

深入浅出西门子自动化产品系列丛书

深入浅出

西门子S7-200PLC



西门子(中国)有限公司
自动化与驱动集团



北京航空航天大学出版社

深入浅出 西门子 S7-200 PLC

西门子(中国)有限公司自动化与驱动集团

北京航空航天大学出版社

内容简介

本书主要介绍了西门子公司小型可编程序控制器 S7-200 PLC 的软硬件功能,以实用、易用为主线,涉及 S7-200 的方方面面;同时编者也将其多年宝贵的应用经验贯穿内容始终,使读者能够有所借鉴。本书主要针对大专院校的师生,电气设计人员以及调试编程人员等等。

本书分为三部分:第一部分是 S7-200 功能概述,主要介绍 S7-200 的硬件部分和通讯功能;第二部分主要介绍 S7-200 编程软件 Step7-Micro/WIN32 的使用和联机调试;第三部分介绍了 S7-200 的常用功能,并辅以简单的编程实例。

图书在版编目(CIP)数据

深入浅出西门子 S7-200 PLC/西门子(中国)有限公司自动化与驱动集团编. —北京:北京航空航天大学出版社,2003.12

ISBN 7-81077-426-3

I. 深… II. 西… III. 可编程序控制器, 西门子 S7-200 PLC IV. TP332.3

中国版本图书馆 CIP 数据核字(2003)第 112443 号

版权声明:本书版权(著作权)归西门子(中国)有限公司自动化与驱动集团所有。

深入浅出西门子 S7-200 PLC

西门子(中国)有限公司自动化与驱动集团

责任编辑 胡 敏

*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号(100083) 发行部电话:010-82317024 传真:010-82328026

<http://www.buaapress.com.cn> E-mail:bhpress@263.net

涿州市新华印刷有限公司印装 各地书店经销

*

开本: 787 × 1092 1/16 印张: 13.25 字数: 238 千字

2003 年 12 月第 1 版 2004 年 4 月第 2 次印刷 印数: 7 001~11 000 册

ISBN 7-81077-426-3 定价: 25.00 元(含光盘)

主 编

蔡行健

编 委

鲁	炜	朱震忠
王	颖	王东滨

前 言

SIMATIC S7-200 系列可编程序控制器(PLC)是德国西门子公司的产品。S7-200 PLC 是 SIMATIC S7 家族中的小型可编程序控制器,家族中还有中型可编程序控制器 S7-300 系列以及大型可编程序控制器 S7-400 系列。本书是《深入浅出西门子×××》系列丛书之一,《深入浅出西门子 S7-300 PLC》、《深入浅出西门子 S7-400 PLC》、《深入浅出西门子 Win CC》等等都将会陆续与读者见面。S7-200 凭借其强大的组网能力、友好易用的编程软件、极高的性价比和不断的创新而成为市场上众多小型可编程序控制器的领跑者,深受中国用户的喜爱。

本书是由西门子公司资深的 S7-200 技术支持工程师们凭借多年的实践经验,潜心推敲,花费一年的时间精心编写而成。本书以实用、易学为主旨,阐述功能时包含了许多应用实例,辅以程序代码及清晰明了的程序注释。编者凭借对 S7-200 PLC 的透彻理解,用较短的篇幅对 S7-200 的软硬件功能进行了精炼,真正达到了简洁又不失深度的境界。同时,编者的应用经验也贯串本书始终,使读者能够有所借鉴。无论您是大专院校的师生,电气设计人员还是调试编程人员,相信本书都会对您有所裨益。

本书分为四章:第 1 章是 S7-200 系统概述,讲述了 S7-200 的硬件和通讯功能,相信第 1 章会使您成为独立进行 S7-200 硬件选型和配置的行家;第 2 章讲述了 S7-200 编程软件的使用和电脑与 PLC 的联机,这些内容会使您对 S7-200 编程软件驾轻就熟;第 3 章讲述了 S7-200 CPU 的常用功能,每种功能都辅以编程实例与程序注释,图文并茂,可令您轻松自如地掌握 S7-200 CPU 的常用功能。

为了您更有效地掌握 S7-200 PLC,我们还特地为您准备了免费随书附赠的光盘一张。在您阅读本书之前,我们强烈建议您先安装光盘中的 S7-200 PLC 演示版编程软件,这样您在阅读的同时,还可以进行软件操作,更利于您对 S7-200 PLC 软件的学习。当您读了本书,想特别深入地了解某一部分内容时,您还可以查阅光盘中的《S7-200 系统手册》。该光盘还包括了《S7-200 应用论文集》,我们从 S7-200 众多的应用中精选出来 45 种应用,编辑成册,希望能对您有所借鉴。另外,光盘中的 S7-200 PLC 高级通讯指南包含了讲述 S7-200 高级通讯的文章,如使用无线调制解调器通讯,以太网通讯等等;其中,还有 S7-200 使用技巧案例集(Tips&Tricks),Tips&Tricks 是西门子总部制作的电子版 S7-200 使用技巧集,每个技巧都辅以程序实例,认真学习,定会达到事半功倍的效果。

如果您想得到更多关于 S7-200 的资料和信息,请您登陆:
www.S7-200.com。

本书难免有不足之处,欢迎您多提宝贵意见和建议。

好了,下面让我们一起来体会轻松学习 S7-200 的乐趣吧!



请记住,我们的口号是:

EASY BOOK for EASY LIFE!!!



目 录

第 1 章 S7-200 系统概述

1.1	系统功能概述	2
1.2	S7-200 CPU 和扩展模块	3
1.2.1	S7-200 CPU	3
1.2.2	扩展模块	6
1.2.3	电源计算	11
1.2.4	最大 I/O 配置	12
1.3	数据保存	15
1.4	通讯和网络功能	18
1.4.1	PPI 网络通讯	18
1.4.2	Profibus-DP 网络通讯	20
1.4.3	自由口通讯	20
1.4.4	USS 和 Modbus RTU 从站指令库	21
1.4.5	网络通讯硬件	22
1.4.6	以太网通讯	23
1.4.7	Modem 远程通讯	24
1.5	系统开发条件	25
1.5.1	编程软件和运行环境	25
1.5.2	编程通讯方式	26

第 2 章 S7-200 编程软件——Step7-Micro/WIN32

2.1	软件安装和设置	28
2.1.1	安装条件	28
2.1.2	安 装	28
2.1.3	安装 SP 升级包(Service Pack)	35



2.1.4	指令库和 Toolbox	40
2.2	Step7 - Micro/WIN32 简介	40
2.2.1	Step7 - Micro/WIN32 窗口元素	40
2.2.2	项目及其组件	43
2.2.3	定制 Step7 - Micro/WIN32	45
2.2.4	使用帮助	48
2.3	编程计算机与 CPU 通讯	49
2.3.1	设置通讯	49
2.3.2	PLC 信息	54
2.3.3	实时时钟	57
2.4	系统块设置	58
2.4.1	通讯口	59
2.4.2	数据保持区	59
2.4.3	S7 - 200 CPU 密码保护	61
2.4.4	输出表	63
2.4.5	输入滤波器	64
2.4.6	模拟量输入滤波器	65
2.4.7	脉冲捕捉功能	66
2.5	编程	67
2.5.1	任务	67
2.5.2	输入和编辑程序	69
2.5.3	编译和下载	78
2.5.4	运行和调试	79
2.6	符号表	86
2.7	交叉参考	87
2.8	数据块	89
2.9	Tools(工具)	90

第 3 章 S7 - 200 常用功能及编程简介

3.1	S7 - 200 寻址与基本指令	94
3.1.1	S7 - 200 如何工作	94



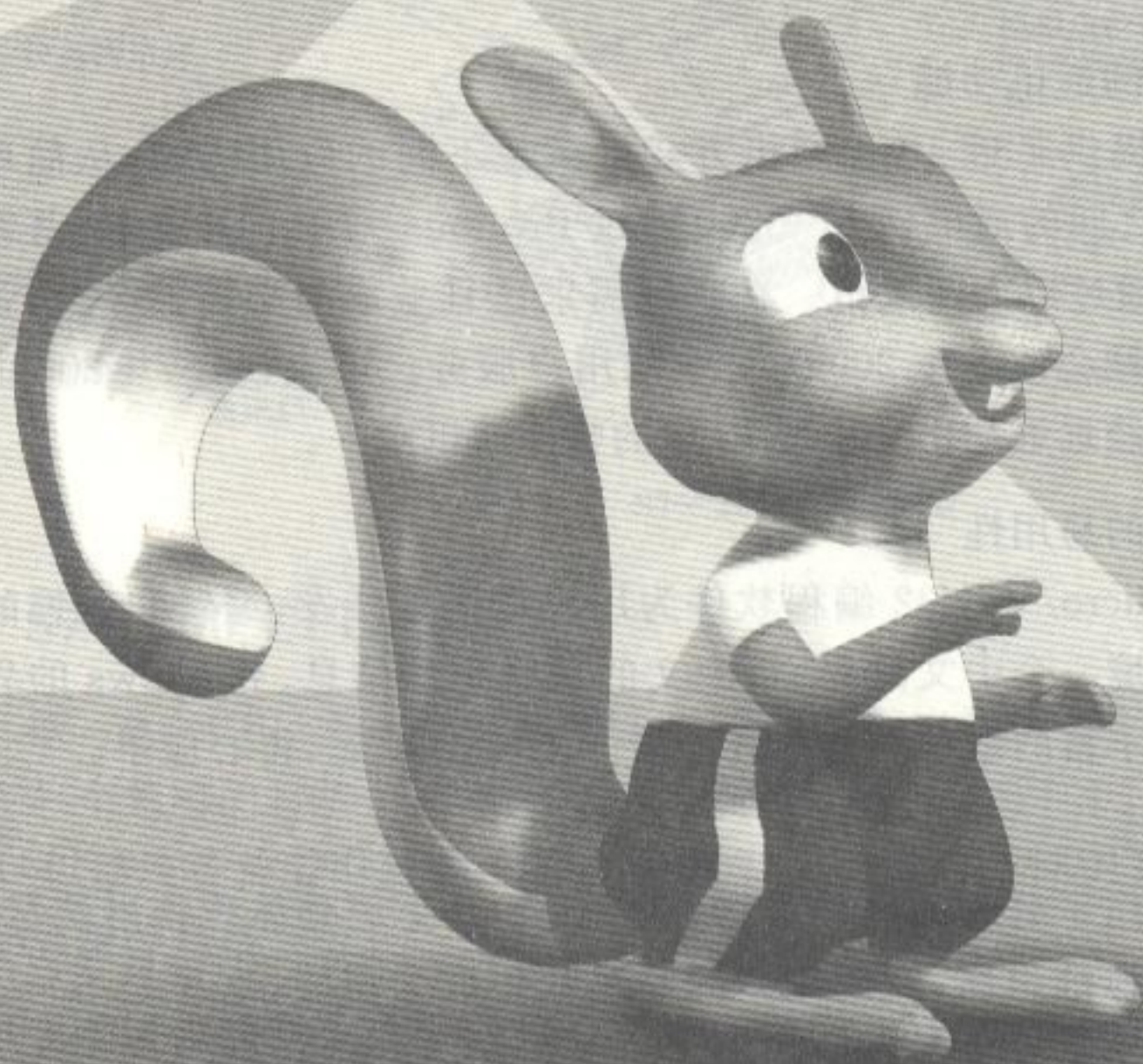
3.1.2	S7-200 CPU 的工作模式	95
3.1.3	S7-200 寻址	95
3.1.4	S7-200 的集成 I/O 和扩展 I/O	100
3.1.5	基本指令	101
3.2	定时器和计数器	105
3.2.1	定时器	105
3.2.2	计数器	110
3.3	系统时钟	113
3.4	子程序和中断服务程序	116
3.4.1	子程序	116
3.4.2	中断服务程序	123
3.4.3	程序的密码保护	127
3.5	高速计数器	129
3.6	高速脉冲输出	138
3.7	网络读写	147
3.8	自由口通讯	152
3.8.1	发送指令	154
3.8.2	接收指令	159
3.9	TD200 组态	166
3.10	PID 功能	177

第 4 章 附 录

4.1	CPU 规格	182
4.2	S7-200 CPU 接线	185
4.3	数字量扩展模块	186

第 1 章

S7 - 200 系统概述





1.1 系统功能概述

S7-200 PLC 系统是紧凑型可编程序控制器。系统的硬件构架由成系统的 CPU 模块和丰富的扩展模块组成。它能够满足各种设备的自动化控制需求。S7-200 除具有 PLC 基本的控制功能外,更在如下方面有其独到之处,想必这些也正是 S7-200 如此受欢迎的原因。

● 功能强大的指令集

指令内容包括位逻辑指令、计数器、定时器、复杂数学运算指令、PID 指令、字符串指令、时钟指令、通讯指令,以及和智能模块配合的专用指令等。

● 丰富强大的通讯功能

S7-200 提供了近 10 种通讯方式以满足不同的应用需求,从简单的 S7-200 之间的通讯到 S7-200 通过 Profibus-DP 网络通讯,甚至到 S7-200 通过以太网通讯。在联网需求已日益成为必需的今天,强大的通讯无疑会使 S7-200 为更多的用户服务。可以说,S7-200 的通讯功能已经远远超出了小型 PLC 的整体通讯水平。

● 编程软件的易用性

Step7-Micro/WIN32 编程软件为用户提供了开发、编辑和监控的良好编程环境。全中文的界面、中文的在线帮助信息、Windows 的界面风格以及丰富的编程向导,能使用户快速进入状态,得心应手。

● 不断地创新

创新是西门子公司的一贯风格,在 S7-200 上更是体现得淋漓尽致。永不停歇地推出新品,使更多的需求梦想成为现实,这正是 S7-200 成为市场佼佼者的原动力。

“千里之行,始于足下”,下面让我们从 S7-200 的 CPU 模块开始我们的轻松之旅吧!



1.2 S7-200 CPU 和扩展模块

1.2.1 S7-200 CPU

S7-200 CPU

S7-200 CPU 外形如图 1-1 所示。

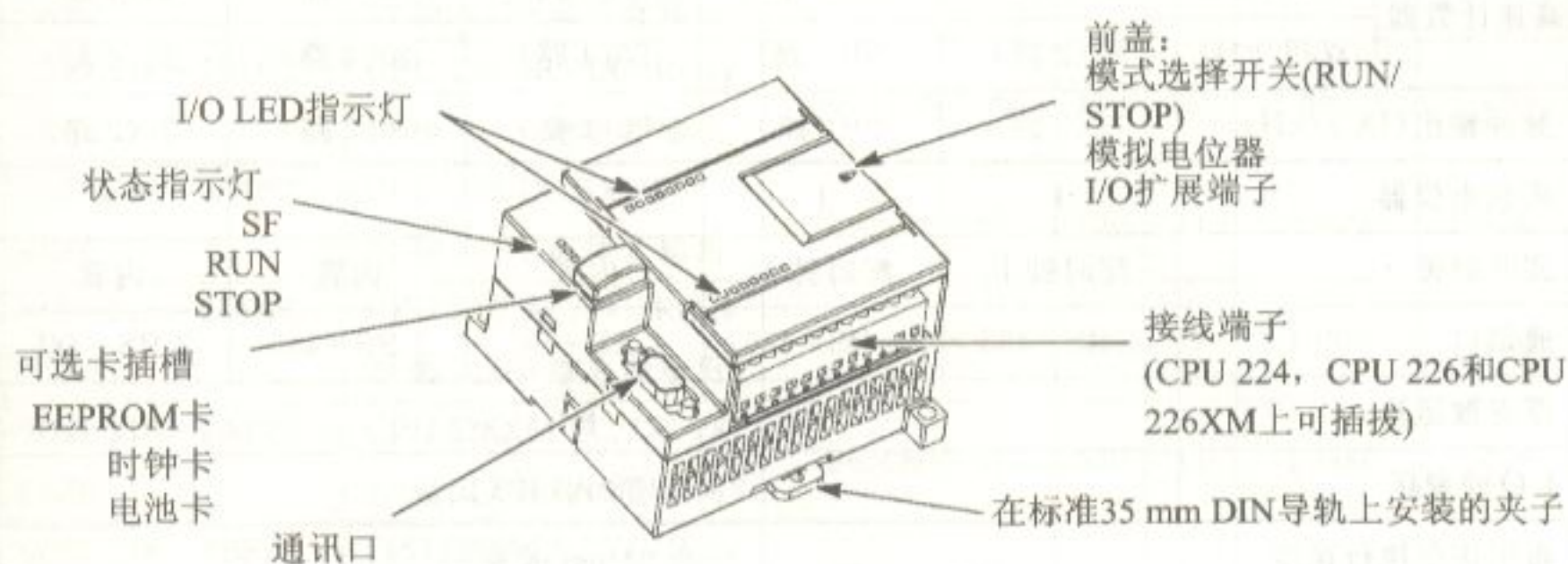


图 1-1 S7-200 CPU 外型

S7-200 CPU 规格

S7-200 CPU 将一个微处理器、一个集成的电源和若干数字量 I/O 点集成在一个紧凑的封装中,组成一个功能强大的 PLC。西门子提供多种类型的 CPU 以适应各种应用要求。不同类型的 CPU 具有不同的数字量 I/O 点数、内存容量等规格参数。

目前提供的 S7-200 CPU 有:CPU 221、CPU 222、CPU 224、CPU 226 和 CPU 226XM。



S7-200 CPU 规格如表 1-1 所列。

表 1-1 S7-200 CPU 规格表

特 性		CPU 221	CPU 222	CPU 224	CPU 226	CPU 226XM
外形尺寸/ (mm×mm×mm)		90×80×62	90×80×62	120.5×80×62	190×80×62	190×80×62
程序存储区/字节		4 096	4 096	8 192	8 192	16 384
数据存储区/字节		2 048	2 048	5 120	5 120	10 240
掉电保持时间/h		50	50	190	190	190
本机 I/O		6 入/4 出	8 入/6 出	14 入/10 出	24 入/16 出	24 入/16 出
扩展模块数量		0	2	7	7	7
高速计数器	单相/kHz	30(4 路)	30(4 路)	30(6 路)	30(6 路)	30(6 路)
	双相/kHz	20(2 路)	20(2 路)	20(4 路)	20(4 路)	20(4 路)
脉冲输出(DC)/kHz		20(2 路)	20(2 路)	20(2 路)	20(2 路)	20(2 路)
模拟电位器		1	1	2	2	2
实时时钟		配时钟卡	配时钟卡	内置	内置	内置
通讯口		1 RS-485	1 RS-485	1 RS-485	2 RS-485	2 RS-485
浮点数运算		有				
I/O 映象区		256 (128 入/128 出)				
布尔指令执行速度		0.37 μs/指令				

对于每个型号,西门子提供 DC(24 V)和 AC(120~220 V)两种电源供电的 CPU 类型。如 CPU 224 DC/DC/DC 和 CPU 224 AC/DC/Relay。每个类型都有各自的订货号,可以单独订货。

★ DC/DC/DC:说明 CPU 是直流供电,直流数字量输入,数字量输出点是晶体管直流电路的类型。

★ AC/DC/Relay:说明 CPU 是交流供电,直流数字量输入,数字量输出点是继电器触点的类型。

CPU 通用规范如表 1-2 所列。



表 1-2 CPU 通用规范

订货号	模板名称和描述	尺寸 /(mm×mm×mm)	质量 /g	损耗 /W	供电能力/mA	
					+5 V DC	+24 V DC
6ES7 211-0AA22-0XB0	CPU 221 DC/DC/DC 6 输入/4 晶体管输出	90×80×62	270	3	0	180
6ES7 211-0BA22-0XB0	CPU 221 AC/DC/Relay 6 输入/4 继电器输出	90×80×62	310	6	0	180
6ES7 212-1AB22-0XB0	CPU 222 DC/DC/DC 8 输入/6 晶体管输出	90×80×62	270	5	340	180
6ES7 212-1BB22-0XB0	CPU 222 AC/DC/Relay 8 输入/6 继电器输出	90×80×62	310	7	340	180
6ES7 214-1AD22-0XB0	CPU 224 DC/DC/DC 14 输入/10 晶体管输出	120.5×80×62	360	7	660	280
6ES7 214-1BD22-0XB0	CPU 224 AC/DC/Relay 14 输入/10 继电器输出	120.5×80×62	410	10	660	280
6ES7 216-2AD22-0XB0	CPU 226 DC/DC/DC 24 输入/16 晶体管输出	196×80×62	550	11	1 000	400
6ES7 216-2BD22-0XB0	CPU 226 AC/DC/Relay 24 输入/16 继电器输出	196×80×62	660	17	1 000	400
6ES7 216-2AF22-0XB0	CPU 226XM DC/DC/DC 24 输入/16 晶体管输出	196×80×62	550	11	1 000	400
6ES7 216-2BF22-0XB0	CPU 226XM AC/DC/Relay 24 输入/16 继电器输出	196×80×62	660	17	1 000	400

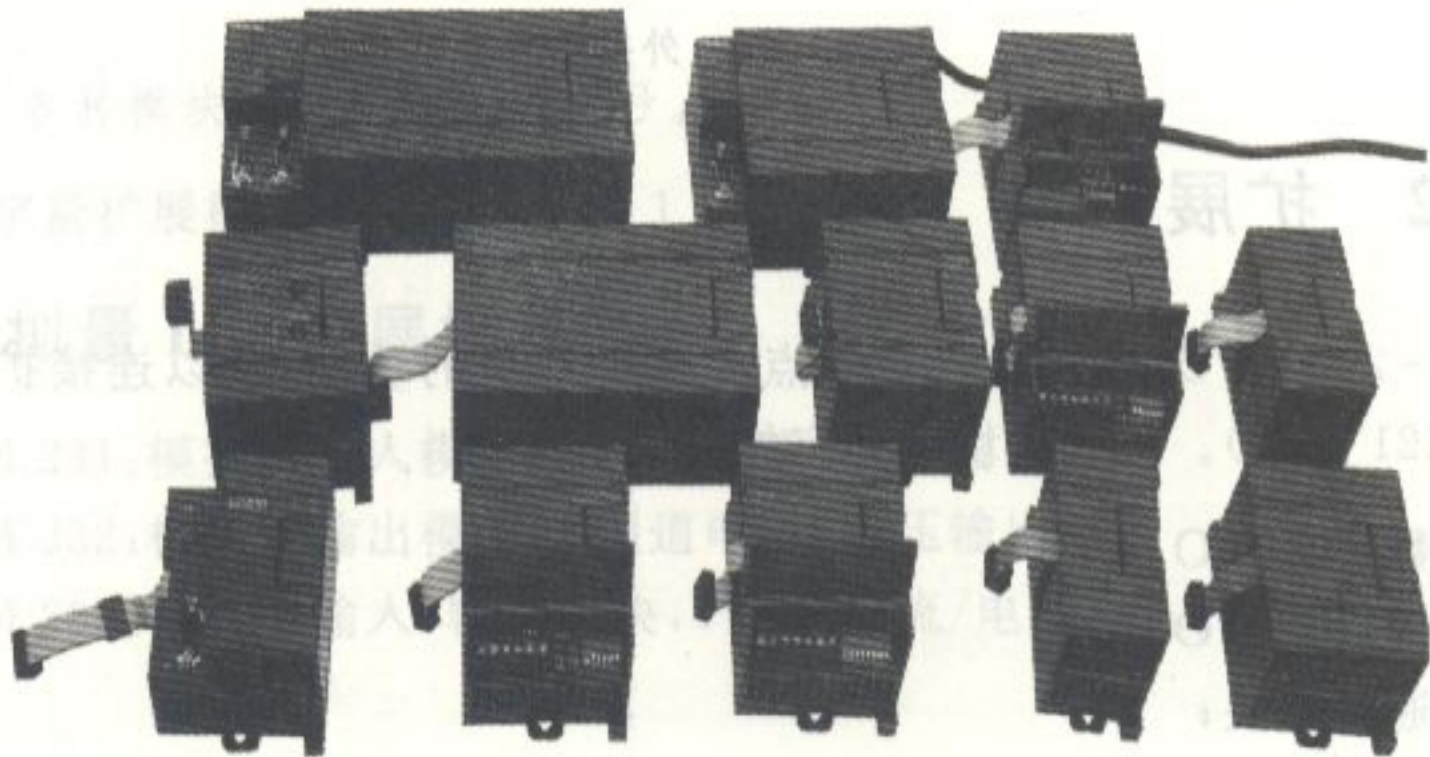


图 1-2 S7-200 CPU 和扩展模块(部分)



S7-200 CPU 外插卡

CPU 提供了一个可选卡插槽,可根据需要插入三种插卡中的一种(如图 1-3 所示)。外插卡须单独订货。

- MC291:存储器卡,提供 EEPROM 存储单元。在 CPU 上插入存储器卡后,可使用编程软件 Step7-Micro/WIN32 将 CPU 中的存储内容(系统块、程序块和数据块等)复制到卡中;把存储卡插到其他 CPU 上,通电时存储卡的内容会自动复制到 CPU 中。存储器卡用于传递程序,被写入的 CPU 必须和提供内容来源的 CPU 相同,或更新,型号更高。
- CC292:日期/时钟电池卡。用于 CPU 221 和 CPU 222 两种不具备内部实时时钟的 CPU,以提供日期/时钟功能,同时提供内存后备电池。
- BC293:电池卡。为所有类型的 CPU 提供数据保持的后备电池。电池在超级电容放电完毕后起作用。

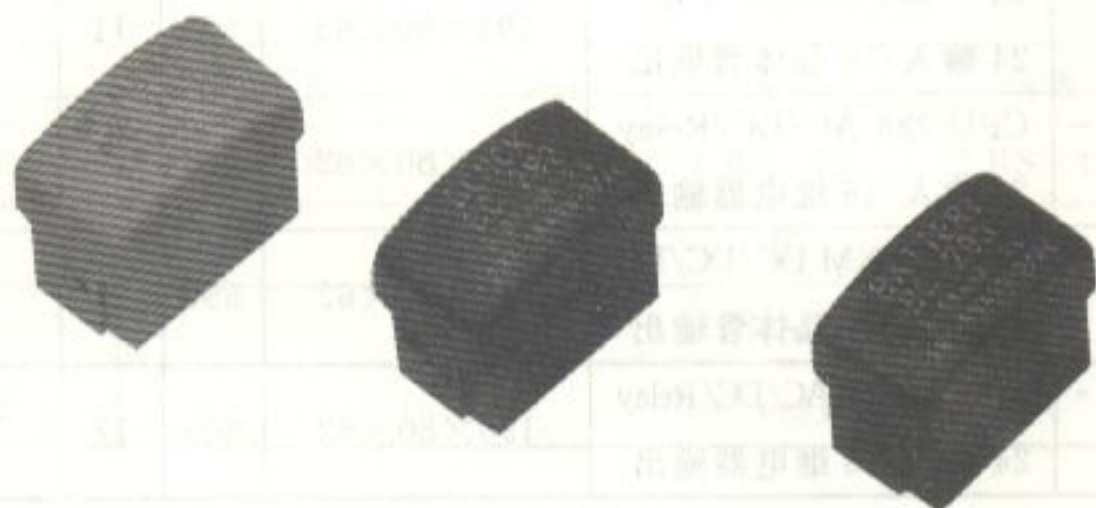


图 1-3 外插卡

1.2.2 扩展模块

S7-200 CPU 为了扩展 I/O 点和执行特殊的功能,可以连接扩展模块(CPU 221 除外)。扩展模块主要有如下几类:

- 数字量 I/O 模块;
- 模拟量 I/O 模块;
- 通讯模块;
- 特殊功能模块。



数字量 I/O 扩展模块

EM 221: 数字量输入扩展模块。包括 3 种类型:

- 8 点 24 V DC 输入;
- 8 点 AC 120/230 V 输入;
- 16 点 24 V DC 输入。

EM 222: 数字量输出扩展模块, 包括 5 种类型:

- 8 点 24 V DC(晶体管)输出;
- 8 点继电器输出;
- 8 点 AC 120/230 V 输出;
- 4 点 24 V DC 输出, 每点 5 A;
- 4 点继电器输出, 每点 10 A。

EM 223: 数字量输入/输出扩展模块, 共有 6 种类型:

- 4 点 24 V DC 输入/4 V DC 输出;
- 4 点 24 V DC 输入/4 V 继电器输出;
- 8 点 24 V DC 输入/8 V DC 输出;
- 8 点 24 V DC 输入/8 V 继电器输出;
- 16 点 24 V DC 输入/16 V DC 输出;
- 16 点 24 V DC 输入/16 V 继电器输出。



各种模块具有各自的订货号。

数字量扩展模块通用规范如表 1-3 所列。

模拟量 I/O 扩展模块

EM 231: 模拟量输入模块, 4 通道电流/电压输入;

EM 232: 模拟量输出模块, 2 通道电流/电压输出;

EM 235: 模拟量输入/输出模块, 4 通道电流/电压输入、1 通道电流/电压输出。

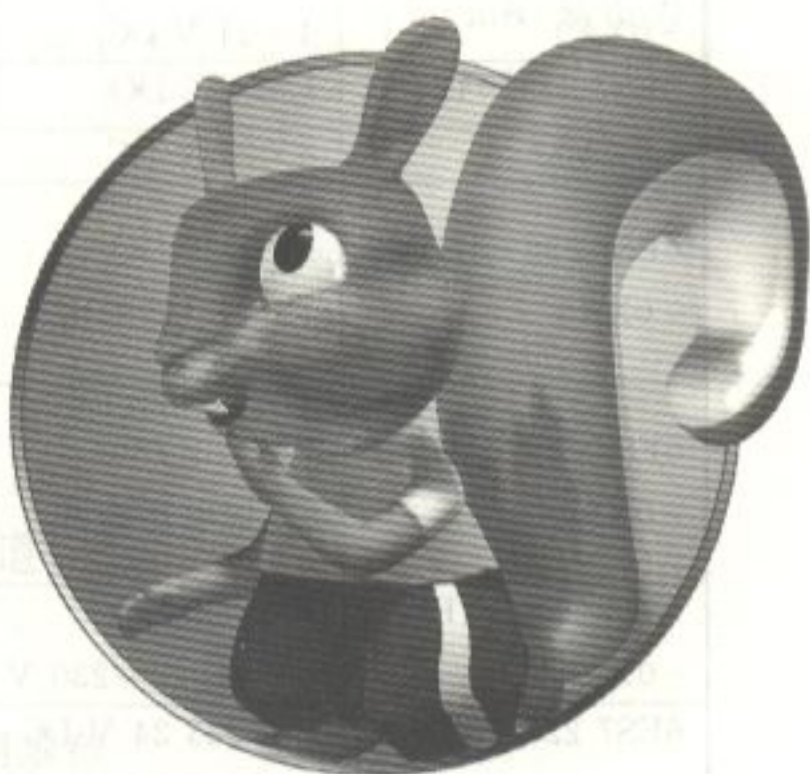




表 1-3 数字量扩展模块通用规范

定货号	模块名称和描述	尺寸(W×H×D) /(mm×mm×mm)	质量 /g	损耗 /W	电源要求	
					+5 V DC	+24 V DC
6ES7 221-1BF22-0XA0	EM 221 DI 8×24 V DC	46×80×62	150	2	30 mA	—
6ES7 221-1EF22-0XA0	EM 221 DI 8×AC 120/230 V	71.2×80×62	160	3	30 mA	—
6ES7 221-1BH22-0AX0	EM 221 DI 16×24 V DC	71.2×80×62	160	3	70 mA	—
6ES7 222-1BD22-0XA0	EM 222 DO 4×24 V DC	46×80×62	120	3	40 mA	—
6ES7 222-1BF22-0XA0	EM 222 DO 8×24 V DC	46×80×62	150	2	50 mA	—
6ES7 222-1HD22-0XA0	EM 222 DO 4×继电器输出	46×80×62	150	4	30 mA	接通时:20 mA/ 输出点 20.4~ 28.8 V DC
6ES7 222-1HF22-0XA0	EM 222 DO 8×继电器输出	46×80×62	170	2	40 mA	接通时:9 mA/ 输出点 20.4~ 28.8 V DC
6ES7 222-1EF22-0XA0	EM 222 DO 8×AC 120/230 V	71.2×80×62	165	4	110 mA	—
6ES7 223-1BF22-0XA0	EM 223 24 V DC 4 In/4 Out	46×80×62	160	2	40 mA	—
6ES7 223-1HF22-0XA0	EM 223 24 V DC 4 In/4 继电器输出	46×80×62	170	2	40 mA	接通时:9 mA/ 输出点 20.4~ 28.8 V DC
6ES7 223-1BH22-0AX0	EM 223 24 V DC 8 In/8 Out	71.2×80×62	200	3	80 mA	—
6ES7 223-1PH22-0XA0	EM 223 24 V DC 8 In/8 继电器输出	71.2×80×62	300	3	80 mA	接通时:9 mA/ 输出点 20.4~ 28.8 V DC
6ES7 223-1BL22-0XA0	EM 223 24 V DC 16 In/16 Out	137.3×80×62	360	6	160 mA	—
6ES7 223-1PL22-0XA0	EM 223 24 V DC 16 In/16 继电器输出	137.3×80×62	400	6	150 mA	接通时:9 mA/ 输出点 20.4~ 28.8 V DC



模拟量扩展模块通用规范如表 1-4 所列。

表 1-4 模拟量扩展模块通用规范

定货号	模块名称和描述	尺寸(W×H×D) /(mm×mm×mm)	质量 /g	损耗 /W	电源要求/mA	
					+5 V DC	+24 V DC
6ES7 231-0HC22-0XA0	EM 231 模拟输入, 4 输入	71.2×80×62	183	2	20 mA	60 mA
6ES7 232-0HB22-0XA0	EM 232 模拟输出, 2 输出	46×80×62	148	2	20 mA	70 mA (两个输出都是 20 mA)
6ES7 235-0KD22-0XA0	EM 235 模拟量混合模块 4 输入/1 输出	71.2×80×62	186	2	30 mA	60 mA (输出为 20 mA)

温度测量扩展模块

温度测量模块是模拟量模块的特殊形式。它们是：

EM 231 TC:热电偶输入模块,4 输入通道；

EM 231 RTD:热电阻输入模块,2 输入通道。

温度模块通用规范如表 1-5 所列。

表 1-5 温度模块通用规范

定货号	模块名称和描述	尺寸(W×H×D) /(mm×mm×mm)	质量 /g	损耗 /W	电源要求/mA	
					+5 V DC	+24 V DC
6ES7 231-7PD22-0XA0	EM 231 模拟输入 热电偶,4 输入	71.2×80×62	210	1.8	87	60
6ES7 231-7PB22-0XA0	EM 231 模拟输入 RTD,2 输入	71.2×80×62	210	1.8	87	60

特殊功能模块

S7-200 还提供了一些特殊模块,用以完成特定的任务。例如：

EM 253:定位控制模块。它能产生脉冲串,用于步进电机和伺服电机的速度和位置的开环控制。

EM 253 位控模块通用规范如表 1-6 所示。

表 1-6 EM 253 位控模块通用规范

订货号	模块名称和描述	尺寸(W×H×D) /(mm×mm×mm)	质量 /kg	损耗 /W	电源要求	
					+5 V DC	+24 V DC
6ES7 253-1AA22 -0XA0	EM 253 位控模块	71.2×80×62	0.190	2.5	190	见附录

通讯模块

S7-200 系统提供以下几种通讯模块,以适应不同的通讯方式。

EM 277: Profibus-DP 从站通讯模块,同时也支持 MPI 从站通讯;

EM 241: 调制解调器(Modem)通讯模块;

CP 243-1: 工业以太网通讯模块;

CP243-1 IT: 工业以太网通讯模块,同时提供 Web/E-mail 等 IT 应用;

CP243-2: AS-i 主站模块,可连接最多 62 个 AS-i 从站。

总线延长电缆

如果 S7-200 CPU 和扩展模块不能安装在一条导轨上,可以选用总线延长电缆,以适应灵活安装的需求。电缆长度 0.8 m,一个 S7-200 系统只能安装一条总线延长电缆(如图 1-4 所示)。

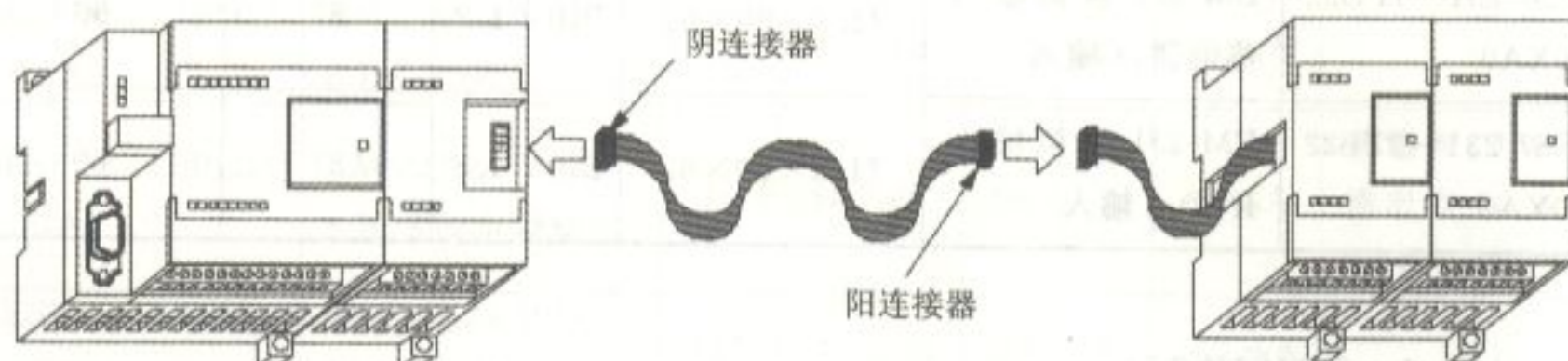


图 1-4 S7-200 系统的总线延长电缆



1.2.3 电源计算

所有的 S7-200 CPU 都有内部电源,为 CPU 自身、扩展模块和其他用电设备提供 5 V、24 V 直流电源。

扩展模块通过与 CPU 连接的总线连接电缆取得 5 V 直流电源(5 V DC)。

CPU 还向外提供一个 24 V DC 电源,从电源输出点(L+,M)引出。此电源可为 CPU 和扩展模块上的 I/O 点供电,也为一些特殊或智能模块提供电源。此电源还从 S7-200 CPU 上的通讯口输出,提供给 PC/PPI 编程电缆,或 TD200 文本显示操作界面等设备。

S7-200 CPU 供电能力如表 1-7 所列。

表 1-7 S7-200 CPU 供电能力 mA

CPU 型号	5 V DC	24 V DC
CPU 221	—(不能加扩展模块)	180
CPU 222	340	180
CPU 224	660	280
CPU 226/CPU 226XM	1 000	400

由表 1-7 可见,不同规格的 CPU 提供 5 V DC 和 24 V DC 电源的容量不同。每个实际应用项目都要就电源容量进行规划计算。

每个扩展模块都需要 5 V DC 电源,应当检查所有扩展模块的 5 V DC 电源需求是否超出 CPU 的供电能力,如果超出,就必须减少或改变模块配置。

有些扩展模块需要 24 V DC 电源供电,I/O 点也可能需要 24 V DC 电源,TD200 等也需要 24 V DC 电源。这些电源也要根据 CPU 的供电能力进行计算。如果所需电源超出电源的容量,需要增加外接 24 V DC 电源。S7-200 CPU 上提供的电源不能和外接电源并联,但它们必须共地。

一个电源计算的示例如表 1-8 所列。



表 1-8 电源计算算例

CPU 电源预算	5 V DC	24 V DC
CPU 224 AC/DC/继电器	660 mA	280 mA
	减去以下电源需求	减去以下电源需求
系统要求	5 V DC	24 V DC
CPU 224,14 输入		$14 \times 4 \text{ mA} = 56 \text{ mA}$
3 EM 223,5 V 电源需求	$3 \times 80 \text{ mA} = 240 \text{ mA}$	
1 EM 221,5 V 电源需求	$1 \times 30 \text{ mA} = 30 \text{ mA}$	
3 EM 223,每个 8 输入		$3 \times 8 \times 4 \text{ mA} = 96 \text{ mA}$
3 EM 223,每个 8 继电器线圈		$3 \times 8 \times 9 \text{ mA} = 216 \text{ mA}$
1 EM 221,每个 8 输入		$1 \times 8 \times 4 \text{ mA} = 32 \text{ mA}$
总需求	270 mA	400 mA
电压差额	5 V DC	24 V DC
总电流差额	剩 390 mA	缺 120 mA

1.2.4 最大 I/O 配置

I/O 地址分配规则

S7-200 按照 I/O 的类型为其分配不同的地址,共有 4 类:

- DI: 数字量输入;
- DO: 数字量输出;
- AI: 模拟量输入;
- AO: 模拟量输出。

每一类 I/O 分别排列地址。从 CPU 开始算起,I/O 点从左到右按由小到大的规律排列。扩展模块的类型和位置一旦确定,则它的 I/O 点地址也随之决定。



最大 I/O

S7-200 CPU 虽然具有相同的 I/O 映象区,但不同 CPU 的最大 I/O,实际上取决于它们所能带的扩展模块数目。

- CPU 221:0 个扩展模块;
- CPU 222:2 个扩展模块;
- CPU 224:7 个扩展模块;
- CPU 226/CPU 226 XM:7 个扩展模块。

S7-200 最大 I/O 如表 1-9 所列。

表 1-9 S7-200 最大 I/O

模 块	5 V 电源 /mA	数字量 输入	数字量 输出	模拟量 输入	模拟量 输出
CPU 221	不能扩展				
CPU 222					
最大数字输入/输出					
CPU	340	8	6		
2×EM 223 DI16/DO16×24 V DC 或者	—320 或者	32	32		
2×EM 223 DI16/DO16×24 V DC/继电器	—300				
总和	>0	40	38		
最大模拟量输入					
CPU	340	8	6		
2×EM 235 AI4/AQ1	—60			8	2
总和	>0	8	6	8	2
最大模拟量输出					
CPU	340	8	6		
2×EM 232 AQ2	—40			0	4
总和	>0	8	6	0	4
CPU 224					
最大数字量输入/继电器输出					
CPU	660	14	10		
4×EM 223 DI16/DO16×24 V DC/继电器	—600	64	64		



续表 1-9

模 块	5 V 电源 /mA	数字量 输入	数字量 输出	模拟量 输入	模拟量 输出
2×EM 221 DI8×24 V DC	-60	16			
总和	0	94	74		
最大数字量输入/DC 输出					
CPU	660	14	10		
4×EM 223 DI16/DO16×24 V DC	-640	64	64		
总和	>0	78	74		
数字量输入/最大继电器输出					
CPU	660	14	10		
4×EM 223 DI16/DO16×24 V DC/继电器	-600	64	64		
1×EM 222 DO8×继电器	-40		8		
总和	>0	78	82		
CPU 226					
最大数字量输入/继电器输出					
CPU	1 000	24	16		
6×EM 223 DI16/DO16×24 V DC/继电器	-900	96	96		
1×EM 223 DI8/DO8×24 V DC/继电器	-80	8	8		
总和	>0	128	120		
最大数字量输入/DC 输出					
CPU	1 000	24	16		
6×EM 223 DI16/DO16×24 V DC	-960	96	96		
1×EM 221 DI8×24 V DC	-30	8			
总和	>0	128	112		
CPU 224 或 CPU 226					
最大模拟量输入					
CPU	>660	14(24)	10(16)		
7×EM 235 AI4/AQ1	-210			28	7
总和	>0	14(24)	10(16)	28	7
最大模拟量输出					
CPU	>660	14(24)	10(16)		
7×EM 232 AQ2	-140			0	14
总和	>0	14(24)	10(16)	0	14



✌ 最大 I/O 数目不但取决于 CPU 所能扩展的模块数量,还取决于 CPU 内部电源所能提供的 5 V DC 电源容量。

1.3 数据保持

S7-200 提供了几种保持数据的方法,用户根据需要可以灵活选用:

- CPU 中内置超级电容,在不太长的断电期间内为保持数据和时钟提供电源,不需要附件;
- CPU 上附加电池卡,与内置超级电容配合,长期为时钟和数据保持提供电源;
- 使用数据块,永久保存不需要更改的数据;
- 编程时设置系统块,可在 CPU 断电时自动永久保存至多 14 个字节的数据;
- 在用户程序中编程,根据需要永久保存数据。

S7-200 的数据存储区

S7-200 CPU 中的数据存储区分为两类:易失性的 RAM 存储区,以及永久保存的 EEPROM 存储区。RAM 存储区需要为其提供电源方能保持其中的数据不丢失。

S7-200 中的 V 数据存储区、M 存储区都属于易失性数据存储区。要保存 T(定时器)和 C(计数器)数据,也需要提供电源。

S7-200 CPU 提供了 EEPROM 存储器。EEPROM 不需要另外的供电就能永久保存数据。EEPROM 对应于 RAM 中的 V 存储区和 M 存储区的一部分。要把数据存入 EEPROM,需要做一些设置,或者编程。



在 S7-200 项目的系统块中,有设置 RAM 数据保持区的选项。如果选中某个数据区,则 CPU 会在断电时通过内置超级电容和电池卡(如果有)保持其中的数据;如果内置超级电容放电完毕而数据消失,或者选择了不保持某个数据区的数据,则下次 CPU 上电时,会把 EEPROM 中相应的区域的内容复制到 RAM 中。

内置电容保持数据

CPU 内置超级电容,在短期断电期间为数据保持和实时时钟(如果有)提供电源。

断电后,CPU 221 和 CPU 222 的超级电容可提供约 50 h 的数据保持,CPU 224、CPU 226、CPU 226XM 可保持数据约 190 h。



超级电容在 CPU 上电时充电。为保证获得上述指标的数据保持时间,需要充电至少 24 h。

内置电容+电池卡保持数据

可以在 S7-200 CPU 的可选卡插槽上,插入电池卡 BC293 以提供额外的数据保持时间。对于 CPU 221 和 CPU 222,还可以选用时钟/电池卡 CC292,同时获得电池备份的数据保持和实时时钟。

CPU 断电后,首先依靠内置的超级电容为数据提供电源。超级电容放电完毕后,电池才起作用。

完全靠电池为 CPU 提供数据备份电源时,电池寿命约 200 天。

使用数据块

用户编程时可以编辑数据块。数据块用于给 S7-200 CPU 的 V 存储区赋予初始值。



由于数据块在 S7-200 项目下载到 CPU 中时,直接存储到 EEPROM 中,所以数据块的内容永远不会丢失。

数据块可以用于保存程序中用到的不改变的一些参数。

☞ 参见 2.8 节。

断电自动保存

S7-200 CPU 的 M 存储区有 14 字节(MB0~MB13),可以在 CPU 断电时自动将其中的内容写入到 EEPROM 的相应区域中,则数据可以永久保存。

默认情况下,M 存储区的这 14 个字节未设置为在断电时自动保存,需要在 S7-200 项目的系统块中进行设置。

☞ 参见 2.4.2 节。

编程保存数据

在程序中利用 SMB31 和 SMW32 特殊存储器,可以把 V 存储区中的任意地址的数据写到相应的 EEPROM 单元中,达到永久保存的目的。每次操作可以写入 1 个字节、字或者双字长度的数据。多次执行操作,可以写入多个数据。

✌ 由于 EEPROM 的写操作次数有限(最少 10 万次,典型 100 万次),在程序中必须注意写入操作的频度。

✌ 对于类似由操作人员不定期更改的工艺参数等数据,可以在用户程序中判断其状态,在变化之后执行写入 EEPROM 的操作。



1.4 通讯和网络功能

1.4.1 PPI 网络通讯

PPI(点对点接口)是西门子为 S7-200 系统开发的通讯协议。PPI 是一种主—从协议:主站设备发送要求到从站设备,从站设备响应。从站不主动发信息,只是等待主站的要求和对要求作出响应。

PPI 网络中可以有多主站(如图 1-5 所示)。PPI 并不限制与任意一个从站通讯的主站数量,但是在一个网段中,通讯站的个数不能超过 32。带中继器的网络结构如图 1-6 所示。

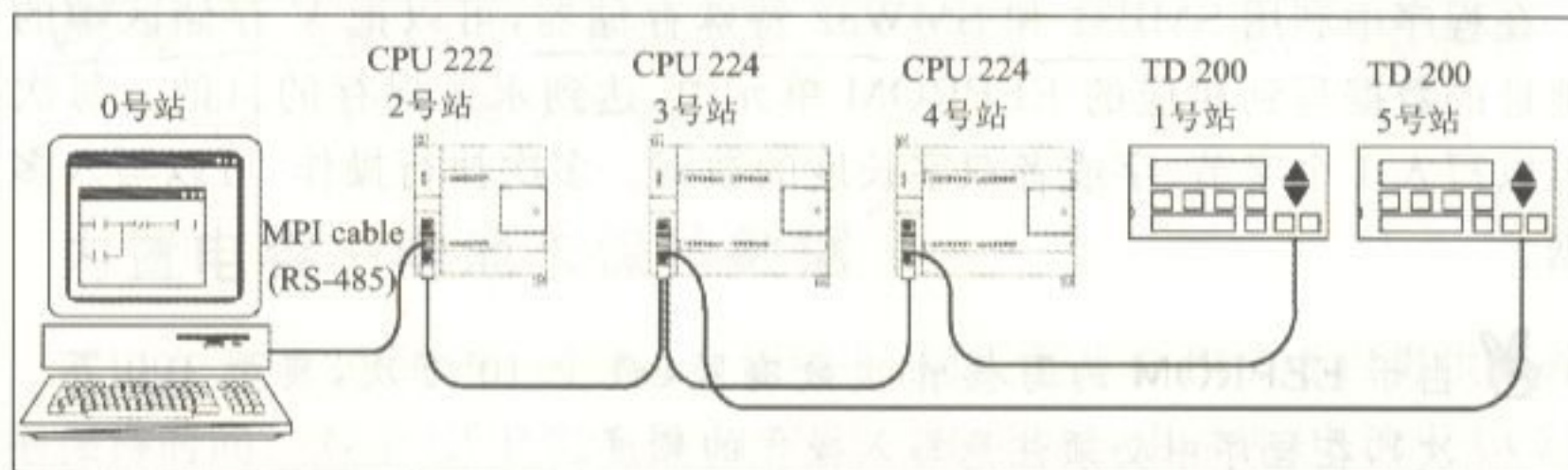


图 1-5 PPI 网络举例(图中计算机使用了 CP5611 等通讯卡)

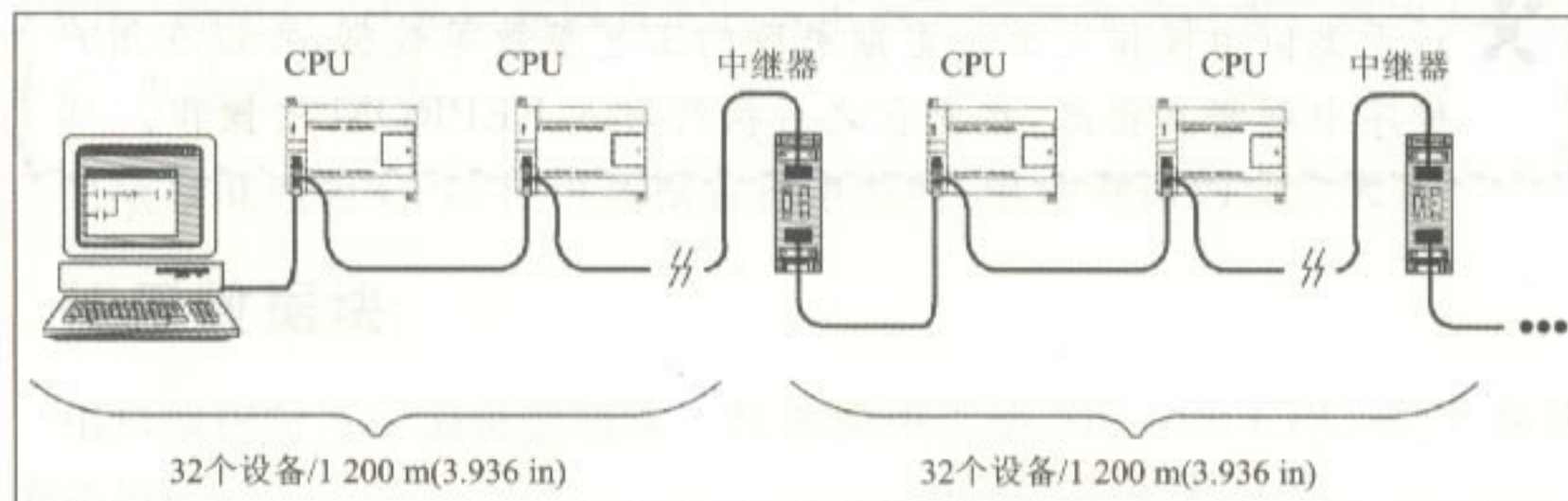


图 1-6 带中继器的网络结构



S7-200 CPU 上集成的通讯口支持 PPI 通讯。不隔离的 CPU 通讯口支持的标准 PPI 通讯距离为 50 m, 如果使用一对 RS-485 中继器, 可以达到 RS-485 的标准通讯距离 1 200 m。PPI 支持的通讯速率为 9.6 K 波特、19.2 K 波特和 187.5 K 波特。

✌ 运行编程软件 Step7-Micro/WIN32 的计算机可以认为是一个 PPI 主站。要获得 187.5 K 波特的 PPI 通讯速率, 必须有 CP 卡、Smart RS-232/PPI 电缆或 Smart USB/PPI 电缆作为编程接口。

其他设备, 如 TD200 文本显示器、TP170 micro 等操作面板(HMI), 也可以通过 PPI 协议和 S7-200 CPU 连接。图 1-7 所示为挂在 Profibus-DP 网络上的 S7-200。

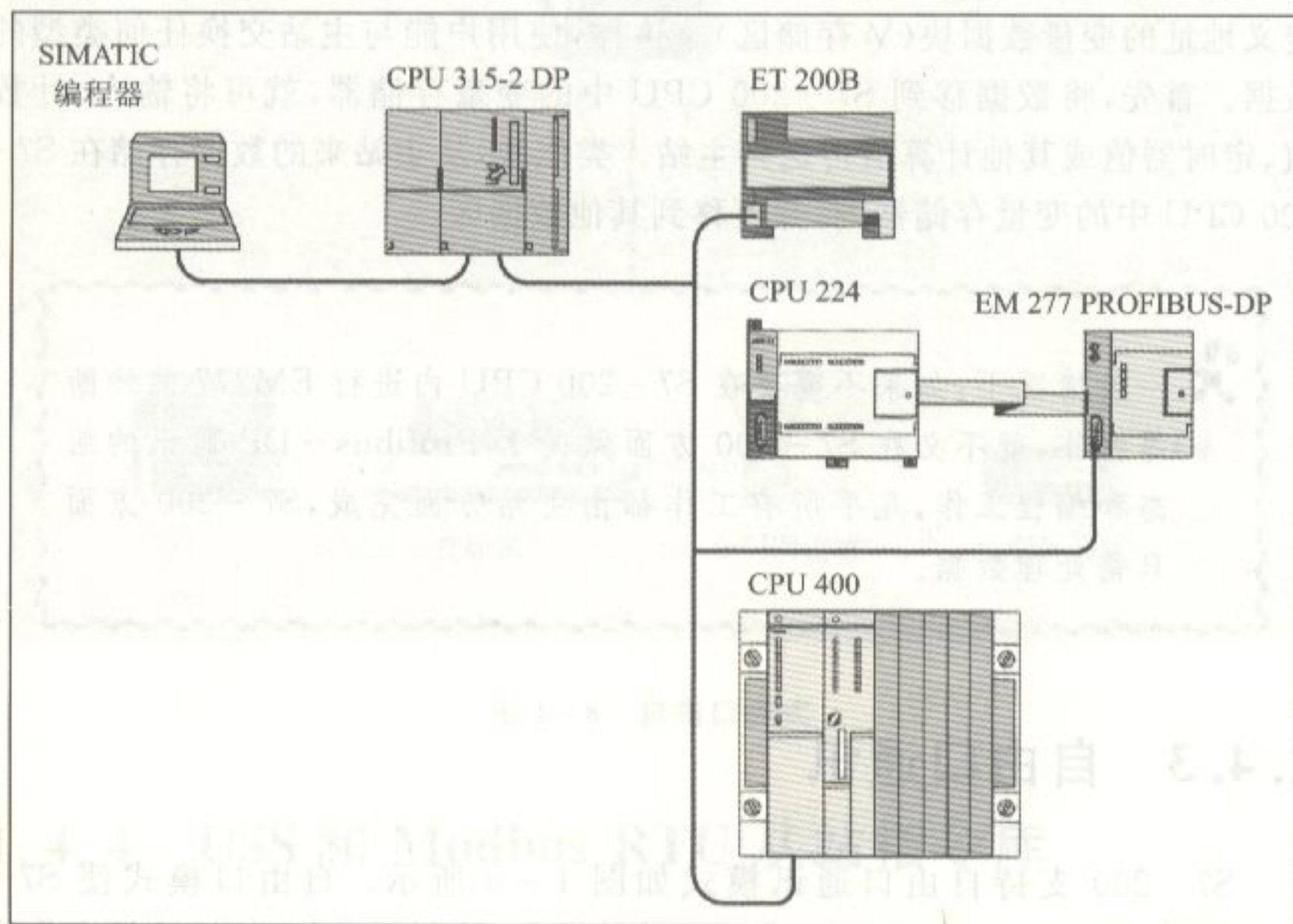


图 1-7 挂在 Profibus-DP 网络上的 S7-200



PPI 通讯是最容易实现的 S7-200 CPU 之间的网络通讯。您只需要编程设置主站通讯端口的工作模式,然后就可以用网络读写指令(NetR/NetW)读写从站的数据。

1.4.2 Profibus - DP 网络通讯

在 S7-200 系列的 CPU 中,CPU 222、CPU 224、CPU 226、CPU 226XM 都可以通过 EM277 Profibus - DP 扩展模块支持 Profibus - DP 网络协议。EM 277 通过模块扩展电缆连接到 S7-200 CPU。EM 277 Profibus - DP 模块的端口可运行于 9 600 波特和 12 M 波特之间的任何 Profibus 波特率。

作为 DP 从站,EM 277 模块接受从主站来的多种不同的 I/O 配置,向主站发送和接收不同数量的数据。EM 277 能读写 S7-200 CPU 中在主站方面定义地址的变量数据块(V 存储区)。这样,使用户能与主站交换任何类型的数据。首先,将数据移到 S7-200 CPU 中的变量存储器,就可将输入、计数值、定时器值或其他计算值传送到主站。类似地,从主站来的数据存储在 S7-200 CPU 中的变量存储器内,并可移到其他数据区。



一般情况下,如果不需要在 S7-200 CPU 内进行 EM277 的诊断等操作,就不必在 S7-200 方面做关于 Profibus - DP 通讯的组态和编程工作,几乎所有工作都由主站方面完成,S7-200 方面只需处理数据。

1.4.3 自由口通讯

S7-200 支持自由口通讯模式如图 1-8 所示。自由口模式使 S7-200 PLC 可以与许多通讯协议公开的其他设备、控制器进行通讯,波特率范围为 1 200~115 200 bit/s(可调整)。



自由口模式的数据字节格式总是有一个起始位、一个停止位,您可以选择7位或者8位数据,也可以选择是否有校验位,以及是奇校验还是偶校验。

在自由口模式下,使用 XMT(发送)和 RCV(接收)指令,为所有的通讯活动编程。通讯协议应符合通讯对象的要求或者由用户决定。

✌ CPU 的通讯口工作在自由口模式下时,此通讯口不能同时工作于其他通讯模式下,例如 PPI 编程状态。

✌ CPU 的通讯口是 RS-485 标准,如果通讯对象是 RS-232 设备,则需要 RS-232/PPI 电缆。

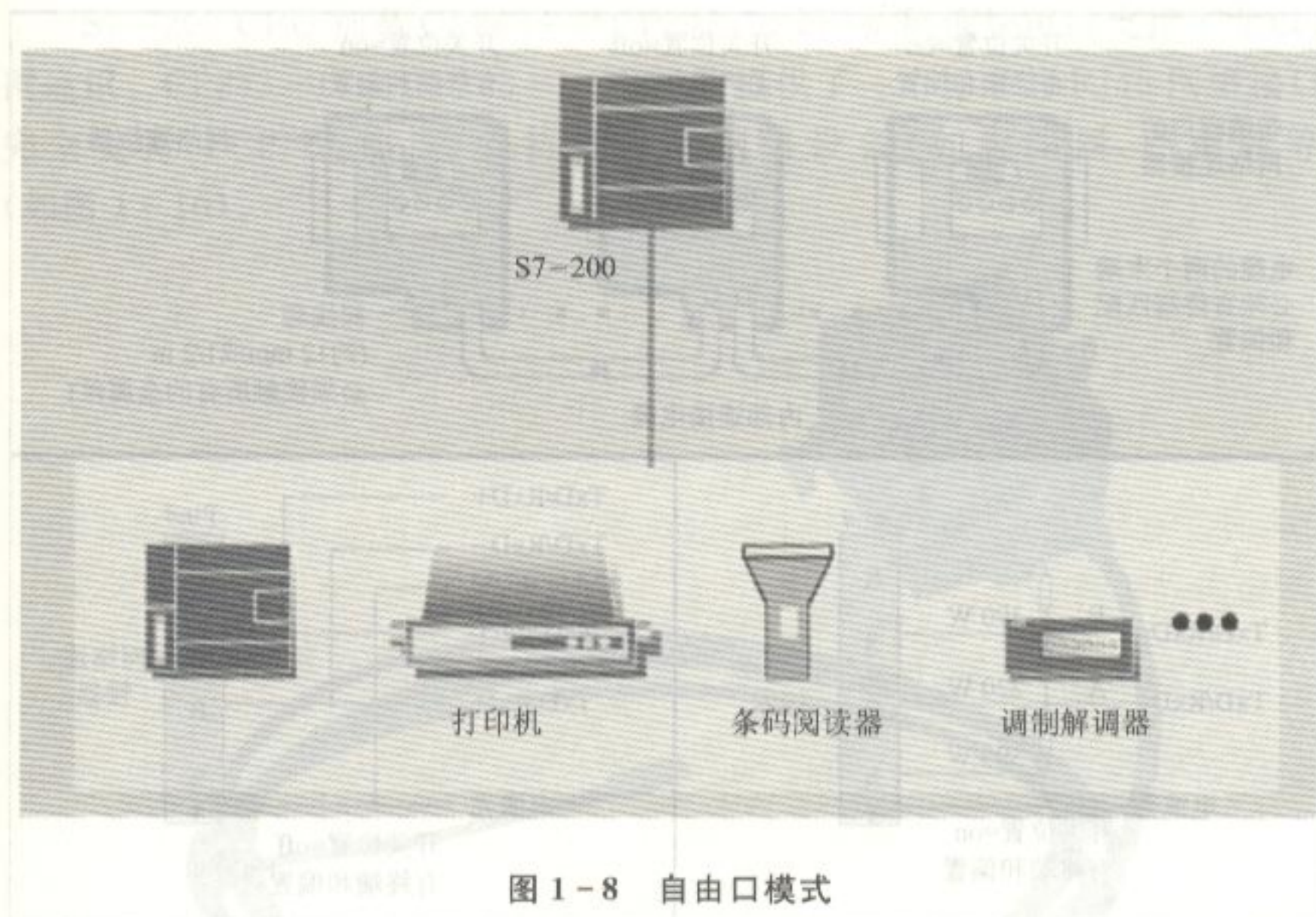


图 1-8 自由口模式

1.4.4 USS 和 Modbus RTU 从站指令库

Step7 - Micro/WIN32 V3.2 以上版提供了 USS 和 Modbus RTU 从站指令库。USS 指令库可以对 SIEMENS 生产的 MM420、MM430 和 MM440 变频器进行串行通讯控制; Modbus RTU 指令库为 S7-200 CPU 提供了



Modbus RTU 从站功能。

USS 和 Modbus 指令库都使用 S7-200 CPU 的自由口通讯模式编程实现。

1.4.5 网络通讯硬件

S7-200 支持的 PPI、Profibus-DP、自由口通讯模式都是建立在 RS-485 的硬件基础上。为保证足够的传输距离和通讯速率,建议使用 SIEMENS 制造的网络电缆和网络连接器(插头)。图 1-9 所示为网络连接器和网络电缆示意图。

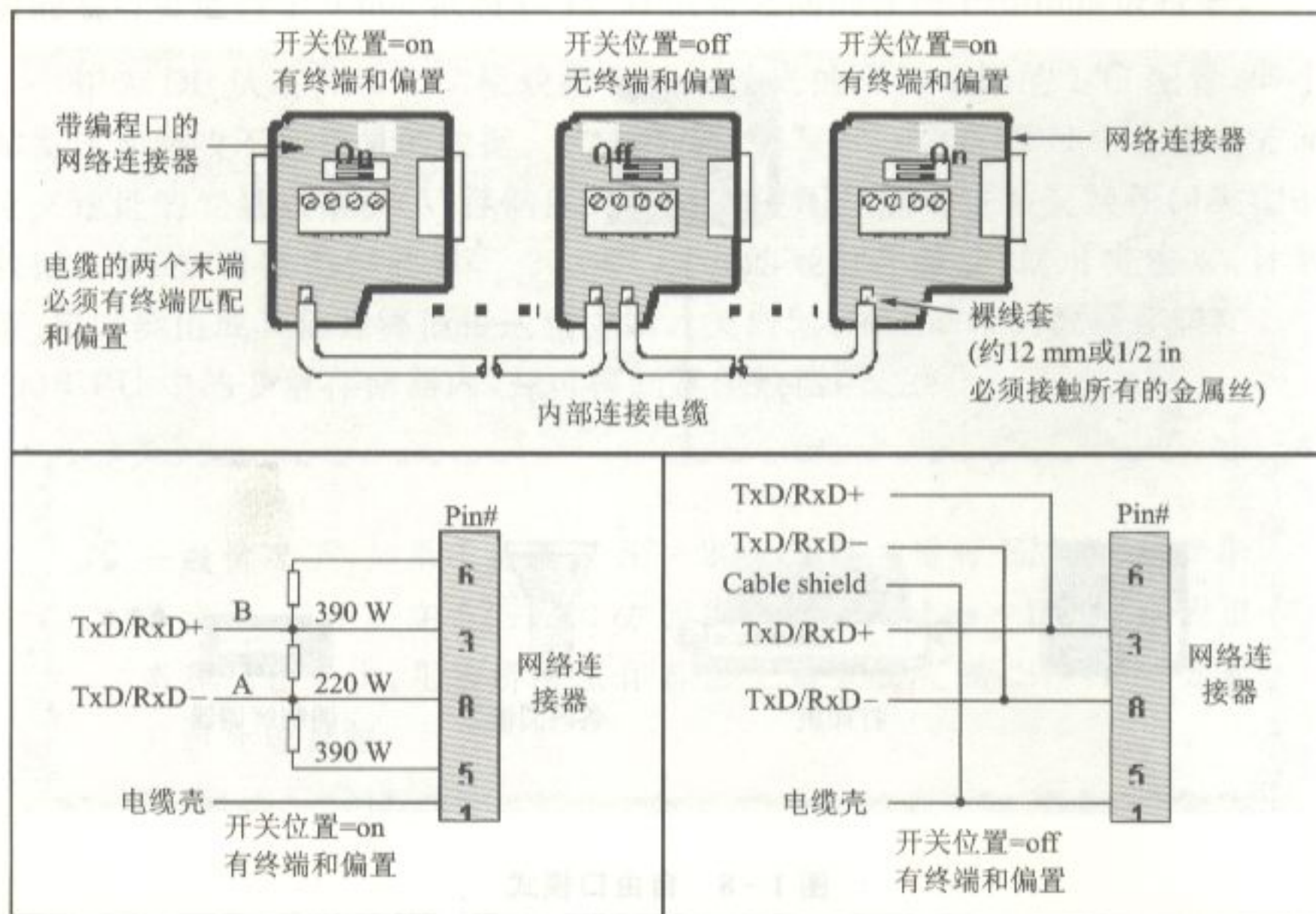


图 1-9 网络连接器和网络电缆示意图



SIEMENS 提供多种网络连接器,一些连接器仅提供连接到 CPU 的接口,而另一些连接器增加了一个编程接口。增加的编程接口可以用来连接运行 Step7 - Micro/WIN32 的计算机,或者 TD200 文本显示器等人机界面设备。

1.4.6 以太网通讯

S7-200 CPU 加装 CP243-1(CP243-1 IT)扩展模块可以支持工业以太网通讯。CP243-1(或 CP243-1 IT)模块提供了一个标准的 RJ45 网络接口,完全支持 TCP/IP 协议,支持标准的网络设备(如集线器、路由器等)(如图 1-10)。

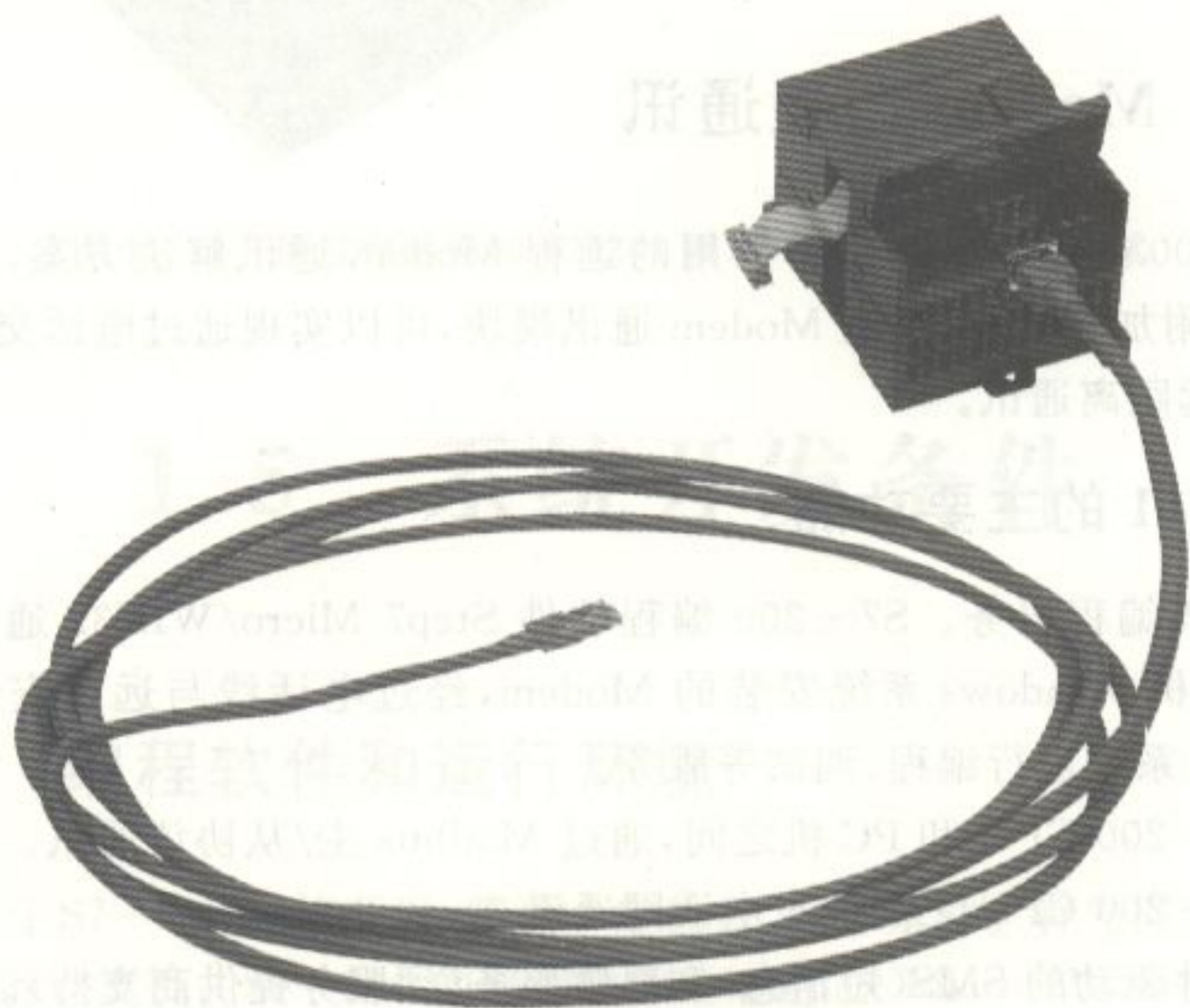


图 1-10 EM243-1 模块通过 RJ-45 接口连接工业以太网电缆



通过在 CPU 上扩展 CP243-1(或 CP243-1 IT)模块,您可以:

- 支持 10/100 Mbit/s 工业以太网、支持半双工/全双工通讯、TCP/IP, 最多 8 个连接;
- 与运行 Step7 - Micro/WIN32 的计算机通讯,支持通过工业以太网的远程编程服务;
- 连接其他 SIMATIC S7 系列远程组件,例如,S7-300 上的 CP343-1, 或其他 CP243-1;
- 连接基于 OPC 的 PC 应用程序,如组态软件等;
- 支持集成的 Web 网页服务,E-mail 服务等(CP243-1 IT)。



使用 Step7 - Micro/WIN32 V3.2 以上版本中的 Ethernet Wizard,可以方便地组态 CP243-1。

1.4.7 Modem 远程通讯

S7-200 提供了一个简单易用的远程 Modem 通讯解决方案。S7-200 CPU 通过附加 EM241 扩展 Modem 通讯模块,可以实现通过电话交换机和电话网络的远距离通讯。

EM241 的主要功能

- 远程编程服务。S7-200 编程软件 Step7 Micro/WIN32 通过在本地的 PC 机 Windows 系统安装的 Modem,经过电话线与远程安装的 S7-200 系统进行编程、调试等服务。
- S7-200 CPU 和 PC 机之间,通过 Modbus 主/从协议通讯。
- S7-200 CPU 之间通过电话网通讯。
- 事件驱动的 SMS(短消息)和寻呼服务(须服务提供商支持)。



Step7 - Micro/WIN32 V3.2 版或以上, 提供了一个 Modem Expansion Wizard, 用于组态包括 EM241 在内的 Modem 应用。

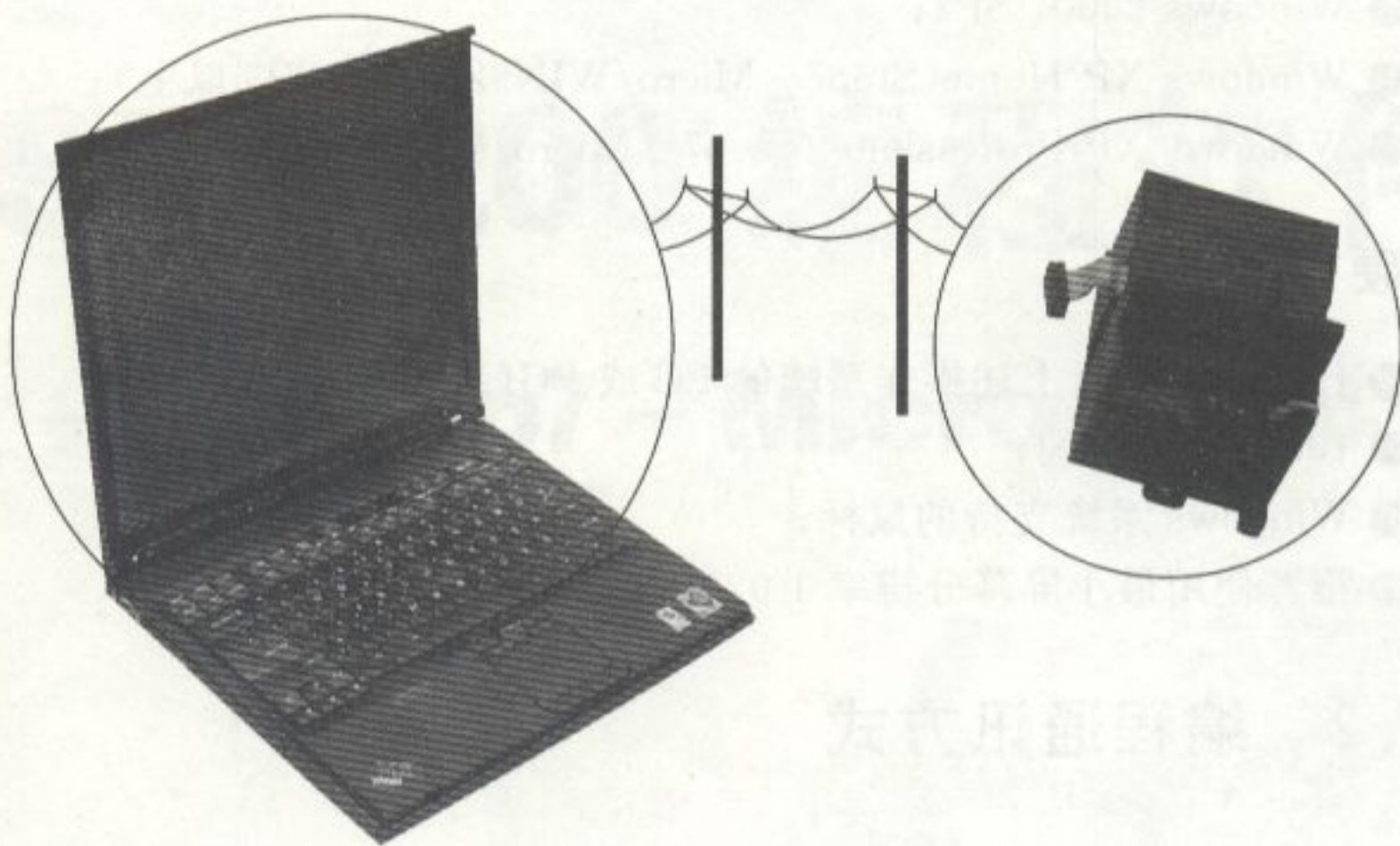


图 1-11 通过电话网和 EM241 模块可进行远程编程调试

1.5 系统开发条件

1.5.1 编程软件和运行环境

要进行 S7-200 系统开发, 需要一定的软、硬件条件。首先需要能够运行编程软件的计算机。S7-200 的编程软件是 Step7 - Micro/WIN32, 目前的版本是 Step7 - Micro/WIN32 V3.2 系列。

Step7 - Micro/WIN32 V3.2 可以在 Microsoft 公司出品的如下操作系统环境下安装:



- Windows 95;
- Windows 98, 2nd Edition;
- Windows NT, Version 4.0, SP6;
- Windows Me Edition;
- Windows 2000, SP2;
- Windows XP Home(Step7 - Micro/WIN32 V3.2 SP3 以上);
- Windows XP Professional(Step7 - Micro/WIN32 V3.2 SP3 以上)。

硬件要求

- 任何能够运行上述操作系统的 PC 或 PG(编程器);
- 100M 硬盘空间;
- Windows 系统支持的鼠标;
- 推荐使用最小屏幕分辨率 1 024×768, 小字体。

1.5.2 编程通讯方式

要对 S7-200 CPU 进行编程、调试, 还需要连接运行编程软件的 PC 机和 S7-200 CPU 的通讯连接。一般使用如下几种编程通讯方式:

- RS-232 PPI 电缆, 连接 PG/PC 的串行通讯口(COM 口)和 CPU 通讯口;
- Smart RS-232/PPI 电缆, 连接 PG/PC 的串行通讯口(COM 口)与 CPU 通讯口(Step7 - Micro/WIN32 V 3.2 SP4 以上);
- Smart USB/PPI 电缆, 连接 PG/PC 的 USB 端口和 CPU 通讯口(Step7 - Micro/WIN32 V 3.2 SP4 以上);
- PG/PC 上安装 CP(通讯处理器)卡, 通过 MPI 电缆连接 CPU 通讯口。
(PCI 接口卡 CP5611 配合台式 PC 使用; PCMCIA 卡 CP5511 配合便携机使用)



使用 EM241, 可以在正确安装了普通 Modem 的计算机上, 通过电话网直接与扩展了 EM241 的 S7-200 CPU 进行编程连接, 而无须设置其他参数。

第 2 章

S7 — 200 编程软件

——Step7 — Micro/WIN32





2.1 软件安装和设置

2.1.1 安装条件

Step7 - Micro/WIN32 V3.2 可以在 Microsoft 公司出品的如下操作系统环境下安装:

- Windows 95;
- Windows 98, 2nd Edition;
- Windows NT, Version 4.0, SP6;
- Windows Me Edition;
- Windows 2000, SP2;
- Windows XP Home (Step7 - Micro/WIN32 V3.2 SP3 以上);
- Windows XP Professional (Step7 - Micro/WIN32 V3.2 SP3 以上)。

硬件要求

- 任何能够运行上述操作系统的 PC 或 PG (编程器);
- 100M 硬盘空间;
- Windows 系统支持的鼠标;
- 推荐使用最小屏幕分辨率 1 024×768, 小字体。

2.1.2 安 装

正版 Step7 - Micro/WIN32 V3.2 软件光盘和包装盒

正版软件光盘和包装盒如图 2-1 所示。



图 2-1 正版 Step7 - Micro/WIN32 V3.2 软件光盘和包装盘



要在 Microsoft Windows NT、Windows 2000 或 Windows XP 操作系统下安装 Step7 - Micro/WIN 32, 必须有管理员 (administrator) 权限; 在上述操作系统下使用 Step7 - Micro/WIN 32, 至少需要 Power User 权限。

安装步骤

- 关闭所有应用程序, 包括 Microsoft Office 快捷工具栏; 在光盘驱动器内插入安装光盘。如果没有禁止光盘插入自动运行, 安装程序会自动运行; 或者在 Windows 资源管理器中打开安装光盘上的“Setup.exe”文件;



- 按照安装程序的提示完成安装。

☞ 运行 Setup 程序, 选择安装程序界面语言, 缺省使用英语(如图 2-2 所示)。

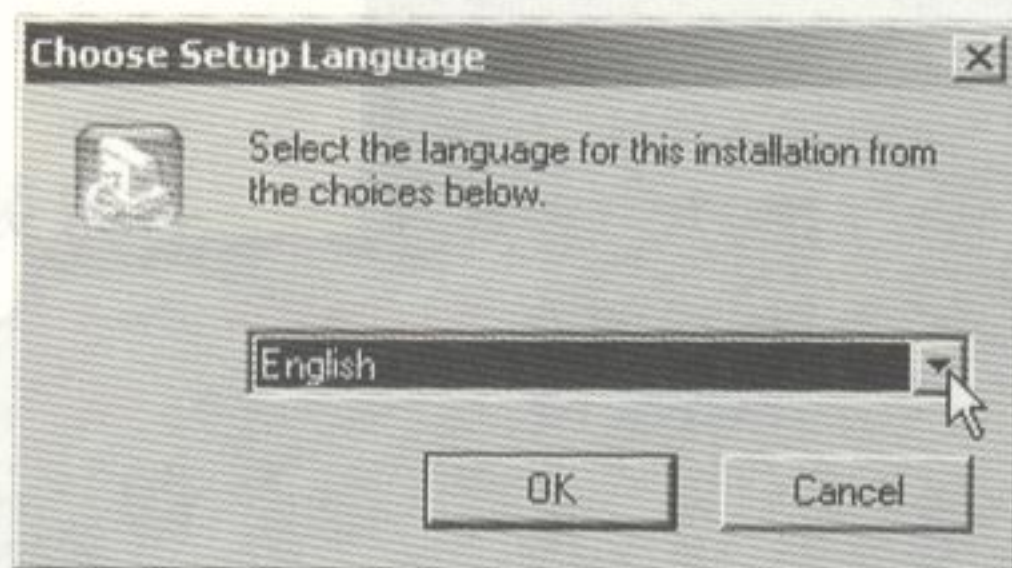


图 2-2 选择安装程序界面语言

☞ 输入用户信息(如图 2-3 所示)。

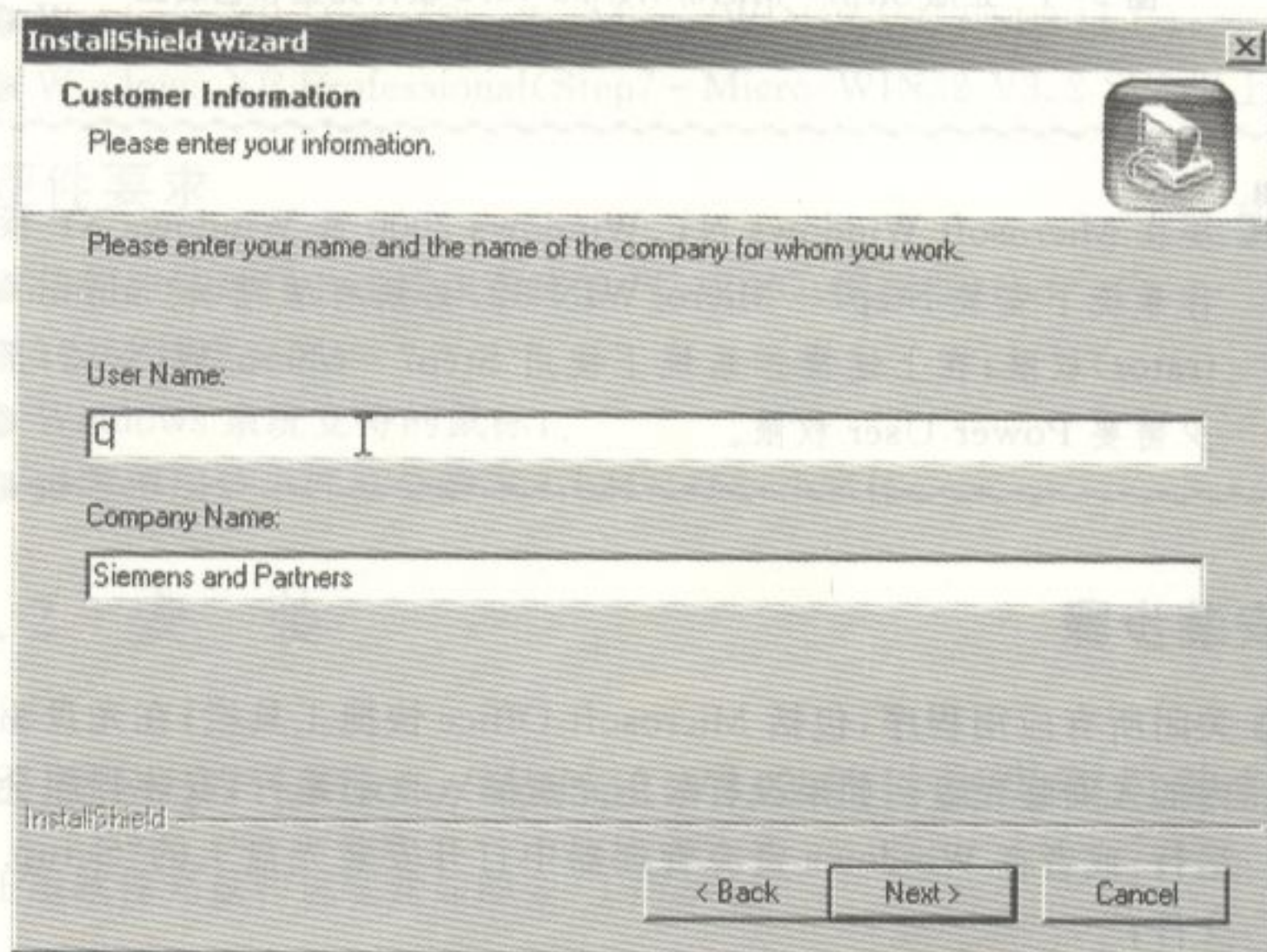
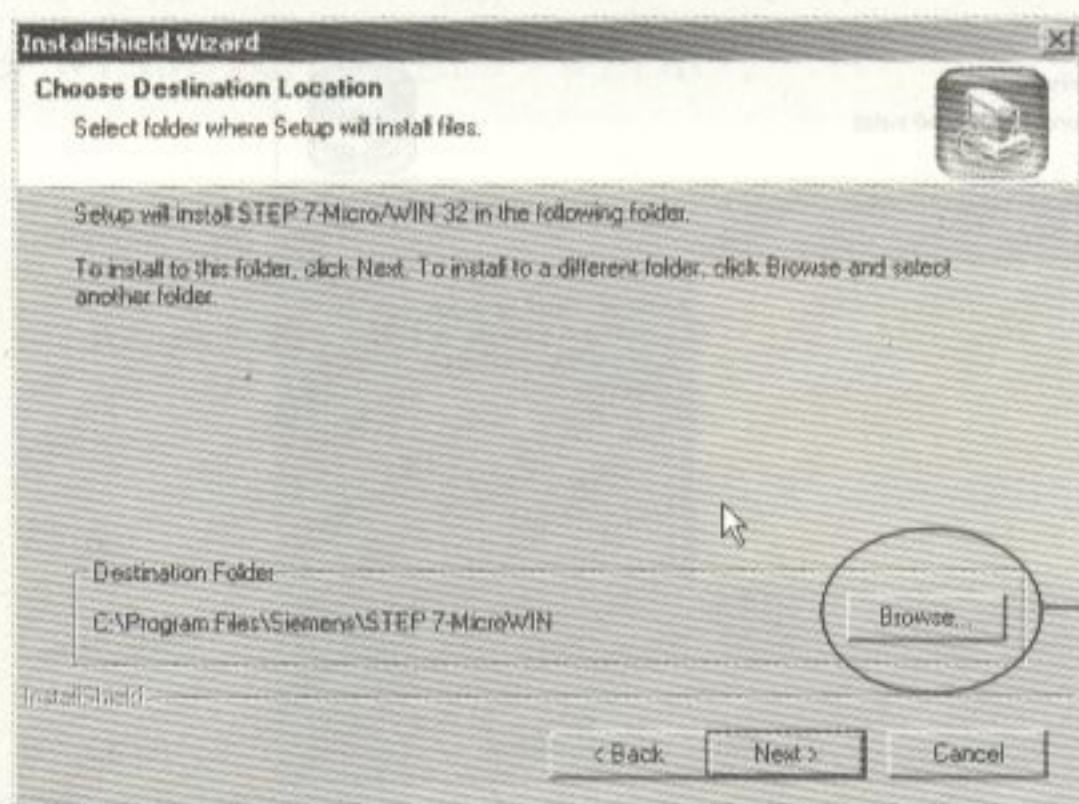


