SIEMENS

SIMATIC

ET 200S
串行接口模块

操作说明

2005 年 8 月版

A5E01156018-04

安全技术提示

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

仅允许安装和驱动与本文件相关的附属设备或系统。设备或系统的调试和运行仅允许由**合格的专业人员**进行。本文件安全技术提示中的合格专业人员是指根据安全技术标准具有从事进行设备、系统和电路的运行，接地和标识资格的人员。

按规定使用

请注意下列说明：

⚠警告

设备仅允许用在目录和技术说明中规定的使用情况下，并且仅允许使用西门子股份有限公司推荐的或指定的其他制造商生产的设备和部件。设备的正常和安全运行必须依赖于恰当的运输，合适的存储、安放和安装以及小心的操作和维修。

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
Automation and Drives
Postfach 48 48
90327 NÜRNBERG
德国

A5E01156018-04
℗ 09/2007

目录

表格

图形

1 前言

本手册的结构

本手册是对手册*《分布式 I/O 系统 ET 200S》*的补充。

它包含可用于串行通讯的 ET-200S 模块的说明。

如何自学

在每章的开头，您都能找到**产品概述**，其中列出了所述模块的特性和应用。还能找到该模块的订货号以及所需软件的名称和版本。要获得当前的 GSD 文件，请访问：

http://support.automation.siemens.com

随后，您会在每章中相关模块名称后面发现一个标题为**『有关调试的简要说明』**的小节。这些简要说明为您提供了安装和组态模块、将模块集成到用户程序以及在用户程序中测试模块的一系列简要步骤。

索引

索引包含了出现在本手册中的关键字。

其它支持

如果您还存在与所述产品相关但本文档中并未给出解答的任何问题，请与您当地的西门子合作伙伴联系。

可通过以下网址找到您的合作伙伴：

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

可通过以下网址找到各个 SIMATIC 产品和系统的技术文档：

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

可通过以下网址找到在线目录和在线订购系统：

http://mall.automation.siemens.com/

培训中心

我们提供了相应课程，以帮助您熟悉 SIMATIC S7 自动化系统。请联系您当地的培训中心或位于德国纽伦堡 D 90327 的培训中心总部。

电话：+49 (911) 895-3200

网址：http://www.sitrain.com

技术支持

可通过以下方式与所有 A&D 产品的技术支持部门联系

- 通过 Web 表单提出支持请求

  http://www.siemens.de/automation/support-request

- 电话：+ 49 180 5050 222
- 传真：+ 49 180 5050 223

可通过以下网址找到有关技术支持的更多信息：
http://www.siemens.com/automation/service

Internet 上的服务与支持

除在线文档外，我们还在 Internet 上提供了全面的在线知识库。

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

您可以在该网站上找到：

- 不断为您提供产品最新信息的新闻快递。
- 通过服务和支持中的搜索 (Search) 功能找到的所需文档。
- 论坛，世界各地的用户和专家可在此交流他们的经验。
- 您当地的自动化与驱动集团的合作伙伴。
- 有关现场服务、维修和备件的信息。在“服务”(Services) 下也可获取更多信息。

串行接口模块

2

2.1 产品概述

订货号

6ES7 138-4DF01-0AB0

产品说明

ET 200S 1SI 串行接口模块是 ET 200S 产品范围内的插入式模块。它通过三个硬件接口（RS 232C、RS 422 和 RS 485）和两个软件协议（ASCII 和 3964[R]）提供对串行通讯的访问。

可以使用 ET 200S 1SI 接口模块在自动化系统或计算机之间通过点对点连接交换数据。所有通讯均基于串行异步传输。

当在 STEP 7 硬件组态或其它某组态应用程序中参数化模块时，可以选择此通讯模式。硬件目录中会出现六个版本的模块：

- ASCII (4B)
- ASCII (8B)
- ASCII (32B)
- 3964(R) (4B)
- 3964(R) (8B)
- 3964(R) (32B)

8 字节或 32 字节的数据传输可增大吞吐率，但是需要的 ET 200S 机架上的 I/O 存储器较多。相反，4 字节的数据传输需要的 ET 200S 机架上的 I/O 存储器较少，但是提供的吞吐率较低。您选择的模块类型取决于您的应用要求。

ET 200S 1SI 串行接口模块的功能

ET 200S 1SI 串行接口模块具有以下功能：

- 符合 RS 232C、RS 422 或 RS 485 的集成接口
- 传输率最高 115.2 Kbaud，半双工
- 在模块固件中集成以下传输协议：
  - 3964(R) 程序
  - ASCII 驱动程序

驱动程序的功能取决于模块的参数化方式。

下表列出了各个驱动程序接口的功能。

表格 2-1　　ET 200S 1SI 串行接口模块驱动程序的功能

功能	RS 232C	RS 422	RS 485
ASCII 驱动程序	**是**	**是**	**是**
使用 RS 232C 伴随信号	是	否	否
使用 FB 控制/读取 RS 232C 伴随信号	是	否	否
RTS/CTS 流量控制	是	否	否
XON/XOFF 流量控制	是	是	否
3964(R) 程序	**是**	**是**	**否**

通讯

ET 200S 1SI 串行接口模块允许与不同的 Siemens 模块和非 Siemens 产品进行点对点通讯，包括以下模块或产品：

- 通过 3964(R) 驱动程序的 SIMATIC S5，相应的接口模块位于 S5 端
- 通过 3964(R) 驱动程序的 ES 2 系列 Siemens PDA 端子
- 通过 3964(R) 驱动程序的 MOBY I（ASM 420/421、SIM）、MOBY L (ASM 520) 和 ES 030K 数据采集端子
- 通过 ASCII 驱动程序 (ET 200S SI RS 422/485) 的 SIMOVERT 和 SIMOREG（USS 协议），使用 STEP 7 程序进行适当的协议自适应
- 通过 3964(R) 程序的 PC（以下开发工具用于在 PC 上编程：适用于 MS–DOS 的 PRODAVE DOS 64R [6ES5 897–2UD11]、适用于 Windows 或 ASCII 驱动程序的 PRODAVE WIN 64R [6ES5 897-2VD01]）
- 通过 3964(R) 或 ASCII 驱动程序的条形码阅读器
- 通过 3964(R) 或 ASCII 驱动程序的非 Siemens PLC
- 通过能够对 ASCII 驱动程序进行适当协议自适应的具有简单协议结构的其它设备
- 也具有 3964(R) 驱动程序的其它设备

LED 显示

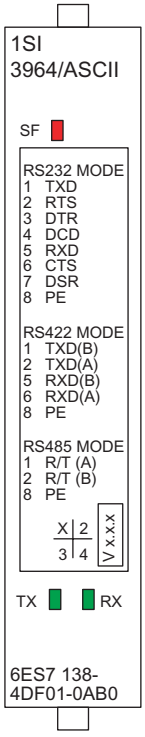
以下状态 LED 位于 ET 200S 1SI 接口模块的前面板上：

- SF （红色） 出错 LED
- TX （绿色） 接口发送
- RX （绿色） 接口接收

在『诊断（页码 101）』一节中描述了由这些 LED 指示的操作模式和错误。

前面板

下图显示了 ET 200S 1SI 串行接口模块前面板上的标签。



图片 2-1 ET 200S 1SI 串行接口模块前面板上的标签

2.2 有关调试串行接口模块的简要说明

简介

根据在串行接口模块间发送和接收数据的示例，这些简要说明介绍了如何设置一个可以正常运行的应用、串行接口模块（硬件和软件）基本操作的工作原理，以及如何测试硬件和软件。

在本示例中，我们将在 RS 232C ASCII 模式下运行两个 ET 200S 串行接口模块。

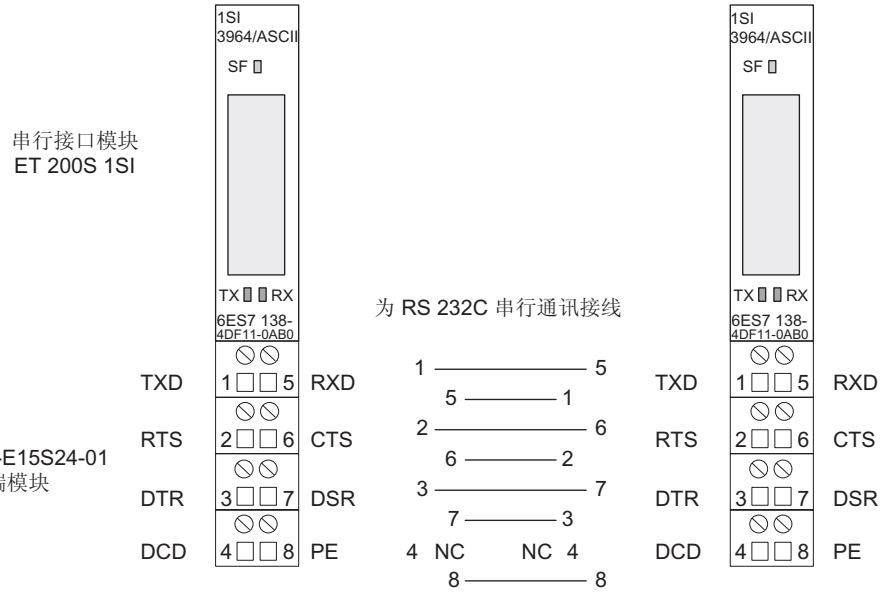
要求

必须满足以下要求：

- 必须通过 DP 主站在 S7 站上调试 ET 200S 站。
- 您将需要以下组件：
  - 两个 TM-E15S24-01 终端模块
  - 两个 ET 200S 串行接口模块
  - 必需的接线材料

安装、接线和装配

安装两个 TM-E15S24-01 终端模块并对其进行接线（请参见下图）。将两个 ET 200S 串行接口模块连接至这些终端模块。（手册 *《ET 200S 分布式 I/O 设备》* 对此进行了详细介绍）。



图片 2-2　　该示例的端子分配

所用的组态

下表显示了该实例程序所用的组态。

表格 2-2　　该应用实例的参数化

参数	值
组诊断	取消激活
接口	RS 232C
接收线路的初始状态	
数据流量控制（初始状态）	无
波特率	9600
数据位	8
停止位	1
奇偶校验	偶数
接收结束检测	字符延迟时间结束
字符延迟时间 (ms)	4
结束字符 1	
结束字符 2	
接收的字符数	
动态消息帧缓冲区	是
防止消息帧缓冲区被覆盖	是
启动时清空接收缓冲区	是

所用的块

下表显示了该实例程序所用的块。

块	符号	注释
OB1	CYCLE	循环程序处理
OB100	RESTART	启动处理重新启动
DB21	SEND_IDB_SI_0	S_SEND_SI FB 的背景数据块
DB22	RECV_IDB_SI_1	S_RECV_SI FB 的背景数据块
DB40	SEND_WORK_DB_SI_0	标准 FB3 的工作 DB
DB41	RECV_WORK_DB_SI_1	标准 FB2 的工作 DB
DB42	SEND_SRC_DB_SI_0	发送的数据块
DB43	RECV_DST_DB_SI_1	接收数据块
FB2	S_RECV_SI	数据的接收标准 FB
FB3	S_SEND_SI	数据的发送标准 FB
FC21	SEND_SI_0	发送数据
FC22	RECV_SI_1	接收数据

供货和安装类型

ET 200S 1SI 模块和功能块的实例程序可通过以下网址获得：

http://support.automation.siemens.com/WW/view/en/10805265/133100

根据安装，您可以在 zXX21_10_1SI_ASCII 项目中找到实例程序。

通过选择“文件”(File) > “打开”(Open) > “实例项目”(Sample projects) 在 STEP 7 SIMATIC 管理器中打开此项目。

该实例程序既可以作为已编译的程序使用，也可以作为 ASCII 源文件使用。还有一个符号表，其中包含了该示例中所用的符号。

如果没有第二个 ET 200S 1SI 模块可作为通讯伙伴，则您必须在 HW Config 中通过选择“编辑”(Edit) > “删除”(Delete) 删除第二个 ET 200S 1SI。另外，必须在 OB1 中暂停 FC22 调用（FC 用于接收）。

下载至 CPU

已完成示例的硬件设置并已连接编程设备。

CPU 存储器复位后（STOP 操作模式），将整个示例传输至用户存储器。将模式选择器从 STOP 切换至 RUN。

错误行为

如果在启动期间发生错误，则不会执行循环处理的块调用命令，并将设置错误 LED。

如果出现错误消息，则设置块的 ERROR 参数输出。然后，有关错误的更多详细说明将存储在块的 STATUS 参数中。如果 STATUS 参数包含 16#1E0E 或 16#1E0F 错误消息，则更多详细说明将存储在背景数据块的 SFCERR 变量中。

激活、启动程序

启动程序位于 OB100 中。

控制位和计数器在启动期间复位。

循环程序

循环程序位于 OB1 中。

在此示例中，功能块 FB2 S_RECV_SI 和 FB3 S_SEND_SI 与功能 FC21 和 FC22 一起工作，与数据块 DB21 和 DB22 一起用作背景数据块，与 DB42 和 DB43 一起用作传输和接收 DB。

在该示例中，功能块部分通过常量进行参数化，部分通过符号寻址的实际地址进行参数化。

说明

数据在插槽 2 上的 ET 200S 1SI 和插槽 3 上的 ET 200S 1SI 之间传输。如果您使用不同的通讯伙伴，则不应用 FC 22 调用 (RECEIVE)。

对 FC21 (SEND) 的说明

程序部分“生成沿 S_SEND_SI_REQ”：

S_SEND_SI 最初在 S_SEND_SI_REQ=0 时执行一次。然后，S_SEND_SI_REQ 设置为 1。如果在 S_SEND_SI_REQ 控制参数处检测到信号状态从 0 变为 1，则会启动 S_SEND_SI 作业。

如果 S_SEND_SI_DONE=1 或 S_SEND_SI_ERROR=1，则 S_SEND_SI_REQ 复位为 0。

程序部分“S_SEND_SI_DONE=1”：

如果传输成功，则 S_SEND_SI_DONE 参数在 S_SEND_SI 的参数输出处设置为 1。

为了区分连续传输，源数据块 DB42 的数据字 0 中含有一个发送计数器 (S_SEND_SI_COUNTER_OK)。

程序部分“S_SEND_SI_ERROR=1”：

如果 S_SEND_SI_ERROR=1 时执行 S_SEND_SI，则错误计数器 S_SEND_SI_COUNTER_ERR 的数据字 2 将增加。另外，将复制 S_SEND_SI_WORK_STAT，因为它将在下一个周期中被 0 覆盖，从而使其无法读出。

对 FC22 (RECEIVE) 的说明

程序部分“启用接收数据”：

为了接收数据，S_RECV_SI 块上的 S_RECV_SI_EN_R 接收启用器必须设置为 1。

程序部分“S_RECV_SI_NDR=1”：

如果设置了 S_RECV_SI_NDR，便意味着已收到新数据，从而接收计数器 S_RECV_SI_WORK_CNT_OK 将增加。

程序部分“S_RECV_SI_ERROR=1”：

如果出现错误，即已在 S_RECV_SI 的参数输出处设置了错误位，则 S_RECV_SI_WORK_CNT_ERR 错误计数器将增加。另外，将复制 S_RECV_SI_WORK_STAT，因为它将在下一个周期中被 0 覆盖，从而使其无法读出。

所有相关值均可以在 VAT 中监视以进行测试。

2.3 具有端子分配的电路图

接线规则

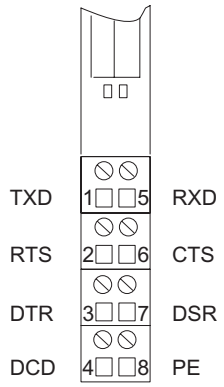
必须屏蔽电缆（端子 1 到 8）。两端必须都连接屏蔽。可以使用屏蔽触点实现此目的（请参见手册 *《ET 200S 1SI 分布式 I/O 设备》*）。

端子分配

RS 232C 通讯的端子分配

下表显示了设置 RS 232C 通讯协议后 ET 200S 1SI 串行接口模块的端子分配。

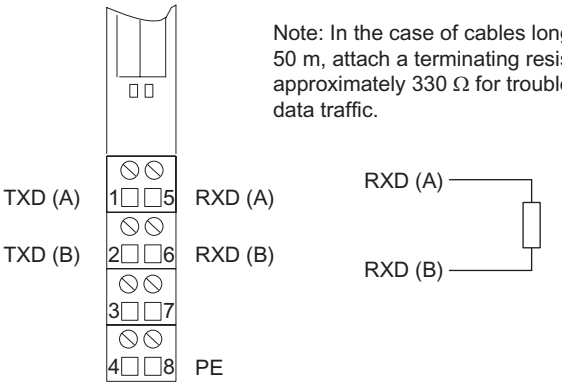
表格 2-3　用于 RS 232C 通讯的 ET 200S 1SI 串行接口模块的端子分配

视图	备注		
	模式：半双工和全双工		
	端子		
	1	TXD	发送的数据
	5	RXD	接收的数据
	2	RTS	请求发送
	6	CTS	明确发送
	3	DTR	数据终端准备就绪
	7	DSR	数据集准备就绪
	4	DCD	检测到的数据载体
	8	PE	接地

RS 422 通讯的端子分配

下表显示了设置 RS 422 通讯协议后 ET 200S 1SI 串行接口模块的端子分配。

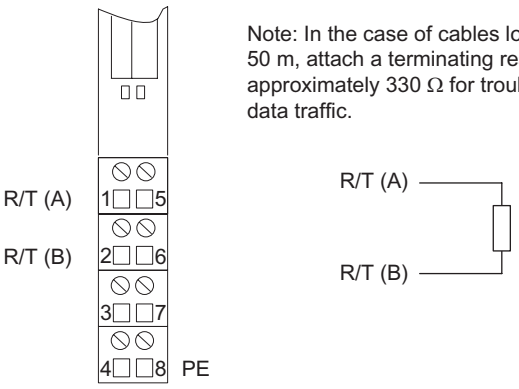
表格 2-4　　用于 RS 422 通讯的 ET 200S 1SI 串行接口模块的端子分配

视图	端子分配	备注		
		模式：全双工 端子		
		1	TXD	(A)-
		5	RXD	(A)-
		2	TXD	(B)+
		6	RXD	(B)+
		8	PE	接地

RS 485 通讯的端子分配

下表显示了设置 RS 485 通讯协议后 ET 200S 1SI 串行接口模块的端子分配。

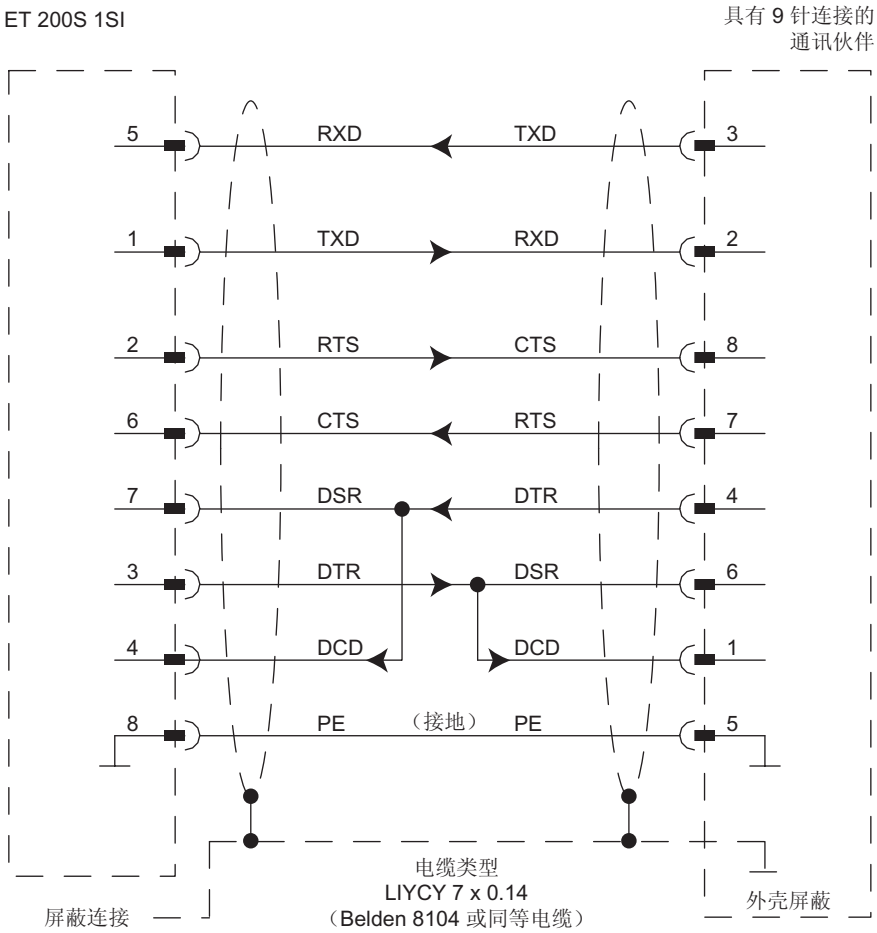
表格 2-5　　用于 RS 485 通讯的 ET 200S 1SI 串行接口模块的端子分配

视图	端子分配	备注		
		模式：半双工 端子		
		1	R/T	(A)-
		2	R/T	(B)+
		8	PE	接地

用于 9 针电缆连接器的 RS 232C 连接电缆的端子分配

下图显示了 ET 200S 1SI 串行接口模块和具有 9 针 D 型连接插槽的通讯伙伴之间的 RS 232C 点对点通讯的电缆连接。

- 在 ET 200S 1SI 端，信号线连接至相应编号的端子。
- 使用通讯伙伴上的 9 针 D 型连接插槽。

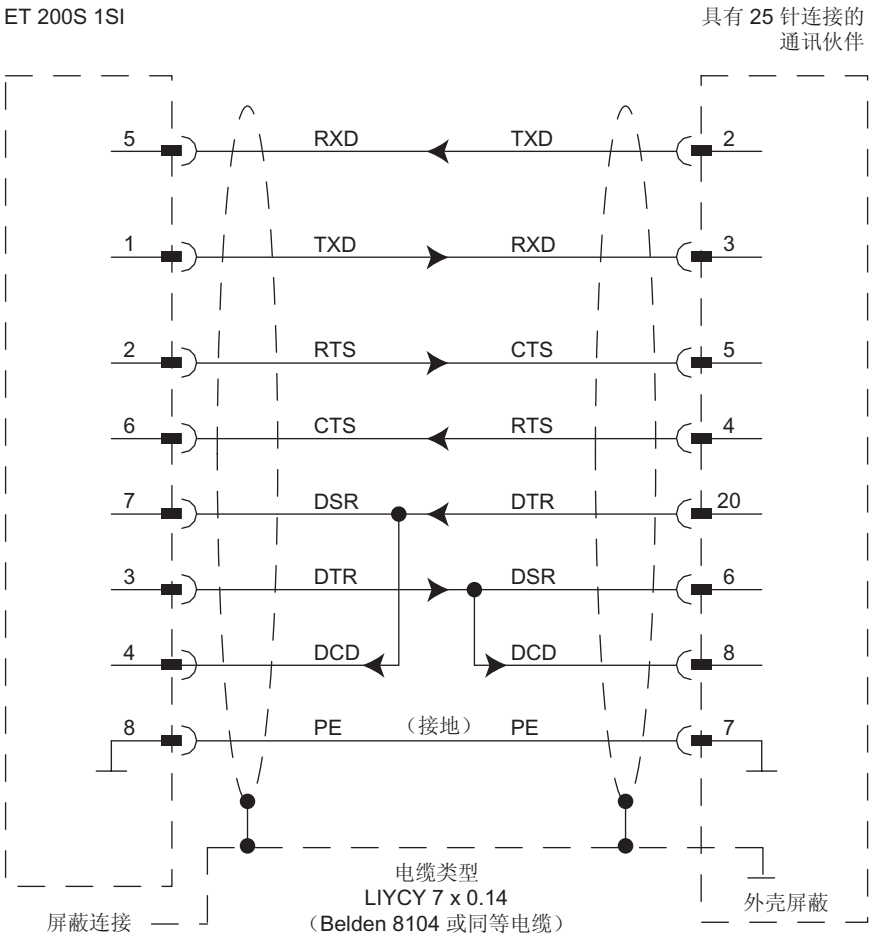


图片 2-3　　用于 9 针电缆连接器的 RS 232C 连接电缆的端子分配

用于 25 针电缆连接器的 RS 232C 连接电缆的端子分配

下图显示了 ET 200S 1SI 串行接口模块和具有 25 针 D 型电缆连接器的通讯伙伴之间的 RS 232C 点对点通讯的电缆连接。

- 在 ET 200S 1SI 端，信号线连接至相应编号的端子。
- 使用通讯伙伴上的 25 针 D 型连接器。

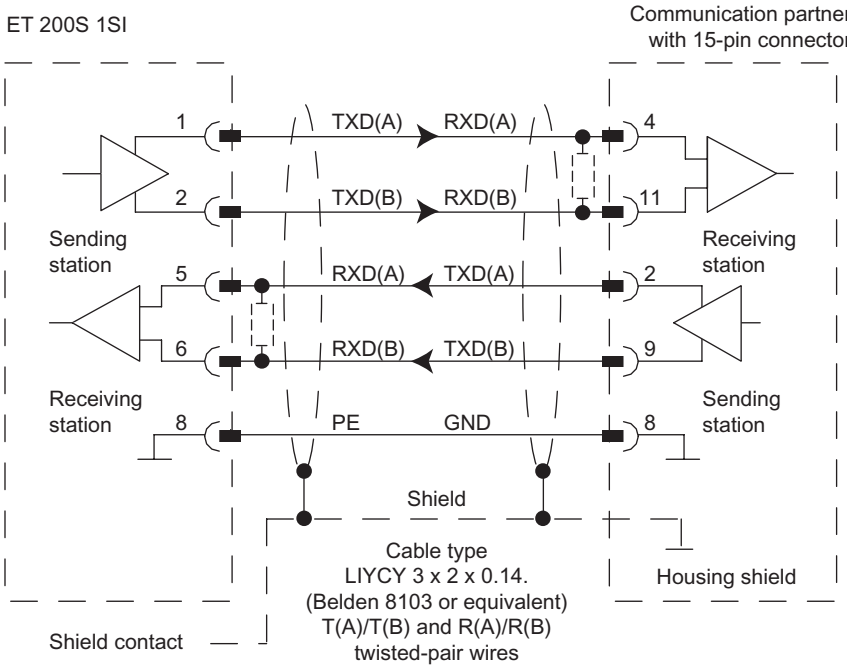


图片 2-4　　用于 25 针电缆连接器的 RS 232C 连接电缆的端子分配

用于 15 针电缆连接器的 RS 422 连接电缆的端子分配

下图显示了 ET 200S 1SI 串行接口模块和具有 15 针 D 型电缆连接器的通讯伙伴之间的 RS 422 点对点通讯的电缆连接。

- 在 ET 200S 1SI 端，信号线连接至相应编号的端子。
- 使用通讯伙伴上的 15 针 D 型连接器。



图片 2-5　　用于 15 针电缆连接器的 RS 422 连接电缆的端子分配

说明

如果电缆长度超过 50 m，则附加一个约 330 Ω（请参见上图）的终端电阻以确保数据通讯畅通。

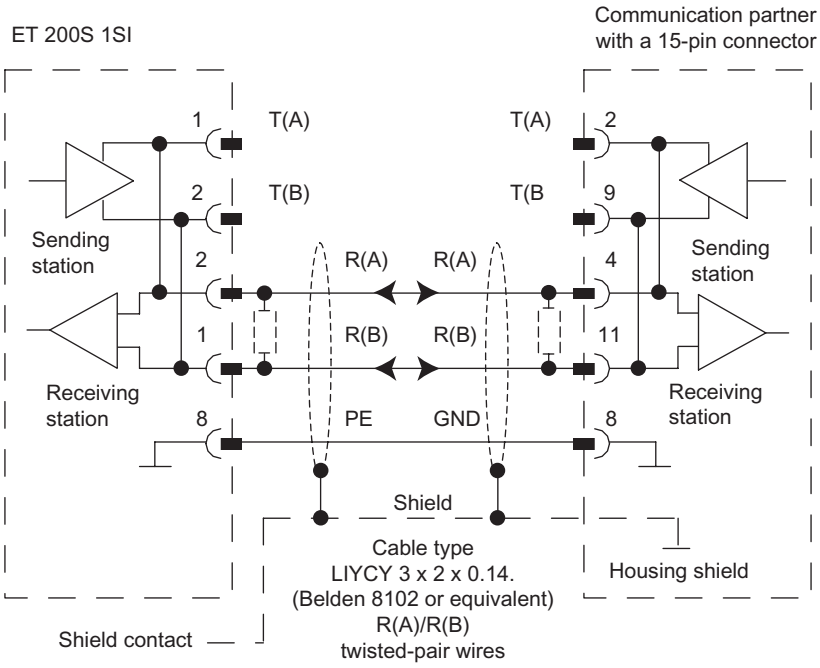
使用在这种情况下所用的电缆类型，作为通讯伙伴的 ET 200S 1SI 模块支持以下长度：

- 19,200 波特时，最长为 1200 m
- 38,400 波特时，最长为 500 m
- 76,800 波特时，最长为 250 m

用于 15 针电缆连接器的 RS 485 连接电缆的端子分配

下图显示了 ET 200S 1SI 串行接口模块和具有 15 针 D 型电缆连接器的通讯伙伴之间的 RS 485 点对点通讯的电缆连接。

- 在 ET 200S 1SI 端，信号线连接至相应编号的端子。
- 使用通讯伙伴上的 15 针 D 型连接器。



图片 2-6　　用于 15 针电缆连接器的 RS 485 连接电缆的端子分配

说明

如果电缆长度超过 50 m，则附加一个约 330 Ω（请参见上图）的终端电阻以确保数据通讯畅通。

使用在这种情况下所用的电缆类型，作为通讯伙伴的 ET 200S 1SI 模块支持以下长度：

- 19,200 波特时，最长为 1200 m
- 38,400 波特时，最长为 500 m
- 76,800 波特时，最长为 250 m
- 115,200 波特时，最长为 200 m

2.4 RS 232C 接口

定义

RS 232C 接口是一种符合 RS 232C 标准的、用于串行数据传输的电压接口。

特性

RS 232C 接口具有以下特性：

类型：	电压接口
前连接器：	8 针标准 ET 200S 端子连接器
RS 232C 信号：	TXD、RXD、RTS、CTS、DTR、DSR、DCD、GND
传输率：	最高 115.2 Kbaud（3964[R] 程序） 最高 115.2 Kbaud（ASCII 驱动程序）
电缆长度：	最长 15 m，电缆类型 LIYCY 7 x 0.14
相关标准：	DIN 66020、DIN 66259、EIA RS 232C、CCITT V.24/V.28
防护等级：	IP20

RS 232C 信号

下表介绍了 RS 232C 信号：

信号	说明	含义
TXD	发送的数据	在空闲状态下，传输线路保持在逻辑“1”处。
RXD	接收的数据	接收线路必须通过通讯伙伴保持在逻辑“1”处。
RTS	请求发送	ON：ET 200S 1SI 明确发送。 OFF：ET 200S 1SI 不处于发送模式下。
CTS	明确发送	通讯伙伴可以从 ET 200S 接收数据。接口模块希望这是对 RTS = ON 的响应。
DTR	数据终端准备就绪	ON：ET 200S SI 接通电源并已做好运行准备。 OFF：ET 200S SI 未接通电源并且未做好运行准备。
DSR	数据集准备就绪	ON：通讯伙伴接通电源并已做好运行准备。 OFF：通讯伙伴未接通电源并且未做好运行准备。
DCD	数据载体检测	连接调制解调器时的载波信号。

2.5 RS 422/485 接口

定义

RS 422/485 接口是一种符合 RS 422/485 标准的、用于串行数据传输的差分电压接口。

特性

RS 422/485 接口具有以下特性：

类型：	差分电压接口
前连接器：	8 针标准 ET 200S 端子连接器
RS 422 信号：	TXD (A)-、RXD (A)-、TXD (B)+、RXD (B)+、GND
RS 485 信号：	R/T (A)-、R/T (B)+、GND
传输率：	最高 115.2 Kbaud（3964[R] 程序） 最高 115.2 Kbaud（ASCII 驱动程序）
电缆长度：	最长 1,200 m，电缆类型 LIYCY 7 x 0.14
相关标准：	EIA RS 422/485、CCITT V.11/V.27
防护等级：	IP20

2.6 串行数据传输的基本原理

2.6.1 串行数据传输

点对点连接

系统提供了多种在两个或更多通讯伙伴之间进行数据交换的联网选项。最简单的数据交换形式是在两个通讯伙伴之间进行点对点连接。

在点对点通讯中，串行接口模块形成了可编程控制器与通讯伙伴之间的接口。在与 ET 200S 1SI 串行接口模块的点对点通讯中，数据通过串行接口传输。

串行数据传输

在串行数据传输中，以固定顺序依次发送信息的每个字节的各个位。

ET 200S 1SI 串行接口模块通过其串行接口自主处理与通讯伙伴的数据传输。这就是为什么模块具有两个不同的、用于双向数据通讯的驱动程序的原因。

- ASCII 驱动程序
- 3964(R) 程序

双向数据通讯 — 操作模式

ET 200S 1SI 具有两种进行双向数据通讯的操作模式：

- 半双工操作（3964[R] 程序、ASCII 驱动程序）

  通讯伙伴在两个方向上轮流交换数据。使用半全工操作意味着在给定的任意时刻要么正在发送数据，要么正在接收数据。而数据流量控制的各个控制字符（例如 XON/XOFF）可能是个例外，这些字符可以在接收操作过程中发送，也可以在发送操作过程中接收。

- 全双工操作（ASCII 驱动程序）

  通讯伙伴在两个方向上同时交换数据。双全工操作意味着可以同时发送和接收数据。每个通讯伙伴都必须能够同时处理发送和接收操作。

下表列出了使用 ASCII 驱动程序进行数据通讯时的操作模式。

表格 2-6　用于 ET 200S 1SI 串行接口模块的数据通讯操作模式

数据通讯	RS 232C	RS 422	RS 485
半双工	是	是	是
全双工	是	是	不可以

声明

在进行串行数据传输前，必须在两个通讯伙伴之间发表声明。包括以下声明：

- 传输率（波特率）
- 字符延迟时间和确认时间
- 奇偶校验
- 数据位个数
- 停止位个数
- 尝试建立连接和传输的次数

『使用 3964(R) 程序的数据传输（页码 38）』和『有关使用 ASCII 驱动程序进行数据传输的基本信息（页码 46）』两节中介绍了这些声明在各种传输程序中的作用及其参数化方式。

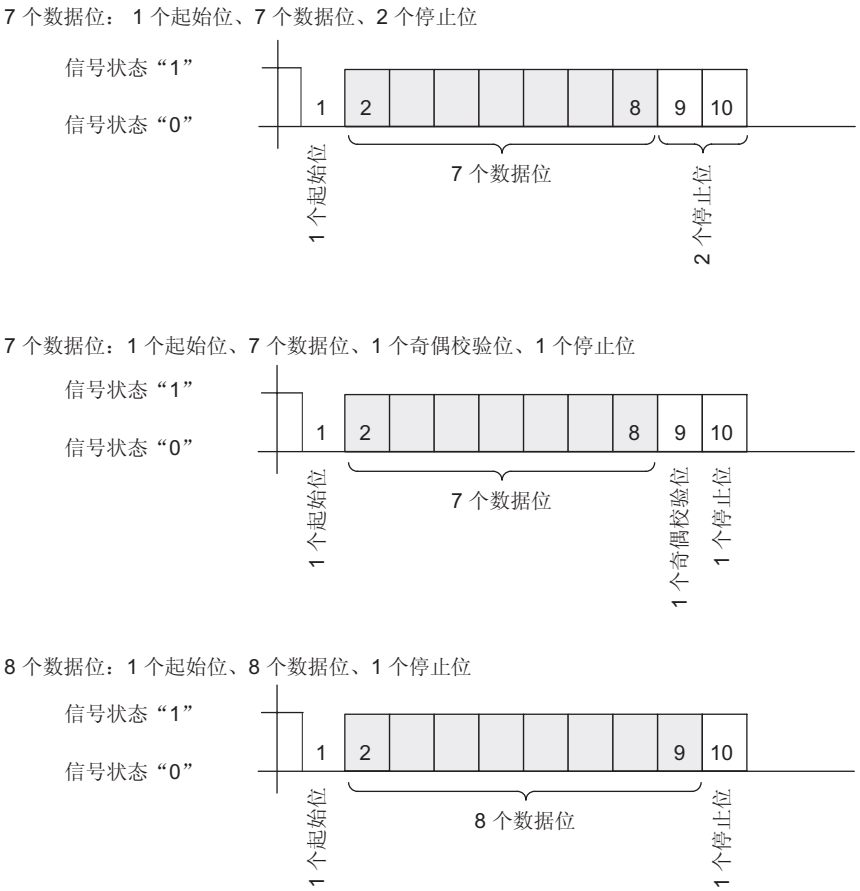
2.6.2 字符帧

原理

数据通过串行接口在 ET 200S 1SI 串行接口模块和通讯伙伴之间以 10 位或 11 位字符帧传输。每个字符帧可以使用三种数据格式。可以在 STEP 7 中参数化所需的格式。

10 位字符帧

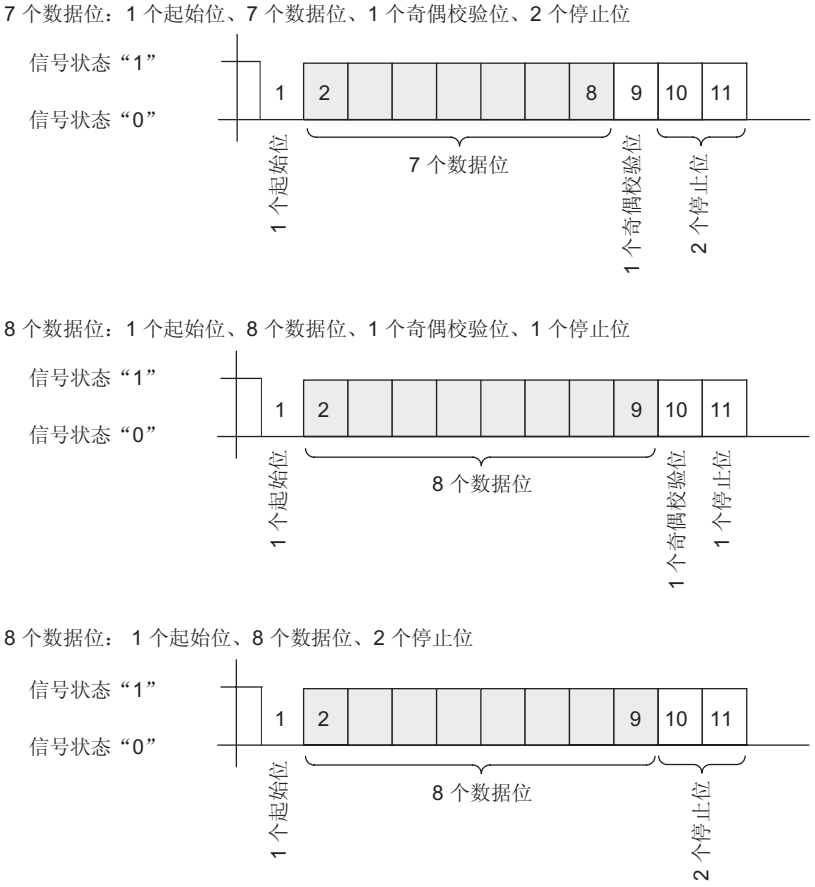
下图显示了 10 位字符帧的三种数据格式。



图片 2-7 10 位字符帧

11 位字符帧

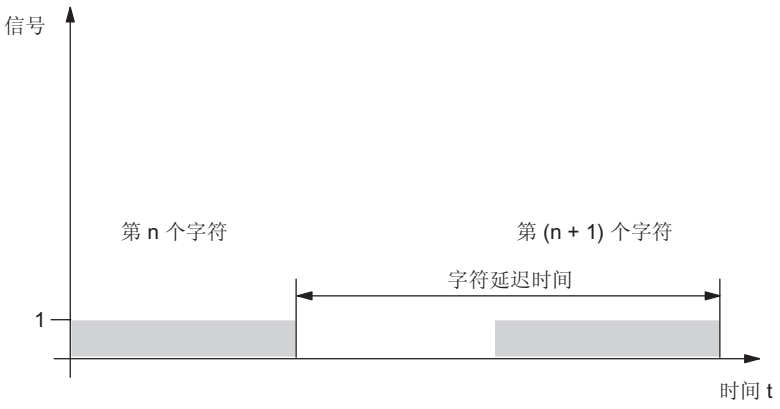
下图显示了 11 位字符帧的三种数据格式。



图片 2-8　　11 位字符帧

字符延迟时间

下图显示了消息帧中两个字符之间的最大接收时间间隔。这就是字符延迟时间。



图片 2-9　　字符延迟时间

2.6.3 ISO 7 层参考模型

简介

传输数据时，涉及的所有通讯伙伴都必须遵循处理和执行数据通讯的固定规则。ISO 已定义了一个 7 层模型，该模型被公认为用于计算机之间通讯的国际标准化传输协议的基础。

协议

传输数据时，涉及的所有通讯伙伴都必须遵循处理和执行数据通讯的固定规则。这些规则称为协议。

协议定义了以下内容：

- **模式**

  半双工模式或全双工模式

- **启动**

  有关哪些通讯伙伴可以启动数据传输以及在什么条件下启动的协议

- **控制字符**

  哪些控制字符将用于数据传输？

- **字符帧**

  哪些字符帧将用于数据传输？

- **数据备份**

  将使用的数据备份程序

- **字符延迟时间**

  必须在其中接收引入字符的时间周期

- **传输速度**

  以位/秒 (bps) 为单位定义

ISO 7 层参考模型

该参考模型定义了通讯伙伴的外部特性。每个协议层（最低层除外）都嵌入相邻的下一层中。各层如下：

1. **物理层**
   - 通讯的物理条件，例如传输介质和传输率
2. **数据链路层**
   - 传输的安全程序
   - 访问模式
3. **网络层**
   - 网络连接
   - 在两个伙伴间的通讯进行寻址
4. **传输层**
   - 错误检测程序
   - 调试
   - 握手
5. **会话层**
   - 数据传输的建立和终止
   - 通讯控制
6. **表示层**
   - 将通讯系统的数据表达式的标准形式转换为设备特定的形式（数据解释规则）
7. **应用层**
   - 定义通讯任务及其所需的功能

处理协议

发送通讯伙伴从协议的最高层（第 7 层 — 应用层）向最低层（第 1 层 — 物理层）处理，而接收通讯伙伴以相反的顺序处理协议，即从第 1 层开始。

并非所有协议都必须考虑全部 7 个层。如果发送伙伴和接收伙伴使用同一个协议，则可忽略第 6 层。

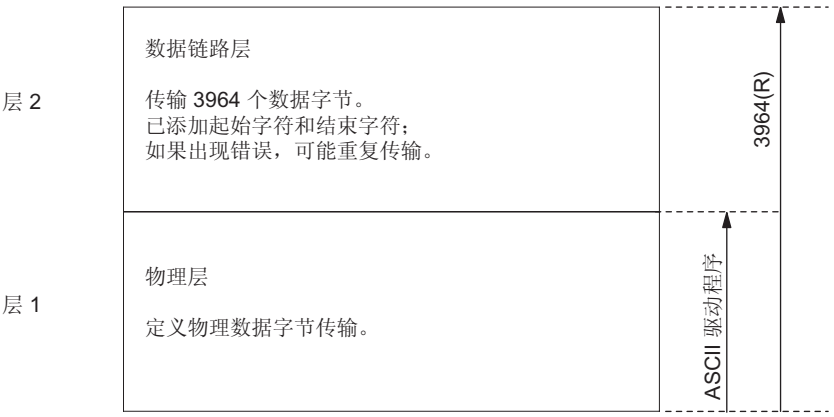
2.6.4 传输完整性

原理

传输完整性在数据传输和传输程序的选择上起着重要作用。一般来说，处理的参考模型层数越多，传输完整性越高。

支持的协议

下图显示了 ET 200S 1SI 接口模块支持的 ASCII 和 3964(R) 协议如何适应 ISO 参考模型。



图片 2-10　支持的协议如何适应参考模型

使用 ASCII 驱动程序时的传输完整性

使用 ASCII 驱动程序时，请遵循以下准则来增强数据完整性：

- 除了使用奇偶校验位（根据字符帧设置，也可以取消选择该位）外，其它方法都不能确保使用 ASCII 驱动程序进行数据传输时的数据完整性。这意味着就数据吞吐量而言，使用 ASCII 驱动程序进行数据传输效率很高，但不能确保完好的数据完整性。
- 使用奇偶校验位可以检测到要传输的字符中的反转位。如果字符中有两个或更多位被反转，则不再能检测到该错误。
- 要增强传输完整性，可使用消息帧的校验和以及长度规范。这些测量必须由用户执行。
- 通过对发送或接收消息帧进行响应的确认消息帧可以进一步增强数据完整性。这也适用于使用高层协议进行数据传输的情况（请参见 ISO 7 层参考模型）。

使用 3964(R) 时的传输完整性

3964(R) 程序可提供增强的数据完整性：

- 使用 3964(R) 时的汉明间距为 3。这可测量数据传输的完整性。
- 3964(R) 程序可确保传输线路上的高传输完整性。可通过设置和清除固定的消息帧，并使用块校验字符 (BCC, block check character) 来实现此高完整性。

可以使用两种不同的程序进行数据传输，带有或不带有块校验字符：

- 不带有块校验字符的数据传输：**3964**
- 带有块校验字符的数据传输：**3964(R)**

在本手册中，标识 **3964(R)** 用于指出说明和注释是针对两种数据传输程序的。

使用 3964(R) 时的性能限制

- 不能保证会通过通讯伙伴中的 PLC 程序进一步处理发送/接收数据。只能通过使用可编程的确认机制确保这一点。
- 3964(R) 程序的块校验（EXOR 逻辑运算）检测不到丢失的零（作为一个完整字符），因为在 EXOR 运算中零不影响计算结果。

  尽管丢失一个完整字符（该字符一定为零）的可能性很小，但如果传输条件极差，这种情况也有可能发生。

  将数据消息的长度随数据本身一起发送并在另一端检查该长度，可以保护传输避免此类错误。

2.7 使用 3964(R) 程序的数据传输

2.7.1 使用 3964(R) 程序的数据传输

原理

3964(R) 程序通过 ET 200S 模块和通讯伙伴之间的点对点连接控制数据传输。除了物理层（第 1 层），3964(R) 程序还合并了数据链路层（第 2 层）。

控制字符

数据传输期间，3964(R) 程序将控制字符添加到用户数据（数据链路层）。这些控制字符使通讯伙伴可以检查数据是否已全部无错到达。

3964(R) 程序评估下列控制字符：

- **STX**：正文开始；

  要传输的字符串的起点

- **DLE**：数据链路转义字符

  数据链路转义字符

- **ETX**：正文结束；

  要传输的字符串的终点

- **BCC**：块校验字符（仅对于 3964[R]）

  块校验字符

- **NAK**：否定确认

  否定确认

说明

如果 DLE 作为信息字符发送，则在传输线路上发送两次（DLE 副本），以将其与在建立和清除连接的上下文中使用的 DLE 控制字符区分开来。然后接收器将恢复 DLE 副本。

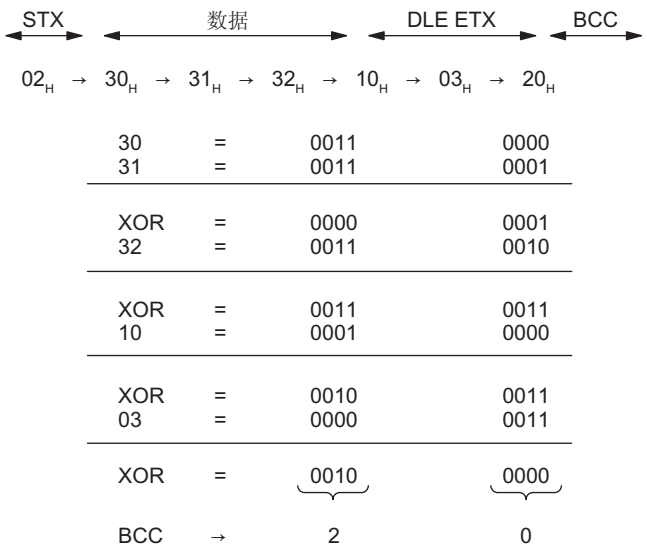
优先级

使用 3964(R) 程序时，必须为一个通讯伙伴指定较高的优先级，为另一个伙伴指定较低的优先级。如果两个伙伴同时请求发送作业，则优先级较低的伙伴将延迟其发送作业。

