

PROFIBUS-OEM2 调试实验系统

用户手册

V 1.0



北京鼎实创新科技有限公司

目 录

第一章 PROFIBUS-OEM2 调试实验系统概述	4
1. 什么是嵌入式 PROFIBUS 接口 OEM2	4
2. OEM2 调试实验系统主要用途.....	4
3. 调试实验系统调试模式	4
(1) 调试模式 1: 实验板调试模式	4
(2) 调试模式 2: OEM2 接口板调试模式.....	4
(3) 调试模式 3: PC 机调试模式.....	5
4. OEM2 调试实验系统主要部件及选件.....	6
(1) 基本配置	6
(2) 主站部分	6
5. 安全指南	6
第二章 OEM2 开发实验板	7
1. PB-OEM2-SAMPLE 布局图	7
2. OEM2 开发实验板原理图	9
(1) 原理图-1.....	14
(2) 原理图-2.....	14
(3) 原理图-3.....	14
(4) 原理图-4.....	14
第三章 实验板程序、GSD 文件	15
1. 实验板程序	15
(1) 实验板 CPU	15
(2) 实验板程序	15
2. 关于 GSD 文件 (Electronic Data Sheet)	15
3. PB-OEM2-SE 的 ID 号及 GSD 文件	16
4. 用户产品的 ID 号、GSD 文件及产品认证	16
第四章 实验板程序举例	18
例 1. 简单工作模式	18
(1) 技术参数	18
(2) SE 初始化报文.....	18
(3) 数据交换状态下实验板与 SE 接口的数据报文.....	19
(4) 实验板初始化 SE 接口板的过程.....	19
(5) 数据交换状态下实验板简单工作模式.....	20
(6) 实验板程序清单	20
(7) 例 1 的 GSD 文件	29
例 2. 监测 SE 接口状态的工作模式.....	31
(1) 技术参数:	31
(2) SE 初始化报文.....	31
(3) 数据交换状态下实验板与 SE 接口的数据报文.....	32
(4) SE 接口状态信号 REQ_IT 与 S_RTS/P17 的时序图.....	32
(5) 实验板初始化 SE 接口板的过程.....	33
(6) 数据交换状态下实验板带监测接口状态功能工作模式.....	34
(7) 实验板程序清单	35

(8) 例 2 的 GSD 文件	44
例 3. 带用户参数功能的工作模式	46
(1) 需要使用“用户参数 user_prm”的情况	46
(2) 实验 I/O 功能及实现方法	46
(3) 具体确定“用户参数”类型、个数、取值范围	46
(4) 带有“用户参数”描述的 GSD 文件	47
(5) 如何在主站配置中选择用户参数	49
(6) 更便于用户使用的 GSD 文件	51
(7) 带用户参数功能的 SE 接口的初始化	56
(8) 带用户参数功能数据交换过程	57
(9) 实验板初始化 SE 接口板的过程框图	59
(10) 数据交换状态下实验板工作流程	59
(11) 例 3 实验 I/O 的程序清单	61
第五章 建立一个调试实验系统	73
1. 建立 OEM2 调试实验系统 I	73
(1) 系统 I 设备清单	73
(2) 系统 I 结构图	74
(3) 安装实验调试系统 I	74
(一) 硬件安装	74
(二) 主站 PC 机软件安装	74
(4) 按照系统提供的例 2 实现系统连通实验	75
(一) 在 COM PROFIBUS 配置软件中实现简单连通	75
(二) 自己完成配置例 2 实现简单连通	80
2. CP5611+Wincc 做主站实现系统 I 的 PROFIBUS 通信	81
(1) 设置“Set PG/PC Interface...”:	81
(2) 建立 WinCC 项目	83
(3) 运行 WinCC	83
3. 建立 OEM2 调试实验系统 II	84
(1) 系统 II 设备清单	84
(2) 系统 II 结构图	85
(3) 安装实验调试系统 II	85
(一) 硬件安装	85
(二) 主站 PC 机软件安装	85
(4) 按照系统提供的例 2 实现系统连通实验	86
(5) 在 STEP 7 中观察 PROFIBUS 输入输出变化	88
(6) PROFIBUS 输入/输出数据地址	89
(7) 自己完成例 2 配置实现系统连通实验	90
4. 三种调试模式	95
(1) 调试模式 1: 实验板调试模式	95
① 主要调试功能	95
② MODE 开关设置	96
③ 操作方法	96
(2) 调试模式 2: OEM2 接口板调试模式	96
① 主要调试功能	96
② MODE 开关设置	96
③ 操作方式	97

(3) 实验板程序下载	98
① 主要调试功能	98
② MODE 开关设置.....	99
③ 操作方式	99
(4) 调试模式 3: PC 机调试模式.....	99
① 主要调试功能	99
② MODE 开关设置.....	99
③ 操作方式	99
附件 A: 下载实验板程序操作指南及下载线制作方法	104
1. 实验板的下载模式选择	104
2. Flash Magic 软件的程序下载操作.....	104
3. 恢复实验板运行状态	106
4. 下载线的制作及连接方法	106

第一章 PROFIBUS-OEM2 调试实验系统概述

OEM2 调试实验系统（以下简称“**调试系统**”），专为应用嵌入式 PROFIBUS 通信接口 OEM2 的用户，提供一个完整、最小化、最便捷的调试实验系统平台。

1. 什么是嵌入式 PROFIBUS 接口 OEM2

嵌入式 PROFIBUS 接口 OEM2（以下有时称 OEM2 卡）专为自主开发具有 PROFIBUS-DP 通信功能产品的用户，以 OEM 方式提供从站的通信接口。产品型号是 PB-OEM2-SE，《PB-OEM2-SE 产品手册》中有详细使用及原理说明。

2. OEM2 调试实验系统主要用途

- (1) 为 PROFIBUS 产品开发用户提供一个完整、最小化、最便捷的调试实验平台。
- (2) OEM2 调试实验系统包括：主站、从站实验板、PROFIBUS 电缆及接插件、相关配置组态和调试软件、产品开发例程等。调试实验系统共提供 3 种调试模式。
- (3) 系统包括一块实验板。这是 OEM2 卡的评估板，其硬件原理图和 C 代码源程序完全公开，为用户产品开发提供样板。

3. 调试实验系统调试模式

(1) 调试模式 1：实验板调试模式

主要调试功能：建立主站、从站的配置、PROFIBUS 系统的连通、PROFIBUS I/O 通信。用户可以先使用实验板按照本手册完成与 PROFIBUS 主站的连接,然后再将自主开发的用户板与 PROFIBUS 主站连接,见图 1-1。

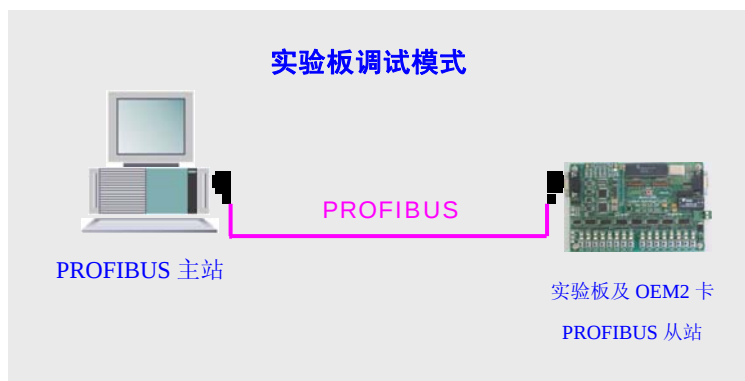


图 1-1 调试模式 1—实验板调试模式

(2) 调试模式 2：OEM2 接口板调试模式

主要调试功能：监测实验板与 OEM2 接口板的通信数据,可检验初始化数据、PROFIBUS I/O 数据、用户参数数据等,是学习、调试 OEM2 接口板的主要方法,见图 1-2。

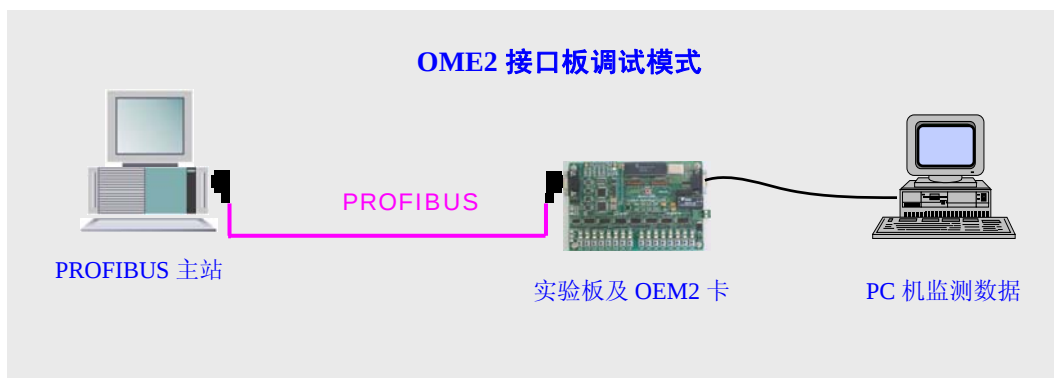


图 1-2 调试模式 2—OEM2 接口板调试模式

实验板程序下载：可将实验板 CPU 的 C 源程序编译、连接生成 .HEX 目标程序，通过软件 WINISP 下载到实验板中。此工作可以配合调试模式 2 使用，用户可以自行修改实验板的程序，实现不同 PROFIBUS 配置的连接,见图 1-3。

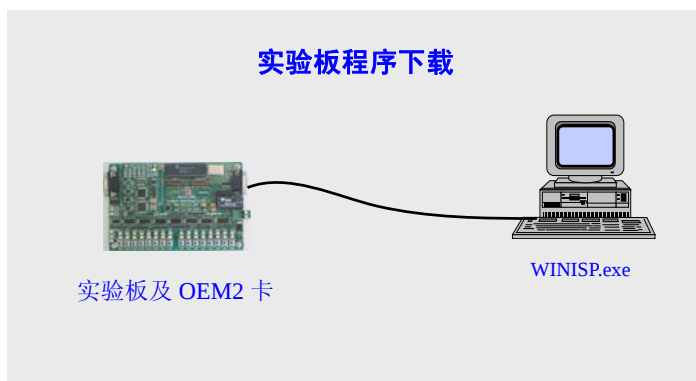


图 1-3 实验板程序下载

(3) 调试模式 3：PC 机调试模式

主要调试功能：“调试模式 3”可替代“调试模式 2”方式。使用 PC 机 OEM2 调试软件：oem2_test.exe，用 PC 机代替实验板 CPU，实现与 OEM2 接口板的数据交换。这种调试模式可方便设定不同初始化数据、设定 PROFIBUS 输入数据，监测主站的 PROFIBUS 输出数据、用户参数数据，是学习、调试 OEM2 接口板的另一种方法，见图 1-4。

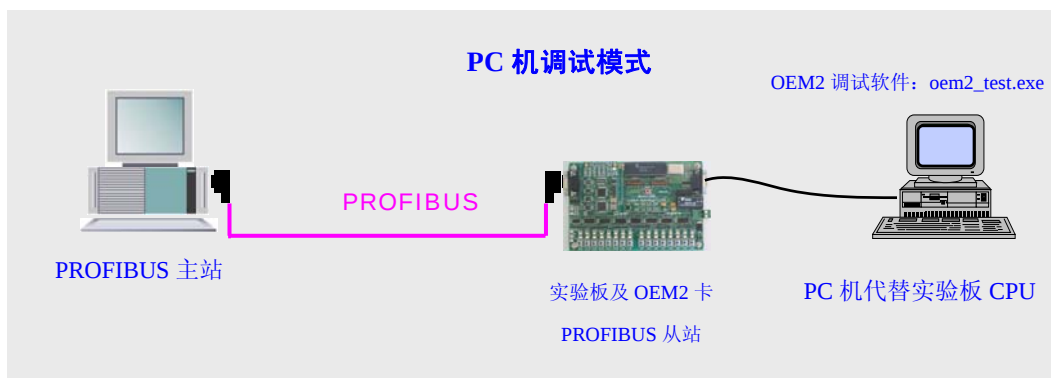


图 1-4 调试模式 3—PC 机调试模式

4. OEM2 调试实验系统主要部件及选件

(1) 基本配置

PROFIBUS-OEM2 调试实验系统基本配置（从站部分必选）				
序号	名称	制造商	数量	备注
1	PB-OEM2-SE 嵌入式 PROFIBUS 接口	鼎实科技	1 块	
2	PB-OEM2-SAMPLE OEM2 开发实验板	鼎实科技	1 块	
2	PROFIBUS 电缆插头：10 米电缆+2 插头	西门子	1 套	
3	文件资料（光盘）包括： 手册、例程 1~3（COM PROFIBUS 配置、实验板 C 源 及目标程序、系统 I WinCC 项目文件、系统 II Step7 项 目文件、GSD 文件）及 GSD 文件、PC 机调试模式软件、 单片机程序下载软件、实验板硬件原理图、SE 外转接口 原理图。	鼎实科技	1 张 CD	随系统赠送
4	手册（印刷品）： 《PB-OEM2-SE 产品手册》 《PROFIBUS-OEM2 调试实验系统使用手册》	鼎实科技	1 套	随系统赠送

(2) 主站部分

主站部分有 2 种选择：

① 选择 CP5611+PC 机做主站，优点：价格低

选择 CP5611+PC 机做主站				
序号	名称	制造商	数量	备注
1	CP5611 PROFIBUS 主站网卡	西门子	1 块	
3	组态软件（光盘）： COM PROFIBUS V5.1(DEMO 版)； SOFTNET-DP V5.2(DEMO 版)； WinCC V5.1(DEMO 版)	西门子	3CD	随系统赠送

② 选择 PLC 做主站，优点：系统可靠，可做为批量生产产品时，产品出厂的测试系统。

选择 PLC 做主站				
序号	名称	制造商	数量	备注
1	S7-300PLC/CPU313-2DP	西门子	1 块	
2	MPI 编程电缆	西门子	1 根	
3	组态软件（光盘）： STEP 7 V5.1+SP1(DEMO 版)	西门子	1CD	随系统赠送

5. 安全指南

本手册包括应该遵守的注意事项，以保护产品和所连接的设备免受损坏。



警告 (Warning):
禁止带电插拔 RS232 电缆和 PROFIBUS 电缆。

第二章 OEM2 开发实验板

OEM2 开发实验板是调试系统中 PROFIBUS 的从站部件，是 OEM2 接口板的评估板。型号：PB-OEM2-SAMPLE。

开发实验板为用户提供了一块开发样板。实验板的全部硬件原理图和 C 源码程序可在本手册中找到。

用户使用 OEM2 开发实验板可以实现：

- ① 与 PROFIBUS 主站连接，了解如何实现 PROFIBUS 组态、配置；
- ② 依照实验板硬件原理图和 C 源码程序，可以更容易理解 PB-OEM2-SE 接口板的应用，设计用户产品的软硬件。
- ③ 实现调试实验系统的 3 种调试模式；

1. PB-OEM2-SAMPLE 布局图

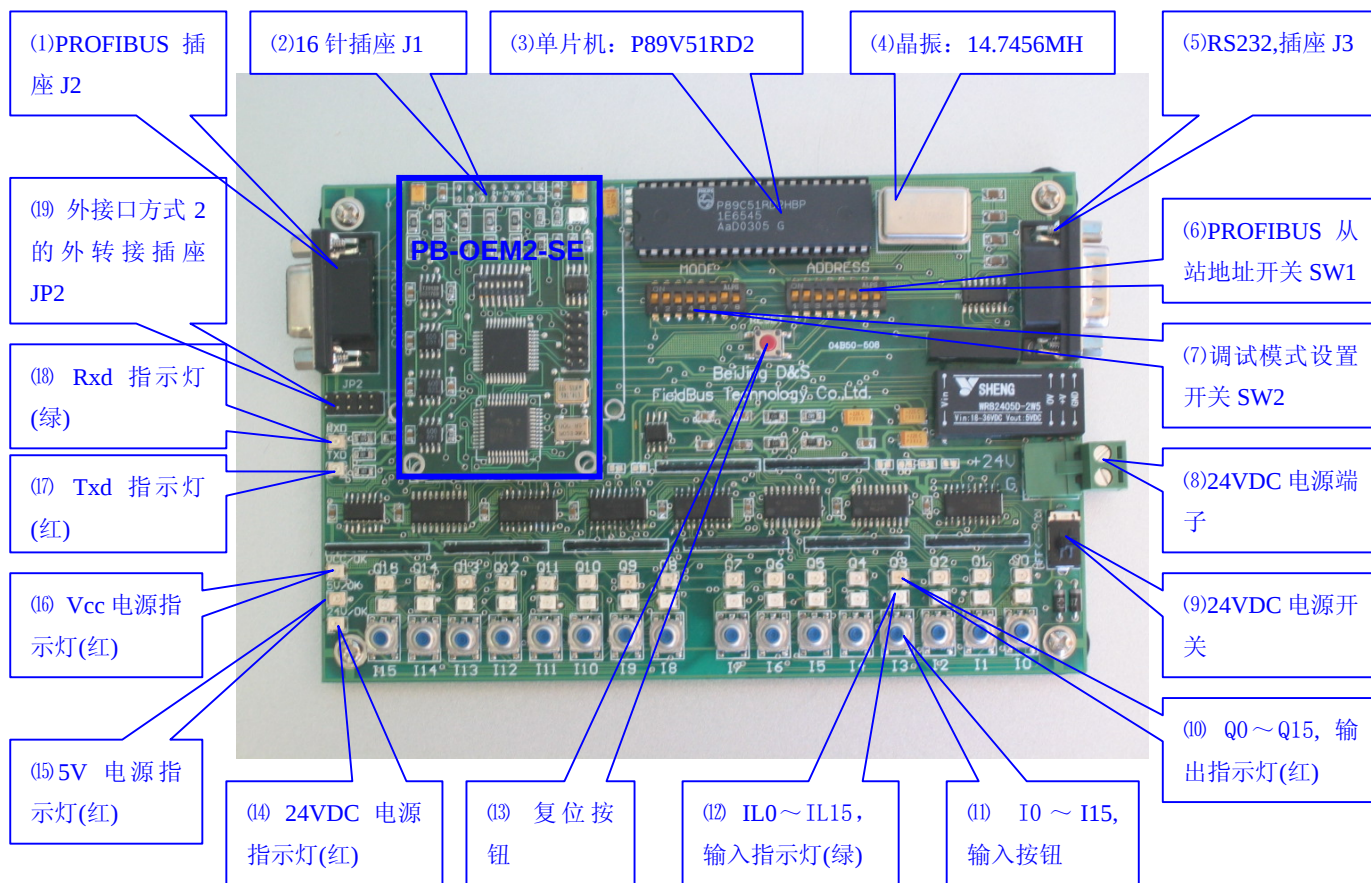


图 2-1 PB-OEM2-SAMPLE 布局图

- (1) **PROFIBUS 插座 J2**：用于连接 PROFIBUS 电缆插座。插座为 9 孔型（female）D 型插座。
- (2) **16 针插座 J1**：实验板上连接 PB-OEM2-SE 的 16 针（针）插座；
- (3) **晶振 14.7456MH**：有源晶振插座可任意更换晶振，如果与接口板通信波特率选择为 9.6K, 19.2K, 38.4K, 57.6K, 115.2K，推荐使用 14.7456MH。
- (4) **单片机**：P89V51RD2

(5) **RS232 插座 J3**: 9 针 D 型插座

(6) **PROFIBUS 地址开关 SW1**: PROFIBUS 从站地址设置开关, 这个地址必须和主站系统硬件配置中该从站的地址设置一致。



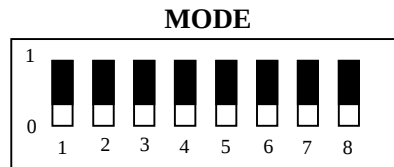
SW1 设置 PROFIBUS 从站地址例:

从站地址_(max)=126_(Dec)=7E_(Hex)=01111110_(Bin);

从站地址=50_(Dec)=32_(Hex)=00110010_(Bin);

从站地址=19_(Dec)=13_(Hex)=00010011_(Bin);

(7) **调试模式设置开关 SW2**



① **调试模式 1: 实验板调试模式**, **MODE**=11000000_(Bin);

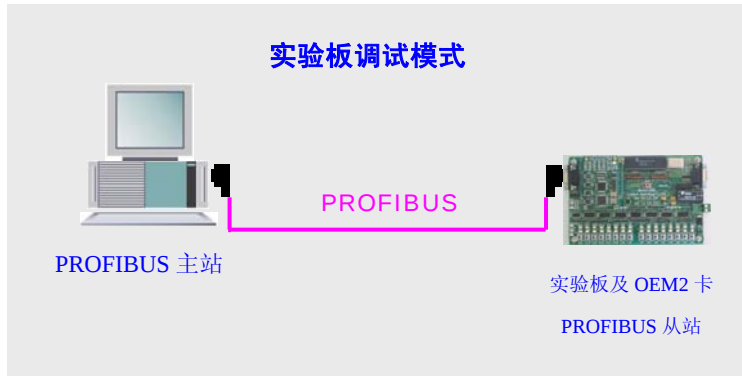


图 2-2 调试模式 1—实验板调试模式

② **调试模式 2: OEM2 接口板调试模式**, **MODE**=11000010_(Bin);

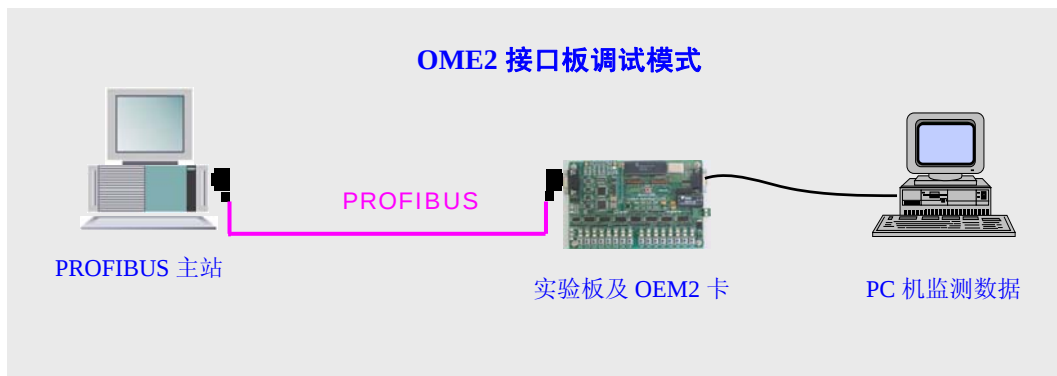


图 2-3 调试模式 2—OEM2 接口板调试模式

③ 实验板程序下载，MODE=00000111_(Bin);

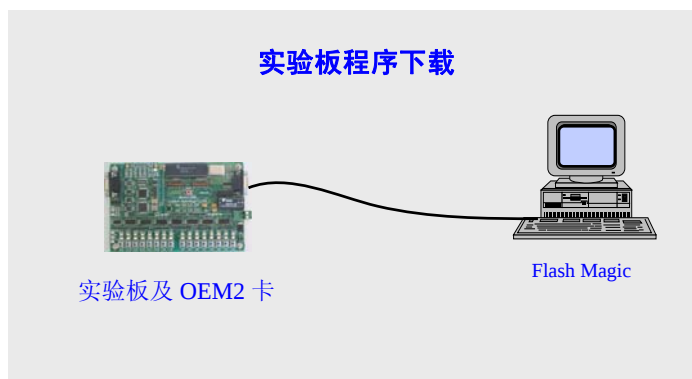


图 2-4 实验板程序下载

④ 调试模式 3：PC 机调试模式，MODE=00100000_(Bin);

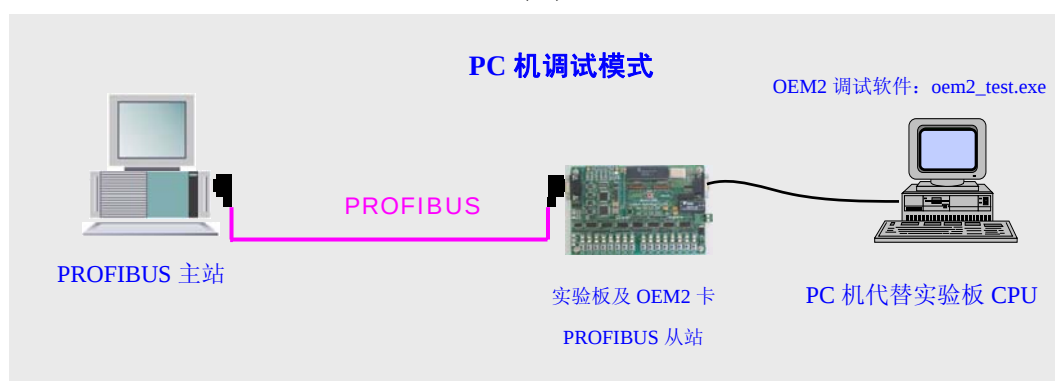


图 2-5 调试模式 3—PC 机调试模式

(8) 24VDC 电源端子：实验板直流 24V 供电（18~36V），注意实验板上标定的正负极。

(9) 24VDC 电源开关

(10) Q0~Q15，输出指示灯(红)

(11) I0~I15，输入按钮

(12) IL0~IL15，输入指示灯(绿)

(13) 复位按钮

(14) 24VDC 电源指示灯(红)

(15) 5V 电源指示灯(红)

(16) Vcc 电源指示灯(红)

(17) Txd 指示灯(红)：实验板 CPU 与 OEM2 卡串行通信，实验板 CPU 发送指示灯；

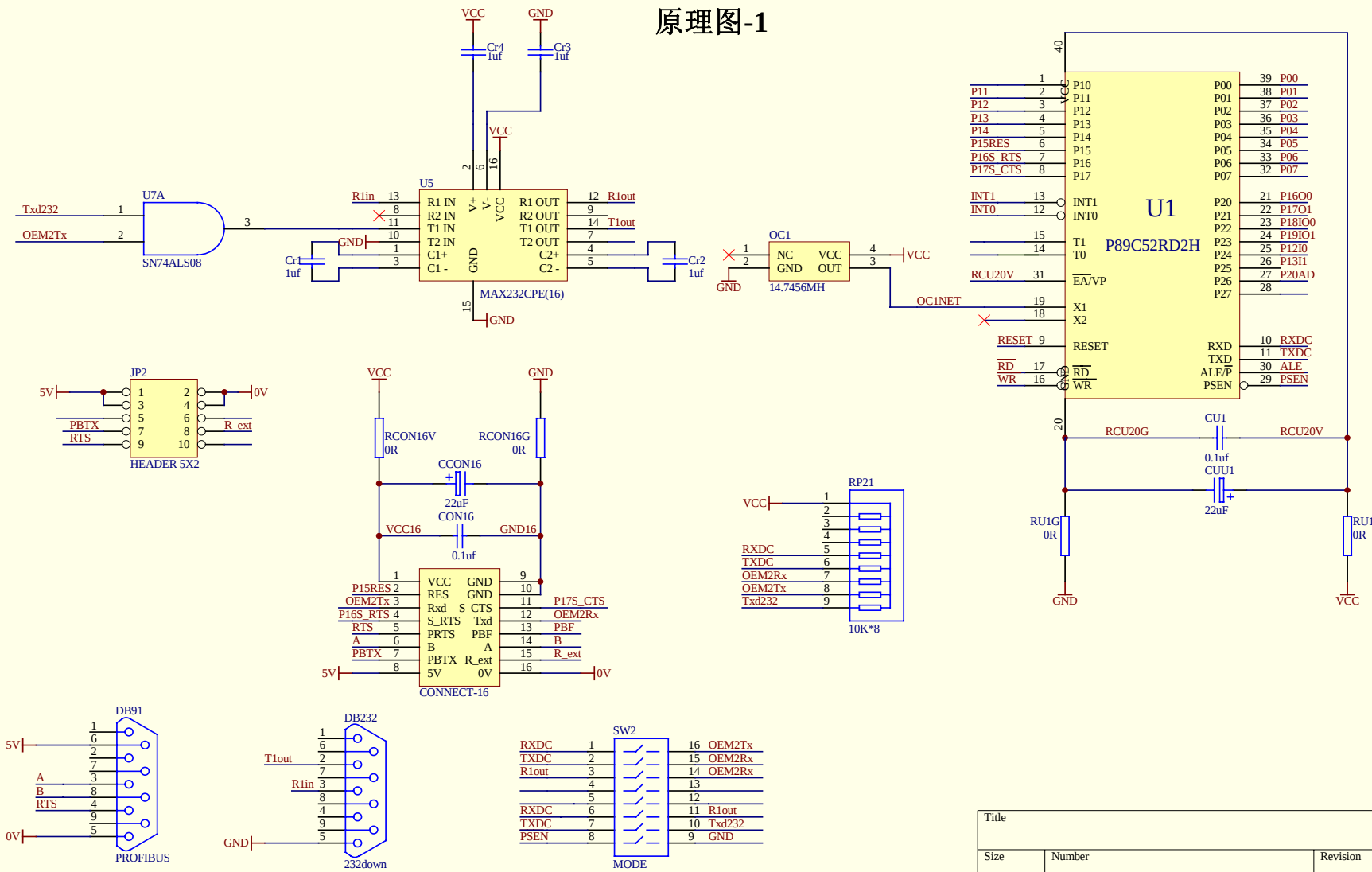
(18) Rxd 指示灯(绿)：实验板 CPU 与 OEM2 卡串行通信，实验板 CPU 接收指示灯；

(19) 外接口方式 2 的外转接插座 JP2

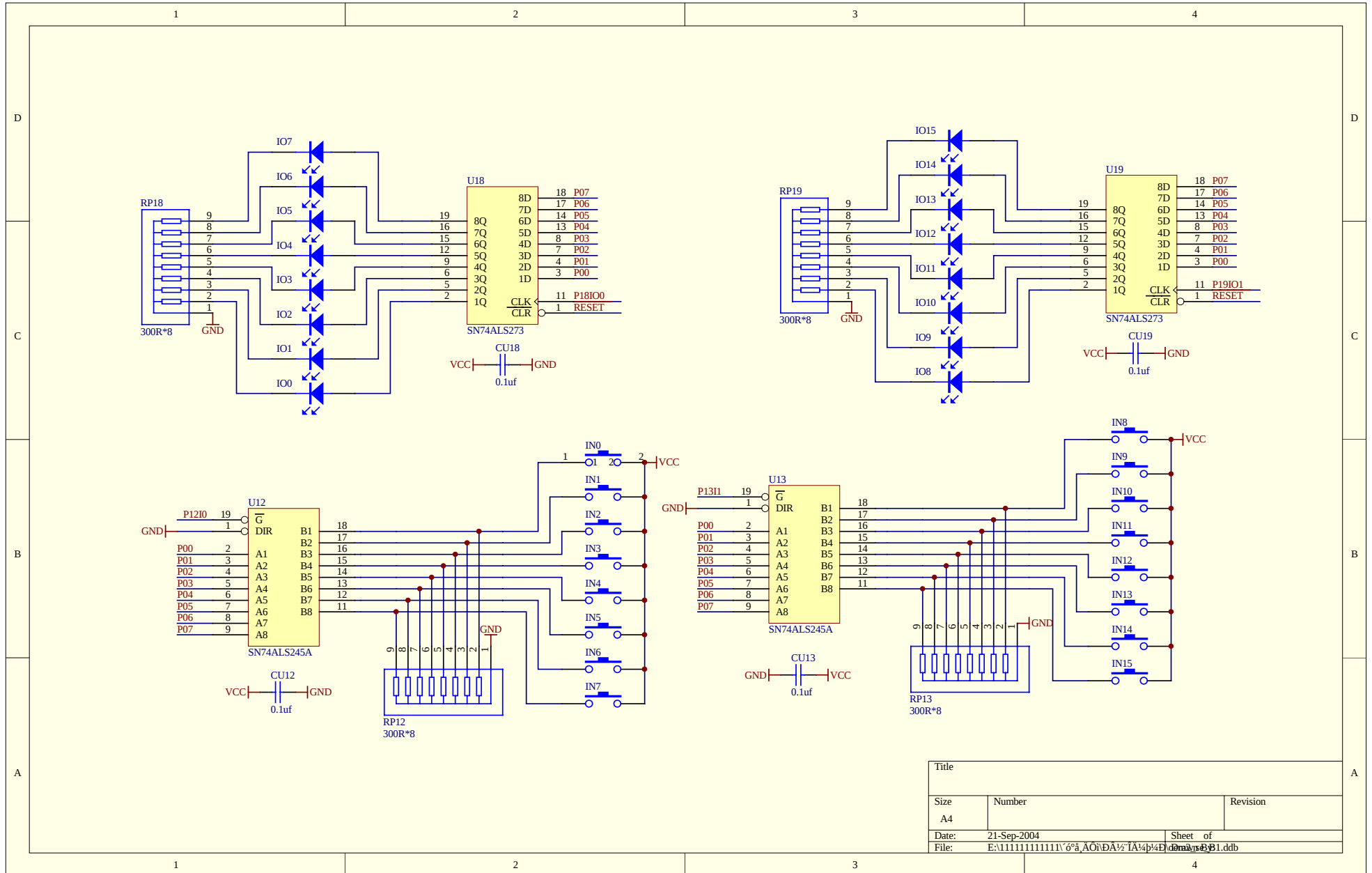
2. OEM2 开发实验板原理图

在实验系统中，开发实验板与 OEM2 模块组成 PROFIBUS 从站，开发实验板是用户样板，用户在详细阅读《PB-OEM2-SE 产品手册》、了解 OEM2 模块应用技术后，可以参考实验板原理图开发用户从站。

原理图-1



Title		
Size	Number	Revision
A4		
Date:	21-Sep-2004	Sheet of
File:	E:\1111111111\6*8_AO\DA\TA\p4\B1.ddb	



Title		
Size	Number	Revision
A4		
Date:	21-Sep-2004	Sheet of
File:	E:\111111111111\63A\0\DA\2\TA\4\p4\0\msb\B1.dcb	

(1) 原理图-1

- ① **U7A:** SN74ALS08, 四与门
- ② **U5:** MAX232, RS232 驱动芯片
- ③ **U1:** 实验板 CPU, PHILIP/P89V52RD2
- ④ **OC1:** 14.7456MH 晶振
- ⑤ **JP2:** 外接口方式 2 的外转接插座 JP2
- ⑥ **CONNECT-16:** 16 针插座 J1, 连接 PB-OEM2-SE 模块
- ⑦ **RP21:** 信号上拉排电组
- ⑧ **DB91:** PROFIBUS 插座 J2
- ⑨ **DB232:** RS232 插座 J3
- ⑩ **SW2:** 调试模式设置开关 SW2

(2) 原理图-2

- ① **U2:** 复位电路
- ② **U4:** DC/DC, 24VDC 转 5V/0V
- ③ **U9:** DC/DC, 24VDC 转 VCC/GND
- ④ **24V/OK、5V/OK、VCC/OK:** 24V、5V、VCC 电源指示灯
- ⑤ **RXD、TXD:** 实验板 CPU 与 OEM2 卡串行通信, 实验板 CPU 发送、接收指示灯;
- ⑥ **24VP/24K/D1/D2:** 24V 电源插座、开关、极性保护二级管;

