S7-300/400 PLC 应用技术

廖常初 主编

本书配套光盘中包含：
中英文软件手册、硬件手册、通信手册、例程和软件

机械工业出版社
CHINA MACHINE PRESS

ISBN 7-11
策划：胡毓坚
封面设计：饶　薇

机械工业出版社出版的相关图书：

- S7-300/400 PLC 应用技术　书号：15530　定价：48.00 元
- 大中型 PLC 应用教程
- PLC 编程及应用　书号：10877　定价：26.00 元
- PLC 基础及应用　书号：12295　定价：20.00 元
- 可编程控制器应用技术　书号：09251　定价：22.00 元

ISBN 7-111-15530-0
9 787111 155300 >
定价：48.00 元(含 1CD)

S7-300/400 PLC 应用技术

廖常初　主编

机 械 工 业 出 版 社

西门子的 S7-300/400 是应用最广的大中型 PLC，本书介绍：S7-300/400 的硬件结构、性能指标和硬件组态的方法；指令系统、程序结构、编程软件 STEP 7 的使用方法；梯形图的经验设计法、继电器电路转换法和顺序控制设计法，以及使用顺序功能图语言 S7 Graph 的设计方法。这些设计方法易学易用，可以节约大量的设计时间。本书还介绍了 S7-300/400 的网络结构，AS-i 和工业以太网，详细介绍了 MPI、PROFIBUS 网络、点对点通信、PRODAVE 通信软件的组态、参数设置和编程的方法，以及使用系统功能块实现 PID 控制的方法。

本书包含了 S7-300/400 的编程手册和常用的用户手册中的主要内容。配套的光盘附有大量的中英文用户手册、软件和例程。本书介绍了基于 STEP 7 编程软件和 PLCSIM 仿真软件的学习和实验的方法，通过这种方法没有 PLC 也可以较快地掌握 S7-300/400 的使用方法。

本书注重实际，强调应用，可供工程技术人员自学和作为培训教材使用，对 S7-300/400 的用户有很大的参考价值。也可以作为大专院校有关专业的参考教材。

图书在版编目（CIP）数据

S7-300/400 PLC应用技术/廖常初主编. —北京：机械工业出版社，2005.1
ISBN 7-111-15530-0

Ⅰ. S... Ⅱ. 廖... Ⅲ. 可编程序控制器
Ⅳ. TP332.3

中国版本图书馆 CIP 数据核字（2004）第 111901 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）
策 划：胡毓坚
责任编辑：李馨馨
责任印制 石 冉

北京中兴印刷有限公司印刷·新华书店北京发行所发行

2005 年 3 月第 1 版·第 2 次印刷
787mm×1092mm 1/16·28 印张·690 千字
5001-12 000 册
定价：48.00 元（含 1CD）

凡购本图书，如有缺页、倒页、脱页，由本社发行部调换
本社购书热线电话：（010）68326294
封面无防伪标均为盗版

前　言

可编程序控制器(PLC)是应用十分广泛的通用微机控制装置,是自动控制系统中的关键设备。西门子公司的 S7-300/400 在大中型 PLC 中应用最广,市场占有率最高。S7-300/400 及其编程软件 STEP 7 和通信网络的功能强大,程序结构复杂,其用户手册和编程手册有好几十本,中文资料很少,要求用户具有较高的计算机应用能力和英语水平,很多人觉得西门子 PLC 的技术门槛太高,初学者入门非常困难。因此,广大工程技术人员和大专院校师生迫切需要一本能帮助他们学习 S7-300/400 的书籍。

本书作者编著的 PLC 教材已发行 10 万余册,其中以 S7-200 为背景的《PLC 编程及应用》在两年中已重印 5 次,该书在华储网上电脑书店(www.huachu.com.cn)销售排行榜的自动化类书籍中稳居第 1 名。

本书对 S7-300/400 的硬件、编程语言、指令、程序结构、编程软件、通信网络和闭环控制等方面都作了较为全面深入的介绍。书中包含了 S7-300/400 的语句表、梯形图编程手册和用户手册中的主要内容。加上配套的光盘中大量的编程手册、用户手册和一些软件,以及作者编写的与书中内容配套的例程(见附录中的光盘说明),本书既能帮助初学者入门,又是工程应用中内容丰富的资料库。

为了保证内容的准确性和新颖性,本书编写时以英文资料为准,参考了西门子公司提供的和在西门子网站(包括在德国的网站)下载的大量的最新资料,同时参考了编程软件 STEP 7 (V5.2)内容丰富的帮助文件和例程。书中介绍的各种组态、编程和监控操作都经过编程软件的检验,绝大部分程序都用仿真软件 PLCSIM 或 PLC 作了验证。

本书前 5 章是基础篇,第 5 章介绍了一整套先进完整的开关量控制系统的编程方法。它们易学易用,是作者在长期教学、科研和 PLC 工程应用实践中总结出来的。用它们能很容易地设计出任意复杂的开关量控制系统的梯形图,可以节约大量的设计时间。

第 5 章通过实例首次详细地介绍了使用顺序功能图语言 S7 Graph 的编程方法,S7 Graph 的功能强大,使用方便,是设计顺序控制程序的理想工具。

通信是当今自动控制系统设计和应用的重点和难点。本书全面介绍了 S7-300/400 的各种通信网络、通信模块和软件工具,包括 MPI、PROFIBUS、工业以太网和 AS-i 通信网络,点对点通信和通信软件 PRODAVE,通信网络的组态和通信的编程方法。光盘中作者编写的通信测试软件可以用于 PLC 与计算机的通信实验。

本书详细介绍了模拟量闭环控制的有关问题,如模拟量模块的使用,PID 控制功能块在闭环控制中的应用和闭环系统的软件仿真,PID 控制器的参数整定方法等。

学好 S7-300/400 的关键是实践,由于昂贵的培训费用和硬件价格,一般人很难有用大量的 PLC 硬件进行实际操作的机会。本书详细介绍了基于 STEP 7 编程软件和 PLCSIM 仿真软件的实验方法,通过这种方法可以较快地掌握用 STEP 7 对 S7-300/400 的硬件和通信网络进行组态和设置参数的方法,用 PLCSIM 在计算机上可以模拟运行和监控 PLC 的用户程序,使读者可以较快地掌握 S7-300/400 的使用方法。

本书的光盘中没有 STEP 7，光盘中的 STEP 7 _ V5 _ 1 _ SP6.zip 和 STEP 7 _ V52 _ SP1.zip 是补丁程序，安装了 STEP 7 后才能安装它们。建议读者通过西门子产品的经销商获得 STEP 7 的演示版光盘，也可以在网上搜索有关的信息。演示版的功能与正式版没有区别，只是在使用过程中会经常搜索授权。

为了方便使用，在附录中提供了指令一览表、组织块、系统功能与系统功能块一览表、光盘说明和常用缩写词。

本书的篇幅较大，内容较多，较适合企业的技术人员使用。为了满足大专院校教学的需要，机械工业出版社已经出版了本书的教材版《大中型 PLC 应用教程》（书号为 15849），该书的篇幅适中，配有习题和实验指导书。

本书的编写得到了西门子（中国）有限公司的大力支持，马笑潇经理和王东滨先生对本书的编写提供了大量的资料、编程软件和实验用的硬件，在此表示衷心的感谢。

本书由廖常初主编，陈晓东、王云杰、李远树、周林、陈曾汉、侯世英、郑连清、范占华、徐波、孙伟、陈俊明、刘玉孝、张嵩、关朝旺、郑群英、余秋霞、张学锋、申敏、罗盛波、廖亮、孙明渝、唐世友、万莉、左源洁、孙剑、聂世珍参加了编写工作。

因作者水平有限，书中难免有错漏之处，恳请读者批评指正。

作者 E-mail：liaosun@cqu.edu.cn。

重庆大学电气工程学院　廖常初
2004 年 8 月

目　　录

第1章 概 述

1.1 PLC的基本概念

随着微处理器、计算机和数字通信技术的飞速发展，计算机控制已经广泛地应用在几乎所有的工业领域。现代社会要求制造业对市场需求作出迅速的反应，生产出小批量、多品种、多规格、低成本和高质量的产品，为了满足这一要求，生产设备和自动生产线的控制系统必须具有极高的可靠性和灵活性，可编程序控制器正是顺应这一要求出现的，它是以微处理器为基础的通用工业控制装置。

可编程序控制器(Programmable Logic Controller)简称为PLC，它的应用面广、功能强大、使用方便，已经成为当代工业自动化的主要支柱之一，在工业生产的所有领域得到了广泛的使用。

国际电工委员会(IEC)在1985年的PLC标准草案第3稿中，对PLC作了如下定义："可编程序控制器是一种数字运算操作的电子系统，专为在工业环境下应用而设计。它采用可编程序的存储器，用来在其内部存储执行逻辑运算、顺序控制、定时、计数和算术运算等操作的指令，并通过数字式、模拟式的输入和输出，控制各种类型的机械或生产过程。可编程序控制器及其有关设备，都应按易于使工业控制系统形成一个整体，易于扩充其功能的原则设计。"

PLC已经广泛地应用在各种机械设备和生产过程的自动控制系统中，PLC在其他领域，例如在民用和家庭自动化设备中的应用也得到了迅速的发展。

1.1.1 模块式PLC的基本结构

本书以西门子公司的S7-300/400系列大中型PLC为主要讲授对象。西门子的PLC以其极高的性能价格比，在国内占有很大的市场份额，在我国的各行各业得到了广泛的应用。S7-300/400属于模块式PLC，主要由机架、CPU模块、信号模块、功能模块、接口模块、通信处理器、电源模块和编程设备组成(见图1-1)，各种模块安装在机架上。通过CPU模块或通信模块上的通信接口，PLC被连接到通信网络上，可以与计算机、其他PLC或其他设备通信。

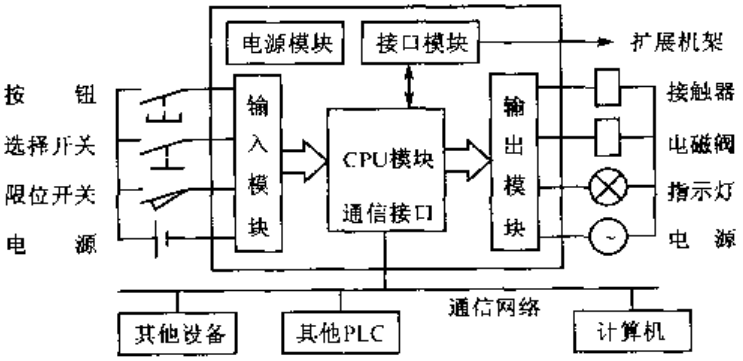


图1-1 PLC控制系统示意图

1. CPU 模块

CPU 模块主要由微处理器(CPU 芯片)和存储器组成。在 PLC 控制系统中,CPU 模块相当于人的大脑和心脏,它不断地采集输入信号,执行用户程序,刷新系统的输出;存储器用来储存程序和数据。S7-300/400 将 CPU 模块简称为 CPU。

2. 信号模块

输入(Input)模块和输出(Output)模块简称为 I/O 模块,开关量输入、输出模块简称为 DI 模块和 DO 模块,模拟量输入、输出模块简称为 AI 模块和 AO 模块,它们统称为信号模块。信号模块是系统的眼、耳、手、脚,是联系外部现场设备和 CPU 模块的桥梁。

输入模块用来接收和采集输入信号,开关量输入模块用来接收从按钮、选择开关、数字拨码开关、限位开关、接近开关、光电开关、压力继电器等来的开关量输入信号;模拟量输入模块用来接收电位器、测速发电动机和各种变送器提供的连续变化的模拟量电流电压信号。

开关量输出模块用来控制接触器、电磁阀、电磁铁、指示灯、数字显示装置和报警装置等输出设备,模拟量输出模块用来控制电动调节阀、变频器等执行器。

CPU 模块内部的工作电压一般是 DC 5V,而 PLC 的输入/输出信号电压一般较高,例如 DC 24V 或 AC 220 V。从外部引入的尖峰电压和干扰噪声可能损坏 CPU 模块中的元器件,或使 PLC 不能正常工作。在信号模块中,用光耦合器、光敏晶闸管、小型继电器等器件来隔离 PLC 的内部电路和外部的输入、输出电路。信号模块除了传递信号外,还有电平转换与隔离的作用。

3. 功能模块

为了增强 PLC 的功能,扩大其应用领域,减轻 CPU 的负担,PLC 厂家开发了各种各样的功能模块。它们主要用于完成某些对实时性和存储容量要求很高的控制任务。

4. 接口模块

CPU 模块所在的机架称为中央机架,如果一个机架不能容纳全部模块,可以增设一个或多个扩展机架。接口模块用来实现中央机架与扩展机架之间的通信,有的接口模块还可以为扩展机架供电。

5. 通信处理器

通信处理器用于 PLC 之间、PLC 与远程 I/O 之间、PLC 与计算机和其他智能设备之间的通信,可以将 PLC 接入 MPI、PROFIBUS-DP、AS-i 和工业以太网,或者用于实现点对点通信等。CPU 模块集成有 MPI 通信接口,有的还集成了其他通信接口。

6. 电源模块

PLC 一般使用 AC 220V 电源或 DC 24V 电源,电源模块用于将输入电压转换为 DC 24V 电压和背板总线上的 DC 5V 电压,供其他模块使用。

7. 编程设备

S7-300/400 使用安装了编程软件 STEP 7 的个人计算机作为编程设备,在计算机屏幕上直接生成和编辑各种文本程序或图形程序,可以实现不同编程语言之间的相互转换。程序被编译后下载到 PLC,也可以将 PLC 中的程序上传到计算机。程序可以存盘或打印,通过网络,可以实现远程编程和传送。编程软件还具有对网络和硬件组态、参数设置、监控和故障诊断等功能。

1.1.2 PLC的特点

1. 编程方法简单易学

梯形图是使用得最多的PLC的编程语言，其电路符号和表达方式与继电器电路原理图相似，梯形图语言形象直观，易学易懂，熟悉继电器电路图的电气技术人员只需花几天时间就可以熟悉梯形图语言，并用来编制用户程序。

2. 功能强，性能价格比高

一台小型PLC内有成百上千个可供用户使用的编程元件，可以实现非常复杂的控制功能。与相同功能的继电器系统相比，具有很高的性能价格比。PLC可以通过通信联网，实现分散控制，集中管理。

3. 硬件配套齐全，用户使用方便，适应性强

PLC产品已经标准化、系列化、模块化，配备有品种齐全的各种硬件装置供用户选用，用户能灵活方便地进行系统配置，组成不同功能、不同规模的系统。PLC的安装接线也很方便，一般用接线端子连接外部接线。PLC有较强的带负载能力，可以直接驱动一般的电磁阀和中小型交流接触器。

硬件配置确定后，通过修改用户程序，就可以方便快速地适应工艺条件的变化。

4. 可靠性高，抗干扰能力强

传统的继电器控制系统中使用了大量的中间继电器、时间继电器。由于触点接触不良，容易出现故障。PLC用软件代替中间继电器和时间继电器，仅剩下与输入和输出有关的少量硬件元件，接线可减少到继电器控制系统的十分之一以下，大大减少了因触点接触不良造成的故障。

PLC使用了一系列硬件和软件抗干扰措施，具有很强的抗干扰能力，平均无故障时间达到数万小时以上，可以直接用于有强烈干扰的工业生产现场，PLC已被广大用户公认为最可靠的工业控制设备之一。

5. 系统的设计、安装、调试工作量少

PLC用软件功能取代了继电器控制系统中大量的中间继电器、时间继电器、计数器等器件，使控制柜的设计、安装、接线工作量大大减少。

PLC的梯形图程序可以用顺序控制设计法来设计。这种设计方法很有规律，很容易掌握。对于复杂的控制系统，如果掌握了正确的设计方法，设计梯形图的时间比设计继电器系统电路图的时间要少得多。

可以在实验室模拟调试PLC的用户程序，用小开关来模拟输入信号，通过各输出点对应的发光二极管的状态来观察输出信号的状态。完成了系统的安装和接线后，在现场调试过程中，一般通过修改程序就可以解决发现的问题，系统的调试时间比继电器系统少得多。

6. 维修工作量小，维修方便

PLC的故障率很低，并且有完善的故障诊断功能。PLC或外部的输入装置和执行机构发生故障时，可以根据PLC上的发光二极管或编程软件提供的信息，方便地查明故障的原因，用更换模块的方法可以迅速地排除故障。

7. 体积小，能耗低

对于复杂的控制系统，使用PLC后，可以减少大量的中间继电器和时间继电器，小型PLC的体积仅相当于几个继电器的大小，因此可以将开关柜的体积缩小到原来的1/2～1/10。

PLC控制系统与继电器控制系统相比，配线用量少，安装接线工时短，加上开关柜体积的缩小，因此可以节省大量的费用。

1.1.3 PLC的应用领域

在发达的工业国家，PLC已经广泛地应用在所有的工业部门，随着其性能价格比的不断提高，应用范围不断扩大，主要有以下几个方面：

1. 开关量逻辑控制

PLC具有“与”、“或”、“非”等逻辑指令，可以实现触点和电路的串、并联，代替继电器进行组合逻辑控制、定时控制与顺序逻辑控制。开关量逻辑控制可以用于单台设备，也可以用于自动生产线，其应用领域已遍及各行各业，甚至深入到民用和家庭中。

2. 运动控制

PLC使用专用的指令或运动控制模块，对直线运动或圆周运动的位置、速度和加速度进行控制，可以实现单轴、双轴、3轴和多轴联动的位置控制，使运动控制与顺序控制功能有机结合在一起。PLC的运动控制功能广泛用于各种机械，例如金属切削机床、金属成形机械、装配机械、机器人、电梯等场合。

3. 闭环过程控制

闭环过程控制是指对温度、压力、流量等连续变化的模拟量的闭环控制。PLC通过模拟量I/O模块，实现模拟量(Analog)和数字量(Digital)之间的A/D转换与D/A转换，并对模拟量实行闭环PID(比例-积分-微分)控制。S7-300/400有闭环控制模块、用于闭环控制的系统功能块和闭环控制软件包供用户选用。其闭环控制功能已经广泛地应用于塑料挤压成形机、加热炉、热处理炉、锅炉等设备，以及轻工、化工、机械、冶金、电力、建材等行业。

4. 数据处理

现代的PLC具有整数四则运算、矩阵运算、函数运算、字逻辑运算、求反、循环、移位、浮点数运算等运算功能，和数据传送、转换、排序、查表、位操作等功能，可以完成数据的采集、分析和处理。这些数据可以与存储在存储器中的参考值比较，也可以用通信功能传送到别的智能装置，或者将它们打印制表。

5. 通信联网

PLC的通信包括PLC与远程I/O之间的通信、多台PLC之间的通信、PLC与其他智能控制设备(例如计算机、变频器、数控装置)之间的通信。PLC与其他智能控制设备一起，可以组成“集中管理、分散控制”的分布式控制系统。

1.1.4 PLC的主要生产厂家

我国有不少的厂家研制和生产过PLC，但是还没有出现有影响力和有较大市场占有率的产品，目前我国使用的PLC几乎都是国外品牌的产品。

在全世界上百个PLC制造厂中，有几家举足轻重的公司。它们是德国的西门子(Siemens)公司，美国Rockwell自动化公司所属的A-B(Allen & Bradly)公司，GE-Fanuc公司，法国的施耐德(Schneider)公司，日本的三菱公司和欧姆龙(OMRON)公司。

与个人计算机相比，PLC在标准化方面的工作做得较差，PLC的软、硬件体系结构是封闭的而不是开放的，绝大多数PLC使用专用的总线、专用通信网络及协议，各种PLC产品的编

程语言在表示方式、寻址方式和语法结构上都不一致，使得各种 PLC 互不兼容。国际电工委员会(IEC)的 IEC 61131-3(PLC 的编程软件标准)为 PLC 编程的标准化铺平了道路。

目前有的厂家已经推出了符合或基本符合 IEC 61131-3 标准的编程软件，有的厂家正在开发之中。但是仍然有相当多的 PLC 产品的编程语言与 IEC 61131-3 有较大的差异。尽管如此，各种 PLC 产品在软件上还是比较接近的，掌握了一种 PLC 的编程语言，其他的 PLC 编程语言就比较容易理解了。

1.1.5 怎样下载西门子 PLC 的资料和软件

可以在西门子自动化与驱动集团的中文网站(www.ad.siemens.com.cn)下载西门子的 PLC 资料。在该网站的主页中点击“中文下载目录”、“英文下载资料”或“软件下载目录”，进入“下载中心”后，可以下载各种工控产品的中英文说明书、使用手册、产品介绍和一些软件。也可以用网址 http://www.ad.siemens.com.cn/download/直接进入下载中心。

如果需要更多的资料和软件，可以访问西门子自动化与驱动集团在德国的网站，网址为 http://www.ad.siemens.de/。点击“English”将语言由德文改为英文，点击“Service & Support”，在“Document type”下面点击“Manual”，在“Please Type Your Question”下面的方框内输入要搜索的手册的关键字，例如“CP 5511”，点击【GO】按钮，就会列出与 CP 5511 有关的手册。点击感兴趣的手册，在出现的画面中点击“Download”，可以下载该手册；如果在计算机中安装了 Adobe 阅读器，点击【Display】，可以阅读该手册。

从这两个网站还可以下载一些软件的补丁，以及与某些模块配套的软件、例程和使用手册等。

1.2 PLC 的工作原理

1.2.1 逻辑运算

在数字量(或称开关量)控制系统中，变量仅有两种相反的工作状态，例如高电平和低电平、继电器线圈的通电和断电、触点的接通和断开，可以分别用逻辑代数中的 1 和 0 来表示这些状态，在波形图中，用高电平表示 1 状态，用低电平表示 0 状态。

使用继电器电路或 PLC 的梯形图可以实现数字量的逻辑运算。图 1-2 的上面是 PLC 的梯形图，下面是对应的数字门电路。

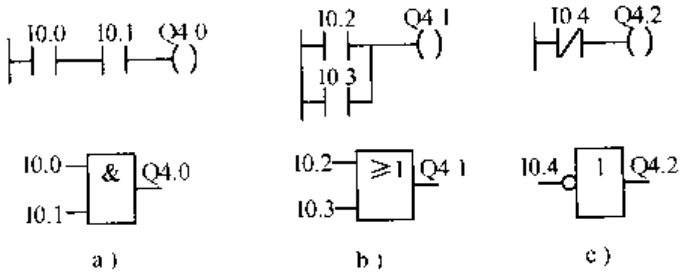


图 1-2 基本逻辑运算

a) 与 b) 或 c) 非

图 1-2 中的 I0.0～I0.4 为数字输入变量，Q4.0～Q4.2 为数字输出变量，它们之间的“与”、“或”、“非”逻辑运算关系如表 1-1 所示。用继电器电路或梯形图可以实现基本的逻辑运

算，触点的串联可以实现“与”运算，触点的并联可以实现“或”运算，用常闭触点控制线圈可以实现“非”运算。多个触点的串、并联电路可以实现复杂的逻辑运算，例如图 1-3 中的继电器电路实现的逻辑运算可以用逻辑代数表达式表示为

$$KM = (SB1 + KM) \cdot \overline{SB2} \cdot \overline{FR}$$

式中的加号表示逻辑或，乘号（小圆点）表示逻辑与，变量上面的横线表示“非”运算。

表 1-1　逻辑运算关系表

与			或			非	
Q4.0 = I0.0·I0.1			Q4.1 = I0.2 + I0.3			Q4.2 = $\overline{I0.4}$	
I0.0	I0.1	Q4.0	I0.2	I0.3	Q4.1	I0.4	Q0.2
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1		
1	1	1	1	1	1		

1.2.2　PLC 的循环处理过程

1. PLC 的循环处理过程

CPU 中的程序分为操作系统和用户程序。操作系统用来处理 PLC 的起动、刷新输入/输出过程映像区、调用用户程序、处理中断和错误、管理存储区和通信等任务。

用户程序由用户生成，用来实现用户要求的自动化任务。STEP 7 将用户编写的程序和程序所需的数据放置在块中，功能块 FB 和功能 FC 相当于用户编写的子程序，系统功能块 SFC 和系统功能块 SFB 是操作系统提供给用户使用的标准子程序，这些块统称为逻辑块。

PLC 采用循环执行用户程序的方式，这种运行方式也称为扫描工作方式。OB1 是用于循环处理的组织块，相当于用户程序中的主程序，它可以调用别的逻辑块，或被中断程序（组织块）中断。

PLC 得电或由 STOP 模式切换到 RUN 模式时，CPU 执行起动操作，清除没有保持功能的位存储器、定时器和计数器，清除中断堆栈和块堆栈的内容，复位保存的硬件中断等。此外还要执行一次用户编写的“系统启动组织块”OB100，完成用户指定的初始化操作。以后将进入周期性的循环运行。下面是循环处理各个阶段的任务（见图 1-4）：

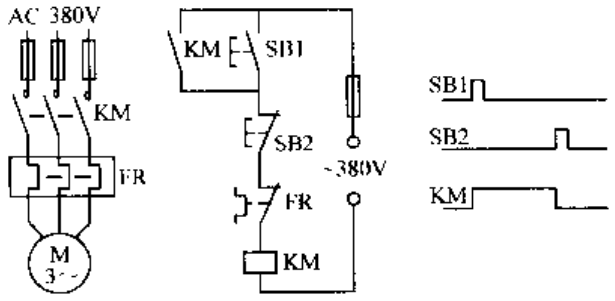


图 1-3　异步电动机控制电路

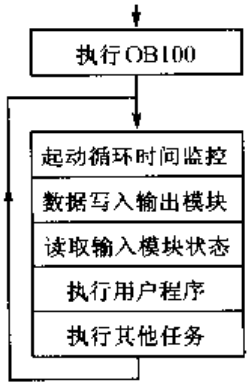


图 1-4　扫描过程

（1）操作系统启动循环时间监控；

（2）CPU 将输出过程映像区的数据写到输出模块。

(3) CPU 读取输入模块的输入状态，并存入输入过程映像区。

(4) CPU 处理用户程序，执行用户程序中的指令。

(5) 在循环结束时，操作系统执行所有挂起的任务，例如下载和删除块，接收和发送全局数据等。

(6) CPU 返回第一阶段，重新启动循环时间监控。

在启动完成后，不断地循环调用 OB1，在 OB1 中可以调用其他逻辑块(FB，SFB，FC 或 SFC)。OB1 具有很低的优先级，除了 OB90 外，所有的组织块都能中断 OB1。

循环程序处理过程可以被某些事件中断。如果有中断事件出现，当前正在执行的块被暂停执行，并调用分配给该事件的组织块。该组织块被执行完后，被暂停执行的块将从被中断的地方开始继续执行。

在 PLC 的存储器中，设置了一片区域用来存放输入信号和输出信号的状态，它们分别称为输入过程映像区和输出过程映像区。PLC 梯形图中的其他编程元件也有对应的映像存储区。

在循环程序处理过程中，CPU 并不直接访问 I/O 模块中的输入地址区和输出地址区，而是访问 CPU 内部的过程映像区。

在写输出模块阶段，CPU 将输出过程映像区的状态传送到输出模块。梯形图中某一输出位(例如 Q4.0)的线圈“通电”时，对应的输出过程映像位为 1 状态。信号经输出模块隔离和功率放大后，继电器型输出模块中对应的硬件继电器的线圈通电，其常开触点闭合，使外部负载通电工作。

若梯形图中输出位的线圈“断电”，对应的输出过程映像位为 0 状态，在写输出模块阶段之后，继电器型输出模块中对应的硬件继电器的线圈断电，其常开触点断开，外部负载断电，停止工作。

在读输入模块阶段，PLC 把所有外部输入电路的接通/断开状态读入输入过程映像区。

外部输入电路接通时，对应的输入过程映像位(例如 I0.0)为 1 状态，梯形图中对应的输入位的常开触点接通，常闭触点断开。外部输入触点电路断开时，对应的输入过程映像位为 0 状态，梯形图中对应的输入位的常开触点断开，常闭触点接通。

某一编程元件对应的过程映像位为 1 状态时，称该编程元件为 ON，过程映像位为 0 状态时，称该编程元件为 OFF。

在程序执行阶段，即使外部输入信号的状态发生了变化，输入过程映像位的状态也不会随之而变，输入信号变化了的状态只能在下一个循环扫描周期的读输入模块阶段被读入。

PLC 的用户程序由若干条指令组成，指令在存储器中顺序排列。在没有跳转指令和块调用指令时，CPU 从第一条指令开始，逐条顺序地执行用户程序，直到用户程序结束之处。在执行指令时，从输入过程映像区或别的存储区中将有关编程元件的 0、1 状态读出来，并根据指令的要求执行相应的逻辑运算，运算的结果写入到对应的存储区中，因此，各编程元件对应的存储区(输入过程映像区除外)的内容随着程序的执行而变化。

2. 扫描循环时间

循环时间(Cycle Time)是指操作系统执行一次如图 1-4 所示的循环操作所需的时间，包括执行 OB1 中的程序段和中断该循环的系统操作的时间，循环时间又称为扫描循环时间(Scan Cycle Time)或扫描周期。循环时间与用户程序的长短、指令的种类和 CPU 执行指令的速度

有很大的关系。当用户程序较长时，指令执行时间在循环时间中占相当大的比例。

在 PLC 处于运行模式时，利用编程软件的监控功能，在 OB1 的局域数据表中，可以获得最大循环时间、最小循环时间和上一次的循环时间。

使用 STEP 7 可以修改默认的最大循环时间。如果实际的循环时间超出设置的最大时间，CPU 调用处理循环时间超时的组织块 OB80，在 OB80 中可以编写 CPU 响应这个故障的程序。若 OB80 未编写程序，CPU 将转入停止(STOP)模式。

如果 OB1 执行的启动时间间隔必须一致，或者希望循环时间不要太短，不要过于频繁地刷新过程映像区，S7-400 CPU 和 CPU 318 可以设置最小循环时间。

循环时间会因为下述事件而延长：中断处理、诊断和故障处理、测试和调试功能、通信、传送和删除块、压缩用户程序存储器、读/写微存储器卡 MMC 等。

可以用 CPU 参数 Scan Cycle Load from Communication(与通信有关的扫描循环负载)，控制处理通信过程的时间。

S7 400 CPU 和 CPU 318 可以直接访问 I/O 或者用系统功能 SFC 26(UPDAT _ PI)和 SFC 27(UPDAT _ PO)刷新一个或多个过程映像的输入或输出部分。

3. PLC 的工作原理

下面用一个简单的例子来进一步说明 PLC 的循环工作过程，图 1-5 中的 PLC 控制系统与图 1-3 中的继电器控制电路的功能相同。

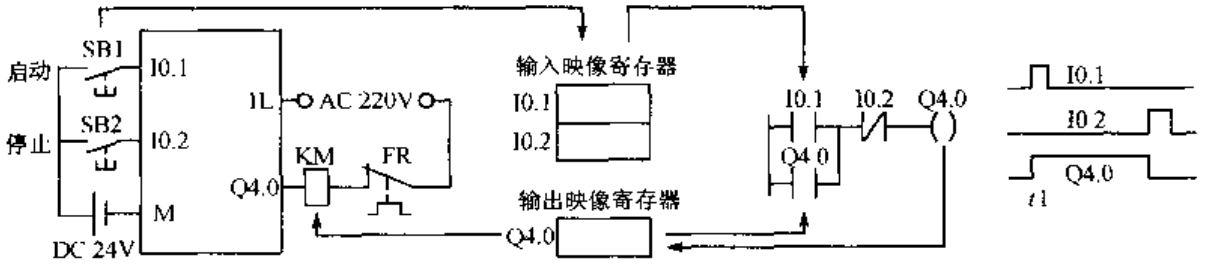


图 1-5　PLC 外部接线图与梯形图

假设 CPU 的型号为 CPU 313C，插在 4 号槽的 8 点输入模块的地址为 I0.0～I0.7，插在 5 号槽的 8 点输出模块的地址为 Q4.0～Q4.7。启动按钮 SB1 和停止按钮 SB2 的常开触点分别接在输入模块上 I0.1 和 I0.2 对应的输入端，接触器 KM 的线圈接在输出模块上 Q4.0 对应的输出端。如果热继电器 FR 动作(其常闭触点断开)后需手动复位，可以将 FR 的常闭触点与接触器 KM 的线圈串联，这样可以少用一个 PLC 的输入点。

图 1-5 梯形图中的 I0.1 与 I0.2 是输入变量，Q4.0 是输出变量。梯形图中的 I0.1 与输入过程映像位 I0.1 和接在对应的输入端子的 SB1 的常开触点相对应，梯形图中的 Q4.0 与输出过程映像位 Q4.0 和接在对应的输出端子的输出模块的输出电路相对应。

梯形图以指令的形式储存在 PLC 的用户程序存储器中，图 1-5 中的梯形图与下面的 6 条指令相对应("//"之后是该指令的注释)：

```
A(
O       I0.1          // 接在左侧母线上的 I0.1 的常开触点
O       Q4.0          // 与 I0.1 的常开触点并联的 Q4.0 的常开触点
)
```

```
AN      I0.2        // 与并联电路串联的 I0.2 的常闭触点
=       Q4.0        // Q4.0 的线圈
```

O(Or,或)指令表示常开触点并联,AN(And Not)表示常闭触点串联,"="表示将逻辑运算的结果传送给输出过程映像位 Q4.0。图 1-5 中的梯形图完成的逻辑运算为

$$Q4.0 = (I0.1 + Q4.0) \cdot \overline{I0.2}$$

在读取输入模块阶段,CPU 将 SB1,SB2 的常开触点的 ON/OFF 状态读入相应的输入过程映像位,外部触点接通时将二进制数 1 存入输入过程映像位,反之存入 0。

执行第 1 条 O 指令时,从输入过程映像位 I0.1 中取出二进制数。

执行第 2 条 O 指令时,从输出过程映像位 Q4.0 中取出二进制数,将两个二进制数相"或",因为触点的并联对应"或"运算。

执行 AN 指令时,取出输入过程映像位 I0.2 中的二进制数,因为是常闭触点,首先将它取"反",然后与前面"或"运算的结果相"与",因为电路的串联对应"与"运算。

执行第 6 条指令时,将前面的二进制数运算的结果送入 Q4.0 的输出过程映像位。

在数据写入输出模块阶段,CPU 将各输出过程映像位中的二进制数传送给输出模块,并由后者将数据锁存起来。如果输出过程映像位 Q4.0 中存放的是二进制数 1,外接的 KM 线圈将通电,反之将断电。

图 1-5 中 I0.1、I0.2 和 Q4.0 的波形图中的高电平表示按下按钮或 KM 线圈通电,当 $t < t1$ 时,读入输入过程映像位 I0.1 和 I0.2 的均为二进制数 0,此时输出过程映像位 Q4.0 中存放的亦为 0,在程序执行阶段,经过上述逻辑运算过程之后,运算结果仍为 Q4.0 = 0,所以 KM 的线圈处于断电状态。在 $t < t1$ 区间,虽然输入、输出信号的状态没有变化,用户程序仍一直反复不断地执行着。$t = t1$ 时按下启动按钮 SB1,I0.1 变为 1 状态,经逻辑运算后 Q4.0 也变为 1 状态,在输出处理阶段,将 Q4.0 对应的输出过程映像位中的 1 送到输出模块,输出模块中与 Q4.0 对应的物理继电器的常开触点接通,接触器 KM 的线圈通电。

4. 输入/输出滞后时间

输入/输出滞后时间又称为系统响应时间,是指 PLC 的外部输入信号发生变化的时刻至它控制的外部输出信号发生变化的时刻的时间间隔,它由输入电路滤波时间、输出电路的滞后时间和因扫描工作方式产生的滞后时间这三部分组成。

输入模块的 RC 滤波电路用来滤除由输入端引入的干扰噪声,消除因外接输入触点动作时产生的抖动引起的不良影响,滤波电路的时间常数决定了输入滤波时间的长短,其典型值为 10 ms 左右。

输出模块的滞后时间与模块的类型有关,继电器型输出电路的滞后时间一般在 10 ms 左右;双向晶闸管型输出电路在负载通电时的滞后时间约为 1 ms,负载由通电到断电时的最大滞后时间为 10 ms;晶体管型输出电路的滞后时间一般在 1 ms 以下。

由扫描工作方式引起的滞后时间最长可达两三个扫描周期。

PLC 总的响应延迟时间一般只有几毫秒到几十毫秒,对于一般的系统是无关紧要的。要求输入输出信号之间的滞后时间尽量短的系统,可以选用扫描速度快的 PLC 或采取中断等措施。

第 2 章　S7-300/400 的硬件组成

西门子公司的 PLC 产品有 SIMATIC S7、M7 和 C7 等几大系列。S7 系列是传统意义的 PLC 产品，其中的 S7-200 是针对低性能要求的小型 PLC。S7-300 是针对低性能要求的模块式中小型 PLC，最多可以扩展 32 个模块。S7-400 是用于中高级性能要求的大型 PLC，可以扩展 300 多个模块。S7-300/400 可以组成 MPI（多点接口）、PROFIBUS 网络和工业以太网等。

SIMATIC M7-300/400 PLC 采用与 S7-300/400 相同的结构，它可以作为 CPU 或功能模块使用。其显著特点是具有 AT 兼容计算机的功能，使用 S7-300/400 的编程软件 STEP 7 和可选的 M7 软件包，可以用 C、C++ 或 CFC（Continuous Function Chart，连续功能图）这类高级语言来对 M7-300/400 PLC 编程。M7 适合于需要处理的数据量大，对数据管理、显示和实时性有较高要求的系统使用。

SIMATIC C7 由 S7-300 PLC、HMI（人机接口）操作面板、I/O、通信和过程监控系统组成，整个控制系统结构紧凑，面向用户的配置/编程、数据管理与通信集成在一起，具有很高的性能价格比。由于高度集成，节约了 30% 的安装空间，可以和谐地集成到 SIMATIC 控制产品大家族中，保证正确的数据交换。

SIMATIC _ WinAC 基于 Windows NT/2000 操作系统和标准的接口（ActiveX、OPC），提供软件 PLC 或插槽 PLC。

WinAC 基本型用于常规 PLC 控制系统且有大量数据处理要求的场合。WinAC 实时型用于实时性、确定性要求非常高的控制场合，例如运动控制和快速控制等。WinAC 插槽型具有硬件 PLC 的所有特性，适用于实时性、安全性、可靠性要求均较高的场合。

WinAC 具有良好的开放性和灵活性，可以方便地集成第三方的软件和硬件，例如运动控制卡、快速 I/O 卡或控制算法等。它与其他 SIMATIC 系列控制器一样，用 STEP 7 编程和组态。

2.1　S7-200 系列 PLC 简介

2.1.1　S7-200 的基本结构

S7-200 系列属于整体式小型 PLC，用于代替继电器的简单控制场合，也可以用于复杂的自动化控制系统。

整体式 PLC 将 CPU 模块、I/O 模块和电源装在一个箱型机壳内，S7-200 称为 CPU 模块。图 2-1 中的前盖下面有 RUN/STOP 开关、模拟量电位器和扩展 I/O 连接器。S7-200 系列 PLC 提供多种具有不同 I/O 点数的 CPU 模块和数字量、模拟量 I/O 扩展模块供用户选用，CPU 模块和扩展模块用扁平电缆连接。

整体式 PLC 还配备有许多专用的特殊功能模块，例如模拟量输入/输出模块、热电偶、热电阻模块、通信模块等，使 PLC 的功能得到扩展。

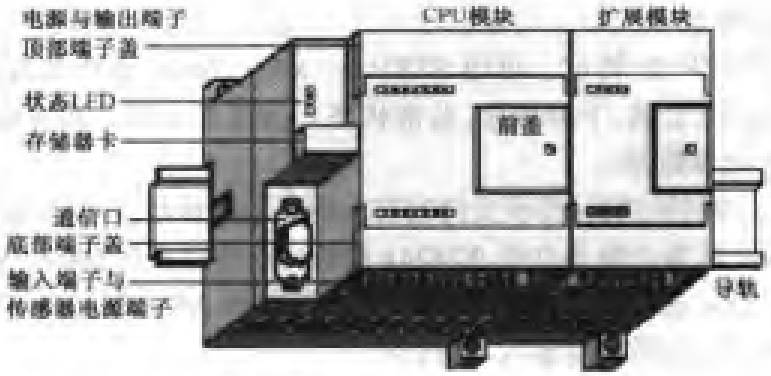


图 2-1 S7-200 PLC

S7-200 可以选用梯形图、语句表(即指令表)和功能块图语言来编程。它的指令丰富,指令功能强,易于掌握,操作方便,内置有高速计数器、高速输出、PID 控制器、RS-485 通信/编程接口、PPI 通信协议、MPI 通信协议和自由方式通信功能。最多可以扩展到 248 点数字量 I/O 或 35 路模拟量 I/O,最多有 26 KB 程序和数据存储空间。

2.1.2 S7-200 的 CPU 模块

S7-200 有 5 种 CPU 模块,CPU221 无扩展功能,适于作小点数的微型控制器。CPU222 有扩展功能,CPU224 是具有较强控制功能的控制器,CPU226 和 CPU226 XM 适用于复杂的中小型控制系统。

S7-200 有传送、比较、移位、循环移位、求补码、调用子程序、脉冲宽度调制、脉冲序列输出、跳转、数制转换、算术运算、字逻辑运算、浮点数运算、开平方、三角函数和 PID 控制指令等,采用主程序、最多 8 级子程序和中断程序的程序结构,用户可以使用 1～255 ms 的定时中断。用户程序可设 3 级口令保护,有监控定时器(看门狗)功能。

数字量输入中有 4 个用作硬件中断,6 个用于高速功能。32 位高速加/减计数器的最高计数频率为 30 kHz,可以对增量式编码器的两个互差 90°的脉冲列计数,计数值等于设定值或计数方向改变时产生中断,在中断程序中可以及时地对输出进行操作。两点高速输出可以输出频率最高为 20 kHz、频率和宽度可调的脉冲列。

可选的存储器卡可以永久保存程序、数据和组态信息,可选的电池卡保存数据的典型时间值为 200 天。DC 输出型电路用场效应晶体管(MOSFET)作为功率放大元件,仅 DC 输出型有高速脉冲输出,最高输出频率为 20 kHz。

2.1.3 S7-200 的通信能力

S7-200 的 CPU 模块自带的 RS-485 串行通信口支持 PPI、DP/T、自由通信口协议和 PROFIBUS 点对点协议。每个网络最多 126 个站,最多 32 个主站。通信接口可以实现与下列设备的通信:运行编程软件的计算机、文本显示器 TD 200、OP(操作员面板),以及 S7-200 CPU 之间的通信;通过自由通信口协议,可以与其他厂商的设备进行串行通信。

EM 277 PROFIBUS-DP 从站模块用于将 S7-200 CPU 连接到 PROFIBUS-DP 网络。通信速率为 9600～12 Mbit/s。

工业以太网通信模块 CP243-1 的通信速率为 10 Mbit/s 或 100 Mbit/s，半双工/全双工通信，RJ-45 接口，使用 TCP/IP 协议。可用 STEP 7 - Micro/WIN 软件实现通过工业以太网配置系统和远程编程服务（上载、下载程序，监视状态），通过工业以太网连接其他的 CPU，通过 S7-OPC 在计算机上处理数据。

EM241 Modem（调制解调器）模块支持远程维护或远程诊断、PLC 之间的通信、PLC 与 PC 的通信、给手机或寻呼机发送短消息等，EM241 参数化向导集成在 Micro/WIN V3.2 中。

通过 CP 243-2 AS-i 通信处理器，S7-200 CPU 可以作 AS-i 的主站，最多可以连接 62 个 AS-i 从站，接入 496 个远程数字量输入/输出点。

2.1.4 S7-200 的编程软件

STEP 7-Micro/WIN 32 是专门为 S7-200 设计的、在个人计算机 Windows 操作系统下运行的编程软件。CPU 通过 PC/PPI 电缆或插在计算机中的 CP 5511 或 CP 5611 通信卡与计算机通信。通过 PC/PPI 电缆，可以在 Windows 下实现多主站通信方式。

STEP 7-Micro/WIN 32 的用户程序结构简单清晰，通过一个主程序调用子程序或中断程序，还可以通过数据块进行变量的初始化设置。用户可以用语句表（STL）、梯形图（LAD）和功能块图（FBD）编程，不同的编程语言编制的程序可以相互转换，可以用符号表来定义程序中使用的变量地址对应的符号，使程序便于设计和理解。

STEP 7-Micro/WIN 32 为用户提供了两套指令集，即 SIMATIC 指令集（S7-200 方式）和国际标准指令集（IEC1131-3 方式）。通过调制解调器可以实现远程编程，可以用单次扫描和强制输出等方式来调试程序和进行故障诊断。

S7-200 是在美国德州仪器公司的小型 PLC 的基础上发展起来的，S7300/400 的前身是西门子公司的 S5 系列 PLC，其编程软件为 STEP 7。S7-200 和 S7-300/400 虽然有许多共同之处，但是在指令系统、程序结构和编程软件等方面均有相当大的差异。

2.2 S7-300 系列 PLC 简介

2.2.1 S7-300 的概况

大、中型 PLC（例如西门子的 S7-300 和 S7-400 系列）一般采用模块式结构，用搭积木的方式来组成系统，模块式 PLC 由机架和模块组成。S7-300 是模块化的中小型 PLC（见图 2-2），适用于中等性能的控制要求。品种繁多的 CPU 模块、信号模块和功能模块能满足各种领域的自动控制任务，用户可以根据系统的具体情况选择合适的模块，维修时更换模块也很方便。当系统规模扩大和更为复杂时，可以增加模块，对 PLC 进行扩展。简单实用的分布式结构和强大的通信联网能力，使其应用十分灵活。

S7-300 的 CPU 模块（简称为 CPU）集成了过程控制功能，用于执行用户程序。每个 CPU 都有一个编程用的 RS-485 接口，有的还带有集成的现场总线 PROFIBUS-DP 接口或 PtP（点对点）串行通信接口，S7-300 不需要附加任何硬件、软件和编程，就可以建立一个 MPI（多点接口）网络，如果有 PROFIBUS-DP 接口，可以建立一个 DP 网络。

S7-300 的通信功能、通信模块、通信的设置与编程的详细情况见第 7～9 章。

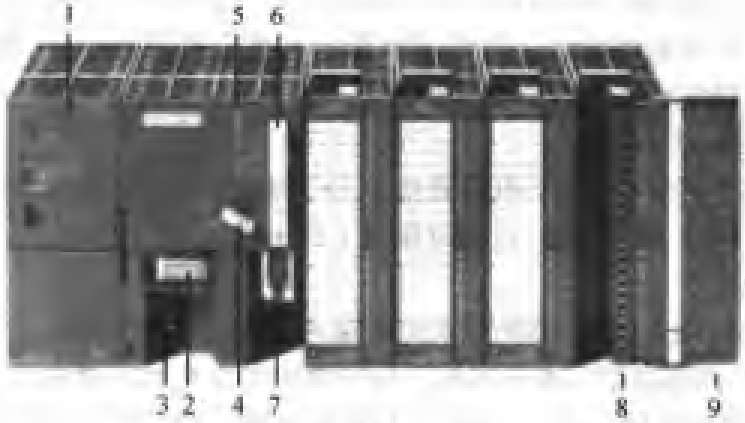


图 2-2 S7-300 PLC

1—电源模块 2—后备电池 3—DC 24V 连接器 4—模式开关 5—状态和故障指示灯
6—存储器卡(CPU 313 以上) 7—MPI 多点接口 8—前连接器 9—前盖

功能最强的 CPU 的 RAM 存储容量为 512KB,有 8192 个存储器位,512 个定时器和 512 个计数器,数字量通道最大为 65536 点,模拟量通道最大为 4096 个。由于使用 Flash EPROM,CPU 断电后无需后备电池也可以长时间保持动态数据,使 S7-300 成为完全无维护的控制设备。

S7-300/400 有很高的电磁兼容性和抗振动抗冲击能力。S7-300 标准型的环境温度为 0～60℃。环境条件扩展型的温度范围为 －25～＋60℃,有更强的耐振动和耐污染性能。

通过系统功能和系统功能块的调用,用户可以使用集成在操作系统内的程序,从而显著地减少所需要的用户存储器容量,它们可以用于中断处理、出错处理、复制和处理数据等。

S7-300/400 的编程软件 STEP 7 功能强大,使用方便。S7-300 有 350 多条指令。

STEP 7 的功能块图和梯形图编程语言符合 IEC 61131 标准,语句表编程语言与标准 IEC 稍有不同,以保证与 STEP 5 的兼容,3 种编程语言可以相互转换。用转换程序可以将西门子的 STEP 5 或 TISOFT 编写的程序转换到 STEP 7。STEP 7 还有 SCL、GRAPH 和 HiGrahp 等编程语言供用户选购。

计数器的计数范围为 1～999,定时器的定时范围为 10 ms～9990 s。可以使用 IEC 标准的定时器和计数器。

STEP 7 通过带标准用户接口的软件工具来为所有的模块设置参数,可以节省用户入门的时间和培训的费用。

CPU 用智能化的诊断系统连续监控系统的功能是否正常、记录错误和特殊系统事件(例如超时、模块更换等)。S7-300 有看门狗中断、过程报警、日期时间中断和定时中断功能。

操作员控制和监视显得日益重要,S7-300/400 已将 HMI(人机接口)服务集成到操作系统内,因此大大减少了人机对话的编程要求。SIMATIC 人机界面从 S7-300 中获得数据,S7-300/400按用户指定的刷新速度自动地传送这些数据。

2.2.2 S7-300 的组成部件

S7-300 PLC 是模块式的 PLC,它由以下几部分组成:

(1) 中央处理单元(CPU)

各种 CPU 有不同的性能,例如有的 CPU 集成有数字量和模拟量输入/输出点,有的 CPU 集成有 PROFIBUS-DP 等通信接口。CPU 前面板上有状态故障指示灯、模式开关、24 V 电源端子、电池盒与存储器模块盒(有的 CPU 没有)。

(2) 负载电源模块(PS)

负载电源模块用于将 AC 220 V 电源转换为 DC 24 V 电源,供 CPU 和 I/O 模块使用。额定输出电流有 2 A、5 A 和 10 A 3 种,过载时模块上的 LED 闪烁。

(3) 信号模块 (SM)

信号模块是数字量输入/输出模块和模拟量输入/输出模块的总称,它们使不同的过程信号电压或电流与 PLC 内部的信号电平匹配。信号模块主要有数字量输入模块 SM321 和数字量输出模块 SM322,模拟量输入模块 SM331 和模拟量输出模块 SM332。模拟量输入模块可以输入热电阻、热电偶、DC 4～20 mA 和 DC 0～10 V 等多种不同类型和不同量程的模拟信号。每个模块上有一个背板总线连接器,现场的过程信号连接到前连接器的端子上。

(4) 功能模块 (FM)

功能模块主要用于对实时性和存储容量要求高的控制任务,例如计数器模块、快速/慢速进给驱动位置控制模块、电子凸轮控制器模块、步进电动机定位模块、伺服电动机定位模块、定位和连续路径控制模块、闭环控制模块、工业标识系统的接口模块、称重模块、位置输入模块、超声波位置解码器等。

(5) 通信处理器 (CP)

通信处理器用于 PLC 之间、PLC 与计算机和其他智能设备之间的通信,可以将 PLC 接入 PROFIBUS-DP、AS-i 和工业以太网,或用于实现点对点通信等。通信处理器可以减轻 CPU 处理通信的负担,并减少用户对通信的编程工作。

(6) 接口模块(IM)

接口模块用于多机架配置时连接主机架(CR)和扩展机架(ER)。S7-300 通过分布式的主机架和 3 个扩展机架,最多可以配置 32 个信号模块、功能模块和通信处理器。

(7) 导轨

铝质导轨用来固定和安装 S7-300 上述的各种模块。

2.2.3 S7-300 的系统结构

S7-300 采用紧凑的、无槽位限制的模块结构,电源模块(PS)、CPU、信号模块(SM)、功能模块(FM)、接口模块(IM)和通信处理器(CP)都安装在导轨上。导轨是一种专用的金属机架,只需将模块钩在 DIN 标准的安装导轨上,然后用螺栓锁紧就可以了。有多种不同长度规格的导轨供用户选择。

电源模块总是安装在机架的最左边,CPU 模块紧靠电源模块。如果有接口模块,它放在 CPU 模块的右侧。

S7-300 用背板总线将除电源模块之外的各个模块连接起来。背板总线集成在模块上,模块通过 U 形总线连接器相连,每个模块都有一个总线连接器,后者插在各模块的背后(见图 2-3)。安装时先将总线连接器插在 CPU 模块上,并固定在导轨上,然后依次装入各个模块。

外部接线接在信号模块和功能模块的前连接器的端子上,前连接器用插接的方式安装在

模块前门后面的凹槽中，前连接器与模块是分开订货的。

S7-300 的电源模块通过电源连接器或导线与 CPU 模块相连，为 CPU 模块提供 DC 24 V 电源。PS307 电源模块还有一些端子可以为信号模块提供 24 V 电源。

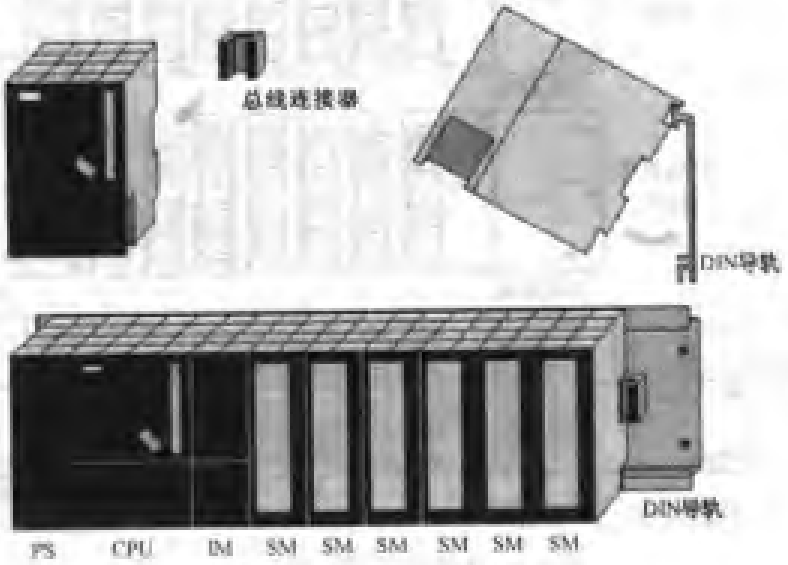


图 2-3　S7-300 的安装

更换模块时只需松开安装螺钉，拔下已经接线的前连接器。前连接器上的编码块用于防止将已接线的连接器插到其他模块上。

信号模块和通信处理器模块可以不受限制地插到任何一个槽上，系统可以自动分配模块的地址。每个机架最多只能安装 8 个信号模块、功能模块或通信处理器模块。如果系统任务需要的这些模块超过 8 块，则可以增加扩展机架，有的低端 CPU 没有扩展功能。

除了带 CPU 的中央机架(CR)，最多可以增加 3 个扩展机架(ER)，每个机架可以插 8 个模块(不包括电源模块、CPU 模块和接口模块 IM)，4 个机架最多可以安装 32 个模块。

机架的最左边是 1 号槽，最右边是 11 号槽，电源模块总是在 1 号槽的位置。中央机架(0 号机架)的 2 号槽上是 CPU 模块，3 号槽是接口模块。这 3 个槽号被固定占用，信号模块、功能模块和通信处理器使用 4～11 号槽。

因为模块是用总线连接器连接的，而不是像其他模块式 PLC 那样，用焊在背板上的总线插座来安装模块，所以槽号是相对的，在机架导轨上并不存在物理槽位。例如在不需要扩展机架时，中央机架上没有接口模块，此时虽然 3 号槽位仍然被实际上并不存在的接口模块占用，中央机架上的 CPU 模块和 4 号槽的模块实际上是挨在一起的。

如果有扩展机架，接口模块占用 3 号槽位，负责与其他扩展机架自动地进行数据通信。

如果只需要扩展一个机架，可以使用价格便宜的 IM 365 接口模块对，两个接口模块用 1m 长的固定电缆连接，由于 IM 365 不能给机架 1 提供通信总线，机架 1 上只能安装信号模块，不能安装通信模块和其他智能模块。扩展机架的电源由 IM 365 提供，两个机架的 DC 5 V 电源的总电流应在允许值之内。

使用 IM 360/361 接口模块可以扩展 3 个机架，中央机架(CR)使用 IM 360，扩展机架(ER)使用 IM 361，各相邻机架之间的电缆最长为 10 m。每个 IM 361 需要一个外部 DC 24 V 电源，向扩展机架上的所有模块供电，可以通过电源连接器连接 PS 307 负载电源。所有的

S7-300模块均可以安装在 ER 上。接口模块是自组态的，无需进行地址分配。

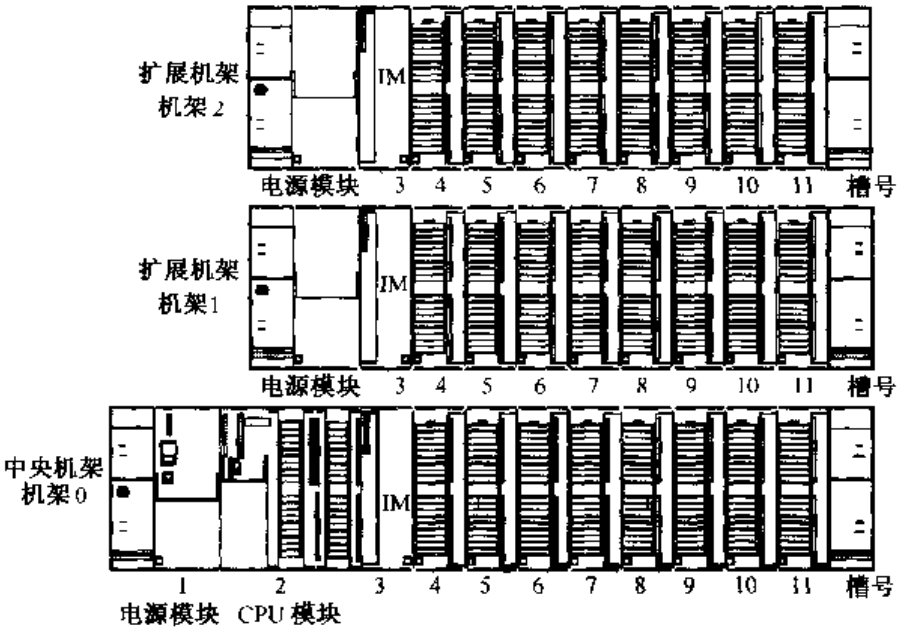


图 2-4　多机架的 S7-300 PLC

每个机架上安装的信号模块、功能模块和通信处理器除了不能超过 8 块外，还受到背板总线 DC 5 V 供电电流的限制。0 号机架的 DC 5 V 电源由 CPU 模块产生，其额定电流值与 CPU 的型号有关。扩展机架的背板总线的 DC 5 V 电源由接口模块 IM361 产生，各类模块消耗的电流可查 S7-300 模板手册。

2.2.4　I/O 模块地址的确定

S7-300 的开关量地址由地址标识符、地址的字节部分和位部分组成，一个字节由 0～7 这 8 位组成。地址标识符 I 表示输入，Q 表示输出，M 表示存储器位。例如 I3.2 是一个数字输入量的地址，小数点前面的 3 是地址的字节部分，小数点后的 2 表示这个输入点是 3 号字节中的第 2 位。

开关量除了按位寻址外，还可以按字节、字和双字寻址。例如输入量 I2.0～I2.7 组成输入字节 IB2，B 是 Byte 的缩写；字节 IB2 和 IB3 组成一个输入字 IW2，W 是 Word 的缩写，其中的 IB2 为高位字节；IB2～IB5 组成一个输入双字 ID2，D 是 Double Word 的缩写，其中的 IB2 为最高位的字节。以组成字和双字的第一个字节的地址作为字和双字的地址。

S7-300 的信号模块的字节地址与模块所在的机架号和槽号有关，位地址与信号线接在模块上的哪一个端子有关。

对于数字量模块，从 0 号机架的 4 号槽开始，每个槽位分配 4B(4 个字节)的地址，相当于 32 个 I/O 点(见表 2-1)。最多可能有 32 个数字量模块，共占用 32×4B = 128B。

模拟量模块以通道为单位，一个通道占一个字地址，或两个字节地址。例如模拟量输入通道 IW640 由字节 IB640 和 IB641 组成。S7-300 为模拟量模块保留了专用的地址区域，字节地址范围为 IB256～767。可以用装载指令和传送指令访问模拟量模块。

一个模拟量模块最多有 8 个通道，从 256 开始，给每一个模拟量模块分配 16B(8 个字)的地址。

I/O 模块的字节地址见表 2-1，信号模块的地址举例见表 2-2。

表 2-1 I/O 模块的字节地址

机架号	模块类型	槽号							
		4	5	6	7	8	9	10	11
0	数字量	0～3	4～7	8～11	12～15	16～19	20～23	24～27	28～31
	模拟量	256～271	272～287	288～303	304～319	320～335	336～351	352～367	368～383
1	数字量	32～35	36～39	40～43	44～47	48～51	52～55	56～59	60～63
	模拟量	384～399	400～415	416～431	432～447	448～463	464～479	480～495	496～511
2	数字量	64～67	68～71	72～75	76～79	80～83	84～87	88～91	92～95
	模拟量	512～527	528～543	544～559	560～575	576～591	592～607	608～623	624～639
3	数字量	96～99	100～103	104～107	108～111	112～115	116～119	120～123	124～127
	模拟量	640～655	656～671	672～687	688～703	704～719	720～735	736～751	752～767

表 2-2 信号模块地址举例

机架号	模块类型	槽号					
		4	5	6	7	8	9
0	模块类型	16 点数字量输入	16 点数字量输入	32 点数字量输入	32 点数字量输入	16 点数字量输出	8 通道模拟量输入
	地址	I0.0～I1.7	I4.0～I5.7	I8.0～I11.7	I12.0～I15.7	Q16.0～Q17.7	IW336～IW350
1	模块类型	2 通道模拟量输入	8 通道模拟量输出	2 通道模拟量输出	8 点数字量输出	32 点数字量输出	—
	地址	IW384，IW386	QW400～QW414	QW416，QW418	Q44.0～Q44.7	Q48.0～Q51.7	—

数字量输入/输出模块内最低的位地址（例如 I0.0）对应的端子位置最高，最高的位地址（例如 16 点输入模块的 I1.7）对应的端子的位置最低。

2.2.5 模块诊断与过程中断

1. 模块诊断功能

S7-300 有的信号模块具有对信号进行监视（诊断）和过程中断的智能功能。通过诊断可以确定数字量模块获取的信号是否正确，或模拟量模块的处理是否正确。

数字量输入/输出模块可以诊断出以下故障：无编码器电源、无外部辅助电压、无内部辅助电压、熔断器熔断、看门狗故障、EPROM 故障、RAM 故障、过程报警丢失。

模拟量输入模块可以诊断出无外部电压、共模故障、组态/参数错误、断线、测量范围上溢出或下溢出。模拟量输出模块可以诊断出无外部电压、组态/参数错误、断线和对地短路。

2. 过程中断

通过过程中断，可以对过程信号进行监视和响应。

根据设置的参数，可以选择数字量输入模块每个通道组是否在信号的上升沿、下降沿，或两个边沿都产生中断。信号模块可以对每个通道的一个中断进行暂存。

模拟量输入模块通过上限值和下限值定义一个工作范围，模块将测量值与上、下限值进行比较。如果超限，则执行过程中断。

执行过程中断时，CPU 暂停执行用户程序，或暂停执行低优先级的中断程序，来处理相应的诊断中断功能块（OB40）。

2.3 S7-300 的 CPU 模块

2.3.1 CPU 模块的元件

S7-300 有 20 种不同型号的 CPU，分别适用于不同等级的控制要求。有的 CPU 模块集成了数字量 I/O，有的同时集成了数字量 I/O 和模拟量 I/O。

CPU 内的元件封装在一个牢固而紧凑的塑料机壳内，面板上有状态和故障指示 LED、模式选择开关和通信接口。大多数 CPU 还有后备电池盒，存储器插槽可以插入多达数兆字节的 Flash EPROM 微存储器卡(简称为 MMC)，用于掉电后程序和数据的保存。CPU 318-2 的面板如图 2-5 所示。

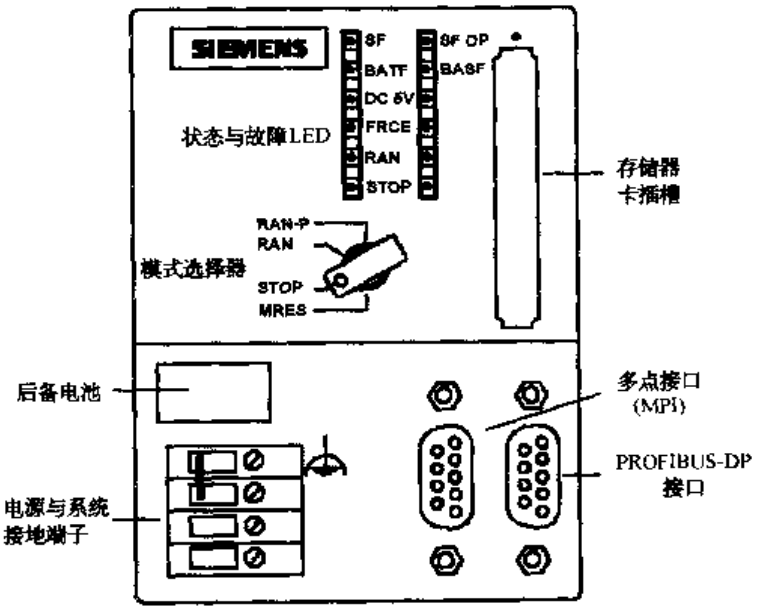


图 2-5 CPU 318-2 的面板

1. 状态与故障显示 LED

CPU 模块面板上的 LED(发光二极管)的意义如下：

(1) SF(系统出错/故障显示，红色)：CPU 硬件故障或软件错误时亮。

(2) BATF(电池故障，红色)：电池电压低或没有电池时亮。

(3) DC 5V(+5 V 电源指示，绿色)：CPU 和 S7 300 总线的 5 V 电源正常时亮。

(4) FRCE(强制，黄色)：至少有一个 I/O 被强制时亮。

(5) RUN(运行方式，绿色)：CPU 处于 RUN 状态时亮；重新启动时以 2 Hz 的频率闪亮；HOLD 状态时以 0.5 Hz 的频率闪亮。

(6) STOP(停止方式，黄色)：CPU 处于 STOP、HOLD 状态或重新启动时常亮；请求存储器复位时以 0.5 Hz 的频率闪亮，正在执行存储器复位时以 2 Hz 的频率闪亮。

(7) BUSF(总线错误，红色)：PROFIBUS-DP 接口硬件或软件故障时亮，集成有 DP 接口的 CPU 才有此 LED。集成有两个 DP 接口的 CPU 有两个对应的 LED(BUS1F 和 BUS2F)。

2. CPU 的运行模式

CPU 有 4 种操作模式:STOP(停机)、STARTUP(启动)、RUN(运行)和 HOLD(保持)。在所有的模式中,都可以通过 MPI 接口与其他设备通信。

(1) STOP 模式:CPU 模块通电后自动进入 STOP 模式,在该模式不执行用户程序,可以接收全局数据和检查系统。

(2) RUN 模式:执行用户程序,刷新输入和输出,处理中断和故障信息服务。

(3) HOLD 模式:在起动和 RUN 模式执行程序时遇到调试用的断点,用户程序的执行被挂起(暂停),定时器被冻结。

(4) STARTUP 模式:启动模式,可以用钥匙开关或编程软件启动 CPU。如果钥匙开关在 RUN 或 RUN-P 位置,通电时自动进入启动模式。

3. 模式选择开关

有的 CPU 的模式选择开关(模式选择器)是一种钥匙开关,操作时需要插入钥匙,用来设置 CPU 当前的运行方式。钥匙拔出后,就不能改变操作方式。这样可以防止未经授权的人员非法删除或改写用户程序。还可以使用多级口令来保护整个数据库,使用户有效地保护其技术机密,防止未经允许的复制和修改。钥匙开关各位置的意义如下:

(1) RUN-P(运行-编程)位置:CPU 不仅执行用户程序,在运行时还可以通过编程软件读出和修改用户程序,以及改变运行方式。在这个位置不能拔出钥匙开关。

(2) RUN (运行)位置:CPU 执行用户程序,可以通过编程软件读出用户程序,但是不能修改用户程序,在这个位置可以取出钥匙开关。

(3) STOP(停止)位置:不执行用户程序,通过编程软件可以读出和修改用户程序,在这个位置可以取出钥匙开关。

(4) MRES(清除存储器):MRES 位置不能保持,在这个位置松手时开关将自动返回 STOP 位置。将钥匙开关从 STOP 状态扳到 MRES 位置,可复位存储器,使 CPU 回到初始状态。工作存储器、RAM 装载存储器中的用户程序和地址区被清除,全部存储器位、定时器、计数器和数据块均被删除,即复位为零,包括有保持功能的数据。CPU 检测硬件,初始化硬件和系统程序的参数,系统参数、CPU 和模块的参数被恢复为默认设置,MPI(多点接口)的参数被保留。如果有快闪存储器卡,CPU 在复位后将它里面的用户程序和系统参数复制到工作存储区。

复位存储器按下述顺序操作:PLC 通电后将钥匙开关从 STOP 位置扳到 MRES 位置,STOP LED 熄灭 1 s,亮 1 s,再熄灭 1 s 后保持亮。放开开关,使它回到 STOP 位置,然后又回到 MRES,STOP LED 以 2 Hz 的频率至少闪动 3 s,表示正在执行复位,最后 STOP LED 一直亮,可以松开模式开关。

存储器卡被取掉或插入时,CPU 发出系统复位请求,STOP LED 以 0.5Hz 的频率闪动。此时应将模式选择开关扳到 MRES 位置,执行复位操作。

4. 微存储器卡

Flash EPROM 微存储卡(MMC)用于在断电时保存用户程序和某些数据,它可以扩展 CPU 的存储器容量,也可以将有些 CPU 的操作系统保存在 MMC 中,这对于操作系统的升级是非常方便的。MMC 用作装载存储器或便携式保存媒体。MMC 的读写直接在 CPU 内进行,不需要专用的编程器。由于 CPU 31xC 没有安装集成的装载存储器,在使用 CPU 时必须插入 MMC,CPU 与 MMC 是分开订货的。

如果在写访问过程中拆下 SIMATIC 微存储卡，卡中的数据会被破坏。在这种情况下，必须将 MMC 插入 CPU 中并删除它，或在 CPU 中格式化存储卡。只有在断电状态或 CPU 处于 STOP 状态时，才能取下存储卡。

5. 通信接口

所有的 CPU 模块都有一个多点接口 MPI，有的 CPU 模块有一个 MPI 和一个 PROFIBUS-DP 接口，有的 CPU 模块有一个 MPI/DP 接口和一个 DP 接口。

MPI 用于 PLC 与其他西门子 PLC、PG/PC(编程器或个人计算机)、OP(操作员接口)通过 MPI 网络的通信。CPU 通过 MPI 接口或 PROFIBUS-DP 接口在网络上自动地广播它设置的总线参数(即波特率)，PLC 可以自动地"挂到"MPI 网络上。

PROFIBUS-DP 的传输速率最高 12 Mbit/s，用于与其他西门子带 DP 接口的 PLC、PG/PC、OP 和其他 DP 主站和从站的通信。

6. 电池盒

电池盒是安装锂电池的盒子，在 PLC 断电时，锂电池用来保证实时钟的正常运行，并可以在 RAM 中保存用户程序和更多的数据，保存的时间为 1 年。有的低端 CPU(例如 312IFM 与 313)因为没有实时钟，没有配备锂电池。

7. 电源接线端子

电源模块的 L1、N 端子接 AC 220 V 电源，电源模块的接地端子和 M 端子一般用短路片短接后接地，机架的导轨也应接地。

电源模块上的 L+ 和 M 端子分别是 DC 24 V 输出电压的正极和负极。用专用的电源连接器或导线连接电源模块和 CPU 模块的 L+ 和 M 端子。

8. 实时钟与运行时间计数器

CPU 312 IFM 与 CPU 313 因为没有锂电池，只有软件实时钟，PLC 断电时停止计时，恢复供电后从断电瞬时的时刻开始计时。有后备锂电池的 CPU 有硬件实时钟，可以在 PLC 电源断电时继续运行，运行小时计数器的计数范围为 0～32 767h。

9. CPU 模块上的集成 I/O

某些 CPU 模块上有集成 I/O，图 2-6 中标有①的是模拟量输入端子和模拟量输出端子，标有②的端子是 8 点数字量(即开关量)输入端子，标有③的是 8 点开关量输出端子。

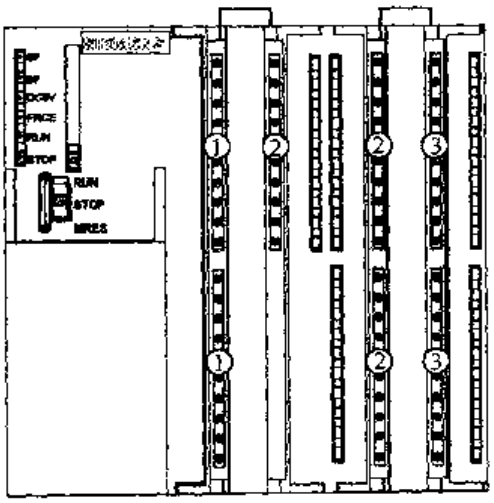


图 2-6　CPU 314C-2 PtP 的集成 I/O

2.3.2　CPU 模块的技术规范

1. 存储器

存储器分为系统程序存储器和用户程序存储器。系统程序相当于个人计算机的操作系统，它使 PLC 具有基本的智能，能够完成 PLC 设计者规定的各种工作。系统程序由 PLC 生产厂家设计并固化在 ROM(只读存储器)中，用户不能读取。用户程序由用户设计，它使 PLC 能完成用户要求的特定功能。用户程序存储器的容量以字(16 位二进制数)为单位。

PLC 使用以下几种物理存储器：

(1) 随机存取存储器(RAM)

用户可以用编程装置读出RAM中的内容,也可以将用户程序写入RAM,因此RAM又叫读/写存储器。它是易失性的存储器,电源中断后,储存的信息将会丢失。

RAM的工作速度高,价格便宜,改写方便。在中断PLC的外部电源后,可用锂电池保存RAM中的用户程序和某些数据。需要更换锂电池时,由PLC发出信号,通知用户。现在部分PLC仍用RAM来储存用户程序。

(2) 只读存储器(ROM)

ROM的内容只能读出,不能写入。它是非易失的,电源消失后,仍能保存储存的内容,ROM一般用来存放PLC的系统程序。

(3) 快闪存储器和EEPROM

快闪存储器(Flash EPROM)的简称为FEPROM,可电擦除可编程的只读存储器的简称为EEPROM或E^2PROM。它们是非易失性的,可以用编程装置对它编程,兼有ROM的非易失性和RAM的随机存取优点,但是将信息写入它所需的时间比RAM长得多。它们用来存放用户程序和需要长期保存的重要数据。

2. S7-300 CPU的分类

S7-300的CPU模块大致可以分为以下几类:

(1) 6种紧凑型CPU,带有集成的功能和I/O:CPU 312C,313C,313C-PtP,313C-2DP,314C-PtP和314C-2DP。

(2) 3种重新定义的CPU:CPU 312,314和315-2DP。

(3) 5种标准的CPU:CPU 313,314,315,315-2DP和316-2DP。

(4) 4种户外型CPU:CPU 312 IFM,314 IFM,314户外型和315-2DP。

(5) 高端CPU:317-2DP和CPU 318-2DP。

(6) 故障安全型CPU:CPU 315F。

3. 紧凑型CPU

S7-31xC有6种紧凑型CPU,有的有集成的DI/DO(数字量输入/输出)和AI/AO(模拟量输入/输出,见表2-4)。

CPU 314C-2 DP和CPU 314C-2 PtP有定位功能,带有模拟量输出和数字量输出。

各CPU均有计数、频率测量和脉冲宽度调制功能,脉宽调制频率最高为2.5 kHz。

CPU 313C-2 PtP和CPU 314C-2 PtP集成有点对点通信接口,ASCII协议的通信速率为19.2 kbit/s(全双工),38.4 kbit/s(半双工),3964R协议为38.4 kbit/s,RK512协议为38.4 kbit/s。

表2-3 S7-31xC的集成功能

型 号	定位通道数	计数通道数	最高可测频率	点对点通信协议	闭环控制功能
CPU 312C	—	2	10 kHz	—	—
CPU 313C	—	3	30 kHz	—	有
CPU 313C-2 DP	—	3	30 kHz	—	有
CPU 313C-2 PtP	—	3	30 kHz	ASCII,3964R	有
CPU 314C-2 DP	1	4	60 kHz	—	有
CPU 314C-2 PtP	1	4	60 kHz	ASCII,3964R,RK512	有

S7-31xC 的 RAM 不能扩展，没有集成的装载存储器，运行时需要插入 MMC 卡，通过 MMC 卡执行程序和保存数据，MMC 卡为免维护的 Flash EPROM(FEPROM)，可以扩展至 4MB。各 CPU 均有实时钟，CPU 312C 的时钟没有电池后备功能。CPU 有一个运行小时计数器，有日期时间同步功能。FB(功能块)、FC(功能)和 DB(数据块)的最大容量为 16 KB。

CPU 312C 有集成的数字量 I/O，适用于有较高要求的小型系统。

CPU 313C 有集成的数字量 I/O 和模拟量 I/O，适用于有较高要求的系统。

CPU 313C-2PtP，CPU 314C-2PtP 有集成的数字 I/O 和第二个串口，两个接口均有点对点(PtP)通信功能。CPU 314C-2PtP 还有集成的模拟量 I/O，适用于有较高要求的系统。

CPU 313C-2DP 和 CPU 314C-2DP 有集成的数字 I/O 和两个 PROFIBUS-DP 主站、从站接口，通过 CP(通信处理器)各 CPU 可以扩展一个 DP 主站。CPU 314C-2DP 还有集成的模拟量 I/O，适用于有较高要求的系统。

4 路集成的模拟量输入信号的量程为 ±10 V、0～10 V、±20 mA、4～20 mA，积分时间可调为 2.5 ms、16.6 ms、20 ms，单极性输入为 11 位 + 符号位，25℃时的基本误差为 0.7%。

1 路集成的模拟量输入通道可测 0～600 Ω 电阻，或接 Pt 100 热电阻。

两路集成的模拟量输出的输出范围为 10 V，0～10 V，±20 mA，4～20 mA 和 0～20 mA。各通道的转换时间为 1 ms，25℃时的基本误差为 0.7%。

CPU 模块的第一个通信接口有 PG/OP 通信和全局数据(GD)通信功能，发送方和接收方的 GD 环最大个数均为 4 个，GD 包最大 22 B。有 S7 标准通信功能，每个作业的最大用户数据为 76 B。S7 通信中可以作为服务器，每个作业的最大用户数据为 64 B。

MPI 的最大电缆长度为 50 m，最高传输频率为 187.5 k bit/s。

紧凑型 CPU 技术参数见表 2-4。

表 2-4　紧凑型 CPU 技术参数

CPU	312C	313C	313C-2PtP	313C-2DP	314C-2PtP	314C-2DP
集成式 RAM	16 KB	32 KB	32 KB	32 KB	48 KB	48 KB
装载存储器 MMC 卡	最大 4 MB	最大 4 MB	最大 4 MB	最大 4 MB	最大 4 MB	最大 4 MB
最小位操作时间 最小浮点数加法时间	0.2～0.4 μs 30 μs	0.1～0.2 μs 15 μs	0.1～0.2 μs 15 μs	0.1～0.2 μs 15 μs	0.1～0.2 μs 15 μs	0.1～0.2 μs 15 μs
集成 DI/DO 集成 AI/AO	10/6	24/16 4 + 1/2	16/16	16/16	24/16 4 + 1/2	24/16 4 + 1/2
FB 最大块数 FC 最大块数 DB 最大块数	64 64 63(DB0 保留)	128 128 127(DB0 保留)	128 128 127(DB0 保留)	128 128 127(DB0 保留)	128 128 127(DB0 保留)	128 128 127(DB0 保留)
位存储器	1024 B	2048 B	2048 B	2048 B	2048 B	2048 B
定时器/计数器	128/128	256/256	256/256	256/256	256/256	256/256
全部 I/O 地址区 I/O 过程映像 最大数字量 I/O 总数	1024 B/1024 B 128 B/128 B 256/256	1024 B/1024 B 128 B/128 B 992/992	1024 B/1024 B 128 B/128 B 992/992	1024 B/1024 B 128 B/128 B 992/992	1024 B/1024 B 128 B/128 B 992/992	1024 B/1024 B 128 B/128 B 992/992
最大模拟量 I/O 总数	64/32	248/124	248/124	248/124	248/124	248/124
模块总数	8	31	31	31	31	31

（续）

CPU-	312C	313C	313C-2PtP	313C-2DP	314C-2PtP	314C-2DP
通信的连接总数 报文功能可定义的站数	6 3	8 5	8 5	8 5	12 7	12 7
最大机架数/模块总数 通信接口与功能	1/8 MPI 接口	4/31 MPI 接口	4/31 2 个 PtP 接口	4/31 2 个 DP 接口	4/31 2 个 PtP 接口	4/31 2 个 DP 接口

4. 标准型 CPU

CPU 312 适用于对处理速度有中等要求的小规模应用。

CPU 314 适用于对程序量有中等要求的应用，对二进制和浮点数有较高的处理性能。

CPU 315-2DP 具有大中规模的程序容量，对二进制和浮点数有较高的处理性能，有两个 PROFIBUS-DP 主站/从站接口，可以用于建立分布式 I/O 结构和大规模的 I/O 配置。

各 CPU 的 RAM 不能扩展，CPU 312，314 和 315-2DP 运行时需要 MMC 卡。CPU 312 有软件实时钟，其余的均有硬件实时钟。FB，FC 和 DB 的最大容量为 16KB。有 8 个时钟位存储器，有一个运行小时计数器，有实时钟同步功能。

CPU 最多监视 30 个变量，强制 14 个 I/O 变量，包括位存储器、DB、定时器和计数器。有状态块和单步功能，可以设置两个断点。

CPU 模块的第一个通信接口是内置的 RS-485 接口，没有隔离，有 MPI 的 PG/OP 通信和全局数据(GD)通信功能，发送方和接收方的 GD 环最大个数均为 4 个，GD 包最大 22 B。有 S7 标准通信功能，每个作业的最大用户数据为 76 B。S7 通信中每个作业的最大用户数据为 180 B。CPU 315-2DP 可以作路由器，有点对点通信功能。

5. 户外型 CPU

户外型 CPU 可以在恶劣的环境下使用。CPU312IFM 和 CPU314IFM 是户外紧凑型 CPU，带有集成的数字量 I/O，CPU 312 IFM 适用于小系统，具有特殊功能，无实时钟。CPU 314 IFM 适用于对响应时间和特殊功能有较高要求的系统。

CPU 314 户外型 CPU 的处理速度高，具有中等规模的 I/O 配置，适用于要求中等规模的程序量和中等的指令执行速度的系统。

CPU 312 IFM 和 CPU 314 IFM 有一个计数器，最高计数频率为 10 kHz，1 个通道可以测量频率，最高 10 kHz；CPU 314 IFM 有 1 个通道可用增量式编码器进行位置检测。

CPU 314 IFM 有 4 路集成的模拟量输入，信号量程为 ±10 V 和 ±20mA，11 位 + 符号位。1 路集成的模拟量输出的输出范围为 ±10 V 为 ±20 mA，11 位 + 符号位。

MPI 接口可以连接 32 个站，可以与 PG/PC，OP，其他 S7-300 /400 和 C7 通信，最多有 2 个动态连接和 4 个静态连接。最大传输速率为 187.5 k bit/s，10 个中继器串联时最大距离为 9100 m，通过光纤通信可达 23 800 m。

户外型 CPU 的全局数据(GD)通信功能与标准型的相同。CPU 315-2DP 可以作路由器，有点对点通信功能。

标准型与户外型 CPU 的技术参数见表 2-5。

表 2-5 标准型与户外型 CPU 的技术参数

CPU-	312	314	315-2DP	312IFM 户外型	314IFM 户外型	314 户外型
集成式 RAM	16 KB	48 KB	128 KB	6 KB	32 KB	24 KB
集成装载存储器 RAM/Flash 插入式存储卡 FEPROM	— 最大 4 MB	— 最大 8 MB	— 最大 8 MB	20 KB/20 KB —	48 KB/— —	40 KB/— 最大 4 MB
最大位操作指令执行时间 浮点数指令执行时间	0.2 μs 6 μs	0.1 μs 6 μs	0.1 μs 6 μs	1.2 μs 60 μs	0.6 μs 50 μs	0.6 μs 50 μs
集成 DI/DO 集成 AI/AO	10/6	24/16 4 + 1/2	16/16	10/6,4 个可中断	20/16,4 个可中断 4/1	
FB 最大块数/大小 FC 最大块数/大小 DB 最大块数/大小 OB 最大容量 块嵌套深度/错误 OB 中增加	512/16 KB 512/16 KB 511/16 KB 16 KB 8/2	512/16 KB 512/16 KB 511/16 KB 16 KB 8/2	2048/16 KB 2048/16 KB 1023/16 KB 16 KB 8/2	32/6 KB 32/6 KB 63/6 KB	128/8 KB 128/8 KB 127/8 KB	128/8 KB 128/8 KB 127/8 KB
位存储器	128 B	256 B	2048 B	1024 B	2048 B	2048 B
定时器/计数器	128/128	256/256	256/256	64/32	128/64	128/64
每个优先级的最大局部数据	256 B	512 B	1024 B			
全部 I/O 地址区 最大分布式 I/O 地址区 过程 I/O 映像 最大数字量 I/O 总数 中央区最大数字量 I/O	1024 B/1024 B — 128 B/128 B 256 256	1024 B/1024 B — 128 B/128 B 1024 1024	1024 B/1024 B 2000 B 128 B/128 B 16384 1024	128 B/128 B 32 B/32 B 256	512 B/512 B 128 B/128 B 992	512 B/512 B 128 B/128 B 1024
最大模拟量 I/O 总数 中央区最大模拟量量 I/O	64 64	256 256	1024 256	64/32	248/124	256/128
本机数字量 I/O 本机模拟量 I/O	— —	— —	— —	10/6 —	20/16 4/1	— —
最大机架数/模块总数 MPI 通信接口 内置/通过 CP 的 DP 接口数 通过 CP 连接的 DP 站	1/8 1 个 0/1 —	4/32 1 个 0/1 —	4/32 2 个 1/1 —	1/8 1 个 0/1 8	4/32 1 个 0/1 16	4/32 1 个 0/1 32
报文功能可定义的站数	6	12	16	—	—	—
S7 通信作服务器/客户机	可以/不可以	可以/可以	可以/可以	可以/不可以	可以/不可以	可以/不可以
最大连接数量	6	12	16	8	12	12

6. 其他 CPU

CPU 317-2DP 和 318-2DP 具有大容量程序存储器和 PROFIBUD-2DP 主站/从站接口，用于大规模 I/O 配置和建立分布式 I/O 结构。

CPU 315F 带有 PROFIBUD-2DP 主站/从站接口，可以组态为故障安全型自动化系统，满足安全运行的要求，不需要对故障 I/O 进行额外的布线，使用 PROFIsave 协议的 PROFIBUD-DP 实现与安全有关的通信。ET 200M 和 ET 200S 可以使用故障安全的数字量模块，也可以在自动化系统中使用与安全无关的标准模块。

其他 CPU 的技术参数见表 2 6。

表 2-6 其他 CPU 的技术参数

CPU-	317-2DP	318-2DP	315F-2DP
工作存储器 RAM	512 KB	512 KB	128 KB
集成装载存储器 RAM/ROM 插入存储卡 FEPROM/RAM	— 最大 8 MB	64 KB 最大 4 MB	64 KB 最大 4 MB
最小位操作指令执行时间 浮点数运算指令执行时间	0.1 μs 1.5 μs	0.1 μs 0.6 μs	0.1 μs
FB 最大块数/大小 FC 最大块数/大小 DB 最大块数/大小	2048/64 KB 2048/64 KB 2047/64KB	1024/64 KB 1024/64 KB 2047/64 KB	
块嵌套深度/在错误 OB 中增加	16/4	20	
每个优先级的局域数据	1024 B		
位存储器	4096 B	8192 B	2048 B
定时器/计数器个数	512/512	512/512	256/256
最大 I/O 地址区 最大分布式 I/O 地址区 I/O 过程映像 最大数字量 I/O 总数 最大本地数字量 I/O	8192 B/8192 B 8192 B 256 B/256 B 65536/65536 1024	8192 B/8192 B 256 B/256 B 可扩展到 2048 B 65536/65536 1024	1024 B 1000/1000 —
最大模拟量 I/O 总数 本地最大模拟量 I/O	4096/4096 256/256	4096/4096 256/128	248/124 —
最大机架数/模块总数	4/32	4/32	4/32
通信接口 内置/通过 CP 的 DP 总站数 最大 GD 环数/GD 包数/GD 包字节数	两个 DP 接口 2/2 8/8/22 B	两个 DP 接口 2/2 8/8/22 B	

2.4 S7-300 的输入/输出模块

输入/输出模块统称为信号模块(SM),包括数字量(或称开关量)输入模块、数字量输出模块、数字量输入/输出模块、模拟量输入模块、模拟量输出模块和模拟量输入/输出模块。

S7-300 的输入/输出模块的外部接线接在插入式的前连接器的端子上,前连接器插在前盖后面的凹槽内。不需断开前连接器上的外部连线,就可以迅速地更换模块。第一次插入连接器时,有一个编码元件与之啮合,这样该连接器就只能插入同样类型的模块中。

信号模块面板上的 LED 用来显示各数字量输入/输出点的信号状态,模块安装在 DIN 标准导轨上,通过总线连接器与相邻的模块连接。模块的默认地址由模块所在的位置决定,也可以用 STEP 7 指定模块的地址。

信号模块和接口模块的尺寸为 40 mm(宽)×125 mm(高)×120 mm(深)。有少量模块的宽度为80 mm。

2.4.1 数字量输入模块

数字量输入模块用于连接外部的机械触点和电子数字式传感器,例如二线式光电开关和接近开关等。数字量输入模块将从现场传来的外部数字信号的电平转换为 PLC 内部的信号电平。

输入电路中一般设有 RC 滤波电路，以防止由于输入触点抖动或外部干扰脉冲引起的错误输入信号，输入电流一般为数毫安。

图 2-7 是直流输入模块的内部电路和外部接线图，图中只画出了一路输入电路，M 和 N 是同一输入组内各输入信号的公共点。

当图 2-7 中的外接触点接通时，光耦合器中的发光二极管点亮，光敏三极管饱和导通；外接触点断开时，光耦合器中的发光二极管熄灭，光敏三极管截止，信号经背板总线接口传送给 CPU 模块。

交流输入模块的额定输入电压为 AC 120 V 或 230 V。在图 2-8 中用电容隔离输入信号中的直流成分，用电阻限流，交流成分经桥式整流电路转换为直流电流。外接触点接通时，光耦合器中的发光二极管和显示用的发光二极管点亮，光敏三极管饱和导通。外接触点断开时，光耦合器中的发光二极管熄灭，光敏三极管截止，信号经背板总线接口传送给 CPU 模块。

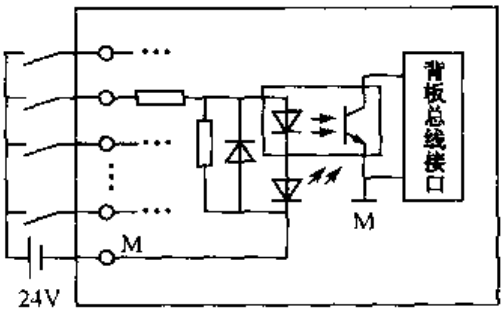


图 2-7　数字量输入模块

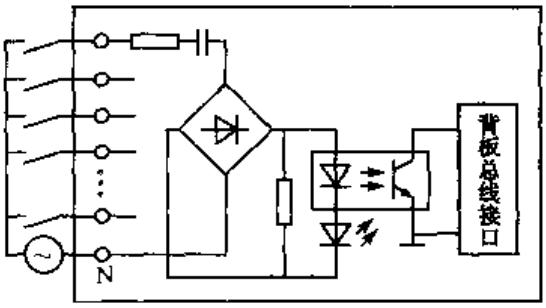


图 2-8　数字量输入模块

直流输入电路的延迟时间较短，可以直接与接近开关、光电开关等电子输入装置连接，DC 24 V 是一种安全电压。如果信号线不是很长，PLC 所处的物理环境较好，电磁干扰较轻，应考虑优先选用 DC 24 V 的输入模块。交流输入方式适合于在有油雾、粉尘的恶劣环境下使用。

数字量输入模块可以直接连接两线式接近开关(BERO)，两线式 BERO 的输出信号为 0 时，其输出电流(漏电流)不为 0。在选型时应保证两线式 BERO 的漏电流小于输入模块允许的静态电流，否则将会产生错误的输入信号。

根据输入电流的流向，可以将输入电路分为源输入电路和漏输入电路。

漏输入电路(见图 2-7)输入回路的电流从模块的信号输入端流进来，从模块内部输入电路的公共点 M 流出去。PNP 集电极开路输出的传感器应接到漏输入的数字量输入模块。

源输入电路输入回路的电流从模块的信号输入端流出去，从模块内部输入电路的公共点 M 流进来。NPN 集电极开路输出的传感器应接到源输入的数字量输入模块。

数字量模块的输入/输出电缆最大长度为 1000 m(屏蔽电缆)或 600 m(非屏蔽电缆)。

SM 321 数字量输入模块技术参数见表 2-7。

表 2-7　SM321 数字量输入模块技术参数

6ES7 321-	1BH02-0AA0 1BH82-0AA0	1BH50-0AA0	1BL00-0AA0 1BL80-0AA0	1CH00-0AA0	1CH80-0AA0
输入点数	16	16 源输入	32	16	16
额定输入电压	DC 24 V	DC 24 V	DC 24 V	AC/DC 24～48 V	DC 48～125 V
隔离，分组数	光耦，16 组	16 组	16 组	光耦，1 组	光耦，8 组

（续）

6ES7 321-	1BH02-0AA0 1BH82-0AA0	1BH50-0AA0	1BL00-0AA0 1BL80-0AA0	1CH00-0AA0	1CH80-0AA0
输入电流	9 mA	7 mA	7 mA	8 mA	2.6 mA
输入延迟时间	1.2～4.8 ms	1.2～4.8 ms	1.2～4.8 ms	最大 15 ms	1～3 ms
允许最大静态电流	1.5 mA	1.5 mA	1.5 mA	1.0 mA	1.0 mA
6ES7 321-	7BH00-0AB0 7BH80-0AB0	1FH00-0AA0	1EL00-0AA0	1FF01-0AA0 1FF81-0AA0	1FF10-0AA0
输入点数	16,有中断功能	16	32	8	8
额定输入电压	DC 24 V	AC 120/230 V	AC 120 V	AC 120/230 V	AC 120/230 V
隔离与分组数	光耦,16 组	光耦,4 组	光耦,8 组	光耦,2 组	光耦,1 组
输入电流	7 mA	17.3mA(AC 264V)	21 mA	11 mA(230 V)	17.3mA(230V)
输入延迟时间	0.1/0.5/3/ 15/20 ms 可选	25 ms	25 ms	25 ms	25 ms
允许最大静态电流	1.5 mA	2 mA	4 mA	2 mA	2 mA

2.4.2 数字量输出模块

SM 322 数字量输出模块用于驱动电磁阀、接触器、小功率电动机、灯和电动机起动器等负载。数字输出模块将 S7-300 的内部信号电平转化为控制过程所需的外部信号电平，同时有隔离和功率放大的作用。

输出模块的功率放大元件有驱动直流负载的大功率晶体管和场效应晶体管、驱动交流负载的双向晶闸管或固态继电器，以及既可以驱动交流负载又可以驱动直流负载的小型继电器。输出电流的典型值为 0.5～2 A，负载电源由外部现场提供。

SM 322 数字量输出模块技术参数见表 2-8。

表 2-8 SM322 数字量输出模块技术参数

6ES7 322-	1BH01-0AA0 1BH81-0AA0	1BL00-0AA0	8BF00-0AB0 8BF80-0AB0	5GH00-0AB0	1CF80-0AA0	1BF01-0AA0
输出点数	16	32	8,有中断功能	16	8	8
额定输入电压	DC 24 V	DC 24 V	DC 24 V	DC 24/48 V	DC 48～125 V	DC 24 V
分组数,隔离	8,光耦	8,光耦	8,光耦	1,光耦	4,光耦	4,光耦
最大输出电流(60℃) 最大灯负载 最小电流	0.5 A 5 W 5 mA	0.5 A 5 W 5 mA	0.5 A 5 W 10 mA	0.5 A 5 W	1.5 A 40 W/120 V 10 mA	2 A 10 W 5 mA
感性负载最大输出频率 阻性负载最大输出频率 灯负载最大输出频率	100 Hz 0.5 Hz 100 Hz	100 Hz 0.5 Hz 100 Hz	100 Hz 2 Hz 100 Hz	0.5 Hz — —	20 Hz 0.5 Hz 10 Hz	100 Hz 0.5 Hz 100 Hz
短路保护	电子式	电子式	电子式	外部提供	电子式	电子式

（续）

6ES7 322-	1FF01-0AA0	5FF00-0AB0	1FH00-0AA0	1EL00-0AA0
输出点数	8	8	16	32
诊断	LED 显示保险丝熔断或无负载电压	关断，上一次值/替代值	红色 LED 显示保险丝熔断	红色 LED 显示保险丝熔断
额定负载电压	AC 120/230 V	AC 120/230 V	AC 120/230 V	AC 120 V
分组数，隔离	4，光耦	1，光耦	8，光耦	8，光耦
输出最大电流/每组总电流 最小电流	1 A/2 A 10 mA	1 A/1 A 10 mA	1 A/2 A 10 mA	1 A/3 A 10 mA
阻性负载最大输出频率 感性负载最大输出频率 灯负载最大输出频率，功率	10 Hz 0.5 Hz 1 Hz，50 W	10 Hz 0.5 Hz 1 Hz，50 W	10 Hz 0.5 Hz 1 Hz，25 W	10 Hz 0.5 Hz 1 Hz，25 W
短路保护	熔断器	外部提供	组后备	熔断器

表 2-9　SM322 继电器型数字量输出模块技术参数

6ES7 322-	1HF01-0AA0	1HF10-0AA0 1HF80-0AA0	5HF00-0AB0	1HH01-0AA0
输出点数	8 继电器	8 继电器	8 继电器	16 继电器
诊断	—	关断，上一次值/替代值	—	—
最高电压	DC 120 V/AC 230 V			
每组点数，隔离	2，光耦	1，光耦	1，光耦	8，光耦
每组总输出电流(60℃)	4 A	—	5 A	8 A
阻性负载最大输出电流	2 A/AC 230 V， 2 A/DC 24 V	AC 8 A/230 V， 5 A/DC 24 V	5 A	2 A/AC 230 V， 2 A/DC 24 V
感性负载最大输出电流	2 A/AC 230 V， 2 A/DC 24 V	3 A/AC 230 V， 2 A/DC 24 V	5 A	2 A/AC 230 V， 2 A/DC 24 V
阻性负载最大输出频率 感性负载最大输出频率 灯负载最大输出频率 机械负载最大输出频率	2 Hz 0.5 Hz 2 Hz 10 Hz	2 Hz 0.5 Hz 2 Hz 10 Hz	2 Hz 0.5 Hz 2 Hz 10 Hz	1 Hz 0.5 Hz 1 Hz 10 Hz
触点寿命(AC 230 V)	2 A，10^5	3 A，10^5	5 A，10^5	2 A，10^5
短路保护	外部提供			

图 2-9 是继电器输出电路，某一输出点为 1 状态时，梯形图中的线圈“通电”，通过背板总线接口和光耦合器，使模块中对应的微型硬件继电器线圈通电，其常开触点闭合，使外部的负载工作。输出点为 0 状态时，梯形图中的线圈“断电”，输出模块中的微型继电器的线圈也断电，其常开触点断开。

图 2-10 是固态继电器输出电路，小框内的光敏晶闸管和小框外的双向晶闸管等组成固态继电器(SSR)。SSR 的输入功耗低，输入信号电平与 CPU 内部的电平相同，同时又实现了隔离，并且有一定的带负载能力。梯形图中某一输出点为 1 状态时，其线圈“通电”，光敏晶闸管中的发光二极管点亮，光敏双向晶闸管导通，使另一个容量较大的双向晶闸管导通，模块外部的负载得电工作。图 2-10 中的 RC 电路用来抑制晶闸管的关断过电压和外部的浪涌电压。这类模块只能用于交流负载，因为是无触点开关输出，其开关速度快，工作寿命长。

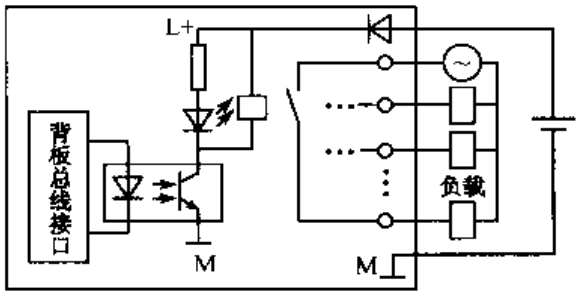


图 2-9　数字量输出模块

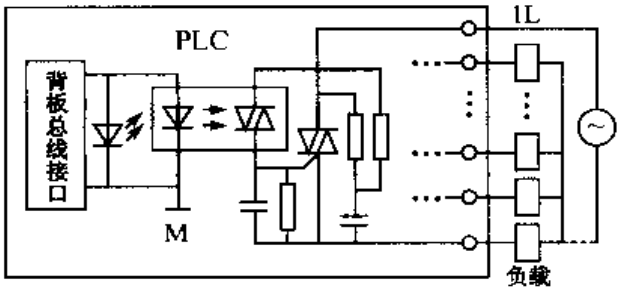


图 2-10　数字量输出模块

双向晶闸管由关断变为导通的延迟时间小于 1 ms,由导通变为关断的最大延迟时间为10 ms(工频半周期)。如果因负载电流过小晶闸管不能导通,可以在负载两端并联电阻。

图 2-11 是晶体管或场效应晶体管输出电路,只能驱动直流负载。输出信号经光耦合器送给输出元件,图中用一个带三角形符号的小方框表示输出元件。输出元件的饱和导通状态和截止状态相当于触点的接通和断开。这类输出电路的延迟时间小于 1 ms。

继电器输出模块的负载电压范围宽,导通压降小,承受瞬时过电压和过电流的能力较强,但是动作速度较慢,寿命(动作次数)有一定的限制。如果系统输出量的变化不是很频繁,建议优先选用继电器型的。

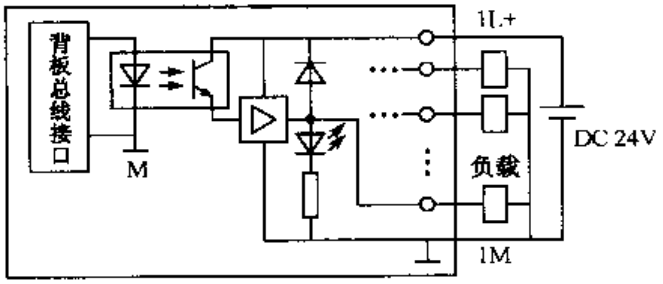


图 2-11　数字量输出模块

固态继电器型输出模块只能用于交流负载,晶体管型、场效应晶体管型输出模块只能用于直流负载,它们的可靠性高,响应速度快,寿命长,但是过载能力稍差。

在选择数字量输出模块时,应注意负载电压的种类和大小、工作频率和负载的类型(电阻性、电感性负载、机械负载或白炽灯)。除了每一点的输出电流外,还应注意每一组的最大输出电流。

2.4.3　数字量输入/输出模块

SM 323 是 S7-300 的数字量输入/输出模块,它有两种型号可供选择。一种是 8 点输入和 8 点输出的模块,输入点和输出点均只有一个公共端。另外一种有 16 点输入(8 点 1 组)和 16 点输出(8 点 1 组)。输入、输出的额定电压均为 DC 24 V,输入电流为 7 mA,最大输出电流为 0.5 A,每组总输出电流为 4 A。输入电路和输出电路通过光耦合器与背板总线相连,输出电路为晶体管型,有电子保护功能。

2.4.4　模拟量输入模块

S7-300 的模拟量 I/O 模块包括模拟量输入模块 SM331、模拟量输出模块 SM332 和模拟

量输入/输出模块 SM334 和 SM335。

1. 模拟量变送器

生产过程中有大量的连续变化的模拟量需要用 PLC 来测量或控制。有的是非电量，例如温度、压力、流量、液位、物体的成分(例如气体中的含氧量)和频率等。有的是强电电量，例如发电动机组的电流、电压、有功功率和无功功率、功率因数等。变送器用于将传感器提供的电量或非电量转换为标准的直流电流或直流电压信号，例如 DC 0～10 V 和 DC 4～20 mA。

2. SM331 模拟量输入模块的基本结构

模拟量输入模块用于将模拟量信号转换为 CPU 内部处理用的数字信号，其主要组成部分是 A/D (Analog/Digit)转换器。模拟量输入模块的输入信号一般是模拟量变送器输出的标准直流电压、电流信号。SM331 也可以直接连接不带附加放大器的温度传感器(热电偶或热电阻)，这样可以省去温度变送器，不但节约了硬件成本，控制系统的结构也更加紧凑。

塑料机壳面板上的红色 LED 用于显示故障和错误，前门的后面是前连接器，前面板上有标签区。模块安装在 DIN 标准导轨上，并通过总线连接器与相邻模块连接，输入通道的地址由模块所在的位置决定。

一块 SM331 模块中的各个通道可以分别使用电流输入或电压输入，并选用不同的量程。有多种分辨率可供选择(9～15 位＋符号位，与模块有关)，分辨率不同转换时间也不同。

模拟量转换是顺序执行的，每个模拟量通道的输入信号是被依次轮流转换的。由图 2-12 可知，模拟量输入模块由多路开关、A/D 转换器(ADC)、光隔离元件、内部电源和逻辑电路组成。8 个模拟量输入通道共用一个 A/D 转换器，通过多路开关切换被转换的通道，模拟量输入模块各输入通道的 A/D 转换和转换结果的存储与传送是顺序进行的。

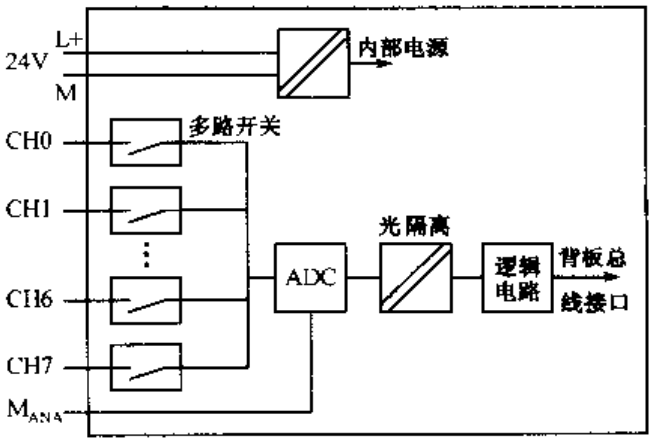


图 2-12　模拟量输入模块

各个通道的转换结果被保存到各自的存储器，直到被下一次的转换值覆盖。可以用装入指令“L PIW...”来访问转换的结果。

有的 SM331 模块具有中断功能，通过中断将诊断信息传送给 CPU 模块。

3. 模拟量输入模块的扫描时间

通道的转换时间由基本转换时间和模块的电阻测试和断线监控时间组成，基本转换时间取决于模块量输入模块的转换方法(例如积分法和瞬时值转换法)。对于积分转换法，积分时间直接影响转换时间，积分时间可在 STEP 7 中设置。

扫描时间是指模拟量输入模块对所有被激活的模拟量输入通道进行转换和处理的时间的总和。如果模拟量输入通道进行了通道分组，还需要考虑通道组之间的转换时间。

为了减小扫描时间，应使用 STEP 7 中的硬件组态工具屏蔽掉未用的模拟量输入通道，在硬件上还需要将未用的通道的输入端短路。

4. 模拟量输入模块的误差

运行误差极限是指在模块的整个允许的温度范围内，在模块的正常测量范围或输出范围，模拟量模块的最大相对测量误差或相对输出误差。

基本误差极限是指在模块的正常工作范围内，25℃时模拟量模块的测量误差或输出误差。

【例 2-1】 某模拟量输出模块为 4 通道 12 位模拟量输出模块，假设输出范围为 0～10 V，模块的环境工作温度为 30℃，模块的电压输出运行极限为 ±0.5%，因此在整个模块的正常输出范围内，最大输出误差应为 ±0.05 V(10 V 的 ±0.5%)。

如果实际输出电压为 1 V，模块的输出范围应为 0.95～1.05 V。此时的相对误差为

$$(0.05V/1V)\times 100\% = \pm 5\ \%$$

5. S7-300 模拟量输入模块的技术参数(见表 2-10)

表 2-10 SM331 模拟量输入模块技术规范

6ES7 331-	7KF02-0AB0	1KF00-0AB0	7KB02-0AB0 7KB82-0AB0	7PF00-0AB0	7PF10-0AB0	7NF00-0AB0	7NF10-0AB0
输入点数 用于电阻测量	8 4	8 8	2 1	8	8	8	8
极限值中断 诊断中断	可组态 通道 0,2	— —	可组态 通道 0	可组态 每个通道	可组态 每个通道	可组态通道 0,2 可组态	所有通道 可组态
额定输入电压 反极性保护	DC 24V 有		DC 24V 有	DC 24V 有	DC 24V 有		
电压输入量程/ 输入阻抗	±80mV/10MΩ ±250mV/10MΩ ±500mV/10MΩ ±1V/10MΩ ±2.5V/100kΩ ±5V/100kΩ 1～5V/100kΩ ±10V/100kΩ	±50mV/10MΩ ±500mV/10MΩ ±1V/10MΩ ±5V/100kΩ 1-5V/100kΩ ±10V/100kΩ 0～10V/100kΩ	±80mV/10MΩ ±250mV/10MΩ ±500mV/10MΩ ±1V/10MΩ ±2.5V/100kΩ ±5V/100kΩ 1～5V/100kΩ ±10V/100kΩ			±5V/2MΩ 1～5V/2MΩ ±10V/2MΩ	±5V/10MΩ 1～5V/10MΩ ±10V/10MΩ
电流输入量程/ 输入阻抗	±10mA/25Ω ±3.2mA/25Ω ±20mA/25Ω 0～20mA/25Ω 4～20mA/25Ω	±20mA/50Ω 0-20mA/50Ω 4-20mA/50Ω	±10mA/25Ω ±3.2mA/25Ω ±20mA/25Ω 0～20mA/25Ω 4～20mA/25Ω			±20mA/250Ω 0～20mA/250Ω 4～20mA/250Ω	±20mA/250Ω 0～20mA/250Ω 4～20mA/250Ω
电阻输入量程/ 输入阻抗	150 /10MΩ 300 /10MΩ 600 /10MΩ	0～600 /10MΩ 0～6000 /10MΩ	150 /10MΩ 300 /10MΩ 600 /10MΩ	150 /10MΩ 300 /10MΩ 600 /10MΩ			
热电偶的型号	E, N, J, K / 10MΩ	E, N, J, K / 10MΩ	E, N, J, K / 10MΩ		B,E,J,K,L,N,R,S,T,U		
热电阻的型号/ 输入阻抗	Pt100 /10MΩ 标准型 Ni100 标准型	Pt100 /10MΩ 标准型 Ni100 气候型	Pt100 /10MΩ 标准型 Ni100 标准型	Pt100,Pt200 Pt500,Pt1000 Ni100,Ni120 Ni200,Ni500 Ni1000,Cu10			
2 线电流变送器 4 线电流变送器	可以 可以	可以，外部供电 可以	可以 可以			带外部变送器 可以	带外部变送器 可以

（续）

6ES7 331-	7KF02-0AB0	1KF00-0AB0	7KB02-0AB0 7KB82-0AB0	7PF00-0AB0	7PF10-0AB0	7NF00-0AB0	7NF10-0AB0
转换时间/通道	2.5/16.7/20/100ms	1.67/20ms	2.5/16.7/20/100ms			2.5/16.7/20/100ms	8通道 23/72/83/95ms
基本转换时间 最多4通道 5通道以上	— —	— —	—	10ms/模块 190ms/模块	10ms/模块 190ms/模块	— —	— —
单极性分辨率 双极性分辨率	9/12/12/14位 9/12/12/14位+符号位	13/13位 12/12位+符号位	9/12/12/14位 9/12/12/14位+符号位	— —	— —	15/15/15/15位 15/15/15/15位+符号位	15/15/15/15位 15/15/15/15位+符号位
干扰抑制频率	400/60/50/10Hz	60/50Hz	400/60/50/10Hz	400/60/50/10Hz	400/60/50/10Hz	400/60/50/10Hz	400/60/50/10Hz
运行误差极限 对应输入范围	±0.1%	±0.6%, ±1.2K	±1%	±0.1%, ±1K	±0.1%, ±1K	±0.1%(电压) ±0.3%(电流)	±0.1%(电压) ±0.3%(电流)
基本误差,25℃ 对应输入范围	±0.6%	±0.4%, ±1.2K	±0.6%	±0.05%, ±0.5K	±0.05%, ±0.5K	±0.05%	±0.05%

除了1KF00-0AB0,其余模块均用红色LED指示组故障,可以读取诊断信息。模块与背板总线之间有隔离,热电偶、热电阻输入时均有线性化处理。使用蔽电缆时最大距离为200 m,输入信号为50 mV或80 mV时,最大距离为50 m。

6. 模拟输入量转换后的模拟值表示方法

模拟量输入/输出模块中模拟量对应的数字称为模拟值,模拟值用16位二进制补码定点数来表示。最高位(第15位)为符号位,正数的符号位为0,负数的符号位为1。

模拟量模块的模拟值位数(即转换精度)可以设置为9～15位(与模块的型号有关,不包括符号位),如果模拟值的精度小于15位,则模拟值左移,使其最高位(符号位)在16位字的最高位(第15位),模拟值左移后未使用的低位则填入"0",这种处理方法称为"左对齐"。设模拟值的精度为12位加符号位,未使用的低位(第0～2位)为0,相当于实际的模拟值被乘以8。

表2-11给出了模拟量输入模块的模拟值与模拟量之间的对应关系,模拟量量程的上、下限(±100%)分别对应于十六进制模拟值6C00H和9400H(H表示十六进制数)。

表2-11 SM331模拟量输入模块的模拟值

范围	双极性						单极性					
	百分比	十进制	十六进制	±5V	±10V	±20mA	百分比	十进制	十六进制	0～10V	0～20mA	4～20mA
上溢出	118.515%	32767	7FFFH	5.926V	11.851V	23.70mA	118.515%	32767	7FFFH	11.852 V	23.70mA	22.96mA
超出范围	117.589%	32511	7EFFH	5.879V	11.759V	23.52mA	117.589%	32511	7EFFH	11.759V	23.52mA	22.81mA
正常范围	100.000%	27648	6C00H	5V	10V	20mA	100.000%	27648	6C00H	10V	20mA	20mA
	0%	0	0H	0V	0V	0mA	0%	0	0H	0V	0mA	4mA
	-100.000%	-27648	9400H	-5V	-10V	-20mA						
低于范围	-117.593%	-32512	8100H	-5.879V	11.759V	-23.52mA	-17.593%	-4864	ED00H		-3.52mA	1.185mA
下溢出	-118.519%	-32768	8000H	5.926V	-11.851V	-23.70mA						

模拟量输入模块在模块通电前或模块参数设置完成后第一次转换之前,或上溢出时,其模拟值为7FFFH,下溢出时模拟值为8000H。上下溢出时SF指示灯闪烁,有诊断功能的模块可

以产生诊断中断。

7. 模拟量输入模块测量范围的设置

模拟量输入模块的输入信号种类用安装在模块侧面的量程卡(或称量程模块)来设置(见图 2-13)。量程卡安装在模拟量输入模块的侧面,每两个通道为一组,共用一个量程卡,图 2-13 中的模块有 8 个通道,因此有 4 个量程卡。量程卡插入输入模块后,如果量程卡上的标记 C 与输入模块上的标记相对,则量程卡被设置在 C 位置。模块出厂时,量程卡预设在 B 位置。

以模拟量输入模块 6ES7 331-7KF02-0AB0 为例,量程卡的 B 位置(见表 2-12)包括 4 种电压输入;C 位置包括 5 种电流输入;D 位置的测量范围只有 4～20 mA。其余的 21 种温度传感器、电阻测量或电压测量的测量范围均应选择位置 A。使用 STEP 7 中的硬件组态功能可以进一步确定测量范围。

供货时量程卡被设置在默认的 B 位置。如果需要的话,必须重新设置量程卡,以更改测量方法和测量范围。各位置对应的测量方法和测量范围都印在模拟量模块上。设置量程卡时先用螺钉旋具将量程卡从模拟量输入模块中取出来,根据要设置的量程,确定量程卡的位置,再按新的设置将量程卡插入模拟量输入模块中。

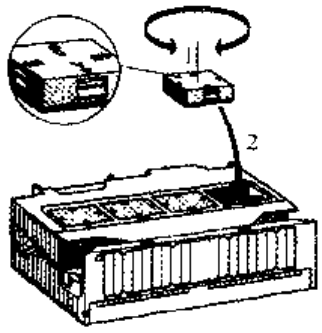


图 2-13 量程卡

表 2-12 模拟量输入模块的默认设置

量程卡设置	测量方法	量 程
A	电压	±1000 mV
B	电压	±10 V
C	4 线变送器电流	4～20 mA
D	2 线变送器电流	4～20 mA

如果没有正确地设置量程卡,将会损坏模拟量输入模块。将传感器连接至模块之前,应确保量程卡在正确的位置。

没有量程卡的模拟量模块可以通过不同的端子接线方式来设置测量的量程。

8. 传感器与模拟量输入模块的接线

为了减少电磁干扰,传送模拟信号时应使用双绞线屏蔽电缆,模拟信号电缆的屏蔽层应两端接地。如果电缆两端存在电位差,将会在屏蔽层中产生等电位线连接电流,造成对模拟信号的干扰。在这种情况下,应将电缆的屏蔽层一点接地。

(1) 带隔离的模拟量输入模块

一般情况下 CPU 的接地端子与 M 端子用短接片相连。带隔离的模拟量输入模块的测量电流参考点 M_{ANA}(见图 2-12)与 CPU 模块的 M 端子之间没有电气连接。

如果测量电流参考点 M_{ANA}和 CPU 的 M 端存在电位差 U_{ISO},必须选用带隔离的模拟量输入模块。通过在 M_{ANA}端子和 CPU 的 M 端子之间使用一根等电位连接导线,可以确保 U_{ISO}不会超过允许值。

(2) 不带隔离的模拟量输入模块

在 CPU 的 M 端子和不带隔离的模拟量输入模块的测量电流参考点 M_{ANA}之间,必须建立电气连接。应连接输入模块的 M_{ANA}端子和 CPU 模块、IM 153 接口模块的 M 端子,否则这些

端子之间的电位差会破坏模拟量信号。

在输入通道的测量线负端 M－和模拟量测量电路的参考点 M_{ANA}之间只会发生有限的电位差 U_{CM}(共模电压)。为了防止超过允许值,应根据传感器的接线情况,采取不同的措施。

(3) 连接带隔离的传感器

带隔离的传感器没有与本地接地电位连接(M 为本地接地端子)。在不同的带隔离的传感器之间会引起电位差。这些电位差可能是因为干扰或传感器的布局造成的。为了防止在具有强烈电磁干扰的环境中运行时超过 U_{CM}的允许值,建议将测量线的负端 M－与 M_{ANA}连接。在连接用于电流测量的两线式变送器、阻性传感器和没有使用的输入通道时,禁止将 M－连接至 M_{ANA}。

(4) 连接不带隔离的传感器

不带隔离的传感器与本地接地电位连接(本地接地)。如果使用不带隔离的传感器,必须将 M_{ANA}连接至本地接地。

由于本地条件或干扰信号,在本地分布的各个测量点之间会造成静态或动态电位差 E_{CM}。如果 E_{CM}超过允许值,必须用等电位连接导线将各测量点的负端 M－连接起来。

如果将不带隔离的传感器连接到有光隔离的模块,CPU 既可以在接地模式下运行(M_{ANA}与 M 点相连),也可以在不接地模式下运行。

如果将不带隔离的传感器连接到不带隔离的输入模块,CPU 只能在接地模式下运行。必须用等电位连接导线将各测量点的负端 M－连接后,再与接地母线相连。

不带隔离的双线变送器和不带隔离的阻性传感器不能与不带隔离的模拟量输入模块一起使用。

2.4.5 将模拟量输入模块的输出值转换为实际的物理量

转化时应考虑变送器的输入/输出量程和模拟量输入模块的量程,找出被测物理量与A/D转换后的数字之间的比例关系。

【例 2-2】 压力变送器的量程为 0～10 MPa,输出信号为 4～20 mA,模拟量输入模块的量程为 4～20 mA,转换后的数字量为 0～27 648,设转换后得到的数字为 N,试求以 kPa 为单位的压力值。

解:0～10 MPa(0～10 000 kPa)对应于转换后的数字 0～27 648,转换公式为

$$P = 10\,000 \times N / 27\,648 \text{ kPa}$$

注意在运算时一定要先乘后除,否则可能会损失原始数据的精度。

【例 2-3】 某发电动机的电压互感器的电压比为 10 kV/100 V(线电压),电流互感器的电流比为 1000 A/5A,功率变送器的额定输入电压和额定输入电流分别为 AC 100 V 和 5 A,额定输出电压为 DC ± 10 V,模拟量输入模块将 DC ± 10 V 输入信号转换为数字 + 27648 和 － 27649。设转换后得到的数字为 N,求以 kW 为单位的有功功率值。

解:根据互感器额定值计算的原边有功功率额定值为

$$\sqrt{3} \times 10\,000 \times 1\,000 \text{ W} = 17\,321\,000 \text{ W} - 17\,321 \text{ kW}$$

由以上关系不难推算出互感器原边的有功功率与转换后的数字之间的关系为 17321/27648 = 0.62648 kW / 字。转换后的数字为 N 时,对应的有功功率为 0.6265 N kW,如果以 kW 为单位显示功率 P,使用定点数运算时的计算公式为

$$P = N \times 6\,265 / 10\,000 \text{ kW}$$

【例 2-4】 用于测量锅炉炉膛压力(－60～60 Pa)的变送器的输出信号为 4～20 mA,模拟量输入模块将 0～20 mA 转换为数字 0～27 648,设转换后得到的数字为 N,试求以 0.1 Pa 为单位的压力值。

解:4～20 mA 的模拟量对应于数字量 5 530～2 7648,即－600～600(0.1 Pa)对应于数字量 5 530～27 648,压力的计算公式应为

$$P = \left[\frac{1200}{(27648-5530)}(N-5530)-600\right]0.1\text{ Pa} = \left[\frac{1200}{22118}(N-5530)-600\right]0.1\text{ Pa}$$

2.4.6 模拟量输出模块

1. 模拟量输出模块的基本结构

S7-300 的模拟量输出模块 SM332 用于将 CPU 送给它的数字信号转换为成比例的电流信号或电压信号,对执行机构进行调节或控制,其主要组成部分是 D/A 转换器(见图 2-14)。可以用传送指令"T PQW..."向模拟量输出模块写入要转换的数值。

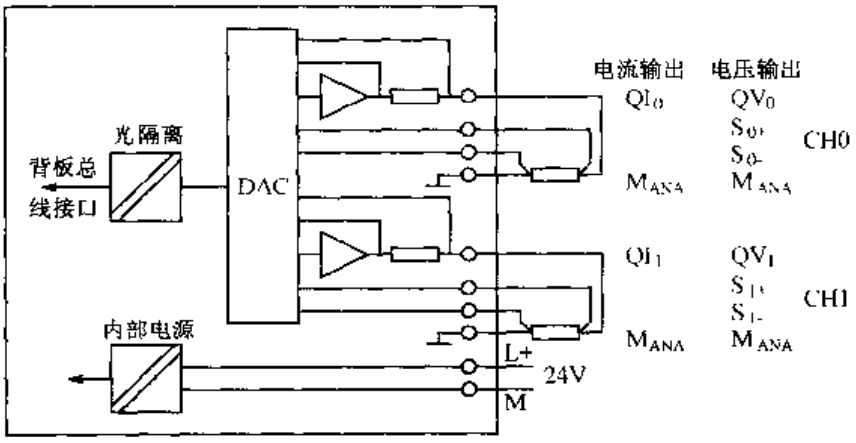


图 2-14 模拟量输出模块

2. 模拟量输出模块的响应时间

模拟量输出模块未通电时输出一个 0 mA 或 0 V 的信号。在处于 RUN 模式、模块有 DC 24 V 电源,且在参数设置之前,将输出前一数值。进入 STOP 模式、模块有 DC 24 V 电源时,可以选择不输出电流电压、保持最后的输出值或采用替代值。在上、下溢出时模块的输出值均为 0 mA 或 0 V。

模拟量输出通道的转换时间由内部存储器传送数字输出值的时间和数字量到模拟量的转换时间组成。循环时间 t_Z 是模拟量输出模块所有被激活的模拟量输出通道的转换时间的总和。应关闭没有使用的模拟量通道,以减小循环时间。

建立时间 t_E 是指从转换结束到模拟量输出到达指定的值的时间,它与负载的性质(阻性负载、容性负载或感性负载)有关。模块的技术规范给出了模拟量输出模块的建立时间与负载之间的函数关系。

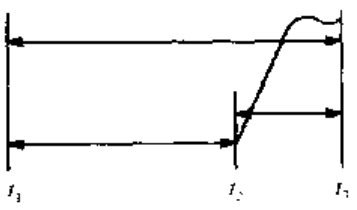


图 2-15

响应时间 t_A 是指内部存储器中得到数字量输出值到模拟量输出达到指定值的时间(见图 2-15),在最坏的情况下,该时间为

循环时间 t_Z 和建立时间 t_E 之和。

3. 模拟量输出模块的技术参数

SM332 的 4 种模拟量输出模块均有诊断中断功能，用红色 LED 指示组故障，可以读取诊断信息。额定负载电压均为 DC 24 V。模块与背板总线有光隔离，使用屏蔽电缆时最大距离为 200 m。都有短路保护，短路电流最大 25 mA，最大开路电压 18 V。SM322 模拟量输出模块技术参数见表 2-13。

表 2-13 SM332 模拟量输出模块技术参数

6ES7 332-	5HB01-0AB0 5HB81-0AB0	5HD01-0AB0	5HF00-0AB0	7ND00-0AB0
输出点数	2	4	8	4
输出范围	0～10 V，±10V，1～5 V，4～20 mA，0～20 mA，±20 mA			
最大负载阻抗	电压输出 1 kΩ，电流输出 0.5 kΩ，容性输出 1 μF，感性 1 mH			
最大转换时间/通道	0.8 ms			1.5 ms
建立时间	阻性负载 0.2 ms，容性负载 3.3 ms，感性负载 0.5 ms			
分辨率	±10 V，±20 mA 时为 11 位＋符号位，其余为 12 位			15 位＋符号位
0～60℃ 工作极限，对应于输出范围	电压 ±0.5%，电流 ±0.6%			电压 ±0.12%，电流 ±0.18%
25℃ 时基本误差，对应于输出范围	电压 ±0.4%，电流 ±0.5%			电压电流均为 ±0.01%

4. 模拟量输出模块与负载或执行器的接线

模拟量输出模块为负载和执行器提供电流和电压，模拟信号应使用屏蔽电缆或双绞线电缆来传送。电缆线 QV 和 S_+，M_{ANA} 和 S_-（见图 2-14）应分别绞接在一起，这样可以减轻干扰的影响，应将电缆两端的屏蔽层接地。

如果电缆两端有电位差，将会在屏蔽层中产生等电动势连接电流，干扰传输的模拟信号。在这种情况下应将电缆屏蔽层一点接地。

对于带隔离的模拟量输出模块，在 CPU 的 M 端和测量电路的参考点 M_{ANA} 之间没有电气连接。如果 M_{ANA} 点和 CPU 的 M 端子之间有电位差 E_{ISO}，必须选用隔离型的模拟量输出模块。在 M_{ANA} 端子和 CPU 的 M 端子之间使用一根等电位连接导线，可以使 E_{ISO} 不超过允许值。

2.4.7 模拟量输入/输出模块

模拟量输入/输出模块 SM 334 和 SM 335 的技术规范如表 2-14 所示。

快速模拟量输入/输出模块 SM 335 提供：

(1) 4 个快速模拟量输入通道，基本转换时间最大为 1 ms；

(2) 4 个快速模拟量输出通道，每通道最大转换时间为 0.8 ms；

(3) 10 V/25 mA 的编码器电源；

(4) 一个计数器输入(24 V/500 Hz)。

SM 355 有两种特殊工作模式：

（1）只进行测量：模块不断地测量模拟量输入值，而不更新模拟量输出。它可以快速测量模拟量值（<0.5 ms）。

（2）比较器：SM 335 对设定值与测量的模拟量输入值进行快速比较。

SM 355 有循环周期结束中断和诊断中断。

表 2-14　SM334、SM335 模拟量输入/输出模块的技术参数

6ES7	334-0CE01-0AA0	334-0KE00-0AB0 334-0KE80-0AB0	335-7HG01-0AB0 快速模拟量输入/输出模块
输入点数	4	4	4
输入范围/输入阻抗	0～10 V/100kΩ, 0～20 mA/50Ω	0～10 V/100kΩ, 电阻 10 kΩ, Pt100	±1V, ±10V, ±2.5V,0～2V,0～10V:10MΩ ±10 mA,0～20 mA,4～20 mA:100Ω
分辨率	8 位	12 位	双极性 13 位＋符号位，单极性 14 位
转换时间		每通道最大 85 ms	200 μs,4 通道最大 1 ms
运行极限	电压 ±0.9%，电流 ±0.8%	电压 ±0.7%，Pt100 ±1.0%	电压 ±0.15%，电流 ±0.25%
基本误差限制	电压 ±0.7%，电流 ±0.6%	电压 ±0.5%，Pt100 ±0.8%	±0.13%
输出点数	2	2	4
输出范围	0～10 V,0～20 mA	0～10 V	0～10 V, ±10 V
负载阻抗	电压输出最小 5 kΩ 电流输出最大 300 Ω	电压输出最小 2.5 kΩ	
分辨率	8 位	12 位	双极性 11 位＋符号位，单极性 12 位
转换时间	每通道最大 0.5 ms	每通道最大 0.5 ms	每通道最大 0.8 ms
运行极限	电压 ±0.8%，电流 ±1.0%	电压 ±1.0%	±0.5%
基本误差限制	电压 ±0.4%，电流 ±0.8%	电压 ±0.85%	±0.2%
扫描时间（AI＋AO）	所有通道 5 ms	所有通道 85 ms	

2.4.8　模拟量模块的诊断与中断

1. 模拟量模块的诊断

诊断报文分为可编程诊断报文和不可编程的诊断报文。需要在 STEP 7 的“诊断”参数区中使能（enable）诊断，才能获得可编程的诊断报文。不管是否使能诊断，通过模拟量模块都可以获得不可编程的诊断报文。

有故障出现时将会执行下列操作：

（1）将诊断报文送入模拟量模块的诊断区中，并传送到 CPU。

（2）点亮模拟量模块中的故障指示灯。

（3）如果已经用 STEP 7 使能产生“诊断中断”，将触发一个诊断中断，并调用 OB 82。

可以通过用户程序中的 SFC，读出详细的诊断报文。在模块诊断中，可以查看 STEP 7 中的故障原因（参见 STEP 7 的在线帮助）。

检测到错误时，不管参数如何设置，模拟量输入模块输出模拟测量值 7FFFH，它意味着上溢出、故障或通道被禁止使用。

每个模拟量模块都通过 SF 指示灯(组故障指示灯)指示出现错误。一旦模拟量模块触发诊断报文,SF 指示灯就被点亮。故障被全部排除后,SF 指示灯熄灭。

模拟量输入模块在下列故障时发出诊断报文:外部辅助电源故障、组态/参数设置出错、共模错误,断线、下溢出和上溢出。只能对 4～20 mA 的输入模块检测断线故障。

模拟量输出模块在下列故障时发出诊断报文:外部辅助电源故障、组态/参数设置出错、M 点短路和断线。

2. 模拟量模块的中断

模拟量模块可以产生诊断中断和过程中断,并不是所有的模拟量模块都具有中断功能,有的只具有下述的部分中断功能。

模拟量模块是否产生中断可以用 STEP 7 来设置,如果没有使能中断,中断将被禁止。

(1) 诊断中断

如果已经允许产生诊断中断,被激活的错误事件(故障产生的报文)和错误事件的消除(故障排除后的报文)都可以通过中断来报告。

出现诊断中断时,CPU 暂时停止用户程序的执行,去处理诊断报警组织块 OB 82。

在用户程序中,OB 82 可以调用 SFC 51 或 SFC 59,从模块中获得更为详细的诊断信息。

(2) 模拟量模块"超出上限或下限"触发的硬件中断

通过设置上限和下限定义一个工作范围。如果过程信号(例如温度)超出或低于该范围,模块将触发一个被允许的过程中断,中断用户程序的执行,去处理硬件中断组织块(OB 40)。应对 OB 40 中的用户程序编程,对超出上限或下限的异常情况进行处理。

(3) "扫描循环结束"时触发的硬件中断

在设置模块的参数时允许扫描循环结束时产生硬件中断,可以使一个过程与模拟量输入模块的扫描循环同步。

一个扫描循环包括转换模拟量输入模块所有被使用的通道的测量值。模块将一个一个地处理通道,在所有被测值都转换完后,将产生报告所有通道中都有新的测量值可以使用的中断,可以在中断程序中处理当前转换的模拟值。

2.4.9 EX 系列与 F 系列输入/输出模块

1. EX 系列数字量模拟量输入/输出模块

EX 模块可以在化工等行业的自动化仪表和控制系统中使用。它们的主要作用是将外部的本质-安全回路与 PLC 的非本质-安全内部回路隔离开。

EX 系列模块包括 EX 数字量输入/输出模块和 EX 模拟量输入/输出模块,可以用于 S7-300 或 ET 200M 分布式 I/O 装置,作为所有 SIMATIC PLC 的分布式 I/O 及 PROFIBUS-DP 网络的标准从站。它们属于"本质-安全型保护"的电子器件,包括非本质-安全回路和本质-安全回路。EX 模块本身应安装在有爆炸危险的区域之外。除非附加另一种类型的保护(例如增压防护),才能应用于有爆炸危险的区域。

将外部的 EX 区域的本质-安全数字设备(用于有爆炸危险区域的传感器和执行器)连接到 EX 模块上,可以实现有爆炸危险的区域与 PLC 系统的非本质-安全内部回路的隔离。传感器和执行元件由模块供电。

本质-安全型防护有以下优点:在操作过程中可以方便地更换本质-安全设备和对被测系

统进行测量和校准，PLC 不需要昂贵的增压防护的防爆机壳。

2. F 系列数字量模拟量输入/输出模块

F 系列输入/输出模块是 S7-400F/FH 和 S7-300F 的输入/输出模块。SM 326 F 数字量输入-安全集成模块用于连接有爆炸危险区域的触点和两线式接近开关(BERO)，SM 326 F 数字量输出-安全集成模块用于连接执行阀和指示灯等负载，SM 336F 模拟量输入-安全集成模块用于连接模拟量电压、电流信号传感器或变送器。这些模块具有故障安全运行的集成安全功能，适用于在 ET 200M 分布式 I/O 或 S7-300F 中使用，可以像 S7-300 模块一样在标准运行中使用。

2.5 S7-300 的其他模块

2.5.1 计数器模块

1. 计数器模块的共同性能

模块的计数器均为 0～32 位或 31 位加减计数器，可以判断脉冲的方向，模块给编码器供电。有比较功能，达到比较值时，通过集成的数字量输出响应信号，或通过背板总线向 CPU 发出中断。可以 2 倍频和 4 倍频计数，4 倍频是指在两个互差 90°的 A、B 相信号的上升沿、下降沿都计数。通过集成的数字量输入直接接收起动、停止计数器等数字量信号。

2. FM 350-1 计数器模块

FM 350-1 是智能化的单通道计数器模块，可以检测最高达 500 kHz 的脉冲，有连续计数、单向计数、循环计数 3 种工作模式。有 3 种特殊功能：设定计数器、门计数器和用门功能控制计数器的启/停。达到基准值、过零点和超限时可以产生中断。有 3 个数字量输入，2 个数字量输出

3. FM 350-2 计数器模块

FM 350-2 是 8 通道智能型计数器模块，有 7 种不同的工作方式：连续计数、单次计数、周期计数、频率测量、速度测量、周期测量和比例运算。

对于 24 V 增量编码器，计数的最高频率为 10 kHz，对于 24 V 方向传感器，24 V 起动器和 NAMUR 编码器，为 20 kHz。

4. CM 35 计数器模块

CM 35 是 8 通道智能计数器模块，可以执行通用的计数和测量任务，也可以用于最多 4 轴的简单定位控制。CM 35 有 4 种工作方式：加计数或减计数、8 通道定时器、8 通道周期测量和 4 轴简易定位。8 个数字量输出点用于对模块的高速响应输出，也可以由用户程序指定输出功能，计数频率每通道最高 10 kHz。

2.5.2 位置控制与位置检测模块

1. 位置控制模块概述

FM 351 双通道定位模块用于控制对动态调节特性要求高的轴的定位，该模块用于控制变级调速电动机或变频器。

定位模块可以用编码器来测量位置并向编码器供电，使用步进电动机的位置控制系统一般不需要位置测量，建议时钟脉冲速率高和对动态调节特性要求高的定位系统选用 FM 353

步进电动机定位模块。对于不仅要求很高的动态性能，还要求高精度的定位系统，最好使用FM 354 伺服电动机定位模块。

FM 357 可以用于最多 4 个插补轴的协同定位，既能用于伺服电动机也能用于步进电动机。

在定位控制系统中，定位模块控制步进电动机或伺服电动机的功率驱动器，CPU 模块用于顺序控制和起动、停止定位操作，计算机用集成在 STEP 7 中的参数设置屏幕格式，对定位模块进行参数设置，并建立运动程序，设置的数据存储在定位模块中。操作面板在运行时用来实现人机接口和故障诊断功能。CPU 或组态软件选择目标位置或移动速度，定位模块完成定位任务，用模块集成的数字量输出点来控制快速进给、慢速进给和运动方向等。根据与目标的距离，确定慢速进给或快速进给，定位完成后给 CPU 发出一个信号。定位模块的定位功能独立于用户程序。

2. FM 351 快速/慢速进给驱动位置控制模块

FM 351 是双通道定位模块，可以控制两个相互独立的轴的定位。有下述定位功能：

(1) 设置：按点动按钮来操作快速移动或慢速移动，使轴到达准确位置(微动方式)。

(2) 绝对增量方式：轴移动到一个绝对的目标位置，数值存储在 FM 351 的表格中。

(3) 相对增量方式：轴移动一个预设的距离。

(4) 参考点方式：使用增量式编码器时，接通控制器后同步用。

特殊功能包括零点偏置、设定基准点和删除剩余行程。

3. FM 352 电子凸轮控制器

FM 352 高速电子凸轮控制器是机械式凸轮控制器的低成本替代产品，它有 32 个凸轮轨迹，13 个集成的数字输出端用于动作的直接输出，采用增量式编码器或绝对式编码器。

FM 352 用编码器检测位置，通过集成的输出端触发控制指令。

S7-300 CPU 用于顺序控制、凸轮处理的起动和停止、凸轮参数的传输和凸轮轨迹分析。

凸轮个数可以设置为 32、64 和 128 个。凸轮可以被定义为位置凸轮或时间凸轮，可以改变凸轮的方向，为每个凸轮提供动态补偿。FM 352 具有下列特殊功能：长度测量、设定基准点和实际值、零点补偿、改变凸轮的轨迹，可以进行仿真。

4. FM 352 高速布尔处理器

FM 352 高速布尔处理器高速地进行布尔控制(即数字量控制)，集成了 12 点数字量输入和 8 点数字量输出。指令集包括位指令、定时器、计数器、分频器、频率发生器和移位寄存器指令。1 个通道用于连接 24V 增量式编码器，1 个 RS-422 串口用于连接增量式或绝对式编码器。

5. FM 353 步进电动机定位模块

FM 353 是在高速机械设备中使用的步进电动机定位模块。它可以满足从简单的点到点定位，到对响应、精度和速度有极高要求的复杂运动模式。它将脉冲传送到步进电动机的功率驱动器，通过脉冲数量控制移动距离，用脉冲的频率控制移动速度。

FM 353 有使用按钮的点动模式和增量模式，有手动数据输入功能，自动/单段控制用于运行复杂的定位路径。FM 353 具有下列特殊功能：长度测量、变化率限制、运行中设置实际值、通过高速输入使定位运动起动或停止。

6. FM 354 伺服电动机定位模块

FM 354 是在高速机械设备中使用的伺服电动机的智能定位模块，用于从点到点定位任

务到对响应、精度和速度要求极高的复杂运动方式。它用模拟驱动接口(－10～＋10 V)控制驱动器,利用编码器检测的轴位置来修正输出电压。FM 354 与 FM 353 的工作模式和定位功能相同。

7. FM 357-2 定位和连续路径控制模块

模块用于从独立的单独定位轴控制到最多 4 轴直线、圆弧插补连续路径控制。可以控制步进电动机或伺服电动机。4 个测量回路用于连接伺服轴、步进驱动器或外部主轴。

可以通过联动运动或曲线图表(电子曲线盘)进行轴同步,也可以通过外部主信号实现。采用编程或软件加速的运动控制和可转换的坐标系统,有高速再起动的特殊急停程序。有点动、增量进给、参考点、手动数据输入、自动、自动单段等工作方式。

8. FM STEPDRIVE 步进电动机功率驱动器

它与 FM 353,FM 357-2 定位模块配套使用,用来控制 5～600 W 的步进电动机。

9. SM 338 超声波位置解码器模块

SM 338 用超声波传感器检测位置,具有无磨损、保护等级高、精度稳定不变、与传感器的长度无关等优点。模块最多接 4 个传感器,每个传感器最多有 4 个测量点,测量点数最多 8 个。测量范围 3～6 m,分辨率 0.05 mm(测量范围最多 3 m)或 0.1 m,可编程的测量时间为 0.5～16 ms。RS-422 接口抗干扰能力强,电缆最长 50 m。

10. SM 338 位置输入模块

SM 338 可以提供最多 3 个绝对值编码器 (SSI) 和 CPU 之间的接口,将 SSI 的信号转换为 S7-300 的数字值,可以为编码器提供 DC 24 V 电源。此外可以提供两个内部数字输入点将 SSI 位置编码器的状态锁住,可以在位置编码区域内处理对时间要求很高的应用。

2.5.3 闭环控制模块

1. FM 355 闭环控制模块

FM 355 有 4 个闭环控制通道,用于压力、流量、液位等控制,有自优化温度控制算法和 PID 算法。FM 355C 是有 4 个模拟量输出端的连续控制器,FM 355S 是有 8 个数字输出点的步进或脉冲控制器。CPU 停机或出现故障后 FM 355 仍能继续运行,控制程序存储在模块中。

FM 355 的 4 个模拟量输入端用于采集模拟数值和前馈控制,附加的一个模拟量输入端用于热电偶的温度补偿。可以使用不同的传感器,例如热电偶、Pt100 热电阻、电压传感器和电流传感器。FM 355 有 4 个单独的闭环控制通道,可以实现定值控制、串级控制、比例控制和 3 分量控制,几个控制器可以集成到一个系统中使用。有自动、手动、安全、跟随、后备这几种操作方式。12 位分辨率时的采样时间为 20～100 ms,14 位分辨率时为 100～500 ms。

自优化温度控制算法存储在模块中,当设定点变化大于 12%时自动启动自优化;可以使用组态软件包对 PID 控制算法进行优化。

CPU 有故障或 CPU 停止运行时控制器可以独立地继续控制。为此在“后备方式”功能中,设置了可调的安全设定点或安全调节变量。

可以读取和修改模糊温度控制器的所有参数,或在线修改其他参数。

2. FM 355-2 闭环控制模块

FM 355-2 是适用于温度闭环控制的 4 通道闭环控制模块,可以方便地实现在线自优化温度控制,包括加热、冷却控制,以及加热、冷却的组合控制。FM 355-2C 是有 4 个模拟量输出端

的连续控制器，FM 355-2S 是有 8 个数字输出端的步进或脉冲控制器。CPU 停机或出现故障后 FM 355 仍能继续运行。

2.5.4 称重模块

1. SIWAREX U 称重模块

SIWAREX U 是紧凑型电子称，用于化学工业和食品工业等行业来测定料仓和贮斗的料位，对起重机载荷进行监控，对传送带载荷进行测量或对工业提升机、轧机超载进行安全防护等。可以作为功能模块集成到 S7/M7-300 中，也可以通过 ET 200M 连接到 S7 系列 PLC。

SIWAREX U 有下列功能：衡器的校准、重量值的数字滤波、重量测定、衡器置零、极限值监控和模块的功能监视，模块有多种诊断功能。

SIWAREX U 有单通道和双通道两种型号，分别连接 1 台或 2 台衡器。SIWAREX U 有两个串行接口，RS-232C 接口用于连接设置参数用的计算机，TTY 接口用于连接最多 4 台数字式远程显示器。模块的参数可以用组态软件 SIWATOOL 设置，并存入磁盘。

2. SIWAREX M 称重模块

SIWAREX M 是有校验能力的电子称重和配料单元。可以用它组成多料秤称重系统。可以准确无误地关闭配料阀，达到最佳的配料精度。它可以作为功能模块集成到 S7/M7-300，也可以通过 ET 200M 连接到 S5/S7 系列 PLC。

SIWAREX M 有下列功能：置零和称皮重、自动零点追踪、设置极限值（Min/Max/空值/过满）、操纵配料阀（粗/精配料）、称重静止报告和配料误差监视。

SIWAREX M 可以安装在易爆区域，可选的 Ex-i 接口保证对称重传感器的馈电符合本征安全条件。SIWAREX M 还可以作为独立于 PLC 的现场仪器使用。它有一个称重传感器通道，3 个数字输入端和 4 个数字输出端用于选择称重功能，1 个模拟量输出端用于连接模拟显示器或在线记录仪等。RS-232C 串行接口用于连接 PC 机或打印机，TTY 串行接口用于连接有校验能力的数字远程显示器或主机。

2.5.5 电源模块

PS 307 电源模块将 AC 120/230 V 电压转换为 DC 24 V 电压，为 S7-300/400、传感器和执行器供电。输出电流有 2 A、5 A 或 10 A 3 种。

电源模块安装在 DIN 导轨上的插槽 1，紧靠在 CPU 或扩展机架上 IM 361 的左侧，用电源连接器连接到 CPU 或 IM 361 上。

PS 307 10 A 电源模块的框图如图 2-16 所示，模块的输入和输出之间有可靠的隔离。输出正常电压 24 V 时，绿色 LED 亮；输出过载时 LED 闪烁；输出电流大于 13 A 时，电压跌落，跌落后自动恢复。输出短路时输出电压消失，短路消失后电压自动恢复。

电源模块除了给 CPU 模块提供电源外，还要给输入/输出模块提供 DC 24 V 电源。

CPU 模块上的 M 端子（系统的参考点）一般是接地的，接地端子与 M 端子用短接片连接。某些大型工厂（例如化工厂和发电厂）为了监视对地的短路电流，可能采用浮动参考电位，这时应将 M 点与接地点之间的短接片去掉，可能存在的干扰电流通过集成在 CPU 中 M 点与接地点之间的 RC 电路（见图 2-17）对接地母线放电。

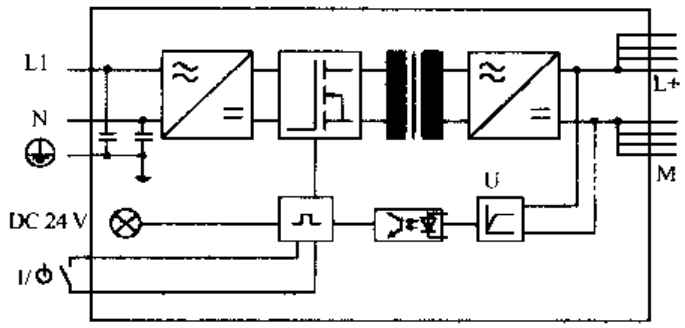


图 2-16　PS 307 电源模块框图

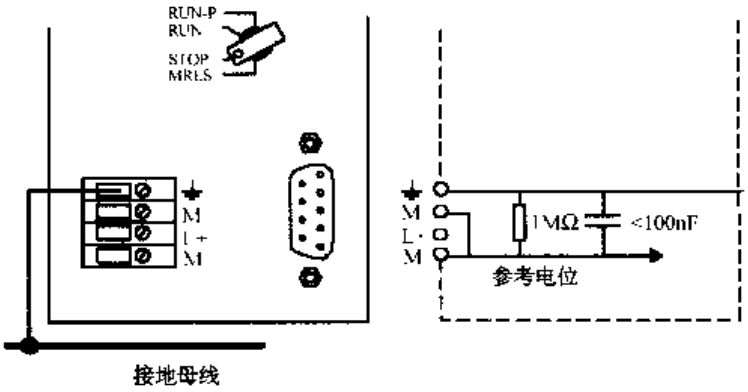


图 2-17　S7-300 的浮动参考电位

2.5.6　前连接器与其他模块

1. 前连接器

前连接器用于将传感器和执行元件连接到信号模块。它被插入到模块上，有前盖板保护。更换模块时接线仍然在前连接器上，只需要拆下前连接器，不用花费很长时间重新接线。模块上有两个带顶罩的编码元件，第一次插入时，顶罩永久地插入到前连接器上。为避免更换模块时发生错误，第一次插入前连接器时，它就已被编码，前连接器以后只能插入同样类型的模块。

20 针的前连接器用于信号模块(32 通道模块除外)、功能模块和 312 IFM CPU。40 针的前连接器用于 32 通道信号模块。

2. TOP 连接器

TOP 连接器包括前连接器模块、连接电缆和端子块。所有部件均可以方便地连接，并可以单独更换。TOP 全模块化端子允许方便、快速和无错误地将传感器和执行元件连接到 S7-300，最长距离 30 m。模拟信号模块的负载电源 L + 和地 M 的允许距离为 5 m。超过 5 m 时前连接器一端和端子块一端均需要加电源。

前连接器模块代替前连接器插入到信号模块上，用于连接 16 通道或 32 通道信号模块。

如果总电流超过 4 A，不要通过连接电缆将外部电源送给信号模块。这种情况下，电源应直接接到前连接器模块。

3. 仿真模块

仿真模块 SM 374 用于调试程序，用开关来模拟实际的输入信号，用 LED 显示输出信号的状态。模块上有一个功能设置开关，可以仿真 16 点输入、16 个点输出，或 8 点输入/8 点输

出，具有相同的起始地址。

用 STEP 7 给仿真模块的参数赋值时，应使用被仿真的模块的型号。例如 SM 374 被设置为 16 点输入时，组态时应输入某一 16 点数字量输入模块的订货号。

4. 占位模块

占位模块 DM 370 为模块保留一个插槽，如果用一个其他模块替换占位模块，整个配置和地址设置保持不变。占用两个插槽的模块，必须使用两个占位模块。

模块上有一个开关，开关在 NA 位置时，占位模块为一个接口模块保留插槽，NA 表示没有地址，即不保留地址空间，不用 STEP 7 进行组态。

开关在 A 位置时，占位模块为一个信号模块保留插槽，A 表示保留地址，需要用 STEP 7 对占位模块进行组态。

2.6 S7-400 系列 PLC 的硬件组成

2.6.1 S7-400 的基本结构与特点

1. S7-400 的基本结构

S7-400 是具有中高档性能的 PLC，采用模块化无风扇设计，适用于对可靠性要求极高的大型复杂的控制系统。S7-400 采用大模块结构，大多数模块的尺寸为 25 mm(宽)×290 mm(高)×210 mm(深)。

如图 2-18 所示，S7-400 由机架、电源模块(PS)、中央处理单元(CPU)、数字量输入/输出(DI/DO)模块、模拟量输入/输出(AI/AO)模块、通信处理器(CP)、功能模块(FM)和接口模块(IM)组成，DI/DO 模块和 AI/AO 模块统称为信号模块(SM)。

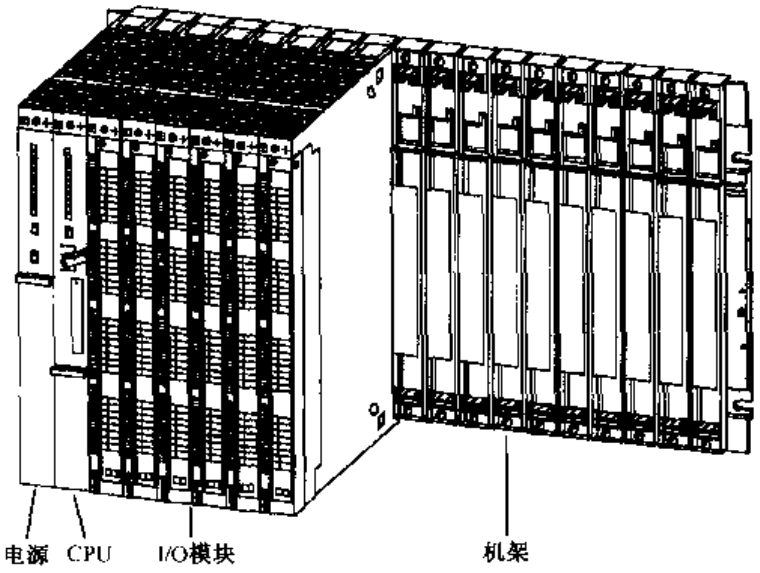


图 2-18 S7-400 模块式 PLC

机架用来固定模块、提供模块工作电压和实现局部接地，并通过信号总线将不同模块连接在一起。

S7-400 的模块插座焊在机架中的总线连接板上，模块插在模块插座上，有不同槽数的机架供用户选用，如果一个机架容纳不下所有的模块，可以增设一个或数个扩展机架，各机架之间用接口模块和通信电缆交换信息，如图 2-19 所示。

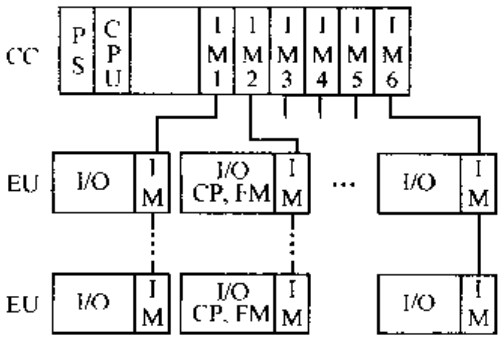


图 2-19　S7-400 的多机架连接

S7-400 提供了多种级别的 CPU 模块和种类齐全的通用功能的模块，使用户能根据需要组合成不同的专用系统。S7-400 采用模块化设计，性能范围宽广的不同模块可以灵活组合，扩展十分方便。

中央机架(或称中央控制器，CC)必须配置 CPU 模块和一个电源模块，可以安装除用于接收的 IM(接口模块)外的所有 S7-400 模块。如果有扩展机架，中央机架和扩展机架都需要安装接口模块。

扩展机架(或称扩展单元，EU)可以安装除 CPU、发送 IM、IM 463-2 适配器外的所有 S7-400 模块。但是电源模块不能与 IM 461-1(接收 IM)一起使用。

集中式扩展方式适用于小型配置或一个控制柜中的系统。CC 和 EU 的最大距离为 1.5 m(带 5 V电源)或 3 m(不带 5 V 电源)。

分布式扩展适用于分布范围广的场合，CC 与最后一个 EU 的最大距离为 100 m(S7 EU)或 600 m(S5 EU)。CC 最多插 6 块发送 IM，最多只有 2 个 IM 可以提供 5 V 电源。通过 C 总线(通信总线)的数据交换仅限于 CC 和 6 个 EU 之间。

用 ET 200 分布式 I/O 可以进行远程扩展，用于分布范围很广的系统。通过 CPU 中的 PROFIBUS-DP 接口，最多连接 125 个总线节点。使用光缆时 CC 和最后一个节点的距离为 23 km。

电源模块应安装在机架的最左边(第 1 槽)，有冗余功能的电源模块是一个例外。中央机架只能插入最多 6 块发送型的接口模块，每个模块有两个接口，每个接口可以连接 4 个扩展机架，最多能连接 21 个扩展机架。中央机架中同时传送电源的发送接口模块(IM 460-1)不能超过两块，IM 460-1 的每个接口只能带一个扩展机架。

扩展机架中的接口模块只能安装在最右边的槽(第 18 槽或第 9 槽)。通信处理器 CP 只能安装在编号不大于 6 的扩展机架中。

2. S7-400 的特点

(1) 运行速度高，CPU 416 执行一条二进制指令只要 0.08 μs。

(2) 存储器容量大，例如 CPU 417-4 的 RAM 可以扩展到 16 MB，装载存储器(FEPROM 或 RAM)可以扩展到 64 MB。

(3) I/O 扩展功能强，可以扩展 21 个机架，S7 417-4 最多可以扩展 262144 个数字量 I/O 点和 16384 个模拟量 I/O。

(4) 有极强的通信能力，容易实现分布式结构和冗余控制系统，集成的 MPI(多点接口)能建立最多 32 个站的简单网络。大多数 CPU 集成有 PROFIBUS-DP 主站接口，可以用来建立高速的分布式系统，使操作大大简化。从用户的角度看，分布式 I/O 的处理与集中式 I/O 没有什么区别，具有相同的配置、寻址和编程方法。CPU 能与在通信总线和 MPI 上的站点建立联系，最多 16～44 个站点，通信速率最高 12 M bit/s。

(5) 通过钥匙开关和口令实现安全保护。

(6) 诊断功能强，最新的故障和中断时间保存在 FIFO(先入先出)缓冲区中。

(7) 集成的 HMI(人机接口)服务，用户只需要为 HMI 服务定义源和目的地址，系统会自动地传送信息。

S7-400 与 S7-300 一样，都用 STEP 7 编程软件编程，编程语言与编程方法完全相同。

2.6.2 机架与接口模块

1. S7-400 的机架

S7-400 的模块是用机架上的总线连接起来的。机架上的 P 总线(I/O 总线)用于 I/O 信号的高速交换和对信号模块数据的高速访问。C 总线(通信总线，或称 K 总线)用于在 C 总线各站之间的高速数据交换，C 和 K 分别是英语单词 Communication 和德语单词 Kommunikation(通信)的缩写。两种总线分开后，控制和通信分别有各自的数据通道，通信任务不会影响控制的快速性。如图 2-20 所示。

(1) 通用机架 UR1/UR2

UR1(18 槽)和 UR2(9 槽)有 P 总线和 K 总线，可以用作中央机架(CC)和扩展机架(EU)。它们用作中央机架时，可以安装除接收 IM 外的所有 S7-400 模块。

(2) 中央机架 CR2/CR3

CR2 是 18 槽的中央机架，P 总线分为两个本地总线段，分别有 10 个插槽和 8 个插槽。两个总线段都可以对 K 总线进行访问。CR2 需要一个电源模块和两个 CPU 模块，每个 CPU 有它自己的 I/O 模块，它们能相互操作和并行运行。

CR3 是 4 槽的中央机架，有 I/O 总线和通信总线。

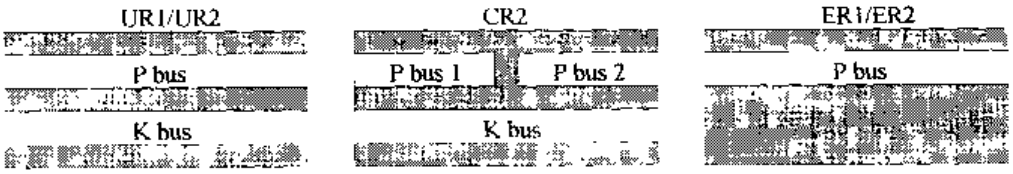


图 2-20 机架与总线

(3) 扩展机架 ER1/ER2

ER1 和 ER2 是扩展机架，分别有 18 槽和 9 槽，只有 I/O 总线，未提供中断线，没有给模块供电的 24 V 电源，可以使用电源模块、接收 IM 模块和信号模块。但是电源模块不能与 IM 461-1 接收 IM 一起使用。

(4) UR2-H 机架

UR2-H 机架用于在一个机架上配置一个完整的 S7-400H 冗余系统，也可以用于配置两个具有电气隔离的独立运行的 S7-400 CPU，每个均有自己的 I/O。UR2-H 需要两个电源模块和两个冗余 CPU 模块。

2. 接口模块

IM 460-x 是用于中央机架 UR1,UR2 和 CR2 的发送接口模块;IM 461-x 是用于扩展机架 UR1,UR2 和 ER1,ER2 的接收接口模块。

(1) IM 460-0 和 IM 461-0 分别是配合使用的发送接口模块和接收接口模块,属于集中式扩展,最大距离 3 m。IM 460-0 有两个接口,每个接口最多扩展 4 个机架,模块最多可扩展 8 个机架,中央机架可以插 6 块 IM 461-3。

(2) IM 460-1 和 IM 461-1 分别是配合使用的发送接口模块和接收接口模块,属于集中式扩展,最大距离 1.5 m。中央控制器通过接口模块给扩展机架提供 5 V 电源(最大 5 A),最多能连接两个扩展机架,每个接口 1 个,中央控制器最多使用两块 IM 460-1,只传输 P 总线。

(3) IM 460-3 和 IM 461-3 分别是配合使用的发送接口模块和接收接口模块,属于分布式扩展,最大距离 100 m,传输 C 总线和 P 总线。IM 460-3 有两个接口,每个接口最多扩展 4 个机架,模块最多扩展 8 个机架,中央机架可以插 6 块 IM 461-3。

(4) IM 460-4 和 M 461-4 分别是发送接口模块和接收接口模块,它们必须配合使用,属于分布式扩展,最大距离 605 m,通过 P 总线传输数据。IM 460-4 有两个接口,每个接口最多扩展 4 个机架,模块最多可扩展 8 个机架,中央机架可以插 6 块 IM 461-4。

(5) IM463-2 是发送接口模块,用于 S5 扩展机架的分布式扩展,最大距离 600 m,有两个接口,最多可扩展 8 个 S5 扩展机架,每个接口最多扩展 4 个机架,只能与 IM 314 配合使用。中央机架最多插 4 块 IM463-2。

(6) IM 467 和 IM 467 FO 将 S7-400 作为主站接入 PROFIBUS-DP 网络,可以将多达 14 条 DP 线连接到 S7-400,IM 467 FO 集成了光纤接口。它们提供 PROFIBUS-DP 通信服务和 PG/OP 通信,以及通过 PROFIBUS-DP 的编程和组态。支持 SYNC/FREEZE、等距离和站点间通信功能。

2.6.3 S7-400 的通信功能

S7-400 有很强的通信功能,CPU 模块集成有 MPI 和 DP 通信接口,有 PROFIBUS-DP 和工业以太网的通信模块,以及点对点通信模块。通过 PROFIBUS-DP 或 AS-i 现场总线,可以周期性地自动交换 I/O 模块的数据(过程映像数据交换)。在自动化系统之间,PLC 与计算机和 HMI(人机接口)站之间,均可以交换数据。数据通信可以周期性地自动进行或基于事件驱动,由用户程序块调用。

S7/C7 通信对象的通信服务通过集成在系统中的功能块来进行。可提供的通信服务有:使用 MPI 的标准 S7 通信;使用 MPI、C 总线、PROFIBUS-DP 和工业以太网的 S7 通信,S7-300 只能作为服务器;与 S5 通信对象和第三方设备的通信,可用非常驻的块来建立。这些服务包括通过 FROFIBUS-DP 和工业以太网的 S5 兼容通信和标准通信(第三方系统)。

S7-400 的通信功能、通信模块、通信的设置与编程的详细情况见 7~9 章。

2.6.4 冗余设计的容错自动化系统 S7-400H

1. S7-400H 的使用场合

在许多生产领域中,要求容错和高度可靠性的应用越来越多,某些领域由于故障引起的停机将会带来重大的经济损失,西门子的高可靠性 S7-400H 容错 PLC 已有成千上万台在实际中

使用，可以满足高度可靠性的要求。S7-400H 特别适合于在下列场合使用：

（1）停机将会造成重大的经济损失；

（2）过程控制系统发生故障后再起动的费用十分昂贵；

（3）某些使用贵重的原材料的过程控制（例如制药工业）会因突发的停机而产生废品；

（4）无人管理的场合或需要减少维修人员的场合。

西门子的 S7 Sofiware Redundancy（软件冗余性）可选软件可以在 S7-300 和 S7-400 标准系统上运行。生产过程出现故障时，在几秒内切换到替代系统，可以用于水厂水处理系统或交通流量控制系统等场合。

S7-400H 是按冗余方式设计的，主要器件都是双重的，可以在事件发生后继续使用备用的器件。设计成双重器件的有中央处理器 CPU、电源模块以及连接两个中央处理器的硬件。用户可以自行决定系统中是否需要更多的双重器件，以增强设备的冗余性。

2. S7-400H 的结构

S7-400H 由两个子系统组成，典型的结构是使用分为两个区（每个区 9 个槽）的机架 UR2H，每个区可以视为一个中央控制器（见图 2-21 中的 Rack 0 和 Rack 1），也可以使用两个独立的中央控制器（即中央机架）UR1/UR2。每个中央控制器有一块有容错功能的 CPU 414-4H 或 CPU 417-4H，一块 PS 407 电源模块。

同步子模块用于连接两个中央处理器，它们放置在中央处理器内部，并由光缆互连。

每个中央控制器上有 S7 I/O 模块，中央控制器也可以有扩展机架或 ET 200M 分布式 I/O。中央功能总是冗余配置的，I/O 模块可以是常规配置、切换型配置或冗余配置。

若要提高供电的冗余能力，每个子系统可以采用冗余供电的方式。在这种情况下需使用 PS 407 10AR 电源模块，其额定电压为 AC 120/230 V，输出电流为 10 A。

SIMATIC S7 系统所有的 I/O 模块都可以在 S7-400H 中使用。I/O 模块可以插入到中央控制器、扩展机架或分布式 I/O 站。I/O 模块可按下列方式配置：

（1）常规单通道单路 I/O 配置

两个子系统中只有一个有一套 I/O 模块（单通道），它们可以在一个中央控制器中，或者是分布式的 I/O 站。I/O 模块只能被该子系统访问，读出的 I/O 信息同时提供给两个中央控制器。如果出现故障，属于故障控制器的 I/O 模块退出运行。

（2）单通道切换式配置

单通道切换式配置的 I/O 模块虽然是单通道设计，但是两个中央控制器都可以通过冗余的 PROFIBUS-DP 网络访问 I/O 模块（见图 2-21）。切换式 I/O 模块只能在 ET-200M 远程 I/O 站中。

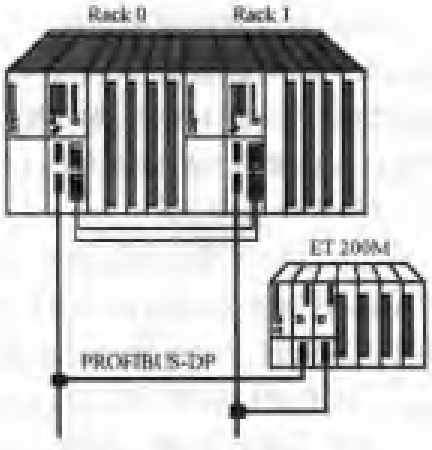


图 2-21　冗余控制系统

（3）双通道 I/O 模块容错冗余配置

系统中有两套相同的容错冗余配置的 I/O 模块，每一个子系统都可以访问这两套 I/O 模块。

（4）FM 和 CP 的冗余

功能模块（FM）和通信处理器（CP）有两种冗余配置方法：

可切换的冗余配置：FM 和 CP 分别插到可切换的 ET 200 中。

双通道冗余配置:FM 和 CP 分别插到两个子单元或两个子单元的扩展设备中。

(5) 通信

S7-400H 可以使用系统总线(例如工业以太网)或点对点通信,从简单的线性网络结构到冗余式双光缆环路。S7 的通信功能完全支持 PROFIBUS 或工业以太网的容错通信。

出现通信故障时,通过最多 4 个冗余连接,使通信继续下去。切换过程不需要用户编程,冗余功能在参数设置时建立,用户的通信程序与标准通信程序一样。S7-400H 和 PC 支持冗余通信,PC 冗余需要有连接程序软件包。由于对冗余的要求不同,网络可以配置为冗余的或非冗余的总线,可以是总线型或环形结构。

3. S7-400H 冗余控制 PLC 的工作原理

CPU417-H 的操作系统自动地执行 S7-400H 需要的附加功能,包括数据通信,故障响应(切换到备用控制器),两个子单元的同步和自检功能等。

S7-400H 采用"热备用"模式的主动冗余原理,在发生故障时,无扰动地自动切换。无故障时两个子单元都处于运行状态,如果发生故障,正常工作的子单元能独立完成整个过程的控制。

为了保证无扰动切换,必须实现中央控制器链路之间的快速、可靠的数据交换。两个控制器必须使用相同的用户程序,自动地接收相同的数据块、过程映像和相同的内部数据,例如定时器、计数器、位存储器等。这样可以确保两个子控制器同步地更新内容,在任意一个子系统有故障时,另一个可以承担全部控制任务。

S7-400H 采用"事件驱动同步",当两个子单元的内部状态不同时,例如在直接 I/O 访问、中断、报警和修改实时钟时,就会进行同步操作。通过通信功能修改数据,由操作系统自动执行同步功能,不需要用户编程。

S7-400H 对中央控制器之间的链接、CPU 模块、处理器/ASIC 和存储器进行自检。再起动后每个子单元完整地执行所有的测试功能。自检功能被分为几部分,每个周期只执行部分自检功能,以减轻 CPU 的负担。

4. S7-400H 冗余控制 PLC 的编程与组态

容错式连接只需要进行组态,不需要其他专门的编程工作。从用户程序的观点看,S7-400H 的作用几乎和标准系统一样。运行容错功能所需的通信功能和同步功能都已经集成在容错 CPU 的操作系统中,通信连接的监视以及发生故障事件时的自动切换在后台自动运行。在用户程序中完全没有必要考虑这些功能。

S7-400H 用 STEP 7 进行组态和编程,完成配置后可以把 S7-400H 看成一般的 S7-400 系统。冗余单元的工作由操作系统来监视,出现故障后可以独立地执行切换工作,用 STEP 7 组态时已经将所需信息组态进去,并通知系统。

组态和编程需要可选的 H 软件包,能在 S7-400 系统上使用的所有的标准软件工具、工程用软件工具和运行软件工具都可以在 S7-400H 上使用。

适合标准 S7-400 系统设计和编程的规则同样适用于 S7-400H,用户程序以相同的形式存储在两个中央处理器中,并且被同时执行。

除了那些既可以在 S7-400 上使用也可以在 S7-400H 上使用的功能块外,S7-400H 系列还提供了一些与冗余功能有关的组织块,例如 OB70(I/O 冗余故障)和 OB72(CPU 冗余故障)。使用系统功能 SFC 90"H_CTRL",用户可以禁用或重新启用容错 CPU 的链接和刷新。

2.6.5 安全型自动化系统 S7-400F/FH

1. S7-400F/FH 的应用场合

S7-400F/FH 安全型自动化系统(见图 2-22)适用于对安全性要求很高的工厂,控制过程(直接关闭某些输出)不会对人和环境产生危害。S7-400F/FH 有两种基本类型:

(1) S7-400F

安全型自动化系统,如果在系统中出现故障,生产过程转为安全状态,并执行中断。

(2) S7-400FH

安全及容错自动化系统,如果系统出现故障,冗余控制使生产过程能继续执行。

S7-400F/FH 可以使用标准模块和安全型模块,配置一个安全型集成控制系统,在无安全要求及有部分安全要求的工厂中使用,整个工厂可以用相同的标准工具软件来配置和编程。

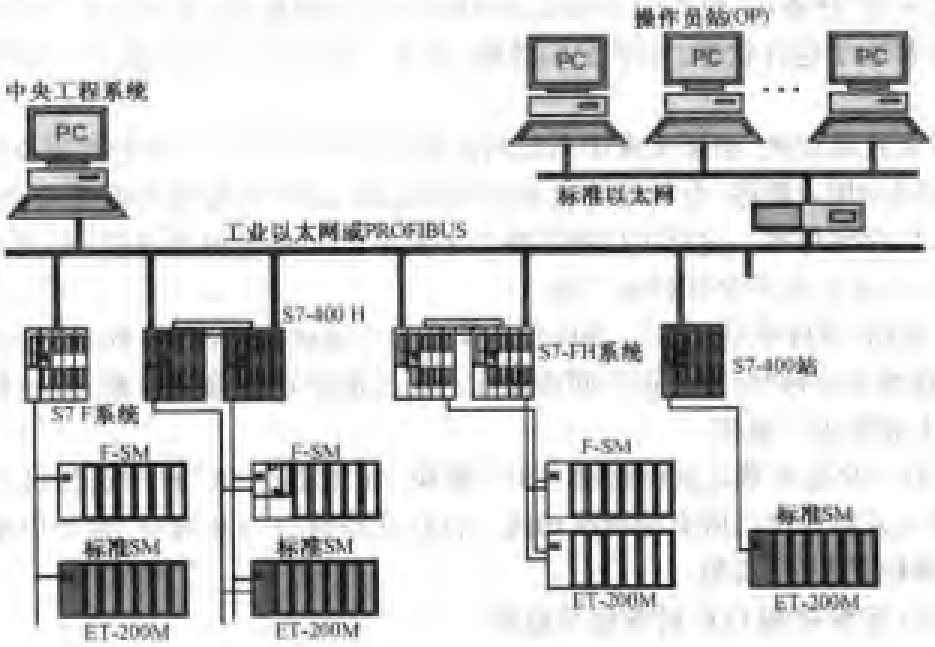


图 2-22 安全型自动化系统

2. S7-400F/FH 的工作原理

S7-400F/FH 的安全性功能包含在 CPU 的 F 程序和安全型信号模块(F-SM)中。信号模块利用偏差分析和注入测试信号的方法,来监视输入信号和输出信号。