图 2-3 输入用户信息



☞ 选择安装目的文件夹(如图 2-4 所示)。



按此按钮在弹出对话框中指定其他文件夹

图 2-4 选择安装的文件夹

☞ 选择“All Languages”安装所有语言版本;选择“Select Languages”挑选程序界面语言(如图 2-5 所示)。

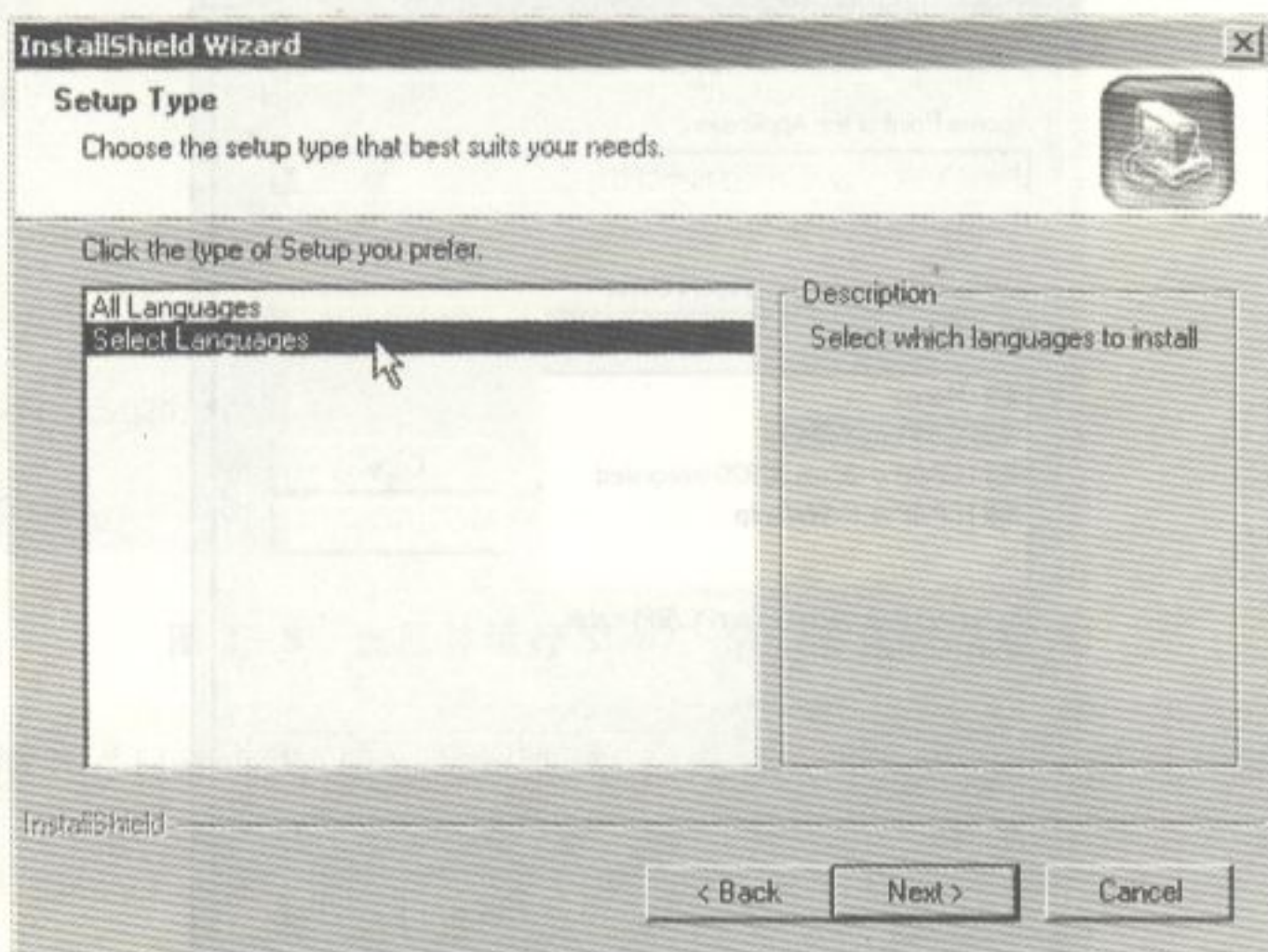


图 2-5 挑选程序界面语言



☞ 选择所需语言(如图 2-6 所示)。

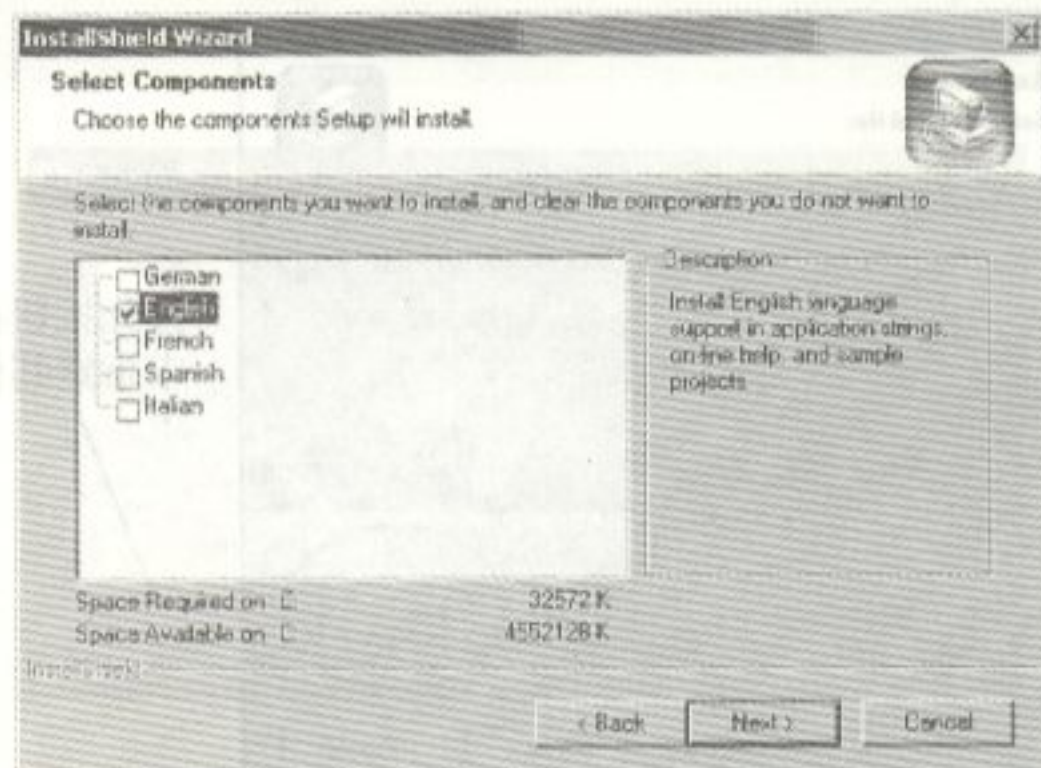


图 2-6 选择所需语言

☞ 安装过程中,会出现“Set PG/PC Interface”窗口,按“OK”(如图 2-7 所示)。

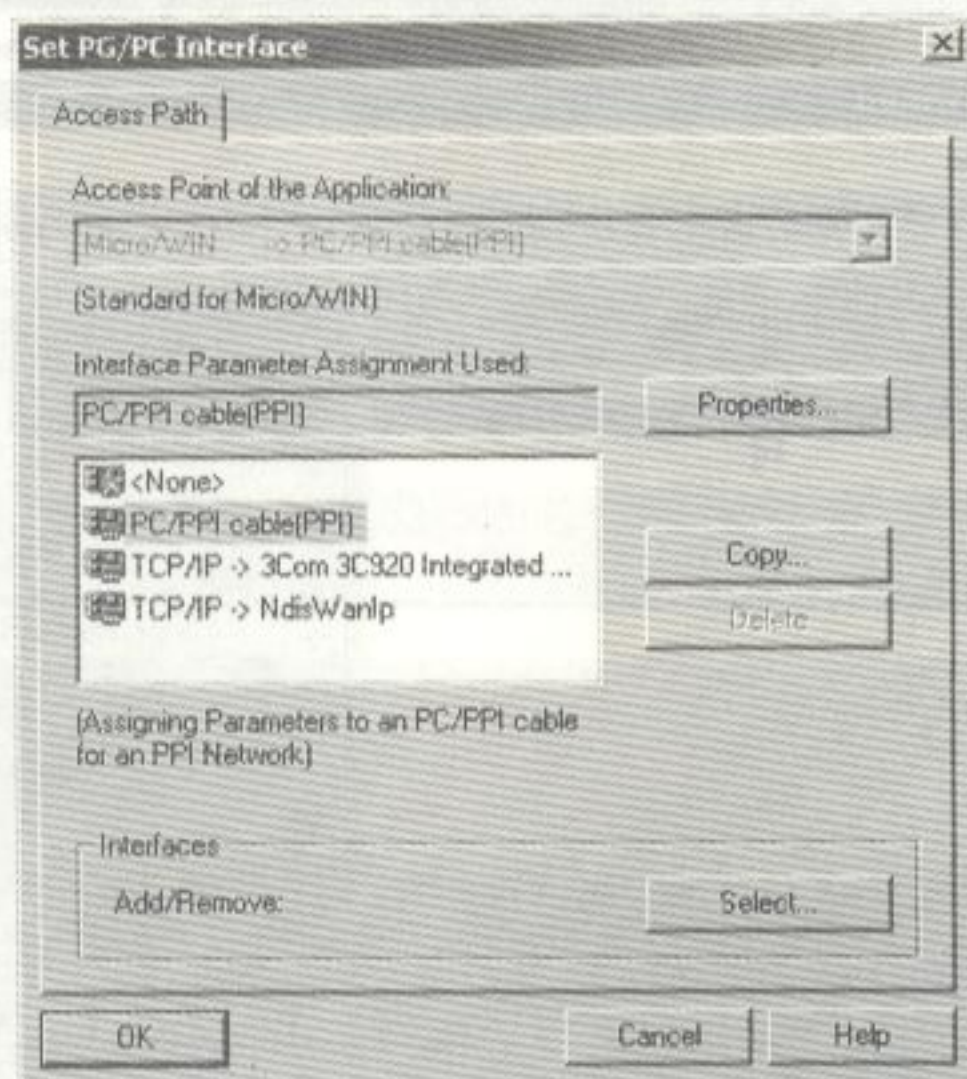


图 2-7 “Set PG/PC Interface”窗口



完成安装后,用鼠标双击 Windows 桌面上的 Step7-Micro/WIN32 图标,或者在 Windows 的“开始”菜单找到相应的快捷方式,运行 Step7-Micro/WIN32 软件(如图 2-8 所示)。

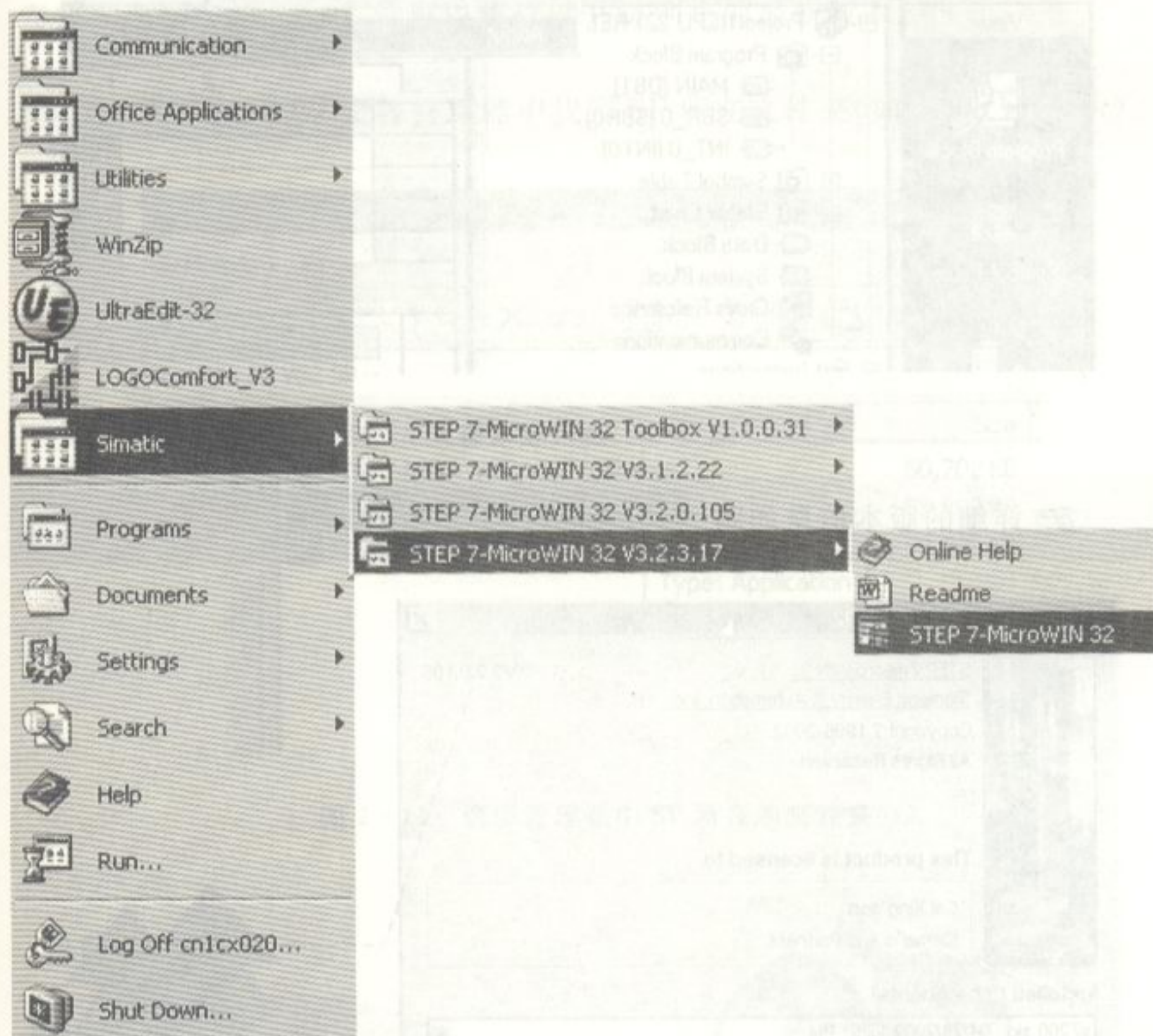


图 2-8 选取并运行 Step7-Micro/WIN32 软件

使用“Help>About”的菜单命令查看软件版本信息(如图 2-9 所示)。

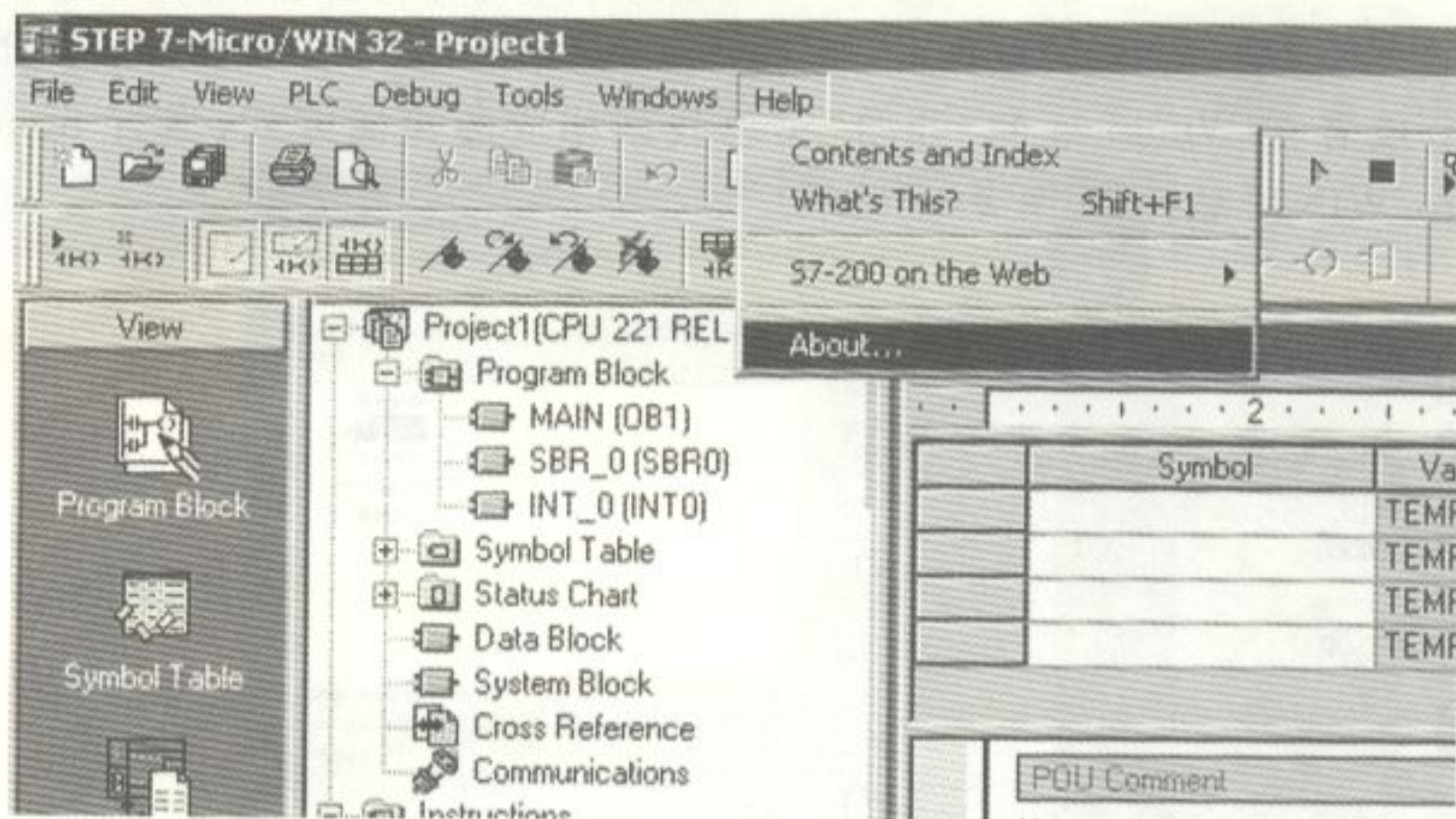
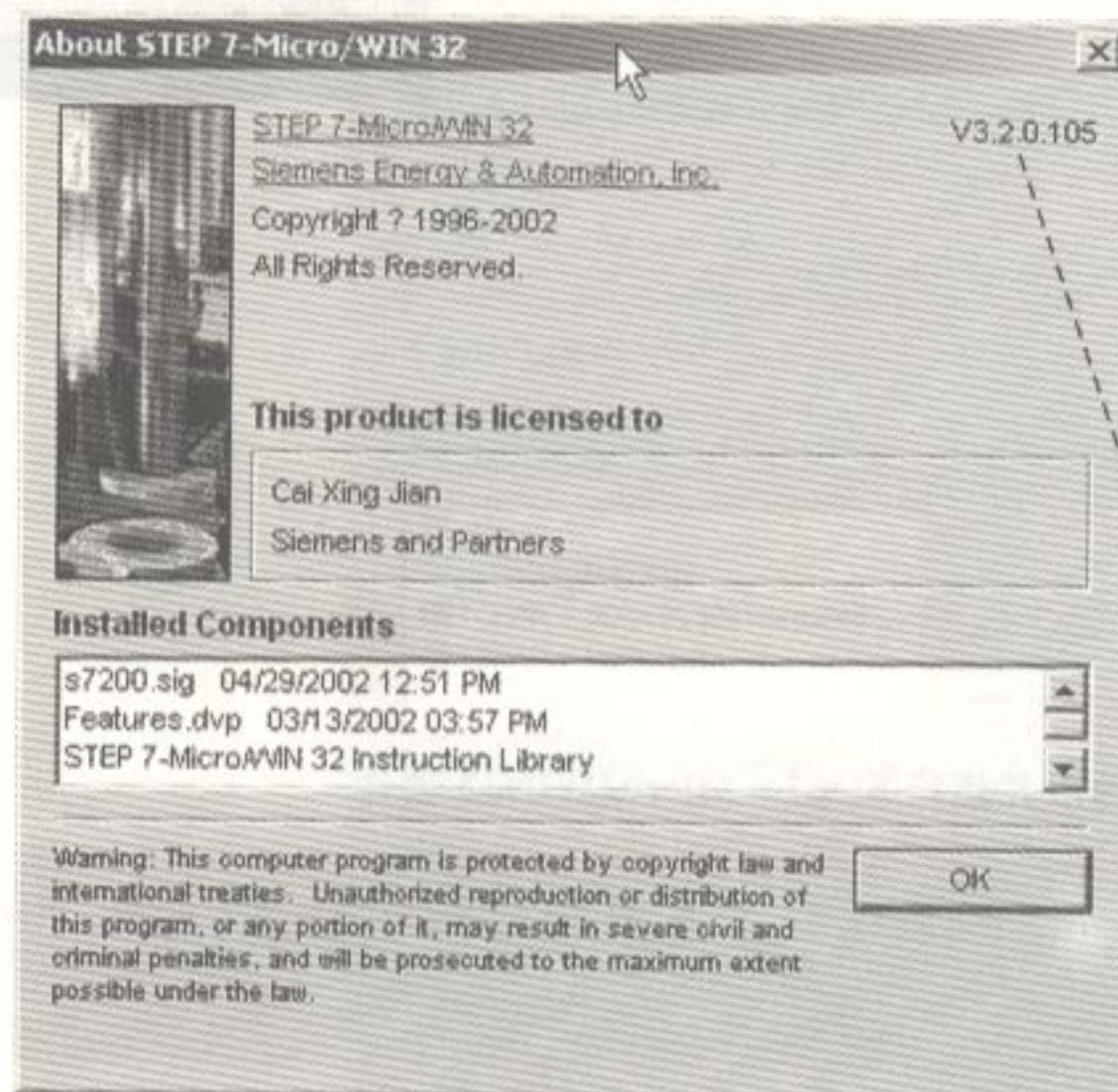


图 2-9 查看版本信息

☞ 详细的版本信息如图 2-10 所示。



详细的版本信息

图 2-10 详细的版本信息



2.1.3 安装 SP 升级包(Service Pack)

Step7 - Micro/WIN32 的 SP 升级包可从西门子互联网站上下载。只须安装一次最新的 SP 包,就可以将软件升级到当前最新版本。

☞ 在 Windows 资源管理器中找到 SP 所在文件夹(如图 2-11 所示)。

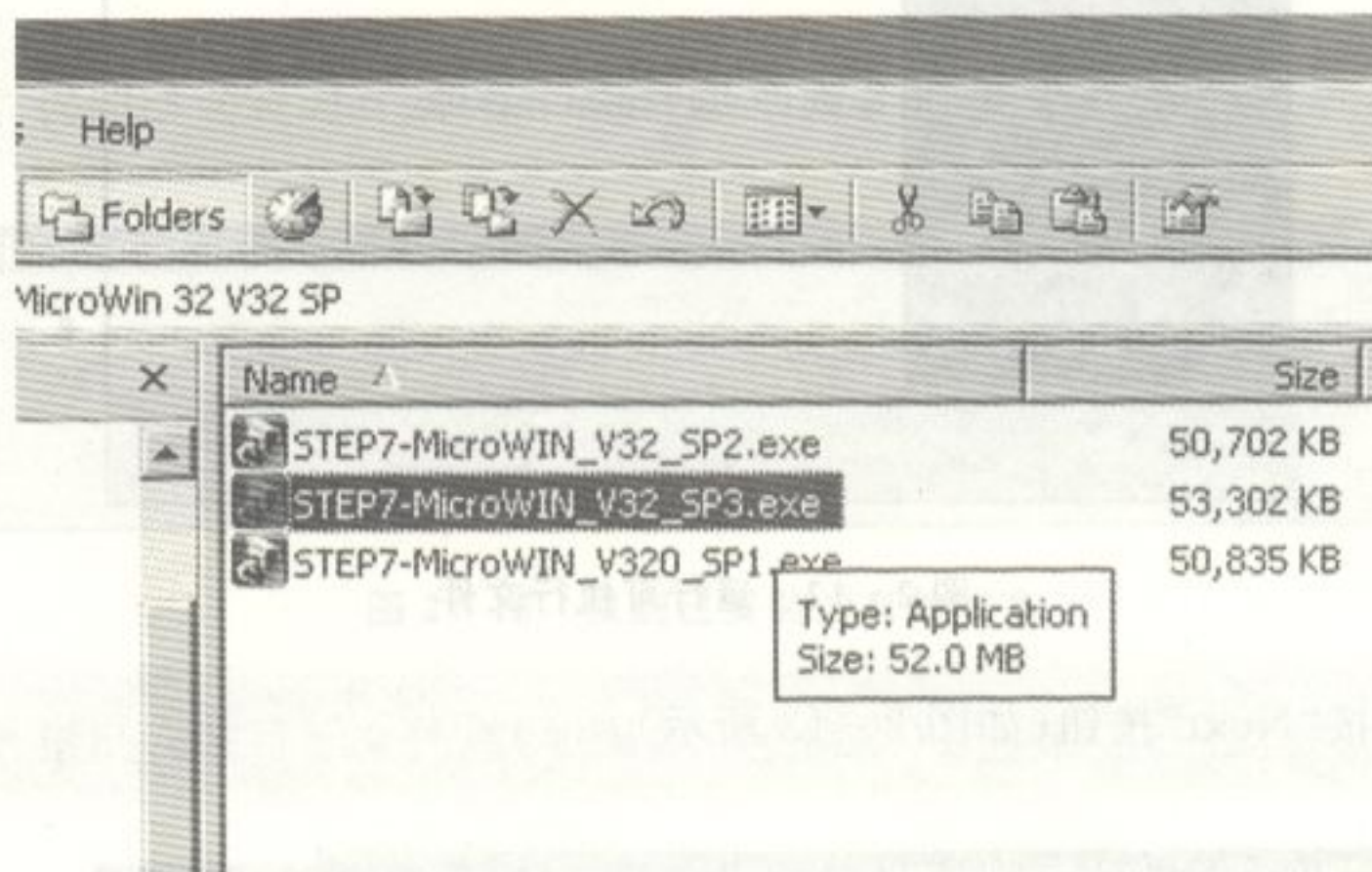
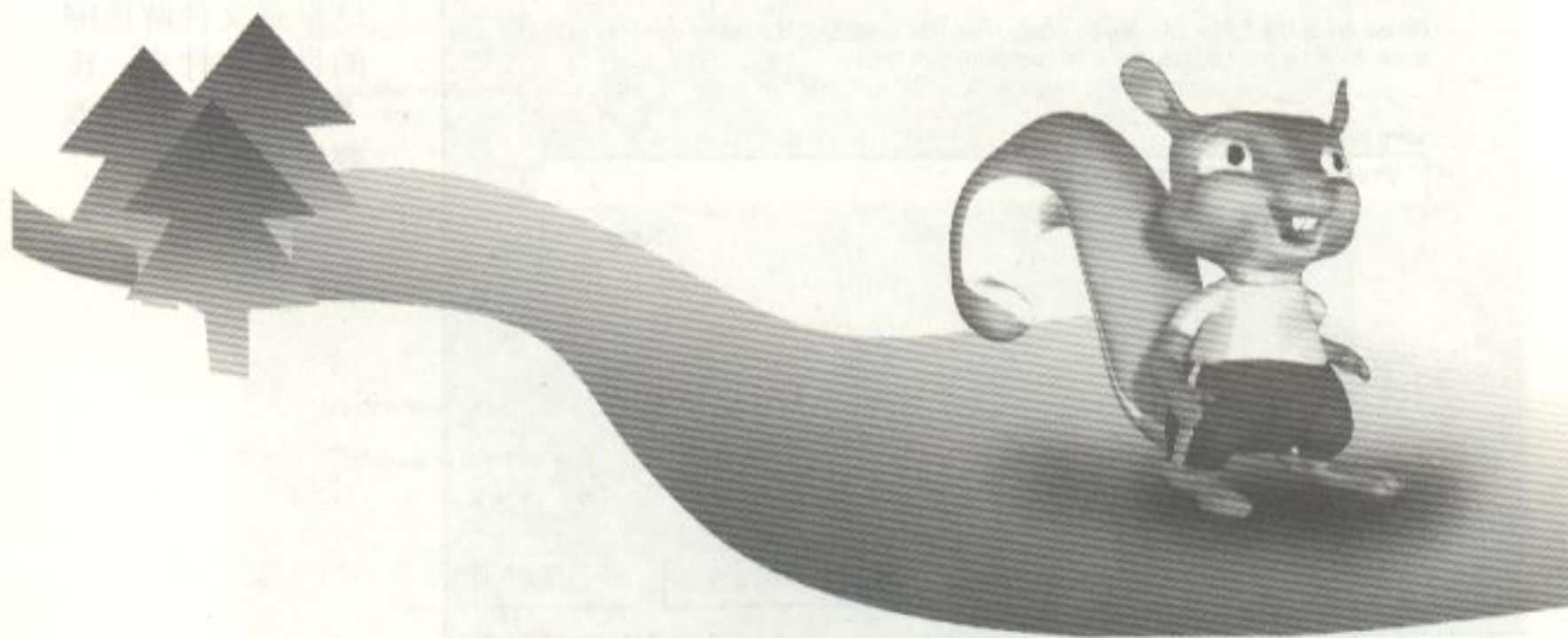


图 2-11 资源管理器中 SP 所在的文件夹



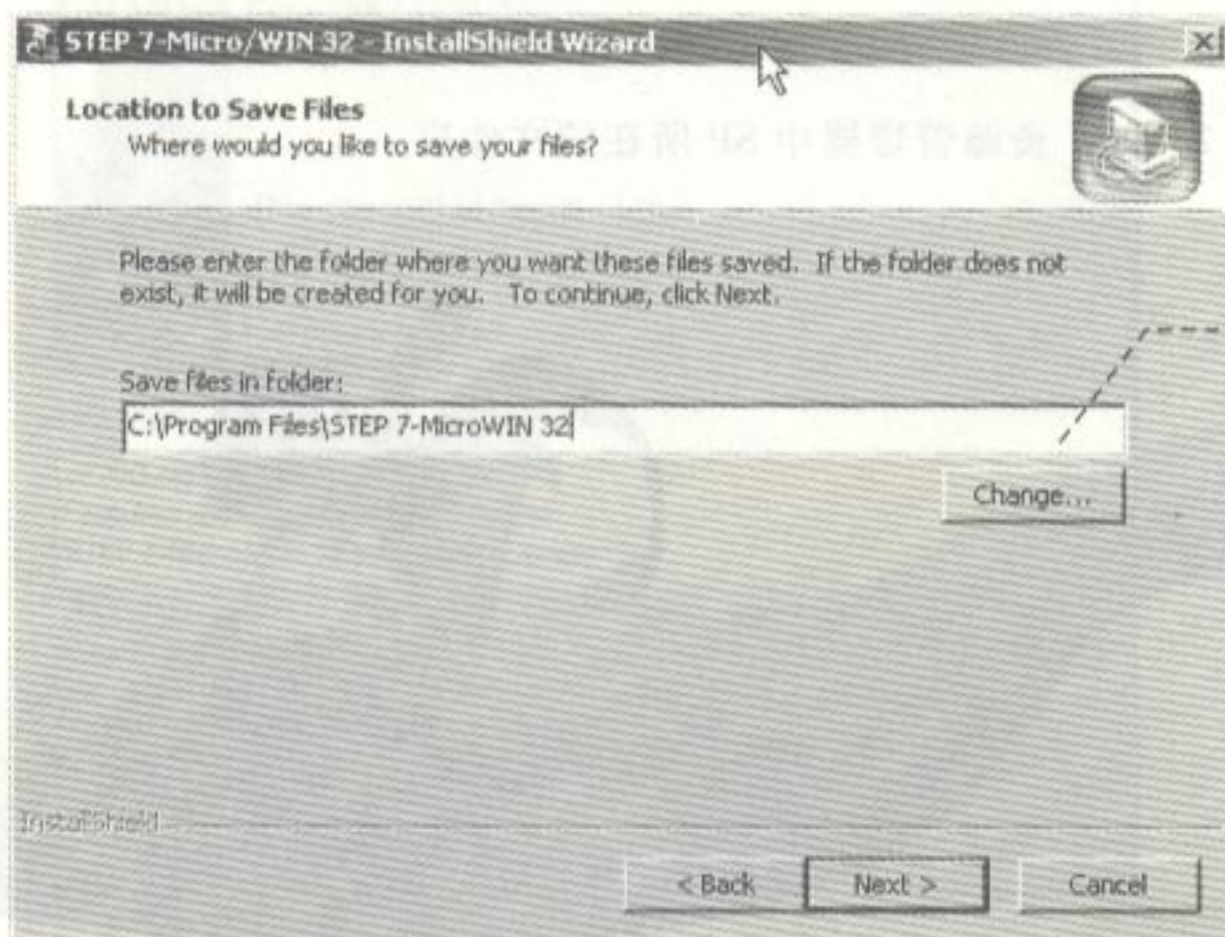


☞ 运行相应的可执行文件(如图 2-12 所示)。



图 2-12 运行可执行文件

☞ 按“Next”按钮(如图 2-13 所示)。



这里是文件解压缩的目标文件夹；按“Change”按钮改变缺省文件夹

图 2-13 选“Next”继续升级操作



☞ 解压缩完成后会自动执行安装。首先寻找已在本地硬盘上安装的有效版本(如图 2-14 所示)。

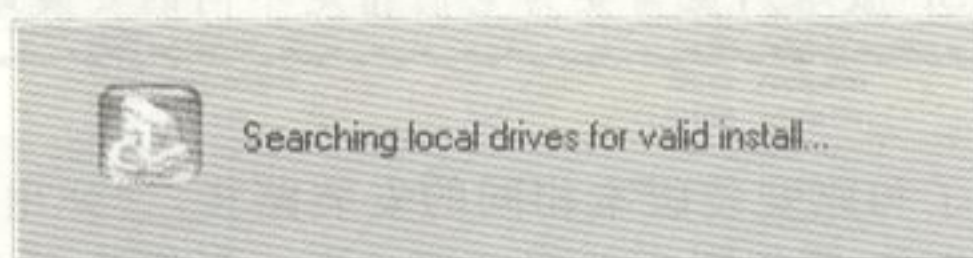


图 2-14 寻找本地硬盘上安装的有效版本

☞ 找到已安装的有效版本后显示(如图 2-15 所示)。



图 2-15 找到已安装的有效版本

☞ 指定安装目标文件夹(如图 2-16 所示)。

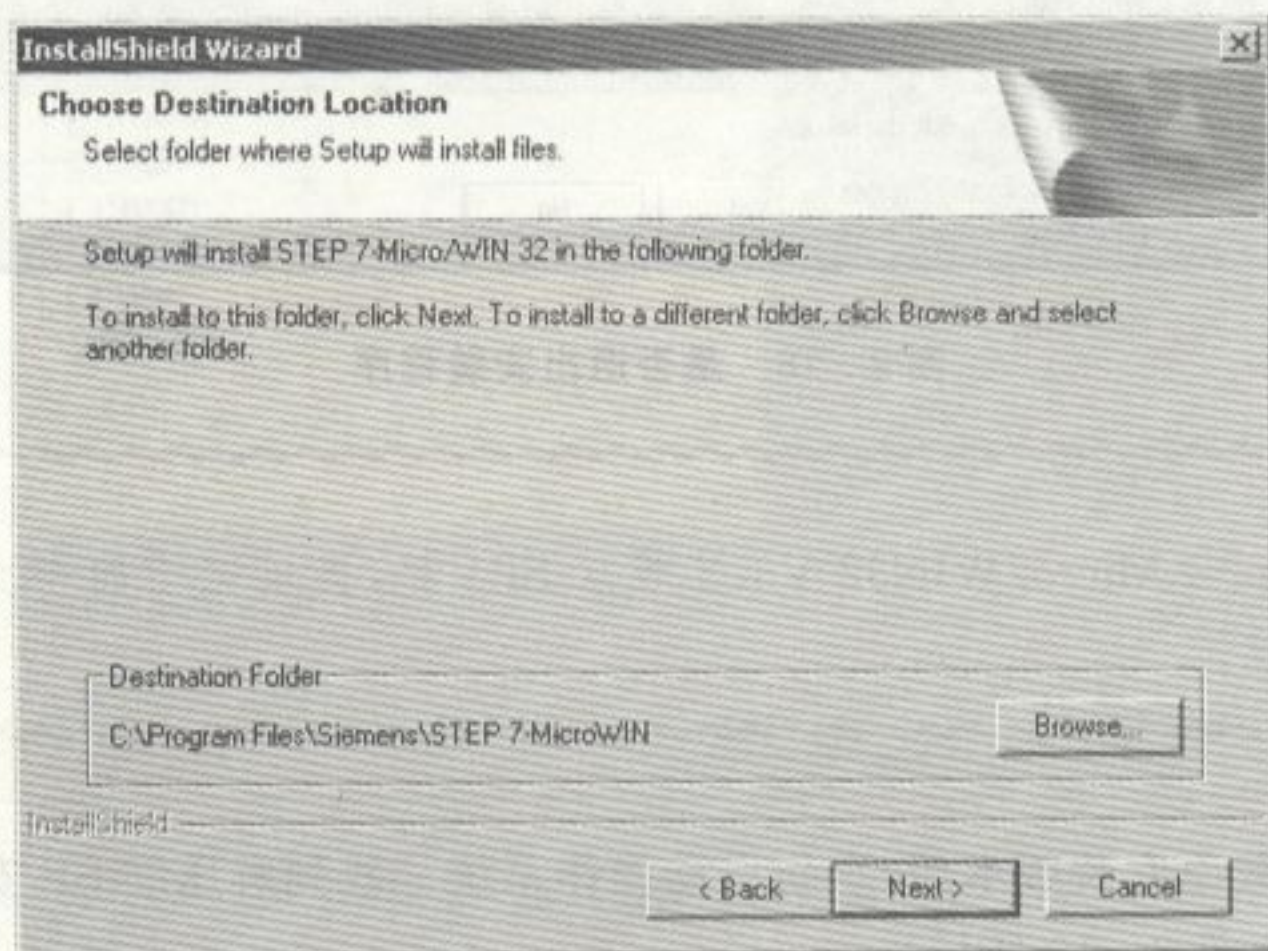


图 2-16 选取安装目标文件夹



Step7 - Micro/WIN32 支持在一个操作系统内安装多个版本(包括不同的 SP 版本),但需要为它们指定不同的安装路径。

☞ 如果指定了和原安装版本相同的安装路径,会弹出一个消息框。要在 Windows 的“安装/卸载程序”中卸载原先版本(如图 2-17 所示)。再次安装 SP 前,必须重新启动操作系统。

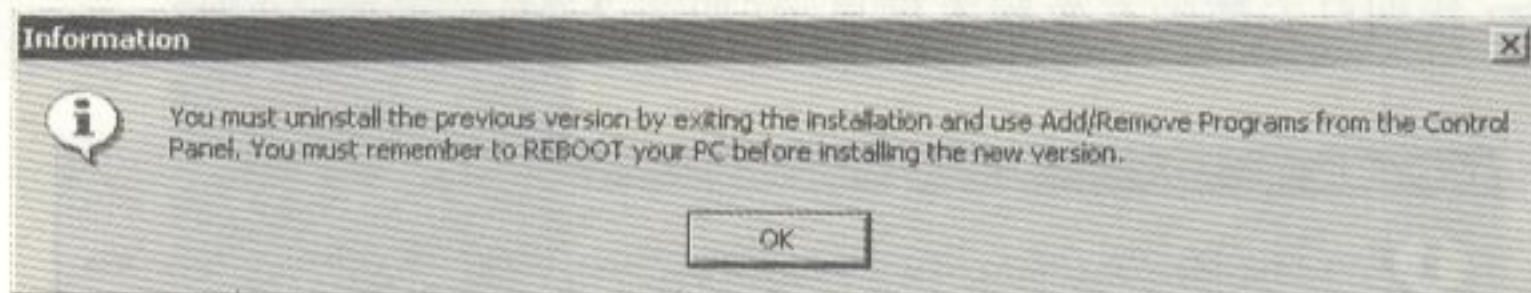


图 2-17 卸载原始版本提示

☞ 按“Yes”确认退出安装程序(如图 2-18 所示)。

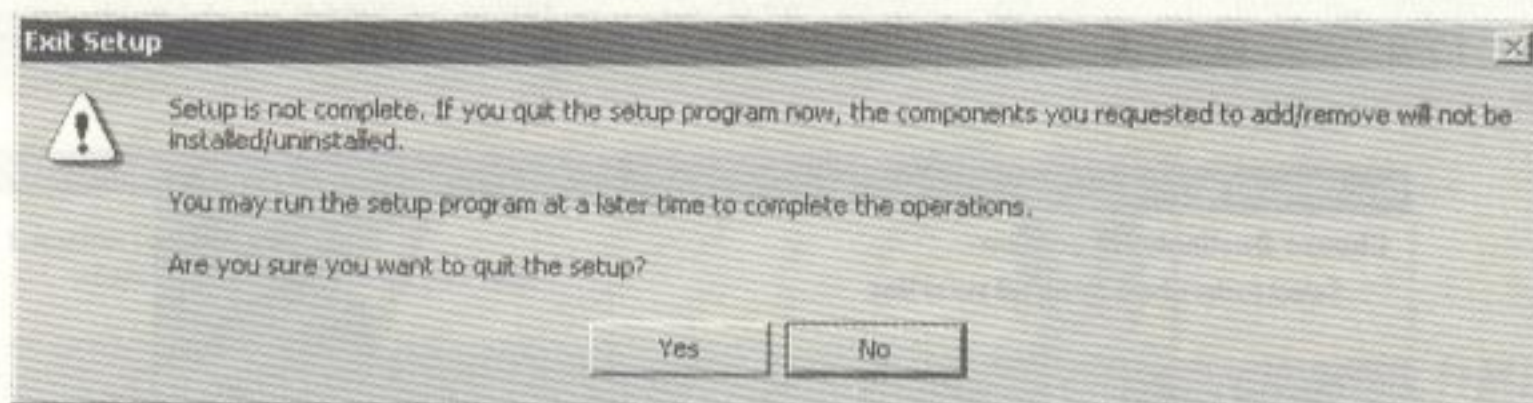


图 2-18 是否退出安装程序



Step7 - Micro/WIN32 V3.2 版从 SP1 起,提供完全的中文编程环境。

☞ 卸载原安装版本后,重新启动操作系统。找到解压缩 SP 安装包文件的文件夹(如图 2-19 所示)。

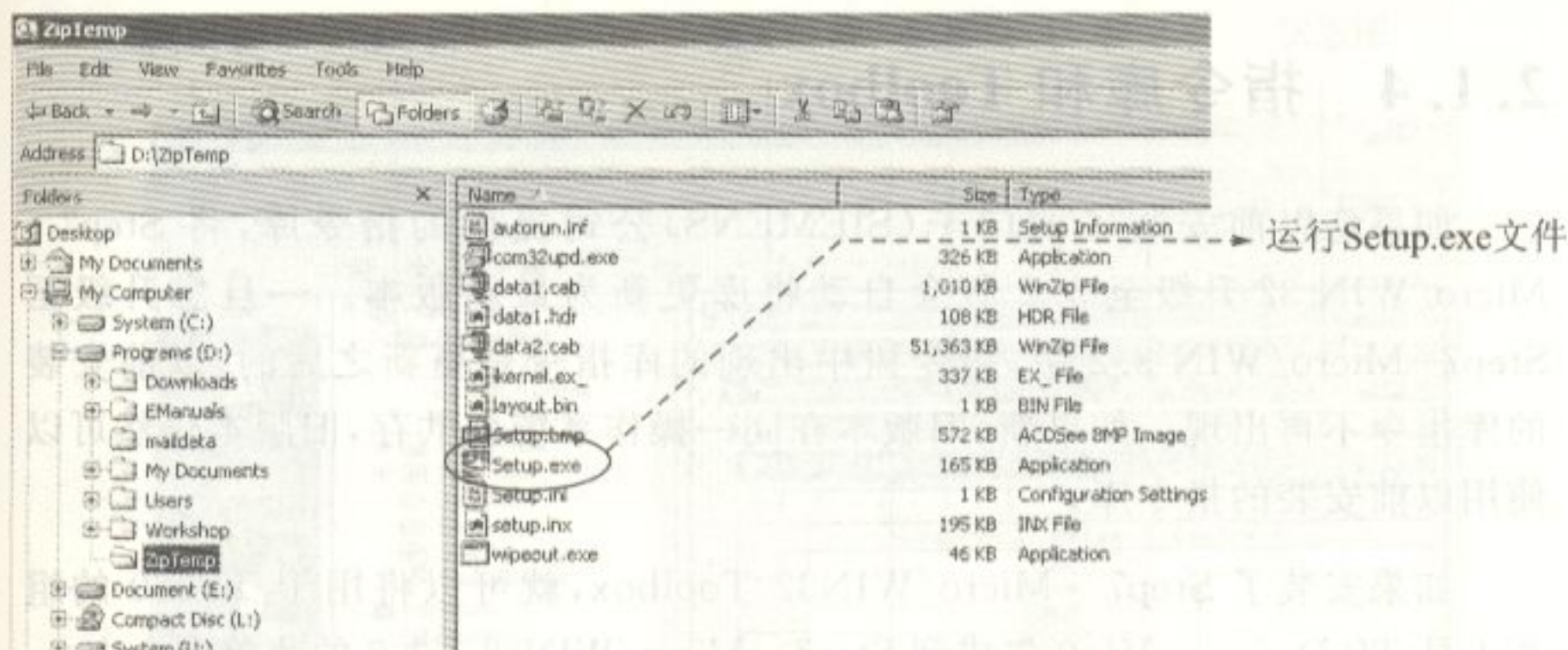


图 2-19 找到 SP 解压缩的文件夹

继续安装，您可以选择 Step7 - Micro/WIN32 支持的语言（如图 2-20 所示）。

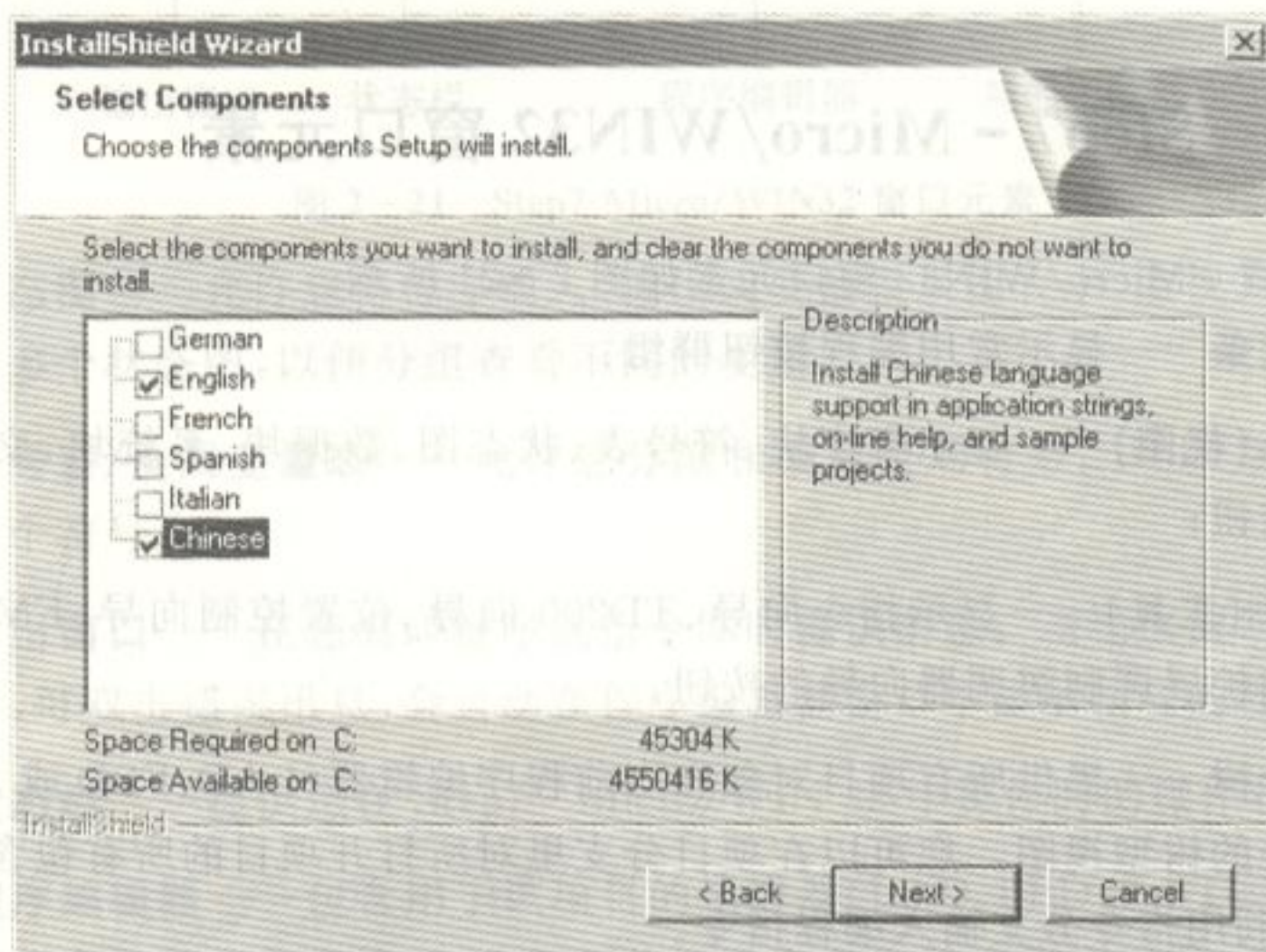


图 2-20 选择支持语言



2.1.4 指令库和 Toolbox

如果您以前安装了西门子(SIEMENS)公司提供的指令库,将 Step7 - Micro/WIN 32 升级至 3.2 版会自动将库更新为最新版本。一旦您升级至 Step7 - Micro/WIN 3.2 版,指令树中出现的库指令是更新之后的,以前安装的库指令不再出现。如果新、旧版本在同一操作系统内共存,旧版本仍然可以使用以前安装的指令库。

如果安装了 Step7 - Micro/WIN32 Toolbox,就可以将用于 TP070 的组态工具 TP Designer V1.0 集成到 Step7 - Micro/WIN32 V3.2 的菜单命令中。

2.2 Step7 - Micro/WIN32 简介

2.2.1 Step7 - Micro/WIN32 窗口元素

Step7 - Micro/WIN32 窗口元素如图 2-21 所示。

浏览条——显示常用编程按钮群组:

View(视图)——显示程序块、符号表、状态图、数据块、系统块、交叉参考及通讯按钮;

Tools(工具)——显示指令向导、TD200 向导、位置控制向导、EM 253 控制面板和扩展调制解调器向导的按钮。

指令树——提供所有项目对象和当前程序编辑器(LAD、FBD 或 STL)的所有指令的树型视图。您可以在项目分支里对所打开项目的所有包含对象进行操作;利用指令分支输入编程指令。

交叉参考——查看程序的交叉引用和元件使用信息。

数据块——显示和编辑数据块内容。



浏览条 指令树 交叉参考 数据块 符号表 状态图

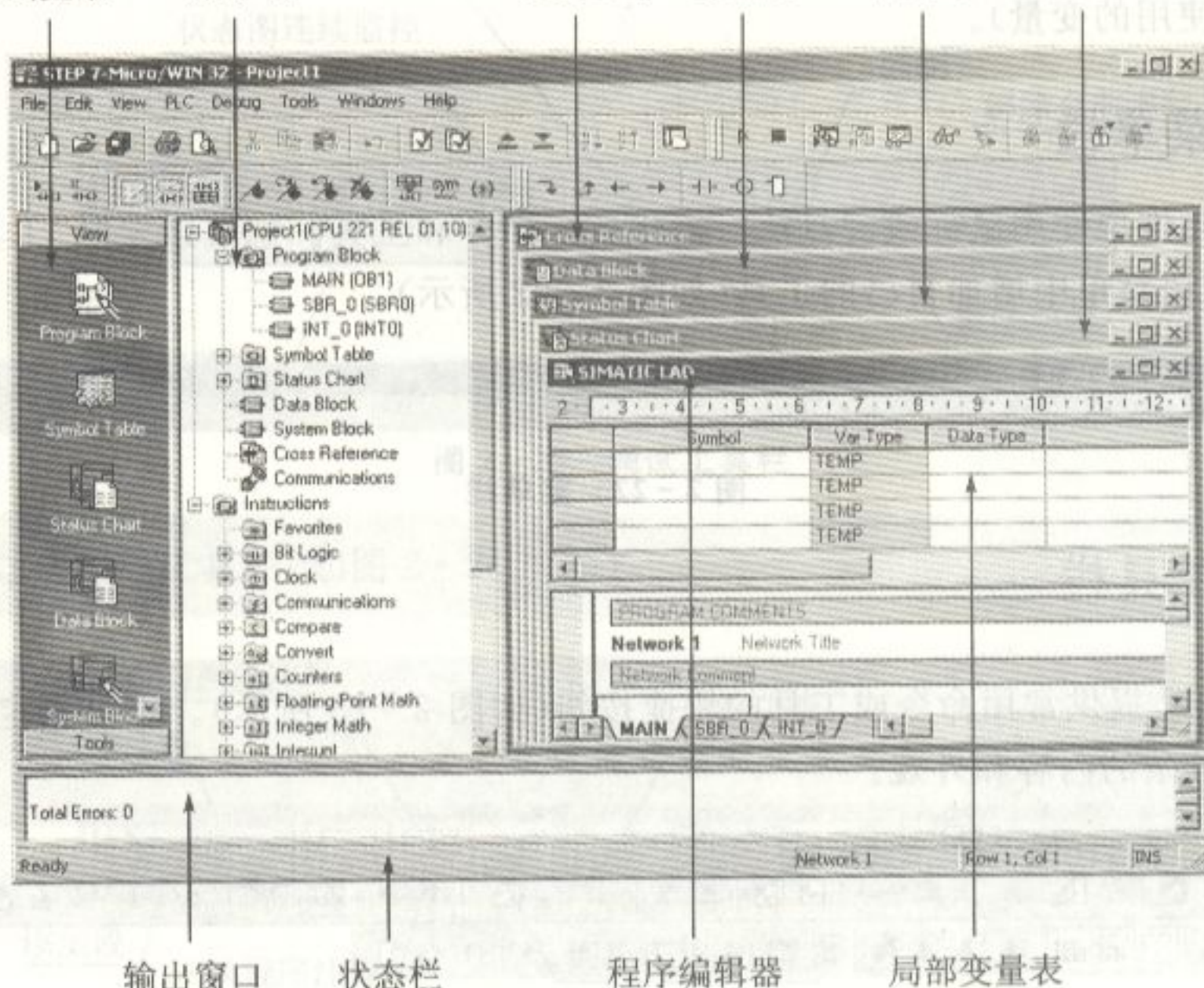


图 2-21 Step7 Micro/WIN32 窗口元素

状态图——允许您将程序输入、输出或变量置入图表中，监视其状态。可以建立多个状态图，以便分组查看不同的变量。

符号表/全局变量表——允许您分配和编辑全局符号。可以为一个项目建立多个符号表。

输出窗口——在您编译程序或指令库时提供消息。当输出窗口列出程序错误时，可双击错误讯息，会自动在程序编辑器窗口中显示相应的程序网络。

状态栏——提供您在 Step7 - Micro/WIN 32 中操作时的操作状态信息。

程序编辑器——包含用于该项目的编辑器(LAD、FBD 或 STL)的局部变量表和程序视图。如果需要，您可以拖动分割条以扩充程序视图，并覆盖局部变量表。单击程序编辑器窗口底部的标签，可以在主程序、子程序和中断服务程序之间移动。



局部变量表——包含对局部变量所作的定义赋值(即子程序和中断服务程序使用的变量)。

菜单栏

允许您使用鼠标或键盘执行操作各种命令和工具。您可以定制“工具”菜单,在该菜单中增加自己的工具(如图 2-22 所示)。



图 2-22 菜单栏

工具栏

提供常用命令或工具的快捷按钮(如图 2-23 所示)。您可以定制每个工具条的内容和外观。



图 2-23 工具栏

标准工具栏(如图 2-24 所示)。

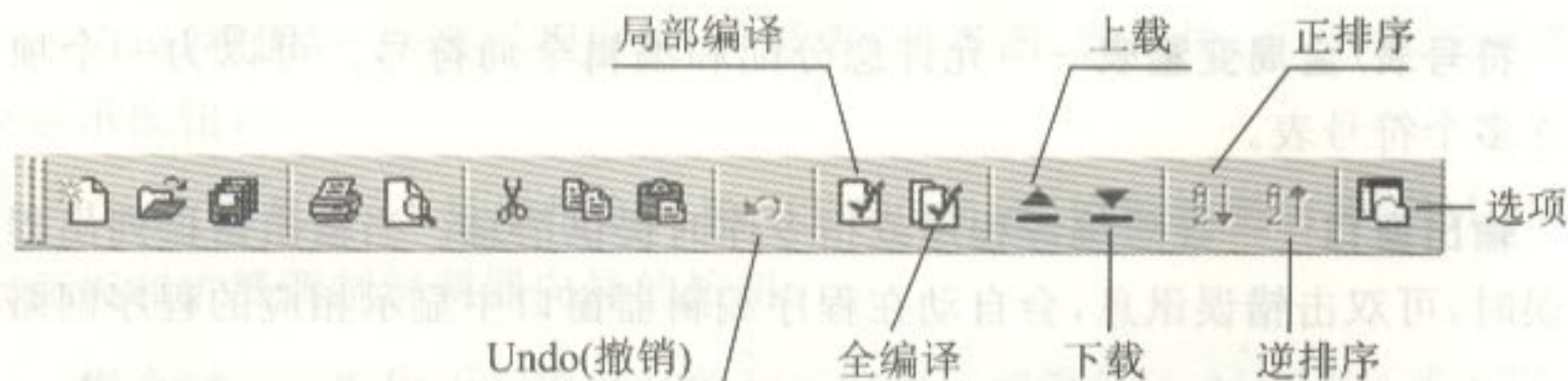


图 2-24 标准工具栏

调试工具栏(如图 2-25 所示)。

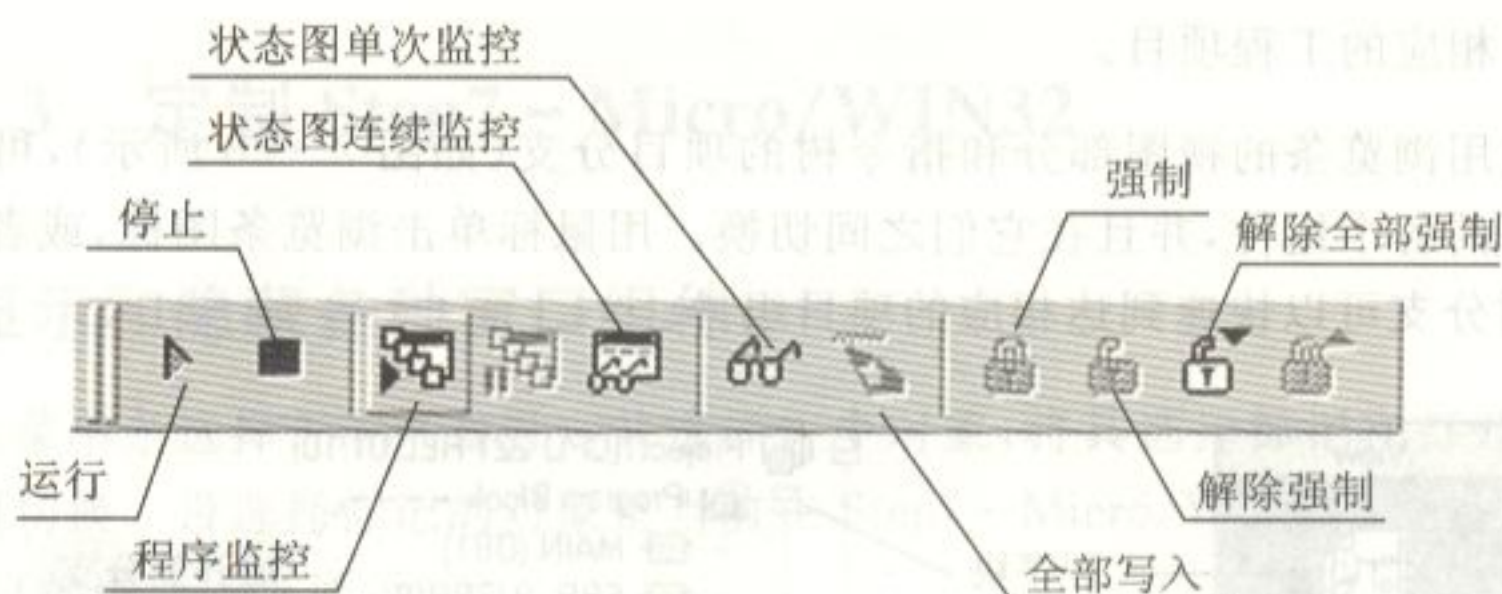


图 2-25 调试工具栏

☞ 常用工具栏(如图 2-26 所示)。

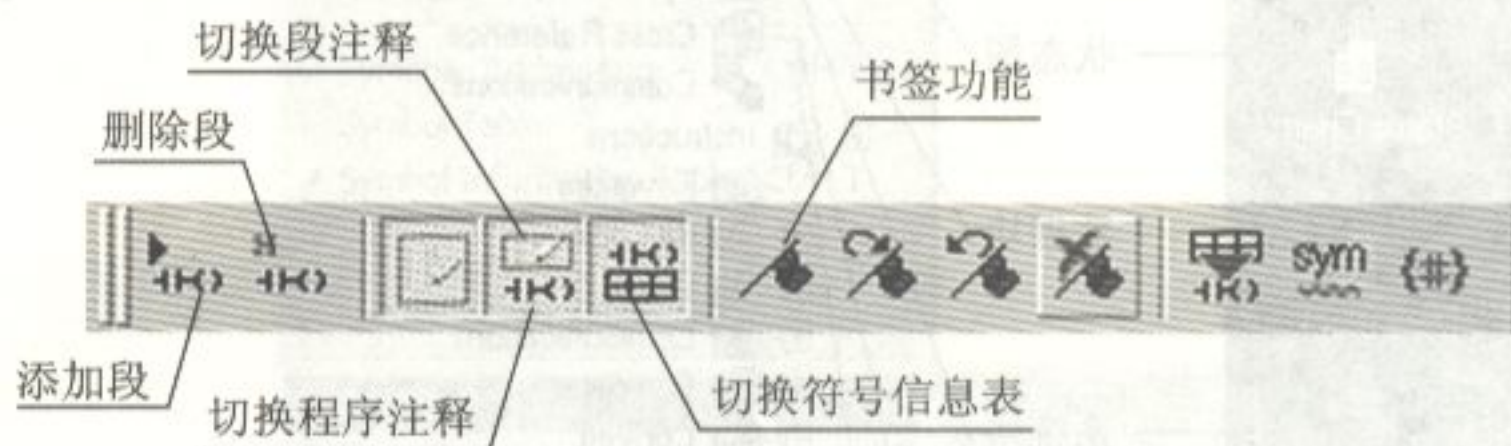


图 2-26 常用工具栏

☞ LAD 指令工具栏(如图 2-27 所示)。

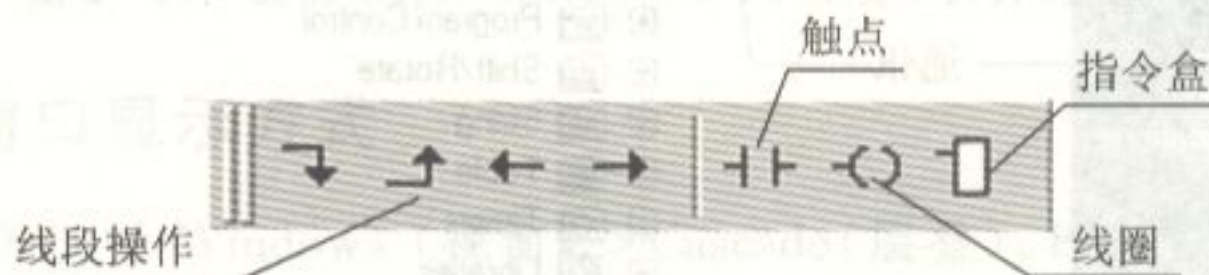


图 2-27 LAD 指令工具栏

2.2.2 项目及其组件

Step7-Micro/WIN32 为每个实际的 S7-200 系统的软件部分生成一个项目,项目以扩展名为 .mwp 的单一文件格式保存。打开一个 .mwp 文件就



打开了相应的工程项目。

使用浏览条的视图部分和指令树的项目分支(如图 2-28 所示),可以查看项目的各个组件,并且在它们之间切换。用鼠标单击浏览条图标,或者双击指令树分支可以快速到达相应的项目组件。

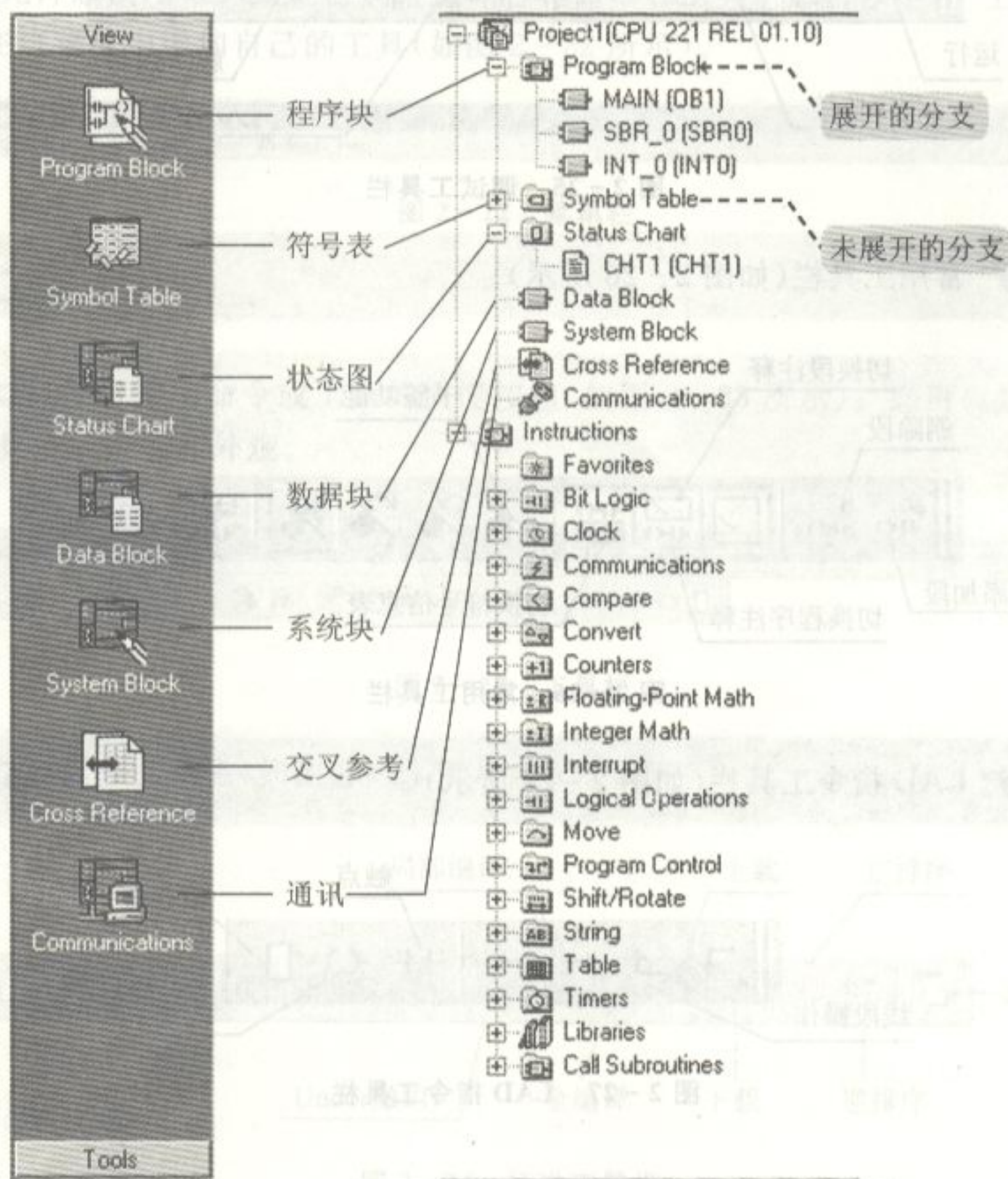


图 2-28 浏览条的视图部分和指令树的项目分支

按 Communications(通讯)图标可以设置编程计算机与 S7-200 CPU 之间的通讯参数。



2.2.3 定制 Step7 - Micro/WIN32

显示和隐藏各种窗口组件

从菜单条选择 View(查看),并选择一个对象,将其选择标记在打开和关闭之间切换。带选择标记的对象是当前在 Step7 - Micro/WIN32 环境中打开的对象(如图 2-29 所示)。

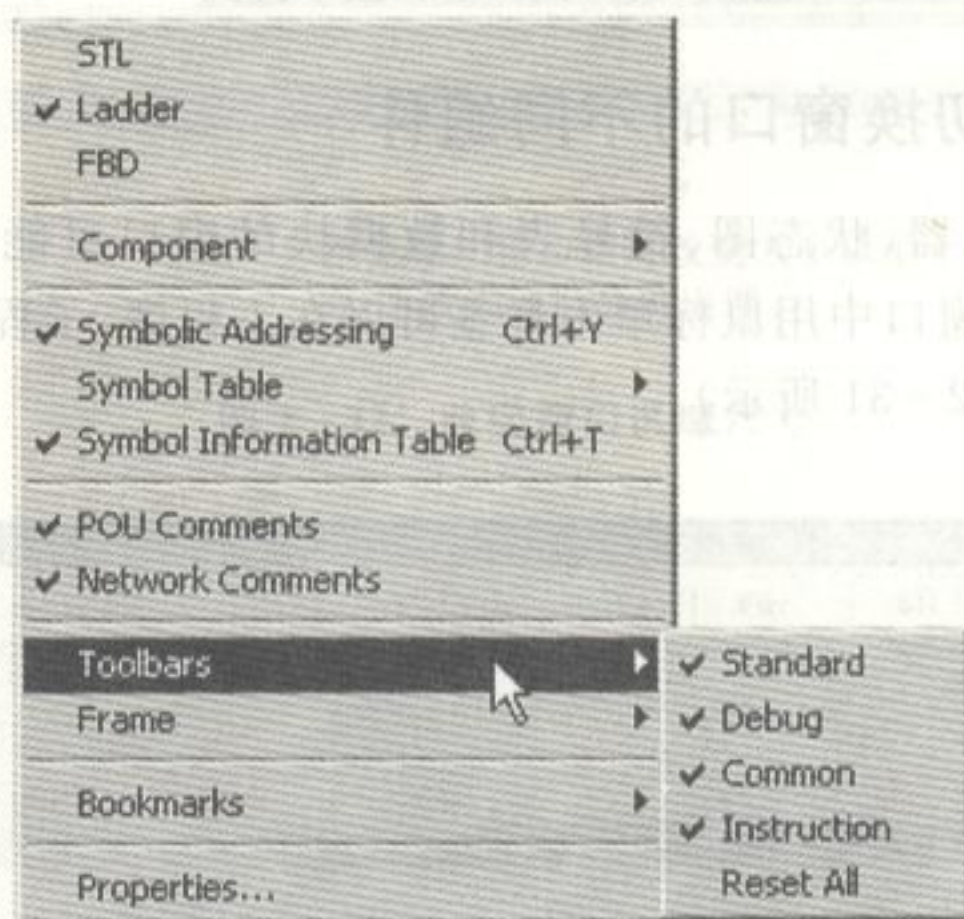


图 2-29 当前 Step7 - Micro/WIN32 环境中打开的对象

选择窗口显示方式

从菜单条选择 Windows (视窗) > Cascade (层叠); Horizontal (水平); Vertical (垂直) 可以改变窗口排列方式。也可在不同窗口间切换(如图 2-30 所示)。



当前窗口最大化之后,会自动隐藏其他窗口。

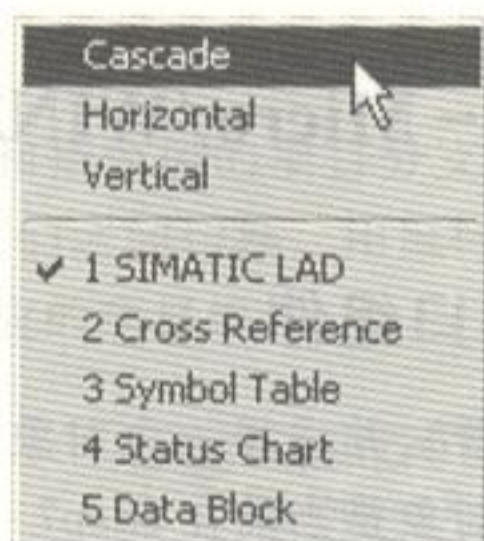


图 2-30 选择窗口显示方式

使用标签切换窗口的不同组件

诸如程序编辑器、状态图、符号表和数据块的窗口可能有多个标签。例如，在程序编辑器窗口中用鼠标单击标签可以在主程序、子程序和中断服务程序之间浏览(如图 2-31 所示)。



图 2-31 使用标签切换窗口的不同组件

✎ 改变窗口区域的尺寸(如图 2-32 所示)。

选择中文环境

Step7 - Micro/WIN32 V3.2 从 SP1 起,支持完全汉化的工作环境。

在菜单 Tools(工具) > Options(选项)中,选择 General(常规)选项卡,可以设置语言环境(如图 2-33 所示)。改变设置后,退出 Step7 - Micro/WIN32,再次启动后生效。

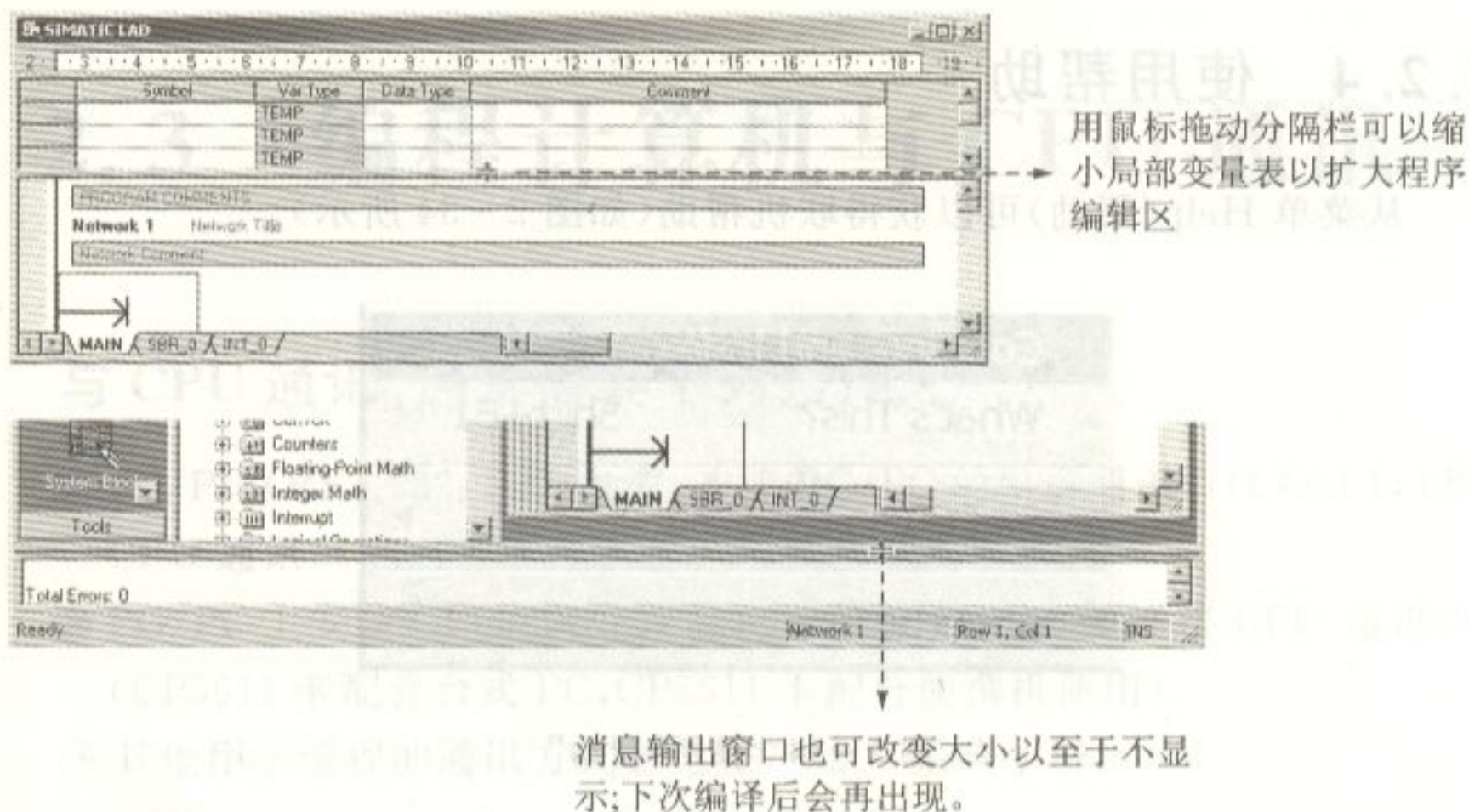


图 2-32 改变窗口区域尺寸

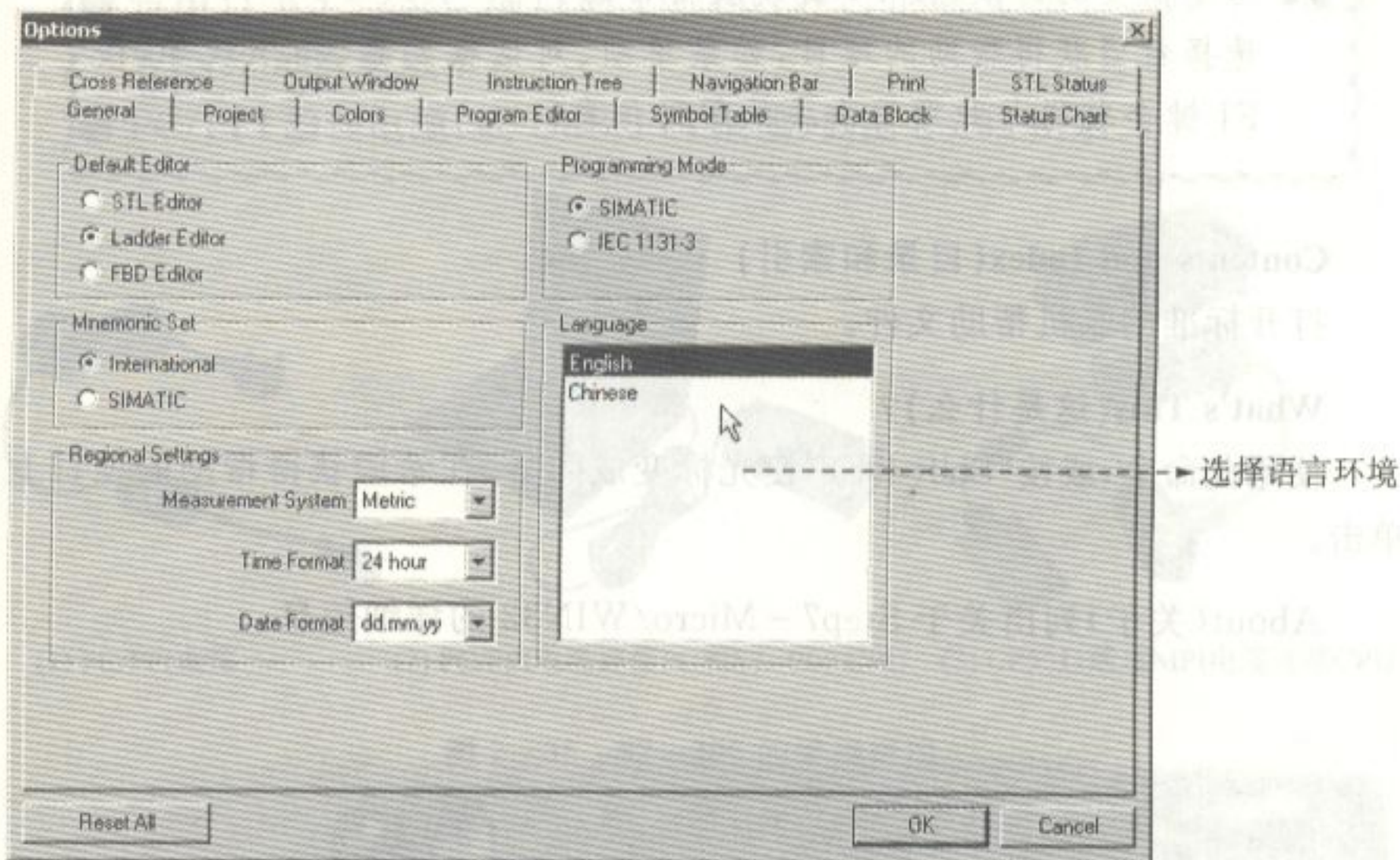


图 2-33 选择语言环境



2.2.4 使用帮助

从菜单 Help(帮助)可以获得联机帮助(如图 2-34 所示)。

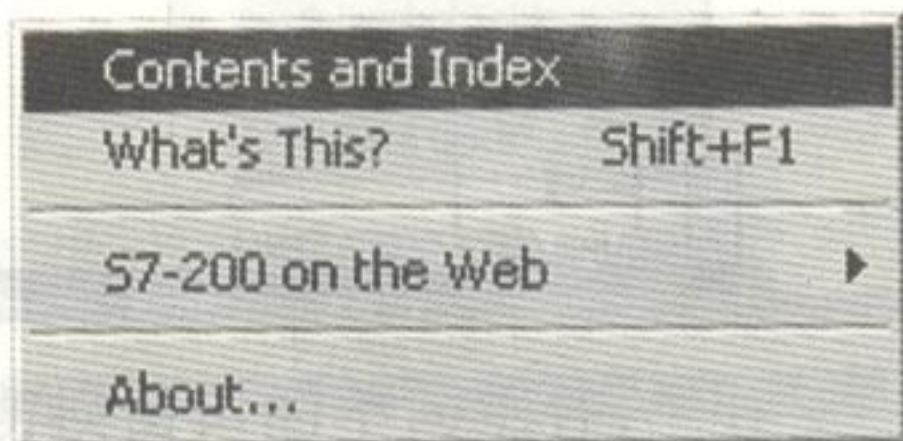


图 2-34 使用“帮助”



当您所需要的时候,按 F1 键就获得帮助!

选择希望获得帮助的项目,如菜单项、对话框元素、指令块等,按 F1 键会获得与此项目有关的帮助信息。

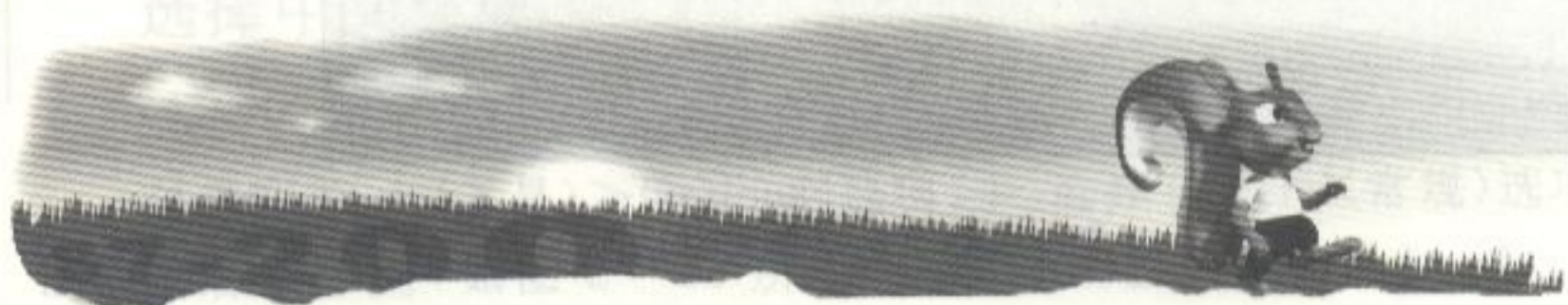
Contents and Index(目录和索引)

打开标准的联机帮助文件。

What's This(这是什么)?