块校验字符

使用 3964(R) 传输协议时，通过发送附加块校验字符 (BCC, block check character) 来增强数据完整性（请参见下图）。

消息帧：



图片 2-11　　块校验字符

块校验字符相当于发送或接收块的纵向偶校验（所有数据字节的 EXOR 逻辑运算）。它从建立连接后的第一个用户数据字节（消息帧的第一个字节）开始，在清除连接时的 DLE ETX 后结束。

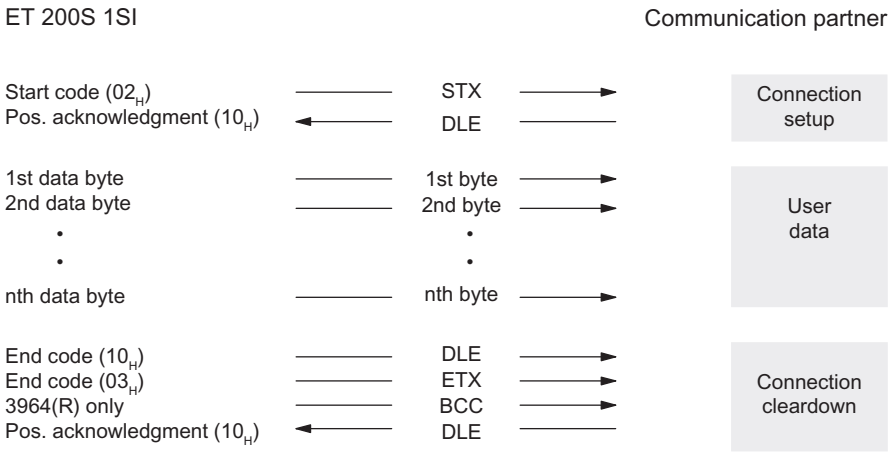
说明

在使用 DLE 副本的情况下，会在生成 BCC 时将 DLE 字符计算两次。

2.7.2 使用 3964(R) 程序发送数据

使用 3964(R) 发送数据

下图说明了使用 3964(R) 程序发送数据时的传输顺序。



图片 2-12 使用 3964(R) 程序发送时的数据通讯

建立发送连接

若要建立连接，3964(R) 程序将发送控制代码 STX。如果通讯伙伴在确认延迟时间结束前以 DLE 代码进行响应，则程序将切换至发送模式。

如果通讯伙伴以 NAK 或任何其它字符（DLE 除外）进行应答，或在确认延迟时间结束之前无响应，则程序将重复建立连接。尝试建立连接失败的次数达到定义的次数后，程序将取消连接建立，并将 NAK 字符发送给通讯伙伴。系统程序将向 P_SEND 功能块（STATUS 输出参数）报告错误。

发送数据

如果已成功建立连接，则使用选定的传输参数将 ET 200S 模块输出缓冲区中包含的用户数据发送给通讯伙伴。该伙伴将监视引入字符之间的时间。两个字符间的间隔不得超过字符延迟时间。

发送连接清除

如果通讯伙伴在活动的发送操作期间发送 NAK 字符，则程序将取消传输块，然后按上述步骤重试一次。如果发送其它字符，则程序将首先等待字符延迟时间结束，然后发送 NAK 字符将通讯伙伴的状态更改为空闲。然后程序将开始通过 STX 建立连接以再次发送数据。

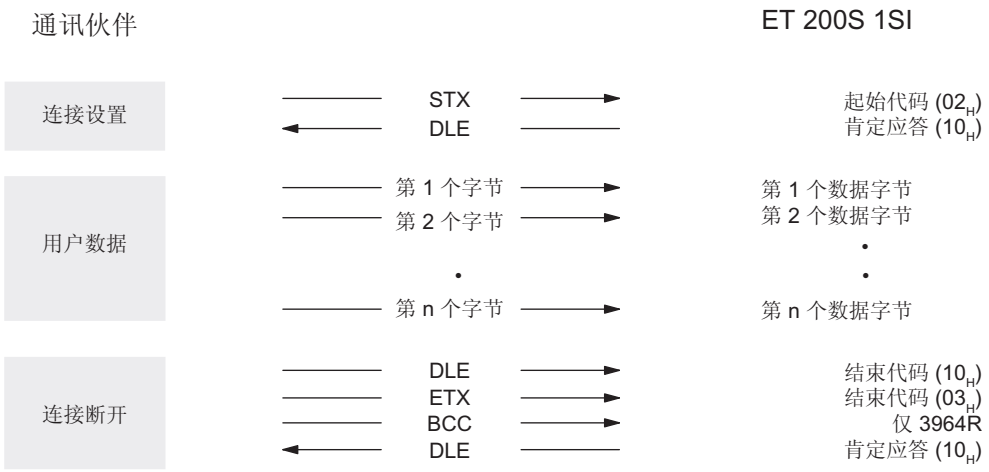
发送完缓冲区中的内容后，程序将添加字符 DLE、ETX（**仅对于 3964[R]**）和 BCC 块校验字符作为结束代码，并等待确认字符。如果通讯伙伴在确认延迟时间内发送 DLE 字符，则意味着数据块已无错接收。如果通讯伙伴以 NAK、任何其它字符（DLE 除外）或损坏的字符进行响应，或在确认延迟时间结束之前无响应，则程序将开始通过 STX 建立连接以再次发送数据。

尝试发送数据块的次数达到定义的次数后，程序将停止尝试，并将 NAK 发送给通讯伙伴。系统程序将向 P_SEND 功能块（STATUS 输出参数）报告错误。

2.7.3 使用 3964(R) 程序发送数据

使用 3964(R) 接收数据

下图说明了使用 3964(R) 程序接收数据时的传输顺序。



图片 2-13 使用 3964(R) 程序接收时的数据通讯

建立接收连接

在空闲状态下，如果没有要处理的发送作业，程序将等待通讯伙伴建立连接。

如果空闲程序接收了任何控制字符（STX 或 NAK 除外），它将等待字符延迟时间结束，然后发送 NAK 字符。

接收数据

如果程序接收了 STX 字符并且有可用的空接收缓冲区，则以 DLE 进行响应。引入的接收字符随后会存储在接收缓冲区中。如果接收到两个连续的 DLE 字符，则只有其中一个存储在接收缓冲区中。

每接收一个字符，程序都要等到字符延迟时间结束后再接收下一个字符。如果在接收到另一个字符前字符延迟时间结束，则将 NAK 字符发送给通讯伙伴。系统程序将向 P_RCV 功能块（STATUS 输出参数）报告错误。

如果通过 STX 建立连接期间没有可用的空接收缓冲区，则将开始 400 ms 的等待时间。如果此时间结束后仍没有空接收缓冲区，则系统程序将报告错误（FB 的 STATUS 输出中的错误消息），然后程序将发送 NAK 字符并返回空闲状态。否则程序将发送 DLE 字符并按上述步骤接收数据。

接收连接清除

如果接收过程中发生传输错误（丢失字符、帧出错、奇偶校验出错等），程序将继续接收直到连接关闭，然后将 NAK 发送给通讯伙伴。然后重复以上步骤。如果在参数化期间达到指定的重复尝试次数后仍无法在无错状态下接收块，或者通讯伙伴没有在 4 秒的块等待时间内开始重复，则程序将取消接收操作。系统程序将向 P_RCV 功能块（STATUS 输出参数）报告错误。

如果 **3964(R)** 程序检测到 DLE ETX 字符串，它将停止接收并通过向通讯伙伴发送 DLE 字符确认已成功接收块。如果在接收的数据中发现错误，程序将向通讯伙伴输出 NAK 字符。然后重复以上步骤。

如果 **3964(R)** 程序检测到字符串 DLE ETX BCC，它将停止接收。它将接收到的 BCC 块校验字符与内部计算的纵向奇偶校验加以比较。如果块校验字符正确且未出现其它接收错误，则 3964(R) 程序将发送 DLE 字符并返回空闲状态。如果 BCC 错误或者出现其它接收错误，则向通讯伙伴发送一个 NAK 字符。然后重复以上步骤。

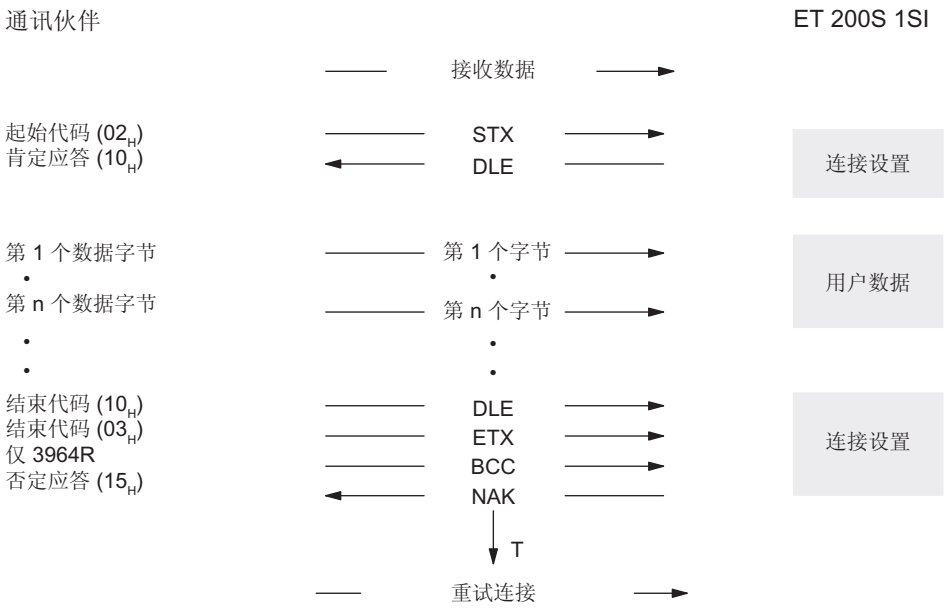
说明

一旦准备就绪，3964(R) 程序会立即将一个 NAK 字符发送给通讯伙伴，将通讯伙伴设置为空闲状态。

2.7.4 使用 3964(R) 程序的数据传输

处理错误数据

下图说明了使用 3964(R) 程序处理错误数据的方法。



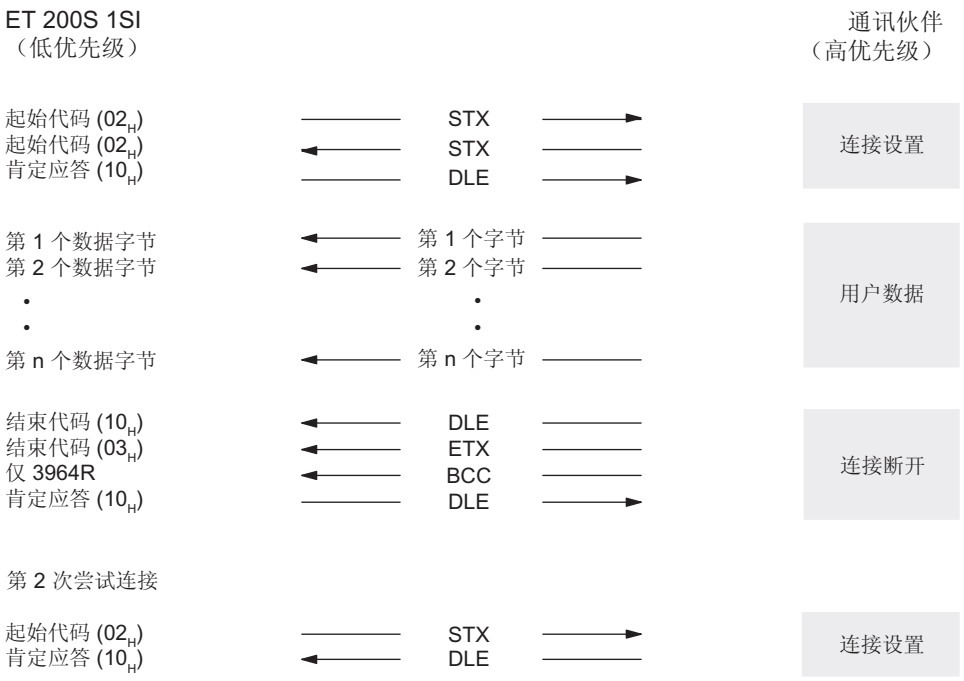
图片 2-14 接收到错误数据时的数据通讯

接收 DLE、ETX、BCC 之后，ET 200S 1SI 模块会将通讯伙伴的 BCC 与其内部计算的值进行比较。如果 BCC 正确且未发生其它接收错误，则 ET 200S 1SI 模块将以 DLE 进行响应。

否则，其将以 NAK 字符进行响应并等待 4 秒的块等待时间 (T) 以进行新尝试。如果在定义的传输尝试次数内接收不到该块，或者在块等待时间内未再次尝试，则 ET 200S 1SI 模块将取消接收操作。

初始化冲突

下图说明了初始化冲突时的传输顺序。



图片 2-15　　初始化冲突时的数据通讯

如果一个设备在确认延迟时间内通过发送 STX 字符而不是发送 DLE 或 NAK 确认来响应通讯伙伴的发送请求 (STX)，则会出现初始化冲突。双方设备都想执行未决发送作业。低优先级的设备撤销其发送作业并以 DLE 字符进行响应。高优先级的设备将按上述方式发送其数据。一旦终止连接，低优先级设备便可执行其发送作业。

为了解决初始化冲突，必须为通讯伙伴参数化不同的优先级。

程序错误

程序可识别由通讯伙伴故障引起的错误和由线路故障引起的错误。

在这两种情况下，程序都将重复尝试正确地发送/接收数据块。如果在设置的最大重复尝试次数内无法实现（或者如果出现新的错误状态），程序将取消发送或接收过程。程序将报告检测到的第一个错误的错误编号，然后返回空闲状态。这些错误消息在 FB 的 STATUS 输出中显示。

如果系统程序时常在 FB 的 STATUS 输出中报告发送和接收重复的错误编号，则表明数据通讯中偶尔有干扰。但是，高重复频率可以将其抵消。在这种情况下，建议检查传输链接以查找可能的干扰源，因为频繁重复会降低用户数据的传输率和传输的完整性。但是，干扰也可能是由通讯伙伴方的故障引起。

如果接收线路中断，则系统程序将报告 BREAK 状态（通过 ET 200S 模块上的诊断中断显示中断）（请参见『诊断（页码 101）』一节）。不启动任何重复。恢复线路连接后，BREAK 状态会自动复位。

对于检测到的每个传输错误（丢失字符、帧或奇偶校验错误），无论该错误是在发送还是接收数据块期间检测到的，都将报告标准编号。但仅在前一次重复尝试失败后才会报告该错误。

2.8 使用 ASCII 驱动程序的数据传输

2.8.1 有关使用 ASCII 驱动程序进行数据传输的基本信息

简介

ASCII 驱动程序通过 ET 200S 1SI 模块和通讯伙伴之间的点对点连接控制数据传输。ASCII 驱动程序包含物理层（第 1 层）。

消息帧的结构保持开放，因为是由 S7 用户将完整的发送消息帧传输给 ET 200S 1SI 模块。对于接收方向，必须参数化消息帧的结束标准。发送消息帧的结构与接收消息帧的结构可能会不同。

ASCII 驱动程序允许发送和接收任何结构的数据（所有可打印的 ASCII 字符以及从 00 到 FF_H [如果是包含 8 个数据位的字符帧] 或从 00 到 $7F_H$ [如果是包含 7 个数据位的字符帧] 的所有其它字符）。

也参见

有关通过功能块进行通讯的基本信息（页码 67）

启动特性和操作模式（页码 86）

2.8.2 使用 ASCII 驱动程序发送数据

使用 ASCII 驱动程序发送数据

发送数据时，指定调用 P_SEND 功能块时 LEN 参数中要传输的用户数据的字节数。用户数据必须包含所需的任何起始和结束字符。

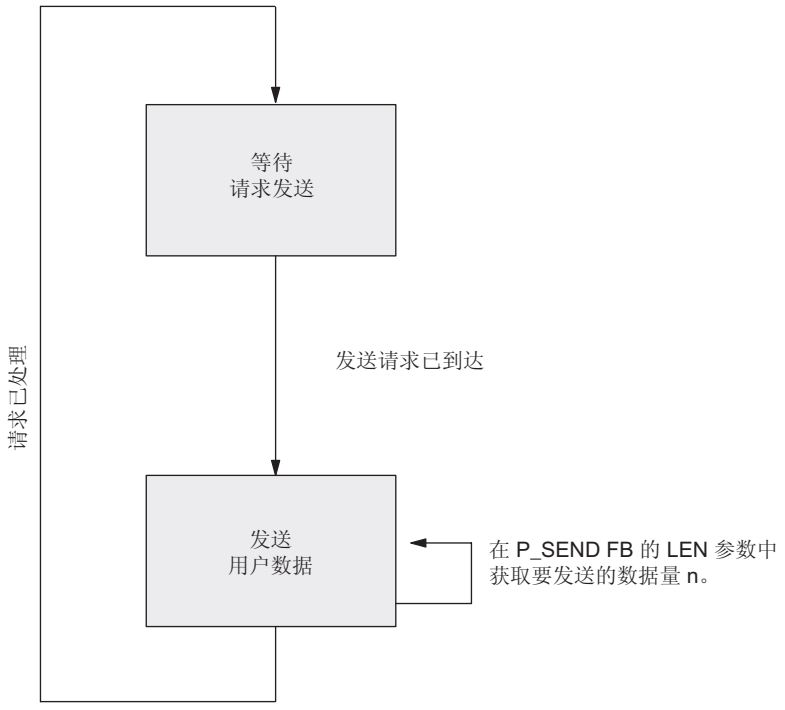
如果接收数据时您使用结束标准“字符延迟时间结束”，则 ASCII 驱动程序将在发送时在两个消息帧之间暂停。可以随时调用 P_SEND FB，但是因为最后一个消息帧已发送，所以直到比参数化的字符延迟时间还长的周期结束后，ASCII 驱动程序才开始输出。

说明

在参数化 XON/XOFF 流量控制后，用户数据不得包含任何已参数化的 XON 或 XOFF 字符。缺省设置为 DC1 = 11_H（对于 XON）和 DC3 = 13_H（对于 XOFF）。

发送数据

下图显示了传输顺序。



图片 2-16 发送操作的顺序

2.8.3 使用 ASCII 驱动程序接收数据

使用 ASCII 驱动程序接收数据

对于使用 ASCII 驱动程序的数据传输，可在三种不同的结束标准中进行选择。结束标准定义了完整接收消息帧的位置。可组态的结束标准有：

- **字符延迟时间结束**

  消息帧没有固定长度，也没有定义的结束字符；消息帧的结束由线路上的暂停（字符延迟时间结束）定义。各个波特率的最小值将在稍后列出。

- **结束字符的接收**

  消息帧的结束由一个或两个定义的结束字符标记。

- **固定数目字符的接收**

  接收消息帧的长度始终相同。

代码透明度

程序的代码透明度取决于所组态的结束标准和流量控制的选择：

- 使用一个或两个结束字符
  - 非代码透明
- 结束标准“字符延迟时间结束”或“固定数目字符的接收”
  - 代码透明
- 代码透明的操作不适用于 XON/XOFF 流量控制操作。

“代码透明”表示用户数据可以包含任意字符组合，无需识别结束标准。

取决于波特率的最小字符延迟时间

字符延迟时间的最小值取决于波特率。下表列出了各种波特率对应的最小字符延迟时间（单位：ms）。

表格 2-7　　最小字符延迟时间

波特率	最小字符延迟时间
115	365 ms
300	130 ms
600	65 ms
1,200	32 ms
2,400	16 ms
4,800	8 ms
9,600	4 ms
19,200	2 ms
38,400	1 ms
57,600	1 ms
76,800	1 ms
115,200	1 ms

ET 200S 模块的接收缓冲区

ET 200S 1SI 接口模块接收缓冲区的大小为 1,024 个字节。参数化期间，可以指定是否应在启动时清空接收缓冲区，或是否应通过防止覆盖来保留接收缓冲区中的数据。另外，您还可以激活或取消激活对接收到的消息帧的缓冲。

ET 200S 1SI 串行接口模块的接收缓冲区是个环形缓冲区。

- 如果多个消息帧被写入 ET 200S 1SI 模块的接收缓冲区：ET 200S 1SI 模块始终将最早的消息帧发送给 CPU。
- 如果您仅想将最新的消息帧发送给 CPU，则必须取消激活动态消息帧且关闭覆盖保护。

说明

如果在用户程序中连续读出接收数据的过程被中断片刻，您可能会发现当再次请求接收数据时，CPU 在接收到最新的消息帧之前从 ET 200S 1SI 模块接收了一个旧消息帧。

旧消息帧是在中断时 ET 200S 1SI 和 CPU 之间的帧，或已由 FB 接收的帧。

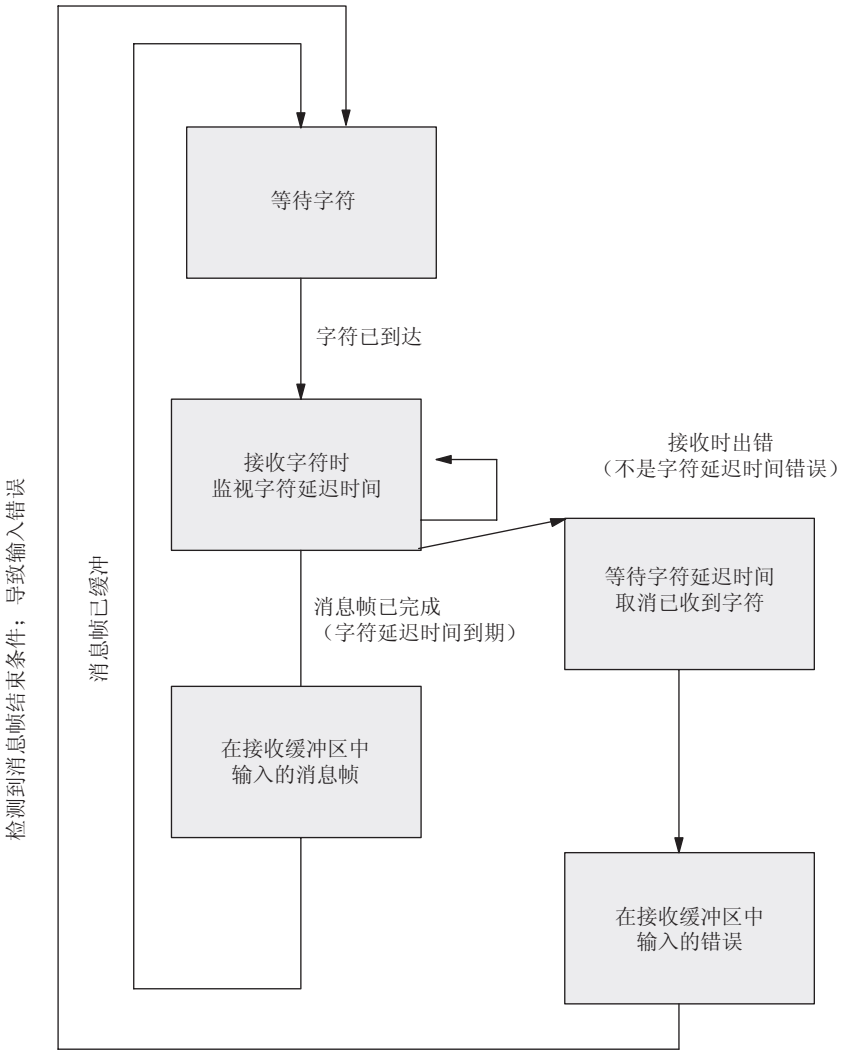
2.8.4 使用 ASCII 驱动程序进行数据传输的结束标准

结束标准“字符延迟时间结束”

接收数据时，如果字符延迟时间结束，则识别为消息帧结束。接收的数据由 CPU 使用 S_RCV 功能块接受。

在这种情况下，字符延迟时间必须足够短，以在两个连续消息帧的第二个帧之前结束。但是，它还应该足够长，以防在消息帧发送（由通讯伙伴启动）过程中对消息帧结束错误暂停。

下图说明了使用结束标准“字符延迟时间结束”的接收操作。



图片 2-17　使用结束标准“字符延迟时间结束”接收的流程图

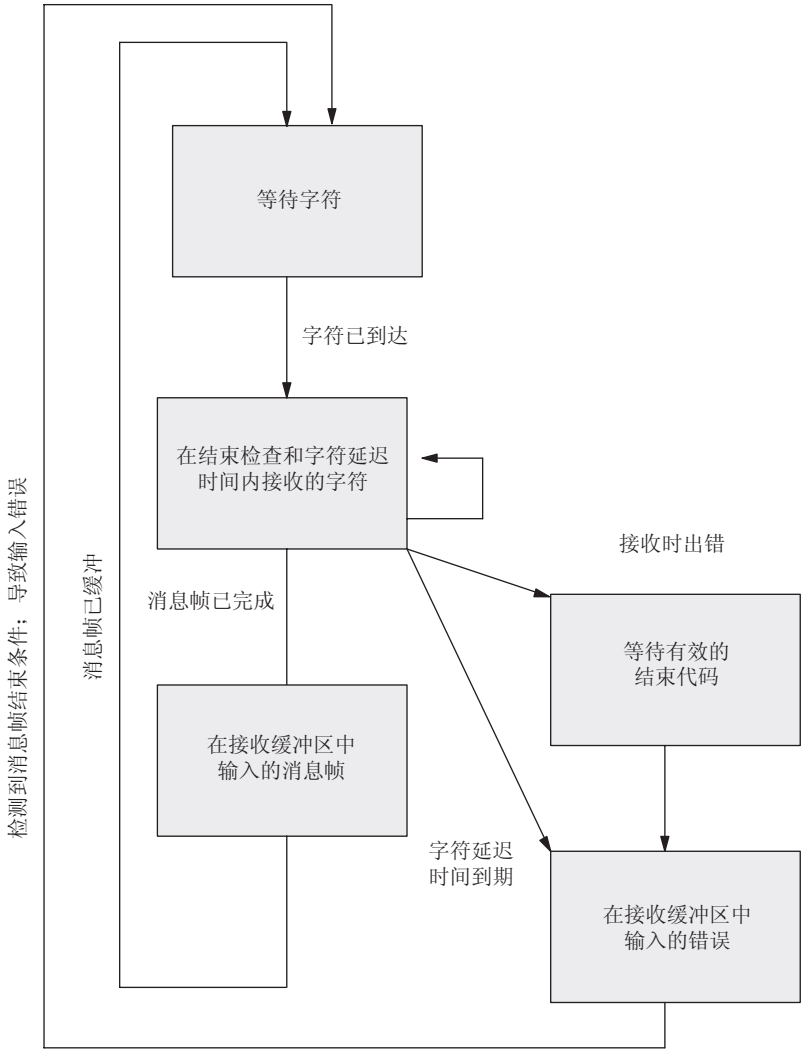
结束标准“结束字符的接收”

接收数据时，如果接收到定义的结束字符，则识别为消息帧结束。接收的数据（包括结束字符）由 CPU 使用 S_RCV 功能块接受。

如果在接收数据的同时字符延迟时间结束，则接收操作将结束。发出一条错误消息并丢弃消息帧碎片。

如果使用结束字符，则传输不是代码透明的。因此，必须确保结束代码不包含在用户数据中。

下图说明了使用结束标准“结束字符的接收”的接收操作。



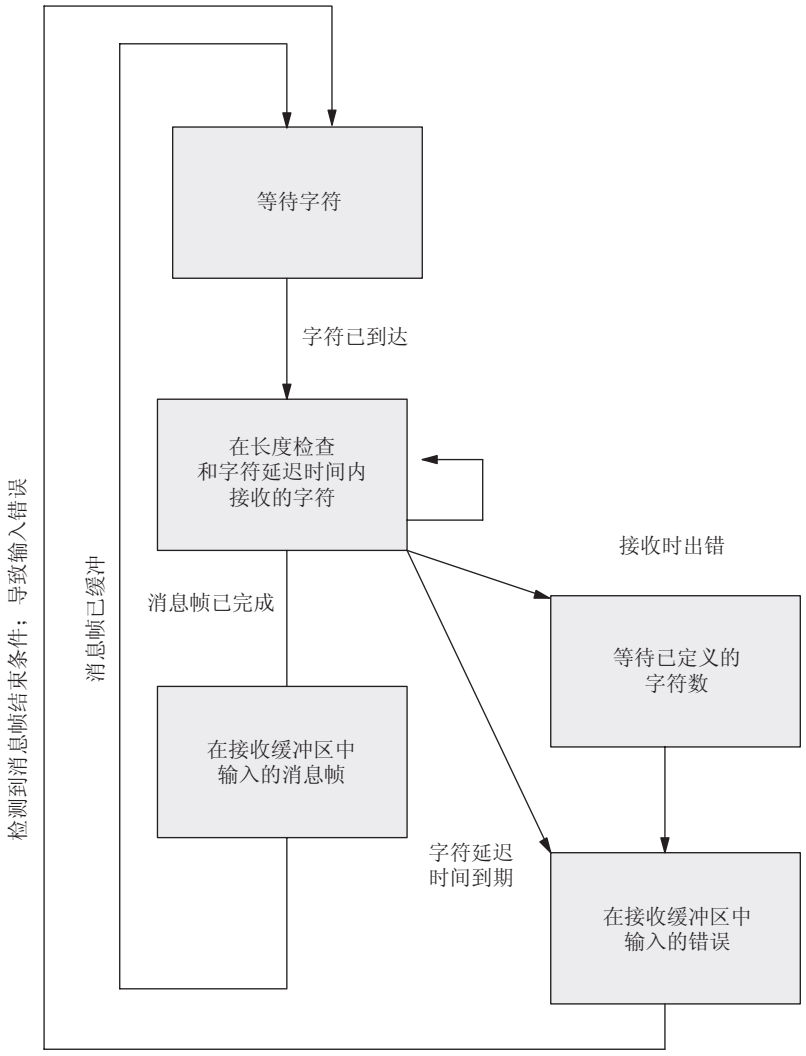
图片 2-18　　使用结束标准“结束字符的接收”接收的流程图

接收标准“固定数目字符的接收”

接收数据时，接收到定义的字符数目后，识别为消息帧结束。接收的数据由 CPU 使用 S_RCV 功能块接受。

如果在达到定义的字符数目之前字符延迟时间结束，则接收操作将结束。发出一条错误消息并丢弃消息帧碎片。

下图说明了使用结束标准“固定数目字符的接收”的接收操作。



图片 2-19　　使用结束标准“固定数目字符的接收”接收的流程图

2.8.5 用于使用 ASCII 驱动程序进行数据传输的 RS 232C 辅助信号是什么？

RS 232C 伴随信号

ET 200S 1SI 模块支持以下 RS 232C 伴随信号：

- DCD （输入） 数据载体检测
- DTR （输出） 数据终端准备就绪；ET 200S 1SI 已做好运行准备
- DSR （输入） 数据集准备就绪；通讯伙伴已做好运行准备
- RTS （输出） 请求发送；ET 200S 1SI 明确发送
- CTS （输入） 明确发送；通讯伙伴可以从 ET 200S 1SI 模块接收数据（对 ET 200S 1SI 的 RTS = ON 进行响应）。

ET 200S 1SI 模块通电后，输出信号处于 OFF 状态（取消激活）。

可以通过参数化接口组态 DTR/DSR 和 RTS/CTS 控制信号的使用，或通过用户程序中的功能 (FC) 控制它们的使用：

使用 RS 232C 伴随信号

RS 232C 伴随信号可在下列情况下使用：

- 在组态所有 RS 232C 伴随信号的自动控制后
- 在组态数据流量控制 (RTS/CTS) 后
- 通过 V24_STAT 和 V24_SET 功能块 (FB)

说明

在组态 RS 232C 伴随信号的自动控制后，无论是 RTS/CTS 数据流量控制还是 RTS 和 DTR 控制，都无法通过 S_VSET FB 实现。

在组态 RTS/CTS 数据流量控制后，则无法通过 S_VSET FB 实现 RTS 控制。

另一方面，始终可以通过 S_VSET FB 来读取所有的 RS 232C 二次信号。

以下章节说明了控制和评估 RS 232C 伴随信号的基本原理。

RS 232C 伴随信号的自动控制

在 ET 200S 1SI 模块上执行 RS 232C 伴随信号的自动控制的操作如下：

- 一旦组态 ET 200S 1SI 模块使其使用 RS 232C 伴随信号的自动操作模式，便将 RTS 线路设置为 OFF 并将 DTR 线路设置为 ON（ET 200S 1SI 已做好运行准备）。

  这可以防止消息帧的传输，除非 DTR 线路设置为 ON。只要 DTR = OFF，就无法通过 RS 232C 接口接收到任何数据。所有发送作业都将取消，同时生成一条相应的错误消息。

- **发送作业**排队时，模块将设置 RTS = ON，并触发已组态的数据输出等待时间。当超过数据输出时间并且 CTS = ON 时，数据通过 RS 232C 接口发送。

- 如果在数据输出等待时间内 CTS 线路未设置为 ON 或在传输过程中 CTS 更改为 OFF，则模块将取消发送作业并生成错误消息。

- 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。ET 200S 1SI 不会等待 CTS 更改为 OFF。

- DSR 线路设置为 ON 后，即可通过 RS 232C 接口**接收**数据。如果 ET 200S 1SI 模块的接收缓冲区即将发生溢出，则 ET 200S SI 模块不会进行响应。

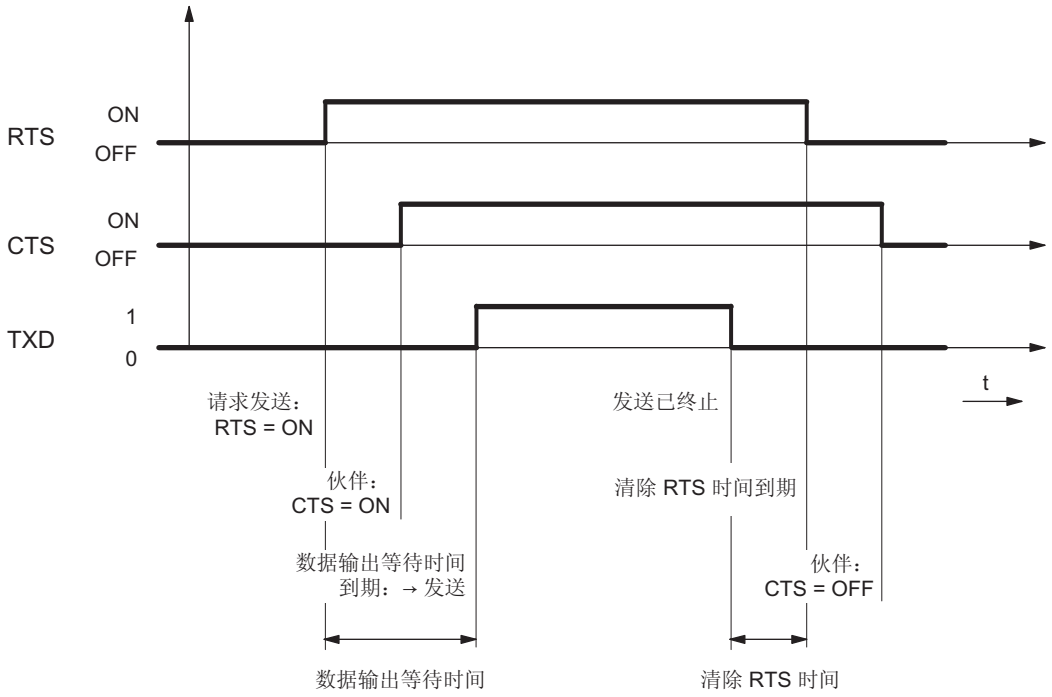
- 如果 DSR 从 ON 更改为 OFF，则将取消活动的发送作业或数据接收操作，并输出错误消息。

说明

在组态 RS 232C 伴随信号的自动控制后，无论是 RTS/CTS 数据流量控制还是 RTS 和 DTR 控制，都无法通过 S_VSET FB 实现。

时序图

下图说明了发送作业的时间顺序。



图片 2-20　　RS 232C 伴随信号的自动操作时序图

数据流量控制/握手

握手用于控制两个通讯伙伴之间的数据流量。握手可以确保数据在以不同速度运行的设备之间传输时不会丢失。握手有两种基本类型：

- 软件握手（例如 XON/XOFF）
- 硬件握手（例如 RTS/CTS）

ET 200S 1SI 模块的数据流量控制按照如下所述执行：

- 一旦组态 ET 200S 1SI 模块使其适用于在使用流量控制的模式下运行，即发送 XON 字符或将 RTS 线路设置为 ON。
- 在接收缓冲区发生溢出（接收缓冲区的大小：1,024 个字节）之前达到定义的消息帧数或者 50 个字符时，ET 200S 1SI 模块会发送 XOFF 字符或将 RTS 线路设置为 OFF。如果通讯伙伴忽略此状态并继续传输，则在接收缓冲区发生溢出时会生成一条错误消息。在最后一个消息帧中接收到的数据将被丢弃。
- 一旦 S7 CPU 获取一个消息帧且接收缓冲区可以进行接收，ET 200S 1SI 模块就会发送 XON 字符或将 RTS 线路设置为 ON。
- 如果 ET 200S 1SI 模块接收到 XOFF 字符或 CTS 控制信号设置为 OFF，则 ET 200S 1SI 模块将中断发送过程。如果既没有接收到 XON 字符，也没有将 CTS 设置为 ON，则一旦超过组态时间，就取消传输，并在功能块的 STATUS 输出中生成一条相应的错误消息 (0708_H)。

通过 FB S_VSTAT 和 FB S_VSET 读取/控制

S_VSTAT 功能块可用于确定每个 RS 232C 伴随信号的状态。S_VSET 功能块可用于控制 DTR 和 RTS 输出信号。在『有关通过功能块进行通讯的基本信息（页码 67）』一节中，可以找到有关如何将功能块用作 CPU 和 ET 200S 1SI 模块之间的接口的信息。

2.9 组态和参数化串行接口模块

2.9.1 组态串行接口模块

原理

如果您要通过 PROFIBUS 网络使用 S7 主站与 ET 200S 1SI 接口模块通讯，则需要使用 STEP 7 硬件组态在 PROFIBUS 网络上安装模块并设置其通讯参数。

如果在硬件目录中选择 ET 200S 1SI 模块并将其添加到网络组态中的 ET 200S 基本模块中，模块的订货号、插槽号和输入与输出地址将自动传输到组态表中。然后，您可以调用 ET 200S 1SI 模块的属性对话框，并设置通讯类型和其它参数。

2.9.2 为 ASCII 驱动程序分配参数

原理

下表列出了可以为串行接口模块的 ASCII 驱动程序设置的参数。

表格 2-8 ASCII 驱动程序的参数

参数	说明	值范围	缺省值
诊断中断	指定在出现严重错误时，模块是否应生成诊断中断。	• 否 • 是	否
激活 BREAK 检测	如果发生线路断路或未连接接口电缆，该模块将产生错误消息“断路”。	• 否 • 是	否
接口类型	指定要使用的电接口（请参见『RS 232C 接口（页码 28）』和『RS 422/485 接口（页码 29）』两节）。	• RS 232C • RS 422（全双工） • RS 485（半双工）	RS 232C
接收线路的半双工和全双工初始状态	指定 RS 422 和 RS 485 操作模式下接收线路的初始状态。不是在 RS 232C 操作模式下使用的初始状态。 仅在更换零件时需要确保兼容性的情况下，才需要“反转信号电平”设置。	RS 422： R(A) 5 V/R(B) 0 V (BREAK) R(A) 0 V/R(B) 5 V 反转信号电平 RS 485： 无 R(A) 0 V/R(B) 5 V	RS 422： R(A) 5 V/R(B) 0 V (BREAK) RS 485： R(A) 0 V/R(B) 5 V

参数	说明	值范围	缺省值
数据流量控制 （使用缺省参数；在用户程序中更改缺省值）	您可以发送和接收具有数据流量控制的数据。如果一个通讯伙伴比另一个运行快，则可以通过数据流量控制使数据传输同步。选择数据流量控制类型并设置相关参数（请参见『有关使用 ASCII 驱动程序进行数据传输的基本信息（页码 46）』一节）。 注意：使用 RS 485 接口无法进行数据流量控制。使用“RTS/CTS”和“V.24 信号的自动控制”的数据流量控制仅在 RS 232C 接口上受到支持。	• 无 • XON/OFF • RTS/CTS • V.24 信号的自动控制	无
波特率	选择数据传输率（单位：位/秒）。	• 110 • 300 • 600 • 1,200 • 2,400 • 4,800 • 9,600 • 19,200 • 38400 • 57,600 • 76,800 • 115,200	9,600
数据位	选择一个字符对应的位数。	• 7 • 8	8
停止位	选择数据传输期间附加到每个字符的停止位数，以指示字符的结束。	• 1 • 2	1

参数	说明	值范围	缺省值
奇偶校验	可以将数据位序列扩展一个字符以包括奇偶校验位。附加值（0 或 1）将所有位（数据位和奇偶校验位）的值设置为已定义状态。 **无**：发送的数据不包含奇偶校验位。 **奇数**：设置奇偶校验位，以便所有数据位（包括奇偶校验位）的总数返回一个信号状态为“1”的奇数值。 **偶数**：设置奇偶校验位，以便所有数据位（包括奇偶校验位）的总数返回一个信号状态为“1”的偶数值。 **任意**：与奇偶校验位的信号状态无关。接收数据时不检查奇偶校验位，但是发送数据时奇偶校验位始终设置为“0”。	• 无 • 奇数 • 偶数 • 任意	偶数
指示接收消息帧结束	如果通过 ASCII 驱动程序传输数据，则有三种不同的方法可以检测接收消息帧的结束。在此，您可以选择三种传输模式之一并输入特定参数。 **注意**：如果在接收数据期间字符延迟时间结束，则在三种模式下都会提前取消接收操作。在其它非“字符延迟时间结束”模式下，消息帧将被丢弃。 • 字符延迟时间结束：在字符延迟时间结束时，检测到消息帧。 • 结束字符的接收：接收到定义的结束字符时，检测到消息帧结束。 • 固定数目字符的接收：根据定义的消息帧长度检测消息帧结束。要接收的所有消息帧长度都一样。	• 字符延迟时间结束 • 结束字符的接收 • 固定数目字符的接收	字符延迟时间结束
字符延迟时间结束（单位：ms）	两个字符之间的最大接收时间间隔。[1]	4 ms 到 65,535 ms	4 ms
结束字符 1[2]	可以定义两个结束字符的最大值以接收具有结束字符的数据。选定的结束字符将限制消息帧的长度。	• 具有 7 个数据位：[3] 1 到 $7F_H$ • 具有 8 个数据位：[3] 1 到 FF_H	3
结束字符 2[2]	可以定义两个结束字符的最大值以接收具有结束字符的数据。选定的结束字符将限制消息帧的长度。 第二个结束代码（如果指定）	• 具有 7 个数据位：[3] 0 到 $7F_H$ • 具有 8 个数据位：[3] 0 到 FF_H	0
接收时的消息帧长度 [4]	如果要接收包含固定字符数的数据，则指定消息帧长度。消息帧长度必须与通讯伙伴要接收的数据字节数完全对应。	1 到 224 个字节	100

参数	说明	值范围	缺省值
动态消息帧	为了接收消息，可以指定是仅缓冲一个消息还是动态缓冲消息。如果激活了动态消息帧选项，则模块可以缓冲多个不同长度的消息。相关缓冲区是个环形缓冲区。如果缓冲区已满，则覆盖最早的消息，除非激活了参数“防止覆盖缓冲区”。在激活了该参数的情况下，将丢弃最新的消息。在这两种情况下，诊断中断都指示数据已丢失。	• 激活 • 取消激活	激活
防止覆盖缓冲区	该参数可防止在模块接收到新消息帧但接收缓冲区已满时覆盖缓冲的消息帧。这可以防止丢失以前接收到的消息帧。	• 否 • 是	是
启动时清空 ET 200S 1SI 接收缓冲区	指定当 CPU 从 STOP 操作模式更改为 RUN 操作模式时（CPU 启动时）模块的接收缓冲区是否应自动清空。通过这种方式，可以确保模块的接收缓冲区仅包含 CPU 启动后接收的消息帧。	• 否 • 是	是

[1] 最小字符延迟时间取决于波特率。

[2] 仅当结束标准是“结束字符的接收”时才设置。

[3] 取决于是将字符帧设置为 7 个数据位还是 8 个数据位。

[4] 仅当结束标准是“固定数目字符的接收”时才设置。

2.9.3 为 3964(R) 协议的驱动程序分配参数

原理

下表列出了可以为串行接口模块的 3964(R) 协议设置的参数。

表格 2-9 3964(R) 协议的驱动程序参数

参数	说明	值范围	缺省值
诊断中断	指定在出现严重错误时，模块是否应生成诊断中断。	• 否 • 是	否
激活 BREAK 检测	如果发生线路断路或未连接接口电缆，该模块将产生错误消息“断路”。	• 否 • 是	否
接口类型	指定要使用的电气接口。	• RS 232C • RS 422	RS 232C
接收线路初始状态	指定 RS 422 操作模式下接收线路的初始状态。 不是在 RS 232C 操作模式下使用的初始状态。 仅在更换零件时需要确保兼容性的情况下，才需要“反转信号电平”设置。	R(A) 5 V/R(B) 0 V (BREAK) R(A) 0 V/R(B) 5 V 反转信号电平	R(A) 5 V/R(B) 0 V (BREAK)
协议操作模式	指定发送数据时是否应附有块校验字符 (BCC, block check character) 以增强数据完整性。 块校验字符相当于发送/接收块的纵向偶校验（所有数据字节的 EXOR 逻辑运算）。如果通讯伙伴在接收数据时检测到块校验字符，那么它会将 BCC 与内部计算的纵向奇偶校验进行比较。如果块校验字符无效，则等待 4 秒钟（块等待时间），然后重复进行数据传输。 如果尝试的次数超过定义的次数后仍未接收到块，或者在块等待时间内未再次尝试传输数据，那么将取消接收操作。	• 无 • 块校验 • 块校验	块校验
波特率	选择数据传输率（单位：位/秒）。	• 110 • 300 • 600 • 1,200 • 2,400 • 4,800 • 9,600 • 19,200 • 38,400 • 57,600 • 76,800 • 115,200	9,600

参数	说明	值范围	缺省值
数据位	选择一个字符对应的位数。	• 7 • 8	8
停止位	选择数据传输期间附加到每个字符的停止位数，以指示字符的结束。	• 1 • 2	1
奇偶校验	可以将数据位序列扩展一个字符以包括奇偶校验位。附加值（0 或 1）将所有位（数据位和奇偶校验位）的值设置为已定义状态。 • 无：发送的数据不包含奇偶校验位。 • 奇数：设置奇偶校验位，以便所有数据位（包括奇偶校验位）的总数返回一个信号状态为“1”的奇数值。 • 偶数：设置奇偶校验位，以便所有数据位（包括奇偶校验位）的总数返回一个信号状态为“1”的偶数值。 • 任意顺序：与奇偶校验位的信号状态无关。接收数据时不检查奇偶校验位，但是发送数据时奇偶校验位始终设置为“0”。	• 无 • 奇数 • 偶数 • 任意	偶数
字符延迟时间 (ms)	接收两个相邻字符时允许的最大时间间隔。 为您的应用设置最小字符延迟时间。请记住，必须根据波特率为字符延迟时间设置一个具体的最小值。	20 到 655,350 ms （增量为 10 ms）	220 ms
确认时间 (ms)	指定建立连接和清除连接时在接收到通讯伙伴的确认之前允许等待的最大时间。请记住，必须根据波特率为确认时间设置一个具体的最小值。	10 到 655,350 ms （增量为 10 ms）	2,000 ms （不带块校验时为 550 ms）
连接尝试次数	定义尝试建立连接的次数 (*n*)。 （在 *n* 次尝试失败后，该功能将被取消，并在 S_SEND 功能块的 STATUS 输出处显示错误。）	1 到 255	6
传输尝试次数	定义尝试传输消息帧的次数 (*n*)。（在不出错的情况下 *n* 次尝试发送消息帧失败后，该功能将被取消，并在 S_SEND 功能块的 STATUS 输出处显示错误。） 取消的可能原因： 奇偶校验错误， BCC 错误；奇偶校验错误， 在通讯伙伴所进行的参数化方式方面（例如，波特率、奇偶校验、字符帧、块校验字符、不同的协议）有差异	1 到 255	6
优先级	如果两个伙伴同时请求发送作业，则优先级较低的伙伴将延迟其发送作业。要进行数据传输，必须为一个通讯伙伴指定较高的优先级，而为另一个通讯伙伴指定较低的优先级。	• 高 • 低	低
启动时清空 ET 200S 1SI 接收缓冲区	指定当 CPU 从 STOP 操作模式更改为 RUN 操作模式时（CPU 启动时）模块的接收缓冲区是否应自动清空。通过这种方式，可以确保模块的接收缓冲区仅包含 CPU 启动后接收的消息帧。	• 否 • 是	是