(3) 原理图-3

- ① **SW1/U20/RP20:** PROFIBUS 地址开关 SW1 及相关电路。
- ② **U16/Q0~Q7/RP16:** PROFIBUS 输出锁存及 Q0~Q7 指示灯。
- ③ **U17/Q8~Q15/RP17:** PROFIBUS 输出锁存及 Q8~Q15 指示灯。

(4) 原理图-4

- ① **U18/IO0~IO7/RP18:** I0~I7 输入键对应的 IO0~IO7 指示灯 (绿)。
- ② **U19/IO8~IO15/RP19:** I8~I15 输入键对应的 IO8~IO15 指示灯 (绿)。
- ③ **U12/IN0~IN7/RP12:** I0~I7 输入键及输入缓冲电路。注意: 输入键不带自锁。
- ④ **U13/IN8~IN15/RP13:** I8~I15 输入键及输入缓冲电路。注意: 输入键不带自锁。

第三章 实验板程序、GSD 文件

1. 实验板程序

(1) 实验板 CPU

采用 PHILIPS 公司 P89V51RD2，主要技术指标：

- 80C51 核心处理单元；
- 5V 的工作电压，操作频率为 0~40MHZ；
- 64KB 的片内 Flash 程序存储器，具有 ISP（在系统编程）和 IAP（在应用中编程）功能；
- 通过软件或 ISP 选择支持 12 时钟（默认）或 6 时钟模式；
- SPI（串行外围接口）和增强型 UART；
- PCA（可编程计数器阵列），具有 PWM 和捕获/比较模式；
- 4 个 8 位 I/O 口，含有 3 个高电流 PI 口（每个 I/O 口的电流为 16mA）；
- 3 个 16 位定时器/计数器；
- 可编程看门狗定时器（WDT）；
- 8 个中断源，4 个中断优先级；
- 2 个 DPTR 寄存器；
- 低 EMI 方式（ALE 禁能）；
- 兼容 TTL 和 CMOS 逻辑电平；
- 掉电检测；
- 低功耗模式；
 - ◆ 掉电模式，外部中断唤醒；
 - ◆ 空闲模式；
- PDIP40，PLC44 和 TQFP44 的封装；

(2) 实验板程序

① 出厂时实验板 CPU 程序是“例 2：监测 SE 接口状态的工作模式”

② 产品 CD 光盘中有实验板 CPU 程序：

例 1：简单工作模式，包括 C 源代码 user.c、目标码 b1.hex

例 2：监测 SE 接口状态的工作模式，包括 C 源代码 user.c、目标码 b2.hex

例 3：带用户参数功能工作模式，包括 C 源代码 user.c、目标码 b3.hex

③ 如果用户希望该变出厂时实验板 CPU 程序“例 2：监测 SE 接口状态的工作模式”，可以按照实验板工作模式“调试模式 2”中“实验板程序下载”方法，选择下载 b1.hex、b2.hex、b3.hex。

④ 关于例 1、例 2、例 3 功能详解及实验方法可以参考第四章、第五章内容。

2. 关于 GSD 文件（Electronic Data Sheet）

① 每一个 PROFIBUS 从站都要有一个“设备描述文件”称为 GSD 文件，用来描述该 PROFIBUS-DP 设备的特性。

② GSD 文件包含了设备所有定义参数，如下：

- 支持的波特率；
- 支持的信息长度；

- 输入/输出数据数量;
- 诊断数据的含义;
- 可选模块种类等。

③ GSD 文件是文本类文件, 可用“记事本”编辑。

④ 无论使用什么样的系统配置软件, 都要根据 GSD 文件来对 DP 设备配置。

⑤ 国际 PROFIBUS 组织 PI 提供了 GSD 文件编辑软件: gsdedit.exe。该软件依照 profibus 技术标准格式规定, 对用户编辑的 GSD 文件进行格式检查。该软件的“帮助”功能强大, 也是一种快速学习 GSD 文件技术的途径。见图 3-1、图 3-2。

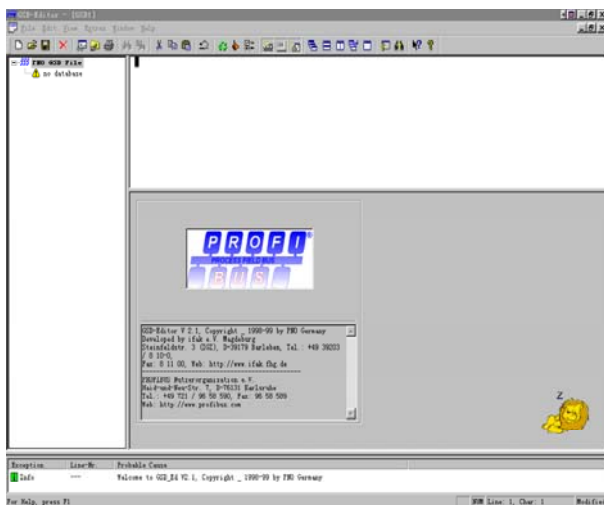


图 3-1 GSEEDIT 打开一个空文件

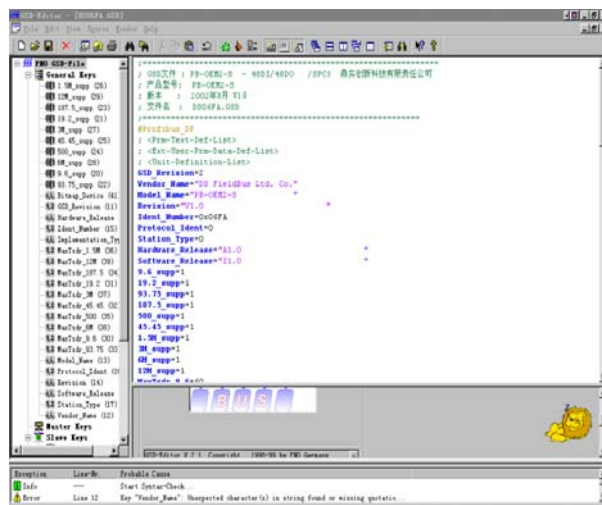


图 3-2 GSEEDIT 打开 DS_06FA.GSD 文件

3. PB-OEM2-SE 的 ID 号及 GSD 文件

PB-OEM2-SE 产品已经在国际 PROFIBUS 组织 PI 备案, 已取得 PI 授予 ID 号: 06FA 和 GSD 文件名: DS_06FA.GSD。

4. 用户产品的 ID 号、GSD 文件及产品认证

(1) SE 初始化报文中的 ID 号必须和 GSD 文件中的 ID 号一致才能连通;

(2) 由于本产品以 OEM 方式销售, 用户对应应用本产品开发的 PROFIBUS 设备有自主知识产权和品牌;

因此, 当用户产品正式销售提供给你的用户时, 本产品的 ID 号和 GSD 文件名不宜作为用户产品的 ID 号和 GSD 文件名;

(3) 如果用户需要产品的测试认证, 可以委托“中国 PROFIBUS 组织 CPO”向国际 PROFIBUS 组织 PI 办理申请产品认证手续, 那时, 用户可以得到自己的产品 ID 号和 GSD 文件名。用户还应与“中国 PROFIBUS 产品测试实验室 CPPTL”联系进行产品测试。产品测试合格后 CPPTL 将出据“测试报告”; 国际 PROFIBUS 组织 PI 根据产品的“测试报告”决定给您的产品正式认证证书。

(4) 用户也可以暂时自定义一个 ID 号, 在产品开发时期使用。在一条 PROFIBUS 总线上, 不同类型或相同类型具有同一 ID 号的从站并不影响系统连通。

(5) 用户产品的 GSD 文件

用户产品的 GSD 文件可以在本产品 GSD 文件基础上，在用户公司名、产品型号、系列号等处置换成用户产品信息，即可成为用户的 GSD 文件。产品 CD 光盘备有例 1、例 2、例 3 的 GSD 文件：OEM2_B1.GSD、OEM2_B2.GSD、OEM2_B3.GSD 及实验板图标文件 SE_B.BMP,关于例 1、例 2、例 3GSD 文件细节，详见第四章有关内容。

第四章 实验板程序举例

例 1. 简单工作模式

(1) 技术参数

PROFIBUS 输入/输出：48 字节输入/48 字节输出；

用户参数：无；

通信模式：不检测 SE 接口状态条件；

串口波特率：115.2K

(2) SE 初始化报文

I/O 配置数据长度 CFG_LEN=6；

I/O 配置数据为：0x1f, 0x2f, 0x1f, 0x2f, 0x1f, 0x2f

用户参数长度 User_Prm_Data_Len=0；

接收数据长 SD_in_len≥48+1+1=50

发送数据长 SD_out_len≥48+1+1=50

byte0	站号=实验板上 SW1
byte1	ID 号高位=0x06
byte2	ID 号低位=0xfa
byte3	CFG_LEN=0x06
byte 4	I/O 配置数据 1=0x1f
byte 5	I/O 配置数据 2=0x2f
byte 6	I/O 配置数据 3=0x1f
byte 7	
byte 23	
byte 24	接收数据长 SD_in_len=50
byte 25	发送数据长 SD_out_len=50
byte 26	用户参数长度 User_Prm_Data_Len=0
	***不用
byte 47	
byte 48	校验和 Σ (0) ~ (47)

实验板发送初始化报文

byte0	SE 接口波特率号 BR=1~7
byte 1	初始化报文错误号 N=0
byte 2	初始化成功标志=0xaa
byte 3	初始化成功标志=0xaa
byte 4	初始化成功标志=0xaa
byte 5	初始化成功标志=0xaa
byte 6	
byte 7	
byte 23	
byte 24	
byte 25	
byte 26	
byte 47	
byte 48	校验和 Σ (0) ~ (47)

SE 接口板回答的初始化成功报文

byte0	SE 接口波特率号 BR=1~7
byte 1	初始化报文错误号 N=01~06
byte 2	初始化失败标志=0x55
byte 3	初始化失败标志=0x55
byte 4	初始化失败标志=0x55
byte 5	初始化失败标志=0x55
byte 6	
byte 7	
byte 23	
byte 24	
byte 25	
byte 26	
byte 47	
byte 48	校验和 Σ (0) ~ (47)

接口板回答的初始化失败报文

(3) 数据交换状态下实验板与 SE 接口的数据报文

实验板不使用用户参数功能（即：初始化报文中 User_Prm_Data_Len=0），永远发请求 PROFIBUS 数据命令 req_com=0；接口板一定回答“输出数据报文 A”。见图 4-1-1，图 4-1-2

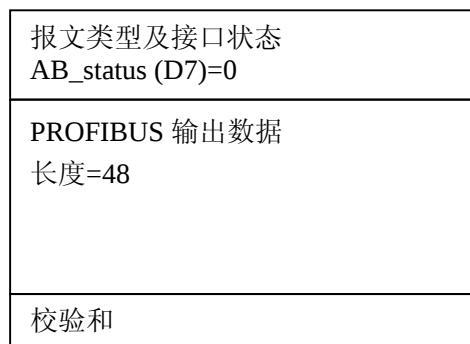
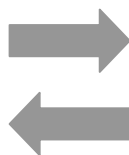
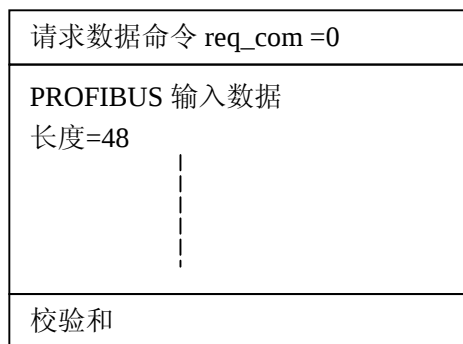


图 4-1-1 实验板发—输入数据报文 req_com=0

图 4-1-2 SE 接口板答—输出数据报文格式 A

(4) 实验板初始化 SE 接口板的过程

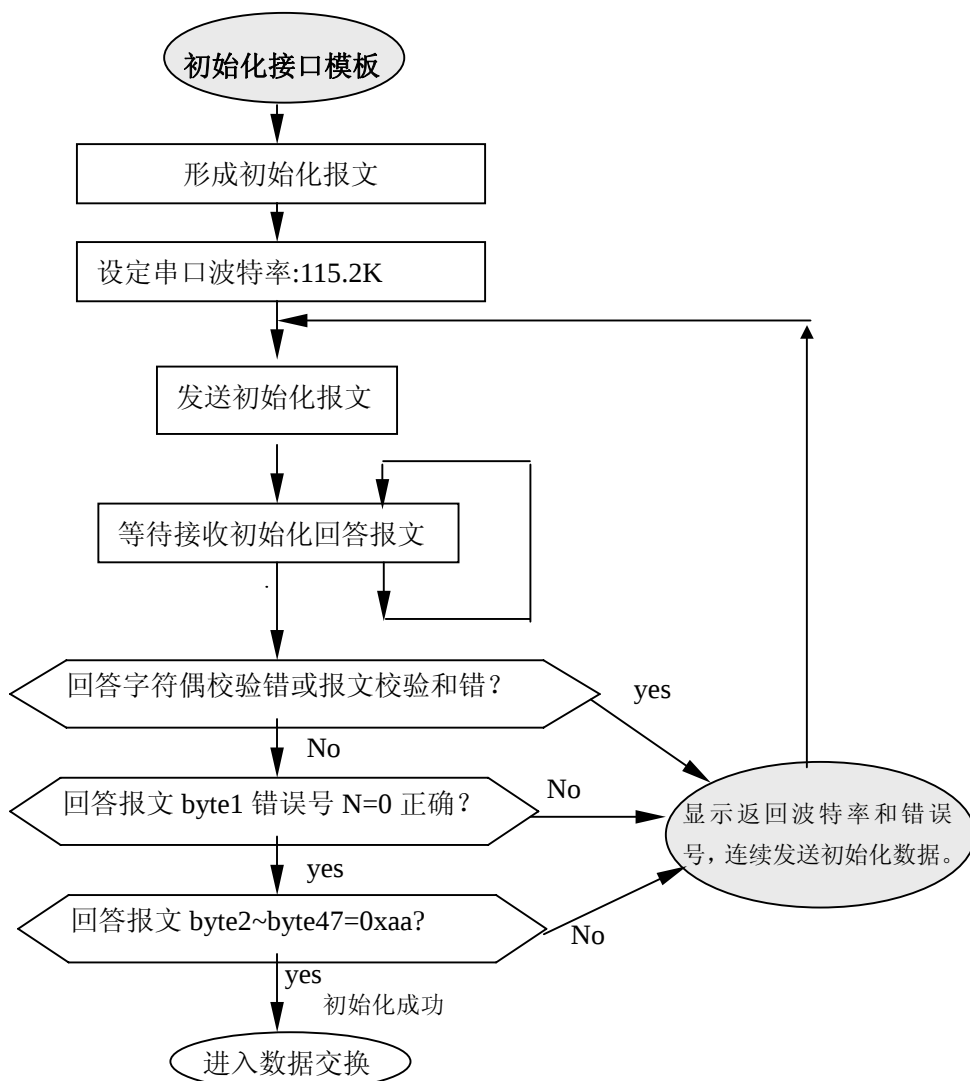


图 4-1-3 实验板初始化接口模板的过程

(5) 数据交换状态下实验板简单工作模式

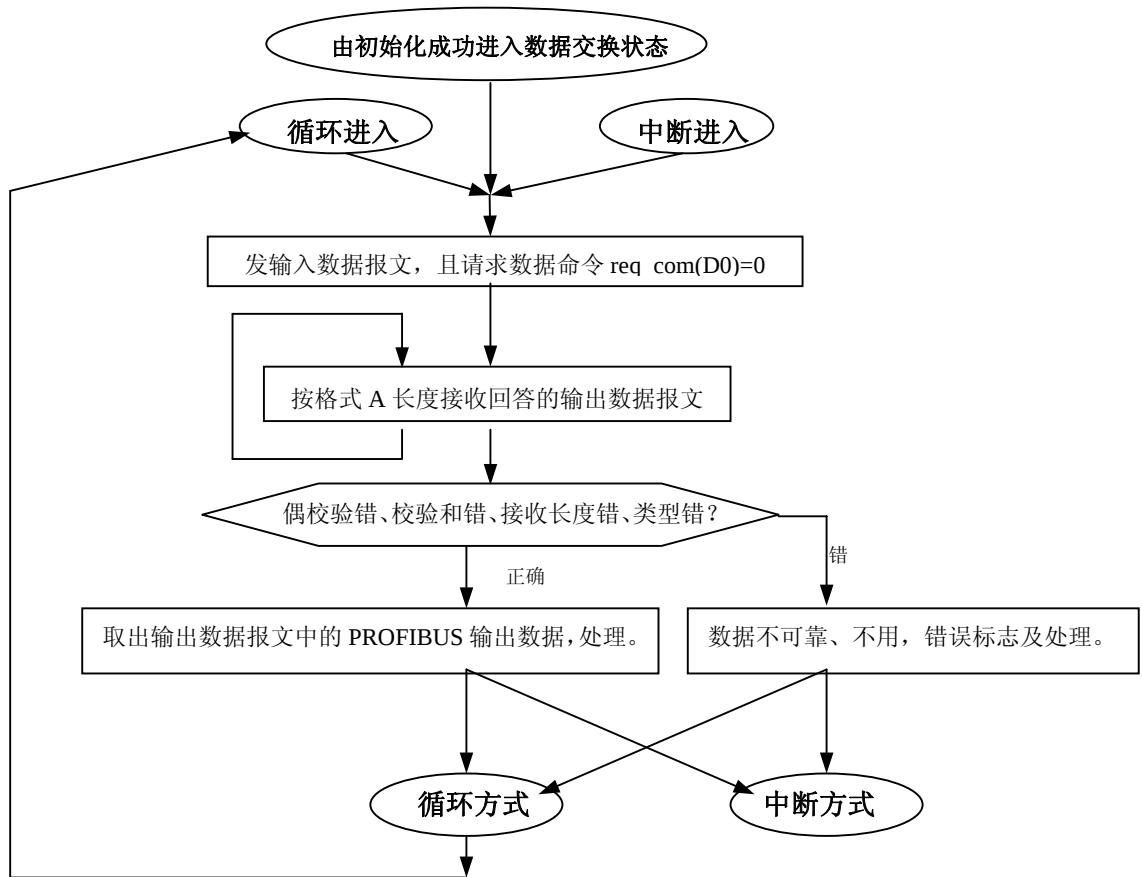


图 4-1-4 简单工作模式下实验板程序流程图

(6) 实验板程序清单

```

/*+-----+
|文件名称: 实验板板串口通信程序—简单工作模式|
|版 本: V1.1|
|制作单位: 北京鼎实创新科技公司|
|版权许可:|
|网址: www.c-profibus.com.cn|
|email: tangjy@c-profibus.com.cn|
|日期: 2004/9|
+-----+*/

#include <c:\51\c51\INC\reg52.h>
#define UBYTE unsigned char
#define UWORD unsigned int
/*=====*/
/* 以下是输入键、指示灯等设备变量定义，用户可以越过阅读 */
/*-----*/

sbit out0_7=P2^0; /* 定义 out0_7=P2.0=P16O0,输出指示灯 Q0-Q7 的输出控制,见实验板原理图 U16 -----*/
sbit out8_15=P2^1; /* 定义 out8_15=P2.1=P17O1,输出指示灯 Q8-Q15 的输出控制,见实验板原理图 U17 -----*/
sbit inled0_7=P2^2; /* 定义 inled0_7=P2.2=P18IO0,输入指示灯 I0-I7 的输出控制,见实验板原理图 U18 -----*/
sbit inled8_15=P2^3; /* 定义 inled8_15=P2.3=P19IO1,输入指示灯 I8-I15 的输出控制,见实验板原理图 U19 -----*/
sbit in0_7=P2^4; /* 定义 in0_7=P2.4=P12IO,输入键 I0-I7 的读控制,见实验板原理图 U12 -----*/
sbit in8_15=P2^5; /* 定义 in8_15=P2.5=P2.5=P13I1,输入键 I8-I15 的读控制,见实验板原理图 U13 -----*/
sbit inadd=P2^6; /* 定义 inadd=P2.6=P2.6=P20AD,PROFIBUS 地址输入 AD0-AD7 的读控制, -----*/
/* 见实验板原理图 U20 -----*/

sbit REQ_IT=P1^7; /* 定义 REQ_IT=P1^7=P1.7=P17S_CTS,输入,连接 OEM2-SE-REQ_IT, -----*/
/* 见实验板原理图 CONNECT-16 -----*/

sbit S_RTS=P1^6; /* 定义 S_RTS=P1^6=P1.6=P17S_RTS,输入,连接 OEM2-SE-S_RTS, -----*/
/* 见实验板原理图 CONNECT-16 -----*/

```

```

sbit P15RES=P1^5;      /* 定义 P15RES=P1^5=RES,输入,连接 OEM2-SE-RES,          -----*/
                        /* 见实验板原理图 CONNECT-16                      -----*/

/***** 定义: di0_7:字节型,是输入键 I0-I7 当前值; *****/
/***** 定义: di8_15:字节型,是输入键 I8-I15 当前值; *****/
bdata UBYTE di0_7,di8_15; /* di0_7:字节型,输入键 I0-I7;di8_15:字节型,输入键 I8-I15 -----*/
sbit di0=di0_7^0;        /* 定义 di0=di0_7^0=输入键 I0          -----*/
sbit di1=di0_7^1;        /* 定义 di1=di0_7^1=输入键 I1          -----*/
sbit di2=di0_7^2;        /* 定义 di2=di0_7^2=输入键 I2          -----*/
sbit di3=di0_7^3;        /* 定义 di3=di0_7^3=输入键 I3          -----*/
sbit di4=di0_7^4;        /* 定义 di4=di0_7^4=输入键 I4          -----*/
sbit di5=di0_7^5;        /* 定义 di5=di0_7^5=输入键 I5          -----*/
sbit di6=di0_7^6;        /* 定义 di6=di0_7^6=输入键 I6          -----*/
sbit di7=di0_7^7;        /* 定义 di7=di0_7^7=输入键 I7          -----*/
sbit di8=di8_15^0;       /* 定义 di8=di8_15^0=输入键 I8          -----*/
sbit di9=di8_15^1;       /* 定义 di9=di8_15^1=输入键 I9          -----*/
sbit di10=di8_15^2;      /* 定义 di10=di8_15^2=输入键 I10        -----*/
sbit di11=di8_15^3;      /* 定义 di11=di8_15^3=输入键 I11        -----*/
sbit di12=di8_15^4;      /* 定义 di12=di8_15^4=输入键 I12        -----*/
sbit di13=di8_15^5;      /* 定义 di13=di8_15^5=输入键 I13        -----*/
sbit di14=di8_15^6;      /* 定义 di14=di8_15^6=输入键 I14        -----*/
sbit di15=di8_15^7;      /* 定义 di15=di8_15^7=输入键 I15        -----*/

/***** 定义: key0_7:字节型,是输入键 I0-I7 扫描读入值; *****/
/***** 定义: key8_15:字节型,是输入键 I8-I15 扫描读入值; *****/
bdata UBYTE key0_7,key8_15;
sbit key0=key0_7^0;       /* 定义 key0=key0_7^0=输入键 I0          -----*/
sbit key1=key0_7^1;       /* 定义 key1=key0_7^1=输入键 I1          -----*/
sbit key2=key0_7^2;       /* 定义 key2=key0_7^2=输入键 I2          -----*/
sbit key3=key0_7^3;       /* 定义 key3=key0_7^3=输入键 I3          -----*/
sbit key4=key0_7^4;       /* 定义 key4=key0_7^4=输入键 I4          -----*/
sbit key5=key0_7^5;       /* 定义 key5=key0_7^5=输入键 I5          -----*/
sbit key6=key0_7^6;       /* 定义 key6=key0_7^6=输入键 I6          -----*/
sbit key7=key0_7^7;       /* 定义 key7=key0_7^7=输入键 I7          -----*/
sbit key8=key8_15^0;      /* 定义 key8=key8_15^0=输入键 I8          -----*/
sbit key9=key8_15^1;      /* 定义 key9=key8_15^1=输入键 I9          -----*/
sbit key10=key8_15^2;     /* 定义 key10=key8_15^2=输入键 I10        -----*/
sbit key11=key8_15^3;     /* 定义 key11=key8_15^3=输入键 I11        -----*/
sbit key12=key8_15^4;     /* 定义 key12=key8_15^4=输入键 I12        -----*/
sbit key13=key8_15^5;     /* 定义 key13=key8_15^5=输入键 I13        -----*/
sbit key14=key8_15^6;     /* 定义 key14=key8_15^6=输入键 I14        -----*/
sbit key15=key8_15^7;     /* 定义 key15=key8_15^7=输入键 I15        -----*/

/***** 定义: P00-P07 对应 P0 口的 P0.0-P0.7 *****/
sbit P00=P0^0;
sbit P01=P0^1;
sbit P02=P0^2;
sbit P03=P0^3;
sbit P04=P0^4;
sbit P05=P0^5;
sbit P06=P0^6;
sbit P07=P0^7;

/***** PP00-PP07 是上面 P00-P07 的中间变量 *****/
bdata UBYTE PP0;
sbit PP00=PP0^0;
sbit PP01=PP0^1;
sbit PP02=PP0^2;
sbit PP03=PP0^3;
sbit PP04=PP0^4;
sbit PP05=PP0^5;
sbit PP06=PP0^6;
sbit PP07=PP0^7;

```

```

/**** 定义：AB_status:字节型，报文类型及接口状态，见《PB-OEM2-SE 产品手册》第四章、三、数据交换 *****/
bdata UBYTE AB_status;
sbit AB_sD0=AB_status^0;      /**** AB_sD0=AB_status^0:用户参数更新标记 *****/
sbit AB_sD7=AB_status^7;      /**** AB_sD7=AB_status^7:报文格式 *****/

/**** 定义：req_com:字节型，请求数据命令，见《PB-OEM2-SE 产品手册》第四章、三、数据交换 *****/
bdata UBYTE req_com;
sbit req_comD0=req_com^0;      /**** req_comD0=req_com^0:请求数据/参数命令 *****/
sbit req_comD1=req_com^1;      /**** req_comD1=req_com^1:清参数命令 *****/

/*-----*/
/*          输入键、指示灯等设备变量定义完毕          */
/*=====*/

/*=====*/
/*          以下是有关输入键、指示灯控制子程序，用户了解功能即可          */
/*-----*/

/***** 将 V 值送指示灯 Q0-Q7 输出 *****/
void steout0_7(UBYTE v)
{
    out0_7=0;
    P0=v;
    P0=v;
    P0=v;
    P0=v;
    out0_7=1;
    out0_7=0;
}

/***** 将 V 值送指示灯 Q8-Q15 输出 *****/
void steout8_15(UBYTE v)
{
    out8_15=0;
    P0=v;
    P0=v;
    P0=v;
    P0=v;
    out8_15=1;
    out8_15=0;
}

/***** 将 V 值送输入键指示灯 ILED0-LIED7 输出 *****/
void steinled0_7(UBYTE v)
{
    inled0_7=0;
    P0=v;
    P0=v;
    P0=v;
    P0=v;
    inled0_7=1;
    inled0_7=0;
}

/***** 将 V 值送输入键指示灯 ILED8-LIED15 输出 *****/
void steinled8_15(UBYTE v)
{
    inled8_15=0;
    P0=v;
    P0=v;
    P0=v;
    P0=v;
    inled8_15=1;
    inled8_15=0;
}

```

```

/***** 读 PROFIBUS 地址开关值 *****/

```

```

UBYTE rd_address()
{
    UBYTE data x;
    P0=0xff;
    inadd=0;
    x=P0;
    x=P0;
    x=P0;
    x=P0;
    inadd=1;
    return(x);
}

```

```

/***** 扫描一次输入键 I0-I15，更新一次输入键当前值 *****/
/***** 对输入键 I0-I15 有 0-1 者，当前值取反 *****/

```

```

void scanin()
{
    P0=0xff;
    in0_7=0;
    PP0=P0;
    PP0=P0;
    PP0=P0;
    PP0=P0;
    if (key0_7!=PP0)
    {
        if ((PP0==1)&&(key0==0)) { di0=~di0;};
        if ((PP1==1)&&(key1==0)) { di1=~di1;};
        if ((PP2==1)&&(key2==0)) { di2=~di2;};
        if ((PP3==1)&&(key3==0)) { di3=~di3;};
        if ((PP4==1)&&(key4==0)) { di4=~di4;};
        if ((PP5==1)&&(key5==0)) { di5=~di5;};
        if ((PP6==1)&&(key6==0)) { di6=~di6;};
        if ((PP7==1)&&(key7==0)) { di7=~di7;};
        key0_7=PP0;
    };
    in0_7=1;
    inled0_7=0;
    P0=di0_7;
    P0=di0_7;
    P0=di0_7;
    P0=di0_7;
    inled0_7=1;
    inled0_7=0;

    P0=0xff;
    in8_15=0;
    PP0=P0;
    PP0=P0;
    PP0=P0;
    PP0=P0;
    if (key8_15!=PP0)
    {
        if ((PP0==1)&&(key8==0)) { di8=~di8;};
        if ((PP1==1)&&(key9==0)) { di9=~di9;};
        if ((PP2==1)&&(key10==0)) { di10=~di10;};
        if ((PP3==1)&&(key11==0)) { di11=~di11;};
        if ((PP4==1)&&(key12==0)) { di12=~di12;};
        if ((PP5==1)&&(key13==0)) { di13=~di13;};
        if ((PP6==1)&&(key14==0)) { di14=~di14;};
        if ((PP7==1)&&(key15==0)) { di15=~di15;};
        key8_15=PP0;
    };
    in8_15=1;
}

```

```

inled8_15=0;
P0=di8_15;
P0=di8_15;
P0=di8_15;
P0=di8_15;
inled8_15=1;
inled8_15=0;
}

/*=====*/
/*                               主程序                               */
/*-----*/

void    main ()
{
    UBYTE data i,j,k,x,y,count,frist;
    UWORD data nw,nn;
    UBYTE xdata * data p;
    UBYTE data address;                /*-----从站地址-----*/
    UBYTE data re_er;                  /*-----字符偶校验错-----*/
    UBYTE data br_num;                /*-----串口波特率号-----*/
    UBYTE data error;                 /*-----校验和错-----*/
    UBYTE xdata rebox[113];           /*-----串口接收缓冲-----*/
    UBYTE xdata trbox[113];           /*-----串口发送缓冲-----*/
    UBYTE xdata pb_out[112];          /*-----PROFIBUS 输出数据区-----*/
    UBYTE xdata pb_in[112];           /*-----PROFIBUS 输入数据区-----*/
    UBYTE xdata user_prm[41];         /*-----PROFIBUS 用户参数区-----*/
    UBYTE data data_in_len;           /*-----数据交换输入（实验板发送）数据长度-----*/
    UBYTE data dil;                   /*-----不包括校验和的 data_in_len 长度-----*/
    UBYTE data data_out_len;          /*-----数据交换输出（实验板接收）数据长度-----*/
    UBYTE data dol;                   /*-----不包括校验和的 data_out_len 长度-----*/
    UBYTE data user_prm_len;          /*-----用户参数长度-----*/
    UBYTE data se_er;                 /*-----OEM2-SE 接收数据错误类型号-----*/

    P1=0xff;
    P0=0;
    P2=0xf0;
    di0_7=di8_15=0;

    steout0_7(0);                     /*-----输出指示灯 Q0-Q7 清零-----*/
    steout8_15(0);                     /*-----输出指示灯 Q8-Q15 清零-----*/
    steinled0_7(0);                   /*-----输入键指示灯 ILED0-ILED7 清零-----*/
    steinled8_15(0);                  /*-----输入键指示灯 ILED8-ILED15 清零-----*/

    /*=====*/
    /*  以下是实验板上电后，检测 P89V51RD2 单片机内部 RAM 的一段输出演示程序；          */
    /*  这段程序与 PROFIBUS 接口板 SE 通信无关，用户可跳过此段程序。                      */
    /*-----*/

    p=0;
    x=1;
    k=1;
    for (j=0;j<32;j++)
    {
        for (i=0;i<24;i++)
        {
            *(p+i)=0xaa;
        };
        y=0;
        for (i=0;i<24;i++)
        {
            count=*(p+i);
            if (count!=0xaa) { y=1; };
        };
        if (y==0)
        {
            switch (k)
            {

```

```

        case 1:{ steout0_7(x);};break;
        case 2:{ steout8_15(x);};break;
        case 3:{ steinled0_7(x);};break;
        case 4:{ steinled8_15(x);};break;
        default:break;
    };
    if (x<255) {x=x*2+1;};else{x=1;k++;};
}
else
{
    for (;;)
    {
        for (nw=0;nw<6000;nw++)
        {
            steout0_7(0);
            steout8_15(0);
            steinled0_7(0);
            steinled8_15(0);
        };
        for (nw=0;nw<6000;nw++)
        {
            steout0_7(0xff);
            steout8_15(0xff);
            steinled0_7(0xff);
            steinled8_15(0xff);
        };
    };
    p=p+24;
    for (nw=0;nw<8000;nw++)
    {};
};

/*-----*/
/* 实验板输出演示程序结束。 */
/*=====*/

steout0_7(0); /*-----输出指示灯 Q0-Q7 清零 -----*/
steout8_15(0); /*-----输出指示灯 Q8-Q15 清零 -----*/
steinled0_7(0); /*-----输入键指示灯 ILED0-ILED7 清零 -----*/
steinled8_15(0); /*-----输入键指示灯 ILED8-ILED15 清零 -----*/

/*=====*/
/* 清串口输入缓冲区 trbox[i], 清串口输出缓冲区 rebox[i] */
/*-----*/
for (i=0;i<113;i++)
{
    rebox[i]=trbox[i]=0;
};
for (i=0;i<112;i++)
{
    pb_in[i]=pb_out[i]=0;
};

address=rd_address(); /*-----读实验板上拨码开关，作为站号-----*/

/*=====*/
/* 以下是串口初始化程序 */
/*-----*/
br_num=5; /*-----实验板波特率选择 5, 115.2K-----*/
switch (br_num) /*-----根据波特率号设置 T2-----*/
{
    case 1:{ TH2=0x0ff; TL2=0x0a0;RCAP2H=0x0ff;RCAP2L=0x0a0;};break; /*---9.6K---*/
    case 2:{ TH2=0x0ff; TL2=0x0d0;RCAP2H=0x0ff;RCAP2L=0x0d0;};break; /*---19.2K---*/
    case 3:{ TH2=0x0ff; TL2=0x0e8;RCAP2H=0x0ff;RCAP2L=0x0e8;};break; /*---38.4K---*/
    case 4:{ TH2=0x0ff; TL2=0x0f0;RCAP2H=0x0ff;RCAP2L=0x0f0;};break; /*---57.6K---*/
}

