迹功能的各个步骤：



① 在 STEP 7 的跟踪编辑器中组态跟踪。用户可以组态下列选项：
- 待记录的数据值
- 记录时长
- 记录频率
- 触发条件

② 将轨迹组态从 STEP 7 传送到 PLC。

③ PLC 执行该程序，并在发生触发条件时开始记录轨迹数据。

④ 将记录的值从 PLC 传送到 STEP 7。

⑤ 使用 STEP 7 中的工具分析、以图形方式显示并保存该数据。

S7-1200 支持两个跟踪作业，每个触发事件最多捕捉 16 个变量。每个跟踪作业提供 524288 字节的 RAM 用于记录跟踪值及其相关信息，例如变量地址和时间戳。

将跟踪测量结果保存到存储卡

S7-1200 CPU 只能将跟踪测量数据保存到 SIMATIC 存储卡中。如果 CPU 中无存储卡，则程序试图保存跟踪测量数据时，CPU 将记录诊断缓冲区条目。CPU 对分配给跟踪测量数据的空间有限制，因此必须始终保证有 1 MB 的外部装载存储器可用。如果跟踪测量数据需要的存储空间大于最大限制，则 CPU 将不会保存测量数据，而会记录诊断缓冲区条目。

此外，如果在 STEP 7 中选择“覆盖最早的记录”(Overwrite oldest recording)，那么继续执行写操作会缩短装载存储器的寿命。选择“覆盖最早的记录”(Overwrite oldest recording) 后，CPU 在存储组态的跟踪测量结果数量后会用最新的测量结果替代最早的测量结果，然后继续跟踪和保存测量结果。覆盖最早的测量结果在捕捉间断问题方面非常有效。

CPU 最多支持 999 个跟踪测量结果。CPU 在将跟踪测量结果保存到外部装载存储器的过程中不会检查跟踪作业的触发条件，而在完成保存跟踪测量结果后开始检查触发条件。

访问示例

关于如何编程轨迹、如何下载组态、上传轨迹数据以及在逻辑分析器中显示数据的详细信息，请参见 STEP 7 信息系统。详细示例，请参见“使用在线和诊断功能 > 使用轨迹和逻辑分析器功能”一章。

此外，《工业自动化 SINAMICS/SIMATIC 使用轨迹和逻辑分析器功能》 (https://support.industry.siemens.com/cs/cn/zh/view/64897128)在线手册也是一个很好的参考。

15.16 确定 SM 1231 模块的断路条件类型

如模拟量输入的电压和电流测量范围（SB 和 SM） (页 1556)主题中所述，SM 1231 模块会针对断路条件或溢出条件返回模拟量输入值 32767 (16#7FFF)。如果想要确定发生的是哪个条件，可以在 STEP 7 程序中包含相关逻辑。确定条件类型的方法中包含以下任务：

- 创建诊断错误中断 OB 以供发生进入或离开诊断事件时调用。
- 包含对 RALRM 指令的调用。
- 为 AINFO 参数（包含条件类型的相关信息）创建一个字节数组。
- 在 CPU 触发诊断中断 OB 时评估 RALRM_DB 的 AINFO 结构体的字节 32 和 33。

创建诊断错误中断 OB

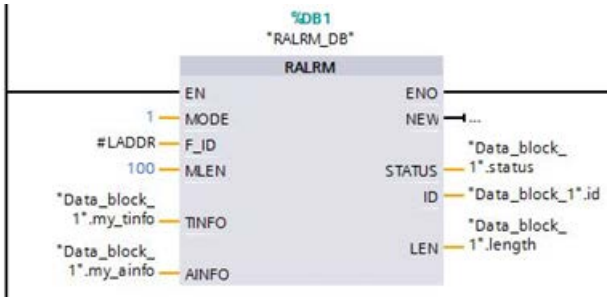
为了能够确定何时发生断线条件，需创建诊断错误中断 OB。CPU 将在发生进入或离开诊断事件时调用该 OB。

当 CPU 调用诊断错误中断 OB 时，输入参数 LADDR 将包含出错模块的硬件标识符。可以在 SM 1231 模块的 STEP 7 设备组态中找到 SM 1231 模块的硬件标识符。

调用 RALRM 指令

要编写 RALRM 指令调用，请按以下步骤操作：

1. 在 STEP 7 程序中添加对 RALRM 的调用。
2. 将 F_ID 输入参数设置为诊断错误中断 OB 的 LADDR 参数中的硬件标识符。
3. 为 TINFO 和 AINFO 输入参数创建一个字节数组。请使用 34 字节或更大的数组。



在发生诊断中断后解析 AINFO

诊断错误中断 OB 执行完毕后，AINFO 字节数组中将包含与模块诊断相关的信息。

字节 0 - 25 为标头信息。下表列出了与模块诊断相关的字节：

字节	说明	
26 和 27	字值 16#8000 - 指示该诊断属于 Profinet 类型	
28 和 29	包含负责该诊断的通道号的字	
30	位模式 aaabb000，指示通道类型 (aaa) 和错误类型 (bb)	
	aaa	bb
	000：预留	00：预留
	001：输入通道	01：进入错误
	010：输出通道	10：离开错误
	011：输入/输出通道	11：离开错误，存在其他错误

字节	说明
31	指示数据格式 0：任意数据格式 1：位 2：两个位 3：四个位 4：字节 5：字（两个字节） 6：双字（四个字节） 7：两个双字（八个字节）
32 和 33	定义错误类型的字： 16#0000：预留 16#0001：短路 16#0002：欠压 16#0003：过压 16#0004：过载 16#0005：过温 16#0006：断线 16#0007：超出上限 16#0008：超出下限 16#0009：错误

以该 AINFO 结构体的字节 26-33 为例：

29	my_ainfo[26]	Byte	16#0	16#80
30	my_ainfo[27]	Byte	16#0	16#00
31	my_ainfo[28]	Byte	16#0	16#00
32	my_ainfo[29]	Byte	16#0	16#00
33	my_ainfo[30]	Byte	16#0	16#28
34	my_ainfo[31]	Byte	16#0	16#05
35	my_ainfo[32]	Byte	16#0	16#00
36	my_ainfo[33]	Byte	16#0	16#07

- 字节 26 和 27 组合的字为 16#8000，表示该诊断属于 Profinet 类型。
- 字节 28 和 29 组合的字表示该诊断针对通道 0 或模块。
- 字节 30 为 16#28，解析为位模式 aaa bb 00 时为 001 01 000。此值表示该诊断的通道类型为输入通道，错误类型为进入错误。

- 字节 31 为 5，表示数据格式为字
- 字节 32 和 33 组合的字值为 16#0007，表示超出上限。

通过捕捉诊断错误中断事件中的 AINFO 信息，可以确定诊断事件的本质。

15.17 备份和恢复数据 CPU

15.17.1 备份与恢复选项

随着时间的推移，您可能对自动化系统进行许多更改，例如，添加新设备，更换现有设备或调整用户程序。如果这些更改导致不良系统行为，则可以将自动化设备恢复到之前的版本（已备份的情况下）。STEP 7 和 S7-1200 CPU 提供多种选项来备份与恢复硬件组态和软件。

备份选项

下表概要列出了 S7 CPU 的备份和恢复选项：

	监视值的快照	从设备上传（软件）	上传设备作为新站（硬件和软件）	从在线设备中下载备份
应用案例	恢复某个特定状态的数据块。 项目中接受的数据块实际值（含时间戳）。	将块从 CPU 上传到项目。	将硬件组态和软件从设备上传到项目。	创建 CPU 的完整备份作为恢复点。备份副本具有一致的数据，并且无法更改或打开。
要求	项目中存在 CPU。在线和离线的数据块必须完全相同。	项目中存在 CPU。	该设备在 TIA Portal 的硬件目录中可用。已安装了所有必要的 HSP 或 GSD 文件。	-
支持的模式	RUN、STOP	RUN、STOP	RUN、STOP	STOP
是否适用于 F-CPU	√	√	-	√
备份可编辑	√	√	√	-

备份内容

下表给出可以下载和备份的数据以及相关操作选项：

	监视值的快照	从设备上传（软件）	上传设备作为新站（硬件和软件）	从在线设备中下载备份
数据块的实际值	可以生成快照	可下载	可下载	可备份
软件块	-	可下载	可下载	可备份
PLC 变量（变量和常量名称）	-	可下载	可下载	可备份
工艺对象	-	可下载	可下载	可备份
硬件配置	-	-	可下载	可备份
监视表（Web 服务器）	-	-	不能下载	可备份
本地数据、位存储器、定时器、计数器和过程画面	不可生成快照	不能下载	不能下载	可备份
归档和配方 (PLC)	-	-	-	可备份
SIMATIC 存储卡上的常规数据，例如，程序块或 GSD 文件的帮助信息	-	-	-	可备份

备份实际值时的特殊注意事项

备份类型为 “从在线设备备份”(Backup from online device) 时，可备份设置为保持性的变量实际值。要保证保持性数据的一致性，在备份过程中需禁用保持性数据的所有写访问操作。

操作模式从 STOP 转换为 RUN 时，会将非保持性数据的实际值设置为起始值。CPU 备份中仅包含非保持性数据的起始值。

15.17.2 备份在线 CPU

如果您想要恢复至特定组态，创建组态备份会十分有用。可以在稍后恢复当前组态。

先决条件

可根据需要创建多个备份，存储 CPU 的不同组态。要创建备份，必须满足以下先决条件：

- 已在 STEP 7 项目中创建 CPU。
- 已将 CPU 通过自身的 PROFINET 接口直接连接到编程设备/PC 上。备份和恢复操作不支持 CM 的 PROFIBUS 接口。
- CPU 在线。（如果不存在在线连接，备份过程中会建立在线连接。）
- CPU 处于“STOP”模式。（如果 CPU 未处于 STOP 模式，备份过程会提示您允许 CPU 进入 STOP 模式。）

操作步骤

要备份 CPU 当前组态，请按以下步骤操作：

1. 在项目树中选择该 CPU。
2. 从 “在线”(Online) 菜单中，选择 “从在线设备备份”(Backup from online device) 命令。
   如果需要，必须输入密码才能对 CPU 进行读访问并确认 CPU 应进入“STOP”模式。

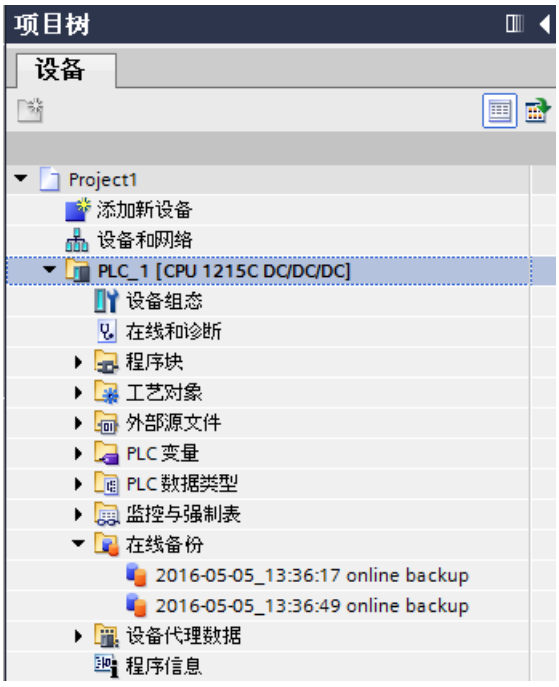
结果

根据 CPU 的名称以及备份的日期和时间对备份进行命名。备份包含恢复 CPU 的特定组态所需的所有数据。CPU 备份以下数据：

- 存储卡的内容（如果有存储卡）
- 数据块、计数器和位存储器的保持性存储区
- 其它保持性存储器内容（例如 IP 地址参数）

备份包含 CPU 的当前值，但不包含诊断缓冲区。

备份文件通常位于 CPU 项目树下的 “在线备份”(Online backups) 文件夹中。下图显示了已创建两个备份的 S7-1200 CPU：



说明

请注意，您还可以通过 SIMATIC 自动化工具 (SAT) 或 Web 服务器在线备份标准 Web 页面 (页 1056)备份在线 CPU。

通过 STEP 7 备份文件时，STEP 7 会将文件保存到 STEP 7 项目中。通过 Web 服务器备份文件时，PC 或设备会将备份文件保存到默认文件夹中，以供下载。无法通过 Web 服务器恢复 STEP 7 备份文件，也不能通过 STEP 7 恢复 Web 服务器备份文件。但是，可以将 STEP 7 备份文件直接保存到 PC 或设备的下载文件夹中。这样一来，便可以通过 Web 服务器恢复这些文件。

将备份文件保存到 PC 或设备

要将备份文件保存到 PC 或设备，请按照以下步骤操作：

1. 右键单击项目树中的“在线备份”(Online backups) 文件夹中的相应文件。
2. 从快捷菜单中选择“另存为”(Save as)。
3. 浏览到要保存文件的文件夹，例如 PC 或设备上供文件下载的默认文件夹。
4. 单击“保存”(Save)。

15.17.3 恢复 CPU

如果之前曾经备份过 CPU 组态，则可以将备份传送回该 CPU。恢复备份时，CPU 进入 STOP 模式。如果为 CPU 组态了一种访问级别，则必须提供输入密码才能对 CPU 进行读访问。

> ⚠ 警告
>
> **恢复内容未知的备份**
>
> 如果恢复内容未知的备份，则在发生故障或程序错误时，可能导致重大财产损失或人员严重受伤。
>
> 确保备份包含内容已知的组态。

先决条件

要恢复备份，必须满足以下先决条件：

- STEP 7 项目包含 CPU 的组态和之前创建的备份。
- 将 CPU 通过自身的 PROFINET 接口直接连接到编程设备/PC 上。
- CPU 处于“STOP”模式。
- 如果为 CPU 组态了访问级别，则完全访问该 CPU 时需知道相应密码。

操作步骤

要恢复备份，请按以下步骤操作：

1. 打开项目树中的 CPU 以显示较低级别的对象。
2. 从“在线备份”(Online backups) 文件夹中选择要恢复的备份文件。
3. 从“在线”(Online) 菜单中，选择“下载到设备”(Download to device) 命令。
   - 如果之前已经建立了网络连接 (页 1386)，则打开“加载预览”(Load preview) 对话框。此对话框显示一些报警并建议了加载操作所需的操作。
   - 如果之前尚未建立网络连接，则打开“延长下载到设备”(Extended download to device) 对话框，然后必须首先选择接口，将通过这些接口建立与 CPU 的在线连接。
4. 在“加载预览”(Load preview) 对话框中检查报警，然后在必要时在“操作”(Action) 列中选择操作。
5. 单击“加载”(Load) 按钮（可以下载时，“加载”(Load) 按钮即可选择。）
6. STEP 7 将备份恢复到 CPU 中。在“加载结果”(Load results) 对话框中，可以检查加载操作是否成功并执行下一步所需操作。
7. 查看“加载结果”(Load results) 对话框后，单击“完成”(Finish) 按钮。
   如果提示输入密码，则需输入密码才能对 CPU 进行完全访问并确认 CPU 可进入“STOP”模式。
   STEP 7 将备份的内容恢复到 CPU 中并重启 CPU。

说明

请注意，您还可以通过 Web 服务器在线备份标准 Web 页面 (页 1056)恢复 CPU 备份。

技术规范 A

A.1 Siemens 在线支持网站

这些产品的相关技术信息可从 Siemens 工业在线支持网站 (https://support.industry.siemens.com/cs/cn/zh/)获取。

A.2 常规技术规范

遵守的标准

S7-1200 自动化系统设计符合以下标准和测试规范。S7-1200 自动化系统的测试标准均基于这些标准和测试规范。

请注意，并非所有 S7-1200 型号都经过这些标准的认证，并且认证状态如果有变化，恕不另行通知。用户有责任通过参考产品上标记的额定值来确定适用的认证。如需更多有关按零件号排列的最新具体认证列表的信息，请咨询当地西门子代表。

CE 认证

S7-1200 自动化系统满足下列 EC 指令提出的要求和安全相关目标，并且符合欧盟的公报中列出的可编程控制器的协调欧洲标准 (EN)。

- EC 指令 2006/95/EC（低压指令）“设计用于特定电压限值内的电气设备”
  - EN 61131-2 可编程控制器 - 设备要求和测试
- EC 指令 2004/108/EC（EMC 指令）“电磁兼容性”
  - 辐射标准
    EN 61000-6:+A1：工业环境
  - 抗干扰标准
    EN 61000-6-2：工业环境
- EC 指令 94/9/EC (ATEX)“拟用于潜在爆炸性环境的设备和保护系统”
  - EN 60079-0:+A11
  - EN 60079-15：保护类型“n”

可向主管部门出具的所持 CE 一致性声明文件位于以下地址：

Siemens AG
DI FA AS

Gleiwitzer Str.555
DE-90475 Nuremberg
Germany

cULus 认证

美国保险商实验室，符合：

- 美国安全检测实验室公司：UL 508 认证（工业控制设备）
- 加拿大标准协会：CSA C22.2 第 142 号（过程控制设备）

说明

SIMATIC S7-1200 系列符合 CSA 标准。

cULus 标志表示 S7-1200 已通过美国安全检测实验室公司 (UL) 检验和认证，其符合标准 UL 508 和 CSA 22.2 第 142 号。

FM 认证

FM 认证
认证标准类别号 3600、3611 (ANSI/UL 121201) 和 3810 (ANSI/UL 61010-1)，
CSA 标准 C22.2 第 0-10 号、C22.2 第 213 号和 C22.2 第 61010-1 号
批准用于：
I 类，2 分区，组别 A、B、C、D，温度类别 T3C Ta = 60 °C [CA、US]
I 类，2 区，组别 IIC，温度类别 T3 Ta = 60 °C [US]
依据 CEC 18-150 [CA] 的加拿大 I 类、2 分区安装

重要例外：有关可同时使用输入或输出数的信息，请参见技术规范。某些型号在 Ta = 60 °C 时额定值会降低。

⚠ 警告

对于危险场所 I 类、2 分区和 2 区 而言，替换组件会影响其安全性。
只能由得到授权的 Siemens 维修中心维修设备。

IECEx 认证

IEC 60079-0：易爆环境 - 一般要求

IEC 60079-15：适用于易爆环境的电气设备
防护类型“nA”

IEC FMG 14.0012X
Ex nA IIC T3 Gc

IECEx 级别信息可能与 FM 危险位置信息一起显示在产品上。

仅批准使用标有 IECEx 级别的产品。如需更多有关按零件号排列的最新具体认证列表的信息，请咨询当地西门子代表。

IECEx 认证不适用于继电器型号。

有关温度额定值，请参见具体的产品铭牌。

根据 IEC 60079-15，在合适的外壳中安装模块可提供最低级别的 IP54 保护。

ATEX 认证

ATEX 认证仅适用于 DC 型号。ATEX 认证不适用于 AC 和继电器型号。

EN 60079-0：爆炸性环境 - 一般要求

EN 60079-15：适用于潜在易爆气体环境的电气设备；
防护类型“nA”
II 3 G Ex nA IIC T4 或 T3 Gc

确保安全使用的特殊条件：

将模块安装在合适的机柜中，根据 EN 60529 至少要提供防护等级 IP54，或安装在可提供同等防护等级的位置。

连接的电缆和导线应在额定条件下测得的实际温度下工作。

应采取措施防止电源端子的额定电压受瞬变干扰而超出 119V 以上。

澳大利亚和新西兰 - RCM 标志（法规符合性标志）

S7-1200 自动化系统满足 AS/NZS 61000.6.4 和 IEC 610000-6-4（A 类）标准的要求。

韩国认证

S7-1200 自动化系统满足韩国认证（KC 标志）的要求。已被定义为 A 类设备，适合工业应用，不适合家庭应用。

欧亚关税同盟认证（白俄罗斯、哈萨克斯坦、俄罗斯联邦）

EAC（欧亚符合性）：关税同盟 (TR CU) 技术规格的符合性声明

海事认证

S7-1200 产品定期向特定机构递交申请以便进行与特定市场和应用有关的认证。如需更多有关按零件号排列的最新具体认证列表的信息，请咨询当地西门子代表。

船级社：

- ABS（American Bureau of Shipping，美国船级社）：美国
- BV（Bureau Veritas，法国船级社）：法国
- DNV（Det Norske Veritas，挪威船级社）：挪威
- GL（Germanischer Lloyd，德国船级社）：德语
- LRS（Lloyds Register of Shipping，英国劳氏船级社）：英国
- Class NK（Nippon Kaiji Kyokai，日本船级社）：日本
- 韩国船级社：韩国
- CSS（China Classification Society，中国船级社）：中国

工业环境

S7-1200 自动化系统设计用在工业环境中。

表格 A-1　工业环境

应用现场	辐射要求	抗扰性要求
工业	EN 61000-6-4:2007+A1	EN 61000-6-2:2005

说明

S7-1200 自动化系统旨在工业区域内使用；在住宅区内使用可能会影响无线电或电视接收。如果在居民区使用 S7-1200，必须确保射频干扰强度符合 EN 55011 的 B 类限制值。

实现 RF 干扰级别 B 的有效措施示例：

- 将 S7-1200 设备安装在接地的控制机柜中
- 在供电线路中使用噪声滤波器

确保射频干扰强度符合 EN 55011 的 B 类要求。

需要单独验收（最终的安装必须满足居民区安装的所有安全和 EMC 要求）。

电磁兼容性

电磁兼容性 (EMC) 是电气设备在电磁环境中按预期运行以及运行时电磁干扰的发射水平 (EMI) 不会干扰周围其它电气设备的能力。

表格 A-2　抗扰度符合 EN 61000-6-2

电磁兼容性 - 抗扰度符合 EN 61000-6-2	
EN 61000-4-2 静电放电	8 kV，对所有表面的空中放电 6 kV，对暴露导电表面的接触放电
EN 61000-4-3 辐射、无线电频率、电磁场抗扰度测试	80 到 1000 MHz，10 V/m，1 kHz 时 80% AM 1.4 到 2.0 GHz，3 V/m，1 kHz 时 80% AM 2.0 到 2.7 GHz，1 V/m，1 kHz 时 80% AM
EN 61000-4-4 快速瞬变脉冲	2 kV，5 kHz，到交流和直流系统电源的耦合网络 2 kV，5 kHz，到 I/O 的耦合夹
EN 6100-4-5 浪涌抗扰度	交流系统 - 2 kV 共模，1 kV 差模 直流系统 - 2 kV 共模，1 kV 差模 对于直流系统，请参见下面的浪涌抗扰度
EN 61000-4-6 传导干扰	150 kHz 到 80 MHz，10 V RMS，1kHz 时 80% AM
EN 61000-4-11 电压骤降	交流系统 60 Hz 时，0% 持续 1 个周期、40% 持续 12 个周期和 70% 持续 30 个周期

表格 A-3　传导和辐射发射符合 EN 61000-6-4

电磁兼容性 - 传导和辐射发射符合 EN 61000-6-4		
传导发射 EN 55016，A 类，1 组	0.15 MHz 到 0.5 MHz	<79dB (μV) 准峰值；<66 dB (μV) 均值
	0.5 MHz 到 5 MHz	<73dB (μV) 准峰值；<60 dB (μV) 均值
	5 MHz 到 30 MHz	<73dB (μV) 准峰值；<60 dB (μV) 均值
辐射发射 EN 55016，A 类，1 组	30 MHz 到 230 MHz	<40dB (μV/m) 准峰值；测量距离为 10m
	230 MHz 到 1 GHz	<47dB (μV/m) 准峰值；测量距离为 10m
	1 GHz 到 3 GHz	< 76dB (uV/m) 准峰值；测量距离为 10m

浪涌抗扰度

受雷击浪涌耦合影响的布线系统必须配备外部保护。用于评估雷击类型浪涌保护的规范之一可以在 EN 61000-4-5 中找到，其中操作限制由 EN 61000-6-2 确定。受到此标准定义的浪涌电压影响时，S7-1200 DC CPU 和信号模块需要外部保护才能保持安全运行。

下面列出了支持所需浪涌抗扰度保护的一些设备。只有根据制造商的建议正确安装了这些设备，它们才能提供相应保护。也可以使用由其他供应商生产、技术参数相同或更佳的设备：

表格 A-4　支持抗浪涌保护的设备

子系统	保护设备
+24 V DC 电源	BLITZDUCTOR VT, BVT AVD 24，零件号 918 422
工业以太网	DEHNpatch DPA M CLE RJ45B 48，零件号 929 121
RS-485	BLITZDUCTOR XT，基座单元 BXT BAS，零件号 920 300
	BLITZDUCTOR XT，模块 BXT ML2 BD HFS 5，零件号 920 271
RS-232	BLITZDUCTOR XT，基座单元 BXT BAS，零件号 920 300
	BLITZDUCTOR XT，模块 BXT ML2 BE S 12，零件号 920 222
+24 V DC 数字量输入	DEHN, Inc.，型号 DCO SD2 E 24，零件号 917 988
+24 V DC 数字量输出和传感器电源	DEHN, Inc.，型号 DCO SD2 E 24，零件号 917 988
模拟量 IO	DEHN, Inc.，型号 DCO SD2 E 12，零件号 917 987
继电器输出	不需要

环境条件

表格 A-5　运输和存储

环境条件 - 运输和存储	
EN 60068-2-2，测试 Bb，干热和 EN 60068-2-1，测试 Ab，寒冷	-40 °C 到 +70 °C
EN 60068-2-30，测试 Db，湿热	25 ℃ 到 55 ℃，湿度 95%
EN 60068-2-14，测试 Na，温度骤变	-40 °C 到 +70 °C，停顿时间 3 小时，2 个周期
EN 60068-2-32，自由落体	0.3 m，5 次，产品包装
大气压	1140 至 660 hPa（相当于海拔 -1000 到 3500 m）

表格 A-6　气候环境条件

环境条件 - 气候环境条件	
S7-1200 自动化系统适用于不受气候影响的固定位置。运行条件符合 DIN IEC 60721-3-3 的要求： • Class 3M3（机械要求） • Class 3K3（气候要求）	
环境温度范围 （设备下部 25 mm 进风距离）	-20 °C 到 60 °C 水平安装 -20 °C 到 50 °C 垂直安装 湿度 95%，不结露 除非另有规定
大气压	1140 至 795 hPa（相当于海拔 -1000 到 2000 m）
污染物浓度	$S0_2$：< 0.5 ppm；H_2S：< 0.1 ppm；RH < 60%，不结露
	ISA-S71.04 严重度 G1、G2、G3
EN 60068-2-14，测试 Nb，温度变化	0 °C 到 60 °C
EN 60068-2-27 机械冲击	15 g，11 ms 脉冲，3 个轴向上 6 次冲击
EN 60068-2-6 正弦振动	DIN 导轨安装：5-9 Hz 时 3.5 mm，8.4 - 150 Hz 时 1G 面板安装：5-8.4 Hz 时 7.0 mm，8.4 - 150 Hz 时 2G 每个轴 10 次摆动，每分 1 倍频程

符合 IEC 61131-2 的污染等级/过压类别

- 污染等级 2
- 过压类别：II

保护等级

- 保护等级 II 符合 EN 61131-2（不需要保护导线）

防护等级

- IP20 机械保护，EN 60529
- 防止手指接触经标准探针测试出的高压。需要针对灰尘、污物、水和直径小于 12.5mm 的异物施加外部保护。

额定电压

表格 A-7　额定电压

额定电压	容错
24 V DC	20.4 V DC 到 28.8 V DC
120/230 V AC	85 V AC 到 264 V AC，47 到 63 Hz

说明

当某个机械触点将输出电源连接到 S7-200 CPU 或其它数字量扩展模块时，该触点将发送信号“1”到数字量输出并持续大约 150 ms。这可能引发意外的机械或过程操作，从而导致死亡、重伤和/或设备损坏。必须考虑这一点，尤其是使用响应短脉冲的设备时。

反向电压保护

反向电压保护电路仅应用于 +24 V DC 电源的每对端子或者 CPU、信号模块 (SM) 和信号板 (SB) 上的用户输入电源。但如果将其它端子对按相反极性接线，仍然有可能会造成系统损坏。

S7-1200 系统中的一些 24 V DC 电源输入端口是互连的，并且通过一个公共逻辑电路连接多个 M 端子。例如，在数据表中指定为“非隔离”时，以下电路是互连的：CPU 的 24 V DC

电源、CPU 的传感器电源、SM 的继电器线圈的电源输入和非隔离模拟量输入的电源。所有非隔离的 M 端子必须连接到同一个外部参考电位。

⚠ 警告
将非隔离的 M 端子连接到不同参考电位将导致意外的电流，该电流可能导致 PLC 和任何连接设备损坏或运行不确定。 不遵守这些准则可能会导致设备损坏或运行不确定，而后者可能导致死亡、人员重伤和/或财产损失。 务必确保 S7-1200 系统中的所有非隔离 M 端子都连接到同一个参考电位。

DC 输出

短路保护电路不适用于 CPU、信号模块 (SM) 和信号板 (SB) 上的 DC 输出。

继电器电气使用寿命

根据抽样试验估计的典型性能数据如下。根据具体应用，实际性能可能不同。根据负载进行调整的外部保护电路可增强触点的使用寿命。在感性负载和灯负载条件下，常闭触点的典型使用寿命约为常开触点的三分之一。

外部保护电路可以延长触点的寿命。

表格 A-8　典型性能数据

用于选择执行器的数据	
连续热电流	最大 2 A
触点的开关容量和使用寿命	

用于选择执行器的数据				
	对于电阻负载	电压	电流	操作循环数（典型值）
		24 V DC	2.0 A	10 万次
		24 V DC	1.0 A	20 万次
		24 V DC	0.5 A	100 万次
		48 V AC	1.5 A	150 万次
		60 V AC	1.5 A	150 万次
		120 V AC	2.0 A	100 万次
		120 V AC	1.0 A	150 万次
		120 V AC	0.5 A	200 万次
		230 V AC	2.0 A	100 万次
		230 V AC	1.0 A	150 万次
		230 V AC	0.5 A	200 万次

用于选择执行器的数据				
	对于感性负载（符合 IEC 947-5-1 DC13/AC15）	电压	电流	操作循环数（典型值）
		24 V DC	2.0 A	5 万次
		24 V DC	1.0 A	10 万次
		24 V DC	0.5 A	50 万次
		24 V AC	1.5 A	100 万次
		48 V AC	1.5 A	100 万次
		60 V AC	1.5 A	100 万次
		120 V AC	2.0 A	70 万次
		120 V AC	1.0 A	100 万次
		120 V AC	0.5 A	150 万次
		230 V AC	2.0 A	70 万次
		230 V AC	1.0 A	100 万次
		230 V AC	0.5 A	150 万次
激活数字量输入		可以		
切换频率				
	机械式	最大 10 Hz		
	电阻负载	最大 1 Hz		
	感性负载（符合 IEC 947-5-1 DC13/AC15）	最大 0.5 Hz		
	灯负载	最大 1Hz		

内部 CPU 内存保持性

- 保持性数据和数据日志数据的寿命：10 年
- 保持性数据掉电时，写入周期使用寿命：2 百万个周期
- 数据日志数据：写入周期使用寿命：5 亿个数据日志条目

说明

数据日志对内部 CPU 内存的影响

每次数据日志写入至少占用 2 KB 的存储空间。如果程序频繁写入少量数据，每次写入至少占用 2 KB 的存储空间。更好的方法是将小数据项累积在一个数据块 (DB) 中，并以更长的时间间隔将数据块写入到数据日志中。

如果程序以高频率写入了许多数据日志条目，请考虑使用可更换的 SD 存储卡。

A.3 PROFINET 接口 X1 端口引脚

S7-1200 CPU 使用标准母头 RJ45 插孔连接到 PROFINET 网络。连接器引脚取决于 CPU 类型。

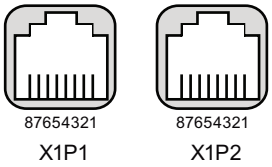
单端口 CPU

单端口 CPU（CPU 1211C、CPU 1212C 和 CPU 1214C）具有以下标准以太网 MDI 引脚组态：

引脚	信号名称	说明	RJ45 母头插孔引脚
1	TD+	传输数据	
2	TD-		
3	RD+	接收数据	
4	GND	接地	
5	GND		
6	RD-	接收数据	
7	GND	接地	
8	GND		

双端口 CPU

双端口 CPU（CPU 1215C 和 CPU1217C）的端口具有以下标准以太网 MDI-X 引脚组态：

引脚	信号名称	说明	RJ45 母头插孔引脚
1	RD+	接收数据	
2	RD-		
3	TD+	传输数据	
4	GND	接地	
5	GND		
6	TD-	传输数据	
7	GND	接地	
8	GND		

说明

双端口 CPU

引脚下方显示的数字表示双端口 CPU 的各引脚之间没有跨接。这些单元中有一个内部以太网交换机：TD+/- 和 RD+/- 对未在内部跨接。

自动协商

如果端口组态启用了自动协商功能，则 S7-1200 CPU 自动检测电缆类型并交换传输/接收线路（必要时）。如果端口组态禁用自动协商功能，则 CPU 也将禁用此自动交换功能。可以在 TIA Portal 的端口选项对话框中组态端口的自动协商设置。此为 CPU 属性 PROFINET 接口 (X1) 的端口特定的高级选项。更多相关信息，请参见 11.2.3.4 部分“在项目中为 CPU 组态 IP 地址” (页 818)的“组态 PROFINET 端口”。

A.4 CPU 1211C

A.4.1 常规规范和特性

表格 A-9 常规规范

技术数据	CPU 1211C AC/DC/继电器	CPU 1211C DC/DC/继电器	CPU 1211C DC/DC/DC
产品编号	6ES7211-1BE40-0XB0	6ES7211-1HE40-0XB0	6ES7211-1AE40-0XB0
尺寸 W x H x D (mm)	90 x 100 x 75		
装运重量	420 g	380 g	370 g
功耗	10 W	8 W	
可用电流（CM 总线）	最大 750 mA(5 V DC)		
可用电流 (24 V DC)	最大 300 mA（传感器电源）		
数字量输入电流消耗 (24 V DC)	所用的每点输入 4 mA		

表格 A-10 CPU 特征

技术数据		说明
用户存储器（请参阅“一般技术数据”(页 1427)、“内部 CPU 内存保持”。）	工作	50 KB
	负载	内置 1 MB，可用 SD 卡扩展，具体视卡容量而定
	保持性	10 KB
板载数字 I/O		6 点输入/4 点输出
板载模拟 I/O		2 路输入
过程映像大小		1024 字节输入 (I)/1024 字节输出 (Q)
位存储器 (M)		4096 个字节
临时（局部）存储器		• 16 KB 用于启动和程序循环（包括相关的 FB 和 FC） • 6 KB 用于其它各中断优先级（包括 FB 和 FC）
信号模块扩展		无
SB、CB、BB 扩展		最多 1 个

技术数据	说明
通信模块扩展	最多 3 个通信模块
高速计数器	最多可组态 6 个使用任意内置或 SB 输入的高速计数器。请参见“硬件输入引脚分配” (页 584)以了解 CPU 1211C：HSC 默认地址分配。 100/[1]80 kHz（Ia.0 到 Ia.5）
脉冲输出 [2]	最多可组态 4 个使用任意内置或 SB 输出的脉冲输出 100 kHz（Qa.0 到 Qa.3）
脉冲捕捉输入	6
延时中断	共 4 个，精度为 1 ms
循环中断	共 4 个，精度为 1 ms
沿中断	6 个上升沿和 6 个下降沿（使用可选信号板时，各为 10 个）
存储卡	SIMATIC 存储卡（选件）
实时时钟精度	+/- 60 秒/月
实时时钟保持时间	通常为 20 天，40 °C 时最少为 12 天（免维护超级电容）

[1] 将 HSC 组态为正交工作模式时，可应用较慢的速度。

[2] 对于具有继电器输出的 CPU 型号，必须安装数字信号板 (SB) 才能使用脉冲输出。

表格 A-11 性能

指令类型		执行速度	
		直接寻址（I、Q 和 M）	DB 访问
布尔运算		0.08 µs/指令	
移动	移动布尔数据	0.3 µs/指令	1.17 µs/指令
	移动字	0.137 µs/指令	1.0 µs/指令
	移动实数	0.72 µs/指令	1.0 µs/指令
实数数学运算	加上实数	1.48 µs/指令	1.78 µs/指令

说明

许多变量影响测量时间。上述性能时间适用于此类别和无错程序中的最快指令。

A.4.2 CPU 1211C 支持的定时器、计数器和代码块

表格 A-12 CPU 1211C 支持的块、定时器和计数器

元素		说明
块	类型	OB、FB、FC、DB
	大小	最大可达工作存储器的大小
	数量	最多可达 1024 个块 (OB + FB + FC + DB)
	FB、FC 和 DB 的地址范围	FB 和 FC：1 到 65535（例如 FB 1 到 FB 65535） DB：1 到 59999
	嵌套深度	16（从程序循环 OB 或启动 OB 开始） 6（从任意中断事件 OB 开始）[1]
	监视	可以同时监视 2 个代码块的状态
OB	程序循环	多个
	启动	多个
	时间延迟中断	4（每个事件 1 个）
	循环中断	4（每个事件 1 个）
	硬件中断	50（每个事件 1 个）
	时间错误中断	1
	诊断错误中断	1
	拔出或插入模块	1
	机架或站故障	1
	日时钟	多个
	状态	1
	更新	1
	配置文件	1
	MC 插补器	1
	MC 伺服电机	1
	MC-PreServo	1
	MC-PostServo	1
定时器	类型	IEC
	数量	仅受存储器大小限制
	存储	DB 结构，每个定时器 16 个字节

元素		说明
计数器	类型	IEC
	数量	仅受存储器大小限制
	存储	DB 结构，大小取决于计数类型 • SInt 和 USInt：3 个字节 • Int 和 UInt：6 个字节 • DInt 和 UDInt：12 个字节

[1] 安全程序使用二级嵌套。因此，用户程序在安全程序中的嵌套深度为四。

表格 A-13 通信

技术数据	说明
端口数	1
类型	以太网
HMI 设备	4
编程设备 (PG)	1
连接	• 8 个连接用于开放式用户通信（主动或被动）：TSEND_C、TRCV_C、TCON、TDISCON、TSEND 和 TRCV • 8 个 CPU 到 CPU 连接（客户端或服务器）用于 GET/PUT 数据 • 6 个连接用于动态分配到 GET/PUT 或开放式用户通信
数据传输率	10/100 Mb/s
隔离（外部信号与逻辑侧）	变压器隔离，1500 V AC（型式测试）[1]
电缆类型	CAT5e 屏蔽电缆
接口	
PROFINET 接口的数量	1
PROFIBUS 接口的数量	0
接口	
接口硬件	
端口数量	1
集成交换机	-
RJ-45（以太网）	√；X1
协议	
PROFINET IO 控制器	√

技术数据	说明
PROFINET IO 设备	√
SIMATIC 通信	√
开放式 IE 通信	√
Web 服务器	√
介质冗余	-
PROFINET IO 控制器	
服务	
PG/OP 通信	√
S7 路由	√
等时同步模式	-
开放式 IE 通信	√
IRT	-
MRP	-
PROFIenergy	√S7-1200 CPU 仅支持 PROFIenergy 实体（具有智能设备功能）。
优先启动	√（最多 16 个 PROFINET 设备）
可连接 I/O 设备的最大数量	16
可进行 RT 连接 I/O 设备的最大数量	16
线形结构中的最大数量	16
可同时激活/取消激活 IO 设备的最大数量	8
更新时间	最短更新时间还取决于 PROFINET IO 上设置的通信组件、IO 设备的数量以及所组态用户数据的数量。
RT 功能	
1 ms 的发送时钟	1 ms 到 512 ms
PROFINET IO 设备	
服务	
PG/OP 通信	√
S7 路由	√
等时同步模式	-
开放式 IE 通信	√
IRT，支持	-
MRP，支持	-

技术数据		说明
PROFIenergy		√
共享设备		√
共享设备的最大 IO 控制器数		2
SIMATIC 通信		
S7 通信，作为服务器		√
S7 通信，作为客户端		√
每个作业中用户数据的最大值		请参见在线帮助（S7 通信、用户数据大小）
开放式 IE 通信		
TCP/IP：		√
	最大数据长度	8 KB
	每个端口支持多个被动连接	√
ISO-on-TCP (RFC1006)：		√
	最大数据长度	8 KB
UDP		√
	最大数据长度	1472 个字节
DHCP		-
SNMP		√
DCP		√
LLDP		√

[1] 以太网端口隔离专用于在危险电压引起短期网络故障时对危险情况进行限制。它不符合常规 AC 线电压隔离的安全要求。

表格 A-14 电源

技术数据		CPU 1211C AC/DC/继电器	CPU 1211C DC/DC/继电器	CPU 1211C DC/DC/DC
电压范围		85 到 264 V AC	20.4 V DC 到 28.8 V DC	
电源频率		47 到 63 Hz	--	
输入电流	最大负载时仅包括 CPU	120 V AC 时 60 mA 240 V AC 时 30 mA	24 V DC 时 300 mA	24 V DC 时 300 mA
	最大负载时包括 CPU 和所有扩展附件	120 V AC 时 180 mA 240 V AC 时 90 mA	24 V DC 时 900 mA	
浪涌电流（最大）		264 V AC 时 20 A	28.8 V DC 时 12 A	

技术数据	CPU 1211C AC/DC/继电器	CPU 1211C DC/DC/继电器	CPU 1211C DC/DC/DC
$I^2 t$	0.8 A^2 s	0.5 A^2 s	
隔离（输入电源与逻辑侧）	1500 V AC	未隔离	
漏地电流，交流线路对功能地	最大 0.5 mA	--	
保持时间（掉电）	120 V AC 时 20 ms 240 V AC 时 80 ms	24 V DC 时 10 ms	
内部保险丝，用户不可更换	3 A，250 V，慢速熔断		

表格 A-15　传感器电源

技术数据	CPU 1211C AC/DC/继电器	CPU 1211C DC/DC/继电器	CPU 1211C DC/DC/DC
电压范围	20.4 到 28.8 V DC	L+ - 4 V DC（最小值）	
额定输出电流（最大）	300 mA（短路保护）		
最大波纹噪声 (<10 MHz)	< 1 V 峰峰值	与输入线路相同	
隔离（CPU 逻辑侧与传感器电源）	未隔离		

A.4.3　数字量输入和输出

表格 A-16　数字量输入

技术数据	CPU 1211C AC/DC/继电器、CPU 1211C DC/DC/继电器和 CPU 1211C DC/DC/DC
输入点数	6
类型	漏型/源型（IEC 1 类漏型）
额定电压	4 mA 时 24 V DC，额定值
允许的连续电压	30 V DC，最大值
浪涌电压	35 V DC，持续 0.5 s
逻辑 1 信号（最小）	2.5 mA 时 15 V DC
逻辑 0 信号（最大）	1 mA 时 5 V DC
隔离（现场侧与逻辑侧）	707 V DC（型式测试）
隔离组	1

技术数据	CPU 1211C AC/DC/继电器、CPU 1211C DC/DC/继电器和 CPU 1211C DC/DC/DC
滤波时间	us 设置：0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0 ms 设置：0.05、0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0
HSC 时钟输入频率（最大） （逻辑 1 电平 = 15 到 26 V DC）	100/80 kHz（Ia.0 到 Ia.5）
同时接通的输入数	6，60 °C（水平）或 50 °C（垂直）时
电缆长度（米）	500 m（屏蔽）；300 m（非屏蔽）；50 m（屏蔽，HSC 输入）

表格 A-17　数字量输出

技术数据	CPU 1211C AC/DC/继电器和 CPU 1211C DC/DC/继电器	CPU 1211C DC/DC/DC
输出点数	4	
类型	继电器，机械式	固态 - MOSFET（源型）
电压范围	5 到 30 V DC 或 5 到 250 V AC	20.4 到 28.8 V DC
最大电流时的逻辑 1 信号	--	20 V DC 最小
具有 10 KΩ 负载时的逻辑 0 信号	--	0.1 V DC 最大
电流（最大）	2.0 A	0.5 A
灯负载	30 W DC/200 W AC	5 W
通态电阻	新设备最大为 0.2 Ω	最大 0.6 Ω
每点的漏电流	--	最大 10 μA
浪涌电流	触点闭合时为 7 A	8 A，最长持续 100 ms
过载保护	×	
隔离（现场侧与逻辑侧）	1500 V AC （线圈与触点） 无（线圈与逻辑侧）	707 V DC（型式测试）
隔离组	1	
电感钳位电压	--	L+ - 48 V DC，1 W 损耗
继电器最大开关频率	1 Hz	--
开关延迟（Qa.0 到 Qa.3）	最长 10 ms	断开到接通最长为 1.0 μs 接通到断开最长为 3.0 μs

技术数据	CPU 1211C AC/DC/继电器和 CPU 1211C DC/DC/继电器	CPU 1211C DC/DC/DC
脉冲串输出频率	不推荐 [1]	100 kHz（Qa.0 到 Qa.3）[2]，最小 2 Hz
机械寿命（无负载）	10,000,000 个断开/闭合周期	--
额定负载下的触点寿命	100,000 个断开/闭合周期	--
RUN 到 STOP 时的行为	上一个值或替换值 （默认值为 0）	
同时接通的输出数	4，60 °C（水平）或 50 °C（垂直）时	
电缆长度（米）	500 m（屏蔽）； 150 m（非屏蔽）	

[1] 对于具有继电器输出的 CPU 型号，必须安装数字信号板 (SB) 才能使用脉冲输出。

[2] 根据所使用的脉冲接收器和电缆的情况，附加的负载电阻（至少为额定电流的 10%）可改进脉冲信号质量和抗扰度。

A.4.4 模拟量输入

表格 A-18 模拟量输入

技术数据	说明
输入点数	2
类型	电压（单侧）
满量程范围	0 到 10 V
满量程范围（数据字）	0 到 27648
过冲范围	10.001 到 11.759 V
过冲范围（数据字）	27649 到 32511
上溢范围	11.760 到 11.852 V
上溢范围（数据字）	32512 到 32767
分辨率	10 位
最大耐压	35 V DC
平滑化	无、弱、中或强 请参见 CPU 模拟量输入的阶跃响应 (ms) (页 1449)表格。
噪声抑制	10、50 或 60 Hz

技术数据	说明
阻抗	≥100 KΩ
隔离（现场侧与逻辑侧）	无
精度（25 °C/-20 到 60 °C）	满量程的 3.0%/3.5%
电缆长度（米）	100 m，屏蔽双绞线

A.4.4.1 CPU 内置模拟量输入的阶跃响应

表格 A-19 阶跃响应 (ms)，0 V 到 10 V（在 95% 处测得）

平滑化选项（采样平均）	抑制频率（积分时间）		
	60 Hz	**50 Hz**	**10 Hz**
无（1 个周期）： 不求平均值	50 ms	50 ms	100 ms
弱（4 个周期）： 4 次采样	60 ms	70 ms	200 ms
中（16 个周期）： 16 次采样	200 ms	240 ms	1150 ms
强（32 个周期）： 32 次采样	400 ms	480 ms	2300 ms
采样时间	**4.17 ms**	**5 ms**	**25 ms**

A.4.4.2 CPU 内置模拟端口的采样时间

表格 A-20 CPU 内置模拟量输入的采样时间

抑制频率（积分时间选项）	采样时间
60 Hz (16.6 ms)	4.17 ms
50 Hz (20 ms)	5 ms
10 Hz (100 ms)	25 ms

A.4.4.3 模拟量输入的电压测量范围 (CPU)

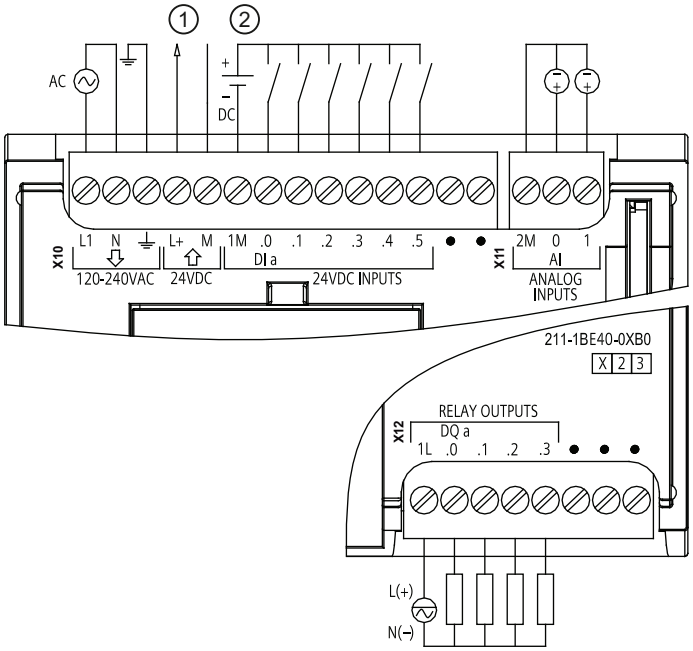
表格 A-21 模拟量输入的电压表示法 (CPU)

系统		电压测量范围	
十进制	十六进制	0 到 10 V	
32767	7FFF	11.852 V	上溢
32512	7F00		

系统		电压测量范围	
十进制	十六进制	0 到 10 V	
32511	7EFF	11.759 V	过冲范围
27649	6C01		
27648	6C00	10 V	额定范围
20736	5100	7.5 V	
34	22	12 mV	
0	0	0 V	
负值		不支持负值	

A.4.5 CPU 1211C 接线图

表格 A-22 CPU 1214C AC/DC/继电器 (6ES7211-1BE40-0XB0)



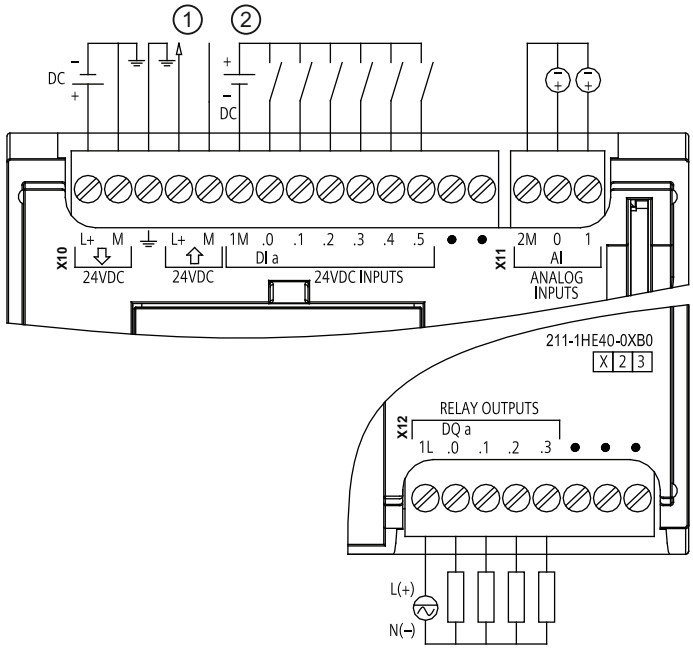
①	24 V DC 传感器电源输出 要获得更好的抗噪声效果，即使未使用传感器电源，也可将“M”连接到机壳接地。
②	对于漏型输入，将“-”连接到“M”（如图所示）。 对于源型输入，将“+”连接到“M”。
注 1：X11 连接器必须镀金。有关产品编号，请参见附录 C“备件”。	
注 2：可将 L1 或 N (L2) 端子连接到最高 240 V AC 的电压源。可将 N 端子视为 L2，无需接地。L1 和 N (L2) 端子无需极化。	
注 3：有关 CPU 以太网端口的信息，请参阅设备配置 (页 147)。	

表格 A-23 CPU 1211C AC/AC/继电器 (6ES7211-1BE40-0XB0) 的连接器引脚位置

引脚	X10	X11（镀金）	X12
1	L1 / 120-240 V AC	2 M	1L
2	N / 120-240 V AC	AI 0	DQ a.0
3	功能性接地	AI 1	DQ a.1
4	L+ / 24 V DC 传感器输出	--	DQ a.2
5	M / 24 V DC 传感器输出	--	DQ a.3

引脚	X10	X11（镀金）	X12
6	1M	--	无连接
7	DI a.0	--	无连接
8	DI a.1	--	无连接
9	DI a.2	--	--
10	DI a.3	--	--
11	DI a.4	--	--
12	DI a.5	--	--
13	无连接	--	--
14	无连接	--	--

表格 A-24　CPU 1211C DC/DC/继电器 (6ES7211-1HE40-0XB0)

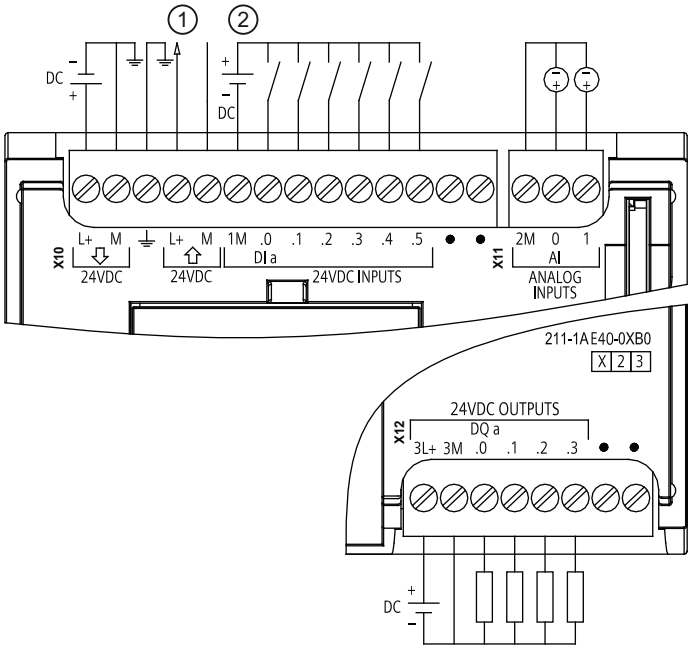


①	24 V DC 传感器电源输出 要获得更好的抗噪声效果，即使未使用传感器电源，也可将“M”连接到机壳接地。
②	对于漏型输入，将“-”连接到“M”（如图所示）。 对于源型输入，将“+”连接到“M”。
注 1：X11 连接器必须镀金。有关产品编号，请参见附录 C“备件”。	
注 2：有关 CPU 以太网端口的信息，请参阅设备配置 (页 147)。	