使用此命令,或按“Shift+F1”使光标变成问号,在希望获得帮助的项目上单击。

About(关于)列出关于 Step7 - Micro/WIN32 的详细信息。





2.3 编程计算机与 CPU 通讯

与 CPU 通讯,通常需要下列条件之一

- PC/PPI(RS-232/PPI)电缆,连接 PG/PC 的串行通讯口(COM 口)和 CPU 通讯口;
- PG/PC 上安装 CP(通讯处理器)卡,通过 MPI 电缆连接 CPU 通讯口(CP5611 卡配合台式 PC,CP5511 卡配合便携机使用)。
- ⊕ 其他用于编程的通讯方式参见《S7-200 系统手册》。

最简单的编程通讯需要

- 带串行 RS-232C 端口的 PG/PC,并已正确安装了 Step7-Micro/WIN32 的有效版本;
- PC/PPI 编程电缆(或 USB/PPI 电缆)如图 2-35 所示。



图 2-35 S7-200 的编程电缆

2.3.1 设置通讯

设置 PC/PPI 电缆小盒中的 DIP 开关(如图 2-36 所示)。设定 PC/PPI



电缆的通讯波特率为 9.6 K 波特。

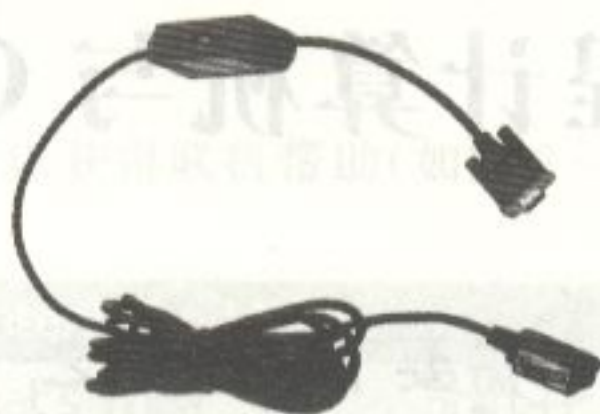


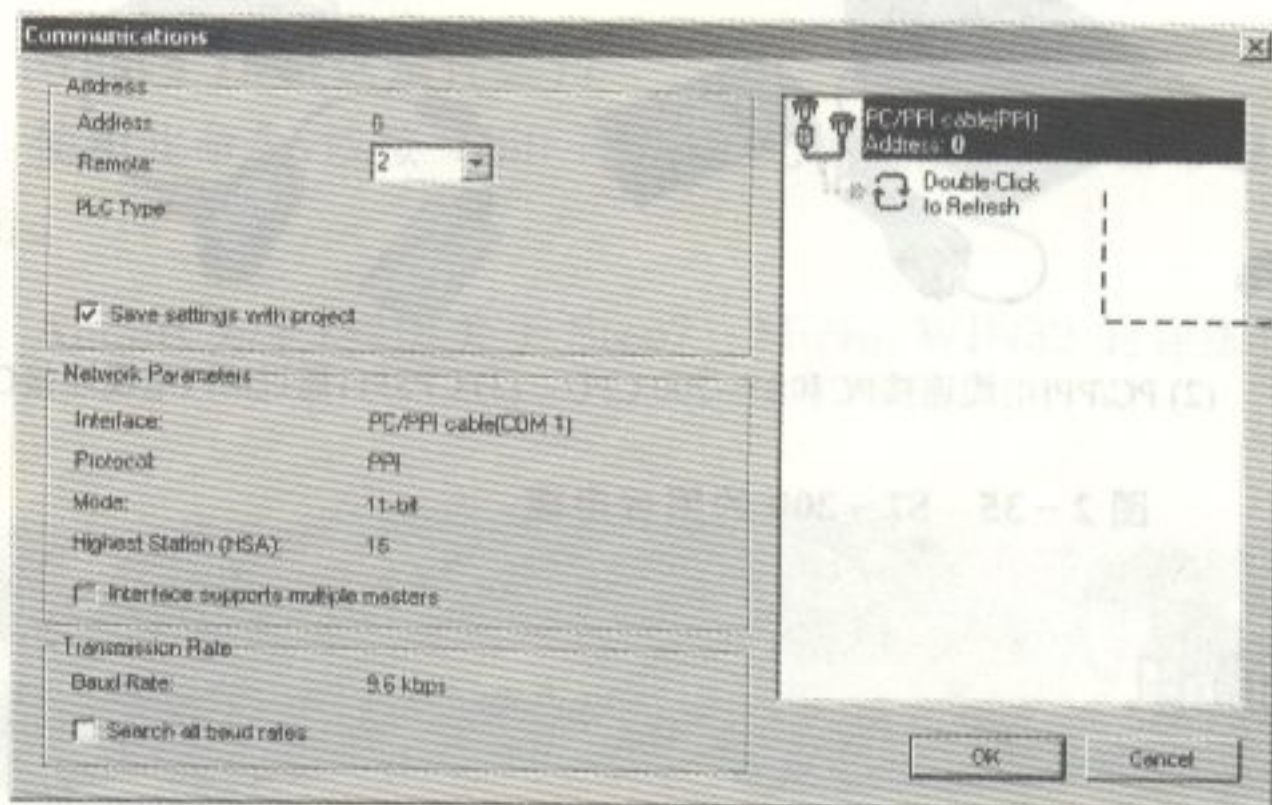
图 2-36 PC/PPI 电缆上的小盒中有 DIP 开关



可以根据需要选择不同的通讯波特率。9.6 K 波特是 S7-200 CPU 默认的通讯速率。使用其他波特率需要在系统块内设置，并下载到 CPU 中才能生效。

用 PC/PPI 电缆连接 PG/PC 和 CPU，将 CPU 前盖内的模式选择开关设置为 STOP，给 CPU 上电。

用鼠标单击浏览条上的“通讯 (Communications)”图标出现通讯设置窗口 (如图 2-37 所示)。



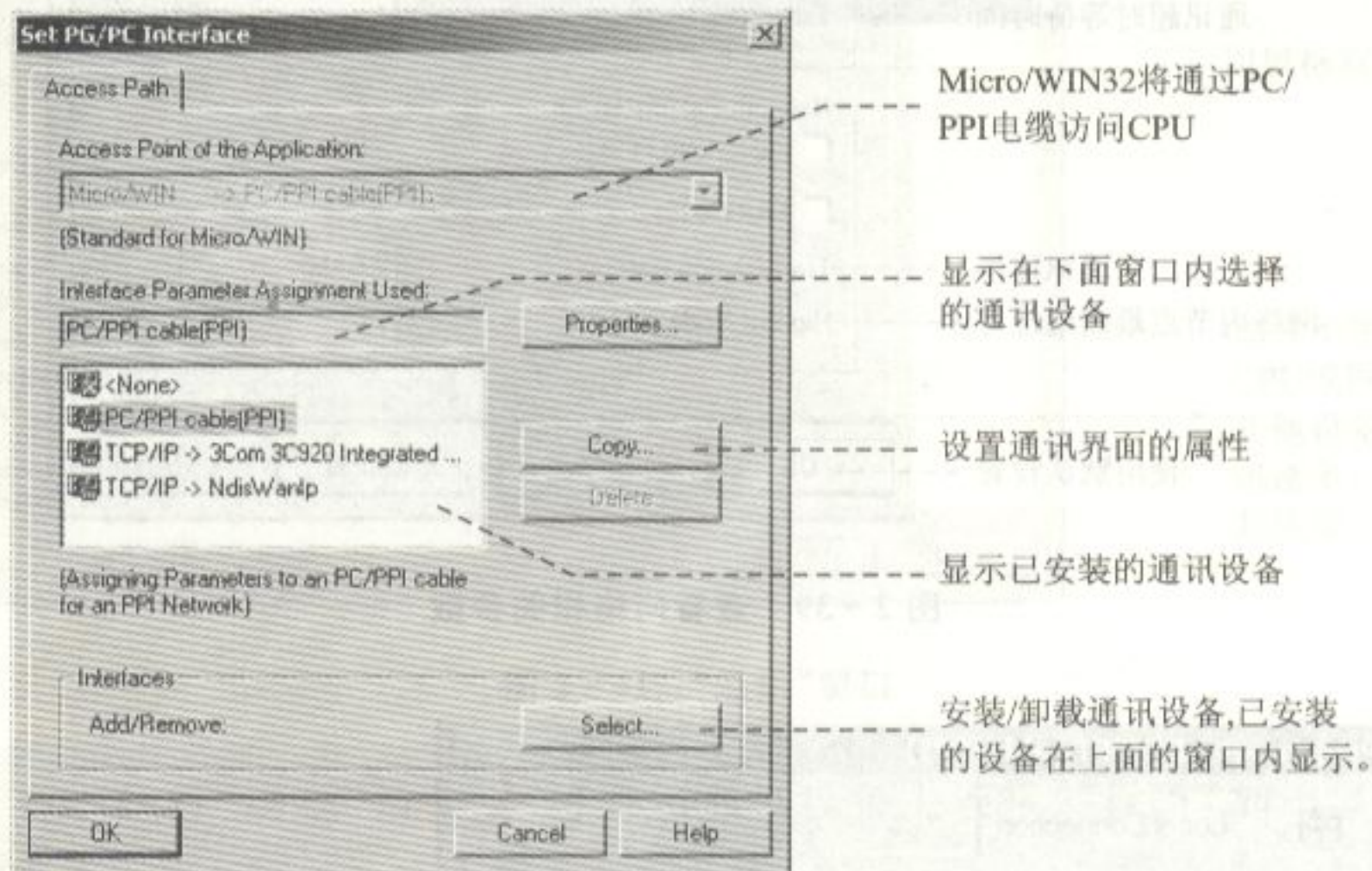
编程计算机与
CPU 连接的设备

图 2-37



窗口右侧显示编程计算机将通过 PC/PPI 电缆尝试与 CPU 通讯,并且本地编程计算机的网络通讯地址是 0。

用鼠标双击 PC/PPI 电缆的图标,出现如图 2-38 所示的窗口。



单击 PC/PPI 电缆旁边的 Properties(属性)按钮,查看、设置 PC/PPI 电缆连接参数。

图 2-38 设置 PG/PC 的界面

在“PPI”标签卡中查看网络相关参数(如图 2-39 所示)。

使此处的通讯速率与 PC/PPI 电缆 DIP 开关的设置一致。CPU 出厂时通讯速率的默认设置为 9.6 K。

在 Local Connection(本地连接)选项卡中,选择编程计算机 COM 口(如图 2-40)所示。

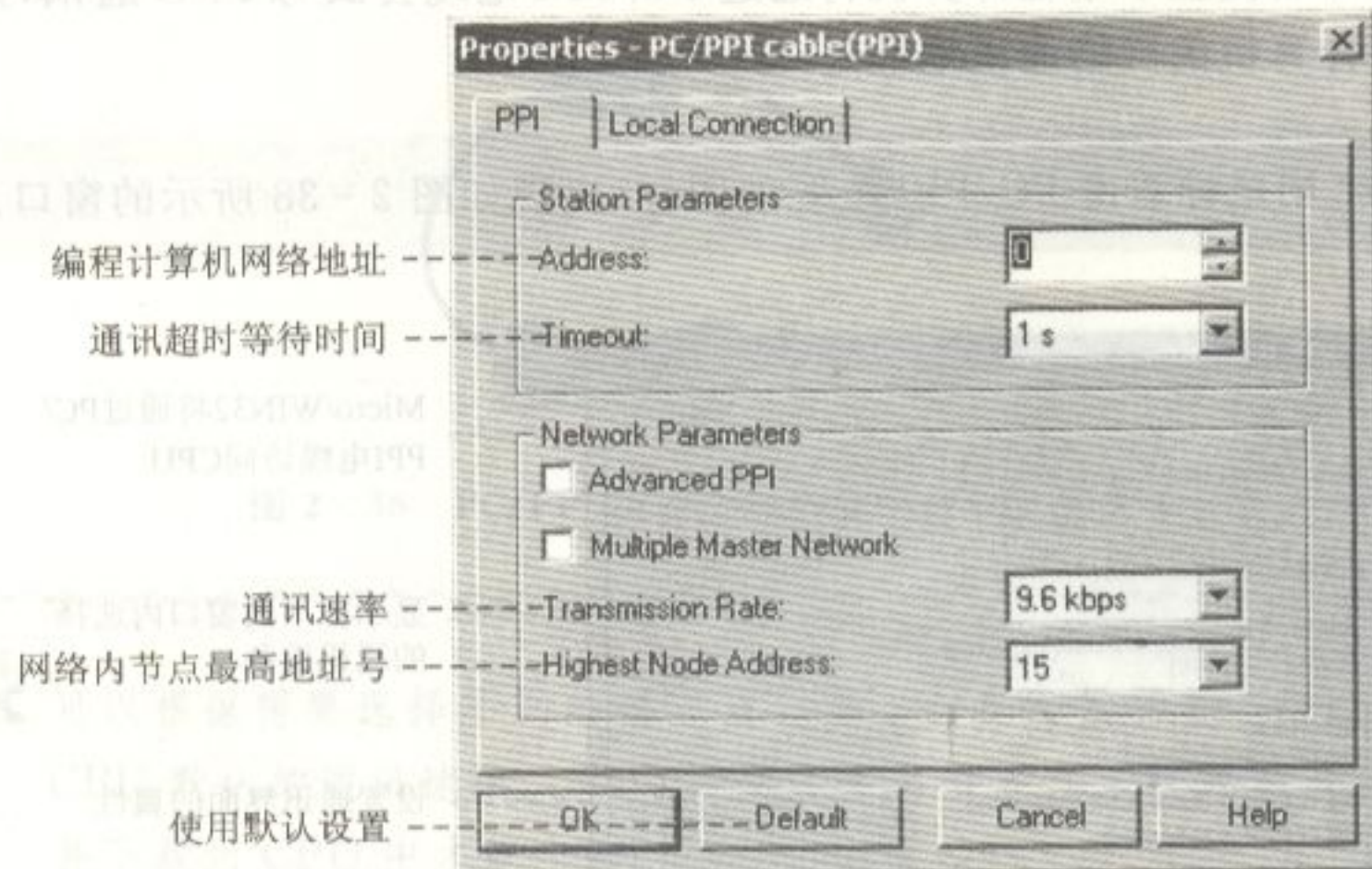


图 2-39 查看网络相关参数

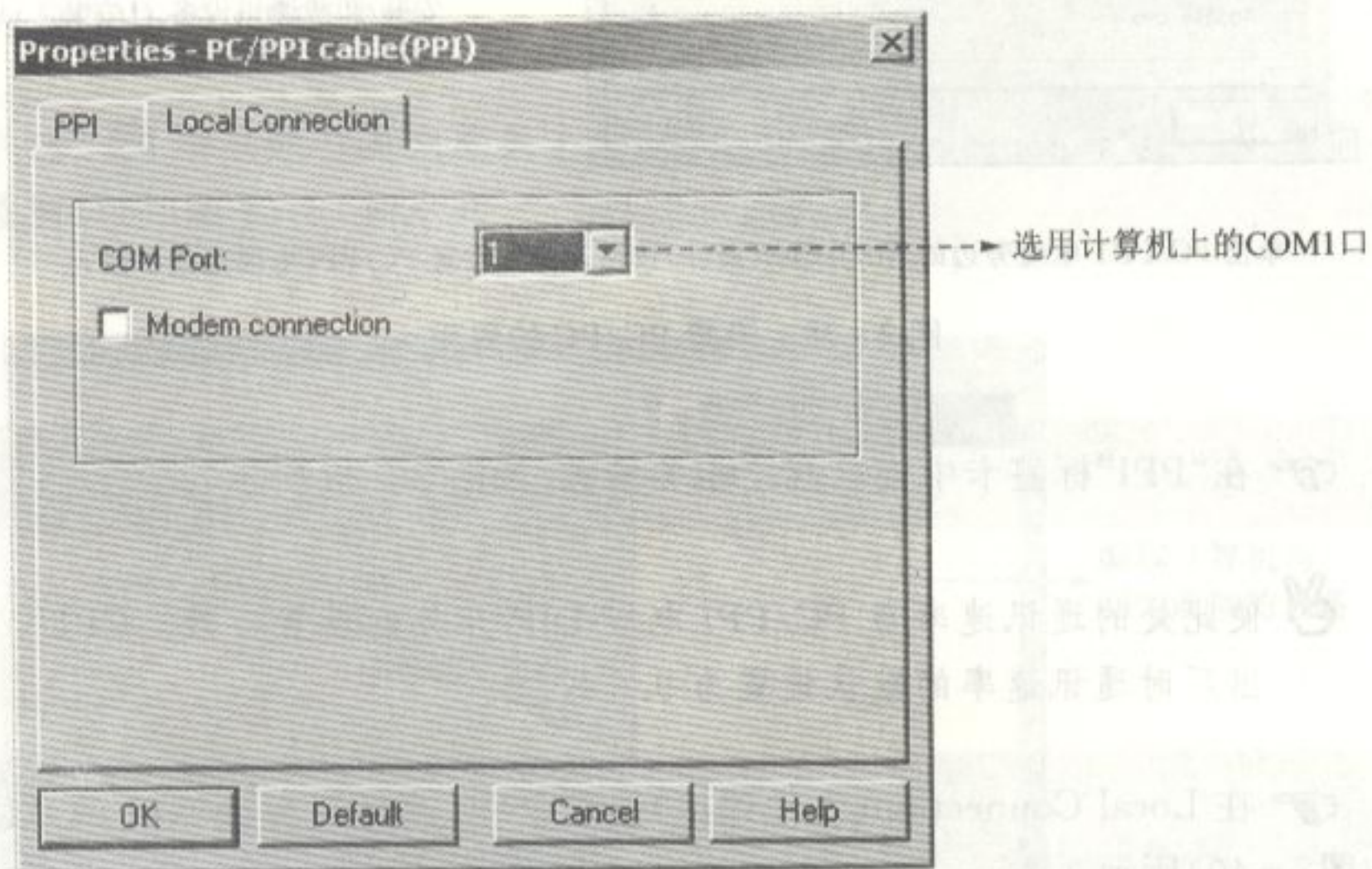


图 2-40 选择编程计算机 COM 口



☞ 按“OK”按钮回到“通讯”窗口，鼠标双击“Refresh(刷新)”图标(如图2-41所示)。

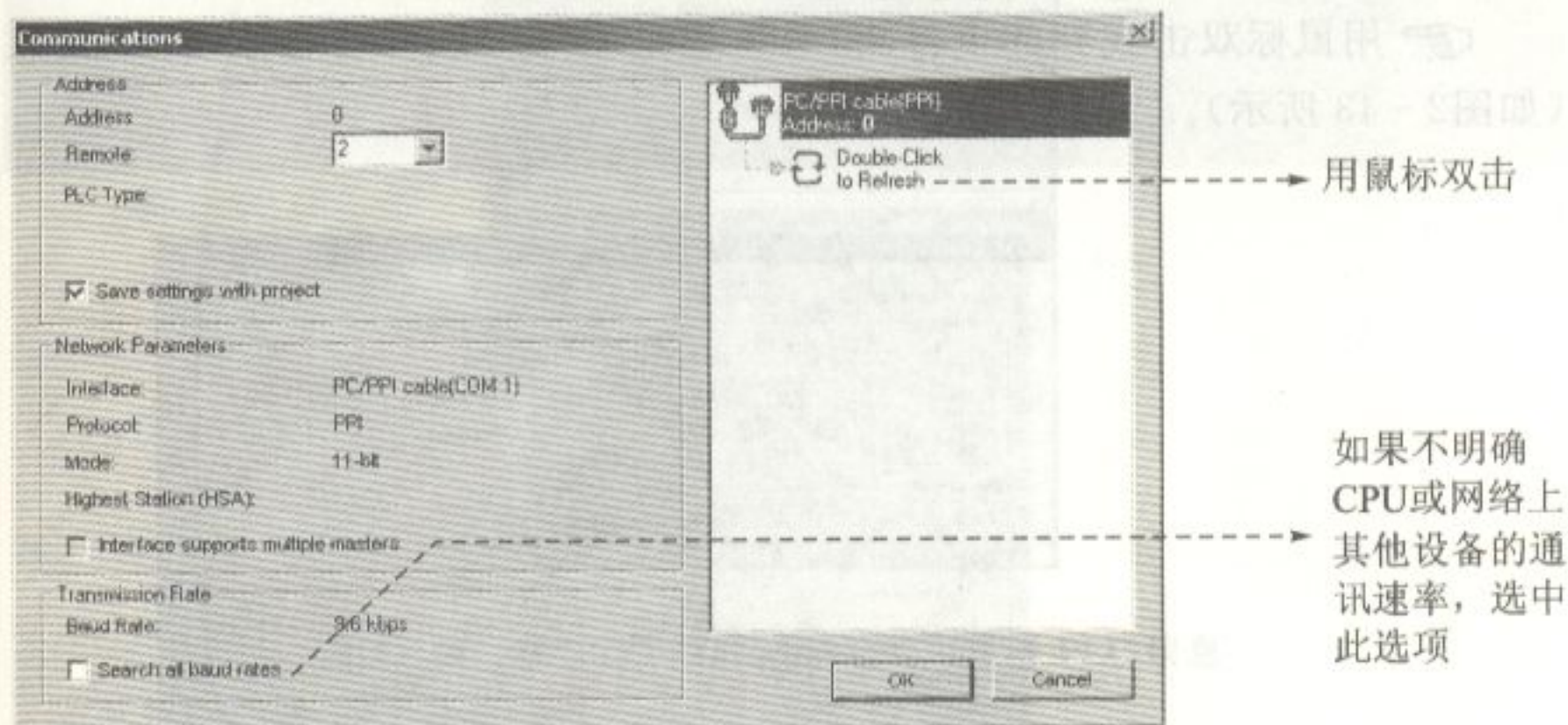


图 2-41 “通讯”窗口

☞ 执行“刷新”指令后，将显示通讯设备上连接的设备(如图2-42所示)。

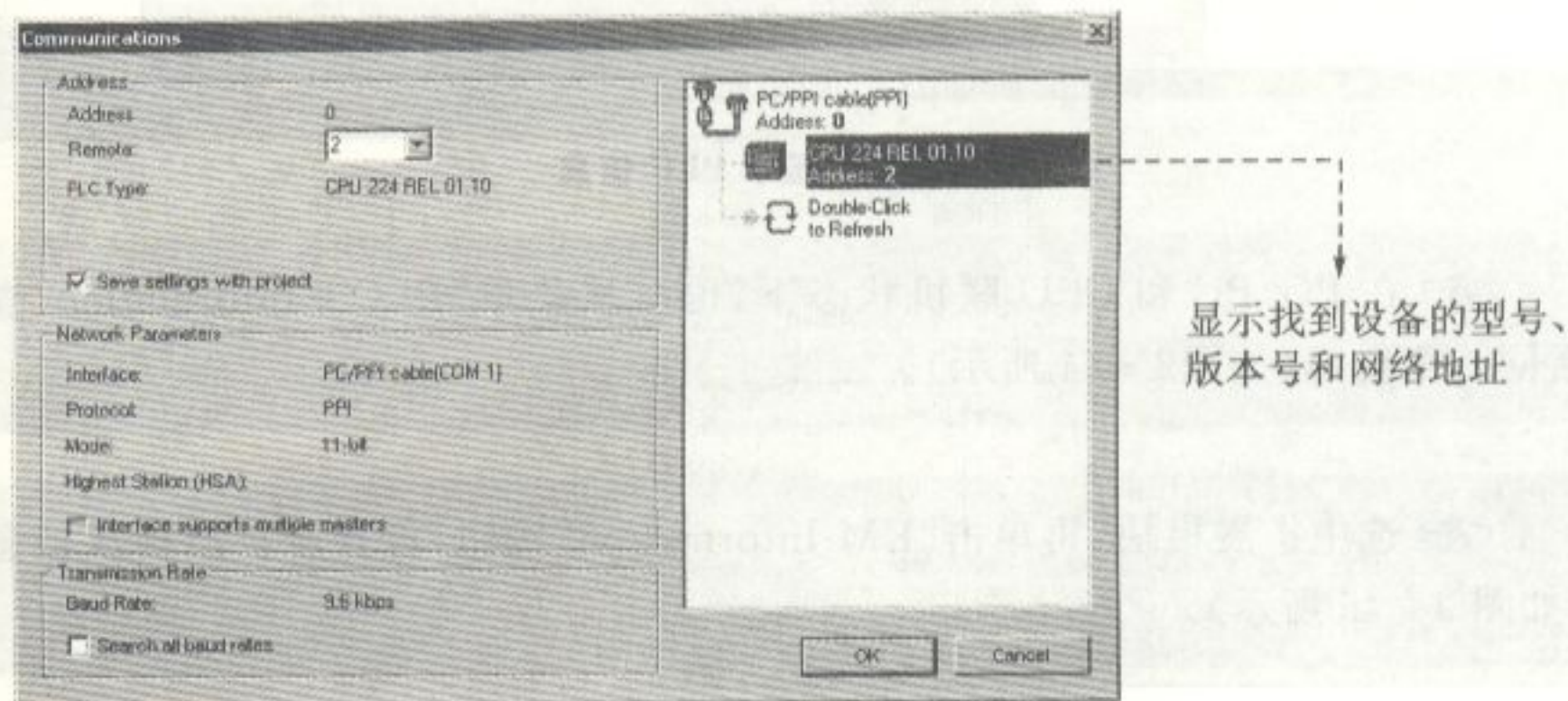


图 2-42 显示通讯设备上连接的设备



2.3.2 PLC 信息

☞ 用鼠标双击找到的设备图标(这里是 CPU 224),将显示 CPU 信息(如图2-43所示)。

CPU本机I/O信息和
扩展模块信息

Operating Mode		STOP				
Versions						
PLC:	CPU 224 REL 01.10		Scan Rates (ms)			
Firmware:	01.10		Last: 0			
ASIC:	01.00		Minimum: 0			
			Maximum: 0			
Errors						
Fatal	0	No fatal errors present				
Non-Fatal	0	No non-fatal errors present				
Last Fatal	0	No fatal errors present				
Total Fatal	0					
I/O Errors						
Number of Errors:	0					
Errors Reported:	No I/O errors present					
Module	Type	In	Start	Out	Status	
0	Discrete	16	I0.0	16	Q0.0	No error
1	EM277 Profibus DP					No error
2						Not present
3						Not present
4						Not present
5						Not present
6						Not present

Buttons: EM Information, Reset Scan Rates, Close

图 2-43 显示 PLC 信息

☞ 在 PG/PC 和 CPU 联机状态下,也可从菜单“PLC>Information”查看同一个窗口(如图2-44所示)。

☞ 选中扩展模块,再单击“EM Information”按钮,可查看扩展模块信息(如图2-45所示)。

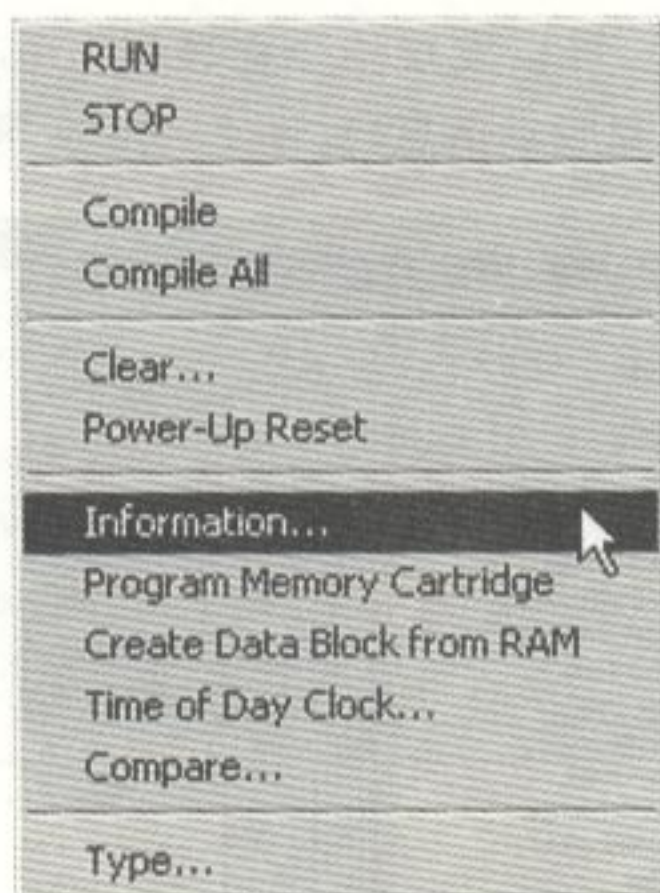


图 2-44 在“PLC>Information”中查看 PLC 信息

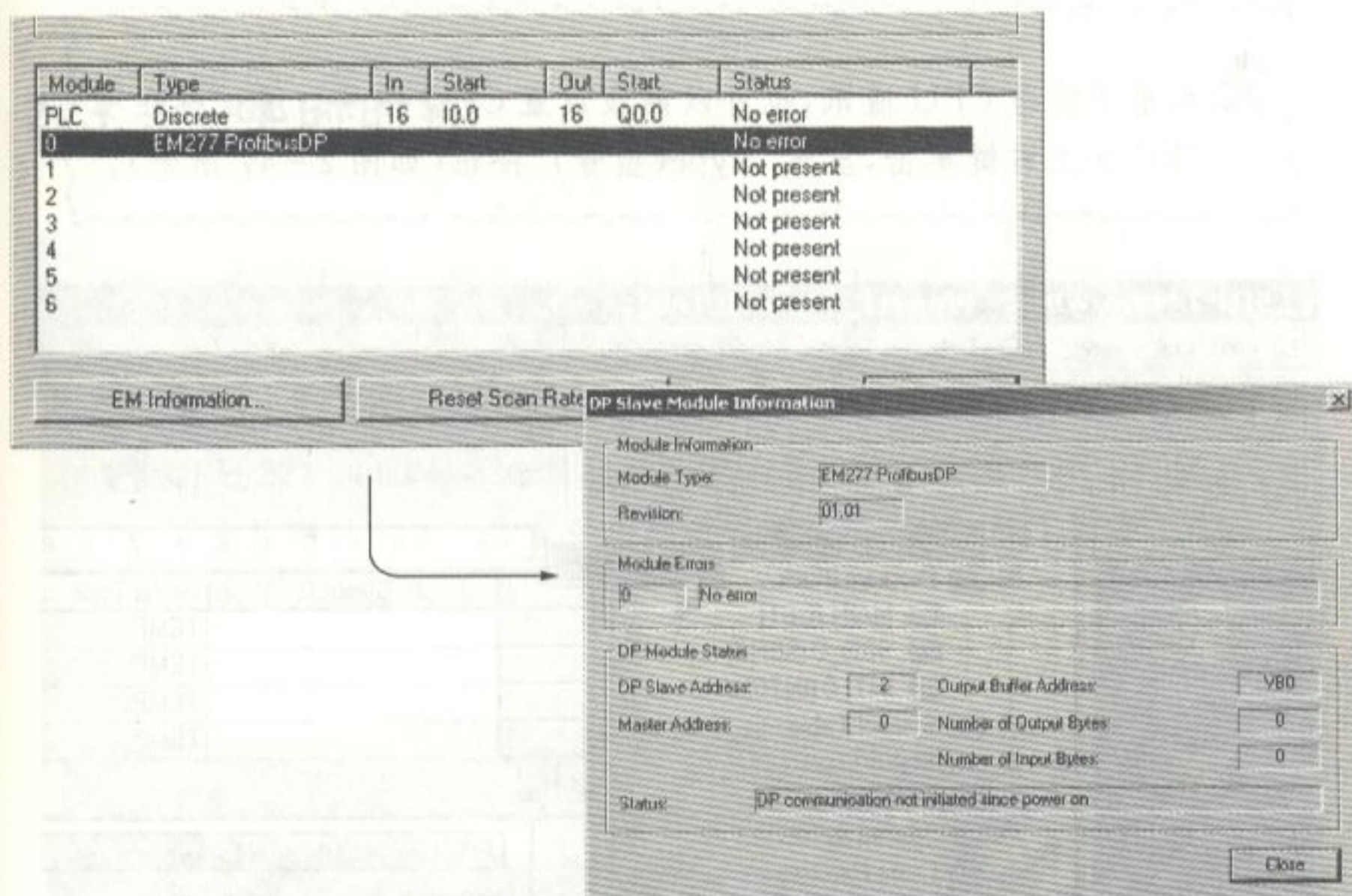
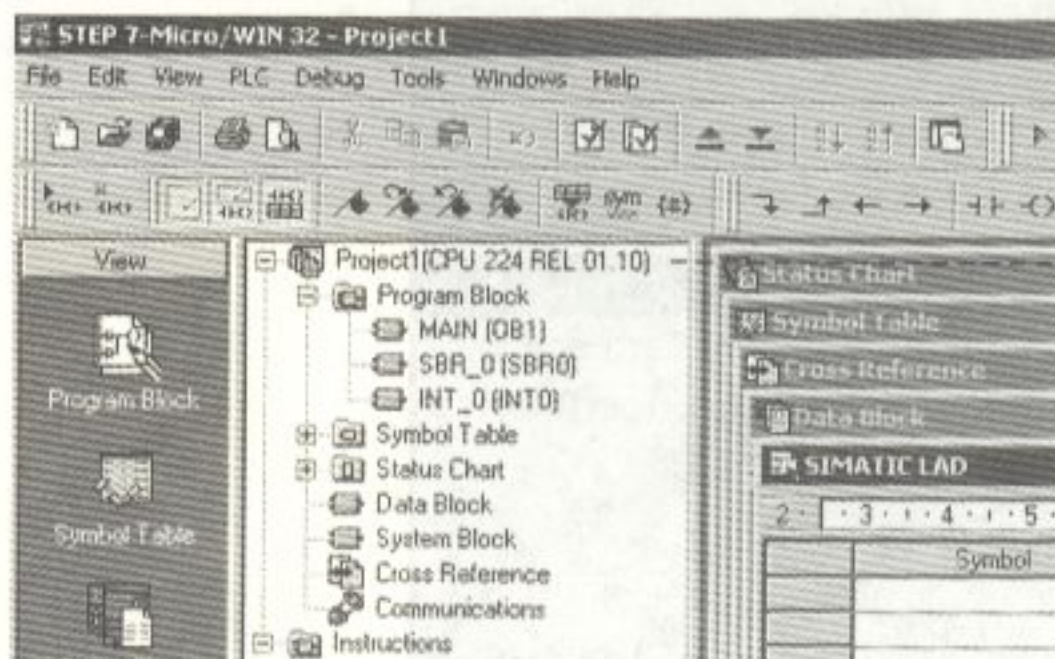


图 2-45 查看扩展模块信息



关闭通讯窗口后,可以发现指令树项目(Project)条目显示实际连接并通讯成功的 CPU 型号和版本信息(如图 2-46 所示)。



本项目中实际使用的CPU

图 2-46 指令树项目条目显示的信息



如果不能与 CPU 通讯,也可以离线设置 CPU 型号。用鼠标在项目分支上右键单击,显示“Type(型号)”按钮(如图 2-47 所示)。

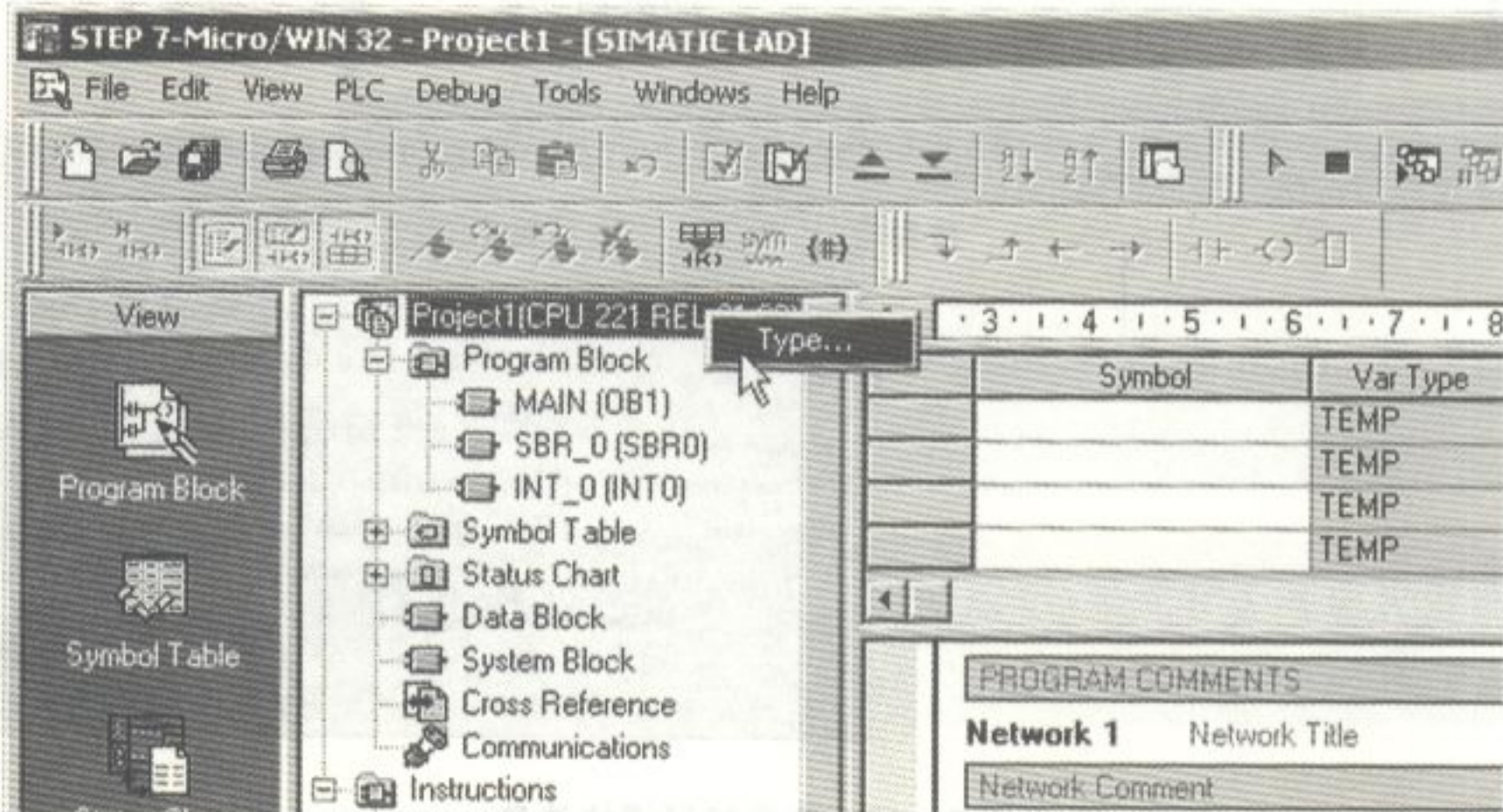
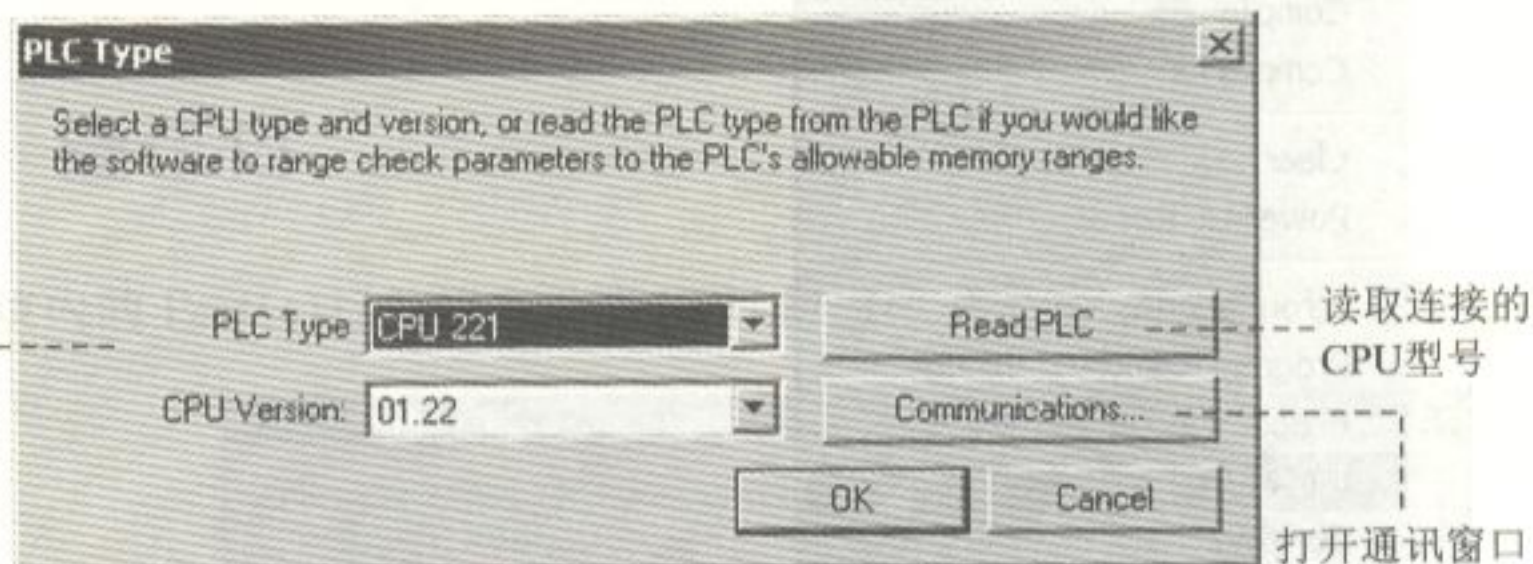


图 2-47 离线设置 CPU 型号



☞ 单击“Type(型号)”按钮(如图 2-48 所示)。



选择CPU型号和版本

图 2-48 选择 CPU 型号

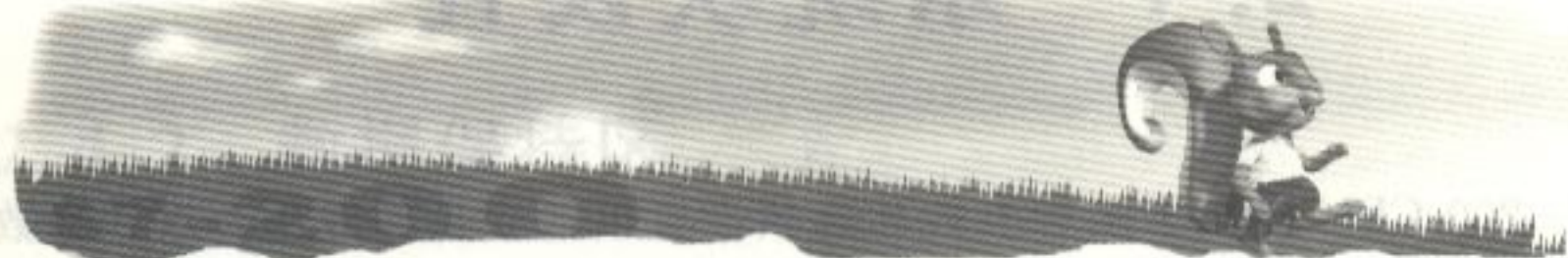
也可以使用菜单命令“PLC>Type”打开上面的对话框。

2.3.3 实时时钟

ONLINE

☞ 在连通状态下,使用菜单命令“PLC>Time of Day Clock...”显示、设置、启动 CPU 实时时钟(如图 2-49 所示)。

☞ CPU 224 和 CPU 226 具有内置实时时钟,CPU 221 和 CPU 222 需要外接时钟电池卡才能使用实时时钟。全新的 CPU 需要设置,时钟才能开始正常走动。



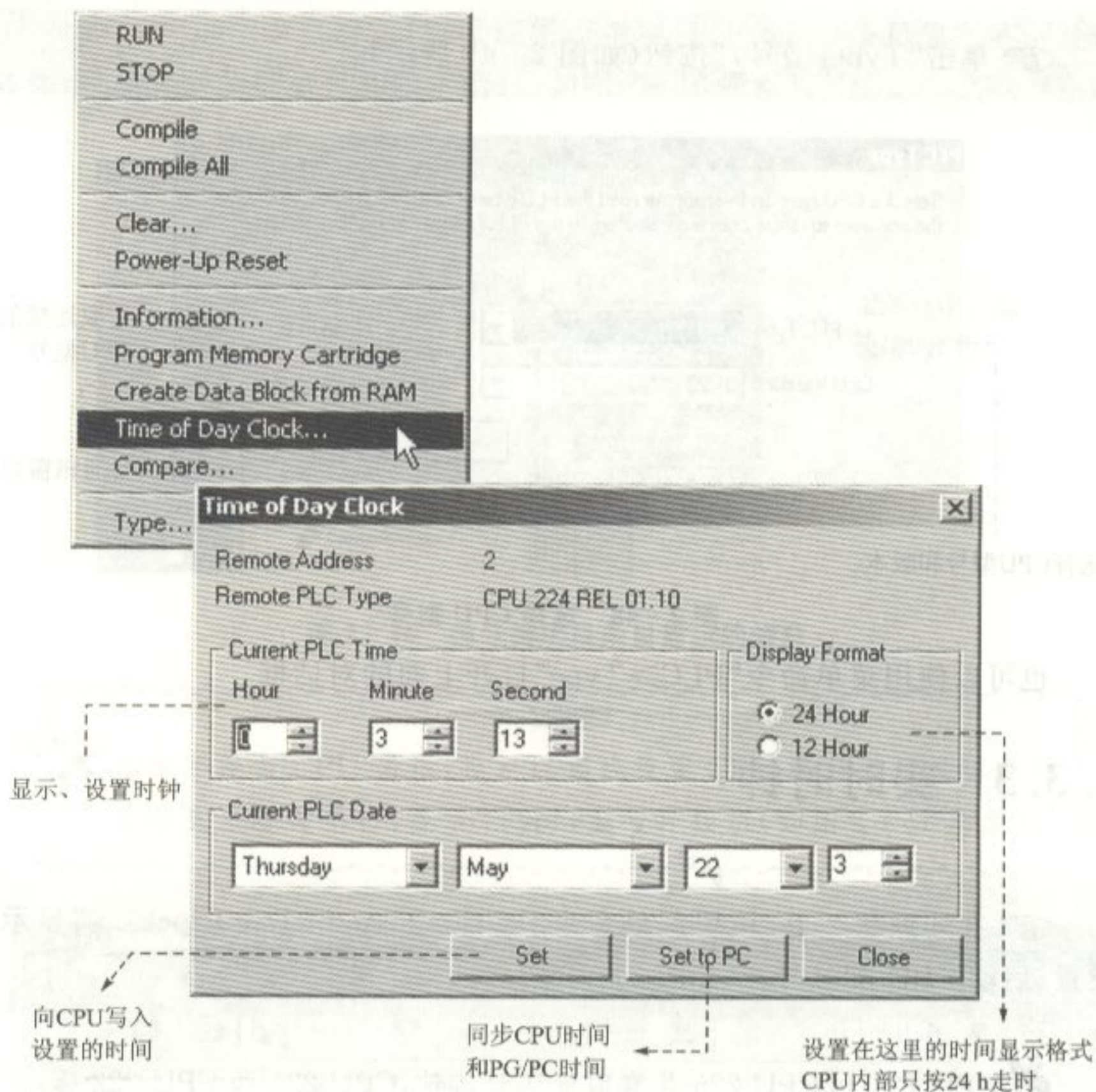


图 2-49 设置 CPU 实时时钟

2.4 系统块设置

S7-200 CPU 提供了多种参数和选项设置以适应具体应用。这些参数和选项在 System Block(系统块)内设置。系统块需经编译、下载到 CPU 内才起作用。



使用 View(查看)浏览条内的按钮,或者使用菜单命令 View>Component>System Block 打开系统块设置窗口。

2.4.1 通讯口

系统块内的 Port(s)标签用来设置 CPU 的通讯口属性(如图 2-50 所示)。

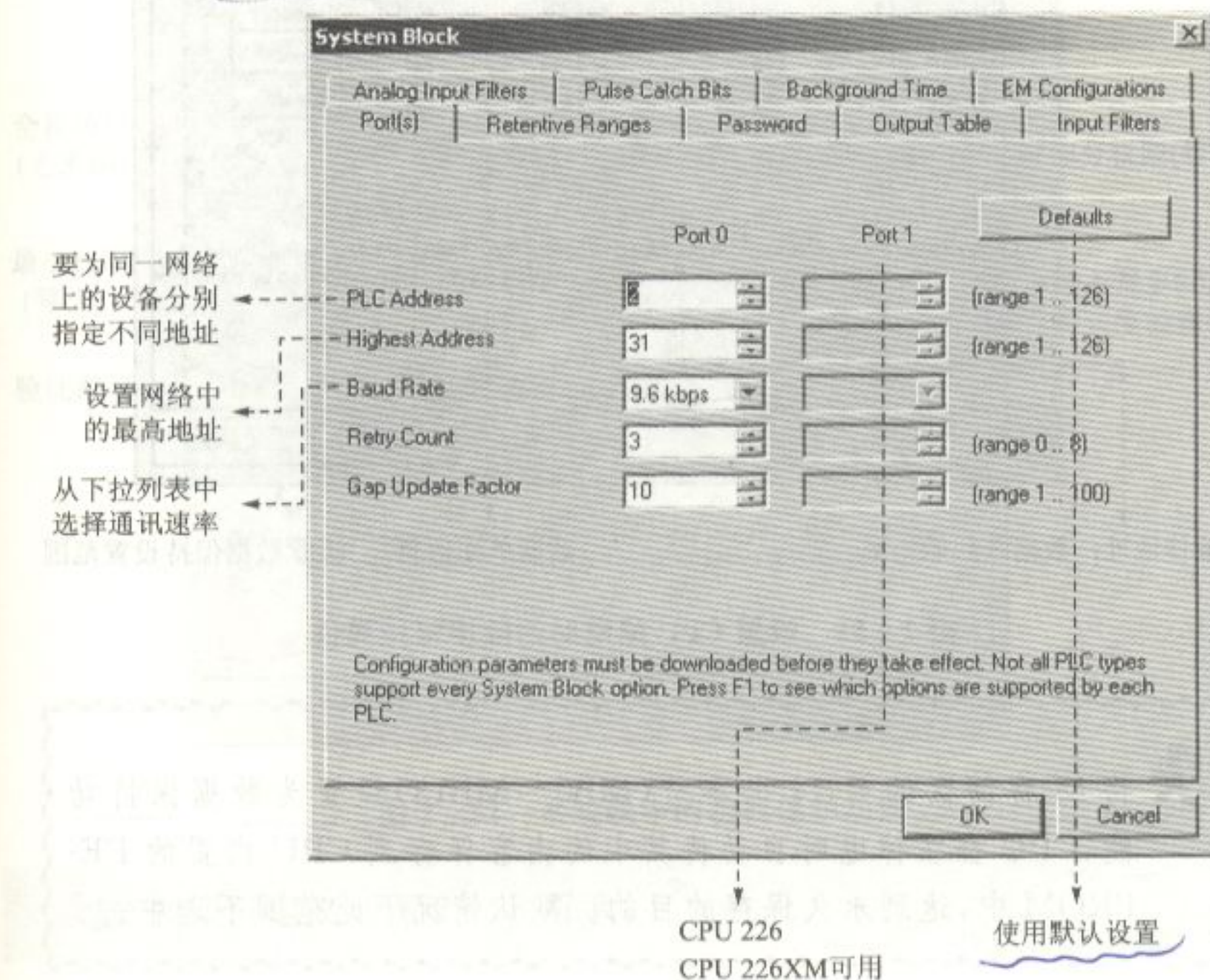


图 2-50 系统块内设置 CPU 通讯口属性

2.4.2 数据保持区

Retentive Ranges(数据保持区)标签用来设置 CPU 掉电时如何保存数据(如图 2-51 所示)。

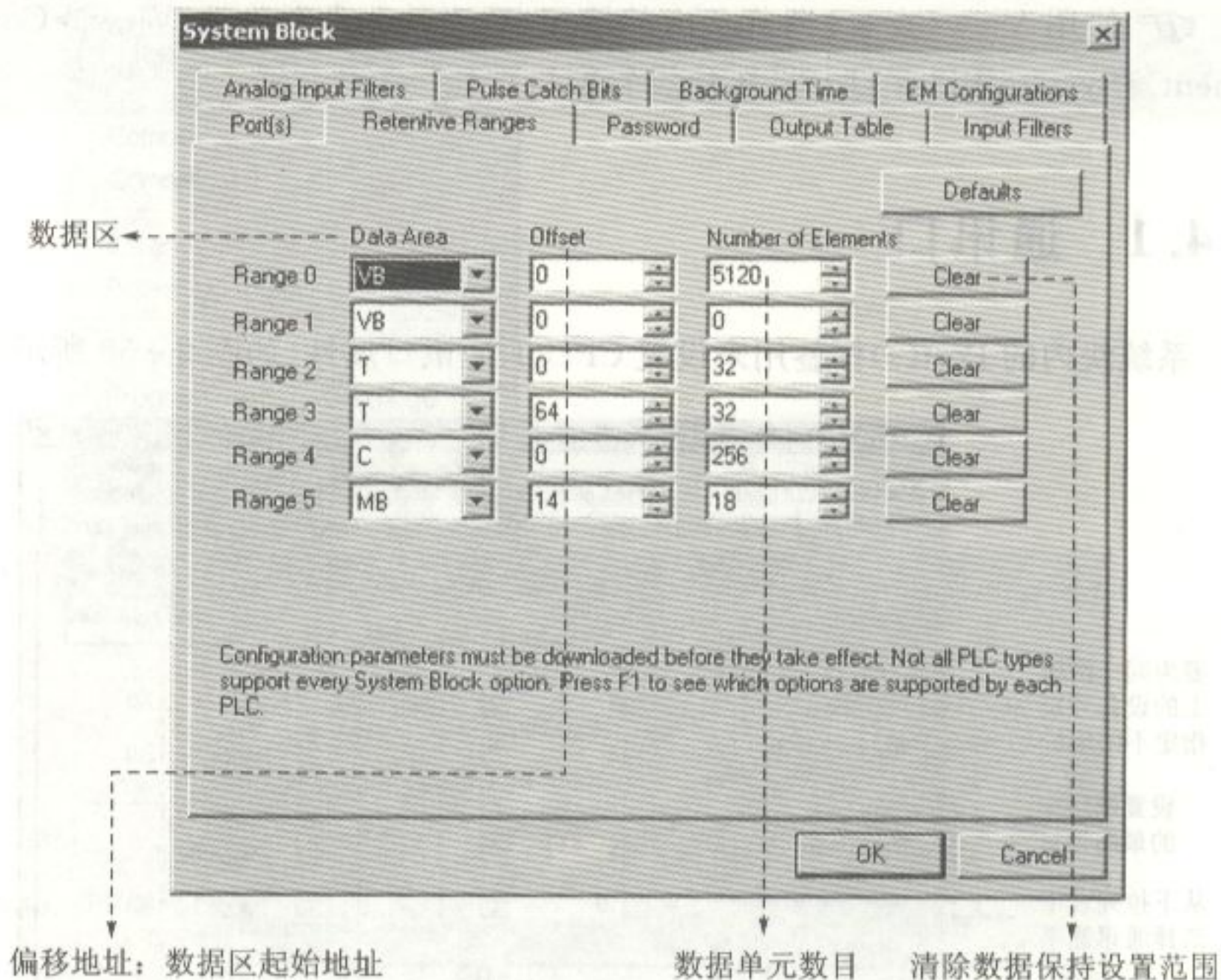


图 2-51 设置 CPU 掉电时的数据保持属性



将 M 存储区的前 14 个字节 (MB0 ~ MB13) 设置为数据保持功能, CPU 会在掉电时自动将其中的内容保存到 CPU 内置的 EEPROM 中, 达到永久保存的目的。默认情况下此范围不选中。



数据保持区设置可用来检验 CPU 的内置 EEPROM 是否正确保存了数据。清除 V 存储区的数据保持设置, 如果关断 CPU 电源后再送电, 观察到 V 存储区相应的单元内还保存有正确的数据, 说明数据已成功写入 CPU 的 EEPROM。



2.4.3 S7-200 CPU 密码保护

可以设置密码以限制访问 S7-200 CPU 的内容或者限制使用某些功能。

System Block(系统块)的 Password(密码)标签用以设置 CPU 的密码保护功能(如图 2-52 所示)。

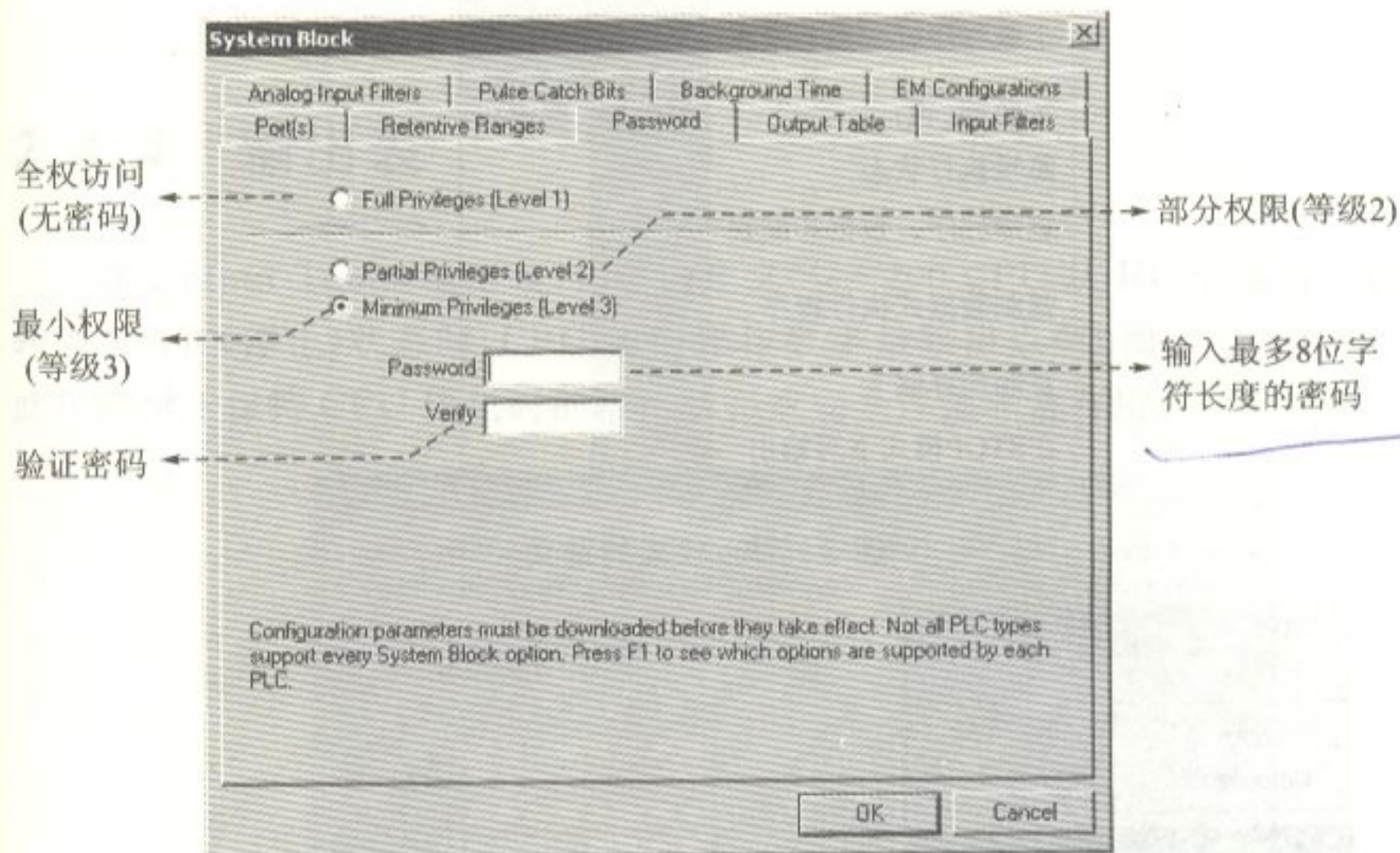


图 2-52 系统块中的密码保护功能

✎ 权限的定义如图 2-53 所示。



清除密码: 如果忘记密码而不能访问 CPU, 建立与 S7-200 CPU 的通讯后, 执行菜单命令 PLC > Clear... (清除) 可以同时清除密码和程序。



在弹出的对话框中,用鼠标按 OK 按钮(如图 2-54 所示)。

CPU功能	1级	2级	3级
读写用户数据	不限制	不限制	不限制
启动、停止重启CPU			
读写时钟			
上载程序、数据和配置	不限制	不限制	要密码
下载到CPU	不限制	要密码	
监视执行状态			
删除程序、数据和配置			
强制数据式执行程序			
复制到存储卡			
在STOP模式下写输出			

图 2-53 权限的定义

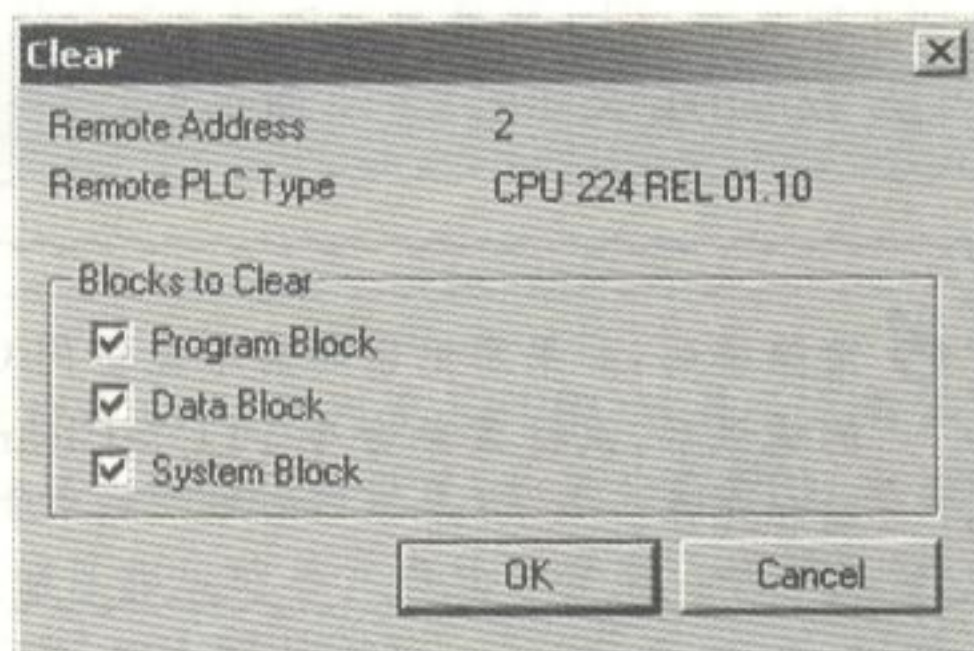
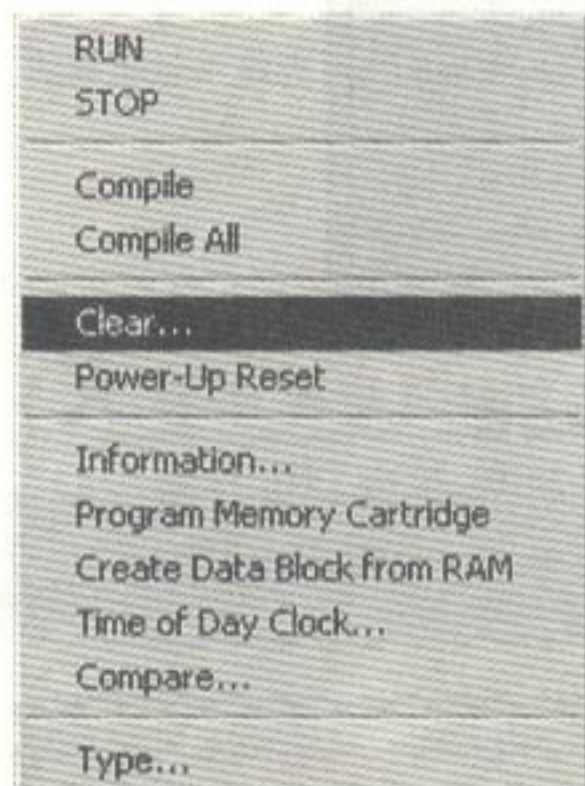


图 2-54 清除密码操作



☞ 输入 CPU 清除密码 CLEARPLC(大小写不限),如图 2-55 所示。

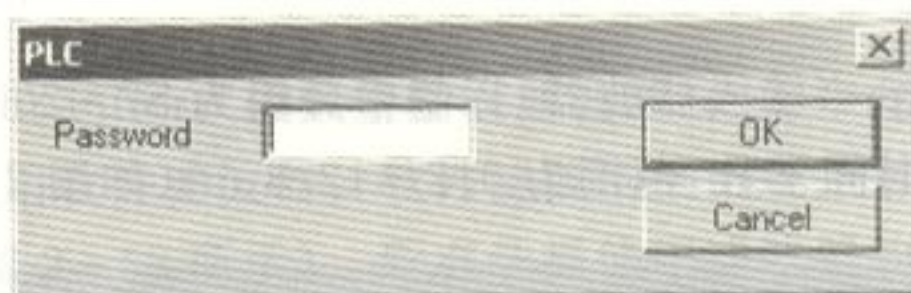
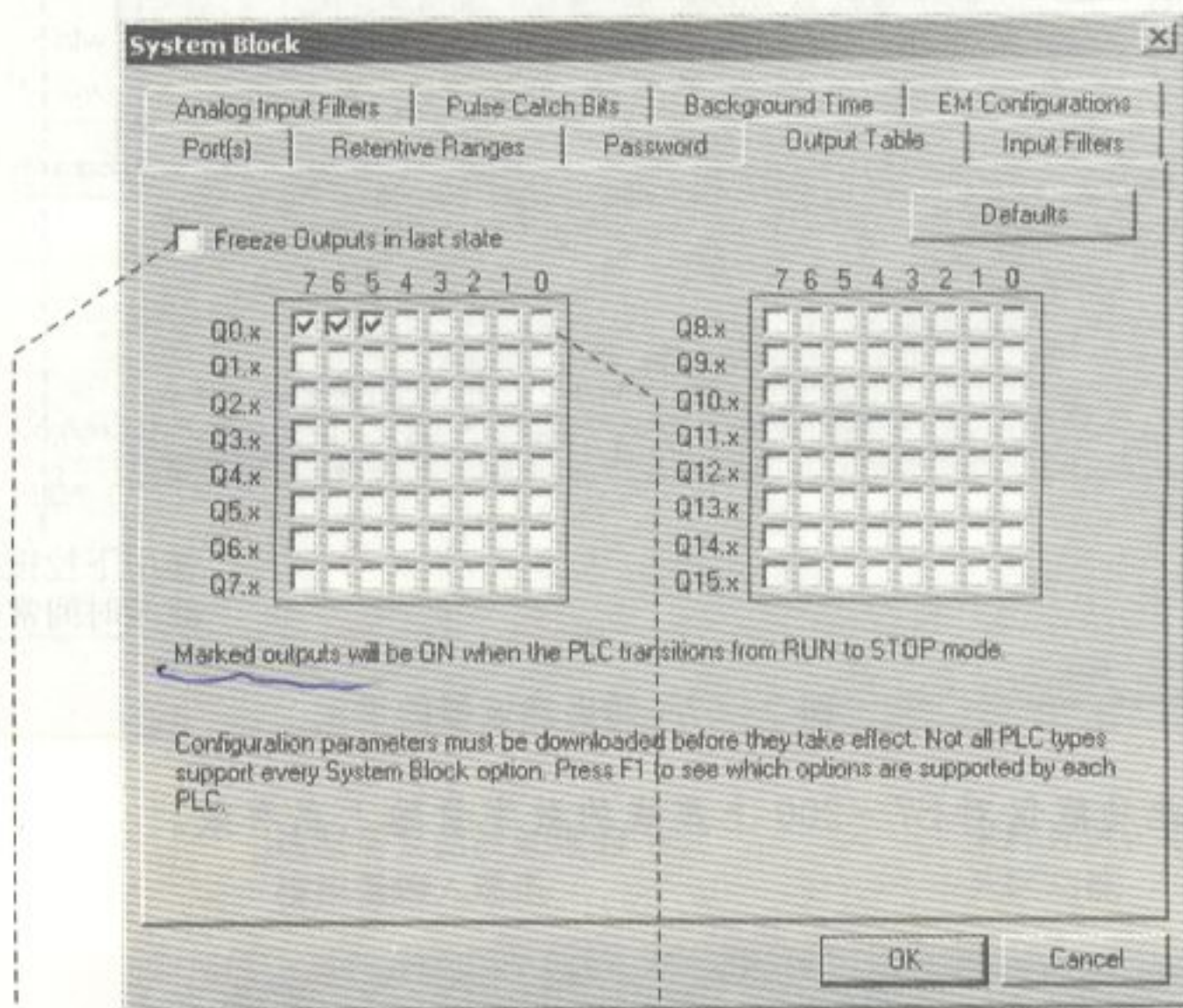


图 2-55 输入 CPU 清除密码

2.4.4 输出表

在 Output Table(输出表)标签内定义 S7-200 CPU 从 RUN(运行)状态转到 STOP(停止)状态时,CPU 如何操作数字量输出点(如图 2-56 所示)。此功能对于保持 CPU 停机时的输出状态或安全连锁非常有用。



选取此处将保持所有数字量输出的最后状态,同时,下面的表格不起作用

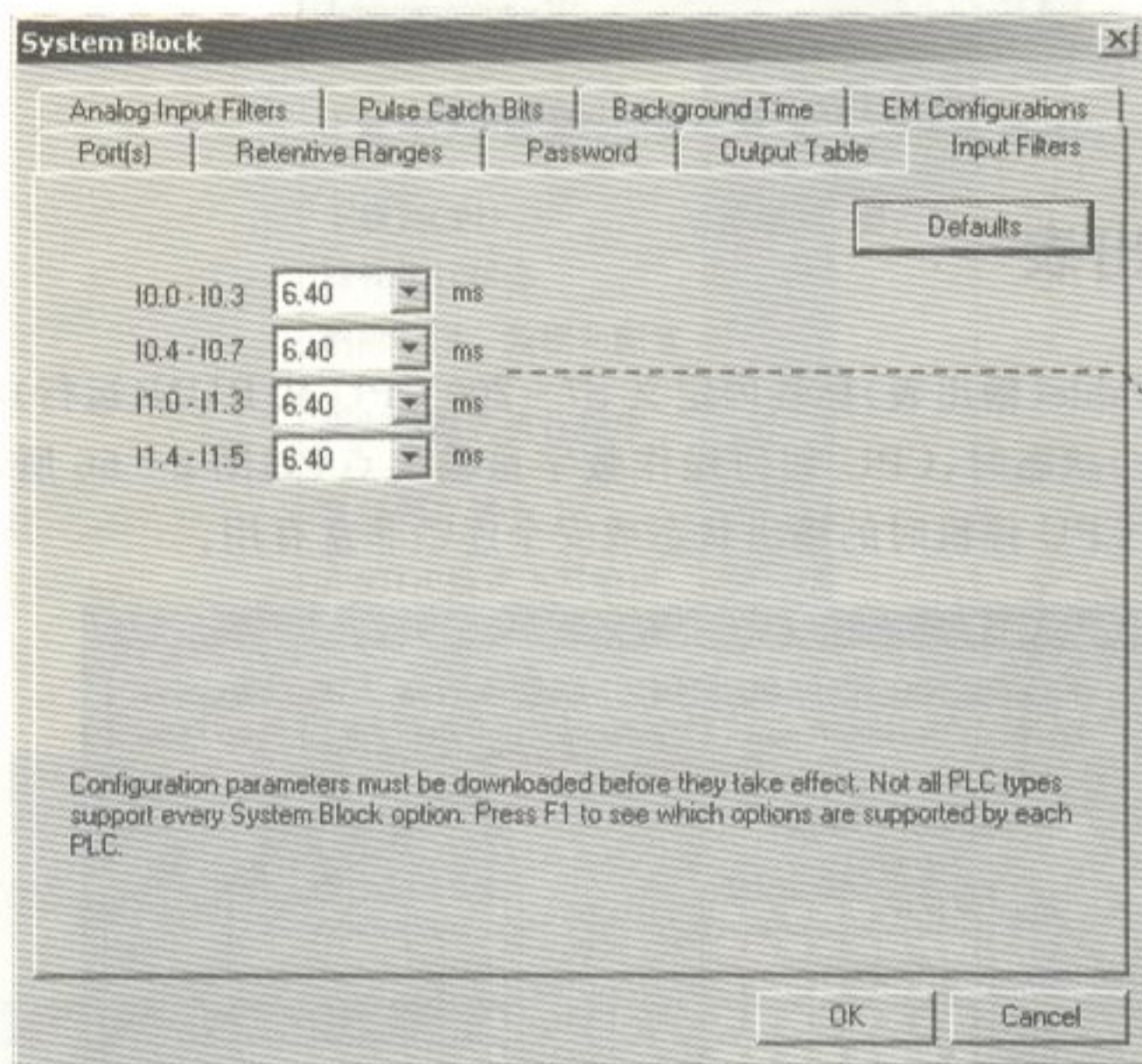
选取表格内的选择框使相应的输出位在 CPU 停机时保持ON(闭合状态)

图 2-56 输出表



2.4.5 输入滤波器

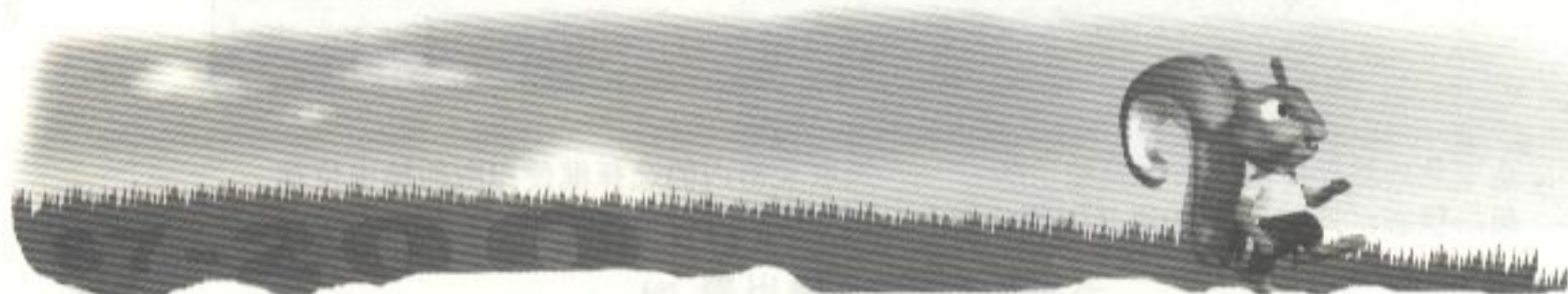
可在 Input Filters(输入滤波器)标签中分组为 S7-200 CPU 的数字量输入点指定输入滤波时间常数,范围为 0.2~12.8 ms(如图 2-57 所示)。



使用下拉按钮选择
滤波时间常数

图 2-57 输出滤波器的设定

✌ 此功能仅对 S7-200 上集成的数字量输入点有效。





2.4.6 模拟量输入滤波器

在 Analog Input Filters(模拟量输入滤波器)标签中允许为单个模拟量输入通道选择是否使用软件滤波器。软件滤波的输出就是对一定采样数的模拟量值取平均值(如图 2-58 所示)。所有选择使用滤波器的通道都具有同样的采样数和死区设置。模拟量输入滤波可以使信号变得比较稳定。

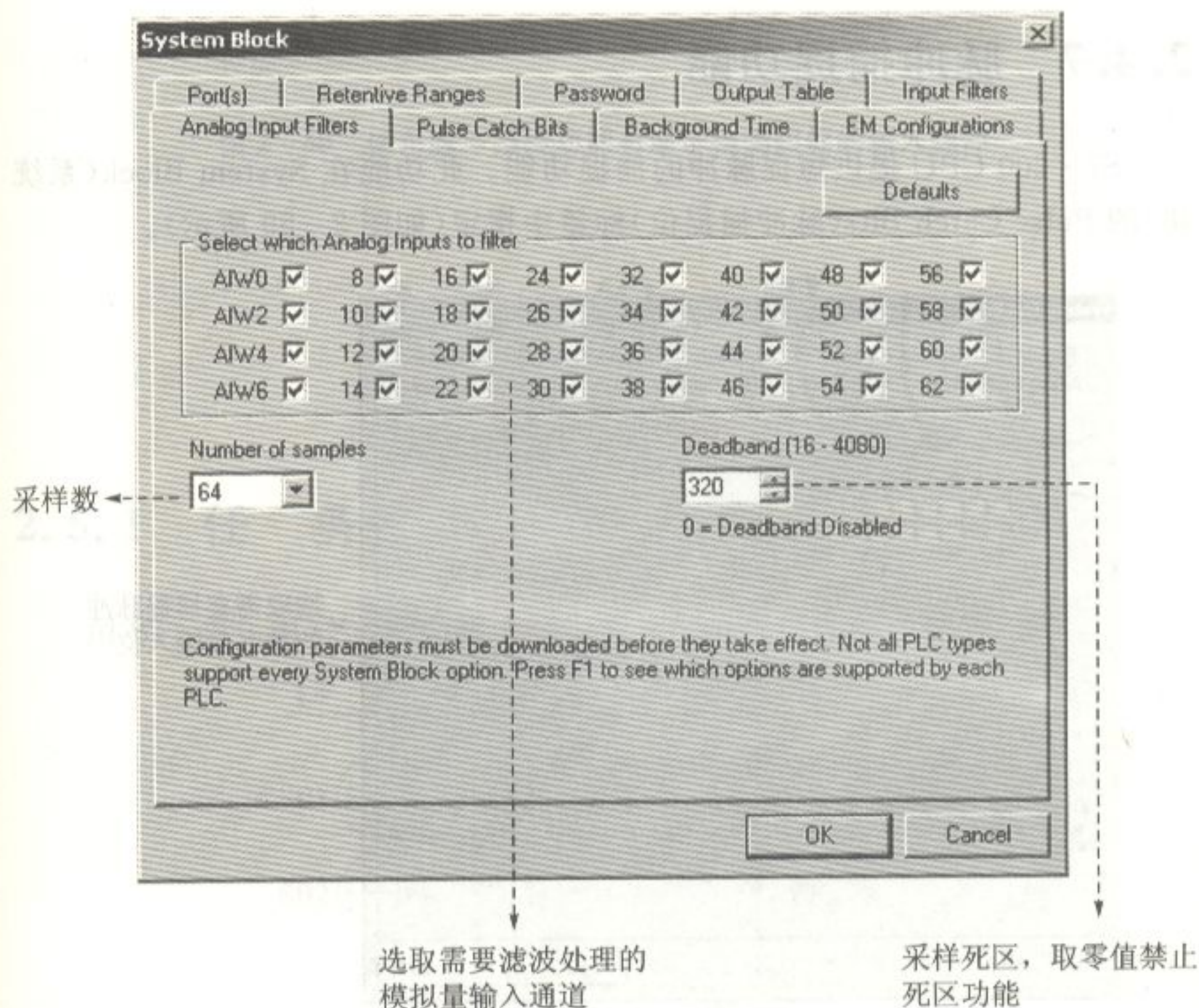


图 2-58 模拟量输入滤波器的设定

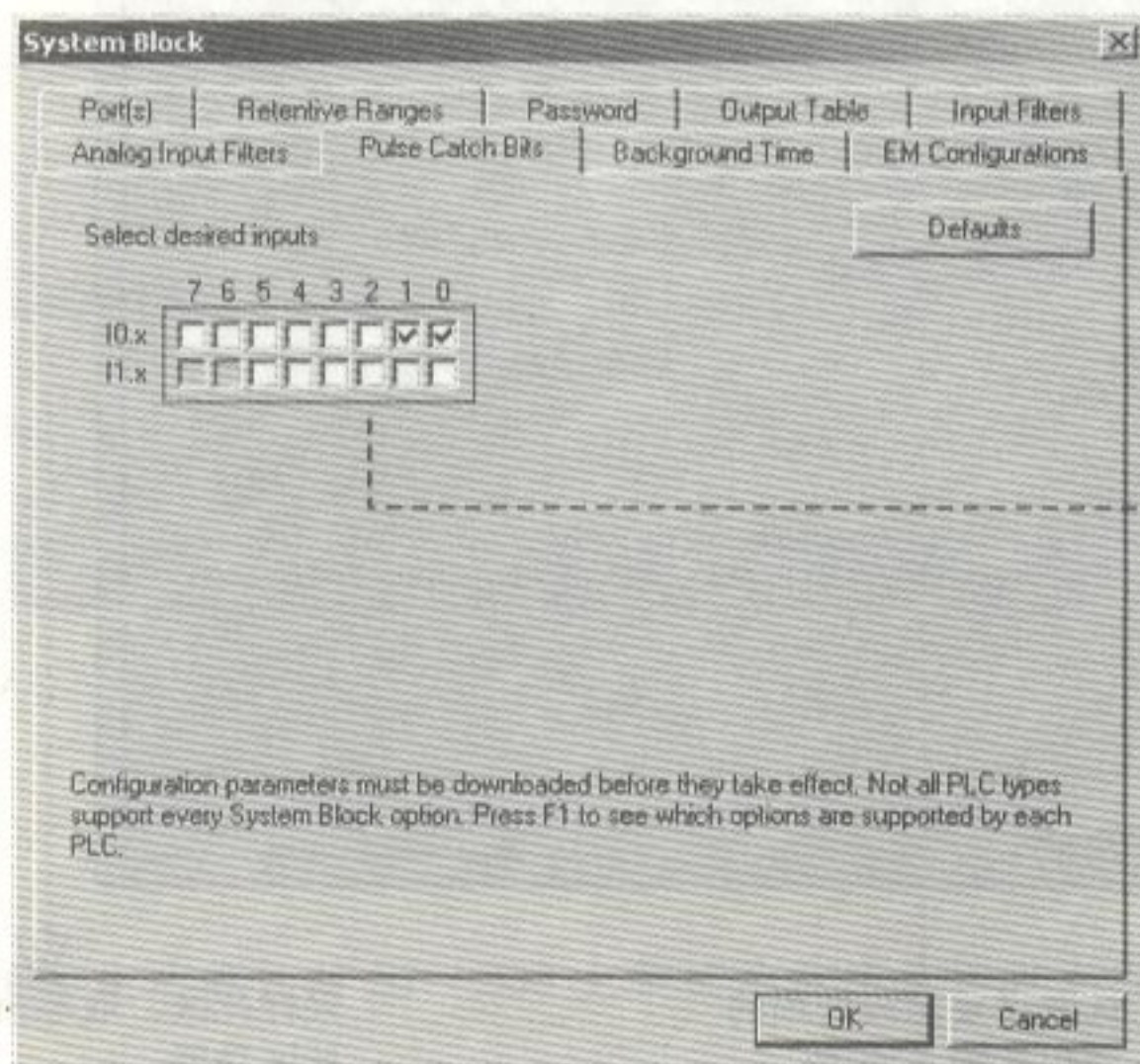
✌ 不使用滤波器的通道, 在程序中访问它时读取的是当前物理输入值。



为变化比较缓慢的模拟量输入值选用滤波器可以抑制波动；为变化较快的模拟量值选用较小的采样数和死区会加快响应速度；对高速变化的模拟量值不要使用滤波器。如果用模拟量传递数值信号，或者使用热电阻、热电偶、AS-Interface 模块时应当不用滤波器。

2.4.7 脉冲捕捉功能

S7-200 CPU 提供短促脉冲的捕捉功能。此功能在 System Block(系统块)的 Pulse Catch Bits(脉冲捕捉位)标签中指定(如图 2-59 所示)。



选取需要捕捉脉冲
的数字量输入位

图 2-59 脉冲捕捉功能的设定

S7-200 CPU 总在两次扫描周期之间读取物理数字量输入位的状态并更新输入映象区(如图 2-60 所示)，因此，短促的脉冲如果在程序扫描周期中



间到达, CPU 可能会丢失这个脉冲。脉冲捕捉功能提高了检测短脉冲信号的可靠性。

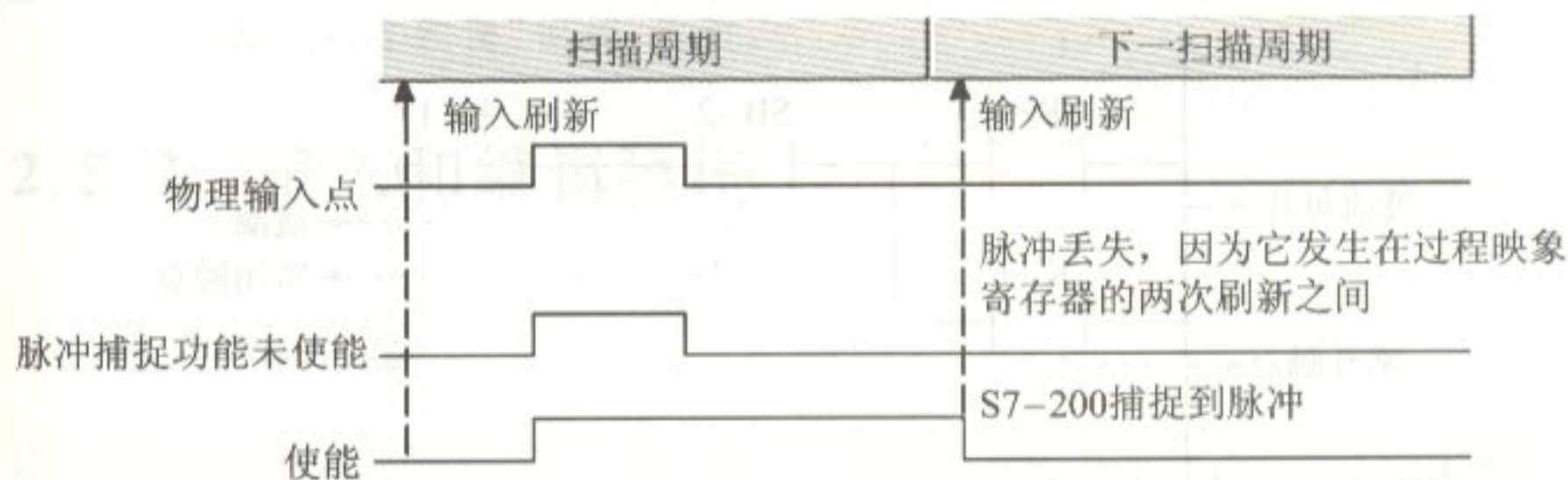


图 2-60 使用脉冲捕捉功能有助于检测短促的输入脉冲

2.5 编程

2.5.1 任务

用 S7-200 PLC 实现如图 2-61 所示的电气回路的功能。

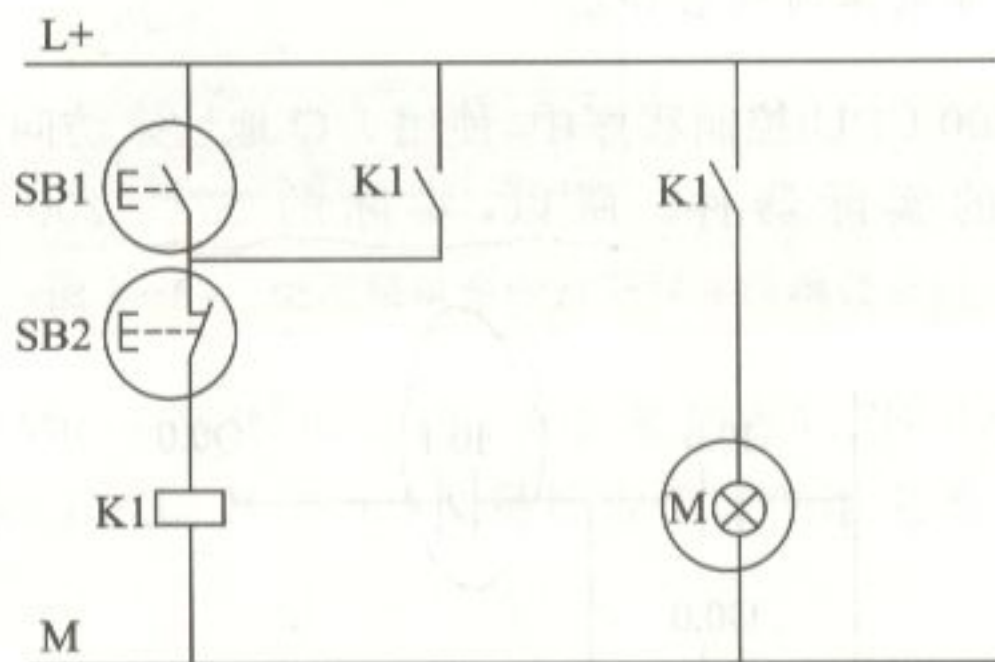


图 2-61 电气回路



✎ 把上面的电气原理图转动方向,并将触点、线圈等换用 Step7 - Micro/WIN32 中约定的符号表示,就可以得到一个梯形逻辑图(如图 2-62 所示)。

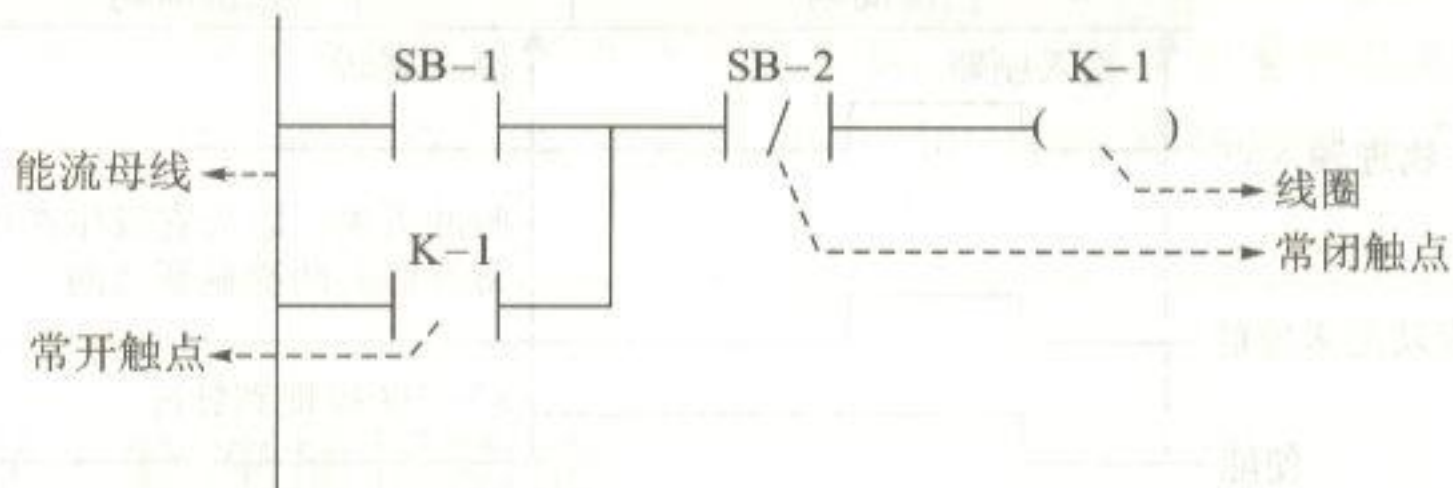


图 2-62 梯形逻辑图

电气图中圆圈内的都是外围器件。S7-200 CPU 通过 I/O(输入/输出)接口与外围器件联系,接受操作指令和检测各种状态,并把控制运算的结果输出。S7-200 CPU 内运行的程序实现了以前用继电器硬件连接实现的逻辑。

每一个实际 I/O(输入/输出)器件,都连接到 S7-200 CPU 上实际的 I/O 端子。S7-200 CPU 通过约定的方法,在程序中访问这些实际的 I/O 端子,称为寻址。

✎ 图 2-61 中 SB2 采用了常闭触点接法。具有“停止”、“急停”等功能的信号一般都应在硬件连接上使用常闭触点,防止因不能及时发现断线等故障而失去作用。

✎ 在 S7-200 CPU 控制程序中,使用 I/O 地址来访问实际连接到 CPU 输入/输出端子的实际器件。所以,实际的 S7-200 PLC 程序应如图 2-63 所示。

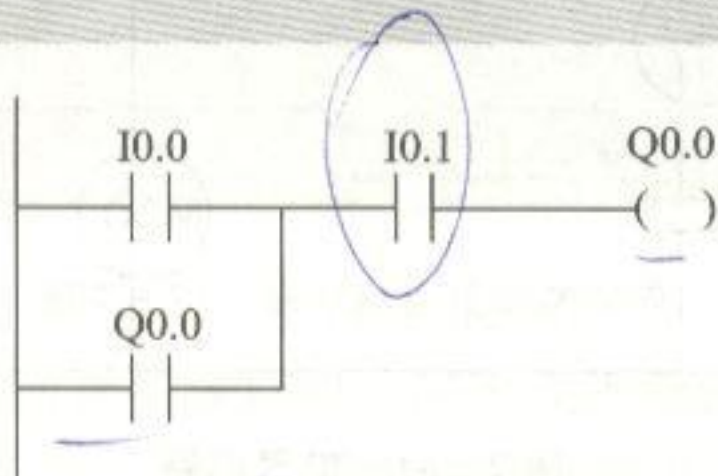
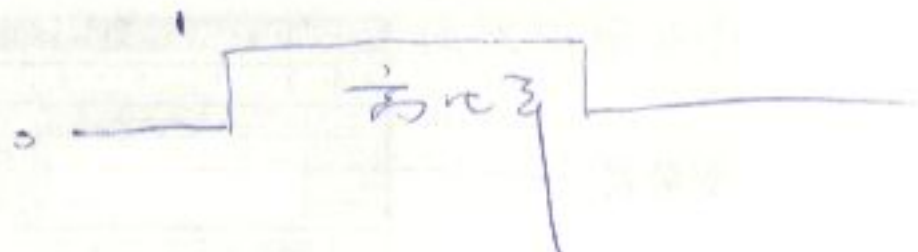