```

```

        case 5:{ TH2=0x0ff; TL2=0x0f8;RCAP2H=0x0ff;RCAP2L=0x0f8;};break;          /*---115.2K-*/
        case 6:{ TH2=0x0ff; TL2=0x0fc;RCAP2H=0x0ff;RCAP2L=0x0fc;};break;        /*---230.4K-*/
        case 7:{ TH2=0x0ff; TL2=0x0fe;RCAP2H=0x0ff;RCAP2L=0x0fe;};break;        /*---460.8K-*/
        default:break;
    };

/*-----*/
/* 注意：本例基于 PHILIP P89V51RD2 单片机：                                  */
/* 波特率=14745600/16/（65536-T）                                          */
/* 各种波特率 9600: T=0xffa0, 19.2K: T=0xffd0, 38.4K: T=0xffe8,              */
/* 57.6K: T=0xffff0, 115.2K:T=0xffff8, 230.4k:T=0xfffc, 460.8k=0xfffe          */
/*-----*/
PCON=0;
T2CON=0x34;          /*---RCLK=1; TCLK=1; TR2=1; C/T2=0 -----*/

/*-----*/
/* 串口设置成模式 3 即：1 个起始位、8 个数据位、1 个偶校验位、1 个停止位；    */
/* 0xd0=11010000B; SM0SM1=11; SM2=0; REN=1;TB8=0; RB8=0; TI=0; RI=0          */
/*-----*/
SCON=0xd0;

/*-----*/
/*                                     串口初始化程序结束                               */
/*=====*/
/*                                     形成接口板的初始化报文                               */
/* rebox[]是串口发送数组，格式见《SE 手册》                                     */
/*-----*/
data_in_len=48+1+1;          /*--- 输入数据报文长度=48+1+1=50; -----*/
user_prm_len=0;              /*--- 用户参数长度=0 -----*/
data_out_len=48+1+1;         /*--- 输入数据报文长度=48+1+1=50; -----*/
trbox[0]=address;            /*--- BYTE0=从站地址 -----*/
trbox[1]=0x06;                /*--- BYTE1=ID 号高位 -----*/
trbox[2]=0x0fa;              /*--- BYTE2=ID 号低位 -----*/

/*=====*/
/*                                     以下形成 I/O 配置数据                               */
/* 本例：I/O 配置数据长度=6; I/O 配置数据：0x1f,0x2f,0x1f,0x2f,0x1f,0x2f      */
/* 共：48bytes input+48bytes output                                           */
/*-----*/
trbox[3]=6;                   /*---BYTE3=I/O 配置数据长度=6 -----*/
trbox[4]=0x1f;                /*---BYTE4=I/O 配置数据： 0x1f -----*/
trbox[5]=0x2f;                /*---BYTE5=I/O 配置数据： 0x2f -----*/
trbox[6]=0x1f;                /*---BYTE6=I/O 配置数据： 0x1f -----*/
trbox[7]=0x2f;                /*---BYTE7=I/O 配置数据： 0x2f -----*/
trbox[8]=0x1f;                /*---BYTE8=I/O 配置数据： 0x1f -----*/
trbox[9]=0x2f;                /*---BYTE9=I/O 配置数据： 0x2f -----*/

trbox[24]=data_in_len;        /*---BYTE24=data_in_len 输入数据报文长度 -----*/
trbox[25]=data_out_len;       /*---BYTE25=data_out_len 输出数据报文长度 -----*/
trbox[26]=user_prm_len;       /*---BYTE26=user_prm_len 用户参数长度 -----*/

x=0;
for (i=0;i<48;i++)
{
    x=x+trbox[i];              /*---求校验和 -----*/
};
trbox[48]=x;                  /*---置校验和 -----*/

/*=====*/
/* 发送接口板的初始化报文、接收接口板的回答，初始不成功继续发初始化报文      */
/*-----*/
y=0;                          /*--- 清初始化成功标志 -----*/
while (y==0)
{
    REN=0;
    for (i=0;i<49;i++)          /*--- 发送一次初始化报文共 49 字节 -----*/

```

```

{
    TI=0;
    ACC=trbox[i];
    TB8=P;          /*---- 形成偶校验位          -----*/
    SBUF=trbox[i];
    while (TI==0) {};
};
/*----- 发送完毕、等待接收回答 -----*/
RI=0;
REN=1;             /*---- 允许接收          -----*/
re_er=0;           /*---- re_er 偶校验错标志 -----*/
for (j=0;j<49;j++)
{
    /*---- 接受回答报文共 49 个字节 -----*/
    nw=0;
    while (RI==0) {};          /*---- 等待接收一个字符 -----*/
    ACC=SBUF;
    if (P!=RB8) { re_er=1;};    /*---- 偶校验错, 置偶校验错标志 -----*/
    rebox[j]=SBUF;             /*---- 接收一个字符送串口输入缓冲区 -----*/
    RI=0;
};

/*----- 接收回答报文结束 -----*/
/*----- 以下分析接口板 SE 的回答报文 -----*/
if (re_er==0)          /*---- 没有字符偶校验错? -----*/
{
    x=0;
    for (i=0;i<48;i++)    /*---- 求校验和 -----*/
    {
        x=x+rebox[i];
    };
    if (x==rebox[48])
    {
        /*---- 校验和正确? -----*/
        if (rebox[0]==br_num) /*---- 返回的波特率号是 5? -----*/
        {
            if (rebox[1]==0) /*---- 初始化报文错误号 N=0? -----*/
            {
                x=0;
                for (i=2;i<48;i++) /*---- byte2~byte47 都是 0xaa? -----*/
                {
                    if (rebox[i]!=0xaa)
                    {
                        x=1;
                    };
                };
                if (x==0)
                {
                    y=1;          /*---- byte2~byte47 都是 0xaa,初始化成功,y=1,准备退出 while (y==0) ----*/
                }
                else
                {x=0x11;};        /*---- byte2~byte47 不都是 0xaa! 错误标记=0x11; -----*/
            }
            else
            {x=rebox[1];};        /*---- 初始化报文错误号 N<>0, 错误标记=返回错误号 -----*/
        }
        else
        {x=0x77;};              /*---- 返回的波特率号不是 5, 错误标记=0x77; -----*/
    }
    else
    {x=0xff;};                  /*---- 校验和不正确正确, 错误标记=0xff; -----*/
}
else
{
    x=0xee;
};
/*---- 发生了字符偶校验错 错误标记=0xee; -----*/

```

```

if (y==0)
{
/*---- 初始化失败 ----*/
for (nw=0;nw<10000;nw++)
{
steout0_7(br_num); /*---- 显示返回波特率号 ----*/
steout8_15(x); /*---- 显示返回错误号 ----*/
};
};
}; /*---- 如果 y==0,返回 while (y==0) ----*/

/*=====*/
/*      实验板对 PROFIBUS 接口板初始化成功, 进数据交换状态的无限循环中      */
/*      trbox[]/rebox[]是串口发送/接收数组, 发送/接收格式见《SE 手册》      */
/*-----*/
dil=data_in_len-1; /*---- data_in_len=50, dil=49 ----*/
dol=data_out_len-1; /*---- data_out_len==50, dol==49 ----*/
for (i=0;i<data_in_len;i++)
{
trbox[i]=0; /*---- 清发送、接收缓冲区 ----*/
};
for (i=0;i<data_out_len;i++)
{
rebox[i]=0; /*---- 清发送、接收缓冲区 ----*/
};

/*===== 进数据交换状态的无限循环中 =====*/
for (;;)
{
trbox[0]=req_com=0; /*---- 没有用户参数, 请求数据命令永远是 req_com=0, 只请求 A 型报文 ----*/
trbox[1]=di0_7; /*---- di0_7 是输入键输入 I0~I7 当前值, 作为 PROFIBUS 输入的第 1 字节 ----*/
trbox[2]=di8_15; /*---- di8_15 是输入键输入 I8~I15 当前值, 作为 PROFIBUS 输入的第 2 字节 ----*/

for (i=3;i<dil;i++)
{
trbox[i]=0x5a; /*---- 其它 PROFIBUS 输入数据(3~48)本例定为 0x5a ----*/
};

x=0;
for (i=0;i<dil;i++) /*---- 求校验和 ----*/
{
x=x+trbox[i];
};
trbox[dil]=x; /*---- 置校验和 ----*/

REN=0;
for (i=0;i<data_in_len;i++) /*---- 发送数据交换报文共 50 字节 ----*/
{
TI=0;
ACC=trbox[i];
TB8=P; /*---- 形成偶校验位 ----*/
SBUF=trbox[i];
while (TI==0) {};
};

/*=====发送数据交换报文结束, 等待接收接口板 SE 的回答。=====*/
re_er=0; /*----- re_er 偶校验错标志 -----*/
RI=0;
REN=1; /*----- 允许接收 -----*/
for (j=0;j<data_out_len;j++) /*----- 等待接收接口板的回答数据交换报文共 50 字节 -----*/
{
while (RI==0) {}; /*----- 等待接收一个字符 -----*/
ACC=SBUF;
if (P!=RB8) {re_er=1;}; /*----- 偶校验错, 置偶校验错标志 -----*/
rebox[j]=SBUF; /*----- 接收一个字符送串口输入缓冲区 -----*/
}
}

```

```

    RI=0;
};
/*=====接收完毕=====*/
if (re_er==0)
{
    /*----- 无偶校验错 -----*/
    x=0;
    for (i=0;i<dol;i++)
    {
        x=x+rebox[i];
        /*----- 求校验和 -----*/
    };
    if (x==rebox[dol])
    {
        y=dol-1;
        /*----- y=49-1=48,PROFIBUS 输出数据长度 -----*/
        for (i=1;i<dol;i++)
        {
            pb_out[i-1]=rebox[i];
            /*----- 接收到的 48 个 PROFIBUS 输出数据送数据区 pb_out[0~47] -----*/
        };
    }
    else
    {
        error=0x81;
        /*----- 校验和错 -----*/
    };
}
else
{
    error=0x83;
    /*----- 偶校验错 -----*/
};
/*=====*/
/*-----显示部分-----*/
/*=====*/
scanin();
/*----- 输入键扫描 -----*/
steout0_7(pb_out[0]);
/*----- PROFIBUS 输出数据第一个字节送 Q0-Q7 显示 -----*/
steout8_15(pb_out[1]);
/*----- PROFIBUS 输出数据第一个字节送 Q0-Q7 显示 -----*/
};/*for(;;)*/
}

```

(7) 例 1 的 GSD 文件

```

;=====
; GSD 文件 : OEM2-B1 - 48DI/48DO /SPC3 鼎实创新科技有限责任公司
; 产品型号: PB-OEM2-SE
; 版本 : 2004 年 9 月 V1.0
; 文件名 : OEM2-B1.GSD
;=====
#Profibus_DP
;<Prm-Text-Def-List>
;<Ext-User-Prm-Data-Def-List>
;<Unit-Definition-List>
GSD_Revision=2
Vendor_Name="DS FieldBus Ltd. Co." ; 公司名, 可按用户名修改
Model_Name="PB-OEM2-SE-B1" ; 模块名称, 也是组态时该产品的名称, 组态软件如 STEP 7, COM PROFIBUS
Revision="V1.0 "
Ident_Number=0x06FA ;ID 号, 必须与初始化报文一致
Protocol_Ident=0
Station_Type=0
Hardware_Release="A1.0 "
Software_Release="Z1.0 "
9.6_supp=1
19.2_supp=1
93.75_supp=1
187.5_supp=1
500_supp=1

```

45.45_supp=1
1.5M_supp=1
3M_supp=1
6M_supp=1
12M_supp=1
MaxTsdR_9.6=60
MaxTsdR_19.2=60
MaxTsdR_45.45=250
MaxTsdR_93.75=60
MaxTsdR_187.5=60
MaxTsdR_500=100
MaxTsdR_1.5M=150
MaxTsdR_3M=250
MaxTsdR_6M=450
MaxTsdR_12M=800
Implementation_Type="spc3"
Bitmap_Device="SE_B" ; 图标文件, 用户可以自制图标, 缺省
; Slave-Specification:
OrderNumber="pb-oem2-s" ; 产品序列号, 可按用户名修改
Freeze_Mode_supp=1
Sync_Mode_supp=1
Auto_Baud_supp=1
Fail_safe=0
Min_Slave_Intervall=1
Max_Diag_Data_Len=6
User_Prm_Data_Len=0
Modular_Station = 0
Modul_Offset=0
Slave_Family=3@TdF@PB-OEM2 ; 组态中的分类名
; <Module-Definition-List>
Module=" 48 Byte In, 48 Byte Out " 0x1f,0x2f,0x1f,0x2f,0x1f,0x2f ; I/O 配置, 必须与初始化报文一致
EndModule

例 2. 监测 SE 接口状态的工作模式

例 2 是在“例 1”基础上，在串口通信中增加监测 SE 接口状态功能。检测 SE 接口状态的通信是一种半握手通信方式，可以避免死等、接收长度错误等故障，提高通信效率。

虽然“例 2”与“例 1”在技术指标、程序结构、GSD 文件方面绝大部分都是一致的，但考虑到有些读者可能没有阅读“例 1”而直接阅读“例 2”，因此本《手册》将例 1 的程序也重复列出，只是将这一部分重复文字用斜体字表示，如果读者不需要可越过这些段落。

(1) 技术参数：

PROFIBUS 输入/输出： 48 字节输入/48 字节输出；

用户参数： 无；

串口波特率： 115.2K

通信模式： 检测 SE 接口状态条件；

(2) SE 初始化报文

I/O 配置数据长度 $CFG_LEN=6$ ；

I/O 配置数据为： 0x1f, 0x2f, 0x1f, 0x2f, 0x1f, 0x2f

用户参数长度 $User_Prm_Data_Len=0$ ；

接收数据长 $SD_in_len \geq 48+1+1=50$

发送数据长 $SD_out_len \geq 48+1+1=50$

byte0	站号=实验板上 SW1
byte1	ID 号高位=0x06
byte2	ID 号低位=0xfa
byte3	CFG_LEN=0x06
byte 4	I/O 配置数据 1=0x1f
byte 5	I/O 配置数据 2=0x2f
byte 6	I/O 配置数据 3=0x1f
byte 7	
byte 23	
byte 24	接收数据长 $SD_in_len=50$
byte 25	发送数据长 $SD_out_len=50$
byte 26	用户参数长度 <i>User_Prm_Data_Len=0</i>
	***不用
byte 47	
byte 48	校验和 $\Sigma (0) \sim (47)$

实验板发送初始化报文

byte0	SE 接口波特率号 BR=1~7
byte 1	初始化报文错误号 N=0
byte 2	初始化成功标志=0xaa
byte 3	初始化成功标志=0xaa
byte 4	初始化成功标志=0xaa
byte 5	初始化成功标志=0xaa
byte 6	
byte 7	
byte 23	
byte 24	
byte 25	
byte 26	
byte 47	
byte 48	校验和 $\Sigma (0) \sim (47)$

SE 接口板回答的初始化成功报文

byte0	SE 接口波特率号 BR=1~7
byte 1	初始化报文错误号 N=01~06
byte 2	初始化失败标志=0x55
byte 3	初始化失败标志=0x55
byte 4	初始化失败标志=0x55
byte 5	初始化失败标志=0x55
byte 6	
byte 7	
byte 23	
byte 24	
byte 25	
byte 26	
byte 47	
byte 48	校验和 $\Sigma (0) \sim (47)$

接口板回答的初始化失败报文

(5) 实验板初始化 SE 接口板的过程

见图 4-2-4，黑色斜体部分与“例 1”相同，增加部分用蓝色正体标出。

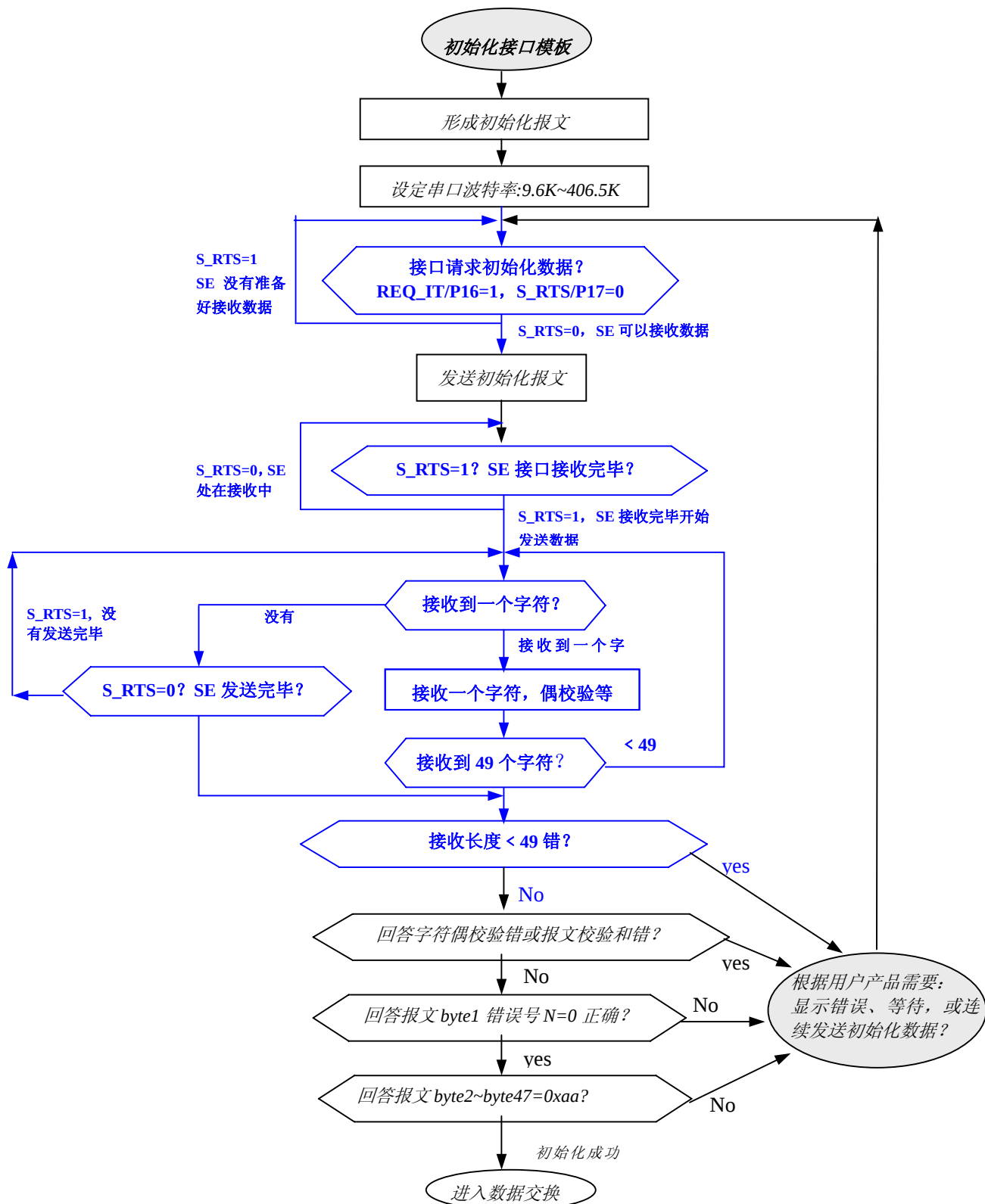


图 4-2-4 用户模板初始化接口模板的过程

(6) 数据交换状态下实验板带监测接口状态功能工作模式

见图 4-2-5，黑色斜体部分与“例 1”相同，增加部分用蓝色正体标出。

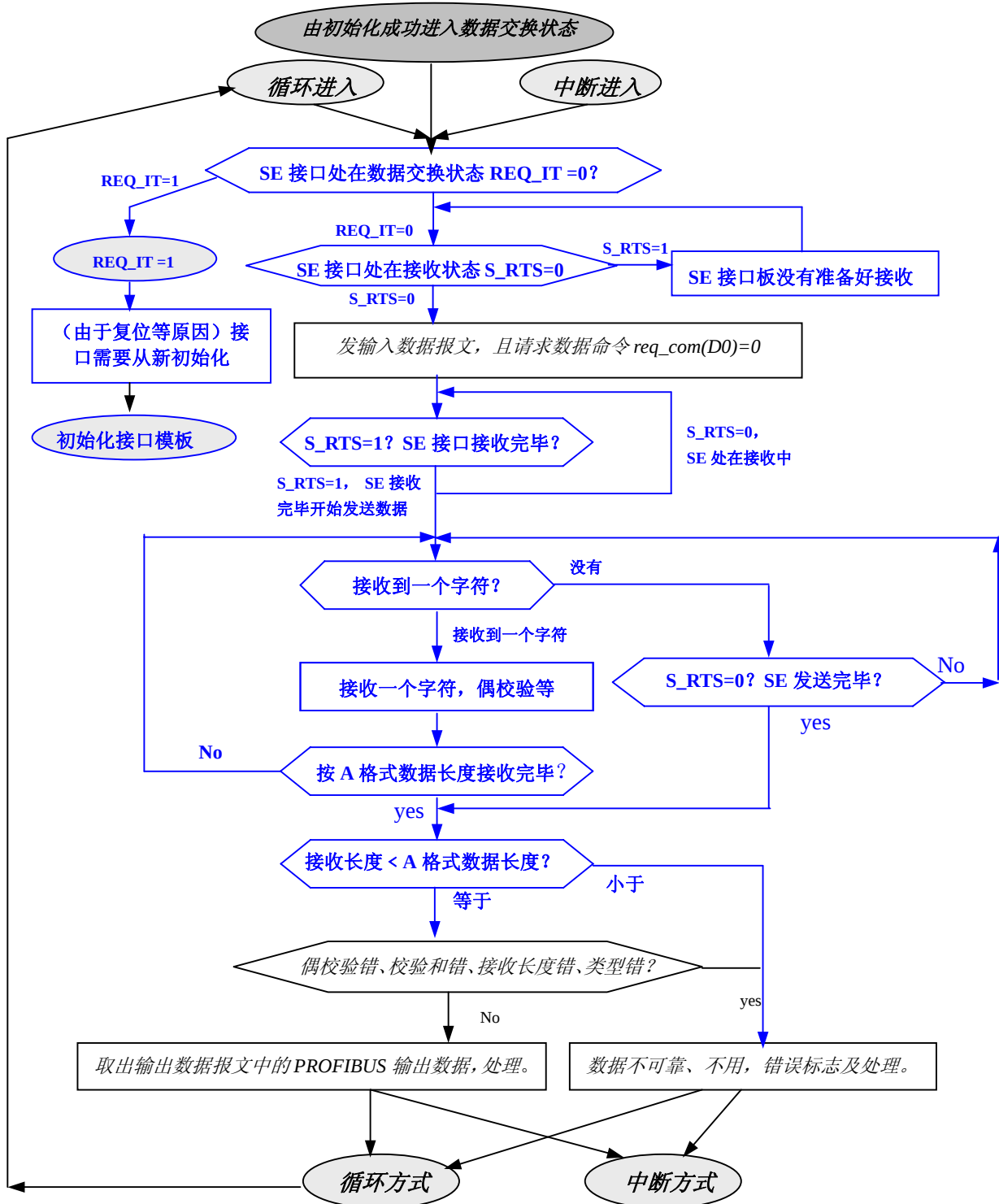


图 4-2-5 数据交换状态下用户模板的程序流程（监测接口状态条件、没有用户参数的工作模式）

(7) 实验板程序清单

例 2 的程序与例 1 相比，只增加了监测 SE 接口状态信号 REQ_IT 与 S_RTS/P17，其它大部分程序一致；以下增加部分是蓝色五号字，与例 1 相同部分仍然是黑色斜体小五号字。

```

/*+-----+
|文件名称: 实验板串口通信程序—监测 SE 接口状态的工作模式
|版 本: V1.1
|制作单位: 北京鼎实创新科技公司
|版权许可:
|网址: www.c-profibus.com.cn
|email: tangjy@c-profibus.com.cn
|日期: 2004/9
+-----+*/

#include <c:\51\c51\INC\reg52.h>
#define UBYTE unsigned char
#define UWORD unsigned int
/*=====*/
/* 以下是输入键、指示灯等设备变量定义，用户可以越过阅读 */
/*-----*/
sbit out0_7=P2^0; /* 定义 out0_7=P2.0=P16O0,输出指示灯 Q0-Q7 的输出控制,见实验板原理图 U16 -----*/
sbit out8_15=P2^1; /* 定义 out8_15=P2.1=P17O1,输出指示灯 Q8-Q15 的输出控制,见实验板原理图 U17 -----*/
sbit inled0_7=P2^2; /* 定义 inled0_7=P2.2=P18IO0,输入指示灯 I0-I7 的输出控制,见实验板原理图 U18 -----*/
sbit inled8_15=P2^3; /* 定义 inled8_15=P2.3=P19IO1,输入指示灯 I8-I15 的输出控制,见实验板原理图 U19 -----*/
sbit in0_7=P2^4; /* 定义 in0_7=P2^4=P2.4=P12IO,输入键 I0-I7 的读控制,见实验板原理图 U12 -----*/
sbit in8_15=P2^5; /* 定义 in8_15=P2^5=P2.5=P13I1,输入键 I8-I15 的读控制,见实验板原理图 U13 -----*/
sbit inadd=P2^6; /* 定义 inadd=P2^6=P2.6=P20AD,PROFIBUS 地址输入 AD0-AD7 的读控制,
/* 见实验板原理图 U20 -----*/
sbit REQ_IT=P1^7; /* 定义 REQ_IT=P1^7=P1.7=P17S_CTS,输入,连接 OEM2-SE-REQ_IT,
/* 见实验板原理图 CONNECT-16 -----*/
sbit S_RTS=P1^6; /* 定义 S_RTS=P1^6=P1.6=P17S_RTS,输入,连接 OEM2-SE-S_RTS,
/* 见实验板原理图 CONNECT-16 -----*/
sbit P15RES=P1^5; /* 定义 P15RES=P1^5=RES,输入,连接 OEM2-SE-RES,
/* 见实验板原理图 CONNECT-16 -----*/

/***** 定义: di0_7:字节型,是输入键 I0-I7 当前值; *****/
/***** 定义: di8_15:字节型,是输入键 I8-I15 当前值; *****/
bdata UBYTE di0_7,di8_15; /* di0_7:字节型,输入键 I0-I7;di8_15:字节型,输入键 I8-I15 -----*/
sbit di0=di0_7^0; /* 定义 di0=di0_7^0=输入键 I0 -----*/
sbit di1=di0_7^1; /* 定义 di1=di0_7^1=输入键 I1 -----*/
sbit di2=di0_7^2; /* 定义 di2=di0_7^2=输入键 I2 -----*/
sbit di3=di0_7^3; /* 定义 di3=di0_7^3=输入键 I3 -----*/
sbit di4=di0_7^4; /* 定义 di4=di0_7^4=输入键 I4 -----*/
sbit di5=di0_7^5; /* 定义 di5=di0_7^5=输入键 I5 -----*/
sbit di6=di0_7^6; /* 定义 di6=di0_7^6=输入键 I6 -----*/
sbit di7=di0_7^7; /* 定义 di7=di0_7^7=输入键 I7 -----*/
sbit di8=di8_15^0; /* 定义 di8=di8_15^0=输入键 I8 -----*/
sbit di9=di8_15^1; /* 定义 di9=di8_15^1=输入键 I9 -----*/
sbit di10=di8_15^2; /* 定义 di10=di8_15^2=输入键 I10 -----*/
sbit di11=di8_15^3; /* 定义 di11=di8_15^3=输入键 I11 -----*/
sbit di12=di8_15^4; /* 定义 di12=di8_15^4=输入键 I12 -----*/
sbit di13=di8_15^5; /* 定义 di13=di8_15^5=输入键 I13 -----*/
sbit di14=di8_15^6; /* 定义 di14=di8_15^6=输入键 I14 -----*/
sbit di15=di8_15^7; /* 定义 di15=di8_15^7=输入键 I15 -----*/

/***** 定义: key0_7:字节型,是输入键 I0-I7 扫描读入值; *****/
/***** 定义: key8_15:字节型,是输入键 I8-I15 扫描读入值; *****/
bdata UBYTE key0_7,key8_15;
sbit key0=key0_7^0; /* 定义 key0=key0_7^0=输入键 I0 -----*/
sbit key1=key0_7^1; /* 定义 key1=key0_7^1=输入键 I1 -----*/
sbit key2=key0_7^2; /* 定义 key2=key0_7^2=输入键 I2 -----*/
sbit key3=key0_7^3; /* 定义 key3=key0_7^3=输入键 I3 -----*/
sbit key4=key0_7^4; /* 定义 key4=key0_7^4=输入键 I4 -----*/
sbit key5=key0_7^5; /* 定义 key5=key0_7^5=输入键 I5 -----*/

```

```

sbit key6=key0_7^6;      /* 定义key6=key0_7^6=输入键I6          -----*/
sbit key7=key0_7^7;      /* 定义key7=key0_7^7=输入键I7          -----*/
sbit key8=key8_15^0;     /* 定义key8=key8_15^0=输入键I8          -----*/
sbit key9=key8_15^1;     /* 定义key9=key8_15^1=输入键I9          -----*/
sbit key10=key8_15^2;    /* 定义key10=key8_15^2=输入键I10        -----*/
sbit key11=key8_15^3;    /* 定义key11=key8_15^3=输入键I11        -----*/
sbit key12=key8_15^4;    /* 定义key12=key8_15^4=输入键I12        -----*/
sbit key13=key8_15^5;    /* 定义key13=key8_15^5=输入键I13        -----*/
sbit key14=key8_15^6;    /* 定义key14=key8_15^6=输入键I14        -----*/
sbit key15=key8_15^7;    /* 定义key15=key8_15^7=输入键I15        -----*/

/***** 定义: P00-P07 对应 P0 口的 P0.0-P0.7 *****/
sbit P00=P0^0;
sbit P01=P0^1;
sbit P02=P0^2;
sbit P03=P0^3;
sbit P04=P0^4;
sbit P05=P0^5;
sbit P06=P0^6;
sbit P07=P0^7;

/***** PP00-PP07 是上面 P00-P07 的中间变量 *****/
bdata UBYTE PP0;
sbit PP0=PP0^0;
sbit PP1=PP0^1;
sbit PP2=PP0^2;
sbit PP3=PP0^3;
sbit PP4=PP0^4;
sbit PP5=PP0^5;
sbit PP6=PP0^6;
sbit PP7=PP0^7;

/**** 定义: AB_status: 字节型, 报文类型及接口状态, 见《PB-OEM2-SE 产品手册》第四章、三、数据交换 *****/
bdata UBYTE AB_status;
sbit AB_sD0=AB_status^0; /***** AB_sD0=AB_status^0: 用户参数更新标记 *****/
sbit AB_sD7=AB_status^7; /***** AB_sD7=AB_status^7: 报文格式 *****/

/**** 定义: req_com: 字节型, 请求数据命令, 见《PB-OEM2-SE 产品手册》第四章、三、数据交换 *****/
bdata UBYTE req_com;
sbit req_comD0=req_com^0; /***** req_comD0=req_com^0: 请求数据/参数命令 *****/
sbit req_comD1=req_com^1; /***** req_comD1=req_com^1: 清参数命令 *****/

/*-----*/
/*          输入键、指示灯等设备变量定义完毕          */
/*=====*/

/*=====*/
/*          以下是有关输入键、指示灯控制子程序, 用户了解功能即可          */
/*-----*/

/***** 将 V 值送指示灯 Q0-Q7 输出 *****/
void steout0_7(UBYTE v)
{
    out0_7=0;
    P0=v;
    P0=v;
    P0=v;
    P0=v;
    out0_7=1;
    out0_7=0;
}

/***** 将 V 值送指示灯 Q8-Q15 输出 *****/
void steout8_15(UBYTE v)
{
    out8_15=0;

```

```
P0=v;
P0=v;
P0=v;
P0=v;
out8_15=1;
out8_15=0;
}
```

/****** 将V 值送输入键指示灯ILED0-LIED7 输出******/

```
void steinled0_7(UBYTE v)
{
inled0_7=0;
P0=v;
P0=v;
P0=v;
P0=v;
inled0_7=1;
inled0_7=0;
}
```

/****** 将V 值送输入键指示灯ILED8-LIED15 输出******/

```
void steinled8_15(UBYTE v)
{
inled8_15=0;
P0=v;
P0=v;
P0=v;
P0=v;
inled8_15=1;
inled8_15=0;
}
```

/****** 读PROFIBUS 地址开关值******/

```
UBYTE rd_address()
{
UBYTE data x;
P0=0xff;
inadd=0;
x=P0;
x=P0;
x=P0;
x=P0;
inadd=1;
return(x);
}
```

/****** 扫描一次输入键I0-I15，更新一次输入键当前值******/
 /****** 对输入键I0-I15 有0-1 者，当前值取反******/

```
void scanin()
{
P0=0xff;
in0_7=0;
PP0=P0;
PP0=P0;
PP0=P0;
PP0=P0;
PP0=P0;
if (key0_7!=PP0)
{
if ((PP0==1)&&(key0==0)) { di0=~di0;};
if ((PP1==1)&&(key1==0)) { di1=~di1;};
if ((PP2==1)&&(key2==0)) { di2=~di2;};
if ((PP3==1)&&(key3==0)) { di3=~di3;};
if ((PP4==1)&&(key4==0)) { di4=~di4;};
if ((PP5==1)&&(key5==0)) { di5=~di5;};
if ((PP6==1)&&(key6==0)) { di6=~di6;};
if ((PP7==1)&&(key7==0)) { di7=~di7;};
}
```

```

    key0_7=PP0;
};
in0_7=1;
inled0_7=0;
P0=di0_7;
P0=di0_7;
P0=di0_7;
P0=di0_7;
inled0_7=1;
inled0_7=0;

P0=0xff;
in8_15=0;
PP0=P0;
PP0=P0;
PP0=P0;
PP0=P0;
if (key8_15!=PP0)
{
    if ((PP0==1)&&(key8==0)) { di8=~di8;};
    if ((PP1==1)&&(key9==0)) { di9=~di9;};
    if ((PP2==1)&&(key10==0)) { di10=~di10;};
    if ((PP3==1)&&(key11==0)) { di11=~di11;};
    if ((PP4==1)&&(key12==0)) { di12=~di12;};
    if ((PP5==1)&&(key13==0)) { di13=~di13;};
    if ((PP6==1)&&(key14==0)) { di14=~di14;};
    if ((PP7==1)&&(key15==0)) { di15=~di15;};
    key8_15=PP0;
};
in8_15=1;
inled8_15=0;
P0=di8_15;
P0=di8_15;
P0=di8_15;
P0=di8_15;
inled8_15=1;
inled8_15=0;
}

/*=====*/
/*                      主程序                      */
/*-----*/

void    main ()
{
    UBYTE data i,j,k,x,y,count,frist;
    UWORD data nw,nn;
    UBYTE xdata * data p;
    UBYTE data address;          /*-----从站地址-----*/
    UBYTE data re_er;            /*-----字符偶校验错-----*/
    UBYTE data br_num;           /*-----串口波特率号-----*/
    UBYTE data error;            /*-----校验和错-----*/
    UBYTE xdata rebox[113];       /*-----串口接收缓冲-----*/
    UBYTE xdata trbox[113];       /*-----串口发送缓冲-----*/
    UBYTE xdata pb_out[112];      /*-----PROFIBUS 输出数据区-----*/
    UBYTE xdata pb_in[112];       /*-----PROFIBUS 输入数据区-----*/
    UBYTE xdata user_prm[41];     /*-----PROFIBUS 用户参数区-----*/
    UBYTE data data_in_len;       /*-----数据交换输入（实验板发送）数据长度-----*/
    UBYTE data dil;               /*-----不包括校验和的data_in_len 长度-----*/
    UBYTE data data_out_len;      /*-----数据交换输出（实验板接收）数据长度-----*/
    UBYTE data dol;               /*-----不包括校验和的data_out_len 长度-----*/
    UBYTE data user_prm_len;      /*-----用户参数长度-----*/
    UBYTE data se_er;             /*-----OEM2-SE 接收数据错误类型号-----*/

    P1=0xff;
    P0=0;
    P2=0xf0;