2.9.4 标识数据

定义

标识数据是存储在模块上的信息，在以下情况下可为您提供支持：

- 对系统进行故障诊断
- 检查系统组态
- 尝试找出系统硬件的更改

标识数据使模块可以在线唯一识别。ET 200S 1SI 模块从 MLFB（西门子订货号）6ES7 138-4DFx1-0AE0 开始具有此数据。

要查看标识数据，请选择 PLC > 模块信息 (Module Information)，或者选择（如下所述）“读取数据记录”(Read Data Record)。

读取标识数据

用户可以通过选择“读取数据记录”(Read Data Record) 来访问特定的 ID 数据。

分配给相应索引的 ID 数据元素位于关联数据记录号下。

- 所有包含 ID 数据的数据记录长度均为 64 个字节。
- 数据记录的结构基于下表中显示的原理。

表格 2-10　包含 ID 数据的数据记录的基本结构

内容	长度（字节）	编码（十六进制）
标题信息		
系统状态列表 ID	2	F1 11
索引	2	00 0x
标识数据的长度	2	00 38
包含 ID 数据的块数	2	00 01
标识数据		
索引	2	00 0x
与相关索引关联的标识数据（请参见下表）	54	

ET 200S 1SI 模块的标识数据

表格 2-11　ET 200S 1SI 模块的标识数据

标识数据	访问	缺省	说明
索引 1（数据记录 231/只读）			
制造商	读取 （2 个字节）	00 2A 十六进制 （= 42 十进制）	将在此处存储制造商名称。 （42 十进制 = Siemens AG）
设备名称	读取 （20 个字节）	6ES7 138-4DFx1-0AB0	模块的订货号 x = 0 (ASCII/3964[R])，1 （模块/USS）
设备序列号	读取 （16 个字节）	模块的序列号保存到此参数中。这使模块具有唯一识别性。	
硬件版本	读取 （2 个字节）	这提供了有关模块产品版本的信息。	
软件版本	读取 （4 个字节）	这提供了有关模块固件版本的信息。	
统计版本号	读取 （2 个字节）	–	不支持
Profile_ID	读取 （2 个字节）	F6 00 十六进制	内部参数（到 PROFIBUS DP）
配置文件特定类型	读取 （2 个字节）	00 04 十六进制 （= 4 十进制）	内部参数（通讯模块，到 PROFIBUS DP）
I&M 版本	读取 （2 个字节）	00 00 十六进制 （= 0 十进制）	内部参数（到 PROFIBUS DP）
受支持的 I&M	读取 （2 个字节）	00 01 十六进制 （= 1 十进制）	内部参数（I&M0 和 I&M1，到 PROFIBUS DP）
索引 2（数据记录 232/读取和写入）			
HID	读取/写入（最多 32 个字符）	–	模块的更高层名称
OKZ	读取/写入（最多 22 个字符）	–	模块的位置指定

2.9.5 固件更新的后续装载

说明

您可以通过将固件更新装载到 ET 200S 1SI 的系统存储器来扩展功能并消除错误。

可以使用 HW Config 装载固件更新。

基本固件

基本固件随 ET 200S 1SI 一起提供。

要求

装载固件更新的要求如下：

- 必须能够从 PG/PC 在线访问 ET 200S 1SI。
- 新固件版本文件必须适用于您的 PG/PC 文件系统。

装载固件

执行以下操作来装载固件更新（仅当 IM 151 支持此功能时才可以装载）：

1. 打开 HW Config，并选择所需的 ET 200S 1SI 模块。
2. 选择 PLC > 更新固件 (Update Firmware)。

其余步骤在 STEP 7 在线帮助中进行了介绍。

说明

尝试为 ET 200S 1SI 模块装载固件文件之前，请将 CPU 切换至 STOP。

系统将输出一条消息指示已成功完成更新，并立即激活新固件。

完成 ET 200S 1SI 固件更新后，贴上一个标有新固件版本的新标签。

更新不成功

如果更新不成功，模块的红色 SF LED 将闪烁。重复更新。如果无法成功执行更新，请与您当地的 Siemens 办事处或销售代表联系。

LED 显示

表格 2-12　装载固件更新时的 LED 显示

状态	SF	TXD	RXD	备注	补救措施
固件更新正在进行	亮	亮	亮	-	-
固件更新已完成	亮	灭	灭	-	-
ET 200S 1SI 没有模块固件	闪烁 (2 Hz)	灭	灭	模块固件已删除，固件更新已取消，仍可以进行固件更新	重新装载固件
固件更新期间硬件出故障	闪烁 (2 Hz)	闪烁 (2 Hz)	闪烁 (2 Hz)	删除/写入操作失败	关闭模块的电源，然后再打开电源并重新装载固件。 检查模块是否有故障。

查看硬件和固件版本

您可以在 **STEP 7** 中的“模块状态”(Module Status) 标签对话框中查看 ET 200S 1SI 的当前硬件和固件版本。访问该对话框的步骤如下：

在 SIMATIC 管理器中：**文件 (File) > 打开 (Open) > 项目 (Project) > 打开 HW Config (Open HW Config) > 站 (Station) > 在线打开 (Open Online)**，然后双击 IM 151 模块。

2.10 通过功能块进行通讯

2.10.1 有关通过功能块进行通讯的基本信息

概述

CPU、ET 200S 1SI 和通讯伙伴之间的通讯基于功能块和 ET 200S 1SI 模块的协议。（您可以在第 2.12 节（页码 88）中找到有关通过非 Siemens [非 S7] 进行通讯的信息。）

功能块构成了 CPU 和 ET 200S 1SI 串行接口模块之间的软件接口。必须从用户程序中循环调用功能块。

与 CPU 建立通讯

无论何时启动 CPU，CPU 的系统服务都会为 ET 200S 1SI 模块分配当前参数。CPU 和 ET 200S 1SI 模块之间建立连接后，必须对 ET 200S 1SI 模块进行初始化。

每个功能块都有其自己的启动机制。必须先完成关联的启动机制，才能处理活动的作业。

ET 200S 1SI 模块可在 CPU 中触发诊断中断。操作系统有 2 个字节的中断信息可供用户使用。用户必须对中断信息 (OB82) 的评估进行编程。在诊断中断编程的过程中，不允许调用功能块。在功能块中并未禁用中断。

ET 200S 1SI 模块中发生了协议转换。根据选定的协议（3964[R] 程序或 ASCII 驱动程序），调整 ET 200S 1SI 模块的接口使其适合通讯伙伴的接口。

ET 200S 1SI 模块的功能块

S7-300 自动化系统为您提供大量功能块，这些功能块用于在用户程序中启动和控制 CPU 和 ET 200S 1SI 串行接口模块之间的通讯。下表列出了 ET 200S 1SI 模块使用的 FB。

表格 2-13　ET 200S 1SI 模块的功能块

FB	名称	含义
FB2	S_RCV	使用 S_RCV 功能块可以接收来自通讯伙伴的数据并将其存储在数据块中。
FB3	S_SEND	使用 S_SEND 功能块可以将数据块的全部或部分发送给通讯伙伴。
FB4	S_STAT	使用 S_STAT 功能块可以通过 ET 200S 1SI 模块的 RS 232C 接口读取信号状态。
FB5	S_SET	使用 S_SET 功能块可以设置/复位 ET 200S 1SI 模块的 RS 232C 接口的输出。
FB6	S_XON	如果已针对 XON/XOFF 流量控制对模块进行了参数化，则使用 S_XON 功能块可以设置其它参数。
FB7	S_RTS	如果已针对 RTS/CTS 流量控制对模块进行了参数化，则使用 S_RTS 功能块可以设置其它参数。
FB8	S_V24	如果已针对 V.24 信号的自动控制对模块进行了组态，则使用 S_V24 功能块可以设置其它参数。

说明

当 SEND/RECEIVE 块通讯处于活动状态时，不能在 CPU 上装载这些背景数据块。

也参见

技术规范（页码 107）

2.10.2 FB3 S_SEND 功能块

FB3 S_SEND：将数据发送给通讯伙伴

S_SEND FB 将某个域（由 DB_NO、DBB_NO 和 LEN 参数指定）从数据块传输到 ET 200S 1SI 模块。为了进行数据传输，将在周期中静态（无条件）调用 S_SEND FB，**或者**在时间控制的程序中调用 S_SEND FB。

通过 REQ 输入处的正跳沿启动数据传输。根据数据量的大小，数据可能通过多次调用（程序周期）进行传输。

通过将参数输入 R 处的信号状态设置为“1”，可以循环调用 S_SEND FB。这会取消到 ET 200S 1SI 模块的传输，并将 S_SEND FB 复位为其初始状态。ET 200S 1SI 模块已接收的数据仍将发送给通讯伙伴。如果输入 R 处的信号状态保持静态“1”，这表示已禁用发送。

要寻址的 ET 200S 1SI 模块的地址在 LADDR 参数中指定。

DONE 输出显示“作业已完成且无错”。ERROR 指示是否发生了错误。如果发生了错误，则在 STATUS 中显示相应的事件号（请参见标题为『诊断（页码 101）』的章节）。如果没有发生错误，STATUS 的值为 0。DONE 和 ERROR/STATUS 也在 S_SEND FB 的 RESET 处输出（请参见时序图）。如果出现错误，则复位二进制结果。如果块终止且无错，则二进制结果的状态为“1”。

启动

S_SEND FB 的 COM_RST 参数通知 FB 启动。

将启动 OB 中的 COM_RST 参数设置为 1。

在循环操作中调用 FB，而不设置或复位 COM_RST 参数。

如果设置了 COM_RST 参数：

- FB 包含有关 ET 200S 1SI 模块的信息（I/O [不管是不是分布式 I/O] 区域中的字节数）。
- FB 将复位并终止先前启动的所有作业（在 CPU 最后一次跳转到 STOP 之前）。

FB 获得有关 ET 200S 1SI 模块的信息后，它将复位 COM_RST 参数本身。

说明

S_SEND 功能块没有参数检查。如果参数化不正确，CPU 可能会切换到 STOP 模式。

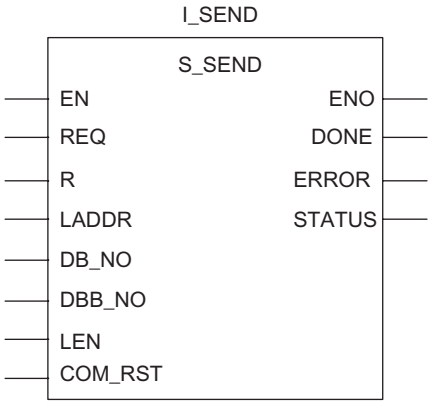
必须先完成 P_PRINT FB 的 ET 200S 模块 CPU 启动机制，才能在 CPU 从 STOP 模式切换至 RUN 模式后通过 ET 200S 1SI 模块处理启动的作业（请参见上文）。在此期间启动的任何作业都不会丢失。启动协调完成后，这些作业将传输至 ET 200S 1SI 模块。

FB3 调用

STL 表达式

```
CALL        S_SEND, I_SEND
   REQ:     =
   R:       =
   LADDR:   =
   DB_NO:   =
   DBB_NO:  =
   LEN:     =
   DONE:    =
   ERROR:   =
   STATUS:  =
   COM_RST: =
```

LAD 表达式



说明

参数 EN 和 ENO 仅存在于图形表达式（LAD 或 FBD）中。编译器使用二进制结果处理这些参数。

如果块终止且无错，则将二进制结果设置为信号状态“1”。如果出现错误，则将二进制结果设置为“0”。

数据存储区中的分配

S_SEND FB 与 I_SEND 背景数据块一起使用。调用时提供 DB 号。背景数据块中的数据无法访问。

说明

例外：如果出现错误 STATUS == W#16#1E0F，可以参考 SFCERR 变量以获得其它详细信息（请参见标题为『诊断』的章节）。该错误变量只能通过对背景数据块进行符号访问来装载。

FB3 S_SEND 参数

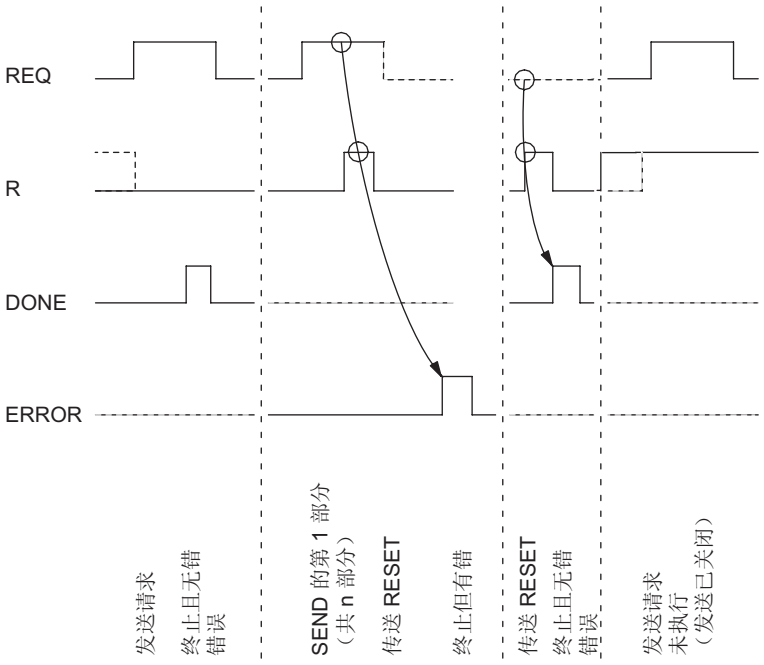
下表列出了 S_SEND (FB3) 参数。

表格 2-14　FB3：S_SEND 参数

名称	类型	数据类型	说明	允许的值、备注
REQ	INPUT	BOOL	出现正跳沿时启动作业	
R	INPUT	BOOL	取消作业	取消正在进行的作业。发送被阻。
LADDR	INPUT	INT	ET 200S 1SI 模块的起始地址	起始地址从 STEP 7 获取。
DB_NO	INPUT	INT	数据块号	发送的数据块号：依 CPU 而定（不允许为零）。
DBB_NO	INPUT	INT	数据字节号	0 ≤ DBB_NO ≤ 8190 从数据字传输数据
LEN	INPUT	INT	数据长度	1 ≤ LEN ≤ 224， 以字节数指定
DONE[1]	OUTPUT	BOOL	作业已完成且无错	STATUS 参数 == 16#00
ERROR[1]	OUTPUT	BOOL	作业已完成但有错	错误信息已写入 STATUS 参数。
STATUS[1]	OUTPUT	WORD	错误规范	如果 ERROR == 1，则 STATUS 参数将包含错误信息。
COM_RST	IN_OUT	BOOL	重新启动 FB	
[1] 在成功完成发送作业后的一个 CPU 周期内，DONE、ERROR 和 STATUS 参数都会用到。				

FB3 S_SEND 的时序图

下图根据 REQ 和 R 输入的接线方式说明了 DONE 和 ERROR 参数的特性。



图片 2-21　　FB3 S_SEND 的时序图

说明

REQ 输入通过沿来触发。REQ 输入处的正跳沿足以将其触发。传输期间，逻辑运算的结果并不一定是“1”。

2.10.3 FB2 S_RCV 功能块

FB S_RCV：从通讯伙伴接收到的数据

S_RCV FB 将数据从 ET 200S 1SI 模块传输到 S7 数据存储区（由 DB_NO 和 DBB_NO 参数指定）。为了进行数据传输，将循环调用 S_RCV FB，或者在时间控制的程序中静态（无条件）调用 S_RCV FB。

EN_R 参数的（静态）信号状态为“1”时会启用检查，以确定是否从 ET 200S 1SI 模块读取数据。EN_R 参数处的信号状态为“0”可以取消活动的传输事件。取消的接收作业终止，且生成错误消息（STATUS 输出）。只要 EN_R 参数处的信号状态为“0”，就会禁用接收。根据数据量的大小，数据可能通过多次调用（程序周期）进行传输。

如果功能块检测到“R”参数处的信号状态为“1”，则将取消当前的传输作业并将 S_RCV FB 复位为其初始状态。只要 R 参数处的信号状态为“1”，就会禁用接收。如果信号状态返回“0”，则从头重新开始接收已取消的消息帧。

要寻址的 ET 200S 1SI 模块的地址在 LADDR 参数中指定。

NDR 输出指示“作业已完成且无错/数据已接受”（已读取所有数据），ERROR 指示是否发生了错误。在 STATUS 中，如果出现错误将显示错误编号。如果接收缓冲区的 2/3 以上区域已满，则在每次调用 S_RCV 后 STATUS 都包含警告。如果未出现错误或警告，则状态值为“0”。

当复位 S_RCV FB（LEN 参数 == 16#00）时，还将输出 NDR 和 ERROR/STATUS（请参见时序图）。如果出现错误，则复位二进制结果。如果块终止且无错，则二进制结果的状态为“1”。

启动

S_RCV FB 的 COM_RST 参数通知 FB 启动。

将启动 OB 中的 COM_RST 参数设置为 1。

在循环操作中调用 FB，而不设置或复位 COM_RST 参数。

如果设置了 COM_RST 参数：

- FB 包含有关 ET 200S 1SI 模块的信息（I/O [不管是不是分布式 I/O] 区域中的字节数）。
- FB 将复位并终止先前启动的所有作业（在 CPU 最后一次跳转到 STOP 之前）。

FB 获得有关 ET 200S 1SI 模块的信息后，它将复位 COM_RST 参数本身。

说明

S_RCV 功能块没有参数检查。如果参数化不正确，CPU 可能会切换到 STOP 模式。

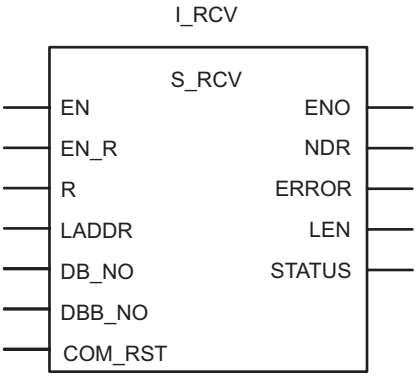
必须先完成 S_RCV FB 的 ET 200S 模块 CPU 启动机制，才能在 CPU 从 STOP 模式切换至 RUN 模式后通过 ET 200S 1SI 模块接收启动的作业（见上文）。

FB2 调用

STL 表达式

```
CALL            S_RCV, I_RCV
      EN_R:     =
      R:        =
      LADDR:    =
      DB_NO:    =
      DBB_NO:   =
      NDR:      =
      ERROR:    =
      LEN:      =
      STATUS:   =
      COM_RST:  =
```

LAD 表达式



说明

参数 EN 和 ENO 仅存在于图形表达式（LAD 或 FBD）中。编译器使用二进制结果处理这些参数。

如果块终止且无错，则将二进制结果设置为信号状态“1”。如果出现错误，则将二进制结果设置为“0”。

数据存储区中的分配

S_RCV FB 与 I_RCV 背景数据块一起使用。调用时提供 DB 号。背景数据块中的数据无法访问。

说明

例外：如果出现错误 STATUS == W#16#1Exx，可以参考 SFCERR 变量以获得其它详细信息。该错误变量只能通过对背景数据块进行符号访问来装载。

FB2 S_RCV 参数

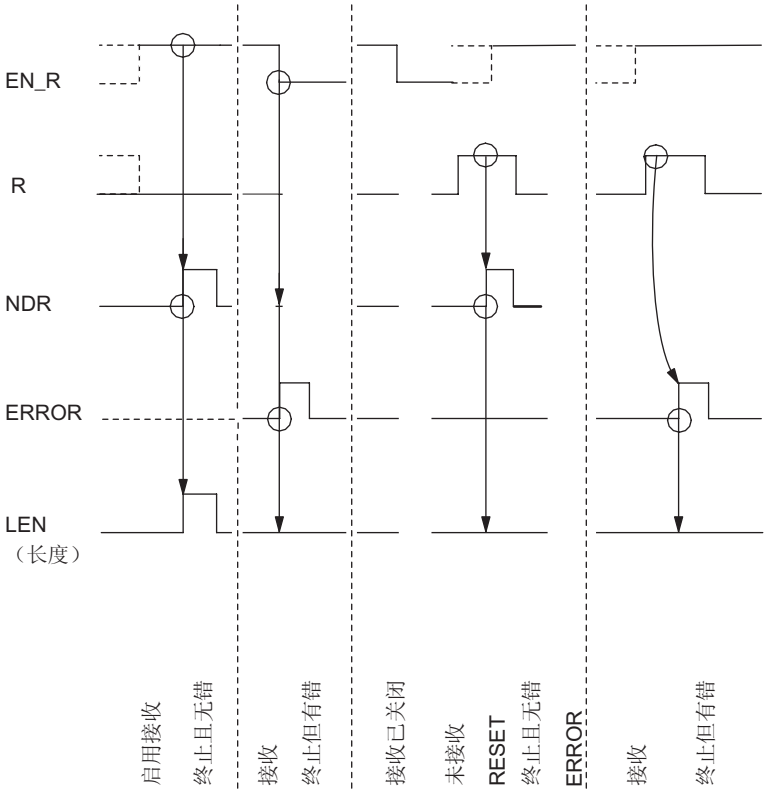
下表列出了 S_RCV (FB) 参数。

表格 2-15　FB2：S_RCV 参数

名称	类型	数据类型	说明	允许的值、备注
EN_R	INPUT	BOOL	启用数据读取	
R	INPUT	BOOL	取消作业	取消正在进行的作业。接收被阻。
LADDR	INPUT	INT	ET 200S 1SI 模块的起始地址	起始地址从 STEP 7 获取。
DB_NO	INPUT	INT	数据块号	接收数据块号：依 CPU 而定，不允许为零
DBB_NO	INPUT	INT	数据字节号	0 ≤ DBB_NO ≤ 8190 从数据字接收数据
NDR[1]	OUTPUT	BOOL	作业已完成且无错，数据已接受	STATUS 参数 == 16#00
ERROR[1]	OUTPUT	BOOL	作业已完成但有错	错误信息已写入 STATUS 参数。
LEN[1]	OUTPUT	INT	已接收消息帧的长度	1 ≤ LEN ≤ 224， 以字节数指定
STATUS[1]	OUTPUT	WORD	错误规范	如果 ERROR == 1，则 STATUS 参数将包含错误信息。
COM_RST	IN_OUT	BOOL	重新启动 FB	
[1] 在成功完成接收作业后的一个 CPU 周期内，NDR、ERROR、LEN 和 STATUS 参数都会用到。				

FB2 S_RCV 的时序图

下图根据 EN_R 和 R 输入的接线方式说明了 NDR、LEN 和 ERROR 参数的特性。



图片 2-22　　FB2 S_RCV 的时序图

说明

必须将 EN_R 输入设置为静态“1”。在整个接收作业过程中，必须为 EN_R 参数提供逻辑运算结果“1”。

2.10.4 数据流量控制选项的参数分配功能

原理

如果您将 ET 200S 1SI 串行接口模块与 S7 CPU 一起使用，并使用 STEP 7 硬件组态程序来组态模块，可以选择以下数据流量控制方法之一：

- 无
- XON/XOFF
- RTS/CTS
- V.24 信号的自动控制

可以为其中每个选项设置其它参数。这些参数的缺省值是典型值，适用于大多数应用场合。但是，您可以通过用户程序和以下功能块更改这些参数。

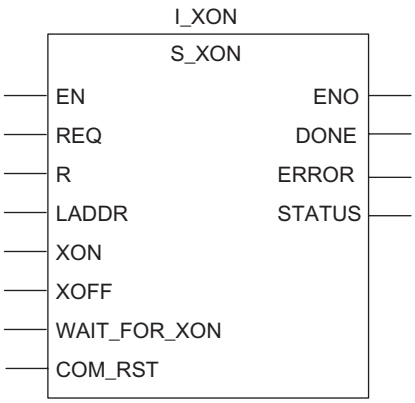
FB6 S_XON：设置 XON/XOFF 的字符

如果已针对 XON/XOFF 流量控制对模块进行了参数化，则使用 S_XON 功能块可以设置其它参数（请参见 FB6 参数）。

STL 表达式

```
CALL        S_XON, I_XON
   REQ:      =
   R:        =
   LADDR:    =
   XON:      =
   XOFF:     =
   WAIT_FOR_XON:  =
   DONE:     =
   ERROR:    =
   STATUS:   =
   COM_RST:  =
```

LAD 表达式



数据存储区中的分配

S_XON FB 与 I_XON 背景数据块一起使用。调用时提供 DB 号。背景数据块中的数据无法访问。

说明

例外：如果出现错误 STATUS == W#16#1Exx，可以参考 SFCERR 变量以获得其它详细信息（请参见诊断）。该错误变量只能通过对背景数据块进行符号访问来装载。

FB6 参数

下表列出了 FB6 参数。

表格 2-16　FB6：S_XON 参数

名称	类型	数据类型	说明	允许的值、备注	缺省
REQ	INPUT	BOOL	出现正跳沿时启动作业		
R	INPUT	BOOL	取消作业	取消正在进行的作业。发送受阻。	
LADDR	INPUT	INT	ET 200S 1SI 模块的起始地址	起始地址从 STEP 7 获取。	
XON	INPUT	BYTE	XON 字符	0 到 $7F_H$（7 个数据位） 0 到 FF_H（8 个数据位）	11 (DC1)
XOFF	INPUT	BYTE	XOFF 字符	0 到 $7F_H$（7 个数据位） 0 到 FF_H（8 个数据位）	13 (DC3)
WAIT_FOR_XON	INPUT	TIME	XOFF 后 XON 的等待时间	20 ms 到 10 min 55 s 350 ms	2 s
DONE[1]	OUTPUT	BOOL	作业已完成且无错	STATUS 参数 == 16#00	
ERROR[1]	OUTPUT	BOOL	作业已完成但有错	错误信息已写入 STATUS 参数。	
STATUS[1]	OUTPUT	WORD	错误规范	如果 ERROR == 1，则 STATUS 参数将包含错误信息。	
COM_RST	IN_OUT	BOOL	重新启动 FB		
[1] 在成功完成作业后的一个 CPU 周期内，DONE、ERROR 和 STATUS 参数都会用到。					

启动

S_XON FB 的 COM_RST 参数通知 FB 启动。

将启动 OB 中的 COM_RST 参数设置为 1。

在循环操作中调用 FB，而不设置或复位 COM_RST 参数。

如果设置了 COM_RST 参数：

- FB 包含有关 ET 200S 1SI 模块的信息（I/O [不管是不是分布式 I/O] 区域中的字节数）。
- FB 将复位并终止先前启动的所有作业（在 CPU 最后一次跳转到 STOP 之前）。

FB 获得有关 ET 200S 1SI 模块的信息后，它将复位 COM_RST 参数本身。

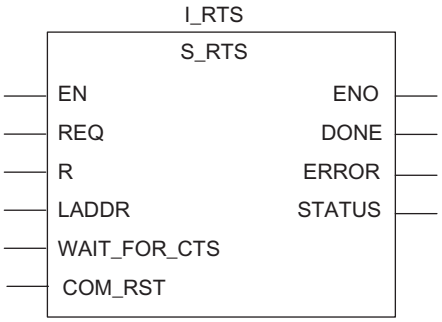
FB7 S_RTS：设置 RTS/CTS 的参数

如果已针对 RTS/CTS 流量控制对模块进行了参数化，则使用 S_RTS 功能块可以设置其它参数（请参见 FB7 参数）。

STL 表达式

```
CALL    S_RTS, I_RTS
   REQ:          =
   R:            =
   LADDR:        =
   WAIT_FOR_CTS: =
   DONE:         =
   ERROR:        =
   STATUS:       =
   COM_RST:      =
```

LAD 表达式



数据存储区中的分配

S_RTS FB 与 I_RTS 背景数据块一起使用。调用时提供 DB 号。背景数据块中的数据无法访问。

说明

例外：如果出现错误 STATUS == W#16#1Exx，可以参考 SFCERR 变量以获得其它详细信息（请参见诊断）。该错误变量只能通过对背景数据块进行符号访问来装载。

FB7 参数

下表列出了 FB7 参数。

表格 2-17　FB7：S_RTS 参数

名称	类型	数据类型	说明	允许的值、备注	缺省
REQ	INPUT	BOOL	出现正跳沿时启动作业		
R	INPUT	BOOL	取消作业	取消正在进行的作业。发送受阻。	
LADDR	INPUT	INT	ET 200S SI 模块的起始地址	起始地址从 STEP 7 获取。	
WAIT_FOR_CTS	INPUT	TIME	CTS = ON 的等待时间	20 ms 到 10 min 55 s 350 ms	2 s
DONE[1]	OUTPUT	BOOL	作业已完成且无错	STATUS 参数 == 16#00	
ERROR[1]	OUTPUT	BOOL	作业已完成但有错	错误信息已写入 STATUS 参数。	
STATUS[1]	OUTPUT	WORD	错误规范	如果 ERROR == 1，则 STATUS 参数将包含错误信息。	
COM_RST	IN_OUT	BOOL	重新启动 FB		
[1] 在成功完成作业后的一个 CPU 周期内，DONE、ERROR 和 STATUS 参数都会用到。					

启动

S_RST FB 的 COM_RST 参数通知 FB 启动。

将启动 OB 中的 COM_RST 参数设置为 1。

在循环操作中调用 FB，而不设置或复位 COM_RST 参数。

如果设置了 COM_RST 参数：

- FB 包含有关 ET 200S 1SI 模块的信息（I/O [不管是不是分布式 I/O] 区域中的字节数）。
- FB 将复位并终止先前启动的所有作业（在 CPU 最后一次跳转到 STOP 之前）。

FB 获得有关 ET 200S 1SI 模块的信息后，它将复位 COM_RST 参数本身。

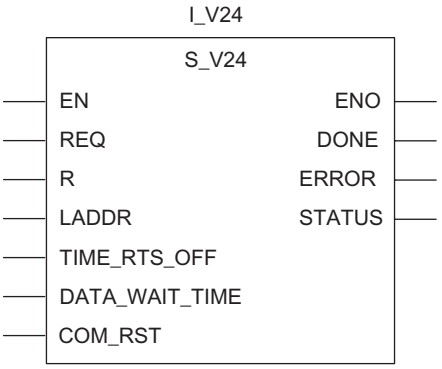
FB8 S_V24：设置 RS 232C 伴随信号的自动控制的参数

如果已针对 RS 232C 伴随信号的自动控制对模块进行了组态，则使用 S_V24 功能块可以设置其它参数（请参见 FB8 参数）。

STL 表达式

```
CALL                S_V24, I_V24
   REQ:             =
   R:               =
   LADDR:           =
   TIME_RTS_OFF:    =
   DATA_WAIT_TIME:  =
   DONE:            =
   ERROR:           =
   STATUS:          =
   COM_RST:         =
```

LAD 表达式



数据存储区中的分配

P_V24 FB 与 I_V24 背景数据块一起使用。调用时提供 DB 号。背景数据块中的数据无法访问。

说明

例外：如果出现错误 STATUS == W#16#1Exx，可以参考 SFCERR 变量以获得其它详细信息（请参见诊断）。该错误变量只能通过对背景数据块进行符号访问来装载。

FB8 参数

下表列出了 FB8 参数。

表格 2-18　　FB8：F_V24 参数

名称	类型	数据类型	说明	允许的值、备注	缺省
REQ	INPUT	BOOL	出现正跳沿时启动作业		
R	INPUT	BOOL	取消作业	取消正在进行的作业。发送受阻。	
LADDR	INPUT	INT	ET 200S 1SI 模块的起始地址	起始地址从 STEP 7 获取。	
TIME_RTS_OFF	INPUT	TIME	传输完成后禁用 RTS 之前必须等待的时间。	0 ms 到 10 min 55 s 350 ms	10 ms
DATA_WAIT_TIME	INPUT	TIME	设置 RTS 后，等待伙伴设置 CTS = ON 的时间。	0 ms 到 10 min 55 s 350 ms	10 ms
DONE[1]	OUTPUT	BOOL	作业已完成且无错	STATUS 参数 == 16#00	
ERROR[1]	OUTPUT	BOOL	作业已完成但有错	错误信息已写入 STATUS 参数。	
STATUS[1]	OUTPUT	WORD	错误规范	如果 ERROR == 1，则 STATUS 参数将包含错误信息。	
COM_RST	IN_OUT	BOOL	重新启动 FB		
[1] 在成功完成作业后的一个 CPU 周期内，DONE、ERROR 和 STATUS 参数都会用到。					

启动

S_V24 FB 的 COM_RST 参数通知 FB 启动。

将启动 OB 中的 COM_RST 参数设置为 1。

在循环操作中调用 FB，而不设置或复位 COM_RST 参数。

如果设置了 COM_RST 参数：

- FB 包含有关 ET 200S 1SI 模块的信息（I/O [不管是不是分布式 I/O] 区域中的字节数）。
- FB 将复位并终止先前启动的所有作业（在 CPU 最后一次跳转到 STOP 之前）。

FB 获得有关 ET 200S 1SI 模块的信息后，它将复位 COM_RST 参数本身。

2.10.5 读取和控制 RS 232C 辅助信号

原理

为了便于读取和控制 RS 232C 伴随信号，您可以使用 FB4 S_VSTAT 检查接口状态和 FB S_VSET 以设置/复位接口输出。

FB4 S_VSTAT：检查 ET 200S 1SI 模块的接口状态。

S_VSTAT FB 读取 ET 200S 1SI 模块的 RS 232C 伴随信号，并使用户可以在块参数中使用它们。为了进行数据传输，将在周期中静态（无条件）调用 S_VSTAT FB，或者在时间控制的程序中调用 S_VSTAT FB。

每次调用（循环轮询）该功能时，都会更新 RS 232C 伴随信号。

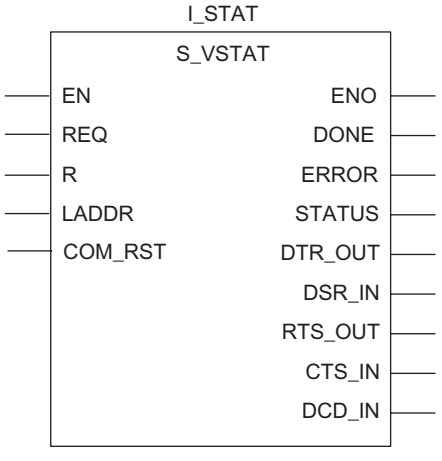
要寻址的 ET 200S 1SI 模块的地址在 LADDR 参数中指定。

FB4 调用：

STL 表达式

```
CALL          S_VSTAT, I_STAT
   REQ:        =
   R:          =
   LADDR:      =
   DONE:       =
   ERROR:      =
   STATUS:     =
   DTR_OUT:    =
   DSR_IN:     =
   RTS_OUT:    =
   CTS_IN:     =
   DCD_IN:     =
   COM_RST:    =
```

LAD 表达式



说明

参数 EN 和 ENO 仅存在于图形表达式（LAD 或 FBD）中。编译器使用二进制结果处理这些参数。

如果块终止且无错，则将二进制结果设置为信号状态“1”。如果出现错误，则将二进制结果设置为“0”。

数据存储区中的分配

S_VSTAT FB 与 I_STAT 背景数据块一起使用。调用时提供 DB 号。背景数据块中的数据无法访问。

说明

检测信号更改需要最小脉冲宽度。确定因素包括 CPU 周期时间、ET 200S 1SI 模块上的更新时间以及通讯伙伴的响应时间。

FB4 V24_STAT 参数

下表列出了 S_VSTAT (FB4) 功能块的参数。

表格 2-19　FB4：V24_STAT 参数

名称	类型	数据类型	说明	允许值、备注
REQ	INPUT	BOOL	出现正跳沿时启动作业	
R	INPUT	BOOL	取消作业	取消正在进行的作业。发送受阻。
LADDR	INPUT	INT	ET 200S 1SI 模块的起始地址	起始地址从 STEP 7 获取。
DONE[1]	OUTPUT	BOOL	指示 FB 是否已终止。	（ET 200S 1SI 输出）
ERROR[1]	OUTPUT	BOOL	作业已完成但有错	错误信息已写入 STATUS 参数。
STATUS[1]	OUTPUT	WORD	错误规范	如果 ERROR == 1，则 STATUS 参数将包含错误信息。
DTR_OUT [1]	OUTPUT	BOOL	数据终端准备就绪； ET 200S 1SI 已做好运行准备。	（ET 200S 1SI 输出）
DSR_IN [1]	OUTPUT	BOOL	数据集准备就绪；通讯伙伴已做好运行准备。	（ET 200S 1SI 输入）
RTS_OUT[1]	OUTPUT	BOOL	请求发送； ET 200S 1SI 明确发送。	（ET 200S 1SI 输出）
CTS_IN[1]	OUTPUT	BOOL	明确发送；通讯伙伴可以从 ET 200S 1SI 模块接收数据（对 ET 200S 1SI 的 RTS = ON 进行响应）。	（ET 200S 1SI 输入）
DCD_IN[1]	OUTPUT	BOOL	数据载体检测	（ET 200S 1SI 输入）
COM_RST	IN_OUT	BOOL	重新启动 FB	
[1] 在成功完成作业后的一个 CPU 周期内，这些参数都会用到。				

启动

S_VSTAT FB 的 COM_RST 参数通知 FB 启动。

将启动 OB 中的 COM_RST 参数设置为 1。

在循环操作中调用 FB，而不设置或复位 COM_RST 参数。

如果设置了 COM_RST 参数：

- FB 包含有关 ET 200S 1SI 模块的信息（I/O [不管是不是分布式 I/O] 区域中的字节数）。
- FB 将复位并终止先前启动的所有作业（在 CPU 最后一次跳转到 STOP 之前）。

FB 获得有关 ET 200S 1SI 模块的信息后，它将复位 COM_RST 参数本身。

FB5 S_VSET：设置/复位 ET 200S 1SI 模块的接口输出

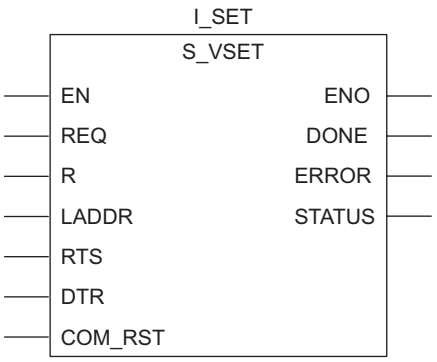
您可以使用 S_VSET FB 的相应参数输入来设置和复位接口输出。循环调用 S_VSET 功能块，或者在时间控制的程序中静态（无条件）调用该功能块。

要寻址的 ET 200S 1SI 模块的地址在 LADDR 参数中指定。

STL 表达式

```
CALL            S_VSET, I_SET
   REQ           =
   R             =
   LADDR:        =
   RTS:          =
   DTR:          =
   DONE:         =
   ERROR:        =
   STATUS:       =
   COM_RST:      =
```

LAD 表达式



说明

参数 EN 和 ENO 仅存在于图形表达式（LAD 或 FBD）中。编译器使用二进制结果处理这些参数。

如果块终止且无错，则将二进制结果设置为信号状态“1”。如果出现错误，则将二进制结果设置为“0”。

数据存储区中的分配

S_VSET FB 与 I_SEND 背景数据块一起使用。调用时提供 DB 号。背景数据块中的数据无法访问。

FB5 S_VSET 参数

下表列出了 S_VSET (FB5) 功能块的参数。

表格 2-20　FB5：S_VSET 参数

名称	类型	数据类型	说明	允许值、备注
REQ	INPUT	BOOL	出现正跳沿时启动作业	
R	INPUT	BOOL	取消作业	取消正在进行的作业。发送受阻。
LADDR	INPUT	INT	ET 200S 1SI 模块的起始地址	起始地址从 STEP 7 获取。
RTS	INPUT	BOOL	请求发送； ET 200S 1SI 明确发送。	（控制 ET 200S 1SI 输出）
DTR	INPUT	BOOL	数据终端准备就绪； ET 200S 1SI 已做好运行准备。	（控制 ET 200S 1SI 输出）
DONE[1]	OUTPUT	BOOL	指示 FB 是否已终止	（ET 200S 1SI 输出）
ERROR[1]	OUTPUT	BOOL	作业已完成但有错	错误信息已写入 STATUS 参数。
STATUS[1]	OUTPUT	WORD	错误规范	如果 ERROR == 1，则 STATUS 参数将包含错误信息。
COM_RST	IN_OUT	BOOL	重新启动 FB	
[1] 在成功完成作业后的一个 CPU 周期内，这些参数都会用到。				

启动

S_VSET FB 的 COM_RST 参数通知 FB 启动。

将启动 OB 中的 COM_RST 参数设置为 1。

在循环操作中调用 FB，而不设置或复位 COM_RST 参数。

如果设置了 COM_RST 参数：

- FB 包含有关 ET 200S 1SI 模块的信息（I/O [不管是不是分布式 I/O] 区域中的字节数）。
- FB 将复位并终止先前启动的所有作业（在 CPU 最后一次跳转到 STOP 之前）。

FB 获得有关 ET 200S 1SI 模块的信息后，它将复位 COM_RST 参数本身。

2.11 启动特性和操作模式

ET 200S 1SI 串行接口模块的操作模式

ET 200S 1SI 模块具有以下操作模式：

- **STOP**：当 ET 200S 1SI 模块处于 STOP 模式时，没有处于活动状态的协议驱动程序，并且 CPU 的所有发送和接收作业都将得到否定确认。ET 200S 1SI 模块将一直保持 STOP 模式，直到消除了导致 STOP 的原因（例如，断线或参数无效）。
- **新参数化**：为 ET 200S 1SI 模块分配新参数时，协议驱动程序将进行初始化。在进行新参数化的过程中，SF LED 变亮。

  无法进行发送和接收操作，且当驱动程序重新启动时存储在 ET 200S 1SI 模块中的发送和接收消息帧将丢失。ET 200S 1SI 模块和 CPU 之间的通讯将重新启动（当前消息帧将被取消）。

  完成新参数化后，ET 200S 1SI 模块会处于 RUN 模式并做好发送和接收准备。
- **RUN**：ET 200S 1SI 模块处理 CPU 发送作业。从通讯伙伴接收的消息帧将准备好传输到 CPU。

ET 200S 1SI 模块的启动属性

启动包含两个阶段：

- **初始化**：ET 200S 1SI 模块应用电压后，串行接口将立即进行初始化，并等待来自 CPU 的参数化数据。
- **参数化**：参数化过程中，ET 200S 1SI 模块将接收使用 STEP 7 分配给当前插槽的模块参数。

CPU 操作模式更改时 ET 200S 1SI 模块的特性

ET 200S 1SI 模块启动后，将通过功能块在 CPU 和 ET 200S 1SI 之间交换所有数据。

- **CPU STOP**：当 CPU 处于 STOP 模式时，不能通过 PROFIBUS 进行通讯。模块和 CPU 之间的任何活动数据传输（包括发送作业和接收作业）都将被取消，并重新建立连接。

  如果未使用流量控制对 ASCII 驱动程序进行参数化，那么 ET 200S 1SI 模块的 RS 232C 接口处的数据通讯将继续。换言之，仍将完成当前的发送作业。如果使用 ASCII 驱动程序，则将一直对接收消息帧进行接收，直到接收缓冲区占满。
- **CPU 启动**：在启动过程中，CPU 将参数传输到 ET 200S 1SI 模块。

  您可以通过分配适当的参数使系统在 CPU 启动时自动清空 ET 200S 1SI 的接收缓冲区。
- **CPU RUN**：当 CPU 处于 RUN 模式时，发送和接收操作不受限制。在 CPU 重新启动后的第一个 FB 周期中，ET 200S 1SI 模块和相应的 FB 保持同步。直到此过程完成后，才能执行新的 P_SEND 或 P_RCV FB。

发送消息帧时的注意事项

仅当 CPU 处于 RUN 模式时，才能发送消息帧。

如果在数据从 CPU 传输到模块的过程中 CPU 切换至 STOP 模式，P_SEND FB 将在暖启动之后输出错误 (05) 02_H。为避免此类事件的发生，用户程序可以从启动 OB 中使用 RESET 输入调用 P_SEND FB。

说明

ET 200S 1SI 模块接收到来自 CPU 的所有数据后，会将数据仅发送给通讯伙伴。

接收消息帧时的注意事项

可以使用 STEP 7 设置参数“启动时清空模块接收缓冲区 = 是/否”。

- 如果您设置了“是”参数，则当 CPU 从 STOP 模式切换至 RUN 模式时，将自动清空 ET 200S 1SI 模块接收缓冲区。
- 如果您设置了“否”参数，则指定的消息帧数将自动缓存在 ET 200S 1SI 模块的接收缓冲区中。

如果在数据从 CPU 传输到 ET 200S 1SI 模块的过程中 CPU 切换至 STOP 模式，该 FB 将在暖启动之后输出错误 (05) 02H。为避免此类事件的发生，用户程序可以从启动 OB 中使用 RESET 输入调用 P_SEND FB。在“启动时清空模块接收缓冲区 = 否”的情况下，模块会再次将消息帧传输到 CPU。

2.12 主站（而不是 S7-PROFIBUS）的参考数据

2.12.1 参考数据的基本信息

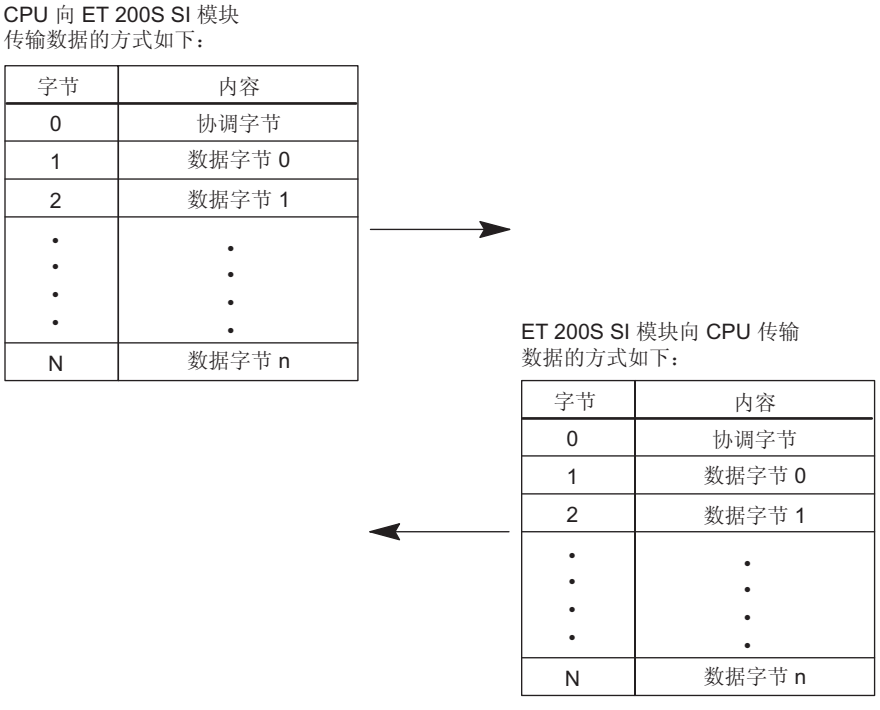
主站和 ET 200S 1SI 模块之间的数据交换

针对 4、8 或 32 字节的传输（输入或输出）对 ET 200S 1SI 模块进行组态，使总长度保持一致。ET 200S 1SI 模块使用 4、8 或 32 字节的输入/输出存储器来实现 CPU 的数据传输（通过 PROFIBUS DP 传输介质）。

CPU 可以随时通过下列方式将数据写入输入和输出以及从输入和输出中读取数据：

- CPU 在模块的输出存储器的第一个字节中将作业传输到 ET 200S 1SI 模块。
- ET 200S 1SI 模块通过将作业代码传输给输入存储器来接受作业。
- CPU 使用 3、7 或 31 字节的段（根据 I/O 大小，使用任意多个段）来交换数据，直到所有作业数据均传输完毕。

段的第一个字节是使 CPU 和 ET 200S 1SI 模块之间的段传输同步的协调字节（请参见下图）。I/O 存储器的其余字节均包含作业数据。



n = 3、7 或 31，具体取决于在组态中选择的模块版本

图片 2-23 CPU 和 ET 200S 1SI 模块之间的数据交换

协调字节的说明

下表介绍了协调字节（字节 0）的内容，这些字节使 CPU 和 ET 200S 1SI 串行接口模块之间的数据传输同步。

表格 2-21　用于数据传输的协调字节 0 的内容

字节段	说明
由 CPU 写入的作业字节	Bit 7 6 5 4 3 2 1 0 Res. | Job code | Error | Execution number
位 7	保留。
作业代码	由 CPU 设置，以启动作业。
执行编号	**发送作业：**当 CPU 将另一个段发送至 ET 200S 1SI 模块时，CPU 将编号增加 1 *... 或者* **接收作业：**每次 CPU 按正确顺序从接口模块接收到新段时，CPU 的输入字节 0 均接受。当设置了错误位时，指示上一个有效的执行编号。（值范围：1 到 7）。
错误	由 CPU 设置，以指示某个段未按正确顺序接收。执行编号域指示上一个有效的执行编号。
由 ET 200S 1SI 模块写入的作业字节	位 7 6 5 4 3 2 1 0 资源 | 作业代码 | 错误 | 执行编号
位 7	保留。
作业代码	由 ET 200S 1SI 模块接受，以确认已接受该作业。
执行编号	**发送作业：**每次模块按正确顺序从 CPU 接收到新段时，由模块的输出字节 0 接受。当设置了错误位时，指示上一个有效的执行编号。 **接收作业：**当模块将另一个段发送至 CPU 时，模块将编号加 1。 （值范围：1 到 7）
错误	发送器监视分段事务接收器的错误位。如果设置了错误位： • **CPU = 发送器**（发送作业）：CPU 从接收器报告的编号后的下一段开始重新发送段。 • **模块 = 发送器**（接收作业）：1SI 模块取消向用户进一步传输 Rx 消息帧，并且状态字中的错误消息为 0x0551。模块将等待此错误消息被确认（空闲）。活动的错误序列完成后，取消的 Rx 消息帧将再次报告给用户，或者用于传输。