图 2-63 实际的 S7-200 PLC 程序




✌ 因为把 SB2 的常闭触点连接到 CPU 的 I0.1, 而 I0.1 平时处在高电平(有电压)输入状态(或称“1”状态), 所以, 程序中必须使用常开触点才能使“能流”通过。



2.5.2 输入和编辑程序

程序编辑器

☞ 在浏览条 View 视图中用鼠标按 Program Block(程序块)按钮, 或在使用菜单命令 View>Component>Program Editor 打开程序编辑器窗口(如图 2-64 所示)。

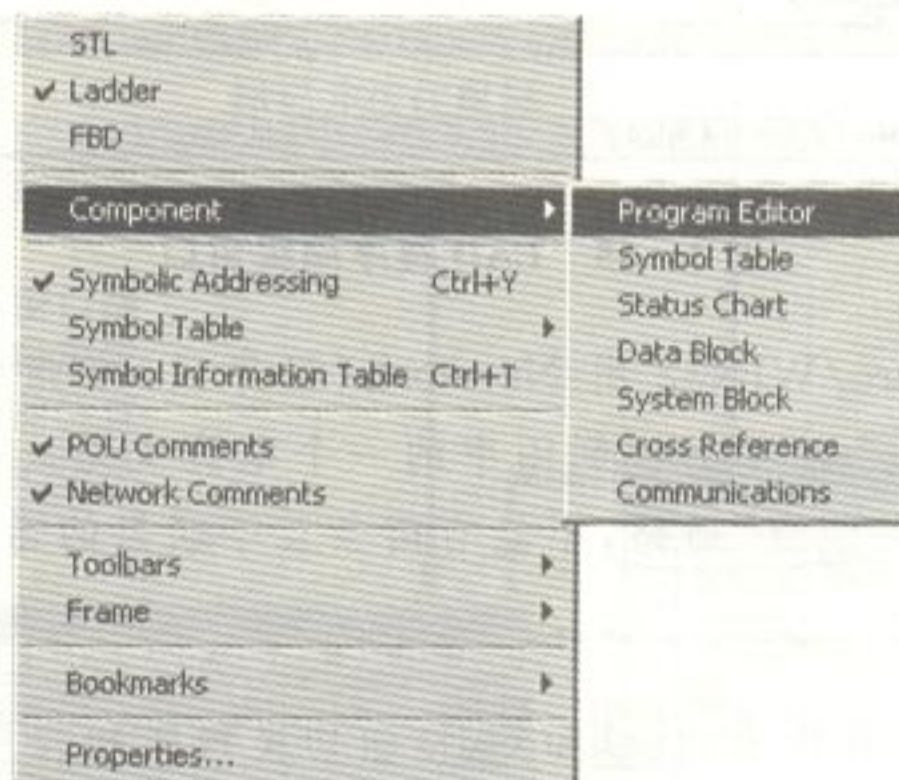


图 2-64 使用菜单命令打开程序编辑器窗口

☞ Step7-Micro/WIN32 支持 LAD(梯形图)、FBD(功能块图)和 STL(语句表)三种编程方式。其中, LAD 最接近传统的继电器逻辑电路, 也是默认的编程模式。



LAD 程序编辑窗口(如图 2-65 所示)

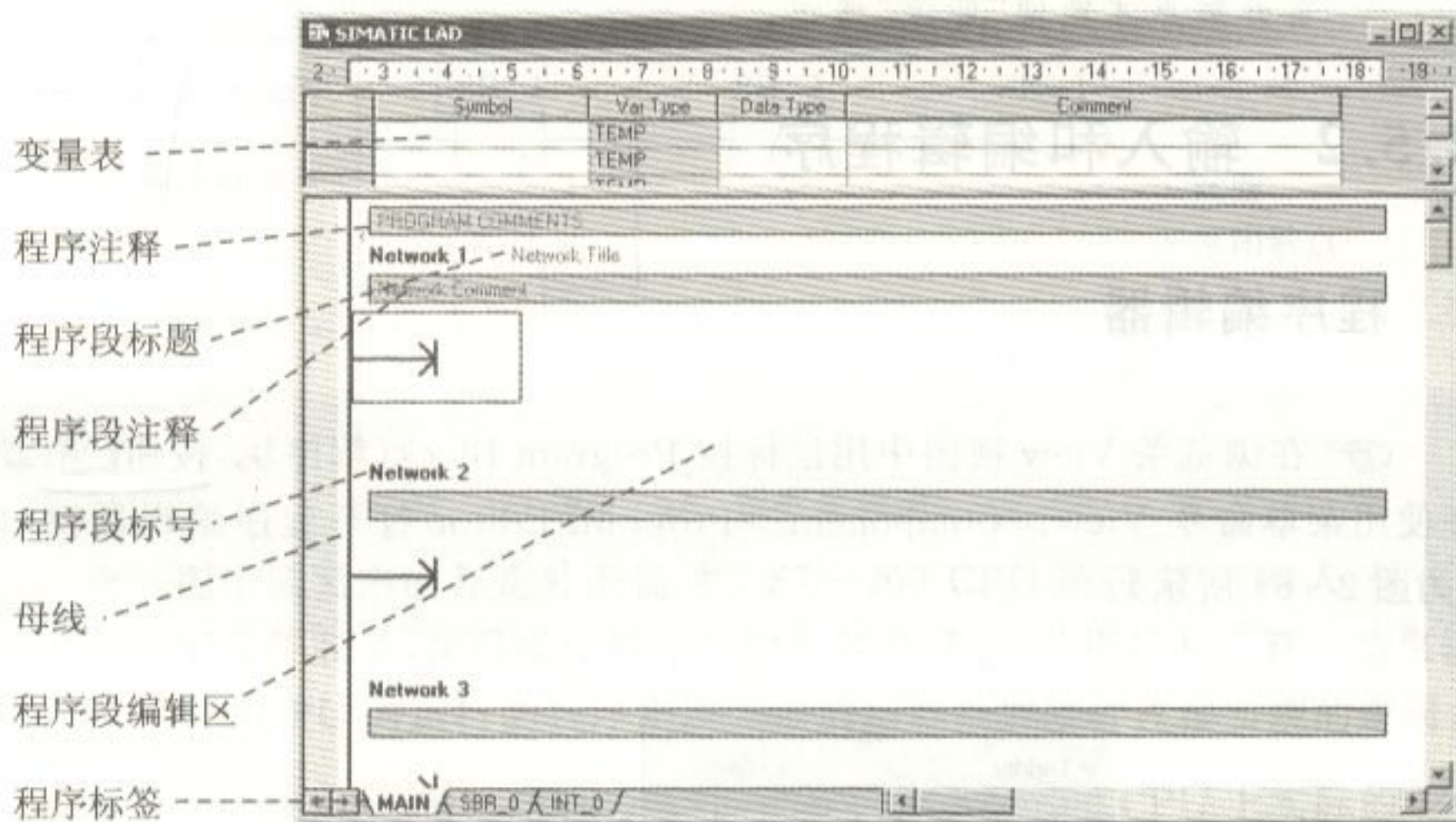




图 2-65 LAD 程序编辑窗口

✌ Step7 - Micro/WIN32 用 Network(程序段)来组织程序。每个程序段相当于继电器控制图中的一个电流支路。一个程序段内只能有一个“能流”通路 不能 有两条互不联系的通路。

🔧 可以使用工具栏上的  和  按钮切换程序和程序段注释显示。用鼠标在注释处单击就可直接编辑注释。

在 LAD 编辑器中有以下几种输入程序指令的方法:

- 鼠标拖放;
- 特殊功能键(F4, F6, F9);
- 鼠标双击;
- LAD 指令工具栏按钮。



鼠标拖放

鼠标单击打开指令树中类别分支,选择指令标记,按住鼠标左键不放,将其拖到编辑器窗口内合适的位置上再释放(如图2-66所示)。

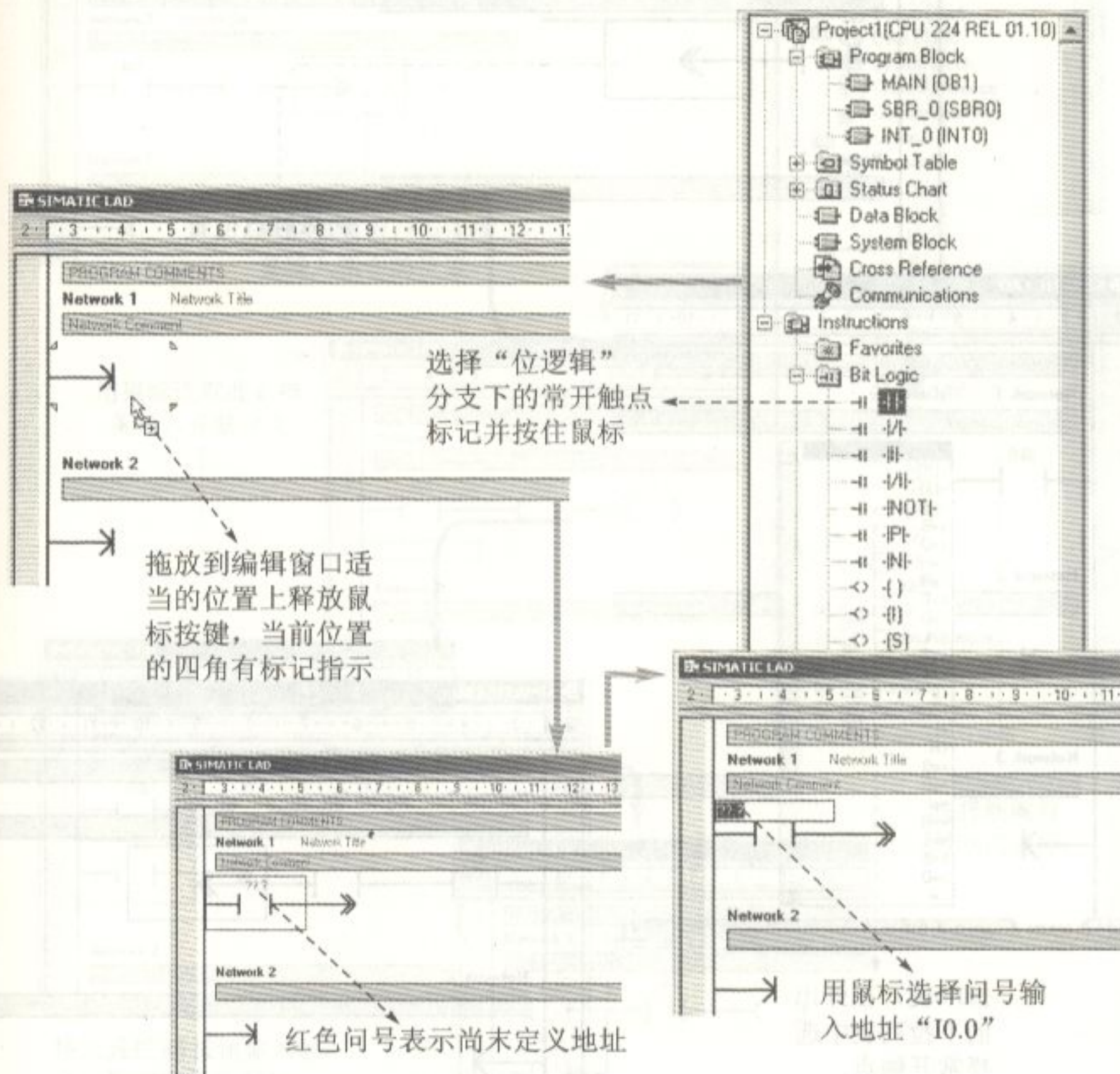


图 2-66 使用鼠标拖放编辑程序



特殊功能键(如图 2-67 所示)

对边补漏

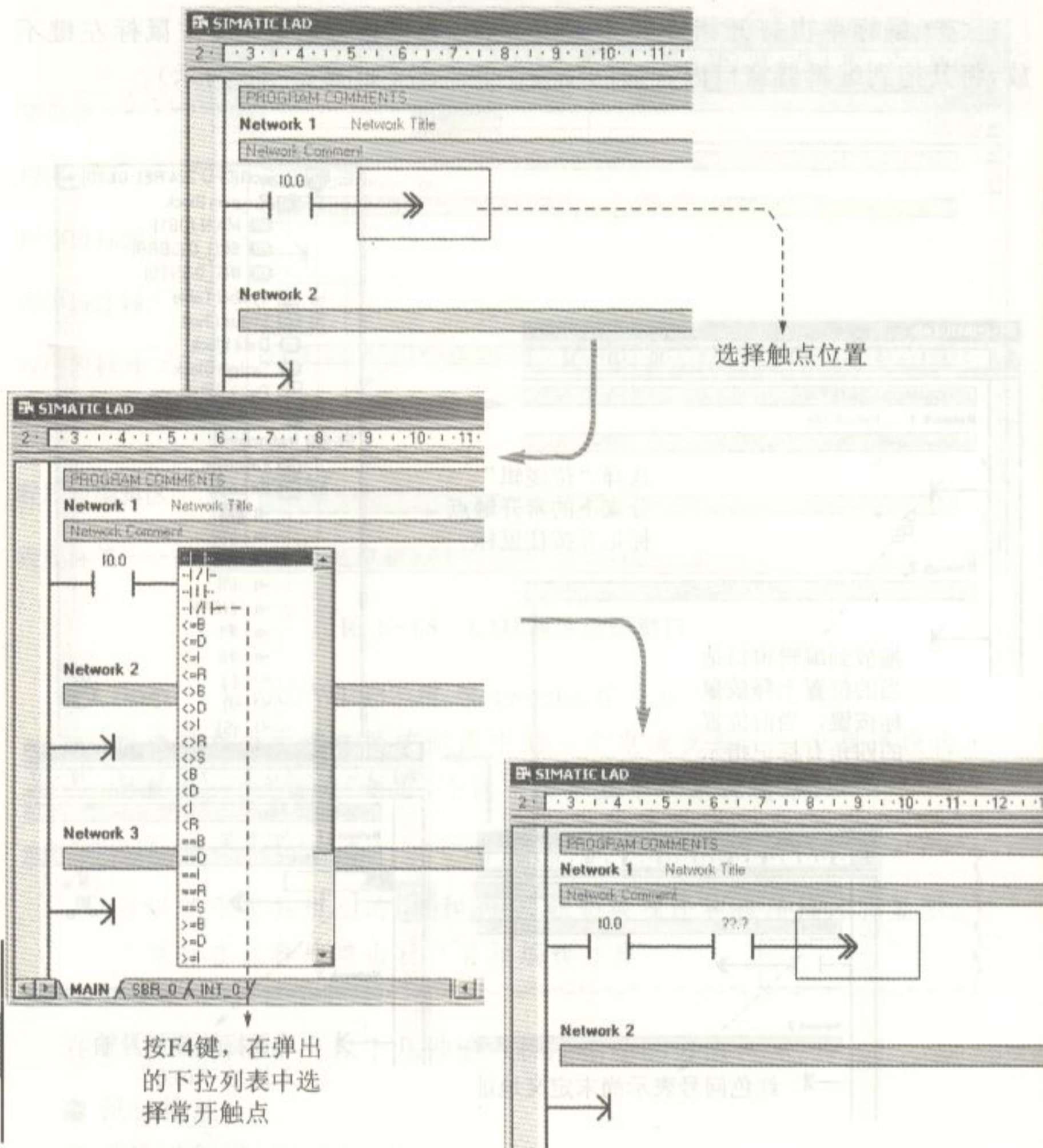


图 2-67 使用特殊功能键编辑程序



鼠标双击(如图 2-68 所示)

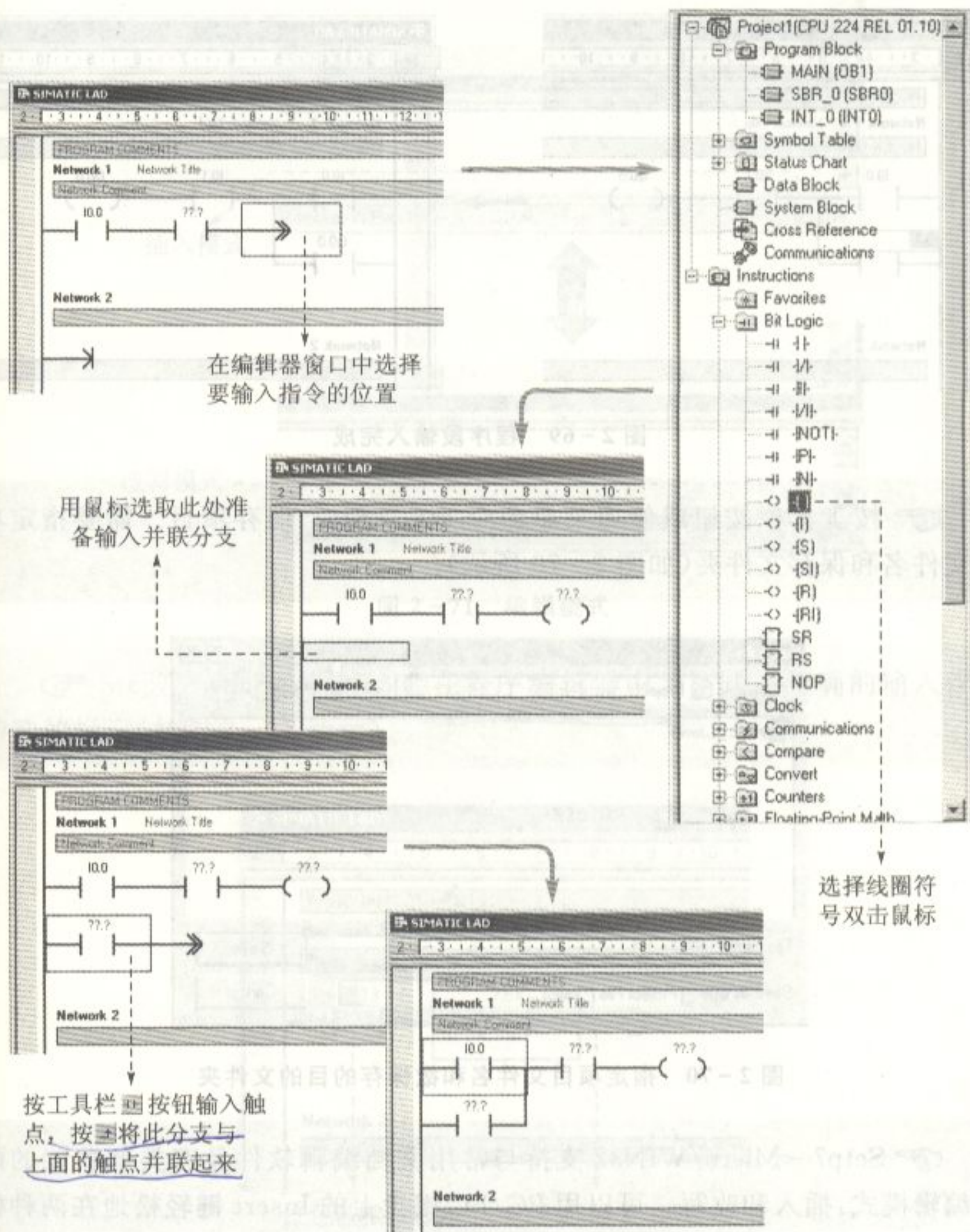


图 2-68 双击鼠标输入程序



☞ 输入触点的地址,完成程序段的输入(如图 2-69 所示)。

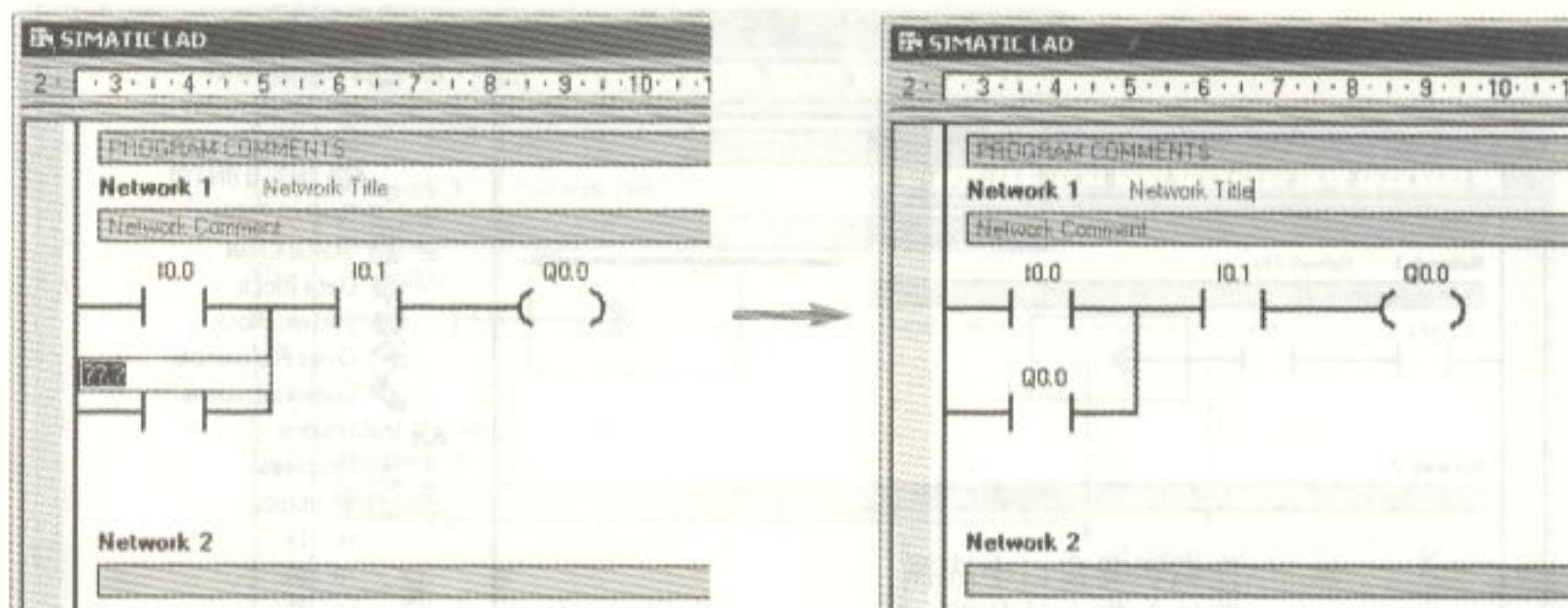


图 2-69 程序段输入完成

☞ 按工具栏按钮或使用菜单命令 File>Save,保存项目。需要指定项目文件名和保存文件夹(如图 2-70 所示)。

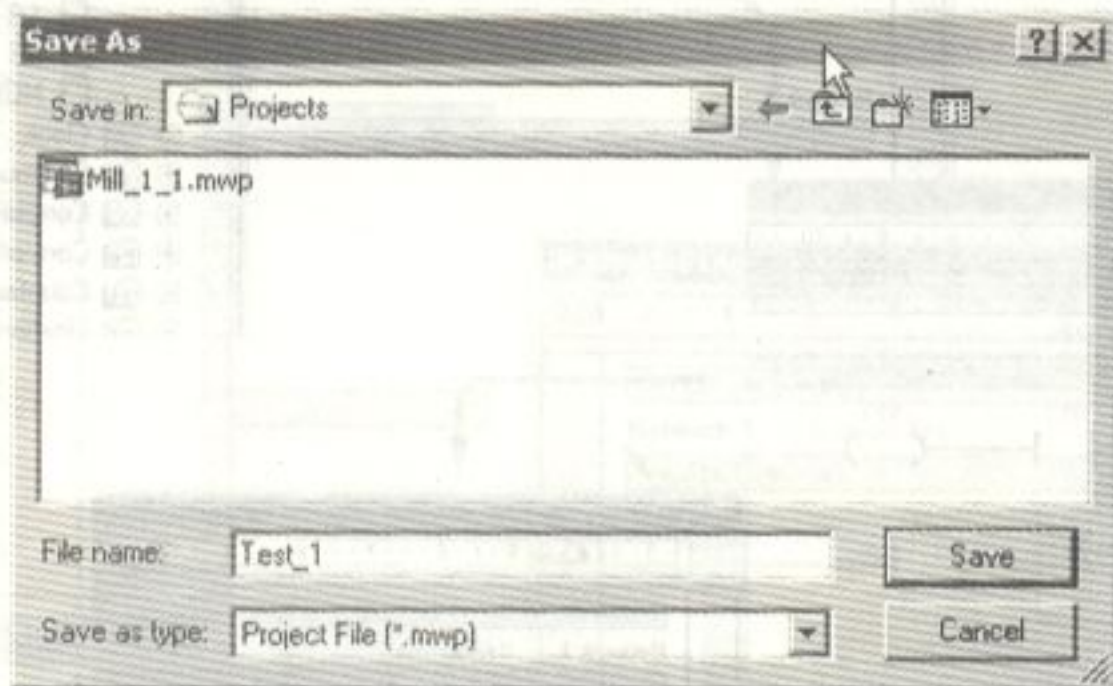


图 2-70 指定项目文件名和欲保存的目的文件夹

☞ Setp7 - Micro/WIN32 支持与常用文档编辑软件具有类似功能的两种编辑模式:插入和改写。可以用 PG/PC 键盘上的 Insert 键轻松地在两种模式间切换,视窗状态栏右下角显示了当前编辑模式(如图 2-71 所示)。

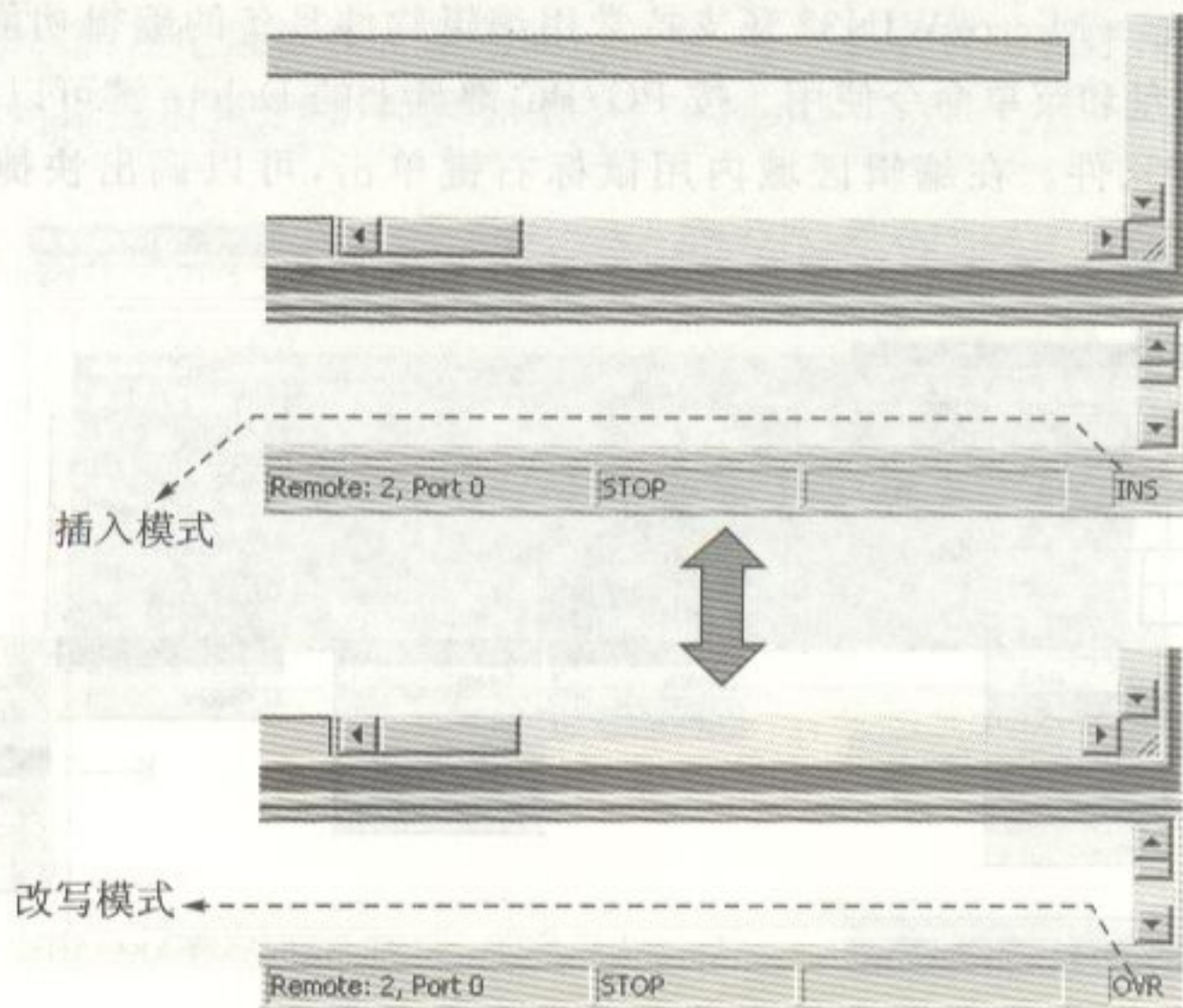
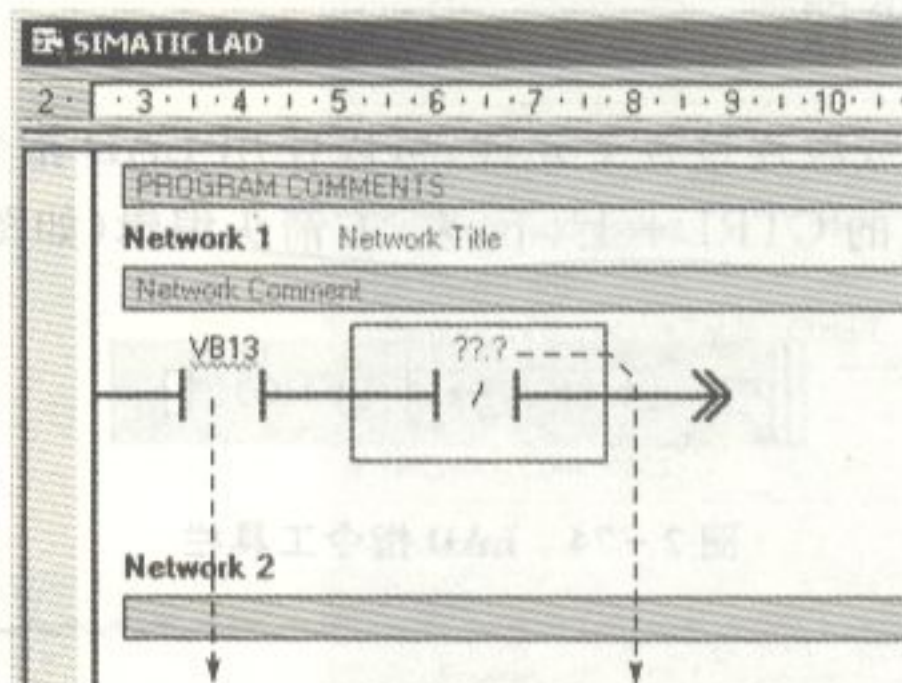


图 2-71 编辑模式

Step7—Micro/WIN32 会在程序编辑器中为格式不正确的输入作出特殊的标记(如图 2-72 所示)。



红色波浪线表示
不合法的输入

红色问号表示需
要指定操作数

图 2-72 程序编辑器不能识别的标记



☞ Step - Micro/WIN32 还支持常用编辑软件具备的编辑功能,可以方便地通过键盘和菜单命令使用。按 PG/PC 键盘上的 Delete 键可以删除光标所在位置的元件。在编辑区域内用鼠标右键单击,可以调出快捷菜单(如图 2-73 所示)。

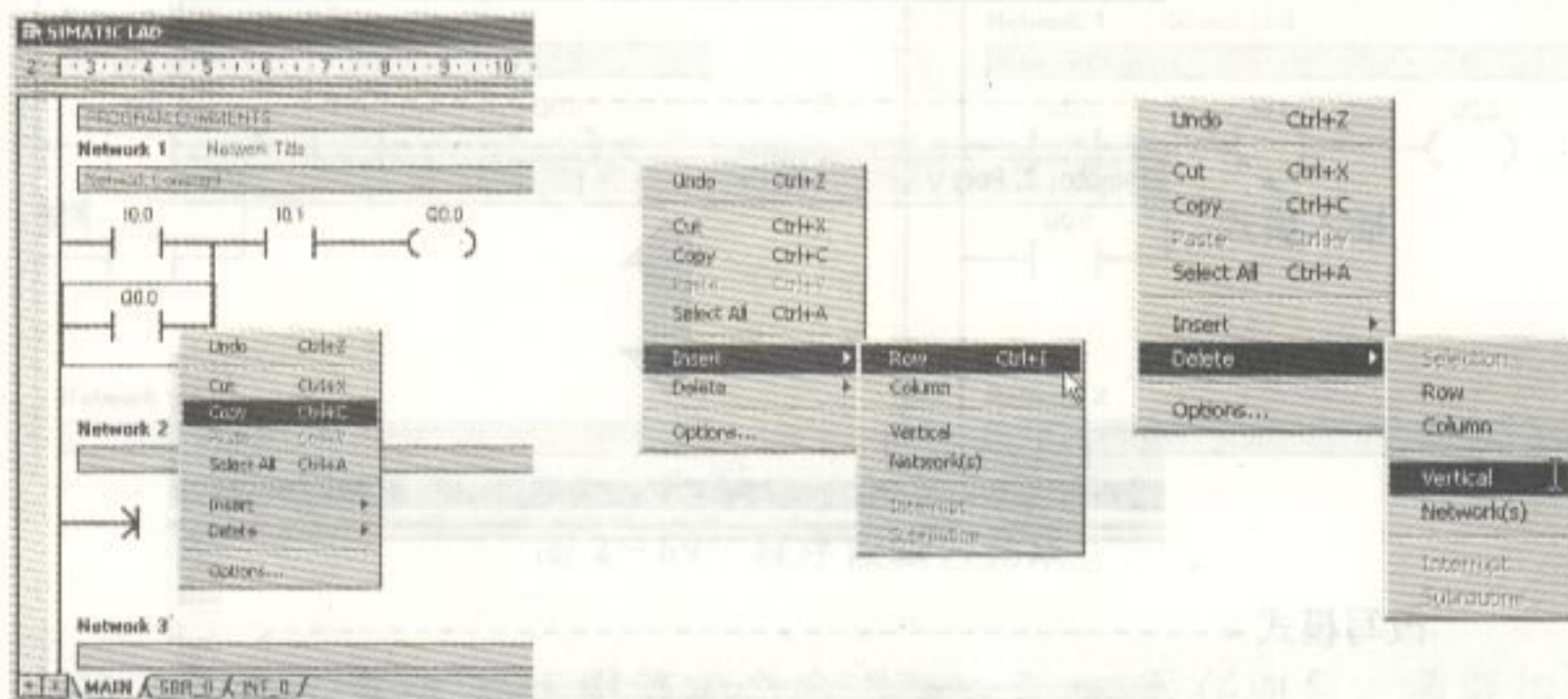


图 2-73 快捷菜单的调取

✌ 插入指令总是在鼠标当前位置的左边或上面插入新的元件。Vertical 用来插入和删除垂直的并联线段。如果选择插入程序段,将在当前段的前面插入新段。同样的规律也适用于粘贴等操作。

编辑 LAD 线段

LAD 程序使用线段连接各个元件,可以使用 LAD 指令工具栏上的连线按钮,或者用键盘上的 CTRL+上、下、左、右箭头编辑(如图 2-74 所示)。



图 2-74 LAD 指令工具栏



编辑中可能会出现指令显示不整齐的现象,这种情况不用特别处理,执行一次 Compile(编译)命令后就会自动排列整齐。



☞ 在编辑器电源母线左侧用鼠标单击,可以选取整个程序段;按住鼠标左键拖动,可以选取多个程序段(如图 2-75 所示)。

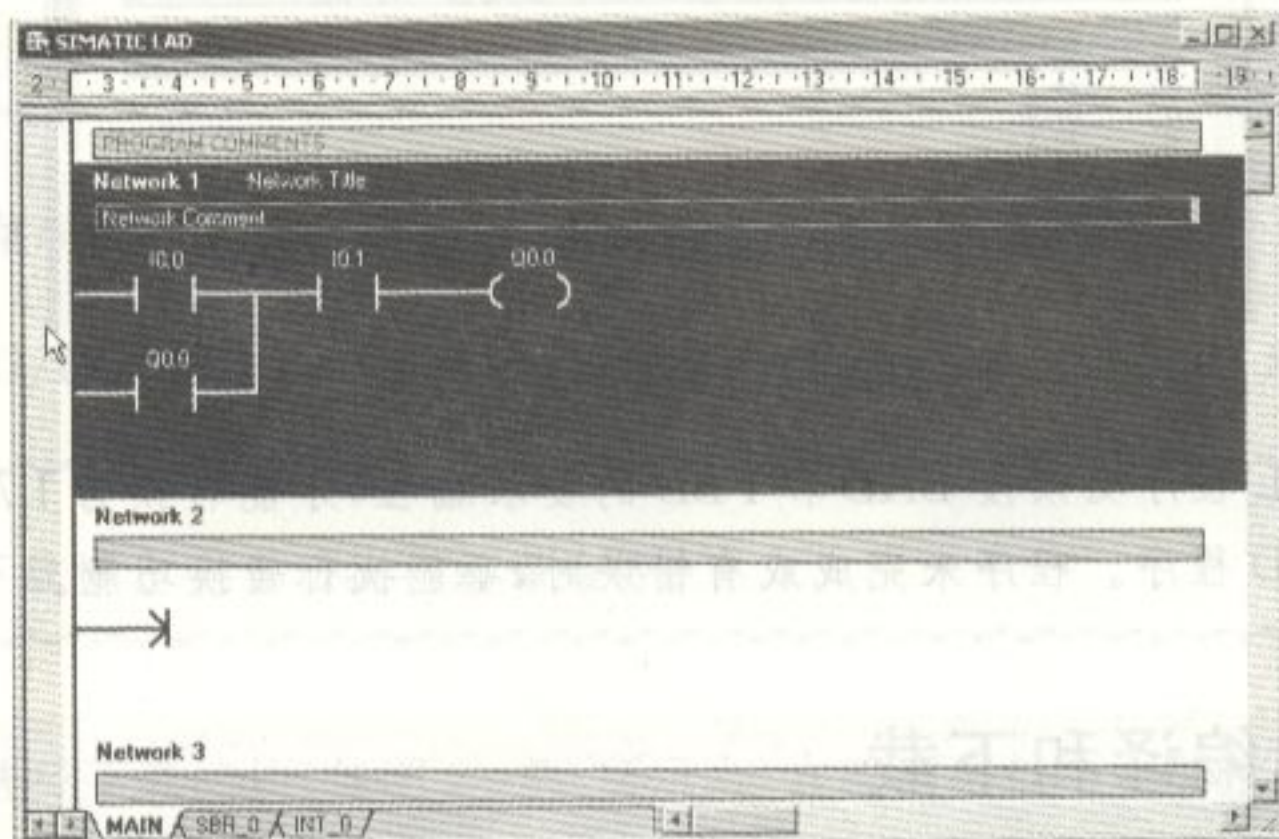


图 2-75 程序段的选取

切换编程语言

使用菜单 View 中的命令,可在三种编程语言间切换(如图 2-76 所示)。

选取编程语言 ←

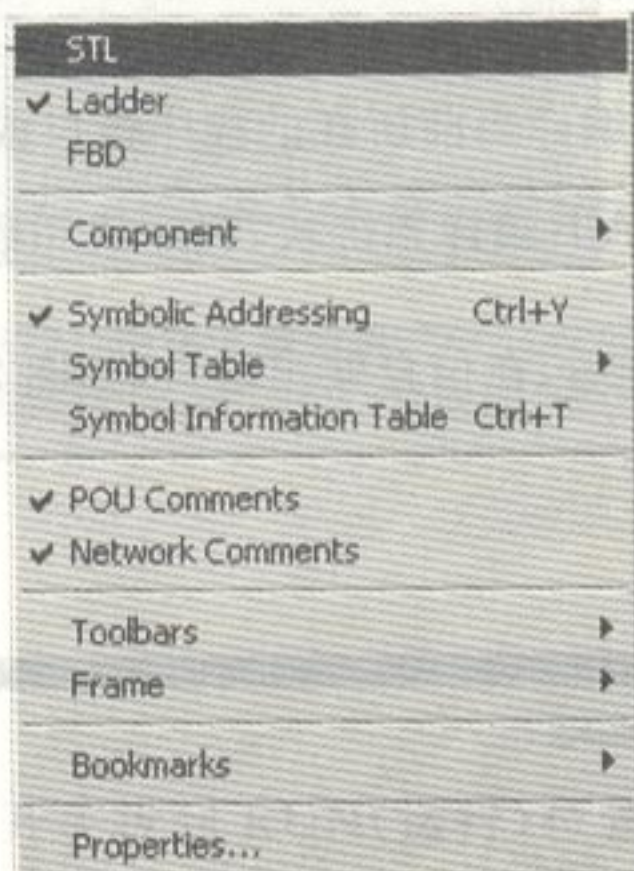


图 2-76 编程语言的切换



☞ 把我们的程序改用 STL 显示(如图 2-77 所示)。

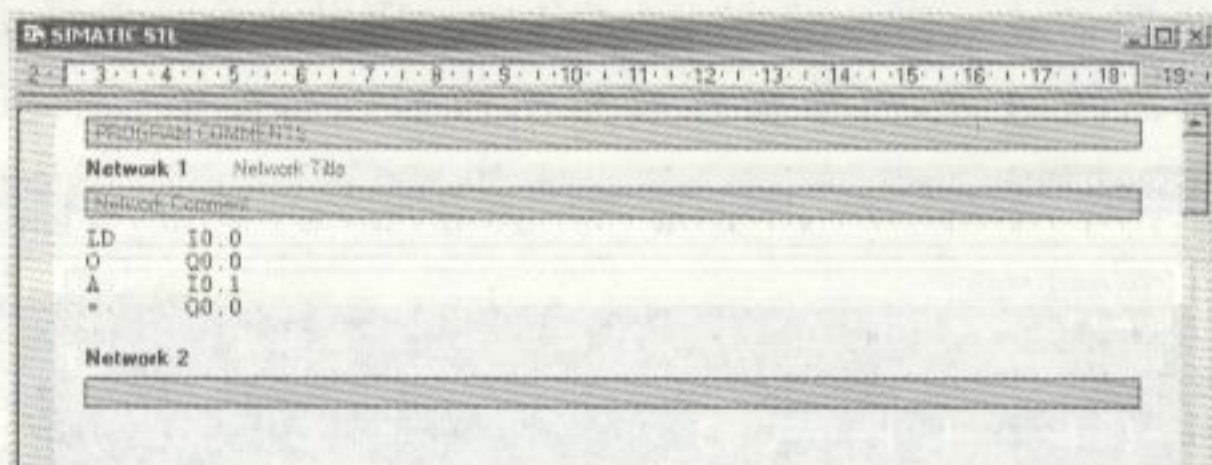





图 2-77 STL 显示的程序


✌ STL 程序必须按 LAD 和 FBD 的要求编程,才能转换为 LAD 和 FBD 程序。程序未完成或有错误时,不能执行转换功能。

2.5.3 编译和下载

在 Step7 - Micro/WIN32 中编辑的程序必须编译成 S7-200 CPU 能识别的机器指令,才能下载到 S7-200 CPU 内运行。

使用菜单 PLC>Compile 和 PLC>Compile All 命令,或者用工具栏按钮  或  执行编译功能。

 Compile:编译当前所在的程序窗口或数据块窗口。

 Compile All:编译系统块、程序块和数据块。

☞ 执行编译后,在信息输出窗口(如图 2-78 所示)会显示相关的结果。

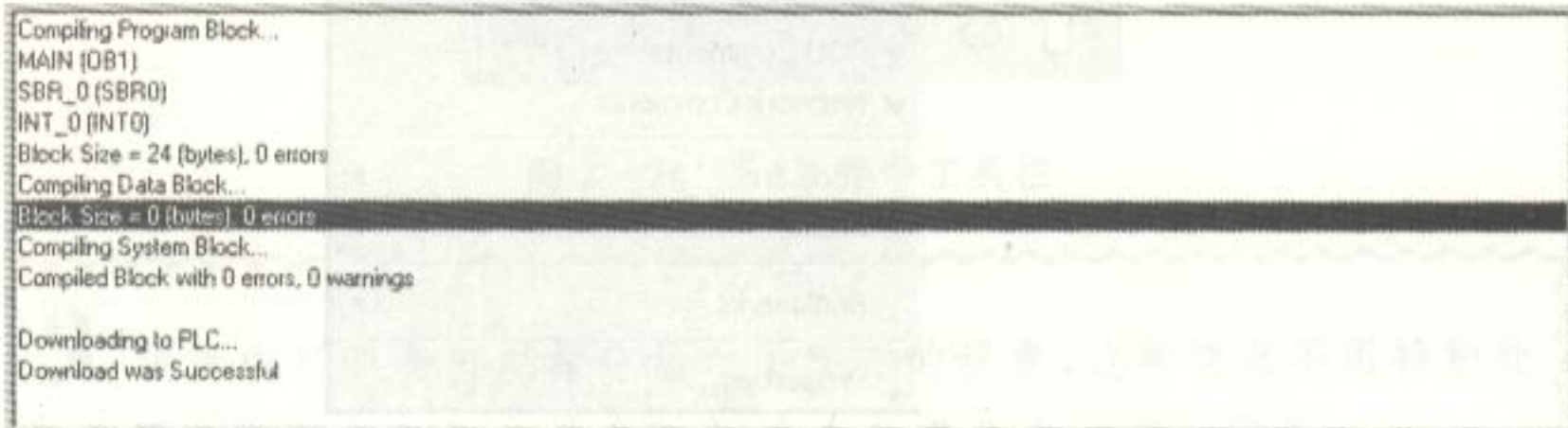



图 2-78 信息输出窗口显示的结果



输出窗口会显示程序块和数据块的大小,也会显示编译中发现的错误。用鼠标双击错误信息可以在程序编辑器中跳转到相应程序段以便修改。

使用菜单命令 File>Download...,或按工具栏按钮执行下载。



PG/PC→S7-200 CPU 为下载;S7-200 CPU→PG/PC 为上载。



下载操作会自动执行编译命令。

选择下载的块(如图 2-79 所示)。在输出窗口中会显示相关消息。

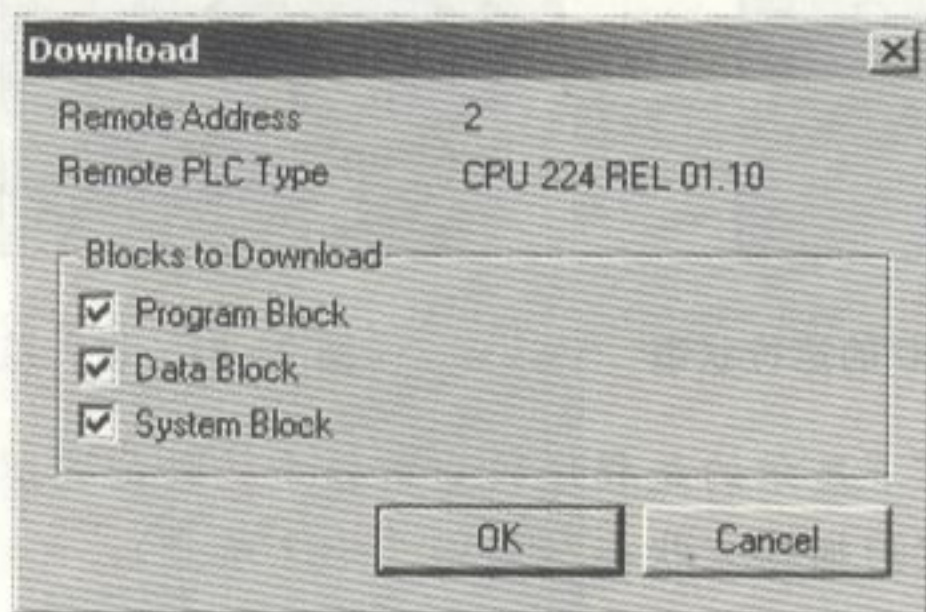




图 2-79 选择下载的块

2.5.4 运行和调试

运行程序

将 S7-200 CPU 上的状态开关拨到 RUN(运行)位置,CPU 上的黄色 STOP(停止)状态指示灯灭,绿色 RUN 灯点亮。



☞ 如果 S7-200 CPU 上的状态开关处于 RUN(运行)或 TERM(终端)位置,您还可以在 Step7 - Micro/WIN32 软件中使用菜单命令 PLC>RUN 和 PLC>STOP,或者工具栏按钮  和  改变 CPU 的运行状态(如图 2-80 所示)。

☞ 监控程序状态

使用菜单命令 Debug>Program Status 或者工具栏上的按钮,进入程序状态监控(如图 2-81 所示)。

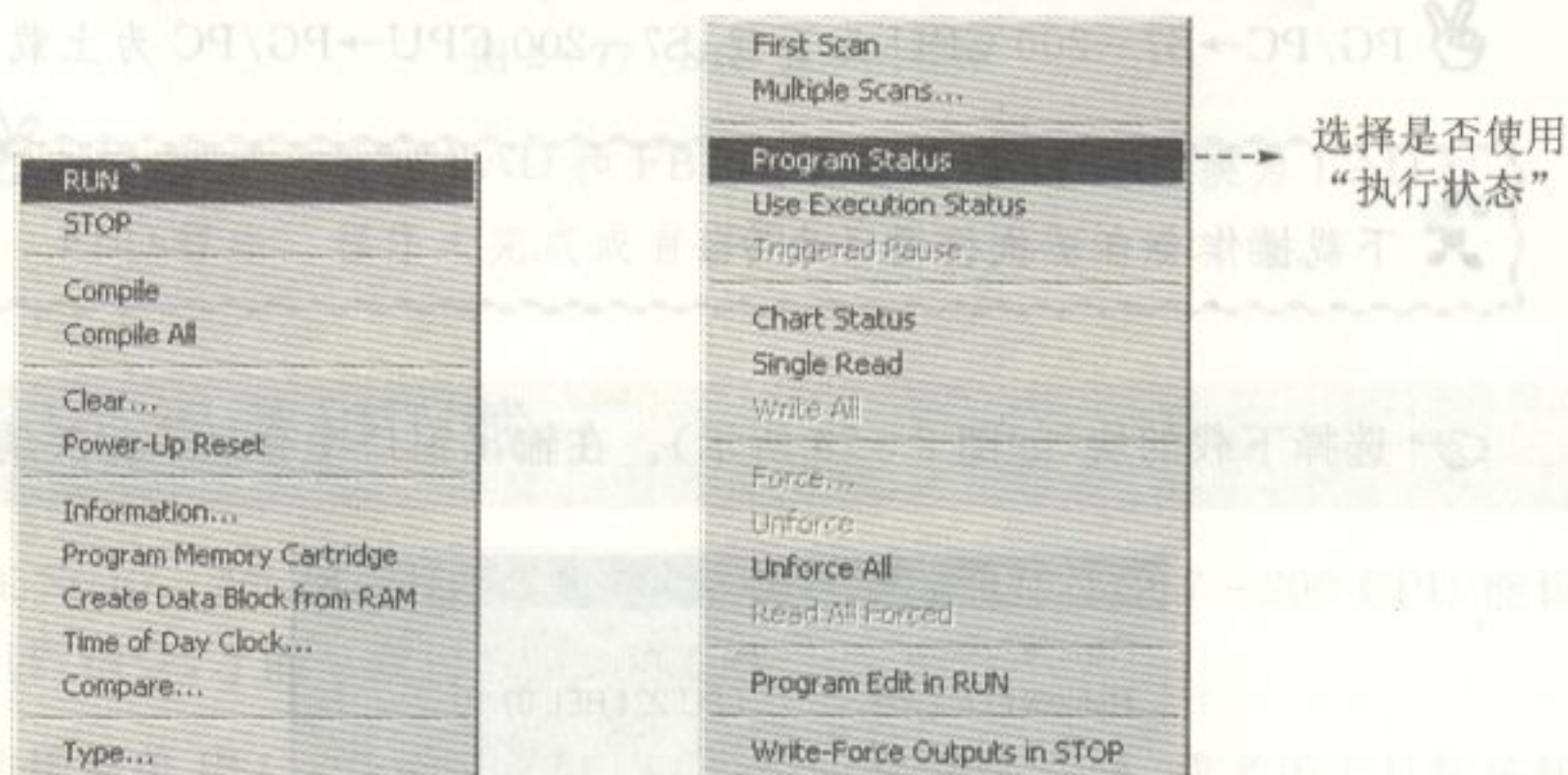


图 2-80 改变 CPU 的运行状态

图 2-81 进入监控程序状态

✌ “执行状态”下显示的是程序段当前每个元件的实际状态,如不选中,将监控“扫描结束状态”(如图 2-82 所示)。

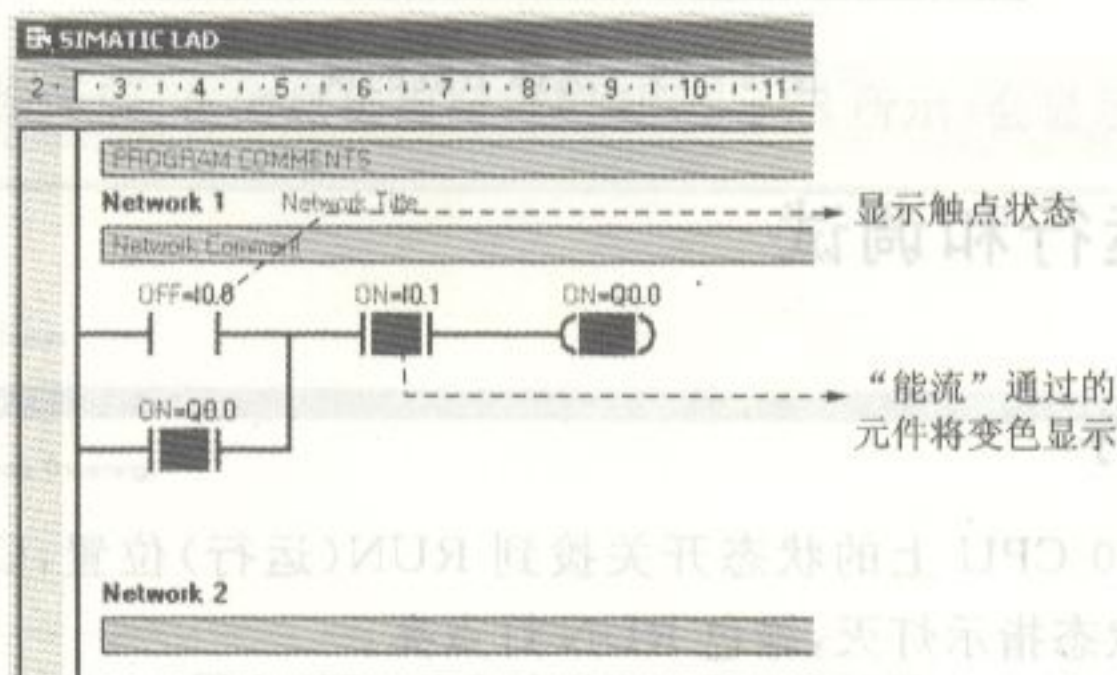


图 2-82 程序状态监控



程序监控状态不能完全如实显示变化迅速的元件的状态。屏幕刷新的速率取决于 PG/PC 与 S7-200 CPU 的通讯速率。

状态图

使用状态图可以监控数据。

使用鼠标单击浏览条 View 栏的 Status Chart(状态图)图标,或使用菜单命令 View>Component>Status Chart 打开状态图窗口(如图 2-83、图 2-84 所示)。

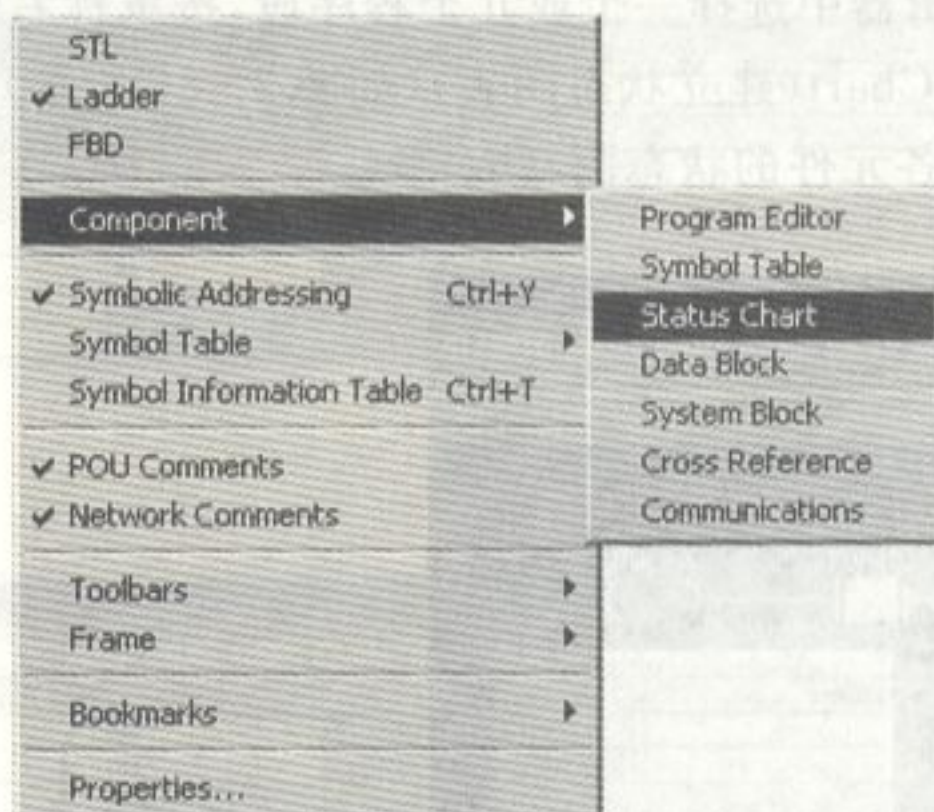


图 2-83 打开状态图


Status Chart				
	Address	Format	Current Value	New Value
1	I0.0	Bit		
2	I0.1	Bit		
3		Signed		
4	Q0.0	Bit		
5		Signed		
6		Signed		

输入数据地址

鼠标单击此处可选择数据显示格式

图 2-84 状态图窗口



使用菜单命令 Debug>Chart Status, 或按工具栏按钮  监控状态图表格内的数据值(如图 2-85 所示)。

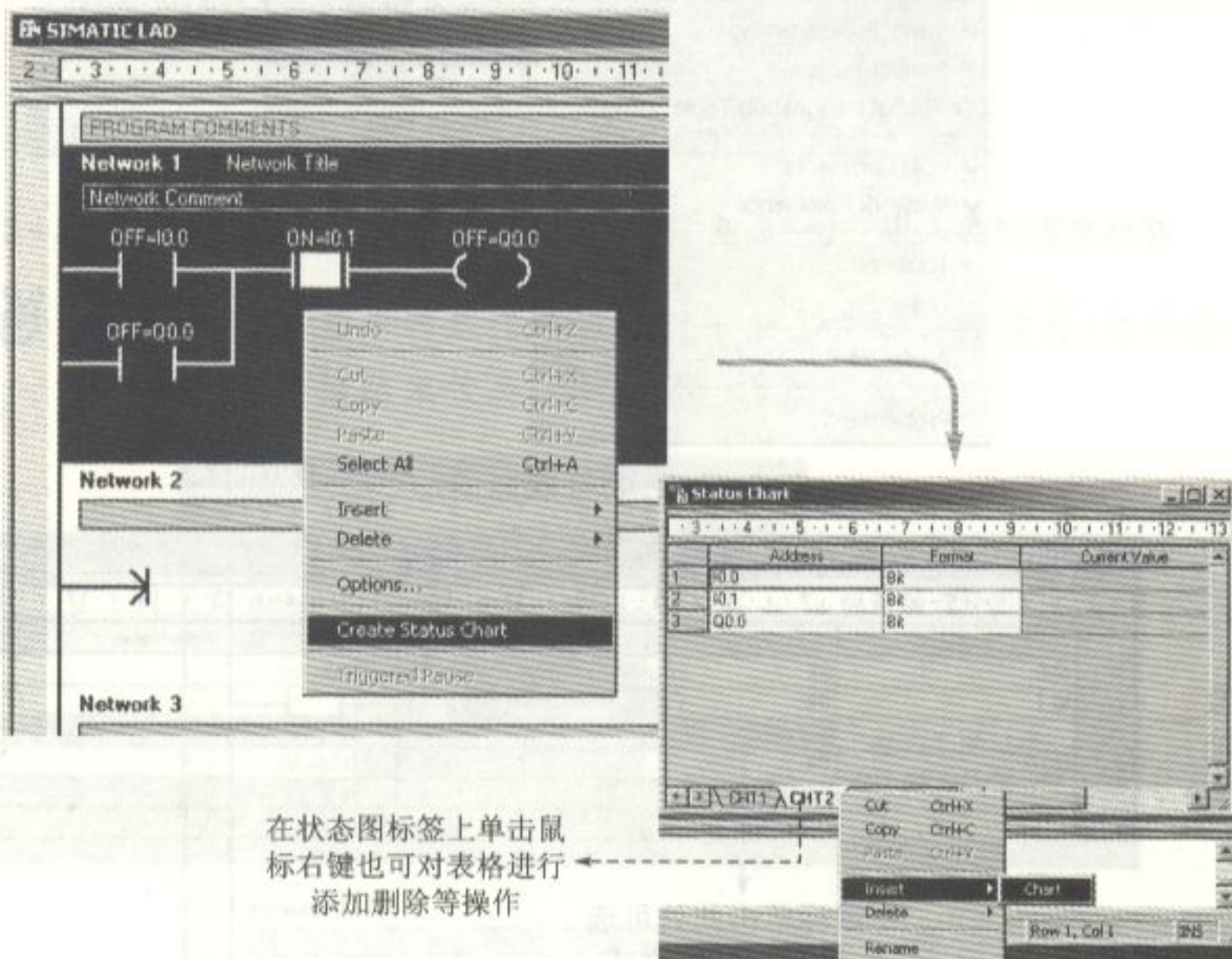
Status Chart																	
	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	
	Address		Format		Current Value		New Value										
1	I0.0		Bit		280												
2	I0.1		Bit		281												
3			Signed														
4	Q0.0		Bit		280												
5			Signed														
6			Signed														

当前值

新值在此设置

图 2-85 监控表格内数据值

在程序编辑器中选择一个或几个程序段, 按鼠标右键, 在快捷菜单中选择 Create Status Chart(建立状态图表)(如图 2-86 所示), 能快速生成一个包含所选程序段内各元件的状态图新表格。



The screenshot shows the SIMATIC LAD editor with a context menu open over Network 1. The menu includes options like Undo, Cut, Copy, Paste, Select All, Insert, Delete, Options..., Create Status Chart, and Triggered Pause. An arrow points from the 'Create Status Chart' option to a newly generated 'Status Chart' window. This window displays a table with columns for Address, Format, and Current Value, containing data for I0.0, I0.1, and Q0.0. A second context menu is shown over the 'Status Chart' window, with an arrow pointing to the 'Chart' option in the 'Insert' submenu.

在状态图标签上单击鼠标右键也可对表格进行添加删除等操作


图 2-86 快速建立状态图表

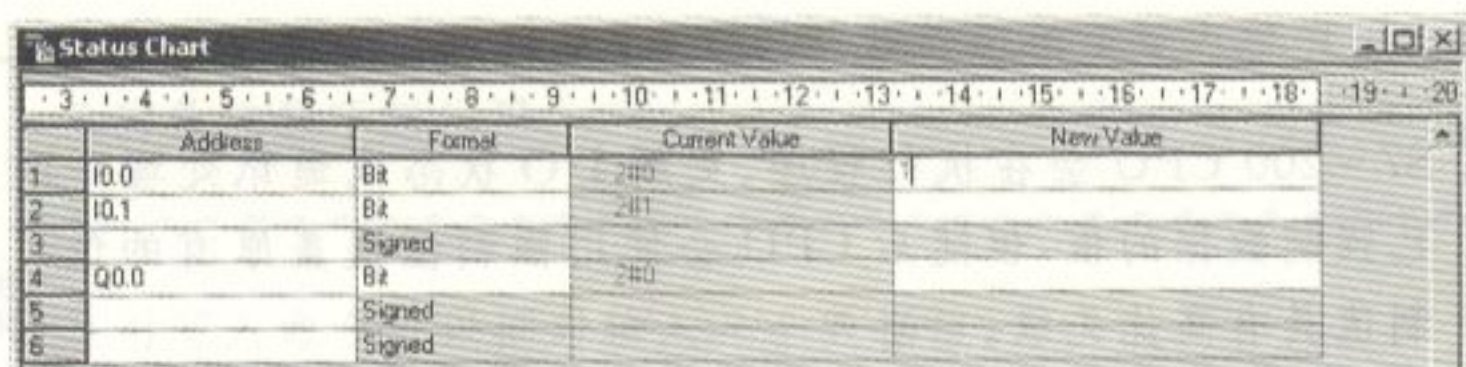


强制功能

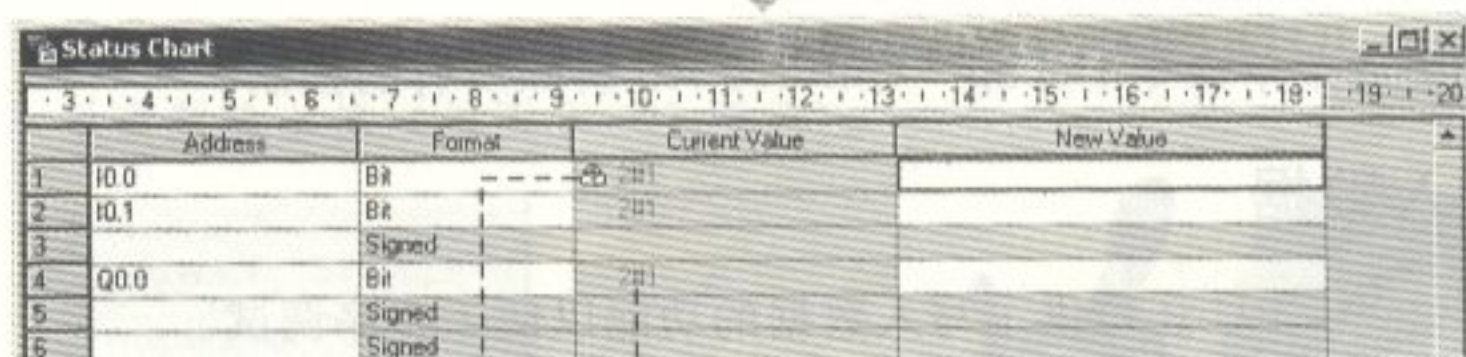
S7-200 CPU 提供了 Force(强制)功能以方便程序调试工作(例如,在现场不具备某些外部条件的情况下)。您可以对所有的 I/O 以及多达 16 个内部存储器数据或模拟 I/O 量(AI 或 AO)进行强制。

如果没有实际的 I/O 接线,也可以用强制功能调试例子程序。

☞ 显示状态图并且使其处于监控状态,在 New Value(新值)列中写入希望强制成的数据,然后按工具栏按钮 ,或者使用菜单命令 Debug>Force 强制数据(如图 2-87 所示)。



	Address	Format	Current Value	New Value
1	I0.0	Bit	240	
2	I0.1	Bit	241	
3		Signed		
4	Q0.0	Bit	240	
5		Signed		
6		Signed		

	Address	Format	Current Value	New Value
1	I0.0	Bit	241	
2	I0.1	Bit	241	
3		Signed		
4	Q0.0	Bit	241	
5		Signed		
6		Signed		


处在强制状态的标记  符合逻辑的结果

图 2-87 强制功能

☞ 对于无须改变值的变量,只需在 Current Value(当前值)列中选中它,然后使用强制命令(如图 2-88 所示)。

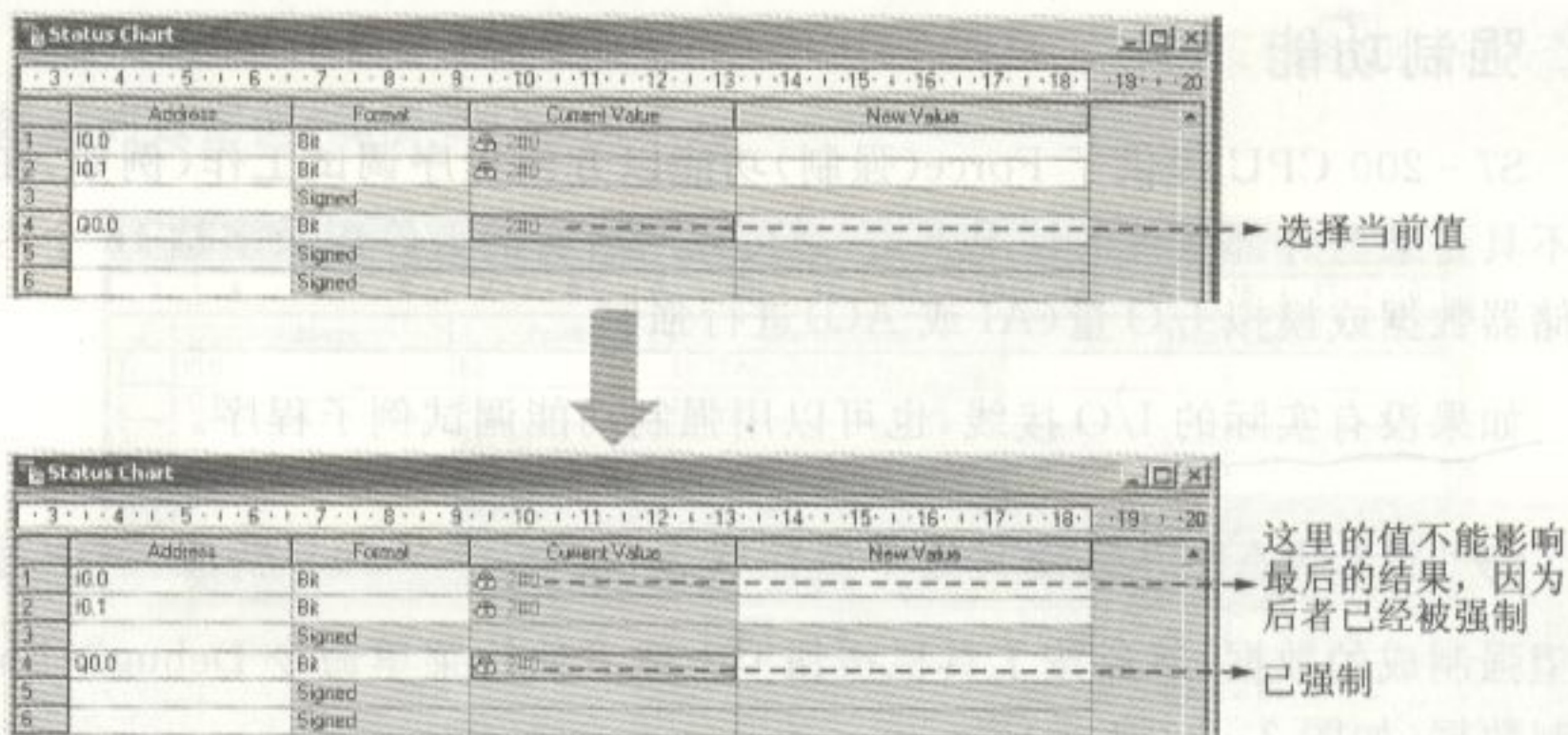


图 2-88 使用强制命令



S7-200 CPU 会在执行程序、更新 I/O 状态或通讯处理中改变已强制数据的值，但随后 CPU 会使用强制值覆盖原有的值。强制值具有高优先级。

使用工具栏按钮 ，或菜单命令 Debug>Unforce 解除强制；使用工具栏按钮  或菜单命令 Debug>Unforce All 解除所有的强制。

写入数据

S7-200 CPU 还提供了写入数据的功能以便于程序调试。



 在状态图表格(如图 2-89 所示)中输入 Q0.0 的新值“1”。



图 2-89 状态图表中 Q0.0 写入新值



☞ 输入新值后,按工具栏按钮,或使用菜单命令 Debug>Write All 写入数据(如图 2-90 所示)。

Status Chart																				
		3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
	Address	Format		Current Value		New Value														
1	I0.0	Bit		280																
2	I0.1	Bit		281																
3		Signed																		
4	Q0.0	Bit		281																
5		Signed																		
6		Signed																		

图 2-90 写入数据

☞ 写入数据命令不具有强制功能那样的优先级别。如果在程序中对相应的数据进行操作,写入的数据值可能改变。如果对 I0.0 和 I0.1 使用写入命令,或者逻辑运算的结果与写入值有抵触,写入的数值都不起作用。

☞ 应用写入命令可以同时输入几个数据值。

☞ 要在 S7-200 CPU 停止时能够强制或写入输出变量值,需要在菜单 Debug 中选择 Write-Force Outputs in STOP(STOP 模式下写强制输出值)(如图 2-91 所示)。

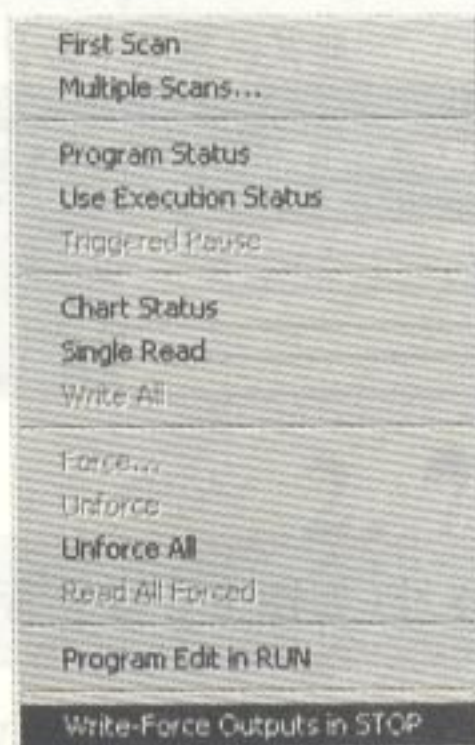



图 2-91 STOP 模式下写强制输出值



2.6 符号表

变量数目较多不便于编辑和调试程序。Step7 - Micro/WIN32 允许使用符号表为每个变量取一个惟一的符号名称。

☞ 在浏览条上 View(视图)中用鼠标单击  图标,或使用菜单命令 View>Component>Symbol Table,打开符号表。在 Symbol 列中输入符号名,在 Address 列中输入地址(如图 2-92 所示)。






Symbol Table																					
		3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20		
			Symbol			Address			Comment												
1			Lamp_ON			I0.0															
2			Lamp_OFF			I0.1															
3																					
4			Lamp			Q0.0															
5																					

图 2-92 符号表

☞ 执行一次 Compile(编译)指令,就可以使符号表应用于程序中(如图 2-93 所示)。




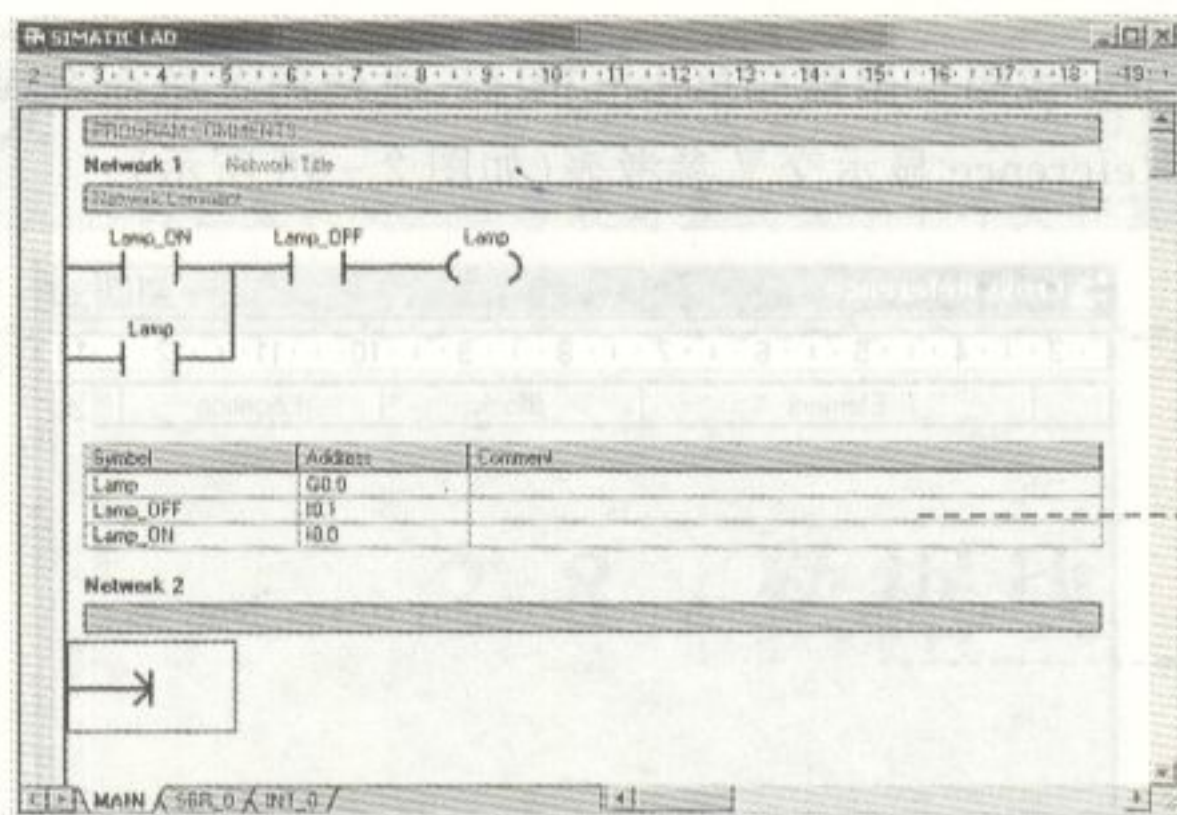
Symbol Table																				
3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20			
			Symbol		Address															
1			Lamp_ON		I0.0															
2			Lamp_OFF		I0.1															
3																				
4			Lamp		Q0.0															
5																				

图 2-93 编译后的符号表

☞ 打开程序块查看,变量都已经改为符号寻址(如图 2-94 所示)。

使用工具栏上的  按钮,或使用菜单命令 View>Symbol Information



符号信息表

图 2-94 使用符号寻址的程序

Table 切换符号信息表的显示。使用菜单命令 View>Symbolic Addressing 在符号寻址和绝对寻址之间切换。

再看一个符号表例子,如图 2-95 所示。

	Symbol	Address	Comment
1	Lamp_ON	I0.0	
2	Lamp_OFF	I0.1	
3			
4	Lamp	Q0.0	
5			
6	Unused	---	---
7		I0.0	

此符号表示变量与其他变量的地址重叠,实际上I0.0和I0.1是I0.0的一部分


未定义的符号

图 2-95 符号表举例

2.7 交叉参考

交叉参考表能显示程序中使用的元件详细信息。



在浏览条 View 视图下用鼠标单击  图标, 或使用菜单命令 View > Component > Cross Reference 显示交叉参考表(如图 2-96 所示)。

如程序未经编译,
显示提示消息

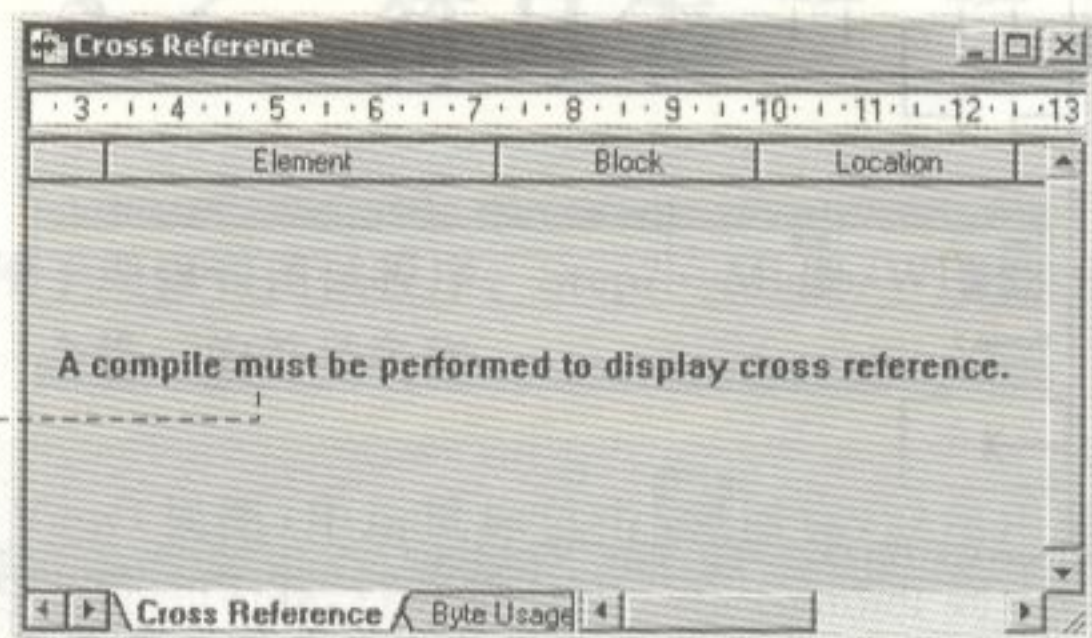


图 2-96 程序未经编译的交叉参考表

编译后的显示如图 2-97 所示。

查看“位”
使用状态

查看“字节”
使用状态

	Element	Block	Location	Context
1	Lamp_ON	MAIN (OB1)	Network 1	11
2	Lamp_OFF	MAIN (OB1)	Network 1	11
3	Lamp	MAIN (OB1)	Network 1	11
4	Lamp	MAIN (OB1)	Network 1	11

图 2-97 交叉参考表

同样, 可以用菜单命令 View > Symbolic Addressing 在符号寻址方式和绝对寻址方式之间切换(如图 2-98 所示)。

	Element	Block	Location	Context
1	R0	MAIN (OB1)	Network 1	11
2	I0.1	MAIN (OB1)	Network 1	11
3	Q0.0	MAIN (OB1)	Network 1	11
4	Q0.0	MAIN (OB1)	Network 1	11


图 2-98 以绝对寻址方式显示的交叉参考表



在交叉参考表中,用鼠标双击某一行可以立即跳转到引用相应元件的位置。交叉参考表对查找程序中冲突和重叠的数据地址十分有用。

2.8 数据块

数据块用于为 V 存储器区指定初始值。可以使用不同的长度(字节、字或双字)在 V 存储器中保存不同格式的数据。

用鼠标单击浏览条 View 视图中的  图标,或使用菜单命令 View>Component>Data Block 打开数据块窗口。数据块窗口是一个文本编辑器,编辑时直接在窗口内输入地址和数据。下面是一个数据块的示例(如图 2-99 所示)。



“//”符号后是注释

三列之间用Tab(制表符)或空格分隔,换行后编辑器会自动格式化

图 2-99 数据块举例

下载后我们可以使用状态图观察 V 存储区,注意为变量选择的格式(如图 2-100 所示)。



	Address	Format	Current Value	New Value
1	VB0	Unsigned	255	
2	VW2	Signed	-255	
3	VD4	Floating Point	700.50	
4	VB8	Signed	10	
5	VW10	Hexadecimal	1680014	
6		Signed		
7	VD14	Unsigned	46819	
8		Signed		
9	VW20	Signed	-2	
10	VW22	Signed	-4	
11	VW24	Signed	-6	
12	VW26	Signed	-8	
13	VW28	Signed	-10	
14	VW30	Signed	-12	
15	VW32	Signed	-14	
16	VW34	Signed	-16	
17	VW36	Signed	-18	
18	VW38	Signed	-20	
19	VW40	Signed	-22	
20		Signed		
21	VW46	ASCII	Up	
22		Signed		
23	VW50	Unsigned	15075	

图 2-100 状态图



为同一个变量选择不同的监视格式,同样的数值可以有不同的显示结果。



数据块下载到 S7-200 CPU 的 EEPROM 内,所以永远不会丢失。

2.9 Tools(工具)

☞ 用鼠标单击 Step7 - Micro/WIN32 浏览条的 Tools 按钮显示 Tools (工具)视图(如图 2-101 所示)。

☞ 也可以用菜单命令 Tools 找到向导和工具入口(如图 2-102 所示)。

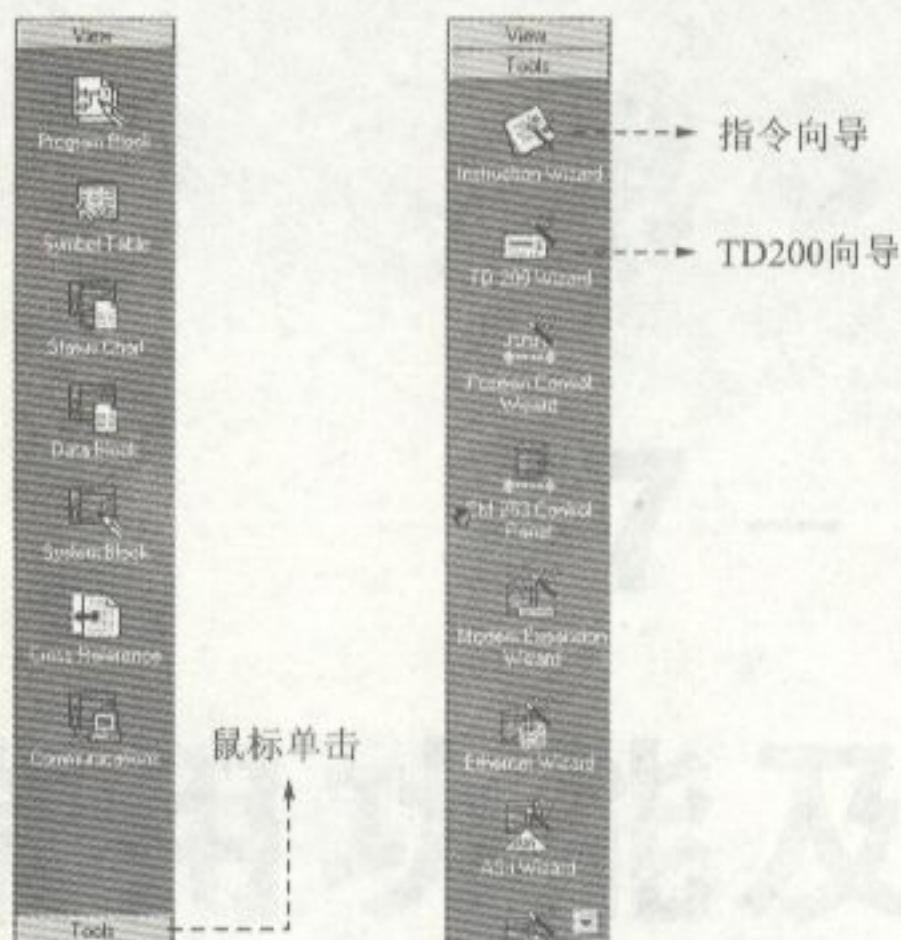


图 2-101 Tools 视图



图 2-102 向导与工具入口

☞ Instruction Wizard(指令向导)开始界面(如图 2-103 所示)。

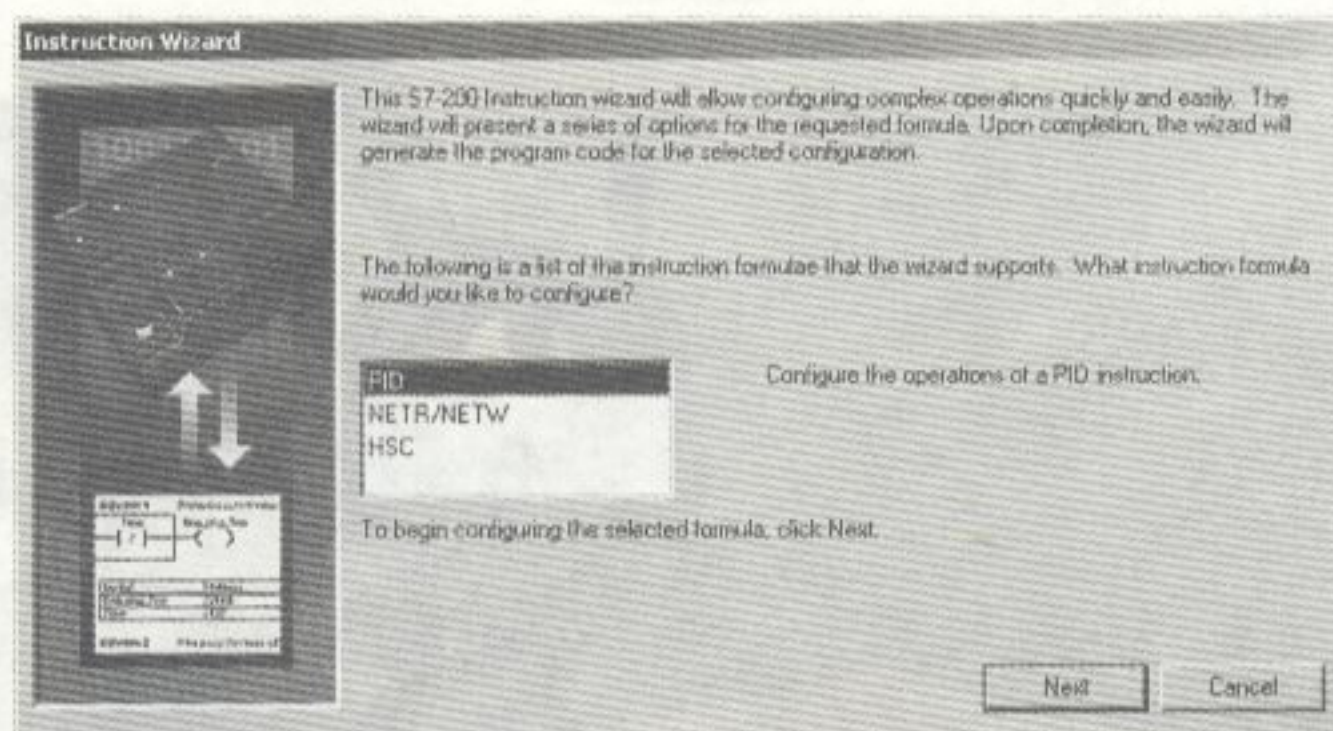
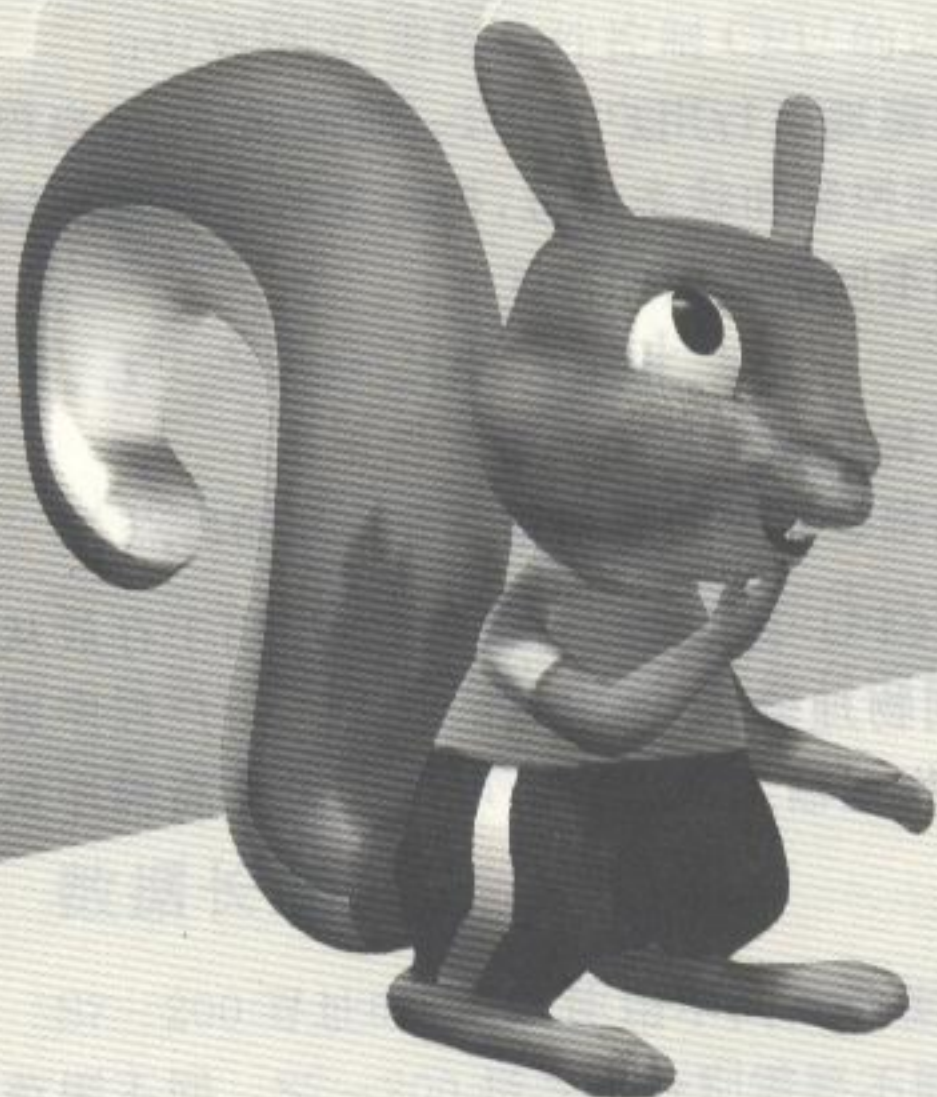