CPU 通过常规自检、结构检查,以及逻辑和顺序程序流程控制来检查 PLC 的有关操作和 I/O 模块的功能,如果发现故障,I/O 转为安全状态。

必须将 F 运行许可证安装到 S7-400F/FH 的 CPU147-4H 上,每个 CPU 需要一个许可证。

3. S7-400F/FH 的编程

S7-400F/FH 的编程方法与其他 S7 系统的编程方法相同,无安全性要求的部分可以用 STEP 7 来编写。编写有安全性要求的程序时需要"S7 F 系统"可选软件包,软件包包括用于生成 F 程序的所有功能和功能块。在计算机上应安装下列软件:

STEP 7 V5.1 或更高的版本;

CFC V5.0 与 Service Pack 3 或更高的版本;

S7 SCL V5.0 或更高的版本；

S7 F V5.1(对于 S7-400FH 为可选项)。

在带 CFC 的 F 库中调用特殊功能块，并从内部连接到含有安全功能的 F 程序中。CFC 的使用简化了设备的配置和编程工作，编程者可以将精力全部集中到安全性要求的应用问题。

4. S7-400F/FH 的通信

中央控制器和 ET 200M 之间的安全型通信和标准通信通过 FROFIBUS－DP 进行，由于 PRFISafe PROFIBUS 规范的发展，允许安全型功能的数据和标准报文帧一起传送。

5. S7-400F/FH 的结构

(1) 单通道单路 I/O 配置

设备需要带安全性保护的 PLC 来控制，不必是容错性的。配置如下：1 个带 F 运行许可证的 CPU417-4H，1 条 PROFIBUS-DP 通信线，带 IM153-2FO 的 ET 200M，和无冗余设计的安全型信息模块。出现故障时 I/O 停止工作，安全型信号模块被关闭。

(2) 单通道切换式 I/O 配置

设备需要带安全性保护的 PLC 来控制，CPU 一侧采用容错技术，配置如下：两个带 F 运行许可证的 CPU417-4H，两条 PROFIBUS-DP 通信线，两个有两块 IM153-2FO(冗余)的 ET 200M，和无冗余设计的安全型信息模块。如果其中的 1 个 CPU417-4H，1 条 PROFIBUS-DP 通信线或 1 块 IM153-2FO 出现故障，系统还能继续工作。如果安全型信号模块或 ET 200M 出现故障，I/O 停止工作，安全型模块被关闭。

(3) S7-400FH 冗余切换式 I/O 配置

设备需要带安全性保护的 PLC 来控制，CPU 一侧采用容错技术，配置如下：两个带 F 运行许可证的 CPU417-4H，两条 PROFIBUS-DP 通信线，两个有两块 IM153-2FO(冗余)的 ET 200M，和冗余设计的安全型信息模块。如果其中的 1 个 CPU，1 条 PROFIBUS-DP 通信线，1 套安全型信号模块或 1 个 ET 200M 出现故障，系统还能继续工作。在 S7-400F/FH 自动化系统中可以使用标准模块，但是它们不能与 ET 200M 一起使用。

2.6.6 多 CPU 处理

多 CPU 处理运行是指在 S7-400 中央机架上，最多 4 个具有多 CPU 处理能力的 CPU 同时运行。这些 CPU 自动地、同步地变换其运行模式。也就是说它们同时启动，同时进入 STOP 模式，这样可以同步地执行控制任务。

多 CPU 处理适用于以下情况：

对于一个 CPU 来说用户程序太长，或者存储空间不够，需要将程序分配给多个 CPU 执行。如果整个系统由多个不同的部分组成，并且这些部分可以很容易地彼此拆开并可以单独控制，则各 CPU 分别处理不同的部分，每个 CPU 访问分配给它的模块。

通过通信总线，CPU 彼此互连。如果组态正确，通过编程软件可以访问 MPI 网络上的全部 CPU。

在启动时，多 CPU 运行的 CPU 将自动检查彼此间是否能同步。只有满足下列条件，才能同步：组态的所有 CPU 必须插好；已创建了正确的组态数据(SDB)，并已下载到已插入的所有 CPU 中。如果有一条不满足，在诊断缓冲区中将出现错误信息。

退出 STOP 模式时，将比较 RESTART/REBOOT 启动类型。如果启动类型不同，CPU 将

不会进入 RUN 模式。

在多 CPU 处理运行时，每个 CPU 可以访问用 STEP 7 为其组态分配的模块，模块的地址区总是单独分配给一个 CPU。每个具有中断能力的模块被分配给一个 CPU，这样的模块产生的中断不能被其他 CPU 接收。

过程中断和诊断中断只能发送给一个 CPU，在模块有故障或插/拔某一模块时，通过在 STEP 7 参数赋值时分配的 CPU 处理中断。有机架故障时，每个 CPU 调用 OB 86。

使用多 CPU 中断(OB 60)可以在相应的 CPU 中同步地响应一个事件。与通过模块触发过程中断相比，通过调用 SFC 35 "MP _ ALM"触发的多 CPU 中断只能通过 CPU 输出。

分段的机架 CR2 属于物理分段，不是通过参数赋值分段，每段只能有一个 CPU，它不是多 CPU 处理，每个分段的机架上的 CPU 构成一个独立的子系统，它们没有共享的逻辑地址区，多 CPU 处理不能在分段的机架上运行。

2.6.7 CPU 模块的元件

S7-400 有 7 种不同型号的 CPU，分别适用于不同等级的控制要求。不同型号的 CPU 面板上的元件不完全相同，如图 2-23 所示。

CPU 内的元件封装在一个牢固而紧凑的塑料机壳内，面板上有状态和故障指示 LED、方式选择钥匙开关和通信接口。大多数 CPU 还有后备电池盒，存储器插槽可插入多达数兆字节的存储器卡。

1. S7-400 CPU 的指示灯与模式选择开关

S7-400 CPU 模块面板上的 LED 指示灯的功能见表 2-15，有的 CPU 只有部分指示灯。

表 2-15 CPU 的指示灯的功能

指 示 灯	颜 色	说 明
INTF	红色	内部故障，例如用户程序运行超时
EXTF	红色	外部故障，例如电源故障，模块故障
FRCE	黄色	有输入/输出处于被强制的状态
RUN	绿色	运行模式
STOP	黄色	停止模式
BUS1F	红色	MPI/PROFIBUS-DP 接口 1 的总线故障
BUS2F	红色	MPI/PROFIBUS-DP 接口 2 的总线故障
MSTR	黄色	CPU 处理 I/O，仅用于 CPU 41x-4H
REDF	红色	冗余错误，仅用于 CPU 41x-4H
RACK0	黄色	CPU 在机架 0 中，仅用于 CPU 41x-4H
RACK1	黄色	CPU 在机架 1 中，仅用于 CPU 41x-4H
IFM1F	红色	接口子模块 1 故障
IFM2F	红色	接口子模块 2 故障

S7-400 CPU 模块面板上的模式选择开关是一种钥匙开关，其外形和使用方法与 S7-300 的完全相同。钥匙开关各位置的意义与 S7-300 的相同。

当模式选择开关从 STOP 扳到 RUN 位置时，如果 CPU 模块上的类型选择开关在 CRST 位置，执行全启动(冷启动)；如果类型选择开关在 WRST 位置，执行一个热起动。

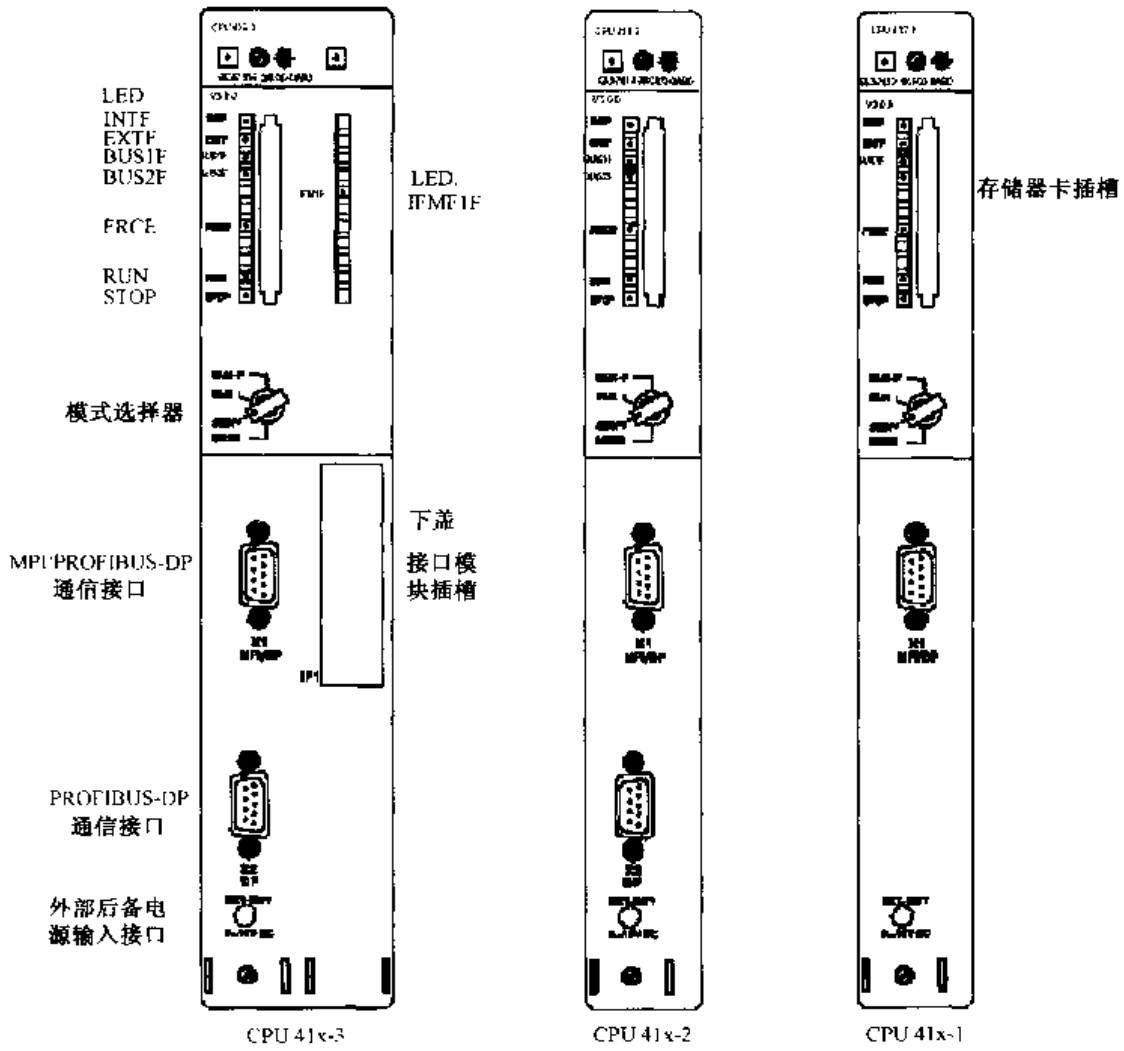


图 2-23　S7-400 的 CPU 模块

2. 存储器卡插槽

CPU 417 的工作存储器可以扩展，在 CPU 模块的存储器卡插槽内插入 RAM 存储卡，可以增加装载存储器的程序容量。

Flash EPROM(快闪存储器)卡用来存储程序和数据，即使在没有后备电池的情况下，其内容也不会丢失。可以在编程器或 CPU 上编写 Flash 卡的内容，Flash 卡也可以扩展 CPU 装载存储区的容量。CPU 417-4 和 CPU 417-4H 还有存储器扩展接口，可以扩展工作存储器。

集成式 RAM 不能扩展，集成装载存储器为 256KB(RAM)，用存储器卡扩展 FEPROM 和 RAM 最大各 64MB。电池可以对所有的数据提供后备电源。

3. CPU 的通信接口

CPU 模块上有集成的 MPI 和 PROFIBUS-DP 接口。MPI 可以连接计算机、操作员面板和其他 S7-400 或 300 控制器。也可以将 MPI 接口组态为 PROFIBUS-DP 主站接口，最多可连接 32 个 DP 从站。PROFIBUS-DP 接口可连接分布式 I/O、PG(编程器)/OP(操作员面板)和其他 DP 主站。

可以将接口模块插入 CPU 41x-3 及 CPU 41x-4 的接口模块插槽中，也可以将 H-SYNC 模块插入 CPU 414-4H 和 CPU 417-4H 的接口模块插槽中。

4. 后备电源

根据模块类型的不同，在 S7-400 的电源模块中可以使用一个或两个后备电池，为存储在内置的装载存储器和外部装载存储器、工作存储器的 RAM 中的用户程序和内部时钟提供后备电源，保持存储器中的存储器位、定时器、计数器、系统数据和数据块中的变量。

也可以通过“EXT.-BATT”(外接电池)插口提供 DC 5～15 V 外部后备电源，插口的直径为 2.5 mm，“EXT.-BATT.”具有反极性保护。在更换电源模块时，如果想保存存储在 RAM 中的用户程序和数据，需要通过“EXT-BATT”插座提供外部后备电源。

2.6.8 CPU 模块与电源模块的技术规范

1. CPU 模块概述

S7-400 有 7 种 CPU，此外 S7-400H 还有两种 CPU。CPU 412-1 和 CPU 412-2 用于中等性能的经济型中小型项目，集成的 MPI 接口允许 PROFIBUS-DP 总线操作。CPU 412-2 有两个 PROFIBUS-DP 接口。

CPU 414-2 和 CPU 414-3 具有中等性能，适用于对程序规模、指令处理速度及通信要求较高的场合。

CPU 417-4DP 适用于最高性能要求的复杂场合，有两个插槽供 IF 接口模块(串口)使用。CPU 417H 用于 S7-400H 容错控制 PLC。

通过 IF 964DP 接口子模块，CPU 414-3 和 CPU 416-3 可以扩展一个 PROFIBUS-DP 接口，CPU 417-4 可以扩展两个 PROFIBUS-DP 接口。

除了 CPU 412-1 之外，集成的 DP 接口使 CPU 可作 PROFIBUS-DP 的主站。

2. S7-400 CPU 模块的共同特性

下面是 S7-400 CPU 模块的一些共同的特性。

(1) S7-400 有 1 个中央机架，可扩展 21 个扩展机架。使用 UR1 或 UR2 机架的多 CPU 处理最多安装 4 个 CPU。每个中央机架最多使用 6 个 IM(接口模块)，通过适配器在中央机架上可以连接 6 块 S5 模块。

(2) 实时钟功能：CPU 有后备时钟和 8 个小时计数器，8 个时钟存储器位，有日期时间同步功能，同步时在 PLC 内和 MPI 上可以作为主站和从站。

(3) S7-400 都有 IEC 定时器/计数器(SFB 类型)，每一优先级嵌套深度 24 级，在错误 OB 中附加 2 级。S7 信令功能可以处理诊断报文。

(4) 测试功能：可以测试 I/O，位操作，DB(数据块)，分布式 I/O，定时器和计数器；可以强制 I/O，位操作和分布式 I/O。有状态块和单步执行功能，调试程序时可以设置断点。

(5) FM(功能模块)和 CP(通信处理器)的块数只受槽的数量和通信的连接数量的限制。S7-400 可以与编程器和 OP(操作员面板)通信，有全局数据通信功能。在 S7 通信中，可以作服务器和客户机，分别为 PG(编程器)和 OP 保留了一个连接。

(6) CPU 模块内置的第一个通信接口的功能

第一个通信接口可以作 MPI(默认设置)和 DP 主站，有光隔离。

作 MPI 接口时，可以与编程器和 OP 通信，可以作路由器。全局数据通信的 GD 包最大 64 KB。S7 标准通信每个作业的用户数据最大 76 B，S7 通信每个作业的用户数据最大 64 KB，S5 兼容通信每个作业的用户数据最大 8 KB，通过 CP 和可装载的 FC 可以进行标准通信。内

置的各通信接口最大传输速率为 12 M bit/s。

作DP 主站时，可以与编程器和 OP 通信，支持内部节点通信，有等时线和 SYNC/FREEZE 功能，除 S7-412 外，有全局数据通信、S7 基本通信和 S7 通信功能。最多 32 个 DP 从站，可以作路由器，插槽数最多 512 个。最大地址区 2KB，每个 DP 从站的最大可用数据为 244 B 输入/244 B 输出。

(7) CPU 模块内置的第二个通信接口的功能

第二个通信接口可以作 DP 主站(默认设置)和点对点连接，有光隔离。作 DP 主站时，可以与编程器和 OP 通信，支持内部节点通信，有等时线和 SYNC/FREEZE 功能。每个 DP 从站的最大可用数据为 244 B 输入/244 B 输出。

3. 新一代 S7-400 CPU 模块的技术规范

2004 年发布的新一代 S7-400 CPU 模块的技术参数如表 2-16 和 2-17 所示。

表 2-16 CPU 412 与 CPU 414 的技术参数

型 号	CPU 412-1	CPU 412-2	CPU 414-2	CPU 414-3
集成式 RAM 用于程序/用于数据	72 KB/72 KB	128 KB/128 KB	256 KB/256 KB	700 KB/700 KB
最小位操作指令执行时间 浮点数加法指令执行时间	0.1 μs 0.3 μs	0.1 μs 0.3 μs	0.06 μs 0.18 μs	0.06 μs 0.18 μs
位存储器	4 KB	4 KB	8 KB	8 KB
定时器/计数器	2048/2048	2048/2048	2048/2048	2048/2048
OB 最大容量 FB 最大块数/最大容量 FC 最大块数/最大容量 DB 最大块数(DB0 保留)	受工作存储器限制 256/受工作存储器限制 256/受工作存储器限制 512/受工作存储器限制	64 KB 256/64 KB 256/64 KB 512/64 KB	64 KB 2048/64 KB 2048/64 KB 4095/64 KB	64 KB 2048/64 KB 2048/64 KB 4095/64 KB
最大局部数据字节数	8 KB	8 KB	16 KB	16 KB
看门狗中断/过程报警 日期时间中断/延时中断	2/2 2/2	2/2 2/2	4/4 4/4	4/4 4/4
最大 I/O 地址区 最大分布式 I/O 地址区 MPI/DP 接口最大 I/O 地址区 第 1 个 DP 接口模块最大 I/O 地址区 第 2 个 DP 接口模块最大 I/O 地址区	4 KB/4 KB 2 KB/2 KB — — —	4 KB/4 KB 2 KB/2 KB — — —	8 KB/8 KB — 2 KB/2 KB 6 KB/6 KB —	8 KB/8 KB — 2 KB/2 KB 6 KB/6 KB 6 KB/6 KB
I/O 过程映像(可修改) 默认值 子过程最大映像数	4 KB/4 KB 128 B/128 B 15	4 KB/4 KB 128 B/128 B 15	8 KB/8 KB 256 B/256 B 15	8 KB/8 KB 256 B/256 B 15
最大模拟量 I/O 通道 中央区最大模拟量 I/O 通道	2048/2048 2048/2048	2048/2048 2048/2048	4096/4096 4096/4096	4096/4096 4096/4096
集成式 DP 个数/接口子模块主站个数 使用 IM 467 块数/CP 的 DP 主站数量	1/— 4/10	2/— 4/10	2/— 4/10	2/1 4/10
S7 报文功能：报文功能所登录的站数	8	8	8	8
诊断缓冲区条目数(可设置)	200	400	400	3200
连接总数/带报文处理连接数	16/8	16/8	32/8	32/8
第一个通信接口				

（续）

型　　号	CPU 412-1	CPU 412-2	CPU 414-2	CPU 414-3
MPI 连接数	16	16	16	16
发送和接收方最大 GD 包数量	8/16	8/16	8/16	8/16
DP 主站连接数量	16	16	16	16
第二个通信接口				
DP 主站连接数	—	16	32	32
DP 从站个数	—	64	96	96
最大槽数/地址区	—	1024/2 KB	1536/6 KB	1536/6 KB
第三个通信接口	—	—	—	技术数据见第二接口
适合的接口子模块	—	—	—	IF964-DP 作 DP 主站

表 2-17　CPU 416 与 CPU 417 的技术参数

型　　号	CPU 416-2	CPU 416-3	CPU 417-4
集成工作存储器：用于程序/用于数据	1.4 GB/1.4 GB	2.8 MB/2.8 MB	10 MB/10 MB
集成装载存储器/Flash 存储卡/RAM 存储卡	64 MB/64 MB/64 MB	64 MB/64 MB/64 MB	64 MB/64 MB/64 MB
最小位操作指令执行时间	0.04 μs	0.04 μs	0.03 μs
浮点数加法指令执行时间	0.12 μs	0.12 μs	0.09 μs
位存储器	16 KB	16 KB	16 KB
定时器/计数器	2048/2048	2048/2048	2048/2048
FB 最大块数/最大容量	2048/64 KB	2048/64 KB	6144/64 KB
FC 最大块数/最大容量	2048/64 KB	2048/64 KB	2048/64 KB
DB 最大块数(DB0 保留)	4096/64 KB	4096/64 KB	8192/64 KB
局部最大数据	32 KB	32 KB	32 KB
看门狗中断/过程报警	9/8	9/8	9/8
日期时间中断/延时中断	8/4	8/4	8/4
最大 I/O 地址区	16 KB/16 KB	16 KB/16 KB	16 KB/16 KB
最大分布式 I/O 地址区	2 KB/2 KB	2 KB/2 KB	2 KB/2 KB
MPI/DP 接口最大 I/O 地址区	8 KB/8 KB	8 KB/8 KB	8 KB/8 KB
第 1 个 DP 接口模块最大 I/O 地址区	—	8 KB/8 KB	8 KB/8 KB
第 2 个 DP 接口模块最大 I/O 地址区	—	—	8 KB/8 KB
I/O 过程映像(可修改)	16 KB/16 KB	16 KB/16 KB	16 KB/16 KB
默认值	512 B/512 B	512 B/512 B	1024 B/1024 B
子过程最大映像数	15	15	15
最大数字量通道	131072/131072	131072/131072	131072/131072
中央区最大数字量通道	131072/131072	131072/131072	131072/131072
最大模拟量 I/O 通道	8192/8192	8192/8192	8192/8192
中央区最大模拟量 I/O 通道	8192/8192	8192/8192	8192/8192
集成式 DP 主站数量/使用接口子模块	2/—	2/1	2/2
使用 IM467/使用 CP 的 DP 主站数量	4/10	4/10	4/10
S7 报文功能：报文功能所登录的站数	12	12	16
诊断缓冲区条目数(可设置)	3200	3200	3200
连接总数/带报文处理连接数	64/12	64/12	64/16

（续）

型　　号	CPU 416-2	CPU 416-3	CPU 417-4
第一个通信接口			
MPI 连接数	44	44	44
发送和接收方最大 GD 包数量	16/32	16/32	16/32
GD 包大小/DP 主站连接数量	64 B/32	64 B/32	64 B/32
最大 DP 从站数/最大插槽数/最大地址区	32/512/2 KB	32/512/2 KB	32/512/2 KB
第二个通信接口			
最大 DP 从站数/最大插槽数	96/1536	125/2048	125/2048
最大地址区	6 KB	8 KB	8 KB
第三个通信接口	技术数据见第二个接口	技术数据见第二个接口	技术数据见第二个接口
适合的接口子模块	—	IF964-DP 模块作 DP 主站	IF964-DP 模块作 DP 主站
第四个通信接口	—	—	技术数据见第二个接口
适合的接口子模块	—	—	IF964-DP 模块作 DP 主站

4. S7-400H/F/FH 的 CPU

CPU 414-4H 和 CPU 417-4H 是 S7-400H 容错式自动控制系统和 S7-400F/FH 安全型自动控制系统的 CPU 模块，带有一个 F 运行授权，可以编译和运行 F 用户程序。它们带有两个用于安装同步模块的插槽。

CPU 414-4H 和 CPU 417-4H 的技术特性与 CPU 414-4 和 CPU 417-4 比较接近。主要区别如下：CPU 417-4H 用于程序和用于数据的可扩充 RAM 分别为 10 MB；CPU 414-4H 和 CPU 417-4H 有两个集成的 DP 主站接口，不能使用接口子模块和 IM 467，通过 CP 可以扩展 10 个 DP 主站。第一个通信接口的 MPI 连接数为 44 个，没有全局数据通信和 S7 全局通信功能。第二个通信接口不支持内部节点通信、等时线和 SYNC/FREEZE 功能。

集成的 MPI 的最大通信速率为 187.5 k bit/s，最多 32 个站，可以建立 64 个连接。诊断缓冲区可以保存 120 个故障和中断时间。

5. 电源模块

电源模块通过背板总线向 S7-400 提供 DC 5V 和 DC 24V 电源。输出电流额定值有 4 A、10 A 和 20 A。PS 405 的输入为直流电压，PS 407 的输入为直流电压或交流电压，S7-400 有带冗余功能的电源模块。如果没有使用传送 5 V 电源的接口模块，每个扩展机架都需要一块电源模块。

6. 电源模块的 LED 指示灯和开关

LED“INTF”：内部故障；

LED “BAF”：电池故障，背板总线上的电池电压过低；

LED “BATT1F”和“BATT2F”：电池 1 或电池 2 接反、电压不足或电池不存在；

LED “DC5V”和“DC24V”：相应的直流电源电压正常时亮；

FMR 开关：故障解除后用于确认和复位故障信息的开关；

ON/Off 保持开关：通过控制电路把输出的 DC 24 V/5 V 电压切断，LED 熄灭。在进线电压没有切断时，电源处于待机模式。

2.6.9 输入/输出模块

1. 模块的技术参数(见表 2-18～表 2-20)

表 2-18 SM421 数字量输入模块技术参数

6ES7 421-	7BH00-0AB0	1BL01-0AA0	1EL00-0AA0	1FH20-0AA0	7DH00-0AB0	5EH00-0AA0
输入点数	16	32	32	16	16	16
中断	过程中断诊断中断	—	—	—	过程中断诊断中断	—
诊断	内部/外部故障	—	—	—	内部/外部故障	—
额定输入电压	DC 24 V	DC 24 V	AC/DC 120 V	AC/DC 120 V/230 V	AC/DC 24～60 V	AC 120 V
频率	—	—	47～63 Hz	47～63 Hz	47～63 Hz	47～63 Hz
隔离,分组数	有隔离,8 组	有隔离,32 组	有隔离,8 组	有隔离,4 组	有隔离,1 组	有隔离,1 组
输入电流(mA)	6～8	7	2～5	14,230 V AC	4～10	6～20
输入延迟时间(ms)	0.1/0.5/3 可组态	3	10/20	25	0.5/3/10/20 可组态	2～15
允许最大静态电流	3 mA	1.5 mA	1 mA	5 mA	2 mA	4 mA

无屏蔽电缆最大长度 600 m,有屏蔽电缆最大长度 1000 m。表中的"允许最大静态电流"是接收接近开关输出的"0"信号时允许的最大电流。如果实际电流超出允许值,将会出现错误的输入信号。

表 2-19 SM422 数字量输出模块技术参数

6ES7 422-	1FH00-0AA0	1HH00-0AB0	5EH00 0AB0	1BH11-0AA0	5EH10-0AB0	1BL00-0AA0	7BL00-0AB0
输出点数	16	16,继电器型	16	16	16,诊断中断	32	32
诊断	—	—	—	—	内部外部故障		内部外部故障
额定负载电压	AC 120/230V	AC 230 V/ DC 60 V	AC 20～120 V	DC 24 V	DC 20～125 V	DC 24 V	DC 24 V
分组数,隔离	4,有隔离	2,有隔离	1,有隔离	8,有隔离	8,有隔离	32,有隔离	8,有隔离
最大输出电流 总输出电流(60℃) 最大灯负载	2 A 2A,4 个相邻 25 W	5 A — —	2 A 7 A —	2 A 2A,2 个相邻 10 W	1.5 A 8 A 8 W	0.5 A 2A,2 个相邻 5 W	0.5 A 2 A/组 5 W
阻性负载最大输出频率 感性负载最大输出频率	10 Hz 0.5 Hz	100 Hz 0.5 Hz	— —	100 Hz 0.1 Hz	100 Hz 0.1 Hz	100 Hz 0.5 Hz	100 Hz 2 Hz
短路保护	熔断器	—	—	电子式	电子式	电子式	电子式

表 2-20 SM431 模拟量输入模块技术参数

6ES7 431	0HH00-0AB0	1KF00-0AB0	1KF10-0AB0	1KF20-0AB0	7QH00-0AB0	7KF00-0AB0	7KF10-0AB0
输入点数 用于电阻测量	16 —	8 4	8 4	8 4	16 8	8 --	8 8
极限值中断 诊断中断	— —	— —	— —	— —	可组态 可组态	可以 可以	可以 可以

（续）

6ES7 431	0HH00-0AB0	1KF00-0AB0	1KF10-0AB0	1KF20-0AB0	7QH00-0AB0	7KF00-0AB0	7KF10-0AB0
额定输入电压 反极性保护	DC 24 V 有	--- ---	DC 24 V 有	DC 24 V 有	DC 24 V 有	— —	— —
输入量程/ 输入阻抗	±1V/10MΩ ±10V/100kΩ 1～5V/100kΩ ±20mA/50Ω 4～20mA/50Ω	±1V/200kΩ ±10V/100kΩ 1～5V/200kΩ ±20mA/80Ω 4～20mA/80Ω 0～600	±80mV/1MΩ ±250mV/1MΩ ±500mV/1MΩ ±1V/1MΩ ±2.5V/1MΩ ±5V/1MΩ 1～5V/MΩ ±10V/1MΩ 0～20mA/50Ω 4～20mA/50Ω 0～48 0～150 0～300 0～600 0～6000 热电偶 B,R,S,T,E,J,K,N,U,L Pt100,Pt200 Pt500,Pt1000 Ni100,Ni1000	±1V/10MΩ ±10V/10MΩ 1～5V/10MΩ ±5V/10MΩ ±20mA/50Ω 4～20mA/50Ω 0～600	±25mV/1MΩ ±50mV/1MΩ ±80mV/1MΩ ±250mV/1MΩ ±500mV/1MΩ ±1V/1MΩ ±2.5V/1MΩ ±5V/1MΩ 1～5V/1MΩ ±10V/100kΩ 0～20mA/50Ω ±5mA/50Ω ±10mA/50Ω ±20mA/50Ω 4～20mA/50Ω 0～48,0～150Ω 0～300 0～600 0～6000 热电偶 B,R,S,T,E,J,K,N,U,L Pt100,Pt200 Pt500,Pt1000 Ni100,Ni1000	±25mV/1MΩ ±50mV/2MΩ ±80mV/2MΩ ±100mV/2MΩ ±250mV/2MΩ ±500mV/2MΩ ±1V/2MΩ ±2.5V/2MΩ ±5V/2MΩ ±10V/2MΩ 1～5V/2MΩ ±5mA/50Ω ±10mA/50Ω ±20mA/50Ω ±3.2mA/50Ω 0 20mA/50Ω 4～20mA/50Ω 热电偶 B,R,S,T,E,J,K,N,U,L	Pt100,Pt200 Pt500, Pt1000 Ni100, Ni1000
2线电流变送器 4线电流变送器	可以 可以	带外部变送器 可以	可以 可以	可以 可以	可以 可以	— 可以	— —
内部/外部隔离 通过通道隔离	无 无	有 无	有 无	有 无	有 无	有 有	有 无
基本转换时间	55ms,56ms	23ms,25ms	20.1ms/ 23.5ms	52μs	6ms/21.1ms/ 23.5ms	—	—
分辨率	12位+符号位/13位	13位	14位	14位	16位	15位+符号位/16位	15位+符号位/16位
干扰抑制频率	60/50Hz	60/50Hz	60/50Hz	400/60/50Hz	400/60/50Hz	400/60/50Hz	60/50Hz
运行误差极限 对应输入范围	±0.65% 1.0% (1～5V)	±1.25%	±0.5%	±0.9%	±0.4%	根据需要	±1℃
基本误差,25℃ 对应输入范围	±0.25% 0.5% (1～5V)	±0.8%	±0.3%	±0.75%,	±0.3%,	根据需要	±2℃

模拟量输出模块 SM 432 只有一个型号，订货号为 6ES7 432-1HF00-0AB0。输出点数为 8 点，额定负载电压 DC 24 V，输出电压范围为±10 V，0～10 V 和 1～5 V；输出电流范围为±20 mA，0～20 mA 和 4～20 mA。电压输出的最小负载阻抗为 1 kΩ，有短路保护，短路电流 28 mA；电流输出的最大阻抗 500 Ω，开路电压最大 18 V。在模拟量部分、总线和屏蔽之间有隔离，每通道最大转换时间为 420 μs。运行误差极限（0～60℃，对应输出范围）为±0.5%（电压）和±1%（电流）。基本误差（25℃，对应输出范围）为±0.2%（电压）和±0.3%（电流）。

2. 信号模块的地址

S7-400 的信号模块地址是在 STEP 7 中用硬件组态工具将模块配置到机架时自动生成

的，根据同类模块所在的机架号和在机架中的插槽号按从小到大的顺序自动连续分配地址，用户可以修改模块的起始地址。每个 8 点、16 点和 32 点数字量模块分别占用 1 个、2 个和 4 个字节地址。假设 32 点数字量输入模块各输入点的地址为 I44.0～I47.7，模块内各点的地址从上到下顺序排列。其中 I44.0 对应的接线端子在最上面，I47.7 对应的接线端子在最下面。

S7-400 的模拟量模块默认的起始地址从 512 开始，每个模拟量输入/输出占 2 B(1 个字)，同类模块的地址按顺序连续排列。模块内最上面的通道使用模块的起始地址。例如某 8 通道模拟量输出模块的起始地址为 832，从上到下各通道的地址分别为 QW832，QW834，……，QW846。表 2-21 给出了信号模块默认地址的例子。

表 2-21　模块地址举例

0 号 机 架			1 号 机 架		
槽　　号	模 块 种 类	地　　址	槽　　号	模 块 种 类	地　　址
1	PS 417 10A 电源模块		1	32 点 DI	IB4～IB7
2			2	16 点 DO	QB2，QB3
3	CPU 412-2DP		3	16 点 DO	QB4，QB5
4	16 点 DO	QB0，QB1	4	8 点 AO	QW528～QW542
5	16 点 DI	IB0，IB1	5	8 点 AI	IW544～IW558
6	8 点 AO	QW512～QW526	6	16 点 DO	QB6，QB7
7	16 点 AI	IW512～IW542	7	8 点 AI	IW560～IW574
8	16 点 DI	IB2，IB3	8	32 点 DI	IB8～IB11
9	IM460-1	4093	9	IM461-0	4092

3. 模块的诊断与硬件中断功能

S7-400 有部分输入/输出模块具有对信号进行监视(诊断)和对过程信号进行监视(过程中断)的智能功能。这些功能与 S7-300 的相同(见 2.2.5)。

2.6.10　功能模块

S7-400 有许多功能模块的技术规范与 S7-300 的几乎完全相同，或者差别很小，模块编号最低两位也相同，例如 FM351 和 FM451，这类模块的对应关系如表 2-22 所示。

表 2-22　S7-300 与 S7-400 性能比较接近的功能模块

功 能 模 块	S7-300 系列	S7-400 系列
计数器模块	FM 350-1	FM 450-1
定位模块	FM 351，双通道	FM 451，3 通道
定位模块	FM 353，双通道	FM 453，3 通道
电子凸轮控制器	FM 352，13 个数字量输出	FM 452，16 个数字量输出
闭环控制模块	FM 355，4 通道	FM 455，16 通道

1. FM 453 定位模块

FM 453 可以控制 3 个独立的伺服电动机或步进电动机，以高时钟频率控制机械运动，用

于简单的点到点定位到对响应、精度和速度有极高要求的复杂运动控制。从增量式或绝对式编码器输入位置信号，步进电动机作执行器时可以不用编码器。控制伺服电动机时输出－10～＋10V模拟信号，控制步进电动机时输出的是脉冲和方向信号。每个通道有 6 点数字量输入，4 点数字量输出。

FM 453 有使用按钮的点动模式和增量模式，有手动数据输入功能，自动/单段控制用于运行复杂的定位路径。FM453 具有下列特殊功能：长度测量、变化率限制、运行中设置实际值、通过高速输入使定位运动起动或停止。

2. FM 455 闭环控制模块

12 位分辨率时的采样时间为 20～180 ms，14 位时为 100～1700 ms，与实际使用的模拟量输入的数量有关，有 16 点数字量输入。

3. FM 458-1DP 应用模块

FM458-1DP 是为自由组态闭环控制设计的，有包含 300 个功能块的库函数和 CFC 连续功能图图形化组态软件，带有 PROFIBUS-DP 接口。

FM458-1DP 的基本模块可以执行计算、开环和闭环控制，通过扩展模块可以对 I/O 和通信进行扩展。

EXM 438-1 I/O 扩展模块是 FM458-1DP 的可选插入式扩展模块，用于读取和输出有时间要求的信号。有数字量/模拟量输入/输出模块，可连接增量式和绝对式编码器，有 4 个 12 位模拟量输出。

EXM 448 通信扩展模块是 FM458-1DP 的可选插入式扩展模块。可以使用 PROFIBUS-DP 或 SIMOLINK 进行高速通信，带有一个备用插槽，可以插入 MASTERRIVES 可选模块，用于建立 SIMOLINK 光纤通信。

FM458-1DP 还有一些附件接口模块，包括数字量输入、数字量输出和程序存储模块。

4. S5 智能 I/O 模块

S5 智能 I/O 模块可以用于 S7-400，配置专门设计的适配器后，可以直接插入 S7-400。可以使用 IP 242B 计数器模块，IP 244 温度控制模块，WF 705 位置解码器模块，WF 706 定位、位置测量和计数器模块，WF707 凸轮控制器模块，WF721 和 WF 723A/B/C 定位模块。

智能 I/O 模块的优点是它们能完全独立地执行实时任务，减轻了 CPU 的负担，使它能将精力完全集中于更高级的开环或闭环控制任务上。

2.7 S7-300/400 的维护

1. 更换 S7-300 的后备电池

一般应在使用一年后更换锂电池，只能在系统通电时更换，否则会丢失用户存储器中的程序和数据。更换时打开 CPU 模块的前盖，用螺钉旋具把旧的后备电池从电池盒中取出，新的电池的连接器插入电池盒，把电池推入电池盒，盖上 CPU 模块的前盖。

应在断电的情况下拔出或插入存储器卡。

2. 更换 S7-300 的信号模块

更换信号模块时把 CPU 切换到 STOP 状态，切断负载供电电源，打开信号模块的前盖，松开并取下模块的前连接器，松开模块的紧固螺钉，取下旧的模块。

取下新模块编码器的上半部分，将新模块固定在导轨上，将接好线的前连接器插入模块，并将它放到正常工作位置，关好模块的前盖，接通负载电源，执行一次 CPU 的完全再起动。

CPU 正在通过 MPI 交换数据时不能更换模块，如果不能确定，可以拔下 CPU 的 MPI 口上的连接器。

3. 更换 S7-300 信号模块的保险管

SM322 16×AC 120V 和 SM322 8×AC 230 V 数字量输出模块有熔丝管，应使用 8 A/250 V 的熔丝管。更换时模式开关应在 STOP 位置，切断负载电源，取下前连接器，松开模块的紧固螺钉，取下模块，拧下模块的熔丝管座，更换熔丝管后重新拧紧熔丝管座，安装模块，插入前连接器，重新接通负载电源。

4. 更换 S7-400 的后备电池

一般应在使用一年后更换锂电池，只能在系统通电时更换，否则会丢失用户存储器中的程序和数据。更换时打开电源盖，用带子把旧的后备电池从电池盒中拉出，插入新电池，插入时注意电池的极性。

"BATT INDIC"开关用于设置被监视的电源模块和电池的特征。"BAT"位置为单宽度电源模块(只占一个槽位)，"1BAT"位置用于双宽度或 3 宽度电源模块和一个电池，"2BAT"位置用于双宽度或 3 宽度电源模块和两个电池。最后应按 FMR 按钮以清除错误信息，然后盖上电源模块的前盖。

5. 更换 S7-400 的信号模块

更换信号模块时把 CPU 切换到 STOP 状态，或确保用户程序允许可以在 RUN 模式下更换模块。松开并取下模块的前连接器，松开模块的紧固螺钉，取下旧的模块。

安装新的模块，用螺钉紧固它。取下新模块编码器的上半部分，将接好线的前连接器插入模块并拧紧。如果 CPU 在 STOP 模式，将它切换到 RUN 模式。

CPU 正在通过 MPI 交换数据时不能更换模块，如果不能确定，可以拔下 CPU 的 MPI 口上的连接器。

2.8 ET 200 分布式 I/O

2.8.1 ET 200 的特点

西门子的 ET 200 是基于 PROFIBUS-DP 现场总线的分布式 I/O，可以与经过认证的非西门子公司生产的 PROFIBUS-DP 主站协同运行。

PROFIBUS 是为全集成自动化定制的开放的现场总线系统，它将现场设备连接到控制装置，并保证在各个部件之间的高速通信，从 I/O 传送信号到 PLC 的 CPU 模块只需毫秒级的时间。

全集成自动化概念和 STEP 7 使 ET 200 能与西门子的其他自动化系统协同运行，实现了从硬件配置到共享数据库等所有层次上的集成。所有的 I/O 均在一个软件的控制之下，因此用户在增加程序时不需要额外的培训。

因为 ET 200 只需要很小的空间，能使用体积更小的控制柜。集成的连接器代替了过去密密麻麻、杂乱无章的电缆，加快了安装过程，紧凑的结构使成本大幅度降低。

ET 200 能在非常严酷的环境(例如酷热、严寒、强压、潮湿或多粉尘)中使用。能提供连接光纤 PROFIBUS 网络的接口,可以节省费用昂贵的抗电磁干扰措施。

ET 200 集成了以下的功能:

(1) 电动机起动器

集成的电动机起动器用于异步电动机的单向或可逆起动,可以直接控制 7.5 kW 以下的电动机,节省了动力电缆,馈电电缆最大电流达 40 A,一个站可以带 6 个电动机起动器。

通过 PROFIBUS 现场总线网络可以调用开关状态和诊断信息,运行时能更换电动机起动器。

(2) 气动系统

经过适当的配置,ET 200 能用于阀门控制。ET 200X 很容易安装上这种阀门,直接由 PROFIBUS 总线控制,并由 STEP 7 软件包组态。

(3) 变频器

ET 200X 用于电气传动工程的模块提供变频器的所有功能。

(4) 智能传感器

光电式编码器或光电开关等可以与使用 IQ Sense 智能传感器的 ET 200S 进行通信。可以直接在控制器上进行所有设置,然后将数值传送到传感器。传感器出现故障时,系统诊断功能自动发出报警信号。

(5) 安全技术

ET 200 可以在冗余设计的容错控制系统或安全自动化系统中使用。集成的安全技术能显著地降低接线费用。安全技术包括紧急断开开关,安全门的监控以及众多与安全有关的电路。通过 ET 200S 故障防止模块、故障防止 CPU 和 PROFISafe 协议,与故障有关的信号亦能同标准功能一样在 PROFIBUS 网络上进行传送。

(6) 分布式智能

ET 200S 中的 IM 151/CPU 类似于大型 S7 控制器的功能,可以用 STEP 7 对它编程。它用分布式智能传送 I/O 子任务,因而减轻了中央控制器的负担,能对时间要求很高的信号快速作出响应,和简化对部件的管理。

(7) 功能模块

它们是用于 ET 200M 和 ET 200S 的附加模块,例如计数器、步进电动机或定位模块,以模块化的方法扩展分布式 I/O 的功能。

在启动前 ET 200 检查总线电缆、接口,可以通过 BT 200 总线测试单元来检查部件的状态。在运行时用监视和诊断工具提供不同部件的状态信息。PLC 可以通过 PROFIBUS 从 I/O 设备调用诊断信息,还可以接收到易于理解的报文。STEP 7 软件包自动地检测系统故障,并采取必要的响应措施。在运行时,可以使用诊断中继器对电缆进行检查和 S7-DIAG 诊断工具快速和高效地确定过程中发生的故障。

2.8.2 ET 200 的分类

ET 200 分为以下几个子系列:

(1) ET 200S 是分布式 I/O 系统,特别适用于需要电动机起动器和安全装置的开关柜,一个站最多可接 64 个子模块,模块种类丰富,有带通信功能的电动机起动器和集成的安全防护

系统(适用于机床及重型机械行业)和 IQ Sense 传感器等,集成有光纤接口。

(2) ET 200M 是多通道模块化的分布式 I/O,采用 S7-300 全系列模块,最多可扩展 8 个模块,可以连接 256 个 I/O 通道,适用于大点数、高性能的应用。它有支持 HART 协议的模块,可以将 HART 仪表接入现场总线。它具有集成的模块诊断功能,在运行时可以更换有源模块。提供与 S7-400H 系统相连的冗余接口模块和 IM153-2 集成光纤接口。

ET 200M 户外型是为野外应用设计的,其温度范围可达 -25～+60℃。

(3) ET 200is 是本质安全系统,通过紧固和本质安全的设计,ET200is 适用于有爆炸危险的区域。以点为单位的模块化 I/O 可以直接安装在 Zone 1 区域,而其传感器和执行器甚至能安装在 Zone 0 区域。能在运行时更换各种模块。

(4) ET 200X 是具有高保护等级 IP65/67 (NEMA4)的分布式 I/O 设备,其功能相当于 S7-300 的 CPU 314,最多 7 个具有多种功能的模块连接在一块基板上,可以连接电动机起动器、气动元件以及变频器,有气动模块和气动接口,实现了机、电、气动一体化。可以直接安装在机器上,节省开关柜的投资。它封装在一个坚固的玻璃纤维的塑料外壳中,可以用于有粉末和水流喷溅的场合。

(5) ET 200eco 是经济实用的 I/O,低成本的 ET 200eco 数字量 I/O 具有很高的保护等级(IP67),能在运行时更换模块,不会中断总线或供电。

(6) ET 200R 适用于机器人,用于恶劣的工业环境。例如在汽车生产过程中,ET 200R 直接安装在机器人内部。坚固的金属外壳和高的保护等级(IP65)使 ET 200R 能抗焊接火花的飞溅。由于 ET 200R 中集成有转发器功能,因而能减少机器人硬件部件的数量。ET 200R 使 PROFIBUS 能直接连接到焊接机器人的焊钳上。

由于有高的保护等级,抗冲击、防尘和不透水,ET 200X,ET 200eco 和 ET 200R 能适应严酷的工业环境,可以用于没有控制柜的 I/O 系统。它们只需要很少的附加部件。

(7) ET 200L 是小巧经济的分布式 I/O,像明信片大小的 I/O 模块适用于小规模的任务,十分方便地安装在 DIN 导轨上。ET 200L 分为以下 3 种:

- ET 200L 是整体式单元,不可扩展,只有数字量 I/O 模块。
- ET 200L-SC 是整体式单元,可以通过 Smart Connect(灵活连接系统)扩展最多 8 块数字量、模拟量模块。
- ET 200 L-SC IM-SC 是完全模块化的灵活连接系统,最多可以扩展 16 块模块。

(8) ET 200B:整体式的一体化分布式 I/O,有交流或直流的数字量 I/O 模块和模拟量 I/O模块,具有模块诊断功能。

第 3 章　S7-300/400 的编程语言与指令系统

3.1　S7-300/400 的编程语言

3.1.1　PLC 编程语言的国际标准

IEC(国际电工委员会)是为电子技术的所有领域制定全球标准的世界性组织。IEC 61131 是 PLC 的国际标准，1979 年成立了 IEC 61131 工作组，1992～1995 年发布了 IEC 61131 标准中的 1～4 部分，我国在 1995 年 11 月发布了 GB/T 15969-1/2/3/4(等同于 IEC 61131-1/2/3/4)。

IEC 61131 由以下 5 部分组成：

(1) 通用信息

定义 PLC 的术语，PLC 的主要功能和特点。包括典型的 PLC 中一般概念的定义和功能特征，例如用户程序的循环处理、输入输出过程映像，以及编程设备、PLC 和人机接口的分工。

(2) 设备要求与测试

具体说明对 PLC 电气、机械和功能的要求，以及对产品的检验方法，对下述各项指标都作了要求：温度、湿度、供电范围、接口保护、数字量信号的工作范围，以及机械应力等。

(3) 编程语言

通过对词汇、句法和语义的描述和例子，定义了 PLC 的软件模型，编程语言的标准和 5 种编程语言：梯形图、功能块图、指令表、顺序功能图和结构化文本。

(4) 用户指南

作为一个指南，对从事自动化项目的各阶段的用户提供帮助，从系统分析开始，到具体化阶段，例如 PLC 的选择与应用，安全和保护，安装与维护。

(5) 通信服务规范

描述了不同厂商生产的 PLC 之间、PLC 与其他设备之间的通信。包括设备功能选择、数据交换、报警处理、访问控制与网络管理、通信模式、通信块、与 ISO 协议的对应关系等。

其中的第三部分(IEC 61131-3)是 PLC 的编程语言标准，它鼓励不同的 PLC 制造商提供在外观和操作上相似的指令。IEC 61131-3 标准使用户在使用新的 PLC 时可以减少重新培训的时间；对于生产厂家，使用标准将减少产品开发的时间，可以投入更多的精力去满足用户的特殊要求。

由于 IEC 61131-3 自动化编程语言的诸多优点，它已成为自动化工业中拥有广泛应用基础的国际标准，已有越来越多的 PLC 厂家提供符合 IEC 61131-3 标准的产品，世界上著名的自动化设备制造商，例如西门子、罗克威尔、ABB、施耐德、GE、三菱、富士等公司都推出了不同程度与 IEC 61131-3 兼容的产品。不仅限于 PLC，IEC 61131-3 还广泛地应用于集散控制系统(DCS)和工业控制计算机、在个人计算机上运行的“软件 PLC”软件包、数控系统、远程终端单

元等产品。

IEC61131-3 包括以下内容：

(1) 编译为标准代码的规则：定义了 PLC 必须满足 IEC 61131 标准的哪些要求。在文献中必须包含一个符合标准的声明，或者系统必须生成一个这样的声明。

(2) 软件模型、通信模型和编程模型。

(3) 可编程逻辑控制语言中的通用元件，例如数据类型和变量、功能和功能块、程序和任务。

(4) 句法、语义和下述 5 种编程语言(见图 3-1)：

1) 指令表 IL(Instruction List)：语言语义的定义，这里只定义了 20 种基本操作。

2) 结构文本 ST(Structured Text)：西门子称为结构化控制语言(SCL)。

3) 梯形图 LD(Ladder Diagram)：西门子简称为 LAD。

4) 功能块图 FBD (Function Block Diagram)：标准中称为功能方框图语言。

5) 顺序功能图 SFC(Sequential Function Chart)：对应于西门子的 S7 Graph(见 5.6 节)。

(5) 附加的语法规则和编程实例。

标准中有两种图形语言——梯形图和功能块图，还有两种文字语言——指令表和结构文本，可以认为顺序功能图是一种结构块控制程序流程图。

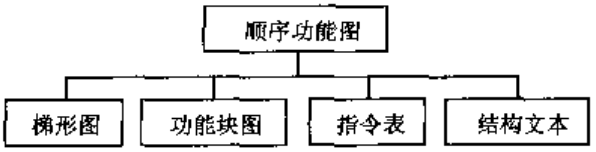


图 3-1 PLC 的编程语言

3.1.2 STEP 7 中的编程语言

STEP 7 是 S7-300/400 系列 PLC 的编程软件。梯形图、语句表（即指令表）和功能块图是标准的 STEP 7 软件包配备的 3 种基本编程语言，这 3 种语言可以在 STEP 7 中相互转换。STEP 7 还有多种编程语言可供用户选用，但是在购买软件时对可选的部分需要附加的费用。

1. 顺序功能图(SFC)

这是一种位于其他编程语言之上的图形语言，用来编制顺序控制程序，在 5.6 节中将对顺序功能图作详细的介绍。

STEP 7 中的 S7 Graph 顺序控制图形编程语言属于可选的软件包。在这种语言中，工艺过程被划分为若干个顺序出现的步，步中包含控制输出的动作，从一步到另一步的转换由转换条件控制。用 Graph 表达复杂的顺序控制过程非常清晰，用于编程及故障诊断更为有效，使 PLC 程序的结构更加易读，它特别适合于生产制造过程。S7 Graph 具有丰富的图形、窗口和缩放功能。系统化的结构和清晰的组织显示使 S7 Graph 对于顺序过程的控制更加有效。

2. 梯形图(LAD)

梯形图是使用得最多的 PLC 图形编程语言。梯形图与继电器电路图很相似，具有直观易懂的优点，很容易被工厂熟悉继电器控制的电气人员掌握，特别适合于数字量逻辑控制。有时把梯形图称为电路或程序。

梯形图由触点、线圈和用方框表示的指令框组成。触点代表逻辑输入条件，例如外部的开关、按钮和内部条件等。线圈通常代表逻辑运算的结果，常用来控制外部的指示灯、交流接触器和内部的标志位等。指令框用来表示定时器、计数器或者数学运算等附加指令。

使用编程软件可以直接生成和编辑梯形图，并将它下载到 PLC。

触点和线圈等组成的独立电路称为网络(Network),见图 3-2,编程软件自动为网络编号。

梯形图中的触点和线圈可以使用物理地址,例如 I0.2, Q1.3 等。如果在符号表中对某些地址定义了符号,例如令 I0.0 的符号为"起动",在程序中可用符号地址"起动"来代替物理地址 I0.0,使程序易于阅读和理解。

用户可以在网络号的右边加上网络的标题,在网络号的下面为网络加上注释。还可以选择在梯形图下面自动加上该网络中使用的符号的信息(Symbol Information)。

如果将两块独立电路放在同一个网络内,将会出错。本书为了节约篇幅,在插图中梯形图一般没有标出网络号,但是相邻网络左边的垂直线是断开的,以此来表示网络的分界点。

在分析梯形图中的逻辑关系时,为了借用继电器电路图的分析方法,可以想象在梯形图的左右两侧垂直"电源线"之间有一个左正右负的直流电源电压,当图 3-2 网络 1 中 I0.0 与 I0.1 的触点同时接通,或 Q0.0 与 I0.1 的触点同时接通时,有一个假想的"能流"(Power Flow)流过 Q0.0 的线圈。利用能流这一概念,可以帮助我们更好地理解和分析梯形图,能流只能从左向右流动。

如果没有跳转指令,在网络中,程序中的逻辑运算按从左往右的方向执行,与能流的方向一致。网络之间按从上到下的顺序执行,执行完所有的网络后,下一次循环返回最上面的网络(网络 1)重新开始执行。

3. 语句表(STL)

S7 系列 PLC 将指令表称为语句表(Statement List),它是一种类似于微机的汇编语言中的文本语言,多条语句组成一个程序段。语句表比较适合经验丰富的程序员使用,可以实现某些不能用梯形图或功能块图表示的功能。

4. 功能块图(FBD)

功能块图(FBD)使用类似于布尔代数的图形逻辑符号来表示控制逻辑。一些复杂的功能(例如数学运算功能等)用指令框来表示,有数字电路基础的人很容易掌握。功能块图用类似于与门、或门的方框来表示逻辑运算关系,方框的左侧为逻辑运算的输入变量,右侧为输出变量,输入、输出端的小圆圈表示"非"运算,方框被"导线"连接在一起,信号自左向右流动。图 3-4 中的控制逻辑与图 3-2 和图 3-3 中的相同。西门子公司的"LOGO!"系列微型 PLC 使用功能块图编程,除此之外,国内很少有人使用功能块图语言。

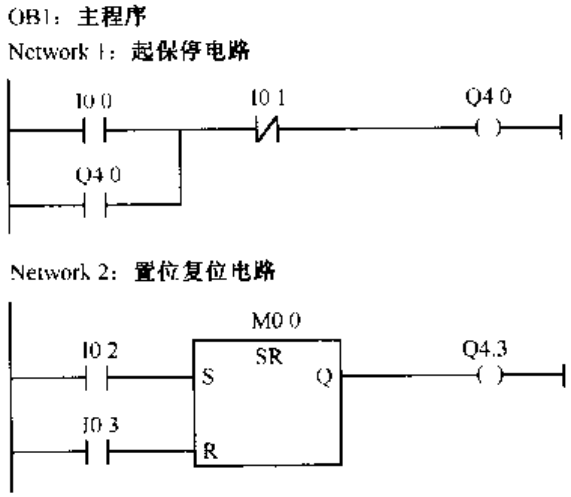


图 3-2 梯形图

```
OB1: 主程序
Network 1: 起保停电路
      A(
      O     I     0.0
      O     Q     4.0
      )
      AN    I     0.1
      =     Q     4.0

Network 2: 置位复位电路
      A     I     0.2
      S     M     0.0
      AN    I     0.3
      R     M     0.0
      A     M     0.0
      =     Q     4.3
```

图 3-3 语句表

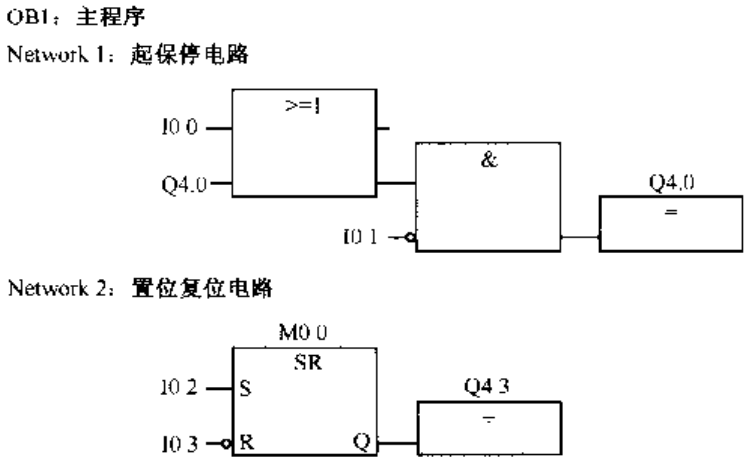


图 3-4　功能块图

5．结构文本(ST)

结构文本(ST)是为 IEC 61131-3 标准创建的一种专用的高级编程语言。与梯形图相比，它能实现复杂的数学运算，编写的程序非常简洁和紧凑。

STEP 7 的 S7 SCL(结构化控制语言)是符合 IEC 61131-3 标准的高级文本语言。它的语言结构与编程语言 Pascal 和 C 相似，所以特别适合于习惯使用高级编程语言的人使用。

SCL 适合于复杂的公式计算和最优化算法，或管理大量的数据等。

6．S7 HiGraph 编程语言

图形编程语言 S7 HiGraph 属于可选软件包，它用状态图(State Graphs)来描述异步、非顺序过程的编程语言。系统被分解为几个功能单元，每个单元呈现不同的状态，各功能单元的同步信息可以在图形之间交换。需要为不同状态之间的切换定义转换条件，用类似于语句表的语言描述指定给状态的动作和状态之间的转换条件。

7．S7 CFC 编程语言

可选软件包 CFC(Continuous Function Chart，连续功能图)用图形方式连接程序库中以块的形式提供的各种功能，包括从简单的逻辑操作到复杂的闭环和开环控制等领域。编程时将这些块复制到图中并用线连接起来即可。

不需要用户掌握详细的编程知识以及 PLC 的专门知识，只需要具有行业所必需的工艺技术方面的知识，就可以用 CFC 来编程。

8．编程语言的相互转换与选用

在 STEP 7 编程软件中，如果程序块没有错误，并且被正确地划分为网络，在梯形图、功能块图和语句表之间可以转换。用语句表编写的程序不一定能转换为梯形图。不能转换的网络仍然保留语句表的形式，但是并不表示该网络有错误。

语句表可供习惯用汇编语言编程的用户使用，在运行时间和要求的存储空间方面最优。语句表的输入方便快捷，还可以在每条语句的后面加上注释，便于复杂程序的阅读和理解。在设计通信、数学运算等高级应用程序时建议使用语句表。

梯形图与继电器电路图的表达方式极为相似，适合于熟悉继电器电路的用户使用。语句表程序较难阅读，其中的逻辑关系很难一眼看出，在设计和阅读有复杂的触点电路的程序时最好使用梯形图语言。

功能块图适合于熟悉数字电路的用户使用。

S7 SCL 编程语言适合于熟悉高级编程语言(例如 Pascal 或 C 语言)的用户使用,适合于数据处理程序。

S7 Graph、HiGraph 和 CFC 可供有技术背景,但是没有 PLC 编程经验的用户使用。S7 Graph 对顺序控制过程的编程非常方便,HiGraph 适合于异步非顺序过程的编程,CFC 适合于连续过程控制的编程。

9. S7-PLCSIM 仿真软件

即使没有 PLC 的硬件,使用 S7-PLCSIM 仿真软件也可以在计算机上对 SIMATIC S7 用户程序块进行功能测试,它对于用户程序的调试和 PLC 编程的学习是非常有用的。

它可以用于用下列语言编写的程序的仿真:LAD、FBD、STL、S7-GRAPH、S7-HiGraph、S7 SCL 和 CFC。

3.2 S7-300/400 CPU 的存储区

3.2.1 数制

1. 二进制数

二进制数的 1 位(bit)只能取 0 和 1 这两个不同的值,可以用来表示开关量(或称数字量)的两种不同的状态,例如触点的断开和接通,线圈的通电和断电等。如果该位为 1,表示梯形图中对应的位编程元件(例如位存储器 M 和输出过程映像 Q)的线圈"通电",其常开触点接通,常闭触点断开,以后称该编程元件为 1 状态,或称该编程元件 ON(接通)。如果该位为 0,对应的编程元件的线圈和触点的状态与上述的相反,称该编程元件为 0 状态,或称该编程元件 OFF(断开)。二进制常数用 2#表示,例如 2#1111_0110_1001_0001 是 16 位二进制常数.在编程手册和编程软件中,位编程元件的 1 状态和 0 状态常用 TURE 和 FALSE 来表示。

2. 十六进制数

十六进制的 16 个数字是 0～9 和 A～F(对应于十进制数 10～15),每个数字占二进制数的 4 位。B#16#,W#16#,DW#16#分别用来表示十六进制字节、字和双字常数,例如 W#16#13AF。在数字后面加"H"也可以表示十六进制数,例如 16#13AF 可以表示为 13AFH。

十六进制数的运算规则为逢 16 进 1,例如 B#16#3C=3×16+12=60。

3. BCD 码

BCD 码用 4 位二进制数表示一位十进制数,例如十进制数 9 对应的二进制数为 1001。4 位二进制数共有 16 种组合,有 6 种(1010～1111)没有在 BCD 码中使用。

BCD 码的最高 4 位二进制数用来表示符号,16 位 BCD 码字的范围为(999～+999。32 位 BCD 码双字的范围为(9 999 999～+9 999 999。

BCD 码实际上是十六进制数,但是各位之间的关系是逢十进一。十进制数可以很方便地转换为 BCD 码,例如十进制数 296 对应的 BCD 码为 W#16#296,或 2#0000 0010 1001 0110。

二进制整数 2#0000 0001 0010 1000 对应的十进制数也是 296,因为它的第 3 位、第 5 位

和第 8 位为 1，对应的十进制数为 $2^8+2^5+2^3=256+32+8=296$。

3.2.2 基本数据类型

STEP 7 有 3 种数据类型：

(1) 基本数据类型；

(2) 用户通过组合基本数据类型生成的复合数据类型；

(3) 可用来定义传送 FB(功能块)和 FC(功能)参数的参数类型。

下面介绍 STEP 7 的基本数据类型：

1. 位(bit)

位数据的数据类型为 BOOL(布尔)型，在编程软件中 BOOL 变量的值 1 和 0 常用英语单词 TURE(真)和 FALSE(假)来表示。

位存储单元的地址由字节地址和位地址组成，例如 I3.2 中的区域标示符“I”表示输入(Input)，字节地址为 3，位地址为 2(见图 3-5)。这种存取方式称为“字节．位”寻址方式。输入字节 IB3(B 是 Byte 的缩写)由 I3.0～I3.7 这 8 位组成。

2. 字节(Byte)

8 位二进制数组成 1 个字节 (Byte，见图 3-5)，其中的第 0 位为最低位(LSB)，第 7 位为最高位(MSB)。

3. 字(Word)

相邻的两个字节组成一个字，字用来表示无符号数。MW100 是由 MB100 和 MB101 组成的 1 个字(见图 3-6)，MB100 为高位字节。MW100 中的 M 为区域标示符，W 表示字，100 为字的起始字节 MB100 的地址。字的取值范围为 W#16#0000～W#16#FFFF。

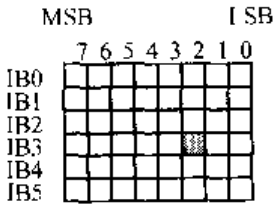


图 3-5 位数据的存放

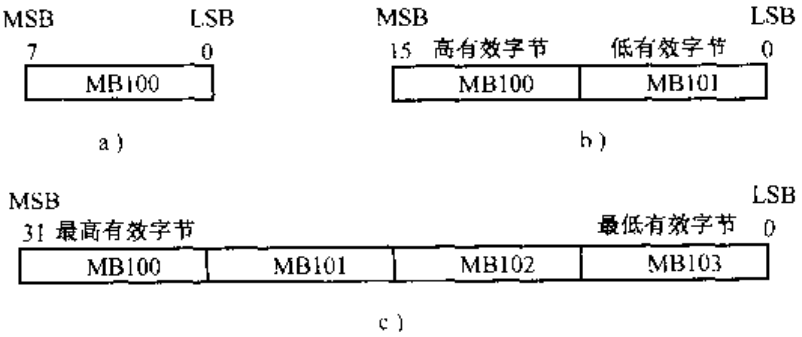


图 3-6 字节、字和双字

a) MB100 b) MW100 c) MD100

4. 双字(Double Word)

两个字组成 1 个双字，双字用来表示无符号数。MD100 是由 MB100～MB103 组成的 1 个双字(见图 3-6)，MB100 为高位字节，D 表示双字，100 为双字的起始字节 MB100 的地址。双字的取值范围为 DW#16#0000_0000～DW#16#FFFF_FFFF。

5. 16 位整数(INT，Integer)

整数是有符号数，整数的最高位为符号位，最高位为 0 时为正数，为 1 时为负数，取值范围为 -32 768～32 767。整数用补码来表示，正数的补码就是它的本身，将一个正数对应的二进制数的各位求反后加 1，可以得到绝对值与它相同的负数的补码。

6. 32 位整数(DINT,Double Integer)

32 位整数的最高位为符号位,取值范围为 -2 147 483 648~2 147 483 647。

7. 32 位浮点数

浮点数又称实数(REAL),浮点数可以表示为 $1.m/2^E$,例如 123.4 可表示为 1.234×10^2。符合 ANSI/IEEE 标准 754_1985 的基本格式的浮点数可表示为

$$浮点数=1.m\times2^e$$

式中指数 e = E +127($1\leqslant e\leqslant254$),为 8 位整数。

ANSI/IEEE 标准浮点数格式如图 3-7 所示,共占用一个双字(32 位)。最高位(第 31 位)为浮点数的符号位,最高位为 0 时为正数,为 1 时为负数;8 位指数占 23~30 位;因为规定尾数的整数部分总是为 1,只保留了尾数的小数部分 m(0~22 位)。浮点数的表示范围为 $\pm1.175495/10^{-38}\sim\pm3.402\ 823\times10^{38}$。

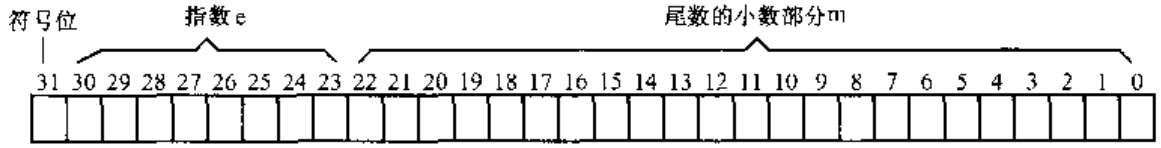


图 3-7 浮点数的结构

浮点数的优点是用很小的存储空间(4 B)可以表示非常大和非常小的数。PLC 输入和输出的数值大多是整数(例如模拟量输入值和模拟量输出值),用浮点数来处理这些数据需要进行整数和浮点数之间的相互转换,浮点数的运算速度比整数运算的慢得多。

8. 常数的表示方法

常数值可以是字节、字或双字,CPU 以二进制方式存储常数,常数也可以用十进制、十六进制 ASCII 码或浮点数形式来表示。如表 3-1 所示。

表 3-1 常数

符 号	说 明
B#16#,W#16#,DW#16#	十六进制字节,字和双字常数
D#	IEC 日期常数
L#	32 位整数常数
P#	地址指针常数
S5T#	S5 时间常数(16 位)
T#	IEC 时间常数
TOD#	实时时间常数(16 位/32 位)
C#	计数器常数(BCD 编码)
2#	二进制常数
B(b1,b2) B(b1,b2, b3,b4)	常数,2 B 或 4 B

B#16#,W#16#,DW#16# 分别用来表示十六进制字节、字和双字常数。2# 用来表示二进制常数,例如 2#1101 1010。

L# 为 32 位双整数常数,例如 L# +5。

P# 为地址指针常数,例如 P#M2.0 是 M2.0 的地址。

S5T# 是 16 位 S5 时间常数,格式为 S5T#aD_bH_cM_dS_eMS。其中 a, b, c, d, e 分别是日、小时、分、秒和毫秒的数值。输入时可以省掉下划线,例如 S5T#4S30MS = 4s30

ms，S5T#2H15M30S = 2 小时 15 分 30 秒。

S5 时间常数的取值范围为 S5T#0H_0M_0S_0MS～S5T#2H_46M_30S_0MS，时间增量为 10ms。

T#为带符号的 32 位 IEC 时间常数，例如 T#1D_12H_30M_0S_250MS，时间增量为 1ms，取值范围为 T#24D_20H_31M_23S_648MS～T#24D_20H_31M_23S_647MS。

DATE 是 IEC 日期常数，例如 D#2004-1-15。取值范围为 D#1990-1-1～D#2168-12-31。

TOD#是 32 位实时时间(Time of day)常数，时间增量为 1ms，例如 TOD#23:50:45.300。

C#为计数器常数(BCD 码)，例如 C#250。

8 位 ASCII 字符用单引号表示，例如 'ABC'。

3.2.3 复合数据类型与参数类型

1. 复合数据类型

通过组合基本数据类型和复合数据类型可以生成下面的数据类型：

(1) 数组(ARRAY)将一组同一类型的数据组合在一起，形成一个单元。

(2) 结构(STRUCT)将一组不同类型的数据组合在一起，形成一个单元。

(3) 字符串(STRING)是最多有 254 个字符(CHAR)的一维数组。

(4) 日期和时间(DATE_AND_TIME)用于存储年、月、日、时、分、秒、毫秒和星期，占用 8 个字节，用 BCD 格式保存。星期天的代码为 1，星期一～星期六的代码为 2～7。

例如 DT#2004-07-15-12:30:15.200 为 2004 年 7 月 15 日 12 时 30 分 15.2 秒。

(5) 用户定义的数据类型 UDT(User-Defined Data Types)：由用户将基本数据类型和复合数据类型组合在一起，形成的新的数据类型。

可以在数据块 DB 和变量声明表中定义复合数据类型。

2. 参数类型

参数类型是为在逻辑块之间传递参数的形参(Formal Parameter，形式参数)定义的数据类型：

(1) TIMER(定时器)和 COUNTER(计数器)：指定执行逻辑块时要使用的定时器和计数器，对应的实参(Actual Parameter，实际参数)应为定时器或计数器的编号，例如 T3，C21。

(2) BLOCK(块)：指定一个块用作输入和输出，参数声明决定了使用的块的类型，例如 FB、FC、DB 等。块参数类型的实参应为同类型的块的绝对地址编号(例如 FB2)或符号名(例如"Motor")。

(3) POINTER(指针)：指针指向一个变量的地址，即用地址作为实参。例如 P#M50.0 是指向 M50.0 的双字地址指针。

(4) ANY：用于实参的数据类型未知或实参可以使用任意数据类型的情况，占 10 B。

3.2.4 CPU 的存储区分布

S7 CPU 的存储器有 3 个基本区域(见图 3-8)。

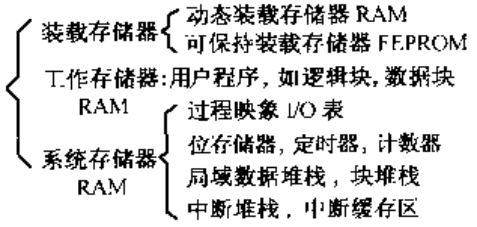


图 3-8 存储区分布

1. 装载存储器

装载存储器可能是 RAM 和 FEPROM,用于保存不包含符号地址和注释的用户程序和系统数据(组态、连接和模块参数等)。有的 CPU 有集成的装载存储器,有的可以用微存储器卡(MMC)来扩展,CPU 31xC 的用户程序只能装入插入式的 MMC。

断电时数据保存在 MMC 存储器中,因此数据块的内容基本上被永久保留。

下载程序时,用户程序(逻辑块和数据块)被下载到 CPU 的装载存储器,CPU 把可执行部分复制到工作存储器,符号表和注释保存在编程设备中。

2. 工作存储器

它是集成的高速存取的 RAM 存储器,用于存储 CPU 运行时的用户程序和数据,例如组织块、功能块、功能和数据块。为了保证程序执行的快速性和不过多地占用工作存储器,只有与程序执行有关的块被装入工作存储器。

STL 程序中的数据块可以被标识为"与执行无"(UNLINKED),它们只是存储在装载存储器中。有必要时可以用 SFC 20"BLKMOV"将它们复制到工作存储器。

复位 CPU 的存储器时,RAM 中的程序被清除,FEPROM 中的程序不会被清除。

3. 系统存储器

系统存储器是 CPU 为用户程序提供的存储器组件,被划分为若干个地址区域。使用指令可以在相应的地址区内对数据直接进行寻址。系统存储器为不能扩展的 RAM,用于存放用户程序的操作数据,例如过程映像输入、过程映像输出、位存储器、定时器和计数器、块堆栈(B 堆栈)、中断堆栈(I 堆栈)和诊断缓冲区等。

系统存储器还提供临时存储器(局域数据堆栈,即 L 堆栈),用来存储程序块被调用时的临时数据。访问局域数据比访问数据块中的数据更快。用户生成块时,可以声明临时变量(TEMP),它们只在执行该块时有效,执行完后就被覆盖了

4. 外设 I/O 存储区

通过外设 I/O 存储区(PI 和 PQ),用户可以不经过过程映像输入和过程映像输出,直接访问输入模块和输出模块。不能以位(bit)为单位访问外设 I/O 存储区,只能以字节、字和双字为单位访问。

3.2.5 系统存储器

1. 过程映像输入/输出(I/Q)表

在扫描循环开始时,CPU 读取数字量输入模块的输入信号的状态,并将它们存入过程映像输入表(Process Image Input,PII)中。

用户程序访问 PLC 的输入(I)和输出(Q)地址区时,不是去读写数字信号模块中的信号状态,而是访问 CPU 中的过程映像区。在扫描循环中,用户程序计算输出值,并将它们存入过程映像输出表(Process Image Output,PIQ)。在循环扫描开始时将过程映像输出表的内容写入数字量输出模块。

I 和 Q 均可以按位、字节、字和双字来存取,例如 I0.0、IB0、IW0 和 ID0。

与直接访问 I/O 模块相比,访问过程映像表可以保证在整个程序周期内,过程映像的状态始终一致。即使在程序执行过程中接在输入模块的外部信号状态发生了变化,过程映像表中的信号状态仍然保持不变,直到下一个循环被刷新。由于过程映像保存在 CPU 的系统存储

器中，访问速度比直接访问信号模块快得多。见表 3-2。

表 3-2 系统存储区

存储区	访问的单位	标识符	说明
输入过程映像(I)	输入位 输入字节 输入字 输入双字	I IB IW ID	在每次执行 OB 1 扫描循环开始之前，CPU 将输入模块的输入数值复制到输入过程映像表中
输出过程映像(Q)	输出位 输出字节 输出字 输出双字	Q QB QW QD	在一个程序循环扫描过程中，将程序运算得到的输出值写入输出过程映像表中。在下一次 OB 1 循环扫描开始时，CPU 将这些数值传送到输出模块
位存储器(M)	存储位 位存储字节 位存储字 位存储双字	M MB MW MD	该区域用于存储用户程序的中间运算结果或标志位
定时器(T)	定时器	T	在该区域提供定时器的存储区
计数器(C)	计数器	C	在该区域提供计数器的存储区
外设输出(PQ)	外设输出字节 外设输出字 外设输出双字	PQB PQW PQD	通过该区域用户程序直接访问输出模块
外设输入(PI)	外设输入字节 外设输入字 外设输入双字	PIB PIW PID	通过该区域用户程序直接访问输入模块
共享数据块(DB)	数据块 数据位 数据字节 数据字 数据双字	DB DBX DBB DBW DBD	共享数据块可供所有逻辑块使用，可以用“OPN DB”指令打开一个共享数据块
背景数据块(DIB)	数据块 数据位 数据字节 数据字 数据双字	DI DIX DIB DIW DID	背景数据块与某一功能块或系统功能块相关联，可以用“OPN DI”指令打开一个背景数据块
局域数据(L)	局域数据位 局域数据字节 局域数据字 局域数据双字	L LB LW LD	在处理组织块、功能块和系统功能块时，相应块的临时数据保存到该块的局域数据区。

输入过程映像在用户程序中的标识符为 I，是 PLC 接收外部输入的数字量信号的窗口。输入端可以外接常开触点或常闭触点，也可以接多个触点组成的串并联电路。PLC 将外部电路的通/断状态读入并存储在输入过程映像中，外部输入电路接通时，对应的输入过程映像为 ON(1 状态)；反之为 OFF(0 状态)。在梯形图中，可以多次使用输入过程映像的常开触点和常闭触点。

输出过程映像在用户程序中的标识符为 Q，在循环周期开始时，CPU 将输出过程映像的数据传送给输出模块，再由后者驱动外部负载。如果梯形图中 Q0.0 的线圈“通电”，继电器型输出模块中对应的硬件继电器的常开触点闭合，使接在 Q0.0 对应的输出端子的外部负载工

作。输出模块中的每一个硬件继电器仅有一对常开触点，但是在梯形图中，每一个输出位的常开触点和常闭触点都可以多次使用。

除了操作系统对过程映像的自动刷新外，S7-400 CPU 可以将过程映像划分为最多 15 个区段，这意味着如果需要，可以独立于循环，刷新过程映像表的某些区段，用 STEP 7 指定的过程映像区段中的每一个 I/O 地址不再属于 OB1 过程映像输入/输出表。需要定义哪些 I/O 模块地址属于哪些过程映像区段。

可以在用户程序中用 SFC(系统功能)刷新过程映像。SFC 26"UPDAT _ PI"用来刷新整个或部分过程映像输入表，SFC 27"UPDAT _ PO"用来刷新整个或部分过程映像输出表。

某些 CPU 也可以调用 OB(组织块)由系统自动地对指定的过程映像分区刷新。

2. 内部存储器标志位(M)存储器区

内部存储器标志位用来保存控制逻辑的中间操作状态或其他控制信息。虽然名为"位存储器区"，表示按位存取，但是也可以按字节、字或双字来存取。

3. 定时器(T)存储器区

定时器相当于继电器系统中的时间继电器。给定时器分配的字用于存储时间基值和时间值(0～999)。时间值可以用二进制或 BCD 码方式读取。

4. 计数器(C)存储器区

计数器用来累计其计数脉冲上升沿的次数，有加计数器、减计数器和加减计数器。给计数器分配的字用于存储计数当前值(0～999)。计数值可以用二进制或 BCD 码方式读取。

5. 数据块(DB)与背景数据块(DI)

DB 为数据块，DBX 是数据块中的数据位，DBB、DBW 和 DBD 分别是数据块中的数据字节、数据字和数据双字。

DI 为背景数据块，DIX 是背景数据块中的数据位，DIB、DIW 和 DID 分别是背景数据块中的数据字节、数据字和数据双字。

6. 外设 I/O 区(PI/PQ)

外设输入(PI)和外设输出(PQ)区允许直接访问本地的和分布式的输入模块和输出模块。可以按字节(PIB 或 PQB)、字(PIW 或 PQW)或双字(PID 或 PQD)存取，不能以位为单位存取 PI 和 PO。

3.2.6 CPU 中的寄存器

1. 累加器(ACCUx)

32 位累加器用于处理字节、字或双字的寄存器。S7-300 有两个累加器(ACCU1 和 ACCU2)，S7-400 有 4 个累加器(ACCU1～ACCU4)。可以把操作数送入累加器，并在累加器中进行运算和处理，保存在 ACCU1 中的运算结果可以传送到存储区。处理 8 位或 16 位数据时，数据放在累加器的低端(右对齐)。

2. 状态字寄存器(16 位)

状态字(见图 3-9)是一个 16 位的寄存器，用于存储 CPU 执行指令的状态。状态字中的某些位用于决定某些指令是否执行和以什么样的方式执行，执行指令时可能改变状态字中的某些位，用位逻辑指令和字逻辑指令可以访问和检测它们。

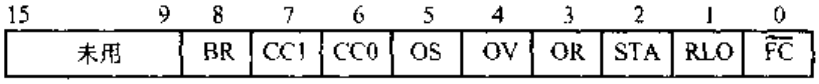


图 3-9　状态字的结构

(1) 首次检测位

状态字的第 0 位称为首次检测位($\overline{FC}$)。若该位的状态为 0,则表明一个梯形逻辑网络的开始,或指令为逻辑串的第一条指令。CPU 对逻辑串第一条指令的检测(称为首次检测)产生的结果直接保存在状态字的 RLO 位中,经过首次检测存放在 RLO 中的 0 或 1 称为首次检测结果。该位在逻辑串的开始时总是 0,在逻辑串指令执行过程中该位为 1,输出指令或与逻辑运算有关的转移指令(表示一个逻辑串结束的指令)将该位清 0。

(2) 逻辑运算结果(RLO)

状态字的第 1 位称为逻辑运算结果 RLO(Result of Logic Operation)。该位用来存储执行位逻辑指令或比较指令的结果。RLO 的状态为 1,表示有能流流到梯形图中运算点处;为 0 则表示无能流流到该点。可以用 RLO 触发跳转指令。

(3) 状态位(STA)

状态字的第 2 位称为状态位,执行位逻辑指令时,STA 总是与该位的值一致。

(4) 或位(OR)

状态字的第 3 位称为或位(OR),在先逻辑"与"后逻辑"或"的逻辑运算中,OR 位暂存逻辑"与"的操作结果,以便进行后面的逻辑"或"运算。其他指令将 OR 位复位。

(5) 溢出位(OV)

状态字的第 4 位称为溢出位,如果算术运算或浮点数比较指令执行时出现错误(例如溢出、非法操作和不规范的格式),溢出位被置 1。如果后面的同类指令执行结果正常,该位被清 0。

(6) 溢出状态保持位(OS)

状态字的第 5 位称为溢出状态保持位,或称为存储溢出位。OV 位被置 1 时 OS 位也被置 1,OV 位被清 0 时 OS 仍保持,所以它保存了 OV 位,用于指明前面的指令执行过程中是否产生过错误。只有 JOS(OS=1 时跳转)指令、块调用指令和块结束指令才能复位 OS 位。

(7) 条件码 1(CC1)和条件码 0(CC0)

状态字的第 7 位和第 6 位称为条件码 1 和条件码 0。这两位综合起来用于表示在累加器 1 中产生的算术运算或逻辑运算的结果与 0 的大小关系、比较指令的执行结果或移位指令的移出位状态(见表 3-3 和表 3-4)。

表 3-3　算术运算后的 CC1 和 CC0

CC1	CC0	算术运算无溢出	整数算术运算有溢出	浮点数算术运算有溢出
0	0	结果=0	整数相加下溢出(负数绝对值过大)	正数、负数绝对值过小
0	1	结果<0	乘法下溢出;加减法上溢出(正数过大)	负数绝对值过大
1	0	结果>0	乘除法上溢出,加减法下溢出	正数上溢出
1	1	-	除法或 MOD 指令的除数为 0	非法的浮点数

表 3-4 指令执行后的 CC1 和 CC0

CC1	CC0	比较指令	移位和循环移位指令	字逻辑指令
0	0	累加器 2 = 累加器 1	移出位为 0	结果为 0
0	1	累加器 2 < 累加器 1	-	-
1	0	累加器 2 > 累加器 1	-	结果不为 0
1	1	非法的浮点数	移出位为 1	-

(8) 二进制结果位(BR)

状态字的第 8 位称为二进制结果位。它将字处理程序与位处理联系起来,在一段既有位操作又有字操作的程序中,用于表示字操作结果是否正确。将 BR 位加入程序后,无论字操作结果如何,都不会造成二进制逻辑链中断。在梯形图的方框指令中,BR 位与 ENO 有对应关系,用于表明方框指令是否被正确执行:如果执行出现了错误,BR 位为 0,ENO 也为 0;如果功能被正确执行,BR 位为 1,ENO 也为 1。

在用户编写的 FB 和 FC 程序中,必须对 BR 位进行管理,功能块正确执行后,使 BR 位为 1,否则使其为 0。使用 SAVE 指令可将 RLO 存入 BR 中,从而达到管理 BR 位的目的。当 FB 或 FC 执行无错误时,使 RLO 为 1,并存入 BR;否则在 BR 中存入 0。

状态字的 9～15 位未使用。

3. 数据块寄存器

DB 和 DI 寄存器分别用来保存打开的共享数据块和背景数据块的编号。

4. 诊断缓冲区

诊断缓冲区是系统状态列表的一部分,包括系统诊断事件和用户定义的诊断事件的信息。这些信息按它们出现的顺序排列,第一行中是最新的事件。

诊断事件包括模块的故障、写处理的错误、CPU 中的系统错误、CPU 的运行模式切换错误、用户程序中的错误和用户用系统功能 SFC 52 定义的诊断错误。

3.2.7 寻址方式

操作数是指令操作或运算的对象,寻址方式是指令取得操作数的方式,操作数可以直接给出或间接给出。

1. 立即寻址

立即寻址的操作数直接在指令中,有些指令的操作数是惟一的,为简化起见不在指令中写出。表 3-5 是立即寻址的示例。下面是使用立即寻址的程序实例:

```
SET             //把 RLO 置 1
L     1352      //把常数 1352 装入累加器 1
AW    W#16#3A12 //常数 16#3A12 与累加器 1 的低字相"与",运算结果在累加器 1 的低字中。
```

表 3-5 立即寻址举例

举例	说明
L +27	将 16 位整数常数"27"装入 ACCU1 中
L L#-1	将 32 位整数常数"-1"装入 ACCU1 中

（续）

举　例	说　明
L　2#1010_1010_1010_1010	将16位二进制常数装入ACCU1中
L　DW#16#A0F0_BCFD	将十六进制双字常数装入ACCU1中
L　'END'	将ASCII字符装入ACCU1中
L　T#500MS	将时间值500ms装入ACCU1中
L　C#100	将计数值装入ACCU1中
L　B#(100,12)	装入2字节无符号常数
L　B=(100,12,50,8)	装入4字节无符号常数
L　P#I0.0	将内部区域指针装入ACCU1中
L　P#Q20.6	将交叉区域指针装入ACCU1中
L　-2.5	将实数(浮点数)装入ACCU1中
L　D#1995-01-20	将日期装入ACCU1中
L　TOD#13:20:33.125	将实时时间装入ACCU1中

2. 直接寻址

直接寻址在指令中直接给出存储器或寄存器的区域、长度和位置，例如用MW200指定位存储区中的字，地址为200；MB100表示以字节方式存取，MW100表示存取MB100、VB101组成的字，MD100表示存取MB100～MB103组成的双字。下面是直接寻址的程序实例：

```
A     Q0.5
L     MW4          //把MW 4装入累加器1
T     DBW2         //把累加器1低字中的内容传送给数据字DBW2
```

直接寻址举例见表3-6。

表3-6　直接寻址举例

A I0.0	输入位I0.0的"与"(AND)操作
L　IB2	将输入字节IB2装入ACCU1的低字节
L　IW6	将输入字IW6装入ACCU1的低字
L　ID30	将输入双字ID30装入ACCU1

3. 存储器间接寻址

在存储器间接寻址指令中，给出一个作地址指针的存储器，该存储器的内容是操作数所在存储单元的地址。使用存储器间接寻址可以改变操作数的地址，在循环程序中经常使用存储器间接寻址。

地址指针可以是字或双字，定时器(T)、计数器(C)、数据块(DB)、功能块(FB)和功能(FC)的编号范围小于65 535，使用字指针就够了。

其他地址则要使用双字指针，如果要用双字格式的指针访问一个字、字节或双字存储器，必须保证指针的位编号为0，例如P#Q20.0。双字指针的格式如图3-10所示：位0～2为被寻址地址中位的编号(0～7)，位3～18为被寻址的字节的编号(0～65535)。只有双字MD、LD、DBD和DID能作地址指针。下面是存储器间接寻址的例子：

```
L     QB[DBD10]    //将输出字节装入累加器1，输出字节的地址指针在数据双字DBD10中
                   //如果DBD10的值为2#0000 0000 0000 0000 0000 0000 0010 0000，装入
```

```
                    的是 QB4
A       M[LD 4]     //对存储器位作"与"运算,地址指针在数据双字 LD4 中
                    //如果 LD4 的值为 2#0000 0000 0000 0000 0000 0000 0010 0011,则是对
                      M4.3 进行操作
```

4. 寄存器间接寻址

S7 中有两个地址寄存器 AR1 和 AR2,通过它们可以对各存储区的存储器内容作寄存器间接寻址。地址寄存器的内容加上偏移量形成地址指针,后者指向数值所在的存储单元。

地址寄存器存储的双字地址指针见图 3-11。其中第 0～2 位(xxx)为被寻址地址中位的编号(0～7),第 3～18 位(bbbb bbbb bbbb bbbb)为被寻址地址的字节的编号(0～65535)。第 24～26 位(rrr)为被寻址地址的区域标识号,第 31 位 x = 0 为区域内的间接寻址,第 31 位 x=1为区域间的间接寻址。

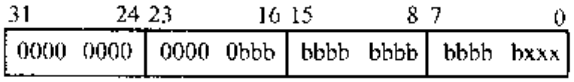


图 3-10 存储器间接寻址的双字指针格式

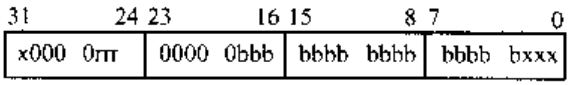


图 3-11 寄存器间接寻址的双字指针格式

第一种地址指针格式包括被寻址数值所在存储单元地址的字节编号和位编号,存储区的类型在指令中给出,例如 L DBB[AR1, P#6.0]。这种指针格式适用于在某一存储区内寻址,即区内寄存器间接寻址。第 24～26 位(rrr)应为 0。

第二种地址指针格式的第 24～26 位还包含了说明数值所在存储区的存储区域标识符的编号 rrr,用这几位可实现跨区寻址,这种指针格式用于区域间寄存器间接寻址。

区域间寄存器间接寻址的区域标识位如表 3-7 所示。

表 3-7 区域间寄存器间接寻址的区域标识位

区域标识符	存 储 区	位 26～24 的二进制数
P	外设输入输出	000
I	输入过程映像	001
Q	输出过程映像	010
M	位存储区	011
DBX	共享数据块	100
DIX	背景数据块	101
L	正在执行的块的局域数据	111

如果要用寄存器指针访问一个字节、字或双字,必须保证指针中的位地址编号为 0。

指针常数 P#5.0 对应的二进制数为 2#0000 0000 0000 0000 0000 0000 0010 1000。下面是区内间接寻址的例子:

```
L       P#5.0               //将间接寻址的指针装入累加器 1
LAR1                        //将累加器 1 中的内容送到地址寄存器 1
A       M[AR1, P#2.3]       //AR1 中的 P#5.0 加偏移量 P#2.3, 实际上是对 M7.3 进行操作
=       Q[AR1, P#0.2]       //逻辑运算的结果送 Q5.2
L       DBW[AR1, P#18.0]    //将 DBW23 装入累加器 1
```

下面是区域间间接寻址的例子:

```
L       P#M6.0              //将存储器位 M6.0 的双字指针装入累加器 1
LAR1                        //将累加器 1 中的内容送到地址寄存器 1
T       W[AR1, P#50.0]      //将累加器 1 的内容传送到存储器字 MW56
```

P#M6.0 对应的二进制数为 2#1000 0011 0000 0000 0000 0000 0011 0000。因为地址指针 P#M6.0 中已经包含有区域信息，使用间接寻址的指令 T W[AR1, P#50]中没有必要再用地址标识符 M。

3.3 位逻辑指令

位逻辑指令用于二进制数的逻辑运算，二进制数只有 0 和 1 这两个数。1 相当于编程元件的线圈通电，0 相当于线圈断电。位逻辑运算的结果(Result of Logic Operation)简称为 RLO。

3.3.1 触点指令

1. 触点与线圈

在语句表中，用 A(AND，与)指令来表示串联的常开触点。用 O (OR，或)指令来表示并联的常开触点。触点指令中变量的数据类型为 BOOL(布尔)型。常开触点对应的地址位为 1 状态时，该触点闭合。

在语句表中，用 AN (AND NOT，与非)来表示串联的常闭触点，用 ON (OR NOT，或非)来表示并联的常闭触点，触点符号中间的“/”表示常闭。常闭触点对应的地址位为 0 状态时该触点闭合。

输出指令“=”将 RLO 写入地址位，输出指令与线圈相对应。驱动线圈的触点电路接通时，有“能流”流过线圈，RLO = 1，对应的地址位为 1 状态，反之则 RLO = 0，对应的地址位为 0 状态。线圈应放在梯形图的最右边。下面是图 3-12 对应的语句表，其中的 L20.0 是用来保存运算结果的局域变量，局域变量只能在程序所在的逻辑块中使用。将梯形图转换为语句表时，局域变量 L20.0 是自动分配的。

```
A(
A       I 0.0
AN      I 0.1
O       I 0.2
)
A       I 0.3
ON      C 5
=       L 20.0
A       L 20.0
=       Q 4.3
A       L 20.0
=       Q 4.4
A       L 20.0
AN      I 3.4
=       Q 4.6
```

2. 取反触点

取反触点的中间标有“NOT”，用来将它左边电路的逻辑运算结果 RLO 取反(见图 3-13)，该运算结果若为 1 则变为 0，为 0 则变为 1，该指令没有操作数。换句话说，能流到达该触点时即停止流动；若能流未到达该触点，该触点给右侧供给能流。图 3-13 中左边的两个触点均闭合时，Q4.5 的线圈断电。

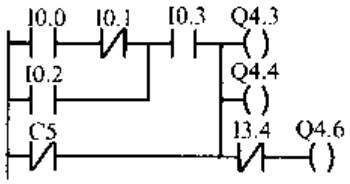


图 3-12　触点与输出指令

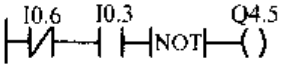


图 3-13　取反触点

3. 电路块的串联和并联

电路块的串、并联电路如图 3-14、3-15 所示。触点的串并联指令只能将单个触点与其他触点电路串并联。逻辑运算时采用先“与”(串联)后“或”(并联)的规则，例如(I0.0 + M3.3)·(M0.0 + I0.Z)。要想将图 3-14 中由 I0.5 和 I0.2 的触点组成的串联电路与它上面的电路并联，需要在两个串联电路块对应的指令之间使用没有地址的 O 指令。

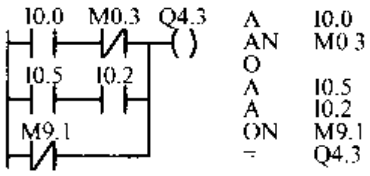


图 3-14　电路块的并联

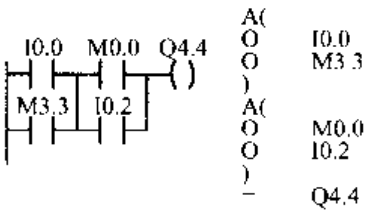


图 3-15　电路块的串联

将电路块串联时，应将需要串联的两个电路块用括号括起来，并在左括号之前使用 A 指令，就像对单独的触点使用 A 指令一样。

电路块用括号括起来后，在括号之前还可以使用 AN、O、ON、X 和 XN 指令。

表 3-8 给出了位逻辑指令。

表 3-8　位逻辑指令

指　令	说　明
A	AND，逻辑与，电路或触点串联
AN	AND NOT，逻辑与非，常闭触点串联
O	OR，逻辑或，电路或触点并联
ON	OR NOT，逻辑或非，常闭触点并联
X	XOR，逻辑异或
XN	XOR NOT，逻辑异或非
A(	逻辑与加左括号
AN(	逻辑与非加左括号
O(	逻辑或加左括号
ON(	逻辑或非加左括号
X (	逻辑异或加左括号
XN (	逻辑异或非加左括号

（续）

指　令	说　明
)	右括号
=	赋值
R	RESET，复位指定的位或定时器、计数器
S	SET，置位指定的位或设置计数器的预置值
NOT	将 RLO 取反
SET	将 RLO 置位为 1
CLR	将 RLO 清 0
SAVE	将状态字中的 RLO 保存到 BR 位
FN	下降沿检测
FP	上升沿检测

4. 中线输出指令

中线输出是一种中间赋值元件，用该元件指定的地址来保存它左边电路的逻辑运算结果（RLO 位，或能流的状态）。中间标有“#”号的中线输出线圈与其他触点串联，就像一个插入的触点一样。中线输出只能放在梯形图的中间，不能接在左侧的垂直“电源线”上，也不能放在电路最右端结束的位置。图 3-16a 可以用中线输出指令等效为图 3-16b。

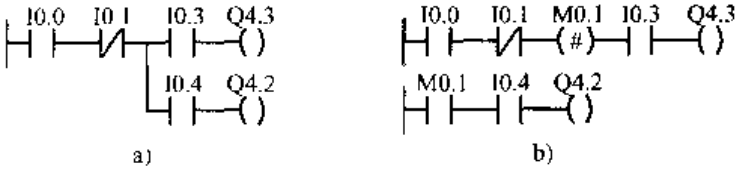


图 3-16　中线输出指令

如果该指令使用局域数据区（L 区）的地址，在逻辑块（FC、FB 和 OB）的变量声明表中，该地址应声明为 TEMP 类型。下面是图 3-16b 中第一行对应的语句表。

```
A     I0.0
AN    I0.1
=     M0.1
A     M0.1
A     I0.3
=     Q4.3
```

5. 异或指令与同或指令

异或指令的助记符为 X，图 3-17 是异或指令的等效电路。图 3-17 中的 I0.0 和 I0.2 的状态不同时，运算结果 RLO 为 1，反之为 0。

同或指令的助记符为 XN，图 3-18 是同或指令的等效电路。图 3-18 中的 I0.0 和 I0.2 的状态相同时，运算结果 RLO 为 1，反之为 0。

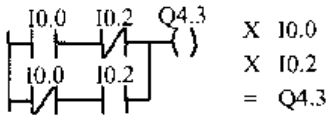


图 3-17　异或

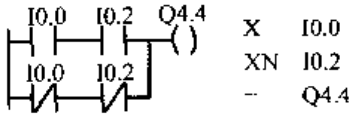


图 3-18　同或

3.3.2 输出类指令

1. 赋值指令

赋值指令(=)将逻辑运算结果 RLO 写入指定的地址位,对应于梯形图中的线圈。

2. 置位与复位

S (Set,置位或置 1)指令将指定的地址位置位(变为 1 并保持)。

R (Reset,复位或置 0)指令将指定的地址位复位(变为 0 并保持)。

如果图 3-19 中 I0.1 的常开触点接通,Q4.3 变为 1 并保持该状态,即使 I0.1 的常开触点断开,它也仍然保持 1 状态。I0.3 的常开触点闭合时,Q4.3 变为 0,并保持该状态,即使 I0.3 的常开触点断开,它也仍然保持 0 状态。如果被指定复位的是定时器(T)或计数器(C),将清除定时器/计数器的定时/计数当前值,并将它们的地址位复位。

3. RS 触发器

如果图 3-20 左边的 R 输入 I0.4 为 1 且 S 输入 I0.6 为 0,RS 复位置位触发器被复位,M0.0 与 Q4.1 均为 0 状态。如果 S 输入 I0.6 为 1 且 R 输入 I0.4 为 0,则被置位,M0.0 与 Q4.1 均为 1 状态。如果两个输入信号的状态均为 1,因为先执行复位指令,后执行置位指令,执行完后 RS 触发器被置位,M0.0 与 Q4.1 均为 1 状态。

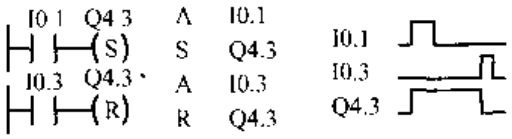


图 3-19 置位与复位

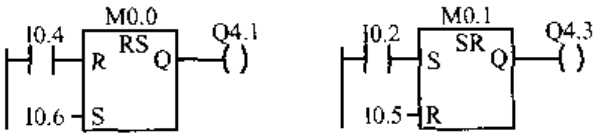


图 3-20 RS 触发器与 SR 触发器

4. SR 触发器

如果图 3-20 右边的 S 输入 I0.2 为 1 且 R 输入 I0.5 为 0,SR 置位复位触发器被置位,M0.1 与 Q4.3 均为 1 状态。如果 R 输入 I0.5 为 1 且 S 输入 I0.2 为 0,则触发器被复位,M0.1 与 Q4.3 均为 0 状态。如果两个输入均为 1,因为先执行置位指令,后执行复位指令,执行完后 SR 触发器被复位,M0.1 与 Q4.3 均为 0 状态。

3.3.3 其他指令

1. RLO 边沿检测指令

图 3-21 中的 I0.3 和 I0.0 组成的串联电路由断开变为接通时,中间标有"P"的上升沿检测线圈左边的 RLO(逻辑运算结果)由 0 变为 1(即波形的上升沿),检测到一次正跳变,能流将在一个扫描周期内流过检测元件,Q4.5 的线圈仅在这一个扫描周期内"通电"。检测元件的地址(例如图 3-21 中的 M0.0 和 M0.1)为边沿存储位,用来储存上一次循环的 RLO。在波形图中,用高电平表示 1 状态。

图 3-21 中的 I0.3 和 I0.0 组成的串联电路由接通变为断开时,中间标有"N"的检测元件左边的 RLO 由 1 变为 0(即波形的下降沿),检测到一次负跳变,能流将在一个扫描周期内流过检测元件,Q4.3 的线圈仅在

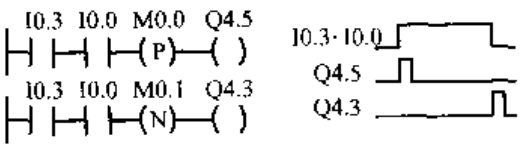


图 3-21 上升沿与下降沿检测

这一个扫描周期内“通电”。

正/负跳变语句的助记符分别为 FP(Positive RLO Edge,上升沿)和 FN(Negative RLO Edge,下降沿)。下面是图 3-21 对应的语句表程序:

```
Network 1:
        A       I0.3
        A       I0.0
        FP
        =       Q4.5
Network 2:
        A       I0.3
        A       I0.0
        FN
        =       Q4.3
```

2. 信号边沿检测指令

ROS 是单个地址位提供的信号的上升沿检测(Positive RLO Edge Detection)指令,如果图 3-22 中 I0.1 的常开触点接通,且 I0.2 由 0 变为 1(即输入信号 I0.2 的上升沿),Q4.3 的线圈“通电”一个扫描周期。M0.0 为边沿存储位,用来储存上一次循环时 I0.2 的状态。

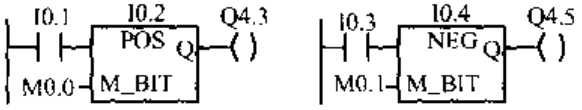


图 3-22 上升沿检测与下降沿检测

NEG 是单个地址位提供的信号的下降沿检测(Negative RLO Edge Detection)指令,如果图 3-22 中 I0.3 的常开触点接通,且 I0.4 由 1 变为 0(即输入信号 I0.4 的下降沿),Q4.5 的线圈“通电”一个扫描周期。M0.1 为边沿存储位,用来储存上一次循环时 I0.4 的状态。

下面是图 3-22 中右图对应的语句表程序,其中的 BLD 指令与梯形图的显示有关。

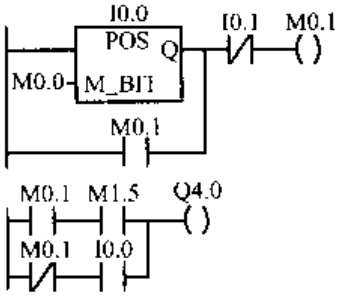
```
A       I0.3
A(
A       I0.4
BLD     100
FN      M0.1
)
=       Q4.5
```

【例 3-1】 设计故障信息显示电路,若故障信号 I0.0 为 1,使 Q4.0 控制的指示灯以 1 Hz 的频率闪烁。操作人员按复位按钮 I0.1 后,如果故障已经消失,指示灯熄灭。如果没有消失,指示灯转为常亮,直至故障消失。

故障信息显示电路如图 3-23 所示,在设置 CPU 的属性时(见 4.2 节),令 M1 为时钟存储器字节,其中的 M1.5 提供周期为 1s 的时钟脉冲。出现故障时,将 I0.0 提供的故障信号用 M0.1 锁存起来,使 Q4.0 控制的指示灯以 1Hz 的频率闪烁。按复位按钮 I0.1 后,将故障锁存信号 M0.1 复位为 0 状态,如果这时故障已经消失,指示灯熄灭。如果没有消失,M0.1 的常闭触点与 I0.0 的常开触点组成的串联电路使指示灯转为常亮,直至故障消失,I0.0 变为 0 状态。

3. 将 RLO 保存在 BR 寄存器中

SAVE 指令(见图 3-24)将 RLO 保存到状态字的 BR 位,首次检查位"/FC"不会被复位,由于这个原因,在下一个网络中,BR 位的状态将参加"与"逻辑运算。



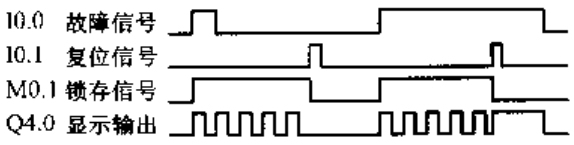


图 3-23 故障信息显示

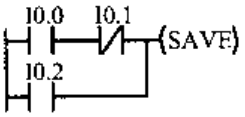


图 3-24 保存 RLO

建议一般不要用 SAVE 指令保存 RLO,并在本逻辑块或下一个逻辑块中检查保存的 BR 位的值,因为在保存和检查操作之间,BR 的值可能已被很多指令修改了。

但是在退出逻辑块之前可以使用 SAVE 指令,因为使能输出 ENO(即 BR 位)被设置为 RLO 位的值,可以用于块的错误检查。

4. SET 与 CLR 指令

SET 与 CLR(Clear)指令将 RLO(逻辑运算结果)置位或复位,紧接在它们后面的赋值语句中的地址将变为 1 状态或 0 状态。

```
SET             //将 RLO 置位
=     M0.2      //M0.2 的线圈"通电"
CLR             //将 RLO 复位
=     Q4.7      //Q4.7 的线圈"断电"
```

3.4 定时器与计数器指令

3.4.1 定时器指令

1. 定时器的种类和存储区

定时器相当于继电器电路中的时间继电器,S7 300/400 的定时器分为脉冲定时器(SP)、扩展脉冲定时器(SE)、接通延时定时器(SD)、保持型接通延时定时器(SS)和断开延时定时器(SF)。图 3-25 中的"t"是定时器的时间设定值。

S5 是西门子 PLC 老产品的系列号,S5 定时器是 S5 系列 PLC 的定时器,在梯形图中用指令框(Box)的形式表示。此外每一个 S5 定时器都有功能相同的用线圈形式表示的定时器。

S7 CPU 为定时器保留了一片存储区域。每个定时器有一个 16 位的字和一个二进制位,定时器的字用来存放它当前的定时时间值,定时器触点的状态由它的位的状态来决定。用定时器地址(T 和定时器号,例如 T6)来存取它的时间值和定时器位,带位操作数的指令存取定时器位,带字操作数的指令存取定时器的时间值。不同的 CPU 支持 32～512 个定时器。

2. 定时器字的表示方法

用户使用的定时器字由 3 位 BCD 码时间值(0～999)和时基组成(见图 3-26),时基是时间基准的简称,时间值以指定的时基为单位。在 CPU 内部,时间值以二进制格式存放,占定时器字的 0～9 位。

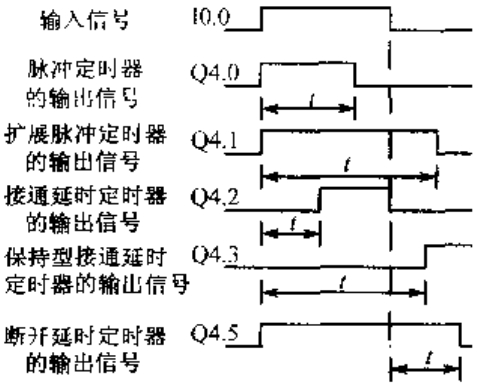


图 3-25 定时器功能

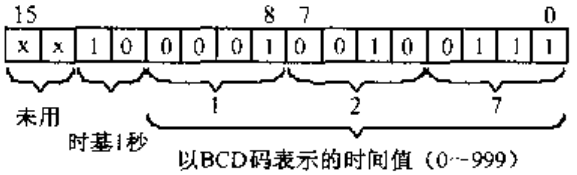


图 3-26 定时器字

可以按下列的形式将时间预置值装入累加器的低位字:

(1) 十六进制数 W#16#wxyz,其中的 w 是时间基准,xyz 是 BCD 码格式的时间值。

(2) S5T#aH_bM_cS_dMS,其中 H 表示小时,M 为分钟,S 为秒,MS 为毫秒,a, b, c, d 为用户设置的值。可输入的最大时间值为 9990s,或 2H_46M_30S。例如 S5T#1H_12M_18S 为 1 h 12min18 s,S5T#18S 为 18 s。时基是 CPU 自动选择的,选择的原则是在满足定时范围要求的条件下选择最小的时基。

定时器指令见表 3-9。

表 3-9 定时器指令

指 令	说 明
FR	允许定时器再启动
L	将定时器的二进制时间值装入累加器 1
LC	将定时器的 BCD 时间值装入累加器 1
R	复位定时器
SD	接通延时定时器
SE	扩展的脉冲定时器
SF	断开延时定时器
SP	脉冲定时器
SS	保持型接通延时定时器

3. 时基

定时器字的第 12 位和第 13 位用于时基(时间基准),时基代码为二进制数 00、01、10 和 11 时,对应的时基分别为 10ms、100ms、1s 和 10s。实际的定时时间等于时间值乘以时基值。例如定时器字为 W#16#3999 时,时基为 10s,定时时间为 9990s。时基反映了定时器的分辨率,时基越小分辨率越高,可定时的时间越短,时基越大分辨率越低,可定时的时间越长。

4. 接通延时定时器的定时过程

接通延时定时器的线圈通电,定时器被启动,操作系统自动地将累加器低字的内容(定时时间预置值)装入定时器。如果用语句表编程,在定时器起动之前,建议用下面两条指令中的一条将定时器的预置值装入累加器:

```
L     W#16#wxyz          //w 和 xyz 均为十进制数,时基 w=1～3,时间值 xyz=1～999
```

```
L     S5T#aH_bM_cS_dMS   //a, b, c, d 分别为小时、分、秒和毫秒、自动选择时基
```

S5 格式的时间预置值范围为 0S～2H_46M_30S(9990 s)，时间增量为 10 ms。

定时器被启动后，从预置值开始，在每一个时间基准内，它的时间值减 1，直到减为 0，表示定时时间到，这时定时器位被置为 1，梯形图中该定时器的常开触点闭合，常闭触点断开。

5. S5 脉冲定时器(Pulse S5 Timer)

脉冲定时器的功能类似于数字电路中上升沿触发的单稳态电路。图 3-27 左边的指令框中，S 为脉冲定时器的设置输入端，TV 为预置值输入端，R 为复位输入端；Q 为定时器位输出端，BI 输出十六进制格式的当前时间值，BCD 输出当前时间值的 BCD 码。

在 I0.0 提供的启动输入信号 S 的上升沿，脉冲定时器开始定时，输出 Q4.0 变为 1。定时器的当前时间值等于 TV 端输入的预置值(即初值)减去启动后的时间值。定时时间到时，当前时间值变为 0，Q 输出变为 0 状态。在定时期间，如果 I0.0 的常开触点断开，则停止定时，当前时间值变为 0，Q4.0 的线圈断电。图 3-28 中的 t 是定时器的预置值。

R 是复位输入端，在定时器输出为 1 时，如果复位输入 I0.1 由 0 变为 1，定时器被复位，复位后输出 Q4.0 变为 0 状态，当前时间值和时标被清 0。

BI 输出端输出不带时基的十六进制整数格式的定时器当前值，BCD 输出端输出 BCD 码格式的当前时间值和时基。

定时器中的 S、R、Q 为 BOOL(位)变量，BI 和 BCD 为 WORD(字)变量，TV 为 S5TIME 变量。各变量均可以使用 I、Q、M、L、D 存储区，TV 也可以使用定时时间常数 S5T#。

6. 脉冲定时器线圈 (Pulse Timer Coil)

脉冲定时器线圈的功能和与时序图和 S5 脉冲定时器的相同，定时器位为 1 时，定时器的常开触点闭合，常闭触点断开。在图 3-29 中，当 I0.0 的常开触点由断开变为接通时(即逻辑运算结果 RLO 的上升沿)，定时器开始定时，T0 的常开触点闭合。定时时间到时，T0 的常开触点断开。在定时期间，如果 I0.0 变为 0 状态，或者复位输入 I0.1 变为 1 状态，T0 的常开触点都将断开，定时器的当前值被清 0。

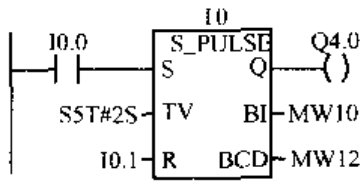


图 3-27　S5 脉冲定时器

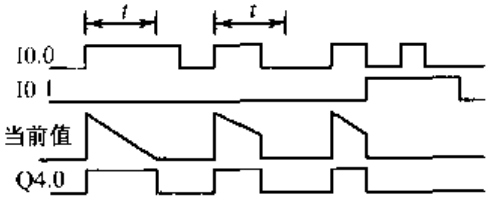


图 3-28　S5 脉冲定时器时序图

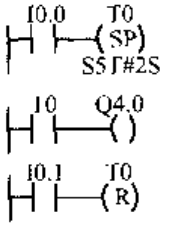


图 3-29　脉冲定时器

下面是用语句表编写的脉冲定时器的程序。其中仅在语句表中使用的 FR 指令允许定时器再启动，即控制 FR 的 RLO(I1.2)由 0 变为 1 状态时，重新装入定时时间，定时器又从预置值开始定时。再启动只是在定时器的启动条件满足(图 3-28 中的 I0.0＝1)时起作用。该指令可以用于所有的定时器，但是它不是启动定时器定时的必要条件。

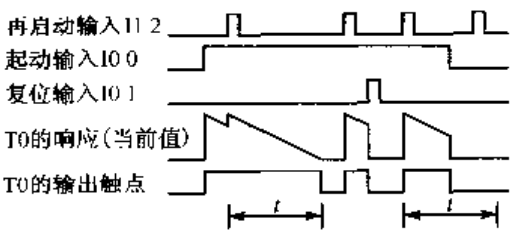


图 3-30　脉冲定时器的再启动时序图

```
A       I 1.2
FR      T0      //允许定时器 T1 再起动
A       I 0.0
L       S5T#2s  //预置值 2s 送入累加器 1
SP      T0      //启动 T0
A       I 0.1
R       T0      //复位 T0
L       T0      //将 T0 的十六进制时间当前值装入累加器 1
T       MW10    //将累加器 1 的内容传送到 MW10
LC      T0      //将 T0 的 BCD 时间当前值装入累加器 1.
T       MW12    //将累加器 1 的内容传送到 MW12
A       T0      //检查 T0 的信号状态
=       Q 4.0   //T0 的定时器位为 1 时,Q4.0 的线圈通电
```

在语句表中,用装入指令(L)将不带时基的十六进制整数格式的当前值传送到累加器 1 的低字,用 LC 指令将 BCD 码格式的定时器当前值和时基装入累加器 1 的低字。

7. S5 扩展脉冲定时器(Extended Pulse S5 Timer)

S5 扩展脉冲定时器(见图 3-31)各输入输出端的意义与 S5 脉冲定时器相同。在启动输入信号 S 的上升沿,脉冲定时器开始定时,在定时期间,Q 输出端为 1 状态,直到定时结束。在定时期间即使 S 输入变为 0 状态,仍继续定时,Q 输出端为 1 状态,直到定时结束。在定时期间,如果 S 输入又由 0 变为 1 状态,定时器被重新启动,开始以预置的时间值定时。

R 输入由 0 变为 1 状态时,定时器被复位,停止定时。复位后 Q 输出端变为 0 状态,当前时间和时标被清 0。如图 3-32 所示。

8. 扩展的脉冲定时器线圈

在图 3-33 中,当 I0.2 的常开触点由断开变为接通时(RLO 的上升沿),定时器 T1 开始定时,在定时期间,T1 的常开触点闭合。定时时间到时,T1 的常开触点断开。在定时期间,即使 I0.2 变为 0 状态,仍继续定时。定时期间如果 I0.2 又由 0 变为 1 状态,定时器被重新启动。复位输入 I0.3 由 0 变为 1 状态时,T1 被复位,其常开触点断开。下面是图 3-33 对应的指令表程序。

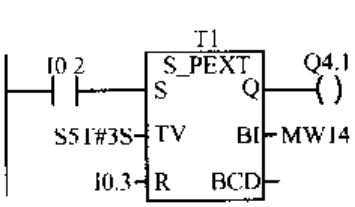


图 3-31 S5 扩展脉冲定时器

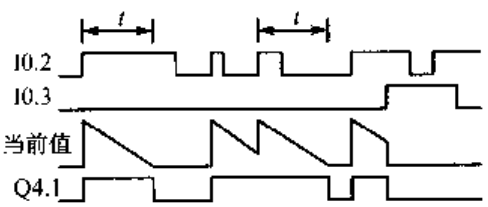


图 3-32 时序图

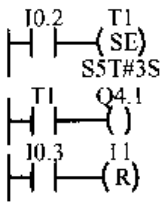


图 3-33 扩展的脉冲定时器

```
A       I 0.2
L       S5T#3s  //预置值 3s 送入累加器 1
SE      T1      //启动 T1
A       T1      //检查 T1 的信号状态
=       Q4.1
```

```
A       I 0.3
R       T1          //复位 T1
```

9. S5 接通延时定时器(On-Delay S5 Timer)

接通延时定时器是使用得最多的定时器,有的厂家的 PLC 只有接通延时定时器。定时器各输入端和输出端的意义与 S5 脉冲定时器相同。在启动输入信号 S 的上升沿,定时器开始定时。定时器的当前时间值等于预置值(即初值)TV 减去启动后的时间值。如果定时期间 S 的状态一直为 1,定时时间到时,当前时间值变为 0,Q 输出端变为 1 状态,使 Q4.2 的线圈通电。此后如果 S 输入由 1 变为 0,Q 输出端的信号状态也变为 0。

在定时期间,如果 S 输入由 1 变为 0,则停止定时,当前时间值保持不变。S 又变为 1 时,又从预置值开始定时(见图 3-35)。

R 是复位输入信号,定时器的 S 输入为 1 时,不管定时时间是否已到,只要复位输出 R 由 0 变为 1,定时器都要被复位,复位后当前时间和时基被清 0。如果定时时间已到,复位后输出 Q 由 1 变为 0。

10. 接通延时定时器线圈(On Delay Timer Coil)

在图 3-34、图 3-36 中,当 I0.4 的常开触点由断开变为接通时(RLO 的上升沿),定时器 T2 开始定时。如果 I0.4 一直为 1,定时时间到时,T2 的常开触点闭合。在定时期间如果 SD 的线圈断电,T2 的当前时间保持不变。线圈重新通电时,又从预置值开始定时。复位输入 I0.5 变为 1 时,T2 的常开触点断开,时间值被清 0。

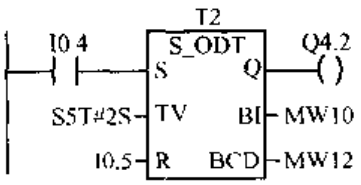


图 3-34 S5 接通延时定时器

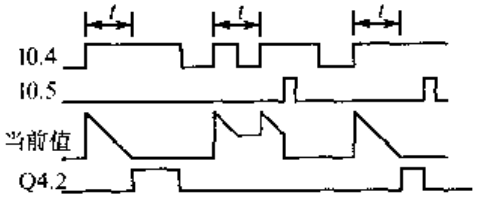


图 3-35 时序图

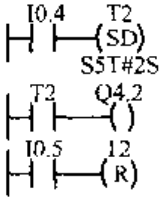


图 3-36 接通延时定时器

【例 3-2】 用定时器设计延时接通延时断开电路。

图 3-37 中的电路用 I0.0 控制 Q4.6,I0.0 的常开触点接通后,T6 开始定时,4s 后 T6 的常开触点接通,使断开延时定时器 T7 的线圈通电,T7 的常开触点接通,使 Q4.6 的线圈通电。I0.0 变为 0 状态后 T7 开始定时,3s 后 T7 的定时时间到,其常开触点断开,使 Q4.6 变为 0 状态。

【例 3-3】 用定时器设计周期和占空比可调的振荡电路。

图 3-38 中 I0.0 的常开触点接通后,T8 的线圈通电,开始定时。2 s 后定时时间到,T8 的常开触点接通,使 Q4.7 变为 1 状态,同时 T9 开始定时。3s 后 T9 的定时时间到,它的常闭触点断开,使 T8 的线圈断电,T8 的常开触点断开,使 Q4.7 和 T9 的线圈断电。下一个扫描周期因 T9 的常闭触点接通,T8 又从预置值开始定时,以后 Q4.7 的线圈将这样周期性地通电和断电,直到 I0.0 变为 0 状态。Q4.7 线圈通电和断电的时间分别等于 T9 和 T8 的预置值。振荡电路实际上是一个有正反馈的电路,T8 和 T9 通过它们的触点分别控制对方的线圈,形成了正反馈。

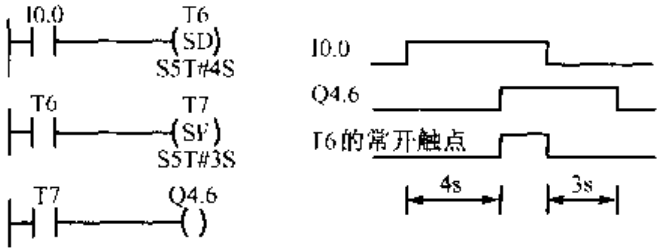


图 3-37　延时接通/断开电路

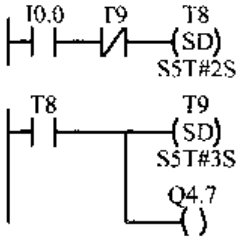


图 3-38　振荡电路

CPU 的时钟存储器字节中的各位输出周期为 0.1～2 s 的时钟脉冲(见 4.2.3),它们输出高电平和低电平时间相等的方波信号,可以用它们的触点来控制需要闪烁的指示灯。

11. S5 保持型接通延时定时器(Retentive On-Delay S5 Timer)

定时器各输入端和输出端的意义与 S5 接通延时定时器相同。在启动输入信号 S 的上升沿,定时器开始定时(见图 3-39、图 3-40),定时期间即使输入 S 变为 0,仍继续定时。定时时间到时,输出 Q 变为 1 并保持。在定时期间,如果输入 S 又由 0 变为 1,定时器被重新启动,又从预置值开始定时。不管输入 S 是什么状态,只要复位输入 R 从 0 变为 1,定时器就被复位,输出 Q 变为 0。

12. 保持型接通延时定时器线圈(Retentive On-Delay Timer Coil)

在图 3-41 中,当 I0.6 的常开触点由断开变为接通时(RLO 的上升沿),定时器开始定时。定时期间即使 T3 的线圈断电,仍继续定时。定时时间到时,T3 的定时器位变为 1,其常开触点闭合。只有复位输入 I0.7 变为 1,才能使 T3 复位,复位后其定时器位变为 0,常开触点断开。在定时期间,I0.6 的常开触点如果断开后又变为接通,定时器将被重新启动,以设置的预置值重新开始定时。

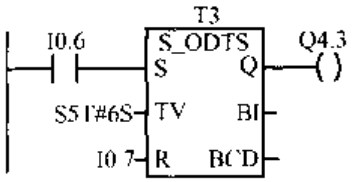


图 3-39　保持型接通延时

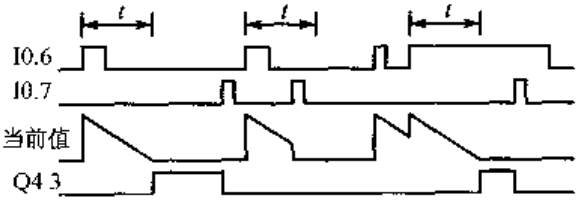


图 3-40　时序图

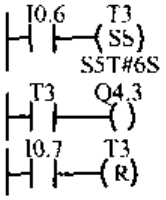


图 3-41　S5 定时器

13. S5 断开延时定时器 (Off-Delay S5 Timer)

定时器各输入输出端的意义与 S5 脉冲延时定时器相同。在图 3-42、图 3-43 中,在启动输入信号 S 的上升沿,定时器的 Q 输出信号变为 1 状态,当前时间值为 0。在 S 输入的下降沿,定时器开始定时。定时时间到时,输出 Q 变为 0 状态。

正在定时的时候,如果 S 信号由 0 变为 1,定时器的时间值保持不变,停止定时。如果输入 S 重新变为 0,定时器从预置值开始重新启动定时。

复位输入 I1.1 为 1 状态时,定时器被复位,时间值被清 0,输出 Q 变为 0 状态。

14. 断开延时定时器线圈(Off-Delay Timer Coil)

在图 3-44 中,当 I1.0 的常开触点由断开变为接通(RLO 的上升沿)时,T5 的输出变为 1,其常开触点闭合。在 I1.0 的下降沿,定时器开始定时。定时时间到时,T5 的时间值变为 0,其常开触点断开。

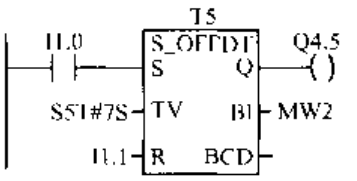


图 3-42　S5 断开延时定时器

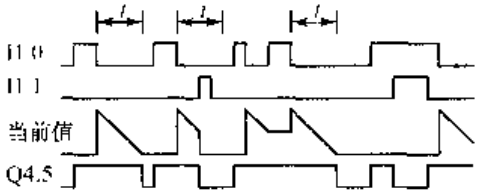


图 3-43　时序图

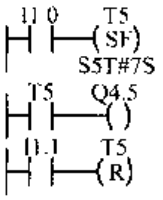


图 3-44　断开延时定时器

正在定时的时候，如果 I1.0 的常开触点由断开变为接通，定时器的时间值保持不变，停止定时。如果 I1.0 的常开触点重新断开，定时器从预置值开始重新起动定时。

复位输入 I1.1 为 1 状态时，定时器被复位，时间值被清 0，Q4.5 的线圈断电。

3.4.2　计数器指令

1. 计数器的存储器区

S7 CPU 为计数器保留了一片计数器存储区。每个计数器有一个 16 位的字和一个二进制位，计数器的字用来存放它的当前计数值，计数器触点的状态由它的位的状态来决定。用计数器地址(C 和计数器号，如 C24)来存取当前计数值和计数器位，带位操作数的指令存取计数器位，带字操作数的指令存取计数器的计数值。不同的 CPU 支持 32～512 个计数器，只有计数器指令能访问计数器存储器区。

2. 计数值

计数器字的 0～11 位是计数值的 BCD 码，计数值的范围为 0～999。

计数器字的计数值为 BCD 码 127 时，计数器单元中的各位如图 3-45 所示，用格式 C#127 表示 BCD 码 127。二进制格式的计数值只占用计数器字的 0～9 位。

3. 加计数器(S_CU)

图 3-46 左边的指令框中，S 为加计数器(Up Counter)的设置输入端，PV 为预置值输入端，CU 为加计数脉冲输入端，R 为复位输入端；Q 为计数器位输出端，CV 输出十六进制格式的当前计数值，CV_BCD 输出当前计数值的 BCD 码。见表 3-10。

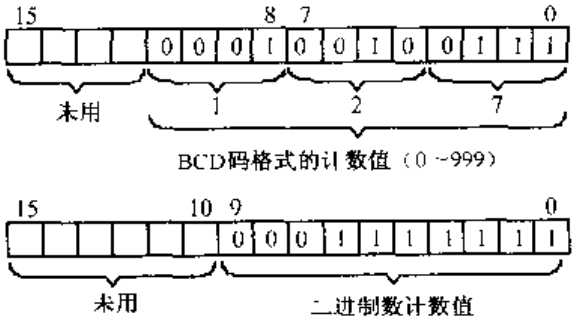


图 3-45　计数器字

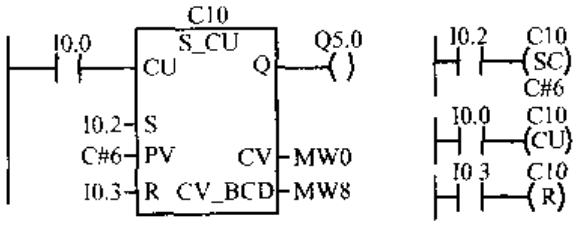


图 3-46　加计数器

在“设置”输入信号 I0.2 的上升沿，将预置值 PV 指定的值送入计数器字。在“加计数脉冲”输入信号 I0.0 的上升沿，如果计数值小于 999，计数值加 1。“复位”输入信号 I0.3 为 1 时，计数器被复位，计数值被清 0。计数值大于 0 时计数器位(即输出 Q)为 1；计数值为 0 时，计数器位亦为 0。

如果在用“设置”输入 S 设置计数器时 CU 输入为 1，即使信号没有变化，下一扫描周期也会计数。

计数器中的 CU、S、R、Q 为 BOOL（位）变量，PV、CV 和 CV_BCD 为 WORD（字）变量。各变量均可以使用 I、Q、M、L、D 存储区，PV 还可以使用计数器常数 C#。

表 3-10 计数器指令

指 令	说 明
FR	允许计数器再起动
L	将计数器的二进制计数值装入累加器 1
LC	将计数器的 BCD 计数值装入累加器 1
R	复位计数器
S	将计数器的预置值送入计数器字
CU	加计数器
CD	减计数器

4. 加计数器线圈（S_CU）

设置计数值线圈 SC（Set Counter Value）用来设置计数值，该指令仅在 RLO 的上升沿（由 0 变为 1）时执行，此时预置值被送入指定的计数器。图 3-46 中 I0.2 的触点由断开变为接通时，预置值 6 被送入计数器 C10。

图中标有 CU 的线圈为加计数器线圈（Up Counter Coil）。在 I0.0 的上升沿，如果计数值小于 6，计数值加 1。复位输入 I0.3 为 1 时，计数器被复位，计数值被清 0。

装入指令（L）将计数器的当前值（整数）传送到累加器的低字。LC 指令将 BCD 码格式的计数器当前值装入累加器的低字。下面是图 3-46 中左边的电路对应的语句表：

```
A     I0.0      //在 I0.0 的上升沿
CU    C10       //加计数器 C10 的当前值加 1
BLD   101
A     I0.2      //在 I0.2 的上升沿
L     C#6       //计数器的预置值 6 被装入累加器的低字
S     C10       //将预置值装入计数器 C10
A     I0.3      //如果 I0.3 为 1
R     C10       //复位 C10
L     C10       //将 C10 的二进制计数当前值装入累加器 1
T     MW0       //将累加器 1 的内容传送到 MW0
LC    C10       //将 C10 的 BCD 计数当前值装入累加器 1
T     MW8       //将累加器 1 的内容传送到 MW8
A     C10       //如果 C10 的当前值非 0
=     Q 5.0     //Q 5.0 为 1 状态
```

5. 减计数器（S_CD）

在图 3-47 中的设置输入 S 的上升沿，用 PV 指定的值预置减计数器（Down Counter）。在减计数输入信号 CD 的上升沿，如果计数值大于 0，计数值减 1。复位输入 R 为 1 时，计数器被复位，计数值被清 0。计数值大于 0 时计数器的输出 Q 为 1；计数值为 0 时，Q 亦为 0。

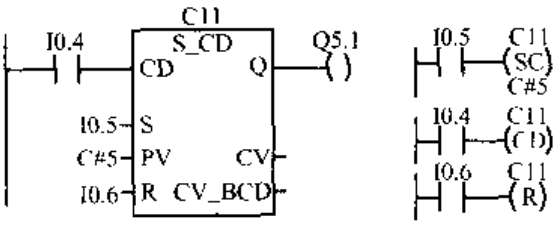


图 3-47 减计数器

如果在设置计数器时 CD 输入为 1，即使信号没有变化，下一扫描周期也会计数。

6. 减计数器线圈(S_CU)

图 3-47 中标有 CD 的线圈为减计数线圈(Down Counter Coil),I0.5 的触点由断开变为接通时,预置值 5 被送入计数器 C11。在 I0.4 的上升沿,如果计数值大于 0,计数值减 1。计数值非 0 时,C11 的常开触点闭合,为 0 时 C11 的常开触点断开。复位输入 I0.6 为 1 时,计数器被复位,计数值被清 0。

为了使计数器能计预置值指定的脉冲数,可将预置值送入减计数器,其计数值减为 0 时,其常闭触点闭合,表示计了预置值指定的数。

【例 3-4】 用计数器扩展定时器的定时范围。

S7-300/400 的定时器的最长定时时间为 9990s,如果需要更长的定时时间,可以使用图 3-48 所示的电路。I0.0 为 0 状态时,计数器 C0 被复位。

I0.0 变为 1 状态时,其常开触点接通,使 T11 和 T12 组成的振荡电路(见图 3-48)开始工作,计数器的预置值 999 被送入计数器 C0。I0.0 的常闭触点断开,C0 被解除复位。

振荡电路的振荡周期为 T11 和 T12 预置值之和,图中的振荡电路相当于周期为 4 小时的时钟脉冲发生器。每隔 4 小时,当 T12 的定时时间到,T11 的常开触点由接通变为断开,其脉冲的下降沿通过减计数线圈 CD 使 C0 的计数值减 1。计满 999 个数(即 3996h)后,C0 的当前值减为 0,它的常闭触点闭合,使 Q5.4 的线圈通电。总的定时时间等于振荡电路的振荡周期乘以 C0 的计数预置值。

7. 加减计数器(S_CUD)

在设置输入 S 的上升沿(见图 3-49),用 PV 指定的预置值设置可逆计数器(Up Down Counter)。复位输入 R 为 1 时,计数器被复位,计数值被清 0。在加计数输入信号 CU 的上升沿,如果计数器值小于 999,计数器加 1。在减计数输入信号 CD 的上升沿,如果计数器值大于 0,计数值减 1。如果两个计数输入均为上升沿,两条指令均被执行,计数值保持不变。计数值大于 0 时输出信号 Q 为 1;计数值为 0 时,Q 亦为 0。

如果在设置计数器时 CU 或 CD 输入为 1,即使信号没有变化,下一扫描周期也会计数。

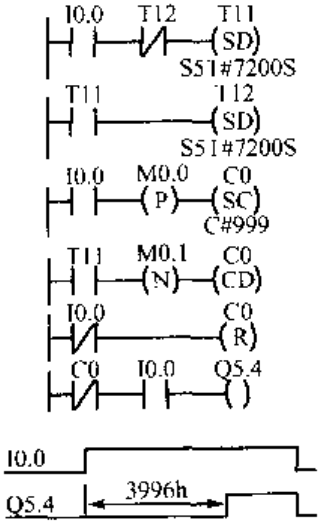


图 3-48 定时范围的扩展

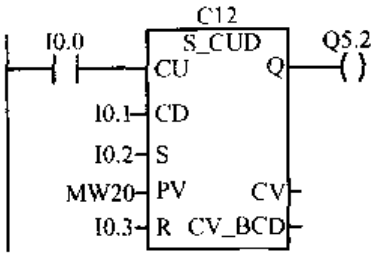


图 3-49 加减计数器

3.5 数据处理指令

S7 可以按字节、字和双字访问存储区。数据处理指令包括装入和传送指令、比较指令和数据类型转换指令。

累加器是 CPU 中的专用寄存器，数据的传送与变换一般通过累加器进行，而不是直接在存储区进行。S7-300 的 CPU 有两个 32 位的累加器，即累加器 1 和累加器 2。S7-400 的 CPU 有 4 个累加器，即累加器 1～累加器 4。累加器 1 是主累加器，其余的是辅助累加器。与累加器 1 进行运算的数据存储在累加器 2 中。

3.5.1 装入指令与传送指令

1. 装入指令与传送指令

装入(L，Load)指令和传送(T，Transfer)指令用于在存储区之间或存储区与过程输入、过程输出之间交换数据。

装入(L)指令将源操作数装入累加器 1，而累加器 1 原有的数据移入累加器 2。

装入指令可以对字节(8 位)、字(16 位)、双字(32 位)数据进行操作，数据长度小于 32 位时，数据在累加器中右对齐，即被操作的数据放在累加器的低端，其余的高位字节填 0。

传送(T)指令将累加器 1 中的内容写入目的存储区中，累加器 1 的内容不变。被复制的累加器中的字节数取决于目的地址中表示的数据长度。数据从累加器 1 传送到直接 I/O 区(外设输出区 PQ)的同时，也被传送到相应的过程映像输出区(Q)。

装入指令与传送指令见表 3-11。

表 3-11 装入指令与传送指令

指　令	说　明
L <地址>	装入指令，将数据装入累加器 1，累加器 1 原有的数据装入累加器 2
L STW	将状态字装入累加器 1
LAR1 AR2	将地址寄存器 2 的内容装入地址寄存器 1
LAR1 <D>	将 32 位双字指针<D>装入地址寄存器 1
LAR2 <D>	将 32 位双字指针<D>装入地址寄存器 2
LAR1	将累加器 1 的内容(32 位指针常数)装入地址寄存器 1
LAR2	将累加器 1 的内容(32 位指针常数)装入地址寄存器 2
T <地址>	传送指令，将累加器 1 的内容写入目的存储区，累加器 1 的内容不变
T STW	将累加器 1 中的内容传送到状态字
TAR1 AR2	将地址寄存器 1 的内容传送到地址寄存器 2
TAR1 <D>	将地址寄存器 1 的内容传送到 32 位指针
TAR2 <D>	将地址寄存器 2 的内容传送到 32 位指针
TAR1	将地址寄存器 1 的内容传送到累加器 1，累加器 1 中的内容保存到累加器 2
TAR2	将地址寄存器 2 的内容传送到累加器 1，累加器 1 中的内容保存到累加器 2
CAR	交换地址寄存器 1 和地址寄存器 2 中的数据

L，T 指令的执行与状态位无关，也不会影响到状态位。S7-300 不能用 L STW 指令装入状态字中的 FC、STA 和 OR 位。

可以不经过累加器 1，直接将操作数装入或传送出地址寄存器，或将两个地址寄存器的内

容直接交换，指令 TAR1 <D> 和 TAR2 <D>可能的目的区为双字 MD、LD、DBD 和 DID。

装入指令和传送指令有三种寻址方式：立即寻址、直接寻址和间接寻址。

2. 立即寻址的装入与传送指令

立即寻址的操作数直接在指令中，下面是使用立即寻址的装入指令的例子：

```
L    -35                    //将 16 位十进制常数(35 装入累加器 1 的低字 ACCU1-L
L    L#5                    //将 32 位常数 5 装入累加器 1
L    B#16#5A                //将 8 位十六进制常数装入累加器 1 最低的字节 ACCU1-LL
L    W#16#3E4F              //将 16 位十六进制常数装入累加器 1 的低字 ACCU1-L
L    DW#16#567A3DC8         //将 32 位十六进制常数装入累加器 1
L    2#0001_1001_1110_0010  //将 16 位二进制常数装入累加器 1 的低字 ACCU1-L
L    2.538000e+001          //将 32 位浮点数常数(25.38)装入累加器 1
L    'XY'                   //将两个字符装入累加器 1 的低字 ACCU1-L
L    'ABCD'                 //将 4 个字符装入累加器 1
L    TOD#12:30:3.0          //将 32 位实时时间常数装入累加器 1
L    D#2004-2-3             //将 16 位日期常数装入累加器 1 的低字 ACCU1-L
L    C#50                   //将 16 位计数器常数装入累加器 1 的低字 ACCU1-L
L    T#1M20S                //将 16 位定时器常数装入累加器 1 的低字 ACCU1-L
L    S5T#2S                 //将 16 位定时器常数装入累加器 1 的低字 ACCU1-L
L    P#M5.6                 //将指向 M5.6 的指针装入累加器 1
```

3. 直接寻址的装入与传送指令

直接寻址在指令中直接给出了存储器或寄存器的地址。下面是直接寻址的装入与传送指令的例子：

```
L    MB10                   //将 8 位存储器字节装入累加器 1 最低的字节 ACCU1-LL
L    DIW15                  //将 16 位背景数据字装入累加器 1 的低字 ACCU1-L
L    LD22                   //将 32 位局域数据双字装入累加器 1
T    QB10                   //将 ACCU1-LL 中的数据传送到过程映像输出字节 QB10
T    MW14                   //将 ACCU1-L 中的数据传送到存储器字 MW14
T    DBD2                   //将 ACCU1 中的数据传送到数据双字 DBD2
```

4. 间接寻址的装入与传送指令

在存储器间接寻址指令中，给出了一个存储器的地址，该存储器的内容是操作数所在存储单元的地址，该地址被称为地址指针，只有双字 MD、LD、DBD 和 DID 能作地址指针。

在寄存器间接寻址指令中，地址寄存器 AR1 或 AR2 的内容加上偏移量后形成地址指针，后者指向数值所在的存储单元。

下面是间接寻址的装入指令与传送指令的例子：

```
L   QB[LD 10]          //将输出字节 QB 装入 ACCU1-LL，其地址在数据双字 LD10 中
L   DBW[AR2, P#8.0]    //将 DBW 装入 ACCU1 L，其地址为 AR2 中的地址加上偏移量 P#8.0
T   W[AR1, P#5.0]      //累加器 1 的低字传送到字，其地址为 AR1 中的地址加上偏移量 P#5.0
                       //数据区的类型由 AR1 中的地址标识符决定
```

5. 装入时间值或计数值

可以用 L 指令将定时器字中的二进制剩余时间值装入累加器 1 的低字中，称为直接装载。也可以用 LC 指令以 BCD 码格式将剩余时间值装入累加器 1 的低字中。使用 LC 指令可以同时获得时间值和时基，时基与时间值相乘得到实际的定时剩余时间。

可以用 L 指令将二进制计数值装入累加器 1 的低字中，或用 LC 指令将 BCD 码格式的计数值装入累加器 1 的低字中。

```
L     T5      //将定时器 T5 中的二进制时间值装入累加器 1 的低字中
LC    T5      //将定时器 T5 中的 BCD 码格式的时间值装入累加器 1 低字中
L     C3      //将计数器 C3 中的二进制计数值装入累加器 1 的低字中
LC    C16     //将计数器 C16 中的 BCD 码格式的计数值装入累加器 1 的低字中
```

6. 地址寄存器的装入与传送指令

可以不经过累加器 1，直接将操作数装入到地址寄存器 AR1 和 AR2，或从 AR1 和 AR2 将数据传送出来。下面是应用实例：

```
LAR1    DBD20      //将数据双字 DBD20 中的指针装入 AR1
LAR2    LD180      //将局域数据双字 LD180 中的指针装入 AR2
LAR1    P#M10.2    //将带存储区标识符的 32 位指针常数装入 AR1
LAR2    P#24.0     //将不带存储区标识符 32 位指针常数装入 AR2
LAR1               //将累加器 1 的内容(32 位指针常数)装入 AR1
LAR2               //将累加器 1 的内容(32 位指针常数)装入 AR2
CAR                //交换 AR1 和 AR2 的内容
TAR1               //将 AR1 的数据传送到累加器 1，累加器 1 中的数据保存到累加器 2
TAR2               //将 AR2 的数据传送到累加器 1，累加器 1 中的数据保存到累加器 2
TAR1    DBD20      //AR1 中的内容传送到数据双字 DBD20
TAR2    MD24       //AR2 中的内容传送到存储器双字 MD24
```

7. 梯形图中的传送指令

在梯形图中，用指令框(Box)表示某些指令。指令框的输入端均在左边，输出端均在右边。梯形图中有一条提供“能流”的左侧垂直“电源”线，图 3-50 中 I0.1 的常开触点接通时，能流流到左边指令框的使能输入端 EN(Enable)，该输入端有能流时，指令框中的指令才能被执行。

如果指令框的 EN 输入有能流并且执行时无错误，则 ENO(Enable Output，使能输出)将能流传递给下一元件。如果执行过程中有错误，能流在出现错误的指令框终止。

ENO 可以与下一指令框的 EN 端相连，即几个指令框可以在一行中串联(见图 3-50)，只有前一个指令框被正确执行，后一个才能被执行。EN 和 ENO 的操作数均为能流，数据类型为 BOOL(布尔)型。

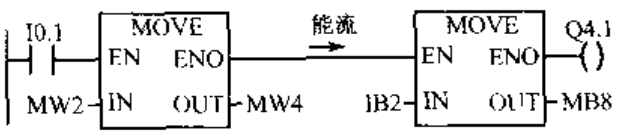


图 3-50　传送指令

方框传送(MOVE)指令为变量赋值，如果使能输入端 EN 为 1，执行传送操作，将输入 IN

指定的数据送入输出 OUT 指定的地址，并使 ENO 为 1，ENO 与 EN 的逻辑状态相同。

如果 EN 为 0，不进行传送操作，并使 ENO 为 0。

使用 MOVE 方框指令，能传送数据长度为 8 位、16 位或 32 位的基本数据类型（包括常数）。如果要传送用户定义的数据类型，例如数组或结构，必须使用系统功能 BLKMOV（SFC20）。下面是图 3-50 中左边的传送指令框对应的语句表。

```
      A       I1.0
      JNB     _001        //如果 I1.0 = 0，则跳转到标号_001 处
      L       MW2         //MW2 的值装入累加器 1 的低字
      T       MW4         //累加器 1 低字的内容传送到 MW4
      SET                 //将 RLO 置为 1
      SAVE                //将 RLO 保存到 BR 位
      CLR                 //将 RLO 置为 0
_001: A       BR
      ……
```

在梯形图的方框指令中，BR 位用于表明方框指令是否被正确执行：如果执行出现了错误，BR 位为 0，ENO 也为 0；如果功能被正确执行，BR 位为 1，ENO 也为 1。

3.5.2 比较指令

比较指令用于比较累加器 1 与累加器 2 中的数据大小，被比较的两个数的数据类型应该相同，数据类型可以是整数、双整数或浮点数（即实数）。如果比较的条件满足，则 RLO 为 1，否则为 0。状态字中的 CC0 和 CC1 位用来表示两个数的大于、小于和等于关系。

比较指令影响状态字，用指令测试状态字的有关位，可以得到更多的信息。

整数比较指令用来比较两个整数字的大小，指令助记符中用 I 表示整数；双整数比较指令用来比较两个双字的大小，指令助记符中用 D 表示双整数；浮点数比较指令用来比较两个浮点数的大小，指令助记符中用 R 表示浮点数。表 3-12 中的“?”可取 = =、<>、>、<、>= 和<=。

表 3-12 比较指令

语句表指令	梯形图中的符号	说 明
? I	CMP? I	比较累加器 2 和累加器 1 低字中的整数是否 = =，<>，>，<，>=，<=，如果条件满足，RLO=1
? D	CMP? D	比较累加器 2 和累加器 1 中的双整数是否 = =，<>，>，<，>=，<=，如果条件满足，RLO=1
? R	CMP? R	比较累加器 2 和累加器 1 中的浮点数是否 = =，<>，>，<，>=，<=，如果条件满足，RLO=1

下面是比较两个整数的例子：

```
A       I0.6
A(
L       MW2         //MW2 装入累加器 1
L       MW4         //MW4 装入累加器 1，MW2 装入累加器 2
```

```
<=I                                    //比较累加器 1 和累加器 2 的值
)
S          Q4.1                        //如果 I0.6 为 1 状态,且 MW2≤MW4, Q4.1 被置位为 1
```

下面是比较两个浮点数的例子：

```
L          MD4                         //MD4 中的浮点数装入累加器 1
L          2.345E+02                   //浮点数常数装入累加器 1,MD4 装入累加器 2
>R                                     //比较累加器 1 和累加器 2 的值
=          Q4.2                        //如果 MD4 > 2.345E+02,则 Q4.2 为 1
```

梯形图中的方框比较指令用来比较两个同类型的数，与语句表中的比较指令类似，可以比较整数(I)、双整数(D)和浮点数(R)。在使能输入信号为 1 时，比较 IN1 和 IN2 输入的两个操作数。方框比较指令在梯形图中相当于一个常开触点，可以与其他触点串联和并联。如果被比较的两个数满足指令指定的大于、等于、小于等条件，比较结果为“真”，等效触点闭合，指令框有能流流过。图 3-51 给出了部分方框比较指令，图 3-52 是上面的用语句表编写的整数比较程序对应的梯形图。如果 I0.6 和 I0.3 的常开触点闭合，且 MW2<=MW4，Q4.1 被置位为 1。

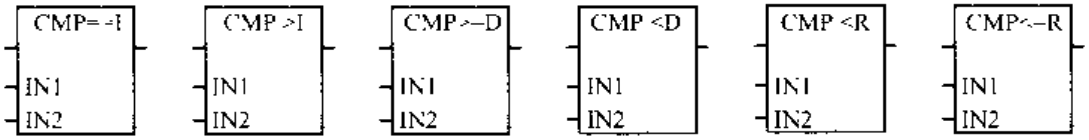


图 3-51　比较指令

梯形图中指令框的输入和输出均为 BOOL 变量，可以取 I、Q、M、L 和 D；被比较数 IN1 和 IN2 的数据长度与指令有关，可以取整数、双整数和浮点数。数据类型为 I、Q、M、L、D 或常数。

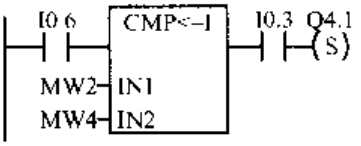


图 3-52　比较指令

3.5.3　数据转换指令

数据转换指令将累加器 1 中的数据进行数据类型的转换，转换的结果仍然在累加器 1 中。数据转换指令见表 3-13。

1. BCD 码的数据格式

在 STEP 7 中，16 位格式的 BCD 码(3 位 BCD 码)的数值范围为 -999～+999，32 位格式的 BCD 码(7 位 BCD 码)的数值范围为 -9 999 999～+9 999 999(见图 3-53 和图 3-54)。二进制整数和双整数都是以补码的形式存储和处理。

表 3-13　数据转换指令

语　句　表	梯　形　图	说　　明
BTI	BCD_I	将累加器 1 中的 3 位 BCD 码转换成整数
ITB	I_BCD	将累加器 1 中的整数转换成 3 位 BCD 码
BTD	BCD_DI	将累加器 1 中的 7 位 BCD 码转换成双整数
DTB	DI_BCD	将累加器 1 中的双整数转换成 7 位 BCD 码

（续）

语 句 表	梯 形 图	说 明
DTR	DI R	将累加器 1 中的双整数转换成浮点数
ITD	I_DI	将累加器 1 中的整数转换成双整数
RND	ROUND	将浮点数转换为四舍五入的双整数
RND+	CEIL	将浮点数转换为大于等于它的最小双整数
RND-	FLOOR	将浮点数转换为小于等于它的最大双整数
TRUNC	TRUNC	将浮点数转换为截位取整的双整数
CAW	-	交换累加器 1 低字中 2 B 的位置
CAD	-	交换累加器 1 中 4 B 的顺序

16 位格式的 BCD 码的第 0～11 位用来表示 3 位 BCD 码（见图 3-53），每 4 位二进制数用来表示 1 位 BCD 码，每位的数值范围为 2#0000～2#1001（对应于十进制数 0～9）。第 15 位用来表示 BCD 码的符号，正数为 0，负数为 1，第 12～14 位未用，一般取与符号位相同的数。图 3-53 中的 BCD 码为 - 862。

32 位格式的 BCD 码的第 0～27 位用来表示 7 位 BCD 码（见图 3-54），每 4 位二进制数用来表示 1 位 BCD 码。第 31 位是 BCD 码的符号位：正数为 0、负数为 1。第 28～30 位未用，一般取与符号位相同的数。

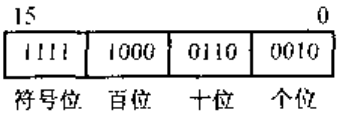


图 3-53　3 位 BCD 码的格式

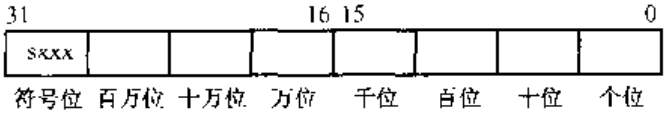


图 3-54　7 位 BCD 码的格式

2. BCD 码转换为整数

BTI 指令将累加器 1 低字中的 3 位 BCD 码转换为 16 位整数，结果仍在累加器 1 的低字中，累加器 1 的高字不变。

BTD 指令将累加器 1 中的 7 位 BCD 码转换为 32 位整数，结果仍在累加器 1 中。

在执行上述指令时，如果 BCD 码的某位为无效数据（2#1010～2#1111，对应的十进制数为 10～15），将得不到正确的转换结果，会出现“BCDF”错误。在这种情况下，CPU 通常将进入 STOP 状态，“BCD 转换错误”信息被写入诊断缓冲区。用户可以在组织块 OB121 中编写错误响应程序，以处理这种同步编程错误。

3. 整数转换为 BCD 码

ITB 指令将累加器 1 低字中的 16 位整数转换为 3 位 BCD 码，结果仍在累加器 1 的低字中，累加器 1 的高字不变。DTB 指令将累加器 1 中的 32 位双整数转换为 7 位 BCD 码，结果仍在累加器 1 中。

16 位整数的表示范围为 - 32 768～+ 32 767，而 3 位 BCD 码的表示范围为 - 999～+ 999。如果被转换的整数超出 BCD 码的允许范围，在累加器 1 的低字中得不到有效的转换结果，同时状态字中的溢出位（OV）和溢出保持位（OS）将被置 1。

在程序中，应根据状态位 OV 或 OS 判断转换后累加器 1 低字中的结果是否有效，以免造

成进一步的运算错误。在执行 DTB 指令时，也有类似问题需要注意。

下面是双整数转换为 BCD 码的例子：

```
        A     I0.2     //如果 I0.2 为 1
        L     MD10     //将 MD10 中的双整数装入累加器 1
        BTI            //将累加器 1 中的数据转换为 BCD 码，结果仍在累加器 1 中
        JO    OVER     //如果运算结果超出允许范围(OV=1)则跳转到标号 OVER 处
        T     MD20     //将转换结果传送到 MD20
        A     M4.0
        R     M4.0     //复位溢出标志
        JU    NEXT     //无条件跳转到标号 NEXT 处
OVER:   AN    M4.0
        S     M4.0     //置位溢出标志
NEXT:   ……
```

输入语句表中的标号时不能使用汉字的冒号。

4. 整数转换为双整数

ITD 指令将累加器 1 低字中的 16 位整数转换成 32 位双整数，结果仍在累加器 1 中，符号位被扩展。

以上的语句表转换指令，都有对应的梯形图方框指令(见图 3-55 和图 3-56，各指令的意义见表 3-13)。图 3-56 给出了一个数据转换指令的应用实例。图中的 EN 为转换允许输入端，ENO 为转换允许输出端。IN 为被转换数的输入端，OUT 为转换结果输出端。如果 I2.6 为 1，MW2 中的 16 位整数被装入累加器 1 的低字，转换为 32 位双整数后传送到 MD6。如果转换成功执行，Q4.4 为 1 状态。

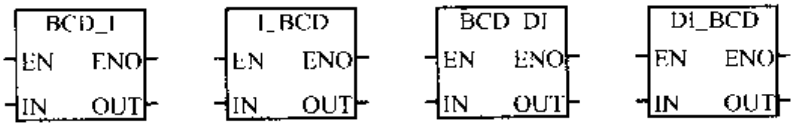


图 3-55 转换指令

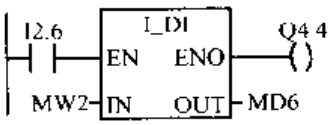


图 3-56 转换指令

下面是图 3-56 对应的语句表程序：

```
        A     I2.6
        JNB   _001     //如果 I2.6=0，则跳转到标号_001 处
        L     MW2      //MW2 的值装入累加器 1 的低字
        ITD            //累加器 1 低字中的整数转换为双整数
        T     MD6      //累加器 1 的内容传送到 MD6
        SET            //将 RLO 置为 1
        SAVE           //将 RLO 保存到 BR 位
        CLR            //将 RLO 置为 0
_001:   A     BR
        =     Q4.4
```

5. 交换累加器 1 中字节的位置

CAW(Change Byte Sequence in ACCU 1-L)指令交换累加器 1 低字中两个字节的位置，累

加器 1 的高字不变。

CAD(Change Byte Sequence in ACCU 1)指令交换累加器 1 中 4 个字节的顺序(见表 3-14)。

表 3-14 CAD 指令的功能

累加器的字节	ACCU1-HH	ACCU1-HL	ACCU1-LH	ACCU1-LL
CAD 指令交换前的内容	A	B	C	D
CAD 指令交换后的内容	D	C	B	A

6. 双整数与浮点数之间的转换

(1) 双整数转换为浮点数

DTR 指令将累加器 1 中的 32 位双整数转换为 32 位 IEEE 浮点数(实数),结果仍在累加器 1 中。因为 32 位双整数的精度比浮点数的高,指令将转换结果四舍五入。

(2) 浮点数转换为整数

RND(Round)指令将累加器 1 中的 IEEE 浮点数转换为 32 位双整数,结果仍在累加器 1 中,小数部分被舍去,得到的是最接近的整数(即四舍五入)。如果转换结果刚好在两个相邻的整数之间,则选择偶数为转换结果。

"RND+"指令将累加器 1 中的浮点数转换为大于等于该浮点数的最小双整数,结果仍在累加器 1 中。"RND-"指令将累加器 1 中的浮点数转换为小于等于该浮点数的最大双整数,结果仍在累加器 1 中。

TRUNC 指令将 32 位浮点数转换为 32 位带符号整数,结果仍在累加器 1 中。浮点数的整数部分被转换,小数部分被舍去。

上述的指令都是将累加器 1 中的浮点数转换为 32 位整数,因为化为整数的规则不同,在累加器 1 中得到的结果也不相同,表 3-15 给出了不同的取整格式的比较。

表 3-15 不同的取整格式举例

指 令	取整前	取整后	说 明
RND	+100.5 -100.5	+100 -100	将浮点数转换为四舍五入的双整数
RND+	+100.5 -100.5	+101 -100	将浮点数转换为大于等于它的最小双整数
RND-	+100.5 -100.5	+100 -101	将浮点数转换为小于等于它的最大双整数
TRUNC	+100.5 -100.5	+100 -100	将浮点数转换为截位取整的双整数

因为浮点数的数值范围远远大于 32 位整数,有的浮点数不能成功地转换为 32 位整数。如果被转换的浮点数超出了 32 整数的表示范围,在累加器 1 中得不到有效的结果,此时状态字中的 OV 和 OS 位被置 1。

【例 3-5】 将 101 in(英寸)转换为以 cm(厘米)为单位的整数,送到 MW0 中。

```
L     101        //将 16 位常数 101(65H)装入累加器 1
ITD              //转换为 32 位双整数
```

```
DTR                    //转换为浮点数 101.0
L       2.54           //浮点数常数 2.54 装入累加器 1,累加器 1 的内容装入累加器 2
*R                     //101.0 乘以 2.54,转换为 256.54 cm
RND                    //四舍五入转换为整数 257(101H)
T       MW30
```

图 3-57 中是梯形图中的双整数与浮点数之间的转换指令。其中的 DI_R 指令与语句表中的 DTR 指令相对应,CEIL 和 FLOOR 指令分别与语句表中的 RND+ 和 RND- 指令相对应。

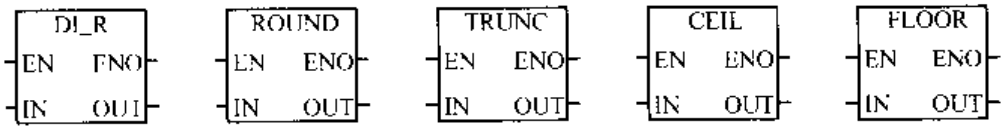


图 3-57 双整数与浮点数转换指令

7. 取反与求补指令

取反与求补指令如表 3-16 所示。整数取反(求反码)指令 INVI 将累加器 1 低字中的 16 位整数逐位取反,即各位二进制数由 0 变 1,由 1 变 0(见表 3-17),运算结果仍在累加器 1 的低字中。

表 3-16 取反与求补指令

语句表指令	梯形图指令	说 明
INVI	INV_I	求累加器 1 低字中的 16 位整数的反码
INVD	INV_DI	求累加器 1 中双整数的反码
NEGI	NEG_I	求累加器 1 低字中的 16 位整数的补码
NEGD	NEG_DI	求累加器 1 中双整数的补码
NEGR	NEG_R	将累加器 1 中的浮点数的符号位取反

双整数取反指令 INVD 将累加器 1 中的双整数逐位取反,结果仍在累加器 1 中。

整数求补指令 NEGI 将累加器 1 低字中的整数取反后再加 1,运算结果仍在累加器 1 的低字中。求补码相当于求一个数的相反数,即将该数乘以 -1。

表 3-17 取反与求补

内 容	累加器 1 的低字
变换前的数	0101 1101 0011 1000
取反的结果	1010 0010 1100 0111
求补的结果	1010 0010 1100 1000

双整数求补指令 NEGD 将累加器 1 中的双整数取反后再加 1,运算结果仍在累加器 1 中。

浮点数取反指令 NEGR 将将累加器 1 中的浮点数的符号位(第 31 位)取反,运算结果仍在累加器 1 中。下面的例子将 MD20 中的双整数求补后传送到 MD30 中:

```
L       MD20           //将 32 位双整数装入累加器 1
NEGD                   //求补
T       MD30           //运算结果传送到 MD30
```

图 3-58 是梯形图中的取反与求补指令。

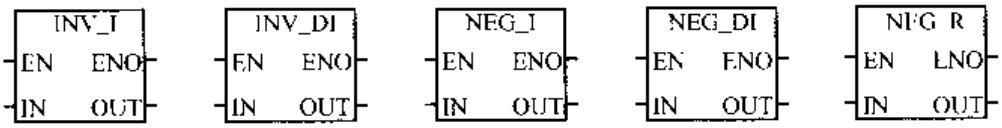


图 3-58　取反与求补指令

3.6　数学运算指令

数学运算指令包括整数运算指令、浮点数运算指令、循环移位指令和逻辑运算指令。

3.6.1　整数数学运算指令

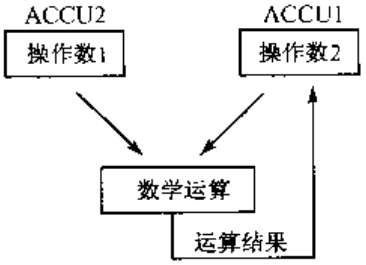


图 3-59　数学运算中的累加器

整数数学运算指令对累加器 1 和累加器 2 中的整数进行运算，运算结果保存在累加器 1 中(见图 3-59)。对于有 4 个累加器的 CPU，累加器 3 的内容复制到累加器 2，累加器 4 的内容传送到累加器 3，累加器 4 原有的内容保持不变。

1. 整数数学运算指令对状态字的影响

表 3-18 给出了有效的运算结果对状态字的影响。表 3-19 给出了无效的运算结果对状态字的影响。

表 3-18　有效的运算结果对状态字的影响

运算结果	CC1	CC0	OV	OS
运算结果 = 0	0	0	0	无影响
−32768 <= 16 位运算结果 < 0，或 −2 147 483 648 <= 32 位运算结果 < 0(负数)	0	1	0	无影响
32767 >= 16 位运算结果 > 0，或 2 147 483 647 >= 32 位运算结果 > 0(正数)	1	0	0	无影响

表 3-19　无效的运算结果对状态字的影响

运算结果	CC1	CC0	OV	OS
加法下溢出：16 位运算结果 = −65 536，或 32 位运算结果 = −4 294 967 296	0	0	1	1
乘法下溢出：16 位运算结果 < 32 767，或 32 位运算结果 < −2 147 483 648(负数)	0	1	1	1
加减法溢出：16 位运算结果 > 32 767，或 32 位运算结果 > 2 147 483 647(正数)	0	1	1	1
乘除法溢出：16 位运算结果 > 32 767，或 32 位运算结果 > 2 147 483 647(正数)	1	0	1	1
加减法下溢出：16 位运算结果 < −32 767，或 32 位运算结果 < −2 147 483 648(负数)	1	0	1	1
双字加法的运算结果 = −4 294 967 296	0	0	1	1
除法指令或 MOD 指令的除数为 0	1	1	1	1

2. 整数运算指令

整数加法指令“+I”将累加器 1、2 低字中的 16 位整数相加，16 位整数运算结果在累加器 1 的低字中。指令的执行与 RLO 无关，也不会对 RLO 产生影响。下面是整数加法运算的例子：

```
L     IW10        //IW10 的内容装入累加器 1 的低字
L     MW14        //累加器 1 的内容装入累加器 2,MW14 的值装入累加器 1 的低字
+I                //累加器 1 与累加器 2 低字的值相加,结果储存在累加器 1 的低字
T     DB1.DBW25   //累加器 1 低字中的运算结果传送到数据块 DB1 的 DBW25 中
```

整数减法指令“-I”将累加器 2 低字中的 16 位整数减去累加器 1 低字中的 16 位整数,16 位整数运算结果在累加器 1 的低字中。

整数乘法指令“*I”将累加器 1、2 低字中的 16 位整数相乘,32 位双整数运算结果在累加器 1 中。如果运算后状态字的 OV 和 OS 位均为 1,表示运算结果超出了 16 位整数允许的范围。

整数除法指令“/I”将累加器 2 低字中的 16 位整数除以累加器 1 低字中的 16 位整数,16 位商在累加器 1 的低字中,余数在累加器 1 的高字中。下面是整数除法运算的例子:

```
L     IW10        //IW10 的内容装入累加器 1 的低字
L     MW14        //累加器 1 的内容装入累加器 2,MW14 的值装入累加器 1 的低字
/I                //累加器 2 低字的值除以累加器 1 低字的值,结果存放在累加器 1 的低字
T     DB1.DBW2    //累加器 1 低字中的运算结果传送到数据块 DB1 的 DBW2 中
```

设 IW10 的值为 13,MW14 的值为 4,13 除以 4,指令执行后的商(3)存放在累加器 1 的低字中,余数(1)存放在累加器 1 的高字中。

整数数字运算指令见表 3-20。

表 3-20　整数数学运算指令

语句表	梯形图	描　述
+I	ADD_I	将累加器 1,2 低字中的整数相加,运算结果在累加器 1 的低字中
-I	SUB_I	累加器 2 低字中的整数减去累加器 1 低字中的整数,运算结果在累加器 1 的低字中
*I	MUL_I	将累加器 1,2 低字中的整数相乘,32 位双整数运算结果在累加器 1 中
/I	DIV_I	累加器 2 低字中的整数除以累加器 1 低字中的整数,商在累加器 1 的低字,余数在累加器 1 的高字
+		累加器的内容与 16 位或 32 位常数相加,运算结果在累加器 1 中
+D	ADD_DI	将累加器 1,2 中的双整数相加,双整数运算结果在累加器 1 中
-D	SUB_DI	累加器 2 中的双整数减去累加器 1 中的双整数,双整数运算结果在累加器 1 中
*D	MUL_DI	将累加器 1,2 中的双整数相乘,32 位双整数运算结果在累加器 1 中
/D	DIV_DI	累加器 2 中的双整数除以累加器 1 中的双整数,32 位商在累加器 1 中,余数被丢掉
MOD	MOD_DI	累加器 2 中的双整数除以累加器 1 中的双整数,32 位余数在累加器 1 中

加法指令“+”将累加器 1 低字中的 16 位整数与 16 位常数(-32 768～+32 767)相加,16 位整数运算结果在累加器 1 的低字中。也可以将累加器 1 中的 32 位整数与 32 位常数(-2 147 483 648～+2 147 483 647)相加,32 位整数运算结果在累加器 1 中。

双整数加法指令“+D”将两个 32 位双整数相加,32 位运算结果在累加器 1 中。

双整数减法指令“-D”将累加器 2 中的 32 位双整数减去累加器 1 中的 32 位双整数,32 位运算结果在累加器 1 中。

在语句表中输入程序时,不能使用中文的加号和减号。

【例 3-6】 编写实现整数双字运算 MD20 + MD24 - 200 的程序，运算结果送 MD28。

```
L     MD20          //MD20 的内容装入累加器 1
L     MD24          //累加器 1 的内容装入累加器 2，MD24 的值装入累加器 1
+D                  //累加器 1、2 的值相加，结果存放在累加器 1 中
+     L#  -200      //累加器 1 的值减去 200，结果储存在累加器 1 中
T     MD28          //累加器 1 的运算结果传送到 MD28 中
```

双整数乘法指令“*D”将累加器 1，2 中的 32 位双整数相乘，32 位运算结果在累加器 1 中。

双整数除法指令“/D”将累加器 2 中的双整数除以累加器 1 中的双整数，32 位商在累加器 1 中，余数被丢掉。可以用下面的 MOD 指令来求双整数除法的余数。

指令“MOD”将累加器 2 中的双整数除以累加器 1 中的双整数，32 位余数在累加器 1 中，可以用指令“/D”来求双整数除法的商。

图 3-60 和图 3-62 给出了梯形图中整数数学运算的方框指令，图中的 EN 为使能输入端，ENO 为使能输出端。IN1 和 IN2 为操作数输入端，OUT 为运算结果输出端。

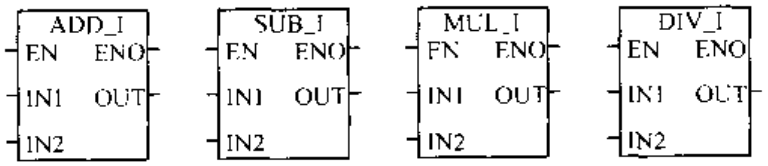


图 3-60 整数算术运算指令

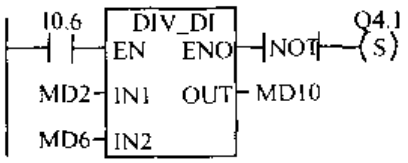


图 3-61 整数除法指令

图 3-62 双字算术运算指令

ADD、SUB、MUL 和 DIV 分表示加、减、乘、除，I 表示 16 位整数运算，DI 表示 32 位整数运算，MOD 为求余数的运算。

在加减法指令中，IN1 + IN2 = OUT，IN1 - IN2 = OUT。

在乘除法指令中，IN1 * IN2 = OUT，IN1 / IN2 = OUT。

在图 3-61 中，如果 I0.6 为 1，MD2 中的双整数除以 MD6 中的双整数，运算结果传送到 MD10。如果运算未能成功的完成，则状态字的 OV 和 OS 位为 1，且使 ENO 为 0，Q4.1 为 1 状态；若运算成功地完成，则状态字的 OV 被清 0，OS 位保持原状态不变，且使 RLO 为 1。下面是图 3-61 对应的语句表程序：

```
A(
A     I0.6
JNB   _002
L     MD2           //MD2 的内容装入累加器 1
L     MD6           //累加器 1 的内容装入累加器 2，MD6 的值装入累加器 1
```

```
      /D                    //累加器2的值除以累加器1的值,商在累加器1中
      T     MD10            //累加器1的运算结果传送到MD10
      AN    OV
      SAVE
      CLR
002:  A     BR
      )
      NOT
      S     Q4.1
```

【例 3-7】 压力变送器的量程为 0～10MPa,输出信号为 4～20mA,S7-300 的模拟量输入模块的量程为 4～20mA,转换后的数字量为 0～27 648,设转换后的数字为 N,试求以 kPa 为单位的压力值。