表格 A-25　CPU 1211C DC/DC/继电器 (6ES7211-1HE40-0XB0) 的连接器引脚位置

引脚	X10	X11（镀金）	X12
1	L+ / 24 V DC	2 M	1L
2	M / 24 V DC	AI 0	DQ a.0
3	功能性接地	AI 1	DQ a.1
4	L+ / 24 V DC 传感器输出	--	DQ a.2
5	M / 24 V DC 传感器输出	--	DQ a.3
6	1M	--	无连接
7	DI a.0	--	无连接
8	DI a.1	--	无连接
9	DI a.2	--	--
10	DI a.3	--	--
11	DI a.4	--	--
12	DI a.5	--	--
13	无连接	--	--
14	无连接	--	--

表格 A-26　CPU 1211C DC/DC/DC (6ES7211-1AE40-0XB0)



①	24 V DC 传感器电源输出 要获得更好的抗噪声效果，即使未使用传感器电源，也可将“M”连接到机壳接地。
②	对于漏型输入，将“-”连接到“M”（如图所示）。 对于源型输入，将“+”连接到“M”。
注 1：X11 连接器必须镀金。有关产品编号，请参见附录 C“备件”。	
注 2：有关 CPU 以太网端口的信息，请参阅设备配置 (页 147)。	

表格 A-27　CPU 1211C DC/DC/DC (6ES7211-1AE40-0XB0) 的连接器引脚位置

引脚	X10	X11（镀金）	X12
1	L+ / 24 V DC	2 M	3L+
2	M / 24 V DC	AI 0	3M
3	功能性接地	AI 1	DQ a.0
4	L+ / 24 V DC 传感器输出	--	DQ a.1
5	M / 24 V DC 传感器输出	--	DQ a.2
6	1M	--	DQ a.3
7	DI a.0	--	无连接
8	DI a.1	--	无连接
9	DI a.2	--	--
10	DI a.3	--	--
11	DI a.4	--	--

引脚	X10	X11（镀金）	X12
12	DI a.5	--	--
13	无连接	--	--
14	无连接	--	--

说明

应将未使用的模拟量输入短路。

A.5 CPU 1212C

A.5.1 常规规范和特性

表格 A-28 常规

技术数据	CPU 1212C AC/DC/继电器	CPU 1212C DC/DC/继电器	CPU 1212C DC/DC/DC
产品编号	6ES7212-1BE40-0XB0	6ES7212-1HE40-0XB0	6ES7212-1AE40-0XB0
尺寸 W x H x D (mm)	90 x 100 x 75		
装运重量	425 g	385 g	370 g
功耗	11 W	9 W	
可用电流（SM 和 CM 总线）	最大 1000 mA(5 V DC)		
可用电流 (24 V DC)	最大 300 mA（传感器电源）		
数字量输入电流消耗 (24 V DC)	所用的每点输入 4 mA		

表格 A-29　CPU 特征

技术数据		说明
用户存储器（请参阅“一般技术数据 (页 1427)”、“内部 CPU 内存保持”。）	工作	75 KB
	负载	内置 2 MB，可用 SD 卡扩展，具体视卡容量而定
	保持性	10 KB
板载数字 I/O		8 点输入/6 点输出
板载模拟 I/O		2 路输入
过程映像大小		1024 字节输入 (I)/1024 字节输出 (Q)
位存储器 (M)		4096 个字节
临时（局部）存储器		• 16 KB 用于启动和程序循环（包括相关的 FB 和 FC） • 6 KB 用于其它各中断优先级（包括 FB 和 FC）
信号模块扩展		最多 2 个信号模块
SB、CB、BB 扩展		最多 1 个
通信模块扩展		最多 3 个通信模块
高速计数器		最多可组态 6 个使用任意内置或 SB 输入的高速计数器。请参见“硬件输入引脚分配” (页 584)以了解 CPU 1212C：HSC 默认地址分配。 • 100/[1]80 kHz（Ia.0 到 Ia.5） • 30/[1]20 kHz（Ia.6 到 Ia.7）
脉冲输出 [2]		最多可组态 4 个使用任意内置或 SB 输出的脉冲输出 • 100 kHz（Qa.0 到 Qa.3） • 20 kHz（Qa.4 到 Qa.5）
脉冲捕捉输入		8
延时中断		共 4 个，精度为 1 ms
循环中断		共 4 个，精度为 1 ms
沿中断		8 个上升沿和 8 个下降沿（使用可选信号板时，各为 12 个）
存储卡		SIMATIC 存储卡（选件）
实时时钟精度		+/- 60 秒/月
实时时钟保持时间		通常为 20 天，40 °C 时最少为 12 天（免维护超级电容）

[1] 将 HSC 组态为正交工作模式时，可应用较慢的速度。

[2] 对于具有继电器输出的 CPU 型号，必须安装数字信号板 (SB) 才能使用脉冲输出。

表格 A-30　性能

指令类型		执行速度	
		直接寻址（I、Q 和 M）	DB 访问
布尔运算		0.08 μs/指令	
移动	移动布尔数据	0.3 μs/指令	1.17 μs/指令
	移动字	0.137 μs/指令	1.0 μs/指令
	移动实数	0.72 μs/指令	1.0 μs/指令
实数数学运算	加上实数	1.48 μs/指令	1.78 μs/指令

说明

许多变量影响测量时间。上述性能时间适用于此类别和无错程序中的最快指令。

A.5.2　CPU 1212C 支持的定时器、计数器和代码块

表格 A-31　CPU 1212C 支持的块、定时器和计数器

元素		说明
块	类型	OB、FB、FC、DB
	大小	最大可达工作存储器的大小
	数量	最多可达 1024 个块 (OB + FB + FC + DB)
	FB、FC 和 DB 的地址范围	FB 和 FC：1 到 65535（例如 FB 1 到 FB 65535） DB：1 到 59999
	嵌套深度	16（从程序循环 OB 或启动 OB 开始） 6（从任意中断事件 OB 开始）[1]
	监视	可以同时监视 2 个代码块的状态

元素		说明
OB	程序循环	多个
	启动	多个
	时间延迟中断	4（每个事件 1 个）
	循环中断	4（每个事件 1 个）
	硬件中断	50（每个事件 1 个）
	时间错误中断	1
	诊断错误中断	1
	拔出或插入模块	1
	机架或站故障	1
	日时钟	多个
	状态	1
	更新	1
	配置文件	1
	MC 插补器	1
	MC 伺服电机	1
	MC-PreServo	1
	MC-PostServo	1
定时器	类型	IEC
	数量	仅受存储器大小限制
	存储	DB 结构，每个定时器 16 个字节
计数器	类型	IEC
	数量	仅受存储器大小限制
	存储	DB 结构，大小取决于计数类型 • SInt 和 USInt：3 个字节 • Int 和 UInt：6 个字节 • DInt 和 UDInt：12 个字节

[1] 安全程序使用二级嵌套。因此，用户程序在安全程序中的嵌套深度为四。

表格 A-32　通信

技术数据	说明
端口数	1
类型	以太网
HMI 设备	4
编程设备 (PG)	1
连接	• 8 个连接用于开放式用户通信（主动或被动）：TSEND_C、TRCV_C、TCON、TDISCON、TSEND 和 TRCV • 8 个 CPU 到 CPU 连接（客户端或服务器）用于 GET/PUT 数据 • 6 个连接用于动态分配到 GET/PUT 或开放式用户通信
数据传输率	10/100 Mb/s
隔离（外部信号与逻辑侧）	变压器隔离，1500 V AC（型式测试）[1]
电缆类型	CAT5e 屏蔽电缆
接口	
PROFINET 接口的数量	1
PROFIBUS 接口的数量	0
接口	
接口硬件	
端口数量	1
集成交换机	-
RJ-45（以太网）	√；X1
协议	
PROFINET IO 控制器	√
PROFINET IO 设备	√
SIMATIC 通信	√
开放式 IE 通信	√
Web 服务器	√
介质冗余	-
PROFINET IO 控制器	
服务	
PG/OP 通信	√

技术数据	说明
S7 路由	√
等时同步模式	-
开放式 IE 通信	√
IRT	-
MRP	-
PROFIenergy	√S7-1200 CPU 仅支持 PROFIenergy 实体（具有智能设备功能）。
优先启动	√（最多 16 个 PROFINET 设备）
可连接 I/O 设备的最大数量	16
可进行 RT 连接 I/O 设备的最大数量	16
线形结构中的最大数量	16
可同时激活/取消激活 IO 设备的最大数量	8
更新时间	最短更新时间还取决于 PROFINET IO 上设置的通信组件、IO 设备的数量以及所组态用户数据的数量。
RT 功能	
1 ms 的发送时钟	1 ms 到 512 ms
PROFINET IO 设备	
服务	
PG/OP 通信	√
S7 路由	√
等时同步模式	-
开放式 IE 通信	√
IRT，支持	-
MRP，支持	-
PROFIenergy	√
共享设备	√
共享设备的最大 IO 控制器数	2
SIMATIC 通信	
S7 通信，作为服务器	√
S7 通信，作为客户端	√
每个作业中用户数据的最大值	请参见在线帮助（S7 通信、用户数据大小）
开放式 IE 通信	

技术数据		说明
TCP/IP：		√
	最大数据长度	8 KB
	每个端口支持多个被动连接	√
ISO-on-TCP (RFC1006)：		√
	最大数据长度	8 KB
UDP		√
	最大数据长度	1472 个字节
DHCP		-
SNMP		√
DCP		√
LLDP		√

[1] 以太网端口隔离专用于在危险电压引起短期网络故障时对危险情况进行限制。它不符合常规 AC 线电压隔离的安全要求。

表格 A-33 电源

技术数据		CPU 1212C AC/DC/继电器	CPU 1212C DC/DC/继电器	CPU 1212C DC/DC/DC
电压范围		85 到 264 V AC	20.4 V DC 到 28.8 V DC	
电源频率		47 到 63 Hz	--	
输入电流（最大负载时）	仅 CPU	120 V AC 时 80 mA 240 V AC 时 40 mA	24 V DC 时 400 mA	
	具有所有扩展附件的 CPU	120 V AC 时 240 mA 240 V AC 时 120 mA	24 V DC 时 1200 mA	
浪涌电流（最大）		264 V AC 时 20 A	28.8 V DC 时 12 A	
I^2t		0.8 A^2 s	0.5 A^2 s	
隔离（输入电源与逻辑侧）		1500 V AC	未隔离	
漏地电流，交流线路对功能地		最大 0.5 mA	--	
保持时间（掉电）		120 V AC 时 20 ms 240 V AC 时 80 ms	24 V DC 时 10 ms	
内部保险丝，用户不可更换		3 A，250 V，慢速熔断		

表格 A-34　传感器电源

技术数据	CPU 1212C AC/DC/继电器	CPU 1212C DC/DC/继电器	CPU 1212C DC/DC/DC
电压范围	20.4 到 28.8 V DC	L+ - 4 V DC 最小值	
额定输出电流（最大）	300 mA（短路保护）		
最大波纹噪声 (<10 MHz)	< 1 V 峰峰值	与输入线路相同	
隔离（CPU 逻辑侧与传感器电源）	未隔离		

A.5.3　数字量输入和输出

表格 A-35　数字量输入

技术数据	CPU 1212C AC/DC/继电器、DC/DC/继电器和 DC/DC/DC
输入点数	8
类型	漏型/源型（IEC 1 类漏型）
额定电压	4 mA 时 24 V DC，额定值
允许的连续电压	30 V DC，最大值
浪涌电压	35 V DC，持续 0.5 s
逻辑 1 信号（最小）	2.5 mA 时 15 V DC
逻辑 0 信号（最大）	1 mA 时 5 V DC
隔离（现场侧与逻辑侧）	707 V DC（型式测试）
隔离组	1
滤波时间	us 设置：0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0 ms 设置：0.05、0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0
HSC 时钟输入频率（最大） （逻辑 1 电平 = 15 到 26 V DC）	100/80 kHz（Ia.0 到 Ia.5） 30 /20 kHz（Ia.6 到 Ia.7）
同时接通的输入数	4（无相邻点），60 ℃（水平）或 50 ℃（垂直）时 8，55 °C（水平）或 45 °C（垂直）时
电缆长度（米）	500 m（屏蔽）；300 m（非屏蔽）；50 m（屏蔽，HSC 输入）

表格 A-36　数字量输出

技术数据	CPU 1212C AC/DC/继电器 和 DC/DC/继电器	CPU 1212C DC/DC/DC
输出点数	6	
类型	继电器，机械式	固态 - MOSFET（源型）
电压范围	5 到 30 V DC 或 5 到 250 V AC	20.4 到 28.8 V DC
最大电流时的逻辑 1 信号	--	20 V DC 最小
具有 10 KΩ 负载时的逻辑 0 信号	--	0.1 V DC 最大
电流（最大）	2.0 A	0.5 A
灯负载	30 W DC/200 W AC	5 W
通态电阻	新设备最大为 0.2 Ω	最大 0.6 Ω
每点的漏电流	--	最大 10 μA
浪涌电流	触点闭合时为 7 A	8 A，最长持续 100 ms
过载保护	×	
隔离（现场侧与逻辑侧）	1500 V AC（线圈与触点） 无（线圈与逻辑侧）	707 V DC（型式测试）
隔离组	2	1
隔离（组间）	1500 V AC[1]	--
电感钳位电压	--	L+ - 48 V DC，1 W 损耗
开关延迟（Qa.0 到 Qa.3）	最长 10 ms	断开到接通最长为 1.0 μs 接通到断开最长为 3.0 μs
开关延迟（Qa.4 到 Qa.5）	最长 10 ms	断开到接通最长为 5 μs 接通到断开最长为 20 μs
继电器最大开关频率	1 Hz	--
脉冲串输出频率	不推荐 [2]	100 kHz（Qa.0 到 Qa.3）[3]，最小 2 Hz 20 kHz（Qa.4 到 Qa.5）[3]
机械寿命（无负载）	10,000,000 个断开/闭合周期	--
额定负载下的触点寿命	100,000 个断开/闭合周期	--
RUN 到 STOP 时的行为	上一个值或替换值 （默认值为 0）	上一个值或替换值（默认值为 0）

技术数据	CPU 1212C AC/DC/继电器 和 DC/DC/继电器	CPU 1212C DC/DC/DC
同时接通的输出数	3（无相邻点），60 ℃（水平）或 50 ℃（垂直）时 6，55 °C（水平）或 45 °C（垂直）时	
电缆长度（米）	500 m（屏蔽）； 150 m（非屏蔽）	

[1] 继电器的组间隔离可将线电压与 SELV/PELV 分离，并将高达 250 V AC 线对地电压的各个相分离。

[2] 对于具有继电器输出的 CPU 型号，必须安装数字信号板 (SB) 才能使用脉冲输出。

[3] 根据脉冲接收器和电缆，附加的负载电阻（至少为额定电流的 10%）可改善脉冲信号质量和抗扰性。

A.5.4 模拟量输入

表格 A-37 模拟量输入

技术数据	说明
输入点数	2
类型	电压（单侧）
满量程范围	0 到 10 V
满量程范围（数据字）	0 到 27648
过冲范围	10.001 到 11.759 V
过冲范围（数据字）	27649 到 32511
上溢范围	11.760 到 11.852 V
上溢范围（数据字）	32512 到 32767
分辨率	10 位
最大耐压	35 V DC
平滑化	无、弱、中或强 请参见 CPU 模拟量输入的阶跃响应 (ms) (页 1465)表格。
噪声抑制	10、50 或 60 Hz
阻抗	≥100 KΩ
隔离（现场侧与逻辑侧）	无
精度（25 °C/-20 到 60 °C）	满量程的 3.0%/3.5%
电缆长度（米）	100 m，屏蔽双绞线

A.5.4.1 CPU 内置模拟量输入的阶跃响应

表格 A-38 阶跃响应 (ms)，0 V 到 10 V（在 95% 处测得）

平滑化选项（采样平均）	抑制频率（积分时间）		
	60 Hz	**50 Hz**	**10 Hz**
无（1 个周期）： 不求平均值	50 ms	50 ms	100 ms
弱（4 个周期）： 4 次采样	60 ms	70 ms	200 ms
中（16 个周期）： 16 次采样	200 ms	240 ms	1150 ms
强（32 个周期）： 32 次采样	400 ms	480 ms	2300 ms
采样时间	**4.17 ms**	**5 ms**	**25 ms**

A.5.4.2 CPU 内置模拟端口的采样时间

表格 A-39 CPU 内置模拟量输入的采样时间

抑制频率（积分时间选项）	采样时间
60 Hz (16.6 ms)	4.17 ms
50 Hz (20 ms)	5 ms
10 Hz (100 ms)	25 ms

A.5.4.3 模拟量输入的电压测量范围 (CPU)

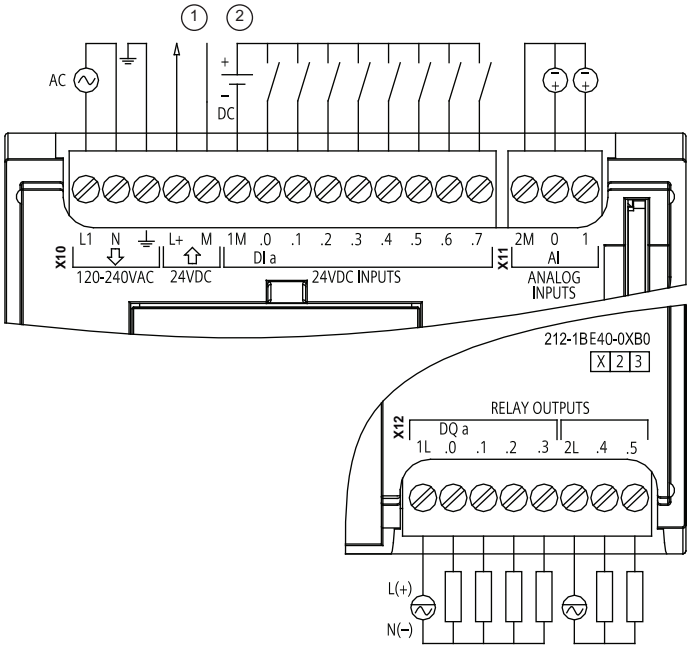
表格 A-40 模拟量输入的电压表示法 (CPU)

系统		电压测量范围	
十进制	十六进制	**0 到 10 V**	
32767	7FFF	11.852 V	上溢
32512	7F00		
32511	7EFF	11.759 V	过冲范围
27649	6C01		

系统		电压测量范围	
十进制	十六进制	0 到 10 V	
27648	6C00	10 V	额定范围
20736	5100	7.5 V	
34	22	12 mV	
0	0	0 V	
负值		不支持负值	

A.5.5 CPU 1212C 接线图

表格 A-41 CPU 1212C AC/DC/继电器 (6ES7212-1BE40-0XB0)

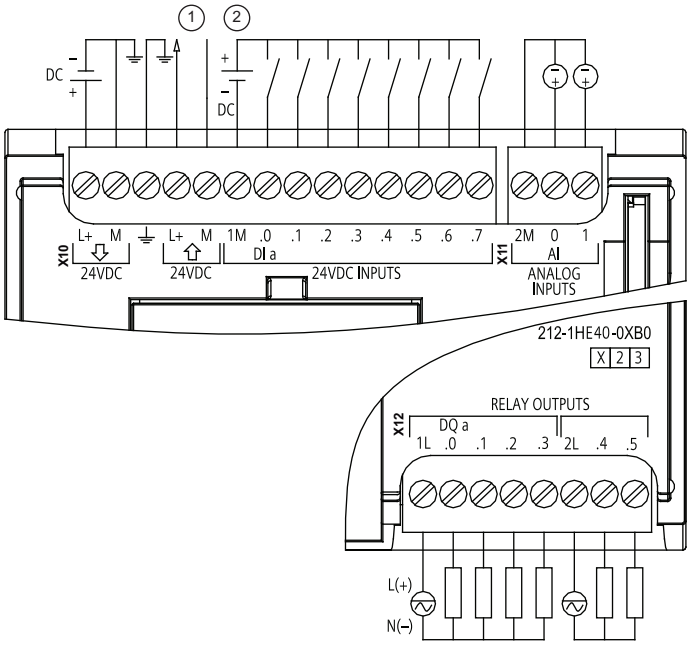


①	24 V DC 传感器电源输出 要获得更好的抗噪声效果，即使未使用传感器电源，也可将“M”连接到机壳接地。
②	对于漏型输入，将“-”连接到“M”（如图所示）。 对于源型输入，将“+”连接到“M”。
注 1：X11 连接器必须镀金。有关产品编号，请参见附录 C“备件”。	
注 2：可将 L1 或 N (L2) 端子连接到最高 240 V AC 的电压源。可将 N 端子视为 L2，无需接地。L1 和 N (L2) 端子无需极化。	
注 3：有关 CPU 以太网端口的信息，请参阅设备配置 (页 147)。	

表格 A-42　CPU 1212C AC/DC/继电器 (6ES7212-1BE40-0XB0) 的连接器引脚位置

引脚	X10	X11（镀金）	X12
1	L1 / 120-240 V AC	2 M	1L
2	N / 120-240 V AC	AI 0	DQ a.0
3	功能性接地	AI 1	DQ a.1
4	L+ / 24 V DC 传感器输出	--	DQ a.2
5	M / 24 V DC 传感器输出	--	DQ a.3
6	1M	--	2L
7	DI a.0	--	DQ a.4
8	DI a.1	--	DQ a.5
9	DI a.2	--	--
10	DI a.3	--	--
11	DI a.4	--	--
12	DI a.5	--	--
13	DI a.6	--	--
14	DI a.7	--	--

表格 A-43　CPU 1212C DC/DC/继电器 (6ES7212-1HE40-0XB0)



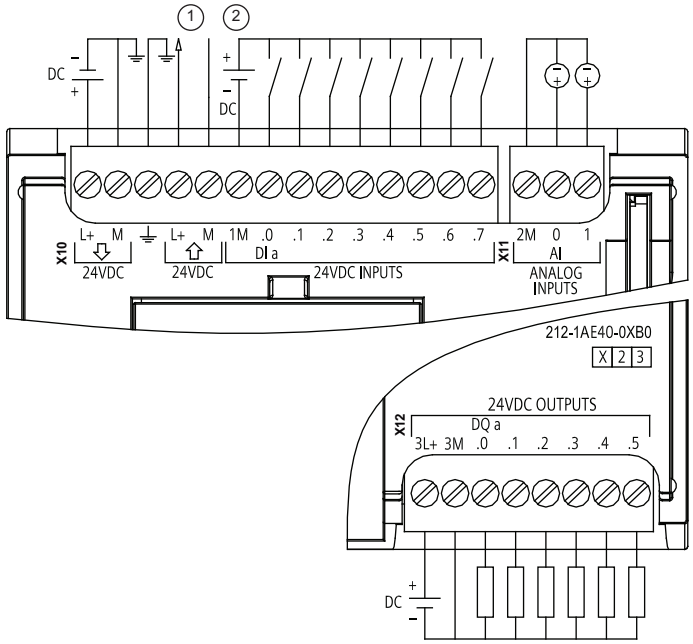
①	24 V DC 传感器电源输出 要获得更好的抗噪声效果，即使未使用传感器电源，也可将“M”连接到机壳接地。
②	对于漏型输入，将“-”连接到“M”（如图所示）。 对于源型输入，将“+”连接到“M”。
注 1：X11 连接器必须镀金。有关产品编号，请参见附录 C“备件”。	
注 2：有关 CPU 以太网端口的信息，请参阅设备配置 (页 147)。	

表格 A-44　CPU 1212C DC/DC/继电器 (6ES7212-1HE40-0XB0) 的连接器引脚位置

引脚	X10	X11（镀金）	X12
1	L+ / 24 V DC	2 M	1L
2	M / 24 V DC	AI 0	DQ a.0
3	功能性接地	AI 1	DQ a.1
4	L+ / 24 V DC 传感器输出	--	DQ a.2
5	M / 24 V DC 传感器输出	--	DQ a.3
6	1M	--	2L
7	DI a.0	--	DQ a.4
8	DI a.1	--	DQ a.5
9	DI a.2	--	--
10	DI a.3	--	--
11	DI a.4	--	--

引脚	X10	X11（镀金）	X12
12	DI a.5	--	--
13	DI a.6	--	--
14	DI a.7	--	--

表格 A-45　CPU 1212C DC/DC/DC (6ES7212-1AE40-0XB0)



①	24 V DC 传感器电源输出 要获得更好的抗噪声效果，即使未使用传感器电源，也可将“M”连接到机壳接地。
②	对于漏型输入，将“-”连接到“M”（如图所示）。 对于源型输入，将“+”连接到“M”。
注 1：X11 连接器必须镀金。有关产品编号，请参见附录 C“备件”。	
注 2：有关 CPU 以太网端口的信息，请参阅设备配置 (页 147)。	

表格 A-46　CPU 1212C DC/DC/DC (6ES7212-1AE40-0XB0) 的连接器引脚位置

引脚	X10	X11（镀金）	X12
1	L+ / 24 V DC	2 M	3L+
2	M / 24 V DC	AI 0	3M
3	功能性接地	AI 1	DQ a.0
4	L+ / 24 V DC 传感器输出	--	DQ a.1
5	M / 24 V DC 传感器输出	--	DQ a.2

引脚	X10	X11（镀金）	X12
6	1M	--	DQ a.3
7	DI a.0	--	DQ a.4
8	DI a.1	--	DQ a.5
9	DI a.2	--	--
10	DI a.3	--	--
11	DI a.4	--	--
12	DI a.5	--	--
13	DI a.6	--	--
14	DI a.7	--	--

说明

应将未使用的模拟量输入短路。

A.6 CPU 1214C

A.6.1 常规规范和特性

表格 A-47 常规

技术数据	CPU 1214C AC/DC/继电器	CPU 1214C DC/DC/继电器	CPU 1214C DC/DC/DC
产品编号	6ES7214-1BG40-0XB0	6ES7214-1HG40-0XB0	6ES7214-1AG40-0XB0
尺寸 W x H x D (mm)	110 x 100 x 75		
装运重量	475 g	435 g	415 g
功耗	14 W	12 W	
可用电流（SM 和 CM 总线）	最大 1600 mA(5 V DC)		
可用电流 (24 V DC)	最大 400 mA（传感器电源）		
数字量输入电流消耗 (24 V DC)	所用的每点输入 4 mA		

表格 A-48　CPU 特征

技术数据		说明
用户存储器（请参阅“一般技术数据”、(页 1427)“内部 CPU 内存保持”。）	工作	100 KB
	负载	内置 4 MB，可用 SD 卡扩展，具体视卡容量而定
	保持性	10 KB
板载数字 I/O		14 点输入/10 点输出
板载模拟 I/O		2 路输入
过程映像大小		1024 字节输入 (I)/1024 字节输出 (Q)
位存储器 (M)		8192 个字节
临时（局部）存储器		• 16 KB 用于启动和程序循环（包括相关的 FB 和 FC） • 6 KB 用于其它各中断优先级（包括 FB 和 FC）
信号模块扩展		最多 8 个信号模块
SB、CB、BB 扩展		最多 1 个
通信模块扩展		最多 3 个通信模块
高速计数器		最多可组态 6 个使用任意内置或 SB 输入的高速计数器。请参见“硬件输入引脚分配” (页 584)以了解 CPU 1214C：HSC 默认地址分配。 • 100/[1]80 kHz（Ia.0 到 Ia.5） • 30/[1]20 kHz（Ia.6 到 Ib.5）
脉冲输出 [2]		最多可组态 4 个使用任意内置或 SB 输出的脉冲输出 • 100 kHz（Qa.0 到 Qa.3） • 20 kHz（Qa.4 到 Qb.1）
脉冲捕捉输入		14
延时中断		共 4 个，精度为 1 ms
循环中断		共 4 个，精度为 1 ms
沿中断		12 个上升沿和 12 个下降沿（使用可选信号板时，各为 16 个）
存储卡		SIMATIC 存储卡（选件）
实时时钟精度		+/- 60 秒/月
实时时钟保持时间		通常为 20 天，40 °C 时最少为 12 天（免维护超级电容）

[1] 将 HSC 组态为正交工作模式时，可应用较慢的速度。

[2] 对于具有继电器输出的 CPU 型号，必须安装数字信号板 (SB) 才能使用脉冲输出。

表格 A-49　性能

指令类型		执行速度	
		直接寻址（I、Q 和 M）	DB 访问
布尔运算		0.08 μs/指令	
移动	移动布尔数据	0.3 μs/指令	1.17 μs/指令
	移动字	0.137 μs/指令	1.0 μs/指令
	移动实数	0.72 μs/指令	1.0 μs/指令
实数数学运算	加上实数	1.48 μs/指令	1.78 μs/指令

说明

许多变量影响测量时间。上述性能时间适用于此类别和无错程序中的最快指令。

A.6.2　CPU 1214C 支持的定时器、计数器和代码块

表格 A-50　CPU 1214C 支持的块、定时器和计数器

元素		说明
块	类型	OB、FB、FC、DB
	大小	OB、FB、FC：64 KB DB：最大可达工作存储器的大小
	数量	最多可达 1024 个块 (OB + FB + FC + DB)
	FB、FC 和 DB 的地址范围	FB 和 FC：1 到 65535（例如 FB 1 到 FB 65535） DB：1 到 59999
	嵌套深度	16（从程序循环 OB 或启动 OB 开始） 6（从任意中断事件 OB 开始）[1]
	监视	可以同时监视 2 个代码块的状态

元素		说明
OB	程序循环	多个
	启动	多个
	延时中断	4（每个事件 1 个）
	循环中断	4（每个事件 1 个）
	硬件中断	50（每个事件 1 个）
	时间错误中断	1
	诊断错误中断	1
	拔出或插入模块	1
	机架或站故障	1
	日时钟	多个
	状态	1
	更新	1
	配置文件	1
	MC 插补器	1
	MC 伺服电机	1
	MC-PreServo	1
	MC-PostServo	1
定时器	类型	IEC
	数量	仅受存储器大小限制
	存储	DB 结构，每个定时器 16 个字节
计数器	类型	IEC
	数量	仅受存储器大小限制
	存储	DB 结构，大小取决于计数类型 • SInt 和 USInt：3 个字节 • Int 和 UInt：6 个字节 • DInt 和 UDInt：12 个字节

[1] 安全程序使用二级嵌套。因此，用户程序在安全程序中的嵌套深度为四。

表格 A-51　通信

技术数据	说明
端口数	1
类型	以太网
HMI 设备	4
编程设备 (PG)	1
连接	• 8 个连接用于开放式用户通信（主动或被动）：TSEND_C、TRCV_C、TCON、TDISCON、TSEND 和 TRCV • 8 个 CPU 到 CPU 连接（客户端或服务器）用于 GET/PUT 数据 • 6 个连接用于动态分配到 GET/PUT 或开放式用户通信
数据传输率	10/100 Mb/s
隔离（外部信号与逻辑侧）	变压器隔离，1500 V AC（型式测试）[1]
电缆类型	CAT5e 屏蔽电缆
接口	
PROFINET 接口的数量	1
PROFIBUS 接口的数量	0
接口	
接口硬件	
端口数量	1
集成交换机	-
RJ-45（以太网）	√；X1
协议	
PROFINET IO 控制器	√
PROFINET IO 设备	√
SIMATIC 通信	√
开放式 IE 通信	√
Web 服务器	√
介质冗余	-
PROFINET IO 控制器	
服务	
PG/OP 通信	√

技术数据	说明
S7 路由	√
等时同步模式	-
开放式 IE 通信	√
IRT	-
MRP	-
PROFIenergy	√S7-1200 CPU 仅支持 PROFIenergy 实体（具有智能设备功能）。
优先启动	√（最多 16 个 PROFINET 设备）
可连接 I/O 设备的最大数量	16
可进行 RT 连接 I/O 设备的最大数量	16
线形结构中的最大数量	16
可同时激活/取消激活 IO 设备的最大数量	8
更新时间	最短更新时间还取决于 PROFINET IO 上设置的通信组件、IO 设备的数量以及所组态用户数据的数量。
RT 功能	
1 ms 的发送时钟	1 ms 到 512 ms
PROFINET IO 设备	
服务	
PG/OP 通信	√
S7 路由	√
等时同步模式	-
开放式 IE 通信	√
IRT，支持	-
MRP，支持	-
PROFIenergy	√
共享设备	√
共享设备的最大 IO 控制器数	2
SIMATIC 通信	
S7 通信，作为服务器	√
S7 通信，作为客户端	√
每个作业中用户数据的最大值	请参见在线帮助（S7 通信、用户数据大小）
开放式 IE 通信	

技术数据		说明
TCP/IP：		√
	最大数据长度	8 KB
	每个端口支持多个被动连接	√
ISO-on-TCP (RFC1006)：		√
	最大数据长度	8 KB
UDP		√
	最大数据长度	1472 个字节
DHCP		-
SNMP		√
DCP		√
LLDP		√

[1] 以太网端口隔离专用于在危险电压引起短期网络故障时对危险情况进行限制。它不符合常规 AC 线电压隔离的安全要求。

表格 A-52 电源

技术数据		CPU 1214C AC/DC/继电器	CPU 1214C DC/DC/继电器	CPU 1214C DC/DC/DC
电压范围		85 到 264 V AC	20.4 V DC 到 28.8 V DC	
电源频率		47 到 63 Hz	--	
输入电流（最大负载时）	仅 CPU	120 V AC 时 100 mA 240 V AC 时 50 mA	24 V DC 时 500 mA	
	具有所有扩展附件的 CPU	120 V AC 时 300 mA 240 V AC 时 150 mA	24 V DC 时 1500 mA	
浪涌电流（最大）		264 V AC 时 20 A	28.8 V DC 时 12 A	
I^2t		0.8 A^2 s	0.5 A^2 s	
隔离（输入电源与逻辑侧）		1500 V AC	未隔离	
漏地电流，交流线路对功能地		最大 0.5 mA	-	
保持时间（掉电）		120 V AC 时 20 ms 240 V AC 时 80 ms	24 V DC 时 10 ms	
内部保险丝，用户不可更换		3 A，250 V，慢速熔断		

表格 A-53　传感器电源

技术数据	CPU 1214C AC/DC/继电器	CPU 1214C DC/DC/继电器	CPU 1214C DC/DC/DC
电压范围	20.4 到 28.8 V DC	L+ - 4 V DC（最小值）	
额定输出电流（最大）	400 mA（短路保护）		
最大波纹噪声 (<10 MHz)	< 1 V 峰峰值	与输入线路相同	
隔离（CPU 逻辑侧与传感器电源）	未隔离		

A.6.3　数字量输入和输出

表格 A-54　数字量输入

技术数据	CPU 1214C AC/DC/继电器	CPU 1214C DC/DC/继电器	CPU 1214C DC/DC/DC
输入点数	14		
类型	漏型/源型（IEC 1 类漏型）		
额定电压	4 mA 时 24 V DC，额定值		
允许的连续电压	30 V DC，最大值		
浪涌电压	35 V DC，持续 0.5 s		
逻辑 1 信号（最小）	2.5 mA 时 15 V DC		
逻辑 0 信号（最大）	1 mA 时 5 V DC		
隔离（现场侧与逻辑侧）	707 V DC（型式测试）		
隔离组	1		
滤波时间	us 设置：0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0 ms 设置：0.05、0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0		
HSC 时钟输入频率（最大） （逻辑 1 电平 = 15 到 26 V DC）	100/80 kHz（Ia.0 到 Ia.5） 30/20 kHz（Ia.6 到 Ib.5）		
同时接通的输入数	• 7（无相邻点），60 ℃（水平）或 50 ℃（垂直）时 • 14，55 ℃（水平）或 45 ℃（垂直）时		
电缆长度（米）	500 m（屏蔽）；300 m（非屏蔽）；50 m（屏蔽，HSC 输入）		

表格 A-55　数字量输出

技术数据	CPU 1214C AC/DC/继电器 和 DC/DC/继电器	CPU 1214C DC/DC/DC
输出点数	10	
类型	继电器，机械式	固态 - MOSFET（源型）
电压范围	5 到 30 V DC 或 5 到 250 V AC	20.4 到 28.8 V DC
最大电流时的逻辑 1 信号	--	20 V DC 最小
具有 10 KΩ 负载时的逻辑 0 信号	--	0.1 V DC 最大
电流（最大）	2.0 A	0.5 A
灯负载	30 W DC/200 W AC	5 W
通态电阻	新设备最大为 0.2 Ω	最大 0.6 Ω
每点的漏电流	--	最大 10 μA
浪涌电流	触点闭合时为 7 A	8 A，最长持续 100 ms
过载保护	×	
隔离（现场侧与逻辑侧）	1500 V AC（线圈与触点） 无（线圈与逻辑侧）	707 V DC（型式测试）
隔离组	2	1
隔离（组间）	1500 V AC[1]	--
电感钳位电压	--	L+ - 48 V DC， 1 W 损耗
开关延迟（Qa.0 到 Qa.3）	最长 10 ms	断开到接通最长为 1.0 μs 接通到断开最长为 3.0 μs
开关延迟（Qa.4 到 Qb.1）	最长 10 ms	断开到接通最长为 5 μs 接通到断开最长为 20 μs
继电器最大开关频率	1 Hz	--
脉冲串输出频率	不推荐 [2]	100 kHz（Qa.0 到 Qa.3）[3]，最小 2 Hz 20 kHz（Qa.4 到 Qb.1）[3]
机械寿命（无负载）	10,000,000 个断开/闭合周期	--
额定负载下的触点寿命	100,000 个断开/闭合周期	--
RUN 到 STOP 时的行为	上一个值或替换值（默认值为 0）	
同时接通的输出数	• 5（无相邻点），60 ℃（水平）或 50 ℃（垂直）时 • 10，55 ℃（水平）或 45 ℃（垂直）时	
电缆长度（米）	500 m（屏蔽）；150 m（非屏蔽）	

[1] 继电器的组间隔离可将线电压与 SELV/PELV 分离，并将高达 250 V AC 线对地电压的各个相分离。

[2] 对于具有继电器输出的 CPU 型号，必须安装数字信号板 (SB) 才能使用脉冲输出。

[3] 根据所使用的脉冲接收器和电缆的情况，附加的负载电阻（至少为额定电流的 10%）可改进脉冲信号质量和抗扰度。

A.6.4 模拟量输入

表格 A-56 模拟量输入

技术数据	说明
输入点数	2
类型	电压（单侧）
满量程范围	0 到 10 V
满量程范围（数据字）	0 到 27648
过冲范围	10.001 到 11.759 V
过冲范围（数据字）	27649 到 32511
上溢范围	11.760 到 11.852 V
上溢范围（数据字）	32512 到 32767
分辨率	10 位
最大耐压	35 V DC
平滑化	无、弱、中或强 请参见 CPU 模拟量输入的阶跃响应 (ms) (页 1480)表格。
噪声抑制	10、50 或 60 Hz
阻抗	≥100 KΩ
隔离（现场侧与逻辑侧）	无
精度（25 °C/-20 到 60 °C）	满量程的 3.0%/3.5%
电缆长度（米）	100 m，屏蔽双绞线

A.6.4.1 CPU 内置模拟量输入的阶跃响应

表格 A-57 阶跃响应 (ms)，0 V 到 10 V（在 95% 处测得）

平滑化选项（采样平均）	抑制频率（积分时间）		
	60 Hz	50 Hz	10 Hz
无（1 个周期）： 不求平均值	50 ms	50 ms	100 ms
弱（4 个周期）： 4 次采样	60 ms	70 ms	200 ms
中（16 个周期）： 16 次采样	200 ms	240 ms	1150 ms
强（32 个周期）： 32 次采样	400 ms	480 ms	2300 ms
采样时间	4.17 ms	5 ms	25 ms

A.6.4.2 CPU 内置模拟端口的采样时间

表格 A-58 CPU 内置模拟量输入的采样时间

抑制频率（积分时间选项）	采样时间
60 Hz (16.6 ms)	4.17 ms
50 Hz (20 ms)	5 ms
10 Hz (100 ms)	25 ms

A.6.4.3 模拟量输入的电压测量范围 (CPU)

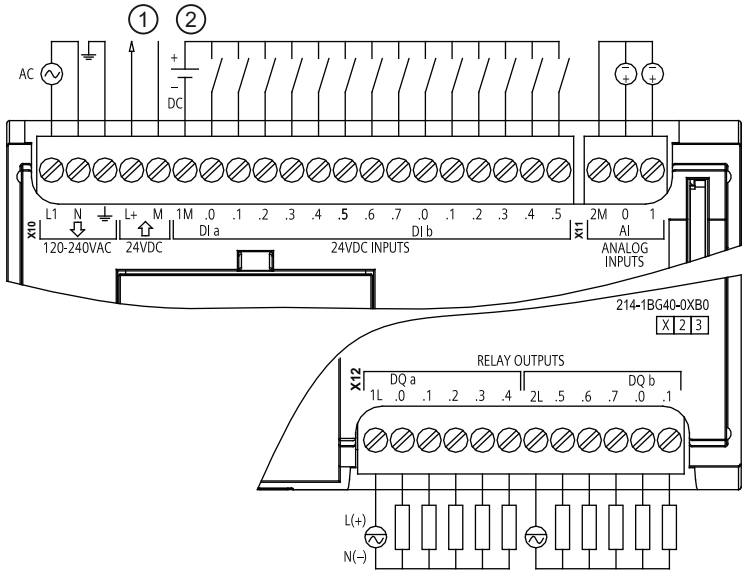
表格 A-59 模拟量输入的电压表示法 (CPU)

系统		电压测量范围	
十进制	十六进制	0 到 10 V	
32767	7FFF	11.852 V	上溢
32512	7F00		
32511	7EFF	11.759 V	过冲范围
27649	6C01		

系统		电压测量范围	
十进制	十六进制	0 到 10 V	
27648	6C00	10 V	额定范围
20736	5100	7.5 V	
34	22	12 mV	
0	0	0 V	
负值		不支持负值	

A.6.5 CPU 1214C 接线图

表格 A-60 CPU 1214C AC/DC/继电器 (6ES7214-1BG40-0XB0)

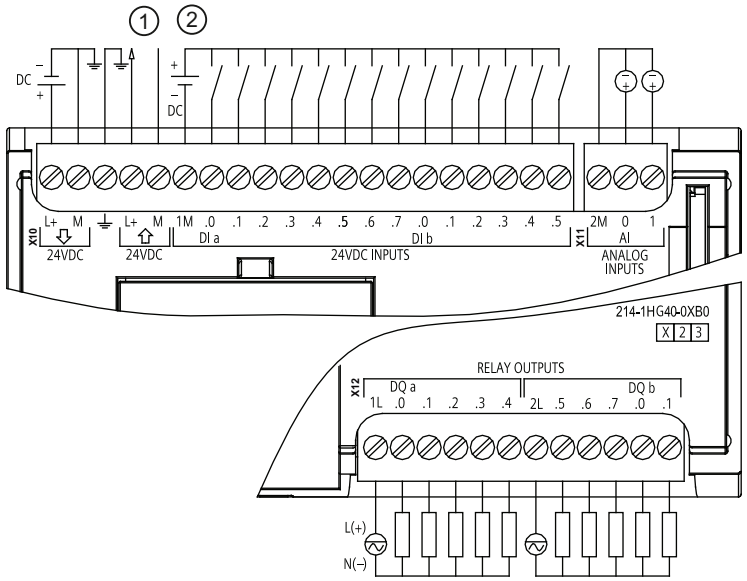


①	24 V DC 传感器电源输出 要获得更好的抗噪声效果，即使未使用传感器电源，也可将“M”连接到机壳接地。
②	对于漏型输入，将“-”连接到“M”（如图所示）。 对于源型输入，将“+”连接到“M”。
注 1：X11 连接器必须镀金。有关产品编号，请参见附录 C“备件”。	
注 2：可将 L1 或 N (L2) 端子连接到最高 240 V AC 的电压源。可将 N 端子视为 L2，无需接地。L1 和 N (L2) 端子无需极化。	
注 3：有关 CPU 以太网端口的信息，请参阅设备配置 (页 147)。	

表格 A-61　CPU 1214C AC/DC/继电器 (6ES7214-1BG40-0XB0) 的连接器引脚位置

引脚	X10	X11（镀金）	X12
1	L1 / 120-240 V AC	2 M	1L
2	N / 120-240 V AC	AI 0	DQ a.0
3	功能性接地	AI 1	DQ a.1
4	L+ / 24 V DC 传感器输出	--	DQ a.2
5	M / 24 V DC 传感器输出	--	DQ a.3
6	1M	--	DQ a.4
7	DI a.0	--	2L
8	DI a.1	--	DQ a.5
9	DI a.2	--	DQ a.6
10	DI a.3	--	DQ a.7
11	DI a.4	--	DQ b.0
12	DI a.5	--	DQ b.1
13	DI a.6	--	--
14	DI a.7	--	--
15	DI b.0	--	--
16	DI b.1	--	--
17	DI b.2	--	--
18	DI b.3	--	--
19	DI b.4	--	--
20	DI b.5	--	--

表格 A-62　CPU 1214C DC/DC/继电器 (6ES7214-1HG40-0XB0)



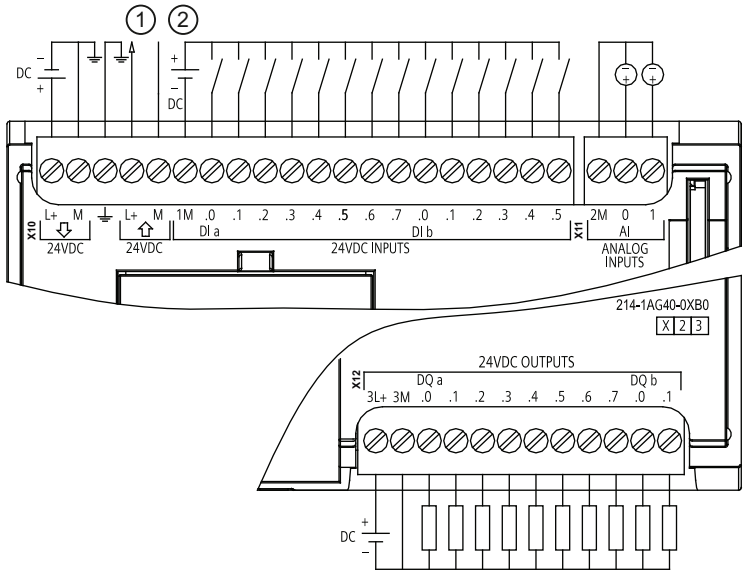
①	24 V DC 传感器电源输出 要获得更好的抗噪声效果，即使未使用传感器电源，也可将“M”连接到机壳接地。
②	对于漏型输入，将“-”连接到“M”（如图所示）。 对于源型输入，将“+”连接到“M”。
注 1：X11 连接器必须镀金。有关产品编号，请参见附录 C“备件”。	
注 2：有关 CPU 以太网端口的信息，请参阅设备配置 (页 147)。	

表格 A-63　CPU 1214C DC/DC/继电器 (6ES7214-1HG40-0XB0) 的连接器引脚位置

引脚	X10	X11（镀金）	X12
1	L+ / 24 V DC	2 M	1L
2	M / 24 V DC	AI 0	DQ a.0
3	功能性接地	AI 1	DQ a.1
4	L+ / 24 V DC 传感器输出	--	DQ a.2
5	M / 24 V DC 传感器输出	--	DQ a.3
6	1M	--	DQ a.4
7	DI a.0	--	2L
8	DI a.1	--	DQ a.5
9	DI a.2	--	DQ a.6
10	DI a.3	--	DQ a.7
11	DI a.4	--	DQ b.0

引脚	X10	X11（镀金）	X12
12	DI a.5	--	DQ b.1
13	DI a.6	--	--
14	DI a.7	--	--
15	DI b.0	--	--
16	DI b.1	--	--
17	DI b.2	--	--
18	DI b.3	--	--
19	DI b.4	--	--
20	DI b.5	--	--

表格 A-64　CPU 1214C DC/DC/DC (6ES7214-1AG40-0XB0)



①	24 V DC 传感器电源输出 要获得更好的抗噪声效果，即使未使用传感器电源，也可将“M”连接到机壳接地。
②	对于漏型输入，将“-”连接到“M”（如图所示）。 对于源型输入，将“+”连接到“M”。
注 1：X11 连接器必须镀金。有关产品编号，请参见附录 C“备件”。	
注 2：有关 CPU 以太网端口的信息，请参阅设备配置 (页 147)。	

表格 A-65　CPU 1214C DC/DC/DC (6ES7214-1AG40-0XB0) 的连接器引脚位置

引脚	X10	X11（镀金）	X12
1	L+ / 24 V DC	2 M	3L+
2	M / 24 V DC	AI 0	3M
3	功能性接地	AI 1	DQ a.0
4	L+ / 24 V DC 传感器输出	--	DQ a.1
5	M / 24 V DC 传感器输出	--	DQ a.2
6	1M	--	DQ a.3
7	DI a.0	--	DQ a.4
8	DI a.1	--	DQ a.5
9	DI a.2	--	DQ a.6
10	DI a.3	--	DQ a.7
11	DI a.4	--	DQ b.0
12	DI a.5	--	DQ b.1
13	DI a.6	--	--
14	DI a.7	--	-
15	DI b.0	--	--
16	DI b.1	--	--
17	DI b.2	--	--
18	DI b.3	--	--
19	DI b.4	--	--
20	DI b.5	--	--

说明

应将未使用的模拟量输入短路。

A.7 CPU 1215C

A.7.1 常规规范和特性

表格 A-66 常规

技术数据	CPU 1215C AC/DC/继电器	CPU 1215C DC/DC/继电器	CPU 1215C DC/DC/DC
产品编号	6ES7215-1BG40-0XB0	6ES7215-1HG40-0XB0	6ES7215-1AG40-0XB0
尺寸 W x H x D (mm)	130 x 100 x 75		
装运重量	585 g	550 g	520 g
功耗	14 W	12 W	
可用电流（SM 和 CM 总线）	最大 1600 mA(5 V DC)		
可用电流 (24 V DC)	最大 400 mA（传感器电源）		
数字量输入电流消耗 (24 V DC)	所用的每点输入 4 mA		

表格 A-67 CPU 特征

技术数据		说明
用户存储器（请参见“一般技术数据 (页 1427)”，“内部 CPU 存储器掉电保持”。）	工作	125 KB
	负载	内置 4 MB，可用 SD 卡扩展，具体视卡容量而定
	保持性	10 KB
板载数字 I/O		14 点输入/10 点输出
板载模拟 I/O		2 点输入/2 点输出
过程映像大小		1024 字节输入 (I)/1024 字节输出 (Q)
位存储器 (M)		8192 个字节
临时（局部）存储器		• 16 KB 用于启动和程序循环（包括相关的 FB 和 FC） • 6 KB 用于其它各中断优先级（包括 FB 和 FC）
信号模块扩展		最多 8 个信号模块
SB、CB、BB 扩展		最多 1 个

技术数据	说明
通信模块扩展	最多 3 个通信模块
高速计数器	最多可组态 6 个使用任意内置或 SB 输入的高速计数器。请参见“硬件输入引脚分配” (页 584)以了解 CPU 1215C：HSC 默认地址分配。 • 100/[1]80 kHz（Ia.0 到 Ia.5） • 30/[1]20 kHz（Ia.6 到 Ib.5）
脉冲输出 [2]	最多可组态 4 个使用任意内置或 SB 输出的脉冲输出 • 100 kHz（Qa.0 到 Qa.3） • 20 kHz（Qa.4 到 Qb.1）
脉冲捕捉输入	14
延时中断	共 4 个，精度为 1 ms
循环中断	共 4 个，精度为 1 ms
沿中断	12 个上升沿和 12 个下降沿（使用可选信号板时，各为 16 个）
存储卡	SIMATIC 存储卡（选件）
实时时钟精度	+/- 60 秒/月
实时时钟保持时间	通常为 20 天，40 °C 时最少为 12 天（免维护超级电容）

[1] 将 HSC 组态为正交工作模式时，可应用较慢的速度。

[2] 对于具有继电器输出的 CPU 型号，必须安装数字信号板 (SB) 才能使用脉冲输出。

表格 A-68 性能

指令类型		执行速度	
		直接寻址（I、Q 和 M）	DB 访问
布尔运算		0.08 µs/指令	
移动	移动布尔数据	0.3 µs/指令	1.17 µs/指令
	移动字	0.137 µs/指令	1.0 µs/指令
	移动实数	0.72 µs/指令	1.0 µs/指令
实数数学运算	加上实数	1.48 µs/指令	1.78 µs/指令

