SIEMENS

SIMATIC

S7-300/S7-400 用于点对点 CP 的可加载驱动程序：MODBUS 协议，RTU 格式，S7 为主站

操作说明

09/2009

A5E02294897-06

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
带有警告三角，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。

小心
不带警告三角，表示如果不采取相应的小心措施，可能导致财产损失。

注意
表示如果不注意相应的提示，可能会出现不希望的结果或状态。

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。

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 ® 的都是西门子股份有限公司的注册商标。标签中的其他符号可能是一些其他商标，这是出于保护所有者权利的 目地由第三方使用而特别标示的。

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
Industry Sector
Postfach 48 48
90026 NÜRNBERG
德国

A5E02294897-06
Ⓟ 05/2011

目录

前言

1

本手册的用途

本手册提供的信息可帮助您建立 CP（作为“具有 modbus 功能”的主站）和 Modbus 从站控制系统之间的数据链接，并调试该链接。

所需的基本知识

使用本手册需要具备自动化工程的基本知识。

此外，还应该了解如何使用装有 Microsoft® Windows ® 操作系统的计算机或具有相似功能的设备（例如，编程设备），并且应该对 STEP 7 编程有一些了解。

本手册内容

本手册适用于以下软件：

产品	订货号	起始版本
用于点对点 CP 的可加载驱动程序	6ES7870-1AA01-0YA0	3.0

说明

手册包含了驱动程序和功能块的描述，自出版发行起生效。

指南

本手册描述了可加载驱动程序的功能及其到 CP 341 和 CP 441-2 通信处理器的硬件和软件的集成。

本手册包括以下主题：

- 产品描述/安装
- 调试驱动程序/安装/参数分配
- CPU – CP 接口
- 传输协议/功能代码
- 驱动程序的诊断
- 应用示例

约定

本手册使用通用术语 CP，或者 CP 341 和 CP 441-2。

特别说明

本手册中描述的驱动程序可用作 CP 的可加载协议，用它来替换 3964R、RK512、ASCII 和打印机标准协议。

说明

使用此驱动程序可以对 CP 和 CPU 之间的通信顺序进行修改或扩展。

特别是可以对诊断的现有事件类别和事件编号进行修改和扩展。

另外注意，本手册仅描述了相对于标准功能的修改和扩充。 可以在所使用的 CP 的手册中找到所有基本信息。

为了确保安全使用本驱动程序，您应该具备如何使用 CP 功能的详细知识。

技术支持

可以使用技术支持请求的 Web 表单获得所有工业自动化产品的技术支持，Web 表单的网址为：
http://www.siemens.com/automation/support-request。

关于我们的技术支持方面的更多信息，请访问 Internet 网址：
http://www.siemens.de/automation/service。

因特网上的“服务与支持”

除文档外，我们还提供了在线技术知识，网址为：

http://www.siemens.com/automation/service&support

在此可以找到：

- 不断为您提供产品最新信息的新闻快递。
- 所需文档，可以使用产品支持搜索功能找到。
- 论坛，世界各地的用户和专家在此交流想法。
- 当地的工业自动化合作伙伴。
- 有关维修、备件和咨询的信息。

更多支持

如果您对使用本手册中介绍的产品还有疑问，且在手册中未找到正确的答案，请联系当地西门子代表：

可在以下网站找到有关联系人的信息：

http://www.siemens.com/automation/partner

我们为各 SIMATIC 产品和系统提供的技术文档的向导位于：

http://www.siemens.com/simatic-tech-doku-portal

在线目录和订购系统位于：

http://mall.automation.siemens.com

培训中心

我们提供了一系列课程，来帮助您熟悉 SIMATIC S7 自动化系统。 请联系当地的培训中心或位于德国纽伦堡的培训中心总部，以获取详细资料，网址为：

http://www.sitrain.com

产品说明 2

2.1 应用程序选件

在系统环境中的位置

该驱动程序是用于 CP 341 (S7-300) 和 CP 441-2 (S7-400) 通信处理器的软件产品。

CP 341 和 CP 441-2 可用于 S7 自动化系统，用于建立与伙伴系统之间的串行通信链接。

驱动程序的功能

本驱动程序可帮助建立 CP 341 或 CP 441-2 通信模块和“具有 Modbus 功能”的控制系统（比如：Modicon 控制器或者 Honeywell TDC 3000）之间的通信链接。

使用的传输协议为 **RTU 格式** 的 **GOULD - MODBUS 协议**。 数据传输依照主站-从站原则执行。
在传输过程中**主站** (SIMATIC S7) 具有主动权。

功能代码 01、02、03、04、05、06、07、08、11、12、15 和 **16** 可用于 CP 和 主机系统之间的通信。

可用的接口和协议

CP 441-2 的两个串行接口可以采用不同的标准协议或者可加载协议，其操作相互独立。

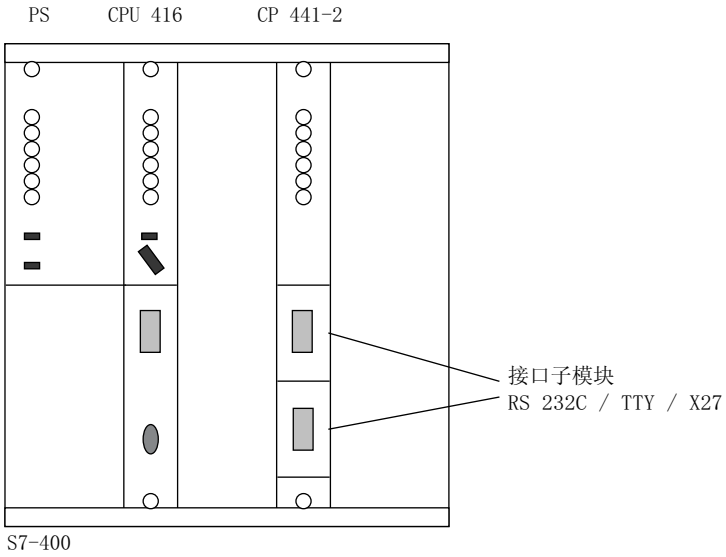
可以将 RS232C、TTY 或者 RS422/485 (X27) 用作 CP 的接口。

使用本驱动程序，可以在 2 线和 4 线操作模式下使用 RS422/485 (X27) 接口子模块。

在 2 线操作模式下，如果为半双工操作模式，那么一个主站最多可以连接 32 个从站，从而创建了一个多点连接，一个网络。

允许的系统组态

下图显示了一种允许的系统组态的原理示意图。



数据一致性

S7 CPU 和 CP 之间的数据交换通过集成的系统功能逐块进行。

另请参考本手册的“CPU-CP 接口 (页 81)”部分。

2.2 硬件和软件需求

可用的模块

本驱动程序可运行在 CP 341 和部件编号为 MLFB 6ES7441-2AA02-0AE0 或更新版本的 CP 441-2 上。

部件编号为 MLFB 6ES7 441-1AA0x-0AE0 的 CP 441-1 和部件编号为 MLFB 6ES7 441-2AA00-0AE0 或 6ES7 441-2AA01-0AE0 的 CP 441-2 不能与可加载驱动程序一起使用。

软件狗

要使用具有可加载驱动程序的 CP，需要有软件狗。 软件狗和驱动程序一起提供。

CPU 的装载存储器（存储卡）

使用 CP 441-2 时，可以基于其参数分配将可加载驱动程序装载到 CPU 的装载存储器中，并在 CPU 启动时传送到 CP 存储器中。

因此，CPU 必须有足够的装载存储空间。 要实现该目的需要有一个 RAM 或者 FLASH **存储卡**，其部件编号为 MLFB 6ES7952-…。

分配了可加载驱动程序的**每个** CP 接口大约需要 25 KB 的 CPU 装载存储器。

当使用 CP 341 时，将可加载驱动程序直接装载到 CP 341。因此 S7-300 CPU 上的装载存储器不是必需的。 但是注意，在没有编程设备的情况下将不能更换模块。

软件发布状态

已安装 STEP 7 Basis V5.3 或更高版本。

已安装为点对点连接分配参数的可选软件包 CP PtP Param V5.1 或更高版本。

数据结构

在组态 S7 数据结构之前，应该确保其与 MODBUS 从站系统的用户程序兼容。 说明将使用哪些功能代码和 Modbus 地址。

2.3 GOULD-MODBUS 协议概述

功能代码

MODBUS 系统之间的数据交换类型是由功能代码 (FC) 控制的。

数据交换

下列的 FC 可用于执行 **逐位**的数据交换：

- FC 01 读线圈（输出）状态，
- FC 02 读输入状态，
- FC 05 写单个线圈，
- FC 15 写多个线圈。

下列的 FC 可用于执行 **逐个寄存器** 的数据交换：

- FC 03 读保持寄存器，
- FC 04 读输入寄存器，
- FC 06 预置单个寄存器，
- FC 16 预置多个寄存器。

数据区域

作为规则，各 FC 的操作将依照下表进行：

功能代码	数据	数据类型		访问类型
01, 05, 15	线圈（输出）状态	位	输出	读/写
02	输入状态	位	输入	只读
03, 06, 16	保持寄存器	寄存器（16 位）	输出寄存器	读/写
04	输入寄存器	寄存器（16 位）	输入寄存器	只读

地址表示

和划分成读/写区和只读区相似，用户层级的数据可用下表显示的方法表示：

功能代码	数据类型	用户层级的地址表示（十进制）
01, 05, 15	输出位	**0**xxxx
02	输入位	**1**xxxx
04	输入寄存器	**3**xxxx
03, 06, 16	保持寄存器	**4**xxxx

在串行传输线上的**传输消息帧**中，MODBUS 用户系统使用的地址以 **0** 为基准。

在 **MODBUS 用户系统**里，这些地址从 **1** 开始计算！

示例

- 用户系统的第一个保持寄存器表示为寄存器 **4**0001。在传输消息帧中，当使用 FC 03、06 或 16 时，十六进制值 0000 作为寄存器地址传送。
- 用户系统中的第 127 个线圈表示为线圈 **0**0127，并且在传输消息帧中为该线圈分配的地址为十六进制的 007E。

安装

3

3.1 使用软件狗

简介

要使用具有可加载驱动程序的 CP，需要有软件狗。 当插入软件狗时，可加载驱动程序。对于 CP 441-2，可为两个接口下载驱动程序。

插入软件狗

要插入软件狗，需要将 CP 从机架上移除。 插入软件狗的模块插槽位于 CP 的背面，在背板总线可插拔连接器的上方。

3.2 接口连接

RS 232C / TTY

可以创建和从站系统之间的点对点连接。

更多关于接口连接的信息可在手册《CP 341 或 CP 441-2 点对点通信》中找到。

X27/RS485（2 线）

可以直接创建多点连接和网络，在一个主站系统中最多可以连接 32 个从站。

CP 的驱动程序将会使接收的 2 线制线路在发送和接收之间切换。

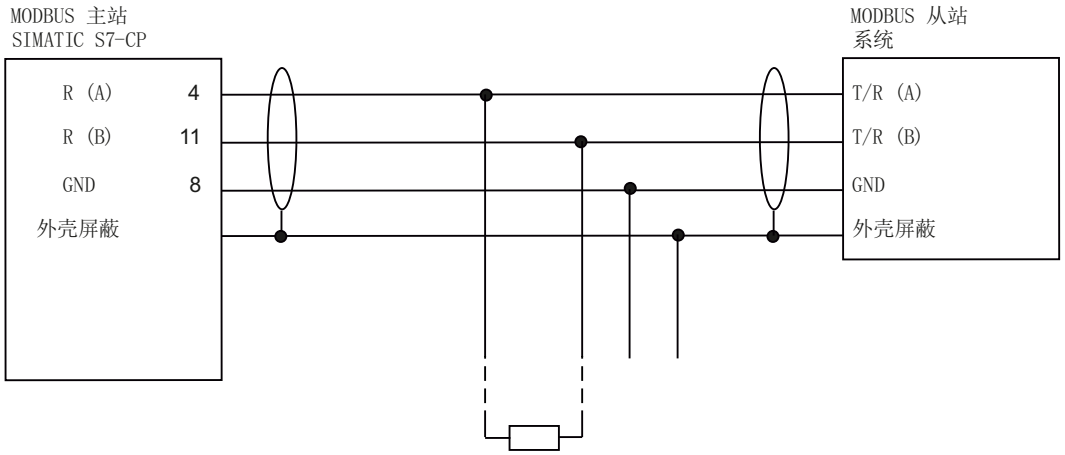


图 3-1 连接示意图： 1 个主站系统，1 个从站系统在总线上

更多关于接口连接的信息可在手册《CP 341 或 CP 441-2 点对点通信》中找到。

X27/RS485（4 线）

可以创建和从站系统之间的点对点连接。

如果 MODBUS 从站系统的硬件支持多点连接，则可以直接创建与多个从站的多点连接（即网络）。 MODBUS 从站系统必须能够在其发送器不发送数据时将它们切换为高阻态。

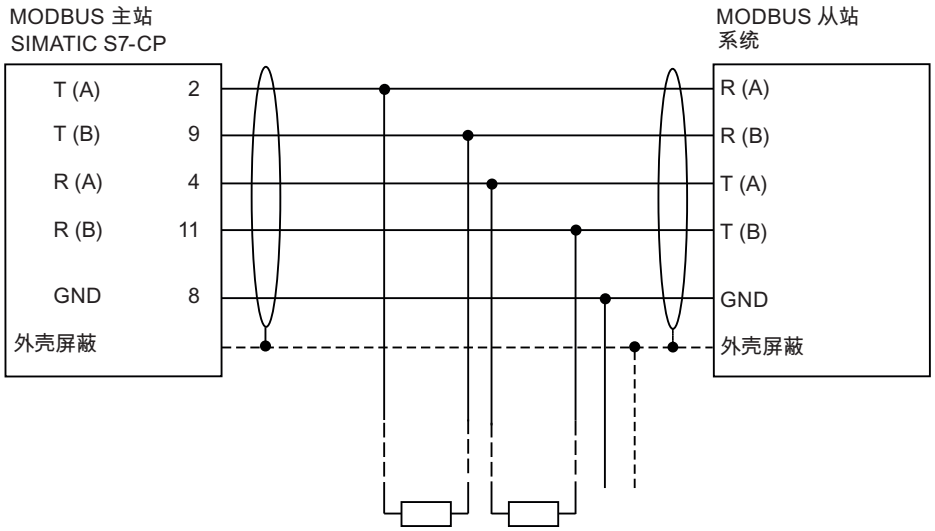


图 3-2　　连接示意图： 1 个主站系统，1 个从站系统

有关接口连接的更多信息，请参考“多点接线图 (页 143)”部分以及手册《CP 341 或 CP 441-2 点对点通信》。

调试驱动程序

4

4.1 调试驱动程序

常规信息

下述与 STEP 7 相关的任务基于 STEP 7 V5.3。

序列、名称和目录信息可能不同于后续的版本。

4.2 在 STEP 7 编程设备/PC 中安装驱动程序

程序

要安装由驱动代码和驱动程序特定屏蔽文件组成的驱动程序，请按下列步骤执行：

1. 将 MODBUS 主站 CD 插入到 CD-ROM 驱动器中。
2. 在 Windows 中，双击“控制面板”(Control Panel) 中的“添加和删除程序”(Add and Remove Programs) 图标启动安装软件对话框。
3. 在对话框中，选择 CD-ROM 驱动器然后选择 **setup.exe** 文件，启动安装过程。
4. 按照安装程序显示的说明逐步操作。

结果：驱动程序和参数分配屏幕窗体将安装到以下目录： **Step7\S7fptp\S7Driver**.