解:0～10MPa(0～10 000kPa)对应于转换后的数字 0～27 648,转换公式为

$$P=(10\ 000\times N)/27\ 648\ \text{kPa} \tag{3-1}$$

值得注意的是在运算时一定要先乘后除,否则会损失原始数据的精度。假设 A/D 转换后的数据 N 在 MD6 中,以 kPa 为单位的运算结果在 MW10 中。图 3-63 是实现式(3-1)中的运算的梯形图程序。

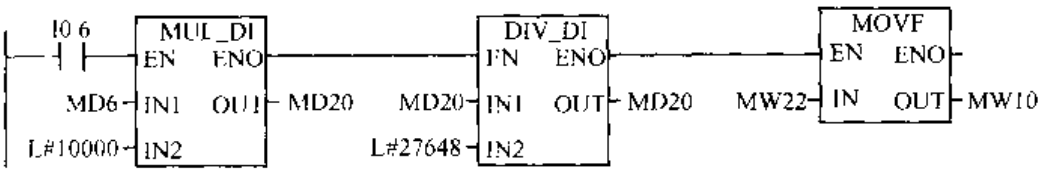


图 3-63 算术运算指令

如果某一方框指令的运算结果超出了整数运算指令的允许范围,状态位 OV 和 OS 将为 1,使能输入 ENO 为 0,不会执行在该方框指令右边的指令。

值得注意的是语句表中的指令"*I"的运算结果为 32 位整数,梯形图中的 MUL_I 指令的运算结果为 16 位整数。A/D 转换后的最大数字为 27 648,乘以 10 000 以后可能超过 16 位整数的允许范围,所以需要使用双字乘法指令 MUL_DI。双字除法指令 DIV_DI 的运算结果为双字,但是由式(3-1)可知运算结果实际上不会超过 16 位正整数的最大值(32 767),所以可以用 MOVE 指令将 MD20 的低字 MW22 中的 16 位整数运算结果传送到 MW10 中。

3.6.2 浮点数数学运算指令

1. 浮点数数学运算指令概述

浮点数(实数)数学运算指令对累加器 1 和累加器 2 中的 32 位 IEEE 格式的浮点数进行运算,运算结果在累加器 1 中,在双累加器的 CPU 中,浮点数数学运算不会改变累加器 2 的值。对于有 4 个累加器的 CPU,累加器 3 的内容复制到累加器 2,累加器 4 的内容传送到累加器 3,累加器 4 原有的内容保持不变。

32 位 IEEE 格式的浮点数的数据类型为 REAL,浮点数数学指令如表 3-21 所示。

表 3-21　浮点数数学指令

语句表	梯形图	描　述
+R	ADD_R	将累加器 1,2 中的浮点数相加,浮点数运算结果在累加器 1 中
-R	SUB_R	累加器 2 中的浮点数减去累加器 1 中的浮点数,浮点数运算结果在累加器 1 中
*R	MUL_R	将累加器 1,2 中的浮点数相乘,浮点数乘积在累加器 1 中
/R	DIV_R	累加器 2 中的浮点数除以累加器 1 中的浮点数,浮点数商在累加器 1 中,余数被丢掉
ABS	ABS	对累加器 1 中的浮点数取绝对值
SQR	SQR	求浮点数的平方
SQRT	SQRT	求浮点数的平方根
EXP	EXP	求浮点数的自然指数
LN	LN	求浮点数的自然对数
SIN	SIN	求浮点数的正弦函数
COS	COS	求浮点数的余弦函数
TAN	TAN	求浮点数的正切函数
ASIN	ASIN	求浮点数的反正弦函数
ACOS	ACOS	求浮点数的反余弦函数
ATAN	ATAN	求浮点数的反正切函数

2. 浮点数数学运算指令对状态字的影响

表 3-22 给出了运算结果在有效范围内时对状态字的影响,表 3-23 给出了运算结果在无效范围内时对状态字的影响。

表 3-22　运算结果在有效范围内对状态字的影响

运算结果	CC1	CC0	OV	OS
运算结果为+0 或(0(Null)	0	0	0	无影响
(3.402 823E+38 <运算结果< (1.175 494E-38(负数)	0	1	0	无影响
+1.175 494E(38 <运算结果< 3.402 824E+38(正数)	1	0	0	无影响

3. 浮点数基本数学指令

浮点数(即实数)加法指令"+R"将累加器 1、2 中的 32 位 IEEE 浮点数相加,运算结果在累加器 1 中。指令的执行与 RLO 无关,也不会对 RLO 产生影响。作为指令功能的一部分,指令执行后会影响状态字中的 CC1、CC2、OV 和 OS 位(见表 3-22 和表 3-23)。下面是浮点数加法运算的例子:

表 3-23　运算结果在无效范围内对状态字的影响

运算结果	CC1	CC0	OV	OS
负数下溢出:(1.175 494E(38 <运算结果< -1.401 298E-45	0	0	1	1
正数下溢出:+1.401 298E(45 <运算结果< +1.175 494E-38	0	0	1	1
溢出:运算结果<(3.402 823E+38(负数)	0	1	1	1
溢出:运算结果> 3.402 823E+38(正数)	1	0	1	1
不是有效的浮点数或非法的指令(输入值超出允许范围)	1	1	1	1

```
OPN     DB10      //打开数据块 DB10
L       MD10      //MD10 的内容装入累加器 1
L       MD14      //累加器 1 的内容装入累加器 2,MD14 的值装入累加器 1
+R                //累加器 1 和累加器 2 的值相加,运算结果在累加器 1
T       DBD25     //累加器 1 中的运算结果传送到数据块 DB10 的数据双字 DBD25 中
```

浮点数减法指令“-R”将累加器 2 中的 32 位浮点数减去累加器 1 中的 32 位浮点数,运算结果在累加器 1 中。

浮点数乘法指令“*R”将累加器 1,2 中的 32 位浮点数相乘,运算结果在累加器 1 中。

浮点数除法指令“/R”用累加器 2 中的 32 位浮点数除以累加器 1 中的 32 位浮点数,运算结果(商)在累加器 1 中。

ABS(Absolute Value)指令求累加器 1 中的 32 位 IEEE 浮点数的绝对值,运算结果在累加器 1 中。下面是求浮点数的绝对值的例子:

```
L       MD8       //将 MD8 的内容装入累加器 1,例如 MD8=(1.5E+02
ABS               //求累加器 1 中的浮点数的绝对值,结果仍在累加器 1 中
T       MD14      //累加器 1 中的运算结果(1.5E+02)传送到 MD14
```

4. 梯形图中的浮点数数学运算指令

图 3-64 和图 3-65 给出了梯形图中浮点数算术运算的方框指令,图中的 EN 为使能输入端,ENO 为使能输出端。IN1 和 IN2 为操作数输入端,OUT 为运算结果输出端。

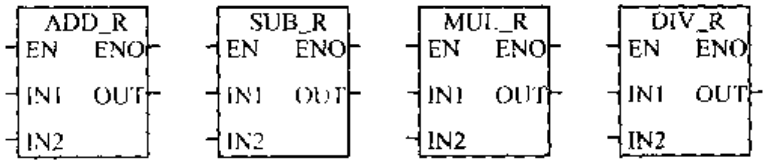


图 3-64 浮点数算术运算指令

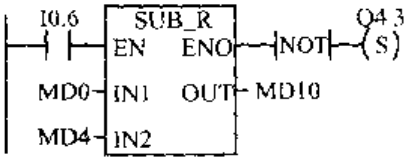


图 3-65 浮点数减法指令

ADD_R,SUB_R,MUL_R 和 DIV_R 分别为浮点数加、减、乘、除运算指令。

在加减法指令中,IN1 + IN2 = OUT,IN1 - IN2 = OUT。

在乘除法指令中,IN1 * IN2 = OUT,IN1 / IN2 = OUT。

在图 3-65 中,如果 I0.6 为 1,MD0 中的浮点数减去 MD4 中的浮点数,运算结果传送到 MD8。如果运算结果超出浮点数的允许范围,或指令没有成功地执行,则 ENO 为 0,Q4.3 被置位。下面是图 3-65 对应的语句表程序:

```
A(
A       I0.6
JNB     _001
L       MD0       //MD0 的内容装入累加器 1
L       MD4       //累加器 1 的内容装入累加器 2,MD4 的值装入累加器 1
-R                //累加器 2 的值减去累加器 1 的值,商在累加器 1 中
T       MD10      //累加器 1 中的运算结果传送到 MD10
AN      OV
SAVE
```

```
      CLR
_001:       A       BR
      )
      NOT
      S       Q4.3
```

5. **扩展的浮点数数学指令**

(1) 浮点数平方指令与平方根指令

浮点数平方指令 SQR(Generate the Square)求累加器 1 中的 32 位浮点数的平方,得到的浮点数运算结果在累加器 1 中。下面的指令用来求 DB17.DBD0 的平方,如果运算没有出错,结果存在 DB17.DBD4 中。

```
      OPN     DB17        //打开数据块 DB17
      L       DBD0        //数据块 DB17 的 DBD0 中的浮点数装入累加器 1
      SQR                 //求累加器 1 中的浮点数的平方,运算结果在累加器 1 中
      AN      OV          //如果运算时没有出错
      JC      OK          //跳转到标号 OK 处
      BEU                 //如果运算时出错,功能块无条件结束
OK:   T       DBD4        //累加器 1 中的运算结果传送到数据块 DB17 的 DBD4 中
```

浮点数开平方指令 SQRT(Generate the Square Root)将累加器 1 中的 32 位浮点数开平方,得到的浮点数运算结果在累加器 1 中。输入值应大于等于 0,运算结果为正数或 0。

(2) 浮点数自然指数指令

浮点数自然指数指令 EXP(Natural Exponential)求累加器 1 中的 32 位浮点数的自然指数(底数 e= 2.71828),得到的运算结果在累加器 1 中。

(3) 浮点数自然对数指令

浮点数自然对数指令 LN(Natural Logarithm)求累加器 1 中的 32 位浮点数的自然对数,得到的运算结果在累加器 1 中。

求以 10 为底的对数时,需将自然对数值除以 2.302585(10 的自然对数值)。例如

$$\lg 100 = \ln 100 / 2.302585 = 4.605170 / 2.302585 = 2$$

【例 3-8】 用浮点数对数指令和指数指令求 5 的立方。计算公式为

$$5^3 = EXP(3 * LN(5)) = 125$$

下面是对应的程序

```
L       L#5
DTR
LN
L       3.0
*R
EXP
RND
T       MW40
```

RND 是将浮点数四舍五入转换为整数的指令。

(4) 浮点数三角函数指令

浮点数正弦函数指令 SIN 求累加器 1 中的浮点数的正弦值，浮点数余弦函数指令 COS 求累加器 1 中的浮点数的余弦值，浮点数正切函数指令 TAN 求累加器 1 中的浮点数的正切值，得到的浮点数运算结果在累加器 1 中。

输入值如果是以角度为单位的浮点数，求三角函数前应先将角度值乘以 π/180，转换为弧度值。

(5) 浮点数反三角函数指令

浮点数反正弦函数指令 ASIN 求累加器 1 中的浮点数的正弦值，浮点数反余弦函数指令 ACOS 求累加器 1 中的浮点数的反余弦值，得到的以弧度为单位的浮点数运算结果在累加器 1 中。 $-1 \leqslant$ 输入值 $\leqslant +1$，$-\pi/2 \leqslant$ 运算结果 $\leqslant +\pi/2$。

浮点数反正切函数指令 ATAN 求累加器 1 中的浮点数的反正切值，得到的以弧度为单位的浮点数运算结果在累加器 1 中。$-\pi/2 \leqslant$ 运算结果 $\leqslant +\pi/2$。

6. 梯形图中扩展的浮点数数学指令

图 3-66 给出了梯形图中扩展的浮点数数学运算的方框指令，图中的 EN 为使能输入端，ENO 为使能输出端，IN 为操作数输入端，OUT 为运算结果输出端。方框指令中的指令助记符与语句表中的相同。

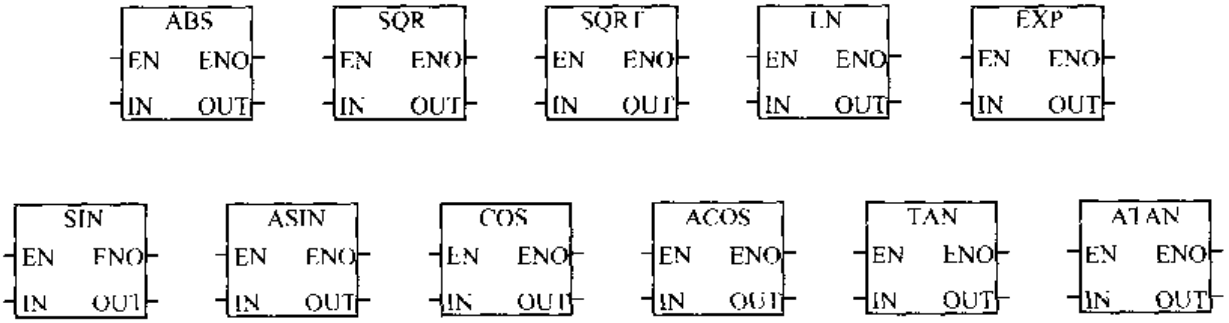


图 3-66 扩展的浮点数数学运算指令

3.6.3 移位指令

1. 移位指令概述

移位指令将累加器 1 的低字或累加器 1 的全部内容左移或右移若干位(见表 3-24)。左移 n 位相当于乘以 2^n，例如将十进制数 3 对应的二进制数 2#11 左移 2 位，相当于乘以 4，左移后得到的二进制数 2#1100 对应于十进制数 12。右移 n 位相当于除以 2^n，例如将十进制数 24 对应的二进制数 2#11000 右移 3 位，相当于除以 8，右移后得到的二进制数 2#11 对应于十进制数 3。

表 3-24 移位指令

语句表	梯形图	描述
SSI	SHR_I	将累加器 1 低字中的有符号整数逐位右移，空出的位添上与符号位相同的数
SSD	SHR_DI	将累加器 1 中的有符号双整数逐位右移，空出的位添上与符号位相同的数
SLW	SHL_W	将累加器 1 低字中的 16 位字逐位左移，空出的位添 0
SRW	SHR_W	将累加器 1 低字中的 16 位字逐位右移，空出的位添 0
SLD	SHL_DW	将累加器 1 中的双字逐位左移，空出的位添 0
SRD	SHR_DW	将累加器 1 中的双字逐位右移，空出的位添 0

无符号数(字或双字)移位后空出来的位填以0,有符号数(整数或双整数)右移后空出来的位填以符号位对应的二进制数(正数的符号位为0,负数的符号位为1)。最后移出的位被装入状态字的CC1位。

移位的位数可以用下面的两种方法来指定:

(1) 用指令中的参数<number>来指定移位位数,16位移位指令的允许值为0~15,32位移位指令的允许值为0~32。如果<number>大于0,状态字的CC0和OV被清0;如果<number>等于0,移位指令被当作NOP(空操作)指令来处理。

(2) 指令没有参数<number>,移位位数放在累加器2的最低字节中,移位位数的允许值为0~255。如果移位位数等于0,移位指令被当作NOP(空操作)指令来处理。

2. 有符号整数右移

有符号整数右移指令SSI <number> (Shift Right With Sign Integer)将累加器1低字中的内容逐位右移,移位后高端空出的位用符号位(第15位)来填充,即负数移位时用1来填充,正数移位时用0来填充。最后移出的位装入状态字中的CC1位。

下面的有符号数右移指令用指令中的<number>来指定移位位数:

```
L     MW4     //将MW4的内容装入累加器1的低字
SSI   6       //累加器1低字中的有符号数右移6位,结果仍在累加器1的低字中
T     MW8     //累加器1低字中的运算结果传送到MW8中
```

表3-25给出了移位前后累加器1中的二进制数的值,应注意两个问题:

(1) 累加器低字中的数字为负数,右移位后低字的高位添了6个1。

(2) 移位前后累加器1的高字没有变化。

表3-25 整数右移6位前后的数据

内 容	累加器1的高字	累加器1的低字
移位前	0101 1111 0110 0100	1001 1101 0011 1011
右移6位后	0101 1111 0110 0100	1111 1110 0111 0100

在下面的例子中,移位位数(3)放在累加器2的最低字节中。移位位数的允许值为0~255。移位位数>16时,总是产生同样的结果,即ACCU1-L = 16#0000, CC 1 = 0, 或ACCU1-L = 16#FFFF, CC 1 = 1。换句话说,因为移位次数超过被移位数的位数,移位后被移位的数的各位全部变成了符号位。如果0<移位位数≤16,状态字的CC0和OV被清0;移位位数等于0时移位指令被当作NOP(空操作)指令来处理。

下面是移位位数在累加器2的最低字节中的例子:

```
L     +3      //将+3装入累加器1
L     MW20    //将累加器1的内容装入累加器2,MW20的内容装入累加器1
SSI           //累加器1低字中的有符号数右移,移位位数在累加器2的最低字节中
              //右移3位,空出来的位用累加器1低字的符号位来填充
JP    NEXT    //如果最后移入CC1的位为1,跳转到标号NEXT处
```

3. 有符号双整数右移

有符号双整数右移指令SSD <number>(Shift Sign Double Integer)将累加器1中的内容

逐位右移，移位后高端空出的位用符号位(第 31 位)来填充，即负数移位时用 1 来填充，正数移位时用 0 来填充。最后移出的位装入状态字中的 CC1 位。移位位数 number 的允许值为 0～32。移位位数也可以放在累加器 2 的最低字节中，允许值为 0～255，这时 SSD 指令不带移位位数 number。移位位数＞32 时，移位后累加器 1 所有的位和 CC1 取符号位的值。

4. 16 位字移位指令

16 位字左移指令 SLW ＜number＞ (Shift Left Word)将累加器 1 低字中的内容逐位左移，移位后低端空出来的位用 0 来填充，最后移出的位装入状态字中的 CC1 位。

16 位字右移指令 SRW ＜number＞ (Shift Right Word)将累加器 1 低字中的内容逐位右移，移位后高端空出的位用 0 来填充，最后移出的位装入状态字中的 CC1 位。

移位位数可以用指令中的＜number＞来设置，设置的方法与 SSI 指令的相同。

表 3-26 给出了字左移 5 位移位前后累加器 1 中的二进制数的值，注意移位前后累加器 1 的高字没有变化。

表 3-26　字右移 5 位移位前后的数据

内　　容	累加器 1 的高字	累加器 1 的低字
移位前	0101 1111 0110 0100	0101 1101 0011 1011
右移 5 位后	0101 1111 0110 0100	1010 0111 0110 0000

表 3-27 给出了字右移 6 位，移位前后累加器 1 中的二进制数的值，注意移位前后累加器 1 的高字没有变化。

表 3-27　字右移 6 位移位前后的数据

内　　容	累加器 1 的高字	累加器 1 的低字
移位前	0101 1111 0110 0100	0101 1101 0011 1011
右移 6 位后	0101 1111 0110 0100	0000 0001 0111 0100

移位位数可以放在累加器 2 的最低字节中，允许值为 0～255。移位位数＞16 时，因为数据中各位被全部移出去后添上了 0，指令执行后 ACCU1-L、CC1、CC0 和 OV 均为 0。如果 0＜移位位数≤16，状态字的 CC0 和 OV 被清 0；移位位数等于 0 时移位指令被当作 NOP(空操作)指令来处理。

5. 双字移位指令

双字左移指令 SLD ＜number＞ (Shift Left Double Word)将累加器 1 中的内容逐位左移，移位后低端空出的位用 0 来填充。最后移出的位装入状态字中的 CC1 位。

双字右移指令 SRD ＜number＞ (Shift Right Double Word)将累加器 1 中的内容逐位右移，移位后高端空出的位用 0 来填充。最后移出的位装入状态字中的 CC1 位。

移位位数可以用指令中的参数 number(0～15)来设置，也可以放在累加器 2 的最低字节中，允许值为 0～255。移位位数＞32 时，指令执行后 ACCU1-L、CC1、CC0 和 OV 均为 0。如果 0＜移位位数≤32，状态字的 CC0 和 OV 被清 0；移位位数等于 0 时移位指令被当作 NOP(空操作)指令来处理。

6. 梯形图中的移位指令

图 3-67 是有符号整数右移指令 SHR_I(Shift Right Integer)的方框指令，图 3-68 是梯形图中移位操作的方框指令。方框中的 EN 为使能输入端，ENO 为使能输出端，IN 为操作数输入端，OUT 为运算结果输出端，N 输入端指定移位的位数。IN 和 OUT 为 16 位整数，N 为 WORD 变量。

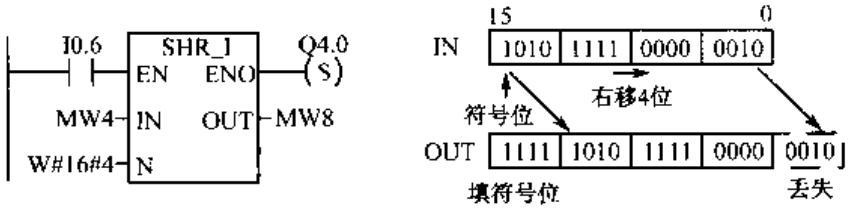


图 3-67 有符号数右移指令

下面是图 3-67 中的梯形图对应的语句表程序，移位位数为 4，图 3-67 给出了移位的效果。

```
      A     I0.6
      JNB   _001
      L     W#16#4    //移位位数 4 装入累加器 1 的最低字节
      L     MW4       //累加器 1 的内容装入累加器 2，MW4 的值装入累加器 1 的低字
      SSI             //累加器 1 低字的有符号整数右移 4 位
      T     MW8       //累加器 1 低字的运算结果传送到 MW8
      SET
      SAVE
      CLR
_001: A     BR
      S     Q4.0
```

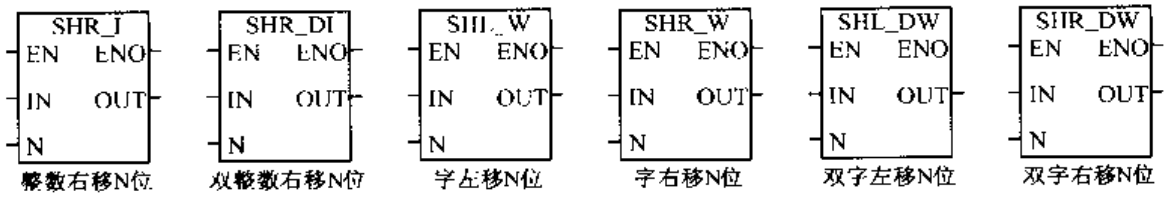


图 3-68 移位指令

3.6.4 循环移位指令

1. 循环移位指令概述

循环移位指令将累加器 1 的整个内容逐位循环左移或循环右移若干位(见表 3-28)，即从累加器 1 移出来的位又送回累加器 1 另一端空出来的位。

循环移位的位数可以用指令中的参数＜number＞来指定，移位位数也可以放在累加器 2 的最低字节中。移位位数等于 0 时，循环移位指令被当作 NOP(空操作)指令来处理。

表 3-28 循环移位指令

语 句 表	梯 形 图	描 述
RLD	ROL_DW	累加器 1 中的双字循环左移
RRD	ROR_DW	累加器 1 中的双字循环右移
RLDA	–	累加器 1 中的双字通过 CC1 循环左移
RRDA	–	累加器 1 中的双字通过 CC1 循环右移

2. 累加器 1 中的双字循环移位指令

双字循环左移指令 RLD ＜number＞ (Rotate Left Double Word)将累加器 1 的内容逐位左移,移出来的最高位返回空出来的最低位,最后移出的位装入状态字中的 CC1 位。

双字循环右移指令 RRD ＜number＞ (Rotate Right Double Word)将累加器 1 的内容逐位右移,移出来的最低位返回空出来的最高位,最后移出的位装入状态字中的 CC1 位。

循环移位的位数可以用指令中的无符号整数＜number＞来指定,移位位数的允许值为 0～32。循环移位的位数也可以放在累加器 2 的最低字节中,允许值为 0～255。如果移位位数大于 0,状态字的 CC0 和 OV 被清 0;如果等于 0,移位指令被当作 NOP(空操作)指令来处理。

表 3-29 给出了双字循环左移 4 位,移位前后累加器 1 中的二进制数的值。

表 3-29 循环左移 4 位前后累加器中的数据

内 容	累加器 1 的高字	累加器 1 的低字
移位前	0101 1111 0110 0100	0101 1101 0011 1011
右移 4 位后	1111 0110 0100 0101	1101 0011 1011 0101

3. 累加器 1 中的双字通过 CC1 循环移位指令

双字通过 CC1 循环左移指令 RLDA＜number＞(Rotate Left Double Word via CC1)将累加器 1 中的整个内容逐位左移 1 位,移出来的最高位装入 CC1,CC1 原有的内容装入累加器 1 的最低位。

双字通过 CC1 循环右移指令 RRDA ＜number＞(Rotate Right Double Word via CC1)将累加器 1 中的整个内容逐位右移 1 位,移出来的最低位装入 CC1,CC1 原有的内容装入累加器 1 的最高位。

RLDA 和 RRDA 实际上是一种 33 位(累加器 1 的 32 位加状态字的 CC1)的循环移位,累加器中移出来的位装入状态字的 CC1 位,状态字的 CC0 和 OV 被复位为 0。

表 3-30 给出了循环左移 1 位,移位前后累加器 1 中的二进制数的值。表中的 x = 0 或 1,是 CC1 在循环移位之前的值。

表 3-30 通过 CC1 循环左移 1 位前后累加器中的数据

内 容	CC1	累加器 1 的高字	累加器 1 的低字
移位前	X	0101 1111 0110 0100	0101 1101 0011 1011
左移后	0	1011 1110 1100 1000	1011 1010 0111 011X

4. 梯形图中的循环移位指令

图 3-69 是双字循环左移移位的方框指令,方框中的 EN 为使能输入端,ENO 为使能输出

端。IN 为操作数输入端,OUT 为运算结果输出端,N 输入端指定的 MW4 中是移位的位数。IN 和 OUT 为双字变量,N 为 16 位字变量。

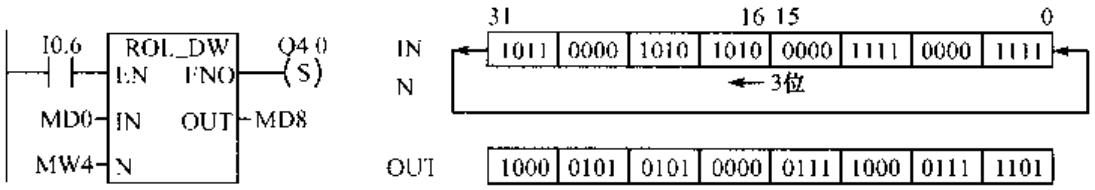


图 3-69 循环移位指令

假设 MW4 中的数字为 3,当 I0.6 为 1 时,MD0 中的双整数被循环左移 3 位,移位后的结果写入 MD8。如果循环移位指令被成功地执行,Q4.0 被置位为 1。下面是图 3-69 中的梯形图对应的语句表程序:

```
      A     I0.6
      JNB   _001
      L     MW4      //MW4 的内容装入累加器 1 的低字
      L     MD0      //累加器 1 的内容装入累加器 2,MD0 的值装入累加器 1
      RLD            //累加器 1 中的双整数循环左移
      T     MD8      //累加器 1 的运算结果传送到 MD8
      SET
      SAVE
      CLR
_001: A     BR
      S     Q4.0
```

3.6.5 字逻辑运算指令

字逻辑运算指令(见表 3-31)对两个 16 位字或 32 位双字逐位进行逻辑运算,一个操作数在累加器 1 中,另一个操作数在累加器 2 中,或在指令中用立即数的形式给出。字逻辑运算的结果在累加器 1 的低字中,双字逻辑运算的结果在累加器 1 中。

表 3-31 字逻辑运算指令

语 句 表	梯 形 图	描 述
AW	WAND_W	字与
OW	WOR_W	字或
XOW	WXOR_W	字异或
AD	WAND_DW	双字与
OD	WOR_DW	双字或
XOD	WXOR_DW	双字异或

如果字逻辑运算的结果为 0,状态字的 CC1 位为 1,反之为 0。在任何情况下,状态字中的 CC0 和 OV 位都被清 0。

1. 字逻辑运算指令

字与指令 AW(And Word)将两个输入字的对应位相“与”,两个输入字的同一位均为 1 时,运算结果的对应位为 1,否则为 0(见表 3-32)。

表 3-32 字逻辑运算的结果

位	15 0
逻辑运算前累加器 1 的低字	0101 1001 0011 1011
逻辑运算前累加器 2 的低字或常数	1111 0110 1011 0101
“与”运算后累加器 1 的低字	0101 0000 0011 0001
“或”运算后累加器 1 的低字	1111 1111 1011 1111
“异或”运算后累加器 1 低字	1010 1111 1000 1110

字或指令 OW(Or Word)将两个输入字的对应位相“或”,两个输入字的同一位均为 0 时,运算结果的对应位为 0,否则为 1。

字异或指令 XOW(Exclusive OR Word)将两个输入字的对应位相“异或”。两个输入字的同一位不同时,运算结果的对应位为 1,否则为 0。

字逻辑运算的一个输入字在累加器 1 的低字中,另一个输入字在累加器 2 的低字中,或者是指令中的常数,得到的一个字的结果装入累加器 1 的低字。下面是用语句表编写的实现字逻辑或运算的程序,该操作将 QW10 中的低 4 位置为 1,其余各位保持不变。

```
L   QW10          //QW10 的内容装入累加器 1 的低字
L   W#16#000F     //累加器 1 的内容装入累加器 2,常数 W#16#000F 装入累加器 1 的低字
OW                //累加器 1 低字的内容与 W#16#000F 逐位相或,结果在累加器 1 的低字中
T   QW10          //累加器 1 低字中的运算结果传送到 QW10 中
```

下面是累加器 1 低字的内容与字逻辑与指令中的立即数逐位作“与”运算的程序,该操作将 IW20 中高 4 位去掉,只保留低 12 位的数据。

```
L   IW20          //IW20 的内容装入累加器 1 的低字
AWW#16#0FFF       //累加器 1 低字的内容与 W#16#0FFF 逐位相与,结果在累加器 1 的低字中
JP  NEXT          //如果运算结果非 0(CC1=1),跳转到标号 NEXT 处
```

2. 双字逻辑运算指令

双字与指令 AD (AND Double Word)将两个输入双字的对应位相“与”,两个输入双字的同一位均为 1 时,运算结果的对应位为 1,否则为 0。

双字或指令 OD(Or Double Word)将两个输入双字的对应位相“或”,两个输入双字的同一位均为 0 时,运算结果的对应位为 0,否则为 1。

双字异或指令 XOD(Exclusive OR double Word)将两个输入双字的对应位相“异或”。两个输入双字的同一位不同时,运算结果的对应位为 1,否则为 0。

3. 梯形图中的字逻辑运算方框指令

图 3-70 是梯形图中的字逻辑运算指令,方框中的 EN 为使能输入端,ENO 为使能输出端。IN1 和 IN2 为两个操作数的输入端,OUT 为运算结果输出端,各方框指令的意义见表 3-31。

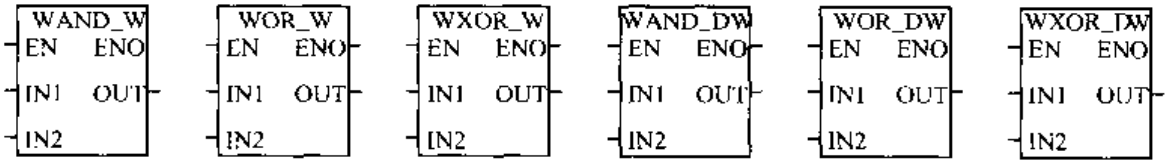


图 3-70　字逻辑运算指令

4. 立即读入与立即写出

图 3-71 中的字逻辑与指令 WAND_W 用来立即(Immediate)读取 I0.1 和 I0.2。通过字逻辑与指令和访问外设输入区 PI,CPU 直接读取输入模块上的物理输入点,而不是读取输入映像存储区(I)中的数据。因为输入映像存储区的数据是在一次扫描循环的开始将输入信号成批读入的,从外设输入区读取的数据更为及时。外设输入区只能按字节、字和双字来读取,不能按位(bit)来立即读取单个数字量输入。因此在读出包括 16 个输入点的外设输入字 PIW0 之后,用字与指令保留 I0.1 和 I0.2 的值,将它们存放在 MW8 中。MB9 是 MW8 中的低字节,M9.1 和 M9.2 对应于输入信号 I0.1 和 I0.2。

图 3-72 中用 MOVE 指令将过程映像输出 Q5.0 新的值通过外设输出 PQB5 立即写到对应的输出模块。PQ 只能按字节、字和双字来写出,不能按位(bit)立即写单个数字量输出。

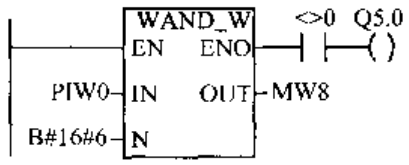


图 3-71　立即读入

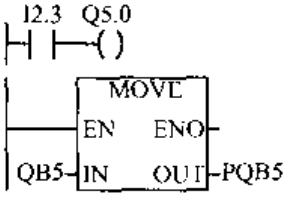


图 3-72　立即写出

3.6.6　累加器指令

语句表中的累加器指令用于处理单个或多个累加器的内容(见表 3-33)。

指令的执行与 RLO(逻辑运算结果)无关,也不会对 RLO 产生影响。对于有 4 个累加器的 CPU,累加器 3、4 的内容保持不变。

表 3-33　累加器指令

语　句　表	描　　述
TAK	交换累加器 1,2 的内容
PUSH	入栈
POP	出栈
ENT	进入 ACCU 堆栈
LEAVE	离开 ACCU 堆栈
INC	累加器 1 最低字节加上 8 位常数
DEC	累加器 1 最低字节减去 8 位常数
+AR1	AR1 的内容加上地址偏移量
+AR2	AR2 的内容加上地址偏移量
BLD	程序显示指令(空指令)
NOP 0	空操作指令
NOP 1	空操作指令

1. TAK 指令

指令 TAK(Toggle ACCU 1 with ACCU 2)交换累加器 1 和累加器 2 的内容。

【例 3-9】　下面的程序用 MW10 和 MW12 中较大的数减去较小的数,运算结果存放在 MW14 中。

```
L     MW10      //MW10 的内容装入累加器 1 的低字
L     MW12      //累加器 1 的内容装入累加器 2,MW12 的值装入累加器 1 的低字
>I              //如果 MW10 > MW12,RLO = 1
```

```
JC      NEX1        //跳转到标号 NEX1 处
TAK                 //交换累加器 1,2 低字的内容
NEX1：-I            //累加器 2 低字的内容减去累加器 1 低字的内容
T       MW14        //运算结果传送到 MW14
```

2. **堆栈指令**

S7-300 的 CPU 有两个累加器，S7-400 的 CPU 有 4 个累加器。CPU 中的累加器组成了一个堆栈，堆栈用来存放需要快速存取的数据，堆栈中的数据按“先进后出”的原则存取。堆栈指令是否执行与状态字无关，也不会影响状态字。

对于只有两个累加器的 CPU 来说，PUSH(入栈)指令将累加器 1 的内容复制到累加器 2，累加器 1 的内容不变。POP(出栈)指令将累加器 2 的内容复制到累加器 1，累加器 2 的内容不变。

对于有 4 个累加器的 CPU 来说，PUSH(入栈)指令使堆栈中各层原有的数据依次向下移动一层，栈底(累加器 4)的值被推出丢失(见图 3-73)。栈顶(累加器 1)的值保持不变。即累加器 3 的内容复制到累加器 4，累加器 2 的内容复制到累加器 3，累加器 1 的内容复制到累加器 2，累加器 1 的内容不变。

POP(出栈)指令使堆栈中各层原有的数据向上移动一层(见图 3-74)，原来第 2 层(累加器 2)中的数据成为堆栈新的栈顶值，原来栈顶(累加器 1)中的数据从栈内消失。即累加器 2 的内容复制到累加器 1，累加器 3 的内容复制到累加器 2，累加器 4 的内容复制到累加器 3、累加器 4 的内容不变。

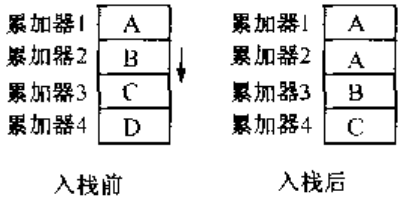


图 3-73　入栈指令执行前后

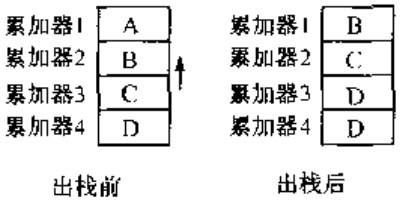


图 3-74　出栈指令执行前后

“进入累加器堆栈”指令 ENT (Enter Accumulator Stack)将累加器 3 的内容复制到累加器 4、累加器 2 的内容复制到累加器 3。使用 ENT 指令可以在累加器 3 中保存中间结果。

“离开累加器堆栈”指令 LEAVE (Leave Accumulator Stack) 将累加器 3 的内容复制到累加器 2，累加器 4 的内容复制到累加器 3，累加器 1,4 保持不变。

【例 3-10】　用语句表程序实现浮点数运算(DBD0 + DBD4)/(DBD8 - DBD12)。

```
L       DBD0        //DBD0 中的浮点数装入累加器 1
L       DBD4        //累加器 1 的内容装入累加器 2,DBD4 中的浮点数装入累加器 1
+R                  //累加器 1,2 中的浮点数相加,结果保存在累加器 1 中
L       DBD8        //累加器 1 的内容装入累加器 2,DBD8 中的浮点数装入累加器 1
ENT                 //累加器 3 的内容装入累加器 4,累加器 2 的中间结果装入累加器 3
L       DBD12       //累加器 1 的内容装入累加器 2,DBD12 中的浮点数装入累加器 1
-R                  //累加器 2 的内容减去累加器 1 的内容,结果保存在累加器 1 中
LEAVE               //累加器 3 的内容装入累加器 2,累加器 4 的中间结果装入累加器 3
/R                  //累加器 2 的内容(DBD0 + DBD4)除以累加器 1 的内容(DBD8 - DBD12)
```

```
T     DBD16     //累加器 1 中的运算结果传送到 DBD16
```

3. 加、减 8 位整数指令

字节加指令 INC(Increment ACCU 1－LL) 和字节减指令 DEC (Decrement ACCU 1－LL)将累加器 1 的最低字节(ACCU 1－LL)的内容加上或减去指令中的 8 位常数(0～255),运算结果仍在累加器的最低字节中。累加器 1 的其他 3 个字节不变。

这些指令并不适合于 16 位或 32 位算术运算,因为累加器 1 的最低字节和它的相邻字节之间没有进位产生,16 位或 32 位加法运算可以分别使用 INC 和 DEC 指令。

```
L     MB4       //MB4 的内容装入累加器 1 的最低字节
INC 1           //累加器 1 最低字节的内容加 1,结果存放在累加器 1 的最低字节
T     MB4       //运算结果传回 MB4
```

4. 地址寄存器指令

+AR1 (Add to AR1) 指令将地址寄存器 AR1 的内容加上作为地址偏移量的累加器 1 中低字的内容,或加上指令中的 16 位常数(－32 768 ～ ＋32 767),结果在 AR1 中。

+AR2 (Add to AR2) 指令将地址寄存器 AR2 的内容加上作为地址偏移量的累加器 1 中低字的内容,或加上指令中的 16 位常数,结果在 AR2 中。

16 位有符号整数首先被扩充为 24 位,其符号位不变 ,然后与 AR1 中的低 24 位有效数字相加。地址寄存器中的存储区域标识符(第 24～26 位)保持不变。

```
L     +300      //常数"+300"装入累加器 1 的低字
+AR1            //AR1 与累加器 1 的低字中的内容相加,运算结果送 AR1
+AR2  P#300.0   //AR2 的内容加上地址偏移量 300.0,运算结果送 AR2
```

5. 空操作指令

BLD ＜number＞ (程序显示指令,空指令) 并不执行什么功能,也不会影响状态位。该指令只是用于编程设备的图形显示,在 STEP 7 中将梯形图或功能块图转换为语句表时,将会自动生成 BLD 指令。指令中的＜number＞是编程设备自动生成的。

NOP 0 和 NOP 1 指令并不执行什么功能,也不会影响状态位,它们的指令代码中分别由 16 个 0 或 16 个 1 组成,其作用与 BLD 指令类似。

3.7 逻辑控制指令

3.7.1 跳转指令

逻辑控制指令是逻辑块内的跳转和循环指令。在没有执行跳转和循环指令时,各条语句按从上到下的先后顺序逐条执行,这种执行方式称为线性扫描。逻辑控制指令中断程序的线性扫描,跳转到指令中的地址标号所在的目的地址。跳转时不执行跳转指令与标号之间的程序,跳到目的地址后,程序继续按线性扫描的方式执行。跳转可以是从上往下的,也可以是从下往上的。

逻辑控制指令与状态位触点指令见表 3-34。

表 3-34　逻辑控制指令与状态位触点指令

语句表中的逻辑控制指令	梯形图中的状态位触点指令	说　明
JU	–	无条件跳转
JL	–	多分支跳转
JC	–	RLO=1 时跳转
JCN	–	RLO=0 时跳转
JCB	–	RLO=1 且 BR=1 时跳转
JNB	–	RLO=0 且 BR=1 时跳转
JBI	BR	BR=1 时跳转
JNBI	–	BR=0 时跳转
JO	OV	OV=1 时跳转
JOS	OS	OS=1 时跳转
JZ	= =0	运算结果为 0 时跳转
JN	<>0	运算结果非 0 时跳转
JP	>0	运算结果为正时跳转
JM	<0	运算结果为负时跳转
JPZ	>=0	运算结果大于等于 0 时跳转
JMZ	<=0	运算结果小于等于 0 时跳转
JUO	UO	指令出错时跳转
LOOP	–	循环指令

只能在同一逻辑块内跳转，即跳转指令与对应的跳转目的地址应在同一逻辑块内。在一个块中，同一个跳转目的地址只能出现一次。最长的跳转距离为程序代码中的－32 768 或 +32 767 个字。实际可以跳转的最大语句条数与每条语句的长度(1～3 个字)有关。跳转指令只能在 FB、FC 和 OB 内部使用，即不能跳转到别的 FB、FC 和 OB 中去。

跳转或循环指令的操作数为地址标号，标号由最多 4 个字符组成，第一个字符必须是字母，其余的可以是字母或数字。在语句表中，目标标号与目标指令用冒号分隔。在梯形图中，目标标号必须是一个网络的开始。

1. 无条件跳转指令

无条件跳转(Jump Unconditional)指令的格式为 JU <跳转标号>，JU 指令中断程序的线性扫描，跳转到标号所在的目的地址，无条件跳转与状态字的内容无关。

2. 多分支跳转指令

多分支跳转指令 JL(Jump Via Jump to List)必须与无条件跳转指令 JU 一起使用，指令格式为 JL <跳转标号>，多分支的路径参数在累加器 1 中。跳步目标表最多 255 个入口通道，从 JL 指令的下一行开始，在 JL 指令中指定的跳步标号之前结束。每个跳步目标由一条 JU 指令和一个标号组成。跳步目标号在累加器 1 的最低字节 ACCU 1－LL 中。

当累加器 1 最低字节 ACCU1－LL 中的跳步目标号小于 JL 指令和它给出的标号之间的 JU 指令的条数时，执行 JL 指令后将根据跳步目标号跳到对应的 JU 指令指定的标号。ACCU1－LL＝0 时跳转到第一条 JU 指令指定的标号，ACCU 1－LL＝1 时跳转到第二条 JU 指令指定的标号……如果跳步目标号过大，JL 指令将跳到跳步目标表中最后一条 JU 指令后面的第一条指令。

跳步目标表必须由在 JL 指令中的跳步标号之前的 JU 指令组成，其他任何指令都是非法的。图 3-75 给出了下面的程序执行的情况。

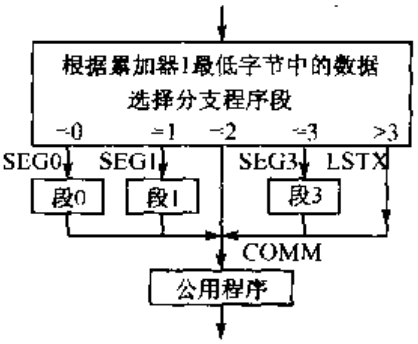


图 3-75　多分支跳转指令

```
        L     MB0          //MB0 的跳步目标号装入 ACCU1－LL(累加器 1 的最低字节)
        JL    LSTX         //如果 ACCU1－LL ＞3,则跳转到标号 LSTX 处
        JU    SEG0         //如果 ACCU1－LL ＝0,则跳转到标号 SEG0 处
        JU    SEG1         //如果 ACCU1－LL ＝1,则跳转到标号 SEG1 处
        JU    COMM         //如果 ACCU1－LL ＝2,则跳转到标号 COMM 处
        JU    SEG3         //如果 ACCU1－LL ＝3,则跳转到标号 SEG3 处
LSTX:   JU    COMM
SEG0:   ……                //程序段 0
        ……
        JU    COMM
SEG1:   ……                ///程序段 1
        ……
        JU    COMM
SEG3:   ……                ///程序段 3
        ……
        JU    COMM
COMM:   ……                //公用程序
        ……
```

3．与 RLO 和 BR 有关的跳转指令

这些指令检查前一条指令执行后 RLO(逻辑运算结果)和 BR(二进制结果位)的状态，满足条件时则中断程序的线性扫描，跳转到标号所在的目的地址，不满足条件时不跳转。

如果逻辑运算结果 RLO ＝ 1，跳转指令 JC 将跳转到标号所在的目的地址。

如果逻辑运算结果 RLO ＝ 0，跳转指令 JCN 将跳转到标号所在的目的地址。

如果逻辑运算结果 RLO ＝ 1，且 BR ＝ 1，跳转指令 JCB 将跳转到标号所在的目的地址。

如果逻辑运算结果 RLO ＝ 0，且 BR ＝ 1，跳转指令 JNB 将跳转到标号所在的目的地址。

4．与信号状态位有关的跳转指令

这些指令检查前一条指令执行后信号状态位 BR(二进制结果位)、OV(溢出位)和 OS(溢出状态保持位)的状态，满足条件时则中断程序的线性扫描，跳转到标号所在的目的地址，不满足条件时不跳转。

如果 BR ＝ 1，跳转指令 JBI 将跳转到标号所在的目的地址。

如果 BR = 0,跳转指令 JNBI 将跳转到标号所在的目的地址。

如果 OV = 1,跳转指令 JO 将跳转到标号所在的目的地址。

如果 OS = 1,跳转指令 JOS 将跳转到标号所在的目的地址。

5. 与条件码 CC0 和 CC1 有关的跳转指令

这些指令根据前一条指令执行后与运算结果有关的条件码 CC0 和 CC1 的状态,确定是否中断程序的线性扫描,跳转到标号所在的目的地址。

如果运算结果为 0(CC0=0,CC1=0),跳转指令 JZ 将跳转到标号所在的目的地址。

如果运算结果非 0(CC 1=0/CC 0=1 或 CC 1=1/CC 0=0),跳转指令 JN 将跳转到标号所在的目的地址。

如果运算结果为正(CC 1 = 1 与 CC 0 = 0),跳转指令 JP 将跳转到标号所在的目的地址。

如果运算结果为负(CC 1 = 0 与 CC 0 = 1),跳转指令 JM 将跳转到标号所在的目的地址。

如果运算结果大于等于 0(CC 1=0/CC 0=0 或 CC 1=1/CC 0=0),跳转指令 JPZ 将跳转到标号所在的目的地址。

如果运算结果小于等于 0(CC 1=0/CC 0=0 或 CC 1=1/CC 0=0),跳转指令 JMZ 将跳转到标号所在的目的地址。

如果 CC0 = CC1 = 1,表示指令出错(除数为 0;使用了非法的指令;浮点数比较时使用了非法的格式),跳转指令 JUO 将跳转到标号所在的目的地址。

【例 3-11】 IW8 与 MW12 的异或结果如果为 0,将 M4.0 复位,非 0 则将 M4.0 置位。

下面是实现要求的程序,图 3-76 给出了程序的流程图。

```
        L     IW8        //IW8 的内容装入累加器 1 的低字
        L     MW12       //累加器 1 的内容装入累加器 2,MW12 的内容装入累加器 1 的低字
        XOW              //累加器 1,2 低字的内容逐位异或
        JN    NOZE       //如果累加器 1 的内容非 0,则跳转到标号 NOZE 处
        R     M4.0
        JU    NEXT
NOZE:   AN    M4.0
        S     M4.0
NEXT:   NOP 0
```

6. 梯形图中的跳转指令

梯形图中有 3 条用线圈表示的跳转指令(见图 3-77),无条件跳转(Unconditional Jump)指令与条件跳转(Conditional Jump)指令的助记符均为 JMP(Jump),其区别在于跳转指令是否受触点电路的控制。

无条件跳转指令直接与右边的垂直电源线相连,执行无条件跳转指令后马上跳转到指令给出的标号处。

条件跳转指令的线圈受触点电路的控制,它前面的逻辑运算结果 RLO=1 时,跳转线圈“通电”,跳转到指令给出的标号处。

JMPN(Jump-If-Not)指令在它右边的电路断开(RLO=0)时跳转(见图 3-78)。

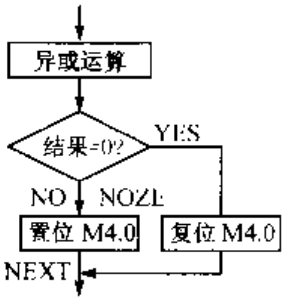


图 3-76　跳转指令

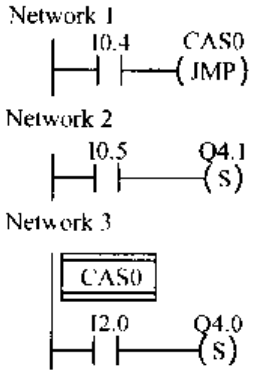


图 3-77　条件跳转指令

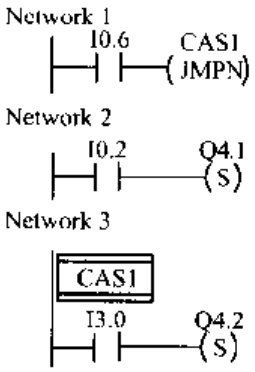


图 3-78　JMPN 跳转指令

标号用于指示跳转指令的目的地址，它最多由 4 个字符组成，第一个字符必须是字母或下划线。标号必须放在一个网络开始的地方。可以向前跳，也可以向后跳。双击梯形图编辑器右边的指令浏览器窗口中的“Jumps”文件夹中的“LABEL”图标，一个空的标号框将出现在梯形图编辑区光标所在的地方。也可以用鼠标左键按住 LEBEL 图标，将它“拖”到梯形图中。

3.7.2　梯形图中的状态位触点指令

梯形图中的状态位指令以常开触点或常闭触点的形式出现。这些触点的通断取决于状态字中的状态位 BR、OV、OS、CC0 和 CC1 的状态（见表 3-34）。数学运算的结果等于 0、不等于 0、大于 0、小于 0、大于等于 0、小于等于 0 都有对应的状态位常开触点和常闭触点。CC0 和 CC1 均为 1 时，表示数学运算指令有错误，UO 常开触点闭合。

以标有 OV 的触点为例，OV（溢出位）为 1 时，标有 OV 的常开触点闭合，常闭触点断开。

图 3-79 中的 I0.6 为 1 时，执行整数减法指令 SUB _ I，如果运算结果有溢出（超出允许的范围），状态位 OV 为 1，梯形图中 OV 的常开触点闭合。若 I0.2 的常开触点也闭合，Q4.0 被置位。

在梯形图中，状态位触点可以与别的触点串并联。

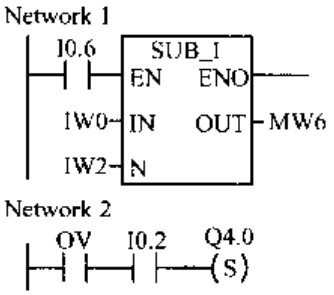


图 3-79　状态位触点指令

3.7.3　循环指令

如果需要重复执行若干次同样的任务，可以使用循环指令。循环指令 LOOP <jump label>用 ACCU 1 - L 作循环计数器，每次执行 LOOP 指令时 ACCU 1 - L 的值减 1，若减 1 后 ACCU 1 - L 非 0，将跳转到<jump label>指定的标号处，在跳步目标处又恢复线性程序扫描。可以往前跳，也可以往后跳，跳步目标号应是惟一的，跳步只能在同一个逻辑块内进行。

【例 3-12】　用循环指令求 5!（5 的阶乘）。

```
      L     L#1     //32 位整数常数装入累加器 1，置阶乘的初值
      T     MD20    //累加器 1 的内容传送到 MD20，保存阶乘的初值
      L     5       //循环次数装入累加器的低字
BACK: T     MW10    //累加器 1 低字的内容保存到循环计数器 MW10
      L     MD20    //取阶乘值
```

```
*D          //MD20 与 MW10 的内容相乘
T    MD20   //乘积送 MD20
L    MW10   //循环计数器内容装入累加器 1
LOOP BACK   //累加器 1 低字的内容减 1,如果减 1 后大于 0,跳转到标号 BACK 处
……          //循环结束后,恢复线性扫描
```

3.8 程序控制指令

3.8.1 逻辑块指令

逻辑块包括功能、功能块、系统功能和系统功能块。程序控制指令包括逻辑块结束指令、逻辑块调用指令、主控继电器指令和操作数据块的指令。程序控制指令见表 3-35。

1. 逻辑块结束指令

逻辑块结束指令包括块无条件结束指令 BEU (Block End Unconditional)和块结束指令 BE,以及块条件结束指令 BEC(Block End Conditional)。

执行块结束指令时,将中止当前块的程序扫描,返回调用它的块。BEU 和 BE 是无条件执行的,而 BEC 只是在 RLO=1 时执行。

表 3-35 程序控制指令

语句表指令	梯形图指令	描 述
BE	–	块结束
BEU	–	块无条件结束
BEC	–	块条件结束
CALL FCn	–	调用功能
CALL SFCn	–	调用系统功能
CALL FBn1,DBn2	–	调用功能块
CALL SFBn1,DBn2	–	调用系统功能块
CC FCn 或 SFCn	CALL	RLO=1 时条件调用
UC FCn 或 SFCn	CALL	无条件调用
RET	RET	条件返回
MCRA	MCRA	起动主控继电器功能
MCRD	MCRD	取消主控继电器功能
MCR(	MCR<	打开主控继电器区
)MCR	MCR>	关闭主控继电器区

假设在逻辑块 A 中调用逻辑块 B,执行逻辑块 B 中的无条件结束指令 BEU 或在条件满足时执行 BEC 指令,将会中止逻辑块 B(当前块)的程序扫描,返回逻辑块 A 中调用逻辑块 B 的调用(CALL)指令下面一条指令,继续程序扫描。逻辑块 B 结束后,它的局域数据区被释放出来,调用它的块 A 的局域数据区变为当前局域数据区。在块 A 调用块 B 时打开的数据块被重

新打开。块 A 的主控继电器(MCR)被恢复,RLO 从块 B 被带到块 A。

BEU 指令的执行不需要任何条件,但是如果 BEU 指令被跳转指令跳过,当前程序扫描不会结束,在块内的跳转目标处,程序将被继续起动。

使用 S7 系列 PLC 的硬件时,块结束指令 BE (Block End) 与 BEU 的功能相同。

下面是使用 BEC 的程序:

```
A       I0.1      //刷新 RLO
BEC               //如果 RLO=1,结束块
L       IW4       //如果 RLO=0,不执行 BEC,继续程序扫描
        T         MW10
```

2. 逻辑块调用指令

块调用指令(CALL)用来调用功能块(FB)、功能(FC)、系统功能块(SFB)或系统功能(SFC),或调用西门子预先编好的其他标准块。

在 CALL 指令中,FC、SFC、FB 和 SFB 是作为地址输入的,逻辑块的地址可以是绝对地址或符号地址。CALL 指令与 RLO 和其他任何条件无关。在调用 FB 和 SFB 时,应提供与它们配套的背景数据块(Instance DB)。调用 FC 和 SFC 时,不需要背景数据块。处理完被调用的块后,调用它的程序继续其逻辑处理。在调用 SFB 和 SFC 后,寄存器的内容被恢复。

使用 CALL 指令时,应将实参(Actual Parameter)赋给被调用的功能块中的形参(Formal Parameter),并保证实参与形参的数据类型一致。

使用语句表编程时,CALL 指令中被调用的块应是已经存在的块,其符号名也应该是已经定义过的。

在调用块时可以通过变量表交换参数,用编程软件编写语句表程序时,如果被调用的逻辑块的变量声明表中有 IN、OUT 和 IN_OUT 类型的变量,输入 CALL 指令后编程软件会自动打开变量表,只需对各形参填写对应的实参就可以了。

在调用 FC 和 SFC 时,必须为所有的形参指定实参。调用 FB 和 SFB 时,只需指定上次调用后必须改变的实参。因为 FB 被处理后,实参储存在背景数据块中。如果实参是数据块中的地址,必须指定完整的绝对地址,例如 DB1.DBW2。

逻辑块的 IN(输入)参数可以指定为常数、绝对地址或符号地址。OUT(输出)和 IN_OUT(输入_输出)参数必须指定为绝对地址或符号地址。

CALL 指令保存被停止执行的块的编号和返回地址,以及当时打开的数据块的编号。此外,CALL 指令关闭 MCR 区,生成被调用的块的局域数据区。

在下面的例子中,功能块 FB1 的背景数据块是 DB1,":="前面是用符号地址表示的形参,":="后面是实参。

```
CALL   FB1, DB1
  Switch_On              :=I20.0     //将实参 I20.0 赋给形参 Switch_On
  Switch_Off             :=I20.1
  Failure                :=I20.2
  Actual_Speed           :=MW2
  Engine_On              :=Q5.0
  Preset_Speed_Reached   :=Q5.1
```

```
CALL        SFC 43        //调用 SFC43,重新触发监控定时器(无参数)
```

每一个 FB 和 SFB 都必须有一个背景数据块,上例中在调用 FB1 之前,FB1 和背景数据块 DB1 必须是已经存在的。

无条件调用指令 UC (Unconditional Block Call)和条件调用指令 CC (Conditional Block Call)用于调用没有参数的 FC 和 SFC。其使用方法与 CALL 指令相同,只是在调用时不能传递参数。CC 指令在逻辑运算结果 RLO=1 时才调用块。用 CC 指令和 UC 指令调用块时,不能使用背景数据块。下面是使用 CC 指令和 UC 指令的例子:

```
A     I0.1     //刷新 RLO
CC    FC6      //如果 RLO=1,调用没有参数的功能 FC6
L     IW4      //从 FC6 返回后执行,或在 I0.1=0 时不调用 FC6,直接执行本指令
UC    FC2      //无条件调用没有参数的功能 FC2
```

3. 梯形图中的逻辑块调用指令

梯形图中的 CALL 线圈可以调用功能 FC 或系统功能 SFC,调用时不能传递参数。调用可以是无条件的,CALL 线圈直接与左侧垂直线相连,相当于语句表中的 UC 指令;也可以是有条件的,条件由控制 CALL 线圈的触点电路提供,相当于语句表中的 CC 指令。

调用逻辑块时如果需要传递参数,可以用方框指令来调用功能块。图 3-80 方框中的 FB10 是被调用的功能块,DB3 是调用 FB10 时的背景数据块。

条件返回指令 RET (Return)以线圈的形式出现,用于有条件地离开逻辑块,条件由控制它的触点电路提供,RET 线圈不能直接连接在左侧垂直"电源线"上。如果是无条件地返回调用它的块,在块结束时并不需要使用 RET 指令。

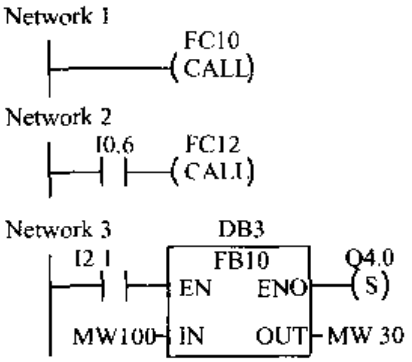


图 3-80 逻辑块调用

3.8.2 主控继电器指令

主控继电器(Master Control Relay)简称为 MCR。主控继电器指令用来控制 MCR 区内的指令是否被正常执行,相当于一个用来接通和断开"能流"的主令开关。

MCRA 为激活 MCR 区(Activate MCR Area)指令,表明按 MCR 方式操作的区域的开始;MCRD 为取消 MCR 区(Deactivate MCR Area)指令,表示按 MCR 方式操作的区域的结束。MCRA 和 MCRD 指令应成对使用,这两条指令之间的程序的执行与否与 MCR 位的状态有关,MCR 区之外的指令不受 MCR 位的影响。

打开主控继电器区指令"MCR(" (Open a Master Control Relay zone) 在 MCR 堆栈中保存该指令之前的逻辑运算结果 RLO(即 MCR 位)。MCR 指令可以嵌套使用,即 MCR 区可以在另一个 MCR 区之内。MCR 堆栈是一种后进先出的堆栈,允许的最大嵌套深度为 8 级。如果堆栈已经装满,该指令将产生"MCRF"(MCR 堆栈故障)信息。

关闭主控继电器区指令")MCR"(Close the Last Opened MCR Zone)从 MCR 堆栈中取出保存在里面的 RLO。如果堆栈已经是空的,该指令将产生"MCRF"(MCR 堆栈故障)信息。

"MCR("与")MCR"指令必须成对使用,以表示受控临时"电源线"的形成与终止。

若在 MCRA 和 MCRD 之间有块结束指令 BEU，CPU 执行 BEU 的同时也会结束 MCR 区。如果在 MCR 区内有块调用指令，MCR 的激活状态不能继承到被调用的块中，必须在被调用的块内重新激活新的 MCR 区。

在图 3-81 中，为了节省版面，没有标出各网络(Network)的标号，用左侧垂直电源线的中断表示两个相邻网络的交界处。

在梯形图中，上述 4 条与 MCR 有关的指令用线圈的形式表示，语句表中的“MCR(”与“)MCR”指令分别用线圈中的“MCR＜”和“MCR＞”来表示。在图 3-81 中，MCR 位受到 I0.2 的控制，I0.2 为 1 时，MCR 堆栈中的 MCR 位为 1，I0.2 为 0 时，MCR 位也为 0。MCR 控制区内的 Q4.0 的线圈和 MOVE 指令的执行与否都与 MCR 位的状态有关。

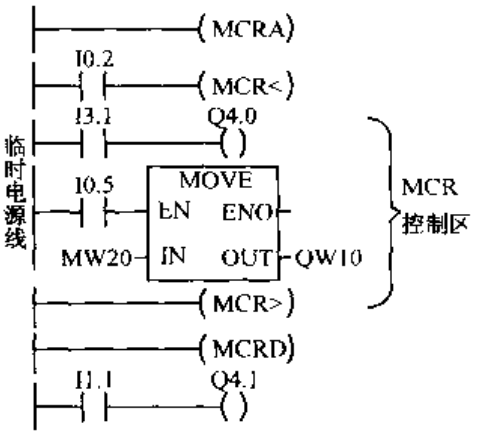


图 3-81　主控继电路指令

打开 MCR 区后，如果保存在 MCR 堆栈中的 MCR 位的状态为 1，可以视为受它控制的左侧的临时“电源线”“通电”，MCR 区内的程序正常执行。

如果 SCR 位的状态为 0，临时“电源线”“断电”，程序按下面的方式处理：

“＝”指令(输出线圈、中间输出线圈)中的存储位被写入 0，即线圈断电；被置位和复位(S，R)的存储位保持当前状态不变；传送或赋值指令(T)中的地址被写入 0。

下面是与图 3-81 中的梯形图对应的语句表程序：

```
Network 1：
        MCRA    //激活 MCR 区
Network 2：
        A  I0.2
        MCR(    // RLO 保存到 MCR 堆栈，打开 MCR 区。I0.2＝1 时 MCR 位为 1，反之均为 0
Network 3：
        A  I3.1
        =  Q4.0 //如果 MCR 位为 0 状态，不管 I3.1 的状态如何，Q4.0 被置为 0
Network 4：
        A  I0.5
        JNB _001
        L  MW20
        T  QW10 //如果 MCR 位为 0 状态，0 送入 QW10
_001：  NOP 0
Network 5：
        )MCR    //结束 MCR 控制区
Network 6：
        MCRD    //关闭 MCR 区
Network7：
        A       I1.1
        =  Q 4.1//这两条指令在 MCR 区外，不受 MCR 位的控制
```

3.8.3 数据块指令

数据块指令见表 3-36。

表 3-36 数据块指令

指 令	描 述
OPN	打开数据块
CDB	交换共享数据块和背景数据
L DBLG	共享数据块的长度装入累加器 1
L DBNO	共享数据块的编号装入累加器 1
L DILG	背景数据块的长度装入累加器 1
L DINO	背景数据块的编号装入累加器 1

在语句表中，OPN(Open a Data Block)指令用来打开共享数据块和背景数据块。同时只能打开一个共享数据块和一个背景数据块。访问已经打开的数据块内的存储单元时，其地址中不必指明是哪一个数据块的数据单元。例如在打开 DB10 后，DB10.DBW35 可简写为 DBW35。

```
OPN    DB10    //打开数据块 DB10 作为共享数据块
L      DBW35   //将打开的 DB10 中的数据字 DBW35 装入累加器 1 的低字
T      MW12    //累加器 1 低字的内容装入 MW12
OPN    DI20    //打开作为背景数据块的数据块 DB20
L      DIB35   //将打开的背景数据块 DB20 中的数据字节 DIB35 装入累加器 1 的最低字节
T      DBB27   //累加器 1 最低字节传送到被打开的共享数据块 DB10 的数据字节 DBB27
```

CDB 指令交换两个数据块寄存器的内容，即交换共享数据块和背景数据块，使共享数据块变为背景数据块，背景数据块变为共享数据块。两次使用 CDB 指令，使两个数据块还原。

L DBLG (Load Length of Shared Data Block) 指令将共享数据块的长度装入累加器 1。

L DBNO (Load Number of Shared Data Block) 指令将共享数据块的编号装入累加器 1。

L DILG (Load Length of Instance Data Block) 指令将背景数据块的长度装入累加器 1。

L DINO (Load Number of Instance Data Block) 指令将背景数据块的编号装入累加器 1。

在梯形图中，与数据块操作有关的只有一条无条件打开共享数据块或背景数据块的指令(见图 3-82)。在网络 2 中，因为数据块 DB10 已经被打开，其中的数据位 DBX1.0 相当于 DB10.DBX1.0。

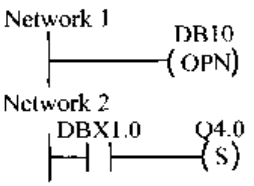


图 3-82 打开数据块

3.8.4 梯形图的编程规则

下面是梯形图编程时应遵守的一些规则：

(1) 每个梯形图程序段都必须以输出线圈或指令框(Box)结束，比较指令框(相当于触点)、中线输出线圈和上升沿、下降沿线圈不能用于程序段结束。

(2) 指令框的使能输出端"ENO"可以和右边的指令框的使能输入端"EN"连接(见图 3-50)。

(3) 下列线圈要求布尔逻辑，即必须用触点电路控制它们，它们不能与左侧垂直"电源线"直接相连：输出线圈、置位(S)、复位(R)线圈；中线输出线圈和上升沿、下降沿线圈；计数器和

定时器线圈；逻辑非跳转（JMPN）；主控继电器接通（MCR＜）；将 RLO 存入 BR 存储器（SAVE）和返回线圈（RET）。

下面的线圈不允许布尔逻辑，即这些线圈必须与左侧垂直“电源线”直接相连：主控继电器激活（MCRA）；主控继电器关闭（MCRD）和打开数据块（OPN）。

其他线圈既可以用布尔逻辑操作也可以不用。

（4）下列线圈不能用于并联输出：逻辑非跳转（JMPN）；跳转（JMP）；调用（CALL）和返回（RET）。

（5）如果分支中只有一个元件，删除这个元件时，整个分支也同时被删掉。删除一个指令框时，该指令框除主分支外所有的布尔输入分支都将同时被删除。

（6）能流只能从左到右流动，不允许生成使能流流向相反方向的分支。例如图 3-83 中的 I0.3 的常开触点断开时，能流流过 I0.4 的方向是从右到左，这是不允许的。从本质上来说，该电路不能用触点的串、并联指令来表示。

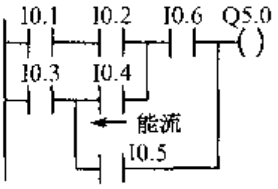


图 3-83　错误的电路

（7）不允许生成引起短路的分支。

第 4 章　STEP 7 编程软件的使用方法

4.1　STEP 7 编程软件简介

4.1.1　STEP 7 概述

STEP 7 编程软件用于 SIMATIC S7、M7、C7 和基于 PC 的 WinAC，是供它们编程、监控和参数设置的标准工具。本书对 STEP 7 操作的描述，都是基于 STEP 7 V5.2 版的。

为了在个人计算机上使用 STEP 7，应配置 MPI 通信卡或 PC/MPI 通信适配器，将计算机连接到 MPI 或 PROFIBUS 网络，来下载和上载 PLC 的用户程序和组态数据。STEP 7 允许两个或多个用户同时处理一个工程项目，但是禁止两个或多个用户同时写访问。

STEP 7 具有以下功能：硬件配置和参数设置、通信组态、编程、测试、启动和维护、文件建档、运行和诊断功能等。STEP 7 的所有功能均有大量的在线帮助，用鼠标打开或选中某一对象，按 F1 键可以得到该对象的在线帮助，

在 STEP 7 中，用项目来管理一个自动化系统的硬件和软件。STEP 7 用 SIMATIC 管理器对项目进行集中管理，它可以方便地浏览 SIMATIC S7、M7、C7 和 WinAC 的数据。实现 STEP7 各种功能所需的 SIMATIC 软件工具都集成在 STEP 7 中。

STEP 7 中的转换程序可以转换在 STEP 5 或 TISOFT 中生成的程序。

4.1.2　STEP 7 的硬件接口

PC/MPI 适配器用于连接安装了 STEP 7 的计算机的 RS-232C 接口和 PLC 的 MPI 接口。计算机一侧的通信速率为 19.2 kbit/s 或 38.4 kbit/s，PLC 一侧的通信速率为 19.2 kbit/s～1.5 Mbit/s。除了 PC 适配器，还需要一根标准的 RS-232C 通信电缆。

使用计算机的通信卡 CP 5611（PCI 卡）、CP 5511 或 CP 5512（PCMCIA 卡），可以将计算机连接到 MPI 或 PROFIBUS 网络，通过网络实现计算机与 PLC 的通信。

也可以使用计算机的工业以太网通信卡 CP 1512（PCMCIA 卡）或 CP 1612（PCI 卡），通过工业以太网实现计算机与 PLC 的通信。

在计算机上安装好 STEP7 后，在管理器中执行菜单命令“Option”→“Setting the PG/PC Interface”，打开“Setting PG/PC Interface”对话框。在中间的选择框中，选择实际使用的硬件接口。点击【Select…】按钮，打开“Install/Remove Interfaces”对话框，可以安装上述选择框中没有列出的硬件接口的驱动程序。点击【Properties…】按钮，可以设置计算机与 PLC 通信的参数。

4.1.3　STEP 7 的授权

使用 STEP 7 编程软件时需要产品的特别授权（用户权），STEP 7 与可选的软件包需要不同的授权。

STEP 7 的授权存放在一张只读的授权软盘中。STEP 7 的光盘上的程序 AuthorsW 用于

显示、安装和取出授权。每安装一个授权，授权磁盘上的授权计数器减 1，当计数值为 0 时，不能用这张磁盘再安装授权。

没有授权也可以使用 STEP 7，以便熟悉用户接口和功能，但是在使用时每隔一段时间将会搜索授权，提醒使用者安装授权，只有安装了授权才能有效地使 STEP 7 工作。

如果因为硬盘出现故障而丢失授权，可以使用授权盘上的紧急授权，它允许 STEP 7 继续运行一段有限的时间。在此期间应与当地西门子代表处联系，以获得丢失授权的替换授权。

AuthorsW 程序的默认位置是“开始”→“SIMATIC”→“AuthorsW”→“AuthorsW”，可以在第一次安装 STEP 7 软件时安装授权，也可以以后安装它。安装授权的步骤如下：

(1) 把授权磁盘插入软盘驱动器，启动硬盘上的程序 AuthorsW.EXE。

(2) 在出现的对话框的 Move Authorization(s) 选项卡中有两个目录框，一个目录框应选择含有授权的驱动器，另一个应选择目标驱动器，目录框中将显示两个驱动器上所有的授权。

(3) 选择所需授权，点击按钮【<--】或【-->】，选择的授权将被传送到另一个驱动器。

如果授权出了问题，与热线联系，能用 AuthorsW 中的菜单命令“Authorization”→“Recover”恢复授权。

使用 AuthorsW 程序可以把授权传回授权磁盘，以后可以用这张磁盘再次安装一个授权，也可以在硬盘的不同分区之间移动授权。

4.1.4 STEP 7 的编程功能

1. 编程语言

STEP 7 的标准版只配置了 3 种基本的编程语言：梯形图(LAD)、功能块图(FBD) 和语句表(STL)，有鼠标拖放、复制和粘贴功能。语句表是一种文本编程语言，使用户能节省输入时间和存储区域，并且“更接近硬件”。

用户可以按“增量”方式输入，立即检查每一个输入的正确性；或者先在文本编辑器上用字符生成整个程序的源文件，然后将它编译为软件块。

STEP 7 专业版的编程语言包括 S7-SCL (结构化控制语言)；S7-GRAPH(顺序功能图语言)；S7 HiGraph 和 CFC。这 4 种编程语言对于标准版是可选的。

2. 符号表编辑器

STEP 7 用符号表编辑器工具管理所有的全局变量；用于定义符号名称、数据类型和全局变量的注释。使用这一工具生成的符号表可供所有应用程序使用，所有工具自动识别系统参数的变化。

3. 增强的测试和服务功能

测试功能和服务功能包括设置断点、强制输入和输出、多 CPU 运行(仅限于 S7-400)、重新布线、显示交叉参考表、状态功能、直接下载和调试块、同时监测几个块的状态。

程序中的特殊点可以通过输入符号名或地址快速查找。

4. STEP 7 的帮助功能

(1) 在线帮助功能

选定想得到在线帮助的菜单项目，或打开对话框，按<F1>键便可以得到与它们有关的在线帮助

(2) 从帮助菜单获得帮助

利用菜单命令“Help”→“Contents”进入帮助窗口，借助目录浏览器寻找需要的帮助主题，窗口中的检索部分提供了按字母顺序排列的主题关键词，可以查找与某一关键词有关的帮助。

点击工具栏上有问号和箭头的图标，出现带问号的光标，用它点击画面上的对象时，将会进入相应的帮助窗口。

4.1.5 STEP 7 的硬件组态与诊断功能

1. 硬件组态

英语单词 configuring(配置、设置)一般被翻译为“组态”。硬件组态工具用于对自动化工程中使用的硬件进行配置和参数设置。

(1) 系统组态：从目录中选择硬件机架，并将所选模块分配给机架中希望的插槽。分布式 I/O 的配置与集中式 I/O 的配置方式相同。

(2) CPU 的参数设置：可以设置 CPU 模块的多种属性，例如启动特性、扫描监视时间等，输入的数据储存在 CPU 的系统数据块中。

(3) 模块的参数设置：用户可以在屏幕上定义所有硬件模块的可调整参数，包括功能模块(FM)与通信处理器(CP)，不必通过 DIP 开关来设置。

在参数设置屏幕中，有的参数由系统提供若干个选项，有的参数只能在允许的范围输入，因此可以防止输入错误的数据。

2. 通信组态

通信的组态包括：

(1) 连接的组态和显示。

(2) 设置用 MPI 或 PROFIBUS-DP 连接的设备之间的周期性数据传送的参数，选择通信的参与者，在表中输入数据源和数据目的地后，通信过程中数据的生成和传送均是自动完成的。

(3) 设置用 MPI、PROFIBUS 或工业以太网实现的事件驱动的数据传输，包括定义通信链路。从集成块库中选择通信块(CFB)，用通用的编程语言(例如梯形图)对所选的通信块进行参数设置。

3. 系统诊断

系统诊断为用户提供自动化系统的状态，可以通过两种方式显示：

(1) 快速浏览 CPU 的数据和用户编写的程序在运行中的故障原因。

(2) 用图形方式显示硬件配置，例如显示模块的一般信息和模块的状态；显示模块故障，例如集中 I/O 和 DP 子站的通道故障；显示诊断缓冲区的信息等。

CPU 可以显示更多的信息，例如显示循环周期；显示已占用和未用的存储区；显示 MPI 通信的容量和利用率；显示性能数据，例如可能的输入/输出点数、位存储器、计数器、定时器和块的数量等。

4.2 硬件组态与参数设置

4.2.1 项目的创建与项目的结构

1. 项目的创建

创建项目时，首先双击桌面上的 STEP 7 图标，进入 SIMATIC Manager(管理器)窗口，并

弹出标题为"STEP 7 Wizart:'New Project'"(新项目向导)的小窗口。

点击[NEXT]按钮,在新项目中选择 CPU 模块的型号为 CPU 315。

点击[NEXT]按钮,选择需要生成的逻辑块,至少需要生成作为主程序的组织块 OB1。

点击[NEXT]按钮,输入项目的名称"星三角起动"。生成的项目如图 4-1 所示。

图 4-1 SIMATIC 管理器中项目的结构

生成项目后,可以先组态硬件,然后生成软件程序。也可以在没有组态硬件的情况下,首先生成软件。

下面用一个简单的例子来介绍在 STEP 7 中生成项目和组态硬件的方法,以及项目的结构和符号表在程序设计中的应用。

图 4-2 是异步电动机星形-三角形降压起动的主电路和 PLC 的外部接线图,图 4-3 是 OB1 中的梯形图程序。主电路中的接触器 KM1 和 KM2 动作时,异步电动机运行在星形接线方式;KM1 和 KM3 动作时,运行在升角形接线方式。

按下起动按钮,KM1 和 KM2 同时动作,电动机按星形接线方式运行,定时器 T0 的线圈通电。9 s 后 T0 的常闭触点断开,通过 Q4.1 使 KM2 的线圈断电,T0 的常开触点闭合,通过 Q4.2 使 KM3 的线圈通电,电动机改为三角形接线方式运行。按下停车按钮,I0.1 的常闭触点断开,使 KM1 和 KM3 的线圈断电,电动机停止运行。

如果 KM2 和 KM3 同时动作,将会造成三相电源相间短路。图 4-3 中控制 KM2、KM3 的 Q4.1 和 Q4.2 的线圈串联了对方的常闭触点,实现了软件互锁。实践表明因 KM2 和 KM3 的状态几乎是同时切换,如果没有 PLC 外部的硬件互锁,将会造成三相电源瞬间短路,使熔断器熔断。

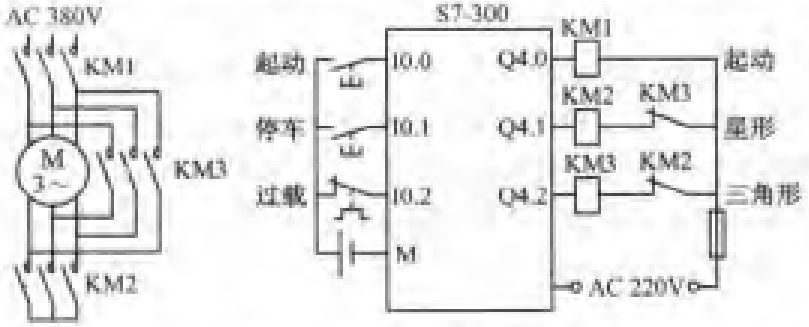


图 4-2 PLC 的外部接线图

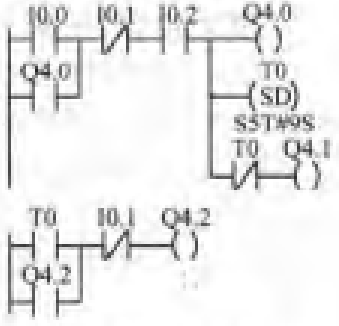


图 4-3 梯形图

为了解决这一问题,在 PLC 外部用 KM2 和 KM3 的常闭触点设置硬件互锁(见图 4-2),只有当 KM2 的主触点断开,它的辅助常闭触点闭合后,KM3 的线圈才能通电,电动机才能改接

为三角形接线方式。

2. 项目的分层结构

在项目中，数据在分层结构中以对象的形式保存，右边窗口内的树(Tree)显示项目的结构(见图 4-1)。第一层为项目，第二层为站(Station)，站是组态硬件的起点。“S7 Program”文件夹是编写程序的起点，所有的软件均存放在该文件夹中。

用鼠标选中图 4-1 中某一层的对象，在管理器右边的工作区将显示所选文件夹内的对象和下一级的文件夹。双击工作区中的图标，可以打开并编辑对象。

项目对象中包含站对象和 MPI 对象，站(Station)对象包含硬件(Hardware)和 CPU，CPU 对象包含 S7 程序(S7 Program)和连接(Connection)对象，S7 Program 对象包含源文件(Source)、块(Blocks)和符号表(Symbols)。生成程序时会自动生成一个空的符号表。

Blocks(块)对象包含程序块(Blocks)、用户定义的数据类型(UDT)、系统数据(System data)和调试程序用的变量表(VAT)。程序块包括逻辑块(OB、FB、FC)和数据块(DB)，需要把它们下载到 CPU 中，用于执行自动控制任务，符号表、变量表和 UDT 不用下载到 CPU。生成项目时会在块文件夹中自动生成一个空的组织块 OB1。

在用户程序中可以调用系统功能(SFC)和系统功能块(SFB)，但是用户不能编写或修改 SFC 和 SFB。

选中最上层的项目图标后，用菜单命令“Insert”→“Station”插入新的站，用类似的方法插入程序和逻辑块等。也可以用鼠标右键点击项目图标，在弹出的菜单中选择插入站。

STEP 7 的鼠标右键功能是很强的，用右键点击图 4-1 中的某一对象，在弹出的菜单中选择某一菜单项，可以执行相应的操作。建议在使用软件的过程中逐渐熟悉右键功能，并充分利用它。

用户生成的变量表(VAT)在调试用户程序时用于监视和修改变量。系统数据块(SDB)中的系统数据含有系统组态和系统参数的信息，它是用户进行硬件组态时提供的数据自动生成的。

除了系统数据块，用户程序中其他的块都需要用相应的编辑器进行编辑。这些编辑器在双击相应的块时自动打开。

4.2.2 硬件组态

1. 硬件组态的任务

在 PLC 控制系统设计的初期，首先应根据系统的输入、输出信号的性质和点数，以及对控制系统的功能要求，确定系统的硬件配置，例如 CPU 模块与电源模块的型号，需要哪些输入/输出模块(即信号模块 SM)、功能模块(FM)和通信处理器模块(CP)，各种模块的型号和每种型号的块数等。对于 S7-300 来说，如果 SM、FM 和 CP 的块数超过 8 块，除了中央机架外还需要配置扩展机架和接口模块(IM)。确定了系统的硬件组成后，需要在 STEP 7 中完成硬件配置工作。

硬件组态的任务就是在 STEP 7 中生成一个与实际的硬件系统完全相同的系统，例如要生成网络、网络中各个站的机架和模块，以及设置各硬件组成部分的参数，即给参数赋值。所有模块的参数都是用编程软件来设置的，完全取消了过去用来设置参数的硬件 DIP 开关。硬件组态确定了 PLC 输入/输出变量的地址，为设计用户程序打下了基础。

组态时设置的 CPU 的参数保存在系统数据块 SDB 中，其他模块的参数保存在 CPU 中。在 PLC 启动时 CPU 自动地向其他模块传送设置的参数，因此在更换 CPU 之外的模块后不需要重新对它们赋值。

PLC 在起动时，将 STEP 7 中生成的硬件设置与实际的硬件配置进行比较，如果二者不符，将立即产生错误报告。

模块在出厂时带有预置的参数，或称为默认的参数，一般可以采用这些预置的参数。通过多项选择和限制输入的数据，系统可以防止不正确的输入。

对于网络系统，需要对以太网、PROFIBUS-DP 和 MPI 等网络的结构和通信参数进行组态，将分布式 I/O 连接到主站。例如可以将 MPI(多点接口)通信组态为时间驱动的循环数据传送或事件驱动的数据传送。

对于硬件已经装配好的系统，用 STEP 7 建立网络中各个站对象后，可以通过通信从 CPU 中读出实际的组态和参数。

2. 硬件组态的步骤

(1) 生成站，双击“Hardware”图标，进入硬件组态窗口；

(2) 生成机架，在机架中放置模块；

(3) 双击模块，在打开的对话框中设置模块的参数，包括模块的属性和 DP 主站和从站的参数；

(4) 保存硬件设置，并将它下载到 PLC 中去。

在项目管理器左边的树中选择 SIMATIC 300 Station(站)对象(见图 4-1)，双击工作区中的“Hardware”(硬件)图标，进入“HW Config”(硬件组态)窗口(见图 4-4)。

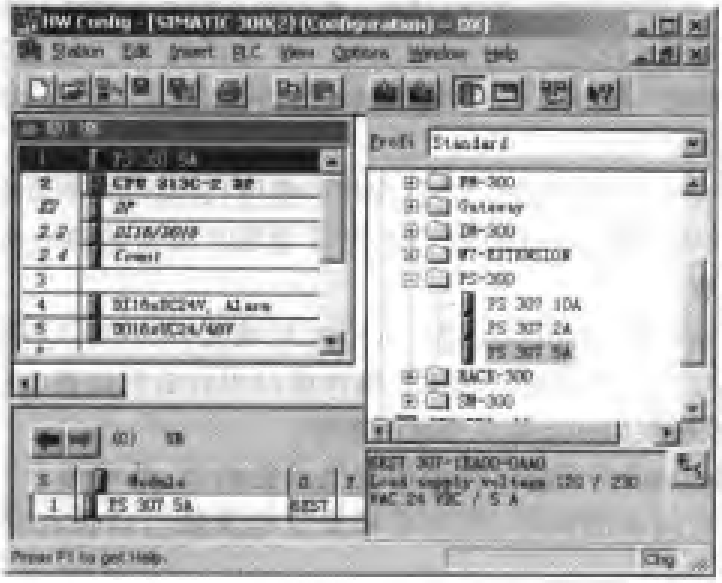


图 4-4 S7-300 的硬件组态窗口

图 4-4 左上部的窗口是一个组态简表，它下面的窗口列出了各模块详细的信息，例如订货号、MPI 地址和 I/O 地址等。右边是硬件目录窗口，可以用菜单命令“View”→“Catalog”打开或关闭它。左下角的窗口中向左和向右的箭头用来切换机架。

组态时用组态表来表示机架，可以用鼠标将右边硬件目录中的元件“拖放”到组态表的某

一行中，就好像将真正的模块插入机架上的某个槽位一样。也可以双击硬件目录中选择的硬件，它将被放置到组态表中预先被鼠标选中的槽位上。

用鼠标右键点击某一 I/O 模块，在出现的菜单中选择“Edit Symbolic names”，可以打开和编辑该模块的 I/O 元件的符号表。

3. 硬件组态举例

对站对象组态时，首先从硬件目录窗口中选择一个机架，S7-300 应选硬件目录窗口文件夹 \ SIMATIC 300 \ RACK-300 中的导轨(Rail)。

在硬件目录中选择需要的模块，将它们安排在机架中指定的槽位上。

S7-300 中央机架(Slot 0)的电源模块占用 1 号槽，CPU 模块占用 2 号槽，3 号槽用于接口模块(或不用)，4～11 号槽用于其他模块。

以在 1 号槽配置电源模块为例，首先选中 1 号槽，即用鼠标单击左边 0 号中央机架 UR 的 1 号槽(表格中的第 1 行)，使该行的显示内容反色，背景变为深蓝色。然后在右边硬件目录窗口中选择 \ SIATIC 300 \ PS 300，目录窗口下面的灰色小窗口中将会出现选中的电源模块的订货号和详细的信息。

用鼠标双击目录窗口中的“PS 307 5A”，1 号槽所在的行将会出现“PS 307 5A”，该电源模块就被配置到 1 号槽了。

也可以用鼠标左键点击并按住右边硬件目录窗口中选中的模块，将它“拖”到左边窗口中指定的行，然后放开鼠标左键，该模块就被配置到指定的槽了。

用同样的方法，在文件夹 \ SIATIC 300 \ CPU-300 中选择 CPU 313C-2DP 模块，并将后者配置到 2 号槽。因为没有接口模块，3 号槽空置。在 4 号槽配置 16 点 DC 24V 数字量输入模块(DI)，在 5 号槽配置 16 点继电器输出模块(DO)。它们属于硬件目录的 \ SIMATIC 300 \ SM-300 子目录中 S7-300 的信号模块(SM)。

双击左边机架中的某一模块，打开该模块的属性窗口后，可以设置该模块的属性。硬件设置结束后应保存和下载到 CPU 中。

STEP 7 根据模块在组态表中的位置(即模块的槽位)自动地安排模块的默认地址，例如图 4-4 中的数字量输入模块的地址为 IB0 和 IB1，数字量输出模块的地址为 QB4 和 QB5。用户可以修改模块默认的地址。

执行菜单命令“View”→“Address Overview”(地址概况)或点击工具条中的地址概况按钮(图 4-4 中工具条内右起第 3 个按钮)，在地址概况窗口中将会列出各 I/O 模块所在的机架号(R)和插槽号(S)，以及模块的起始地址和结束地址。

执行菜单命令“Station”→“Save”可以保存当前的组态，菜单命令“Station”→“Save and Compile”在保存组态和编译的同时，把组态和设置的参数自动保存到生成的系统数据块(SDB)中。

4.2.3 CPU 模块的参数设置

S7-300/400 各种模块的参数用 STEP 7 编程软件来设置。在 STEP 7 的 SIMATIC 管理器中点击“hardware”(硬件)图标，进入“HW Config”(硬件组态)画面后，双击 CPU 模块所在的行，在弹出的“Properties”(属性)窗口中点击某一选项卡，便可以设置相应的属性。下面以 S7 313-2DP 为例，介绍 CPU 主要参数的设置方法。

1. 起动特性参数

在"Properties"窗口中点击"Startup"(启动)选项卡(见图 4-5),设置启动特性。

用鼠标点击某小正方形的检查框,框中出现一个"√",表示选中(激活)了该选项,再点击一下,"√"消失,表示没有选中该选项,该选项被禁止。

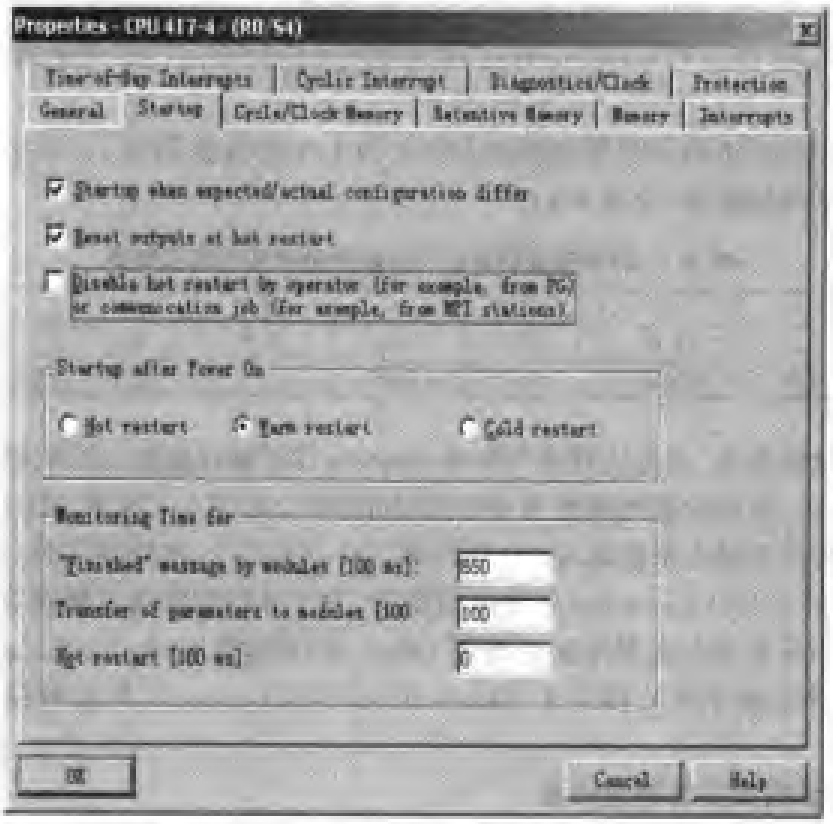


图 4-5 CPU 属性设置对话框

如果没有选中检查框"Startup if preset configuration not equal to actual configuration"(预设置的组态不等于实际的组态时启动),并且至少一个模块没有插在组态时指定的槽位,或者某个槽插入的不是组态的模块,CPU 将进入 STOP 状态。

如果选择了该检查框,即使有上述的问题,CPU 也会启动,除了 PROFIBUS-DP 接口模块外,CPU 不会检查 I/O 组态。

检查框"Reset outputs on hot restart"(热启动时复位输出)和"Disable hot restart by operator..."(禁止操作员热启动)仅用于 S7-400。

在"Startup after Power On"(接通电源后的启动)区,可以选择单选框"Hot restart"(热启动),"Warm restart"(暖启动)和"Cold restart."(冷启动)。

电源接通后,CPU 等待所有被组态的模块发出"完成信息"的时间如果超过"Finished message from modules(100ms)"选项设置的时间,表明实际的组态不等于预置的组态。该时间的设置范围为 1~650,单位为 100ms,默认值为 650。

"Transfer of parameters to modules (100 ms)"(参数传送到模块)是 CPU 将参数传送给模块的最大时间,单位为 100ms。对于有 DP 主站接口的 CPU,可以用这个参数来设置 DP 从站起动的监视时间。如果超过了上述的设置时间,CPU 按"Startup if preset configuration not e-

qual to actual configuration"的设置进行处理。

2. 时钟存储器

在"Properties"窗口中点击"Cycle/Clock Memory"(循环/时钟存储器)选项卡,可以设置"Scan cycle monitoring time"(以 ms 为单位的扫描循环监视时间),默认值为 150ms。如果实际的循环扫描时间超过设定的值,CPU 将进入 STOP 模式。

"Scan Cycle Load from Communication"用来限制通信处理占扫描周期的百分比,默认值为 20%。

时钟脉冲是一些可供用户程序使用的占空比为 1:1 的方波信号,一个字节的时钟存储器的每一位对应一个时钟脉冲(见表 4-1)。

表 4-1 时钟存储器各位对应的时钟脉冲周期与频率

位	7	6	5	4	3	2	1	0
周期/s	2	1.6	1	0.8	0.5	0.4	0.2	0.1
频率/Hz	0.5	0.625	1	1.25	2	2.5	5	10

如果要使用时钟脉冲,首先应选中"Clock memory"(时钟存储器)选项,然后设置时钟存储器(M)的字节地址。假设设置的地址为 100(即 MB100),由表 4-1 可知,M100.7 的周期为 2s,如果用 M100.7 的常开触点来控制 Q0.0 的线圈,Q0.0 将以 2s 的周期闪烁(亮 1s,熄灭 1s)。

"OB85-Call up at I/O access error"用来预设置 CPU 对系统修改过程映像时发生的 I/O 访问错误的响应。如果希望在出现错误时调用 OB85,建议选择"Only for incoming and outgoing errors"(仅在错误产生和消失),相对于"On each individual access"(每次单独的访问),不会增加扫描循环时间。

3. 系统诊断参数与实时钟的设置

系统诊断是指对系统中出现的故障进行识别、评估和作出相应的响应,并保存诊断的结果。通过系统诊断可以发现用户程序的错误、模块的故障和传感器、执行器的故障等。

在"Properties"窗口中点击"Diagnostics / Clock"(诊断与时钟)选项卡,可以选择"Report cause of STOP"(报告引起 STOP 的原因)等选项。

在某些大系统(例如电力系统)中,某一设备的故障会引起连锁反应,相继发生一系列事件,为了分析故障的起因,需要查出故障发生的顺序。为了准确地记录故障顺序,系统中各计算机的实时钟必须定期作同步调整。

可以用下面 3 种方法使实时钟同步(见图 4-6):"In the PLC"(在 PLC 内部)、"On MPI"(通过 MPI 接口)和"On MFI"(通过第二个接口)。每个设置方法有 3 个选项,"As Master" 是指用该 CPU 模块的实时钟作为标准时钟,去同步别的时钟;"As Slave" 是指该时钟被别的时钟同步,"None"为不同步。

图 4-6 时钟同步的设置

"Time Intervals"是时钟同步的周期，从1s～24h，一共有7个选项可供选择。

"Correction factor"是对每24h时钟误差时间的补偿(以ms为单位)，可以指定补偿值为正或为负。例如当实时钟每24h慢3s时，校正因子应为 +3000ms。

4. 保持区的参数设置

在电源掉电或CPU从RUN模式进入STOP模式后，其内容保持不变的存储区称为保持存储区。CPU安装了后备电池后，用户程序中的数据块总是被保护的。

"Retentivity Memory"(保持存储器)页面的"Number of memory bytes from MB0"，"Number of S7 timers from T0"和"Number of S7 counters from C0"分别用来设置从MB0、T0和C0开始的需要断电保持的存储器字节数、定时器和计数器的数量，设置的范围与CPU的型号有关，如果超出允许的范围，将会给出提示。没有电池后备的S7-300可以在数据块中设置保持区域。

5. 保护级别的选择

在"Protection"(保护)页面的"Protection Level"(保护级别)框中，可以选择3个保护级别：

- 保护级别1是默认的设置，没有口令。CPU的钥匙开关(工作模式选择开关)在RUN-P和STOP位置时对操作没有限制，在RUN位置只允许读操作。S7-31xC系列CPU没有钥匙开关，运行方式开关只有RUN和STOP两个位置。
- 被授权(知道口令)的用户可以进行读写访问，与钥匙开关的位置和保护级别无关。
- 对于不知道口令的人员，保护级别2只能读访问，保护级别3不能读写，均与钥匙开关的位置无关。

在执行在线功能之前，用户必须先输入口令：

(1) 在SIMATIC管理器中选择被保护的模块或它们的S7程序。

(2) 选择菜单命令"PLC"→"Access Rights"→"Setup"，在对话框中输入口令。

输入口令后，在退出用户程序之前，或取消访问权利之前，访问权一直有效。

6. 运行方式的选择

在"Protection"(保护)页面的"Process Mode"(处理模式)区中，可以选择：

(1) Operation(运行模式)：测试功能(例如程序状态或监视/修改变量)是被限制的，不允许断点和单步方式。

(2) Test(测试模式)：允许通过编程软件执行所有的测试功能，这可能引起扫描循环时间显著的增加。

7. 日期-时间中断参数的设置

大多数CPU有内置的实时钟，可以产生日期-时间中断，中断产生时调用组织块OB10～OB17。在"Time-Of-Day Interrupts"(日期-时间中断)选项卡，可以设置中断的优先级(Priority)，通过"Active"选项决定是否激活中断，选择执行方式(Execution)有执行一次(Once)，每分钟、每小时、每天、每星期、每月、每年执行一次。可以设置启动的日期(Start date)和时间(Time of)，以及要处理的过程映像分区(仅用于S7-400)。

8. 循环中断参数的设置

在"Cyclic Interrupts"页面，可以设置循环执行组织块OB30～OB38的参数，包括中断的优先级(Priority)、执行的时间间隔(Execution，以ms为单位)和相位偏移(Phase offset，仅用于S7-400)。相位偏移用于将几个中断程序错开来处理。

9. 中断参数的设置

在"Interrupts"页面，可以设置硬件中断(Hardware Interrupts)、延迟中断(Time-Delay Interrupts)、DPV1(PROFIBUS-DP)中断和异步错误中断(Asynchronous Error Interrupts)的参数。

S7-300 不能修改当前默认的中断优先级。S7-400 根据处理的硬件中断 OB 可以定义中断的优先级。默认的情况下，所有的硬件中断都由 OB40 来处理。可以用优先级"0"删掉中断。

PRIFIBUS-DPV1 从站可以产生一个中断请求，以保证主站 CPU 处理中断触发的事件。

10. 通信参数的设置

在"Communication"(通信)选项卡中，需要设置 PG(编程器或计算机)通信、OP(操作员面板)通信和 S7 standard(标准 S7)通信使用的连接的个数。至少应该为 PG 和 OP 分别保留 1 个连接。

11. DP 参数的设置

对于有 PRIFIBUS-DP 通信接口的 CPU 模块，例如 CPU 313C-2DP，双击图 4-4 中左边窗口内 DP 所在行(第 3 行)，在弹出的 DP 属性窗口中的"General"(常规)选项卡(见图 4-7)中点击"Interface"栏中的[Properties]按钮，可以设置站地址或 DP 子网络的属性，生成或选择其他子网络。

在"Addresses"(地址)选项卡中，可以设置 DP 接口诊断缓冲区的地址，如果选择"System selection"，由系统自动指定地址。

在"Operation Mode"选项卡中，可以选择 DP 接口作 DP 主站(master)或 DP 从站(slave)。

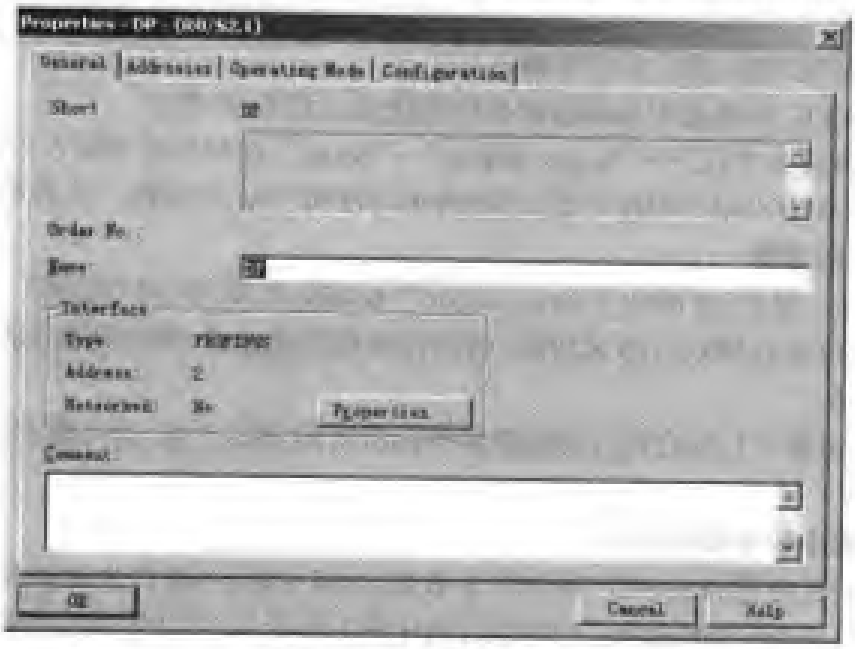


图 4-7 DP 接口属性的设置

在"Configuration"选项卡中，可以组态主-从通信(Master-Slave，MS)方式或直接数据交换(Direct Data Exchange，DX)方式，详细的设置方法将在第 8 章中介绍。

12. 集成 I/O 参数的设置

CPU 313-2DP 有集成的 16DI(数字量输入)、16DO(数字量输出)。双击图 4-4 左边窗口第 4 行中的"16DI/16DO"，可以设置集成 DI 和集成 DO 的参数，设置的方法与普通的 DI、DO

的设置方法基本上相同。

在"Addresses"(地址)选项卡中,集成 DI 的默认地址为 IB124 和 IB125,集成 DO 的默认地址为 QB124 和 QB125,用户可以修改它们的地址。

点击"Input"选项卡,可以设置是否允许各集成的 DI 点产生硬件中断(Hardware Interrupt)。可以逐点选择上升沿中断(rising edge)或下降沿中断(falling edge)。

输入延迟时间可以抑制输入触点接通或断开时的抖动的不良影响。可以按每 4 点一组设置各组的输入延迟时间(Input delay,以 ms 单位)。点击某一组的延迟时间输入框,在弹出的菜单中选择延迟时间。

4.2.4 数字量输入模块的参数设置

输入/输出模块的参数在 STEP 7 中设置,参数设置必须在 CPU 处于 STOP 模式下进行。设置完所有的参数后,应将参数下载到 CPU 中。当 CPU 从 STOP 模式转换为 RUN 模式时,CPU 将参数传送到每个模块。

参数分为静态参数和动态参数,可以在 STOP 模式下设置动态参数和静态参数,通过系统功能 SFC,可以修改当前用户程序中的动态参数。但是在 CPU 由 RUN 模式进入 STOP 模式,然后又返回 RUN 模式后,将重新使用 STEP 7 设定的参数。

在 STEP 7 的 SIMATIC 管理器中点击"hardware"(硬件)图标,进入"HW Config"(硬件组态)画面(见图 4-4)。双击图中左边机架 4 号槽中的"DI16×DC 24V"(订货号为 6ES7 321-7BH00-0AB0),出现图 4-8 所示的属性(Properties)窗口。点击"Addresses"(地址)选项卡,可以设置模块的起始字节地址。

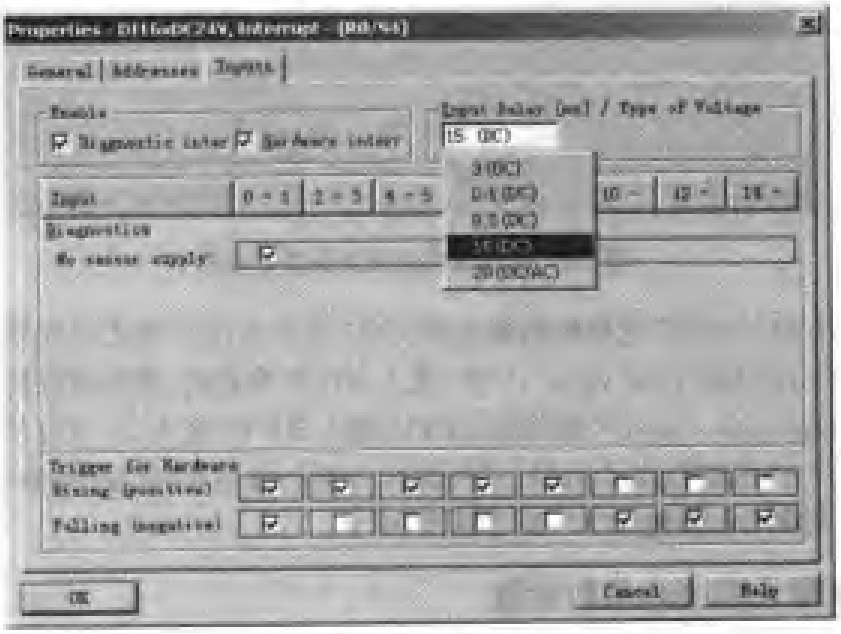


图 4-8 数字量输入模块的参数设置

点击"Inputs"选项卡,用鼠标点击检查框(check box),可以设置是否允许产生硬件中断(Hardware Interrupt)和诊断中断(Diagnostics Interrupt)。检查框内出现"√"表示允许产生中断。

模块给传感器提供带熔断器保护的电源。以 8 点为单位，可以设置是否诊断传感器电源丢失。传感器电源丢失时，模块将这个诊断事件写入诊断数据区，用户程序可以用系统功能 SFC 51 读取系统状态表中的诊断信息。

选择了允许硬件中断后，以组为单位（每组两个输入点），可以选择上升沿中断（Rising）、下降沿中断（Falling）或上升沿和下降沿均产生中断。出现硬件中断时，CPU 的操作系统将调用组织块 OB 40。

点击“Input Delay”（输入延迟）输入框，在弹出的菜单中选择以 ms 为单位的整个模块的输入延迟时间，有的模块可以分组设置延迟时间。

4.2.5 数字量输出模块的参数设置

双击图 4-4 左边窗口 5 号槽中的“DO16（DC 24V”（订货号为 6ES7 322-5GH00-0AB0），出现图 4-9 所示的属性（Properties）窗口。点击“Outputs”选项卡，用鼠标点击检查框可以设置是否允许产生诊断中断（Diagnostics Interrupt）。

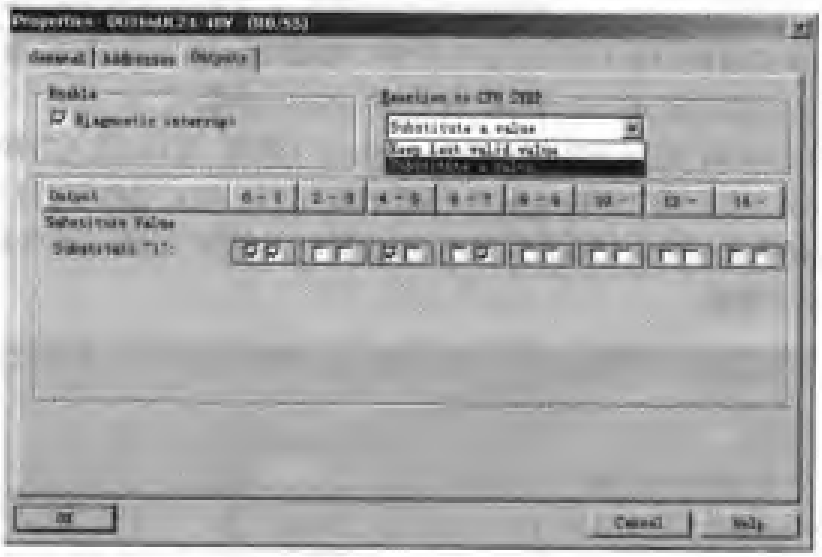


图 4-9 数字量输出模块的参数设置

“Reaction to CPU STOP”选择框用来选择 CPU 进入 STOP 模式时模块各输出点的处理方式。如果选择“Keep last valid value”，CPU 进入 STOP 模式后，模块将保持最后的输出值。

如果选择“Substitute a value”（替代值），CPU 进入 STOP 模式后，可以使各输出点分别输出“0”或“1”。窗口中间的“Substitute“1”：”所在行中某一输出点对应的检查框如果被选中，进入 STOP 模式后该输出点将输出“1”，反之输出“0”。

4.2.6 模拟量输入模块的参数设置

1. 模块诊断与中断的设置

图 4-10 是 8 通道 12 位模拟量输入模块（订货号为 6ES7 331-7KF02-0AB0）的参数设置对话框。点击“Inputs”（输入）选项卡，可以设置是否允许诊断中断和模拟值超过限制值的硬件中断，有的模块还可以设置模拟量转换的循环结束时的硬件中断和断线检查。如果选择了超限中断，窗口下面的“High limit”（上限）和“Low limit”（下限）由灰变白，可以设置通道 0 和通

道 1 产生超限中断的上限值和下限值。每两个通道为一组，可以设置是否对各组进行诊断。

图 4-10　模拟量输入模块的参数设置

2. 模块测量范围的选择

可以分别对模块的每一通道组选择允许的任意量程，每两个通道为一组。例如在“Inputs”选项卡中点击 0 号和 1 号通道的测量种类输入框，在弹出的菜单中选择测量的种类，图中选择的“4DMU”是 4 线式传感器电流测量；“R-4L”是 4 线式热电阻；“TC-I”是热电偶；“E”表示测量种类为电压。

如果未使用某一组的通道，应选择测量种类中的“Deactivated”（禁止使用），以减小模拟量输入模块的扫描时间。点击测量范围输入框，在弹出的菜单中选择量程，图中第一组的测量范围为 4－20 mA。量程框的下面的“[C]”表示 0 号和 1 号通道对应的量程卡的位置应设置为“C”（见图 2-13），即量程卡上的“C”旁边的三角形箭头应对准输入模块上的标记。在选择测量种类时，应保证量程卡的位置与 STEP 7 中的设置一致。

3. 模块测量精度与转换时间的设置

SM 331 采用积分式 A/D 转换器，积分时间直接影响到 A/D 转换时间、转换精度和干扰抑制频率。积分时间越长，精度越高，快速性越差。积分时间与干扰抑制频率互为倒数。在积分时间为 20ms 时，对 50Hz 的干扰噪声有很强的抑制作用。为了抑制工频频率，一般选用 20ms 的积分时间。

SM 331 的转换时间由积分时间、电阻测量的附加时间（1 ms）和断线监视的附加时间（10 ms）组成。以上均为每一通道的处理时间，如果一块模块中使用了 N 个通道，总的转换时间（称为循环时间）为各个通道的转换时间之和。

模拟量输入模块 6ES7 331-7KF02 的积分时间、干扰抑制频率、转换时间和转换精度的关系如表 4-2 所示。点击图 4-10 中“积分时间”所在行最右边的“integration time”（积分时间）所在的方框，在弹出的菜单内选择按积分时间设置或按干扰抑制频率来设置参数。点击某一组的积分时间设置框后，在弹出的菜单内选择需要的参数。

表 4-2　6ES7 331-7KF02 模拟量输入模块的参数关系

积分时间/ms	2.5	16.7	20	100
基本转换时间/ms(包括积分时间)	3	17	22	102
附加测量电阻转换时间/ms	1	1	1	1
附加开路监控转换时间/ms	10	10	10	10
附加测量电阻和开路监控转换时间/ms	16	16	16	16
精度/bit(包括符号位)	9	12	12	14
干扰抑制频率/Hz	400	60	50	10
模块的基本响应时间/ms(所有通道使能)	24	136	176	816

4. 设置模拟值的平滑等级

有些模拟量输入模块用 STEP 7 设置模拟值的平滑等级。模拟值的平滑处理可以保证得到稳定的模拟信号。这对于缓慢变化的模拟值(例如温度测量值)是很有意义的。

平滑处理用平均值数字滤波来实现,即根据系统规定的转换次数来计算转换后的模拟值的平均值。用户可以在平滑参数的四个等级(无,低,平均,高)中进行选择。这四个等级决定了用于计算平均值的模拟信号数量。所选的平滑等级越高,平滑后的模拟值越稳定,但是测量的快速性越差。随书光盘中的 S7-300 模板规范参考手册给出了模拟量四个平滑等级的阶跃响应曲线。

4.2.7　模拟量输出模块的参数设置

模拟量输出模块的设置与模拟量输入模块的设置有很多类似的地方。模拟量输出模块可能需要设置下列参数:

(1) 确定每一通道是否允许诊断中断。

(2) 选择每一通道的输出类型为"Deactivated"(关闭)、电压输出或电流输出。选定输出类型后,再选择输出信号的量程。

(3) CPU 进入 STOP 时的响应:可以选择不输出电流电压(0CV)、保持最后的输出值(KLV)和采用替代值(SV)。

4.3　符号表与逻辑块

4.3.1　符号表

1. 符号地址

在程序中可以用绝对地址(例如 I0.3)访问变量,但是使用符号地址可使程序更容易阅读和理解。共享符号(全局符号)在符号表中定义,可供程序中所有的块使用。

在符号表中定义了符号地址后,STEP 7 可以自动地将绝对地址转换为符号地址。例如在符号表中定义 I1.0 为"起动汽油机",在程序中就可以用"起动汽油机"来代替地址 I1.0。可以设置在输入地址时自动起动一个弹出式的地址表,在地址表中选择要输入的地址,双击它就可以完成该地址的输入。也可以直接输入符号地址或绝对地址,如果选择了显示符号地址,输入

绝对地址后，将自动地转换为符号地址。

在梯形图、功能块图及语句表这三种编程语言中，都可以使用绝对地址或符号来输入地址、参数和块。

2. 生成与编辑符号表

点击管理器左边的"S7 Program"图标，右边的工作区将出现"Symbols"（符号表）图标，双击它后进入符号表窗口（见图 4-11）。CPU 将自动地为程序中的全局符号加双引号，在局部变量的前面自动加"#"号。生成符号表和块的局域变量表时用户不用为变量添加引号和#号。打开某个块后，可以用菜单命令"View"→"Display with"→"Symbolic Representation"选择显示符号地址或显示绝对地址。

图 4-11 符号表

在符号表中需要输入符号（Symbol）和地址（Address），符号不能多于 24 个字符。

数据块中的地址（DBD、DBW、DBB 和 DBX）不能在符号表中定义。它们的名字应在数据块的声明表中定义。

组织块（OB）、系统功能块（SFB）和系统功能（SFC）已预先被赋予了符号名，编辑符号表时可以引用这些符号名。

输入地址后，软件将自动添加数据类型（Data type），用户也可以修改它。如果所作的修改不适合该地址或存在语法错误，在退出该区域时会显示一条错误信息。

注释"Comment"是可选的输入项，简短的符号名与更详细的注释混合使用，使程序更易于理解，注释最长 80 个字符。输入完后需保存符号表。

用菜单命令"View"→"Columns R, O, M, C, CC"可以选择是否显示表中的"R, O, M, C, CC"列，它们分别表示监视属性、在 WinCC 里是否被控制和监视、信息属性、通信属性和触点控制。用菜单命令"Edit"→"Special Object Properties"选择打开或关闭某一对象属性。

各种块的名称可以在符号表中定义，也可以在生成块时定义。在符号表中，用菜单命令"View"→"Filter"可以筛选符号显示的内容。

可以用菜单命令"View"→"Sort"选择符号表中变量的排序方法。

用菜单命令"Symbol Table"→"Import/ Export"（导入/导出），可将当前符号表存入文本文件，用文本编辑器进行编辑。可以导出整个符号表或导出选择的若干行，也可将其他应用程序生成的符号表导入当前的符号表。

在管理器中选择块文件夹，执行"Edit"→"Object Properties"菜单命令，在"Address Priority"选项卡中，可以选择符号（Symbolic）优先或绝对地址（Absolute）优先。如果选择符号优先，修改了符号表中某个变量的地址后，变量保持其符号不变。

用下述方法可以在编程时输入单个共享符号：在程序中选中使用绝对地址的某元件，用菜单“Edit”→“Symbols”编辑它，新变量会自动进入总的符号表。

3. 共享符号与局域符号

(1) 共享符号

共享符号可以被所有的块使用，在所有的块中的含义是一样的。在整个用户程序中，同一个共享符号不能定义两次或多次。共享符号由字母、数字及特殊字符组成，可以用汉字来表示共享符号。可以为 I、Q、PI、PQ、M、T、C、FB、FC、SFB、SFC、DB、UDT(用户定义的数据类型)和 VAT(变量表)定义符号。

(2) 局域符号

局域符号在某个块的变量声明表中定义，局域符号只在定义它的块中有效，同一个符号名可以在不同的块中用于不同的局域变量。局域符号只能使用字母、数字和下划线，不能使用汉字。可以为块参数(输入、输出及输入/输出参数)、块的静态数据(STAT)和块的临时数据(TEMP)定义局域符号。

4. 过滤器(Filter)

过滤器用来有选择地显示部分符号。在符号表中执行菜单命令“View”→“Filter”，在打开的对话框中，可以按以下标准进行过滤：

(1) 按符号名称、地址、数据类型和注释进行过滤

例如在“Address”(地址)属性中，“I＊”表示显示所有的输入，“I＊.＊”表示所有的输入位，“I2.＊”表示 IB2 中的位等。

(2) 对具有某些属性的符号进行过滤

例如对监控、操作员控制及监控、通信、报文(Message)用的符号进行过滤，选择“＊”，“YES”和“NO”可以选择显示所有的符号、显示符合条件的符号和显示不符合条件的符号。

(3) “Valid”和“Invalid”分别只显示有效的符号或无效的符号(不是惟一的、不完整的符号)。

只有满足条件的数据才能出现在过滤后的符号表中，几种过滤条件可以结合起来同时使用。

4.3.2 逻辑块

1. 逻辑块的组成

逻辑块包括组织块 OB、功能块 FB 和功能 FC。逻辑块由变量声明表、程序指令和属性组成。

(1) 变量声明表：在变量声明表中，用户可以设置变量的各种参数，例如变量的名称、数据类型、地址和注释等。

(2) 程序指令：在程序指令部分，用户编写能被 PLC 执行的指令代码。可以用梯形图(LAD)、功能块图(FBD)或语句表(STL)来生成程序指令。

(3) 块属性：块属性中有块的信息，例如由系统自动输入的时间标记和存放块的路径。此外用户可以输入块名、系列名、版本号和块的作者等。

2. 选择程序的输入方式

根据生成程序时选用的编程语言，可以用增量输入方式或源代码方式(或称文本方式、自

由编辑方式)输入程序。

(1) 增量编辑器

编辑器适用于梯形图、功能块图、语句表以及 S7-GRAPH 等编程语言,这种编程方式适合于初学者。编辑器对输入的每一行或每个元素立即进行句法检查。只有改正了指出的错误才能完成当前的输入,检查通过的输入经过自动编译后保存到用户程序中。

必须事先定义用于语句中的符号,如果在程序块中使用没有定义的符号,该块不能完全编译,但是可以在计算机中保存。

(2) 源代码(文本)编辑器

源代码(文本)编辑器适用于语句表、S7 SCL、S7 HiGraph 编程语言,用源文件(文本文件)的形式生成和编辑用户程序,再将该文件编译成各种程序块。这种编辑方式又称为自由编辑方式,可以快速输入程序。

文本文件(源文件)存放在项目中"S7 Program"对象下的"Source File"文件夹中,一个源文件可以包含一个块或多个块的程序代码。用文本编辑器和 STL 和 SCL 来编程,生成 OB、FB、FC、DB 及 UDT(用户定义数据类型)的代码,或生成整个用户程序。CPU 的所有程序(即所有的块)可以包含在一个文本文件中。

在文件中使用的符号必须在编译之前加以定义,在编译过程中编译器将报告错误。只有将源文件编译成程序块后,才能执行句法检查功能。

3. 选择编程语言

可以选择 3 种基本编程语言:梯形图(LAD)、语句表(STL)和功能块图(FBD)。程序没有错误时,可以用"View"菜单中的命令切换这 3 种语言。STL 编写的某个网络不能切换为 LAD 和 FBD 时,仍然用语句表表示。此外还有 4 种作为可选软件包的编程语言:

S7 SCL(结构化控制)语言;S7 Graph(顺序控制)编程语言;S7 HiGraph(状态图形)编程语言和 S7 CFC(连续功能图)编程语言。

4. 用 STL 和增量式输入方式生成逻辑块的步骤

(1) 在 SIMATIC 管理器中生成逻辑块(FB、FC 或 OB)。

(2) 编辑块的变量声明表,这部分的内容将在第 6 章介绍。

(3) 编辑块的程序指令部分。

(4) 编辑块的属性。

(5) 用菜单命令"File"→"Save"保存块。

5. 生成逻辑块

在 SIMATIC 管理器中用菜单命令"Insert"→"S7 Block"生成逻辑块,也可以用右键点击管理器中右边的块工作区,在弹出的菜单中选择命令"Insert New Object"(插入新的对象),生成新的块。双击工作区中的某一个块,将进入程序编辑器。

程序指令部分(见图 4-12 右下部分的窗口)以块标题和块注释开始。在程序指令部分的代码区,用户通过输入 STL 的语句或图形编程语言中的元素来组成逻辑块中的程序。输入完一条语句或一个图形元素后,编辑器立即起动句法检查,发现的错误用红色斜体字符显示。

用菜单命令"View"→"Toolbar"可以打开或关闭工具条。点击工具条上的触点图标,将在光标所在的位置放置一个触点,放置线圈的方法与此相同。点击触点或线圈上面的红色问号"??.?",输入该元件的绝对地址或符号地址。点击工具条上中间有两个问号的指令框图标,

在出现的下拉式菜单中选择需要输入的指令，也可以在最上面的文本输入框内直接输入指令助记符。放置指令框后，点击同时出现的红色问号"?? .?"，输入绝对地址、符号地址或其他参数。点击带箭头的转折线，可以生成分支电路或并联电路。

用菜单命令"View"→"Overview"可以打开或关闭指令的分类目录(见图 4-12 右边的窗口)，可以直接使用目录中的指令。例如在"Timer"(定时器)文件夹中找到 SD 线圈(接通延时定时器线圈)后，用鼠标左键双击它，就可以将它放置在梯形图内光标所在的位置。也可以用鼠标"拖放"的方法将它"拖"到梯形图中某个地方，即用鼠标左键点击并按住它，将它"拖"到需要的地方后再放开它。如果元件被放置到错误的位置，将会给出提示信息。

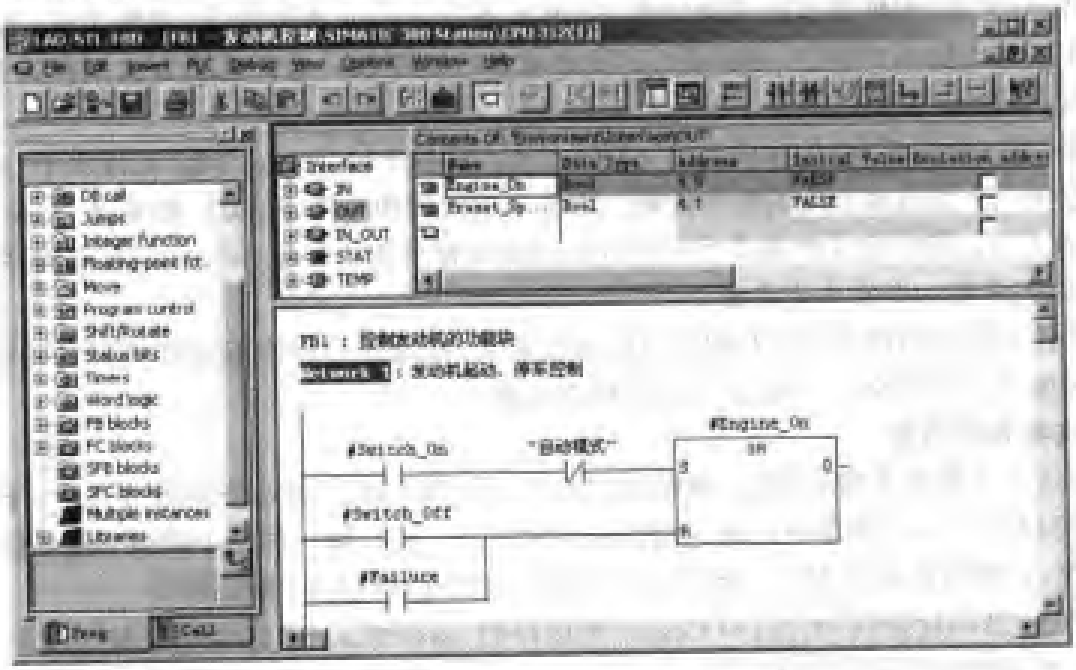


图 4-12　梯形图编辑器

6. 网络

程序被划分为若干个网络(Network)，在梯形图中，每块独立电路就是一个网络，每个网络都有网络编号。如果在一个网络中放置一个以上的独立电路，编译时将会出错。

执行菜单命令"Insert"→"Network"，或双击工具条中的"New Network"图标，可以在用鼠标选中的当前网络的下面生成一个新的网络。

每个网络由网络编号(例如 Network 1)开始，网络标题在网络编号的右边，网络注释在网络标题的下面。网络注释下面的语句或图形是网络的主体。

点击网络标题域或网络注释域，打开文字输入框，可以输入标题或注释，标题最多由 64 个字符组成。可以用菜单命令"View"→"Display"→"Comments"来激活或取消块注释和网络注释。

可以用剪贴板在块内部和块之间复制和粘贴网络，按住<Ctrl>键，用鼠标可以选中多个需要同时复制的网络。

7. 打开和编辑块的属性

可以在生成块时编辑块的属性，生成块后可以在块编辑器中用菜单命令"File"→"Properties"来查看和编辑块属性。块属性使用户更容易识别生成的各程序块，还可以对程序块加以保护，防止非法修改。

8. 程序编辑器的设置

进入程序编辑器后用菜单命令“Option”→“Customize”打开对话框，可以进行下列设置：

(1) 在“General”选项卡的“Font”窗口点击按钮【Select】，设置编辑器使用的字体和字符的大小。

(2) 在“STL”(语句表)选项卡和“LAD/FDB”(梯形图/功能块图)选项卡中选择这些程序编辑器的显示特性。在梯形图编辑器中还可以设置地址域的宽度(Address Field Width)，即触点或线圈所占的字符数。

(3) 在“Block”(块)选项卡中，可以选择生成功能块时是否同时生成参考数据、功能块是否有多重背景功能，还可以选择编程语言。

(4) 在“View”选项卡中的“View after Open Block”区，选择在块刚刚被打开时显示的方式，例如是否需要显示符号信息，是否需要显示符号地址等。

9. 显示方式的设置

执行“View”菜单中的“Zoom In”和“Zoom Out”命令，可以放大、缩小梯形图或功能块图的显示比例，“Zoom Factor...”命令可以任意设置显示比例。

使用菜单命令“View”→“Display”→“Symbolic Representation”，可以在绝对地址和符号地址两种显示方式之间进行切换。

为了方便程序的编写和阅读，可以用符号信息(Symbol Information)来说明网络中使用的符号的绝对地址和符号的注释，但是不能编辑符号信息，对符号信息的修改需要在符号表或块的变量声明表中进行。菜单命令“View”→“Display”→“Symbol information”用来打开或关闭符号信息。

在梯形图的下面显示网络中使用的符号信息(见图 4-13)。在指令表中每条语句的右边显示在该语句中使用的符号信息。

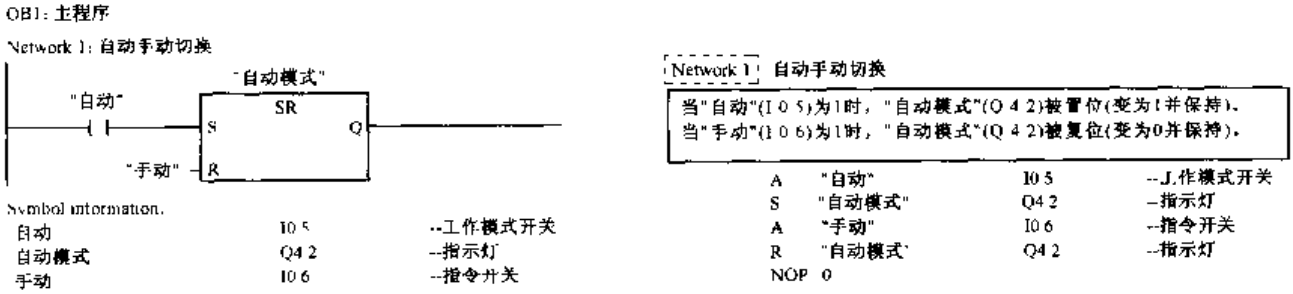


图 4-13 符号信息的显示

在输入指令中的地址时，用右键点击要输入地址的位置，在弹出的窗口中执行命令“Insert Symbol”，将弹出包括共享符号和变量声明表中的符号的表，选中并双击表中的某一符号，该符号将会自动写入指令中。可以用菜单命令“View”→“Display”→“symbol selection”来触发用梯形图和功能块图输入地址时，是否自动显示已定义的符号。

4.4 S7-PLCSIM 仿真软件在程序调试中的应用

设计好 PLC 的用户程序后，需要对程序进行调试，一般用 PLC 的硬件来调试程序。在以下情况下需要对程序进行仿真调试：

(1) 设计好程序后,PLC 的硬件尚未购回;

(2) 控制设备不在本地,设计者需要对程序进行修改和调试;

(3) PLC 已经在现场安装好了,但是在实际系统中进行某些调试有一定的风险。

为了解决这些问题,西门子公司提供了用来代替 PLC 硬件调试用户程序的仿真软件 S7-PLCSIM,西门子的"LOGO!"可编程逻辑模块的编程软件也有仿真功能。

S7-PLCSIM 是一个功能非常强大的仿真软件,它与 STEP 7 编程软件集成在一起,用于在计算机上模拟 S7-300 和 S7-400 CPU 的功能,可以在开发阶段发现和排除错误,从而提高用户程序的质量和降低试车的费用。

因为 S7-300 /400 的硬件价格较高,一般的单位和个人都很难配备较为齐全的实验装置,所以 S7-PLCSIM 也是学习 S7-300 /400 编程、程序调试和故障诊断的有力工具。

4.4.1 S7-PLCSIM 的主要功能

STEP 7 专业版包含 S7-PLCSIM,安装 STEP 7 的同时也安装了 S7-PLCSIM。对于标准版的 STEP 7,在安装好 STEP 7 后再安装 S7-PLCSIM,S7-PLCSIM 将自动嵌入 STEP 7。

在 STEP 7 的 SIMATIC 管理器窗口中,执行菜单命令"Options"→"Simulate Modes"或直接点击仿真图标"Simulation On/Off",都能打开如图 4-15 所示的 S7-PLCSIM 仿真窗口。

S7-PLCSIM 可以在计算机上对 S7-300/400 PLC 的用户程序进行离线仿真与调试,因为 S7-PLCSIM 与 STEP 7 是集成在一起的,仿真时计算机不需要连接任何 PLC 的硬件。

S7-PLCSIM 提供了用于监视和修改程序中使用的各种参数的简单的接口,例如使输入变量变为 ON 或 OFF。和实际 PLC 一样,在运行仿真 PLC 时可以使用变量表和程序状态等方法来监视和修改变量。

S7-PLCSIM 可以模拟 PLC 的输入/输出存储器区,通过在仿真窗口中改变输入变量的 ON/OFF 状态,来控制程序的运行,通过观察有关输出变量的状态来监视程序运行的结果。

S7-PLCSIM 可以实现定时器和计数器的监视和修改,通过程序使定时器自动运行,或者手动对定时器复位。

S7-PLCSIM 还可以模拟对下列地址的读写操作:位存储器(M)、外设输入(PI)变量区和外设输出(PQ)变量区,以及存储在数据块中的数据。

除了可以对数字量控制程序仿真外,还可以对大部分组织块(OB)、系统功能块(SFB)和系统功能(SFC)仿真,包括对许多中断事件和错误事件仿真。可以对语句表、梯形图、功能块图和 S7 Graph(顺序功能图)、S7 HiGraph,S7-SCL 和 CFC 等语言编写的程序仿真。

此外,S7-PLCSIM 还可以在仿真 PLC 中使用中断组织块测试程序的特性,记录一系列的操作事件(例如对输入/输出、位存储器、定时器、计数器的操作等),并可以回放记录,从而自动测试程序。

4.4.2 快速入门

S7-PLCSIM 用仿真 PLC 来模拟实际 PLC 的运行,用户程序的调试是通过视图对象(View Objects)来进行的。S7-PLCSIM 提供了多种视图对象,用它们可以实现对仿真 PLC 内的各种变量、计数器和定时器的监视与修改。

1. 使用 S7-PLCSIM 仿真软件调试程序的步骤

(1) 在 STEP 7 编程软件中生成项目,编写用户程序。

(2) 点击 STEP 7 的 SIMATIC 管理器工具条中的[Simulation on/off]按钮,或执行菜单命令"Options"→"Simulate Modules",打开 S7-PLCSIM 窗口(见图 4-15),窗口中自动出现 CPU 视图对象。与此同时,自动建立了 STEP 7 与仿真 CPU 的连接。

(3) 在 S7-PLCSIM 窗口中用菜单命令"PLC"→"Power On"接通仿真 PLC 的电源;在 CPU 视图对象中点击 STOP 小框,令仿真 PLC 处于 STOP 模式。执行菜单命令"Execute"→"Scan Mode"→"Continuous Scan"或点击"Continuous Scan"按钮,令仿真 PLC 的扫描方式为连续扫描。

(4) 在 SIMATIC 管理器中打开要仿真的用户项目,选中"块"对象,点击工具条中的下载按钮,或执行菜单命令"PLC" → "Download",将块对象下载到仿真 PLC 中。

对于下载时的提问"Do you want to load the system data?"(你想下载系统数据吗?),一般应回答"Yes"。

(5) 点击 S7-PLCSIM 工具条中标有"I"的按钮,或执行菜单命令"Insert"→"Input Variable"(插入输入变量),创建输入 IB 字节的视图对象。用类似的方法生成输出字节 QB、位存储器 M、定时器和计数器的视图对象,输入和输出一般以字节中的位的形式显示(见图 4-15),根据被监视变量的情况确定 M 视图对象的显示格式。

(6) 用视图对象来模拟实际 PLC 的输入/输出信号,用它来产生 PLC 的输入信号,或通过它来观察 PLC 的输出信号和内部元件的变化情况,检查下载的用户程序的执行是否能得到正确的结果。

(7) 退出仿真软件时,可以保存仿真时生成的 LAY 文件及 PLC 文件,以便于下次仿真时直接使用本次的各种设置。

2. 应用举例

下面以调试电动机的控制程序(见图 4-14)为例,介绍用 S7-PLCSIM 进行仿真的步骤。

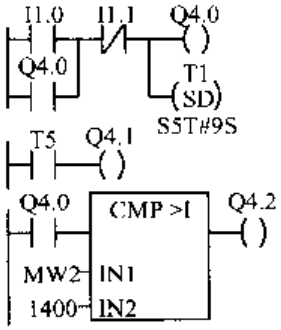


图 4-14 梯形图

OB1 中的控制程序实现下述功能:按下开机按钮 I1.0,Q4.0 变为 1 状态,电动机串电阻降压起动,同时定时器 T1 开始定时。9s 后定时时间到,Q4.1 变为 1 状态,起动电阻被短接,电动机全压运行。MW2 中电动机的实际转速与程序中预置的转速(本例中为 1400r/m)进行比较,超速时发出报警信号 Q4.2;按下停机按钮 I1.1,Q4.0 和 Q4.1 变为 0 状态,电动机停止运行。输入完程序后,将它下载到仿真 PLC。

在 PLCSIM 中创建输入字节 IB1、输出字节 QB4、位存储器 MW2 和定时器 T1 的视图对象(见图 4-15),IB1 和 QB4 以位的形式显示,MW2 以十进制形式显示。点击 CPU 视图对象中标有 RUN 或 RUN-P 的小框,将仿真 PLC 的 CPU 置于运行模式。

(1) 开机控制

给 IB1 的第 0 位(I1.0)施加一个脉冲,模拟按下启动按钮,即用鼠标点击 IB1 视图对象中第 0 位的单选框,出现符号"√",IB1.0 变为 ON;再点击一次"√"消失,IB1.0 变为 OFF,相当于放开起动按钮。

IB1.0 变为 ON 后,观察到视图对象 QB4 中的第 0 位的小框内出现符号"√",表示 Q4.0 变为 ON,即电动机开始降压起动。与此同时,视图对象 T1 的时间值由 0 变为 900(因为此时系统自动选择的时间分辨率为 10ms,900 相当于 9s),并不断减少。9s 后减为 0,定时时间到,T1 的常开触点接通,视图对象 QB4 中的第 1 位(即 Q4.1)ON,电动机全压运行。

(2) 速度监视

Q4.0 变为 ON 后，为了模拟采集到的实际转速，在 MW2 视图对象中分别输入十进制数 1399、1400 和 1401(电动机的实际转速分别低于、等于和高于预置转速)，观察到 Q4.2 的状态分别为 OFF、OFF 和 ON，说明超速报警功能正常。在 MW2 视图对象中输入数据后，需要按<Enter>键确认。

(3) 停机控制

给 I1.1 施加一个脉冲，观察到 Q4.0～Q4.2 立即变为 OFF，表示电动机停止运行。

在用 S7-PLCSIM 进行仿真时，可以同时打开 OB1 中的梯形图程序，用菜单命令"Debug"→"Monitor"在梯形图中监视程序的运行状态。

4.4.3 视图对象

1. 插入视图对象

使用"Insert"(插入)菜单或工具条上相应的按钮，可以在 PLCSIM 窗口中生成下列元件的视图对象：输入变量(I)、输出变量(Q)、位存储器(M)、定时器(T)、计数器(C)、通用变量、累加器与状态字、块寄存器、嵌套堆栈(Nesting Stacks)、垂直位变量等。它们用于访问和监视相应的数据区，可选的数据格式有位、二进制、十进制、十六进制 BCD 码、S5Time、日期时间(DATE_AND_TIME，简写为 DT)，S7 格式(例如 W#16#0)、字符和字符串。

视图对象 MW2 上的"Value"选择框用来选择设置变量的值(Value)、最大值(Max)或最小值(Min)。用鼠标拖动滑动条(Slider)上的滑动块，可以快速地设置这些值。

字节变量只能用滑动条设置十进制数(Dec)，字变量可以用滑动条设置十进制数和整数(Int)，双字变量可以用滑动条设置十进制数、整数和实数(Real)。

2. CPU 视图对象

图 4-15 中标有"CPU"的小窗口是 CPU 视图对象。开始新的仿真时，将自动出现 CPU 视图对象，用户可以用单选框来选择运行(RUN)、停止(STOP)和暂停(RUN-P)模式。

图 4-15 S7-PLCSIM 仿真窗口

选择菜单命令"PLC"→"Clear/Reset"或点击 CPU 视图对象中的 MRES 按钮，可以复位仿真 PLC 的存储器，删除程序块和硬件组态信息，CPU 将自动进入 STOP 模式。

CPU 视图对象中的 LED 指示灯"SF"表示有硬件、软件错误；"RUN"与"STOP"指示灯表示运行模式与停止模式；"DP"(分布式外设或远程 I/O)用于指示 PLC 与分布式外设或远程 I/O 的通信状态；"DC"(直流电源)用于指示电源的通断情况。用"PLC"菜单中的命令可以接通或断开仿真 PLC 的电源。

3. 其他视图对象

通用变量(Generic Variable)视图对象用于访问仿真 PLC 所有的存储区(包括数据块)。垂直位(Vertical Bits)视图对象可以用绝对地址或符号地址来监视和修改 I、Q、M 等存储区。

累加器与状态字视图对象用来监视 CPU 中的累加器、状态字和用于间接寻址的地址寄存器 AR1 和 AR2。S7-300 有两个累加器，S7-400 有 4 个累加器。

块寄存器视图对象用来监视数据块地址寄存器的内容，也可以显示当前和上一次打开的逻辑块的编号，以及块中的步地址计数器 SAC 的值。

嵌套堆栈(Nesting Stacks)视图对象用来监视嵌套堆栈和 MCR(主控继电器)堆栈。嵌套堆栈有 7 个项，用来保存嵌套调用逻辑块时状态字中的 RLO(逻辑运算结果)和 OR 位。每一项用于逻辑串的起始指令(A、AN、O、ON、X、XN)。MCR 堆栈最多可以保存 8 级嵌套的 MCR 指令的 RLO 位。

定时器视图对象和计数器视图对象用于监视和修改它们的实际值，在定时器视图对象中可以设置定时器的时间基准。视图对象和工具条内标有“T = 0”的按钮分别用来复位指定的定时器或所有的定时器。可以在“Execute”菜单中设置定时器为自动方式或手动方式。

手动方式允许修改定时器的时间值或将定时器复位，自动方式时定时器受用户程序的控制。

4.4.4 仿真软件的设置与存档

1. 设置扫描方式

S7-PLCSIM 可以用两种方式执行仿真程序：

(1) 单次扫描：每次扫描包括读外设输入、执行程序和将结果写到外设输出。CPU 执行一次扫描后处于等待状态，可以用“Execute”→“Next Scan”菜单命令执行下一次扫描。通过单次扫描可以观察每次扫描后各变量的变化。

(2) 连续扫描：这种运行方式与实际的 CPU 执行用户程序相同，CPU 执行一次扫描后又开始下一次扫描。可以用工具条中的按钮或用“Execute”菜单中的命令选择扫描方式。

2. 符号地址

为了在仿真软件中使用符号地址，使用菜单命令“Tools”→“Options”→“Attach Symbols …”，在出现的“Open”对话框的项目中找到并双击符号表(Symbols)图标。

使用菜单命令“Tools”→“Options”→“Show Symbols”，可以显示或隐藏符号地址。垂直位视图对象可以显示每一位的符号地址，其他视图对象在地址域显示符号地址。

3. 组态 MPI 地址

使用菜单命令“PLC”→“MPI Address…”，可以设置仿真 PLC 在指定的网络中的节点地址。用菜单命令“Save PLC”或“Save PLC As…”保存新地址。

4. LAY 文件和 PLC 文件

用 S7-PLCSIM 仿真时自动生成 LAY 文件和 PLC 文件，退出仿真软件时将会询问是否保存 LAY 文件或 PLC 文件。LAY 文件用于保存仿真时各视图对象的信息，例如各视图对象选择的数据格式等；PLC 文件用于保存上次仿真运行时设置的数据和动作等，包括程序、硬件组态、CPU 工作方式的选择、运行模式(单周期运行模式或连续运行模式)的选择、I/O 状态、定时器的值、符号地址、电源的通/断等。下一次仿真时，不需要重复上次的操作，可以直接调用这两个文件。

4.4.5 仿真 PLC 与实际 PLC 的区别

1. 仿真 PLC 特有的功能

仿真 PLC 有下述实际 PLC 没有的功能：

(1) 可以立即暂时停止执行用户程序,对程序状态不会有什么影响。

(2) 由 RUN 模式进入 STOP 模式不会改变输出的状态。

(3) 在视图对象中的变动立即使对应的存储区中的内容发生相应的改变。实际的 CPU 要等到扫描结束时才会修改存储区。

(4) 可以选择单次扫描或连续扫描。

(5) 可使定时器自动运行或手动运行,可以手动复位全部定时器或复位指定的定时器。

(6) 可以手动触发下列中断 OB:OB40～OB47 (硬件中断)、OB70 (I/O 冗余错误)、OB72 (CPU 冗余错误)、OB73 (通信冗余错误)、OB80 (时间错误)、OB82 (诊断中断)、OB83 (插入/拔出模块)、OB85 (程序顺序错误)与 OB86 (机架故障)。

(7) 对映像存储器与外设存储器的处理:如果在视图对象中改变了过程输入的值,S7-PLCSIM 立即将它复制到外设存储区。在下一次扫描开始外设输入值被写到过程映像寄存器时,希望的变化不会丢失。在改变过程输出值时,它被立即复制到外设输出存储区。

2. 仿真 PLC 与实际 PLC 的区别

(1) PLCSIM 不支持写到诊断缓冲区的错误报文,例如不能对电池失电和 EEPROM 故障仿真,但是可以对大多数 I/O 错误和程序错误仿真。

(2) 工作模式的改变(例如由 RUN 转换 STOP 模式)不会使 I/O 进入"安全"状态。

(3) 不支持功能模块和点对点通信。

(4) 支持有 4 个累加器的 S7-400 CPU,在某些情况下 S7-400 与只有两个累加器的 S7-300 的程序运行可能不同。

(5) S7-300 的大多数 CPU 的 I/O 是自动组态的,模块插入物理控制器后被 CPU 自动识别。仿真 PLC 没有这种自动识别功能。如果将自动识别 I/O 的 S7-300 CPU 的程序下载到仿真 PLC,系统数据没有包括 I/O 组态。因此在用 PLCSIM 仿真 S7-300 程序时,如果想定义 CPU 支持的模块,首先必须下载硬件组态。

4.5 程序的下载与上载

4.5.1 装载存储器与工作存储器

用户程序被编译后,逻辑块、数据块、符号表和注释(见图 4-16)保存在计算机的硬盘中。在完成组态、参数赋值、程序创建和建立在线连接后,可以将整个用户程序或个别的块下载到 PLC。系统数据(System Data)包括硬件组态、网络组态和连接表,也应下载到 CPU。

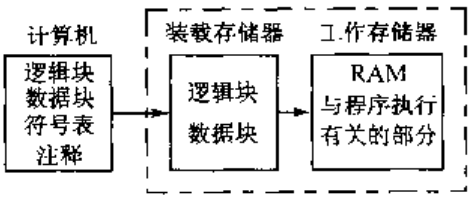


图 4-16 装载存储器与工作存储器

CPU 中的装载存储器用来存储没有符号表和注释的完整的用户程序,这些符号和注释保存在计算机的存储器中。为了保证快速地执行用户程序,CPU 只是将块中与程序执行有关的部分装入 RAM 组成的工作存储器。

在源程序中用 STL 生成的数据块可以标记为"与执行无关",其关键字为"UNLINKED",它们被下载到 CPU 时只是保存在装载存储器中。如果需要,可以用 SFC 20"BLKMOV"复制到工作存储器中,这样处理可以节省存储空间。

1. 装载存储器

装载存储器可以用存储器卡来扩展。在 S7-300 CPU 中,装载存储器可能是集成的 E-PROM 或集成的 RAM。

在 S7-400 中,用一个存储卡(RAM 或 EPROM)来扩展装载存储器。集成的装载存储器主要用来重新装载或修改块,新的 S7-400 附加的工作存储器也是插入式的。

装载存储器为 RAM 时,可以下载和删除单个的块、下载和删除整个用户程序、以及重新装入单个的块。

装载存储器如果是集成的(仅 S7-300)或外插的 EPROM 时,只能下载整个用户程序。

2. 工作存储器

工作存储器是集成的 RAM,用来存储程序处理需要的那一部分用户程序。复位 CPU 中的存储器时,存储在 RAM 中的程序丢失。虽然没有后备电池,保存在 EPROM 存储器卡中的程序不会因为复位 CPU 的存储器而被擦除。

现在的装载存储器卡使用的都是 Flash EPROM(快闪存储器,简称为 FEPROM),下载的用户程序保存在 FEPROM 中,断电时其中的信息也不会丢失,在硬件组态时可以定义断电保持区。取下或插入存储器卡时,CPU 要求存储器复位。插入 RAM 卡时,用户程序必须从编程器装入。插入 FEPROM 卡时,复位存储器后,用户程序从 FEPROM 卡拷入工作存储器。

上载时上传的是工作存储器中的内容。要保存修改后的程序块,应将它保存到硬盘上,或保存到 FEPROM 中。使用菜单命令"PLC"→"Download to EPROM Memory Card on CPU"可以直接下载到 CPU 的存储器卡中,存储器卡的内容必须先擦除。

在 PLC 中,没有电池后备的 RAM 在掉电时保存在它里面的数据将会丢失. 存储卡是便携式数据记录媒体,用编程设备来写入,块或用户程序被保存在 FEPROM 存储卡中,后者插在 CPU 的一个插槽里。电源关断和 CPU 复位时,存储器卡内的数据不会丢失。在 CPU 存储器复位且电源掉电之后,电源又重新恢复时,EPROM 中的内容被重新复制到 CPU 存储器的 RAM 区。

3. 系统存储器

系统存储器包含下列的存储器区域:过程映像输入/输出表(PII、PIQ),位存储器(M),定时器、计数器和局域堆栈(L)。

4.5.2 在线连接的建立与在线操作

打开 STEP 7 的 SIMATIC 管理器时,建立的是离线窗口,看到的是计算机硬盘上的项目信息。Block(块)文件夹中包含硬件组态时产生的系统数据和程序编辑器生成的块。

STEP 7 与 CPU 成功地建立起连接后,将会自动生成在线窗口,该窗口中显示的是通过通信得到的 CPU 中的项目结构。块文件夹中包含系统数据块、用户生成的块(OB、FB 和 FC)以及 CPU 中的系统块(SFB 和 SFC)。用菜单命令"View"→"Online"、"View"→"Offline"或相应的工具条中的按钮,可以切换在线窗口和离线窗口。用管理器的"Windows"菜单命令可以同时显示在线窗口和离线窗口。

1. 建立在线连接

下面的操作需要在编程设备和 PLC 之间建立在线连接:下载 S7 用户程序或块、从 PLC 上载程序到计算机;测试用户程序;比较在线和离线的块;显示和改变 CPU 的操作模式;为 CPU 设置时间和日期;显示模块信息和硬件诊断。

为了建立在线连接，计算机和 PLC 必须通过硬件接口（例如多点接口 MPI）连接，然后通过在线的项目窗口或“Accessible Nodes(可访问站)”窗口访问 PLC。

(1) 通过在线的项目窗口建立在线连接

如果在 STEP 7 的项目中有已经组态的 PLC，可以选择这种方法。

在 SIMATIC 管理器中执行菜单命令“View”→“Online”进入在线(Online)状态，执行菜单命令“View”→“Offline”进入离线(Offline)状态。也可以用管理器工具条中的“Online”和“Offline”图标来切换两种状态。在线状态意味着 STEP 7 与 CPU 成功地建立了连接。

使用菜单命令“View”→“Online”打开一个在线窗口，该窗口最上面的标题栏中的背景变为浅蓝色。在块工作区出现了 CPU 中大量的系统功能块 SFB、系统功能 SFC 和已下载到 CPU 的用户编写的块。SFB 和 SFC 在 CPU 的操作系统中，无需下载，也不能用编程软件删除。在线窗口显示的是 PLC 中的内容，而离线窗口显示的是计算机中的内容。

SIMATIC 管理器的“PLC”菜单中的某些功能只能在在线窗口中激活，不能在离线窗口中使用。

(2) 通过“Accessible Nodes”窗口建立在线连接

在 SIMATIC 管理器中用菜单命令“PLC”→“Display Accessible Nodes”，打开“Accessible Nodes”(可访问的站)窗口，用“Accessible Nodes”对象显示网络中所有可访问的可编程模块。如果编程设备中没有关于 PLC 的项目数据，可以选择这种方式。那些不能用 STEP 7 编程的站(例如编程设备或操作面板)也能显示出来。

如果 PLC 与 STEP 7 中的程序和组态数据是一致的，在线窗口显示的是 PLC 与 STEP 7 中的数据的组合。例如在在线项目中打开一个 S7 块，将显示来自 PLC 的 CPU 中的块的指令代码部分，以及来自编程设备数据库中的注释和符号。

如果没有通过项目结构，而是直接打开连接的 CPU 中的块，显示的程序没有符号和注释。因为在下载时没有下载符号和注释。

2. 访问 PLC 的口令保护

使用口令可以保护 CPU 中的用户程序和数据，未经授权不能改变它们(有写保护)，还可以用“读保护”来保护用户程序中的编程专利、对在线功能的保护可以防止可能对控制过程的人为的干扰。保护级别和口令可以在设置 CPU 属性的“Protection”选项卡中设置，需将它们下载到 CPU 模块。

设置了口令后，执行在线功能时，会显示出“Enter Password”对话框。若输入的口令正确，就可以访问该模块。此时可以与被保护的模块建立在线连接，并执行属于指定的保护级别的在线功能。

执行菜单命令“PLC”→“Access Rights”→ “Setup”，在出现的“Enter Password”对话框中输入口令，以后在线访问时，将不再询问。输入的口令将一直有效，至到 SIMATIC 管理器被关闭，或使用菜单命令“PLC” →“Access Rights” → “Cancel”取消口令。

3. 处理模式与测试模式

可以在设置 CPU 属性的对话框中的“Protection”(保护)选项卡选择处理(Process)模式或测试(Test)模式，这两种模式与 S7-400 和 CPU 318-2 无关。

在处理模式，为了保证不超过在“Protection”选项卡中设置的循环扫描时间的增量，像程序状态或监视/修改变量这样的测试功能是受到限制的。因此在处理模式中不能使用断点测

试和程序的单步执行功能。

在测试模式，所有的测试功能都可以不受限制地使用，即使这些功能可能会使循环扫描时间显著地增加。

4. 刷新窗口内容

用户操作（例如下载或删除块）对在线的项目窗口的修改不会在已打开的“Accessible Nodes”（可访问的站）窗口自动刷新。要刷新一个打开的窗口，必须使用菜单命令“View”→“Update View”（刷新显示）或用功能键＜F5＞将该窗口刷新。

5. 显示和改变 CPU 的运行模式

进入在线状态后，在项目管理器左边的树形结构中选择某一个站，然后执行菜单命令“PLC ”→“Diagnostics/Settings”→“Operating Mode”，打开的对话框显示当前和最近一次运行模式以及在 CPU 模块当前的模式选择开关的设置。对于那些无法显示其当前开关设置的模块，将显示文本“Undefined”。

可以用对话框中的起动按钮和停止按钮改变 CPU 的模式。只有当这些按钮是激活的（按钮上的字是黑色的），才能在当前运行模式使用。

6. 显示与设置时间和日期

显示与设置时间和日期的操作条件与显示和改变运行模式的相同，执行菜单命令“PLC”→“Diagnostics / Settings”→“Set Time of Day”，在打开的对话框中将显示 CPU 和编程设备/计算机（PG/PC）中当前的日期和时间。

可以在“Date”（日期）和“Time”（时间）栏中输入新的值，或者用默认选项接收 PC 的时间和日期。

如果 CPU 模块没有实时时钟，对话框的时间显示为“00:00:00”，日期显示为“00.00.00”。

7. 压缩用户存储器（RAM）

删除或重装块之后，用户存储器（装载存储器和工作存储器）内将出现块与块之间的“间隙”，减少了可用的存储区。用压缩功能可以将现有的块在用户存储器中无间隙地重新排列，同时产生一个连续的空的存储区间。

在 STOP 模式下压缩存储器才能去掉所有的间隙。在 RUN-P 模式时因为当前正在处理的块被打开而不能在存储器中移动。RUN 模式有写保护功能，不能执行压缩功能。

有两种压缩用户存储器的方法：

（1）向 PLC 下载程序时，如果没有足够的存储空间，将会出现一个对话框报告这个错误。可以点击对话框中的【Compress】按钮压缩存储器。

（2）进入在线状态后，打开“HW Config”（硬件组态）窗口，双击 CPU 模块，打开 CPU 模块的“模块信息”对话框，选择“Memory”选项卡，点击压缩存储器的【Compress】按钮。

4.5.3 下载与上载

1. 下载的准备工作

（1）计算机与 CPU 之间必须建立起连接，编程软件可以访问 PLC；

（2）要下载的程序已编译好；

（3）CPU 处在允许下载的工作模式下（STOP 或 RUN-P）。

在 RUN-P 模式一次只能下载一个块，这种改写程序的方式可能会出现块与块之间的时

间冲突或不一致性，运行时 CPU 会进入 STOP 模式，因此建议在 STOP 模式下载。

在保存块或下载块时，STEP 7 首先进行语法检查。错误种类、出错的原因和错误在程序中的位置都显示在对话框中，在下载或保存块之前应改正这些错误。如果没有发现语法错误，块将被编译成机器码并保存或下载。建议在下载块之前，一定要先保存块(将块存盘)。

下载前用编程电缆连接 PC(个人计算机)和 PLC，接通 PLC 的电源，将 CPU 模块上的模式选择开关扳到"STOP"位置，"STOP"LED 亮。

下载用户程序之前应将 CPU 中的用户存储器复位，以保证 CPU 内没有旧的程序。存储器复位完成以下的工作：删除所有的用户数据(不包括 MPI 参数分配)，进行硬件测试与初始化；如果有插入的 EPROM 存储器卡，存储器复位后 CPU 将 EPROM 卡中的用户程序和 MPI 地址拷贝到 RAM 存储区。如果没有插存储器卡，保持设置的 MPI 地址。复位时诊断缓冲区的内容保持不变。复位后块工作区只有 SDB、SFC 和 SFB。

将模式选择开关从 STOP 位置扳到 MRES 位置，STOP LED 慢速闪烁两次后松开模式开关，它自动回到 STOP 位置。再将模式开关扳到 STOP 位置，"STOP"LED 快速闪动时，CPU 已被复位。复位完成后将模式开关重新置于"STOP"位置。

也可以用 STEP 7 复位存储器：将模式开关置于 RUN-P 位置，执行菜单命令"PLC"→"Diagnostic/Settings"→"Operation Mode"，使 CPU 进入 STOP 模式，再执行菜单命令"PLC"→"Clear/Reset"，点击【OK】按钮确认存储器复位。

2. 下载的方法

(1)在离线模式和 SIMATIC 管理器窗口中下载

在块工作区选择块，可用＜Ctrl＞键和＜Shift＞键选择多个块，用菜单命令"PLC"→"Download"将被选择的块下载到 CPU。

也可以在管理器左边的目录窗口中选择 Blocks 对象(包括所有的块和系统数据)，用菜单命令"PLC"→"Download"下载它们。

(2)在离线模式和其他窗口下载

对块编程或组态硬件和网络时，可以在当时的应用程序的主窗口中，用菜单命令"PLC"→"Download"下载当前正在编辑的对象

(3)在线模式下载

用菜单命令"View"→"Online"或"PLC"→"Display Accessible Nodes"打开一个在线窗口查看 PLC，在"Windows"菜单中可以看到这时有一个在线的管理器，还有一个离线的管理器，可以用"Windows"菜单同时打开和显示这两个窗口。用鼠标按住离线窗口中的块(即 STEP 7 中的块)，将它"拖放"到在线窗口中去，就完成了下载任务。可以一次下载所有的块，也可以只下载部分块。应先下载子程序块，再下载高一级的块。如果顺序相反，将进入 STOP 模式。

下载完成后，将 CPU 的运行模式选择开关扳到 RUN-P 位置，绿色的"RUN"LED 亮，开始运行程序。

(4) 上载程序

可以用装载功能从 CPU 的 RAM 装载存储器中，把块的当前内容上载到计算机编程软件打开的项目中，该项目原来的内容将被覆盖。

(5) 在线编程

在调试程序时，可能需要修改已下载的块，可以在在线窗口中双击要修改的块的图标，然

后进行修改。编完的块会立即在 CPU 中起作用。

3. 删除 CPU 内的 S7 块

在 CPU 程序的测试阶段，可能需要删除 CPU 内单个的块。块被保存在 CPU 用户存储器的 EPROM 中或 RAM 中。RAM 中的块可以被直接删除，装载存储器或工作存储器被占据的空间将会空出来供重新使用。

CPU 存储器被复位后，集成 EPROM 中的块被复制到 RAM 区。RAM 中的备份可以直接删除，被删除的块在 EPROM 中被标记为无效。在下一次存储器复位或没有后备电池的 RAM 电源掉电时，“删除”的块从 EPROM 被复制到 RAM，又会起作用。

4.6 用变量表调试程序

4.6.1 系统调试的基本步骤

(1) 硬件调试

可以用变量表来测试硬件，通过观察 CPU 模块上的故障指示灯，或使用 4.8 节介绍的故障诊断工具来诊断故障。

(2) 下载用户程序

下载程序之前应将 CPU 的存储器复位，将 CPU 切换到 STOP 模式，下载用户程序时应同时下载硬件组态数据。

(3) 排除停机错误

起动时程序中的错误可能导致 CPU 停机，可以使用 4.8.2 中的“模块信息”工具诊断和排除编程错误。

(4) 调试用户程序

通过执行用户程序来检查系统的功能，如果用户程序是结构化程序，可以在组织块 OB1 中逐一调用各程序块，一步一步地调试程序。在调试时应记录对程序的修改。调试结束后，保存调试好的程序。

在调试时，最先调试启动组织块 OB100，然后调试 FB 和 FC。应先调试嵌套调用最深的块，例如首先调试图 4-17 中的 FB1。图中括号内的数字为调试的顺序，例如调试好 FB1 后调试调用 FB1 的 FC3 等等。调试时可以在完整的 OB1 的中间临时插入 BEU(块无条件结束)指令，只执行 BUE 指令之前的部分，调试好后将它删除掉。

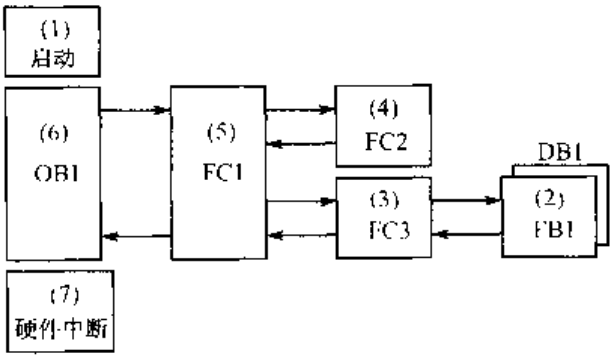


图 4-17 程序调试的顺序

最后调试不影响 OB1 的循环执行的中断处理程序，或者在调试 OB1 时调试它们。

4.6.2 变量表的基本功能

使用下一节将要介绍的程序状态功能，可以在梯形图、功能块图或语句表程序编辑器中形象

直观地监视程序的执行情况，找出程序设计中存在的问题。但是程序状态功能只能在屏幕上显示一小块程序，在调试较大的程序时，往往不能同时显示和调试某一部分程序所需的全部变量。

变量表可以有效地解决上述问题。使用变量表可以在一个画面中同时监视、修改和强制用户感兴趣的全部变量。一个项目可以生成多个变量表，以满足不同的调试要求。

在变量表中可以赋值或显示的变量包括输入、输出、位存储器、定时器、计数器、数据块内的存储器和外设 I/O。

1. 变量表的功能

(1) 监视(Monitor)变量：在编程设备或 PC(计算机)上显示用户程序中或 CPU 中每个变量的当前值。

(2) 修改(Modify)变量：将固定值赋给用户程序或 CPU 中的变量。

(3) 对外设输出赋值：允许在停机状态下将固定值赋给 CPU 中的每个输出点 Q。

(4) 强制变量：给用户程序或 CPU 中的某个变量赋予一个固定值，用户程序的执行不会影响被强制的变量的值。

(5) 定义变量被监视或赋予新值的触发点和触发条件。

2. 用变量表监视和修改变量的基本步骤

(1) 生成新的变量表或打开已存在的变量表，编辑和检查变量表的内容。

(2) 建立计算机与 CPU 之间的硬件连接，将用户程序下载到 PLC。在变量表窗口中用菜单命令“PLC”→“Connect to”建立当前变量表与 CPU 之间的在线连接。

(3) 用菜单命令“Variable”→“Trigger”选择合适的触发点和触发条件。

(4) 将 PLC 由 STOP 模式切换到 RUN-P 模式。

(5) 用菜单命令“Variable”→“Monitor”或“Variable”→“Modify”激活监视或修改功能。

4.6.3 变量表的生成

1. 生成变量表的几种方法

(1) 在 SIMATIC 管理器中用菜单命令“Insert”→“S7 Block”→“Variable Table”生成新的变量表。或者用鼠标右键点击 SIMATIC 管理器的块工作区，在弹出的菜单中选择“Insert New Object”→“Variable Table”命令来生成新的变量表。在出现的对话框中，可以给变量表取一个符号名，一个变量表最多有 1024 行。

(2) 在 SIMATIC 管理器中执行菜单命令“View”→“Online”，进入在线状态，选择块文件夹；或用“PLC”→“Display Accessible Nodes”命令，在可访问站(Accessible Nodes)窗口中选择块文件夹，用菜单命令“PLC”→“Monitor/Modify Variables”(监视/修改变量)生成一个无名的在线变量表。

(3) 在变量表编辑器中，用菜单命令“Table”→“New”生成一个新的变量表。可以用菜单命令“Table”→“Open”打开已存在的表，也可以在工具栏中用相应的图标来生成或打开变量表。

像其他文件一样，可以通过剪贴板复制、剪切和粘贴来复制和移动变量表，目标程序的符号表中已有的符号将被修改。在移动变量表时，源程序符号表中相应的符号也被移动到目标程序的符号表中。

如果需要监视的变量很多，可以为一个用户程序生成几个变量表。

2. 在变量表中输入变量

图 4-18 是调试某电动机控制系统时使用的变量表的一部分。

在输入变量时应将逻辑块中有关联的变量放在一起。

可以在“符号”(Symbol)栏输入在符号表中定义过的符号，在地址栏将会自动出现该符号的地址。也可以在“地址”(Address)栏输入地址，如果该地址已在符号表中定义了符号，将会在符号栏自动地出现它的符号。符号名中如果含有特殊的字符，必须用引号括起来，例如“Motor.off”和“Motor-off”等。

在变量表编辑器中使用菜单命令“Options”→“Symbol Table”，可以打开符号表，定义新的符号。可以从符号表中复制地址，将它粘贴到变量表。

可以在变量表的显示格式(Display format)栏直接输入格式，也可以执行菜单命令“View”→“Select Display Format”，或用右键点击该列，在弹出的格式菜单中选择需要的格式。图 4-18 的变量表中最后一行的 IW2 用二进制数(Binary，简写为 BIN)显示，可以同时显示和分别修改 I 2.0～I 3.7 这十六点数字量输入变量。这一方法用于 I，Q 和 M，可以用字节(8 位)、字(16 位)或双字(32 位)来监视和修改位变量。

图 4-18　变量表

在变量表中输入变量时，每行输入结束时都要执行语法检查，不正确的输入被标为红色。如果把光标放在红色的行上，可以从状态栏读到错误的原因。按<F1>键可以得到纠正错误的信息。变量表每行最多 255 个字符，不能用<Enter>键进入第二行。

通过“View”菜单最上面一组中的 9 条命令，可以打开或关闭变量表中对应的显示对象。

如果想使某个变量的“修改值”(Modify Value)列中的数据无效，可以使用菜单命令“Variable”→“Modify /Force Value as Comment”，在变量的修改值或强制值前将会自动加上注释符号“//”，表示它已经无效，变为注释了。在“Modify Value”列的修改值或强制值前用键盘加上注释符号“//”，其作用与菜单命令相同，再次执行该命令或用键盘删除“Modify Value”列的注

释符号，可以使修改值重新有效。

4.6.4 变量表的使用

1. 建立与 CPU 的连接

为了监视或修改在当前变量表(VAT)中输入的变量，必须与要监视的 CPU 建立连接。

可以在变量表中用菜单命令“PLC”→“Connect To”→“…”来建立与 CPU 的连接，以便进行变量监视或修改，也可以点击工具栏中相应的按钮。

菜单命令“PLC”→“Connect To”→“Configured CPU”用于建立被激活的变量表与 CPU 的在线连接。如果同时已经建立了与另外一个 CPU 的连接，这个连接被视为“Configured”(组态)的 CPU，直到变量表关闭。

菜单命令“PLC”→“Connect To”→“Direct CPU”用于建立被激活的变量表与直接连接的 CPU 之间的在线连接，例如“MPI = 2 (directly)”。直接连接的 CPU 是指与计算机用编程电缆连接的 CPU，在“Accessible Nodes”(可访问的站)窗口中被标记为“(directly)”。

菜单命令“PLC”→“Connect To”→“Accessible CPU”用于建立被激活的变量表与可以选择的 CPU 之间的在线连接。如果用户程序已经与一个 CPU 连接了，可以用这个命令来打开一个对话框，在对话框中选择另外一个想建立连接的 CPU。

使用菜单命令“PLC”→“Disconnect”，可以断开变量表和 CPU 的连接。

如果建立了在线连接，变量表窗口标题栏中将显示“ONLINE”(在线)。变量表下面的状态栏显示 PLC 的运行模式和连接状态。

2. 定义变量表的触发方式

用菜单命令“Variable”→“Trigger”打开图 4-19 所示的对话框，选择在程序处理过程中的某一特定点(触发点)来监视或修改变量，变量表显示的是被监视的变量在触发点的数值。触发点可以选择循环开始、循环结束和从 RUN 转换到 STOP。触发条件可以选择触发一次或在定义的触发点每个循环触发一次。如果设置为触发一次，点击一次图 4-18 中的监视变量或修改变量的按钮，执行一次相应的操作。

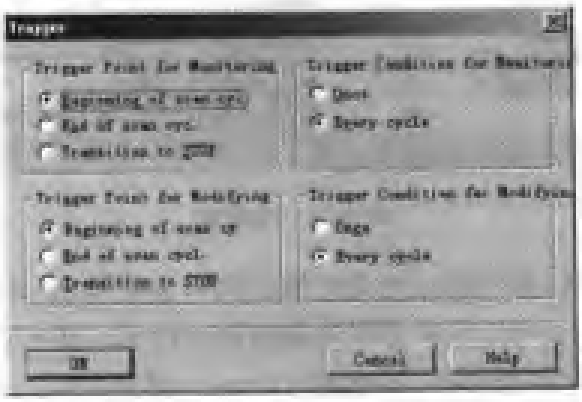


图 4-19 定义变量表的触发方式

3. 监视变量

将 CPU 的模式开关扳到 RUN-P 位置，执行菜单命令“Variable”→“Monitor”或点击标有眼镜的图标，起动监视功能。变量表中的状态值(Status Value)按设定的触发点和触发条件显

示在变量表中。如果触发条件设为“Every Cycle”(每一循环),用菜单命令“Variable”→“Monitor”可以关闭监视功能。

可以用菜单命令“Variable”→“Update Monitor Values”,对所选变量的数值作一次立即刷新,该功能主要用于停机模式下的监视和修改。

如果在监视功能被激活的状态下按<Esc>键,不经询问就会退出监视功能。

4. 修改变量

可以用下述方法修改变量表中的变量:

首先在要修改的变量的“Modify Value”栏输入变量新的值,显示格式为 BOOL 的数字量输入 0 或 1,输入后自动变为“false”或“true”。按工具栏中的激活修改值按钮或用菜单命令“Variable”→“Activate Modify Values”,将修改值立即送入 CPU。执行修改功能后不能用“Edit”→“Undo”命令取消。

在程序运行时如果修改变量值出错,可能导致人身或财产的损害。在执行修改功能前,要确认不会有危险情况出现。

如果在执行“Modifying”(修改)过程中按了<Esc>键,不经询问就会退出修改功能。

在 STOP 模式修改变量时,因为没有执行用户程序,各变量的状态是独立的,不会互相影响。I、Q、M 这些数字量都可以任意地设置为 1 状态或 0 状态,并且有保持功能,相当于对它们置位和复位。STOP 模式的这种变量修改功能常用来测试数字量输出点的硬件功能是否正常,例如将某个输出点置位后,观察相应的执行机构是否动作。

在 RUN 模式修改变量时,各变量同时又受到用户程序的控制。假设用户程序运行的结果使某数字量输出为 0,用变量表不可能将它修改为 1。在 RUN 模式不能改变数字量输入(I 映像区)的状态,因为它们的状态取决于外部输入电路的通/断状态。