说明

许多变量影响测量时间。上述性能时间适用于此类别和无错程序中的最快指令。

A.7.2 CPU 1215C 支持的定时器、计数器和代码块

表格 A-69 CPU 1215C 支持的块、定时器和计数器

元素		描述
块	类型	OB、FB、FC、DB
	大小	OB、FB、FC：64 KB DB：最大可达工作存储器的大小
	数量	最多可达 1024 个块 (OB + FB + FC + DB)
	FB、FC 和 DB 的地址范围	FB 和 FC：1 到 65535（例如 FB 1 到 FB 65535） DB：1 到 59999
	嵌套深度	16（从程序循环 OB 或启动 OB 开始） 6（从任意中断事件 OB 开始）[1]
	监视	可以同时监视 2 个代码块的状态
OB	程序循环	多个
	启动	多个
	延时中断	4（每个事件 1 个）
	循环中断	4（每个事件 1 个）
	硬件中断	50（每个事件 1 个）
	时间错误中断	1
	诊断错误中断	1
	拔出或插入模块	1
	机架或站故障	1
	日时钟	多个
	状态	1
	更新	1
	配置文件	1
	MC 插补器	1
	MC 伺服电机	1
	MC-PreServo	1
	MC-PostServo	1

元素		描述
定时器	类型	IEC
	数量	仅受存储器大小限制
	存储	DB 结构，每个定时器 16 个字节
计数器	类型	IEC
	数量	仅受存储器大小限制
	存储	DB 结构，大小取决于计数类型 • SInt 和 USInt：3 个字节 • Int 和 UInt：6 个字节 • DInt 和 UDInt：12 个字节

[1] 安全程序使用二级嵌套。因此，用户程序在安全程序中的嵌套深度为四。

表格 A-70 通信

技术数据	描述
端口数	2
类型	以太网
HMI 设备	4
编程设备 (PG)	1
连接	• 8 个连接用于开放式用户通信（主动或被动）：TSEND_C、TRCV_C、TCON、TDISCON、TSEND 和 TRCV • 8 个 CPU 到 CPU 连接（客户端或服务器）用于 GET/PUT 数据 • 6 个连接用于动态分配到 GET/PUT 或开放式用户通信
数据传输率	10/100 Mb/s
隔离（外部信号与逻辑侧）	变压器隔离，1500 V AC（型式测试）[1]
电缆类型	CAT5e 屏蔽电缆
接口	
PROFINET 接口的数量	1
PROFIBUS 接口的数量	0
接口	
接口硬件	
端口数量	2
集成交换机	√

技术数据	描述
RJ-45（以太网）	√；X1
协议	
PROFINET IO 控制器	√
PROFINET IO 设备	√
SIMATIC 通信	√
开放式 IE 通信	√
Web 服务器	√
介质冗余	√
PROFINET IO 控制器	
服务	
PG/OP 通信	√
S7 路由	√
等时同步模式	-
开放式 IE 通信	√
IRT	-
MRP	√；作为 MRP 客户端
PROFIenergy	√S7-1200 CPU 仅支持 PROFIenergy 实体（具有智能设备功能）。
优先启动	√（最多 16 个 PROFINET 设备）
可连接 I/O 设备的最大数量	16
可进行 RT 连接 I/O 设备的最大数量	16
线形结构中的最大数量	16
可同时激活/取消激活 IO 设备的最大数量	8
更新时间	最短更新时间还取决于 PROFINET IO 上设置的通信组件、IO 设备的数量以及所组态用户数据的数量。
RT 功能	
1 ms 的发送时钟	1 ms 到 512 ms
PROFINET IO 设备	
服务	
PG/OP 通信	√
S7 路由	√
等时同步模式	-

技术数据		描述
开放式 IE 通信		√
IRT，支持		-
MRP，支持		√
PROFIenergy		√
共享设备		√
共享设备的最大 IO 控制器数		2
SIMATIC 通信		
S7 通信，作为服务器		√
S7 通信，作为客户端		√
每个作业中用户数据的最大值		请参见在线帮助（S7 通信、用户数据大小）
开放式 IE 通信		
TCP/IP：		√
	最大数据长度	8 KB
	每个端口支持多个被动连接	√
ISO-on-TCP (RFC1006)：		√
	最大数据长度	8 KB
UDP：		√
	最大数据长度	1472 个字节
DHCP		-
SNMP		√
DCP		√
LLDP		√

[1] 以太网端口隔离专用于在危险电压引起短期网络故障时对危险情况进行限制。它不符合常规 AC 线电压隔离的安全要求。

表格 A-71 电源

技术数据	CPU 1215C AC/DC/继电器	CPU 1215C DC/DC/继电器	CPU 1215C DC/DC/DC
电压范围	85 到 264 V AC	20.4 V DC 到 28.8 V DC	
电源频率	47 到 63 Hz	--	

技术数据		CPU 1215C AC/DC/继电器	CPU 1215C DC/DC/继电器	CPU 1215C DC/DC/DC
输入电流（最大负载时）	仅 CPU	120 V AC 时 100 mA 240 V AC 时 50 mA	24 V DC 时 500 mA	
	具有所有扩展附件的 CPU	120 V AC 时 300 mA 240 V AC 时 150 mA	24 V DC 时 1500 mA	
浪涌电流（最大）		264 V AC 时 20 A	28.8 V DC 时 12 A	
I^2t		0.8 A^2 s	0.5 A^2 s	
隔离（输入电源与逻辑侧）		1500 V AC	未隔离	
漏地电流，交流线路对功能地		最大 0.5 mA	-	
保持时间（掉电）		120 V AC 时 20 ms 240 V AC 时 80 ms	24 V DC 时 10 ms	
内部保险丝，用户不可更换		3 A，250 V，慢速熔断		

表格 A-72　传感器电源

技术数据	CPU 1215C AC/DC/继电器	CPU 1215C DC/DC/继电器	CPU 1215C DC/DC/DC
电压范围	20.4 到 28.8 V DC	L+ - 4 V DC（最小值）	
额定输出电流（最大）	400 mA（短路保护）		
最大波纹噪声 (<10 MHz)	< 1 V 峰峰值	与输入线路相同	
隔离（CPU 逻辑侧与传感器电源）	未隔离		

A.7.3　数字量输入和输出

表格 A-73　数字量输入

技术数据	CPU 1215C AC/DC/继电器	CPU 1215C DC/DC/继电器	CPU 1215C DC/DC/DC
输入点数	14		
类型	漏型/源型（IEC 1 类漏型）		
额定电压	4 mA 时 24 V DC，额定值		
允许的连续电压	30 V DC，最大值		
浪涌电压	35 V DC，持续 0.5 s		

技术数据	CPU 1215C AC/DC/继电器	CPU 1215C DC/DC/继电器	CPU 1215C DC/DC/DC
逻辑 1 信号（最小）	2.5 mA 时 15 V DC		
逻辑 0 信号（最大）	1 mA 时 5 V DC		
隔离（现场侧与逻辑侧）	707 V DC（型式测试）		
隔离组	1		
滤波时间	us 设置：0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0 ms 设置：0.05、0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0		
HSC 时钟输入频率（最大） （逻辑 1 电平 = 15 到 26 V DC）	100/80 kHz（Ia.0 到 Ia.5） 30/20 kHz（Ia.6 到 Ib.5）		
同时接通的输入数	• 7（无相邻点），60 ℃（水平）或 50 ℃（垂直）时 • 14，55 ℃（水平）或 45 ℃（垂直）时		
电缆长度（米）	500 m（屏蔽）；300 m（非屏蔽）；50 m（屏蔽，HSC 输入）		

表格 A-74　数字量输出

技术数据	CPU 1215C AC/DC/继电器 和 CPU 1215C DC/DC/继电器	CPU 1215C DC/DC/DC
输出点数	10	
类型	继电器，机械式	固态 - MOSFET（源型）
电压范围	5 到 30 V DC 或 5 到 250 V AC	20.4 到 28.8 V DC
最大电流时的逻辑 1 信号	--	20 V DC 最小
具有 10 KΩ 负载时的逻辑 0 信号	--	0.1 V DC 最大
电流（最大）	2.0 A	0.5 A
灯负载	30 W DC/200 W AC	5 W
通态电阻	新设备最大为 0.2 Ω	最大 0.6 Ω
每点的漏电流	--	最大 10 μA
浪涌电流	触点闭合时为 7 A	8 A，最长持续 100 ms
过载保护	×	
隔离（现场侧与逻辑侧）	1500 V AC（线圈与触点） 无（线圈与逻辑侧）	707 V DC（型式测试）
隔离组	2	1
隔离（组间）	1500 V AC[1]	--

技术数据	CPU 1215C AC/DC/继电器 和 CPU 1215C DC/DC/继电器	CPU 1215C DC/DC/DC
电感钳位电压	--	L+ - 48 V DC， 1 W 损耗
开关延迟（Qa.0 到 Qa.3）	最长 10 ms	断开到接通最长为 1.0 μs 接通到断开最长为 3.0 μs
开关延迟（Qa.4 到 Qb.1）	最长 10 ms	断开到接通最长为 5 μs 接通到断开最长为 20 μs
继电器最大开关频率	1 Hz	--
脉冲串输出频率	不推荐 [2]	100 kHz（Qa.0 到 Qa.3）[3]，最小 2 Hz 20 kHz（Qa.4 到 Qb.1）[3]
机械寿命（无负载）	10,000,000 个断开/闭合周期	--
额定负载下的触点寿命	100,000 个断开/闭合周期	--
RUN 到 STOP 时的行为	上一个值或替换值（默认值为 0）	
同时接通的输出数	• 5（无相邻点），60 ℃（水平）或 50 ℃（垂直）时 • 10，55 ℃（水平）或 45 ℃（垂直）时	
电缆长度（米）	500 m（屏蔽）；150 m（非屏蔽）	

[1] 继电器的组间隔离可将线路电压与 SELV/PELV 分开，并将高达 250 V AC 线对地电压的各个相分开。

[2] 对于具有继电器输出的 CPU 型号，必须安装数字信号板 (SB) 才能使用脉冲输出。

[3] 根据所使用的脉冲接收器和电缆的情况，附加的负载电阻（至少为额定电流的 10%）可改进脉冲信号质量和抗扰度。

A.7.4 模拟量输入和输出

表格 A-75 模拟量输入

技术数据	说明
输入点数	2
类型	电压（单侧）
满量程范围	0 到 10 V
满量程范围（数据字）	0 到 27648
过冲范围	10.001 到 11.759 V
过冲范围（数据字）	27649 到 32511

技术数据	说明
上溢范围	11.760 到 11.852 V
上溢范围（数据字）	32512 到 32767
分辨率	10 位
最大耐压	35 V DC
平滑化	无、弱、中或强 请参见 CPU 模拟量输入的阶跃响应 (ms) (页 1495)表格。
噪声抑制	10、50 或 60 Hz
阻抗	≥100 KΩ
隔离（现场侧与逻辑侧）	无
精度（25 °C/-20 到 60 °C）	满量程的 3.0%/3.5%
电缆长度（米）	100 m，屏蔽双绞线

A.7.4.1 CPU 内置模拟量输入的阶跃响应

表格 A-76 阶跃响应 (ms)，0 V 到 10 V（在 95% 处测得）

平滑化选项（采样平均）	抑制频率（积分时间）		
	60 Hz	**50 Hz**	**10 Hz**
无（1 个周期）： 不求平均值	50 ms	50 ms	100 ms
弱（4 个周期）： 4 次采样	60 ms	70 ms	200 ms
中（16 个周期）： 16 次采样	200 ms	240 ms	1150 ms
强（32 个周期）： 32 次采样	400 ms	480 ms	2300 ms
采样时间	**4.17 ms**	**5 ms**	**25 ms**

A.7.4.2 CPU 内置模拟端口的采样时间

表格 A-77 CPU 内置模拟量输入的采样时间

抑制频率（积分时间选项）	采样时间
60 Hz (16.6 ms)	4.17 ms
50 Hz (20 ms)	5 ms
10 Hz (100 ms)	25 ms

A.7.4.3 模拟量输入的电压测量范围 (CPU)

表格 A-78 模拟量输入的电压表示法 (CPU)

系统		电压测量范围	
十进制	十六进制	0 到 10 V	
32767	7FFF	11.852 V	上溢
32512	7F00		
32511	7EFF	11.759 V	过冲范围
27649	6C01		
27648	6C00	10 V	额定范围
20736	5100	7.5 V	
34	22	12 mV	
0	0	0 V	
负值		不支持负值	

A.7.4.4 模拟量输出规格

表格 A-79 模拟量输出

技术数据	说明
输出点数	2
类型	电流
满量程范围	0 到 20 mA
满量程范围（数据字）	0 到 27648
过冲范围	20.01 到 23.52 mA
过冲范围（数据字）	27649 到 32511
上溢范围	参见脚注 [1]
上溢范围数据字	32512 到 32767
分辨率	10 位
输出驱动阻抗	最大 500 Ω
隔离（现场侧与逻辑侧）	无
精度（25 °C/-20 到 60 °C）	满量程的 3.0%/3.5%

技术数据	说明
稳定时间	2 ms
电缆长度（米）	100 m，屏蔽双绞线

[1] 在上溢情况下，模拟量输出的行为将符合设备组态属性设置。 在“对 CPU STOP 的响应”参数中，选择其中一项：“使用替换值”或“保持上一个值”。

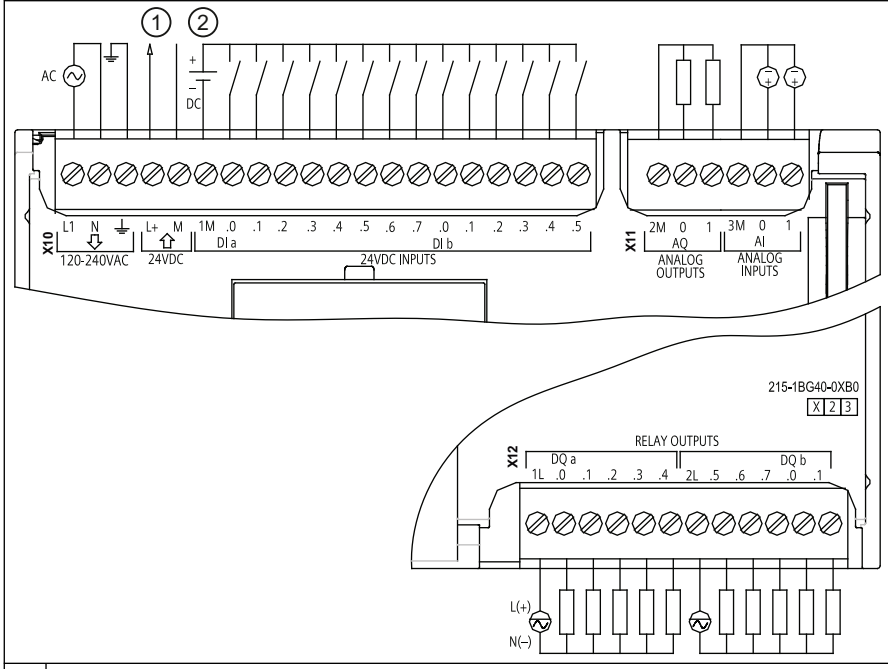
表格 A-80 模拟量输出的电流表示法（CPU 1215C 和 CPU 1217C）

系统		当前输出范围	
十进制	十六进制	0 mA 到 20 mA	
32767	7FFF	请参见注 1	上溢
32512	7F00	请参见注 1	
32511	7EFF	23.52 mA	过冲范围
27649	6C01		
27648	6C00	20 mA	额定范围
20736	5100	15 mA	
34	22	0.0247 mA	
0	0	0 mA	
负值		不支持负值	

[1] 在上溢情况下，模拟量输出的行为将符合设备组态属性设置。 在“对 CPU STOP 的响应”参数中，选择其中一项：“使用替换值”或“保持上一个值”。

A.7.5 CPU 1215C 接线图

表格 A-81 CPU 1215C AC/DC/继电器 (6ES7215-1BG40-0XB0)



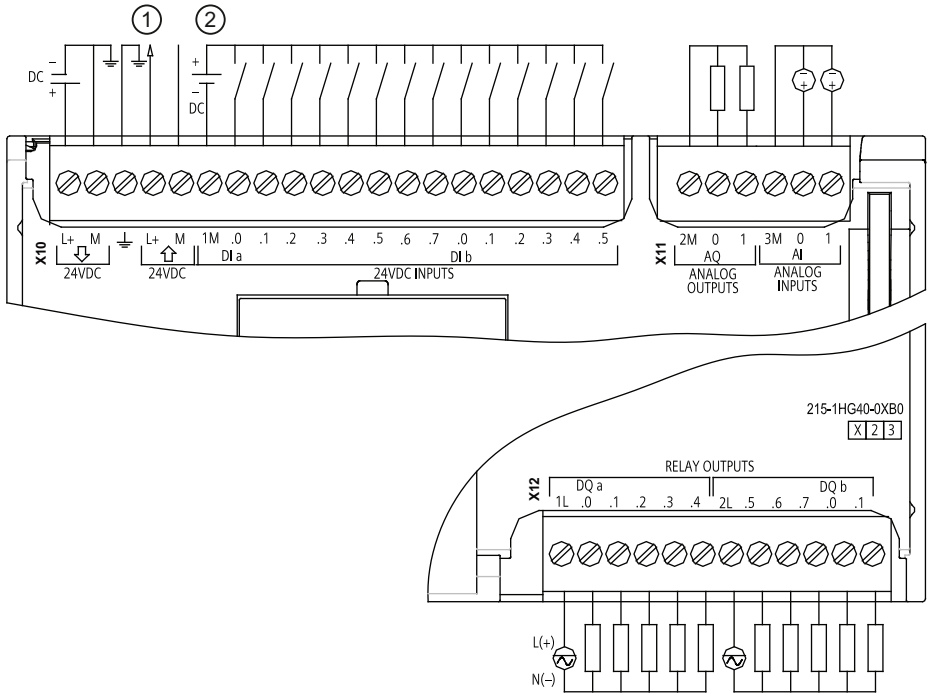
①	24 V DC 传感器电源输出 要获得更好的抗噪声效果，即使未使用传感器电源，也可将“M”连接到机壳接地。
②	对于漏型输入，将“-”连接到“M”（如图所示）。 对于源型输入，将“+”连接到“M”。
注 1：X11 连接器必须镀金。有关产品编号，请参见附录 C“备件”。	
注 2：可将 L1 或 N (L2) 端子连接到最高 240 V AC 的电压源。可将 N 端子视为 L2，无需接地。L1 和 N (L2) 端子无需极化。	
注 3：有关 CPU 以太网端口的信息，请参见“设备配置 (页 147)”。	

表格 A-82 CPU 1215C AC/DC/继电器 (6ES7215-1BG40-0XB0) 的连接器引脚位置

引脚	X10	X11（镀金）	X12
1	L1 / 120-240 V AC	2 M	1L
2	N / 120 - 240 V AC	AQ 0	DQ a.0
3	功能性接地	AQ 1	DQ a.1
4	L+ / 24 V DC 传感器输出	3M	DQ a.2
5	M / 24 V DC 传感器输出	AI 0	DQ a.3

引脚	X10	X11（镀金）	X12
6	1M	AI 1	DQ a.4
7	DI a.0	--	2L
8	DI a.1	--	DQ a.5
9	DI a.2	--	DQ a.6
10	DI a.3	--	DQ a.7
11	DI a.4	--	DQ b.0
12	DI a.5	--	DQ b.1
13	DI a.6	--	--
14	DI a.7	--	--
15	DI b.0	--	--
16	DI b.1	--	--
17	DI b.2	--	--
18	DI b.3	--	--
19	DI b.4	--	--
20	DI b.5	--	--

表格 A-83　CPU 1215C DC/DC/继电器 (6ES7215-1HG40-0XB0)



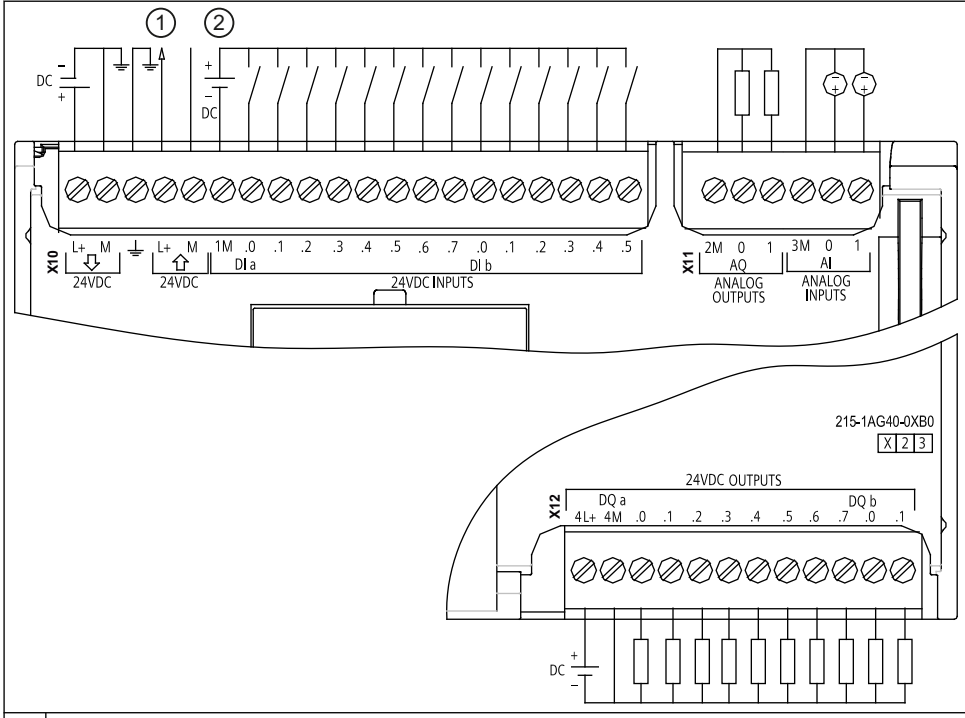
①	24 V DC 传感器电源输出 要获得更好的抗噪声效果，即使未使用传感器电源，也可将“M”连接到机壳接地。
②	对于漏型输入，将“-”连接到“M”（如图所示）。对于源型输入，将“+”连接到“M”。
注 1：X11 连接器必须镀金。有关产品编号，请参见附录 C“备件”。	
注 2：有关 CPU 以太网端口的信息，请参见“设备配置 (页 147)”。	

表格 A-84　CPU 1215C DC/DC/继电器 (6ES7215-1HG40-0XB0) 的连接器引脚位置

引脚	X10	X11（镀金）	X12
1	L+ / 24 V DC	2 M	1L
2	M / 24 V DC	AQ 0	DQ a.0
3	功能性接地	AQ 1	DQ a.1
4	L+ / 24 V DC 传感器输出	3M	DQ a.2
5	M / 24 V DC 传感器输出	AI 0	DQ a.3
6	1M	AI 1	DQ a.4
7	DI a.0	--	2L
8	DI a.1	--	DQ a.5
9	DI a.2	--	DQ a.6

引脚	X10	X11（镀金）	X12
10	DI a.3	--	DQ a.7
11	DI a.4	--	DQ b.0
12	DI a.5	--	DQ b.1
13	DI a.6	--	--
14	DI a.7	--	--
15	DI b.0	--	--
16	DI b.1	--	--
17	DI b.2	--	--
18	DI b.3	--	--
19	DI b.4	--	--
20	DI b.5	--	--

表格 A-85　CPU 1215C DC/DC/DC (6ES7215-1AG40-0XB0)



①	24 V DC 传感器电源输出 要获得更好的抗噪声效果，即使未使用传感器电源，也可将“M”连接到机壳接地。
②	对于漏型输入，将“-”连接到“M”（如图所示）。对于源型输入，将“+”连接到“M”。

注 1：X11 连接器必须镀金。有关产品编号，请参见附录 C“备件”。
注 2：有关 CPU 以太网端口的信息，请参见“设备配置 (页 147)”。

表格 A-86 CPU 1215C DC/DC/DC (6ES7215-1AG40-0XB0) 的连接器引脚位置

引脚	X10	X11（镀金）	X12
1	L1 / 24 V DC	2 M	4L+
2	M / 24 V DC	AQ 0	4M
3	功能性接地	AQ 1	DQ a.0
4	L+ / 24 V DC 传感器输出	3M	DQ a.1
5	M / 24 V DC 传感器输出	AI 0	DQ a.2
6	1M	AI 1	DQ a.3
7	DI a.0	--	DQ a.4
8	DI a.1	--	DQ a.5
9	DI a.2	--	DQ a.6
10	DI a.3	--	DQ a.7
11	DI a.4	--	DQ b.0
12	DI a.5	--	DQ b.1
13	DI a.6	--	--
14	DI a.7	--	--
15	DI b.0	--	--
16	DI b.1	--	--
17	DI b.2	--	--
18	DI b.3	--	--
19	DI b.4	--	--
20	DI b.5	--	--

说明

应将未使用的模拟量输入短路。

A.8 CPU 1217C

A.8.1 常规规范和特性

表格 A-87 常规

技术数据	CPU 1217C DC/DC/DC
产品编号	6ES7217-1AG40-0XB0
尺寸 W x H x D (mm)	150 x 100 x 75
装运重量	530 g
功耗	12 W
可用电流（SM 和 CM 总线）	最大 1600 mA(5 V DC)
可用电流 (24 V DC)	最大 400 mA（传感器电源）
数字量输入电流消耗 (24 V DC)	所用的每点输入 4 mA

表格 A-88 CPU 特征

技术数据		说明
用户存储器（请参见“一般技术数据 (页 1427)”，“内部 CPU 存储器掉电保持”。）	工作	150 KB
	负载	内置 4 MB，可用 SD 卡扩展，具体视卡容量而定
	保持性	10 KB
板载数字 I/O		14 点输入/10 点输出
板载模拟 I/O		2 点输入/2 点输出
过程映像大小		1024 字节输入 (I)/1024 字节输出 (Q)
位存储器 (M)		8192 个字节
临时（局部）存储器		• 16 KB 用于启动和程序循环（包括相关的 FB 和 FC） • 6 KB 用于其它各中断优先级（包括 FB 和 FC）
信号模块扩展		最多 8 个信号模块
SB、CB、BB 扩展		最多 1 个
通信模块扩展		最多 3 个通信模块

技术数据	说明
高速计数器	最多可组态 6 个使用任意内置或 SB 输入的高速计数器（请参见 CPU 1217C 数字量输入 (DI) H/W 组态表） (页 1509) • 1 MHz（Ib.2 到 Ib.5） • 100/[1]80 kHz（Ia.0 到 Ia.5） • 30/[1]20kHz（Ia.6 到 Ib.1）
脉冲输出	最多可组态 4 个使用任意内置或 SB 输出的脉冲输出（请参见 CPU 1217C 数字量输出 (DQ) H/W 组态表） (页 1509) • 1 MHz（Qa.0 到 Qa.3） • 100 kHz（Qa.4 到 Qb.1）
脉冲捕捉输入	14
延时中断	共 4 个，精度为 1 ms
循环中断	共 4 个，精度为 1 ms
沿中断	12 个上升沿和 12 个下降沿（使用可选信号板时，各为 16 个）
存储卡	SIMATIC 存储卡（选件）
实时时钟精度	+/- 60 秒/月
实时时钟保持时间	通常为 20 天，40 °C 时最少为 12 天（免维护超级电容）

[1] 将 HSC 组态为正交工作模式时，可应用较慢的速度。

表格 A-89 性能

指令类型		执行速度	
		直接寻址（I、Q 和 M）	DB 访问
布尔运算		0.08 µs/指令	
移动	移动布尔数据	0.3 µs/指令	1.17 µs/指令
	移动字	0.137 µs/指令	1.0 µs/指令
	移动实数	0.72 µs/指令	1.0 µs/指令
实数数学运算	加上实数	1.48 µs/指令	1.78 µs/指令

说明

许多变量影响测量时间。上述性能时间适用于此类别和无错程序中的最快指令。

A.8.2 CPU 1217C 支持的定时器、计数器和代码块

表格 A-90 CPU 1217C 支持的块、定时器和计数器

元素		描述
块	类型	OB、FB、FC、DB
	大小	OB、FB、FC：64 KB DB：最大可达工作存储器的大小
	数量	最多可达 1024 个块 (OB + FB + FC + DB)
	FB、FC 和 DB 的地址范围	FB 和 FC：1 到 65535（例如 FB 1 到 FB 65535） DB：1 到 59999
	嵌套深度	16（从程序循环 OB 或启动 OB 开始） 6（从任意中断事件 OB 开始）[1]
	监视	可以同时监视 2 个代码块的状态
OB	程序循环	多个
	启动	多个
	延时中断	4（每个事件 1 个）
	循环中断	4（每个事件 1 个）
	硬件中断	50（每个事件 1 个）
	时间错误中断	1
	诊断错误中断	1
	拔出或插入模块	1
	机架或站故障	1
	日时钟	多个
	状态	1
	更新	1
	配置文件	1
	MC 插补器	1
	MC 伺服电机	1
	MC-PreServo	1
	MC-PostServo	1

元素		描述
定时器	类型	IEC
	数量	仅受存储器大小限制
	存储	DB 结构，每个定时器 16 个字节
计数器	类型	IEC
	数量	仅受存储器大小限制
	存储	DB 结构，大小取决于计数类型 • SInt 和 USInt：3 个字节 • Int 和 UInt：6 个字节 • DInt 和 UDInt：12 个字节

[1] 安全程序使用二级嵌套。因此，用户程序在安全程序中的嵌套深度为四。

表格 A-91 通信

技术数据	描述
端口数	2
类型	以太网
HMI 设备	4
编程设备 (PG)	1
连接	• 8 个连接用于开放式用户通信（主动或被动）：TSEND_C、TRCV_C、TCON、TDISCON、TSEND 和 TRC • 8 个 CPU 到 CPU 连接（客户端或服务器）用于 GET/PUT 数据 • 6 个连接用于动态分配到 GET/PUT 或开放式用户通信
数据传输率	10/100 Mb/s
隔离（外部信号与逻辑侧）	变压器隔离，1500 V AC（型式测试）[1]
电缆类型	CAT5e 屏蔽电缆
接口	
PROFINET 接口的数量	1
PROFIBUS 接口的数量	0
接口	
接口硬件	
端口数量	2
集成交换机	√

技术数据	描述
RJ-45（以太网）	√；X1
协议	
PROFINET IO 控制器	√
PROFINET IO 设备	√
SIMATIC 通信	√
开放式 IE 通信	√
Web 服务器	√
介质冗余	√
PROFINET IO 控制器	
服务	
PG/OP 通信	√
S7 路由	√
等时同步模式	-
开放式 IE 通信	√
IRT	-
MRP	√；作为 MRP 客户端
PROFIenergy	√S7-1200 CPU 仅支持 PROFIenergy 实体（具有智能设备功能）。
优先启动	√（最多 16 个 PROFINET 设备）
可连接 I/O 设备的最大数量	16
可进行 RT 连接 I/O 设备的最大数量	16
线形结构中的最大数量	16
可同时激活/取消激活 IO 设备的最大数量	8
更新时间	最短更新时间还取决于 PROFINET IO 上设置的通信组件、IO 设备的数量以及所组态用户数据的数量。
RT 功能	
1 ms 的发送时钟	1 ms 到 512 ms
PROFINET IO 设备	
服务	
PG/OP 通信	√
S7 路由	√
等时同步模式	-

技术数据		描述
开放式 IE 通信		√
IRT，支持		-
MRP，支持		√
PROFIenergy		√
共享设备		√
共享设备的最大 IO 控制器数		2
SIMATIC 通信		
S7 通信，作为服务器		√
S7 通信，作为客户端		√
每个作业中用户数据的最大值		请参见在线帮助（S7 通信、用户数据大小）
开放式 IE 通信		
TCP/IP：		√
	最大数据长度	8 KB
	每个端口支持多个被动连接	√
ISO-on-TCP (RFC1006)：		√
	最大数据长度	8 KB
UDP：		√
	最大数据长度	1472 个字节
DHCP		-
SNMP		√
DCP		√
LLDP		√

[1] 以太网端口隔离专用于在危险电压引起短期网络故障时对危险情况进行限制。它不符合常规 AC 线电压隔离的安全要求。

表格 A-92 电源

技术数据	CPU 1217C DC/DC/DC
电压范围	20.4 V DC 到 28.8 V DC
电源频率	--

技术数据		CPU 1217C DC/DC/DC
输入电流（最大负载时）	仅 CPU	24 V DC 时 600 mA
	具有所有扩展附件的 CPU	24 V DC 时 1600 mA
浪涌电流（最大）		28.8 V DC 时 12 A
$I^2 t$		0.5 A^2 s
隔离（输入电源与逻辑侧）		未隔离
保持时间（自掉电起）		24 V DC 时 10 ms
内部保险丝，用户不可更换		3 A，250 V，慢速熔断

表格 A-93　传感器电源

技术数据	CPU 1217C DC/DC/DC
电压范围	L+ - 4 V DC（最小值）
额定输出电流（最大）	400 mA（短路保护）
最大波纹噪声 (<10 MHz)	与输入线路相同
隔离（CPU 逻辑侧与传感器电源）	未隔离

A.8.3　数字量输入和输出

表格 A-94　数字量输入

技术数据	CPU 1217C DC/DC/DC
输入点数	14：总计： 10：漏型/源型（IEC 1 类漏型） 4：差分 (RS422/RS485)
类型：漏型/源型（IEC 1 类漏型）	Ia.0 到 Ia.7，Ib.0 到 Ib.1
额定电压	4 mA 时 24 V DC，额定值
允许的连续电压	30 V DC，最大值
浪涌电压	35 V DC，持续 0.5 s
逻辑 1 信号（最小）	2.5 mA 时 15 V DC

技术数据	CPU 1217C DC/DC/DC
逻辑 0 信号（最大）	1 mA 时 5 V DC
隔离（现场侧与逻辑侧）	707 V DC（型式测试）
隔离组	1
滤波时间	us 设置：0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0 ms 设置：0.05、0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0
HSC 时钟输入频率（最大） （逻辑 1 电平 = 15 到 26 V DC）	100/80 kHz（Ia.0 到 Ia.5） 30/20 kHz（Ia.6 到 Ib.1）
类型：差分输入 (RS422/RS485)	Ib.2 到 Ib.5（.2+ .2- 到 .5+ .5-）
共模电压范围	-7 V 到 +12 V，1 秒，3 VRMS 连续（RS422/RS485 特性）
内置终端电阻和偏置	Ib'-' 上 390 Ω 对 2M，Ib'-' 上 390 Ω 对 +5 V（T/B 开路时偏置为关闭状态） Ib'+' 和 Ib'-' 之间为 220 Ω
接收器输入阻抗	100 Ω，包括偏置和终端
差分接收器 阈值/灵敏度	最低 +/- 0.2 V，典型滞后 60 mV（RS422/RS485 特性）
隔离（现场侧与逻辑侧）	707 V DC（型式测试）
隔离组	1
滤波时间	us 设置：0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0 ms 设置：0.05、0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0
HSC 时钟输入频率（最大）	单相：1 MHz（Ib.2 到 Ib.5） 正交相位：1 MHz（Ib.2 到 Ib.5）
差分输入通道间的时间偏差	最大 40 ns
常规规范（所有数字量输入）	
同时接通的输入数	5，漏型/源型输入（无相邻点）；4，差分输入，60 ℃（水平）或 50 ℃（垂直）时 14，55 ℃（水平）或 45 ℃（垂直）时
电缆长度（米）	500 m（屏蔽）；300 m（非屏蔽） 50 m（屏蔽，HSC 输入）（漏型/源型） 50 m（屏蔽，双绞线）（针对所有差分输入）

表格 A-95　CPU 1217C 数字量输入 (DI) H/W 组态表

输入	类型和速率
DIa.0	类型：24 V，源型 - 漏型 1 类输入 高速计数器输入速率：最大 100 kHz
DIa.1	类型：24 V，源型 - 漏型 1 类输入 高速计数器输入速率：最大 100 kHz
DIa.2	类型：24 V，源型 - 漏型 1 类输入 高速计数器输入速率：最大 100 kHz
DIa.3	类型：24 V，源型 - 漏型 1 类输入 高速计数器输入速率：最大 100 kHz
DIa.4	类型：24 V，源型 - 漏型 1 类输入 高速计数器输入速率：最大 100 kHz
DIa.5	类型：24 V，源型 - 漏型 1 类输入 高速计数器输入速率：最大 100 kHz
DIa.6	类型：24 V，源型 - 漏型 1 类输入 高速计数器输入速率：最大 30 kHz
DIa.7	类型：24 V，源型 - 漏型 1 类输入 高速计数器输入速率：最大 30 kHz
DIb.0	类型：24 V，源型 - 漏型 1 类输入 高速计数器输入速率：最大 30 kHz
DIb.1	类型：24 V，源型 - 漏型 1 类输入 高速计数器输入速率：最大 30 kHz
DIb.2+　.2-	类型：RS422/RS485 差分输入 高速计数器输入速率：最大 1 MHz
DIb.3+　.3-	类型：RS422/RS485 差分输入 高速计数器输入速率：最大 1 MHz
DIb.4+　.4-	类型：RS422/RS485 差分输入 高速计数器输入速率：最大 1 MHz
DIb.5+　.5-	类型：RS422/RS485 差分输入 高速计数器输入速率：最大 1 MHz

表格 A-96　数字量输出

技术数据	CPU 1217C DC/DC/DC
输出点数	共 10 个 6：固态 - MOSFET（源型） 4：差分 (RS422/RS485)
类型：固态 - MOSFET（源型输出）	Qa.4 到 Qb.1
电压范围	20.4 到 28.8 V DC
最大电流时的逻辑 1 信号	20 V DC 最小
具有 10 KΩ 负载时的逻辑 0 信号	0.1 V DC 最大
电流（最大）	0.5 A
灯负载	5 W
通态电阻	最大 0.6 Ω
每点的漏电流	最大 10 μA
浪涌电流	8 A，最长持续 100 ms
过载保护	×
隔离（现场侧与逻辑侧）	707 V DC（型式测试）
隔离组	1
电感钳位电压	L+ - 48 V DC，1 W 损耗
开关延迟（Qa.4 到 Qb.1）	断开到接通最长为 1.0 μs 接通到断开最长为 3.0 μs
继电器最大开关频率	--
脉冲串输出频率	最大 100 kHz（Qa.4 到 Qb.1）[1]，最小 2 Hz
类型：差分输出 (RS422/RS485)	Qa.0 到 Qa.3（.0+ 0- 到 .3+ .3-）
共模电压范围	-7 V 到 +12 V，1 秒，3 VRMS 连续（RS422/RS485 特性）
发送器差动输出电压	RL = 100 Ω 时，最小 2 V；RL = 54 Ω 时，最小 1.5 V（RS422/RS485 特性）
内置终端电阻	Qa'+' 和 Qa'-' 之间为 100 Ω
驱动器输出阻抗	100 Ω，包括终端
隔离（现场侧与逻辑侧）	707 V DC（型式测试）

技术数据	CPU 1217C DC/DC/DC
隔离组	1
开关延迟（DQa.0 到 DQa.3）	最大 100 ns
差分输出通道间的时间偏差	最大 40 ns
脉冲串输出频率	1 MHz（Qa.0 到 Qa.3），最小 2 Hz
常规规范 **（所有数字量输出）**	
RUN 到 STOP 时的行为	上一个值或替换值（默认值为 0）
同时接通的输出数	3，固态 - MOSFET（源型）输出（无相邻点），4（差分输出），60 ℃（水平）或 50 ℃（垂直）时 10，55 ℃（水平）或 45 ℃（垂直）时
电缆长度（米）	500 m（屏蔽）；150 m（非屏蔽）

[1] 根据所使用的脉冲接收器和电缆的情况，附加的负载电阻（至少为额定电流的 10%）可能改进脉冲信号质量和抗扰度。

表格 A-97　CPU 1217C 数字量输出 (DQ) H/W 组态表

输出	类型和速率
DQa.0+　.0-	类型：RS422/RS485 差分输出 脉冲串输出频率：最大 1 MHz，最小 2 Hz
DQa.1+　.1-	类型：RS422/RS485 差分输出 脉冲串输出频率：最大 1 MHz，最小 2 Hz
DQa.2+　.2-	类型：RS422/RS485 差分输出 脉冲串输出频率：最大 1 MHz，最小 2 Hz
DQa.3+　.3-	类型：RS422/RS485 差分输出 脉冲串输出频率：最大 1 MHz，最小 2 Hz
DQa.4	类型：24 V 源型输出 脉冲串输出频率：最大 100 kHz，最小 2 Hz
DQa.5	类型：24 V 源型输出 脉冲串输出频率：最大 100 kHz，最小 2 Hz
DQa.6	类型：24 V 源型输出 脉冲串输出频率：最大 100 kHz，最小 2 Hz
DQa.7	类型：24 V 源型输出 脉冲串输出频率：最大 100 kHz，最小 2 Hz

输出	类型和速率
DQb.0	类型：24 V 源型输出 脉冲串输出频率：最大 100 kHz，最小 2 Hz
DQb.1	类型：24 V 源型输出 脉冲串输出频率：最大 100 kHz，最小 2 Hz

A.8.4 模拟量输入和输出

A.8.4.1 模拟量输入规范

表格 A-98 模拟量输入

技术数据	说明
输入点数	2
类型	电压（单侧）
满量程范围	0 到 10 V
满量程范围（数据字）	0 到 27648
过冲范围	10.001 到 11.759 V
过冲范围（数据字）	27649 到 32511
上溢范围	11.760 到 11.852 V
上溢范围（数据字）	32512 到 32767
分辨率	10 位
最大耐压	35 V DC
平滑化	无、弱、中或强 请参见 CPU 模拟量输入的阶跃响应 (ms) (页 1515)表格。
噪声抑制	10、50 或 60 Hz
阻抗	≥100 KΩ
隔离（现场侧与逻辑侧）	无
精度（25 °C/-20 到 60 °C）	满量程的 3.0%/3.5%
电缆长度（米）	100 m，屏蔽双绞线

A.8.4.2 CPU 内置模拟量输入的阶跃响应

表格 A-99 阶跃响应 (ms)，0 V 到 10 V（在 95% 处测得）

平滑化选项（采样平均）	抑制频率（积分时间）		
	60 Hz	**50 Hz**	**10 Hz**
无（1 个周期）： 不求平均值	50 ms	50 ms	100 ms
弱（4 个周期）： 4 次采样	60 ms	70 ms	200 ms
中（16 个周期）： 16 次采样	200 ms	240 ms	1150 ms
强（32 个周期）： 32 次采样	400 ms	480 ms	2300 ms
采样时间	**4.17 ms**	**5 ms**	**25 ms**

A.8.4.3 CPU 内置模拟端口的采样时间

表格 A-100 CPU 内置模拟量输入的采样时间

抑制频率（积分时间选项）	采样时间
60 Hz (16.6 ms)	4.17 ms
50 Hz (20 ms)	5 ms
10 Hz (100 ms)	25 ms

A.8.4.4 模拟量输入的电压测量范围 (CPU)

表格 A-101 模拟量输入的电压表示法 (CPU)

系统		电压测量范围	
十进制	十六进制	**0 到 10 V**	
32767	7FFF	11.852 V	上溢
32512	7F00		
32511	7EFF	11.759 V	过冲范围
27649	6C01		

系统		电压测量范围	
十进制	十六进制	0 到 10 V	
27648	6C00	10 V	额定范围
20736	5100	7.5 V	
34	22	12 mV	
0	0	0 V	
负值		不支持负值	

A.8.4.5 模拟量输出规格

表格 A-102 模拟量输出

技术数据	说明
输出点数	2
类型	电流
满量程范围	0 到 20 mA
满量程范围（数据字）	0 到 27648
过冲范围	20.01 到 23.52 mA
过冲范围（数据字）	27649 到 32511
上溢范围	参见脚注 [1]
上溢范围数据字	32512 到 32767
分辨率	10 位
输出驱动阻抗	最大 500 Ω
隔离（现场侧与逻辑侧）	无
精度（25 °C/-20 到 60 °C）	满量程的 3.0%/3.5%
稳定时间	2 ms
电缆长度（米）	100 m，屏蔽双绞线

[1] 在上溢情况下，模拟量输出的行为将符合设备组态属性设置。 在“对 CPU STOP 的响应”参数中，选择其中一项：“使用替换值”或“保持上一个值”。

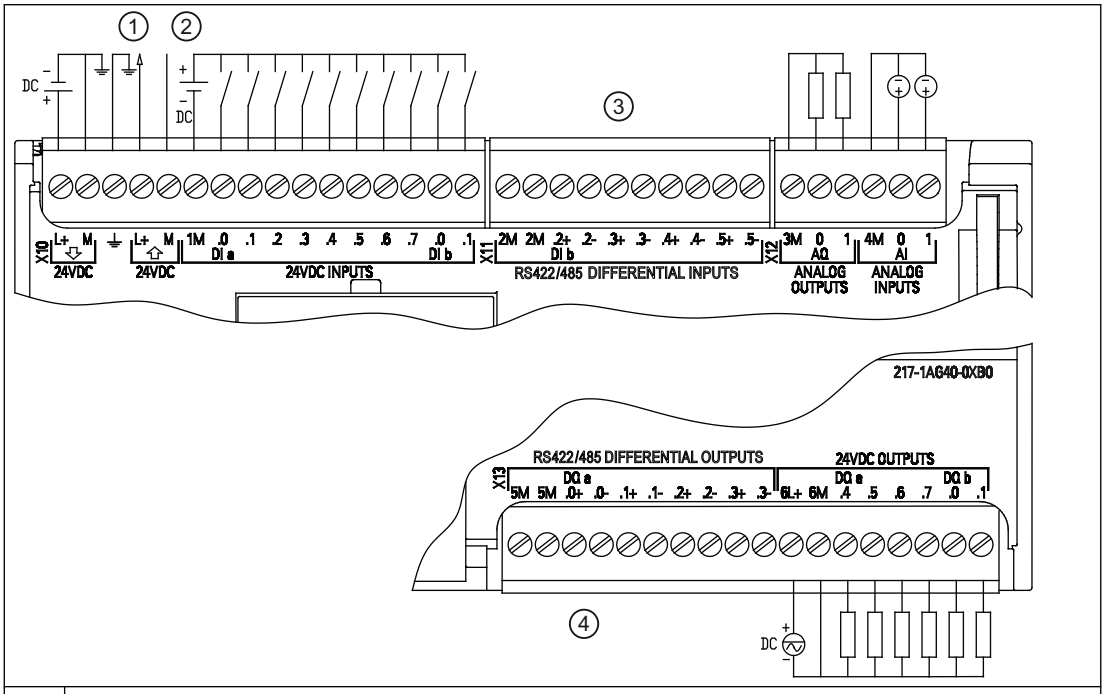
表格 A-103 模拟量输出的电流表示法（CPU 1215C 和 CPU 1217C）

系统		当前输出范围	
十进制	十六进制	0 mA 到 20 mA	
32767	7FFF	请参见注 1	上溢
32512	7F00	请参见注 1	
32511	7EFF	23.52 mA	过冲范围
27649	6C01		
27648	6C00	20 mA	额定范围
20736	5100	15 mA	
34	22	0.0247 mA	
0	0	0 mA	
负值		不支持负值	

[1] 在上溢情况下，模拟量输出的行为将符合设备组态属性设置。 在“对 CPU STOP 的响应”参数中，选择其中一项：“使用替换值”或“保持上一个值”。

A.8.5 CPU 1217C 接线图

表格 A-104 CPU 1217C DC/DC/DC (6ES7217-1AG40-0XB0)



①	24 V DC 传感器电源输出 要获得更好的抗噪声效果，即使未使用传感器电源，也可将“M”连接到机壳接地。
②	对于漏型输入，将“-”连接到“M”（如图所示）。对于源型输入，将“+”连接到“M”。
③	请参见 CPU 1217C 差分输入 (DI) 的详细信息和应用示例 (页 1520)。
④	请参见 CPU 1217C 差分输出 (DQ) 的详细信息和应用示例 (页 1521)。
注 1：X12 连接器必须镀金。有关产品编号，请参见附录 C，“备件” (页 1657)。	
注 2：有关 CPU 以太网端口的信息，请参见“设备配置 (页 147)”。	

表格 A-105 CPU 1217C DC/DC/DC (6ES7217-1AG40-0XB0) 的连接器引脚位置

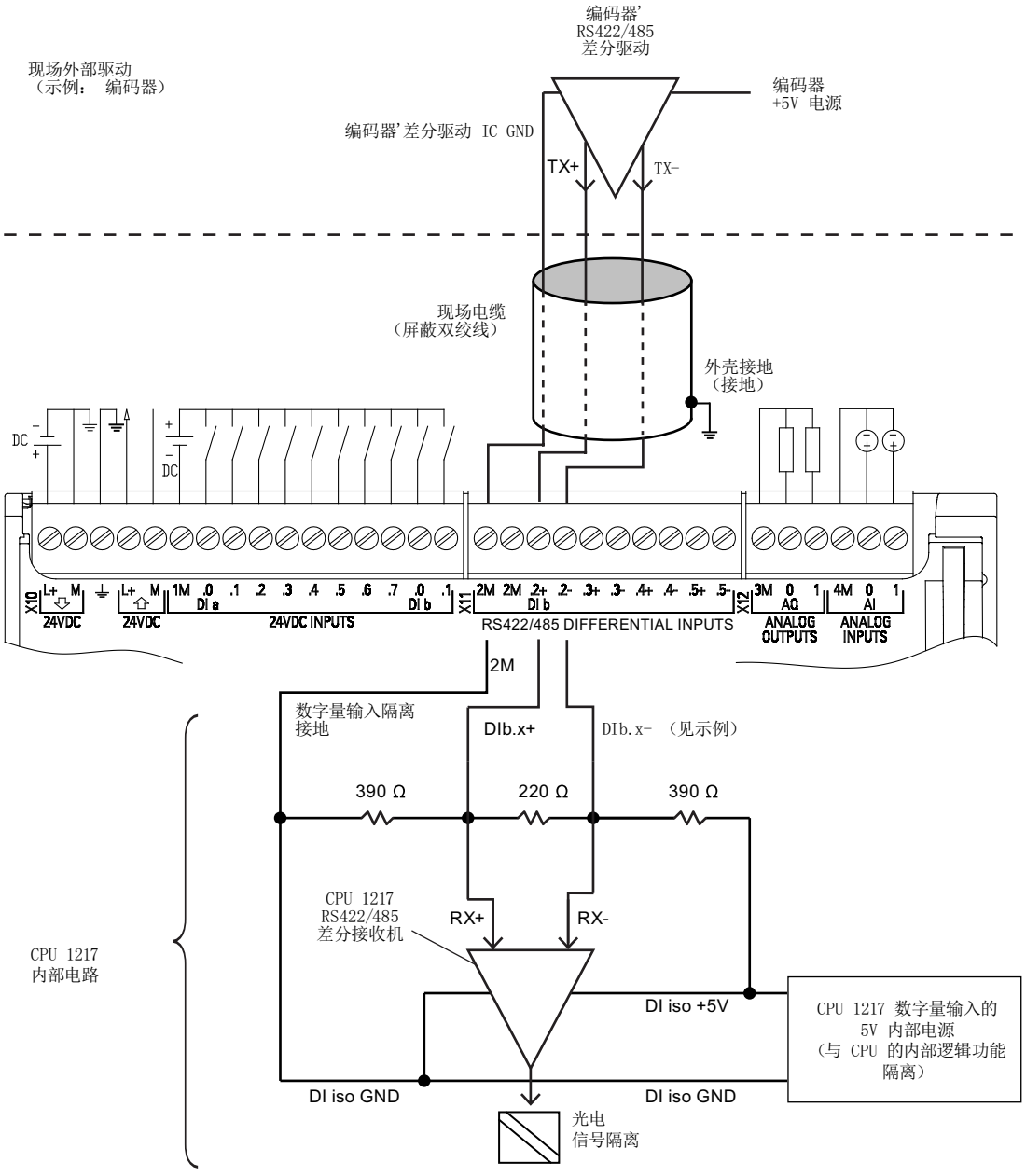
引脚	X10	X11	X12（镀金）	X13
1	L+ / 24 V DC	2M	3M	5M
2	M / 24 V DC	2M	AQ 0	5M
3	功能性接地	DI b.2+	AQ 1	DQ a.0+
4	L+ / 24 V DC 传感器输出	DI b.2-	4M	DQ a.0-
5	M / 24 V DC 传感器输出	DI b.3+	AI 0	DQ a.1+
6	1M	DI b.3-	AI 1	DQ a.1-

引脚	X10	X11	X12（镀金）	X13
7	DI a.0	DI b.4+	--	DQ a.2+
8	DI a.1	DI b.4-	--	DQ a.2-
9	DI a.2	DI b.5+	--	DQ a.3+
10	DI a.3	DI b.5-	--	DQ a.3-
11	DI a.4	--	--	6L+
12	DI a.5	--	--	6M
13	DI a.6	--	--	DQ a.4
14	DI a.7	--	--	DQ a.5
15	DI b.0	--	--	DQ a.6
16	DI b.1	--	--	DQ a.7
17	--	--	--	DQ b.0
18	--	--	--	DQ b.1