图 2-103 Instruction Wizard 开始界面

第 3 章

S7 — 200

常用功能及编程简介





3.1 S7-200 寻址与基本指令

3.1.1 S7-200 如何工作

S7-200 CPU 的基本功能就是监视现场的输入信号,根据用户的控制逻辑进行控制运算,输出信号去控制现场设备的运行。

在 S7-200 系统中,控制逻辑由用户编程实现。用户程序要下载到 S7-200 CPU 中执行。S7-200 CPU 按照循环扫描的方式,完成包括执行用户程序在内的各项任务。

S7-200 CPU 周而复始地执行一系列任务。任务执行一次称为一个扫描周期。在一个扫描周期内,CPU 执行如下操作:

- **读输入**: S7-200 CPU 读取物理输入点上的状态并复制到输入过程映像寄存器中;
 - **执行用户控制逻辑**: 从头至尾地执行用户程序,一般情况下,用户程序从输入映像寄存器获得外部控制和状态信号,把运算的结果写到输出映像寄存器中,或者存入到不同的数据保存区中;
 - **处理通讯任务**;
 - **执行自诊断**: S7-200 CPU 检查整个系统是否工作正常;
 - **写输出**: 复制输出过程映像寄存器中的数据状态到物理输出点。
- ★ **过程映像寄存器**: S7-200 CPU 中的特殊存储区,专门用于存放从物理输入/输出点读取或写到物理输入/输出点的状态。用户程序通过过程映像寄存器访问实际物理输入、输出点,可以大大提高程序执行效率。



为保证某些任务对执行时间的要求,S7-200 也允许用户程序直接访问物理输入、输出点;S7-200 也使用硬件执行诸如高速脉冲、通讯等任务,用户程序通过特殊寄存器控制这些硬件的工作。

3.1.2 S7-200 CPU 的工作模式

S7-200 有两种操作模式:停止模式和运行模式。CPU 前面板上的 LED 状态显示了当前的操作模式。在停止模式下,S7-200 不执行程序,你可以下载程序、数据和 CPU 系统设置。在运行模式下,S7-200 运行程序。

要改变 S7-200 CPU 的操作模式,有以下几种方法:

- 使用 S7-200 CPU 上的模式开关:开关拨到 RUN 时,CPU 运行;开关拨到 STOP 时,CPU 停止;开关拨到 TERM 时,不改变当前操作模式。如果需要 CPU 在上电时自动运行,模式开关必须在 RUN 位置。
- CPU 上的模式开关在 RUN 或 TERM 位置时,可以使用 Step7 - Micro/WIN32 编程软件控制 CPU 的运行和停止。
- 在程序中插入 STOP 指令,可以在条件满足时将 CPU 设置为停止模式。

3.1.3 S7-200 寻址

S7-200 CPU 将信息存储在不同的存储器单元,每个单元都有惟一的地址。S7-200 CPU 使用数据地址访问所有的数据,称为寻址。输入/输出点,中间运算数据等各种数据类型具有各自的地址定义方式。S7-200 的大部分指令都需要指定数据地址。

数据长度

S7-200 寻址时,可以使用不同的数据长度。不同的数据长度表示的数值范围不同。S7-200 指令也分别需要不同的数据长度。



数据长度和数据范围(如表 3-1 所列)

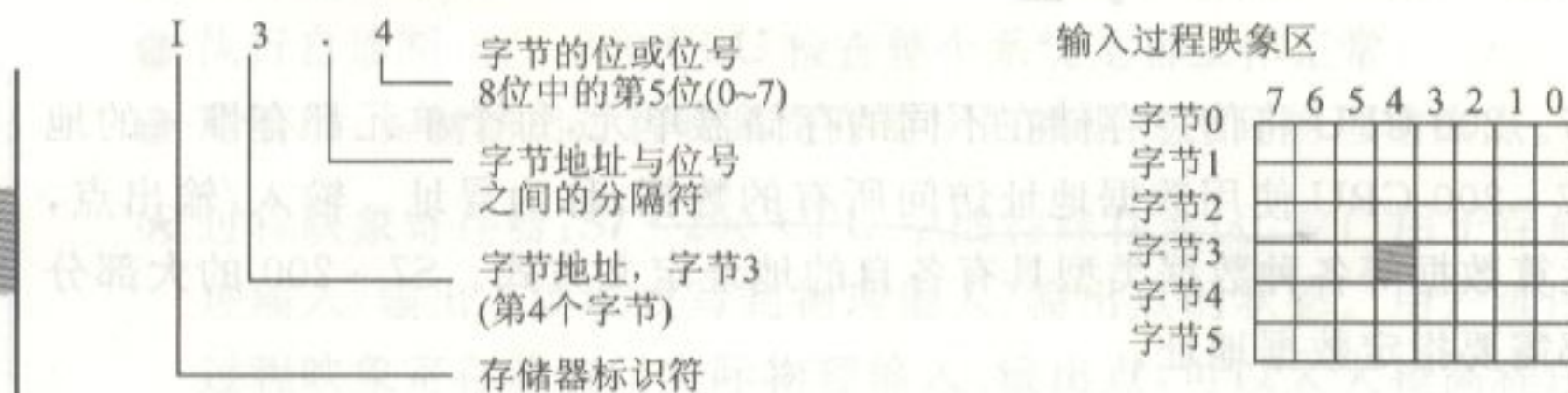
表 3-1 数据长度和数值范围

数据长度	字节/B	字/W	双字/D
无符号整数	0~255(十进制) 0~FF(十六进制)	0~65 535(十进制) 0~FFFF(十六进制)	0~4 294 967 295(十进制) 0~FFFF FFFF(十六进制)
符号整数	-128~+127(十进制) 80~7F(十六进制)	-32 768~ +32 767(十进制) 8000~7FFF (十六进制)	-2 147 483 648~ +2 147 483 647(十进制) 8000 0000~7FFF FFFF (十六进制)
实数(单精度) IEEE 32 位浮 点数			+1.175 495E-38~ +3.402 823E+38(正数) -1.175 495E-38~ -3.402 823E+38(负数)(十进制)

在 S7-200 系统中,可以按位、字节、字和双字对存储单元寻址。

寻址时,数据地址以代表存储区类型的字母开始,随后是表示数据长度的标记,然后是存储单元编号;对于二进制位寻址,还需要在一个小数点分隔符后指定位编号。

位寻址的举例(如图 3-1 所示)



注:I表示存储器是输入过程映象区

图 3-1 位寻址举例



字节寻址举例(如图 3-2 所示)

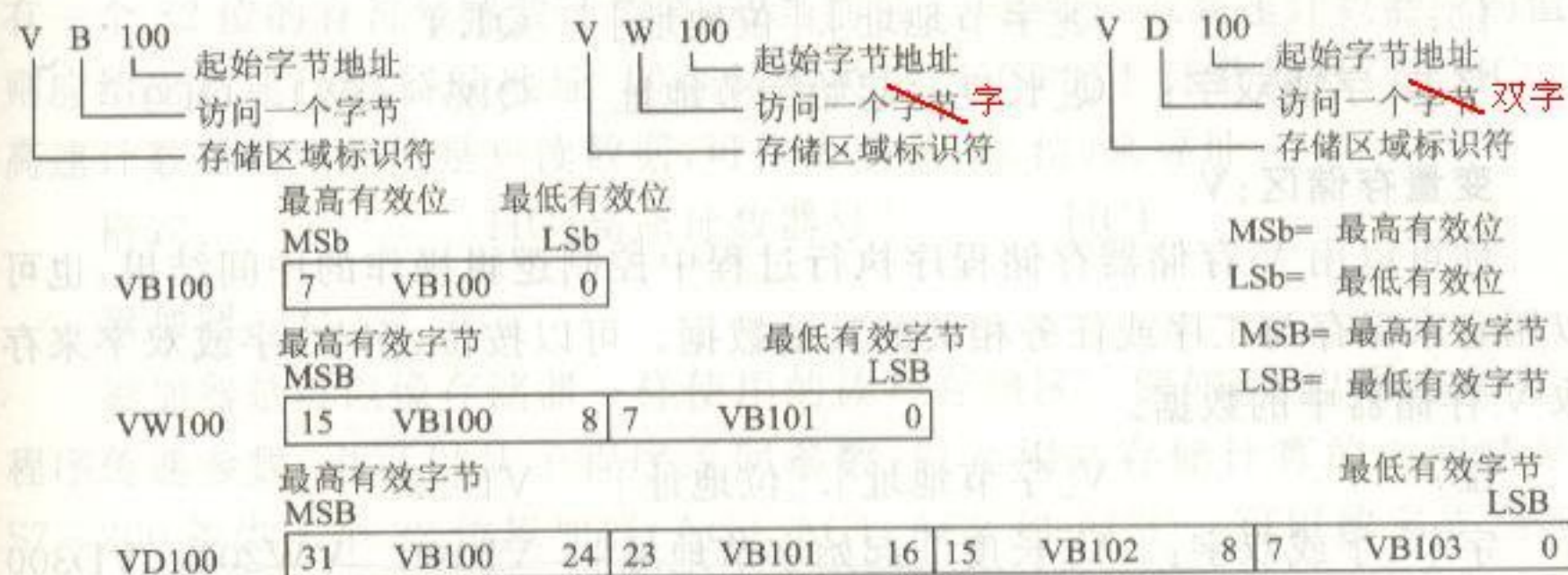


图 3-2 字节寻址举例

✌ 可以看出, VW100 包括 VB100 和 VB101; VD100 包含 VW100 和 VW102, 即 VB100、VB101、VB102、VB103 四个字节。值得注意的是, 地址是互相交叠的。

✌ 当涉及到多字节组合寻址时, S7-200 遵循“高地址、低字节”的规律。如果将 16#AB(十六进制数值)送入 VB100, 16#CD 送入 VB101, 那 VW100 的值将是 16#ABCD。即 VB101 作为高地址字节, 保存数据的低字节部分。

各数据存储区寻址

输入过程映象寄存器:I

在每次扫描周期的开始, CPU 对物理输入总进行采样, 并将采样值写入输入过程映象寄存器中。可以按位、字节、字或双字来存取输入过程映象寄存器中的数据。

位: I[字节地址].[位地址] I0.1

字节、字或双字: I[长度][起始字节地址] IB4 IW1 ID0

输出过程映象寄存器:Q

在每次扫描周期的结尾, CPU 将输出过程映象寄存器中的数值复制到物



理输出点上。可以按位、字节、字或双字来存取输出过程映象寄存器中的数据。

位: Q[字节地址].[位地址] Q1.1

字节、字或双字: Q[长度][起始字节地址] QB5 QW1 QD0

变量存储区:V

你可以用 V 存储器存储程序执行过程中控制逻辑操作的中间结果,也可以用它来保存与工序或任务相关的其他数据。可以按位、字节、字或双字来存取 V 存储器中的数据。

位: V[字节地址].[位地址] V10.2

字节、字或双字: V[长度][起始字节地址] VB100 VW200 VD300

位存储区:M

可以用位存储区作为控制继电器来存储中间操作状态和控制信息。可以按位、字节、字或双字来存取位存储区中的数据。

位: M[字节地址].[位地址] M26.7

字节、字或双字: M[长度][起始字节地址] MB0 MW13 MD20

定时器存储区:T

S7-200 CPU 中,定时器可用于时间累计。定时器寻址有两种形式:

- 当前值:16 位有符号整数,存储定时器所累计的时间。
- 定时器位:按照当前值和预置值的比较结果置位或者复位。

两种寻址使用同样的格式,用定时器地址(T+定时器号,如 T33)来存取这两种形式的定时器数据。究竟使用哪种形式取决于所使用的指令。

计数器存储区:C

在 S7-200 CPU 中,计数器可以用于累计其输入端脉冲电平由低到高的次数。计数器有两种寻址形式:

- 当前值:16 位有符号整数,存储累计值。
- 计数器位:按照当前值和预置值的比较结果来置位或者复位。

可以用计数器地址(C+计数器号,如 C0)来存取这两种形式的计数器数据。究竟使用哪种形式取决于所使用的指令。



高速计数器:HC

高速计数器对高速事件计数,它独立于 CPU 的扫描周期。高速计数器有一个 32 位的有符号整数计数值(或当前值)。若要存取高速计数器中的值,则应给出高速计数器的地址,即存储器类型(HC)加上计数器号(如 HC0)。高速计数器的当前值是只读数据,可作为双字(32 位)来寻址。

格式: HC[高速计数器号] HC1

累加器:AC

累加器是可以像存储器一样使用的读写存储区。例如,可以用它来向子程序传递参数,也可以从子程序返回参数,以及用来存储计算的中间结果。S7-200 提供 4 个 32 位累加器 AC0、AC1、AC2 和 AC3。可以按字节、字或双字的形式来存取累加器中的数值。被操作的数据长度取决于访问累加器时所使用的指令。

特殊存储器:SM “S7-200系统手册”中有SM特殊寄存器各位的介绍

SM 位为 CPU 与用户程序之间传递信息提供了一种手段。可以用这些位选择和控制 S7-200 CPU 的一些特殊功能。你可以按位、字节、字或者双字的形式来存取。

位: SM[字节地址].[位地址] SM0.1

字节、字或者双字: SM[长度][起始字节地址] SMB86

模拟量输入:AI

S7-200 将模拟量值(如温度或电压)转换成 1 个字长(16 位)的数据。可以用区域标识符(AI)、数据长度(W)及字节的起始地址来存取这些值。因为模拟值输入为 1 个字长,且从偶数位字节(如 0,2,4)开始,所以必须用偶数字节地址(如 AIW0, AIW2, AIW4)来存取这些值。模拟量输入值为只读数据。模拟量转换的实际精度是 12 位。

格式: AIW[起始字节地址] AIW4

模拟量输出:AQ

S7-200 把 1 个字长(16 位)数字值按比例转换为电流或电压。可以用区域标识符(AQ)、数据长度(W)及字节的起始地址来改变这些值。因为模拟量为一个字长,且从偶数字节(如 0,2,4)开始,所以必须用偶数字节地址(如 AQW0, AQW2, AQW4)来改变这些值。模拟量输出值为只写数据。模拟量



转换的实际精度是 12 位。

格式: AQW[起始字节地址] AQW4

常数(如表 3-2 所列)

表 3-2 常 数

数 制	格 式	举 例
十进制	[十进制值]	20 047
十六进制	16#[十六进制值]	16#4E4F
二进制	2#[二进制数]	2#1010_0101_1010_0101
ASCII	'[ASCII 码文本]'	'Text goes between single quotes.'
实 数	ANSI/IEEE 754—1985	+1.175 495E-38(正数)-1.175 495E-38(负数)

3.1.4 S7-200 的集成 I/O 和扩展 I/O

CPU 提供的集成 I/O 具有固定的 I/O 地址。可以将扩展模块连接到 CPU 的右侧来增加 I/O 点,形成 I/O 链。对于同种类型的输入输出模块而言,模块的 I/O 地址取决于 I/O 类型和模块在 I/O 链中的位置。举例来说,输出模块不会影响输入模块上的点地址,反之亦然。类似地,模拟量模块不会影响数字量模块的地址,反之亦然。



CPU 和扩展模块的数字量地址总是以 8 位(1 个字节)递增。如果 CPU 或模块在为物理 I/O 点分配地址时未用完一个字节,则那些未用的位不能分配给 I/O 链中的后续模块。对于输入模块,这些字节中保留的未用位会在每个输入刷新周期中被清零。



每个模拟量扩展模块的输入点地址总是以 4 个通道(4 个 16 位的字)递增;输出点地址总是以 2 个通道(2 个 16 位的字)递增。如果模块没有占用完输入/输出通道,那么,这些通道地址也不能够分配给后续模拟量模块。



I/O 地址分配举例(如图 3-3 所示)

CPU 224		4 In / 4 Out	8 In	4 Analog In 1 Analog Out	8 Out	4 Analog In 1 Analog Out
IO.0	Q0.0	Module 0	Module 1	Module 2	Module 3	Module 4
IO.1	Q0.1	I2.0	I3.0	AIW0	Q3.0	AIW8
IO.2	Q0.2	I2.1	I3.1	AIW2	Q3.1	AIW10
IO.3	Q0.3	I2.2	I3.2	AIW4	Q3.2	AIW12
IO.4	Q0.4	I2.3	I3.3	AIW6	Q3.3	AIW14
IO.5	Q0.5	I2.4	I3.4		Q3.4	
IO.6	Q0.6	I2.5	I3.5		Q3.5	
IO.7	Q0.7	I2.6	I3.6		Q3.6	
IO.8	Q1.0	I2.7	I3.7		Q3.7	
IO.9	Q1.1	Expansion I/O				
IO.10	Q1.2					
IO.11	Q1.3					
IO.12	Q1.4					
IO.13	Q1.5					
IO.14	Q1.6					
IO.15	Q1.7					
IO.16	Q1.8					
IO.17	Q1.9					
Local I/O						

图 3-3 I/O 地址分配举例

✌ 图 3-3 中灰色为未分配而不能使用的地址。



在编程计算机和 S7-200 CPU 联机状态下,使用 Step7 - Micro/WIN32 的菜单命令“PLC>Information”,可以方便地查看 CPU 和扩展模块的地址分配。

3.1.5 基本指令

位逻辑指令

位逻辑指令的基础是触点和线圈。触点对二进制位的状态测试,测试的结果用于进行位逻辑运算;线圈是用来改变二进制位的状态,其状态根据它前面的逻辑运算结果而定。

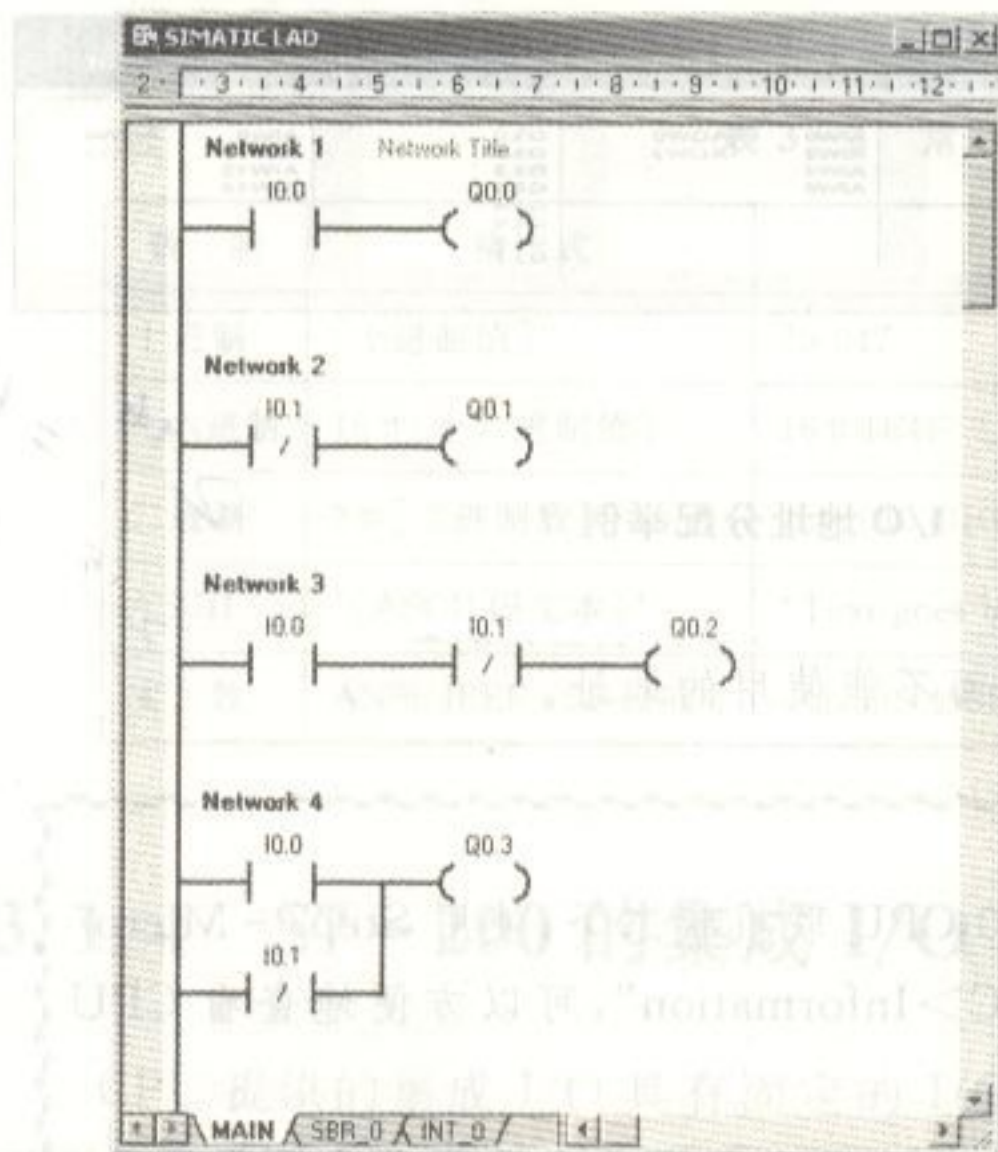


一个二进制位,既可以在程序中作为触点,也可以作为线圈。线圈可以作为触点在程序中被多次引用;如果同一地址的线圈在不止一个程序段中出现,其状态以最后一次运算的结果为准。

位逻辑运算的基本关系是“与”和“或”。



程序举例(如图 3-4 所示)



常开触点:
I0.0为“1”时, Q0.0为“1”

常闭触点:
I0.1为“0”时, Q0.1为“1”

与运算:
I0.0为“1”且I0.1为“0”时,
Q0.2为“1”

或运算:
I0.0为“1”或I0.1为“0”时,
Q0.3为“1”

图 3-4 程序举例

取反指令(**|NOT|**)的作用是改变它前面逻辑运算结果的状态,把“1”变成“0”,或把“0”变成“1”。

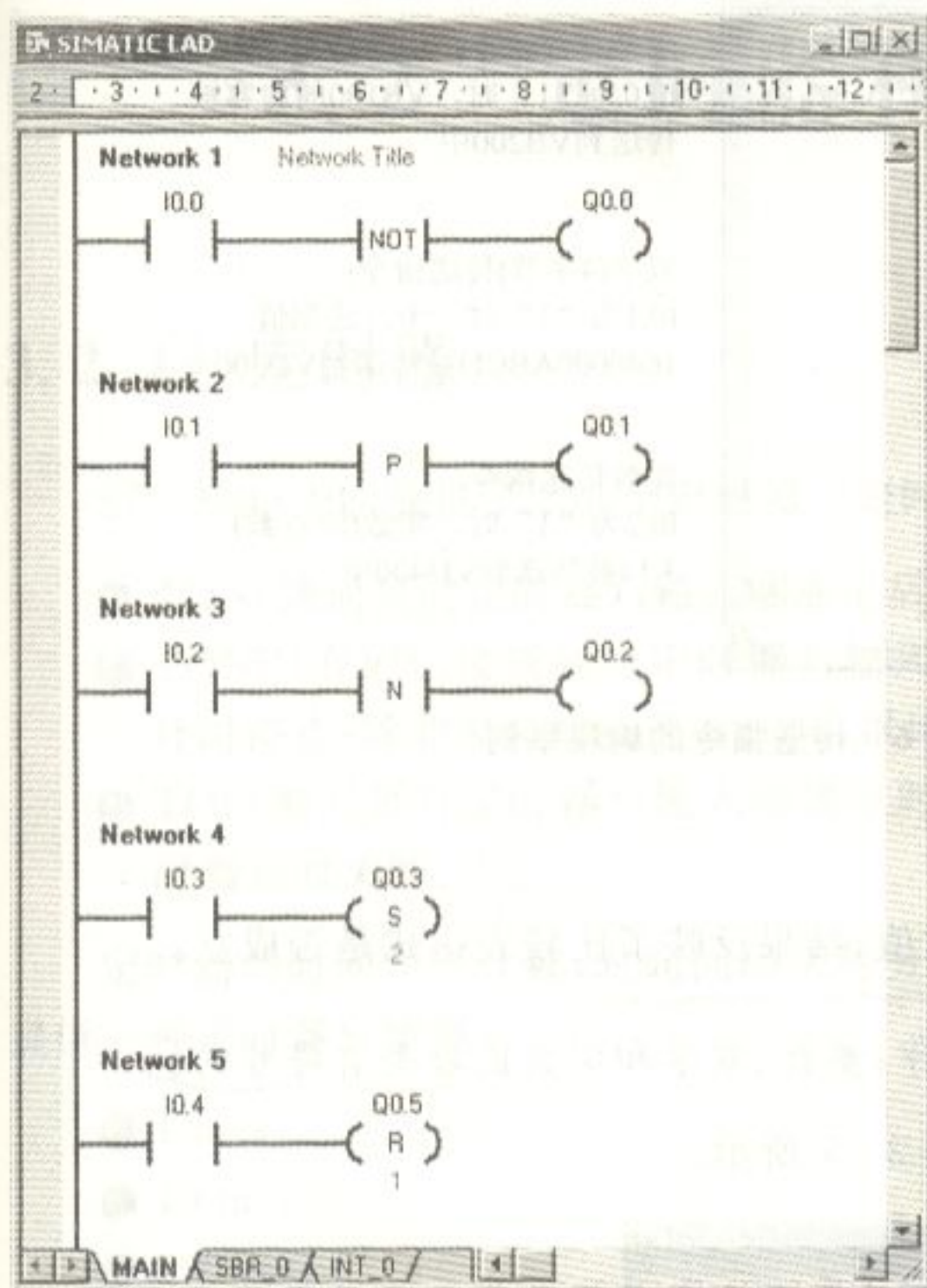
正跳变(**|P|**)指令检测它前面的逻辑状态。如果上个程序扫描周期是“0”,本周期是“1”,则它后面的逻辑状态在本周期的剩余扫描时间内为“1”。该指令仅在一个周期内有效。

负跳变(**|N|**)指令指令检测它前面的逻辑状态,如果上个程序扫描周期是“1”,本周期是“0”,则它后面的逻辑状态在本周期的剩余扫描时间内为“1”。该指令仅在一个周期内有效。





五种指令的编程举例如图 3-5 所示。



取反指令:

I0.0为“0”时, Q0.0为“1”

正跳变指令:

I0.1从“0”变为“1”时, Q0.1为“1”一个周期

负跳变指令:

I0.2从“1”变为“0”时, Q0.2为“1”一个周期

置位指令:

I0.3为“1”时, 从Q0.3开始的2个位(即Q0.3, Q0.4)被置为“1”

复位指令:

I0.4为“1”时, 从Q0.5开始的1位被置为“0”

图 3-5 编程举例

传送指令

数据传送指令在不改变原值的情况下, 将 IN(输入端)的值传送到 OUT(输出端)。

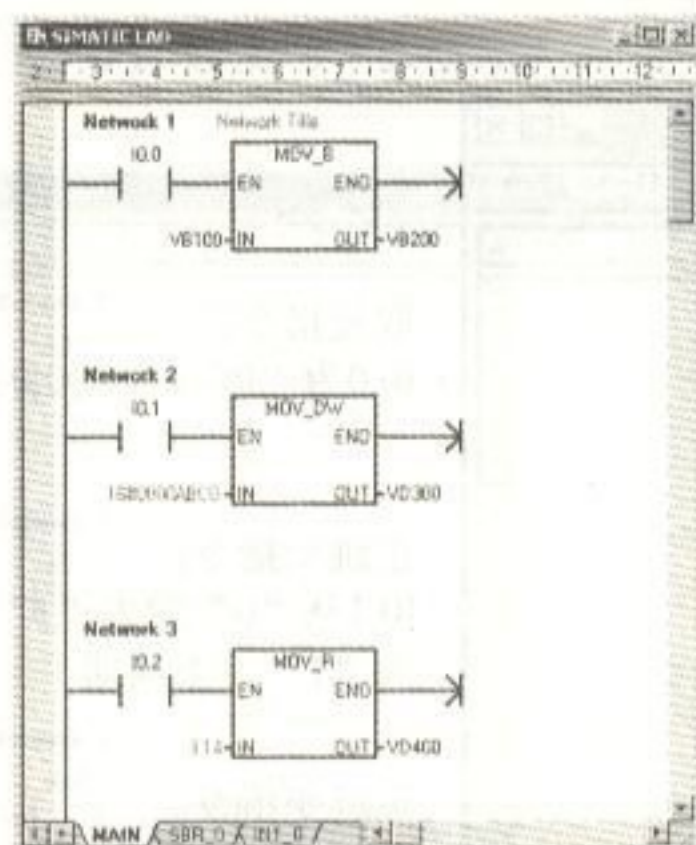


传送指令的输入/输出数据长度应当一致。

传送指令的编程举例如图 3-6 所示。



方



字节传送指令:

I0.0为“1”时, VB100的内容被传送到VB200中

双字(4字节)传送指令:

I0.1为“1”时, 十六进制值16#0000ABCD被传送到VD300中

实数传送指令:

I0.2为“1”时, 实数(浮点数)3.14被传送到VD400中

图 3-6 传送指令的编程举例

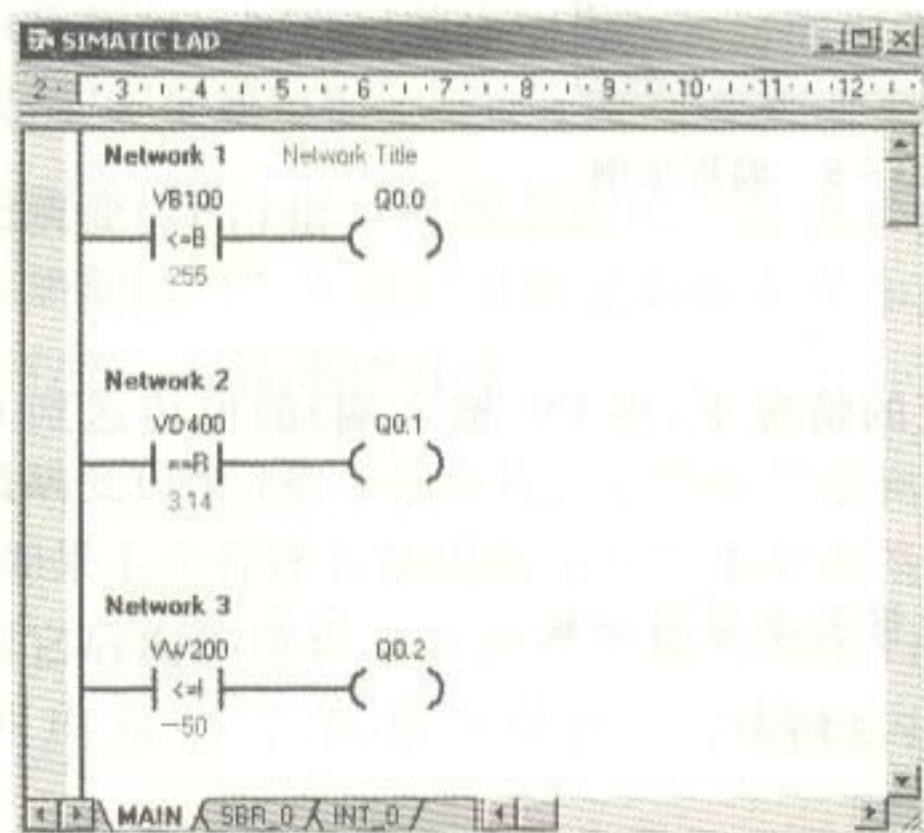
比较指令

比较指令用来比较两个数值, 结果反映了比较表达式是否成立。



字节比较是无符号操作; 整数、双字和实数比较是有符号的。

比较指令的编程举例如图 3-7 所示。



字节比较指令:

VB100的内容小于等于255时, Q0.0为“1”

实数比较指令:

VD400的内容等于3.14时, Q0.1为“1”

整数(字)比较指令:

VW200的内容小于-50时, Q0.2为“1”

图 3-7 比较指令的编程举例



3.2 定时器和计数器

3.2.1 定时器

S7-200 CPU 提供了 256 个定时器。定时器分为三种类型:

- **TON(接通延时定时器)**: 输入端通电后, 定时器延时接通。
- **TONR(有记忆接通延时定时器)**: 输入端通电时定时器计时, 断开时计时停止; 除非复位端接通, 计时值累计。
- **TOF(断开延时定时器)**: 输入端通电时输出端接通, 输入端断开时定时器延时关断。

定时器对时间间隔计数, 时间间隔又称分辨率(或时基)。S7-200 CPU 提供三种定时器分辨率:

- 1 ms;
- 10 ms;
- 100 ms。

✌ 选择不同的定时器号就决定了定时器的类型和分辨率。建议在一个项目中, 一个定时器号只使用一次。

定时器规格如表 3-3 所列。

✌ 最长定时值 = 时基 × 最大定时计数值

定时器用使用 一个字长的有符号整数 对时基计数, 最大值为 32 767。

2¹⁵



表 3-3 定时器规格

定时器类型	分辨率/ms	最长定时值/s	定时器号
TONR	1	32.767	T0, T64
	10	327.67	T1~T4, T65~T68
	100	3 276.7	T5~T31, T69~T95
TON, TOF	1	32.767	T32, T96
	10	327.67	<u>T33~T36, T97~T100</u>
	100	3 276.7	T37~T63, T101~T255

定时器指令的梯形图格式如图 3-8 所示。

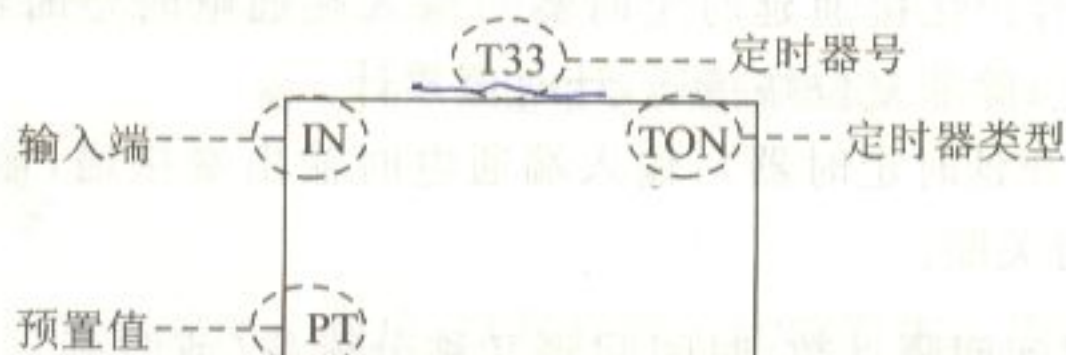


图 3-8

定时器指令接受操作数如表 3-4 所列。

表 3-4 定时器指令接受的操作数

输入/输出	数据类型	操作数
Txx	WORD	<u>常数(T0~T255), 指定定时器号</u>
IN	BOOL	<u>I、Q、V、M、SM、S、T、C、L、能流, 启动定时器</u>
PT	INT	IW、QW、VW、MW、SMW、T、C、LW、AC、AIW、* VD、* LD、* AC、常数, 规定预置值

定时器按表 3-5 所列的规律工作。



表 3-5 定时器工作规律

定时器数型	当前值 \geq 预设值时	IN(使能)输入接通	IN(使能)输入断开	上电周期/首次扫描
TON	定时器位 ON, 当前值连续计数到 32 767	当前值对时间间隔计数, 定时器工作	定时器位 OFF, 当前值=0	定时器位 OFF, 当前值=0
TONR	定时器位 ON, 当前值连续计数到 32 767	当前值对时间间隔计数, 定时器工作	<u>定时器位和当前值保持最后状态</u>	<u>定时器位 OFF, 当前值保持 1</u>
TOF	定时器位 OFF, <u>当前值 = 预设值</u> , 停止计数	<u>定时器位 ON, 当前值=0</u> , 定时器工作	发生 ON 到 OFF 的跳变之后, 定时器开始计数	定时器位 OFF, 当前值=0

注: 有记忆定时器的当前值可以在电源掉电时保持记忆

接通延时定时器指令程序举例如图 3-9 所示。

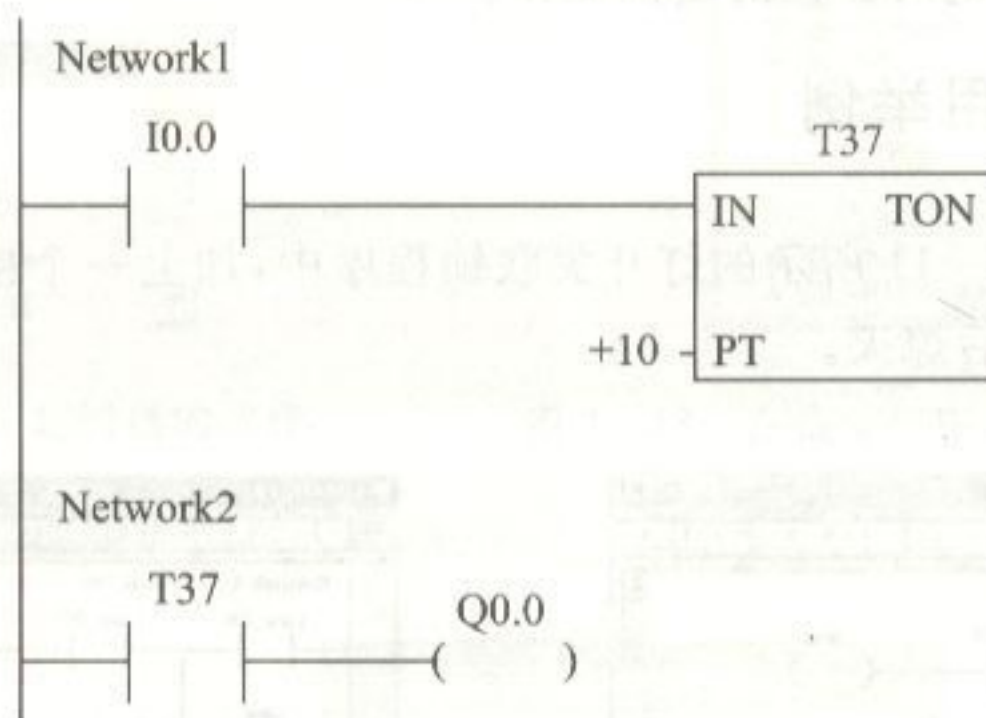


图 3-9 接通延时定时器指令程序举例

定时器 T37 时基为 100 ms, 预置值设定为 10, 实际延时时间为 1 s。这个程序的时序图如图 3-10 所示。



定时器有两种寻址类型: Word(字)和 Bit(位)。按字访问定时器标号时, 返回定时器当前值; 按位访问时, 返回定时器的位状态, 即是否到达定时值。

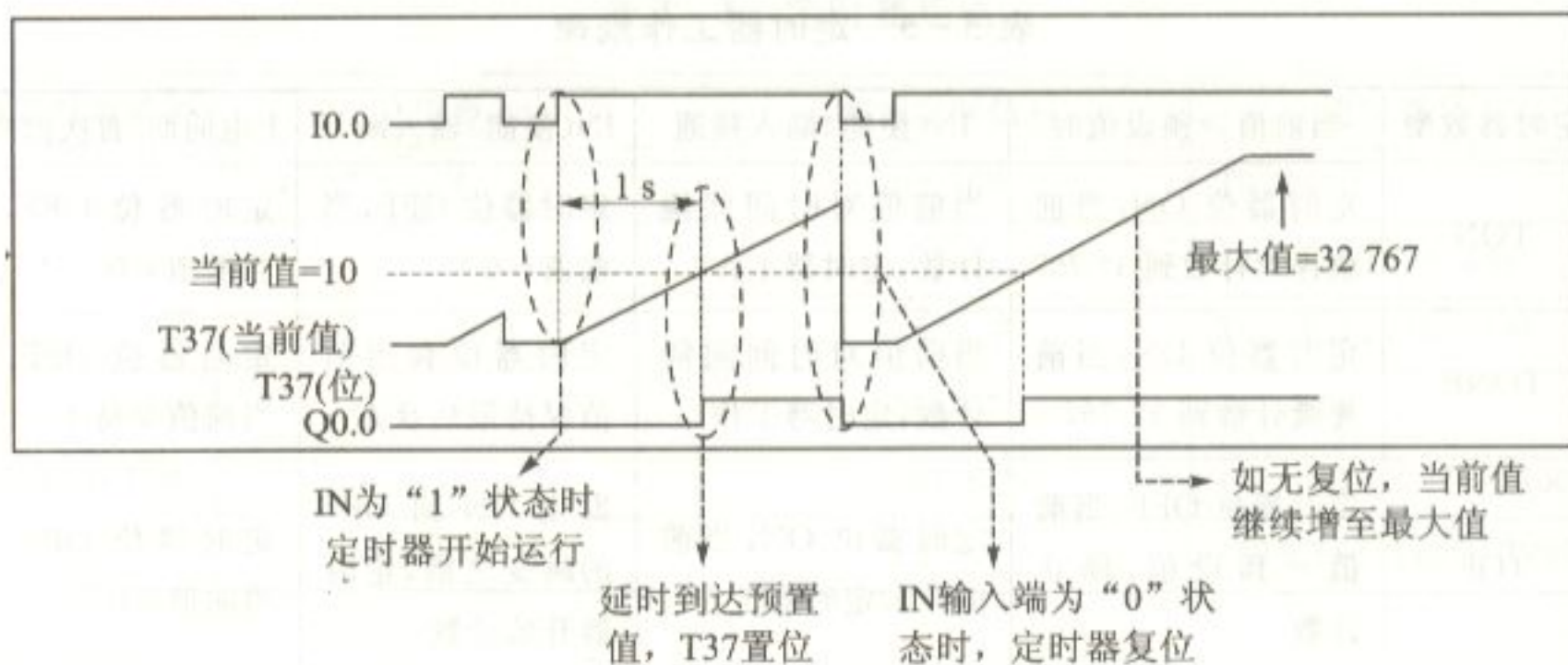


图 3-10 时序图

定时器号既可以参加字指令操作,例如定时值的数值计算;也可以参加位指令计算,例如对定时器位的复位操作。

定时器应用举例

在如图 3-11 所示的灯开关联锁程序中,加上一个接通延时定时器,控制灯点亮 10 s 后熄灭。

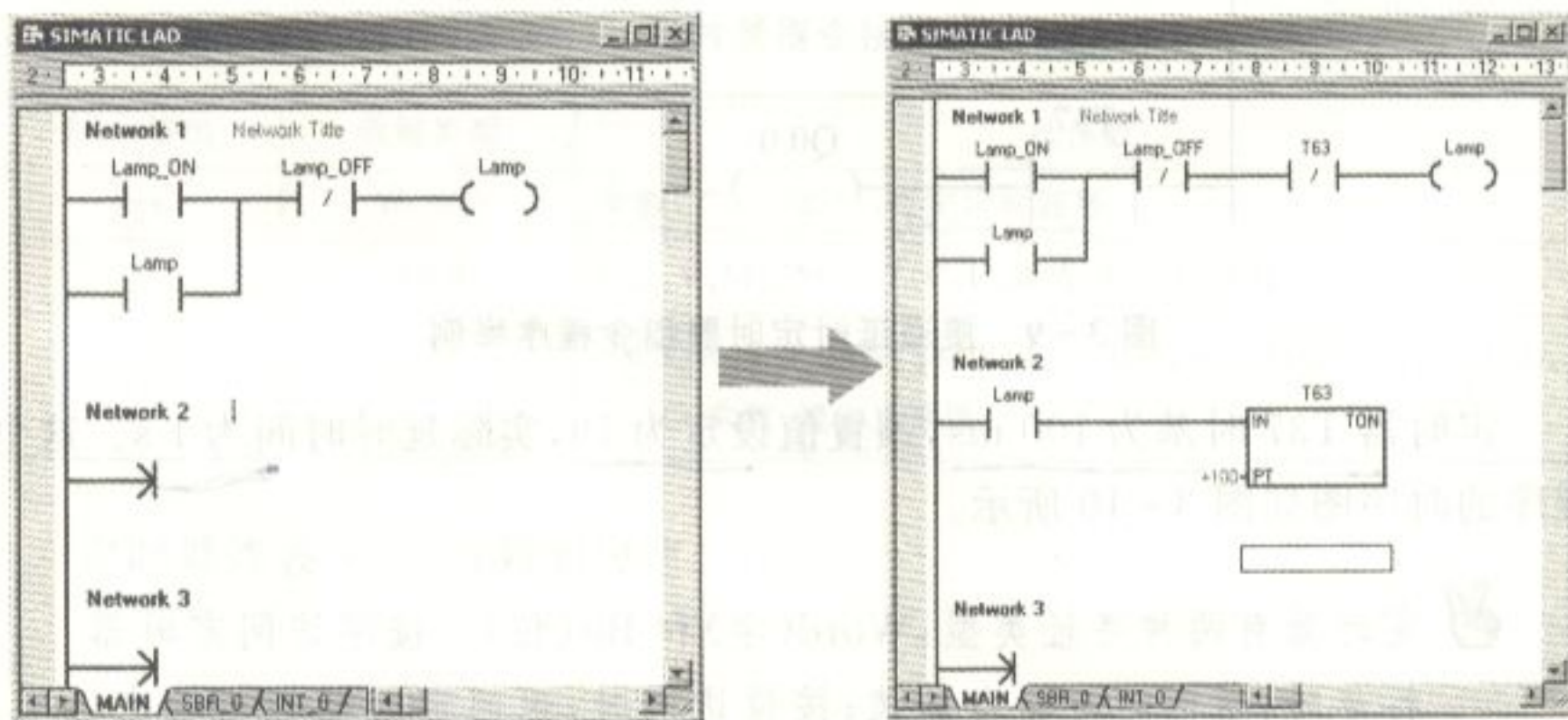



图 3-11 定时器应用举例



插入定时器

☞ 在需要插入定时器的位置上,使用工具栏按钮 ,或按 F6 弹出选择框,选择 TON 定时器(如图 3-12 所示)。

☞ 或者在指令树 Timer(定时器)分支中直接用鼠标选取(如图 3-13 所示)。

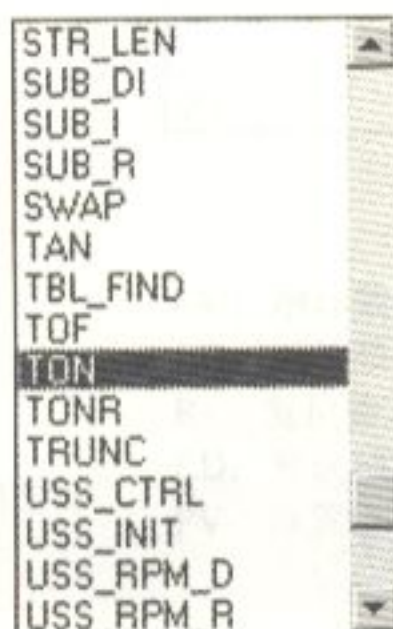


图 3-12 TON 定时器的选择

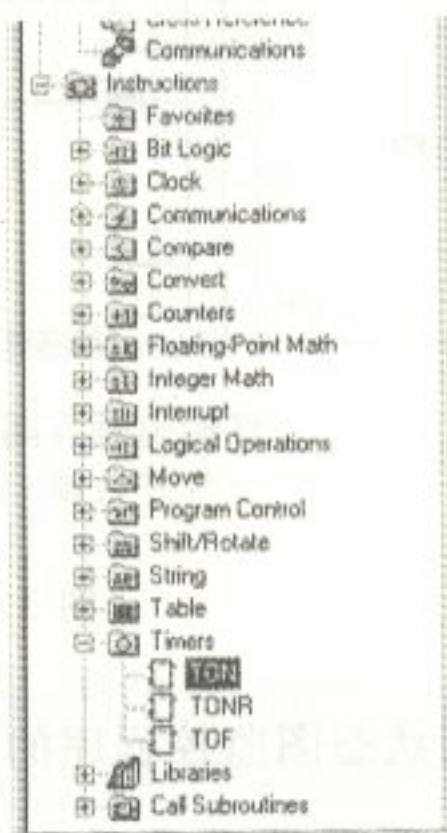


图 3-13 在指令树中选取 TON 定时器

☞ 完成后的程序如图 3-14 所示。

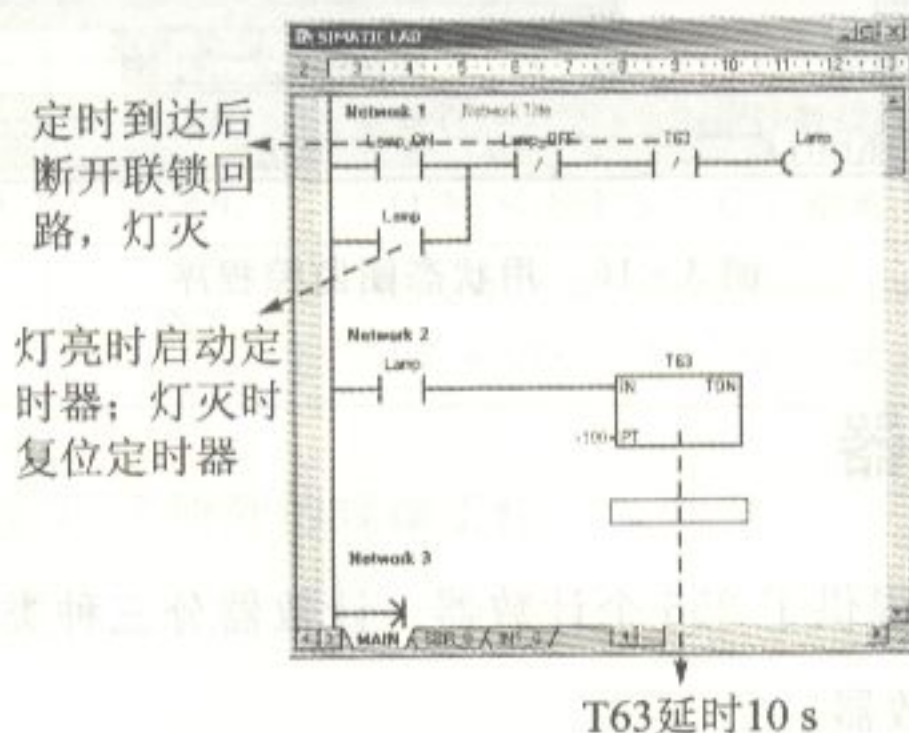
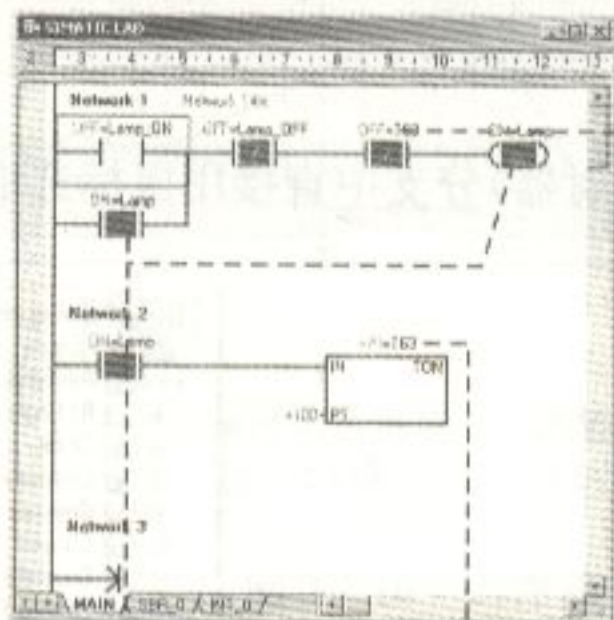


图 3-14 完成编程的窗口

编译下载程序到 S7-200 CPU, 监控程序的运行, 按 Lamp_On(开灯)按钮(如图 3-15 所示)。



按Bit(位)访问
定时器, 判断
定时是否到达

灯输出触点闭
合并自锁

定时器当前
计数值

图 3-15 监控程序运行状态

用状态图监控程序的运行(如图 3-16 所示)。

Status Chart				
	Address	Format	Current Value	New Value
1	Lamp_ON	Bit	281	
2	Lamp_OFF	Bit	280	
3		Signed		
4	Lamp	Bit	281	
5		Signed		
6	T63	Signed	22	
7	T63	Bit	280	

同一个定时器的
两种寻址,
数据格式不同

图 3-16 用状态图监控程序

3.2.2 计数器

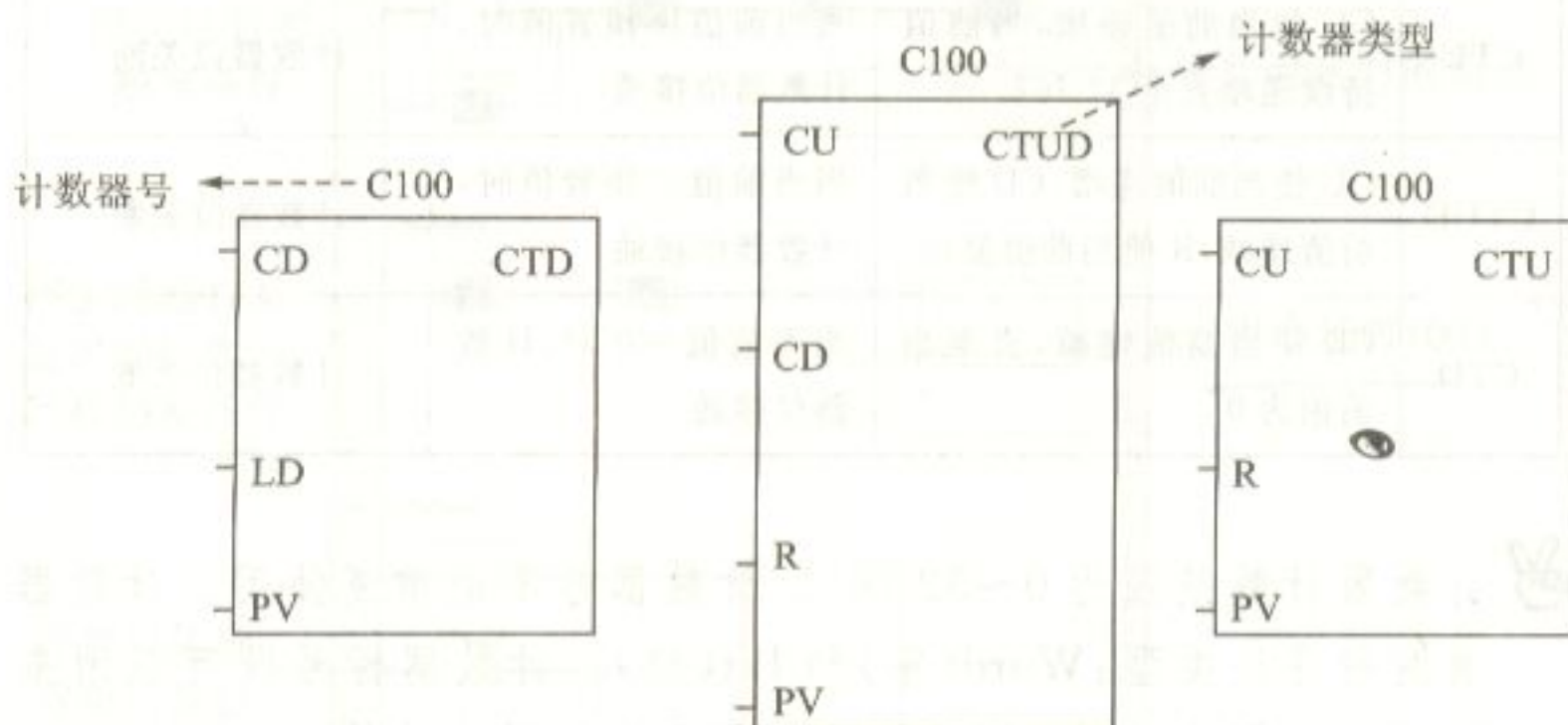
S7-200 CPU 提供了 256 个计数器。计数器分三种类型:

- CTU: 增计数器;
- CTD: 减计数器;



● CTUD:增/减计数器。

计数器指令的梯形图格式如图 3-17 所示。



CU: 增计数信号输入端;

CD: 减计数信号输入端;

R: 复位输入;

LD: 装载预置值;

PV: 预置值

图 3-17 计数器指令的梯形格式

计数器指令接受操作数如表 3-6 所列。

表 3-6 计数器指令接受操作数

输入/输出	数据类型	操作数
Cxx	WORD	常数(C0~C255),指定计数信号
CU,CD,LD,R	BOOL	I、Q、V、M、SM、S、T、C、L、能流
PV	INT	IW、QW、VW、MW、SMW、SW、LW、T、C、AC、AIW、* VD、* LD、* AC、常数,规定预置值

计数器按如表 3-7 所列的规律工作。

表 3-7 计数器工作规律

类 型	操 作	计数器位	上电周期/首次扫描
CTU	CU 使当前值递增, 当前值持续递增直至 32 767	当当前值 \geq 预置值时, 计数器位接通	计数器位关断
CTUD	CU 使当前值递增, CD 使当前值递减, R 使当前值复位	当当前值 \geq 预置值时, 计数器位接通	计数器位关断
CTD	CD 使当前值递减, 直至当前值为 0	当当前值 = 0 时, 计数器位接通	计数器位关断

✌ 计数器计数范围为 0~32 767。计数器号不能重复使用。计数器有两种寻址类型: Word(字)和 Bit(位)。计数器标号既可以用来访问计数器当前值, 也可以用来表示计数器位的状态。

增/减计数器指令举例如图 3-18 所示, 时序图如图 3-19 所示。

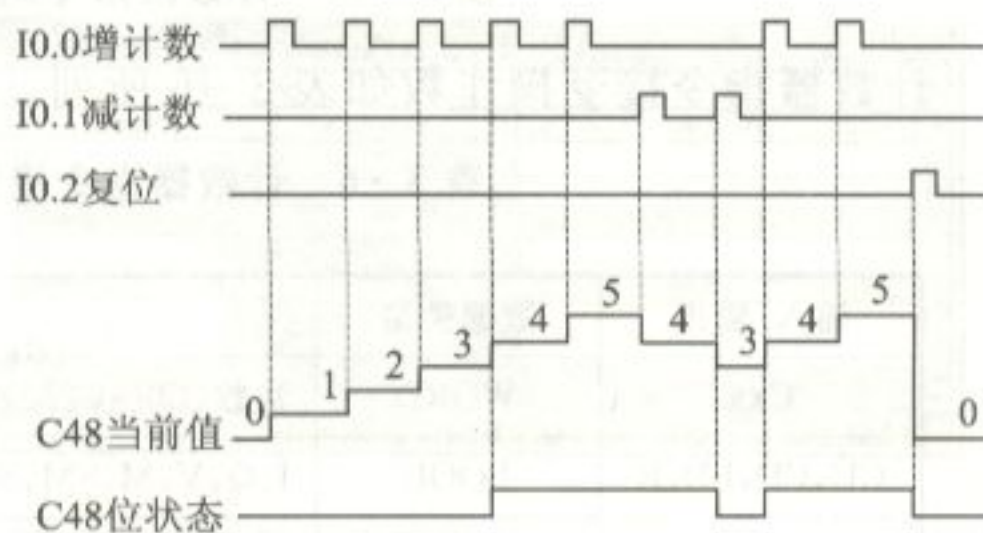
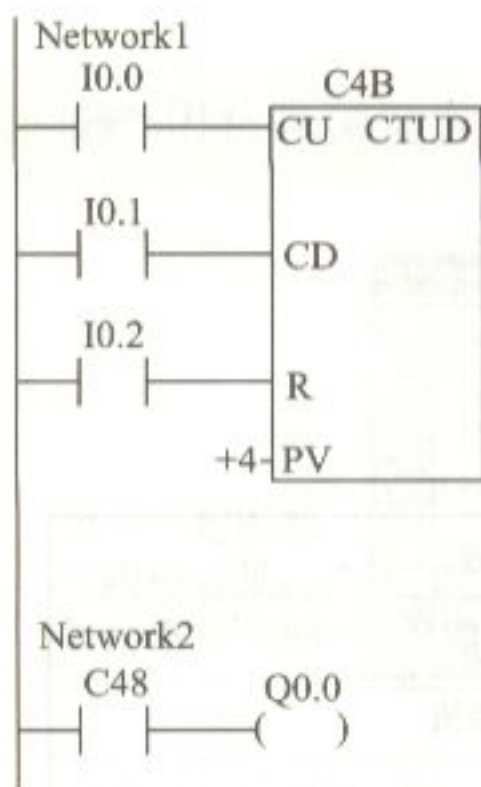


图 3-18 增/减计数器指令

图 3-19 时序图

在 3.1.1 节的例子中, 加上一个计数器, 使灯的延时时间达到 100 s(如图 3-20 所示)。

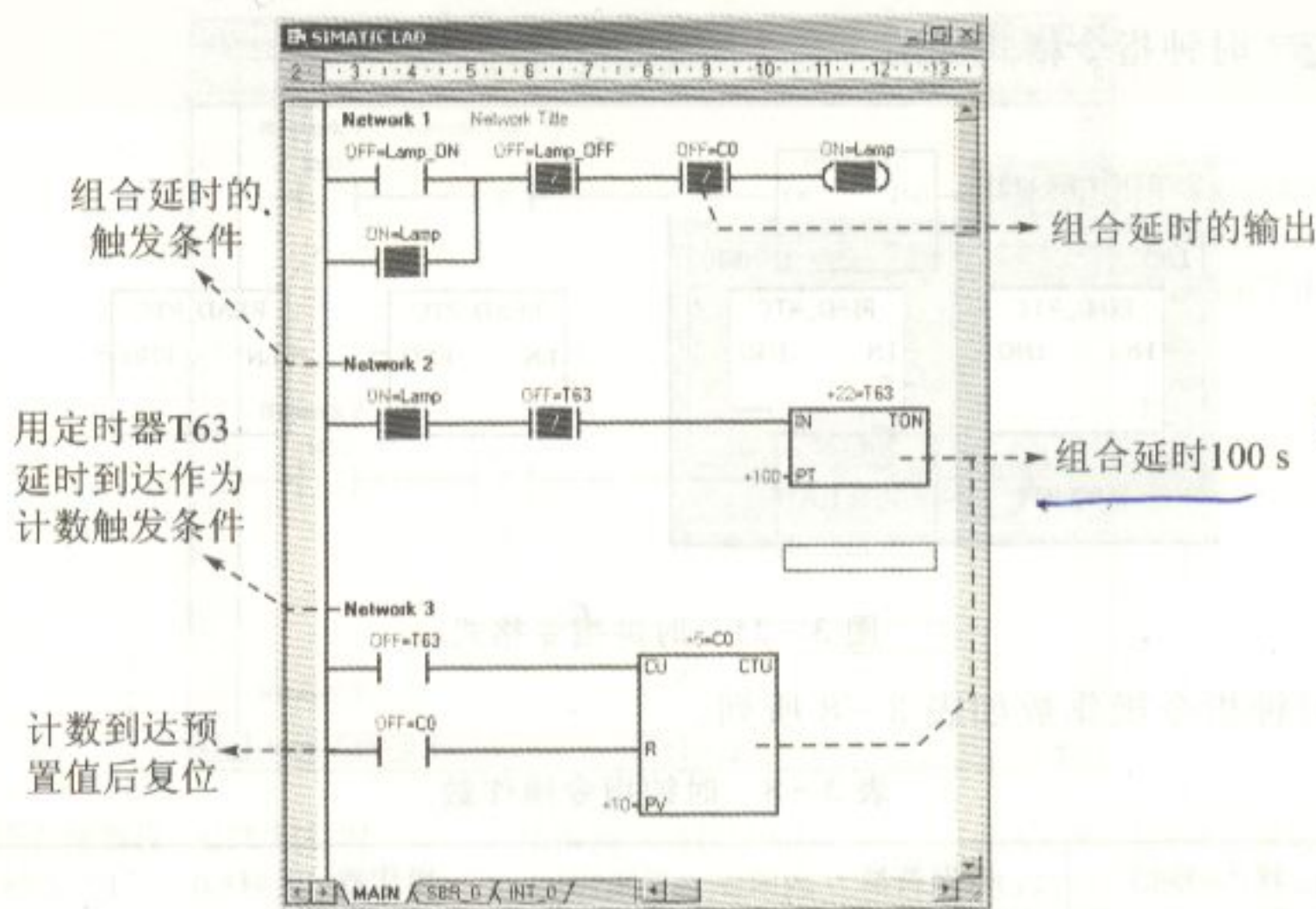


图 3-20



在这个例子中, T63 和 C0 组成一个延时指令组合。设置不同的定时器时基和计数器预置值可以组合出范围广泛的延时时间, 例如实现长时间延时。

3.3 系统时钟

S7-200 CPU 提供时钟指令对 CPU 的系统时钟进行操作:

● READ_RTC: 读系统时钟;

● SET_RTC: 写系统时钟。

EN: 使能输入端, 执行指令;

T: 以 T 开始的 8 个字节的时钟缓冲区。



👉 时钟指令格式如图 3-21 所示。

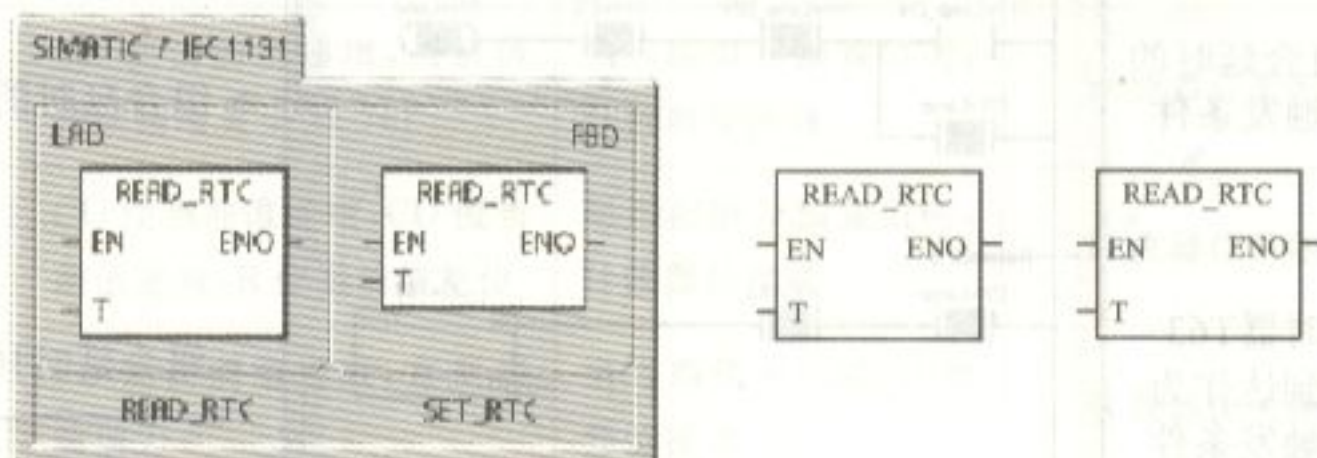


图 3-21 时钟指令格式

时钟指令操作数如表 3-8 所列。

表 3-8 时钟指令操作数

输入/输出	数据类型	操作数
T	BYTE	IB、QB、VB、MB、SMB、SB、LB、* VD、* LD、* AC

两个时钟指令的缓冲区具有同样的格式：

T	T+1	T+2	T+3	T+4	T+5	T+6	T+7
年	月	日	小时	分钟	秒	0	星期
00~99	01~12	01~31	00~23	00~59	00~59		0~7 *

* T+7 1=星期日,7=星期六;0=禁用。

✌ 日期和时间值按 BCD 格式表示。按 16 进制查看时钟缓冲区得到正确的数据；写时钟时要按 BCD 格式编码日期时间值。



S7-200 CPU 224/CPU 226/CPU 226 XM 已经内置系统实时时钟；CPU 221/CPU 222 需要外插时钟/电池卡。

👉 下面的例子程序(如图 3-22 所示)不断读取实时时钟,并在 M0.0 为“1”时将预设的日期时间写入实时时钟。

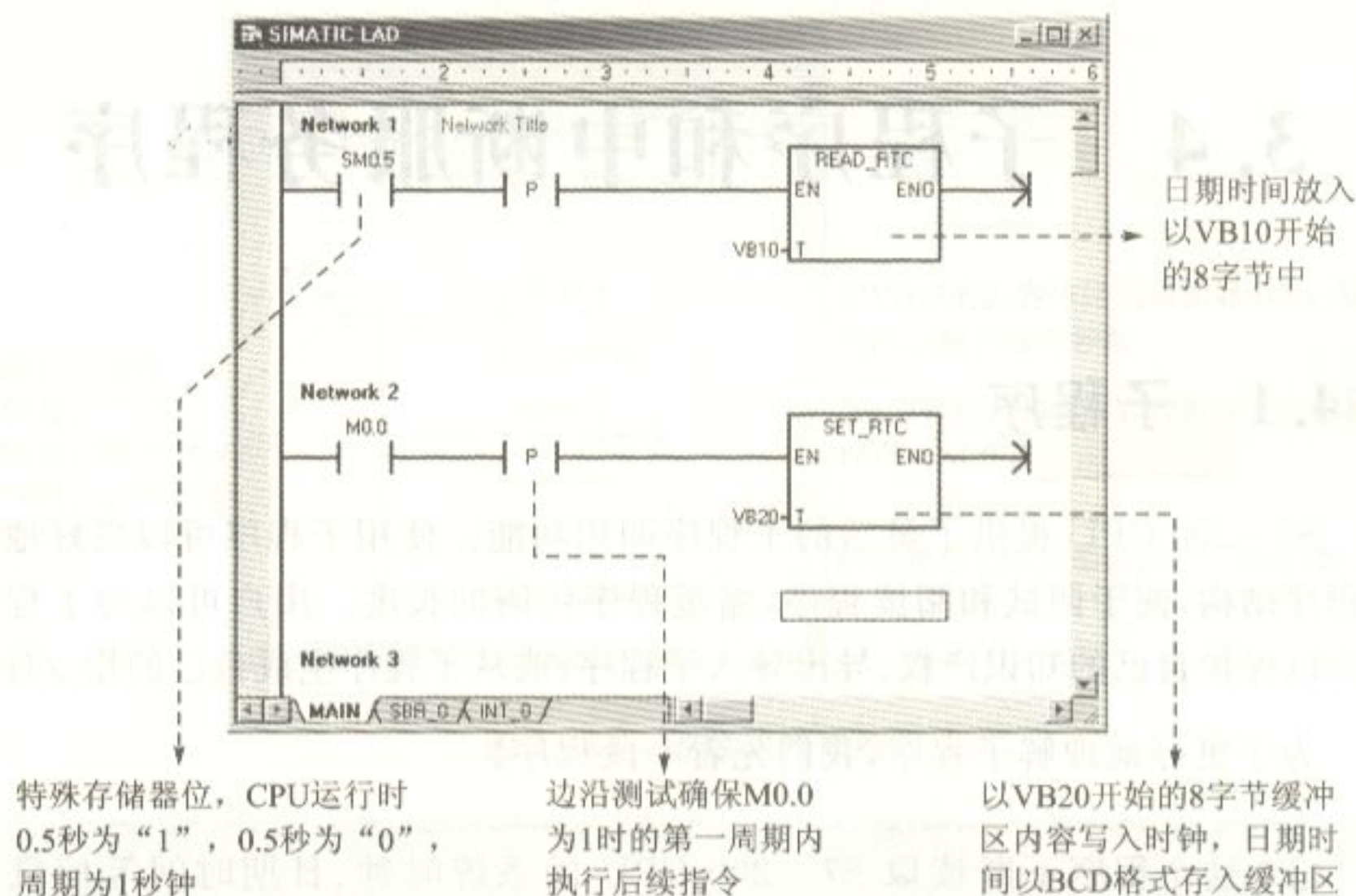


图 3-22 读写实时时钟程序举例

✌ 由于可读取的系统时钟最小单位为秒, 程序中对特殊存储器位 SM0.5 进行上升沿测试, 每秒钟读取一次。这样可以提高程序的执行效率。

🔍 运行程序, 使用状态图监控程序的运行如图 3-23 所示。

Status Chart																	
	Address	Format	Current Value	New Value													
1	VB10	Hexadecimal	15B03														
2	VB11	Hexadecimal	15B04														
3	VB12	Hexadecimal	15B05														
4	VB13	Hexadecimal	15B06														
5	VB14	Hexadecimal	15B07														
6	VB15	Hexadecimal	15B08														
7	VB16	Hexadecimal	15B09														
8	VB17	Hexadecimal	15B0A														
9		Signed															

图 3-23 使用状态图监控



3.4 子程序和中断服务程序

3.4.1 子程序

S7-200 CPU 提供了灵活子程序调用功能。使用子程序可以更好地组织程序结构,便于调试和阅读程序,缩短程序代码的长度。用户可以为子程序加密以保护自己的知识产权;导出导入子程序;或从子程序生成自己的指令库。

为了更好地理解子程序,我们先看一段程序。

☞ 这个程序不断读取 S7-200 CPU 的系统时钟、日期时间等信息以 BCD 格式保存在从 VB10 开始的 8 个字节中(如图 3-24 所示)。

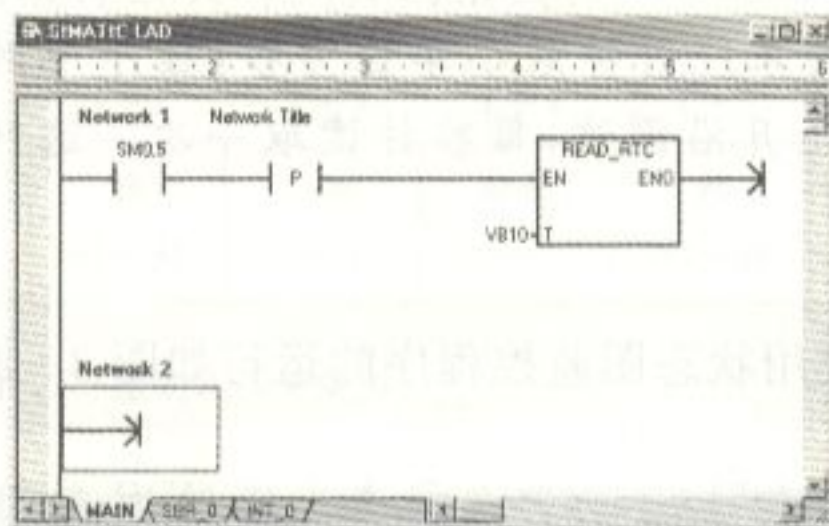


图 3-24 读系统时钟程序

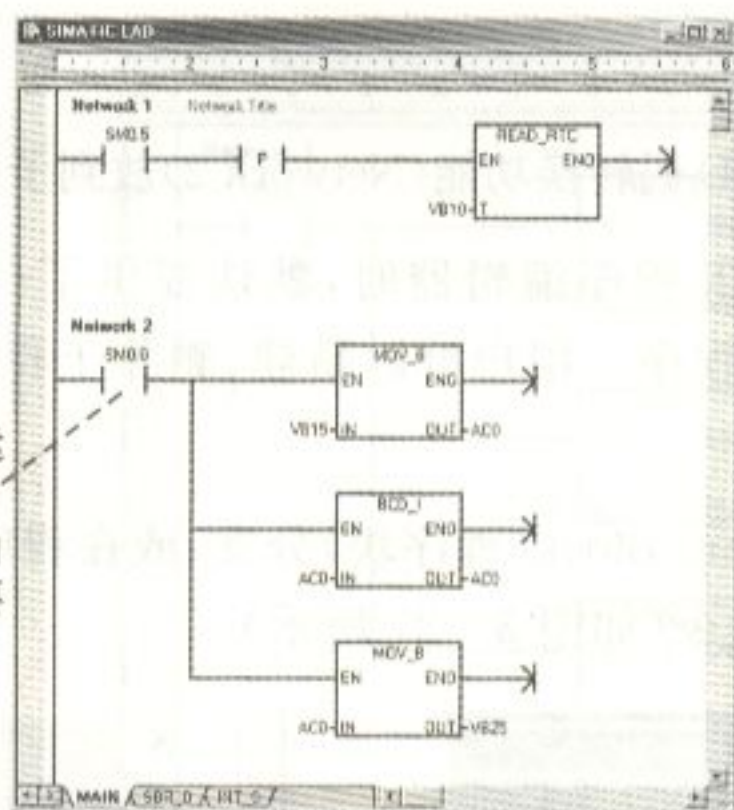
如果要计算日期时间,就需要把数据从 BCD 格式转换为十进制整数格式。

☞ 下面的程序把时钟的秒数转换为十进制整数(如图 3-25 所示)。

时钟的秒数只有一个字节长,而转换指令需要一个字(两个字节)长的操作数,因此使用累加器进行运算。



特殊存储器位，运行时总为“1”，在所有扫描周期内必须执行的指令要以SM0.0开始。



把VB15的内容传送到累加器AC0，VB15按BCD格式保存秒数

按1字长把AC0的内容转换为十进制整数再保存在AC0中

将转换结果从累加器AC0传送到VB25中

图 3-25 秒数的转换程序



对累加器 AC0、AC1、AC2 和 AC3 来说，参与运算的数据长度取决于所使用的指令。可以灵活应用累加器以配合指令对数据长度的要求。

运行程序，在状态图中对程序运行进行监视(如图 3-26 所示)。

按16进制查看 BCD格式数据

按无符号整数模式查看十进制数据

VB25中保存了十进制的时钟秒数值

	Address	Format	Current Value
1	VB10	Hexadecimal	16B03
2	VB11	Hexadecimal	16B05
3	VB12	Hexadecimal	16B14
4	VB13	Hexadecimal	16B17
5	VB14	Hexadecimal	16B55
6	VB15	Hexadecimal	16B21
7	VB16	Hexadecimal	16B00
8	VB17	Hexadecimal	16B07
9		Signed	
10	VB20	Unsigned	0
11	VB21	Unsigned	0
12	VB22	Unsigned	0
13	VB23	Unsigned	0
14	VB24	Unsigned	0
15	VB25	Unsigned	23
16	VB26	Unsigned	0
17	VB27	Unsigned	0

图 3-26 状态图监视



子程序举例

可以把在上面的例子中的 BCD 码转换功能(Network 2)放到子程序中执行。

Step7 - Micro/WIN32 在打开程序编辑器时,默认提供了一个空的子程序 SBR_0,可以直接在其中输入程序。用户可以新建、删除子程序,或者给子程序改名。

可用鼠标在指令树的 Program Block(程序块)分支,或在程序编辑器的子程序标签上单击右键执行上述命令(如图 3-27 所示)。

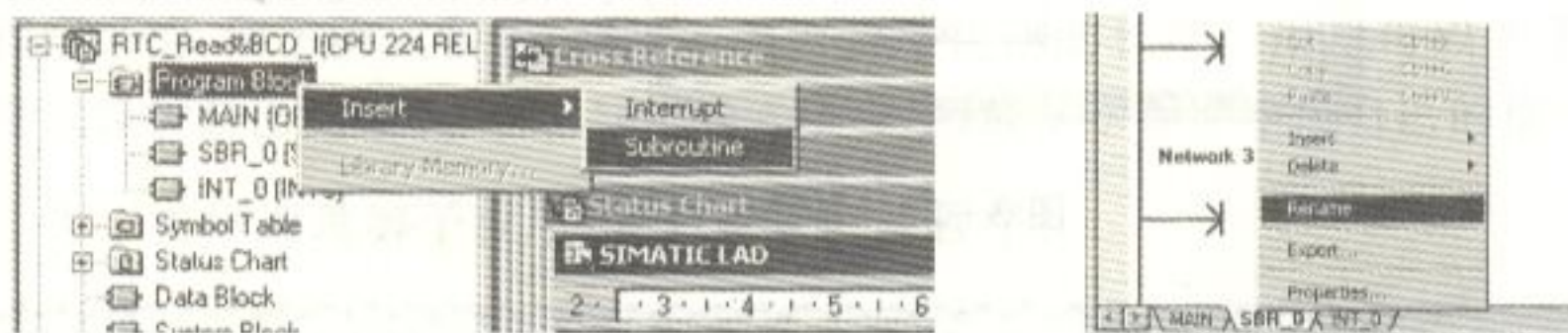


图 3-27 操作子程序



可以使用 Files(程序)菜单中的 Save as...(另存为)命令复制一个 Project(项目)在其中编辑、修改;或新建一个项目。当同时打开两个 Step7 - Micro/WIN32 窗口并分别编辑不同项目时,可以在它们之间进行复制、粘贴等操作。



在子程序 SBR_0 中插入转换程序(如图 3-28 所示)。



在主程序中调用子程序(如图 3-29 所示)。



Step7 - Micro/WIN32 会自动在子程序末尾加上返回指令。S7-200 系统还提供了 RET(条件返回)指令,根据条件选择是否提前返回调用它的程序。返回指令在指令树的 Program Control(程序控制)分支中。

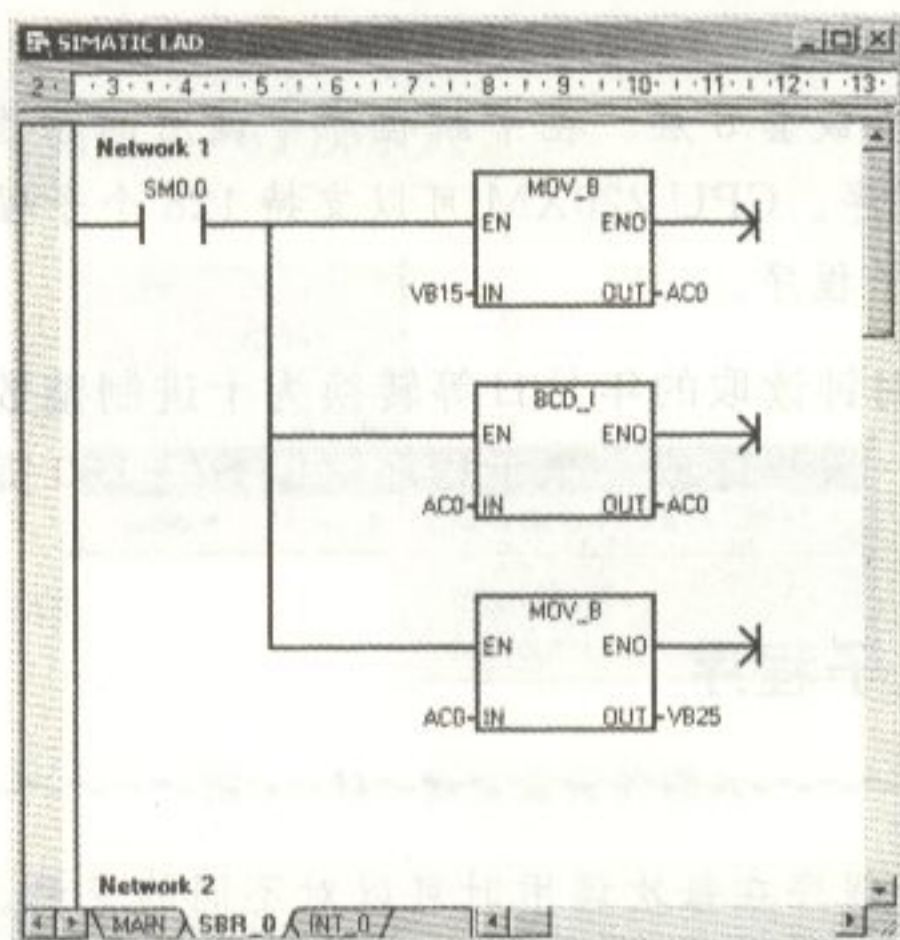


图 3-28 子程序中插入转换程序

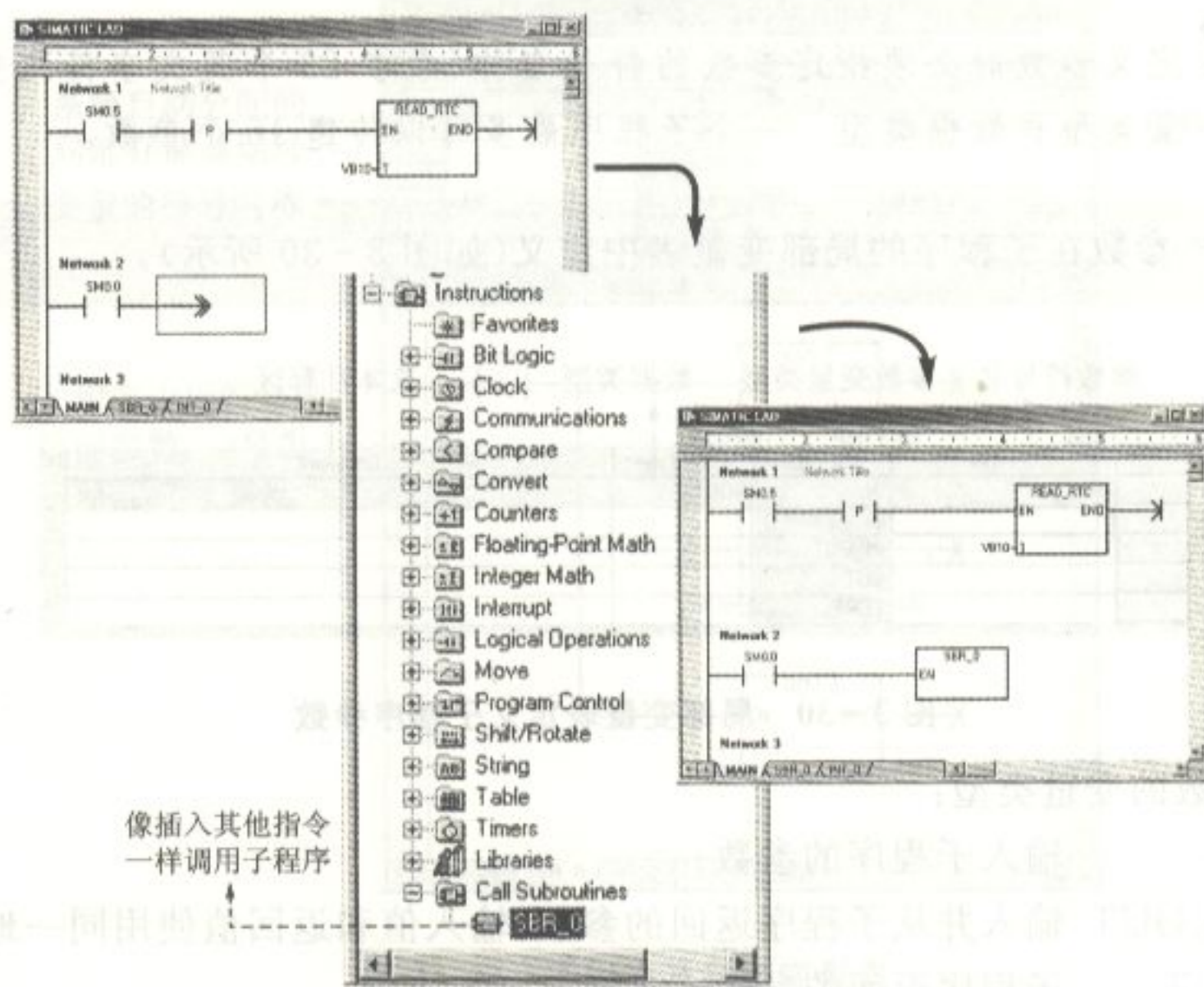


图 3-29 子程序的调用



子程序可以嵌套调用(即在子程序中调用子程序)。从主程序算起,一共可以嵌套8层。在中断程序中调用的子程序,不能再调用其他子程序。CPU 226XM 可以支持128个子程序,其他CPU支持64个子程序。