修改定时器的值时,显示格式最好用 SIMATIC_TIME,在这种情况下以 ms 为单位输入定时值,但是个位被舍去,例如输入 123 时将显示 S5T#120ms。输入 12345 将显示 S5T#12s300ms,因为时间值只保留 3 位有效数字。输入 12.3 将显示 S5T#12s300ms。

只有在通电延时定时器的线圈“通电”时,将时间修改值写入定时器才会起作用,定时器将按写入的时间定时,定时期间其常开触点断开,修改后的定时时间到达时其常开触点闭合。定时器的线圈由断开变为接通时,重新使用程序设定的时间值定时。

计数器的当前值的修改与定时器类似,例如输入 123,将显示 C#123。输入值的上限为 C#999。

5. 强制变量

强制变量操作给用户程序中的变量赋一个固定的值,这个值不会因为用户程序的执行而改变。被强制的变量只能读取,不能用写访问来改变其强制值。这一功能只能用于某些 CPU。强制功能用于用户程序的调试,例如用来模拟输入信号的变化。

只有当“强制数值”(Force Values)窗口处于激活状态,才能选择用于强制的菜单命令。用菜单命令“Variable”→“Display Force Values”打开该窗口,被强制的变量和它们的强制值都显示在窗口中。当前在线连接的 CPU 或网络中的站的名称显示在标题栏中。状态条显示从 CPU 读出的强制操作的日期和时间。如果没有已经激活的强制操作,该窗口是空的。

在“强制数值”窗口中显示的黑体字表示该变量在 CPU 中已被赋予固定值;普通字体表示该变量正在被编辑;变为灰色的变量表示该变量在机架上不存在、未插入模块,或变量地址错

误，将显示错误信息。

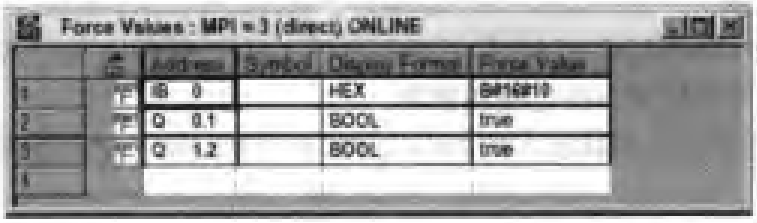


图 4-20 强制数值窗口

可以用菜单命令“Table”→“Save As”将“强制数值”窗口的内容存为一个变量表，或者选择菜单命令“Variable”→“Force”，将当前窗口的内容写到 CPU 中，作为一个新的强制操作。

变量的监视和修改只能在变量表中进行，不能在“强制数值”窗口进行。使用菜单命令“Insert”→“Variable Table”，可以在一个强制数值窗口中重新插入已存储的内容。

使用“强制”功能时，任何不正确的操作都可能会危及人员的生命或健康，或者造成设备或整个工厂的损失。强制作业只能用菜单命令“Variable”→“Stop Forcing”来删除或终止，关闭强制数值窗口或退出“监视和修改变量”应用程序并不能删除强制作业。强制功能不能用菜单命令“Edit”→“Undo”取消。

如果用菜单命令“Variable”→“Enable Peripheral Output”（使能外设输出）解除输出封锁，所有被强制的输出模块输出它们的强制值。

强制与修改变量的区别，见表 4-3。

表 4-3 强制与修改变量的区别

特点/功能	用 S7-400 强制	用 S7-300 强制	修 改
位存储器(M)	yes	–	yes
定时器与计数器(T,C)	–	–	yes
数据块(DB)	–	–	yes
外设输入(PIB,PIW,PID)	yes	–	–
外设输出(PQB,PQW,PQD)	yes	–	yes
输入与输出(I,Q)	yes	yes	yes
用户程序可以覆盖修改/强制值	–	yes	yes
无中断地有效替换强制数值	yes	yes	–
应用程序退出时变量仍保持其数值	yes	yes	–
与 CPU 的连接断开后变量仍保持其数值	yes	yes	–
允许的寻址错误：如 IW1 修改/强制值为 1 或 0	–	–	最后的有效
设置触发	立即触发	立即触发	一次或每一循环
功能只影响在激活的窗口的可视区变量	影响所有的强制值	影响所有的强制值	yes

4.7 用程序状态功能调试程序

4.7.1 程序状态功能的启动与显示

1. 启动程序状态

可以通过在程序编辑器中显示执行语句表、梯形图或功能块图程序时的状态（简称为程序

状态，Program Status)，来了解用户程序的执行情况，对程序进行调试。

进入程序状态之前，必须满足下列3个条件：

(1) 经过编译的程序下载到CPU；

(2) 打开逻辑块，用菜单命令"Debug"→"Monitor"进入在线监控状态；

(3) 将CPU切换到RUN或RUN-P模式。

如果在程序运行时测试程序出现功能错误或程序错误，将会对人员或财产造成严重损害，应确保不会出现这样的危险情况。

建议不要一下子调试整个程序，而是在OB1中一次调用一个块，单独地调试它们。

2. 语句表程序状态的显示

从光标选择的网络开始监视程序状态，程序状态的显示是循环刷新的。在图4-21所示的语句表编辑器中，右边窗口显示每条指令执行后的逻辑运算结果(RLO)和状态位STA(Status)、累加器1(STANDARD)、累加器2(ACCU 2)和状态字(STATUS…)，以及其他内容。用菜单命令"Options"→"Customize"打开的对话框中，用STL选项卡选择需要监视的内容，用LAD/FBD选项卡可以设置梯形图(LAD)和功能块图(SFB)程序状态的显示方式。

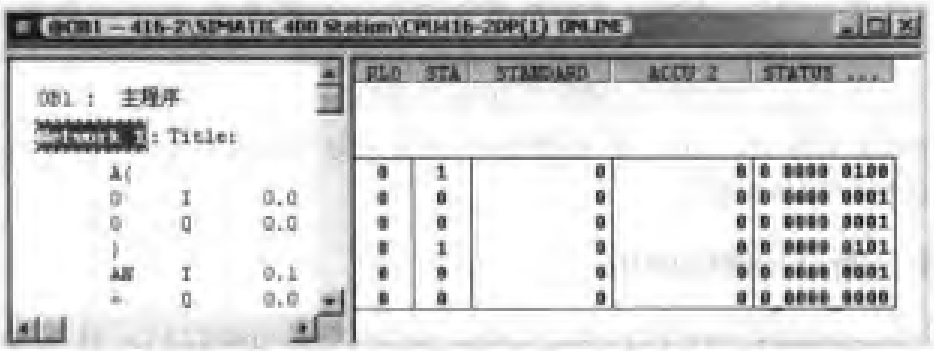


图4-21 用程序状态监视语句表程序

3. 梯形图程序状态的显示

LAD和FBD中用绿色连续线来表示状态满足，即有"能流"流过，见图4-22左边较粗较浅的线；用蓝色点状细线表示状态不满足，没有能流流过；用黑色连续线表示状态未知。

在梯形图或功能块图编辑器中执行菜单命令"Options"→"Customize"，在"LAD/FBD"选项卡中可以改变线型和颜色的设置。

进入程序状态之前，梯形图中的线和元件因为状态未知，全部为黑色。上述3个条件满足后，从梯形图左侧垂直的"电源"线开始的连线均为绿色(见图4-22)，表示有能流从"电源"线流出。有能流流过的处于闭合状态的触点、方框指令、线圈和"导线"均用绿色表示。

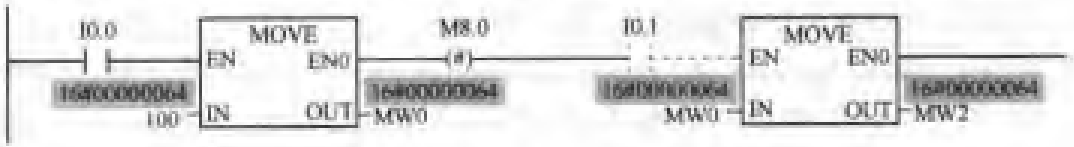


图4-22 梯形图程序状态的显示

如果有能流流入指令框的使能输入端EN，该指令被执行。如果指令框的使能输出ENO端接有后续元件，有能流从它的ENO端流到与它相连的元件，该指令框为绿色。如果ENO

端未接后续元件，则该指令框和 ENO 输出线均为黑色。

如果 CALL 指令成功地调用了逻辑块，CALL 线圈为绿色。

如果跳转条件满足，跳转被执行，跳转线圈为绿色。被跳过的网络中的指令没有被执行，这些网络中的梯形图为黑色。

NOT 触点左侧和右侧能流的状态刚好相反，即 NOT 触点左侧有能流时，其右侧没有能流；左侧没有能流时，其右侧有能流。

梯形图中加粗的字体显示的参数值是当前值，细体字显示的参数值来自以前的循环，即该程序区在当前扫描循环中未被处理。

4. 使用程序状态功能监视数据块

数据块必须使用数据显示方式(Data View)在线查看数据块的内容，在线数值在“Actual Value”(实际数值)列中显示. 程序状态被激活后，不能切换为声明显示方式(Declaration View)方式。

程序状态结束后，“Actual Value”列将显示程序状态之前的有效内容，不能将刷新的在线数值传送至离线数据块。

复合数据类型 DATE_AND_TIME 和 STRING 不能刷新，在复合数据类型 ARRAY、STRUCT、UDT、FB 和 SFB 中，只能刷新基本数据类型元素。程序状态被激活时，包含没有刷新的数据的“Actual Value”列中的区域将用灰色背景显示。

在背景数据块中的 IN_OUT 声明类型中，只显示复合数据类型的指针，不显示数据类型的元素，不刷新指针和参数类型。

4.7.2 单步与断点功能的使用

单步与断点是调试程序的有力工具，有单步与断点调试功能的 PLC 并不多见。允许设置的断点个数可以参考 CPU 的资料。

单步与断点功能在程序编辑器中设置与执行。单步模式不是连续执行指令，而是一次只执行一条指令。在用户程序中可以设置多个断点，进入 RUN 或 RUN-P 模式后将停留在第一个断点处，可以查看此时 CPU 内寄存器的状态。

“Debug”(调试)菜单中的命令用来设置、激活或删除断点。执行菜单命令“View”→“Breakpoint Bar”后，在工具条中将出现一组与断点有关的图标，可以用它们来执行与断点有关的命令。

1. 设置断点与进入单步模式的条件

(1) 只能在语句表中使用单步和断点功能，菜单命令“View”→“STL”将梯形图或功能块图转换为语句表。

(2) 设置断点前应在语句表编辑器中执行菜单命令“Options”→“Customize”，在对话框中选择 STL 标签页，激活“Activate new breakpoints immediately”(立即激活新断点)选项。

(3) CPU 必须工作在测试(Test)模式，可以用菜单命令“Debug”→“Operation”选择测试模式。

(4) 在 SIMATIC 管理器中进入在线模式，在线打开被调试的块，在调试过程中如果块被修改，需要重新下载它。

(5) 设置断点时不能启动程序状态(Monitor)功能。

(6) STL 程序中有断点的行、调用块的参数所在的行、空的行或注释行不能设置断点。

2. 设置断点与单步操作

满足上述条件时，在语句表中将光标放在要设置断点的指令所在的行。在 STOP 或 RUN-P 模式执行菜单命令“Debug”→“Set Breakpoint”，在选中的语句左边将出现一个紫色的小圆(见图 4-23)，表示断点设置成功，同时会出现一个显示 CPU 内寄存器的可移动小窗口。执行菜单命令“View”→“PLC Registers”可以打开或关闭该窗口。执行菜单命令“Options”→“Customize”，在 STL 选项卡中可以设置该窗口中需要显示哪些内容。

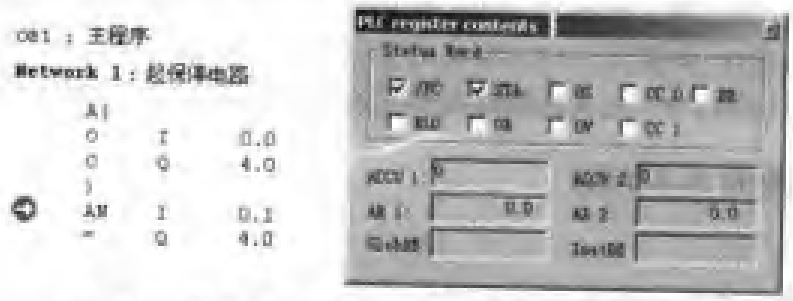


图 4-23 断点与断点处 CPU 寄存器和状态字的内容

如果在设置断点时起动了激活断点功能，即在菜单命令“Debug”→“Breakpoints Active”前有一个“√”(默认的状态)，表示断点的小圆是实心的。执行该菜单命令后“√”消失，表示断点的小圆变为空心的。要使断点起作用，应执行该命令，以激活断点。

将 CPU 切换到 RUN 或 RUN-P 模式，将在第一个表示断点的紫色圆球内出现一个向右的黄色的箭头(见图 4-23)，CPU 进入 HOLD(保持)模式，同时小窗口中出现断点处的状态字、累加器、地址寄存器和块寄存器的值。

执行菜单命令“Debug”→“Execute Next Statement”，或点击工具条上对应的按钮，断点处小圆内的黄色箭头移动到下一条语句，表示用单步功能执行下一条语句。如果下一条语句是调用块的语句，执行块调用后将跳到块调用语句的下一条语句。执行菜单命令“Debug”→“Execute Call”(执行调用)将进入调用的块，在调用的块中可以使用单步模式，也可以用该块内预先设置的断点来进行调试。块结束时将返回块调用语句的下一条语句。

为使程序继续运行至下一个断点，执行菜单命令“Debug”→“Resume”(继续)。

将光标放在断点所在的行，用菜单命令“Debug”→“Delete Breakpoint”可以删除该断点，菜单命令“Debug”→“Delete All Breakpoint”用于删除所有的断点。

执行菜单命令“Show Next Breakpoint”，光标将跳到下一个断点。

3. 保持模式

在执行程序时遇到断点，PLC 进入保持(HOLD)模式，“RUN”LED 闪烁，“STOP”LED 亮。这时不执行用户程序，停止处理所有的定时器，但是实时时钟继续运行。由于安全的原因，在 HOLD 模式下输出总是被禁止的。

在 HOLD 模式，可以通过图 4-23 中的信息窗口，查看 CPU 内的寄存器的状态。

在 HOLD 模式下，有后备电池的 PLC 在电源掉电后又重新恢复供电时，进入 STOP 模式，CPU 不执行自动再启动。在 STOP 模式下用户可以决定处理的方式，如设置/清除断点，执行手动再启动等。

没有后备电池的 PLC 没有记忆功能，所以电源恢复后不考虑断点以前的操作模式，而是

执行自动暖启动。

4.8 故障诊断

S7-300/400 有非常强大的故障诊断功能，通过 STEP 7 编程软件可以获得大量的硬件故障与编程错误的信息，使用户能迅速地查找到故障。

诊断是指 S7-300/400 内部集成的错误识别和记录功能，错误信息在 CPU 的诊断缓冲区内。有错误或事件(例如模式转换)发生时，标有日期和时间的信息被保存到诊断缓冲区，时间保存到系统的状态表中，如果用户已对有关的错误处理组织块编程，CPU 将调用该组织块。

诊断功能可以识别 CPU 或其他模块中的系统错误或 CPU 中的程序错误。

4.8.1 故障诊断的基本方法

1. 诊断符号

诊断符号用来形象直观地表示模块的运行模式和模块的故障状态(见图 4-24)。如果模块有诊断信息，在模块符号上将会增加一个诊断符号，或者模块符号的对比度降低。

图 4-24 诊断符号

在下列窗口可以观察到诊断符号：在线的管理器窗口、在线硬件诊断功能打开的快速视窗和在线的硬件组态窗口(诊断视窗)。

图 4-24 中的诊断符号"当前组态与实际组态不匹配"表示被组态的模块不存在，或者插入了与组态的模块的型号不同的模块。

诊断符号"模块故障"可能的原因：诊断中断，I/O 访问错误，或检测到故障 LED 亮。

诊断符号"无法诊断"表示无在线连接，或该模块不支持模块诊断信息，例如电源模块或子模块。

"强制"符号表示在该模块上有变量被强制，即在模块的用户程序中有变量被赋予一个固定值，该数据值不能被程序改变。强制符号可以与其他符号组合在一起显示，例如在图 4-24 中强制与运行符号的组合。

2. 故障诊断的基本方法

在管理器中用菜单命令"View"→"Online"打开在线窗口。打开所有的站，查看是否有 CPU 显示了指示错误或故障的诊断符号。可以用<F1>键打开解释诊断符号的帮助。

通过观察诊断符号，可以判断 CPU 模块的运行模式，是否有强制变量，CPU 模块和功能模块(FM)是否有故障(见图 4-24)。

打开在线窗口，在 SMATIC 管理器中执行菜单命令"PLC"→"Diagnostic/Setting"→"Hardware Diagnostics"，将打开硬件诊断快速浏览窗口。在该窗口中显示 PLC 的状态，看到带诊断功能的模块的硬件故障，双击故障模块可以获得详细的故障信息。

4.8.2 模块信息在故障诊断中的应用

1. 打开模块信息窗口

建立与 PLC 的在线连接后，在 SIMATIC 管理器中或在“Accessible Nodes”（可访问站）窗口中选择要检查的站，执行菜单命令“PLC”→“Diagnostics/ Settings”→“Module Information”，将打开模块信息窗口，显示该站中 CPU 模块的信息。

在快速视窗中使用【Module Information】按钮，或在硬件组态窗口双击 CPU 模块，也可以打开模块信息窗口。

在模块信息窗口的诊断缓冲区（Diagnostic Buffer）选项卡中，给出了 CPU 中发生的事件一览表，选中“Events”窗口中某一行的某一事件，下面灰色的“Details on”窗口将显示所选事件的详细信息。使用诊断缓冲区可以对系统的错误进行分析，查找停机原因，并对出现的诊断事件分类。

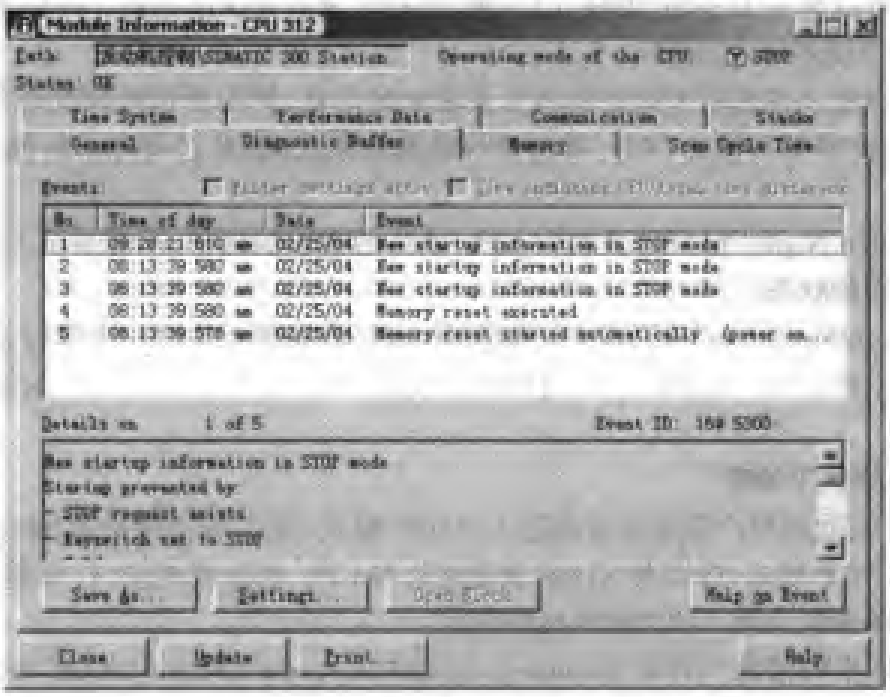


图 4-25 CPU 模块的在线模块信息窗口

诊断事件包括模块故障、过程写错误、CPU 中的系统错误、CPU 运行模式的切换、用户程序的错误和用户用系统功能 SFC 52 定义的诊断事件。

在模块信息窗口中，编号为 1，位于最上面的事件是最近发生的事件。如果显示因编程错误造成 CPU 进入 STOP 模式，选择该事件，并点击【Open Block】按钮，将在程序编辑器中打开与错误有关的块，显示出错的程序段。

诊断中断和 DP 从站诊断信息用于查找模块和 DP 从站中的故障的原因。

“Memory”（存储器）选项卡给出了所选的 CPU 或 M7 功能模块的工作存储器和装载存储器当前的使用情况，可以检查 CPU 或功能模块的装载存储器中是否有足够的空间用来存储新的块。

“Scan Cycle Time”（扫描循环时间）选项卡用于显示所选 CPU 或 M7 功能模块的最小循

环时间、最大循环时间和当前循环时间。

如果最长循环时间接近组态的最大扫描循环时间，由于循环时间的波动可能产生时间错误，此时应增大设置的用户程序最大循环时间（监控时间）。

如果循环时间小于设置的最小循环时间，CPU 自动延长循环至设置的最小循环时间。在这个延长时间内可以处理背景组织块（OB90）。组态硬件时可以设置最大和最小循环时间。

“Time System”（时间系统）选项卡显示当前日期、时间、运行的小时数以及时钟同步的信息。

“Performance Data”（性能数据）选项卡给出了所选模块（CPU/FM）可以使用的地址区和可以使用的 OB、SFB 和 SFC。

“Communication”（通信）选项卡给出了所选模块的传输速率、可以建立的连接个数和通信处理占扫描周期的百分比。

“Stacks”（堆栈）选项卡只能在 STOP 模式或 HOLD（保持）模式下调用，显示所选模块的 B（块）堆栈。还可以显示 I（中断）堆栈、L（局域）堆栈以及嵌套深度堆栈。可以跳转到使块中断的故障点，判明引起停机的原因。

在模块信息窗口各选项卡的上面显示了附加的信息，例如所选模块的在线路径、CPU 的操作模式和状态（例如出错或 OK）、所选模块的操作模式，如果它有自己的操作模式的话（例如 CP342-5）。

从“Accessible Nodes”窗口打开的非 CPU 模块的模块信息中，不能显示 CPU 本身的操作模式和所选模块的状态。

在“Module Information”对话框中切换选项卡时，数据都要从模块中读过来。但是显示某一页时其内容不再刷新。点击【Updete】（刷新）按钮，可以在不改变选项卡的情况下从模块读新的数据。

2. 在停机模式下诊断

如果 CPU 在处理用户程序时自动进入 STOP 模式，或下载程序后无法将 CPU 从 STOP 模式切换到 RUN-P 模式，可以在 STOP 状态建立与 CPU 的在线连接，打开模块信息对话框，根据诊断缓冲区中最上面一项判断停机的原因。

诊断缓冲区中的信息“STOP because programming error OB not loaded”表示 CPU 因为试图启动一个不存在的 OB 块去处理一个编程错误而停机，它前面一条信息指出了该编程错误。

选择描述编程错误的信息，点击【Open Block】（打开块）按钮，将在程序编辑器中打开包含程序错误的块，出错的程序段被加亮。

3. 停机模式下堆栈的内容

在模块信息对话框中选择“Stacks”（堆栈）选项卡，通过诊断缓冲区和堆栈中的内容，可以判定用户程序执行过程中引起故障的原因。

例如由于编程错误或停机指令使 CPU 进入停机状态时，可以用【I Stack】（中断堆栈），【L Stacks】（局域堆栈）和【Nesting Stack】（嵌套堆栈）按钮显示停机时这些堆栈中的内容。堆栈内容将提供哪个块中的哪条指令引起 CPU 进入停机的信息。B 堆栈（块堆栈）列出了所有停机前已经被调用但还未完全处理的块。

中断堆栈包含着中断时的数据或状态，例如累加器和寄存器的内容、打开的数据块及其大小、状态字的内容、优先级（嵌套层次）、中断的块和中断后程序将继续处理的块。

对于每个在 B 堆栈中列出的块，都可以通过选择该块并点击【L Stack】按钮显示相应的局域数据。它包含中断时用户程序正在处理的块的局域数据。

嵌套堆栈是逻辑操作"A(、AN(、O(、ON(、X(、XN ("使用的存储区域。如果中断时没有处理括号操作，该按钮为灰色(不能操作)。

4.8.3 用快速视窗和诊断视窗诊断故障

1. 用快速视窗诊断故障

在 SIMATIC 管理器中选择要检查的站，执行菜单命令"PLC"→"Diagnostics/Settings"→"Diagnose Hardware"，将打开 CPU 的硬件诊断快速视窗(Quick View)，显示该站中的故障模块。执行菜单命令"Option→Customize"，在打开的对话框的"View"选项卡中，应激活"诊断时显示快速视窗"。在快速视窗中显示如图 4-26 所示信息。

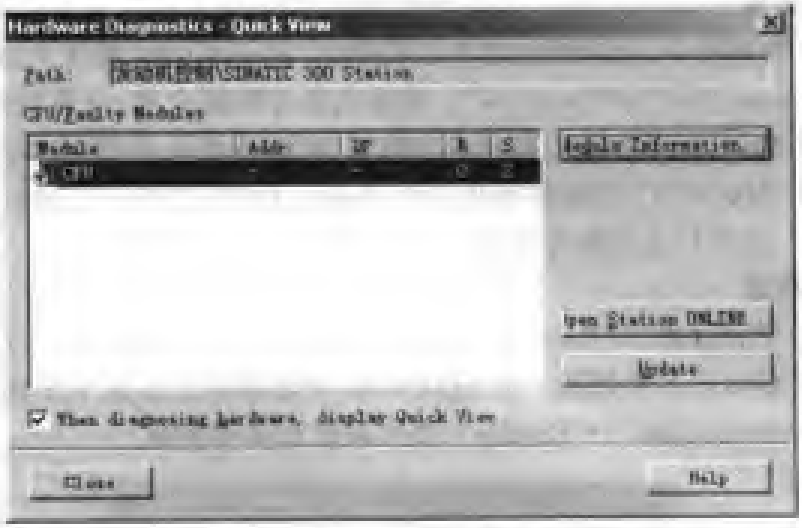


图 4-26 快速视窗

在线连接的 CPU 的数据和诊断符号，CPU 检查到故障的模块的诊断符号(例如诊断中断、I/O 访问错误)，模块类型和地址(Addr)，DP 主系统的站号(DP)，机架号(R)和槽号(S)。在快速视窗中选择故障模块后，点击【Module Information】(模块信息)按钮，可以获得该模块的信息。

2. 打开诊断视窗

诊断视窗实际上就是在线的硬件组态窗口。

(1) 在快速视窗中打开诊断视窗

在快速视窗中点击【Open Station Online】(在线打开站)按钮，将打开硬件组态的在线诊断视窗，它包含该站机架中所有的模块。

如果已经打开了离线的组态表，用菜单命令"Station"→"Open Online"也能打开诊断视窗。

(2) 在 SIMATIC 在线管理器中打开诊断视窗

在 SIMATIC 管理器中，使用菜单命令"View"→"Online"与 PLC 建立在线连接。双击打开一个站，然后打开其中的"Hardware"对象，也可以打开诊断视窗。

3. 诊断视窗的信息功能

与快速视窗相比，诊断视窗显示整个站在线的组态。包括机架组态和所有组态模块的诊

断符号，可以读取每个模块的状态以及 CPU 模块的操作模式、模块类型、序列号和地址，以及有关组态的注释。用这种方法可以得到那些没有故障因而没有在快速视窗中显示的模块信息。在诊断视窗中选择一个模块，用菜单命令“PLC”→“Module Information”可以查看其模块状态的详细信息。

4.9 显示参考数据

4.9.1 参考数据的生成与显示

STEP 7 可以为用户提供许多用于用户程序的调试和修改的参考数据，参考数据也可以打印存档，供最终用户使用。

1. 生成参考数据

在 SIMATIC 管理器中，选择要生成参考数据的“Blocks”（块）文件夹，然后执行菜单命令“Options”→“Reference Data”→“Generate”。在生成参考数据之前，计算机检查是否已有参考数据。如果有参考数据，但是不是当前的参考数据，可以选择是否刷新这些参考数据，或是否重新全部生成。在 SIMATIC 管理器的我中单击鼠标右键，在弹出的窗口中选择“Reference Data”→“Generate”，也可以生成参考数据。

2. 显示参考数据

使用菜单命令“Options”→“Reference Data”→“Display”，将出现图 4-27 所示的对话框，在对话框中选择需要显示的参考数据。在 SIMATIC 管理器的工作区中单击鼠标右键，在弹出的窗口中选择“Reference Data”→“Display”，也可以显示参考数据。

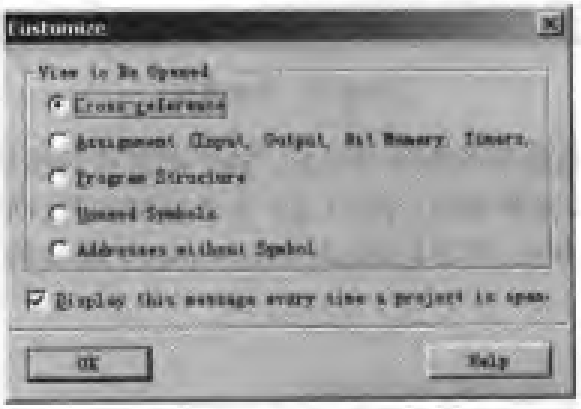


图 4-27 选择需要显示的参考数据

在图 4-27 中可以选择显示“Cross References”（交叉参考表）、“Assignment”（赋值表）、“Program Structure”（程序结构）、“Unused Symbols”（未用的符号）或“Addresses Without Symbols”（无符号的地址）。

打开某个参考数据显示窗口后，可以在“View”菜单内选择显示别的参考数据，也可以用工具条中对应的按钮打开别的参考数据显示窗口。

执行菜单命令“Windows”→“New Window”，将生成新的参考数据显示窗口。在“Windows”菜单中可以选择显示哪一个窗口。可以同时打开多个参考数据窗口，例如通过过滤器

(Filter)功能，在一个交叉参考表中只显示 PLC 的输入位，而在另一个交叉参考表中只显示输出位。

4.9.2 交叉参考表

1. 交叉参考表的内容

交叉参考表(见图 4-28)给出了 S7 用户程序中使用的地址的概况，包括输入(I)、输出(Q)、位存储(M)，定时器(T)、计数器(C)、功能块(FB)、功能(FC)、系统功能块(SFB)、系统功能(SFC)、I/O(P)和数据块(DB)，显示它们的绝对地址、符号地址以及使用的情况。

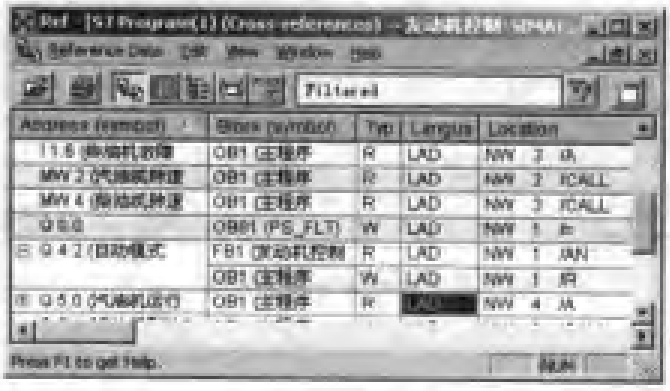


图 4-28 交叉参考表

窗口中的每一行对应着交叉参考表的一个输入项，使用菜单命令“Edit”→“Find”可以很容易地找到指定的地址或符号。

交叉参考表的每一行由以下参数组成：

“Address”：绝对地址；

“Symbol”：符号地址；

“Block”：使用该地址的块；

“Type”：对有关地址的访问类型，读(R)和/或写(W)；

“Language”：生成块的编程语言；

“Location”：变量的位置，例如“NW 3/A”表示在 Network 3 中的“A”(与)指令。可以用鼠标拖动表中最上面灰色的行中各列的分界点，来调节交叉参考表中各列的宽度。

2. 交叉参考表的参数设置

在交叉参考表中执行菜单命令“View”→“Filter”(过滤器)，将出现图 4-29 中的对话框。在“Cross References”(交叉参考表)选项卡中，可以选择下列的参数：

(1) 选择显示对象

在“Show objects”栏中，用鼠标选择要显示的地址区，打勾表示要显示该地址区。

例如选择 Input(输入)后，在“With Number”输入框内输入“*”号，表示显示范围为整个输入区，输入“10-20；24；30-50”，表示显示范围为 IB10－IB20；IB24 和 IB30－IB50。

选中“Display absolutely and symbolically”，将同时显示绝对地址和符号。

(2) 根据访问类型分类

在“Display by according to Access Type”区中，定义要显示的访问类型：

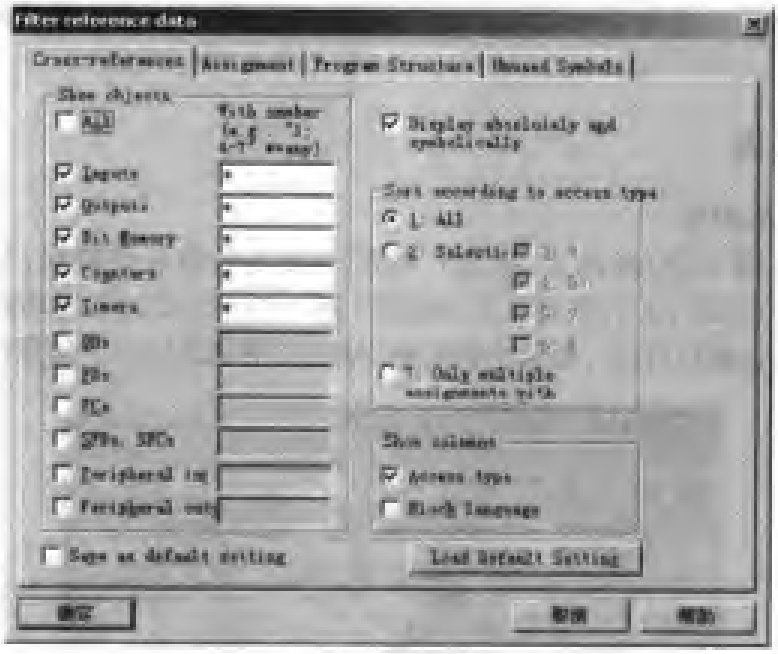


图 4-29 交叉参考表的参数设置

"All":显示所有的类型;

"Selection":有限制地显示访问类型,"W"为只写,"RW"为读写,"?"为编译时访问类型不能确定,"R"为只读;

"Only multiple assignments with operation "="":用于搜索用户程序中是否用"="指令对位地址多次赋值,即在梯形图中,同一地址的线圈是否多次出现。

(3) 选择显示的列

"Show columns"用于选择是否显示"Access type"(访问类型)和"Block language"(块使用的语言)。

(4) 默认设置的保存与装载

"Save as default":将当前的设置保存为默认的设置;

"Load Default Setting":用按钮装载默认的设置。

交叉参考表的默认选项是按存储区域分类,如果用鼠标选中某一列,可以对该列的输入项重新进行排列。在交叉参考表中执行菜单命令"View"→"Sort",可以选择按地址或块,递增(ascending)或递减(descending)的顺序排列表中各行的参考数据。

4.9.3 程序结构

1. 程序结构的显示内容

程序结构(Program Structure)显示用户程序中块的分层调用结构,通过它可以对程序所用的块、它们的从属关系以及它们对局域数据的需求有一个概括的了解(见图 4-30)。

"Block","Instance DB"列显示程序块和它的背景数据块(DB)。如果在过滤器中选中"Display absolutely and symbolically",在该栏将同时显示块的绝对地址和符号。

"Local Data (in the path)"列显示调用结构中需要的最大的局域数据字节数,包括每个OB需要的最大局域数据和每个路径需要的局域数据。

"Language"是调用的块的编程语言。

"Location"列显示与编程语言有关的调用块中调用点的位置。"NW"是 Network 的缩写。没有被调用的块显示在程序结构的底部,并且用黑叉标记。

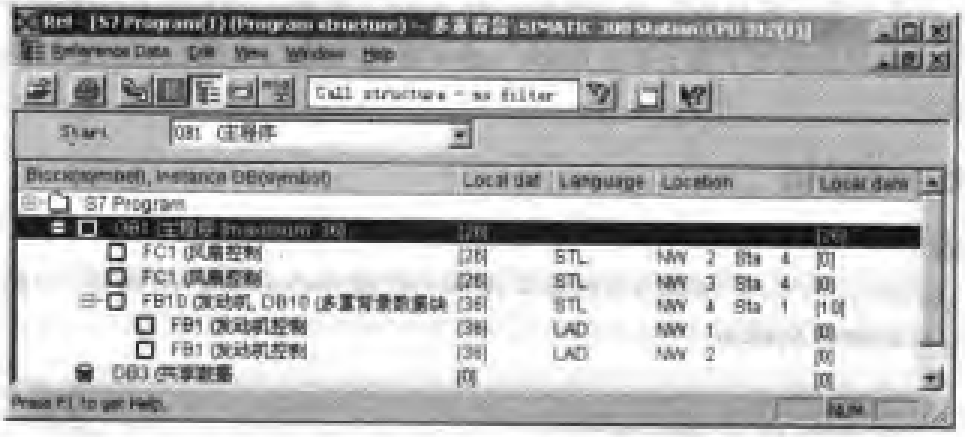


图 4-30 程序结构的显示

2. 显示程序结构的参数设置

在图 4-30 中用菜单命令"View"→"Filter"打开过滤器对话框,在"Program Structure"选项卡(见图 4-31)中,可以设置程序结构的显示方式。

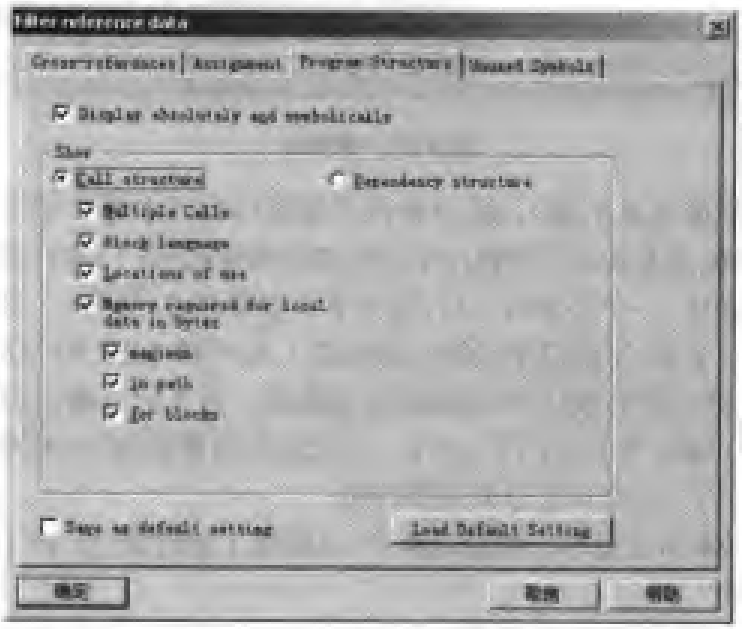


图 4-31 程序结构的参数设置

选中"Display absolutely and symbolically"将同时显示绝对地址和符号。

选择"Show"参数区中的选项"Dependency Structure",将显示块与块之间的从属关系。选择"Call structure(调用结构)"后,可以选择下列的显示内容:

"Multiple calls":显示多次调用。如果清除该选项,被多次调用的块将只显示一次调用;

"Block Language":显示块的编程语言;

"Locations of use":显示块被调用的位置;

"Memory Requirement for Local Data in Byte":显示下列选项对局域数据存储器的以字节为单位的需求:"Maximum":显示调用结构开始处需要的最大局域数据字节数。如果出现同步错误 OB 121 和 OB 122,则在要求的最大局域数据的字节数后面显示一个加号,加号后面是同步错误 OB 需要的额外的局域数据字节数;"In path":显示在调用结构中路径的结束处,程序路径需要的局域数据的字节数;"For block":显示在调用结构中路径结束处的块需要的局域数据的字节数。

4.9.4 其他参考数据

1. 赋值表

赋值表(Assignment List,见图 4-32)显示在用户程序中已经赋值的地址,它可以用于用户程序的故障检查和程序的修改。

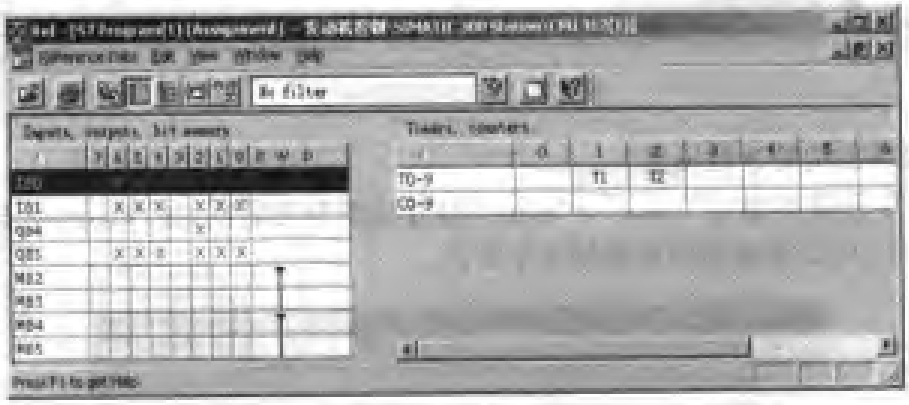


图 4-32 赋值表

左边的 I/Q/M 赋值表显示输入、输出和位存储器(M)区中哪个字节中的哪一位被使用了。定时器和计数器赋值表显示用户程序中已经使用的定时器(T)和计数器(C)。

在 I/Q/M 赋值表中,一个字节占一行,每个字节中有 8 位(第 0~7 位)。"B W D"列用来表示按什么存储单位(字节、字或双字)访问,例如图 4-32 中的 MB2 和 MB3 以字为单位访问。

空白的方格表示该位没有被访问,因此没有被赋值。打叉的方格表示该位被直接访问,例如从图 4-32 可以看出 QB4 中只使用了 Q4.2。有天蓝色的背景的行(例如图 4-32 中的 MB2~MB5)表示该地址以字节、字或双字为单位被访问。

从图 4-32 中可以看出在显示出的定时器中,只使用了定时器 T1 和 T2。

2. 未使用的符号

在参考数据窗口中执行"View"菜单中的命令"Unused Symbols",可以显示在符号表中已经定义,但是没有在用户程序中使用的符号。显示的内容有符号(Symbol)、地址(address)、数据类型(data type)和注释(comment)。

3. 没有在符号表中定义的地址

在参考数据窗口中执行"View"菜单中的命令"Addresses Without Symbols",可以显示已经在用户程序中使用、但是没有在符号表中定义的绝对地址。显示的内容有地址(address)、数据类型(data type)和该地址在用户程序中使用的次数(Number)。

4.9.5 在程序中快速查找地址的位置

如果没有生成参考数据或参考数据需要刷新，首先在 SIMATIC 管理器中，选择菜单命令“Options”→“Reference Data”→“Generate”，生成当前的参考数据。

1. “Go To Location”对话框的内容

在一个打开的程序块中选择一个地址(例如 Q1.0)，然后执行菜单命令“Edit”→“Go To”→“Location”，在出现的对话框(见图 4-33)中，显示出该地址在程序中出现的位置列表。

图 4-33 快速查找地址

最上面的“Address”输入框显示调用“Go To Location”对话框时指定的地址。为了显示别的地址被使用的位置，可以在该输入框中输入新的地址，或在下拉式列表中选择以前输入的地址，然后点击“Display”(显示)键，选择的地址出现的位置将以表格的形式出现，每一行对应一个该变量出现的位置。

对话框中的位置表由以下参数组成：

“Block”：使用该地址的块。

“Block Symbol”：块的符号名，如果该块有符号名的话。

“Details”：详细数据，与编程语言有关的信息，例如图 4-33 中的“NW 2 Stat 3 /=”表示 Q1.0 用于网络 2 语句(Statement)3 的“=”语句中。

“Type”：对该地址的访问类型，W 为只写，RW 为读写，“?”为编译时访问类型不能确定，R 为只读。

“Language”：块使用的编程语言。

选择“Type of Access”域中的“All”，显示该地址被访问的所有的位置。

如果需要对显示的位置进行筛选，只显示对该地址的某些访问类型的位置，可以选择“Selected”，然后选择上述的 4 种访问类型。

如果想显示与输入的地址重叠的物理地址或地址区，可以选择选项“Overlapping access to memory areas”(存储区域的重叠访问)。选择该选项后，在地址表的最左边将出现名为“Address”的附加的列。

参考数据只能在离线模式使用，因此该功能总是与离线块的交叉参考一起使用。即使在一个在线块中调用该功能也是这样。

2. 使用地址位置列表查找信号关系的例子

在分析梯形图时，需要分析各信号之间的关系，这种关系有时是极为复杂的，有关的语句分散在程序中的各个地方，地址位置列表为这种分析提供了很大的方便。

以下面的程序为例，网络 1 的第 1 条指令是 Q1.0 的常开触点指令，要查找 Q1.0 的线圈，将光标放在指令“A Q1.0”所在的行，然后执行菜单命令“Edit”→“Go To”→“Location >>”，或用鼠标右键执行命令“Go to”→“Location>>”，光标将转移到 Q1.0 的线圈指令“= Q1.0”所在的网络 2 的第 3 条指令处。

使用“Go To” →“Location Application <<”功能可以查找到某线圈的触点。以下面的程序为例，网络 2 的第 3 条指令是 Q1.0 的线圈指令，要查找 Q1.0 的触点指令所在的位置，将光标放在指令“= Q1.0”所在的行，然后执行菜单命令“Edit”→“Go To”→“Location <<”，或用鼠标右键执行命令“Go to”→“Location <<”，光标将转移到 Q1.0 的常开触点指令“A Q1.0”所在的网络 1 的第 1 条指令处。

下面是 OB1 中的语句表程序，其信号关系如图 4-34 所示。

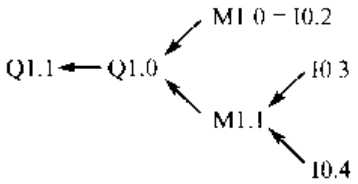


图 4-34 信号关系图

```
Network 1:
        A       Q 1.0
        =       Q 1.1
Network 2:
        A       M1.0
        A       M1.1
        =       Q 1.0
Network 3:
        A       I 0.2
        =       M1.0
Network 4:
        A       I 0.3
        A       I 0.4
        =       M1.1
```

从网络 1 中可以看出，Q1.1 的线圈受到 Q1.0 的控制，首先需要找到 Q1.0 的线圈所在的位置。使用地址位置表分析上述信号关系的步骤如下：

(1) 在 LAD/STL/FBD 编辑器中将光标放在 OB1 的网络 1 中的指令 A Q1.0 上。

(2) 选择菜单命令“Edit” →“Go To”→“Location”或用鼠标右键选择“Go to” →“Location”，在出现的对话框中显示 Q1.0 的所有赋值关系：

OB1	Cycle Execution	Nw 2	Inst 3	/=	W	STL
OB1	Cycle Execution	Nw 1	Inst 1	/A	R	STL

对话框中的第 1 行表示在循环执行的 OB1 的网络 2 中的第 3 条指令为“= Q1.0”，写访问，OB1 使用语句表(STL)编程语言。

(3) 在对话框中选中表中第一行的“=”指令，按【GO TO】按钮，将跳到 STL 编辑器中

OB1 的第 2 个网络中的第 3 条指令处(NW2 Inst3)。

(4) 由网络 2 可知 Q1.0 的线圈受 M1.0 和 M1.1 的控制,需要进一步检查 M1.0 和 M1.1 的线圈受谁的控制。首先将光标放在网络 2 第 1 条指令 A M1.0 上。

(5) 执行菜单命令"Edit"→"Go To"→"Location" 或用鼠标右键执行"Go to"→"Location",在出现的对话框中显示 M1.0 的所有赋值关系:

OB1	Cycle Execution	Nw 3	Inst 2	=	W	STL
OB1	Cycle Execution	Nw 2	Inst 1	A	R	STL

(6) 在对话框中选中表中第- -行的"="指令,按【GO TO】按钮,将跳到编辑器中 OB1 的第 3 个网络中的第 2 条指令处(NW3 Inst2)。由网络 3 可知,M1.0 的线圈受 I0.2 的常开触点的控制。用同样的方法,可以查找到 M1.1 受 I0.3 和 I0.4 的常开触点的控制。

第 5 章　数字量控制系统梯形图设计方法

5.1　梯形图的经验设计法与继电器电路转换法

数字量控制系统又称为开关量控制系统，继电器控制系统就是典型的数字量控制系统。

5.1.1　用经验法设计梯形图

1. 起动、保持与停止电路

起动、保持和停止电路简称为起保停电路，在梯形图中得到了广泛的应用。图 5-1 中起动按钮和停止按钮提供的起动信号 I0.0 和停止信号 I0.1 为 1 状态的时间很短。只按起动按钮，I0.0 的常开触点和 I0.1 的常闭触点均接通，Q4.1 的线圈“通电”，它的常开触点同时接通。放开起动按钮，I0.0 的常开触点断开，“能流”经 Q4.1 和 I0.1 的触点流过 Q4.1 的线圈，这就是所谓的“自锁”或“自保持”功能。只按停止按钮，I0.1 的常闭触点断开，使 Q4.1 的线圈“断电”，其常开触点断开，以后即使放开停止按钮，I0.1 的常闭触点恢复接通状态，Q4.1 的线圈仍然“断电”。这种功能也可以用图 5-2 中的 S(置位)和 R(复位)指令来实现。

在实际电路中，起动信号和停止信号可能由多个触点组成的串、并联电路提供。

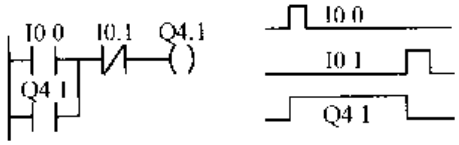


图 5-1　起保停电路

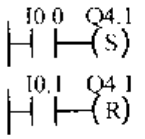


图 5-2　置位复位电路

可以用设计继电器电路图的方法来设计比较简单的数字量控制系统的梯形图，即在一些典型电路的基础上，根据被控对象对控制系统的具体要求，不断地修改和完善梯形图。有时需要反复多次地调试和修改梯形图，增加一些中间编程元件和触点，最后才能得到一个较为满意的结果。电工手册中常用的继电器电路图可以作为设计梯形图的参考电路。

这种方法没有普遍的规律可以遵循，具有很大的试探性和随意性，最后的结果不是惟一的，设计所用的时间、设计的质量与设计者的经验有很大的关系，所以有人把这种设计方法叫做经验设计法，它可以用于较简单的梯形图(例如手动程序)的设计。

2. 三相异步电动机的正反转控制

图 5-3 是三相异步电动机正反转控制的主电路和继电器控制电路图，KM1 和 KM2 分别是控制正转运行和反转运行的交流接触器。用 KM1 和 KM2 的主触点改变进入电动机的三相电源的相序，即可以改变电动机的旋转方向。图中的 FR 是热继电器，在电动机过载时，它的常闭触点断开，使 KM1 或 KM2 的线圈断电，电动机停转。

图 5-3 中的控制电路由两个起保停电路组成，为了节省触点，FR 和 SB1 的常闭触点供两个起保停电路公用。

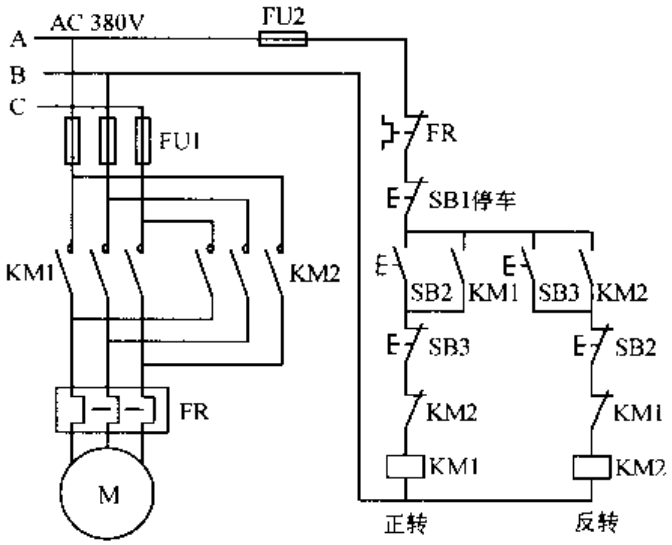


图 5-3　异步电动机正反转控制电路图

按下正转起动按钮 SB2，KM1 的线圈通电并自保持，电动机正转运行。按下反转起动按钮 SB3，KM2 的线圈通电并自保持，电动机反转运行。按下停止按钮 SB1，KM1 或 KM2 的线圈断电，电动机停止运行。

为了方便操作和保证 KM1 和 KM2 不会同时为 ON，在图 5-3 中设置了“按钮联锁”，即将正转起动按钮 SB2 的常闭触点与控制反转的 KM2 的线圈串联，将反转起动按钮 SB3 的常闭触点与控制正转的 KM1 的线圈串联。设 KM1 的线圈通电，电动机正转，这时如果想改为反转，可以不按停止按钮 SB1，直接按反转起动按钮 SB3，它的常闭触点断开，使 KM1 的线圈断电，同时 SB3 的常开触点接通，使 KM2 的线圈得电，电动机由正转变为反转。

由主回路可知，如果 KM1 和 KM2 的主触点同时闭合，将会造成三相电源相间短路的故障。在二次回路中，KM1 的线圈串联了 KM2 的辅助常闭触点，KM2 的线圈串联了 KM1 的辅助常闭触点，它们组成了硬件互锁电路。

假设 KM1 的线圈通电，其主触点闭合，电动机正转。因为 KM1 的辅助常闭触点与主触点是联动的，此时与 KM2 的线圈串联的 KM1 的常闭触点断开，因此按反转起动按钮 SB3 之后，要等到 KM1 的线圈断电，它的常闭触点闭合，KM2 的线圈才会通电，因此这种互锁电路可以有效地防止短路故障。

图 5-4 是实现上述功能的 PLC 的外部接线图和梯形图。在将继电器电路图转换为梯形图时，首先应确定 PLC 的输入信号和输出信号。三个按钮提供操作人员的指令信号，按钮信号必须输入到 PLC 中去，热继电器的常开触点提供了 PLC 的另一个输入信号。显然，两个交流接触器的线圈是 PLC 的输出负载。

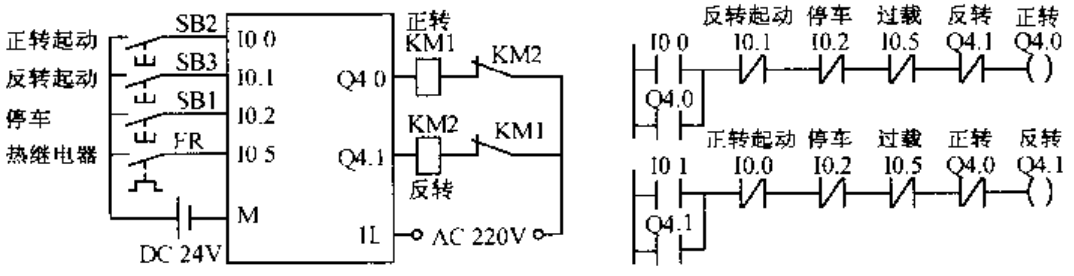


图 5-4　PLC 的外部接线图和梯形图

画出 PLC 的外部接线图后，同时也确定了外部输入/输出信号与 PLC 内的输入/输出过程映象位的地址之间的关系。可以将继电器电路图“翻译”为梯形图。如果在 STEP 7 中用梯形图语言输入程序，可以采用与图 5-3 中的继电器电路完全相同的结构来画梯形图。各触点的常开、常闭的性质不变，根据 PLC 外部接线图中给出的关系，来确定梯形图中各触点的地址。

CPU 在处理图 5-5a 中的梯形图时，实际上使用了局域数据位（例如 L20.0）来保存 A 点的运算结果，将它转换为语句表后，有 8 条语句。将图中的两个线圈的控制电路分离开后变为两个网络，一共只有 6 条指令。

```
//图 5-5a 中的程序
Network 1:
      A     I 1.0
      =     L20.0
      A     L20.0
      A     I 1.1
      =     Q4.3
      A     L20.0
      A     I 1.2
      =     Q4.4
```

```
//图 5-5b 中的程序
Network 1
      A     I 1.0
      A     I 1.1
      =     Q4.3
Network 2
      A     I 1.0
      A     I 1.2
      =     Q4.4
```

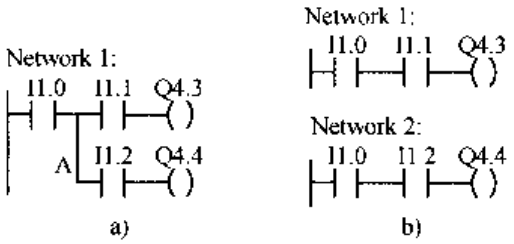


图 5-5　梯形图

如果将图 5-3 中的继电器电路图“原封不动”地转换为梯形图，也存在着同样的问题。图 5-4 中的梯形图将控制 Q4.0 和 Q4.1 的两个起保停电路分离开来，虽然多用了两个常闭触点，但是避免了使用与局域数据位有关的指令。此外，将各线圈的控制电路分离开后，电路的逻辑关系也比较清晰。

在图 5-4 中使用了 Q4.0 和 Q4.1 的常闭触点组成的软件互锁电路，它们只能保证输出模块中与 Q4.0 和 Q4.1 对应的硬件继电器的常开触点不会同时接通。如果从正转马上切换到反转，由于切换过程中电感的延时作用，可能会出现原来接通的接触器的主触点还未断弧，另一个接触器的主触点已经合上的现象，从而造成交流电源瞬间短路的故障。

此外，如果因主电路电流过大或接触器质量不好，某一接触器的主触点被断电时产生的电弧熔焊而被粘结，其线圈断电后主触点仍然是接通的，这时如果另一个接触器的线圈通电，仍将造成三相电源短路事故。为了防止出现这种情况，应在 PLC 外部设置由 KM1 和 KM2 的辅助常闭触点组成的硬件互锁电路（见图 5-4）。这种互锁与图 5-3 中的继电器电路的互锁原理相同，假设 KM1 的主触点被电弧熔焊，这时它的与 KM2 线圈串联的辅助常闭触点处于断开状态，因此 KM2 的线圈不可能得电。

3. 常闭触点输入信号的处理

前面在介绍梯形图的设计方法时，输入的数字量信号均由外部常开触点提供，但是在实际的系统中有些输入信号只能由常闭触点提供。

在图 5-4 中，如果将热继电器 FR 的常开触点换成常闭触点，没有过载时 FR 的常闭触点闭合，I0.5 为 1 状态，其常开触点闭合，常闭触点断开。为了保证没有过载时电动机的正常运行，显然应在 Q4.0 和 Q4.1 的线圈回路中串联 I0.5 的常开触点，而不是像继电器系统那样，串联 I0.5 的常闭触点。过载时 FR 的常闭触点断开，I0.5 为 0 状态，其常开触点断开，使 Q4.0 或 Q4.1 的线圈“断电”，起到了保护作用。

这种处理方法虽然能保证系统的正常运行，但是作过载保护的 I0.5 的触点类型与继电器电路中的刚好相反，熟悉继电器电路的人看起来很不习惯，在将继电器电路“转换”为梯形图时也很容易出错。

为了使梯形图和继电器电路图中触点的常开/常闭的类型相同，建议尽可能地用常开触点作 PLC 的输入信号。如果某些信号只能用常闭触点输入，可以按输入全部为常开触点来设计，然后将梯形图中相应的输入位的触点改为相反的触点，即常开触点改为常闭触点，常闭触点改为常开触点。

4. 小车控制程序的设计

图 5-6 中的小车开始时停在左边，左限位开关 SQ1 的常开触点闭合。要求按下列顺序控制小车：

(1) 按下右行起动按钮 SB2，小车右行。

(2) 走到右限位开关 SQ2 处停止运动，延时 8 s 后开始左行。

(3) 回到左限位开关 SQ1 处时停止运动。

在异步电动机正反转控制电路的基础上设计的满足上述要求的梯形图如图 5-7 所示。在控制右行的 Q4.0 的线圈回路中串联了 I0.4 的常闭触点，小车走到右限位开关 SQ2 处时，I0.4 的常闭触点断开，使 Q4.0 的线圈断电，小车停止右行。同时 I0.4 的常开触点闭合，T0 的线圈通电，开始定时。8 s 后定时时间到，T0 的常开触点闭合，使 Q4.1 的线圈通电并自保持，小车开始左行。离开限位开关 SQ2 后，I0.4 的常开触点断开，T0 的常开触点。因为其线圈断电而断开。小车运行到左边的起始点时，左限位开关 SQ1 的常开触点闭合，I0.3 的常闭触点断开，使 Q4.1 的线圈断电，小车停止运动。

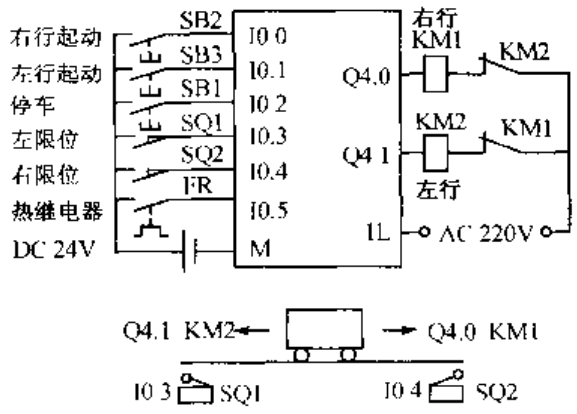


图 5-6 PLC 的外部接线图

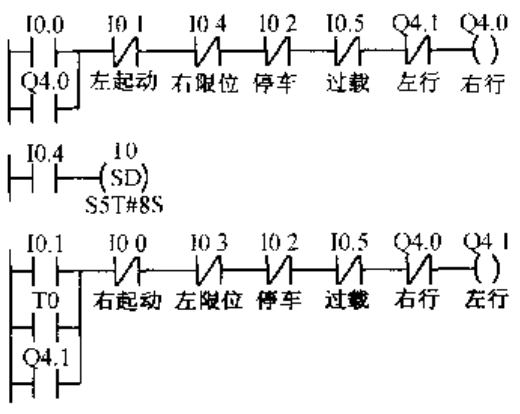


图 5-7 梯形图

在梯形图中(见图 5-7),保留了左行起动按钮 I0.1 和停止按钮 I0.2 的触点,使系统有手动操作的功能。串联在起保停电路中的左限位开关 I0.3 和右限位开关 I0.4 的常闭触点在手动时可以防止小车的运动超限。

5.1.2 根据继电器电路图设计梯形图

PLC 使用与继电器电路图极为相似的梯形图语言。如果用 PLC 改造继电器控制系统,根据继电器电路图来设计梯形图是一条捷径。这是因为原有的继电器控制系统经过长期使用和考验,已经被证明能完成系统要求的控制功能,而继电器电路图又与梯形图有很多相似之处,因此可以将继电器电路图"翻译"成梯形图,即用 PLC 的外部硬件接线图和梯形图软件来实现继电器系统的功能。这种设计方法一般不需要改动控制面板,保持了系统原有的外部特性,操作人员不用改变长期形成的操作习惯。

1. 基本方法

继电器电路图是一个纯粹的硬件电路图,将它改为 PLC 控制时,需要用 PLC 的外部接线图和梯形图来等效继电器电路图。可以将 PLC 想象成是一个控制箱,其外部接线图描述了这个控制箱的外部接线,梯形图是这个控制箱的内部"线路图",梯形图中的输入位(I)和输出位(Q)是这个控制箱与外部世界联系的"接口继电器",这样就可以用分析继电器电路图的方法来分析 PLC 控制系统。在分析梯形图时可以将输入位的触点想象成对应的外部输入器件的触点,将输出位的线圈想象成对应的外部负载的线圈。外部负载的线圈除了受梯形图的控制外,还可能受外部触点的控制。

将继电器电路图转换为功能相同的 PLC 的外部接线图和梯形图的步骤如下:

(1) 了解和熟悉被控设备的工作原理、工艺过程和机械的动作情况,根据继电器电路图分析和掌握控制系统的工作原理.

(2) 确定 PLC 的输入信号和输出负载。继电器电路图中的交流接触器和电磁阀等执行机构如果用 PLC 的输出位来控制,它们的线圈接在 PLC 的输出端。按钮、操作开关和行程开关、接近开关、压力继电器等提供 PLC 的数字量输入信号。继电器电路图中的中间继电器和时间继电器(例如图 5-8 中的 KA1 和 KT)的功能用 PLC 内部的存储器位和定时器来完成,它们与 PLC 的输入位、输出位无关。

(3) 选择 PLC 的型号,根据系统所需的功能和规模选择 CPU 模块、电源模块和数字量输入/输出模块,对硬件进行组态,确定输入/输出模块在机架中的安装位置和它们的起始地址。S7-300 的输入/输出模块的起始地址由模块所在的槽号确定,S7－400 的输入/输出模块的起始地址在组态时由系统自动指定,用户也可以进行修改。

(4) 确定 PLC 各数字量输入信号与输出负载对应的输入位和输出位的地址,画出 PLC 的外部接线图。各输入量和输出量在梯形图中的地址取决于它们所在的模块的起始地址和模块中的接线端子号。

(5) 确定与继电器电路图的中间继电器、时间继电器对应的梯形图中的存储器位(M)和定时器、计数器的地址。第(4)步和第(5)步建立了继电器电路图中的元件和梯形图中的位地址之间的对应关系.

(6) 根据上述的对应关系画出梯形图。

2. 根据继电器电路图设计梯形图的实例

液压动力滑台是机床加工工件时完成进给运动的动力部件，由液压系统驱动，完成加工的自动循环。滑台开始停在最左边，左限位开关 SQ1(I0.3)的常开触点闭合(见图 5-10 上面的运动示意图)。

在自动运行模式，开关 SA 闭合，I0.5 为 1 状态。按下起动按钮 SB1(I0.0)，Q4.0、Q4.2 变为 1 状态，YV11 和 YV2 的线圈通电，动力滑台向右快速进给(简称快进)；碰到中限位开关 SQ2(I0.1)时变为工作进给(简称工进)，Q4.2 变为 0 状态，YV2 的线圈断电；碰到右限位开关 SQ3(I0.2)时暂停 8 s，Q4.0 变为 0 状态，YV11 的线圈断电，滑台停止运动；时间到时动力滑台快速退回(简称快退)，Q4.1 变为 1 状态，YV12 的线圈通电；返回初始位置时限位开关 SQ1 动作，Q4.1 变为 0 状态，YV12 的线圈断电，停止运动。

图 5-8 是实现上述控制要求的继电器电路图，图 5-9 和图 5-10 是实现相同功能的 PLC 控制系统的外部接线图和梯形图。

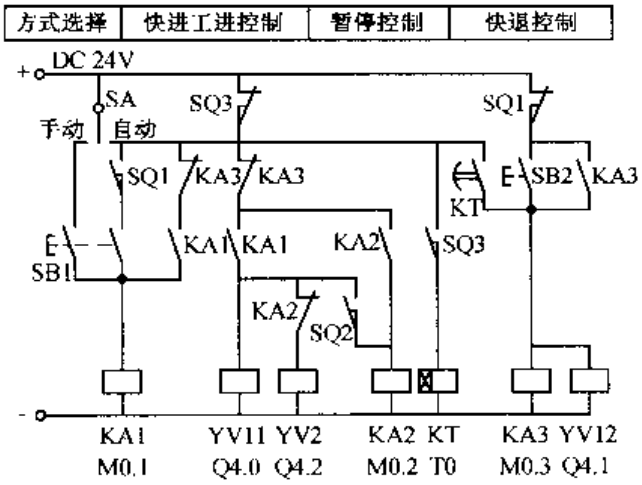


图 5-8 继电器电路图

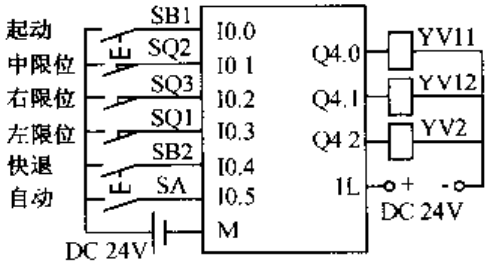


图 5-9 PLC 外部接线图

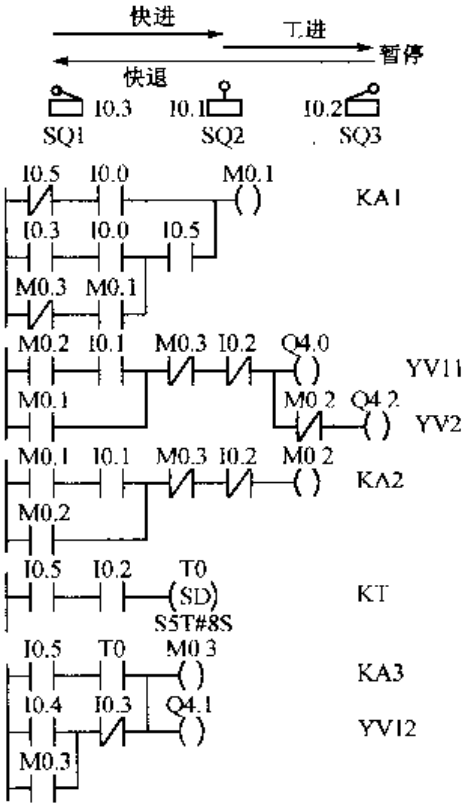


图 5-10 梯形图

图 5 10 中 M0.3 和 Q4.1 的状态完全相同，因此可以省略 M0.3，用 Q4.1 的触点来代替 M0.3 的触点。不过 M0.3 是用软件实现的，使用它不会增加硬件成本。

在设计时应注意梯形图与继电器电路图的区别，梯形图是一种软件，是 PLC 图形化的程序。在继电器电路图中，各继电器可以同时动作，而 PLC 的 CPU 是串行工作的，即 CPU 同时只能处理 1 条指令。

3. 注意事项

根据继电器电路图设计 PLC 的外部接线图和梯形图时应注意以下问题：

(1) 应遵守梯形图语言中的语法规定

由于工作原理不同，梯形图不能照搬继电器电路中的某些处理方法。例如在继电器电路中，触点可以放在线圈的两侧，但是在梯形图中，线圈必须放在电路的最右边。

(2) 适当分离继电器电路图中的某些电路

设计继电器电路原理图时的一个基本原则是尽量减少图中使用的触点的个数，因为这意味着成本的节约，但是这往往会使某些线圈的控制电路交织在一起。在设计梯形图时，首要的问题是设计的思路要清楚，设计出的梯形图容易阅读和理解，并不特别在意是否多用几个触点，因为这不会增加硬件的成本，只是在输入程序时需要多花一点时间。

在将图 5-8 中的继电器电路图改画为梯形图时，如果完全“原封不动”地改画，这种梯形图读起来也很费力，将它转换为语句表时，将会使用较多的局域数据变量(L)。在将继电器电路图改画为梯形图时，最好将各线圈的控制电路分开。仔细观察继电器电路图中每个线圈受哪些触点的控制，画出分离后各线圈的控制电路。

(3) 尽量减少 PLC 的输入信号和输出信号

PLC 的价格与 I/O 点数有关，因此减少输入信号和输出信号的点数是降低硬件费用的主要措施。

在 PLC 的外部输入电路中，各输入端可以接常开触点或常闭触点，也可以接触点组成的串并联电路。PLC 不能识别外部电路的结构和触点类型，只能识别外部电路的通断。

与继电器电路不同，一般只需要同一输入器件的一个常开触点给 PLC 提供输入信号，在梯形图中，可以多次使用同一输入位的常开触点和常闭触点。

图 5-8 中的选择开关 SA 的常开触点和常闭触点分别表示自动模式和手动模式。PLC 的输入端只需外接 SA 的一个触点(例如常开触点)，在梯形图中，I0.5 的常开触点和常闭触点分别对应于自动模式和手动模式。

如果在继电器电路图中，某些触点总是以相同的串并联电路的形式出现，可以将这种串并联电路作为一个整体接在 PLC 的一个输入点上。串并联电路接通时对应的输入位(I)为 1，梯形图中该输入位的常开触点闭合，常闭触点断开。

设计输入电路时，应尽量采用常开触点，如果只能使用常闭触点，梯形图中对应触点的常开/常闭类型应与继电器电路图中的相反。

某些器件的触点如果在继电器电路图中只出现一次，并且与 PLC 输出端的负载串联，例如有锁存功能的热继电器的常闭触点，不必将它们作为 PLC 的输入信号，可以将它们放在 PLC 外部的输出回路，仍与相应的外部负载串联(见图 1-5)。

继电器控制系统中某些相对独立且比较简单的部分，可以用继电器电路控制，这样可以减少所需的 PLC 的输入点和输出点。

(4) 时间继电器的处理

时间继电器有 4 种延时触点，图 5-11 是它们的图形符号和动作时序。图中上面两个触点是通电延时时间继电器的触点，线圈通电后触点延时动作，线圈断电后触点立即动作，恢复常态。这种时间继电器对应于 S7-300/400 的接通延时定时器。PLC 的定时器触点是延时动作的触点，虽然它们的形状与普通的触点形状一样。

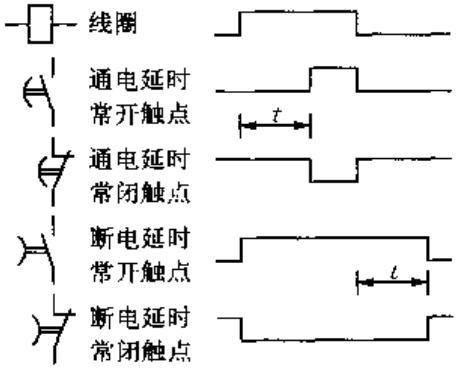


图 5-11 时间继电器

图 5-11 中下面两个触点是断电延时时间继电器的触点，线圈通电后触点立即动作，线圈断电后触点延时恢复常态。这种时间继电器对应于 S7-300/400 的断开延时定时器。

时间继电器除了有延时动作的触点外，还有在线圈通电瞬间接通的瞬动触点。在梯形图中，可以在定时器的线圈两端并联存储器位(M)的线圈，它的触点相当于定时器的瞬动触点。

(5) 设置中间单元

在梯形图中，若多个线圈都受某一触点串并联电路的控制，为了简化电路，在梯形图中可以设置中间单元，即用该电路来控制某存储器位(M)，在各线圈的控制电路中使用其常开触点。这种中间元件类似于继电器电路中的中间继电器。

(6) 设立外部互锁电路

控制异步电动机正反转的交流接触器如果同时动作，将会造成三相电源短路。为了防止出现这样的事故，应在 PLC 外部设置硬件互锁电路(见图 5-4)。

在继电器电路中采取了互锁措施的其他电路，例如异步电动机的星形-三角形起动电路等，在改用 PLC 控制时也应采取同样的硬件互锁措施。

(7) 外部负载的额定电压

PLC 的继电器输出模块和双向晶闸管输出模块一般只能驱动额定电压 AC 220 V 的负载，如果系统原来的交流接触器的线圈电压为 380 V 的，应将线圈换成 220 V 的，或设置外部中间继电器。

5.2 顺序控制设计法与顺序功能图

5.2.1 顺序控制设计法

用经验设计法设计梯形图时，没有一套固定的方法和步骤可以遵循，具有很大的试探性和随意性，对于不同的控制系统，没有一种通用的容易掌握的设计方法。在设计复杂系统的梯形图时，用大量的中间单元来完成记忆、联锁和互锁等功能，由于需要考虑的因素很多，它们往往又交织在一起，分析起来非常困难，一般不可能把所有的问题都考虑得很周到，程序设计出来后，需要模拟调试或在现场调试，发现问题后再针对问题对程序进行修改。即使是非常有经验的工程师，也很难做到设计出的程序能一次成功。修改某一局部电路时，很可能会引发出别的问题，对系统的其他部分产生意想不到的影响，因此梯形图的修改也很麻烦，往往花了很长的时间还得不到一个满意的结果。用经验法设计出的梯形图很难阅读，给系统的维修和改进带来了很大的困难。

所谓顺序控制，就是按照生产工艺预先规定的顺序，在各个输入信号的作用下，根据内部状态和时间的顺序，在生产过程中各个执行机构自动地有秩序地进行操作。使用顺序控制设计法时首先根据系统的工艺过程，画出顺序功能图(Sequential function chart)，然后根据顺序功能图画出梯形图。STEP 7 的 S7 Graph 就是一种顺序功能图语言，在 S7 Graph 中生成顺序功能图后便完成了编程工作。

顺序控制设计法是一种先进的设计方法，很容易被初学者接受，对于有经验的工程师，也会提高设计的效率，节约大量的设计时间。程序的调试、修改和阅读也很方便。只要正确地画出了描述系统工作过程的顺序功能图，一般都可以做到调试程序时一次成功。

顺序控制设计法最基本的思想是将系统的一个工作周期划分为若干个顺序相连的阶段，这些阶段称为步(Step)，然后用编程元件(例如存储器位 M)来代表各步。步是根据输出量的

ON/OFF 状态的变化来划分的，在任何一步之内，各输出量的状态不变，但是相邻两步输出量总的状态是不同的，步的这种划分方法使代表各步的编程元件的状态与各输出量的状态之间有着极为简单的逻辑关系。

使系统由当前步进入下一步的信号称为转换条件，转换条件可以是外部的输入信号，例如按钮、指令开关、限位开关的接通/断开等；也可以是 PLC 内部产生的信号，例如定时器、计数器的触点提供的信号，转换条件还可能是若干个信号的与、或、非逻辑组合。

顺序控制设计法用转换条件控制代表各步的编程元件，让它们的状态按一定的顺序变化，然后用代表各步的编程元件去控制 PLC 的各输出位。

顺序功能图是描述控制系统的控制过程、功能和特性的一种图形，也是设计 PLC 的顺序控制程序的有力工具。

顺序功能图并不涉及所描述的控制功能的具体技术，它是一种通用的直观的技术语言，可以供进一步设计和不同专业的人员之间进行技术交流之用。对于熟悉设备和生产流程的现场情况的电气工程师来说，顺序功能图是很容易画出的。

在 IEC 的 PLC 标准（IEC 61131）中，顺序功能图是 PLC 位居首位的编程语言。我国在 1986 年颁布了顺序功能图的国家标准 GB 6988.6－1986。顺序功能图主要由步、有向连线、转换、转换条件和动作（或命令）组成。

5.2.2 步与动作

1. 步

图 5-12 是液压动力滑台的进给运动示意图和输入输出信号的时序图，为了节省篇幅，将几个脉冲输入信号的波形画在一个波形图中。设动力滑台在初始位置时停在左边，限位开关 I0.3 为 1 状态，Q4.0～Q4.2 是控制动力滑台运动的 3 个电磁阀。与图 5-10 中的系统相同，按下起动按钮后，动力滑台的一个工作周期由快进、工进、暂停和快退组成，返回初始位置后停止运动。根据 Q4.0～Q4.2 的 ON/OFF 状态的变化，一个工作周期可以分为快进、工进、暂停和快退这 4 步，另外还应设置等待起动的初始步，图中分别用 M0.0～M0.4 来代表这 5 步。图 5-12 的右边是描述该系统的顺序功能图，图中用矩形方框表示步，方框中可以用数字表示各步的编号，也可以用代表各步的存储器位的地址作为步的编号，例如 M0.0 等，这样在根据顺序功能图设计梯形图时较为方便。

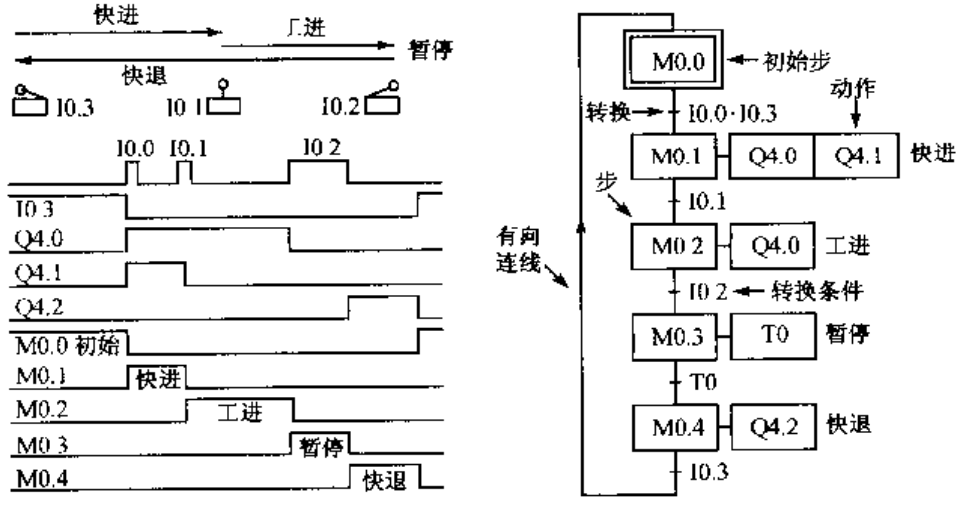


图 5-12　液压动力滑台的顺序功能图

2. 初始步

初始状态一般是系统等待启动命令的相对静止的状态。系统在开始进行自动控制之前，首先应进入规定的初始状态。与系统的初始状态相对应的步称为初始步，初始步用双线方框来表示，每一个顺序功能图至少应该有一个初始步。

3. 与步对应的动作或命令

可以将一个控制系统划分为被控系统和施控系统，例如在数控车床系统中，数控装置是施控系统，而车床是被控系统。对于被控系统，在某一步中要完成某些“动作”(action)；对于施控系统，在某一步中则要向被控系统发出某些“命令”(command)。为了叙述方便，下面将命令或动作统称为动作，并用矩形框中的文字或符号来表示动作，该矩形框与相应的步的方框用水平短线相连。

如果某一步有几个动作，可以用图 5-13 中的两种画法来表示，但是并不隐含这些动作之间的任何顺序。

图 5-13　动作

当系统正处于某一步所在的阶段时，该步处于活动状态，称该步为“活动步”。步处于活动状态时，相应的动作被执行；处于不活动状态时，相应的非存储型动作被停止执行。

说明命令的语句应清楚地表明该命令是存储型的还是非存储型的。非存储型动作“打开 1 号阀”，是指该步为活动步时打开 1 号阀，为不活动时关闭 1 号阀。非存储型动作与它所在的步是“同生共死”的，例如图 5-12 中的 M0.4 与 Q4.2 的波形完全相同，它们同时由 0 状态变为 1 状态，又同时由 1 状态变为 0 状态。

某步的存储型命令“打开 1 号阀并保持”，是指该步为活动步时 1 号阀被打开，该步变为不活动步时继续打开，直到在某一步 1 号阀被复位。在表示动作的方框中，可以用 S 和 R 来分别表示对存储型动作的置位(例如打开阀门并保持)和复位(例如关闭阀门)。

在图 5-12 的暂停步中，PLC 所有的输出量均为 0 状态。接通延时定时器 T0 用来给暂停步定时，在暂停步，T0 的线圈应一直通电，转换到下一步后，T0 的线圈断电。从这个意义上来说，T0 的线圈相当于暂停步的一个非存储型的动作，因此可以将这种为某一步定时的接通延时定时器放在与该步相连的动作框内，它表示定时器的线圈在该步内“通电”。

除了以上的基本结构之外，使用动作的修饰词可以在一步中完成不同的动作。修饰词允许在不增加逻辑的情况下控制动作。例如，可以使用修饰词 L 来限制某一动作执行的时间。不过在使用动作的修饰词时比较容易出错，除了修饰词 S 和 R(动作的置位与复位)以外，建议初学者使用其他动作的修饰词时要特别小心。

在顺序控制功能图语言 S7 Graph 中，将动作的修饰词称为动作中的命令，在 5.6 节中将详细地介绍它。

5.2.3　有向连线与转换

1. 有向连线

在顺序功能图中，随着时间的推移和转换条件的实现，将会发生步的活动状态的进展，这种进展按有向连线规定的路线和方向进行。在画顺序功能图时，将代表各步的方框按它们成为活动步的先后次序顺序排列，并且用有向连线将它们连接起来。步的活动状态习惯的进展

方向是从上到下或从左至右，在这两个方向有向连线上的箭头可以省略。如果不是上述的方向，应在有向连线上用箭头注明进展方向。在可以省略箭头的有向连线上，为了更易于理解也可以加箭头。

如果在画图时有向连线必须中断，例如在复杂的图中，或用几个图来表示一个顺序功能图时，应在有向连线中断之处标明下一步的标号和所在的页数。

2. 转换

转换用有向连线上与有向连线垂直的短划线来表示，转换将相邻两步分隔开。步的活动状态的进展是由转换的实现来完成的，并与控制过程的发展相对应。

3. 转换条件

转换条件是与转换相关的逻辑命题，转换条件可以用文字语言来描述，例如“触点 A 与触点 B 同时闭合”，也可以用表示转换的短线旁边的布尔代数表达式来表示，例如 I0.1 + $\overline{\text{I2.0}}$。S7 Graph 中的转换条件用梯形图或功能块图来表示（见图 5-14），如果没有使用 S7 Graph 语言，一般用布尔代数表达式来表示转换条件。

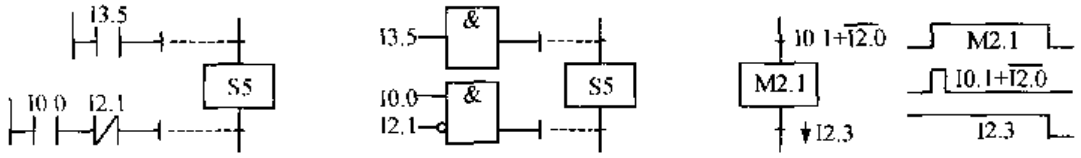


图 5-14　转换与转换条件

图 5-14 中用高电平表示步 M2.1 为活动步，反之则用低电平来表示。转换条件 I0.0 表示 I0.0 为 1 状态时转换实现，转换条件表示$\overline{\text{I0.0}}$为 0 状态时转换实现。转换条件 I0.1 + $\overline{\text{I2.0}}$表示 I0.1 的常开触点闭合或 I2.0 的常闭触点闭合时转换实现，在梯形图中则用两个触点的并联来表示这样的“或”逻辑关系。

符号 ↑I2.3 和 ↓I2.3 分别表示当 I2.3 从 0 状态变为 1 状态和从 1 状态变为 0 状态时转换实现。实际上转换条件 ↑I2.3 和 I2.3 是等效的，因为一旦 I2.3 由 0 状态变为 1 状态（即在 I2.3 的上升沿），转换条件 I2.3 也会马上起作用。

在图 5-12 中，转换条件 T0 相当于接通延时定时器 T0 的常开触点，即在 T0 的定时时间到时转换条件满足。

5.2.4　顺序功能图的基本结构

1. 单序列

单序列由一系列相继激活的步组成，每一步的后面仅有一个转换，每一个转换的后面只有一个步（见图 5-15a），单序列的特点是没有分支与合并。

2. 选择序列

选择序列的开始称为分支（见图 5-15b），转换符号只能标在水平连线之下。如果步 5 是活动步，并且转换条件 h = 1，则发生由步 5→步 8 的进展。如果步 5 是活动步，并且 k = 1，则发生由步 5→步 10 的进展。

在步 5 之后选择序列的分支处，每次只允许选择一个序列，如果将选择条件 k 改为 kh，则当 k 和 h 同时为 ON 时，将优先选择 h 对应的序列。

选择序列的结束称为合并(见图 5-15b),几个选择序列合并到一个公共序列时,用需要重新组合的序列相同数量的转换符号和水平连线来表示,转换符号只允许标在水平连线之上。

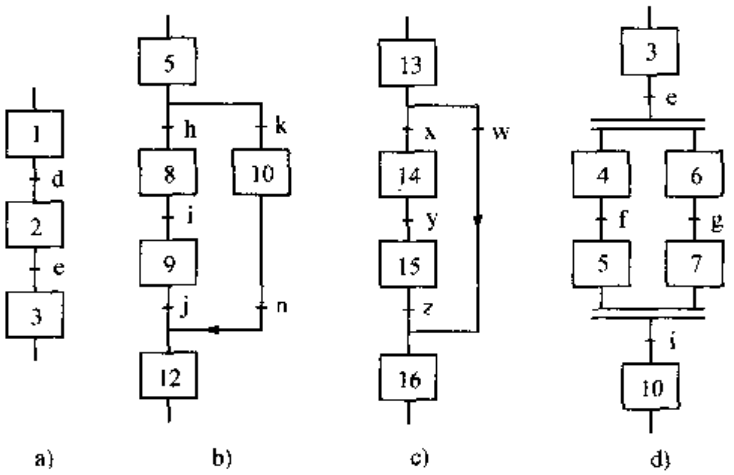


图 5-15　单序列、选择序列与并行序列

如果步 9 是活动步,并且转换条件 j=1,则发生由步 9→步 12 的进展。如果步 10 是活动步,并且 n=1,则发生由步 10→步 12 的进展。

允许选择序列的某一条分支上没有步,但是必须有一个转换。这种结构称为"跳步"(见图 5-15c)。跳步是选择序列的一种特殊情况。

3. 并行序列

并行序列的开始称为分支(见图 5-15d),当转换的实现导致几个序列同时激活时,这些序列称为并行序列。当步 3 是活动的,并且转换条件 e=1,4 和 6 这两步同时变为活动步,同时步 3 变为不活动步。为了强调转换的同步实现,水平连线用双线表示。步 4、6 被同时激活后,每个序列中活动步的进展将是独立的。在表示同步的水平双线之上,只允许有一个转换符号。并行序列用来表示系统的几个同时工作的独立部分的工作情况。

并行序列的结束称为合并(见图 5-15d),在表示同步的水平双线之下,只允许有一个转换符号。当直接连在双线上的所有前级步(步 5、7)都处于活动状态,并且转换条件 i=1 时,才会发生步 5、7 到步 10 的进展,即步 5、7 同时变为不活动步,而步 10 变为活动步。

4. 复杂的顺序功能图举例

某专用钻床用来加工圆盘状零件上均匀分布的 6 个孔(见图 5-16),上面是侧视图,下面是工件的俯视图。在进入自动运行之前,两个钻头应在最上面,上限位开关 I0.3 和 I0.5 为 ON,系统处于初始步,计数器 C0 的设定值 3 被送入计数器。在图 5-16 中用存储器位 M 来代表各步,顺序功能图中包含了选择序列和并行序列。操作人员放好工件后,按下起动按钮 I0.0,转换条件满足,由初始步转换到步 M0.1,Q4.0 变为 ON,工件被夹紧。夹紧后压力继电器 I0.1 为 ON,由步 M0.1 转换到步 M0.2 和 M0.5,Q4.1 和 Q4.3 使两只钻头同时开始向下钻孔。大钻头钻到由限位开关 I0.2 设定的深度时,进入步 M0.3,Q4.2 使大钻头上升,升到由限位开关 I0.3 设定的起始位置时停止上升,进入等待步 M0.4。小钻头钻到由限位开关 I0.4 设定的深度时,进入步 M0.6,Q4.4 使小钻头上升,升到由限位开关 I0.5 设定的起始位置时停止上升,进入等待步 M0.7,设定值为 3 的计数器 C0 的当前值减 1。减 1 后当前值为 2(非 0),C0

的常开触点闭合，转换条件 C0 满足，将转换到步 M1.0。Q4.5 使工件旋转 120°，旋转到位时 I0.6 为 ON，又返回步 M0.2 和 M0.5，开始钻第二对孔。3 对孔都钻完后，计数器的当前值变为 0，其常闭触点闭合，转换条件$\overline{C0}$满足，进入步 M1.1，Q4.6 使工件松开。松开到位时，限位开关 I0.7 为 ON，系统返回初始步 M0.0。

因为要求两个钻头向下钻孔和钻头提升的过程同时进行，采用并行序列来描述上述的过程。由 M0.2～M0.4 和 M0.5～M0.7 组成的两个单序列分别用来描述大钻头和小钻头的工作过程。在步 M0.1 之后，有一个并行序列的分支。当 M0.1 为活动步，且转换条件 I0.1 得到满足（I0.1 为 1 状态），并行序列中两个单序列中的第 1 步（步 M0.2 和 M0.5）同时变为活动步。此后两个单序列内部各步的活动状态的转换是相互独立的，例如大孔和小孔钻完时的转换一般不是同步的。

两个单序列中的最后 1 步（步 M0.4 和 M0.7）应同时变为不活动步。但是两个钻头一般不会同时上升到位，不可能同时结束运动，所以设置了等待步 M0.4 和 M0.7，它们用来同时结束两个并行序列。当两个钻头均上升到位，限位开关 I0.3 和 I0.5 分别为 1 状态，大、小钻头两个子系统分别进入两个等待步，并行序列将会立即结束。

在步 M0.4 和 M0.7 之后，有一个选择序列的分支。没有钻完 3 对孔时 C0 的常开触点闭合，转换条件满足 C0，如果两个钻头都上升到位，将从步 M0.4 和 M0.7 转换到步 M1.0。如果已钻完 3 对孔，C0 的常闭触点闭合，转换条件$\overline{C0}$满足，将从步 M0.4 和 M0.7 转换到步 M1.1。

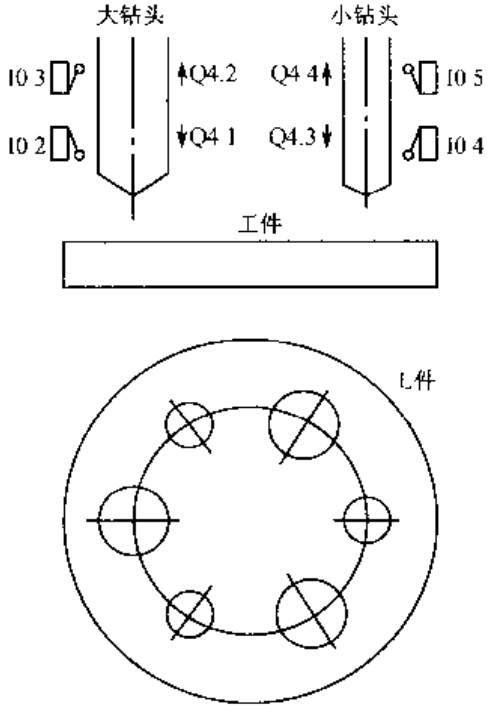


图 5-16　组合钻床示意图

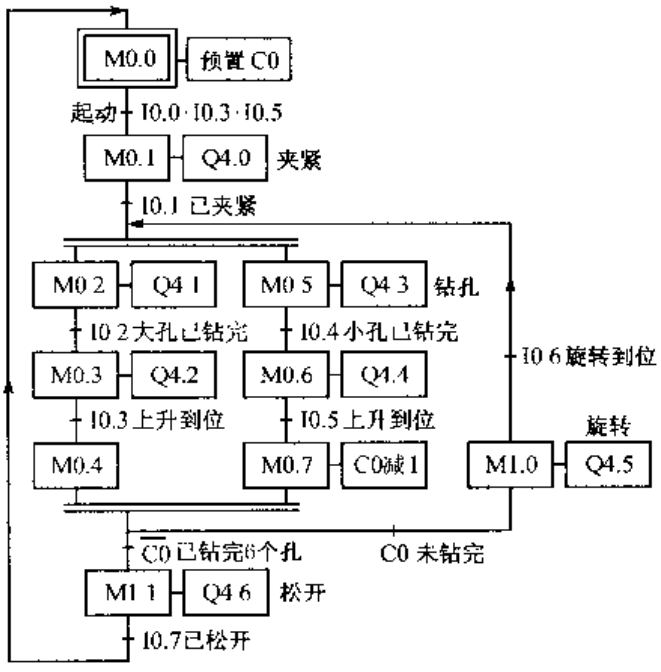


图 5-17　组合钻床的顺序功能图

在步 M0.1 之后，有一个选择序列的合并。当步 M0.1 为活动步，而且转换条件 I0.1 得到满足（I0.1 为 ON），将转换到步 M0.2 和 M0.5。当步 M1.0 为活动步，而且转换条件 I0.6 得到满足，也会转换到步 M0.2 和 M0.5。

5.2.5 顺序功能图中转换实现的基本规则

1. 转换实现的条件

在顺序功能图中，步的活动状态的进展是由转换的实现来完成的。转换实现必须同时满足两个条件：

(1) 该转换所有的前级步都是活动步；

(2) 相应的转换条件得到满足。

如果转换的前级步或后续步不止一个，转换的实现称为同步实现(见图 5-18)。为了强调同步实现，有向连线的水平部分用双线表示。

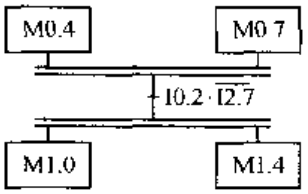


图 5-18 转换的同步实现

2. 转换实现应完成的操作

转换实现时应完成以下两个操作：

(1) 使所有由有向连线与相应转换符号相连的后续步都变为活动步；

(2) 使所有由有向连线与相应转换符号相连的前级步都变为不活动步。

以上规则可以用于任意结构中的转换，其区别如下：在单序列中，一个转换仅有一个前级步和一个后续步。在选择序列的分支与合并处，一个转换也只有一个前级步和一个后续步，但是一个步可能有多个前级步或多个后续步(见图 5-15)。在并行序列的分支处，转换有几个后续步(见图 5-18)，在转换实现时应同时将它们对应的编程元件置位。在并行序列的合并处，转换有几个前级步，它们均为活动步时才有可能实现转换，在转换实现时应将它们对应的编程元件全部复位。

转换实现的基本规则是根据顺序功能图设计梯形图的基础，它适用于顺序功能图中的各种基本结构，也是下面要介绍的各种设计顺序控制梯形图的方法的基础。

在梯形图中，用编程元件(例如存储器位 M)来代表步，当某步为活动步时，该步对应的编程元件为 1 状态。当该步之后的转换条件满足时，转换条件对应的触点或电路接通，因此可以将该触点或电路与代表所有前级步的编程元件的常开触点串联，作为与转换实现的两个条件同时满足对应的电路。例如，图 5-18 中转换条件的布尔代数表达式为 $I0.2 \cdot \overline{I2.7}$，它的两个前级步用 M0.4 和 M0.7 来代表，所以应将 I2.7 的常闭触点和 I0.2、M0.4、M0.7 的常开触点串联，作为转换实现的两个条件同时满足对应的电路。在梯形图中，该电路接通时，应使所有代表前级步的编程元件(M0.4 和 M0.7)复位，同时使所有代表后续步的编程元件(M1.0 和 M1.4)置位(变为 1 状态并保持)，完成以上任务的电路将在本章后面的内容中介绍。

5.2.6 绘制顺序功能图的注意事项

下面是针对绘制顺序功能图时常见的错误提出的注意事项：

(1) 两个步绝对不能直接相连，必须用一个转换将它们隔开。

(2) 两个转换也不能直接相连，必须用一个步将它们隔开。

(3) 顺序功能图中的初始步一般对应于系统等待启动的初始状态，这一步可能没有什么输出处于 ON 状态，因此在画顺序功能图时很容易遗漏这一步。初始步是必不可少的，一方面因为该步与它的相邻步相比，从总体上说输出变量的状态各不相同；另一方面如果没有该步，无法表示初始状态，系统也无法返回停止状态。

(4) 自动控制系统应能多次重复执行同一工艺过程，因此在顺序功能图中一般应有由步和有向连线组成的闭环，即在完成一次工艺过程的全部操作之后，应从最后一步返回初始步，系统停留在初始状态(单周期操作，见图 5-12)，在连续循环工作方式时，将从最后一步返回下一工作周期开始运行的第一步(见图 5-16)。

(5) 如果选择有断电保持功能的存储器位(M)来代表顺序控制图中的各位，在交流电源突然断电时，可以保存当时的活动步对应的存储器位的地址。系统重新上电后，可以使系统从断电瞬时的状态开始继续运行。如果用没有断电保持功能的存储器位代表各步，进入 RUN 工作方式时，它们均处于 OFF 状态，必须在 OB100 中将初始步预置为活动步，否则因顺序功能图中没有活动步，系统将无法工作。如果系统有自动、手动两种工作方式，顺序功能图是用来描述自动工作过程的，这时还应在系统由手动工作方式进入自动工作方式时，用一个适当的信号将初始步置为活动步，并将非初始步置为不活动步(见 5.5 节)。

在硬件组态时，双击 CPU 模块所在的行，打开 CPU 模块的属性对话框，选择“Retentive Memory”(有保持功能的存储器)选项卡，可以设置有断电保持功能的存储器位(M)的地址范围。

5.2.7 顺序控制设计法的本质

经验设计法实际上是试图用输入信号 I 直接控制输出信号 Q(见图 5-19a)，如果无法直接控制，或为了实现记忆、联锁、互锁等功能，只好被动地增加一些辅助元件和辅助触点。由于不同的系统的输出量 Q 与输入量 I 之间的关系各不相同，以及它们对联锁、互锁的要求千变万化，不可能找出一种简单通用的设计方法。

顺序控制设计法则是用输入量 I 控制代表各步的编程元件(例如存储器位 M)，再用它们控制输出量 Q(见图 5-19b)。步是根据输出量 Q 的状态划分的，M 与 Q 之间具有很简单的“与”的逻辑关系，输出电路的设计极为简单。任何复杂系统的代表步的 M 存储器位的控制电路，其设计方法都是相同的，并且很容易掌握，所以顺序控制设计法具有简单、规范、通用的优点。由于 M 是依次顺序变为 1 状态的，实际上已经基本上解决了经验设计法中的记忆、联锁等问题。

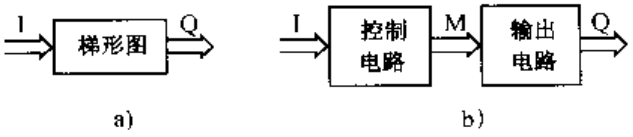


图 5-19 信号关系图

5.3 使用起保停电路的顺序控制梯形图编程方法

5.3.1 设计顺序控制梯形图的一些基本问题

S7 300/400 的编程软件 STEP 7 中的 S7 Graph 是一种顺序功能图编程语言。如果购买 STEP 7 的标准版，S7 Graph 属于可选的编程语言，需要单独付费，学习使用 S7 Graph 也需要花一定的时间。此外现在大多数 PLC(包括西门子的 S7－200 系列)还没有顺序功能图语言。因此有必要学习根据顺序功能图来设计顺序控制梯形图的编程方法。5.3～5.4 节首先介绍两种通用的编程方法，即使用起保停电路的编程方法和以转换为中心的编程方法，5.5 节介绍具有多种工作方式的控制系统的编程方法，5.6 节介绍 S7 Graph 的使用方法。

本章介绍的两种通用的编程方法很容易掌握，用它们可以迅速地、得心应手地设计出任意复杂的数字量控制系统的梯形图，它们的适用范围广，可以用于所有生产厂家的各种型号的PLC。

1．程序的基本结构

绝大多数自动控制系统除了自动工作模式外，还需要设置手动工作模式。在下列两种情况下需要工作在手动模式：

(1) 启动自动控制程序之前，系统必须处于要求的初始状态。如果系统的状态不满足启动自动程序的要求，需要进入手动工作模式，用手动操作使系统进入规定的初始状态，然后再回到自动工作模式。一般在调试阶段使用手动工作模式。

(2) 顺序自动控制对硬件的要求很高，如果有硬件故障，例如某个限位开关有故障，不可能正确地完成整个自动控制过程。在这种情况下，为了使设备不至于停机，可以进入手动工作模式，对设备进行手动控制。

有自动、手动工作方式的控制系统的两种典型的程序结构如图5-20所示，公用程序用于处理自动模式和手动模式都需要执行的任务，以及处理两种模式的相互切换。

图5-20中的I2.0是自动/手动切换开关，在左边的梯形图中，当I2.0为1时第一条条件跳转指令(JMP)的跳步条件满足，将跳过自动程序，执行手动程序，I2.0为0时第二条条件跳转指令的跳步条件满足，将跳过手动程序，执行自动程序。

在图5-20右边的梯形图中，当I2.0为1时调用处理手动操作的功能"MAN"，为0时调用处理自动操作的功能"AUTO"。

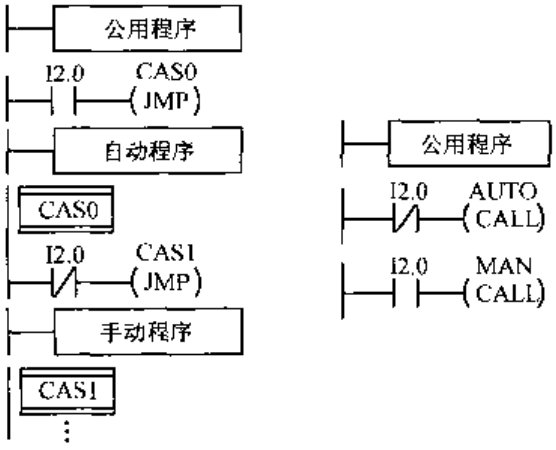


图5-20　自动/手动程序

2．执行自动程序的初始状态

开始执行自动程序之前，要求系统处于规定的初始状态。如果开机时系统没有处于初始状态，则应进入手动工作方式，用手动操作使系统进入初始状态后，再切换到自动工作方式，也可以设置使系统自动进入初始状态的工作方式(见5.5节)。

系统满足规定的初始状态后，应将顺序功能图的初始步对应的存储器位置1，使初始步变为活动步，为启动自动运行作好准备。同时还应将其余各步对应的存储器位复位为0状态，这是因为在没有并行序列或并行序列未处于活动状态时，同时只能有一个活动步。

在5.3节和5.4节中，假设用来代表步的存储器位没有被设置为有断电保持功能，刚开始执行用户程序时，系统已处于要求的初始状态，并通过OB100将初始步对应的存储器位(M)置1，其余各步对应的存储器位均为0状态，为转换的实现作好了准备。

3．双线圈问题

在图5-20的自动程序和手动程序中，都需要控制PLC的输出Q，因此同一个输出位的线圈可能会出现两次或多次，称为双线圈现象。

在跳步条件相反的两个程序段(例如图5-20中的自动程序和手动程序)中，允许出现双线圈，即同一元件的线圈可以在自动程序和手动程序中分别出现一次。实际上CPU在每一次循

环中，只执行自动程序或只执行手动程序，不可能同时执行这两个程序。对于分别位于这两个程序中的两个相同的线圈，每次循环只处理其中的一个，因此在本质上并没有违反不允许出现双线圈的规定。

在图 5-20 中用相反的条件调用功能(FC)时，也允许同一元件的线圈在自动程序功能和手动程序功能中分别出现一次。因为两个功能的调用条件相反，在一个扫描周期内只会调用其中的一个功能，而功能中的指令只是在该功能被调用时才执行，没有调用时则不执行。因此实际上 CPU 只处理被调用的功能中的双线圈元件中的一个线圈。

4. 设计顺序控制程序的基本方法

根据顺序功能图设计梯形图时，可以用存储器位 M 来代表步。为了便于将顺序功能图转换为梯形图，用代表各步的存储器位的地址作为步的代号，并用编程元件地址的逻辑代数表达式来标注转换条件，用编程元件的地址来标注各步的动作。

由图 5-19 可知，顺序控制程序分为控制电路和输出电路两部分。输出电路的输入量是代表步的编程元件 M，输出量是 PLC 的输出位 Q。它们之间的逻辑关系是极为简单的相等或相"或"的逻辑关系，输出电路是很容易设计的。

控制电路用 PLC 的输入量来控制代表步的编程元件，5.2 节中介绍的转换实现的基本规则是设计控制电路的基础。

某一步为活动步时，对应的存储器位 M 为 1 状态，某一转换实现时，该转换的后续步应变为活动步，前级步应变为不活动步。可以用一个串联电路来表示转换实现的这两个条件，该电路接通时，应将该转换所有的后续步对应的存储器位 M 置为 1 状态，将所有前级步对应的 M 复位为 0 状态。由 5.3.2 的分析可知，转换实现的两个条件对应的串联电路接通的时间只有一个扫描周期，因此应使用有记忆功能的电路或指令来控制代表步的存储器位。起保停电路和置位、复位电路都有记忆功能，本节和下一节将分别介绍使用起保停电路和置位复位电路的编程方法。

5.3.2 单序列的编程方法

起保停电路只使用与触点和线圈有关的指令，任何一种 PLC 的指令系统都有这一类指令，因此这是一种通用的编程方法，可以用于任意型号的 PLC。

1. 控制电路的编程方法

图 5-21 给出了图 5-12 中的液压动力滑台的进给运动示意图、顺序功能图和梯形图。在初始状态时动力滑台停在左边，限位开关 I0.3 为 1 状态。按下起动按钮 I0.0，动力滑台在各步中分别实现快进、工进、暂停和快退，最后返回初始位置和初始步后停止运动。

如果使用的 M 区被设置为没有断电保持功能，在开机时 CPU 调用 OB100 将初始步对应的 M0.0 置为 1 状态，开机时其余各步对应的存储器位被 CPU 自动复位为 0 状态。

设计起保停电路的关键是确定它的起动条件和停止条件。根据转换实现的基本规则，转换实现的条件是它的前级步为活动步，并且相应的转换条件满足。以控制 M0.2 的起保停电路为例，步 M0.2 的前级步为活动步时，M0.1 的常开触点闭合，它前面的转换条件满足时，I0.1 的常开触点闭合。两个条件同时满足时，M0.1 和 I0.1 的常开触点组成的串联电路接通。因此在起保停电路中，应将代表前级步的 M0.1 的常开触点和代表转换条件的 I0.1 的常开触点串联，作为控制 M0.2 的起动电路。

在快进步，M0.1 一直为 1 状态，其常开触点闭合。滑台碰到中限位开关时，I0.1 的常开触点闭合，由 M0.1 和 I0.1 的常开触点串联而成的 M0.2 的起动电路接通，使 M0.2 的线圈通电。在下一个扫描周期，M0.2 的常闭触点断开，使 M0.1 的线圈断电，其常开触点断开，使 M0.2 的起动电路断开。由以上的分析可知，起保停电路的起动电路只能接通一个扫描周期，因此必须用有记忆功能的电路来控制代表步的存储器位。

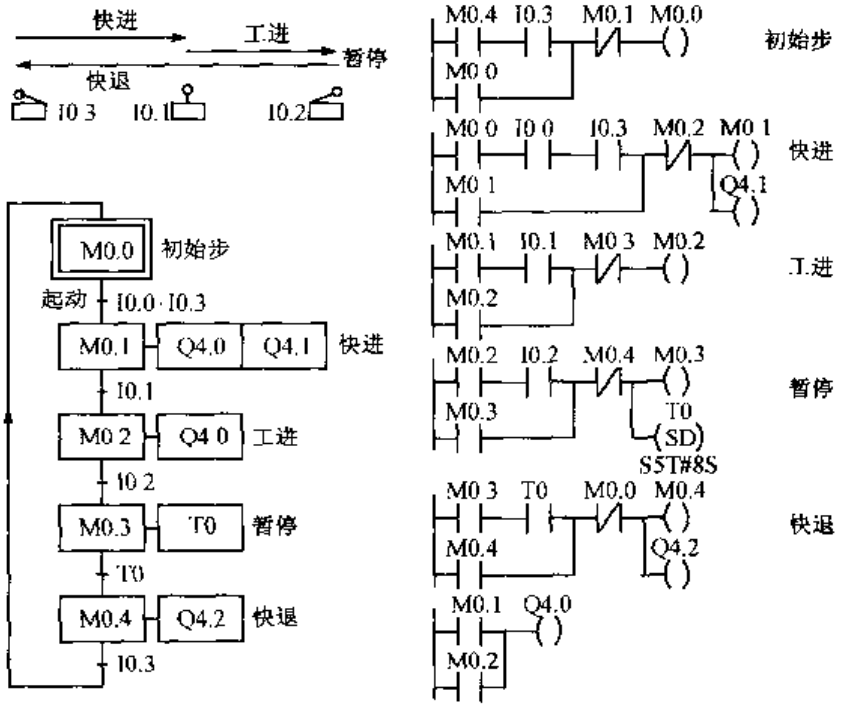


图 5-21　液压动力滑台的顺序的功能图

当 M0.2 和 I0.2 的常开触点均闭合时，步 M0.3 变为活动步，这时步 M0.2 应变为不活动步，因此可以将 M0.3 = 1 作为使存储器位 M0.2 变为 0 状态的条件，即将 M0.3 的常闭触点与 M0.2 的线圈串联。上述的逻辑关系可以用逻辑代数式表示为

$$M0.2=(M0.1+M0.2)\cdot\overline{M0.3}$$

在这个例子中，可以用 I0.2 的常闭触点代替 M0.3 的常闭触点。但是当转换条件由多个信号“与、或、非”逻辑运算组合而成时，需要将它的逻辑表达式求反，经过逻辑代数运算后再将对应的触点串并联电路作为起保停电路的停止电路，不如使用后续步对应的常闭触点这样简单方便。

根据上述的编程方法和顺序功能图，很容易画出梯形图。以步 M0.1 为例，由顺序功能图可知，M0.0 是它的前级步，二者之间的转换条件为 I0.0·I0.3，所以应将 M0.0，I0.0 和 I0.3 的常开触点串联，作为 M0.1 的起动电路。起动电路并联了 M0.0 的自保持触点。后续步 M0.2 的常闭触点与 M0.1 的线圈串联，M0.2 为 1 时 M0.1 的线圈“断电”，步 M0.1 变为不活动步。

2. 输出电路的编程方法

下面介绍设计梯形图的输出电路部分的方法。因为步是根据输出变量的状态变化来划分的，它们之间的关系极为简单，可以分为两种情况来处理：

某一输出量仅在某一步中为 ON，例如图 5-21 中的 Q4.1 就属于这种情况，可以将它的线圈与对应步的存储器位 M0.1 的线圈并联。从顺序功能图还可以看出可以将定时器 T0 的线圈与 M0.3 的线圈并联，将 Q4.2 的线圈和 M0.4 的线圈并联。

有人也许觉得既然如此，不如用这些输出位来代表该步，例如用 Q4.1 代替 M0.1。这样可以节省一些编程元件，但是存储器位 M 是完全够用的，多用一些不会增加硬件费用，在设计和输入程序时也多花不了多少时间。全部用存储器位来代表步具有概念清楚、编程规范、梯形图易于阅读和查错的优点。

如果某一输出在几步中都为 1 状态，应将代表各有关步的存储器位的常开触点并联后，驱动该输出的线圈。图 5-21 中 Q4.0 在 M0.1 和 M0.2 这两步中均应工作，所以用 M0.1 和 M0.2 的常开触点组成的并联电路来驱动 Q4.0 的线圈。

5.3.3 选择序列的编程方法

1. 选择序列的分支的编程方法

图 5-22 中步 M0.0 之后有一个选择序列的分支，设 M0.0 为活动步，当它的后续步 M0.1 或 M0.2 变为活动步时，它都应变为不活动步(M0.0 变为 0 状态)，所以应将 M0.1 和 M0.2 的常闭触点与 M0.0 的线圈串联。

如果某一步的后面有一个由 N 条分支组成的选择序列，该步可能转换到不同的 N 步去，则应将这 N 个后续步对应的存储器位的常闭触点与该步的线圈串联，作为结束该步的条件。

2. 选择序列的合并的编程方法

图 5-22 中，步 M0.2 之前有一个选择序列的合并，当步 M0.1 为活动步(M0.1 为 1)，并且转换条件 I0.1 满足，或步 M0.0 为活动步并且转换条件 I0.2 满足，步 M0.2 都应变为活动步，即代表该步的存储器位 M0.2 的起动条件应为，M0.1·I0.1 + M0.0·I0.2，对应的起动电路由两条并联支路组成，每条支路分别由 M0.1，I0.1 或 M0.0，I0.2 的常开触点串联而成(见图 5-23)。

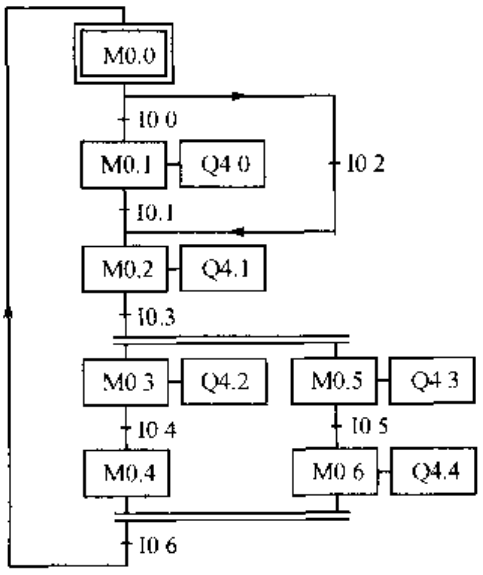


图 5-22 选择序列与并行序列

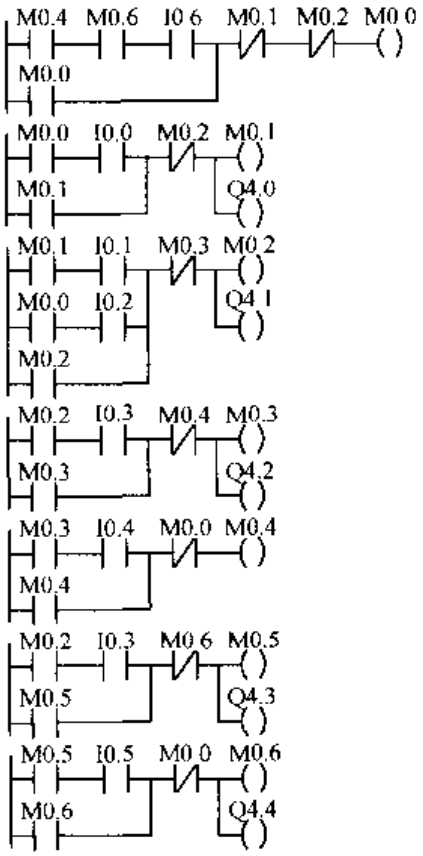


图 5-23 梯形图

一般来说，对于选择序列的合并，如果某一步之前有 N 个转换，即有 N 条分支进入该步，则代表该步的存储器位的起动电路由 N 条支路并联而成，各支路由某一前级步对应的存储器位的常开触点与相应转换条件对应的触点或电路串联而成。

5.3.4 并行序列的编程方法

1. 并行序列的分支的编程方法

图 5-22 中的步 M0.2 之后有一个并行序列的分支，当步 M0.2 是活动步并且转换条件 I0.3 满足时，步 M0.3 与步 M0.5 应同时变为活动步，这是用 M0.2 和 I0.3 的常开触点组成的串联电路分别作为 M0.3 和 M0.5 的起动电路来实现的；与此同时，步 M0.2 应变为不活动步。步 M0.3 和 M0.5 是同时变为活动步的，只需将 M0.3 或 M0.5 的常闭触点与 M0.2 的线圈串联即可。

2. 并行序列的合并的编程方法

步 M0.0 之前有一个并行序列的合并，该转换实现的条件是所有的前级步（即步 M0.4 和 M0.6）都是活动步和转换条件 I0.6 满足。由此可知，应将 M0.4、M0.6 和 I0.6 的常开触点串联，作为控制 M0.0 的起保停电路的起动电路。M0.4 和 M0.6 的线圈都串联了 M0.0 的常开触点，使步 M0.4 和步 M0.6 在转换实现时同时变为不活动步。

任何复杂的顺序功能图都是由单序列、选择序列和并行序列组成的，掌握了单序列的编程方法和选择序列、并行序列的分支、合并的编程方法，就不难迅速地设计出任意复杂的顺序功能图描述的数字量控制系统的梯形图。

5.3.5 仅有两步的闭环的处理

如果在顺序功能图中有仅由两步组成的小闭环（见图 5-24a），用起保停电路设计的梯形图不能正常工作。例如 M0.2 和 I0.2 均为 1 时，M0.3 的启动电路接通，但是这时与 M0.3 的线圈串联的 M0.2 的常闭触点却是断开的，所以 M0.3 的线圈不能“通电”。出现上述问题的根本原因在于步 M0.2 既是步 M0.3 的前级步，又是它的后续步。将图 5-24b 中的 M0.2 的常闭触点改为转换条件 I0.3 的常闭触点，就可以解决这个问题。

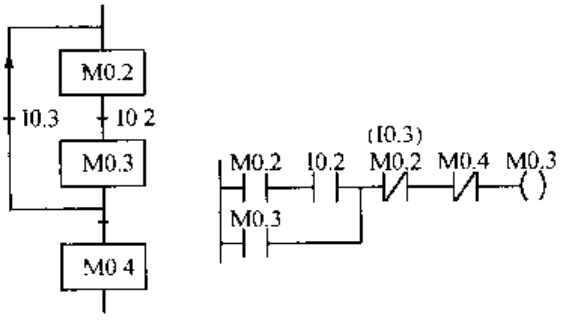


图 5-24 仅有两步的闭环的处理

5.3.6 应用举例

图 5-25 中的物料混合装置用来将粉末状的固体物料（粉料）和液体物料（液料）按一定的比例混合在一起，经过一定时间的搅拌后便得到成品。粉料和液料都用电子称来计量。

初始状态时粉料秤秤斗、液料秤秤斗和搅拌器都是空的，它们底部的排料阀关闭；液料仓的放料阀关闭，粉料仓下部的螺旋输送机的电动机和搅拌机的电动机停转；Q4.0～Q4.4 均为 0 状态。

PLC 开机后用 OB100 将初始步对应的 M0.0 置为 1 状态，将其余各步对应的存储器位复位为 0 状态，并将 MW10 和 MW12 中的计数预置值分别送给减计数器 C0 和 C1。

按下起动按钮 I0.0,Q4.0 变为 1 状态,螺旋输送机的电动机旋转,粉料进入粉料秤的秤斗;同时 Q4.1 变为 1 状态,液料仓的放料阀打开,液料进入液料秤的秤斗。电子秤的光电码盘输出与秤斗内物料重量成正比的脉冲信号。减计数器 C0 和 C1 分别对粉料秤和液料秤产生的脉冲计数。粉料脉冲计数值减至 0 时,其常闭触点闭合,粉料秤的秤斗内的物料等于预置值。Q4.0 变为 0 状态,螺旋输送机的电动机停机。液料脉冲计数值减至 0 时,其常闭触点闭合,液料秤的秤斗内的物料等于预置值。Q4.1 变为 0 状态,关闭液料仓的放料阀。

计数器的当前值非 0 时,计数器的输出位为 1,反之为 0。粉料称量结束后,C0 的常闭触点闭合,转换条件 $\overline{C0}$ 满足,粉料秤从步 M0.1 转换到等待步 M0.2,预置值送给 C0,为下一次称量做好准备。同样地,液料称量结束后,液料秤从步 M0.3 转换到等待步 M0.4,预置值送给 C1。步 M0.2 和 M0.4 后面的转换条件"= 1"表示转换条件为二进制常数 1,即转换条件总是满足的。因此在两个秤的称量都结束后,M0.2 和 M0.4 同时为活动步,系统将"无条件地"转换到步 M0.5,Q4.2 变为 1 状态,打开电子秤下部的排料门,两个电子秤开始排料,排料过程用定时器 T0 定时。同时 Q4.3 变为 1 状态,搅拌机开始搅拌。T0 的定时时间到时排料结束,转换到步 M0.6,搅拌机继续搅拌。T1 的定时时间到时停止搅拌,转换到步 M0.7,Q4.4 变为 1 状态,搅拌器底部的排料门打开,经过 T2 的定时时间后,关闭排料门,一个工作循环结束。

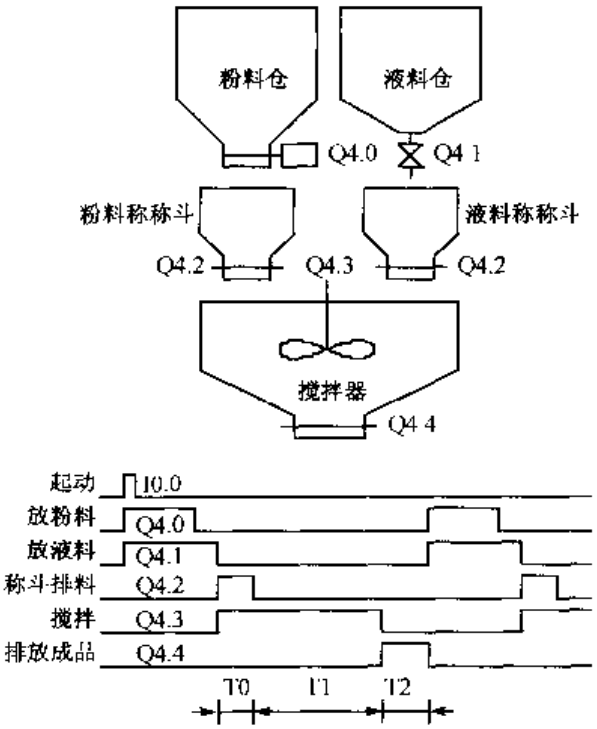


图 5-25 物料混合控制系统顺序功能图

本系统要求在按了起动按钮 I0.0 后,能连续不停地工作下去。按了停止按钮 I0.1 后,并不立即停止运行,要等到当前工艺周期的全部工作完成,成品排放结束后,再从步 M0.7 返回到初始步 M0.0。

图 5-27 中的第一个起保停电路用来实现上述要求,按下起动按钮 I0.0,M1.0 变为 1 状态,系统处于连续工作模式。在顺序功能图最下面一步执行完后,T2 的常开触点闭合,转换条件 T2·M1.0 满足,将从步 M0.7 转换到步 M0.1 和 M0.3,开始下一个周期的工作。在工作循环中的任意一步(步 M0.1～M0.7)为活动步时按下停止按钮 I0.1,"连续"标志位 M1.0 变为 0 状态,但是它不会马上起作用,要等到最后一步 M0.7 的工作结束,T2 的常开触点闭合,转换条件 T2·$\overline{M1.0}$满足,才会从步 M0.7 转换到初始步 M0.0,系统停止运行。

步 M0.7 之后有一个选择序列的分支,当它的后续步 M0.0,M0.1 和 M0.3 变为活动步时,它都应变为不活动步。但是 M0.1 和 M0.3 是同时变为 1 状态的,所以只需要将 M0.0 和 M0.1 的常闭触点或 M0.0 和 M0.3 的常闭触点与 M0.7 的线圈串联。

步 M0.1 和步 M0.3 之前有一个选择序列的合并,当步 M0.0 为活动步并且转换条件 I0.0 满足,或步 M0.7 为活动步并且转换条件满足,步 M0.1 和步 M0.3 都应变为活动步,即代表这两步的存储器位 M0.1 和步 M0.3 的起动条件应为 M0.0·I0.0 + M0.7·T2·M1.0,对

应的起动电路由两条并联支路组成，每条支路分别由 M0.0，I0.0 或 M0.7，T2，M1.0 的常开触点串联而成（见图 5-27）。

图 5-26 中步 M0.0 之后有一个并行序列的分支，当 M0.0 是活动步，并且转换条件 I0.0 满足；或者 M0.7 是活动步，并且转换条件 T2·M1.0 满足，步 M0.1 与步 M0.3 都应同时变为活动步。M0.1 和 M0.3 的起动电路完全相同，保证了这两步同时变为活动步。

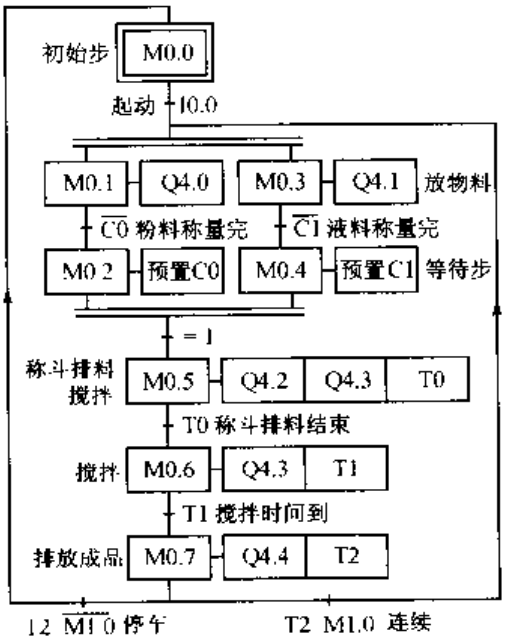


图 5-26　物料混合控制系统的梯形图

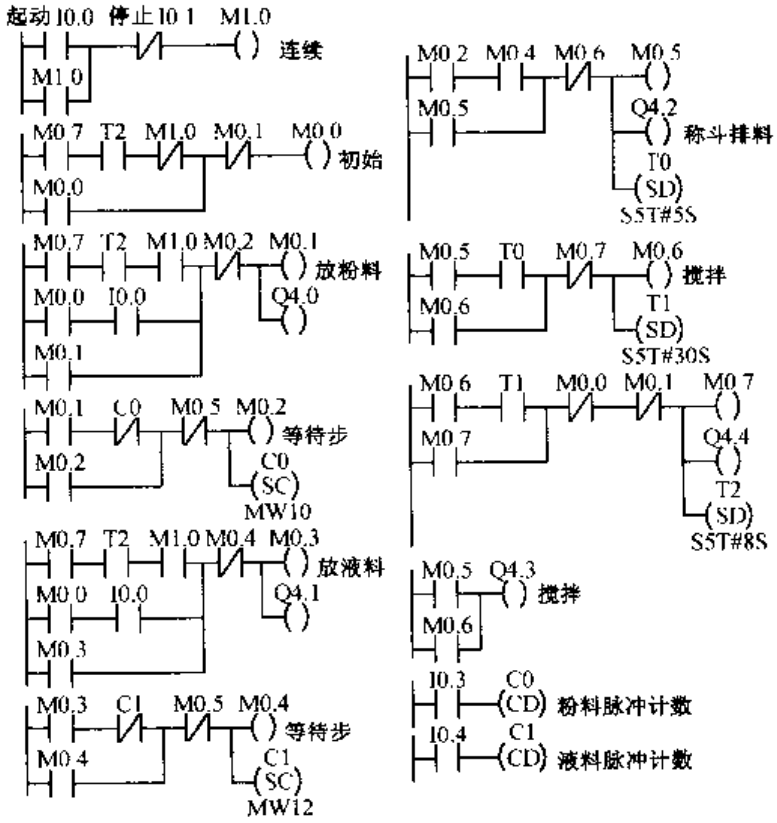


图 5-27　物料混合控制系统的梯形图

步 M0.1 与步 M0.3 是同时变为活动步的，它们的常闭触点同时断开，因此 M0.0 的线圈只需串联 M0.1 或 M0.3 的常闭触点即可。当然也可以同时串联 M0.1 与 M0.3 的常闭触点，但是要多用一条指令。

步 M0.5 之前有一个并行序列的合并，由步 M0.2 和步 M0.4 转换到步 M0.5 的条件是所有的前级步（即步 M0.2 和 M0.4）都是活动步和转换条件（=1）满足。因为转换条件总是满足的，所以只需将 M02 和 M0.4 的常开触点串联，作为 M0.5 的起动电路就可以了。可以将转换条件“=1”理解为起动电路中的一条看不见的短接线。

为了进一步提高生产效率，两个电子秤的称量过程与搅拌过程可以同时进行，它们的工作过程可以用有 3 条单序列的并行系列来描述，在称量和搅拌都完成后排放成品，然后开始搅拌和将秤斗中的原料放入搅拌机中。放料结束后关闭称斗底部的卸料门，两个秤的料斗又开始进料和称量的过程。

实际的物料混合系统（例如混凝土搅拌系统和橡胶工业中的密炼机配料控制系统）要复杂得多，输入/输出量要多得多。本例为了突出重点，减少读者熟悉系统的时间，使读者尽快地掌握顺序控制梯形图的编程方法，对实际的系统作了大量的简化。

本例中使用的是PLC的普通计数器，其计数频率较低，只有几十赫兹，最大计数值为999。在实际系统中一般用高速计数器来对编码器发出的脉冲计数。

5.4 使用置位复位指令的顺序控制梯形图编程方法

5.4.1 单序列的编程方法

使用置位复位指令的顺序控制梯形图编程方法又称为以转换为中心的编程方法。图5-28给出了顺序功能图与梯形图的对应关系。实现图中的转换需要同时满足两个条件：

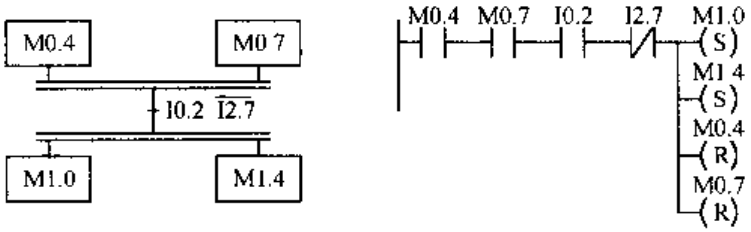


图 5-28 以转换为中心的编程方法

(1) 该转换所有的前级步都是活动步，即M0.4和M0.7均为1状态，M0.4和M0.7的常开触点同时闭合；

(2) 转换条件I0.2·I2.7满足，即I0.2的常开触点和I2.7的常闭触点组成的电路接通。

在梯形图中，可用M0.4、M0.7和I0.2的常开触点与I2.7的常闭触点组成的串联电路来表示上述两个条件同时满足。这种串联电路实际上就是使用起保停电路的编程方法中的起动电路。根据上一节的分析，该电路接通的时间只有一个扫描周期。因此需要用有记忆功能的电路来保持它引起的变化，本节用置位、复位指令来实现记忆功能。

该电路接通时，应执行两个操作：

(1) 应将该转换所有的后续步变为活动步，即将代表后续步的存储器位变为1状态，并使它保持1状态。这一要求刚好可以用有保持功能的置位指令(S指令)来完成。

(2) 应将该转换所有的前级步变为不活动步，即将代表前级步的存储器位变为0状态，并使它们保持0状态。这一要求刚好可以用复位指令(R指令)来完成。

这种编程方法与转换实现的基本规则之间有着严格的对应关系，在任何情况下，代表步的存储器位的控制电路都可以用这一个统一的规则来设计，每一个转换对应一个图5-28所示的控制置位和复位的电路块，有多少个转换就有多少个这样的电路块。这种编程方法特别有规律，在设计复杂的顺序功能图的梯形图时既容易掌握，又不容易出错。用它编制复杂的顺序功能图的梯形图时，更能显示出它的优越性。

相对而言，使用起保停电路的编程方法的规则较为复杂，选择序列的分支与合并、并行序列的分支与合并都有单独的规则需要记忆。

某工作台旋转运动的示意图如图5-29所示。工作台在初始状态时停在限位开关I0.1处，I0.1为1状态。按下起动按钮I0.0，工作台正转，旋转到限位开关I0.2处改为反转，返回限位开关I0.1处时又改为正转，旋转到限位开关I0.3处又改为反转，回到起始点时停止运动。图5-29同时给出了系统的顺序功能图和用以转换为中心的编程方法设计的梯形图。

以转换条件 I0.2 对应的电路为例，该转换的前级步为 M0.1，后续步为 M0.2，所以用 M0.1 和 I0.2 的常开触点组成的串联电路来控制对后续步 M0.2 的置位和对前级步 M0.1 的复位。每一个转换对应一个这样的“标准”电路，有多少个转换就有多少这样的电路。设计时应注意不要遗漏掉某一个转换对应的电路。

使用这种编程方法时，不能将输出位 Q 的线圈与置位指令和复位指令并联，这是因为前级步和转换条件对应的串联电路接通的时间只有一个扫描周期，转换条件满足后前级步马上被复位，下一个扫描周期该串联电路就会断开，而输出位的线圈至少应该在某一步对应的全部时间内被接通。所以应根据顺序功能图，用代表步的存储器位的常开触点或它们的并联电路来驱动输出位的线圈。

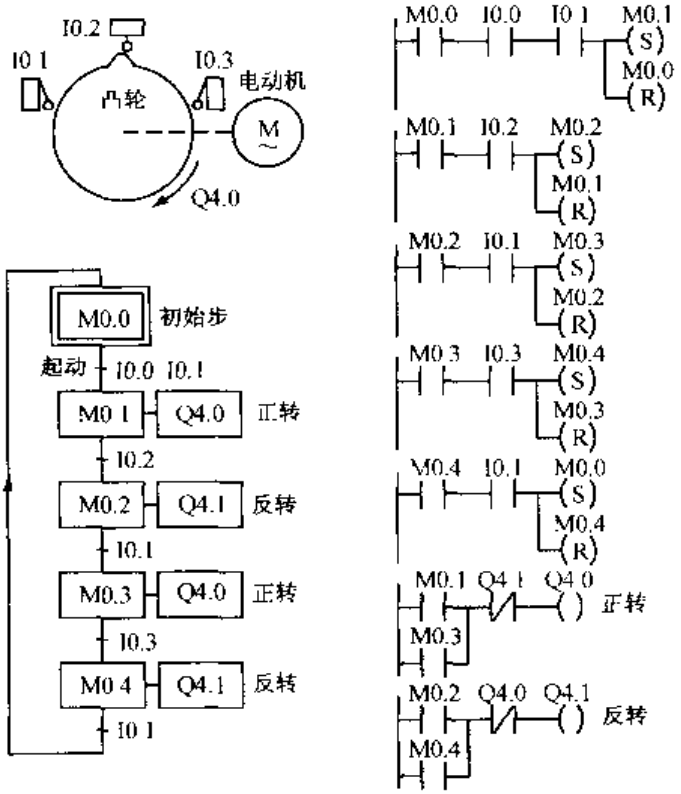


图 5-29　工作台旋转运动的顺序功能图与梯形图

5.4.2　选择序列的编程方法

使用起保停电路的编程方法时，用起保停电路来控制代表步的存储器位，实际上是站在步的立场上看问题。在选择序列的分支与合并处，某一步有多个后续步或多个前级步，所以需要使用不同的设计规则。

如果某一转换与并行序列的分支、合并无关，站在该转换的立场上看，它只有一个前级步和一个后续步（见图 5-30），需要复位、置位的存储器位也只有一个，因此选择序列的分支与合并的编程方法实际上与单序列的编程方法完全相同。

图 5-30 所示的顺序功能图中，除 I0.3 与 I0.6 对应的转换以外，其余的转换均与并行序列的分支、合并无关，I0.0～I0.2 对应的转换与选择序列的分支、合并有关，它们都只有一个前级步和一个后续步。与并行序列无关的转换对应的梯形图是非常标准的，每一个控制置位、复位的电路块都由前级步对应的存储器位和转换条件对应的触点组成的串联电路、对 1 个后续步

的置位指令和对 1 个前级步的复位指令组成。

5.4.3 并行序列的编程方法

图 5-30 中步 M0.2 之后有一个并行序列的分支，当 M0.2 是活动步，并且转换条件 I0.3 满足时，步 M0.3 与步 M0.5 应同时变为活动步，这是用 M0.2 和 I0.3 的常开触点组成的串联电路使 M0.3 和 M0.5 同时置位来实现的；与此同时，步 M0.2 应变为不活动步，这是用复位指令来实现的。

I0.6 对应的转换之前有一个并行序列的合并，该转换实现的条件是所有的前级步（即步 M0.4 和 M0.6）都是活动步和转换条件 I0.6 满足。由此可知，应将 M0.4、M0.6 和 I0.6 的常开触点串联，作为使后续步 M0.0 置位和使前级步 M0.4、M0.6 复位的条件。

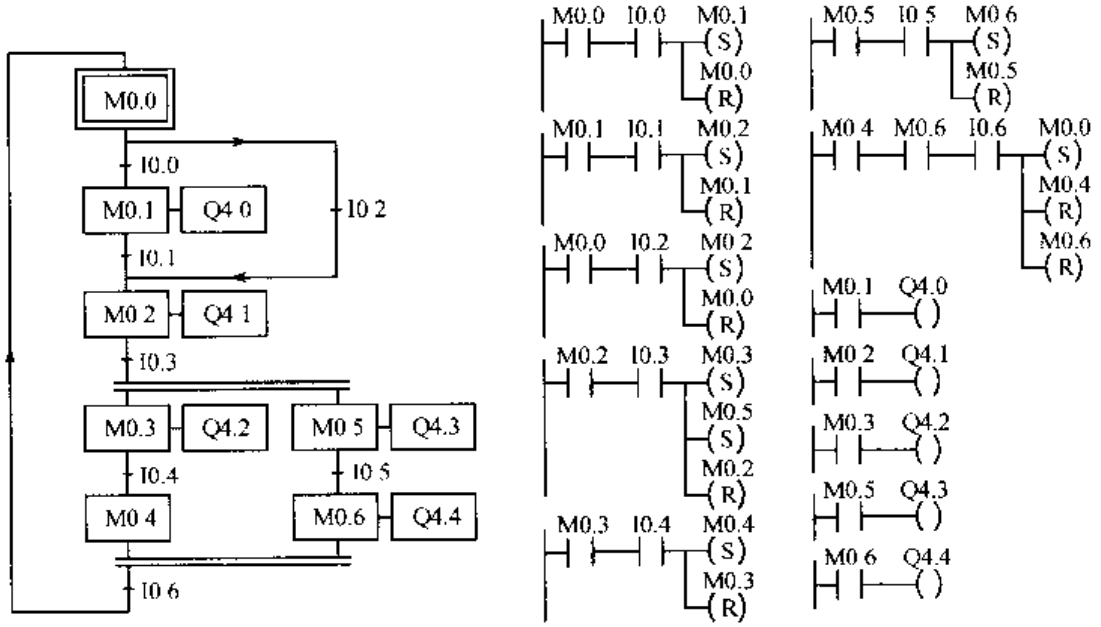


图 5-30 选择序列与并行序列

5.4.4 应用举例

图 5-31 重新给出了图 5-16 中的专用钻床控制系统的顺序功能图，图 5-32 是用以转换为中心的方法编制的梯形图。

图 5-31 中分别由 M0.2～M0.4 和 M0.5～M0.7 组成的两个单序列是并行工作的，设计梯形图时应保证这两个序列同时开始工作和同时结束，即两个序列的第一步 M0.2 和 M0.5 应同时变为活动步，两个序列的最后一步 M0.4 和 M0.7 应同时变为不活动步。

并行序列的分支的处理是很简单的，在图 5-31 中，当步 M0.1 是活动步，并且转换条件 I0.1 为 ON 时，步 M0.2 和 M0.5 同时变为活动步，两个序列开始同时工作。在梯形图中，用 M0.1 和 I0.1 的常开触点组成的串联电路来控制对 M0.2 和 M0.5 的同时置位，和对前级步 M0.1 的复位。

另一种情况是当步 M1.0 为活动步，并且转换条件 I0.6 为 ON 时，步 M0.2 和 M0.5 也应同时变为活动步，两个序列开始同时工作。在梯形图中，用 M1.0 和 I0.6 的常开触点组成的串联电路来控制对 M0.2 和 M0.5 的同时置位，和对前级步 M1.0 的复位。

图 5-31 中并行序列合并处的转换有两个前级步 M0.4 和 M0.7，根据转换实现的基本规则，当它们均为活动步并且转换条件 C0 满足，将实现并行序列的合并。未钻完 3 对孔时，减

计数器 C0 的当前值非 0，其常开触点闭合，转换条件 C0 满足，将转换到步 M1.0。在梯形图中，用 M0.4、M0.7 和 C0 的常开触点组成的串联电路使 M1.0 置位，后续步 M1.0 变为活动步；同时用 R 指令将 M0.4 和 M0.7 复位，使前级步 M0.4 和 M0.7 变为不活动步。

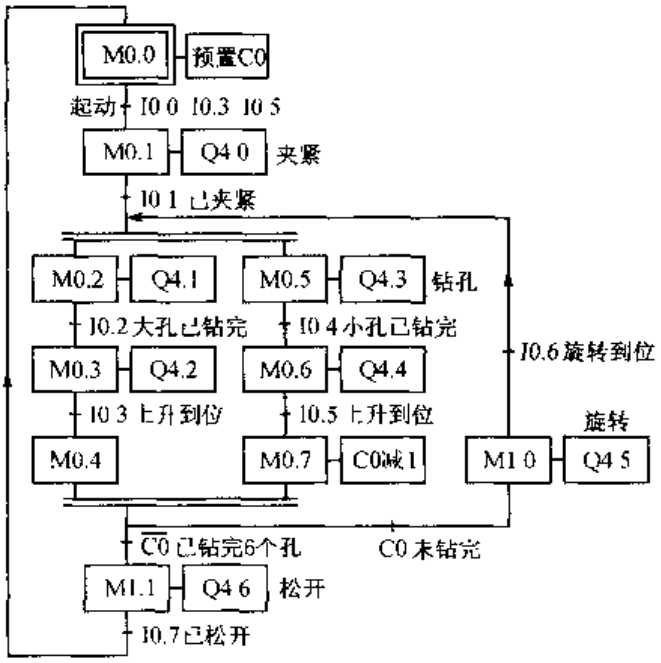


图 5-31　组合钻床的顺序功能图

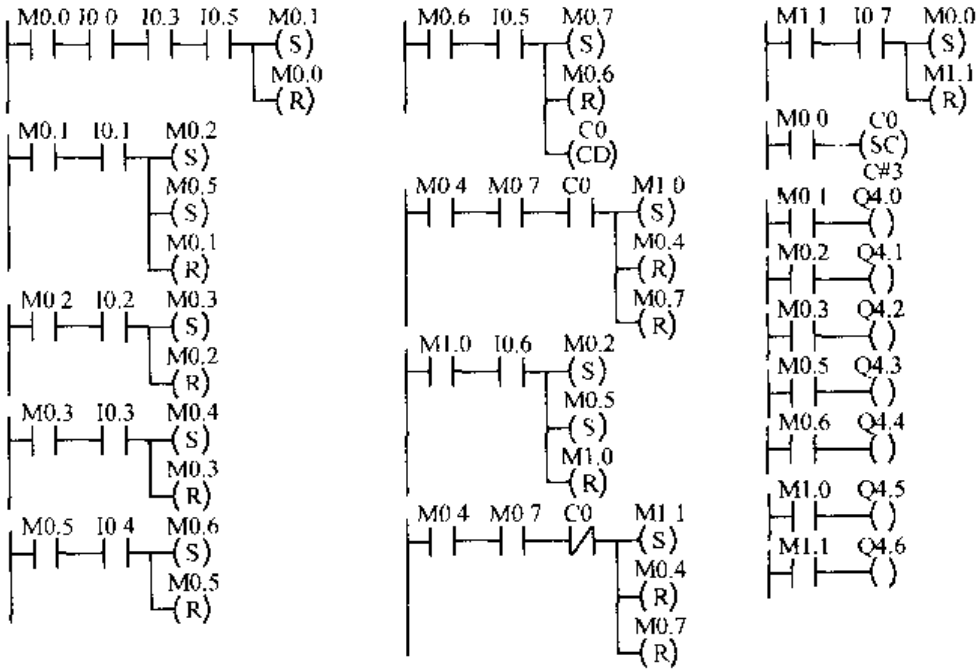


图 5-32　组合钻床控制系统的梯形图

钻完 3 对孔时，C 0 的当前值减至 0，其常闭触点闭合，转换条件$\overline{C\ 0}$满足，将转换到步 M1.1。在梯形图中，用 M0.4、M0.7 的常开触点和 C 0 的常闭触点组成的串联电路使 M1.1 置位，后续步 M1.1 变为活动步；同时用 R 指令将 M0.4 和 M0.7 复位，前级步 M0.4 和 M0.7 变为不活动步。

值得注意的是标有“CD”的 C0 的减计数线圈必须“紧跟”在图 5-32 中使 M0.7 置位的指令

后面。这是因为如果 M0.4 先变为活动步，M0.7 的“生存周期”非常短，M0.7 变为活动步后，在本次循环扫描周期内的下一个网络就被复位了。如果将 C0 的减计数线圈放在使 M0.7 复位的指令的后面，C0 还没有计数 M0.7 就被复位了，将不能执行计数操作。

5.5 具有多种工作方式的系统的顺序控制梯形图编程方法

5.5.1 机械手控制系统简介

为了满足生产的需要，很多设备要求设置多种工作方式，如手动方式和自动方式，后者包括连续、单周期、单步、自动返回初始状态几种工作方式。手动程序比较简单，一般用经验法设计，复杂的自动程序一般根据系统的顺序功能图用顺序控制法设计。

如图 5-33 所示，某机械手用来将工件从 A 点搬运到 B 点，控制面板如图 5-34 所示，图 5-35 是 PLC 的外部接线图。输出 Q4.1 为 1 时工件被夹紧，为 0 时被松开。

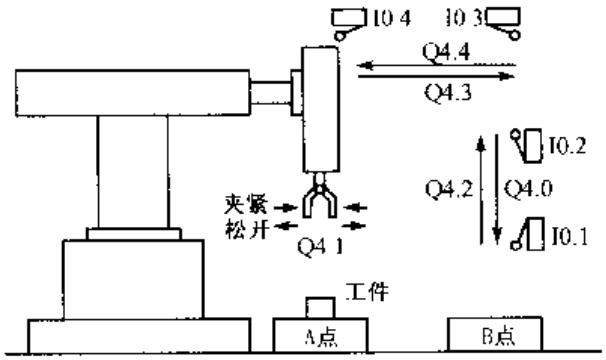


图 5-33 机械手示意图

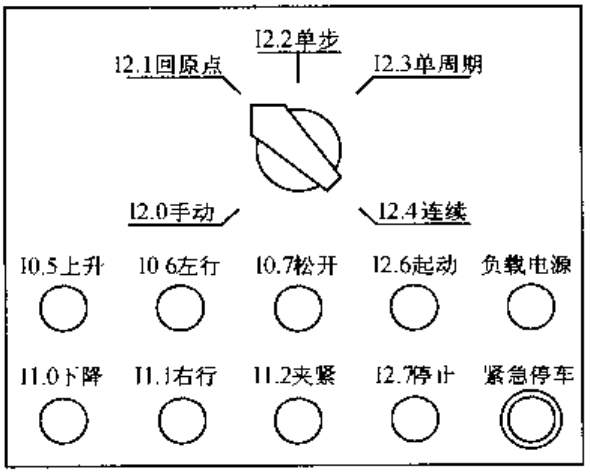


图 5-34 操作面板

工作方式选择开关的 5 个位置分别对应于 5 种工作方式，操作面板左下部的 6 个按钮是手动按钮。为了保证在紧急情况下（包括 PLC 发生故障时）能可靠地切断 PLC 的负载电源，设置了交流接触器 KM（见图 5-35）。在 PLC 开始运行时按下“负载电源”按钮，使 KM 线圈得电并自锁，KM 的主触点接通，给外部负载提供交流电源，出现紧急情况时用“紧急停车”按钮断开负载电源。

系统设有手动、单周期、单步、连续和回原点 5 种工作方式，机械手在最上面和最左边且松开时，称为系统处于原点状态（或称初始状态）。在公用程序中，左限位开关 I0.4、上限位开关 I0.2 的常开触点和表示机械手松开的 Q4.1 的常闭触点的串联电路接通时，“原点条件”存储器位 M0.5 变为 ON。

如果选择的是单周期工作方式，按下起动按钮 I2.6 后，从初始步 M0.0 开始，机械手按顺序功能图（见图 5-40）的规定完成一个周期的工作后，返回并停留在初始步。如果选择连续工作方式，在初始状态按下起动按钮后，机械手从初始步开始一个周期接一个周期地反复连续工作。按下停止按钮，并不马上停止工作，完成最后一个周期的工作后，系统才返回并停留在初始步。在单步工作方式，从初始步开始，按一下起动按钮，系统转换到下一步，完成该步的任务后，自动停止工作并停在该步，再按一下起动按钮，又往前走一步。单步工作方式常用于系统的调试。

在进入单周期、连续和单步工作方式之前，系统应处于原点状态；如果不满足这一条件，可选择回原点工作方式，然后按起动按钮 I2.6，使系统自动返回原点状态。在原点状态，顺序功能图中的初始步 M0.0 为 ON，为进入单周期、连续和单步工作方式作好了准备。

5.5.2 使用起保停电路的编程方法

1. 程序的总体结构

项目的名称为“机械手控制”，在主程序 OB1 中(见图 5-36)，用调用功能(FC)的方式来实现各种工作方式的切换。公用程序 FC 1 是无条件调用的，供各种工作方式公用。由外部接线图可知，工作方式选择开关是单刀 5 掷开关，同时只能选择一种工作方式。选择手动方式时调用手动程序 FC 2，选择回原点工作方式时调用回原点程序 FC 4，选择连续、单周期和单步工作方式时，调用自动程序 FC 3。

在 PLC 进入 RUN 运行模式的第一个扫描周期，系统调用组织块 OB100，在 OB100 中执行初始化程序。

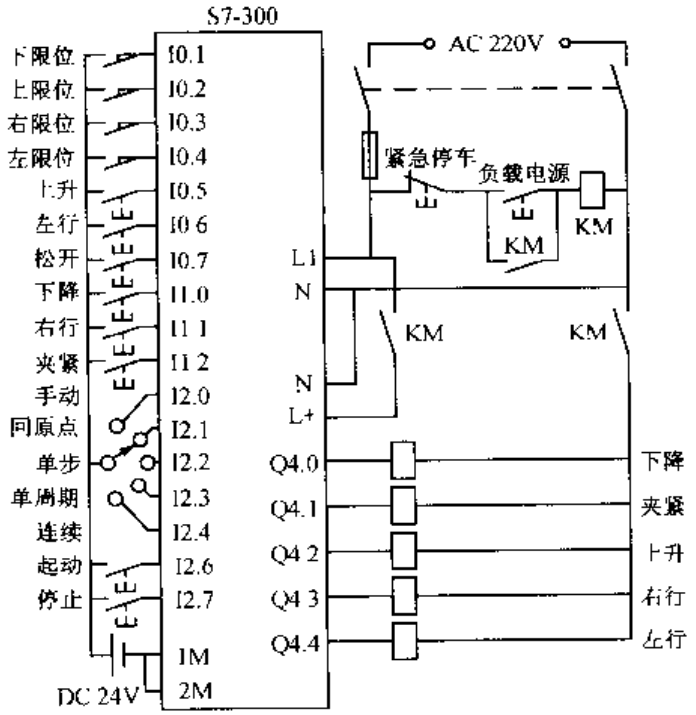


图 5-35 外部接线图

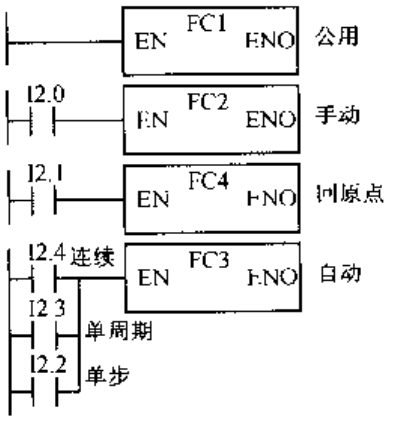


图 5-36 BOI 程序结构

2. OB 100 中的初始化程序

机械手处于最上面和最左边的位置、夹紧装置松开时，系统处于规定的初始条件，称为“原点条件”，此时左限位开关 I0.4、上限位开关 I0.2 的常开触点和表示夹紧装置松开的 Q4.1 的常闭触点组成的串联电路接通，存储器位 M0.5 为 1 状态(见图 5-37)。

对 CPU 组态时，代表顺序功能图中的各位的 MB0～MB2 应设置为没有断电保持功能，CPU 起动时它们均为 0 状态。CPU 刚进入 RUN 模式的第一个扫描周期执行图 5-37 中的组织块 OB 100 时，如果原点条件满足，M0.5 为 1 状态，顺序功能图中的初始步对应的 M0.0 被置位，为进入单步、单周期和连续工作方式作好准备。如果此时 M0.5 为 0 状态，M0.0 将被复位，初始步为不活动步，禁止在单步、单周期和连续工作方式工作。

3. 公用程序

图 5-38 中的公用程序用于自动程序和手动程序相互切换的处理。当系统处于手动工作方式和回原点方式，I2.0 或 I2.1 为 1 状态。与 OB 100 中的处理相同，如果此时满足原点条件，顺序功能图中的初始步对应的 M0.0 被置位，反之则被复位。

当系统处于手动工作方式时，I2.0 的常开触点闭合，用 MOVE 指令将顺序功能图中除初始步以外的各步对应的存储器位（M2.0～M2.7）复位，否则当系统从自动工作方式切换到手动工作方式，然后又返回自动工作方式时，可能会出现同时有两个活动步的异常情况，引起错误的动作。在非连续方式，将表示连续工作状态的标志 M0.7 复位。

4. 手动程序

图 5-39 是手动程序（见图 5-39），手动操作时用 I0.5～I1.2 对应的 6 个按钮控制机械手的升、降、左行、右行和夹紧、松开。为了保证系统的安全运行，在手动程序中设置了一些必要的联锁，例如限位开关对运动的极限位置的限制；上升与下降之间、左行与右行之间的互锁用来防止功能相反的两个输出同时为 ON。上限位开关 I0.2 的常开触点与控制左、右行的 Q4.4 和 Q4.3 的线圈串联，机械手升到最高位置才能左右移动，以防止机械手在较低位置运行时与别的物体碰撞。

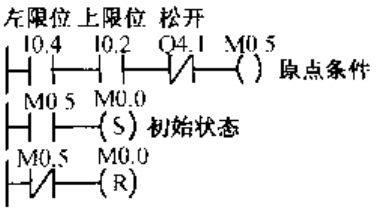


图 5-37 OB 100 初始化程序

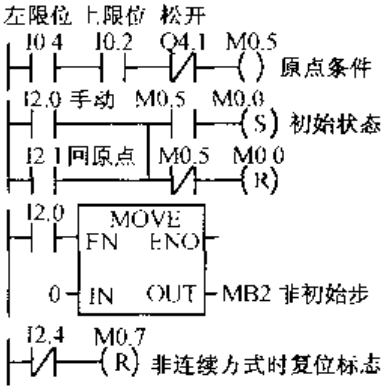


图 5-38 公用程序

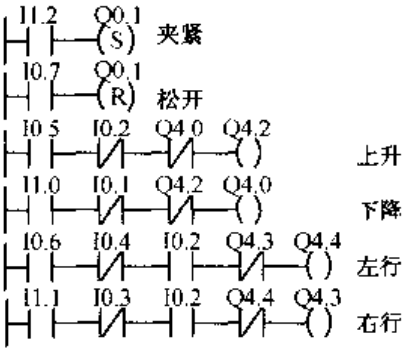


图 5-39 手动程序

5. 单周期、连续和单步程序

图 5-40 是处理单周期、连续和单步工作方式的功能 FC 3 的顺序功能图和梯形图程序。M0 和 M20～M27 用典型的起保停电路来控制。

单周期、连续和单步这 3 种工作方式主要是用“连续”标志 M0.7 和“转换允许”标志 M0.6 来区分的。

（1）单步与非单步的区分

M0.6 的常开触点接在每一个控制代表步的存储器位的起动电路中，它们断开时禁止步的活动状态的转换。如果系统处于单步工作方式，I2.2 为 1 状态，它的常闭触点断开，“转换允许”存储器位 M0.6 在一般情况下为 0 状态，不允许步与步之间的转换。当某一步的工作结束后，转换条件满足，如果没有按起动按钮 I2.6，M0.6 处于 0 状态，起保停电路的起动电路处于断开状态，不会转换到下一步。一直要等到按下起动按钮 I2.6，M0.6 在 I2.6 的上升沿 ON 一个扫描周期，M0.6 的常开触点接通，系统才会转换到下一步。

系统工作在连续、单周期（非单步）工作方式时，I2.2 的常闭触点接通，使 M0.6 为 1 状态，串联在各起保停电路的起动电路中的 M0.6 的常开触点接通，允许步与步之间的正常转换。

（2）单周期与连续的区分

在连续工作方式，I2.4 为 1 状态。在初始状态按下起动按钮 I2.6，M2.0 变为 1 状态，机械手下降。与此同时，控制连续工作的 M0.7 的线圈“通电”并自保持。

当机械手在步 M2.7 返回最左边时，I0.4 为 1 状态，因为“连续”标志位 M0.7 为 1 状态，转换条件 M0.7·I0.4 满足，系统将返回步 M2.0，反复连续地工作下去。

按下停止按钮 I2.7 后，M0.7 变为 0 状态，但是系统不会立即停止工作，在完成当前工作周期的全部操作后，在步 M2.7 返回最左边，左限位开关 I0.4 为 1 状态，转换条件$\overline{M0.7}$·I0.4 满足，系统才返回并停留在初始步。

在单周期工作方式，M0.7 一直处于 0 状态。当机械手在最后一步 M2.7 返回最左边时，左限位开关 I0.4 为 1 状态，转换条件$\overline{M0.7}$·I0.4 满足，系统返回并停留在初始步。按一次起动按钮，系统只工作一个周期。

(3) 单周期工作过程

在单周期工作方式，I2.2(单步)的常闭触点闭合，M0.6 的线圈“通电”，允许转换。在初始步时按下起动按钮 I2.6，在 M2.0 的起动电路中，M0.0、I2.6、M0.5(原点条件)和 M0.6 的常开触点均接通，使 M2.0 的线圈“通电”，系统进入下降步，Q4.0 的线圈“通电”，机械手下降；碰到下限位开关 I0.1 时，转换到夹紧步 M2.1，Q4.1 被置位，夹紧电磁阀的线圈通电并保持。同时接通延时定时器 T0 开始定时，定时时间到时，工件被夹紧，1s 后转换条件 T0 满足，转换到步 M2.2。以后系统将这样一步一步地工作下去，直到步 M2.7，机械手左行返回原点位置，左限位开关 I0.4 变为 1 状态，因为连续工作标志 M0.7 为 0 状态，将返回初始步 M0.0，机械手停止运动。

(4) 单步工作过程

在单步工作方式，I2.2 为 1 状态，它的常闭触点断开，“转换允许”辅助继电器 M0.6 在一般情况下为 0 状态，不允许步与步之间的转换。设系统处于原点状态，M0.5 和 M0.0 为 1 状态，按下起动按钮 I2.6，M0.6 变为 1 状态，使 M2.0 的启动电路接通，系统进入下降步。放开起动按钮后，M0.6 变为 0 状态。在下降步，Q4.0 的线圈“通电”，当下限位开关 I0.1 变为 1 状态时，与 Q4.0 的线圈串联的 I0.1 的常闭触点断开(见图 5-41 输出电路中最上面的梯形图)，使 Q4.0 的线圈“断电”，机械手停止下降。I0.1 的常开触点闭合后，如果没有按起动按钮，I2.6 和 M0.6 处于 0 状态，不会转换到下一步。一直要等到按下起动按钮，I2.6 和 M0.6 变为 1 状态，M0.6 的常开触点接通，转换条件 I0.1 才能使图 5-40 中的 M2.1 的启动电路接通，M2.1 的线圈“通电”并自保持，系统才能由步 M2.0 进入步 M2.1。以后在完成某一步的操作后，都必须按一次起动按钮，系统才能转换到下一步。

图 5-40 中控制 M0.0 的起保停电路如果放在控制 M2.0 的起保停电路之前，在单步工作方式步 M2.7 为活动步时按起动按钮 I2.6，返回步 M0.0 后，M2.0 的起动条件满足，将马上进入步 M2.0。在单步工作方式，这样连续跳两步是不允许的。将控制 M2.0 的起保停电路放在控制 M0.0 的起保停电路之前和 M0.6 的线圈之后可以解决这一问题。在图 5-40 中，控制 M0.6(转换允许)的是起动按钮 I2.6 的上升沿检测信号，在步 M2.7 按起动按钮，M0.6 仅 ON 一个扫描周期，它使 M0.0 的线圈通电后，下一扫描周期处理控制 M2.0 的起保停电路时，M0.6 已变为 0 状态，所以不会使 M2.0 变为 1 状态，要等到下一次按起动按钮时，M2.0 才会变为 1 状态。

(5) 输出电路

输出电路(见图 5-41)是自动程序 FC 3 的一部分，输出电路中 I0.1～I0.4 的常闭触点是为单步工作方式设置的。以下降为例，当小车碰到限位开关 I0.1 后，与下降步对应的存储器

位 M2.0 或 M2.4 不会马上变为 OFF，如果 Q4.0 的线圈不与 I0.1 的常闭触点串联，机械手不能停在下限位开关 I0.1 处，还会继续下降，对于某些设备，可能造成事故。

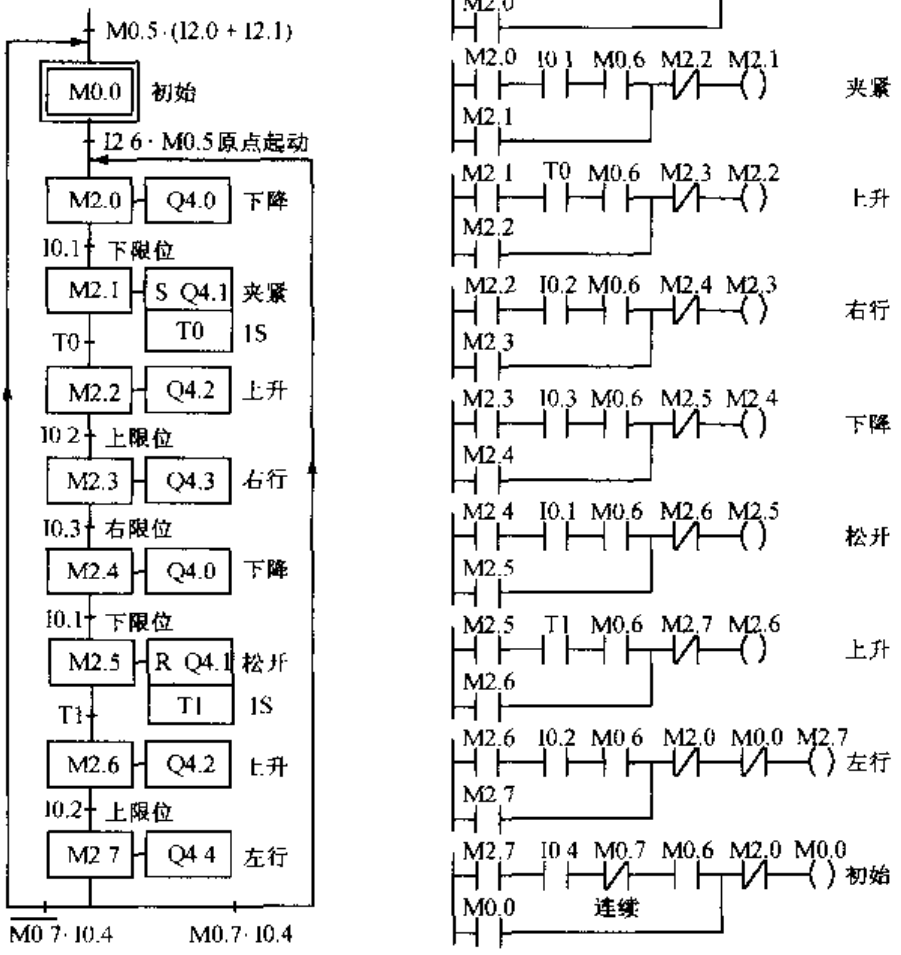


图 5-40　顺序功能图与梯形图

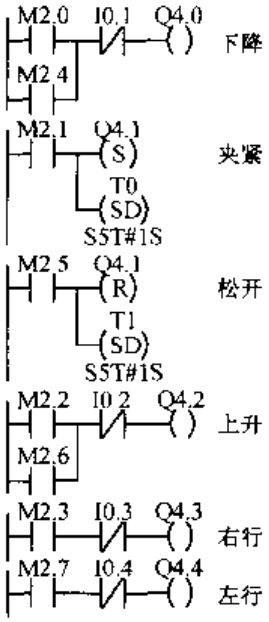


图 5-41　输出电路

6. 自动返回原点程序

图 5-42a、b 是自动回原点程序的顺序功能图和梯形图。在回原点工作方式，I2.1 为 1 状态，按下起动按钮 I2.6，M1.0 变为 1 状态并保持，机械手上升，升到上限位开关时改为左行，到左限位开关时，I0.4 变为 1 状态，将步 M1.1 和 Q4.1 复位。机械手松开后原点条件满足，M0.5 变为 1 状态，在公用程序中，FC3 中的初始步 M0.0 被置位，为进入单周期、连续或单步工作方式作好了准备，因此可以认为初始步 M0.0 是步 M1.1 的后续步。

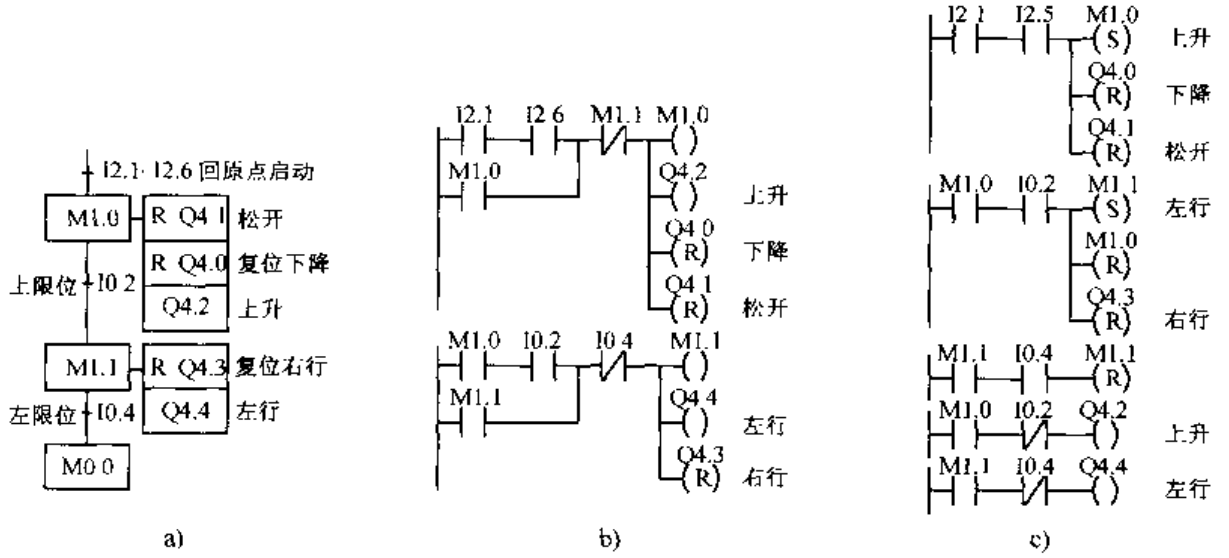


图 5-42　自动返回原点的顺序功能图与梯形图

5.5.3　使用置位复位指令的编程方法

与使用起保停电路的编程方法相比，OB1、OB 100、顺序功能图（见图 5-43）、公用程序、手动程序和自动程序中的输出电路完全相同。仍然用存储器位 M0.0 和 M2.0～M2.7 来代表各步，它们的控制电路如图 5-44 所示。该图中控制 M0.0 和 M2.0～M2.7 置位、复位的触点串联电路，与图 5-40 起保停电路中相应的起动电路相同。M0.7 与 M0.6 的控制电路与图 5-40 中的相同，

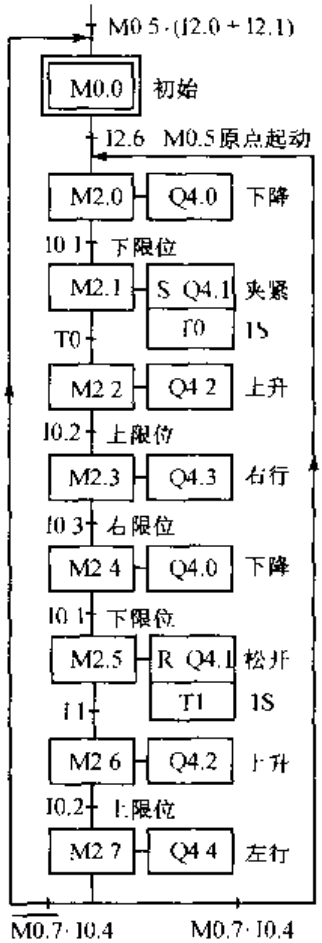


图 5-43　顺序功能图

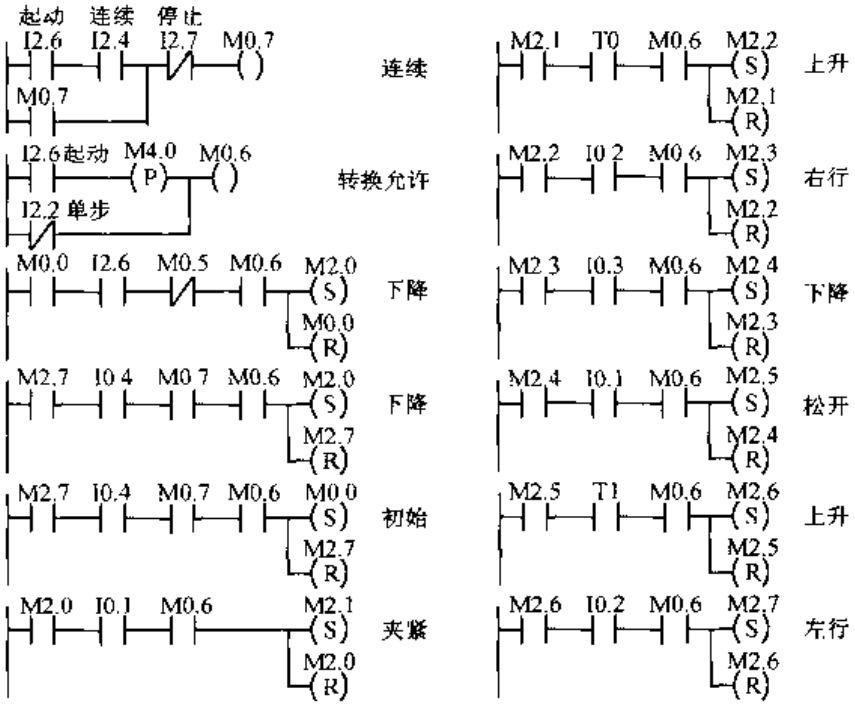


图 5-44　梯形图

自动返回原点的程序如图 5-42c 所示。

5.6 顺序功能图语言 S7 Graph 的应用

5.6.1 S7 Graph 语言概述

S7 Graph 语言是 S7-300/400 用于顺序控制程序编程的顺序功能图语言，遵从 IEC 61131-3 标准中的顺序控制语言“Sequential Function Chart”的规定。