说明

应将未使用的模拟量输入短路。

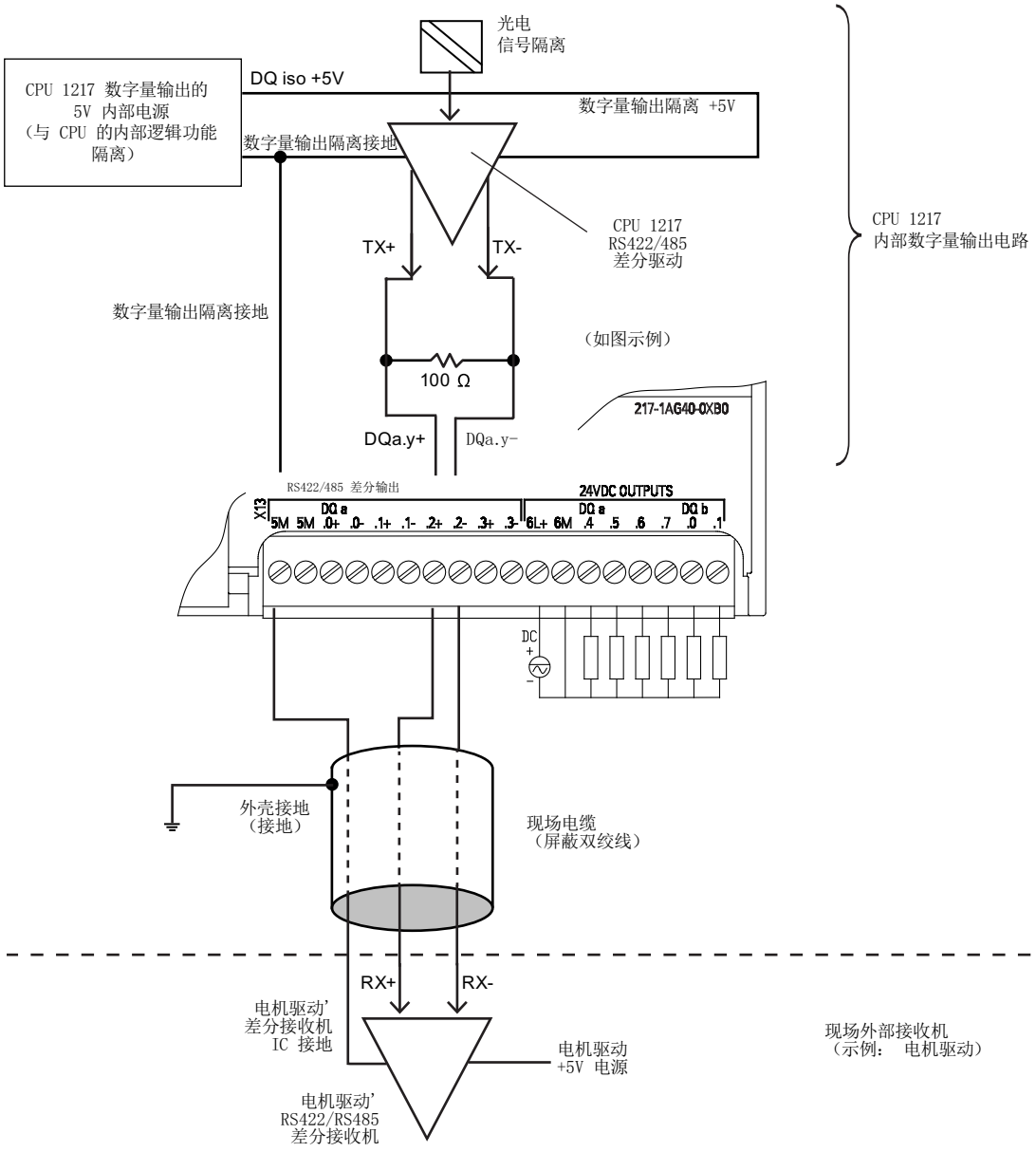
A.8.6 CPU 1217C 差分输入 (DI) 的详细信息和应用示例



注意

- 螺钉接线板为开路时，每个差分 DI 会偏置为“关闭”状态。
- 内置 DI 终端电阻和偏置 = 100 Ω 等效阻抗。
- 内置数字量输入端接电阻和偏置电阻限制了连续共模电压范围。 详细信息，请参见“电气技术规格”。

A.8.7 CPU 1217C 差分输出 (DQ) 的详细信息和应用示例



注意事项

- 内置数字量输出端接电阻限制了连续共模电压范围。 详细信息，请参见“电气技术规格”。

A.9 数字信号模块 (SM)

A.9.1 SM 1221 数字量输入规范

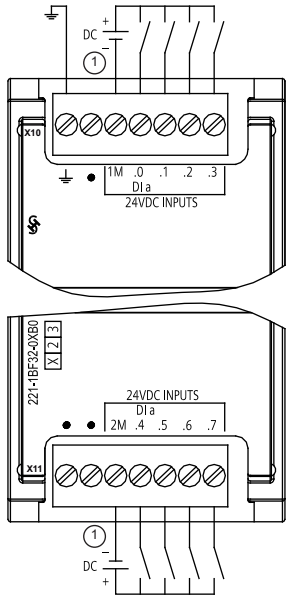
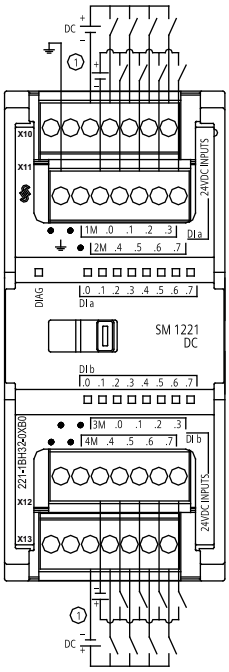
表格 A-106 常规规范

型号	SM 1221 DI 8 x 24 V DC	SM 1221 DI 16 x 24 V DC
产品编号	6ES7221-1BF32-0XB0	6ES7221-1BH32-0XB0
尺寸 W x H x D (mm)	45 x 100 x 75	
重量	170 g	210 g
功耗	1.5 W	2.5 W
电流消耗（SM 总线）	105 mA	130 mA
电流消耗 (24 V DC)	所用的每点输入 4 mA	

表格 A-107 数字量输入

型号	SM 1221 DI 8 x 24 V DC	SM 1221 DI 16 x 24 V DC
输入点数	8	16
类型	漏型/源型（IEC 1 类漏型）	
额定电压	4 mA 时 24 V DC，额定值	
允许的连续电压	30 V DC，最大值	
浪涌电压	35 V DC，持续 0.5 s	
逻辑 1 信号（最小）	2.5 mA 时 15 V DC	
逻辑 0 信号（最大）	1 mA 时 5 V DC	
隔离（现场侧与逻辑侧）	707 V DC（型式测试）	
隔离组	2	4
滤波时间	0.2、0.4、0.8、1.6、3.2、6.4 和 12.8 ms（可选择，4 个为一组）	0.2、0.4、0.8、1.6、3.2、6.4 和 12.8 ms（可选择，4 个为一组）
同时接通的输入数	8	16
电缆长度（米）	500 m（屏蔽）；300 m（非屏蔽）	

表格 A-108 数字量输入 SM 的接线图

SM 1221 DI 8 x 24 V DC (6ES7221-1BF32-0XB0)	SM 1221 DI 16 x 24 V DC (6ES7221-1BH32-0XB0)
	

① 对于漏型输入，将“-”连接到“M”（如图所示）。对于源型输入，将“+”连接到“M”。

表格 A-109 SM 1221 DI 8 x 24 V DC (6ES7221-1BF32-0XB0) 的连接器引脚位置

引脚	X10	X11
1	功能性接地	无连接
2	无连接	无连接
3	1M	2M
4	DI a.0	DI a.4
5	DI a.1	DI a.5
6	DI a.2	DI a.6
7	DI a.3	DI a.7

表格 A-110 SM 1221 DI 16 x 24 V DC (6ES7221-1BH32-0XB0) 的连接器引脚位置

引脚	X10	X11	X12	X13
1	无连接	功能性接地	无连接	无连接
2	无连接	无连接	无连接	无连接
3	1M	2M	3 M	4 M
4	DI a.0	DI a.4	DI b.0	DI b.4
5	DI a.1	DI a.5	DI b.1	DI b.5
6	DI a.2	DI a.6	DI b.2	DI b.6
7	DI a.3	DI a.7	DI b.3	DI b.7

A.9.2 SM 1222 8 点数字量输出规范

表格 A-111 常规规范

型号	SM 1222 DQ 8 x 继电器	SM 1222 DQ 8 继电器切换	SM 1222 DQ 8 x 24 V DC
订货号	6ES7222-1HF32-0XB0	6ES7222-1XF32-0XB0	6ES7222-1BF32-0XB0
尺寸 W x H x D (mm)	45 x 100 x 75	70 x 100 x 75	45 x 100 x 75
重量	190 g	310 g	180 g
功耗	4.5 W	5 W	1.5 W
电流消耗（SM 总线）	120 mA	140 mA	120 mA
电流消耗 (24 V DC)	所用的每个继电器线圈 11 mA	所用的每个继电器线圈 16.7 mA	50 mA

表格 A-112 数字量输出

型号	SM 1222 DQ 8 x 继电器	SM 1222 DQ 8 继电器切换	SM 1222 DQ 8 x 24 V DC
输出点数	8	8	8
类型	继电器，机械式	继电器切换触点	固态 - MOSFET（源型）
电压范围	5 到 30 V DC 或 5 到 250 V AC	5 到 30 V DC 或 5 到 250 V AC	20.4 到 28.8 V DC

型号	SM 1222 DQ 8 x 继电器	SM 1222 DQ 8 继电器切换	SM 1222 DQ 8 x 24 V DC
最大电流时的逻辑 1 信号	--	--	20 V DC 最小
具有 10 KΩ 负载时的逻辑 0 信号	--	--	0.1 V DC 最大
电流（最大）	2.0 A	2.0 A	0.5 A
灯负载	30 W DC/200 W AC	30 W DC/200 W AC	5 W
通态触点电阻	新设备最大为 0.2 Ω	新设备最大为 0.2 Ω	最大 0.6 Ω
每点的漏电流	--	--	最大 10 μA
浪涌电流	触点闭合时为 7 A	触点闭合时为 7 A	8 A，最长持续 100 ms
过载保护	×	×	
隔离（现场侧与逻辑侧）	1500 V AC（线圈与触点） 无（线圈与逻辑侧）	1500 V AC（线圈与触点）	707 V DC（型式测试）
隔离组	2	8	1
每个公共端的电流（最大）	10 A	2 A	4 A
电感钳位电压	--	--	L+ - 48 V，1 W 损耗
开关延迟	最长 10 ms	最长 10 ms	断开到接通最长为 50 μs 接通到断开最长为 200 μs
继电器最大开关频率	1 Hz	1 Hz	--
机械寿命（无负载）	10,000,000 个断开/闭合周期	10,000,000 个断开/闭合周期	--
额定负载下的触点寿命（常开触点）	100,000 个断开/闭合周期	100,000 个断开/闭合周期	--
RUN 到 STOP 时的行为	上一个值或替换值（默认值为 0）	上一个值或替换值（默认值为 0）	上一个值或替换值（默认值为 0）
同时接通的输出数	8	• 4（无相邻点），60 ℃（水平）或 50 ℃（垂直）时 • 8，55 ℃（水平）或 45 ℃（垂直）时	8
电缆长度（米）	500 m（屏蔽）； 150 m（非屏蔽）	500 m（屏蔽）； 150 m（非屏蔽）	500 m（屏蔽）； 150 m（非屏蔽）

A.9.3 SM 1222 16 点数字量输出规范

表格 A-113 常规规范

型号	SM 1222 DQ 16 x 继电器	SM 1222 DQ 16 x 24 V DC	SM 1222 DQ 16 x 24 V DC 漏型
订货号	6ES7222-1HH32-0XB0	6ES7222-1BH32-0XB0	6ES7222-1BH32-1XB0
尺寸 W x H x D (mm)	45 x 100 x 75	45 x 100 x 75	45 x 100 x 75
重量	260 g	220 g	220 g
功耗	8.5 W	2.5 W	2.5 W
电流消耗（SM 总线）	135 mA	140 mA	140 mA
电流消耗 (24 V DC)	所用的每个继电器线圈 11 mA	100 mA	40 mA

表格 A-114 数字量输出

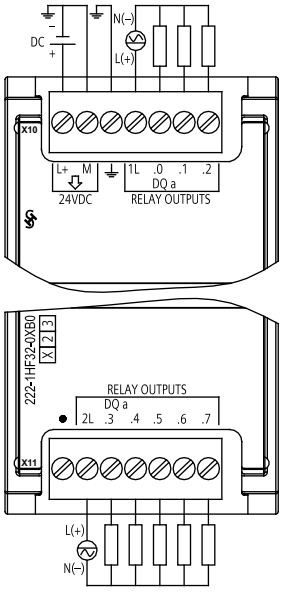
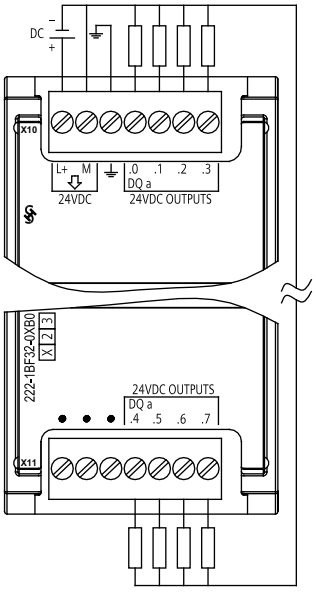
型号	SM 1222 DQ 16 x 继电器	SM 1222 DQ 16 x 24 V DC	SM 1222 DQ 16 x 24 V DC 漏型
输出点数	16	16	16
类型	继电器，机械式	固态 - MOSFET（源型）	固态 - MOSFET（漏型）
电压范围	5 到 30 V DC 或 5 到 250 V AC	20.4 到 28.8 V DC	20.4 到 28.8 V DC
最大电流时的逻辑 1 信号	-	20 V DC 最小	0.5 V DC
具有 10 KΩ 负载时的逻辑 0 信号	-	0.1 V DC 最大	24 V（典型值）- 0.75 V DC
电流（最大）	2.0 A	0.5 A	0.5 A
灯负载	30 W DC/200 W AC	5 W	5 W
通态触点电阻	新设备最大为 0.2 Ω	最大 0.6 Ω	最大 0.5 Ω
每点的漏电流	--	最大 10 μA	最大 75 μA
浪涌电流	触点闭合时为 7 A	8 A，最长持续 100 ms	8 A，最长持续 100 ms
过载保护	×		是，限流保护范围 1 A 到 3.5 A

型号	SM 1222 DQ 16 x 继电器	SM 1222 DQ 16 x 24 V DC	SM 1222 DQ 16 x 24 V DC 漏型
隔离（现场侧与逻辑侧）	1500 V AC（线圈与触点） 无（线圈与逻辑侧）	707 V DC（型式测试）	707 V DC（型式测试）
隔离组	4	1	1
每个公共端的电流（最大）	10 A	8 A	限流保护
电感钳位电压	-	L+ - 48 V，1 W 损耗	45 V
开关延迟	最长 10 ms	断开到接通最长为 50 µs 接通到断开最长为 200 µs	断开到接通最长为 20 µs 接通到断开最长为 350 µs
继电器最大开关频率	1 Hz	-	-
机械寿命（无负载）	10,000,000 个断开/闭合周期	-	-
额定负载下的触点寿命（常开触点）	100,000 个断开/闭合周期	-	-
RUN 到 STOP 时的行为	上一个值或替换值（默认值为 0）	上一个值或替换值（默认值为 0）	上一个值或替换值（默认值为 0）
同时接通的输出数	• 8（无相邻点），60 ℃（水平）或 50 ℃（垂直）时 • 16，55 ℃（水平）或 45 ℃（垂直）时	16	16
电缆长度（米）	500 m（屏蔽）；150 m（非屏蔽）		

参见

新功能 (页 37)

表格 A-115 8 点数字量输出 SM 的接线图

SM 1222 DQ 8 x 继电器 (6ES7222-1HF32-0XB0)	SM 1222 DQ 8 x 24 V DC (6ES7222-1BF32-0XB0)
	

表格 A-116 SM 1222 DQ 8 x 继电器 (6ES7222-1HF32-0XB0) 的连接器引脚位置

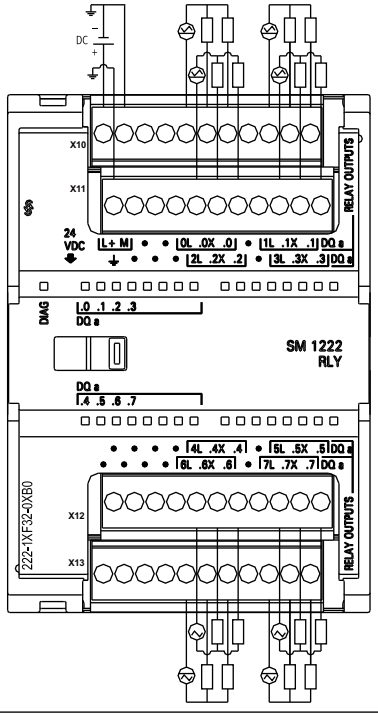
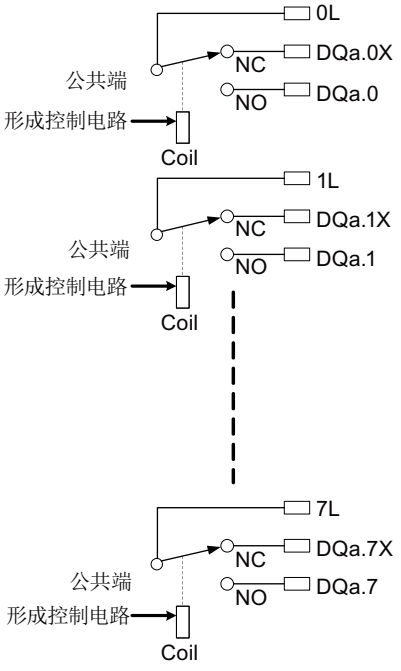
引脚	X10	X11
1	L+ / 24 V DC	无连接
2	M / 24 V DC	2L
3	功能性接地	DQ a.3
4	1L	DQ a.4
5	DQ a.0	DQ a.5
6	DQ a.1	DQ a.6
7	DQ a.2	DQ a.7

表格 A-117 SM 1222 DQ 8 x 24 V DC (6ES7222-1BF32-0XB0) 的连接器引脚位置

引脚	X10	X11
1	L+ / 24 V DC	无连接
2	M / 24 V DC	无连接
3	功能性接地	无连接
4	DQ a.0	DQ a.4

引脚	X10	X11
5	DQ a.1	DQ a.5
6	DQ a.2	DQ a.6
7	DQ a.2	DQ a.7

表格 A-118 8 点数字量输出继电器（切换）SM 的接线图

SM 1222 DQ 8 x 继电器切换 (6ES7222-1XF32-0XB0)	
	

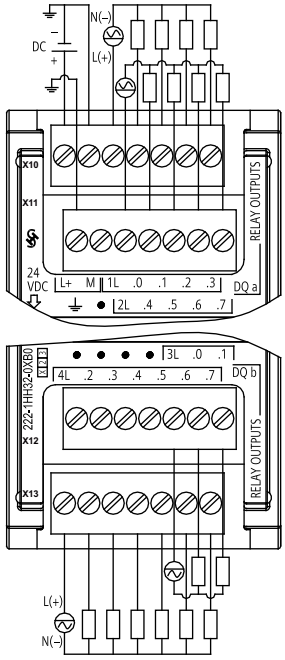
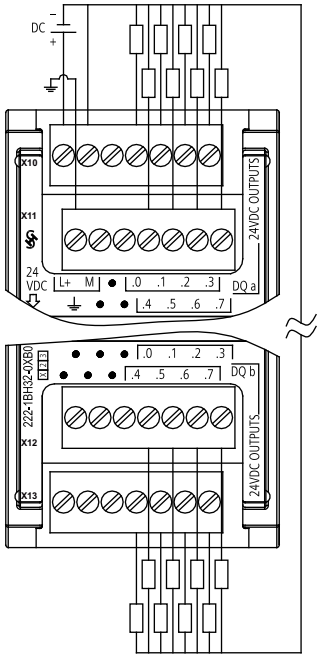
切换继电器输出使用公共端子控制两个电路：一个常闭触点和一个常开触点。例如输出“0”，当输出点断开时，公共端子 (0L) 与常闭触点 (.0X) 相连并与常开触点 (.0) 断开。当输出点接通时，公共端子 (0L) 与常闭触点 (.0X) 断开并与常开触点 (.0) 相连。

表格 A-119 SM 1222 DQ 8 x 继电器切换 (6ES7222-1XF32-0XB0) 的连接器引脚位置

引脚	X10	X11	X12	X13
1	L+ / 24 V DC	功能性接地	无连接	无连接
2	M / 24 V DC	无连接	无连接	无连接
3	无连接	无连接	无连接	无连接
4	无连接	无连接	无连接	无连接
5	0L	2L	4L	6L

引脚	X10	X11	X12	X13
6	DQ a.0X	DQ a.2X	DQ a.4X	DQ a.6X
7	DQ a.0	DQ a.2	DQ a.4	DQ a.6
8	无连接	无连接	无连接	无连接
9	1L	3L	5L	7L
10	DQ a.1X	DQ a.3X	DQ a.5X	DQ a.7X
11	DQ a.1	DQ a.3	DQ a.5	DQ a.7

表格 A-120 16 点数字量输出 SM 的接线图

SM 1222 DQ 16 x 继电器 (6ES7222-1HH32-0XB0)	SM 1222 DQ 16 x 24 V DC (6ES7222-1BH32-0XB0)
	

表格 A-121 SM 1222 DQ 16 x 继电器 (6ES7222-1HH32-0XB0) 的连接器引脚位置

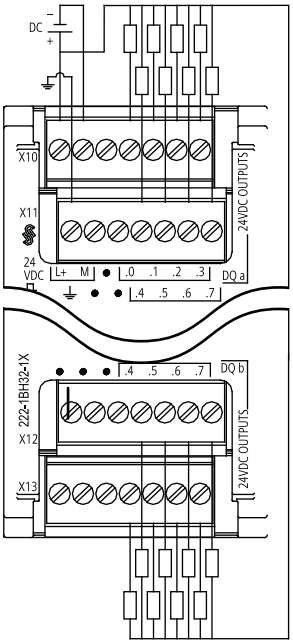
引脚	X10	X11	X12	X13
1	L+ / 24 V DC	功能性接地	无连接	4L
2	M / 24 V DC	无连接	无连接	DQ b.2
3	1L	2L	无连接	DQ b.3
4	DQ a.0	DQ a.4	无连接	DQ b.4

引脚	X10	X11	X12	X13
5	DQ a.1	DQ a.5	3L	DQ b.5
6	DQ a.2	DQ a.6	DQ b.0	DQ b.6
7	DQ a.3	DQ a.7	DQ b.1	DQ b.7

表格 A-122 SM 1222 DQ 16 x 24 V DC (6ES7222-1BH32-0XB0) 的连接器引脚位置

引脚	X10	X11	X12	X13
1	L+ / 24 V DC	功能性接地	无连接	无连接
2	M / 24 V DC	无连接	无连接	无连接
3	无连接	无连接	无连接	无连接
4	DQ a.0	DQ a.4	DQ b.0	DQ b.4
5	DQ a.1	DQ a.5	DQ b.1	DQ b.5
6	DQ a.2	DQ a.6	DQ b.2	DQ b.6
7	DQ a.3	DQ a.7	DQ b.3	DQ b.7

表格 A-123 16 点数字量输出 24 V DC 漏型 SM 的接线图

SM 1222 DQ 16 x 24 V DC 漏型 (6ES7222-1BH32-1XB0)	
	

表格 A-124 SM 1222 DQ 16 x 24 V DC 漏型 (6ES7222-1BH32-1XB0) 的连接器引脚位置

引脚	X10	X11	X12	X13
1	L+ / 24 V DC	功能性接地	无连接	无连接
2	M / 24 V DC	无连接	无连接	无连接
3	无连接	无连接	无连接	无连接
4	DQ a.0	DQ a.4	DQ b.0	DQ b.4
5	DQ a.1	DQ a.5	DQ b.1	DQ b.5
6	DQ a.2	DQ a.6	DQ b.2	DQ b.6
7	DQ a.3	DQ a.7	DQ b.3	DQ b.7

A.9.4 SM 1223 数字量输入/输出 V DC 规范

表格 A-125 常规规范

型号	SM 1223 DI 8 x 24 V DC、 DQ 8 x 继电器	SM 1223 DI 16 x 24 V DC, DQ 16 x 继电器	SM 1223 DI 8 x 24 V DC、 DQ 8 x 24 V DC	SM 1223 DI 16 x 24 V DC, DQ 16 x 24 V DC	SM 1223 DI 16 x 24 V DC、DQ 16 x 24 V DC 漏型
订货号	6ES7223-1PH32-0XB0	6ES7223-1PL32-0XB0	6ES7223-1BH32-0XB0	6ES7223-1BL32-0XB0	6ES7223-1BL32-1XB0
尺寸 W x H x D (mm)	45 x 100 x 75	70 x 100 x 75	45 x 100 x 75	70 x 100 x 75	70 x 100 x 75
重量	230 g	350 g	210 g	310 g	310 g
功耗	5.5 W	10 W	2.5 W	4.5 W	4.5 W
电流消耗（SM 总线）	145 mA	180 mA	145 mA	185 mA	185 mA
电流消耗 (24 V DC)	所用的每点输入 4 mA 所用的每个继电器线圈 11 mA		150 mA	200 mA	40 mA

表格 A-126 数字量输入

型号	SM 1223 DI 8 x 24 V DC，DQ 8 x 继电器	SM 1223 DI 16 x 24 V DC, DQ 16 x 继电器	SM 1223 DI 8 x 24 V DC，DQ 8 x 24 V DC	SM 1223 DI 16 x 24 V DC, DQ 16 x 24 V DC	SM 1223 DI 16 x 24 V DC、DQ 16 x 24 V DC 漏型
输入点数	8	16	8	16	16
类型	漏型/源型（IEC 1 类漏型）	漏型/源型（IEC 1 类漏型）	漏型/源型（IEC 1 类漏型）	漏型/源型（IEC 1 类漏型）	漏型/源型（IEC 1 类漏型）
额定电压	4 mA 时 24 V DC，额定值	4 mA 时 24 V DC，额定值	4 mA 时 24 V DC，额定值	4 mA 时 24 V DC，额定值	4 mA 时 24 V DC，额定值
允许的连续电压	30 V DC，最大值	30 V DC，最大值	30 V DC，最大值	30 V DC，最大值	30 V DC，最大值
浪涌电压	35 V DC，持续 0.5 s	35 V DC，持续 0.5 s	35 V DC，持续 0.5 s	35 V DC，持续 0.5 s	35 V DC，持续 0.5 s
逻辑 1 信号（最小）	2.5 mA 时 15 V DC	2.5 mA 时 15 V DC	2.5 mA 时 15 V DC	2.5 mA 时 15 V DC	2.5 mA 时 15 V DC
逻辑 0 信号（最大）	1 mA 时 5 V DC	1 mA 时 5 V DC	1 mA 时 5 V DC	1 mA 时 5 V DC	1 mA 时 5 V DC
隔离（现场侧与逻辑侧）	707 V DC（型式测试）	707 V DC（型式测试）	707 V DC（型式测试）	707 V DC（型式测试）	707 V DC（型式测试）
隔离组	2	2	2	2	2
滤波时间	0.2、0.4、0.8、1.6、3.2、6.4 和 12.8 ms（可选 4 个一组）	0.2、0.4、0.8、1.6、3.2、6.4 和 12.8 ms（可选 4 个一组）	0.2、0.4、0.8、1.6、3.2、6.4 和 12.8 ms（可选 4 个一组）	0.2、0.4、0.8、1.6、3.2、6.4 和 12.8 ms（可选 4 个一组）	0.2、0.4、0.8、1.6、3.2、6.4 和 12.8 ms（可选 4 个一组）

型号	SM 1223 DI 8 x 24 V DC，DQ 8 x 继电器	SM 1223 DI 16 x 24 V DC, DQ 16 x 继电器	SM 1223 DI 8 x 24 V DC，DQ 8 x 24 V DC	SM 1223 DI 16 x 24 V DC, DQ 16 x 24 V DC	SM 1223 DI 16 x 24 V DC、DQ 16 x 24 V DC 漏型
同时接通的输入数	8	• 8（无相邻点），60 ℃（水平）或 50 ℃（垂直）时 • 16，55 ℃（水平）或 45 ℃（垂直）时	8	16	16
电缆长度（米）	500 m（屏蔽）； 300 m（非屏蔽）	500 m（屏蔽）， 300 m（非屏蔽）	500 m（屏蔽）， 300 m（非屏蔽）	500 m（屏蔽）， 300 m（非屏蔽）	500 m（屏蔽）， 300 m（非屏蔽）

表格 A-127 数字量输出

型号	SM 1223 DI 8 x 24 V DC，DQ 8 x 继电器	SM 1223 DI 16 x 24 V DC，DQ 16 x 继电器	SM 1223 DI 8 x 24 V DC，DQ 8 x 24 V DC	SM 1223 DI 16 x 24 V DC，DQ 16 x 24 V DC	SM 1223 DI 16 x 24 V DC、DQ 16 x 24 V DC 漏型
输出点数	8	16	8	16	16
类型	继电器，机械式	继电器，机械式	固态 - MOSFET（源型）	固态 - MOSFET（源型）	固态 - MOSFET（漏型）
电压范围	5 到 30 V DC 或 5 到 250 V AC	5 到 30 V DC 或 5 到 250 V AC	20.4 到 28.8 V DC	20.4 到 28.8 V DC	20.4 到 28.8 V DC
最大电流时的逻辑 1 信号	--	--	20 V DC，最小值	20 V DC，最小值	0.5 V DC
具有 10 KΩ 负载时的逻辑 0 信号	--	--	0.1 V DC，最大值	0.1 V DC，最大值	24 V（典型值）- 0.75 V DC
电流（最大）	2.0 A	2.0 A	0.5 A	0.5 A	0.5 A
灯负载	30 W DC/200 W AC	30 W DC/200 W AC	5 W	5 W	5 W

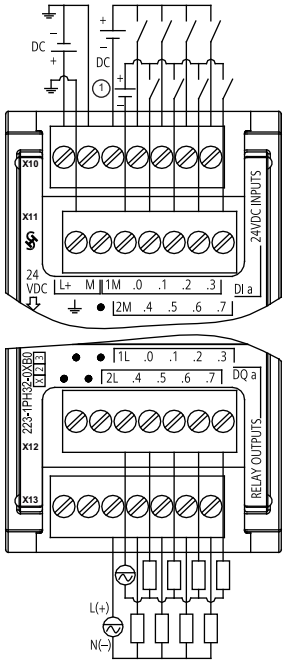
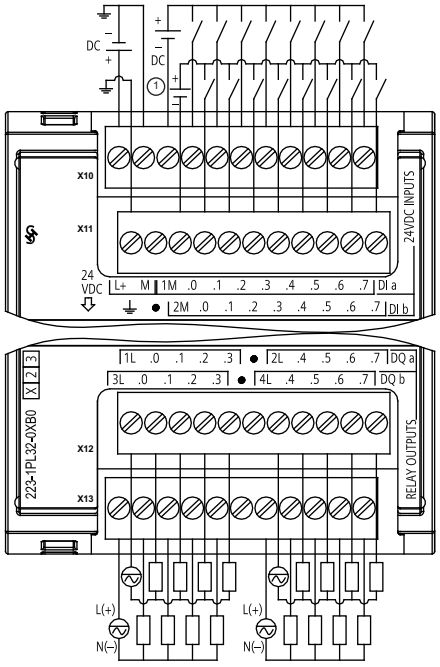
型号	SM 1223 DI 8 x 24 V DC， DQ 8 x 继电器	SM 1223 DI 16 x 24 V DC， DQ 16 x 继电器	SM 1223 DI 8 x 24 V DC， DQ 8 x 24 V DC	SM 1223 DI 16 x 24 V DC， DQ 16 x 24 V DC	SM 1223 DI 16 x 24 V DC、DQ 16 x 24 V DC 漏型
通态触点电阻	新设备最大为 0.2 Ω	新设备最大为 0.2 Ω	最大 0.6 Ω	最大 0.6 Ω	最大 0.5 Ω
每点的漏电流	--	--	最大 10 μA	最大 10 μA	最大 75 μA
浪涌电流	触点闭合时为 7 A	触点闭合时为 7 A	8 A，最长持续 100 ms	8 A，最长持续 100 ms	限流保护
过载保护	×	×	×	×	是，限流保护范围 1 A 到 3.5 A
隔离（现场侧与逻辑侧）	1500 V AC（线圈与触点） 无（线圈与逻辑侧）	1500 V AC（线圈与触点） 无（线圈与逻辑侧）	707 V DC（型式测试）	707 V DC（型式测试）	707 V DC（型式测试）
隔离组	2	4	1	1	1
每个公共端的电流	10A	8 A	4 A	8 A	8 A
电感钳位电压	--	--	L+ - 48 V， 1 W 损耗	L+ - 48 V， 1 W 损耗	45 V
开关延迟	最长 10 ms	最长 10 ms	断开到接通最长为 50 μs 接通到断开最长为 200 μs	断开到接通最长为 50 μs 接通到断开最长为 200 μs	断开到接通最长为 20 μs 接通到断开最长为 350 μs
继电器最大开关频率	1 Hz	1 Hz	--	--	--
机械寿命（无负载）	10,000,000 个断开/闭合周期	10,000,000 个断开/闭合周期	--	--	--
额定负载下的触点寿命（常开触点）	100,000 个断开/闭合周期	100,000 个断开/闭合周期	--	--	--
RUN 到 STOP 时的行为	上一个值或替换值（默认值为 0）	上一个值或替换值（默认值为 0）	上一个值或替换值（默认值为 0）	上一个值或替换值（默认值为 0）	上一个值或替换值（默认值为 0）

型号	SM 1223 DI 8 x 24 V DC， DQ 8 x 继电器	SM 1223 DI 16 x 24 V DC， DQ 16 x 继电器	SM 1223 DI 8 x 24 V DC， DQ 8 x 24 V DC	SM 1223 DI 16 x 24 V DC， DQ 16 x 24 V DC	SM 1223 DI 16 x 24 V DC、DQ 16 x 24 V DC 漏型
同时接通的输出数	8	• 8（无相邻点），60 ℃（水平）或 50 ℃（垂直）时 • 16，55 ℃（水平）或 45 ℃（垂直）时	8	16	16
电缆长度（米）	500 m（屏蔽）； 150 m（非屏蔽）	500 m（屏蔽）， 150 m（非屏蔽）	500 m（屏蔽）， 150 m（非屏蔽）	500 m（屏蔽）， 150 m（非屏蔽）	500 m（屏蔽）， 150 m（非屏蔽）

参见

新功能 (页 37)

表格 A-128 数字量输入 V DC/输出继电器 SM 的接线图

SM 1223 DI 8 x 24 V DC, DQ 8 x 继电器 (6ES7223-1PH32-0XB0)	SM 1223 DI 16 x 24 V DC, DQ 16 x 继电器 (6ES7223-1PL32-0XB0)	注意
		① 对于漏型输入，将“-”连接到“M”（如图所示）。 对于源型输入，将“+”连接到“M”。

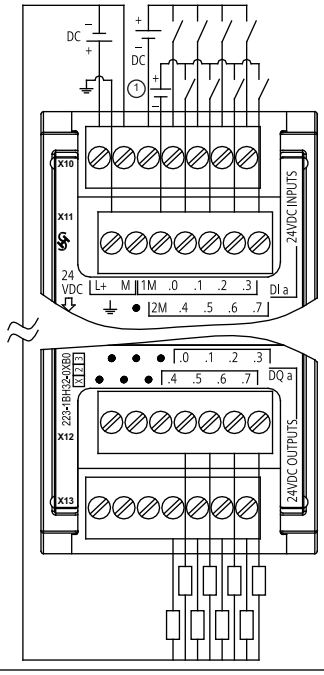
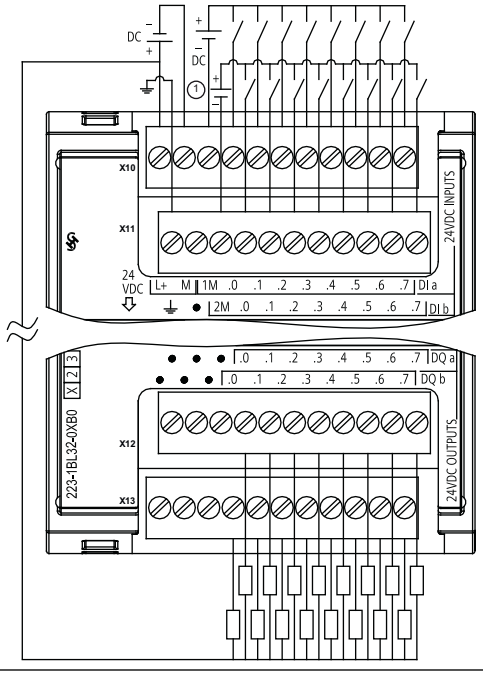
表格 A-129 SM 1223 DI 8 x 24 V DC, DQ 8 x 继电器 (6ES7223-1PH32-0XB0) 的连接器引脚位置

引脚	X10	X11	X12	X13
1	L+ / 24 V DC	功能性接地	无连接	无连接
2	M / 24 V DC	无连接	无连接	无连接
3	1M	2M	1L	2L
4	DI a.0	DI a.4	DQ a.0	DQ a.4
5	DI a.1	DI a.5	DQ a.1	DQ a.5
6	DI a.2	DI a.6	DQ a.2	DQ a.6
7	DI a.3	DI a.7	DQ a.3	DQ a.7

表格 A-130 SM 1223 DI 16 x 24 V DC, DQ 16 x 继电器 (6ES7223-1PL32-0XB0) 的连接器引脚位置

引脚	X10	X11	X12	X13
1	L+ / 24 V DC	功能性接地	1L	3L
2	M / 24 V DC	无连接	DQ a.0	DQ b.0
3	1M	2M	DQ a.1	DQ b.1
4	DI a.0	DI b.0	DQ a.2	DQ b.2
5	DI a.1	DI b.1	DQ a.3	DQ b.3
6	DI a.2	DI b.2	无连接	无连接
7	DI a.3	DI b.3	2L	4L
8	DI a.4	DI b.4	DQ a.4	DQ b.4
9	DI a.5	DI b.5	DQ a.5	DQ b.5
10	DI a.6	DI b.6	DQ a.6	DQ b.6
11	DI a.7	DI b.7	DQ a.7	DQ b.7

表格 A-131 数字量输入 V DC/输出 SM 的接线图

SM 1223 DI 8 x 24 V DC, DQ 8 x 24 V DC (6ES7223-1BH32-0XB0)	SM 1223 DI 16 x 24 V DC, DQ 16 x 24 V DC (6ES7223-1BL32-0XB0)	注意
		① 对于漏型输入，将“-”连接到“M”（如图所示）。 对于源型输入，将“+”连接到“M”。

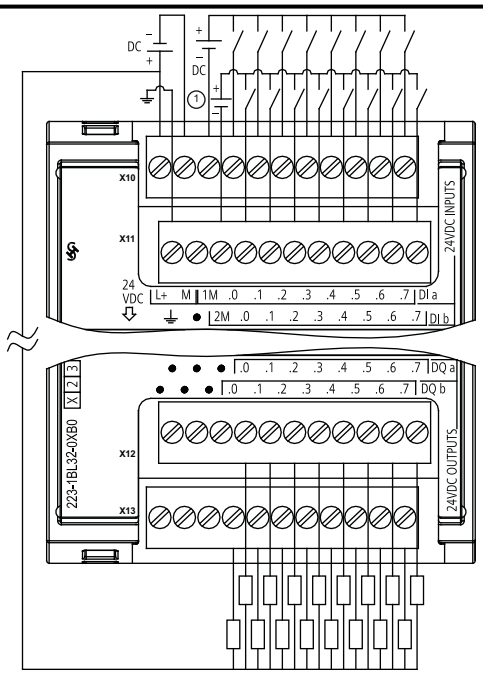
表格 A-132 SM 1223 DI 8 x 24 V DC, DQ 8 x 24 V DC (6ES7223-1BH32-0XB0) 的连接器引脚位置

引脚	X10	X11	X12	X13
1	L+ / 24 V DC	功能性接地	无连接	无连接
2	M / 24 V DC	无连接	无连接	无连接
3	1M	2M	无连接	无连接
4	DI a.0	DI a.4	DQ a.0	DQ a.4
5	DI a.1	DI a.5	DQ a.1	DQ a.5
6	DI a.2	DI a.6	DQ a.2	DQ a.6
7	DI a.3	DI a.7	DQ a.3	DQ a.7

表格 A-133 SM 1223 DI 16 x 24 V DC, DQ 16 x 24 V DC (6ES7223-1BL32-0XB0) 的连接器引脚位置

引脚	X10	X11	X12	X13
1	L+ / 24 V DC	功能性接地	无连接	无连接
2	M / 24 V DC	无连接	无连接	无连接
3	1M	2M	无连接	无连接
4	DI a.0	DI b.0	DQ a.0	DQ b.0
5	DI a.1	DI b.1	DQ a.1	DQ b.1
6	DI a.2	DI b.2	DQ a.2	DQ b.2
7	DI a.3	DI b.3	DQ a.3	DQ b.3
8	DI a.4	DI b.4	DQ a.4	DQ b.4
9	DI a.5	DI b.5	DQ a.5	DQ b.5
10	DI a.6	DI b.6	DQ a.6	DQ b.6
11	DI a.7	DI b.7	DQ a.7	DQ b.7

表格 A-134 数字量输入 V DC/输出 SM 的接线图

SM 1223 DI 16 x 24 V DC、DQ 16 x 24 V DC 漏型 (6ES7223-1BL32-1XB0)	
	

表格 A-135 SM 1223 DI 16 x 24 V DC、DQ 16 x 24 V DC 漏型 (6ES7223-1BL32-1XB0) 的连接器引脚位置

引脚	X10	X11	X12	X13
1	L+ / 24 V DC	功能性接地	无连接	无连接
2	M / 24 V DC	无连接	无连接	无连接
3	1M	2M	无连接	无连接
4	DI a.0	DI b.0	DQ a.0	DQ b.0
5	DI a.1	DI b.1	DQ a.1	DQ b.1
6	DI a.2	DI b.2	DQ a.2	DQ b.2
7	DI a.3	DI b.3	DQ a.3	DQ b.3
8	DI a.4	DI b.4	DQ a.4	DQ b.4
9	DI a.5	DI b.5	DQ a.5	DQ b.5
10	DI a.6	DI b.6	DQ a.6	DQ b.6
11	DI a.7	DI b.7	DQ a.7	DQ b.7

A.9.5 SM 1223 数字量输入/输出 V AC 规范

表格 A-136 常规规范

型号	SM 1223 DI 8 x120/230 V AC/DQ 8 x 继电器
产品编号	6ES7223-1QH32-0XB0
尺寸 W x H x D (mm)	45 x 100 x 75 mm
重量	190 g
功耗	7.5 W
电流消耗（SM 总线）	120 mA
电流消耗 (24 V DC)	接通时每点输出为 11 mA

表格 A-137 数字量输入

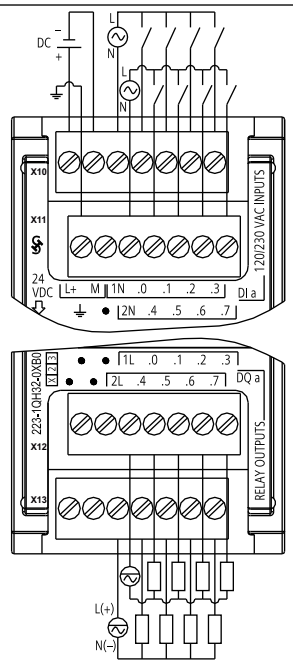
型号	SM 1223 DI 8 x 120/230 V AC/DQ 8 x 继电器
输入点数	8
类型	IEC 1 类
额定电压	6 mA 时 120 V AC，9 mA 时 230 V AC
允许的连续电压	264 V AC
浪涌电压	--
逻辑 1 信号（最小）	2.5 mA 时 79 V DC
逻辑 0 信号（最大）	1 mA 时 20 V DC
漏电流（最大值）	1 mA
隔离（现场侧与逻辑侧）	1500 V AC
隔离组 [1]	4
输入延迟时间	典型：0.2 到 12.8 ms，用户可选择 最大值：-
连接 2 线制接近传感器 (Bero)（最大值）	1 mA
电缆长度	非屏蔽：300 m 屏蔽：500 m
同时接通的输入数	8

[1] 组中通道的相位必须相同。

表格 A-138 数字量输出

型号	SM 1223 DI 8 x 120/230 V AC/DQ 8 x 继电器
输出点数	8
类型	继电器，机械式
电压范围	5 到 30 V DC 或 5 到 250 V AC
最大电流时的逻辑 1 信号	--
具有 10 KΩ 负载时的逻辑 0 信号	--
电流（最大）	2.0 A
灯负载	30 W DC/200 W AC
通态触点电阻	新设备最大为 0.2 Ω
每点的漏电流	--
浪涌电流	触点闭合时为 7 A
过载保护	×
隔离（现场侧与逻辑侧）	1500 V AC（线圈与触点） 无（线圈与逻辑侧）
隔离组	2
每个公共端的电流（最大）	10 A
电感钳位电压	--
切换延迟（最大值）	10 ms
继电器最大开关频率	1 Hz
机械寿命（无负载）	10,000,000 个断开/闭合周期
额定负载下的触点寿命	100,000 个断开/闭合周期
RUN 到 STOP 时的行为	上一个值或替换值（默认值为 0）
同时接通的输出数	• 4（无相邻点），60 ℃（水平）或 50 ℃（垂直）时 • 8，55 ℃（水平）或 45 ℃（垂直）时
电缆长度（米）	500 m（屏蔽）；150 m（非屏蔽）

表格 A-139 SM 1223 DI 8 x 120/230 V AC，DQ 8 x 继电器 (6ES7223-1QH32-0XB0)



表格 A-140 SM 1223 DI 8 x 120/240 V AC，DQ 8 x 继电器 (6ES7223-1QH32-0XB0) 的连接器引脚位置

引脚	X10	X11	X12	X13
1	L+ / 24 V DC	功能性接地	无连接	无连接
2	M/24 V DC	无连接	无连接	无连接
3	1N	2N	1L	2L
4	DI a.0	DI a.4	DQ a.0	DQ a.4
5	DI a.1	DI a.5	DQ a.1	DQ a.5
6	DI a.2	DI a.6	DQ a.2	DQ a.6
7	DI a.3	DI a.7	DQ a.3	DQ a.7

A.10 模拟信号模块 (SM)

A.10.1 SM 1231 模拟量输入模块规范

表格 A-141 常规规范

型号	SM 1231 AI 4 x 13 位	SM 1231 AI 8 x 13 位	SM 1231 AI 4 x 16 位
产品编号	6ES7231-4HD32-0XB0	6ES7231-4HF32-0XB0	6ES7231-5ND32-0XB0
尺寸 W x H x D (mm)	45 x 100 x 75		
重量	180 g		
功耗	2.2 W	2.3 W	2.0 W
电流消耗（SM 总线）	80 mA	90 mA	80 mA
电流消耗 (24 V DC)	45 mA		65 mA

表格 A-142 模拟量输入

型号	SM 1231 AI 4 x 13 位	SM 1231 AI 8 x 13 位	SM 1231 AI 4 x 16 位
输入点数	4	8	4
类型	电压或电流（差动）：可 2 个选为一组		电压或电流（差动）
范围	±10 V、±5 V、±2.5 V、0 到 20 mA 或 4 mA 到 20 mA		±10 V、±5 V、±2.5 V、±1.25 V、0 到 20 mA 或 4 mA 到 20 mA
满量程范围（数据字）	-27648 到 27648 电压/0 到 27648 电流		
过冲/下冲范围（数据字） 请参见电压和电流 (页 1556)的模拟量输入范围部分。	电压：32511 到 27649/-27649 到 -32512 电流：32511 到 27649/0 到 -4864		
上溢/下溢（数据字） 请参见电压和电流 (页 1556)的输入范围部分。	电压：32767 到 32512/-32513 到 -32768 电流 0 到 20 mA：32767 到 32512/-4865 到 -32768 电流 4 到 20 mA：32767 到 32512（值小于 -4864 时表示开路）		
Resolution1	12 位 + 符号位		15 位 + 符号位

型号	SM 1231 AI 4 x 13 位	SM 1231 AI 8 x 13 位	SM 1231 AI 4 x 16 位
最大耐压/耐流	±35 V/±40 mA		
平滑化	无、弱、中或强 请参见阶跃响应时间 (页 1555)部分。		
噪声抑制	400、60、50 或 10 Hz 请参见采样率 (页 1555)部分。		
输入阻抗	≥ 9 MΩ（电压）/ ≥ 270 Ω，< 290 Ω（电流）		≥ 1 MΩ（电压）/ < 315 Ω，> 280 Ω（电流）
隔离 现场侧与逻辑侧 逻辑侧与 24 V DC 现场侧与 24 V DC 通道与通道	无		 707 V DC（型式测试） 707 V DC（型式测试） 500 V DC（型式测试） 无
精度（25 °C/-20 到 60 °C）	满量程的 ±0.1%/±0.2%		满量程的 ±0.1%/±0.3%
测量原理	实际值转换		
共模抑制	40 dB，DC 到 60 Hz		
工作信号范围 [1]	信号加共模电压必须小于 +12 V 且大于 -12 V		
电缆长度（米）	100 m，屏蔽双绞线		

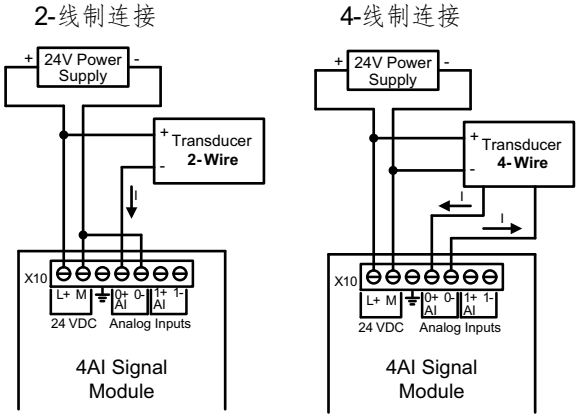
[1] 施加至某一通道的电压超出工作范围可能导致对其它通道造成干扰。

表格 A-143 诊断

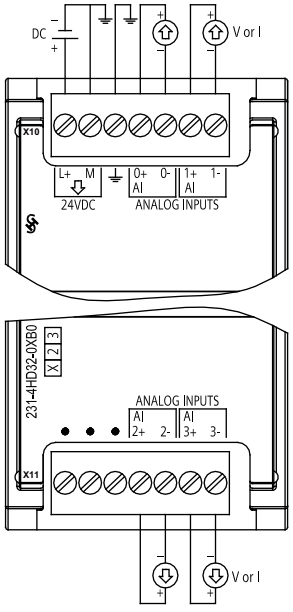
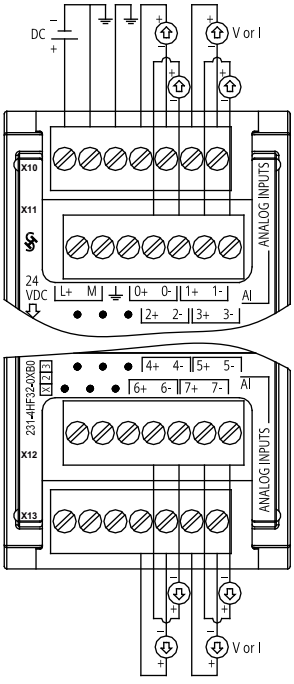
型号	SM 1231 AI 4 x 13 位	SM 1231 AI 8 x 13 位	SM 1231 AI 4 x 16 位
上溢/下溢	√		
24 V DC 低压	√		
开路	仅限 4 到 20 mA 范围（如果输入低于 -4864；1.185 mA）		

SM 1231 电流测量

可以使用 2 线制传感器或 4 线制传感器进行电流测量，如下图所示:



表格 A-144 模拟量输入 SM 的接线图

SM 1231 AI 4 x 13 位 (6ES7231-4HD32-0XB0)	SM 1231 AI 8 x 13 位 (6ES7231-4HF32-0XB0)
	
注：连接器必须镀金。有关订货号，请参见附录 C“备件”。	

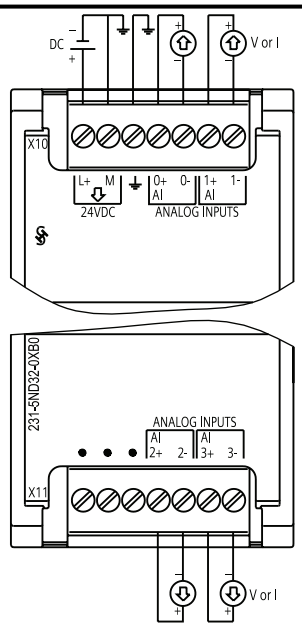
表格 A-145 SM 1231 AI 4 x 13 位 (6ES7231-4HD32-0XB0) 的连接器引脚位置

引脚	X10（镀金）	X11（镀金）
1	L+ / 24 V DC	无连接
2	M / 24 V DC	无连接
3	功能性接地	无连接
4	AI 0+	AI 2+
5	AI 0-	AI 2-
6	AI 1+	AI 3+
7	AI 1-	AI 3-

表格 A-146 SM 1231 AI 8 x 13 位 (6ES7231-4HF32-0XB0) 的连接器引脚位置

引脚	X10（镀金）	X11（镀金）	X12（镀金）	X13（镀金）
1	L+ / 24 V DC	无连接	无连接	无连接
2	M / 24 V DC	无连接	无连接	无连接
3	功能性接地	无连接	无连接	无连接
4	AI 0+	AI 2+	AI 4+	AI 6+
5	AI 0-	AI 2-	AI 4-	AI 6-
6	AI 1+	AI 3+	AI 5+	AI 7+
7	AI 1-	AI 3-	AI 5-	AI 7-

表格 A-147 模拟量输入 SM 的接线图

SM 1231 AI 4 x 16 位 (6ES7231-5ND32-0XB0)	
	