```

```

di0_7=di8_15=0;

steout0_7(0);          /*-----输出指示灯Q0-Q7 清零          -----*/
steout8_15(0);         /*-----输出指示灯Q8-Q15 清零          -----*/
steinled0_7(0);        /*-----输入键指示灯ILED0-ILED7 清零        -----*/
steinled8_15(0);       /*-----输入键指示灯ILED8-ILED15 清零       -----*/

/*=====*/
/*  以下是实验板上电后, 检测 P89V51RD2 单片机内部 RAM 的一段输出演示程序;          */
/*  这段程序与 PROFIBUS 接口板 SE 通信无关, 用户可跳过此段程序。                      */
/*-----*/

p=0;
x=1;
k=1;
for (j=0;j<32;j++)
{
    for (i=0;i<24;i++)
    {
        *(p+i)=0xaa;
    };
    y=0;
    for (i=0;i<24;i++)
    {
        count=*(p+i);
        if (count!=0xaa) { y=1; };
    };
    if (y==0)
    {
        switch (k)
        {
            case 1:{ steout0_7(x);};break;
            case 2:{ steout8_15(x);};break;
            case 3:{ steinled0_7(x);};break;
            case 4:{ steinled8_15(x);};break;
            default:break;
        };
        if (x<255) {x=x*2+1;}else{x=1;k++;};
    }
    else
    {
        for (;;)
        {
            for (nw=0;nw<6000;nw++)
            {
                steout0_7(0);
                steout8_15(0);
                steinled0_7(0);
                steinled8_15(0);
            };
            for (nw=0;nw<6000;nw++)
            {
                steout0_7(0xff);
                steout8_15(0xff);
                steinled0_7(0xff);
                steinled8_15(0xff);
            };
        };
        p=p+24;
        for (nw=0;nw<8000;nw++)
        {
            ;
        };
    };
}

/*-----*/
/*  实验板输出演示程序结束。                      */
/*=====*/

```

```

steout0_7(0);          /*-----输出指示灯Q0-Q7 清零-----*/
steout8_15(0);         /*-----输出指示灯Q8-Q15 清零-----*/
steinled0_7(0);        /*-----输入键指示灯ILED0-ILED7 清零-----*/
steinled8_15(0);       /*-----输入键指示灯ILED8-ILED15 清零-----*/

/*=====*/
/* 清串口输入缓冲区 trbox[i], 清串口输出缓冲区 rebox[i] */
/*-----*/
for (i=0;i<113;i++)
{
    rebox[i]=trbox[i]=0;
};
for (i=0;i<112;i++)
{
    pb_in[i]=pb_out[i]=0;
};

address=rd_address(); /*-----读实验板上拨码开关, 作为站号-----*/

/*=====*/
/* 以下是串口初始化程序 */
/*-----*/
br_num=5;              /*-----实验板波特率选择5, 115.2K-----*/
switch (br_num)        /*-----根据波特率号设置T2-----*/
{
    case 1:{ TH2=0x0ff; TL2=0x0a0;RCAP2H=0x0ff;RCAP2L=0x0a0;};break; /*---9.6K---*/
    case 2:{ TH2=0x0ff; TL2=0x0d0;RCAP2H=0x0ff;RCAP2L=0x0d0;};break; /*---19.2K---*/
    case 3:{ TH2=0x0ff; TL2=0x0e8;RCAP2H=0x0ff;RCAP2L=0x0e8;};break; /*---38.4K---*/
    case 4:{ TH2=0x0ff; TL2=0x0f0;RCAP2H=0x0ff;RCAP2L=0x0f0;};break; /*---57.6K---*/
    case 5:{ TH2=0x0ff; TL2=0x0f8;RCAP2H=0x0ff;RCAP2L=0x0f8;};break; /*---115.2K---*/
    case 6:{ TH2=0x0ff; TL2=0x0fc;RCAP2H=0x0ff;RCAP2L=0x0fc;};break; /*---230.4K---*/
    case 7:{ TH2=0x0ff; TL2=0x0fe;RCAP2H=0x0ff;RCAP2L=0x0fe;};break; /*---460.8K---*/
    default:break;
};

/*-----*/
/* 注意: 本例基于 PHILIP P89V51RD2 单片机: */
/* 波特率=14745600/16/ (65536-T) */
/* 各种波特率 9600: T=0xffa0, 19.2K: T=0xffd0, 38.4K: T=0xffe8, */
/* 57.6K: T=0xfff0, 115.2K: T=0xfffb, 230.4k: T=0xfffc, 460.8k: T=0xfffe */
/*-----*/
PCON=0;
T2CON=0x34;          /*---RCLK=1; TCLK=1; TR2=1; C/T2=0-----*/
/*-----*/
/* 串口设置成模式3 即: 1 个起始位、8 个数据位、1 个偶校验位、1 个停止位; */
/* 0xd0=11010000B; SM0SM1=11; SM2=0; REN=1;TB8=0; RB8=0; TI=0; RI=0 */
/*-----*/
SCON=0x0d0;

/*-----*/
/* 串口初始化程序结束 */
/*-----*/
while (REQ_IT==1)
{
    /*=====*/
    /* 形成接口板的初始化报文 */
    /* rebox[]是串口发送数组, 格式见《SE 手册》 */
    /*-----*/
    data_in_len=48+1+1; /*---输入数据报文长度=48+1+1=50;-----*/
    user_prm_len=0;     /*---用户参数长度=0-----*/
    data_out_len=48+1+1; /*---输入数据报文长度=48+1+1=50;-----*/
    trbox[0]=address;   /*---BYTE0=从站地址-----*/
    trbox[1]=0x06;      /*---BYTE1=ID 号高位-----*/
    trbox[2]=0x0fa;     /*---BYTE2=ID 号低位-----*/
    /*=====*/
    /* 以下形成 I/O 配置数据 */
    /*-----*/

```

```

/* 本例: I/O 配置数据长度=6; I/O 配置数据: 0x1f,0x2f,0x1f,0x2f,0x1f,0x2f */
/* 共: 48bytes input+48bytes output */
/*-----*/
trbox[3]=6; /*---BYTE3=I/O 配置数据长度=6 -----*/
trbox[4]=0x1f; /*---BYTE4=I/O 配置数据: 0x1f -----*/
trbox[5]=0x2f; /*---BYTE5=I/O 配置数据: 0x2f -----*/
trbox[6]=0x1f; /*---BYTE6=I/O 配置数据: 0x1f -----*/
trbox[7]=0x2f; /*---BYTE7=I/O 配置数据: 0x2f -----*/
trbox[8]=0x1f; /*---BYTE8=I/O 配置数据: 0x1f -----*/
trbox[9]=0x2f; /*---BYTE9=I/O 配置数据: 0x2f -----*/

trbox[24]=data_in_len; /*---BYTE24=data_in_len 输入数据报文长度 -----*/
trbox[25]=data_out_len; /*---BYTE25=data_out_len 输出数据报文长度 -----*/
trbox[26]=user_prm_len; /*---BYTE26=user_prm_len 用户参数长度 -----*/

x=0;
for (i=0;i<48;i++)
{
    x=x+trbox[i]; /*--- 求校验和 -----*/
};
trbox[48]=x; /*--- 置校验和 -----*/

/*=====*/
/* 发送接口板的初始化报文、接收接口板的回答, 初始不成功继续发初始化报文 */
/*-----*/
while (S_RTS==1) {}; /*---- P1.6=S_RTS=1: SE 没有准备好接收 -----*/

y=0; /*---- 清初始化成功标志 -----*/
while (y==0)
{
    REN=0;
    for (i=0;i<49;i++) /*---- 发送一次初始化报文共 49 字节 -----*/
    {
        TI=0;
        ACC=trbox[i];
        TB8=P; /*---- 形成偶校验位 -----*/
        SBUF=trbox[i];
        while (TI==0) {}
    };

    while (S_RTS==0) {}; /*---- SE 没有接收完毕 -----*/

    /*----- 发送完毕、等待接收回答 -----*/
    RI=0;
    REN=1; /*---- 允许接收 -----*/
    re_er=0; /*---- re_er 偶校验错标志 -----*/
    for (j=0;j<49;j++) /*---- 接受回答报文共 49 个字节 -----*/
    {
        while ((RI==0)&&(S_RTS==1)) {}; /*---- SE 正在发送、实验板正在接收一个字符 -----*/
        if (RI==1)
        {
            /*---- RI==1:实验板接收到一个字符 -----*/
            ACC=SBUF;
            if (P!=RB8){ re_er=1;}; /*---- 偶校验错, 置偶校验错标志 -----*/
            rebox[j]=SBUF; /*---- 接收一个字符送串口输入缓冲区 -----*/
            RI=0;
        }
        else
        {
            /*---- (RI==0)&&(S_RTS==0):SE 报文全部发送完毕而实验板还没有接收到预期的长度 -----*/
            j=100; /*---- 退出等待接收 -----*/
        }
    }
};

/*----- 接收回答报文结束 -----*/
/*----- 以下分析接口板 SE 的回答报文 -----*/

```

```

if (j==49)
{ /*---- (j==49):实验板接收回答报文长度正确 -----*/
if (re_er==0) /*---- 没有字符偶校验错? -----*/
{
x=0;
for (i=0;i<48;i++) /*---- 求校验和 -----*/
{
x=x+rebox[i];
};
if (x==rebox[48])
{ /*---- 校验和正确? -----*/
if (rebox[0]==br_num) /*---- 返回的波特率号是5? -----*/
{
if (rebox[1]==0) /*---- 初始化报文错误号 N=0? -----*/
{
x=0;
for (i=2;i<48;i++) /*---- byte2~byte47 都是 0xaa? -----*/
{
if (rebox[i]!=0xaa)
{
x=1;
};
};
if (x==0)
{
y=1; /*---- byte2~byte47 都是 0xaa,初始化成功,y=1,准备退出 while (y==0) ----*/
}
else
{x=0x11;}; /*---- byte2~byte47 不都是 0xaa ! 错误标记=0x11; ----*/
}
else
{x=rebox[1];}; /*---- 初始化报文错误号 N<>0, 错误标记= 返回错误号 ----*/
}
else
{x=0x77;}; /*---- 返回的波特率号不是 5, 错误标记=0x77; ----*/
}
else
{x=0xff;}; /*---- 校验和不正确正确, 错误标记=0xff; ----*/
}
else
{x=0xee;}; /*---- 发生了字符偶校验错 错误标记=0xee; ----*/
}
else
{
x=0xcc; /*---- j!=49 发生了接收报文长度错 错误标记=0xee; ----*/
};

if (y==0)
{ /*---- 初始化失败 ----*/
for (nw=0;nw<10000;nw++)
{
steout0_7(br_num); /*---- 显示返回波特率号 ----*/
steout8_15(x); /*---- 显示返回错误号 ----*/
};
};

}; /*---- 如果 y==0,返回 while (y==0) ----*/

/*=====*/
/* 实验板对 PROFIBUS 接口板初始化成功, 进数据交换状态的无限循环中 */
/* trbox[]/rebox[] 是串口发送/接收数组, 发送/接收格式见《SE 手册》 */
/*-----*/

dil=data_in_len-1; /*---- data_in_len=50, dil=49 ----*/
dol=data_out_len-1; /*---- data_out_len==50, dol==49 ----*/
for (i=0;i<data_in_len;i++)

```

```

{
    trbox[i]=0;          /*---- 清发送、接收缓冲区 ----*/
};
for (i=0;i<data_out_len;i++)
{
    rebox[i]=0;          /*---- 清发送、接收缓冲区 ----*/
};

/*===== 进数据交换状态的无限循环中 =====*/
while (REQ_IT==0)
{ /*---- (REQ_IT==0):SE 处在数据交换状态 ----*/
    trbox[0]=req_com=0;   /*---- 没有用户参数，请求数据命令永远是 req_com=0，只请求 A 型报文 ----*/
    trbox[1]=di0_7;       /*---- di0_7 是输入键输入 I0~I7 当前值，作为 PROFIBUS 输入的第 1 字节 ----*/
    trbox[2]=di8_15;      /*---- di8_15 是输入键输入 I8~I15 当前值，作为 PROFIBUS 输入的第 2 字节 ----*/

    for (i=3;i<dil;i++)
    {
        trbox[i]=0x5a;    /*---- 其它 PROFIBUS 输入数据(3~48)本例定为 0x5a ----*/
    };

    x=0;
    for (i=0;i<dil;i++)   /*---- 求校验和 ----*/
    {
        x=x+trbox[i];
    };
    trbox[dil]=x;         /*---- 置校验和 ----*/

    REN=0;

    while (S_RTS==1) {};  /*---- S_RTS==1: 接口板没有准备好接收数据 ----*/

    for (i=0;i<data_in_len;i++) /*---- 发送数据交换报文共 50 字节 ----*/
    {
        TI=0;
        ACC=trbox[i];
        TB8=P;           /*---- 形成偶校验位 ----*/
        SBUF=trbox[i];
        while (TI==0) {};
    };

    while (S_RTS==0) {};  /*---- SE 没有接收完毕 ----*/

    /*===== 发送数据交换报文结束，等待接收接口板 SE 的回答。 =====*/
    re_er=0;              /*----- re_er 偶校验错标志 ----*/
    RI=0;
    REN=1;                /*----- 允许接收 ----*/
    for (j=0;j<data_out_len;j++) /*----- 等待接收接口板的回答数据交换报文共 50 字节 ----*/
    {
        while ((RI==0)&&(S_RTS==1)) {}; /*---- SE 正在发送、实验板正在接收一个字符 ----*/
        if (RI==1)
        { /*---- RI==1:实验板接收到一个字符 ----*/
            ACC=SBUF;
            if (P!=RB8) {re_er=1;}; /*----- 偶校验错，置偶校验错标志 ----*/
            rebox[j]=SBUF; /*----- 接收一个字符送串口输入缓冲区 ----*/
            RI=0;
        }
        else
        { /*---- (RI==0)&&(S_RTS==0):SE 报文全部发送完毕而实验板还没有接收到预期的长度 ----*/
            j=data_out_len+1; /*---- 退出等待接收 ----*/
        };
    };
};

/*===== 接收完毕 =====*/

if (j==data_out_len)
{ /*---- (j==data_out_len):实验板接收回答报文长度正确 ----*/

```

```

if (re_er==0)
{
/*----- 无偶校验错 -----*/
    x=0;
    for (i=0;i<dol;i++)
    {
        x=x+rebox[i]; /*----- 求校验和 -----*/
    };
    if (x==rebox[dol])
    {
        y=dol-1; /*----- y=49-1=48,PROFIBUS 输出数据长度 -----*/
        for (i=1;i<dol;i++)
        {
            pb_out[i-1]=rebox[i]; /*----- 接收到的48 个 PROFIBUS 输出数据送数据区 pb_out[0~47] -----*/
        };
    }
    else
    {
        error=0x81; /*----- 校验和错 -----*/
    };
}
else
{
    error=0x83; /*----- 偶校验错 -----*/
};
}
else
{
    error=0x84; /*----- 接收字符数不够 -----*/
};
/*=====*/
/*-----显示部分-----*/
/*=====*/

scanin(); /*----- 输入键扫描 -----*/
steout0_7(pb_out[0]); /*----- PROFIBUS 输出数据第一个字节送 Q0-Q7 显示 -----*/
steout8_15(pb_out[1]); /*----- PROFIBUS 输出数据第一个字节送 Q0-Q7 显示 -----*/

};/*---- while (REQ_IT==0):SE 处在数据交换状态,返回数据交换状态的无限循环中 -----*/
};/*---- (REQ_IT==1):SE 退出数据交换状态,返回初始化 -----*/
}

```

(8) 例 2 的 GSD 文件

例 2 的 GSD 文件与例 1 完全一致，故一律用斜体小五号字表示。

```

;=====
; GSD 文件 : OEM2-B1 - 48DI/48DO /SPC3 鼎实创新科技有限责任公司
; 产品型号: PB-OEM2-SE
; 版本 : 2004 年 9 月 V1.0
; 文件名 : OEM2-B1.GSD
;=====
#Profibus_DP
; <Prm-Text-Def-List>
; <Ext-User-Prm-Data-Def-List>
; <Unit-Definition-List>
GSD_Revision=2
Vendor_Name="DS FieldBus Ltd. Co." ; 公司名, 可按用户名修改
Model_Name="PB-OEM2-SE-B1" ; 模块名称, 也是组态时该产品的名称, 组态软件如 STEP 7, COM PROFIBUS
Revision="V1.0 "
Ident_Number=0x06FA ;ID 号, 必须与初始化报文一致
Protocol_Ident=0
Station_Type=0
Hardware_Release="A1.0 "
Software_Release="Z1.0 "

```

```

9.6_supp=1
19.2_supp=1
93.75_supp=1
187.5_supp=1
500_supp=1
45.45_supp=1
1.5M_supp=1
3M_supp=1
6M_supp=1
12M_supp=1
MaxTsdR_9.6=60
MaxTsdR_19.2=60
MaxTsdR_45.45=250
MaxTsdR_93.75=60
MaxTsdR_187.5=60
MaxTsdR_500=100
MaxTsdR_1.5M=150
MaxTsdR_3M=250
MaxTsdR_6M=450
MaxTsdR_12M=800
Implementation_Type="spc3"
Bitmap_Device="SE_B"           ; 图标文件, 用户可以自制图标, 缺省
; Slave-Specification:
OrderNumber="pb-oem2-s"       ; 产品序列号, 可按用户名修改
Freeze_Mode_supp=1
Sync_Mode_supp=1
Auto_Baud_supp=1
Fail_safe=0
Min_Slave_Intervall=1
Max_Diag_Data_Len=6
User_Prm_Data_Len=0
Modular_Station = 0
Modul_Offset=0
Slave_Family=3@TdF@PB-OEM2    ; 组态中的分类名
; <Module-Definition-List>
Module=" 48 Byte In, 48 Byte Out " 0x1f,0x2f,0x1f,0x2f,0x1f,0x2f ;I/O 配置, 必须与初始化报文一致
EndModule

```

例 3. 带用户参数功能的工作模式

例 3 仍以实验板为模型，模拟实现一个 PROFIBUS I/O 模块；为方便起见称为实验 I/O。实验 I/O 设计将使用“用户参数”功能。因此本例将全面地讲解用户参数功能、带用户参数的 SE 接口初始化和带用户参数功能的数据交换过程。

(1) 需要使用“用户参数 user_prm”的情况

对于工业现场设备，常需要用户根据现场应用设定一些参数，其中有些参数不需要在设备运行中实时改变，如变频器的电流上限保护与报警值、温度传感器的测量温度范围、热电偶选型、输出 4-20mA/1-5V 选择等等。如果这些参数作为 PROFIBUS 主站的 I/O 输出，将占用 PROFIBUS 主站 I/O 资源和周期性轮循 PROFIBUS 从站的时间资源。

将这些参数处理成“用户参数”，将会缩短 PROFIBUS 主站通信时间、减小通信报文长度、提高总线通信效率。使用“用户参数”技术，只需要在主站配置中做出参数选择，主站在与从站连接时，一次性地将这些参数传送到从站，从站就可以使用这些用户选择的参数对从站进行参数化（初始化、参数设定等）。

(2) 实验 I/O 功能及实现方法

- ① 以实验板为模型，模拟实现一个 PROFIBUS I/O 从站模块，称为实验 I/O；
- ② 实验 I/O 具有配置灵活的特点，用户可以通过配置软件（如 STEP 7、COM PROFIBUS），将实验 I/O 配置成具有 1 路 AI+1 路 AO 的模拟量和 16DI+16DO 开关量的 PROFIBUS-DP I/O；
- ③ 对模拟量 I/O 可以通过配置选择输入/输出信号类型、数据格式等等；对开关量 I/O 可以通过配置选择输入/输出电压/功率等级、正/负逻辑等功能。
- ④ 为实现上述功能方法是：
 - （i）在实验 I/O 的 GSD 文件中详细描述用于配置的用户参数；
 - （ii）主站配置中由用户选择这些配置参数；
 - （iii）当主站与从站实验 I/O 连通时，主站将发送“参数化”命令，将用户配置参数下传至实验 I/O；
 - （iv）实验 I/O 将根据用户参数设置软硬件，实现 1 路 AI+1 路 AO 模拟量和 16DI+16DO 开关量 I/O 的功能。

(3) 具体确定“用户参数”类型、个数、取值范围

根据实验 I/O 的上述要求，首先需要具体确定“用户参数”的类型、个数、取值范围。

- ① 用户参数 1：输入类型 AI_M，字节型

AI_M=0~7：代表输入类型 1~5V、0~10V、0~5V、-10~+10V、4~20mA、0~20mA、0~10mA、-10~+10 mA 共八种选择；

② 用户参数 2: 数据类型 AI_D, 字节型

AI_D=0-2: 分别表示 BCD (0000~9999)、无符号整型 (0~65535)、有符号整型 (-32767~+32767) 共三种选择;

③ 用户参数 3: 单/双端 AI_CH, 字节型

AI_CH=0-1: 分别表示单端输入、双端输入, 共二种选择;

④ 用户参数 4: 输出类型 AO_M, 字节型

AO_M=0~7: 代表输出类型 1~5V、0~10V、0~5V、-10~+10V、4~20mA、0~20mA、0~10mA 共八种选择;

⑤ 用户参数 5: 数据类型 AO_D, 字节型

AO_D=0-2: 分别表示 BCD (0000~9999)、无符号整型 (0~65535)、有符号整型 (-32767~+32767) 共三种选择;

⑥ 用户参数 6: 单/双端 AO_CH, 字节型

AO_CH=0-1: 分别表示单端输出、双端输出共二种选择;

⑦ 用户参数 7: 正/负逻辑 DI_PN, 字节型

DI_PN=0-1: 分别表示正逻辑 (高有效)、负逻辑 (低有效) 这 2 种输入模式;

⑧ 用户参数 8: 电压等级 DI_V, 字节型

DI_V=0-3 分别表示 5V/DC、24V/DC、110V/AC、220V/AC 4 种输入电压等级;

⑨ 用户参数 9: 正/负逻辑 DO_PN, 字节型

DO_PN=0-1: 分别表示正逻辑 (高有效)、负逻辑 (低有效) 这 2 种输出模式;

⑩ 用户参数 10: 电压功率等级 DO_V, 字节型

DO_V=0-3 分别表示 5VDC/0.5A、24VDC/0.5A、110VAC/0.5A、220VAC/0.5A 4 种输出电压功率等级;

因此, 本例需要 10 个字节用户参数, 用户参数长度 User_Prm_Data_Len=10。

(4) 带有“用户参数”描述的 GSD 文件

与例 1、例 2 相比, 实验 I/O 的 GSD 文件至少有如下关于“用户参数”的描述, 见红色下划线部分;

下面是关于实验 I/O 的一个简单并带有“用户参数”描述的 GSD 文件:

```

;=====
; GSD文件 :   OEM2-B3  实验I/O  鼎实创新科技有限责任公司
; 产品型号:   PB-OEM2-实验I/O
; 版本   :    2004年9月 V1.0
; 文件名   :   OEM2-B3.GSD
;=====
#Profibus_DP
; <Prm-Text-Def-List>
; <Ext-User-Prm-Data-Def-List>
; <Unit-Definition-List>

```

```

GSD_Revision=2
Vendor_Name="DS FieldBus Ltd. Co." ;公司名，可按用户名修改
Model_Name="PB-OEM2-SE-B3" ;模块名称，也是组态时该产品的名称，组态软件如STEP 7，COM PROFIBUS
Revision="V1.0 "
Ident_Number=0x06FA ;ID号，必须与初始化报文一致
Protocol_Ident=0
Station_Type=0
Hardware_Release="A1.0 "
Software_Release="Z1.0 "
9.6_supp=1
19.2_supp=1
93.75_supp=1
187.5_supp=1
500_supp=1
45.45_supp=1
1.5M_supp=1
3M_supp=1
6M_supp=1
12M_supp=1
MaxTsdr_9.6=60
MaxTsdr_19.2=60
MaxTsdr_45.45=250
MaxTsdr_93.75=60
MaxTsdr_187.5=60
MaxTsdr_500=100
MaxTsdr_1.5M=150
MaxTsdr_3M=250
MaxTsdr_6M=450
MaxTsdr_12M=800
Implementation_Type="spc3"
Bitmap_Device="SE_B" ;图标文件，用户可以自制图标，缺省
; Slave-Specification:
OrderNumber="pb-oem2-s" ;产品序列号，可按用户名修改
Freeze_Mode_supp=1
Sync_Mode_supp=1
Auto_Baud_supp=1
Fail_safe=0
Min_Slave_Intervall=1
Max_Diag_Data_Len=6
User_Prm_Data_Len=10 ;用户参数长度
user_prm_data=0x00,0x01,0x00,0x00,0x01,0x00,0x00,0x00,0x00,0x00 ;用户参数默认值
Modular_Station = 0
Modul_Offset=0
Slave_Family=3@TdF@PB-OEM2 ;组态中的分类名
; <Module-Definition-List>
Module="1AI+1AO/16DI+16DO " 0x70, 0x31 ;1 word input +1 word output / 2 bytes input+ 2 bytes output
EndModule

```

这是实现实验I/O最简单的一个GSD文件，但用户使用却不方便。为了使用户更加了解每个参数含义，顺利完成实验I/O的配置，我们给用户提供了—张参数配置表，如表4-1所示：

表4-1 实验I/O的用户参数表

序号	名称	取值范围和设定
1	输入类型AI_M	AI_M=0~7: 代表输入类型 1~5V、0~10V、0~5V、-10~+10V、4~20mA、0~20mA、0~10mA、-10~+10 mA 共八种选择;

2	数据类型AI_D	AI_D=0-2: 分别表示 BCD (0000~9999)、无符号整型 (0~65535)、有符号整型 (-32767~+32767) 共三种选择;
3	单/双端AI_CH	AI_CH=0-1: 分别表示单端输入、双端输入, 共二种选择;
4	输出类型 AO_M	AO_M=0~7: 代表输出类型 1~5V、0~10V、0~5V、-10~+10V、4~20mA、0~20mA、0~10mA、-10~+10 mA 共八种选择;
5	数据类型AO_D	AO_D=0-2: 分别表示BCD (0000~9999)、无符号整型 (0~65535)、有符号整型 (-32767~+32767) 共三种选择;
6	单/双端AO_CH	AO_CH=0-1: 分别表示单端输出、双端输出, 共二种选择;
7	正/负逻辑 DI_PN	DI_PN=0-1: 分别表示正逻辑 (高有效)、负逻辑 (低有效) 这2种输入模式;
8	电压等级DI_V	DI_V=0-3 分别表示 5V/DC、24V/DC、110V/AC、220V/AC 4 种输入电压等级;
9	正/负逻辑 DO_PN	DO_PN=0-1: 分别表示正逻辑 (高有效)、负逻辑 (低有效) 这 2 种输出模式;
10	电压功率等级 DO_V	DO_V=0-3 分别表示 5VDC/0.5A、24VDC/0.5A、110VAC/0.5A、220VAC/0.5A 4 种输出电压功率等级;

由 GSD 文件中用户参数默认值:

user_prm_data=0x00,0x01,0x00,0x00,0x01,0x00,0x00,0x00,0x00,0x00

实验I/O的默认配置是:

AI_M=0: 代表输入类型1~5V;

AI_D=1: 表示无符号整型 (0~65535);

AI_CH=0: 表示单端输入;

AO_M=0: 代表输出类型1~5V;

AO_D=1: 表示无符号整型 (0~65535);

AO_CH=0: 表示单端输出;

DI_PN=0: 表示正逻辑 (高有效);

DI_V=0: 表示5V/DC;

DO_PN=0: 表示正逻辑 (高有效);

DO_V=0: 表示5VDC/0.5A;

(5) 如何在主站配置中选择用户参数

首先以STEP 7为例:

① 实验I/O的GSD文件OEM2_B3.GSD拷至: SIEMENS\step7\S7data\gsd\

实验I/O的图标文件SE_B.BMP拷至: SIEMENS\step7\S7data\nsbmp\

② 进入SIMATIC Manager → HARDWARE, 选择Options → Updata Catalog;

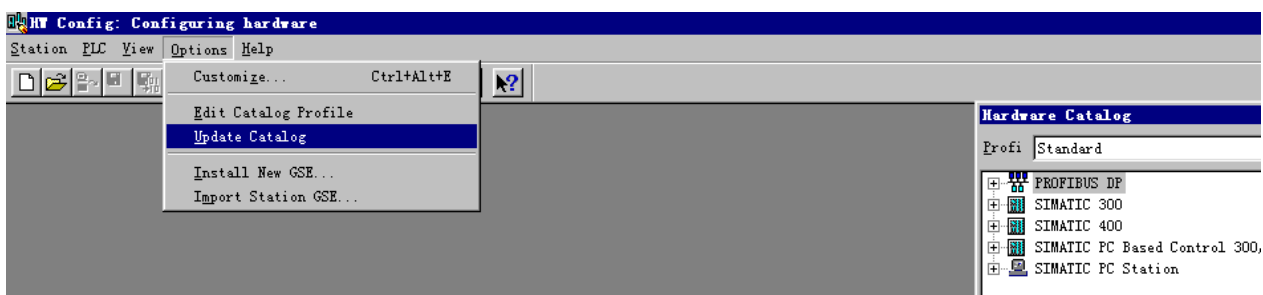


图 4-3 选择 Options → Update Catalog

③ 配置实验I/O从站。用户可以按照表4-1:实验I/O的用户参数表来改变用户参数配置实验I/O。

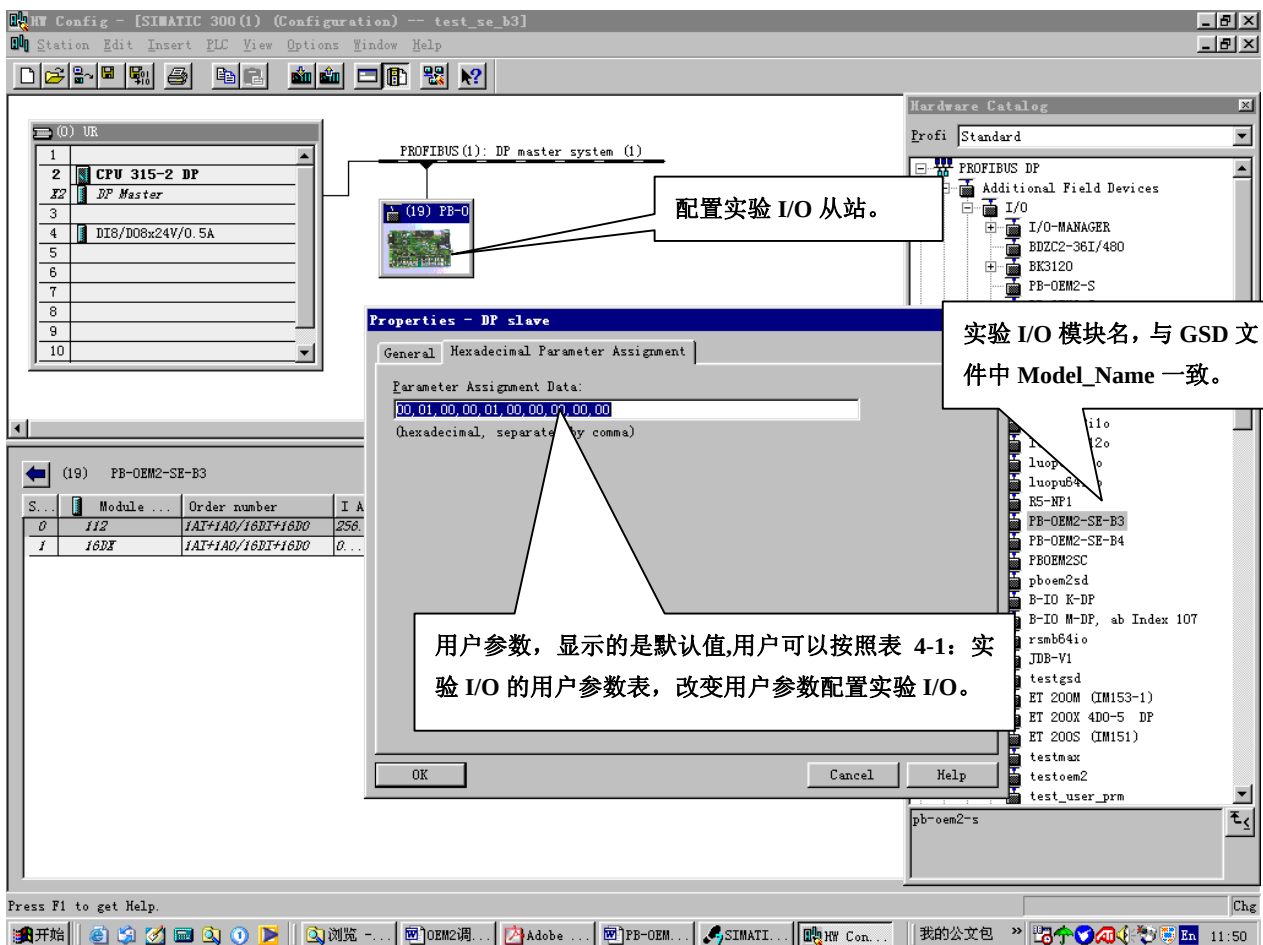


图 4-4 STEP 7—改变用户参数配置实验 I/O

再以COM PROFIBUS V5.1为例：

① 实验I/O的GSD文件OEM2_B3.GSD拷至：CPBV51\gsd\

实验I/O的图标文件SE_B.BMP改名SE_B.DIB拷至：CPBV51\bitmaps\

② 进入COM PROFIBUS V 5.1 → File → 选择 Read in GSD Files

③ 配置实验I/O从站。用户可以按照表4-1：实验I/O的用户参数表，改变用户参数配置实验I/O。

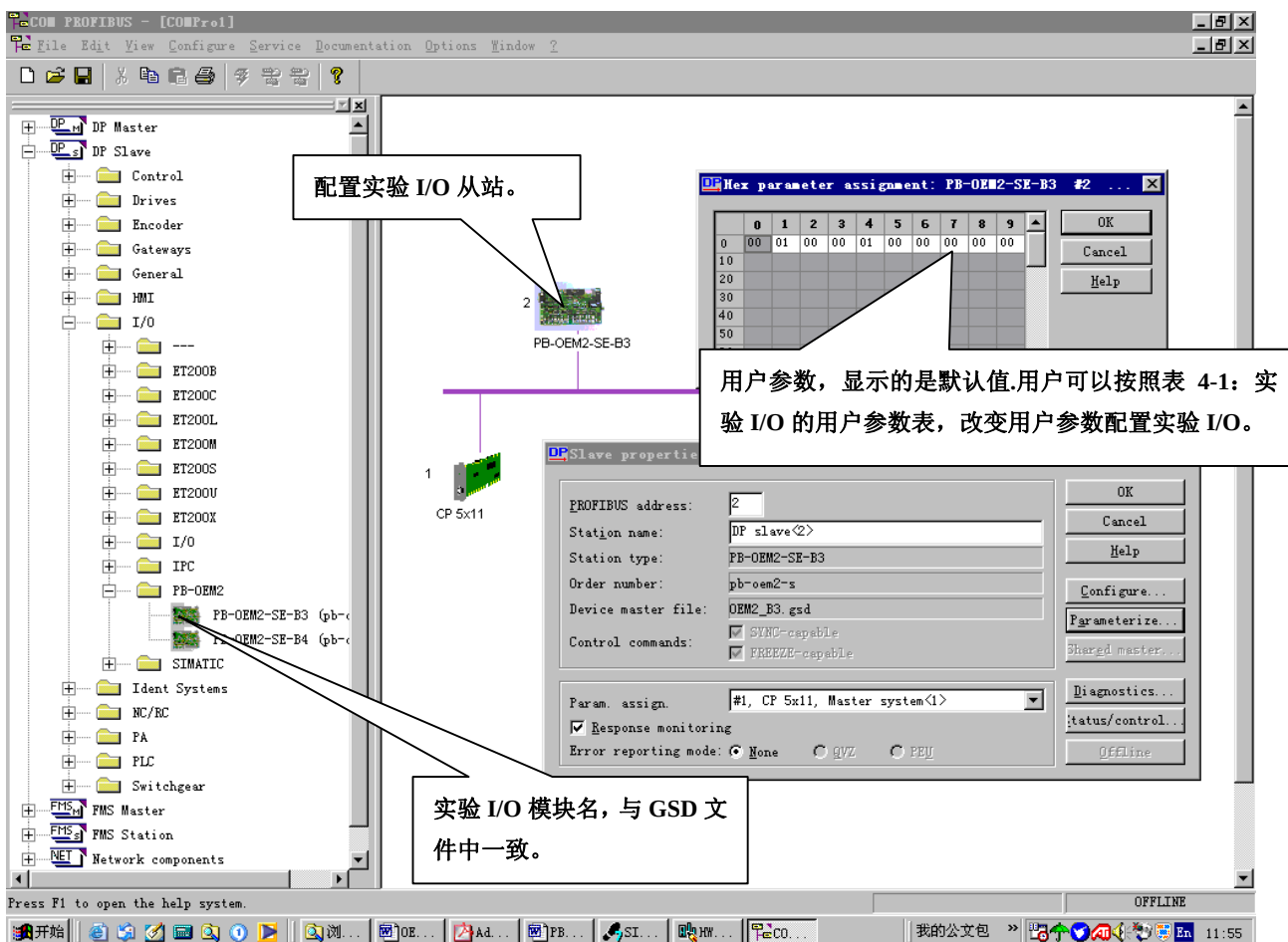


图 4-5 COM PROFIBUS V5.1—用户可按照上表改变用户参数配置实验 I/O

(6) 更便于用户使用的 GSD 文件

上述实验 I/O 的 GSD 文件，配置用户参数时需要对照用户参数表，见“图 4-4：STEP7-改变用户参数配置实验 I/O”，这对用户来说极不方便。下面是改进的实验 I/O GSD 文件,可在配置中通过菜单来配置用户参数,这种方法对用户来说比较方便，但 GSD 文件结构相对复杂。

```

;=====
; GSD 文件： OEM2-B4 实验 I/O 鼎实创新科技有限责任公司
; 产品型号： PB-OEM2-实验 I/O; 版本： 2004 年 9 月 V1.0
; 文件名： OEM2-B4.GSD
;=====
#Profibus_DP
; User-Parameter-Definition
PrmText=1
Text(0)="1--5V"
Text(1)="0--10V"
Text(2)="0--5V"
Text(3)="-10V--+10V"
Text(4)="4--20mA"
Text(5)="0--20mA"
Text(6)="0--10mA"
Text(7)="-10--+10mA"
EndPrmText
PrmText=2
Text(0)="BCD (0000~9999) "