在 S7 Graph 中，控制过程被划分为许多明确定义了功能范围的步(Step)，用图形清楚地表明整个过程的执行情况。可以为每一步指定该步要完成的动作，由每一步转向下一步的进程通过转换条件进行控制，用梯形图和功能块图语言为转换、互锁和监控等编程。

1. 顺序控制程序的结构

用 S7 Graph 编写的顺序功能图程序以功能块(FB)的形式被主程序 OB1 调用。S7 Graph FB 包含许多系统定义的参数，通过参数设置来对整个顺序系统进行控制，从而实现系统的初始化和工作方式的转换等功能。

一个顺序控制项目至少需要 3 个块(见图 5-45)：

(1) 一个调用 S7 Graph FB 的块，它可以是组织块(OB)、功能(FC)或功能块(FB)。

(2) 一个用来描述顺序控制系统各子任务(步)和相互关系(转换)的 S7 Graph FB，它由一个或多个顺序控制器(Sequencer)组成。

(3) 一个指定给 S7 Graph FB 的背景数据块(DB)，它包含了顺序控制系统的参数。

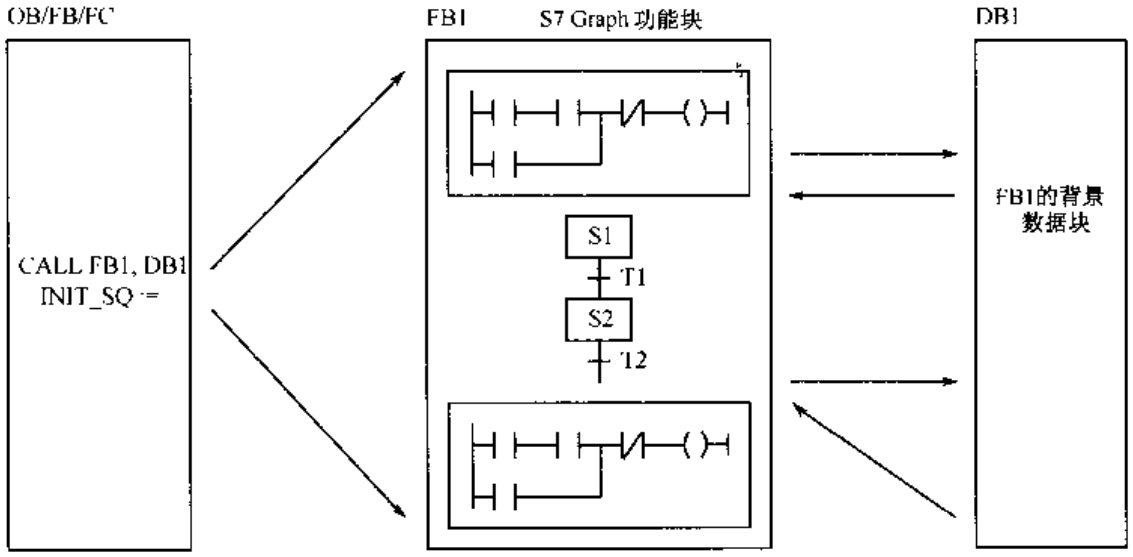


图 5-45 顺序控制系统中的块

一个 S7 Graph FB 最多可以包含 250 步和 250 个转换。

调用 S7 Graph FB 时，顺序控制器从第 1 步或从初始步开始启动。

一个顺序控制器最多包含 256 个分支，249 条并行序列的分支和 125 条选择序列的分支。实际上这与 CPU 的型号有关，一般只能用 20～40 条分支，否则执行的时间将会特别长。

可以在路径结束时，在转换之后添加一个跳步(Jump)或一个支路的结束点(Stop)。结束点将使正在执行的路径变为不活动的路径。

2. S7 Graph 编辑器

图 5-46 是 S7 Graph 的编辑器屏幕，右边的窗口是生成和编辑程序的工作区，左边的窗口是浏览窗口(Overview Window)，图中显示的是浏览窗口中的变量(Variables)选项卡，其中的变量是编程时可能用到的各种基本元素。在该选项卡可以编辑和修改现有的变量，也可以定义新的变量。可以删除，但是不能编辑系统变量。

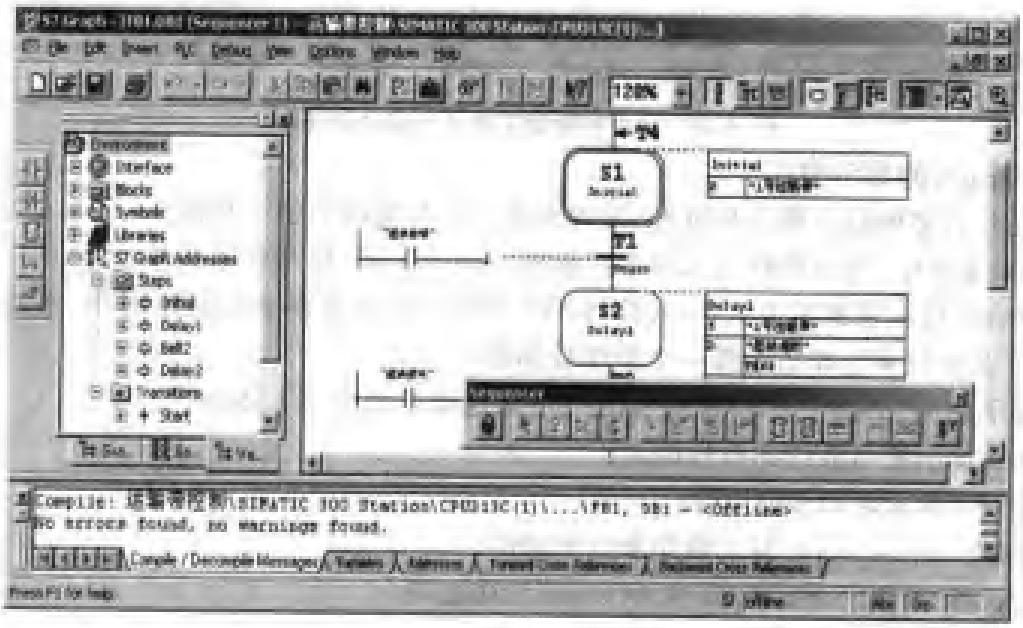


图 5-46 S7 Graph 编辑器

在保存和编译时，在屏幕下部将会出现"Details"窗口，可以获得程序编译时发现的错误和警告信息。该窗口中还有变量、符号地址和交叉参考表等大量的信息。

浏览窗口中的图形(Graphics)选项卡(见图 5-47)的中间是顺序控制器，它的上面和下面是永久性指令(Permanent instructions)。如果顺序控制器的步很多，用顺序控制器(Sequencers)选项卡(见图 5-48)来浏览顺序控制器的总体结构，或显示顺序控制器不同的部分是很方便的。

图 5-47 Graphic 选项卡

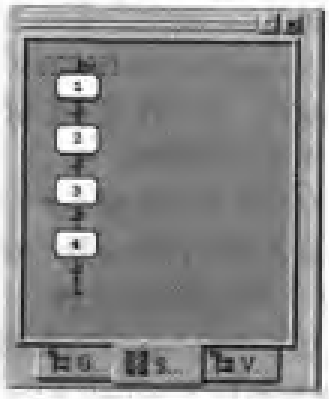


图 5-48 Sequence 选项卡

可以用图 5-46 右边窗口中浮动的"Sequencer"工具条(见图 5-49)上的按钮来放置步、转换、选择序列和跳步等。该工具条可以任意"拖放"到工作区窗口中的其他位置，也可以放到窗口上部的工具条区内，或与有触点图标的工具条垂直地放在屏幕的左边。

图 5-49　顺序控制器工具条与移动的图形

3. S7 Graph 的显示模式

S7 Graph 有多种显示模式和设置，某些设置可以与编辑的块一起保存。

在 View 菜单中，可以选择显示顺序控制器(Sequencer)、单步和永久性指令。

(1) 在顺序控制器显示方式中(见图 5-51)，如果 FB 中有多个顺序控制器，用浏览窗口中的"Graphic"选项卡来选择显示哪一个顺序控制器。

执行菜单命令"View"→"Display with"，可以选择是否显示下述内容：

Symbols：显示符号表中的符号地址；

Comments：显示块和步的注释；

Conditions and Actions：显示转换条件和动作；

Symbol List：在输入地址时显示下拉式符号地址表。

(2) 单步显示模式

在单步显示模式，只显示一个步和转换的组合(见图 5-52)，除了可以在 Sequencer 显示方式显示的内容外，还可以显示和编辑下述内容：

Supervision：监控被显示的步的条件；

Interlock：对被显示的步互锁的条件；

Step comments：执行菜单命令"View"→"Display with"→"comments"将显示和编辑步的注释。

用〈↑〉键或〈↓〉键可以显示上一个或下一个步与转换的组合。

(3) 在"permanent instructions"(永久性指令)显示方式，可以对顺序控制器之前或之后的永久性指令编程。永久性指令包括条件和块调用，不管顺序控制器的状态如何，每个扫描循环都要执行一次永久性指令。

在顺序控制器中必须在不止一处满足的条件可以作为永久性条件集中编程一次。可以用梯形图中的触点和比较器对条件编程，条件的运算结果储存在线圈内。每个永久性条件最多可以使用 32 个梯形图中的元件。

可以在永久性指令区永久性地调用使用 S7 Graph 之外的编程语言编写的块。执行了调用的块后，继续执行 S7 Graph FB。使用块调用时应注意以下问题：

可以调用使用 STL、LAD、FBD 或 SCL 语言编写的功能 FC、功能块 FB、系统功能 SFC 和系统功能块 SFB。调用 FB 和 SFB 时应指定背景数据块。在调用块之前，被调用的块应已经存在。

5.6.2　使用 S7 Graph 编程的例子

图 5-50 中的两条运输带顺序相连，为了避免运送的物料在 1 号运输带上堆积，起动时应

先起动 1 号运输带，延时 6 s 后自动起动 2 号运输带。

停机时为了避免物料的堆积，应尽量将皮带上的余料清理干净，使下一次可以轻载起动，停机的顺序应与起动的顺序相反，即按了停止按钮后，先停 2 号运输带，5 s 后再停 1 号运输带。图 5-50 给出了输入输出信号的波形图和顺序功能图。控制 1 号运输带的 Q1.0 在步 M0.1～M0.3 中都应为 1。为了简化顺序功能图和梯形图，在步 M0.1 将 Q1.0 置为 1，在初始步将 Q1.0 复位为 0。

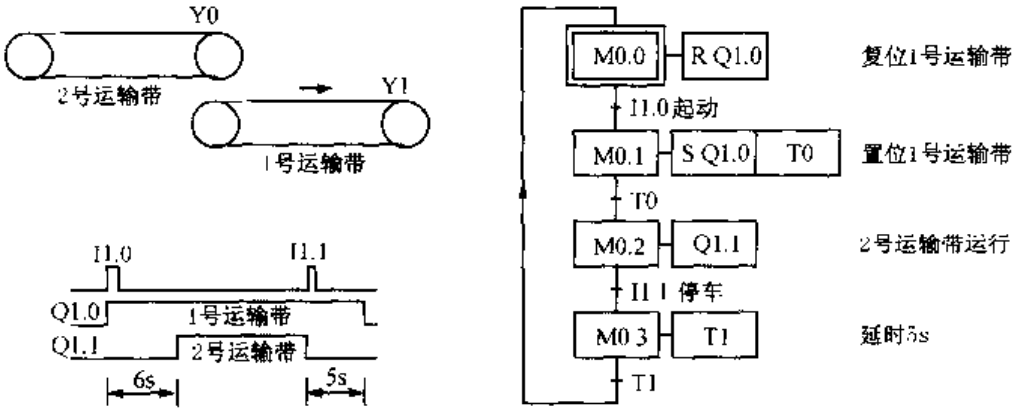


图 5-50　运输带控制系统示意图与顺序功能图

1. 创建使用 S7 Graph 语言的功能块 FB

(1) 打开 SIMATIC 管理器中的“Blocks”文件夹。

(2) 用右键点击屏幕右边的窗口，在弹出的菜单中执行命令“Insert New Object→Function Block”。

(3) 在“Properties - Function Block”对话框中选择编程语言为 GRAPH，功能块的编号为 FB 1。单击【OK】按钮确认后，自动打开刚生成的 FB 1，FB 1 中有自动生成的第 1 步 Step1 和第 1 个转换 Trans1。

2. S7 Graph 的两种编辑模式

(1) “Direct”(直接)编辑模式

执行菜单命令“Insert”→“Direct”将进入“Direct”编辑模式。

如果希望在某一元件的后面插入新的元件，首先用鼠标选择该元件，点击工具条上希望插入的元件对应的按钮，或从“Insert”菜单中选择要插入的元件。

为了在同一位置增加同类型的元件，可以连续点击工具条上同一个按钮或执行“Insert”菜单中相同的命令。

(2) “Drag and Drop”编辑模式

执行菜单命令“Insert”→“Drag-and-Drop”，将进入“Drag and Drop”(拖放)编辑模式。也可以点击工具条上最左边的【Preselected/Direct】(预选/直接)按钮，在“拖放”模式和“直接”模式之间切换。

在“拖放”模式点击工具条上的按钮，或从“Insert”菜单中选择要插入的元件后，鼠标将会带着图 5-49 右边被点击的图标移动。

如果鼠标附带的图形有“prohibited”(禁止)信号，即图 5-49 右边带红色边框的圆圈(中间有一条 45°的红线)，则表示该元件不能插在鼠标当前的位置。在允许插入该元件的区域“禁止”标志消失，点击鼠标便可以插入一个拖动的元件。

插入完同类元件后，在禁止插入的区域点击鼠标的左键，跟随鼠标移动的图形将会消失。

3．生成顺序控制器的基本框架

(1) 在 Direct 编辑模式，用鼠标选中刚打开的 FB 1 窗口中工作区内初始步下面的转换，该转换变为浅紫色。点击 3 次工具条中的步与转换按钮，将自上而下增加 3 个步和 3 个转换(见图 5-51)。

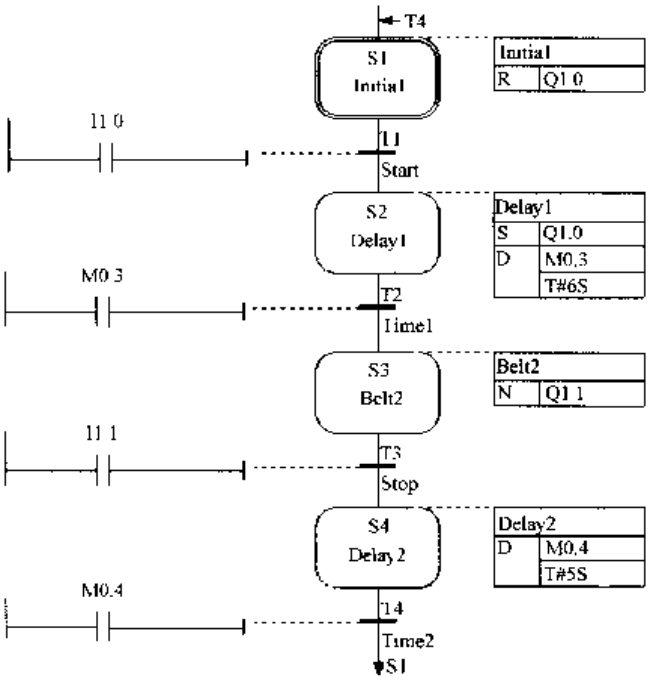


图 5-51　运输带控制系统的顺序功能图

(2) 用鼠标选中最下面的转换，点击工具条中的跳步按钮，输入跳步的目标步 S1。在步 S1 上面的有向连线上，自动出现一个水平的箭头，它的右边标有转换 T4，相当于生成了一条起于 T4，止于步 S1 的有向连线(见图 5-51)。至此步 S1～S4 形成了一个闭环。

4．步与动作的编程

表示步的方框内有步的编号(例如 S2)和步的名称(例如 Delay1)，点击后可以修改它们，不能用汉字作步和转换的名称。

执行菜单命令“View”→“Display with”→“Conditions and Actions”，可以显示或关闭各步的动作和转换条件。在“直接”模式，用鼠标右键点击步右边的动作框，在弹出的菜单中执行命令“Insert New Object”→“Action”，将插入一个空的动作行。

一个动作行由命令和地址组成，它右边的方框用来写入命令，下面是一些常用的命令：

(1) 命令 S：当步为活动步时，使输出置位为 1 状态并保持。

(2) 命令 R：当步为活动步时，使输出复位为 0 状态并保持。

(3) 命令 N：当步为活动步时，输出为 1；该步变为不活动步时，输出被复位为 0。

(4) 命令 L：用来产生宽度受限的脉冲，当该步为活动步时，该输出被置 1 并保持一段时间，该时间由 L 命令下面一行中的时间常数决定，格式为“T＃n”，n 为延时时间，例如 T＃5S。

(5) 命令 CALL：用来调用块，当该步为活动步时，调用命令中指定的块。

(6) 命令 D：使某一动作的执行延时，延时时间在该命令右下方的方框中设置，例如 T＃

5S 表示延时 5 s。延时时间到时，如果步仍然保持为活动步，则使该动作输出为 1；如果该步已变为不活动步，使该动作输出为 0。

在“直接”模式用鼠标右键点击图 5-51 中第 2 步(S2)的动作框，在弹出的菜单中选择插入动作行，在新的动作行中输入命令 S，地址为 Q1.0，即在第 2 步将控制 1 号运输带的 Q1.0 置位。

第 2 步需要延时 6 s，用右键点击第 2 步的动作框，生成新的动作行，输入命令 D(延时)，地址为 M0.3，在地址下面的空格中输入时间常数“T#6S”(6 s)。

M0.3 是步 S2 和 S3 之间的转换条件。起动延时时间到时，M0.3 的常开触点闭合，使系统从步 S2 转换到步 S3。

5. 对转换条件编程

转换条件可以用梯形图或功能块图来表示，在“View”菜单中用“LAD”或“FBD”命令来切换两种表示方法，下面介绍用梯形图来生成转换条件的方法。

点击用虚线与转换相连接的转换条件中要放置元件的位置，在图 5-46 的窗口最左边的工具条中点击常开触点、常闭触点或方框形的比较器(相当于一个触点)，用它们组成的串并联电路来对转换条件编程。生成触点后，点击触点上方的“?? .?”，输入绝对地址或符号地址。用左键选中某一地址，再用右键点击它，在弹出的菜单中执行命令“insert symbols”，将会出现符号表，使符号地址的输入更加方便。

在用比较器编程时，可以将步的系统信息作为地址来使用。下面是这些地址的意义：

Step _ name.T：步当前或最后一次被激活的时间。

Step _ name.U：步当前或最后一次被激活的时间，不包括有干扰(disturbance)的时间。

如果监控条件的逻辑运算满足，表示有干扰事件发生。

6. 对监控功能编程

双击步 S3 后，切换到单步视图(见图 5-52)，选中 Supervision(监控)线圈左边的水平线的缺口处，点击图 5-46 最左边的工具条中用方框表示的比较器图标，在比较器左边第一个引脚输入 Belt2.T，Belt2 是第 3 步的名称(2 号运输带)，在比较器左边下面的引脚输入“T#2H”，设置的监视时间为 2 h。如果该步的执行时间超过 2 h，该步被认为出错，出错步被显示为红色。

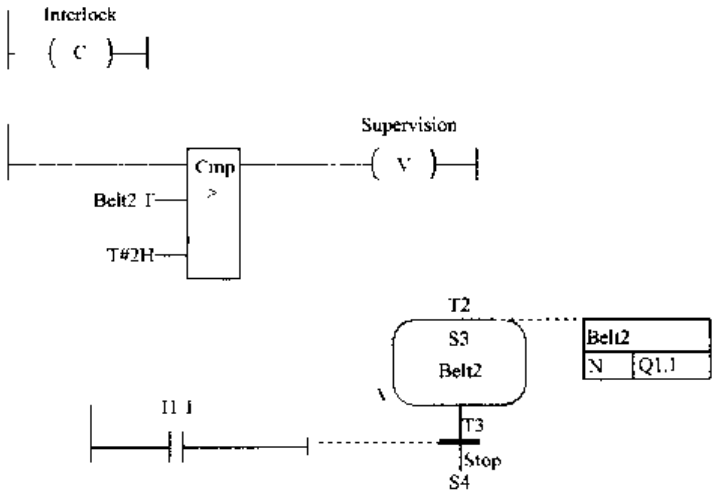


图 5-52　单步显示模式中的监控与互锁条件

7. 保存和关闭顺序控制器编辑窗口

用菜单命令“File”→“Save”保存顺序控制器时，它将被自动编译。如果程序有错误，在“Details”窗口给出错误提示和警告，改正错误后才能保存。选择菜单命令“File”→“Close”关闭顺序控制器编辑窗口。

8. 在主程序中调用 S7 Graph FB

完成了对 S7 Graph 程序 FB1 的编程后，需要在主程序 OB1 中调用 FB1，同时应指定 FB1 对应的背景数据块。为此应在 SIMATIC 管理器中首先生成 FB1 的背景数据块 DB1。

在管理器中打开“Blocks”文件夹，双击 OB1 图标，打开梯形图编辑器。选中网络 1 中用来放置元件的水平“导线”。

在 S7 Graph 编辑器中将 FB1 的参数设为 Minimum(最小)，调用它时 FB1 只有一个参数 INIT_SQ，指定用 M0.0 作 INIT_SQ 的实参。在线模式时可以用这个参数来对初始步 S1 置位。

打开编辑器左侧浏览窗口中的“FB Blocks”文件夹，双击其中的 FB1 图标，在 OB1 的网络 1 中调用顺序功能图程序 FB1，在模块的上方输入 FB1 的背景功能块 DB1 的名称。

最后用菜单命令“File”→“Save”保存 OB1，用菜单命令“File”→“Close”关闭梯形图编辑器。

9. 用 S7-PLCSIM 仿真软件调试 S7 Graph 程序

使用 S7－PLCSIM 仿真软件调试 S7 Graph 程序的步骤如下：

(1) 在 STEP 7 编程软件中生成前述的名为“运输带控制”的项目，用 S7 Graph 语言编写控制程序 FB1，其背景数据块为 DB1，在组织块 OB1 中编写调用 FB1 的程序并保存。

(2) 点击 SIMATIC 管理器工具条中的【Simulation on/off】按钮，或执行菜单命令“Options”→“Simulate Modules”，打开 S7－PLCSIM 窗口，窗口中自动出现 CPU 视图对象。与此同时，自动建立了 STEP 7 与仿真 CPU 的连接。

(3) 在 S7－PLCSIM 窗口中点击 CPU 视图对象中的 STOP 框，令仿真 PLC 处于 STOP 模式。执行菜单命令“Execute”→“Scan Mode”→“Continuous Scan”或点击【Continuous Scan】按钮，令仿真 PLC 的扫描方式为连续扫描。

(4) 在 SIMATIC 管理器左边的窗口中选中“Blocks”对象，点击工具条中的【下载】按钮，或执行菜单命令“PLC”→“Download”，将块对象下载到仿真 PLC 中。

(5) 点击 S7－PLCSIM 工具条中标有【I】的按钮，或执行菜单命令“Insert”→“Input Variable”(插入输入变量)，创建输入字节 IB1 的视图对象。用类似的方法生成输出字节 QB1、IB1 和 QB1 以位的方式显示。

图 5-53 是在 RUN 模式时监控顺序控制器的画面，图中的“起动延时”和“停止延时”分别是图 5-51 中的 M0.3 和 M0.4 的符号地址。

(6) 在 S7-PLCSIM 中模拟实际系统的操作

点击 CPU 视图对象中标有 RUN 或 RUN-P 的小框，将仿真 PLC 的 CPU 置于运行模式。在 S7 Graph 编辑器中执行菜单命令“Debug”→“Monitor”，或点击工具条内标有眼镜符号的“监控”图标，对顺序控制器的工作进程进行监控。刚开始监控时只有初始步为绿色，表示它为活动步。点击 PLCSIM 中 I1.0 对应的方框(按下起动按钮)，接着再点击 1 次，使方框内的“√”消失，模拟放开起动按钮。可以看到步 S1 变为白色，步 S2 变为绿色，表示由步 S1 转换到了步 S2。

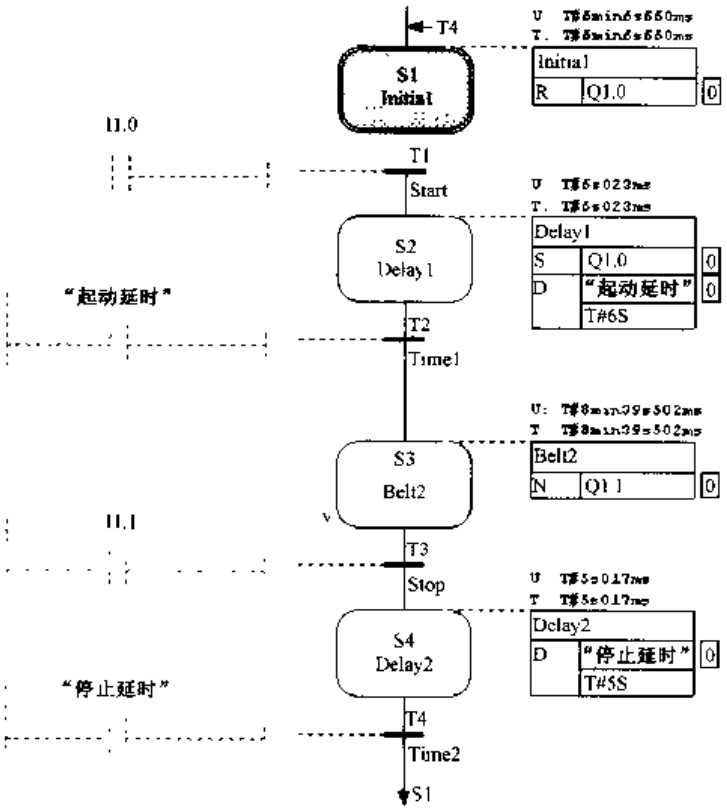


图 5-53　RUN 模式下的顺序控制器

进入步 S2 后，它的动作方框上方的两个监控定时器开始定时．它们用来计算当前步被激活的时间，其中定时器 U 不包括干扰出现的时间。定时时间达到设定值 6 s 时，步 S2 下面的转换条件满足，将自动转换到步 S3。在 PLCSIM 中用 I1.1 模拟停止按钮的操作，将会观察到由步 3 转换到步 4 的过程，延时 5 s 后自动返回初始步。

各个动作右边的小方框内是该动作的 0、1 状态。用梯形图表示的转换条件中的触点接通时，触点和它右边有“能流”流过的“导线”将变为绿色。

5.6.3　顺序控制器的运行模式与监控操作

计算机与 CPU 建立起通信联系后，将 S7 Graph FB 和它的背景数据块下载到 CPU，在 S7 Graph 编辑器中执行菜单命令“Debug”→“Control Sequencer”，在出现的对话框中（见图 5-54），可以对顺序控制器进行各种监控操作。有 4 种运行模式：自动（Automatic）、手动（Manual）、单步（Inching）、自动或切换到下一步（Automatic or switch to next）。

PLC 在 RUN 模式时，不能切换工作方式，在 RUN-P 模式时，可以在前 3 种模式之间切换。切换到新模式后，原来的模式用加粗的字体显示。

1．自动模式

在自动模式点击【Acknowledge】按钮，将确认被挂起的错误信息。当监控发生错误时，例如某步的执行时间超过监控时间，该步变为红色，功能块会产生一个错误信息。在确认错误之前，应保证产生错误的条件已不再满足。当顺序控制器转换到下一步的转换条件满足时，通过确认错误，将会强制性的转换到下一步，不会停留在出错的步。

点击【Initialize】(初始化)按钮,将重新起动顺序控制器,使之返回初始步。

点击【Disable】(禁止)按钮,使顺序控制器中所有的步变为不活动步,要激活顺序控制器应点击初始化按钮。

图 5-54 顺序控制器监控对话框

2. 手动模式

选择 Manual(手动)模式后(见图 5-54),用"【Disable】(禁止)"按钮关闭当前的活动步。在"Step Number"输入框中输入希望控制的步的编号,用【Activate】(激活)按钮或【Unactivate】(去活)按钮使该步变为活动步或不活动步。因为在单序列顺序控制器中,同时只能有 1 步是活动步,需要把当前的活动步变为不活动步后,才能激活其他的步。

3. 单步(Inching)模式

在单步模式,某一步之后的转换条件满足时,不会转换到下一步,需要点击【Continue】继续)按钮,才能使顺序控制器转换到下一步。

使用此模式应满足下述条件:

使用至少是 S7 Graph V5.0 以上的版本,S7 Graph FB 应能使用 FC 72/FC 73 在自动模式下运行,设置块的功能的选项卡"Compile/Save"中没有选择"Lock operating mode"。

4. Automatic or switch to next 模式

在该模式,即使转换条件未满足,用【Continue】按钮也能从当前步转换到后续步。如果转换条件满足,将自动转换到下一步。

5. 错误显示

没有互锁(Interlock)错误或监控(Supervision)错误时,相应的检查框为绿色,反之为红色。

点击图 5-54 中的【More】按钮,可以显示对话框中能设置的其他附加参数,详细的信息可以按(F1)键,在出现的在线帮助中得到。

5.6.4 顺序控制器中的动作

可以将动作分为标准动作和与事件有关的动作,动作中可以有定时器、计数器和算术运算。

1. 标准动作

对标准动作可以设置互锁(在命令的后面加"C"),仅在步处于活动状态和互锁条件满足时,有互锁的动作才被执行。没有互锁的动作在步处于活动状态时就会被执行。标准动作中的命令见表 5-1。

表 5-1 标准动作中的命令

命 令	地址类型	
N(或 NC)	Q,I,M,D	只要步为活动步(且互锁条件满足),动作对应的地址为 1 状态,无锁存功能
S(或 SC)	Q,I,M,D	置位:只要步为活动步(且互锁条件满足),该地址被置为 1 并保持为 1 状态
R(或 RC)	Q,I,M,D	复位:只要步为活动步(且互锁条件满足),该地址被置为 0 并保持为 0 状态

（续）

命　令	地址类型	
D(或 DC)	Q、I、M、D	延迟：(如果互锁条件满足)，步变为活动步 n 秒后，如果步仍然是活动的，该地址被置为 1 状态，无锁存功能
	T#＜常数＞	有延迟的动作的下一行为时间常数
L(或 LC)	Q、I、M、D	脉冲限制：步为活动步(且互锁条件满足)，该地址在 n 秒内为 1 状态，无锁存功能
	T#＜常数＞	有脉冲限制的动作的下一行为时间常数
CALL(或 CALL C)	FC，FB，SFC，SFB	块调用：只要步为活动步(且互锁条件满足)，指定的块被调用

表中的 Q、I、M、D 均为位地址，括号中的内容用于有互锁的动作。

2. 与事件有关的动作

动作可以与事件结合，事件是指步、监控信号、互锁信号的状态变化，信息(message)的确认(acknowledgment)或记录(registration)信号被置位。命令只能在事件发生的那个循环周期执行。见图 5-55。

图 5-55　控制动作的事件

除了命令 D(延迟)和 L(脉冲限制)外，其他命令都可以与事件进行逻辑组合。

在检测到事件，并且互锁条件被激活(对于有互锁的命令 NC、RC、SC 和 CALLC)，在下一个循环内，使用 N(NC)命令的动作为 1 状态，使用 R(RC)命令的动作被置位 1 次，使用 S(SC)命令的动作被复位 1 次。使用 CALL(CALLC)命令的动作的块被调用 1 次。

3. ON 命令与 OFF 命令

用 ON 命令或 OFF 命令分别可以使命令所在的步之外的其他步变为活动步或不活动步。

ON 和 OFF 命令取决于“步”事件，即该事件决定了该步变为活动步或变为不活动步的时间，这两条指令可以与互锁条件组合，即可以使用命令 ONC 和 OFFC。

指定的事件发生时，可以将指定的步变为活动步或不活动步。如果命令 OFF 的地址标识符为 S_ALL，将除了命令“S1(V1，L1) OFF”所在的步之外其他的步变为不活动步。

图 5-56 中的步 3 变为活动步后，各动作按下述方式执行。

控制动作的事件见表 5-2。

表 5-2　控制动作的事件

名　称	事件意义
S1	步变为活动步
S0	步变为不活动步
V1	发生监控错误(有干扰)
V0	监控错误消失(无干扰)
L1	互锁条件解除
L0	互锁条件变为 1
A1	信息被确认
R1	在输入信号 REG_EF/REG_S 的上升沿，记录信号被置位

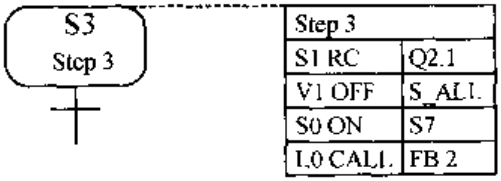


图 5-56　步与动作

(1) 一旦 S3 变为活动步和互锁条件满足，命令“S1 RC”使输出 Q2.1 复位为 0 并保持为 0。

(2) 一旦监控错误发生(出现 V1 事件),除了动作中的命令"V1 OFF"所在的步 S3,其他的活动步变为不活动步。

(3) S3 变为不活动步时(出现事件 S0),将步 S7 变为活动步。

(4) 只要互锁条件满足(出现 L0 事件),就调用指定的功能块 FB 2。

4. 动作中的计数器和定时器

(1) 计数器

动作中的计数器的执行与指定的事件有关。互锁功能可以用于计数器,对于有互锁功能的计数器,只有在互锁条件满足和指定的事件出现时,动作中的计数器才会计数。计数值为 0 时计数器位为 0,计数值非 0 时计数器位为 1。

事件发生时,计数器指令 CS 将初值装入计数器。CS 指令下面一行是要装入的计数器的初值,它可以由 IW、QW、MW、LW、DBW、BIW 来提供,或用常数 C#0~C#999 的形式给出。

事件发生时,CU、CD、CR 指令使计数值分别加 1、减 1 或将计数值复位为 0。计数器命令与互锁组合时,命令后面要加上"C"。

(2) TL 命令

动作中的定时器与计数器的使用方法类似,事件出现时定时器被执行。互锁功能也可以用于定时器。

TL 为扩展的脉冲定时器命令,该命令的下面一行是定时器的定时时间"time",定时器位没有闭锁功能。定时器的定时时间可以由 IW、QW、MW、LW、DBW、BIW 来提供,或用 S5T#time_constant 的形式给出。"#"后面是时间常数值。

一旦事件发生,定时器被起动。起动后将继续定时,而与互锁条件和步是否是活动步无关。在"time"指定的时间内,定时器位为 1,此后变为 0。正在定时的定时器可以被新发生的事件重新起动,重新起动后,在"time"指定的时间内,定时器位为 1。

(3) TD 命令

TD 命令用来实现定时器位有闭锁功能的延迟。一旦事件发生,定时器被起动。互锁条件 C 仅仅在定时器被起动的那一时刻起作用。定时器被起动后将继续定时,而与互锁条件和步的活动性无关。在"time"指定的时间内,定时器位为 0。正在定时的定时器可以被新发生的事件重新起动,重新起动后,在"time"指定的时间内,定时器位为 0,定时时间到时,定时器位变为 1。

(4) TR 命令

TR 是复位定时器命令,一旦事件发生,定时器停止定时,定时器位与定时值被复位为 0。

当图 5-57 中的步 S4 变为活动步,事件 S1 使计数器 C4 的值加 1。C4 可以用来计步 S4 变为活动步的次数。只要步 S4 变为活动步,事件 S1 使 A 的值加 1。

S4 变为活动步后,T3 开始定时,T3 的定时器位为 0 状态。4 s 后 T3 的定时器位变为 1 状态。

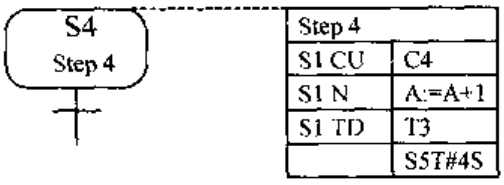


图 5-57 步与动作

5. 动作中的算术运算

在动作中可以使用下列的简单的算术表达式语句:

A:=B;

A:=函数(B);

A:=B<运算符号>C。

注意必须使用英文中的符号。包含算术表达式的动作应使用“N”命令。动作可以用事件来决定,可以设置为事件出现时执行一次,或步处于活动状态时每一扫描循环周期都执行。动作也可以与互锁组合(命令后面加“C”)。

(1) 直接指定(Direct Assignments)

直接指定可以使用下面的数据类型,用表达式 A:=B 直接指定:

8 bits:BYTE, CHAR;

16 bits:WORD, INT, DATE, S5TIME;

32 bits:DWORD, DINT, REAL, TIME, TIME_OF_DAY。

(2) 使用内置的函数

通过格式“A:=函数(B)”可以使用 S7 Graph 内置的函数,例如数据类型的转换,常用的浮点数函数,求补码、反码和循环移位等。

(3) 使用运算符号指定数学运算

格式为“A:=B<运算符号>C”,例如“A:= B + C”和“A:= B AND C”等。

5.6.5 顺序控制器中的条件

条件由梯形图或功能块图中的元件根据布尔逻辑组合而成。逻辑运算的结果(RLO)可能影响某步个别的动作、整个步、到下一步的转换或整个顺序控制器。

条件可以是事件,例如退出活动步,也可以是状态,例如输入量 I2.1 等。

条件可以在转换(Transition)、互锁(Interlock)、监控(Supervision)和永久性指令(Permanent instructions)中出现。

1. 转换条件

转换中的条件使顺序控制器从一步转换到下一步。

没有对条件编程的转换称为空转换,空转换相当于不需要转换条件的转换。

如果某一步前后的转换同时满足,该步不会变为活动步。

为此必须在 S7 Graph 编辑器中进行下面的设置:执行菜单命令“Options”→“Block Settings”,在打开的对话框的“Compile/Save”选项卡(见图 5-58)的“Sequencer Properties”栏中,选择选项“Skip Steps”(跳过步)。

2. 互锁条件

互锁是可以编程的条件,用于步的连锁,能影响某个动作的执行。

如果互锁条件的逻辑满足,受互锁控制的动作被执行,例如在互锁条件满足时,执行动作中的命令“L0 CALL FC10”,调用功能 FC10。

如果互锁条件的逻辑不满足,不执行受互锁控制的动作,发出互锁错误信号(事件 L1)。在单步显示模式对互锁编程。

3. 监控条件

监控(Supervision)是可编程的条件,用于监视步,可能影响顺序控制器从一步转换到下一步的方式。

在单步显示模式对监控编程,在所有的显示模式,用步的左下角外的字母“V”来表示该步已对监控编程(见图 5-53 中的步 S3)。

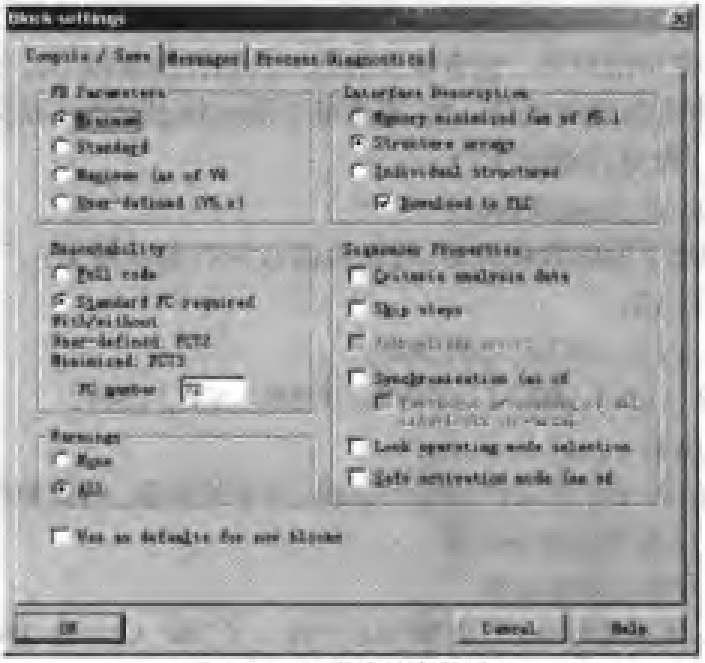


图 5-58　功能块的参数设置

如果监控条件的逻辑运算满足，表示有干扰事件 V1 发生。顺序控制器不会转换到下一步，保持当前步为活动步。监控条件满足时立即停止对步的活动时间值 Si.U 的定时。

如果监控条件的逻辑运算不满足，表示没有干扰，如果后续步的转换条件满足，顺序控制器转换到下一步。

每一步都可以设置监控条件，但是只有活动步被监控。

发出和确认监控信号之前，必须在 S7 Graph 编辑器中先执行菜单命令“Options”→“Block Settings”，在“Block Settings”对话框的“Compile/Save”选项卡中作下面的设置：

在“FB Parameters”区中选择“Standard”、“Maximum”或“User-Definable”，这样 S7 Graph 可以用功能块的输出参数 ERR _ FLT 发出监控错误信号。

在“Sequencer Properties”区中选择“Acknowledge errors”。在运行时发生监控错误，必须用功能块的输入参数 ACK _ EF 确认。

必须确认的错误只影响有关的顺序控制器序列，只有在错误被确认后，受影响的序列才能重新被处理。

4. S7 Graph 地址在条件中的应用

可以在转换、监控、互锁、动作和永久性的指令中，以地址的方式使用关于步的系统信息（见表 5-3）。

表 5-3　S7 Graph 地址

地　址	意　义	应 用 于
S.T	步 i 当前或前一次处于活动状态的时间	比较器，设置
S.U	步 i 处于活动状态的总时间，不包括干扰的时间	比较器，设置
S.X	指示步 i 是否是活动的	常开触点、常闭触点
Transi.TT	检查转换 i 所有的条件是否满足	常开触点、常闭触点

【例 5-1】 监视步的活动时间。

在很多场合需要监视减去干扰出现的时间之后，步的活动时间。例如某产品需要搅拌 50 s，可以在监控条件中监视地址 Si.U。比较 32 位整数的指令用来比较地址 Si.U 和 50 s(见图5-59)，步3 被激活的时间(不包括干扰时间)与 50s 比较，如果步 3 被激活的时间大于等于 50 s，条件满足。

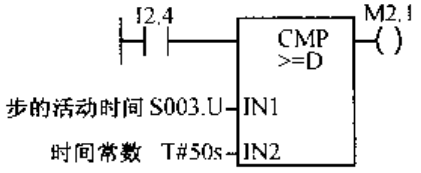


图 5-59 步的活动时间监控

5.6.6 S7 Graph 功能块的参数设置

1. 顺序控制系统的运行模式

通过对 S7 Graph FB 的参数设置，可以选择顺序控制系统的 4 种运行模式(见图 5-54)，从而决定顺序控制器对步与步之间的转换的处理方式。

(1) 自动(Automatic)模式

在自动模式，当转换条件满足时，由当前步转换到下一步。

(2) 手动(Manual)模式

与自动模式相反，在手动模式时，转换条件满足并不能转换到后续步，步的活动或不活动状态的控制是用手动完成的。

(3) 单步(Inching)模式

单步模式与自动模式的区别在于它对步与步之间的转换有附加的条件，即只能在转换条件满足和输入参数 T_PUSH 的上升沿，才能转换到下一步。

(4) 自动或切换到下一步(Automatic or step-by-step)模式

在该模式，只要转换条件满足或在功能块的输入信号 T_PUSH(见表 5-5)的上升沿，都能转换到下一步。

在 RUN 模式下可以用功能块的输入参数来选择 4 种工作模式，在下列参数的上升沿激活相应的工作模式：

SW_AUTO：自动模式；

SW_MAN：手动模式；

SW_TAP：单步(Inching)模式；

SW_TOP：自动或切换到下一步 (Automatic or Switch to next) 模式。

2. S7 Graph FB 的参数集

S7 Graph FB 有 4 种不同的参数集(见表 5-4)，图 5-60 是梯形图中最小参数集的 S7 Graph FB 符号，V5 版的“Definable/ Maximum”(可定义/最大)参数集使用表 5-5 和表 5-6 中所有的参数。

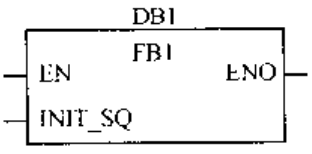


图 5-60 GRAPH 的功能块

FB 的参数集见表 5-4，S7 Graph FB 的输入参数见表 5-5，S7 Graph FB 的输出参数见表 5-6，工作模式与 S7 Graph FB 的输入参数子关系见表 5-7，4 种工作模式都要使用的 S7 Graph FB 的输出参数见表 5-8。

表 5-4　FB 的参数集

名　　称	任　　务
Minimum	最小参数集，只用于自动模式，不需要其他控制和监视功能
Standard	标准参数集，有多种操作方式，需要反馈信息，可选择确认报文
Maximum (< - V4)	最大参数集，用于 V4 及以下的版本，需要更多的操作员控制和用于服务和调试的监视功能
Definable/Maximum(V5)	可定义最大参数集，需要更多的操作员控制和用于服务和调试的监视功能，它们由 V5 的块提供

在 S7 Graph 程序编辑器中执行菜单命令“Options”→“Block Settings”，在出现的对话框的“Compile/Save”选项卡（见图 5-58）的“FB Parameters”区中，可以选择需要的参数集。为了选择不同的运行模式，必须指定除“Minimum”之外的参数集。

表 5-5　S7 Graph FB 的输入参数

参　　数	数据类型	参数说明	标准	最大
EN	BOOL	使能输入，控制 FB 的执行，如果直接连接 EN，将一直执行 FB	√	√
OFF_SQ	BOOL	OFF_SEQUENCE：关闭顺序控制器，使所有的步变为不活动步	√	√
INIT_SQ	BOOL	INIT_SEQUENCE：激活初始步，复位顺序控制器	√	√
ACK_EF	BOOL	ACKNOWLEDGE_ERROR_FAULT：确认错误和故障，强制切换到下一步	√	√
REG_EF	BOOL	REGISTRATE_ERROR_FAULT：记录所有的错误和干扰		
ACK_S	BOOL	ACKNOWLEDGE_STEP：确认在 S_NO 参数中指明的步		
REG_S	BOOL	REGISTRATE_STEP：记录在 S_NO 参数中指明的步		
HALT_SQ	BOOL	HALT_SEQUENCE：暂停/重新激活顺控器		√
HALT_TM	BOOL	HALT_TIMES：暂停/重新激活所有步的活动时间和顺控器与时间有关的命令（L 和 N）		√
ZERO_OP	BOOL	ZERO_OPERANDS：将活动步中 L、N 和 D 命令的地址复位为 0，并且不执行动作/重新激活的地址和 CALL 指令		√
EN_IL	BOOL	ENABLE_INTERLOCKS：禁止/重新激活互锁（顺控器就像互锁条件没有满足一样）		√
EN_SV	BOOL	ENABLE_SUPERVISIONS：禁止/重新激活监控（顺控器就像监控条件没有满足一样）		√
EN_ACKREQ	BOOL	ENABLE_ACKNOWLEDGE_REQUIRED：激活强制的确认请求		
DISP_SACT	BOOL	DISPLAY_ACTIVE_STEPS：只显示活动步		
DISP_SEF	BOOL	DISPLAY_STEPS_WITH_ERROR_OR_FAULT：只显示有错误和故障的步		
DISP_SALL	BOOL	DISPLAY_ALL_STEPS：显示所有的步		
S_PREV	BOOL	PREVIOUS_STEP：自动模式从当前活动步后退一步，步序号在 S_NO 中显示手动模式在 S_NO 参数中指明序号较低的前一步	√	√
S_NEXT	BOOL	NEXT_STEP：自动模式从当前活动步前进一步，步序号在 S_NO 中显示手动模式在 S_NO 参数中显示下一步（下一个序号较高的步）	√	√
SW_AUTO	BOOL	SWITCH_MODE_AUTOMATIC：切换到自动模式	√	√
SW_TAP	BOOL	SWITCH_MODE_TRANSITION_AND_PUSH：切换到 Inching（半自动）模式	√	√
SW_TOP	BOOL	SWITCH_MODE_TRANSITION_OR_PUSH：切换到“自动或转向下一步”模式		
SW_MAN	BOOL	SWITCH_MODE_MANUAL：切换到手动模式，不能触发自动执行	√	√
S_SEL	INT	STEP_SELECT：选择用于输出参数 S_ON 的指定的步，手动模式用 S_ON 和 S_OFF 激活或禁止步	√	√
S_SELOK	BOOL	STEP_SELECT_OK：将 S_SEL 中的数值用于 S_ON		

（续）

参　数	数据类型	参数说明	标准	最大
S_ON	BOOL	STEP_ON:在手动模式激活显示的步	√	√
S_OFF	BOOL	STEP_OFF:在手动模式使显示的步变为不活动步	√	√
T_PREV	BOOL	PREVIOUS_TRANSITION:在T_NO参数中显示前一个有效的转换		
T_NEXT	BOOL	NEXT_TRANSITION:在T_NO参数中显示下一个有效的转换		
T_PUSH	BOOL	PUSH_TRANSITION:条件满足并且在T_PUSH的上升沿时,转换实现。只用于单步和"automatic or step-by-step (SW_TOP)"模式。如果块是V4或更早的版本,第一个有效的转换将实现。如果块的版本为V5,且设置了输入参数T_NO,被显示编号的转换将实现;否则第一个有效转换实现	√	√
EN_SSKIP	BOOL	ENABLE_STEP_SKIPPING:激活跳步		

表 5-6　S7 Graph FB 的输出参数

参　数	数据类型	参数说明	标准	最大
ENO	BOOL	Enable output:使能输出,FB被执行且没有出错,ENO为1,否则为0	√	√
S_NO	INT	STEP_NUMBER:显示步的编号	√	√
S_MORE	BOOL	MORE_STEPS:有其他步是活动步	√	√
S_ACTIVE	BOOL	STEP_ACTIVE:被显示的步是活动步	√	√
S_TIME	TIME	TEP_TIME:步变为活动步的时间		
S_TIMEOK	TIME	STEP_TIME_OK::在步的活动期内没有错误发生		
S_CRITLOC	DWORD	STEP_CRITERIA:互锁标准位		
S_CRITLOCERR	DWORD	S_CRITERIA_IL_LAST_ERROR::用于L1事件的互锁标准位		
S_CRITSUP	DWORD	STEP_CRITERIA:监控标准位		
S_STATE	WORD	STEP_STATE::步的状态位		
T_NO	INT	TRANSITION_NUMBER:有效的转换编号		
T_MORE	BOOL	MORE_TRANSITIONS:其他用于显示的有效转换		
T_CRIT	DWORD	TRANSITION_CRITERIA:转换的标准位		
T_CRITOLD	DWORD	T_CRITERIA_LAST_CYCLE:前一周期的转换标准位		
T_CRITFLT	DWORD	T_CRITERIA_LAST_FAULT:事件V1的转换标准位		
ERROR	BOOL	INTERLOCK_ERROR:任何一步的互锁错误		
FAULT	BOOL	SUPERVISION_FAULT:任何一步的监控错误		
ERR_FLT	BOOL	IL_ERROR OR_SV_FAULT:组故障	√	√
SQ_ISOFF	BOOL	SEQUENCE_IS_OFF:顺控器完全停止(没有活动步)		
SQ_HALTED	BOOL	SEQUENCE_IS_HALTED:顺控器暂停	√	
TM_HALTED	BOOL	TIMES_ARE_HALTED:定时器停止		√
OP_ZEROED	BOOL	OPERANDS_ARE_ZEROED:地址被复位		√
IL_ENABLED	BOOL	INTERLOCK_IS_ENABLED:互锁被使能		√
SV_ENABLED	BOOL	SUPERVISION_IS_ENABLED:监控被使能		√
ACKREQ_ENABLED	BOOL	ACKNOWLEDGE_REQUIRED_IS_ENABLED:强制的确认被激活		
SSKIP_ENABLED	BOOL	STEP_SKIPPING_IS_ENABLED:跳步被激活		
SACT_DISP	BOOL	ACTIVE_STEPS_WERE_DISPLAYED:只显示S_NO参数中的活动步		
SEF_DISP	BOOL	STEPS_WITH_ERROR_FAULT_WERE_CVDISPLAYED:在S_NO参数中只显示出错的步和有故障的步		
SALL_DISP	BOOL	ALL_STEPS_WERE_DISPLAYED:在S_NO参数中显示所有的步		
AUTO_ON	BOOL	AUTOMATIC_IS_ON:显示自动模式	√	√
TAP_ON	BOOL	T_AND_PUSH_IS_ON:显示单步自动模式		
TOP_ON	BOOL	T_OR_PUSH_IS_ON:显示SW_TOP模式	√	√
MAN_ON	BOOL	MANUAL IS_ON:显示手动模式	√	√

表 5-7　工作模式与 S7 Graph FB 的输入参数的关系

参　数	数据类型	参数说明	自动	手动	单步	自动或切换到下一步
OFF_SQ	BOOL	关闭顺控器，使所有的步变为不活动	√	√	√	√
INIT_SQ	BOOL	激活初始步，初始化顺控器	√	√	√	√
ACK_EF	BOOL	确认故障，强制切换到下一步	√	√	√	√
S_PREV	BOOL	从当前的活动步后退，步序号在 S_NO 中显示	√	√	√	√
S_NEXT	BOOL	从当前的活动步向前进，步序号在 S_NO 中显示	√	√	√	√
SW_AUTO	BOOL	自动模式请求		√	√	√
SW_TAP	BOOL	单步模式请求	√	√		√
SW_TOP	BOOL	"自动或切换到下一步"模式请求	√	√	√	
SW_MAN	BOOL	手动模式请求	√	√		√
S_SEL	INT	步的选择，在手动模式用 S_ON 和 S_OFF 激活/禁止的步		√		
S_ON	BOOL	手动模式激活 S_SEL 指定的步		√		
S_OFF	BOOL	手动模式禁止 S_SEL 指定的步		√		
T_PUSH	BOOL	条件满足并且在 T_PUSH 的上升沿时，转换到下一步		√	√	

表 5-8　4 种工作模式都要使用的 S7 Graph FB 的输出参数

参　数	数据类型	参数说明
S_NO	INT	显示用 S_PREV 或 S_NEXT 选择的活动步的编号
S_MORE	BOOL	有其他步是活动步
S_ACTIVE	BOOL	S_NO 显示的步是活动步
ERR_FLT	BOOL	有故障或干扰出现
AUTO_ON	BOOL	指示自动模式
TAP_ON	BOOL	指示半自动模式
TOP_ON	BOOL	指示"自动或切换到下一步"模式
MAN_ON	BOOL	指示手动模式

手动方式用 S_SEL 选择步的方法如下：

(1) 用输入参数 SW_MAN 的上升沿选择手动模式。

(2) 用输入参数 S_SEL 指定要选择的步。

(3) 用输入参数 S_ON 或 S_OFF 的上升沿激活或禁止(去活)选择的步。

对于顺序控制器的并行序列，如果需要选择的不止一步，重复步骤(2)和(3)的操作。

5.6.7　用 S7 Graph 编写具有多种工作方式的控制程序

下面介绍的机械手控制系统的程序中，自动程序是用 S7 Graph 语言编写的。系统的功能、工作方式、操作面板和硬件接线图均与 5.5 节中的相同。

1. 符号表

表 5-9 是系统的符号表中主要的符号。

表 5-9 符 号 表

符 号	地 址	符 号	地 址	符 号	地 址	符 号	地 址	符 号	地 址
自动数据块	DB1	松开按钮	I0.7	单步	I2.2	自动方式	M0.3	下降阀	Q4.0
下限位	I0.1	下降按钮	I1.0	单周期	I2.3	原点条件	M0.5	夹紧阀	Q4.1
上限位	I0.2	右行按钮	I1.1	连续	I2.4	转换允许	M0.6	上升阀	Q4.2
右限位	I0.3	夹紧按钮	I1.2	起动按钮	I2.6	连续标志	M0.7	右行阀	Q4.3
左限位	I0.4	确认故障	I1.3	停止按钮	I2.7	回原点上升	M1.0	左行阀	Q4.4
上升按钮	I0.5	手动	I2.0	自动允许	M0.0	回原点左行	M1.1	错误报警	Q4.5
左行按钮	I0.6	回原点	I2.1	单周连续	M0.2	夹紧延时	M1.2		

2. 初始化程序、手动程序与自动回原点程序

在 PLC 进入 RUN 模式的第一个扫描周期，系统调用组织块 OB100。OB100 中的初始化程序与 5.5 节中的图 5-37 完全相同。手动程序 FC 2 与 5.5 节中的图 5-39 完全相同。自动返回原点的梯形图程序 FC 3 与 5.5 节图 5-42b 中的相同。

3. 主程序 OB1

与 5.5 节一样，在 OB1 中，用块调用的方式来实现各种工作方式的切换。公用程序（功能 FC1）是无条件调用的，供各种工作方式公用。手动工作方式时调用功能 FC2（见图 5-61），回原点工作方式时调用功能 FC3，连续、单周期和单步工作方式（总称为“自动方式”）时，调用用 S7 Graph 语言编写的功能块 FB1，它的背景数据块 DB1 的符号名为“自动数据块”。

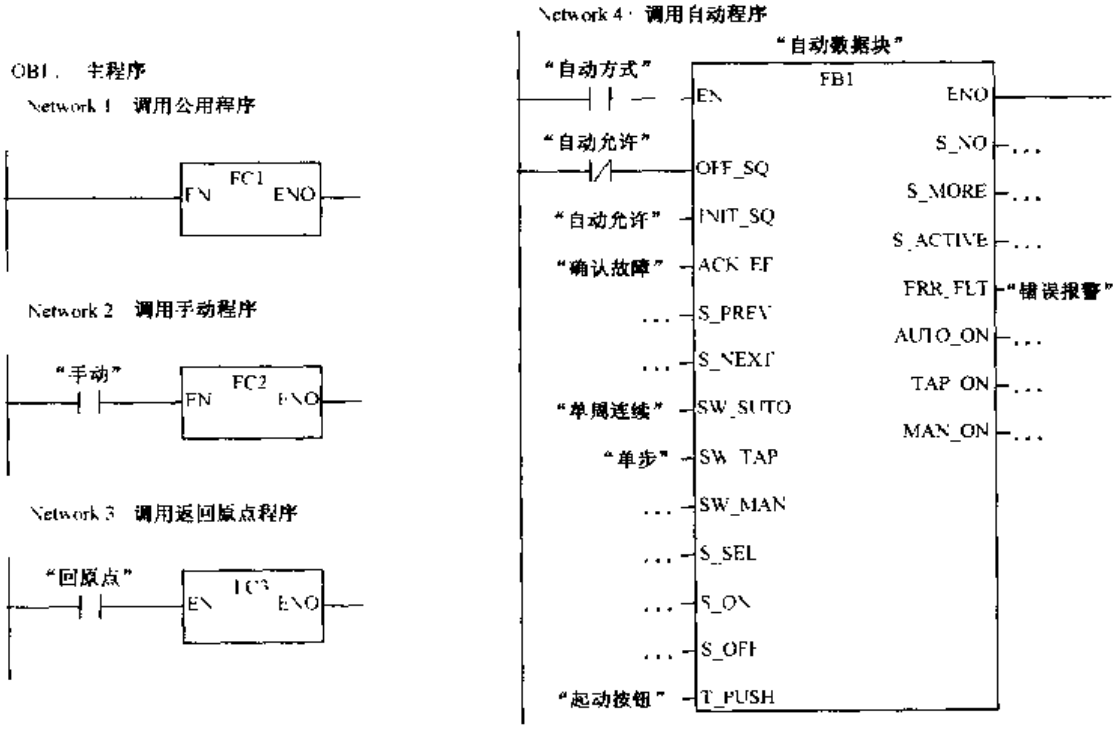


图 5-61 主程序 OB1

4. 公用程序

图 5-62 是公用程序 FC1，在手动方式或自动回原点方式，如果原点条件满足，图中的“自

动允许"(M0.0)被置位为1,M0.0的常开触点闭合,使FB1的输入参数INIT_SQ(激活初始步)为1,它使初始步变为活动步,为自动程序的执行做好准备。原点条件不满足时,M0.0被复位为0,M0.0的常闭触点使FB1的输入信号OFF_SQ(关闭顺序控制器)为1状态,将顺序控制器中所有的活动步变为不活动步,禁止自动程序的执行。

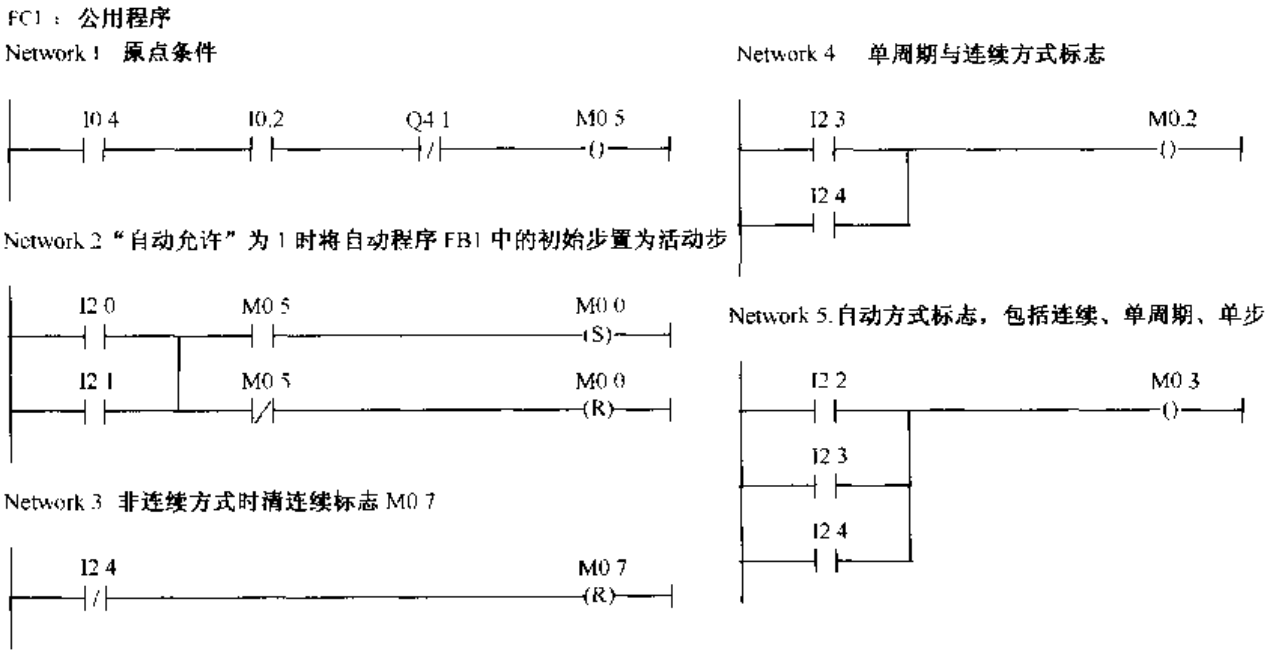


图5-62 公用程序

在公用程序中将控制单步、单周期和连续这3种自动方式的I2.2、I2.3和I2.4的常开触点并联,来控制符号名为"自动方式"的M0.3,用M0.3作为FB1的使能输入(EN)信号,即只在这3种工作方式调用FB1。

在公用程序中将控制单周期和连续这两种自动方式的I2.3和I2.4的常开触点并联,来控制符号名为"单周连续"的M0.2,它用来为FB1提供输入信号SW_AUTO(自动工作方式)。

在单步工作方式,符号名为"单步"的I2.2为1,它的常开触点给FB1提供输入信号SW_TAP(单步工作方式),起动按钮(I2.6)为FB1提供输入信号T_PUSH。在单步方式,即使转换条件满足,也必须按一下起动按钮I2.6,才能转换到下一步去。

【确认故障】按钮(I1.3)给FB1提供输入信号ACK_ET,某步出现了监控事件,例如该步处于活动状态的时间超过了设定值,该步变为红色。如果转换条件满足,需要按一下确认故障按钮,才能转换到下一步去。

5. 自动程序

自动程序FB1是用S7 Graph语言编写的,前面已经介绍了怎样用FB1的输入参数来区分单步方式和非单步(单周期和连续)方式。与5.5节一样,单周期和连续方式是用M0.7(连续标志)和顺序控制器中的选择序列来区分的。M0.7的控制电路放在FB1的顺序控制器之前的永久性指令中(见图5-63),每次扫描都要执行永久性指令。

图5-64是FB1中的顺序控制器,在步S27之后生成选择序列的分支时,首先用鼠标选中步S27,然后点击"Sequencer"工具条上的【Opening an Alternative Branch】打开【选择序列的分

支】按钮。生成选择序列的分支后，分别对两条支路上的转换条件编程。最后在两个转换上生成跳步(Jump)，分别跳到步 S1 和步 S20。S1 和 S20 之前标有 T9 和 T10 的箭头是自动生成的，它们用来表示选择序列的合并。

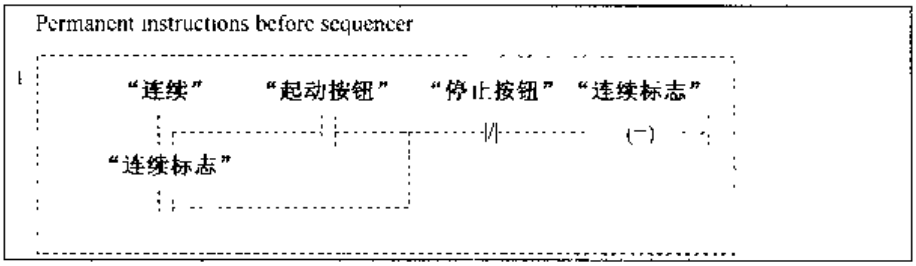


图 5-63　顺序控制器之前的永久性指令

在单周期工作方式，连续标志 M0.7 处于 0 状态。当机械手在最后一步 S27 返回最左边时，左限位开关 I0.4 为 1 状态，因为连续标志的常闭触点闭合，转换条件 T9 满足，使系统返回并停留在初始步 S1。按一次起动按钮，系统只工作一个从步 S0 到步 S27 的工作周期。

连续工作方式时 I2.4 为 1 状态，在初始状态按下起动按钮 I2.6，"连续标志"M0.7 的线圈"通电"并自保持，步 S20 变为 1 状态，机械手下降。以后的工作过程与单周期工作方式相同。机械手在步 S27 返回最左边时，左限位开关 I0.4 为 1 状态，因为这时连续标志 M0.7 也为 1 状态，它们的常开触点均闭合，转换条件 T10 满足，系统返回步 S20，以后将这样反复连续地工作下去。

按下【停止】按钮 I2.7 以后，连续标志 M0.7 变为 0 状态，但是系统不会立即停止工作。在完成当前工作周期的全部操作后，小车在步 S27 返回最左边，左限位开关 I0.4 为 1 状态，此时连续标志 M0.7 的常闭触点闭合，转换条件 T9 满足，系统才会返回并停留在初始步 S1。

在单步工作方式，转换条件满足时，操作人员必须按一下【起动】按钮 I2.6，才会转换到下一步。以下行步 S20 为例，下限位开关 I0.1 为 1 时，不会马上转换到下一步，但是控制下降的电磁阀 Q4.0 应变为 0 状态。为此在编程时用鼠标双击步 S20，进入单步显示模式，用 I0.1 的常闭触点控制步 S20 的中间标有大写字母"C"的互锁线圈。同时还应将控制该步的动作 Q4.0 的命令 N 改为 NC，即步 S20 为活动步和互锁条件同时满足(I0.1 = 0，I0.1 的常闭触点闭合)时，Q4.0 才为 1 状态。因此在下限位开关动作，I0.1 = 1，互锁条件步不满足时，该步变为红色，Q4.0 变为 0 状态。对其余各步的动作，均应作相同的处理，并将延时命令"D"改为"DC"，才能保证在单步工作模式时，转换条件满足后及时停止该步的动作。图 5-64 中除初始步外，各步的左上角均标有"C"，表示这些步均有互锁功能。

5.6.8　S7 Graph 功能块的参数优化设置

生成 S7 Graph FB 时，在 S7 Graph 编辑器中执行菜单命令"Options→Application Settings"，在打开的对话框中的"Compile / Save"选项卡(见图 5-58)的"Executability"中，有两种方式可供选择：

(1) Full code

每一个 S7 Graph FB 中都包含了执行 S7 Graph FB 所需的全部代码。如果有多个 S7 Graph FB，对存储器容量的需求将会显著增加。

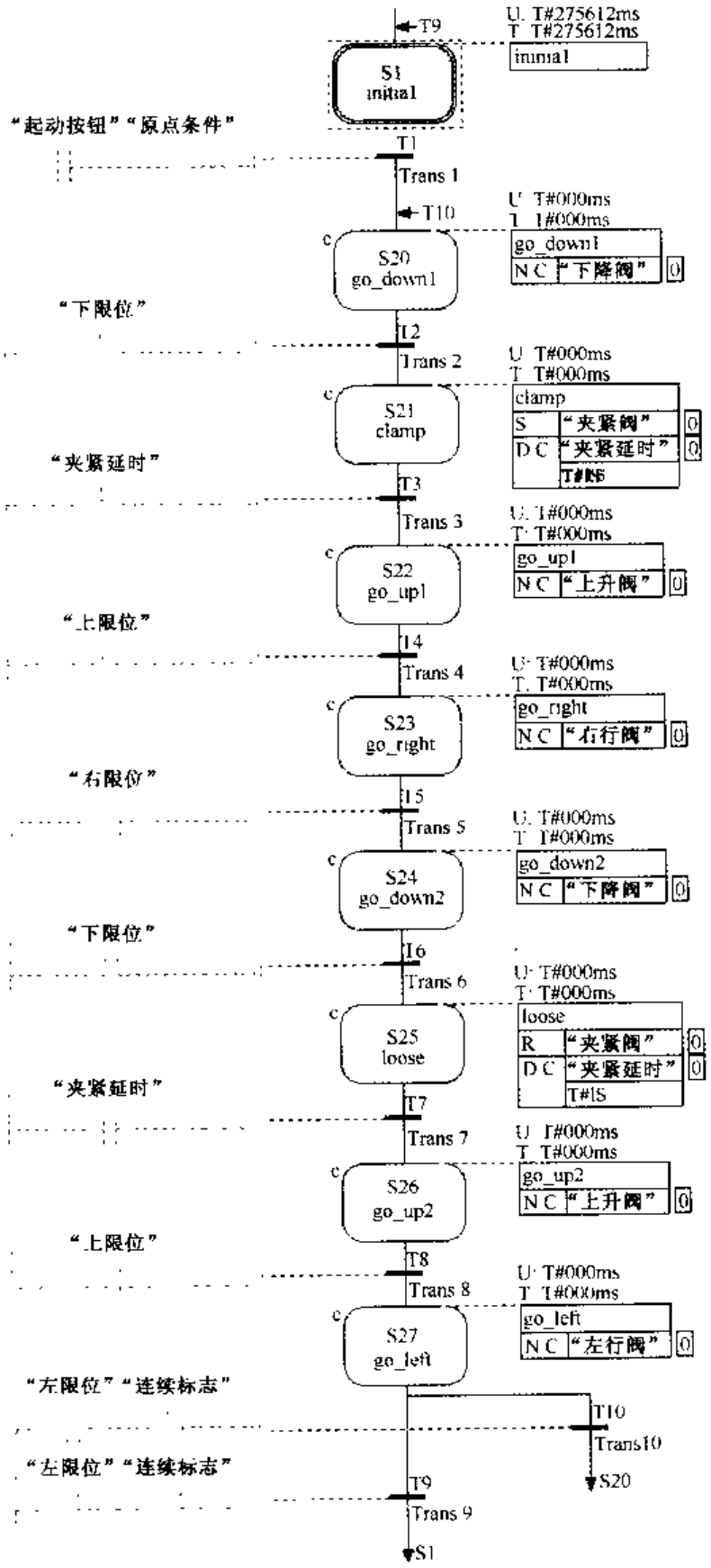


图 5-64　自动控制程序中的顺序控制器

(2) Standard FC required

该选项可以减小对存储器的需求，标准功能 FC 70、FC 71、FC 72 或 FC 73 用于所有的 S7 Graph FB，它们包含所有的 S7 Graph FB 使用的主要的代码，如果选择这一选项，有关的 FC 将被自动地复制到项目中，用这个方法生成的 FB 较小。

需要在输入框"FC number"中输入标准 FC 的编号。FC 72 是默认的设置，它要求 CPU 能处理大于 8 KB 的块。

FC 70 和 FC 71 小于 8 KB，可供较小的 CPU 使用。FC 70 只能用于可执行 SFC17/18 的诊断功能的 CPU，如果 CPU 没有该功能，应使用 FC71。要想知道 CPU 是否有 SFC17/18，与 CPU 建立起通信联系后，执行菜单命令"PLC"→"Display Accessible Nodes"，在块文件夹中出现该 CPU 中的系统功能块。

FC 73 需要的存储容量小于 8 KB，可以在所有的 CPU 中使用，它可以显著地减少 S7 Graph FB 的存储器需求。应在"Compile / Save"选项卡的"Interface Description"(接口描述)中选择"Memory minimized"(存储器最小化)。

S7 Graph FB 用预先设置好的默认的显示模式打开，执行菜单命令"Options"→"Application Settings"，在出现的对话框的"General"选项卡的"New Window View"区中，可选择打开 FB 时默认的显示模式和显示的内容(见图 5-65)。

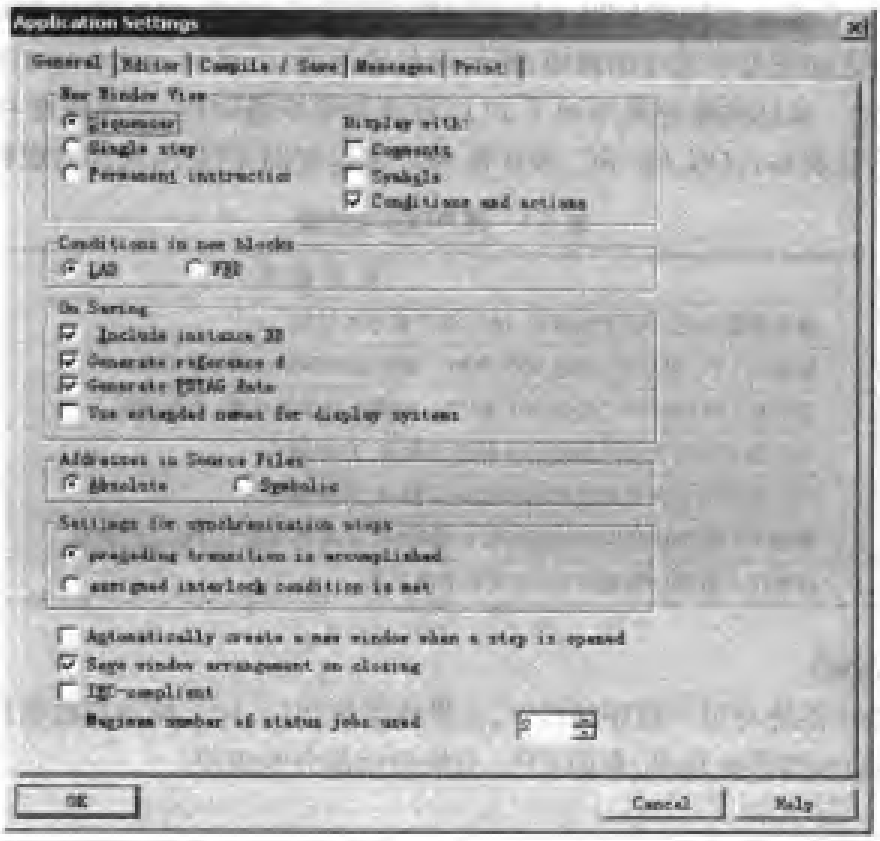


图 5-65 参数设置

第 6 章　S7-300/400 的用户程序结构

6.1　用户程序的基本结构

6.1.1　用户程序中的块

PLC 中的程序分为操作系统和用户程序，操作系统用来实现与特定的控制任务无关的功能，处理 PLC 的起动、刷新输入/输出过程映像表、调用用户程序、处理中断和错误、管理存储区和处理通信等。

用户程序由用户在 STEP 7 中生成，然后将它下载到 CPU。用户程序包含处理用户特定的自动化任务所需要的所有功能，例如指定 CPU 暖起动或热起动的条件、处理过程数据、指定对中断的响应和处理程序正常运行中的干扰等。

STEP 7 将用户编写的程序和程序所需的数据放置在块中，使单个的程序部件标准化。通过在块内或块之间类似子程序的调用，使用户程序结构化，可以简化程序组织，使程序易于修改、查错和调试。块结构显著地增加了 PLC 程序的组织透明性、可理解性和易维护性。各种块的简要说明见表 6-1，OB、FB、FC、SFB 和 SFC 都包含部分程序，统称为逻辑块。

表 6-1　用户程序中的块

块	简 要 描 述
组织块(OB)	操作系统与用户程序的接口，决定用户程序的结构
系统功能块(SFB)	集成在 CPU 模块中，通过 SFB 调用一些重要的系统功能，有存储区
系统功能(SFC)	集成在 CPU 模块中，通过 SFC 调用一些重要的系统功能，无存储区
功能块(FB)	用户编写的包含经常使用的功能的子程序，有存储区
功能(FC)	用户编写的包含经常使用的功能的子程序，无存储区
背景数据块(DI)	调用 FB 和 SFB 时用于传递参数的数据块，在编译过程中自动生成数据
共享数据块(DB)	存储用户数据的数据区域，供所有的块共享

1　组织块(OB)

组织块是操作系统与用户程序的接口，由操作系统调用，用于控制扫描循环和中断程序的执行、PLC 的起动和错误处理等，有的 CPU 只能使用部分组织块。

(1) OB1

OB1 的功能已在第 1 章中作了介绍，OB1 用于循环处理，是用户程序中的主程序。操作系统在每一次循环中调用一次组织块 OB1。一个循环周期分为输入、程序执行、输出和其他任务，例如下载、删除块，接收和发送全局数据等。

(2) 事件中断处理

如果出现一个中断事件，例如时间日期中断、硬件中断和错误处理中断等，当前正在执行

的块在当前语句执行完后被停止执行(被中断),操作系统将会调用一个分配给该事件的组织块。该组织块执行完后,被中断的块将从断点处继续执行。

这意味着部分用户程序可以不必在每次循环中处理,而是在需要时才被及时地处理。用户程序可以分解为分布在不同的组织块中的"子程序"。如果用户程序是对一个重要事件的响应,并且这个事件出现的次数相对较少,例如液位达到了最大上限,处理中断事件的程序放在该事件驱动的 OB 中。

(3) 中断的优先级

OB 按触发事件分成几个级别,这些级别有不同的优先级,高优先级的 OB 可以中断低优先级的 OB。当 OB 启动时,提供触发它的初始化启动事件的详细信息,这些信息可以在用户程序中使用。

2. 临时局域数据

生成逻辑块(OB、FC、FB)时可以声明临时局域数据。这些数据是临时的,退出逻辑块时不保留临时局域数据。它们又是一些局域(Local,或称局部)数据,只能在生成它们的逻辑块内使用。CPU 按优先级划分局域数据区,同一优先级的块共用一片局域数据区。可以用 STEP 7 改变 S7-400 每个优先级的局域数据的数量。

除了临时局域数据外,所有的逻辑块都可以使用共享数据块中的共享数据。

3. 功能(FC)

功能是用户编写的没有固定的存储区的块,其临时变量存储在局域数据堆栈中,功能执行结束后,这些数据就丢失了。可以用共享数据区来存储那些在功能执行结束后需要保存的数据,不能为功能的局域数据分配初始值。

调用功能和功能块时用实参(实际参数)代替形参(形式参数),例如将实参"I3.6"赋值给形参"Start"。形参是实参在逻辑块中的名称,功能不需要背景数据块。功能和功能块用输入(IN)、输出(OUT)和输入/输出(IN_OUT)参数做指针,指向调用它的逻辑块提供的实参。功能被调用后,可以为调用它的块提供一个数据类型为 RETURN 的返回值。

4. 功能块(FB)

功能块是用户编写的有自己的存储区(背景数据块)的块,每次调用功能块时需要提供各种类型的数据给功能块,功能块也要返回变量给调用它的块。这些数据以静态变量(STAT)的形式存放在指定的背景数据块(DI)中,临时变量存储在局域数据堆栈中。功能块执行完后,背景数据块中的数据不会丢失,但是不会保存局域数据堆栈中的数据。

在编写调用 FB 或系统功能块(SFB)的程序时,必须指定 DI 的编号,调用时 DI 被自动打开。在编译 FB 或 SFB 时自动生成背景数据块中的数据。可以在用户程序中或通过 HMI(人机接口)访问这些背景数据。

一个功能块可以有多个背景数据块,使功能块用于不同的被控对象。

可以在 FB 的变量声明表中给形参赋初值,它们被自动写入相应的背景数据块中。在调用块时,CPU 将实参分配给形参的值存储在 DI 中。如果调用块时没有提供实参,将使用上一次存储在背景数据块中的参数。

5. 数据块

数据块(DB)是用于存放执行用户程序时所需的变量数据的数据区。与逻辑块不同,在数据块中没有 STEP 7 的指令,STEP 7 按数据生成的顺序自动地为数据块中的变量分配地址。

数据块分为共享数据块和背景数据块。数据块的最大允许容量与 CPU 的型号有关。

数据块中基本的数据类型有 BOOL(二进制位)、REAL(实数或浮点数)和 INTEGER(整数,简称为 INT)等。结构化数据类型(数组和结构)由基本数据类型组成。可以用符号表中定义的符号来代替数据块中的数据的地址,这样更便于程序的编写和阅读。

(1) 共享数据块(Share Block)

共享数据块存储的是全局数据,所有的 FB、FC 或 OB(统称为逻辑块)都可以从共享数据块中读取数据,或将数据写入共享数据块。CPU 可以同时打开一个共享数据块和一个背景数据块。如果某个逻辑块被调用,它可以使用它的临时局域数据区(即 L 堆栈)。逻辑块执行结束后,其局域数据区中的数据丢失,但是共享数据块中的数据不会被删除。

(2) 背景数据块(Instance Data Block)

背景数据块中的数据是自动生成的,它们是功能块的变量声明表中的数据(不包括临时变量 TEMP)。背景数据块用于传递参数,FB 的实参和静态数据存储在背景数据块中。调用功能块时,应同时指定背景数据块的编号或符号,背景数据块只能被指定的功能块访问。

应首先生成功能块,然后生成它的背景数据块。在生成背景数据块时,应指明它的类型为背景数据块(Instance),并指明它的功能块的编号,例如 FB2。

背景数据块的功能块被执行完后,背景数据块中存储的数据不会丢失。

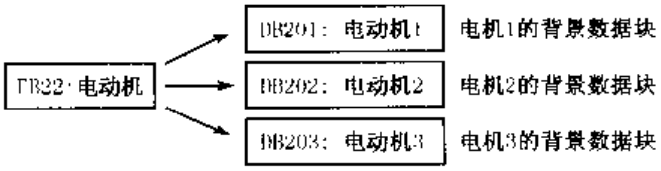


图 6-1 用于不同对象的背景数据块

在调用功能块时使用不同的背景数据块,可以控制多个同类的对象(见图 6-1)。

6. 系统功能块(SFB)

系统功能块和系统功能是为用户提供的已经编好程序的块,可以在用户程序中调用这些块,但是用户不能修改它们。它们作为操作系统的一部分,不占用程序空间。SFB 有存储功能,其变量保存在指定给它的背景数据块中。

7. 系统功能(SFC)

系统功能是集成在 S7 CPU 的操作系统中预先编好程序的逻辑块,例如时间功能和块传送功能等。SFC 属于操作系统的一部分,可以在用户程序中调用。与 SFB 相比,SFC 没有存储功能。S7 CPU 提供以下的 SFC:复制及块功能,检查程序,处理时钟和运行时间计数器,数据传送,在多 CPU 模式的 CPU 之间传送事件,处理日期时间中断和延时中断,处理同步错误、中断错误和异步错误,有关静态和动态系统数据的信息,过程映像刷新和位域处理,模块寻址,分布式 I/O,全局数据通信,非组态连接的通信,生成与块相关的信息等。

8. 系统数据块(SDB)

系统数据块是由 STEP 7 产生的程序存储区,包含系统组态数据,例如硬件模块参数和通信连接参数等用于 CPU 操作系统的数据。

9. 块的调用

可以用 CALL 指令调用没有参数的 FC 和有参数的 FC 和 FB,用 CU(无条件调用)和 CC(RLO = 1 时调用)指令调用没有参数的 FC 和 FB。用 CALL 指令调用 FB 和 SFB 时必须指定背景数据块,静态变量和临时变量不能出现在调用指令中。

6.1.2 用户程序使用的堆栈

堆栈(见图 6-2)是 CPU 中的一块特殊的存储区,它采用"先入后出"的规则存入和取出数据。堆栈中最上面的存储单元称为栈顶,要保存的数据从栈顶"压入"堆栈时,堆栈中原有的数据依次向下移动一个位置,最下面的存储单元中的数据丢失。在取出栈顶的数据后,堆栈中所有的数据依次向上移动一个位置。堆栈的这种"先入后出"的存取规则刚好满足块的调用(包括中断处理时块的调用)的要求,因此在计算机的程序设计中得到了广泛的应用。下面介绍 STEP 7 中 3 种不同的堆栈。

1. 局域数据堆栈(L)

局域数据堆栈用来存储块的局域数据区的临时变量、组织块的启动信息、块传递参数的信息和梯形图程序的中间结果。局域数据可以按位、字节、字和双字来存取,例如 L 0.0,LB9,LW4 和 LD52。

各逻辑块均有自己的局域变量表,局域变量仅在它被创建的逻辑块中有效。对组织块编程时,可以声明临时变量(TEMP)。临时变量仅在块被执行的时候使用,块执行完后将被别的数据覆盖。

在首次访问局域数据堆栈时,应对局域数据初始化。每个组织块需要 20 B 的局域数据来存储它的起动信息。

CPU 分配给当前正在处理的块的临时变量(即局域数据)的存储器容量是有限的,这一存储区(即局域堆栈)的大小与 CPU 的型号有关。CPU 给每一优先级分配了相同数量的局域数据区,这样可以保证不同优先级的 OB 都有它们可以使用的局域数据空间。

图 6-3 中的 FB1 调用功能 FC2,FC2 的执行被组织块 OB81 中断,图中给出了局域数据堆栈中局域数据的存放情况。

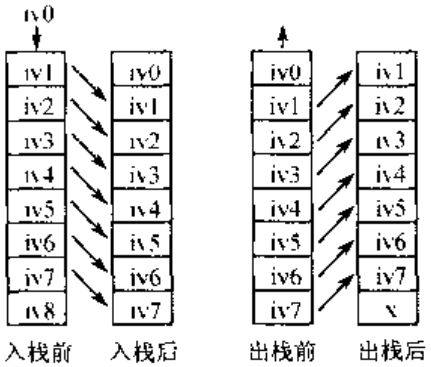


图 6-2 堆栈操作

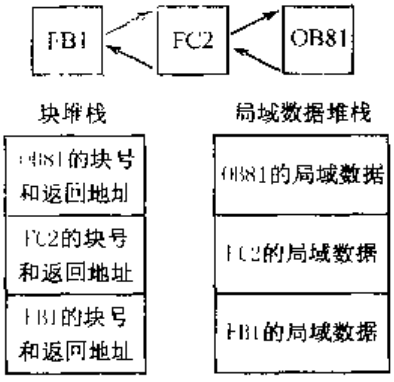


图 6-3 块堆栈与局域数据堆栈

在局域数据堆栈中,并非所有的优先级都需要相同数量的存储区。通过在 STEP 7 中设置参数,可以给 S7-400 CPU 和 CPU 318 的每一优先级指定不同大小的局域数据区。其余的 S7-300 CPU 每一优先级的局域数据区的大小是固定的。

2. 块堆栈(B 堆栈)

如果一个块的处理因为调用另外一个块,或者被更高优先级的块中止,或者被对错误的服务中止,CPU 将在块堆栈中存储以下信息:

- 被中断的块的类型(OB、FB、FC、SFB、SFC)、编号和返回地址;

- 从 DB 和 DI 寄存器中获得的块被中断时打开的共享数据块和背景数据块的编号；
- 局域数据堆栈的指针。

利用这些数据，可以在中断它的任务处理完后恢复被中断的块的处理。在多重调用时，堆栈可以保存参与嵌套调用的几个块的信息。

CPU 处于 STOP 模式时，可以在 STEP 7 中显示 B 堆栈中保存的在进入 STOP 模式时没有处理完的所有的块，在 B 堆栈中，块按照它们被处理的顺序排列(见图 6-3)。

每个中断优先级对应的块堆栈中可以储存的数据的字节数与 CPU 的型号有关。

3. 中断堆栈(I 堆栈)

如果程序的执行被优先级更高的 OB 中断，操作系统将保存下述寄存器的内容：当前的累加器和地址寄存器的内容、数据块寄存器 DB 和 DI 的内容、局域数据的指针、状态字、MCR(主控继电器)寄存器和 B 堆栈的指针。

新的 OB 执行完后，操作系统从中断堆栈中读取信息，从被中断的块被中断的地方开始继续执行程序。

CPU 在 STOP 模式时，可以在 STEP 7 中显示 I 堆栈中保存的数据，用户可以由此找出使 CPU 进入 STOP 模式的原因。

6.1.3 线性化编程与结构化编程

STEP 7 有 3 种设计程序的方法，即线性化编程、模块化编程和结构化编程。

1. 线性化编程

线性化编程类似于硬件继电器控制电路，整个用户程序放在循环控制组织块 OB1(主程序)中，循环扫描时不断地依次执行 OB1 中的全部指令。这种方式的程序结构简单，不涉及功能块、功能、数据块、局部变量和中断等比较复杂的概念，容易入门。建议只是在为 S7-300 编写简单的程序时使用。

由于所有的指令都在一个块中，即使程序中的某些部分在大多数时候并不需要执行，每个扫描周期都要执行所有的指令，因此没有有效地利用 CPU。此外如果要求多次执行相同或类似的操作，需要重复编写程序。

2. 模块化编程

程序被分为不同的逻辑块，每个块包含完成某些任务的逻辑指令。组织块 OB1(即主程序)中的指令决定在什么情况下调用哪一个块，功能和功能块(即子程序)用来完成不同的过程任务。被调用的块执行完后，返回到 OB1 中程序块的调用点，继续执行 OB1。

模块化编程的程序被划分为若干个块，易于几个人同时对一个项目编程。由于只是在需要时才调用有关的程序块，提高了 CPU 的利用效率。

3. 结构化编程

结构化编程将复杂的自动化任务分解为能够反映过程的工艺、功能或可以反复使用的小任务，这些任务由相应的程序块(或称逻辑块)来表示，程序运行时所需的大量数据和变量存储在数据块中。某些程序块可以用来实现相同或相似的功能。这些程序块是相对独立的，它们被 OB1 或别的程序块调用。

在块调用中，调用者可以是各种逻辑块，包括用户编写的组织块(OB)、FB、FC，和系统提供的 SFB(系统功能块)与系统功能(SFC)，被调用的块是 OB 之外的逻辑块。调用功能块时需

要为它指定一个背景数据块，后者随功能块的调用而打开、在调用结束时自动关闭。

在给功能块编程时使用的是“形参”(形式参数)，调用它时需要将“实参”(实际参数)赋值给形参。在一个项目中，可以多次调用同一个块，例如在调用控制发动机的块时，将不同的实参赋值给形参，就可以实现对类似但是不完全相同的被控对象(例如汽油机和柴油机)的控制。

块调用即子程序调用，块可以嵌套调用，即被调用的块又可以调用别的块。允许嵌套调用的层数(嵌套深度)与 CPU 的型号有关。

块嵌套调用的层数还受到 L 堆栈大小的限制。每个 OB 需要至少 20 B 的 L 内存。当块 A 调用块 B 时，块 A 的临时变量将压入 L 堆栈。

在图 6-4 中，OB1 调用 FB1，FB1 调用 FC1，应按下面的顺序创建块：FC1→FB1 及其背景数据块→OB1，即编程时被调用的块应该是已经存在的。

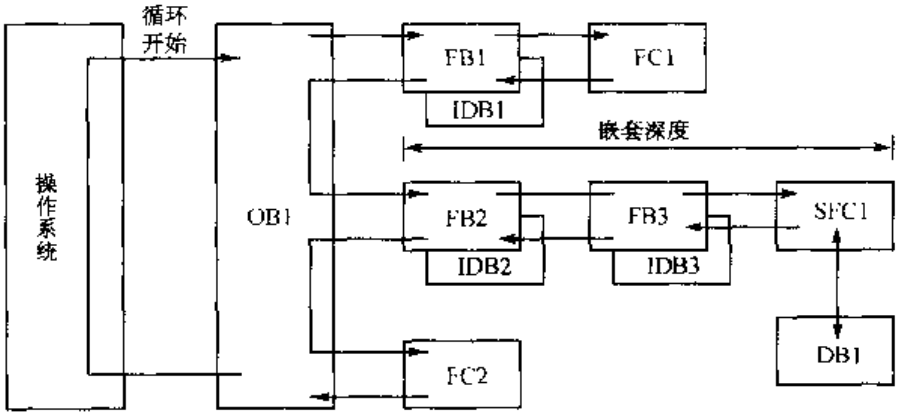


图 6-4　块调用的分层结构

6.2　功能块与功能的生成与调用

6.2.1　发动机控制系统的用户程序结构

下面以发动机控制系统的用户程序为例，介绍生成和调用功能块和功能的方法。

1. 项目的创建

在计算机的桌面上双击“SIMATIC Manager”图标，在弹出的新项目向导中点击【NEXT】，依次选择 CPU 的型号和 MPI 站地址、需要编程的组织块和使用的编程语言等，最后设置项目的名称为“发动机控制”。

2. 用户程序结构

图 6-5 中的组织块 OB1 是主程序，用一个名为“发动机控制”的功能块 FB1 来分别控制汽油机和柴油机(见图 6-5)，控制参数在背景数据块 DB1 和 DB2 中。控制汽油机时调用 FB1 和名为“汽油机数据”的背景数据块 DB1，控制柴油机时调用 FB1 和名为“柴油机数据”的背景数据块 DB2。此外控制汽油机和柴油机时还用不同的实参分别

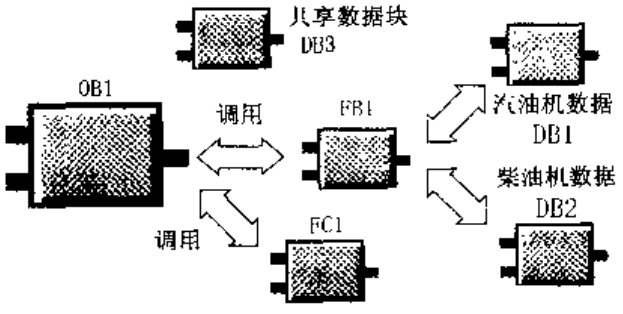


图 6-5　程序结构

调用名为"风扇控制"的功能 FC1。图 6-6 是程序设计好后 SIMATIC 管理器中的块。

图 6-6 SIMATIC 管理器

6.2.2 符号表与变量声明表

1. 符号表

为了使程序易于理解，可以给变量指定符号。图 6-7 是发动机控制项目的符号表，符号表中定义的变量是全局变量，可供所有的逻辑块使用。

	Status	Symbol	Address	Data type	Comment
1		汽油机数据	DB 1	FB 1	背景数据块
2		柴油机数据	DB 2	FB 1	背景数据块
3		共享数据	DB 3	DB 3	共享数据块
4		发动机控制	FB 1	FB 1	功能块
5		风扇控制	FC 1	FC 1	功能
6		自动	I 0.5	BOOL	自动按钮
7		手动	I 0.6	BOOL	手动按钮
8		起动汽油机	I 1.0	BOOL	控制按钮
9		关闭汽油机	I 1.1	BOOL	控制按钮
10		汽油机故障	I 1.2	BOOL	故障输入
11		起动柴油机	I 1.4	BOOL	控制按钮
12		关闭柴油机	I 1.5	BOOL	控制按钮
13		柴油机故障	I 1.6	BOOL	故障输入
14		汽油机转速	MW 2	INT	实际转速
15		柴油机转速	MW 4	INT	实际转速
16		主程序	OB 1	OB 1	用户主程序
17		自动模式	Q 4.2	BOOL	指示灯
18		汽油机运行	Q 5.0	BOOL	控制汽油机运行的输出
19		汽油机到达设置转速	Q 5.1	BOOL	指示灯
20		汽油机风扇运行	Q 5.2	BOOL	控制汽油机风扇的输出
21		柴油机运行	Q 5.4	BOOL	控制柴油机运行的输出
22		柴油机到达设置转速	Q 5.5	BOOL	指示灯
23		柴油机风扇运行	Q 5.6	BOOL	控制柴油机风扇的输出
24		汽油机风扇延时	T 1	TIMER	脉冲延时定时器
25		柴油机风扇延时	T 2	TIMER	脉冲延时定时器
26					

图 6-7 符号表

2. 变量声明表

图 6-8 中的梯形图编辑器的右上半部分是变量声明表，右下半部是程序指令部分，左边是指令列表。用户在变量声明表中声明本块中专用的变量，即局域变量，包括块的形参(形式参数)和参数的属性，局域变量只是在它所在的块中有效。声明后在局域数据堆栈中为临时变量(TEMP)保存有效的存储空间。对于功能块，还要为配合使用的背景数据块的静态变量

(STAT)保留空间。通过设置 IN(输入)、OUT(输出)和 IN _ OUT(输入/输出)类型变量,声明块调用时的软件接口(即形参)。用户在功能块中声明变量时,除了临时变量外,它们将自动出现在功能块对应的背景数据块中。

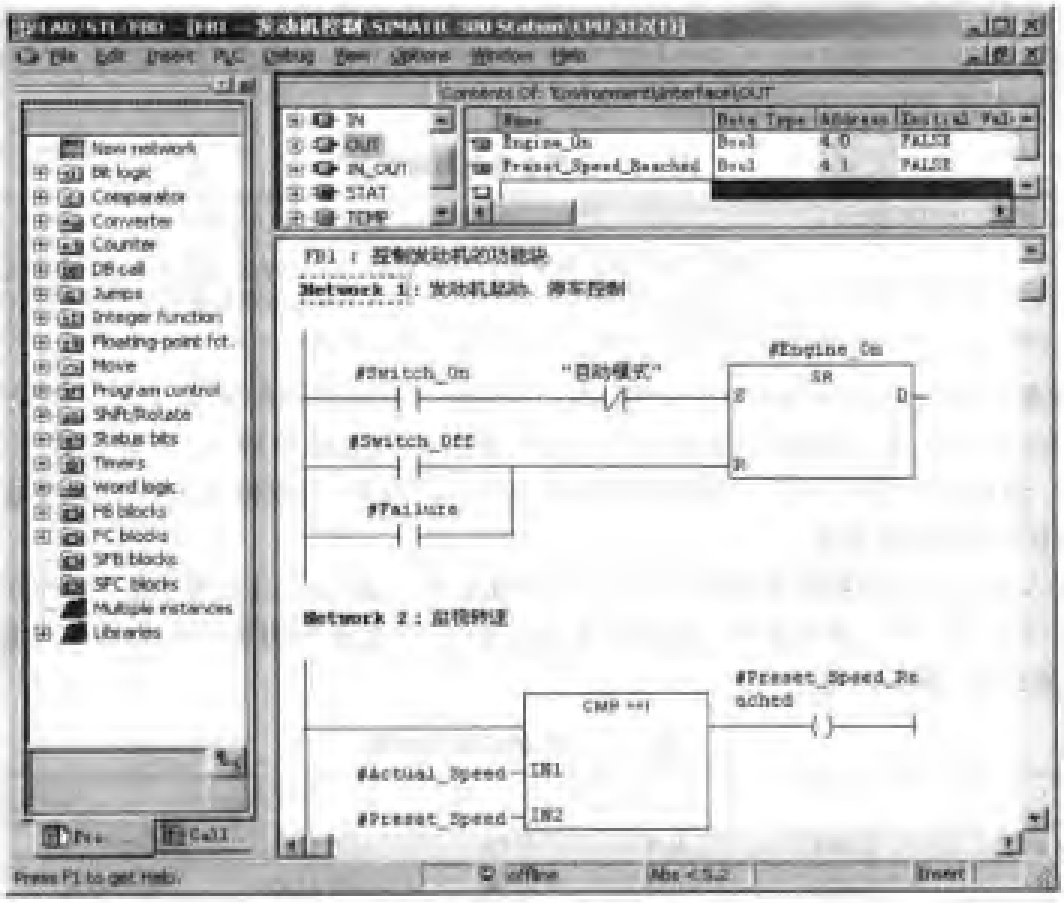


图 6-8 梯形图编辑器

如果在块中只使用局域变量,不使用绝对地址或全局符号,可以将块移植到别的项目。

块中的局域变量名必须以字母开始,只能由英语字母、数字和下划线组成,不能使用汉字,但是在符号表中定义的共享数据的符号名可以使用其他字符(包括汉字)。在程序中,操作系统在局域变量前面自动加上"#"号,共享变量名被自动加上双引号,共享变量可以在整个用户程序中使用。

在图 6-8 中,变量声明表的左边给出了该表的总体结构,点击某一变量类型,例如"OUT",在表的右边将显示出该类型局域变量的详细情况。

将图 6-8 中变量声明表与程序指令部分的水平分隔条拉至程序编辑器视窗的顶部,不再显示变量声明表,但它仍然存在。将分隔条下拉,将再次显示变量声明表。

3. 局域变量的类型

由图 6-8 可知,功能块的局域变量分为 5 种类型:

(1) IN(输入变量):由调用它的块提供的输入参数。

(2) OUT(输出变量):返回给调用它的块的输出参数。

(3) IN _ OUT(输入_输出变量):初值由调用它的块提供,被子程序修改后返回给调用它的块。

(4) TEMP (临时变量):暂时保存在局域数据区中的变量。只是在执行块时使用临时变量,执行完后,不再保存临时变量的数值。在 OB1 中,局域变量表只包含 TEMP 变量。

(5) STAT(静态变量):在功能块的背景数据块中使用。关闭功能块后,其静态数据保持不变。功能(FC)没有静态变量。

在变量声明表中赋值时,不需要指定存储器地址;根据各变量的数据类型,程序编辑器自动地为所有局域变量指定存储器地址。见表 6-2。

表 6-2 全局变量与局域变量

全局变量(在整个程序中使用)	局域变量(只能在一个块中使用)	
PII/PIQ、I/O、M/T/C、DB 区	临时变量:对应的块执行完后被删除,临时存储在 L 堆栈中,可以在 FB、FC 和 OB 中使用	静态变量:对应的块执行完后永久保留在背景数据块中,只能用于 FB

在变量声明表中选择 ARRAY(数组)时,用鼠标点击相应行的地址单元。如果想选中一个结构(Structure),用鼠标选中结构的第一行或最后一行的地址单元,即有关键字 STRUCT 或 END_ STRUCT 的那一行。若要选择结构中的某一参数,用鼠标点击该行的地址单元。

4. FB1 中的局域变量

表 6-3 列出了发动机控制例程中 FB1 的局域变量。表中 Bool 变量(数字量)的初值(Initial Value)FALSE 即二进制数 0。预置转速是固定值,在变量声明表中作为静态参数(STAT)来存储,被称为"静态局域变量"。

表 6-3 FB1 的变量声明表

Name	Data Type	Address	Declare	Initial Value	Comment
Switch_On	Bool	0.0	IN	FALSE	起动按钮
Switch_Off	Bool	0.1	IN	FALSE	停车按钮
Failure	Bool	0.2	IN	FALSE	故障信号
Actual_Speed	Int	2.0	IN	0	实际转速
Engine_On	Bool	4.0	OUT	FALSE	控制发动机的输出信号
Preset_Speed_Reached	Bool	4.1	OUT	FALSE	达到预置转速
Preset_Speed	Int	6.0	STAT	1500	预置转速

5. 程序库

程序库用来存放可以多次使用的程序部件,可以从已有的项目中将它们复制到程序库,也可以在程序库中直接生成程序部件。在管理器中用菜单命令"File"→"New"打开"New Project"对话框,在"Libraries"选项卡可以生成新的程序库。用菜单命令"Option"→"Customize"打开"Customize"窗口,用"General"选项卡中的"Storage location for libraries"可以设置新库存放在计算机中的目录。用程序编辑器中的菜单命令"View"→"Overviews"可以显示或关闭图 6-8 右边的指令目录和程序库(Libraries)。

STEP 7 标准软件包提供下列的标准程序库:

(1) 系统功能块(SFB)、系统功能(SFC)和标准组织块(OB);

(2) S5-S7 转换块和 TI-S7 转换块:用于转换 STEP 5 程序或 TI 程序;

(3) IEC 功能块:处理时间和日期信息、比较操作、字符串处理与选择最大值和最小值等;

(4) PID 控制块与通信块:用于 PID 控制和通信处理器(CP);

(5) 其他功能块(Miscellaneous Blocks),例如用于时间标记和实时钟同步等的块。

用户安装可选软件包后,还会增加其他的程序库。例如安装 S7 Graph 后会自动增加 S7 Graph 库。

6.2.3 功能块与功能

1. 功能块 FB1 的程序

图 6-8 的下半部分是 FB1 的梯形图程序,SR 指令块用来控制发动机的运行,输入变量 Switch _ on 和 Switch _ off 分别是起动命令和停车命令。Failure(故障)信号在无故障时为 0,有故障时为 1。功能块的输出信号 Engine _ On 为 1 时发动机运行,为 0 时发动机停车。

FB1 用比较指令来监视转速,检查实际速度是否大于等于预置转速。如果满足条件,输出信号 # Preset _ Speed _ Reached(达到预置速度)被置位为 1。

2. 功能的生成与编辑

如果控制功能不需要保存它自己的数据,可以用功能 FC 来编程。与功能块 FB 相比较,FC 不需要配套的背景数据块。

在功能的变量声明表中可以使用的参数类型有 IN、OUT、IN _ OUT、TEMP 和 RETURN(返回参数),功能不能使用静态(STAT)局域数据。

在管理器中打开 Block 文件夹,用鼠标右键点击右边的窗口,在弹出的菜单中选择"Insert New Object→Function"(插入一个功能)。表 6-4 是功能 1 中使用的变量,在变量声明表中不能用汉字作变量的名称。

表 6-4 FC1 的变量声明表

Name	DataType	Declare	Comment
Engine _ On	Bool	IN	输入信号,发动机运行
Timer _ Function	Timer	IN	停机延时的定时器功能

功能 FC1 用来控制发动机的风扇,要求在起动发动机的同时起动风扇,发动机停车后,风扇继续运行 4 s 后关断。

在 FC1 中,使用了延时断开定时器(S _ OFFDT)。在功能的变量声明表中定义了输入变量和输出变量,调用 FC1 时将延时断开定时器作为功能的输入变量,数据类型为 Timer,FC1 用于不同的发动机时指定不同的定时器。图 6-9 是对应的梯形图程序,下面是 FC1 中的语句表程序:

```
FC1    用于风扇控制的功能
Network 1: 风扇控制
```

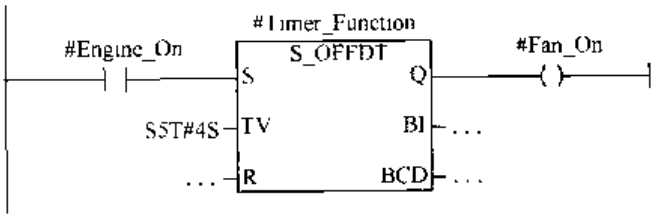


图 6-9 梯形图

```
A       #Engine_On
L       S5T#4S
SF      #Timer_Function
A       #Timer_Function
=       #Fan_On
```

6.2.4 功能块与功能的调用

组织块 OB1 是循环执行的主程序，生成项目时系统自动生成空的 OB1。在管理器中双击 OB1 图标后进入编辑器窗口，可以用“View”菜单命令选择编程语言。

在发动机控制程序中，OB1 用来实现自动/手动工作模式的切换，通过两次调用 FB1 和 FC1 实现对汽油机和柴油机的控制。图 6-10 只给出了控制汽油机的程序，控制柴油机的程序与之相似。

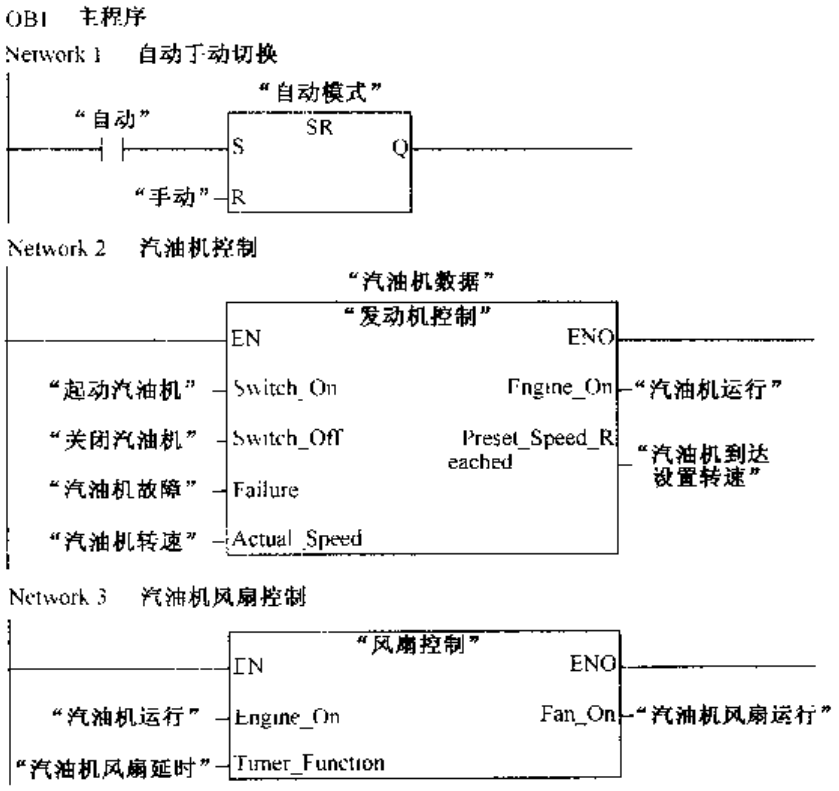


图 6-10 主程序 BO1

通过置位/复位指令 SR，用符号名分别为“自动”和“手动”的按钮来控制符号名为“自动模式”的输出量 Q 4.2。符号名为“自动”和“手动”的变量不是某一发动机的属性，它用于整个程序，因此它不是块的参数，它是在共享符号表中定义的。

1. 功能块的调用

块调用分为条件调用和无条件调用。用梯形图调用块时，块的 EN(Enable，使能)输入端有能流流入时执行块，反之则不执行。条件调用时 EN 端受到触点电路的控制。块被正确执行时 ENO(Enable Output，使能输出端)为 1，反之为 0。

调用功能块之前，应为它生成一个背景数据块，调用时应指定背景数据块的名称。生成背景数据块时应选择数据块的类型为背景数据块，并设置调用它的功能块的名称。图 6-10 中的

“汽油机数据”(DB1)是功能块“发动机控制”(FB1)的背景数据块。

调用功能块时应将实参赋值给形参,例如将符号名为“起动汽油机”的实参赋值给形参“Switch _ On”,实参可以是绝对地址或符号地址。如果调用时没有给形参赋以实参,功能块就调用背景数据块中形参的数值。该数值可能是在功能块的变量声明表中设置的形参的初值,也可能是上一次调用时储存在背景数据块中的数值。

在 OB1 中,用 CALL 指令调用功能块 FB1。方框内的“发动机控制”是功能块 FB1 的符号名,方框上面的“汽油机数据”是对应的背景数据块 DB1 的符号名。方框内是功能块的形参,方框外是对应的实参。方框的左边是块的输入量,右边是块的输出量。功能块的符号名是在符号表中定义的。

两次调用功能块“发动机控制”时,功能块的输入变量和输出变量不同,除此之外,分别使用汽油机的背景数据块“汽油机数据”和柴油机的背景数据块“柴油机数据”。两个背景数据块中的变量相同,区别仅在于变量的实际参数(即实参)不同和静态参数(例如预置转速)的初值不同。

背景数据块中的变量与“发动机控制”功能块的变量声明表中的变量相同(不包括临时变量 TEMP)。

2. 功能的调用

功能 FC 没有背景数据块,不能给功能的局域变量分配初值,所以必须给功能分配实参。STEP7 为功能提供了一个特殊的输出参数——返回值(RET _ VAL),调用功能时,可以指定一个地址作为实参来存储返回值。

方框“风扇控制”是控制风扇的功能 FC1,它用于在发动机停机后风扇继续运行 4 s 后再停止运行。在符号表中定义了两次调用 FC1 时使用的定时器、用于起动风扇的 FC1 的输入变量和输出变量的符号,并定义了 FC1 的符号。

下面是用语句表编写的部分 OB1 程序,值得注意的是它不能全部转换为梯形图,不能转换的部分仍用语句表表示。为了能全部转换为图 6-10 中的梯形图,还需要增加一些语句。

```
Network1:自动手动切换
        A       "自动"
        S       "自动模式"
        A       "手动"
        R       "自动模式"
Network2:汽油机控制
        CALL    "发动机控制","汽油机数据"
          Switch _ On                 :="起动汽油机"
          Switch _ Off                :="关闭汽油机"
          Failure                     :="汽油机故障"
          Actual _ Speed              :="汽油机转速"
          Engine _ On                 :="汽油机运行"
          Preset _ Speed _ Reached    :="汽油机到达设置转速"
Network3:汽油机风扇控制
        CALL    "风扇控制"
          Engine _ On                 :="汽油机运行"
          Timer _ Function            :="汽油机风扇延时"
```

```
        Fan _ On                    := "汽油机风扇运行"
Network4:柴油机控制
    CALL  "发动机控制","柴油机数据"
        Switch _ On                 := "起动柴油机"
        Switch _ Off                := "关闭柴油机"
        Failure                     := "柴油机故障"
        Actual _ Speed              := "柴油机转速"
        Engine _ On                 := "柴油机运行"
        Preset  Speed  Reached      := "柴油机到达设置转速"
Network5:柴油机风扇控制
    CALL  "风扇控制"
        Engine _ On                 := "柴油机运行"
        Timer _ Function            := "柴油机风扇延时"
        Fan _ On                    := "柴油机风扇运行"
```

6.2.5 时间标记冲突与一致性检查

如果修改了块与块之间的软件接口(块内的输入/输出变量)或程序代码,可能会造成时间标记(Time Stamp)冲突,引起调用块和被调用块(基准块)之间的不一致,在打开调用块时,在块调用指令中被调用的有冲突的块将用红色标出。

块中包含一个代码(Code)时间标记和一个接口(Interface)时间标记,这些时间标记可以在块属性对话框中查看。STEP 7 在进行时间标记比较时,如果发现下列问题,就会显示时间标记冲突:

(1) 被调用的块比调用它的块的时间标记更新;

(2) 用户定义数据类型(UDT)比使用它的块或使用它的用户数据的时间标记更新;

(3) FB 比它的背景数据块的时间标记更新;

(4) FB2 在 FB1 中被定义为多重背景,FB2 的时间标记比 FB1 的更新。

即使块与块之间的时间标记的关系是正确的,如果块的接口的定义与它被使用的区域中的定义不匹配(有接口冲突),也会出现不一致性。

如果用手工来消除块的不一致性,工作是很繁重的。可用下面的方法自动修改一致性错误:

(1) 在 SIMATIC 管理器的项目窗口中选择要检查的块文件夹,执行菜单命令"Edit"→"Check Block Consistency"(检查块的一致性)。在出现的窗口中执行菜单命令"Program"→"Compile"(程序→编译),STEP 7 将自动地识别有关块的编程语言,并打开相应的编辑器去进行修改。时间标记冲突和块的不一致性被自动地尽可能地消除,同时对块进行编译。将在视窗下面的输出窗口中显示不能自动消除的时间冲突和不一致性。所有的块被自动地重复进行上述处理。如果是用可选的软件包生成的块,可选软件包必须有一致性检查功能,才能作一致性检查。

(2) 如果在编译过程中不能自动清除所有的块的不一致性,在输出窗口中给出有错误的块的信息。用鼠标右键点击某一错误,调用弹出菜单中的错误显示,对应的错误被打开,程序将跳到被修改的位置。清除块中的不一致性后,保存并关闭块。对于所有标记为有错误的块,重复这一过程。

(3) 重新执行步骤(1)和(2),直至在信息窗口中不再显示错误信息。

6.3 数据块

6.3.1 数据块中的数据类型

1. 基本数据类型

基本数据类型包括位(Bool),字节(Byte)、字(Word)、双字(Dword)、整数(INT)、双整数(DINT)和浮点数(Float,或称实数 Real)等。

2. 复合数据类型

复合数据类型包括日期和时间(DATE _ AND _ TIME),字符串(STRING),数组(ARRAY),结构(STRCT)和用户定义数据类型(UDT)。

(1) 日期和时间

日期和时间用 8 个字节的 BCD 码来存储。第 0～5 个字节分别存储年、月、日、时、分和秒,毫秒存储在字节 6 和字节 7 的高 4 位,星期存放在字节 7 的低 4 位。例如 2004 年 7 月 27 日 12 点 30 分 25.123 秒可以表示为 DT#04-07-27-12:30:25.123。

(2) 字符串

字符串(STRING)由最多 254 个字符(CHAR)和 2 B 头部组成。字符串的默认长度为 254,通过定义字符串的长度可以减少它占用的存储空间。

3. 数组

数组(ARRAY)是同一类型的数据组合而成的一个单元。生成数组时,应指定数组的名称,例如 PRESS(见图 6-12),声明数组的类型时要使用关键字 ARRAY,用下标(Index)指定数组的维数和大小,数组的维数最多为 6 维。例如图 6-11 给出了一个二维数组 ARRAY[1..2,1..3]的结构,它共有 6 个整数元素,图中的每一小格为二进制的 1 位,每个元素占两行(两个字节)。方括号中的数字用来定义每一维的起始元素和结束元素在该维中的编号,可以取-32 768～32 767 之间的整数。各维之间的数字用逗号隔开,每一维开始和结束的编号用两个小数点隔开,如果某一维有 n 个元素,该维的起始元素和结束元素的编号一般采用 1 和 n,例如 ARRAY[1..2,1..3]。

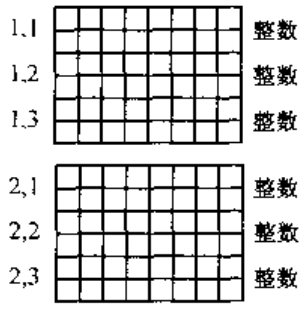


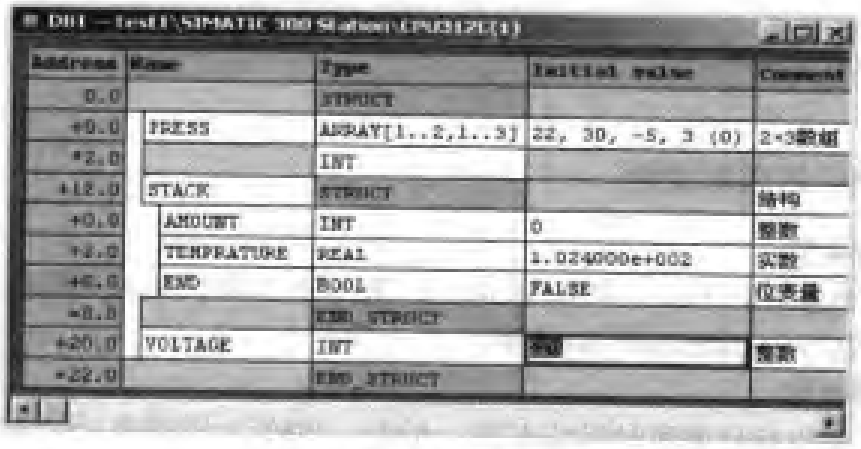
图 6-11 二维数组的结构

(1) 生成数组

数组可以在数据块中定义,也可以在逻辑块的变量声明表中定义。下面介绍在数据块中定义的方法。在 SIMATIC 管理器中用菜单命令"Insert"→"S7 block"→"Data Block"生成一个数据块,点击该数据块的图标,在出现的窗口中用声明表显示方式来生成一个用户定义的数组(见图 6-12)。

在新生成的数据块的声明表中的第一行和最后一行,标有 STRUCT(结构)和 END _ STRUCT(结构结束)。图 6-12 中名为 PRESS 的二维数组有 2×3 个元素,ARRAY 下面一行的"INT"用来定义数组元素均为 16 位二进制整数,INT 所在行的地址列中的"*2.0"表示一个数组元素占用 2 B。地址列中的"+12.0"表示该数组的 6 个元素一共占用 12 B,地址列中的

数字和加号等都是自动生成的。



Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	PRESS	ARRAY[1..2,1..3]	22, 30, -5, 3 (0)	2×3数组
+2.0		INT		
+12.0	STACK	STRUCT		结构
+0.0	AMOUNT	INT	0	整数
+2.0	TEMPRATURE	REAL	1.024000e+002	实数
+6.0	END	BOOL	FALSE	位变量
=8.0		END_STRUCT		
+20.0	VOLTAGE	INT	0	整数
=22.0		END_STRUCT		

图 6-12　定义数组与结构

数组中的第 1 个元素为 PRESS[1,1](见图 6-11)，第 3 个元素为 PRESS[1,3]，第 4 个元素为 PRESS[2,1]，第 6 个元素为 PRESS[2,3]。

(2) 给数组元素赋初值

定义数组时可以在 ARRAY 所在的行的"initial value"列中给数组元素赋初值，各元素的初值之间用英语逗号分隔，例如上例中 6 个元素的初值可以写成"22、30、-5、0、0、0"，结束时不用标点符号。若相邻元素的初值相同，写法可以简化，例如上述初值可以简写为"22、30、-5、3(0)"(见图 6-12)。

(3) 访问数组中的数据

本例中的数组是数据块的一部分，访问数组中的数据时，需要指出数据块和数组的名称，以及数组元素的下标，例如"TANK".PRESS[2,1]。其中 TANK 是数据块的符号名，PRESS 是数组的名称，它们用英语的句号分开。方括号中是数组元素的下标，该元素是数组中的第 4 个元素(见图 6-11)。

(4) 用数组传递参数

如果在块的变量声明表中声明形参的类型为 ARRAY，可以将整个数组而不是某些元素作为参数来传递。在调用块时也可以将某个数组元素赋值给同一类型的参数。

将数组作为参数传递时，并不要求作为形参和实参的两个数组有相同的名称，但是它们应该有相同的结构，例如都是由整数组成的 2×3 格式的数组。

4. 结构

结构(STRUCT)是不同类型的数据的组合。可以用基本数据类型、复杂数据类型(包括数组和结构)和用户定义数据类型 UDT 作为结构中的元素，例如一个结构由数组和结构组成，结构可以嵌套 8 层。用户可以把过程控制中有关的数据统一组织在一个结构中，作为一个数据单元来使用，而不是使用大量的单个的元素，为统一处理不同类型的数据或参数提供了方便。

(1) 结构的生成

与数组一样，结构可以在数据块中定义，也可以在逻辑块的变量声明表中定义，下面介绍在数据块中定义的方法。图 6-12 的数据块 DB1 中，同时定义了一个数组和一个结构。名为 STACK 的结构由一个整数、一个实数和一个位变量组成。图 6-12 中同时还定义了一个名为

VOLTAGE 的独立的整型变量。

为了生成一个结构，在 STACK 所在行的“Type”列输入“STRUCT”，在结构最后一个元素下面一列输入“END_STRUCT”，分别表示用户定义的结构 STACK 的开始和结束。在 STRUCT 和 END_STRUCT 之间的各行输入结构的元素，其中的“Address”列中的地址是自动生成的。

图中 STACK 所在行的地址列中的 +12.0 表示结构在数据块中的起始地址为 DBB12。结构各元素的地址列中的“+2.0”等表示结构元素在结构中的相对起始地址，“=8.0”表示该结构一共占用 8 B。最后一行地址列中的“=22.0”表示表中的数组、结构和变量一共占用 22 B。

可以为结构中各个元素设置初值(Initial Value)和加上注释(Comment)。在图 6-12 中输入实数的初值 102.4 后，被自动转换为 1.024000e+002。

(2) 访问结构中的元素

可以用结构中的元素的绝对地址或符号地址来访问结构中的元素。访问结构中的数据时，需要指出结构所在的数据块的名称、结构的名称，以及结构元素的名称，数据块 TANK 内结构 STACK 的元素 AMOUNT 应表示为“TANK”.STACK.AMOUNT。因为 AMOUNT 从数据块 TANK(DB1)的第 12 个字节开始存放，它的绝对地址为 DB1.DBW12。

(3) 用结构传递参数

如果在块的变量声明表中，声明形参的类型为 STRUCT，可以将整个结构而不是某些元素作为参数来传递。在调用块时也可以将某个结构元素赋值给同一类型的参数。

将结构作为参数传递时，作为形参和实参的两个结构必须有相同的数据结构，即相同数据类型的结构元素和相同的排列顺序。

5. 用户定义数据类型

用户定义数据类型(UDT)是一种特殊的数据结构，由用户自己生成，定义好后可以在用户程序中多次使用。用户定义数据类型由基本数据类型或复杂数据类型组成。定义好后可以在符号表中为它指定一个符号名，使用 UDT 可以节约录入数据的时间。

在 SIMATIC 管理器中用菜单命令“Insert”→“S7 Block”→“Data Type”生成 UDT，默认的名称为 UDT1。也可以用鼠标右键点击 SIMATIC 管理器的块工作区，在弹出的菜单中选择“Insert New Object”→“Data Type”命令，生成新的 UDT。在生成 UDT 的元素时，可以设置它的初值(Initial Value)和加上注释(Comment)。

从表面上看，图 6-13 中的 UDT1 与图 6-12 中定义的结构 STACK 完全相同，但是它们有本质的区别。结构(STRUCT)是在数据块的声明表中或在逻辑块的变量声明表中与别的变量一起定义的，但是 UDT 必须在名为 UDT 的特殊数据块内单独定义，并单独存放在一个数据块内。生成 UDT 后，在定义变量时将它作为一个数据类型来多次使用，例如在变量声明表中定义一个变量，其数据类型为 UDT1，名称为 ProData(见表 6-5)。由该表可以看出，UDT 在数据块中的使用方法与其他数据类型(例如 INT)是一样的。

表 6-5　在数据块 TANK 中使用 UDT 的例子

Address	Name	Type	Initial Value	Comment
0.0	T_STAMP	STRUCT		
+0.0	Pressure	INT		
+2.0	ProData	UDT1		
=10.0		END_STRUCT		

UDT 可以在逻辑块(FC、FB、OB)的变量声明表中作为基本数据类型或复杂数据类型来使用,或者在数据块(DB)中作为变量的数据类型来使用。

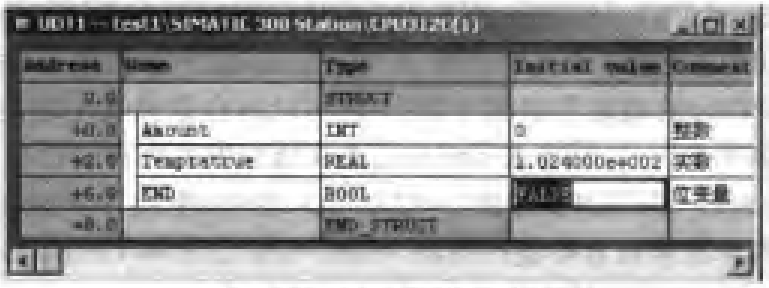


图 6-13 用户定义数据结构

要访问数据块 TANK 中数据类型为 UDT1 的变量 ProData 中的元素 AMOUNT,其符号地址为"TANK".ProData. AMOUNT。

可以将具有数据类型 UDT 的变量作为参数来传递。如果在块的变量声明表中,声明形参的类型为 UDT,在调用块时应使用具有相同结构的 UDT 来传递参数。在调用块时也可以将某个 UDT 的元素赋值给同一类型的参数。

使用用户定义数据类型时,只需要对它定义一次,就可以用它来产生大量的具有相同数据结构的数据块,可以用这些数据块来输入用于不同目的的实际数据。例如可以生成用于颜料混合配方的 UDT,然后用它生成用于不同颜色配方的数据组合。

用户定义数据类型也可以用来作为生成具有相同数据结构的数据块的模板。

6.3.2 数据块的生成与使用

数据块(DB)用来分类储存设备或生产线中变量的值,数据块也是用来实现各逻辑块之间的数据交换、数据传递和共享数据的重要途径。数据块丰富的数据结构便于提高程序的执行效率和进行数据管理。与逻辑块不同,数据块只有变量声明部分,没有程序指令部分。

不同的 CPU 允许建立的数据块的块数和每个数据块可以占用的最大字节数是不同的,具体的参数可以查看选型手册。

1. 数据块的类型

数据块分为共享数据块(DB)和背景数据块(DI)两种。

共享数据块又称为全局数据块,它不附属于任何逻辑块。在共享数据块中和全局符号表中声明的变量都是全局变量。用户程序中所有的逻辑块(FB、FC、OB 等)都可以使用共享数据块和全局符号表中的数据。

背景数据块是专门指定给某个功能块(FB)或系统功能块(SFB)使用的数据块,它是 FB 或 SFB 运行时的工作存储区。当用户将数据块与某一功能块相连时,该数据块即成为该功能块的背景数据块,功能块的变量声明表决定了它的背景数据块的结构和变量。不能直接修改背景数据块,只能通过对应的功能块的变量声明表来修改它。调用 FB 时,必须同时指定一个对应的背景数据块。只有 FB 才能访问存放在它的背景数据块中的数据。

在符号表中,共享数据块的数据类型是它本身,背景数据块的数据类型是对应的功能块。

多次使用同一个功能块时需要调用不同的背景数据块,可以将这些数据块中的数据存放在一个多重背景数据块中,但是需要增加一个管理多重背景的功能块。多重背景数据块将在

下一节中介绍。

2. 生成共享数据块

在 SIMATIC 管理器中用菜单命令"Insert"→"S7 Block"→"Data Block"生成数据块，在出现的对话框中选择是共享数据块还是背景数据块。也可以用鼠标右键点击 SIMATIC 管理器的块工作区，在弹出的菜单中选择"Insert New Object"→"Data Block"命令，生成新的数据块。

数据块有两种显示方式，即声明表显示方式和数据显示方式，菜单命令"View"→"Declaration View"和"View"→"Data View"分别用来指定这两种显示方式。图 6-14 和图 6-15 是发动机控制系统中的共享数据块 DB3 的两种不同的显示状态。

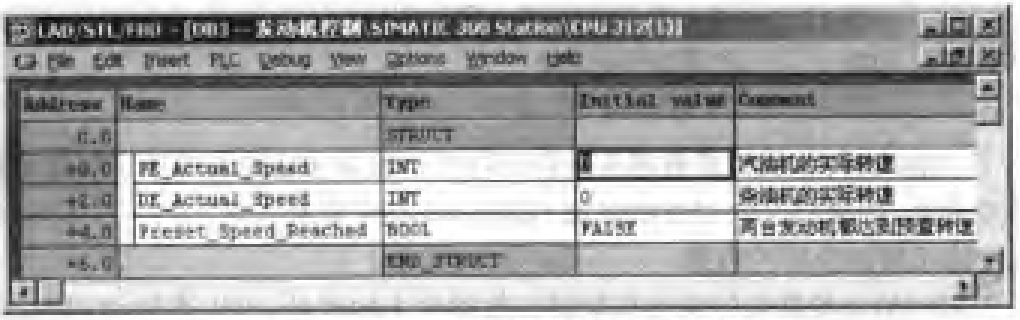


图 6-14 声明表显示状态下的共享数据块 DB3

图 6-15 数据显示状态下的共享数据块 DB3

声明表显示状态用于定义和修改共享数据块中的变量，指定它们的名称、类型和初值，STEP 7 根据数据类型给出默认的初值，用户可以修改初值。可以用中文给每个变量加上注释，声明表中的名称只能使用字母、数字和下划线，地址是 CPU 自动指定的。

在数据显示状态，显示声明表中的全部信息和变量的实际值，用户只能改变每个元素的实际值。复杂数据类型变量的元素（例如数组中的各元素）用全名列出。如果用户输入的实际值与变量的数据类型不符，将用红色显示错误的数据。在数据显示状态下，用菜单命令"Edit"→"Inicialize Data Block"可以恢复变量的初始值。

3. 生成背景数据块

要生成背景数据块，首先应生成对应的功能块（FB），然后再生成背景数据块。

在 SIMATIC 管理器中，用菜单命令"Insert"→"S7 Block"→"Data Block"生成数据块，在弹出的窗口中，选择数据块的类型为背景数据块（Instance），并输入对应的功能块的名称。操作系统在编译功能块时将自动生成功能块对应的背景数据块中的数据，其变量与对应的功能块的变量声明表中的变量相同，不能在背景数据块中增减变量，只能在数据显示（Data View）方式修改其实际值。在数据块编辑器的"View"菜单中选择是声明表显示方式还是数据显示方式。

4. 访问数据块

在访问数据块时，需要指明被访问的是哪一个数据块，以及访问该数据块中的哪一个数

据。有两种访问数据块中的数据的方法：

(1) 先打开后访问

访问数据块中的数据时，需要先打开它，由于只有两个数据块寄存器，即 DB 寄存器和 DI 寄存器，同时只能打开一个共享数据块和一个背景数据块。它们的块号分别存放在 DB 寄存器和 DI 寄存器中。打开新的数据块后，原来打开的数据块自动关闭。

下面的例程说明了这种访问方法：

```
OPN     DB2     //打开数据块 DB2
A       DBX4.5  //如果 DB2.DBX4.5 的常开触点接通
L       DBW12   //将 DB2.DBW12 装入累加器 1
OPN     DB3     //打开数据块 DB3
T       DBW4    //将累加器 1 中的数据传送到 DB3.DBW4
```

调用一个功能块时，它的背景数据块被自动打开。如果该功能块调用了其他的块，调用结束后返回该功能块，原来打开的背景数据块不再有效，必须重新打开它。

(2) 直接访问数据块中的数据

在指令中同时给出数据块的编号和数据在数据块中的地址，可以直接访问数据块中的数据。访问时可以使用绝对地址，也可以使用符号地址。数据块中的存储单元的地址由两部份组成，例如 DB2.DBX2.0。DB2 是数据块的名称，DBX2.0 是数据块内第 2 个字节的第 0 位。如果打开了数据块 DB2，可以省略第一个小数点前面的数据块编号。

这种访问方法不容易出错，建议尽量使用这种方法。上面的指令可以等效为：

```
A       DB2.DBX4.5
L       DB2.DBW12       //将 DB2.DBW12 装入累加器 1
T       DB3.DBW4        //将累加器 1 中的数据传送到 DB3.DBW4
```

6.4 多重背景

在用户程序中使用多重背景可以减少背景数据块的数量。以发动机控制程序为例，原来用 FB1 控制汽油机和柴油机时，分别使用了背景数据块 DB1 和 DB2。使用多重背景时只需要一个背景数据块(DB10)，但是需要增加一个功能块 FB10 来调用作为"局域背景"的 FB1，FB1 的数据存储在 FB10 的背景数据块 DB10 中，DB10 是自动生成的。不需要给 FB1 分配背景数据块，即原来的 DB1 和 DB2 被 DB10 代替，但是需要在 FB10 的变量声明表中声明静态局域数据(STAT)FB1。

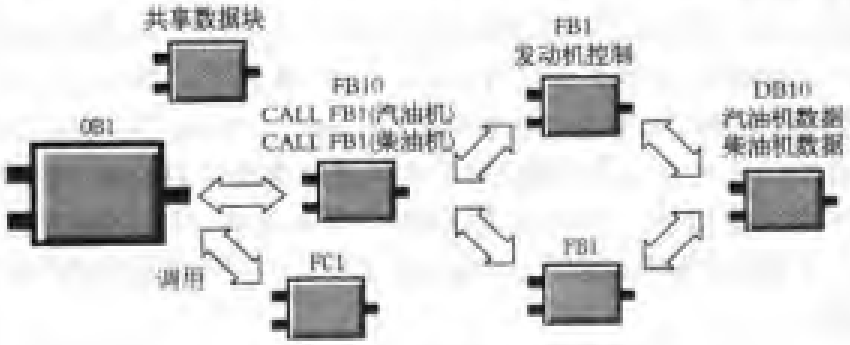


图 6-16 多重背景的程序结构

6.4.1 多重背景功能块

生成多重背景功能块 FB10 时，应激活“Multiple Instance FB”（多重背景功能块）选项。

生成 FB10 时，应首先生成 FB1。为调用 FB1，在 FB10 的变量声明表中（见图 6-17），声明了两个名为“Petrol _ Engine”（汽油机）和“Diesel _ Engine”（柴油机）的静态变量（STAT），其数据类型为 FB1。图 6-17 中“Petrol _ Engine”和“Diesel _ Engine”下面的 7 个子变量来自 FB1 的变量声明表，不是用户输入的。生成 FB10 后，“Petrol _ Engine”和“Diesel _ Engine”将出现在程序编辑器编程元件目录的“Multiple Instances”（多重背景）文件夹内。可以将它们“拖放”到 FB 10 中，然后指定它们的输入参数和输出参数。

图 6-17　FB10 的变量声明表

图 6-18 是 FB10 的梯形图程序，下面是用语句表编写的 FB10 的程序：

```
Network1：汽油机控制
        CALL    # Petrol _ Engine
          Switch _ On              :="起动汽油机"
          Switch _ Off             :="关闭汽油机"
          Failure                  :="汽油机故障"
          Actual _ Speed           :="汽油机转速"
          Engine _ On              :="汽油机运行"
          Preset _ Speed _ Reached :=# PE _ Preset _ Speed _ Reached      //汽油机达到预置转速
Network2：柴油机控制
        CALL    # Diesel _ Engine
          Switch _ On              :="起动柴油机"
          Switch _ Off             :="关闭柴油机"
          Failure                  :="柴油机故障"
```

```
Actual _ Speed              :="柴油机转速"
Engine _ On                 :="柴油机运行"
Preset _ Speed _ Reached    := #DE _ Preset _ Speed _ Reached      //柴油机达到预置转速
Network3:两台发动机都达到预置转速
      A      #PE _ Preset _ Speed _ Reached      //汽油机达到预置转速
      A      #DE _ Preset _ Speed _ Reached      //柴油机达到预置转速
      =      #Preset _ Speed _ Reached           //汽油机柴油机都达到预置转速
```

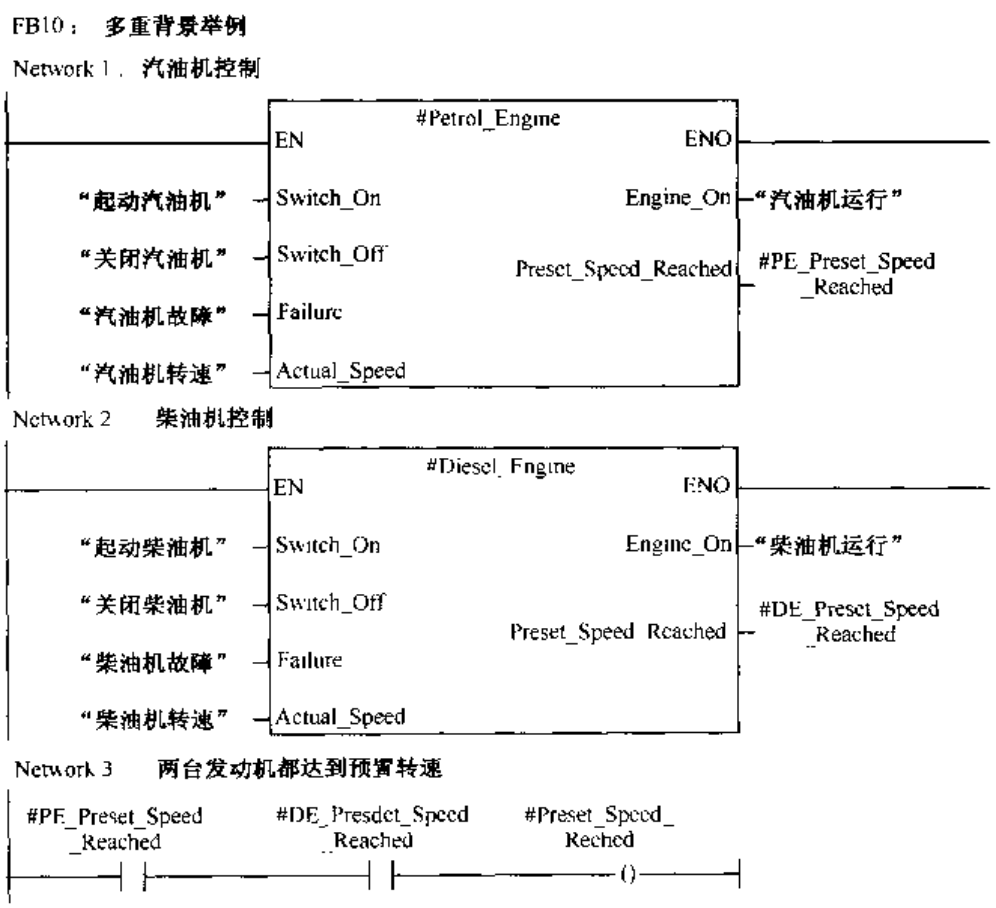


图 6-18 多重背景功能块 FB10

6.4.2 多重背景数据块

汽油机和柴油机的数据均存储在多重背景数据块 DB10 中，DB10 代替了原有的背景数据块 DB1 和 DB2。生成 DB10 时，应将它设置为背景数据块，对应的功能块为 FB10。DB10 中的变量是自动生成的，与 FB10 的变量声明表中的相同。

打开 DB10(见图 6-19)，执行菜单命令"View"→"Data View"，可以修改预置转速的实际值。FB1 中的变量仍保持它们的符号名，例如"Switch _ On"，局域背景的名称"Petrol _ Engine"和"Diesel _ Engine"加在 FB1 的变量之前，例如"Petrol _ Engine.Switch _ On"。

6.4.3 在 OB1 中调用多重背景

给 OB1 编程之前，打开符号表，输入 FB10 和 DB10 的符号名(见图 6-20)，保存后退出。前面的"发动机控制"项目中 OB1 对 FB1 的两次调用，被图 6-21 中 OB1 对 FB10(符号名为"发

动机")的调用代替,调用时还指定了背景数据块 DB10(符号名为"多重背景数据块")。FB10 的输出信号"Preset _ Speed _ Reached"送给符号名为"两台达到设置转速"的共享数据 Q5.1。

DB10 -- 多重背景\SIMATIC 300 Station\CPU 312(1)

	Address	Declaration	Name	Type	Initial value	Actual value
1	0.0	out	Preset_Speed_Reached	BOOL	FALSE	FALSE
2	2.0	stat:in	Petrol_Engine.Switch_On	BOOL	FALSE	FALSE
3	2.1	stat:in	Petrol_Engine.Switch_Off	BOOL	FALSE	FALSE
4	2.2	stat:in	Petrol_Engine.Failure	BOOL	FALSE	FALSE
5	4.0	stat:in	Petrol_Engine.Actual_Speed	INT	0	0
6	6.0	stat:out	Petrol_Engine.Engine_On	BOOL	FALSE	FALSE
7	6.1	stat:out	Petrol_Engine.Preset_Speed_Reached	BOOL	FALSE	FALSE
8	8.0	stat	Petrol_Engine.Preset_Speed	INT	1500	1500
9	10.0	stat:in	Diesel_Engine.Switch_On	BOOL	FALSE	FALSE
10	10.1	stat:in	Diesel_Engine.Switch_Off	BOOL	FALSE	FALSE
11	10.2	stat:in	Diesel_Engine.Failure	BOOL	FALSE	FALSE
12	12.0	stat:in	Diesel_Engine.Actual_Speed	INT	0	0
13	14.0	stat:out	Diesel_Engine.Engine_On	BOOL	FALSE	FALSE
14	14.1	stat:out	Diesel_Engine.Preset_Speed_Reached	BOOL	FALSE	FALSE
15	16.0	stat	Diesel_Engine.Preset_Speed	INT	1500	1500

图 6-19　多重背景数据块 DB10 中的变量

Symbol Editor - S7 Program(1) (Symbols)

Symbol Table Edit Insert View Options Window Help

S7 Program(1) (Symbols) -- 多重背景\SIMATIC 300 Station\CPU 312(1)

	Status	Symbol	Address	Data type	Comment
1		共享数据	DB 3	DB 3	共享数据块
2		多重背景数据块	DB 10	FB 10	存放汽油机柴油机数据
3		发动机控制	FB 1	FB 1	FB10调用的功能块
4		发动机	FB 10	FB 10	用于多重背景的功能块
5		风扇控制	FC 1	FC 1	功能
6		自动	I 0.5	BOOL	自动按钮
7		手动	I 0.6	BOOL	手动按钮
8		起动汽油机	I 1.0	BOOL	控制按钮
9		关闭汽油机	I 1.1	BOOL	控制按钮
10		汽油机故障	I 1.2	BOOL	故障输入
11		起动柴油机	I 1.4	BOOL	控制按钮
12		关闭柴油机	I 1.5	BOOL	控制按钮
13		柴油机故障	I 1.6	BOOL	故障输入
14		汽油机转速	MW 2	INT	实际转速
15		柴油机转速	MW 4	INT	实际转速
16		主程序	OB 1	OB 1	用户主程序
17		自动模式	Q 4.2	BOOL	工作模式指示灯
18		汽油机运行	Q 5.0	BOOL	控制汽油机运行的输出
19		两台都达到设置转速	Q 5.1	BOOL	控制指示灯的输出
20		汽油机风扇运行	Q 5.2	BOOL	控制汽油机风扇的输出
21		柴油机运行	Q 5.4	BOOL	控制柴油机运行的输出
22		柴油机风扇运行	Q 5.5	BOOL	控制柴油机风扇的输出
23		汽油机风扇延时	T 1	TIMER	断电延时定时器
24		柴油机风扇延时	T 2	TIMER	断电延时定时器
25					

图 6-20　符号表

图 6-21 中调用 FB10 的语句表为:

```
Network4:调用多重背景
CALL  "发动机","多重背景数据块"
Preset _ Speed _ Reached:="两台都达到设置转速"
```

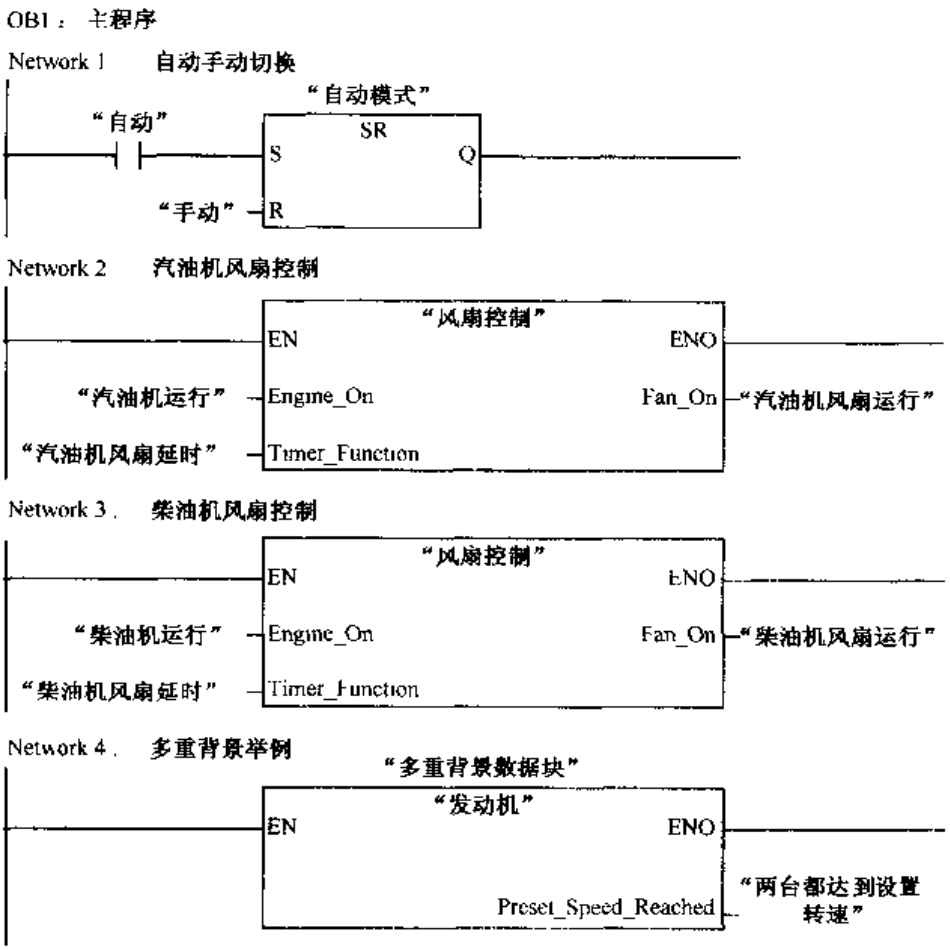


图 6-21 多重背景的 OB1

使用多重背景时应注意以下问题：

(1) 首先应生成需要多次调用的功能块(例如上例中的 FB1)。

(2) 管理多重背景的功能块(例如上例中的 FB10)必须设置为有多重背景功能。

(3) 在管理多重背景的功能块的变量声明表中，为被调用的功能块的每一次调用定义一个静态(STAT)变量，以被调用的功能块的名称(例如 FB1)作为静态变量的数据类型。

(4) 必须有一个背景数据块(例如上例中的 DB10)分配给管理多重背景的功能块。背景数据块中的数据是自动生成的。

(5) 多重背景只能声明为静态变量(声明类型为"Stat")。

6.5 组织块与中断处理

组织块是操作系统与用户程序之间的接口。S7 提供了各种不同的组织块(OB)，用组织块可以创建在特定的时间执行的程序和响应特定事件的程序，例如延时中断 OB、外部硬件中断 OB 和错误处理 OB 等。

6.5.1 中断的基本概念

1. 中断过程

中断处理用来实现对特殊内部事件或外部事件的快速响应。如果没有中断，CPU 循环执

行组织块 OB1。因为除背景组织块 OB90 以外，OB1 的中断优先级最低，CPU 检测到中断源的中断请求时，操作系统在执行完当前程序的当前指令（即断点处）后，立即响应中断。CPU 暂停正在执行的程序，调用中断源对应的中断程序。在 S7-300/400 中，中断用组织块（OB）来处理。执行完中断程序后，返回被中断的程序的断点处继续执行原来的程序。

PLC 的中断源可能来自 I/O 模块的硬件中断，或是 CPU 模块内部的软件中断，例如日期时间中断、延时中断、循环中断和编程错误引起的中断。

如果在执行中断程序（组织块）时，又检测到一个中断请求，CPU 将比较两个中断源的中断优先级。如果优先级相同，按照产生中断请求的先后次序进行处理。如果后者的优先级比正在执行的 OB 的优先级高，将中止当前正在处理的 OB，改为调用较高优先级的 OB。这种处理方式称为中断程序的嵌套调用。

一个 OB 被另一个 OB 调用时，操作系统对现场进行保护。被中断的 OB 的局域数据压入 L 堆栈（局域数据堆栈），被中断的断点处的现场信息保存在 I 堆栈（中断堆栈）和 B 堆栈（块堆栈）中。

中断程序不是由程序块调用，而是在中断事件发生时由操作系统调用。因为不能预知系统何时调用中断程序，中断程序不能改写其他程序中可能正在使用的存储器，应在中断程序中尽可能地使用局域变量。

编写中断程序时，应使中断程序尽量短小，以减少中断程序的执行时间，减少对其他处理的延迟，否则可能引起主程序控制的设备操作异常。设计中断程序时应遵循“越短越好”的格言。

2. 组织块的分类

组织块只能由操作系统启动，它由变量声明表和用户编写的控制程序组成。

（1）启动组织块

启动组织块用于系统初始化，CPU 上电或操作模式改为 RUN 时，根据起动的方式执行启动程序 OB100～OB102 中的一个。

（2）循环执行的组织块

需要连续执行的程序存放在 OB1 中，执行完后又开始新的循环。

（3）定期执行的组织块

包括日期时间中断组织块 OB10～OB17 和循环中断组织块 OB30～OB38，可以根据设定的日期时间或时间间隔执行中断程序。

（4）事件驱动的组织块

延时中断 OB20～OB23 在过程事件出现后延时一定的时间再执行中断程序；硬件中断 OB40～OB47 用于需要快速响应的过程事件，事件出现时马上中止循环程序，执行对应的中断程序。异步错误中断 OB80～OB87 和同步错误中断 OB121、OB122 用来决定在出现错误时系统如何响应。

3. 中断的优先级

中断的优先级也就是组织块的优先级，较高优先级的组织块可以中断较低优先级的组织块的处理过程。如果同时产生的中断请求不止一个，最先执行优先级最高的 OB，然后按照优先级由高到低的顺序执行其他 OB。

下面是优先级的顺序（后面的比前面的优先）：背景循环、主程序扫描循环、日期时间中断、时间延时中断、循环中断、硬件中断、多处理器中断、I/O 冗余错误、异步故障（OB8087）、启动

和 CPU 冗余，背景循环的优先级最低。

S7-300 CPU(不包括 CPU 318)中组织块的优先级是固定的，可以用 STEP 7 修改 S7-400 CPU 和 CPU 318 下述组织块的优先级：OB10～OB47(优先级 2～23)，OB70～OB72(优先级 25 或 28，只适用于 H 系列 CPU)，以及在 RUN 模式下的 OB81～OB87(优先级 26 或 28)。

同一个优先级可以分配给几个 OB，具有相同优先级的 OB 按启动它们的事件出现的先后顺序处理。被同步错误启动的故障 OB 的优先级与错误出现时正在执行的 OB 的优先级相同。

生成逻辑块 OB、FB 和 FC 时，同时生成临时局域变量数据，CPU 的局域数据区按优先级划分。可以用 STEP 7 在“优先级”参数块中改变 S7-400 每个优先级的局域数据区的大小。

每个组织块的局域数据区都有 20 B 的起动信息，它们是只在该块被执行时使用的临时变量(TEMP)，这些信息在 OB 启动时由操作系统提供，包括启动事件、起动日期与时间，错误及诊断事件。将优先级赋值为 0，或分配小于 20 B 的局域数据给某一个优先级，可以取消相应的中断 OB。

4. 对中断的控制

日期时间中断和延时中断有专用的允许处理中断(或称激活、使能中断)和禁止中断的系统功能(SFC)。

SFC 39“DIS_INT”用来禁止中断和异步错误处理，可以禁止所有的中断，有选择地禁止某些优先级范围的中断，或者只禁止指定的某个中断。

SFC 40“EN_INT”用来激活(使能)新的中断和异步错误处理，可以全部允许或有选择地允许。

SFC 41“DIS_AIRT”延迟处理比当前优先级高的中断和异步错误，直到用 SFC 42 允许处理中断或当前的 OB 执行完毕。

SFC 42“EN AIRT”用来允许立即处理被 SFC 41 暂时禁止的中断和异步错误，SFC 42 和 SFC 41 配对使用。

6.5.2 组织块的变量声明表

组织块(OB)是操作系统调用的，OB 没有背景数据块，也不能为 OB 声明静态变量，因此 OB 的变量声明表中只有临时变量。OB 的临时变量可以是基本数据类型、复合数据类型或数据类型 ANY。

操作系统为所有的 OB 块声明了一个 20 B 的包含 OB 的起动信息的变量声明表，声明表中变量的具体内容与组织块的类型有关。用户可以通过 OB 的变量声明表获得与起动 OB 的原因有关的信息。

OB 的变量声明表见表 6-6。

表 6-6 OB 的变量声明表

地址(字节)	内 容
0	事件级别与标识符，例如 OB40 为 B#16#11，表示硬件中断被激活
1	用代码表示与启动 OB 的事件有关的信息
2	优先级，例如 OB40 的优先级为 16
3	OB 块号，例如 OB40 的块号为 40
4～11	附加信息，例如 OB40 的第 5 个字节为产生中断的模块的类型，16#54 为输入模块，16#55 为输出模块；第 6，7 字节组成的字为产生中断的模块的起始地址；第 8～11 字节组成的双字为产生中断的通道号
12～19	OB 被起动的日期和时间(年、月、日、时、分、秒、毫秒与星期)

6.5.3 日期时间中断组织块

CPU 318 只能使用 OB10 和 OB11，其余的 S7-300 CPU 只能使用 OB10。S7-400 CPU 可以使用的日期时间中断 OB(OB10～OB17)的个数与 CPU 的型号有关。

日期时间中断 OB 可以在某一特定的日期和时间执行一次，也可以从设定的日期时间开始，周期性地重复执行，例如每分钟、每小时、每天、甚至每年执行一次。可以用 SFC 28～SFC 30 取消、重新设置或激活日期时间中断。

只有设置了中断的参数，并且在相应的组织块中有用户程序存在，日期时间中断才能被执行。如果不满足上述条件，操作系统将会在诊断缓冲区中产生一个错误信息，并执行异步错误处理。如果设置从 1 月 31 日开始每月执行一次 OB10，只在有 31 天的那些月启动它。

日期时间中断在 PLC 暖启动或热启动时被激活，而且只能在 PLC 起动过程结束之后才能执行。暖启动后必须重新设置日期时间中断。

1. 设置和起动日期时间中断

为了启动日期时间中断，用户首先必须设置日期时间中断的参数，然后再激活它。有以下三种方法可以启动日期时间中断：

(1) 在用户程序中用 SFC 28“SET_TINT”和 SFC 30“ACT_TINT”设置和激活日期时间中断。

(2) 在 STEP 7 中打开硬件组态工具，双击机架中 CPU 模块所在的行，打开设置 CPU 属性的对话框，点击“Time-Of-Day Interrupts”选项卡，设置启动时间日期中断的日期和时间，选中“Active”(激活)多选框，在“Execution”列表框中选择执行方式。将硬件组态数据下载到 CPU 中，可以实现日期时间中断的自动启动。

(3) 用上述方法设置日期时间中断的参数，但是不选择“Active”，而是在用户程序中用 SFC 30“ACT_TINT”激活日期时间中断。

2. 查询日期时间中断

要想查询设置了哪些日期时间中断，以及这些中断什么时间发生，用户可以调用 SFC 31“QRY_TINT”，或查询系统状态表中的“中断状态”表。

SFC 31 输出的状态字节 STATUS 见表 6-7。

表 6-7 SFC 31 输出的状态字节 STATUS

位	取值	意义
0	0	日期时间中断已被激活
1	0	允许新的日期时间中断
2	0	日期时间中断未被激活或时间已过去
3	0	-
4	0	没有装载日期时间中断组织块
5	0	日期时间中断组织块的执行没有被激活的测试功能禁止
6	0	以基准时间为日期时间中断的基准
	1	以本地时间为日期时间中断的基准

3. 禁止日期时间中断

用户可以用 SFC 29“CAN_TINT”取消(禁止)日期时间中断，用 SFC 28“SET_TINT”重

新设置那些被禁止的日期时间中断，用 SFC 30“ACT _ TINT”重新激活日期时间中断。

4. 日期时间中断的优先级

8 个日期时间中断 OB 均具有相同的默认优先级(第 2 级)，它们之间的优先级是按启动事件发生的顺序处理的，用户可以通过选择适当的参数来改变优先级。

日期时间中断组织块 OB10 的局域变量表见表 6-8。

表 6-8 日期时间中断组织块 OB10 的局域变量表

参 数	数据类型	描 述
OB10 _ EV _ CLASS	BYTE	事件级别与标识符：B#16#11 为中断被激活
OB10 _ STRT _ INFO	BYTE	B#16#11～B#16#18：OB10～OB17 的起动请求
OB10 _ PRIORITY	BYTE	优先级，默认值为 2
OB10 _ OB _ NUMBR	BYTE	OB 号(10 -17)
OB10 _ RESERVED _ 1	BYTE	保留
OB10 _ RESERVED _ 2	BYTE	保留
OB10 _ PERIOD _ EXE	WORD	OB 运行的时间间隔
OB10 _ RESERVED _ 3	INT	保留
OB10 _ RESERVED _ 4	INT	保留
OB10 _ DATE _ TIME	DATE _ AND _ TIME	OB 被调用时的日期和时间

在调用 SFC 28 时，如果参数“OB10 _ PERIOD _ EXE”为十六进制数 W#16#0000、W#16#0201、W#16#0401、W#16#1001、W#16#1201、W#16#1401、W#16#1801 和 W#16#2001，分别表示执行一次、每分钟、每小时、每天、每周、每月、每年和月末执行一次。

【例 6-1】 在 I0.0 的上升沿时启动日期时间中断 OB10，在 I0.1 为 1 时禁止日期时间中断，每次中断使 MW2 加 1。从 2004 年 7 月 1 日 8 时开始，每分钟中断一次，每次中断 MW2 被加 1。

在 STEP 7 中生成项目“OB10 例程”，为了便于调用，例程中对日期时间中断的操作都放在功能 FC 12 中，在 OB1 中用指令 CALL FC 12 调用它。下面是用 STL 编写的 FC 12 的程序代码，它有一个临时局域变量“OUT _ TIME _ DATE”。

IEC 功能 D _ TOD _ TD(FC3)在程序编辑器左边的指令目录与程序库窗口的文件夹 \ Libraries \ Standard Library \ IEC Function Blocks 中。

```
Network 1:查询 OB10 的状态
      CALL SFC 31                          //查询日期时间中断 OB10 的状态
       OB _ NO      := 10                  //日期时间中断 OB 的编号
       RET _ VAL    := MW208               //保存执行时可能出现的错误代码，为 0 时无错误
       STATUS       := MW16                //保存日期时间中断的状态字，MB17 为低字节
Network 2:合并日期时间
      CALL FC 3                            //调用 IEC 功能 D _ TOD _ TD
       IN1          := D#2004-7-1          //设置启动中断的日期和时间
       IN2          := TOD#8:0:0.0
       RET _ VAL    := #OUT _ TIME _ DATE  //合并日期和时间
```

```
Network 3:在 I0.0 的上升沿设置和激活日期时间中断
        A       I0.0
        FP      M1.0            //如果在 I0.0 的上升沿,M1.0 为 1
        AN      M17.2           //如果日期时间中断已被激活时,M17.2 的常闭触点闭合
        A       M17.4           //如果装载了日期时间中断 OB 时,M17.4 的常开触点闭合
        JNB     m005            //没有同时满足以上 3 个条件则跳转
        CALL    SFC28           //同时满足则调用 SFC"SET_TINT",设置日期时间中断参数
         OB_NO   := 10          //日期时间中断 OB 编号
         SDT     := #OUT_TIME_DATE   //启动中断的时间,秒和毫秒被省略(置为 0)
         PERIOD  := W#16#201//设置产生中断的周期为每分钟一次
         RET_VAL:=MW 200        //保存执行时可能出现的错误代码,为 0 时无错误
        CALL    SFC 30          //调用 SFC"ACT_TINT",激活日期时间中断
         OB_NO   := 10          //日期时间中断 OB 编号
         RET_VAL:= MW204        //保存执行时可能出现的错误代码,为 0 时无错误
m005:   NOP 0
Network 4:在 I0.1 的上升沿禁止日期时间中断
        A       I0.1
        FP      M1.1            //检测 I0.1 的上升沿
        JNB     m004            //不是 I0.1 上升沿则跳转
        CALL    SFC 29          //调用 SFC"CAN_TINT",禁止日期时间中断
          OB_NO  := 10          //日期时间中断 OB 编号
          RET_VAL:=MW 210       //保存执行时可能出现的错误代码,为 0 时无错误 m004:
    NOP 0
```

下面是用 STL 编写的 OB10 中断程序,每分钟 MW2 被加 1 一次。

```
Network 1
        L       MW2
        +       1
        T       MW2
```

有时间错误出现时,CPU 的操作系统调用 OB80。时间错误包括:

(1) 实际循环时间超过在 CPU 模块属性中设置的最大循环时间;

(2) 执行 OB 时的应答错误;

(3) 因为向前修改时间而跳过日期时间中断 OB 的起动时间;

(4) CiR(在 CPU 中组态)之后恢复为 RUN 方式。

如果 OB80 未编写程序,CPU 将转换到 STOP 模式。下面是用 STL 编写的 OB80 的程序代码,如果出现了时间错误,Q4.1 将被置位,并将 OB80 的启动事件信息保存到 MW110~MW119 中。

```
Network 1:
        AN      Q4.1
        S       Q4.1
        CALL    SFC 20                  //数据块传送
```

```
SRCBLK      := #OB80_EV_CLASS          //指定源地址
RET_VAL     := MW210                   //保存可能的错误信息
DSTBLK      := P#M110.0 Byte 20        //指定目的地址,拷贝 20 个字节
```

可以在 PLCSIM 仿真软件中运行上述例程,运行时监视 M 17.2,M17.4 和 MW2。M17.2 为 1 时表示日期时间中断被激活,17.4 为 1 时表示已经装载了日期时间中断组织块 OB10。用 I0.0 激活日期时间中断,M17.2 变为 1 状态,每分钟 MW2 将被加 1。用 I0.1 禁止日期时间中断,M17.2 变为 0 状态,MW2 停止加 1。

6.5.4 延时中断组织块

PLC 中的普通定时器的工作与扫描工作方式有关,其定时精度受到不断变化的循环周期的影响。使用延时中断可以获得精度较高的延时,延时中断以 ms 为单位定时。

S7 提供了 4 个延时中断 OB(OB20～OB23),它们用 SFC 32"SRT_DINT"启动,延时时间在 SFC 32 中设置,启动后经过设定的延时时间后触发中断,调用 SFC 32 指定的 OB。需要延时执行的操作放在 OB 中,例如立即输出一个数字量信号。必须将延时中断 OB 作为用户程序的一部分下载到 CPU。

CPU 318 只能使用 OB20 和 OB21,其余的 S7-300 CPU 只能使用 OB20。S7－400 CPU 可以使用的日期时间中断 OB 的个数与 CPU 的型号有关。

只有当该中断设置了参数,并且在相应的组织块中有用户程序存在时,延时中断才能被执行。如果不满足上述条件,操作系统会在诊断缓冲区中输入一个错误信息,并执行异步错误处理。

延时中断 OB 优先级的默认设置值为 3～6 级,用户可以通过设置参数改变优先级。

如果延时中断已被启动,延时时间还没有到达,可以用 SFC 33"CAN_DINT"取消延时中断的执行。SFC 34"QRY_DINT"用来查询延时中断的状态。

只有在 CPU 处于运行状态时才能执行延时中断 OB,暖启动或冷启动都会清除延时中断 OB 的起动事件.