作业代码定义

下表根据位 4 至位 6 在协调字节 0 中的分配方式列出了各作业。

表格 2-22　作业代码

位 6 5 4	十六进制值	定义
0 0 0	0_H	空闲状态
0 0 1	1_H	发送
0 1 0	2_H	接收
0 1 1	3_H	读取 V.24 信号状态
1 0 0	4_H	写入 V.24 信号
1 0 1	5_H	传输参数：通过此作业，您可以设置未在 GSD 文件中指定的其它参数。
1 1 0	6_H	保留
1 1 1	7_H	作业结束确认

写入作业代码的规则

以下规则适用于在协调字节中写入作业代码以便 CPU 和 ET 200S 1SI 模块可以使数据传输同步的情况：

- 必须从 ET 200S 1SI 模块的输入协调字节中读到空闲代码，CPU 用户程序才能将作业代码写入输出协调字节。
- 然后，必须在模块的输入协调字节中读到作业确认代码（即，接受的作业代码），CPU 用户程序才能将第一段写入输出字节 1 至 n。
- 如果用户程序读到任意作业确认代码（而不是程序发送的代码），它会等到再次从 ET 200S 1SI 模块的输入协调字节中读到空闲代码时才写入输出字节 0 至 n。

  例如，如果在同一个周期中执行两个单独的作业，这两个作业均读到空闲代码，并分别将不同的作业代码写入输出字节，则会出现上述情况。由于 CPU 周期和 PROFIBUS DP 周期不同步，因此不能确定哪个作业先到达模块。因此，每个作业都必须等待另一个作业结束后才能得以处理。

状态字定义

在后续页面上显示的数据传输示例中，ET 200S 1SI 模块在对 CPU 的某些响应中使用状态消息的字节 1 和 2。标题为“STATUS 参数中的诊断消息”的表中列出了状态字及其定义。

字中的字节顺序

在所有 16 位字的情况下（例如，状态和长度），在 CPU 和 ET 200S 1SI 模块之间传输数据时，将先发送最重要的字节。

1SI 模块的接收状态

当模块处于空闲状态时（作业确认字节 0 = 00_H），始终对用户显示 1SI 模块的接收缓冲区状态。然后，状态将被存储在字节 1 和 2 中。

状态	含义
0000H	不存在接收的消息
0001H	存在接收的消息/接收消息帧
0B01H	接收缓冲区的 2/3 以上区域已满。

2.12.2 将数据从 CPU 发送到模块的示例

示例

下表显示了 CPU 发送包含字母表中前 22 个字符的消息的示例。I/O 存储器有 8 个字节。DP 周期与 CPU 周期大致相同，因此当模块与执行编号相对应时，将存在一个周期的延迟。

表格 2-23　发送示例

CPU 周期	CPU 写入 ET 200S 1SI	CPU 从 ET 200S 1SI 中读取
1.	用户程序读到以下模块空闲代码： 字节 0 1 2 3 4 5 6 7 ← 00_H | $nnnn_H$ | xx_H | xx_H | xx_H | xx_H | xx_H 作业确认 | 状态 | 不相关	
	CPU 写入发送作业： Byte 0 1 2 3 4 5 6 7 10_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H → Job | Irrelevant	
2.	用户程序仍读取模块空闲代码： ← 00_H | $nnnn_H$ | xx_H | xx_H | xx_H | xx_H | xx_H 作业确认 | 状态 | 不相关	
	CPU 重复发送作业： 10_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H → 请求 | 不相关	
3.	用户程序从模块读取以下响应： ← 10_H | $nnnn_H$ | xx_H | xx_H | xx_H | xx_H | xx_H 作业确认 | 状态 | 不相关	
	CPU 发送第一段： 11_H | 0016_H | ‘a’ | ‘b’ | ‘c’ | ‘d’ | ‘e’ → 请求 | 发送长度 | 数据	
4.	用户程序从模块读取以下响应： ← 10_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H 作业确认 | 不相关	
	CPU 重复第一段： 12_H | ‘f’ | ‘g’ | ‘h’ | ‘i’ | ‘j’ | ‘k’ | ‘l’ → 请求 | 数据	

CPU 周期	CPU 写入 ET 200S 1SI	CPU 从 ET 200S 1SI 中读取
5.	用户程序从模块读取以下响应： ← 11_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H 作业确认 | 不相关 由于没有指示出错且执行正确，CPU 将发送第二段： 12_H | 'm' | 'n' | 'o' | 'p' | 'q' | 'r' | 's' 请求 | 数据 →	
6.	用户程序从模块读取以下响应： ← 12_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H 作业确认 | 不相关 由于没有指示出错且执行正确，CPU 将发送第三段： 13_H | 't' | 'u' | 'v' | xx_H | xx_H | xx_H | xx_H Job | Data | Irrelevant →	
7.	用户程序从模块读取以下响应： ← 13_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H Job ackn. | Irrelevant 由于没有指示出错且执行正确，CPU 将发送第四段： 14_H | 't' | 'u' | 'v' | xx_H | xx_H | xx_H | xx_H 请求 | 数据 | 不相关 →	
8.	用户程序从模块读取以下响应： ← 13_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H 作业确认 | 不相关 CPU 等待对第四段的确认： 14_H | 't' | 'u' | 'v' | xx_H | xx_H | xx_H | xx_H 请求 | 数据 | 不相关 →	
9.	用户程序从模块读取以下响应： ← 14_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H Job ackn. | Irrelevant CPU 不发送任何新内容（输出保持相同）并等待最后一个模块确认；表明消息已发送到通讯伙伴： 14_H | 't' | 'u' | 'v' | xx_H | xx_H | xx_H | xx_H 请求 | 数据 | 不相关 →	
n.	几个 CPU 周期过后，用户程序将从模块读到以下响应： ← 74_H | $nnnn_H$ | xx_H | xx_H | xx_H | xx_H | xx_H 作业确认 | 状态 | 不相关	
—	CPU 将空闲代码写入作业并终止作业。	

2.12.3 从模块接收至 CPU 的数据示例

示例

下表显示了 CPU 如何从串行接口模块接收消息的示例。I/O 存储器有 8 个字节。DP 周期比 CPU 周期短。这意味着模块中不存在等待时间。

表格 2-24 接收示例

<table>
<tr><th>CPU 周期</th><th>CPU 写入 ET 200S 1SI</th><th>CPU 从 ET 200S 1SI 中读取</th></tr>
<tr><td>n</td><td colspan="2">用户程序在多个周期内读取模块空闲代码，直至状态指示收到的信息可用为止：
Byte 0 1 2 3 4 5 6 7
← | 00_H | $nnnn_H$ | xx_H | xx_H | xx_H | xx_H | xx_H
Job ackn. | Status | Irrelevant
状态：
0000_H = 收到的信息不可用。
0001_H = 收到的信息可用。
$0B01_H$ = 接收缓冲区占用超出整体的 2/3。</td></tr>
<tr><td></td><td colspan="2">CPU 写入接收作业：
字节 0 1 2 3 4 5 6 7
10_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H →
请求 | 不相关</td></tr>
<tr><td>下一个周期 (n + 1)</td><td colspan="2">用户程序从模块中读取以下响应（模块确认收到，对第一个区段进行响应，并增加执行号）：
← | 21_H | 0006_H | ‘a’ | ‘b’ | ‘c’ | ‘d’ | ‘e’
作业确认 | 长度 | 数据</td></tr>
<tr><td></td><td colspan="2">CPU 写入作业以确认第一个区段：
21_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H →
请求 | 不相关</td></tr>
<tr><td>下一个周期 (n + 2)</td><td colspan="2">用户程序读取模块的第二个区段：
← | 22_H | ‘f’ | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H
Job ackn. | Data | Irrelevant</td></tr>
<tr><td></td><td colspan="2">CPU 写入作业以确认第二个区段：
22_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H →
Job | Irrelevant</td></tr>
<tr><td>下一个周期 (n + 3)</td><td colspan="2">第一个接收事务终止后模块返回空闲状态。
← | 00_H | $nnnn_H$ | xx_H | xx_H | xx_H | xx_H | xx_H
作业确认 | StatusUS | 不相关</td></tr>
<tr><td>—</td><td colspan="2">CPU 终止作业。</td></tr>
</table>

2.12.4 读取 V.24 信号状态的示例

示例

下表显示了 CPU 如何从串行接口模块读取 V.24 信号状态的示例。I/O 存储器有 8 个字节。

表格 2-25 读取 V.24 信号状态的示例

CPU 周期	CPU 写入 ET 200S 1SI	CPU 从 ET 200S 1SI 中读取
1.	用户程序读取模块空闲代码：	
	CPU 写入作业以读取 V.24 信号状态：	
2.	用户程序从模块读取以下响应：	
	CPU 写入确认并接受执行号。	
3.	第一个事务终止后模块返回空闲状态。	
—	CPU 终止作业。	

字节 0 1 2 3 4 5 6 7

← 00_H	$nnnn_H$	xx_H	xx_H	xx_H	xx_H	xx_H	
作业确认	状态	不相关					

Byte 0 1 2 3 4 5 6 7

30_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H	→
Job	Irrelevant							

← 31_H	$nnnn_H$	xx_H	xx_H	xx_H	xx_H	xx_H		
作业确认	信号	不相关						

MSB							LSB
00	0	0	0	DCD	CTS	RTS	DSR	DTR
	7	6	5	4	3	2	1	0

31_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H	→
请求	不相关							

← 00_H	$nnnn_H$	xx_H	xx_H	xx_H	xx_H	xx_H	
Job ackn.	Status	Irrelevant					

2.12.5 写入 V.24 信号示例

写入 V.24 信号示例

下表显示了 CPU 如何将 V.24 信号写入串行接口模块的示例。I/O 存储器有 8 个字节。

表格 2-26 写入 V.24 信号示例

CPU 周期	CPU 写入 ET 200S 1SI	CPU 从 ET 200S 1SI 中读取
1.	用户程序读取模块空闲代码： Byte 0 1 2 3 4 5 6 7 ← 00_H | $nnnn_H$ | xx_H | xx_H | xx_H | xx_H | xx_H Job ackn. | Status | Irrelevant	
	CPU 写入作业以写入 V.24 信号： Byte 0 1 2 3 4 5 6 7 40_H | $nnnn_H$ | xx_H | xx_H | xx_H | xx_H | xx_H → Job | Signal states | Irrelevant MSB LSB 00 | 0 | 0 | 0 | DCD | CTS | RTS | DSR | DTR 7 6 5 4 3 2 1 0	
2.	用户程序从模块读取以下响应： ← 40_H | $nnnn_H$ | xx_H | xx_H | xx_H | xx_H | xx_H 作业确认 | 状态 | 不相关	
	CPU 将空闲状态写入作业字节： 00_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H → 请求 | 不相关	
3.	用户程序从模块中读取以下响应（模块在事务结束时返回空闲状态）： ← 00_H | $nnnn_H$ | xx_H | xx_H | xx_H | xx_H | xx_H 作业确认 | 状态 | 不相关	
—	CPU 将空闲代码写入作业并终止作业。	

2.12.6 数据流量控制的参数

数据流量控制的参数

用于通过 ASCII 驱动程序传输参数的作业代码使您可以设置其它参数。

这取决于 GSD 文件中所选的数据流量控制类型。下表对三种数据流量控制类型进行了说明。

表格 2-27 数据流量控制的参数

使用 XON/XOFF 进行数据流量控制的参数帧			
字节	**说明**	**值范围**	**缺省值**
1	参数块编号	20_H	
2 和 3	长度	0004_H	0004_H
4	XON 字符	0 至 127（7 个数据位） 0 至 255（8 个数据位）	11 (DC1)
5	XOFF 字符	0 至 127（7 个数据位） 0 至 255（8 个数据位）	13 (DC3)
6 和 7	XOFF 后 XON 的等待时间	20 至 655,350，增量为 10 ms	200 (2000 ms)
使用 RTS/CTS 进行数据流量控制的参数帧			
字节	**说明**	**值范围**	**缺省值**
1	参数块编号	21_H	
2 和 3	长度	0002_H	0002_H
4 和 5	CTS = ON 的等待时间	20 至 655,350，增量为 10 ms	200 (2000 ms)
RS 232C 伴随信号的自动控制参数帧			
字节	**说明**	**值范围**	**缺省值**
1	参数块编号	22_H	
2 和 3	长度	0004_H	0004_H
4 和 5	传输后 RTS = OFF 的时间	0 至 655,350，增量为 10 ms	1 (10 ms)
6 和 7	RTS = ON 后 CTS = ON 的等待时间	0 至 655,350，增量为 10 ms	1 (10 ms)

XON/XOFF 的示例

下表显示了 CPU 如何设置 XON/XOFF 参数的示例。I/O 存储器有 4 个字节。

表格 2-28　XON/XOFF 的示例

CPU 周期	CPU 写入 ET 200S 1SI	CPU 从 ET 200S 1SI 中读取
1.	用户程序发现以下模块空闲代码：→	Byte 0 / 1 / 2 / 3 00_H | $nnnn_H$ | xx_H Job ackn. | Status | Irrelevant
	Byte 0 / 1 / 2 / 3 50_H | xx_H | xx_H | xx_H Job | Irrelevant	← 作业：发送参数代码（1 0 1 或 5_H）和执行号 0
2.	用户程序从模块中发现以下响应：→	50_H | xx_H | xx_H | xx_H 作业确认 | 不相关
	由于作业已被接受，因此 CPU 将发送第一个区段。	
	51_H | 20_H | 0004_H Job | Data flow | Send length	← 作业：继续参数并增加执行号 ← 数据流量：数据流量参数的代码
3.	用户程序从模块中发现以下响应：→	51_H | xx_H | xx_H | xx_H 作业确认 | 不相关
	由于尚未指出任何错误，因此 CPU 将发送第二个区段：	
	52_H | $0B_H$ | $0D_H$ | 00_H 请求 | DC1 | DC3 | XOFF — LSB 后，XON 的等待时间	
4.	用户程序从模块中发现以下响应：→	52_H | xx_H | xx_H | xx_H 作业确认 | 不相关
	由于尚未指出任何错误，因此 CPU 将发送第三个区段：	
	53_H | $C8_H$ | xx_H | xx_H Job | Waiting time for XON to XOFF - LSB | Irrelevant	

<table>
<tr><th>CPU 周期</th><th>CPU 写入 ET 200S 1SI</th><th>CPU 从 ET 200S 1SI 中读取</th></tr>
<tr><td rowspan="4">5.</td><td>用户程序从模块中发现以下响应：→</td><td><table><tr><td>53_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td></tr><tr><td>Job ackn.</td><td colspan="3">Irrelevant</td></tr></table></td></tr>
<tr><td colspan="2">CPU 将重复第三个区段并等待作业结束确认。</td></tr>
<tr><td><table><tr><td>53_H</td><td>$C8_H$</td><td>xx_H</td><td>xx_H</td></tr><tr><td>请求</td><td>XOFF — LSB 后，XON 的等待时间</td><td colspan="2">不相关</td></tr></table></td><td></td></tr>
<tr><td rowspan="4">6.</td><td>用户程序从模块中发现以下响应：→</td><td><table><tr><td>73_H</td><td>$nnnn_H$</td><td>xx_H</td></tr><tr><td>Job ackn.</td><td>Status</td><td>Irrelevant</td></tr></table></td></tr>
<tr><td colspan="2">CPU 将空闲代码写入作业并终止作业。</td></tr>
<tr><td><table><tr><td>00_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td></tr><tr><td>请求</td><td colspan="3">不相关</td></tr></table></td><td></td></tr>
</table>

2.12.7 错误处理

错误情况

当出现以下情况时，串行接口模块将发出错误：

- 如果发送作业长度大于 224 个字节，则模块响应作业结束确认，且状态字包含错误代码。然后，CPU 将空闲代码写入作业并终止作业。
- 如果接收作业已发送至模块，但收到的消息包含错误，则模块会接受执行号为零的接收作业代码，且状态字包含错误代码。然后，CPU 将空闲代码写入作业并终止作业。
- 如果接收作业已发送至模块，但收到的消息不可用，则该模块会接受执行号为零的接收作业代码，且状态字包含值 0101_H。这不算错误情况，但它的确防止了模块在接收作业模式下被取消激活，也防止了模块等待接收消息。结果，发送作业仍然可以执行。CPU 将空闲代码写入作业并终止作业。

例外

前面提到过，个别操作（例如发送作业）在模块处于空闲状态之前不能在用户程序中启动。作业发送后，该操作必须等到模块接受作业代码后才能执行涉及的操作。如果启动时操作带有分段，则可能出现以下例外：

说明

在以下发送或参数化操作的说明中，发送器为 CPU，接收器为串行接口模块。如果是接收操作，则发送器为串行接口模块，接收器为 CPU。

- **错误：**发送器将监视分段事务接收器的错误位。如果设置了错误位，则发送器从接收器报告的编号后的下一个区段开始再次发送这些区段。
- **执行号的顺序不正确：**在分段操作期间，如果接收器收到的区段的执行号不等于上一个执行号加 1，则它必须报告错误，并响应上一个执行号。
- **修改的作业代码：**
  - 如果接收器收到的区段的作业代码与分段操作开头的代码不一样，并且不是 000 或 111，则接收器将忽略其它代码并丢弃该关联数据。
  - 如果接收器在分段操作期间收到带有空闲状态作业代码的区段，则会取消该操作并采用空闲状态，而不设置错误位。
  - 如果接收器在分段操作期间收到带有作业结束确认的区段，则会取消该操作并采用空闲状态，而不设置错误位。
  - 在分段操作期间，如果发送器收到的响应带有另一个作业代码，则必须取消此消息。然后再次发送空闲代码，模块必须进入空闲状态，并且必须再次执行此操作。

2.13 诊断

概述

ET 200S 1SI 模块的诊断功能使您可以快速发现错误，即使是在操作期间。可以使用以下诊断选项：

- 通过 ET 200S 1SI 模块前面板上的状态 LED 进行诊断
- 通过功能块的 STATUS 输出进行诊断
- 通过 PROFIBUS 从站诊断进行诊断

使用状态 LED 的诊断信息

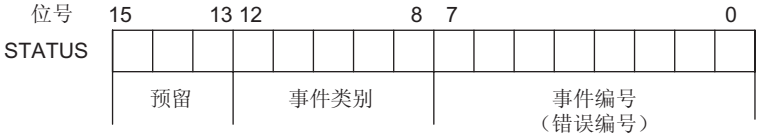
以下状态 LED 位于 ET 200S 接口模块的前面板上：

- **TX**（绿色）：当模块通过接口发送数据时点亮。
- **RX**（绿色）：当模块通过接口接收数据时点亮。
- **SF**（红色）：指示可能出现以下某种错误/故障：
  - 硬件故障
  - 参数分配错误
  - 模块和通讯伙伴之间发生断线或电缆松动：

    如果接收线路的初始状态设置为 R(A) 5 V/R(B) 0 V，则仅当具有 RS 422 接口连接时可以检测到。
  - 通讯错误（奇偶校验错误、帧错误、缓冲区溢出）

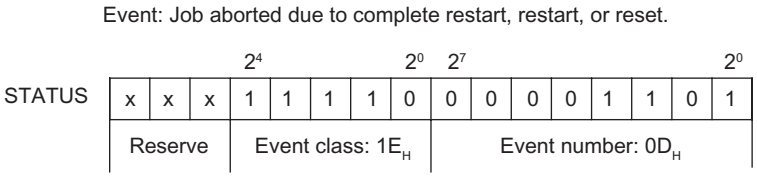
功能块诊断消息的结构

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。下图说明了 STATUS 参数的结构。



图片 2-24　　STATUS 参数的结构

示例：下图说明了“作业因重新启动、暖启动或复位而取消”事件的 STATUS 参数的内容（事件类别 $1E_H$，事件编号 $0D_H$）。



图片 2-25　　示例：“事件类别 1EH，事件 0DH”的 STATUS 参数

功能块诊断消息

下表说明了事件类别、事件编号定义和每种错误情况的建议解决方法。

表格 2-29　　STATUS 参数中的诊断消息

事件编号	事件	补救措施
事件类别 2 ($0x02_H$)：“初始化错误”		
(02) 01_H	未进行（有效的）参数化	为模块分配有效的参数。如有必要，请检查系统的安装是否正确。
事件类别 5 (05_H)：“处理 CPU 作业时出错”		
(05) 02_H	ET 200S 1SI 模块在该操作模式下不允许的作业（例如未参数化的设备接口）。	发送消息帧的长度大于 224 个字节。ET 200S 1SI 模块已取消了发送作业。 选择较短的消息帧长度。
(05) $0E_H$	消息帧长度无效	消息帧的长度大于 224 个字节。ET 200S 1SI 模块仍在接收消息帧（大于 224 个字节）的其余部分；这意味着消息帧的第一部分被丢弃。 选择较短的帧长度。
(05) 50_H	参数更新作业对当前所选的 ET 200S 1SI 模块数据流量控制的形式无效。	更改 AS 程序中功能块（FB6 S_XON、FB7 S_RTS、FB8 S_V24）的参数，或者更改硬件组态中 ET 200S 1SI 模块的数据流量控制形式，以便其进行响应。
(05) 51_H	在 ET 200S 1SI 模块和自动化系统之间进行通讯期间出现帧执行错误。从 ET 200S 1SI 模式传送收到的消息帧期间，自动化系统中出错。	模块和自动化系统已取消了该传输。重复接收作业；ET 200S 1SI 模块再次发送收到的消息。

事件编号	事件	补救措施
事件类别 7 (07H)：“发送错误”		
(07) 02H	仅适用于 3964(R)： 建立连接时出错： 发送 STX 之后，接收到 NAK 或任何其它代码（DLE 或 STX 除外）。	检查伙伴设备是否发生故障；您可能需要使用转换为传输线路的接口测试设备 (FOXPG)。
(07) 03H	仅适用于 3964(R)： 超过确认延迟时间 (QVZ)： 发送 STX 之后，伙伴在确认延迟时间内没有响应。	伙伴设备过慢或尚未准备好接收，或者例如出现传输线路断路的情况。检查伙伴设备是否发生故障；您可能需要使用转换为传输线路的接口测试设备 (FOXPG)。
(07) 04H	仅适用于 3964(R)： 被伙伴取消： 在发送期间接收到来自伙伴的一个或多个字符。	检查伙伴是否也指出错误，可能的原因包括并未收到所有发送的数据（例如传输线路断路）、致命错误未解决或伙伴设备发生故障。检查伙伴设备是否发生故障；您可能需要使用转换为传输线路的接口测试设备 (FOXPG)。
(07) 05H	仅适用于 3964(R)： 发送时出现否定确认	检查伙伴是否也指出错误，可能的原因包括并未收到所有已发送的数据（例如传输线路断路）、致命错误未解决或伙伴设备发生故障。检查伙伴设备是否发生故障；您可能需要使用转换为传输线路的接口测试设备 (FOXPG)。
(07) 06H	仅适用于 3964(R)： 连接结束错误： • 伙伴在连接结束时以 NAK 或一个随机字符串（DLE 除外）拒绝了消息帧，或 • 过早收到确认字符 (DLE)。	检查伙伴是否也指出错误，可能的原因包括并未收到所有已发送的数据（例如传输线路断路）、致命错误未解决或伙伴设备发生故障。检查伙伴设备是否发生故障；您可能需要使用转换为传输线路的接口测试设备 (FOXPG)。
(07) 07H	仅适用于 3964(R)： 连接结束时超过确认延迟时间，或发送消息帧之后超过响应监视时间： 用 DLE ETX 终止连接后，在确认延迟时间内没有收到伙伴的响应。	伙伴设备过慢或发生故障。如有必要，请使用转换为传输线路的接口测试设备进行检查。
(07) 08H	仅适用于 ASCII 驱动程序： XON 或 CTS = ON 的等待时间已结束。	通讯伙伴发生故障、过慢或已离线。检查通讯伙伴；您可能需要更改参数分配。
(07) 0BH	仅适用于 3964(R)： 由于两个伙伴均为高优先级，因此无法解决初始化冲突。	更改参数分配。
(07) 0CH	仅适用于 3964(R)： 由于两个伙伴均为低优先级，因此无法解决初始化冲突。	更改参数分配。

事件编号	事件	补救措施
事件类别 8 (08_H)：“接收错误”		
(08) 02_H	仅适用于 3964(R)： 建立连接时出错： • 在空闲模式下，收到了一个或多个随机代码（NAK 或 STX 除外），或 • 收到 STX 后，伙伴在没有等待响应 DLE 的情况下发送了更多字符。 伙伴通电后： • 伙伴接通电源时，模块收到一个未定义的字符。	检查伙伴设备是否发生故障；您可能需要使用转换为传输线路的接口测试设备 (FOXPG)。
(08) 05_H	仅适用于 3964(R)： 接收时出现逻辑错误： 收到 DLE 后，又收到一个随机代码（DLE 或 ETX 除外）。	检查伙伴是否总是复制消息帧头中的 DLE 和数据字符串，或连接是否通过 DLE ETX 断开。检查伙伴设备是否发生故障；您可能需要使用转换为传输线路的接口测试设备 (FOXPG)。
(08) 06_H	超过字符延迟时间 (ZVZ)： • 在字符延迟时间内未收到两个连续字符，或 仅适用于 3964(R)： • 在字符延迟时间内，未收到在建立连接时发送 DLE 后的第一个字符。	伙伴设备过慢或发生故障。检查伙伴设备是否发生故障；您可能需要使用转换为传输线路的接口测试设备 (FOXPG)。
(08) 07_H	仅适用于 3964(R)： 消息帧长度非法： 收到一个长度为零的消息帧。	收到长度为零的消息帧不会引发错误。 检查通讯伙伴为何发送不含用户数据的消息帧。
(08) 08_H	仅适用于 3964(R)： 块检查字符 (BCC) 中出错： 内部计算的 BCC 的值与在终止连接时伙伴接收到的 BCC 不匹配。	检查连接是否被严重破坏；此时也可以不时地查看错误代码。检查伙伴设备是否发生故障；您可能需要使用转换为传输线路的接口测试设备 (FOXPG)。
(08) 09_H	仅适用于 3964(R)： 重复次数必须设置为相同的值。	在通讯伙伴方声明一个与您模块中相同的块等待时间。检查通讯设备是否发生故障；您可能需要使用转换为传输线路的接口测试设备。
(08) $0A_H$	没有可用的空闲接收缓冲区： 没有可用于接收数据的接收缓冲区空间。	必须更频繁地调用 S_RCV FB。
(08) $0C_H$	传输错误： • 检测到传输错误（奇偶校验/停止位/溢出错误）。 仅适用于 3964(R)： • 如果在发送或接收操作过程中发生此错误，则启动重复。 • 如果在空闲模式下收到一个损坏的字符，则会立即报告错误，以便可以及早检测到传输线路上的干扰。 • 如果 SF LED（红灯）点亮，则两个通讯伙伴之间的连接电缆断路。	传输线路上的干扰造成消息帧重复，因此降低了用户数据的吞吐量。漏检错误的风险增加。更改系统设置或电缆接线。检查通讯伙伴的连接电缆，或检查双方设备对波特率、奇偶校验和停止位数的设置是否相同。

事件编号	事件	补救措施
(08) $0D_H$	BREAK：到伙伴的接收线路断路。	重新连接或接通伙伴电源。
(08) 10_H	仅适用于 ASCII 驱动程序： 奇偶校验错误： • 如果 SF LED（红灯）亮起，则两个通讯伙伴之间的连接电缆断路。	检查通讯伙伴的连接电缆，或检查双方设备对波特率、奇偶校验和停止位数的设置是否相同。 更改系统设置或电缆接线。
(08) 11_H	仅适用于 ASCII 驱动程序： 字符帧错误： • 如果 SF LED（红灯）亮起，则两个通讯伙伴之间的连接电缆断路。	检查通讯伙伴的连接电缆，或检查双方设备对波特率、奇偶校验和停止位数的设置是否相同。 更改系统设置或电缆接线。
(08) 12_H	仅适用于 ASCII 驱动程序： 在模块发送 XOFF 或将 CTS 设置为 OFF 后收到更多字符。	重新组态通讯伙伴或更快速地读取模块数据。
(08) 18_H	仅适用于 ASCII 驱动程序： DSR = OFF 或 CTS = OFF	在传输之前或传输期间，伙伴将 DSR 或 CTS 信号切换为“OFF”。 检查伙伴对 RS 232C 伴随信号的控制。
(08) 50_H	接收消息帧的长度大于 224 个字节或定义的消息帧长度。	调整伙伴的消息帧长度
事件类别 11 ($0B_H$)：警告		
(0B) 01_H	接收缓冲区占用超出整体的 2/3	
事件类别 30 ($1E_H$)：“模块与 CPU 之间发生通讯错误”		
(1E) $0D_H$	“作业因重新启动、暖启动或复位而取消”	
(1E) $0E_H$	调用 SFC DP_RDDAT 期间出现静态错误。SFC 的 RET_VAL 返回值可用于背景数据块中 SFCERR 变量的判断。	从背景数据块中装载 SFCERR 变量。
(1E) $0F_H$	调用 SFC DP_WRDAT 期间出现静态错误。SFC 的 RET_VAL 返回值可用于背景数据块中 SFCERR 变量的判断。	从背景数据块中装载 SFCERR 变量。
(1E) 10_H	调用 SFC RD_LGADR 期间出现静态错误。SFC 的 RET_VAL 返回值可用于背景数据块中 SFCERR 变量的判断。	从背景数据块中装载 SFCERR 变量。
(1E) 11_H	调用 SFC RDSYSST 期间出现静态错误。SFC 的 RET_VAL 返回值可用于背景数据块中 SFCERR 变量的判断。	从背景数据块中装载 SFCERR 变量。
(1E) 20_H	参数超出范围。	更改功能块的输入以使其位于有效的范围内。
(1E) 41_H	在 FB 的 LEN 参数中指定的字节数未获允许。	值必须在 1 至 200 个字节的范围内。

判断 SFCERR 变量

有关属于事件类别 30 的错误 (1E) $0E_H$、(1E) $0F_H$、(1E) 10_H 和 (1E) 11_H 的更多详细信息，可以通过 SFCERR 变量获得。

您可以从相应功能块的背景数据块中装载 SFCERR 变量。

*《用于 S7-300/400 系统和标准功能的系统软件》*参考手册中有关“DPRD_DAR”和 SFC15 “DPWR_DAT”系统功能的章节中说明了在 SFCERR 变量中输入的错误消息。

PROFIBUS 从站诊断

从站诊断数据符合“EN 50170，卷 2，PROFIBUS”的要求。根据 DP 主站，符合该标准的所有 DP 从站的诊断数据都可以用 STEP 5 或 STEP 7 读取。

PROFIBUS 从站诊断包括模块诊断、模块状态诊断和通道特定的诊断。有关 DP 从站诊断的详细信息可以在名为*《ET 200S 分布式 I/O 设备》*手册中找到。

通道特定的诊断：通道特定的诊断可以在模块状态后提供模块和启动中有关通道错误的信息。下表列出了通道特定错误的类型。

表格 2-30　ET 200S 1SI 串行接口模块的通道错误类型

故障类型	含义	补救措施
00110：断线	发生断线或断开连接。	检查端子的接线。检查连接至伙伴的电缆。
00111：上溢	缓冲区上溢；消息长度上溢	必须更频繁地调用 S_RCV FB。
01000：下溢	仅适用于 3964(R)：长度为 0 的消息已发送	检查通讯伙伴为何发送不含用户数据的消息帧。
01001：错误	出现内部模块错误。	更换模块。
10000：参数分配错误	模块未参数化。	纠正参数化。
10110：消息错误	帧错误、奇偶校验错误	检查通讯设置。

2.14 技术规范

协议和接口的技术数据

表格 2-31 ET 200S 1SI 模块的常规技术数据

常规技术数据	
指示器元素：	• LED（绿色）：TX（传输） • LED（绿色）：RX（接收） • LED（红色）：SF（组故障）
协议驱动程序已提供	3964(R) 驱动程序 ASCII 驱动程序
3964(R) 协议的波特率 ASCII 驱动程序的波特率	110, 300, 600, 1,200, 2,400, 4,800, 9,600, 19,200, 38,400, 57,600, 76,800, 115,200
字符帧（10 或 11 位）	每个字符的位数：7 或 8 启动/停止位数：1 或 2 奇偶校验：无、偶数、奇数、任意
标准块 (FB) 的存储器需求	发送和接收：约 4,300 个字节
RS 232C 接口的技术数据	
接口	RS 232C，8 个端子
RS 232C 信号	TXD、RXD、RTS、CTS、DTR、DSR、DCD、PE 全部与 ET 200S 1SI 模块的内部电源电隔离。
最大传输距离	15 m
RS 422/485 接口的技术数据	
接口	• RS 422，5 个端子 • RS 485，3 个端子
RS 422 信号 RS 485 信号	TXD (A)-、RXD (A)-、TXD (B)+、RXD (B)+、PE R/T (A)-、R/T (B)+、PE 全部与 ET 200S 1SI 模块的内部电源电隔离。
最大传输距离	1,200 m

技术数据

尺寸和重量	
尺寸：宽度 x 高度 x 深度（单位：mm）	15 x 81 x 52
重量	约 50 g
模块特定的数据	
RS 232C	
• 输入数	4
• 输出数	3
RS 422	
• 输入对数	1
• 输出对数	1
RS 485	
I/O 对数	1
电缆长度	
• 屏蔽 (RS 232C)	最大 15 m
• 屏蔽 (RS 422/485)	最大 1,200 m
防护等级 [1]	IEC 801-5
电压、电流、电位	
电子装置的额定电源电压 (L+)	24 V DC
反极性保护	是
电位隔离	
• 通道和背板总线之间	是
• 通道和电子装置的电源之间	是
• 通道之间	否
• 通道和 PROFIBUS DP 之间	是
绝缘测试位置	
• 通道到背板总线及通道到负载电压 L+	500 V DC
• 负载电压 L+ 到背板总线	500 V AC
电流源	
• 来自背板总线	最大 10 mA
• 来自电源 L+	最大 120 mA， 通常为 50 mA
模块的功率损耗	通常为 1.2 W

状态、中断、诊断	
状态指示器	• 绿色 LED (TX) • 绿色 LED (RX)
诊断功能 • 组故障显示 • 可以显示诊断信息	可能为红色 LED (SF)
输出	
输出，RS 232C 范围	± 最大 10 V
• 适用于容性负载	最大 2500 pF
• 短路保护	是
• 短路电流	约 60 mA
• PE（接地）输出或输入的电压	最大 25 V
输出，RS 422/485	
负载电阻	最小 50 kΩ
• 短路保护	是
• 短路电流	约 60 mA

[1] 用户提供的输入线路链接中所需的外部保护设备：

- Blitzductor 标准安装导轨适配器
- Blitzductor 保护模块 KT AD-24V

Modbus/USS

3

3.1 产品概述

订货号

6ES7 138-4DF11-0AB0

产品说明

ET 200S Modbus/USS 串行接口模块是 ET 200S 产品系列的插件模块，它基于三个硬件接口（RS 232C、RS 422 和 RS 485）和两个软件协议提供了对串行通讯的访问。

- Modbus
- USS 主站

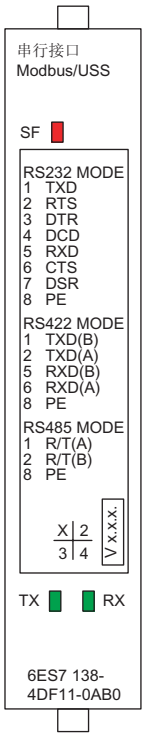
可以使用 ET 200S Modbus/USS 串行接口模块以通过点对点连接在自动化系统或计算机之间进行数据交换。所有通讯均基于串行异步传输。

当在 STEP 7 硬件组态或其它某组态应用程序中参数化模块时，可以选择此通讯模式。硬件目录中会出现九个版本的模块：

- Modbus 主站（4 个字节）
- Modbus 主站（8 个字节）
- Modbus 主站（32 个字节）
- Modbus 从站（4 个字节）
- Modbus 从站（8 个字节）
- Modbus 从站（32 个字节）
- USS 主站（4 个字节）
- USS 主站（8 个字节）
- USS 主站（32 个字节）

8 或 32 个字节的数据传输提高了吞吐效率，但需要的 ET 200S 机架上的 I/O 空间更多。通过对比，4 个字节的数据传输需要的机架上的 I/O 空间较少，但是吞吐效率较低。您选择的模块类型取决于您的应用要求。

下图显示了 ET 200S Modbus/USS 串行接口模块。



图片 3-1　　ET 200S Modbus/USS 串行接口模块

ET 200S Modbus/USS 串行接口模块具有以下功能：

- 符合 RS 232C、RS 422 或 RS 485 的集成接口
- 传输率最高 115.2 Kbaud，半双工
- 在模块固件中集成以下传输协议：
  - Modbus 主站驱动程序
  - Modbus 从站驱动程序
  - USS 主站驱动程序

驱动程序的功能取决于模块的参数化方式。下表列出了各个驱动程序接口的功能。

表格 3-1　　Modbus/USS 模块驱动程序的功能

功能	RS 232C	RS 422	RS 485
Modbus 驱动程序	是	是	是
RS 232C 信号的自动控制	是	否	否
USS 主站驱动程序	是	否	是

下表提供了模块的状态 LED 的简要说明。

表格 3-2　　LED

LED	说明
SF	该 LED（红色）指示出现错误情况。（组故障）
TX	该 LED（绿色）指示接口正在传输。
RX	该 LED（绿色）指示接口正在接收。

3.2 有关调试串行接口模块的简要说明

任务

根据在串行接口模块间发送和接收数据的示例，这些简要说明介绍了如何设置一个可以正常运行的应用、如何对串行接口模块（硬件和软件）进行基本操作以及如何测试硬件和软件。

在该示例中，我们将两个 ET 200S 1SI 串行接口模块作为一个 RS 232C Modbus 主站 **<->** Modbus 从站连接运行。

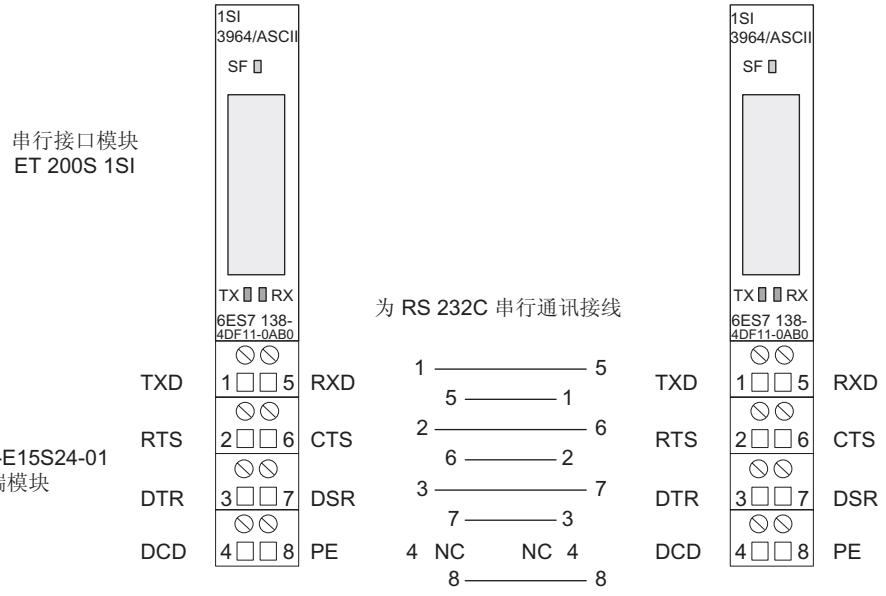
要求

必须满足以下要求：

- 必须通过 DP 主站在 S7 站上调试 ET 200S 站。
- 您将需要以下组件：
  - 两个 TM-E15S24-01 终端模块
  - 两个 ET 200S 1SI 串行接口模块
  - 必需的接线材料

安装、接线和装配

安装两个 TM-E15S24-01 终端模块并对其进行接线（请参见下图）。将两个 ET 200S 1SI 串行接口模块连接至这些终端模块。（名为 *《分布式 I/O》* 的手册对此进行了详细介绍。）



图片 3-2　　该示例的端子分配

所用的组态

下表显示了该实例程序所用的组态。

表格 3-3　　该实例应用的参数化

参数	**值**
组诊断	取消激活
接口	RS 232C
接收线路的初始状态	
模式	常规操作
从站地址 [1]	1
数据流量控制（初始状态）	无
波特率	9,600
停止位	1
奇偶校验	偶数
运行时间的倍数	1
响应时间 (ms) [2]	2,000
RTS = off 的时间 (ms)	
数据判断的等待时间 (ms)	
启动时清空接收缓冲区	是
[1] 仅使用 Modbus 从站 [2] 仅使用 Modbus 主站	

所用的块

下表显示了该实例程序所用的块。

模块	**符号**	**注释**
OB1	CYCLE	循环程序处理
OB100	RESTART	启动处理重新启动
DB21	SEND_IDB_SI_0	S_SEND_SI FB 的背景数据块
DB22	RECV_IDB_SI_1	S_RECV_SI FB 的背景数据块
DB40	SEND_WORK_DB_SI_0	标准 FB3 的工作 DB
DB41	RECV_WORK_DB_SI_1	标准 FB2 的工作 DB
DB42	SEND_SRC_DB_SI_0	发送的数据块
DB43	RECV_DST_DB_SI_0	接收数据块
DB81	MODSL_IDB_SI_1	S_MODB FB 的背景数据块
DB100	CONVERSION_DB	S_MODB FB 的转换 DB
FB2	S_RECV_SI	数据的接收标准 FB
FB3	S_SEND_SI	数据的发送标准 FB
FB81	S_MODB	Modbus 从站通讯的标准 FB
FC 10	Initiation	初始化数据块
FC 21	SEND_SI_0	发送数据
FC 22	RECV_SI_1	接收数据

供货和安装类型

ET 200S 1SI 模块和功能块的实例程序可通过以下网址获得：

http://support.automation.siemens.com/WW/view/en/10805265/133100

根据安装，您可以在 zXX21_11_1SI_MODBUS 项目中找到实例程序。

通过选择“文件”(File) > “打开”(Open) > “实例项目”(Sample projects) 在 STEP 7 SIMATIC 管理器中打开此项目。

该实例程序既可以作为已编译的程序使用，也可以作为 ASCII 源文件使用。还有一个符号表，其中包含了该示例中所用的符号。

如果没有第二个 ET 200S 1SI 模块可作为通讯伙伴，则您必须在 HW Config 中通过选择“编辑”(Edit) > “删除”(Delete) 删除第二个 ET 200S 1SI。另外，在 OB1 中，FB81 调用（Modbus 从站 FB）必须改为注释。

下载至 CPU

已完成示例的硬件设置并已连接编程设备。

CPU 存储器复位后（STOP 操作模式），将整个示例传输至用户存储器。将模式选择器从 STOP 切换至 RUN。

错误行为

如果在启动期间发生错误，则不会执行循环处理的块调用命令，并将设置出错指示 LED。

如果出现错误消息，则设置块的 ERROR 参数输出。然后，有关错误的更多详细说明将存储在块的 STATUS 参数中。如果 STATUS 参数包含 16#1E0E 或 16#1E0F 错误消息，则更多详细说明将存储在背景数据块的 SFCERR 变量中。

激活、启动程序

启动程序位于 OB100 中。

控制位和计数器在启动期间复位。

循环程序

循环程序位于 OB1 中。

在此示例中，如果涉及到 Modbus 主站，则功能块 FB2 S_RECV_SI 和 FB3 S_SEND_SI 与功能 FC 21 和 FC 22 一起工作，与数据块 DB21 和 DB22 一起用作背景数据块，与 DB42 和 DB43 一起用作传输和接收 DB。

如果涉及到 Modbus 从站，则 FB81 S_MODB 与 DB81 一起用作背景数据块，与 DB100 一起用作转换 DB。

在该示例中，功能块部分通过常量进行参数化，部分通过符号寻址的实际地址进行参数化。

说明

数据传输期间，插槽 2（Modbus 主站）上的 ET 200S 1SI 从插槽 3（Modbus 从站）上的 ET 200S 1SI“获取”数据。如果您使用的是其它通讯伙伴，则不调用 FB 81 (S_MODB)。

对 FC 21 (SEND) 的说明

程序段“生成沿 S_SEND_SI_REQ”：

S_SEND_SI 最初在 S_SEND_SI_REQ=0 时执行一次。然后，S_SEND_SI_REQ 设置为 1。如果在 S_SEND_SI_REQ 控制参数处检测到信号状态从 0 变为 1，则会启动 S_SEND_SI 作业。

如果 S_SEND_SI_DONE=1 或 S_SEND_SI_ERROR=1，则 S_SEND_SI_REQ 复位为 0。

程序段“S_SEND_SI_DONE=1”：

如果传输成功，则 S_SEND_SI_DONE 参数在 S_SEND_SI 的参数输出处设置为 1。

为了区分连续传输，工作数据块 DB40 的数据字 18 中含有一个发送计数器 (S_SEND_SI_WORK_CNT_OK)。

程序段“S_SEND_SI_ERROR=1”：

如果 S_SEND_SI_ERROR=1 时执行 S_SEND_SI，则错误计数器 S_SEND_SI_WORK_CNT_ERR 的数据字 20 将增加。另外，将复制 S_SEND_SI_WORK_STAT，因为它将在下一个周期中被 0 覆盖，从而使其无法读出。

对 FC 22 (RECEIVE) 的说明

程序段“启用接收数据”：

为了接收数据，S_RECV_SI 块上的 S_RECV_SI_EN_R 接收启用器必须设置为 1。

程序段“S_RECV_SI_NDR=1”：

如果设置了 S_RECV_SI_NDR，便意味着已收到新数据，从而接收计数器 S_RECV_SI_WORK_CNT_OK 将增加。

程序段“S_RECV_SI_ERROR=1”：

如果出现错误，即已在 S_RECV_SI 的参数输出处设置了错误位，则 S_RECV_SI_WORK_CNT_ERR 错误计数器将增加。另外，将复制 S_RECV_SI_WORK_STAT，因为它将在下一个周期中被 0 覆盖，从而使其无法读出。

所有相关值均可以在 VAT 中监视以进行测试。

对 DB42 的说明

在该示例中组态的作业功能代码 1（读取线圈状态）中，地址为“1”的 Modbus 从站将从起始地址“0”开始读取 16 位。读取的 16 位要使用 FC 22 (RECV) 从偏移地址为 0 开始存储在接收 DB (DB43) 中。

Modbus 主站作业 (FC 21 [SEND]) 的参数存储在发送的数据块 (DB24) 中。请参见下表：

地址	名称	类型	初始值	注释
0.0		STRUC		
+ 0.0	slave_address	BYTE	B#16#01	由 Modbus 从站“1”
+ 1.0	function_code	BYTE	B#16#01	使用 FC 1 （读取线圈状态）
+ 2.0	bit_start_addr	WORD	W#16#0000	从 Modbus 起始地址 0 开始
+ 4.0	bit_count	INT	16	读取 16 位（1 个字）
+ 6.0	a	ARRAY [1 到 1194]		
* 1.0		BYTE		
= 1,200.0		END_STRUCT		