该目录包含了除此之外的下列文件：

- S7wfpa1a.dll
- S7wfpa1x.cod
- S7wfpa2x.cod

4.3 卸载驱动程序

程序

在 Windows 中使用“控制面板” (Control Panel)，“添加/删除软件”(Add/Remove Software) 和“卸载”(Uninstall)，可以将驱动程序从 STEP 7 软件包中卸载。

随后，您可以检查 Step7\S7fptp\S7Driver 目录下名称为 S7wfpb1?.*、S7wfpb2?.*、S7wfpb3?.* 的文件是否全部被删除。

说明

在卸载软件包“**参数分配工具 CP： 为点对点连接分配参数**”(Parameter Assignment Tool CP: Assigning Parameters to Point-To-Point Connections) 之前，需要先卸载所有的可加载驱动程序。

4.4 组态数据链接

4.4.1 组态数据链接

简介

组态数据链接包含使用 HW config 在组态表中放置硬件。 可以使用 STEP 7 软件进行数据链接组态。

4.4.2 使用 CP 341 组态数据链接

S7 项目

在进行组态之前，必须使用 STEP 7 创建一个 **S7 项目**。

项目组件

使用 SIMATIC 管理器将所需的项目组件插入到打开的项目中：

- SIMATIC 300 站

  在每次插入组件之前，必须单击选择所需的项目。

  插入 > 站 > SIMATIC 300 站 (Insert > Station > SIMATIC 300 station)

硬件配置

硬件配置包含定义硬件组件本身及其属性。

要启动硬件配置，选择 SIMATIC 300 站并双击“硬件”(Hardware)，或者选择菜单命令“编辑 > 打开项目”(Edit > Open Object)。

使用菜单命令 “插入 > 硬件组件”(Insert > Hardware Components) 插入下列组件：

- 从 SIMATIC 300 中选择一个 RACK-300，一个 PS-300 和一个 CPU-300
- 从 CP-300 中选择合适订货号的 CP PtP。

STEP 7 的用户手册中详细介绍了如何组态 S7-300 模块。

4.4.3 使用 CP 441-2 组态数据链接

简介

对于点对点数据链接，必须先配置 SIMATIC 400 站、连接伙伴站、PtP 节点和 PtP 网络。

S7 项目

在进行组态之前，必须使用 STEP 7 创建一个 **S7 项目**。

项目组件

使用 SIMATIC 管理器将所需的项目组件插入到打开的项目中：

- SIMATIC 站，其它站，PtP 网络。

  在每次插入组件之前，必须单击选择所需的项目。

  – 使用 **插入 > 站 > SIMATIC 400 站 (Insert > Station > SIMATIC 400 Station)** 创建自己的 S7 程序（机架、电源、CPU、CP 441-2, ...），

  – 使用 **插入 > 站 > 其它站 (Insert > Station > Other Station)** 创建数据连接伙伴，

  – 使用 **插入 > 子网 > PtP (Insert > Subnet > PtP)** 创建一个 SIMATIC 400 站和数据连接伙伴之间的 PtP 网络。

硬件配置 (Hardware Configuration)

硬件配置包含定义硬件组件本身及其属性。

要启动硬件配置，选择 SIMATIC 400 站并双击“硬件”(Hardware)，或者选择菜单命令“编辑 > 打开对象”(Edit > Open Object)。

使用菜单命令 “插入 > 硬件组件”(Insert > Hardware Components) 插入下列组件：

- 从 SIMATIC 400 中选择一个 RACK-400，一个 PS-400 和一个 CPU-400
- 从 CP-400 中选择合适订货号的 CP PtP。

关于如何组态 S7 400 模块的详细描述，请参见 STEP 7 的用户手册。

4.5 为 CP 分配参数

为 CP 分配参数

使用“硬件配置”(Hardware Configuration) 在机架上放置好模块后，必须为它们分配参数。

要启动参数分配工具，在“硬件配置”(Hardware Configuration) 中双击 CP，或者单击 CP 并选择菜单命令 **编辑 > 对象属性 (Edit > Object Properties)**。

4.5.1 为 CP 341 分配参数

步骤如下：

1. 属性 — CP > 基本参数 (Properties - CP > Basic Parameters)

   单击“**参数**”(Parameters) 按钮（单击）打开协议选择接口“**为点对点连接分配参数**”(Assigning Parameters to Point-To-Point Connections)。 这里可以选择所需的传输协议。

   选择 **协议** (protocol) 后，就可以执行**驱动程序的参数分配 (Parameter Assignment of the Driver)**。 双击字母符号启动。

   关于如何为可加载驱动程序，选择协议和为对话框分配参数的详细描述可从“为可加载驱动程序分配参数 (页 30)”部分中找到。

   参数分配完成后，返回到“**属性 CP**”(Properties - CP) 对话框。

2. 属性 — CP > 地址 (Properties - CP > Addresses)

   在“属性 - CP”(Properties - CP) 对话框中的“**地址**”(Addresses) 选项卡中**无需**任何设置。

3. 属性 — CP > 常规 (Properties - CP > General)

   在“属性 - CP”(Properties - CP) 对话框中的“**常规**”(General) 选项卡中**无需**任何设置。

   通过在“属性 — CP”(Properties - CP) 对话框中单击“确定”完成 CP 的参数分配。 然后返回到“硬件配置”(Hardware Configuration) 对话框。

   保存参数分配并且关闭“硬件配置”(Hardware Configuration) 对话框。 返回到 STEP 7 项目的主菜单。

4.5.2 为 CP 441-2 分配参数

步骤如下：

1. 属性 — CP 441-2 > 基本参数 (Properties - CP 441-2 > Basic Parameters)

   在“**基本参数**“(Basic Parameters) 选项卡中指定 CP 441 模块所需的“**接口**”(interface)（1=上面的接口, 2=下面的接口）。 选择已插入的接口子模块作为“**模块**”(Module)。

   单击“**参数**”(Parameters) 按钮（单击）打开协议选择接口“**为点对点连接分配参数**”(Assigning Parameters to Point-To-Point Connections)。

   这里可以选择所需的传输协议。

   选择 **协议** (protocol) 后，就可以执行**驱动程序的参数分配** (Parameter Assignment of the Driver)。 双击字母符号启动。

   关于如何为可加载驱动程序，选择协议和为对话框分配参数的详细描述可从“为驱动程序分配参数”部分中找到.

   参数分配完成后，返回到“**属性 - CP 441-2**”(Properties - CP 441-2) 对话框。

2. 属性 > CP 441-2 > 地址 (Properties > CP 441-2 > Addresses)

   在“属性 - CP 441-2”(Properties - CP 441-2) 对话框中的“**地址**”(Addresses) 选项卡中**无需**任何设置。

3. 属性 > CP 441-2 > 常规 (Properties > CP 441-2 > General)

   在“属性 - CP 441-2”(Properties - CP 441-2) 对话框中的“**常规**”(General) 选项卡中，可以指定 CP 的接口连接哪个 **PtP 网络** (PtP network)。

   PtP(1) 对应 CP 上面的接口，PtP(2) 对应 CP 下面的接口。

   单击 **PtP(1)** 或 **PtP(2)** 按钮打开用于组态子网的对话框。

   选择所需的**子网** (subnet) 并且激活复选框“伙伴已连接到所选的网络”(Partner is connected to the selected network)。

   所选的子网表示 CP 接口和连接伙伴接口之间的连接。

   单击“确定”返回到“属性 — CP 441-2”(Properties - CP 441-2) 对话框。 这里单击“确定”完成 CP 的参数分配，并且返回到“硬件配置”(Hardware Configuration) 对话框。

   保存参数分配并且关闭“硬件配置”(Hardware Configuration) 对话框。 返回到 STEP 7 项目的主菜单。

为连接伙伴分配参数

在将连接伙伴站插入到 STEP 7 项目中后，如“项目组件： 插入 > 其它站”(Project Components:Insert > Other Station) 中所描述，将需要给该伙伴站点指定对象属性。

从打开的 STEP 7 项目开始，单击选择连接伙伴站（其它站）。

选择菜单命令**编辑 > 对象属性** (Edit > Object Properties)。

打开“属性 — 其它站”(Properties - Other Station) 对话框。

1. 属性 > 其它站 > 节点列表 (Properties > Other Station > Node List)

   在“**节点列表**”(Node List) 选项卡中选择“新建”(New) 按钮。

   在“选择类型”(Select Type) 中，选择“PTP 节点”(PTP Nodes) 并单击“确定”。

   显示“网络连接”(Network Connection) 对话框。

   选择所需的 **子网** (subnet)，该子网表示 CP 接口和连接伙伴接口之间的连接，同时激活复选框“节点已经连接到所选网络”(Node is connected to selected network)。

   单击“确定”返回到“节点列表”(Node List) 选项卡。

2. 属性 > 其它站 > 常规 (Properties > Other Station > General)

   在“**常规**”(General) 选项卡中无须进行任何设置。

   单击“**确定**”返回到 STEP 7 项目的主菜单。

   伙伴站也可以有几个接口（=PtP 节点），因此也可以连接到不同的点对点网络。

4.6 数据链接的组态

简介

本章只和 CP 441-2 相关。如果使用的是 CP 341，可以跳过本章。

通讯链接

CP 代表了通过点对点数据链接建立的 S7 CPU 和通讯伙伴/总线之间的数据链接。 必须为每个要连接到连接伙伴/总线的串行接口组态数据链接。

数据链接的组态

在 STEP 7 项目中打开的 S7-400 站上选择 CPU，并且通过双击“**连接**”(Connections) 打开数据链接组态。

将显示“执行项目的连接组态”(Carry Out Project Configuration of Connections) 对话框。

选择菜单命令**插入 > 连接** (Insert > Connection) 打开“新建连接”(New Connection) 对话框。 这里可以选择用于新数据链接的连接伙伴（其它站），并且选择“点对点连接”(Point-to-Point Connection) 作为连接类型。

单击“确定”确认。 现在“连接属性”(Connection Properties) 对话框将打开。

连接属性

将分配一个 **ID**，您可以修改该 ID 以满足您的要求。

选择“通讯方向”(Communication Direction) 3： 本地 **<->** 伙伴 (Local **<->** Partner)

将显示参数化的路由。

两个 CPU 编号的指示与本驱动程序的操作无关。

单击“确定”接受所有设置。

保存“数据链接的项目组态”(Project Configuration of Data Link)，并且关闭对话框。

要注意当您在用户程序中调用 SFB 时，连接 ID（本地 ID）必须再次使用。

4.7 为可加载驱动程序分配参数

打开参数分配工具 CP-PtP

选择 SIMATIC 站然后双击“硬件”(Hardware) 或者选择“编辑 > 打开对象”(Edit > Open object) 启动“配置硬件”(Configure hardware)。

选择 CP，然后选择**编辑 > 对象属性** (Edit > Object Properties)。

选择接口（仅限 CP 441-2）和接口模块（仅限 CP 441-2），然后选择“**参数**”(Parameters) 按钮打开协议选择对话框。

协议选择

除了标准协议之外，选择对话框还将显示所有已安装的可加载驱动程序。 为该驱动程序选择“**MODBUS 主站**”(MODBUS Master)。

双击邮箱图标，以设置传输协议。 将显示用于设置协议特定参数的对话框。

驱动程序特定参数

以下描述的用于本驱动程序的参数可以在各对话框中设置。

4.7.1 MODBUS 主站协议

传输参数概述

表格 4- 1　速率，字符帧

参数	说明	值范围	默认值
波特率	数据传输速度（位/秒）	300 600 1200 2400 4800 9600 19200 38400 76800	9600
具有其它波特率的 CP 341 的订货号如下： • 6ES7341-1xH01-0AE0 • 6ES7341-1xH02-0AE0		57600	
具有其它波特率的 CP 441-2 的订货号如下： • 6ES7441-2AA03-0AE0 • 6ES7441-2AA04-0AE0		57600 115200	
数据位	每个字符的位数	8	8
停止位	停止位个数	1 2	1
奇偶校验	没有传送奇偶校验位。	无	偶校验
	数据位数量将补充成奇数。	奇校验	
	数据位数量将补充成偶数。	偶校验	

传输速率

传输速率为数据传输的速度，单位：位/秒（波特）。

注意 CP 441-2 的最大总传输速率。总传输速率是两个接口的已参数化的传输速率的总和。

TTY 接口的最大传输速率为 19200 波特。

数据位

数据位数说明了要传输的一个字符映射的位数。

停止位

停止位数定义了要传输的两个字符之间的最小允许时间间隔。

奇偶校验

奇偶校验位用于数据安全。 根据参数分配，将要传输的数据位数补充成偶数或奇数。

如果奇偶校验已设置为“无校验”(none)，则不传输奇偶校验位。这会降低传输的完整性。

协议参数概述

表格 4- 2　协议参数

参数	说明	值范围	默认值
应答监视时间	应答监视时间是用于监视从站应答开始的时间。	5 到 65500	2000
激活 RS485 的操作模式选择	启用在“半双工”（RS485）两线制模式下的“常规操作”选择。	是 否	否
操作模式	“常规”操作模式 “干扰抑制” (Interference suppression)	常规操作 干扰抑制	• 在“全双工 (RS422) 四线模式”下：常规操作 • 在“半双工 (RS485) 两线模式”下：干扰抑制
倍增字符延时	用于传输速率的倍增因子—取决于字符延迟时间	1 到 10	1

应答监视时间

应答监视时间是主站发出请求消息帧后，用于等待从站的应答消息帧的时间。

激活 RS485 的操作模式选择

在激活“RS485 的操作模式选择”后，如果您已经选择了“半双工 (RS485) 两线制模式”，您还可以在“接口”选项卡下将操作模式切换为“常规操作”。

常规操作

在本操作模式中，在接收连接伙伴的消息帧之前和之后的所有探测到的传输错误或者断路 (BREAK)，都将在用户程序中生成相应的错误消息。

帧的第一个字符必须是一个的有效的从站地址。

仅当字符延迟时间结束时，才识别为消息帧结束。

干扰抑制

如果在开始接收消息帧时接收线探测到断路 (BREAK) 信号，或者 CP 接口块探测到传输错误，将不会在用户程序中生成错误消息。

如果传输错误或 BREAK 是在接收消息帧（CRC 代码）结束后出现的，则也会被忽略。

正确接收到的从站地址可以检测到来自连接伙伴的接收消息帧的起始位置。

倍增字符延时

如果一个连接伙伴无法满足 MODBUS 规范的时间需求，允许使用倍增字符延时 ZVZ，通过倍增因子 f_{MUL} 实现。 只有当要求的时间无法满足时，才可以调节字符延时。

产生的字符延迟时间 t_{ZVZ} 计算方法如下：

- $t_{ZVZ} = t_{ZVZ\_TAB} \times f_{MUL}$
- t_{ZVZ_TAB}： ZVZ 的查表值（参见“传输协议”章节）
- f_{MUL}： 倍增因子

4.7.2 RS422/485 (X27) 接口

概述

表格 4- 3 RS422/485 (X27) 接口

参数	说明	值范围	默认值
操作模式	指定 RS 422/485 (X27) 接口是在全双工 (RS 422) 模式下运行还是在半双工 (RS 485) 模式下运行。	• 全双工 (RS 422) 四线制模式 • 半双工 (RS 485) 两线制模式	全双工 (RS 422) 四线制模式
接收线路的初始状态	**无 (None)：** 两线模式下线 R(A) 和 R(B) 没有初始化。 此时将由连接伙伴进行初始化。	• 无 • 信号 R(A) 5V / 信号 R(B) 0V（断路检测）[1] • 信号 R(A) 0V/ 信号 R(B) 5V	• 在“全双工 (RS422) 四线模式”下： 信号 R(A) 5V/ 信号 R(B) 0V（断路检测）[1] • 在“半双工 (RS485) 两线制模式”下： 信号 R(A) 0V/ 信号 R(B) 5V
	信号 R(A) 5V/ 信号 R(B) 0V（断路检测）： 该缺省设置支持“全双工 (RS422) 4 线制模式”中的断点检测。 不能针对“半双工 (RS485) 两线模式”进行选择		
	信号 R(A) 0 V/ 信号 R(B) 5V： 该初始状态对应于“半全工 (RS 485) 两线制模式”下的空闲状态（无激活的发送器）。在此初始状态下，不能进行断点检测。		
[1] 仅在“全双工 (RS 422) 四线制模式”下。			

“全双工 (RS422) 四线操作”

在该操作模式下，发送是通过发送线 T(A)- 和 T(B)+，而接收是通过接收线 R(A)- 和 R(B)+。

根据“驱动工作模式”参数的功能设置进行错误处理（常规操作或干扰抑制）。

“半双工 (RS485) 两线操作”

在此操作模式下，该驱动程序将在发送和接收操作之间切换接口的 2 线制接收线 R(A)-、R(B)+。

在此操作模式的初始设置下，所有在接收消息帧之前和之后识别的传输错误和/或断路 (BREAK) 都将被忽略。

在消息帧暂停期间的断路 (BREAK) 电平也将忽略。

正确接收到的从站地址可以检测到来自连接伙伴的接收消息帧的起始位置。

推荐“信号 R(A) 0V,R(B) 5V”作为节点的接收线的初始状态设置。 所有其他节点的“接受线初始状态”应设置为“无”。

接收线路初始状态

- **“无”**

两线模式下线 R(A)- 和 R(B)+ 没有初始化。 此时将由连接伙伴进行初始化。

- **初始化“R(A) 5V, R(B) 0V”（断路）**

两线模式下的线 R(A)- 和 R(B)+ 将通过 CP 按如下方法初始化：
R(A) --> +5V, R(B) --> 0V (V_A - V_B ≥ +0.3V)。
这就意味着在断线情况下 CP 将产生断路 (BREAK) 电平。

此选项仅在“全双工 (RS 422) 四线制模式”下可以选择。

- **初始化“R(A) 0V, R(B) 5V”**

两线模式下的线 R(A)- 和 R(B)+ 将通过 CP 按如下方法初始化：
R(A) --> 0V, R(B) --> +5V (V_A - V_B ≤ 0.3V)。
这就意味着在线上没有传输时，在线缆断开和/或空闲状态情况下将产生高电平。 无法检测断路线路状态。

参数选择

选择您的连接所必需的设置，并单击“确定”(OK) 退出单个屏幕窗口。

4.8 装载 CP 341 的组态和参数分配数据

数据存储

当您关闭“硬件配置”(hardware configuration) 时，数据将自动保存到 STEP 7 项目中。

装载组态和参数

现在可以将组态和参数分配数据从编程设备在线装载到 CPU 中。 选择**目标系统 > 装载** (Target system > Load) 菜单命令将数据传送到 CPU 中。

当 CPU 启动时，并且在 CPU 从 STOP（停止）切换到 RUN（运行）或者反过来切换时，只要通过 S7300 背板总线可以访问到 CP，CP 的模块参数将自动从 CPU 传送到 CP 中。

驱动程序代码不是存储在 CPU 中，而是使用参数分配接口直接存储在 CP 341 的保持性存储器中。 但是注意，在没有编程设备的情况下将不能更换模块。

更多信息

STEP 7 用户手册提供了以下操作的详细说明：

- 保存组态和参数
- 将组态和参数装载到 CPU
- 读取、修改、复制和打印组态和参数。

4.9 装载驱动程序到 CP 341

要求

可以在线连接到 CPU。

装载驱动程序

1. 在"分配点到点连接参数"(Assigning Parameters to Point-to-Point Connection) 窗口内的"协议"(Protocol) 下拉框选择所需的可加载驱动程序。
2. 点击“装载驱动程序”(Load drivers) 图标。