如果下列任何一种情况发生,操作系统将会调用异步错误 OB:

(1) OB 已经被 SFC 32 启动,但是没有下载到 CPU。

(2) 延时中断 OB 正在执行延时,又有一个延时中断 OB 被启动。

延时中断组织块的 OB20 的局域变量表见表 6-9。

表 6-9 延时中断组织块 OB20 的局域变量表

参 数	数据类型	描 述
OB20_EV_CLASS	BYTE	事件级别和标识码,B#16#11:中断已被激活
OB20_STRT_INF	BYTE	B#16#20～B#16#23:OB20～OB23 的起动请求
OB20_PRIORITY	BYTE	优先级,默认值为 3 (OB 20)～6(OB23)
OB20_O__NUMBR	BYTE	OB 号(20～23)
OB20_RESERVED_1	BYTE	保留
OB20_RESERVED_2	BYTE	保留
OB20_SIGN	WORD	用户号:调用 SFC 32 (SRT_DINT)时输入的参数标记
OB20_DTIME	TIME	以 ms 为单位的延时时间
OB20_DATE_TIME	DATE_AND_TIME	OB 被调用时的日期和时间

【例 6-2】 在主程序 OB1 中实现下列功能：

(1) 在 I0.0 的上升沿用 SFC 32 起动延时中断 OB20，10s 后 OB20 被调用，在 OB20 中将 Q4.0 置位，并立即输出。

(2) 在延时过程中如果 I0.1 由 0 变为 1，在 OB1 中用 SFC 33 取消延时中断，OB20 不会再被调用。

(3) I0.2 由 0 变为 1 时 Q4.0 被复位。

SFC 34 输出的状态字节 STATUS 见表 6-10。

表 6-10 SFC 34 输出的状态字节 STATUS

位	取 值	意 义
0	0	延时中断已被允许
1	0	未拒绝新的延时中断
2	0	延时中断未被激活或已完成
3	0	-
4	0	没有装载延时中断组织块
5	0	日期时间中断组织块的执行没有被激活的测试功能禁止

项目的名称为“OB20 例程”，下面是用 STL 编写的 OB1 程序代码：

```
Network 1：I0.0 的上升沿时启动延时中断
          A       I0.0
          FP      M1.0
          JNB     m001                  //不是 I0.0 的上升沿则跳转
          CALL    SFC 32                //启动延时中断 OB20
            OB_NO      :=20             //组织块编号
            DTME       :=T#10S          //延时时间为 10 s
            SIGN       :=MW12           //保存延时中断是否启动的标志
            RET_VAL    :=MW100          //保存执行时可能出现的错误代码，为 0 时无错误
m001：    NOP 0
Network 2：查询延时中断
          CALL    SFC 34                //查询延时中断 OB20 的状态
            OB_NO      :=20             //组织块编号
            RET_VAL    :=MW102          //保存执行时可能出现的错误代码，为 0 时无错误
            STATUS     :=MW4            //保存延时中断的状态字，MB5 为低字节
Network 3：I0.1 上升沿时取消延时中断
          A       I0.1
          FP      M1.1                  // I0.1 的上升沿检测
          A       M5.2                  //延时中断未被激活或已完成(状态字第 2 位为 0)时跳转
          JNB     m002
          CALL    SFC 33                //禁止 OB20 延时中断
            OB_NR      :=20             //组织块编号
            RET_VAL    :=MW104          //保存执行时可能出现的错误代码，为 0 时无错误
m002：    NOP     0
          A       I0.2
          R       Q4.0                  //I0.2 为“1”时复位 Q4.0
```

下面是用 STL 编写的 OB20 的程序代码：

```
Network 1：
        SET
        =       Q 4.0           //将 Q4.0 无条件置位
Network 2：
        L       QW 4            //立即输出 Q4.0
        T       PQW 4
```

可以用 PLCSIM 仿真软件模拟运行上述例程，运行时监视 M5.2 和 M5.4。将程序下载到仿真 PLC，进入 RUN 模式时，M5.4 马上变为 1 状态，表示 OB20 已经下载到了 CPU 中。用 I0.0 起动延时中断后，M5.2 变为 1 状态，延时时间到时 Q4.0 变为 1 状态，M5.2 变为 0 状态。在延时过程中用 I0.1 禁止 OB20 延时，M5.2 也会变为 0 状态。

6.5.5 循环中断组织块

循环中断组织块用于按一定时间间隔循环执行中断程序，例如周期性地定时执行闭环控制系统的 PID 运算程序，间隔时间从 STOP 切换到 RUN 模式时开始计算。

用户定义时间间隔时，必须确保在两次循环中断之间的时间间隔中有足够的时间处理循环中断程序。

循环中断组织块的临时局域变量见表 6-11。

表 6-11 循环中断组织块的临时局域变量

参 数	数据类型	描 述
OB35_EV_CLASS	BYTE	事件级别与标识码：11H 为中断被激活
OB35_STRT_INF	BYTE	B#16#30：特殊标准的循环中断 OB 的起动请求，仅用于 H-CPU； B#16#31～B#16#39：OB30～OB38 的起动请求
OB35_PRIORITY	BYTE	分配的优先级，默认值为 7（OB30）～15（OB38）
OB35_OB_NUMBR	BYTE	OB 号（30～38）
OB35_RESERVED_1	BYTE	保留
OB35_RESERVED_2	BYTE	保留
OB35_PHASE_OFFSET	WORD	相位偏移（ms）
OB35_RESERVED_3	INT	保留
OB35_EXC_FREQ	INT	执行的时间间隔（ms）
OB35_DATE_TIME	DATE_AND_TIME	OB 被调用时的日期和时间

循环中断组织块 OB30～OB38 默认的时间间隔和中断优先级见表 6-12，用户可以通过参数设置来改变优先级。CPU 318 只能使用 OB32 和 OB35，其余的 S7-300 CPU 只能使用 OB35。S7-400 CPU 可以使用的日期时间中断 OB 的个数与 CPU 的型号有关。

如果两个 OB 的时间间隔成整倍数，不同的循环中断 OB 可能同时请求中断，造成处理循环中断服务程序的时间超过指定的循环时间。为了避免出现这样的错误，用户可以定义一个相位偏移。相位偏移用于在循环时间间隔到达时，延时一定的时间后再执行循环中断。相位偏移 m 的单位为 ms，应有 $0 \leqslant m < n$，式中 n 为循环的时间间隔。

假设OB38和OB37的时间间隔分别为10 ms和20 ms,它们的相位偏移分别为0 ms和3 ms。OB38分别在$t=10$ ms,20 ms,……,60 ms时产生中断,而OB37分别在$t=23$ ms,43 ms,63 ms时产生中断。

表6-12 循环OB默认的参数

OB号	时间间隔	优先级
OB30	5s	7
OB31	2s	8
OB32	1s	9
OB33	500ms	10
OB34	200ns	11
OB35	100ms	12
OB36	50ms	13
OB37	20ms	14
OB38	10ms	15

没有专用的SFC来激活和禁止循环中断,可以用SFC 40和SFC 39来激活和禁止它们。SFC 40"EN_INT"是用于激活新的中断和异步错误的系统功能,其参数MODE为0时激活所有的中断和异步错误,为1时激活部分中断和错误,为2时激活指定的OB编号对应的中断和异步错误。SFC 39"DIS_INT"是禁止新的中断和异步错误的系统功能,MODE为2时禁止指定的OB编号对应的中断和异步错误,MODE必须用十六进制数来设置。

【例6-3】 在I0.0的上升沿时起动OB35对应的循环中断,在I0.1的上升沿禁止OB35对应的循环中断,在OB35中使MW2加1。

在STEP 7中生成名为"OB35例程"的项目,选用CPU 312C,在硬件组态工具中打开CPU属性的组态窗口,由"Cyclic Interrupts"选项卡可知只能使用OB35,其循环周期的默认值为100ms,将它修改为1000ms,将组态数据下载到CPU中。下面是用STL编写的OB1的程序代码:

```
Network 1:在I0.0的上升沿激活循环中断
          A       I0.0
          FP      M1.1
          JNB   m001                    //不是I0.0的上升沿时跳转
          CALL    SFC 40                //激活OB35对应的循环中断
           MODE       :=B#16#2          //用OB编号指定中断
           OB_NO      :=35              //OB编号
           RET_VAL    :=MW100           //保存执行时可能出现的错误代码,为0时无错误
m001:     NOP 0
Network 2:在I0.1的上升沿禁止循环中断
          A       I0.1
          FP      M1.2
          JNB   m002                    //不是I0.1的上升沿则跳转
          CALL    SFC 39                //禁止OB35对应的循环中断
           MODE       :=B#16#2          //用OB编号指定中断
           OB_NR      :=35              //OB编号
           RET_VAL    :=MW104           //保存执行时可能出现的错误代码,为0时无错误
m002:     NOP 0
```

下面是用STL编写的OB35中断程序,每经过1000 ms,MW2被加1一次。

```
Network 1
          L       MW2                   //MW2加1
          +       1
          T       MW2
```

可以用 PLCSIM 仿真软件模拟运行上述例程，将程序和硬件组态参数下载到仿真 PLC，进入 RUN 模式后，可以看每秒钟 MW2 的值加 1。用鼠标模拟产生 I0.1 的脉冲，循环中断被禁止，MW2 停止加 1。用鼠标模拟 I0.0 的脉冲，循环中断被激活，MW2 又开始加 1。

6.5.6 硬件中断组织块

硬件中断组织块(OB40～OB47)用于快速响应信号模块(SM，即输入/输出模块)、通信处理器(CP)和功能模块(FM)的信号变化。具有中断能力的信号模块将中断信号传送到 CPU 时，或者当功能模块产生一个中断信号时，将触发硬件中断。

CPU 318 只能使用 OB40 和 OB41，其余的 S7-300 CPU 只能使用 OB40。S7-400 CPU 可以使用的硬件中断 OB 的个数与 CPU 的型号有关。

用户可以用 STEP 7 的硬件组态功能来决定信号模块哪一个通道在什么条件下产生硬件中断，将执行哪个硬件中断 OB，OB40 被默认用于执行所有的硬件中断。对于 CP 和 FM，可以在对话框中设置相应的参数来启动 OB。

只有用户程序中有相应的组织块，才能执行硬件中断。否则操作系统会向诊断缓冲区中输入错误信息，并执行异步错误处理组织块 OB80。

硬件中断 OB 的默认优先级为 16～23，用户可以设置参数改变优先级。

硬件中断被模块触发后，操作系统将自动识别是哪一个槽的模块和模块中哪一个通道产生的硬件中断。硬件中断 OB 执行完后，将发送通道确认信号。

如果在处理硬件中断的同时，又出现了其他硬件中断事件，新的中断按以下方法识别和处理：如果正在处理某一中断事件，又出现了同一模块同一通道产生的完全相同的中断事件，新的中断事件将丢失，即不处理它。在图 6-22 中数字量模块输入信号的第一个上升沿时触发中断，由于正在用 OB40 处理中断，第 2 个和第 3 个上升沿产生的中断信号丢失。

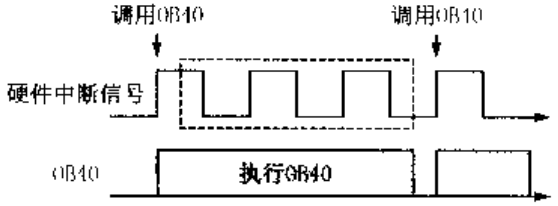


图 6-22 硬件中断信号的处理

如果正在处理某一中断信号时同一模块中其他通道产生了中断事件，新的中断不会被立即触发，但是不会丢失。在当前已激活的硬件中断执行完后，再处理被暂存的中断。

如果硬件中断被触发，并且它的 OB 被其他模块中的硬件中断激活，新的请求将被记录，空闲后再执行该中断。

用 OB39～OB42 可以禁止、延迟和再次激活硬件中断。

硬件中断组织块 OB40 的临时变量见表 6-13。

表 6-13 硬件中断组织块 OB40 的临时变量

参 数	数据类型	描 述
OB40_EV_CLASS	BYTE	事件级别与标识码，B#16#11：中断被激活
OB40_STRT_INF	BYTE	B#16#41：通过中断线 1 的中断 B#16#42～B#16#44：通过中断线 2～4 (S7-400)的中断 B#16#45：WinAC 通过 PC 触发的中断
OB40_PRIORITY	BYTE	优先级，默认值为 16 (OB40)～23(OB47)
OB40_OB_NUMBR	BYTE	OB 号(40～47)

（续）

参　数	数据类型	描　述
OB40_RESERVED_1	BYTE	保留
OB40_IO_FLAG	BYTE	I/O标志：输入模块为B#16#54，输出模块为B#16#55
OB40_MDL_ADDR	WORD	触发中断的模块的起始字节地址
OB40_POINT_ADDR	DWORD	数字量输入模块内的位地址，第0位对应第一个输入；或模拟量模块超限的通道对应的位域。对于CP和FM是模块的中断状态（与用户无关）
OB40_DATE_TIME	DATE_AND_TIME	OB被调用时的日期和时间

以S7-300插在4号槽的16点数字量输入模块为例，模块的起始地址为0(IB0)，模块内输入点I0.0～I1.7的位地址为0～15。

【例6-4】 CPU 313C－2DP集成的16点数字量输入I124.0I125.7可以逐点设置中断特性，通过OB40对应的硬件中断，在I124.0的上升沿将CPU 313C－2DP集成的数字量输出Q124.0置位，在I124.1的下降沿将Q124.0复位。此外要求在I0.2的上升沿时激活OB40对应的硬件中断，在I0.3的下降沿禁止OB40对应的硬件中断。

在STEP 7中生成名为“OB40例程”的项目，选用CPU 313C-2DP，在硬件组态工具中打开CPU属性的组态窗口，由“Interrupts”选项卡可知在硬件中断中，只能使用OB40。双击机架中CPU 313C－2DP内的集成I/O“DI16/DO16”所在的行（见图4-4），在打开的对话框的“Input”选项卡中，设置在I124.0的上升沿和I124.1的下降沿产生中断。下面是用STL编写的OB1的程序代码：

```
Network 1:在I0.2的上升沿激活硬件中断
          A       I0.2
          FP      M1.2
          JNB   m001                    //不是I0.2的上升沿时则跳转
          CALL    SFC 40                //激活OB40对应的硬件中断
           MODE       := B#16#2         //用OB编号指定中断
           OB_NO      :=40              //OB编号
           RET_VAL    :=MW 100          //保存执行时可能出现的错误代码，为0时无错误
m001:     NOP 0
Network 2:在I0.3的上升沿禁止硬件中断
          A       I0.3
          FP      M1.3
          JNB   m002                    //不是I0.3的上升沿时则跳转
          CALL    SFC 39                //禁止OB40对应的硬件中断
           MODE       := B#16#2         //用OB编号指定中断
           OB_NR      :=40              //OB编号
           RET_VAL    :=MW104           //保存执行时可能出现的错误代码，为0时无错误
m002:     NOP       0
```

下面是用STL编写的硬件中断组织块OB40的程序代码，在OB40中通过比较指令

"==="判别是哪一个模块和哪一点输入产生的中断。在 I124.0 的上升沿将 Q124.0 置位,在 I124.1 的下降沿将 Q124.0 复位。

OB40_POINT_ADDR 是数字量输入模块内的位地址(第 0 位对应第一输入),或模拟量模块超限的通道对应的位域。对于 CP 和 FM 是模块的中断状态(与用户无关)。

```
Network 1:
        L       # OB40_MLD_ADDR
        L       124
        ==I
        =M0.0                          //如果模块起始地址为 IB124,则 M0.0 为 1 状态
Network 2:
        L       # OB40_POINT_ADDR
        L       0
        ==I
        =M0.1                          //如果是第 0 位产生的中断,则 M0.1 为 1 状态
Network 3:
        L       # OB40_POINT_ADDR
        L       1
        ==I
        =M0.2                          //如果是第 1 位产生的中断,则 M0.2 为 1 状态
Network 4:
        A       M0.0
        A       M0.1
        S       Q124.0                 //如果是 I124.0 产生的中断,将 Q124.0 置位
Network 5:
        A       M0.0
        A       M0.2
        R       Q124.0                 //如果是 I124.1 产生的中断,将 Q124.0 复位
```

下面介绍在 PLCSIM 仿真软件中模拟硬件中断的方法。将仿真 PLC 切换到 RUN 模式,用鼠标模拟产生一个 I0.2 的脉冲输入信号,激活 OB40 对应的硬件中断。用 PLCSIM 的菜单命令"Execute"→"Trigger Error OB"→"Hardware Interrupt (OB40-OB47)…"打开"Hardware Interrupt (OB40-OB47)"对话框(见图 6-23),在文本框"Module address"内输入模块的起始地址 124,在文本框"Module status(POINT_ADDR)"内输入模块内的位地址 0。单击【Apply】键触发指定的硬件中断,执行一次 OB40,使 Q124.0 变为 1 状态,同时在"Interrupt"显示框内将自动显示出对应的 OB 编号 40。将位地址改为 1,单击【Apply】使 Q124.0 变为 0 状态。单击【OK】将执行与【Apply】键同样的操作,同时退出对话框。

图 6-23 硬件中断的模拟

在 PLCSIM 中用鼠标模拟 I0.3 输入的脉冲信号,OB40 对应的硬件中断被禁止。这时再

用图 6-23 中的对话框模拟硬件中断的操作不会起作用。

在仿真时指定的硬件模块必须是在硬件组态中存在的模块。

6.5.7 启动时使用的组织块

1. CPU 模块的启动方式

CPU 有 3 种起动方式:暖启动、热启动(仅 S7-400 有)和冷起动,在用 STEP 7 设置 CPU 的属性时可以选择 S7-400 上电后启动的方式。

S7-300 CPU(不包括 CPU 318)只有暖启动,用 STEP 7 可以指定存储器位、定时器、计数器和数据块在电源掉电后的保持范围。

在起动期间,不能执行时间驱动的程序和中断驱动的程序,运行时间计数器开始工作,所有的数字量输出信号都为"0"状态。

(1) 暖启动(Warm Restart)

暖启动时,过程映像数据以及非保持的存储器位、定时器和计数器被复位。具有保持功能的存储器位、定时器、计数器和所有数据块将保留原数值。程序将重新开始运行,执行启动 OB 或 OB1。

手动暖启动时,将模式选择开关扳到 STOP 位置,"STOP"LED 亮,然后扳到 RUN 或 RUN－P 位置。

(2) 热启动(Hot Restart)

在 RUN 状态时如果电源突然丢失,然后又重新上电,S7-400 CPU 将执行一个初始化程序,自动地完成热启动。热启动从上次 RUN 模式结束时程序被中断之处继续执行,不对计数器等复位。热启动只能在 STOP 状态时没有修改用户程序的条件下才能进行。

(3) 冷启动(Cold Restart,CPU 417 和 CPU 417H)

冷启动时,过程数据区的所有过程映像数据、存储器位、定时器、计数器和数据块均被清除,即被复位为零,包括有保持功能的数据。用户程序将重新开始运行,执行启动 OB 和 OB1。

手动冷启动时将模式选择开关扳到 STOP 位置,STOP LED 亮,再扳到 MRES 位置,STOP LED 灭 1 s,亮 1 s,再灭 1 s 后保持亮。最后将它扳到 RUN 或 RUN-P 位置。

2. 启动组织块(OB100～OB102)

下列事件发生时,CPU 执行起动功能:PLC 电源上电后;CPU 的模式选择开关从 STOP 位置扳到 RUN 或 RUN-P 位置;接收到通过通信功能发送来的启动请求;多 CPU 方式同步之后和 H 系统连接好后(只适用于备用 CPU)。

启动用户程序之前,先执行启动 OB。在暖启动、热启动或冷启动时,操作系统分别调用 OB100、OB101 或 OB102,S7-300 和 S7-400H 不能热启动。

用户可以通过在启动组织块 OB100～OB102 中编写程序,来设置 CPU 的初始化操作,例如开始运行的初始值,I/O 模块的起始值等。

启动程序没有长度和时间的限制,因为循环时间监视还没有被激活,在起动程序中不能执行时间中断程序和硬件中断程序。

CPU 318-2 只允许手动暖启动或冷启动。对于某些 S7-400 CPU,如果允许用户通过 STEP 7 的参数设置手动启动,用户可以使用状态选择开关和启动类型开关(CRST/WRST)进行手动启动。

在设置 CPU 模块属性的对话框中，选择“Startup”选项卡，可以设置启动的各种参数。

启动 S7-400 CPU 时，作为默认的设置，将输出过程映像区清零。如果用户希望在启动之后继续在用户程序中使用原有的值，也可以选择不将过程映像区清零。

用户设置组态表的参数时，可以决定是否在组态表中检查模块是否存在，以及模块类型与启动前是否匹配。如果激活模块检查功能，发现组态表与实际的组态不相符时，CPU 将不会启动。

为了在启动时监视是否有错误，用户可以选择以下的监视时间：

(1) 向模块传递参数的最大允许时间；

(2) 上电后模块向 CPU 发送“准备好”信号允许的最大时间；

(3) S7-400 CPU 热起动允许的最大时间，即电源中断的时间或由 STOP 转换为 RUN 的时间。一旦超过监视时间，CPU 将进入停机状态或只能暖启动。如果监控时间设置为 0，表示不监控。

OB100 的变量声明表中的 OB100 _ STRTUP 用代码表示各种不同的启动方式，OB100 _ STOP 是引起停机的事件号，OB100 _ STRT _ INFO 是当前启动的更详细的信息。各参数的具体意义参见有关的参考手册。

6.5.8 异步错误组织块

1. 错误处理概述

S7-300/400 有很强的错误(或称故障)检测和处理能力。这里所说的错误是 PLC 内部的功能性错误或编程错误，而不是外部传感器或执行机构的故障。CPU 检测到某种错误后，操作系统调用对应的组织块，用户可以在组织块中编程，对发生的错误采取相应的措施。对于大多数错误，如果没有给组织块编程，出现错误时 CPU 将进入 STOP 模式。

系统程序可以检测出下列错误：不正确的 CPU 功能、系统程序执行中的错误、用户程序中的错误和 I/O 中的错误。根据错误类型的不同，CPU 被设置为进入 STOP 模式或调用一个错误处理 OB。

当 CPU 检测到错误时，会调用适当的组织块(见表 6-14)，如果没有相应的错误处理 OB，CPU 将进入 STOP 模式。用户可以在错误处理 OB 中编写如何处理这种错误的程序，以减小或消除错误的影响。

表 6-14 错误处理组织块

OB 号	错误类型	优先级
OB 70	I/O 冗余错误(仅 H 系列 CPU)	25
OB 72	CPU 冗余错误(仅 H 系列 CPU)	28
OB 73	通信冗余错误(仅 H 系列 CPU)	35
OB 80	时间错误	26
OB 81	电源故障	26/28
OB 82	诊断中断	
OB 83	插入/取出模块中断	
OB 84	CPU 硬件故障	
OB 85	优先级错误	
OB 86	机架故障或分布式 I/O 的站故障	
OB 87	通信错误	
OB 121	编程错误	引起错误的 OB 的优先级
OB 122	I/O 访问错误	

为避免发生某种错误时 CPU 进入停机状态，可以在 CPU 中建立一个对应的空的组织块。

操作系统检测到一个异步错误时，将启动相应的 OB。异步错误 OB 具有最高等级的优先级，如果当前正在执行的 OB 的优先级低于 26，异步错误 OB 的优先级为 26，如果当前正在执行的 OB 的优先级为 27(启动组织块)，异步错误 OB 的优先级为 28，其他 OB 不能中断它们。如果同时有多个相同优先级的异步错误 OB 出现，将按出现的顺序处理它们。

用户可以利用 OB 中的变量声明表提供的信息来判别错误的类型，OB 的局域数据中的变量 OB8x _ FLT _ ID 和 OB12x _ SW _ FLT 包含有错误代码。它们的具体含义见《S7-300/400 的系统软件和标准功能参考手册》。

2. 错误的分类

被 S7 CPU 检测到并且用户可以通过组织块对其进行处理的错误分为两个基本类型：

(1) 异步错误

异步错误是与 PLC 的硬件或操作系统密切相关的错误，与程序执行无关。异步错误的后果一般都比较严重。异步错误对应的组织块为 OB70～OB73 和 OB80～OB87(见表 6-14)，有最高的优先级。

(2) 同步错误

同步错误是与程序执行有关的错误，OB121 和 OB122 用于处理同步错误，它们的优先级与出现错误时被中断的块的优先级相同，即同步错误 OB 中的程序可以访问块被中断时累加器和状态寄存器中的内容。对错误进行适当处理后，可以将处理结果返回被中断的块。

3. 电源故障处理组织块(OB81)

电源故障包括后备电池失效或未安装，S7-400 的 CPU 机架或扩展机架上的 DC 24V 电源故障。电源故障出现和消失时操作系统都要调用 OB81。

OB81 的变量声明表见表 6-15。

表 6-15　OB81 的变量声明表

参　数	数据类型	描　述
OB81 _ EV _ CLASS	BYTE	错误级别与标识：B#16#38 为故障消失，B#16#39 为故障产生
OB81 _ FLT _ ID	BYTE	错误代码： B#16#21 = 中央机架至少有一个后备电池耗尽/问题排除 B#16#22 = 中央机架后备电压故障/问题排除 B#16#23 = 中央机架 24 V 电压故障/问题排除 B#16#25 = 至少一个冗余的中央机架中至少一个后备电池耗尽/问题排除 B#16#26 = 至少一个冗余的中央机架中后备电压故障/问题排除 B#16#27 = 至少一个冗余中央机架 24 V 电源故障/问题排除 B#16#31 = 至少一个扩展机架至少一个后备电池耗尽/问题排除 B#16#32 = 至少一个扩展机架后备电压故障/问题排除 B#16#33 = 至少一个扩展机架 24V 电源故障/问题排除
OB81 _ PRIORITY	BYTE	优先级，可以用 STEP 7 的硬件组态功能设置，RUN 模式的可能值为 2～26
OB81 _ OB _ NUMBR	BYTE	OB 号(81)
OB81 _ RESERVED _ 1	BYTE	保留
OB81 _ RESERVED _ 2	BYTE	保留
OB81 _ MDL _ ADDR	INT	第 0～2 位为机架号，第 3 位=0，1 分别为备用 CPU 和主 CPU，4～7 位为 1111
OB81 _ RESERVED _ 3	BYTE	只与错误代码 B#16#31，B#16#32，B#16#33 有关

（续）

参　数	数据类型	描　述
OB81_RESERVED_4	BYTE	第0~5位为1分别表示16~21号扩展机架有故障
OB81_RESERVED_5	BYTE	第0~7位为1分别表示8~15号扩展机架有故障
OB81_RESERVED_6	BYTE	第1~7位为1分别表示1~7号扩展机架有故障
OB81_DATE_TIME	DATE_AND_TIME	OB被调用时的日期和时间

【例6-5】 在“中央机架后备电压消失故障”刚出现时将Q4.2置为1，该故障刚消失时将Q4.2置为0。下面是实现上述要求的OB81程序。

```
Network 1:后备电池电压消失事件处理
      L     B#16#22                 //“中央机架后备电压故障”代码
      L     #OB81_FLT_ID            //与OB81的错误代码比较
      ==I                           //如果相同
      =     M10.1                   //M10.1为1状态
      L     #OB81_EV_CLASS
      L     B#16#39
      ==I                           //如果相同
      =     M10.2                   //M10.2为1状态
      A     M10.1
      A     M10.2                   //“中央机架后备电压消失故障”刚出现
      S     Q4.2                    //置输出“中央机架后备电压故障”为1
Network 1:后备电池电压恢复正常的处理
      L     #OB81_EV_CLASS
      L     B#16#38                 //故障消失(outgoing event)的代码
      ==I                           //如果相同
      =     M10.3                   //M10.3为1状态
      A     M10.1
      A     M10.3                   //“中央机架后备电压消失故障”刚消失
      R     Q4.2                    //复位输出“中央机架后备电压故障”
```

利用系统功能(SFC)，用户可以屏蔽、延迟或禁止各种OB的起动事件。

如果用户希望忽略中断，更有效的方法不是禁止它们，而是下载一个只有块结束指令BEU的空的OB。

4. 时间错误处理组织块(OB80)

循环监控时间的默认值为150ms，时间错误包括实际循环时间超过设置的循环时间、因为向前修改时间而跳过日期时间中断、处理优先级时延迟太多等。

为OB80编程时应判断是哪个日期时间中断被跳过，使用SFC 29“CAN_TINT”可以取消被跳过的日期时间中断。只有新的日期时间中断才会被执行。

如果没有在OB80中取消跳过的日期时间中断，则执行第一个跳过的日期时间中断，其他的被忽略。

5. 诊断中断处理组织块(OB82)

如果模块有诊断功能并且激活了它的诊断中断，当它检测到错误时，以及错误消失时，操

作系统都会调用 OB82。当一个诊断中断被触发时，有问题的模块自动地在诊断中断 OB 的起动信息和诊断缓冲区中存入 4 B 的诊断数据和模块的起始地址。在编写 OB82 的程序时，要从 OB82 的起动信息中获得与出现的错误有关的更确切的诊断信息，例如是哪一个通道出错，出现的是哪种错误。使用 SFC 51“RDSYSST”可以读出模块的诊断数据，用 SFC 52“WR _ USMSG”可以将这些信息存入诊断缓冲区。也可以发送一个用户定义的诊断报文到监控设备。

OB82 在下列情况时被调用：有诊断功能的模块的断线故障，模拟量输入模块的电源故障，输入信号超过模拟量模块的测量范围等。

6. 插入/拔出模块中断组织块(OB83)

S7-400 可以在 RUN，STOP 或 STARTUP 模式下带电拔出和插入模块，但是不包括 CPU 模块、电源模块、接口模块和带适配器的 S5 模块，上述操作将会产生插入/拔出模块中断。

安装模块后 CPU 对它进行检查，如果没有错误就给它分配参数，以后就可以使用它了；如果分配参数时出现了错误，将起动诊断中断 OB82。

在下列情况下调用 OB83：被组态的模块插入或拔出，用 STEP 7 修改了模块的参数，并且在 RUN 模式时将它下载。可以用 SFC 39～SFC 42 来禁止、延时和激活 OB83。

S7-400 CPU 以大约 1 s 的间隔监视中央机架和扩展机架上的模块。电源上电时，CPU 检测由 STEP 7 生成的组态表中列出的模块是否都插入了。如果是，这个实际的组态被保存并作为对模块进行循环监控的依据。在每一扫描循环比较刚检测到的实际组态与原来检测到的组态。如果发现两个组态有差异，则发出插入/拔出模块中断信号，并且将有关信息存入诊断缓冲区和系统状态表。

如果在 RUN 模式下拔出组态的模块，将启动 OB83。因为 CPU 以大约 1 s 的间隔监视模块，在模块被直接访问或过程映像刷新时，可能首先检测出访问错误。

在拔出和插入模块这两个操作之间至少应相隔 2 s，以便让 CPU 检测到取走的或插入的模块。如果一个模块在 RUN 模式下插入，CPU 会检测新模块的类型与原来的模块是否相同。如果相同，OB83 被起动，将默认参数或用 STEP 7 指定的参数传送到该模块中。

在编写 OB83 的程序时，应根据 OB83 的起动信息，调用系统功能 SFC 55～SFC 59 对新插入的模块的参数赋值(见 8.4.6)。

7. CPU 硬件故障处理组织块(OB84)

当 CPU 检测到 MPI 网络的接口故障、通信总线的接口故障或分布式 I/O 网卡的接口故障时，操作系统调用 OB84。故障消除时也会调用该 OB 块，即事件到来和离开时都调用该 OB。在编写 OB84 的程序时，应根据 OB84 的起动信息，用系统功能 SFC 52“WR _ USMSG”发送报文到诊断缓存区。

8. 优先级错误处理组织块(OB85)

在以下情况下将会触发优先级错误中断：

(1) 产生了一个中断事件，但是对应的 OB 块没有下载到 CPU；

(2) 访问一个系统功能块的背景数据块时出错。

(3) 刷新过程映像表时 I/O 访问出错，模块不存在或有故障。

在编写 OB85 的程序时，应根据 OB85 的起动信息，判定是哪个模块损坏或没有插入。可以用 SFC 49“LGC _ GADR”查找有关模块所在的槽。

9. 机架故障组织块(OB86)

出现下列故障或故障消失时,都会触发机架故障中断,操作系统将调用OB86:扩展机架故障(不包括CPU 318),DP主站系统故障或分布式I/O故障。故障产生和故障消失时都会产生中断。

在编写OB86的程序时,应根据OB86的起动信息,判断是哪个机架损坏或找不到。可以用系统功能SFC 52"WR_USMSG"将报文存入诊断缓冲区,并将报文发送到监控设备。

10. 通信错误组织块(OB87)

在使用通信功能块或全局数据(GD)通信进行数据交换时,如果出现下列通信错误,操作系统将调用OB87:

(1) 接收全局数据时,检测到不正确的帧标识符(ID);

(2) 全局数据通信的状态信息数据块不存在或太短;

(3) 接收到非法的全局数据包编号。

如果用于全局数据通信状态信息的数据块丢失,需要用OB87生成该数据块并将它下载到CPU。

6.5.9 同步错误组织块

1. 同步错误

同步错误是与执行用户程序有关的错误,程序中如果有不正确的地址区、错误的编号或错误的地址,都会出现同步错误,操作系统将调用同步错误OB。

OB121用于对程序错误的处理;OB122用于处理模块访问错误。

同步错误OB的优先级与检测到出错的块的优先级一致。因此OB121和OB122可以访问中断发生时累加器和其他寄存器中的内容。用户程序可以用它们来处理错误,例如出现对某个模拟量输入模块的访问错误时,可以在OB122中用SFC 44定义一个替代值。

同步错误可以用SFC 36"MASK_FLT"来屏蔽,使某些同步错误不触发同步错误OB的调用,但是CPU在错误寄存器中记录发生的被屏蔽的错误。用错误过滤器中的一位来表示某种同步错误是否被屏蔽。错误过滤器分为程序错误过滤器和访问错误过滤器,分别占一个双字。错误过滤器的详细信息见《S7-300/400的系统软件和标准功能》的第11章。

调用SFC 37"DMSK_FLT"并且在当前优先级被执行完后,将解除被屏蔽的错误,并且清除当前优先级的事件状态寄存器中相应的位。

可以用SFC 38"READ_ERR"读出已经发生的被屏蔽的错误。

对于S7-300(CPU 318除外),不管错误是否被屏蔽,错误都会被送入诊断缓冲区,并且CPU的"组错误"LED会被点亮。

可以在不同的优先级屏蔽某些同步错误。在这种情况下,在特定的优先级中发生这类错误时不会停机,CPU把该错误存放到错误寄存器中。但是无法知道是什么时候发生的错误,也无法知道错误发生的频率。

表6-16中的变量PRGFLT_SET_MASK和ACCFLT_SET_MASK分别用来设置程序错误过滤器和访问错误过滤器,某位为1表示该位对应的错误被屏蔽。屏蔽后的错误过滤器可以用变量PRGFLT_MASKED和ACCFLT_MASKED读出。错误信息返回值RET_VAL为0时表示没有错误被屏蔽,为1时表示至少有一个错误被屏蔽。

表 6-16　SFC 36"MSK _ FLT"的局域变量表

参　数	声　明	数据类型	存储区	描　述
PRGFLT _ SET _ MASK	INPUT	DWORD	I,Q,M,D,L,常数	要屏蔽的程序错误
ACCFLT _ SET _ MASK	INPUT	DWORD	I,Q,M,D,L,常数	要屏蔽的访问错误
RET _ VAL	OUTPUT	INT	I,Q,M,D,L	错误信息返回值
PRGFLT _ MASKED	OUTPUT	DWORD	I,Q,M,D,L	被屏蔽的程序错误
ACCFLT _ MASKED	OUTPUT	DWORD	I,Q,M,D,L	被屏蔽的访问错误

S7-400 CPU 的同步错误 OB 可以启动另一个同步错误 OB，S7-300 CPU 没有这个功能。

2．编程错误组织块(OB121)

出现编程错误时，CPU 的操作系统将调用 OB121．OB121 的变量声明如表 6-17 所示。OB121 错误代码如表 6-18 所示。

表 6-17　OB121 的变量声明表

参　数	数据类型	描　述
OB121 _ EV _ CLASS	BYTE	错误级别与标识：B#16#25
OB121 _ SW _ FLT	BYTE	错误代码，见表 6-18
OB121 _ PRIORITY	BYTE	优先级，与出现错误的 OB 的优先级相同
OB121 _ OB _ NUMBR	BYTE	OB 号(121)
OB121 _ BLK _ TYPE	BYTE	S7-400 出错的块的类型：16#88：OB，16#8A：DB，16#8C：FC，16#8E：FB
OB121 _ RESERVED _ 1	BYTE	备用
OB121 _ FLT _ REG	WORD	错误源，例如出错的地址、定时器、计数器和块的编号，出错的存储器区
OB121 _ BLK _ NUM	WORD	引起错误的 MC7 命令的块的编号，S7-300 未用
OB121 _ PRG _ ADDR	WORD	引起错误的 MC7 命令的相对地址，S7-300 未用
OB121 _ DATE _ TIME	DATE _ AND _ TIME	OB 被调用时的日期和时间

表 6-18　OB121 错误代码表

B#16#21 OB121 _ FLT _ REG	BCD 转换错误 有关寄存器的标识符，例如累加器 1 的标识符为 0
B#16#22 B#16#23 B#16#28 B#16#29 OB121 _ FLT _ REG OB121 _ RESERVED _ 1	读操作时的区域长度错误 写操作时的区域长度错误 用指针读字节、字和双字时位地址不为 0 用指针写字节、字和双字时位地址不为 0 不正确的字节地址，可以从 OB121 _ RESERVED _ 1 读出数据区和访问类型 第 4～7 位为访问类型，为 0～3 分别表示访问位、字节、字和双字 第 0～3 位为存储器区，为 0～7 分别表示 I/O 区、过程映像输入表、过程映像输出表、位存储器、共享 DB、背景 DB、自己的局域数据和调用者的局域数据
B#16#24 B#16#25 OB121 _ FLT _ REG	读操作时的范围错误 写操作时的范围错误 低字节有非法区域的标识符（B#16#86 为自己的数据区）
B#16#26 B#16#27 OB121 _ FLT _ REG	定时器编号错误 计数器编号错误 非法的编号

（续）

B#16#21 OB121_FLT_REG	BCD转换错误 有关寄存器的标识符，例如累加器1的标识符为0
B#16#30 B#16#31 B#16#32 B#16#33 OB121_FLT_REG	对有写保护的全局DB的写操作 对有写保护的背景DB的写操作 访问共享DB时的DB编号错误 访问背景DB时的DB编号错误 非法的DB编号
B#16#34 B#16#35 B#16#3A B#16#3C B#16#3D B#16#3E B#16#3F OB121_FLT_REG	调用FC时的FC编号错误 调用FB时的FB编号错误 访问未下载的DB，DB编号在允许范围 访问未下载的FC，FC编号在允许范围 访问未下载的SFC，SFC编号在允许范围 访问未下载的FB，FB编号在允许范围 访问未下载的SFB，SFB编号在允许范围 非法的编号

3. I/O访问错误组织块(OB122)

STEP 7指令访问有故障的模块，例如直接访问I/O错误(模块损坏或找不到)，或者访问了一个CPU不能识别的I/O地址，此时CPU的操作系统将会调用OB122。OB122的变量声明如表6-19所示。

表6-19 OB122的变量声明表

参数	数据类型	描述
OB122_EV_CLASS	BYTE	错误级别与标识：B#16#29
OB122_SW_FLT	BYTE	错误代码，见本表下面的说明 B#16#42：S7-300和CPU 417的I/O读访问错误，其他S7-400 CPU是错误出现后首次读访问错误 B#16#43：S7-300和CPU 417的I/O写访问错误，其他S7-400 CPU是错误出现后首次写访问错误 B#16#44仅用于S7-400(不包括CPU 417)，错误出现之后的第n次(n>1)读访问错误 B#16#45仅用于S7-400(不包括CPU 417)，错误出现之后的第n次(n>1)写访问错误
OB122_PRIORITY	BYTE	优先级，与出现错误的OB的优先级相同
OB122_OB_NUMBR	BYTE	OB号(122)
OB122_BLK_TYPE	BYTE	S7-400出错的块类型：B#16#88：OB，B#16#8A：DB，B#16#8C：FC，B#16#8E：FB
OB122_MEM_AREA	BYTE	存储区与访问类型：4～7位为0～3时，分别表示访问位、字节、字或双字； 0～3位为0～2时，对应的存储区为I/O区、过程映像输入或过程映像输出
OB122_MEM_ADDR	WORD	出现错误的存储器地址
OB122_BLK_NUM	WORD	引起错误的MC7命令的块的编号，S7-300未用
OB122_PRG_ADDR	WORD	引起错误的MC7命令的相对地址，S7-300未用
OB122_DATE_TIME	DATE_AND_TIME	OB被调用时的日期和时间

错误代码B#16#44和B#16#45表示错误相当严重，例如可能是因为访问的模块不存在，导致多次访问出错，这时应采取停机的措施。

对于某些同步错误，可以调用系统功能SFC 44，为输入模块提供一个替代值来代替错误

值，以便使程序能继续执行。如果错误发生在输入模块，可以在用户程序中直接替代。如果是输出模块错误，输出模块将自动地用组态时定义的值替代。替代值虽然不一定能反映真实的过程信号，但是可以避免终止用户程序和进入 STOP 模式。

【例 6-6】 同步错误组织块举例。

建立一个名为“OB121 例程”的项目，生成 FC1 和 FC2。FC2 中是一段错误的指令（超出了定时器的地址范围）：

```
A     T600
=     M2.0
```

OB1 无条件调用 FC1，FC1 在 I0.0 为 1 时调用 FC2。用仿真软件模拟运行程序。I0.0 为 0 时程序正常运行，令 I0.0 为 1，程序调用有错误的 FC2，CPU 视图对象上的红色 SF 灯亮，绿色的 RUN 灯熄灭，橙色的 STOP 灯亮，PLC 切换到 STOP 状态。

在 SIMATIC 管理器中执行菜单命令“PLC”→“Diagnostics/Settings”→“Module Information”，打开如图 4-25 所示的模块信息对话框，选中诊断缓冲区选项卡，可以看到红色的错误标志。点击【Help】按钮可以得到有关的帮助信息。

诊断缓冲区的第 1 条是最新的信息，选中图中的第 2 条信息，下面的“Details on”窗口指出停机原因的详细信息：因为没有下载处理错误的 OB，程序在 OB1 中断。选中第 3 条信息，可以看到在 FC2 发生了定时器编号错误，请求调用 OB121。点击对话框中的【Open Block】按钮，将会打开出错的块 FC2。选中对话框中的“Stacks”选项卡，在块堆栈中可以看到从上到下排列着 OB1、FC1 和 FC2，表示出错时程序的调用路径。点击该选项卡中的【I Stack】按钮，打开中断堆栈，可以看到发生中断时累加器、地址寄存器和状态字的内容，在“Point of Interruption”区可以查到断点的位置。

返回 SIMATIC 管理器，生成 OB121（可以是一个空的块），下载后重新运行，可以看到用 I0.0 调用 FC2 时不会停机，但是 SF 灯会亮。

6.5.10 背景组织块

CPU 可以保证设置的最小扫描循环时间，如果它比实际的扫描循环时间长，在循环程序结束后 CPU 处于空闲的时间内可以执行背景组织块（OB90）。如果没有对 OB90 编程，CPU 要等到定义的最小扫描循环时间到达为止，再开始下一次循环的操作。用户可以将对运行时间要求不高的操作放在 OB90 中去执行，以避免出现等待时间。

背景 OB 的优先级为 29（最低），不能通过参数设置进行修改。OB90 可以被所有其他的系统功能和任务中断。

由于 OB90 的运行时间不受 CPU 操作系统的监视，用户可以在 OB90 中编写长度不受限制的程序。

第 7 章　计算机通信网络与 S7-300/400 的通信功能

7.1　计算机通信方式与串行通信接口

近年来，计算机控制已被迅速地推广和普及，相当多的企业已经在大量地使用各式各样的可编程设备，例如工业控制计算机、PLC、变频器、机器人、数控机床、柔性制造系统等。有的企业已实现了全车间或全厂的综合自动化，即将不同厂家生产的可编程设备连接在单层或多层网络上，相互之间进行数据通信，实现分散控制和集中管理。因此通信与网络已经成为控制系统不可缺少的重要组成部分，也是控制系统的设计和维护的重点和难点之一，本章首先介绍有关数字通信与工厂自动化通信网络及其国际标准的知识，然后介绍 S7-300/400 系列 PLC 的通信功能，最后介绍 MPI 网络与全局数据通信、执行器传感器接口 AS－i 网络和工业以太网。

7.1.1　计算机的通信方式

1. 并行通信与串行通信

并行数据通信是以字节或字为单位的数据传输方式，除了 8 根或 16 根数据线、一根公共线外，还需要通信双方联络用的控制线。并行通信的传送速度快，但是传输线的根数多，抗干扰能力较差，一般用于近距离数据传送，例如 PLC 的模块之间的数据传送。

串行数据通信是以二进制的位(bit)为单位的数据传输方式，每次只传送一位，最少只需要两根线(双绞线)就可以连接多台设备，组成控制网络。串行通信需要的信号线少，适用于距离较远的场合。计算机和 PLC 都有通用的串行通信接口，例如 RS-232C 或 RS-485 接口，工业控制中计算机之间的通信一般采用串行通信方式。

2. 异步通信与同步通信

在串行通信中，接收方和发送方应使用相同的传输速率。接收方和发送方的标称传输速率虽然相同，它们之间总是有一些微小的差别。如果不采取措施，在连续传送大量的信息时，将会因积累误差造成发送和接收的数据错位，使接收方收到错误的信息。为了解决这一问题，需要使发送过程和接收过程同步。按同步方式的不同，串行通信可以分为异步通信和同步通信。

异步通信的字符信息格式如图 7-1 所示，发送的字符由一个起始位、7～8 个数据位、1 个奇偶校验位(可以没有)和停止位(1 位或两位)组成。通信双方需要对采用的信息格式和数据的传输速率作相同的约定。接收方检测到停止位和起始位之间的下降沿后，将它作为接收的起始点，在每一位的中点接收信息。由于一个字符中包含的位数不多，即使发送方和接收方的收发频率略有不同，也不会因为两台设备之间的时钟周期的积累误差而导致信息的发送和接收错位。异步通信的缺点是传送附加的非有效信息

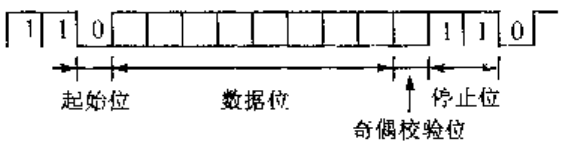


图 7-1　异步通信的信息格式

较多，传输效率较低，但是随着通信速率的提高，可以满足控制系统通信的要求，PLC 一般采用异步通信。

奇偶校验用来检测接收到的数据是否出错。如果指定的是奇校验，发送方发送的每一个字符的数据位和奇偶校验位中“1”的个数为奇数，接收方对接收到的每一个字符的奇偶性进行校验，可以检验出传送过程中的错误。例如某字符中包含以下 8 个数据位：

1 0 1 0 0 0 1 1

其中“1”的个数是 4 个。如果选择了偶校验，奇偶校验位将是 0，使“1”的个数仍然是 4 个。如果选择了奇校验，奇偶校验位将是 1，使“1”的个数是 5 个。如果选择不进行奇偶校验，传输时没有校验位，也不进行奇偶校验检测。

同步通信以字节为单位，一个字节由 8 位二进制数组成。每次传送 1～2 个同步字符、若干个数据字节和校验字符。同步字符起联络作用，用它来通知接收方开始接收数据。在同步通信中，发送方和接收方应保持完全的同步，这意味着发送方和接收方应使用同一个时钟脉冲，可以通过调制解调的方式在数据流中提取出同步信号，使接收方得到与发送方同步的接收时钟信号。

由于同步通信方式不需要在每个数据字符中增加起始位、停止位和奇偶校验位，只需要在要发送的数据之前加一两个同步字符，所以传输效率高，但是对硬件的要求较高。

3. 单工与双工通信

单工通信方式只能沿单一方向传输数据，双工通信方式的信息可以沿两个方向传送，每一个站既可以发送数据，也可以接收数据。双工方式又分为全双工和半双工。

全双工方式中数据的发送和接收分别用两组不同的数据线传送，通信的双方都能在同一时刻接收和发送信息（见图 7-2）。半双工方式用同一组线接收和发送数据，通信的双方在同一时刻只能发送数据或接收数据（见图 7-3）。

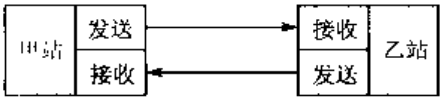


图 7-2　全双工方式

图 7-3　半双工方式

4. 传输速率

在串行通信中，传输速率（又称波特率）的单位是波特，即每秒传送的二进制位数，其符号为 bit/s。常用的传输速率为 300～38 400 bit/s，从 300 开始成倍数增加。不同的串行通信网络的传输速率差别极大，有的只有数百 bit/s，高速串行通信网络的传输速率可达 1 G bit/s。

7.1.2　串行通信接口的标准

1. RS-232C

RS-232C 是美国 EIC（电子工业联合会）在 1969 年公布的通信协议，至今仍在计算机和控制设备通信中广泛使用。这个标准对串行通信接口有关的问题，例如各信号线的功能和电气特性等都作了明确的规定。

RS-232C 标准最初是为远程通信连接数据终端设备 DTE（Data Terminal Equipment）与数据通信设备 DCE（Data Communication Equipment）制定的，因此这个标准的制定并未考虑计算机系统的应用要求，但是它实际上广泛地用于计算机与终端或外设之间的近距离通信。

RS-232C 一般使用 9 针和 25 针 DB 型连接器，工业控制中 9 针连接器用得较多。

当通信距离较近时，通信双方可以直接连接，最简单的情况在通信中不需要控制联络信号，只需要三根线（发送线、接收线和信号地线，见图 7-4）便可以实现全双工异步串行通信。RS-232C 采用负逻辑，用 −15～−5 V 表示逻辑状态“1”，用 +5～+15 V 表示逻辑状态“0”，最大通信距离为 15 m，最高传输速度速率为 20 kbit/s，只能进行一对一的通信。

RS-232C 使用单端驱动、单端接收电路（见图 7-5），是一种共地的传输方式，容易受到公共地线上的电位差和外部引入的干扰信号的影响。

2. RS-422A 与 RS-485

RS-422A 采用平衡驱动、差分接收电路（见图 7-6），从根本上取消了信号地线。平衡驱动器相当于两个单端驱动器，其输入信号相同，两个输出信号互为反相信号，图中的小圆圈表示反相。外部输入的干扰信号是以共模方式出现的，两根传输线上的共模干扰信号相同，因接收器是差分输入，共模信号可以互相抵消。只要接收器有足够的抗共模干扰能力，就能从干扰信号中识别出驱动器输出的有用信号，从而克服外部干扰的影响。

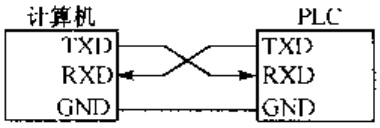


图 7-4　RS-232 的信号线连接

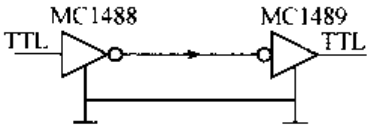


图 7-5　单端驱动单端接收

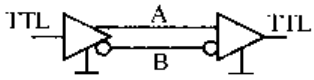


图 7-6　平衡驱动差分接收

RS-422A 在最大传输速率（10 Mbit/s）时，允许的最大通信距离为 12m。传输速率为 100 kbit/s 时，最大通信距离为 1 200 m，一台驱动器可以连接 10 台接收器。

在 RS-422A 模式，数据通过 4 根导线传送（四线操作）。RS-422A 是全双工，两对平衡差分信号线分别用于发送和接收。

3. RS-485

RS-485 是 RS-422A 的变形，RS-485 为半双工，只有一对平衡差分信号线，不能同时发送和接收。使用 RS-485 通信接口和双绞线可以组成串行通信网络（见图 7-8），构成分布式系统，系统中最多可以有 32 个站，新的接口器件已允许连接 128 个站。

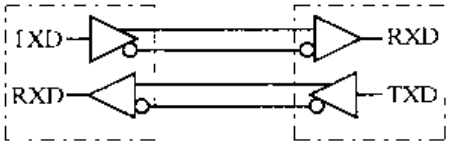


图 7-7　RS-422A 通信接线图

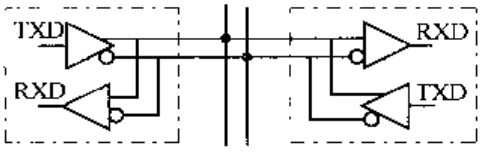


图 7-8　RS-485 网络

7.2　计算机通信的国际标准

7.2.1　开放系统互连模型

如果没有一套通用的计算机网络通信标准，要实现不同厂家生产的智能设备之间的通信，将会付出昂贵的代价。

国际标准化组织 ISO 提出了开放系统互连模型 OSI，作为通信网络国际标准化的参考模型，它详细描述了软件功能的 7 个层次(见图 7-9)。

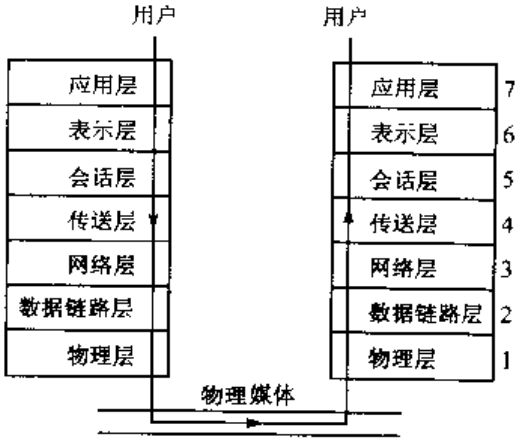


图 7-9　开放系统互连模型

7 层模型分为两类，一类是面向用户的第 5～7 层，另一类是面向网络的第 1～4 层。前者给用户提供适当的方式去访问网络系统，后者描述数据怎样从一个地方传输到另一个地方。

1. 物理层

物理层的下面是物理媒体，例如双绞线、同轴电缆等。物理层为用户提供建立、保持和断开物理连接的功能，RS-232C、RS-422A、RS-485 等就是物理层标准的例子。

2. 数据链路层

数据以帧(Frame)为单位传送，每一帧包含一定数量的数据和必要的控制信息，例如同步信息、地址信息、差错控制和流量控制信息。数据链路层负责在两个相邻节点间的链路上，实现差错控制、数据成帧、同步控制等。

3. 网络层

网络层的主要功能是报文包的分段、报文包阻塞的处理和通信子网中路径的选择。

4. 传输层

传输层的信息传送单位是报文(Message)，它的主要功能是流量控制、差错控制、连接支持，传输层向上一层提供一个可靠的端到端(end-to-end)的数据传送服务。

5. 会话层

会话层的功能是支持通信管理和实现最终用户应用进程之间的同步，按正确的顺序收发数据，进行各种对话。

6. 表示层

表示层用于应用层信息内容的形式变换，例如数据加密/解密、信息压缩/解压和数据兼容，把应用层提供的信息变成能够共同理解的形式。

7. 应用层

应用层作为 OSI 的最高层，为用户的应用服务提供信息交换，为应用接口提供操作标准。

不是所有的通信协议都需要 OSI 参考模型中的全部 7 层，例如有的现场总线通信协议只采用了 7 层协议中的第 1、第 2 和第 7 层。

7.2.2　IEEE 802 通信标准

IEEE(国际电工与电子工程师学会)的 802 委员会于 1982 年颁布了一系列计算机局域网分层通信协议标准草案，总称为 IEEE802 标准。它把 OSI 参考模型的底部两层分解为逻辑链路控制层(LLC)、媒体访问层(MAC)和物理传输层。前两层对应于 OSI 参考模型中的数据链路层，数据链路层是一条链路(Link)两端的两台设备进行通信时所共同遵守的规则和约定。

IEEE802 的媒体访问控制层对应于三种已建立的标准，即带冲突检测的载波侦听多路访问(CSMA/CD)协议、令牌总线(Token Bus)和令牌环(Token Ring)。

1. CSMA/CD

CSMA/CD 通信协议的基础是 XEROX 公司研制的以太网(Ether net),各站共享一条广播式的传输总线,每个站都是平等的,采用竞争方式发送信息到传输线上,也就是说,任何一个站都可以随时广播报文,并为其他各站接收。当某个站识别到报文上的接收站名与本站的站名相同时,便将报文接收下来。由于没有专门的控制站,两个或多个站可能因同时发送信息而发生冲突,造成报文作废,因此必须采取措施来防止冲突。

发送站在发送报文之前,先监听一下总线是否空闲,如果空闲,则发送报文到总线上,称之为"先听后讲"。但是这样做仍然有发生冲突的可能,因为从组织报文到报文在总线上传输需要一段时间,在这一段时间中,另一个站通过监听也可能会认为总线空闲并发送报文到总线上,这样就会因两站同时发送而发生冲突。

为了防止冲突,可以采取两种措施:一种是发送报文开始的一段时间,仍然监听总线,采用边发送边接收的办法,把接收到的信息和自己发送的信息相比较,若相同则继续发送,称之为"边听边讲";若不相同则发生冲突,立即停止发送报文,并发送一段简短的冲突标志(阻塞码序列)。通常把这种"先听后讲"和"边听边讲"相结合的方法称为 CSMA/CD(带冲突检测的载波侦听多路访问技术),其控制策略是竞争发送、广播式传送、载体监听、冲突检测、冲突后退和再试发送。

另一种措施是准备发送报文的站先监听一段时间(大约是总线传输延时的 2 倍),如果在这段时间中总线一直空闲,则开始作发送准备,准备完毕,真正要将报文发送到总线之前,再对总线作一次短暂的检测,若仍为空闲,则正式开始发送;若不空闲,则延时一段时间后再重复上述的二次检测过程。

CSMA/CD 允许各站平等竞争,实时性好,适合于工业自动控制计算机网络。

以太网首先在个人计算机网络系统,例如办公自动化系统和管理信息系统(MIS)中得到了极为广泛的应用,以太网的硬件(例如网卡和集线器)非常便宜。在以太网发展的初期,通信速率较低。如果网络中的设备较多,信息交换比较频繁,可能会经常出现竞争和冲突,影响信息传输的实时性。随着以太网传输速率的提高(100~1000 M bit/s),这一问题已经解决,现在以太网在工业控制中也得到了广泛的应用,大型工业控制系统中最上层的网络几乎全部采用以太网。使用以太网很容易实现管理网络和控制网络的一体化。

2. 令牌总线

IEEE802 标准中的工厂媒质访问技术是令牌总线,其编号为 802.4。它吸收了通用汽车公司支持的 MAP(Manufacturing Automation Protocol,制造自动化协议)系统的内容。

在令牌总线中,媒体访问控制是通过传递一种称为令牌的特殊标志来实现的。按照逻辑顺序,令牌从一个装置传递到另一个装置,传递到最后一个装置后,再传递给第一个装置,如此周而复始,形成一个逻辑环。令牌有"空"、"忙"两个状态,令牌网开始运行时,由指定站产生一个空令牌沿逻辑环传送。任何一个要发送信息的站都要等到令牌传给自己,判断为空令牌时才能发送信息。发送站首先把令牌置成"忙",并写入要传送的信息、发送站名和接收站名,然后将载有信息的令牌送入环网传输。令牌沿环网循环一周后返回发送站时,信息已被接收站拷贝,发送站将令牌置为"空",送上环网继续传送,以供其他站使用。

如果在传送过程中令牌丢失,由监控站向网内注入一个新的令牌。

令牌传递式总线能在很重的负荷下提供实时同步操作,传送效率高,适于频繁、较短的数

据传送，因此它最适合于需要进行实时通信的工业控制网络系统。

3. 令牌环

令牌环媒质访问方案是 IBM 公司开发的，它在 IEEE 802 标准中的编号为 802.5，它有些类似于令牌总线。在令牌环上，最多只能有一个令牌绕环运动，不允许两个站同时发送数据。令牌环从本质上看是一种集中控制式的环，环上必须有一个中心控制站负责网的工作状态的检测和管理。

7.2.3 现场总线及其国际标准

IEC（国际电工委员会）对现场总线（Fieldbus）的定义是"安装在制造和过程区域的现场装置与控制室内的自动控制装置之间的数字式、串行、多点通信的数据总线称为现场总线"。它是当前工业自动化的热点之一。现场总线以开放的、独立的、全数字化的双向多变量通信代替 0～10 mA 或 4～20 mA 现场电动仪表信号。现场总线 I/O 集检测、数据处理、通信为一体，可以代替变送器、调节器、记录仪等模拟仪表，它不需要框架、机柜，可以直接安装在现场导轨槽上。现场总线 I/O 的接线极为简单，只需一根电缆，从主机开始，沿数据链从一个现场总线 I/O 连接到下一个现场总线 I/O。使用现场总线后，自控系统的配线、安装、调试和维护等方面的费用可以节约三分之二左右，现场总线 I/O 与 PLC 可以组成廉价的 DCS 系统。

使用现场总线后，操作员可以在中央控制室实现远程监控，对现场设备进行参数调整，还可以通过现场设备的自诊断功能预测故障和寻找故障点。

由于历史的原因，现在有多种现场总线标准并存，IEC 的现场总线国际标准（IEC 61158）是迄今为止制订时间最长、意见分歧最大的国际标准之一。它的制订时间超过 12 年，先后经过 9 次投票，在 1999 年底获得通过。经过多方的争执和妥协，最后容纳了 8 种互不兼容的协议，这 8 种协议在 IEC61158 中分别为 8 种现场总线类型：

类型 1：原 IEC61158 技术报告，即现场总线基金会（FF）的 H1；

类型 2：Control Net（美国 Rockwell 公司支持）；

类型 3：PROFIBUS（德国西门子公司支持）；

类型 4：P-Net（丹麦 Process Data 公司支持）；

类型 5：FF 的 HSE（原 FF 的 H2，高速以太网，美国 Fisher Rosemount 公司支持）；

类型 6：Swift Net（美国波音公司支持）；

类型 7：WorldFIP（法国 Alstom 公司支持）；

类型 8：Interbus（德国 Phoenix contact 公司支持）。

各类型将自己的行规纳入 IEC 61158，且遵循两个原则：

（1）不改变 IEC 61158 技术报告的内容。

（2）不改变各行规的技术内容，各组织按 IEC 技术报告（类型 1）的框架组织各自的行规，并提供对类型 1 的网关或链接器。用户在使用各种类型时仍需使用各自的行规。因此 IEC 61158 标准不能完全代替各行规，除非今后出现完整的现场总线标准。

IEC 标准的八种类型都是平等的，类型 2～8 都对类型 1 提供接口，标准并不要求类型 2～8 之间提供接口。

IEC 62026 是供低压开关设备与控制设备使用的控制器电气接口标准，于 2000 年 6 月通过。它包括：

(1) IEC 62026-1:一般要求;
(2) IEC 62026-2:执行器传感器接口 AS-i (Actuator Sensor Interface);
(3) IEC 62026-3:设备网络 DN (Device Network);
(4) IEC 62026-4:Lonworks(Local Operating Networks)总线的通信协议 LonTalk;
(5) IEC 62026-5:灵巧配电(智能分布式)系统 SDS (Smart Distributed System);
(6) IEC 62026-6:串行多路控制总线 SMCB(Serial Multiplexed Control Bus)。

7.3 S7-300/400 的通信功能

一个典型的工厂自动化系统一般是三级网络结构:

1. 现场设备层

现场设备层的主要功能是连接现场设备,例如分布式 I/O、传感器、驱动器、执行机构和开关设备等,完成现场设备控制及设备间连锁控制。主站(PLC、PC 或其他控制器)负责总线通信管理及与从站的通信。总线上所有设备生产工艺控制程序存储在主站中,并由主站执行。

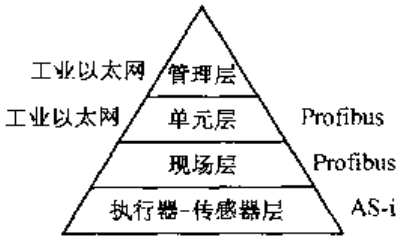


图 7-10 SIMASTIC NET

西门子的 SIMATIC NET 网络系统(见图 7-10)将执行器和传感器单独分为一层,主要使用 AS-i(执行器-传感器接口)网络。

AS-i 是国家标准 GB/T 18858.2-2002/IEC 62026-2:2000 低压开关设备和控制设备控制器——设备接口(CDI)的第 2 部分:执行器与传感器接口(AS-i)。

2. 车间监控层

车间监控层又称为单元层,用来完成车间主生产设备之间的连接,实现车间级设备的监控。车间级监控包括生产设备状态的在线监控、设备故障报警及维护等。通常还具有诸如生产统计、生产调度等车间级生产管理功能。车间级监控通常要设立车间监控室,有操作员工作站及打印设备。车间级监控网络可采用 PROFIBUS-FMS 或工业以太网,PROFIBUS-FMS 是一个多主网络,这一级数据传输速度不是最重要的,但是应能传送大容量的信息。

3. 工厂管理层

车间操作员工作站可以通过集线器与车间办公管理网连接,将车间生产数据送到车间管理层。车间管理网作为工厂主网的一个子网,通过交换机、网桥或路由器等连接到厂区骨干网,将车间数据集成到工厂管理层。

工厂管理层通常采用符合 IEC802.3 标准的以太网,即 TCP/IP 通信协议标准。厂区骨干网可以根据工厂实际情况,采用 FDDI 或 ATM 等网络。

S7-300/400 有很强的通信功能,CPU 模块集成有 MPI 和 DP 通信接口,有 PROFIBUS-DP 和工业以太网的通信模块,以及点对点通信模块。通过 PROFIBUS-DP 或 AS-i 现场总线,CPU 与分布式 I/O 模块之间可以周期性地自动交换数据(过程映像数据交换)。在自动化系统之间,PLC 与计算机和 HMI(人机接口)站之间,均可以交换数据。数据通信可以周期性地自动进行,或基于事件驱动(由用户程序块调用)。

S7/C7 通信对象的通信服务通过集成在系统中的功能块来进行。可以提供的通信服务有:

(1) 使用 MPI 的标准 S7 通信；

(2) 使用 MPI，K 总线，PROFIBUS-DP 和工业以太网的 S7 通信(S7-300 只能作服务器)；

(3) 与 S5 通信对象和第三方设备的通信，可以用非常驻的块来建立。这些服务包括通过 FROFIBUS-DP 和工业以太网的 S5 兼容通信和标准通信。

7.3.1 S7-300/400 的通信网络

1. 通过多点接口(MPI)协议的数据通信

MPI 是多点接口(Multi Point Interface)的简称，S7-300/400 CPU 都集成了 MPI 通信协议，MPI 的物理层是 RS-485，最大传输速率为 12M bit/s。PLC 通过 MPI 能同时连接运行 STEP 7 的编程器、计算机、人机界面(HMI)及其他 SIMATIC S7、M7 和 C7。这是一种经济而有效的解决方案。STEP 7 的用户界面提供了通信组态功能，使得通信的组态非常简单。

联网的 CPU 可以通过 MPI 接口实现全局数据(GD)服务，周期性地相互进行数据交换。

每个 CPU 可以使用的 MPI 连接总数与 CPU 的型号有关，为 6～64 个。

S7-300/400 的通信网络如图 7-11 所示。

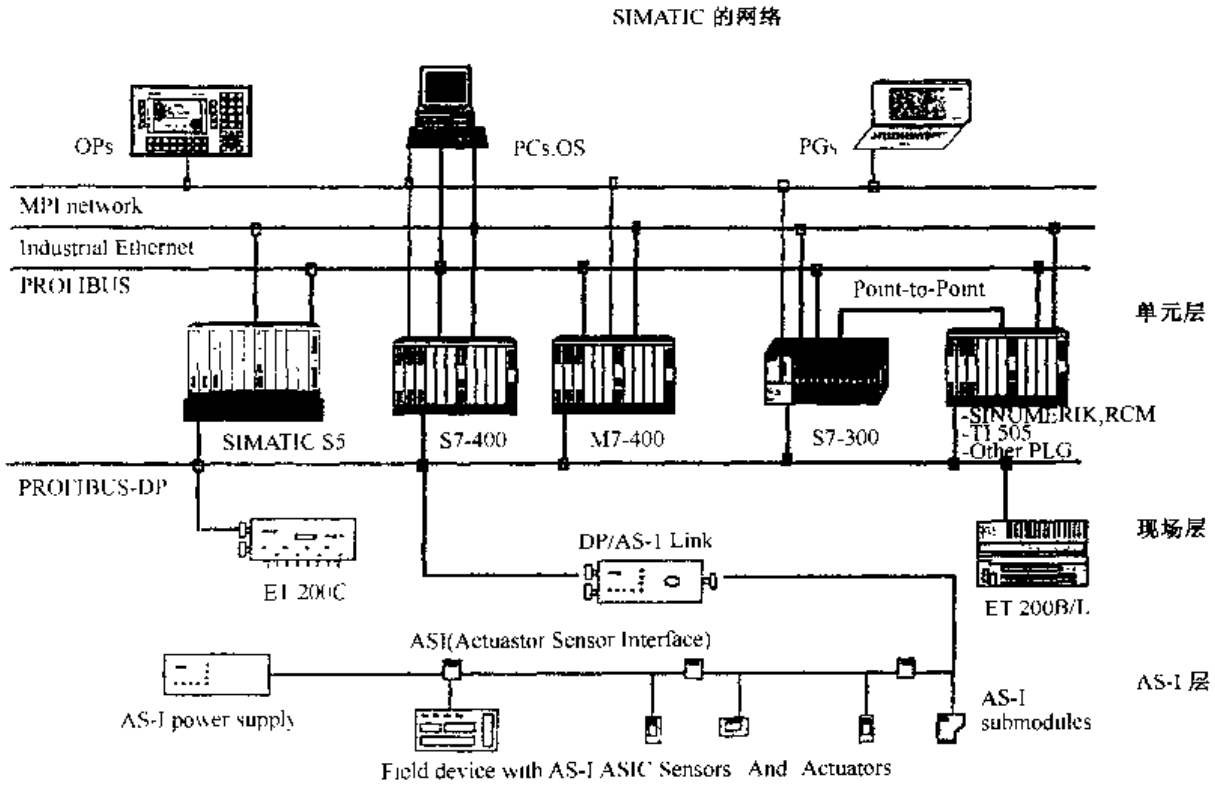


图 7-11 S7-300/400 的通信网络

2. PROFIBUS

工业现场总线 PROFIBUS 是用于车间级监控和现场层的通信系统，它符合 IEC 61158 标准(是该标准中的类型 3)，具有开放性，符合该标准的各厂商生产的设备都可以接入同一网络中。S7-300/400 PLC 可以通过通信处理器或集成在 CPU 上的 PROFIBUS-DP 接口连接到 PROFIBUS-DP 网络上。

带有 PROFIBUS-DP 主站/从站接口的 CPU 能够实现高速和使用方便的分布式 I/O 控制

(见图 7-12)。对于用户来说,处理分布式 I/O 就像处理集中式 I/O 一样,系统组态和编程的方法完全相同。

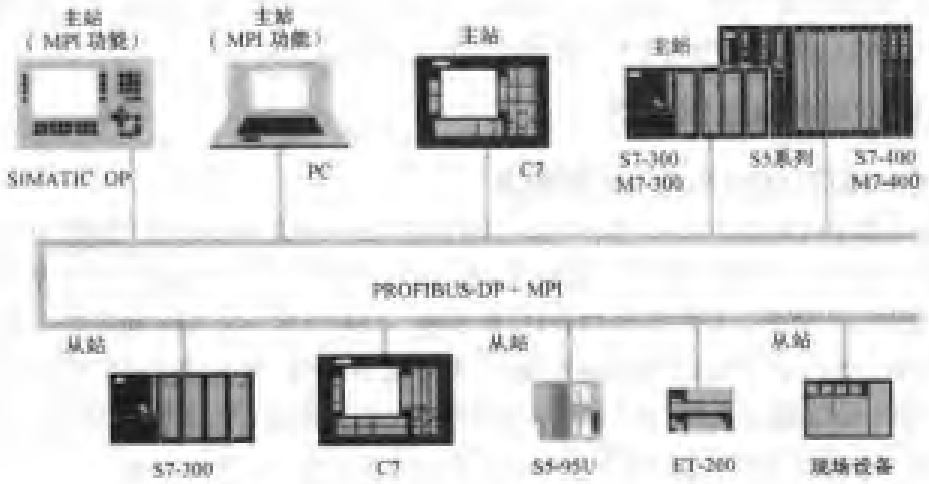


图 7-12 PROFIBUS-DP 通信网络

PROFIBUS 的物理层是 RS-485,最大传输速率 12 M bit/s,最多可以与 127 个网络上的节点进行数据交换。网络中最多可以串接 10 个中继器来延长通信距离。使用光纤作通信介质,通信距离可达 90 km。

如果 PROFIBUS 网络采用 FMS 协议,工业以太网采用 TCP/IP 或 ISO 协议,S7-300 可以与其他公司的设备实现数据交换。

可以通过 CP 342/343 通信处理器将 SIMATIC S7-300 与 PROFIBUS-DP 或工业以太网总线系统相连。可以连接的设备包括 S7-300/400、S5-115U/H、编程器、个人计算机、SIMATIC 人机界面(HMI)、数控系统、机械手控制系统、工业 PC 机、变频器和非西门子装置。

下列设备可以作为主站:带有 PROFIBUS-DP 接口的 S7-300/400 的 CPU,CP 443-5 和 IM467;CP 342-5 或 CP 343-5;带有 DP 接口或 DP 通信处理器的 C7;以及西门子某些老型号的 PLC,编程器 PG 和操作员面板 OP。

下列设备可以作为从站:分布式 I/O 设备 ET200B/L/M/S/X;通过通信处理器 CP342-5 的 S7-300,带 DP 接口的 S7-300 CPU,S7-400(只能通过 CP 443-5),C7-633/P DP,C7-633DP,C7-634/P DP,C7-634DP,C7-626DP,带 EM277 通信模块的 S7-200。

3. 工业以太网

工业以太网(Industrial Ethernet)是用于工厂管理和单元层的通信系统,符合 IEEE 802.3 国际标准,用于对时间要求不太严格,需要传送大量数据的通信场合,可以通过网关来连接远程网络。它支持广域的开放型网络模型,可以采用多种传输媒体。西门子的工业以太网的传输速率为 10 M/100 M bit/s,最多 1024 个网络节点,网络的最大范围为 150 km。

西门子的 S7 和 S5 这两代 PLC 通过 PROFIBUS(FDL 协议)或工业以太网 ISO 协议,可以利用 S5 和 S7 的通信服务进行数据交换。

CP 通信处理器不会加重 CPU 的通信服务负担,S7-300 最多可以使用 8 个通信处理器,每个通信处理器最多能建立 16 条链路。

4. 点对点连接

点对点连接(Point-to-Point Connections)可以连接两台 S7 PLC 和 S5 PLC,以及计算机、打印机、机器人控制系统、扫描仪和条码阅读器等非西门子设备。使用 CP 340,CP 341 和 CP 441 通信处理模块,或通过 CPU 313C-2PtP 和 CPU 314C-2PtP 集成的通信接口,可以建立起经济而方便的点对点连接。

点对点通信可以提供的接口有 20 mA(TTY),RS-232C 和 RS-422A/RS-485。点对点通信可以使用的通信协议有 ASCII 驱动器、3964(R)和 RK 512(只适用于部分 CPU)。

全双工模式(RS-232C)的最高传输速率为 19.2 kbit/s,半双工模式(RS-485)的最高传输速率为 38.4 kbit/s。

使用西门子的通信软件 PRODAVE 和编程用的 PC/MPI 适配器,通过 PLC 的 MPI 编程接口,可以很方便地实现计算机与 S7-300/400 的通信。

5. 通过 AS-i 的过程通信

执行器-传感器接口(Actuator-Sensor-Interface)简称 AS-i,是位于自动控制系统最底层的网络,用来连接有 AS-i 接口的现场二进制设备,只能传送少量的数据,例如开关的状态。

CP 342-2 通信处理器是用于 S7-300 和分布式 I/O ET 200M 的 AS-i 主站,它最多可以连接 62 个数字量或 31 个模拟量 AS-i 从站。通过 AS-i 接口,每个 CP 最多可以访问 248 个数字量输入和 186 个数字量输出。通过内部集成的模拟量处理程序,可以像处理数字量值那样非常容易地处理模拟量值。

7.3.2 S7 通信的分类

S7 通信可以分为全局数据通信、基本通信及扩展通信 3 类,如图 7-13 所示。

1. 全局数据通信

全局数据(GD)通信通过 MPI 接口在 CPU 间循环交换数据,用全局数据表来设置各 CPU 之间需要交换的数据存放的地址区和通信的速率,通信是自动实现的,不需要用户编程。当过程映像被刷新时,在循环扫描检测点进行数据交换。S7-400 的全局数据通信可以用 SFC 来起动。全局数据可以是输入、输出、标志位(M)、定时器、计数器和数据区。

S7-300 CPU 每次最多可以交换 4 个包含 22 B 的数据包,最多可以有 16 个 CPU 参与数据交换。

S7-400 CPU 可以同时建立最多 64 个站的连接,MPI 网络最多 32 个节点。任意两个 MPI 节点之间可以串联 10 个中继器,以增加通信的距离。每次程序循环最多 64 B,最多 16 个 GD 数据包。在 CR2 机架中,两个 CPU 可以通过 K 总线用 GD 数据包进行通信。

通过全局数据通信,一个 CPU 可以访问另一个 CPU 的数据块、存储器位和过程映像等。全局通信用 STEP 7 中的 GD 表进行组态。对 S7、M7 和 C7 的通信服务可以用系统功能块来建立。

MPI 默认的传输速率为 187.5 kbit/s,与 S7-200 通信时只能指定 19.2 kbit/s 的传输速率。通过 MPI 接口,CPU 可以自动广播其总线参数组态(例如波特率)。然后 CPU 可以自动检索正确的参数,并连接至一个 MPI 子网。

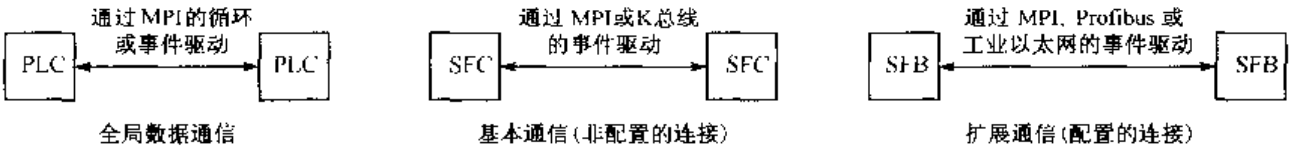


图 7-13　S7 通信的分类

2. **基本通信(非配置的连接)**

这种通信可以用于所有的 S7-300/400 CPU,通过 MPI 或站内的 K 总线(通信总线)来传送最多 76 B 的数据。在用户程序中用系统功能(SFC)来传送数据。在调用 SFC 时,通信连接被动态地建立,CPU 需要一个自由的连接。

3. **扩展通信(配置的通信)**

这种通信可以用于所有的 S7-300/400 CPU,通过 MPI,PROFIBUS 和工业以太网最多可以传送 64 KB 的数据。通信是通过系统功能块(SFB)来实现的,支持有应答的通信。在 S7-300 中可以用 SFB 15"PUT"和 SFB 14"GET"来写出或读入远端 CPU 的数据。

扩展的通信功能还能执行控制功能,例如控制通信对象的起动和停机。这种通信方式需要用连接表配置连接,被配置的连接在站起动时建立并一直保持。

7.4　MPI 网络与全局数据通信

7.4.1　MPI 网络

MPI 是多点接口(Multi Point Interface)的简称,每个 S7-300/400 CPU 都集成了多点接口通信协议,MPI 的物理层是 RS-485。通过 MPI,PLC 可以同时与多个设备建立通信连接,可以连接的设备包括编程器或运行 STEP 7 的计算机、人机界面(HMI)及其他 SIMATIC S7,M7 和 C7。同时连接的通信对象的个数与 CPU 的型号有关,例如 CPU 312 为 6 个,CPU 418 为 64 个。

西门子有两种硬件 MPI 连接器,一种带有 PG(编程器)接口,一种没有 PG 接口。在计算机上应插一块 MPI 卡,或使用 PC/MPI 适配器。位于网络终端的站,应将其连接器上的"终端电阻"开关合上,以接入终端电阻。

通过 MPI 可以访问 PLC 所有的智能模块,例如功能模块。这是一种经济而有效的解决方案。STEP 7 的用户界面提供了全局数据通信组态功能,使得通信的组态非常简单。

联网的 CPU 可以通过 MPI 接口实现全局数据(GD)服务,周期性地相互交换少量的数据。最多可以与在一个项目中的 15 个 CPU 之间建立全局数据通信。

每个 MPI 节点都有自己的 MPI 地址(0～126),编程设备、人机接口和 CPU 的默认地址分别为 0、1、2。

在 S7-300 中,MPI 总线在 PLC 中与 K 总线(通信总线)连接在一起,S7-300 机架上 K 总线的每一个节点(功能模块 FM 和通信处理器 CP)也是 MPI 的一个节点,有自己的 MPI 地址。

在 S7-400 中,MPI(187.5 k bit/s)通信模式被转换为内部 K 总线(10.5 Mbit/s)。S7-400 只有 CPU 有 MPI 地址,其他智能模块没有独立的 MPI 地址。

通过全局数据通信,一个 CPU 可以访问另一个 CPU 的位存储器、输入/输出映像区、定时

器、计数器和数据块中的数据。对 S7、M7 和 C7 的通信服务可以用系统功能块来建立。

MPI 默认的传输速率为 187.5 k bit/s 或 1.5 M bit/s，与 S7-200 通信时只能指定 19.2 k bit/s。两个相邻节点间的最大传送距离为 50 m，加中继器后为 1000 m，使用光纤和星形连接时为 23.8 km。

通过 MPI 接口，CPU 可以自动广播其总线参数组态(例如波特率)。然后 CPU 可以自动检索正确的参数，并连接至一个 MPI 子网。

7.4.2 全局数据包

参与全局数据包交换的 CPU 构成了全局数据环(GD circle)。同一个 GD 环中的 CPU 可以向环中其他的 CPU 发送数据或接收数据。在一个 MPI 网络中，可以建立多个 GD 环。

具有相同的发送者和接收者的全局数据可以集合成一个全局数据包(GD Packet)。每个数据包有数据包编号，数据包中的变量有变量号。例如 GD1.2.3 是 1 号 GD 环、2 号 GD 包中的 3 号数据。

S7-300 CPU 可以发送和接收的 GD 包的个数(4 个或 8 个)与 CPU 的型号有关，每个 GD 包最多 22 B 数据，最多 16 个 CPU 参与全局数据交换。

S7-400 CPU 可以发送和接收的 GD 包的个数与 CPU 的型号有关，可以发送 8 个或 16 个 GD 包，接收 16 个或 32 个 GD 包，每个 GD 包最多 64 B 数据。S7-400 CPU 具有对全局数据交换的控制功能，支持事件驱动的数据传送方式。

7.4.3 MPI 网络的组态

下面用一个例子来介绍对 MPI 网络组态的方法。在 STEP 7 中生成名为"MPI 全局数据通信"的项目，首先在 SMATIC 管理器中生成 3 个站，它们的 CPU 分别为 CPU 413-1、CPU 313C 和 CPU 312C。选中管理器左边窗口中的项目对象，在右边的工作区内双击 MPI 图标，打开 NetPro 工具，出现了一条红色的标有 MPI(1)的网络，和没有与网络相连的 3 个站的图标。双击某个站标有小红方块的区域(不要双击小红方块)，打开 CPU 的属性设置对话框，在"General"选项卡中点击"Interface"(接口)区内的[Properties]按钮，打开"Properties-MPI Interface"窗口，通过"Parameters"选项卡中的"Adress"列表框，设置 MPI 站地址。一般可以使用系统指定的地址，用户也可以修改它，各站的 MPI 地址应互不重叠。

在"Subnet"(子网)显示框中，如果选择 MPI(1)，该 CPU 就被连接到 MPI(1)子网上，选择"not networked"，将断开与 MPI(1)子网的连接。"Parameters"选项卡中的[New]按钮用来生成新的子网(例如 MPI(2))，[Delete]按钮用来删除选中的子网。[Properties]按钮用来设置选中的子网的属性，例如修改子网的名称，设置子网的传输速率等。图 7-14 是在 NetPro 中组态好的 MPI 网络。

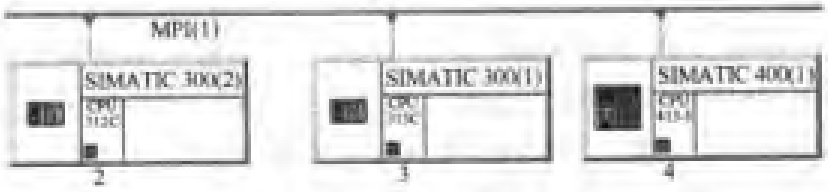


图 7-14 MPI 网络的组态

在硬盘上保存 CPU 的配置参数，用 PROFIBUS 电缆连接 MPI 节点，用点对点的方式将它们分别下载到各 CPU 中。可以用 SIMATIC 管理器的“Accessible Nodes”功能来测试可以访问的节点(即 MPI 网络中的站)。

7.4.4 全局数据表

1. 全局数据通信的组态步骤

全局数据(GD)通信用全局数据表(GD 表)来设置。全局数据通信的组态步骤如下：

(1) 生成和填写 GD 表；

(2) 第一次编译 GD 表；

(3) 设置 GD 包状态双字的地址和扫描速率(可选的操作)；

(4) 第二次编译 GD 表；

(5) 下载 GD 表。

2. 生成和填写 GD 表

在“NetPro”窗口中选中要设置的 MPI 网络线，网络线变粗，然后执行菜单命令“Options”→“Define Global Data”(定义全局数据)，或用右键点击 MPI 网络线，在弹出的窗口中执行同样的命令。在出现的 GD 窗口(见图 7-15)中对全局数据通信进行配置。

双击“GD ID”所在的最上面一行中“GD ID”右边的方格，在出现的“Select CPU”对话框中，双击第一个站的 CPU 413-1 的图标，CPU 413-1 便出现在最上面一行指定的方格中(见图 7-15)，同时自动退出“Select CPU”对话框。用同样的方法将另外两个 CPU 放置在最上面一行。

在 CPU 下面一行中生成 1 号 GD 环 1 号 GD 包中的 1 号数据，即将 CPU 413-1 的 MW0 发送到 CPU 313C 的 QW0。

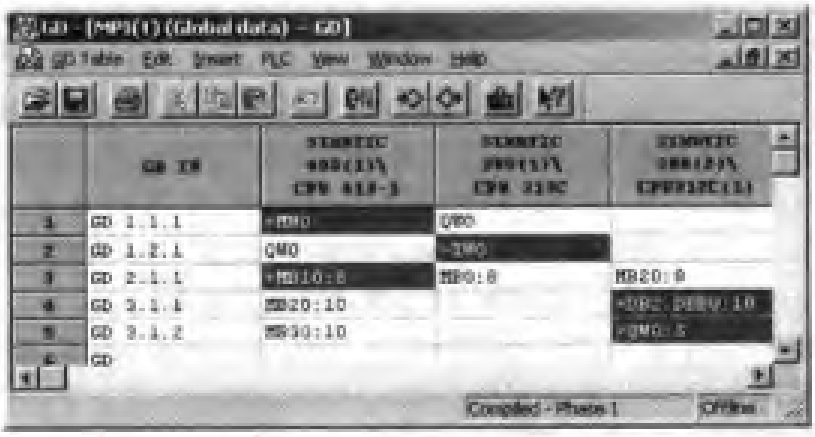


图 7-15 全局数据表

首先用鼠标右键点击 CPU 413-1 下面的单元(方格)，在出现的菜单中选择“Sender”(发送者)，该方格变为深色，同时在单元中的左端出现符号“>”，表示在该行中 CPU 413-1 为发送站，在该单元中输入要发送的全局数据的地址 MW0。只能输入绝对地址，不能输入符号地址。包含定时器和计数器地址的单元只能作为发送方。在每一行中应定义一个并且只能有一个 CPU 作为数据的发送方。同一行中各个单元的字节数应相同。

点击 CPU 313C 下面的单元，输入 QW0，该格的背景为白色，表示在该行中 CPU 313C 是接收站。

变量的复制因子用来定义连续的数据区的长度，例如 MB20:4 表示从 MB20 开始的 4 B。S7-300 的数据包可以由 MB0:22 或 MW0:11 组成，MB0:22 表示从 MB0 开始的 22 B，MW0:11 表示从 MW0 开始的 11 个字。如果数据包由若干个连续的数据区组成，一个连续的数据区占用的空间为数据区内的字节数加上两个头部说明字节。一个单独的双字占 6 B，一个单独的字占 4 B，一个单独的字节占 3 B，一个单独的位也占 3 B，例如 DB2.DBB0:10 和 QW0:5 一共占用 22 B。值得注意的是第一个连续数据区的两个头部说明字节不包括在 22 B 之内。

在图 7-15 的第 1 行和第 2 行中，CPU413-1 和 CPU 313C 组成 1 号 GD 环，两个 CPU 向对方发送 GD 包，同时接收对方的 GD 包，相当于全双工点对点通信方式。

图 7-15 中的第 3 行是 CPU 413-1 向 CPU 313C 和 CPU 312C 发送 GD 包，相当于 1:N 的广播通信方式。

图 7-15 中的第 4 行和第 5 行都是 CPU 312C 向 CPU 413-1 发送数据，它们属于 3 号 GD 环 1 号 GD 包中的两组数据。

发送方 CPU 自动地周期性地将指定地址中的数据发送到接收方指定的地址区中。例如图 7-15 中的第 5 行意味着 CPU 312C 定时地将 QW0～QW4 中的数据发送到 CPU 413-1 的 MB30～MB39 中。CPU 413-1 对它自己的 MB30～MB39 的访问，就好像在访问 CPU 312C 的 QW0～QW4 一样。

完成全局数据表的输入后，应执行菜单命令“GD Table”→“Compile...”，对它进行第一次编译，将各单元中的变量组合为 GD 包，同时生成 GD 环。

表中的 GD ID 列中的 GD 标识符是在编译时自动生成的。GD 3.1.2 表示 3 号 GD 环的 1 号 GD 包中的第 2 组变量。

3. 设置扫描速率和状态双字的地址

扫描速率用来定义 CPU 刷新全局数据的时间间隔。在第一次编译后，执行菜单命令“View”→“Scan Rates”，每个数据包将增加标有“SR”的行（见图 7-16），用来设置该数据包的扫描速率（1－255），扫描速率的单位是 CPU 的循环扫描周期，S7-300 默认的扫描速率为 8，S7-400 的为 22，用户可以修改默认的扫描速率。如果选择 S7-400 的扫描速率为 0，表示是事件驱动的 GD 发送和接收。

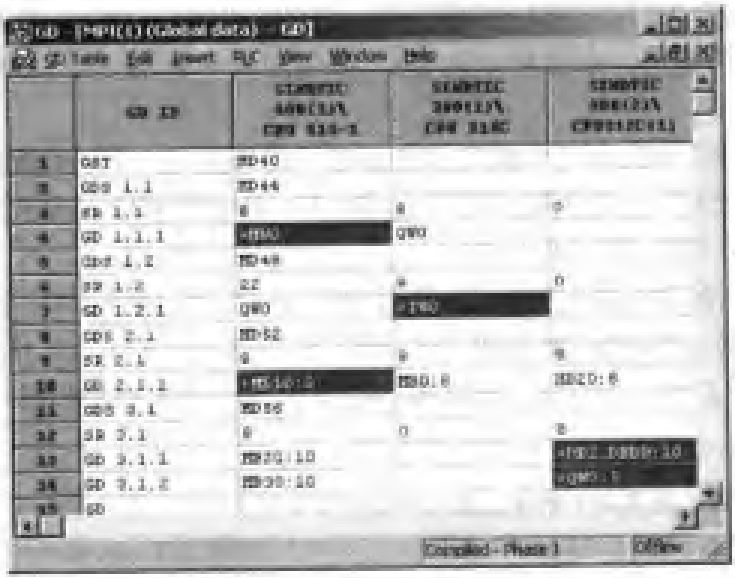


图 7-16　第一次编译后的全局数据表

可以用 GD 数据传输的状态双字来检查数据是否被正确地传送，第一次编译后执行菜单命令“View”→“Status”，在出现的 GDS 行中可以给每个数据包指定一个用于状态双字的地址。最上面一行的全局状态双字 GST 是各 GDS 行中的状态双字相“与”的结果。状态双字中使用的各位的意义如表 7-1 所示，被置位的位将保持其状态不变，直到它被用户程序复位。

表 7-1 GD 通信状态双字

位 号	说 明	状态位设定者
0	发送方地址区长度错误	发送或接收 CPU
1	发送方找不到存储 GD 的数据块	发送或接收 CPU
3	全局数据包在发送方丢失	发送 CPU
	全局数据包在接收方丢失	发送或接收 CPU
	全局数据包在链路上丢失	接收 CPU
4	全局数据包语法错误	接收 CPU
5	全局数据包 GD 对象遗漏	接收 CPU
6	接收方发送方数据长度不匹配	接收 CPU
7	接收方地址区长度错误	接收 CPU
8	接收方找不到存储 GD 的数据块	接收 CPU
11	发送方重新起动	接收 CPU
31	接收方接收到新数据	接收 CPU

状态双字使用户程序能及时了解通信的有效性和实时性，增强了系统的故障诊断能力。

设置好扫描速率和状态字的地址后，应对全局数据表进行第二次编译，使扫描速率和状态双字地址包含在配置数据中。第二次编译完成后，在 CPU 处于 STOP 模式时将配置数据下载到 CPU 中。下载完成后将各 CPU 切换到 RUN 模式，各 CPU 之间将开始自动地交换全局数据。

在循环周期结束时发送方的 CPU 发送数据，在循环周期开始时，接收方的 CPU 将接收到的数据传送到相应的地址区中。

7.4.5 事件驱动的全局数据通信

使用 SFC 60“GD _ SEND”和 SFC 61“GD _ RCV”，S7-400 可以用事件驱动的方式发送和接收 GD 包，实现全局通信。在全局数据表中，必须对要传送的 GD 包组态，并将扫描速率设置为 0。

SFC 60 和 SFC 61 可以在用户程序中的任何一点被调用。全局数据表中设置的扫描速率不受调用 SFC 60 和 SFC 61 的影响。SFC 60 和 SFC 61 能够被更高优先级的块中断。在高优先级的块中可以使用 SFC 60 和 SFC 61。

为了保证全局数据交换的连续性，在调用 SFC 60 之前应调用 SFC 39“DIS _ IRT”或 SFC 41“DIS _ AIRT”来禁止或延迟更高级的中断和异步错误。SFC 60 执行完后调用 SFC 40 “EN _ IRT”或 SFC 42“EN _ AIRT”，再次确认高优先级的中断和异步错误。下面是用 SFC 60 发送 GD3.1 的程序。

```
Network 1：延迟处理高中断优先级的中断和异步错误
      CALL "DIS _ AIRT "          //调用 SFC 41，延迟处理高中断优先级的中断
                                    和异步错误
```

```
      RET_VAL   :=MW100                    //返回的故障信息
Network 2：发送全局数据
      CALL "GD_SND"                        //调用 SFC 60
       CIRCLE_ID :=B#16#3                  //GD 环编号,允许值为 1~16
       BLOCK_ID  :=B#16#1                  //GD 包编号,允许值为 1~4
       RET_VAL   :=MW102                   //返回的故障信息
Network 3：允许处理高中断优先级的中断和异步错误
      CALL "EN_AIRT"                       //调用 SFC 42,允许处理高中断优先级的中
                                             断和异步错误
       RET_VAL   :=MW104                   //返回的故障信息
```

CIRCLE_ID 和 BLOCK_ID 分别是要发送的全局数据包的 GD 环和 GD 包的编号,允许的取值范围可以查阅 CPU 的技术数据。上述编号是用 STEP 7 配置 GD 数据表时设置的。

RET_VAL 是返回的故障信息,故障信息代码可以查阅有关的文献。

7.4.6 不用连接组态的 MPI 通信

不用连接组态的 MPI 通信用于 S7-300 之间、S7-300/400 之间、S7-300/400 与 S7-200 之间的通信,是一种应用广泛、经济的通信方式。

1. 需要双方编程的 S7-300/400 之间的通信

首先建立一个项目,对两个 PLC 的 MPI 网络组态,假设 A 站和 B 站的 MPI 地址分别设置为 2 和 3。将 A 站中 M20~M24 中的数据发送到 B 站的 M30~M34。

在 A 站的循环中断组织块 OB35 中调用系统功能 SFC 65"X_SEND",将 MB20~MB24 中 5 B 的数据发送到 B 站。在 B 站的 OB1 中调用系统功能 SFC"66 X_RCV",接收 A 站发送的数据,并存放到 MB30~34 中。

下面是发送方(A 站)的 OB35 中的程序:

```
Network 1：通过 MPI 发送数据
      CALL "X_SEND"
       REQ          :=TURE                 //激活发送请求
       CONT         :=TURE                 //发送完成后保持连接
       DEST_ID      :=W#16#3               //接收方的 MPI 地址
       REQ_ID       :=DW#16#1              //任务标识符
       SD           :=P#M20.0 BYTE 5       //本地 PLC 发送区
       RET_VAL      :=LW0                  //BUSY :=L2.0 //=1:发送未完成
```

输入 REQ 等号之后的值时输入"1",输入后自动变为"TURE"。下面是接收方(B 站)的 OB1 中的程序:

```
Network 1：从 MPI 接收数据
      CALL "X_RCV"
       EN_DT        :=TURE                 //将接收到的数据复制到接收区
       RET_VAL      :=LW0                  //返回的错误代码,=W#16#7000 无错误
       REQ_ID       :=LD2                  //SFC 65"X_SEND"的任务标识符
```

```
NDA          :=L6.0                    //为 0 没有新的排队数据,
                                       //为 1 且 EN_DT 为 1 时新数据被复制
RD           :=P#M30.0 BYTE 5          //本地 PLC 的数据接收区
```

2. 只需一个站编程的 S7-300/400 之间的通信

假设 A 站和 B 站的 MPI 地址分别为 2 和 3,B 站不用编程,在 A 站的循环中断组织块 OB35 中调用发送功能 SFC 68"X_PUT",将 MB40～MB49 中的 10 B 的数据发送到 B 站的 MB50～59 中;调用接收功能 SFC 67"X_GET",将对方的 MB60～MB69 中的 10 B 的数据读入到本地的 MB70～79 中。下面是 A 站的 OB35 中的程序:

```
Network 1 :用 SFC 68 从 MPI 发送数据
        CALL "X_PUT"
          REQ          :=TURE              //激活发送请求
          CONT         :=TURE              //发送完成后保持连接
          DEST_ID      :=W#16#3            //接收方的 MPI 地址
          VAR_ADDR     :=P#M50.0 BYTE 10   //对方的数据接收区
          SD           :=P#M40.0 BYTE 10   //本地的数据发送区
          RET_VAL      :=LW0               //返回的故障信息
          BUSY         :=L2.1              //为 1 发送未完成
Network 2 : 用 FSC 67 从 MPI 读取对方的数据到本地 PLC 的数据区
        CALL "X_GET"
          REQ          :=TURE              //激活请求
          CONT         :=TURE              //接收完成后保持连接
          DEST_ID      :=W#16#3            //对方的 MPI 地址
          VAR_ADDR     :=P#M60.0 BYTE 10   //要读取的对方的数据区
          RET_VAL      :=LW4               //返回的故障信息
          BUSY         :=L2.2              //为 1 发送未完成
          RD           :=P#M70.0 BYTE 10   //本地的数据接收区
```

SFC 69"X_ABORT"可以中断一个由"SFC X_SEND"、"X_GET"或"X_PUT"建立的连接。如果上述 SFC 的工作已完成(BUSY=0),调用 SFC 69"X_ABORT"后,通信双方的连接资源被释放。

7.5 执行器传感器接口 AS-i 网络

AS-i 是执行器传感器接口(Actuator Sensor Interface)的缩写,它是用于现场自动化设备(即传感器和执行器)的双向数据通信网络,位于工厂自动化网络的最底层。AS-i 已被列入 IEC 62026 国际标准的第 2 部分。AS-i 特别适用于连接需要传送开关量的传感器和执行器,例如读取各种接近开关、光电开关、压力开关、温度开关、物料位置开关的状态,控制各种阀门、声光报警器、继电器和接触器等,AS-i 也可以传送模拟量数据。

7.5.1 AS-i 的网络结构

AS-i 属于主从式网络,每个网段只能有一个主站。主站是网络通信的中心,负责网络的

初始化，以及设置从站的地址和参数等。它具有错误校验功能，发现传输错误将重发报文。传输的数据很短，一般只有 4 位。

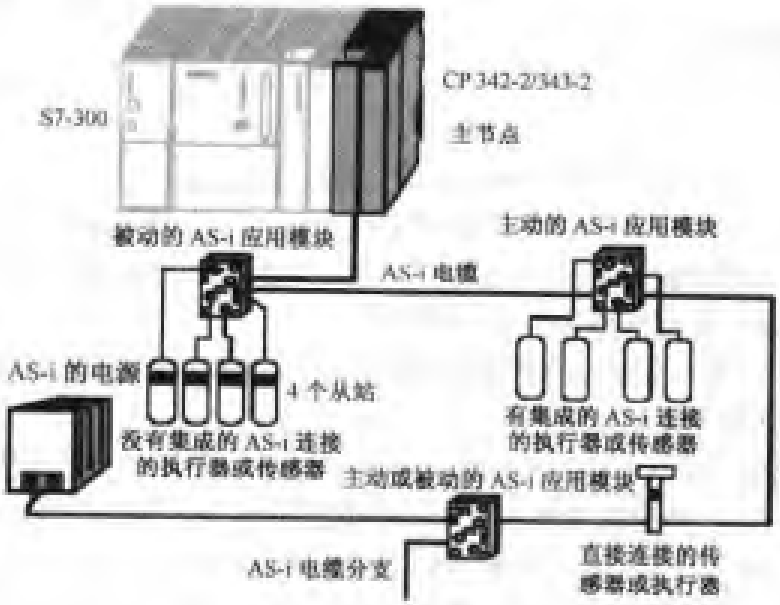


图 7-17　AS-i 网络示意图

AS-i 从站是 AS-i 系统的输入通道和输出通道，它们仅在被 AS-i 主站访问时才被激活。接到命令时，它们触发动作或者将现场信息传送给主站。

AS-i 的电源模块的额定电压为 DC 24 V，最大输出电流为 2 A。AS-i 所有分支电路的最大总长度为 100 m，可以用中继器延长。传输介质可以是屏蔽的或非屏蔽的两芯电缆，支持总线供电，即两根电缆同时可以作信号线和电源线。网络的树形结构允许电缆中的任意点作为新的分支的起点。

7.5.2　AS-i 的寻址模式

1. 标准寻址模式

AS-i 的节点(从站)地址为 5 位二进制数，每一个标准从站占一个 AS-i 地址，最多可以连接 31 个从站，地址 0 仅供产品出厂时使用，在网络中应改用别的地址。每一个标准 AS-i 从站可以接收 4 位数据或发送 4 位数据，所以一个 AS-i 总线网段最多可以连接 124 个二进制输入点和 124 个输出点，对 31 个标准从站的典型轮询时间为 5 ms，因此 AS-i 适用于工业过程开关量高速输入输出的场合。

用于 S7-200 的通信处理器 CP 242-2 和用于 S7-300，ET200M 的通信处理器 CP 342-2 属于标准 AS-i 主站。

2. 扩展的寻址模式

在扩展的寻址模式中，两个从站分别作为 A 从站和 B 从站，使用相同的地址，这样使可寻址的从站的最大个数增加到 62 个。由于地址的扩展，使用扩展的寻址模式的每个从站的二进制输出减少到 3 个，每个从站最多 4 点输入和 3 点输出。一个扩展的 AS-i 主站可以操作 186 个输出点和 248 个输入点。使用扩展的寻址模式时对从站的最大轮询时间为 10 ms。

用于 S7-200 的通信处理器 CP 243-2 和用于 S7-300,ET200M 的通信处理器 CP 343-2 属于扩展的 AS-i 主站。

7.5.3 AS-i 主站模块

1. CP 243-2

CP 243-2 是 S7-200 CPU 22x 的 AS-i 主站。通过连接 AS-i 可以显著地增加 S7-200 的数字量输入和输出点数,每个 CP 的 AS-i 上最多可以连接 124 个开关量输入和 124 个开关量输出。S7-200 同时可以处理最多 2 个 CP 243 2。CP 243-2 与 S7-200 的连接方法与其他扩展模块相同。它有 2 个端子直接连接 AS-i 接口电缆。

前面板的 LED 用来显示模块的状态、所有连接的从站模块的状态,以及监控 AS-i 网络的通信电压等。两个按钮用来切换运行状态。

在 S7-200 的映像区中,CP 243-2 占用 1 个数字量输入字节作为状态字节,1 个数字量输出字节作为控制字节。8 个模拟量输入字和 8 个模拟量输出字用于存放 AS-i 从站的数字量/模拟量输入/输出数据、AS-i 的诊断信息、AS-i 命令与响应数据等。

用户程序用状态字节和控制字节设置 CP 243-2 的工作模式。根据工作模式的不同,CP 243-2 在 S7-200 模拟地址区既可以存储 AS-i 从站的 I/O 数据或诊断值,也可以使能主站调用,例如改变一个从站地址。通过按钮,可以设置连接的所有 AS-i 从站。

CP 243-2 支持扩展 AS-i 特性的所有特殊功能。通过双重地址(A-B)赋值,最多可以处理 62 个 AS-i 从站。由于集成了模拟量值处理系统,CP 243-2 也可以访问模拟量。

2 CP 343-2

CP 343-2 通信处理器是用于 S7-300 PLC 和分布式 I/O ET 200 的 AS-i 主站,它具有以下功能:最多连接 62 个数字量或 31 个模拟量 AS-i 从站。支持所有 AS-i 主站功能,在前面板上用 LED 显示从站的运行状态、运行准备信息和错误信息,例如 AS-i 电压错误和组态错误。通过 AS-i 接口,每个 CP 最多可以访问 248 个数字量输入和 186 个数字量输出,可以对模拟量值进行处理。CP 342-2 占用 PLC 模拟区的 16 个输入字节和 16 个输出字节。通过它们来读写从站的输入数据和设置从站的输出数据。

3. CP 142-2

AS-i 主站 CP 142-2 用于 ET 200X 分布式 I/O 系统。CP 142-2 通信处理器通过连接器与 ET 200X 模块相连,并使用其标准 I/O 范围。AS-i 网络无需组态,最多 31 个从站可以由 CP 142-2(最多 124 点输入和 124 点输出)寻址。

4. DP/AS-i 接口网关模块

DP/AS-i 网关(Gateway)用来连接 PROFIBUS-DP 和 AS-i 网络。DP/AS-interface Link 20 和 DP/AS-interface Link 20E 可以作 DP/AS-i 的网关,后者具有扩展的 AS-i 功能。

CP 242-8 是标准的 AS-i 主站,它不仅有 CP 242-2 的功能,还可以作为 DP 从站连接到 PROFIBUS-DP。DP/AS-i 20E DP/AS-i 网络链接器以最高 12 M bit/s 的传输速率连接 PROFIBUS-DP 与 AS-i。它既是 PROFIBUS-DP 的从站,也是 AS-i 的集成主站,其防护等级为 IP20。它由 AS-i 电缆供电,因此系统无需增加 DC 24 V 电源。

5. SIMATIC C7 621 AS-i

SIMATIC C7 621 AS-i 把 AS-i 主站 CP 342-2、S7-300 的 CPU 以及 OP3 操作面板组合在一个外壳内，适合于高速方便地执行自动化任务，自带人机界面。这种紧凑型控制器可以直接访问和控制 31 个从站的 124 点数字量输入和 124 点数字量输出，无需在控制器内集成输入和输出，减小了控制器的体积。

6. 用于个人计算机的 AS-i 通信卡 CP 2413

CP 2413 是用于 PC(个人计算机)的标准 AS-i 主站，一台计算机可以安装 4 块 CP 2413。

因为在 PC 中还可以运行以太网和 PROFUBUS 总线接口卡，AS-i 从站提供的数据也可以被其他网络中其他的站使用。

SCOPE 是在计算机中运行的 AS-i 的诊断软件。它可以记录和评估在安装和运行过程中 AS-i 网络中的数据交换。

7.5.4 AS-i 从站模块

1. AS-i 从站的功能

从站所有的功能都集成在一片专用的集成电路芯片中，这样 AS-i 连接器可以直接集成在执行器和传感器中，全部元件可以安装在约 2 cm^2 的空间内。从站中的 AS-i 集成电路包含下列元件：4 个可组态的输入和输出，以及 4 个参数输出。在 EEPROM 存储器中存储运行参数、指定 I/O 的组态数据、标识码和从站地址等。

使用 AS-i 从站的参数输出，AS-i 主站可以传送参数值，它们用于控制和切换传感器或执行器的内部操作模式，例如在不同的运行阶段修改标度值。

4 位输入/输出组态(I/O 组态)用来指定从站的哪根数据线用来作为输入、输出或双向输出，从站的类型用标识码来描述。

2. AS-i 从站模块

AS-i 从站模块最多可以连接 4 个传统的传感器和 4 个传统的执行器。带有集成的 AS-i 连接的传感器和执行器可以直接连接到 AS-i 上。

SlimLine 模块的防护等级为 IP20，可以像其他低压设备一样安装在 DIN 导轨上，或用螺钉固定在控制柜的背板上。IP65/67 防护等级的 AS-i 从站模块可以直接安装在环境恶劣的工业现场。

3. “LOGO!”微型控制器

通过内置的 AS-i 模块，LOGO! 可以作为 AS－i 网络中的智能型从站使用。LOGO! 是一种微型 PLC，它具有数字量或模拟量输入和输出、逻辑处理器和实时钟功能，LOGO! 是 AS-i 网络中有分布式控制器功能的从站。使用 LOGO! 面板上的按键和显示器，可以对它编程和进行参数设置。

LOGO! 适合于简单的分布式自动化任务，例如门控系统，又可以通过 AS-i 网络将它纳入高端自动化系统中。在高端控制系统出现故障时，可以继续进行控制。

4. 紧凑型 AS-i 模块

这是一种具有较高保护等级的新一代紧凑型 AS-i 模块，包括数字、模拟、气动和 DV 24 V 电动机起动器模块。模块具有两种尺寸，可以满足各种安装要求，其保护等级为 IP67。

通过一个集成的编址插孔可以对已经安装的模块编址。所有的模块都可以通过与 S7 系

列 PLC 的通信实现参数设置。

西门子还提供了模拟量模块，每个模拟量模块有两个通道。有电流型、电压型、热电阻型传感器输入模块，和电流型、电压型执行器输出模块。

5. 气动控制模块

西门子提供两种类型的 AS-i 气动模块，即带两个集成的 3/2 路阀门的气动用户模块和带两个集成的 4/2 路阀门的气动紧凑型模块。模块有单稳和双稳两种类型，集成了作为气动单元执行器的阀门，接收来自气缸的位置信号。

6. 电动机起动器

西门子有 3 种类型的电动机起动器，在 AS-i 中作标准从站。防护等级均为 IP65，有非熔断器保护，可以进行可逆起动。可以起动的异步电动机最大功率为 4～5.5 kW。

7. DC 24 V 电动机起动器

K60 AS-i DC 24 V 电动机起动器可以驱动 70 W 功率的电动机，它将 DC 24 V 电动机起动器及其传感器直接连接到 AS-i 上。有的有制动器和可选的急停功能。

8. 能源与通信现场安装系统

ECOFAST(能源与通信现场安装系统)是一个开放的控制柜系统解决方案。所有的自动化和相应的安装器件应用标准和接口将数据和动力的传输有机地连成一体。与 AS-i 有关的下列元器件可以集成到 ECOFAST 中：所有的 I/O 模块；安装在电动机接线盒上或电动机附近的可逆起动器和软起动器；集成在电动机上的微型起动器；动力和控制装置(动力电源)与 PLC 和 AS-i 主站的组合装置。

9. 接近开关

BERO 接近开关可以直接连接到 AS-i 或接口模块上。特殊的感应式、光学和声纳 BERO 接近开关适合直接连接到 AS-i 上。它们集成有 AS-i 芯片，除了开关量输出之外，还提供其他信息，例如开关范围和线圈故障。通过 AS-i 电缆可以对这些智能 BERO 设置参数。

10. 按钮和 LED

SIGNUM 3SB4 是一个具有 AS-i 接口的完整的操作员通信系统人机界面。带灯的指令按钮通过 AS-i 电缆供电，通过特殊的 AS-i 从站和独立的辅助电源，可以实现控制设备的单个连接，每个设备最多可以连接 28 个常开触点和 7 个信号输出点。

7.5.5 AS-i 的主从通信方式

AS-i 是单主站系统，AS-i 通信处理器(CP)作为主站控制现场的通信过程。主从通信过程如图 7-18 所示，主站一个接一个地轮流询问每一个从站，询问后等待从站的响应。

地址是 AS-i 从站的标识符。可以用专用的定址(Addressing)单元或主站来设置各从站的地址。

AS-i 使用电流调制的传输技术保证了通信的高可靠性。主站如果检测到传输错误或从站的故障，将会发送报文给 PLC，提醒用户进行处理。在正常运行时增加或减少从站，不会影响其他从站的通信。

扩展的 AS-i 接口技术规范 V2.1 最多允许连接 62 个从站，主站可以对模拟量进行处理。

AS-i 的报文主要有主站呼叫发送报文和从站应答(响应)报文(见图 7-19)，主站的请求帧由 14 个数据位组成：

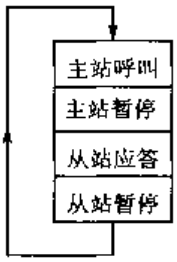


图 7-18　AS-i 的主从通信

图 7-19　AS-i 的通信报文

在主站呼叫发送报文中，ST 是起始位，其值为 0。SB 是控制位，为 0 或为 1 时分别表示传送的是数据或命令。A4～A0 是从站地址，I4～I0 为数据位。PB 是奇偶校验位，在报文中不包括结束位在内的各位中 1 的个数应为偶数。EB 是结束位，其值为 1。在 7 个数据位组成的从站应答报文中，ST、PB 和 EB 的意义与取值与主站呼叫发送报文的相同。

主站通过呼叫发送报文，可以完成下列功能：

(1) 数据交换：主站通过报文把控制指令或数据发送给从站，或让从站把测量数据上传给主站。

(2) 设置从站的参数：例如设置传感器的测量范围，激活定时器和改变测量方法等。

(3) 删除从站地址：把被呼叫的从站地址暂时改为 0。

(4) 地址分配：只能对地址为 0 的从站分配地址。从站把新地址存放在 EEPROM 中。

(5) 复位功能：把被呼叫的从站恢复为初始状态时的地址。

(6) 读从站的 I/O 配置。

(7) 读取从站的 ID(标识符)代码。

(8) 状态读取：读取从站的 4 个状态位，以获得在寻址和复位时出现的错误的信息。

(9) 状态删除：读取从站的状态并删除其内容。

7.5.6　AS-i 从站的通信接口

AS-i 的从站由专用的 AS-i 通信芯片和传感器或执行器部分组成。

AS-i 的从站包括以下功能单元：电源供给单元、通信的发送器和接收器、微处理器、数据输入/输出单元、参数输出单元和 EEPROM 存储器芯片。

微处理器是实现通信功能的核心，它接收来自主节点的呼叫发送报文，对报文进行解码和出错检查，实现主、从站之间的双向通信，把接收到的数据传送给传感器和执行器，向主站发送响应报文。

从站可以带电插拔，短路及过载状态不会影响其他站点的正常通信。

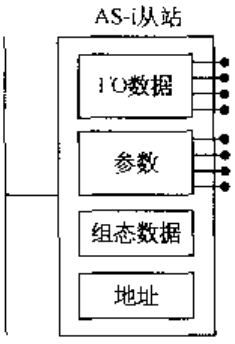


图 7-20　AS-i 从站

7.5.7　AS-i 的工作阶段

1. 离线阶段

离线阶段又称为初始化模式，在该阶段设置主站的基本状态。模块上电后或被重新起动后被初始化。在初始化期间，所有从站的输入和输出数据的映像被设置为 0(未激活)。

电源接通后，组态数据被复制到参数区，后面的激活操作可以使用预置的参数。如果主站

在运行中被重新初始化，参数区中可能已经变化的值被保持。

2. 起动阶段

在起动阶段，主站检测 AS-i 电缆上连接有哪些从站，以及它们的型号。厂家制造 AS-i 从站时通过组态数据，将从站的型号永久性地保存在从站中，主站可以请求上传这些数据。组态文件中包含了 AS-i 从站的 I/O 分配情况和从站的类型(ID 代码)。主站将检测到的从站存放在检测到的从站表中。

3. 激活阶段

在激活阶段，主站检测到 AS-i 从站后，通过发送特殊的呼叫，激活这些从站。

主站处于组态模式时，所有地址不为 0 的被检测到的从站被激活。在这一模式，可以读取实际的值并将它们作为组态数据保存。

主站处于保护模式时，只有储存在主站的组态中的从站被激活。如果在网络上发现的实际组态不同于期望的组态，主站将显示出来。主站把激活的从站存入被激活的从站表中。

4. 工作模式

起动阶段结束后，AS-i 主站切换到正常循环的工作模式。

(1) 数据交换阶段

在正常模式，主站将周期性地发送输出数据给各从站，并接收它们返回的应答报文，即输入数据。如果检测出传输过程中的错误，主站重复发出询问。

(2) 管理阶段

在这一阶段，处理和发送下述可能的控制应用任务：将 4 个参数位发送给从站，例如设置门限值；改变从站的地址，如果从站支持这一特殊功能的话。

(3) 包含(inclusion)阶段

在这一阶段，新加入的 AS-i 从站被包含到已检测到的从站表中，如果它们的地址不为 0，将被激活。主站如果处于保护模式，只有储存在主站的期望组态中的从站被激活。

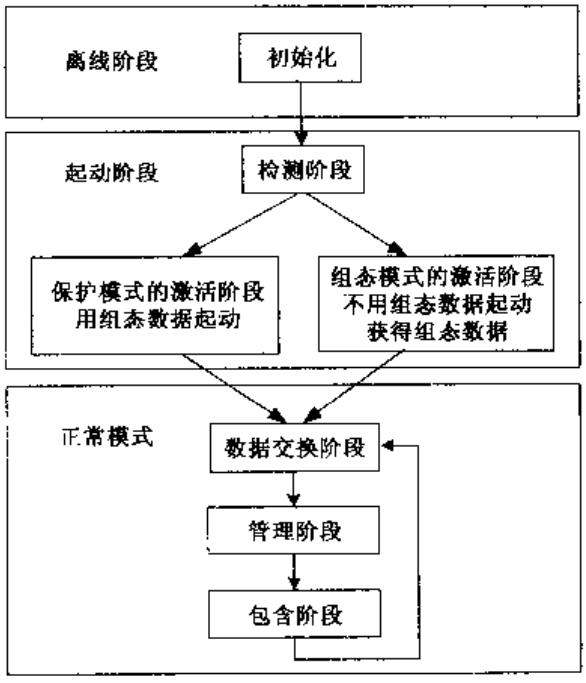


图 7-21　AS-i 的工作阶段

7.6 工业以太网

7.6.1 工业以太网简介

工业以太网是为工业应用专门设计的，它是遵循国际标准 IEEE 802.3 (Ethernet)的开放式、多供应商、高性能的区域和单元网络。工业以太网已经广泛地应用于控制网络的最高层，并且有向控制网络的中间层和底层(现场层)发展的趋势。

1. 以太网的特点

企业内部互联网(Intranet)、外部互联网(Extranet)以及国际互联网(Internet)不但进入了办公室领域，而且已经广泛地应用于生产和过程自动化。继 10 Mbit/s 以太网成功运行之后，具有交换功能、全双工和自适应的 100 Mbit/s 高速以太网(Fast Ethernet，符合 IEEE 802.3u 标准)也已经成功运行多年。SIMATIC NET 可以将控制网络无缝集成到管理网络和互联网。

以太网的市场占有率高达 80%，毫无疑问是当今局域网 (LAN) 领域中首屈一指的网络。以太网有以下优点：

- 可以采用冗余的网络拓扑结构，可靠性高；
- 通过交换技术可以提供实际上没有限制的通信性能；
- 灵活性好，现有的设备可以不受影响地扩张；
- 在不断发展的过程中具有良好的向下兼容性，保证了投资的安全；
- 易于实现管理控制网络的一体化；

以太网可以接入广域网 (WAN)，例如综合服务数字网(ISDN)或互联网，可以在整个公司范围内通信，或实现公司之间的通信。

SIMATIC NET 供应的节点已超过 400 000 个，可以用于严酷的工业环境，包括有强烈电磁干扰的区域。

2. 工业以太网的构成

典型的工业以太网络由以下 4 类网络器件组成：

(1) 连接部件：包括 FC 快速连接插座，电气链接模块 (ELS)，电气交换模块(ESM)，光纤交换模块(OSM)和光纤电气转换模块 (MC TP11)。

(2) 通信介质：可以采用普通双绞线、工业屏蔽双绞线和光纤。

(3) SIMATIC PLC 的工业以太网通信处理器：用于将 PLC 连接到工业以太网。

(4) PG/PC 的工业以太网通信处理器：用于将 PG/PC 连接到工业以太网。

3. 工业以太网的特性

为了在严酷的工业环境应用，确保安全可靠，SIMATIC NET 为工业以太网技术增添了不少重要的性能：

- 与 IEEE802.3/802.3u 兼容，使用 ISO 和 TCP/IP 通信协议；
- 10 Mbit/s 或 100 Mbit/s 自适应传输速率；
- DC 24 V 冗余供电；

简单的机柜导轨安装；

- 能方便地组成星形、总线型和环形拓扑结构；

- 高速冗余的安全网络，最大网络重构时间为 0.3 s；
- 用于严酷环境的网络元件，通过 EMC(电磁兼容性)测试；
- 通过 RJ-45 接口，工业级的 Sub-D 连接技术和安装专用屏蔽电缆的 Fast Connect 连接技术，确保现场电缆安装工作的快速进行；
- 简单高效的信号装置不断地监视网络元件；
- 符合 SNMP(简单的网络管理协议)；
- 可以使用基于 Web 的网络管理器；
- 使用 VB/VC 或组态软件即可以监控管理网络。

7.6.2 工业以太网的网络方案

工业以太网可以采用下面的 3 种方案(见图 7-22)：

1. 三同轴电缆网络

网络以三同轴电缆作为传输介质，由若干条总线段组成，每段的最大长度为 500 m。一条总线段最多可以连接 100 个收发器，可以通过中继器接入更多的网段。

网络为总线型结构，因为采用了无源设计和一致性接地的设计，极其坚固耐用。网络中各设备共享 10 Mbit/s 带宽。

可以混合使用电气网络和光纤网络，使二者的优势互补，网络的分段改善了网络的性能。

三同轴电缆网络有分别带一个或两个终端设备接口的收发器，中继器用来将最长 500 m 的分支网段接入网络中。

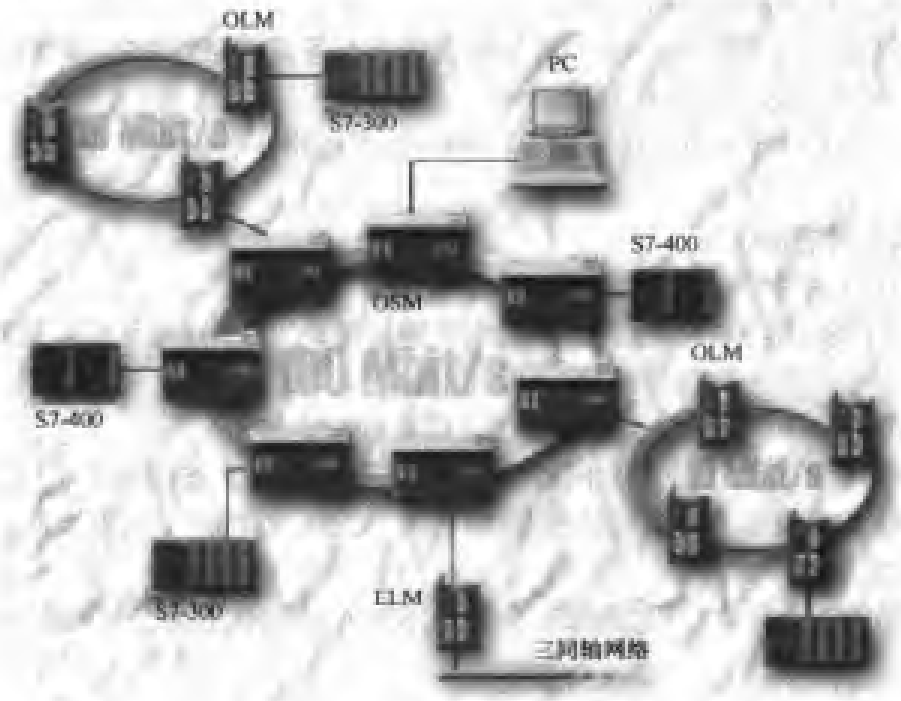


图 7-22 工业以太网

2. 双绞线和光纤网络

双绞线和光纤网络的传输速率为 10 Mbit/s，可以是总线型或星形拓扑结构，使用光纤链接模块(OLM)和电气链接模块(ELM)。

OLM 和 ELM 是安装在 DIN 导轨上的中继器，它们遵循 IEEEE 802.3 标准，带有 3 个工业双绞线接口，OLM 和 ELM 分别有两个和一个 AUI 接口。在一个网络中最多可以级联 11 个 OLM 或 13 个 ELM。

3. 高速工业以太网

高速工业以太网的传输速率为 100 Mbit/s，使用光纤交换模块(OSM)或电气交换模块(ESM)。工业以太网与高速工业以太网的数据格式、CSMA/CD 访问方式和使用的电缆都是相同的，高速以太网最好用交换模块来构建。

4. 以太网和高速以太网的工作过程

以太网使用带冲突检测的载波侦听多路访问(CSMA/CD)协议，各站用竞争方式发送信息到传输线上，两个或多个站可能因同时发送信息而发生冲突。为了保证正确地处理冲突，以太网的规模必须根据一个数据包最大可能的传输延迟来加以限制。在传统的 10 Mbit/s 以太网中，允许的冲突范围为 4520 m，因为传输速率的提高，高速以太网的冲突范围减小为 452 m。为了扩展冲突范围，需要使用有中继器功能的网络部件，例如工业以太网的 OLM 和 ELM。

用具有全双工功能的交换模块来构建较大的网络时，不必考虑高速以太网冲突区域的减小。

7.6.3 工业以太网的交换技术

1. 交换技术

在共享局域网(LAN)中，所有站点共享网络性能和数据传输带宽，所有的数据包都经过所有的网段，在同一时间只能传送一个报文。

在交换式局域网中，每个网段都能达到网络的整体性能和数据传输速率，在多个网段中可以同时传输多个报文。本地数据通信在本网段进行，只有指定的数据包可以超出本地网段的范围。

交换模块是从网桥发展而来的设备，利用终端的以太网 MAC 地址，交换模块可以对数据进行过滤，局部子网的数据仍然是局部的，交换模块只传送发送到其他子网络终端的数据。与一般的以太网相比扩大了可以连接的终端数，可以限制子网内的错误在整个网络上的传输。

虽然较复杂，但交换技术与中继技术相比有下面的优点：

- 可以选择用来构建部分网络或是网段，通过数据交换结构提高了数据吞吐量和网络性能，网络配置规则简单。
- 不必考虑传输延时，可以方便地实现有 50 个 OSM 或 ESM 的网络拓扑结构。通过连接单个的区域或部分网络，可以实现网络规模的无限扩展。

2. 全双工模式

在全双工模式，一个站能同时发送和接收数据。如果网络采用全双工模式，不会发生冲突。全双工模式需要采用发送通道和接收通道分离的传输介质，以及能够存储数据包的部件。

由于在全双工连接中不会发生冲突，支持全双工的部件可以同时以额定传输速率发送和接收数据，因此以太网和高速以太网的传输速率分别提高到 20 Mbit/s 和 200 Mbit/s。

由于不需要检测冲突，全双工网络的距离仅受到它使用的发送部件和接收部件性能的限制，使用光纤网络时更是如此。

100BaseFX 标准规定 62.5/125 μm 的光纤电缆的最大传输距离为 2000 m。采用光纤交换

模块，62.5/125 μm 的光纤电缆的距离可达 3000 m，10/125 μm 的光纤电缆可达 26 km。

3. 电气交换模块与光纤交换模块

电气交换模块(ESM)与光纤交换模块(OSM)用来构建 10 M/100 Mbit/s 交换网络，能低成本、高效率地在现场建成具有交换功能的线性结构或星形结构的工业以太网。

可以将网络划分为若干个部分或网段，并将各网段连接到 ESM 或 OSM 上，这样可以分散网络的负担，实现负载解耦，改善网络的性能。

利用 ESM 或 OSM 中的网络冗余管理器，可以构建环形冗余工业以太网。最大的网络重构时间为 0.3 s。环形网中的数据传输速率为 100 Mbit/s，每个环最多可以用 50 个 ESM 或 50 个 OSM。

除了两个环端口外，ESM 还有另外 6 个端口（可以任选 ITP 或 RJ-45 接口）；除了两对或 3 对环端口外，OSM 还有另外 6 个或 5 个端口，这些端口可以与终端设备或网络段连接。使用集成的后备功能可以将几个环以冗余方式连接在一起。

可以使用以下四种方法监测出错信息和进行网络管理：使用硬连线接点、SNMP 协议、电子邮件和命令行参数 CLI。

通过 ESM 可以方便地构建适用于车间的网络拓扑结构，包括线性结构和星形结构。级联深度和网络规模仅受信号传输时间的限制，使用 ESM 可以使网络总体规模达 5 km，使用 OSM 时网络长度可达 150 km。通过将各个子网络连接到 ESM，可以重构现有的网络。

通过环中的两个 ESM 或 OSM，10 Mbit/s 或 100 Mbit/s 的单个冗余环可以与上层 100 Mbit/s 环相连接。

7.6.4 自适应与冗余网络

1. 自适应与自协商功能

具有自适应功能的网络站点（终端设备和网络部件）能自动检测出信号传输速率（10 Mbit/s 或100 Mbit/s），自适应功能可以实现所有以太网部件之间的无缝互操作性。

自协商是高速以太网的配置协议，该协议使有关站点在数据传输开始之前就能协商，以确定它们之间的数据传输速率和工作方式，例如是全双工或半双工。

也可以不使用自协商功能，以保证各网络站点使用某一特定的传输速率和工作方式。

不支持自协商功能的传统以太网部件（例如工业以太网的 OLM）能通过双绞线连接与具有自协商功能的高速以太网部件协同工作。

2. 冗余网络

冗余软件包 S7-REDCONNECT 用来将 PC 链接到高可靠性的 SIMATIC S7-H 系统。S7-H 冗余系统可以避免设备停机。万一出现子系统故障或断线，系统交换模块会切换到双总线，或者切换到冗余环的后备系统或后备网络，以保证网络的正常通信。

3. SIMATIC NET 的快速重新配置

网络发生故障后，应尽快对网络进行重构。重新配置的时间对工业应用是至关重要的，否则网络上连接的终端设备将会断开连接，从而引起工厂生产过程的失控或紧急停机。SIMATIC NET 采用了专门为此开发的冗余控制程序，对于有 50 个交换模块(OSM/ESM)的 100 Mbit/s 环形网络，重新配置时间不超过 0.3 s。

终端设备不受网络变化的影响，并不清除其逻辑连接，这就确保了任何时候都可以对生产

过程进行控制。在 100 Mbit/s 环形网络中，OSM/ESM 具有环或者网段的高速冗余接口。

只要配置两个 OSM 或 ESM，它们与工业以太网的 OLM 环之间，或任意个数的网段都能相互连接。

4. SNMP-OPC Server

使用 SNMP-OPC 服务器(Server)，用户可以通过 OPC 客户端软件，例如 SIMATIC NET OPC Scout、WinCC、OPC Client、MS Office OPC Client 等，对支持 SNMP 协议的网络设备进行远程管理。SNMP-OPC Server 可以读取网络设备参数，例如交换模块的端口状态、端口数据流量等；可以修改网络设备的状态，例如关闭/开启交换模块的某个端口等。

7.6.5 工业以太网的网卡与通信处理器

1. 用于 PC 的工业以太网网卡

(1) CP 1612 PCI 以太网卡和 CP 1512 PCMCIA 以太网卡提供 RJ-45 接口，与配套的软件包一起支持以下的通信服务：传输协议 ISO 和 TCP/IP、PG/OP 通信、S7 通信、S5 兼容通信(SEND/RECEIVE)，支持 OPC 通信。

(2) CP 1515 是符合 IEEE 802.11b 的无线通信网卡，应用于 RLM(无线链路模块)和可移动计算机。

(3) CP 1613 是带微处理器的 PCI 以太网卡，使用 AUI/TTP 接口或 RJ-45 接口，可以将 PG/PC 连接到以太网网络。用 CP 1613 可以实现时钟的网络同步。与有关的软件一起，CP 1613 支持以下的通信服务：ISO 和 TCP/IP 通信协议、PG/OP 通信、S7 通信、S5 兼容通信和 TF 协议，支持 OPC 通信。

由于集成了微处理器，CP 1613 有恒定的数据吞吐量，支持“即插即用”和自适应(10 Mbit/s 或 100 Mbit/s)功能。支持运行大型的网络配置，可以用于冗余通信，支持 OPC 通信。

2. S7-300/400 的工业以太网通信处理器

S7-300/400 工业以太网通信处理器有下列特点：

- 通过 UDP 连接或群播功能可以向多用户发送数据；
- CP 443-1 和 CP 443-1 IT 可以用网络时间协议（NTP）提供时钟同步；
- 可以选择 KeepAlive 功能；
- 使用 TCP/IP 的 WAP 功能，通过电话网络（例如 ISDN），CP 可以实现远距离编程和对设备进行远程调试；
- 可以实现 OP 通信的多路转换，最多连接 16 个 OP；
- 使用集成在 STEP 7 中的 NCM，提供范围广泛的诊断功能，包括显示 CP 的操作状态，实现通用诊断和统计功能，提供连接诊断和 LAN 控制器统计及诊断缓冲区。

(1) CP 343-1/CP 443-1 通信处理器

CP 343-1/CP 443-1 是分别用于 S7-300 和 S7-400 的全双工以太网通信处理器，通信速率为 10 Mbit/s 或 100 Mbit/s。CP 343-1 的 15 针 D 形插座用于连接工业以太网，允许 AUI 和双绞线接口之间的自动转换。RJ-45 插座用于工业以太网的快速连接，可以使用电话线通过 ISDN 连接互联网。CP 443-1 有 ITP，RJ-45 和 AUI 接口。

CP 343-1/CP 443-1 在工业以太网上独立处理数据通信，有自己的处理器。通过它们 S7-300/400 可以与编程器、计算机、人机界面装置和其他 S7 和 S5 PLC 进行通信。

通信服务包括用 ISO 和 TCP/IP 传输协议建立多种协议格式、PG/OP 通信、S7 通信、S5 兼容通信和对网络上所有的 S7 站进行远程编程。通过 S7 路由，可以在多个网络间进行 PG/OP 通信，通过 ISO 传输连接的简单而优化的数据通信接口，最多传输 8 KB 的数据。

可以使用下列接口：ISO 传输，带 RFC 1006 的（例如 CP 1430 TCP）或不带 RFC 1006 的 TCP 传输，UDP 可以作为模块的传输协议。S5 兼容通信用于 S7 和 S5，S7-300/400 与计算机之间的通信。S7 通信功能用于与 S7-300（只限服务器），S7-400（服务器和客户机），HMI 和 PC 机（用 SOFTNETS7 或 S7-1613）通信。

可以用嵌入 STEP 7 的 NCM S7 工业以太网软件对 CP 进行配置。模块的配置数据存放在 CPU 中，CPU 起动时自动地将配置参数传送到 CP 模块。连接在网络上的 S7 PLC 可以通过网络进行远程配置和编程。

(2) CP 343-1 IT /CP 443-1 IT 通信处理器

CP 343 1 IT /CP 443-1 IT 通信处理器分别用于 S7-300 和 S7-400，除了具有 CP 343-1/CP 443-1 的特性和功能外，CP 343-1 IT /CP 443-1 IT 可以实现高优先级的生产通信和 IT 通信，它有下列 IT 功能：

1) Web 服务器：可以下载 HTML 网页，并用标准浏览器访问过程信息（有口令保护）。

2) 标准的 Web 网页：用于监视 S7-300/400，这些网页可以用 HTML 工具和标准编辑器来生成，并用标准 PC 工具 FTP 传送到模块中。

3) E-mail：通过 FC 调用和 IT 通信路径，在用户程序中用 E-mail 在本地和世界范围内发送事件驱动信息。

(3) CP 444 通信处理器

CP 444 将 S7-400 连接到工业以太网，根据 MAP 3.0（制造自动化协议）标准提供 MMS（制造业信息规范）服务，包括环境管理（起动、停止和紧急退出），VMD（设备监控）和变量存取服务。可以减轻 CPU 的通信负担，实现深层的连接。

第 8 章　现场总线 PROFIBUS 及其应用

8.1　PROFIBUS 的结构与硬件

PROFIBUS 是目前国际上通用的现场总线标准之一，它以其独特的技术特点、严格的认证规范、开放的标准、众多厂商的支持和不断发展的应用行规，已被纳入现场总线的国际标准 IEC 61158 和欧洲标准 EN 50170，并于 2001 年被定为我国的国家标准 JB/T10308.3-2001。

PROFIBUS 是不依赖生产厂家的、开放式的现场总线，各种各样的自动化设备均可以通过同样的接口交换信息。PROFIBUS 用于分布式 I/O 设备、传动装置、PLC 和基于 PC(个人计算机)的自动化系统。

PROFIBUS 在 1999 年 12 月通过的 IEC 61156 中称为 Type 3，PROFIBUS 的基本部分称为 PROFIBUS-V0。在 2002 年新版的 IEC 61156 中增加了 PROFIBUS-V1，PROFIBUS-V2 和 RS-485IS 等内容。新增的 PROFInet 规范作为 IEC 61158 的 Type10。

代表全世界 1 200 多家会员公司的 PROFIBUS 国际组织宣布，截止到 2003 年底，工厂自动化和流程自动化应用系统所安装的 PROFIBUS 节点设备已突破了 1 千万个，PROFIBUS 俱乐部宣称，在中国已突破了 150 万个，PROFIBUS ASIC 接口芯片的销量已超过了 1 千万个。

可以用编程软件 STEP 7 或 SIMATIC NET 软件，对 PROFIBUS 网络设备组态和设置参数，起动或测试网络中的节点。

8.1.1　PROFIBUS 的组成

PROFIBUS 由 3 部分组成，即 PROFIBUS-DP (Decentralized Periphery，分布式外围设备)、PROFIBUS-PA (Process Automation，过程自动化)和 PROFIBUS-FMS (Fieldbus Message Specification，现场总线报文规范)。

1. PROFIBUS-FMS

PROFIBUS-FMS 定义了主站与主站之间的通信模型，它使用 OSI 7 层模型的第 1 层、第 2 层和第 7 层。应用层(第 7 层)包括现场总线报文规范 FMS 和低层接口 LLI (Lower Layer Interface)。

FMS 包含应用层协议，并向用户提供功能很强的通信服务。LLI 协调不同的通信关系，并提供不依赖于设备的第二层访问接口。第 2 层(总线数据链路层)提供总线存取控制和保证数据的可靠性。

FMS 主要用于系统级和车间级的不同供应商的自动化系统之间传输数据，处理单元级(PLC 和 PC)的多主站数据通信，为解决复杂的通信任务提供了很大的灵活性。

2. PROFIBUS-DP

PROFIBUS-DP 用于自动化系统中单元级控制设备与分布式 I/O 的通信，可以取代 4～20 mA 模拟信号传输。

PROFIBUS-DP 使用第 1 层、第 2 层和用户接口层，第 3～7 层未使用，这种精简的结构确保了高速数据传输。直接数据链路映像程序 DDLM 提供对第 2 层的访问。用户接口规定了设备的应用功能、PROFIBUS-DP 系统和设备的行为特性。PROFIBUS-DP 特别适合于 PLC 与现场级分布式 I/O(例如西门子的 ET 200)设备之间的通信。主站之间的通信为令牌方式，主站与从站之间为主从方式，以及这两种方式的混合。

S7-300/400 系列 PLC 有的配备有集成的 PROFIBUS-DP 接口，S7-300/400 也可以通过通信处理器(CP)连接到 PROFIBUS-DP。

3. PROFIBUS-PA

PROFIBUS-PA 用于过程自动化的现场传感器和执行器的低速数据传输，使用扩展的 PROFIBUS-DP 协议，此外还描述了现场设备行为的 PA 行规。由于传输技术采用 IEC 1158-2 标准，确保了本质安全和通过总线对现场设备供电，可以用于防爆区域的传感器和执行器与中央控制系统的通信。使用分段式耦合器可以将 PROFIBUS-PA 设备很方便地集成到 PROFIBUS-DP 网络中。

PROFIBUS-PA 使用屏蔽双绞线电缆，由总线提供电源。在危险区域每个 DP/PA 链路可以连接 15 个现场设备，在非危险区域每个 DP/PA 链路可以连接 31 个现场设备。

此外基于 PROFIBUS，还推出了用于运动控制的总线驱动技术 PROFI-drive 和故障安全通信技术 PROFI-safe。

8.1.2 PROFIBUS 的物理层

ISO/OSI 参考模型的物理层是第 1 层，PROFIBUS 可以使用多种通信介质(电、光、红外、导轨以及混合方式)。传输速率 9.6 k～12 Mbit/s，在一个有 32 个站点的系统中，假设 PROFIBUS-DP 对所有站点传送 512 点输入和 512 点输出，在 12 Mbit/s 时只需 1 ms。每个 DP 从站的输入数据和输出数据最大为 244 B。使用屏蔽双绞线电缆时最长通信距离为 9.6 km，使用光缆时最长 90 km，最多可以接 127 个从站。

PROFIBUS 可以使用灵活的拓扑结构，支持线形、树形、环形结构以及冗余的通信模型。支持基于总线的驱动技术和符合 IEC 61508 的总线安全通信技术。下面介绍用于 DP 和 FMS 的 RS-485 传输、光纤传输，以及用于 PA 的 IEC 1158-2 传输。

1. DP/FMS 的 RS-485 传输

PROFIBUS-DP 和 PROFIBUS-FMS 使用相同的传输技术和统一的总线存取协议，可以在同一根电缆上同时运行。

DP/FMS 符合 EIA RS-485 标准(也称为 H2)，采用价格便宜的屏蔽双绞线电缆，电磁兼容性(EMC)条件较好时也可以使用不带屏蔽的双绞线电缆。一个总线段的两端各有一套有源的总线终端电阻(见图 8-1)。传输速率 9.6 kbit/s～12M bit/s，所选的传输速率适用于连接到总线段上的所有设备。一个总线段最多 32 个站，带中继器最多 127 个站，串联的中继器一般不超过 3 个。中继器没有站地址，但是被计算在每段的最大站数中。

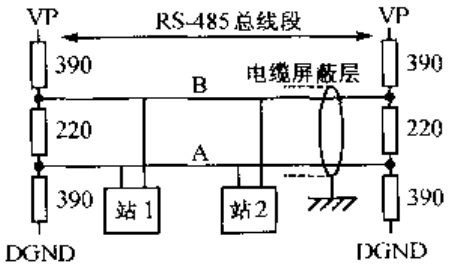


图 8-1 DP/FMS 总线段的结构

每段的电缆最大长度与传输速率有关，例如使用 A 型电缆，3～12M bit/s 时为 100 m，9.6

~93.75 kbit/s 时为 1200 m。

如果用屏蔽编织线和屏蔽箔，应在两端与保护接地连接，数据线必须与高压线隔离。

RS-485 采用半双工、异步的传输方式，1 个字符帧由 8 个数据位、1 个起始位、1 个停止位和 1 个奇偶校验位组成(共 11 位)。

2. D 型总线连接器

PROFIBUS 标准推荐总线站与总线的相互连接使用 9 针 D 型连接器。D 型连接器的插座与总线站相连接，而 D 型连接器的插头与总线电缆相连接。连接器的接线如表 8-1 所示。

在传输期间，A、B 线上的波形相反。信号为 1 时 B 线为高电平，A 线为低电平。各报文间的空闲(Idle)状态对应于二进制“1”信号。

表 8-1 D 型连接器的引脚分配

针脚号	信号名称	说明
1	SHIELD	屏蔽或功能地
2	M24	24 V 辅助电源输出的地
3	RXD/TXD-P	接收/发送数据正端，B 线
4	CNTR-P	方向控制信号正端
5	DGND	数据基准电位(地)
6	VP	供电电压正端
7	P24	24 V 辅助电源输出正端
8	RXD/TXD-N	接收/发送数据负端，A 线
9	CNTR-N	方向控制信号负端

3. 总线终端器

在数据线 A 和 B 的两端均应加接总线终端器(见图 8-1)。总线终端器的下拉电阻与数据基准电位 DGND 相连，上拉电阻与供电正电压 VP 相连。总线上没有站发送数据时，这两个电阻确保总线上有一个确定的空闲电位。几乎所有标准的 PROFIBUS 总线连接器上都集成了总线终端器，可以由跳接器或开关来选择是否使用它。

传输速率大于 1 500 kbit/s 时，由于连接的站的电容性负载引起导线反射，因此必须使用附加有轴向电感的总线连接插头。

4. DP/FMS 的光纤电缆传输

PROFIBUS 另一种物理层通过光纤中光的传输来传送数据。单芯玻璃光纤的最大连接距离为 15 km，价格低廉的塑料光纤为 80 m。光纤电缆对电磁干扰不敏感，并能确保站之间的电气隔离。近年来，由于光纤的连接技术已大大简化，这种传输技术已经广泛地用于现场设备的数据通信。

光链路模块(OLM)用来实现单光纤环和冗余的双光纤环。在单光纤环中，OLM 通过单工光纤电缆相互连接，如果光纤电缆断线或 OLM 出现故障，整个环路将崩溃。在冗余的双光纤环中，OLM 通过两个双工光纤电缆相互连接，如果两根光纤线中的一根出了故障，总线系统将自动地切换为线性结构。光纤导线中的故障排除后，总线系统即返回到正常的冗余环状态。许多厂商提供专用总线插头来转换 RS-485 信号和光纤导体信号。

5. PA 的 IEC 1158-2 传输

PROFIBUS-PA 采用符合 IEC 1158-2 标准的传输技术，这种技术确保本质安全，并通过总线直接给现场设备供电，能满足石油化工业的要求。数据传输使用非直流传输的位同步、曼彻斯特编码线协议（也称 H1 编码）。用曼彻斯特编码传输数据时，从 0（－9 mA）到 1（+9 mA）的上升沿发送二进制数“0”，从 1 到 0 的下降沿发送二进制数“1”。传输速率为 31.25 kbit/s。传输介质为屏蔽或非屏蔽的双绞线，允许使用线性、树形和星形网络。总线段的两端用一个无源的 RC 线终端器来终止（100 Ω 电阻与 1 μF 电容的串联电路），在一个 PA 总线段上最多可以连接 32 个站，总数最多为 126 个，最多可以扩展 4 台中继器。最大的总线段长度取决于供电装置、导线类型和所连接的站的电流消耗。

为了增加系统的可靠性，总线段可以用冗余总线段作备份。段耦合器或 DP/PA 链接器用于 PA 总线段与 DP 总线段的连接。

8.1.3 PROFIBUS-DP 设备的分类

PROFIBUS-DP 设备可以分为以下三种不同类型的设备：

1. 1 类 DP 主站

1 类 DP 主站（DPM1）是系统的中央控制器，DPM1 在预定的周期内与分布式的站（例如 DP 从站）循环地交换信息，并对总线通信进行控制和管理。DPM1 可以发送参数给从站，读取 DP 从站的诊断信息，用 Global_Control（全局控制）命令将它的运行状态告知给各 DP 从站。此外，还可以将控制命令发送给个别从站或从站组，以实现输出数据和输入数据的同步。下列设备可以做 1 类 DP 主站：

（1）集成了 DP 接口的 PLC，例如 CPU 315-2DP，CPU 313C-2DP 等。

（2）没有集成 DP 接口的 CPU 加上支持 DP 主站功能的通信处理器（CP）。

（3）插有 PROFIBUS 网卡的 PC，例如 WinAC 控制器。用软件功能选择 PC 做 1 类主站或是做编程监控的 2 类主站，可以使用 CP 5411、CP 5511 和 CP 5611 等网卡。

（4）IE/PB 链路模块。

（5）ET 200S/ET 200X 的主站模块。

2. 2 类 DP 主站

2 类 DP 主站（DPM2）是 DP 网络中的编程、诊断和管理设备。DPM2 除了具有 1 类主站的功能外，在与 1 类 DP 主站进行数据通信的同时，可以读取 DP 从站的输入/输出数据和当前的组态数据，可以给 DP 从站分配新的总线地址。下列设备可以做 DPM2：

（1）以 PC 为硬件平台的 2 类主站

PC 加 PROFIBUS 网卡可做 2 类主站。西门子公司为其自动化产品设计了专用的编程设备。不过现在一般都用通用的 PC 和 STEP 7 编程软件来做编程设备，用 PC 和 WinCC 组态软件做监控操作站。

（2）操作员面板/触摸屏（OP/TP）

操作员面板用于操作人员对系统的控制和操作，例如参数的设置与修改、设备的起动和停机，以及在线监视设备的运行状态等。有触摸按键的操作员面板俗称触摸屏，它们在工业控制中得到了广泛的应用。西门子公司提供了不同大小和功能的 TP 和 OP 供用户选用。

3. DP 从站

DP 从站是进行输入信息采集和输出信息发送的外围设备，它只与组态它的 DP 主站交换用户数据，可以向该主站报告本地诊断中断和过程中断。

(1) 分布式 I/O

分布式 I/O(非智能型 I/O)没有程序存储和程序执行功能，通信适配器用来接收主站的指令，按主站指令驱动 I/O，并将 I/O 输入及故障诊断等信息返回给主站。通常分布式 I/O 由主站统一编址，对主站编程时使用分布式 I/O 与使用主站的 I/O 没有什么区别。

ET 200 是西门子的分布式 I/O，有 ET200M/B/L/X/S/is/eco/R 等多种类型，详细的性能指标见 2.8 节中的内容。它们都有 PROFIBUS-DP 通信接口，可以作 DP 网络的从站。

(2) PLC 智能 DP 从站(I 从站)

PLC(智能型 I/O)可以做 PROFIBUS 的从站。PLC 的 CPU 通过用户程序驱动 I/O，在 PLC 的存储器中有一片特定区域作为与主站通信的共享数据区，主站通过通信间接控制从站 PLC 的 I/O。

(3) 具有 PROFIBUS-DP 接口的其他现场设备

西门子的 SINUMERIK 数控系统、SITRANS 现场仪表、MicroMaster 变频器、SIMOREG DC-MASTER 直流传动装置都有 PROFIBUS-DP 接口或可选的 DP 接口，可以做 DP 从站。

其他公司支持 DP 接口的输入/输出、传感器、执行器或其他智能设备，也可以接入 PROFIBUS-DP 网络。

4. DP 组合设备

可以将 1 类、2 类 DP 主站或 DP 从站组合在一个设备中，形成一个 DP 组合设备。例如第 1 类 DP 主站与第 2 类 DP 主站的组合，DP 从站与第 1 类 DP 主站的组合。

5. PROFIBUS 网络部件

网络部件包括通信介质(电缆)、总线部件(总线连接器、中继器、耦合器、链路)和网络转接器，后者包括 PROFIBUS 与串行通信、以太网、AS-i 和 EIB 通信网络的转接器。

8.1.4 PROFIBUS 通信处理器

1. CP 342-5 通信处理器

CP 342-5 是将 SIMATIC S7-300 和 S7 系列 PLC 连接到 PROFIBUS-DP 总线系统的低成本的 DP 主/从站接口模块。它减轻了 CPU 的通信负担，通过 FOC 接口可以直接连接到光纤 PROFIBUS 网络，最高通信速率 12 M bit/s。

CP 342-5 提供下列通信服务：PROFIBUS-DP、S7 通信、S5 兼容通信功能和 PG/OP(编程器/操作员面板)通信，通过 PROFIBUS 进行配置和编程。

9 针 D 型插座连接器用于连接 PROFIBUS 总线，4 针端子用于连接外部 DC 24 V 电源，通过接口模块 IM 360/361，CP 342-5 也可以工作在扩展机架上。

CP 342-5 作为 DP 主站自动处理数据传输，通过它将 DP 从站(例如 ET 200)连接到 S7-300。CP 342-5 提供 SYNC(同步)、FREEZE(锁定)和共享输入/输出功能。CP 342-5 也可以作为 DP 从站，允许 S7-300 与其他 PROFIBUS 主站交换数据。这样可以进行 S5/S7、PC、ET 200 和其他现场设备的混合配置。

CP 342-5 的 S7 通信功能用于在 S7 系列 PLC 之间、PLC 与计算机和人机接口(操作员面

板)之间通信。通过 CP 342-5 可以对所有连接到网络上的 S7 站进行远程编程和远程组态。

用嵌入 STEP 7 的 NCM S7 软件对 CP 342-5 进行配置,CP 模块的配置数据存放在 CPU 中,CPU 起动后自动地将配置参数传送到 CP 模块。

PROFIBUS-DP 的功能块包含在 STEP 7 的标准库中。安装 NCM S7 后,用于 S5 兼容通信(发送/接收)的功能块保存在 SIMATIC NET 库中。

2. CP 342-5 FO 通信处理器

CP 342-5 FO 是带光纤接口的 PROFIBUS-DP 主站或从站模块,用于将 S7-300 和 C7 连接到 PROFIBUS,最高传输速率为 12 Mbit/s。通过内置的 FOC 光纤电缆接口直接连接到光纤 PROFIBUS 网络,即使有强烈的电磁干扰也能正常工作。模块的其他性能与 CP 342-5 相同。

S7-300 和 C7 可以与下列部件进行通信:

(1) 带集成光纤接口的 ET 200 I/O;

(2) 带 CP 5613 FO/5614 FO 的 PC;

(3) 使用 IM 467 FO 和 CP 325-5 FO 可以进行 S7-300 和 400 之间的通信;

(4) 使用光纤总线端子(OBT)可与其他 PROFIBUS 节点通信。

3. CP 443-5 通信处理器

CP 443-5 是用于 PROFIBUS-DP 总线的通信处理器,它提供下列通信服务:S7 通信,S5 兼容通信,与计算机、PG/OP 的通信和 PROFIBUS-FMS。可以通过 PROFIBUS 进行配置和远程编程,实现实时钟的同步,在 H 系统中实现冗余的 S7 通信或 DP 主站通信。通过 S7 路由器在网络间进行通信。

CP 443-5 分为基本型和扩展型,扩展型作为 DP 主站运行,支持 SYNC 和 FREEZE 功能、从站到从站的直接通信和通过 PROFIBUS-DP 发送数据记录等。

4. 用于 PC/PG 的通信处理器

用于 PC/PG 的通信处理器(见表 8-2)将工控机/编程器连接到 PROFIBUS 网络中,支持标准 S7 通信、S5 兼容通信、PG/OP 通信和 PROFIBUS-FMS,OPC 服务器随通信软件供货。

表 8-2 用于 PC/PG 的通信处理器

	CP 5613 / CP 5613FO	CP 5614 / CP 5614FO	CP 5611
可以连接的 DP 从站数	122	122	60
可以并行处理的 FDL 任务数	120	120	100
PG/PC 和 S7 的连接数	50	50	8
FMS 的连接数	40	40	-

CP 5613 是带微处理器的 PCI 卡,有一个 PROFIBUS 接口,仅支持 DP 主站。

CP 5614 用于将工控机连接到 PROFIBUS,有两个 PROFIBUS 接口,可以将两个 PROFIBUS 网络连接到 PC,网络间可以交换数据。可以作 DP 主站或 DP 从站。

CP 5613 FO/CP 5614 FO 有光纤接口,用于将 PC/PG 连接到光纤 PROFIBUS 网络。

CP 5611 用于将带 PCMCIA 插槽的笔记本电脑连接到 PROFIBUS 和 S7 的 MPI。有一个 PROFIBUS 接口,支持 PROFIBUS 主站和从站。

8.1.5 GSD 电子设备数据文件

GSD 是可读的 ASCII 码文本文件,包括通用的和与设备有关的通信的技术规范。为了将

不同厂家生产的 PROFIBUS 产品集成在一起，生产厂家必须以 GSD 文件(电子设备数据库文件)方式提供这些产品的功能参数，例如 I/O 点数、诊断信息、传输速率、时间监视等。标准的 GSD 数据将通信扩大到操作员控制级。GSD 文件分为三个部分：

(1) 总规范：包括生产厂商和设备名称、硬件和软件版本、传输速率、监视时间间隔、总线插头指定信号。

(2) 主站规范：包括适用于主站的各项参数，例如最大可以连接的从站个数和上载/下载的选项。

(3) 与 DP 从站有关的规范：例如输入/输出通道个数、类型和诊断数据等。

在 STEP 7 的 SIMATIC 管理器中打开硬件组态工具 HW Config，如果 DP 从站没有在“Hardware Catalog”窗口中出现，应安装制造商提供的 GSD 文件。GSD 文件可以在制造商的网站下载，例如在西门子中文网站的下载中心(http://www.ad.siemens.com.cn/download)，可以下载《西门子自动化产品 GSD 文件大全》。

在硬件组态工具中，执行菜单命令“OPTIONS→INSTALL NEW GSD…”，在出现的对话框中打开 GSD 文件所在的文件夹，安装 GSD 文件。安装成功后，在“Hardware Catalog”窗口的“\ PROFIBUS-DP \ Other Field Devices”文件夹，可以看到刚安装了 GSD 文件的 DP 从站，并将它用于组态。STEP 7 将 GSD 文件存储在“… \ Siemens \ Step7 \ S7 data \ Gsd”文件夹中。

STEP 7 把系统配置的所有的 GSD 文件保存在项目中，可以将它们传送到另一个项目中。只能在离线状态输入 GSD 文件。执行菜单命令“Options”→“Import station GSD…”，在打开的对话框中选择要从哪里输入 GSD 文件的项目和站，当需要的站的名称出现在“Object name”框内时，点击【OK】按钮确认。

8.2 PROFIBUS 的通信协议

8.2.1 PROFIBUS 的数据链路层

1. 总线存取方式

根据 OSI 参考模型，第 2 层(数据链路层)规定总线存取控制、数据安全性以及传输协议和报文的处理，三种 PROFIBUS (DP、FMS、PA) 均使用一致的总线存取协议。PROFIBUS 协议结构如图 8-2 所示。

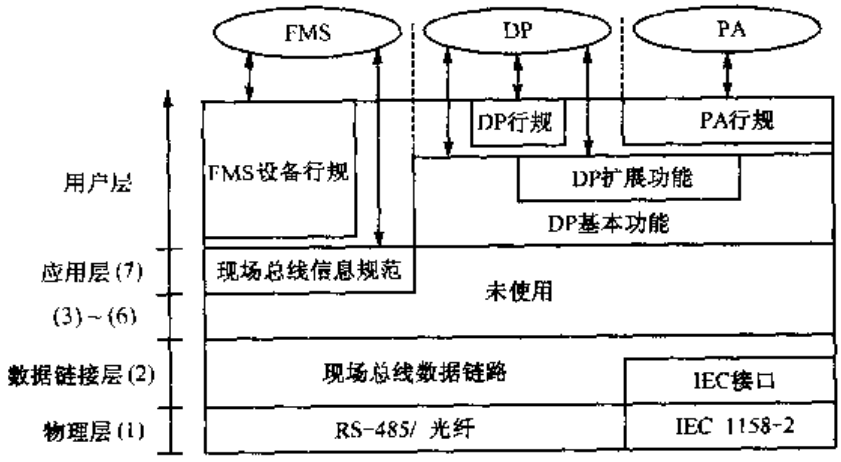


图 8-2 PROFIBUS 协议结构

在 PROFIBUS 中，第 2 层称为现场总线数据链路层(FDL，Fieldbus Data Link)。介质存取控制 MAC(Medium Access Control)具体控制数据传输的程序，MAC 必须确保在任何一个时刻只有一个站点发送数据。

PROFIBUS 协议的设计满足介质控制的两个基本要求：

(1) 在复杂的自动化系统(主站)间的通信，必须保证在确切限定的时间间隔中，任何一个站点有足够的时间来完成通信任务。

(2) 在复杂的 PLC 或 PC 和简单的 I/O 外围设备(从站)间的通信，应尽可能简单快速地完成数据的实时传输，因通信协议增加的数据传输时间应尽量少。

PROFIBUS 采用混合的总线存取控制机制来实现上述目标(见图 8-3)。它包括主站(Master)之间的令牌(Token)传递方式和主站与从站(Slave)之间的主-从方式。令牌实际上是一条特殊的报文，它在所有的主站上循环一周的时间是事先规定的。主站之间构成令牌逻辑环，令牌传递仅在各主站之间进行。令牌按令牌环中各主站地址的升序在各主站之间依次传递。

当某主站得到令牌报文后，该主站可以在一定时间内执行主站工作。在这段时间内，它可以依照主-从通信关系表与所有从站通信，也可以依照主-主通信关系表与所有主站通信。令牌传递程序保证每个主站在一个确切规定的时间内得到总线存取权(即令牌)。

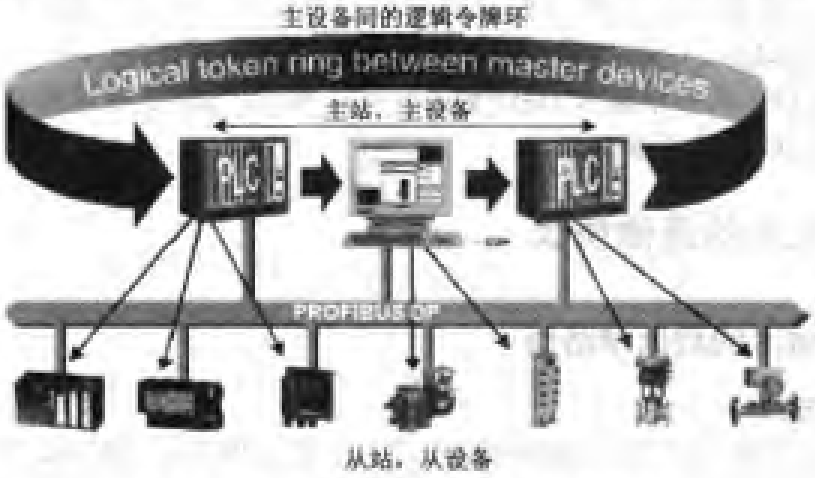


图 8-3 PROFIBUS 现场总线的总线存取方式

在总线初始化和起动阶段，主站介质存取控制 MAC 通过辨认主动节点(主站)来建立令牌环，首先自动地判定总线上所有主动节点的地址，并将它们的节点地址记录在主站表中。在总线运行期间，从令牌环中去掉有故障的主动节点，将新上电的主动节点加入到令牌环中。

PROFIBUS 介质存取控制还可以监视传输介质和收发器是否有故障，检查站点地址是否出错(例如地址重复)，以及令牌是否丢失或有多个令牌。

令牌经过所有主动节点轮转一次所需的时间称为令牌轮转时间。用可调整的令牌时间 T_{TR}(目标令牌时间)来规定令牌轮转一次允许的最大时间。PROFIBUS 在第 2 层按照非连接的模式操作，除提供点对点逻辑数据传输外，还提供多点通信，其中包括广播及选择广播功能。

DP 主站与 DP 从站间的通信基于主-从原理，DP 主站按轮询表依次访问 DP 从站，主站与从站间周期性地交换用户数据。DP 主站与 DP 从站间的一个报文循环由 DP 主站发出的请求帧(轮询报文)和由 DP 从站返回的有关应答或响应帧组成。

这种总线存取方式可以实现下列系统配置：

(1) 纯主-主系统(多主站)。

(2) 纯主-从系统(单主站)，系统有若干个被动节点(从站)和一个主动节点(主站)。主站循环地发送信息给从站或由从站获取信息。

(3) 两种方式的组合(多主站系统)，总线上连有多个主站，它们与各自的从站构成相互独立的子系统。每个子系统包括一个 DPM1、指定的若干从站及可能的 DPM2 设备。任何一个主站均可以读取 DP 从站的输入/输出映像，但是只有一个指定的 DPM1 允许对 DP 从站写入数据。

2. 数据链路层的报文格式

第 2 层的报文格式如图 8-4 所示，所有报文均具有海明距离 HD = 4，即在数据报文中，可以检查出最多 3 个同时出错的位。这一要求是通过使用国际标准 IEC 870-5-1 的规定、选择特殊的报文起始和终止标识符、使用无间隙同步以及使用奇偶校验位和控制字节来实现的。可以检查出下列类型的错误：字符格式出错(奇偶校验、溢出、帧出错)、协议出错、起始和终止定界符出错、帧校验字节出错和报文长度出错。

出错的报文至少被自动重发一次，在第 2 层中，重发次数最多可设定为 8 次。除点对点的数据传输外，第 2 层还允许进行广播和群播通信的多点传输。

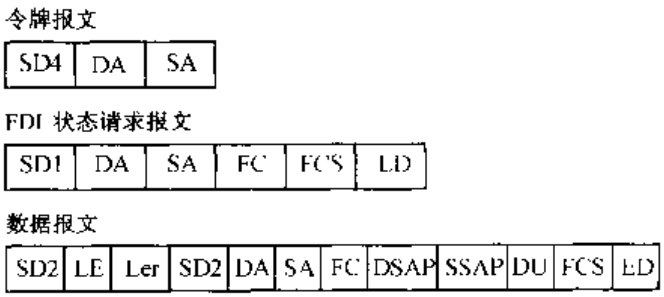


图 8-4 PROFIBUS 的报文结构

在广播通信中，一个主站发送信息给所有其他的主站和从站。在群播通信中，一个主站发送信息给一组站(主站和从站)，数据的接收不需应答。

报文中符号的意义见表 8-3。

表 8-3 报文中符号的意义

符 号	意 义
DA	接收报文的目的站地址字节(Destination Address)
DU	包含报文有用信息的数据单元(Data Unit)
DSAP	目的服务存取点(Destination Service Access Point)
ED	结束分界符字节(End Delimiter)
FC	功能码字节(Function Code)
FCS	帧校验序列字节(Frame Check Sequence)
LE	报文长度字节(Length)
Ler	重复长度字节(Repeated Length)
SA	发送报文的源站地址字节(Source Address)
SD1 - SD4	起始分界符，用于区分不同的报文格式
SSAP	源服务存取点(Source Service Access Point)

上一层通过第 2 层的 SAP(服务存取点)调用这些服务。对于 PROFIBUS-FMS 来说,这些服务存取点被用来编址逻辑通信关系。在 PROFIBUS-DP 和 PROFIBUS-PA 中,每一个服务存取点分配一个确定的功能。所有主站和从站允许同时使用若干个服务存取点。服务存取点分为 SSAP(源服务存取点)和 DSAP(目的服务存取点)。

8.2.2 PROFIBUS-DP

在 PROFIBUS 现场总线中,PROFIBUS-DP 的应用最广。DP 协议主要用于 PLC 与分布式 I/O 和现场设备的高速数据通信。典型的 DP 配置是单主站结构,也可以是多主站结构。

DP 的功能经过扩展,一共有 3 个版本:DP-V0、DP-V1 和 DP-V2。有的用户手册将 DP-V1 简写为 DPV1。

1. **基本功能(DP-V0)**

(1) 总线存取方法

各主站间为令牌传送,主站与从站间为主-从循环传送,支持单主站或多主站系统,总线上最多 126 个站。可以采用点对点用户数据通信、广播(控制指令)方式和循环主-从用户数据通信。

(2) 循环数据交换

DP-V0 可以实现中央控制器(PLC,PC 或过程控制系统)与分布式现场设备(从站,例如 I/O、阀门、变送器和分析仪等)之间的快速循环数据交换,主站发出请求报文,从站收到后返回响应报文。这种循环数据交换是在被称为 MS0 的连接上进行的。

总线循环时间应小于中央控制器的循环时间(约 10 ms),DP 的传送时间与网络中站的数量和传输速率有关。每个从站可以传送 224 B 的输入或输出。

(3) 诊断功能

经过扩展的 PROFIBUS-DP 诊断,能对站级、模块级、通道级这 3 级故障进行诊断和快速定位,诊断信息在总线上传输并由主站采集。

本站诊断操作:对本站设备的一般操作状态的诊断,例如温度过高、压力过低;

模块诊断操作:对站点内部某个具体的 I/O 模块的故障定位;

通道诊断操作:对某个输入/输出通道的故障定位。

(4) 保护功能

所有信息的传输按海明距离 HD=4 进行。对 DP 从站的输出进行存取保护,DP 主站用监控定时器监视与从站的通信,对每个从站都有独立的监控定时器。在规定的监视时间间隔内,如果没有执行用户数据传送,将会使监控定时器超时,通知用户程序进行处理。如果参数“Auto_Clear”为 1,DPM1 将退出运行模式,并将所有有关的从站的输出置于故障安全状态,然后进入清除(Clear)状态。

DP 从站用看门狗(Watchdog Timer,监控定时器)检测与主站的数据传输,如果在设置的时间内没有完成数据通信,从站自动地将输出切换到故障安全状态。

在多主站系统中,从站输出操作的访问保护是必要的。这样可以保证只有授权的主站才能直接访问。其他从站可以读它们的输入的映像,但是不能直接访问。

(5) 通过网络的组态功能与控制功能

通过网络可以实现下列功能:动态激活或关闭 DP 从站,对 DP 主站(DPM1)进行配置,可

以设置站点的数目、DP 从站的地址、输入/输出数据的格式、诊断报文的格式等，以及检查 DP 从站的组态。控制命令可以同时发送给所有的从站或部分从站。

(6) 同步与锁定功能

主站可以发送命令给一个从站或同时发送给一组从站。接收到主站的同步命令后，从站进入同步模式。这些从站的输出被锁定在当前状态。在这之后的用户数据传输中，输出数据存储在从站，但是它的输出状态保持不变。同步模式用“UNSYNC”命令来解除。

锁定(FREEZE)命令使指定的从站组进入锁定模式，即将各从站的输入数据锁定在当前状态，直到主站发送下一个锁定命令时才可以刷新。用“UNFREEZE”命令来解除锁定模式。

(7) DPM1 和 DP 从站之间的循环数据传输

DPM1 与有关 DP 从站之间的用户数据传输是由 DPM1 按照确定的递归顺序自动进行的。在对总线系统进行组态时，用户定义 DP 从站与 DPM1 的关系，确定哪些 DP 从站被纳入信息交换的循环。

DMP1 和 DP 从站之间的数据传送分为 3 个阶段：参数化、组态和数据交换。在前两个阶段进行检查，每个从站将自己的实际组态数据与从 DPM1 接收到的组态数据进行比较：设备类型、格式、信息长度与输入/输出的个数都应一致，以防止由于组态过程中的错误造成系统的检查错误。

只有系统检查通过后，DP 从站才进入用户数据传输阶段。在自动进行用户数据传输的同时，也可以根据用户的需要向 DP 从站发送用户定义的参数。

(8) DPM1 和系统组态设备间的循环数据传输

PROFIBUS-DP 允许主站之间的数据交换，即 DPM1 和 DPM2 之间的数据交换。该功能使组态和诊断设备通过总线对系统进行组态，改变 DPM1 的操作方式，动态地允许或禁止 DPM1 与某些从站之间交换数据。

2. DP-V1 的扩展功能

(1) 非循环数据交换

除了 DP-V0 的功能外，DP-V1 最主要的特征是具有主站与从站之间的非循环数据交换功能，可以用它来进行参数设置、诊断和报警处理。非循环数据交换与循环数据交换是并行执行的，但是优先级较低。

1 类主站 DPM1 可以通过非循环数据通信读写从站的数据块，数据传输在 DPM1 建立的 MS1 连接上进行，可以用主站来组态从站和设置从站的参数。

在起动非循环数据通信之前，DPM2 用初始化服务建立 MS2 连接。MS2 用于读、写和数据传输服务。一个从站可以同时保持几个激活的 MS2 连接，但是连接的数量受到从站的资源的限制。DPM2 与从站建立或中止非循环数据通信连接，读写从站的数据块。数据传输功能向从站非循环地写指定的数据，如果需要，可以在同一周期读数据。

对数据寻址时，PROFIBUS 假设从站的物理结构是模块化的，即从站由称为“模块”的逻辑功能单元构成。在基本 DP 功能中这种模型也用于数据的循环传送。每一模块的输入/输出字节数为常数，在用户数据报文中按固定的位置来传送。寻址过程基于标识符，用它来表示模块的类型，包括输入、输出或二者的结合，所有标识符的集合产生了从站的配置。在系统起动时由 DPM1 对标识符进行检查。

循环数据通信也是建立在这一模型的基础上的。所有能被读写访问的数据块都被认为属

于这些模块，它们可以用槽号和索引来寻址。槽号用来确定模块的地址，索引号用来确定指定给模块的数据块的地址，每个数据块最多 244 B(见图 8-5)。

对于模块化的设备，模块被指定槽号，从 1 号槽开始，槽号按顺序递增，0 号留给设备本身。紧凑型设备被视为虚拟模块的一个单元，也可以用槽号和索引来寻址。

在读/写请求中通过长度信息可以对数据块的一部分进行读写。如果读/写数据块成功，DP 从站发送正常的读写响应。反之将发送否定的响应，并对问题进行分类。

(2) 工程内部集成的 EDD 与 FDT

在工业自动化中，由于历史的原因，GSD(电子设备数据)文件使用得较多，它适用于较简单的应用；EDD(Electronic Device Description，电子设备描述)适用于中等复杂程序的应用；FDT/DTM(Field Device Tool/Device Type manager，现场设备工具/设备类型管理)是独立于现场总线的“万能”接口，适用于复杂的应用场合。

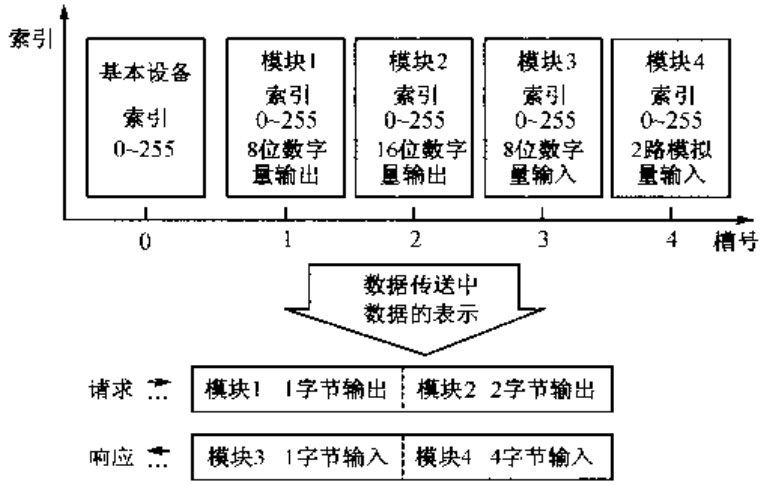


图 8-5 读写服务寻址

(3) 基于 IEC 61131-3 的软件功能块

为了实现与制造商无关的系统行规，应为现存的通信平台提供应用程序接口(API)，即标准功能块。PNO(PROFIBUS 用户组织)推出了“基于 IEC 61131-3 的通信与代理(Proxy)功能块”

(4) 故障-安全通信(PROFIsafe)

PROFIsave 定义了与故障-安全有关的自动化任务，以及故障-安全设备怎样用故障-安全控制器在 PROFIBUS 上通信。PROFIsave 考虑了在串行总线通信中可能发生的故障，例如数据的延迟、丢失、重复，不正确的时序、地址和数据的损坏。

PROFIsave 采取了下列的补救措施：输入报文帧的超时及其确认；发送者与接收者之间的标识符(口令)；附加的数据安全措施(CRC 校验)。

(5) 扩展的诊断功能

DP 从站通过诊断报文将突发事件(报警信息)传送给主站，主站收到后发送确认报文给从站。从站收到后只能发送新的报警信息，这样可以防止多次重复发送同一报警报文。状态报文由从站发送给主站，不需要主站确认。

3. DP-V2 的扩展功能

(1) 从站与从站之间的通信

在2001年发布的PROFIBUS协议功能扩充版本DP-V2中，广播式数据交换实现了从站之间的通信，从站作为出版者(Publisher)，不经过主站直接将信息发送给作为订户(Subscribers)的从站(见图8-6)。这样从站可以直接读入别的从站的数据。这种方式最多可以减少90%的总线响应时间。

(2) 同步(Isochronous)模式功能

同步功能激活主站与从站之间的同步，误差小于1 ms。通过"全局控制"广播报文，所有有关的设备被周期性地同步到总线主站的循环。

(3) 时钟控制与时间标记(Time Stamps)

通过用于时钟同步的新的连接MS3，实时时间(Real Time)主站将时间标记发送给所有的从站，将从站的时钟同步到系统时间，误差小于1 ms。利用这一功能可以实现高精度的事件追踪。在有大量主站的网络中，对于获取定时功能特别有用。主站与从站之间的时钟控制通过MS3服务来进行。

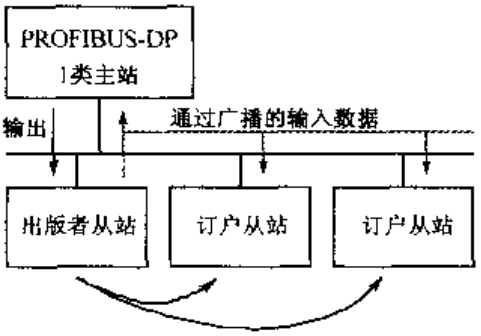


图8-6 从站与从站的数据交换

(4) HARTonDP

HART是一种应用较广的现场总线。HART规范将HART的客户-主机-服务器模型映射到PROFIBUS，HART规范位于DP主站和从站的第7层之上，HART-client(客户)功能集成在PROFIBUS的主站中，HART的主站集成在PROFIBUS的从站中。为了传送HART报文，定义了独立于MS1和MS2的通信通道。

(5) 上载与下载(区域装载)

这一功能允许用少量的命令装载任意现场设备中任意大小的数据区。例如不需要人工装载就可以更新程序或更换设备

(6) 功能请求(Function Invocation)

功能请求服务用于DP从站的程序控制(起动、停止、返回或重新起动)和功能调用。

(7) 从站冗余

在很多应用场合，要求现场设备的通信有冗余功能。冗余的从站有两个PROFIBUS接口，一个是主接口，一个是备用接口。它们可能是单独的设备，也可能分散在两个设备中。这些设备有两个带有特殊的冗余扩展的独立的协议堆栈，冗余通信在两个协议堆栈之间进行，可能是在一个设备内部，也可能是在两个设备之间。

在正常情况下，通信只发送给被组态的主要从站，它也发送给后备从站。在主要从站出现故障时，后备从站接管它的功能。可能是后备从站自己检查到故障，或主站请求它这样做。主站监视所有的从站，出现故障时立即发送诊断报文给后备从站。

冗余从站设备可以在一条PROFIBUS总线或两条冗余的PROFIBUS总线上运行。

4. PROFIBUS-DP的运行方式

PROFIBUS-DP的系统行为主要取决于DPM1的操作状态，这些状态由本地或总线上的组态设备控制，主要有以下三种状态：

(1) 停止状态：DPM1和DP从站之间没有数据传输，有诊断和参数设置功能。

(2) 清除(CLEAR)状态：DPM1读取DP从站的输入信息，并使输出信息保持在故障安全状态。

(3) 运行状态:DPM1 处于数据传输状态,通过循环数据通信,DPM1 从 DP 从站读取输入信息并向从站写入输出信息。

DPM1 设备在一个预先设定的时间间隔内,以有选择的广播方式将它的状态周期性地发送到每个指定的 DP 从站。

如果在 DPM1 的数据传输过程中发生错误,例如一个 DP 从站有故障,且 DPM1 的组态参数自动清除(Auto-Clear)为 1,DPM1 立即将所有有关的 DP 从站的输出数据转入清除状态,DP 从站将不再发送用户数据,然后 DPM1 转入清除状态。如果该参数为 0,在 DP 从站出现错误时,DPM1 仍停留在运行状态,由用户进行处理。

5. PROFIBUS-DP 行规

PROFIBUS-DP 协议明确地规定了用户数据怎样在总线各站之间传递,用户数据的含义在 PROFIBUS 行规中具体说明。另外,行规还具体规定了 PROFIBUS-DP 如何用于应用领域。使用行规可以使不同厂商生产的不同设备互换使用,而工厂操作人员不必关心两者之间的差异,因为与应用有关的含义在行规中均作了准确的规定说明。下面是 PROFIBUS-DP 行规:

(1) PROFIdrive:PROFIBUS 用于各种电气传动的行规;

(2) PA 设备(PA Devices):PROFIBUS 用于过程自动化的行规;

(3) Robots/NC:PROFIBUS 用于装配机器人的行规;

(4) Panel Devices:人机接口(HMI)行规;

(5) Encoders:单圈或多圈旋转、角度、直线编码器接口的行规;

(6) Fluid Power:液压驱动的行规;

(7) SEMI:PROFIBUS 用于半导体制造业的行规;

(8) Low-Voltage Switchgear:PROFIBUS-DP 用于低电压开关设备(开关断路器、汽车起动器等)的行规;

(9) Dosing/Weighing:PROFIBUS-DP 用于配料与称重的行规;

(10) Ident Systems:PROFIBUS 用于识别目的(例如条形码和无线发射应答器)的行规;

(11) Liquid Pumps:PROFIBUS 用于液体泵的行规;

(12) Remote I/O for PA Devices:PA 设备的远程 I/O。

8.2.3 PROFIBUS-PA

1. PROFIBUS-PA

PROFIBUS-PA 适用于过程自动化,PA 将自动化系统和过程控制系统与压力、温度和液位变送器等现场设备连接起来,用来替代 4~20 mA 的模拟技术。PA 具有下列特性:

(1) 适合过程自动化应用的行规使不同厂家生产的现场设备具有互换性。

(2) 即使在本征安全区增加和去除总线站点,也不会影响到其他站。

(3) PROFIBUS-PA 段与 PROFIBUS-DP 总线段之间通过耦合器连接,可以实现两段间的透明通信。

(4) 使用与 IEC 1158-2 技术相同的双绞线完成远程供电和数据传送。

(5) 在潜在的爆炸危险区可以使用防爆型"本征安全"或"非本征安全"功能。

2. PROFIBUS-PA 的传输协议

PROFIBUS-PA 采用 PROFIBUS-DP 的基本功能来传送测量值和状态。并用扩展的 PROFIBUS-DP 功能来制订现场设备的参数和进行设备操作。PROFIBUS-PA 的第一层采用 IEC 1158-2 技术。

3. PROFIBUS-PA 设备行规

PROFIBUS-PA 行规保证了不同厂商生产的现场设备的互换性和互操作性，它是 PROFIBUS-PA 的一个组成部分。

PA 行规已对所有通用的测量变送器和其他一些设备类型作了具体规定，这些设备包括压力、液位、温度和流量变送器，数字量输入和输出，模拟量输入和输出，阀门和定位器等。

8.2.4 PROFIBUS-FMS

PROFIBUS-FMS 用于解决车间监控级通信。在这一层，中央控制器（例如 PLC、PC 等）之间需要比现场层更大量的数据传送，但通信的实时性要求低于现场层。

1. PROFIBUS-FMS 应用层

应用层提供了用户使用的通信服务，包括访问变量、程序传递、事件控制等。应用层包括下列两部分：

（1）现场总线报文规范（Fieldbus Message Specification-FMS）：描述了通信对象和应用服务。

（2）低层接口（Lower Layer Interface-LLI）：FMS 服务到第二层的接口。

2. PROFIBUS-FMS 通信模型

PROFIBUS-FMS 利用通信将分散的应用过程统一到一个共用的过程。在应用过程中，现场设备中用来通信的那部分应用过程称为虚拟现场设备 VFD（Virtual field Device）。在实际现场设备与 VFD 之间设立一个通信关系表，该表是 VFD 通信变量的集合，例如元件数、故障率和停机时间等。VFD 通过通信关系表完成对实际现场设备的通信。

3. 通信对象与通信字典

FMS 面向对象通信，它确认 5 种静态通信对象：简单变量、数组、记录、域和事件，还确认两种动态通信对象：程序调用和变量表。

每个 FMS 设备的所有通信对象都填入对象字典（OD）。对于简单设备，OD 可以预定义，对于复杂设备，OD 可以在本地或远程通过组态加到设备中去。静态通信对象进入静态对象字典，动态通信对象进入动态通信字典。每个对象均有一个惟一的索引，为避免非授权存取，每个通信对象可以选用存取保护。

4. PROFIBUS-FMS 服务

FMS 服务项目是 ISO 9506 制造信息规范 MMS（Manufacturing Message Specification）服务项目的子集。这些服务项目在现场总线应用中已被优化，而且还加上了通信对象的管理和网络管理。

PROFIBUS-FMS 提供大量的管理和服务，满足了不同设备对通信提出的广泛需求，服务项目的选用取决于特定的应用，具体的应用领域在 FMS 行规中规定。

5. 低层接口（LLI）

第七层到第二层服务的映射由 LLI 来解决，其主要任务包括数据流控制和连接监视。用

户通过称为通信关系的逻辑通道与其他应用过程进行通信。FMS 设备的全部通信关系都列入通信关系表 CRL(Communication Relationship List)。每个通信关系通过通信索引(CREF)来查找,CRL 中包含了 CREF 和第二层及 LLI 地址间的关系。

6. 网络管理

FMS 还提供网络管理功能,由现场总线管理层来实现。其主要功能有上、下关系管理、配置管理、故障管理等。

7. PROFIBUS-FMS 行规

为了使 FMS 通信服务适应实际需要的功能范围和定义符合实际应用的设备功能,PNO(PROFIBUS 用户组织)制定了 FMS 行规,由它们保证不同制造商生产的同类设备具有相同的通信功能。目前已制定了以下的 FMS 行规:

(1) 控制器(PLC)之间的通信行规(3.002):定义了用于 PLC 之间的 FMS 服务,根据控制器的等级对 PLC 必须支持的服务、参数和数据类型作了规定。

(2) 楼宇服务自动化的行规(3.011):描述怎样通过 FMS 来处理监视、开闭环控制、操作员控制、报警和楼宇服务自动化系统的归档等.