```

```
Text(1)="无符号整型 (0~65535) "
Text(2)="有符号整型 (-32767~+32767) "
EndPrmText
PrmText=3
Text(0)="单端"
Text(1)="双端"
EndPrmText
PrmText=7
Text(0)="正逻辑(高有效)"
Text(1)="负逻辑(低有效)"
EndPrmText
PrmText=8
Text(0)="5V/DC"
Text(1)="24V/DC"
Text(2)="110V/AC"
Text(1)="220V/AC"
EndPrmText
ExtUserPrmData=1 "输入类型 AI_M: "
Unsigned8 0 0-7
Prm_Text_Ref=1
EndExtUserPrmData
ExtUserPrmData=2 "输入数据类型 AI_D: "
Unsigned8 1 0-2
Prm_Text_Ref=2
EndExtUserPrmData
ExtUserPrmData=3 "输入单端/双端 AI_CH: "
Unsigned8 0 0-1
Prm_Text_Ref=3
EndExtUserPrmData
ExtUserPrmData=4 "输出输出类型 AO_M: "
Unsigned8 0 0-5
Prm_Text_Ref=1
EndExtUserPrmData
ExtUserPrmData=5 "输出数据类型 AO_D: "
Unsigned8 1 0-2
Prm_Text_Ref=2
EndExtUserPrmData
ExtUserPrmData=6 "输出单端/双端 AO_CH: "
Unsigned8 0 0-1
Prm_Text_Ref=3
EndExtUserPrmData
ExtUserPrmData=7 "输入正/负逻辑 DI_PN:"
Unsigned8 0 0-1
Prm_Text_Ref=7
EndExtUserPrmData
ExtUserPrmData=8 "输入电压等级 DI_V:"
Unsigned8 1 0-1
Prm_Text_Ref=8
EndExtUserPrmData
ExtUserPrmData=9 "输出正/负逻辑 DO_PN:"
Unsigned8 0 0-1
Prm_Text_Ref=7
EndExtUserPrmData
ExtUserPrmData=10 "输出电压等级 DO_V:"
Unsigned8 1 0-1
```

```

Prm_Text_Ref=8
EndExtUserPrmData
; Unit-Definition-List:
GSD_Revision=2
Vendor_Name="DS FieldBus Ltd. Co." ;公司名，可按用户名修改
Model_Name="PB-OEM2-SE-B4" ;模块名称，也是组态时该产品的名称，组态软件如 STEP
7, COM PROFIBUS
Revision="V1.0 "
Ident_Number=0x06FA ;ID 号，必须与初始化报文一致
Protocol_Ident=0
Station_Type=0
Hardware_Release="A1.0 "
Software_Release="Z1.0 "
9.6_supp=1
19.2_supp=1
93.75_supp=1
187.5_supp=1
500_supp=1
45.45_supp=1
1.5M_supp=1
3M_supp=1
6M_supp=1
12M_supp=1
MaxTsdr_9.6=60
MaxTsdr_19.2=60
MaxTsdr_45.45=250
MaxTsdr_93.75=60
MaxTsdr_187.5=60
MaxTsdr_500=100
MaxTsdr_1.5M=150
MaxTsdr_3M=250
MaxTsdr_6M=450
MaxTsdr_12M=800
Implementation_Type="spc3"
Bitmap_Device="SE_B" ;图标文件，用户可以自制图标，缺省
; Slave-Specification:
OrderNumber="pb-oem2-s" ;产品序列号，可按用户名修改
Freeze_Mode_supp=1
Sync_Mode_supp=1
Auto_Baud_supp=1
Fail_safe=0
Min_Slave_Intervall=6
Slave_Family=3@TdF@PB-OEM2 ;组态中的分类名
Max_Diag_Data_Len=6
Modul_Offset=0
Modular_Station=0
Max_Input_Len=224
Max_Output_Len=224
Max_Data_Len=448
User_Prm_Data_Len=10
Max_User_Prm_Data_Len=10
User_Prm_Data = 0x00,0x01,0x00,0x00,0x01,0x00,0x00,0x01,0x00,0x01
Ext_User_Prm_Data_Ref(0)=1
Ext_User_Prm_Data_Ref(1)=2
Ext_User_Prm_Data_Ref(2)=3
Ext_User_Prm_Data_Ref(3)=4

```

```

Ext_User_Prm_Data_Ref(4)=5
Ext_User_Prm_Data_Ref(5)=6
Ext_User_Prm_Data_Ref(6)=7
Ext_User_Prm_Data_Ref(7)=8
Ext_User_Prm_Data_Ref(8)=9
Ext_User_Prm_Data_Ref(9)=10

```

; Module-Definitions:

```

Module="1AI+1AO/16DI+16DO" 0x70,0x31 ;1 word input +1 word output/2 bytes input +2 bytes
output

```

EndModule

在 STEP 7 中的配置介面：

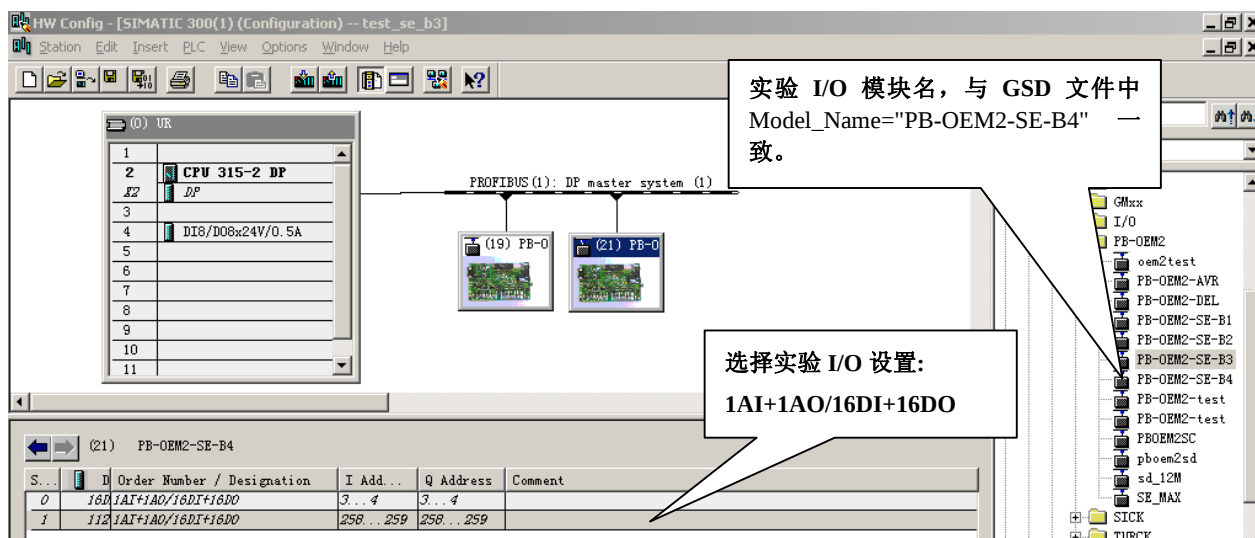


图 4-6 选择实验 I/O 设置:1AI+1AO/16DI+16DO

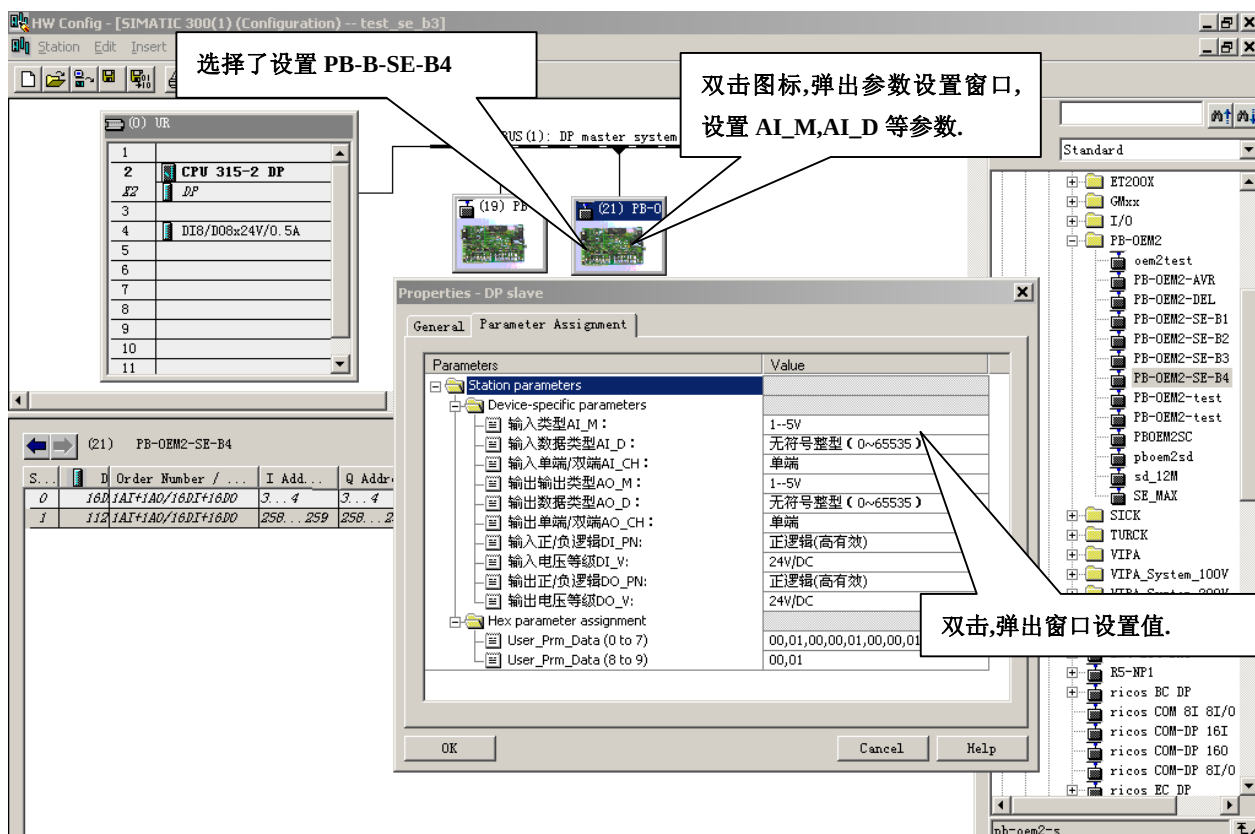


图 4-7 进一步的参数设置

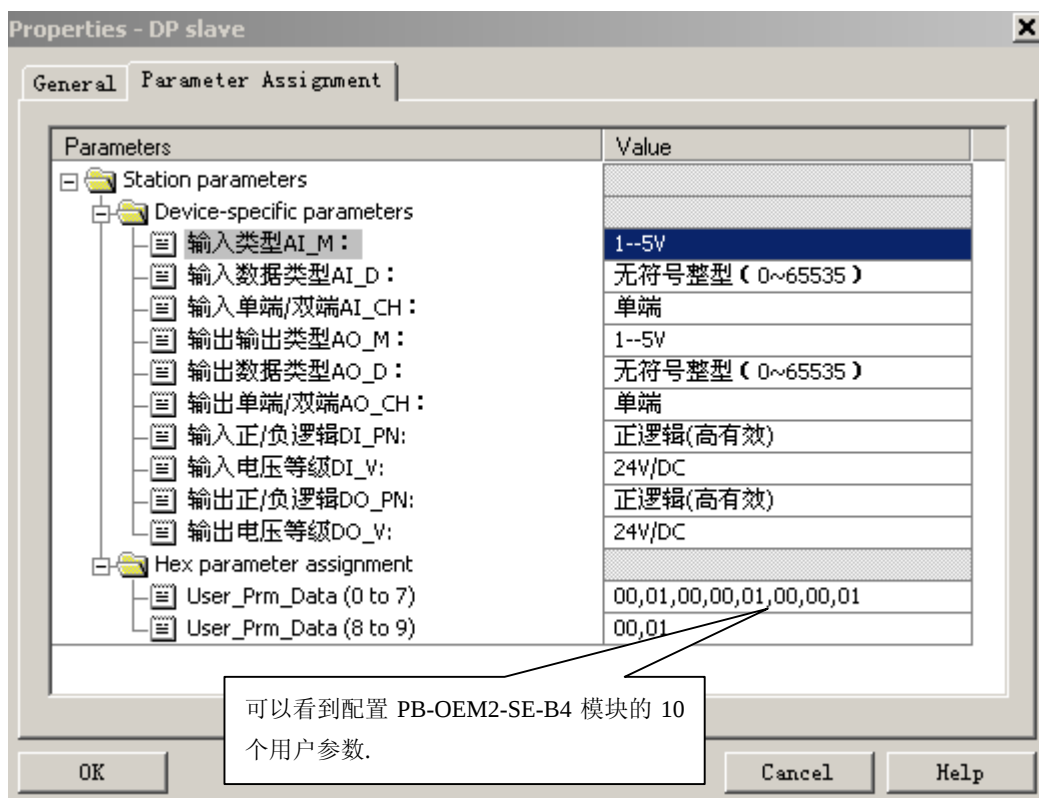


图 4-8 可以看到配置 1AI+1AO 模块的 7 个用户参数

- ① 图 4-8: 可以看到配置 PB-OEM2-SE-B4 模块的 10 个用户参数: 00, 01, 00, 00, 01, 00, 00, 01, 00, 01。
- ② 第 1~8 参数: 00, 01, 00, 00, 01, 00, 00, 01 对应上面的配置: 输入类型 AI_M: 1—5V; 输入数据类型 AI_D: 无符号整型 (0~65535); 输入单端/双端 AI_CH: 单端; 输出类型 AO_M: 1—5V; 输出数据类型 AO_D: 无符号整型 (0~65535); 输出单端/双端 AO_CH: 单端; 输入正/负逻辑 DI_PN: 正逻辑 (高有效); 输入电压等级 DI_V: 24V/DC;
- ④ 第 9~10 参数: 00, 01 对应上面的配置: 输出正/负逻辑 DO_PN: 正逻辑 (高有效)、输出电压等级 DO_V: 24V/DC。
- ⑤ 如果依照菜单改变配置, 第 1~10 用户参数也会该变。

在 COM PROFIBUS V5.1 中的配置介面:

见下页图 4-9: 在 COM PROFIBUS 中配置实验 I/O 模块后各种参数的设置。

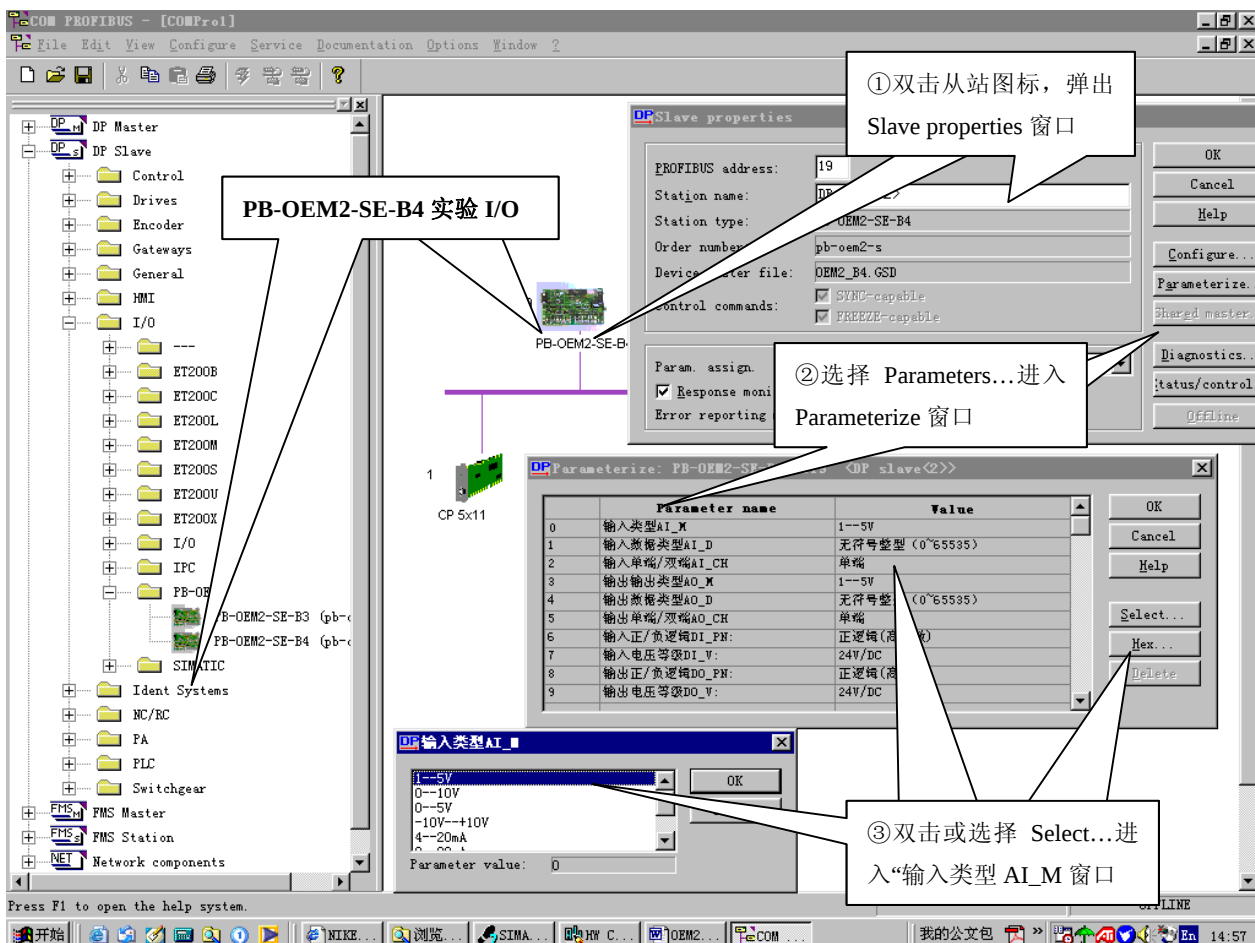


图 4-9 在 COM PROFIBUS 中配置实验 I/O 模块后各种参数的设置

(7) 带用户参数功能的 SE 接口的初始化

例 3 实验 I/O 技术指标摘要：

站号设置为 0x19，ID 号=06FA；

I/O 配置：1 Word Input + 1 Word Output + 2 Bytes Input + 2 Bytes Output

I/O 配置数据长度 CFG_LEN=2；I/O 配置数据为：0x70, 0x31

用户参数长度 User_Prm_Data_Len=10；

接收数据长 SD_in_len $\geq$ 4+1+1=6

发送数据长 SD_out_len $\geq$ 4+1+1=6；

因此，实验 I/O 的 GSD 文件中必须有：

User_Prm_Data_Len=10

Max_User_Prm_Data_Len=10

User_Prm_Data = 0x00, 0x01, 0x00, 0x00, 0x01, 0x00, 0x00, 0x01, 0x00, 0x01

Module="1AI+1AO/16DI+16DO" 0x70, 0x31

EndModule

例 3 带用户参数功能的 SE 接口的初始化报文见下页图 4-9-1、图 4-9-2、图 4-9-3

byte0	站号=0x13
byte1	ID 号高位=0x06
byte2	ID 号低位=0xfa
byte3	CFG_LEN=0x02
byte 4	I/O 配置数据 1=0x70
byte 5	I/O 配置数据 2=0x31
byte 6	不用=0
byte 7	不用=0
byte 23	
byte 24	接收数据长 SD_in_len=6
byte 25	发送数据长 SD_out_len=6
byte 26	用户参数长度 User_Prm_Data_Len=10
byte 27	***不用
byte 47	
byte 48	校验和 Σ (0) ~ (47)

图 4-9-1

例 3：用户模板发送初始化报文

byte0	测试到的波特率号 BR=1~7
byte 1	初始化报文错误号 N=0
byte 2	初始化成功标志=0xaa
byte 3	初始化成功标志=0xaa
byte 4	初始化成功标志=0xaa
byte 5	初始化成功标志=0xaa
byte 6	
byte 7	
byte 23	
byte 24	
byte 25	
byte 26	
byte 27	
byte 47	
byte 48	校验和 Σ (0) ~ (47)

图 4-9-2

例 3：接口板回答的初始化成功报文

byte0	测试到的波特率号 BR=1~7
byte 1	初始化报文错误号 N=01~06
byte 2	初始化失败标志=0x55
byte 3	初始化失败标志=0x55
byte 4	初始化失败标志=0x55
byte 5	初始化失败标志=0x55
byte 6	
byte 7	
byte 23	
byte 24	
byte 25	
byte 26	
byte 27	
byte 47	
byte 48	校验和 Σ (0) ~ (47)

图 4-9-3

例 3：接口板回答的初始化失败报文

(8) 带用户参数功能数据交换过程

为读者方便，将“表 4-2 请求数据命令 req_com”、表 4-3 报文类型及接口状态 AB_status”拷贝到图 4-10 后。

- ① 进入数据交换状态后，实验板首先发送 PROFIBUS 输入数据（例 3 为 1AI+16DI）、并请求读取 PROFIBUS 输出数据，即 req_com =0，SE 接口将回答“输出数据报文 A”，见图 4-10-1。
- ② 报文格式 A 中 AB_status (D7)=0 及包含 PROFIBUS 主站输出数据（1AO+16DO）；实验板除接收 PROFIBUS 输出数据外还要判断用户参数更新标记 AB_status (D0)=？，见图 4-10-2。
- ③ 如果 AB_status (D7)=0 说明用户参数没有更新（可能主站还没有接通），实验板继续发送 PROFIBUS 输入数据并请求 SE 接口将回答“输出数据报文 A”，见图 4-10-3。
- ④ 如果 AB_status (D7)=1 说明用户参数已更新（主站下载了用户参数），见图 4-10-2。

⑤ 实验板将发送 PROFIBUS 输入数据并请求 SE 接口将回答“输出数据报文 B”，即 req_com =1。见图 4-10-4。

⑥ SE 接口板将回答“输出数据报文 B”，包括用户参数数据：AB_status (D7)=1、AB_status (D0)=1 见图 4-10-4、图 4-10-5。

⑦ 实验板得到用户参数（例 3 为 10 个字节）后可在实验板设置参数，并继续发送 PROFIBUS 输入数据并请求 SE 接口回答“输出数据报文 A”，且令 req_com =0x20，清除用户参数更新标记 AB_status (D0)=1。

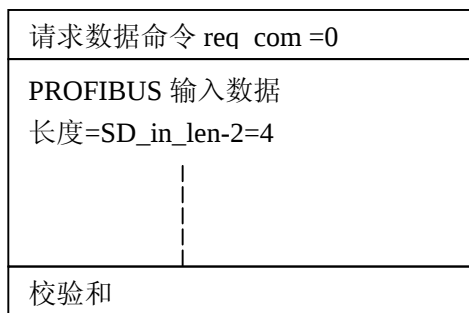


图 4-10-1: 使用用户参数功能条件下，用户板首先发请求数据命令 req_com 0

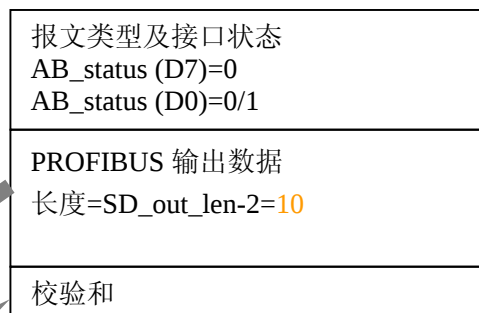


图 4-10-2: 接口板回答输出数据报文格式 A，且 AB_status (D7)=0

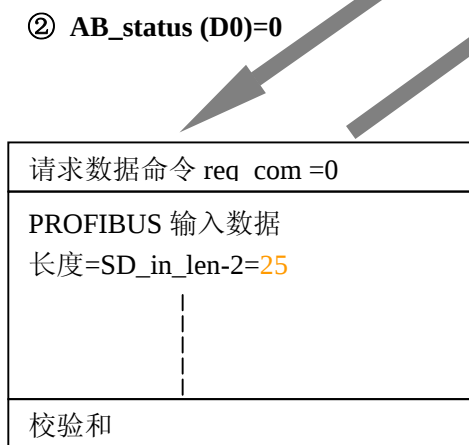


图 4-10-3: 用户板判断 AB_status (D0)=0 继续发 req_com =0，接收格式 A

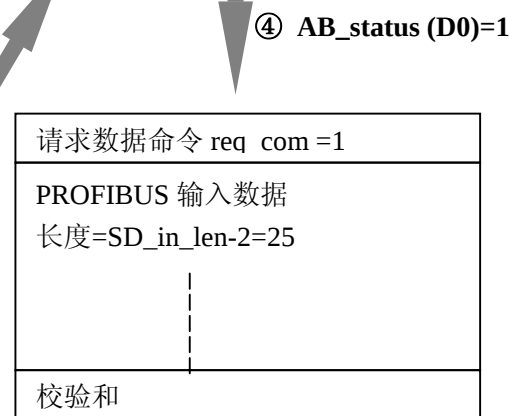


图 4-10-4: 用户板判断 AB_status (D0)=1 发 req_com =1，接收格式 B

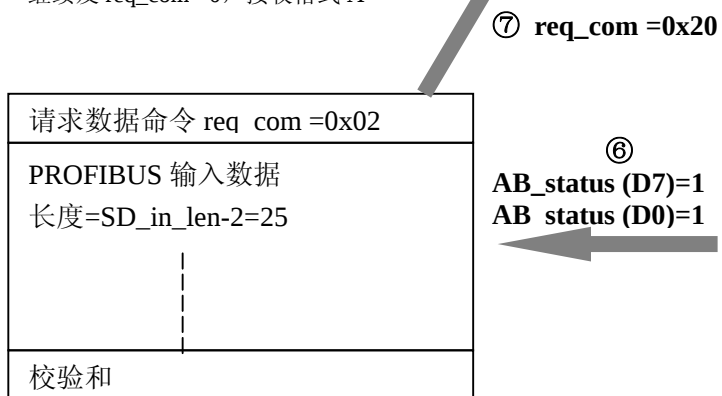


图 4-10-6: 用户板发 req_com =0x20，清除 AB_status (D0)=1，准备接收格式 A

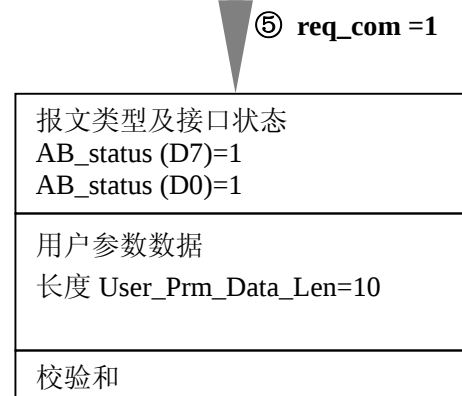


图 4-10-5: 接口板回答报文格式 B, AB_status (D7)=1 且 AB_status (D7)保持 1。

表 4-2 请求数据命令 req_com

请求数据命令 req_com	说 明
D7~D2	不用
D1	=0: 无效
	=1: 清接口板用户参数已由主站更新标记 AB_status (D0)
D0	=0: 请求接口回答 PROFIBUS 输出数据
	=1: 请求接口回答 PROFIBUS 主站更新的用户参数数据

表 4-3 报文类型及接口状态 AB_status

报文类型及接口状态 AB_status	说 明
D7	=0: A 格式, 本报文回答 PROFIBUS 输出数据, 长度= SD_out_len-2
	=1: B 格式, 本报文回答用户参数数据, 长度 User_Prm_Data_Len
D6~D3	不用
D2~D1	=00: 接口板接收用户板报文无错
	=01: 接口板接收用户板报文字符有偶校验错
	=10: 接口板接收用户板报文有校验和错
	=11: 不用
D0	=0: 接口板还没有接收到主站发送的用户参数、或主站没有更新用户参数
	=1: 接口板用户参数数据已由主站更新;

(9) 实验板初始化 SE 接口板的过程框图

例 3 实验板初始化 SE 接口板的过程框图与例 2 完全相同, 见图 4-2-4: 用户模板初始化接口模板的过程。

(10) 数据交换状态下实验板工作流程

因为例 3 有用户参数功能, 因此实验板数据交换状态下工作流程与例 1、例 2 不同。见下页图 4-11: 数据交换状态下用户模板的程序流程 (例 3)。

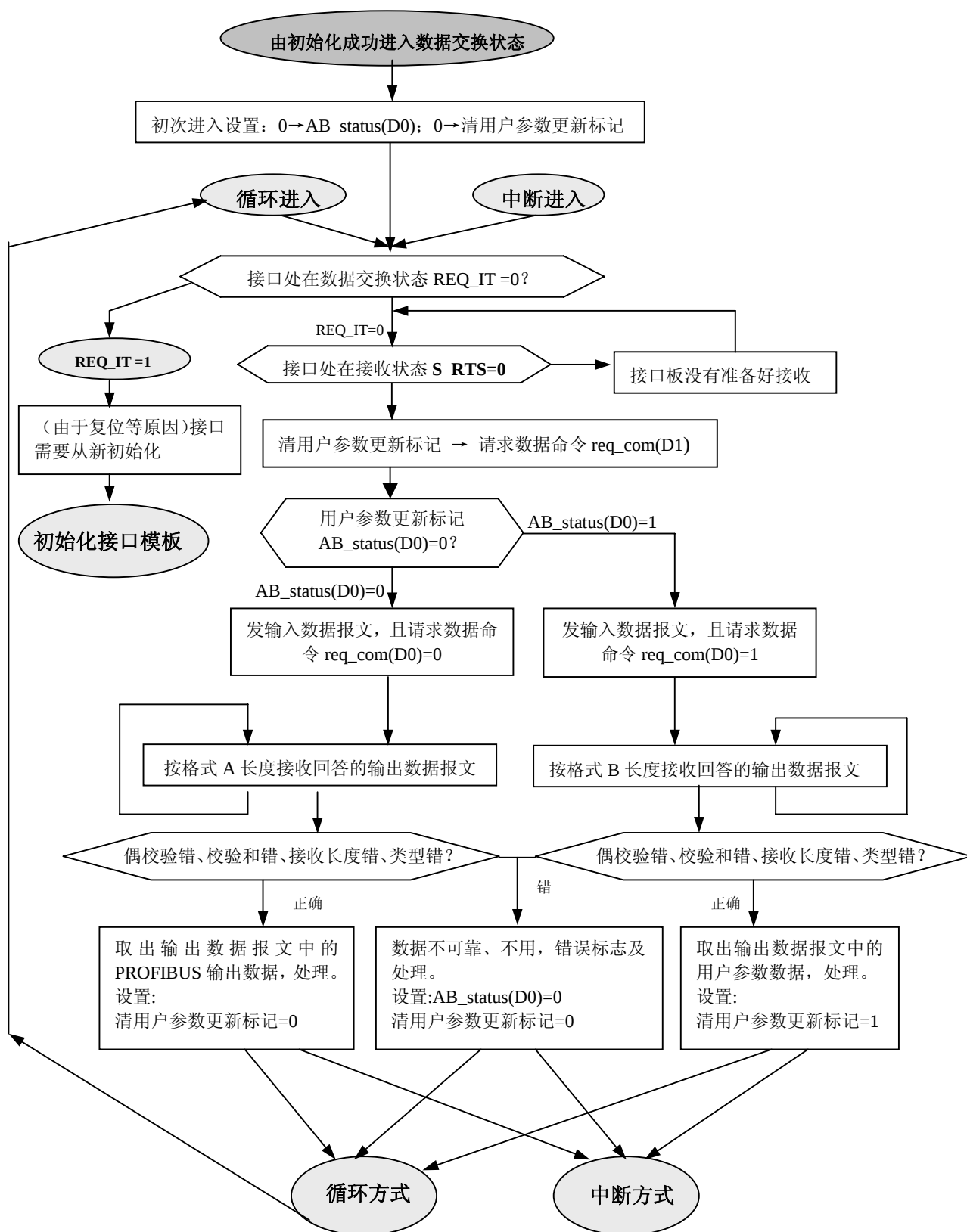


图 4-11 数据交换状态下用户模板的程序流程 (例 3)