   在“装载驱动程序到 CP341”(Load drivers to CP341) 窗口内，您可以看到在线装载到模块的驱动程序版本，以及您在编程设备上离线选择的驱动程序版本。
3. 点击“装载驱动程序”(Load drivers) 按钮并用“是”(Yes) 确认。
   装载驱动程序到 CP 341。
   装载完成后，“模块上的在线驱动程序版本”(Driver version online on the module) 信息将会更新。
   如果装载的驱动程序已经存在于 CP 341，装载操作将取消并显示消息“驱动程序已存在”(Driver already exists)。 这种情况下，点击“确定”(OK) 确认并关闭“下载驱动程序到 CP341”(Download Drivers to CP341) 窗口。

   如果驱动程序文件不存在或者不正确，将会收到错误信息 “模块拒绝驱动程序下载” (Module rejected driver download)。 在这种情况下，您必须重新安装驱动程序。

4.10 装载 CP 441-2 的组态和参数分配数据

数据存储

当您关闭“硬件配置”(hardware configuration) 和/或“连接组态”(configuration of connections) 时，包括模块参数和驱动程序代码在内的数据将自动保存到 STEP 7 项目中。

装载组态和参数

现在可以将组态和参数分配数据从编程设备在线装载到 CPU 中。 选择 **目标系统 > 装载** (Target system > Load) 菜单命令将数据传送到 CPU 中。

当 CPU 启动时，只要通过 S7400 背板总线可以访问到 CP，CP 的模块参数将自动从 CPU 传送到 CP 中。

更多信息

STEP 7 用户手册提供了以下操作的详细说明：

- 保存组态和参数
- 将组态和参数装载到 CPU
- 读取、修改、复制和打印组态和参数。

4.11 CP 的启动特性

简介

CP 的启动分为两个阶段：

- 上电初始化 CP
- 参数分配

初始化

只要 CP 一上电，在执行完硬件测试程序后，CP 的固件即可运行。

参数分配

在参数分配过程中，CP 将接收分配到当前插槽的模块参数。

现在可以操作 CP 了。

4.12 “CPU 启动”的参数分配

简介

本章只和 CP 441-2 相关。如果使用的是 CP 341，可以跳过本章。

硬件配置 (Hardware Configuration)

要避免在 CPU-CP 启动时出现问题，在使用“**硬件配置**”(Hardware configuration) 对 **CPU** 执行 **参数分配** (parameter assignment) 时，应进行以下设置。

通过双击 CPU 或者单击 CPU 选择菜单命令 **编辑 > 对象属性** (Edit > Object properties) 启动参数分配后，将显示“属性 — CPU ”(Properties - CPU) 页面。

将 “**监视时间**” (Monitoring Time for) 选项设置成最小值 *3000* (= 300s)，该选项位于“**启动**”(Startup) 选项卡的点“**到模块的参数传输 (100ms)。**”(Transfer of Parameters to Modules (100ms).) 中。

原因： 在为具有可加载驱动程序的 CP 441-2 接口分配参数时，除了已分配参数之外驱动程序代码也将传送到 CP 中。 在上述时间对整个装载的过程进行监控（时间必须充足）。 .

传输协议

5

常规信息

使用的程序是明码、异步半双工的程序。

数据传输无须握手。

主站-从站关系

作为主站，CP 将初始化传输，随后输出请求消息帧，然后开始等待用于从站应答消息帧的参数化的应答监视时间。

消息帧结构

“主站-从站”和/或“从站-主站”数据交换以**从站地址**开始，然后是**功能代码**。 然后传输数据。 数据域的结构取决于使用的功能代码。 消息帧结束时传送的是 CRC 校验码。

地址	功能	数据	CRC 校验
字节	字节	n 个字节	2 个字节

地址	MODBUS 从站地址
功能	MODBUS 功能代码
数据	消息帧数据： 字节数、线圈编号和数据
CRC 校验	消息帧校验和

从站地址

从站地址设置范围为 1 到 255。地址可分配给总线上已定义的从站。

广播消息

主站使用从站地址 0 对总线上的所有从站进行寻址。

广播消息仅允许与写**功能代码 05、06、15 和 16** 相结合。

广播消息不需要从站的应答消息帧。

功能代码

功能代码定义了消息帧的含义。 同样它也定义了消息帧的结构。 CP 支持以下这些功能代码：

功能代码	符合 MODBUS 规范的功能
01	读线圈状态
02	读输入状态
03	读保持寄存器
04	读输入寄存器
05	写单个线圈
06	预设单个寄存器
07	读异常状态
08	环路测试
11	获取通信事件计数器
12	获取通信事件日志
15	写多个线圈
16	预设多个寄存器

数据域 DATA

数据域 DATA 用于传送功能代码特定数据，例如：

- 字节数、线圈起始地址、寄存器起始地址、线圈数量和寄存器数量等等

请参见“功能代码 (页 51)”部分。

CRC 校验

消息帧的最后是由 2 个字节组成的 CRC 16 校验和。 校验和是按如下多项式计算的：$x^{16} + x^{15} + x^2 + 1$。

先传输低位字节，然后传输高位字节。

消息帧的结尾

当在 3.5 个字符传输时间（字符延迟时间的 3.5 倍）周期内没有任何数据传输，可加载驱动程序将认为消息帧传输已经结束（参见 MODBUS 协议参考指南）。

因此消息帧结束的超时 (TIME_OUT) 取决于传输速率。

传输速率	超时 (TIME_OUT)
76800 波特	0.5 毫秒
38400 波特	1 毫秒
19200 波特	2 毫秒
9600 波特	4 毫秒
4800 波特	8 毫秒
2400 波特	16 毫秒
1200 波特	32 毫秒
600 波特	64 毫秒
300 波特	128 毫秒

在“常规操作”期间，通过连接伙伴接收的 Modbus 消息帧将在收到与 TIME_OUT 对应的帧结束信息之后进行评估和校验。

在“干扰抑制”期间，使用正确的 CRC 码通过格式正确的接收帧识别帧的结束信息。

异常响应

当检测到主站的请求消息有错误时，比如：寄存器地址非法，从站将置位应答消息帧的功能代码的最高位。

随后传输的是一个字节的错误代码，即描述错误原因的异常代码。

上述参数的含义的详细描述可从手册《GOULD MODICON Modbus 协议》中找到。

异常代码消息帧

从站的错误代码应答消息帧的结构如下：

- 例如，从站地址 5，功能代码 5，异常代码 2

从站的响应消息帧 EXCEPTION_CODE_xx：

05H	从站地址
85H	功能代码
02H	异常代码 (1...7)
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

驱动器接收到错误代码应答消息帧后，当前的工作将由于错误而结束。

对应于接收到的错误代码（异常代码 1-7）的错误编号将输入到 SYSTAT 区域。

BRCV 目标数据块中没有生成任何条目。

根据 MODBUS 规范定义了下列的错误代码：

错误代码	符合 MODBUS 规范的含义	原因—短描述*
1	非法功能	功能代码非法
2	非法数据地址	从站具有非法的数据地址
3	非法数据值	从站具有非法的数据值
4	关联设备发生故障	从站出现内部错误
5	确认	功能已执行
6	忙，拒收消息	从站尚未准备好接收消息
7	否定确认	该功能不能执行。
* 检查从站获取更多详细信息。		

RS 232C 辅助信号

若使用的是 RS 232C 接口子模块，则 CP 中存在下列的 RS 232C 辅助信号：

DCD	（输入）	数据载体检测	检测到的数据载体
DTR	（输出）	数据终端就绪	CP 就绪
DSR	（输入）	数据集准备就绪	通信伙伴就绪
RTS	（输出）	请求发送	CP 发送准备就绪
CTS	（输入）	清除发送	通信伙伴可以接收到 CP 的数据（响应 CP 的 RTS = ON）
RI	（输入）	响铃指示	指示呼叫进入

CP 接通时，输出信号状态为 OFF（未激活）。

如果使用了 DTR/DSR 和 RTS/CTS 控制信号，则可以使用参数化接口**为点对点连接分配参数** (Assigning Parameters to Point-To-Point Connections) 设置路径参数，或者通过在用户程序中调用功能 (FB) 来控制它们。

使用 RS 232C 辅助信号

RS 232C 辅助信号可在以下情况下使用：

- 组态了自动使用所有 RS 232C 辅助信号。
- 通过 FB V24_STAT 和 FB V24_SET 功能

说明

当参数设置成自动使用所有 RS 232C 辅助信号时，通过 V24_SET FB 实现的 RTS/CTS 数据流控制以及 RTS 和 DTR 控制都不允许使用!

另一方面，总是允许通过 FB V24_STAT 功能读取所有的 RS 232C 辅助信号。

下面一部分将描述如何控制和评估 RS 232C 辅助信号。

伴随信号的自动使用

CP 中 RS 232C 辅助信号的自动使用是按如下步骤执行的：

- 只要 CP 通过参数设置切换成自动使用 RS 232C 辅助信号操作模式，在上电时它将 RTS 设置成 OFF，同时将 DTR 设置成 ON（CP 就绪）。
- 这将阻止在 DTR 设置成 ON 之前收发消息帧。只要 DTR 仍设置为 OFF，便不能通过 RS 232C 接口接收任何数据。 如果提出发送请求，请求将被中止，并生成相应的错误消息。
- 当发送请求生成时，RTS 将设置成 ON 并且已设置的数据输出等待时间将启动。 当数据输出等待时间到并且 CTS = ON 时，数据将通过 RS 232C 接口发送。
- 如果 CTS 线路在数据输出时间内未设置为 ON 以便可以发送数据，或者 CTS 在传输过程中更改为 OFF，发送请求会被中止，并生成错误消息。
- 一旦数据发送且超过组态的清除 RTS 时间，RTS 线路将立即设置为 OFF。 CP 不会等待 CTS 更改为 OFF。
- 一旦 DSR 线路设置为 ON，即可通过 RS 232C 接口接收数据。 如果 CP 的接收缓冲区预警将要溢出，则 CP 将不响应。

- 如果 DSR 从 ON 转变成 OFF，激活的发送请求和数据接收都将取消，并产生错误消息。 消息“DSR = OFF（自动使用 V24 信号）”将输入到 CP 的诊断缓冲区中。

说明

RS 232C 辅助信号的自动使用仅在半双工模式下可以实现。 当参数设置成自动使用所有 RS 232C 辅助信号时，通过 V24_SET FB 实现的 RTS/CTS 数据流控制以及 RTS 和 DTR 控制都不允许使用!

说明

在接口参数设置中必须设置“RTS 保持关闭状态的时间”，这样通信伙伴才能在 RTS 之前完全接收到消息帧的最后的字符，并且发送请求也会因此取消。“数据输出等待时间”也必须设置，这样通信伙伴才能在超时之前做好接收准备。

时序图

下图说明了发送请求的时间顺序。

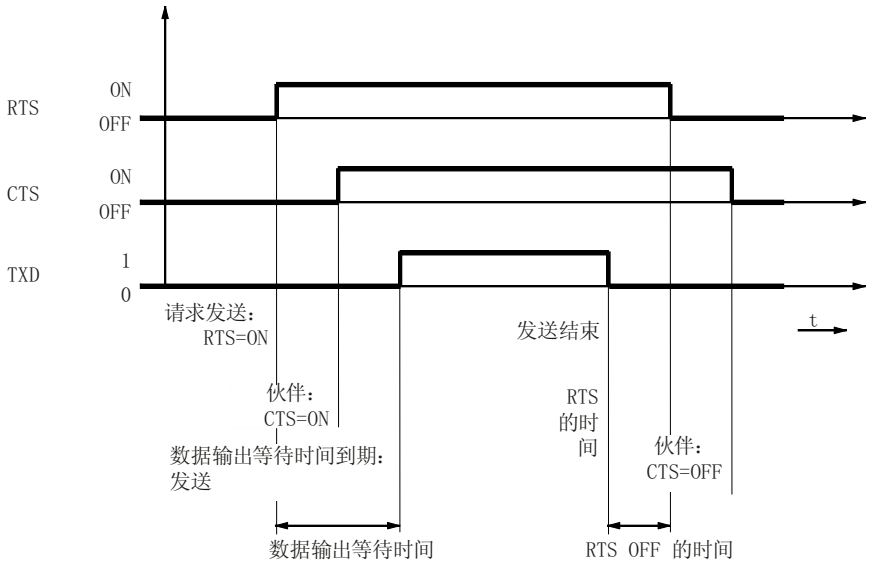


图 5-1　　自动使用 RS 232C 辅助信号的时序图

功能代码

6

6.1 功能代码 01 – 读输出状态

功能

使用该功能可以读取从站的各个位。

起始地址

驱动器不检查参数**位启动地址**，而且在发送时不会改变它。

位数

1 到 **2040** 之间的任何值都可以用作**位数**（线圈数）。

SEND 源 DB

SEND 源区域的结构：

地址	名称	类型	初始值	注释
+0.0	地址	BYTE	B#16#5	从站地址
+1.0	功能	BYTE	B#16#1	功能代码
+2.0	bit_startadr	WORD	W#16#0040	位起始地址
+4.0	bit_anzahl	INT	16	位数

示例

请求消息帧 FUNCTION 01：

05H	从站地址
01H	功能代码
00H	位起始地址“高字节”
40H	位起始地址“低字节”
00H	位数“高字节”
10H	位数“低字节”
xxH	CRC 校验代码 “低字节”
xxH	CRC 校验代码 “高字节”

FUNCTION 01 的从站应答消息帧：

05H	从站地址
01H	功能代码
02H	字节计数器
01H	<数据>
17H	<数据>
xxH	CRC 校验代码 “低字节”
xxH	CRC 校验代码 “高字节”

RCV 目标 DB

RCV 目标区域内容：

地址	名称	类型	当前值	注释
+0.0	data[1]	WORD	W#16#1701	数据

驱动器将应答消息的数据**按字**的顺序输入到目标 DB。

收到的第 1 个字节将存入第 1 个字“data[1]”的低字节中，收到的第 2 个字节将存入第 2 个字“data[2]”的低字节中，依此类推。

如果读到的数据少于 9 位或者只读到一个低字节，剩余的最后一个字的高字节将用 **00H** 填补。

6.2 功能代码 02 — 读输入状态

功能

使用该功能可以读取从站的各个位。

起始地址

驱动程序不检查参数**位启动地址**，而且在发送时不会改变它。

位数

1 到 **2040** 之间的任何值都可以用作**位数**（线圈数）。

SEND 源 DB

SEND 源区域的结构：

地址	名称	类型	初始值	注释
+0.0	地址	BYTE	B#16#5	从站地址
+1.0	功能	BYTE	B#16#2	功能代码
+2.0	bit_startadr	WORD	W#16#0120	位起始地址
+4.0	bit_anzahl	INT	24	位数

示例

请求消息帧 FUNCTION 02：

05H	从站地址
02H	功能代码
01H	位起始地址“高字节”
20H	位起始地址“低字节”
00H	位数“高字节”
18H	位数“低字节”
xxH	CRC 校验代码 “低字节”
xxH	CRC 校验代码 “高字节”

FUNCTION 02 的从站应答消息帧：

05H	从站地址
02H	功能代码
03H	字节计数器
04H	<数据>
26H	<数据>
48H	<数据>
xxH	CRC 校验代码 “低字节”
xxH	CRC 校验代码 “高字节”

RCV 目标 DB

RCV 目标区域内容：

地址	名称	类型	当前值	注释
+0.0	data[1]	WORD	W#16#2604	数据
+2.0	data[2]	WORD	W#16#0048	数据

驱动器将应答消息的数据**按字**的顺序输入到目标 DB。

收到的第 1 个字节将存入第 1 个字“data[1]”的低字节中，收到的第 2 个字节将存入第 2 个字“data[2]”的低字节中，依此类推。

如果读到的数据少于 9 位或者只读到一个低字节，剩余的最后一个字的高字节将用 **00H** 填补。

6.3 功能代码 03 – 读输出寄存器

功能

使用该功能可以读取从站的各个寄存器。

起始地址

驱动器不检查参数**寄存器起始地址**，而且在发送时不会改变它。

寄存器数

最多可以读取 **1** 到 **127 个寄存器**（1 寄存器＝两个字节）。

SEND 源 DB

SEND 源区域的结构：

地址	名称	类型	初始值	注释
+0.0	地址	BYTE	B#16#5	从站地址
+1.0	功能	BYTE	B#16#3	功能代码
+2.0	reg_startadr	WORD	W#16#0040	寄存器起始地址
+4.0	reg_anzahl	INT	2	寄存器数

示例

请求消息帧 FUNCTION 03：

05H	从站地址
03H	功能代码
00H	寄存器起始地址“高字节”
40H	寄存器起始地址“低字节”
00H	寄存器总数“高字节”
02H	寄存器总数“低字节”
xxH	CRC 校验代码 “低字节”
xxH	CRC 校验代码 “高字节”

FUNCTION 03 的从站应答消息帧：

05H	从站地址
03H	功能代码
04H	字节计数器
21H	寄存器地址 40H 数据“高字节”
23H	寄存器地址 40H 数据“低字节”
25H	寄存器地址 41H 数据“高字节”
27H	寄存器地址 41H 数据“低字节”
xxH	CRC 校验代码 “低字节”
xxH	CRC 校验代码 “高字节”