(3) 低压开关设备行规(3.032):规定了使用 FMS 数据通信时低压开关设备的特性。

8.2.5 PROFIBUS 网络的配置方案

1. 现场设备分类

根据是否有 PROFIBUS 接口,可以将现场设备分为三种类型:

(1) 现场设备不具备 PROFIBUS 接口,采用分布式 I/O 作为总线接口与现设备连接。如果现场设备可以分为相对集中的若干组,这种模式能更好地发挥现场总线技术的优点。

(2) 现场设备都有 PROFIBUS 接口,这是一种理想情况。可以使用现场总线技术,实现完全的分布式结构。就目前来看,这种方案的设备成本较高。

(3) 只有部分现场设备有 PROFIBUS 接口,这是一种相当普遍的情况。应采用有 PROFIBUS 接口的现场设备与分布式 I/O 混合使用的办法。

2. PROFIBUS-DP 网络的配置方案

根据实际需要及经费情况,通常有下列结构类型:

(1) PLC 做 1 类主站,不设监控站,在调试阶段配置一台编程设备。由 PLC 完成总线通信管理、从站数据读写、从站远程参数设置工作。

(2) PLC 做 1 类主站,监控站通过串口与 PLC 一对一的连接。因为监控站不在 PROFIBUS 网上,不是 2 类主站,不能直接读取从站数据和完成远程组态工作。监控站所需的从站数据只能通过串口从 PLC 中读取。

(3) 用 PLC 或其他控制器做 1 类主站,监控站(2 类主站)连接在 PROIBUS 总线上。可以完成远程编程、组态以及在线监控功能。

(4) 用配备了 PROFIBUS 网卡的 PC(个人计算机)做 1 类主站,监控站与 1 类主站一体化。这是一个低成本方案,但 PC 应选用具有高可靠性、能长时间连续运行的工业级 PC。对于这种结构类型,PC 的故障将导致整个系统瘫痪。另外通信厂商通常只提供模块的驱动程序,总线控制程序、从站控制程序和监控程序可能要由用户开发,因此开发工作量较大。

(5) 工业控制 PC + PROFIBUS 网卡 + SOFTPLC 的结构形式。近来出现一种称为

SOFTPLC 的软件产品，是将通用型 PC 改造成一台由软件（软逻辑）实现的 PLC。这种软件将符合 IEC 61131 标准的 PLC 的编程、应用程序运行功能、操作员监控站的图形监控开发和在线监控功能等集成到一台 PC 上，形成一个 PLC 与监控站一体化的控制器工作站。

8.3 基于组态的 PROFIBUS 通信

对于某些分布很广的系统，例如大型仓库、码头和自来水厂等，可以采用分布式 I/O，将它们放置在离传感器和执行机构较近的地方，分布式 I/O 通过 PROFIBUS-DP 网络与 PLC 通信，可以减少大量的接线。

本节将通过实例来介绍用 STEP 7 建立和组态使用 PROFIBUS-DP 网络的 SIMATIC S7 自动化系统的过程。

8.3.1 PROFIBUS-DP 从站的分类

1. 紧凑型 DP 从站

紧凑型 DP 从站具有不可更改的固定的输入区域和输出区域。ET 200B 电子终端系列（B 代表 I/O 模块）就是这种紧凑型 DP 从站。ET 200B 模块系列提供不同电压范围和不同数量 I/O 通道的模块。

2. 模块式 DP 从站

模块式 DP 从站的输入区域和输出区域是可变的，可以用 S7 组态软件 HW Config 定义它们。ET 200M 是典型的模块化的分布式 I/O，它使用 S7-300 全系列模块，最多可以扩展 8 个模块，连接 256 个 I/O 通道。它需要 1 块 ET 200M 接口模块（IM 153）来实现与主站的通信。

在组态时 STEP 7 自动分配紧凑型 DP 从站和模块式 DP 从站的输入/输出地址，就像访问主站内部的输入/输出模块一样，DP 主站的 CPU 通过这些地址直接访问它们。

3. 智能从站（I 从站）

在 PROFIBUS-DP 网络中，某些型号的 CPU 可以作 DP 从站。在 SIMATIC S7 系统中，这些现场设备称为“智能（Intelligent）DP 从站”，简称为“I 从站”。智能从站的输入区域和输出区域必须用 S7 组态软件 HW Config 来定义。

智能 DP 从站提供给 DP 主站的输入/输出区域不是实际的 I/O 模块使用的 I/O 区域，而是从站 CPU 专门用于通信的输入/输出映像区。

8.3.2 PROFIBUS-DP 网络的组态

在下面的例子中，DP 网络中的主站是带 DP 接口的 CPU 416-2DP，通过 CPU 内集成的 DP 接口，将 DP 从站 ET 200B-16 DI/16DO、ET 200M 和作为智能从站的 CPU 315-2DP 连接起来，传输速率为 1.5 Mbit/s。

1. 生成一个 STEP 7 项目

在桌面上打开 SIMATIC Manager（管理器），建立一个新的项目，选择第一个站的 CPU 为 CPU 416-2DP，项目名称为“DP 主从通信”。

在管理器中选择已经生成的“SMATIC 400 Station”对象（见图 8-7），双击屏幕右边的“Hardware”图标，进入“HW Config”（硬件组态）窗口后，在 CPU 416-2DP 的机架中添加电源模

块、一块 16 点输入模块和一块 16 点输出模块，并设置该站的参数，硬件组态的具体方法见 4.2 节。图 8-7 是组态完成后的 SIMATIT 管理器。

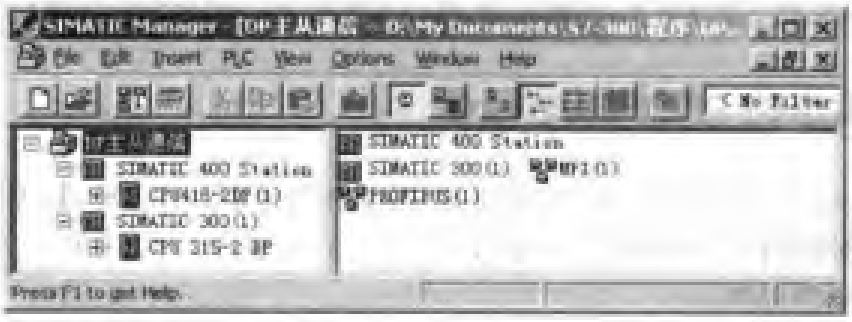


图 8-7　SIMATIC 管理器

2. 设置 PROFIBUS 网络

用鼠标右键点击管理器屏幕左边最上面的“项目”对象(见图 8-7)，在打开的快捷菜单中选择命令“Insert New Object”→“PROFIBUS”，将会生成网络对象 PROFIBUS(1)，在自动打开的网络组态工具 NetPro 中，有红色的 MPI 网络线、紫色的 PROFIBUS 网络线和 CPU 416-2DP 的图标，可以对 MPI 和 PROFIBUS 网络组态。

双击图中的 PROFIBUS 网络线，在出现的对话框中打开“Network Settings”选项卡(见图 8-8)，点击【OK】按钮确认系统推荐的默认的参数，即设置传输速率为 1.5 Mbit/s，总线行规(Profile)为 DP。也可以在对话框中选择 PROFIBUS 子网络的网络参数。

在图 8-8 中，“Highest PROFIBUS Address”(最高站地址)用来优化多主站总线存取控制(令牌管理)，单主站可以使用默认值 126。可以用【Bus Parameters…】按钮访问总线参数，【Option】按钮用于设置一些参数。

传输速率(Transmission rate，9.6 kbit/s～12 Mbit/s)和总线行规(Profile)将用于整个 PROFIBUS 子网络，默认的传输速率为 1.5 Mbit/s。

总线行规为不同的 PROFIBUS 应用提供基准(即默认的设置)，每个总线行规包含一个 PROFIBUS 总线参数集。这些参数由 STEP 7 程序计算和设置，并考虑到特殊的配置、行规和传输速率。这些总线参数适用于整个总线和连接在该 PROFIBUS 子网络中的所有节点。

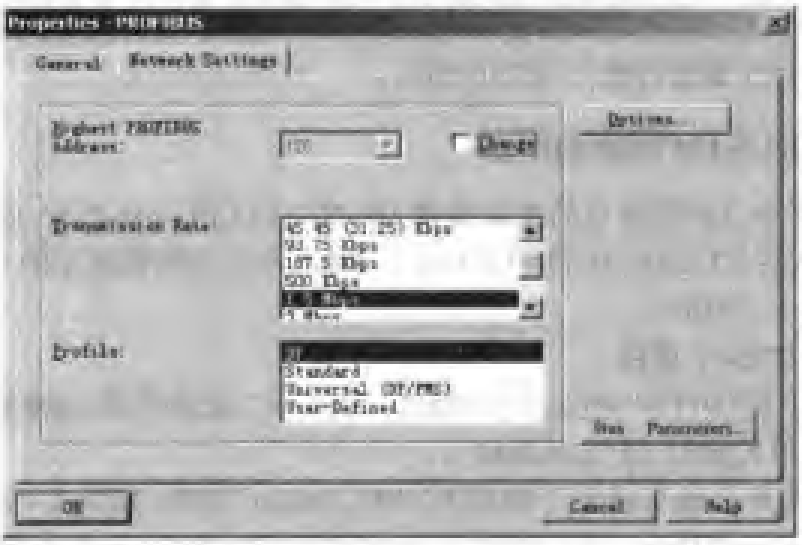


图 8-8　设置 PROFIBUS 网络的参数

对于 PROFIBUS-DP 网络中不同的硬件配置提供不同的总线行规：

(1) DP 行规

纯 PROFIBUS-DP 单主站系统或包含 SIMATIC S7 和 M7 装置的多主站系统选用 DP 行规。这些节点必须是 STEP 7 项目的组成部分，并且已经被组态。

符合欧洲标准 EN 50170 Volume 2/3, Part 8-2 PROFIBUS 的设备可以连接到 PROFIBUS 子网上，这些设备包括 SIMATIC S7、M7、C7 和 S5，以及其他厂家生产的分布式 I/O。

(2) Standard 行规

不能用 STEP 7 组态或不属于当前 STEP 7 项目处理的总线节点可以选用“Standard”(标准)行规。总线参数根据简单的、非优化的算法计算。

(3) Universal(DP/FMS)行规

如果个别的 PROFIBUS 子网节点使用 PROFIBUS FMS 服务，对于 CP 343-5、CP 343-2 (S7 系列)、CP 5431 (S5 系列)和其他厂家生产的 PROFIBUS FMS 设备，可以选择 Universal (通用)行规。

使用 Universal 行规的网络可以使用 SIMATIC S5 系列的部件，例如 CP 5431 通信处理器或 S5-95U 系列 PLC。也可以使用其他 STEP 7 项目中的附加节点。

(4) 用户定义的行规

可以为特殊的应用定义专用的 User-defined(用户定义的)行规。首先选用 DP、Standard 或 Universal(DP/FMS)行规的总线参数设置作为用户定义的行规，然后根据需要修改它们。这种调整和修改只能由具有网络使用经验的工程师来完成。

3. 设置主站的通信属性

退出 NetPro 程序，返回“SIMATIC Manager”的主屏幕(见图 8-7)。选择屏幕左边的 SIMATIC 400 站对象后，双击屏幕右边的 Hardware(硬件)对象，打开 HW Config 工具。图 8-9 给出了已经组态好的 PROFIBUS 网络接线图，此时屏幕左边的窗口中只有生成项目时设置的 S7-400 的机架和机架中的 CPU 416-2DP 模块。

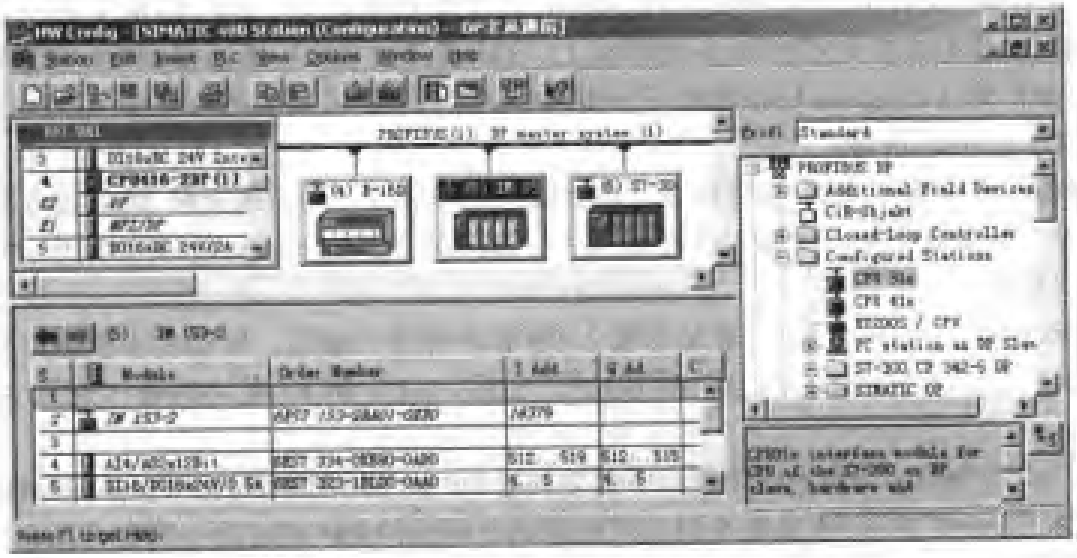


图 8-9 PROFIBUS 网络的组态

双击机架中"DP"所在的行，在打开的对话框的"Operating Mode"选项卡中，选择该站为DP主站(DP Master)。

点击"General"选项卡中的【Properties】按钮，在"Parameters"选项卡(见图8-10)中可以设置站地址，默认的站地址为2。"Subnet"列表框中的"not networked"为不联网，可以选择连接已经建立的PROFIBUS(1)子网络。

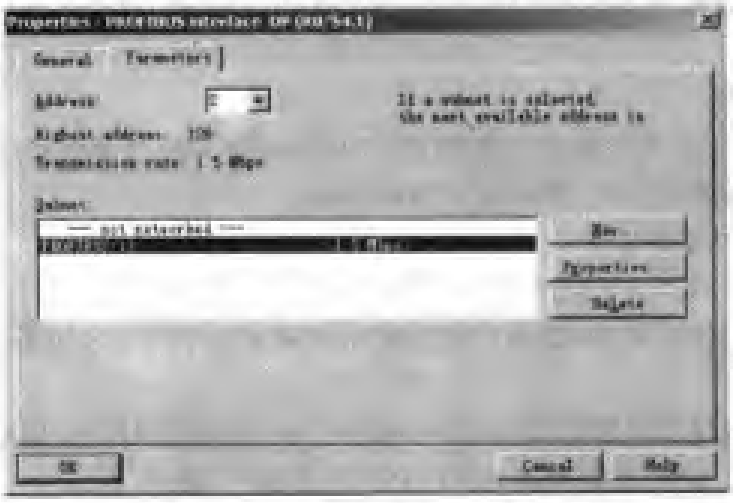


图8-10 PROFIBUS接口的参数设置

在"Parameters"选项卡中点击按钮【New】，可以建立一个新的PROFIBUS子网络。用【Properties】按钮打开PROFIBUS参数设置对话框，可以设置选中的网络的属性。用【Delete】按钮可以删去一个选中的子网络。

点击【OK】按钮返回"HW Config"屏幕，将在"HW Config"屏幕中出现与主站相连的网络线(见图8-9)，但是这时还没有图中的3个从站。

4. 组态DP从站ET 200 B

首先组态第一个从站ET 200B。打开图8-9中屏幕右边硬件目录窗口中最上面的"PROFIBUS-DP"文件夹，在其中的"ET 200 B"文件夹中选择"ET 200 B-16DI/16DO"。用鼠标将它拖到屏幕上方的PROFIBUS网络线上，这样就把该DP从站连接到DP主站系统了。此时将自动打开"Properties-PROFIBUS"对话框，设置该DP从站的站地址为4，点击【OK】按钮返回"HW Config"屏幕。

在网络中选中该从站后，在屏幕左下部的窗口中将显示它的详细资料，例如它占用的输入/输出地址。双击表中某一行输入或输出，在打开的"DP Slave Properties"对话框中，可以更改输入/输出地址。

在PROFIBUS网络系统中，各站的输入/输出自动统一编址。例如在本例中，CPU 416-2DP的16点DI模块的地址为IB0和IB1，ET 200B 16DI/16DO模块的输入地址为IB2和IB3。

双击"HW Config"屏幕上面的窗口中新生成的DP从站的图标，打开"Properties-DP Slave"对话框，在"General"选项卡(见图8-11)，可以看到已组态的DP从站的一些参考信息，例如订货号、设备系列、类型、诊断地址和站地址等。

诊断地址"Diagnostic address"用于组织块OB 86，通过它读出诊断信息，以找到DP从站出

现故障的原因。诊断地址由"HW Config"推荐,用户也可以更改它。

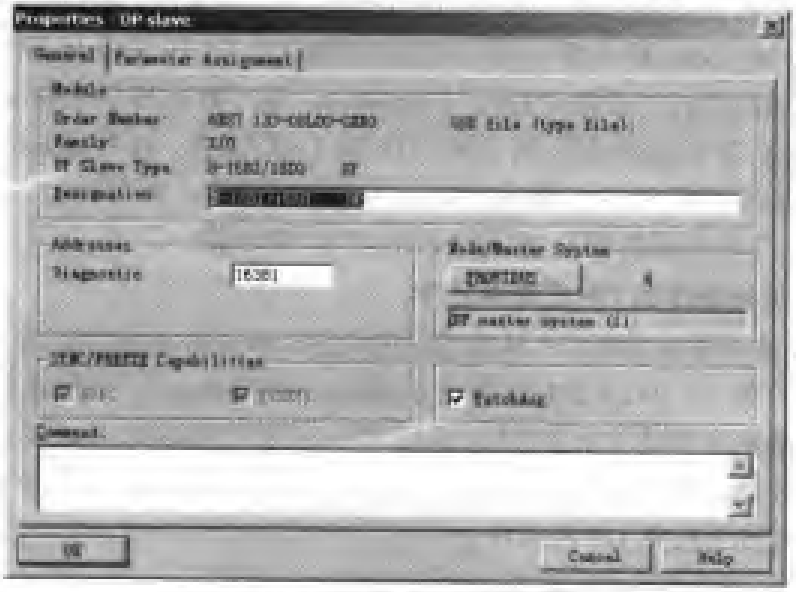


图 8-11　DP 从站的组态

"SYNC/FREEZE Capabilities"指出 DP 从站是否能执行由 DP 主站发出的 SYNC(同步)和 FREEZE(锁定)控制命令。"HW Config"从 DP 从站的 GSD 文件中得到有关的信息,用户不能更改此设置。

如果选择监控定时器(Watchdog)功能,在预定义的响应监视时间内,如果 DP 从站与主站之间没有数据通信,DP 从站将切换到安全状态,所有输出被设置为 0 状态,或输出一个替代值。如果关闭监控定时器,出错时 DP 从站的输出可能不会被置为 0 状态,所以建议只是在调试时才关闭监控定时器。

在"DP Slave Properties"对话框的"Parameters Assigning"选项卡上,可以设置 DP 从站的参数。有关数据的内容和含义,请参看 DP 从站设备的文本。

5. 组态 DP 从站 ET 200 M

ET 200M 是模块化的远程 I/O,与组态 ET 200B 从站相同,在硬件目录中打开"\ PROFIBUS-DP \ ET 200M"文件夹,选择接口模块"IM 153-2"。将它拖到 PROFIBUS 网络线上,就生成了 ET 200M 从站。在出现的"Properties-PROFIBUS Interface IM 153-2"对话框中,设置它的站地址为 5。

在图 8-9 左上部的窗口中选中该从站后,在屏幕左下部的窗口中将显示它的机架结构,其中的 4～11 行最多可以插入 8 块 S7-300 系列的模块。打开硬件目录中的"IM 153-2"文件夹,它里面的各子文件夹列出了可用的 S7-300 模块,其组态方法与普通的 S7-300 的相同。将模拟量模块"SM 334 AI4/AO2"放置到屏幕左下部"ET 200 M"站的槽 4(见图 8-9),数字量模块"SM 323 DI 16/DO 16"放置到槽 5。

6. 组态一个带 DP 接口的智能 DP 从站

下面将建立一个以 CPU 315-2DP 为核心的智能从站。将 S7-300 连接到 DP 主站之前,必须在项目中建立 S7-300 站对象。关闭打开的网络组态工具 NetPro 和 HW Config 硬件组态工具,进入 SIMATIC 管理器,用鼠标右键点击屏幕左边最上面的"DP 主从通信"项目对象,在打

开的快捷菜单中选择命令“Insert New Object”→“SMATIC 300 Station”，插入新的站。双击新站的“HW Config”图标，对该站的硬件组态。首先生成该站的机架，将 CPU 315-2DP 模块插入槽 2，电源模块（PS 307 5A）插入槽 1，数字量模块 SM323 DI 16/DO 16 插入槽 4。

将 CPU 放到机架上时，将会自动打开“Properties - DP”对话框的“Parameter”选项卡。默认的 PROFIBUS 地址为 6，选择连接到 PROFIBUS(1)子网络。在“Operating Mode”选项卡，将该站设置为 DP 从站（DP Slave），最后点击【OK】按钮确认。在“HW Config”中保存对 S7-300 站的组态。图 8-7 是生成智能从站后的 SIMATIC 管理器。

7. 将智能 DP 从站连接到 DP 主站系统中

为了将 S7-300 站组态为智能 DP 从站，返回到组态 S7-400 站硬件的屏幕。在图 8-9 右边的硬件目录窗口中打开“\ PROFIBUS-DP \ Configured Stations”（已经组态的站）文件夹，将“CPU 31x”拖到屏幕左上方的 PROFIBUS 网络线上。“DP Slave Properties”对话框被自动打开，自动分配的站地址为 6。在“Connection”选项卡（见图 8-12）选中“CPU 315-2DP”，点击【Connect】按钮，该站被连接到 DP 网络中。

连接好 4 个站的 PROFIBUS-DP 网络系统如图 8-9 所示。

8.3.3 主站与智能从站主从通信方式的组态

可以将自动化任务划分为用多台 PLC 控制的若干个子任务，这些子任务分别用不同的 CPU 独立和有效地进行处理，这些 CPU 在 DP 网络中作为 DP 主站的智能从站。

DP 主站直接访问“标准”的 DP 从站（例如紧凑型 DP 从站 ET 200B 和模块式 DP 从站 ET 200M）的分布式输入/输出地址区。

DP 主站不是直接访问智能 DP 从站的输入/输出，而是访问 CPU 的输入/输出地址空间的传输区。由智能从站的 CPU 处理该地址区与实际的输入/输出之间的数据交换。组态时指定的用于主站和从站之间交换数据的输入/输出区不能占据 I/O 模块的物理地址区。

主站与从站之间的数据交换是由 PLC 的操作系统周期性自动完成的，不需要用户编程，但是用户必须对主站和智能从站之间的通信连接和数据交换区组态。这种通信方式称为主从（Master/Slave）方式，简称为 MS 方式。

点击图 8-12 中的“Configuration”选项卡，为主-从通信的智能从站配置输入/输出区地址（见图 8-13）。点击图中的【New】按钮，出现如图 8-14 所示的设置 DP 从站输入/输出区地址的对话框。点击图 8-13 中的【Edit】按钮，可以编辑选中的行，点击图 8-13 中的【Delete】按钮，可以删除选中的行。

图 8-13“Configuration”选项卡的表格中各参数的意义如下：

- “Row”：行的编号；
- “Mode”：通信模式，在图 8-14 中可选“MS”（主从）和“DX”（直接数字交换）两种模式；
- “Partner DP Addr.”：DP 通信伙伴的 DP 地址；
- “Partner Addr.”：DP 通信伙伴的输入/输出的地址；
- “Local Addr.”：本站的输入/输出的地址；
- “Length”：连续的输入/输出地址区的长度。
- “Consistency”：数据的连续性。

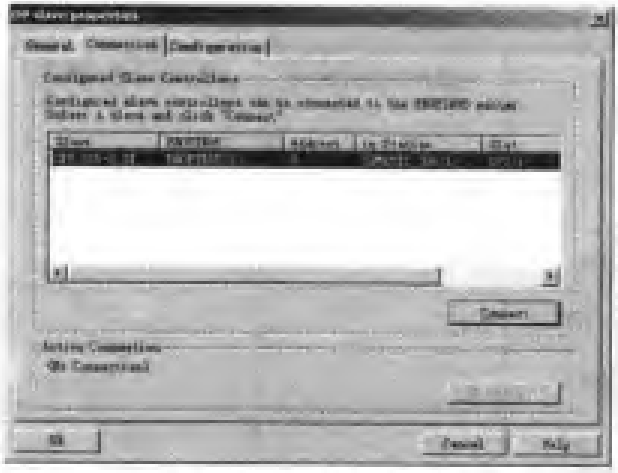


图 8-12　DP 从站的连接

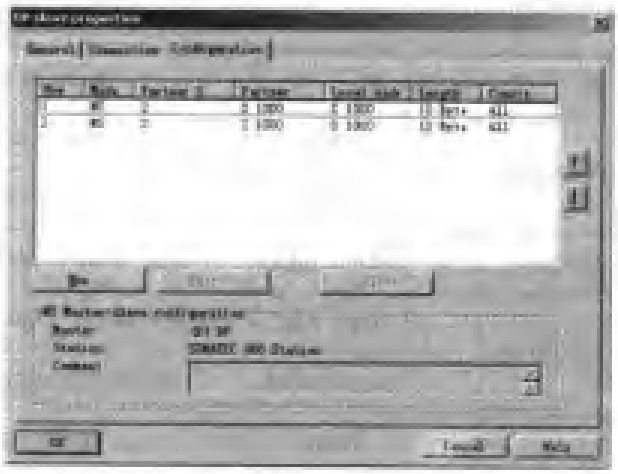


图 8-13　DP 主从通信的组态

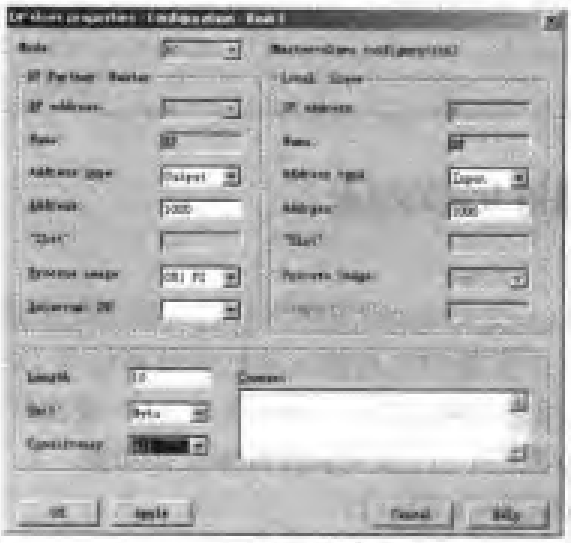


图 8-14　DP 从站属性的组态

设置通信用的输入/输出区时，应确保 DP 主站的一个输出区域分配给 DP 从站的一个输入区域，反之亦然。点击【OK】按钮返回 SIMATIC 400 站的 HW Config 站屏幕。用菜单命令"Station"→"Save and Compile"保存组态的结果。

图 8-15 是组态完成后的 MPI 网络和 DP 网络。表 8-4 是 DP 网络主站和从站中的 I/O 地址。除智能从站以外，主站与其他从站中的站内 I/O 地址是系统按组态的先后顺序统一自动分配的。智能 DP 从站(CPU 315-2DP)内部的输入/输出地址独立于主站和其他从站。

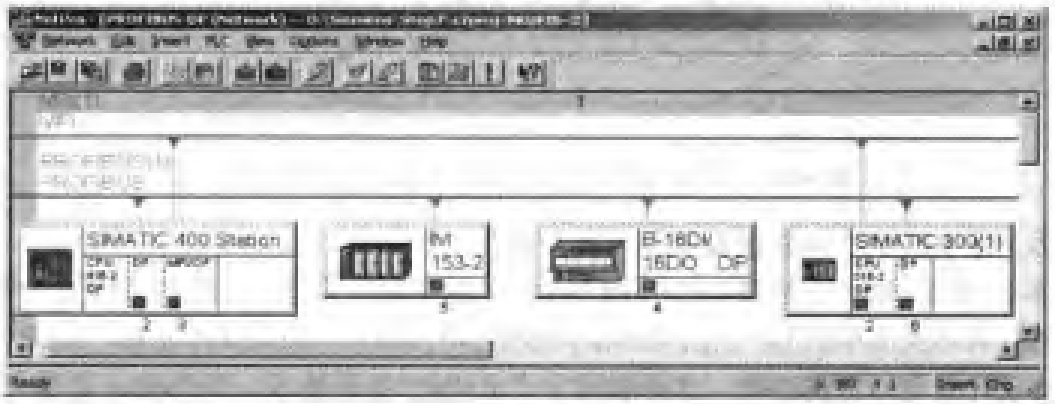


图 8-15　组态后的网络

表 8-4 DP 网络主站和从站中的 I/O 地址

型 号	站 号	模 块	站 内 地 址	主站中的通信地址	从站中的通信地址
CPU416-2DP	2	DI16	IB0 与 IB1		
		DO16	QB0 与 QB1		
ET 200 B	4	16DI / 16DO	IB2 与 IB3 / QB2 与 QB3		
ET 200 M	5	AI4 / AO2	IW512～IW518 / QW512 - QW514		
		DI16 / DO16	IB4 与 IB5 / QB4 与 QB5		
CPU315-2DP	6	DI 16	IB0 与 IB1		
		DO16	QB0 与 QB1		
				O1000～O1009	I1000～I1009
				I1000～I1009	O1000～O1009

8.3.4 直接数据交换通信方式的组态

1. 直接数据交换

直接数据交换(Direct Data Exchange)简称为 DX,又称为交叉通信。在直接数据交换通信的组态中,智能 DP 从站或 DP 主站的本地输入地址区被指定为 DP 通信伙伴的输入地址区。智能 DP 从站或 DP 主站利用它们来接收从 PROFIBUS-DP 通信伙伴发送给它的 DP 主站的输入数据。在选型时应注意某些 CPU 没有直接数据交换功能。

下面是直接数据交换的几种应用场合:

(1) 单主站系统中 DP 从站发送数据到智能从站

如图 8-16 所示,使用这种组态,从 DP 从站来的输入数据可以迅速地传送到 PROFIBUS-DP 子网的智能从站(I 从站)。所有的 DP 从站或其他智能从站原则上都能提供用于 DP 从站之间的直接数据交换的数据,只有智能 DP 从站才能接收这些数据。

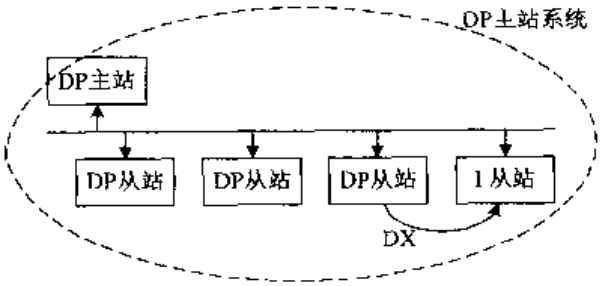


图 8-16 单主站系统中 DP 从站发送数据到智能从站

(2) 多主站系统中从站发送数据到其他主站

如图 8-17 所示,同一个物理 PROFIBUS-DP 子网中有几个 DP 主站的系统称为多主站系统。智能 DP 从站或简单的 DP 从站来的输入数据,可以被同一物理 PROFIBUS-DP 子网中不同 DP 主站系统的主站直接读取。这种通信方式也叫做"共享输入",因为输入数据可以跨 DP

主站系统使用。

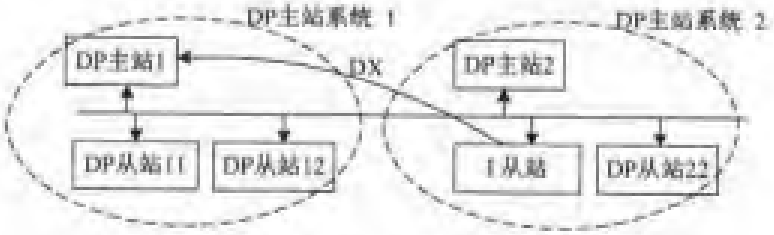


图 8-17　多主站系统中从站发送数据到其他主站

(3) 多主站系统中从站发送数据到智能从站

如图 8-18 所示，在这种组态下，DP 从站来的输入数据可以被同一物理 PROFIBUS-DP 子网的智能从站读取。而这个智能从站可以在同一个主站系统或其他主站系统中。

在这种方式下，来自不同主站系统的 DP 从站的输入数据可以直接传送到智能 DP 从站的输入数据区。

原则上所有 DP 从站都可以提供用于 DP 从站之间进行直接数据交换的输入数据，这些输入数据只能被智能 DP 从站使用。

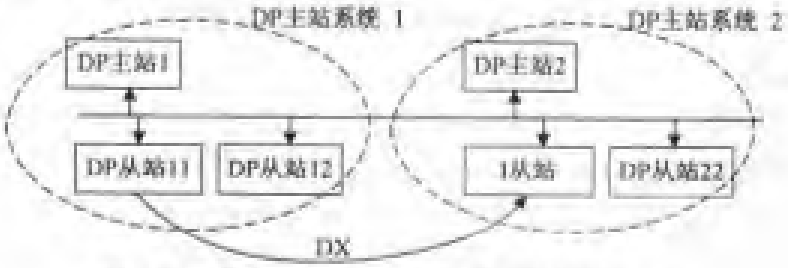


图 8-18　多主站系统中从站发送数据到智能从站

2. 直接数据交换组态举例

DP 主站系统中有 3 个 CPU(见图 8-19)，DP 主站 CPU 417-4 的符号名为“DP 主站 417”，站地址为 2；DP 从站 CPU 315-2 DP 的符号名为“发送从站 315”，站地址为 3；DP 从站 CPU 316-2DP 的符号名为“接收从站 316”，站地址为 4。

通信要求如下：4 号站发送连续的 4 个字到 DP 主站；3 号站发送连续的 8 个字到 DP 主站，4 号站用直接数据交换功能接收这些数据中的第 3 至第 6 个字。

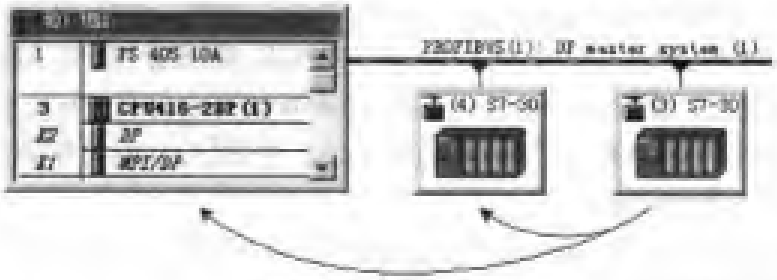


图 8-19　直接数据交换

3. 组态 DP 主站

打开“SIMATIC Manager”(管理器)，建立一个新的项目，选择 CPU 为 CPU 417-4，项目名称为“DP 直接数据交换 DX”(见图 8-20)。

在管理器中选择"SMATIC 400 Station"对象，双击屏幕右边的"Hardware"图标，进入"HW Config"(硬件组态)窗口后，在该站的机架中添加电源模块和 I/O 模块。

双击机架中 CPU 模块内标有 DP 的行，在出现的对话框中的"General"选项卡中点击【Properties】按钮，在"Parameters"选项卡(图 8-10) 中采用默认的站地址 2。点击【New】按钮，在出现的对话框的"Network Settings"选项卡(见图 8-8)中，选择默认的网络参数，传输速率为 1.5 Mbit/s，行规为 DP，点击【OK】按钮确认后回到硬件组态窗口，在 CPU 417-4 的机架右侧出现 PROFIBUS-DP(1)主站系统的网络线(见图 8-19)。此时图中还没有两个从站。

图 8-20　直接数据交换的项目管理器

4. 组态智能从站

在 SIMATIC 管理器中，将主站的站名改为"DP 主站 417"。用鼠标右键点击屏幕左侧最上面的"直接数据交换 DX"项目对象，在打开的快捷菜单中选择命令"Insert New Object"→"SMATIC 300 Station"，插入新的站。选中从站后，双击从站的"HW Config"图标，对该站的硬件组态，生成该站的机架后，将 CPU 315-2DP 插入 2 号槽，电源模块插入 1 号槽。

将 CPU 放到机架上时，"Properties-DP"对话框的"Parameter"选项卡自动打开。将站地址设为 3，选择不连网。回到硬件组态窗口后，双击标有 DP 的行，在打开的对话框中的"Operating Mode"选项卡中将该站设置为 DP 从站(DP Slave)，将从站的站名改为"发送从站 315"，在"HW Config"中保存对 S7-300 站的组态。

用同样的方法生成另一个 DP 从站，CPU 的型号为 CPU 316-2DP，设置该站的站地址为 4，符号名为"接收从站 316"。

5. 将智能从站连接到 DP 网络上

返回 S7-400 主站的硬件组态屏幕，在右边的硬件目录窗口中打开"\ PROFIBUS-DP \ Configured Stations(PROFIBUS-DP 已组态的站)"文件夹，将图标"CPU 31x"拖到屏幕左上方的 PROFIBUS 网络线上。"DP Slave Properties"对话框被自动打开，在"Connection"选项卡中选择列在表中的"SIMATIC 300(1)"站，站地址为 3。点击【Connect】按钮，该站被连接到 DP 网络中。最后点击【OK】按钮确认。

用同样的方法将 CPU 316-2DP 所在的从站连接到 DP 网络中，站地址为 4。

6. 组态发送站的地址区

在主站的硬件组态窗口中，双击 3 号站的图标(见图 8-19)，在出现的对话框的"Configuration"选项卡中，首先为 3 号 DP 从站配置主从通信的输入/输出区地址。点击图中的【New】按钮，出现设置 DP 从站输入/输出区地址的对话框。

按表 8-5 的要求生成"Configuration"中的表格，由表 8-5 可知，DP 主站通过地址 I200 从

CPU 315-2DP 读取数据，同时通过输出区 O180 向 CPU 315-2DP 发送数据。

表 8-5　CPU 315-2DP(3 号从站)的通信区地址组态

行　号	模　式	通信伙伴站地址	通信伙伴地址	本地地址	数据长度	连续性
1	MS	2	I200	O100	8 Word	All
2	MS	2	O180	I80	10 Byte	All

设置好后点击[OK]按钮，返回"Configuration"选项卡，再点击[OK]按钮，最后返回主站的"HW Config"屏幕。

7. 组态接收站的地址区

回到主站的硬件组态窗口后，双击 4 号 DP 从站的图标，进入打开的对话框中的"Configuration"选项卡，按表 8-6 的要求，配置输入/输出区地址。点击图中的[New]按钮，出现设置 DP 从站输入/输出区地址的对话框(见图 8-21)。在最上面的"Mode"选择框内选择"DX"模式，设置表 8-6 中第一行的参数，使 4 号从站通过直接数据交换，接收 CPU 315-2DP 发送到主站的数据中的第 3－第 6 个字。值得注意的是在 DX 通信组态中通信伙伴被自动指定为发送数据的 3 号站，但是通信伙伴的地址必须是主站(2 号站)中接收 3 号站发送的数据的输入区地址。相当于在 3 号从站向主站发送数据时，4 号从站"偷听"其中的部分数据(见图8-19)。

表 8-6　CPU 316-2DP(4 号从站)的通信区地址组态

行　号	模　式	通信伙伴站地址	通信伙伴地址	本地地址	数据长度	连续性
1	DX	3	I 204	I 100	4 Word	All
2	MS	2	I 220	O 140	4 Word	All

按表 8-6 中第二行的要求设置 4 号从站与主站之间的主从(MS)通信的参数。设置好后点击[OK]按钮确认，返回主站的"HW Config"屏幕，用菜单命令"Station→Save and Compile"保存组态的结果。

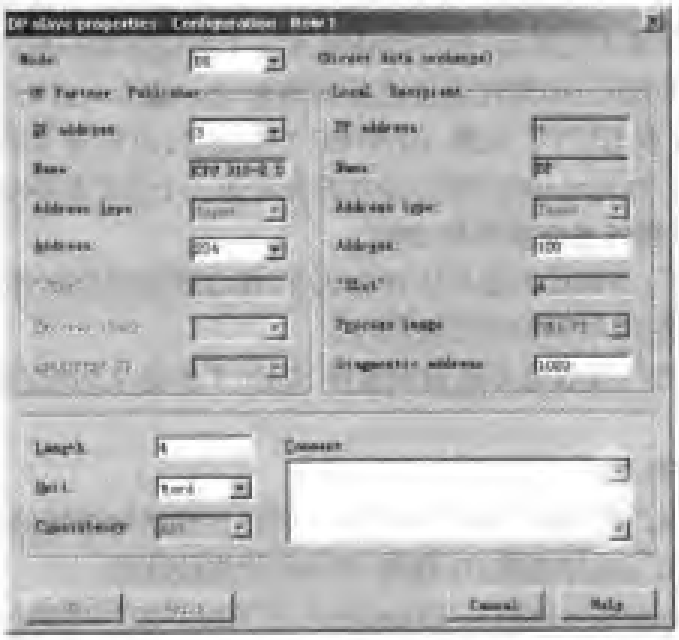


图 8-21　CPU 316-2DP 直接数据交换的参数设置

8.4 系统功能与系统功能块在 PROFIBUS 通信中的应用

8.4.1 用于 PROFIBUS 通信的系统功能与系统功能块

下面简介用于 PROFIBUS 网络通信的系统功能(SFC)和系统功能块(SFB)。

1. 用于数据交换的 SFB/FB(见表 8-7)

表 8-7 用于数据交换的 SFB/FB

编号		助记符	可以传输的字节数		描述
S7-400	S7-300		S7-400	S7-300	
SFB 8	FB 8	U_SEND	440 字节	160 字节	不对等的发送数据给远方通信伙伴,不需要对方的 SFB/FB 应答
SFB 9	FB 9	U_RCV			不对等的接收数据,异步接收通信对象用 U_SEND 发送的数据
SFB12	FB12	B_SEND	64K 字节	32K 字节	发送段数据:要发送的数据区被划分为若干段,各段被单独发送到通信伙伴,通信伙伴接收到最后一段后返回应答
SFB13	FB13	B_RCV			接收段数据:从调用"B_SEND"的通信伙伴接收数据。接收到每一数据段后,发送一个应答,同时参数 LEN(接收到的数据的长度)被刷新
SFB15	FB15	PUT	400 字节	160 字节	写数据到远方 CPU,对方不需要额外的通信功能,接收到后发送执行应答
SFB14	FB14	GET			读取远方 CPU 的数据,对方不需要额外的通信功能
SFB16	-	PRINT			发送数据和指令格式到远方打印机(仅用于 S7-400)

2. S7-400 改变远方设备运行方式的 SFB

(1) SFB 19"START":初始化远方设备的暖起动或冷起动,远方 CPU 应处于 STOP 模式,CPU 的钥匙开关应在 RUN 或 RUN-P 位置。起动完成后,远方设备发送一个肯定的执行应答。

(2) SFB 20"STOP":将远方设备切换到 STOP 状态,远方设备应为 RUN、HOLD 或 STARTUP 状态。操作成功完成后,远方设备发送一个肯定的执行应答。

(3) SFB 21"RESUME":初始化远方设备的热起动。远方 CPU 应处于 STOP 模式,CPU 的钥匙开关应在 RUN 或 RUN-P 位置,用 STEP 7 组态时,应设置为手动起动模式,且没有任何阻止热起动的条件。远方起动完成后,远方设备发送一个肯定的执行应答。

3. 查询远方 CPU 操作系统状态的 SFB

SFB 22"STATUS":查询远方通信伙伴的状态,接收到的通信伙伴的应答用来判断它是否有问题。如果没有错误,接收到的状态被保存。

SFB 23"USTATUS":接收远方通信设备在 STEP 7 中定义的状态发生变化时主动提供的状态信息。

4. 查询连接的 SFC/FC

SFC 62"CONTROL":查询 S7-400 本地通信 SFB 的背景数据块的连接的状态。

FC 62"C_CNTRL":通过连接 ID 查询 S7-300 的连接状态。

可以在路径"\Siemens\Step7\Examples\ZEN01_10"中找到名为"Step7_com_sfb"的实例程序,它给出了使用通信用的 SFB 的编程方法。

5. 分布式 I/O 使用的 SFC

(1) SFC 7"DP _ PRAL":在 DP 主站上触发硬件中断。在智能从站的用户程序中调用 SFC 7,可以触发 DP 主站上的硬件中断,使 DP 主站执行一次 OB40 中用户编写的中断程序。

(2) SFC 11"DPSYC _ FR":同步 DP 从站组。通过 SFC 11,可以同步激活多个从站的输出信号,或锁定多个从站中同一时刻的输入信号。

此功能包括发送下述的一个命令或多个命令的组合到指定的组:

- SYNC:一组 DP 从站同步输出,并且保持这些输出的状态;
- UNSYNC:取消 SYNC 控制命令;
- FREEZE:锁定 DP 从站的输入状态,使主站可以读取同一瞬时各从站的输入状态;
- UNFREEZE:取消 FREEZE 控制命令。

(3) SFC 12 "D _ ACT _ DP":取消或激活 DP 从站。配置了 DP 从站后,即使某些从站已经不存在了或者暂时不需要,但是 CPU 还是会定时地去访问它们。用 SFC 12"D _ ACT _ DP"可以取消这样的 DP 从站,CPU 对它们的访问就会停止,这样可以缩短 DP 总线循环时间,并且不会出现有关的报警信息。也可以用 SFC 12 重新激活被取消的 DP 从站。

(4) SFC 13"DPNRM _ DG":读 DP 从站的诊断数据(从站诊断)。

(5) 用系统功能在用户程序中读、写 DP 从站的模块参数数据。

如果使用装载指令(L 指令)或传送指令(T 指令)访问 I/O 或输入/输出映像区,最多只能读出 4 个连续的字节,即一个双字。用下面的系统功能可以访问 DP 标准从站中的连续数据,最大长度与 CPU 的型号有关。

- SFC 14"DPRD _ DAT":读取 DP 标准从站的连续数据。
- SFC 15"DPWR _ DAT":向 DP 标准从站写连续数据。

6. IEC 61131-5 准则与 S7 用于通信的系统功能块的关系

IEC 61131-5 准则在 S7-300/400 中通过下列块来实现:

- SFB 8"U _ SEND"/ SFB 9"U _ RCV";
- SFB 12"B _ SEND"/ SFB 13"B _ RCV";
- SFB 15"PUT"/ SFB 14"GET";
- IEC 61131-5 准则在 S7-400 中还通过下列块来实现:
- SFB 22"STATUS"/ SFB 23"USTATUS";
- SFB 33"ALARM";
- SFB 36"NOTIFY";
- SFB 19"START",SFB 20"STOP"和 SFB 21"RESCME"用于实现程序控制功能的调用接口。

8.4.2 用 SFC 14 和 SFC 15 传输连续的数据

有的 DP 从站用来实现复杂的控制功能,例如模拟量闭环控制或电气传动等,通常不能用简单的数据结构(例如字节、字和双字)来完成这些任务。这些 DP 从站需要更大的输入区域和输出区域,而且在这些 I/O 区域中的信息常常是连续的。可以用系统功能 SFC 14"DPRD _ DAT"和 SFC 15"DPWR _ DAT"来访问这些模块中连续的输入/输出数据区域。

1. 用 SFC 14"DPRD _ DAT"读取 DP 标准从站的连续数据

用装载(L)指令访问 I/O 或输入映像区时,最多只能读取 4 个连续的字节(双字)。用 SFC 14"DPRD _ DAT"可以读取 DP 标准从站的多个连续数据,最大长度与 CPU 的型号有关。如果数据传输没有错误,读出的数据存放在参数 RECORD 指定的目的数据区。LADDR 是一个地址指针,它指向从站中要读出的输入映像区(I 区)的起始地址。

如果从站是模块式结构,每次只能访问一个模块。

SFC 14"DPRD _ DAT"的参数见表 8-8。

表 8-8 SFC 14"DPRD _ DAT"的参数

参 数	声 明	数据类型	说 明
LADDR	IN	WORD	要读出数据的模块输入映像区的起始地址,必须使用十六进制格式
RECORD	OUT	ANY	存放读取的用户数据的目的数据区,只能使用 BYTE 数据类型
RET _ VAL	OUT	INT	SFC 的返回值,执行时出现错误则返回故障代码

2. 用 SFC 15"DPWR _ DAT"写标准从站的连续数据

用传送(T)指令访问 I/O 或输出映像区时,最多只能写 4 个连续的字节(双字)。用 SFC 15"DPWR _ DAT"可以将 RECORD 指定的连续数据传送到 DP 从站,可以传送的数据长度与 CPU 的型号有关。数据传送是同步的,也就是说 SFC 的执行结束时,写工作也结束。

如果从站是模块式结构或有几个 DP 标识符,每次调用 SFC 14 或 SFC 15 只能访问一个模块或一个 DP 标识符。

SFC 15"DPRW _ DAT"的参数如表 8-9 所示。

表 8-9 SFC 15"DPWR _ DAT"的参数

参 数	声 明	数据类型	说 明
LADDR	IN	WORD	要写入数据的模块输出映像区的起始地址,必须使用十六进制格式
RECORD	OUT	ANY	存放要写出的用户数据的源区域,只能使用 BYTE 数据类型
RET _ VAL	OUT	INT	SFC 的返回值,执行时出现错误则返回故障代码

3. 程序实例

通过下面的实例可以更深入地了解 SFC 14 和 SFC 15 的使用方法。假设项目中有一个 S7 DP 主站(S7-400)和一个智能 DP 从站(CPU 315-2 DP)。智能从站连续的输入数据区和输出数据区分别有 10 个字节长,用系统功能 SFC14 和 SFC15 来传送 I/O 数据。

DP 主站用 SFC 15 发送的输出数据被智能从站用 SFC 14 读出,并作为其输入数据(见图 8-22)保存,反之也适用于从智能从站发送的作为 DP 主站的输入数据的处理。

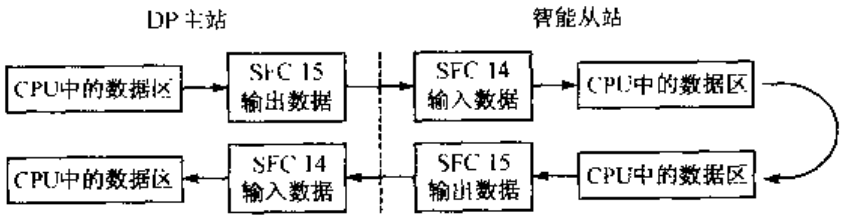


图 8-22 DP 主站与智能从站的通信

应将主站发送的数据存放在从站未被实际的输入/输出模块占用的过程映像区中，例如可以使用 IB100～IB109 和 QB100～QB109。在用户程序中，可以用读写位、字节、字和双字的指令来访问这些数据。

(1) 智能 DP 从站(CPU 315-2DP)OB1 中的用户程序：

```
CALL    SFC 14
  LADDR       :=W#16#3E8              //从站输入区的起始地址(十进制数 1000)
  RET_VAL     :=MW 200                //返回值在存储器字 MW200 中
  RECORD      :=P#I 100.0 BYTE 10     //指向 CPU 存放输入数据的输入映像区的指针
L       IB 100                        //将输入数据 IB100 装入累加器 1
T       QB 100                        //将累加器 1 中的数据传送到 QB100
CALL    SFC 15
  LADDR       :=W#16#3E8              //从站输出区的起始地址(十进制数 1000)
  RECORD      :=P#Q 100.0 BYTE 10     //指向 CPU 存放输出数据的输出映像区的指针
  RET_VAL     :=MW 202                //返回值在存储器字 MW202 中
```

输入程序后保存 OB1，关闭程序编辑器，切换到 SIMATIC 管理器。在子站的“Blocks”文件夹中，应包含块对象“System data”、OB1、SFC14 和 SFC15。

当 DP 主站改变它的运行模式或崩溃时，从站的操作系统将要分别调用 OB 82(诊断中断)和 OB86(机架故障)。如果从站没有对这些 OB 编程，CPU 立即自动地切换到 STOP 模式。因此应在从站上建立相关的出错 OB，以防止 CPU 在此情况下进入 STOP 模式。

同样地，为了避免因为没有诊断和出错 OB 而使 DP 主站的 CPU 进入 STOP 模式，应在 DP 主站生成 OB82 和 OB86。

(2) DP 主站(CPU 416-2DP)OB1 中的用户程序

程序中使用数据块 DB10 和 DB20 作为存放输入数据和输出数据的数据区，生成 OB1 之前，应首先生成 DB10 和 DB20，打开 DB 编辑器，生成长度为 10 个字节的数组(ARRAY)。下面是 OB1 中的用户程序：

```
CALL    SFC14
  LADDR       :=W#16#3E8              //输入区的起始地址(十进制数 1000)
  RET_VAL     :=MW200                 //返回值在存储器字 MW200 中
  RECORD      :=P#DB10.DBX0.0 BYTE 10 //指向存放输入数据的数据区的指针
CALL    SFC15
  LADDR       :=W#16#3E8              //输出区的起始地址(十进制数 1000)
  RECORD      :=P#DB20.DBX0.0 BYTE 10 //指向存放输出数据的数据区的指针
  RET_VAL     :=MW202                 //返回值在存储器字 MW202 中
```

为了便于监视数据通信，在从站的用户程序中，用装载(L)和传送(T)指令将从站 IB100 中接收到的第 1 个数据字节传送到要发送的第 1 个数据字节 QB100。因此通过主站与从站之间的周期性通信和从站内部的数据传送，来自主站输出数据区 DB20.DBB0 的数据立刻被从站返回到主站存放输入数据的数据区中的第 1 个字节 DB10.DBB0。

4. 测试 DP 主站和智能从站之间的数据交换

为了测试主站和从站之间的输入/输出数据的交换，将程序下载到两台 PLC。用 MPI 电

缆连接装有 STEP 7 的计算机和 CPU 416-2DP，用 PROFIBUS 电缆连接两个 CPU 的 DP 接口，令两台 PLC 运行在 RUN-P 模式。如果与 DP 出错有关的 LED("SF DP"或"BUSF"LED)没有点亮或闪烁，说明 DP 主站与智能从站之间的 DP 数据通信在正确地执行。

在 SIMATIC 管理器中，执行菜单命令"View"→"ONLINE"，进入在线状态。打开 SIMATIC 400(1)文件夹，用右键点击主站中的 CPU 416-2DP 图标，打开快捷菜单，用命令"PLC"→"MONITOR/MODIFY VARIABLES"打开变量表，在变量表中监视主站发送的第 1 个输出数据字节 DB20.DBB0 和主站接收到的第 1 个输入数据字节 DB10.DBB0。

在主站的变量表中执行菜单命令"VARIABLE"→"MONITOR"来监视这些变量，给变量 DB20.DBB0 赋值，例如"B # 16 # 55"。可以看到 DB10.DBB0 的监视值立即变为用 DB20.DBB0 设置的值。这是因为从站的用户程序将 DP 主站发送的第一个数据字节立即返回给了主站。

8.4.3 分布式 I/O 触发主站的硬件中断

与本地的中央机架或扩展机架中的 I/O 一样，分布式 I/O 设备也可以产生硬件中断，或称为过程中断。在 PROFIBUS 网络中，硬件中断可以由支持中断处理的 DP 从站或由 DP 从站设备中的某个模块产生。例如当超出测量限定值时，具有硬件中断功能的模拟量输入模块可以触发硬件中断。用户程序被硬件中断中止执行，并调用一个中断组织块，例如 OB40。

1. 用 SFC 7 触发 DP 主站上的过程中断

在智能从站的用户程序中调用 SFC 7"DP _ PRAL"，在它的输入信号 REQ 的脉冲上升沿，触发 DP 主站的硬件中断，使 DP 主站执行一次 OB40 中的程序，执行过程如图 8-23 所示。

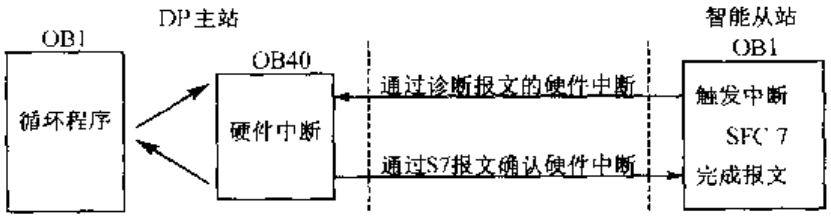


图 8-23 硬件中断的执行过程

SFC 7"DP _ PRAL"的参数如表 8-10 所示。

表 8-10 SFC 7"DP _ PRAL"的参数

参 数	声 明	数据类型	说 明
REQ	IN	BOOL	REQ 为 1 时从站触发主站的硬件中断
IOID	IN	WORD	DP 从站发送存储器地址区的标识符
LADDR	IN	WORD	DP 从站发送存储器地址区的起始地址
AL _ INFO	IN	DWORD	中断标识符，传送给 DP 主站上的 OB40 中的变量 OB40 _ POINT _ ADDR
RET _ VAL	OUT	INT	SFC 的返回值，如果执行过程中出现错误，则返回故障代码
BUSY	OUT	BOOL	BUSY 为 1 表示从站触发的硬件中断还未被 DP 主站确认

AL _ INFO 为中断标识符，用来说明触发硬件中断的原因，它发送给 DP 主站。在 DP 主站的 OB40 中，用变量 POINT _ ADDR 来访问这个标识符。

DP 从站发送存储器地址区的标识符 IOID = ＃16＃54 时，为外设输入(PI)地址区，IOID = B＃16＃55 时，为外设输出(PQ)地址区。对于既有输入又有输出的混合模块，区域标识符为两个地址中较低的那一个。若两个地址相同，则指定为 B＃16＃54。

IOID 和 LADDR 惟一确定了被请求的硬件中断。在发送存储器中每个被组态的地址区，可以在任意的时间准确地触发一个硬件中断。

SFC 7 是异步执行的，需要执行多个 SFC 调用周期。调用 SFC 7 并令 REQ = 1，就可以触发一个硬件中断请求。

SFC 7 的执行状态由输出参数 RET_VAL 和 BUSY 提供。当主站中 OB40 执行结束时，SFC 7 的任务完成。如果 DP 从站是标准从站，只要主站得到诊断帧，则从站触发的硬件中断完成。

如果 SFC 7 还未被 DP 主站确认，则 BUSY = 1，SFC 7 的执行过程中发生错误时，返回的故障代码在输出参数 RET_VAL 中。

2. 从站触发过程中断的程序设计

下面的实例中智能从站为 CPU 315-2DP，主站为 S7-400 PLC。智能从站中起始地址为 1000 的输出模块触发一个硬件中断。为了使中断功能的测试和监视更容易一些，在智能从站上循环地触发硬件中断。

从站发送两条附加信息给 DP 主站：在 SFC 7 的双字输入参数 AL_INFO(中断标识符)的前半部分，传送 SFC 7 的中断 ID“W＃16＃ABCD”。参数 AL_INFO 的后半部分(MW106)是中断次数计数器，每中断一次，该计数器的值加 1。与此同时，中断 ID 被作为硬件中断报文发送给 DP 主站。DP 主站处理中断组织块 OB40 时，通过局域变量 OB40_POINT_ADDR 可以获得中断 ID。

为了触发硬件中断，在从站的 CPU 的 OB1 中写入下面的 STL 语句，保存后下载给 CPU 315-2DP。

```
L       W＃16＃ABCD                    //预设置的中断标识符
T       MW104
CALL“DP_PRAL”
  REQ           :=M100.0
  IOID          :=W＃16＃55             //模块的地址区域标识符，即外设输出(PQ)地址区
  LADDR         :=W＃16＃3E8            //模块的起始地址，即十进制数 1000
  AL_INFO       :=MD104                 //与应用有关的中断 ID
  RET_VAL       :=MW102                 //返回的故障代码
  BUSY          :=M100.1                //主站未确认时从站 BUSY 标志为 1
A       M 100.1                         //如果主站未确认
BEC                                     //结束对 OB1 的执行
=       M 100.0                         //否则触发新的硬件中断
L       MW106
+ 1                                     //中断计数器加 1
T       MW106
```

BEC 为块结束指令，如果主站未确认，即 BUSY 为 1 时，结束对 OB1 的执行，不执行后面

的程序。下一次循环扫描再从第一条指令开始执行;如果主站确认了,即 BUSY 为 0 时,则执行 BEC 指令后面的程序。

3. S7-400 DP 主站处理硬件中断的程序

由智能从站触发并通过 PROFIBUS-DP 网络发送的硬件中断被 DP 主站的 CPU 识别后,主站 CPU 的操作系统调用硬件中断组织块 OB40。OB40 的局域数据包含产生中断的模块的逻辑基准地址和中断源的其他信息。对于更复杂的模块,OB40 的局域数据还包含中断标识符和状态信息。在 OB40 执行结束后,DP 主站的 CPU 自动发送一个确认信号给触发此中断的智能从站,使从站中系统功能 SFC 7 的输出参数 BUSY 的状态从 1 变为 0。

DP 主站 SIMATIC 400(1)的组织块 OB40 中的 STL 语句如下所示:

```
L     #OB40_MDL_ADDR          //保存触发中断的模块的逻辑基准地址
T     MW10
L     #OB40_POINT_ADDR        //保存智能从站发送的中断 ID(即 W#16#ABCD)
T     MD12
```

装载(L)与传送(T)指令将产生中断 I/O 模块的基地址复制到存储器字 MW10,将用户的中断 ID 复制到存储器双字 MD12。可以用 STEP 7 的菜单命令"Monitor/Modify Variables"打开变量表,通过监视 MW10 和 MD12 来观察中断的执行情况。

将 OB40 下载到作为主站的 CPU 416-2DP,两台 PLC 都切换到 RUN 模式。

4. 测试 DP 主站对硬件中断的响应

为了测试 DP 主站对硬件中断的反应,用 MPI 电缆连接编程装置和 CPU 416-2DP,用 PROFIBUS 电缆连接两个 CPU 的 DP 接口,令两台 PLC 运行在 RUN-P 模式。在 SIMATIC Manager 中用菜单命令"VIEW"→"ONLINE"切换到在线查看。

在 SIMATIC 400(1)文件夹中,打开"Blocks"文件夹。双击后在线查看 OB40,用 STEP 7 观察它的执行情况。通过菜单命令"DEBUG"→"MONITOR"起动 OB40 的程序状态功能,观察 DP 主站对中断的处理。

8.4.4 一组从站的输出同步与输入锁定

系统功能 SFC 11"DPSYC_FR"用于将控制命令 SYNC(同步输出)、UNSYNC(解除同步)、FREEZE(锁定或冻结输入)和 UNFREEZE(取消锁定)发送给一个或多个 DP 从站。这些命令用来实现一组 DP 从站的同步输出或锁定它们的输入。

DP 主站使用全局控制报文(广播报文)同时发送控制命令 SYNC 和/或 FREEZE 给一组 DP 从站。

在用 SFC 11 发送上述控制命令之前,应使用 STEP 7 的硬件组态工具将有关的 DP 从站组合到 SYNC/FREEZE DP 组中,一个主站系统最多可以建立 8 个组。

1. 同步输出与解除同步

通常情况下,DP 主站周期性地将输出数据发送到 DP 从站的输出模块上。使用 SYNC 控制命令,可以将一组选择的 DP 从站切换到同步方式。DP 主站发送当前的输出数据,并命令 DP 从站锁定它们的输出。被选择的 DP 从站组将主站的输出数据存放在它们的内部缓冲区,将它们送到输出模块,并保持输出状态不变。这样可以同步激活一组 DP 从站上的输出数据。每执

行一次 SYNC 控制命令，该组从站将新的输出数据发送到输出模块上。

只有用 SFC 11 发送控制命令 UNSYNC，才可以取消所寻址的 DP 从站的 SYNC 模式。此后 DP 从站返回正常的循环数据传送状态，即 DP 主站发送的数据立即被传送到从站的输出。

2. 输入信号的锁定与解除锁定

通常情况下，DP 主站按照 PROFIBUS-DP 的总线周期，周期性地读取 DP 从站的输入数据，供 CPU 使用。

如果需要得到一组 DP 从站上同一时刻的输入数据，可以通过 SFC 11 将 FREEZE 控制命令发送到该组 DP 从站来实现。

当 FREEZE 命令被发送到一组 DP 从站时，组内所有的 DP 从站切换到 FREEZE 模式，即它们的输入模块上的信号被锁定，并将它们传送到 CPU 的输入过程映像区，以便 DP 主站来读取这些信号。接收到下一个 FREEZE 命令时 DP 从站更新和重新锁定它们的输入数据。

用 SFC 11 发送 UNFREEZE 命令，可以取消所寻址的 DP 从站的 FREEZE 模式，使它们恢复与 DP 主站之间的正常的循环数据传送。此后输入数据立即由 DP 从站更新，并被 DP 主站读取，DP 主站又能接收到周期性刷新的 DP 从站的输入信号。

在重新起动和热起动后，DP 从站不进入 SYNC 或 FREEZE 模式，只有当它们接收到由 DP 主站发出的第一个 SYNC 或 FREEZE 命令之后才进入 SYNC 或 FREEZE 模式。

SFC"11 DPSYC _ FR"的参数见表 8-11。

表 8-11 SFC 11"DPSYC _ FR"的参数

参 数	声 明	数据类型	说 明
REQ	IN	BOOL	电平触发的控制参数，REQ=1 时触发或解除 SYNC 或 FREEZE 操作
LADDR	IN	BYTE	DP 主站的逻辑地址
GROUP	IN	BYTE	用于选择组，第 0 位～第 7 位为 1，分别表示选择第 1 组～第 8 组
MODE	IN	BYTE	SYNC/FREEZE 操作的标识符，见表 8-12
RET _ VAL	OUTPUT	INT	SFC 的返回值，如果执行过程中出现故障，则返回故障代码
BUSY	OUTPUT	BOOL	BUSY=1 表示 SYNC/FREEZE 操作未完成

SFC 11 用输入参数 MODE 指定的控制命令可能的组合见表 8-12。

表 8-12 SFC 11 的控制命令可能的组合

位 号	7	6	5	4	3	2	1	0	取 值
MODE				UNSYNC					B#16#10
				UNSYNC		UNFREEZE			B#16#14
				UNSYNC	FREEZE				B#16#18
			SYNC						B#16#20
			SYNC			UNFREEZE			B#16#24
			SYNC		FREEZE				B#16#28
						UNFREEZE			B#16#04
					FREEZE				B#16#08

输入参数 GROUP 用来指定哪一组将被 SFC11 寻址。第 0～7 位为 1，分别表示第 1～8 组。例如要寻址 4 组和 5 组，只有 GROUP 的第 3 位和第 4 位为 1，因此 SFC 11 的输入信号 GROUP＝B＃16＃18。

在 LADDR 中声明 DP 主站的逻辑基准地址。如果已触发的系统功能还未结束执行，则 BUSY＝1，在块执行过程中发生错误时，返回的故障代码在 RET _ VAL 参数中输出。

SFC11 是异步执行的，需要执行多个 SFC 调用周期。通过 REQ ＝ 1 调用 SFC 7 来执行同步和锁定操作。

若使用了 SFC 15"DPWR _ DAT"（写 DP 数据），在发送 SYNC 给有关的输出之前，SFC 15 必须执行完毕。若使用了 SFC 14"DPRD _ DAT"（读 DP 数据），在发送 FREEZE 给有关的输入之前，SFC 14 必须执行完毕。

在用户程序启动时，若需要在 SYNC 模式下对一组或多组 DP 从站进行输出操作，必须在启动组织块 OB100 中调用带有 SYNC 控制命令的 SFC 11。

在用户程序启动时，若需要在 FREEZE 模式下对一组或多组 DP 从站进行输入操作，必须在启动组织块 OB100 中调用带有 FREEZE 控制命令的 SFC 11。

在同一时间只能初始化一条 SYNC/UNSYNC 命令或一条 FREEZE/UNFREEZE 命令。

3. DP 主站 IM467 使用 SYNC/FREEZE 命令的实例

打开 SIMATIC 管理器，生成一个名为"同步与锁定"的项目，在项目中生成一个名为 SIMATIC 400(1)的新站。

双击管理器左边窗口中的"SIMATIC 400(1)"文件夹，打开新建的站，双击管理器右边工作区中的"Hardware"对象，对新站进行硬件配置。从硬件目录中选择机架"UR2"，将电源模块"PS 407 10A"放在 1 号槽中。应选择支持 SYNC 和 FREEZE 功能的 CPU，例如选订货号为"6ES7 416-1 XJ02-0AB0"的 CPU416-1，将把它放在 3 号槽中。

为了组态插入式 DP 主站模块（IM467），在硬件目录中打开文件夹"＼SIMATIC 400＼IM-400"，选择订货号为"6ES7 467-5GJ01-OAB0"的 IM467 模块，并将它放在 4 号槽中。

IM467 放入机架时，对话框"Properties IM467"和"General"选项卡自动地出现在屏幕上。点击按钮【Properties】，进入"Properties-PROFIBUS interface IM467"对话框，点击【New】按钮，并用【OK】按钮确认默认值，就建立了一个新的 PROFIBUS(1)子网络，该子网络具有 1.5 Mbit/s 传输速率和 DP 类型的总线参数行规。确认 IM467 的默认站地址"2"，用【OK】按钮关闭此对话框。在"Properties IM467"对话框的"Addresses"选项卡中设置模块的地址为 512（即 W＃16＃200）。

点击【OK】按钮返回硬件组态窗口，此时 IM467 模块已插入在槽 4 中，并且用一根水平的直线表示 IM467 的 DP 主站系统（见图 8-24）。

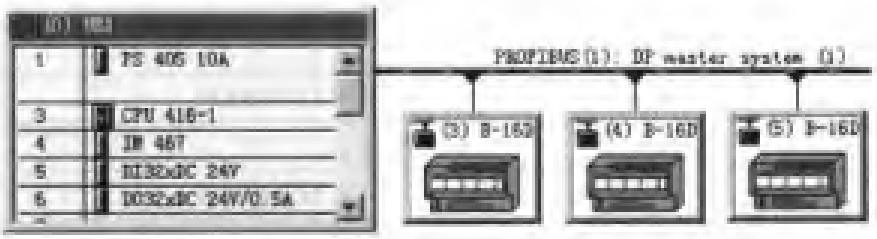


图 8-24　网络组态

下一步组态支持 SYNC 和 FREEZE 控制命令的 ET 200B 从站。在硬件目录中打开 PROFIBUS-DP 模块的文件夹，在子目录 ET 200B 中选择模块“B-16DI/16DO DP”。将该模块拖到图 8-24 中 IM467 的 DP 主站系统网络线上。“Properties-PROFIBUS interface B-16DI/16DO DP”对话框被自动打开，将 PROFIBUS 站地址设置为 3，点击【OK】按钮退出屏幕。

用同样的方法将另一个 B-16 DI/DO DP 组态到 DP 主站系统中，默认的从站地址为 4。将 B-16 DI DP 组态到 DP 主站系统中，默认的从站地址为 5。

下一步设置 SYNC/FREEZE 功能，为此双击图 8-24 中的“PROFIBUS(1)：DP master system (1)”网络线，出现“Properties-DP master system”对话框。首先指定组的特性，为此打开“Group Properties”选项卡（见图 8-25），用“Properties”下面的小方框选择要指定给各组的特性。图 8-25 中定义组 1 为 FREEZE 组，组 2 为 SYNC 组。在“Comment”列可以为各组附加注释或标志。

在“Group assignment”选项卡（见图 8-26），将 DP 从站分配到各组。列表框中的每一行对应一个 DP 从站，最左边是从站的地址和型号，例如“(3)B-16DI/16DO”。列表框的上面给出了每一组的属性，例如第一组下面的“--”表示它不是“SYNC（同步）”组，“X”表示它是“FREEZE(锁定)”组。

选中显示框中第一行(3 号从站 B-16DI/16DO)，用鼠标在列表框下面的 1 和 2 前面的小方框中打勾，第一行中与第一组和第二组交叉的位置出现两个“X”，表示 3 号从站分别属于第 1 组和第 2 组。

用同样的方法，使 4 号从站和 5 号从站分别属于第 2 组和第 1 组。从图 8-26 可以看出，3 号从站和 5 号从站属于锁定组（第 1 组），3 号从站和 4 从号站属于同步组（第 2 组）。设置好后点击【OK】按钮退出对话框，用菜单命令“STATION”→“SAVE AND COMPILE”保存组态的结果。将 SIMATIC 400 站切换到 STOP 模式，并将硬件组态下载到 S7-400 CPU。SIMATIC 400 站的实际硬件结构必须与在“HW Config”中的组态相匹配。

用 PROFIBUS 电缆连接 IM467 模块和 3 个 ET 200B 模块，并将 CPU 416-1 的运行模式切换到 RUN-P，使 CPU 进入 RUN 模式，所有红色的出错 LED 必须是熄灭的。下载完成后关闭“HW Config”程序。

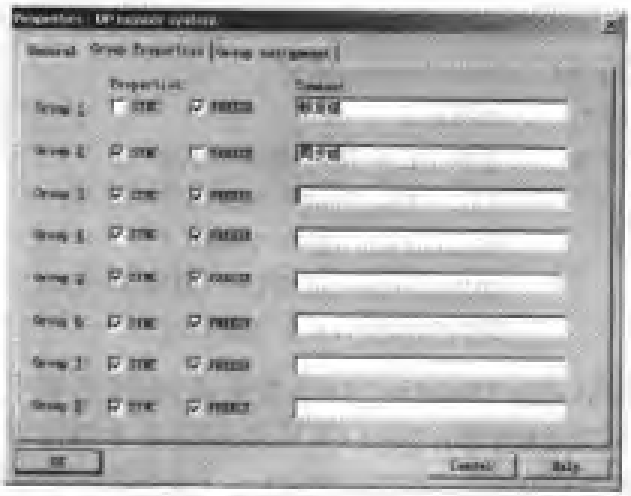


图 8-25　设置 SYNC/FREEZE 组的属性

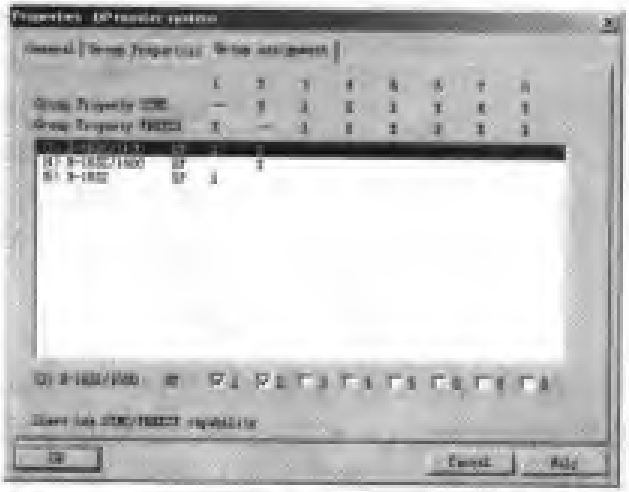


图 8-26　设置 SYNC/FREEZE 组

4. 测试 SYNC/FREEZE 功能的用户程序

双击管理器中“CPU 416-1”的“OB1”图标，将它打开，在“LAD/STL/FBD”程序编辑器中用“View”菜单选择“STL”(语句表)语言，然后输入下面的程序，在 I0.0 的上升沿调用 FC11 “DPSYC_FR”，发送 FREEZE 命令，在 I0.1 的上升沿发送 SYNC 命令。

在调用 SFC 11 时，打开屏幕右边指令树中的文件夹“\ Libraries \ Standard Library \ System Function Blocks”，将 SFC11“DPSYC_FR”“拖放”到程序中。程序编好后，保存并下载到 CPU 416-1。

```
Network 1:检测 I0.0 的上升沿
          A       I0.0
          FP      M10.1                    //在 I0.0 的脉冲上升沿
          =       M10.2                    //M10.2 在一个循环周期为 1 状态,起动 SFC11
Network 2:发送 FREEZE 命令
          G01:    CALL    SFC11            //调用 SFC11
                  REQ     :=M10.2          //触发信号为 M10.2
                  LADDER  :=W#16#200       //DP 主站接口模块 IM467 的输入地址(十
                                           //进制数 512)
                  GROUP   :=B#16#1         //选择第 1 组
                  MODE    :=B#16#8         //选择 FREEZE 模式(见表 8-12)
                  RET_VAL :=MW12           //返回值 RET_VAL 存放在 MW12 中
                  BUSY    :=M10.3          //输出位 BUSY 保存在 M10.3 中
          A       M10.3                    //如果没有执行完 SFC11 (M10.3 = 1)
          JC      G01                      //跳转到标号 G01 处继续执行
Network 3:检测 I0.1 的上升沿
          A       I0.1
          FP      M10.5                    //在 I0.1 的脉冲上升沿
          =       M10.6                    //M10.6 在一个循环周期为 1,启动 SFC 11
Network 4:发送 SYNC 命令
          G02:    CALL    SFC 11           //调用 SFC 11
                  REQ     :=M10.6          //在 I0.1 的脉冲上升沿触发同步操作
                  LADDER  :=W#16#200       //IM467 的输入地址
                  GROUP   :=B#16#2         //选择组 2
                  MODE    :=B#16#20        //选择 SYNC 模式(见表 8-12)
                  RET_VAL :=MW14           //RET_VAL 存放在 MW14 中
                  BUSY    :=M10.7          //输出位 BUSY 保存在 M10.7
          A       M10.7                    //如果没有执行完 SFC11 (M10.7=1)
          JC      G02                      //跳转到标号 G02 处继续执行
```

在 RUN 模式，用菜单命令“PLC”→“Monitor/Modify Variables”打开变量表，在变量表中监视 QB4、IB4、I0.0 和 I0.1 等。QB4 是 3 号站“ET 200B-16DI/16DO”模块的第 1 个输出字节，IB4 是 3 号站的第 1 个输入字节。I0.0 用来触发 FREEZE 组的操作，I0.1 用来触发 SYNC 组的操作。

起动 DP 总线系统后，主站与各 DP 从站循环地传送数据。将 I0.0 置为 1 状态，SFC 11 发

送 FREEZE 控制命令，使 3 号站和 5 号站的输入处于 FREEZE 模式。改变 3 号站实际的输入信号的状态，因为处于锁定模式，这些变化不会传送给主站的 CPU，在主站的变量表中也不能观察到这些变化。

将 I0.1 置为 1 状态时，SFC 11 发送 SYNC 命令，使 3 号站和 4 号站的输出处于 SYNC 模式。在变量表中修改 QB4 的值后，不能传送到 3 号站 ET 200B-16DI/16DO 的输出模块。

在 I0.0 的下一次上升沿，将重新发送 FREEZE 命令，读取 3 号站和 5 号站当前的输入数据。在 I0.1 的下一次上升沿，将重新发送 SYNC 命令，把设置好的数据传送到 3 号站和 4 号站的输出。

8.4.5 用系统功能诊断 DP 从站

1. 用 SFC 13 读 DP 从站的标准诊断数据

可以用 SFC 13“DPNRM_DG”来读取 DP 从站的诊断数据，诊断数据的基本结构见表 8-13。

表 8-13 DP 从站的诊断数据结构

字节	站状态 1
1	站状态 2
3	DP 主站的 PROFIBUS 地址
4	制造商标识符的高字节
5	制造商标识符的低字节
6	附加的与从站有关的诊断数据

表 8-14 是 SFC 13 的输入参数和输出参数。系统功能被调用后，在控制参数 REQ 的上升沿触发读请求。

LADDR 用来设置要读取诊断数据的 DP 从站的诊断地址。读取的数据存放在 RECORD 指定的目的区域。用 RET_VAL 参数报告 SFC 13 的出错代码。SFC 13 是异步执行的，需要多次调用它。

标准从站的标准诊断数据在 241 和 244 B 之间时，如果 RECORD 指定的长度小于 240 B，数据被丢弃，并在 RET_VAL 中返回故障代码。

表 8-14 SFC 13“DPNRM_DG”的参数

参数	声明	数据类型	说明
REQ	IN	BOOL	为 1 时请求读取 DP 从站的诊断数据
LADDR	IN	WORD	DP 从站的诊断地址，必须以十六进制格式输入
RET_VAL	OUT	INT	返回值，执行时出现错误返回故障代码，否则存放实际传送的数据字节数
RECORD	OUT	ANY	存放读取的诊断数据的目的区域，只允许 BYTE 数据类型，6～240 字节。
BUSY	OUT	BOOL	BUSY=1 表示读取过程未完成

如果 RECORD 指定的长度大于等于 240 B，则前 240 B 的标准诊断数据被传送到目的区域，并且数据中的溢出位被置位。

2. 系统状态表 SSL

系统状态表(System Status List，SSL)用来描述 PLC 的当前状态。

SSL 只能用 SFC 51“RDSYSST”读取，但是不能改写它。与 DP 有关的局部系统状态表(局部 SSL)是虚拟表，即仅仅在有请求时，才由 PLC 的操作系统生成它们。

SSL 包含以下的信息：

(1) 系统数据

系统数据包括 CPU 的固定的和可以调整的属性数据，用来描述 CPU 的硬件配置、优先权

等级和通信的状态等。

(2) CPU 内的诊断状态数据

诊断状态数据描述系统诊断功能监视的所有部件的当前状态。

(3) 模块的诊断数据

除了 CPU 之外,其他有诊断功能的模块生成和存储的模块诊断信息和诊断数据。

(4) 诊断缓冲区

所有的诊断事件按它们出现的先后次序登录在诊断缓冲区中。

3. 局部系统状态表的结构

局部系统状态表(局部 SSL)由标题(Header)和数据记录组成。

标题由局部 SSL 的标识符 SSL _ ID、索引(INDEX)、包含在局部 SSL 中的数据记录的字节长度和数据记录的个数组成。

SSL 的标识符 SSL _ ID 占一个字,它由局部 SSL 的编号(低字节)、局部 SSL 的摘录号(第 8~11 位)和模块类别(第 12~15 位)组成。SSL 的每个局部 SSL 有它自己的标识符(SSL _ ID)。为了读取一个完整的局部 SSL 或它的摘录,必须指定相关的 ID。

标识符 SSL _ ID 中的模块类别占用的 4 个位指定要读取的局部 SSL 或它的摘录的模块类型。CPU、IM、FM 和 CP 的模块类别分别为二进制数 0000、0100、1000 和 1100。

某些局部 SSL 或它的摘录需要一个对象类型标识符或一个对象号,此时必须使用索引 INDEX。

SFC 51 的参数 SSL _ ID 和 INDEX 决定将要读哪一个局部 SSL 或哪一个局部 SSL 的摘录。

局部 SSL 可能的摘录是预先定义的,它们用标识号来标识,不能更改。利用局部 SSL 的摘录号,可以指明要读取局部 SSL 的哪一部分。

4. 用 SFC 51"RDSYSST"读局部系统状态表

用系统功能 SFC 51"RDSYSST"(Read System Status)可以读取一个局部 SSL 或其摘录的内容。SFC 51"RDSYSST"的参数见表 8-15。

表 8-15 SFC 51"RDSYSST"的参数

参 数	声 明	数据类型	说 明
REQ	IN	BOOL	起动信号,为 1 时请求读
SSL _ ID	IN	WORD	要读取的 SSL 或局部 SSL 的标识符(ID)
INDEX	IN	WORD	局部 SSL 中一个对象的类型或编号
RET _ VAL	OUT	INT	SFC 的返回值,执行时出错则返回故障代码
BUSY	OUT	BOOL	BUSY 为 1 表示读过程还未结束
SSL _ HEADER	OUT	STRUCT	SSL 的标题,见表后的说明
DR	OUT	ANY	存放读取的 SSL 或局部 SSL 的目标区域,如果只读取了 SSL 中的标题信息,不用查看数据记录,只需查看 SSL _ HEADER。否则 LENTHDR 与 N _ DR 的乘积等于写入数据记录的字节数

表中的参数 SSL _ HEADER 是一种结构,其定义如下:

```
SSL_HEADER:STRUCT
  LENTHDR:      WORD
  N_DR:         WORD
ENT_STRUCT
```

读操作之后，LENTHDR 是读取的 SSL 或局部 SSL 的数据记录的长度。如果仅仅读出 SSL 中的标题信息，N_DR 包含读取的数据记录的编号；否则包含传送到目标区的数据记录的编号。

表 8-16 给出了可使用的局部 SSL 和它们的 SSL_ID 号。《S7-300/400 的系统软件和标准功能参考手册》的第 30 章给出了详细的信息。

表 8-16 局部系统状态表简介

SSL_ID	子 表
W#16#xy 11	模块标识符
W#16#xy 12	CPU 特性
W#16#xy 13	用户存储区
W#16#xy 14	系统存储区
W#16#xy 15	块的类型
W#16#xy 19	模块 LED 的状态
W#16#xy 22	中断状态
W#16#xy 25	过程映像区与 OB 块之间的参数设置
W#16#xy 32	通信状态数据
W#16#xy 71	冗余 CPU 组信息
W#16#xy 74	模块 LED 的状态
W#16#xy 75	在冗余系统中切换 DP 从站
W#16#xy 91	模块状态信息
W#16#xy 92	机架/站状态信息
W#16#xy 95	扩展 DP 主站系统信息
W#16#xyA0	CPU 的诊断缓冲区
W#16#00B1	模块诊断信息(数据记录 0)
W#16#00B2	模块诊断信息(数据记录 1)，物理地址
W#16#00B3	模块诊断信息(数据记录 1)，逻辑地址
W#16#00B4	DP 从站的诊断数据

8.4.6 用系统功能传送数据记录与参数

S7-300/400 允许用户程序在运行期间将数据记录传送给 SIMATIC S7 模块。这种在线数据记录的传送可以用于集中式的模块和分布式 I/O 子站中的模块。可以传送的数据记录分为动态数据记录和静态数据记录。动态数据记录通常由用户程序提供，静态数据记录通过 HW Config 程序生成，并且能长期保存在 CPU 的系统数据块中。SIMATIC S7 用系统功能(SFC)来将数据记录传送给 S7 模块。

1. 传送数据记录的系统功能 SFC 的公用参数

- REQ：系统功能在用户程序中被调用后，在控制输入 REQ 的脉冲上升沿传送数据记录。
- IOID：模块地址区的标识符，B#16#54 和 B#16#55 分别表示外设输入和外设输出。如果模块既有输入又有输出，IOID 为两个地址中较低的那个。若二者的地址相同，则

指定为 B#16#54。

- RECNUM:模块接收参数的数据记录号。
- BUSY:如果系统功能执行还未结束,则 BUSY 为 1。
- RET _ VAL:如果在执行过程中发生错误,用输出参数 RET _ VAL 返回故障代码。

2. 用 SFC 55 写动态参数

系统功能 SFC 55"WR _ PARM"将包含动态参数的数据记录 RECORD 传送给用 LADDER 和 IOID 指定地址的模块。传送给模块的参数不会被 STEP 7 组态的参数覆盖。

3. 用 SFC 56 写默认的参数

系统功能 SFC 56"WR _ DPARM"把数据记录传送给用 LADDR 和 IOID 指定地址的模块。数据记录是在 STEP 7 组态时生成的,可以是动态的,也可以是静态的,其编号为 RECNUM。

4. 用 SFC 57 给模块指定参数

SFC57"PARM _ MOD"将在 STEP 7 中组态的模块的数据记录全部传送给由 LADDR 和 IOID 指定地址的模块。数据记录可以是动态的,也可以是静态的。

"实际"的错误信息(错误代码为 W#16#8XYZ)可以分成两组:

(1) 暂时性错误(错误代码 W#16#80A2~80A4 和 80Cx)

这些错误可能在不采取任何措施的情况下消失,可以用再次调用 SFC 来纠正。例如错误信息 W#16#80C3 是暂时错误,它指出需要的存储器空间被其他功能占用。

(2) 永久性错误(错误代码 W#16#809x,80A1,80Bx,和 80Dx)

不采取措施永久性错误不会自动消除,只有报告的错误被纠正之后才可以再次调用 SFC。

5. 用 SFC 58"WR _ REC"写数据记录

SFC 58 将 RECORD 中的数据记录传送给由 LADDR 和 IOID 指定地址的模块。与 SFC 55 比较,SFC 58 仅能用于数据记录号为 2~240 的数据记录。

6. 用 SFC 59"RD _ REC"读数据记录

SFC 59 从用 IOID 和 LADDR 指定地址的模块中读取编号为 RECNUM 的数据记录,并把它存放在由参数 RECORD 指定的目的区。

8.4.7 向模块传送数据记录与参数的例子

下面的实例描述怎样用 SFC 55 和 SFC 56 将模块的数据记录或参数写入 S7 模块。

SFC 55 用来传送内容可以按需要定义的动态数据记录。SFC 56 用来传送用"HW Config"生成的"静态"数据记录,并存放在 CPU 的系统数据块(SDB)中。在系统起动时,这种数据记录被自动地传送到相应的模块。

【例 8-1】 用 SFC 55 更改已组态的 ET 200M 从站上的模拟量输入模块的测量范围。将测量范围从 ±10 V 改为 ±2.5 V,当被测变量的绝对值小于 2.5 V 时,改变模块的量程可以提高测量的精度。

本例中 ET 200M 使用的模拟输入模块是 S7-300 系列的 SM331 AI2×12 Bit。它有两个分辨率为 12~14 位的模拟量输入通道,组成了通道组 0,没有通道组 1~3。

在《S7-300 和 M7-300 可编程序控制器模板规范参考手册》的附录 A.4 中,可以查找到 S7-300 PLC 的模拟输入模块的数据记录 1(DR1)的格式,其长度为 14 字节,其结构如下:

字节 0 的第 2,6,7 位为 1 时分别表示启用循环结束中断、诊断中断和极限值中断。

字节 1 用来设置 A/D 转换的积分时间,每个通道组占两位,通道组 0 占最低的两位,积分时间为 20 ms 时的代码为 2#10。

2~5 号字节分别用来设置通道组 0~3 的测量类型和测量范围。高 4 位为测量类型,测电压时为 2#0001。低 4 位为测量范围,±2.5 V 时为 2#0101,±10 V 时为 2#1001。

6~13 号字节用来设置上限值和下限值(未用)。

用"HW Config"组态时,已经对 ET 200 M 站的模拟输入模块作了如下的设置:测量类型为电压,测量范围为±10 V,积分时间为 20 ms。

网络中的主站为 CPU 416-2DP,在它的"Blocks"文件夹中,生成如表 8-17 所示的数据块 DB30,即要发送到模块的数据记录 1。

表 8-17 DB30 中模拟输入模块的数据记录 1

字节号	类型	起始值	注释
0.0	STRUCT		
+0.0	BYTE	B#16#00	禁止极限值中断与诊断中断
+1.0	BYTE	B#16#02	通道组 0 的积分时间为 20 ms
+2.0	BYTE	B#16#15	通道组 0 的测量类型和范围:电压,±2.5 V
+3.0~+5.0	BYTE		通道组 1~3 的测量类型和范围(未用)
+6.0~+13.0	BYTE		通道组 0~3 的上、下限值(未用)
=14.0	END_STRUCT		

下面是 OB1 中调用系统功能 SFC 55"WR_PARM"的程序:

```
CALL    "WR_PARM"
  REQ          :=M30.0                    //用 M30.0 触发操作
  IOID         :=B#16#54                  //输入模块的标识符
  LADDR        :=W#16#200                 //输入模块的地址(对应十进制数 512)
  RECNUM       :=B#16#1                   //数据记录号为 1(DR1)
  RECORD       :=P#DB30.DBX0.0 BYTE 14    //指向 DB30 中数据记录 DR1 的指针
  RET_VAL      :=MW32                     //错误代码返回值
  BUSY         :=M30.1
AN      M30.1                             //如果操作完成
R       M30.0                             //复位操作的起动信号
```

用电缆连接好主站和 ET 200M 从站,将程序和组态数据下载到 CPU 416-2DP,令它处于 RUN 模式,如果它与 ET 200M 上与 DP 有关的错误 LED("SF DP"LED 或"BUSF"LED)没有亮或闪烁,说明 DP 主站与 ET 200M 从站之间的用户数据通信在正常运行。

当 DP 主站被重新起动时,用这种方法更改的模拟量输入模块的参数将会丢失,模拟输入模块将从存放在系统数据块中的静态 DR1 中接收它的参数。

调试时令 M30.0 为 1,DB30 中的数据记录就被传到从站的模拟量输入模块了。

【例 8-2】 用 SFC 56"WR_DPARM",将数据块 DR1 中用"HW Config"定义的模块参数传送给 ET 200M 从站的模拟量输入模块。DR1 是为模拟输入模块预定义的,并存放在 CPU 相应的 SDB 中。

打开 SIMATIC 400(1)站"Blocks"文件夹中的组织块 OB1,用 STL 语言调用 SFC 56"WR

_DPARM",编写好下面的程序后,保存 OB1。

```
CALL    "WR_DPARM"
  REQ          :=M40.0          //触发传送操作
  IOID         :=B#16#54        //输入模块的标识符
  LADDR        :=W#16#200       //输入模块的地址(对应十进制数 512)
  RECNUM       :=B#16#1         //数据记录号(DR1)
  RET_VAL      :=MW42           //错误代码返回值
  BUSY         :=M40.1
AN      M40.1                   //如果操作完成
R       M40.0                   //复位操作的起动信号
```

8.5 PROFINet

PROFINet 是为实现 PROFIBUS 与外部系统横向纵向整合的需要而提出的解决方案。它以互联网和以太网标准为基础,建立了一条 PROFIBUS 与外部系统的透明通道。

PROFInet 首次明确了 PROFIBUS 和工业以太网之间数据交换的格式,使跨厂商、跨平台的系统通信问题得到了彻底的解决。该技术为当前的用户提供了一套完整高性能可伸缩的升级至工业以太网平台的解决方案。PROFInet 技术基于开放、智能的分布式自动化设备,将成熟的 PROFIBUS 现场总线技术的数据交换技术和基于工业以太网的通信技术整合到一起,定义了一个满足 IT 标准的统一的通信模型。

PROFInet 提供了一种全新的工程方法,即基于组件对象模型(Component Object Model,COM)的分布式自动化技术;PROFINet 规范以开放性和一致性为主导,以微软公司的 OLE/COM/DCOM 为技术核心,最大程度地实现了开放性和可扩展性,向下兼容传统工控系统,使分散的智能设备组成的自动化系统模块化。PROFINet 指定了 PROFIBUS 与国际 IT 标准之间的开放和透明的通信;提供了一个独立于制造商,包括设备层和系统层的完整系统模型,保证了 PROFIBUS 和 PROFINet 之间的透明通信。

1. PROFINet 的通信机制

PROFINet 的基础是组件技术,在 PROFINet 中,每个设备都被看作一个具有组件对象模型(COM)接口的自动化设备,同类设备都具有相同的 COM 接口,系统通过调用 COM 接口来实现设备功能。组件模型使不同的制造商能遵循同一原则,它们创建的组件能在一个系统中混合应用,并能极大地减少编程的工作量。同类设备具有相同的内置组件,对外提供相同的 COM 接口,使不同厂家的设备具有良好的互换性和互操作性。COM 对象之间通过 DCOM(分布式 COM)连接协议进行互联和通信。传统的 PROFIBUS 设备通过代理设备(Proxy)与 PROFINet 中的 COM 对象进行通信。COM 对象之间的调用是通过 OLE(Object Linking and Embedding,对象链接与嵌入)自动化接口实现的。

PROFINet 用标准以太网作为连接介质,使用标准的 TCP/UDP/IP 协议和应用层的 RPC/DCOM 来完成节点之间的通信和网络寻址。

设备在建立连接时可以选择使用哪种实时通信协议,这样可以满足系统对较高的通信实时性的需求。

PROFIBUS 网段可以通过代理设备连接到 PROFINet，PROFIBUS 设备和协议可以原封不动地在 PROFINet 中使用。

2. PROFINet 的技术特点

PROFINet 的开放性基于以下的技术：微软公司的 COM/DCOM 标准、OLE、ActiveX 和 TCP/UDP/IP 协议。

PROFINet 定义了一个运行对象模型，每个 PROFINet 都必须遵循这个模型。该模型给出了设备中包含的对象和外部都能通过 OLE 进行访问的接口和访问的方法，对独立的对象之间的联系也进行了描述。

在运行对象模型中，提供了一个或多个 IP 网络之间的网络连接，一个物理设备可以包含一个或多个逻辑设备，一个逻辑设备代表一个软件程序或由软硬件结合体组成的固件包，它在分布式自动化系统中对应于执行器、传感器和控制器等。

在应用程序中将可以使用的功能组织成固定功能，可以下载到物理设备中。软件的编制严格独立于操作系统，PROFINet 的内核经过改写后可以下载到各种控制器和系统中，并不要求一定是 Windows 2000/NT 或 Windows CE 操作系统。

组件技术不仅实现了现场数据的集成，也为企业管理人员通过公用数据网络访问过程数据提供了方便。在 PROFINet 中使用了 IT 技术，支持从办公室到工业现场的信息集成，PROFINet 为企业的制造执行系统 MES 提供了一个开放式的平台。

从图 8-27 可以看出，PROFINet 技术的核心是代理设备(Proxy)，代理设备负责将所有的 PROFIBUS 网段、以太网设备和 PLC、变频器、现场设备等集成到 PROFINet 中，代理设备完成的是 COM 对象中的交互，它将挂接的设备抽象为 COM 服务器，设备之间的交互变为 COM 服务器之间的相互调用。只要设备能够提供符合 PROFINet 标准的 COM 服务器，该设备就可以在 PROFINet 网络中正常运行。

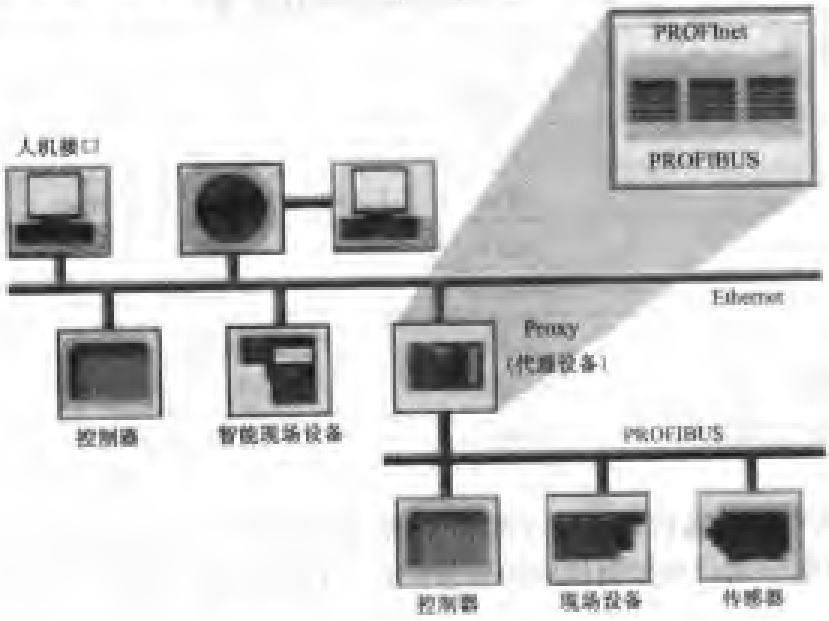


图 8-27 PROFINet 系统结构图

第 9 章　点对点通信

9.1　点对点通信的硬件与通信协议

点对点(Point to Point)通信简称为 PtP 通信,使用带有 PtP 通信功能的 CPU 或通信处理器,可以与 PLC、计算机或别的带串口的设备通信,例如打印机、机器人控制器、调制解调器、扫描仪和条形码阅读器等。

9.1.1　点对点通信处理器与集成的点对点通信接口

没有集成 PtP 串口功能的 S7-300 CPU 模块用通信处理器 CP 340 或 CP 341 实现点对点通信。S7-400 CPU 模块用 CP 440 和 CP 441 实现点对点通信。

1. CP 340 通信处理器

CP 340 通信处理器是串行通信较经济的解决方案,用于 S7-300 和 ET 200 M (S7 作为主站)的点对点串行通信,它有 1 个通信接口,有 4 种不同的型号,都有中断功能。一种模块的通信接口为 RS-232C(V.24),可以使用通信协议 ASCII 和 3964(R)。另外 3 种模块的通信接口分别为 RS-232C(V.24),20 mA(TTY)和 RS-422 / RS-485(X.27),可以使用的通信协议有 ASCII,3964(R)和打印机驱动软件。CP340 通信处理器技术规范如表 9-1 所示。

表 9-1　CP 340 通信处理器技术规范

接口类型	RS-232C(V.24)	20 mA(TTY)	RS-422/485(X.27)
数量	1 个,隔离	1 个,隔离	1 个,隔离
传输速率	2.4～19.2 kbit/s	2.4～9.6 kbit/s	2.4～19.2 kbit/s
电缆长度	15 m	100 m/1000 m (主动/被动)	1200 m
ASCII 最大帧长 ASCII 最大传输率	1024 B 9.6 kbit/s	1024 B 9.6 kbit/s	1024 B 9.6 kbit/s
3964(R) 最大帧长 3964(R) 最大传输速率	1024 B 19.2 kbit/s	1024 B 19.2 kbit/s	1024 B 19.2 kbit/s
打印机驱动软件的最大传输速率	9.6 kbit/s	9.6 kbit/s	9.6 kbit/s

ASCII 采用简单的传输协议连接外部系统,采用起始字符、结束字符和带块校验字符的协议,可以通过用户程序询问和控制接口的握手操作。打印机驱动器用于在打印机上记录过程状态和事件。

通过标准化的开放的西门子 3964(R)协议,PLC 可以与西门子的设备或第三方设备通信。

通过集成在 STEP 7 中的硬件组态工具,对各种点对点通信处理器(CP)进行参数设置,也可以通过 CPU 中的数据块来设置通信参数。

2. CP 341 通信处理器

CP 341 是点对点的快速、功能强大的串行通信处理器模块，有一个通信接口，用于 S7-300 和 ET 200M（S7 作为主站），可以减轻 CPU 的负担。CP 341 有 6 种不同的型号，可以使用的通信协议包括 ASCII、3964(R)、RS 512 协议和可装载的驱动程序，包括 MODBUS 主站协议、MODBUS 从站协议和 Data Highway(DF1 协议)，RK 512 协议用于连接计算机。

CP 341 有 3 种不同的传输接口：RS-232C(V.24)、20 mA(TTY)和 RS-422 / RS-485(X.27)。每种通信接口分别有两种类型的模块，其区别在于一种有中断功能，而另一种则没有。RS-232C(V.24)和 RS-422 / RS-485(X.27)接口的传输速率提高到 76.8 kbit/s，20 mA(TTY)接口的最高为 19.2 kbit/s。

通过装载单独购买的驱动程序，CP 341 可以使用 RTU 格式的 MODBUS 协议，在 MODBUS 网络中可以作主站或从站。

3. S7-300C 集成的点对点通信接口

CPU 313-2PtP 和 314C-2PtP 有一个集成的串行通信接口 X27(即 RS422/485)，CPU 313C -2PtP 可以使用 ASCII 和 3964(R)通信协议；CPU 314C-2PtP 可使用 ASCII，3964(R)和 RK512 协议。它们都有诊断中断功能。最多传输 1 024 个字节。全双工的传输速率为 19.2 kbit/s，半双工的传输速率为 38.4 kbit/s。

4. CP 440 点对点通信处理器

CP 440 用于点对点串行通信，物理接口为 RS-422 / RS-485(X.27)。最多 32 个节点，最高传输速率为 115.2 kbit/s，通信距离最长 1 200 m。可以使用的通信协议为 ASCII 和 3964(R)。

5. CP 441-1/CP 441-2 点对点通信处理器

CP 441-1 有 4 种不同的型号，通信处理模块可以插入一块分别带一个 20 mA(TTY)，RS-232C 或 RS-422/485 接口的 IF 963 子模块。有一种只有 3964(R) 通信协议，其余 3 种均有 ASCII、3964(R) 和打印机通信协议，有两种有多 CPU 功能。只有一种模块同时有多 CPU 和诊断中断功能。

CP 441-1 的 20 mA(TTY)接口的最大通信速率为 19.2 kbit/s，其余的接口为 38.4 kbit/s。最大通信距离同 CP 340。

CP 441-2 通信处理模块有 4 种不同的型号，可以插入两块分别带 20 mA(TTY)、RS-232C 和 RS-422/485 的 IF 963 子模块。有一种只有 RK512 和 3964(R) 通信协议，其余 3 种均有 RK512、ASCII、3964(R)和打印机通信协议，有多 CPU 功能，还可以实现用户定制的协议。只有一种模块同时有多 CPU 和诊断中断功能。

CP 441-2 的 20mA(TTY)接口的最大通信速率为 19.2 kbit/s，其余的接口为 115.2 kbit/s。最大通信距离同 CP 340。

9.1.2 ASCII Driver 通信协议

S7-300/400 的点对点串行通信可以使用的通信协议主要有 ASCII driver，3964(R)和 RK512。它们在 ISO 七层参考模型中的位置如图 9-1 所示。

1. ASCII Driver 的报文帧格式

ASCII driver 用于控制 CPU 和一个通信伙伴之间的点对点连接的数据传输，可以将全部

发送报文帧发送到 PtP 接口，提供一种开放式的报文帧结构。接收方必须在参数中设置一个报文帧的结束判据，发送报文帧的结构可能不同于接收报文帧的结构。

使用 ASCII driver 可以发送和接收开放式的数据(所有可以打印的 ASCII 字符)，8 个数据位的字符帧可以发送和接收所有 00～FFH 的其他字符。7 个数据位的字符帧可以发送和接收所有 00～7FH 的其他字符。

ASCII driver 可以用结束字符、帧的长度和字符延迟时间作为报文帧结束的判据。用户可以在三个结束判据中选择一个。

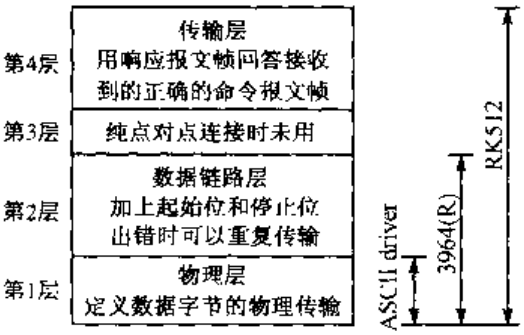


图 9-1　PtP 协议在 ISO 参考模型中的位置

(1) 用结束字符作为报文帧结束的判据

报文帧用一个或两个用户可以定义的结束字符来表示报文帧的结束，有 3 种选择：

1) 传输包括结束字符的数据，结束字符必须包含在被发送的数据中。即使在 SFB 中规定的数据长度很大，数据也只能传送到结束字符。

2) 传输数据的最大长度用 SFB 的参数声明，最后一个或最后两个字符必须为结束字符。

3) 传输数据的最大长度用 SFB 的参数声明，并自动添加结束字符，即结束字符不包括在被传输的字符个数中(最大为 1024 B)。

如果使用结束字符，应保证在用户数据中不包括结束字符。

(2) 用固定的字节长度作为报文帧结束的判据

接收方收到约定个数的字符(1～1024 B)即认为报文帧结束。用如图 9-2 所示的字符延迟时间作为监控时间，如果在接收完设置数量的字符之前，字符延迟时间到，将关闭接收操作，同时生成一个出错报文。

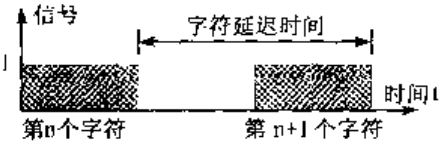


图 9-2　字符延迟时间

如果接收到的字符长度大于设置的固定长度，接收到的多余的字符将被删除。如果接收到的字符长度小于设置的固定长度，报文帧将被删除。

(3) 用字符延迟时间作为报文帧结束的判据

报文帧既没有设置固定的长度，也没有用户定义的结束符；接收方在约定的时间(字符延迟时间)内未收到新的字符则认为报文帧结束(超时结束)。

组态时应保证字符延迟时间小于两个报文帧之间的间隔时间，但是字符延迟时间又不能太短，以保证不会将通信伙伴在发送报文帧时的暂停，错误地识别为是报文帧的结束。

2. **数据流控制/握手(Data Flow Control/ Handshaking)**

数字通信中常用"握手"方式控制两个通信伙伴之间的数据流。握手可以保证两个以不同速度运行的设备之间传输的数据不会丢失。有两种不同的握手方式：

(1) 软件方式，例如通过向对方发送特定的字符(例如 XON/XOFF)实现数据流控制，报文帧中不允许出现 XON 和 XOFF 字符。

(2) 硬件方式，例如用信号线 RTS/CTS 实现数据流控制，接口应使用 RS-232C 完整的接线。

一只通信处理器被切换到流控制运行状态，如果报文帧从接收缓冲区中被取走，接收缓冲区已经准备好接收数据，就会发送 XON 字符或使输出信号 RTS 线为 ON，表示可以接收

数据。

如果报文帧接收完成，或接收缓冲区(CPU 31xC 为 2048 B，CP 340 为 1024 B)只剩 50 B，CPU 31xC-2PtP 或 CP 将发送字符 XOFF，或使 RTS 线变为 OFF，表示不能接收数据。如果发送方继续发送，造成接收缓冲区溢出，将生成出错报文，最后一个报文帧中接收的数据将被丢弃。一旦报文帧被 CPU 从接收缓冲区中取走，并且接收缓冲区已准备好接收数据，CPU 31xC-2PtP 或 CP 将发送 XON 字符，或将 RTS 线置为 ON。

如果接收到 XOFF 字符，或通信伙伴的 CTS 控制信号被置为 OFF，将中断数据传输。

如果在预定的时间内未收到 XON 字符，或通信伙伴的 CTS 控制信号为 OFF，将取消发送操作，并且在功能块的输出参数 STATUS 中生成一个出错信息。

3. CPU 31xC-2PtP 中的接收缓冲区

接收缓冲区的容量为 2048 B。在参数设置时，可以设置禁止改写接收缓冲区中的某些数据，还可以规定允许缓冲区接收的报文帧的最大个数(1～10)，或使用整个接收缓冲区。可以设置在起动时清除接收缓冲区。

接收缓冲区是一个 FIFO(先入先出)缓冲区，如果有多个报文帧被写入接收缓冲区，总是第一个接收到的报文帧被传送到目标块中。如果想将最新接收的报文帧传送到目标块中，必须将缓存的报文帧个数设置为 1，并取消改写保护。

9.1.3 ASCII Driver 通信协议的参数设置

启动 SIMATIC 管理器，在项目中调用硬件组态工具“HW Config”。如果使用点对点通信处理器模块，例如 CP 341 或 CP 440 等，双击通信模块。如果使用 CPU 31xC -2PtP，双击 CPU 模块中的“PtP”子模块，打开“属性”对话框。设置好通信参数后，用菜单命令“Station”→“Save and compile”保存。在 CPU 处于 STOP 模式时，将参数数据下载到 CPU 中。下面介绍 CPU 31xC-2PtP 的 ASCII 通信参数的设置。

1. 基本参数的设置

打开 PtP 属性对话框中，首先在最上面的选择框中选择通信协议为“ASCII”(见图 9-3)，在“Addresses”选项卡中，可以定义输入的起始地址，关闭该选项卡时自动修改结束地址。系统选择的默认起始地址为 1023。

在“Basic Parameters”(基本参数)选项卡，可以选择是否允许诊断中断和 CPU 进入 STOP 模式时对通信的处理(停止或继续)。在“Transfer”(传输)选项卡(见图 9-3)，可以设置通信速率(300～38 400 kbit/s)，数据位的位数(7 位或 8 位)，结束位的位数(1 位或 2 位)，和奇偶校验的方式，Odd、Even 和 None 分别是奇校验、偶校验和无校验，设置为 7 个数据位时不能选择无校验。

如果选择了“data flow control”(数据流控制)，可以设置 XON 和 XOFF 字符，默认值分别为十六进制数 11 H 和 13 H。还可以设置在发送之后等待接收到 XON 字符的时间(20～65530 ms)。它以 10 ms 为增量，默认值为 20 000 ms。

在“Data Reception”(数据接收)选项卡，如果选择“Clear the CPU Receive Buffer at Start-up”，CPU 从 STOP 模式切换到 RUN 模式时清除接收缓冲区。

如果选择“Preventing overwriting”，可以防止在接收缓冲区装满时数据被改写。

如果选择“Use entire buffer”(使用整个接收缓冲区，2048 B)，保存在接收缓冲区内的报文

帧的个数取决于帧的长度。

未选"Use entire buffer"时，可以设置在接收缓冲区中保存的报文帧的最大个数(Maximum number of buffered received message frames)，可以选择1～10个报文帧，默认值为10。如果选择1，将禁止"防止改写"功能，在用户程序中定期读取接收到的数据，当前报文帧被传送给CPU。

2. 报文帧结束判据的设置

在图9-4中的"End Delimiter"(结束分界符)选项卡，可以选择3种报文帧结束的判据：

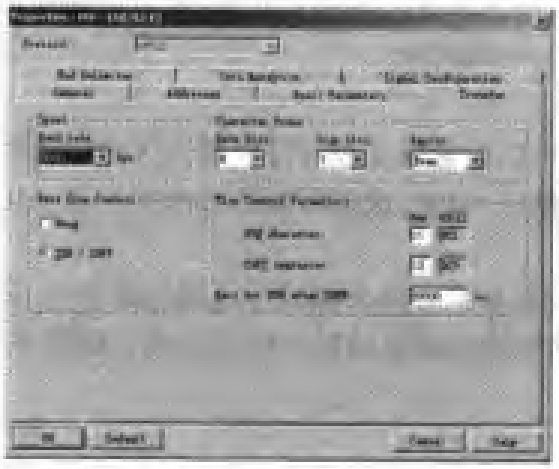


图9-3 PtP通信接口的参数设置

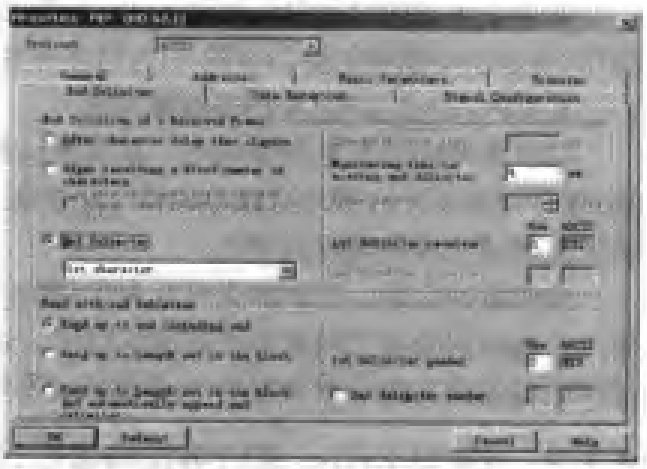


图9-4 End Delimiter选项卡

(1) 若选择系统默认的"After character delay time elapses"，则用字符延迟时间(1～65535 ms，默认值为4 ms)作为报文帧结束的判据，最短的字符延迟时间与传输速率有关，传输速率为38.4 kbit/s时为1 ms。

(2) 若选择"After receiving a fixed number of characters"，则用固定字节长度作为报文帧结束的判据。在这种情况，接收到的报文帧的长度总是相同的。

在输入域"Message frame length"输入报文帧的字节数(1～1024 B)，默认值为200 B。

在输入域"Monitoring time for missing end-of-message criteria"设置字符延迟时间，作为报文结束判据的监视时间，默认值为4 ms。它用于下列组态：

1) 接收到固定数量的字符后报文帧结束；

2) 接收到结束分界符后报文帧结束。

若选择"Pause in transmission between frames"(报文帧之间的发送暂停监视时间)，在发送的两个相邻的报文帧之间将暂停一段时间，它与丢失结束字符的监视时间相对应，以便允许通信伙伴识别接收的报文帧，从而使之同步。

(3) 若选择报文帧的结束判据为"End delimiter"，用一个或两个结束符来表示报文帧的结束，在报文的正文中不允许出现与结束标志相同的字符，以避免通信伙伴误认为报文帧结束。而使用前两种报文结束判据时，报文帧是完全透明的，即任何字符串都可以在报文帧中使用。

BCC(block check characters)为块校验字符，BCC是正文中的所有字符"异或"运算的结果，各字符同一位中"1"的个数为奇数时，异或的结果为"1"，为偶数时异或的结果为"0"(见图9-5)。这种校验方式又称为"纵向奇偶校验"。

结束分界符有6种不同的选项，可以选择1个或2个结束符，同时还可以选择1个、2个

或没有块校验字符(BCC)。默认的设置为1个结束符,无BCC。发送方和接收方的BCC计算和检查由用户程序完成。

在"1st delimiter sender"和"2nd delimiter sender"文本框内,可以设置第一个和第二个结束字符,它们的默认值分别为03H和0(即ASCII码ETX和NUL)。

选择默认的设置"Send up to and including end"时,结束字符应包含在被发送的数据中。即使在SFB中设置了较大的数据长度,数据只能传送到结束字符。

选择"Send up to the length set in the block"时,只传送在SFB中规定的长度的数据。最后一个字符必须为结束字符。

选择"Send up to the length set in the block and automatically append end delimiter"时,传送数据长度最大为SFB参数中设置的数据,并自动附加文本结束符。在被传输的数据中不包括结束字符。根据结束字符的个数,可以将比SFB中的规定(最大为1024 B)多一个或两个字符传送给通信伙伴。

3. 信号组态

在"Signal Configuration"(信号组态)选项卡左边的运行模式(Operating Mode)区内(见图9-6),可以选择X27(RS-422/485)接口的3种运行方式。

30H = 0011 0000
31H = 0011 0001
32H = 0011 0010
10H = 0001 0000
03H = 0000 0011
XOR = 0010 0000

图9-5 BCC计算举例

图9-6 信号组态选项卡

- "Full Duplex (RS-422) Four-Wire Point-to-point Mode"是4线点对点全双工(RS-422)模式。
- "Full Duplex (RS-422) Four-Wire Multipoint Master Mode"是全双工(RS-422)4线多点主站模式。
- "Half Duplex (RS-485) Two-Wire Mode"是半双工(RS-485)2线模式,CPU可以作主站或从站。

在"Receive line initial state(接收线的初始状态)"区内,有3个选项:

- "None(无)":R(A)和R(B)信号线无初始电压(见图9-7),该设置只能用于总线连网的专用驱动器。
- "Signal R(A) 5 volts, Signal R(B) 0 volts":R(A)信号线为5 V,R(B)信号线为0 V,

只能进行断路识别，不能用于全双工(RS-422)4 线多点主站模式和半双工(RS485)双线操作模式。

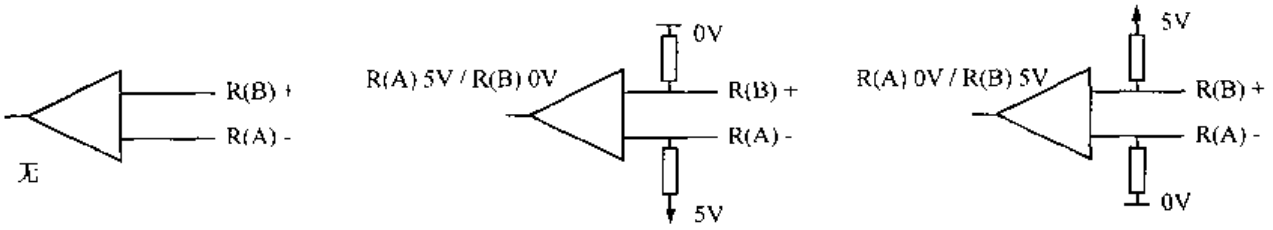


图 9-7　RS-422/485 接收器布线图

- "Signal R(A) 0 volts, Signal R(B) 5 volts"：R(A) 信号线为 0 V，R(B) 信号线为 5 V，表示空闲状态(没有发送站被激活)，在该状态不能进行断路识别。

9.1.4　3964(R)通信协议

3964(R)协议用于 CP 或 CPU 31xC-2PtP 和一个通信伙伴之间的点对点数据传输。

1. 3964(R)协议使用的控制字符与报文帧格式

3964(R)协议将控制字符(见表 9-2)添加到用户数据中，控制字符用来表示报文帧的开始和结束，它们也是通信双方的"握手"信号。通信伙伴使用这些控制字符，检查数据是否被正确和完整地接收。

表 9-2　3964(R)协议使用的控制字符

控制字符	数　值	说　明
STX	02H	被传送文本的起始点
DLE	10H	数据链路转换(Data Link Escape)或肯定应答
ETX	03H	被传送文本的结束点
BCC		块校验字符(Block Check Character)，只用于 3964(R)
NAK	15H	否定应答(Negative Acknowledge)

3964(R)传输协议的报文帧(见图 9-8)有附加的块校验字符(BCC)，用来增强数据传输的完整性，3964 协议的报文帧没有块校验字符。

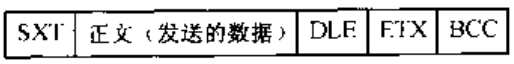


图 9-8　3964(R)报文帧格式

BCC 是所有正文中的字符(包括正文中连发的 DLE)和报文帧结束标志(DLE 和 ETX)的"异或"运算的结果。

3964(R)报文帧的传输过程如图 9-9 所示。首先用控制字符建立通信链路，然后用通信链路传输正文，最后在传输完成后用控制字符断开通信链路。

3964(R)的正文字符是完全透明的，即任何字符都

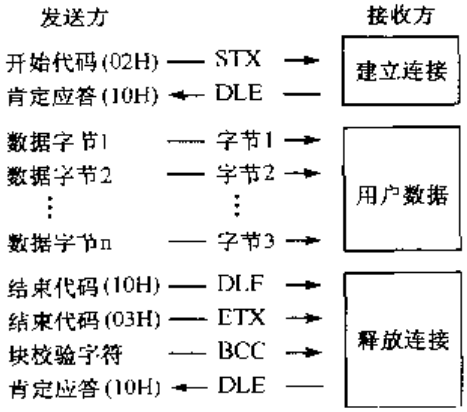


图 9-9　3964(R)报文帧传输过程

可以用在正文中。为了避免接收方将正文中的字符 10H(即 DLE)误认为是报文结束标志,正文中如果有字符 10H,在发送时将会自动重发一次。接收方在收到两个连续的 10H 时将会自动地剔除一个。

2. 建立发送数据的连接

为了建立连接,发送方首先应发送控制字符 STX(见图 9-9)。如果在"应答延迟时间(ADT)"到来之前,接收到接收方发来的控制字符 DLE,则表示通信链路已成功地建立,切换到发送模式,可以开始传输正文。

如果通信伙伴返回 NAK 或返回除 DLE 和 STX 之外的其他控制代码,或应答延迟时间到时没有应答,程序将再次发送 STX,重试连接。若约定的重试次数到后,都没有成功建立通信链路,程序将放弃建立连接,并发送 NAK 给通信伙伴,同时通过输出参数 STATUS 向功能块 P_SND_RK 报告出错。

3. 使用 3964(R)通信协议发送数据

成功建立连接后,将使用选择的传输参数,把发送缓冲区中的用户数据发送给通信伙伴。通信伙伴监控接收到的相邻两个字符之间的时间间隔,该时间间隔不能超过字符延迟时间。

在传输过程中,如果通信伙伴发送了控制代码 NAK,传输过程中止,并重试建立连接。如果接收到其他字符,也中止传输过程,并延时到"字符延迟时间"后发送 NAK 字符,将通信伙伴置于空闲状态。然后,通过再发送 STX,重新起动发送操作。

发送完缓冲区的内容后,自动加上代码 DLE、TX 和 BCC,BCC 由 CP 或 CPU 31xC-2PTP 计算。

发送完成后,等待接收方回送肯定应答字符 DLE。如果通信伙伴在应答延迟时间内发送了 DLE,即表示数据块被正确接收。发送缓冲区内的数据被删除,并断开通信链路。

如果通信伙伴返回 NAK 或返回除了 DLE 之外的其他控制代码,或返回损坏的代码,或应答延迟时间到时没有应答,程序将再次发送 STX,重试连接。若约定的重试次数到后,都没有成功建立通信链路,程序将放弃建立连接,并发送 NAK 给通信伙伴,同时通过输出参数"STATUS"向发送功能块报告出错。

4. 使用 3964(R)通信协议接收数据

在准备操作时,3964(R)协议将发送一个 NAK 字符,以便将通信伙伴置于空闲状态。在空闲状态,如果没有发送请求被处理,程序将等待通信伙伴建立连接。

用 STX 建立连接时,如果没有空的接收缓冲区可用,将等待 400 ms。延时时间到后仍然没有空的接收缓冲区,将发送 NAK 给对方,然后进入空闲状态。通信功能块的"STATUS"输出将报告出错。若延时后有接收缓冲区可用,将发送一个 DLE 字符,并进入接收状态。

如果在空闲状态接收到除 STX 或 NAK 之外的其他控制代码,将等待"字符延迟时间"到,然后发送 NAK 字符,同时通过输出参数"STATUS"向发送功能块报告出错。

成功建立连接后,接收到的字符被写入接收缓冲区。如果接收到两个连续的 DLE 字符,只有一个被保存在接收缓冲区中。

接收到每个字符后,如果在字符延迟时间到时还没有接收到下一个字符,将发送一个 NAK 给通信伙伴。系统程序将通过输出参数"STATUS"向发送功能块报告出错,3964(R)程序不再重新初始化。

如果在接收过程中出现传输错误,如丢失字符、帧错误和奇偶校验错误等,将继续接收数据,

直到连接被释放。然后将向通信伙伴发送 NAK 字符,期待对方再次建立通信链路,重发报文帧。

如果在设置的重试次数后还没有正确地接收到报文帧,或者在规定的块等待时间内(4 s),通信伙伴没有重发报文帧,将取消接收操作。通过输出参数"STATUS"报告第一次错误的传输,最后中止接收。

5. 故障与传输冲突的处理

在接收到 DLE、ETX 和 BCC 后,根据接收到的数据计算 BCC,并与通信伙伴发送过来的 BCC 进行比较。如果二者相等,并且没有其他接收错误发生,接收方的 CPU 将发送 DLE,断开通信连接。

如果二者不等,将发送 NAK,在规定的块等待时间内(4 s)等待重新发送。如果在设置的重试次数内没有接收到报文,或者在块等待时间内没有进一步的尝试,将取消接收操作。

如果一台设备通过发送代码 STX,而不是应答 DLE 或 NAK,来响应应答延迟时间(ADT)内通信伙伴发送的请求代码 STX,就出现了初始化冲突,即两台设备都请求发送。具有较低优先级的设备将暂时放弃其发送请求,向对方发送控制字符 DLE。具有较高优先级的设备将以上述方式发送其数据。等到高优先级的传输结束,连接被释放,具有较低优先级的设备就可以执行其发送请求。为了解决初始化冲突,通信的双方必须设置不同的优先级。

程序可以识别出通信伙伴和线路故障引起的错误。在这两种情况下,程序将重复尝试正确地接收和发送数据块。如果在设置的重试次数内没有成功,或出现了新的错误,程序将取消发送或接收过程,并报告识别的第 1 个错误的错误编号,然后返回空闲状态。这些错误信息将出现在通信功能块的输出参数 STATUS。

如果系统程序在通信功能块的输出 STATUS 频繁地报告一个重复发送和接收错误,说明数据传输被偶然干扰。大量地重试传输可以对此进行补偿,但是此时应检查通信电路的干扰源,因为频繁的重复尝试会降低用户数据的传输速率和传输的完整性,但是干扰也可能由通信伙伴的故障造成。

如果接收连接中断,在通信功能块的输出"STATUS"中将显示一个出错报文,并且不会启动重试。在通信线路重新连接后,将立刻自动复位通信功能块的"STATUS"输出中的"BREAK"(断开)状态。

6. 3964(R)通信协议的参数设置

设置 3964(R)通信协议的参数时,"Addresses","Basic Parameters","Data Reception" 与 "Signal Assignment"选项卡中的参数设置方法与 ASCII driver 通信协议中的相同。

在"Transfer"选项卡中(见图 9-10),除了设置通信速率、数据位和结束位的位数,以及奇偶校验位以外,还可以设置下面的参数:

(1) 使用块校验(With block check)

选择使用块校验时,当接收方检测到字符串"DLE,ETX,BCC"便中断数据接收。它将进行上述的块校验操作。

未选择使用块校验时,接收方检测到 DLE,ETX 字符串便停止接收操作。如果是无错误的接收,它将发送 DLE 字符给通信伙伴,否则将发送 NAK 字符。

(2) 优先级(Priority)

在优先级选择框内,可以选择高优先级(High)或低优先级(Low)。默认的设置为"High"。可以选择是否使用块校验(With block check)。必须为两个通信伙伴设置不同的优先级,即一

个为高优先级，另一个为低优先级。

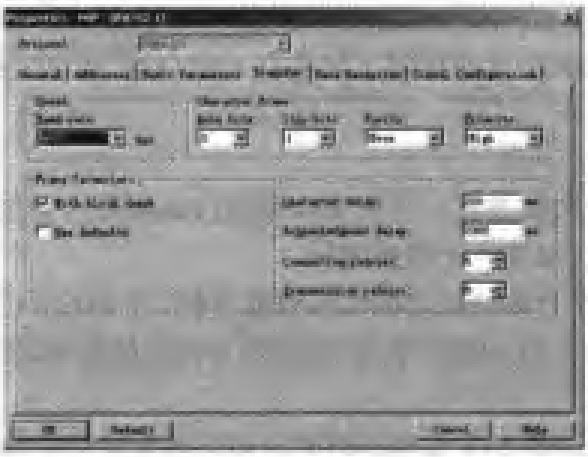


图 9-10　3964(R)通信协议的参数设置

(3) 字符延迟时间(Character delay time)

定义了报文中接收的两个相邻字符之间允许的最大时间间隔。设置范围为 20～65530 ms，间隔为 10 ms，默认值为 220 ms。

(4) 应答延迟时间(Acknowledgement delay time)

定义了当建立连接或关闭连接时与通信伙伴之间的应答的最大允许时间。建立连接时的应答延迟时间是指发送 STX 和通信伙伴返回 DLE 应答之间的延迟时间，断开连接时的应答延迟时间是发送方发出的 DLE，ETX 和接收方发出的 DLE 应答之间的延迟时间。设置范围为 20－65530 ms，间隔为 10 ms，默认值为 2000 ms。

(5) 连接尝试(Connection retries)

定义了建立一个连接的最大尝试次数(1～255 次)，默认值为 6 次。

(6) 传输尝试(Transmit retries)

定义了出错时传输一个报文帧的最大尝试次数，包括第 1 个报文帧，可以设置为 1－255，默认值为 6 次。

9.1.5　RK 512 通信协议

RK 512 协议又称为 RK 512 计算机连接，用于控制与一个通信伙伴之间的点对点数据传输。与 3964(R)协议相比，RK 512 协议包括 ISO 参考模型的物理层(第 1 层)、数据链路层(第 2 层)和传输层(第 4 层)，提供了较高的数据完整性和较好的寻址功能。

1. RK 512 的报文帧

(1) 响应报文帧

RK 512 协议用响应报文帧来响应每个正确接收到的命令帧。

(2) 命令报文帧

表 9-3 给出了命令报文帧的标题结构。命令帧包括 SEND 或 FETCH 报文帧：

1) SEND(发送)报文帧：SEND 报文帧用来将用户数据写入通信伙伴的数据区，它是带有要写入对方的用户数据的命令帧，通信伙伴返回一个不带用户数据的响应报文帧。

2) FETCH(读取)报文帧：FETCH 报文帧用来读取通信伙伴的数据区，它是带用户数据

区地址的命令帧。通信伙伴返回一个带用户数据的响应报文帧。

(3) 连续报文帧(Continuation Message Frame)

如果数据长度超过 128 B,发送的报文帧将自动地分为 SEND(或 FETCH)报文帧和连续报文帧。

(4) 报文帧的标题

对于 RK 512,每个报文帧都有一个报文帧标题(Header)。它包括报文帧标识符(ID)、数据源和数据目的地的信息,以及一个错误编号。

RK 512 寻址描述以字为单位的数据源和数据目标,在 SIMATIC S7 中,被自动地转换为字节地址。字节 3 和字节 4 为 ASCII 字符,连续命令报文帧的标题仅由 1～4 号字节组成。

(5) 响应报文帧

在发送命令报文帧后,RK 512 在监控时间内等待通信伙伴的响应报文帧。监控时间的长短取决于传输速率(波特率),300～76.8 kbit/s 时为 10 s。

表 9-3 命令报文帧的标题结构

字 节	说 明
1	报文帧 ID,命令报文帧为 00H,连续命令报文帧为 FFH
2	报文帧 ID(00H)
3	'A'(41H):带目标 DB 的 SEND 请求,'O'(4FH):带目标 DX 的 SEND 请求,'E'(45H):FETCH 请求
4	被传送的数据来自(发送时只能选"D"): 'D'(44H)= 数据块,'X'(58H)= 扩展数据块,'E'(45H):输入字节,'A'(41H)= 输出字节, 'M'(4DH)= 存储字节,'T'(54H)= 时间单元,'Z'(5AH):计数器单元
5,6	SEND 请求的数据目标,或 FETCH 请求的数据源,例如字节 5 = DB 号,字节 6 = DW 号
7,8	数据长度:根据类型,被传送数据的长度的字节数或字数
9	处理器通信标志位(KM)的字节编号;如果没有指定处理器通信标志位,输入数值 FFH
10	位 0～3:处理器通信标志位的位编号。如果没有指定处理器通信标志位,输入数值 FH 位 4～7:CPU 编号(1～4);如果没有设置 CPU 编号(其值为 0),但是设置了处理器通信标志位,输入数值 0H 如果没有设置 CPU 编号或处理器通信标志位,输入数值 FH

响应报文帧由 4 个字节组成(见表 9-4),包含处理请求的信息。

表 9-4 响应报文帧

字 节	说 明
1	报文帧 ID(标识符):响应报文帧为 00H,连续响应报文帧为 FFH
2	报文帧 ID(00H)
3	指定为 00H
4	响应报文帧中通信伙伴的错误编号:00H 表示传输过程中没有出现错误,>00H 为错误编号

根据响应报文帧中的错误编号,将自动生成功能块的输出参数"STATUS"中的事件号。

2. SEND 报文帧的数据传输过程

图 9-11 给出了 RK 512 协议用 SEND 报文帧发送数据的传输过程。

SEND 请求按下面的顺序执行:

(1) 主动通信伙伴(Active partner)发送一个 SEND 报文帧,包括报文帧的标题和数据。

(2) 被动通信伙伴(Passive partner)接收报文帧,检查标题和数据,将数据写入目标块后,用一个响应报文帧进行应答。

(3) 主动通信伙伴接收响应报文帧,如果用户数据长度超过 128 B,它将发送连续 SEND

报文帧。

(4) 被动通信伙伴接收连续 SEND 报文帧,检查标题和数据,将数据传送入目标块后,用一个连续响应报文帧进行应答。

(5) 如果接收到一个错误的 SEND 报文帧,或者在报文帧的标题中出现错误,通信伙伴在响应报文帧的第 4 个字节中输入一个错误编号,这不适用于协议出错的情况。

3. 连续 SEND 报文帧

如果用户数据长度超过 128 B,它将启动一个连续 SEND 报文帧。其处理方法与 SEND 报文帧相同。

发送的字节如果超出 128 B,多余的字节将自动地在一个或多个连续报文帧中发送。

图 9-12 给出了发送一个带连续响应报文帧的连续 SEND 报文帧的数据传输过程。

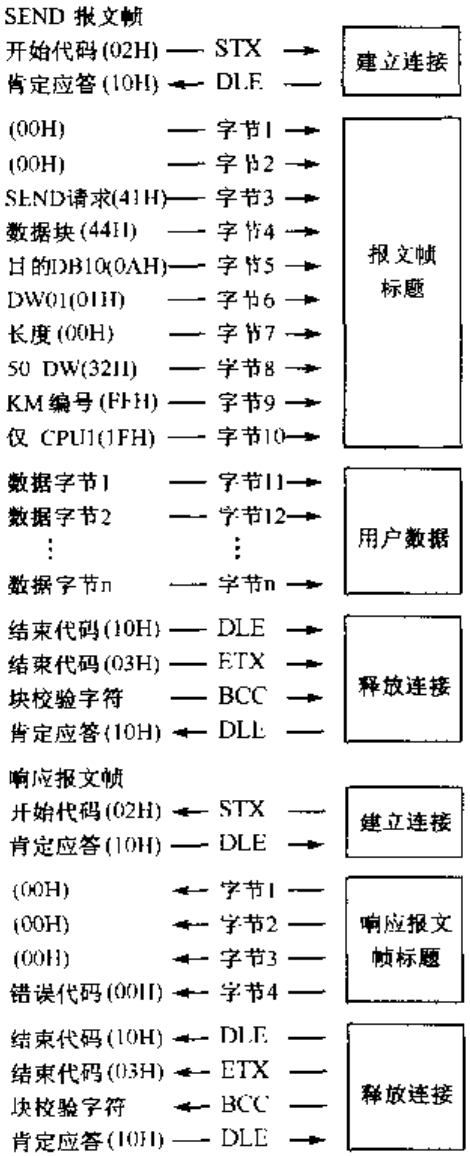


图 9-11 SEND 报文帧传输过程

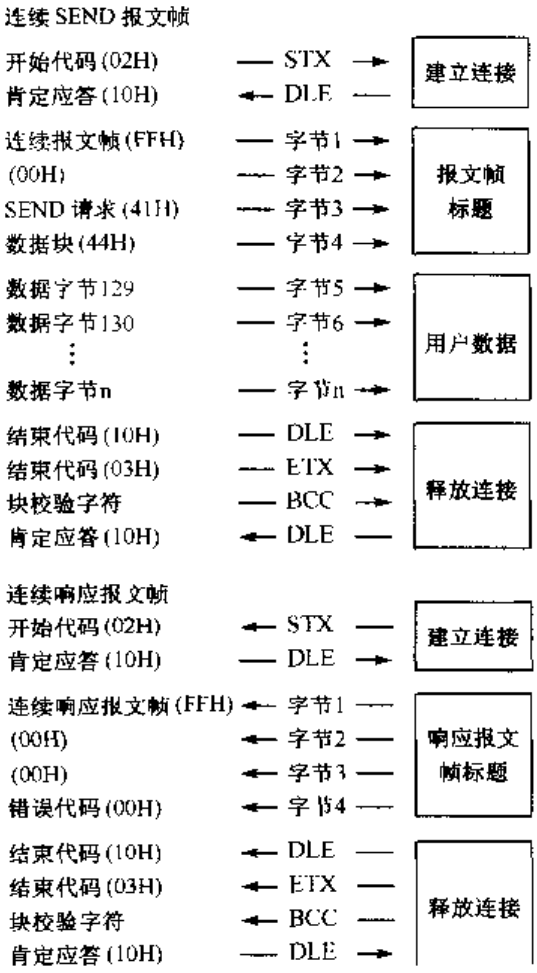


图 9-12 连续 SEND 报文帧传输过程

4. FETCH 报文帧的数据传输过程

图 9-13 给出了 RK 512 协议用 FETCH 报文帧读取数据的传输过程。

FETCH 请求按下面的顺序执行：

(1) 主动通信伙伴发送一个包括标题的 FETCH 报文帧。

(2) 被动通信伙伴接收报文帧，检查报文帧的标题，从 CPU 中读取数据，并用一个带有数据的响应报文帧进行应答。

(3) 主动通信伙伴接收响应报文帧。如果用户数据长度超过 128 B，它将发送一个连续 FETCH 报文帧，该报文帧的标题只有 1～4 号字节。

(4) 被动通信伙伴接收连续 FETCH 报文帧，检查报文帧的标题，从 CPU 中读取数据，并用一个包括剩余的数据的连续响应报文帧进行应答。

如果在第 4 个字节中有一个不等于 0 的出错编号，响应报文帧中不包含任何数据。

如果被请求的数据超过 128 B，将自动地用一个或多个连续报文帧读取额外的字节。

如果接收到一个错误的 FETCH 报文帧，或者在报文帧的标题中出现一个错误，通信伙伴在响应报文帧的第 4 个字节中输入一个错误编号。出现协议错误时在响应报文帧中不包含信息。图 9-14 给出了 RK 512 协议用一个连续响应报文帧读取数据的传输过程。

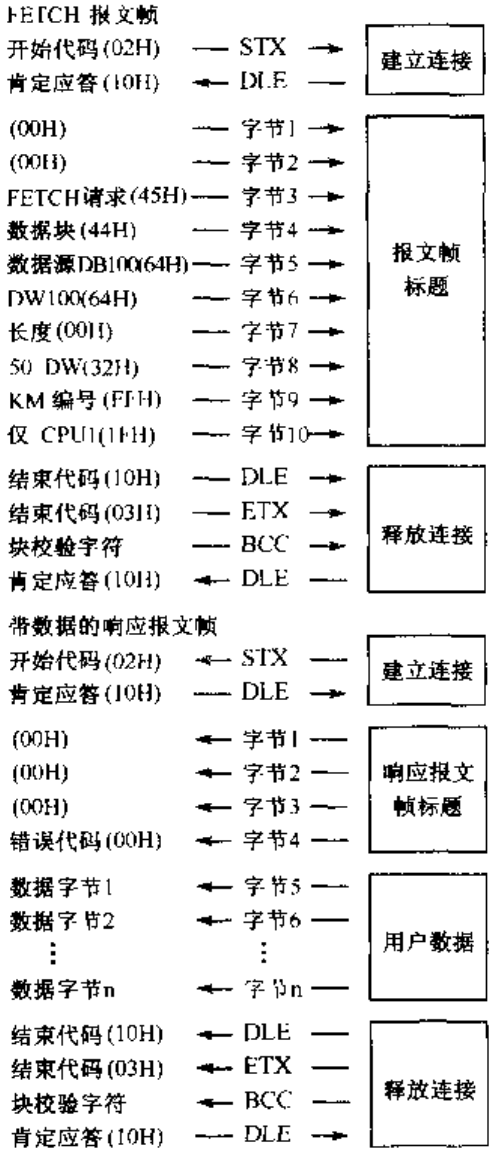


图 9-13 FETCH 报文帧传输过程

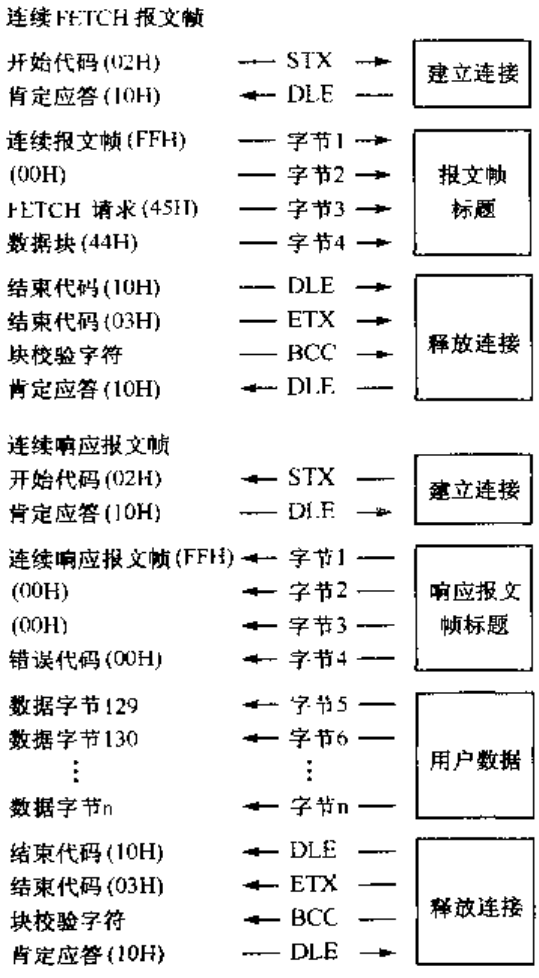


图 9-14 连续 FETCH 报文帧传输过程

5 伪全双工操作(Quasi-Full-Duplex Operation)

伪全双工操作是指只要其他伙伴没有发送报文,通信伙伴也可以在任何时候发送命令报文帧和响应报文帧。命令报文帧和响应报文帧的最大嵌套深度为1,即只有前一个报文帧被响应报文帧应答后,才能处理下一个命令报文帧。

在某些情况下,如果两个伙伴都请求发送,在响应报文帧之前,通信伙伴可以发送一个SEND报文帧。例如,在响应报文帧之前,通信伙伴的SEND报文帧已经进入了发送缓冲区。

在图9-15中,直到通信伙伴发送完SEND报文帧,才发送响应第1个连续SEND报文帧的连续响应报文帧。

图9-15 伪全双工方式

6. RK 512通信的参数设置

由于3964(R)是RK 512通信的一部分,RK 512协议的参数与3964(R)协议的参数基本相同。但是二者有下列区别:RK 512的字符固定设为8位,没有接收缓冲区,也没有接收数据的参数。必须在使用的系统功能块(SFB)中规定数据目标和数据源的参数。

9.2 用于CPU 31xC-2PtP点对点通信的系统功能块

在用户程序中,用专用的功能块来实现点对点串行通信。S7-31xC-2PtP用于点对点通信的系统功能块为SFB 60~65。SFB 60~62用于ASCII/3964(R)的通信,SFB 63~65用于RK 512的通信。它们在程序编辑器左边的指令树窗口的"Libraries"→"Standard Library"→"System Function Blocks"(系统功能块)文件夹中。

9.2.1 用于ASCII/3964(R)协议的系统功能块

表9-5列出了CPU 31xC-2PtP用于点对点通信的系统功能块。SFB不作参数检查,如果参数设置出错,CPU将进入STOP模式。

表9-5 CPU 31xC-2PtP用于点对点通信的系统功能块

系统功能块		说 明
SFB 60	SEND_PTP	将整个数据块或部分数据块区发送给一个通信伙伴
SFB 61	RCV_PTP	从一个通信伙伴接收数据,并将它们保存在一个数据块中
SFB 62	RES_RCVB	复位CPU的接收缓冲区
SFB 63	SEND_RK	将整个数据块或部分数据块区发送给一个通信伙伴
SFB 64	FETCH_RK	从一个通信伙伴处读取数据,并将它们保存在一个数据块中
SFB 65	SERVE_RK	从一个通信伙伴处接收数据,并将它们保存在一个数据块中;为通信伙伴提供数据

1. 用SFB60"SEND_PTP"发送数据(ASCII/3964(R))

块被调用后,在控制输入REQ的脉冲上升沿发送数据。SD_1为发送数据区(数据块编号和起始地址),LEN是要发送的数据块的长度。

用参数LADDR声明在HW Config(硬件组态)中指定的子模块的I/O地址。

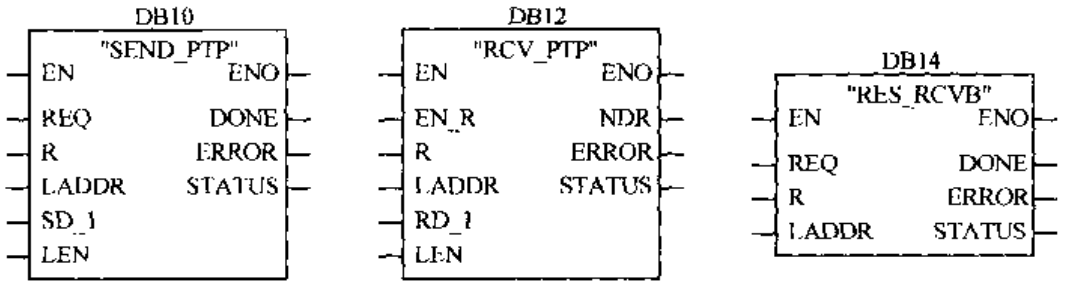


图 9-16　用于 ASCII/9364(R)通信协议的系统功能块

在控制输入 R 的脉冲上升沿，当前的数据发送被取消，SFB 被复位为基本状态。被取消的请求用一个出错报文(STATUS 输出)结束。为了使系统功能块正常工作，调用时必须保证 R 为 FALSE(0 状态)，它才能处理发送请求。

如果块执行没有错误，DONE 被置为 1 状态。如果出错，ERROR 被置为 1 状态。如果块被正确执行后 DONE 为 1，意味着：

(1) 使用 ASCII driver 时，数据被传送给通信伙伴，但是不能保证被对方正确地接收。

(2) 使用 3964(R)协议时，数据被传送给通信伙伴，并得到对方的肯定确认。但是不能保证数据被传送给对方的 CPU。

如果出现错误或报警，STATUS 将显示相应的事件标识符(ID)。在 SFB 的 R 为 1 时，也会输出 DONE 或 ERROR/STATUS。如果出现一个错误，CPU 的二进制结果位 BR 将被复位。如果块无错误结束，BR 的状态将被置为 1。

SFB 不作参数检查，如果参数设置出错，CPU 将进入 STOP 模式。

SFB 最多只能发送 206 个连续的字节。必须在参数 DONE 被置为 1，发送过程结束时，才能向 SD_1 指定的发送区写入新的数据。

2. 用 SFB 61"RCV_PTP"接收数据

SFB 61 用来接收数据，并将它们保存到一个数据块中。

调用 SFB 61 后，令控制输入 EN_R 为 1，接收数据的准备就绪。令 EN_R 为 0 可以取消数据传输。被取消的请求用一个出错报文(STATUS 输出)结束。只要信号 EN_R 的状态为 0，接收操作就被闭锁。RD_1 为接收区(数据块编号和起始地址)，LEN 是数据块的长度。参数 R，LADDR，ERROR 和 STATUS 的意义与 SFB 60 的相同(见表 9-6)。

块被正确执行时 NDR 被置为 1 状态。如果请求因出错被关闭，ERROR 被置为 1 状态。

如果出现错误或报警，STATUS 将显示相应的事件标识符(ID)。在 SFB 的 R 输入为 1 时，也会输出 DONE 或 ERROR/STATUS(参数 LEN = 16#00)。如果出现一个错误，CPU 的二进制结果位 BR 将被复位。如果块无错误结束，BR 将被置位为 1。

数据的连续性被限制为 206 B，要传送的数据如果超过 206 B，在所有数据已全部接收完(NDR = TRUE)之前，不要访问接收数据块。然后闭锁 DB(令 EN_R = FALSE)，直到已经处理完该数据。

3. 用 SFB 62"RES_RCVB"清空接收缓冲区

SFB 62 用于清空 CPU 的整个接收缓冲区，所有保存的报文帧都被删除。在调用 SFB 62 时接收到的报文帧将被保存。块被调用后，在控制输入 REQ 的脉冲上升沿，工作过程被激活，请求可以在多个循环周期执行。各参数的意义见表 9-6。

表 9-6 SFB 60～SFB 62 的参数

参 数	声 明	数据类型	说 明
REQ	IN	BOOL	控制参数"请求":在信号的上升沿时激活操作
R	IN	BOOL	控制参数"复位":取消请求,中止操作
LADDR	IN	WORD	在"HW Config"中指定的子模块 I/O 地址
DONE	OUT	BOOL	状态参数(只在 1 次调用期间置位):为 0(FALSE)表示请求还没有起动或正在执行,为 1(TRUE)表示请求已经正确完成
ERROR	OUT	BOOL	状态参数(只在 1 次调用期间置位):请求完成,但是出错
STATUS	OUT	WORD	状态参数(0～FFFFH,只在 1 次调用期间置位,为了显示"STATUS",应将它复制到一个空的数据区)。根据"ERROR"位,STATUS 具有下列含义: ERROR = FALSE,STATUS=0000H:没有报警和错误 ERROR = FALSE,STATUS≠0000H:有报警,STATUS 提供详细的信息 ERROR = TRUE:有错误,STATUS 提供与错误类型有关的详细信息
SD_1	IN_OUT	ANY	发送参数:用于设置存放发送数据的 DB 编号和和起始数据字节编号。例如 DB10 的字节 2 表示为 DB10.DBB2
LEN	IN_OUT	INT	指定用于传输数据的数据块的字节长度(1 - 1024)
EN_R	IN	BOOL	控制参数"允许接收"
NDR	OUT	BOOL	状态参数"新的数据准备好":请求已经正确完成,接收到数据 FALSE 表示请求还没有启动或正在执行,TRUE 表示请求已经成功地完成
RD_1	IN_OUT	ANY	接收参数:设置存放接收到的数据的 DB 编号和和起始数据字节编号。例如 DB20 的字节 5 表示为 DB20.DBB5

9.2.2 用于 RK 512 协议的系统功能块

在 RK 512 协议通信中,SEND 请求把数据写入通信伙伴的某一存储区,FETCH 请求从通信伙伴某一存储区中读取数据。不能同时在用户程序中激活 SEND 请求和 FETCH 请求。

为了设置时的初始化和 SFB 之间的操作同步,所有 RK 512 通信的 SFB 都需要一个公共的数据块。用参数 SYNC_DB 设置该数据块的编号,用户程序中所有 RK 512 通信的 SFB 中,该数据块的编号必须相同,长度应不少于 240 B。

SFB 65"SERVE_RK"支持 SIMATIC S5 中的处理器通信标志位的功能,以便在 CPU 中能协调处理数据,以及在接收数据或提供数据时能异步改写。

1. 用 SFB 63"SEND_RK"发送数据

SFB 63 用来发送数据块中的数据。块被调用后,在控制输入 REQ 的脉冲上升沿激活数据传输。SD_1 为发送数据区(数据块编号和起始地址),LEN 是要发送的数据块的长度。

表 9-7 中是 SFB 63 和 SFB64 的部分参数,其余参数的意义与 SFB60 的相同(见表 7-6)。

当 DONE 为 TRUE,数据被发送到通信伙伴的 CPU 中,并返回一个肯定应答。

数据的连续性被限制为 128 B。如果发送的连续数据超过 128 B,必须注意下列问题:在传输过程结束之前,禁止写入当前正在使用的发送区 SD_1。状态参数 DONE 被置为 TRUE 时,将出现这种情况。

发送数据时应注意以下问题:

- RK 512 协议只允许发送偶数个数据。如果指定的数据长度(LEN)为奇数,则在传送的数据后面自动添加为 0 的填充字节。
- RK 512 协议的偏移量只能为偶数。如果指定了奇数偏移量,通信伙伴将使用相邻的较

小的偶数偏移量。例如偏移量设置为 7 时，数据将从字节 6 开始存放。

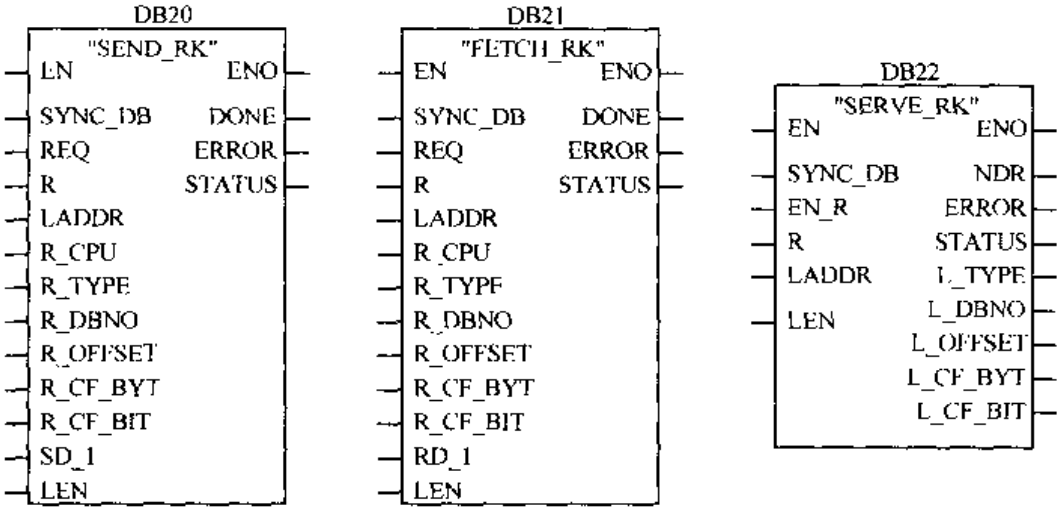


图 9-17　用于 RK 512 的系统功能块

表 9-7　SFB 63 和 SFB64 的部分参数

参　数	声　明	数据类型	说　明
SYNC_DB	IN	INT	储存 RK 512 同步的公共数据的数据块编号，由 CPU 指定，不能为 0
R_CPU	IN	INT	通信伙伴的 CPU 编号(0～4)，仅用于多 CPU 模式
R_TYPE	IN	CHAR	通信伙伴 CPU 的地址类型(只能用大写字母)，"D"为数据块，"X"为扩展的数据块
R_DBNO	IN	INT	通信伙伴 CPU 的数据块编号(0～255)
R_OFFSET	IN	INT	通信伙伴 CPU 数据块内的偏移量，即起始数据字节编号(0～510，只能为偶数)
R_CF_BYT	IN	INT	通信伙伴 CPU 的处理器通信标志字节(0～255，255 表示没有通信标志)
R_CF_BIT	IN	INT	通信伙伴 CPU 的处理器通信标志位(0～7)

表 9-8 给出了 RK 512 报文帧标题中的命令。

表 9-8　RK 512-报文帧标题中的命令

S7 自动化系统中的数据源(本地 CPU)	至目标(通信伙伴 CPU)	报文帧标题		
		字节 3,4:命令类型	字节 5,6:Z-DBNR/Z-Offset	字节 7,8:单位
数据块	数据块	A:发送,D:数据块	DB,DW	字
数据块	扩展数据块	A:发送,D:数据块	DB,DW	字

Z-DBNR 为目的数据块编号，Z-Offset 为目的起始地址(偏移量)，DW 是以字为单位的偏移量。

2．用 SFB 64"FETCH_RK"读取数据

SFB 64 用于从通信伙伴中读取指定区域中的数据，并将它们保存在自己的数据块中。

块被调用后，在控制输入 REQ 的脉冲上升沿激活发送过程。RD_1 为保存读取的数据的数据区(数据块编号和起始地址)，LEN 是要读取的数据块的长度。

使用 SFB 64 还可以指定通信伙伴存放要发送或读取的数据的数据区，该信息在报文帧的标题中由 CPU 输入，并传送给通信伙伴。

参数 REQ、R、LADDR、DONE、ERROR、STATUS、RD_1 和 LEN 的意义和取值范围与表 9-6 中的相同。

参数 SYNC_DB、R_CPU、R_TYPE、R_DBNO、R_OFFSET、R_CF_BYT 和 R_CF_BIT 的意义和取值范围见表 9-7。

参数 SYNC_DB 用来指定一个数据块，它用来保存起动初始化和同步程序的所有 SFB 的公共数据。在用户程序中，调用 SFB 63～SFB 65 时该数据块的编号应相同。

R_TYPE 为要读取的通信伙伴 CPU 中数据源的数据类型，其代码意义见表 9-9。

表 9-9　SFB 中通信伙伴 CPU 数据源的参数

通信伙伴 CPU 中的数据源	R_TYPE	R_DBNO	R_OFFSET(由通信伙伴的 CPU 设置)
数据块	'D'	0～255	0～510(只能为偶数)
扩展数据块	'X'	0～255	0～510(只能为偶数)
存储器标记	'M'	无关	0～255
输入	'E'	无关	0～255
输出	'A'	无关	0～255
计数器	'Z'	无关	0～255
定时器	'T'	无关	0～255

数据的连续性要求与 SFB 63 的相同，从数据块和扩展数据块读取数据时应注意的问题与 SFB 63 发送数据时的注意事项类似。在读取通信伙伴的定时器和计数器时，每个定时器和计数器占两个字节。报文帧标题中的命令如表 9-10 所示。

表 9-10　RK 512 报文帧标题中的命令

通信伙伴 CPU 的数据源	S7 系统中本地 CPU 的目标区	报文帧标题		
		字节 3,4:命令类型	字节 5,6:Q-DBNR/Q-Offset	字节 7,8:单位
数据块	数据块	E,D	B/DW	字
扩展数据块	数据块	E,X	DB/DW	字
存储器标志	数据块	E,M	字节地址	字节
输入	数据块	E,E	字节地址	字节
输出	数据块	E,A	字节地址	字节
计数器	数据块	E,Z	计数器号	字
定时器	数据块	E,T	定时器号	字

字节 3 中的 E 表示 FETCH，Q-DBNR 为源数据块编号，Q-Offset 为源起始地址(偏移量)。

3. 用 SFB 65"SERVE_RK"接收或提供数据

(1) 接收数据：数据被保存在由通信伙伴在 RK 512 报文帧标题中指定的数据区。如果通

信伙伴执行一个“发送数据”请求(SEND 请求),需要调用 SFB 65。

(2) 提供数据:从通信伙伴在 RK 512 报文帧标题中指定的数据区中读取数据。如果通信伙伴执行一个“读取数据”请求(FETCH 请求),需要调用 SFB 65。

功能块被调用后,在控制输入 EN _ R 为 TURE 时,功能块准备好接收数据。令 EN _ R 的信号状态为 FALSE,可以取消当前的数据传输。被取消的请求用一个出错报文(STATUS 输出)结束。只要参数 EN _ R(允许接收)提供的信号状态为 FALSE,就锁定接收操作。SFB 65 的部分参数如表 9-11 所示。

表 9-11　SFB 65 的部分参数

参　数	声　明	数据类型	说　明
SYNC _ DB	IN	INT	储存 RK 512 同步的公共数据的数据块编号,由 CPU 指定,不能为 0,最少 240 字节
EN _ R	IN	BOOL	控制参数“允许接收”:请求的使能信号
L _ TYPE	IN	CHAR	本地 CPU 的源或目的数据区类型(只能用大写字母):“D”为数据块,“M”为存储器位,“E”为输入,“A”为输出,“Z”为计数器,“T”为定时器
L _ DBNO	IN	INT	本地 CPU 的数据块编号(由 CPU 指定)
L _ OFFSET	IN	INT	本地 CPU 的数据字节号(0～510,只能为偶数)
L _ CF _ BYT	IN	INT	本地 CPU 的处理器通信标志字节(0～255,255 表示没有通信标志)
L _ CF _ BIT	IN	INT	本地 CPU 的处理器通信标志位(0～7)

在用户程序中,调用 SFB 63～SFB 65 时用 SYNC _ DB 指定的数据块的编号应相同。

参数 R、LADDR、NDR、DONE、ERROR、STATUS 的意义与表 9-6 中的相同。

若 SFB 被调用且 NDR 为 1,用参数 L _ TYPE、L _ DBNO 和 L _ OFFSET 指明保存或读取数据的区域。相关的参数还有 L _ CF _ BYT、L _ CF _ BIT 和长度 LEN。

SFB 65 需要背景数据块,DB 的块号通过调用传递,不能直接访问背景数据块。

4. 处理器通信标志位的应用

可以通过处理器通信标志位允许或封锁通信伙伴的 SEND 和 FETCH 操作,以防止改写或读取还没有处理的数据。可以为每个请求指定处理器通信标志位。

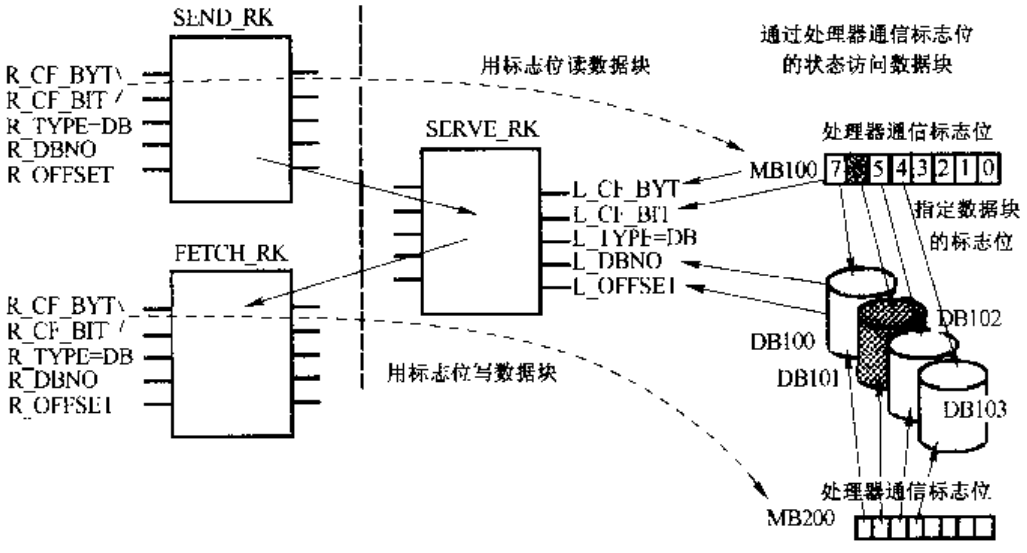


图 9-18　RK 512 通信示意图

【例 9-1】 通信伙伴用 SFB 63“SEND _ RK”将数据发送到本地 CPU 的 DB101。

(1) 在本地 CPU 中,将处理器通信标志位 100.6 复位为 0。

(2) 通信伙伴用 SEND 请求中的参数 R _ CF _ BYT 和 R _ CF _ BIT,指定本地 CPU 的处理器通信标志位为 100.6。RK 512 报文帧标题中的处理器通信标志位将传送到本地 CPU 中。

在处理请求之前,本地 CPU 检查在 RK 512 报文帧标题中指定的处理器通信标志位。只有在本地 CPU 上的处理器通信标志状态为 0,才能处理请求。如果其状态为 1,将在响应报文帧中向通信伙伴返回一个出错信息。

数据被传送到 DB101 后,SFB 65“SERVE _ RK”将本地 CPU 上的处理器通信标志位 100.6 置为 1。如果 NDR 为 1,在 SFB 65 上将输出持续一次调用时间的标志字节和标志位。

(3) 在本地用户程序中检查处理器通信标志位 100.6 是否为 1,可以确定请求是否完成,要处理的数据是否已经准备就绪。

(4) 在本地用户程序中处理完数据后,必须将处理器通信标志位 100.6 复位为 0,使本地通信伙伴可以再次无错误地执行请求。

传送的连续数据被限定为 128 B。如果超过 128 B,应注意以下问题:

使用处理器通信标志功能,在所有的数据被传输完之前,不要访问数据。检查为该请求指定的处理器通信标志位,如果 NDR 为 1,在调用 SFB 时处理器通信标志位被激活。在数据处理完之前,不要将处理器通信标志状态复位为 0。

9.3 用于点对点通信处理器的功能块

9.3.1 点对点通信软件包的下载与安装

S7-300/400 的点对点通信处理器(CP)包括 CP 340、CP 341、CP 440 和 CP 441。

在安装好 STEP 7 的计算机中,执行本书配套的光盘中的 Setup _ PtP.EXE,解压后在 SETUP _ PtP 文件夹中有 CP 的用户手册和待安装的软件。安装好软件后,在 STEP 7 中将会增加点对点通信处理器的组态信息和表 9-12 所示的功能块。它们在程序编辑器指令列表的“/Libraries/CP PtP”文件夹中。同时还会自动安装使用点对点通信处理器的例程“zXX21 _ 01 _ PtP _ Com _ CP34x”、“zXX21 02 _ PtP _ Com _ CP440”和“zXX21 _ 03 _ PtP _ Com _ CP441”。