触发 Modbus 主站作业

通过在 VAT 中设置标志 F 120.7 TRUE 触发 Modbus 主站作业。

对 DB100 的说明

在 Modbus 从站结束时，可以通过调用 FB81 (S_MODB) 使用请求的数据。

Modbus 主站消息帧中所用的地址存储在 SIMATIC 数据存储区中组态的转换 DB (DB100) 中，如下所示：

地址	名称	类型	初始值	注释
0.0		STRUCT		
+0.0	FC01_MOD_STRT_ADR_1	WORD	W#16#0	将 Modbus 地址 0 到 255 从 0 映射到 SIMATIC 标志区
+2.0	FC01_MOD_END_ADR_1	WORD	W#16#0FF	
+4.0	FC01_CNV_TO_FLAG_A	WORD	W#16#0	
+6.0	FC01_MOD_STRT_ADR_2	WORD	W#16#100	
+8.0	FC01_MOD_END_ADR_2	WORD	W#16#1FF	
+10.0	FC01_CNV_TO_OUTPUT	WORD	W#16#0	
+12.0	FC01_MOD_STRT_ADR_3	WORD	W#16#200	
+14.0	FC01_MOD_END_ADR_3	WORD	W#16#2FF	
+16.0	FC01_CNV_TO_TIMER	WORD	W#16#0	
+18.0	FC01_MOD_STRT_ADR_4	WORD	W#16#300	
+20.0	FC01_MOD_END_ADR_4	WORD	W#16#3FF	
+22.0	FC01_CNV_TO_COUNTER	WORD	W#16#0	
+24.0	FC02_MOD_STRT_ADR_5	WORD	W#16#0	
+26.0	FC02_MOD_END_ADR_5	WORD	W#16#0FF	
+28.0	FC02_CNV_TO_FLAG_B	WORD	W#16#0	
+30.0	FC02_MOD_STRT_ADR_6	WORD	W#16#100	
+32.0	FC02_MOD_END_ADR_6	WORD	W#16#2FF	
+34.0	FC02_CNV_TO_INPUT	WORD	W#16#0	
+36.0	FC03_06_16_DB_NO	WORD	W#16#02A	
+38.0	FC04_DB_NO	WORD	W#16#02A	
+40.0	DB_MIN	WORD	W#16#02A	
+42.0	DB_MAX	WORD	W#16#02A	
+44.0	FLAG_MIN	WORD	W#16#0	启用标志区 0 到 255
+46.0	FLAG_MAX	WORD	W#16#0FF	
+48.0	OUTPUT_MIN	WORD	W#16#0	
+50.0	OUTPUT_MAX	WORD	W#16#0FF	
=52.0		END_STRUCT		

在此特定示例中，Modbus 地址 0 到 255（通过 FC 1 请求）通过 DB100 的地址 0 到 4 从 0 开始映射到 SIMATIC 标志区中。

DB100 找到 44 和 46 的地址后，启用 SIMATIC 标志区 0 到 255 以进行 Modbus 主站作业。

3.3 端子分配图

3.3.1 端子分配

接线准则

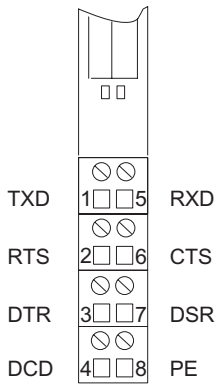
必须将电缆（端子 1 到 8）屏蔽，且必须在两端连接屏蔽。可以使用屏蔽触点实现此目的。有关这些触点的信息可以在名为*《ET 200S 分布式 I/O 系统》*的手册的*『附件』*一节中找到。

RS 232C 通讯的端子分配

可以使用从站系统建立点对点连接。不支持 RS 232C 接口的反向通道。

下表显示了设置了 RS 232C 通讯协议后 ET 200S Modbus/USS 串行接口模块的端子分配。

表格 3-4　RS 232C 通讯的端子分配

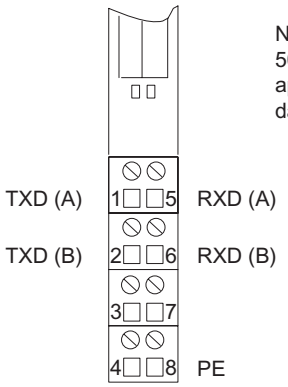
视图	备注		
	模式：全双工		
	端子		
	1	TXD	发送的数据
	5	RXD	接收的数据
	2	RTS	请求发送
	6	CTS	明确发送
	3	DTR	数据终端准备就绪
	7	DSR	数据集准备就绪
	4	DCD	检测到的数据载体
	8	PE	接地

RS 422 通讯的端子分配

可以使用从站系统建立点对点连接。

下表显示了设置了 RS 422 通讯协议后 ET 200S Modbus/USS 串行接口模块的端子分配。

表格 3-5　　RS 422 通讯的端子分配

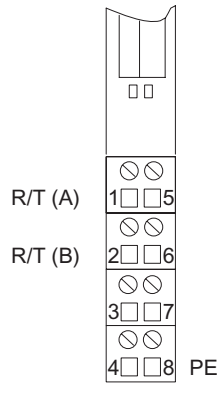
视图	端子分配	备注		
		模式：全双工		
		端子		
		1	TXD	(A)-
		5	RXD	(A)-
		2	TXD	(B)+
		6	RXD	(B)+
		8	PE	接地

RS 485 通讯的端子分配

可以使用主站系统建立最多包含 32 个从站的多端点连接（网络）。模块驱动程序在发送和接收之间切换接收线路。

下表显示了设置了 RS 485 通讯协议后 ET 200S Modbus/USS 串行接口模块的端子分配。

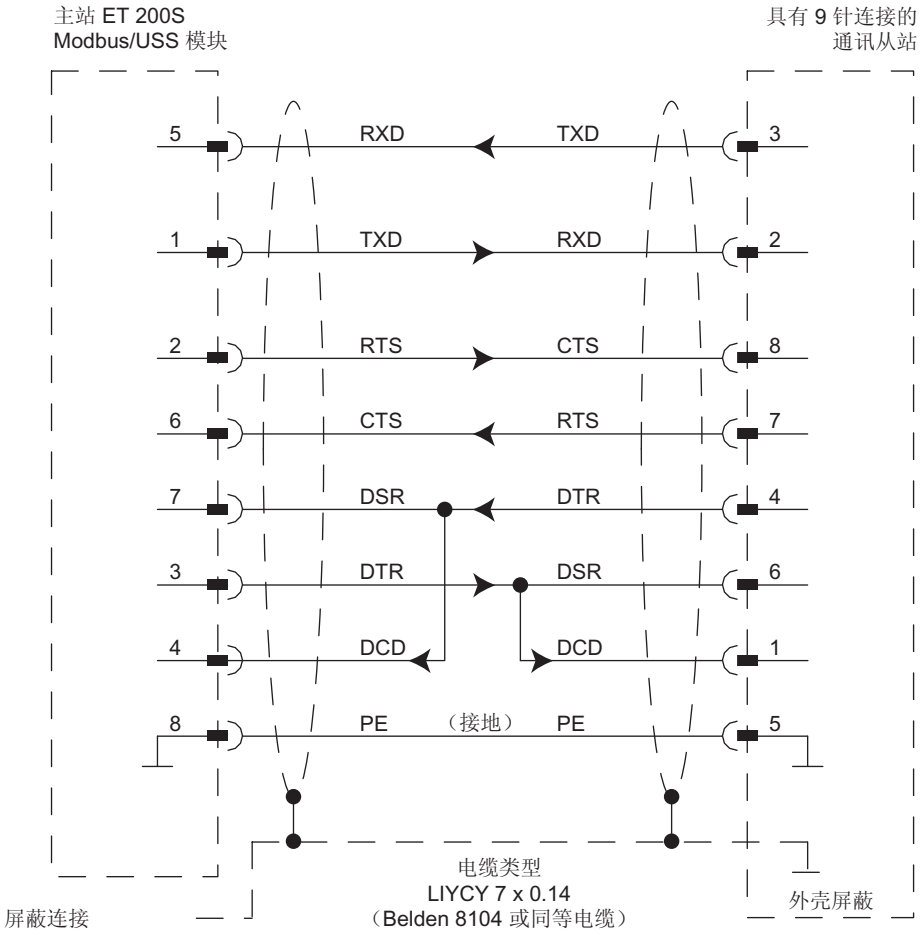
表格 3-6　　RS 485 通讯的端子分配

视图	端子分配	备注		
		模式：全双工		
		端子		
		1	R/T	(A)-
		2	R/T	(B)+
		8	PE	接地

用于 9 针电缆连接器的 RS 232C 连接电缆的端子分配

下图显示了该模块和具有 9 针 D 型连接插槽的通讯从站之间的 RS 232C 点对点通讯的电缆连接。

- 在 ET 200S 端上，信号线连接至相应编号的端子。
- 使用通讯从站上的 9 针 D 型连接插槽。

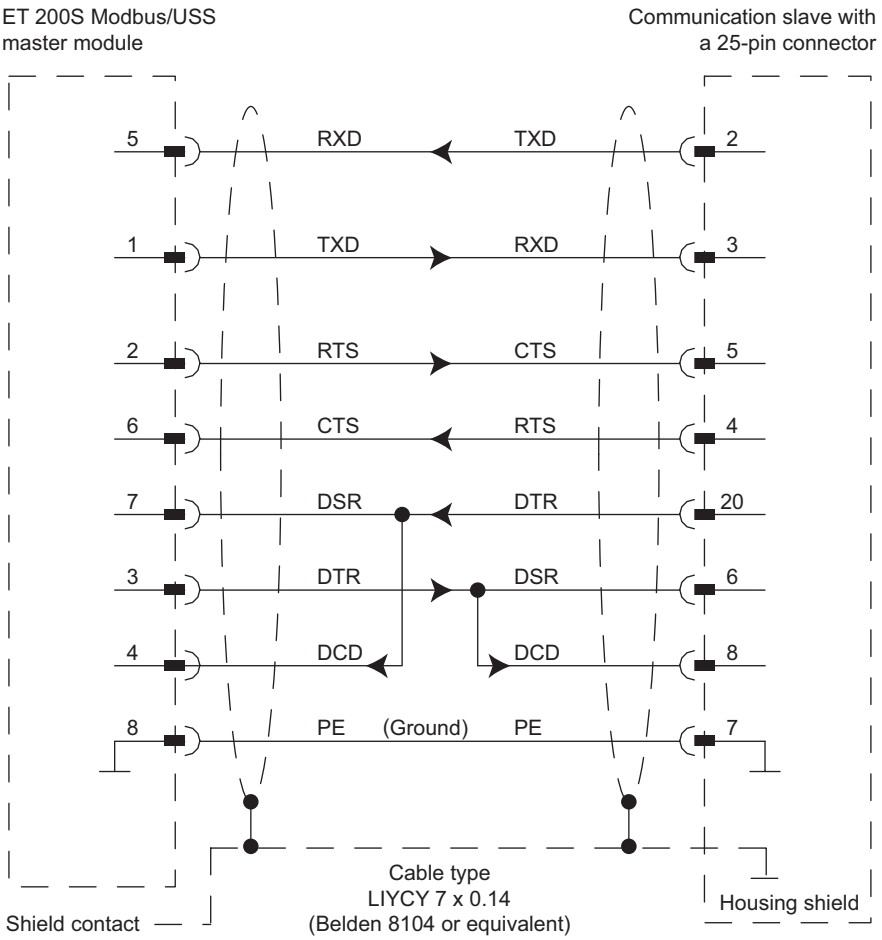


图片 3-3　　用于 9 针电缆连接器的 RS 232C 连接电缆（1 个主站、1 个从站系统）

用于 25 针电缆连接器的 RS 232C 连接电缆的端子分配

下图显示了该模块和具有 25 针 D 型电缆连接器的通讯从站之间的 RS 232C 点对点通讯的电缆连接。

- 在 ET 200S 端上，信号线连接至相应编号的端子。
- 使用通讯从站上的 25 针 D 型电缆连接器。

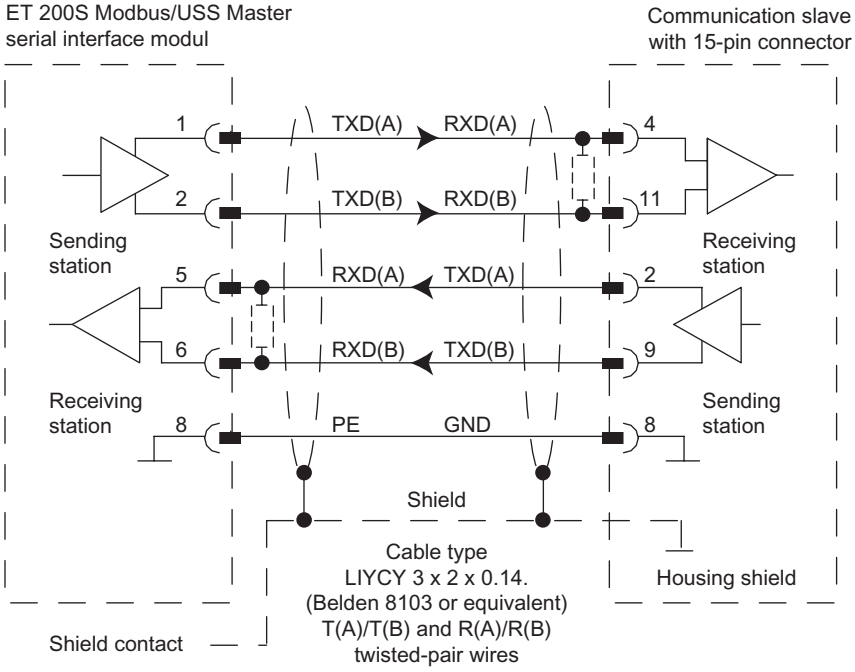


图片 3-4　　用于 25 针电缆连接器的 RS 232C 连接电缆（1 个主站、1 个从站系统）

用于 15 针电缆连接器的 RS 422 连接电缆的端子分配

下图显示了该模块和具有 15 针 D 型电缆连接器的通讯从站之间的 RS 422 通讯的电缆连接。

- 在 ET 200S 端上，信号线连接至相应编号的端子。
- 使用通讯从站上的 15 针 D 型电缆连接器。



图片 3-5　　用于 15 针电缆连接器的 RS 422 连接电缆（1 个主站、1 个从站系统）

说明

如果电缆长度超过 50 m，则附加一个约 330 Ω（请参见下图）的终端电阻以确保数据通讯畅通。

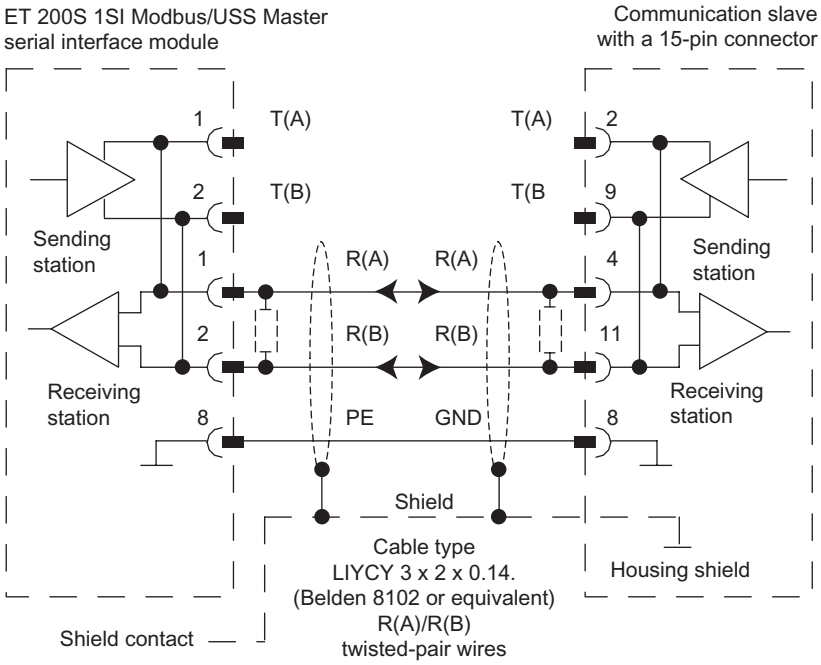
38,400 波特时，这种电缆的最大长度为 1,200 m。

- 19,200 波特时，最长为 1200 m
- 38,400 波特时，最长为 500 m
- 76,800 波特时，最长为 250 m

用于 15 针电缆连接器的 RS 485 连接电缆的端子分配

下图显示了该模块和具有 15 针 D 型电缆连接器的通讯从站之间的 RS 485 通讯的电缆连接。

- 在 ET 200S 端上，信号线连接至相应编号的端子。
- 使用通讯从站上的 15 针 D 型电缆连接器。



图片 3-6　　用于 15 针电缆连接器的 RS 485 连接电缆（1 个主站、1 个从站系统）

说明

如果电缆长度超过 50 m，则附加一个约 330 Ω（请参见下图）的终端电阻以确保数据通讯畅通。

38,400 波特时，这种电缆的最大长度为 1,200 m。

- 19,200 波特时，最长为 1200 m
- 38,400 波特时，最长为 500 m
- 76,800 波特时，最长为 250 m
- 115,200 波特时，最长为 200 m

3.3.2 RS 232C 接口

RS 232C 接口特性

RS 232C 接口是一种符合 RS 232C 标准的、用于串行数据传输的电压接口。下表显示了 RS 232C 的特性。

表格 3-7 RS 232C 接口信号

特性	说明
类型	电压接口
前连接器	8 针标准 ET 200S 端子连接器
RS 232C 信号	TXD、RXD、RTS、CTS、DTR、DSR、DCD、GND
传输率	最高 115.2 Kbaud
电缆长度	最长 15 m，电缆类型 LIYCY 7 x 0.14
相关标准	DIN 66020、DIN 66259、EIA RS 232C、CCITT V.24/V.28
保护类型	IP20

RS 232C 信号

Modbus/USS 模块支持 RS 232C 信号。

表格 3-8 RS 232C 接口信号

信号	说明	含义
TXD	发送的数据	在空闲状态下，传输线路保持在逻辑“1”处。
RXD	接收的数据	接收线路必须通过通讯伙伴保持在逻辑“1”处。
RTS	请求发送	ON：模块明确发送。 OFF：模块不处于发送模式。
CTS	明确发送	通讯伙伴可以从 ET 200S 接收数据。串行接口模块将此作为对 RTS = ON 的响应。
DTR	数据终端准备就绪	ON：模块接通电源并已做好运行准备。 OFF：模块未接通电源并且未做好运行准备。
DSR	数据集准备就绪	ON：通讯伙伴接通电源并已做好运行准备。 OFF：通讯伙伴未接通电源并且未做好运行准备。
DCD	检测到的数据载体	连接调制解调器时的载波信号。

伴随信号的自动控制

按照如下所述在模块上执行 RS 232C 伴随信号的自动控制：

- 模块被组态为以 RS 232C 伴随信号的自动控制模式运行，会立即将 RTS 线路设置为 OFF 并将 DTR 线路设置为 ON（模块已做好运行准备）。

  这可以防止消息帧的传输，除非 DTR 线路设置为 ON。只要 DTR = OFF，就无法通过 RS 232C 接口接收到任何数据。所有发送作业都将取消，同时生成一条相应的错误消息。

- 发送作业排队后，模块将设置 RTS = ON，并触发已组态的数据输出等待时间。当超过数据输出时间并且 CTS = ON 时，数据通过 RS 232C 接口发送。
- 如果在数据输出等待时间内 CTS 线路未设置为 ON 或在传输期间 CTS 更改为 OFF，则模块将取消发送作业并生成错误消息。
- 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。ET 200S 不会等待 CTS 更改为 OFF。
- DSR 线路设置为 ON 后，即可通过 RS 232C 接口接收数据。如果模块的接收缓冲区即将发生上溢，则模块不会进行响应。
- 如果 DSR 从 ON 更改为 OFF，则将取消活动的发送作业或数据接收操作，并输出错误消息。

说明

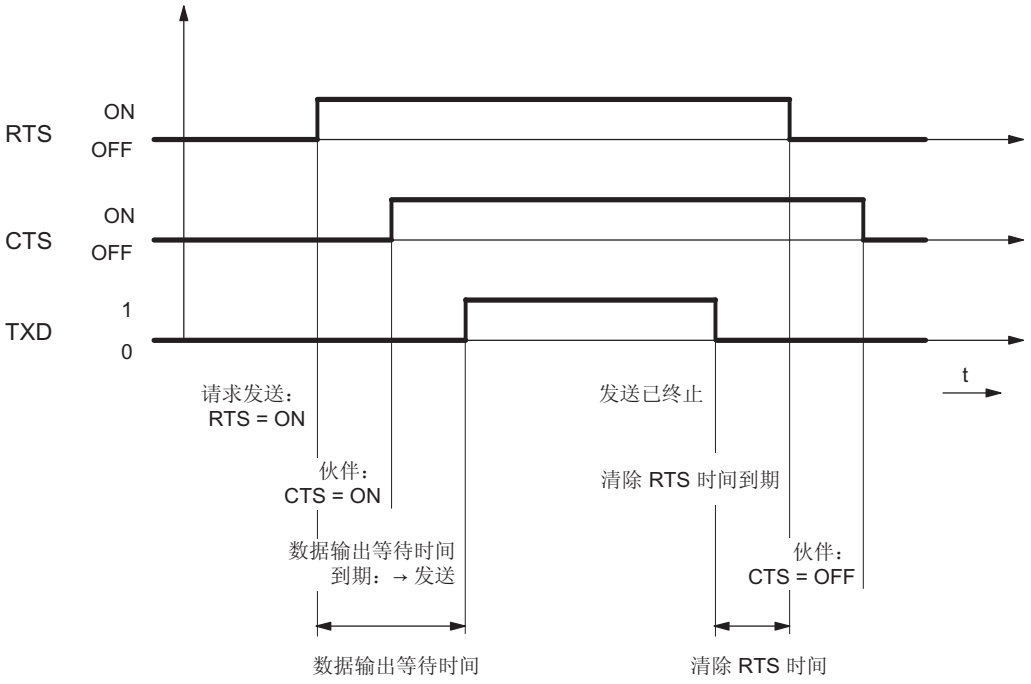
RS 232C 伴随信号的自动控制仅可以在半双工模式下进行。

说明

必须在参数化接口上设置“到 RTS 变为 OFF 的时间”，以便通讯伙伴可以在 RTS（以及相应的发送作业）取消之前接收到消息帧的全部最新字符。必须设置“数据输出等待时间”，以便通讯伙伴在超出该时间前已做好接收准备。

伴随信号的时序图

下图说明了发送作业的时间顺序：



图片 3-7　　RS 232C 伴随信号的自动操作时序图

3.3.3　RS 422/485 接口

RS 422/485 接口特性

RS 422/485 接口是一种符合 RS 422/485 标准的、用于串行数据传输的差分电压接口。下表显示了 RS 422/485 接口的特性。

表格 3-9　　RS 422/485 接口特性

特性	说明
类型	差分电压接口
前连接器	8 针标准 ET 200S 端子连接器
RS 422 信号	TXD (A)-、RXD (A)-、TXD (B)+、RXD (B)+、GND
RS 485 信号	R/T (A)-、R/T (B)+、GND
传输率	最高 115.2 Kbaud
电缆长度	最长 1,200 m，电缆类型 LIYCY 7 x 0.14
相关标准	EIA RS 422/485、CCITT V.11/V.27
保护类型	IP20

3.4 Modbus 传输协议

3.4.1 属性和消息帧结构

特性

Modbus 的传输过程是一个代码透明的异步半双工过程。数据不通过握手进行传输。

模块启动传输（作为主站）。发送作业消息帧之后，模块将在响应监视时间内等待来自从站的响应消息帧。

消息结构

“主站-从站”和/或“从站-主站”的数据交换自从站地址开始，其后是功能代码。然后传输数据。“主站-从站”和/或“从站-主站”的数据交换包括以下元素：

从站地址	Modbus 从站地址
功能代码	Modbus 功能代码
数据	消息帧数据：Byte_Count、Coil_Number、Data
CRC 校验	消息帧校验和

数据域的结构取决于使用的功能代码。CRC 校验在消息末尾进行传输。下表显示了消息帧结构的组成部分。

表格 3-10　消息结构

地址	功能	数据	CRC 校验
字节	字节	n 字节	2 字节

3.4.2 从站地址

说明

从站地址的范围为 1 至 255。该地址用于对总线上定义的从站进行寻址。

传输消息帧

主站使用从站地址 0 对总线上的所有从站进行寻址。

说明

传输消息帧仅允许与功能代码 05、06、15 和/或 16 组合。

传输消息帧后面不跟着来自从站的响应消息帧。

3.4.3 主站和从站功能代码

主站和从站功能代码

功能代码定义了消息帧的含义和结构。下表列出了主站和从站的功能代码及其可用性。

表格 3-11 主站和从站功能代码

功能代码	说明	主站	从站
01	读取线圈状态	√	√
02	读取输入状态	√	√
03	读取保持寄存器	√	√
04	读取输入寄存器	√	√
05	强制单个线圈	√	√
06	预设单个寄存器	√	√
07	读取例外状态	√	-
08	回送测试	√	√
11	获取通讯事件计数器	√	-
12	获取通讯事件日志	√	-
15	强制多个线圈	√	√
16	预设多个寄存器	√	√

3.4.4 数据域 DATA

说明

数据域 DATA 用于传输以下功能代码特定的数据：

- 字节计数
- 线圈起始地址
- 寄存器起始地址
- 线圈数
- 寄存器数

3.4.5 消息结束和 CRC 校验

说明

消息帧的结束通过循环冗余码校验的校验和 16（包括 2 个字节）来形成，它按照以下多项式进行计算：

$x^{16} + x^{15} + x^2 + 1$

先传输低位字节，然后传输高位字节。

消息帧结束的检测

如果在传输三个半字符所需的时间（为字符延迟时间的 3.5 倍）内没有进行任何传输，则 Modbus/USS 模块会将此解释为消息帧的结束。

消息帧结束的超时取决于波特率。

超过消息帧结束的超时后，将判断自从站收到的响应消息帧并校验其格式。

表格 3-12　消息帧结束

传输率	超时
115,200 bps	1 ms
76,800 bps	1 ms
57,600 bps	1 ms
38,400 bps	1 ms
19,200 bps	2 ms
9,600 bps	4 ms
4,800 bps	8 ms
2,400 bps	16 ms
1,200 bps	32 ms
600 bps	65 ms
300 bps	130 ms
115 bps	364 ms

3.4.6 异常响应

出错时的响应消息帧

如果从站检测到主站的作业消息帧中有错误（例如寄存器地址非法），它会执行以下操作：

- 从站将设置响应消息帧的功能代码中最重要的位。
- 从站将发送一个字节的错误代码（异常代码）指明错误原因。

示例：异常代码消息帧

例如，错误代码响应消息帧可以按照如下进行构造：从站地址 5、功能代码 5、异常代码 2。

来自从站 EXCEPTION_CODE_xx 的响应消息帧	05H	从站地址
	85H	功能代码
	02H	异常代码（1 至 7）
	xxH	循环冗余码校验代码“低”
	xxH	循环冗余码校验代码“高”

当自驱动程序接收到错误代码响应消息帧时，当前作业会因出错而终止。

此外，与收到的错误代码（异常代码 1 至 7）对应的错误编号会输入到 SYSTAT 区域中。

S_RCV 目标数据块中不进行任何输入操作。

错误代码表

下表列出了模块发送的错误代码。

表格 3-13　错误代码

异常代码	说明	可能的原因
01	功能非法	接收到的功能代码非法。
02	数据地址非法	访问未启用的 SIMATIC 区域（请参见 Modbus 数据转换表）
03	数据值非法	长度大于 2,040 位或 127 个寄存器；FC 05 的数据域不是 FF00 或 0000；FC 08 的诊断子码 <> 0000。
04	关联设备故障	尚未通过 Modbus 通讯 FB 执行初始化，或 FB 报告错误。 模块和 CPU 之间存在数据传输错误（例如：没有 DB）；超过最大可传输数据长度（块大小 CPU <-> 模块）。

3.5 Modbus 主站驱动程序

3.5.1 使用 Modbus 主站驱动程序

用途

ET 200S Modbus 驱动程序可以在 S7 自动化系统中使用，并可以与伙伴系统建立串行通讯连接。

该驱动程序用于设置 ET 200S Modbus 主站驱动程序和启用了 Modbus 的控制系统之间的通讯连接。

传输顺序

使用 RTU 格式的 Modbus 协议进行传输。使用主站-从站原理传输数据。

主站启动传输。

Modbus 主站可以使用功能代码 01、02、03、04、05、06、07、08、11、12、15 和 16。

可用的接口和协议

可以将 RS 232 或 RS 422/485 (X27) 接口用于模块。

通过该驱动程序，可以将 RS 422/485 接口用于两线制操作和四线制操作。在半双工两线制操作中，可以将多达 32 个从站连接到同一个主站。这可以创建多点连接（网络）。在半双工四线制操作 (RS 422) 中，只能有 1 个主站和 1 个从站。

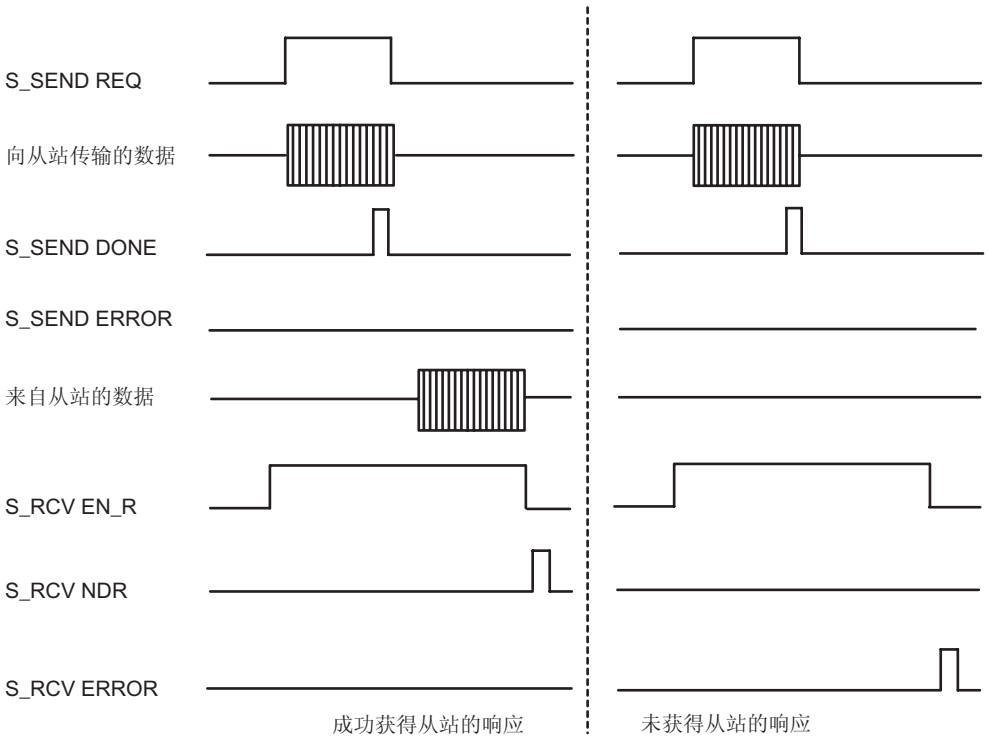
3.5.2 ET 200S Modbus 主站的数据传输

简介

通过 S_SEND 和 S_RCV FB 在模块和 CPU 之间传输数据。数据要输出时，通过 REQ 输入处的沿激活 S_SEND FB。EN_R=1 时，S_RCV FB 已准备好接收数据。所有读取功能代码均需要 S_RCV。

FB3 S_SEND：将数据发送给通讯伙伴

必须激活 S_SEND 和 S_RCV FB 才能执行 Modbus 主站作业。数据要输出到模块时，通过 REQ 输入处的沿激活 S_SEND FB。EN_R=1 时，S_RCV FB 已准备好从模块接收数据。所有读取功能代码均需要 S_RCV。下图显示了执行 Modbus 作业时 S_SEND 和 S_RCV 参数的整体特性。



图片 3-8 Modbus 作业的时序图

通过 REQ 输入处的正跳沿启动数据传输。根据数据量的大小，数据可能通过多次调用（程序周期）进行传输。

通过将参数输入 R 处的信号状态设置为“1”，可以循环调用 S_SEND FB。这会取消到模块的传输并将 S_SEND FB 复位为其初始状态。模块已接收到的数据仍发送给通讯伙伴。如果输入 R 的信号状态保持为静态“1”，则发送已被取消激活。

要寻址的串行接口模块 ET 200S 1SI 的地址在 LADDR 参数中指定。

DONE 输出显示“作业已完成且无错”。ERROR 指示是否发生了错误。如果发生了错误，则在 STATUS 中显示相应的事件编号。如果没有发生错误，STATUS 的值为 0。还会在 S_SEND FB 的 RESET 处输出 DONE 和 ERROR/STATUS。如果出现错误，则复位二进制结果。如果块终止且无错，则二进制结果的状态为“1”。

启动

S_SEND FB 的 COM_RST 参数通知 FB 启动。

将启动 OB 中的 COM_RST 参数设置为 1。

在循环操作中调用 FB，而不设置或复位 COM_RST 参数。

如果设置了 COM_RST 参数：

- FB 包含有关 ET 200S 1SI 模块的信息（I/O [不管是不是分布式 I/O] 区域中的字节数）。
- FB 将复位并终止先前启动的所有作业（在 CPU 最后一次跳转到 STOP 之前）。

FB 获得有关 ET 200S 1SI 模块的信息后，它将复位 COM_RST 参数本身。

下表显示了 FB3 S_SEND 的 STL 和 LAD 表达式。

说明

REQ 输入通过沿来触发。REQ 输入处的正跳沿足以将其触发。传输期间，逻辑运算的结果并不一定是“1”。

说明

必须将 EN_R 输入设置为静态“1”。在整个接收作业过程中，必须为 EN_R 参数提供逻辑运算结果“1”。

说明

S_SEND 功能块没有参数检查。如果参数无效，CPU 会切换为 STOP 模式。

必须先完成 S_SEND FB 的 ET 200S-CPU 启动机制，才能在 CPU 从 STOP 更改为 RUN 模式后处理触发的作业。在此期间启动的任何作业都不会丢失。启动协调完成后，这些作业将传输至模块。

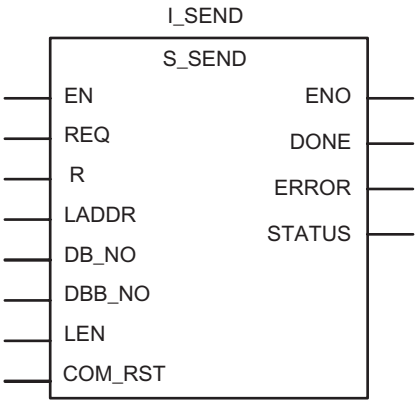
FB3 调用

下表显示了 FB3 S_SEND 的 STL 和 LAD 表达式。

表格 3-14　FB3 S_SEND 的 STL 和 LAD 表达式

STL 表达式	LAD 表达式

```
CALL            S_SEND, I_SEND
     REQ:        =
     R:          =
     LADDR:      =
     DB_NO:      =
     DBB_NO:     =
     LEN:        =
     DONE:       =
     ERROR:      =
     STATUS:     =
     COM_RST:    =
```



说明

参数 EN 和 ENO 仅存在于图形表达式（LAD 或 FBD）中。编译器使用二进制结果处理这些参数。

如果块终止且无错，则将二进制结果设置为信号状态“1”。如果出现错误，则将二进制结果设置为“0”。

数据区中的分配

S_SEND FB 与 I_SEND 背景数据块一起使用。调用时提供 DB 号。背景数据块中的数据无法访问。

说明

例外：如果出现错误 STATUS == W#16#1E0F，可以参考 SFCERR 变量获得其它详细信息。该错误变量只能通过对背景数据块进行符号访问来装载。

FB3 S_SEND 参数

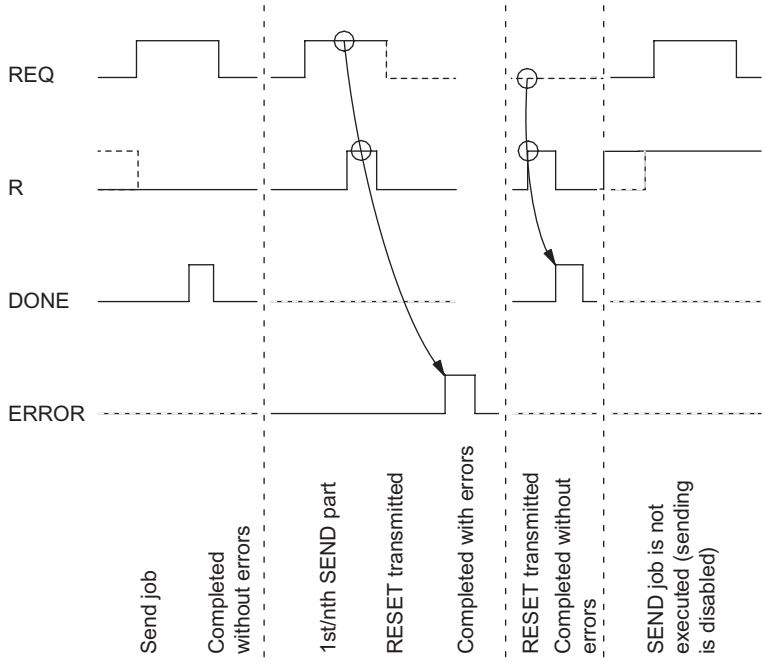
下表列出了 S_SEND (FB3) 参数。

表格 3-15　FB3：S_SEND 参数

名称	类型	数据类型	说明	允许的值、备注
REQ	INPUT	BOOL	出现正跳沿时启动作业	
R	INPUT	BOOL	取消作业	取消正在进行的作业。发送被阻。
LADDR	INPUT	INT	ET 200S 串行接口的起始地址	起始地址从 STEP 7 获取。
DB_NO	INPUT	INT	数据块号	发送的数据块号：依 CPU 而定，不允许为零
DBB_NO	INPUT	INT	数据字节号	0 ≤ DBB_NO ≤ 8190 按数据字发送的数据
LEN	INPUT	INT	数据长度	1 ≤ LEN ≤ 224，通过字节数指定
DONE[1]	OUTPUT	BOOL	作业已完成且无错	STATUS 参数 == 16#00
ERROR[1]	OUTPUT	BOOL	作业已完成但有错	错误信息已写入 STATUS 参数。
STATUS[1]	OUTPUT	WORD	错误规范	如果 ERROR == 1，则 STATUS 参数将包含错误信息。
COM_RST	IN_OUT	BOOL	重新启动 FB	
[1] 这些参数可用于成功执行发送作业之后的一个 CPU 周期。				

FB3 S_SEND 的时序图

下图根据 REQ 和 R 输入的接线方式说明了 DONE 和 ERROR 参数的特性。



图片 3-9　　FB3 P_SEND 的时序图

说明

REQ 输入通过沿来触发。REQ 输入处的正跳沿足以将其触发。传输期间，逻辑运算的结果并不一定是“1”。

FB2 S_RCV：从通讯伙伴接收到的数据

S_RCV FB 将数据从模块发送到由参数 DB_NO 和 DBB_NO 指定的 S7 数据区中。为了进行数据传输，S_RCV FB 可以循环调用，也可以在时间控制的程序中静态（无条件）调用。

EN_R 参数的（静态）信号状态为“1”时会启用检查，以确定是否要从串行接口读取数据。EN_R 参数处的信号状态为“0”可以取消活动的传输事件。取消的接收作业终止，且生成错误消息（STATUS 输出）。只要 EN_R 参数处的信号状态为“0”，就会禁用接收。根据数据量的大小，数据可能通过多次调用（程序周期）进行传输。

如果功能块检测到“R”参数处的信号状态为“1”，则将取消当前的传输作业并将 S_RCV FB 复位为其初始状态。只要 R 参数处的信号状态为“1”，就会禁用接收。如果信号状态返回“0”，则从头重新开始接收已取消的消息帧。

要寻址的串行接口模块 ET 200S 1SI 在 LADDR 参数中指定。

NDR 输出指示“作业已完成且无错/数据已接受”（已读取所有数据），ERROR 指示是否发生了错误。如果发生了错误，则当接收缓冲区占用超出整体的 2/3 时，则在 STATUS 中显示相应的错误编号。如果未设置 ERROR，则无论何时调用 S_RCV，STATUS 都将包含警告。如果未出现错误或警告，则 STATUS 的值为 0。

如果 S_RCV FB 复位（参数 LEN = 16#00），也会输出 NDR 或 ERROR/STATUS。如果出现错误，则复位二进制结果。如果块终止且无错，则二进制结果的状态为“1”。

启动

S_RCV FB 的 COM_RST 参数通知 FB 启动。

将启动 OB 中的 COM_RST 参数设置为 1。

在循环操作中调用 FB，而不设置或复位 COM_RST 参数。

如果设置了 COM_RST 参数：

- FB 包含有关 ET 200S 1SI 模块的信息（I/O [不管是不是分布式 I/O] 区域中的字节数）。
- FB 将复位并终止先前启动的所有作业（在 CPU 最后一次跳转到 STOP 之前）。

FB 获得有关 ET 200S 1SI 模块的信息后，它将复位 COM_RST 参数本身。

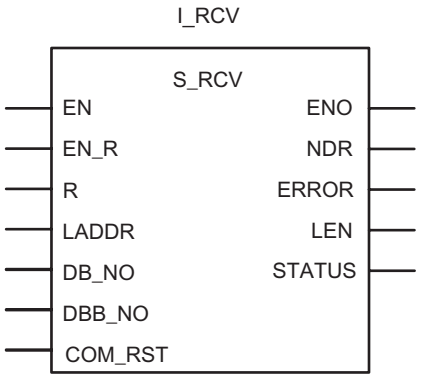
说明

S_RCV 功能块没有参数检查。如果参数无效，CPU 可能会切换为 STOP 模式。

必须先完成 S_RCV FB 的 ET 200S-CPU 启动机制，才能在 CPU 从 STOP 更改为 RUN 模式后接收作业。

下表显示了 FB2 S_RCV 的 STL 和 LAD 表达式。

表格 3-16　FB2 S_RCV 的 STL 和 LAD 表达式

STL 表达式	LAD 表达式
CALL S_RCV, I_RCV EN_R: = R: = LADDR: = DB_NO: = DBB_NO: = NDR: = ERROR: = LEN: = STATUS: = COM_RST: =	

说明

参数 EN 和 ENO 仅存在于图形表达式（LAD 或 FBD）中。编译器使用二进制结果处理这些参数。

如果块终止且无错，则将二进制结果设置为信号状态“1”。如果出现错误，则将二进制结果设置为“0”。

数据区中的分配

S_RCV FB 与 I_RCV 背景数据块一起使用。调用时提供 DB 号。背景数据块中的数据无法访问。

下表列出了 FB2 S_RCV 参数。

说明

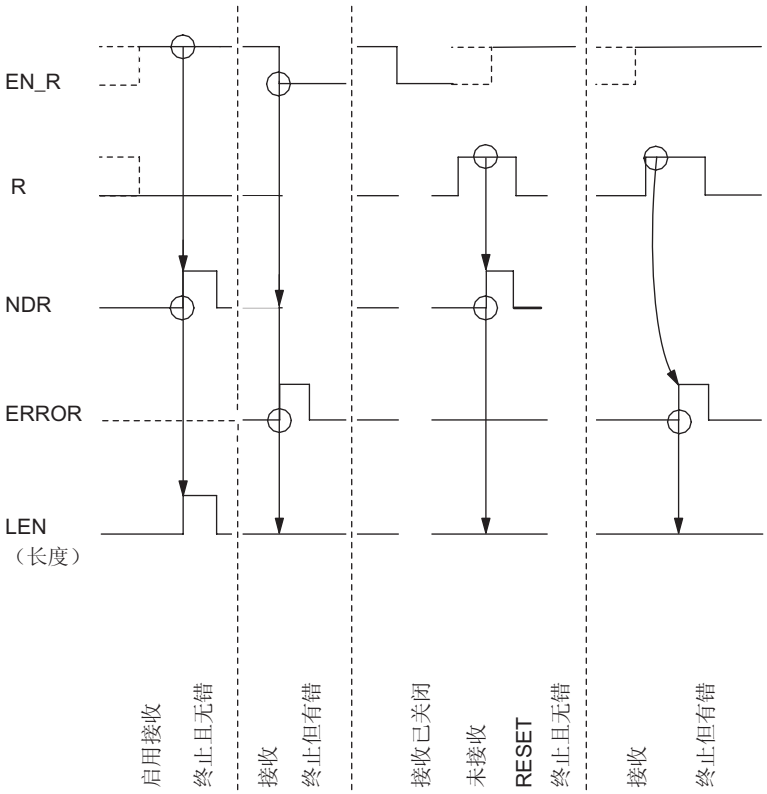
例外：如果出现错误 STATUS == W#16#1E0D，可以参考 SFCERR 变量获得其它详细信息。该错误变量只能通过对背景数据块进行符号访问来装载。

表格 3-17　FB2：S_RCV 参数

名称	类型	数据类型	说明	允许的值、备注
EN_R	INPUT	BOOL	启用数据读取	
R	INPUT	BOOL	取消作业	取消正在进行的作业。接收被阻。
LADDR	INPUT	INT	ET 200S 串行接口的起始地址	起始地址从 STEP 7 获取。
DB_NO	INPUT	INT	数据块号	接收数据块号：依 CPU 而定，不允许为零
DBB_NO	INPUT	INT	数据字节号	0 ≤ DBB_NO ≤ 8190 按数据字接收的数据
NDR[1]	OUTPUT	BOOL	作业已完成且无错，数据已接受	STATUS 参数 == 16#00
ERROR[1]	OUTPUT	BOOL	作业已完成但有错	错误信息已写入 STATUS 参数。
LEN[1]	OUTPUT	INT	已接收消息帧的长度	1 ≤ LEN ≤ 224，通过字节数指定
STATUS[1]	OUTPUT	WORD	错误规范	如果 ERROR == 1，则 STATUS 参数将包含错误信息。
COM_RST	IN_OUT	BOOL	重新启动 FB	
[1] 这些参数可用于成功执行接收作业之后的一个 CPU 周期。				

FB2 S_RCV 的时序图

下图根据 EN_R 和 R 输入的接线方式说明了参数 NDR、LEN 和 ERROR 的特性。



图片 3-10　　FB2 S_RCV 的时序图

说明

必须将 EN_R 输入设置为静态“1”。在整个接收作业过程中，必须为 EN_R 参数提供逻辑运算结果“1”。

3.5.3 为 Modbus 主站组态和设置参数

组态 Modbus 模块

如果您要通过 PROFIBUS 网络使用 S7 主站与模块通讯，则需要使用 STEP 7 硬件组态设备在 PROFIBUS 网络上安装模块并设置其通讯参数。

如果在硬件目录中选择 Modbus 主站并将其添加到网络组态中的 ET 200S 基础模块，则模块的订货号、插槽号以及输入与输出的地址将自动传输到组态表中。然后，便可以调用 Modbus 主站的属性对话框，并设置通讯类型和其它参数。

参数化主站驱动程序

下表列出了可以为模块的 Modbus 驱动程序设置的参数。

表格 3-18　Modbus 主站驱动程序的参数

参数	说明	值范围	缺省值
诊断中断	指定在出现严重错误时，模块是否应生成诊断中断。	• 否 • 是	否
激活 BREAK 检测	如果发生线路断路或未连接接口电缆，该模块将产生错误消息“断路”。	• 否 • 是	否
接口类型	指定要使用的电接口。	• RS 232C • RS 422（全双工） • RS 485（半双工）	RS 232C
接收线路的半双工和全双工初始状态	指定 RS 422 和 RS 485 操作模式下接收线路的初始状态。不是在 RS 232C 操作模式下使用的初始状态。 仅在更换零件时需要确保兼容性的情况下，才需要“反转信号电平”设置。	RS 422： R(A) 5 V/R(B) 0 V (BREAK) R(A) 0 V/R(B) 5 V 反转信号电平 RS 485： 无 R(A) 0 V/R (B) 5 V	RS 422： R(A) 5 V/R(B) 0 V (BREAK) RS 485： R(A) 0 V/R (B) 5 V
数据流量控制 （使用缺省参数；在用户程序中更改缺省值）	您可以发送和接收具有数据流量控制的数据。如果一个通讯伙伴比另一个运行快，则可以通过数据流量控制使数据传输同步。选择数据流量控制类型并设置相关参数。 注意：使用 RS 485 接口无法进行数据流量控制。仅 RS 232C 接口支持使用“V.24 信号的自动控制”进行数据流量控制。	• 无 • V.24 信号的自动控制	无

参数	说明	值范围	缺省值
传输率	选择数据传输率（单位：位/秒）。	• 110 • 300 • 600 • 1,200 • 2,400 • 4,800 • 9,600 • 19,200 • 38,400 • 57,600 • 76,800 • 115,200	9,600
停止位	选择数据传输期间附加到每个字符的停止位数，以指示字符的结束。	• 1 • 2	1
奇偶校验	可以将数据位序列扩展一个字符以包括奇偶校验位。附加值（0 或 1）将所有位（数据位和奇偶校验位）的值设置为已定义状态。 **无**：发送的数据不包含奇偶校验位。 **奇数**：设置奇偶校验位，以便所有数据位（包括奇偶校验位）的总数返回一个信号状态为“1”的奇数值。 **偶数**：设置奇偶校验位，以便所有数据位（包括奇偶校验位）的总数返回一个信号状态为“1”的偶数值。	• 无 • 奇数 • 偶数	偶数
响应时间	允许从站进行响应的时间。	50 ms 至 655,000 ms	2,000 ms
模式	“常规操作” “噪声抑制”	• 常规 • 噪声抑制	常规
字符延迟时间倍增器	使用字符延迟时间倍增器（范围为 1 至 10）。	1 至 10	1
启动时清除串行接口接收缓冲器	指定当 CPU 从 STOP 更改为 RUN 模式（CPU 启动）时串行接口的接收缓冲器是否应自动清除。通过这种方式，可以确保串行接口的接收缓冲器仅包含 CPU 启动后接收的消息帧。	• 否 • 是	是