如果需要把从时钟读取的年月日等转换为十进制整数格式,可以为每一个字节编写专门的一段程序或一个子程序。但S7-200提供了更好的功能:带参数调用子程序。

带参数调用子程序



带参数的子程序在每次调用时可以对不同的变量、数据进行相同的运算、处理,提高程序编辑和执行的效率,节省程序存储空间。



定义参数时必须指定参数的符号名称(最多23个英文字符)、变量类型和数据类型。一个子程序最多可以传递16个参数。



参数在子程序的局部变量表中定义(如图3-30所示)。

参数符号名	参数变量类型	数据类型	文本注释区
Symbol	Val Type	Data Type	Comment
EN	IN	BOOL	
	IN		
	IN_OUT		
	OUT		
	TEMP		

图3-30 局部变量表定义子程序参数

参数的变量类型:

IN 输入子程序的参数;

IN_OUT 输入并从子程序返回的参数,输入值和返回值使用同一地址;

OUT 子程序返回的参数;

TEMP 临时变量,仅用于子程序内部暂存数据。



把光标放在局部变量表中要加入参数的区域,按鼠标右键,使用弹出菜单插入新变量行(如图 3-31 所示)。

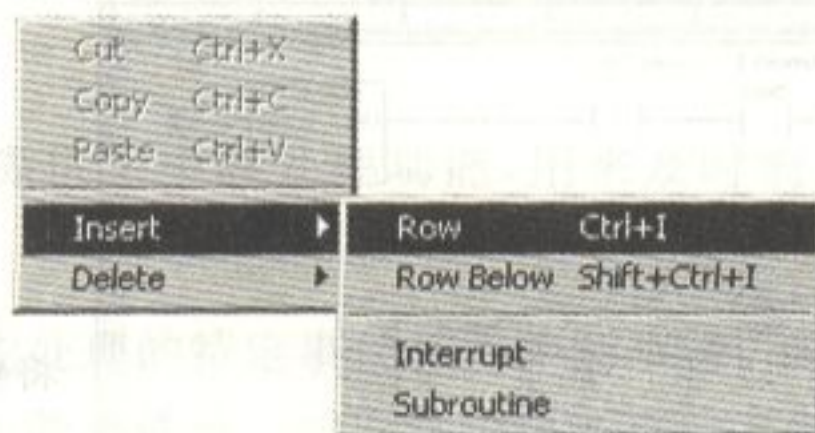


图 3-31 新变量行的插入

编辑完成的子程序 SBR_0 及其局部变量表(如图 3-32 所示)。

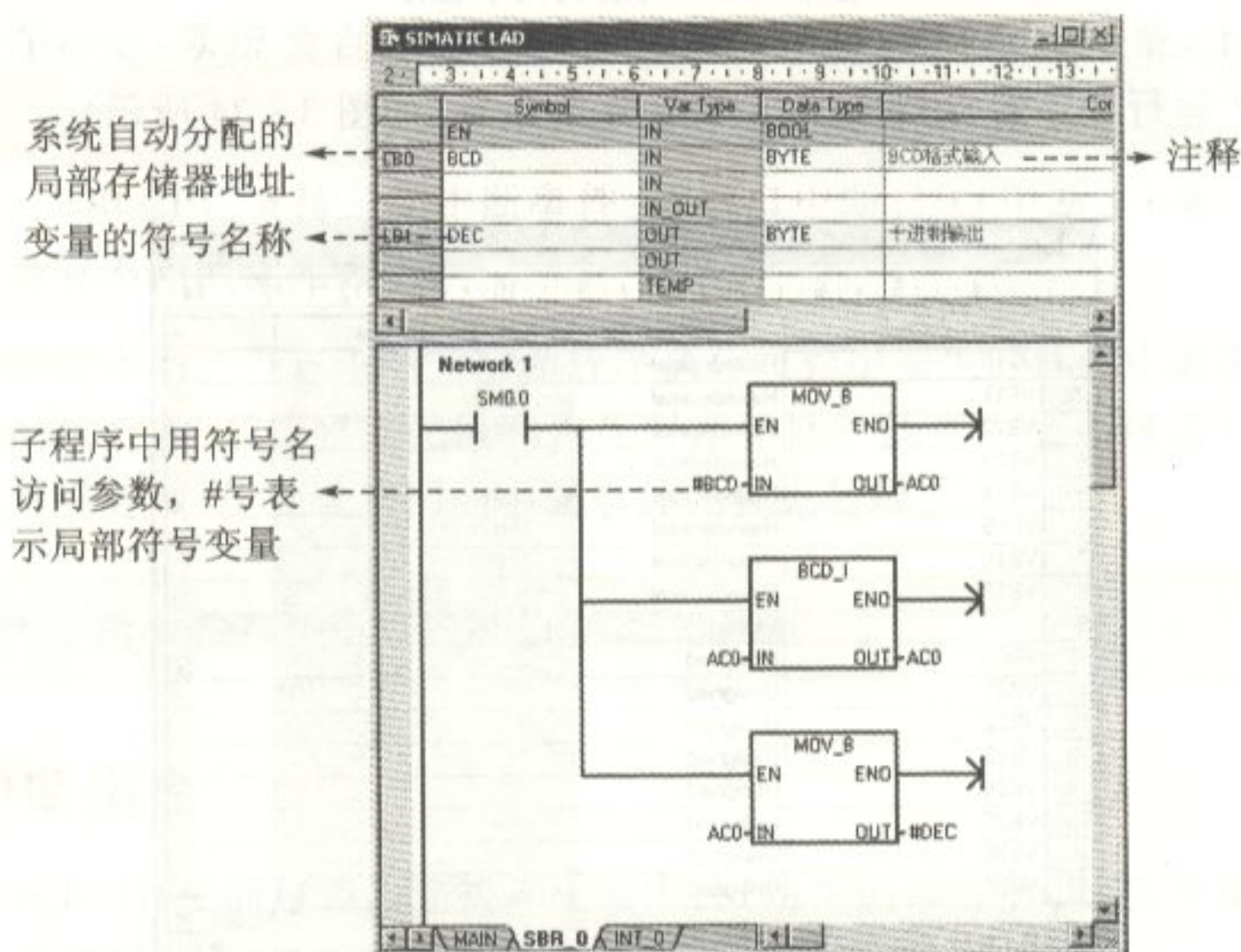


图 3-32 子程序及局部变量表



在主程序中两次调用子程序 SBR_0(如图 3-33 所示)。

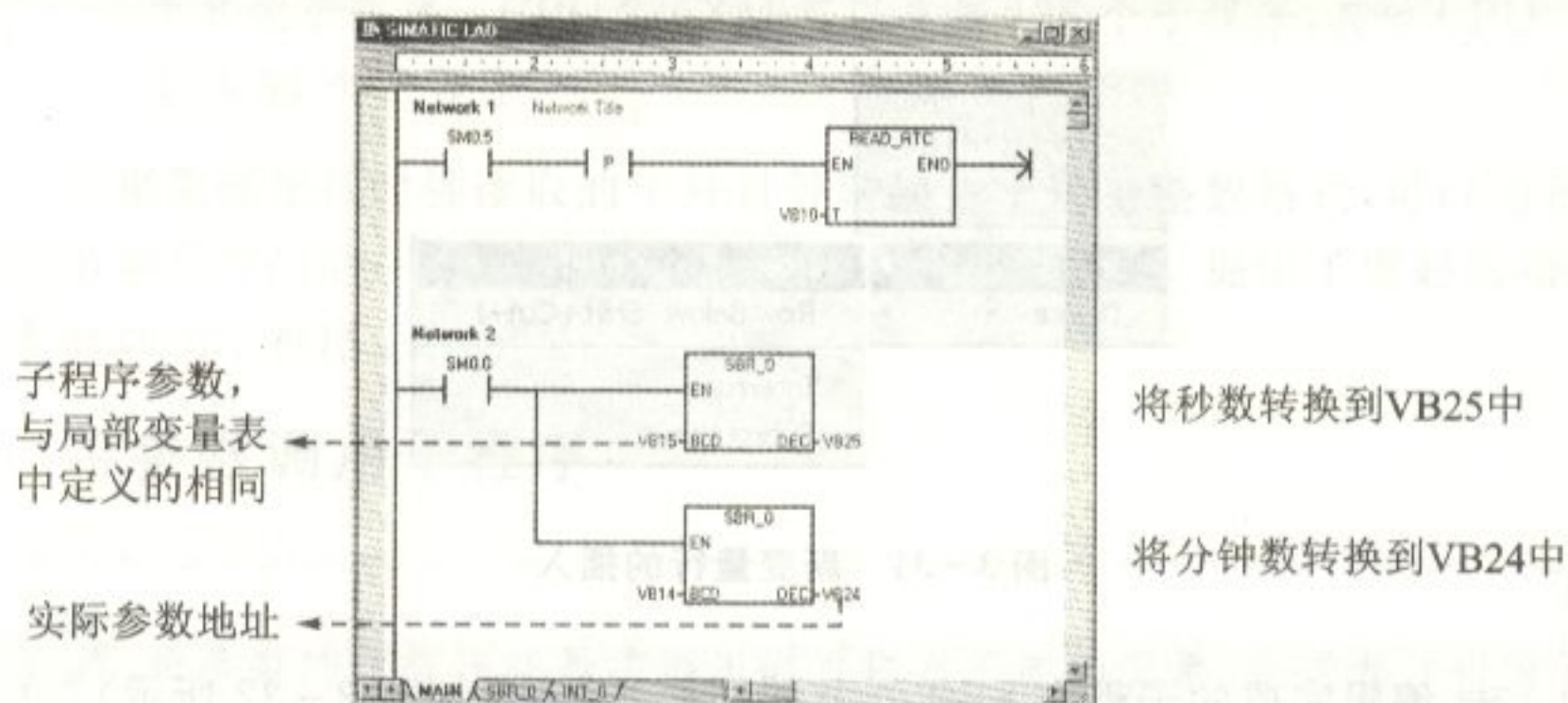


图 3-33 两次调用子程序

运行程序后，用状态图监视程序的运行(如图 3-34 所示)。

Status Chart			
3 4 5 6 7 8 9 10 11 12 13 14			
	Address	Format	Current Value
1	VB10	Hexadecimal	16#03
2	VB11	Hexadecimal	16#05
3	VB12	Hexadecimal	16#14
4	VB13	Hexadecimal	16#23
5	VB14	Hexadecimal	16#10
6	VB15	Hexadecimal	16#24
7	VB16	Hexadecimal	16#00
8	VB17	Hexadecimal	16#07
9		Signed	
10	VB20	Unsigned	0
11	VB21	Unsigned	0
12	VB22	Unsigned	0
13	VB23	Unsigned	0
14	VB24	Unsigned	10
15	VB25	Unsigned	24
16	VB26	Unsigned	0
17	VB27	Unsigned	0

注：分钟和秒数转换为十进制数

图 3-34 状态图监视



3.4.2 中断服务程序

中断事件


S7-200 CPU 提供了中断处理功能,用来及时响应特定的内部或外部事件。

能够用中断功能处理的特定事件称为中断事件。S7-200 系统为每个中断事件规定了一个中断事件号。响应某个中断事件发生而需要执行的程序称为中断服务程序,把中断事件号和中断服务程序联系起来才能执行中断处理功能。多个中断事件可以调用同一个中断程序,一个中断事件不可以连接多个中断程序。

中断事件会在 S7-200 CPU 程序循环周期中任何时刻发生。执行中断服务程序前后,系统会自动保护和恢复被中断的程序运行环境,不会造成混乱。

S7-200 CPU 支持三类中断事件:通讯口中断、I/O 中断、时基中断。中断事件各有不同的优先级别。

中断程序不会再被中断。中断程序执行过程中发生的其他中断事件不会影响它的执行,而是按照优先级和发生时序排队。队列中优先级高的中断事件首先得到处理;优先级相同的中断事件先到先处理。

 中断事件号及其优先级如表 3-9 所列。

中断指令

- ATCH(中断连接):连接某中断事件(由中断事件号指定)所要调用的程序段(由中断程序号指定)。
- ENI(全局允许中断):开放中断处理功能。
- DISI(全局禁止中断):禁止处理中断服务程序,但中断事件仍然会排队等候。



表 3-9 中断事件号及其优先级

事件号	中断描述	优先组	优先组中的优先级
8	端口 0:接收字符	通讯(最高)	0
9	端口 0:发送完成		0
23	端口 0:接收信息完成		0
24	端口 1:接收信息完成		1
25	端口 1:接收字符		1
26	端口 1:发送完成		1
19	PTO 0 完成中断	I/O(中等)	0
20	PTO 1 完成中断		1
0	上升沿, I0.0		2
2	上升沿, I0.1		3
4	上升沿, I0.2		4
6	上升沿, I0.3		5
1	下降沿, I0.0		6
3	下降沿, I0.1		7
5	下降沿, I0.2		8
7	下降沿, I0.3		9
12	HSC0 CV=PV(当前值=预置值)		10
27	HSC0 输入方向改变		11
28	HSC0 外部复位		12
13	HSC1 CV=PV(当前值=预置值)		13
14	HSC1 输入方向改变		14
15	HSC1 外部复位		15
16	HSC2 CV=PV(当前值=预置值)		16
17	HSC2 输入方向改变		17
18	HSC2 外部复位		18
32	HSC3 CV=PV(当前值=预置值)		19
29	HSC4 CV=PV(当前值=预置值)		20
30	HSC4 输入方向改变		21
31	HSC4 外部复位		22
33	HSC5 CV=PV(当前值=预置值)		23
10	定时中断 0, SMB34	定时(最低)	0
11	定时中断 1, SMB35		1
21	定时器 T32 CT=PT 中断		2
22	定时器 T96 CT=PT 中断		3



- DTCH(中断分离): 将中断事件号与中断服务程序之间的关联切断, 并禁止该中断事件。

- RETI(条件中断返回): 根据逻辑操作的条件, 从中断服务程序中返回。

可在 Step7 - Micro/WIN32 指令树的 Interrupt(中断)分支中找到与中断处理有关的指令。



中断服务程序执行完毕后会自动返回。RETI(条件中断返回)指令用来在中断程序中间, 根据逻辑运算的结果决定是否返回。

中断程序示例

S7-200 CPU 提供了时基中断处理功能, 用来执行精确定时的周期性任务。例如, 对模拟量信号采样, 或执行 PID 回路控制功能。时基中断包括两个特殊存储器定时中断和两个定时器中断。以 1 ms 为单位, 可以指定从 1~255 ms 的周期范围。



如果在中断服务程序中对定时中断事件进行计数, 选择性地执行某些运算操作, 就可以得到长于 255 ms 的周期。

本例使用定时中断实现对 100 ms 定时周期计数。我们使用定时中断 0。

查中断事件表 3-9, 可以得知定时中断 0 的中断事件号为 10, 确定周期的特殊存储器字节是 SMB34。

在程序中:

SBR_0: 中断初始化程序;

INT_0: 中断服务程序。





在主程序中调用 SBR_0(如图 3-35 所示)。

初始化中断程序只需调用一次，因此用SM0.1作为条件

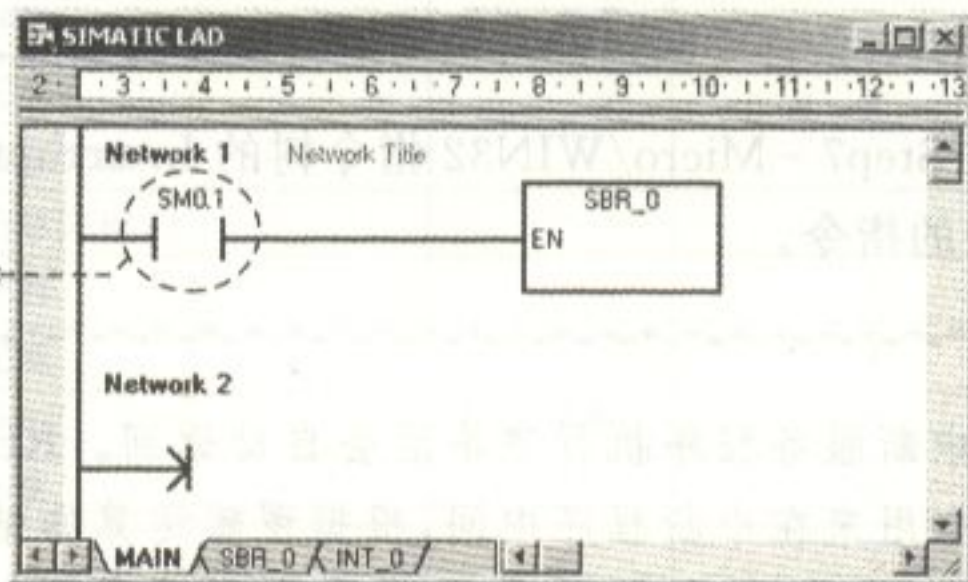
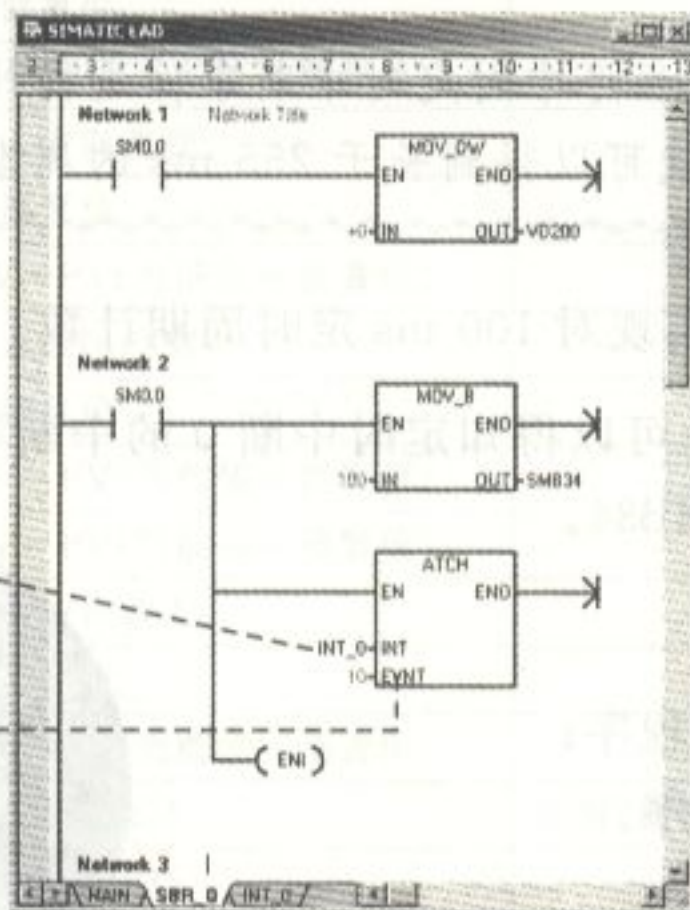


图 3-35 SBR_0 的调用



中断程序的初始化，只需执行一次，除非重新定义中断事件。

SBR_0 编程如图 3-36 所示。



计数存储区清零

写入100 ms周期

连接中断事件和程序

全局允许中断

指定中断服务程序名称

连接10号中断事件，即定时中断0

图 3-36 SBR_0 编程



☞ INT_0 编程(如图 3-37 所示)。

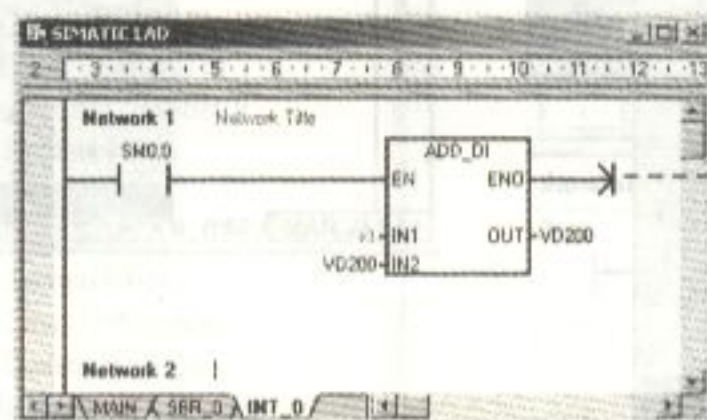


图 3-37 INT_0 编程

在 S7-200 CPU 中运行程序,INT_0 会自动根据定时中断事件的发生而执行。

☞ 使用状态图监视,VD200 的内容就是 100 ms 周期到达的次数(如图 3-38 所示)。

	Address	Format	Current Value
1	VD200	Unsigned	4274
2		Signed	
3		Signed	
4		Signed	
5		Signed	

图 3-38 状态图监视

3.4.3 程序的密码保护

Step7 - Micro/WIN32 提供了完善的密码保护功能,以保护用户的知识产权。

☞ 在 Step7 - Micro/WIN32 指令树的项目分支中,或者在程序编辑器中,用鼠标右键单击主程序、子程序或者中断程序标记,在弹出的快捷菜单中选择 Properties...(属性),如图 3-39 所示。

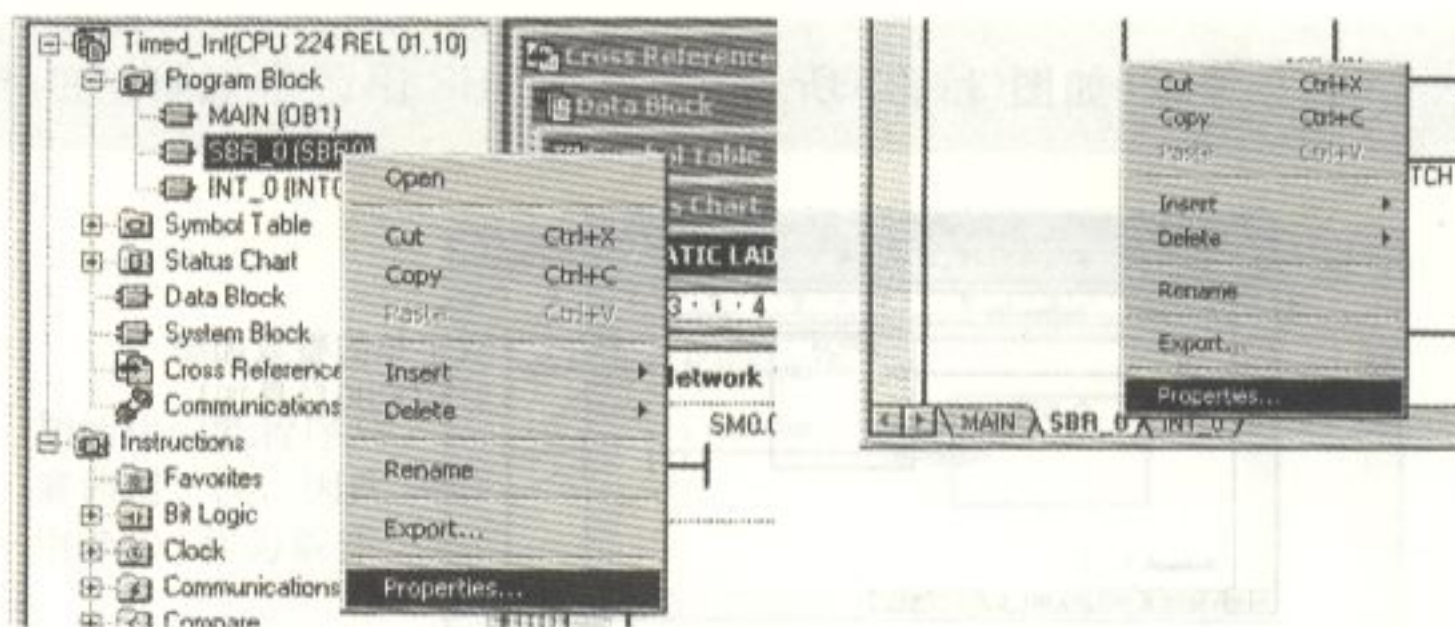


图 3-39 查看主程序、子程序或中断程序标记的属性

☞ 在程序的属性标签中,选择 Protection(保护)标签(如图 3-40 所示)。

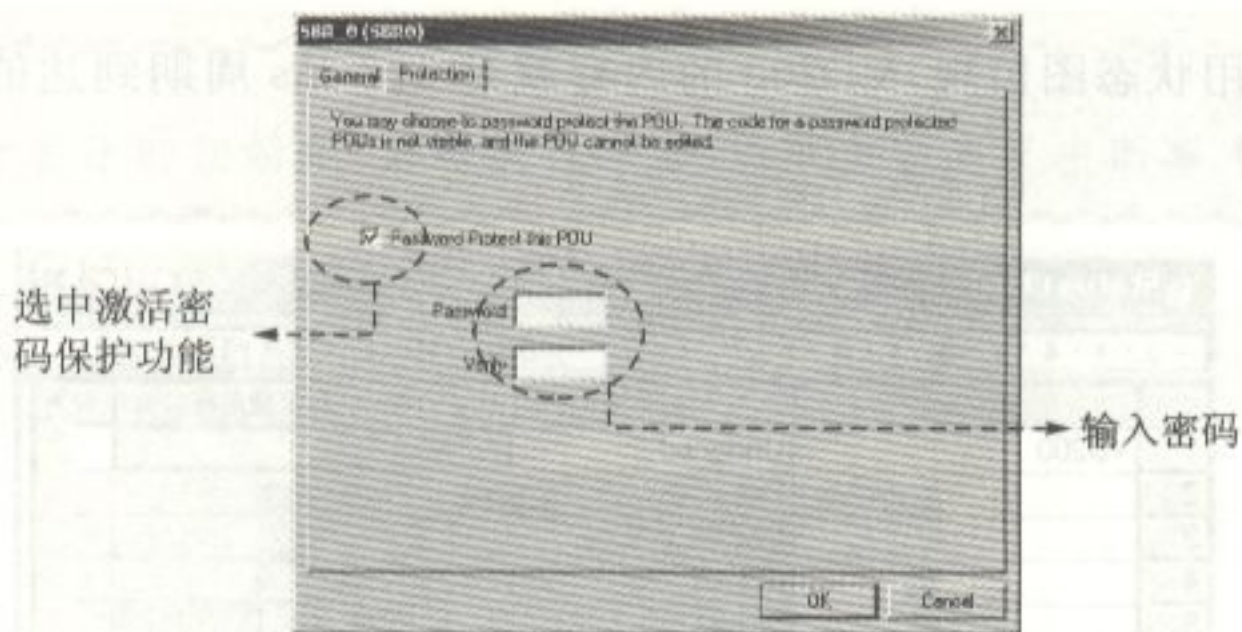


图 3-40 “保护”标签

☞ 输入密码,按 OK 按钮后,程序即被加密保护。在指令树相应的条目上会显示加密标记。在程序编辑器中也不能查看程序的内容(如图 3-41 所示)。

☞ 要显示程序内容,需打开程序属性,并在 Protection(保护)标签中输入密码,再按 Authorize(验证)按钮(如图 3-42 所示)。



加密的程序下载到 S7-200 CPU 中,再使用 Step7 - Micro/WIN32 上载后也处在加密状态。

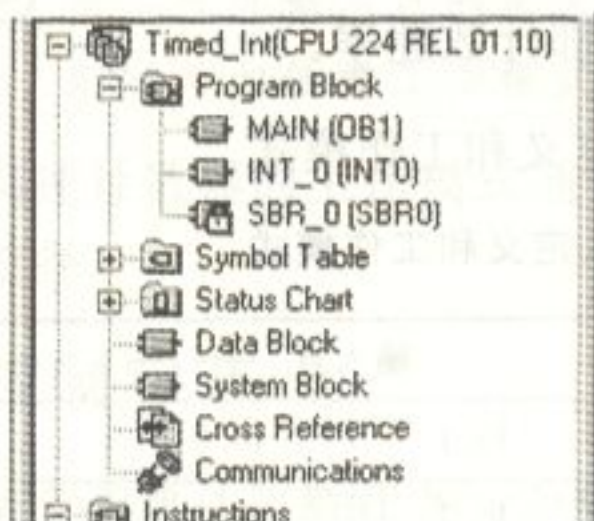


图 3-41 加密标志在命令树上的显示

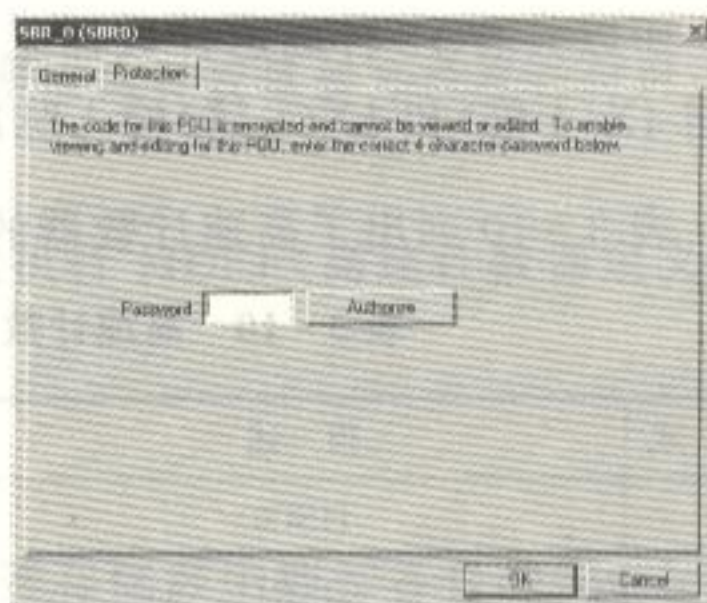


图 3-42 验证密码

3.5 高速计数器

S7-200 CPU 提供了多个高速计数器(HSC0~HSC5)以响应快速的脉冲输入信号。高速计数器独立于用户程序工作,不受程序扫描时间的限制。用户通过相关指令,设置相应的特殊存储器控制高速计数器的工作。

高速计数器的工作模式

S7-200 CPU 高速计数器可以分别定义为 4 种工作类型:

- 单相计数器,内部方向控制;
- 单相计数器,外部方向控制;
- 双相增/减计数器,双脉冲输入;
- A/B 相正交脉冲输入计数器。

每种高速计数器类型可以设定为 3 种工作状态:

- 无复位、无启动输入;
- 有复位、无启动输入;
- 既有复位、又有启动输入。





所以共有 12 种高速计数器工作模式。对于 A/B 相正交输入,可以选择使用 4X(4 倍)和 1X(1 倍)输入脉冲频率的内部计数速率。

表 3-10 列出了高速计数器的硬件输入定义和工作模式。


表 3-10 高速计数器的硬件输入定义和工作模式

模 式	描 述	输入点			
	HSC0	I0.0	I0.1	I0.2	
	HSC1	I0.6	I0.7	I1.0	I1.1
	HSC2	I1.2	I1.3	I1.4	I1.5
	HSC3	I0.1			
	HSC4	I0.3	I0.4	I0.5	
	HSC5	I0.4			
0	带有内部方向控制的单相计数器	计数脉冲			
1		计数脉冲		复位	
2		计数脉冲		复位	启动
3	带有外部方向控制的单相计数器	计数脉冲	方向		
4		计数脉冲	方向	复位	
5		计数脉冲	方向	复位	启动
6	带有增/减计数脉冲的双相计数器	增计数脉冲	减计数脉冲		
7		增计数脉冲	减计数脉冲	复位	
8		增计数脉冲	减计数脉冲	复位	启动
9	A/B 相正交计数器	计数脉冲 A	计数脉冲 B		
10		计数脉冲 A	计数脉冲 B	复位	
11		计数脉冲 A	计数脉冲 B	复位	启动

✌ S7-200 CPU 221、CPU 222 没有 HSC1 和 HSC2 两个计数器；CPU 224、CPU 226 和 CPU 226XM 拥有全部 6 个计数器。

✌ 高速计数器的硬件输入接口与普通数字量输入接口使用相同的地址。已定义用于高速计数器的输入点不再具有其他功能,但某个模式下没有用到的输入点还可以用作普通开关量输入点。



 由于硬件输入点的定义不同,不是所有的计数器都可以在任何时刻定义为任意工作模式。

高速计数器的工作模式通过一次性地执行 HDEF(高速计数器定义)指令来选择。

控制字节

每个高速计数器在 S7-200 CPU 的特殊存储区中拥有各自的控制字节。控制字节用来定义计数器的计数方式和其他一些设置,以及在用户程序中对计数器的运行进行控制。控制字节的各个位的 0/1 状态具有不同的设置功能。

高速计数器控制字节的位地址分配如表 3-11 所列。

表 3-11 高速计数器控制字节的位地址分配

HSC0	HSC1	HSC2	HSC3	HSC4	HSC5	描 述
SM37.0	SM47.0	SM57.0	—	SM147.0	—	复位有效电平控制位:0=复位高电平有效;1=复位低电平有效
—	SM47.1	SM57.1	—	—	—	启动有效电平控制位:0=启动高电平有效;1=启动低电平有效
SM37.2	SM47.2	SM57.2	—	SM147.2	—	正交计数器计数速率选择:0=4X 计数率;1=1X 计数率
SM37.3	SM47.3	SM57.3	SM137.3	SM147.3	SM157.3	计数方向控制位:0=减计数;1=增计数
SM37.4	SM47.4	SM57.4	SM137.4	SM147.4	SM157.4	向 HSC 中写入计数方向:0=不更新;1=更新计数方向
SM37.5	SM47.5	SM57.5	SM137.5	SM147.5	SM157.5	向 HSC 中写入预置值:0=不更新;1=更新预置值
SM37.6	SM47.6	SM57.6	SM137.6	SM147.6	SM157.6	向 HSC 中写入新的初始值:0=不更新;1=更新初始值
SM37.7	SM47.7	SM57.7	SM137.7	SM147.7	SM157.7	HSC 允许:0=禁止 HSC;1=允许 HSC

高速计数器数值寻址

每个高速计数器都有一个初始值和一个预置值,它们都是 32 位有符号整



数。初始值是高速计数器计数的起始值；预置值是计数器运行的目标值，当前计数值等于预置值时会发生一个内部中断事件。必须先设置控制字节以允许装入新的初始值和预置值，并且把初始值和预置值存入特殊存储器中，然后执行 HSC 指令使其有效。

高速计数器数值如表 3-12 所列。

表 3-12 高速计数器数值寻址

计数器号	HSC0	HSC1	HSC2	HSC3	HSC4	HSC5
初始值	SMD38	SMD48	SMD58	SMD138	SMD148	SMD158
预置值	SMD42	6SMD52	SMD62	SMD142	SMD152	SMD162
当前值	HC0	HC1	HC2	HC3	HC4	HC5

当前值也是一个 32 位的有符号整数。HSC0 的当前值，在 HC0 中读取。

中断功能

所有的计数器模式都会在当前值等于预置值时产生中断；使用外部复位端的计数模式支持外部复位中断；除模式 0、1 和 2 之外，所有计数器模式还支持计数方向改变中断。每种中断条件都可以分别使能或者禁止。

S7-200 CPU 还在特殊存储区中为高速计数器提供了状态字节，以在中断服务程序中使用。状态字节只在中断程序中有效。

高速计数器编程

使用高速计数器，需完成下列步骤：

- 根据选定的计数器工作模式，设置相应的控制字节；
- 使用 HDEF 指令定义计数器号；
- 设置计数方向(可选)；
- 设置初始值(可选)；
- 设置预置值(可选)；
- 指定并使能中断服务程序(可选)；
- 执行 HSC 指令，激活高速计数器。

若在计数器运行中改变其设置，需执行下列步骤：



- 根据需要设置控制字节;
- 设置计数方向(可选);
- 设置初始值(可选);
- 设置预置值(可选);
- 执行 HSC 指令,使 CPU 确认。



用户还可以使用指令向导中的 HSC 向导生成程序。

内部方向控制无复位单相计数器编程

在下面的例子中,使 HSC0 工作在内部方向控制、无复位状态,即模式 0。所谓内部方向控制,就是通过高速计数器控制字节的方向位来控制计数的增/减方向。

为此,须将 HSC0 的控制字节 SMB37 设置为如图 3-43 所示的形式。

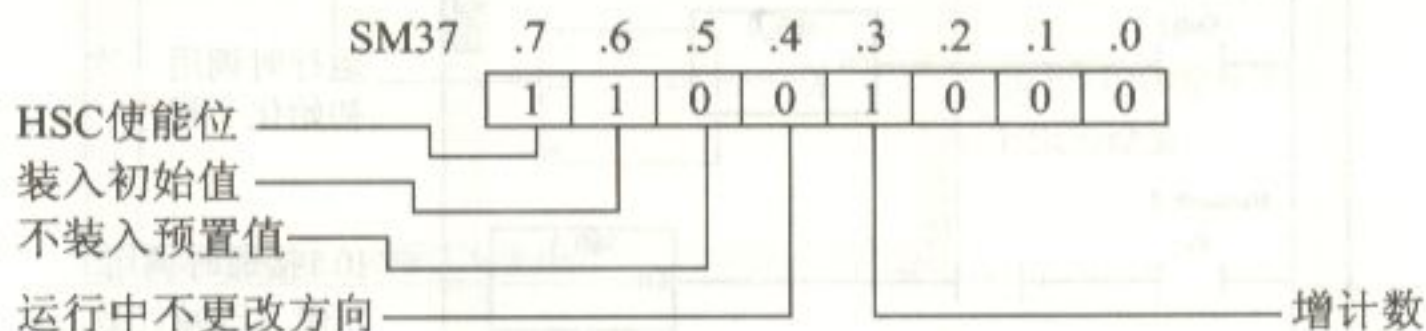


图 3-43 控制字节设置

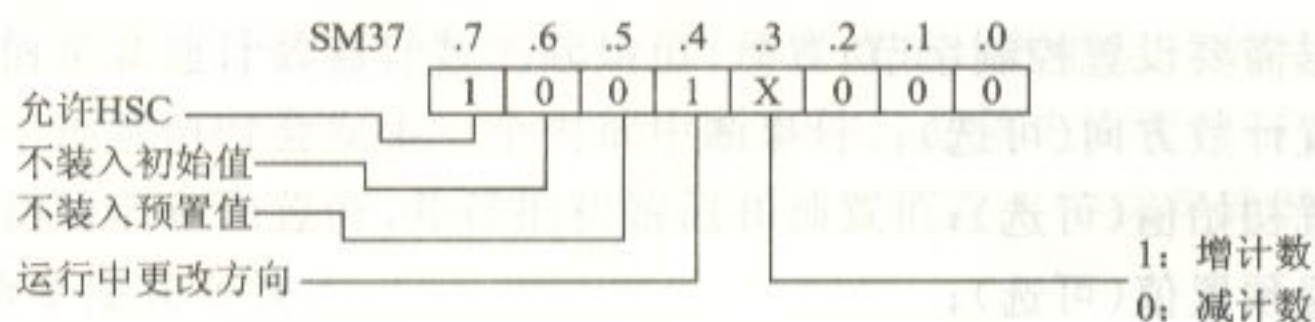
上述 SMB37 的内容用二进制表示为 11001000b,为了方便可以换算成十六进制格式的 C8h,用 S7-200 格式表示即为:

$$2\#11001000 = 16\#C8$$

如果在运行中改变方向,须设置控制字节的各个位,如图 3-44 所示。



如果感觉不便,也可以直接按二进制格式写入控制字节。



上述SMB37内容可表示为:

2#10011000=16#98或2#10010000=16#90

图 3-44 控制字节的位设置

在例子程序中:

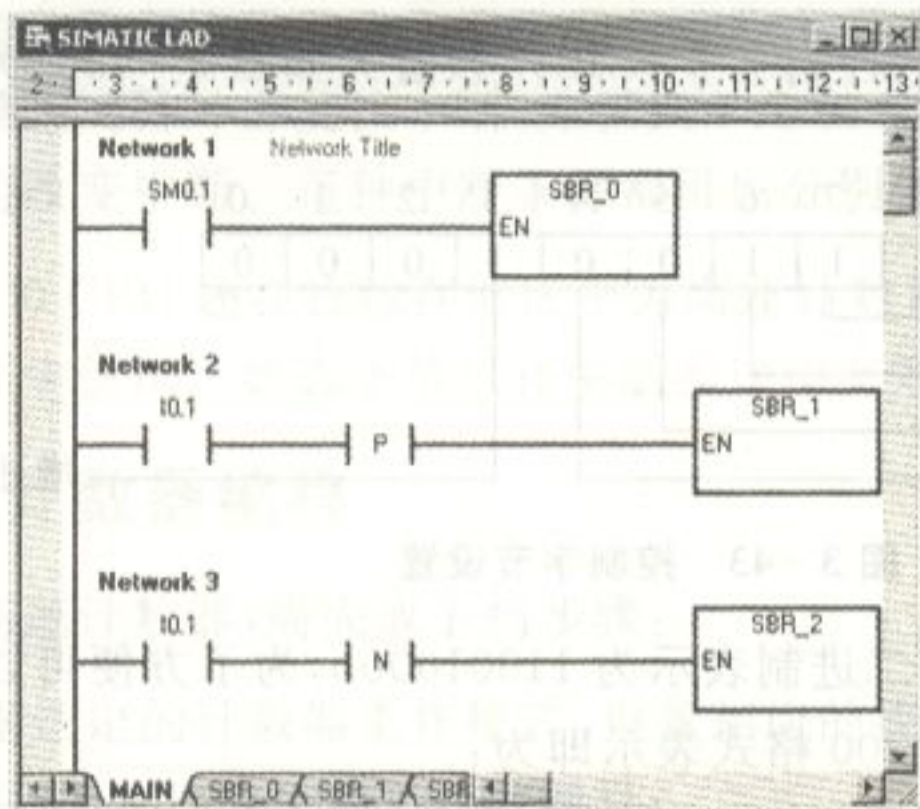
主程序:调用初始化子程序,并根据 I0.1 的状态调用子程序改变计数方向;

SBR_0:初始化 HSC0;

SBR_1:改计数方向为减计数;

SBR_2:改计数方向为增计数。

主程序编程如图 3-45 所示。



运行时调用一次
初始化子程序

I0.1接通时调用
SBR_1改减计数

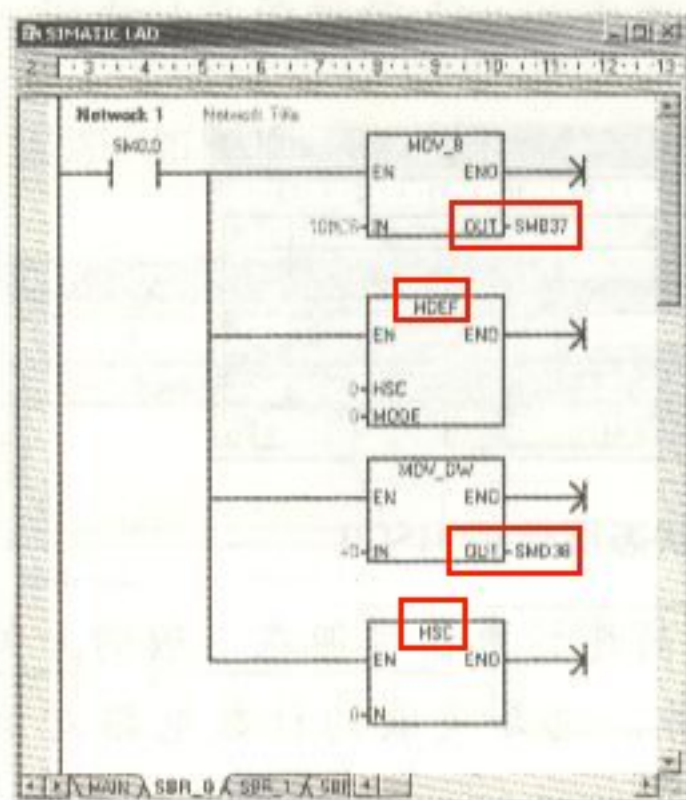
I0.1关断时调用
SBR_2改增计数

图 3-45 主程序编程

SBR_0 编程如图 3-46 所示。

SBR_1 编程如图 3-47 所示。

SBR_2 编程如图 3-48 所示。



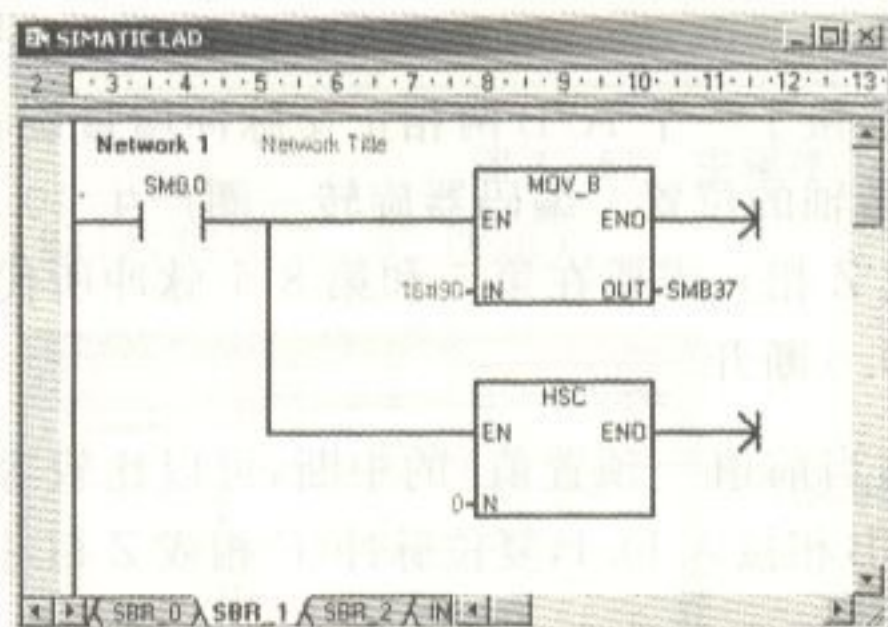
写入控制字节

定义HSC0为模式0

装入初始值0

执行HSC指令写入HSC0设置

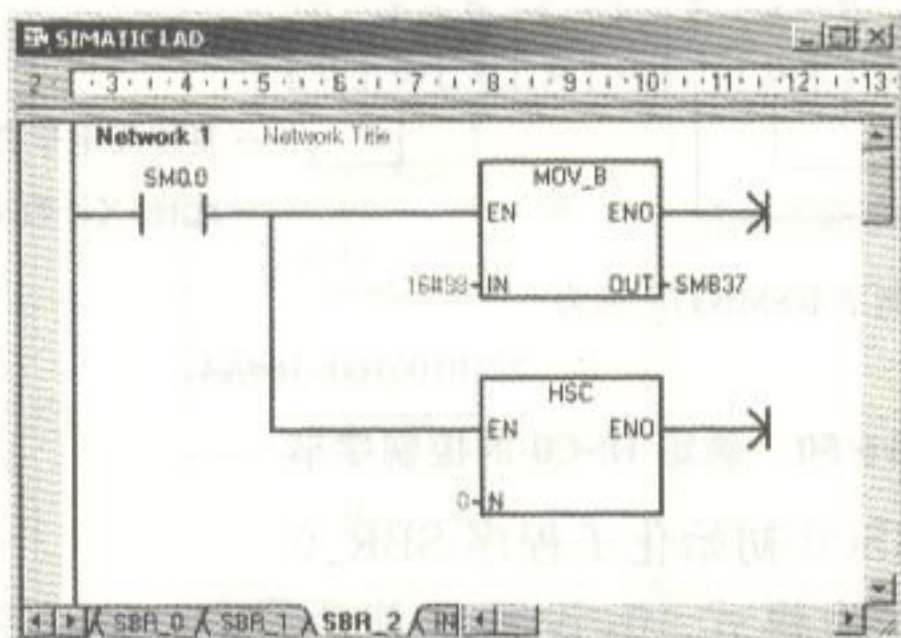
图 3-46 SBR_0 编程



写控制字节改变方向
为减计数

执行HSC指令写入
HSC0设置

图 3-47 SBR_1 编程



写控制字节改变方向
为增计数

执行HSC指令
写入HSC0设置

图 3-48 SBR_2 编程



✎ 使用状态图监视 HSC0 的当前值 HC0, 如图 3-49 所示。

	Address	Format	Current Value
1	HC0	Signed	100
2		Signed	
3	SMB37	Binary	2#10011000
4		Signed	
5	I0.1	Bit	0

图 3-49 用状态图监视 HSC0

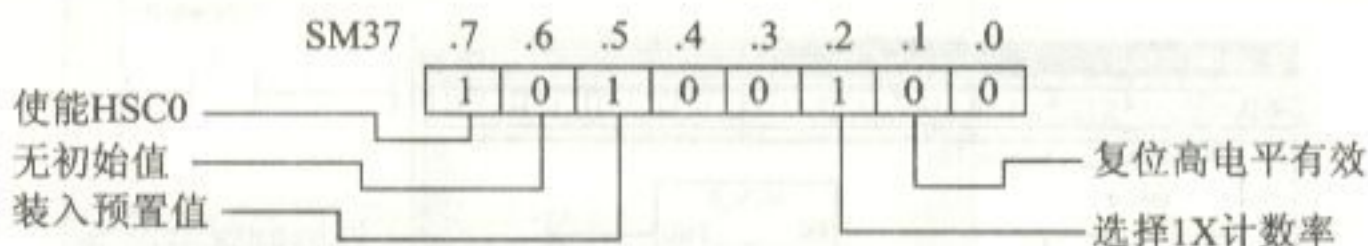
✎ 如果没有脉冲信号发生装置, 高速计数器可能在一般的开关或触点接通一次时计数出多个脉冲。这是灵敏的计数电路对信号输入的抖动或毛刺作出的反映。

模式 10 应用举例

假设某单向旋转机械上连接了一个 A/B 两相正交脉冲增量旋转编码器, 计数脉冲的个数就代表了旋转轴的位置。编码器旋转一圈产生 10 个 **A/B 相脉冲** 和一个 **复位脉冲(C 相或 Z 相)**, 需要在第 5 和第 8 个脉冲所代表的位置之间接通 Q0.0, 其余位置 Q0.0 断开。

利用 HSC0 的 $CV = PV$ (当前值 = 预置值) 的中断, 可以比较容易地实现这一功能。把 A 相接入 I0.0, B 相接入 I0.1, 复位脉冲(C 相或 Z 相)接入 I0.2。

✎ 确定 HSC0 的控制字节(如图 3-50 所示):



因此HSC0的控制字节SMB37应当为

$$2\#10100100 = 16\#A4。$$

图 3-50 确定 HSC0 的控制字节

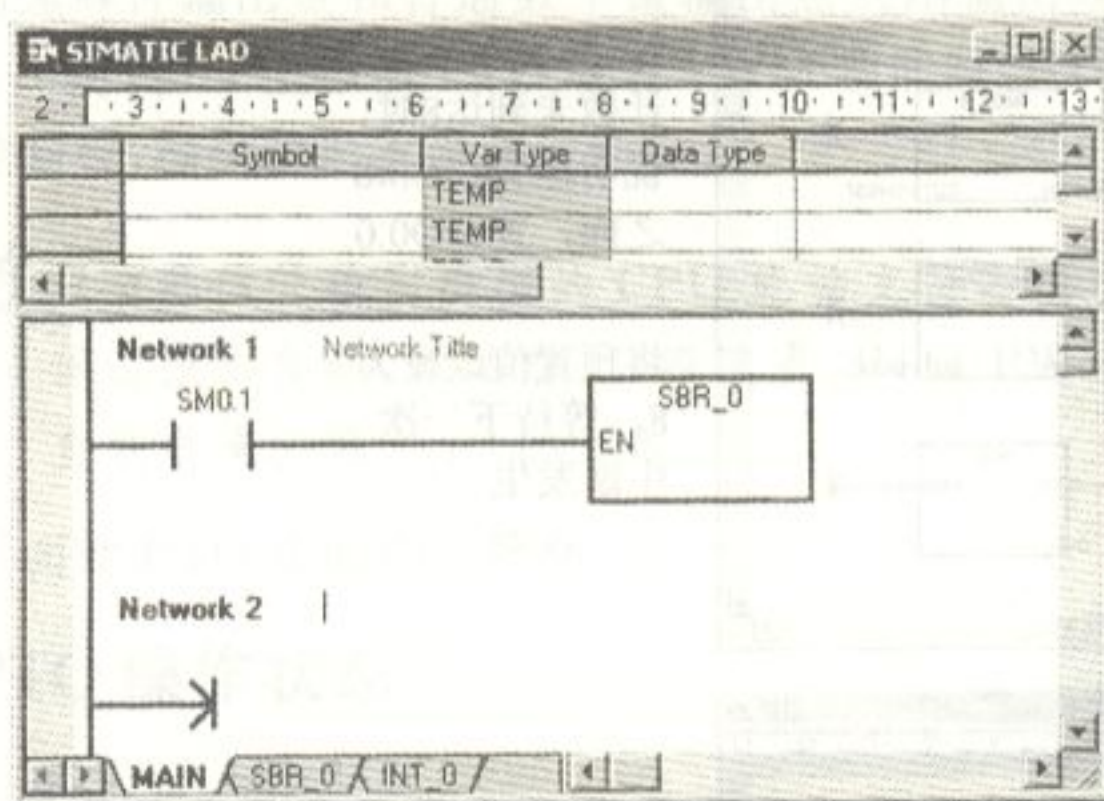
主程序: 一次性调用 HSC0 初始化子程序 SBR_0。

SBR_0: 初始化 HSC0 为模式 10, 设预置值为 5, 并连接中断事件 12 (HSC0 的 $CV = PV$) 到 INT_0。



INT_0:根据计数值置位 Q0.0,并重设预置值。

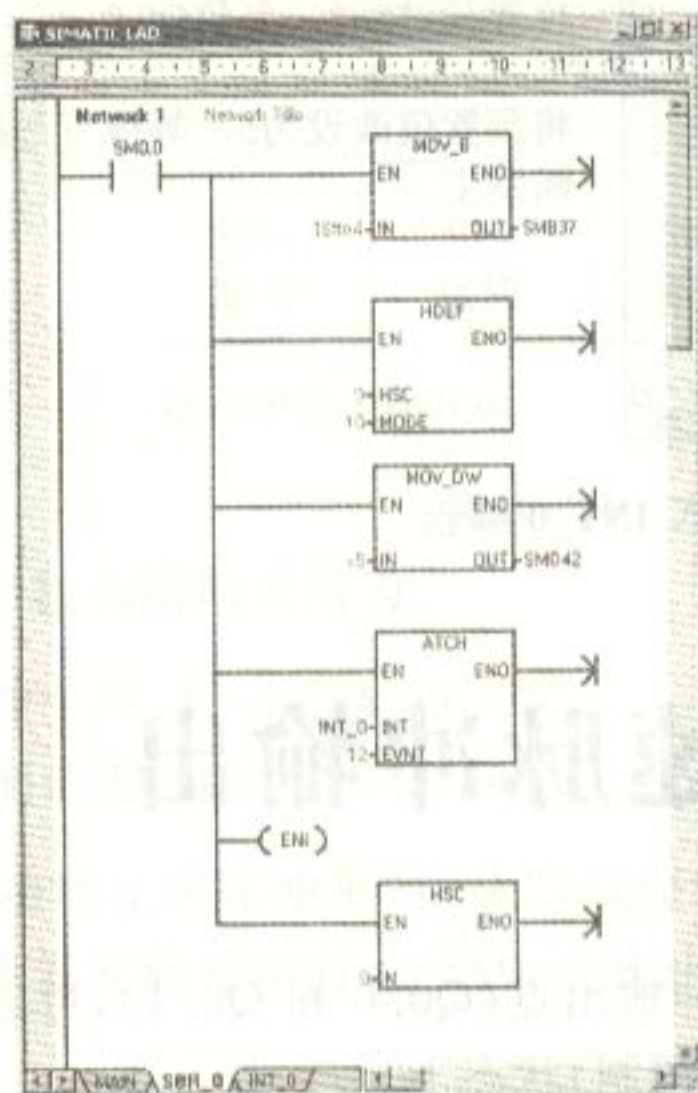
☞ 主程序(如图 3-51 所示):



运行时调用一次初始化子程序

图 3-51 主程序

SBR_0 编程(如图 3-52 所示):



写HSC0的控制字节

定义HSC0为模式10

装入预置值5

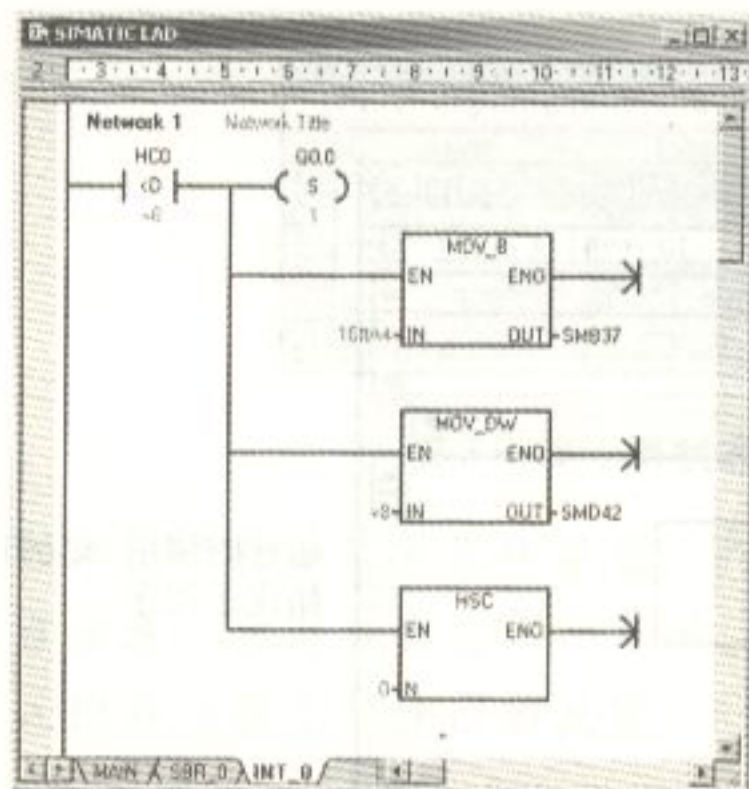
连接中断事件和INT_0, 开中断

写入HSC0设置

图 3-52 SBR_0 编程

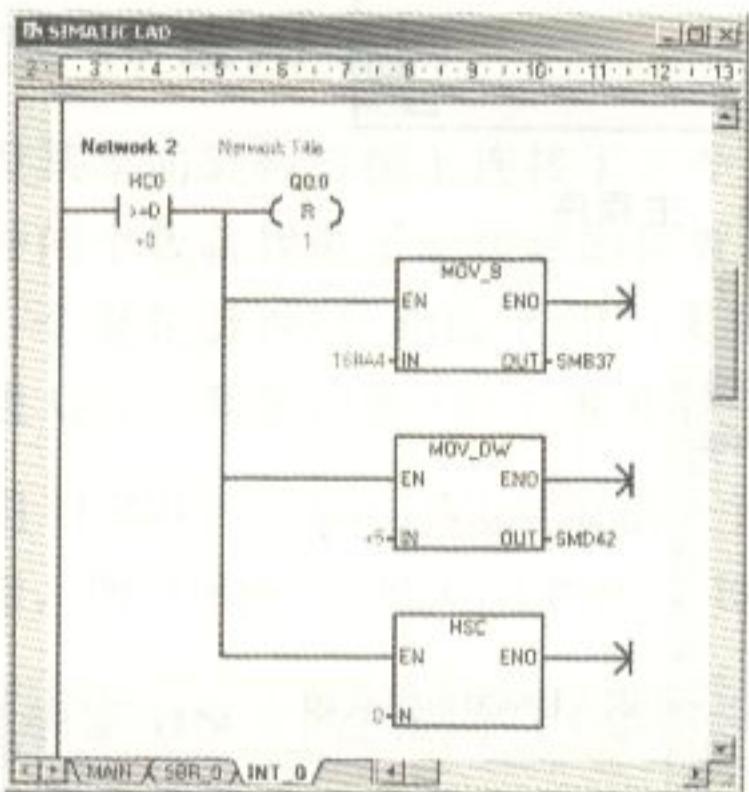


INT_0 编程如图 3-53 所示。



计数未到达8时，
说明位置在5和8
之间，置位Q0.0

将预置值改设为
8，等待下一次
中断发生



位置超过8时，复位Q0.0

将预置值改设为5，等待下一次中
断发生

图 3-53 INT_0 编程

3.6 高速脉冲输出


S7-200 CPU 提供两个高速脉冲输出点(Q0.0 和 Q0.1)，可以分别工作在 PTO(脉冲串输出)和 PWM(脉宽调制)状态下。

PTO 可以输出一串脉冲，用户可以控制脉冲的周期(频率)和个数。



PWM 可以连续输出一串占空比可调的脉冲,用户可以控制脉冲的周期和脉宽(占空比)。

高速脉冲输出点和普通数字量输出点共用输出映象 Q0.0 和 Q0.1。当在 Q0.0 或 Q0.1 上激活 PTO 或者 PWM 功能时,PTO/PWM 发生器对输出拥有控制权,输出波形不受其他影响。

 只有晶体管输出类型的 CPU 能够支持高速脉冲输出功能。为保证波形良好,脉冲前、后沿陡直,PTO/PWM 在高电平输出时负载电流必须大于 140 mA。

下面就 PTO 功能作一简介。

PTO 操作状态


 PTO 按照给定的脉冲个数和周期输出一串方波(占空比 50%)(如图 3-54 所示)。




图 3-54 方波输出

脉冲个数和周期范围为:

- 脉冲个数: 1~4 294 967 295;
- 周期范围: 50~65 535 μ s 或 2~65 535 ms。

PTO 功能允许脉冲串“排队”,以保证脉冲输出的连续进行。PTO 功能也支持在未发完脉冲串时,立刻中止脉冲输出。

 如果要控制输出脉冲的频率(如步进电机的速度/频率控制),须将频率换算为周期。为保证占空比为 50%,请设定周期值为偶数。



PTO 支持两种工作模式:

- 单段管线:每次用特殊寄存器设定规格后输出一个脉冲串。单段管线支持排队,可以在发送当前脉冲串时,为下一个脉冲串重新定义特殊寄存器。队列中只能有一个脉冲串在等待。
- 多段管线:CPU 自动从 V 存储器区的包络表中读出多个脉冲串的特性并顺序发送脉冲。包络表使用 8 个字节保存一个脉冲串的属性,包括一个字长的起始周期值,一个字长的周期增量值和一个双字长的脉冲个数。一个包络表可以包含 1~255 个脉冲串。

PTO 的控制

☞ S7-200 使用两套特殊寄存器控制两个 PTO 独立工作。其定义如表 3-13 所列。

表 3-13 PTO 控制和状态寄存器

Q0.0	Q0.1	状态字节
SM66.4	SM76.4	PTO 包络由于增量计算错误而终止 0=无错误;1=终止
SM66.5	SM76.5	PTO 包络由于用户命令而终止 0=无错误;1=终止
SM66.6	SM76.6	PTO 管线上溢/下溢 0=无上溢;1=上溢/下溢
SM66.7	SM76.7	PTO 空闲 0=执行中;1=PTO 空闲
Q0.0	Q0.1	控制字节
SM67.0	SM77.0	PTO/PWM 更新周期值 0=不更新;1=更新周期值
SM67.1	SM77.1	PWM 更新脉冲宽度值 0=不更新;1=脉冲宽度值
SM67.2	SM77.2	PTO 更新脉冲数 0=不更新;1=更新脉冲数
SM67.3	SM77.3	PTO/PWM 时间基准选择 0=1/时基;1=1 ms/时基
SM67.4	SM77.4	PWM 更新方法:0=异步更新;1=同步更新
SM67.5	SM77.5	PTO 操作:0=单段操作;1=多段操作
SM67.6	SM77.6	PTO/PWM 模式选择 0=选择 PTO;1=选择 PWM
SM67.7	SM77.7	PTO/PWM 允许 0=禁止 PTO/PWM;1=允许 PTO/PWM
Q0.0	Q0.1	其他 PTO/PWM 寄存器
SMW68	SMW78	PTO/PWM 周期值(范围:2~65 535)
SMW70	SMW80	PWM 脉冲宽度值(范围:0~65 535)
SMD72	SMD82	PTO 脉冲计数值(范围:1~4 294 967 295)
SMB166	SMB176	进行中的段数(仅用在多段 PTO 操作中)
SMW168	SMW178	包络表的起始位置,用从 V0 开始的字节偏移表示(仅用在多段 PTO 操作中)



在上表中, SMB66 为 Q0.0 的状态字节, 如果 SM66.7 为 1, 就说明 PTO 执行中。MB67 为 Q0.0 的控制字节, 如需在下次 PTO 输出时更新周期, 就需要将 SM67.1 置为“1”, 然后再将周期值装入 SMW68 中。根据上表, 我们可以得出控制字节取值快速参考表, 即表 3-14。

表 3-14 PTO 控制字节取值快速参考表

控制寄存器 (16 进制)	执行 PLS 指令的结果					
	允 许	模式选择	PTO 段操作	时 基	脉冲数	周 期
16 # 81	Yes	PTO	单段	1 μ s/周期		装入
16 # 84	Yes	PTO	单段	1 μ s/周期	装入	
16 # 85	Yes	PTO	单段	1 μ s/周期	装入	装入
16 # 89	Yes	PTO	单段	1 ms/周期		装入
16 # 8C	Yes	PTO	单段	1 ms/周期	装入	
16 # 8D	Yes	PTO	单段	1 ms/周期	装入	装入
16 # A0	Yes	PTO	多段	1 μ s/周期		
16 # A8	Yes	PTO	多段	1 ms/周期		

在上表中, 如果我们希望在设置单段 PTO 脉冲时只装入周期值, 就可以取值 16 # 81; 需要同时装入周期和脉冲个数时, 就可以取值 16 # 85。

PTO 编程

对单段管线, 可在主程序中调用初始化子程序。在子程序中:

- 设置 PTO/PWM 控制字节;
- 写入周期值;
- 写入脉冲串计数值;
- 连接中断事件和中断服务程序, 允许中断(可选);
- 执行 PLS 指令, 使 S7-200 CPU 对 PTO 硬件编程。

如果要修改 PTO 的周期、脉冲数, 可以在子程序或者中断程序中执行:

- 根据要修改的内容, 写入相应的控制字节值;
- 写入新的周期、脉冲数;



- 执行 PLS 指令,使 S7-200 CPU 确认设置。

对于多段 PTO 操作,可在主程序中调用初始化子程序。在子程序中:

- 设置控制字节,选择多段操作;
- 写入包络表起始地址到相应特殊寄存器(包络表的具体内容可另行计算、编写);
- 连接中断事件和程序,允许中断(可选);
- 执行 PLS 指令,使 S7-200 CPU 确认设置。

如果要在脉冲输出执行过程中,停止脉冲输出:

- 设置控制字节,将 PTO/PWM 使能位置为 0;
- 执行 PLS 指令,使 CPU 确认。

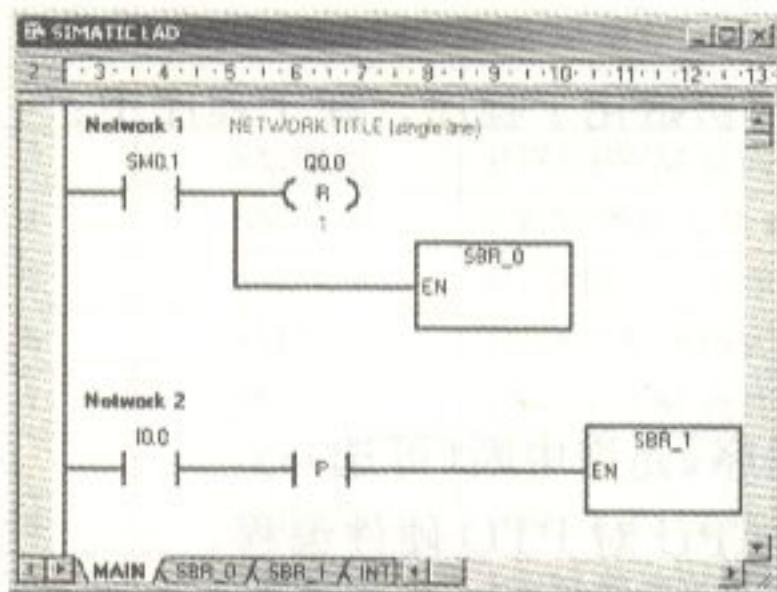
单段 PTO 编程举例

主程序:一次性调用初始化子程序 SBR_0;I0.0 接通时调用 SBR_1,改变脉冲周期;

SBR_0:设定脉冲个数、周期并发出起始脉冲串;

SBR_1:改变脉冲串周期。

☞ 主程序(如图 3-55 所示):



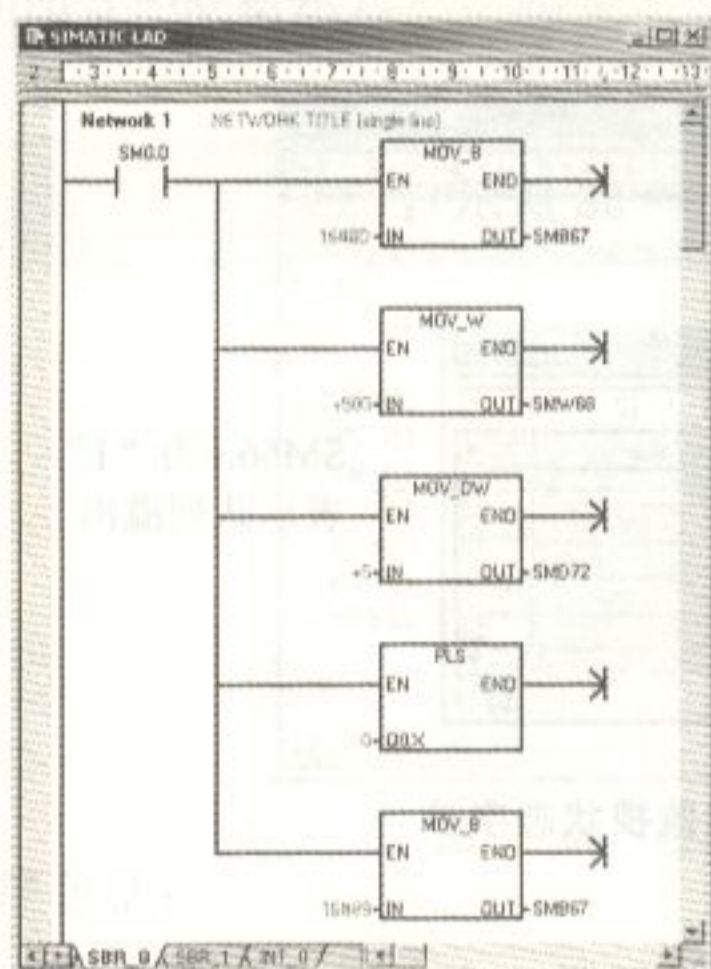
CPU运行时一次性清除输出映象区,调用初始化子程序SBR_0

I0.0接通时调用SBR_1

图 3-55 主程序



☞ SBR_0 编程(如图 3-56 所示):



控制字节写入16#8D, 准备装入周期值和脉冲个数

装入周期值500 ms

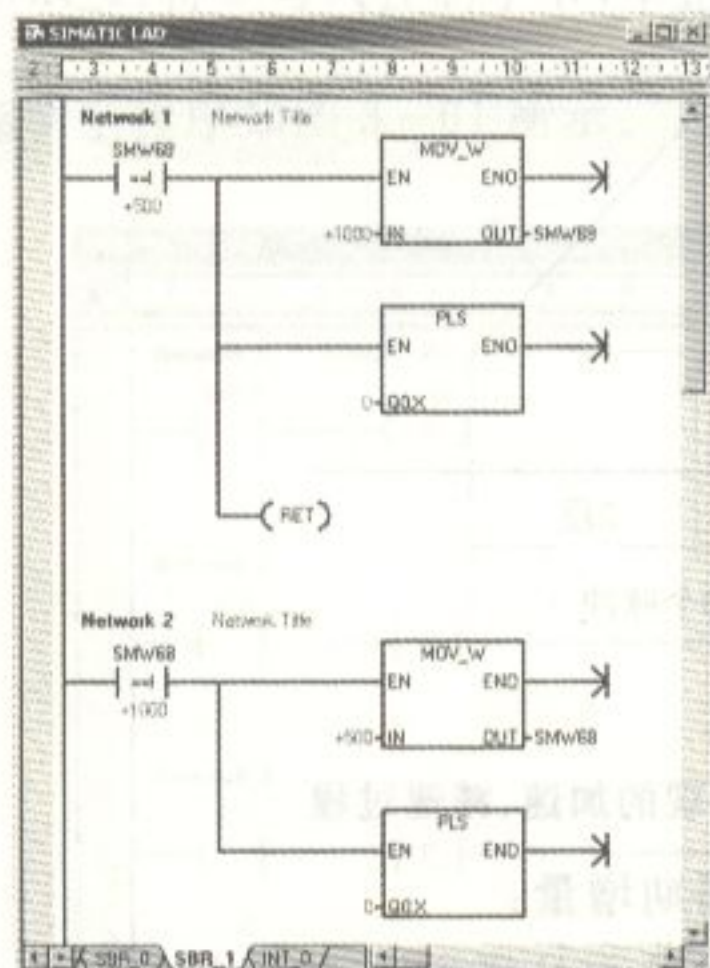
装入脉冲个数5

执行PLS, 编程Q0.0为PTO模式

再写入控制字节16#89, 为更改周期值做准备(以后在SBR_1中只更改周期值, 而不更改脉冲数)

图 3-56 SBR_0 编程

☞ SBR_1 编程(如图 3-57 所示):



如果当前周期为500 ms, 将其改为1 000 ms, 写入特殊寄存器

执行PLS指令, 确认更改生效并发出脉冲串

从子程序返回

如果当前周期为1 000 ms, 将其改为500 ms, 写入特殊寄存器

执行PLS指令, 确认更改生效并发出脉冲串

图 3-57 SBR_1 编程



在程序执行时,可以尝试在当前脉冲串没有结束再次接通 I0.0,观察脉冲串的排队。当前脉冲串结束时,第二串立刻发出。如果连续多次触发 I0.0,会造成队列溢出。

可用状态图观察状态字节(如图 3-58 所示):

Status Chart											
3	4	5	6	7	8	9	10	11	12	13	1
	Address			Format			Current Value				
1	SMB66			Binary			2#0100 0000				
2				Signed							
3	SMW68			Signed			+1000				
4				Signed							
5	I0.0			Bit			2#1				
CHT1											

SM66.6为“1”
表示队列溢出

图 3-58 用状态图监视状态字节

多段 PTO 编程举例

使用多段 PTO 实现一个两段加速、减速过程(如图 3-59 所示)。先将脉冲频率从 1 Hz(周期 1 000 ms)上升到 2 Hz(周期 500 ms),然后再从 2 Hz 降回到 1 Hz,总共在 100 个脉冲内完成。

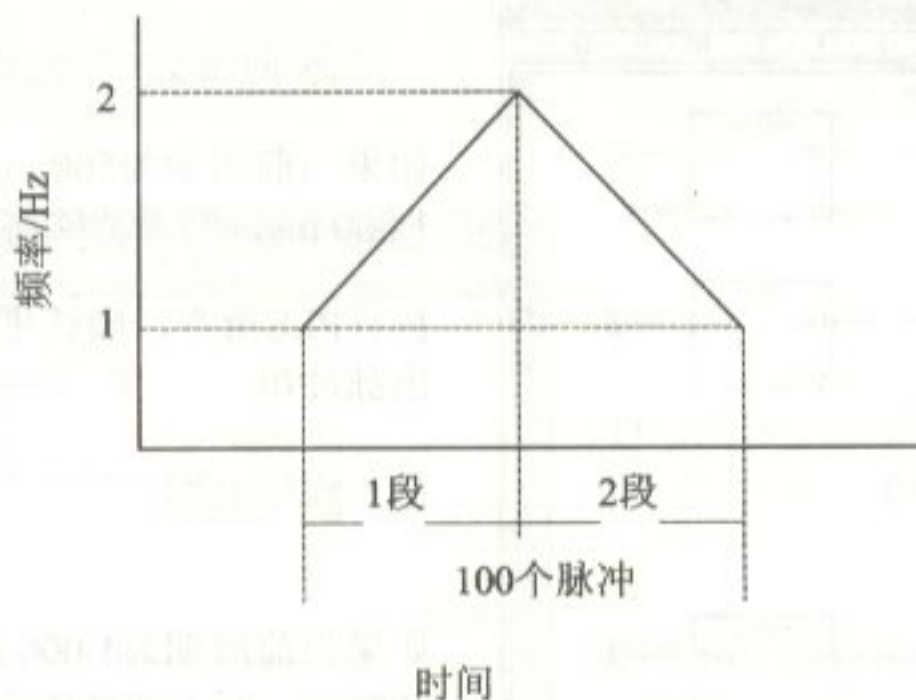


图 3-59 多段 PTO 实现的加速、减速过程

根据公式准备包络表,求得每段的周期增量:

$$\text{段周期增量} = |\text{段终止周期} - \text{段初始周期}| / \text{脉冲数量}$$

可以求出两段脉冲的周期增量为 -10 和 +10。



☞ 把包络表保存在以 VB300 开始的 V 存储区内,我们用数据块直接写入(如图 3-60 所示)。

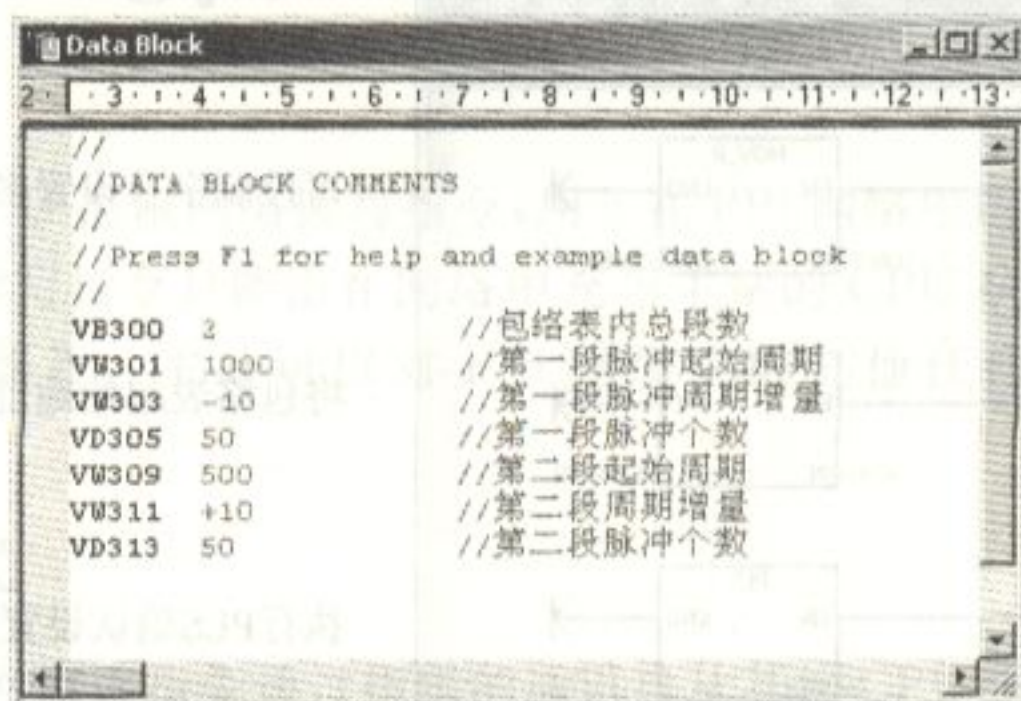


图 3-60 在数据块中写入包络表

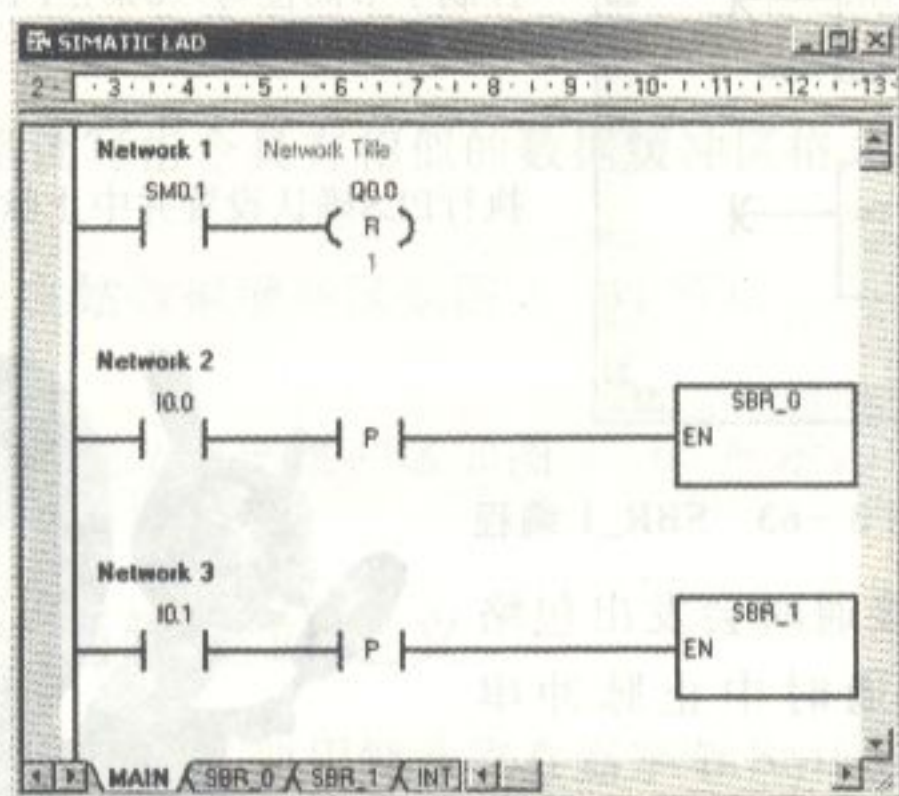
该程序包括:

主程序:清除 Q0.0 输出映象区,在 I0.0 接通时调用 SBR_0, I0.1 接通时调用 SBR_1;

SBR_0:设置多段 PTO 控制字节和相关参数;

SBR_1:将 0 写入控制字节,中止 PTO 输出。

☞ 主程序如图 3-61 所示。



清除Q0.0输出映象区

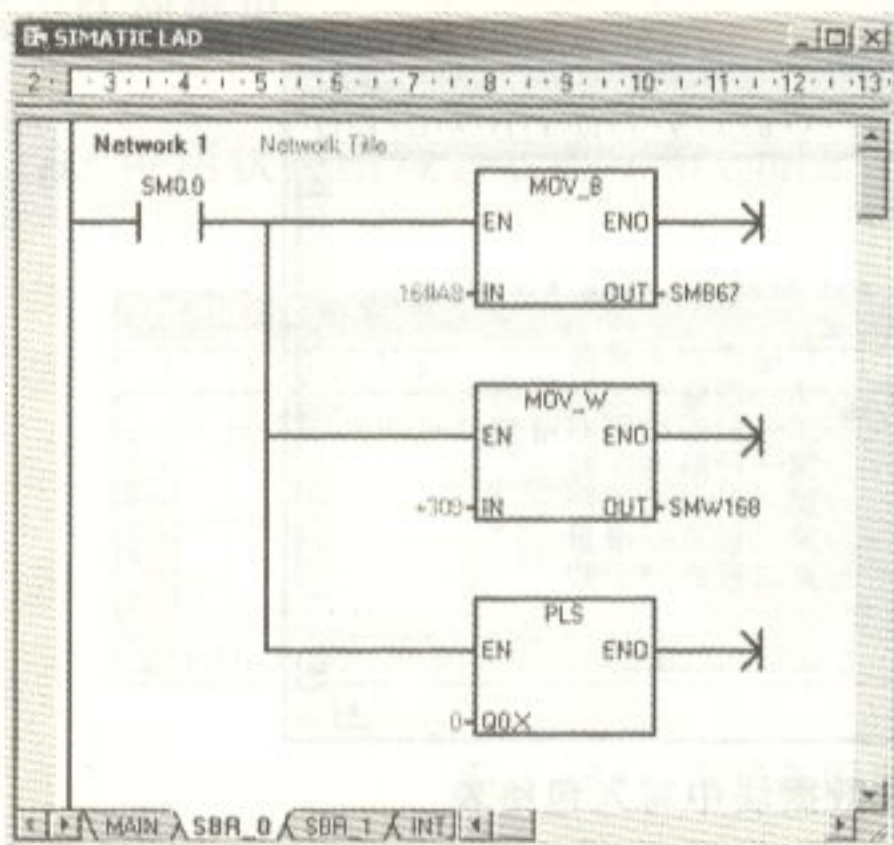
I0.0接通时调用SBR_0

I0.1接通时调用SBR_1

图 3-61 主程序



☞ SBR_0 编程如图 3-62 所示。



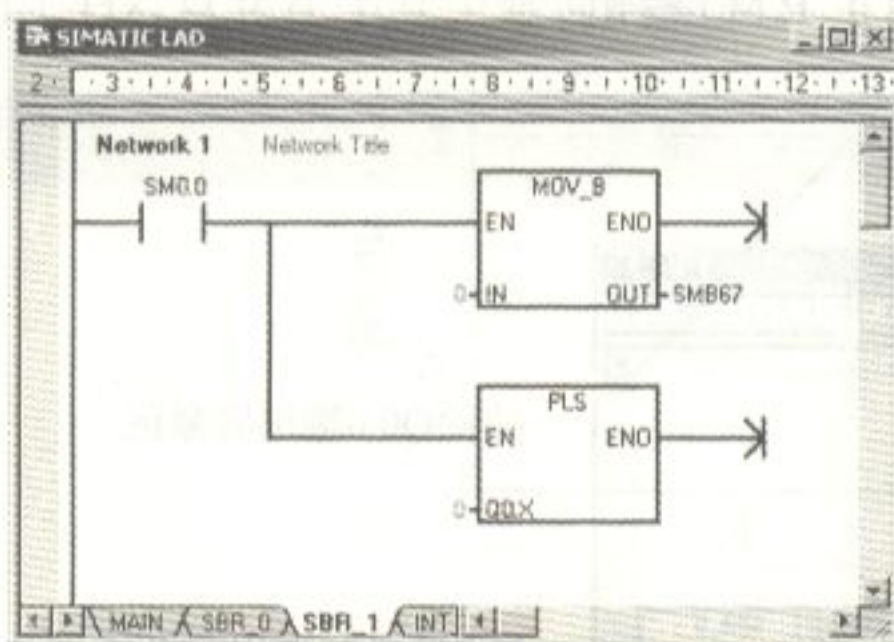
写控制字节设置多段PTO

将包络表起始地址写入SMW168

执行PLS确认设置并发出脉冲串

图 3-62 SBR_0 编程

☞ SBR_1 编程如图 3-63 所示。



控制字节高位写入0禁止PTO

执行PLS确认设置并中止脉冲串

图 3-63 SBR_1 编程

下载并运行程序。I0.0 接通时会发出包络表中定义的脉冲串；I0.1 接通时中止脉冲串输出。





3.7 网络读写

S7-200 CPU 提供网络读写指令,用于在 PPI 网络中的 S7-200 CPU 之间通讯。网络读写指令只能由在网络中充当主站的 CPU 执行;从站 CPU 不必专门做通讯编程。主站可以对 PPI 网络中的其他任何 CPU 进行网络读写。

指令格式

- NETR:网络读指令通过指定的通讯口从其他 CPU 中指定地址的数据区读取最多 16 个字节的信息,存入本 CPU 中指定地址的数据区。
- NETW:网络写指令通过指定的通讯口,把本 CPU 中指定地址的数据区内容写到其他 CPU 中指定地址的数据区内,最多可以写入 16 个字节。