注：连接器必须镀金。有关订货号，请参见附录 C“备件”。	

表格 A-148 SM 1231 AI 4 x 16 位 (6ES7231-5ND32-0XB0) 的连接器引脚位置

引脚	X10（镀金）	X11（镀金）
1	L+ / 24 V DC	无连接
2	M / 24 V DC	无连接
3	功能性接地	无连接
4	AI 0+	AI 2+
5	AI 0-	AI 2-
6	AI 1+	AI 3+
7	AI 1-	AI 3-

说明

应将未使用的电压输入通道短路。

应将未使用的电流输入通道设置在 0 至 20 mA 范围内，和/或禁用断路错误报告功能。

除非模块已上电且已组态，否则组态为电流模式的输入不会传导回路电流。

除非通过外部电源为发送器供电，否则电流输入通道不会工作。

A.10.2 SM 1232 模拟量输出模块规范

表格 A-149 常规规范

技术数据	SM 1232 AQ 2 x 14 位	SM 1232 AQ 4 x 14 位
产品编号	6ES7232-4HB32-0XB0	6ES7232-4HD32-0XB0
尺寸 W x H x D (mm)	45 x 100 x 75	
重量	180 g	
功耗	1.8 W	2.0 W
电流消耗（SM 总线）	80 mA	
电流消耗 (24 V DC)	45 mA（无负载）	

表格 A-150 模拟量输出

技术数据	SM 1232 AQ 2 x 14 位	SM 1232 AQ 4 x 14 位
输出点数	2	4
类型	电压或电流	
范围	±10 V、0 到 20 mA 或 4 mA 到 20 mA	
分辨率	电压：14 位 电流：13 位	
满量程范围（数据字）	电压：-27,648 到 27,648；电流：0 到 27,648 请参见电压和电流 (页 1557)的输出范围。	
精度（25 °C/-20 到 60 °C）	满量程的 ±0.3%/±0.6%	
稳定时间（新值的 95%）	电压：300 µs (R)，750 µs (1 uF) 电流：600 µs (1 mH)，2 ms (10 mH)	
负载阻抗	电压：≥ 1000 Ω 电流：≤ 600 Ω	
最大输出短路电流	电压模式：≤ 24 mA 电流模式：≥ 38.5 mA	
RUN 到 STOP 时的行为	上一个值或替换值（默认值为 0）	
隔离（现场侧与逻辑侧）	无	
隔离（24 V 与输出）	无	
电缆长度（米）	100 m 屏蔽双绞线	

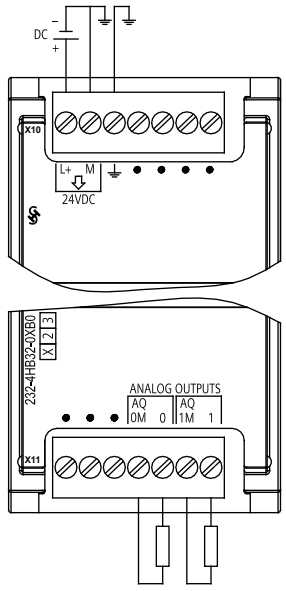
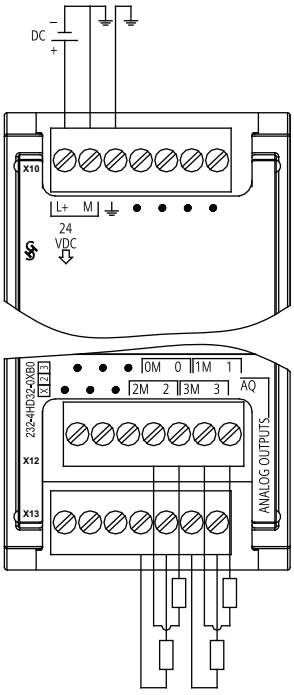
表格 A-151 诊断

技术数据	SM 1232 AQ 2 x 14 位	SM 1232 AQ 4 x 14 位
上溢/下溢	√	
对地短路（仅限电压模式）	√	
断线（仅限电流模式）[1]	√	
24 V DC 低压 [2]	√	

[1] 仅当输出电压小于 -0.5 V 或大于 +0.5 V 时，才能进行短路检测。

[2] 仅当输出电流大于 1 mA 时，才能进行断线检测。

表格 A-152 模拟量输出 SM 的接线图

SM 1232 AQ 2 x 14 位 (6ES7232-4HB32-0XB0)	SM 1232 AQ 4 x 14 位 (6ES7232-4HD32-0XB0)
	
注：连接器必须镀金。有关订货号，请参见附录 C“备件”。	

表格 A-153 SM 1232 AQ 2 x 14 位 (6ES7232-4HB32-0XB0) 的连接器引脚位置

引脚	X10（镀金）	X11（镀金）
1	L+ / 24 V DC	无连接
2	M / 24 V DC	无连接
3	功能性接地	无连接

引脚	X10（镀金）	X11（镀金）
4	无连接	AQ 0M
5	无连接	AQ 0
6	无连接	AQ 1M
7	无连接	AQ 1

表格 A-154 SM 1232 AQ 4 x 14 位 (6ES7232-4HD32-0XB0) 的连接器引脚位置

引脚	X10（镀金）	X12（镀金）	X13（镀金）
1	L+ / 24 V DC	无连接	无连接
2	M / 24 V DC	无连接	无连接
3	功能性接地	无连接	无连接
4	无连接	AQ 0M	AQ 2M
5	无连接	AQ 0	AQ 2
6	无连接	AQ 1M	AQ 3M
7	无连接	AQ 1	AIQ 3

A.10.3 SM 1234 模拟量输入/输出模块规范

表格 A-155 常规规范

技术数据	SM 1234 AI 4 x 13 位/AQ 2 x 14 位
订货号	6ES7234-4HE32-0XB0
尺寸 W x H x D (mm)	45 x 100 x 75
重量	220 g
功耗	2.4 W
电流消耗（SM 总线）	80 mA
电流消耗 (24 V DC)	60 mA（无负载）

表格 A-156 模拟量输入

型号	SM 1234 AI 4 x 13 位/AQ 2 x 14 位
输入点数	4
类型	电压或电流（差动）：可 2 个选为一组
范围	±10 V、±5 V、±2.5 V、0 到 20 mA 或 4 mA 到 20 mA
满量程范围（数据字）	-27648 到 27648
过冲/下冲范围 （数据字）	电压：32511 到 27649/-27649 到 -32512 电流：32511 到 27649/0 到 -4864 请参见电压和电流 (页 1556)的输入范围部分。
上溢/下溢（数据字）	电压：32767 到 32512/-32513 到 -32768 电流：32767 到 32512/-4865 到 -32768 请参见电压和电流 (页 1556)的输入范围部分。
分辨率	12 位 + 符号位
最大耐压/耐流	±35 V/±40 mA
平滑化	无、弱、中或强 请参见阶跃响应时间 (页 1555)部分。
噪声抑制	400、60、50 或 10 Hz 请参见采样率 (页 1555)部分。
输入阻抗	≥ 9 MΩ（电压）/≥ 270 Ω，< 290 Ω（电流）
隔离（现场侧与逻辑侧）	无
精度（25 °C/-20 到 60 °C）	满量程的 ±0.1%/±0.2%
模数转换时间	625 µs（400 Hz 抑制）
共模抑制	40 dB，DC 到 60 Hz
工作信号范围 [1]	信号加共模电压必须小于 +12 V 且大于 -12 V
电缆长度（米）	100 m，屏蔽双绞线

[1] 施加至某一通道的电压超出工作范围可能导致对其它通道造成干扰。

表格 A-157 模拟量输出

技术数据	SM 1234 AI 4 x 13 位/AQ 2 x 14 位
输出点数	2
类型	电压或电流
范围	±10 V 或 0 到 20 mA 或 4 mA 到 20 mA

技术数据	SM 1234 AI 4 x 13 位/AQ 2 x 14 位
分辨率	电压：14 位；电流：13 位
满量程范围（数据字）	电压：-27648 到 27648；电流：0 到 27648 请参见电压和电流 (页 1557)的输出范围部分。
精度（25 °C/-20 到 60 °C）	满量程的 ±0.3%/±0.6%
稳定时间（新值的 95%）	电压：300 µs (R)，750 µs (1 uF) 电流：600 µs (1 mH)，2 ms (10 mH)
负载阻抗	电压：≥ 1000 Ω 电流：≤ 600 Ω
最大输出短路电流	电压模式：≤ 24 mA 电流模式：≥ 38.5 mA
RUN 到 STOP 时的行为	上一个值或替换值（默认值为 0）
隔离（现场侧与逻辑侧）	无
隔离（24 V 与输出）	无
电缆长度（米）	100 m 屏蔽双绞线

表格 A-158 诊断

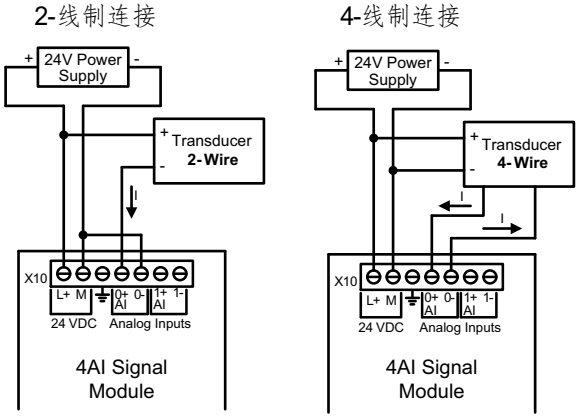
型号	SM 1234 AI 4 x 13 位/AQ 2 x 14 位
上溢/下溢	√
对地短路（仅限电压模式）[1]	输出端有
断线（仅限电流模式）[2]	输出端有
24 V DC 低压	√

[1] 仅当输出电压小于 -0.5 V 或大于 +0.5 V 时，才能进行短路检测。

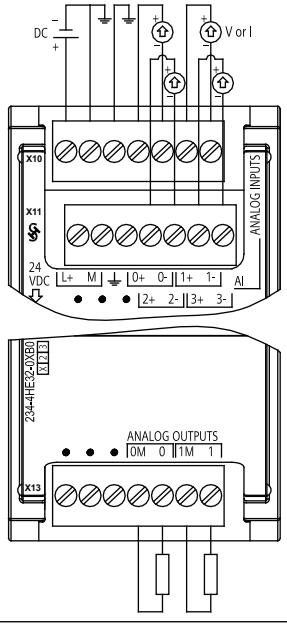
[2] 仅当输出电流大于 1 mA 时，才能进行断线检测。

SM 1234 电流测量

可以使用 2 线制传感器或 4 线制传感器进行电流测量，如下图所示:



表格 A-159 模拟量输入/输出 SM 的接线图

SM 1234 AI 4 x 13 位/AQ 2 x 14 位 (6ES7234-4HE32-0XB0)	
	
注：连接器必须镀金。有关订货号，请参见附录 C“备件”。	

表格 A-160 SM 1234 AI 4 x 13 位/AQ 2 x 14 位 (6ES7234-4HE32-0XB0) 的连接器引脚位置

引脚	X10（镀金）	X11（镀金）	X13（镀金）
1	L+ / 24 V DC	无连接	无连接
2	M / 24 V DC	无连接	无连接
3	功能性接地	无连接	无连接
4	AI 0+	AI 2+	AQ 0M
5	AI 0-	AI 2-	AQ 0
6	AI 1+	AI 3+	AQ 1M
7	AI 1-	AI 3-	AQ 1

说明

应将未使用的电压输入通道短路。

应将未使用的电流输入通道设置在 0 至 20 mA 范围内，和/或禁用断路错误报告功能。

除非模块已上电且已组态，否则组态为电流模式的输入不会传导回路电流。

除非通过外部电源为发送器供电，否则电流输入通道不会工作。

A.10.4 模拟量输入的阶跃响应

表格 A-161 阶跃响应 (ms)，0 到满量程（在 95% 处测得）

平滑化选项（采样平均）	噪声消减/抑制频率（积分时间选项）			
	400 Hz (2.5 ms)	60 Hz (16.6 ms)	50 Hz (20 ms)	10 Hz (100 ms)
无（1 个周期）：不求平均值	4 ms	18 ms	22 ms	100 ms
弱（4 个周期）：4 次采样	9 ms	52 ms	63 ms	320 ms
中（16 个周期）：16 次采样	32 ms	203 ms	241 ms	1200 ms
强（32 个周期）：32 次采样	61 ms	400 ms	483 ms	2410 ms

A.10.5 模拟量输入的采样时间和更新时间

表格 A-162 所有通道的采样时间和模块更新时间

抑制频率（积分时间）	所有通道的采样时间和模块更新时间			
	400 Hz (2.5 ms)	60 Hz (16.6 ms)	50 Hz (20 ms)	10 Hz (100 ms)
4 通道 x 13 位 SM	0.625 ms	4.17 ms	5 ms	25 ms
8 通道 x 13 位 SM	1.25 ms	4.17 ms	5 ms	25 ms
4 通道 x 16 位 SM	0.417 ms	0.397 ms	0.400 ms	0.400 ms

A.10.6 模拟量输入的电压和电流测量范围（SB 和 SM）

表格 A-163 模拟量输入的电压表示法（SB 和 SM）

系统		电压测量范围				
十进制	十六进制	±10 V	±5 V	±2.5 V	±1.25 V	
32767	7FFF[1]	11.851 V	5.926 V	2.963 V	1.481 V	上溢
32512	7F00					
32511	7EFF	11.759 V	5.879 V	2.940 V	1.470 V	过冲范围
27649	6C01					
27648	6C00	10 V	5 V	2.5 V	1.250 V	额定范围
20736	5100	7.5 V	3.75 V	1.875 V	0.938 V	
1	1	361.7 µV	180.8 µV	90.4 µV	45.2 µV	
0	0	0 V	0 V	0 V	0 V	
-1	FFFF					
-20736	AF00	-7.5 V	-3.75 V	-1.875 V	-0.938 V	
-27648	9400	-10 V	-5 V	-2.5 V	-1.250 V	
-27649	93FF					下冲范围
-32512	8100	-11.759 V	-5.879 V	-2.940 V	-1.470 V	
-32513	80FF					下溢
-32768	8000	-11.851 V	-5.926 V	-2.963 V	-1.481 V	

[1] 返回 7FFF 可能由以下原因之一所致：上溢（如上表所述）、有效值可用前（例如上电时立即返回）或者检测到断路。

表格 A-164 模拟量输入的电流表示法（SB 和 SM）

系统		电流测量范围		
十进制	十六进制	0 mA 到 20 mA	4 mA 到 20 mA	
32767	7FFF	> 23.52 mA	> 22.81 mA	上溢
32511	7EFF	23.52 mA	22.81 mA	过冲范围
27649	6C01			
27648	6C00	20 mA	20 mA	额定范围
20736	5100	15 mA	16 mA	
1	1	723.4 nA	4 mA + 578.7 nA	
0	0	0 mA	4 mA	
-1	FFFF			下冲范围
-4864	ED00	-3.52 mA	1.185 mA	
32767[1]	7FFF		< 1.185 mA	断路（4 至 20 mA）
-32768	8000	< -3.52 mA		下溢（0 到 20 mA）

[1] 无论断路报警的状态如何，始终会返回断路值 32767 (16#7FFF)。

参见

确定 SM 1231 模块的断路条件类型 (页 1417)

A.10.7 模拟量输出的电压和电流测量范围（SB 和 SM）

表格 A-165 模拟量输出的电压表示法（SB 和 SM）

系统		电压输出范围	
十进制	十六进制	±10 V	
32767	7FFF	请参见注 1	上溢
32512	7F00	请参见注 1	
32511	7EFF	11.76 V	过冲范围
27649	6C01		

系统		电压输出范围	
十进制	十六进制	±10 V	
27648	6C00	10 V	额定范围
20736	5100	7.5 V	
1	1	361.7 µ V	
0	0	0 V	
-1	FFFF	-361.7 µ V	
-20736	AF00	-7.5 V	
-27648	9400	-10 V	
-27649	93FF		下冲范围
-32512	8100	-11.76 V	
-32513	80FF	请参见注 1	下溢
-32768	8000	请参见注 1	

[1] 在上溢或下溢情况下，模拟量输出将采用 STOP 模式的替代值。

表格 A-166 模拟量输出的电流表示法（SB 和 SM）

系统		当前输出范围		
十进制	十六进制	0 mA 到 20 mA	4 mA 到 20 mA	
32767	7FFF	请参见注 1	请参见注 1	上溢
32512	7F00	请参见注 1	请参见注 1	
32511	7EFF	23.52 mA	22.81 mA	过冲范围
27649	6C01			
27648	6C00	20 mA	20 mA	额定范围
20736	5100	15 mA	16 mA	
1	1	723.4 nA	4 mA + 578.7 nA	
0	0	0 mA	4mA	
-1	FFFF		4 mA 到 578.7 nA	下冲范围
-6912	E500		0 mA	
-6913	E4FF			不可能。 输出值限制在 0 mA。
-32512	8100			

系统		当前输出范围		
十进制	十六进制	0 mA 到 20 mA	4 mA 到 20 mA	
-32513	80FF	请参见注 1	请参见注 1	下溢
-32768	8000	请参见注 1	请参见注 1	

[1] 在上溢或下溢情况下，模拟量输出将采用 STOP 模式的替代值。

A.11 热电偶和 RTD 信号模块 (SM)

A.11.1 SM 1231 热电偶

表格 A-167 常规规范

型号	SM 1231 AI 4 x 16 位 TC	SM 1231 AI 8 x 16 位 TC
产品编号	6ES7231-5QD32-0XB0	6ES7231-5QF32-0XB0
尺寸 W x H x D (mm)	45 x 100 x 75	
重量	180 g	190 g
功耗	1.5 W	
电流消耗（SM 总线）	80 mA	
电流消耗 (24 V DC) [1]	40 mA	

[1] 20.4 到 28.8 V DC（2 类受限制电源，或 PLC 提供的传感器电源）

表格 A-168 模拟量输入

型号		SM 1231 AI 4 x 16 位 TC	SM 1231 AI 8 x 16 位 TC
输入点数		4	8
范围 额定范围（数据字） 过量程/欠量程（数据字） 上溢/下溢（数据字）		请参见热电偶选型表 (页 1563)。	
分辨率	温度	0.1 °C/0.1 °F	
	电压	15 位 + 符号	
最大耐压		±35 V	

型号		SM 1231 AI 4 x 16 位 TC	SM 1231 AI 8 x 16 位 TC
噪声抑制		85 dB，对于所选滤波器设置（10 Hz、50 Hz、60 Hz 或 400 Hz）	
共模抑制		120 VAC 时大于 120 dB	
阻抗		≥ 10 MΩ	
隔离	现场侧与逻辑侧	707 VDC（型式测试）	
	现场侧与 24 V DC	707 V DC（型式测试）	
	24 V DC 与逻辑侧	707 V DC（型式测试）	
通道间		120 V AC	
精度		请参见热电偶选型表 (页 1563)。	
可重复性		±0.05% FS	
测量原理		积分型	
模块更新时间		请参见噪声消减选项表 (页 1563)。	
冷端误差		±1.5 °C	
电缆长度（米）		到传感器最长为 100 米	
导线电阻		最大 100 Ω	

表格 A-169 诊断

型号	SM 1231 AI 4 x 16 位 TC	SM 1231 AI 8 x 16 位 TC
上溢/下溢 [1]	√	
断路 [2, 3]	√	
24 V DC 低压 [1]	√	

[1] 上溢、下溢和低压诊断报警信息将以模拟数据值的形式报告，即使在模块组态中禁用这些报警也会如此。

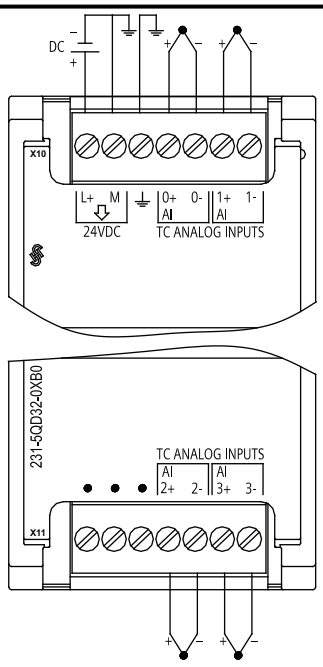
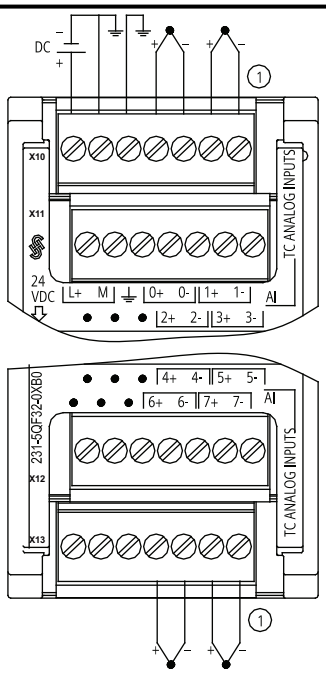
[2] 如果断线报警已禁用，但传感器接线存在开路情况，则模块可能会报告随机值。

[3] 模块每 6 秒执行一次断路测试，这样每 6 秒会针对每个使能通道将更新时间延长 9 ms。

SM 1231 热电偶 (TC) 模拟量信号模块可测量连接到模块输入的电压值。温度测量类型可以是“热电偶”或“电压”类型。

- “热电偶”：将度数乘 10 得到该值（例如，25.3 度将报告为十进制数 253）。
- “电压”：额定范围的满量程值将是十进制数 27648。

表格 A-170 热电偶 SM 的接线图

SM 1231 AI 4 x TC 16 位 (6ES7231-5QD32-0XB0)	SM 1231 AI 8 x TC 位 (6ES7231-5QF32-0XB0)
	
注：连接器必须镀金。有关订货号，请参见附录 C“备件”。	

① 为清晰起见，未显示 TC 2、3、4 和 5 的连接。

表格 A-171 SM 1231 AI 4 x TC 16 位 (6ES7231-5QD32-0XB0) 的连接器引脚位置

引脚	X10（镀金）	X11（镀金）
1	L+ / 24 V DC	无连接
2	M / 24 V DC	无连接
3	功能性接地	无连接
4	AI 0+/TC	AI 2+/TC
5	AI 0-/TC	AI 2-/TC
6	AI 1+/TC	AI 3+/TC
7	AI 1-/TC	AI 3-/TC

表格 A-172 SM 1231 AI 8 x TC 位 (6ES7231-5QF32-0XB0) 的连接器引脚位置

引脚	X10（镀金）	X11（镀金）	X12（镀金）	X13（镀金）
1	L+ / 24 V DC	无连接	无连接	无连接
2	M / 24 V DC	无连接	无连接	无连接
3	功能性接地	无连接	无连接	无连接
4	AI 0+/TC	AI 2+/TC	AI 4+ /TC	AI 6+ /TC
5	AI 0-/TC	AI 2-/TC	AI 4- /TC	AI 6- /TC
6	AI 1+/TC	AI 3+/TC	AI 5+ /TC	AI 7+ /TC
7	AI 1-/TC	AI 3-/TC	AI 5- /TC	AI 7- /TC

说明

应将未使用的模拟量输入短路。

可以取消激活热电偶的未使用通道。如果取消激活未使用的通道，不会出现任何错误。

A.11.1.1 热电偶的基本操作

两种不同的金属彼此之间存在电气连接时，便会形成热电偶。 热电偶产生的电压与结点温度成正比。 电压很小；一微伏能表示很多度。 测量热电偶产生的电压，对额外的结点进行补偿，然后将测量结果线性化，这些是使用热电偶测量温度的基础。

将热电偶连接到 SM 1231 热电偶模块时，两条不同的金属线需连接到模块的信号连接器上。 这两条不同的金属线互相连接的位置即形成了传感器热电偶。

在这两条不同的金属线与信号连接器相连的位置，构成了另外二个热电偶。 连接器温度会引起一定的电压，该电压将添加到传感器热电偶产生的电压中。 如果不对该电压进行修正，结果报告的温度将偏离传感器温度。

冷端补偿便是用于对连接器热电偶进行补偿。 热电偶表是基于参比端温度（通常是零摄氏度）得来的。 冷端补偿用于将连接器温度修正为零摄氏度。 冷端补偿可消除连接器热电偶增加的电压。 模块的温度在内部测量，然后转换为数值并添加到传感器换算中。 之后是使用热电偶表对修正后的传感器换算值进行线性化。

为使冷端补偿取得最佳效果，必须将热电偶模块安装在温度稳定的环境中。 符合模块规范的模块环境温度的缓慢变化（低于 0.1 ℃/分钟）能够被正确补偿。 穿过模块的空气流动也会引起冷端补偿误差。

如果需要更佳的冷端误差补偿效果，则可使用外部 iso 热端子块。 热电偶模块可以使用 0 ℃ 基准值或 50 ℃ 基准值端子块。

A.11.1.2 SM 1231 热电偶选型表

下表给出了 SM 1231 热电偶信号模块支持的不同热电偶类型对应的测量范围和精度。

表格 A-173 热电偶选型表

类型	低于范围最小值 [1]	额定范围下限	额定范围上限	超出范围最大值 [2]	25 °C 时的额定范围 [3, 4] 精度	-20 °C 到 60 °C 时的额定范围 [1, 2] 精度
J	-210.0 ℃	-150.0 ℃	1200.0 ℃	1450.0 ℃	±0.3 °C	±0.6 °C
	-346.0 °F	-238.0 °F	2192.0 °F	2642.0 °F	±0.5 °F	±1.1 °F
K	-270.0 ℃	-200.0 ℃	1372.0 ℃	1622.0 ℃	±0.4 °C	±1.0 °C
	-454.0 °F	-328.0 °F	2501.6 °F	2951.6 °F	±0.7 °F	±1.8 °F
T	-270.0 ℃	-200.0 ℃	400.0 ℃	540.0 ℃	±0.5 °C	±1.0 °C
	-454.0 °F	-328.0 °F	752.0 °F	1004.0 °F	±0.9 °F	±1.8 °F
E	-270.0 ℃	-200.0 ℃	1000.0 ℃	1200.0 ℃	±0.3 °C	±0.6 °C
	-454.0 °F	-328.0 °F	1832.0 °F	2192.0 °F	±0.5 °F	±1.1 °F
R & S	-50.0 ℃	100.0 ℃	1768.0 ℃	2019.0 ℃	±1.0 °C	±2.5 °C
	-58.0 °C	212.0 °F	3214.4 °F	3276.6 °F [5]	±1.8 °F	±4.5 °F
B	0.0 ℃	200.0 ℃	800.0 ℃	--	±2.0 °C	±2.5 °C
	32.0 °F	392.0 °F	1472.0 °F	--	±3.6 °F	±4.5 °F
	--	800.0 ℃	1820.0 ℃	1820.0 ℃	±1.0 °C	±2.3 °C
	--	1472.0 °F	3276.6 °F [5]	3276.6 °F [5]	±1.8 °F	±4.1 °F
N	-270.0 ℃	-200.0 ℃	1300.0 ℃	1550.0 ℃	±1.0 °C	±1.6 °C
	-454.0 °F	-328.0 °F	2372.0 °F	2822.0 °F	±1.8 °F	±2.9 °F
C	0.0 ℃	100.0 ℃	2315.0 ℃	2500.0 ℃	±0.7 °C	±2.7 °C
	32.0 °F	212.0 °F	3276.6 °F [5]	3276.6 °F [5]	±1.3 °F	±4.9 °F

类型	低于范围最小值 [1]	额定范围下限	额定范围上限	超出范围最大值 [2]	25 °C 时的额定范围 [3, 4] 精度	-20 °C 到 60 °C 时的额定范围 [1, 2] 精度
TXK/XK(L)	-200.0 ℃	-150.0 ℃	800.0 ℃	1050.0 ℃	±0.6 °C	±1.2 °C
	-328.0 °F	302.0 °F	1472.0 °F	1922.0 °F	±1.1 °F	±2.2 °F
电压	-32512	-27648 -80mV	27648 80mV	32511	±0.05%	±0.1%

1 “低于范围最小值”以下的热电偶值报告为 -32768。

2 “超过范围最大值”以上的热电偶值报告为 32767。

3 所有范围的内部冷端误差均为 ±1.5 °C。该误差已包括到本表的误差中。模块需要至少 30 分钟的预热时间才能满足该规范。

4 若是暴露在 970 MHz 到 990 MHz 的无线电辐射频率下，SM 1231 AI 4 x 16 位 TC 的精度可能会有所下降。

5 下限 3276.6，含 °F 报告

说明

热电偶通道

热电偶信号模块上的各个通道可组态为不同的热电偶类型（可在组态模块期间进行选择）。

表格 A-174 SM 1231 热电偶的噪声消减和更新时间

抑制频率选择	积分时间	4 通道模块更新时间（秒）	8 通道模块更新时间（秒）
400 Hz (2.5 ms)	10 ms [1]	0.143	0.285
60 Hz (16.6 ms)	16.67 ms	0.223	0.445
50 Hz (20 ms)	20 ms	0.263	0.525
10 Hz (100 ms)	100 ms	1.225	2.450

1 当选择 400 Hz 抑制时，为保证模块分辨率及精度，积分时间应为 10 ms。同时，该选择也会抑制频率为 100 Hz 和 200 Hz 的噪声。

测量热电偶时建议使用 100 ms 的积分时间。使用更小的积分时间将增大温度读数的重复性误差。

说明

对模块上电后，模块将对模数转换器执行内部校准。在此期间，模块将报告每个通道的值为 32767，直到相应通道出现有效值为止。用户程序可能需要考虑这段初始化时间。由于模块的组态可能改变初始化时长，因此，应验证组态中模块的行为。如果需要，可以在用户程序中包含逻辑，以适应模块的初始化时间。

J 型热电偶模拟值的表示

J 型热电偶模拟值的表示如下表所示。

表格 A-175 J 型热电偶模拟值的表示

用 °C 表示的 J 型	功能单元		用 °F 表示的 J 型	功能单元		范围
	十进制	十六进制		十进制	十六进制	
> 1450.0	32767	7FFF	> 2642.0	32767	7FFF	上溢
1450.0 : 1200.1	14500 : 12001	38A4 : 2EE1	2642.0 : 2192.2	26420 : 21922	6734 : 55A2	超出上限
1200.0 : -150.0	12000 : -1500	2EE0 : FA24	2192.0 : -238.0	21920 : -2380	55A0 : F6B4	额定范围
< -150.0	-32768	8000	< -238.0	-32768	8000	下溢 [1]

[1] 如果发生接线错误（例如极性接反或输入开路），或者传感器在负测量范围内出现故障（例如，热电偶类型错误），可能会导致热电偶模块信号超出下限。

A.11.2 SM 1231 RTD

SM 1231 RTD 规范

表格 A-176 常规规范

技术数据	SM 1231 AI 4 x RTD x 16 位	SM 1231 AI 8 x RTD x 16 位
产品编号	6ES7231-5PD32-0XB0	6ES7231-5PF32-0XB0
尺寸 W x H x D (mm)	45 x 100 x 75	70 x 100 x 75
重量	220 g	270 g
功耗	1.5 W	
电流消耗（SM 总线）	80 mA	90 mA
电流消耗 (24 V DC) [1]	40 mA	

[1] 20.4 到 28.8 V DC（2 类受限制电源，或 CPU 提供的传感器电源）

表格 A-177 模拟量输入

技术数据		SM 1231 AI 4 x RTD x 16 位	SM 1231 AI 8 x RTD x 16 位
输入点数		4	8
类型		模块参考 RTD 和 Ω	
范围 额定范围（数据字） 过冲/下冲范围（数据字） 上溢/下溢（数据字）		请参见 RTD 传感器选型表 (页 1570)。	
分辨率	温度	0.1 °C/0.1 °F	
	电阻	15 位 + 符号	
最大耐压		±35 V	
噪声抑制		对于所选噪声消减为 85 dB（10 Hz、50 Hz、60 Hz 或 400 Hz）	
共模抑制		> 120dB	
阻抗		≥ 10 MΩ	
隔离	现场侧与逻辑侧	707 V DC（型式测试）	
	现场侧与 24 V DC	707 V DC（型式测试）	
	24 V DC 与逻辑侧	707 V DC（型式测试）	

技术数据	SM 1231 AI 4 x RTD x 16 位	SM 1231 AI 8 x RTD x 16 位
通道间隔离	无	
精度	请参见 RTD 传感器选型表 (页 1570)。	
可重复性	±0.05% FS	
最大传感器功耗	0.5 m W	
测量原理	积分型	
模块更新时间	请参见噪声消减选项表 (页 1570)。	
电缆长度（米）	到传感器最长为 100 米	
导线电阻	20 Ω，对于 10 Ω RTD，最大为 2.7 Ω	

表格 A-178 诊断

技术数据	SM 1231 AI 4 x RTD x 16 位	SM 1231 AI 8 x RTD x 16 位
上溢/下溢 [1,2]	√	
断线 [3]	√	
24 V DC 低压 [1]	√	

1 上溢、下溢和低压诊断报警信息将以模拟数据值的形式报告，即使在模块组态中禁用这些报警也会如此。

2 对于电阻范围，始终会禁用下溢检测。

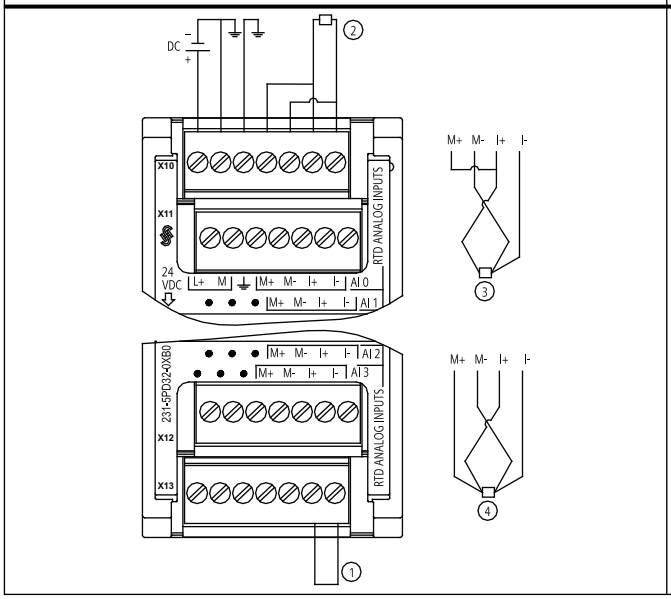
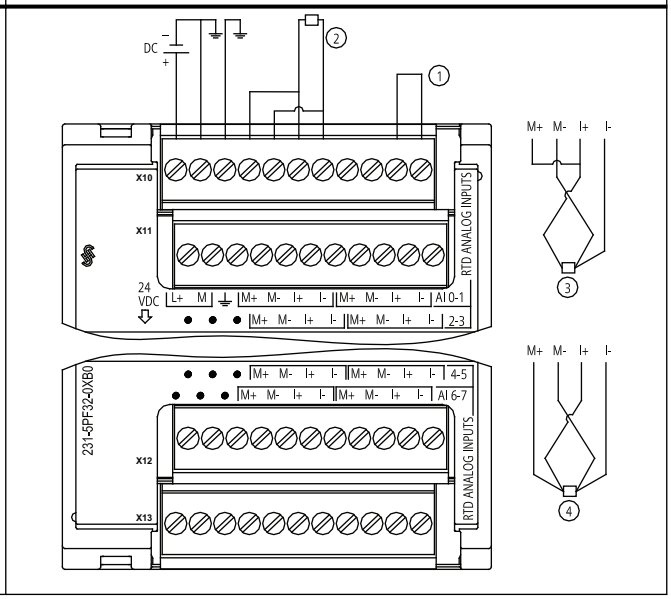
3 如果断线报警已禁用，但传感器接线存在开路情况，则模块可能会报告随机值。

SM 1231 RTD 模拟量信号模块可测量连接到模块输入的电阻值。测量类型可选为“电阻”型或“热电阻”型。

- “电阻”：额定范围的满量程值将是十进制数 27648。
- “热电阻”：将度数乘 10 得到该值（例如，25.3 度将报告为十进制数 253）。将度数乘 100 得到气候范围值（例如，25.34 度将报告为十进制数 2534）。

SM 1231 RTD 模块支持采用 2 线、3 线和 4 线制方式连接到传感器电阻进行测量。

表格 A-179 RTD SM 的接线图

SM 1231 RTD 4 x 16 位 (6ES7231-5PD32-0XB0)	SM 1231 RTD 8 x 16 位 (6ES7231-5PF32-0XB0)
	

① 环接未使用的 RTD 输入

② 2 线制 RTD ③ 3 线制 RTD ④ 4 线制 RTD

注：连接器必须镀金。有关订货号，请参见附录 C“备件”。

表格 A-180 SM 1231 RTD 4 x 16 位 (6ES7231-5PD32-0XB0) 的连接器引脚位置

引脚	X10（镀金）	X11（镀金）	X12（镀金）	X13（镀金）
1	L+ / 24 V DC	无连接	无连接	无连接
2	M / 24 V DC	无连接	无连接	无连接
3	功能性接地	无连接	无连接	无连接
4	AI 0 M+/RTD	AI 1 M+/RTD	AI 2 M+/RTD	AI 3 M+/RTD
5	AI 0 M-/RTD	AI 1 M-/RTD	AI 2 M-/RTD	AI 3 M-/RTD
6	AI 0 I+/RTD	AI 1 I+/RTD	AI 2 I+/RTD	AI 3 I+/RTD
7	AI 0 I-/RTD	AI 1 I-/RTD	AI 2 I-/RTD	AI 3 I-/RTD

表格 A-181 SM 1231 RTD 8 x 16 位 (6ES7231-5PF32-0XB0) 的连接器引脚位置

引脚	X10（镀金）	X11（镀金）	X12（镀金）	X13（镀金）
1	L+ / 24 V DC	无连接	无连接	无连接
2	M / 24 V DC	无连接	无连接	无连接
3	功能性接地	无连接	无连接	无连接
4	AI 0 M+/RTD	AI 2 M+/RTD	AI 4 M+/RTD	AI 6 M+/RTD
5	AI 0 M-/RTD	AI 2 M-/RTD	AI 4 M-/RTD	AI 6 M-/RTD
6	AI 0 I+/RTD	AI 2 I+/RTD	AI 4 I+/RTD	AI 6 I+/RTD
7	AI 0 I-/RTD	AI 2 I-/RTD	AI 4 I-/RTD	AI 6 I-/RTD
8	AI 1 M+/RTD	AI 3 M+/RTD	AI 5 M+/RTD	A7 M+ /RTD
9	AI 1 M-/RTD	AI 3 M-/RTD	AI 5 M-/RTD	AI 7 M-/RTD
10	AI 1 I+/RTD	AI 3 I+/RTD	AI 5 I+/RTD	AI 7 I+/RTD
11	AI 1 I-/RTD	AI 3 I-/RTD	AI 5 I-/RTD	AI 7 I-/RTD

说明

可以取消激活 RTD 的未使用通道。如果取消激活未使用的通道，不会出现任何错误。

RTD 模块需要使电流环不中断，以消除自动添加到未使用通道（未激活）的额外稳定时间。为保持一致性，RTD 模块应连接一个电阻（如 2 线制 RTD 连接）。

A.11.2.1 SM 1231 RTD 选型表

表格 A-182 RTD 模块支持的不同传感器的范围和精度

温度系数	RTD 类型	低于范围最小值 [1]	额定范围下限	额定范围上限	超出范围最大值 [2]	25 °C 时的额定范围精度	-20 °C 到 60 °C 时的额定范围精度
Pt 0.003850 ITS90 DIN EN 60751	Pt 100 气候型	-145.00 °C	-120.00 °C	145.00 °C	155.00 °C	±0.20 °C	±0.40 °C
	Pt 10	-243.0 °C	-200.0 °C	850.0 °C	1000.0 °C	±1.0 °C	±2.0 °C
	Pt 50	-243.0 °C	-200.0 °C	850.0 °C	1000.0 °C	±0.5 °C	±1.0 °C
	Pt 100						
	Pt 200						
	Pt 500						
	Pt 1000						
Pt 0.003902 Pt 0.003916 Pt 0.003920	Pt 100	-243.0 °C	-200.0 °C	850.0 °C	1000.0 °C	±0.5 °C	±1.0 °C
	Pt 200	-243.0 °C	-200.0 °C	850.0 °C	1000.0 °C	±0.5 °C	±1.0 °C
	Pt 500						
	Pt 1000						
Pt 0.003910	Pt 10	-273.2 °C	-240.0 °C	1100.0 °C	1295 °C	±1.0 °C	±2.0 °C
	Pt 50	-273.2 °C	-240.0 °C	1100.0 °C	1295 °C	±0.8 °C	±1.6 °C
	Pt 100						
	Pt 500						
Ni 0.006720 Ni 0.006180	Ni 100	-105.0 °C	-60.0 °C	250.0 °C	295.0 °C	±0.5 °C	±1.0 °C
	Ni 120						
	Ni 200						
	Ni 500						
	Ni 1000						
LG-Ni 0.005000	LG-Ni 1000	-105.0 °C	-60.0 °C	250.0 °C	295.0 °C	±0.5 °C	±1.0 °C
Ni 0.006170	Ni 100	-105.0 °C	-60.0 °C	180.0 °C	212.4 °C	±0.5 °C	±1.0 °C
Cu 0.004270	Cu 10	-240.0 °C	-200.0 °C	260.0 °C	312.0 °C	±1.0 °C	±2.0 °C

温度系数	RTD 类型	低于范围最小值 [1]	额定范围下限	额定范围上限	超出范围最大值 [2]	25 °C 时的额定范围精度	-20 °C 到 60 °C 时的额定范围精度
Cu 0.004260	Cu 10	-60.0 °C	-50.0 °C	200.0 °C	240.0 °C	±1.0 °C	±2.0 °C
	Cu 50	-60.0 °C	-50.0 °C	200.0 °C	240.0 °C	±0.6 °C	±1.2 °C
	Cu 100						
Cu 0.004280	Cu 10	-240.0 °C	-200.0 °C	200.0 °C	240.0 °C	±1.0 °C	±2.0 °C
	Cu 50	-240.0 °C	-200.0 °C	200.0 °C	240.0 °C	±0.7 °C	±1.4 °C
	Cu 100						

[1] “低于范围最小值”以下的 RTD 值报告为 -32768。

[2] 超出范围最大值以上的 RTD 值报告为 +32767。

表格 A-183 电阻

范围	低于范围最小值	额定范围下限	额定范围上限	超出范围最大值 [1]	25 °C 时的额定范围精度	-20 °C 到 60 °C 时的额定范围精度
150 Ω	不适用	0 (0 Ω)	27648 (150 Ω)	176.383 Ω	±0.05%	±0.1%
300 Ω	不适用	0 (0 Ω)	27648 (300 Ω)	352.767 Ω	±0.05%	±0.1%
600 Ω	不适用	0 (0 Ω)	27648 (600 Ω)	705.534 Ω	±0.05%	±0.1%

[1] 超出范围最大值以上的电阻值报告为 +32767。

说明

对于没有连接传感器的激活通道，模块会报告 32767。 如果还启用了开路检测，模块会使相应的红色 LED 闪烁。

对于其它值较低的电阻使用 500 Ω 和 1000 Ω RTD 范围时，误差可能增加到指定误差的两倍。

若使用 4 线制连接，对于 10 Ω RTD 范围，将得到最高精度。

2 线模式的连接线电阻会导致传感器读数误差，因此无法保证精度。

表格 A-184 RTD 模块的噪声消减和更新时间

抑制频率选择	积分时间	更新时间（秒）	
		4 通道模块	8 通道模块
400 Hz (2.5 ms)	10 ms [1]	4/2 线制： 0.142 3 线制： 0.285	4/2 线制： 0.285 3 线制： 0.525
60 Hz (16.6 ms)	16.67 ms	4/2 线制： 0.222 3 线制： 0.445	4/2 线制： 0.445 3 线制： 0.845
50 Hz (20 ms)	20 ms	4/2 线制： 0.262 3 线制： .505	4/2 线制： 0.524 3 线制： 1.015
10 Hz (100 ms)	100 ms	4/2 线制： 1.222 3 线制： 2.445	4/2 线制： 2.425 3 线制： 4.845

[1] 在选择 400 Hz 滤波器时，要维持模块的分辨率和精度，积分时间应为 10 ms。该滤波器还可抑制 100 Hz 和 200 Hz 的噪声。

说明

对模块上电后，模块将对模数转换器执行内部校准。 在此期间，模块将报告每个通道的值为 32767，直到相应通道出现有效值为止。 用户程序可能需要考虑这段初始化时间。 由于模块的组态可能改变初始化时长，因此，应验证组态中模块的行为。 如果需要，可以在用户程序中包含逻辑，以适应模块的初始化时间。

RTD 模拟值的表示

RTD 标准温度范围传感器数字化测量值的表示如下表所示。

表格 A-185 电阻温度计 PT 100、200、500、1000 和 PT 10、50、100、500 GOST (0.003850) 标准型的模拟值表示

Pt x00 标准型 (°C)（1 位数字 = 0.1 °C）	功能单元		Pt x00 标准型 (°F)（1 位数字 = 0.1 °F）	功能单元		范围
	十进制	十六进制		十进制	十六进制	
> 1000.0	32767	7FFF	> 1832.0	32767	7FFF	上溢
1000.0 : 850.1	10000 : 8501	2710 : 2135	1832.0 : 1562.1	18320 : 15621	4790 : 3D05	超出上限

Pt x00 标准型 (°C)（1 位数字 = 0.1 °C）	功能单元		Pt x00 标准型 (°F)（1 位数字 = 0.1 °F）	功能单元		范围
	十进制	十六进制		十进制	十六进制	
850.0 : -200.0	8500 : -2000	2134 : F830	1562.0 : -328.0	15620 : -3280	3D04 : F330	额定范围
-200.1 : -243.0	-2001 : -2430	F82F : F682	-328.1 : -405.4	-3281 : -4054	F32F : F02A	超出下限
< -243.0	-32768	8000	< -405.4	-32768	8000	下溢

A.12 工艺模块

A.12.1 SM 1278 4xIO-Link 主站 SM

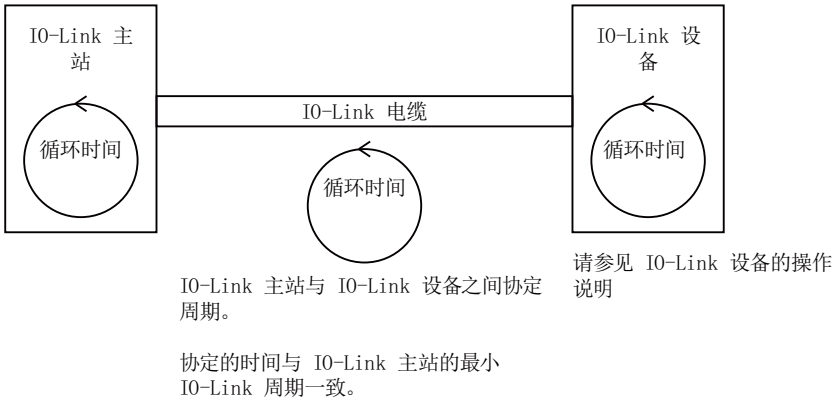
表格 A-186 常规规范

技术数据		SM 1278 4xIO-Link 主站信号模块
产品编号		6ES7278-4BD32-0XB0
尺寸 W x H x D (mm)		45 x 100 x 75
重量		150 g
常规信息		
	I&M 数据	√；IM0 到 IM3
供电电压		
	额定电压（直流）	24 V DC
	直流电压下限	19.2 V；如果使用 IO-Link，则为 20.5 V（主站中 IO-Link 设备的供电电压必须至少为 20 V）
	直流电压上限	28.8 V DC
	反极性保护	√
输入电流		
	电流消耗	65 mA；无负载
编码器电源		

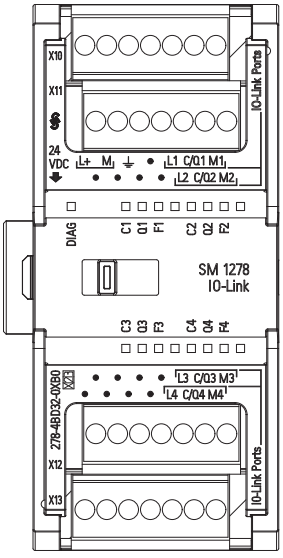
技术数据			SM 1278 4xIO-Link 主站信号模块
	输出点数		4
	输出电流，额定值		200 mA
功率损耗			
	典型功耗		1 W，不包括端口加载
数字量输入/输出			
	电缆长度（米）		最大 20 m（非屏蔽）
SDLC			
	电缆长度（米）		最大 20 m（非屏蔽）
IO-Link			
	端口数		4
	可同时控制的端口数		4
	IO-Link 协议 1.0		√
	IO-Link 协议 1.1		√
工作模式			
	IO-Link		√
	DI		√
	DQ		√；最大 100 mA
IO-Link 设备连接			
	端口类型 A		√
		传输率	4.8 kBd (COM1)
			38.4 kBd (COM2)
			230.4 kBd (COM3)
	最短周期时间		2 ms，动态，取决于用户数据长度
	过程数据大小，每个端口的输入量		最大 32 个字节
	过程数据大小，每个模块的输入量		32 个字节
	过程数据大小，每个端口的输出量		最大 32 个字节
	过程数据大小，每个模块的输出量		32 个字节
	设备参数的存储器大小		2 KB
	非屏蔽电缆的最大长度（米）		20 m
中断/诊断/状态信息			
	状态显示		√

技术数据			SM 1278 4xIO-Link 主站信号模块
中断			
	诊断中断		√；端口诊断仅适用于 IO-Link 模式
诊断报警			
	诊断		
		电源电压监视	√
		短路	√
诊断指示器 LED			
	电源电压监视		√；红色闪烁 DIAG LED
	通道状态显示		√；每个通道有一个绿色 LED，用于显示通道状态 Qn（SIO 模式）和端口状态 Cn（IO-Link 模式）
	通道诊断		√；红色 Fn LED
	模块诊断		有；绿色/红色 DIAG LED
电气隔离			
	电气隔离通道		
		通道之间	×
		通道和背板总线之间	√
绝缘			
	绝缘测试		707 V DC（型式测试）
环境条件			
	工作温度		
		最小值	-20 °C
		最大值	60 °C
		水平安装时的最低温度	-20 °C
		水平安装时的最高温度	60 °C
		垂直安装时的最低温度	-20 °C
		垂直安装时的最高温度	50 °C

响应时间总览



表格 A-187 SM 1278 IO-Link 主站接线图

SM 1278 IO-Link 主站 (6ES7278-4BD32-0XB0)	
	

表格 A-188 SM 1278 IO-Link 主站 (6ES7278-4BD32-0XB0) 的连接器引脚位置

针脚	X10	X11	X12	X13
1	L+ / 24 V DC	无连接	无连接	无连接
2	M / 24 V DC	无连接	无连接	无连接
3	功能性接地	无连接	无连接	无连接
4	无连接	无连接	无连接	无连接

针脚	X10	X11	X12	X13
5	L_1	L_2	L_3	L_4
6	C/Q_1	C/Q_2	C/Q_3	C/Q_4
7	M_1	M_2	M_3	M_4

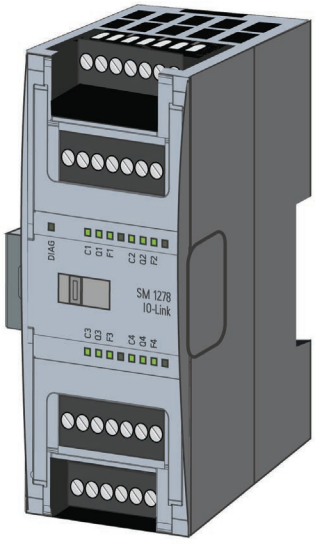
A.12.1.1 SM 1278 4xIO-Link 主站概述

SM 1278 4xIO-Link 主站是一个 4 端口模块，同时具有信号模块功能和通信模块功能。每个端口均可以 IO-Link 模式、单个 24 V DC 数字量输入或 24 V DC 数字量输出方式工作。

IO-Link 主站使用 STEP 7 S7-1200 控制器程序中的 IO_LINK_DEVICE 功能块 (FB) 对与 IO-Link 设备的非循环通信进行编程。IO_LINK_DEVICE FB 指示程序使用的 IO-Link 主站，以及主站用于进行数据交换的端口。

有关使用 IO-Link 库的详细信息，请访问西门子工业在线支持网站 (https://support.industry.siemens.com/cs/cn/zh)。在网站的搜索框中输入“IO-Link”可访问 IO-Link 产品及其使用的相关信息。

模块视图



属性

技术特性

- 符合 IO-Link 规范 V1.1 的 IO-Link 主站（有关详细信息，请参见 IO-Link 联盟网站 (http://io-link.com/en/index.php)）
- 具有四个端口（通道）的串行通信模块
- 数据传输速率 COM1 (4.8 kbaud)、COM2 (38.4 kbaud)、COM3 (230.4 kbaud)
- SIO 模式（标准 IO 模式）
- 最多四个 IO-Link 设备（3 线制连接）、四个标准执行器或标准编码器的连接
- 可按端口编程的诊断功能

支持的功能

- I&M（安装和维护）标识数据
- 固件更新
- 通过 S7-PCT 端口组态工具、STEP 7 Professional 以及 S7-1200 V4.0 或更高版本的 CPU. 对 IO-Link 进行参数分配。在 STEP 7 Professional V15（使用 V2.1 HSP 或更高版本）中，可以使用功能有限的 TIA Portal 完成 IO-Link 参数分配。
- 端口限定符信息 (PQI) 位
- 使用 IO-Link 库 FB 进行备份与恢复