- **全双工 (RS 422) 四线制操作**

  在这种操作模式下，数据通过传输线路 T(A)、T(B) 发送，通过接收线路 R(A)、R(B) 接收。按照通过“驱动程序操作模式”参数设置的功能（常规或噪声抑制）执行故障诊断。

- **半双工 (RS 485) 两线制操作**

  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两线制接收线路 R(A)、R(B)。通过正确接收的从站地址检测来自从站的接收消息帧的开始。进行点对点通讯时，建议将设置 R (A) 0 V、R(B) 5 V 作为接收线路的初始状态。

- **接收线路初始状态**

  该参数指定了 RS 422 和 RS 485 模式下接收线路的初始状态。但它不用于 RS 232C 模式。

  – **R(A) 5 V、R(B) 0 V (BREAK)**
    模块按照如下所示设置两线制线路 R(A)、R(B) 的初始状态：
    R(A) --> +5 V，R(B) --> 0 V ($V_A - V_B$ = +0.3 V)。
    这意味着模块上的 BREAK 电平出现了线路断路。

  – **R(A) 0 V、R(B) 5 V (High)**
    模块按照如下所示设置两线制线路 R(A)、R(B) 的初始状态：
    R(A) --> 0 V、R(B) --> +5 V ($V_A - V_B$ = -0.3 V)。
    这意味着模块上的 HIGH 电平出现了线路断路（如果没有正在传输的从站，则意味着主站处于空闲状态）。无法检测 BREAK 线路状态。

  – **无（仅 RS 485）**
    多点连接的接收线路初始状态关闭。

- **传输率**

  最大传输率是指数据传输速度，单位是位/秒 (bps)。进行半双工操作时，模块的最大传输率为 38,400 bps。

- **数据位**

  数据位数说明了要传输的一个字符对应的位数。该设置必须始终是 8 个数据位。必须始终使用 11 位的字符帧。如果您将奇偶校验设置为“无”(none)，则必须选择 2 个停止位。

- **停止位**

  停止位数定义了要传输的两个字符之间的最小可能时间间隔。必须始终使用 11 位的字符帧。如果您将奇偶校验设置为“无”(none)，则必须选择 2 个停止位。

- **奇偶校验**

  奇偶校验位有助于确保数据的完整性。根据该设置，它会补充要传输的数据位数，使其为一个偶数或奇数。如果奇偶校验已设置为“无”(none)，则不传输奇偶校验位。这会降低传输的完整性。必须始终使用 11 位的字符帧。如果您将奇偶校验设置为“无”(none)，则必须选择 2 个停止位。

- **响应时间**

  响应监视时间是指主站在输出请求消息帧之后等待来自从站的响应消息帧所花费的时间。

- **常规操作**

  在此操作模式下，在来自从站的接收消息帧之前和之后检测到的所有传输错误或 BREAK 都会产生相应的错误消息。

- **噪声抑制**

  如果在接收消息帧开始时检测到接收线路上发生 BREAK，或模块接口微处理器检测到传输错误，则该驱动程序会认为收到的消息有错误并忽略它。通过正确接收的从站地址检测来自从站的接收消息帧的开始。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。

- **字符延迟时间倍增器**

  如果通讯伙伴无法满足 Modbus 规范的时间要求，则可以将字符延迟时间 t_{ZVZ} 与倍增因子 f_{MUL} 相乘。如果通讯伙伴无法满足要求的时间，则仅调整字符延迟时间即可。产生的字符延迟时间 t_{ZVZ} 计算方法如下：

$t_{ZVZ} = t_{ZVZ\_TAB} * f_{MUL}$；

t_{ZVZ_TAB}：	t_{ZVZ} 的表值
f_{MUL}：	倍增因子

说明

请参见标识数据（页码 63）和固件更新的后续装载（页码 65）中涉及的主题。

3.5.4 Modbus 主站使用的功能代码

功能代码表

下表列出了 Modbus 主站驱动程序支持的功能代码。

表格 3-19 Modbus 主站驱动程序的参数

功能代码	说明	SIMATIC S7 中的功能	
01	读取输出状态	逐位读取	标志 F
		逐位读取	输出 Q
		逐位读取 （16 位间隔）	定时器 T
		逐位读取 （16 位间隔）	计数器 C
02	读取输入状态	逐位读取	标志 F
		逐位读取	输入 I
03	读取输出寄存器	逐字读取	数据块 DB
04	读取输入寄存器	逐字读取	数据块 DB
05	强制单个线圈	逐位写入	标志 F
		逐位写入	输出 Q
06	预设单个寄存器	逐字写入	数据块 DB
07	读取异常状态	逐位读取	8 位状态
08	回送诊断测试	-	-
11	获取通讯事件计数器	读取 2 个字	事件状态和计数器
12	获取通讯事件日志	读取 70 个字节	事件日志
15	强制多个线圈	逐位写入 （1 至 2,040 位）	标志 F
		逐位写入 （1 至 2,040 位）	输出 Q
16	预设多个寄存器	逐字写入 （1 至 127 个寄存器）	数据块 DB

3.5.5 功能代码 01 — 读取输出状态

用途和结构

功能	使用该功能可以自从站读取各个位。
起始地址	位起始地址参数不通过该驱动程序检查，并且发送时不做任何更改。
位数	1 至 2,040 之间的任何值都可以作为位数（线圈数）。
LEN（单位：字节）	6

SEND 源 DB

下表显示了 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	位起始地址	WORD	W#16#0040	位起始地址
+4.0	位数	INT	16	位数

RCV 目标 DB

下表显示了 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#1701	数据

该驱动程序将响应消息帧数据逐字写入目标 DB。收到的第一个字节作为第一个字“data[1]”的低位字节输入，收到的第三个字节作为第二个字“data[2]”的低位字节输入，依此类推。如果读取的位数少于 9 或仅读取了一个低位字节，则在最后一个字的剩余高位字节中输入值 00H。

3.5.6 功能代码 02 — 读取输入状态

用途和结构

功能	使用该功能可以自从站读取各个位。
起始地址	位起始地址参数不通过该驱动程序检查，并且发送时不做任何更改。
位数	1 至 2040 之间的任何值都可以作为位数（线圈数）。
LEN（单位：字节）	6

SEND 源 DB

下表显示了 SEND 源区域的结构：

地址	名称	类型	起始值	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	位起始地址	WORD	W#16#0120	位起始地址
+4.0	位数	INT	24	位数

RCV 目标 DB

下表显示了 RCV 源区域的结构：

地址	名称	类型	实际值	注释
+0.0	Data[1]	WORD	W#16#2604	数据
+2.0	Data[2]	WORD	W#16#0048	数据

该驱动程序将回复消息数据逐字写入目标 DB。收到的第一个字节作为第一个字 Data[1] 的低位字节输入，收到的第三个字节作为第二个字 Data[2] 的低位字节输入，依此类推。

如果读取的位数少于 9 或仅读取了一个低位字节，则在最后一个字的剩余高位字节中输入值 00H。

3.5.7 功能代码 03 — 读取输出寄存器

用途和结构

功能	使用该功能可以自从站读取各个寄存器。
起始地址	寄存器起始地址参数不通过该驱动程序检查，并且发送时不做任何更改。
位数	最多可以读取 127 个寄存器（1 个寄存器 = 2 个字节）。
LEN（单位：字节）	6

SEND 源 DB

下表显示了 SEND 源区域的结构：

地址	名称	类型	起始值	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	寄存器起始地址	WORD	W#16#0040	寄存器起始地址
+4.0	寄存器数	INT	2	寄存器数

RCV 目标 DB

下表显示了 RCV 源区域的结构：

地址	名称	类型	实际值	注释
+0.0	Data[1]	WORD	W#16#2123	数据
+2.0	Data[2]	WORD	W#16#2527	数据

3.5.8 功能代码 04 — 读取输入寄存器

用途和结构

功能	使用该功能可以自从站读取各个寄存器。
起始地址	寄存器起始地址参数不通过该驱动程序检查，并且发送时不做任何更改。
位数	最多可以读取 127 个寄存器（1 个寄存器 = 2 个字节）。
LEN（单位：字节）	6

SEND 源 DB

下表显示了 SEND 源区域的结构：

地址	名称	类型	起始值	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	寄存器起始地址	WORD	W#16#0050	寄存器起始地址
+4.0	寄存器数	INT	3	寄存器数

RCV 目标 DB

下表显示了 RCV 源区域的结构：

地址	名称	类型	实际值	注释
+0.0	Data[1]	WORD	W#16#2123	数据
+2.0	Data[2]	WORD	W#16#2527	数据
+4.0	Data[3]	WORD	W#16#3536	数据

3.5.9 功能代码 05 — 强制单个线圈

用途和结构

功能	使用该功能可以设置或删除从站中的各个位。
位地址	位地址参数不通过该驱动程序检查，并且发送时不做任何更改。
位状态	以下两个值可以作为位状态： FF00H => 设置位 0000H => 删除位
LEN（单位：字节）	6

SEND 源 DB

下表显示了 SEND 源区域的结构：

地址	名称	类型	起始值	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	位地址	WORD	W#16#0019	位地址
+4.0	位状态	WORD	W#16#FF00	位状态

从站必须将请求消息原样返回主站（回复）。

RCV 目标 DB

下表显示了 RCV 源区域的结构：

地址	名称	类型	实际值	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	位地址	WORD	W#16#0019	位地址
+4.0	位状态	WORD	W#16#FF00	位状态

3.5.10 功能代码 06 — 预设单个寄存器

用途和结构

功能	使用该命令可以用新值覆盖从站寄存器。
寄存器地址	寄存器地址参数不通过该驱动程序检查，并且发送时不做任何更改。
寄存器值	任何值都可以用作寄存器值。
LEN（单位：字节）	6

SEND 源 DB

下表显示了 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	寄存器地址	WORD	W#16#0180	寄存器地址
+4.0	寄存器值	WORD	W#16#3E7F	寄存器值

RCV 目标 DB

下表显示了 RCV 目标区域的内容：

地址	名称	类型	当前值	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	寄存器地址	WORD	W#16#0180	寄存器地址
+4.0	寄存器值	WORD	W#16#3E7F	寄存器值

3.5.11 功能代码 07 — 读取异常状态

用途和结构

功能 使用该功能代码可以自连接的从站读取 8 个事件位。事件位的起始位号由连接的设备确定，因此并不一定通过 SIMATIC 用户程序指定。

LEN（单位：字节） 2

SEND 源 DB

下表显示了 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

RCV 目标 DB

下表显示了 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

该驱动程序将响应消息帧的各个位输入到目标 DB data[1] 的高位字节。data[1] 的低位字节保持不变。将显示值 1 作为 LEN 参数中的长度。接收长度始终为 1。

3.5.12 功能代码 08 — 回送诊断测试

用途和结构

功能	使用该功能可以检查通讯连接。该功能代码仅支持诊断代码 0000。
诊断代码	只允许使用 0000 作为诊断代码参数的值。
测试值	任何值都可以用作测试值。
LEN（单位：字节）	6

SEND 源 DB

下表显示了 SEND 源区域的结构：

地址	名称	类型	起始值	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	诊断代码	WORD	B#16#0000	诊断代码
+4.0	注册值	WORD	B#16#A5C3	测试值

RCV 目标 DB

下表显示了 RCV 源区域的结构：

地址	名称	类型	实际值	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	诊断代码	WORD	B#16#0000	诊断代码
+4.0	测试值	WORD	B#16#A5C3	测试值

3.5.13 功能代码 11 — 获取通讯事件计数器

用途和结构

功能 使用该功能代码可以自从站读取状态字（2 个字节长）和事件计数器（2 个字节长）。

LEN（单位：字节） 2

SEND 源 DB

下表显示了 SEND 源区域的结构：

地址	名称	类型	起始值	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

RCV 目标 DB

下表显示了 RCV 源区域的结构：

地址	名称	类型	实际值	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

3.5.14 功能代码 12 — 获取通讯事件日志

用途和结构

功能 使用该功能代码可以自从站读取以下内容：
-- 2 个字节的状态字
-- 2 个字节的事件计数器
-- 2 个字节的消息计数器
-- 64 个字节的事件字节

LEN（单位：字节） 2

SEND 源 DB

下表显示了 SEND 源区域的结构：

地址	名称	类型	起始值	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

RCV 目标 DB

下表显示了 RCV 源区域的结构：

地址	名称	类型	实际值	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

3.5.15 功能代码 15 — 强制多个线圈

用途和结构

功能	使用该功能代码最多可以在从站中更改 2,040 位。
起始地址	位起始地址参数不通过该驱动程序检查，并且发送时不做任何更改。
位数	1 至 2,040 之间的任何值都可以作为位数（线圈数）。这指定了从站中将被覆盖的位数？该驱动程序根据传输的“位数”参数生成请求消息帧中的“字节计数器”参数。
LEN（单位：字节）	> 6

SEND 源 DB

下表显示了 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	位起始地址	WORD	W#16#0058	位起始地址
+4.0	位数	INT	10	位数
+6.0	coil_state[1]	WORD	W#16#EFCD	状态线圈 5FH..58H/57H..50H

RCV 目标 DB

下表显示了 RCV 目标区域的内容：

地址	名称	类型	当前值	注释
+0.0	地址	BYTE	B#16#5	从站地址
+1.0	功能	BYTE	B#16#F	功能代码
+2.0	位地址	WORD	W#16#0058	位地址
+4.0	位数	INT	10	位数

该驱动程序从源目标 DB 逐字发送数据。首先发送 DB 中的高位字节（字节 1）或字地址“EF”，然后发送 DB 字地址“CD”的低位字节（字节 0）。如果发送了奇数个字节，则最后一个字节是高位字节（字节 1）。

3.5.16 功能代码 16 — 预设多个寄存器

用途和结构

功能 使用功能代码 16 可以用一个请求消息帧在从站中覆盖最多 127 个寄存器。

起始地址 寄存器起始地址参数不通过该驱动程序检查，并且发送时不做任何更改。

寄存器数 最多可以读取 127 个寄存器（1 个寄存器 = 2 个字节）。该驱动程序根据传输的“寄存器数”参数生成请求消息帧中的“字节计数器”参数。

LEN（单位：字节） > 6

SEND 源 DB

下表显示了 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	寄存器起始地址	WORD	W#16#0060	寄存器起始地址
+4.0	寄存器数	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

RCV 目标 DB

下表显示了 RCV 目标区域的内容：

地址	名称	类型	当前值	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	寄存器起始地址	WORD	W#16#0060	寄存器起始地址
+4.0	寄存器数	INT	3	寄存器数

3.6 Modbus 从站驱动程序

3.6.1 主站-从站连接的组件

简介

该驱动程序与相应的功能块一起以支持 Modbus 的系统的形式在 Modbus 主站控制系统和 ET 200s Modbus 从站通讯模块之间建立通讯连接。

数据传输原理

使用 RTU 格式的 Modbus 协议进行传输。根据主站-从站原理进行数据传输。主站在传输期间被初始化，以便模块和 S7 CPU 作为从站操作。功能代码 01、02、03、04、05、06、08、15 和 16 可用于模块和主站系统间的通讯。主站请求消息帧中的 Modbus 地址通过类似 S7 这样的驱动程序来评估。即，可以从 S7 CPU 中读取以下内容：

- 读取和写入标志、输出和数据块
- 读取标志、输入、定时器和计数器

现有的连接使 Modbus 协议可以对 SIMATIC S7 CPU 的特定存储区进行数据访问。

数据结构

在对 S7 数据结构进行项目组态之前，必须确保数据与 Modbus 主站系统的用户程序兼容。

Modbus 从站连接

模块的 Modbus 从站连接包含两部分：

- Modbus 从站驱动程序
- SIMATIC S7 CPU 的 Modbus 通讯功能块

Modbus 从站通讯 FB

除了 Modbus 从站驱动程序之外，Modbus 从站连接需要 S7 CPU 中的一个特殊通讯 FB。

该 Modbus 通讯 FB 用于处理连接所需的所有功能。

FB81 (S_MODB) 接收 Modbus 协议并将 Modbus 地址转换至 SIMATIC 存储区。

FB81 必须在用户程序的循环程序中调用。该 Modbus 通讯 FB 使用背景数据块作为工作区。

3.6.2 ET 200S Modbus 从站的数据传输

传输顺序

必须在用户程序中循环激活 S_MODB FB，才能执行 Modbus 从站作业。S_MODB 从 ET 200S 1SI 串行接口模块接收作业，然后执行该作业并将响应返回给模块。CPU 和模块间的通讯由 S_SEND 和 S_RCV 功能块（通过 S_MODB 调用）执行。

每次暖启动 CPU 后，用户程序都必须初始化该 Modbus 通讯 FB。通过 CP_START 输入处的上升沿激活初始化。该 FB 将 CPU 的 I、Q、F、T 和 C 地址区的大小记录在其背景数据块中。成功完成初始化后，该 FB 将设置 CP_START_OK 输出。

初始化错误由 CP_START_ERROR 输出指定。在这种情况下，无法实现 Modbus 通讯，并且在响应 Modbus 主站的任何请求时都会生成例外代码消息。

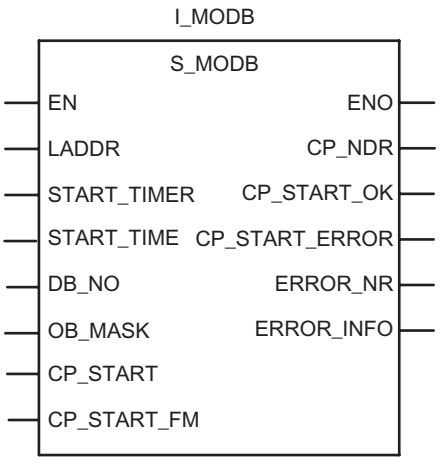
S_MODB 使用位于数据块中的 Modbus 数据转换表将 Modbus 地址映射到 SIMATIC S7 存储区中。

可以使用 OB_MASK 输入参数引导该 Modbus FB 避免 I/O 访问错误。对不存在的 I/O 进行写访问时，CPU 不切换到 STOP 模式，也不调用错误 OB。该 FB 检测访问错误，并且该功能会因对 Modbus 主站的错误响应而终止。

STL 表达式

```
CALL                    S_MODB,
                        I_MODB
      LADDR              =
      START_TIMER        =
      START_TIME         =
      DB_NO              =
      OB_MASK            =
      CP_START           =
      CP_START_FM        =
      CP_NDR             =
      CP_START_OK        =
      CP_START_ERROR     =
      ERROR_NR           =
      ERROR_INFO         =
```

LAD 表达式



说明

参数 EN 和 ENO 仅存在于图形表达式（LAD 或 FBD）中。编译器使用二进制结果处理这些参数。

如果块终止且未出错，则二进制结果的信号状态将设置为“1”。如果出现错误，则二进制结果将设置为“0”。

3.6.3 SIMATIC CPU 中的数据区

Modbus 数据转换表

消息帧中的 Modbus 地址由 FB81 (S_MODB) 以“类似 S7”的形式解释，并转换至 SIMATIC 存储区。用户可以通过将 DB 作为 FB81 (S_MODB) 的输入进行传输来访问各个 SIMATIC 存储区。（参见下表）

表格 3-20　转换表

地址	名称	类型	初始值	当前值	注释	可用的功能代码
0.0	aaaaa	WORD	W#16#0	W#16#0	Modbus 地址的开头	01, 05, 15
2.0	bbbbb	WORD	W#16#0	W#16#7F7	Modbus 地址的结尾	
4.0	uuuuu	WORD	W#16#0	W#16#1F4	标志	
6.0	ccccc	WORD	W#16#0	W#16#7F8	Modbus 地址的开头	01, 05, 15
8.0	ddddd	WORD	W#16#0	W#16#FEF	Modbus 地址的结尾	
10.0	ooooo	WORD	W#16#0	W#16#15	输出	
12.0	eeeee	WORD	W#16#0	W#16#FF0	Modbus 地址的开头	01, 05, 15
14.0	fffff	WORD	W#16#0	W#16#17E7	Modbus 地址的结尾	
16.0	ttttt	WORD	W#16#0	W#16#28	定时器	
18.0	ggggg	WORD	W#16#0	W#16#17E8	Modbus 地址的开头	01, 05, 15
20.0	hhhhh	WORD	W#16#0	W#16#1FDF	Modbus 地址的结尾	
22.0	zzzzz	WORD	W#16#0	W#16#28	计数器	
24.0	kkkkk	WORD	W#16#0	W#16#1FE0	Modbus 地址的开头	02
26.0	lllll	WORD	W#16#0	W#16#27D7	Modbus 地址的结尾	02
28.0	vvvvv	WORD	W#16#0	W#16#320	标志	02
30.0	nnnnn	WORD	W#16#0	W#16#27D8	Modbus 地址的开头	02
32.0	rrrrr	WORD	W#16#0	W#16#2FCF	Modbus 地址的结尾	02
34.0	sssss	WORD	W#16#0	W#16#11	输入	02
36.0	DB_Number_FC_03_06_16	WORD	W#16#0	W#16#6	DB	03, 06, 15
38.0	DB_Number_FC_04	WORD	W#16#0	W#16#2	DB	04
40.0	DB_Min	WORD	W#16#0	W#16#1	使用的最小 DB 号	限制
42.0	DB_Max	WORD	W#16#0	W#16#6	使用的最大 DB 号	限制
44.0	F_Min	WORD	W#16#0	W#16#1F4	使用的最小标志	限制
46.0	F_Max	WORD	W#16#0	W#16#4B0	使用的最大标志	限制
48.0	Q_Min	WORD	W#16#0	W#16#0	使用的最小输出	限制
50.0	Q_Max	WORD	W#16#0	W#16#64	使用的最大输出	限制

3.6.4 为数据链接组态参数

硬件组态的参数

必须在该驱动程序的硬件组态中设置以下参数和操作模式。

- 传输率、奇偶校验
- 模块的从站地址
- 操作模式（常规、噪声抑制）
- 字符延迟时间的倍增因子

FB81 的输入 DB 处的参数

必须使用 FB81 (S_MODB) 的输入 DB 设置下列参数。

- 功能代码 01、05、15 的地址区
- 功能代码 02 的地址区
- 功能代码 03、06、16 的基本 DB 号
- 功能代码 04 的基本 DB 号
- 写访问的限制

参数化从站驱动程序

下表列出了可以为模块的 Modbus 驱动程序设置的参数。

表格 3-21 Modbus 从站驱动程序的参数

参数	说明	值范围	缺省值
诊断中断	指定在出现严重错误时，模块是否应生成诊断中断。	• 否 • 是	否
激活 BREAK 检测	如果发生线路断路或未连接接口电缆，该模块将产生错误消息“断路”。	• 否 • 是	否
接收线路的半双工和全双工初始状态	指定 RS 422 和 RS 485 操作模式下接收线路的初始状态。不是在 RS 232C 操作模式下使用的初始状态。 仅在更换零件时需要确保兼容性的情况下，才需要“反转信号电平”设置。	RS 422： R(A) 5 V/R(B) 0 V (BREAK) R(A) 0 V/R(B) 5 V 反转信号电平 RS 485： 无 R(A) 0 V/R(B) 5 V	RS 422： R(A) 5 V/R(B) 0 V (BREAK) RS 485： R(A) 0 V/R(B) 5 V

参数	说明	值范围	缺省值
数据流量控制 （使用缺省参数；在用户程序中更改缺省值）	您可以发送和接收具有数据流量控制的数据。如果一个通讯伙伴比另一个运行快，则可以通过数据流量控制使数据传输同步。选择数据流量控制类型并设置相关参数。 注意：使用 RS 485 接口无法进行数据流量控制。仅 RS 232C 接口支持使用“V.24 信号的自动控制”进行数据流量控制。	• 无 • V.24 信号的自动控制	无
传输率	选择数据传输率（单位：位/秒）。	• 110 • 300 • 600 • 1,200 • 2,400 • 4,800 • 9,600 • 19,200 • 38,400 • 57,600 • 76,800 • 115,200	9,600
停止位	选择数据传输期间附加到每个字符的停止位数，以指示字符的结束。	• 1 • 2	1
奇偶校验	可以将数据位序列扩展一个字符以包括奇偶校验位。附加值（0 或 1）将所有位（数据位和奇偶校验位）的值设置为已定义状态。 **无**：发送的数据不包含奇偶校验位。 **奇数**：设置奇偶校验位，以便所有数据位（包括奇偶校验位）的总数返回一个信号状态为“1”的奇数值。 **偶数**：设置奇偶校验位，以便所有数据位（包括奇偶校验位）的总数返回一个信号状态为“1”的偶数值。	• 无 • 奇数 • 偶数	偶数
从站地址	模块自身的从站地址	1-247	222
模式	• 常规操作 • 噪声抑制	• 常规 • 噪声抑制	常规
字符延迟时间倍增器	使用字符延迟时间倍增器（范围为 1 至 10）。	1 至 10	1
启动时清除串行接口接收缓冲器	指定当 CPU 从 STOP 更改为 RUN 模式（CPU 启动）时串行接口的接收缓冲器是否应自动清除。通过这种方式，可以确保串行接口的接收缓冲器仅包含 CPU 启动后接收的消息帧。	• 否 • 是	是
[1] 最小字符延迟时间取决于波特率。			

以下列出了各个参数或值的说明：

- **全双工 (RS 422) 四线制操作**

  在这种操作模式下，数据通过传输线路 T(A)、T(B) 发送，通过接收线路 R(A)、R(B) 接收。按照通过“驱动程序操作模式”参数设置的功能（常规或噪声抑制）执行故障诊断。

- **半双工 (RS 485) 两线制操作**

  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两线制接收线路 R(A)、R(B)。通过正确接收的从站地址检测来自从站的接收消息帧的开始。进行点对点通讯时，建议将设置 R (A) 0 V、R(B) 5 V 作为接收线路的初始状态。

- **接收线路初始状态**

  该参数指定了 RS 422 和 RS 485 模式下接收线路的初始状态。但它不用于 RS 232C 模式。

  – **R(A) 5 V、R(B) 0 V (BREAK)**
  模块按照如下所示设置两线制线路 R(A)、R(B) 的初始状态：
  R(A) --> +5 V、R(B) --> 0 V ($V_A - V_B \geq +0.3$ V)。
  这意味着模块上的 BREAK 电平出现了线路断路。

  – **R(A) 0 V、R(B) 5 V (High)**
  模块按照如下所示设置两线制线路 R(A)、R(B) 的初始状态：
  R(A) --> 0 V、R(B) --> +5 V ($V_A - V_B \leq -0.3$ V)。
  这意味着模块上的 HIGH 电平出现了线路断路（如果没有正在传输的从站，则意味着主站处于空闲状态）。无法检测 BREAK 线路状态。

  – 无（仅 RS 485）
  多点连接的接收线路初始状态关闭。

- **传输率**

  传输率是指数据传输速度，单位是位/秒 (bps)。进行半双工操作时，模块的传输率为 38,400 bps。

- **数据位**

  数据位数说明了要传输的一个字符对应的位数。对于该驱动程序，该设置必须始终为 8 个数据位。必须始终使用 11 位的字符帧。如果您将奇偶校验设置为“无”(none)，则必须选择 2 个停止位。

- **停止位**

  停止位数定义了要传输的两个字符之间的最小可能时间间隔。必须始终使用 11 位的字符帧。如果您将奇偶校验设置为“无”(none)，则必须选择 2 个停止位。

- **奇偶校验**

  奇偶校验位有助于确保数据的完整性。根据该设置，它会补充要传输的数据位数，使其为一个偶数或奇数。如果奇偶校验已设置为“无”(none)，则不传输奇偶校验位。这会降低传输的完整性。必须始终使用 11 位的字符帧。如果您将奇偶校验设置为“无”(none)，则必须选择 2 个停止位。

- **从站地址**

  这里指定了模块应该响应的单独 Modbus 从站地址。模块仅响应收到的从站地址与参数化的从站地址相同的消息帧。不检查其它从站的消息帧，并且不接收响应。

- **常规操作**

  在此操作模式下，在来自从站的接收消息帧之前和之后检测到的所有传输错误或 BREAK 都会产生相应的错误消息。

- **噪声抑制**

  如果在接收消息帧开始时检测到接收线路上发生 BREAK，或模块接口微处理器检测到传输错误，则该驱动程序会认为收到的消息有错误并忽略它。通过正确接收的从站地址检测来自从站的接收消息帧的开始。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。

- **字符延迟时间倍增器**

  如果通讯伙伴无法满足 Modbus 规范的时间要求，则可以将字符延迟时间 t_{ZVZ} 与倍增因子 f_{MUL} 相乘。仅当通讯伙伴无法满足要求的时间，才应调整字符延迟时间。

  产生的字符延迟时间 t_{ZVZ} 计算方法如下：

$$t_{ZVZ} = t_{ZVZ\_TAB} * f_{MUL}$$

t_{ZVZ_TAB} = t_{ZVZ} 的表值

f_{MUL} = 倍增因子

说明

请参见标识数据（页码 63）和固件更新的后续装载（页码 65）中涉及的主题。

3.6.5 从站功能代码

Modbus 从站驱动程序功能代码

Modbus 从站驱动程序支持下表所列的功能代码。

说明

下表所列的所有 Modbus 地址都适用于传输消息帧层（并非 Modbus 主站系统中的用户层）。这意味着传输消息帧中的 Modbus 地址以 0000（十六进制）开头。

表格 3-22 从站功能代码

功能代码	说明	SIMATIC S7 中的功能	
01	读取线圈状态	逐位读取	标志 F
		逐位读取	输出 Q
		逐位读取 （16 位间隔）	定时器 T
		逐位读取 （16 位间隔）	计数器 C
02	读取输入状态	逐位读取	标志 F
		逐位读取	输入 I
03	读取保持寄存器	逐字读取	数据块 DB
04	读取输入寄存器	逐字读取	数据块 DB
05	强制单个线圈	逐位写入	标志 F
		逐位写入	输出 Q
06	预设单个寄存器	逐字写入	数据块 DB
08	回送测试	-	-
15	强制多个线圈	逐位写入（1 至 2,040 位）	标志 F
		逐位写入（1 至 2,040 位）	输出 Q
16	预设多个（保持）寄存器	逐字写入 （1 至 127 个寄存器）	数据块 DB

3.6.6 功能代码 01 — 读取线圈（输出）状态

用途和结构

功能代码 01 — 读取线圈（输出）状态的特性如下：

功能	该功能使 Modbus 主站系统可以从下列 SIMATIC 存储区读取各个位。				
请求消息帧	ADDR	FUNC	start_address	bit_number	CRC
响应消息帧	ADDR	FUNC	start_address	n 个字节的 DATA	CRC
LEN（单位：字节）	6				

start_address

该驱动程序解释了 Modbus 位地址“start_address”。示例：FB81 (S_MODB) 检查“start_address”是否位于 FC 01、05、15 的转换 DB 指定的区域之一（从/至：标志、输出、定时器、计数器）。

如果 Modbus 位地址“start_address”位于此区域中	**则访问以下 SIMATIC 存储区**	
从 *aaaaa* 到 *bbbbb*	起始标志	F *uuuuu*.0
从 *ccccc* 到 *ddddd*	起始输出	Q *ooooo*.0
从 *eeeee* 到 *fffff*	起始定时器	T *ttttt*
从 *ggggg* 到 *hhhhh*	起始计数器	C *zzzzz*

访问地址（地址转换）如下计算：

访问开始处（使用 SIMATIC）	**转换公式**	
标志字节	= ((start_address - *aaaaa*)/8)	+ *uuuuu*
输出字节	= ((start_address - *ccccc*)/8)	+ *ooooo*
定时器	=((start_address - *eeeee*)/16)	+ *ttttt*
计数器	=((start_address - *ggggg*)/16)	+ *zzzzz*

访问标志和输出

当访问 SIMATIC 标志和输出区域时，计算剩余的 bit_number 并使用它对第一个/最后一个标志或输出字节中相应的位进行寻址。

访问定时器和计数器

当计算该地址时，结果

- (start_address – eeeee) 或
- (start_address – ggggg)

必须可以被 16 整除（仅限于逐字访问，从字限制开始）。

bit_number

1 到 2,040 之间的值都可以作为 bit_number（线圈数）。读取该位数。

当访问 SIMATIC 定时器和计数器区域时，“bit_number”必须可以被 16 整除（仅限于逐字访问）。

应用示例

表格 3-23　转换 Modbus 寻址的示例：

转换功能代码 FC 01、05、15 的 Modbus 寻址	
传输消息帧中的 Modbus 地址	SIMATIC 存储区
从 *0* 到 *2,047*	从标志 F *1000*.0 开始
从 *2,048* 到 *2,559*	从输出 Q *256*.0 开始
从 *4,096* 到 *4,607*	从定时器 T *100* 开始
从 *4,608* 到 *5,119*	从计数器 C *200* 开始

SEND 源 DB

表格 3-24　下表显示了 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	位起始地址	WORD	W#16#0040	位起始地址
+4.0	位数	INT	16	位数

RCV 目标 DB

表格 3-25　下表显示了 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#1701	数据

该驱动程序将响应消息帧数据逐字写入目标 DB。收到的第一个字节作为第一个字“data[1]”的低位字节输入，收到的第三个字节作为第二个字“data[2]”的低位字节输入，依此类推。如果读取的位数少于 9 或仅读取了一个低位字节，则在最后一个字的剩余高位字节中输入值 00H。

地址计算：

表格 3-26　Modbus 地址“start_address”0040 十六进制（64 十进制）位于标志区域中

标志字节	= ((start_address - *aaaaa*)	/ 8)	+ *uuuuu*
	=((64 – *0*)	/ 8)	+ *1,000*
	=1,008;		

表格 3-27　剩余的 bit_number 的结果如下

剩余的 bit_no.	= ((start_address - *aaaaa*)	% 8)	[模数为 8]
	=((64 - *0*)	% 8)	
	= 0;		

访问标志 F 1008.0 到 F 1011.7（含）。

位数：

Modbus 位“bit_number”0020 十六进制（32 十进制）的数目表示要读取 32 位（4 个字节）。下表列出了数据访问的其它示例。

表格 3-28　　数据访问的其它示例

start_address： 十六进制、十进制		**地址计算**	**地址**
0000	0	标志 ((0 – *0*)/8) + *1,000*	-> F 1000.0
0021	33	标志 ((33 – *0*)/8) + *1,000*	-> F 1004.1
0400	1024	标志 ((1,024 – *0*)/8) + *1,000*	-> F 1128.0
0606	1542	标志 ((1,542 – *0*)/8) + *1,000*	-> F 1192.6
0840	2112	输出 ((2,112 – *2,048*)/8) + *256*	-> Q 264.0
09E4	2532	输出 ((2,532 – *2,048*)/8) + *256*	-> Q 316.4
1010	4112	定时器 ((4,112 – *4,096*)/16) + *100*	-> T 101
10C0	4288	定时器 ((4,288 – *4,096*)/16) + *100*	-> T 112
1200	4608	计数器 ((4,608 – *4,608*)/16) + *200*	-> C 200
13E0	5088	计数器 ((5,088 – *4,608*)/16) + *200*	-> C 230

3.6.7 功能代码 02 — 读取输入状态

用途和结构

功能代码 02 — 读取输入状态的特性如下：

功能	该功能使 Modbus 主站系统可以从下列 SIMATIC 存储区读取各个位。				
请求消息帧	ADDR	FUNC	start_address	bit_number	CRC
响应消息帧	ADDR	FUNC	Byte_count n	n 个字节的 DATA	CRC
LEN（单位：字节）	6				

start_address

该驱动程序对 Modbus 位地址“start_address”的解释如下：

该驱动程序检查“start_address”是否位于 FC 02 的转换 DB 中输入的区域之一（从/至：标志、输入）。

如果 Modbus 位地址“start_address”位于此区域中	**则访问以下 SIMATIC 存储区**	
从 *kkkkk* 到 *lllll*	起始标志	F *vvvvv*.0
从 *nnnnn* 到 *rrrrr*	起始输入	I *sssss*. 0

访问地址（地址转换）如下计算：

访问开始处（使用 SIMATIC）	**转换公式**	
标志字节	= ((start_address - *kkkkk*)/8)	+ *vvvvv*
输入字节	= ((start_address - *nnnnn*)/8)	+ *sssss*

访问标志和输入

当访问 SIMATIC 标志和输入区域时，计算剩余的 bit_number 并使用它对第一个/最后一个标志或输入字节中相应的位进行寻址。

bit_number

1 到 2,040 之间的任何值都可以作为 bit_number（线圈数）。读取该位数。

应用示例

转换 Modbus 地址分配的示例：

表格 3-29　转换功能代码 FC 02 的 Modbus 寻址

传输消息帧中的 Modbus 地址	**SIMATIC 存储区**	
从 *0* 到 *4,095*	起始标志	F *2000*.0
从 *4,096* 到 *5,119*	起始输入 0	I *128*

SEND 源 DB

表格 3-30　下表显示了 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	位起始地址	WORD	W#16#0120	位起始地址
+4.0	位数	INT	24	位数

RCV 目标 DB

表格 3-31　下表显示了 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#2604	数据
+2.0	Data[2]	WORD	W#16#0048	数据

该驱动程序将响应消息帧数据逐字写入目标 DB。收到的第一个字节作为第一个字“data[1]”的低位字节输入，收到的第三个字节作为第二个字“data[2]”的低位字节输入，依此类推。

如果读取的位数少于 9 或仅读取了一个低位字节，则在最后一个字的剩余高位字节中输入值 00H。

地址计算：

Modbus 地址“start_address”1030 十六进制（4144 十进制）位于输入区域中：

输入字节	=((start_address - *nnnnn*)	/ 8)	+ *sssss*
	=((4,144 - *4,096*)	/ 8)	+ *128*
	=134;		

剩余的 bit_number 的结果如下：

剩余的 bit_no.	= ((start_address - *aaaaa*)	% 8)	[模数为 8]
	=((4,144 - *4,096*)	% 8)	
	= 0;		

访问输入 I 134.0 到 I 136.7（含）。

位数：

Modbus 位“bit_number”0018 十六进制（24 十进制）的数目表示要读取 24 位（3 个字节）。

下表列出了数据访问的其它示例。

表格 3-32　数据访问的其它示例

start_address： 十六进制、十进制		**地址计算**	**地址**
0000	0	标志 ((0 – *0*)/8) + *2,000*	-> F 2000.0
0071	113	标志 ((113 - *0*)/8) + *2,000*	-> F 2014.1
0800	2048	标志 ((2,048 - *0*)/8) + *2,000*	-> F 2256.0
0D05	3333	标志 ((3,333 - *0*)/8) + *2,000*	-> F 2416.5
1000	4096	输入 ((4,096 - *4,096*)/8) + *128*	-> I 128.0
10A4	4260	输入 ((4,260 - *4,096*)/8) + *128*	-> I 148.4

3.6.8 功能代码 03 — 读取输出寄存器

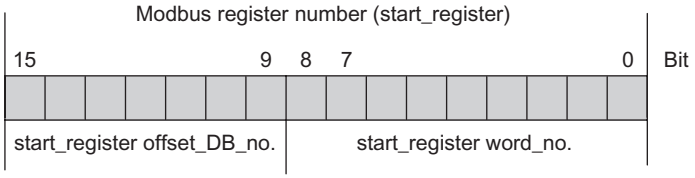
用途和结构

功能代码 03 — 读取输出状态的特性如下：

功能	此功能使 Modbus 主站系统可以从数据块中读取数据字				
请求消息帧	ADDR	FUNC	start_address	register_number	CRC
响应消息帧	ADDR	FUNC	Byte_count n	n/2 寄存器 DATA（高、低）	CRC
LEN（单位：字节）	6				

start_address

该驱动程序对 Modbus 寄存器地址“start_register”的解释如下：



图片 3-11　解释 Modbus 寄存器号

为了进一步生成地址，FB81 (S_MODB) 将使用在 FC 03、06、16 的转换 DB 中输入的基础 DB 号（从 DB xxxxx 开始）。

访问地址（地址转换）分两步计算：

访问 SIMATIC	转换公式
数据块 DB（生成的 DB）	= (基本 DB 号 *xxxxx* + start_register offset_DB_no.)
数据字 DBW	= (start_register word_no. *2

start_register 的计算公式

如果已知要读取的生成的 DB，则可以使用以下公式计算主站系统中所需的 Modbus 地址 start_register：

start_register = ((生成的 DB – 基本 DB 号) * 512) + (data word_DBW/2)

只有偶数数据字编号可用于此公式。

register_number

1 到 127 之间的任何值都可以作为 register_number（寄存器数）。读取该寄存器数。

应用示例

表格 3-33　转换功能代码 FC 03、06、16 的 Modbus 寻址

传输消息帧中的 Modbus 地址	SIMATIC 存储区
0	从数据块 DB *800*（基本 DB 号）开始

SEND 源 DB

表格 3-34　下表显示了 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	寄存器起始地址	WORD	W#16#0040	寄存器起始地址
+4.0	寄存器数	INT	2	寄存器数

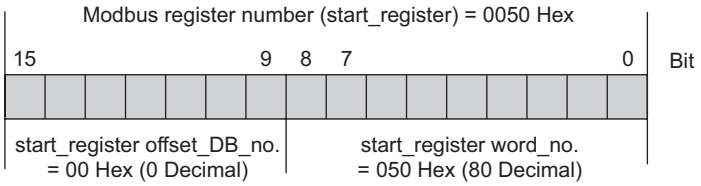
RCV 目标 DB

表格 3-35　下表显示了 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#2123	数据
+2.0	Data[2]	WORD	W#16#2527	数据

地址计算：

Modbus 地址“start_register”0050 十六进制（80 十进制）的解释如下所示：



图片 3-12　　解释 Modbus 寄存器号 0050（十六进制）

数据块 DB（生成的 DB）	= (基本 DB 号 *xxxxx* + start_register - offset_DB_no.)
	= (*800* + 0)
	= 800 ;

数据字 DBW	= (start_register word_no.* 2)
	= (80 * 2)
	= 60 ;

DB800，访问数据字 DBW160。

寄存器数：

Modbus 寄存器数“register_number”0002 十六进制（2 十进制）表示要读取 2 个寄存器（2 个数据字）。

下表列出了数据访问的其它示例。

表格 3-36　　数据访问的其它示例

start_register							
start_register		基本 DB_no.	偏移 DB_no.	字编号		生成的 DB	DBW
十六进制	十进制	十进制	十进制	十六进制	十进制	十进制	十进制
0000	0	800	0	000	0	800	0
01F4	500	800	0	1F4	500	800	1000
0200	512	800	1	000	0	801	0
02FF	767	800	1	0FF	255	801	510
0300	768	800	1	100	256	801	512
03FF	1023	800	1	1FF	511	801	1022
0400	1024	800	2	000	0	802	0

3.6.9 功能代码 04 — 读取输入寄存器

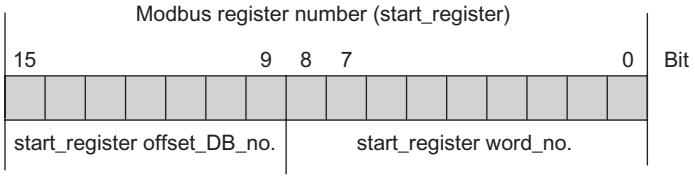
用途和结构

功能代码 04 — 如下读取输入寄存器的特性：

功能	此功能使 Modbus 主站系统可以从数据块中读取数据字				
请求消息帧	ADDR	FUNC	start_register	register_number	CRC
响应消息帧	ADDR	FUNC	Byte_count n	n/2 寄存器 DATA（高、低）	CRC
LEN（单位：字节）	6				

start_address

该驱动程序如下解释 Modbus 寄存器地址“start_register”：



图片 3-13　解释 Modbus 寄存器号

为了生成更多的地址，FB81 (S_MODB) 使用在 FC 04 的转换 DB 中输入的基本 DB 号（从 DB xxxxx 开始）。

访问地址（地址转换）分两步计算：

访问 SIMATIC	**转换公式**
数据块 DB（生成的 DB）	= (基本 DB 号 *xxxxx* + start_register offset_DB_no.)
数据字 DBW	= (start_register word_no.* 2)

start_register 的计算公式

如果已知要读取的生成的 DB，则可以使用以下公式计算主站系统中所需的 Modbus 地址 start_register：

start_register = ((生成的 DB – 基本 DB 号) * 512) + (data word_DBW/2)

只有偶数数据字编号可用于此公式。

register_number

1 到 127 之间的任何值都可以作为 register_number（寄存器数）。读取该寄存器数。

应用示例

表格 3-37　转换功能代码 FC 04 的 Modbus 寻址

传输消息帧中的 Modbus 地址	SIMATIC 存储区
0	从数据块 DB *900*（基本 DB 号）开始

SEND 源 DB

表格 3-38　下表显示了 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	寄存器起始地址	WORD	W#16#0050	寄存器起始地址
+4.0	寄存器数	INT	3	寄存器数

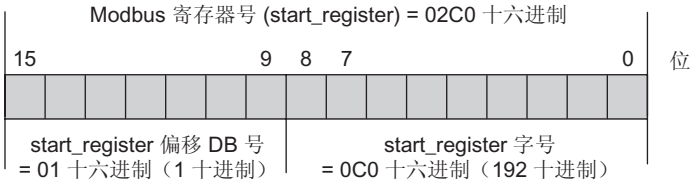
RCV 目标 DB

表格 3-39　下表显示了 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#2123	数据
+2.0	Data[2]	WORD	W#16#2527	数据
+4.0	Data[3]	WORD	W#16#3536	数据

地址计算：

Modbus 地址“start_register”02C0 十六进制（704 十进制）的解释如下：



图片 3-14　　解释 Modbus 寄存器号 0270（十六进制）

数据块 DB（生成的 DB）	= (基本 DB 号 *xxxxx* + start_register - offset_DB_no.)
	= (900+ 0)
	= 901;

数据字 DBW	= (start_register word_no.* 2)
	= (192 * 2)
	= 384;

DB901，访问数据字 DBW384。

寄存器数：

Modbus 寄存器数“register_number”0003 十六进制（3 十进制）表示要读取 3 个寄存器（3 个数据字）。

下表列出了数据访问的其它示例。

表格 3-40　　数据访问的其它示例

start_register							
start_register		基本 DB_no.	偏移 DB_no.	字编号		生成的 DB	DBW
十六进制	十进制	十进制	十进制	十六进制	十进制	十进制	十进制
0000	0	900	0	000	0	900	0
0064	100	900	0	064	100	900	200
00C8	200	900	0	0C8	200	900	400
0190	400	900	0	190	400	900	800
1400	5120	900	10	000	0	910	0
1464	5220	900	10	064	100	910	200
14C8	5320	900	10	0C8	200	910	400

3.6.10 功能代码 05 — 强制单个线圈

用途和结构

功能代码 05 — 强制单个线圈的特性如下：

功能	该功能使 Modbus 主站系统可以向下面列出的 SIMATIC 存储区写入一个位。				
请求消息帧	ADDR	FUNC	coil_address	DATA 开/关	CRC
响应消息帧	ADDR	FUNC	coil_address	DATA 开/关	CRC
LEN（单位：字节）	6				

coil_address

该驱动程序对 Modbus 位地址“coil_address”的解释如下：

FB81 (S_MODB) 检查“coil_address”是否位于在 FC 01、05、15 的转换 DB 中指定的其中一个区域（从/到：标志、输出、定时器、计数器）。

如果 Modbus 位地址“start_address”位于此区域中	**则访问以下 SIMATIC 存储区**	
从 *aaaaa* 到 *bbbbb*	起始标志	F *uuuu*.0
从 *ccccc* 到 *ddddd*	起始输出	Q *oooo*.0

访问地址（地址转换）分两步计算：

访问开始处（使用 SIMATIC）	**转换公式**	
标志字节	= ((start_address *aaaa*)/8)	+ *ooooo*
输出字节	= ((start_address *aaaa*)/8)	+ *uuuuu*

访问标志和输出

当访问 SIMATIC 标志区和输出区时，会计算剩余的 bit_number 并将其用于寻址标志字节或输出字节内的相关位。

访问定时器和计数器

功能代码 FC 05 不允许访问 SIMATIC 定时器区和计数器区，此类访问将被驱动程序拒绝，并输出错误消息帧。

DATA 开/关

在 DATA 开/关时允许以下两个值：

FF00H = 设置位

0000H = 删除位

应用示例

表格 3-41　转换功能代码 FC 01、05、15 的 Modbus 寻址

传输消息帧中的 Modbus 地址	SIMATIC 存储区
从 *0* 到 *2,047*	从标志 F *1000*.0 开始
从 *2,048* 到 *2,559*	从输出 Q *256*.0 开始

SEND 源 DB

表格 3-42　下表显示了 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	位地址	WORD	W#16#0019	位地址
+4.0	位状态	WORD	W#16#FF00	位状态

从站必须将请求消息帧返回未更改的主站（回复）。

RCV 目标 DB

表格 3-43　下表显示了 RCV 目标区域的内容：

地址	名称	类型	当前值	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	位地址	WORD	W#16#0019	位地址
+4.0	位状态	WORD	W#16#FF00	位状态