✌ 缺省情况下,S7-200 CPU 工作在 PPI 从站模式,要执行网络读写指令,必须用程序把 CPU 设置为 PPI 主站模式。

✌ 同一个 CPU 内可以有任意条网络读写指令,但同一时刻只能有最多 8 条网络读或写指令激活。建议必要时使用相关的读写指令标志位控制指令间的切换。

网络读写指令具有相似的数据缓冲区格式,缓冲区以一个状态字节起始。

☞ 主站数据缓冲区如图 3-64 所示。

☞ 远程站数据缓冲区如图 3-65 所示。

PPI 通讯主站定义

S7-200 CPU 使用特殊寄存器字节 SMB30(对 Port0,端口 0)和 SMB130(对 Port1,端口 1)定义通讯口。

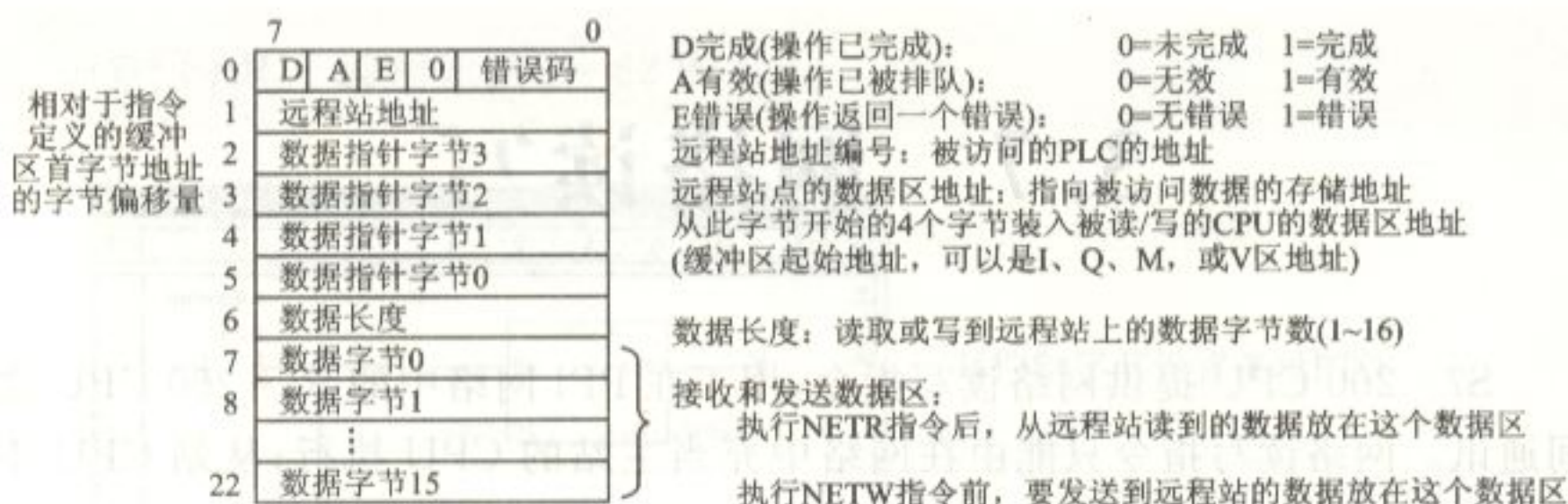


图 3-64 主站数据缓冲区

相对于主站网络读写指令定义的远程站缓冲区首字节地址的字节偏移量

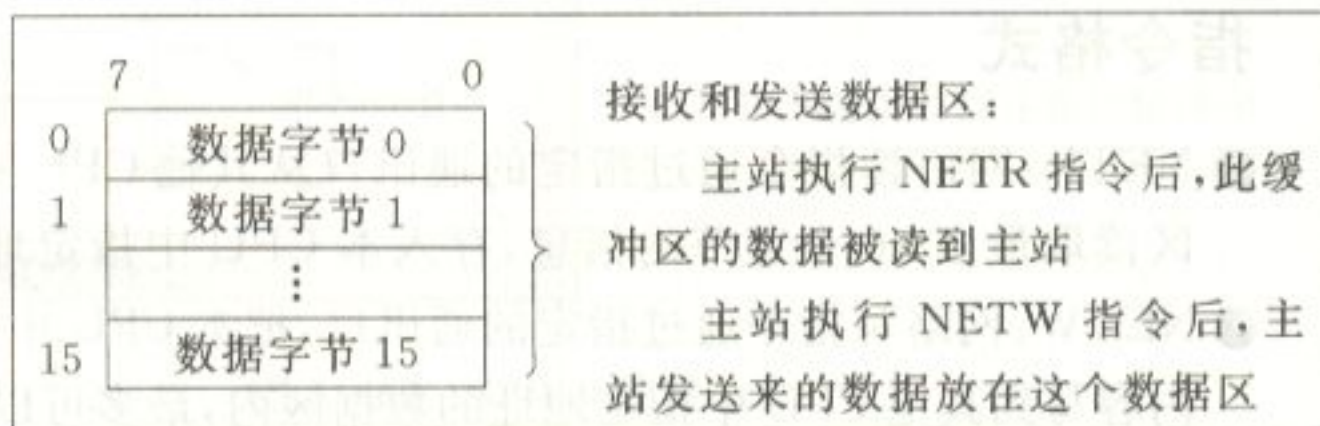
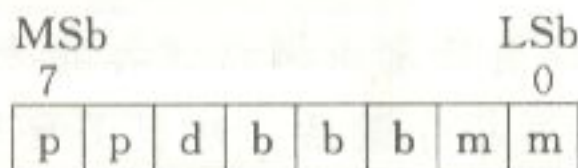


图 3-65 远程站数据缓冲区

控制位定义:



控制字节的最低两位, 即 mm 用来决定相应通讯口的工作模式。其中:

- mm=00: PPI 从站模式(缺省设置为 PPI/从站模式);
- mm=01: 自由口模式;
- mm=10: PPI 主站模式。

所以, 只要向 SMB30 或 SMB130 中写入数值 2(即二进制的 10), 就可以将通讯口设置为 PPI 主站模式。PPI 通讯速率在“系统块”中设置。

网络读写指令编程举例

在下面的例子中, CPU 226 和 CPU 224 连成一个 PPI 网络, 其中,



CPU 226 为主站, CPU 224 为从站。我们的任务是把 CPU 226 内 V 存储区保存的时钟信息用网络写指令写入 CPU 224 的 V 存储区;把 CPU 224 内 V 存储区保存的时钟信息读取到 CPU 226 的 V 存储区。在两个 CPU 中,分别编程把对方实时时钟的“秒”信息以 BCD 格式,传送到自身开关量输出字节 QB0 显示。

在组成 PPI 网络时,将 CPU 226 的 Port1 和 CPU 224 的 Port0 用网络连接器和 Profibus 电缆连接起来。分别通过 CPU 226 和 CPU 224 的系统块,设置它们的 PPI 网络地址。保持 CPU 226 的两个通讯口地址为 2,将 CPU 224 的地址设置为 3。



CPU 226/CPU 226 XM 的两个通讯口可以各自定义其功能。在这个例子中 Port1 用于 PPI 网络通讯,Port0 可用于 Step7 - Micro/WIN32 监控。

规划通讯数据缓冲区(如表 3-15、表 3-16 所列):

表 3-15 CPU226 发送缓冲区

VB200	状态字节
VB201	CPU 224 地址(3)
VD202	&VB400
	CPU224 接收缓冲区地址
VB206	8(字节数)
VB207	CPU226 时钟信息:“年”
VB208	“月”
VB209	“日”
VB210	“时”
VB211	“分”
VB212	“秒”
VB213	“0”
VB214	“星期”



表 3-16 CPU224 接收缓冲区

	CPU 226 时钟信息
VB400	“年”
VB401	“月”
VB402	“日”
VB403	“时”
VB404	“分”
VB405	“秒”
VB406	“0”
VB407	“星期”

CPU 226(主站)编程(如表 3-17、表 3-18 所列):



表 3-17 CPU226 接收缓冲区

VB300	状态字节
VB301	CPU224 地址(3)
VB302	&VB300
	CPU224 发送缓冲区地址
VB306	8(字节数)
VB307	CPU 224 时钟信息:“年”
VB308	“月”
VB309	“日”
VB310	“时”
VB311	“分”
VB312	“秒”
VB313	“0”
VB314	“星期”

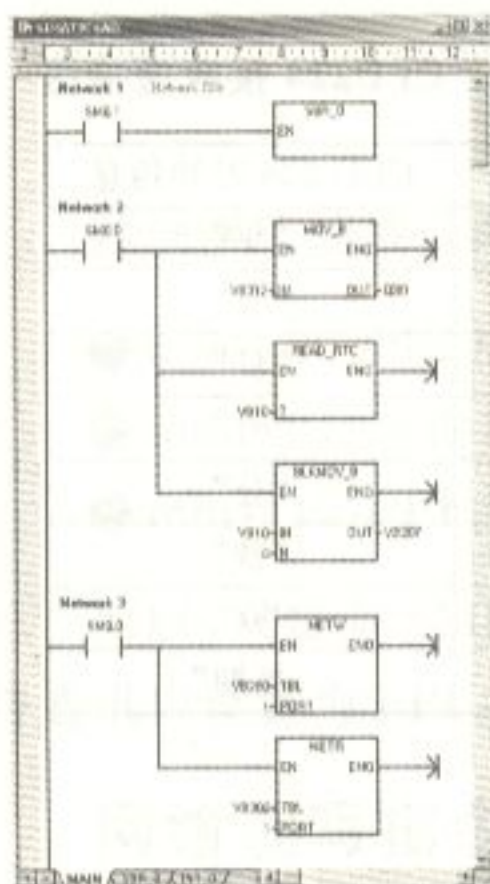
表 3-18 CPU224 发送缓冲区

	CPU 224 时钟信息
VB300	“年”
VB301	“月”
VB302	“日”
VB303	“时”
VB304	“分”
VB305	“秒”
VB306	“0”
VB307	“星期”



主程序:调用通讯初始化子程序,读取本 CPU 的实时时钟信息,执行网络读写指令(如图 3-66 所示);

SBR_0:初始化通讯口,为网络读写指令准备数据缓冲区。



一次性调用通讯初始化子程序

将网络读取指令缓冲区的一个字节(秒)送往输出字节QB0

读取本机实时时钟

块传送指令将时钟信息的8个字节一次传送到网络写指令的缓冲区

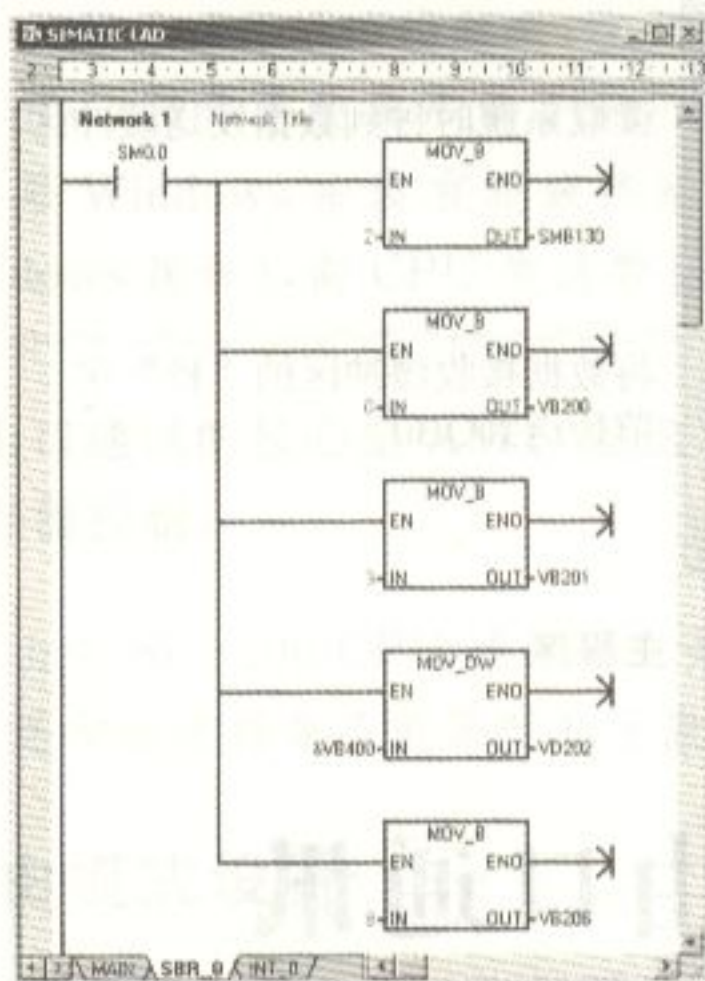
连续执行网络写指令

连续执行网络读指令

图 3-66 主程序



☞ CPU 226 的 SBR_0 编程如图 3-67 所示。



将CPU 226的Port1定义为PPI主站通讯口

清除网络写缓冲区状态字节

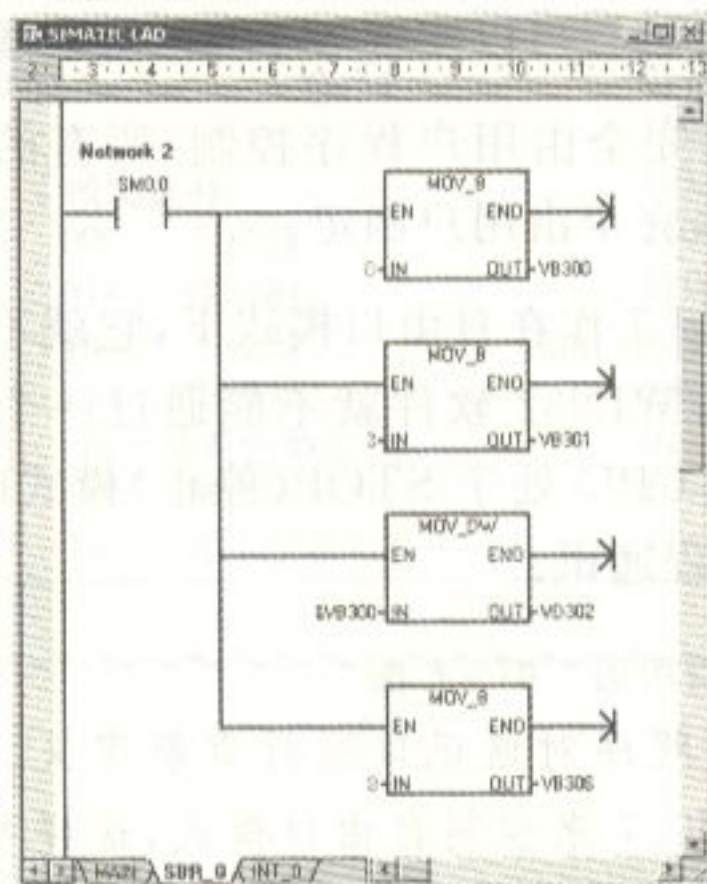
装入远程站地址

装入CPU 224数据输入缓冲区地址

装入传送字节数

图 3-67 SBR_0 编程

☞ CPU 226 的 SBR_0 编程(续)如图 3-68 所示。



清除网络读指令缓冲区状态字节

装入远程站地址

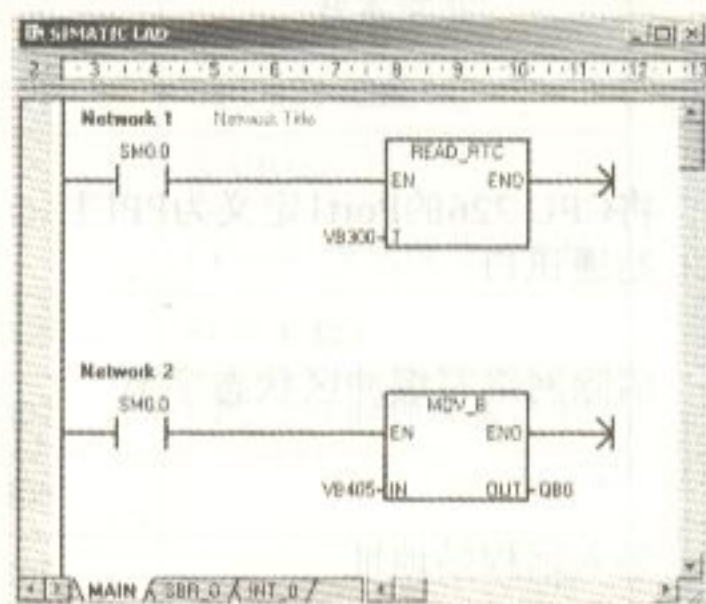
装入CPU 224数据输出缓冲区地址

装入读取字节数

图 3-68 SBR_0 编程(续)



☞ CPU 224 主程序如图 3-69 所示。



读取系统时钟到数据发送缓冲区

将数据接收缓冲区的“秒”字节传送到QB0

图 3-69 主程序

3.8 自由口通讯

S7-200 CPU 拥有自由口通讯能力。自由口通讯是建立在 RS-485 硬件基础上的一种通讯方式,它允许用户自己定义一些简单、基本的通讯协议设置,如数据长度、奇偶校验等等。灵活运用自由口,可以实现比较复杂的通讯功能,以适应各种通讯协议。例如,通过编程可以实现 Modbus 协议通讯。

处于自由口通讯模式时,通讯功能完全由用户程序控制,所有的通讯任务,必须由用户编程完成。信息的定义,完全由用户制定。

如果 S7-200 CPU 的某个通讯端口工作在自由口模式下,它就不能用于其他模式的通讯。例如,Step7-Micro/WIN32 软件就不能通过一个定义为自由口模式的通讯口与 CPU 通讯。当 CPU 处于 STOP(停止)模式时,自由口便不能工作,从而可以建立正常的编程通讯。



在 CPU 运行状态下,可以通过程序对通讯口进行重新定义。例如,可以使用特殊寄存器位 SM0.7 来控制自由口模式,这样可以在 CPU 处于运行模式时,使用 Step7-Micro/WIN32 软件监控。



调试 S7-200 CPU 的自由口通讯时, 可以用 PC/PPI 电缆将 CPU 和 PC 机连接起来, 在 PC 机上运行串口调试软件, 如 Windows 操作系统集成的 HyperTerminal(超级终端)应用程序(如果 Windows 中没有超级终端程序, 您可能需要添加安装 Windows 组件), 向 CPU 发送数据, 或从 CPU 接收数据。

自由口通讯的核心是 XMT(发送)和 RCV(接收)两条指令, 以及相应的特殊寄存器控制。



由于 S7-200 CPU 通讯端口是 RS-485 半双工通讯口, 因此发送和接收指令不能同时处于激活状态。

自由模式设置

S7-200 CPU 使用 SMB30(对 Port0)和 SMB130(对 Port1)定义通讯口的工作模式(如图 3-70 所示)。

MSb 7							LSb 0	bbb: 自由口波特率	
P	P	d	b	b	b	m	m	000=38 400波特率 ¹	S7-200 CPU版本在 1.2以上支持57.6 K波特率和 115.2 K波特率。
								001=19 200波特率	
								010=9 600波特率	
								011=4 800波特率	
SMB30=	端口0							100=2 400波特率	
SMB130	= 端口1							101=1 200波特率	
PP:	校验选择							110=115.2K波特率 ¹	
	00= 不校验							111=57.6K波特率 ¹	
	01= 偶校验								
	10= 不校验								
	11= 奇校验								
d:	每个字符的数据位							mm: 协议选择	默认设置为PPI/从站模式
	0= 每个字符8位							00=PPI/从站模式	
	1= 每个字符7位							01=自由口模式	
								10=PPI/主站模式	
								11=保留	

图 3-70 通讯口工作模式的定义



3.8.1 发送指令

XMT(发送指令)向指定通讯口以字节为单位发送一串数据字符,要发送的字符以数据缓冲区指定,一次发送的字符最多为 255 个。

发送指令执行完成后,会产生一个中断事件(Port0 为中断事件 9,对 Port1 为中断事件 26)。在 SMB4 中也有相应的位对应于发送指令状态(SM4.5 置位对应 Port0 空闲,SM4.6 对应 Port1 空闲)。

XMT 指令缓冲区格式(如表 3-19 所列)

表 3-19 XMI 指令缓冲区格式

T+0	发送字节的个数
T+1	数据字节
T+2	数据字节
T+3	数据字节
⋮	⋮
T+255	数据字节

XMT 指令编程举例

本例把 CPU 224 的 Port0 定义为自由口通讯模式。在一个定时中断程序中对定时中断次数计数,并将计数值转换为 ASCII 字符串,再从 Port0 发送出去。我们还将使用 HyperTerminal 显示 S7-200 CPU 的通讯内容。




自由口通讯模式以字节为单位发送数据,而不考虑其表示形式。在这个例子中使用 ASCII 字符只是为了便于在 PC 机上显示。





我们规定发送缓冲区从 VB100 开始(如表 3-20 所列):

表 3-20 发送缓冲区

VB100	14	发送数据字节数
VB101	.	12 字节
VB102	.	ASCII 字符串
VB103	.	
VB104	.	
VB105	.	
VB106	.	
VB107	.	
VB108	.	
VB109	.	
VB110	.	
VB111	.	
VB112	.	
VB113	16 # 0D	消息结束字符
VB114	16 # 0A	即“回车”符

 如果通讯对象需要固定的起始或者结束字符,也须在发送缓冲区中设置。

 在本例中设置 16 # 0D0A 为结束字符,是因为在 HyperTerminal (超级终端)中 16 # 0D0A 正好是字符“回车”,可用来换行显示。

 使用 Data Block(数据块)定义发送缓冲区(如图 3-71 所示):

程序结构如下:

主程序:初始化自由口通讯设置,并根据“模式选择开关”的状态重新设置通讯端口 0

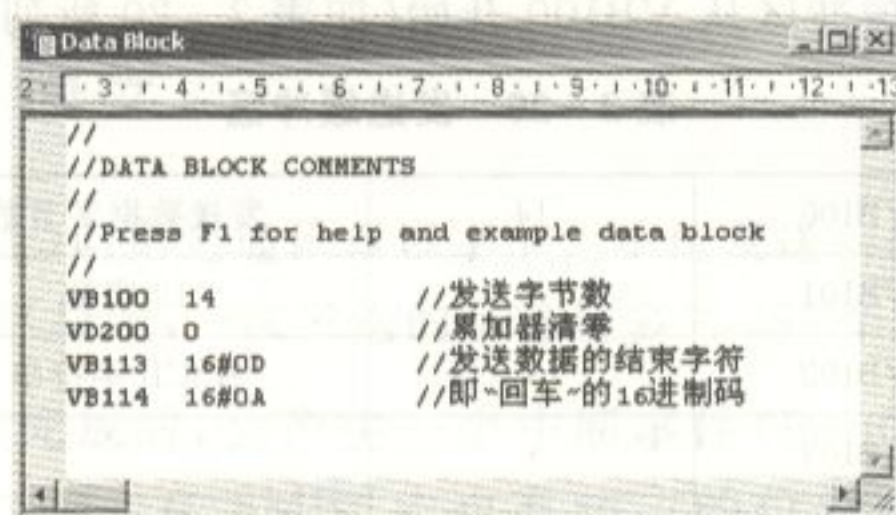


图 3-71 发送缓冲区的定义

SBR_0: 定义通讯端口 0 为自由口, 初始化定时中断

SBR_1: 定义通讯端口 0 为普通 PPI 从站通讯口

INT_0: 对定时中断计数并从 Port0(端口 0)发送计数值

☞ 主程序编程(如图 3-72 所示):



初始化自由通讯口

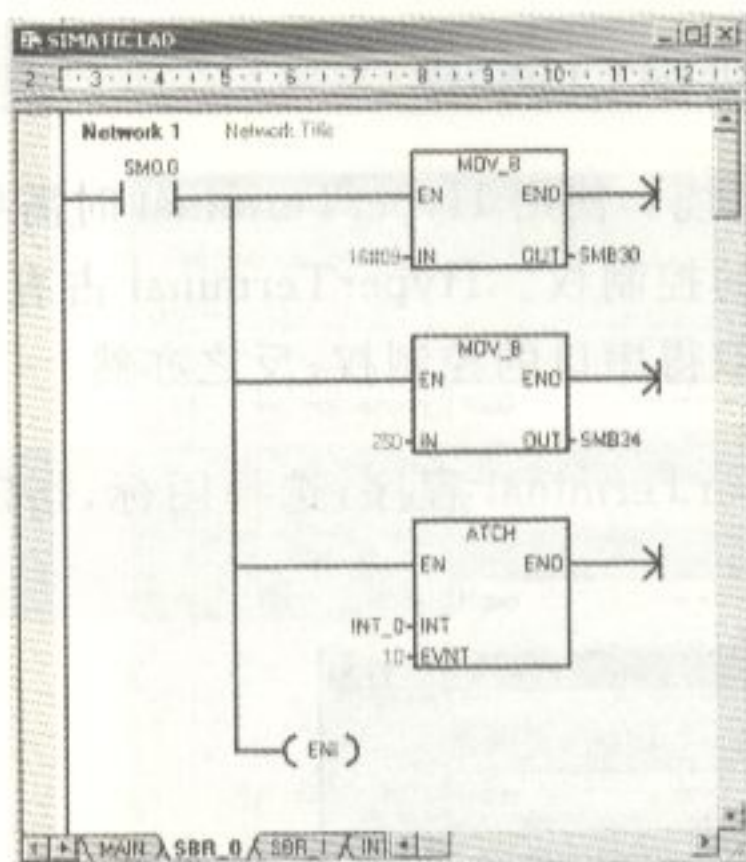
恢复普通PPI通讯口设置

图 3-72 主程序编程

☞ SBR_0 编程如图 3-73 所示。

☞ SBR_1 编程如图 3-74 所示。

☞ INT_0 编程如图 3-75 所示。



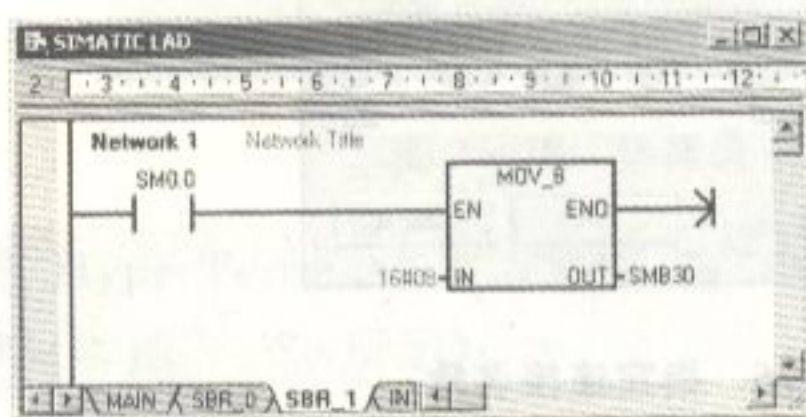
定义通讯口0为自由口模式, 16#09定义自由口工作在9.6 K波特率、无校验、8数据位模式下

写入定时中断周期250 ms

连接定时中断事件10到中断服务程序0

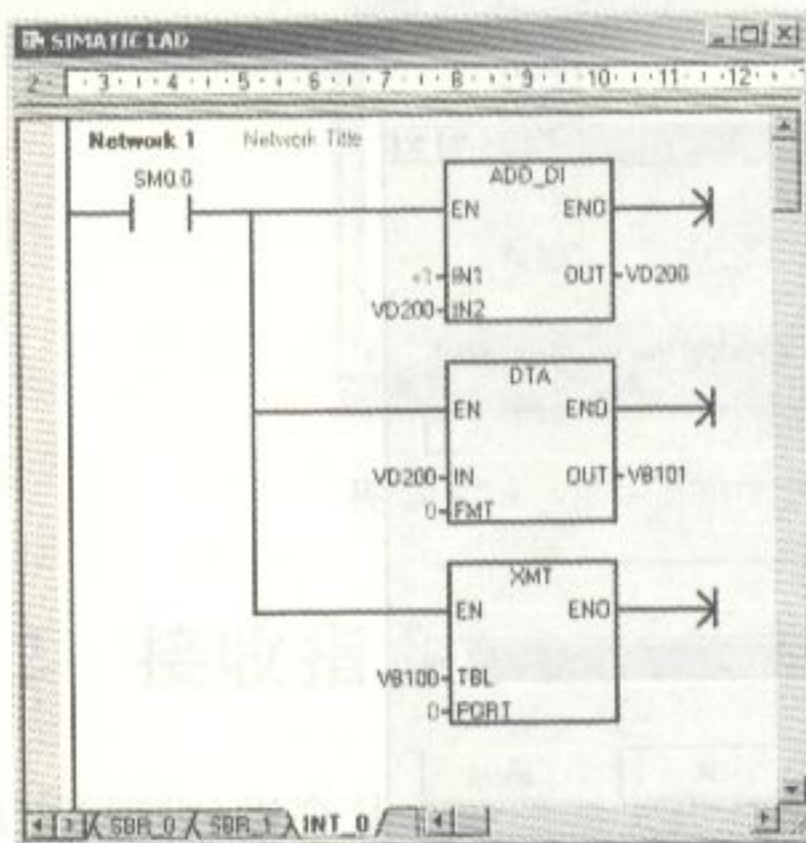
允许中断

图 3-73 SBR_0 编程



设置端口0为PPI从站模式

图 3-74 SBR_1 编程



VD200用作累加器, 每次中断加1

将VD200内的整数转换为12个ASCII字符并填入发送缓冲区

从Port0发送缓冲区内容

图 3-75 INT_0 编程



使用 HyperTerminal 监视串口通讯

如同编程时那样连接 PC/PPI 电缆。使用 HyperTerminal 时需要注意不要让多个应用程序争夺串行通讯口的控制权。HyperTerminal 占有 COM 口时, Step7 - Micro/WIN32 就不能再取得串口的控制权;反之亦然。

☞ 打开 Windows 系统的 HyperTerminal 程序, 选择图标, 指定一个连接名称, 如图 3-76 所示。

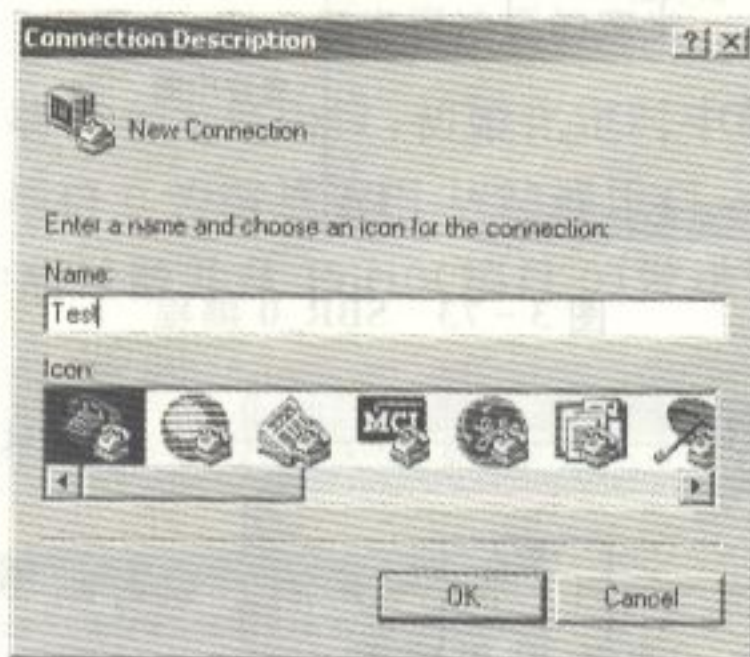


图 3-76 指定连接名称

☞ 选择连接到 PC 机连接 PC/PPI 电缆的串行通讯端口(这里是 COM1), 如图 3-77 所示。

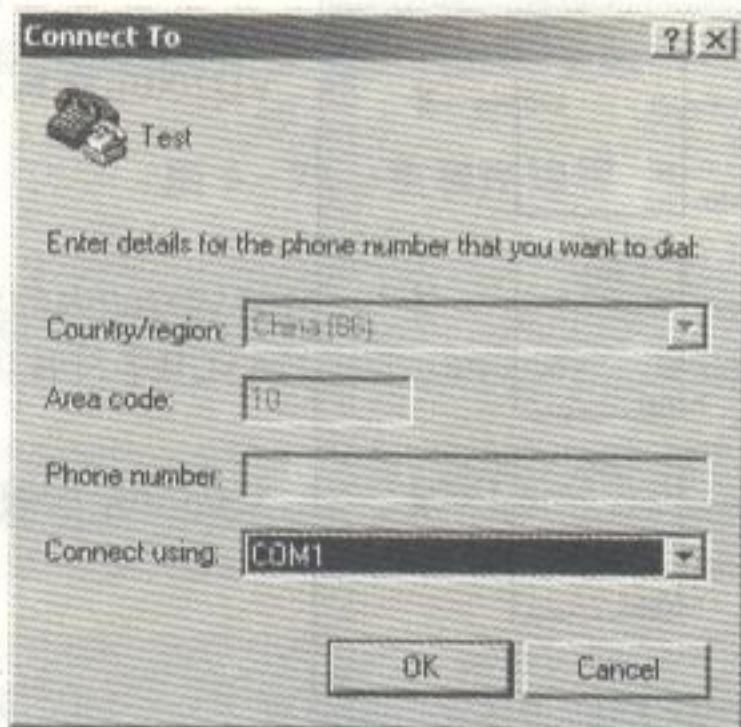


图 3-77 选择串行通讯端口



☞ 选择通讯口参数,如图 3-78 所示。

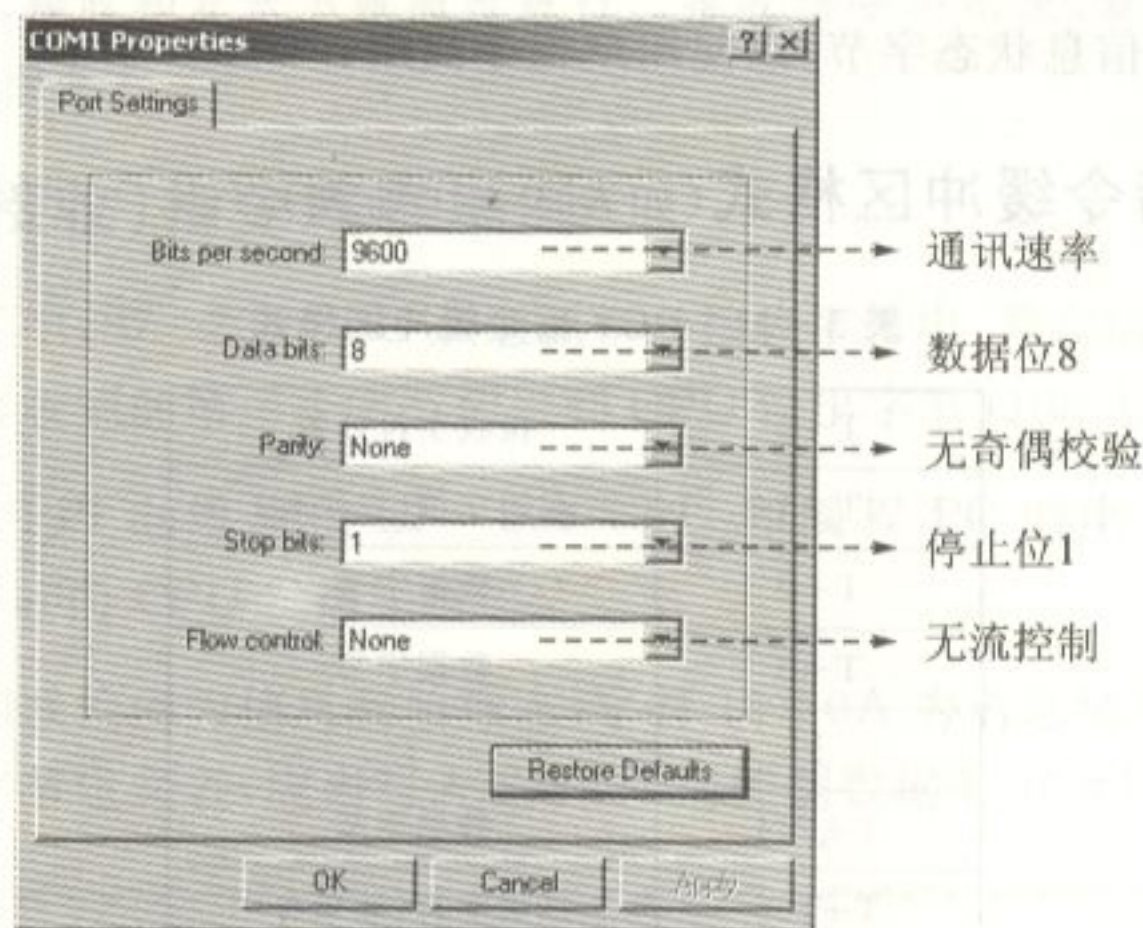


图 3-78 选择通讯口参数

☞ 在 HyperTerminal(超级终端)窗口中应当显示由 S7-200 CPU 发送来的字符串(如图 3-79 所示)。

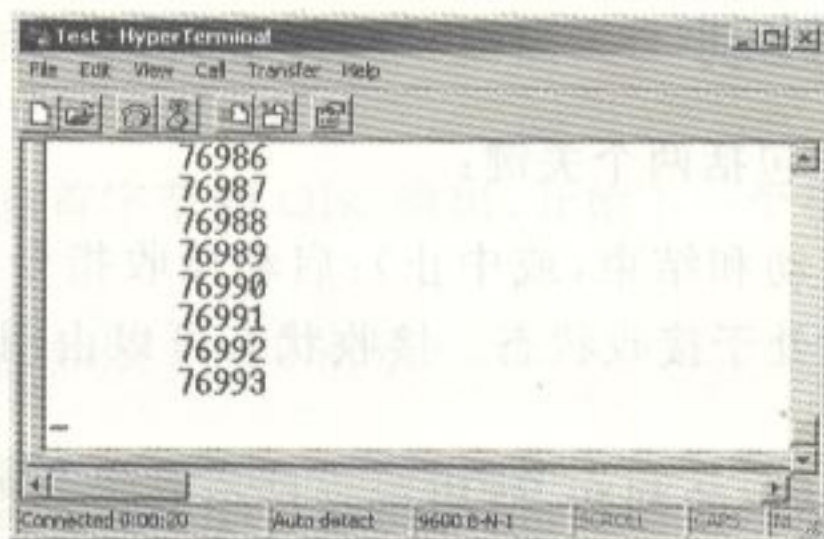


图 3-79 S7-200 CPU 发送的字符串

3.8.2 接收指令

RCV(接收)指令从 S7-200 CPU 的通讯口接收一个或多个数据字节。接收的数据字节保存在接收数据缓冲区内。



接收指令完成后,会产生一个中断事件(对 Port0 为中断事件 23,对 Port1 为中断事件 24)。特殊寄存器 SMB86(对 Port0)和 SMB186(对 Port1)也提供了接收信息状态字节,以使用户程序使用。

RCV 指令缓冲区格式(如表 3-21 所列)

表 3-21 RCV 指令缓冲区格式

T+0	接收字符计数
T+1	起始字符(如果有)
T+2	数据字节
T+3	数据字节
⋮	⋮
T+244	数据字节
T+255	结束字符(如果有)

RCV 指令控制特殊寄存器

RCV 指令的所有控制均通过程序设置接收指令控制特殊寄存器完成。

RCV 指令的控制

RCV 指令的工作包括两个关键:

- 接收指令的启动和结束(或中止):启动接收指令后,S7-200 CPU 的通讯控制器就处于接收状态。接收状态可以由用户程序中止,接收指令结束。
- 消息串起始/结束的判断:使用接收指令时需要设置消息起始和结束的判断条件。接收指令启动后,通讯控制器用这些条件来判断消息的开始和结束。判断消息结束时,接收状态终止;否则,通讯口会一直处在接收状态。

接收指令的启动、结束(或终止),以及消息起始和结束条件,都通过接收指令控制字节(SMB87 对 Port0,SMB187 对 Port1)和其他一些控制特殊存储器设置。



✌ 由于 S7-200 CPU 的通讯建立在 RS-485 半双工通讯的硬件基础上,接收和发送不能同时进行。接收指令不结束,就不能执行发送指令。

RCV(接收)指令编程举例

在本例子中,S7-200 CPU 从通讯口 0 接收字符串,并在信息接收中断服务程序中把接收到的第一个字节传送到 CPU 输出字节 QB0 上显示。

使用 PC/PPI 电缆连接 S7-200 CPU 和编程 PC 的串口。我们使用 HyperTerminal 向 CPU 发送字符串。

选择空闲线检测为信息起始标志,字符 $16\#0A$ 为消息结束字符,根据接收字节控制字节定义表,应当写入 SMB87 的控制数据为 $16\#B0$ 。

✌ 这里选择 $16\#0A$ 为结束字符,是因为 $16\#0D0A$ 在 HyperTerminal 软件中是“回车”(换行)符。

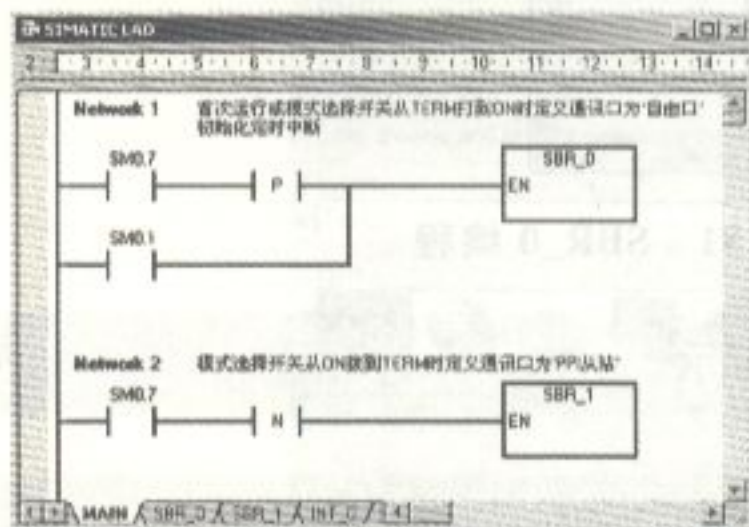
主程序:根据 CPU 模式开关的状态,定义通讯口

SBR_0:定义自由口接收指令参数,连接接收结束中断,开始接收

SBR_1:重定义 PPI 通讯口

INT_0:传送消息首字节到 QB0 输出,开始下一个接收过程

✌ 主程序编程(如图 3-80 所示)



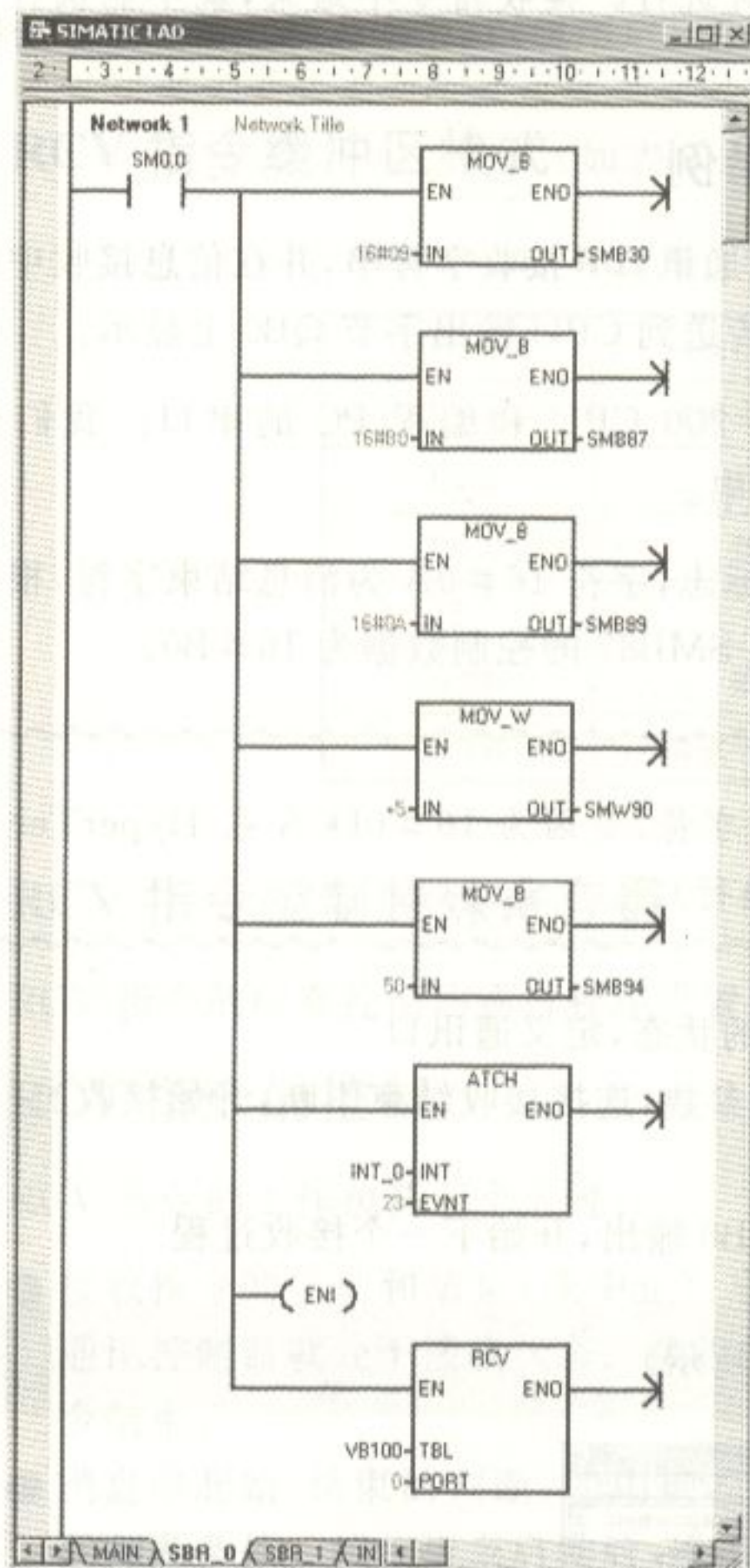
调用自由口初始化子程序

恢复普通PPI通讯口设置

图 3-80 主程序编程



☞ SBR_0 编程如图 3-81 所示。



定义通讯口0为自由口模式
16#09定义自由口工作在9.6K波特、无校验、8数据位模式下

写接收指令控制字，允许RCV，检测信息结束字符、空闲线起始条件

设置信息结束字符为16#0A(换行符)

设置空闲线检测时间间隔为5 ms

设置最大字符数为50

连接端口0接收结束中断到INT_0

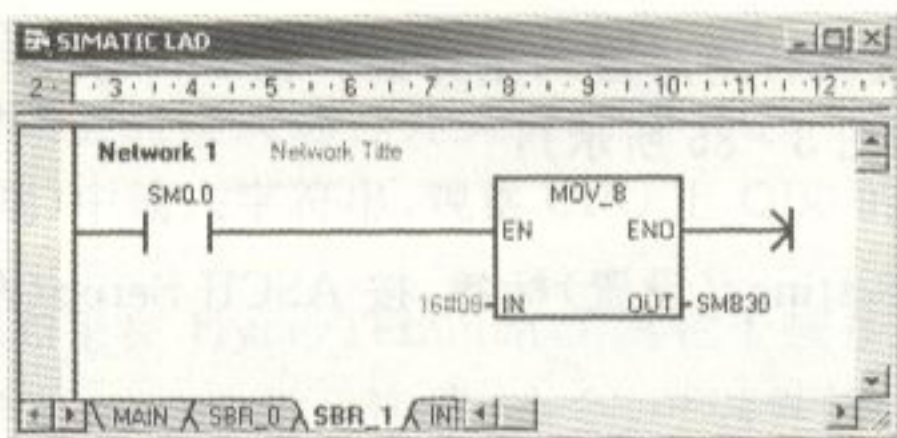
允许中断

启动接收指令，接收缓冲区以VB100开始

图 3-81 SBR_0 编程

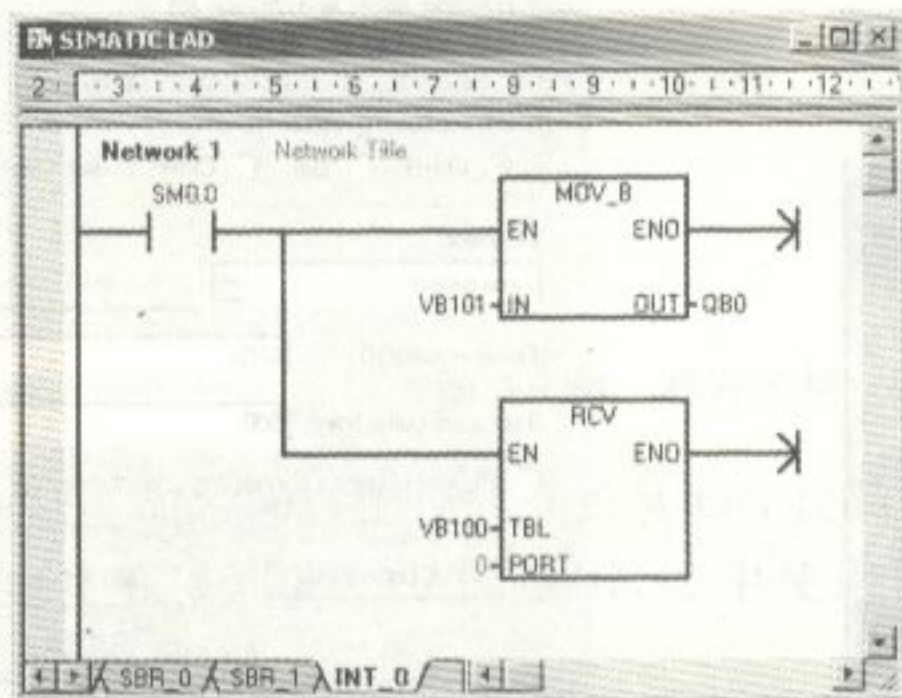
☞ SBR_1 编程如图 3-82 所示。

☞ INT_0 编程如图 3-83 所示。



设置端口0为普通PPI口

图 3-82 SBR_1 编程



在QB0输出接收到的
的第一个字节

开始下一次接收

图 3-83 INT_0 编程

使用 HyperTerminal 调试

☞ 打开 Windows 系统的 HyperTerminal 程序,选择图标,指定一个连接名称(如图 3-84 所示)。

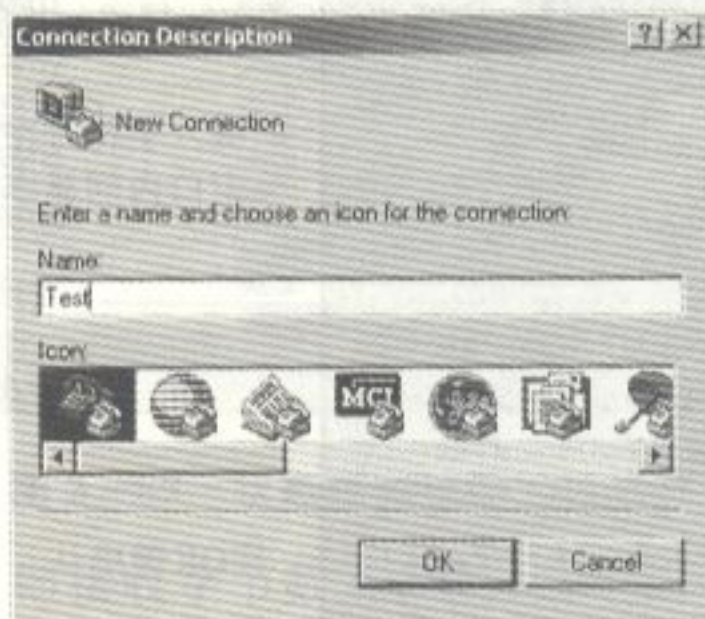


图 3-84 指定连接名称



☞ 选择监控通讯口,设置波特率后进入主界面。在 File(文件)菜单中选择 Properties(属性)命令(如图 3-85 所示):

☞ 在属性窗口中,选择 Settings(设置)标签,按 ASCII Setup(ASCII 设置)按钮(如图 3-86 所示)。



图 3-85 选择“属性”命令

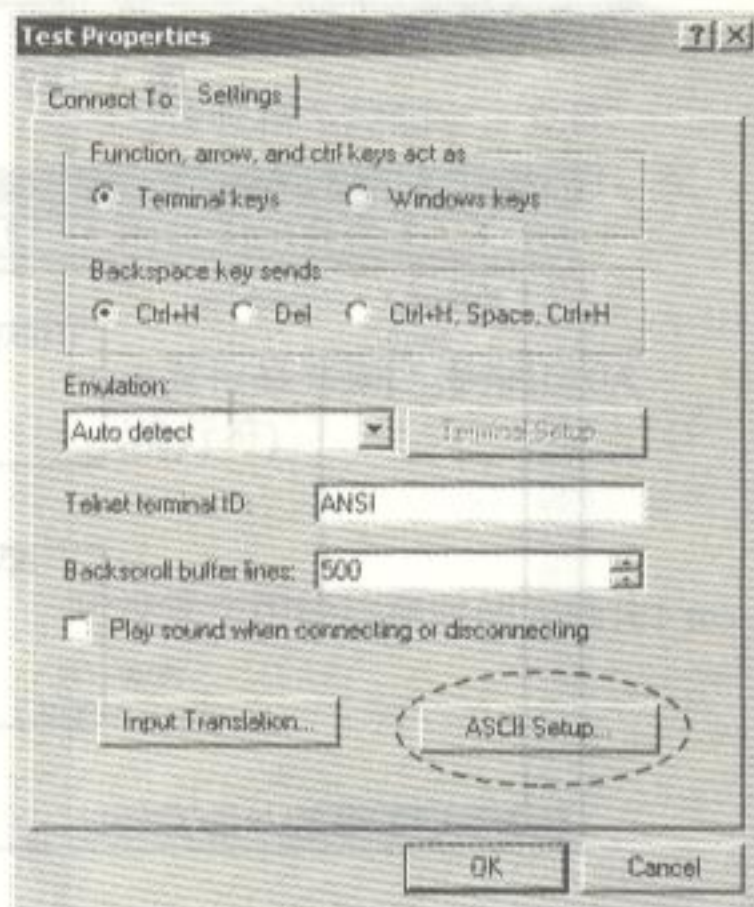
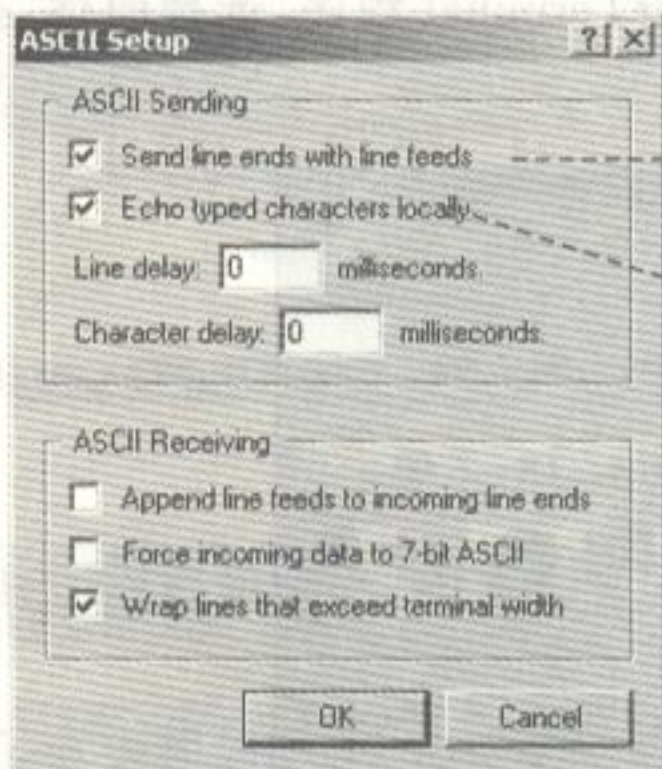


图 3-86 监控参数设定

☞ 选中下列字符发送属性(如图 3-87 所示):



→ 在行尾添加换行符

→ 键入字符时在HyperTerminal窗口内回显字符

按OK键确认

图 3-87 字符发送属性的设定



下载 S7-200 项目后,断开 Step7-Micro/WIN32 与 CPU 的连接。将 S7-200 CPU 上的模式选择开关拨动到 RUN(运行)位置。在 HyperTerminal (超级终端)中输入字符串,观察 CPU 上 QB0 的状态。

如果在 HyperTerminal 工具栏上按挂断按钮,或在 Call(呼叫)菜单中选择 Disconnect(断开连接)命令,可以释放 HyperTerminal 对 PC 机串行口的占用(如图 3-88 所示)。

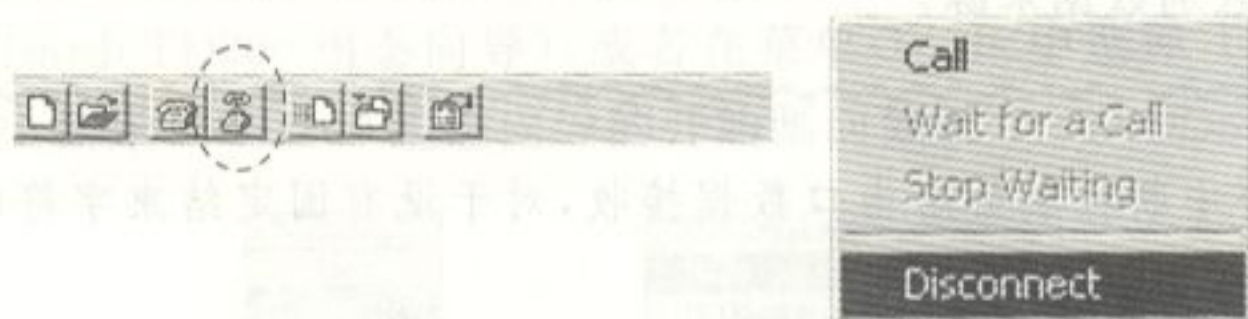


图 3-88 断开连接

将 S7-200 CPU 上模式开关从 ON 拨到 TERM,重新定义自由口为 PPI 从站模式。在 Step7-Micro/WIN32 中使用状态图,在线观察缓冲区内容(如图 3-89 所示)。

Status Chart			
	Address	Format	Current Value
1	VB100	Unsigned	0
2	VB101	ASCII	0
3	VB102	ASCII	0
4	VB103	ASCII	0
5	VB104	ASCII	0
6	VB105	ASCII	0
7	VB106	ASCII	0
8	VB107	ASCII	0
9	VB108	ASCII	0
10	VB109	ASCII	0
11	VB110	ASCII	0
12	VB111	ASCII	0
13	VB112	Hexadecimal	1E800
14	VB113	Hexadecimal	1E80A
15	VB114	ASCII	007
16	VB115	ASCII	007
17	VB116	ASCII	007
18	VB117	ASCII	007
19	VB118	ASCII	007
20		Signed	
21	QB0	ASCII	0
22		Signed	

图 3-89 状态图监控缓冲区内容

注意:这时 VB100(即接收字节数)为零。这是因为在未接收到任何字符前重新设置了 S7-200 CPU 的通讯模式,恢复为普通 PPI 从站模式。



使用字符中断控制接收数据

通讯口接收每个字符时都会产生中断,每个接收到的字符都会暂存在特殊存储器 SMB2 中,校验结果保存在 SMB3 中。这也可以用于在自由口模式下接收数据。

通讯端口 Port0 和 Port1 共用 SMB2 和 SMB3,但可由不同的中断事件号(8 和 25)来区别数据来源。



字符中断控制的自由口数据接收,对于没有固定结束字符的通讯协议,十分有效。

3.9 TD200 组态

TD200(Text Display 200)是专用于 S7-200 系列的文本显示和操作员界面。TD200 支持中文操作和文本显示。

TD200 包装中提供了专用电缆(TD/CPU 电缆)用以与 S7-200 CPU 连接,电缆能从 CPU 通讯口上取得 TD200 所需的 24 V DC 电源。TD200 同时提供了 24 V DC 电源输入接口,仅供通过 Profibus 电缆连接到 CPU 或 PPI 网络上时使用。

TD200 作为主站在 PPI 网络上工作。网络上的 TD200(包括其他设备)都有惟一的地址。1 个 S7-200 CPU 最多可以连接 4 个 TD200;1 个 TD200 只能与 1 个 S7-200 CPU 建立连接。

连接到同一个 S7-200 CPU 的多个 TD200 可以访问同一个参数块,也可设置不同的数据块偏移地址按不同的参数块工作。不同的参数块可以分多次调用 TD200 向导定义。

Step7-Micro/WIN32 提供了集成的 TD200 组态工具。TD200 的组态



信息全部保存在 S7-200 CPU 中,可以方便地更换 TD200 而不必重新组态。

TD200 通过设定 S7-200 CPU 的地址决定访问哪个 CPU,使用数据块偏移地址访问自身的组态信息(即 TD 参数块)。这些参数,包括 TD200 自身的地址和通讯速率等在 TD200 的系统菜单中设置。

打开 TD200 组态向导

在 Step7 - Micro/WIN32 中选择 Tools(工具)浏览条,在其中用鼠标点击 TD 200 Wizard(TD200 组态向导),或者在菜单 Tools 中选择 TD200 Wizard(如图 3-90 所示),TD200 组态向导的开始画面如图 3-91 所示。

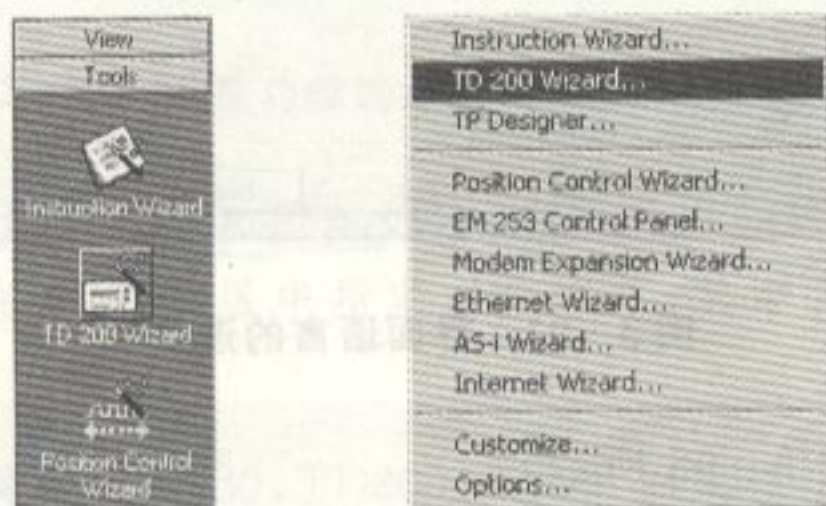


图 3-90 打开 TD200 组态向导

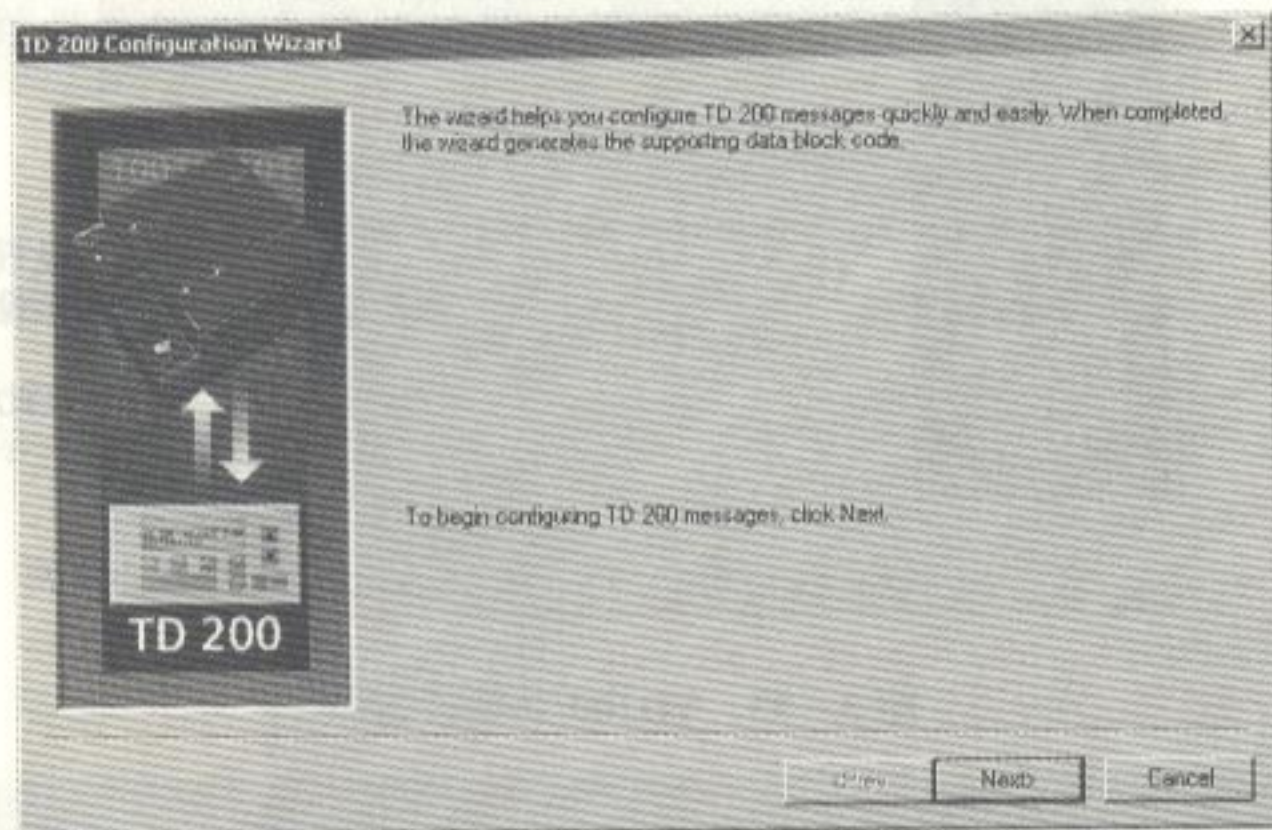


图 3-91 TD200 组态向导开始画面



☞ 按 Next>(下一步)键,选择界面语言(如图 3-92 所示):

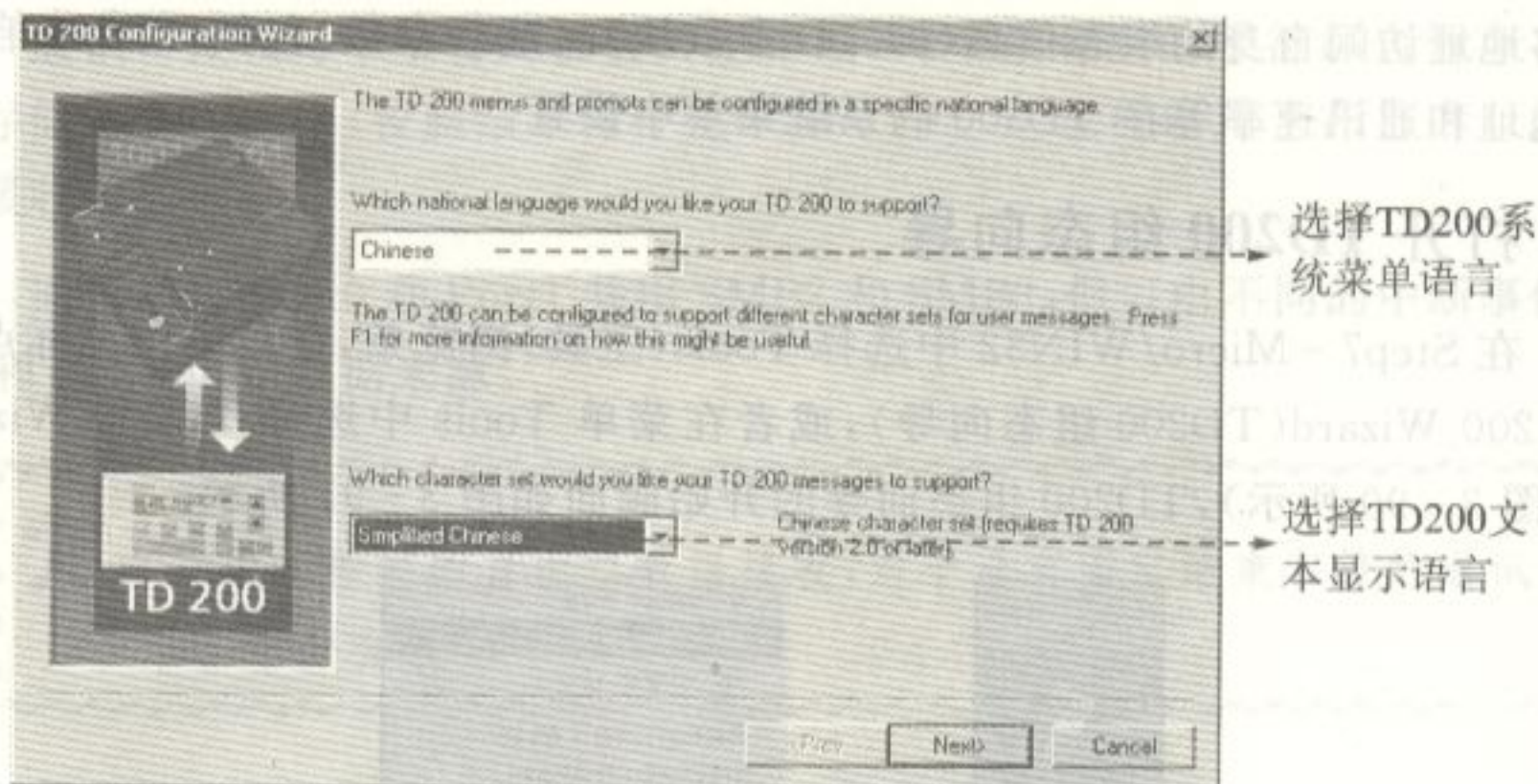


图 3-92 界面语言的选择

☞ 下一步(如图 3-93 所示):

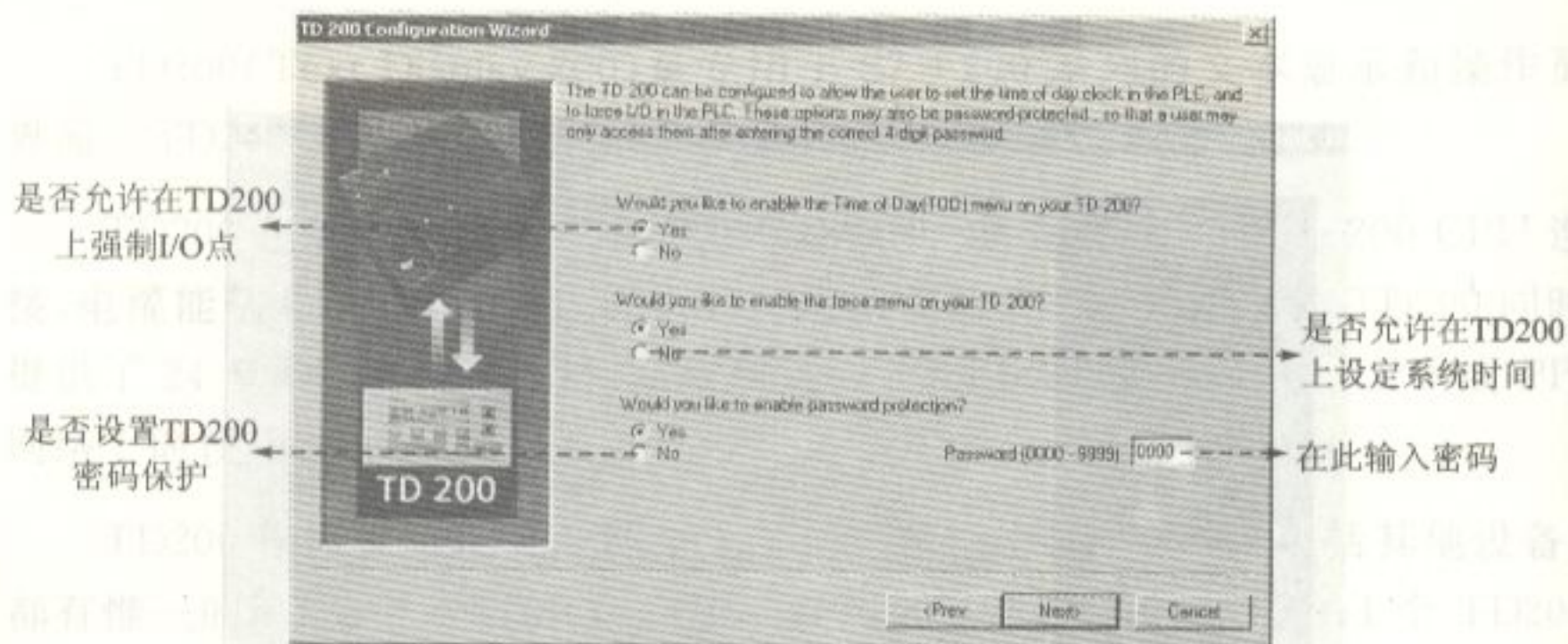


图 3-93 TD200 功能设定

☞ 我们选择不设置密码,按 Next(下一步)键(如图 3-94 所示):

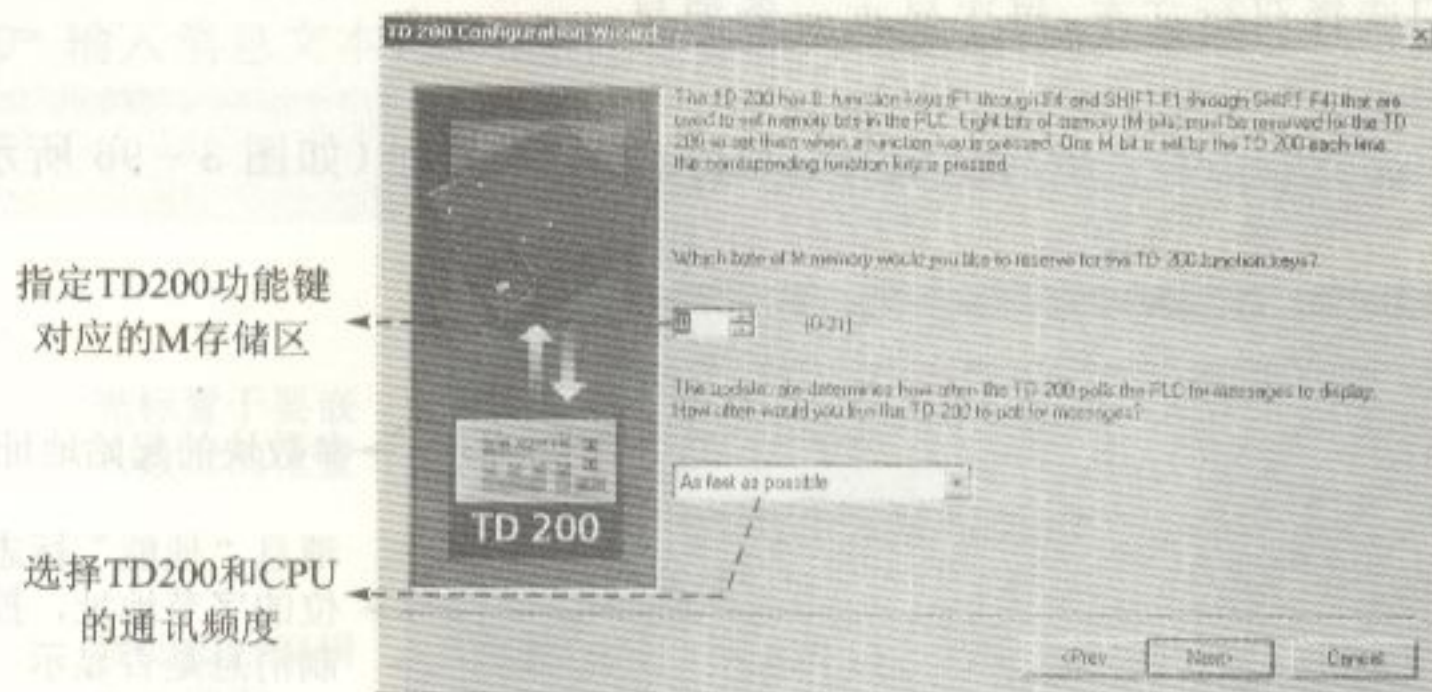


图 3-94 设置功能键对应存储区和通讯频度

✌ TD200 面板上有 8 个功能键 (F1~F4, Shift+F1~Shift+F4), 可以为它们在 M 存储区中指定对应字节, 其中各位对应于功能键的状态。

✌ 如组态功能键对应 MB0, TD200 工作时如果按下 F1 键, 则 S7-200 CPU 内存中的 M0.0 会置位为“1”。

☞ 继续选择消息格式和数目(如图 3-95 所示)。

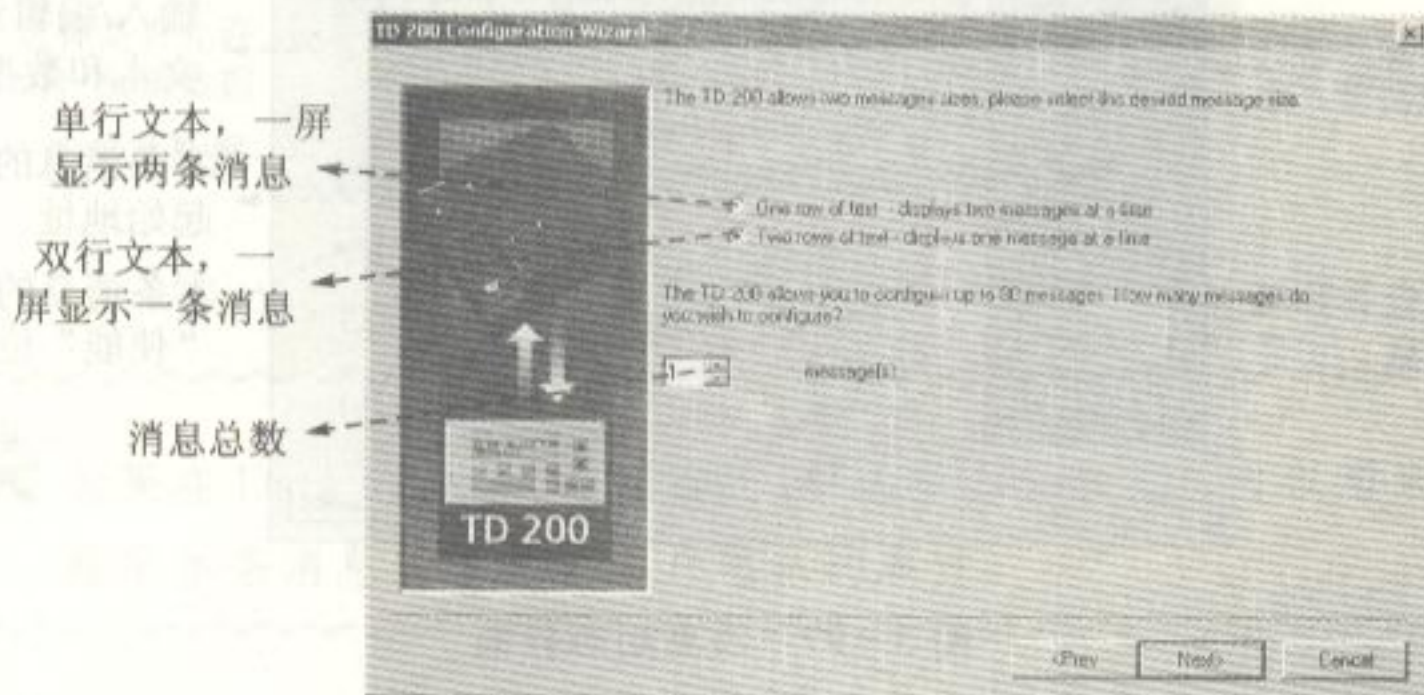


图 3-95 选取消息格式和数目



我们选择双行文本,每次显示一条消息。

☞ 按 Next(下一步)键,定义 TD200 参数块地址(如图 3-96 所示)。

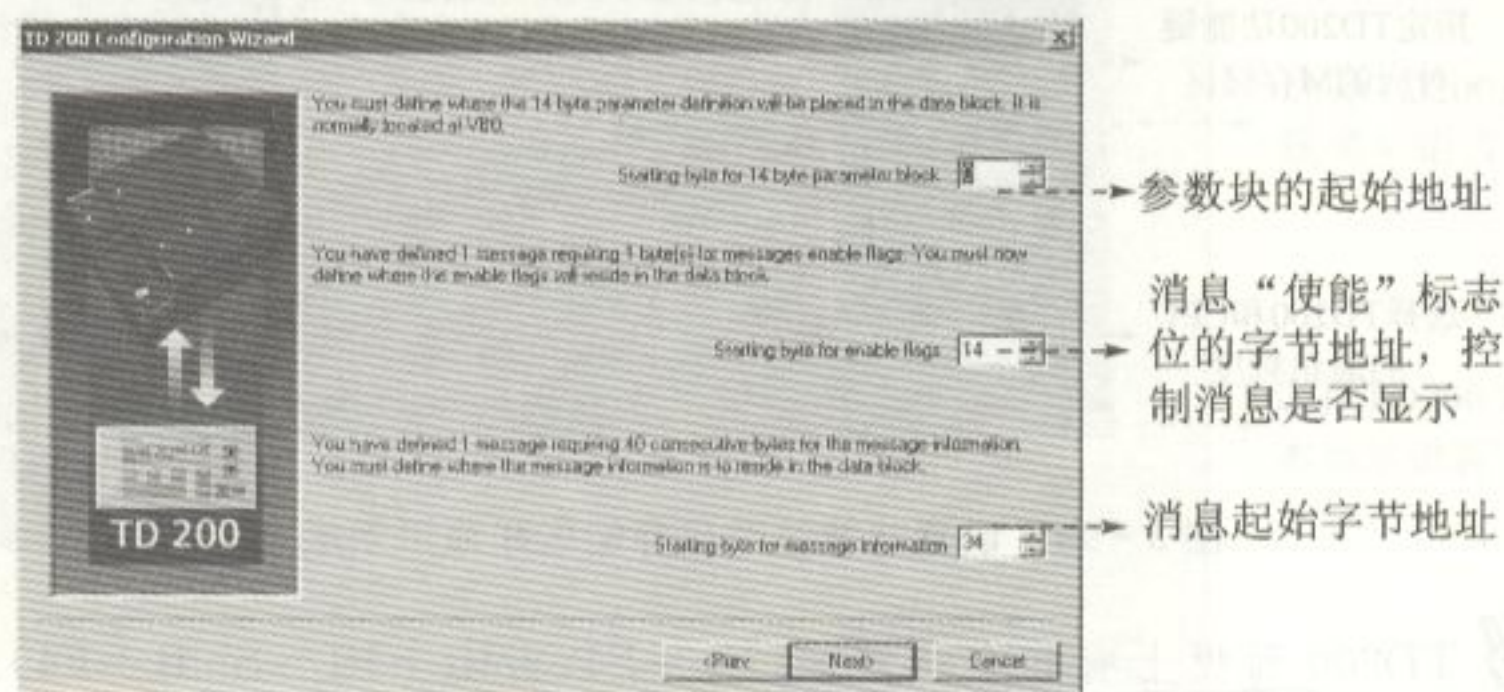


图 3-96 TD200 参数块地址的定义

☞ 上述地址,全部在 V 存储区内。14 实际上指 VB14。

☞ 按 Next 键进入消息组态界面(如图 3-97 所示)。

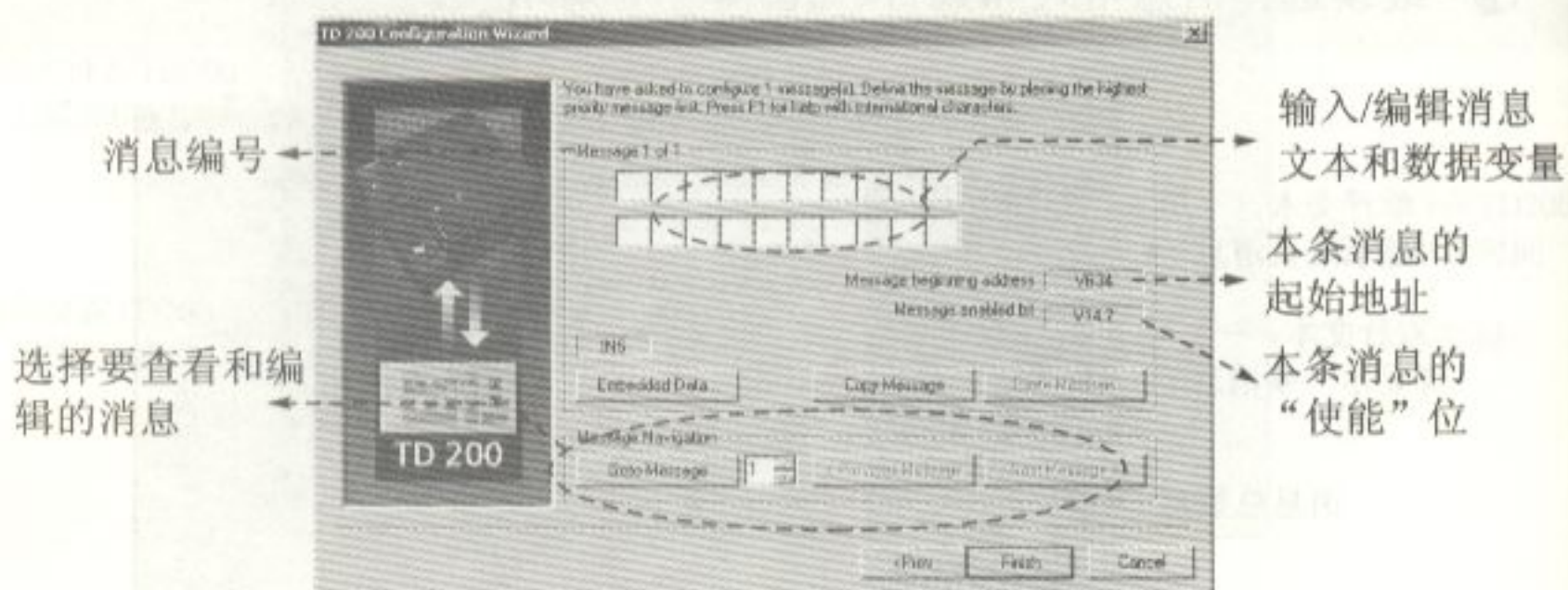


图 3-97 消息组态界面

☞ 具体的消息“使能”位由 TD200 组态向导自动分配,“使能”位置“1”时,本条消息才能被 TD200 显示。



☞ 输入消息文本,把光标置于要嵌入数据的位置(如图 3-98 所示)。

光标置于要嵌入数据的位置

按此按钮编辑嵌入数据

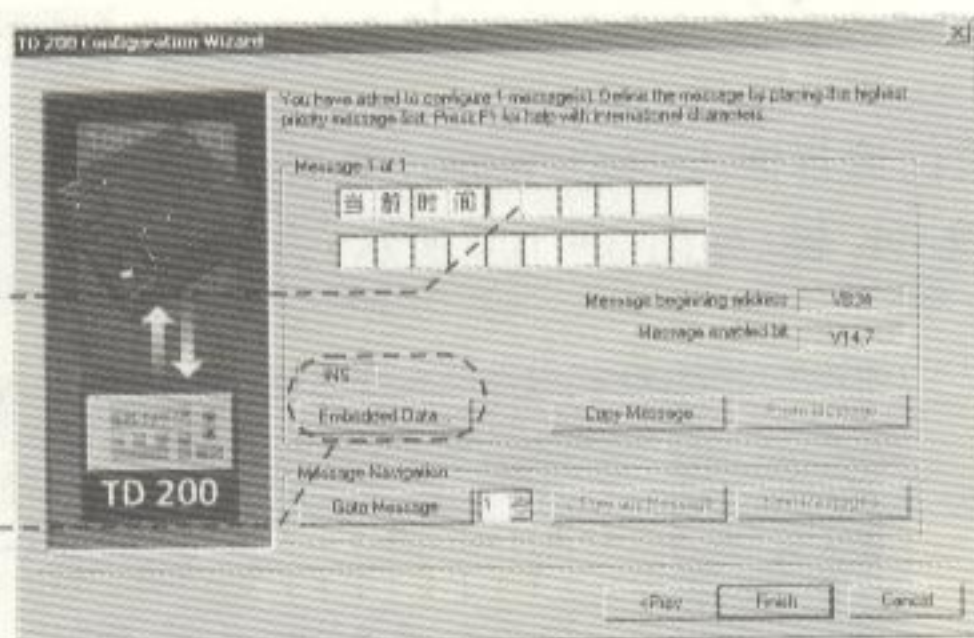


图 3-98 输入消息文本

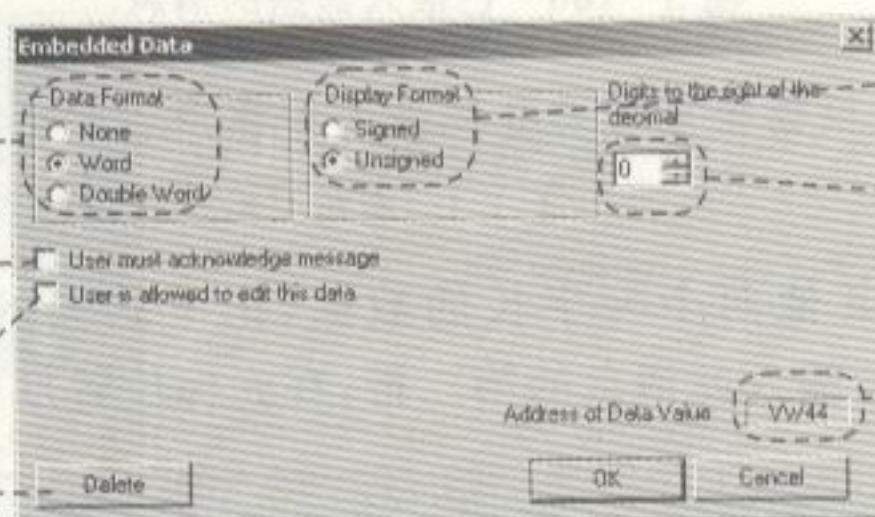
☞ 按 Embedded Data(嵌入数据)按钮进入嵌入数据编辑对话框(如图 3-99 所示)。