IO-Link 是主站和设备之间的点对点连接。通过采用成熟的 3 线制技术的非屏蔽标准电缆，传统和智能传感器/执行器都可以用作 IO-Link 设备。IO-Link 与传统数字传感器和执行器向后兼容。电路状态和数据通道设计均基于可靠的 24 V DC 技术。

有关 SIMATIC-IO-Link 技术的更多信息，请参见 Siemens 工业在线支持网站 (http://support.automation.siemens.com)上的“IO-Link 系统功能手册”。

说明

IO-Link 参数数据

更换 SM 4xIO-Link 主站时，不会为其自动分配参数数据。

⚠ 小心

卸下和插入

如果在负载接通时插入 SM 4xIO-Link 主站，则可能导致设备出现危险情况。
从而可能导致 S7-1200 自动化系统发生物理损害。
仅可在负载关闭时移除或插入 SM 4xIO-Link 主站。

复位为出厂设置的影响

使用“复位为出厂设置”功能将通过 S7-PCT 执行的参数分配恢复为出厂状态。

“复位为出厂设置”后，SM 1278 4xIO-Link 模块的参数将按如下方式分配：

- 端口处于 DI 模式
- 端口映射到相对地址 0.0 至 0.3
- 禁用 PortQualifier
- 删除维护数据 1 至 3

说明

复位为出厂设置时，将删除设备参数并恢复为出厂状态。
如果移除 SM 1278 4xIO-Link 信号模块，请在入库前将其复位为出厂设置。

步骤

有关“复位为出厂设置”的执行步骤，请参见 S7-PCT“主站组态 > 命令”(Master Configuration > Commands) 选项卡中的在线帮助。

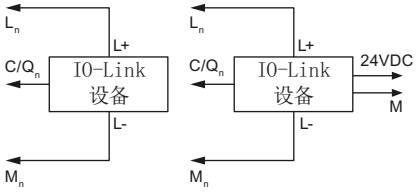
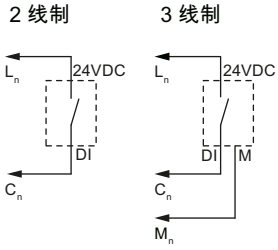
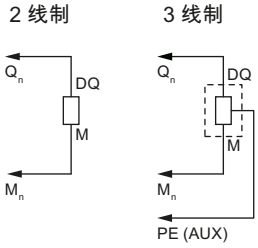
A.12.1.2 连接

有关引脚分配的详细信息，请参见表格，SM 1278 IO-Link 主站 (6ES 278-4BD32-0XB0) (页 1573)的连接器引脚位置。

下表显示了 SM 1278 4xIO-Link 主站的端子分配：

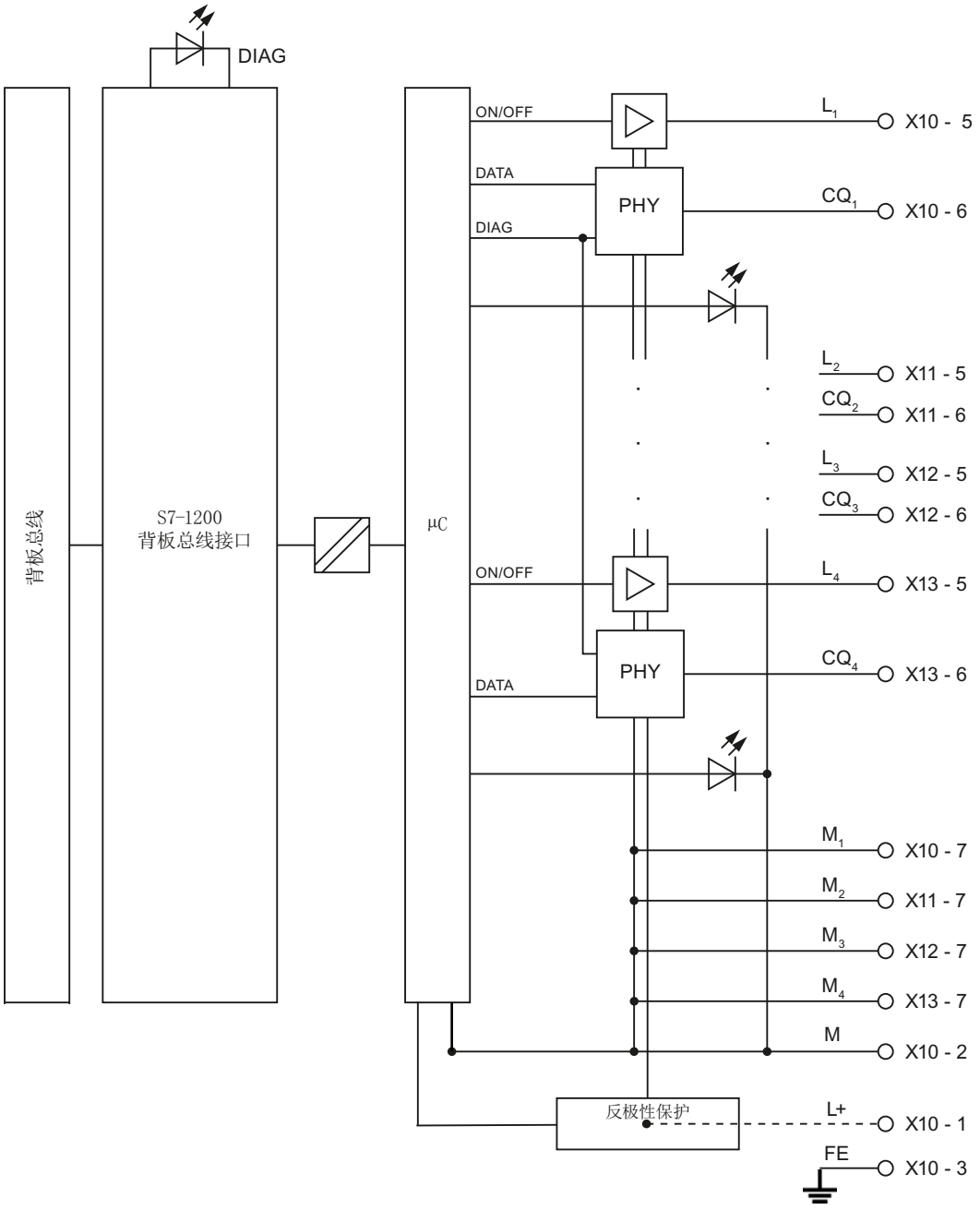
引脚	X10	X11	X12	X13	注意	BaseUnit
7	M_1	M_2	M_3	M_4	• M_n：地到从站 • C/Q_n：SDLC、DI 或 DQ • L_n：24 V DC 到从站 • M：接地 • L+：24 V DC 到主站 • RES：保留；可能不分配	A1
6	C/Q_1	C/Q_2	C/Q_3	C/Q_4		
5	L_1	L_2	L_3	L_4		
4	RES	RES	RES	RES		
3	⏚ （功能性接地）	RES	RES	RES		
2	M	RES	RES	RES		
1	L+	RES	RES	RES		

下表包含连接示例的示意图，其中 n = 端口号：

IO-Link 工作模式	工作模式 DI	工作模式 DQ
		

说明

所连接的传感器必须使用主站模块 L_n 连接提供的设备电源。


DIAG
背板总线
S7-1200
背板总线接口
μC
ON/OFF
DATA
DIAG
PHY
L1
X10 - 5
CQ1
X10 - 6
L2
X11 - 5
CQ2
X11 - 6
L3
X12 - 5
CQ3
X12 - 6
ON/OFF
DATA
PHY
L4
X13 - 5
CQ4
X13 - 6
M1
X10 - 7
M2
X11 - 7
M3
X12 - 7
M4
X13 - 7
M
X10 - 2
反极性保护
L+
X10 - 1
FE
X10 - 3

A.12.1.3 参数/地址空间

组态 SM 1278 4xIO-Link 主站

对于模块集成、参数分配和调试，需要 STEP 7 V13 或更高版本。对于某些功能，还需要 S7-PCT（端口组态工具）。

下表显示了需要 S7-PCT 时的条件：

	SM 1278 V2.0 4xIO-Link 主站	SM 1278 V2.1 4xIO-Link 主站
STEP7 V13.x 和 STEP V14.x	需要 S7-PCT	需要 S7-PCT
STEP 7 15.x	需要 S7-PCT	基本功能不需要 S7-PCT 高级功能需要 S7-PCT。

更多相关信息，请参见 SIMATIC IO-Link 系统手册 (https://support.industry.siemens.com/cs/cn/zh/view/65949252)。

下表显示了 SM 1278 4xIO-Link 主站的参数：

参数	取值范围	默认值	在 RUN 模式下进行组态	有效范围
诊断端口 1	• 禁用 • 启用	禁用	有	端口（通道）
诊断端口 2	• 禁用 • 启用	禁用	有	端口（通道）
诊断端口 3	• 禁用 • 启用	禁用	有	端口（通道）
诊断端口 4	• 禁用 • 启用	禁用	有	端口（通道）

为端口 1 到端口 4 参数启用诊断

该参数允许为四个 IO-Link 端口的特定端口启用诊断。

端口分配如下所示：

端口 1 -> 通道 1

端口 2 -> 通道 2

端口 3 -> 通道 3

端口 4 -> 通道 4

每种情况下，SM 4xIO-Link Master 输入和输出地址的最大大小为 32 字节。可以使用端口组态工具 S7-PCT 分配地址空间，自 V15 V2.1 HSP 起，还可使用 TIA Portal 硬件配置分配地址空间。

参数数据记录

用户程序中的参数分配

可以在运行过程中组态设备。

在运行过程中更改参数

模块参数包含在数据记录 128 中。可以通过 WRREC 指令将可修改的参数传输到模块中。

复位（循环上电）CPU 时，CPU 将覆盖参数化过程中由 WRREC 指令发送到模块的参数。

参数分配的指令

可以通过以下指令在用户程序中为 I/O 模块分配参数：

指令	应用
SFB 53 WRREC	将可更改参数传送到模块。

错误消息

出错时将报告以下返回值：

错误代码	含义
$80B1_H$	数据长度出错
$80E0_H$	标头信息出错
$80E1_H$	参数错误

数据记录结构

下表列出了 IO-Link 参数：

偏移	标签	类型	默认值	说明
0	版本	1 字节	0x02	表示 IO-Link 主站记录 0x02 的结构符合 IO-Link V1.1
1	参数长度	1 字节	0x02	参数长度（2 字节 + 2 个标头）
IO-Link 起始参数				
2	端口诊断（端口 1 至 n）	1 字节	0x00	激活端口 1 至 n 的诊断
3	IOL 属性	1 字节	0x00	模块属性

下表列出了数据记录版本：

位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
保留		主要版本 (00)		次要版本 (0010)			

以下表列出了数据记录端口诊断：

位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
保留				EN_Port 4	EN_Port 3	EN_Port 2	EN_Port 1

EN_Portx：

0 = 诊断已停用

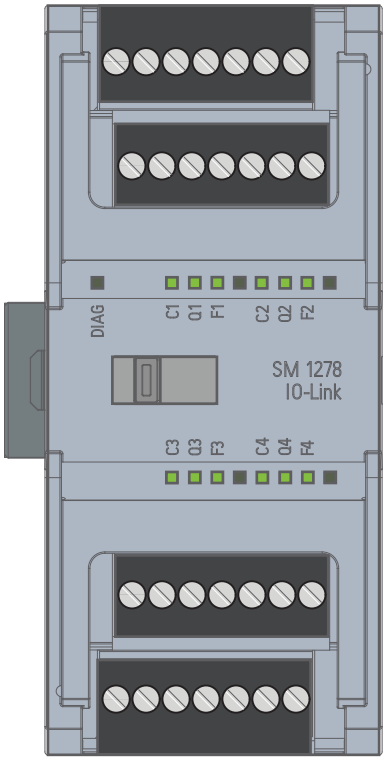
1 = 诊断已激活

下表列出了数据记录 IOL 属性：

位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
保留							

A.12.1.4 中断、错误和系统报警

LED 显示



LED 指示灯的含义

下表说明了状态和错误指示灯的含义。 有关诊断报警的补救措施，请参见“诊断报警”部分。

DIAG LED 指示灯

DIAG	含义
灭	S7-1200 的背板总线电源不正常
闪烁	模块未组态

DIAG	含义
■ 亮	模块已参数设置但没有进行模块诊断
☼ 闪烁	模块已参数设置且进行了模块诊断 或 L+ 电源未连接

LED 端口状态

对 IO-Link 端口模式下的 IO-Link 有效。

COM/1 ... COM/4	含义
□ 灭	端口已禁用
☼ 闪烁	端口已激活，设备未连接或 端口未连接到组态的设备
■ 亮	端口已激活，设备已连接

通道状态的 LED 指示灯

对 DI/Q 模式下的 IO-Link 端口有效。

DI/Q1 ... DI/Q4	含义
□ 灭	过程信号 = 0
■ 亮	过程信号 = 1

LED 端口错误

F1...F4	含义
□ 灭	无错误
■ 亮	错误

模块错误信息仅指示 IO-Link 模式下的诊断（模块状态）。

诊断报警	错误代码（十进制）	STATUS (W#16#...)	含义（IO-Link 错误代码）	IO-Link 主站	IO-Link 设备
短路	1	1804	IO-Link 设备的过程电缆短路	X	
		7710	IO 设备短路		X
欠压	2	5111 5112	供电电压过低		X
过压	3	5110	电源电压过高		X
过热	5	1805	主站温度过高	X	
		4000 4210	设备温度过高		X
断路	6	1800	• 未连接 IO-Link 任何设备 • 信号线与 IO-Link 设备之间存在断路 • IO-Link 设备由于其它错误无法进行通信	X	
上溢	7	8C10 8C20	超出过程变量范围		X
		8C20	超出测量范围		
下溢	8	8C30	过程变量范围太小		X
错误	9	---	此处未列出的所有与该 PROFIBUS DP 错误对应的 IO-Link 错误代码		X
参数分配错误	16	1882 1883	IO-Link 主站无法组态	X	
		1802	设备不正确		
		1886	存储错误		
		6320 6321 6350	设备组态错误		X
电源电压缺失	17	1806	设备 L+ 电源电压缺失	X	
		1807	设备 L+ 电源电压过低 (<20 V)		
保险丝故障	18	5101	设备上的保险丝发生故障		X

诊断报警	错误代码（十进制）	STATUS (W#16#...)	含义（IO-Link 错误代码）	IO-Link 主站	IO-Link 设备
安全关闭	25	1880	严重错误（必须更换主站）	X	
外部故障	26	1809 180A 180B 180C 180D	数据存储出错	X	
		1808	IO-Link 设备同时存在 6 个以上未决错误		

A.13 数字信号板 (SB)

A.13.1 SB 1221 200 kHz 数字量输入规范

表格 A-189 常规规范

技术数据	SB 1221 DI 4 x 24 V DC，200 kHz	SB 1221 DI 4 x 5 V DC，200 kHz
产品编号	6ES7221-3BD30-0XB0	6ES7221-3AD30-0XB0
尺寸 W x H x D (mm)	38 x 62 x 21	
重量	35 g	
功耗	1.5 W	1.0 W
电流消耗（SM 总线）	40 mA	
电流消耗 (24 V DC)	7 mA/输入 + 20 mA	15 mA/输入 + 15 mA

表格 A-190 数字量输入

技术数据	SB 1221 DI 4 x 24 V DC，200 kHz	SB 1221 DI 4 x 5 V DC，200 kHz
输入点数	4	
类型	源型	
额定电压	7 mA 时 24 V DC，额定值	15 mA 时 5 V DC，额定值
允许的连续电压	28.8 V DC	6 V DC
浪涌电压	35 V DC，持续 0.5 s	6 V

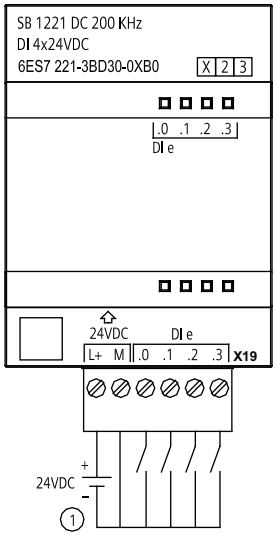
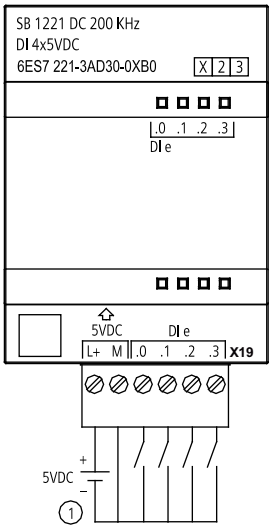
技术数据		SB 1221 DI 4 x 24 V DC，200 kHz	SB 1221 DI 4 x 5 V DC，200 kHz
逻辑 1 信号		0 V (10 mA) 至 L+ - 10 V (2.9 mA)	0 V (20 mA) 至 L+ - 2.0 V (5.1 mA)
逻辑 0 信号		L+ - 5 V (1.4 mA) 至 L+ (0 mA)	L+ - 1.0 V (2.2 mA) 至 L + (0 mA)
HSC 时钟输入频率（最大）		单相：200 kHz 正交相位：160 kHz	
隔离（现场侧与逻辑侧）		707 V DC（型式测试）	
隔离组		1	
滤波时间	us 设置	0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0	
	ms 设置	0.05、0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0	
同时接通的输入数		• 2（无相邻点），60 ℃（水平）或 50 ℃（垂直）时 • 4，55 ℃（水平）或 45 ℃（垂直）时	4
电缆长度（米）		50 m 屏蔽双绞线	

说明

开关频率高于 20 kHz 时，数字量输入接收方波甚为重要。请考虑采取以下措施提高提供给输入的信号质量：

- 使电缆尽可能短
- 将纯漏型激励器换成漏型和源型混合的激励器
- 使用质量更好的电缆
- 将电路/组件电压从 24 V 降为 5 V
- 在输入端连接外部负载

表格 A-191 200 kHz 数字量输入 SB 的接线图

SB 1221 DI 4 x 24 V DC，200 kHz (6ES7221-3BD30-0XB0)	SB 1221 DI 4 x 5 V DC，200 kHz (6ES7221-3AD30-0XB0)
	

① 仅支持源型输入

表格 A-192 SB 1221 DI 4 x 24 V DC，200 kHz (6ES7221-3BD30-0XB0) 的连接器引脚位置

针脚	X19
1	L+/24 V DC
2	M/24 V DC
3	DI e.0
4	DI e.1
5	DI e.2
6	DI e.3

表格 A-193 SB 1221 DI 4 x 5 V DC，200 kHz (6ES7221-3AD30-0XB0) 的连接器引脚位置

针脚	X19
1	L+/5 V DC
2	M/5 V DC
3	DI e.0
4	DI e.1

针脚	X19
5	DI e.2
6	DI e.3

A.13.2 SB 1222 200 kHz 数字量输出规范

表格 A-194 常规规范

技术数据	SB 1222 DQ 4 x 24 V DC，200 kHz	SB 1222 DQ 4 x 5 V DC，200 kHz
产品编号	6ES7222-1BD30-0XB0	6ES7222-1AD30-0XB0
尺寸 W x H x D (mm)	38 x 62 x 21	
重量	35 g	
功耗	0.5 W	
电流消耗（SM 总线）	35 mA	
电流消耗 (24 V DC)	15 mA	

表格 A-195 数字量输出

技术数据	SB 1222 DQ 4 x 24 V DC，200 kHz	SB 1222 DQ 4 x 5 V DC，200 kHz
输出点数	4	
输出类型	固态 - MOSFET 漏型和源型 [1]	
电压范围	20.4 到 28.8 V DC	4.25 到 6.0 V DC
最大电流时的逻辑 1 信号	L+ - 1.5 V	L+ - 0.7 V
最大电流时的逻辑 0 信号	1.0 V DC，最大值	0.2 V DC，最大值
电流（最大）	0.1 A	
灯负载	--	
通态触点电阻	最大 11 Ω	最大 7 Ω
断态电阻	最大 6 Ω	最大 0.2 Ω
每点的漏电流	--	
脉冲串输出频率	最大 200 kHz，最小 2 Hz	
浪涌电流	0.11 A	

技术数据	SB 1222 DQ 4 x 24 V DC，200 kHz	SB 1222 DQ 4 x 5 V DC，200 kHz
过载保护	×	
隔离（现场侧与逻辑侧）	707 V DC（型式测试）	
隔离组	1	
每个公共端的电流	0.4 A	
电感钳位电压	-	
开关延迟	上升沿 1.5 µs + 300 ns 下降沿 1.5 µs + 300 ns	上升沿 200 ns + 300 ns 下降沿 200 ns + 300 ns
RUN 到 STOP 时的行为	上一个值或替换值（默认值为 0）	
同时接通的输出数	• 2（无相邻点），60 ℃（水平）或 50 ℃（垂直）时 • 4，55 ℃（水平）或 45 ℃（垂直）时	4
电缆长度（米）	50 m 屏蔽双绞线	

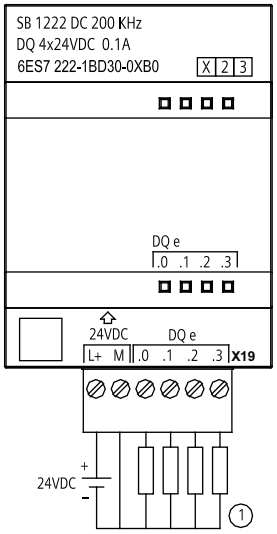
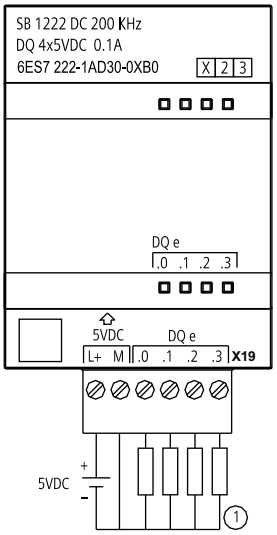
[1] 因为通过同一电路来支持漏型和源型配置，所以源型负载的激活状态与漏型负载的相反。源型输出表现为正逻辑（当负载有电流时，Q 位接通且 LED 亮起），而漏型输出表现为负逻辑（当负载有电流时，Q 位断开且 LED 熄灭）。如果插入模块且无用户程序，则此模块的默认值是 0 V，这意味着漏型负载将接通。

说明

开关频率高于 20 kHz 时，数字量输入接收方波甚为重要。请考虑采取以下措施提高提供给输入的信号质量：

- 使电缆尽可能短
- 将纯漏型激励器换成漏型和源型混合的激励器
- 使用质量更好的电缆
- 将电路/组件电压从 24 V 降为 5 V
- 在输入端连接外部负载

表格 A-196 200 kHz 数字量输出 SB 的接线图

SB 1222 DQ 4 x 24 V DC，200 kHz (6ES7222-1BD30-0XB0)	SB 1222 DQ 4 x 5 V DC，200 kHz (6ES7222-1AD30-0XB0)
	

① 对于源型输出，将“负载”连接到“-”（如图所示）。对于漏型输出，将“负载”连接到“+”。因为通过同一电路来支持漏型和源型配置，所以源型负载的激活状态与漏型负载的相反。源型输出表现为正逻辑（当负载有电流时，Q 位接通且 LED 亮起），而漏型输出表现为负逻辑（当负载有电流时，Q 位断开且 LED 熄灭）。如果插入模块且无用户程序，则此模块的默认值是 0 V，这意味着漏型负载将接通。

说明

确保 M 连接线安全接地。高速 DQ SB 若不接地，可能产生足以激活直流负载的漏电流。如果输出端用于临界直流负载应用，则务必多使用一根地线连接至 SB 进行额外的预防。

表格 A-197 SB 1222 DQ 4 x 24 V DC，200 kHz (6ES7222-1BD30-0XB0) 的连接器引脚位置

针脚	X19
1	L+ / 24 V DC
2	M / 24 V DC
3	DQ e.0
4	DQ e.1
5	DQ e.2
6	DQ e.3

表格 A-198 SB 1222 DQ 4 x 5 V DC，200 kHz (6ES7222-1AD30-0XB0) 的连接器引脚位置

针脚	X19
1	L+/5 V DC
2	M/5 V DC
3	DQ e.0
4	DQ e.1
5	DQ e.2
6	DQ e.3

A.13.3 SB 1223 200 kHz 数字量输入/输出规范

表格 A-199 常规规范

技术数据	SB 1223 DI 2 x 24 V DC/ DQ 2 x 24 V DC，200 kHz	SB 1223 DI 2 x 5 V DC/ DQ 2 x 5 V DC，200 kHz
产品编号	6ES7223-3BD30-0XB0	6ES7223-3AD30-0XB0
尺寸 W x H x D (mm)	38 x 62 x 21	
重量	35 g	
功耗	1.0 W	0.5 W
电流消耗（SM 总线）	35 mA	
电流消耗 (24 V DC)	7 mA/输入 + 30 mA	15 mA/输入 + 15 mA

表格 A-200 数字量输入

技术数据	SB 1223 DI 2 x 24 V DC/ DQ 2 x 24 V DC，200 kHz	SB 1223 DI 2 x 5 V DC/ DQ 2 x 5 V DC，200 kHz
输入点数	2	
类型	源型	
额定电压	7 mA 时 24 V DC，额定值	15 mA 时 5 V DC，额定值
允许的连续电压	28.8 V DC	6 V DC
浪涌电压	35 V DC，持续 0.5 s	6 V
逻辑 1 信号	0 V (10 mA) 至 L+ - 10 V (2.9 mA)	0 V (20 mA) 至 L+ - 2.0 V (5.1 mA)
逻辑 0 信号	L+ - 5 V (1.4 mA) 至 L+ (0 mA)	L+ - 1.0 V (2.2 mA) 至 L + (0 mA)

技术数据		SB 1223 DI 2 x 24 V DC/ DQ 2 x 24 V DC，200 kHz	SB 1223 DI 2 x 5 V DC/ DQ 2 x 5 V DC，200 kHz
HSC 时钟输入频率（最大）		单相：200 kHz 正交相位：160 kHz	
隔离（现场侧与逻辑侧）		707 V DC（型式测试）	
隔离组		1（与输出无隔离）	
滤波时间	us 设置	0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0	
	ms 设置	0.05、0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0	
同时接通的输入数		2	
电缆长度（米）		50 m 屏蔽双绞线	

表格 A-201 数字量输出

技术数据	SB 1223 DI 2 x 24 V DC/ DQ 2 x 24 V DC，200 kHz	SB 1223 DI 2 x 5 V DC/ DQ 2 x 5 V DC，200 kHz
输出点数	2	
输出类型	固态 - MOSFET 漏型和源型 [1]	
电压范围	20.4 到 28.8 V DC	4.25 到 6.0 V DC
额定值	24 V DC	5 V DC
最大电流时的逻辑 1 信号	L+ - 1.5 V	L+ - 0.7 V
最大电流时的逻辑 0 信号	1.0 V DC，最大值	0.2 V DC，最大值
电流（最大）	0.1 A	
灯负载	--	
通态触点电阻	最大 11 Ω	最大 7 Ω
断态电阻	最大 6 Ω	最大 0.2 Ω
每点的漏电流	--	
脉冲串输出频率	最大 200 kHz，最小 2 Hz	
浪涌电流	0.11 A	
过载保护	×	
隔离（现场侧与逻辑侧）	707 V DC（型式测试）	
隔离组	1（与输入无隔离）	
每个公共端的电流	0.2 A	
电感钳位电压	-	

技术数据	SB 1223 DI 2 x 24 V DC/ DQ 2 x 24 V DC，200 kHz	SB 1223 DI 2 x 5 V DC/ DQ 2 x 5 V DC，200 kHz
开关延迟	上升沿 1.5 µs + 300 ns 下降沿 1.5 µs + 300 ns	上升沿 200 ns + 300 ns 下降沿 200 ns + 300 ns
RUN 到 STOP 时的行为	上一个值或替换值（默认值为 0）	
同时接通的输出数	2	
电缆长度（米）	50 m 屏蔽双绞线	

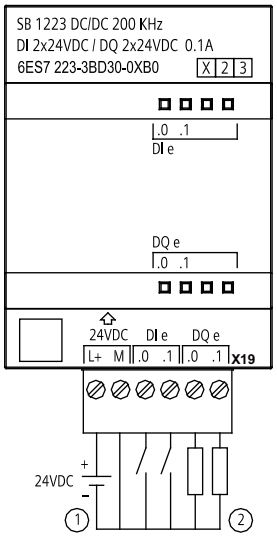
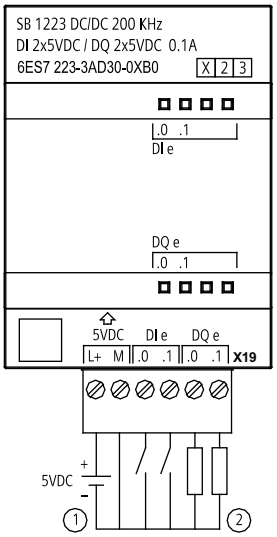
[1] 因为通过同一电路来支持漏型和源型配置，所以源型负载的激活状态与漏型负载的相反。源型输出表现为正逻辑（当负载有电流时，Q 位接通且 LED 亮起），而漏型输出表现为负逻辑（当负载有电流时，Q 位断开且 LED 熄灭）。如果插入模块且无用户程序，则此模块的默认值是 0 V，这意味着漏型负载将接通。

说明

开关频率高于 20 kHz 时，数字量输入接收方波甚为重要。请考虑采取以下措施提高提供给输入的信号质量：

- 使电缆尽可能短
- 将纯漏型激励器换成漏型和源型混合的激励器
- 使用质量更好的电缆
- 将电路/组件电压从 24 V 降为 5 V
- 在输入端连接外部负载

表格 A-202 200 kHz 数字量输入/输出 SB 的接线图

SB 1223 DI 2 x 24 V DC/DQ 2 x 24 V DC，200 kHz (6ES7223-3BD30-0XB0)	SB 1223 DI 2 x 5 V DC / DQ 2 x 5 V DC，200 kHz (6ES7223-3AD30-0XB0)
	

① 仅支持源型输入

② 对于源型输出，将“负载”连接到“-”（如图所示）。对于漏型输出，将“负载”连接到“+”。[1] 因为通过同一电路来支持漏型和源型配置，所以源型负载的激活状态与漏型负载的相反。源型输出表现为正逻辑（当负载有电流时，Q 位接通且 LED 亮起），而漏型输出表现为负逻辑（当负载有电流时，Q 位断开且 LED 熄灭）。如果插入模块且无用户程序，则此模块的默认值是 0 V，这意味着漏型负载将接通。

说明

确保 M 连接线安全接地。高速 DQ SB 若不接地，可能产生足以激活直流负载的漏电流。如果输出端用于临界直流负载应用，则务必多使用一根地线连接至 SB 进行额外的预防。

表格 A-203 SB 1223 DI 2 x 24 V DC/DQ 2 x 24 V DC，200 kHz (6ES7223-3BD30-0XB0) 的连接器引脚位置

针脚	X19
1	L+/24 V DC
2	M/24 V DC
3	DI e.0
4	DI e.1

针脚	X19
5	DQ e.0
6	DQ e.1

表格 A-204 SB 1223 DI 2 x 5 V DC / DQ 2 x 5 V DC，200 kHz (6ES7223-3AD30-0XB0) 的连接器引脚位置

针脚	X19
1	L+/5 V DC
2	M/5 V DC
3	DI e.0
4	DI e.1
5	DQ e.0
6	DQ e.1

A.13.4 SB 1223 2 X 24 V DC 输入/2 X 24 V DC 输出规格

表格 A-205 常规规范

技术数据	SB 1223 DI 2 x 24 V DC，DQ 2 x 24 V DC
订货号	6ES7223-0BD30-0XB0
尺寸 W x H x D (mm)	38 x 62 x 21
重量	40 g
功耗	1.0 W
电流消耗（SM 总线）	50 mA
电流消耗 (24 V DC)	所用的每点输入 4 mA

表格 A-206 数字量输入

技术数据	SB 1223 DI 2 x 24 V DC，DQ 2 x 24 V DC
输入点数	2
类型	IEC 1 类漏型
额定电压	4 mA 时 24 V DC，额定值

技术数据		SB 1223 DI 2 x 24 V DC，DQ 2 x 24 V DC
允许的连续电压		30 V DC，最大值
浪涌电压		35 V DC，持续 0.5 s
逻辑 1 信号（最小）		2.5 mA 时 15 V DC
逻辑 0 信号（最大）		1 mA 时 5 V DC
HSC 时钟输入频率（最大）		单相：30 kHz（15 到 26 V DC） 正交相位：20 kHz（15 到 26 V DC）
隔离（现场侧与逻辑侧）		707 V DC（型式测试）
隔离组		1
滤波时间	us 设置	0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0
	ms 设置	0.05、0.1、0.2、0.4、0.8、1.6、3.2、6.4、10.0、12.8、20.0
同时接通的输入数		2
电缆长度（米）		500（屏蔽）；300（非屏蔽）

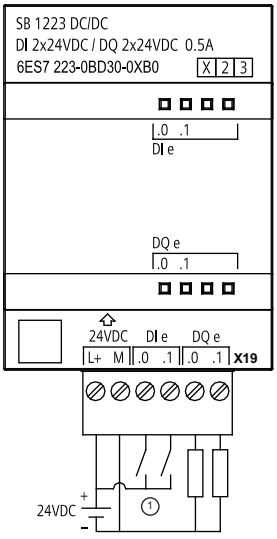
表格 A-207 数字量输出

技术数据	SB 1223 DI 2 x 24 V DC，DQ 2 x 24 V DC
输出点数	2
输出类型	固态 - MOSFET（源型）
电压范围	20.4 到 28.8 V DC
最大电流时的逻辑 1 信号	20 V DC 最小
具有 10 KΩ 负载时的逻辑 0 信号	0.1 V DC 最大
电流（最大）	0.5 A
灯负载	5 W
通态触点电阻	最大 0.6 Ω
每点的漏电流	最大 10 μA
脉冲串输出 (PTO) 频率	最大 20 kHz，最小 2 Hz [1]
浪涌电流	5 A，最长持续 100 ms
过载保护	-
隔离（现场侧与逻辑侧）	707 V DC（型式测试）
隔离组	1

技术数据	SB 1223 DI 2 x 24 V DC，DQ 2 x 24 V DC
每个公共端的电流	1 A
电感钳位电压	L+ - 48 V，1 W 损耗
开关延迟	断开到接通最长为 2 μs 接通到断开最长为 10 μs
RUN 到 STOP 时的行为	上一个值或替换值（默认值为 0）
同时接通的输出数	2
电缆长度（米）	500 m（屏蔽）；150 m（非屏蔽）

[1] 根据所使用的脉冲接收器和电缆的情况，附加的负载电阻（至少为额定电流的 10%）可能改进脉冲信号质量和抗扰度。

表格 A-208 数字量输入/输出 SB 的接线图

SB 1223 DI 2 x 24 V DC，DQ 2 x 24 V DC (6ES7223-0BD30-0XB0)	
	

① 支持漏型输入

表格 A-209 SB 1223 DI 2 x 24 V DC，DQ 2 x 24 V DC (6ES7223-0BD30-0XB0) 的连接器引脚位置

针脚	X19
1	L+/24 V DC
2	M/24 V DC
3	DI e.0

针脚	X19
4	DI e.1
5	DQ e.0
6	DQ e.1

A.14 模拟信号板 (SB)

A.14.1 SB 1231 1 路模拟量输入规范

说明

要使用此 SB，CPU 固件必须为 V2.0 或更高版本。

表格 A-210 常规规范

技术数据	SB 1231 AI 1 x 12 位
订货号	6ES7231-4HA30-0XB0
尺寸 W x H x D (mm)	38 x 62 x 21
重量	35 g
功耗	0.4 W
电流消耗（SM 总线）	55 mA
电流消耗 (24 V DC)	无

表格 A-211 模拟量输入

技术数据	SB 1231 AI 1 x 12 位
输入点数	1
类型	电压或电流（差动）
范围	±10 V、±5 V、±2.5 或 0 到 20 mA
分辨率	11 位 + 符号位
满量程范围（数据字）	-27648 到 27648

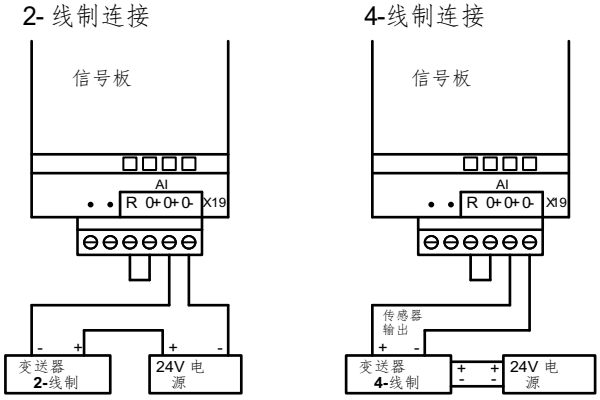
技术数据	SB 1231 AI 1 x 12 位
超出/低于范围（数据字）	电压：32511 到 27649/-27649 到 -32512 电流：32511 到 27649/0 到 -4864 （请参见模拟量输入的电压表示法以及模拟量输入的电流表示法 (页 1607)。）
上溢/下溢（数据字）	电压：32767 到 32512/-32513 到 -32768 电流：32767 到 32512/-4865 到 -32768 （请参见模拟量输入的电压表示法以及模拟量输入的电流表示法 (页 1607)。）
最大耐压/耐流	±35 V/±40 mA
平滑化	无、弱、中或强（请参见模拟量输入的响应时间 (页 1606)以了解阶跃响应时间。）
噪声抑制	400、60、50 或 10 Hz（请参考模拟输入的响应时间 (页 1606)以了解采样速率。）
精度（25 °C/-20 到 60 °C）	满量程的 ±0.3%/±0.6%
输入阻抗	电压：150 kΩ；电流：250 Ω
测量原理	实际值转换
共模抑制	40 dB，DC 到 60 Hz
工作信号范围	信号加共模电压必须小于 +35 V 且大于 -35 V
隔离（现场侧与逻辑侧）	无
电缆长度（米）	100 m，屏蔽双绞线

表格 A-212 诊断

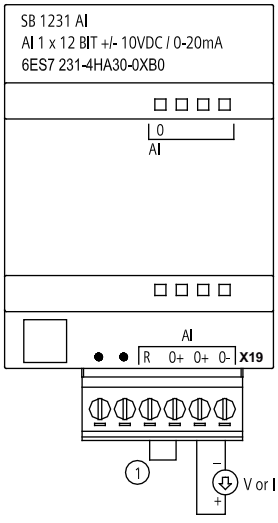
技术数据	SB 1231 AI 1 x 12 位
上溢/下溢	√
24 V DC 低压	-

SB 1231 接线电流变送器

接线电流变送器可用作 2 线制变送器和 4 线制变送器，如下图所示。



表格 A-213 模拟量输入 SB 的接线图

SB 1231 AI x 12 位 (6ES7231-4HA30-0XB0)	
	① 如果要施加电流，请连接“R”和“0+”。 注：连接器必须镀金。有关订货号，请参见附录 C“备件”。

表格 A-214 SB 1231 AI x 12 位 (6ES7231-4HA30-0XB0) 的连接器引脚位置

引脚	X19（镀金）
1	无连接
2	无连接
3	AI R
4	AI 0+

引脚	X19（镀金）
5	AI 0+
6	AI 0-

A.14.2 SB 1232 1 路模拟量输出规范

表格 A-215 常规规范

技术数据	SB 1232 AQ 1 x 12 位
订货号	6ES7232-4HA30-0XB0
尺寸 W x H x D (mm)	38 x 62 x 21
重量	40 g
功耗	1.5 W
电流消耗（SM 总线）	15 mA
电流消耗 (24 V DC)	40 mA（无负载）

表格 A-216 模拟量输出

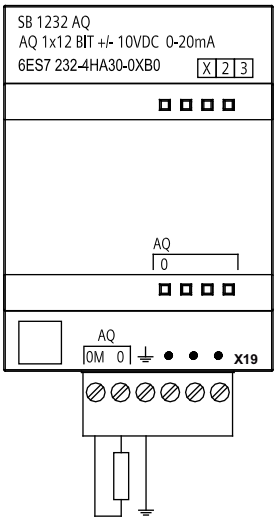
技术数据	SB 1232 AQ 1 x 12 位
输出点数	1
类型	电压或电流
范围	±10 V 或 0 到 20 mA
分辨率	电压：12 位 电流：11 位
满量程范围（数据字） 请参见电压和电流 (页 1608)的输出范围。	电压：-27648 到 27648 电流：0 到 27648
精度（25 °C/-20 到 60 °C）	满量程的 ±0.5%/±1%
稳定时间（新值的 95%）	电压：300 µs (R)，750 µs (1 uF) 电流：600 µs (1 mH)，2 ms (10 mH)
负载阻抗	电压 ：≥ 1000 Ω 电流：≤ 600 Ω
RUN 到 STOP 时的行为	上一个值或替换值（默认值为 0）

技术数据	SB 1232 AQ 1 x 12 位
隔离（现场侧与逻辑侧）	无
电缆长度（米）	100 m，屏蔽双绞线

表格 A-217 诊断

技术数据	SB 1232 AQ 1 x 12 位
上溢/下溢	√
对地短路（仅限电压模式）	√
断路（仅限电流模式）	√

表格 A-218 SB 1232 AQ 1 x 12 位的接线图

SB 1232 AQ 1 x 12 位 (6ES7232-4HA30-0XB0)	
	
注：连接器必须镀金。有关订货号，请参见附录 C“备件”。	

表格 A-219 SB 1232 AQ 1 x 12 位 (6ES7232-4HA30-0XB0) 的连接器引脚位置

引脚	X19（镀金）
1	AQ 0M
2	AQ 0
3	功能性接地
4	无连接

引脚	X19（镀金）
5	无连接
6	无连接

A.14.3 模拟量输入和输出的测量范围

A.14.3.1 模拟量输入的阶跃响应

表格 A-220 阶跃响应 (ms)，0 V 到 10 V（在 95% 处测得）

平滑化选项（采样平均）	积分时间选项			
	400 Hz (2.5 ms)	60 Hz (16.6 ms)	50 Hz (20 ms)	10 Hz (100 ms)
无（1 个周期）： 不求平均值	4.5 ms	18.7 ms	22.0 ms	102 ms
弱（4 个周期）： 4 次采样	10.6 ms	59.3 ms	70.8 ms	346 ms
中（16 个周期）： 16 次采样	33.0 ms	208 ms	250 ms	1240 ms
强（32 个周期）： 32 次采样	63.0 ms	408 ms	490 ms	2440 ms
采样时间	**0.156 ms**	**1.042 ms**	**1.250 ms**	**6.250 ms**

A.14.3.2 模拟量输入的采样时间和更新时间

表格 A-221 采样时间和更新时间

选项	采样时间	SB 更新时间
400 Hz (2.5 ms)	0.156 ms	0.156 ms
60 Hz (16.6 ms)	1.042 ms	1.042 ms
50 Hz (20 ms)	1.250 ms	1.25 ms
10 Hz (100 ms)	6.250 ms	6.25 ms

A.14.3.3 模拟量输入的电压和电流测量范围（SB 和 SM）

表格 A-222 模拟量输入的电压表示法（SB 和 SM）

系统		电压测量范围				
十进制	十六进制	±10 V	±5 V	±2.5 V	±1.25 V	
32767	7FFF[1]	11.851 V	5.926 V	2.963 V	1.481 V	上溢
32512	7F00					
32511	7EFF	11.759 V	5.879 V	2.940 V	1.470 V	过冲范围
27649	6C01					
27648	6C00	10 V	5 V	2.5 V	1.250 V	额定范围
20736	5100	7.5 V	3.75 V	1.875 V	0.938 V	
1	1	361.7 µV	180.8 µV	90.4 µV	45.2 µV	
0	0	0 V	0 V	0 V	0 V	
-1	FFFF					
-20736	AF00	-7.5 V	-3.75 V	-1.875 V	-0.938 V	
-27648	9400	-10 V	-5 V	-2.5 V	-1.250 V	
-27649	93FF					下冲范围
-32512	8100	-11.759 V	-5.879 V	-2.940 V	-1.470 V	
-32513	80FF					下溢
-32768	8000	-11.851 V	-5.926 V	-2.963 V	-1.481 V	

[1] 返回 7FFF 可能由以下原因之一所致：上溢（如上表所述）、有效值可用前（例如上电时立即返回）或者检测到断路。

表格 A-223 模拟量输入的电流表示法（SB 和 SM）

系统		电流测量范围		
十进制	十六进制	0 mA 到 20 mA	4 mA 到 20 mA	
32767	7FFF	> 23.52 mA	> 22.81 mA	上溢
32511	7EFF	23.52 mA	22.81 mA	过冲范围
27649	6C01			
27648	6C00	20 mA	20 mA	额定范围
20736	5100	15 mA	16 mA	
1	1	723.4 nA	4 mA + 578.7 nA	
0	0	0 mA	4 mA	

系统		电流测量范围		
十进制	十六进制	0 mA 到 20 mA	4 mA 到 20 mA	
-1	FFFF			下冲范围
-4864	ED00	-3.52 mA	1.185 mA	
32767[1]	7FFF		< 1.185 mA	断路（4 至 20 mA）
-32768	8000	< -3.52 mA		下溢（0 到 20 mA）

[1] 无论断路报警的状态如何，始终会返回断路值 32767 (16#7FFF)。

A.14.3.4 模拟量输出的电压和电流测量范围（SB 和 SM）

表格 A-224 模拟量输出的电压表示法（SB 和 SM）

系统		电压输出范围	
十进制	十六进制	±10 V	
32767	7FFF	请参见注 1	上溢
32512	7F00	请参见注 1	
32511	7EFF	11.76 V	过冲范围
27649	6C01		
27648	6C00	10 V	额定范围
20736	5100	7.5 V	
1	1	361.7 µ V	
0	0	0 V	
-1	FFFF	-361.7 µ V	
-20736	AF00	-7.5 V	
-27648	9400	-10 V	
-27649	93FF		下冲范围
-32512	8100	-11.76 V	
-32513	80FF	请参见注 1	下溢
-32768	8000	请参见注 1	

[1] 在上溢或下溢情况下，模拟量输出将采用 STOP 模式的替代值。

表格 A-225 模拟量输出的电流表示法（SB 和 SM）

系统		当前输出范围		
十进制	十六进制	0 mA 到 20 mA	4 mA 到 20 mA	
32767	7FFF	请参见注 1	请参见注 1	上溢
32512	7F00	请参见注 1	请参见注 1	
32511	7EFF	23.52 mA	22.81 mA	过冲范围
27649	6C01			
27648	6C00	20 mA	20 mA	额定范围
20736	5100	15 mA	16 mA	
1	1	723.4 nA	4 mA + 578.7 nA	
0	0	0 mA	4mA	
-1	FFFF		4 mA 到 578.7 nA	下冲范围
-6912	E500		0 mA	
-6913	E4FF			不可能。 输出值限制在 0 mA。
-32512	8100			
-32513	80FF	请参见注 1	请参见注 1	下溢
-32768	8000	请参见注 1	请参见注 1	

[1] 在上溢或下溢情况下，模拟量输出将采用 STOP 模式的替代值。

A.14.4 热电偶信号板 (SB)

A.14.4.1 SB 1231 1 路热电偶模拟量输入规范

说明

要使用此 SB，CPU 固件必须为 V2.0 或更高版本。

表格 A-226 常规规范

技术数据	SB 1231 AI 1 x 16 位热电偶
订货号	6ES7231-5QA30-0XB0
尺寸 W x H x D (mm)	38 x 62 x 21
重量	35 g
功耗	0.5 W
电流消耗（SM 总线）	5 mA
电流消耗 (24 V DC)	20 mA

表格 A-227 模拟量输入

技术数据		SB 1231 AI 1x16 位热电偶
输入点数		1
类型		浮动 TC 和 mV
范围 ● 额定范围（数据字） ● 过量程/欠量程（数据字） ● 上溢/下溢（数据字）		请参见热电偶滤波器选型表 (页 1611)。
分辨率	温度	0.1° C/0.1° F
	电压	15 位 + 符号
最大耐压		±35 V
噪声抑制		对于所选滤波器设置 （10 Hz、50 Hz、60 Hz、400 Hz）为 85 dB
共模抑制		120 VAC 时大于 120 dB
阻抗		≥ 10 M Ω
精度		请参见热电偶选型表 (页 1611)。
可重复性		±0.05% FS
测量原理		积分型
模块更新时间		请参见热电偶滤波器选型表 (页 1611)。
冷端误差		±1.5° C
隔离（现场侧与逻辑侧）		707 V DC（型式测试）
电缆长度（米）		到传感器最长为 100 m
导线电阻		最大 100 Ω

表格 A-228 诊断

技术数据	SB 1231 AI 1 x 16 位热电偶
上溢/下溢 [1]	√
断路 [2, 3]	√

[1] 上溢和下溢诊断报警信息将以模拟数据值的形式报告，即使在模块组态中禁用这些报警也会如此。

[2] 如果断线报警已禁用，但传感器接线存在开路情况，则模块可能会报告随机值。

[3] 模块每 6 秒执行一次断路测试，这样每 6 秒会针对每个使能通道将更新时间延长 9 ms。

SM 1231 热电偶 (TC) 模拟量信号模块可测量连接到模块输入的电压值。

SB 1231 热电偶模拟信号板可测量连接到信号板输入的电压值。温度测量类型可以是“热电偶”或“电压”类型。

- “热电偶”：将度数乘 10 得到该值（例如，25.3 度将报告为十进制数 253）。
- “电压”：额定范围的满量程值将是十进制数 27648。

A.14.4.2 热电偶的基本操作

两种不同的金属彼此之间存在电气连接时，便会形成热电偶。 热电偶产生的电压与结点温度成正比。 电压很小；一微伏能表示很多度。 测量热电偶产生的电压，对额外的结点进行补偿，然后将测量结果线性化，这些是使用热电偶测量温度的基础。

将热电偶连接到 SM 1231 热电偶模块时，两条不同的金属线需连接到模块的信号连接器上。这两条不同的金属线互相连接的位置即形成了传感器热电偶。

在这两条不同的金属线与信号连接器相连的位置，构成了另外二个热电偶。 连接器温度会引起一定的电压，该电压将添加到传感器热电偶产生的电压中。 如果不对该电压进行修正，结果报告的温度将偏离传感器温度。

冷端补偿便是用于对连接器热电偶进行补偿。 热电偶表是基于参比端温度（通常是零摄氏度）得来的。 冷端补偿用于将连接器温度修正为零摄氏度。 冷端补偿可消除连接器热电偶增加的电压。 模块的温度在内部测量，然后转换为数值并添加到传感器换算中。 之后是使用热电偶表对修正后的传感器换算值进行线性化。

为使冷端补偿取得最佳效果，必须将热电偶模块安装在温度稳定的环境中。 符合模块规范的模块环境温度的缓慢变化（低于 0.1 ℃/分钟）能够被正确补偿。 穿过模块的空气流动也会引起冷端补偿误差。

如果需要更佳的冷端误差补偿效果，则可使用外部 iso 热端子块。 热电偶模块可以使用 0 ℃ 基准值或 50 ℃ 基准值端子块。

SB 1231 热电偶选型表

下表给出了 SB 1231 热电偶信号板支持的不同热电偶类型对应的测量范围和精度。

表格 A-229 热电偶选型表

类型	欠范围最小值 [1]	额定范围下限	额定范围上限	过范围最大值 [2]	25 °C 时的正常范围 [3, 4] 精度	-20 °C 到 60 °C 时的正常范围 [1, 2] 精度
J	-210.0 ℃	-150.0 ℃	1200.0 ℃	1450.0 ℃	±0.3 °C	±0.6 °C
	-346.0 °F	-238.0 °F	2192.0 °F	2642.0 °F	±0.5 °F	±1.1 °F
K	-270.0 ℃	-200.0 ℃	1372.0 ℃	1622.0 ℃	±0.4 °C	±1.0 °C
	-454.0 °F	-328.0 °F	2501.6 °F	2951.6 °F	±0.7 °F	±1.8 °F
T	-270.0 ℃	-200.0 ℃	400.0 ℃	540.0 ℃	±0.5 °C	±1.0 °C
	-454.0 °F	-328.0 °F	752.0 °F	1004.0 °F	±0.9 °F	±1.8 °F
E	-270.0 ℃	-200.0 ℃	1000.0 ℃	1200.0 ℃	±0.3 °C	±0.6 °C
	-454.0 °F	-328.0 °F	1832.0 °F	2192.0 °F	±0.5 °F	±1.1 °F
R & S	-50.0 ℃	100.0 ℃	1768.0 ℃	2019.0 ℃	±1.0 °C	±2.5 °C
	-58.0 °C	212.0 °F	3214.4 °F	3276.6 °F [5]	±1.8 °F	±4.5 °F
B	0.0 ℃	200.0 ℃	800.0 ℃	--	±2.0 °C	±2.5 °C
	32.0 °F	392.0 °F	1472.0 °F	--	±3.6 °F	±4.5 °F
	--	800.0 ℃	1820.0 ℃	1820.0 ℃	±1.0 °C	±2.3 °C
	--	1472.0 °F	3276.6 °F [5]	3276.6 °F [5]	±1.8 °F	±4.1 °F