地址计算：

表格 3-44　Modbus 地址“coil_address”0809 十六进制（2057 十进制）位于输出区中

输出字节	= ((coil_address - *ccccc*)/8)	+ *ooooo*
	=((2,057 - *2,048*) / 8)	+ *256*
	=257	

表格 3-45　剩余的 bit_number 的结果如下

剩余的 bit_no.	= ((coil_address - *ccccc*)) % 8)	[模数为 8]
	=((2,057 -*2,048*) % 8)	
	= 1 ;	

访问输出 Q 257.1。

更多示例

有关访问标志和输出的更多示例，请参考 FC 01。

3.6.11 功能代码 06 — 预设单个寄存器

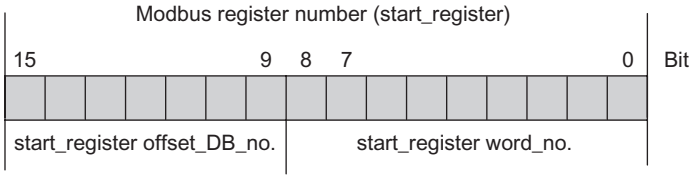
用途和结构

功能代码 06 — 预设单个寄存器的特性如下：

功能	该功能使 Modbus 主站系统能够在 CPU 的数据块中写入一个数据字。				
请求消息帧	ADDR	FUNC	start_register	DATA 值（高、低）	CRC
响应消息帧	ADDR	FUNC	start_register	DATA 值（高、低）	CRC
LEN（单位：字节）	6				

start_register

该驱动程序对 Modbus 寄存器地址“start_register”的解释如下：



图片 3-15　解释 Modbus 寄存器号

为了生成更多的地址，FB81 (S_MODB) 使用在 FC 03、06、16 的转换 DB 中输入的基本 DB 号（从 DB xxxxx 开始）。

访问地址（地址转换）分两步计算：

访问 SIMATIC	转换公式
数据块 DB（生成的 DB）	= (基本 DB 号 xxxxx + start_register - offset_DB_no.)
数据字 DBW	= (start_register word_no.* 2)

如果已知要读取的生成的 DB，则可以使用以下公式计算主站系统中所需的 Modbus 地址 start_register：

start_register = ((生成的 DB – 基本 DB 号) * 512) + (data word_DBW/2)

只有偶数数据编号可以用于此公式。

DATA 值

任何值都可用作 DATA 值（寄存器值）。

参数化的应用示例：

表格 3-46　转换功能代码 FC 03、06、16 的 Modbus 寻址

传输消息帧中的 Modbus 地址	SIMATIC 存储区
0	从数据块（基本 DB 号）DB*800* 开始

SEND 源 DB

表格 3-47　下表显示了 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	寄存器地址	WORD	W#16#0180	寄存器地址
+4.0	寄存器值	WORD	W#16#3E7F	寄存器值

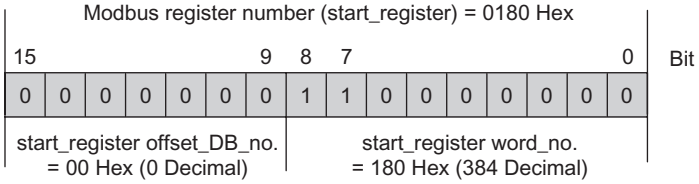
RCV 目标 DB

表格 3-48　下表显示了 RCV 目标区域的内容：

地址	名称	类型	当前值	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	寄存器地址	WORD	W#16#0180	寄存器地址
+4.0	寄存器值	WORD	W#16#3E7F	寄存器值

地址计算：

Modbus 地址“start_register”0180 十六进制（384 十进制）的解释如下所示：



图片 3-16　　解释 Modbus 寄存器号 0180（十六进制）

数据块 DB（生成的 DB）	= (基本 DB 号 *xxxxx* + start_register offset_DB_no.)
	= (*800* + 0)
	= 800 ;

数据字 DBW	= (start_register word_no.* 2)
	= (384 * 2)
	= 768 ;

访问 DB800，数据字 DBW768。

更多示例

可在 FC 03 下找到更多访问示例。

3.6.12 功能代码 08 — 回送诊断测试

用途和结构

功能代码 08 — 回送诊断测试的特性如下：

功能	该功能用于检查通讯连接。它对 S7 CPU、用户程序或用户数据没有影响。接收的消息帧通过驱动程序独立返回主站系统中。				
请求消息帧	ADDR	FUNC	诊断代码（高、低）	测试数据	CRC
响应消息帧	ADDR	FUNC	诊断代码（高、低）	测试数据	CRC
诊断代码	仅支持诊断代码 0000。				
测试数据	任何值（16 位）。				
LEN（单位：字节）	6				

应用示例

SEND 源 DB

下表显示了 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	诊断代码	WORD	B#16#0000	诊断代码
+4.0	寄存器值	WORD	B#16#A5C3	测试值

RCV 目标 DB

下表显示了 RCV 目标区域的内容：

地址	名称	类型	当前值	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	诊断代码	WORD	B#16#0000	诊断代码
+4.0	测试值	WORD	B#16#A5C3	测试值

3.6.13 功能代码 15 — 强制多个线圈

用途和结构

功能代码 15 — 强制多个线圈的特性如下：

功能	该功能使 Modbus 主站系统可以向下列 SIMATIC 存储区写入多个位。
请求消息帧	ADDR FUNC start_address 数量 byte_count N n DATA CRC
响应消息帧	ADDR FUNC start_address n 个字节的 DATA CRC
LEN（单位：字节）	> 6

start_address

该驱动程序对 Modbus 位地址“start_address”的解释如下：

FB81 (S_MODB) 检查“start_address”是否位于在 FC 01、05、15 的转换 DB 中指定的其中一个区域（从/到：标志、输出、定时器、计数器）。

如果 Modbus 位地址“start_address”位于此区域中	则访问以下 SIMATIC 存储区	
从 *aaaaa* 到 *bbbbb*	起始标志	F *uuuu*.0
从 *ccccc* 到 *ddddd*	起始输出	Q *ooooo*.0

访问地址（地址转换）的计算如下所示：

访问开始处（使用 SIMATIC）	转换公式	
标志字节	= ((start_address *aaaa*)/8)	+ *uuuu*
输出字节	= ((start_address - *aaaa*)/8)	+*oooo*

访问标志和输出

当访问 SIMATIC 标志区和输出区时，会计算剩余的 bit_number 并将其用于寻址标志字节或输出字节内的相关位。

访问定时器和计数器

功能代码 FC 15 不允许访问 SIMATIC 定时器区和计数器区，此类访问将被驱动程序拒绝，并输出错误消息帧。

数量

允许使用 1 与 2,040 之间的任何值作为数量（位数）值。

DATA

DATA 域包含位状态（任何值）。

应用示例

表格 3-49　转换功能代码 FC 01、05、15 的 Modbus 寻址

传输消息帧中的 Modbus 地址	SIMATIC 存储区
从 *0* 到 *2,047*	从标志 F *1000*.0 开始
从 *2,048* 到 *2,559*	从输出 Q *256*.0 开始

操作

Modbus 主站系统要将以下位状态写入标志 F 1144.0 …… F 1144.7 和 F 1145.0 …… F 1145.3：

标志	7	6	5	4	3	2	1	0	位
F 1144	ON	ON	OFF	OFF	ON	ON	OFF	ON	

标志	7	6	5	4	3	2	1	0	位
F 1145	-	-	-	-	ON	OFF	OFF	ON	

SEND 源 DB

表格 3-50　下表显示了 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	位起始地址	WORD	W#16#0058	位起始地址
+4.0	位数	INT	10	位数
+6.0	coil_state[1]	WORD	W#16#EFCD	状态线圈 5FH..58H/57H..50H

地址计算：

Modbus 地址“coil_address”0480 十六进制（1152 十进制）位于标志区中：

标志字节	= ((start_address - *aaaaa*)	/8)	+ *uuuuu*
	=((1,152 – *0*)	/8)	+ *1000*
	=1144；		

剩余的 bit_number 的结果如下：

剩余的 bit_no.	= ((start_address - *aaaaa*))	% 8)	[模数为 8]
	=((1,152 –*0*)	% 8)	
	= 0;		

自 F 1144.0 开始访问标志。

更多示例

有关访问标志和输出的更多示例，请参考 FC 01。

3.6.14 功能代码 16 — 预设多个寄存器

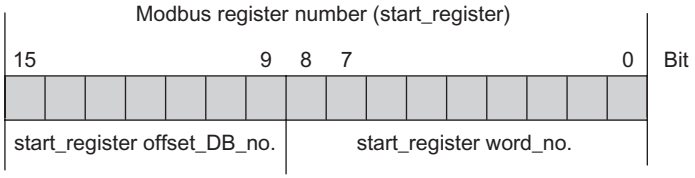
用途和结构

功能代码 16 — 预设多个寄存器的特性如下：

功能	该功能代码使 Modbus 主站系统能够在 SIMATIC CPU 的数据块中写入多个数据字。
请求消息帧	ADDR FUNC start_register 数量 byte_count N n DATA（高、低） CRC
响应消息帧	ADDR FUNC start_register 数量 CRC
LEN（单位：字节）	> 6

start_register

该驱动程序对 Modbus 寄存器地址“start_register”的解释如下：



图片 3-17 解释 Modbus 寄存器号

为了生成更多的地址，FB81 (S_MODB) 使用参数化期间在 FC 03、06、16 的转换 DB 中输入的基本 DB 号（从 DB xxxxx 开始）。

访问地址（地址转换）分两步计算：

访问 SIMATIC	转换公式
数据块 DB（生成的 DB）	= (基本 DB 号 *xxxxx* + start_register - offset_DB_no.)
数据字 DBW	= (start_register word_no.* 2)

如果要写入的生成的 DB 未知，则可以使用以下公式计算主站系统中所需的 Modbus 地址 start_register：

start_register = ((生成的 DB – 基本 DB 号) * 512) + (data word_DBW/2)

只有偶数数据字编号可用于此公式。

数量

允许使用 1 与 127 之间的任何值作为数量（寄存器数）值。

DATA（高、低）

任何值都可用作 DATA 值（高、低）（寄存器值）。Modbus 主站系统要将值 CD09（十六进制）、DE1A（十六进制）和 EF2B（十六进制）写入 DB800 的数据字 DBW100、DBW102 和 DBW104。

应用示例

表格 3-51　转换功能代码 FC 03、06、16 的 Modbus 寻址

传输消息帧中的 Modbus 地址	SIMATIC 存储区	
0	从数据块（基本 DB 号）开始	DB*800*

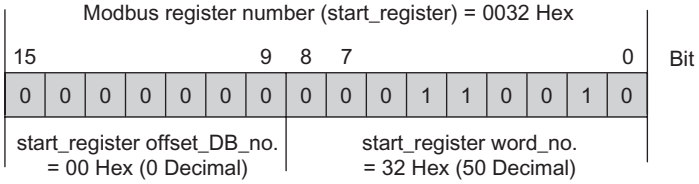
SEND 源 DB

表格 3-52　下表显示了 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	寄存器起始地址	WORD	W#16#0060	寄存器起始地址
+4.0	寄存器数	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

地址计算：

Modbus 地址“start_register”0032 十六进制（50 十进制）的解释如下所示：



图片 3-18　　解释 Modbus 寄存器号 0032（十六进制）

数据块 DB （生成的 DB）	= (基本 DB 号 *xxxxx* + start_register offset_DB_no.)
	=(*800* + 0)
	= 800 ;

数据字 DBW	= (start_register word_no.* 2)
	=(50 * 2)
	= 100;

访问 DB800，数据字 DBW100。

更多示例

可在 FC 03 下找到更多访问示例。

3.6.15 面向位的功能代码转换

说明

面向位的功能代码 02 允许对 SIMATIC 标志存储区和输入存储区进行写保护访问。

转换 DB 可用于指定访问标志和输入的最低/最高 Modbus 地址。也可以参数化应当从其开始访问的 SIMATIC 存储区中的数据元素。

FC 02 的 Modbus 地址区和 SIMATIC 存储区可以独立于 FC 01、05 和 15 的 Modbus 地址区和 SIMATIC 存储区之外单独选择。

表格 3-53　地址区

传输消息帧中的 Modbus 地址	SIMATIC 存储区	
从 kkkkk 开始	标志	从 F vvvv.0 开始
到 lllll		
从 nnnnn 开始	输入	从 I sssss.0 开始
到 rrrr		

3.6.16 面向寄存器的功能代码转换

功能代码 03、06、16

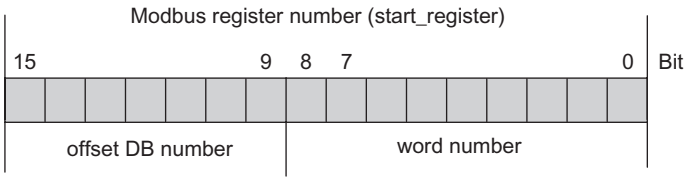
面向寄存器的功能代码 03、06 和 16 允许对 SIMATIC 数据块存储区进行读写访问。

面向寄存器的功能代码 04 仅允许对 SIMATIC 数据块存储区进行读访问。

说明

所需的数据块编号是通过两步计算得出的。

1. 参数化接口可用于指定基本 DB 号。然后此基本 DB 将成为第一个可访问的 DB。
2. 在消息帧中发送的 Modbus 地址 start_register（寄存器号）的解释，如下所示：



图片 3-19 解释 Modbus 寄存器号

生成的 DB 号

随后访问的生成的 DB 号由以下公式计算得出：
基本 DB 号 + 偏移 DB 号。

可使用该生成的 DB 号在整个可寻址数据块区（65,535 个 DB）内访问包含 128 个连续数据块的数据块区。

DB 中的字数

字数可用于在每个数据块内寻址从 DBW0 到 DBW1022 的区域。

其基本结构按字节组织的 DB 由驱动程序逐字进行解释。

功能代码 04 的注意事项

面向寄存器的功能代码 04 仅允许对 SIMATIC 数据块存储区进行读访问。

其访问方式与功能代码 03、06、16 的方式相同。

使用转换 DB 可以为功能代码 04 随意参数化特定的基本 DB 号。这将使您可以选择第二个包含 128 个 DB 的独立区域。

但是，只能对这些 DB 进行读访问。

3.6.17 启用/禁用写访问

功能代码 05、06、15、16

对于写功能代码 05、06、15 和 16，可以禁用或限制对相关 SIMATIC 存储区的访问。

您可以使用转换 DB 指定一个启用区域，供 Modbus 主站系统进行写访问。

如果主站试图访问该启用区域之外的 SIMATIC 存储区，则访问会被拒绝，同时输出一条错误消息帧（例外）。下表说明了如何启用写访问。

表格 3-54 启用写访问

38.0	DB_Number _FC_04	WORD	W#16#0	W#16#2	DB	04
40.0	DB_Min	WORD	W#16#0	W#16#1	使用的最小 DB 号	限制
42.0	DB_Max	WORD	W#16#0	W#16#6	使用的最大 DB 号	
44.0	F_Min	WORD	W#16#0	W#16#1F4	使用的最小标志	
46.0	F_Max	WORD	W#16#0	W#16#4B0	使用的最大标志	
48.0	Q_Min	WORD	W#16#0	W#16#0	使用的最小输出	
50.0	Q_Max	WORD	W#16#0	W#16#64	使用的最大输出	

3.6.18 转换位功能的 Modbus 地址

功能代码 01、05、15

以下部分说明了如何为面向位的功能代码 01、05 和 15 以及 02 转换 Modbus 地址。

01、05、15 概述

面向位的功能代码 01、05 和 15 允许对标志、输出、定时器和计数器的 SIMATIC 存储区进行读写访问。

使用 FC 01 对定时器和计数器进行写保护。

转换 DB 可用于指定访问输出、定时器和计数器的最低/最高 Modbus 地址。也可以参数化应当从其开始访问的 SIMATIC 存储区中的数据元素。

转换 Modbus 地址概述

表格 3-55　转换功能代码 FC 01、05、15 的 Modbus 寻址

参数 DB		**输入**	**含义**
SIMATIC 标志区			
传输消息帧中的 *Modbus 地址*（位号）	从 aaaa 开始	从 0 到 65,535（十进制）	以该 Modbus 地址开始
	到 bbbb	从 0 到 65,535（十进制）	包括该 Modbus 地址
SIMATIC 存储区 标志 （标志）	从 F uuuuu.0 开始	从 0 到 65,535（十进制）	从该标志字节开始
SIMATIC 输出区			
传输消息帧中的 *Modbus 地址*（位号）	自 cccc 开始	从 0 到 65,535（十进制）	以该 Modbus 地址开始
	到 dddd	从 0 到 65,535（十进制）	包括该 Modbus 地址
SIMATIC 存储区 *输出* （输出字节号）	从 Q ooooo.0 开始	从 0 到 65,535（十进制）	从该输出字节开始
SIMATIC 定时器区			
传输消息帧中的 *Modbus 地址*（位号）	自 eeee 开始	从 0 到 65,535（十进制）	以该 Modbus 地址开始
	到 ffff	从 0 到 65,535（十进制）	包括该 Modbus 地址
SIMATIC *定时器*存储区 （定时器号）	从 To ttttt 开始	从 0 到 65,535（十进制）	从该定时器开始 （= 16 位字）
SIMATIC 计数器区			
传输消息帧中的 *Modbus 地址*（位号）	从 gggg 开始	从 0 到 65,535（十进制）	以该 Modbus 地址开始
	到 hhhh	从 0 到 65,535（十进制）	包括该 Modbus 地址
SIMATIC *计数器*存储区 （计数器号）	从 C zzzzz 开始	从 0 到 65,535（十进制）	从该计数器开始 （= 16 位字）

“从/到”Modbus 地址

“从”地址可用于参数化相关区（如标志、输出等）开头的 Modbus 地址。也就是说，它是该区中的第一个位号。

“到”地址可用于参数化相关区（如标志、输出等）末尾的 Modbus 地址。也就是说，它是该区中的最后一个位号。

对于功能代码 FC 01、05 和 15，“从”/“到”地址指传输消息帧中的 Modbus 地址（位号从 0 开始）。

各个“从/到”区间不能重叠。

允许各个“从/到”区间之间存在间隔。

SIMATIC“起始”存储区

“起始”可用于指定“从/到”Modbus 区映射到的 SIMATIC 区的起始位置（= SIMATIC 区的第一个标志字节/输出字节/定时器号/计数器号）。

FC 01、05、15 示例

表格 3-56　转换功能代码 FC 01、05、15 的 Modbus 寻址

参数 DB		**输入**	**含义**
SIMATIC 标志区			
传输消息帧中的 *Modbus 地址*（位号）	从 0 开始	从 0 到 65,535（十进制）	以该 Modbus 地址开始
	最高到 2,047	从 0 到 65,535（十进制）	包括该 Modbus 地址
SIMATIC 存储区 标志 （标志）	从 F 1000.0 开始	从 0 到 65,535（十进制）	从该标志字节开始
SIMATIC 输出区			
传输消息帧中的 *Modbus 地址*（位号）	从 2,048 开始	从 0 到 65,535（十进制）	以该 Modbus 地址开始
	最高到 2,559	从 0 到 65,535（十进制）	包括该 Modbus 地址
SIMATIC 存储区 *输出* （输出字节号）	从 Q 256.0 开始	从 0 到 65,535（十进制）	从该输出字节开始
SIMATIC 定时器区			
传输消息帧中的 *Modbus 地址*（位号）	从 4,096 开始	从 0 到 65,535（十进制）	以该 Modbus 地址开始
	最高到 4,255	从 0 到 65,535（十进制）	包括该 Modbus 地址
SIMATIC *定时器*存储区 （定时器号）	从 T 100 开始	从 0 到 65,535（十进制）	从该定时器开始 （= 16 位字）

参数 DB		输入	含义
SIMATIC 计数器区			
传输消息帧中的 *Modbus 地址*（位号）	从 4,256 开始	从 0 到 65,535（十进制）	以该 Modbus 地址开始
	最高到 4,415	从 0 到 65,535（十进制）	包括该 Modbus 地址
SIMATIC *计数器*存储区（计数器号）	从 C 120 开始	从 0 到 65,535（十进制）	从该计数器开始（= 16 位字）

Modbus 地址从 0 到 2,047 访问 SIMATIC 标志（从标志 F 1000.0 开始）。
即，该区的长度 = 2,048 位 = 256 个字节，这意味着最后一个标志位 = F 1255.7。

Modbus 地址从 2,048 到 2,559 访问 SIMATIC 输出（从位输出 Q 256.0 开始）。
即，该区的长度 = 512 位 = 64 个字节，这意味着最后一个输出位 = Q 319.7。

Modbus 地址从 4,096 到 4,255 访问 SIMATIC 定时器（从定时器 T 100 开始）。
即，该区的长度 = 160 位 = 10 个字，这意味着最后一个定时器 = T 109。

Modbus 地址从 4,256 到 4,415 访问 SIMATIC 计数器（从计数器 C 120 开始）。
即，该区的长度 = 160 位 = 10 个字，这意味着最后一个计数器 = C 129。

FC 02 概述

表格 3-57　　转换 FC 02 的 Modbus 寻址

参数 DB		输入	含义
SIMATIC 标志区			
传输消息帧中的 *Modbus 地址*（位号）	从	从 0 到 65,535（十进制）	以该 Modbus 地址开始
	到	从 0 到 65,535（十进制）	包括该 Modbus 地址
SIMATIC *标志*区	从	从 0 到 65,535（十进制）	从该标志字节开始
SIMATIC 输入区			
传输消息帧中的 *Modbus 地址*（位号）	从	从 0 到 65,535（十进制）	以该 Modbus 地址开始
	到	从 0 到 65,535（十进制）	包括该 Modbus 地址
SIMATIC *输入*存储区（输入字节号）	从 I	从 0 到 65,535（十进制）	从该输入字节开始

“从/到”Modbus 地址

“从”地址可用于参数化相关区（如标志、输入等）开头的 Modbus 地址。也就是说，它是该区中的第一个位号。

“到”地址可用于参数化相关区末尾的 Modbus 地址。也就是说，它是该区中的最后一个位号。

对于功能代码 FC 02，“从/到”地址指传输消息帧中的 Modbus 地址（位号从 0 开始）。

各个“从/到”区间不能重叠。

允许各个“从/到”区间之间存在间隔。

SIMATIC“起始”存储区

“起始”可用于指定“从/到”Modbus 区映射到的 SIMATIC 区的起始位置（= SIMATIC 区的第一个标志字节/输入字节号）。

FC 02 示例

表格 3-58　转换 FC 02 的 Modbus 寻址

参数 DB		输入	含义
SIMATIC 标志区			
传输消息帧中的 *Modbus 地址*（位号）	从 *0* 开始	从 0 到 65,535（十进制）	以该 Modbus 地址开始
	到 *4,095*	从 0 到 65,535（十进制）	包括该 Modbus 地址
SIMATIC *标志*区	从 F *0*.0 开始	从 0 到 65,535（十进制）	从该标志字节开始
SIMATIC 输入区			
传输消息帧中的 *Modbus 地址*（位号）	从 *4,096* 开始	从 0 到 65,535（十进制）	以该 Modbus 地址开始
	到 *5,119*	从 0 到 65,535（十进制）	包括该 Modbus 地址
SIMATIC *输入*存储区（输入字节号）	从 I *128*.0 开始	从 0 到 65,535（十进制）	从该输入字节开始

Modbus 地址从 0 到 4,095 访问 SIMATIC 标志（从标志 F 0.0 开始）：
即，该区的长度 = 4,096 位 = 512 个字节，这意味着最后一个标志位 = F 511.7。

Modbus 地址从 4,096 到 5,119 访问 SIMATIC 输入（从输入 I 128.0 开始）：
即，该区的长度 = 1,024 位 = 128 个字节，这意味着最后一个输入位 = I 255.7。

说明

为“起始标志”输入的值完全独立于为功能代码 01、05、15 的“起始标志”输入的值。

这意味着使用 FC 02 还可以使用完全独立于第一个 SIMATIC 标志区的第二个 SIMATIC 标志区（只读）。

3.6.19 转换寄存器功能的 MODBUS 地址

功能代码 03、06、16

以下部分说明了如何为面向寄存器的功能代码 03、06 和 16 以及 04 转换 Modbus 地址。

功能代码 03、06、16 概述

表格 3-59 转换 FC 03、06、16 的 Modbus 寻址

参数 DB		输入	含义
SIMATIC 数据块区			
传输消息帧中的 *Modbus 地址* = 0 （寄存器号）意味着访问：			
SIMATIC *数据块*存储区	起始 DB	从 1 到 65,535（十进制）	从该数据块开始 从 DBW0（作为 基本 DB 号）开始

起始 DB

“起始 DB”可用于指定要访问的 SIMATIC 区的第一个数据块（= 基本 DB 号）。

在 Modbus 消息帧的寄存器号的值为 0 时访问该 DB（从数据字 DBW0 开始）。

较高的 Modbus 寄存器号访问其后的数据字/数据块。

最多可以寻址 127 个后续 DB

驱动程序解释 Modbus 寄存器号的 9 — 15 位以访问各个后续 DB。

应用示例

表格 3-60 转换 FC 03、06、16 的 Modbus 寻址

参数 DB		输入	含义
SIMATIC 数据块区			
传输消息帧中的 *Modbus 地址* = 0 （寄存器号）意味着访问：			
SIMATIC *数据块*存储区	从 DB800 开始	从 1 到 65,535（十进制）	从该数据块开始 从 DBW0（作为基本 DB 号）开始

Modbus 寄存器地址 0 可用于在 SIMATIC 系统中访问数据块 800（从 DBW0 开始）。

较高的 Modbus 寄存器地址（>= 512 等）访问其后的 DB，如 DB801 等等。

FC 04 概述

表格 3-61　转换 FC 04 的 Modbus 寻址

参数 DB		输入	含义
SIMATIC 数据块区			
传输消息帧（寄存器号）中的 *Modbus 地址* = 0 意味着访问：			
SIMATIC *数据块*存储区	起始 DB	从 1 到 65,535（十进制）	从该数据块开始 从 DBW0（作为基本 DB 号）开始

起始 DB

“起始 DB”可用于指定要访问的 SIMATIC 区的第一个数据块（= 基本 DB 号）。

在 Modbus 消息帧的寄存器号的值为 0 时访问该 DB（从数据字 DBW0 开始）。

较高的 Modbus 寄存器号访问其后的数据字/数据块。

最多可以寻址 127 个后续 DB。驱动程序解释 Modbus 寄存器号的 9 — 15 位以访问各个后续 DB。

说明

为“起始 DB”输入的值完全独立于为功能代码 03、06 和 16 的“起始 DB”输入的值。

因此，使用 FC 04 可以使用完全独立于第一个 SIMATIC 数据块区的第二个 SIMATIC 数据块区（只读）。

FC 04 示例

表格 3-62　转换 FC 04 的 Modbus 寻址

参数 DB		输入	含义
SIMATIC 数据块区			
传输消息帧（寄存器号）中的 *Modbus 地址* = 0 意味着访问：			
SIMATIC *数据块*存储区	从 DB1200 开始	从 1 到 65,535（十进制）	从该数据块开始 从 DBW0（作为基本 DB 号）开始

Modbus 寄存器地址 0 可用于在 SIMATIC 系统中访问数据块 1200（从 DBW0 开始）。

较高的 Modbus 寄存器地址（>= 512、1,024 等）访问其后的 DB，如 DB1201、DB1202 等等。

3.6.20 写功能的限制

FC 05、06、15、16 概述

表格 3-63 写访问（FC 05、06、16）的 SIMATIC 限制

参数 DB		输入	含义
数据块 DB：生成的 DB 号	DB MIN	从 1 到 65,535	第一个启用的 DB
	DB MAX	从 1 到 65,535	最后一个启用的 DB MAX=0 禁用所有 DB
标志 F （标志字节号）	F MIN	从 0 到 65,535	第一个启用的标志字节
	F MAX	从 1 到 65,535	最后一个启用的标志字节 MAX=0 禁用所有标志
输出 Q （输出字节号）	Q MIN	从 0 到 65,535	第一个启用的输出字节
	Q MAX	从 1 到 65,535	最后一个启用的输出字节 MAX=0 禁用所有输出

SIMATIC 存储区 MIN/MAX

对于写功能代码，可以指定访问的下限和上限 (MIN/MAX)。仅可在此启用区域内进行写访问。

如果上限值为 0，则整个区域被禁用。

在 SIMATIC 中选择区域的大小时，请记住区域的大小取决于 CPU。

如果主站试图对上限/下限之外的区域进行写访问，则模块会拒绝此类访问，并输出错误消息帧。

数据块区的 MIN/MAX 值必须指定为生成的 DB 号。

FC 05、06、16 应用示例

表格 3-64　　写访问（FC 05、06、16）的 SIMATIC 限制

参数 DB		输入	含义
数据块 DB：生成的 DB 号	MIN 600	1 到 65,535	第一个启用的 DB
	MAX 699	1 到 65,535	最后一个启用的 DB MAX=0 禁用所有 DB
标志 F （标志字节号）	MIN 1000	0 到 65,535	第一个启用的标志字节
	MAX 1127	1 到 65,535	最后一个启用的标志字节 MAX=0 禁用所有标志
输出 Q （输出字节号）	MIN 256	0 到 65,535	第一个启用的输出字节
	MAX 319	1 到 65,535	最后一个启用的输出字节 MAX=0 禁用所有输出

可以使用写功能代码（FC 06、16）访问 SIMATIC 数据块 DB600 到 DB699。

可以使用写功能代码（FC 05、15）访问 SIMATIC 标志字节 1000 到 1127。

可以使用写功能代码（FC 05、15）访问 SIMATIC 输出字节 256 到 319。

3.7 诊断

3.7.1 诊断选项

原理

ET 200S Modbus/USS 串行接口模块的诊断功能可用于确定操作期间发生的任何故障的原因。可以使用以下诊断方式：

- 通过 ET 200S Modbus/USS 串行接口模块前面板上的状态 LED 进行诊断
- 通过功能块的 STATUS 输出进行诊断
- 通过 PROFIBUS 从站诊断进行诊断

3.7.2 状态 LED 的诊断信息

状态 LED 的功能

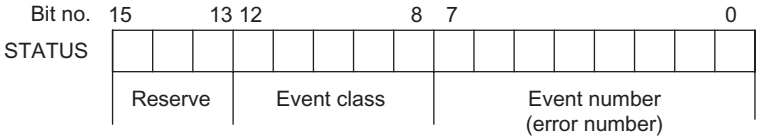
以下状态 LED 位于 ET 200S Modbus/USS 串行接口模块的前面板上：

- **TX**（绿色）：当 ET 200S Modbus/USS 串行接口模块通过接口发送数据时亮起。
- **RX**（绿色）：当 ET 200S Modbus/USS 串行接口模块通过接口接收数据时亮起。
- **SF**（红色）：指示可能发生以下错误/故障之一的一组故障 LED：
  - 硬件故障
  - 参数分配错误
  - ET 200S Modbus/USS 串行接口模块与通讯伙伴之间发生断线或电缆断开：当接收线路的初始状态设置为 R(A) 5 V/R(B) 0 V 时，仅使用 RS 422 从站诊断接口连接可以检测到。
  - 通讯错误（奇偶校验错误、帧错误、缓冲区溢出）

3.7.3 功能块的诊断消息

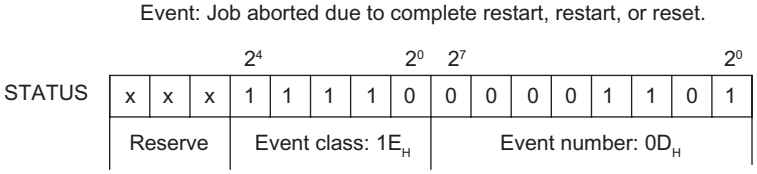
功能块诊断消息的结构

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。下图说明了 STATUS 参数的结构。



图片 3-20　　STATUS 参数的结构

作为示例，下图说明了针对“作业因重新启动、暖启动或复位而取消”事件（事件类别 $1E_H$，事件编号 $0D_H$），STATUS 参数的内容。



图片 3-21　　示例：“事件类别 1EH，事件 0DH”的 STATUS 参数

调用 SFCERR 变量

SFCERR 变量包含有关事件类别 30 中的错误 14 (1E $0E_H$) 和 15 (1E $0F_H$) 的附加信息。

从相应功能块的背景数据块中装载 SFCERR 变量。

*《用于 S7-300/400 系统和标准功能的系统软件》参考*手册中有关 SFC14“DPRD_DAR”和 SFC15“DPWR_DAT”系统功能的章节中说明了在 SFCERR 变量中输入的错误消息。

功能块诊断消息含义

下表说明了事件类别、事件编号的定义和每种错误情况的建议解决方法。

表格 3-65　　事件类别 2（0x02 十六进制）：执行 CPU 作业时出错

事件类别 2（0x02 十六进制）：“初始化错误”			
事件编号	**事件编号（十进制）**	**事件**	**补救措施**
(02) 01_H	1	未进行（有效的）参数化	为模块分配有效的参数。如有必要，请检查系统的安装是否正确。

表格 3-66　事件类别 5（05 十六进制）：执行 CPU 作业时出错

事件类别 5（05 十六进制）：执行 CPU 作业时出错			
事件编号	**事件编号（十进制）**	**事件**	**补救措施**
(05) 02_H	2	不允许在 ET 200S Modbus/USS 串行接口模块的该操作模式（例如，设备接口未被参数化）下进行作业。	判断诊断中断并相应地纠正错误。
(05) $0E_H$	14	消息帧长度无效	发送消息桢的长度大于 256 个字节。 ET 200S 1SI 模块已取消了发送消息帧。 选择较短的帧长度。
(05) 51_H	81	在 ET 200S Modbus/USS 串行接口模块和 CPU 进行通讯期间，帧执行出错。在传输从 ET 200S SI 串行接口模块收到的消息帧期间，CPU 中出错。	模块和 CPU 均已取消传输。重复接收作业。 ET 200S/USS 串行接口模块重新发送收到的消息。

表格 3-67　事件类别 8（08 十六进制）：接收错误

事件类别 8（08 十六进制）：接收错误			
事件编号	**事件编号（十进制）**	**事件**	**补救措施**
(08) 06_H	6	超过字符延迟时间。在字符延迟时间内未收到两个连续字符。	伙伴设备过慢或发生故障。检查伙伴设备是否发生故障；您可能需要使用转换为传输线路的接口测试设备 (FOXPG)。
08 $0A_h$	10	接收响应消息帧时，主站中的接收缓冲区溢出。	检查从站协议设置。
(08) $0C_H$	12	检测到传输错误（奇偶校验/停止位/溢出错误）。	传输线路上的干扰造成帧重复，因此降低了用户数据的吞吐量。漏检错误的风险增加。更改系统设置或电缆接线。 检查通讯伙伴的连接电缆，或检查双方设备对波特率、奇偶校验和停止位数的设置是否相同。
(08) $0D_H$	13	BREAK：到伙伴的接收线路断路。	重新连接或接通伙伴电源。
(08) 10_H	16	奇偶校验错误：如果 SF LED（红色）亮起，则表明两个通讯伙伴之间的连接电缆断开（线路断路）。	检查通讯伙伴的连接电缆，或检查双方设备对波特率、奇偶校验和停止位数的设置是否相同。 更改系统设置或电缆接线。
(08) 11_H	17	字符帧错误：如果 SF LED（红色）亮起，则表明两个通讯伙伴之间的连接电缆断开（线路断路）。	检查通讯伙伴的连接电缆，或检查双方设备对波特率、奇偶校验和停止位数的设置是否相同。 更改系统设置或电缆接线。
(08) 12_H	18	在串行接口将 CTS 设置为 OFF 后收到多个字符。	重新组态通讯伙伴，或更快地从串行接口读取数据。

事件类别 8（08 十六进制）：接收错误			
事件编号	**事件编号（十进制）**	**事件**	**补救措施**
08 30_H	48	**主站**：请求消息帧已发送且响应监视时间已结束，未检测到响应消息帧的开头。 **从站**：该功能代码不允许使用广播。	检查传输线路是否已中断（可能需要进行接口分析）。 检查是否对模块和通讯伙伴的以下协议参数进行了相同的设置：传输率、数据位数、奇偶校验和停止位数。 检查 PtP_PARAM 中的响应监视时间的值是否足够大。 检查指定的从站地址是否存在。 对于能够使用广播功能的功能代码，仅允许 Modbus 主站系统使用广播功能。
08 31_H	49	**主站**：从站响应消息帧中的第一个字符与在请求消息帧中发送的从站地址不同（对于常规操作）。 **从站**：收到的功能代码是不允许的。	错误的从站进行了响应。 检查传输线路是否已中断（可能需要进行接口分析）。 该功能代码不能用于此驱动程序。
08 32_H	50	超出最大位数或寄存器数 或者访问 SIMATIC 定时器存储区或计数器存储区时位数无法被 16 整除。	将最大位数和最大寄存器数分别限制为 2,040 和 127。仅以 16 位间隔访问 SIMATIC 定时器、计数器。
08 33_H	51	功能代码 FC 15/16 的位数和寄存器数与消息帧元素 byte_count 不匹配。	纠正位数/寄存器数或 byte_count。
08 34_H	52	检测到“设置位/重设位”的位代码非法。	对于 FC 05，仅能使用代码 0000 十六进制或 FF00 十六进制。
08 35_H	53	检测到功能代码 FC 08“回送测试”的诊断子代码（_0000 十六进制）非法。	对于 FC 08，仅使用子代码 0000 十六进制。
08 36_H	54	CRC 16 校验和的内部产生的值与收到的 CRC 校验和不匹配。	检查 Modbus 主站系统中 CRC 校验和的生成。
08 37_H	55	消息帧序列错误： Modbus 主站系统在驱动程序发送最后一个响应消息帧之前发送了新的请求消息帧。	增加 Modbus 主站系统的从站响应消息帧的超时时间。
08 50_H	80	接收消息帧的长度大于 224 个字节或定义的消息帧长度。	调整伙伴的消息帧长度。

表格 3-68　事件类别 14（0E 十六进制）：一般处理错误 <参数化>

事件类别 14（0E 十六进制）：一般处理错误 <参数化>			
事件编号	**事件编号（十进制）**	**事件**	**补救措施**
0E 20_H	32	该连接的数据位数必须为 8。 驱动程序未准备好进行此操作。	纠正驱动程序的参数。
0E 21_H	33	已设置的字符延迟时间的倍增因子不在 1 到 10 之内。驱动程序使用缺省设置 1 进行操作。	纠正驱动程序的参数。
0E 22_H	34	驱动程序的操作模式设置非法。必须指定是“常规操作”还是“噪声抑制操作”。 驱动程序未准备好进行此操作。	纠正驱动程序的参数。
0E 23_H	35	**主站**：为响应监视时间设置了非法的值：有效的值在 50 ms 和 655,000 ms 之间。 驱动程序未准备好进行此操作。 **从站**：为从站地址设置了非法的值。从站地址 0 非法。 驱动程序未准备好进行此操作。	纠正驱动程序的参数。 纠正驱动程序的参数。
0E $2E_H$	46	读取接口参数文件时出错。 驱动程序未准备好进行此操作。	重新启动主站 (Mains_ON)。

表格 3-69　事件类别 14（0E 十六进制）：一般处理错误 <处理 S_SEND 作业>

事件类别 14（0E 十六进制）：一般处理错误 <处理 S_SEND 作业>			
事件编号	**事件编号（十进制）**	**事件**	**补救措施**
0E 40_H	64	使用 S_SEND 为 LEN 指定的值太小。	最小长度是 2 个字节。
0E 41_H	65	使用 S_SEND 为 LEN 指定的值太小。发送功能代码需要更大的长度。	该功能代码的最小长度是 6 个字节。
0E 42_H	66	发送的功能代码非法。	仅使用允许的功能代码。
0E 43_H	67	该功能代码不允许使用从站地址 0（= 广播）。	仅为适合的功能代码使用从站地址 0。
0E 44_H	68	发送的“位数”的值不在 1 到 2,040 之内。	“位数”必须在 1 到 2,040 之内。
0E 45_H	69	发送的“寄存器数”的值不在 1 到 127 之内。	“寄存器数”必须在 1 到 127 之内。
0E46_H	70	功能代码 15 或 16： 发送的“位数”/“寄存器数”的值不在 1 到 2,040/1 到 127 之内。	“位数”/“寄存器数”必须在 1 到 2,040/1 到 127 之内。

事件类别 14（0E 十六进制）：一般处理错误 <处理 S_SEND 作业>			
事件编号	**事件编号（十进制）**	**事件**	**补救措施**
0E 47_H	71	功能代码 15 或 16： S_SEND 的 LEN 与发送的“位数”/“寄存器数”不符。 LEN 太小。	为 SEND 增加 LEN，直到足够的用户数据发送到模块。 必须发送更多的用户数据到模块以达到“位数”/“寄存器数”。
0E 48_H	72	功能代码 5： 在 SEND 源 DB 中为“设置位”(FF00H) 或“删除位”(0000H) 指定的代码不正确。	唯一允许的代码是“设置位”(FF00H) 和“删除位”(0000H)。
0E 49_H	73	功能代码 8： 在 SEND 源代码中为“诊断代码”指定的代码不正确。	唯一允许的代码是“诊断代码”0000H。
0E $4A_H$	74	该功能代码的长度大于最大长度。	您将在手册中找到每个功能代码的最大长度。

表格 3-70　事件类别 14（0E 十六进制）：一般处理错误 <接收判断>

事件类别 14（0E 十六进制）：一般处理错误 <接收判断>			
事件编号	**事件编号（十进制）**	**事件**	**补救措施**
0E 50_H	80	主站在未发送任何内容的情况下收到响应。	网络中有从站或另一个主站。 检查传输线路是否已中断（可能需要进行接口分析）。
0E 51_H	81	功能代码不正确： 在响应消息帧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 52_H	82	字节下溢： 收到的字符数少于响应消息帧的字节计数器指示的字符数或该功能代码预期的字符数。	检查从站设备。
0E 53_H	83	字节上溢： 收到的字符数多于响应消息帧的字节计数器指示的字符数或该功能代码预期的字符数。	检查从站设备。
0E 54_H	84	字节计数器不正确： 响应消息帧中收到的字节计数器太小。	检查从站设备。
0E 55_H	85	字节计数器不正确： 响应消息帧中收到的字节计数器不正确。	检查从站设备。
0E 56_H	86	回复不正确： 从站回复的响应消息帧数据（位数等）与在响应消息帧中发送的数据不同。	检查从站设备。
0E 57_H	87	CRC 校验不正确： 校验来自从站的响应消息帧的 CRC 16 校验和时出错。	检查从站设备。

表格 3-71　事件类别 14（0E 十六进制）：一般处理错误 <接收例外代码消息>

事件类别 14（0E 十六进制）：一般处理错误 <接收例外代码消息>			
事件编号	**事件编号（十进制）**	**事件**	**补救措施**
0E 61_H	97	例外代码为 01 的响应消息帧： 功能非法	请参考手册，以获取有关从站设备的信息。
0E 62_H	98	例外代码为 02 的响应消息帧： 数据地址非法	请参考手册，以获取有关从站设备的信息。
0E 63_H	99	例外代码为 03 的响应消息帧： 数据值非法	请参考手册，以获取有关从站设备的信息。
0E 64_H	100	例外代码为 04 的响应消息帧： 关联的设备发生故障	请参考手册，以获取有关从站设备的信息。
0E 65_H	101	例外代码为 05 的响应消息帧： 确认	请参考手册，以获取有关从站设备的信息。
0E 66_H	102	例外代码为 06 的响应消息帧： 已占用；消息帧被拒绝	请参考手册，以获取有关从站设备的信息。
0E 67_H	103	例外代码为 07 的响应消息帧： 否定确认	请参考手册，以获取有关从站设备的信息。

表格 3-72　事件类别 30（1E 十六进制）：在串行接口和 CPU 进行通讯期间出错

事件类别 30（1E 十六进制）：在串行接口和 CPU 进行通讯期间出错			
事件编号	**事件编号（十进制）**	**事件**	**补救措施**
(1E) $0D_H$	13	“作业因重新启动、暖启动或复位而取消”	
(1E) $0E_H$	14	调用 SFC DP_RDDAT 期间出现静态错误。 SFC 的 RET_VAL 返回值可用于背景数据块中 SFCERR 变量的判断。	从背景数据块中装载 SFCERR 变量。
(1E) $0F_H$	15	调用 SFC DP_WRDAT 期间出现静态错误。 SFC 的 RET_VAL 返回值可用于背景数据块中 SFCERR 变量的判断。	从背景数据块中装载 SFCERR 变量。
(1E) 10_H	16	调用 SFC RD_LGADR 期间出现静态错误。 SFC 的 RET_VAL 返回值可用于背景数据块中 SFCERR 变量的判断。	从背景数据块中装载 SFCERR 变量。
(1E) 11H	17	调用 SFC RDSYSST 期间出现静态错误。 SFC 的 RET_VAL 返回值可用于背景数据块中 SFCERR 变量的判断。	从背景数据块中装载 SFCERR 变量。
(1E) 20_H	32	参数超出范围。	为该功能块输入允许范围内的参数。
(1E) 41_H	65	在 FB 的 LEN 参数中指定的字节数未获允许。	值必须在 1 到 256 个字节的范围内。

判断 SFCERR 变量

有关属于事件类别 30 的错误 (1E) $0E_H$、(1E) $0F_H$、(1E) 10_H 和 (1E) 11_H 的更多详细信息，可以通过 SFCERR 变量获得。

您可以从相应功能块的背景数据块中装载 SFCERR 变量。

*《用于 S7-300/400 系统和标准功能的系统软件》参考*手册中有关“DPRD_DAR”、SFC15“DPWR_DAT”和 RD_LGADR 系统功能的章节中说明了在 SFCERR 变量中输入的错误消息。

3.7.4 PROFIBUS 从站诊断

简介

从站诊断数据符合“EN 50170，卷 2，PROFIBUS”的要求。根据 DP 主站，可以使用 STEP 5 或 STEP 7 读取符合该标准的所有 DP 从站的诊断数据。

PROFIBUS 从站诊断包括模块诊断、模块状态诊断和通道特定的诊断。可在*《ET 200S 分布式 I/O 系统，6ES7 151-1AA10-8AA0》*手册中找到有关 DP 从站诊断的详细信息。

通道特定的诊断

通道特定的诊断可以提供有关模块中通道错误的信息，该诊断在模块状态诊断之后启动。下表列出了通道特定错误的类型。

表格 3-73　ET 200S Modbus/USS 串行接口模块的通道特定错误的类型

事件（错误类型）	说明	建议的措施
00110：断线	发生断线或断开连接。	检查到端子的接线。检查连接至伙伴的电缆。
00111：上溢	缓冲区上溢；消息长度上溢。	必须更频繁地调用 S_RCV FB。
01000：下溢	发送了长度为 0 的消息。	检查通讯伙伴为何发送不含用户数据的消息帧。
01001：错误	出现内部模块错误。	更换模块。
10000：参数分配错误	模块未参数化。	纠正参数化。
10110：消息错误	帧错误、奇偶校验错误。	检查通讯设置。

3.7.5 Modbus 从站诊断功能

ERROR_NR 和 ERROR_INFO

Modbus 通讯 FB 具有以下两个输出参数，可以指示发生的错误：

- 参数 ERROR_NR
- 参数 ERROR_INFO

在 ERROR_NR 输出中指示错误。有关 ERROR_NR 中错误的其它详细信息在输出 ERROR_INFO 中显示。

删除错误

在 START 的某个上升沿上删除错误。如有必要，用户可随时删除错误显示。

PB 错误代码

错误代码 1 到 99 的含义如下：

- **ERROR_No 1 到 9**

  初始化 FB 和 CP 期间的错误

  错误编号 1 到 9 指示初始化由于出错而终止。参数 START_ERROR 为 1。

  Modbus 无法与主站系统进行通讯。

- **ERROR_No 10 到 19**

  处理功能代码期间的错误

  错误编号 10 到 19 指示处理功能代码期间出现的错误。模块向通讯 FB 发送了一个非法的处理作业。

  系统还将该错误报告给了驱动程序。

  对后续处理作业仍将继续进行处理。

- **ERROR_No 90 到 99**

  其它错误

  发生了一个处理错误。

  系统未将该错误报告给驱动程序。

  对后续处理作业仍将继续进行处理。

3.7.6 错误

错误编号列表

表格 3-74 初始化错误

错误编号（十进制）	ERROR_INFO	事件	补救措施
0	0	没有错误	
1	SFC 51->RET_VAL	使用 SFC 51 读取系统状态列表时出错。	分析 ERROR_INFO 中的 RET_VAL；消除出错原因。
2	S_SEND->STATUS， S_RCV->STATUS	模块初始化期间超时或出错（S_SEND 作业中有错误）。	检查是否已为“Modbus 从站”设置了适用于该接口的协议。 检查在通讯 FB 中指定的“ID”是否正确。 分析 ERROR_INFO。