选择数据格式

选择是否需要用户确认消息

选择是否允许用户编辑数据

删除此数据



选择数据
显示格式

指定小数点
右边显示的
小数位数

自动分配的
嵌入数据地址

图 3-99 嵌入数据编辑对话框



如果在 Data Format(数据格式)中选择 None(无),可以用来单独指定本条消息是否具有用户确认的属性。

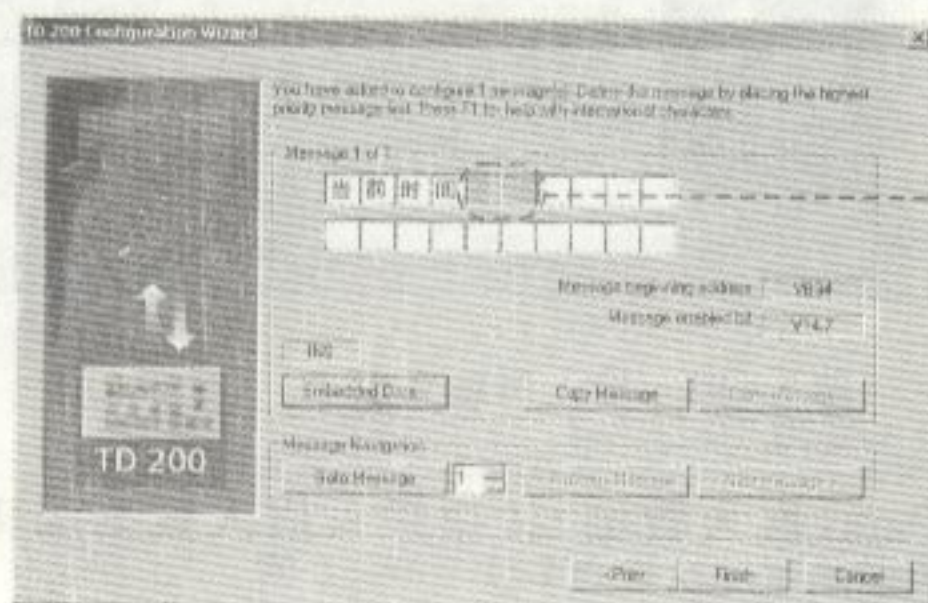


需要用户确认的消息,指消息在 TD200 上显示时不断闪烁,必须由用户按 Enter 键确认,才能进行下一步操作。



允许用户编辑的数据,用于从 TD200 上输入用户设定值等数据。

我们选择嵌入一个 Word(字)长的数据,以 Unsigned(无符号数)格式显示。按 OK 键(如图 3-100 所示)。



阴影部分表示
嵌入数据

图 3-100 已嵌入数据的消息



组态向导会根据需要自动分配嵌入数据所占据的消息长度,用户不必干预。

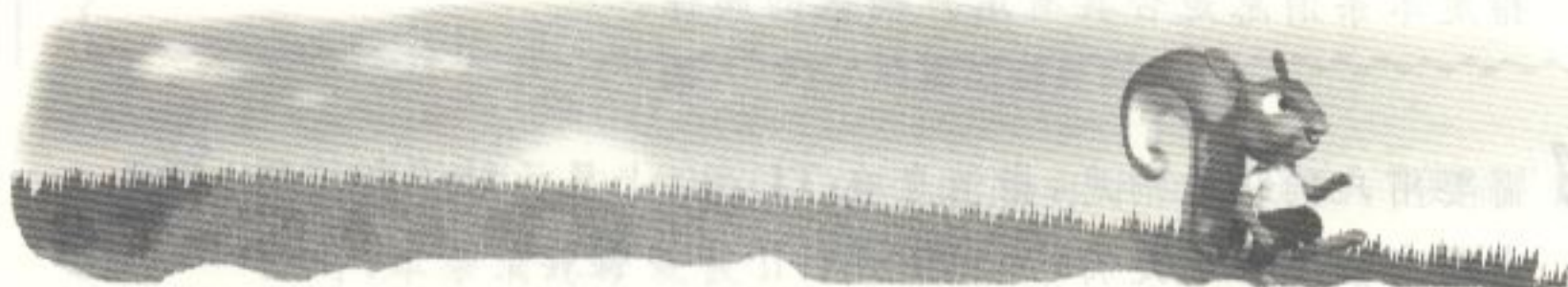
我们继续嵌入其他两个一字长的数据,记下它们的嵌入数据地址为: VW48、VW52。继续输入此消息的第二行,在光标位置准备嵌入另一个数据。



按 Embedded Data(嵌入数据)按钮(如图 3-101 所示)。



数据确认位:用户完成对数据的编辑后,此位置位。



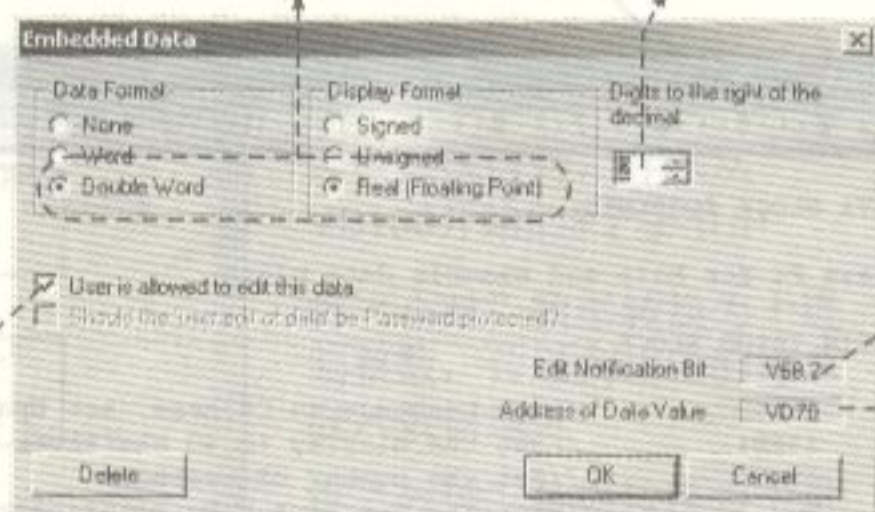


选中双字数据格式才能出现
现实数(浮点数)选项

显示两位小数



允许用户编辑(改动)
这个数据



数据的编
辑确认位

数据地址

图 3-101 嵌入数据窗口

✎ 嵌入数据完成,按 Finish(结束)键完成消息编辑,结束 TD200 向导(如图 3-102 所示)。



图 3-102 完成嵌入数据

TD200 参数块

TD200 所有的组态信息都保存在 S7-200 项目的数据块中,称为参数块。每个参数块以“TD”字符串开始。参数块中为每个参数提供了详细的注释。



☞ 上述 TD200 组态的参数块如图 3-103 所示。

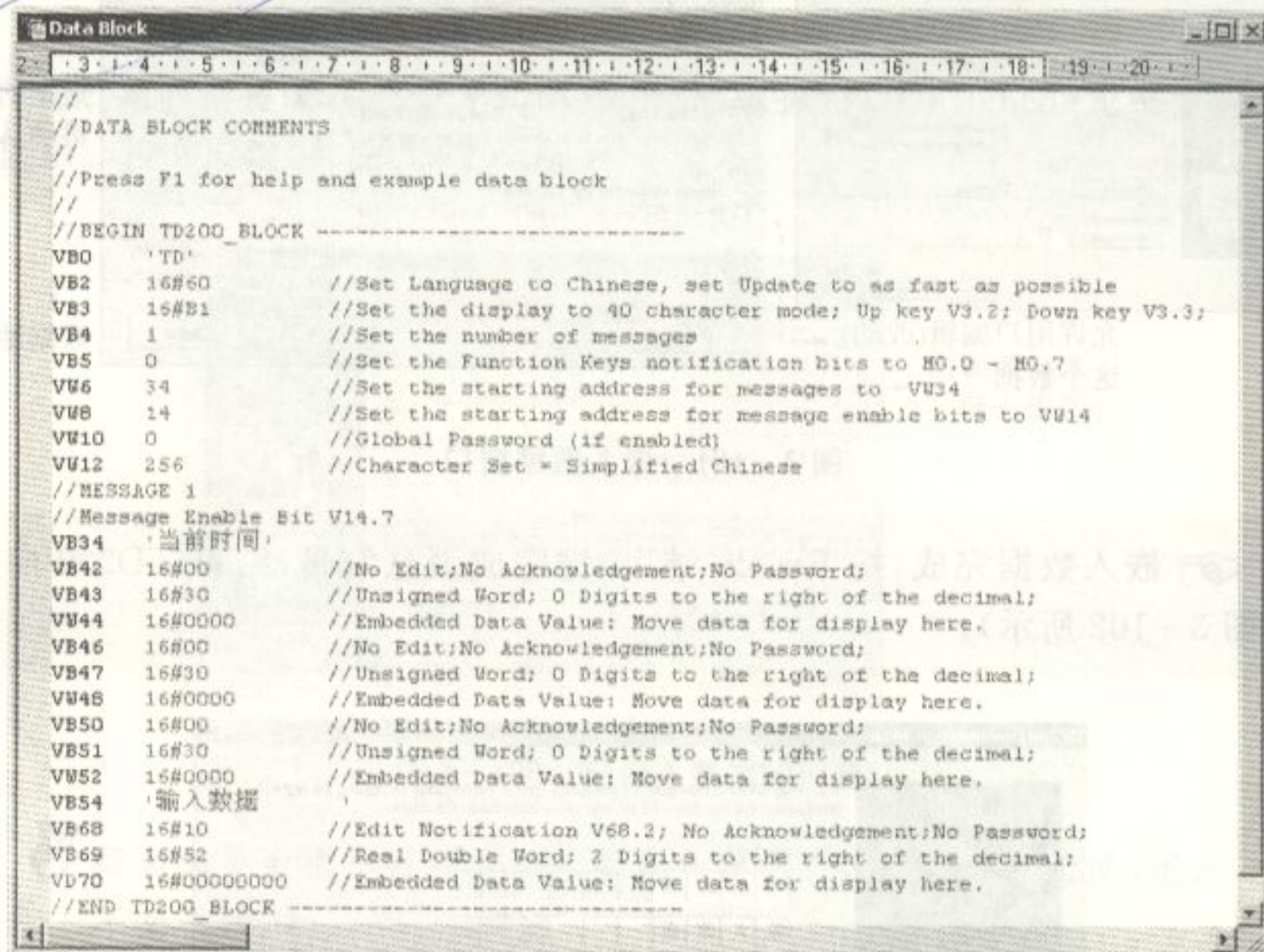


图 3-103 TD200 组态参数块

相关编程

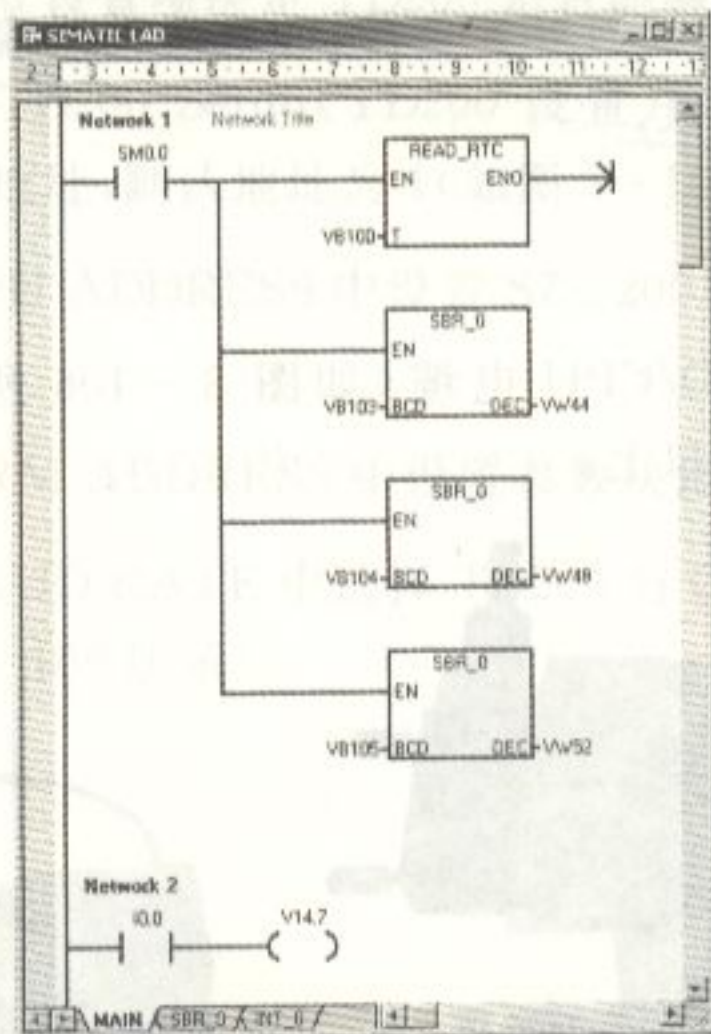
在例子程序中,我们将 S7-200 CPU 的系统时钟的时、分、秒在 TD200 上显示。

主程序:读 S7-200 CPU 系统时钟,将时、分、秒转换为十进制整数,传送到相应的嵌入数据地址中。消息使能位置位后,TD200 会自动读取这些数据并显示出来。

SBR_0:完成数据从 BCD 格式到十进制格式的转换。



主程序编程(如图 3-104 所示):



读取系统时钟, 存入从
VB100开始的8个字节中

将“小时”数据转换为十进
制整数, 传送到相应的消息
嵌入数据地址

将“分钟”数据转换为十进
制整数, 传送到相应的消息
嵌入数据地址

将“秒”数据转换为十进制
整数, 传送到相应的消息嵌
入数据地址

I0.0接通时, 消息使能位
V14.7接通, 消息得以显示

图 3-104 主程序编程

SBR_0 编程(如图 3-105 所示):

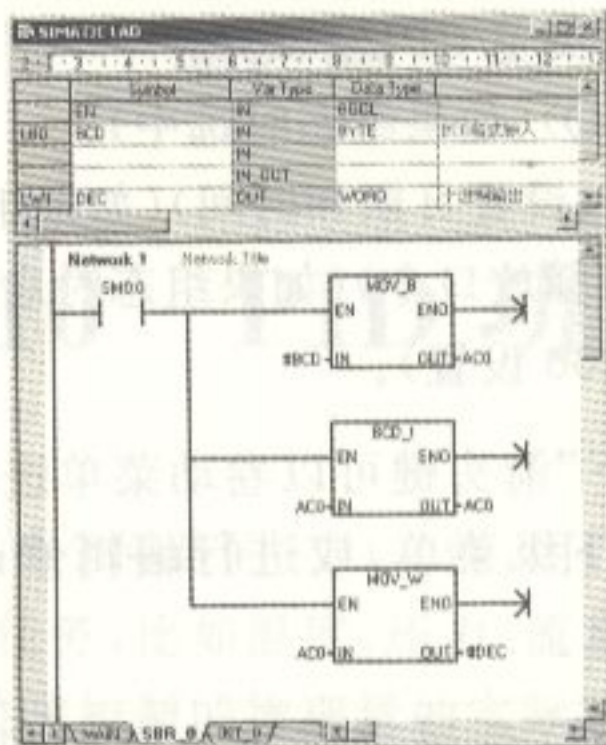


图 3-105 SBR_0 编程



✌ TD200 支持多达 80 条消息。消息的显示与否由消息使能位的状态决定。多条消息的使能位同时置位时,各消息按照编号由小到大的顺序决定显示优先级。

TD200 菜单操作简介

使用随机提供的连接电缆(TD/CPU 电缆(如图 3-106 所示)),将 TD200 连接到 S7-200 CPU 的通讯口上。



图 3-106 连接电缆

接通 S7-200 CPU 电源。按 ESC 键,进入 TD200 菜单方式。

可用的菜单项目有:

- View Messages(浏览消息);
- View CPU Status(浏览 CPU 状态);
- Force I/O(强制 I/O)(如果组态时选中);
- Set Time and Date(设置日期和时间)(如果组态时选中);
- Release Password(释放口令)(如果组态时启用);
- TD200 Setup(TD200 设置)。

按“向上”箭头和“向下”箭头键可以卷动菜单进行选择,当显示出想要的项目时,按 Enter 键进入下级菜单,或进行编辑修改,完成后再按 Enter 键确认。

显示菜单时按 ESC 键,会退出菜单方式。如果 1 分钟内没有操作,TD200 也会自动从菜单方式退出到消息显示方式。



TD200 设置

TD200 设置项决定 TD200 是否能够正确访问 S7-200 CPU 的数据。

进入 TD200 Setup (TD200 设置) 菜单, 选择 TD 200 ADDRESS 设置 TD200 的地址, 默认地址为 1 (如图 3-107 所示)。

在 CPU ADDRESS 中设置 S7-200 CPU 地址, 默认值为 2 (如图 3-108 所示)。

在 ARM ADDRESS 中设置参数块地址, 默认地址为 0。

在 BAUD RATE 中选择 TD200 与 CPU 的通讯速率, 默认值为 9.6 K 波特 (如图 3-109 所示)。



图 3-107



图 3-108



图 3-109

3.10 PID 功能

S7-200 CPU 提供了 8 个回路的 PID 功能, 用以实现需要按照 PID 控制规律进行自动调节的控制任务, 比如温度、压力、流量控制等等。PID 功能一般需要模拟量输入, 以反映被控制的物理量的实际数值, 称为反馈; 而用户设定的调节目标值, 即为给定。PID 运算的任务就是根据反馈与给定的相对差值, 按照 PID 运算规律计算出结果, 输出到加热棒、变频器驱动的水泵等执行



机构进行调节,以达到自动维持被控制的量跟随给定变化。

★ PID:比例/积分/微分控制。一种自动控制方法,目的是使被控制的物理量追随给定值而且稳定,并自动消除各种因素对控制效果的扰动。

S7-200 中 PID 功能的核心是 PID 指令。PID 指令需要为其指定一个以 V 变量存储区地址开始的 PID 回路表,以及 PID 回路号。PID 回路表提供了给定、反馈,以及 PID 参数等数据入口,PID 运算的结果也在回路表输出。

PID 回路表

PID 指令接收如下格式的回路如表 3-22 所列。

表 3-22 PID 指令接收

偏移地址	域	格 式	类 型	描 述
T+0	过程变量(反馈)(PVn)	双字-实数	输入	过程变量,必须在 0.0~1.0 之间
T+4	设定值(给定)(SPn)	双字-实数	输入	给定值,必须在 0.0~1.0 之间
T+8	输出值(Mn)	双字-实数	输入/输出	输出值,必须在 0.0~1.0 之间
T+12	增益(P 参数)(KC)	双字-实数	输入	增益是比例常数,可正可负
T+16	采样时间(TS)	双字-实数	输入	单位为秒,必须是正数
T+20	积分时间(I 参数)(TI)	双字-实数	输入	单位为分钟,必须是正数
T+24	微分时间(D 参数)(TD)	双字-实数	输入	单位为分钟,必须是正数
T+28	积分项前项(MX)	双字-实数	输入/输出	积分项前项,必须在 0.0~1.0 之间
T+32	过程变量前值(PVn-1)	双字-实数	输入/输出	最近一次 PID 运算的过程变量值

在一般的控制系统中,经常只用到 PID 调节,这时需要把微分参数设为零。

回路数据的转换和标准化

由 PID 回路表可见,输入/输出数据都是实数。所以要使用 PID 功能,还需要对实际的过程和设定数据进行转换和标准化。输出数据也要经过转换,才能送到模拟量输出通道或者开关量调节通道。



✌️ 换算关系略图如图 3-110 所示。

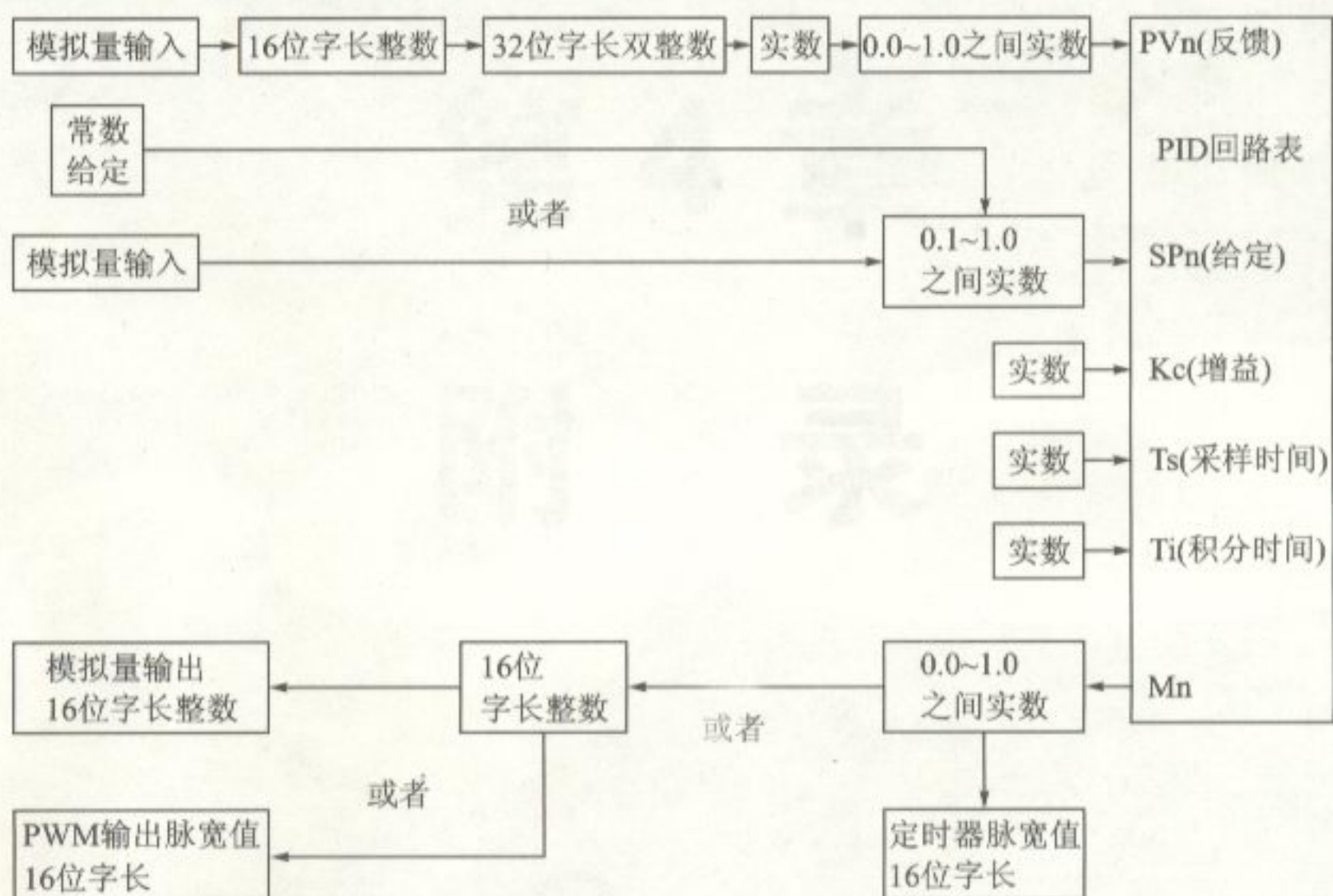


图 3-110 换算关系示意图



Step 7 - Micro/WIN32 提供了 PID Wizard(PID 向导)帮助用户快速建立 PID 控制功能的相关程序

第 4 章

附 录





4.1 CPU 规格

CPU 规格如表 4-1 所列。

表 4-1 CPU 规格

	CPU 221	CPU 222	CPU 224	CPU 226	CPU 226XM
存储器					
用户程序空间/字节	4 096		8 192	8 192	16 384
用户数据(EEPROM) /字节	2 048(永久存储)		5 120 (永久存储)	5 120 (永久存储)	10 240 (永久存储)
装备(超级电容)	50 小时/典型值(40℃时最少 8 小时)		190 小时/典型值(40℃时最少 120 小时)		
(可选电池)	200 天/典型值		200 天/典型值		
I/O					
本机数字输入/输出	6 输入/4 输出	8 输入/6 输出	14 输入/10 输出	24 输入/16 输出	
数字 I/O 映象区	256(128 入/128 出)				
模拟 I/O 映象区	无	32(16 入/16 出)	64(32 入/32 出)		
允许最大的扩展模块	无	2 模块	7 模块		
允许的最大的智能模块	无	2 模块	7 模块		
脉冲捕捉输入	6	8	14		
高速计数	4 个计数器		6 个计数器		
单 相	4 个 30 kHz		6 个 30 kHz		
两 相	2 个 20 kHz		4 个 20 kHz		
脉冲输出	2 个 20 kHz(DC 输出 only)				
常 规					
定时器	256 定时器;4 个定时器(1 ms);16 定时器(10 ms);236 定时器(100 ms)				
计数器	256(由超级电容或电池备份)				
内部存储位	256(由超级电容或电池备份)				
掉电保存	112(存储在 EEPROM)				
时间中断	2 个 1 ms 分辨率				



续表 4-1

	CPU 221	CPU 222	CPU 224	CPU 226	CPU 226XM
边沿中断	4 个上升沿和/或 4 个下降沿				
模拟量调查	1 个 8 位分辨率			2 个 8 位分辨率	
布尔量运算执行速度	0.37 μ s 每条指令				
时 钟	可选卡件			内 置	
卡件选项	存储卡, 电池卡和时钟卡			存储卡和电池卡	
集成的通讯功能					
接 口	一个 RS-485 口			2 RS-485 ports	
PPI, DP/T 波特率	9.6 K、19.2 K、187.5 K 波特				
自由口波特率	1.2 K~115.2 K 波特				
每段最大电缆长度	使用隔离的中继器: 187.5 K 波特可达 1 000 m, 38.4 K 波特可达 1 200 m 未使用隔离中继器: 50 m				
最大站数	每段 32 个站, 每个网络 126 个				
最大主站数	32				
点到点(PPI 主站模式)	是(NETR/NETW)				
MPI 连接	共 4 个, 2 个保留(1 个给 PG, 1 个给 OP)				

CPU 电源规范如表 4-2 所列。

表 4-2 CPU 电源规范

DC			AC	
输入电源				
输入电压	20.4~28.8 V DC		85~264 V AC(47~63 Hz)	
输入电流	仅 CPU, 24 V DC	最大负载 24 V DC	仅 CPU	最大负载
CPU 221	80 mA	450 mA	30/15 mA 120/240 V AC	120/60 mA 120/240 V AC
CPU 222	85 mA	500 mA	40/20 mA 120/240 V AC	140/70 mA 120/240 V AC
CPU 224	110 mA	700 mA	60/30 mA 120/240 V AC	200/100 mA 120/240 V AC
CPU 226/CPU 226XM	150 mA	1 050 mA	80/40 mA 120/240 V AC	320/160 mA 120/240 V AC
冲击电流	10 A 28.8 V DC		20 A,264 V AC	
隔离(现场与逻辑)	不隔离		1 500 V AC	
保持时间(掉电)	10 ms,24 V DC		20/80 ms,120/240 V AC	
保险(不可替换)	3 A,250 V 慢速熔断		2 A,250 V 慢速熔断	
24 V DC 传感器电源				
传感器电压	L+减 5 V		20.4~28.8 V DC	
电流限定	1.5 A 峰值,终端限定非破坏性			
纹波噪声	来自输入电源		小于 1 V 峰分	
隔离(传感器与逻辑)	非隔离			



CPU 数字量输入规范如表 4-3 所列。

表 4-3 CPU 数字量输入规范

常 规	24 V DC 输入	
类 型	漏型/源型(IEC 类型 1 漏型)	
额定电压	24 V DC, 4 mA 典型值	
最大持续允许电压	30 V DC	
浪涌电压	35 V DC, 0.5 s	
逻辑 1(最小)	15 V DC, 2.5 mA	
逻辑 0(最大)	5 V DC 1 mA	
输入延迟	可选(0.2~12.8 ms) CPU 226, CPU 226XM: 输入点 11.6~12.7 具有固定延迟(4.5 ms)	
连接 2 线接近开关传感器 (Bero)允许漏电流(最大)	1 mA	
隔离(现场与逻辑)	是	
光电隔离	500 V AC, 1 min	
隔离组	见接线图	
高速输入速率(最大)	单相	两相
逻辑 1=15~30 V DC	20 kHz	10 kHz
逻辑 1=15~26 V DC	30 kHz	20 kHz
同时接通的输入	55 °C 时所有的	
电缆长度(最大)	普通输入 500 m, HSC 输入 50 m	
屏 蔽	普通输入 300 m	
非屏蔽		





CPU 数字量输出规范如表 4-4 所列。

表 4-4 CPU 数字量输出规范

常 规	24 V DC 输出	继电器输出
类 型	固态-MOSFET1	干触点
额定电压	24 V DC	24 V DC 或 250 V AC
电压范围	20.4~28.8 V DC	5~30 V DC 或 5~250 V AC
浪涌电流(最大)	8 A, 100 ms	7 A 触点闭合
逻辑 1(最小)	20 V DC, 最大电流	—
逻辑 0(最大)	0.1 V DC, 10 k Ω 负载	—
每点额定电流(最大)	0.75 A	2.0 A
每个公共端的额定电流(最大)	6 A	10 A
漏电流(最大)	10 μ A	—
灯负载(最大)	5 W	30 W DC; 200 W AC
感性嵌位电压	L+ 减 48 V DC, 1 W 功耗	—
接通电阻(接点)	0.3 Ω 最大	0.2 Ω (新的时候的最大值)
隔 离		
光电隔离(现场到逻辑)	500 V AC, 1 min	—
逻辑到接点	—	1 500 V AC, 1 min
接点到接点	—	750 V AC, 1 min
电阻(逻辑到接点)	—	100 M Ω
隔离组	见接线图	见接线图
延 时	2/10 μ s (Q0.0 和 Q0.1)	—
断开到接通/接通到断开(最大)	15/100 μ s (其他)	—
切换(最大)	—	10 ms
脉冲频率(最大) Q0.0 和 Q0.1	20 kHz	1 Hz
机械寿命周期	—	10 000 000 (无负载)
触点寿命	—	100 000 (额定负载)
同时接通的输出	55 $^{\circ}$ C 时, 所有的输出	55 $^{\circ}$ C 时, 所有的输出
两个输出并联	是	否
电缆长度(最大)		
屏 蔽	500 m	500 m
非屏蔽	150 m	150 m

4.2 S7-200 CPU 接线

CPU 224 端子接线图如图 4-1 所示。

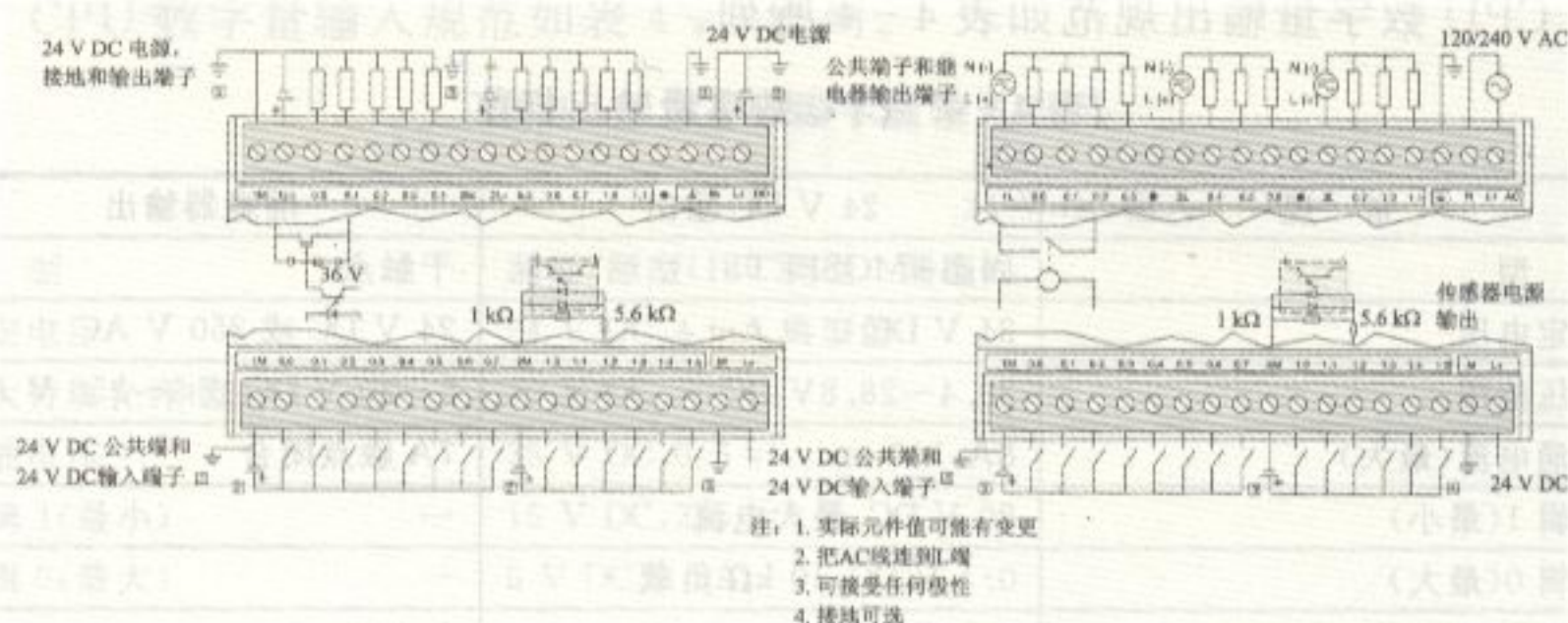


图 4-1 CPU 224 端子接线图

4.3 数字量扩展模块

数字量扩展模块规格

数字量扩展模块输入规范如表 4-5 所列。

表 4-5 数字量扩展模块输入规范

常 规	24 V DC 输入	120/230 V AC 输入(47~63 Hz)
类 型	漏型/源型(IEC 类型 1 漏型)	IEC 类型 1
额定电压	24 V DC, 4 mA	120 V AC, 6 mA 或 230 V AC, 9 mA(通常)
最大持续允许电压	30 V DC	264 V AC
浪涌电压(最大)	35 V DC, 0.5 s	—
逻辑 1(最小)	15 V DC, 2.5 mA	79 V AC, 2.5 mA
逻辑 0(最大)	5 V DC, 1 mA	20 V AC 或 1 mA AC
输入延时(最大)	4.5 ms	15 ms
连接 2 线接近开关传感器(Bero)		
允许的漏电流(最大)	1 mA	1 mA AC
隔 离		
光电隔离(现场到逻辑)	500 V AC, 1 min	1500 V AC 1 min
隔离组	见接线图	1 点
同时接通的输入	55 °C 时所有输入	55 °C 时所有输入
电缆长度(最大)		
屏蔽	500 m	500 m
非屏蔽	300 m	300 m



数字量扩展模块输出规范如表 4-6 所列。

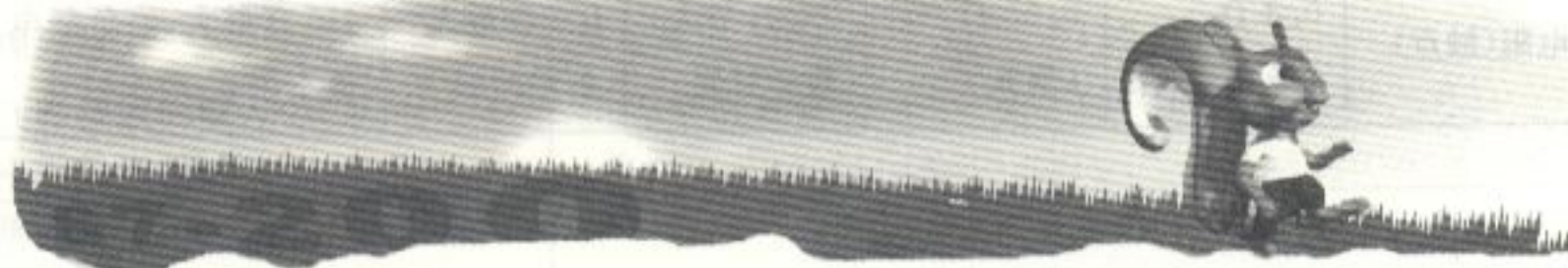
表 4-6 数字量扩展模块输出规范

常 规	24 V DC		继电器输出		120/230 V AC 输出
	0.75 A	5 A	2 A	10 A	
类 型	固态-MOSFET		干触点		过零触发
额定电压	24 V DC		24 V DC 或 250 V AC		120/230 V AC
电压范围	20.4~28.8 V DC		5~30 V DC 或 5~250 V AC		40~264 V AC (47~63 Hz)
24 V DC 线圈电源 电压范围	—		20.4~28.8 V DC		—
浪涌电流(最大)	8 A, 100 ms	30 A	7 A, 触点闭合	7 A, 触点闭合	5 A(均方根值), 2 AC 周期
逻辑 1(最小)	20 V DC		—		LI(-0.9 V(均方根值))
逻辑 0(最大)	0.1 V DC	0.2 V DC	—		—
额定电流/每点 (最大)	0.75 A	5 A	2.00 A	10 A 阻性 2 A DC 感性 3 A AC 感性	0.5 A AC3
额定电流/每个公共 点(最大)	6 A	5 A	8 A	10 A	0.5 A AC
漏电流(最大)	10 μ A	30 μ A	—		1.1 mA(均方根值), 132 V AC 时以及 1.8 mA,(均方根值), 264 V AC 时
灯负载(最大)	5 W	50 W	30 W DC /200 W AC	100 W DC /1 000 W AC	60 W
感性嵌位电压	L+减 48 V	L+减 48 V	—		—
接通状态电阻(触点)	0.3 Ω (最大)	0.05 Ω (最大)	0.2 Ω 最大, 新的时候	0.1 Ω 最大, 新的时候	410 Ω 最大,当负载电 流小于 0.05 A 时



续表 4-6

常 规	24 V DC		继电器输出		120/230 V AC 输出
	0.75 A	5 A	2 A	10 A	
隔 离					
光电隔离(现场到逻辑)	500 V AC, 1 min		—		1 500 V AC, 1 min
线圈到逻辑	—		无		—
线圈到触点	—		1500 V AC, 1 min		—
触点到触点	—		750 V AC, 1 min		—
电阻(线圈到触点)	—		100 M Ω 最小, 新		—
隔离组	见接线图		4 点		1 点
延 时					
断开到接通/接通到	50 μ s 最大	500 μ s 最大	—	—	0.2 ms + 1/2 AC 周期
断 开	/200 μ s	/200 μ s	10 ms	10 ms	—
切换(最大)	—	—	—	—	—
切换频率(最大)	—		1 Hz		10 Hz
机械寿命周期	—		10 000 000 (无负载)	30 000 000 (无负载)	—
触点寿命	—		100 000 (额定负载)	300 000 (额定负载)	—
同时接通的输出	55 $^{\circ}$ C 所有输出		55 $^{\circ}$ C 时所有输出	55 $^{\circ}$ C 时所有输出	55 $^{\circ}$ C 时所有输出
并联两个输出	是		否		否
电缆长度(最大)					
屏 蔽	500 m		500 m		500 m
非屏蔽	150 m		150 m		150 m





数字量扩展模块输出规范(大电流型)如表 4-7 所列。

表 4-7 数字量扩展模块输出规范

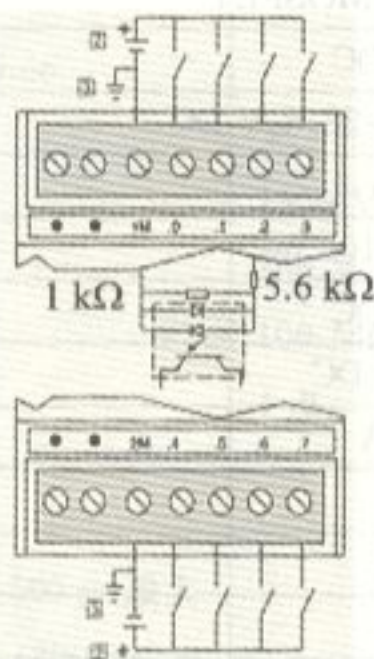
常 规	4 V DC 输出	继电器输出
类 型	固态-MOSFET	干触点
额定电压	24 V DC	24 V DC 或 250 V AC
电压范围	20.4~28.8 V DC	5~30 V DC 或 5~250 V AC
24 V DC 线圈电源电压范围	—	20.4~28.8 V DC
浪涌电流(最大)	30 A, 100 ms	7 A, 触点闭合
逻辑 1(最小)	20 V DC	—
逻辑 0(最大)	0.1 V DC	—
额定电流/每点(最大)	0.75 A	2.00 A
额定电流/每个公共点(最大)	6 A	8 A
漏电流(最大)	10 μ A	—
灯负载(最大)	5 W	30 W DC/200 W AC
感性嵌位电压	L+ 减 48 V	—
接通状态电阻(触点)	0.3 Ω (最大)	0.2 Ω 最大, 新的时候
隔 离		
光电隔离(现场到逻辑)	500 V AC, 1 min	—
线圈到逻辑	—	无
线圈到触点	—	1 500 V AC, 1 min
触点到触点	—	750 V AC, 1 min
电阻(线圈到触点)	—	100 M Ω 最小, 新的时候
隔离组	见接线图	4 点
延 时		
断开到交通/接通到断开	50 μ s 最大/200 μ s	—
切换(最大)	—	10 ms
切换频率(最大)	—	1 Hz
机械寿命周期	—	10 000 000(无负载)
触点寿命	—	100 000(额定负载)
同时接通的输出	55 $^{\circ}$ C 所有输出	55 $^{\circ}$ C 时所有输出
并联两个输出	是	否
电缆长度(最大)		
屏 蔽	500 m	500 m
非屏蔽	150 m	150 m



数字量扩展模块端子接线图

数字量扩展模块端子接线图如图 4-2~图 4-10 所示。

24 V DC 公共端和
24 V DC 输入端

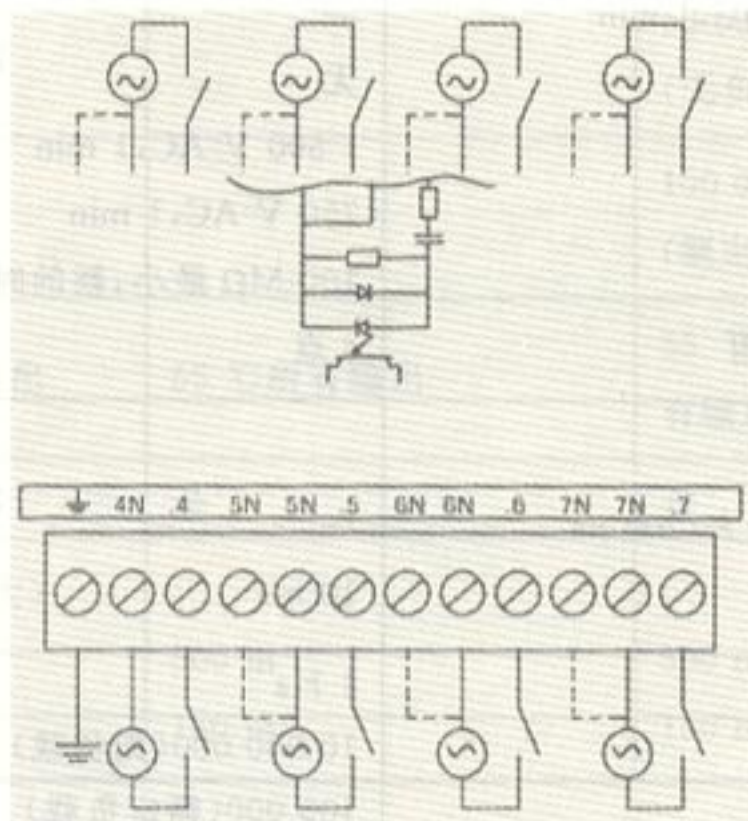


注:

1. 实际元件值可能有变更;
2. 可接受任何极性;
3. 接地可选

24 V DC 公共端和
24 V DC 输入端

图 4-2 EM221 数字输入 8×24 V DC 连接器端子图(6ES7 221-1BF22-0XA0)

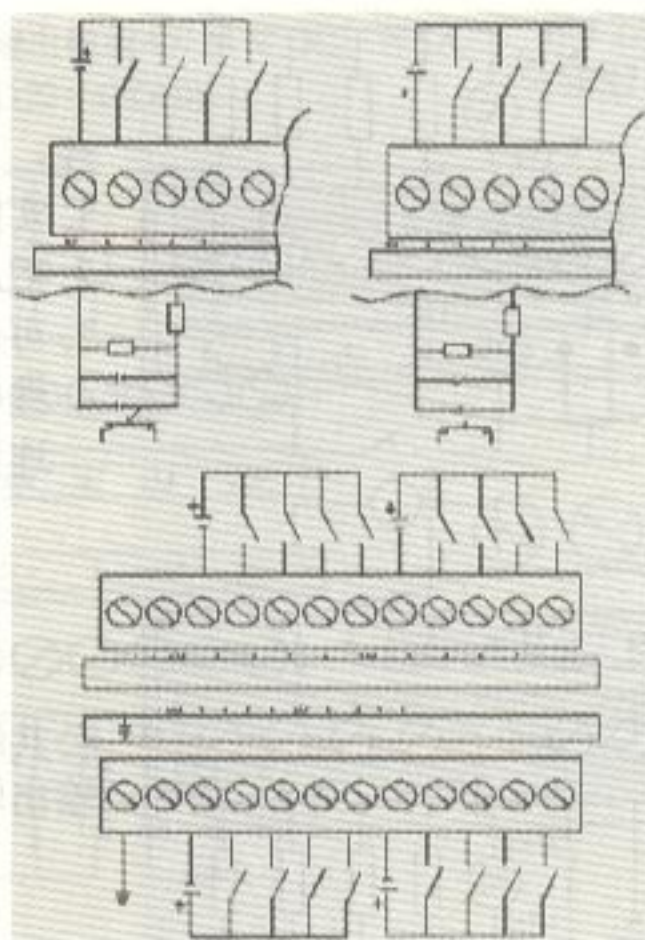


120/230 AC 输入
EM 221 数字量输入 8×AC
120/230 V

图 4-3 EM221 数字输入 8×120/230 V AC 连接器端子图(6ES7 221-1EF22-0XA0)



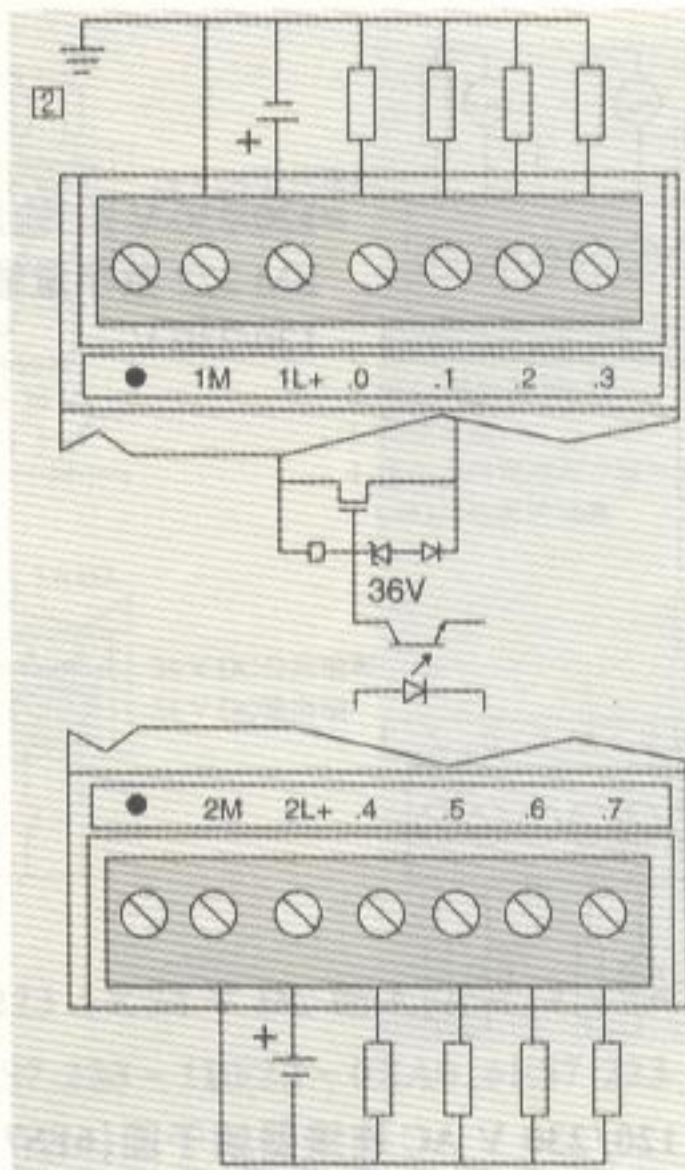
24 V DC输入
用作漏输入



24 V DC输入
用作源输入

图 4-4 EM221 数字量输入模板, 16×24 V DC (6ES7 221-1BH22-0XA)

24 V DC 公共端和
24 V DC 输入端



注:

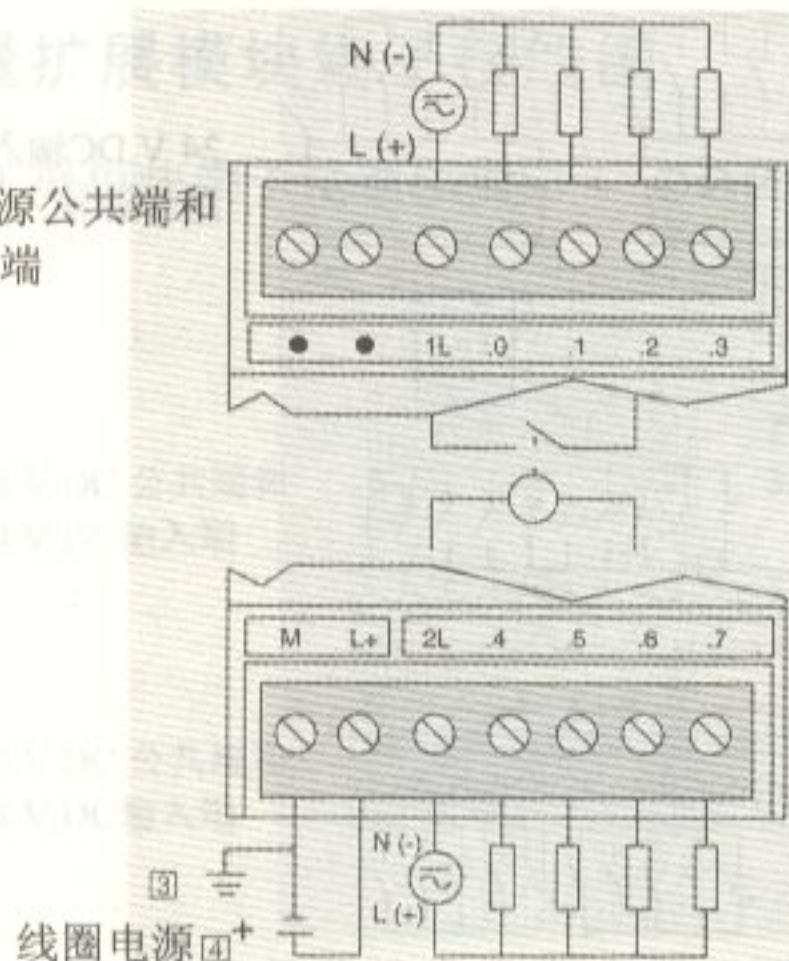
1. 实际元件值可能有变更;
2. 接地可选

24 V DC 公共端和
24 V DC 输入端

图 4-5 EM222 数字输出 8×24 V DC 连接端子图 (6ES7 222-1BF22-0XA0)



24 V DC 电源公共端和
继电器输出端

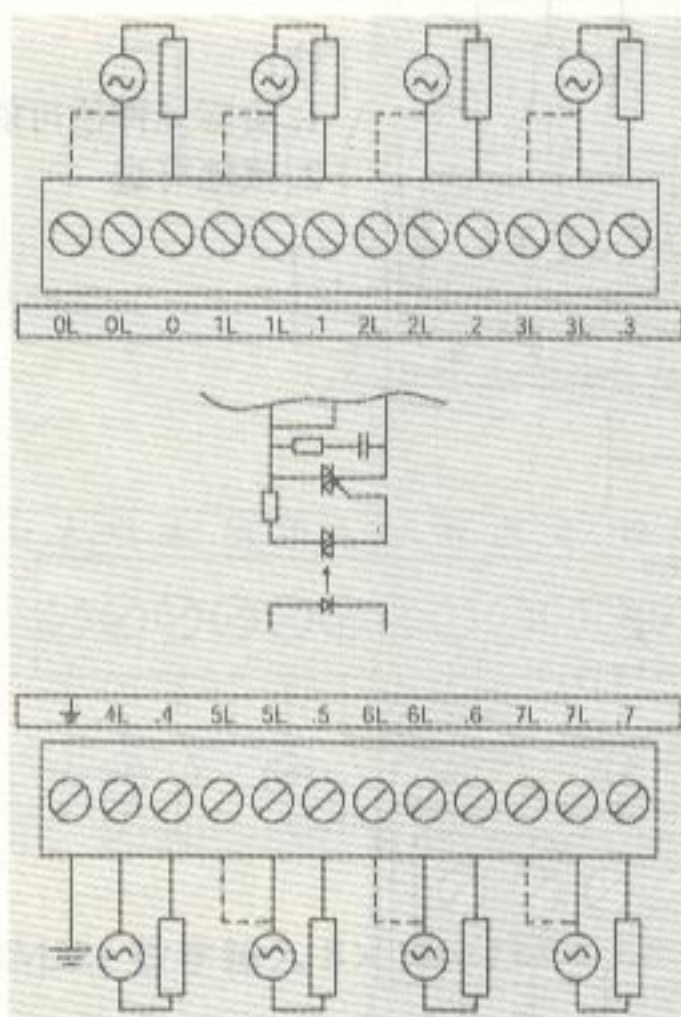


注:

1. 实际元件值可能有变更;
2. 把AC线连接到L端;
3. 可选接地;
4. 继电器线圈电源的M一定要连到CPU的电源的M端

公共端和输出继电器
输出端

图 4-6 EM222 数字输出 8×24 V 继电器连接端子图(6ES7 222-1HF22-0XA0)



120/230 V AC输出
EM 222 数字量输出8×AC
120/230 V

图 4-7 EM222 数字输出 8×120/230 V AC 连接器端子图(6ES7 222-1EF22-0XA0)



EM 222数字量输出模板4×24 V DC——5A
(6ES7 222-1BD22-0XA0)

EM 222数字量输出模板4×继电器—10A
(6ES7 222 1HD22-0XA0)

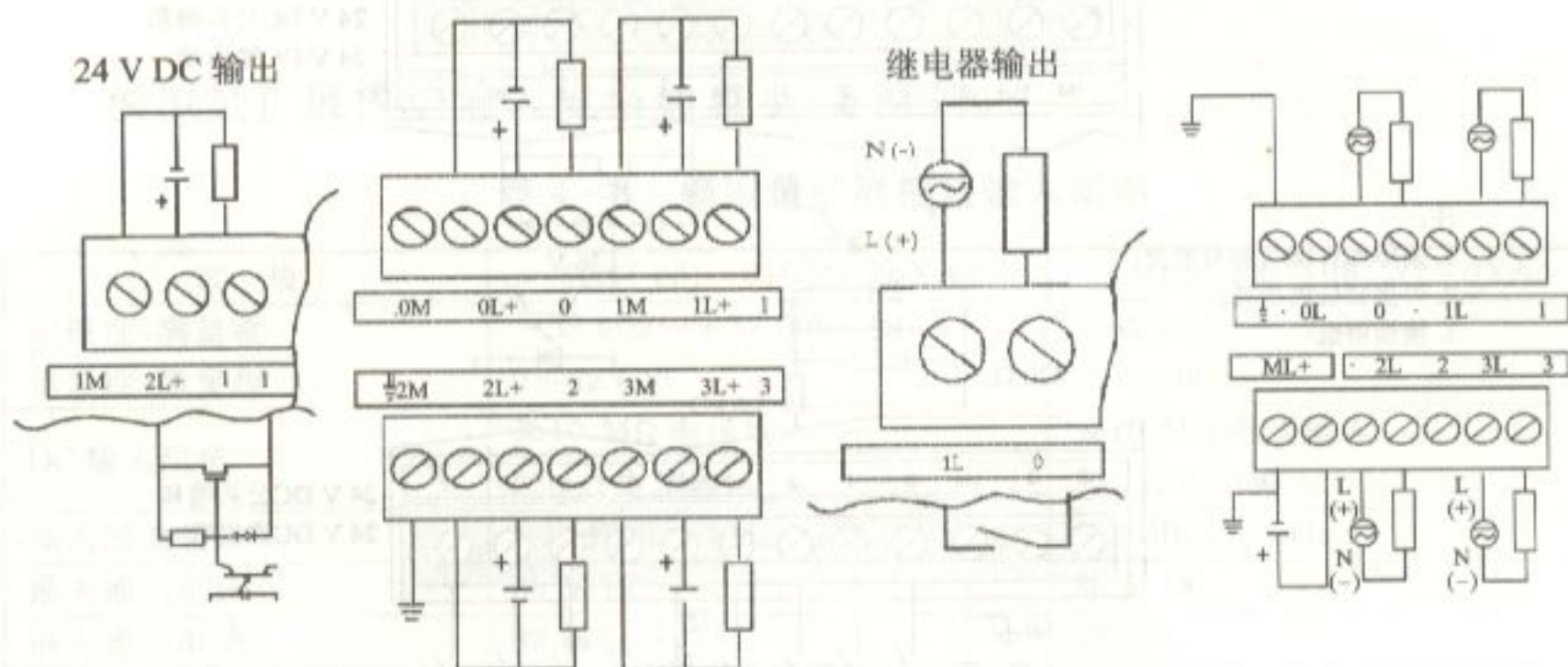


图 4-8 EM222 数字量输出模板 4×24 V DC-5 A、4×继电器-10 A
(6ES7 222-1BD22-0XA0、6ES7 222-1HD22-0XA0)

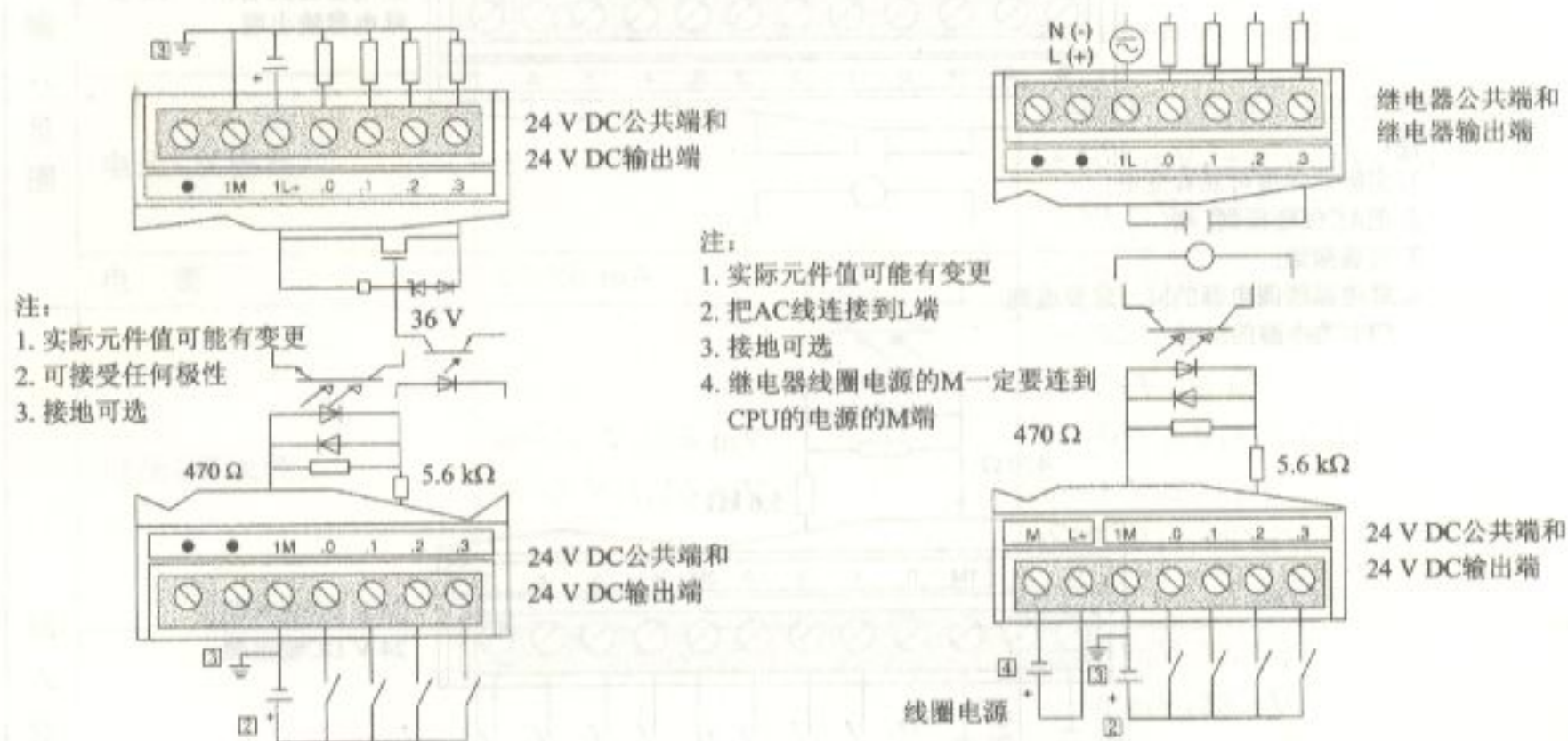


图 4-9 EM223 4×24 V DC 输入/4×24 V DC、继电器输出端子连接图
(6ES7 223-1BF22-0XA0、6ES7 223-1HF22-0XA0)

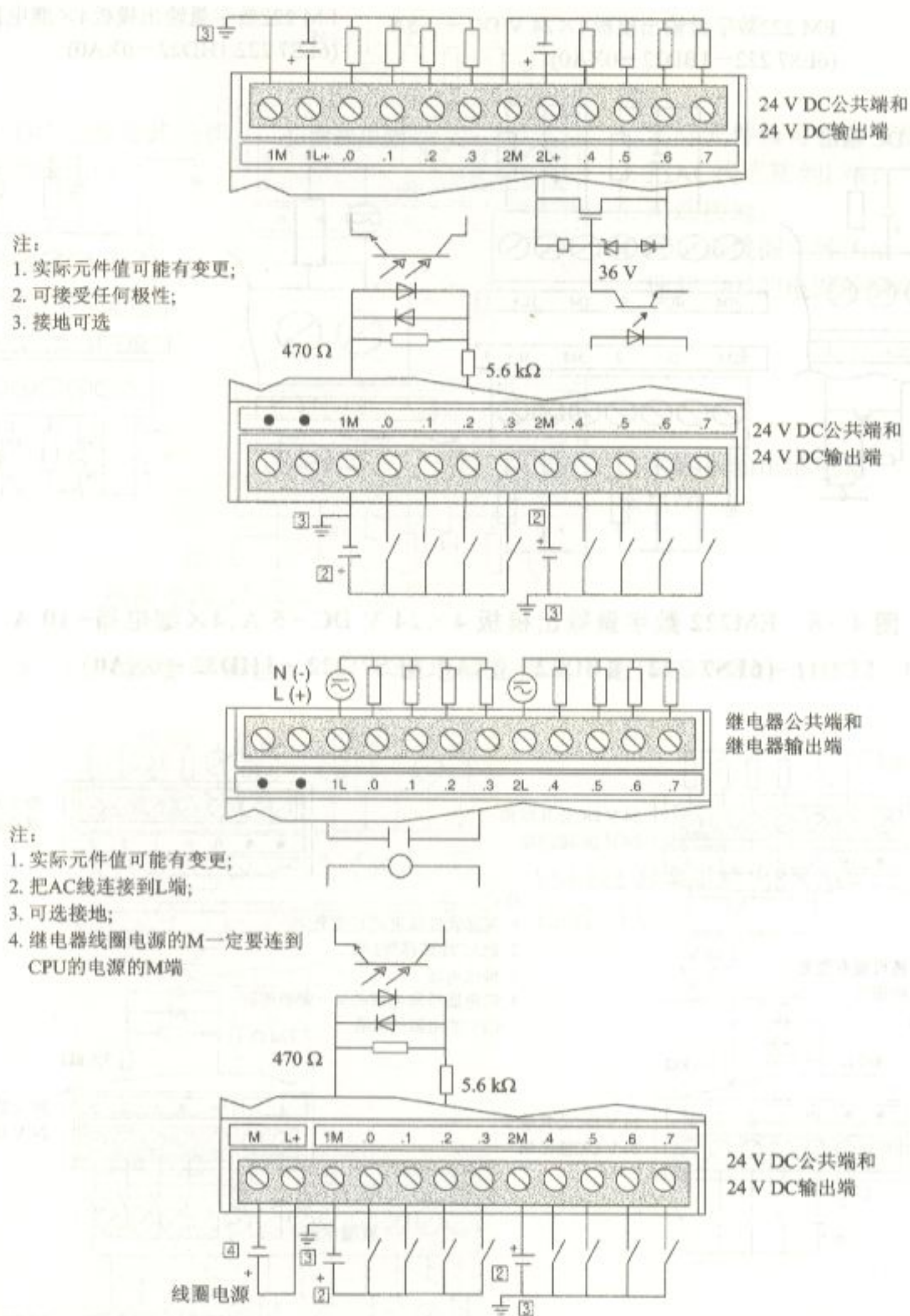


图 4-10 EM223 8×24 V DC 输入/8×24 V DC、继电器输出连接端子图
(6ES7 223-1BH22-0XA0、6ES7 223-1PH22-0XA0)



模拟量扩展模块

模拟量扩展模块输入规格如表 4-8 所列。

表 4-8 模拟量扩展模块输入规格

常 规		6ES7 231-0HC22-0XA0	6ES7 235-0KD22-0XA0
双极性,满量程		-32 000~+32 000	-32 000~+32 000
单极性,满量程		0~32 000	0~32 000
DC 输入阻抗		≥10 MΩ 电压输入 250 Ω 电流输入	≥10 MΩ 电压输入 250 Ω 电流输入
输入滤波衰减		-3 dB,3.1 kHz	-3 dB,3.1 kHz
最大输入电压		30 V DC	30 V DC
最大输入电流		32 mA	32 mA
分辨率		12 位 A/D 转换器	12 位 A/D 转换器
隔离(现场到逻辑)		否	否
输入类型		差 分	差 分
输入范围	电压(单极性)	0~10 V,0~5 V	0~10 V,0~5 V 0~1 V,0~500 mV 0~100 mV,0~50 mV
	电压(双极性)	±5 V,±2.5 V	±10 V,±5 V,±2.5 V,±1 V, ±500 mV,±250 mV,±100 mV, ±50 mV,±25 mV
	电 流	0~20 mA	0~20 mA
输入分辨率	电压(单极性)	0~10 V,2.5 mV 0~5 V,1.25 mV	0~50 mV,12.5 μV 100 mV,25 μV 0~500 mV,125 μV 0~1 V,250 μV 0~5 V,1.25 mV 0~10 V,2.5 mV
	电压(双极性)	±5 V,2.5 mV ±2.5 V,1.25 mV	±25 mV,12.5 μV ±50 mV,25 μV ±100 mV,50 μV ±250 mV,125 μV ±500 mV,250 μV ±1 V,500 μV ±2.5 V,1.25 mV ±5V,2.5 mV ±10 V,5 mV
	电 流	0~20 mA,5 μA	0~20mA,5 μA



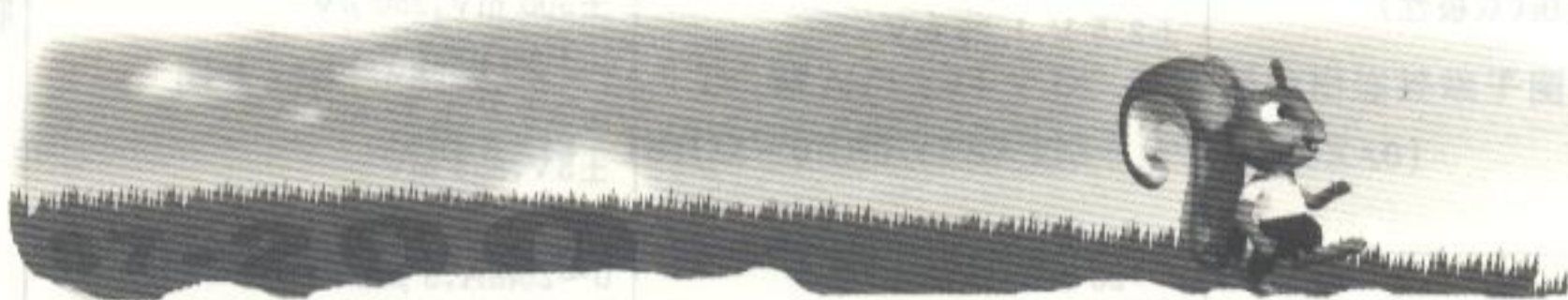
续表 4-8

常 规	6ES7 231-0HC22-0XA0	6ES7 235-0KD22-0XA0
模拟到数字转换时间	$<250 \mu\text{s}$	$<250 \mu\text{s}$
模拟输入阶跃响应	1.5 ms~95%	1.5 ms~95%
共模抑制	40 dB, DC~60 Hz	40 dB, DC~60 Hz
共模电压	信号电压加共模电压必须 $\leq \pm 12 \text{ V}$	信号电压加共模电压必须 $\leq \pm 12 \text{ V}$
24 V DC 电压范围	20.4~28.8	20.4~28.8

模拟量扩展模块输出规范如表 4-9 所列。

表 4-9 模拟量扩展模块输出规范

常 规			6ES7 232-0HB22-0XA0	6ES7 232-0KD22-0XA0
隔离(现场到逻辑)			无	无
信号 范围	电压输出		$\pm 10 \text{ V}$	$\pm 10 \text{ V}$
	电流输出		0~20 mA	0~20 mA
分辨率, 满量程	电 压		12 位	12 位
	电 流		11 位	11 位
数据字 格式	电 压		-32 000~+32 000	-32 000~+32 000
	电 流		0~+32 000	0~+32 000
精 度	最差情况 0~55℃	电压输出	$\pm 2\%$ 满量程	$\pm 2\%$ 满量程
		电流输出	$\pm 2\%$ 满量程	$\pm 2\%$ 满量程
	典型, 25℃	电压输出	$\pm 0.5\%$ 满量程	$\pm 0.5\%$ 满量程
		电流输出	$\pm 0.5\%$ 满量程	$\pm 0.5\%$ 满量程
	设置时间	电压输出	100 μs	100 μs
		电流输出	2 ms	2 ms
最大 驱动	电压输出		5 000 Ω 最小	5 000 Ω 最小
	电流输出		500 Ω 最大	500 Ω 最大





模拟量扩展模块端子接线图(如图 4-11 所示)

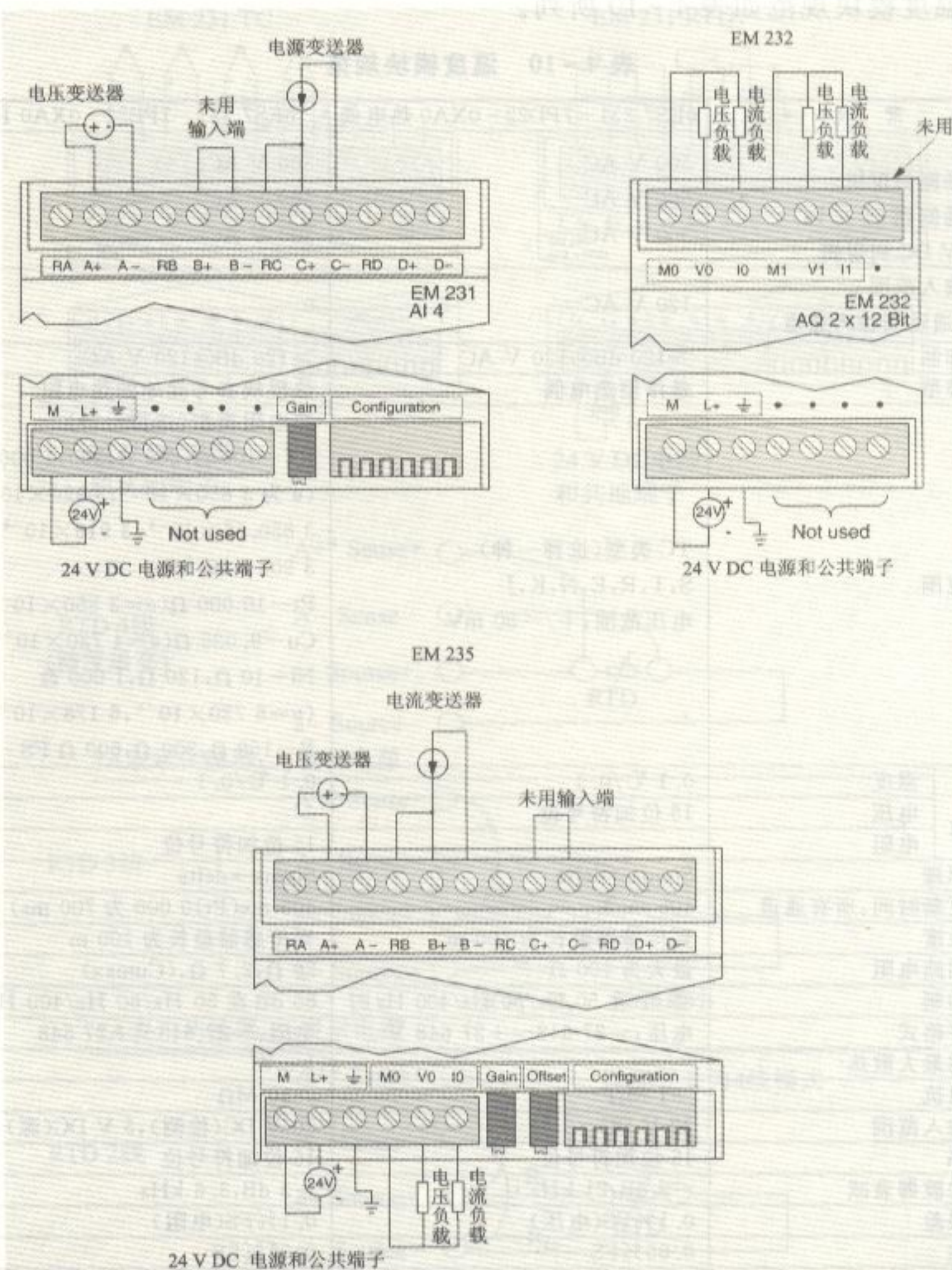


图 4-11 模拟量扩展模块端子接线图

温度测量模块

温度模块规范如表 4-10 所列。

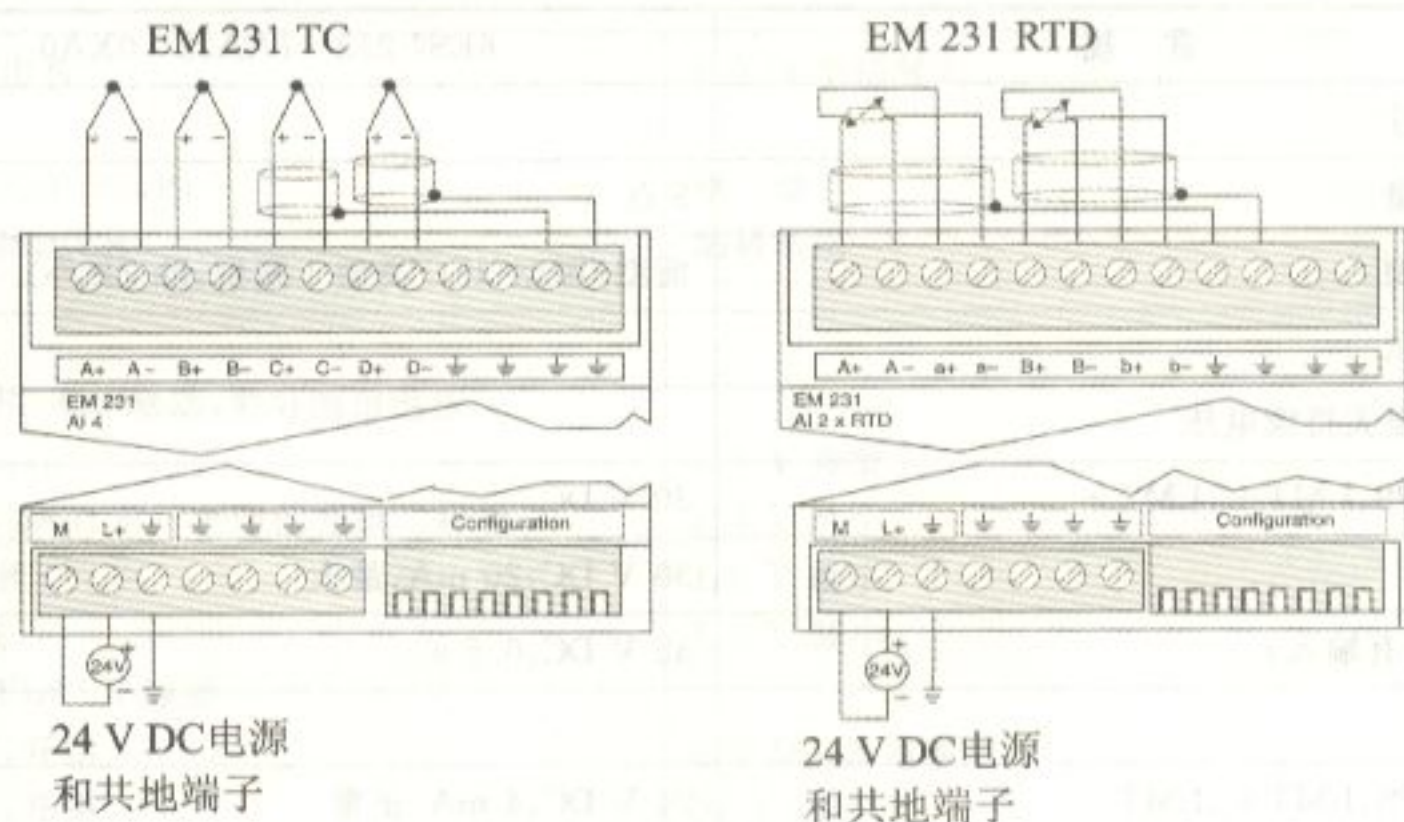
表 4-10 温度模块规范

常 规		6ES7 231-7PD22-0XA0 热电偶	6ES7 231-7PB22-0XA0 RTD
隔离	现场侧到逻辑	500 V AC	500 V AC
	现场侧到 24 V DC	500 V AC	500 V AC
	24 V DC 到逻辑	500 V AC	500 V AC
	共模输入范围 (输入通道到输入通道)	120 V AC	0
共模抑制		>120 dB@120 V AC	>120 dB@120 V AC
输入类型		悬浮型热电偶	热模块参考接地的热电阻
输入范围		TC 类型(选择一种) S,T,R,E,N,K,J 电压范围: ± 80 mV	热电阻类型(选择一种) PT-100 Ω , 200 Ω , 500, 1 000 Ω (α 为 3.850×10^{-6} , 3.920×10^{-6} , $3.850.55 \times 10^{-6}$, 3.916×10^{-6} , 3.902×10^{-6}) Pt-10 000 Ω ($\alpha = 3.850 \times 10^{-6}$) Cu-9.035 Ω ($\alpha = 4.720 \times 10^{-6}$) Ni-10 Ω , 120 Ω , 1 000 Ω ($\alpha = 6.720 \times 10^{-6}$, 6.178×10^{-6}) R-150 Ω , 300 Ω , 600 Ω FS
输入 分辨率	温度	0.1 $^{\circ}\text{C}/0.1$	0.1 $^{\circ}\text{C}/0.1$
	电压	15 位加符号位	
	电阻		15 位加符号位
测量原理		Sigma \rightarrow delta	Sigma \rightarrow delta
模块更新时间:所有通道		405 ms	405 ms(Pt10 000 为 700 ms)
导线长度		到传感器最长为 100 m	到传感器最长为 100 m
导线回路电阻		最大为 100 Ω	20 Ω , 2.7 Ω , (Cumax)
干扰抑制		85 dB 在 50 Hz/60 Hz/400 Hz 时	85 dB 在 50 Hz/60 Hz/400 Hz 时
数据字格式		电压: $-27\,648 \sim +27\,648$ V	电阻: $-27\,648 \sim +27\,648$
传感器最大散热			1 mW
输入阻抗		>1 M Ω	>10 M Ω
最大输入范围		30 V DC	30 V DC(检测), 5 V DC(源)
分辨率		15 位加符号位	15 位加符号位
输入滤波器衰减		-3 dB, 21 kHz	-3 dB, 3.6 kHz
基本误差		0.1%FS(电压)	0.1%FS(电阻)
重复性		0.05%FS	0.05%FS
冷端误差		± 1.5 $^{\circ}\text{C}$	
24 V DC 提供电压范围		20.4~28.8 V DC	20.4~28.8 V DC

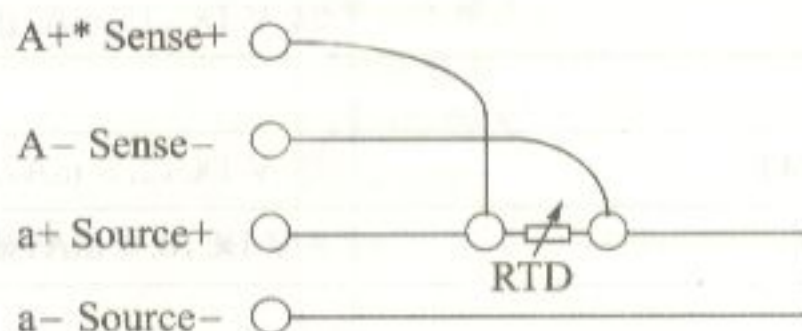
注:输入范围的选择(温度、基于阻抗的电压)对于一个模块的所有通道有效。



温度扩展模块端子接线图(如图 4-12 所示)

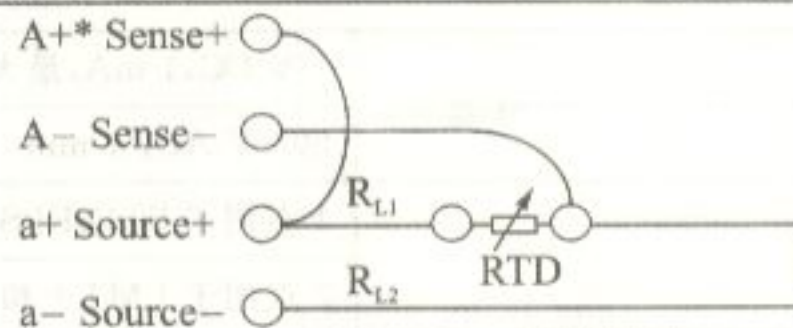


RTD 4线
(精度最高)



*A表示传感器; a表示电源

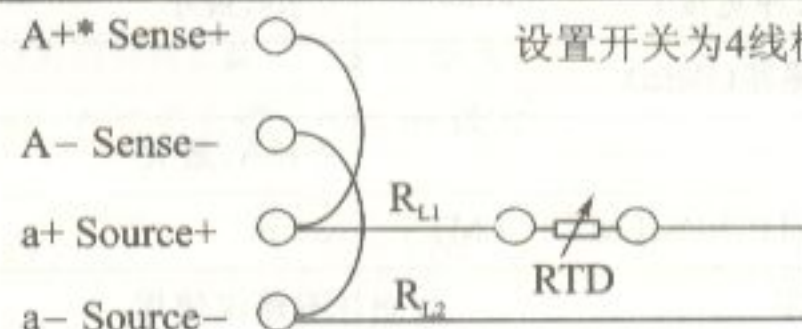
RTD 3线



If $R_{L1}=R_{L2}$, 错误最小

*A表示传感器; a表示电源

RTD 2线



$R_{L1}+R_{L2}=\text{Error}$

*A表示传感器; a表示电源

设置开关为4线模式

图 4-12 温度扩展模块端子接线图



EM 253 位控模块规范如表 4-11 所列。

表 4-11 EM 253 位控模块规范

常 规		6ES7 253-1AA22-0XA0	
输入特性			
输入数量		5 点	
输入类型		漏型/源型(IEC 类型 1 漏型,除 ZP 外)	
输入电压			
允许的最大持续电压			
STP、RPS、LMT+、LMT-		30 V DC	
ZP		30 V DC,20 mA,最大	
浪涌(所有输入)		35 V DC,0.5 s	
额定值			
STP、RPS、LMT+、LMT-		24 V DC,4 mA,正常	
ZP		24 V DC,15 mA,正常	
逻辑“1”信号(最小)			
STP、RPS、LMT+、LMT-		15 V DC,2.5 mA,最小	
ZP		3 V DC,8.0 mA,最小	
逻辑“0”信号(最大)			
STP、RPS、LMT+、LMT-		5 V DC,1 mA,最大	
ZP		1 V DC,1 mA,最大	
隔离(现场到逻辑)	光电隔离	500 V AC, 1 min	
	组隔离	1 点用于 STP、RPS 和 ZP	
		2 点用于 LMT+ 和 LMT-	
输入延迟时间	STP、RPS、LMT+、LMT-	0.2~12.8 ms,用户选择	
	ZP(可计脉冲宽度)	2 μs,最小	
连接 2 线接近开关传感器(Bero)			
允许的源电流		1 mA,最大	
电缆长度	未屏蔽	STP、RPS、LMT+、LMT-	30 m
		ZP	不建议使用
	屏 蔽	STP、RPS、LMT+、LMT-	100 m
		ZP	10 m
同时接通的输入数(55 ℃)		5	



续表 4-11

常 规		6ES7 253-1AA22-0XA0
输出特性		
集成的输出数		6 点(4 个信号)
输出字节		
P0+, P0-, P1+, P1-		驱 动
P0, P1, DIS, CLR		漏极输出
输出电压		
P0, P1, RS-422 驱动, 差分输出电压		
断 路		3.5 V 典型
接入带有 200 Ω 串行电阻的光耦二极管		2.8 V 最小
100 Ω 负载		1.5 V 最小
54 Ω 负载		1.0 V 最小
P0, P1, DIS, CLR 漏型		
建议电压, 开路		5 V DC, 来自模块
允许电压, 开路		30 V DC1
漏电流		50 mA 最大
接通状态电阻		15 Ω 最大
断开状态下漏电流, 30 V DC		10 μ A 最大
电拉电阻, 到 T1 的漏型输出		3.3 k Ω 2
输出电流		
输出组数		1
接通的输出数(最大)		6
每点漏电流		
P0, P1, DIS, CLR		10 μ A 最大
过载保护		无
隔离(现场到逻辑)		
光电隔离		500 V AC, 1 min
输出时延		
DIS, CLR; 断开到接通/接通到断开		30 μ s, 最大
脉冲	P0, P1, 输出, RS-422 驱动, 100 Ω 外部负载	75 ns 最大
畸变	P0, P1 输出, 漏型, 5 V/470 Ω 外部负载	300 ns 最大
切换频率		
P0+, P0-, P1+, P1-, P0 和 P1		200 kHz
电缆	未屏蔽	不推荐
长度	屏 蔽	10 m



续表 4-11

常 规		6ES7 253 - 1AA22 - 0XA0	
电 源			
L+ 提供电压		11 - 30 V DC	
Logic 提供输出		+5 V DC + / - 10 % , 200 mA 最大	
L+ 提供相对于 5 V DC 负载的电流		12 VDC 输入	24 V DC 输入
负载电流		120 mA	70 mA
0 mA (无负载)		300 mA	130 mA
200 mA (额定负载)			
隔 离			
L+ 电源到逻辑		500 V AC, 1 min	
L+ 电源到输入		500 V AC, 1 min	
L+ 电源到输出		无	
反向极性		L+ 输入和 +5 V 输出有二极管保护。在 M 端接入正向电压, 就输出点的连接而言, 可能导致损害性的电流产生。	

注: 1. 高于 5 V DC 的漏型输出可能会增加射频干扰使之超过允许的限定。系统或接线需要射频干扰抑制措施;

2. 根据脉冲接收器和电缆, 一个额外的外部上拉电阻可能会改善脉冲信号的质量和噪声抑制功能。

S7-200PLC

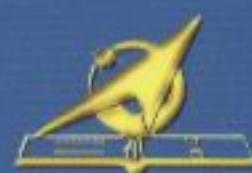


深入浅出西门子S7-200PLC

ISBN 7-81077-426-3



9 787810 774260 >



ISBN 7-81077-426-3

定价：25.00元(含光盘)