(1) 例 3 实验 I/O 的程序清单

以下凡斜体部分与例 2 程序相同，蓝色正体部分为例 3 特殊部分。

```

/*+-----+
|文件名称: 实验板串口通信程序—例3|
|版 本: V1.1|
|制作单位: 北京鼎实创新科技公司|
|版权许可:|
|网址: www.c-profibus.com.cn|
|email: tangjy@c-profibus.com.cn|
|日期: 2004/9|
+-----+*/

#include <c:\51\c51\INC\reg52.h>
#define UBYTE unsigned char
#define UWORD unsigned int
/*=====*/
/* 以下是输入键、指示灯等设备变量定义，用户可以越过阅读 */
/*=====*/
sbit out0_7=P2^0; /* 定义 out0_7=P2.0=P16O0,输出指示灯 Q0-Q7 的输出控制,见实验板原理图 U16 -----*/
sbit out8_15=P2^1; /* 定义 out8_15=P2.1=P17O1,输出指示灯 Q8-Q15 的输出控制,见实验板原理图 U17-----*/
sbit inled0_7=P2^2; /* 定义 inled0_7=P2.2=P18IO0,输入指示灯 I0-I7 的输出控制,见实验板原理图 U18-----*/
sbit inled8_15=P2^3; /* 定义 inled8_15=P2.3=P19IO1,输入指示灯 I8-I15 的输出控制,见实验板原理图 U19-----*/
sbit in0_7=P2^4; /* 定义 in0_7=P2.4=P2.4=P12IO,输入键 I0-I7 的读控制,见实验板原理图 U12 -----*/
sbit in8_15=P2^5; /* 定义 in8_15=P2.5=P2.5=P13I1,输入键 I8-I15 的读控制,见实验板原理图 U13 -----*/
sbit inadd=P2^6; /* 定义 inadd=P2.6=P2.6=P20AD, PROFIBUS 地址输入 AD0-AD7 的读控制 -----*/
/* 见实验板原理图 U20 -----*/
sbit REQ_IT=P1^7; /* 定义 REQ_IT=P1^7=P1.7=P17S_CTS,输入,连接 OEM2-SE-REQ_IT -----*/
/* 见实验板原理图 CONNECT-16 -----*/
sbit S_RTS=P1^6; /* 定义 S_RTS=P1^6=P1.6=P17S_RTS,输入,连接 OEM2-SE-S_RTS -----*/
/* 见实验板原理图 CONNECT-16 -----*/
sbit P15RES=P1^5;

/***** 定义: di0_7: 字节型,是输入键 I0-I7 当前值; *****/
/***** 定义: di8_15: 字节型,是输入键 I8-I15 当前值; *****/
bdata UBYTE di0_7,di8_15; /* di0_7: 字节型,输入键 I0-I7;di8_15: 字节型,输入键 I8-I15 -----*/
sbit di0=di0_7^0; /* 定义 di0=di0_7^0= 输入键 I0 -----*/
sbit di1=di0_7^1; /* 定义 di1=di0_7^1= 输入键 I1 -----*/
sbit di2=di0_7^2; /* 定义 di2=di0_7^2= 输入键 I2 -----*/
sbit di3=di0_7^3; /* 定义 di3=di0_7^3= 输入键 I3 -----*/
sbit di4=di0_7^4; /* 定义 di4=di0_7^4= 输入键 I4 -----*/
sbit di5=di0_7^5; /* 定义 di5=di0_7^5= 输入键 I5 -----*/
sbit di6=di0_7^6; /* 定义 di6=di0_7^6= 输入键 I6 -----*/
sbit di7=di0_7^7; /* 定义 di7=di0_7^7= 输入键 I7 -----*/
sbit di8=di8_15^0; /* 定义 di8=di8_15^0= 输入键 I8 -----*/
sbit di9=di8_15^1; /* 定义 di9=di8_15^1= 输入键 I9 -----*/
sbit di10=di8_15^2; /* 定义 di10=di8_15^2= 输入键 I10 -----*/
sbit di11=di8_15^3; /* 定义 di11=di8_15^3= 输入键 I11 -----*/
sbit di12=di8_15^4; /* 定义 di12=di8_15^4= 输入键 I12 -----*/
sbit di13=di8_15^5; /* 定义 di13=di8_15^5= 输入键 I13 -----*/
sbit di14=di8_15^6; /* 定义 di14=di8_15^6= 输入键 I14 -----*/
sbit di15=di8_15^7; /* 定义 di15=di8_15^7= 输入键 I15 -----*/

/***** 定义: key0_7: 字节型,是输入键 I0-I7 扫描读入值; *****/
/***** 定义: key8_15: 字节型,是输入键 I8-I15 扫描读入值; *****/
bdata UBYTE key0_7,key8_15;
sbit key0=key0_7^0; /* 定义 key0=key0_7^0= 输入键 I0 -----*/
sbit key1=key0_7^1; /* 定义 key1=key0_7^1= 输入键 I1 -----*/
sbit key2=key0_7^2; /* 定义 key2=key0_7^2= 输入键 I2 -----*/
sbit key3=key0_7^3; /* 定义 key3=key0_7^3= 输入键 I3 -----*/
sbit key4=key0_7^4; /* 定义 key4=key0_7^4= 输入键 I4 -----*/
sbit key5=key0_7^5; /* 定义 key5=key0_7^5= 输入键 I5 -----*/
sbit key6=key0_7^6; /* 定义 key6=key0_7^6= 输入键 I6 -----*/
sbit key7=key0_7^7; /* 定义 key7=key0_7^7= 输入键 I7 -----*/
sbit key8=key8_15^0; /* 定义 key8=key8_15^0= 输入键 I8 -----*/
sbit key9=key8_15^1; /* 定义 key9=key8_15^1= 输入键 I9 -----*/

```

```

sbit key10=key8_15^2;      /* 定义key10=key8_15^2=输入键I10          -----*/
sbit key11=key8_15^3;      /* 定义key11=key8_15^3=输入键I11          -----*/
sbit key12=key8_15^4;      /* 定义key12=key8_15^4=输入键I12          -----*/
sbit key13=key8_15^5;      /* 定义key13=key8_15^5=输入键I13          -----*/
sbit key14=key8_15^6;      /* 定义key14=key8_15^6=输入键I14          -----*/
sbit key15=key8_15^7;      /* 定义key15=key8_15^7=输入键I15          -----*/

/***** 定义: P00-P07 对应 P0 口的 P0.0-P0.7 *****/
sbit P00=P0^0;
sbit P01=P0^1;
sbit P02=P0^2;
sbit P03=P0^3;
sbit P04=P0^4;
sbit P05=P0^5;
sbit P06=P0^6;
sbit P07=P0^7;

/***** PP00-PP07 是上面 P00-P07 的中间变量 *****/
bdata UBYTE PP0;
sbit PP0=PP0^0;
sbit PP1=PP0^1;
sbit PP2=PP0^2;
sbit PP3=PP0^3;
sbit PP4=PP0^4;
sbit PP5=PP0^5;
sbit PP6=PP0^6;
sbit PP7=PP0^7;

/**** 定义: AB_status: 字节型, 报文类型及接口状态, 见《PB-OEM2-SE 产品手册》第四章、三、数据交换 *****/
bdata UBYTE AB_status;
sbit AB_sD0=AB_status^0;    /***** AB_sD0=AB_status^0: 用户参数更新标记 *****/
sbit AB_sD7=AB_status^7;    /***** AB_sD7=AB_status^7: 报文格式 *****/

/**** 定义: req_com: 字节型, 请求数据命令, 见《PB-OEM2-SE 产品手册》第四章、三、数据交换 *****/
bdata UBYTE req_com;
sbit req_comD0=req_com^0;    /***** req_comD0=req_com^0: 请求数据/参数命令 *****/
sbit req_comD1=req_com^1;    /***** req_comD1=req_com^1: 清参数命令 *****/

/*-----*/
/*          输入键、指示灯等设备变量定义完毕          */
/*=====*/

/*=====*/
/*          以下是有关输入键、指示灯控制子程序, 用户了解功能即可          */
/*-----*/

/***** 将 V 值送指示灯 Q0-Q7 输出 *****/
void steout0_7(UBYTE v)
{
    out0_7=0;
    P0=v;
    P0=v;
    P0=v;
    P0=v;
    out0_7=1;
    out0_7=0;
}

/***** 将 V 值送指示灯 Q8-Q15 输出 *****/
void steout8_15(UBYTE v)
{
    out8_15=0;
    P0=v;
    P0=v;
    P0=v;
    P0=v;
    out8_15=1;
}

```

```

out8_15=0;
}

/***** 将V 值送输入键指示灯ILED0-LIED7 输出*****/
void steinled0_7(UBYTE v)
{
inled0_7=0;
P0=v;
P0=v;
P0=v;
P0=v;
inled0_7=1;
inled0_7=0;
}

/***** 将V 值送输入键指示灯ILED8-LIED15 输出*****/
void steinled8_15(UBYTE v)
{
inled8_15=0;
P0=v;
P0=v;
P0=v;
P0=v;
inled8_15=1;
inled8_15=0;
}

/***** 读 PROFIBUS 地址开关值*****/
UBYTE rd_address()
{
UBYTE data x;
P0=0xff;
inadd=0;
x=P0;
x=P0;
x=P0;
x=P0;
inadd=1;
return(x);
}

/***** 扫描一次输入键 I0-I15，更新一次输入键当前值*****/
/***** 对输入键 I0-I15 有 0-1 者，当前值取反*****/
void scanin()
{
P0=0xff;
in0_7=0;
PP0=P0;
PP0=P0;
PP0=P0;
PP0=P0;
if (key0_7!=PP0)
{
if ((PP0==1)&&(key0==0)) { di0=~di0;};
if ((PP01==1)&&(key1==0)) { di1=~di1;};
if ((PP02==1)&&(key2==0)) { di2=~di2;};
if ((PP03==1)&&(key3==0)) { di3=~di3;};
if ((PP04==1)&&(key4==0)) { di4=~di4;};
if ((PP05==1)&&(key5==0)) { di5=~di5;};
if ((PP06==1)&&(key6==0)) { di6=~di6;};
if ((PP07==1)&&(key7==0)) { di7=~di7;};
key0_7=PP0;
};
in0_7=1;
inled0_7=0;
}

```

```

P0=di0_7;
P0=di0_7;
P0=di0_7;
P0=di0_7;
inled0_7=1;
inled0_7=0;

P0=0xff;
in8_15=0;
PP0=P0;
PP0=P0;
PP0=P0;
PP0=P0;
PP0=P0;
PP0=P0;
if (key8_15!=PP0)
{
    if ((PP0==1)&&(key8==0)) { di8=~di8;};
    if ((PP01==1)&&(key9==0)) { di9=~di9;};
    if ((PP02==1)&&(key10==0)) { di10=~di10;};
    if ((PP03==1)&&(key11==0)) { di11=~di11;};
    if ((PP04==1)&&(key12==0)) { di12=~di12;};
    if ((PP05==1)&&(key13==0)) { di13=~di13;};
    if ((PP06==1)&&(key14==0)) { di14=~di14;};
    if ((PP07==1)&&(key15==0)) { di15=~di15;};
    key8_15=PP0;
};
in8_15=1;
inled8_15=0;
P0=di8_15;
P0=di8_15;
P0=di8_15;
P0=di8_15;
P0=di8_15;
inled8_15=1;
inled8_15=0;
}
/***** 按照用户参数设置 I/O 通道 *****/
void Config(UBYTE *pp)
{
    /*----pp 是 user_prm[10] 指针，本子程序是按照用户参数 user_prm[10] 设置 I/O 通道----*/
}
/*=====*/
/*                      主程序                      */
/*-----*/

void main ()
{
    UBYTE data i,j,k,x,y,count,frist;
    UWORD data nw,nn;
    UBYTE xdata * data p;
    UBYTE data address; /*-----从站地址 -----*/
    UBYTE data re_er; /*-----字符偶校验错 -----*/
    UBYTE data br_num; /*-----串口波特率号 -----*/
    UBYTE data error; /*-----校验和错 -----*/
    UBYTE xdata rebox[113]; /*-----串口接收缓冲 -----*/
    UBYTE xdata trbox[113]; /*-----串口发送缓冲 -----*/
    UBYTE xdata pb_out[112]; /*-----PROFIBUS 输出数据区 -----*/
    UBYTE xdata pb_in[112]; /*-----PROFIBUS 输入数据区 -----*/
    UBYTE xdata user_prm[10]; /*-----PROFIBUS 用户参数区 -----*/
    UBYTE data data_in_len; /*-----数据交换输入（实验板发送）数据长度 -----*/
    UBYTE data dil; /*-----不包括校验和的 data_in_len 长度 -----*/
    UBYTE data data_out_len; /*-----数据交换输出（实验板接收）数据长度 -----*/
    UBYTE data dol; /*-----不包括校验和的 data_out_len 长度 -----*/
    UBYTE data user_prm_len; /*-----用户参数长度 -----*/
    UBYTE data se_er; /*-----OEM2-SE 接收数据错误类型号 -----*/

    P1=0xff;
    P0=0;
    P2=0xf0;

```

```

di0_7=di8_15=0;

steout0_7(0);          /*-----输出指示灯 Q0-Q7 清零          -----*/
steout8_15(0);         /*-----输出指示灯 Q8-Q15 清零         -----*/
steinled0_7(0);        /*-----输入键指示灯 ILED0-ILED7 清零        -----*/
steinled8_15(0);       /*-----输入键指示灯 ILED8-ILED15 清零       -----*/

/*=====*/
/*  以下是实验板上电后，检测 P89V51RD2 单片机内部 RAM 的一段输出演示程序；          */
/*  这段程序与 PROFIBUS 接口板 SE 通信无关，用户可跳过此段程序。                      */
/*-----*/

p=0;
x=1;
k=1;
for (j=0;j<32;j++)
{
    for (i=0;i<24;i++)
    {
        *(p+i)=0xaa;
    };
    y=0;
    for (i=0;i<24;i++)
    {
        count=*(p+i);
        if (count!=0xaa) { y=1; };
    };
    if (y==0)
    {
        switch (k)
        {
            case 1:{ steout0_7(x);};break;
            case 2:{ steout8_15(x);};break;
            case 3:{ steinled0_7(x);};break;
            case 4:{ steinled8_15(x);};break;
            default:break;
        };
        if (x<255) {x=x*2+1;}else{x=1;k++;};
    }
    else
    {
        for (;;)
        {
            for (nw=0;nw<6000;nw++)
            {
                steout0_7(0);
                steout8_15(0);
                steinled0_7(0);
                steinled8_15(0);
            };
            for (nw=0;nw<6000;nw++)
            {
                steout0_7(0xff);
                steout8_15(0xff);
                steinled0_7(0xff);
                steinled8_15(0xff);
            };
        };
        p=p+24;
        for (nw=0;nw<8000;nw++) { };
    };

/*-----*/
/*  实验板输出演示程序结束。          */
/*-----*/

```

```

steout0_7(0);          /*-----输出指示灯 Q0-Q7 清零          -----*/
steout8_15(0);         /*-----输出指示灯 Q8-Q15 清零         -----*/
steinled0_7(0);        /*-----输入键指示灯 ILED0-ILED7 清零      -----*/
steinled8_15(0);       /*-----输入键指示灯 ILED8-ILED15 清零     -----*/

/*=====*/
/* 清串口输入缓冲区 trbox[i], 清串口输出缓冲区 rebox[i]          */
/*-----*/
for (i=0;i<113;i++)
{
    rebox[i]=trbox[i]=0;
};
for (i=0;i<112;i++)
{
    pb_in[i]=pb_out[i]=0;
};
address=rd_address(); /*-----读实验板上拨码开关, 作为站号-----*/
/*=====*/
/*                      以下是串口初始化程序                      */
/*-----*/

br_num=5;              /*-----实验板波特率选择 5, 115.2K-----*/

switch (br_num)        /*-----根据波特率号设置 T2-----*/
{
    case 1:{ TH2=0x0ff; TL2=0x0a0;RCAP2H=0x0ff;RCAP2L=0x0a0;};break; /*---9.6K---*/
    case 2:{ TH2=0x0ff; TL2=0x0d0;RCAP2H=0x0ff;RCAP2L=0x0d0;};break; /*---19.2K--*/
    case 3:{ TH2=0x0ff; TL2=0x0e8;RCAP2H=0x0ff;RCAP2L=0x0e8;};break; /*---38.4K--*/
    case 4:{ TH2=0x0ff; TL2=0x0f0;RCAP2H=0x0ff;RCAP2L=0x0f0;};break; /*---57.6K--*/
    case 5:{ TH2=0x0ff; TL2=0x0f8;RCAP2H=0x0ff;RCAP2L=0x0f8;};break; /*---115.2K-*/
    case 6:{ TH2=0x0ff; TL2=0x0fc;RCAP2H=0x0ff;RCAP2L=0x0fc;};break; /*---230.4K-*/
    case 7:{ TH2=0x0ff; TL2=0x0fe;RCAP2H=0x0ff;RCAP2L=0x0fe;};break; /*---460.8K-*/
    default:break;
};

/*-----*/
/* 注意: 本例基于 PHILIP P89V51RD2 单片机:          */
/* 波特率=14745600/16/ (65536-T)          */
/* 各种波特率 9600: T=0xffa0, 19.2K: T=0xffd0, 38.4K: T=0xffe8, */
/* 57.6K: T=0xfff0, 115.2K: T=0xfffb, 230.4k: T=0xfffc, 460.8k: T=0xfffe */
/*-----*/

PCON=0;
T2CON=0x34;          /*---RCLK=1; TCLK=1; TR2=1; C/T2=0 -----*/

/*-----*/
/* 串口设置成模式 3 即: 1 个起始位、8 个数据位、1 个偶校验位、1 个停止位; */
/* 0xd0=11010000B; SM0SM1=11; SM2=0; REN=1; TB8=0; RB8=0; TI=0; RI=0 */
/*-----*/

SCON=0x0d0;

/*=====*/
/* 形成接口板的初始化报文。          */
/*-----*/
while (REQ_IT==1)
{
    /*=====*/
    /* 形成接口板的初始化报文          */
    /* rebox[]是串口发送数组, 格式见《SE 手册》          */
    /*-----*/
    data_in_len=4+1+1;          /*--- 输入数据报文长度=4+1+1=6; -----*/
    user_prm_len=10;           /*--- 用户参数长度=10 -----*/
    data_out_len=4+1+1+1;       /*--- 输入数据报文长度=4+1+1=6 ; -----*/
    trbox[0]=address;          /*--- BYTE0=从站地址 -----*/
}

```

```

trbox[1]=0x06;          /*---- BYTE1=ID 号高位          -----*/
trbox[2]=0x0fa;        /*---- BYTE2=ID 号低位          -----*/

/*-----*/
/*          以下形成 I/O 配置数据          */
/* 本例: I/O 配置数据长度=2; I/O 配置数据: 0x31,0x70          */
/* 共: 2 byte in/2 byte out+ 1 word in/1 word out          */
/*-----*/

trbox[3]=2;             /*----BYTE3=I/O 配置数据长度=2          -----*/
trbox[4]=0x31;          /*----BYTE4=I/O 配置数据: 0x31=2 byte in/2 byte out -----*/
trbox[5]=0x70;          /*----BYTE5=I/O 配置数据: 0x70=1 word in/1 word out -----*/

trbox[24]=data_in_len;  /*----BYTE24=data_in_len 输入数据报文长度 -----*/
trbox[25]=data_out_len; /*----BYTE25=data_out_len 输出数据报文长度 -----*/
trbox[26]=user_prm_len; /*----BYTE26=user_prm_len 用户参数长度 -----*/

x=0;
for (i=0;i<48;i++)
{
    x=x+trbox[i];      /*---- 求校验和          -----*/
};
trbox[48]=x;          /*---- 置校验和          -----*/

/*-----*/
/*          发送接口板的初始化报文、接收接口板的回答, 初始不成功继续发初始化报文          */
/*-----*/

while (S_RTS==1) {};   /*---- P1.6=S_RTS=1: SE 没有准备好接收 -----*/

y=0;                  /*---- 清初始化成功标志 -----*/
while (y==0)
{
    REN=0;
    for (i=0;i<49;i++) /*---- 发送一次初始化报文共 49 字节 -----*/
    {
        TI=0;
        ACC=trbox[i];
        TB8=P;         /*---- 形成偶校验位          -----*/
        SBUF=trbox[i];
        while (TI==0) {};
    };

    while (S_RTS==0) { }; /*---- SE 没有接收完毕 -----*/

    /*----- 发送完毕、等待接收回答 -----*/
    RI=0;
    REN=1;             /*---- 允许接收          -----*/
    re_er=0;           /*---- re_er 偶校验错标志 -----*/
    for (j=0;j<49;j++) /*---- 接受回答报文共 49 个字节 -----*/
    {
        nw=0;
        while ((RI==0)&&(S_RTS==1)) {}; /*---- SE 正在发送、实验板正在接收一个字符 -----*/
        if (RI==1)
        { /*---- RI=1: 实验板接收到一个字符 -----*/
            ACC=SBUF;
            if (P!=RB8){ re_er=1;}; /*---- 偶校验错, 置偶校验错标志 -----*/
            rebox[j]=SBUF; /*---- 接收一个字符送串口输入缓冲区 -----*/
            RI=0;
        }
        else
        { /*---- (RI==0)&&(S_RTS==0):SE 报文全部发送完毕而实验板还没有接收到预期的长度 -----*/
            j=100; /*---- 退出等待接收 -----*/
        }
    }
};

```

```

/*----- 接收回答报文结束 -----*/
/*----- 以下分析接口板 SE 的回答报文 -----*/
if (j==49)
{ /*---- (j==49): 实验板接收回答报文长度正确 -----*/
  if (re_er==0) /*---- 没有字符偶校验错? -----*/
  {
    x=0;
    for (i=0;i<48;i++) /*---- 求校验和 -----*/
    {
      x=x+rebox[i];
    };
    if (x==rebox[48])
    { /*---- 校验和正确? -----*/
      if (rebox[0]==br_num) /*---- 返回的波特率号是 5? -----*/
      {
        if (rebox[1]==0) /*---- 初始化报文错误号 N=0? -----*/
        {
          x=0;
          for (i=2;i<48;i++) /*---- byte2~byte47 都是 0xaa? -----*/
          {
            if (rebox[i]!=0xaa)
            {
              x=1;
            };
          };
          if (x==0)
          {
            y=1; /*---- byte2~byte47 都是 0xaa, 初始化成功,y=1,准备退出 while (y==0) ----*/
          }
          else
          {
            {x=0x11;}; /*---- byte2~byte47 不都是 0xaa ! 错误标记=0x11; ----*/
          }
          else
          {
            {x=rebox[1];}; /*---- 初始化报文错误号 N<>0, 错误标记= 返回错误号 ----*/
          }
          else
          {
            {x=0x77;}; /*---- 返回的波特率号不是 5, 错误标记=0x77; ----*/
          }
          else
          {
            {x=0xff;}; /*---- 校验和不正确, 错误标记=0xff; ----*/
          }
          else
          {
            {x=0xee;}; /*---- 发生了字符偶校验错 错误标记=0xee; ----*/
          }
        }
      }
      else
      {
        x=0xcc; /*---- j!=49 发生了接收报文长度错 错误标记=0xee; ----*/
      };
    };

    if (y==0)
    { /*---- 初始化失败 ----*/
      for (nw=0;nw<10000;nw++)
      {
        steout0_7(br_num); /*---- 显示返回波特率号 ----*/
        steout8_15(x); /*---- 显示返回错误号 ----*/
      };
    };

    }; /*---- 如果 y==0, 返回 while (y==0) ----*/

    /*=====
    ==*/
    /*          实验板对 PROFIBUS 接口板初始化成功, 进数据交换状态的无限循环中 */
    /*          trbox[]/rebox[] 是串口发送/接收数组, 发送/接收格式见《SE 手册》 */
    /*-----*/
    dil=data_in_len-1; /*---- data_in_len=50, dil=49 ----*/

```

```

dol=data_out_len-1;          /*---- data_out_len==50, dol==49          ----*/
for (i=0;i<data_in_len;i++)
{
    trbox[i]=0;              /*---- 清发送、接收缓冲区          ----*/
};
for (i=0;i<data_out_len;i++)
{
    rebox[i]=0;              /*---- 清发送、接收缓冲区          ----*/
};

/*=====在与主站连通得到用户参数之前实验 I/O 采用以下默认配置=====*/
user_prm[0]=0x00;            /*---- AI_M: 1-5V          ----*/
user_prm[1]=0x01;            /*---- AI_D: 无符号整型 (0~65535)          ----*/
user_prm[2]=0x00;            /*---- 输入单端/双端 AI_CH: 单端          ----*/
user_prm[3]=0x00;            /*---- 输出类型 AO_M: 1-5V          ----*/
user_prm[4]=0x01;            /*---- 输出数据类型 AO_D: 无符号整型 (0~65535)          ----*/
user_prm[5]=0x00;            /*---- 输出单端/双端 AO_CH: 单端          ----*/
user_prm[6]=0x00;            /*---- 输入正/负逻辑 DI_PN: 正逻辑 (高有效)          ----*/
user_prm[7]=0x01;            /*---- 输入电压等级 DI_V: 24V/DC          ----*/
user_prm[8]=0x00;            /*---- 输出正/负逻辑 DO_PN: 正逻辑 (高有效)          ----*/
user_prm[9]=0x01;            /*---- 输入电压等级 DO_V: 24V/DC          ----*/
p=&user_prm[0];
Config(p);                   /*---- 按照用户参数设置 I/O 通道          ----*/

frist=1;                     /*----首次进入标记          ----*/

/*===== 进数据交换状态的无限循环中 =====*/
while (REQ_IT==0)
{ /*---- (REQ_IT==0):SE 处在数据交换状态  ----*/
    if (frist==1)
    {
        req_com=0;           /*----首次进入, 请求 PROFIBUS 数据=A 格式报文          ----*/
        frist=0;             /*----清首次进入标记          ----*/
    };
    trbox[0]=req_com;         /*----请求数据命令 req_com          ----*/
    trbox[1]=di0_7;           /*----将键 I0_I7 作为 PROFIBUS 输入第 1 字节          ----*/
    trbox[2]=di8_15;          /*----将键 I8_I15 作为 PROFIBUS 输入第 2 字节          ----*/
    trbox[3]=di16_31;         /*----将键 I16_I31 作为 PROFIBUS 输入第 3 字节          ----*/
    trbox[4]=di32_47;         /*----将键 I32_I47 作为 PROFIBUS 输入第 4 字节          ----*/
    x=0;
    for (i=0;i<dil;i++)       /* 求校验和          ----*/
    {
        x=x+trbox[i];
    };
    trbox[dil]=x;             /* 置校验和          ----*/
    REN=0;

    while (S_RTS==1) {};      /*---- S_RTS==1: 接口板没有准备好接收数据          ----*/

    for (i=0;i<data_in_len;i++) /*---- 发送数据交换报文共 6 字节          ----*/
    {
        TI=0;
        ACC=trbox[i];
        TB8=P;                /*---- 形成偶校验位          ----*/
        SBUF=trbox[i];
        while (TI==0) {};
    };

    while (S_RTS==0) {};      /*---- SE 没有接收完毕          ----*/

    /*=====发送数据交换报文结束, 等待接收接口板 SE 的回答。=====*/
    re_er=0;                  /*----- re_er 偶校验错标志          ----*/
    if (req_comD0==0)
    {
        y=data_out_len;
    }
}

```

```

else
{
    y=user_prm_len+2;
};
RI=0;
REN=1; /*----- 允许接收 -----*/
for (j=0;j<y;j++) /*----- 等待接收接口板的回答数据交换报文共 50 字节 -----*/
{
    while ((RI==0)&&(S_RTS==1)) {}; /*---- SE 正在发送、实验板正在接收一个字符 -----*/
    if (RI==1)
    { /*---- RI==1:实验板接收到一个字符 -----*/
        ACC=SBUF;
        if (P!=RB8) {re_er=1;}; /*----- 偶校验错，置偶校验错标志 -----*/
        rebox[j]=SBUF; /*----- 接收一个字符送串口输入缓冲区 -----*/
        RI=0;
    }
    else
    { /*---- (RI==0)&&(S_RTS==0):SE 报文全部发送完毕而实验板还没有接收到预期的长度 -----*/
        j=y+1; /*---- 退出等待接收 -----*/
    };
};
/*=====接收完毕=====*/
if (j==y)
{ /*---- (j==data_out_len):实验板接收回答报文长度正确 -----*/
    if (re_er==0)
    { /*----- 无偶校验错 -----*/
        if (req_comD0==0)
        { /*---- 请求回答 A 格式 ----*/
            x=0;
            for (i=0;i<dol;i++)
            {
                x=x+rebox[i]; /*----- 求校验和 -----*/
            };
            if (x==rebox[dol])
            {
                AB_status=rebox[0];
                if (AB_sD7==0) /*----见头部说明: sbit AB_sD7=AB_status^7: 报文格式 -----*/
                { /*-----接收的确实是 A 格式-----*/
                    y=dol-1; /*----- y=4-1=3 -----*/
                    for (i=1;i<dol;i++)
                    {
                        pb_out[i-1]=rebox[i];
                    };
                    req_comD0=AB_sD0; /*---- 见头部说明: sbit AB_sD0=AB_status^0: 用户参数更新标记-----*/
                    /*---- 若 AB_sD0=1, 令 req_comD=1, 下次请求 B 格式 -----*/
                    /*---- 若 AB_sD0=0, 令 req_comD=0, 下次请求 A 格式 -----*/
                    req_comD1=0; /*---- 见头部说明: req_comD1=req_com^1:清参数命令; -----*/
                    /*---- req_comD1=0;不清用户参数更新标记; -----*/
                    se_er=(AB_status/2)&0x03; /*---- AB_status(D2~D1)=SE 接口接收用户板报文出错标记 -----*/
                    error=0;
                }
            }
            else
            { /*----(AB_sD7!=0)接收的不是 A 格式-----*/
                error=0x85; /*---- 接收格式 A 错 -----*/
            };
        }
        else
        {
            error=0x81; /*----- 校验和 A 错 -----*/
        };
    };
}
} /****** (req_comD0==0) *****/
else
{ /****** (req_comD0==1)请求回答 B 格式 *****/
    y=user_prm_len+1;
    x=0;
    for (i=0;i<y;i++)

```

```

{
    x=x+rebox[i];          /*----- 求校验和 -----*/
};
if (x==rebox[y])
{
    /*----- 无校验和错 -----*/
    AB_status=rebox[0];
    if (AB_sd7==1)          /*----见头部说明: sbit AB_sd7=AB_status^7: 报文格式 -----*/
    { /*----- 接收的确实是 B 格式-----*/
        for (i=1;i<y;i++)
        {
            user_prm[i-1]=rebox[i]; /*---- 传送新接收到的用户参数到 user_prm[10] -----*/
        };
        Config(p);          /*---- 按照新用户参数设置 I/O 通道 -----*/
        req_comD0=0;         /*---- 令 req_comD=0, 下次发报文请求回答 A 格式 -----*/
        req_comD1=1;         /*---- 令 req_comD1=1; 下次发报文清用户参数更新标记 -----*/
        se_er=(AB_status/2)&0x03; /*---- AB_status(D2~D1)=SE 接口接收用户板报文出错标记 -----*/
        error=0;
    }
    else
    { /*----- 接收格式 B 错 -----*/
        error=0x86;
    };
}
else
{ /*----- 校验和 B 错 -----*/
    error=0x82;
};
};
} /****** (re_er==0) *****/
else
{ /*----- 接收报文偶校验错 -----*/
    error=0x83;
};
} /****** (j==y) *****/
else
{ /*----- 接收字符数不够-----*/
    error=0x84;
};

/*=====*/
/*-----显示部分-----*/
/*-----*/
scanin();          /*----- 输入键扫描 -----*/
x=di0_7;y=di8_15;  /*----- 根据输入键来决定显示内容 -----*/

if (x==0)
{ /*----- di_0~7 输入为 0 或 !=1, 显示 PROFIBUS 输出-----*/
    if (y==0)
    {
        /*----- 如果 di_8~15 输入为 0, PROFIBUS 输出数据第 1 个字节送 Q0-Q7 显示,第 2 个字节送 Q0-Q7 显示 -----*/
        steout0_7(pb_out[0]);
        steout8_15(pb_out[1]);
    };
    if (y==1)
    {
        /*----- 如果 di_8~15 输入为 1, PROFIBUS 输出数据第 4 个字节送 Q0-Q7 显示,第 3 个字节送 Q0-Q7 显示 -----*/
        steout0_7(pb_out[3]);
        steout8_15(pb_out[2]);
    };
    if (y>1)
    { /*----- 如果 di_8~15 输入 >1, Q0-Q15 显示 0xff -----*/
        steout0_7(0xff);
        steout8_15(0xff);
    };
};
};

```

```
if (x==1)
{ /*----- di_0~7 输入为 1, 显示当前用户参数-----*/
    steout0_7(0); /*----- Q0-Q7 显示 0 -----*/
    if (y<10)
    {
        /*----- di_8~15 输入为 0, Q8-Q15 显示第 1 个用户参数 user_prm[0] -----*/
        /*----- di_8~15 输入为 y, Q8-Q15 显示第 y+1 个用户参数 user_prm[y] -----*/
        steout8_15(user_prm[y]);
    }
    else
    { /*----- 如果 di_8~15 输入 >= 10, Q0-Q15 显示 0xff -----*/
        steout0_7(0xff);
        steout8_15(0xff);
    };
};

if (x>1)
{ /*----- 如果 di_0~7 输入 > 1, Q0-Q15 显示 0xff -----*/
    steout0_7(0xff);
    steout8_15(0xff);
};

}; /*----- while (REQ_IT==0):SE 处在数据交换状态,返回数据交换状态的无限循环中 -----*/
}; /*----- (REQ_IT==1):SE 退出数据交换状态,返回初始化 -----*/
}
```

第五章 建立一个调试实验系统

OEM2 调试实验系统是一个最小化的 PROFIBUS 现场总线系统,本手册推荐的主站有 2 种选择。

- ① 选择 CP5611+PC 机做主站，本手册暂称为系统 I。
- ② 选择西门子公司 PLC（CPU313-2DP 或 CPU315-2DP）做主站，本手册暂称为系统 II。