RCV 目标 DB

RCV 目标区域内容：

地址	名称	类型	当前值	注释
+0.0	data[1]	WORD	W#16#2123	数据
+2.0	data[2]	WORD	W#16#2527	数据

6.4 功能代码 04 – 读输入寄存器

功能

使用该功能可以读取从站的各个寄存器。

起始地址

驱动器不检查参数**寄存器起始地址**，而且在发送时不会改变它。

寄存器数

最多可以读取 **1** 到 **127 个寄存器**（1 寄存器＝两个字节）。

SEND 源 DB

SEND 源区域的结构：

地址	名称	类型	初始值	注释
+0.0	地址	BYTE	B#16#5	从站地址
+1.0	功能	BYTE	B#16#4	功能代码
+2.0	reg_startadr	WORD	W#16#0050	寄存器起始地址
+4.0	reg_anzahl	INT	3	寄存器数

示例

请求消息帧 FUNCTION 04：

05H	从站地址
04H	功能代码
00H	寄存器起始地址“高字节”
50H	寄存器起始地址“低字节”
00H	寄存器总数“高字节”
03H	寄存器总数“低字节”
xxH	CRC 校验代码 “低字节”
xxH	CRC 校验代码 “高字节”

FUNCTION 04 的从站应答消息帧：

05H	从站地址
04H	功能代码
04H	字节计数器
31H	寄存器地址 50H 数据“高字节”
32H	寄存器地址 50H 数据“低字节”
33H	寄存器地址 51H 数据“高字节”
34H	寄存器地址 51H 数据“低字节”
35H	寄存器地址 52H 数据“高字节”
36H	寄存器地址 52H 数据“低字节”
xxH	CRC 校验代码 “低字节”
xxH	CRC 校验代码 “高字节”

RCV 目标 DB

RCV 目标区域内容：

地址	名称	类型	当前值	注释
+0.0	data[1]	WORD	W#16#3132	数据
+2.0	data[2]	WORD	W#16#3334	数据
+4.0	data[3]	WORD	W#16#3536	数据

6.5 功能代码 05 — 写单个线圈

功能

使用该功能可以设置或删除从站中的各个位。

位地址

驱动程序不会检查**位地址**参数，因此将参数原封不动的发出去。

位状态

下列两个数值可用作**位状态**：

- FF00H = 设置位
- 0000H = 删除位

SEND 源 DB

SEND 源区域的结构：

地址	名称	类型	初始值	注释
+0.0	地址	BYTE	B#16#5	从站地址
+1.0	功能	BYTE	B#16#5	功能代码
+2.0	bit_address	WORD	W#16#0019	位地址
+4.0	bit_state	WORD	W#16#FF00	位状态

示例

请求消息帧 FUNCTION 05：

05H	从站地址
05H	功能代码
00H	位地址“高字节”
19H	位地址“低字节”
FFH	设置位
00H	
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

来自从站 FUNCTION 05 的应答消息帧：

05H	从站地址
05H	功能代码
00H	位地址“高字节”
19H	位地址“低字节”
FFH	位状态“高字节”
00H	位状态“低字节”
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

6.6 功能代码 06 — 预设单个寄存器

功能

使用该命令可以用新值覆盖从站寄存器。

寄存器地址

驱动程序并不检查**寄存器地址**参数，因此将参数原封不动地发送出去。

寄存器值

任何值都可以用作**寄存器值**。

SEND 源 DB

SEND 源区域的结构：

地址	名称	类型	初始值	注释
+0.0	地址	BYTE	B#16#5	从站地址
+1.0	功能	BYTE	B#16#6	功能代码
+2.0	reg_address	WORD	W#16#0180	寄存器地址
+4.0	reg_value	WORD	W#16#3E7F	寄存器值

示例

请求消息帧 FUNCTION 06：

05H	从站地址
06H	功能代码
01H	寄存器地址“高字节”
80H	寄存器地址“低字节”
3EH	寄存器值“高字节”
7FH	寄存器值“低字节”
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

来自从站 FUNCTION 06 的应答消息帧：

05H	从站地址
06H	功能代码
01H	寄存器地址“高字节”
80H	寄存器地址“低字节”
3EH	寄存器值“高字节”
7FH	寄存器值“低字节”
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

6.7 功能代码 07 — 读取异常状态

功能

使用该功能码可以自连接的从站中读取 8 个事件位。

事件位的起始位号是由所连接的设备决定的，因此并不是必须由 SIMATIC 用户程序指定。

SEND 源 DB

SEND 源区域的结构：

地址	名称	类型	初始值	注释
+0.0	地址	BYTE	B#16#5	从站地址
+1.0	功能	BYTE	B#16#7	功能代码

示例

请求消息帧 FUNCTION 07：

05H	从站地址
07H	功能代码
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

来自从站 FUNCTION 07 的应答消息帧：

05H	从站地址
07H	功能代码
3EH	<数据>
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

RCV 目标 DB

RCV 目标区域的内容：

地址	名称	类型	当前值	注释
+0.0	data[1]	WORD	W#16#3Exx	数据

驱动程序在目标 DB data[1] 中的**高字节**内输入应答消息帧的各个字节。

data[1] 的低字节保持不变。

显示数值 1，作为 BRCV 的 LEN 参数中的长度。

6.8 功能代码 08 — 环路诊断测试

功能

此功能用于检查通讯连接。

此功能代码只支持**诊断代码 0000**！

诊断代码

参数**诊断代码**的唯一允许值是 0000。

测试值

任何值都可以用作**测试值**。

SEND 源 DB

SEND 源区域的结构：

地址	名称	类型	初始值	注释
+0.0	地址	BYTE	B#16#5	从站地址
+1.0	功能	BYTE	B#16#8	功能代码
+2.0	diag_code	WORD	W#16#0000	诊断代码
+4.0	test_value	WORD	W#16#A5C3	测试值

示例

请求消息帧 FUNCTION 08：

05H	从站地址
08H	功能代码
00H	诊断代码“高字节”
00H	诊断代码“低字节”
A5H	测试值“高字节”
C3H	测试值“低字节”
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

来自从站 FUNCTION 08 的应答消息帧：

05H	从站地址
08H	功能代码
00H	诊断代码“高字节”
00H	诊断代码“低字节”
A5H	测试值“高字节”
C3H	测试值“低字节”
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

从站必须原封不动地返回从主站接收到的请求消息帧，作为回应。

应答消息帧并不输入到 RCV DB。

6.9 功能代码 11 — 获取通讯事件计数器

功能

此功能代码用于从从站中读取**“状态字”**（2 字节长）和**“事件计数器”**（2 字节长）。

上述参数的含义在手册“GOULD MODICON Modbus 协议”中做了详细描述。

SEND 源 DB

SEND 源区域的结构：

地址	名称	类型	初始值	注释
+0.0	地址	BYTE	B#16#5	从站地址
+1.0	功能	BYTE	B#16#0B	功能代码

示例

请求消息帧 FUNCTION 11：

05H	从站地址
0BH	功能代码
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

来自从站 FUNCTION 11 的应答消息帧：

05H	从站地址
0BH	功能代码
FEH	状态字“高字节”
DCH	状态字“低字节”
01H	事件计数器“高字节”
08H	事件计数器“低字节”
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

RCV 目标 DB

RCV 目标区域的内容：

地址	名称	类型	当前值	注释
+0.0	data[1]	WORD	W#16#FEDC	状态字
+2.0	data[2]	WORD	W#16#0108	事件计数器

6.10 功能代码 12 — 获取通讯事件日志

功能

此功能代码使用户可以从从站中读取下列信息。

- 2 字节**“状态字”**
- 2 字节**“事件计数器”**，
- 2 字节**“消息计数器”**和
- 64 字节**“事件字节”**

上述参数的含义在手册“GOULD MODICON Modbus 协议”中做了详细描述。

SEND 源 DB

SEND 源区域的结构：

地址	名称	类型	初始值	注释
+0.0	地址	BYTE	B#16#5	从站地址
+1.0	功能	BYTE	B#16#0C	功能代码

示例

请求消息帧 FUNCTION 12：

05H	从站地址
0CH	功能代码
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

来自从站 FUNCTION 12 的应答消息帧：

05H	从站地址
0CH	功能代码
46H	字节计数器
87H	状态字“高字节”
65H	状态字“低字节”
01H	事件计数器“高字节”
08H	事件计数器“低字节”
02H	消息计数器“高字节”
20H	消息计数器“低字节”
01H	事件字节 1
12H	事件字节 2
:	:
C2H	事件字节 63
D3H	事件字节 64
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

RCV 目标 DB

RCV 目标区域的内容：

地址	名称	类型	当前值	注释
+0.0	data[1]	WORD	W#16#8765	状态字
+2.0	data[2]	WORD	W#16#0108	事件计数器
+4.0	data[3]	WORD	W#16#0220	消息计数器
+6.0	bytedata[1]	BYTE	B#16#01	事件字节 1
+7.0	bytedata[2]	BYTE	B#16#12	事件字节 2
:	:			:
+68.0	bytedata[63]	BYTE	B#16#C2	事件字节 63
+69.0	bytedata[64]	BYTE	B#16#D3	事件字节 64

6.11 功能代码 15 — 写多个线圈

功能

使用该功能代码最多可以在从站中更改 2,040 个位。

起始地址

驱动程序并不检查**位起始地址**参数，因此将参数原封不动地发送出去。

位数

1 到 **2040** 之间的任何值都可以用作**位数**（线圈数）。

这指定了从站中要覆盖的位数。

请求消息帧中的“字节计数器”参数是由驱动程序根据传送的参数“位数”生成的。

SEND 源 DB

SEND 源区域的结构：

地址	名称	类型	初始值	注释
+0.0	地址	BYTE	B#16#5	从站地址
+1.0	功能	BYTE	B#16#0F	功能代码
+2.0	bit_startadr	WORD	W#16#0058	位起始地址
+4.0	bit_anzahl	INT	10	位数
+6.0	coil_state[1]	WORD	W#16#EFCD	状态 线圈 5FH..58H/57H..50H

示例

请求消息帧 FUNCTION 15：

05H	从站地址
0FH	功能代码
00H	位地址“高字节”
50H	位地址“低字节”
00H	位数“高字节”
0AH	位数“低字节”
02H	字节计数器
CDH	状态线圈 50H..57H
EFH	状态线圈 58H..59H
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

来自从站 FUNCTION 15 的应答消息帧：

05H	从站地址
0FH	功能代码
00H	位地址“高字节”
50H	位地址“低字节”
00H	位数“高字节”
0AH	位数“低字节”
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

6.12 功能代码 16 — 预设多个寄存器

功能

功能代码 16 使用户通过一个请求消息帧即可覆盖从站中的 **127 个寄存器**。

起始地址

驱动程序并不检查**寄存器起始地址**参数，因此将参数原封不动地发送出去。

寄存器数

可以读取 **1** 到**最多 127 个寄存器**（1 个寄存器 = 2 个字节）。

请求消息帧中的“字节计数器”参数是由驱动程序根据传送的参数“寄存器数”生成的。

SEND 源 DB

SEND 源区域的结构：

地址	名称	类型	初始值	注释
+0.0	地址	BYTE	B#16#5	从站地址
+1.0	功能	BYTE	B#16#10	功能代码
+2.0	reg_startadr	WORD	W#16#0060	寄存器起始地址
+4.0	reg_anzahl	INT	3	寄存器数
+6.0	reg_data[1]	WORD	W#16#41A1	寄存器数据
+8.0	reg_data[2]	WORD	W#16#42A2	寄存器数据
+10.0	reg_data[3]	WORD	W#16#43A3	寄存器数据

示例

请求消息帧 FUNCTION 16：

05H	从站地址
10H	功能代码
00H	寄存器地址“高字节”
60H	寄存器地址“低字节”
00H	寄存器数“高字节”
03H	寄存器数“低字节”
06H	字节计数器
41H	<reg_data[1]>“高字节”
A1H	<reg_data[1]>“低字节”
42H	<reg_data[2]>“高字节”
A2H	<reg_data[2]>“低字节”
43H	<reg_data[3]>“高字节”
A3H	<reg_data[3]>“低字节”
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

来自从站 FUNCTION 16 的应答消息帧：

05H	从站地址
10H	功能代码
00H	寄存器地址“高字节”
60H	寄存器地址“低字节”
00H	寄存器数“高字节”
03H	寄存器数“低字节”
xxH	CRC 校验码“低字节”
xxH	CRC 校验码“高字节”

CPU-CP 接口

7

7.1 用于 CP 341 的 CPU-CP 接口

使用的 SFB

在 CP 和 CPU 之间通过 **P_SND_RK** 和 **P_RCV_RK** FB 传送数据。

如果要输出数据，则通过输入 **REQ** 上的边沿激活 FB P_SND_RK。

FB P_RCV_RK 通过 **EN_R=1** 指示已经准备好接收数据。

所有读取功能代码都需要 P_RCV_RK。

请求的并行处理

对于使用的每个 CP 341，在用户程序中只能同时调用一个 FB P_SND_RK 和一个 FB P_RCV_RK。

在用户程序中集成作业

必须多次打开块 P_SND_RK 和 P_RCV_RK，才能完全处理一份 MODBUS 主站作业。 读/写数据量决定这些块的所需打开频率。 如果在 CPU 的 OB1 循环中打开块 P_SND_RK 和 P_RCV_RK，则可加快数据交换。 如果在“慢”循环中断中打开这些块，则数据交换将需要更长的时间。

7.1.1 从 CPU 到 CP 通过 P_SND_RK (CP 341) 的数据传送

激活

通过 SFB **P_SND_RK** 在输入 **REQ** 处的**边沿**来激活 MODBUS 功能代码的执行。

为 SEND 在 SF 参数处输入“S”。

在 LADDR 处输入逻辑模块地址。

必须为扩展数据块输入 “X”，作为伙伴 CPU 的区域类型。 不必为伙伴 CPU (R_...) 的其他参数指定值。

这样就确保了将执行功能代码所需要的参数传送到驱动程序。

数据源

当激活 P_SND_RK 时，通过参数 **DB_NO** 和 **DBB_NO** 指定的**源数据区**传送到 CP，长度为 **LEN**。

长度指示

长度 **LEN** 取决于所使用的功能代码。

功能代码	长度 LEN（以字节为单位）
01	6
02	6
03	6
04	6
05	6
06	6
07	2
08	6
11	2
12	2
15	>6
16	>6

如果传送的数据数量与上面列出的各个功能代码的数据数量不同，则不会执行作业，P_SND_RK 通过输出 ERROR 处的边沿来拒绝该作业。

SEND 源 DB

执行功能代码所需要的参数必须作为用户数据，输入到源数据区中。

“功能代码 (页 51)”部分的相应功能代码说明中详细描述了各个 P_SND_RK 源 DB。

生成消息帧

到从站的请求消息帧是根据传送的 P_SND_RK 源数据生成的，并由 CP 发送。

首先，驱动程序检查在 P_SND_RK 处指定的长度 LEN 是否与此功能代码的长度相符。

如果不是，则不会执行作业，同时在 P_SND_RK 的输出 ERROR 上生成一个边沿信号作为结束。

当使用上面列出的功能代码之外的其他功能代码时，也不会执行激活的作业，而是通过 P_SND_RK 上的 ERROR 来结束该作业。

请求消息帧中的元素“字节计数器”和“CRC 校验”是由 CP 生成的，不需要 P_SND_RK 源 DB 中的条目。

写入功能的作业完成

对于写入功能代码，在接收到应答消息帧且无错误后，激活的 P_SND_RK 完成。 这通过 P_SND_RK 的输出 **DONE** 上的边沿信号传送到 SIMATIC 用户程序。

如果在消息交换过程中检测到**错误**，或者如果从站发送了**错误代码**应答消息帧，则通过输出 **ERROR** 处的边沿信号进行报告。

读取功能的作业完成

对于读取功能，在接收到应答消息帧且无错误，**并且**将接收的数据完全传送到 CPU 之后，激活的 P_SND_RK 完成。

这通过 P_SND_RK 的输出 **DONE** 上的边沿信号传送到 SIMATIC 用户程序。

此时，接收的数据已经在 CPU 中可用。

如果在消息交换过程中检测到**错误**，或者从站发送了**错误代码**应答消息帧，则通过输出 **ERROR** 的沿信号报告这一情况。

在这种情况下，不会传送任何接收数据到 CPU。

作业完成时的 STATUS 条目

对于这些实例，在作业完成时通过 P_SND_RK 上的 **ERROR** 进行指示，同时在**状态**参数中输入附加的错误代码。

可以使用此错误代码确定错误的准确原因。

7.1.2 从 CP 到 CPU 通过 P_RCV_RK (CP 341) 的数据传送

先决条件

所有**读取**功能代码都需要 P_RCV_RK。

数据目标地址

当 FB P_RCV_RK 准备好接收数据时，它接受从 CP 中接收到的数据，然后将数据输入到在参数 **DB_N0** 和 **DBB_N0** 中指定的数据目标地址。