表 9-12 点对点通信处理器的通信功能块

FB/FC	意 义	协 议
FB 2“P _ RCV”	接收通信伙伴的数据,并将它存储在数据块中	ASCII 和 3964(R)
FB 3“P _ SEND”	将数据块中的全部或部分数据发送给通信伙伴	ASCII 和 3964(R)
FB 4“P _ PRINT”	将包含最多 4 个变量的报文文本输出给打印机	打印机驱动器
FC 5“V24 _ STAT”	读取 CP 341 RS-232C 模块的 RS-232C 接口的信号状态	ASCII
FC 6“V24 _ SET”	置位/复位 CP 341 RS-232C 模块的 RS-232C 接口的输出	ASCII
FB 7“P _ RCV _ RK”	接收通信伙伴的数据,存储在数据块中,或准备传输给通信伙伴的数据	ASCII,3964(R),RK 512
FB 8“P _ SND _ RK”	将数据块中的全部或部分数据发送给通信伙伴或从通信伙伴读取数据	ASCII,3964(R),RK 512

点对点通信功能块是CPU模块与点对点通信处理器的软件接口，用于建立和控制CPU和CP之间的数据交换。完成一次发送需要多个循环周期，在用户程序中它们必须被无条件连续调用，用于周期性的或定时程序控制的数据传输。

9.3.2 CP 340的发送功能块与接收功能块

1. CP 340的发送功能块FB 3"P_SEND"

FB 2～4只能用于CP 340，CP 340还可以使用FC5和FC6。发送功能块FB 3"P_SEND"将数据写入CP 340的发送缓冲区，再由后者发送给通信伙伴。FB 3需要大小为40 B的背景数据块，用户不能访问该背景数据块中的数据。

FB 2～FB 4应在循环程序或时间控制的程序中无条件地调用。它将数据块中连续的数据传送给CP 340，在功能块的参数中需要设置存放要发送的数据的数据块编号DB_NO、数据块中的起始字节地址DBB_NO和数据的长度LEN。

在输入信号REQ的上升沿，FB 3"P_SEND"进入发送状态，开始向CP 340传送数据，然后CP 340将数据发送给接收方。数据传送过程可能需要若干个CPU的循环周期。在发送期间，REQ不必始终为1。

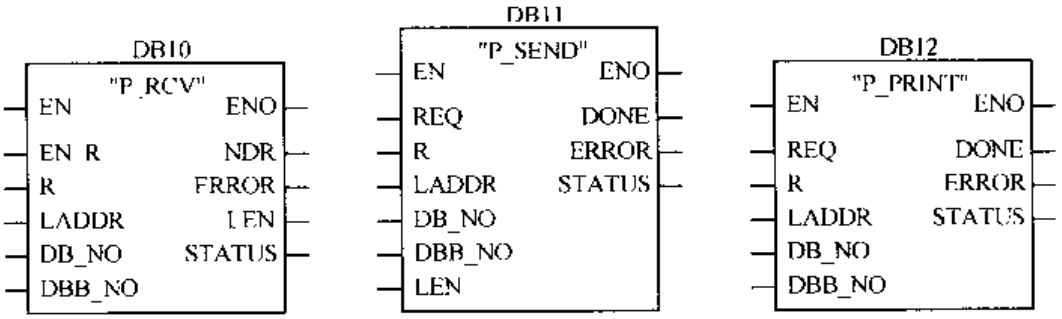


图9-19 用于CP 340的通信功能块

发送过程结束后，如果正确完成了发送任务，DONE为1，输出STATUS的值为0。否则输出ERROR为1，STATUS中为事件号。

在发送期间，如果输入R变为1，将中止发送，并将FB 3"P_SEND"复位（置为初始状态），但是已经传送到CP 340的数据将继续发送。

如果出现错误，CPU的二进制结果位BR将被复位。如果块无错误结束，BR将被置为1状态。

表9-13 FB 2"P_RCV"和FB 3"P_SEND"的参数

参　数	声　明	数据类型	说　明
REQ	IN	BOOL	用上升沿初始化请求
R	IN	BOOL	取消请求，中止操作
LADDR	IN	INT	STEP 7中CP模块的基地址
DB_NO	IN	INT	数据块编号，由CPU指定，不能为0
DBB_NO	IN	INT	数据字节编号（0～8190），以数据字的方式传输数据
LEN	IN	INT	指定用于传输数据的数据块的字节长度（1～1024）
DONE	OUT	BOOL	请求被正确完成，状态参数STATUS为0，只保持一个CPU循环

（续）

参　数	声　明	数据类型	说　明
ERROR	OUT	BOOL	请求完成，有错误
STATUS	OUT	WORD	如果 ERROR 为 1，STATUS 提供详细的错误信息
EN_R	IN	BOOL	允许读数据
NDR	OUT	BOOL	请求已经正确完成，接收到数据，参数 STATUS 为 0

2. CP 340 的接收功能块 FB 2“P_RCV”

FB 2“P_RCV”将 CP 340 的接收缓冲区中的数据存放在数据块中，数据块的编号为 DB_NO，数据块中的起始字节的地址为 DBB_NO，数据的长度为 LEN。FB 2“P_RCV”也需要大小为 40 B 的背景数据块。

如果输入信号 EN_R 为 1 状态，软件查询是否可以从 CP 340 读取数据。EN_R 如果为 0 状态，将中止正在进行的接收工作，且 ERROR 变为 1 状态，并由 STATUS 给出错误信息。数据传输过程可能需要几个循环周期。

如果输入 R 变为 1 状态，FB 2“P_RCV”将被复位（置为初始状态），接收请求被中止。若 R 又变为 0 状态，重新开始接收被中止的报文帧。

如果接收缓冲区内有数据，就转入接收状态。读出 CP 340 的接收缓冲区后，FB 2“P_RCV”从接收状态转入查询状态。

如果数据接收请求正确地完成，已经接收完数据，FB 2 的输出信号 NDR 为 1 状态。如果出现错误，输出 ERROR 为 1 状态，LEN 的值为 0，STATUS 中为错误代码。没有错误时 STATUS 的值为 0。FB 2 被复位时（参数 LEN 为 0），NDR 和 ERROR/STATUS 也有输出。如果出现错误，CPU 的二进制结果位 BR 将被复位。如果块无错误结束，BR 将被置为 1 状态。

9.3.3　向打印机输出报文文本的功能块

CP 340 的打印机驱动器（Printer Driver）通信协议用于向打印机输出带日期和时间的报文文本（Message Texts）。这一功能可以用于监视简单的过程，例如打印出错误和故障信息，或者向操作人员发布命令。

FB 4“P_PRINT”功能块用于将最多包含 4 个变量的报文文本发送给 CP 340，CP 340 在打印机上记录过程信息。FB 4 的背景数据块占用 39 B。

可以用变量和控制语句（例如加粗、斜体和下划线等）来组态报文文本。组态时每一报文文本被指定一个编号。在 FB 4 的格式字符串中指定报文文本编号，对应的报文文本将被打印出来。应在数据块的标题中存储格式字符串和变量。

在硬件组态（HW Cofig）窗口中双击 CP 340 所在的行，在出现的属性对话框的“General”选项卡中点击【Parameter】按钮，将出现点对点连接的参数设置对话框。在对话框的上部可以选择通信协议（Protocol），如果选择“PRINTER（打印机）”，窗口中出现的画面如图 9-20 所示。

在该图中双击“Protocol”框可以设置打印机协议的通信参数（见图 9-21）。

双击图 9-20 中的“Message”框，可以设置报文；双击“Page Layout”框，可以设置打印的页面参数，例如页眉、页脚和页边距等；双击图 9-20 中的“Character Set”框，可以修改打印机的字符集。STEP 7 用字符转换表将 ANSI 字符集转换为打印机的字符集，例如为了获得在某种语言中需要的特殊字符。

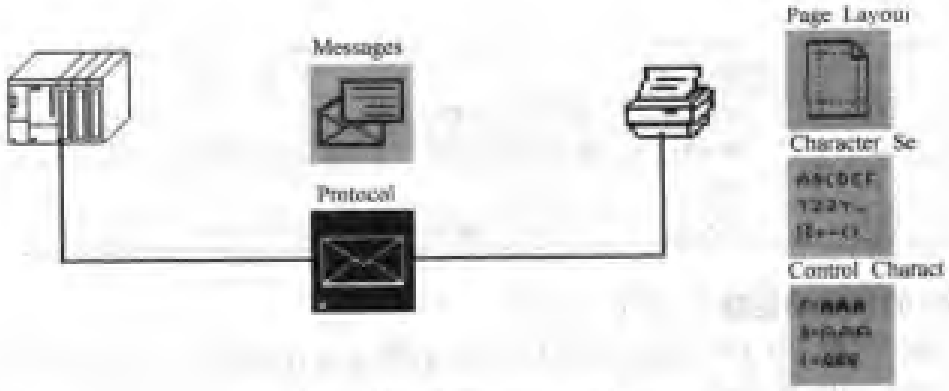


图 9-20　打印机通信协议的组态画面

图 9-21　打印机驱动器通信协议的参数设置

双击图 9-20 中的"Control Characters"(控制字符)框,在控制字符表中可以修改打印的报文文本中的控制语句,例如起动或停止加粗、放大、缩小、斜体、下划线等。

在一个报文文本中最多出现 4 个变量(3+1 个报文文本编号),变量的值从 CPU 传送到 CP 340。可以打印出用户程序计算出的变量的值(例如液位)、日期和时间、字符串(字符串变量)或其他信息。

在组态报文文本时,或在每个变量的格式字符串中,应给出转换语句,变量值的意义和输出格式在语句中编码。

指向用于格式字符串和 4 个变量的数据块的指针用参数 DB_NO(数据块编号)和 DBB-NO(数据块中的数据字节)来设置。指针应无间隙和按指定的顺序保存在指针数据块中(见图 9-22)。如果变量的指针中的数据块号被设为 00H,表示变量不存在。如果格式字符串的 DB 号为 00H,打印请求被中止。

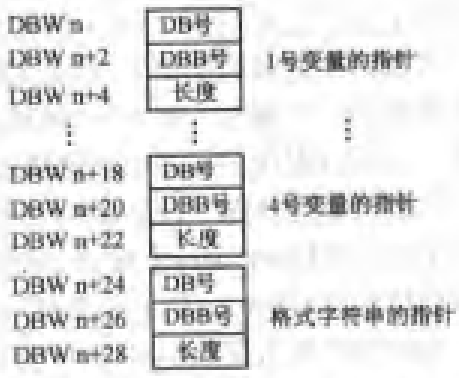


图 9-22　FB P_PRINT 的指针 DB 的结构

变量最长占 32 B,格式字符串最多 150 B。超过最大长度时打印请求将被中止。

在 REQ 输入的上升沿,报文文本的发送被初始化,首先发送报文文本的格式字符串,然

后是1～4号变量。数据的传输可能占用几个循环周期。

如果输入R为1状态,FB 4"P_PRINT"将被置为初始状态,CP 340的数据传输被中止。若R重新变为0状态,从重头开始接收被中止的报文帧。已经被CP 340接收的数据将发送到打印机。如果R输入的信号状态一直为1状态,打印请求的传输被禁止。LADDR、DONE、ERROR、STATUS的意义与FB 2的相同。

在点对点通信处理器的用户手册中,可以找到打印机驱动器的参数设置和用于打印输出的转换与控制语句的详细信息。

9.3.4 读取和控制RS-232C接口的信号状态的功能块

1. CP的RS-232C接口的辅助信号

- DCD(Data carrier detect,输入):数据载波检测;
- DTR(Data terminal ready,输出):数据终端(CP 340)准备好操作;
- DSR(Data set ready,输入):数据设备(通信伙伴)准备好操作;
- RTS(Request to send,输出):CP 340已准备好,请求发送;
- CTS(Clear to send,输入):清除发送,通信伙伴可以从CP 340接收数据,CP 340等待对RTS为1状态的响应。
- RI(Ring Indicator,输入):振铃指示,指示一个输出调用。

可以使用功能FC 5"V24_STAT"和FC 6"V24_SET",通过CP 340对与控制信号DTR/DSR和RTS/CTS有关的参数进行设置。

在以下情况使用RS-232C接口的辅助信号:

(1) 设置为自动使用RS-232C接口的辅助信号,此时不能用RTS/CTS数据流控制,也不能用V24_SET功能进行RTS和DTR控制。

(2) 设置为数据流(RTS/CTS)控制,此时不能用V24_SET功能进行RTS控制。

(3) 在任何情况下都可以用"V24_STAT"功能读取所有的RS-232C接口的辅助信号。

2. 自动使用辅助信号

可以用下述方式实现自动使用RS-232C辅助信号:

(1) 一只CP 340通过参数设置切换到使用RS-232C辅助信号的运行模式,它将RTS线置为OFF(0状态),将DTR置为ON(1状态),CP 340准备好使用。

(2) 在DTR线被置为ON之前,不能发送和接收报文帧。DTR为OFF时,不能通过RS-232C接口接收数据。如果有发送请求,它将被中止,并出现错误信息。

(3) 有发送请求时,RTS变为ON,在设置的数据输出等待时间到达时,如果CTS已变为ON,数据通过RS-232C接口发送(见图9-23)。

(4) 在数据输出等待时间内,如果CTS没有变为ON,或者在传输期间CTS变为OFF,发送请求被中止,同时生成错误信息。

(5) 数据发送完成后,经过设置的时间,RTS线被置为OFF。CP 340并不等待CTS变为OFF。

(6) 只要DSR线为ON,RS-232C接口就可以接收数据。如果CP 340的接收缓冲区有可能溢出,CP 340不会响应。

(7) 如果DSR由ON变为OFF,发送请求或接收将被中止,并产生一个错误信息。在CP

340 的诊断缓冲区将会出现信息“DSR = OFF”。

3. FC 5“V24 _ STAT”和 FC 6“V24 _ SET”

FC 5“V24 _ STAT”用来检查接口的状态，从 CP 中读取 RS-232C 接口的辅助信号，将它们放在块的参数中，供用户使用。功能每次被调用时(循环轮询)RS-232C 的辅助信号被刷新。CP 340 每 20 ms 刷新一次输入/输出的状态。

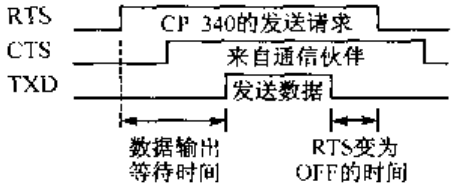


图 9-23 RS-232C 辅助信号时序图

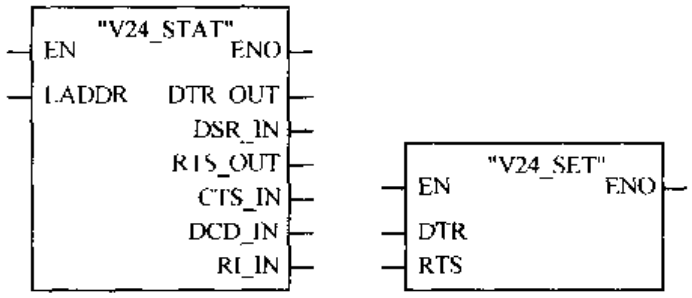


图 9-24 读取和控制 RS-232C 接口的信号状态的功能块

FC 6“V24 _ SET”用来置位/复位 CP 的 RS-232C 接口的输出信号。

表 9-14 给出了 FC 5 和 FC 6 的参数的意义。执行这两个 FC 时二进制结果 BR 位不会受到影响，功能不会产生错误信息。“V24 _ STAT”不占用数据区。

表 9-14 FC5 和 FC6 的参数

参 数	声 明	数据类型	说 明
LADDR	IN	INT	STEP 7 中 CP 模块的基地址
DTR _ OUT	OUT	BOOL	数据终端准备好，CP 准备操作，CP 的输出
DSR _ IN	OUT	BOOL	数据设备准备好，通信伙伴准备操作，CP 的输入
RTS _ OUT	OUT	BOOL	CP 准备好，请求发送，CP 的输出
CTS _ IN	OUT	BOOL	清除发送，通信伙伴可以从 CP 接收数据(响应 CP 的 RTS 输出)，CP 的输入
DCD _ IN	OUT	BOOL	数据载波检测，接收信号电平，CP 的输入
RI _ IN	OUT	BOOL	振铃指示器，轮询信号，CP 的输入
RTS	IN	BOOL	CP 准备好，请求发送，控制 CP 的输出
DTR	IN	BOOL	数据终端准备好，CP 请求操作，控制 CP 的输出

9.3.5 用于 CP 341 的通信功能块

CP 341 用功能块 FB 7“P _ RCV _ RK”接收数据，用 FB 8“P _ SND _ RK”发送数据，可以使用的通信协议有 ASCII 驱动器，3964(R) 程序和 RK 512 计算机连接协议。CP 341 还可以使用功能 FC 5“V24 _ STAT”和 FC 6“V24 _ SET”。

在用户程序中，FB 7 和 FB 8 只能分别使用一次，它们分别只能有一个背景数据块。

FB 7 和 FB 8 不作参数检查，如果参数设置出错，CPU 将进入 STOP 模式。

1. 数据的连续性限制

因为数据的连续性的限制，CPU 和 CP 341 之间最多只能用块传输 32 B 的连续数据。满足下述要求时允许超过 32 B：

(1) 使用 3964(R)协议时，在全部数据已发送完(DONE = 1)之前，发送方不访问发送 DB。

使用 RK 512 协议时，在数据传输结束(DONE = 1)之前，发送方不访问发送 DB。

(2) 使用 3964(R)协议时接收方不访问接收 DB，除非已经接收到全部数据(DNR = 1)。接收完成后，EN _ R 为"0"，关闭接收 DB(EN _ R = 0)，直到数据被处理完。

(3) 用 RK 512 协议接收数据或准备数据时使用了处理器标志位，接收方不访问接收 DB，除非已经接收到全部数据，在 NDR = 1 时处理器通信标志位被置位一个循环周期。在处理完接收的数据之前不要将通信标志位复位。

(4) 使用 RK 512 协议准备数据时，准备方不访问准备的数据，除非全部数据被通信伙伴取走，在 NDR = 1 时处理器通信标志位被置位一个循环周期。在被取走的数据被处理完之前不要将通信标志位复位。

表 9-15 FB 7"P _ RCV _ RK"和 FB 8"P _ SEND _ RK"的参数

参 数	声 明	数据类型	说 明
SF	IN	CHAR	为'S'时发送数据，为'F'时读取数据
REQ	IN	BOOL	在上升沿初始化请求
R	IN	BOOL	取消请求，中止操作
LADDR	IN	INT	STEP 7 中 CP 模块的基地址
DB _ NO	IN	INT	数据块编号，由 CPU 指定，不能为 0
DBB _ NO	IN	INT	数据字节编号(0～8190)，以数据字的方式传输数据
LEN	IN	INT	要发送的帧的字节长度(1～1024)，应为偶数
DONE	OUT	BOOL	请求被正确完成，参数 STATUS 为 0，只保持一个 CPU 循环
ERROR	OUT	BOOL	请求完成，有错误
STATUS	OUT	WORD	如果 ERROR 为 1，STATUS 提供详细的错误信息
R _ CPU _ NO	IN	INT	通信伙伴的 CPU 编号(0～4)，仅用于多 CPU 模式，默认值为 1
R _ TYP	IN	CHAR	通信伙伴 CPU 的地址类型，"D"为数据块，"X"为扩展的数据块
R _ NO	IN	INT	通信伙伴 CPU 的数据块编号(0～255)
R _ OFFSET	IN	INT	通信伙伴 CPU 的数据字节号(0～510，只能为偶数)
R _ CF _ BYT	IN	INT	通信伙伴 CPU 的处理器通信标志字节(0～255，默认值 255 表示没有通信标志)
R _ CF _ BIT	IN	INT	通信伙伴 CPU 的处理器通信标志位(0～7)
EN _ R	IN	BOOL	允许接收数据
NDR	OUT	BOOL	请求已经正确完成，接收到数据，参数 STATUS 为 0
L _ TYP	OUT	CHAR	本地 CPU(目的)地址类型，"D"为数据块
L _ NO	OUT	INT	本地 CPU(目的)数据块编号(0～255)
L _ OFFSET	OUT	INT	本地 CPU(目的)数据字节号(0～510，只能为偶数)
L _ CF _ BYT	OUT	INT	本地 CPU 处理器通信标志字节(0～255，默认值 255 表示没有通信标志)
L _ CF _ BIT	OUT	INT	本地 CPU 处理器通信标志位(0～7)

2. FB 7 和 FB 8 的参数

3964(R) 程序不使用参数 R _ CPU _ NO、R _ TYP、R _ NO、R _ OFFSET、R _ CF _ BYT 与 R _ CF _ BIT，可以忽略它们。

3. 用 FB 8 和 3964(R)，ASCII 协议发送数据

FB 8"P _ SEND _ RK"将数据块中的数据传送给 CP 341，在功能块的参数中需要设置存放要发送的数据的数据块编号 DB _ NO、数据块中的起始字节的地址 DBB _ NO 和数据的长度 LEN。FB 7 和 FB 8 应是无条件调用的，数据的发送或接收可以是循环的，也可以是用时间驱动的。FB 8 需要一个 62 B 的背景数据块。

在输入信号 REQ 的上升沿，数据发送操作被初始化。与发送的数据量有关，发送过程可能需要几个循环周期。

在发送期间，如果输入 R 为 1，将中止发送，并将 FB 8"P _ SEND _ RK"置为初始状态。但是已经传入 CP 341 的数据将继续发送给通信伙伴。如果 R 一直为 1，不能激活发送。

FB 执行出错时二进制结果位 BR 被复位，如果 FB 被正确执行，BR 为 1。

LADDR 用来设置 CP 341 的起始地址。发送过程结束后，如果正确完成发送任务，DONE 为 1，输出 STATUS 的值为 0。否则输出 ERROR 为 1，STATUS 中为事件号。

4. 用 FB 7 和 3964(R)、ASCII 协议接收数据

FB 7"P _ RCV _ RK"将 CP 341 的接收缓冲区中的数据连续存放在指定的数据区中，数据区用参数 DB _ NO、DBB _ NO 和 LEN 来指定。FB7 需要一个 60 B 的背景数据块。

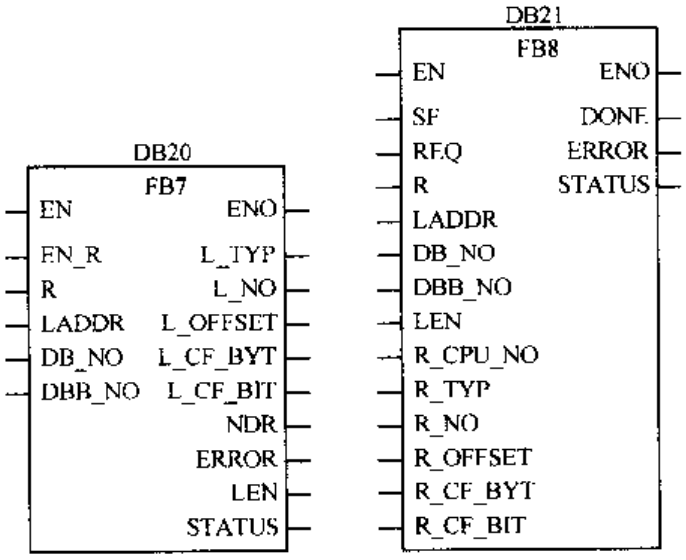


图 9-25　用于 CP 341 的通信功能块

如果输入信号 EN _ R 为 1，软件查询是否可以从 CP 341 的接收缓冲区读取数据。EN _ R 如果为 0，将中止正在进行的接收工作，且 ERROR 变为 1，并由 STATUS 给出错误信息。数据传输过程可能需要几个循环周期。

如果输入 R 为 1，FB 7"P _ RCV _ RK"将被复位(置为初始状态)，接收请求被中止。若 R 重新变为 0，从头开始接收被中止的报文帧。

如果数据接收请求被正确地完成，FB 7"P _ RCV _ RK"的输出信号 NDR 为 1。如果没有

错误,输出 STATUS 的值为 0。否则输出 ERROR 为 1,STATUS 中为错误代码。

5. 用 FB 8 和 RK 512 协议发送和读取数据(主动请求)

FB 8"P _ SEND _ RK"用于 RK 512 协议时,相当于 CPU 31xC 的系统功能块 SFB 63 "SEND _ RK"和 SFB 64"FETCH _ RK",参数 SF = 'S '时相当于 SFB 63,将 S7 CPU 中的数据区发送给 CP 341。SF = 'F '相当于 SFB 64,从远方通信伙伴处读取数据,并将它们保存在本地 CPU 的数据区中。

FB 8"P _ SEND _ RK"中的 DB _ NO 和 DBB _ NO 的作用与 SFB 63"SEND _ RK"中的 SD _ 1 和 SFB 64"FETCH _ RK"中的 RD _ 1 的相同。

用 FB 8 和 RK 512 协议发送和读取数据时,具体的过程和功能块的参数设置可以参考 SFB 63 和 SFB 64 的有关部分。

6. 用 FB 7 和 RK 512 协议接收数据(被动请求)

FB 7"P _ RCV _ RK"通过被动请求的方式,将通信伙伴发送到 CP 341 的接收缓冲区中的数据存放在指定的数据区中,通信伙伴是主动的。EN _ R,LADDR,R,NDR,ERROR 和 STATUS 的意义与使用 3964(R)协议时相同。

如果通信伙伴指定了目的"DB",接收到的数据将存放在通信伙伴在 RK 512 报文帧的标题中指定的目的数据区。数据被存放时,本地 CPU 中目的数据区的类型、编号、偏移量和长度(L _ TYP、L _ NO、L _ OFFSET 和 LEN)出现 1 个循环周期。

如果通信伙伴指定了目的地址"DX",接收到的数据将存放在参数 DB _ NO 和 DBB _ NO 指定的数据块中。

在接收数据之前,检查在 RK 512 报文帧标题中指定的处理器通信标志位。只有该标志为 0 状态,才传输数据。传输结束后,功能块将该标志位置为 1 状态,并使输出 NDR 在一个循环周期内为 1 状态。

在本地用户程序中检查处理器通信标志位,以确定是否可以处理数据传输。一旦数据被处理完,用户必须将处理器通信标志位复位为 0 状态,下一次发送请求可以被通信伙伴起动。

7. 用 FB 7"P _ RCV _ RK"准备数据(被动请求)

如果通信伙伴执行了 FETCH 请求,则没有必要调用 FB 7。FB 7 在 CPU 的数据区内准备好送给 CP 341 的要发送的数据。

如果输入信号 EN _ R 为 1 状态,软件查询是否已为 CP 341 准备好数据。EN _ R 如果为 0 状态,将中止正在进行的主动传输,数据传输过程可能需要几个循环周期。

L _ TYP、L _ NO、L _ OFFSET 和 LEN 分别是在本地 CPU 中准备的源数据块的类型、编号、偏移量和长度,它们由第一个 RK 512 报文帧决定。

功能块从该报文帧分析信息,将请求的数据传送给 CP 341。参数 DB _ NO 和 DBB _ NO 对于 FB 7 没有意义。

LADDR、R、NDR 的意义与使用 3964(R)协议接收数据时相同。源数据被取走时,L _ TYP、L _ NO、L _ OFFSET、L _ CF _ BYT 和 L _ CF _ BIT 出现 1 个循环周期。

在接收数据帧之后,检查在 RK 512 报文帧标题中的处理器通信标志位。该标志位的状态为 0,表示数据已经准备好。传输结束后,功能块将处理器通信标志位置为 1,并使输出 NDR 在一个循环周期内为 1 状态。

在本地用户程序中检查处理器通信标志位,确定要传输的数据是否可以访问。一旦数据

被处理，用户必须将处理器通信标志位复位为 0 状态，为通信伙伴下一次读取数据的请求做好准备。

9.3.6 用于 CP 440 和 CP441 的通信功能块

S7-400 的点对点通信处理器的通信功能块如表 9-16 所示。详细的功能和使用方法参见 CP 440 和 CP 441 的用户手册。

表 9-16 S7-400 的点对点通信处理器的通信功能块

FB	意　义	协　议	CP
FB 9"RECV_440"	接收通信伙伴的数据，并将它存储在数据块中	ASCII driver 3964(R)	CP 440
FB 10"SEND_440"	将数据块中的全部或部分数据发送给通信伙伴	ASCII driver 3964(R)	CP 440
FB 11"RES_RECV"	复位 CP 440 的接收缓冲区	ASCII driver 3964(R)	CP 440
SFB_12"BSEND"	从 S7 数据区将数据发送到固定的通信伙伴目的区	ASCII driver 3964(R)	CP 441
SFB_13"BRCV"	从通信伙伴接收数据，并发送到 S7 数据区	ASCII driver 3964(R)	CP 441
SFB_14"GET"	从通信伙伴读取数据	RK 512	CP 441
SFB_15"PUT"	用动态可变的目的区将数据发送到通信伙伴	RK 512	CP 441
SFB_16"PRINT"	将最多包含 4 个变量的报文文本输出到打印机	PRINT Driver	CP 441
SFB_22"STATUS"	查询通信伙伴的设备状态		CP 441

9.4 Prodave 通信软件在点对点通信中的应用

9.4.1 PRODAVE 简介

PLC 具有极高的可靠性，一般用来执行现场的控制任务，但是它的人机接口功能较差。PLC 与个人计算机(PC)通过通信连接起来，用 PC 作为上位计算机，实现系统的监控、人机接口和与上一级网络(例如工业以太网)的通信等功能，可以使二者的优势互补，组成一个功能强、可靠性高、成本低的控制系统。因此在工业控制系统中，PC 与 PLC 之间的通信是最常见的和最重要的通信之一。

S7-300/400 的 MPI(多点通信接口)和 S7-200 的 PPI(点对点通信接口)用于西门子公司控制产品之间的通信，例如安装在 PC 上的 STEP 7 编程软件与 PLC 之间的通信，但是这些通信协议均未公开。如果用 S7-300/400 的点对点通信协议实现 PC 与 PLC 之间的通信，需要配置专用的通信处理器模块或带点对点通信接口的 CPU 31x-2PtP。其点对点通信协议并不通用，要花较多的时间熟悉它，才能编写出 PLC 和计算机的通信程序。

PRODAVE 是用于 PC 与 S7 系列 PLC 之间的数据链接通信的工具箱。PRODAVE 是"Process Data Traffic"(过程数据交换)的缩写，可以用于 S7-200、S7-300/400、M7 和 C7 等西门子的 S7 系列 PLC。通过下列硬件，可以方便地在 PLC 与 PC 之间建立数据链接：

- 用于 PC 的 MPI 通信处理器，例如 CP5511、CP5611 和 CP5612，通信速率可达 12 Mbit/s；
- 用于 S7-300/400 的 PC/MPI 适配器(PC-Adaptor)；

● 用于 S7-200 的 PC/PPI 编程电缆。

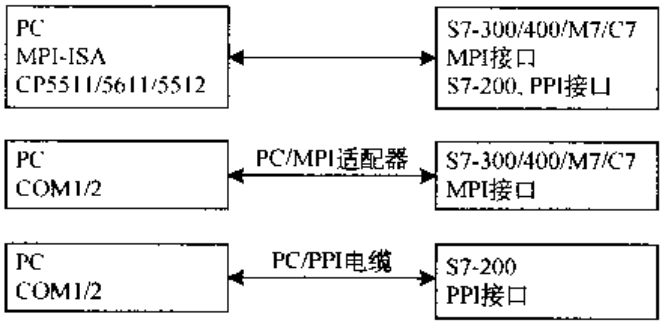


图 9-26　PC 与 PLC 的连接方式

PRODAVE 的动态链接库(DLL)提供了大量的基于 Windows 操作系统的 DDL 函数，供用户解决 PLC 与 PC 之间的数据交换和数据处理问题。可以在 VB 或 VC 等编程环境中调用这些函数。PRODAVE 有以下特点：

(1) 使用简单方便，编程人员不需要熟悉复杂的通信协议，通过调用 PRODAVE 提供的动态链接库(DLL)中的函数就可以实现通信。

(2) 上位机用通信函数直接读写 PLC 中的数据，用户不用编写 PLC 一侧的通信程序。

(3) 如果使用 PC/MPI 适配器或用于 PC 的通信处理器作通信接口，它们同时还可以兼作编程软件与 PLC 的通信接口。

Visual Basic(VB)易学易用，是开发小型 Windows 应用程序的首选。PRODAVE 的用户手册主要是针对 VC 的，对 VB 环境下的应用介绍得很少，用户要花大量的精力去摸索如何在 VB 环境使用 PRODAVE。本节主要介绍在 VB 环境下调用 PRODAVE 的函数的方法，如果在 VC 环境下编程，可以参考 PRODAVE 的用户手册，基本方法没有什么区别。

PRODAVE 的函数分为基本函数、数据处理函数和电话服务函数(TeleService Functions)。

基本函数用于建立、断开和激活 PC 与 PLC 的连接，以及读、写 PLC 中的各种数据。数据处理函数用于 PC 中用户数据的转换和处理，与 PC 和 PLC 之间的通信没有直接的关系。电话服务函数用于 PC 通过电话线与 PLC 建立连接，本节对此不作介绍。

9.4.2　PRODAVE 的硬件配置

用户在安装了 PRODAVE 软件包后，需要设置 PG-PC(PG 是编程器的缩写)的接口参数，对硬件进行配置，假设使用 PC/MPI 适配器(PC Adaptor)，配置的步骤如下：

(1) 在桌面执行菜单命令“开始”→“程序”→“PRODAVE _ S7”→“PG-PC Interface”，打开接口参数设置对话框(见图 9-27)。

(2) 在“Interface Parameter Assignment”(接口参数配置)列表框中如果没有实际实用的硬件接口，点击右下方的【Select】按钮，打开“Installing/Uninstalling Interfaces”对话框，安装实际使用的硬件接口的驱动程序。

(3) 在图 9-27 中选中接口参数配置列表框中的“PC Adapter(MPI)”，在上面的“Access Point of the Application”(应用程序访问点)列表框内选择“S7ONLINE(STEP 7)”。

点击【Properties...】(属性)按钮，打开属性对话框。将“MPI”栏中的“Transmission Rate”(传输速率)设置为 187.5 kbit/s，其他参数可以采用默认的设置。在“Local Connection”选项卡

的"COM Port"选择框中设置实际使用的 PC 串口的编号，传输速率可以设置为 19.2 kbit/s。

(4) 点击【OK】按钮，完成设置。

9.4.3 建立与断开连接

通信之前，首先要用 load _ tool 函数建立上位机与 PLC 的连接，通信结束时必须用 unload _ tool 函数断开 PC 与 PLC 的连接，否则可能引起上位机死机，或者造成上位机系统的异常状况。

图 9-27 PG/PC 接口设置

VB 在调用 DLL 函数之前，需要用 Declare 语句声明要使用的 DLL 函数，其声明必须放在模块级。声明 DLL 函数的主要作用是指明该函数所在的 DLL 库名及路径，以及该函数的参数说明，以便 Windows 能找到该函数，并正确地执行该函数。

【例 9-2】 建立 PC 与 PLC 之间的连接，读取数据块 DB11 中从 DBW0 开始的 20 个字的数据之后断开连接。

在声明 load _ tool 函数之前，应首先声明它使用的用户自定义的数据类型 plcadr：

```
Type plcadrtype
    ADDRESS As Byte      //站地址，默认值为 2
    SEGMENTID As Byte    //段标识符，固定为 0
    SLOTNO As Byte       //槽的编号，默认值为 2
    RACKNO As Byte       //机架号，固定为 0
End Type
```

然后声明程序中使用的函数：

```
Declare Function load _ tool Lib "w95 _ s7.dll "(ByVal nr As Byte, ByVal dev As String, adr As plcadr-
type)As Long                    //nr 是 PC 要激活的连接的个数(1－32 个)，dev 是用户驱动设备的名称
                                //对于 MPI 驱动器，dev 为"S7ONLINE"；adr 是连接的地址列表
Declare Function d _ field _ read Lib "w95 _ s7.dll "(ByVal db As Long, ByVal no As Long, ByVal
```

```
amount As Long, value As Byte) As Long                    //读取数据块 db 从 no 开始的
                                                            amount 个字节,存放到
                                                          //PC 的 value 数组中
Declare Function unload _ tool Lib "w95 _ s7.dll "() As Long   //断开连接
```

load _ tool 初始化适配器,检查是否已装载驱动器,初始化连接的地址,激活选定的接口。

与通信有关的函数在执行之后如果没有错误,返回值为 0,否则返回错误信息对应的十六进制代码。为了找到出错的原因,可以调用函数 error _ message 来查阅错误代码对应的错误信息,或者查阅用户手册附录中的 error _ message(错误信息)。

new _ ss 用来激活 PLC 与 PC 的连接,例如 new _ ss(2)表示激活 PC 与地址为 2 的 PLC 的连接,也可以用它来重新建立已经关闭的连接。如果只有一个连接,不必使用 new _ ss 函数。它的声明格式为:

```
Declare Function new _ ss Lib "w95 _ s7.dll "(ByVal nr As Long) As Long
```

下面是程序代码:

```
Dim buffer(20) As Byte : Dim plcadr (2) As plcadrtype
plcadr(0).ADDRESS = 2 : plcadr(0).SEGMENTID = 0 : plcadr(0).SLOTNO = 2 : plcadr(0)
.RACKNO = 0
plcadr(1).ADDRESS = 0                                      //为 0 表示地址列表结束
res = load _ tool (1, "S7ONLINE ", plcadr)                 //初始化 1 个连接,MPI 驱动
                                                            器,plcadr 为地址列表
res = db  read (11, 0, 20, buffer(0))                      //读取数据块 DB11 中从 DBW0
                                                            开始的 20 个字节的数据
res = unload _ tool()                                      //断开连接
```

9.4.4 数据传输函数

PRODAVE 提供两类数据传输函数,分别用于 S7-200 和 S7-300/400,它们的使用方法相似。PRODAVE 主要用于 S7-300/400,本节不介绍用于 S7-200 的数据传输函数。

因为适配器不检查数据区的大小,从 PLC 中读取数据时,PC 存放读取的数据的数据区必须足够大。

1. 读 PLC 字节的函数

```
X _ field _ read (ByVal no As Long, ByVal amount As Long, value As Byte) As Long
```

这类函数读取 PLC 的 X 地址区中从地址 no 开始的 amount 个字节的数据,存放在 PC 的数组变量 value 中。其中的 X 可取 e(输入 I)、a(输出 Q)和 m(位存储器 M)。

【例 9-3】 用 a _ field _ read 函数读取 QB0 和 QB1,保存在数组变量 value _ byte 中。

首先在 VB 的 Module 中声明该命令:

```
Declare Function a _ field _ read Lib "w95 _ s7.dll "(ByVal no As Long,ByVal amount As Long,value As
Byte) As Long
```

程序代码如下:

```
Dim value_byte(2) As Byte                                    //定义数组变量
error = a_field_read(0, 2, value_byte(0))                    //读取 PLC 的输出字节 QB0 和 QB1
If error <> 0 Then                                           //如果出错
    TxtShow = "a_field  read error:"& Hex(error)             //显示错误信息
    MsgBox ("a_field_read error")
Else                                                         //反之显示读取的数据
    TxtShow.Text = "Output Value="& Hex(value_byte(0)) & ""& Hex(value_byte(1))
End If
```

2. 写 PLC 字节的函数

```
X_field_write (ByVal no As Long, ByVal amount As Long, value As Byte) As Long
```

这类函数将存放在 PC 的数组变量 value 中的数据,写入 PLC 的 X 地址区从地址 no 开始的 amount 个字节中,X 可取 a 和 m。

【例 9-4】 用 a_field_write 函数将数组变量 value_byte 中的数据写入 PLC 的 QB0 和 QB1。

首先在 VB 的 Module 中声明该命令:

```
Declare Function a_field_write Lib "w95_s7 "(ByVal no As Long,ByVal amount As Long,value As
Byte) As Long
```

程序代码如下:

```
Dim value_byte(2) As Byte                                    //定义数组变量
value_byte(0) = &HFF : value_byte(1) = &H3                   //准备要写入的数据
res = a_field_write(0, 2, value_byte(0))                     //写数据
```

3. 读/写数据块中的字节的函数

```
d_field_read/write (ByVal db As Long, ByVal no As Long, ByVal amount As Long, value As Byte)
As Long
```

d_field_read 读取 PLC 的 db 数据块中从地址 no 开始的 amount 个字节的数据,存放在 PC 的变量 value 中。d_field_write 将存放在 PC 的数组变量 value 中的 amount 个字节的数据,写入 PLC 的 db 数据块中从地址 no 开始的区域。

4. 读定时器/计数器字

```
X_field_read (ByVal no As Long, ByVal amount As Long, value As Integer)
```

这类函数读取从 PLC 的地址 no 开始的 amount 个定时器或计数器的当前值,存放在 PC 的数组变量 value 中。X 可取 t(定时器)和 z(计数器)。

【例 9-5】 用 t_field_read 读取 T10 和 T11 的当前值。

首先在 VB 的 Module 中声明该命令:

```
Declare Function t_field_read Lib "w95_s7.dll "(ByVal no As Long, ByVal amount As Long, value
As Integer) As Long
```

程序代码如下:

```
Dim value As Integer                                    //定义存放返回值的变量
res = t_field_read(10,2,value)                          //读取 T10 和 T11 的当前值
```

5. 写计数器字

```
z_field_write(ByVal no As Long, ByVal amount As Long, value As Integer )
```

z_field_write 将存放在 PC 的数组变量 value 中的 amount 个字的数据，写入 PLC 从地址 no 开始的的计数器区，改写的是计数器的当前值。

6. 读/写数据块中的字

```
db_read/write (ByVal db As Long, ByVal no As Long, amount As Long, value As Integer)
```

db_read 读取 PLC 的数据块 db 中从地址 no 开始的 amount 个数据字，存放到 PC 的变量存储区 value 中。db_write 将 PC 的变量存储区 value 中的 amount 个数据字，写入 PLC 的数据块 db 中从 no 开始的数据区。

7. 标志状态测试

mb_bittest 检测地址为 no 的标志字节（即位存储器字节 MB）中的第 bitno 位。返回值 value 与该位的 0/1 状态相同。

【例 9-6】 用 mb_bittest 检测 M0.1 的状态，返回值用位变量 reValue 保存。

首先在 VB 的 Module 中声明该命令：

```
Declare Function mb_bittest Lib "w95_s7.dll "(ByVal no As Long, ByVal bitno As Long, value As
Boolean) As Long
```

程序代码如下：

```
Dim reValue As Boolean                                  //定义存放返回值的位变量
error = mb_bittest(0,1,reValue)                         //测试 M0.1 的状态
If error <> 0 Then                                      //如果操作出错
    TxtShow = "mb_bittest error:"& Hex(error)           //显示错误信息
    MsgBox ("mb_bittest error ")
Else
    If reValue Then                                     //否则显示测试结果
        TxtShow.Text = "This bit is Set (ON)"
    Else
        TxtShow.Text = "This bit is Not Set (OFF)"
    End If
End If
```

8. 置位/复位标志

```
mb_setbit /resetbit (ByVal no As Long, ByVal bitno As Long)
```

mb_setbit 和 mb_resetbit 分别将 PLC 中地址为 no 的标志字节（MB）的第 bitno 位置位和复位。它们并不检测该标志位在 PLC 中是否存在。

9. 读混合数据（mix_read）

函数 mix_read 将 mixdatatype 指定的 PLC 中的数据读入到 PC 的 buffer 缓冲区中。可以读下列数据：输入字节（E）、输出字节（A）、标志字节（M）、定时器字（T）、计数器字（Z）和数据块

中的数据(D)。该函数从 PLC 中最多读取 20 个数据。若数据类型为 A、E 或 M,数据长度可以为“b”或“w”;若数据类型为 T、Z 或 D,数据长度应设置为“w”。其中“b”为字节数据;“w”为字数据。读取的数据按指定的结构和顺序存放。

【例 9-7】 用 mix _ read 函数读取输入字节 IB0、输出字节 QB1 以及 DB11 的数据字 DW5,并将它们存放在变量 e、a、d 中。

首先在 VB 的 Module 中声明用户定义的数据类型 mixdatatype,然后再声明该函数:

```
Type mixdatatype
    type As Byte                          //数据类型,type = 0 表示列表结束
    size As Byte                          //数据长度:"b"或"w"
    dbno As Integer                       //数据块起始地址,不是访问数据块时
                                            没有此项
    no As Integer                         //数据序号
End Type
Declare Function mix _ read Lib "w95 _ s7.dll "(data As mixdatatype, buffer As Byte) As Long
```

程序代码如下:

```
Dim buffer(20) As Byte : Dim e, a As Byte : Dim d As Integer : Dim data(10) As mixdatatype
data(0).type = e : data(0).size = b : data(0).no = 0    //IB0 的参数
data(1).type = a : data(1).size = b : data(1).no = 1    //QB1 的参数
data(2).type = d : data(2).size = w : data(2).dbno = 11 : data(2).no = 6
data(3).type = 0                                        //为 0 表示列表结束
error = mix _ read(data(0), buffer(0))                  //读混合数据
If error < > 0 Then                                     //如果读数据出错
    TxtShow = "mix _ read error:"& Hex(error)           //显示错误信息
    MsgBox ("mix _ read error ")
Else                                                    //否则显示读取的数据
    e = buffer(0) : a = buffer(1) :
    d = CByte(buffer(2)) * 256 + CByte(buffer(3))       //buffer(2)中为 DBW6 的高字节
    TxtShow = "e =" & e & " a =" & a & " d =" & d
End If
```

10. 写混合数据(mix _ write)

函数 mix _ write (data As mixdatatype, buffer As Byte)将 PC 的存储区 value 中的数据写入 mixdatatype 指定的 PLC 的混合地址区中。数据类型 mixdatatype 与 mix _ read 的相同,该函数最多可以写 20 个数据。

9.4.5 读取和检测系统信息的函数

1. 读取 PLC 的信息

ag _ info 用于读取 PLC 的软件版本、PG 接口以及 PLC 的产品订货号。并将这些信息存储在 PG/PC 的存储区中,该信息为一串以 0 为终止符的 ASCII 字符串。

2. 读取 PLC 的状态

ag _ zustand (value As Byte) 用于读取 PLC 的运行模式,若 PLC 处于 RUN 模式,则 value

为 0;若处于 STOP 模式,则 value 非 0。

3. 数据块检测(db _ buch)

该函数用于检测某数据块是否存在。使用此函数必须预先设置至少 512 个字(WORD)大小的缓冲区 buffer(512)。检测后如果 buffer(x)不为 0,则 DBx 存在,否则 DBx 不存在,x 为 0~511之间的常数,同时可以检测任意的数据块是否存在。

9.4.6 数据处理函数

前面介绍的都是 w95 _ s7.dll 中用于 PC 与 PLC 之间通信的函数。PRODAVE 在 komfort.dll 中还提供了一些方便用户使用的与通信无关的数据处理函数,常用的有以下几条:

1. 逻辑型数据(位数据)与字节数据的转换函数

函数 boolean _ byte 将 8 位逻辑值转换为 1 个字节数据,byte _ boolean 作相反的转换。

2. 浮点数格式转换函数

函数 gp _ to _ float 将 S7 的浮点值转换成 IEEE 格式的浮点数,float _ to _ gp 作相反的转换。

3. 高、低字节交换函数(kf _ integer)

kf _ integer 函数交换一个 16 位(2 B)整数变量的高字节与低字节的位置。

4. 位测试函数(testbit)

testbit 函数对字节变量的指定位进行测试,判断该位是否为 1。

5. 错误信息函数(error _ message)

error _ message 函数根据错误代码给出相应的错误信息文本。

第一次调用该指令时读取存放错误文本的 ERROR.DAT 文件。

若传送的错误代码为 0,被装载的错误文本文件的文件名被传送到"buffer"缓冲区。如果没有有效的文件名,或者传送的是一个 ZERO(0)指针,就从当前目录读取 ERROR.DAT 文件。因此应保证 ERROR.DAT 文件存在并且与监控程序在相同的子目录中,否则不能读出与错误有关的文本信息。最多可以储存 100 个错误文本。

error _ message 函数正确执行时返回值为 0;执行时出错返回值非 0。

错误文本文件的结构为:

[错误代码(16 进制的 ASCII 形式)]:[错误文本]

例如:

0207:data segment cannot be disable

假设 VB 的安装目录为"C:\Program Files\Microsoft Visual Studio\VB98"。安装 PRODAVE S7 之后,通过查找发现两个文件 ERROR.DAT 和 ERROR.ENG。其中 ERROR.DAT 是德语文档,ERROR.ENG 是英语文档。将 ERROR.ENG 改名为 ERROR.DAT,然后将其复制到 VB 的安装目录之下,就可以使用 error _ message 函数的英语错误文本。可以用 Windows 的附件"记事本"打开该文件,也可以将该文件翻译为中文形式。

9.4.7 PRODAVE 在水轮发电动机组监控系统中的应用

在水电站综合自动化系统中,工控机是机组 LCU(现地控制单元)的核心设备,它通过以

太网与站级计算机相连，通过 RS 232C 接口和 PC/MPI 适配器（编程电缆）与 PLC 链接，用 VB 编写计算机与 PLC 通信的程序，通过调用 Prodave 中的函数与 PLC 进行通信。通信为主从方式，上位机（主站）读写 PLC 的存储区，主站发出询问后，PLC 自动响应。通过通信主要实现以下两种功能：

1. 计算机定时读取 PLC 上传的信息

计算机每 0.5 s 读取一次 PLC 中的 16 个输入字节、12 个输出字节和 16 个模拟量输入字。可以用不同的函数分别读取不同地址区的数据。为了提高通信的效率和简化计算机的通信程序，在 PLC 的每一次循环中将需要传送的数据集中到数据块中一片连续的区域，在 VB 程序中只需用函数 d_field_read 来读取数据块中准备好的数据。PLC 内部数据区之间的传送是相当快速的和方便的。

2. 上位机发送给 PLC 的随机命令的处理

随机命令包括计算机读写 PLC 的实时钟命令、需要确认的开关量命令和不需要确认的增减功率的命令等。为了传送和执行随机命令，在 PLC 的位存储器区设置了一片专用的区域：

MW10 是上位机写入的命令字，MW12 是命令中的数据字节数或 PLC 返回的数据字节数。MW14（flag）是命令执行标志字，在初始化程序 OB100 中将它置为 1，发送命令时上位机将它清 0，PLC 成功执行命令后 PLC 将它置为 1，执行时出错 PLC 将它置为 2。从 MB16 开始存放命令中附带的数据或 PLC 返回的数据。

上位机用 d_field_write 函数发送命令。以读取 PLC 的实时钟为例，写入 MW10 的命令字为 0250H，写入 MW12 和 MW14 的均为 0（命令中没有数据）。PLC 检查到标志字 MW14 为 0 时，根据 MW10 中的命令字判断为读实时钟命令，因此用系统功能 READ_CLK 读出实时钟内 8B 的当前时间值，送入 MB16～MB23。同时将返回的数据字节数 6 送 MW12，并将标志字 MW14 置为 1。上位机发送命令后用函数 d_field_read 和较高的频率读取 MW12 和标志字 MW14 的值，在 MW14 变为 1 时读取 MB16～MB21 中的时间值。

第 10 章　S7-300/400 在模拟量闭环控制中的应用

10.1　模拟量闭环控制的基本概念

10.1.1　模拟量闭环控制系统的组成

1. 模拟量单闭环控制系统的组成

典型的 PLC 模拟量单闭环控制系统如图 10-1 所示，虚线中的部分是用 PLC 实现的。

在模拟量闭环控制系统中，被控量 $c(t)$（例如压力、温度、流量、转速等）是连续变化的模拟量，大多数执行机构（例如晶闸管调速装置、电动调节阀和变频器等）要求 PLC 输出模拟信号 $mv(t)$，而 PLC 的 CPU 只能处理数字量。$c(t)$ 首先被测量元件（传感器）和变送器转换为标准量程的直流电流信号或直流电压信号 $pv(t)$，例如 4～20 mA，1～5 V，0～10 V，PLC 用 A/D 转换器将它们转换为数字量 $pv(n)$。

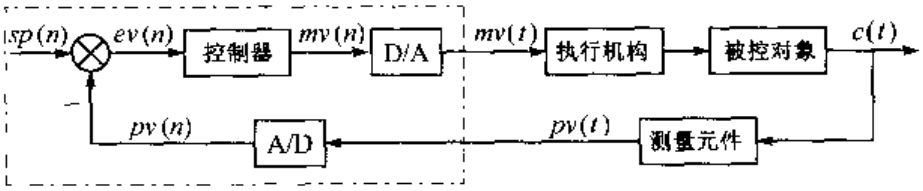


图 10-1　PLC 模拟量闭环控制系统方框图

模拟量与数字量之间的相互转换和 PID 程序的执行都是周期性的操作，其间隔时间称为采样周期 T_s。各数字量括号中的 n 表示该变量是第 n 次采样计算时的数字量。

图 10-1 中的 $sp(n)$ 是给定值，$pv(n)$ 为 A/D 转换后的反馈量，误差 $ev(n) = sp(n) - pv(n)$。

D/A 转换器将 PID 控制器输出的数字量 $mv(n)$ 转换为模拟量（直流电压或直流电流）$mv(t)$，再去控制执行机构。

例如在加热炉温度闭环控制系统中，用热电偶检测炉温，温度变送器将热电偶输出的微弱的电压信号转换为标准量程的电流或电压，然后送给模拟量输入模块，经 A/D 转换后得到与温度成比例的数字量，CPU 将它与温度设定值比较，并按某种控制规律（例如 PID 控制算法）对误差值进行运算，将运算结果（数字量）送给模拟量输出模块，经 D/A 转换后变为电流信号或电压信号，用来控制电动调节阀的开度，通过它控制加热用的天然气的流量，实现对温度的闭环控制。$c(t)$ 为系统的输出量，即被控量，例如加热炉中的温度。

模拟量控制系统分为恒值控制系统和随动系统。恒值控制系统的给定值由操作人员提供，一般很少变化，例如温度控制系统，转速控制系统等。随动系统的输入量是不断变化的随机变量，例如高射炮的瞄准控制系统和电动调节阀的开度控制系统就是典型的随动系统。闭环负反馈控制可以使控制系统的反馈量 $pv(n)$ 等于或跟随给定值 $sp(n)$。以炉温控制系统为

例，假设输出的温度值 $c(t)$低于给定的温度值，反馈量 $pv(n)$小于给定值 $sp(n)$，误差 $ev(n)$为正，控制器的输出量 $mv(t)$将增大，使执行机构(电动调节阀)的开度增大，进入加热炉的天然气流量增加，加热炉的温度升高，最终使实际温度接近或等于给定值。

天然气压力的波动、工件进入加热炉，这些因素称为扰动量，它们会破坏炉温的稳定。闭环控制可以有效地抑制闭环中各种扰动的影响，使被控量趋近于给定值。

闭环控制系统的结构简单，容易实现自动控制，因此在各个领域得到了广泛的应用。

2. 复杂的控制系统的结构

(1) 级联控制

级联控制又称串级控制，两个控制器串行连接，第一个控制器(主控制器)的输出 SP2 是伺服控制器的设定值(见图 10-2)。

级联控制的控制性能可以用附加的过程变量来改进。为此在系统中适当的地方引出辅助过程变量 PV2，将它与主控制器的输出 SP2 比较，得到的误差值作为伺服控制器的输入量。

主控制器的给定值为 SP1，反馈值为 PV1，它调整 SP2，以便尽可能快地使过程变量 PV1 在没有超调的情况下达到设定值。

以电动调节阀为执行机构的控制系统就是一个级联控制系统。电动调节阀有一个位置随动系统，图 10-2 中的 PV2 是阀门的开度，即阀芯的位置。通过辅助回路的调节作用，阀门的开度与主控制器的输出量 SP2 成正比。

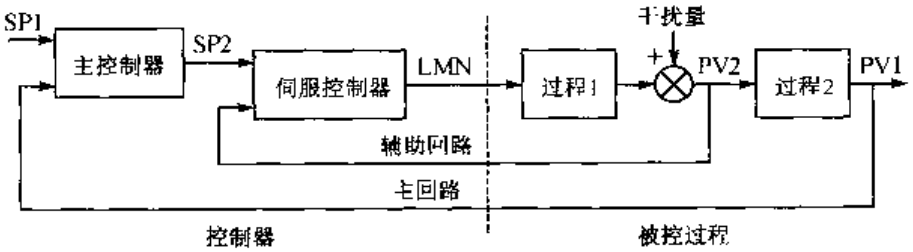


图 10-2　级联控制器

(2) 混合控制器

混合控制器总的设定值 SP 按一定的比例分配给各控制组件，各混合系数之和应为 1，例如图 10-3 中的 FAC1 + … + FAC4 = 1。

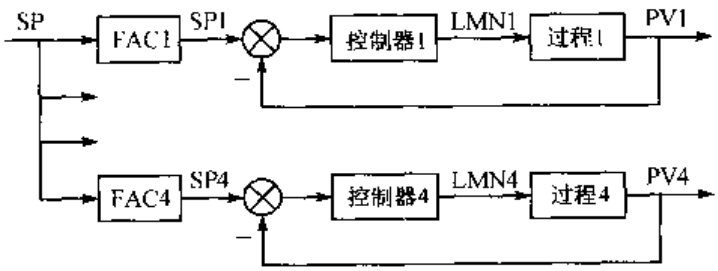


图 10-3　混合控制器

(3) 单闭环比例控制器

对于某些控制系统来说，控制两个过程变量之间的比例比控制它们的绝对数值更重要，例如控制两台需要同步的设备的速度，可以使用图 10-4 中的单闭环比例控制器。

(4) 多闭环比例控制器

多闭环比例控制使两个过程变量 PV1 和 PV2 之比保持为常数。为此用第 1 个控制闭环的过程量 PV1 来计算第 2 个控制闭环的设定值。在过程变量 PV1 动态变化的过程中也能保证 PV1 与 PV2 之间保持设定的比例。如图 10-5 所示。

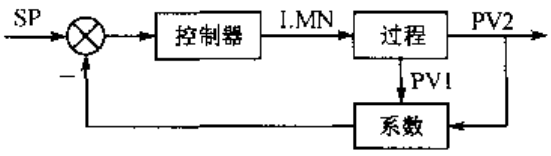


图 10-4 单闭环比例控制器

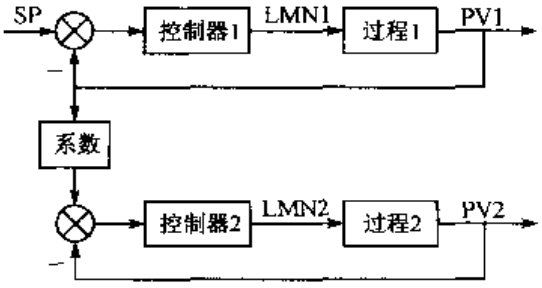


图 10-5 多闭环比例控制器

(5) 二级控制器

二级控制器(Two-Step Controller)只能提供开关量的两个相反的输出状态,例如开关的接通和断开。典型控制为通过继电器输出的脉冲宽度调制信号对加热系统的控制。

(6) 三级控制器

三级控制器只能提供开关量的 3 个输出状态。应区分脉冲宽度调制(例如加热和冷却,加热-关机-冷却)和使用积分式执行机构的步进控制(例如右-停止-左)之间的区别。

3. 变送器的选择

变送器用于将传感器提供的电量或非电量转换为标准量程的直流电流或直流电压信号,例如 DC 0～10 V 和 4～20 mA。变送器分为电流输出型和电压输出型,电压输出型变送器具有恒压源的性质,PLC 模拟量输入模块的电压输入端的输入阻抗很高,例如 100 kΩ～10 MΩ。如果变送器距离 PLC 较远,通过线路间的分布电容和分布电感产生的干扰信号电流在模块的输入阻抗上将产生较高的干扰电压。例如 1 μA 干扰电流在 10 MΩ 输入阻抗上将产生 10 V 的干扰电压信号,所以远程传送模拟量电压信号时抗干扰能力很差。

电流输出具有恒流源的性质,恒流源的内阻很大。PLC 的模拟量输入模块输入电流时,输入阻抗较低(例如 250 Ω)。线路上的干扰信号在模块的输入阻抗上产生的干扰电压很低,所以模拟量电流信号适于远程传送。

电流传送比电压传送的传送距离远得多,S7-300/400 的模拟量输入模块使用屏蔽电缆信号线时允许的最大距离为 200 m。

变送器分为二线式和四线式,四线式变送器有 4 根线:电源线、信号线和公共线。二线式变送器只有两根外部接线,它们既是电源线,也是信号线,输出 4～20 mA 的信号电流,DC 24 V 电源串接在回路中。通过调试,在被检测信号量程的下限时输出电流为 4 mA,被检测信号满量程时输出电流为 20 mA。二线式变送器的接线少,信号可以远传,在工业中得到了广泛的应用。

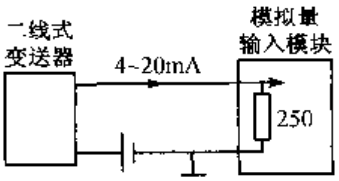


图 10-6 二线式变送器

10.1.2 闭环控制的主要性能指标

由于给定输入信号或扰动输入信号的变化,系统的输出量达到稳态值之前的过程称为过

渡过程或动态过程。系统的动态性能常用阶跃响应(阶跃输入时输出量的变化)的参数来描述。阶跃输入信号在 $t=0$ 之前为 0，$t>0$ 时为某一恒定值。

输出量第一次达到稳态值的时间 t_r 称为上升时间(见图 10-7)，上升时间反映了系统在响应初期的快速性。

系统进入并停留在稳态值 $c(\infty)$ 上下 ±5%(或 2%)的误差带内的时间 t_s 称为调节时间，到达调节时间表示过渡过程已基本结束。

设动态过程中输出量的最大值为 $c_{max}(t)$，如果它大于输出量的稳态值 $c(\infty)$，超调量

$$\sigma\% = \frac{c_{max}(t) - c(\infty)}{c(\infty)} \times 100\%$$

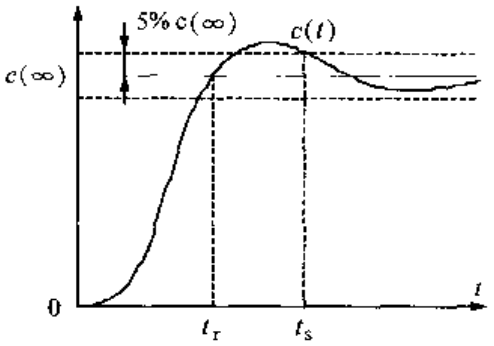


图 10-7　被控对象的阶跃响应曲线

超调量反映了系统的相对稳定性，它越小动态稳定性越好，一般希望超调量小于 10%。

系统的稳态误差是进入稳态后的期望值与实际值之差，它反映了系统的稳态精度。

10.1.3　闭环控制反馈极性的确定

闭环控制必须保证系统是负反馈(误差 = 给定值 - 反馈值)，而不是正反馈(误差 = 给定值 + 反馈值)。如果系统接成了正反馈，将会失控，被控量会往单一方向增大或减小，给系统的安全带来极大的威胁。

闭环控制系统的反馈极性与很多因素有关，例如因为接线改变了变送器输出电流或输出电压的极性，在 PID 控制程序中改变了误差的计算公式，改变了某些直线位移传感器或转角位移传感器的安装方向，都会改变反馈的极性。

可以用下述的方法来判断反馈的极性：在调试时断开 D/A 转换器与执行机构之间的连线，在开环状态下运行 PID 控制程序。如果控制器中有积分环节，因为反馈被断开了，不能消除误差，这时 D/A 转换器的输出电压会向一个方向变化。这时如果假设接上执行机构，能减小误差，则为负反馈，反之为正反馈。

以温度控制系统为例，假设开环运行时给定值大于反馈值，若 D/A 转换器的输出值不断增大，如果形成闭环，将使电动调节阀的开度增大，闭环后温度反馈值将会增大，使误差减小，由此可以判定系统是负反馈。

10.2　数字 PID 控制器

10.2.1　PID 控制器的优点

PID 是比例、微分、积分的缩写，PID 控制器是应用最广的闭环控制器，有人估计现在有 90%以上的闭环控制采用 PID 控制器。这是因为 PID 控制具有以下的优点：

(1) 不需要被控对象的数学模型

自动控制理论中的分析和设计方法主要是建立在被控对象的线性定常数学模型的基础上的。该模型忽略了实际系统中的非线性和时变性，与实际系统有较大的差距。对于许多工业控制对象，根本就无法建立较为准确的数学模型，因此自动控制理论中的设计方法很难用于大

多数控制系统。对于这一类系统，使用 PID 控制可以得到比较满意的效果。

(2) 结构简单，容易实现

PID 控制器的结构典型，程序设计简单，计算工作量较小，各参数有明确的物理意义，参数调整方便，容易实现多回路控制、串级控制等复杂的控制。

(3) 有较强的灵活性和适应性

根据被控对象的具体情况，可以采用 PID 控制器的多种变种和改进的控制方式，例如 PI、PD、带死区的 PID、被控量微分 PID、积分分离 PID 和变速积分 PID 等，但比例控制一般是必不可少的。随着智能控制技术的发展，PID 控制与神经网络控制等现代控制方法结合，可以实现 PID 控制器的参数自整定，使 PID 控制器具有经久不衰的生命力。

(4) 使用方便

现在已有很多 PLC 厂家提供具有 PID 控制功能的产品，例如 PID 闭环控制模块、PID 控制指令和 PID 控制系统功能块等，它们使用简单方便，只需设定一些参数即可，有的产品还具有参数自整定功能。

10.2.2 PID 控制器的数字化

1. PID 控制器在连续控制系统中的表达式

PID 控制器的传递函数为

$$\frac{MV(s)}{EV(s)} = K_p\left(1 + \frac{1}{T_I s} + T_D s\right)$$

模拟量 PID 控制器的输出表达式为

$$mv(t) = K_p[ev(t) + \frac{1}{T_I}\int ev(t)dt + T_D\frac{dev(t)}{dt}] + M \qquad (10\text{-}1)$$

式中控制器的输入量(误差信号) $ev(t) = sp(t) - pv(t)$，$sp(t)$为设定值，$pv(t)$为过程变量(反馈值)；$mv(t)$是控制器的输出信号，K_P 为比例系数，T_I 和 T_D 分别是积分时间常数和微分时间常数，M 是积分部分的初始值。

式(10-1)中等号右边的前 3 项分别是比例、积分、微分部分，它们分别与误差 $ev(t)$、误差的积分和误差的微分成正比。如果取其中的一项或两项，可以组成 P、PD 或 PI 调节器。需要较好的动态品质和较高的稳态精度时，可以选用 PI 控制方式；控制对象的惯性滞后较大时，应选择 PID 控制方式。

2. 积分部分的近似计算

设执行 PID 控制功能块的时间间隔(即 PID 控制的采样周期)为 T_s，第 n 次 PID 运算时的时间为 T_Sn，因为 PID 程序运行时 T_S 为常数，可以将 $t = T_Sn$ 时 PID 的输入量 $ev(T_Sn)$ 简写为 $ev(n)$，输出量 $mv(T_Sn)$简写为 $mv(n)$。

式(10-1)中的积分对应于图 10-8 中曲线 $ev(t)$ 与坐标轴包围的面积，一般用矩形积分来近似精确积分，每块矩形的面积为 $ev(jT_s)T_s$。为了书写方便，将 $ev(jT_s)$ 简写为 $ev(j)$，各块矩形的总面积为 $T_s\sum_{j=1}^{n} ev(j)$。当 T_s 较小时，积分的误差不大。

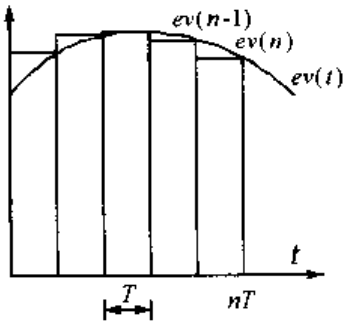


图 10-8 积分的近似运算

3. 微分部分的近似计算

式(8-1)中的微分用差分来近似,即令

$$\frac{de_V(t)}{dt}\approx\frac{\Delta e_V(t)}{\Delta t}=\frac{e_V(n)-e_V(n-1)}{T_s}$$

将积分和微分的近似表达式代入式(10-1),第 n 次采样时控制器的输出为

$$m_V(n)=K_P\left\{e_V(n)+\frac{T_s}{T_I}\sum_{j=1}^{n}e_V(j)+\frac{T_D}{T_s}[e_V(n)-e_V(n-1)]\right\}+M \qquad (10\text{-}2)$$

式中 $e_V(n\text{-}1)$是第 n-1 次采样时的误差值,上式可以以化简为

$$m_V(n)=K_P e_V(n)+K_I\sum_{j=1}^{n}e_V(j)+K_D[e_V(n)-e_V(n-1)]+M \qquad (10\text{-}3)$$

式中,$K_I=K_P T_s/T_I$、$K_D=K_P T_D/T_S$ 分别是积分系数和微分系数。

在 SFB 41"CONT _ C"(连续控制器)中,K_P、T_I、T_D 和 M 分别对应于输入参数 GAIN, TI, TD 和积分初值 I _ ITLVAL。

电动调节阀的控制器用位置传感器检测阀门的开度,通过闭环位置随动系统,使阀门的开度与阀门控制器的输入信号(即 PID 控制器的输出)成正比。

式(10-3)的控制器输出值 $m_V(n)$与电动调节阀的阀门开度(即阀芯的位置)相对应,通常将它称为 PID 的位置式算法。

4. 不完全微分 PID

微分的引入可以改善系统的动态性能,但也容易引入高频干扰,为此在微分部分增加一阶惯性滤波,以平缓输出值的剧烈变化。设滤波时间常数为 T_f,在 SFB 41 中,T_f 对应于微分操作的延迟时间 TM _ LAG。不完全微分 PID 的传递函数为

$$\frac{MV(s)}{EV(s)}=K_P\left(1+\frac{1}{T_I s}+\frac{T_D s}{T_f s+1}\right) \qquad (10\text{-}4)$$

5. 死区特性在 PID 控制中的应用

在控制系统中,某些执行机构如果频繁动作,会导致小幅振荡,造成严重的机械磨损。从控制要求来说,很多系统又允许被控量在一定范围内存在误差。带死区的 PID 控制器(见图 10-9)能防止执行机构的频繁动作。当死区非线性环节的输入量(即误差 $e_V(n)$)的绝对值小于设定值 B 时,死区非线性的输出量(即 PID 控制器的输入量)为 0,这时 PID 控制器的输出分量中,比例部分和微分部分为 0,积分部分保持不变,因此 PID 的输出保持不变,PID 控制器不起调节作用,系统处于开环状态。当误差的绝对值超过设定值时,开始正常的 PID 控制。

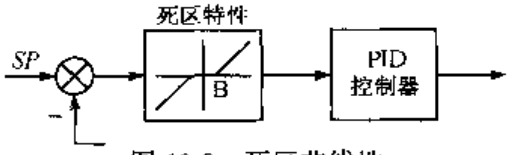


图 10-9 死区非线性

在 SFB 41"CONT _ C"中,死区宽度 DEADB _ W 相当于图 10-9 中的死区设定值 B。

10.3 S7-300/400 的模拟量闭环控制功能

10.3.1 S7-300/400 实现闭环控制的方法

S7-300/400 为用户提供了功能强大、使用简单方便的模拟量闭环控制功能。

1. 闭环控制模块

S7-300 的 FM355 和 S7-400 的 FM455 闭环控制模块是智能化的 4 路和 16 路通用闭环控制模块，可以用于化工和过程控制，模块上带有 A/D 转换器和 D/A 转换器。

2. 闭环控制用的系统功能块

除了专用的闭环控制模块外，S7-300/400 也可以用 PID 控制功能块来实现 PID 控制。但是需要配置模拟量输入模块和模拟量输出模块(或数字量输出模块)。连续控制器通过模拟量输出模块输出模拟量数值，步进控制器输出开关量(数字量)，例如二级控制器和三级控制器用数字量模块输出脉冲宽度可调的方波信号。

系统功能块 SFB 41～SFB 43 用于 CPU 31xC 的闭环控制。SFB 41“CONT _ C”用于连续控制，SFB 42“CONT _ S”用于步进控制，SFB 43“PULSEGEN”用于脉冲宽度调制。本节主要介绍 SFB 41～SFB 43。

3. 闭环控制软件包

安装了标准 PID 控制(Standard PID Control)软件包后，文件夹“ \ Libraries \ Standard Library \ PID Controller”中的 FB 41～FB 43 用于 PID 控制，FB 58 和 FB 59 用于 PID 温度控制。FB 41～FB 43 与 SFB 41～SFB 43 兼容。

模糊控制软件包适合于对象模型难以建立，过程特性缺乏一致性，具有非线性，但是可以总结出操作经验的系统。

神经网络控制系统(Neuronal Systems)适用于不完全了解其结构和解决方法的控制问题。它可以用于自动化的各个层次，从单独的闭环控制器到工厂的最优控制。

PID 自整定(PID Self Tuner)软件包可以提供控制优化支持。

10.3.2 使用系统功能块实现闭环控制

典型的 PID 控制系统的硬件环境如图 10-10 所示。

1. SFB 41～SFB 43 的调用

SFB41～43 可以在程序编辑器左边的指令树中的“ \ Library \ Standard Library \ System Function Blocks”(标准库系统功能块)文件夹中找到。

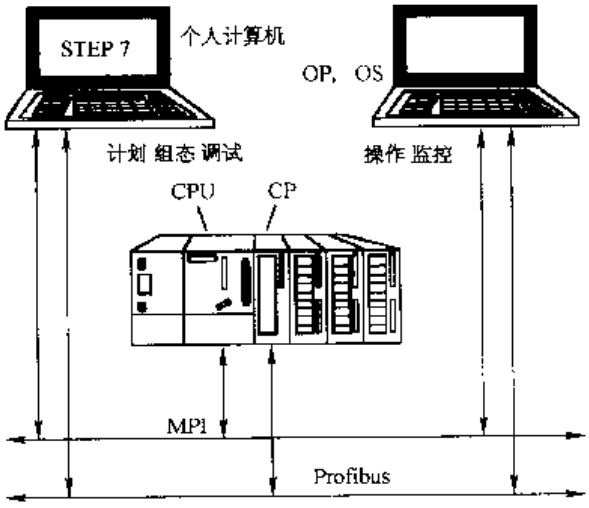


图 10-10 PID 控制的硬件环境

SFB41～43 内有可组态的大量单元，除了创建 PID 控制器以外，还可以处理设定值、过程反馈值，以及对控制器的输出值进行后处理。定期计算所需的数据保存在指定的背景数据块中，允许多次调用 SFB。SFB“PULSEGEN”与 SFB“CONT _ C”组合使用，可以组成脉冲输出的控制器，例如可以用于加热和冷却装置。

PID 控制器的处理速度与 CPU 的性能有关，必须在控制器的数量和控制器的计算频率(采样周期)之间折衷处理。计算频率越高，单位时间的计算量越多，能使用的控制器的数量就越少。PID 控制器可以控制较慢的系统，例如温度和物料的料位等，也可以控制较快的系统，

例如流量和速度等。

2. PID 控制的程序结构

应在启动时执行的组织块 OB 100 中和在定时循环中断 OB(例如 OB35)中调用 SFB 41～43。执行 OB35 的时间间隔(ms,即 PID 控制的采样周期 T_s)在 CPU 属性设置对话框的循环中断选项卡中设置。

调用系统功能块应指定相应的背景数据块,例如 CALL SFB 41, DB 30。

系统功能块的参数保存在背景数据块中,可以通过数据块的编号、偏移地址或符号地址来访问背景数据块。

10.4 连续 PID 控制器 SFB 41

SFB 41"CONT _ C"(连续控制器)的输出为连续变量。可以用 SFB"CONT _ C"作为单独的 PID 恒值控制器,或在多闭环控制中实现级联控制器、混合控制器和比例控制器。控制器的功能基于模拟信号采样控制器的 PID 控制算法,如果需要的话,SFB 41 可以用脉冲发生器 SFB 43 进行扩展,产生脉冲宽度调制的输出信号,来控制比例执行机构的二级或三级(two or three step)控制器。

10.4.1 设定值与过程变量的处理

1. 设定值的输入

浮点数格式的设定值(setpoint)用变量 SP _ INT(内部设定值)输入(见图 10-11)。

2. 过程变量的输入

可以用两种方式输入过程变量(即反馈值):

(1) 用 PV _ IN(过程输入变量)输入浮点格式的过程变量,此时开关量 PVPER _ ON(外围设备过程变量 ON)应为 0 状态。

(2) 用 PV _ PER(外围设备过程变量)输入外围设备(I/O)格式的过程变量,即用模拟量输入模块输出的数字值作为 PID 控制的过程变量,此时开关量 PVPER _ ON 应为 1 状态。

3. 外部设备过程变量转换为浮点数

外部设备(即模拟量输入模块)正常范围的最大输出值(100.0%)为 27648(6C00H),功能 CRP _ IN 将外围设备输入值转换为 -100%～+100%之间的浮点数格式的数值,CPR _ IN 的输出(以%为单位)用下式计算:

$$PV\_R = PV\_PER \times 100 / 27648(\%)$$

4. 外部设备过程变量的标准化

PV _ NORM 功能用下面的公式将 CRP _ IN 的输出 PV _ R 格式化:

$$PV\_NORM\ 的输出 = PV\_R \times PV\_FAC + PV\_OFF$$

式中,PV _ FAC 为过程变量的系数,默认值为 1.0;PV _ OFF 为过程变量的偏移量,默认值为 0.0。PV _ FAC 和 PV _ OFF 用来调节过程输入的范围。

如果设定值有物理意义,实际值(即反馈量)也可以转换为该物理值。图 10-11 中的 PV(过程变量)为 SFB 输出的中间变量。

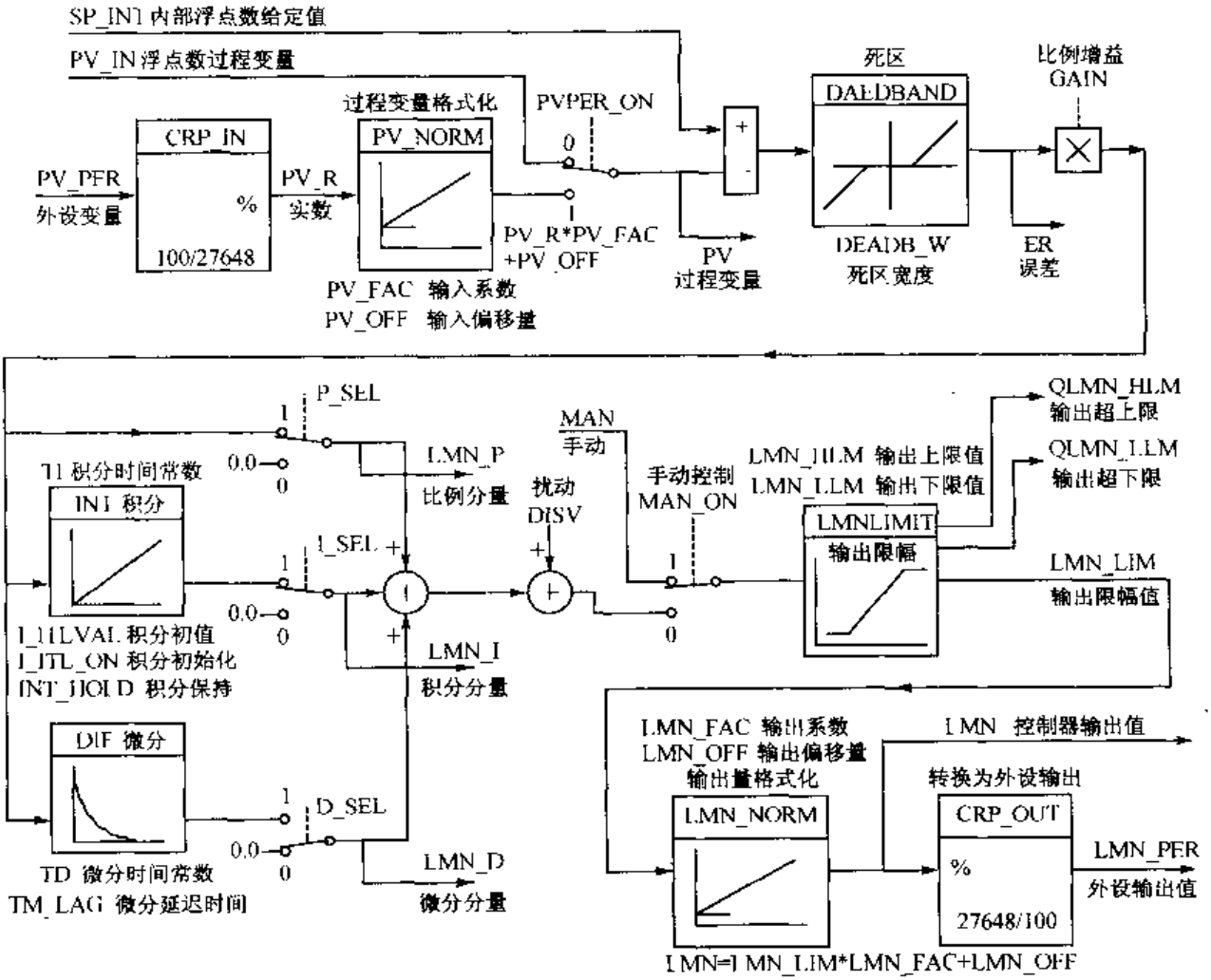


图 10-11 SFB 41 CONT_C 的框图

10.4.2 PID 控制算法

1. 误差的计算与处理

用浮点数格式设定值 SP_INT 减去转换为浮点数格式的过程变量 PV(即反馈值),便得到负反馈的误差。为了抑制由于控制器输出量的量化造成的连续的较小的振荡,例如用 PULSEGEN 进行脉冲宽度调制时可能出现的振荡,用死区(Dead Band)非线性对误差进行处理。死区的宽度由参数 DEADB_W 来定义,如果 DEADB_W = 0,死区被关闭。

图 10-11 中的误差 ER(error)为 SFB 输出的中间变量。

2. 控制器的结构

SFB 41 采用位置式 PID 算法,比例运算、积分运算(INT)和微分运算(DIF)3 部分并联,可以单独激活或取消它们,因此可以将控制器组态为 P、PI、PD 和 PID 控制器。虽然可以组成单独的 I 控制器或 D 控制器,不过很少这样使用。引入扰动量 DISV(Disturbance)可以实现前馈控制。图 10-11 中的 GAIN 为比例部分的增益或比例系数,TI 和 TD 分别为积分时间常数和微分时间常数。输入参数 TM_LAG 为微分操作的延迟时间,建议为 TD/5。

P_SEL（比例作用 on）为 1 时激活比例作用，反之禁止比例作用，默认值为 1。

I_SEL（积分作用 on）为 1 时激活积分作用，反之禁止积分作用，默认值为 1。

D_SEL（微分作用 on）为 1 时激活微分作用，反之禁止微分作用，默认值为 0。

LNM_P、LNM_I 和 LNM_D 分别是 PID 控制器输出中的比例分量、积分分量和微分分量，它们的默认值均为 0.0。

3. 积分器的初始值

SFB"CONT_C"有一个初始化程序，在输入参数 COM_RST（完全重新起动）设置为 1 时该程序被执行。在初始化过程中，如果 I_ITL_ON（积分作用初始化）为 1 状态，将输入 I_ITLVAL 作为积分器的初始值。如果在一个循环中断优先级调用它，它将从该数值开始继续运行，所有其他输出都设置为其默认值。

INT_HOLD 为 1 时积分操作保持，积分输出被冻结。

10.4.3 控制器输出值的处理

1. 手动模式

参数 MAN_ON（手动值 ON）为 1 时为手动模式，为 0 时为自动模式。在手动模式中，控制变量（Manipulated Variable，即控制器的输出值）被手动选择的值 MAN（手动值）代替。

在手动模式时如果令微分项为 0，将积分部分（INT）设置为 LMN - LMN_P - DISV，可以保证手动到自动的无扰切换，即切换时控制器的输出值不会突变，DISV 为扰动输入变量。

2. 输出限幅

LMNLIMIT（输出量限幅）功能用于将控制器输出值（Manipulated Value）限幅。

LMNLIMIT 功能的输入量超出控制器输出值的上极限 LMN_HLM 时，信号位 QLMN_HLM（输出超出上限）变为 1 状态；小于下极限值 LMN_LLM 时，信号位 QLMN_LLM（输出超出下限）变为 1 状态。

3. 输出量的格式化处理

LMN_NORM（输出量格式化）功能用下述公式来将功能 LMNLIMIT 的输出量 LMN_LIM 格式化

$$LMN = LMN\_LIM \times LMN\_FAC + LMN\_OFF$$

式中，LMN 是格式化后浮点数格式的控制器输出值；LMN_FAC 为输出量的系数，默认值为 1.0；LMN_OFF 为输出量的偏移量，默认值为 0.0；LMN_FAC 和 LMN_OFF 用来调节控制器输出量的范围。

4. 输出量转换为外围设备（I/O）格式

控制器输出值如果要送给模拟量输出模块中的 D/A 转换器，需要用"CPR_OUT"功能转换为外围设备（I/O）格式的变量 LMN_PER。转换公式为

$$LMN\_PER = LMN \times 27648/100$$

用参数赋值工具可以进行参数检查，给出错误信息。

10.4.4 SFB 41 的参数

SFB 41 的输入参数见表 10-1,SFB 41 的输出参数见表 10-2。

表 10-1 SFB 41 的输入参数

参数名称	数据类型	地址	说明	默认值
COM_RST	BOOL	0.0	COMPLETE RESTART,完全重新起动,为 1 时执行初始化程序	FALSE
CYCLE	TIME	2	SAMPLE TIME,采样时间,两次块调用之间的时间,取值范围>= 20 ms	T#1s
SP_INT	REAL	6	INTERNAL SETPOINT,内部设定值输入,取值范围=100.0%或物理值	0.0
PV_IN	REAL	10	PROCESS VARIABLE IN,浮点数格式的过程变量输入	0.0
PVPER_ON	BOOL	0.2	PROCESS VARIABLE PERIPHERY ON,使用外围设备输入的过程变量	FALSE
PV_PER	WORD	14	PROCESS VARIABLE PERIPHERY,外部设备输入的 I/O 格式的过程变量值	16#0000
PV_FAC	REAL	48	PROCESS VARIABLE FACTOR,输入的过程变量的系数	1.0
PV_OFF	REAL	52	PROCESS VARIABLE OFFSET,输入的过程变量的偏移量	0.0
DEADB_W	REAL	36	DEAD BAND WIDTH,死区宽度,误差变量死区带的大小,>=0.0 或物理值	0.0
GAIN	REAL	20	PROPORTIONAL GAIN,比例增益输入,用于设置控制器的增益	2.0
TI	TIME	24	RESET TIME,积分时间输入,积分器的响应时间,取值范围应>= CYCLE	T#20s
TD	TIME	28	DERIVATIVE TIME,微分时间输入,微分器的响应时间	T#10s
TM_LAG	TIME	32	TIME LAG OF THE DERIVATIVE ACTION,微分操作的延迟时间输入	T#2s
P_SEL	BOOL	0.3	PROPORTIONAL ACTION ON,为 1 时打开比例(P)操作	TURE
I_SEL	BOOL	0.4	INTEGRAL ACTION ON,为 1 时打开积分(I)操作	TURE
D_SEL	BOOL	0.7	DERIVATIVE ACTION ON,为 1 时打开微分(D)操作	FALSE
I_ITLVAL	REAL	64	INITIALIZATION VALUE OF THE INTEGRAL ACTION,积分操作的初始值	0.0
I_ITL_ON	BOOL	0.6	INITIALIZATION OF THE INTEGRAL ACTION,积分作用初始化,为 1 时将输入 I_ITLVAL 作为积分器的初值	FALSE
INT_HOLD	BOOL	0.5	INTEGRAL ACTION HOLD,为 1 时积分操作保持,为 1 时积分输出被冻结	FALSE
DISV	REAL	68	DISTURBANCE VARIABLE,扰动输入变量	0.0
MAN_ON	BOOL	0.1	MANUAL VALUE ON,为 1 时控制循环将被中断,手动值被设置为操作值	TURE
MAN	REAL	16	MANUAL VALUE,操作员接口输入的手动值,取值范围±100.0%或物理值	0.0

（续）

参数名称	数据类型	地　址	说　明	默认值
LMN_HLM	REAL	40	MANIPULATED VALUE HIGH LIMIT，控制器输出上限值，取值范围 LMN_LLM ~100.0%或物理值	100.0
LMN_LLM	REAL	44	MANIPULATED VALUE LOW LIMIT，控制器输出下限值，取值范围-100.0% ~ LMN_HLM 或物理值	0.0
LMN_FAC	REAL	56	MANIPULATED VALUE FACTOR，控制器输出量的系数	1.0
LMN_OFF	REAL	60	MANIPULATED VALUE OFFSET，控制器输出量的偏移量	0.0

表 10-2　SFB 41 的输出参数

参数名称	数据类型	地　址	说　明	默认值
PV	REAL	92	PROCESS VARIABLE，格式化的过程变量输出	0.0
ER	REAL	96	ERROR SIGNAL，死区处理后的误差输出	0.0
LMN_P	REAL	80	PROPORTIONALITY COMPONENT，控制器输出值中的比例分量	0.0
LMN_I	REAL	84	INTEGRAL COMPONENT，控制器输出值中的积分分量	0.0
LMN_D	REAL	88	DERIVATIVE COMPONENT，控制器输出值中的微分分量	0.0
QLMN_HLM	BOOL	78.0	HIGH LIMIT OF MANIPULATED VALUE REACHED，控制器输出超过上限	FALSE
QLMN_LLM	BOOL	78.1	LOW LIMIT OF MANIPULATED VALUE REACHED，控制器输出小于下限	FALSE
LMN	REAL	72	MANIPULATED VALUE，浮点数格式的控制器输出值	0.0
LMN_PER	WORD	76	MANIPULATED VALUE PERIPHERY I/O，I/O 格式的控制器输出值	16# 0000

10.5　脉冲发生器 SFB 43

10.5.1　脉冲发生器的功能与结构

1. 脉冲发生器的功能

SFB 43“PULSEGEN”（脉冲发生器）与 PID 控制器配合使用（见图 10-12），用脉冲输出来控制比例执行机构。该功能一般与连续控制器“CONT_C”一起使用，使用 SFB 43 可以构建脉冲宽度调制的二级（two step）或三级（three step）PID 控制器。

系统功能块“PULSEGEN”通过调制脉冲宽度，将输入变量 INV（即 PID 控制器的输出量 LMN）转换为具有恒定周期的脉冲列，该恒定周期用周期时间 PER_TM 来设置，对应于刷新输入变量的循环时间，PER_TM 应与 CONT_C 的采样周期 CYCLE 相同。

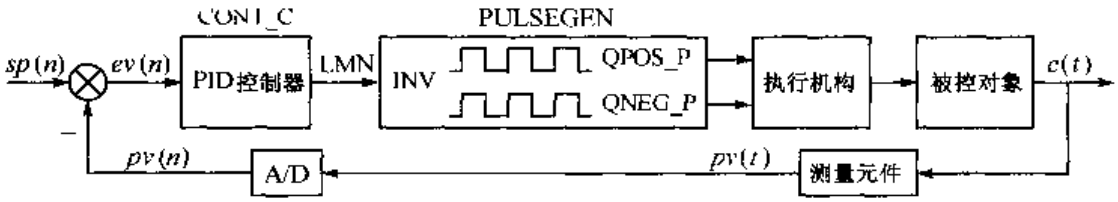


图 10-12　PLC 模拟量闭环控制系统框图

每个周期输出的脉冲宽度与输入变量 INV 成正比，PER _ TM 与 SFB“PULSEGEN”的处理周期是不同的，“PER _ TM”是 SFB“PULSEGEN”处理周期的若干倍(见图 10-13)。每个 PER _ TM 周期调用 SFB“PULSEGEN”的次数反映了脉冲宽度的精度，最小控制值取决于参数 P _ B _ TM。

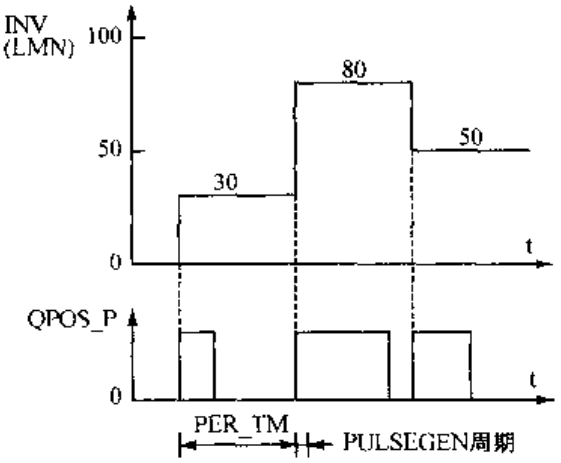


图 10-13　脉宽调制波形图

2. 脉冲宽度调制的方法

脉冲宽度调制简称为脉宽调制。设每个 PER _ TM 周期调用 10 次 SFB“PULSEGEN”，如果输入变量为最大值的 30 %，则前 3 次调用(10 次调用的 30 %)时正脉冲输出 QPOS 为 1 状态。其余 7 次调用(10 次调用的 70 %)时输出 QPOS 为 0 状态。

3. 控制值的精度

在这个例子中，“采样比率”(调用“CONT _ C”与调用“PULSEGEN”次数之比)为 1∶10，因此控制值的精度为 10 %，即输入值 INV 只能映射为以 10%为量化单位的脉冲输出 QPOS 的占空比。

在调用“CONT _ C”的一个周期内增加调用 SFB“PULSEGEN”的次数，可以提高精度。例如调用“PULSEGEN”的次数增加为 100 次时，控制值的分辨率将达到 1 %，建议分辨率不大于 5 %。图 10-14 是 SFB“PULSEGEN”的结构框图。

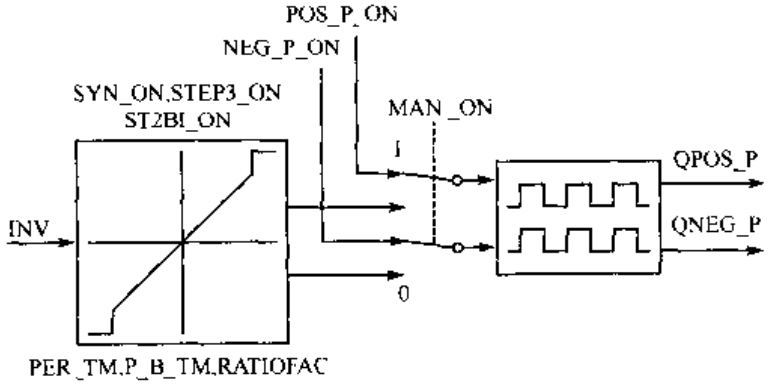


图 10-14　PULSEGEN 框图

4. 自动同步

使用可以刷新 SFB“PULSEGEN”的输入变量“INV”的功能块，例如 CONT _ C，可以使脉

冲输出自动同步，从而保证输入变量的变化尽可能快地以脉冲的形式输出。

脉冲发生器以 PER _ TM 设置的时间间隔为周期，将输入值 INV 转换为对应宽度的脉冲信号。但是因为 INV 一般在较慢的循环中断级中计算，脉冲发生器应在 INV 刷新后尽可能快地将离散的值转换为脉冲信号。

为此，功能块用下述方式对输出脉冲的起动同步：

如果 INV 发生了变化，并且对 SFB 43 的调用不在输出脉冲的第 1 个或最后两个调用周期中，将进行同步。脉冲宽度被重新计算，并在下一个周期中输出一个新的脉冲。

令 SFB 的输入量 SYN _ ON = FALSE，可以关闭自动同步功能。

5. 运行模式的参数设置

根据脉冲发生器设置的参数，PID 控制器可以组态为三级输出、双极性二级输出或单极性二级输出。表 10-3 给出了可能的模式的参数设置。

表 10-3 运行模式的参数设置

操作模式	MAN _ ON	STEP3 _ ON	ST2BI _ ON
三级控制	FALSE	TRUE	Any
双极性二级控制，控制范围 -100 % - 100 %	FALSE	FALSE	TRUE
单极性二级控制，控制范围 0 % ~ 100 %	FALSE	FALSE	FALSE
手动模式	TRUE	Any	Any

6. 二级控制或三级控制中的手动模式

在手动模式(MAN _ ON = TRUE)时，三级控制器或二级控制器的开关量输出可以用信号 POS _ P _ ON 和 NEG _ P _ ON 来设置，而与输入量 INV 无关(见图 10-14 和表 10-4)。

表 10-4 手动模式的输出信号

	POS _ P _ ON	NEG _ P _ ON	QPOS _ P	QNEG _ P
三级控制	FALSE	FALSE	FALSE	FALSE
	TRUE	FALSE	TRUE	FALSE
	FALSE	TRUE	FALSE	TRUE
	TRUE	TRUE	FALSE	FALSE
二级控制	FALSE	Any	FALSE	TRUE
	TRUE	Any	TRUE	FALSE

7. 初始化

SFB"PULSGEN"的初始化程序在输入参数 COM _ RST 为 1 时运行，所有信号输出都被设置为"0"。通过参数赋值工具可以进行参数检查。

10.5.2 三级控制器

1. 三级控制

三级控制用两个开关量信号 QPOS _ P 和 QNEG _ P 产生控制信号的三种状态，用来控制执行机构的状态。表 10-5 给出了用三级控制来控制温度的例子。

表 10-5　温度控制输出信号的状态

输出信号	加　热	执行机构关闭	冷　却
QPOS_P	TRUE	FALSE	FALSE
QNEG_P	FALSE	FALSE	TRUE

CPU 根据输入变量 INV 的大小，通过特性曲线来计算脉冲宽度。特性曲线的形状取决于最小脉冲/最小中断时间 P_B_TM 和比率系数 RATIOFAC，曲线中的拐点是 P_B_TM 引起的。图 10-15 为比率系数为 1 的三级控制器的对称曲线，比率系数通常为 1。

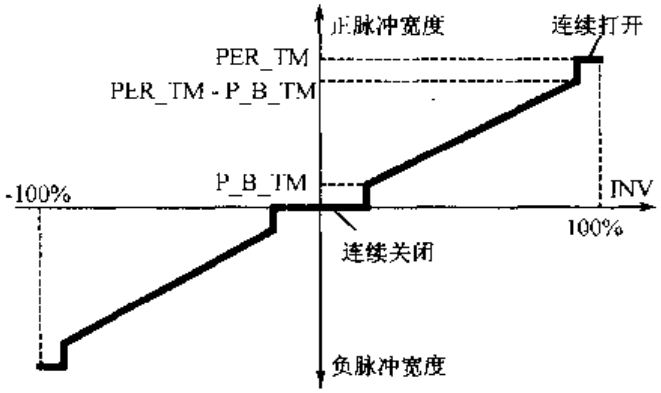


图 10-15　三级控制器的对称曲线

将单位为%的输入变量 INV 与周期时间 PER_TM 相乘，可以计算出正、负脉冲宽度

$$\text{脉冲宽度} = \text{INV} \times \text{PER\_TM} / 100 \tag{10-5}$$

2. 最小脉冲/最小中断时间

正确设置最小脉冲/最小中断时间 P_B_TM，可以防止因短促的接通/断开时间降低开关元件和执行机构的使用寿命。

输入值 INT 过大可能使脉冲宽度大于 PER_TM 与 P_B_TM 的差值，在这种情况下应将它限幅为 100%或-100 %。

3. 比率系数 < 1 的三级控制器

使用比率系数 RATIOFAC，可以改变正脉冲宽度和负脉冲宽度之比。例如可以将不同的时间常数用于热处理的电加热执行机构和水冷却执行机构。比率系数也会影响最小脉冲/断开周期。比率系数小于 1 时脉冲宽度的计算公式为

$$\text{正脉冲宽度} = \text{INV} \times \text{PER\_TM} / 100 \tag{10-5a}$$

$$\text{负脉冲宽度} = \text{INV} \times \text{PER\_TM} \times \text{RATIOFAC} / 100 \tag{10-5b}$$

由式(10-5)可知，小于 1 的比率系数将会减小负脉冲输出的脉冲宽度。图 10-16 为比率系数为 0.5 的二级控制器的不对称曲线。

4. 比率系数 > 1 的三级控制器

比率系数 > 1 时脉冲宽度的计算公式为

$$\text{正脉冲宽度} = \text{INV} \times \text{PER\_TM} /(100 \times \text{RATIOFAC}) \tag{10-6a}$$

$$\text{负脉冲宽度} = \text{INV} \times \text{PER\_TM} / 100 \tag{10-6b}$$

由式(10-6)可知，当 RATIOFAC > 1 时，正脉冲输出的脉冲宽度将会变窄。

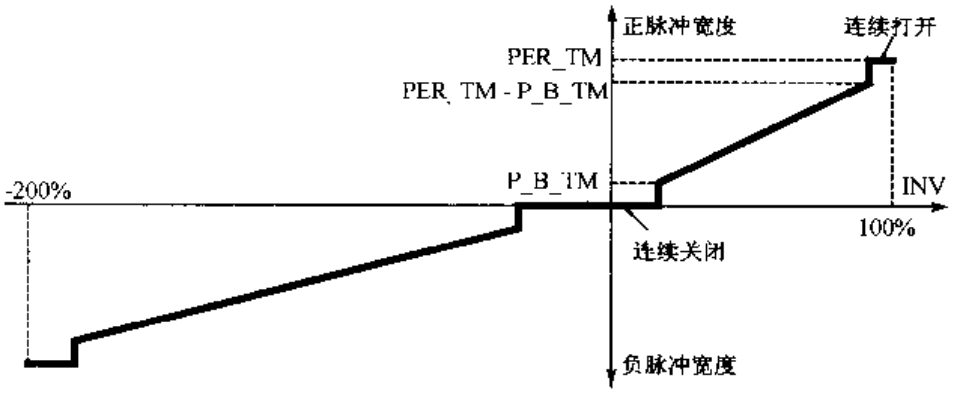


图 10-16　三级控制器的不对称曲线

10.5.3　二级控制器

二级(two-step)控制只用 PULSEGEN 的正脉冲输出 QPOS_P 控制 I/O 执行机构。二级控制器按控制值 INV 的范围分为双极性控制器和单极性控制器(见图 10-17 和图 10-18)。

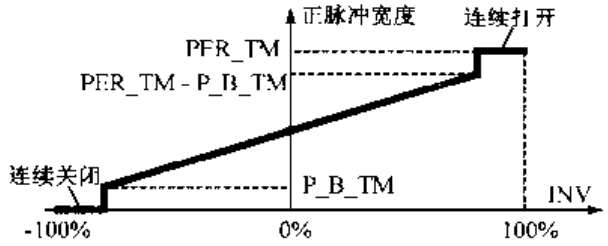


图 10-17　－100%～100%的双极性控制值二级控制

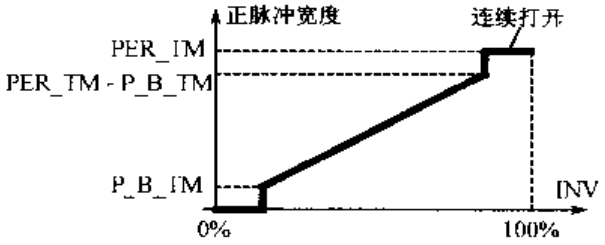


图 10-18　0%～100%的单极性控制值二级控制

如果在控制闭环中二级控制器的执行脉冲需要逻辑状态相反的开关量信号,可以用 QNEG_P 输出负的输出信号(见表 10-6)。

表 10-6　两个输出量的二级控制

脉　冲	执行机构打开	执行机构关闭
QPOS_P	TRUE	FALSE
QNEG_P	FALSE	TRUE

10.5.4　SFB 43 的参数

见表 10-7、10-8。

表 10-7　SFB 43 的输入参数

参数名称	数据类型	地　址	说　明	默认值
INV	REAL	0	INPUT VARIABLE,输入变量,即 SFB 41 输出的模拟量控制值。 对于 RATIOFAC <1 的三级控制:取值范围为 -100/RATIOFAC~100 % 对于 RATIOFAC >1 的三级控制:取值范围为 -100~100/RATIOFAC % 对于双极性二级控制:取值范围为 -100 ~ 100 % 对于单极性二级控制:取值范围为 0~100 %	0.0

（续）

参数名称	数据类型	地　址	说　明	默认值
PER_TM	TIME	4	PERIOD TIME，周期时间，脉冲宽度调制的恒定周期，对应于 PID 控制器的采样时间，脉冲发生器的处理周期和控制器的采样时间之比决定了脉冲宽度调制的精度，应大于等于参数 CYCLE 的 20 倍	T#1s
P_B_TM	TIME	8	MINIMUM PULSE/BREAK TIME，最小脉冲时间或最小断开时间，应不小于参数 CYCLE	T#50 ms
RATIOFAC	REAL	12	RATIO FACTOR，比率系数，用于改变正脉冲宽度和负脉冲宽度之比。例如在热处理中，用于补偿加热和冷却的不同时间常数（例如电加热和水冷却过程）。取值范围 0.1～10.0	1.0
STEP3_ON	BOOL	16.0	THREE STEP CONTROL ON，三级控制打开。在三级控制中，两个输出信号都有效	TRUE
ST2BI_ON	BOOL	16.1	TWO STEP CONTROL FOR BIPOLAR MANIPULATED VALUE RANGE ON，双极型控制值的二级控制打开。用来选择“双极性控制值二级控制”或“单极性控制值二级控制”模式	FALSE
MAN_ON	BOOL	16.2	MANUAL MODE ON，手动模式打开，可以手动设置输出信号	FALSE
POS_P_ON	BOOL	16.3	POSITIVE MODE ON，正脉冲打开。在三级控制的手动模式中，用来控制输出信号 QPOS_P。在二级控制的手动模式中，QNEG_P 必须设置为与 QPOS_P 的状态相反	FALSE
NEG_P_ON	BOOL	16.4	NEGATIVE PULSE ON，负脉冲打开。在三级控制的手动模式中，用来控制输出信号 QPOS_N。在二级控制的手动模式中，QNEG_P 必须设置为与 QPOS_P 的状态相反	FALSE
SYN_ON	BOOL	16.5	SYNCHRONIZATION ON，同步打开。可以用刷新输入变量 INV 的块进行自动同步操作，以保证输入变量的变化尽可能快地以脉冲的形式输出。条件：PER_TM 等于 SFB 41 的采样周期	TRUE
COM_RST	BOOL	16.6	COMPLETE RESTART，若为 1 在起动时执行块的初始化程序。为 0 时控制器运行	FALSE
CYCLE	TIME	18	SAMPLE TIME，采样时间，规定了相邻两次块调用之间间隔的时间。>=20ms	T#10 ms

表 10-8　SFB 43 的输出参数

参数名称	数据类型	地　址	说　明	默认值
QPOS_P	BOOL	22.0	OUTPUT POSITIVE PULSE，输出正脉冲。三级控制总为正脉冲，二级控制时必须与 QPOS_P 的状态相反	FALSE
QNEG_P	BOOL	22.1	OUTPUT NEGATIVE PULSE，输出负脉冲。三级控制总为负脉冲，二级控制必须与 QPOS_P 的状态相反	FALSE

10.6 步进 PI 控制器 SFB 42

10.6.1 步进控制器的结构

SFB 42“CONT _ S”(步进控制器)用开关量输出信号控制积分型执行机构(例如电动调节阀)。可以打开和关闭步进(STEP)控制器的子功能,使控制器满足被控系统的要求。

SFB“CONT _ S”可以用作单独的 PI 恒值控制器,或在级联控制器的辅助控制回路、混合控制器或比例控制器中使用,但是不能作级联控制器的主控制器。控制器的功能基于采样控制器的 PI 控制算法,令参数 TI = T#0 ms 将关闭控制器的积分操作,SFB 42 可以作为 P(比例)控制器使用。

SFB“CONT _ S”有一个初始化程序,它在输入参数 COM _ RST 为 1 时执行。所有其他的输出都设置为默认值。

图 10-19 中的电动调节阀是典型的积分型执行机构,它的两个开关量输入脉冲信号用来控制电动阀的伺服电动机的正转和反转,使调节阀的开度(即阀门阀芯的位置)增大或减小。图中的内环是一个典型的位置随动系统,它的作用是使阀门的开度正比于输入值,即 PI 控制器的输出值。图中的三级(Three step)元件具有带滞环的双向继电器非线性特性,它的作用是将小闭环的误差信号转换为两个开关量信号,它们通过伺服电动机来控制调节阀的开度。

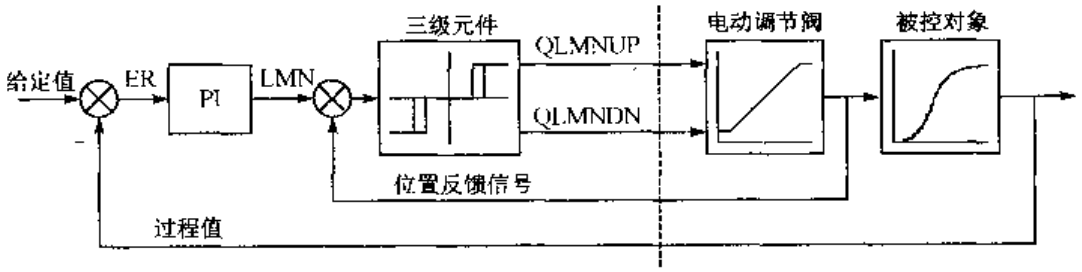


图 10-19 有位置反馈信号的步进控制系统

为了简化系统的结构,图 10-20 中用模拟的阀门位置信号来代替实际的阀门位置反馈信号。图中的参数 MTR _ TM 是执行机构从一个限位位置移动到另一个限位位置所需的时间。

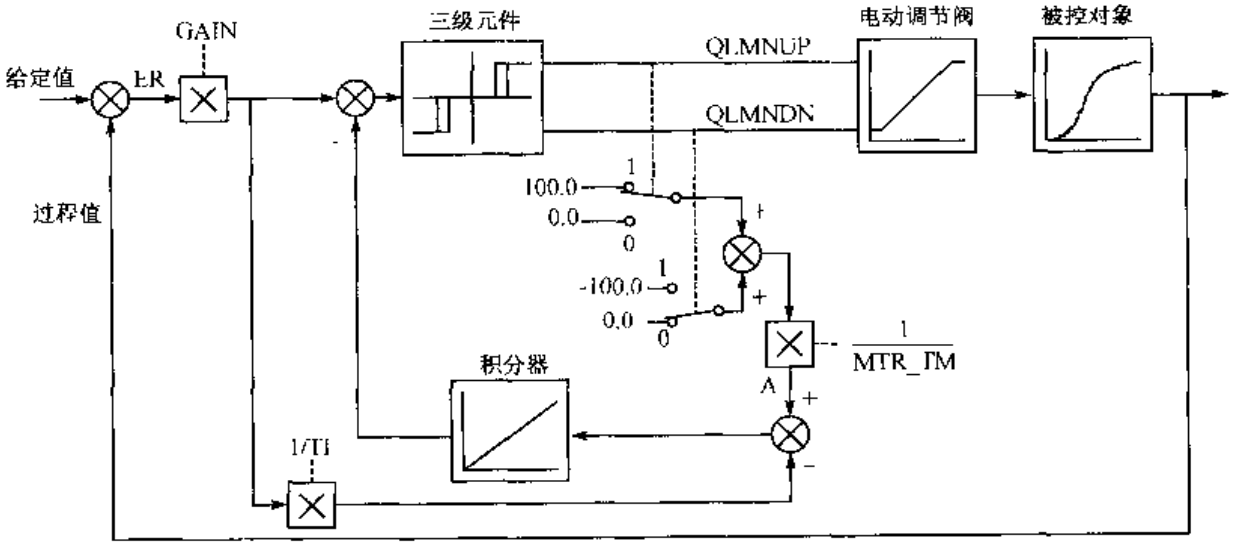


图 10-20 使用模拟的位置反馈信号的步进控制系统

QLMNUP 为 1 时，电动调节阀的开度增大，同时图 10-20 中上面的开关切换到标有“100.0”的位置，积分器对 100.0/MTR _ TM 积分，积分器的输出分量反映了阀门开度增大的情况。当 QLMNDN 为 1 时，积分器对信号-100.0/MTR _ TM 积分，积分器的输出分量反映了阀门开度减小的情况。由上面的分析可知，积分器对图 10-20 中 A 点处的信号 ±100.0/MTR _ TM 积分后的分量可以用来模拟阀门开度(位置)的变化情况。

三级元件的输入信号中有 3 个分量：

(1) ER * GAIN，为 PI 控制器中的比例分量；

(2) ER * GAIN/TI 经积分器积分后的信号，为 PI 控制器中的积分分量；

(3) A 点的信号积分后，得到的模拟的阀门开度(位置)信号。

比例分量与积分分量相加后，得到了 PI 控制器的输出信号，它与模拟的阀门位置信号相减，便得到三级元件的输入信号，因此除了阀门位置信号不同以外，图 10-19 与图 10-20 的结构是相同的。

图 10-21 是 SFB 42“CONT _ S”步进控制器的框图。步进控制器没有使用位置反馈信号，限位停止信号用于限制脉冲输出。

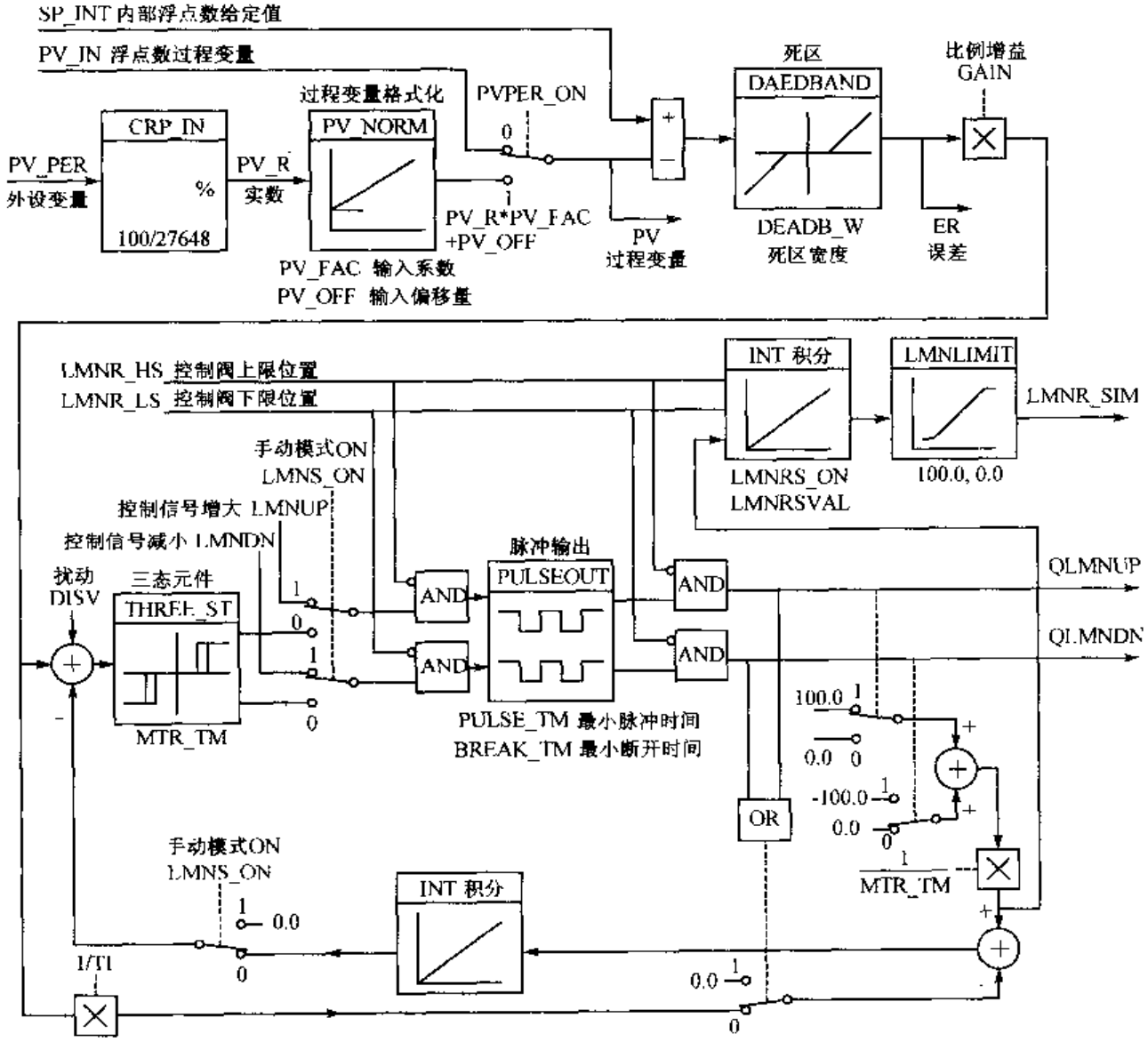


图 10-21　SFB 42“CONT _ S”框图

10.6.2 步进控制器的功能分析

1. 对设定值、过程变量和误差的处理

对设定值与过程变量的处理、误差的计算与处理、死区环节的作用与 SFB 41 的完全相同。有考参数的意义见表 10-7 和表 10-8。

2. PI 步进算法与脉冲的生成

SFB 没有使用阀门开度的实际位置反馈信号，对 PI 算法的积分和对模拟的位置反馈信号的积分使用同一个积分器(INT)。PI 控制器的输出值与模拟的位置反馈信号比较，比较的差值送给三级(three-step)元件和脉冲发生器(PULSEOUT)，后者生成用于执行机构的脉冲。可以通过调整三级元件的阈值来降低控制器的开关频率。

脉冲发生器 PULSEOUT 用来保证最小脉冲时间 PULSE _ TM 和最小断开时间 BREAK _ TM，以减小执行机构的磨损。输出脉冲 QLMNUP 或 QLMNDN 的脉冲宽度应大于 PULSE _ TM，两个脉冲之间断开的时间应大于 BREAK _ TM。

3. 手动模式

LMNS _ ON 为 1 时系统处于手动模式，三级元件后面的两个开关切换到上面标有“1”的触点的位置，此时开关量输出信号 QLMNUP 和 QLMNDN 受手动输入信号 LMNUP(控制信号增大)和 LMNDN(控制信号减小)的控制。LMNS _ ON 为 0 时控制开关返回自动模式，手动与自动的切换过程是平滑的。

4. 控制阀的极限位置保护

控制阀全开时，上限位开关动作，LMNR _ HS 信号为 1，通过图 10-21 中上面两个与门封锁输出量 QLMNUP，使伺服电动机停止开阀。控制阀全关时，下限位开关动作，LMNR _ LS 为 1，通过下面两个与门封锁输出量 QLMNDN，使伺服电动机停止关阀。

10.6.3 SFB 42 的参数

见表 10-9、10-10。

表 10-9 SFB 42 的输入参数

参数名称	数据类型	地址	说明	默认值
COM _ RST	BOOL	0.0	COMPLETE RESTART，为 1 时执行初始化程序，为 0 时控制器运行	FALSE
LMNR _ HS	BOOL	0.1	HIGH LIMIT SIGNAL OF REPEATED MANIPULATED VALUE，为 1 表示控制阀处于上限停止位置，即控制阀全开	FALSE
LMNR _ LS	BOOL	0.2	LOW LIMIT SIGNAL OF REPEATED MANIPULATED VALUE，为 1 表示控制阀处于下限停止位置，即控制阀全关	FALSE
LMNS _ ON	BOOL	0.3	MANIPULATED SIGNALS ON，为 1 时为手动模式	TURE
LMNUP	BOOL	0.4	MANIPULATED SIGNALS UP，控制信号增大。为 1 时输出信号 QLMNUP 受 LMNUP 的控制	FALSE

（续）

参数名称	数据类型	地　址	说　明	默认值
LMNDN	BOOL	0.5	MANIPULATED SIGNALS DOWN，控制信号减小。为 1 时输出信号 QLMNDN 受 LMNDN 的控制	FALSE
PVPER_ON	BOOL	0.6	PROCESS VARIABLE PERIPHERY ON，使用外围设备输入的过程变量	FALSE
CYCLE	TIME	2	SAMPLE TIME，采样时间，两次块调用之间的时间，应大于等于 20 ms	T#1s
SP_INT	REAL	6	INTERNAL SETPOINT，内部设定值输入，取值范围为±100.0%或物理值	0.0
PV_IN	REAL	10	PROCESS VARIABLE IN，浮点数格式的过程变量输入	0.0
PV_PER	WORD	14	PROCESS VARIABLE PERIPHERY I/O，外围设备输入的过程变量实际值	16#0000
GAIN	REAL	16	PROPORTIONAL GAIN，控制器的比例增益，冷却控制时应为负值	2.0
TI	TIME	20	RESET TIME，积分时间常数，应大于等于参数 CYCLE	T#20s
DEADB_W	REAL	24	DEAD BAND WIDTH，死区宽度，误差变量死区带的大小，取值范围 0.0～100.0 %或物理值	1.0
PV_FAC	REAL	28	PROCESS VARIABLE FACTOR，输入的过程变量的系数	1.0
PV_OFF	REAL	32	PROCESS VARIABLE OFFSET，输入的过程变量的偏移量	0.0
PULSE_TM	TIME	36	MINIMUM PULSE TIME，最小脉冲时间，应大于等于 CYCLE，多个循环积分	T#3s
BREAK_TM	TIME	40	MINIMUM BREAK TIME，最小断开时间，应大于等于 CYCLE，多个循环积分	T#3s
MTR_TM	TIME	44	MOTOR MANIPULATED VALUE，执行机构从一个限位位置移动到另一个限位位置所需的时间，应大于等于 CYCLE	T#30s
DISV	REAL	48	DISTURBANCE VARIABLE，反馈控制中的扰动输入变量，±100.0%或物理值	0.0

表 10-10　SFB 42 的输出参数

参数名称	数据类型	地　址	说　明	默认值
QLMNUP	BOOL	52.0	MANIPULATED SIGNAL UP，为 1 时控制信号增大	FALSE
QLMNDN	BOOL	52.1	MANIPULATED SIGNAL DOWN，为 1 时控制信号减小	FALSE
PV	REAL	54	PROCESS VARIABLE，格式化的过程变量输出	0.0
ER	REAL	58	ERROR SIGNAL，死区处理后的误差输出	0.0

10.7 PID 控制的示例程序

10.7.1 示例程序的下载与安装

在西门子的自动化网站 www.ad.siemens.de 可以下载“sample programs for CPU 31xC Technological Functions V1.0”。解压缩后安装该软件，在“\STEP7\EXAMPLES”文件夹中将产生 4 个项目，其中的“ZEn26_04_TF-31xC_PID”是 PID 控制的示例程序。

在 STEP 7 中打开该示例程序后，需要生成一个“Station”对象，在 HW Config 中根据 PLC 的硬件结构对硬件进行组态，并将硬件组态下载到 CPU 中。

ZEn26_04_TF-31xC_PID 中有 3 个程序：

(1) Controlling 1 CONT_S：带被控对象仿真的步进(Step)控制器示例程序。

(2) Controlling 2 CONT_C：带被控对象仿真的连续控制器示例程序。

(3) Controlling 3 PULSEGEN：带被控对象仿真的连续控制器与 PULSEGEN 脉冲发生器示例程序。

ZEn26_04_TF-31xC_PID 帮助使用者用简单的方法生成 PID 控制程序，并且用功能块 FB 100 模拟实际的执行机构和被控对象，不需要模拟量输入模块、模拟量输出模块和 PLC 外部的执行机构和被控对象，就可以模拟闭环控制系统的硬件环境。使用者可以修改 PID 控制器的参数，在 CPU 中运行程序，观察到闭环控制的效果。该例程对于学习 PID 控制功能块 SFB 41～SFB 43 的组态和使用方法，以及掌握 PID 控制器的参数整定方法是非常方便的。

10.7.2 使用连续控制器的示例程序

1. 模拟被控对象的 3 阶环节 PT3

例程 Controlling 2 中的闭环仿真系统由 PID 连续控制器 SFB 41“CONT_C”和模拟被控对象的功能块 FB100“PROC_C”组成(见图 10-22)。

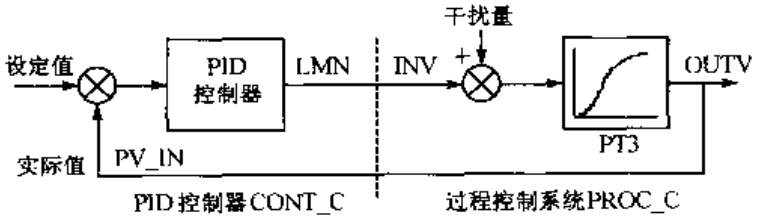


图 10-22 闭环仿真系统的结构

PROC_C 是一个 3 阶环节，其比例增益为 GAIN，3 个串联的惯性环节的时间常数分别为 TM_LAG1～TM_LAG3。其传递函数为

$$\frac{\text{GAIN}}{(\text{TM\_LAG1}s+1)(\text{TM\_LAG2}s+1)(\text{TM\_LAG3}s+1)}$$

将某一时间常数设置为 0，可以减少被控对象的阶数。

2. 连续控制器的示例程序

“Controlling 2 CONT_C”的主体程序是组织块 OB100 和 OB35。起动 PLC 时 CPU 自动

执行 OB100。在 HW Config 中将定时循环中断组织块 OB 35 的时间间隔设置为 100 ms,程序运行后每隔 100 ms 将自动调用一次 OB35。

在 OB100 和 OB35 中用语句 CALL "CONT _ C" , "DI _ CONT _ C"调用连续控制器 SFB 41"CONT _ C",背景数据块为 DI _ CONT _ C(DB101)。用语句 CALL "PROC _ C" , "DI _ PROC _ C"调用模拟被控对象的 FB100(PROC _ C),背景数据块为 DI _ PROC _ C(DB100)。

在 OB100 结束时用下面的语句复位 CONT _ C 和 PROC _ C 的背景数据块中的起动标志位:

```
CLR
=          "DI _ CONT _ C ".COM _ RST
=          "DI _ PROC _ C ".COM _ RST
```

在示例程序中应生成一个新的站(Station)对象,将程序 Controlling 2 "CONT _ C"中的块和符号表拷贝到新建的站中。

在示例程序中设置 CONT _ C 和 CONT _ C 中的参数后,将程序下载到 CPU,运行程序时用变量表 VAT 修改闭环控制系统的设定值 SP _ INT,监视闭环系统输出量 OUTV,即被控变量的实际值,可以观察到 PID 控制的效果。在 CONT _ C 中,干扰量 DISV 与控制器的输出量 LMN 相加,可以手动设置干扰量,观察干扰量变化时系统的响应。

步进 PID 控制器示例程序 Controlling 1 CONT _ S 的程序结构与使用方法与 Controlling 1 CONT _ C 的基本上相同,只是将 SFB 41 换成了 SFB42"CONT _ S"(步进 PID 控制器)。

3. **使用连续控制器与脉冲发生器的示例程序**

"Controlling 3 PULSEGEN"中的闭环仿真系统的结构如图 10-23 所示,PID 控制的输出量 LMN 被转换为两路脉冲,该实例程序的使用方法与"Controlling 2 CONT _ C"的类似。

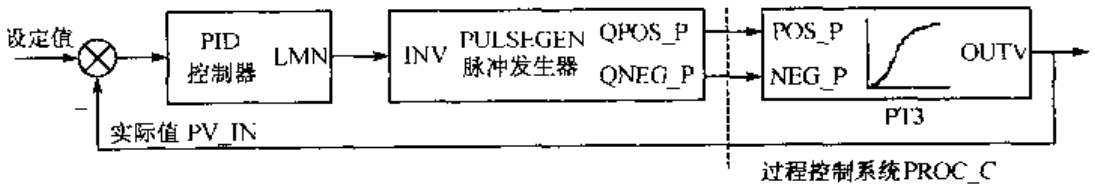


图 10-23 闭环仿真系统的结构

10.8 PID 控制器的参数整定方法

PID 控制器有 4 个主要的参数 T_s、K_C、T_I 和 T_D 需要整定,这些参数的取值对控制效果的影响非常大。在整定参数时首先应了解 PID 参数与系统动态、静态性能之间的定性关系。

10.8.1 PID 控制器的参数与系统动静态性能的关系

在 P、I、D 这三种控制作用中,比例部分与误差信号在时间上是一致的,只要误差一出现,比例部分就能及时地产生与误差成正比的调节作用,具有调节及时的特点。比例系数 K_P 越大,比例调节作用越强,系统的稳态精度越高;但是对于大多数系统,K_P 过大会使系统的输出量振荡加剧,稳定性降低。

控制器中的积分作用与当前误差的大小和误差的历史情况都有关系，只要误差不为零，控制器的输出就会因积分作用而不断变化，一直要到误差消失，系统处于稳定状态时，积分部分才不再变化，因此积分部分可以消除稳态误差，提高控制精度。但是积分作用的动作缓慢，可能给系统的动态稳定性带来不良影响，因此很少单独使用。

积分时间常数 T_I 增大时，积分作用减弱，系统的动态性能（稳定性）可能有所改善，但是消除误差的速度减慢。

根据误差变化的速度（即误差的微分），微分部分提前给出较大的调节作用。微分部分反映了系统变化的趋势，它较比例调节更为及时，所以微分部分具有超前和预测的特点。微分时间常数 T_D 增大时，超调量减小，动态性能得到改善，但是抑制高频干扰的能力下降。如果 T_D 过大，系统输出量在接近稳态值时可能上升缓慢。

选取采样周期 T_S 时，应使它远远小于系统阶跃响应的纯滞后时间或上升时间。为使采样值能及时反映模拟量的变化，T_S 越小越好。但是 T_S 太小会增加 CPU 的运算工作量，相邻两次采样的差值几乎没有什么变化，所以也不宜将 T_S 取得过小。表 10-11 给出了采样周期的经验数据。

表 10-11　采样周期的经验数据

被控制量	流　量	压　力	温　度	液　位	成　分
采样周期(s)	1～5	3～10	15～20	6～8	15～20

10.8.2　确定 PID 控制器参数初值的工程方法

在调节 PID 的参数时，首先需要确定控制器参数的初始值，如果预选的参数初始值与理想的参数值相差甚远（甚至可能相差几个数量级），将给参数调试带来很大的困难。因此如何选择一组较好的 PID 参数的初始值是 PID 参数整定中的关键问题。

下面介绍一种工程中广泛应用的扩充响应曲线法，用这种方法可以初步确定 PID 控制器的参数。具体方法如下：

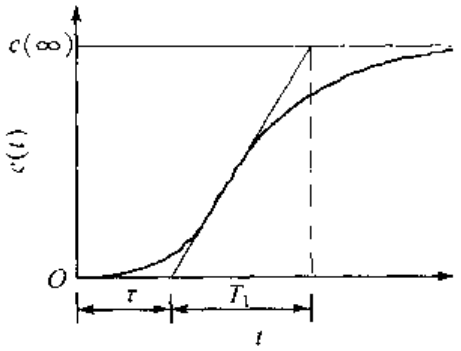


图 10-24　被控对象的阶跃响应曲线

（1）断开系统的反馈，令 PID 控制器为 $K_P=1$ 的比例控制器，在系统输入端加一个阶跃给定信号，测量并画出广义被控对象（包括执行机构）的开环阶跃响应曲线。绝大多数被控对象的响应曲线如图 10-24 所示，图中的 $c(\infty)$ 是系统输出量的稳态值。

（2）在曲线上最大斜率处作切线，求得被控对象的纯滞后时间 τ 和上升时间常数 T_1。

（3）求出系统的控制度。所谓控制度，是指计算机直接数字控制（简称 DDC）与模拟控制器的控制效果之比。控制效果一般用误差平方的积分值函数来表示，即

$$\text{控制度}=\frac{\left[\int_0^\infty e^2(t)\,dt\right]_{DDC}}{\left[\int_0^\infty e^2(t)\,dt\right]_{模拟}}$$

当控制度为 1.05 时，认为二者控制效果相当。

(4) 根据求出的 τ，T_1 和控制度的值，查表 10-12，求得 PID 控制器的 K_P、T_I、T_D 和 T_S，采样周期 T_S 也可以参考表 10-11 选取。

用以上方法确定的 4 个参数只能作为初步的参考值，为了获得良好的控制效果，还需要作闭环调试，根据闭环阶跃响应的特征，反复修改控制参数，使系统达到相对最佳的控制效果。

如果求控制度有困难，例如没有模拟控制系统，可以分别使用表 10-12 中不同控制度的几组参数，检验它们的控制效果，选取一组较好的参数作为控制器的初始值。

表 10-12　扩充响应曲线法参数整定表

控制度	控制方式	K_P	T_I	T_D	T_S
1.05	PI PID	$0.84T_1/\tau$ $1.15T_1/\tau$	3.4τ 2.0τ	— 0.45τ	0.1τ 0.05τ
1.2	PI PID	$0.78T_1/\tau$ $1.0\ T_1/\tau$	3.6τ 1.9τ	— 0.55τ	0.2τ 0.16τ
1.5	PI PID	$0.68\ T_1/\tau$ $0.85\ T_1/\tau$	3.9τ 1.62τ	— 0.65τ	0.5τ 0.34τ
2.0	PI PID	$0.57\ T_1/\tau$ $0.6\ T_1/\tau$	4.2τ 1.5τ	— 0.82τ	0.8τ 0.6τ

附　录

附录 A　S7-300/400 的指令一览表

附表 A-1　S7-300/400 的指令一览表

指令助记符	说　明
+	累加器 1 的内容与 16 位或 32 位整数常数相加,运算结果在累加器 1 中
=	赋值
)	右括号
+AR1	AR1 的内容加上累加器 1 中的地址偏移量,结果存放在 AR1 中
+AR2	AR2 的内容加上累加器 1 中的地址偏移量,结果存放在 AR2 中
+D	将累加器 1,2 中的双整数相加,双整数运算结果在累加器 1 中
-D	累加器 2 中的双整数减去累加器 1 中的双整数,双整数运算结果在累加器 1 中
*D	将累加器 1,2 中的双整数相乘,32 位双整数运算结果在累加器 1 中
/D	累加器 2 中的双整数除以累加器 1 中的双整数,32 位商在累加器 1 中,余数被丢掉
?D	比较累加器 2 和累加器 1 中的双整数是否==,<>,>,<,>=,<=,如果条件满足,RLO=1
+I	将累加器 1,2 低字中的整数相加,运算结果在累加器 1 的低字中
-I	累加器 2 低字中的整数减去累加器 1 低字中的整数,运算结果在累加器 1 的低字中
*I	将累加器 1,2 低字中的整数相乘,32 位双整数运算结果在累加器 1 中
/I	累加器 2 低字中的整数除以累加器 1 低字中的整数,商在累加器 1 的低字,余数在累加器 1 的高字
?I	比较累加器 2 和累加器 1 低字中的整数是否==,<>,>,<,>=,<=,如果条件满足,RLO=1
+R	将累加器 1,2 中的浮点数相加,浮点数运算结果在累加器 1 中
-R	累加器 2 中的浮点数减去累加器 1 中的浮点数,浮点数运算结果在累加器 1 中
*R	将累加器 1,2 中的浮点数相乘,浮点数乘积在累加器 1 中
/R	累加器 2 中的浮点数除以累加器 1 中的浮点数,浮点数商在累加器 1 中,余数被丢掉
?R	比较累加器 2 和累加器 1 中的浮点数是否==,<>,>,<,>=,<=,如果条件满足,RLO=1
A	AND,逻辑与,电路或触点串联
A(	逻辑与加左括号
ABS	求累加器 1 中的浮点数的绝对值
ACOS	求累加器 1 中的浮点数的反余弦函数
AD	将累加器 1 和累加器 2 中的双字的对应位相与,结果存放在累加器 1 中
AN	AND NOT,逻辑与非,常闭触点串联
AN(	AND NOT 加左括号
ASIN	求累加器 1 中的浮点数的反正弦函数

（续）

指令助记符	说　明
ATAN	求累加器 1 中的浮点数的反正切函数
AW	将累加器 1 和累加器 2 中的字的对应位相与，结果存放在累加器 1 的低字中
BE	块结束
BEC	块条件结束
BEU	块无条件结束
BLD<number>	程序显示指令，并不执行什么功能，只是用于编程设备(PG)的图形显示
BTD	将累加器 1 中的 7 位 BCD 码转换成双整数
BTI	将累加器 1 中的 3 位 BCD 码转换成整数
CAD	交换累加器 1 中 4 个字节的顺序
CALL	调用功能(FC)，功能块(FB)，系统功能(SFC)或系统功能块(SFB)
CAR	交换地址寄存器 1 和地址寄存器 2 中的数据
CAW	交换累加器 1 低字中两个字节的位置
CC	RLO＝1 时条件调用
CD	减计数器
CDB	交换共享数据块与背景数据块
CLR	清除 RLO(逻辑运算结果)
COS	求累加器 1 中的浮点数的余弦函数
CU	加计数
DEC	累加器 1 的最低字节减 8 位常数
DTB	将累加器 1 中的双整数转换成 7 位 BCD 码
DTR	将累加器 1 中的双整数转换成浮点数
ENT	进入累加器堆栈，仅用于 S7-400
EXP	求累加器 1 中的浮点数的自然指数
FN	下降沿检测
FP	上升沿检测
FR	使能计数器或使能定时器，允许定时器再启动
INC	累加器 1 的最低字节加 8 位常数
INVD	求累加器 1 中双整数的反码
INVI	求累加器 1 低字中的 16 位整数的反码
ITB	将累加器 1 中的整数转换成 3 位 BCD 码
ITD	将累加器 1 中的整数转换成双整数
JBI	BR＝1 时跳转
JC	RLO＝1 时跳转
JCB	RLO＝1 且 BR＝1 时跳转
JCN	RLO＝0 时跳转
JL	多分支跳转，跳步目标号在累加器 1 的最低字节

（续）

指令助记符	说　明
JM	运算结果为负时跳转
JMZ	运算结果小于等于 0 时跳转
JN	运算结果非 0 时跳转
JNB	RLO＝0 且 BR＝1 时跳转
JNBI	BR＝0 时跳转
JO	OV＝1 时跳转
JOS	OS＝1 时跳转
JP	运算结果为正时跳转
JPZ	运算结果大于等于 0 时跳转
JU	无条件跳转
JUO	指令出错时跳转，例如除数为 0、使用了非法的指令、浮点数比较时使用了非法的格式
JZ	运算结果为 0 时跳转
L<地址>	装入指令，将数据装入累加器 1，累加器 1 原有的数据装入累加器 2
L DBLG	将共享数据块的长度装入累加器 1
L DBNO	将共享数据块的编号装入累加器 1
L DILG	将背景数据块的长度装入累加器 1
L DINO	将背景数据块的编号装入累加器 1
L STW	将状态字装入累加器 1
LAR1	将累加器 1 的内容(32 位指针常数)装入地址寄存器 1
LAR1<D>	将 32 位双字指针<D>装入地址寄存器 1
LAR1 AR2	将地址寄存器 2 的内容装入地址寄存器 1
LAR2	将累加器 1 的内容(32 位指针常数)装入地址寄存器 2
LAR2<D>	将 32 位双字指针<D>装入地址寄存器 2
LC	定时器或计数器的当前值以 BCD 码的格式装入累加器 1
LEAVE	离开累加器堆栈，仅用于 S7-400
LN	求累加器 1 中的浮点数的自然对数
LOOP	循环跳转
MCR(	打开主控继电器区
)MCR	关闭主控继电器区
MCRA	起动主控继电器功能
MCRD	取消主控继电器功能
MOD	累加器 2 中的双整数除以累加器 1 中的双整数，32 位余数在累加器 1 中
NEGD	求累加器 1 中双整数的补码
NEGI	求累加器 1 低字中的 16 位整数的补码
NEGR	将累加器 1 中浮点数的符号位取反
NOP 0	空操作指令，指令各位全为 0

（续）

指令助记符	说　明
NOP 1	空操作指令，指令各位全为 1
NOT	将 RLO 取反
O	OR，逻辑或，电路或触点并联
O(	逻辑或加左括号
OD	将累加器 1 和累加器 2 中的双字的对应位相或，结果存放在累加器 1 中
ON	OR NOT，逻辑或非，常闭触点并联
ON(	OR NOT 加左括号
OPN	打开数据块
OW	将累加器 1 和累加器 2 中的低字的对应位相或，结果存放在累加器 1 的低字
POP	出栈，堆栈由累加器 1，2(S7-300)或累加器 1～4(S7-400)组成
PUSH	入栈，堆栈由累加器 1，2(S7-300)或累加器 1～4(S7-400)组成
R	RESET，复位指定的位或定时器、计数器
RET	条件返回
RLD	累加器 1 中的双字循环左移
RLDA	累加器 1 中的双字通过 CC1 循环左移
RND	将浮点数转换为四舍五入的双整数
RND-	将浮点数转换为小于等于它的最大双整数
RND+	将浮点数转换为大于等于它的最小双整数
RRD	累加器 1 中的双字循环右移
RRDA	累加器 1 中的双字通过 CC1 循环右移
S	SET，将指定的位置位，或设置计数器的预置值
SAVE	将状态字中的 RLO 保存到 BR 位
SD	接通延时定时器
SE	扩展的脉冲定时器
SET	将 RLO 置位为 1
SF	断开延时定时器
SIN	求累加器 1 中的浮点数的正弦函数
SLD	将累加器 1 中的双字逐位左移指定的位数，空出的位添 0，移位位数在指令中或在累加器 2 中
SLW	将累加器 1 低字中的 16 位字逐位左移指定的位数，空出的位添 0，移位位数在指令中或在累加器 2 中
SP	脉冲定时器
SQR	求累加器 1 中的浮点数的平方
SQRT	求累加器 1 中的浮点数的平方根
SRD	将累加器 1 中的双字逐位右移指定的位数，空出的位添 0，移位位数在指令中或在累加器 2 中
SRW	将累加器 1 低字中的 16 位字逐位右移指定的位数，空出的位添 0，移位位数在指令中或在累加器 2 中
SS	保持型接通延时定时器
SSD	将累加器 1 中的有符号双整数逐位右移指定的位数，空出的位添上与符号位相同的数

（续）

指令助记符	说　明
SSI	将累加器 1 低字中的有符号整数逐位右移指定的位数，空出的位添上与符号位相同的数
T<地址>	传送指令，将累中器 1 的内容写入目的存储区，累加器 1 的内容不变
T STW	将累加器 1 中的内容传送到状态字
TAK	交换累加器 1，2 的内容
TAN	求累加器 1 中的浮点数的正切函数
TAR1	将地址寄存器 1 的数据传送到累加器 1，累加器 1 中的数据保存到累加器 2
TAR1<D>	将地址寄存器 1 的内容传送到 32 位指针<D>
TAR1 AR2	将地址寄存器 1 的内容传送到地址寄存器 2
T AR2	将地址寄存器 2 的数据传送到累加器 1，累加器 1 中的数据保存到累加器 2
TAR2<D>	将地址寄存器 2 的内容传送到 32 位指针<D>
TRUNC	将浮点数转换为截位取整的双整数
UC	无条件调用
X	XOR，逻辑异或，两个逻辑变量的状态相反时运算结果为 1
X(	逻辑异或加左括号
XN	XOR NOT，逻辑异或非，两个逻辑变量的状态相同时运算结果为 1
XN(	XOR NOT 加左括号
XOD	将累加器 1 和累加器 2 中的双字的对应位相异或，结果存放在累加器 1 中
XOW	将累加器 1 和累加器 2 中的字的对应位相异或，结果存放在累加器 1 的低字

附录 B　组织块、系统功能与系统功能块一览表

附表 B-1　组织块一览表

OB 编号	启 动 事 件	默认优先级	说　明
OB1	启动或上一次循环结束时执行 OB1	1	主程序循环
OB10～OB17	日期时间中断 0～7	2	在设置的日期和时间起动
OB20～OB23	时间延迟中断 0～3	3～6	延时后起动
OB30～OB38	循环中断 0～8，默认的时间间隔分别为 5s，2s，1s，500ms，200ms，100ms，50ms，20ms 和 10ms	7～15	以设定的时间为周期运行
OB40～OB47	硬件中断 0～7	16　23	检测到来自外部模块的中断请求时起动
OB55	状态中断	2	DPV1 中 断（PROFIBUS-DP 中断）
OB56	刷新中断	2	
OB57	制造厂商特殊中断	2	
OB60	多处理器中断，调用 SFC35 时起动	25	多处理器中断的同步操作
OB61～64	同步循环中断 1～4	25	同步循环中断

（续）

OB 编号	启 动 事 件	默认优先级	说　　明
OB70	I/O 冗余错误	25	冗余故障中断，只用于 H 系列 CPU
OB72	CPU 冗余错误，例如一个 CPU 发生故障	28	
OB73	通信冗余错误中断，例如冗余连接的冗余丢失	25	
OB80	时间错误	26，起动时为 28	异步错误中断
OB81	电源故障	26，起动时为 28	
OB82	诊断中断	26，起动时为 28	
OB83	插入/拔出模块中断	26，起动时为 28	
OB84	CPU 硬件故障	26，起动时为 28	
OB85	优先级错误	26，起动时为 28	
OB86	扩展机架、DP 主站系统或分布式 I/O 站故障	26，起动时为 28	
OB87	通信故障	26，起动时为 28	
OB88	过程中断	28	
OB90	冷、热启动、删除块或背景循环	29	背景循环
OB100	暖启动	27	启动
OB101	热启动	27	
OB102	冷启动	27	
OB121	编程错误	与引起中断的 OB 有相同的优先级	同步错误中断
OB122	I/O 访问错误		

注：优先级 29 相当于 0.29，即背景循环具有最低的优先权。

附表 B-2　系统功能（SFC）一览表

SFC 编号	SFC 名称	说　　明
SFC0	SET _ CLK	设置系统时钟
SFC1	READ _ CLK	读取系统时钟
SFC2	SET _ RTM	设置运行时间定时器
SFC3	CTRL _ RTM	启动/停止运行时间定时器
SFC4	READ _ RTM	读取运行时间定时器
SFC5	GADR _ LGC	查询通道的逻辑地址
SFC6	RD _ SINFO	读取 OB 的起动信息
SFC7	DP _ PRAL	触发 DP 主站的硬件中断
SFC9	EN _ MSG	激活与块相关、符号相关和组状态的信息
SFC10	DIS _ MSG	禁止与块相关、符号相关和组状态的信息
SFC11	SYC _ FR	同步或锁定 DP 从站组
SFC12	D _ ACT _ DP	激活或取消 DP 从站
SFC13	DPNRM _ DG	读取 DP 从站的诊断信息（从站诊断）
SFC14	DPRD _ DAT	读标准 DP 从站的一致性数据
SFC15	DPWR _ DAT	写标准 DP 从站的一致性数据
SFC17	ALARM _ SQ	生成可应答的与块相关的报文
SFC18	ALARM _ S	生成永久性的可应答的与块相关的报文

（续）

SFC 编号	SFC 名称	说　明
SFC19	ALARM _ SC	查询最后的 ALARM _ SQ 状态报文的应答状态
SFC20	BLKMOV	复制多个变量
SFC21	FILL	初始化存储器
SFC22	CREAT _ DB	生成一个数据块
SFC23	DEL _ DB	删除一个数据块
SFC24	TEST _ DB	测试一个数据块
SFC25	COMPRESS	压缩用户存储器
SFC26	UPDAT _ PI	刷新过程映像输入表
SFC27	UPDAT _ PO	刷新过程映像输出表
SFC28	SET _ TINT	设置实时钟中断
SFC29	CAN _ TINT	取消实时钟中断
SFC30	ACT _ TINT	激活实时钟中断
SFC31	QRY _ TINT	查询实时钟中断的状态
SFC32	SRT _ DINT	启动延迟中断
SFC33	CAN _ DINT	取消延迟中断
SFC34	QRY _ DINT	查询延迟中断
SFC35	MP _ ALM	触发多 CPU 中断
SFC36	MSK _ FLT	屏蔽同步错误
SFC37	DMSK _ FLT	解除对同步错误的屏蔽
SFC38	READ _ ERR	读错误寄存器
SFC39	DIS _ IRT	禁止新的中断和异步错误处理
SFC40	EN _ IRT	允许新的中断和异步错误处理
SFC41	DIS _ AIRT	延迟高优先级的中断和异步错误处理
SFC42	EN _ AIRT	允许高优先级的中断和异步错误处理
SFC43	RE _ TRIGR	重新触发扫描时间监视
SFC44	REPL _ VAL	将替换值传送到累加器 1 中
SFC46	STP	将 CPU 切换到 STOP 模式
SFC47	WAIT	延迟用户程序的执行
SFC48	SNC _ RTCB	同步从站的实时钟
SFC49	LGC _ GADR	查询一个逻辑地址的插槽和机架
SFC50	RD _ LGADR	查询模块所有的逻辑地址
SFC51	RDSYSST	读取系统状态表或局部系统状态表
SFC52	WR _ USMSG	将用户定义的诊断事件写入诊断缓冲器
SFC54	RD _ PARM	读定义的参数
SFC55	WR _ PARM	写入动态参数
SFC56	WR _ DPARM	写入默认的参数

（续）

SFC 编号	SFC 名称	说　明
SFC57	PARM _ MOD	指定模块的参数
SFC58	WR _ REC	写入一个数据记录
SFC59	RD _ REC	读取一个数据记录
SFC60	GD _ SND	发送 GD(全局数据)包
SFC61	GD _ RCV	接收全局数据包
SFC62	CONTROL	查询属于 S7-400 的本地通信 SFB 背景的连接状态
SFC63	AB _ CALL	调用汇编代码块
SFC64	TIME _ TCK	读取系统时间
SFC65	X _ SEND	将数据发送到局域 S7 站外的一个通信伙伴
SFC66	X _ RCV	接收局域 S7 站外的一个通信伙伴的数据
SFC67	X _ GET	读取局域 S7 站外的一个通信伙伴的数据
SFC68	X _ PUT	将数据写入局域 S7 站外的一个通信伙伴
SFC69	X _ ABORT	中止与局域 S7 站外的一个通信伙伴的连接
SFC72	I _ GET	从局域 S7 站内的一个通信伙伴读取数据
SFC73	I _ PUT	将数据写入局域 S7 站内的一个通信伙伴
SFC74	I _ ABORT	中止与局域 S7 站内的一个通信伙伴的连接
SFC78	OB _ RT	确定 OB 程序的运行时间
SFC79	SET	置位输出范围
SFC80	RSET	复位输出范围
SFC81	UBLKMOV	不能中断的块传送
SFC82	CREA _ DBL	生成装载存储器中的数据块
SFC83	READ _ DBL	读取装载存储器中的一个数据块
SFC84	WRIT _ DBL	写入装载存储器中的一个数据块
SFC87	C _ DIAG	实际连接状态的诊断
SFC90	H _ CTRL	H 系统的控制操作
SFC100	SET _ CLKS	设置日期时间和日期时间状态
SFC101	RTM	处理运行时间计时器
SFC102	RD _ DPARA	重新定义参数
SFC103	DP _ TOPOL	识别 DP 主系统中的总线拓扑
SFC104	CiR	控制 CiR
SFC105	READ _ SI	读动态系统资源
SFC106	DEL _ SI	删除动态系统资源
SFC107	ALARM _ DQ	生成可应答的与块有关的报文
SFC108	ALARM _ D	生成永久的可应答的与块有关的报文
SFC126	SYNC _ PI	同步刷新过程映像输入表
SFC127	SYNC _ PO	同步刷新过程映像输出表

附表 B-3　系统功能块(SFB)一览表

SFB 编号	SFB 名称	说　明
SFB0	CTU	加计数
SFB1	CTD	减计数
SFB2	CTUD	加/减计数
SFB3	TP	生成一个脉冲
SFB4	TON	产生 ON 延迟
SFB5	TOF	产生 OFF 延迟
SFB8	USEND	不对等的数据发送
SFB9	URCV	不对等的数据接收
SFB12	BSEND	发送段数据
SFB13	BRCV	接收段数据
SFB14	GET	从远程 CPU 读数据
SFB15	PUT	向远程 CPU 写数据
SFB16	PRINT	发送数据到打印机
SFB19	START	初始化远程装置的暖起动或冷起动
SFB20	STOP	将远程装置切换到 STOP 状态
SFB21	RESUME	初始化远程装置的热起动
SFB22	STATUS	查询远程装置的状态
SFB23	USTATUS	接收远程装置的状态
SFB29	HS_COUNT	集成的高速计数器,仅用于 CPU 312 IFM 和 PU 314 IFM
SFB30	FREQ_MES	集成的频率计,仅用于 CPU 312 IFM 和 PU 314 IFM
SFB31	NOTIFY_8P	生成不带应答指示的与块相关的报文
SFB32	DRUM	实现一个顺序控制器
SFB33	ALARM	生成带应答指示的与块相关的报文
SFB34	ALARM_8	生成与 8 个信号值无关的与块相关的报文
SFB35	ALARM_8P	生成与 8 个信号值有关的与块相关的报文
SFB36	NOTIFY	生成不带应答显示的与块相关的报文
SFB37	AR_SEND	发送归档数据
SFB38	HSC_A_B	集成的 A/B 相高速计数器
SFB39	POS	集成的定位功能
SFB41	CONT_C	连续 PID 控制
SFB42	CONT_S	步进 PID 控制
SFB43	PULSEGEN	脉冲发生器
SFB44	ANALOG	使用模拟输出的定位,仅用于 S7-300C CPU
SFB46	DIGITAL	使用数字输出的定位,仅用于 S7-300C CPU
SFB47	COUNT	计数器控制,仅用于 S7-300C CPU
SFB48	FREQUENC	频率测量控制,仅用于 S7-300C CPU

（续）

SFB 编号	SFB 名称	说　明
SFB49	PULSE	脉冲宽度调制控制，仅用于 S7-300C CPU
SFB52	RDREC	从 DP 从站读数据记录
SFB53	WRREC	向 DP 从站写数据记录
SFB54	RALRM	从 DP 从站接收中断
SFB60	SEND _ PTP	发送数据（ASCⅡ协议或 3964(R)协议），仅用于 S7-300C CPU
SFB61	RCV _ PTP	接收数据（ASCⅡ协议或 3964(R)协议），仅用于 S7-300C CPU
SFB62	RES _ RCVB	删除接收缓冲区（ASCⅡ协议或 3964(R)协议），仅用于 S7-300C CPU
SFB63	SEND _ RK	发送数据（RK512 协议），仅用于 S7-300C CPU
SFB64	FETCH _ RK	获取数据（RK512 协议），仅用于 S7-300C CPU
SFB65	SERVE _ RK	接收/提供数据（RK512），仅用于 S7-300C CPU
SFB75	SALRM	向 DP 主站发送中断

附表 B-4　IEC 功能一览表

IECEC 名称	说　明
数据类型格式转换	
FC3 D _ TOD _ DT	将 DATE 和 TIME _ OF _ DAY 数据类型的数据合并为 DT（日期时间）格式的数据
FC6 DT _ DATE	从 DT 格式的数据中提取 DATE（日期）数据
FC7 DT _ DAY	从 DT 格式的数据中提取星期值数据
FC8 DT _ TOD	从 DT 格式的数据中提取 TIME _ OF _ DAY（实时时间）数据
FC33 S5TI _ TIM	将数据类型 S5TIME（S5 格式的时间）转换为 TIME
FC40 TIM _ S5TI	将数据类型 TIME 转换为 S5TIME
FC16 I _ STRNG	将数据类型 INT（整数）转换为 STRING（字符串）
FC5 DI _ STRNG	将数据类型 DINT（双整数）转换为 STRING
FC30 R _ STRNG	将数据类型 REAL（浮点数）转换为 STRING
FC38 STRNG _ I	将数据类型 STRING 转换为 INT
FC37 STRNG _ DI	将数据类型 STRING 转换为 DINT
FC39 STRNG _ R	将数据类型 STRING 转换为 REAL
比较 DT（日期时间）	
FC9 EQ _ DT	DT 等于比较
FC12 GE _ DT	DT 大于等于比较
FC14 GT _ DT	DT 大于比较
FC18 LE _ DT	DT 小于等于比较
FC23 LT _ DT	DT 小于比较
FC28 NE _ DT	DT 不等于比较
字符串变量比较	
FC10 EQ _ STRNG	字符串等于比较

（续）

IECEC 名称	说　明
FC13 GE _ STRNG	字符串大于等于比较
FC15 GT _ STRNG	字符串大于比较
FC19 LE _ STRNG	字符串小于等于比较
FC24 LT _ STRNG	字符串小于比较
FC29 NE _ STRNG	字符串不等于比较
字符串变量编辑	
FC21 LEN	求字符串变量的长度
FC20 LEFT	提供字符串左边的若干个字符
FC32 RIGHT	提供字符串右边的若干个字符
FC26 MID	提供字符串中间的若干个字符
FC2 CONCAT	将两个字符串合并为一个字符串
FC17 INSERT	在一个字符串中插入另一个字符串
FC4 DELETE	删除字符串中的若干个字符
FC31 REPLACE	用一个字符串替换另一个字符串中的若干个字符
FC11 FIND	求一个字符串在另一个字符串中的位置
Time _ of _ Day 功能	
FC1 AD _ DT _ TM	将一个 Time 格式的持续时间与 DT 格式的时间相加，产生一个 DT 格式的时间
FC35 SB _ DT _ TM	将一个 Time 格式的持续时间与 DT 格式的时间相减，产生一个 DT 格式的时间
FC34 SB _ DT _ DT	将两个 DT 格式的时间相减，产生一个 Time 格式的持续时间
数值编辑	
FC22 LIMIT	将变量的数值限制在指定的极限值内
FC25 MAX	在 3 个变量中选取最大值
FC27 MIN	在 3 个变量中选取最小值
FC36 SEL	根据选择开关的值在两个变量中选择

附录 C　光盘说明

光盘中后缀为 PDF 的文件（*.PDF）需要用 Adobe 阅读器阅读。

C.1　软件手册

S7-300 指令列表 CPU 技术参数 CPU 312C-314C-2 DP/PtP，2001

Function Block Diagram（FBD）for S7-300 and S7-400 Programming Reference Manual，2002

Ladder Logic（LAD）for S7-300 and S7-400 Programming Reference Manual，2002

PID Temperature Control Manual，2002

Programming with STEP 7 V5.2 Manual，2003

S7 Graph V5.2 for S7-300/400 Programming Sequential Control Systems Manual，2002

Sample Programs for Technological Functions, 2001
S7-400 Instruction List CPU 412,414,416,417, 2002
S7-PLCSIM V5.2 User Manual, 2002
S7-300/400 的系统软件和标准功能,参考手册,2002
S7-SCL V5.1 for S7-300/S7-400 Manual, 2002
Standard Software for S7-300 and S7-400 Standard Functions Part 2 Reference Manual, 2002
Standard PID Control Manual, 2003
Statement List(STL)for S7-300 and S7-400 Programming Reference Manual, 2002
STEP 7 V5.0 使用入门,1998
STEP 7 V5.1 编程使用手册,1998
Working with STEP 7 V5.2 Getting Started, 2002
System Software for S7-300/400 System and Standard Functions Reference Manual, 2002
S7-300/400 语句表(STL)编程参考手册,2004
S7-300/400 梯形图(LAD)编程参考手册,2004

C.2 硬件手册

Configuring Hardware and Communication Connections STEP 7 V5.2 Manual, 2002
S7-300 Hardware and Installation: CPU 31xC and CPU 31x Installation Manual, 2003
S7-300 Automation System CPU Specifications: CPU 31xC and CPU 31x Reference Manual, 2003
CPU 31xC Technological Functions Manual, 2003
S7-300 可编程控制器 CPU 312C 至 314C-2DP/PtP CPU 技术参数 参考手册,2001
S7-300 自动化系统 CPU 31xC 技术功能 使用手册,2001
Modifying the System during Operation via CiR Manual, 2002
Programmable Controllers S7 F/FH Systems Manual, 2003
S7-300 Automation System, Hardware and Installation: CPU 312IFM-318-2 DP,2003
S7-300 and M7-300 Programmable Controllers Module Specifications Reference Manual, 2001
S7-300 和 M7-300 可编程序控制器模板规范 参考手册,2003
S7-300 可编程序控制器产品目录,2003
Automation System S7-400 CPU Specifications Reference Manual, 2002
S7-400 可编程控制器 CPU 及模板规范手册,2003
S7-400 and M7-400 Programmable Controllers Hardware and Installation Manual, 1999
S7-400 产品目录,2003
S7-400,M7-400 Programmable Controllers Module Specifications Reference Manual, 2003
Automation System S7-400H Fault-tolerant Systems Manual, 2003
自动化系统 S7-400 容错系统 使用手册,2002

C.3 通信手册

AS-Interface-Introduction and Basic Information Manual,1999

CP 340 Point-to-Point Commuinication Installation and Parameter Assignment Manual, 2002
CP 341 Point-to-Point Communication Installation and Parameter Assignment Manual, 2000
Point-to-point connection CP 440 Installation and Parameter Assignment Manual, 2000
Point-to-Point Communication CP 441 Installation and Parameter Assignment Manual, 2000
CP 343－2/CP 343－2 P AS-Interface Master Manual
NCM S7 for PROFIBUS Primer, 2002
NCM S7 for PROFIBUS Manual Volume 1 of 2, 2002
NCM S7 for PROFIBUS/FMS Manual Volume 2/2, 2001
Prodave Operating Instructions, 2004
Profibus PA应用技术手册
SIMATIC NET 网络解决方案　工业以太网交换机与连接模块,2002
SIMATIC NET 工业通讯的 IT 解决方案,2002
工业通讯及现场设备产品目录 IK PI,2001
用于自动控制系统的工业通讯网络,2001

C.4　例程

\ 例程 \ 第 3 章例题

FC1:例 3-5,英寸转换为厘米;FC2:例 3-8,求 5 的立方;FC3:例 3-9,用较大的数减去较小的数;FC4:例 3-10,浮点数运算与堆栈应用;FC5:例 3-11,条件跳转指令应用;FC6:例 3-12,用循环指令求 5 的阶乘;FC7:定时器计数器应用。

\ 例程 \ 第 5 章例题

FC1:图 5-10 动力滑台程序;FC2:图 5-21 动力滑台顺序控制程序;FC3:图 5-23 使用起保停电路的选择序列与并行序列的程序;FC4:图 5-27 物料混合控制系统的程序;FC5:图 5-29 工作台旋转控制系统的程序;FC6:图 5-30 使用 SR 指令的选择序列与并行序列的程序;TC7:图 5-32 组合钻床顺序控制系统的程序。

\ 例程 \ 机械手控制:用梯形图编写的 5.5 节中的机械手控制程序。

\ 例程 \ 机械手 GRAPH:用 S7 Graph 编写的 5.6 节中的机械手控制程序。

\ 例程 \ 运输带控制:用 S7 Graph 编写的 5.6 节中的运输带控制程序。

\ 例程 \ 发动机控制

\ 例程 \ 多重背景

\ 例程 \ OB10 例程

\ 例程 \ OB20 例程

\ 例程 \ OB35 例程

\ 例程 \ OB40 例程

\ 例程 \ MPI 全局数据通信

\ 例程 \ DP 主从通信

\ 例程 \ DP 直接数据交换 DX

\ 例程 \ 同步与锁定

C.5 软件

S7-31xC sample _ programs.exe
Setup _ PtP.exe
NCM _ S7 _ for _ Industrial _ Ethernet _ V52 _ + SPI.zip
NCM _ S7 _ for _ PROFIBUS _ V52 _ + SPI.zip
STEP7 _ V5 _ 1 _ SP6.zip
STEP7 _ V52 _ SP1.zip
S7-300 软件升级检测工具_ E.zip
S7 _ PLCSIM _ V5.0.zip
S7 _ PLCSIM _ V5 _ 0 _ SP1.zip
S7 _ GRAPH _ V5 _ 2 _ SP1.zip
S7 _ GRAPH _ V52 _ SP3.zip
S7 _ SCL _ V5 _ 1 _ SP3.zip
S7 _ HIGRAPH _ V5 _ 1 _ SP1.zip
串口调试软件 .ZIP:作者编写的调试 PLC 与计算机通信的软件,使用方法见软件中的 HELP。

附录 D 常用缩写词

ACCU:累加器
AI/AO:模拟量输入/模拟量输出。
AR1/AR2:地址寄存器 1/地址寄存器 2。
AS-i:执行器传感器接口,一种现场总线。
BCD 码:二进制编码的十进制数。
B:字节(Byte)
B 堆栈:块堆栈。
BOOL:布尔变量,或称开关量、数字量。
C7:由 S7-300、操作面板、I/O、通信和过程监控系统组成的控制装置。
C#:计数器常数(BCD 码)。
CC:中央机架,或称中央控制器。
CiR:RUN 模式时修改系统的设置。
CP:Communications Processor,通信处理器。
CPU:Central Processing Unit,中央处理单元,CPU 模块的简称。
C 总线:通信总线。
DB:Data Block,数据块。
DI:背景数据块。
DI/DO:数字量输入/数字量输出。
DINT:32 位整数,双整数(Double Integer)。

DP:PROFIBUS-DP 的简称。
DPM1:PROFIBUS 中的 1 类 DP 主站,系统的中央控制器。
DPM2:PROFIBUS 中的 2 类 DP 主站,DP 网络中的编程、诊断和管理设备。
DPV1:DP-V1 的简称。
DP-V0、DP-V1 和 DP-V2:PROFIBUS-DP 的 3 个版本。
DW:双字(Double Word)。
EEPROM:可以电擦除的 EPROM。
EPROM:可擦除可编程的只读存储器。
EU:扩展单元,扩展机架。
FALSE:数字量的值(0)。
FB:功能块,带存储器的子程序。
FBD:功能块图。
FC:功能,不带存储器的子程序。
FDL:Fieldbus Data Link, PROFIBUS 的数据链路层。
FEPROM:Flash EPROM,快闪存储器。
FM:功能模块。
GD:用于 MPI 通信的全局数据。
GSD 文件:电子设备数据库文件。
h:小时
HMI:人机接口。
HW Config:集成在 STEP 7 中的硬件组态工具。
I/O:输入/输出。
IDB:背景数据块。
I 堆栈:中断堆栈。
IEC:国际电工委员会。
IM:接口模块。
INT:16 位有符号整数(Integer)。
K 总线:通信总线。
L#:32 位双整数常数。
LAD:梯形图。
LAN:局域网。
L 堆栈:局域堆栈。
LED:发光二极管。
M7-300/400:作为 CPU 或功能模块使用,具有 AT 兼容计算机的功能。
MAC:介质存取控制(Medium Access Control)。
MMC 卡:微存储器卡。
MPI:Multi Point Interface,多点接口。
OB:组织块,操作系统与用户程序的接口。
OB1:用于循环处理的组织块,用户程序中的主程序。

OP:操作员面板。
P#:地址指针常数,例如 P#M2.0 是 M2.0 的地址。
PC:Personal Computer,个人计算机。
PG:Programming Unit,编程器。
PI:外设输入存储区,可以通过它直接访问输入模块。
PLC:Programmable Logic Controller,可编程序控制器。
PLCSIM:STEP 7 的仿真软件。
PNO:PROFIBUS 用户组织。
PQ:外设输出存储区,可以通过它直接访问输出模块。
PRODAVE:用于 PC 与 SIMATIC PLC 通信的软件。
PROFIBUS:一种现场总线。
PtP:点对点通信。
P 总线:I/O 总线。
RAM:随机读写存储器。
REAL:实数,又称浮点数。
RLO:逻辑运算结果(Result of logic operation)。
ROM:只读存储器。
RUN:CPU 模块的运行模式。
s:秒
S5:西门子早期 PLC 的型号。
S5T#:16 位 S5 时间常数。
S7 Graph:STEP 7 的顺序功能图语言。
SDB:系统数据块
SFC:系统功能,集成在 CPU 模块中,通过它调用一些重要的系统功能,没有存储区。
SFB:系统功能块,集成在 CPU 模块中,通过它调用一些重要的系统功能,有存储区。
SIMATIC:SIEMENS AG(西门子自动化集团)的注册商标。
SM:信号模块,数字量输入/输出模块和模拟量输入/输出模块的总称。
STEP 7:S7-300/400 的编程软件。
STL:语句表。
STOP:CPU 的停止模式。
T#:带符号的 32 位 IEC 时间常数。
TOD#:32 位实时时间(Time of day)常数。
TP:触摸屏。
TRUE:数字量的值(1)。
UDT:用户定义的数据类型(user-defined data types)。
VAT:变量表。

参 考 文 献

1 阳宪惠．工业数据通信与控制网络．北京:清华大学出版社,2003
2 中国机电一体化技术应用协会译,运用 PROFIBUS-DP 实现分散自动化．北京
3 廖常初主编．PLC 编程及应用．北京:机械工业出版社,2002
4 廖常初主编．PLC 基础及应用．北京:机械工业出版社,2003
5 Siemens AG. Statement List(STL)for S7-300 and S7-400 Programming Reference Manual,2002
6 Siemens AG. Ladder Logic(LAD)for S7-300 and S7-400 Programming Reference Manual, 2002
7 Siemens AG. Function Block Diagram(FBD)for S7-300 and S7-400 Programming Reference Manual, 2002
8 Siemens AG. S7 Graph V5.2 for S7-300/400 Programming Sequential Control Systems Manual,2002
9 Siemens AG. S7-PLCSIM V5.2 User Manual, 2002
10 Siemens AG. Standard PID Control Manual, 2003
11 Siemens AG. Working with STEP 7 V5.2 Getting Started, 2002
12 Siemens AG. Programming with STEP 7 V5.2 Manual. 2003
13 Siemens AG. System Software for S7-300/400 System and Standard Functions Reference Manual, 2002
14 Siemens AG. Standard Software for S7-300 and S7-400 Standard Functions Part 2 Reference Manual, 2002
15 Siemens AG. Configuring Hardware and Communication Connections STEP 7 V5.2 Manual,2002
16 Siemens AG. Hardware and Installation:CPU 312IFM-318-2 DP,2003
17 Siemens AG. S7-300 and M7-300 Programmable Controllers Module Specifications Reference Manual, 2001
18 Siemens AG. CPU 31xC and CPU 31x Reference Manual, 2003
19 Siemens AG. CPU 31xC Technological Functions Manual, 2003
20 Siemens AG. CPU 31xC and CPU 31x Installation Manual, 2003
21 Siemens AG. S7-400 and M7-400 Programmable Controllers Hardware and Installation Manual, 1999
22 Siemens AG. Automation System S7-400 CPU Specifications Reference Manual, 2002
23 Siemens AG. S7-400,M7-400 Programmable Controllers Module Specifications Reference Manual, 2003
24 PROFIBUS Trade Organization PTO. PROFIBUS System Description, Karlsruhe germany: 2002
25 Siemens AG. CP 340 Point-to-Point Communication Installation and Parameter Assignment Manual, 2002
26 Siemens AG. CP 341 Point-to-Point Communication Installation and Parameter Assignment Manual, 2000
27 Siemens AG. AS-Interface－Introduction and Basic Information Manual
28 Siemens AG. Prodave Operating Instructions, 2004
29 Siemens AG. STEP 7 V5.0 使用入门,1998
30 Siemens AG. STEP 7 V5.1 编程使用手册,1998
31 Siemens AG. S7-300/400 的系统软件和标准功能　参考手册,2002
32 Siemens AG. S7-300 可编程控制器 CPU 312C 至 314C-2DP/PtP CPU 技术参数　参考手册,2001
33 Siemens AG. S7-300 自动化系统 CPU 31xC 技术功能　使用手册,2001
34 Siemens AG. S7-300 和 M7-300 可编程序控制器模板规范　参考手册,2003

35 Siemens AG. S7-300 可编程序控制器产品目录,2003
36 Siemens AG. S7-400 可编程控制器 CPU 及模板规范手册,2003
37 Siemens AG. S7-400 产品目录,2003
38 Siemens AG. 工业通讯及现场设备产品目录 IK PI,2001
39 Siemens AG. 用于自动控制系统的工业通讯网络,2001
40 Siemens AG. SIMATIC NET 网络解决方案,工业以太网交换机与连接模块,2002
41 Siemens AG. SIMATIC NET 工业通讯的 IT 解决方案,2002

35 Siemens AG. S7-300 可编程序控制器产品目录,2003
36 Siemens AG. S7-400 可编程控制器 CPU 及模板规范手册,2003
37 Siemens AG. S7-400 产品目录,2003
38 Siemens AG. 工业通讯及现场设备产品目录 IK PI,2001
39 Siemens AG. 用于自动控制系统的工业通讯网络,2001
40 Siemens AG. SIMATIC NET 网络解决方案,工业以太网交换机与连接模块,2002
41 Siemens AG. SIMATIC NET 工业通讯的 IT 解决方案,2002

35 Siemens AG. S7-300 可编程序控制器产品目录,2003
36 Siemens AG. S7-400 可编程控制器 CPU 及模板规范手册,2003
37 Siemens AG. S7-400 产品目录,2003
38 Siemens AG. 工业通讯及现场设备产品目录 IK PI,2001
39 Siemens AG. 用于自动控制系统的工业通讯网络,2001
40 Siemens AG. SIMATIC NET 网络解决方案,工业以太网交换机与连接模块,2002
41 Siemens AG. SIMATIC NET 工业通讯的 IT 解决方案,2002