类型	欠范围最小值 [1]	额定范围下限	额定范围上限	过范围最大值 [2]	25 °C 时的正常范围 [3, 4] 精度	-20 °C 到 60 °C 时的正常范围 [1, 2] 精度
N	-270.0 ℃	-200.0 °C	1300.0 ℃	1550.0 ℃	±1.0 °C	±1.6 °C
	-454.0 °F	-328.0 °F	2372.0 °F	2822.0 °F	±1.8 °F	±2.9 °F
C	0.0 ℃	100.0 ℃	2315.0 ℃	2500.0 ℃	±0.7 °C	±2.7 °C
	32.0 °F	212.0 °F	3276.6 °F [5]	3276.6 °F [5]	±1.3 °F	±4.9 °F
TXK/ XK(L)	-200.0 ℃	-150.0 ℃	800.0 ℃	1050.0 ℃	±0.6 °C	±1.2 °C
	-328.0 °F	302.0 °F	1472.0 °F	1922.0 °F	±1.1 °F	±2.2 °F
电压	-32512	-27648 -80mV	27648 80mV	32511	±0.05%	±0.1%

[1] “低于范围最小值”以下的热电偶值报告为 -32768。

[2] “超过范围最大值”以上的热电偶值报告为 32767。

[3] 所有范围的内部冷端误差均为 ±1.5 °C。该误差已包括到本表的误差中。模块至少需要 30 分钟的预热时间才能满足本规范要求。

[4] 存在 970 MHz 至 990 MHz 的辐射射频时，SM 1231 AI 4 x 16 位 TC 的精度可能降低。

[5] 下限 3276.6 °F，含 °F 报告

表格 A-230 SB 1231 热电偶的滤波器选型表

抑制频率 (Hz)	积分时间 (ms)	信号板更新时间（秒）
10	100	0.306
50	20	0.066
60	16.67	0.056
400[1]	10	0.036

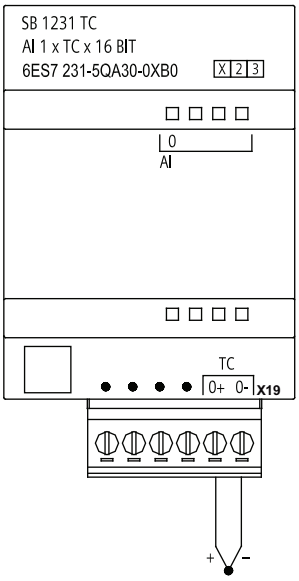
[1] 在选择 400 Hz 抑制频率时，要维持模块的分辨率和精度，积分时间应为 10 ms。该选择还可抑制 100 Hz 和 200 Hz 的噪声。

测量热电偶时建议使用 100 ms 的积分时间。 使用更小的积分时间将增大温度读数的重复性误差。

说明

对模块上电后，模块将对模数转换器执行内部校准。 在此期间，模块将报告每个通道的值为 32767，直到相应通道出现有效值为止。 用户程序可能需要考虑这段初始化时间。

表格 A-231 SB 1231 AI 1 x 16 热电偶的接线图

SB 1231 AI 1 x 16 位热电偶 (6ES7231-5QA30-0XB0)	
	
注意：连接器必须镀金。有关订货号，请参见附录 C“备件”。	

表格 A-232 SB 1231 AI 1 x 16 位热电偶 (6ES7231-5QA30-0XB0) 的连接器引脚位置

引脚	X19（镀金）
1	无连接
2	无连接
3	无连接
4	无连接

引脚	X19（镀金）
5	AI 0-/TC
6	AI 0+/TC

A.14.5 RTD 信号板 (SB)

A.14.5.1 SB 1231 1 路模拟量 RTD 输入的规范

说明

要使用此 SB，CPU 固件必须为 V2.0 或更高版本。

表格 A-233 常规规范

技术数据	SB 1231 AI 1 x 16 位 RTD
产品编号	6ES7231-5PA30-0XB0
尺寸 W x H x D (mm)	38 x 62 x 2
重量	35 g
功耗	0.7 W
电流消耗（SM 总线）	5 mA
电流消耗 (24 V DC)	25 mA

表格 A-234 模拟量输入

技术数据		SB 1231 AI 1 x 16 位 RTD
输入点数		1
类型		模块参考 RTD 和 Ω
范围 ● 额定范围（数据字） ● 过量程/欠量程（数据字） ● 上溢/下溢（数据字）		请参见选型表 (页 1618)。
分辨率	温度	0.1 °C/0.1 °F
	电压	15 位 + 符号

技术数据	SB 1231 AI 1 x 16 位 RTD
最大耐压	±35 V
噪声抑制	85 dB（10 Hz、50 Hz、60 Hz、400 Hz）
共模抑制	> 120 dB
阻抗	≥ 10 MΩ
精度	请参见选型表 (页 1618)。
可重复性	±0.05% FS
最大传感器功耗	0.5 m W
测量原理	积分型
模块更新时间	请参见选型表 (页 1618)。
隔离（现场侧与逻辑侧）	707 V DC（型式测试）
电缆长度（米）	到传感器最长为 100 米
导线电阻	20 Ω，对于 10 Ω RTD，最大为 2.7 Ω

表格 A-235 诊断

技术数据	SB 1231 AI 1 x 16 位 RTD
上溢/下溢 [1, 2]	√
断线 [3]	√

[1] 上溢和下溢诊断报警信息将以模拟数据值的形式报告，即使在模块组态中禁用这些报警也会如此。

[2] 对于电阻范围，始终会禁用下溢检测。

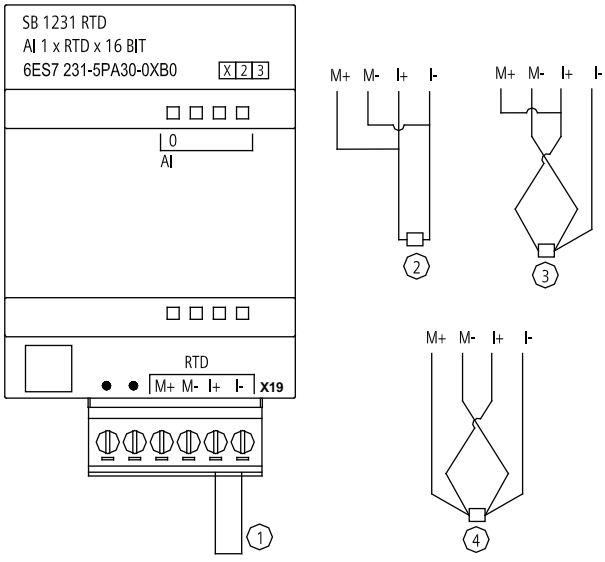
[3] 如果断线报警已禁用，但传感器接线存在开路情况，则模块可能会报告随机值。

SM 1231 RTD 模拟信号板可测量连接到信号板输入的电阻值。测量类型可选为“电阻”型或“热电阻”型。

- “电阻”：额定范围的满量程值将是十进制数 27648。
- “热电阻”：将度数乘 10 得到该值（例如，25.3 度将报告为十进制数 253）。将度数乘 100 得到气候范围值（例如，25.34 度将报告为十进制数 2534）。

SB 1231 RTD 信号板支持采用 2 线、3 线和 4 线制方式连接到传感器电阻进行测量。

表格 A-236 SB 1231 AI 1 x 16 位 RTD 的接线图

SB 1231 AI 1 x 16 位 RTD (6ES7231-5PA30-0XB0)


① 环接未使用的 RTD 输入

② 2 线制 RTD

③ 3 线制 RTD

④ 4 线制 RTD

注：连接器必须镀金。有关订货号，请参见附录 C“备件”。

表格 A-237 SB 1231 AI 1 x 16 位 RTD (6ES7231-5PA30-0XB0) 的连接器引脚位置

引脚	X19（镀金）
1	无连接
2	无连接
3	AI 0 M+/RTD
4	AI 0 M-/RTD
5	AI 0 I+/RTD
6	AI 0 I-/RTD

A.14.5.2 SB 1231 RTD 选型表

表格 A-238 RTD 模块支持的不同传感器的范围和精度

温度系数	RTD 类型	低于范围最小值 [1]	额定范围下限	额定范围上限	超出范围最大值 [2]	25 °C 时的额定范围精度	-20 °C 到 60 °C 时的额定范围精度
Pt 0.003850 ITS90 DIN EN 60751	Pt 100 气候型	-145.00 °C	-120.00 °C	-145.00 °C	-155.00 °C	±0.20 °C	±0.40 °C
	Pt 10	-243.0 °C	-200.0 °C	850.0 °C	1000.0 °C	±1.0 °C	±2.0 °C
	Pt 50	-243.0 °C	-200.0 °C	850.0 °C	1000.0 °C	±0.5 °C	±1.0 °C
	Pt 100						
	Pt 200						
	Pt 500						
	Pt 1000						
Pt 0.003902 Pt 0.003916 Pt 0.003920	Pt 100	-243.0 °C	-200.0 °C	850.0 °C	1000.0 °C	±0.5 °C	±1.0 °C
	Pt 200						
	Pt 500						
	Pt 1000						
Pt 0.003910	Pt 10	-273.2 °C	-240.0 °C	1100.0 °C	1295 °C	±1.0 °C	±2.0 °C
	Pt 50	-273.2 °C	-240.0 °C	1100.0 °C	1295 °C	±0.8 °C	±1.6 °C
	Pt 100						
	Pt 500						
Ni 0.006720 Ni 0.006180	Ni 100	-105.0 °C	-60.0 °C	250.0 °C	295.0 °C	±0.5 °C	±1.0 °C
	Ni 120						
	Ni 200						
	Ni 500						
	Ni 1000						
LG-Ni 0.005000	LG-Ni 1000	-105.0 °C	-60.0 °C	250.0 °C	295.0 °C	±0.5 °C	±1.0 °C
Ni 0.006170	Ni 100	-105.0 °C	-60.0 °C	180.0 °C	212.4 °C	±0.5 °C	±1.0 °C
Cu 0.004270	Cu 10	-240.0 °C	-200.0 °C	260.0 °C	312.0 °C	±1.0 °	±2.0 °C

温度系数	RTD 类型	低于范围最小值 [1]	额定范围下限	额定范围上限	超出范围最大值 [2]	25 °C 时的额定范围精度	-20 °C 到 60 °C 时的额定范围精度
Cu 0.004260	Cu 10	-60.0 °C	-50.0 °C	200.0 °C	240.0 °C	±1.0 °C	±2.0 °C
	Cu 50	-60.0 °C	-50.0 °C	200.0 °C	240.0 °C	±0.6 °C	±1.2 °C
	Cu 100						
Cu 0.004280	Cu 10	-240.0 °C	-200.0 °C	200.0 °C	240.0 °C	±1.0 °C	±2.0 °C
	Cu 50	-240.0 °C	-200.0 °C	200.0 °C	240.0 °C	±0.7 °C	±1.4 °C
	Cu 100						

[1] “低于范围最小值”以下的 RTD 值报告为 -32768。

[2] 超出范围最大值以上的 RTD 值报告为 +32768。

表格 A-239 电阻

范围	低于范围最小值	额定范围下限	额定范围上限	超出范围最大值 [1]	25 °C 时的额定范围精度	-20 °C 到 60 °C 时的额定范围精度
150 Ω	不适用	0 (0 Ω)	27648 (150 Ω)	176.383 Ω	±0.05%	±0.1%
300 Ω	不适用	0 (0 Ω)	27648 (300 Ω)	352.767 Ω	±0.05%	±0.1%
600 Ω	不适用	0 (0 Ω)	27648 (600 Ω)	705.534 Ω	±0.05%	±0.1%

[1] 超出范围最大值以上的电阻值报告为 32767。

说明

对于没有连接传感器的激活通道，模块会报告 32767。 如果还启用了开路检测，模块会使相应的红色 LED 闪烁。

若使用 4 线制连接，对于 10 Ω RTD 范围，将得到最高精度。

2 线模式的连接线电阻会导致传感器读数误差，因此无法保证精度。

表格 A-240 RTD 模块的噪声消减和更新时间

抑制频率选择	积分时间	4/2 线制，1 通道模块 更新时间（秒）	3 线制，1 通道模块 更新时间（秒）
400 Hz (2.5 ms)	10 ms [1]	0.036	0.071
60 Hz (16.6 ms)	16.67 ms	0.056	0.111

抑制频率选择	积分时间	4/2 线制，1 通道模块 更新时间（秒）	3 线制，1 通道模块 更新时间（秒）
50 Hz (20 ms)	20 ms	0.066	1.086
10 Hz (100 ms)	100 ms	0.306	0.611

[1] 在选择 400 Hz 滤波器时，要维持模块的分辨率和精度，积分时间应为 10 ms。该滤波器还可抑制 100 Hz 和 200 Hz 的噪声。

说明

对模块上电后，模块将对模数转换器执行内部校准。 在此期间，模块将报告每个通道的值为 32767，直到相应通道出现有效值为止。 用户程序可能需要考虑这段初始化时间。 由于模块的组态可能改变初始化时长，因此，应验证组态中模块的行为。 如果需要，可以在用户程序中包含逻辑，以适应模块的初始化时间。

A.15 BB 1297 电池板

BB 1297 电池板

S7-1200 BB 1297 电池板适用于实时时钟的长期备份。它可插入 S7-1200 CPU（固件版本 3.0 及更高版本）的单个板插槽中。必须将 BB 1297 添加到设备组态并将硬件配置下载到 CPU 中，BB 才能正常工作。

电池（型号 CR1025）未随 BB 1297 一起提供，必须由用户另行购买。

说明

BB 1297 在机械设计上适合固件版本为 3.0 及以上版本的 CPU。
不要将 BB 1297 与较早版本的 CPU 一起使用，因为 BB 1297 连接器无法插入 CPU 中。

⚠ 警告

在 BB 1297 中安装未规定的电池或将未规定的电池连接到电路，可能会导致火灾或部件元件损坏以及不可预测的设备运行情况。

火灾或不可预测的设备运行状况可能导致死亡、严重人身伤害或财产损坏。
请仅使用规定的 CR1025 电池作为实时时钟的后备电源。

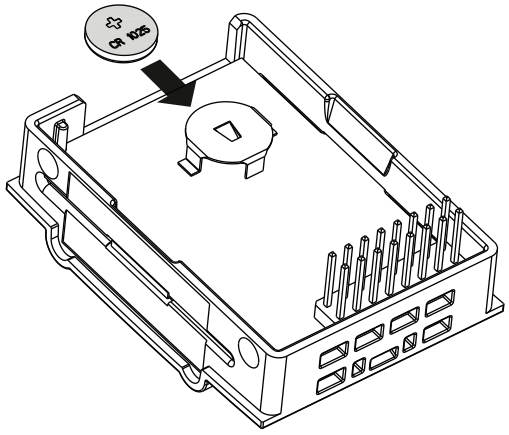
表格 A-241 常规规范

技术数据	BB 1297 电池板
产品编号	6ES7297-0AX30-0XA0
尺寸 W x H x D (mm)	38 x 62 x 21
重量	28 g
功耗	0.5 W
电流消耗（SM 总线）	11 mA
电流消耗 (24 V DC)	无

电池（未包含）	BB 1297 电池板
保持时间	大约 1 年
电池类型	CR1025，请参见安装或更换 BB 1297 电池板中的电池 (页 62)
额定电压	3 V
额定容量	至少 30 mAH

诊断	BB 1297 电池板
临界电池电压	< 2.5 V
电池诊断	低压指示灯： ● 电池电压低会使 CPU MAINT LED 呈琥珀色常亮。 ● 诊断缓冲区事件：16#06:2700“需要子模块维护：至少一个电池已耗尽 (BATTF)”
电池状态	提供的电池状态位 0 = 电池正常 1 = 电池电量低
电池状态更新	电池状态会在开机时更新，之后在 CPU 处于 RUN 模式时，每天更新一次。

表格 A-242 BB 1297 电池板插件图

BB 1297 电池板 (6ES7297-0AX30-0XA0)	
	

A.16 通信接口

A.16.1 PROFIBUS

A.16.1.1 CM 1242-5 PROFIBUS DP 从站

表格 A-243 CM 1242-5 的技术数据

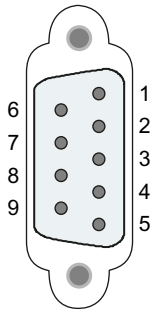
技术数据	
产品编号	6GK7242-5DX30-0XE0
接口	
与 PROFIBUS 的连接	9 针 D 型母连接器
连接网络组件（例如，光纤网络组件）时 PROFIBUS 接口上的最大电流消耗	5 V 时为 15 mA（仅限总线终端）*)
允许的环境条件	
环境温度 • 存储期间 • 运输期间 • 垂直安装（DIN 导轨水平）时的运行 • 水平安装（DIN 导轨垂直）时的运行	 • -40 °C 到 70 °C • -40 °C 到 70 °C • 0 °C 到 55 °C • 0 °C 到 45 °C

技术数据	
在 25 °C 下运行期间的最大相对湿度，无结露	95 %
防护等级	IP20
电源、电流消耗和功耗	
电源类型	DC
背板总线的电源	5 V
电流消耗（典型）	150 mA
有效功耗（典型值）	0.75 W
电气隔离 • PROFIBUS 接口到地 • PROFIBUS 接口到内部电路	700 V DC，持续 1 分钟
尺寸和重量	
• 宽度 • 高度 • 厚度	• 30 mm • 100 mm • 75 mm
重量 • 净重 • 含包装重量	 • 115 g • 152 g

*)VP（引脚 6）和 DGND（引脚 5）之间连接的外部耗电装置的电流负载不得超过总线终端 15 mA 的最大电流（防短路）。

A.16.1.2 CM 1242-5 的 D 型插座的引脚分配

PROFIBUS 接口



表格 A-244 D 型插座的引脚分配

引脚	描述	引脚	描述
1	- 未使用 -	6	P5V2：+5V 电源
2	- 未使用 -	7	- 未使用 -
3	RxD/TxD-P：数据线 B	8	RxD/TxD-N：数据线 A
4	RTS	9	- 未使用 -
5	M5V2：数据参考电位（接地 DGND）	外壳	接地线

A.16.1.3 CM 1243-5 PROFIBUS DP 主站

表格 A-245 CM 1243-5 的技术数据

技术数据	
产品编号	6GK7243-5DX30-0XE0
接口	
与 PROFIBUS 的连接	9 针 D 型母连接器
连接网络组件（例如，光纤网络组件）时 PROFIBUS 接口上的最大电流消耗	5 V 时为 15 mA（仅限总线终端）*)
允许的环境条件	
环境温度 ● 存储期间 ● 运输期间 ● 垂直安装（DIN 导轨水平）时的运行 ● 水平安装（DIN 导轨垂直）时的运行	 ● -40 °C 到 70 °C ● -40 °C 到 70 °C ● 0 °C 到 55 °C ● 0 °C 到 45 °C
在 25 °C 下运行期间的最大相对湿度，无结露	95 %
防护等级	IP20
电源、电流消耗和功耗	
电源类型	DC
电源/外部 ● 最小值 ● 最大值	24 V ● 19.2 V ● 28.8 V

技术数据	
电流消耗（典型） ● 通过 24 V DC ● 通过 S7-1200 背板总线	 ● 100 mA ● 0 mA
有效功耗（典型值） ● 通过 24 V DC ● 通过 S7-1200 背板总线	 ● 2.4 W ● 0 W
电源 24 V DC/外部 ● 最小电缆横截面积 ● 最大电缆横截面积 ● 螺钉型端子的紧固扭矩	 ● 最小值：0.14 mm^2 (AWG 25) ● 最大值：1.5 mm^2 (AWG 15) ● 0.45 Nm (4 lb-in)
电气隔离 ● PROFIBUS 接口到地 ● PROFIBUS 接口到内部电路	700 V DC，持续 1 分钟
尺寸和重量	
● 宽度 ● 高度 ● 厚度	● 30 mm ● 100 mm ● 75 mm
重量 ● 净重 ● 含包装重量	 ● 134 g ● 171 g

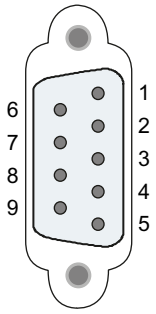
*)VP（引脚 6）和 DGND（引脚 5）之间连接的外部耗电装置的电流负载不得超过总线终端 15 mA 的最大电流（防短路）。

说明

CM 1243-5（PROFIBUS 主站模块）必须由 CPU 的 24 V DC 传感器电源供电。

A.16.1.4 CM 1243-5 的 D 型插座的引脚分配

PROFIBUS 接口



表格 A-246 D 型插座的引脚分配

引脚	描述	引脚	描述
1	- 未使用 -	6	VP： +5 V 电源，仅适用于总线终端电阻，不适合为外部设备供电
2	- 未使用 -	7	- 未使用 -
3	RxD/TxD-P：数据线 B	8	RxD/TxD-N：数据线 A
4	CNTR-P：RTS	9	- 未使用 -
5	DGND： 数据信号和 VP 的地	外壳	接地线

PROFIBUS 电缆

说明

接触 PROFIBUS 电缆的屏蔽层

必须接触 PROFIBUS 电缆的屏蔽层。

为此，从 PROFIBUS 电缆末端剥去绝缘层，然后将屏蔽层连接到功能地。

A.16.2 CP 1242-7

说明

CP 1242-7 未经海事应用认证

CP 1242-7 未经海事认证。

说明

要使用这些模块，CPU 固件必须为 V2.0 或更高版本。

A.16.2.1 CP 1242-7 GPRS

表格 A-247 CP 1242-7 GPRS V2 的技术数据

技术数据	
产品编号	6GK7242-7KX3-0XE0
无线接口	
天线连接器	SMA 插座
额定阻抗	50 欧姆
无线连接	
最大发射功率	• GSM 850，类别 4：+33 dBm ± 2 dBm • GSM 900，类别 4：+33 dBm ± 2 dBm • GSM 1800，类别 1：+30 dBm ± 2 dBm • GSM 1900，类别 1：+30 dBm ± 2 dBm
GPRS	多槽类别 10 B 类设备 编码方式 1...4 (GMSK)
SMS	发出模式：MO 服务：点对点
允许的环境条件	
环境温度 • 存储期间 • 运输期间 • 垂直安装（DIN 导轨水平）时的运行 • 水平安装（DIN 导轨垂直）时的运行	 • -40 °C 到 70 °C • -40 °C 到 70 °C • 0 °C 到 55 °C • 0 °C 到 45 °C

技术数据	
在 25 °C 下运行期间的最大相对湿度，无结露	95 %
防护等级	IP20
电源、电流消耗和功耗	
电源类型	DC
电源/外部 • 最小值 • 最大值	24 V • 19.2 V • 28.8 V
电流消耗（典型） • 通过 24 V DC • 通过 S7-1200 背板总线	 • 100 mA • 0 mA
有效功耗（典型值） • 通过 24 V DC • 通过 S7-1200 背板总线	 • 2.4 W • 0 W
24 V 直流电源 • 最小电缆横截面积 • 最大电缆横截面积 • 螺钉型端子的紧固扭矩	 • 最小值：0.14 mm^2 (AWG 25) • 最大值：1.5 mm^2 (AWG 15) • 0.45 Nm (4 lb-in)
电气隔离 电源单元到内部电路	700 V DC，持续 1 分钟
尺寸和重量	
• 宽度 • 高度 • 厚度	• 30 mm • 100 mm • 75 mm
重量 • 净重 • 含包装重量	 • 133 g • 170 g

说明

防止 CPU 受到天线干扰

如果天线距离过近或未使用推荐的天线，则可能会对 CPU 造成干扰。有关推荐使用的天线，请参见 LTE/UMTS/GSM 紧凑型操作的天线 ANT794-4MR 说明 (https://support.industry.siemens.com/cs/cn/zh/view/23119005/en)（只提供英文和德文版）。

A.16.2.2 GSM/GPRS 天线 ANT794-4MR

ANT794-4MR GSM/GPRS 天线的技术数据

ANT794-4MR	
产品编号	6NH9860-1AA00
移动无线网络	GSM/GPRS
频率范围	• 824 到 960 MHz（GSM 850、900） • 1710 到 1880 MHz (GSM 1800) • 1900 到 2200 MHz (GSM/UMTS)
特性	全向
天线增益	0 dB
阻抗	50 欧姆
驻波比 (SWR)	< 2,0
最大功率	20 W
极性	线性垂直
连接器	SMA
天线电缆长度	5 m
外部材料	硬 PVC，抗 UV
防护等级	IP20
允许的环境条件 • 工作温度 • 运输/存储温度 • 相对湿度	 • -40 °C 到 +70 °C • -40 °C 到 +70 °C • 100 %
外部材料	硬 PVC，抗 UV
结构	天线带 5 m 固定电缆和 SMA 公连接器
尺寸 (D x H) (mm)	25 x 193
重量 • 天线（包括电缆） • 配件	 • 310 g • 54 g
安装	使用随附支架

A.16.2.3 平头天线 ANT794-3M

平头天线 ANT794-3M 的技术数据

ANT794-3M		
产品编号	6NH9870-1AA00	
移动无线网络	GSM 900	GSM 1800/1900
频率范围	890 - 960 MHz	1710 - 1990 MHz
驻波比 (VSWR)	≤ 2:1	≤ 1.5:1
回波损耗 (Tx)	≈ 10 dB	≈ 14 dB
天线增益	0 dB	
阻抗	50 欧姆	
最大功率	10 W	
天线电缆	带 SMA 公连接器的 HF 电缆 RG 174（固定）	
电缆长度	1.2 m	
防护等级	IP64	
允许的温度范围	-40 °C 到 +75 °C	
易燃性	UL 94 V2	
外部材料	ABS Polylac PA-765，浅灰色 (RAL 7035)	
尺寸（W x L x H，以 mm 为单位）	70.5 x 146.5 x 20.5	
重量	130 g	

A.16.3 CM 1243-2 AS-i 主站

A.16.3.1 AS-i 主站 CM 1243-2 的技术数据

表格 A-248 AS-i 主站 CM 1243-2 的技术数据

技术数据	
订货号	3RK7243-2AA30-0XB0
固件版本	V1.0
日期	01.12.2011

技术数据	
接口	
最大电流消耗 通过 S7-1200 背板总线 通过 AS-i 电缆	 最大 250 mA， 电源电压 S7-1200 通信总线 5 V DC 最大 100 mA
ASI+/ASI- 端子间的 最大载流能力	8 A
引脚分配	请参见 AS-i 主站的电气连接 (页 1632)部分
导线横截面	0.2 mm² (AWG 24) ... 3.3 mm² (AWG 12)
ASI 连接器紧固扭矩	0.56 Nm
允许的环境条件	
环境温度 存储期间 运输期间 垂直安装（水平标准安装轨）时的运行阶段 水平安装（垂直标准安装轨）时的运行阶段	 -40 °C 到 70 °C -40 °C 到 70 °C 0 °C 到 55 °C 0 °C 到 45 °C
在 25 °C 下运行阶段的最大相对湿度，无冷凝	95 %
防护等级	IP20
电源、电流消耗和功率损耗	
电源类型	DC
电流消耗（典型值） 通过 S7-1200 背板总线	 200 mA
总功率损耗（典型值）： • 通过 S7-1200 背板总线 • 通过 AS-i 电缆	 1 W 2.4 W
尺寸和重量	

技术数据	
宽度 高度 厚度	30 mm 100 mm 75 mm
重量 净重 含包装重量	 122 g 159 g

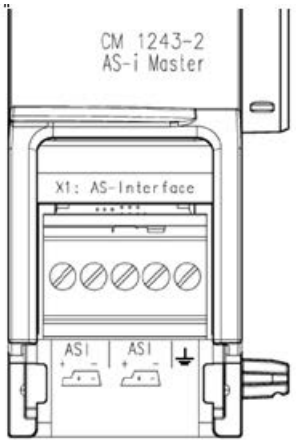
A.16.3.2 AS-i 主站的电气连接

AS-i 主站 CM 1243-2 的电源

通过 S7-1200 的通信总线为 AS-i 主站 CM 1243-2 供电。即，在 AS-i 电源发生故障后仍可向 S7-1200 发送诊断消息。通信总线的连接在 AS-i 主站 CM 1243-2 的右侧。

AS 接口端子

用于连接 AS-i 电缆的可拆卸端子位于 AS-i 主站 CM 1243-2 前面下盖的后面。



如果使用 AS-i 型电缆，则可以通过以下符号识别正确的电缆极性

有关如何拆卸和重新安装端子排的信息，请参见“安装” (页 67)一章。

说明

端子触点的最大载流能力

连接触点的载流能力最大为 8 A。如果 AS-i 电缆的载流能力超过此值，AS-i 主站 CM 1243-2 不能直接连接到 AS-i 电缆，而必须通过分支电缆进行连接（AS-i 主站 CM 1243-2 上只分配了一个连接对）。
另外，如果正在通过 AS-i 主站传输电流并且存在 4 A 以上的电流，请确保使用的电缆适用于最低为 75 °C 的工作温度。
有关连接 AS-i 电缆的更多信息，请参见《SIMATIC S7-1200 的 AS-i 主站 CM 1243-2 和 AS-i 数据解耦单元 DCM 1271》手册的“模块安装、连接和调试”部分。

端子分配

标签	含义
ASI+	AS-i 连接 - 正极性
ASI–	AS-i 连接 - 负极性
⏚	功能性接地

A.16.4 RS232、RS422 和 RS485

A.16.4.1 CB 1241 RS485 规范

说明

要使用此 CB，CPU 固件必须为 V2.0 或更高版本。

表格 A-249 常规规范

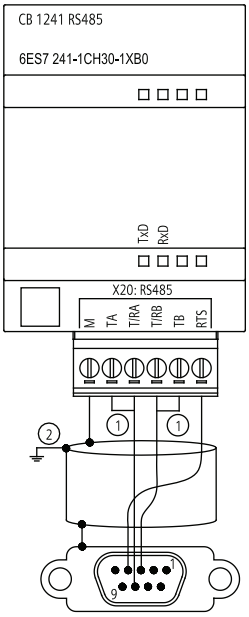
技术数据	CB 1241 RS485
订货号	6ES7241-1CH30-1XB0
尺寸 W x H x D (mm)	38 x 62 x 21
重量	40 g

表格 A-250 发送器和接收器

技术数据	CB 1241 RS485
类型	RS485（2 线制半双工）
共模电压范围	-7 V 到 +12 V，1 秒，3 VRMS 连续
发送器差动输出电压	R_L = 100 Ω 时最小 2 V， R_L = 54 Ω 时最小 1.5 V
端接和偏置	B 上 10K 对 +5 V，RS485 引脚 3 A 上 10K 对 GND，RS485 引脚 4
可选终端	短针 TB 对针 T/RB，有效终端阻抗为 127 Ω，连接至 RS485 针 3 短针 TA 对针 T/RA，有效终端阻抗为 127 Ω，连接至 RS485 针 4
接收器输入阻抗	最小 5.4K Ω，包括终端
接收器阈值/灵敏度	最低 +/- 0.2 V，典型滞后 60 mV
隔离 RS485 信号与机壳接地 RS485 信号与 CPU 逻辑公共端	707 V DC（型式测试）
电缆长度，屏蔽	最长 1000 m
波特率	300 波特、600 波特、1.2 Kb、2.4 Kb、4.8 Kb、9.6 Kb（默认值）、19.2 Kb、38.4 Kb、57.6 Kb、76.8 Kb、115.2 Kb
奇偶校验	无奇偶校验（默认），偶数，奇数，传号（奇偶校验位始终设为 1），空号（奇偶校验位始终设为 0）
停止位的数目	1（默认值），2
流控制	不支持
等待时间	0 到 65535 ms

表格 A-251 电源

技术数据	CB 1241 RS485
功率损失（损耗）	1.5 W
最大电流消耗（SM 总线）	50 mA
最大电流消耗 (24 V DC)	80 mA

CB 1241 RS485 (6ES7241-1CH30-1XB0)	
	
① 如图所示连接“TA”和“TB”以终止网络。（仅端接 RS485 网络上的终端设备。）	
② 使用屏蔽双绞线电缆，并将电缆屏蔽接地。	

只能端接 RS485 网络的两端。不会端接或偏置这两个终端设备之间的设备。请参见 “偏置和端接 RS485 网络连接器 (页 1114)”主题

表格 A-252 CB 1241 RS485 (6ES7241-1CH30-1XB0) 连接器引脚位置

引脚	9 针连接器	X20
1	RS485/逻辑接地	--
2	RS485/未使用	--
3	RS485/TxD+	4 - T/RB
4	RS485/RTS	6 - RTS
5	RS485/逻辑接地	--
6	RS485/5 V 电源	--
7	RS485/未使用	--
8	RS485/TxD-	3 - T/RA
9	RS485/未使用	--
Shell		1 - M

A.16.4.2 CM 1241 RS232 规范

表格 A-253 常规规范

技术数据	CM 1241 RS232
订货号	6ES7241-1AH32-0XB0
尺寸 (mm)	30 x 100 x 75
重量	150 g

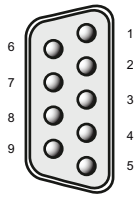
表格 A-254 发送器和接收器

技术数据	CM 1241 RS232
类型	RS232（全双工）
发送器输出电压	R_L = 3K Ω 时最小 +/- 5 V
传送输出电压	+/- 15 V DC，最大值
接收器输入阻抗	最小 3 K Ω
接收器阈值/灵敏度	最低 0.8 V，最高 2.4 V 典型滞后 0.5 V
接收器输入电压	+/- 30 V DC，最大值
隔离 RS 232 信号与机壳接地 RS 232 信号与 CPU 逻辑公共端	707 V DC（型式测试）
电缆长度，屏蔽	最长 10 m
波特率	300 波特、600 波特、1.2 Kb、2.4 Kb、4.8 Kb、9.6 Kb（默认值）、19.2 Kb、38.4 Kb、57.6 Kb、76.8 Kb、115.2 Kb
奇偶校验	无奇偶校验（默认），偶数，奇数，传号（奇偶校验位始终设为 1），空号（奇偶校验位始终设为 0）
停止位的数目	1（默认值），2
流控制	硬件，软件
等待时间	0 到 65535 ms

表格 A-255 电源

技术数据	CM 1241 RS232
功率损失（损耗）	1 W
取自 +5 V DC	200 mA

表格 A-256 RS232 连接器（公）

引脚	说明	连接器（插头式）	引脚	说明
1 DCD	数据载波检测：输入		6 DSR	数据设备就绪：输入
2 RxD	从 DCE 接收数据：输入		7 RTS	请求发送：输出
3 TxD	传送数据到 DCE：输出		8 CTS	允许发送：输入
4 DTR	数据终端就绪：输出		9 RI	振铃指示器（未用）
5 GND	逻辑地		SHELL	机壳接地

A.16.4.3 CM 1241 RS422/485 技术规范

CM 1241 RS422/485 技术数据

表格 A-257 常规规范

技术数据	CM 1241 RS422/485
产品编号	6ES7241-1CH32-0XB0
尺寸 W x H x H (mm)	30 x 100 x 75
重量	155 g

表格 A-258 发送器和接收器

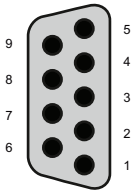
技术数据	CM 1241 RS422/485
类型	RS422 或 RS485，9 针 D 型插孔式连接器
共模电压范围	-7 V 到 +12 V，1 秒，3 VRMS 连续
发送器差动输出电压	R_L = 100 Ω 时最小 2 V， R_L = 54 Ω 时最小 1.5 V

技术数据	CM 1241 RS422/485
端接和偏置	B 上 10K Ω 对 +5 V，PROFIBUS 引脚 3 A 上 10K Ω 对 GND，PROFIBUS 引脚 8 提供内部偏置选项，或无内部偏置。在所有情况下，都需要外部终端，请参见 “偏置和端接 RS485 网络连接器 (页 1114)”与 “S7-1200 可编程控制器系统手册中的 RS422 和 RS485 组态 (页 1168)”
接收器输入阻抗	最小 5.4K Ω，包括终端
接收器阈值/灵敏度	最低 +/- 0.2 V，典型滞后 60 mV
隔离 RS485 信号与机壳接地 RS485 信号与 CPU 逻辑公共端	707 V DC（型式测试）
电缆长度，屏蔽	最长 1000 m（取决于波特率）
波特率	300 波特、600 波特、1.2 Kb、2.4 Kb、4.8 Kb、9.6 Kb（默认值）、19.2 Kb、38.4 Kb、57.6 Kb、76.8 Kb、115.2 Kb
奇偶校验	无奇偶校验（默认），偶数，奇数，传号（奇偶校验位始终设为 1），空号（奇偶校验位始终设为 0）
停止位的数目	1（默认值），2
流控制	对于 RS422 模式，支持 XON/XOFF
等待时间	0 到 65535 ms

表格 A-259 电源

技术数据	CM 1241 RS422/485
功率损失（损耗）	1.1 W
取自 +5 V DC	220 mA

表格 A-260 RS485 或 RS422 连接器（插孔式）

引脚	说明	连接器（插孔式）	引脚	说明
1	逻辑接地或通信接地		6 PWR	+5 V 与 100 Ω 串联电阻：输出
2 TxD+ [1]	用于连接 RS422 不适用于 RS485：输出		7	未连接
3 TxD+ [2]	信号 B (RxD/TxD+)：输入/输出		8 TXD- [2]	信号 A (RxD/TxD-)：输入/输出
4 RTS [3]	请求发送（TTL 电平）输出		9 TXD- [1]	用于连接 RS422 不适用于 RS485：输出
5 GND	逻辑接地或通信接地		SHELL	机壳接地

[1] 引脚 2 (TxD+) 和引脚 9 (TxD-) 是 RS422 的传送信号。

[2] 引脚 3 (RxD/Tx+) 和引脚 8 (RxD/TxD-) 是 RS485 的传送和接收信号。对于 RS422，引脚 3 是 RxD+，引脚 8 是 RxD-。

[3] RTS 是 TTL 电平信号，可用于控制基于该信号进行工作的其它半双工设备。该信号会在发送时激活，在所有其它时刻都不激活。

A.17 远程服务（TS 适配器和 TS 适配器模块）

以下手册包含有关 TS Adapter IE Basic 和 TS 适配器模块的技术规范：

- 《工业软件工程工具》
  模块化 TS 适配器
- 《工业软件工程工具》
  TS Adapter IE Basic

有关该产品以及产品文档的详细信息，请参见 TS 适配器产品目录网站 (https://eb.automation.siemens.com/mall/en/de/Catalog/Search?searchTerm=TS%20Adapter%20IE%20basic&tab=)。

A.18 SIMATIC 存储卡

容量	产品编号
30 GB	6ES7954-8LT02-0AA0
2 GB	6ES7954-8LP01-0AA0

容量	产品编号
256 MB	6ES7954-8LL02-0AA0
24 MB	6ES7954-8LF02-0AA0
12 MB	6ES7954-8LE02-0AA0
4 MB	6ES7954-8LC02-0AA0

A.19 输入仿真器

表格 A-261 常规规范

技术数据	8 位置仿真器	14 位置仿真器	CPU 1217C 仿真器
订货号	6ES7274-1XF30-0XA0	6ES7274-1XH30-0XA0	6ES7274-1XK30-0XA0
尺寸 W x H x D (mm)	43 x 35 x 23	67 x 35 x 23	93 x 40 x 23
重量	20 g	30 g	43 克
点数	8	14	14
配套 CPU	CPU 1211C、CPU 1212C	CPU 1214C、CPU 1215C	CPU 1217C

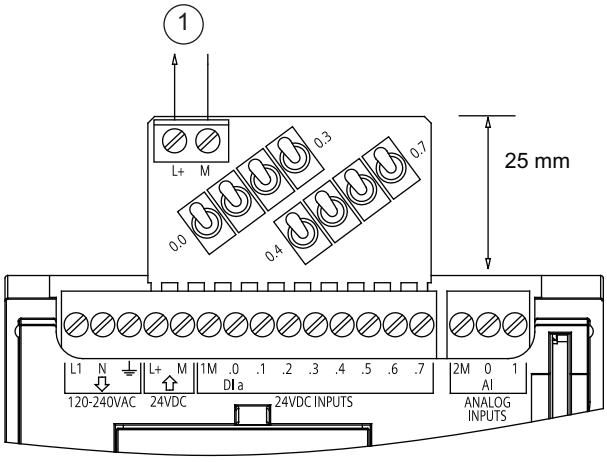
⚠ 警告

安全使用输入仿真器

这些输入仿真器未获准在 Class I DIV 2 或 Class I Zone 2 危险场所使用。如果在 Class I DIV 2 或 Class I Zone 2 场所使用，开关存在潜在的打火危险/爆炸危险。未经批准使用可能导致人员死亡、重伤和/或设备损坏。

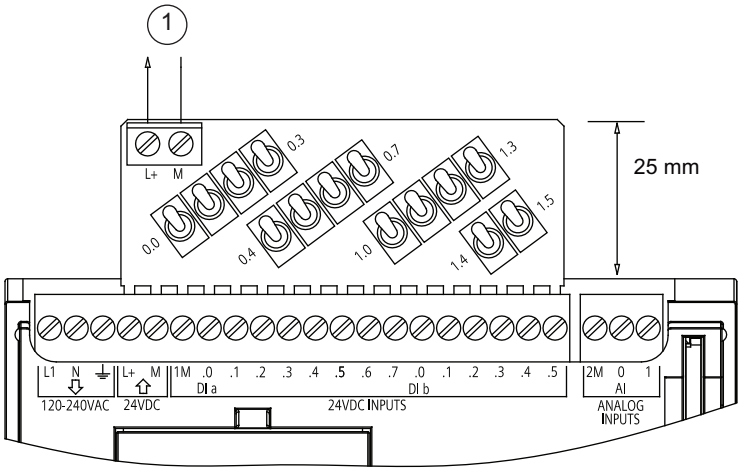
仅在非危险场所使用这些输入仿真器。请勿在 I 类、2 分区或 I 类、2 区危险场所使用。

8 位置仿真器 (6ES7274-1XF30-0XA0)



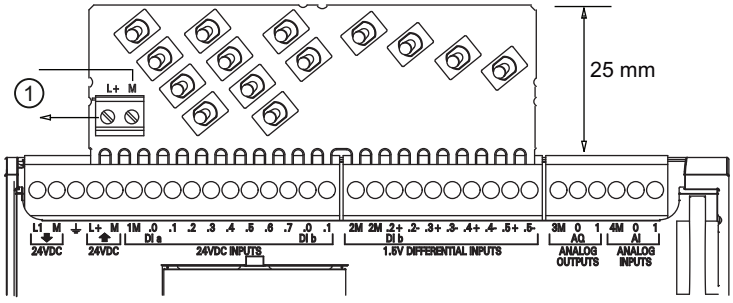
① 24 V DC 传感器电源输出

14 位置仿真器 (6ES7274-1XF30-0XA0)



① 24 V DC 传感器电源输出

CPU 1217C 仿真器 (6ES7274-1XK30-0XA0)



① 24 V DC 传感器电源输出

A.20 S7-1200 电位器模块

S7-1200 电位器模块属于 S7-1200 CPU 附件。每个电位计按照正比于电位计位置的关系输出电压，为两个 CPU 模拟量输入提供 0 V DC 到 10 V DC 的驱动电压。要安装该电位器，请执行以下操作：

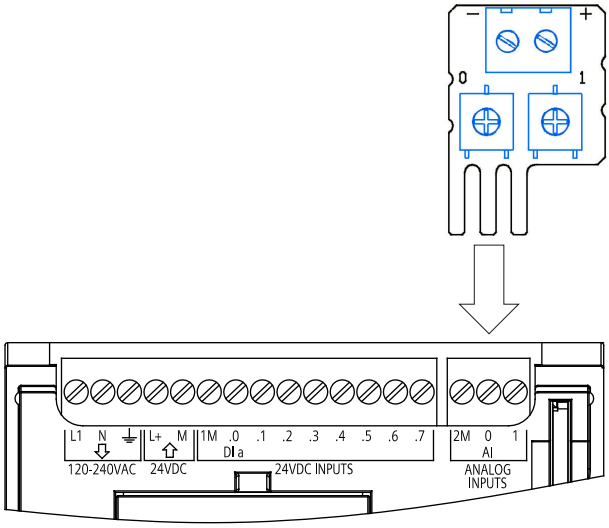
1. 将电路板的“手指”插入任意一个 S7-1200 CPU 模拟量输入端子块，然后将外部 DC 电源连接到电位器模块上的 2 位置连接器。
2. 使用小螺丝刀进行调整：顺时针旋转电位器（向右）将增大电压输出，逆时针旋转（向左）将减小电压输出。

说明

接触 S7-1200 电位器模块时，请遵守 ESD 准则。

技术数据	S7-1200 电位器模块
订货号	6ES7274-1XA30-0XA0
配套 CPU	所有 S7-1200 CPU
电位器数目	2
尺寸 W x H x D (mm)	20 x 33 x 14
重量	26 g
2 位置连接器处的用户供应电压输入 [1] （2 类受限制电源，或 PLC 提供的传感器电源）	16.4 V DC 至 28.8 V DC
电缆长度（米）/类型	<30 m，屏蔽双绞线
输入电流消耗	最大 10 mA
电位器电压输出，至 S7-1200 CPU 模拟量输入 [1]	0 V DC 至 10.5 V DC 最小
隔离	未隔离
环境温度范围	-20 ℃ 到 60 ℃

[1] 电位器模块输出电压是否稳定取决于 2 位置连接器处用户供应输入电压的质量（将其作为模拟量输入电压）。

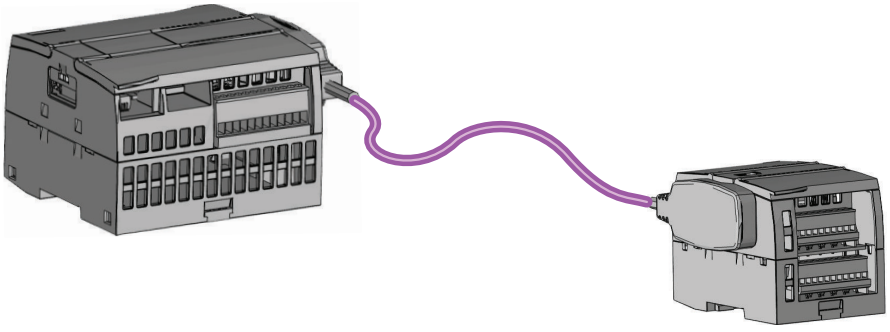


A.21 I/O 扩展电缆

表格 A-262 扩展电缆

技术数据	
订货号	6ES7290-6AA30-0XA0
电缆长度	2 m
重量	200 g

有关安装和拆卸 S7-1200 扩展电缆的信息，请参见安装 (页 68)部分



A.22 随附产品

A.22.1 PM 1207 电源模块

PM 1207 是 SIMATIC S7-1200 的电源模块。它可以提供以下功能：

- 输入 120/230 V AC，输出 24 V DC/2.5A

有关该产品以及产品文档的详细信息，请参见 PM 1207 的产品目录网站 (https://mall.industry.siemens.com/mall/en/de/Catalog/Product/6EP1332-1SH71)。

A.22.2 CSM 1277 紧凑型交换机模块

CSM1277 是工业以太网紧凑型交换机模块。可将其用于增加 S7-1200 的以太网接口，以允许与操作员面板、编程设备或其它控制器同时进行通信。它可以提供以下功能：

- 连接至工业以太网的 4 x RJ45 插座
- 位于顶部的用于外部 24 V DC 电源连接的接线板上的 3 极插头
- 用于工业以太网端口的诊断和状态显示的 LED
- 订货号 6GK7277-1AA00-0AA0

有关该产品以及产品文档的详细信息，请参见 CSM 1277 的产品目录网站 (https://eb.automation.siemens.com/mall/en/de/Catalog/Search?searchTerm=csm%201277&tab=)。

A.22.3 CM CANopen 模块

CM CANopen 是一种插入式模块，可以插入到 SIMATIC S7-1200 PLC 与任意运行 CANopen 的设备之间。 可以将 CM CANopen 组态为主站或从站。 有两种 CM CANopen modules： CANopen 模块（订货号 021620-B）和 CANopen (Ruggedized) 模块（订货号 021730-B）。

CANopen 模块提供以下功能：

- 每个 CPU 可以连接 3 个模块
- 可连接多达 16 个 CANopen 从节点
- 每个模块提供 256 字节输入和 256 字节输出
- 3 个 LED 分别提供模块、网络和 I/O 状态的诊断信息

- 支持将 CANopen 网络组态存储在 PLC 中
- 模块可集成到 TIA Portal 组态套件的硬件目录中
- 可通过内含的 CANopen Configuration Studio 或任何其它外部 CANopen 组态工具实现 CANopen 组态
- 遵循 CANopen 通信配置文件 CiA 301 修订版 4.2 和 CiA 302 修订版 4.1
- 支持透明 CAN 2.0A，用于自定义协议处理
- 预制功能块可用于 TIA Portal 中的所有 PLC 编程
- 包含 CM CANopen 模块；适用于子网络的带螺丝端子的 DSUB。CM CANopen Configuration Studio CD 和 USB 组态电缆

有关该产品以及产品文档的详细信息，请参见 CM CANopen 的产品目录网站。

A.22.4 RF120C 通信模块

通过 RF10C，Siemens RFID 和代码读取系统便能直接并轻松地连接到 S7-1200。读卡器通过点对点连接方式连接到 RF120C。最多可将三个通信模块连接到 CPU 左侧的 S7-1200。通过 TIA Portal 组态 RF120C 通信模块。RF120C 通信模块的订货号为 6GT2002-0LA00。

有关该产品以及产品文档的详细信息，请参见 RF120C 的产品目录网站。

A.22.5 SM 1238 电能表模块

SM 1238 电能表 480 V AC 适用于 S7-1200 系统中的机器级部署。可记录 200 多种不同的电气测量值和能源数据。正因如此，从生产车间到设备级，各组件能源需求变得一目了然。基于 SM 1238 能量计模块提供的测量值，可准确判断具体的能耗和功率需量。

有关该产品、产品文档及规格的详细信息，请参见 CSM 1277 的产品目录网站中的 SM 1238 能量计模块 (https://support.industry.siemens.com/cs/ww/en/view/109483435)部分。

A.22.6 SIWAREX 电子称重系统

SIWAREX WP231、WP241 和 WP251

SIWAREX WP231、WP241 和 WP251 电子称重系统可以用于 S7-1200。该模块使用了现代控制系统中的所有功能，如综合通信、操作和监控、诊断系统以及 TIA Portal 中的组态工具。

- SIWAREX WP231 (https://support.industry.siemens.com/cs/cn/zh/view/90229056)，用于应变规称重传感器的校准器称重电子元件（单通道）/SIMATIC S7-1200、RS485 和以太网接口、板载 I/O 的全桥接器 (1-4 MV/V)：4 DI/4 DO、1 AO (0/4...20 MA)
- SIWAREX WP241 (https://support.industry.siemens.com/cs/cn/zh/view/90229063)，用于应变规称重传感器的皮带秤电子元件（单通道）/SIMATIC S7-1200、RS485 和以太网接口、板载 I/O 的全桥接器 (1-4 M/V)4 DI/4 DO、1 AO (0/4...20 MA)
- SIWAREX WP251，用于应变规称重传感器批处理和填充过程的称重电子元件（单通道）/SIMATIC S7-1200、RS485 和以太网接口、板载 I/O 的全桥接器 (1-4 MV/V)4 DI/4 DO、1 AO (0/4...20 MA)

参见

SIWAREX WP251 (https://support.industry.siemens.com/cs/cn/zh/view/109481751/en)

计算功率预算 B

CPU 有一个内部电源，用于为 CPU 本身和任何扩展模块供电以及满足其它 24 V DC 用户的功率要求。

有四种类型的扩展模块：

- 信号模块 (SM) 安装在 CPU 右侧。在不考虑功率预算的情况下，每个 CPU 可允许的最大信号模块数如下。
  - CPU 1214C、CPU 1215C 和 CPU 1217C 允许 8 个信号模块
  - CPU 1212C 允许 2 个信号模块
  - CPU 1211C 不允许任何信号模块
- 通信模块 (CM) 安装在 CPU 左侧。若不考虑功率预算，任何 CPU 都允许最多 3 个通信模块。
- 信号板 (SB)、通信板 (CB) 和电池板 (BB) 安装在 CPU 顶部。任何 CPU 最多允许使用 1 个信号板、通信板或电池板。

请使用以下信息作为指导，确定 CPU 可为您的组态提供多少电能（或电流）。

每个 CPU 都提供了 5 V DC 和 24 V DC 电源：

- 连接了扩展模块时，CPU 会为这些扩展模块提供 5 V DC 电源。如果扩展模块的 5 V DC 功率要求超出 CPU 的功率预算，则必须拆下一些扩展模块直到其功率要求在功率预算范围内。
- 每个 CPU 都有一个 24 V DC 传感器电源，该电源可以为本地输入点或扩展模块上的继电器线圈提供 24 V DC 电源。如果 24 V DC 的功率要求超出 CPU 的功率预算，则可以增加外部 24 V DC 电源为扩展模块供应 24 V DC。必须将 24 V DC 电源手动连接到输入点或继电器线圈。

> ⚠ **警告**
>
> **将外部 24 V DC 电源与 DC 传感器电源并联会导致这两个电源之间有冲突，因为每个电源都试图建立自己首选的输出电压电平。**
>
> 该冲突可能使其中一个电源或两个电源的寿命缩短或立即出现故障，从而导致 PLC 系统的运行不确定。运行不确定可能导致死亡、人员重伤和/或财产损失。
>
> CPU 上的 DC 传感器电源和任何外部电源应分别给不同位置供电。允许将多个公共端连接到一个位置。

PLC 系统中的一些 24 V DC 电源输入端口是互连的，并且通过一个公共逻辑电路连接多个 M 端子。在数据表中指定为非隔离时，CPU 的 24 V DC 电源输入、SM 继电器线圈电源输入