如何显示数据接收

通过输出 **NDR** 上的边沿信号来通知用户在 CPU 中接收到数据。

此处，接收的数据块长度显示在参数 **LEN** 中。

整个 Modbus 作业的完成可以在 FB P_SND_RK 的输出 DONE 处识别。

如何处理错误

在发生接收或发送错误时，不会传送任何数据到 CPU。 在此实例中，通过输出 ERROR 上的边沿信号来指示 P_SND_RK 已完成。

P_RCV_RK 目标 DB

通过读取功能代码接收到的用户数据输入到 P_RCV_RK 目标地址区域。

关于每个 P_SND_RK 目标 DB 的详细描述可以在“功能代码 (页 51)”一章中找到。

输入数据的长度显示在 P_RCV_RK 的参数 LEN 中。

7.2 用于 CP 441-2 的 CPU-CP 接口

使用的 SFB

CP 和 CPU 之间的数据传送是通过 SFB **BSEND** 和 **BRCV** 执行的。

如果要输出数据，则通过输入 **REQ** 上的边沿激活 SFB BSEND。

SFB BRCV 通过 **EN_R=1** 指示已经准备好接收数据。

所有读取功能代码都需要 BRCV。

7.2.1 从 CPU 到 CP 通过 BSEND (CP 441-2) 的数据传送

通讯链接

参数 ID 描述了到通讯伙伴的唯一通讯链接。 必须在此处指定来自数据链接组态的本地 ID。

块关系

参数 R_ID 描述了通讯链接中唯一的块关系。

通过此驱动程序，可以为 BSEND 上的 R_ID 输入 **0..255** 范围内的**任意**数值。

在读取作业事件中，相关 BRCV 的参数分配必须具有与 BSEND 相同的 R_ID。

激活

通过 SFB **BSEND** 在输入 **REQ** 处的**边沿**来激活 MODBUS 功能代码的执行。

这样就确保了将执行功能代码所需要的参数传送到驱动程序。

数据源

当激活 BSEND 时，通过参数 **SD_1** 指定的**源数据区**传送到 CP，长度为 **LEN**。

长度指示

长度 **LEN** 取决于使用的功能代码。

功能代码	长度 LEN（以字节为单位）
01	6
02	6
03	6
04	6
05	6
06	6
07	2
08	6
11	2
12	2
15	>6
16	>6

如果传送的数据数量与上面列出的各个功能代码的数据数量不同，则不会执行作业，BSEND 通过输出 ERROR 上的边沿来拒绝该作业。

BSEND 源 DB

执行功能代码所需要的参数必须作为用户数据，输入到源数据区中。

关于每个 BSEND 源 DB 的详细描述可以在“功能代码”一章中找到。

生成消息帧

到从站的请求消息帧是根据传送的 BSEND 源数据生成的，并由 CP 发送。

首先，驱动程序检查在 BSEND 处指定的长度 LEN 是否与此功能代码的长度相符。

如果不是，则不会执行作业，并在 BSEND 的输出 ERROR 上生成一个边沿信号作为结束。

当使用上面列出的功能代码之外的其他功能代码时，也不会执行激活的作业，通过 BSEND 上的 ERROR 来结束该作业。

请求消息帧中的元素“字节计数器”和“CRC 校验”是由 CP 生成的，不需要 BSEND 源 DB 中的条目。

写入功能的作业完成

对于写入功能代码，在接收到应答消息帧且无错误后，激活的 BSEND 完成。

这通过 BSEND 的输出 **DONE** 上的边沿信号传送到 SIMATIC 用户程序。

如果在消息交换过程中识别到**错误**，或者如果从站发送了**错误代码**应答消息帧，则通过输出 **ERROR** 处的边沿信号进行报告。

读取功能的作业完成

对于读取功能，在接收到应答消息帧且无错误，**并且**将接收的数据完全传送到 CPU 之后，激活的 BSEND 完成。

这通过 BSEND 的输出 **DONE** 上的边沿信号传送到 SIMATIC 用户程序。

此时，接收的数据已经在 CPU 中可用。

如果在消息交换过程中识别到**错误**，或者如果从站发送了**错误代码**应答消息帧，则通过输出 **ERROR** 处的边沿信号进行报告。

在这种情况下，不会传送任何接收数据到 CPU。

作业完成时的 SYSTAT 条目

对于这些实例，在作业完成时通过 BSEND 上的 **ERROR** 进行指示，同时在 **SYSTAT** 区域中输入附加的错误代码。

可以使用此错误代码确定错误的准确原因。

7.2.2 从 CP 到 CPU 通过 BRCV (CP 441-2) 的数据传送

通讯链接

参数 ID 描述了到通讯伙伴的唯一通讯链接。 必须在此处指定来自数据链接组态的本地 ID。

块关系

参数 R_ID 描述了通讯链接中唯一的块关系。

所有**读取**功能代码都需要 BRCV。
BRCV 上 **R_ID** 的参数分配必须与相应 BSEND 具有**相同的 R_ID**，该参数用于激活此作业（0 到 255 之间的任意值）。

以这种方式，您可以在 SIMATIC 用户程序中对多个 BSEND / BRCV 对进行编程。

然后将从 Modbus 从站中接收到的应答消息帧，根据此作业使用的 R_ID 存储在不同目标地址区域内。

数据目标地址

当 SFB BRCV 准备好接收数据时，它接受从 CP 中接收到的数据，然后将数据输入到在参数 **RD_1** 中指定的数据目标地址。 也就是说数据目标地址是变量。

如何显示数据接收

通过输出 **NDR** 上的边沿信号来通知用户在 CPU 中接收到数据。

此处，接收的数据块长度显示在参数 **LEN** 中。

整个 Modbus 作业的完成可以在 SFB BSEND 的输出 DONE 处识别。

如何处理错误

在发生接收或发送错误时，不会传送任何数据到 CPU。 在此实例中，通过输出 ERROR 上的边沿信号来指示 BSEND 已完成。

BRCV 目标 DB

通过读取功能代码接收到的用户数据输入到 BRCV 目标地址区域。

关于每个 BRCV 目标 DB 的详细描述可以在“功能代码 (页 51)”一章中找到。

输入数据的长度显示在 BRCV 的参数 LEN 中。

驱动程序的诊断

8

诊断功能

CP 的诊断功能使您可以快速地定位发生的错误。 可使用下列诊断功能：

- 通过 CP 的显示元件进行诊断
- 通过功能块的 STATUS 输出进行诊断
- 通过 SYSTAT 错误消息区域进行诊断（仅适用于 CP 441-2）
- CP 的诊断缓冲区

显示元件 (LED)

显示元件提供有关 CP 的操作模式和/或可能的错误状态的信息，并使您对所有的内部或外部错误以及接口特定错误有一个初步了解。

FB/SFB 的 STATUS 输出

每个功能块和系统功能块都有一个用于错误诊断的 STATUS 输出。 读取 STATUS 输出可以获得关于在通讯期间发生的错误的信息。 可以在用户程序中评估 STATUS 参数。

错误消息区 SYSTAT（仅适用于 CP 441-2）

错误消息区 SYSTAT 是 CP 441-2 上的存储区，CP 识别到的所有错误和事件都详细输入到此区域。 可以通过在用户程序中对系统功能块 STATUS 进行编程来读取 SYSTAT 区域。

CP 的诊断缓冲区

在 “错误/事件表 (页 101)”一章中描述的所有错误和事件也会输入到 CP 的诊断缓冲区内。CP 的手册说明了如何读取诊断缓冲区。

8.1 CP 341 上的诊断工具

8.1.1 通过 CP 341 的显示元件进行诊断

显示元件

CP 341 的显示元件提供了有关 CP 341 的信息。有以下显示功能可供使用：

组错误显示	
• SF（红色）	发生的错误或分配的新参数
特殊显示	
• TXD（绿色）	发送 (Send) 激活；在 CP 341 通过接口发送用户数据时亮起。
• RXD（绿色）	接收 (Receive) 激活；在 CP 341 通过接口接收用户数据时亮起。

组错误显示 SF

组错误显示 SF 始终会在通电后亮起，在初始化后熄灭。 如果已经为 CP 341 生成了参数分配数据，则当分配了新参数时 SF LED 再次短暂亮起。

当发生下列错误时，组错误显示 SF 亮起：

- 硬件错误
- 固件错误
- 参数分配错误
- 断路 (BREAK)（CP 341 和通讯伙伴之间的接收线路中断。）

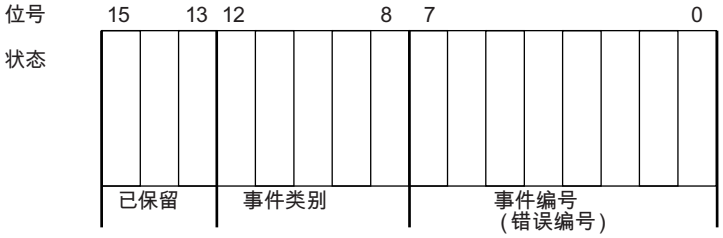
8.1.2 CP 341 功能块的诊断消息

简介

每个功能块都有一个用于错误诊断的 STATUS 参数。无论使用哪个功能块，STATUS 消息编号始终具有相同的含义。

事件类别/事件编号的编号方案

下图说明了 STATUS 参数的结构。



在 “错误/事件表 (页 101)”一章列出了各个错误/事件。

8.2 CP 441-2 上的诊断工具

8.2.1 通过 CP 441-2 的显示元件进行诊断

显示元件

CP 441-2 的显示元件提供了有关 CP 441-2 的信息。有以下显示功能可供使用：

组错误显示	
INTF	内部错误
EXTF	外部错误
特殊显示	
TXD	发送激活；指示灯在 CP 441-2 通过接口发送用户数据时亮起。
RXD	接收激活；指示灯在 CP 441-2 通过接口接收用户数据时亮起。
接口错误显示	
FAULT	接口错误

显示元件的错误消息

下表说明了显示元件的错误消息。

错误显示	错误说明	补救措施
INTF 亮起	CP 441-2 报告内部错误；例如，硬件或软件错误。	对 SFB STATUS 进行编程，以获取详细信息。
EXTF 亮起	CP 441 报告外部错误；例如，接收线路断路。	对 SFB STATUS 进行编程，以获取详细信息。
FAULT 熄灭	接口准备好开始工作，或接口子模块未插入。	-
FAULT 缓慢闪烁	接口已初始化并已准备好开始工作，但是无法通过 S7-400 背板总线进行通信。	检查常规组态和数据链接组态。
FAULT 快速闪烁	参数不正确或错误和/或插入了有故障的接口子模块。（子模块和接口参数不匹配）。	检查参数分配接口和/或接口子模块中的参数设置。
FAULT 亮起	没有接口参数可用，或子模块（硬件）中出现严重故障。	使用参数分配工具进行参数分配和/或检查接口子模块。

8.2.2 CP 441-2 的系统功能块的诊断消息

简介

每个系统功能块都有一个用于错误诊断的 STATUS 参数。 无论使用哪个系统功能块，每个 STATUS 消息编号都始终具有相同的含义。 下表说明了对 CP 最重要的一些 STATUS 消息。 您会在参考手册《S7-300/400 的系统软件、系统功能和标准功能》中找到 STATUS 消息最新的完整描述。

SFB 的 STATUS 输出处的消息

STATUS	错误说明
0	无错误
1	CP 和 CPU 之间的通讯问题
2	否定应答，不能执行功能；例如，链接伙伴无响应，或发送了否定应答。
3	此通讯链接中的 R-ID 未知，设备不可用。
4	数据区数量或各个数据类型不匹配。
5	接收到复位请求
6	远程块处于禁用状态
7	远程伙伴处于不正确的状态。
8	访问远程对象被拒绝，服务器中发生访问错误 (GET/PUT)。
9	过速警告 (ERROR=0)：接收数据被更新的数据覆盖。
10	无法访问本地用户存储器（例如 DB 已删除）。
11	警告 (ERROR=0)：由于前一个作业尚未完成，新作业未激活。
12	实例与系统调用不兼容；未调用实例，但是调用了常规 DB。
13	格式描述中出错
14	引用数据链接（应用程序相关）不存在，ID 未知。 必须指定来自数据链接组态的本地 ID。
15	生成通过 ID 引用的数据链接。
16	由于资源不足，无法生成数据链接。

显示和评估 STATUS 输出

可以使用 STEP 7 变量表来显示和评估系统功能块的 STATUS 输出。

说明

使用 STATUS 作业读取 SYSTAT 区域可以提供错误和事件的详细信息；这些错误和事件是在 CP、相关 CPU 和所连接的链接伙伴之间进行通讯期间发生的。

8.2.3 通过 CP 441-2 的错误消息区 SYSTAT 进行诊断

简介

错误消息区 SYSTAT 是 CP 441-2 上的数据区域，CP 识别到的所有错误和事件都详细输入到此区域。 SYSTAT 区由每个接口的六个事件，以及 CP 的工作状态相关的信息和 SYSTAT 区的状态组成。

读取 SYSTAT 区

可以使用 SFB STATUS 读取 SYSTAT 区。 读取当前事件所需的数据链路应在参数 ID 中输入。16 字节的诊断数据被传送到输出 LOCAL 的数据链路，点对点数据链路不使用参数 PHYS 和 LOG。

说明

由于 STATUS 请求与链接上运行的其余请求是异步执行的，因此无法将带有特定 R_ID 的 SFB 分配给错误消息。 这意味着，尽管 SYSTAT 可以显示数据链接上发生的错误，但是无法显示触发了错误的 SFB 调用。

SYSTAT 区的结构

CP 识别到的前 6 个错误/事件输入到 SYSTAT 区。 只有删除了 SYSTAT 区的内容之后才能输入更多错误/事件。

错误/事件按下述方式输入到参数 LOCAL 中：

字节 0	CP 的操作状态（RUN 为 02H，故障为 05H）	
字节 1	已保留	
字节 2	位 0 - F	输入到 SYSTAT 中的错误
	位 1 - U	错误溢出
	位 2 - B	断路 (BREAK)
字节 3	已保留	
字节 4/5	事件 1	
字节 6/7	事件 2	
字节 8/9	事件 3	
字节 10/11	事件 4	
字节 12/13	事件 5	
字节 14/15	事件 6	

正在删除 SYSTAT 区

使用 SFB STATUS 读取 SYSTAT 区域后，将自动删除所有 SYSTAT 消息。

编号方案

SYSTAT 区中事件的编号方案结构如下：

位号	15		13	12				8	7							0
	已保留			事件类别					事件编号/错误编号							

可以在下列各章节以及手册《CP 441-2：点对点通信》中找到事件类别和事件编号的详细分类。

说明

与标准驱动程序相反，为了与可加载的驱动程序一起使用，对 SYSTAT 区的事件类别/事件编号作了部分修改。

下列章节描述了所有**修改的事件级别/事件号**的**驱动程序特定的含义**。

除非在本手册中提到的事件，否则都可以假设该事件对应标准数据链接，并在 CP 441-2 手册中进行说明。

8.3 错误/事件表

事件类别

定义了下列事件类别：

事件类别	描述	描述位置
1	CP 上的硬件错误	CP 手册
2	初始化时出错	CP 手册
3	PBK 的参数分配期间出错	CP 手册
4	CP 检测到 CP - CPU 数据传输中出错	CP 手册
5	处理 CPU 作业期间出错	CP 手册，驱动程序手册
6	处理伙伴作业期间出错	CP 手册
7	发送错误	CP 手册
8	接收错误	驱动程序手册
9	从链接伙伴接收到的错误消息帧	未使用
10	CP 在来自伙伴的应答消息帧中检测到的错误	未使用
14	可加载驱动程序的常规处理错误	驱动程序手册

8.3.1 SYSTAT 中的“CPU 作业错误”错误代码

事件类别 5 (05H) “CPU 作业错误”			
事件类别/编号（十六进制）	事件编号(十进制)	事件文本	补救措施
05 18H	24	• 传送期间的传输长度过大 (> 4 KB)，或者 • SEND 的传输长度过小。	检查 SEND 的参数 LEN。

8.3.2 SYSTAT 中的“接收错误”错误代码

事件类别 8 (08H) “接收错误”			
事件类别/编号（十六进制）	**事件编号（十进制）**	**事件文本**	**补救措施**
08 06H	6	超过字符延迟时间 (ZVZ)	消除伙伴设备中的故障或对传输线路的干扰。
08 0CH	12	在字符中识别到传输错误（奇偶校验错误、溢出错误、停止位错误（帧））。	检查干扰是否会影响传输线路。如果需要，请更改系统结构和/或电缆铺设。 检查在 CP 和链接伙伴上是否设置了相同的协议参数，如传输速率、数据位数、奇偶校验、停止位数等。
08 0DH	13	断路 (BREAK) 连接到伙伴设备的接收线路中断。	在设备之间建立连接或打开伙伴设备。 对于与 TTY 操作一起使用，检查空闲状态的线路电流。 对于与 RS422/485 (X27) 连接一起使用，检查并在需要时更改 2 线接收线路 R(A) 和 R(B) 的连接器针脚分配。
08 30H	48	已经发送出请求消息，应答监视时间已过，但未识别到应答消息的起始头。	检查传输线路是否中断（可能需要接口分析器）。 检查在 CP 和链接伙伴上是否设置了相同的协议参数，如传输速率、数据位数、奇偶校验、停止位数等。 检查通过 PtP_PARAM 设置的应答监视时间值是否足够大。 检查指定的从站地址是否存在。

事件类别 8 (08H) “接收错误”			
事件类别/编号（十六进制）	**事件编号（十进制）**	**事件文本**	**补救措施**
08 31H	49	来自从站的应答消息中的第一个字符与在请求消息中发送的从站地址不同（对于操作模式“正常”）。	错误从站已应答。 检查传输线路是否中断（可能需要接口分析器）。
08 32H	50	接收应答消息期间，CP 的接收缓冲区溢出。	检查从站的协议设置。

8.3.3 SYSTAT 中“常规处理错误”错误代码

事件类别 14 (0EH) “可加载的驱动程序 — 常规处理错误”			
事件类别/编号（十六进制）	**事件编号（十进制）**	**事件文本**	**解决方法**
0E 01H	1	**在初始化驱动程序特定的 SCC 进程期间出错。**	重新分配驱动程序的参数，然后重新装载。
0E 02H	2	**启动驱动程序期间出错：** 激活了错误的 SCC 进程（SCC 驱动程序）。 驱动程序无法与此 SCC 驱动程序一起工作。	重新分配驱动程序的参数，然后重新装载。
0E 03H	3	**启动驱动程序期间出错：** 激活了错误的数据传送过程（连接到 SFB）。 驱动程序无法与此数据传送进程一起工作。	重新分配驱动程序的参数，然后重新装载。
0E 04H	4	**启动驱动程序期间出错：** 接口子模块非法。 驱动程序无法与参数设置的接口子模块一起工作。	检查并更正参数分配。
0E 05H	5	**驱动程序软件狗出错：** 未插入软件狗，或插入的软件狗有故障。 驱动程序未就绪。	检查是否在 CP 中插入了驱动程序软件狗。 如果插入的软件狗有故障，请使用正确的软件狗替换。