表格 3-75 处理功能代码时出错

错误编号（十进制）	ERROR_INFO	事件	补救措施
11	起始地址	驱动程序向通讯 FB 传输的起始地址非法。	检查 Modbus 主站系统的 Modbus 地址。
12	寄存器数	驱动程序向通讯 FB 传输的寄存器的数目非法。 寄存器数 = 0。	检查 Modbus 主站系统的寄存器数。 如有必要，请重新启动该模块 (Mains_ON)。
13	寄存器数	驱动程序向通讯 FB 传输的寄存器的数目非法： 寄存器数 > 128。	检查 Modbus 主站系统的寄存器数。 如有必要，请重新启动该模块 (Mains_ON)。
14	标志 F — 结束地址	试图访问区域外部的 SIMATIC 标志存储区。 注意： SIMATIC CPU 中的区域长度取决于 CPU 的类型。	减小 Modbus 起始地址的长度或 Modbus 主站系统中的访问长度。
15	输出 Q — 结束地址 输出 I — 结束地址	试图访问区域外部的 SIMATIC 输出存储区。 注意： SIMATIC CPU 中的区域长度取决于 CPU 的类型。	减小 Modbus 起始地址的长度或 Modbus 主站系统中的访问长度。
16	定时器 T — 结束地址	试图访问区域外部的 SIMATIC 定时器存储区。 注意： SIMATIC CPU 中的区域长度取决于 CPU 的类型。	减小 Modbus 起始地址的长度或 Modbus 主站系统中的访问长度。

表格 3-76　处理功能代码时出错

错误编号（十进制）	ERROR_INFO	事件	补救措施
17	计数器 C — 结束地址	试图访问区域外部的 SIMATIC 计数器存储区。 注意： SIMATIC CPU 中的区域长度取决于 CPU 的类型。	减小 Modbus 起始地址的长度或 Modbus 主站系统中的访问长度。
18	0	驱动程序向通讯 FB 传输的 SIMATIC 存储区非法。	如有必要，请重新启动该模块 (Mains_ON)。
19		访问 SIMATIC I/O 时出错。	检查所需的 I/O 是否存在并且无错误。
20	DB#	DB 不存在。	将该 DB 添加到您的项目中。
21	DB#	DB 长度无效	增加 DB 的长度。
22	DB#	DB# 小于最小 DB 值。	更改最小 DB 值。
23	DB#	DB# 大于最大 DB 值。	更改最大 DB 值。
24	标志地址	标志小于下限。	更改转换 DB 中标志的下限。
25	标志地址	标志大于上限。	更改转换 DB 中标志的上限。
26	输出地址	输出小于下限。	更改转换 DB 中输出的下限。
27	输出地址	输出大于上限。	更改转换 DB 中输出的上限。

表格 3-77　其它故障/错误

错误编号（十进制）	ERROR_INFO	事件	补救措施
90	S_SEND-> STATUS	使用 S_SEND 向驱动程序传输确认消息帧时出错。	分析 STATUS 信息。
94	S_RCV->STATUS	使用 S_RCV (STATUS) 读取 SYSTAT 时出错。	分析 STATUS 信息。

3.8 USS 主站

3.8.1 什么是 USS 主站？

简介

使用 USS 协议，用户可以在 ET 200S Modbus/USS 模块（作为主站）和若干从站系统之间建立串行总线通讯。可以将 Siemens 驱动器作为 USS 总线上的从站进行操作。

USS 协议的特性

USS 协议具有以下重要特性：

- 支持多点 RS 485 连接
- 主站-从站访问方法
- 带有主站的系统
- 最多 32 个节点（最多 31 个从站）
- 使用可变长度或固定长度的消息帧进行操作
- 简单可靠的消息帧
- 总线操作与 PROFIBUS 相同（DIN 19245 第 1 部分）
- 到基本驱动器转换器的数据接口基于具有不同速度的 PROFIL 驱动器。换言之，使用 USS 协议时，信息传送到驱动器的方式与使用 PROFIBUS DP 时的方式相同。
- 可用于启动、维护和自动化操作

3.8.2 组态和参数化

组态和参数化

表格 3-78 USS 主站的参数

参数	说明	值范围	缺省值
诊断中断	指定在出现严重错误时，模块是否应生成诊断中断。	• 否 • 是	否
激活 BREAK 检测	如果发生线路断路或未连接接口电缆，该模块将产生错误消息“断路”。	• 否 • 是	否
接口类型	指定要使用的电气接口。	• RS 232 • RS 485（半双工）	RS 485（半双工）
接收线路的半双工初始状态	指定 RS 485 操作模式下接收线路的初始状态。不是在 RS 232C 操作模式下使用的初始状态。 仅在更换零件时需要确保兼容性的情况下，才需要“反转信号电平”设置。	R(A) 5 V/R(B) 0 V R(A) 0 V/R(B) 5 V 反转信号电平 无	R(A) 0 V/R(B) 5 V
波特率	选择数据传输率（单位：位/秒）。	• 110 • 300 • 600 • 1,200 • 2,400 • 4,800 • 9,600 • 19,200 • 38,400 • 57,600 • 76,800 • 115,200	9,600

说明

请参见标识数据（页码 63）和固件更新的后续装载（页码 65）中涉及的主题。

3.8.3 USS 协议

简介

USS 协议是一种简单的串行数据传输协议，旨在满足驱动器技术的要求。

USS 协议定义了一种基于主站-从站原理通过串行总线进行通讯的访问方法。总线可以连接一个主站和最多 31 个从站。主站使用消息帧中的地址字符来选择各个从站。只有通过主站启动的从站才能发送消息。因此，各个从站之间无法直接传输数据。以半双工模式进行通讯。无法传输此主站功能。USS 系统只有一个主站。

消息帧结构

每个消息帧均以一个起始字符 (STX) 开头，后面依次为长度规范 (LGE)、地址字节 (ADR) 和数据域。消息帧以块校验字符 (BCC) 结束。

STX	LGE	ADR	1	2	...	N	BCC

对于网络数据块中的单字（16 位）数据，首先发送高字节，然后发送低字节。相应地，对于双字数据，采取同样的发送方式。

该协议不识别数据域中的任何任务。

数据加密

此数据按如下方式进行加密：

- STX：1 个字节，文本开头，02H
- LGE：1 个字节，包含以二进制数形式表示的消息帧长度
- ADR：1 个字节，包含以二进制代码形式表示的从站地址和消息帧类型
- 数据域：每个数据域 1 个字节，内容与任务相关
- BCC：1 个字节，块校验字符

数据传输步骤

该主站可确保在消息帧中进行循环数据传输。该主站使用任务消息帧对所有从站节点逐个进行寻址。被寻址的节点通过一个响应消息帧进行响应。从站按照主站-从站步骤接收任务消息帧之后必须向主站发送响应消息帧。只有这样，主站才能对下一个从站进行寻址。

网络数据块的一般结构

网络数据块分为两个区域：参数区 (PKW) 和过程数据区 (PZD)。

STX	LGE	ADR	参数 (PKW)	过程数据 (PZD)	BCC

- **参数区** (PKW)

  PKW 区处理两个通讯伙伴（例如控制器和驱动器）之间的参数传输。例如，这包括读取和写入参数值以及读取参数说明和关联的文本。PKW 接口通常包含操作、显示、维护和诊断任务。

- **过程数据区** (PZD)

  PZD 区包含自动化操作所需的信号：

  – 控制字和设定值（从主站到从站）

  – 状态字和实际值（从从站到主站）

  参数区和过程数据区的内容由从站驱动器进行定义。有关这方面的其它信息，请参考驱动器文档。

3.8.4 功能概述

网络数据传输顺序

这些块按照轮询列表（参数 DB）中指定的顺序循环处理最多 31 个驱动器从站的网络数据传输。一次只能为每个从站激活一个作业。用户将每个从站的网络数据存储在一个数据块（网络数据块）中，并从该数据块中调用数据。由于已在轮询列表中的程序定义中指定，因此将通过其它数据存储区（通讯处理器 DB）向通讯处理器中传输数据以及从通讯处理器中调用数据。

该步骤需要两个功能调用（一个发送块，另一个接收块）。其它功能支持生成并预设通讯所需的数据块。

性能特性：

- 根据总线组态为通讯创建数据存储区
- 预设轮询列表
- 消息帧结构符合 USS 规范
- 可根据所需的网络数据结构对网络数据交换进行参数化
- 执行和监视 PKW 作业
- 处理参数更改报告
- 监视整个系统和错误消除情况

可使用不同的网络数据结构发送网络数据。

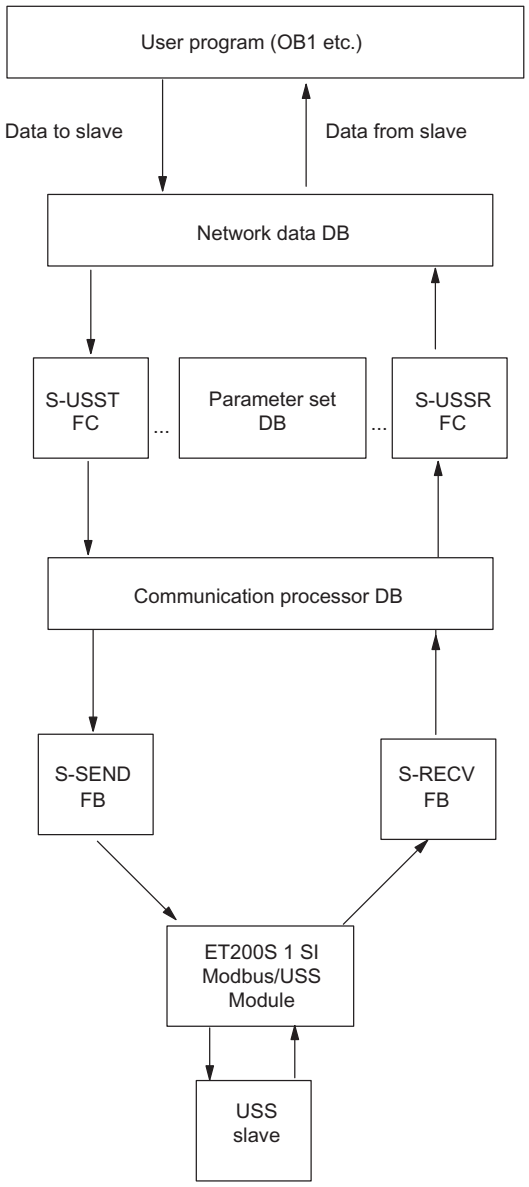
根据所选的结构，网络数据具有 PZD 区用于过程数据，PKW 区用于处理参数。

在 PKW 区中，主站可以读取和写入参数值，而从站可以通过参数更改报告显示参数更改。

PZD 区包含过程控制所需的信号，例如从控制字和设定值（从主站到从站），以及状态字和实际值（从从站到主站）。

功能调用的正确顺序为：S_USST、S_SEND、S_RCV、S_USSR。该顺序很重要，因为 S_SEND 和 S_RCV 功能的输出仅在自动化系统的当前周期中有效。

下图显示了用户程序和 USS 从站之间的数据传输。



图片 3-22　　用户程序和 USS 从站之间的数据传输

3.8.5 FC17 S_USST：将数据传输到从站

说明

S_USST FC 根据所用的网络数据结构处理到从站的网络数据（PZD 和所有 PKW 数据）的传输。

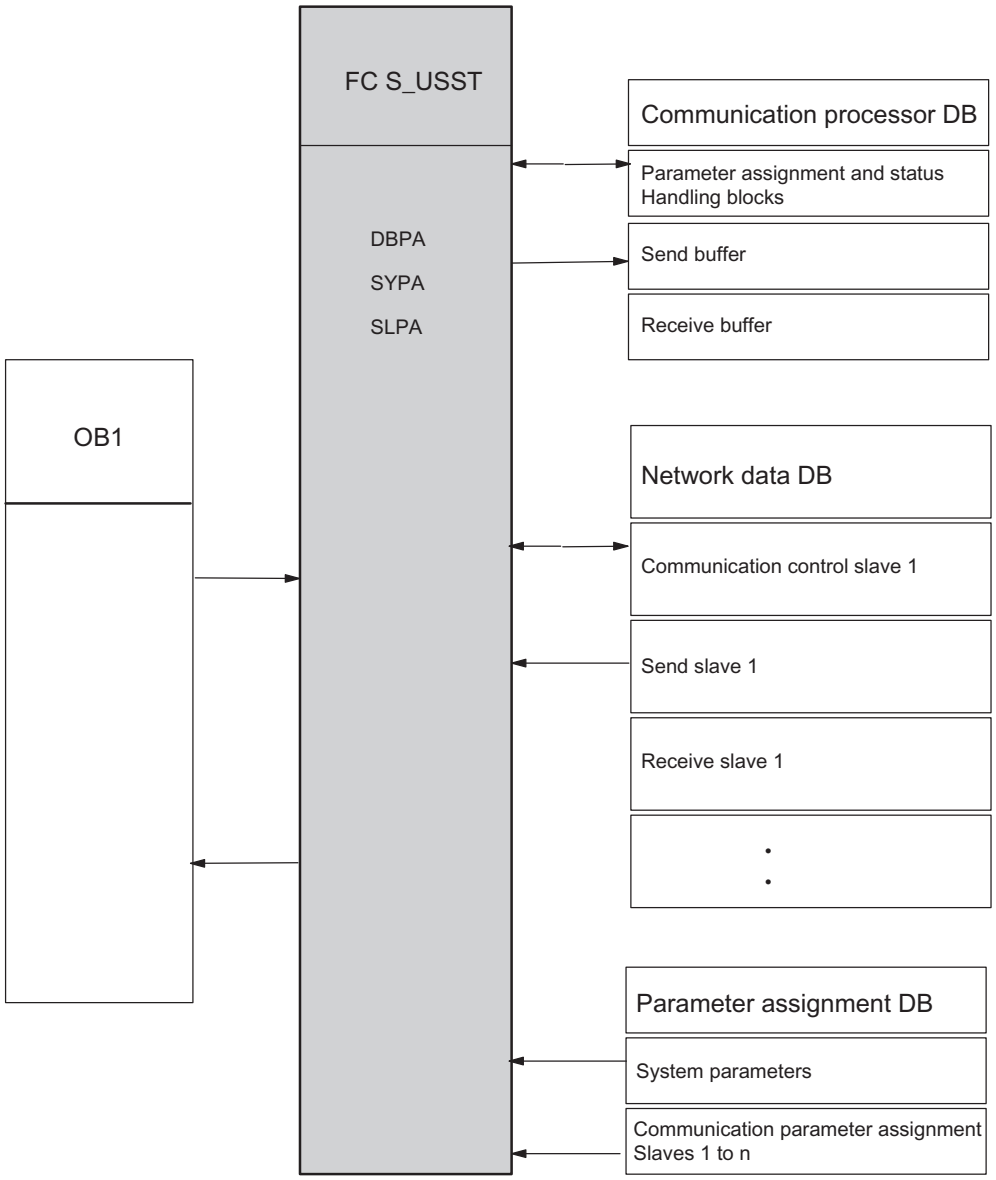
该 FC 从轮询列表中获取当前从站的参数（参数 DB），并从网络数据 DB 中发送数据。它计算当前从站的通讯控制字（触发 PKW 作业/确认参数更改报告），完成 USS 传输数据，并将数据传输至通讯处理器 DB 的发送缓冲区中。最后，它使用 S_SEND FB 触发到从站的网络数据传输。

如果该功能检测到参数 DB 中有参数错误，会将一个错误信号存储在网络数据 DB 的参数错误 2 字节中。

每个自动化系统周期只能调用一次 FC 17。

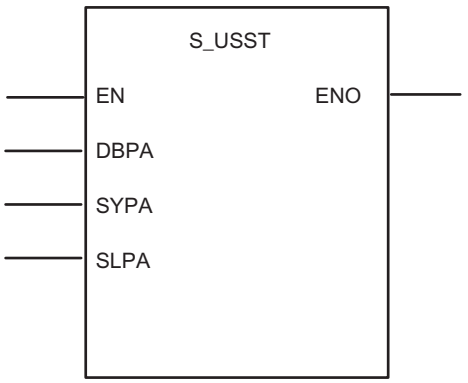
S_USST 的程序结构

下图显示了 S_USST 的程序结构。



图片 3-23　　Modbus 从站诊断功能

表格 3-79　STL 和 LAD 表达式

STL 表达式	LAD 表达式
CALL S_USST DBPA = SYPA = SLPA =	

说明

参数 EN 和 ENO 仅存在于图形表达式（LAD 或 FBD）中。编译器使用二进制结果处理这些参数。

如果块终止且无错，则将二进制结果设置为信号状态“1”。如果出现错误，则将二进制结果设置为“0”。

FC 17 S_USST 参数

下表列出了 S_USST FC 参数。

表格 3-80　S_USST FC 参数

名称	类型	数据类型	说明	注释
DBPA	INPUT	INT	参数 DB 的块编号	依 CPU 而定 （不允许为零）
SYPA	INPUT	INT	参数 DB 中的系统参数的起始地址	0 <= SYPA <= 8174
SLPA	INPUT	INT	参数 DB 中的从站参数的起始地址	0 <= SLPA <= 8184

3.8.6 FC18 S_USSR：接收从站的数据

说明

S_USSR FC 根据所用的网络数据结构处理来自从站的网络数据（PZD 和所有 PKW 数据）的接收。

该 FC 从轮询列表中获取当前从站的参数（参数 DB），并计算 TRANSMIT 块的状态字。

如果当前作业已终止且无错（网络数据 DB 的通讯状态字中的位 9 = 0），则将来自通讯 DB 的接收缓冲区的输入数据传输到网络数据 DB 并进行计算。然后，更新网络数据 DB 中的通讯状态字。

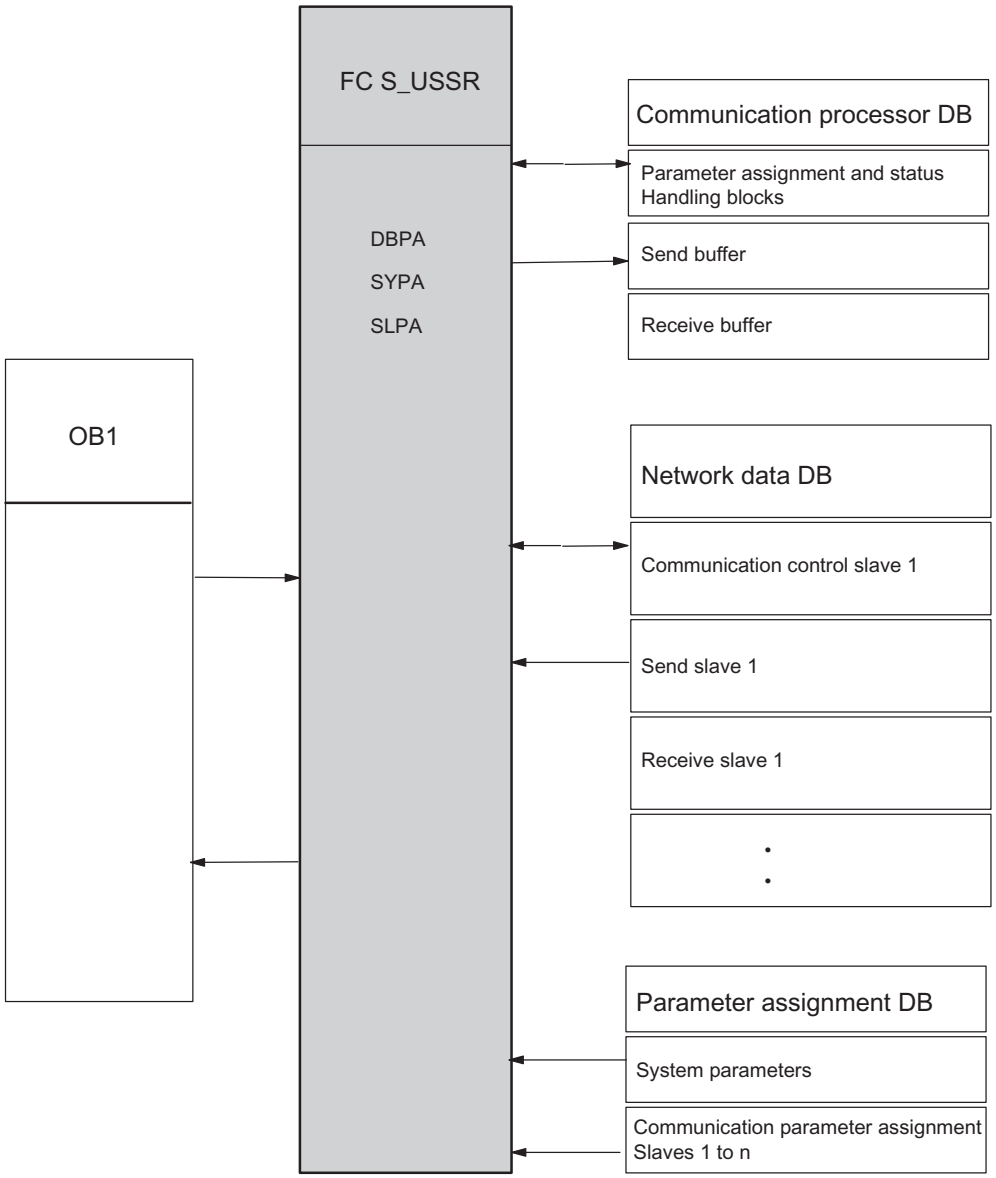
如果当前作业已终止且有错（网络数据 DB 的通讯状态字中的位 9 = 1），则不接受来自通讯处理器 DB 的接收缓冲区的当前从站的数据。FC 18 在网络数据 DB 的通讯状态字中表明该情况，并在通讯错误字中输入出错原因。

如果该块检测到参数 DB 中有参数错误，会将一个错误信号存储在网络数据 DB 的参数错误 1 字节中。

每个自动化系统周期只能调用一次 FC 18。

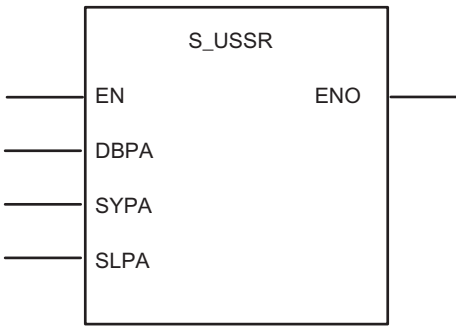
S_USSR 的程序结构

下图显示了 S_USSR 的程序结构。



图片 3-24　　S_USSR 的程序结构

表格 3-81　STL 和 LAD 表达式

STL 表达式	LAD 表达式
CALL S_USSR DBPA = SYPA = SLPA =	

说明

参数 EN 和 ENO 仅存在于图形表达式（LAD 或 FBD）中。编译器使用二进制结果处理这些参数。

如果块终止且无错，则将二进制结果设置为信号状态“1”。如果出现错误，则将二进制结果设置为“0”。

FC 18 S_USSR 参数

下表列出了 S_USSR FC 参数。

表格 3-82　S_USSR FC 参数

名称	类型	数据类型	说明	注释
DBPA	INPUT	INT	参数 DB 的块编号	依 CPU 而定 （不允许为零）
SYPA	INPUT	INT	参数 DB 中的系统参数的起始地址	0 <= SYPA <= 8174
SLPA	INPUT	INT	参数 DB 中的从站参数的起始地址	0 <= SLPA <= 8184

U_USST FC 参数对应于 S_USSR FC 参数。两个功能访问参数 DB 中的同一参数（系统和从站参数），因此必须为其设置相同的参数。

3.8.7 FC19 S_USSI：初始化

说明

S_USSI FC 是可选功能。

如果在 S7 系统启动时调用该 FC，将生成通讯所需的所有通讯处理器 DB、网络数据 DB 和参数 DB。还将预设 DBPA。如果所有从站均具有相同的网络数据结构，则 S_USSI FC 只能用于生成和预设指定的数据区。

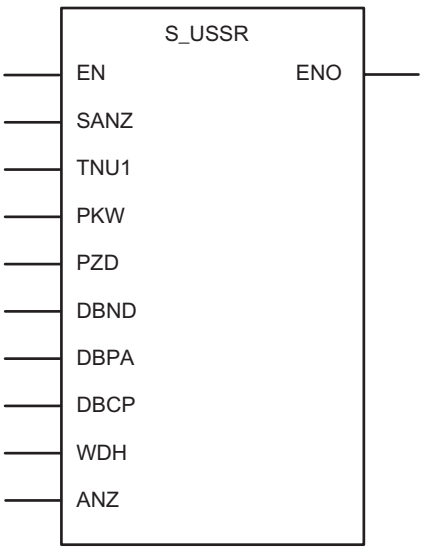
该 FC 在调用时首先针对从站数、网络数据结构、起始节点编号和 PKW 表达式来检查其参数的似然性。如果该块在检查过程中检测到错误，则不会生成或预设数据块。CPU 随即进入 STOP 模式，并且用户会通过 S_USSI FC 的错误字节接收到错误消息。参数错误消除后，必须删除所有已生成的数据块，然后再重新启动。

进行似然性检查后，该块将检查要生成的数据块是否已存在：

- 如果要生成的数据块尚未存在，将生成这些数据块并预设 DBPA。
- 如果要生成的数据块已存在，将检查每个数据块的长度。如果该 DB 足够长，将再次预设参数 DB 并删除网络数据 DB 和通讯处理器 DB 的内容。如果 DB 过短，则 CPU 将进入 STOP 模式。用户可以在 S_USSI FC 的条件代码字节中识别出现故障的 DB。对于要消除的错误，必须完全删除这三个数据块。然后，将在下次重新启动时再次生成这些数据块，并预设参数 DB。

系统启动期间，必须调用 S_USSI 一次 (OB 100)。

表格 3-83　STL 和 LAD 表达式

STL 表达式		LAD 表达式
CALL	S_USSI	
SANZ	=	
TNU1	=	
PKW	=	
PZD	=	
DBND	=	
DBPA	=	
DBCP	=	
WDH	=	
ANZ	=	

说明

参数 EN 和 ENO 仅存在于图形表达式（LAD 或 FBD）中。编译器使用二进制结果处理这些参数。

如果块终止且无错，则将二进制结果设置为信号状态“1”。如果出现错误，则将二进制结果设置为“0”。

FC 19 S_USSI 参数

下表列出了 S_USST FC 参数。

表格 3-84　S_USSI FC 参数

名称	类型	数据类型	说明	注释
SANZ	INPUT	INT	具有相同网络数据结构（DBPA 中的系统参数）的从站数	1 <= SANZ <= 31
TNU1	INPUT	INT	起始节点编号（站号）	0 <= TNU1 <= 31
PKW	INPUT	INT	PKW，数目	PKW 接口的字数，0、3 或 4
PZD	INPUT	INT	PZD，数目	PZD 接口的字数 0 <= PZD <= 16
DBND	INPUT	INT	网络数据 DB 号	依 CPU 而定（不允许为零）。
DBPA	INPUT	INT	参数化 DB 号	依 CPU 而定（不允许为零）。
DBCP	INPUT	INT	通讯处理器 DB 号	依 CPU 而定（不允许为零）。
WDH	INPUT	INT	允许的 PKW 作业重复数	0 <= WDH <= 32,767
ANZ	OUTPUT	BYTE	错误字节	0：没有错误 1：从站数过大 2：不允许的 网络数据结构的条目 3：参数 DB 过短 4：网络数据 DB 过短 5：站号错误 6：通讯 处理器 DB 过短 7：空闲 8：重复计数器： 值不正确

3.8.8 网络数据 DB

说明

可以在 CPU 启动时使用 S_USSI FC 生成和预设这些数据块（仅限 DBPA），也可以手动输入这些数据块。

网络数据 DB 形成了通讯和控制程序之间的接口。用户必须使该块为“空”，并且该块必须足够长。只能将从站的传输数据输入已由控制程序分配给从站的网络数据 DB 的发送缓冲区。从相应的接收缓冲区中接收来自该从站的响应数据（在计算通讯控制字中的位 9 之后）。状态字可用于精确监视通讯，而控制字使您可以精确控制参数分配请求的开始。

对于每一个从站，通讯接口均包含以下数据：

- 从站相关的通讯数据（通讯控制，跟踪，6 个数据字）
- 当前 PKW 作业的缓冲区（仅当存在 PKW 区时）
- 网络数据的发送缓冲区（最多 20 个数据字）
- 网络数据的接收缓冲区（最多 20 个数据字）

发送缓冲区和接收缓冲区的长度取决于所选的网络数据结构。如果不存在 PKW 接口，则不使用当前 PKW 作业的缓冲区。

所需的网络数据 DB 的总长度取决于使用的从站数和网络数据结构。

每个从站的数据字的数目 = 2 x (PKW + PZD) + PKW + 6

其中 PKW = 0、3 或 4 且 0 <= PZD <= 16

示例：具有 3 个字的 PKW 区和 2 个字的 PZD 区的驱动器需要网络数据 DB 中有 19 个数据字。

具有 31 个从站以及最大网络数据长度时，网络数据 DB 的长度为 1,550 个数据字。DBW0 保留。

在 PKW 区中，网络数据 DB 中的从站数据分配为：PKW 区中有 4 个字，而 PZD 区中有 0 到 16 个字

DBWn	通讯控制字 (KSTW)		通讯控制	
DBWn+2	内部			
DBWn+4	通讯状态字		通讯跟踪	
DBWn+6	通讯错误字		错误状态	
DBWn+8	内部		PKW 尝试计数器	
DBWn+10	参数错误 1 字节，参数错误 2 字节		参数错误	
DBWn+12	参数 ID	PKE	用于当前 PKW 作业的缓冲区	
DBWn+14	索引	IND		
DBWn+16	参数值 1	PWE1		
DBWn+18	参数值 2	PWE2		
DBWn+20	参数 ID	PKE	PKW 区	发送缓冲区
DBWn+22	索引	IND		
DBWn+24	参数值 1	PWE1		
DBWn+26	参数值 2	PWE2		
DBWn+28	控制字 (STW)	PZD1	PZD 区 （最多 16 个字的 PZD）	
DBWn+30	主设定值 (HSW)	PZD2		
DBWn+32	设定值/其它控制字	PZD3		
DBWn+34	设定值/其它控制字	PZD4		
...	...			
DBWn+58	设定值/其它控制字	PZD16	PKW 区	接收缓冲区
DBWn+60	参数 ID	PKE		
DBWn+62	索引	IND		
DBWn+64	参数值 1	PWE1		
DBWn+66	参数值 2	PWE2		
DBWn+68	状态字 (ZSW)	PZD1	PZD 区 （最多 16 个字的 PZD）	
DBWn+70	主实际值 (HIW)	PZD2		
DBWn+72	实际值/其它状态字	PZD3		
DBWn+74	实际值/其它状态字	PZD4		
...	...			
DBWn+98	实际值/其它状态字	PZD16		
	•			
（n = 2、4、6...）	•			

说明

如果没有 PKW 区，则当前 PKW 作业的缓冲区以及发送缓冲区中的 PKW 区均不存在。

通讯控制字 KSTW (DBWn)

通讯控制字中的位符合用户程序和 S_USST FC。

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

- 位 0：触发 PKW 作业

  如果发送缓冲区中存在新 PLW 作业并准备进行处理，则由用户设置位 0。接受 PKW 作业后，由 FC 复位该位。

- 位 1：接受参数更改报告

  如果已接受参数更改报告，则由用户设置位 1。由 FC 复位该位以确认该接受。确认后，从站继续处理当前作业或传输下一参数更改报告。

通讯状态字 (DBWn+4)

由 S_USST 和 S_USSR FC 设置通讯状态字中的位。

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

- 位 0：正在处理 PKW 作业

  如果已接受 PKW 作业，则由 S_USST FC 设置位 0，参数 ID (PKE) 包含有效的作业 ID。执行 PKW 作业后（有错或无错），或者 PKW 接口存在问题时，由 S_USSR FC 复位该位。

- 位 1：PKW 作业已完成且无错

  如果 PKW 作业已执行且无错，则由 S_USSR FC 设置位 1。应从接受缓冲区中获得响应。如果已触发新的 PKW 作业，则由 S_USST FC 复位该位。

说明

已轮询列表 (DBPA) 中指定的顺序处理从站的 PKW 作业。一次只能为每个从站激活一个作业。如果在轮询列表中输入了多个从站，则新 PKW 作业的响应数据仅在位 1（或位 2）的正跳沿处有效。

- 位 2：PKW 作业已完成且有错

  对于 PKE 中的响应 ID，由 S_USSR FC 设置位 2。错误编号位于从站响应的 PWE 中。如果已触发新的 PKW 作业，则由 S_USST FC 复位该位。

说明

处理后，由用户传输的上一 PKW 作业存储在发送接口中。重复向从站进行传送，直到输入新作业。如果由于错误（位 2）以及 PKW 接口错误（位 4）而终止状态 PKW 作业时，可能需要用户程序中的其它响应。

- 位 3：PKW 作业 ID 无效。

  如果在 PKE 中设置了作业 ID 15 或者如果在作业 ID 4 中输入了索引 255，则由 S_USST FC 设置位 3。如果使用 PKE 中的有效作业 ID 触发了下一 PKW 作业，则由 S_USST FC 复位该位。

- 位 4：PKW 接口出错（计数器溢出）。

  如果从站未响应可组态重复作业数中的 PKW 作业（参数 DB 中的 WDH 参数）或者 PKE 中的响应 ID 8，则由 S_USSR FC 设置位 4。如果已触发并正确执行新的 PKW 作业，则由 S_USSR FC 复位该位。

- 位 5：响应数据包含参数更改报告。

  如果存在来自从站的参数更改报告，则由 S_USSR FC 设置位 5（对响应 ID 9 到 12 以及切换位 11 取反）。如果用户确认了参数更改报告，则由 S_USST FC 复位该位（通讯控制字，位 1）。

- 位 6：有关从站的运行故障。

  由 S_USSR FC 设置并复位位 6。FC 计算从站的状态字（位 3）。

- 位 7：具有来自从站的警告。

  由 S_USSR FC 设置并复位位 7。FC 计算从站的状态字（位 7）。

- 位 8：需要自动化系统控制。

  由 S_USSR FC 设置并复位位 8。FC 计算状态字（位 9）和控制字（位 10）。

- 位 9：组通讯故障。

  由 S_USSR FC 设置并复位位 9。FC 计算来自 S_SEND 和 S_RCV 标准块的反馈消息，并检查收到的有关 ADR、STX、BCC 和 LGE 的消息帧。此处，FC 还报告已超出消息帧监视时间。

说明

只有位 9 = 0 时，来自网络数据 DB 的接收数据才有效。

通讯错误字 (DBWn+6) 的结构

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

- 位 0：寻址错误 (ADR)
- 位 3：未检测到消息帧的起始（第一个字符不是 STX）
- 位 4：错误的块校验字符 (BCC)
- 位 6：错误的消息帧长度 (LGE)

  如果检查接收的消息帧时出错时，则由 S_USSR FC 设置位 0、3、4 和 6（ADR、STX、BCC、LGE）。
- 位 7：消息帧监视时间已结束

  如果从主站向从站发送消息帧和来自从站的响应到达之间的时间（消息帧监视时间）超出了由程序计算的允许时间，则由 S_USSR FC 设置位 7。

未使用其余位。

参数错误 1 字节

来自 S_USSR FC 的错误消息；参数 DB 中的参数错误

- 值 0：没有错误
- 值 1：PKW/PZD 错误的数据

参数错误 2 字节

来自 S_USST FC 的错误消息；参数 DB 中的参数错误

- 值 0：没有错误
- 值 1：PKW/PZD 错误的数据

发送缓冲区中的参数 ID PKE

用户必须分配参数编号（位 0 到 10）和作业 ID（位 12 到 15）。由功能 S_USSR 和 S_USST 屏蔽参数更改报告的切换位（位 11）。

3.8.9 参数设置 DB

说明

参数 DB 包含通讯控制所需的程序参数。用户必须生成该块并相应地预设通讯系统组态（S_USSI 或手动）。将按从站在 DBPA（轮询列表）中的输入顺序对其进行处理。

也可以在参数 DB 中多次输入一个从站，从而有效地提高其优先级。

参数 DB 的长度取决于要在总线周期中进行寻址的从站数 (n)。
参数 DB 的数据字的数目 = (n x 4) + 5。

每个从站通讯实例需要 4 个数据字，并且一次为系统参数分配 4 个数据字。DBW0 保留。

DBW 0	空闲	系统参数
DBW 2	DBCP	
DBW 4	SANZ	
DBW 6	SLAV	
DBW 8	WDH	
DBW 10	PKW 的数目，PZD 的数目	通讯 参数设置从站 1
DBW 12	TUN	
DBW 14	DBND	
DBW 16	KSTW	
DBW 18	PKW 的数目，PZD 的数目	通讯 参数设置从站 2
DBW 20	TUN	
DBW 22	DBND	
DBW 24	KSTW	
	PKW 的数目，PZD 的数目	通讯 参数设置从站 n
	TUN	
	DBND	
DBW (n x 8 + 8)	KSTW	

系统参数

DBCP	通讯处理器 DB 的块编号
SANZ	参数 DB 中的从站参数设置的总数。如果总线周期中某些从站的寻址频率多于其它从站，则在参数 DB 中输入这些从站的从站参数的次数必须多于一次。必须相应地调整 SANZ 系统参数。
SLAV	当前从站的编号（连续）。S_USST FC 和 S_USSR FC 需要该编号以计算当前参数设置。该数据字必须预设为 1。如果使用 S_USSI FC，则由该 FC 来执行预设操作。
WDH	允许的 PKW 作业重复数（取值范围：0 到 32,767）。如果当前 PKW 作业未在设置的数字处结束，将在 PKW 接口处报告问题。

从站通讯参数

PKW 的数目， PZD 的数目	网络数据结构的定义 左侧字节：PKW 区的字数（0、3、4） 右侧字节：PZD 区的字数（0 到 16） 任何与此不符的情况均被检测为参数错误（由 S_USST 和 S_USSR FC）并在网络数据 DB 的参数错误 1 字节、参数错误 2 字节中输入。
TUN	对应于在驱动器上设置的总线地址的节点编号（0 到 31）。
DBND	网络数据 DB 的块编号。
KSTW	网络数据 DB 中的从站的通讯控制字的地址。

3.8.10 通讯处理器 DB

通讯处理器 DB 的结构

通过该数据块处理 CPU 和 ET 200S 串行接口 Modbus/USS 模块之间的数据交换。用户必须提供具有足够长度的该块。通讯处理器 DB 的长度必须至少为 50 个字（DBW 0 到 98）。

DBW 0	通讯状态		TRANSMIT 和 RECEIVE
DBW 2	等待接收时的最大周期数	等待接收时用户超时计算的周期计数器	FC17
DBW 4	测量的起始脉冲		
DBW 6	上一周期的持续时间 (OB1_MIN_CYCLE)		FC17
DBW 8	发送报文长度 (LEN)		FC17，OB1 TRANSMIT
DBB10	未使用		
DBB 11 : : DBB 54	传输缓冲区		向模块传输帧 （长度取决于当前从站的网络数据结构）
DBB 55 : : DBB 98	接收缓冲区		从模块接收帧 （长度取决于当前从站的网络数据结构）

通讯状态 DBW0

DBW0 包含以下位：

- 位 0：到 S_SEND 的 REQ 输入。

  设置位 8 时复位该位。

- 位 1：到 S_SEND 的 R 输入。

  由 S_USST 周期性复位该位。

- 位 2：自 S_SEND 的 DONE 输出。
- 位 3：自 S_SEND 的 ERROR 输出。
- 位 4：到 S_RCV 的 EN_R 输入。

  由 S_USSR 周期性设置该位。

- 位 5：到 S_RCV 的 R 输入。

  由 S_USSR 周期性复位该位。

- 位 6：自 S_RCV 的 NDR 输出。
- 位 7：自 S_RCV 的 ERROR 输出。
- 位 8：正在进行的请求（存储在 S_SEND 的 DONE 位中）。

  由 S_USST 设置和复位该位。

上一周期 DBW6 的持续时间

S_USST 使用该参数计算从站的响应时间。每次调用 S_USST 之间，用户程序应将该 PLC 扫描周期时间 (OB1_MIN_CYCLE) 复制到该参数中。

3.9 ET 200S 串行接口 Modbus/USS 驱动程序的启动特性和操作模式

3.9.1 装载组态和参数分配数据

数据存储

关闭硬件组态时，数据将自动存储在您的 STEP 7 项目中。

装载组态和参数

可以从编程设备中在线向 CPU 装载组态和参数数据。选择“目标系统”(Target system) > “装载”(Load) 将数据传输到 CPU 中。

CPU 启动时，无论何时在 STOP 模式和 RUN 模式之间进行切换，只要可以通过 S7-300 背板总线访问模块，模块参数均将自动传输到该模块。

模块的保持存储器中的参数接口将保存驱动程序代码。这意味着没有编程设备就无法调换模块。

其它信息

《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。

3.9.2 ET 200S 串行接口 Modbus/USS 模块的操作模式

操作模式

ET 200S Modbus/USS 串行接口模块提供了以下操作模式：

- **STOP**：

  当模块处于 STOP 模式时，没有处于活动状态的协议驱动程序，CPU 的所有发送和接收作业均将得到否定确认。模块将保持 STOP 模式，直到消除导致 STOP 模式的原因（例如断线或无效的参数）。

- **复位参数**：

  复位模块的参数时，将初始化协议驱动程序。复位过程中，SF 组故障 LED 保持亮起。

  无法进行发送和接收操作，驱动程序重新启动时将丢失存储在模块中的发送和接收消息帧。模块和 CPU 之间的通讯将重新启动（取消活动的消息帧）。

  复位参数之后，模块处于 RUN 模式并准备好发送和接收。

- **RUN**：

  该模块处理 CPU 发送作业。可由 CPU 读取从通讯伙伴处接收的消息帧。

3.9.3 ET 200S 串行接口 Modbus/USS 模块的启动特性

启动阶段

启动包含两个阶段：

- **初始化**：只要对模块施加电压，串行接口便进行初始化并等待来自 CPU 的参数数据。
- **参数化**：参数化期间，ET 200S Modbus/USS 串行接口模块接收已分配到 STEP 7 中当前插槽的模块参数。

3.9.4 ET 200S 串行接口 Modbus/USS 模块在 CPU 操作模式转换中的特性

启动后的特性

ET 200S Modbus/USS 串行接口模块启动后，将通过功能块在 CPU 和模块之间交换所有数据。

- **CPU STOP：**

  当 CPU 处于 STOP 模式时，无法通过 PROFIBUS 进行通讯。将取消 ET 200S Modbus/USS 串行接口模块和 CPU 之间所有活动的数据传输（包括发送和接收消息帧），并重新建立连接。

- **CPU 启动：**

  启动期间，CPU 向该模块传输参数。

  可以通过分配适当的参数在 CPU 启动时自动清除模块的接收缓冲区。

- **CPU RUN：**

  当 CPU 处于 RUN 模式时，发送和接收操作不受限制。在 CPU 重新启动后的前几个 FB 周期中，模块和相应的 FB 保持同步。只有完成该过程才能执行新的 S_SEND 或 S_RCV FB。

发送消息帧时的注意事项

只能在 RUN 模式下发送消息帧。

如果数据正在从 CPU 向模块传输时将 CPU 切换为 STOP 模式，则暖启动后将会出现 S_SEND 输出错误 (05) 02_H。为了防止该情况发生，用户程序可以使用启动 OB 中的 RESET 输入调用 S_SEND FB。

说明

ET 200S/USS 串行接口模块只有在其已接收来自该模块的所有数据后才向通讯伙伴发送数据。

接收消息帧时的注意事项

可以使用 STEP 7 设置参数“启动时清空模块接收缓冲区 = 是/否”。

- 如果已设置为“是”参数，则 CPU 从 STOP 切换为 RUN 模式时将自动清除 ET 200S Modbus/USS 串行接口模块的接收缓冲区。
- 如果已设置为“否”参数，则将在 ET 200S Modbus/USS 串行接口模块的接收缓冲区中缓存消息帧。

如果数据正在从 CPU 向 ET 200S Modbus/USS 串行接口模块传输时将 CPU 切换为 STOP 模式，则暖启动后将会出现 S_RCV 输出错误 (05) 02_H。为了防止此情况发生，用户程序可以使用启动 OB 中的 RESET 输入调用 S_SEND FB。当“启动期间清除 ET 200S Modbus/USS 串行接口模块的接收缓冲区 = 否”时，ET 200S Modbus/USS 串行接口模块将再次向 CPU 传输消息帧。

处理时间

可如下计算完成主站-从站处理所需的时间（包括数据更新时间）：

- 总处理时间 (t_8) = 主站作业处理时间 (t_1) + 主站作业发送时间 (t_2) + 从站作业处理时间 (t_3) + 1 个 CPU 周期（处理功能代码的时间）(t_4) + 从站响应处理时间 (t_5) + 从站响应发送时间 (t_6) + 主站响应处理时间 (t_7)

作业/响应处理时间

对于主站和从站，发送和接收时间的计算公式相同。可如下计算该时间：

- 如果 CPU 值远大于（I/O 周期 + 10 ms），则处理时间 = 每 7 个字节 1 个 CPU 周期；否则，处理时间 = 每 7 个字节（2 个 CPU 周期 ＋ 3 个 I/O 周期 ＋ 10 ms）

发送时间/作业的接收时间/响应时间

可如下计算发送/接收作业/响应所需的时间：

- 发送/接收时间 = 10 ms + 传输率 × 消息中的字符数

表格 3-85　总处理时间示例：

读取	波特率	I/O 周期	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8
10 个字	9,600 bps	2 ms	40 ms	12 ms	40 ms	40 ms	160 ms	29 ms	160 ms	483 ms

3.10 技术规范

协议和接口的技术数据

表格 3-86 ET 200S Modbus/USS 模块协议和接口的技术数据

常规技术数据	
指示器元素	绿色 LED，TX（发送） 绿色 LED，RX（接收） 红色 LED，SF（系统错误）
提供的协议驱动程序	Modbus 驱动程序 USS 驱动程序
Modbus 协议波特率 USS 驱动程序波特率	110, 300, 600, 1,200, 2,400, 4,800, 9,600, 19,200, 38,400, 57,600, 76,800, 115,200
字符帧（11 位）	每个字符的位数：8 启动/停止位数：1 或 2 奇偶校验：无、偶数、奇数、任意
标准块 (FB) 的存储器要求	发送和接收：约 4,300 个字节
RS 232C 接口的技术数据	
接口	RS 232C，8 个端子
RS 232C 信号	TXD、RXD、RTS、CTS、DTR、DSR、DCD、PE 全部与 ET 200S 1SI 模块的内部电源电气隔离。
最大传输距离	15 m
RS 422/485 接口的技术数据	
接口	RS 422，5 个端子 RS 485，3 个端子
RS 422 信号 RS 485 信号	TXD (A)、RXD (A)、TXD (B)、RXD (B)、PE R/T (A)、R/T (B)、PE 全部与 ET 200S 1SI 模块的内部电源电气隔离。
最大传输距离	1,200 m

Modbus/USS 的技术数据

表格 3-87　ET 200S Modbus/USS 模块的常规技术数据

常规技术数据	
尺寸和重量	
尺寸：宽度 x 高度 x 深度（单位：mm）	15 × 81 × 52
重量	约 50 g
模块特定的数据	
RS 232C • 输入数 • 输出数	 4 3
RS 422 • 输入对数 • 输出对数	 1 1
RS 485 • I/O 对数	 1
电缆长度 • 已屏蔽 (RS 232C) • 已屏蔽 (RS 422/485)	 最大 15 m 最大 1,200 m
防护等级 [1]	IEC 801-5
电压、电流、电位	
电子装置的额定电源电压 (L+)	24 V DC
• 反极性保护	是
电位隔离	
• 通道和背板总线之间	是
• 通道和电子装置的电源之间	是
• 通道之间	否
• 通道和 PROFIBUS DP 之间	是
绝缘测试电压 • 通道到背板总线及通道到负载电压 L+ • 负载电压 L+ 到背板总线	 500 V DC 500 V AC
电流源 • 来自背板总线 • 来自电源 L+	 最大 10 mA 最大 80 mA（典型值）20 mA
模块的功耗	通常为 1.2 W

常规技术数据	
状态、中断、诊断	
状态指示器	绿色 LED (TX) 绿色 LED (RX)
诊断功能	
• 组故障显示 • 可以显示诊断信息	可能为红色 LED (SF)
输出	
输出，RS 232C 范围	± 最大 10 V
• 适用于容性负载 • 短路保护 • 短路电流 • PE（接地）输出或输入的电压	最大 2500 pF 是 约 60 mA 最大 25 V
输出，RS 422/485	
负载阻抗 • 短路保护 • 短路电流	最小 50 kΩ 是 约 60 mA
[1] 用户提供的输入线路链接中所需的外部保护设备： • Blitzductor 标准安装导轨连接器 • Blitzductor 保护模块 KT AD-24V	

索引

数字

A

B

C

F

G

J

K

M

Q

R

S

T

U

X

Y

Z