以及非隔离模拟电源输入即是一些互连电路的实例。所有非隔离的 M 端子必须连接到同一个外部参考电位。

⚠ 警告

将非隔离的 M 端子连接到不同参考电位将导致意外的电流，该电流可能导致 PLC 和连接设备损坏或运行不确定。

这种损坏或不确定运行可能导致死亡、人员重伤和/或财产损失。

务必确保 PLC 系统中的所有非隔离 M 端子都连接到同一个参考电位。

有关 CPU 功率预算和信号模块功率要求的信息，请参见技术规范 (页 1427)。

说明

若超出 CPU 功率预算，将导致无法连接 CPU 所允许的最大数量的模块。

功率预算示例

以下示例以包含一个 CPU 1214C AC/DC/继电器型、一个 SB 1223 2 x 24 V DC 输入/2 x 24 V DC 输出、一个 CM 1241、三个 SM 1223 8 DC 输入/8 路继电器输出以及一个 SM 1221 8 DC 输入的组态为例，给出了功率要求计算办法。该示例一共有 48 个输入和 36 个输出。

说明

该 CPU 已分配驱动内部继电器线圈所需的功率。功率预算计算中无需包括内部继电器线圈的功率要求。

本例中的 CPU 为 SM 提供了足够的 5 V DC 电流，但没有通过传感器电源为所有输入和扩展继电器线圈提供足够的 24 V DC 电流。I/O 需要 456 mA，而 CPU 只提供 400 mA。该安装额外需要一个至少为 56 mA 的 24 V DC 电源以运行所有包括的 24 V DC 输入和输出。

表格 B-1　采样功率预算

CPU 功率预算	**5 V DC**	**24 V DC**
CPU 1214C AC/DC/继电器	1600 mA	400 mA
	减	
系统要求	**5 V DC**	**24 V DC**
CPU 1214C，14 点输入	-	14 * 4 mA = 56 mA
1 SB 1223 2 x 24 V DC 输入/ 2 x 24 V DC 输出	50 mA	2 * 4 mA = 8 mA

CPU 功率预算	5 V DC	24 V DC
1 个 CM 1241 RS422/485，5 V 电源	220 mA	
3 个 SM 1223，5 V 电源	3 * 145 mA = 435 mA	-
1 个 SM 1221，5 V 电源	1 * 105 mA = 105 mA	-
3 个 SM 1223，各 8 点输入	-	3 * 8 * 4 mA = 96 mA
3 个 SM 1223，各 8 个继电器线圈	-	3 * 8 * 11 mA = 264 mA
1 个 SM 1221，各 8 点输入	-	8 * 4 mA = 32 mA
总要求	810 mA	456 mA
等于		
电流差额	**5 V DC**	**24 V DC**
总电流差额	790 mA	(56 mA)

功率预算计算表

通过下表可以确定 S7-1200 CPU 可为您的组态提供多少电源（或电流）。有关用户 CPU 型号的功率预算和信号模块功率要求信息，请参见技术规范 (页 1427)。

表格 B-2　功率预算计算

CPU 功率预算	5 V DC	24 V DC
减		
系统要求	**5 V DC**	**24 V DC**
总要求		
等于		
电流差额	**5 V DC**	**24 V DC**
总电流差额		

订购信息

C

C.1 CPU 模块

表格 C-1 S7-1200 CPU

CPU 型号		订货号
CPU 1211C	CPU 1211C DC/DC/DC	6ES7211-1AE40-0XB0
	CPU 1211C AC/DC/继电器	6ES7211-1BE40-0XB0
	CPU 1211C DC/DC/继电器	6ES7211-1HE40-0XB0
CPU 1212C	CPU 1212C DC/DC/DC	6ES7212-1AE40-0XB0
	CPU 1212C AC/DC/继电器	6ES7212-1BE40-0XB0
	CPU 1212C DC/DC/继电器	6ES7212-1HE40-0XB0
CPU 1214C	CPU 1214C DC/DC/DC	6ES7214-1AG40-0XB0
	CPU 1214C AC/DC/继电器	6ES7214-1BG40-0XB0
	CPU 1214C DC/DC/继电器	6ES7214-1HG40-0XB0
CPU 1215C	CPU 1215C DC/DC/DC	6ES7215-1AG40-0XB0
	CPU 1215C AC/DC/继电器	6ES7215-1BG40-0XB0
	CPU 1215C DC/DC/继电器	6ES7215-1HG40-0XB0
CPU 1217C	CPU 1217C DC/DC/DC	6ES7217-1AG40-0XB0

C.2 信号模块 (SM)、显示模块 (SB) 和 电池模块 (BB)

表格 C-2 信号模块 (SM)

信号模块		订货号
数字量输入	SM 1221 8 x 24 V DC 输入（漏型/源型）	6ES7221-1BF32-0XB0
	SM 1221 16 x 24 V DC 输入（漏型/源型）	6ES7221-1BH32-0XB0

信号模块		订货号
数字量输出	SM 1222 8 x 24 V DC 输出（源型）	6ES7222-1BF32-0XB0
	SM 1222 16 x 24 V DC 输出（源型）	6ES7222-1BH32-0XB0
	SM 1222 16 x 24 V DC 输出（漏型）	6ES7222-1BH32-1XB0
	SM 1222 8 x 继电器输出	6ES7222-1HF32-0XB0
	SM 1222 8 x 继电器输出（切换）	6ES7222-1XF32-0XB0
	SM 1222 16 x 继电器输出	6ES7222-1HH32-0XB0
数字量输入/输出	SM 1223 8 x 24 V DC 输入（漏型 / 源型）/ 8 x 24 V DC 输出（源型）	6ES7223-1BH32-0XB0
	SM 1223 16 x 24 V DC 输入（源型 / 漏型）/16 x 24 V DC 输出（源型）	6ES7223-1BL32-0XB0
	SM 1223 16 x 24 V DC 输入/16 x 24 V DC 输出（漏型）	6ES7223-1BL32-1XB0
	SM 1223 8 x 24 V DC 输入（源型/漏型）/8 x 继电器输出	6ES7223-1PH32-0XB0
	SM 1223 16 x 24 V DC 输入（源型/漏型）/16 x 继电器输出	6ES7223-1PL32-0XB0
	SM 1223 8 x 120/230 V AC 输入（源型/漏型）/8 x 继电器输出	6ES7223-1QH32-0XB0
模拟量输入	SM 1231 4 x 模拟量输入	6ES7231-4HD32-0XB0
	SM 1231 8 x 模拟量输入	6ES7231-4HF32-0XB0
	SM 1231 4 x 模拟量输入 x 16 位（高性能型）	6ES7231-5ND32-0XB0
	SM 1238 电能表 480 V AC	6ES7238-5XA32-0XB0
模拟量输出	SM 1232 2 x 模拟量输出	6ES7232-4HB32-0XB0
	SM 1232 4 x 模拟量输出	6ES7232-4HD32-0XB0
模拟量输入/输出	SM 1234 4 x 模拟量输入/2 x 模拟量输出	6ES7234-4HE32-0XB0
RTD 和热电偶	SM 1231 TC 4 x 16 位	6ES7231-5QD32-0XB0
	SM 1231 TC 8 x 16 位	6ES7231-5QF32-0XB0
	SM 1231 RTD 4 x 16 位	6ES7231-5PD32-0XB0
	SM 1231 RTD 8 x 16 位	6ES7231-5PF32-0XB0

信号模块		订货号
工艺模块	SM 1278 4xIO-Link 主站	6ES7278-4BD32-0XB0
	SIWAREX WP231，用于应变规称重传感器的校准器称重电子元件（单通道）/SIMATIC S7-1200、RS485 和以太网接口、板载 I/O 的全桥接器 (1-4 MV/V)：4 DI/4 DO、1 AO (0/4...20 MA)	7MH4960-2AA01
	SIWAREX WP241，用于应变规称重传感器的皮带秤电子元件（单通道）/SIMATIC S7-1200、RS485 和以太网接口、板载 I/O 的全桥接器 (1-4 M/V)：4 DI/4 DO、1 AO (0/4...20 MA)	7MH4960-4AA01
	SIWAREX WP251，用于应变规称重传感器批处理和填充过程的称重电子元件（单通道）/SIMATIC S7-1200、RS485 和以太网接口、板载 I/O 的全桥接器 (1-4 MV/V)：4 DI/4 DO、1 AO (0/4...20MA)	7MH4960-6AA01

表格 C-3　信号板 (SB) 和电池板 (BB)

信号板和电池板		订货号
数字量输入	SB 1221 200 kHz 4 x 24 V DC 输入（源型）	6ES7221-3BD30-0XB0
	SB 1221 200 kHz 4 x 5 V DC 输入（源型）	6ES7221-3AD30-0XB0
数字量输出	SB 1222 200 kHz 4 x 24 V DC 输出（漏型/源型）	6ES7222-1BD30-0XB0
	SB 1222 200 kHz 4 x 5 V DC 输出（漏型/源型）	6ES7222-1AD30-0XB0
数字量输入/输出	SB 1223 2 x 24 V DC 输入（漏型）/2 x 24 V DC 输出（源型）	6ES7223-0BD30-0XB0
	SB 1223 200 kHz 2 x 24 V DC 输出（源型）/2 x 24 V DC 输出（漏型/源型）	6ES7223-3BD30-0XB0
	SB 1223 200 kHz 2 x 5 V DC 输入（源型）/2 x 5 V DC 输出（漏型/源型）	6ES7223-3AD30-0XB0
模拟量	SB 1232 1 路模拟量输出	6ES7232-4HA30-0XB0
	SB 1231 1 路模拟量输入	6ES7231-4HA30-0XB0
	SB 1231 1 路模拟量输入热电偶	6ES7231-5QA30-0XB0
	SB 1231 1 路模拟量输入 RTD	6ES7231-5PA30-0XB0
电池	BB 1297 电池板（未提供 CR1025 型号电池）	6ES7297-0AX30-0XA0

C.3 通信

表格 C-4 通信模块 (CM)

通信模块 (CM)			订货号
RS232、RS422 和 RS485	CM 1241 RS232	RS232	6ES7241-1AH32-0XB0
	CM 1241 RS422/485	RS422/485	6ES7241-1CH32-0XB0
PROFIBUS	CM 1243-5	PROFIBUS 主站	6GK7243-5DX30-0XE0
	CM 1242-5	PROFIBUS 从站	6GK7242-5DX30-0XE0
AS-i 主站	CM 1243-2	AS-i 主站	3RK7243-2AA30-0XB0

表格 C-5 通信板 (CB)

通信板 (CB)			订货号
RS485	CB 1241 RS485	RS485	6ES7241-1CH30-1XB0

表格 C-6 通信处理器 (CP)

CP	接口	订货号
CP 1242-7 GPRS V2	GPRS	6GK7242-7KX31-0XE0
CP 1243-7 LTE-US	LTE	6GK7243-7SX30-0XE0
CP 1243-7 LTE-EU	LTE	6GK7243-7KX30-0XE0
CP 1243-1	IE 接口	6GK7243-1BX30-0XE0
CP 1243-8 IRC	IE 和串行接口	6GK7243-8RX30-0XE0

表格 C-7 TeleService

TS 适配器	订货号
TS Adapter IE Basic	6ES7972-0EB00-0XA0
TS Adapter IE Advanced	6ES7972-0EA00-0XA0
TS Module GSM	6GK7972-0MG00-0XA0

TS 适配器	订货号
TS Module RS232	6ES7972-0MS00-0XA0
TS Module Modem	6ES7972-0MM00-0XA0
TS Module ISDN	6ES7972-0MD00-0XA0

表格 C-8　附件

附件			订货号
天线	ANT794-4MR	GSM/GPRS 天线	6NH9860-1AA00
	ANT794-3M	平头天线	6NH9870-1AA00

表格 C-9　连接器

连接器类型		订货号
RS485	35 度电缆输出，螺丝端子连接	6ES7972-0BA42-0XA0
	35 度电缆输出，FastConnect 连接	6ES7972-0BA60-0XA0

C.4　故障安全 CPU 和信号模块

表格 C-10　故障安全 CPU

故障安全 CPU 型号		产品编号
CPU 1212FC	CPU 1212FC DC/DC/DC	6ES7212-1AF40-0XB0
	CPU 1212FC DC/DC/继电器	6ES7212-1HF40-0XB0
CPU 1214FC	CPU 1214FC DC/DC/DC	6ES7214-1AF40-0XB0
	CPU 1214FC DC/DC/继电器	6ES7214-1HF40-0XB0
CPU 1215FC	CPU 1215FC DC/DC/DC	6ES7215-1AF40-0XB0
	CPU 1215FC DC/DC/继电器	6ES7215-1HF40-0XB0

表格 C-11　故障安全信号模块

功能安全信号模块		订货号
数字量输入	SM 1226 F-DI 16 x 24 V DC	6ES7226-6BA32-0XB0
数字量输出	SM 1226 F-DQ 4 x 24 V DC	6ES7226-6DA32-0XB0
	SM 1226 F-DQ 2 x Relay	6ES7226-6RA32-0XB0

C.5　其它模块

表格 C-12　随附产品

物品		订货号
电源	PM 1207 电源	6EP1332-1SH71
以太网交换机	CSM 1277 以太网交换机 - 4 端口	6GK7277-1AA10-0AA0
CM CANopen	用于 SIMATIC S7-1200 的 CANopen	021620-B
	用于 SIMATIC S7-1200 的 CANopen (Ruggedized)	021730-B
RF120C	RF120C 通信模块	6GT2002-0LA00

C.6　存储卡

表格 C-13　存储卡

SIMATIC 存储卡	订货号
SIMATIC MC 32 GB	6ES7954-8LT03-0AA0
SIMATIC MC 2 GB	6ES7954-8LP02-0AA0
SIMATIC MC 256 MB	6ES7954-8LL03-0AA0
SIMATIC MC 24 MB	6ES7954-8LF03-0AA0
SIMATIC MC 12 MB	6ES7954-8LE03-0AA0
SIMATIC MC 4 MB	6ES7954-8LC03-0AA0

C.7 Basic HMI 设备

表格 C-14 HMI 设备

HMI 基本型面板	产品编号
KTP400 Basic（单色，PN）	6AV2123-2DB03-0AX0
KTP700 Basic	6AV2123-2GB03-0AX0
KTP700 Basic DP	6AV2123-2GA03-0AX0
KTP900 Basic	6AV2123-2JB03-0AX0
KTP1200 Basic	6AV2123-2MB03-0AX0
KTP1200 Basic DP	6AV2123-2MA03-0AX0

C.8 备件和其它硬件

表格 C-15 扩展电缆、仿真器和末端保持器

物品		产品编号
I/O 扩展电缆	I/O 扩展电缆，2 m	6ES7290-6AA30-0XA0
I/O 仿真器	仿真器（1211C/1212C - 8 位置）	6ES7274-1XF30-0XA0
	仿真器（1214C/1215C - 14 位置）	6ES7274-1XH30-0XA0
	仿真器，CPU 1217C	6ES7274-1XK30-0XA0
电位器模块	S7-1200 电位器模块	6ES7274-1XA30-0XA0
以太网松紧件	单端口 RJ45 松紧件，10/100 Mbps	6ES7290-3AA30-0XA0
	双端口 RJ45 松紧件，10/100 Mbps	6ES7290-3AB30-0XA0
备用门配件	CPU 1211C/1212C	6ES7291-1AA30-0XA0
	CPU 1214C	6ES7291-1AB30-0XA0
	CPU 1215C	6ES7291-1AC30-0XA0
	CPU 1217C	6ES7291-1AD30-0XA0
	单个模块，45 mm	6ES7291-1BA30-0XA0
	单个模块，70 mm	6ES7291-1BB30-0XA0
	通信模块（与 6ES72xx-xxx32-0XB0 和 6ES72xx-xxx30-0XB0 模块配合使用）	6ES7291-1CC30-0XA0
末端保持器	末端保持器，热塑性塑料材质，10 MM	8WA1808
	末端保持器，钢制，10.3 MM	8WA1805

更换端子排连接器

为模块使用正确的端子排非常重要。参见下表和模块规范确定正确的端子排替换件。

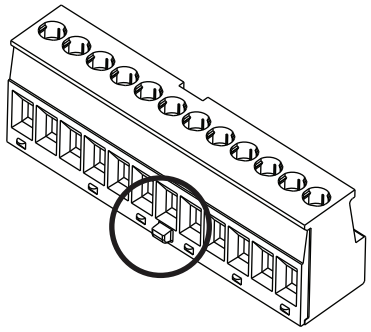
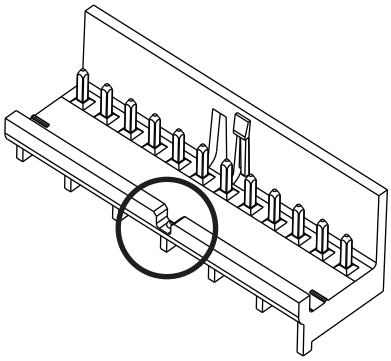
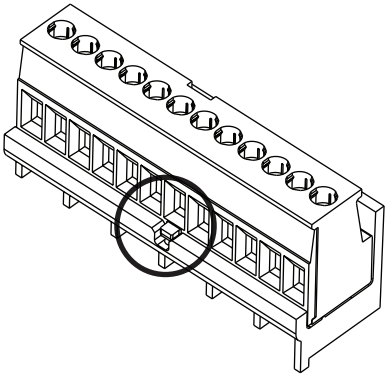
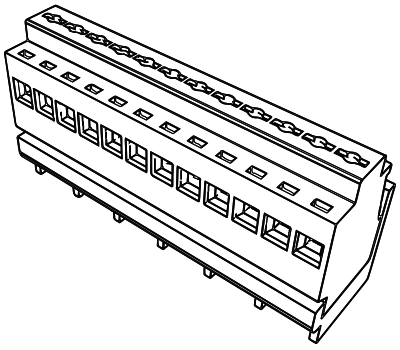
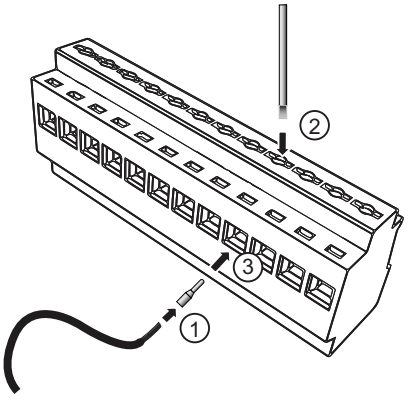
说明

带键可拆卸端子块

PLC 始终需要正确接线，从而确保安全和适当操作。

当更换 CPU 或 SM 中的端子排时，务必为模块使用正确的端子排和正确的接线源。

借助带键这一特性，可避免误将带有高压的端子排接入低压模块，或避免将带有专用电压的端子排接入标准电压模块中。部分端子排特别在左侧、右侧或中间进行了卡槽设计。

		
可拆卸端子排 （带键示例	设备上的键	设备上的键仅和适当可拆卸端子排配合
		
推入式端子排	请按下列步骤操作： 1. 插入电线和压接连接器。 2. 用小螺丝刀压下薄膜以打开插槽。 3. 将压接的连接器插入端子排上打开的插槽中。	

表格 C-16　S7-1200 CPU V4.0 及更高版本 – 端子排备件套件

如果使用 S7-1200 CPU V4.0 及以上版本（订货号）	使用此端子排备用套件（每包 4 件）		
	端子排（螺钉型） 订货号	等效 推入式端子排 订货号	端子排 描述
CPU 1211C DC/DC/DC (6ES7211-1AE40-0XB0)	6ES7292-1BC30-0XA0	6ES7292-2BC30-0XA0	3 针，镀金
	6ES7292-1AH30-0XA0	6ES7292-2AH30-0XA0	8 针，镀锡
	6ES7292-1AP30-0XA0	6ES7292-2AP30-0XA0	14 针，镀锡
CPU 1211C DC/DC/继电器 (6ES7211-1HE40-0XB0)	6ES7292-1BC30-0XA0	6ES7292-2BC30-0XA0	3 针，镀金
	6ES7292-1AH40-0XA0	6ES7292-2AH40-0XA0	8 针，镀锡，带键
	6ES7292-1AP30-0XA0	6ES7292-2AP30-0XA0	14 针，镀锡
CPU 1211C AC/DC/继电器 (6ES7211-1BE40-0XB0)	6ES7292-1BC30-0XA0	6ES7292-2BC30-0XA0	3 针，镀金
	6ES7292-1AH40-0XA0	6ES7292-2AH40-0XA0	8 针，镀锡，带键
	6ES7292-1AP40-0XA0	6ES7292-2AP40-0XA0	14 针，镀锡，带键
CPU 1212C DC/DC/DC (6ES7212-1AE40-0XB0)	6ES7292-1BC30-0XA0	6ES7292-2BC30-0XA0	3 针，镀金
	6ES7292-1AH30-0XA0	6ES7292-2AH30-0XA0	8 针，镀锡
	6ES7292-1AP30-0XA0	6ES7292-2AP30-0XA0	14 针，镀锡

如果使用 S7-1200 CPU V4.0 及以上版本（订货号）	使用此端子排备用套件（每包 4 件）		
	端子排（螺钉型） 订货号	等效 推入式端子排 订货号	端子排 描述
CPU 1212C DC/DC/继电器 (6ES7212-1HE40-0XB0)	6ES7292-1BC30-0XA0	6ES7292-2BC30-0XA0	3 针，镀金
	6ES7292-1AH40-0XA0	6ES7292-2AH40-0XA0	8 针，镀锡，带键
	6ES7292-1AP30-0XA0	6ES7292-2AP30-0XA0	14 针，镀锡
CPU 1212C AC/DC/继电器 (6ES7212-1BE40-0XB0)	6ES7292-1BC30-0XA0	6ES7292-2BC30-0XA0	3 针，镀金
	6ES7292-1AH40-0XA0	6ES7292-2AH40-0XA0	8 针，镀锡，带键
	6ES7292-1AP40-0XA0	6ES7292-2AP40-0XA0	14 针，镀锡，带键
CPU 1214C DC/DC/DC (6ES7214-1AG40-0XB0)	6ES7292-1BC30-0XA0	6ES7292-2BC30-0XA0	3 针，镀金
	6ES7292-1AM30-0XA0	6ES7292-2AM30-0XA0	12 针，镀锡
	6ES7292-1AV30-0XA0	6ES7292-2AV30-0XA0	20 针，镀锡
CPU 1214C DC/DC/继电器 (6ES7214-1HG40-0XB0)	6ES7292-1BC30-0XA0	6ES7292-2BC30-0XA0	3 针，镀金
	6ES7292-1AM40-0XA0	6ES7292-2AM40-0XA0	12 针，镀锡，带键
	6ES7292-1AV30-0XA0	6ES7292-2AV30-0XA0	20 针，镀锡
CPU 1214C AC/DC/继电器 (6ES7214-1BG40-0XB0)	6ES7292-1BC30-0XA0	6ES7292-2BC30-0XA0	3 针，镀金
	6ES7292-1AM40-0XA0	6ES7292-2AM40-0XA0	12 针，镀锡，带键
	6ES7292-1AV40-0XA0	6ES7292-2AV40-0XA0	20 针，镀锡，带键

如果使用 S7-1200 CPU V4.0 及以上版本（订货号）	使用此端子排备用套件（每包 4 件）		
	端子排（螺钉型） 订货号	等效 推入式端子排 订货号	端子排 描述
CPU 1215C DC/DC/DC (6ES7215-1AG40-0XB0)	6ES7292-1BF30-0XB0	6ES7292-2BF30-0XB0	6 针，镀金
	6ES7292-1AM30-0XA0	6ES7292-2AM30-0XA0	12 针，镀锡
	6ES7292-1AV30-0XA0	6ES7292-2AV30-0XA0	20 针，镀锡
CPU 1215C DC/DC/继电器 (6ES7215-1HG40-0XB0)	6ES7292-1BF30-0XB0	6ES7292-2BF30-0XB0	6 针，镀金
	6ES7292-1AM40-0XA0	6ES7292-2AM40-0XA0	12 针，镀锡，带键
	6ES7292-1AV30-0XA0	6ES7292-2AV30-0XA0	20 针，镀锡
CPU 1215C AC/DC/继电器 (6ES7215-1BG40-0XB0)	6ES7292-1BF30-0XB0	6ES7292-2BF30-0XB0	6 针，镀金
	6ES7292-1AM40-0XA0	6ES7292-2AM40-0XA0	12 针，镀锡，带键
	6ES7292-1AV40-0XA0	6ES7292-2AV40-0XA0	20 针，镀锡，带键
CPU 1217C DC/DC/DC (6ES7217-1AG40-0XB0)	6ES7292-1BF30-0XB0	6ES7292-2BF30-0XB0	6 针，镀金
	6ES7292-1AK30-0XA0	6ES7292-2AK30-0XA0	10 针，镀锡
	6ES7292-1AR30-0XA0	6ES7292-2AR30-0XA0	16 针，镀锡
	6ES7292-1AT30-0XA0	6ES7292-2AT30-0XA0	18 针，镀锡

表格 C-17　S7-1200 SM V3.2 及以上版本 - 端子排备件套件

如果使用 S7-1200 SM V3.2 及以上版本（订货号）	使用此端子排备用套件（每包 4 件）		
	端子排（螺钉型） 订货号	等效 推入式端子排 订货号	端子排 描述
SM 1221 DI 8 x DC (6ES7221-1BF32-0XB0)	6ES7292-1AG30-0XA0	6ES7292-2AG30-0XA0	7 针，镀锡
SM 1222 DQ 8 x DC (6ES7222-1BF32-0XB0)	6ES7292-1AG30-0XA0	6ES7292-2AG30-0XA0	7 针，镀锡
SM 1222 DQ 8 x 继电器 (6ES7222-1HF32-0XB0)	6ES7292-1AG40-0XA1	6ES7292-2AG40-0XA1	7 针，镀锡，带键（左侧）
电压输入的 SM 1238 电能表 480 V AC (6ES7238-5XA32-0XB0)（顶部）	6ES7292-1AG40-0XA2	6ES7292-2AG40-0XA2	7 针，镀锡，带键（中间）
电流输入的 SM 1238 电能表 480 V AC (6ES7238-5XA32-0XB0)（底部）	6ES7292-1AG30-0XA0	6ES7292-2AG30-0XA0	7 针，镀锡
SM 1231 AI 4 x 13 位 (6ES7231-4HD32-0XB0)	6ES7292-1BG30-0XA0	6ES7292-2BG30-0XA0	7 针，镀金
SM 1232 AQ 2 x 14 位 (6ES7232-4HB32-0XB0)	6ES7292-1BG30-0XA0	6ES7292-2BG30-0XA0	7 针，镀金
SM 1231 AI 4 x TC (6ES7231-5QD32-0XB0)	6ES7292-1BG30-0XA0	6ES7292-2BG30-0XA0	7 针，镀金
SM 1231 AI 4 x 16 位 (6ES7231-5ND32-0XB0)	6ES7292-1BG30-0XA0	6ES7292-2BG30-0XA0	7 针，镀金
SM 1221 DI 16 x DC (6ES7221-1BH32-0XB0)	6ES7292-1AG30-0XA0	6ES7292-2AG30-0XA0	7 针，镀锡
SM 1222 DQ 16 x DC (6ES7222-1BH32-0XB0)	6ES7292-1AG30-0XA0	6ES7292-2AG30-0XA0	7 针，镀锡
SM 1222 DQ 16 x 继电器 (6ES7222-1HH32-0XB0)	6ES7292-1AG40-0XA0	6ES7292-2AG40-0XA0	7 针，镀锡，带键（右侧）
SM 1223 DI 8 x DC/DQ 8 x DC (6ES7223-1BH32-0XB0)	6ES7292-1AG30-0XA0	6ES7292-2AG30-0XA0	7 针，镀锡

如果使用 S7-1200 SM V3.2 及以上版本（订货号）	使用此端子排备用套件（每包 4 件）		
	端子排（螺钉型） 订货号	等效 推入式端子排 订货号	端子排 描述
SM 1223 8 x DC/8 x 继电器 (6ES7223-1PH32-0XB0)	6ES7292-1AG30-0XA0	6ES7292-2AG30-0XA0	7 针，镀锡
	6ES7292-1AG40-0XA0	6ES7292-2AG40-0XA0	7 针，镀锡，带键（右侧）
SM 1223 8 x AC/8 x 继电器 (6ES7223-1QH32-0XB0)	6ES7292-1AG40-0XA0	6ES7292-2AG40-0XA0	7 针，镀锡，带键（右侧）
SM 1234 AI 4/AQ 2 (6ES7234-4HE32-0XB0)	6ES7292-1BG30-0XA0	6ES7292-2BG30-0XA0	7 针，镀金
SM 1231 AI 8 x 13 位 (6ES7231-4HF32-0XB0)	6ES7292-1BG30-0XA0	6ES7292-2BG30-0XA0	7 针，镀金
SM 1232 AQ 4 x 14 位 (6ES7232-4HD32-0XB0)	6ES7292-1BG30-0XA0	6ES7292-2BG30-0XA0	7 针，镀金
SM 1231 AI 4 x RTD (6ES7231-5PD32-0XB0)	6ES7292-1BG30-0XA0	6ES7292-2BG30-0XA0	7 针，镀金
SM 1231 AI 8 x TC (6ES7231-5QF32-0XB0)	6ES7292-1BG30-0XA0	6ES7292-2BG30-0XA0	7 针，镀金
SM 1278 IO LINK (6ES7278-4BD32 0XB0)	6ES7292-1AG30-0XA0	6ES7292-2AG30-0XA0	7 针，镀锡
SM 1222 DQ 8 x 继电器（切换） (6ES7222-1XF32-0XB0)	6ES7292-1AL30-0XA0	6ES7292-2AL30-0XA0	11 针，镀锡
SM 1223 DI 16 x DC/DQ 16 x DC (6ES7223-1BL32-0XB0)	6ES7292-1AL30-0XA0	6ES7292-2AL30-0XA0	11 针，镀锡
SM 1223 DI 16 x DC/DQ 16 x 继电器 (6ES7223-1PL32-0XB0)	6ES7292-1AL30-0XA0	6ES7292-2AL30-0XA0	11 针，镀锡
	6ES7292-1AL40-0XA0	6ES7292-2AL40-0XA0	11 针，镀锡，带键
SM 1231 AI 8 x RTD (6ES7231-5PF32-0XB0)	6ES7292-1BL30-0XA0	6ES7292-2BL30-0XA0	11 针，镀金

表格 C-18　S7-1200 SB、CB 和 BB - 端子排备件套件

如果使用 S7-1200 SB、CB 或 BB（订货号）	使用此端子排备用套件（每包 4 件）	
	端子排（螺钉型） 订货号	端子排 描述
SB 1221 DI 4 x 5 V DC (6ES7221-3AD30-0XB0)	6ES7292-1BF30-0XA0	6 针
SB 1221 DI 4 x 5 V DC (6ES7221-3AD30-0XB0)		
SB 1221 DI 4 x 24 V DC (6ES7221-3BD30-0XB0)		
SB 1222 DQ 4 x 5 V DC (6ES7222-1AD30-0XB0)		
SB 1222 DQ 4 x 24 V DC (6ES7222-1BD30-0XB0)		
SB 1223 DI 2x24 V DC/DQ 2x24 V DC (6ES7223-0BD30-0XB0)		
SB 1223 DI 2x5 V DC/DQ 2x5 V DC (6ES7223-3AD30-0XB0)		
SB 1223 DI 2x24 V DC/DQ 2x24 V DC (6ES7223-3BD30-0XB0)		
SB 1231 AI 1 x 12 位 (6ES7231-4HA30-0XB0)		
SB 1231 AI 1 x RTD (6ES7231-5PA30-0XB0)		
SB 1231 AI 1 x TC (6ES7231-5QA30-0XB0)		
SB 1232 AQ 1x12 位 (6ES7232-4HA30-0XB0)		
CB 1231 RS485 (6ES7241-1CH30-1XB0)		
BB 1297 电池 (6ES7297-0AX30-0XA0)		

表格 C-19　故障安全 CPU – 端子排备用套件

如果您拥有故障安全 CPU（产品编号）	使用此端子排备用套件（每包 4 件）	
	端子排 订货号	端子排 描述
CPU 1214FC DC/DC/DC (6ES7214-1AF40-0XB0)	6ES7292-1BC30-0XA0	3 针，镀金
	6ES7292-1AM30-0XA0	12 针，镀锡
	6ES7292-1AV30-0XA0	20 针，镀锡
CPU 1214FC DC/DC/继电器 (6ES7214-1HF40-0XB0)	6ES7292-1BC30-0XA0	3 针，镀金
	6ES7292-1AM40-0XA0	12 针，镀锡，带键
	6ES7292-1AV30-0XA0	20 针，镀锡

如果您拥有故障安全 CPU（产品编号）	使用此端子排备用套件（每包 4 件）	
	端子排 订货号	端子排 描述
CPU 1215FC DC/DC/DC (6ES7215-1AF40 0XB0)	6ES7292-1BF30-0XB0	6 针，镀金
	6ES7292-1AM30-0XA0	12 针，镀锡
	6ES7292-1AV30-0XA0	20 针，镀锡
CPU 1215FC DC/DC/继电器 (6ES7215-1HF40 0XB0)	6ES7292-1BF30-0XB0	6 针，镀金
	6ES7292-1AM40-0XA0	2 针，镀锡，带键
	6ES7292-1AV30-0XA0	20 针，镀锡

表格 C-20　故障安全信号模块 – 端子排备用套件

如果您拥有故障安全信号模块（产品编号）	使用此端子排备用套件（每包 4 件）	
	端子排 订货号	端子排 描述
SM 1226 F-DI (6ES7226-6BA32-0XB0)	6ES7292-1AL30-0XA0	11 针，镀锡
SM 1226 F-DQ (6ES7226-6DA32-0XB0)	6ES7292-1AL30-0XA0	11 针，镀锡
SM 1226 F 继电器 (6ES7226-6RA32-0XB0)	6ES7292-1AL40-0XA0	11 针，镀锡，带键

C.9 编程软件

表格 C-21　编程软件

SIMATIC 软件		订货号
编程软件	STEP 7 Basic V16	6ES7822-0AA06-0YA5
	STEP 7 Professional V16	6ES7822-1AA06-0YA5
可视化软件	WinCC Basic V16	6AV2100-0AA06-0AA5
	WinCC Comfort V16	6AV2101-0AA06-0AA5
	WinCC Advanced V16	6AV2102-0AA06-0AA5
	WinCC Professional 512 PowerTags V16	6AV2103-0DA06-0AA5
	WinCC Professional 4096 PowerTags V16	6AV2103-0HA06-0AA5
	WinCC Professional max. PowerTags V16	6AV2103-0XA06-0AA5

C.10 OPC UA 许可证

表格 C-22 S7-1200 的 OPC UA 许可证

OPC UA 许可证		订货号
SIMATIC 许可证	SIMATIC OPC UA S7-1200 Basic DVD	6ES7823-0BA00-2BA0
	SIMATIC OPC UA S7-1200 Basic DL	6ES7823-0BE00-2BA0

设备更换和备件兼容性

D

D.1 用 V4.x CPU 更换 V3.0 CPU

要将 V3.0 CPU 升级到 V4.x CPU，必须替换该 CPU 硬件。无法通过固件更新将 V3.0 CPU 升级到 V4.x CPU。

随后，在 STEP 7 项目中，可以用 V4.x CPU 替换 V3.0 CPU (页 162) 并使用为 V3.0 CPU 设计的现有 STEP 7 项目。

用 V4.x CPU 替换 V3.0 CPU 时，最好同时检查所连接的信号和通信模块的固件更新 (页 143)，并执行固件更新。

说明

在 STEP 7 中无法将设备从 V4.x 更换为 V3.0

可以用 V4.x CPU 更换 V3.0 CPU，但下载组态后，无法用 V3.0 CPU 更换 V4.x CPU。若要查看或使用现有的 STEP 7 V3.0 项目，在更换设备之前需先将您的 STEP 7 V3.0 项目归档。

请注意，如果您尚未下载替换之后的设备组态，您可以撤消替换。下载之后，便无法撤消从 V3.0 到 V4.x 的更换操作。

需要注意两个 CPU 版本之间的一些组态和运行区别：

更新 STEP 7 项目

不能直接将 STEP 7 V11 或 V12 项目升级为 STEP 7 V15。必须首先将这些项目升级至 STEP 7 V13 SP1 或 STEP 7 V13 SP2，然后基于这些项目升级至 STEP 7 V15。

⚠ **警告**

从旧版本 STEP 7 复制粘贴程序逻辑时的风险

从旧版本 STEP 7（例如 STEP 7 V12）复制程序逻辑到 STEP 7 V15 中可能导致程序执行出现意外或者编译失败。不同版本的 STEP 7 执行程序元素的方式不同。如果从旧版本粘贴内容到 STEP 7 V15 后进行了更改，编译器不会一直检测相关差异。如不修正程序，执行不可预测的程序逻辑可能导致严重的人员伤害甚至死亡。

使用 STEP 7 V15 以下版本的 STEP 7 中的程序逻辑时，必须将整个项目升级到 STEP 7 V15。随后，用户可以根据需要复制、剪切、粘贴和编辑程序逻辑。可以在 STEP 7 V15 中打开 STEP 7 V13 SP1 或更高版本的项目。随后，STEP 7 执行必要的兼容性转换并正确升级程序。为确保正确编译和执行程序，必须执行这类升级转换和修正。如果项目版本低于 STEP 7 V13 SP1，则必须将项目升级到 STEP 7 V15。

组织块

用户可以将 OB 执行组态为可间断或不间断 (页 99)。在 V3.0 CPU 项目中，STEP 7 将所有 OB 默认设置为不间断。

STEP 7 将所有 OB 属性 (页 99)设置为 V3.0 CPU STEP 7 项目中的相应值。

随后可根据需要更改中断或优先级设置。

如果没有未决诊断事件，诊断错误中断 OB (页 93) 启动信息将完全参考子模块。

CPU 密码保护

STEP 7 会将 V4.x CPU 的密码保护等级 (页 198)设置为与 V3.0 CPU 相同的密码保护等级，并将 V3.0 密码指定为 V4.x CPU 的“完全访问（无保护）”密码：

V3.0 保护级别	V4.x 中的访问等级
无保护	完全访问（无保护）
写保护	读访问
写/读保护	HMI 访问

请注意，对于 V3.0，不存在 V4.x 的“无访问（完全保护）”访问等级。

Web 服务器

如果在 V3.0 项目中使用用户定义 Web 页面，那么在升级项目前，需要将项目安装文件夹中的项目存储在子文件夹“UserFiles\Webserver”下。如果将用户定义页面存储在这个位置，则保存 STEP 7 项目时还将保存用户定义 Web 页面。

如果将 V3.0 CPU 更换为 V4.x CPU，用于激活 Web 服务器的 Web 服务器项目设置 (页 1028)和 HTTPS 设置会与版本 V3.0 中的相同。然后用户可以按需组态用户、权限、密码 (页 1029)和语言 (页 1028)以使用 Web 服务器。如果不赋予用户更多权限，将只能查看标准 Web 页面 (页 1036)中的内容。S7-1200 V4.x CPU 不支持之前预组态的“admin”用户和密码。

S7-1200 V3.0 Web 服务器数据日志页面提供 “下载并清除”(Download and Clear) 操作。用于访问数据日志的 V4.x Web 服务器文件浏览器页面 (页 1065)不再提供这一功能。不过，该 Web 服务器提供了下载、重命名和删除数据日志文件的功能。

传送卡不兼容

无法使用 V3.0 传送卡 (页 134)将 V3.0 程序传送到 V4.x CPU。用户必须在 STEP 7 中打开 V3.0 项目，将设备更换为 V4.x CPU (页 162)，然后将 STEP 7 项目下载到用户 V4.x CPU 中。将项目更改为 V4.x 项目后，即可创建一个 V4.x 传送卡，以便执行后续的程序传送。

GET/PUT 通信

S7-1200 V3.0 CPU 默认启用 GET/PUT 通信。用 V4.x CPU 替换 V3.0 CPU (页 162) 时，可以在兼容性信息部分看到一条消息，指示 GET/PUT 已启用。

运动控制支持

S7-1200 V4.x CPU 不支持 V1.0 和 V2.0 运动控制库。如果为具有 V1.0 或 V2.0 运动控制库的 STEP 7 项目更换设备，则设备替换编译过程中会使用兼容的 V3.0 运动控制指令 (页 732)替换 V1.0 或 V2.0 运动控制库指令。

对于包含两种不同运动控制指令版本（V3.0 和 V5.0）的 STEP 7 项目，若将设备从 V3.0 CPU 更换为 V4.x CPU，则设备更换编译过程中会使用兼容的 V5.0 运动控制指令 (页 732)。

在设备从 V3.0 CPU 更换为 V4.x CPU 的过程中，运动控制工艺对象 (TO) 版本不会自动从 V3.0 更改为 V5.0。如果要升级到更高版本，必须转至指令树并为项目选择所需的 S7-1200 运动控制版本，如下表中所示：

CPU 版本	允许的运动控制版本
V4.3（运动控制 V5.0）	V6.0、V5.0、V4.0 或 V3.0
V4.2.x（运动控制 V5.0）	V6.0、V5.0、V4.0 或 V3.0
V4.1（运动控制 V5.0）	V5.0、V4.0 或 V3.0
V4.0（运动控制 V4.0）	V4.0 或 V3.0
V3.0（运动控制 V3.0）	V3.0

运动控制版本 V3.0 和 V5.0 的 TO 结构不同。所有相关的块也会更改。块接口、监控表以及跟踪都将更新为新的运动控制 V5.0 结构。有关 V3.0 CPU 和 V4.x CPU 运动控制轴参数的区别，请参见以下两个表格：

V3.0 CPU （运动控制 V3.0）	V4.x CPU （运动控制 V5.0）
Config.General.LengthUnit	Units.LengthUnit
Config.Mechanics.PulsesPerDriveRevolution	Actor.DriveParameter.PulsesPerDriveRevolution
Config.Mechanics.LeadScrew	Mechanics.LeadScrew
Config.Mechanics.InverseDirection	Actor.InverseDirection
Config.DynamicLimits.MinVelocity	DynamicLimits.MinVelocity
Config.DynamicLimits.MaxVelocity	DynamicLimits.MaxVelocity
Config.DynamicDefaults.Acceleration	DynamicDefaults.Acceleration

V3.0 CPU （运动控制 V3.0）	V4.x CPU （运动控制 V5.0）
Config.DynamicDefaults.Deceleration	DynamicDefaults.Deceleration
Config.DynamicDefaults.EmergencyDeceleration	DynamicDefaults.EmergencyDeceleration
Config.DynamicDefaults.Jerk	DynamicDefaults.Jerk
Config.PositionLimits_SW.Active	PositionLimitsSW.Active
Config.PositionLimits_SW.MinPosition	PositionLimitsSW.MinPosition
Config.PositionLimits_SW.MaxPosition	PositionLimitsSW.MaxPosition
Config.PositionLimits_HW.Active	PositionLimitsHW.Active
Config.PositionLimits_HW.MinSwitchedLevel	PositionLimitsHW.MinSwitchLevel
Config.PositionLimits_HW.MaxSwitchedLevel	PositionLimitsHW.MaxSwitchLevel
Config.Homing.AutoReversal	Homing.AutoReversal
Config.Homing.Direction	Homing.ApproachDirection
Config.Homing.SideActiveHoming	Sensor[1].ActiveHoming.SideInput
Config.Homing.SidePassiveHoming	Sensor[1].PassiveHoming.SideInput
Config.Homing.Offset	Sensor[1].ActiveHoming.HomePositionOffset
Config.Homing.FastVelocity	Homing.ApproachVelocity
Config.Homing.SlowVelocity	Homing.ReferencingVelocity
MotionStatus.Position	Position
MotionStatus.Velocity	Velocity
MotionStatus.Distance	StatusPositioning.Distance
MotionStatus.TargetPosition	StatusPositioning.TargetPosition
StatusBits.SpeedCommand	StatusBits.VelocityCommand
StatusBits.Homing	StatusBits.HomingCommand

唯一重新命名的“CommandTable”参数是具有以下命令的数组：

V3.0	V4.x
Config.Command[]	Command[]

注：“Command[]”数组在 V3.0 中是“TO_CmdTab_Config_Command”类型的 UDT，在 V4.x 中则是“TO_Struct_Command”类型的 UDT。

指令变化

以下指令在参数或特性方面发生了变化：

- RDREC 和 WRREC (页 387)
- CONV (页 293)

HMI 面板通信

如果 S7-1200 V3.0 CPU 连有一个或多个 HMI 面板 (页 35)，则与 S7-1200 V4.x CPU 的通信将取决于使用的通信类型以及 HMI 面板的固件版本。重新编译项目，然后将其下载到 CPU 和 HMI，并/或更新 HMI 固件。

重新编译程序块的相关要求

用 V4.x CPU 更换 V3.0 CPU 后，必须重新编译所有程序块，之后才能将其下载到 V4.x CPU。此外，如果任意块采用专有技术保护 (页 201)或与某个 PLC 序列号绑定的复制保护 (页 202)，您必须先取消保护，然后再编译和下载块。（不过，不需要取消激活与某个存储卡绑定的复制保护。）成功编译后，可以重新组态专有技术保护和/或 PLC 序列号防拷贝保护。请注意，如果项目中的任意块采用了 OEM（原始设备制造商）所提供的专有技术保护，则必须联系 OEM 来获取这些块的 V4.x 版本。

更换设备后，Siemens 通常建议您先在 STEP 7 中重新编译硬件组态和软件，然后再将其下载到项目中的所有设备。纠正编译项目时发现的所有错误，并进行重新编译，直到没有任何错误为止。然后，才可将项目下载至 V4.x CPU。

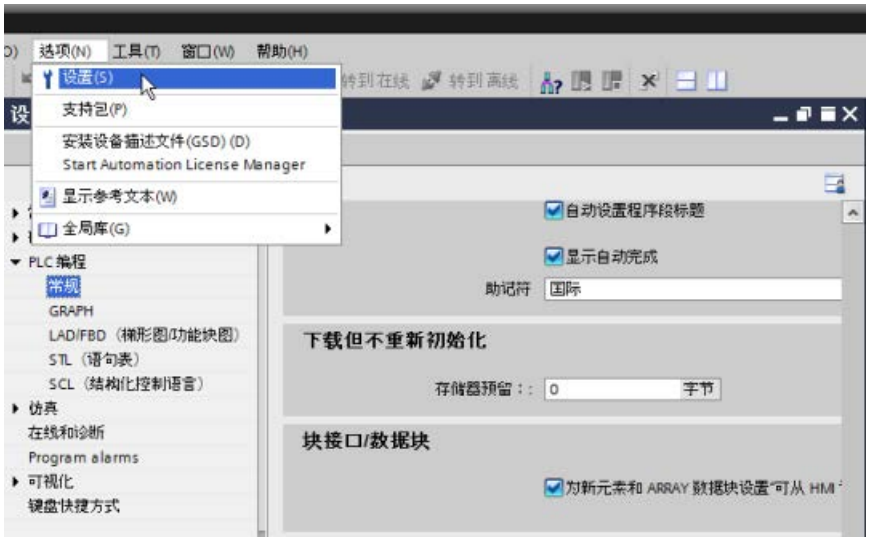
S7-1200 V3.0 项目可能不适用于 S7-1200 V4.x CPU

S7-1200 V4.0 及更高版本在每个 DB 中均增加了一个 100 字节的预留区域，以支持下载而不重新初始化。

在将 V3.0 项目下载至 V4.x CPU 之前，可以从数据块中移除 100 字节的预留区域。

若要删除 100 字节的预留区域，在执行设备更换之前，请执行以下步骤：

1. 在 TIA Portal 主菜单中，选择“选项 > 设置”(Options > Settings) 菜单命令。
2. 从导航树中打开“PLC 编程 > 常规”(PLC programming > General) 节点。
3. 在“下载而不重新初始化”(Download without reinitialization) 区域，将存储器预留区域设置为 0 字节。



如果已执行设备更换，则必须分别从各个块中删除 100 字节的预留区域：

1. 在项目树中，右键单击“程序块”(Program blocks) 文件夹中的数据块，并在快捷菜单中选择“属性”(Properties)。
2. 在“数据块属性”(Data block properties) 对话框中，选择“下载而不重新初始化”(Download without reinitialization) 节点。
3. 将存储器预留区域设置为 0 字节。
4. 对项目中的每个数据块，均重复以上步骤。

D.2 S7-1200 V3.0 及更早版本的端子排备件套件

表格 D-1 S7-1200 CPU V3.0 及更早版本 – 端子排备件套件

如果您拥有 S7-1200 CPU V3.0 及更早版本（产品编号）	使用此端子排备用套件（每包 4 件）	
	端子排产品编号	端子排描述
CPU 1211C DC/DC/DC (6ES7211-1AE31-0XB0)	6ES7292-1BC30-0XA0 6ES7292-1AH30-0XA0 6ES7292-1AP30-0XA0	3 针，镀金 8 针，镀金 14 针，镀锡
CPU 1211C AC/DC/继电器 (6ES7211-1BE31-0XB0)		
CPU 1211C DC/DC/继电器 (6ES7211-1HE31-0XB0)		
CPU 1212C DC/DC/DC (6ES7212-1AE31-0XB0)		
CPU 1212C AC/DC/继电器 (6ES7212-1BE31-0XB0)		
CPU 1212C DC/DC/继电器 (6ES7212-1HE31-0XB0)		
CPU 1214C DC/DC/DC (6ES7214-1AG31-0XB0)	6ES7292-1BC30-0XA0 6ES7292-1AM30-0XA0 6ES7292-1AV30-0XA0	3 针，镀金 12 针，镀锡 20 针，镀锡
CPU 1214C AC/DC/继电器 (6ES7214-1BG31-0XB0)		
CPU 1214C DC/DC/继电器 (6ES7214-1HG31-0XB0)		
CPU 1215C DC/DC/DC (6ES7215-1AG31-0XB0)	6ES7292-1BF30-0XB0 6ES7292-1AM30-0XA0 6ES7292-1AV30-0XA0	6 针，镀金 12 针，镀锡 20 针，镀锡
CPU 1215C AC/DC/继电器 (6ES7215-1BG31-0XB0)		
CPU 1215C DC/DC/继电器 (6ES7215-1HG31-0XB0)		

表格 D-2　S7-1200 SM V3.0 及更早版本 - 端子排备件套件

如果您拥有	使用此端子排备用套件（每包 4 件）	
S7-1200 SM V3.0 及更早版本（产品编号）	端子排产品编号	端子排描述
SM 1221 DI 8 x DC (6ES7221-1BF30-0XB0)	6ES7292-1AG30-0XA0	7 针，镀锡
SM 1222 DQ 8 x DC (6ES7222-1BF30-0XB0)		
SM 1222 DQ 8 x 继电器 (6ES7222-1HF30-0XB0)		
SM 1231 AI 4 x 13 位 (6ES7231-4HD30-0XB0)	6ES7292-1BG30-0XA0	7 针，镀金
SM 1232 AQ 2 x 14 位 (6ES7232-4HB30-0XB0)		
SM 1231 AI 4 x TC (6ES7231-5QD30-0XB0)		
SM 1231 AI 4 x 16 位 (6ES7231-5ND30-0XB0)		
SM 1221 DI 16 x DC (6ES7221-1BH30-0XB0)	6ES7292-1AG30-0XA0	7 针，镀锡
SM 1222 DQ 16 x DC (6ES7222-1BH30-0XB0)		
SM 1222 DQ 16 x 继电器 (6ES7222-1HH30-0XB0)		
SM 1223 DI 8 x DC/DQ 8x DC (6ES7223-1BH30-0XB0)		
SM 1223 8 x DC/8 x 继电器 (6ES7223-1PH30-0XB0)		
SM 1223 8 x AC/8 x 继电器 (6ES7223-1QH30-0XB0)		
SM 1234 AI 4 / AQ 2 (6ES7234-4HE30-0XB0)	6ES7292-1BG30-0XA0	7 针，镀金
SM 1231 AI 8 x 13 位 (6ES7231-4HF30-0XB0)		
SM 1232 AQ 4 x 14 位 (6ES7232-4HD30-0XB0)		
SM 1231 AI 4 x RTD (6ES7231-5PD30-0XB0)		
SM 1231 AI 8 x TC (6ES7231-5QF30-0XB0)		
SM 1222 DQ 8 x 继电器（切换） (6ES7222-1XF30-0XB0)	6ES7292-1AL30-0XA0	11 针，镀锡
SM 1223 DI 16 x DC/DQ 16 x DC (6ES7223-1BL30-0XB0)		
SM 1223 16 x DC/16 x 继电器 (6ES7223-1PL30-0XB0)		
SM 1231 AI 8 x RTD (6ES7231-5PF30-0XB0)	6ES7292-1BL30-0XA0	11 针，镀金

索引

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

X

安

变

标

波

捕

不

布

采

参

操

测

插

存

错

代

灯

登

等

地

点

电

订

定

丢

动

读

端

多

额

发

阀

反

返

防

仿

访

分

服

复

高

格

更

工

功

共

固

故

关

规

轨

过

海

函

韩

环

恢

机

基

级

计

技

继

加

间

监

兼

检

将

阶

接

结

介

紧

绝

开

看

可

客

块

扩

浪

冷

例

连

联

临

灵

流

路

轮

逻

滤

脉

门

密

面

模

目

暖

排

配

片

频

平

其

奇

启

气

嵌

强

切

区

驱

全

热

认

任

日

冗

软

三

扫

上

设

生

时

示

事

收

手

首

输

数

刷

特

添

网

未

位

文

污

西

系

下

夏

显

限

线

相

向

项

消

协

新

信

性

修

许

选

寻

循

延

遥

要

一

移

已

以

引

硬

用

优

与

语

远

运

在

早

站

诊

支

指

智

中

重

主

专

转

装

状

准

子

自

字

总

组

最