事件类别 14 (0EH) “可加载的驱动程序 — 常规处理错误”			
事件类别/编号（十六进制）	**事件编号（十进制）**	**事件文本**	**解决方法**
0E 06H	6	**驱动程序软件狗出错：** 软件狗无有效内容。 驱动程序未就绪。	从为您提供此驱动程序的西门子办事处获取正确的软件狗。
:	:		
0E 10H	16	**程序内部错误：** 在自动化设备的程序中跳转失效 (default branch)。	重新启动 CP (Power_On)
0E 11H	17	**程序内部错误：** 在程序状态发送 (Send)/接收 (Receive) 中跳转失效 (default branch)。	重新启动 CP (Power_On)
0E 12H	18	**主动自动化设备内部错误：** 跳转失效 (default branch)	重新启动 CP (Power_On)
0E 13H	19	**被动自动化设备内部错误：** 跳转失效 (default branch)	重新启动 CP (Power_On)

事件类别 14 (0EH) **“可加载的驱动程序 - 常规处理错误 <参数分配>”**			
事件类别/编号（十六进制）	**事件编号（十进制）**	**事件文本**	**解决方法**
0E 20H	32	**对于此数据链接，数据位数必须设置为 8。** 驱动程序未就绪。	更正驱动程序的参数分配。
0E 21H	33	**字符延迟时间的倍增因子设置不在数值范围 1 到 10 内。** 驱动程序使用缺省设置 1 工作。	更正驱动程序的参数分配。
0E 22H	34	**为驱动程序设置的操作模式非法**。 必须指定“常规操作”或“干扰抑制”。 驱动程序未就绪。	更正驱动程序的参数分配。
0E 23H	35	**为应答监视时间设置了非法值：** 有效值是 5 到 65500 ms。 驱动程序未做好运行准备。	更正驱动程序的参数分配。
:	:		
0E 2EH	46	**读取接口参数文件时发生错误。** 驱动程序未就绪。	重新启动 CP (Power_On)。

事件类别 14 (0EH) “可加载的驱动程序 - 常规处理错误 <CPU-CP>”			
事件类别/编号（十六进制）	**事件编号（十进制）**	**事件文本**	**解决方法**
0E 30H	48	**数据传送到 CPU 期间发生内部错误：** 意外的否定确认。	如果只是间断性地发生，可以忽略。
0E 31H	49	**传送数据到 CPU 时超时。**	检查 CP-CPU 接口。
0E 32H	50	**通过 RCV 传送数据到 CPU 期间发生错误：** 准确故障原因（详细错误）在此条目前面的 SYSTAT 中。	检查 CP-CPU 接口。
0E 33H	51	**数据传送到 CPU 期间发生内部错误：** 自动化设备的状态非法	检查 CP-CPU 接口。
:	:		
0E 3CH	60	**此驱动程序的作业非法。**	只允许 SFB SEND、RCV、STATUS（仅 CP 441-2）。

事件类别 14 (0EH) “可加载的驱动程序 - 常规处理错误 <处理 BSEND 作业>”			
事件类别/编号（十六进制）	**事件编号（十进制）**	**事件文本**	**解决方法**
0E 40H	64	**SFB SEND 处为参数 LEN 指定的数值太小。**	最小长度是 2 个字节。
0E 41H	65	**SFB SEND 处为参数 LEN 指定的数值太小。** 传送的功能代码需要更大长度。	该功能代码的最小长度是 6 个字节。
0E 42H	66	**传送的功能代码非法。**	仅允许使用“功能代码 (页 51)”部分所列的功能代码。
0E 43H	67	**此功能代码不允许使用从站地址 0（= 广播）。**	仅对合适的功能代码使用从站地址 0。
0E 44H	68	**传送的参数“位数”的数值不在范围 1 到 2040 内。**	参数“位数”值必须在范围 1 到 2040 内。
0E 45H	69	**传送的参数“寄存器数”的值不在范围 1 到 127 内。**	参数“寄存器数”值必须在范围 1 到 127 内。

事件类别 14 (0EH) “可加载的驱动程序 - 常规处理错误 <处理 BSEND 作业>”			
事件类别/编号（十六进制）	**事件编号（十进制）**	**事件文本**	**解决方法**
0E 46H	70	**功能代码 15 或 16：** **传送的参数“位数”和/或“寄存器数”的值不在范围 1 到 2040 和/或 1 到 127 内。**	参数“位数”和/或“寄存器数”必须在范围 1 到 2040 和/或 1 到 127 内。
0E 47H	71	**功能代码 15 或 16：** **SFB BSEND 的参数 LEN 与传送的参数“位数”和/或“寄存器数”不对应。** 参数 LEN 太小。	增大 SEND 的参数 LEN，直到将足量的用户数据传送到 CP 为止。 由于“位数”和/或“寄存器数”所限，必须将更多的用户数据传送到 CP。
0E 48H	72	**功能代码 05：** **在 SEND 源 DB 中为“设置位”(FF00H) 或“删除位”(0000H) 指定的代码错误。**	允许的代码只有 FF00H 和 0000H。
0E 49H	73	**功能代码 08：** **在 SEND 源 DB 中为“诊断代码”指定的代码错误。**	允许的代码只有“诊断代码” 0000H。
:	:		
0E 4FH	79	**CP 441：** **为 SFB SEND 指定的 R_ID 对于此驱动程序非法。**	只使用 0 到 255 (00 00 00 00 ... 00 00 00 FFH) 的 R_ID 值
		CP 341： **为 SFB SEND 指定的 R_TYP 对于此驱动程序非法。**	“X”必须作为 R_TYP 输入。

事件类别 14 (0EH) “可加载的驱动程序 - 常规处理错误 <接收评估>”			
事件类别/编号（十六进制）	**事件编号（十进制）**	**事件文本**	**解决方法**
0E 50H	80	**从站地址不正确：** 接收的从站地址与发送从站地址不同。	错误从站已应答。 检查传输线路是否中断（可能需要接口分析器）。
0E 51H	81	**功能代码不正确：** 在应答消息中收到的功能代码与发送的功能代码不同。	检查从站设备。
0E 52H	82	**字节下溢：** 接收到的字符数少于应该从应答消息的字节计数器中返回的字符数，或者少于此功能代码期望的字符数。	检查从站设备。
0E 53H	83	**字节溢出：** 接收到的字符数大于应该从应答消息的字节计数器中返回的字符数，或者大于此功能代码期望的字符数。	检查从站设备。
0E 54H	84	**字节计数器错误：** 应答消息中接收到的字节计数器太小。	检查从站设备。
0E 55H	85	**字节计数器错误：** 应答消息中接收到的字节计数器错误。	检查从站设备。
0E 56H	86	**应答错误：** 从站返回的应答消息的数据（位数、...）与请求消息中发送的数据不同。	检查从站设备。
0E 57H	87	**CRC 校验不正确：** 在检查从站应答消息的 CRC 16 校验和时出错。	检查从站设备。

事件类别 14 (0EH) “可加载驱动程序 - 一般处理错误 <接收异常代码消息>”			
事件类别/编号（十六进制）	**事件编号（十进制）**	**事件文本**	**解决方法**
0E 61H	97	**异常代码为 01 的应答消息： 功能非法**	参见“从站设备手册”
0E 62H	98	**异常代码为 02 的应答消息： 数据地址非法**	参见“从站设备手册”
0E 63H	99	**异常代码为 03 的应答消息： 数据值非法**	参见“从站设备手册”
0E 64H	100	**功能代码为 04 的应答消息： 相关设备故障**	参见“从站设备手册”
0E 65H	101	**异常代码为 05 的应答消息： 确认**	参见“从站设备手册”
0E 66H	102	**异常代码为 06 的应答消息： 忙，拒绝消息**	参见“从站设备手册”
0E 67H	103	**异常代码为 07 的应答消息： 否定应答**	参见“从站设备手册”

应用示例

9

9.1 CP 341 的应用示例

9.1.1 CP 341 的应用示例

常规信息

下面的简单程序实例说明了 FB P_SND_RK 和 P_RCV_RK 的使用方法。

在安装 Modbus 主站时，程序实例存储在 STEP 7 目录 EXAMPLES 中，名称 *Modma* 下。

S7 程序仅用于信息说明，并不能把它当成客户特定安装组态的解决方案。

为了说明基本结构，我们有意将示例程序简单化，避免使用符号显示。

9.1.2 使用的块

使用的块

该程序实例使用了下列块。

块	符号	注释
OB 1	循环执行	循环程序处理
OB 100	完全重启动	启动 OB 以重新启动
FC 10	初始化	用于启动 OB 的 FC
FC 21	执行发送作业	FC 调用 SFB P_SND_RK
FC 23	执行接收作业	FC 调用 SFB P_RCV_RK
DB 50	IDB_P_SND_RK	P_SND_RK 的背景数据块
DB 70	IDB_P_RCV_RK	P_RCV_RK 的背景数据块
DB 40	工作 DB 发送	用于 FC21 和 P_SND_RK 的工作 DB
DB 41	工作 DB 接收	用于 FC23 和 P_RCV_RK 的工作 DB
DB 42	SOURCE_DB	P_SND_RK 源 DB，带发送数据
DB 43	DESTINATION_DB	P_RCV_RK 源 DB，用于接收数据

使用的数据

在程序实例中使用了下列操作数（存储器位、数据位或数据字）。

操作数	注释
M120.7	执行 P_SND_RK 作业的触发位
DB40.DBX 0.0	控制参数 REQuest： 用于激活 P_SND_RK
DB40.DBX 0.1	控制参数 Reset： 用于中止当前 P_SND_RK
DB40.DBX 0.4	DONE 状态参数： 指示当前 P_SND_RK 已经完成且无错误
DB40.DBX 0.5	ERROR 状态参数： 指示当前 P_SND_RK 已经完成且有错误
DB40.DBW 6	P_SND_RK 的成功计数器
DB40.DBW 8	P_SND_RK 的错误计数器
DB40.DBW 10	要传送到 CP 的 P_SND_RK 源数据区的长度 LEN，以字节为单位
DB40.DBW 12	P_SND_RK 中的 STATUS 显示
DB40.DBW 14	存储的 P_SND_RK STATUS 显示
DB41.DBX 0.0	控制参数 EN_R： P_RCV_RK 接收准备就绪
DB41.DBX 0.4	NDR 状态参数： 指示当前 P_RCV_RK 已经从 CP 中接收到新数据
DB41.DBX 0.5	ERROR 状态参数： 指示当前 P_RCV_RK 已经完成且有错误
DB41.DBW 4	存储的 P_RCV_RK 的长度 LEN
DB41.DBW 6	P_RCV_RK 的成功计数器
DB41.DBW 8	P_RCV_RK 的错误计数器
DB41.DBW 10	CP 接收到的 P_RCV_RK 目标数据区的长度 LEN ， 以字节为单位
DB41.DBW 12	P_RCV_RK 中的 STATUS 显示
DB41.DBW 14	存储的 P_RCV_RK STATUS 显示

9.1.3 程序描述

常规信息

该程序实例由以下内容组成：

- 启动块 OB100，FC10
- 循环部分 OB1 调用
- 用于将数据从 CPU 传送到 CP 的功能块 FC21（**发送**）
- FC23 用于从 CP **接收**数据到 CPU

已编程的系统功能块 P_SND_RK、P_RCV_RK 的参数存储在工作 DB **DB 40** 和 **DB 41** 中。

发送数据（SEND 源区域）包含在 **DB 42** 中。

从链接伙伴接收的数据输入到 **DB 43** （RCV 目标区域）中。

P_SND_RK 作业

P_SND_RK 作业可以在程序的循环部分中，通过置位存储器位 **M 120.7** 来激活（例如，通过 CONTROL VARIABLE）。 将 P_SND_RK 源数据区 DB42 中包含的、长度为 LEN 的数据传送到 CP。
触发位 M 120.7 立即复位。

在完成 P_SND_RK 作业且无错误之后，成功计数器加 1；而在完成作业且有错误后，错误计数器加 1。

P_RCV_RK 作业

在 FC23 中编程 SFB **P_RCV_RK** ；而在 FC23 中，**接收使能**始终为**"1"**，以便从链路伙伴中接收数据。 接收数据输入到 **P_RCV_RK 目标区域**，输入的数据量显示在参数 **LEN** 中。

在获得数据且无错误之后，成功计数器加 1；而在完成且有错误后，错误计数器加 1。

对于 P_SND_RK 和 P_RCV_RK 作业，当报告非 0 值时存储输出参数 **STATUS**。

9.1.4 程序实例

程序实例

列出了如下块：

块	注释
OB 100	启动 OB 以重新启动
FC 10	启动用于 OB 100 的 FC
OB 1	循环程序处理
FC 21	FC 调用 FB P_SND_RK
FC 23	FC 调用 FB P_RCV_RK

程序启动

```
OB 100 完全重启动
L 272                                   //逻辑地址
T DB40.DBW 16                           //用于 SEND
T DB40.DBW 16                           //和 RCV
UC FC 10                                //为初始化调用 FC
```

FC 10 初始化

```
// ----------------------------
// 复位控制位
// ----------------------------
 L B#16#0
 T DB40.DBW 0                          //发送工作 DB
 T DB40.DBW 0                          //接收工作 DB
// ----------------------------
// 复位计数器/STATUS
// ----------------------------
 L W#16#0
 T DB40.DBW 6                          //发送工作 DB
 T DB40.DBW 8
 T DB40.DBW 12
 T DB40.DBW 14
 T DB40.DBW 6                          //接收工作 DB
 T DB40.DBW 8
 T DB40.DBW 12
 T DB40.DBW 14
```

循环程序序列

OB 1 可循环 OB

```
 UC FC 21                                  //调用 SEND
 UC FC 23                                  //调用 RCV
```

FC 21 执行 SEND 作业

```
// ----------------------------
// SEND 的联锁
// ----------------------------
 A M 120.7                                 //触发 SEND
 AN DB40.DBX 0.0                           //SEND_REQ
 AN DB40.DBX 0.4                           //SEND_DONE
 AN DB40.DBX 0.5                           //SEND_ERROR
 R M 120.7                                 //复位触发 SEND
 S DB40.DBX 0.0                            //置位 SEND_REQ
// ----------------------------
// 生成边沿 SEND_REQ
// ----------------------------
 A(
 O DB40.DBX 0.4                            //SEND_DONE
 O DB40.DBX 0.5                            //SEND_ERROR
 )
 A DB40.DBX 0.0                            //SEND_REQ
 R DB40.DBX 0.0                            //通过 REQ=0 执行 SEND
// ----------------------------
// 提供 LEN
// ----------------------------
 L W#16#20                                 // SEND 数据的长度
 T DB40.DBW 10                             //SEND-LEN
// ----------------------------
// SEND，带背景数据块
// ----------------------------
 CALL FB 8 , DB50
 SF :=
 REQ :=DB40.DBX0.0
 R :=DB40.DBX0.1
 LADDR :=DB40.DBW16
 DB_NO :=42
```

FC 21 执行 SEND 作业

```
 DBB_NO :=10
 LEN :=DB40.DBW10
 R_CPU_NO :=
 R_TYP :='X'
 R_NO :=
 R_OFFSET :=
 R_CF_BYT :=
 R_CF_BIT :=
 DONE :=DB40.DBX0.4
 ERROR :=DB40.DBX0.5
 STATUS :=DB40.DBW12
// ----------------------------
// 选中"完成且无错误"
// ----------------------------
 A DB40.DBX 0.4                          //DONE ?
 JCN CON1                                //如果为 NO
 L DB40.DBW 6                            //"完成且无错误"
 +1                                      //增量计数器
 T DB40.DBW 6
 :                                       // :
 :                                       //更多用户
 :                                       // 功能
 :                                       // :
 JU LEAV
// ----------------------------
// 选中"完成且有错误"
// ----------------------------
CON1: A DB40.DBX 0.5                     //是否有错?
 JCN CON2                                //如果为 NO
 L DB40.DBW 8                            //"完成且有错误"
 +1                                      //增量计数器
 T DB40.DBW 8
 :                                       // :
 :                                       //错误处理
 :                                       // :
 L 0
 L DB40.DBW 12                           //如果 STATUS <>0
 ==I
 JC LEAV
 T DB40.DBW 14                           //保存 STATUS
```

FC 21 执行 SEND 作业

```
 JU LEAV
// ---------------------------
// 检查"STATUS 中的错误"
// ---------------------------
CON2: L 0
 L DB40.DBW 12                          //如果 STATUS <>0
 ==I
 JC LEAV
 T DB40.DBW 14                          //保存 STATUS
 :                                      // :
 :                                      //错误处理
 :                                      // :
 LEAV:CLR
```

FC 23 执行 RCV-Receive

```
// ---------------------------
// 使能接收数据
// ---------------------------
 SET
 = DB41.DBX 0.0                         //通过 EN_R=1 执行 RCV
// ---------------------------
// RCV 带背景数据块
// ---------------------------
 CALL FB 7 , DB70
 EN_R :=DB41.DBX0.0
 R:=
 LADDR :=DB41.DBW16
 DB_NO := 43
 DBB_NO :=0
 L_TYP :=
 L_NO :=
 L_OFFSET :=
 L_CF_BYT :=
 L_CF_BIT :=
 NDR :=DB41.DBX0.4
 ERROR :=DB41.DBX0.5
 LEN :=DB41.DBW10
 STATUS :=DB41.DBW12
```