本章将详细介绍如何建立 OEM2 调试实验系统 I 或系统 II、如何实现 PROFIBUS 系统配置和介绍本手册例 1、例 2、例 3 连通及演示实验。

1. 建立 OEM2 调试实验系统 I

(1) 系统 I 设备清单

从站				
序号	名称	制造商	数量	备注
1	PB-OEM2-SE 嵌入式 PROFIBUS 接口	鼎实科技	1 块	
2	PB-OEM2-SAMPLE OEM2 开发实验板	鼎实科技	1 块	
主站				
3	CP5611 PROFIBUS 主站网卡（PCI）	西门子	1 块	
4	PC 机：Win2000 操作系统、PCI 插槽	*****	1 台	工控机、兼容机均可
网络器件				
4	PROFIBUS 电缆	LAPP	10 米	
5	PROFIBUS 插头	西门子	2 个	
软件				
6	组态软件（光盘）： COM PROFIBUS V5.1(DEMO 版)；A#光盘 SOFTNET-DP V5.2(DEMO 版，包含在 SIMATIC NET B#光盘中)； WinCC V5.1(DEMO 版)；C#光盘	西门子	3CD	随系统赠送
文件资料				
7	文件资料（D#光盘）包括： GSD 文件 实验板硬件原理图 实验板软件清单（C51 源代码） PROFIBUS 主站系统组态及调试实验软件 《PB-OEM2-SE 产品手册》 《PROFIBUS-OEM2 调试实验系统使用手册》	鼎实科技	1CD	随系统赠送
8	手册（印刷品）： 《PB-OEM2-SE 产品手册》 《PROFIBUS-OEM2 调试实验系统使用手册》	鼎实科技	1 套	随系统赠送

用户自备				
序号	名称	制造商	数量	备注
1	PC 机：Win2000 操作系统、PCI 插槽	*****	1 台	工控机、兼容机均可
2	24VDC（2A 以上）直流电源	*****	1 台	

(2) 系统 I 结构图

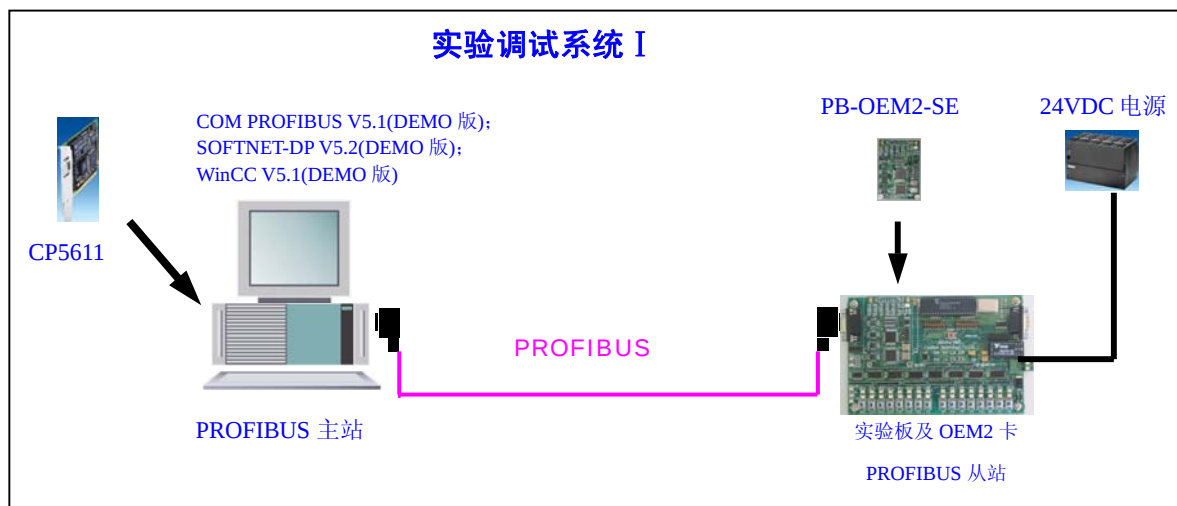


图 5-1 系统 I 结构图

(3) 安装实验调试系统 I

(一) 硬件安装

- ① CP5611: 按照 CP5611 说明书安装在 PC 机的 PCI 槽中;
- ② 安装开发实验板 PB-OEM2-SAMPLE 及嵌入式 PROFIBUS 接口 PB-OEM2-SE。注意 24VDC 电源的正负极性。
- ③ 在实验板上设定 PROFIBUS 从站地址, 为与本手册的例 1~例 3 统一, 将开发实验板的从站地址设置成: 19 (13H) 号从站。
- ④ PROFIBUS 电缆与插头的连接: B 是红色线。注意电缆屏蔽层与 PE 的连接。两个 PROFIBUS 插头的终端电阻都要打到 ON 位置。

(二) 主站 PC 机软件安装

根据本调试实验系统提供的 CD 光盘, 依此安装如下软件, 如有必要可参照 SIEMENS 公司关于这些软件的安装资料。PC 机系统要求: Microsoft Windows 2000:

- ① 安装 COM PROFIBUS V5.1 软件;
- ② PB SOFTNET-DP V5.2(DEMO 版);

注意:

▼由于安装 DEMOD 版, 因此安装中不选择“执行授权”, 见下页图 5-2。

▼安装软件列表中只需要选择 PB SOFTNET-DP V5.2 即可, 见下页图 5-3。

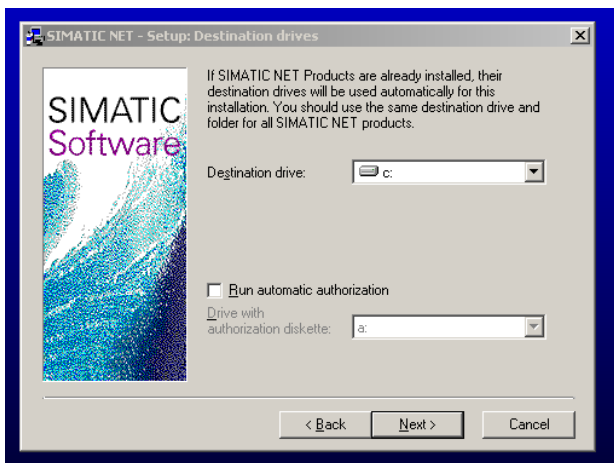


图 5-2 只安装 DEMO 板因此不选择“执行授权”

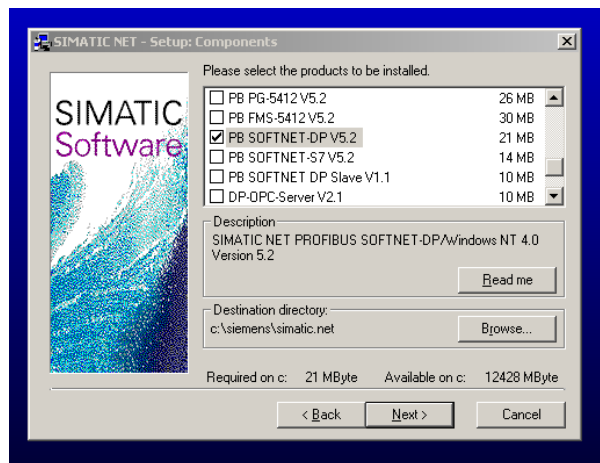


图 5-3 安装软件列表中只选择 PB SOFTNET-DP V5.2 即可

③ 安装 WinCC 软件;

注意:

▼序列号: 可键入“0”;

▼由于安装 WinCC DEMOD 版, 因此安装中选择“稍后执行授权”;

(4) 按照系统提供的例 2 实现系统连通实验

(一) 在 COM PROFIBUS 配置软件中实现简单连通

① 拷贝 GSD 文件、图标文件、例 2 配置文件

▼将 GSD 文件 OEM2_B2.GSD 拷贝至 CPBV51\gsd

▼将图标文件 SE_B.DIB 拷贝至 CPBV51\bitmaps

▼将例 2 配置文件 SE_B2_CONF 拷贝至 CPBV51\data

② 进入 COM PROFIBUS V5.1, 打开例 2 配置文件 SE_B2_CONF.pb5, 如图 5-4 所示:

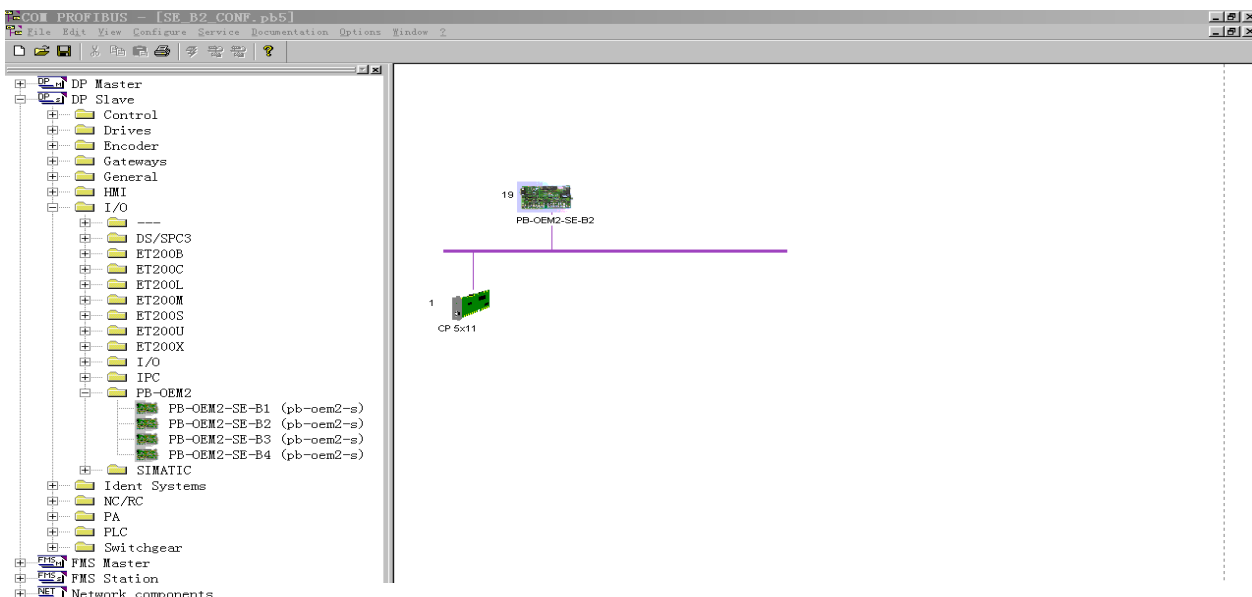


图 5-4 进入 COM PROFIBUS V5.1, 打开例 2 配置文件 SE_B2_CONF.pb5

- ③ 检查 Setting the PG/PC Interface....确认 “COM_PROFIBUS →CP5611(COM PROFFUBUS), 见图 5-5。

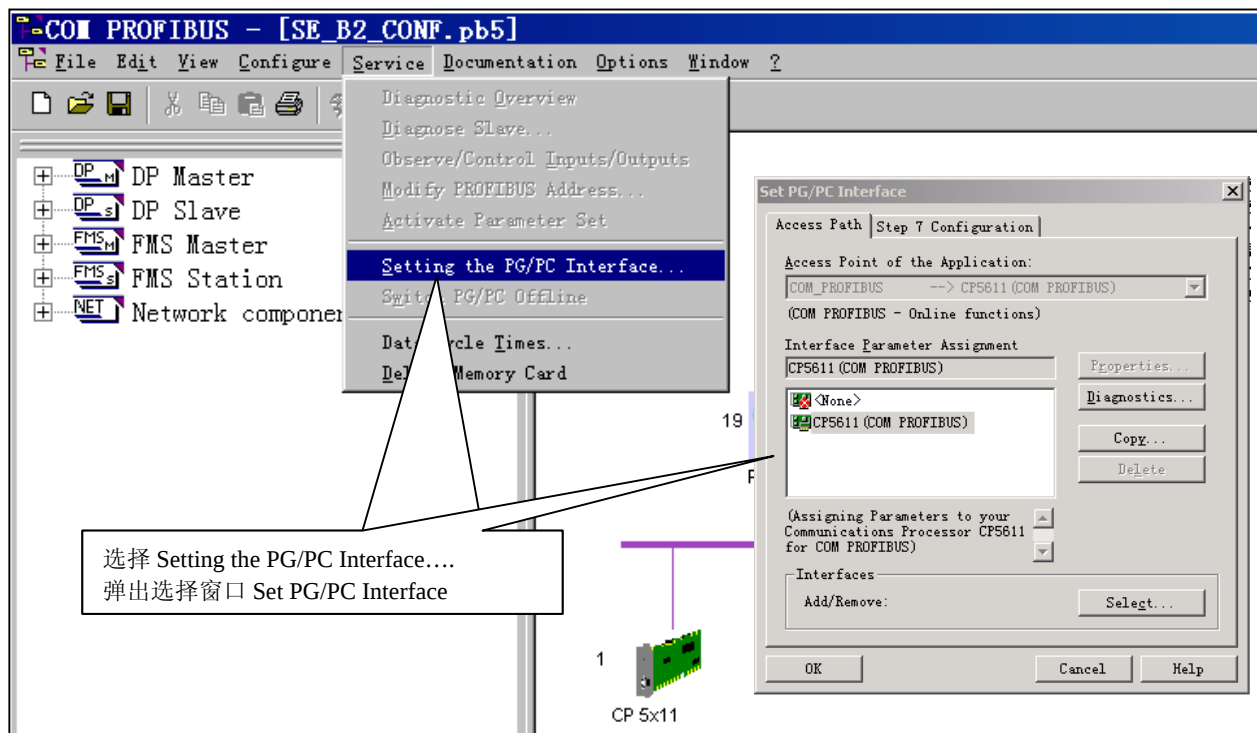


图 5-5 检查 Setting the PG/PC Interface....确认 “COM_PROFIBUS →CP5611(COM PROFFUBUS)

- ④ 系统诊断：双击 19 号从站图标，弹出 Slave properties 窗口。选择 Diagnostics....，等待 PC 主站检测从站,弹出 Busparameters of DP card 窗口设置波特率。通常设置成 187.5K，见图 5-6。

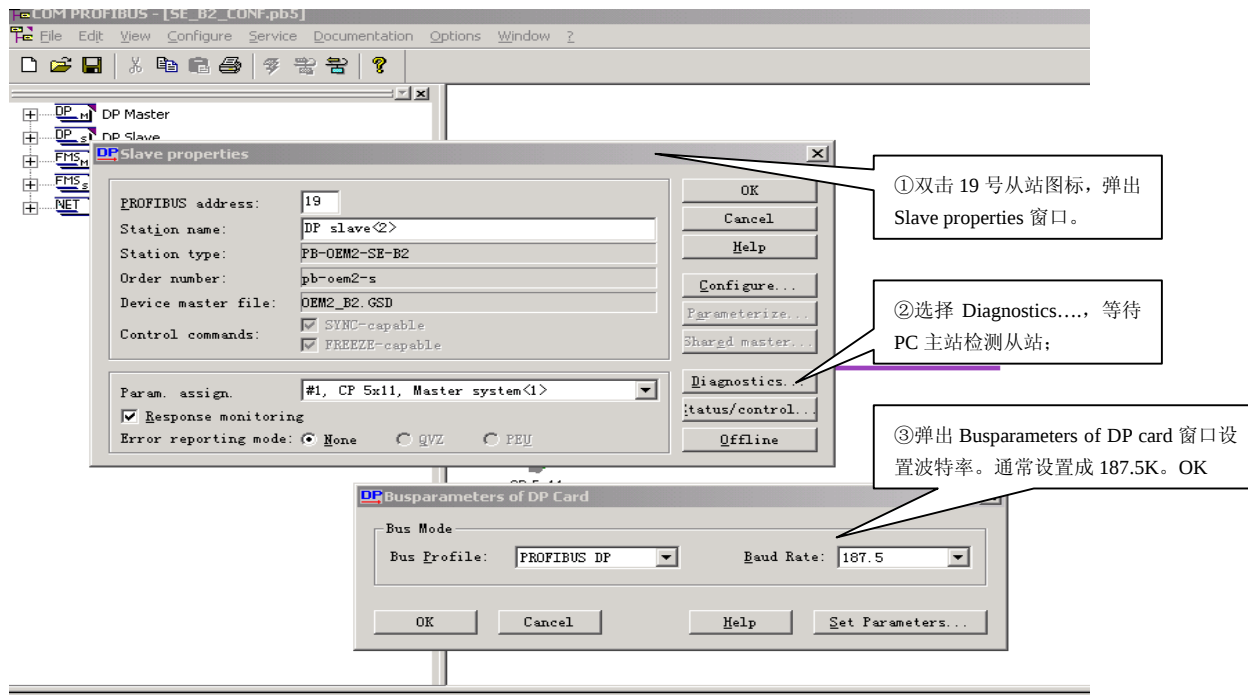


图 5-6 系统诊断

注意 1:弹出 Slave diagnosis: PB-B-OEM2-SE-B2 #19 <DP slave>,如果窗口中没有信息, 见图 5-7: 主站没有检测到 19#实验板从站。说明通信硬件连接有错误。只有弹出象“图 5-8: 主站检测到 19#实验板从站”窗口, 才说明通信硬件连接没有错误。

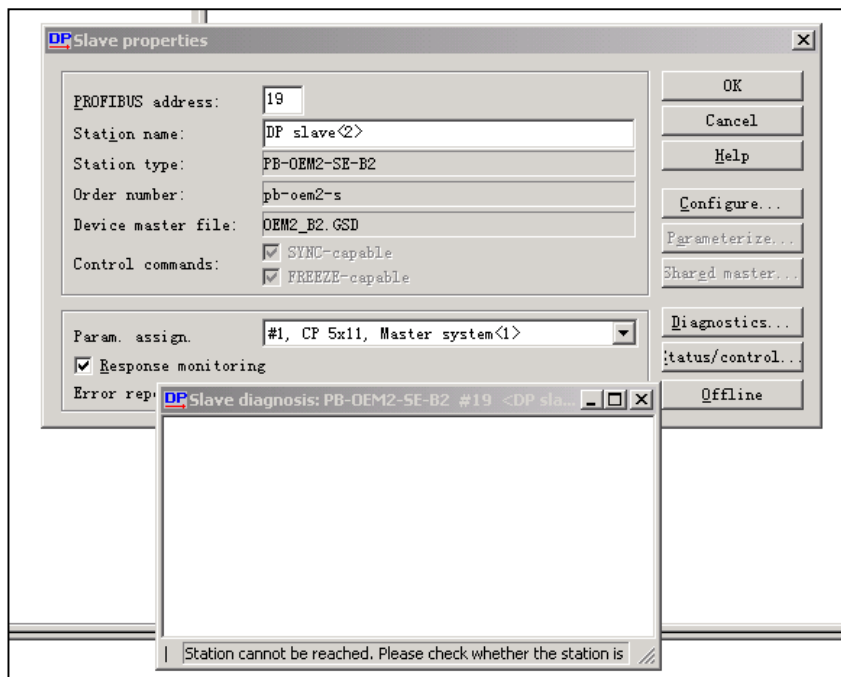


图 5-7 主站没有检测到 19#实验板从站

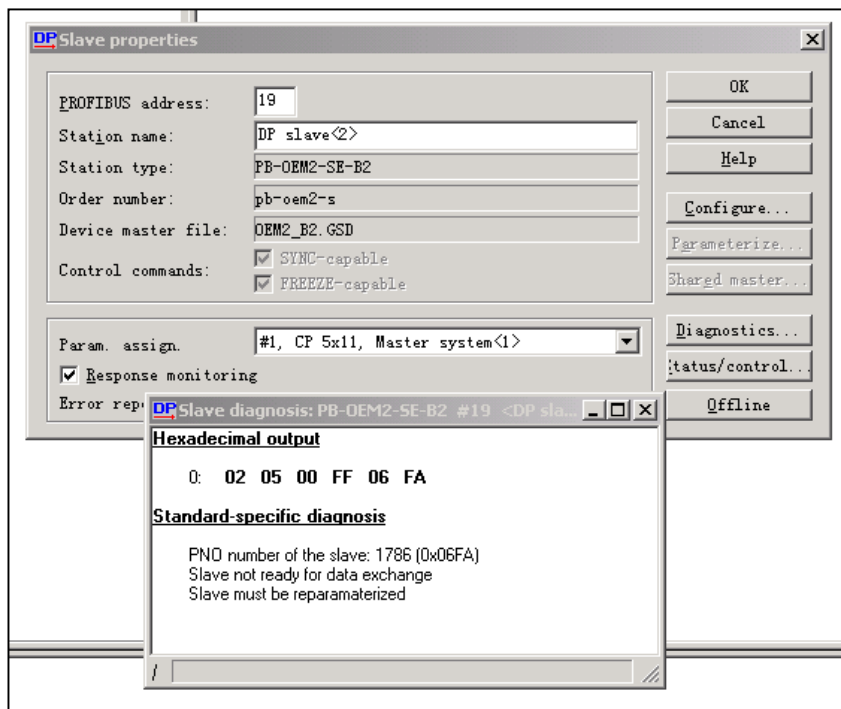


图 5-8 主站检测到 19#实验板从站

注意 2: “主站没有检测到 19# 实验板从站” 的故障原因可能是:

▼实验板从站站号设置错误: 不是 19# (13H);

▼实验板从站硬件安装错误: 24V 电源指示灯 (均是红色) VCC/OK、24V/OK、5V/OK、TXD、RXD 及 OEM2-SE 接口板上的 PB_F 都应该稳定点亮。

▼PROFIBUS 电缆插头安装错误: 电缆与插头的安装。

⑤ 系统连通

(i) 同系统诊断，双击 19 号从站图标，弹出 Slave properties 窗口。选择 Status/control....，等待 PC 主站检测从站并弹出 Busparameters of DP card 窗口设置波特率。通常设置成 187.5K，见图 5-9。

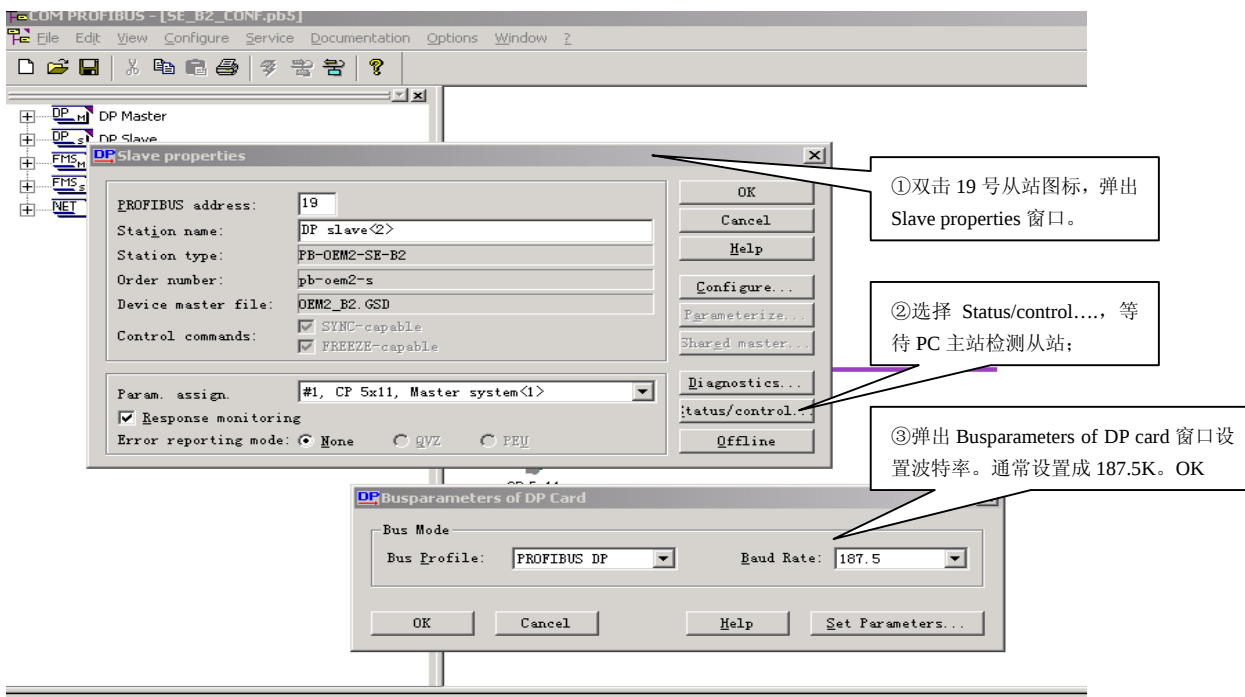


图 5-9 系统连通-1

(ii) 主站检测到 19#从站后，弹出窗口“PB-OEM2-SE-B2 #19 <DP Slave<2>>”及稍后弹出“Note”窗口，选择“确定”。见图 5-10,系统连通-2。

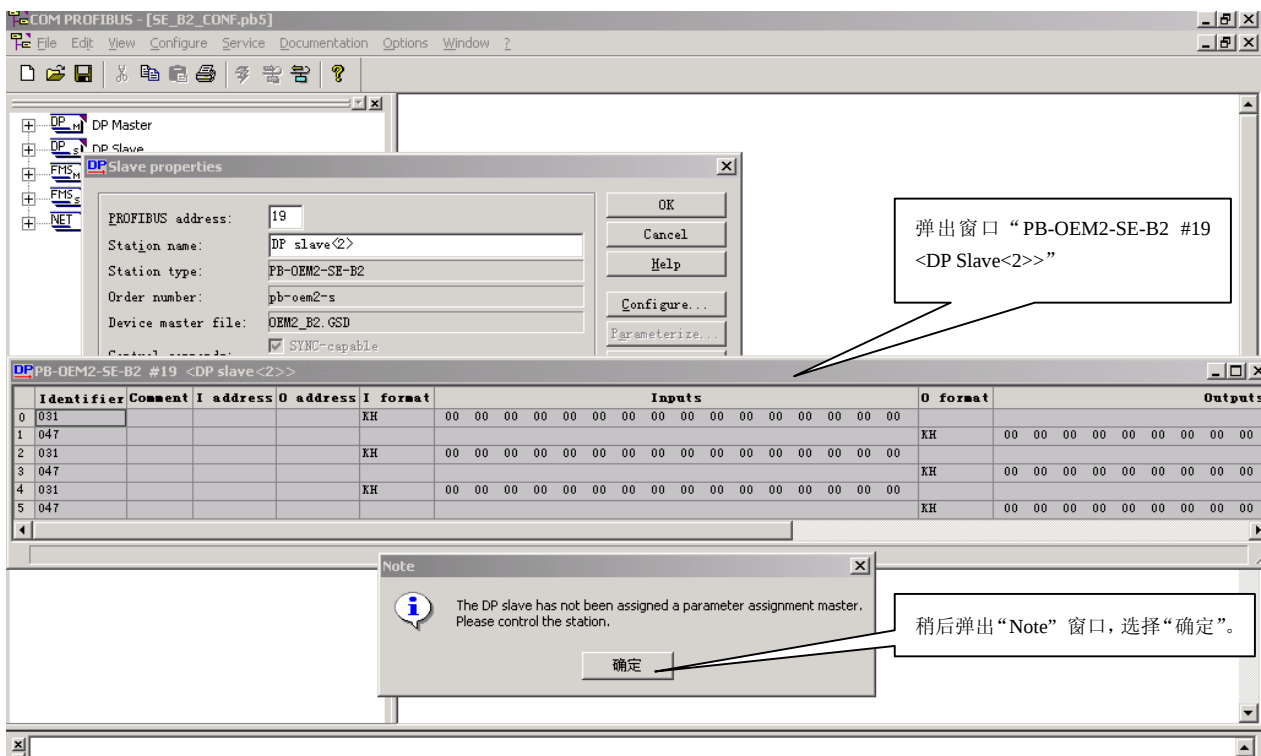


图 5-10 系统连通-2

(iii) 主站弹出“Contro values for position 1”窗口，选择“Control”，见图 5-11，系统连接 3。

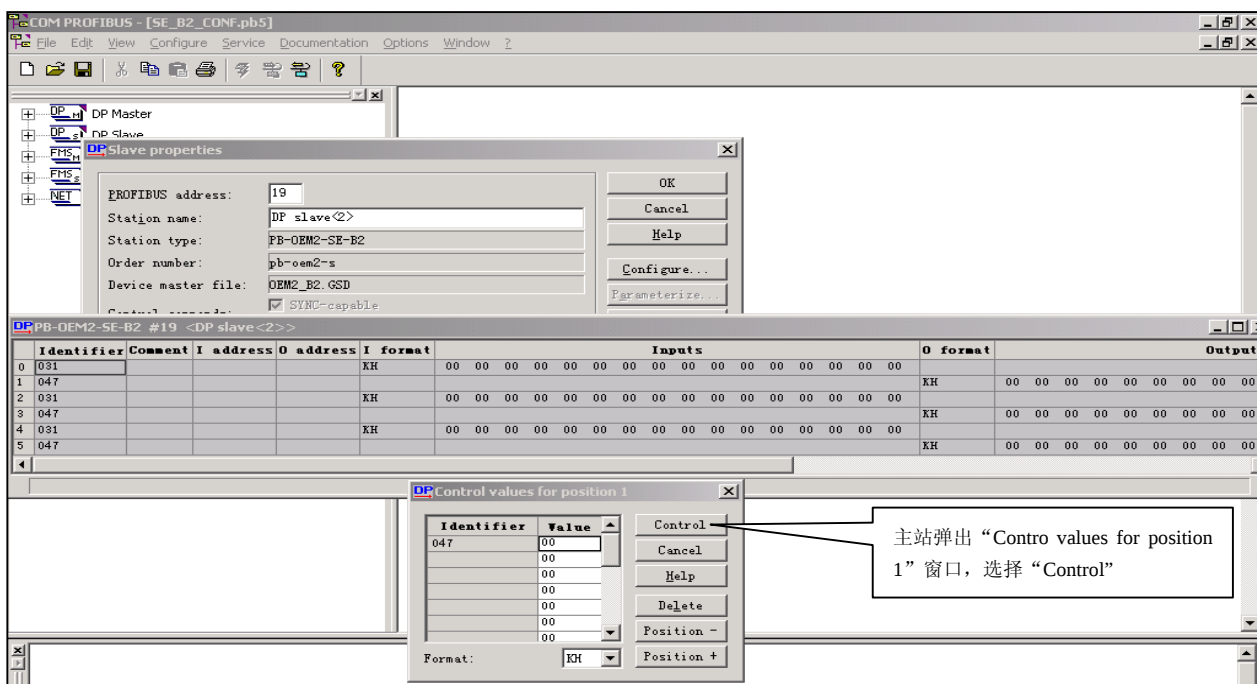


图 5-11 系统连通-3

(iv) 系统连通并观察 PROFIBUS 输入。见图 5-12 系统连通-4，在窗口“PB-OEM2-SE-B2 #19 <DP Slave<2>>”中可以看到“Inputs”—PROFIBUS 的 48 字节输入：除 I0、I1 字节是实验板键输入，其余 I2~I47 均为 5AH，见例 2：实验板程序清单中有如下程序段便可知：

```
trbox[1]=di0_7;          /*---- di0_7 是输入键输入 I0~I7 当前值，作为 PROFIBUS 输入的第 1 字节 ----*/
trbox[2]=di8_15;         /*---- di8_15 是输入键输入 I8~I15 当前值，作为 PROFIBUS 输入的第 2 字节 ----*/
for (i=3;i<di1;i++)
{
    trbox[i]=0x5a;        /*---- 其它 PROFIBUS 输入数据(3~48)本例定为 0x5a ----*/
};
```

注意：系统连通的主要标志是：PB-OEM2-SE 接口板上的指示灯（红）PB F 灭。

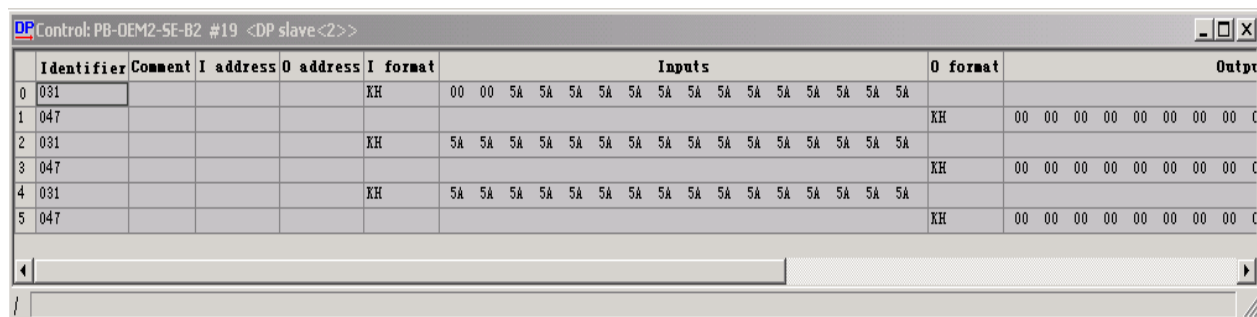


图 5-12 系统连通-4 系统连通并观察 PROFIBUS 输入

(v) 改变 PROFIBUS 输入：按实验板输入键，可在线观察 PROFIBUS 输入 I0、I1 的变化。见图 5-13。

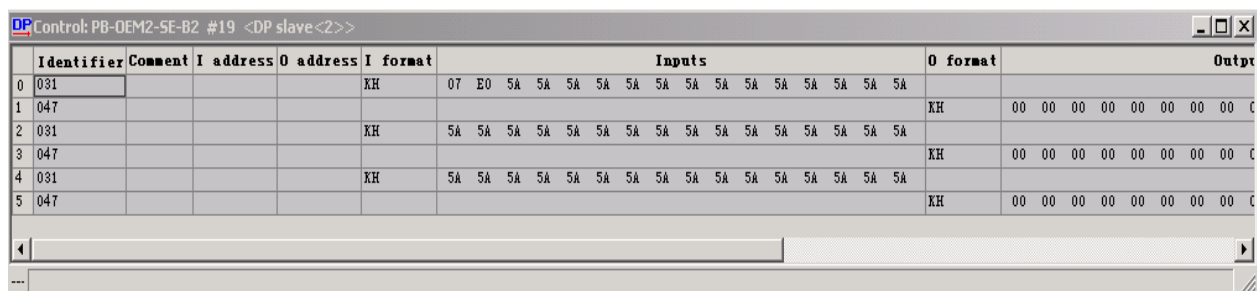


图 5-13 系统连通-5 可在线观察 PROFIBUS 输入 I0、I1 的变化

(vi) 改变 PROFIBUS 输出：双击 PROFIBUS 输出，弹出“Control values for position 1”窗口，在 Q0、Q1 位置置入：33、CC，选择“Control”；主站将 Q0、Q1 值通过 PROFIBUS 送到实验板从站；观察实验板输出指示灯（红），对应显示 00110011（右）、11001100（左）见图 5-14：系统连接-6。