FC 23 执行 RCV-Receive

```
// ----------------------------
// 选中"接收且无错误"
// ----------------------------
 A DB41.DBX 0.4                          //NDR ?
 JCN CON1                                //如果为 NO
 L DB41.DBW 6                            //"接收且无错误"
 +1                                      //增量计数器
 T DB40.DBW 6
 L DB41.DBW 10                           //保存
 T DB40.DBW 4                            //接收长度 LEN
 JU LEAV
// ----------------------------
// 选中"接收且有错误"
// ----------------------------
CON1: A DB41.DBX 0.5                     //是否有错?
 JCN CON2                                //如果为 NO
 L DB41.DBW 8                            //"接收且有错误"
 +1                                      //增量计数器
 T DB40.DBW 8
 L 0
 L DB41.DBW 12                           //如果 STATUS <>0
 ==I
 JC LEAV
 T DB40.DBW 14                           //保存 STATUS
 JU LEAV
// ----------------------------
// 检查"STATUS 中的错误"
// ----------------------------
CON2: L 0
 L DB41.DBW 12                           //如果 STATUS <>0
 ==I
 JC LEAV
 T DB40.DBW 14                           //保存 STATUS
LEAV:CLR
```

9.2 CP 441-2 的应用示例

常规信息

下面的简单程序实例说明了 SFB BSEND、BRCV 和 STATUS 的使用方法。

当安装了 Modbus 主站时，程序实例存储在 STEP 7 目录 EXAMPLES 的名称 *Modma* 下。

S7-400 程序仅用于说明目的，并不能理解为用户特定的安装组态的解决方案。

为了说明基本结构，我们有意将示例程序简单化，避免使用符号显示。

9.2.1 使用的块

使用的块

该程序实例使用了下列块。

块	符号	注释
OB 1	循环执行	循环程序处理
OB 100	完全重启动	启动 OB 以重新启动
FC 100	初始化	用于启动 OB 的 FC
FC 210	执行 BSEND 作业	FC 调用 SFB BSEND
FC 230	执行 BRCV 作业	FC 调用 SFB BRCV
FC 250	执行 STATUS 作业	FC 调用 SFB STATUS
DB 22	IDB_STATUS	STATUS 的背景数据块
DB 50	IDB_BSEND	BSEND 的背景数据块
DB 70	IDB_BRCV	BRCV 的背景数据块
DB 400	工作 DB BSEND	FC 210 和 BSEND 的工作 DB
DB 401	工作 DB BRCV	FC 230 和 BRCV 的工作 DB
DB 410	SOURCE_DB	BSEND 源 DB，带发送数据

块	符号	注释
DB 430	DESTINATION_DB	BRCV 目标 DB，用于接收数据
DB 450	工作 DB STATUS	FC 250 和 STATUS 的工作 DB 和 SYSTAT 接收 DB

使用的数据

在程序实例中使用了下列操作数（存储器位、数据位或数据字）。

操作数	注释
M 119.7	执行 STATUS 作业的触发位
M 120.7	执行 BSEND 作业的触发位
DB400.DBX 0.0	控制参数 REQuest： • 用于激活 BSEND
DB400.DBX 0.1	控制参数 Reset： • 用于中止当前 BSEND
DB400.DBX 0.4	DONE 状态参数： • 指示当前 BSEND 已完成且无错误
DB400.DBX 0.5	ERROR 状态参数： • 指示当前 BSEND 已完成且有错误
DB400.DBW 6	BSEND 的成功计数器
DB400.DBW 8	BSEND 的错误计数器
DB400.DBW 10	要传送到 CP 的 BSEND 源数据区的长度 LEN，以字节为单位
DB400.DBW 12	BSEND 中的 STATUS 显示
DB400.DBW 14	存储的 BSEND STATUS 显示
DB400.DBD 16	BSEND 的参数 R_ID
DB401.DBX 0.0	控制参数 EN_R： • BRCV 接收准备就绪
DB401.DBX 0.4	NDR 状态参数： • 指示当前 BRCV 已经从 CP 中接收到新数据

操作数	注释
DB401.DBX 0.5	ERROR 状态参数： • 指示当前 BRCV 已经完成且有错误
DB401.DBW 4	存储的 BRCV 的长度 LEN
DB401.DBW 6	BRCV 的成功计数器
DB401.DBW 8	BRCV 的错误计数器
DB401.DBW 10	CP 接收到的 BRCV 目标数据区的长度 LEN，以字节为单位
DB401.DBW 12	BRCV 中的 STATUS 显示
DB401.DBW 14	存储的 BRCV STATUS 显示
DB401.DBD 16	BRCV 的参数 R_ID
DB450.DBX 0.0	控制参数 REQuest： • 用于激活 STATUS
DB450.DBX 0.4	NDR 状态参数： • 指示 STATUS 已经从 CP 中获得新 SYSTAT 数据
DB450.DBX 0.5	ERROR 状态参数： • 指示当前 STATUS 已经完成且有错误
DB450.DBW 6	STATUS 的成功计数器
DB450.DBW 8	STATUS 的错误计数器
DB450.DBW 12	STATUS 中的 STATUS 显示
DB450.DBW 14	存储的 STATUS 显示
DB450.DBW 16	参数 PHYS： • 接口的物理状态（未用于点到点链接）
DB450.DBW 18	参数 LOG： 接口的逻辑状态（未用于点到点链接）
DB450.DBW 20 ： DB450.DBW 34	参数 LOCAL： • 接口的接收 SYSTAT 区的目的区域
DB450.DBW 40 ： DB450.DBW 54	存储的 SYSTAT 区

9.2.2 程序描述

常规信息

该程序实例由以下内容组成：

- 启动块 OB 100、FC 100
- 循环部分 OB1 调用
- 用于从 CPU 传送数据到 CP 的功能块 FC 210（**发送**）
- FC 230 用于从 CP **接收**数据到 CPU
- FC 250 用于读取 **SYSTAT** 区。

已编程的系统功能块 BSEND、BRCV 和 STATUS 的参数存储在工作 DB **DB 400** (BSEND)、**DB401** (BRCV) 和 **DB450** (STATUS) 中。

发送数据（BSEND 源区域）包含在 **DB 410** 中。 从链路伙伴中接收的数据输入到 **DB 430**（BRCV 目的区域）。

指定值 **1000** （十六进制）作为 SFB BSEND、BRCV 和 STATUS 的 **ID**。 在**数据链接组态**中，ID 是从 1000（十六进制）开始编号的。 如果您的数据链接导致不同 ID，则必须为合适的 SFB 指定此 ID。 （还可以参见“数据链接的组态 (页 29)”一节）。

BSEND 作业

BSEND 作业可以在程序的循环部分中，通过置位存储器位 **M 120.7** 来激活（例如，通过 CONTROL VARIABLE）。 将 BSEND 源数据区 DB 410 中包含的长度为 LEN 的数据传送到 CP。触发位 M 120.7 立即复位。

在完成 BSEND 作业且无错误之后，成功计数器加 1；而在完成作业且有错误后，错误计数器加 1。

BRCV 作业

在 FC 230 中编程 SFB **BRCV** ，**接收使能**始终为**“1”**，以便从链接伙伴中接收数据。 接收数据输入到 **P_RCV_RK 目标区域**，输入的数据量显示在参数 **LEN** 中。

在获得数据且无错误之后，成功计数器加 1；而在完成且有错误后，错误计数器加 1。

读 SYSTAT

SYSTAT 读作业可以通过置位存储器位 **M 119.7** 来激活（例如，通过 CONTROL VARIABLE）。 触发位立即复位。

在 FC 250 中编程 SFB **STATUS**，它将 CP 的 SYSTAT 区域中的数据输入到 STATUS 块中指定的目标区域内。

在完成 STATUS 作业且无错误之后，成功计数器加 1；而在完成且有错误后，错误计数器加 1。

对于 BSEND、BRCV 和读 SYSTAT 作业，当报告一个非 0 数值时保存输出参数 **STATUS**。

9.2.3 程序实例

程序实例

列出了如下块：

块	注释
OB 100	启动 OB 以重新启动
FC 100	启动 OB 100 的 FC
OB 1	循环程序处理
FC 210	FC 调用 SFB BSEND
FC 230	FC 调用 SFB BRCV
FC 250	FC 调用 SFB STATUS

程序启动

OB 100 完全重启动

```
调用 FC 100                                    //初始化
```

FC 100 初始化

```
// ----------------------------
// 复位控制位
// ----------------------------
 L B#16#0
 T DB400.DBB 0                                 //BSEND 工作 DB
 T DB401.DBB 0                                 //BRCV 工作 DB
 T DB450.DBB 0                                 //STATUS 工作 DB
// ----------------------------
// 复位计数器/STATUS
// ----------------------------
 L W#16#0
 T DB400.DBW 6                                 //BSEND 工作 DB
 T DB400.DBW 8
 T DB400.DBW 12
 T DB400.DBW 14
 T DB401.DBW 6                                 //BRCV 工作 DB
```

FC 100 初始化

```
 T DB401.DBW 8
 T DB401.DBW 12
 T DB401.DBW 14
 T DB450.DBW 6                          //STATUS 工作 DB
 T DB450.DBW 8
 T DB450.DBW 12
 T DB450.DBW 14
```

循环程序序列

OB 1 可循环的 OB

```
 UC FC 210                              //调用 BSEND
 UC FC 230                              //调用 BRCV
 UC FC 250                              //调用 STATUS
```

FC 210 执行 BSEND 作业

```
// ----------------------------
// BSEND 的联锁
// ----------------------------
 A M 120.7                              //触发 BSEND
 AN DB450.DBX 0.0                       //REQuest STATUS
 AN DB400.DBX 0.0                       //BSEND_REQ
 AN DB400.DBX 0.4                       //BSEND_DONE
 AN DB400.DBX 0.5                       //BSEND_ERROR
 R M 120.7                              //复位触发 BSEND
 S DB400.DBX 0.0                        //设置 BSEND_REQ
// ----------------------------
// 生成边沿 BSEND:EQ
// ----------------------------
 A(
 O DB400.DBX 0.4                        //BSEND_DONE
 O DB400.DBX 0.5                        //BSEND_ERROR
 )
 A DB400.DBX 0.0                        //BSEND_REQ
 R DB400.DBX 0.0                        //通过 REQ=0 执行 BSEND
// ----------------------------
// 提供 R_ID, LEN
// ----------------------------
```

FC 210 执行 BSEND 作业

```
 L DW#16#1                                  //使用 R_ID = 1
 T DB400.DBD 16                             //作为 BSEND-R_ID
 L W#16#6                                   // BSEND 数据的长度
 T DB400.DBW 10                             //BSEND-LEN
// ---------------------------
// BSEND，带背景数据块
// ---------------------------
 CALL SFB 12 , DB50
 REQ :=DB400.DBX0.0
 R :=DB400.DBX0.1
 ID :=W#16#1000
 R_ID :=DB400.DBD16
 DONE :=DB400.DBX0.4
 ERROR :=DB400.DBX0.5
 STATUS :=DB400.DBW12
 SD_1 :=P#DB410.DBX 10.0 WORD 1
 LEN :=DB400.DBW10
// ---------------------------
// 选中"完成且无错误"
// ---------------------------
 A DB400.DBX 0.4                            //DONE ?
 JCN CON1                                   //如果为 NO
 L DB400.DBW 6                              //"完成且无错误"
 +1                                         //增量计数器
 T DB400.DBW 6
 :                                          // :
 :                                          //更多用户功能
 :                                          // :
 JU LEAV
// ---------------------------
 选中"完成且有错误"
// ---------------------------
CON1 A DB400.DBX 0.5                        //ERROR ?
 JCN CON2                                   //如果为 NO
 L DB400.DBW 8                              //"完成且有错误"
 +1                                         //增量计数器
 T DB400.DBW 8
 :                                          // :
 :                                          //错误处理
 :                                          // :
```

FC 210 执行 BSEND 作业

```
 L 0
 L DB400.DBW 12                          //如果 STATUS <>0
 ==I
 JC LEAV
 T DB400.DBW 14                          //保存 STATUS
 JU LEAV
// ---------------------------
选中“STATUS 有错误”
// ---------------------------
CON2: L 0
 L DB400.DBW 12                          //如果 STATUS <>0
 ==I
 JC LEAV
 T DB400.DBW 14                          //保存 STATUS
 :                                       // :
 :                                       //错误处理
 :                                       // :
LEAV: CLR
```

FC 230 执行 BRCV 接收

```
// ----------------------------
// 提供 R_ID
// ----------------------------
 L DW#16#1                                  //使用 BRCV-R_ID = 1
 T DB401.DBD 16                             //（与 BSEND-R_ID 相同）
// ----------------------------
// 使能接收数据
// ----------------------------
 SET
 = DB401.DBX 0.0                            //通过 EN_R=1 执行 BRCV
// -------------------------
// BRCV，带背景数据块
// -------------------------
 CALL SFB 13 , DB70
 EN_R :=DB401.DBX0.0
 ID :=W#16#1000
 R_ID :=DB401.DBD16
 NDR :=DB401.DBX0.4
 ERROR :=DB401.DBX0.5
 STATUS :=DB401.DBW12
 RD_1 :=P#DB430.DBX 0.0 WORD 128
 LEN :=DB401.DBW10
// --------------------------------
// 选中"接收且无错误"
// --------------------------------
 A DB401.DBX 0. 4                           //NDR ?
 JCN CON1                                   //如果为 NO
 L DB401.DBW 6                              //"接收且无错误"
 +1                                         //增量计数器
 T DB401.DBW 6
 L DB401.DBW 10                             //保存
 T DB401.DBW 4                              //接收长度 LEN
 JU LEAV
// ----------------------------
// 选中"接收且有错误"
// ----------------------------
```

FC 230 执行 BRCV 接收

```
CON1: A DB401.DBX 0.5                    //ERROR ?
 JCN CON2                                //如果为 NO
 L DB401.DBW 8                           //"接收且有错误"
 +1                                      //增量计数器
 T DB401.DBW 8
 L 0
 L DB401.DBW 12                          //如果 STATUS <>0
 ==I
 JC LEAV
 T DB401.DBW 14                          //保存 STATUS
 JU LEAV
// ----------------------------
// 选中"STATUS 有错误"
// ----------------------------
CON2: L 0
 L DB401.DBW 12                          //如果 STATUS <>0
 ==I
 JC LEAV
 T DB401.DBW 14                          //保存 STATUS
LEAV:CLR
```

FC 250 执行 STATUS 作业

```
// ---------------------------
// STATUS 的联锁
// ---------------------------
 A M 119.7                                  //触发 STATUS
 AN DB400.DBX 0.0                           //BSEND-REQ 激活?
 AN DB450.DBX 0.0                           //STATUS_REQ 激活?
 R M 119.7                                  //复位触发 STATUS
 S DB450.DBX 0.0                            //激活 STATUS_REQ
// ---------------------------
// 生成边沿 STATUS_REQ
// ---------------------------
 A(
 O DB450.DBX 0.4                            //STATUS_NDR
 O DB450.DBX 0.5                            //STATUS_ERROR
 )
 A DB450.DBX 0.0                            //STATUS_REQ
 R DB450.DBX 0.0                            //STATUS，带 REQ=0
// ---------------------------
// STATUS，带背景数据块（= 读 SYSTAT）
// ---------------------------
 CALL SFB 22 , DB22
 REQ :=DB450.DBX0.0
 ID :=W#16#1000
 NDR :=DB450.DBX0.4
 ERROR :=DB450.DBX0.5
 STATUS :=DB450.DBW12
 PHYS :=P#DB450.DBX 16.0 BYTE 2
 LOG :=P#DB450.DBX 18.0 BYTE 2
 LOCAL :=P#DB450.DBX 20.0 BYTE 16
// --------------------------------
// 选中"接收到新数据"
// --------------------------------
 A DB450.DBX 0.4                            //NDR ?
 JCN CON1                                   //如果为 NO
 L DB450.DBW 6                              //"完成且无错误"
 +1                                         //增量计数器
 T DB450.DBW 6
 A DB450.DBX 22.0                           //Bit0：存在错误?
 JCN LEAV                                   //如果为 NO
```

FC 250 执行 STATUS 作业

```
// ----------------------------
// 保存 SYSTAT
// ----------------------------
 L DB450.DBW 22
 T DBW 42
 L DBD 24
 T DBD 44
 L DBD 28
 T DBD 48
 L DBD 32
 T DBD 52
 JU LEAV
// ----------------------------
// 选中"完成且有错误"
// ----------------------------
CON1: A DB450.DBX 0.5                    //ERROR ?
 JCN CON2                                //如果为 NO
 L DB450.DBW 8                           //"完成且有错误"
 +1                                      //增量计数器
 T DB450.DBW 8
 L 0
 L DB450.DBW 12                          //如果 STATUS <>0:
 ==I
 JC LEAV
 T DB450.DBW 14                          //保存 STATUS
 JU LEAV
// ----------------------------
// 检查"STATUS 中有无错误"
// ----------------------------
CON2: L 0
 L DB450.DBW 12                          //如果 STATUS <>0
 ==I
 JC LEAV
 T DB450.DBW 14                          //保存 STATUS
LEAV:CLR
```

技术数据

A

A.1 技术数据

传输时间

下表包含了测量到的不同功能块的传输时间。

使用 S7-300 可编程控制器（带有 CPU 315-2 DP [6ES7315-2AF01-0AB0]、CP 341）以及作为伙伴设备的 S7-400 可编程控制器（带有 CPU 414 [6ES7414- 1XG01-0AB0]、CP 441-2）测量的时间。测量到下列时间:

- 处理时间（从在用户程序中初始化作业开始算起），包括主站上的处理时间在内，
- 作业通过串口传送到伙伴设备所需的时间
- 在从站上进行处理使用的时间，
- 在串口上传送确认所需要的时间。

必须将四个时间加在一起，才能计算出整个传输所使用的时间。

如果使用其他主站或从站（作为伙伴设备），则必须使用所使用的主站或从站的相应时间，而不是表格中的时间。 作业和确认的时间不变；它们只取决于使用的传输速率。

主站是 CP 341

功能代码 1（读） – 读线圈（输出）状态（时间单位为毫秒）

传输速率（波特）	300			
用户数据	主站 CP 341	作业	从站 CP 441-2	确认
1 个字节	236	257	188	184
10 个字节	236	257	190	515
20 个字节	238	257	190	882
50 个字节	244	257	193	1986
100 个字节	280	257	199	3824
200 个字节	286	257	207	7502
255 个字节	288	257	216	9487

传输速率（波特）	9600			
用户数据	主站 CP 341	作业	从站 CP 441-2	确认
1 个字节	33	8	40	6
10 个字节	33	8	43	16
20 个字节	35	8	44	28
50 个字节	42	8	45	62
100 个字节	56	8	56	120
200 个字节	75	8	64	235
255 个字节	82	8	77	296

传输速率（波特）	76800			
用户数据	主站 CP 341	作业	从站 CP 441-2	确认
1 个字节	35	1	23	1
10 个字节	36	1	25	2
20 个字节	37	1	26	3
50 个字节	46	1	27	8
100 个字节	61	1	30	15
200 个字节	82	1	39	29
255 个字节	92	1	48	37

主站是 CP 341

功能代码 15（写） – 写多个线圈（时间单位为毫秒）

传输速率（波特）	300			
用户数据	主站 CP 341	作业	从站 CP 441-2	确认
1 个字节	225	331	223	257
10 个字节	227	662	224	257
20 个字节	227	1030	228	257
50 个字节	227	2132	232	257
100 个字节	229	3971	236	257
200 个字节	230	7648	243	257
255 个字节	237	9634	255	257

传输速率（波特）	9600			
用户数据	主站 CP 341	作业	从站 CP 441-2	确认
1 个字节	64	11	62	8
10 个字节	64	21	63	8
20 个字节	69	32	64	8
50 个字节	69	67	68	8
100 个字节	72	124	70	8
200 个字节	75	239	76	8
255 个字节	75	301	86	8

传输速率（波特）	76800			
用户数据	主站 CP 341	作业	从站 CP 441-2	确认
1 个字节	60	1	56	1
10 个字节	60	3	58	1
20 个字节	62	4	58	1
50 个字节	64	9	60	1
100 个字节	67	16	67	1
200 个字节	72	30	77	1
255 个字节	77	38	84	1

主站是 CP 441

功能代码 1（读） – 读线圈（输出）状态（时间单位为毫秒）

传输速率（波特）	300			
用户数据	主站 CP 441-2	作业	从站 CP 341	确认
1 个字节	229	257	179	184
10 个字节	229	257	179	512
20 个字节	229	257	180	882
50 个字节	232	257	182	1986
100 个字节	236	257	192	3842
200 个字节	243	257	208	7501
255 个字节	251	257	214	9487

传输速率（波特）	9600			
用户数据	主站 CP 441-2	作业	从站 CP 341	确认
1 个字节	74	8	18	6
10 个字节	75	8	19	16
20 个字节	77	8	19	27
50 个字节	83	8	24	62
100 个字节	90	8	34	119
200 个字节	92	8	48	235
255 个字节	95	8	56	296

传输速率（波特）	76800			
用户数据	主站 CP 441-2	作业	从站 CP 341	确认
1 个字节	73	1	13	1
10 个字节	74	1	13	2
20 个字节	76	1	13	4
50 个字节	86	1	20	8
100 个字节	93	1	29	15
200 个字节	95	1	45	29
255 个字节	97	1	50	37

主站是 CP 441

功能代码 15（写） – 写多个线圈（时间单位为毫秒）

传输速率（波特）	300			
用户数据	主站 CP 441-2	作业	从站 CP 341	确认
1 个字节	205	331	199	257
10 个字节	206	662	200	257
20 个字节	206	1028	201	257
50 个字节	208	2132	212	257
100 个字节	211	3971	223	257
200 个字节	217	7648	238	257
255 个字节	221	9634	243	257

传输速率（波特）	9600			
用户数据	主站 CP 441-2	作业	从站 CP 341	确认
1 个字节	48	10	41	8
10 个字节	48	20	41	8
20 个字节	50	32	43	8
50 个字节	52	67	48	8
100 个字节	55	124	56	8
200 个字节	63	239	74	8
255 个字节	67	301	88	8

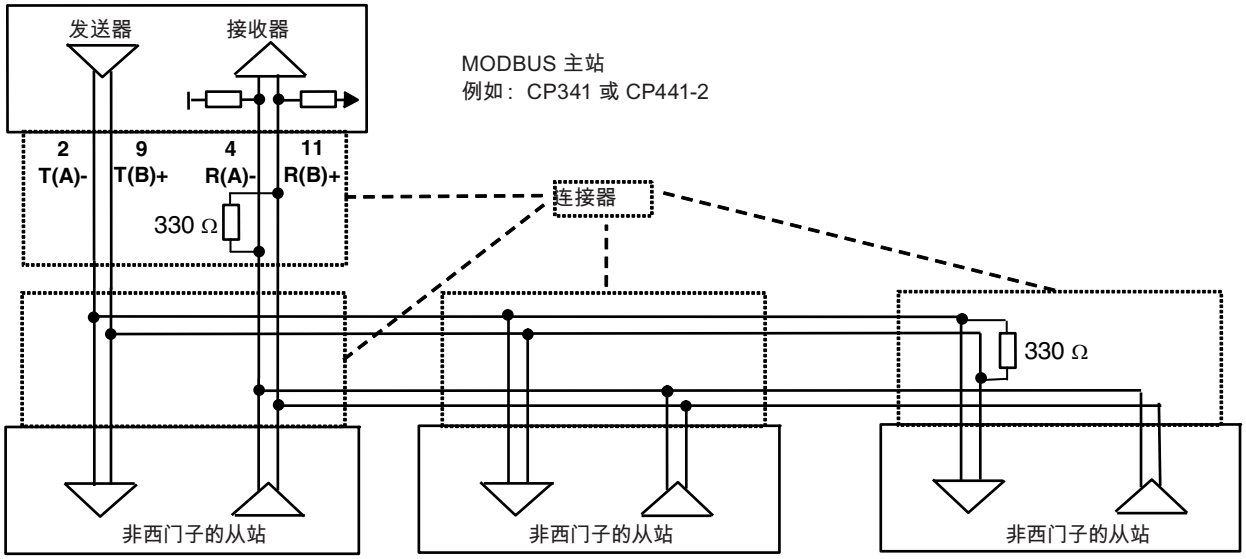
传输速率（波特）	76800			
用户数据	主站 CP 441-2	作业	从站 CP 341	确认
1 个字节	58	1	40	1
10 个字节	61	3	43	1
20 个字节	62	4	43	1
50 个字节	63	8	44	1
100 个字节	64	15	50	1
200 个字节	66	30	69	1
255 个字节	68	38	85	1

A.1 技术数据

多点接线图

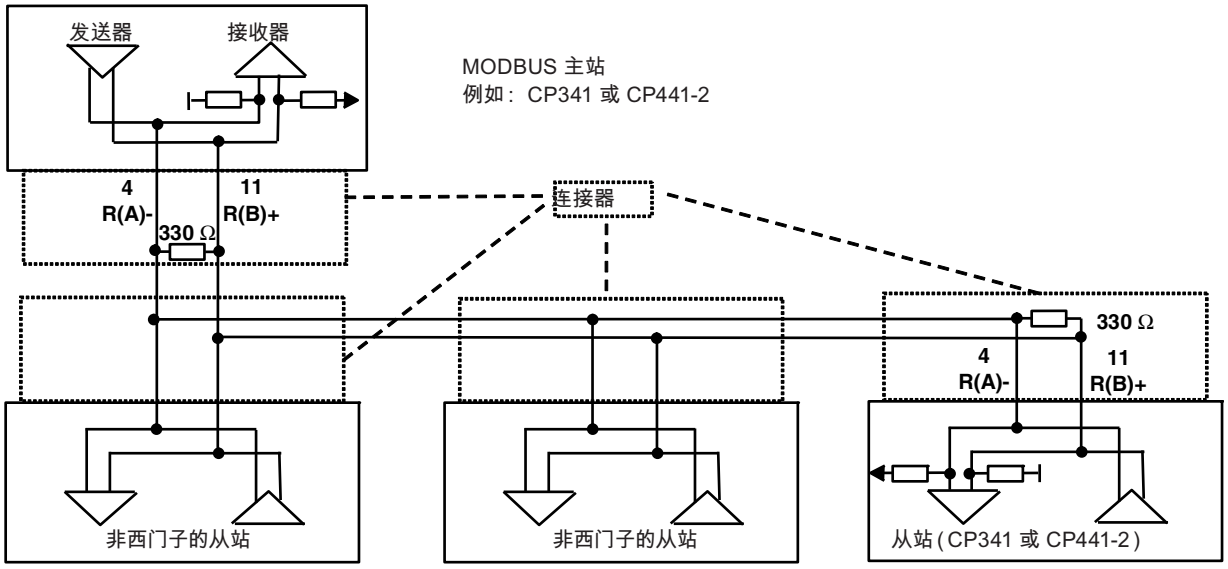
B

RS422 多点接线图（MODBUS 多点）



注意
在 **RS422** 模式下，CP 341 和 CP 441-2 **只能用作"主站"**，原因是它们无法将发送线路切换成"三态"。

RS485 多点接线图（MODBUS 多点）



以下内容对于两个模块都适用：

- 两边的 GND（CP341 / CP441-2 的第 8 脚）都必须连接。
- 无论在什么位置都必须安装外壳屏蔽。
- 节点序列的最后一个接收器的连接器要焊接一个大约为 330 Ω 的终端电阻。
- 推荐的电缆类型： LIYCY 3 x 2 x 0,14 R(A)/R(B) 和 T(A)/T(B) 类型的双绞线。
- 不允许使用“短接线”进行接线。

参考资料

C

Modbus 协议

/1/ Gould Modbus 协议
参考指导
PI-MBUS-300 Rev B
GOULD 电子设备

词汇表

CPU

S7 可编程控制器的中央处理单元具有控制和算法单元、存储器、系统程序以及连接 I/O 模块的接口。

CPU 的操作系统

CPU 操作系统用于组织 CPU 的所有功能和操作，与特定控制任务无关。

CRC

循环冗余校验＝保证能准确识别错误的校验和。

STEP 7

STEP 7 是 SIMATIC S7 的编程软件。

上传

将装载对象（例如代码块）从中央处理单元的装载存储器上传到编程设备中。

下载

将装载对象（例如代码块）从编程设备下载到中央处理单元 (CPU) 的装载存储器中。

中断

中断是一个术语，通过外部报警指示可编程控制器的处理器中的程序处理中断

主存储器

主存储器是 CPU 中的 RAM 存储器，在运行用户程序时处理器将访问主存储器。

功能块 (FB)

功能块是用户程序的组成部分，并且是符合 IEC 标准的“带存储器的块”。 功能块的存储器是已分配好的数据块，即“背景数据块”。 可以为功能块分配参数，也就是说可以带或不带参数使用。

协议

数据交换涉及的通信伙伴必须遵守用于处理和执行数据交换的固定规则。 这些规则称为协议。

参数

参数是

- STEP 7 代码块的变量，
- 该变量用于指定模块的动作，
  交货时，每个模块均具有合适的缺省设置，并且该缺省设置可以通过硬件配置改变。

有两种类型的参数： 静态参数和动态参数

参数分配

参数分配用于设置模块的行为。

参数分配工具 CP： 为点对点连接分配参数

CP：为点对点连接分配参数 工具用于为通信处理器的接口子模块分配参数，并且用于设置驱动程序特定参数。

对于每个可加载驱动程序，扩展了其标准范围。

变量

变量是内容可变的数据元素，它可以在 STEP 7 用户程序中使用。 变量由操作数（例如 M 3.1）和数据类型（例如 BOOL）组成，并且可使用符号（例如 BELT_ON）表示。

可编程控制器

可编程控制器是一种电子控制设备，它由至少一个 CPU、多种输入和输出模块以及操作员监控设备组成。

启动

从 STOP 模式转换到 RUN 模式时，将执行 RESTART 模式。 这可以由下列事件触发：

- 通过激活操作模式开关
- 上电后
- 通过编程设备中的操作员输入

可区分为冷重新启动、重新启动和热重新启动。

在线/离线

当处于在线状态时，PLC 和编程设备之间有数据连接；当处于离线状态时，它们之间将没有数据连接。

在线帮助

在使用编程软件的过程中，STEP 7 允许您在屏幕上显示上下文相关的帮助文本。

块

块是用户程序的组成元素，可根据其功能、结构或者用途进行定义。 STEP 7 有如下述块：

- 代码块 (FB、FC、OB、SFB、SFC)
- 数据块 (DB、SDB)
- 用户自定义数据类型 (UDT)。

块参数

块参数用于在多用块内占位，在调用相关块时这些参数会得到更新值。

块调用

当程序处理跳转到调用块时，发生了块调用。

工具

工具是用于组态和编程的软件工具。

循环时间

循环时间是 CPU 处理一次用户程序所需要的时间。

循环程序处理

在循环程序处理中，用户程序运行在一个程序循环中，或者不断重复的循环中。

接口子模块

CP 441-2 接口模块对信号进行物理转换。 通过改变插入式接口模块，可以使通信处理器与通信伙伴兼容。

操作数

操作数是 STEP 7 指令的组成部分，用来指示处理器要执行的操作。 操作数可以使用绝对寻址和符号寻址。

操作模式

SIMATIC S7 可编程控制器有三种不同的操作模式：

STOP、RESTART 和 RUN。 在不同的操作模式下，CPU 的功能不同。

数据块 (DB)

数据块是包含用户程序所用数据和参数的块。 与其它块不同的是，数据块不包含任何指令。 它们可细分为全局数据块和背景数据块。

数据块中包含的数据可以通过绝对地址或符号地址进行访问。 复杂数据可以通过结构化的形式存储。

数据类型

数据类型允许用户定义如何使用用户程序中变量或者常量的值。 SIMATIC S7 允许使用两种符合 IEC 1131-3 的数据类型：

- 基本数据类型
- 结构化数据类型

数据链接的组态（仅限 CP 441-2）

数据链接组态指的是系统功能块中 connection_ID 的规范。 连接 ID 使得系统功能块可以在两个通信终端之间进行通信。

机架

机架是包含模块插槽的导轨。

模块

模块是用于自动化系统的可插拔 PCB。

模块参数

模块参数是可用于设置模块功能的值。 可区分为静态模块参数和动态模块参数。

点对点连接

在点对点通信中，通信处理器建立了可编程逻辑控制器和对等通信伙伴之间的接口。

用户程序

用户程序包含用于处理信号（控制系统或过程所使用的信号）的所有指令和声明。 在 SIMATIC S7 中，将用户程序结构化，并划分为较小的单元（块）。

硬件

硬件是 PLC 的所有物理和技术设备的通称。

程序

程序指根据特定协议进行数据传输。

系统功能 (SFC)

系统功能 (SFC) 是不带存储器的功能，它集成在 S7-CPU 的操作系统中，并且需要时用户程序可以像调用功能 (FC) 一样调用它。

系统功能块 (SFB)

系统功能块 (SFB) 是带有存储器的功能块，它集成在 S7-CPU 的操作系统中，并且需要时用户程序可以像调用功能块 (FB) 一样调用它。

系统块

系统块和其它块的不同之处在于它们已经集成到 S7-300/S7-400 系统中了，并且可以作为已定义的系统功能使用。 系统块可以再细分为系统数据块、系统功能和系统功能块。

组态

组态是选择和组合各模块以形成 PLC

缺省设置

缺省设置是一个基本设置，在没有指定其它值之前它将一直使用。

背景数据块

背景数据块存储功能块的形式参数和静态数据。 背景数据块可以分配给 FB 调用或者分配给功能块的调用层级。

诊断事件

诊断事件将在 CPU 的诊断缓冲区中触发一个条目。 诊断事件可区分为以下几种：

- 模块错误
- 过程接线错误
- CPU 的系统错误
- CPU 操作模式转换
- 用户程序错误
- 用户自定义诊断事件

诊断功能

诊断功能覆盖整个系统诊断，包括 PLC 中错误的探测、解释以及报告。

诊断缓冲区

诊断缓冲区是 CPU 中带电池备份的存储区，举个例子，该缓冲区具有环形缓冲区的结构。 诊断事件按发生顺序进行存储。

软件

软件是计算机系统中使用的所有程序的总称。 这包括操作系统和用户程序。

过程映像

过程映像是可编程控制器中的一个特殊存储区。 循环程序开始时，将输入模块的信号状态传输到过程输入映像。 循环程序结束时，将过程输出映像作为信号状态传送到输出模块。

通信处理器

通信处理器是用于通信任务（例如，网络或点对点连接）的可编程模块。

索引

十一划

十二划

十三划

十六划