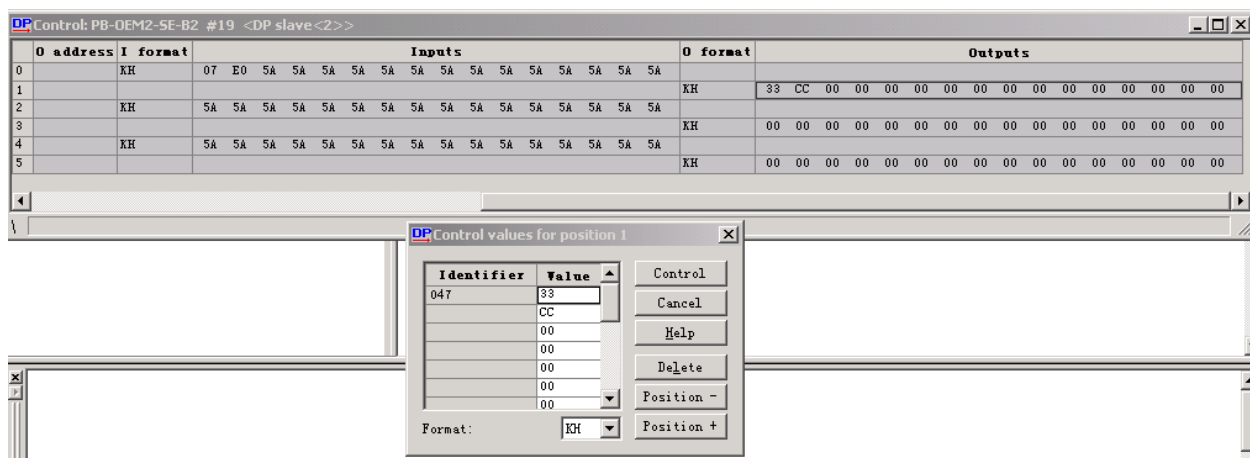


图 5-14 系统连接-6 改变 PROFIBUS 输出

(二) 自己完成配置例 2 实现简单连通

- ① 进入 COM PROFIBUS V5.1
- ② 配置主站：主站选择 CP5×11、设定“Set Bus Parameters..”，通常选择 187.5K，见图 5-15。

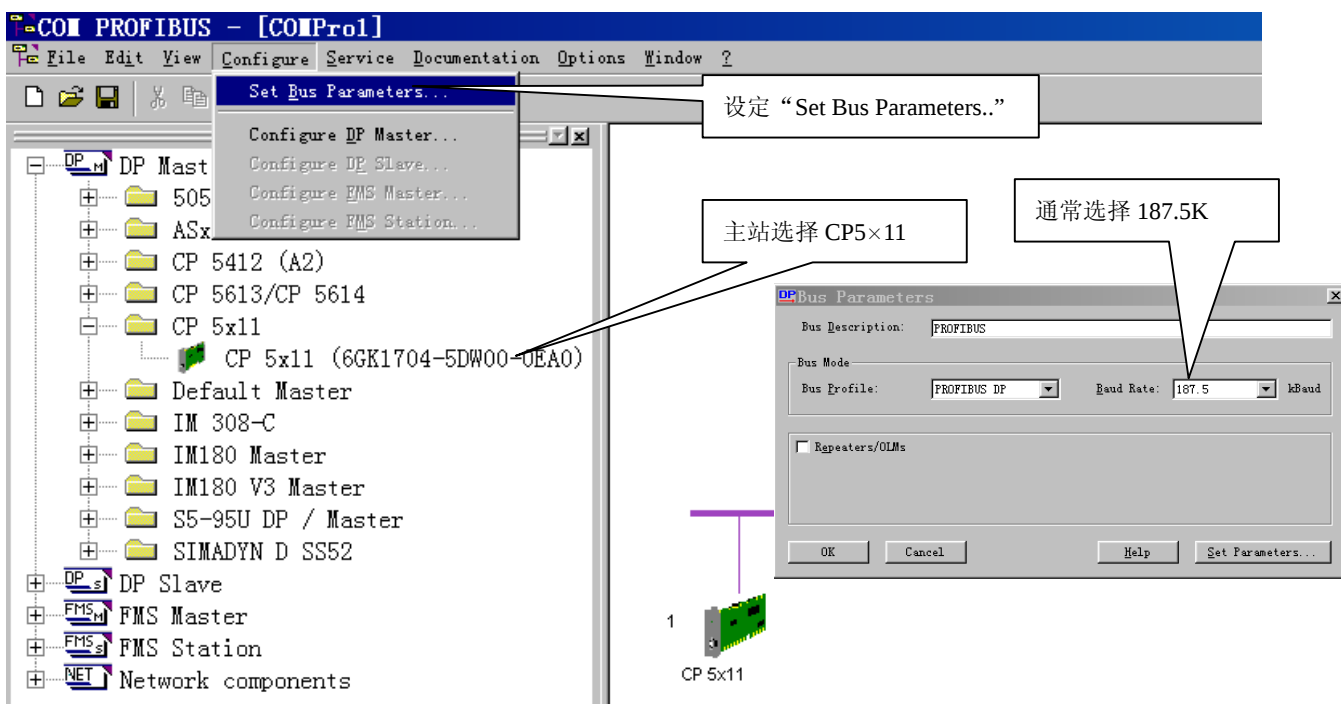


图 5-15 配置主站

- ③ 配置从站：选择“PB-OEM2-SE-B2”，设置站号 19。见图 5-16，配置从站。

注意：如果没有找到“PB-OEM2-SE-B2”，可能是忘记拷贝 GSD 文件到 OEM2_B2.GSD 至 CPBV51\gsd 或应该更新设备目录，见图 5-17 更新设备目录。

- ④ 配置文件存盘

⑤ 转到 “(-) 在 COM PROFIBUS 配置软件中实现简单连通：③ — ⑤”

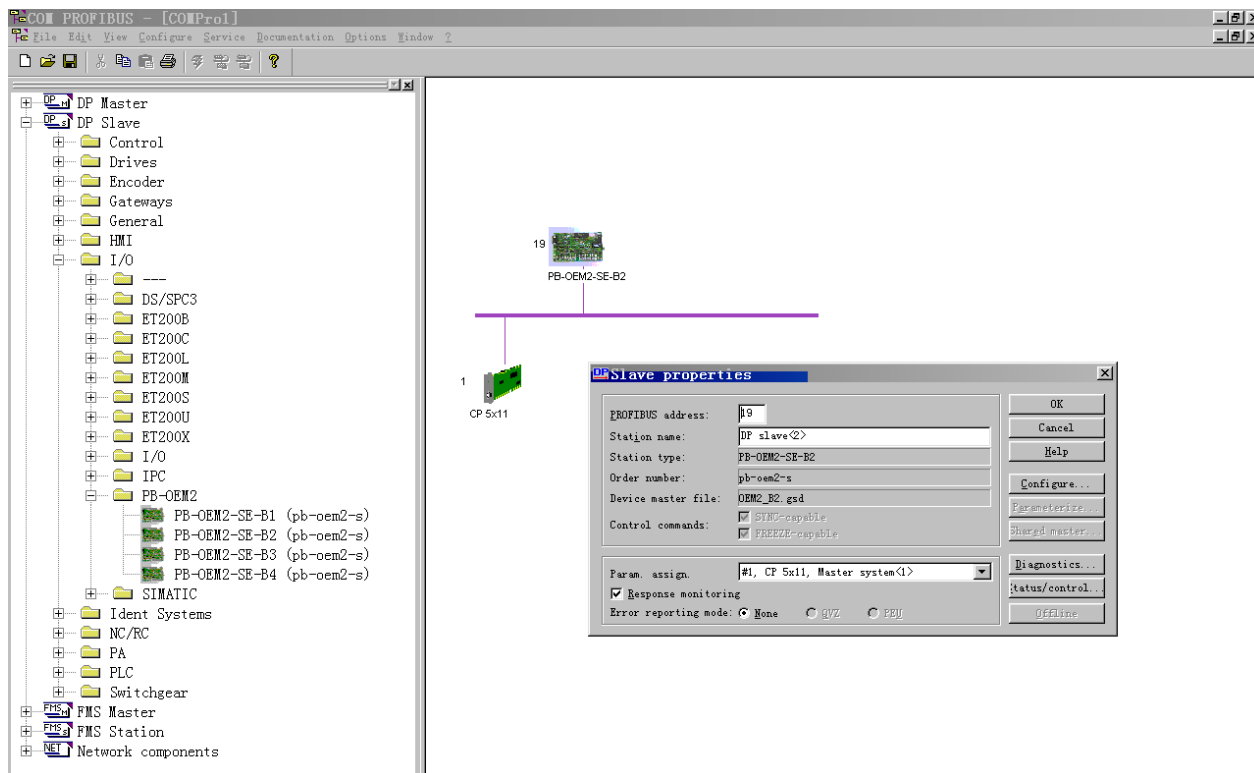


图 5-16 配置从站

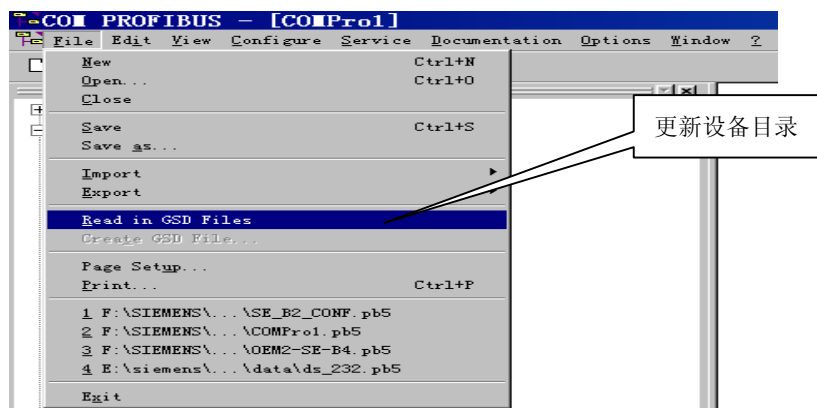


图 5-17 更新设备目录

2. CP5611+Wincc 做主站实现系统 I 的 PROFIBUS 通信

例 1 与例 2 的 GSD 文件相同，因此 CP5611 和 WinCC 配置相同。

(1) 设置 “Set PG/PC Interface...”:

- ① NCM 文件：将例 2 的 COM PROFIBUS 配置的 OEM2_SE_B2.ldb 文件拷贝 CPBV51/ncm；或自主形成 OEM2_SE_B2.ldb，方法见图 5-18 形成 NCM 文件。

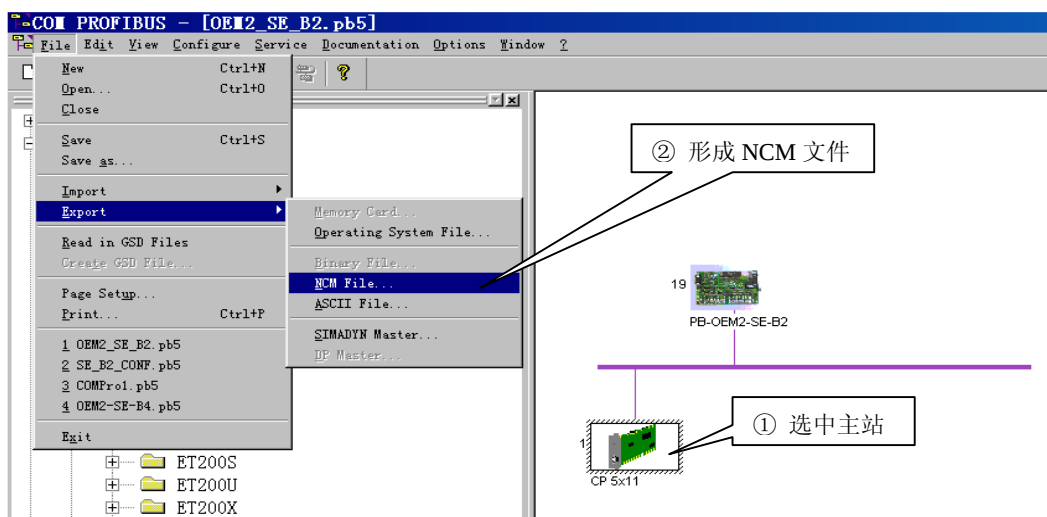


图 5-18 形成 NCM 文件

② 设置 “Set PG/PC Interface...”: 从 “控制面板” 中进入 “Set PG/PC Interface...”, 选择:

“CP_L2_1: → CP5611(PROFIBUS-DP Master)”、进入 “Properties..”、Database name 指向 COM PROFIBUS 形成的 NCM 文件。如果能够 Read 出 19#站及 48Input/48Output,说明 NCM 文件及 Set PG/PC Interface 配置正确, 见图 5-19。

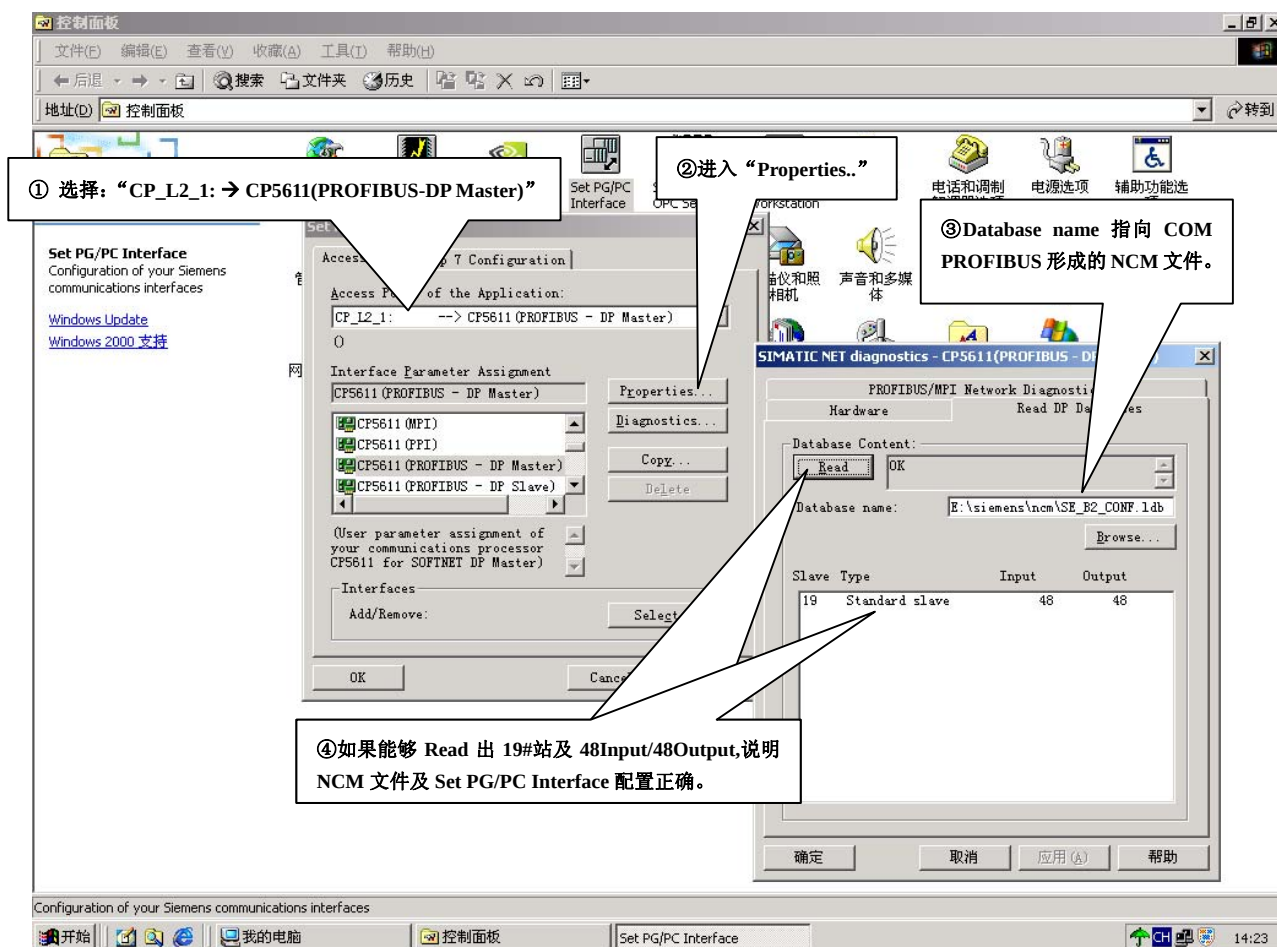


图 5-19 从 “控制面板” 中进入 “Set PG/PC Interface...”

(2) 建立 WinCC 项目

① 将 WinCC 例 1-2 演示实验项目文件夹：TEST_OEM2_B12 拷贝到 WINCC\WinCCProjects\并将文件夹属性由“只读”设置成“存档”。

② 进入 WinCC、打开项目文件夹 TEST_OEM2_B12，改计算机名。见图 5-20

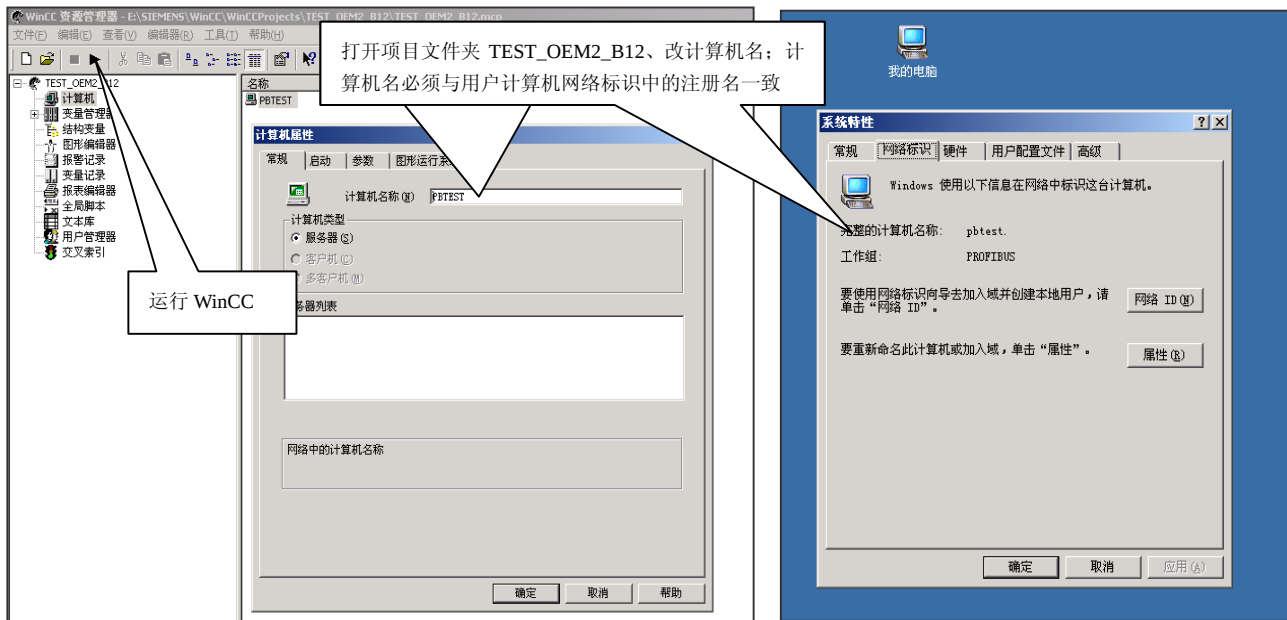


图 5-20 进入 WinCC，打开项目文件夹 TEST_OEM2_B12 并更改计算机名

(3) 运行 WinCC

使 WinCC 进入运行状态，见图 5-20。鼎实科技提供的例 1~2 实验项目 TEST_OEM2_B12 将有如下调试画面，见图 5-21。例 1~2 实验项目 TEST_OEM2_B12 调试界面 I

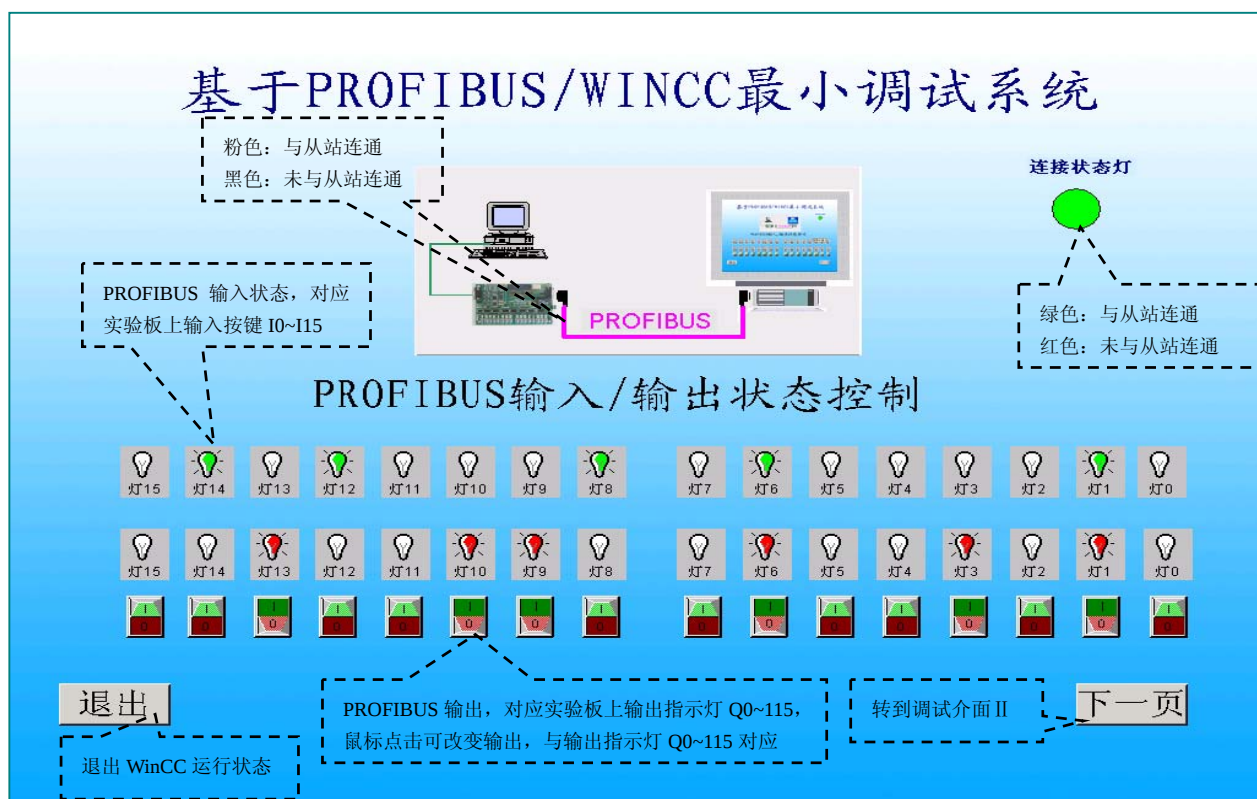


图 5-21 例 1~2 实验项目 TEST_OEM2_B12 调试画面 I

进入“下一页”，可以看到全部 48 BYTES INPUT/48BYTES OUTPUT 数据，见图 5-22：例 1~2 实验项目 TEST_OEM2_B12 调试介面 II。

**PROFIBUS数据接收区
(48 Bytes)**

输入字节0:	42	输入字节24:	5A
输入字节1:	51	输入字节25:	5A
输入字节2:	5A	输入字节26:	5A
输入字节3:	5A	输入字节27:	5A
输入字节4:	5A	输入字节28:	5A
输入字节5:	5A	输入字节29:	5A
输入字节6:	5A	输入字节30:	5A
输入字节7:	5A	输入字节31:	5A
输入字节8:	5A	输入字节32:	5A
输入字节9:	5A	输入字节33:	5A
输入字节10:	5A	输入字节34:	5A
输入字节11:	5A	输入字节35:	5A
输入字节12:	5A	输入字节36:	5A
输入字节13:	5A	输入字节37:	5A
输入字节14:	5A	输入字节38:	5A
输入字节15:	5A	输入字节39:	5A
输入字节16:	5A	输入字节40:	5A
输入字节17:	5A	输入字节41:	5A
输入字节18:	5A	输入字节42:	5A
输入字节19:	5A	输入字节43:	5A
输入字节20:	5A	输入字节44:	5A
输入字节21:	5A	输入字节45:	5A
输入字节22:	5A	输入字节46:	5A
输入字节23:	5A	输入字节47:	5A

**PROFIBUS数据发送区
(48 Bytes)**

输出字节0:	4A	输出字节24:	0
输出字节1:	26	输出字节25:	0
输出字节2:	0	发送字节26:	0
输出字节3:	0	发送字节27:	0
输出字节4:	0	发送字节28:	0
输出字节5:	0	发送字节29:	0
输出字节6:	0	发送字节30:	0
输出字节7:	0	发送字节31:	0
输出字节8:	0	发送字节32:	0
输出字节9:	0	发送字节33:	0
输出字节10:	0	发送字节34:	0
输出字节11:	0	发送字节35:	0
输出字节12:	0	发送字节36:	0
输出字节13:	0	发送字节37:	0
输出字节14:	0	发送字节38:	0
输出字节15:	0	发送字节39:	0
输出字节16:	0	发送字节40:	0
输出字节17:	0	发送字节41:	0
输出字节18:	0	发送字节42:	0
输出字节19:	0	发送字节43:	0
输出字节20:	0	发送字节44:	0
输出字节21:	0	发送字节45:	0
输出字节22:	0	发送字节46:	0
输出字节23:	0	发送字节47:	0

PROFIBUS 输入第 1、2 个字节对应实验板输入键 I0~I15。按输入键 I0~I15 可以看到调试介面“输入字节 0”、“输入字节 1”的变化。

PROFIBUS 输出，键入数据可观察到实验板输出指示灯的变化。

PROFIBUS 输入第 2~48 字节均为 5AH。

返回调试介面 I

返回

图 5-22 例 1~2 实验项目 TEST_OEM2_B12 调试画面 II

3. 建立 OEM2 调试实验系统 II

(1) 系统 II 设备清单

从站				
序号	名称	制造商	数量	备注
1	PB-OEM2-SE 嵌入式 PROFIBUS 接口	鼎实科技	1 块	
2	PB-OEM2-SAMPLE OEM2 开发实验板	鼎实科技	1 块	
主站				
3	S7-300 PLC/CPU313-2DP	西门子	1 块	
4	MPI 编程电缆	西门子	1 根	
网络器件				
4	PROFIBUS 电缆	LAPP	10 米	
5	PPROFIBUS 插头	西门子	2 个	
软件				
6	组态编程软件（E#光盘）： STEP 7 V5.1+SP1(DEMO 版)	西门子	3CD	随系统赠送
文件资料				
7	文件资料（D#光盘）包括： GSD 文件 实验板硬件原理图 实验板软件清单（C51 源代码） PROFIBUS 主站系统组态及调试实验软件 《PB-OEM2-SE 产品手册》	鼎实科技	1CD	随系统赠送

	《PROFIBUS-OEM2 调试实验系统使用手册》			
8	手册（印刷品）： 《PB-OEM2-SE 产品手册》 《PROFIBUS-OEM2 调试实验系统使用手册》	鼎实科技	1 套	随系统赠送
用户自备				
序号	名称	制造商	数量	备注
1	PC 机：Win2000 操作系统，用做编程器	*****	1 台	工控机、兼容机均可
2	24VDC（2A 以上）直流电源	*****	1 台	

(2) 系统 II 结构图

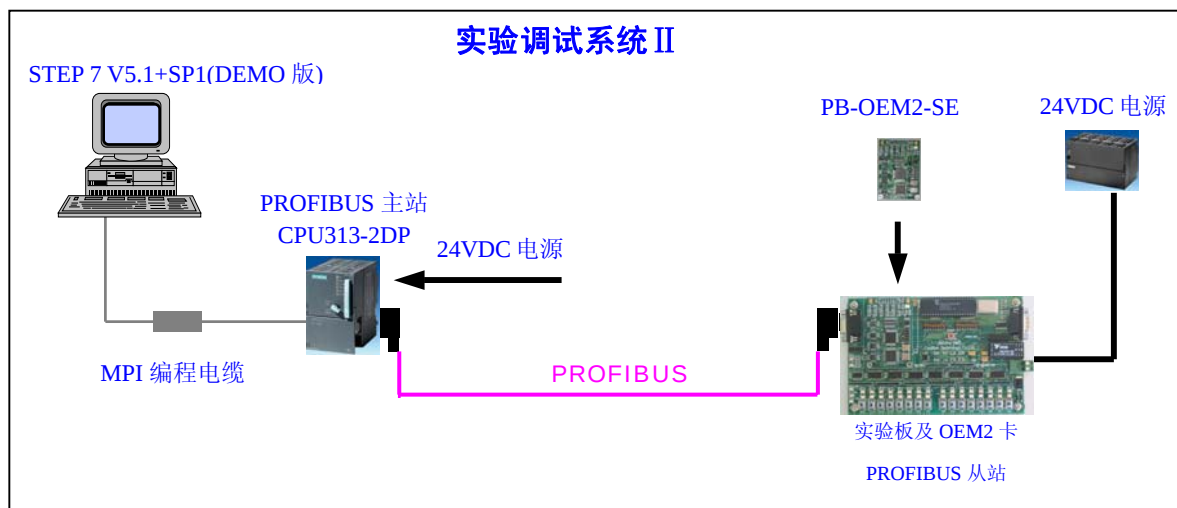


图 5-23 系统 II 结构图

(3) 安装实验调试系统 II

(一) 硬件安装

- ① S7-300/CPU315-2DP: 按照 S7-300 安装手册安装，接通 CPU315-2DP 24V 直流电源。注意电源极性 & PE（接地）的连接。自备 PC 机做 S7-300 编程器，MPI 编程电缆一端连接 PC 机串口，另一端连接 CPU313-2DP 的 MPI 接口。
- ② 安装开发实验板 PB-OEM2-SAMPLE 及嵌入式 PROFIBUS 接口 PB-OEM2-SE；注意 24VDC 电源的正负极性。
- ③ 在实验板上设定 PROFIBUS 从站地址。为适合本手册的例 1~例 3，现将开发实验板从站地址设置成：19（13H）。
- ④ PROFIBUS 电缆与插头的连接：B 是红色线，注意电缆屏蔽层与 PE 的连接。两个 PROFIBUS 插头终端电阻都要打到 ON 位置。

(二) 主站 PC 机软件安装

根据本调试实验系统提供的 CD 光盘，依此安装 SIMATIC STEP 7 V5.1 软件；如有必要可参照 SIEMENS 公司关于软件的安装资料。PC 机系统要求：Microdoft Windows 2000

① 先安装 SIMATIC STEP 7 V5.1: 运行 STEP7/Disk1\Steup.exe

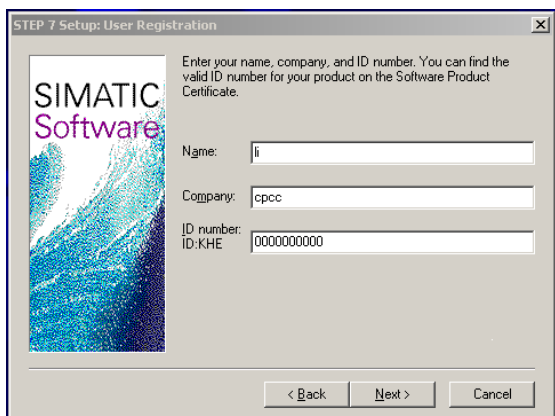


图 5-24 ID 号输入 0

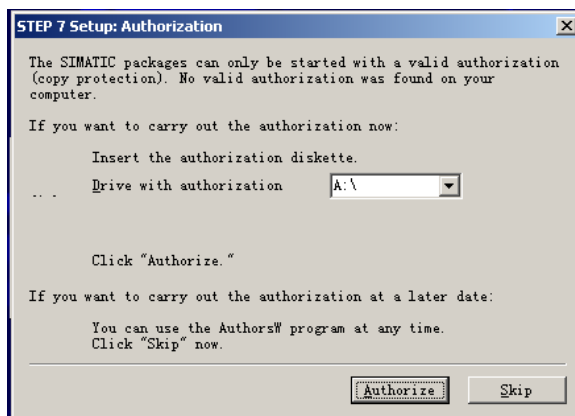


图 5-25 安装 DEMO 版 Skip 授权

② 再安装 SIMATIC STEP 7 V5.1 ServicePack 1: 运行 STEP7/sp1/Disk1\Steup.exe

(4) 按照系统提供的例 2 实现系统连通实验

① 拷备 GSD 文件、图标文件、例 2 的 STEP 7 项目文件

▼将 GSD 文件 OEM2_B2.GSD 拷备至 Step7\S7data\gsd\

▼将图标文件 SE_B.bmp 拷备至 Step7\S7data\nsbmp\

▼将例 2 的 STEP 7 项目文件夹 SE_B2_step7 拷备至 Step7\S7proj\

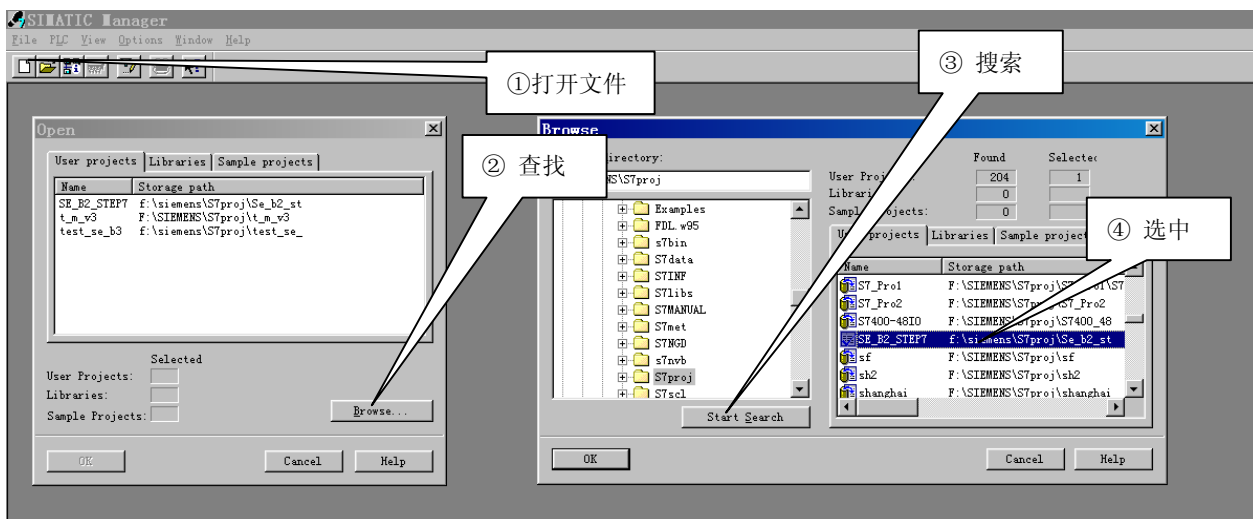
② 进入 打开例 2 项目文件 SE_B2_STEP7, 见图 5-26。

图 5-26 进入 SIMATIC Manager 打开例 2 项目文件 SE_B2_STEP7

② 设置 “Set PG/PC Interface...”, 以便下载程序, 见图 5-27。

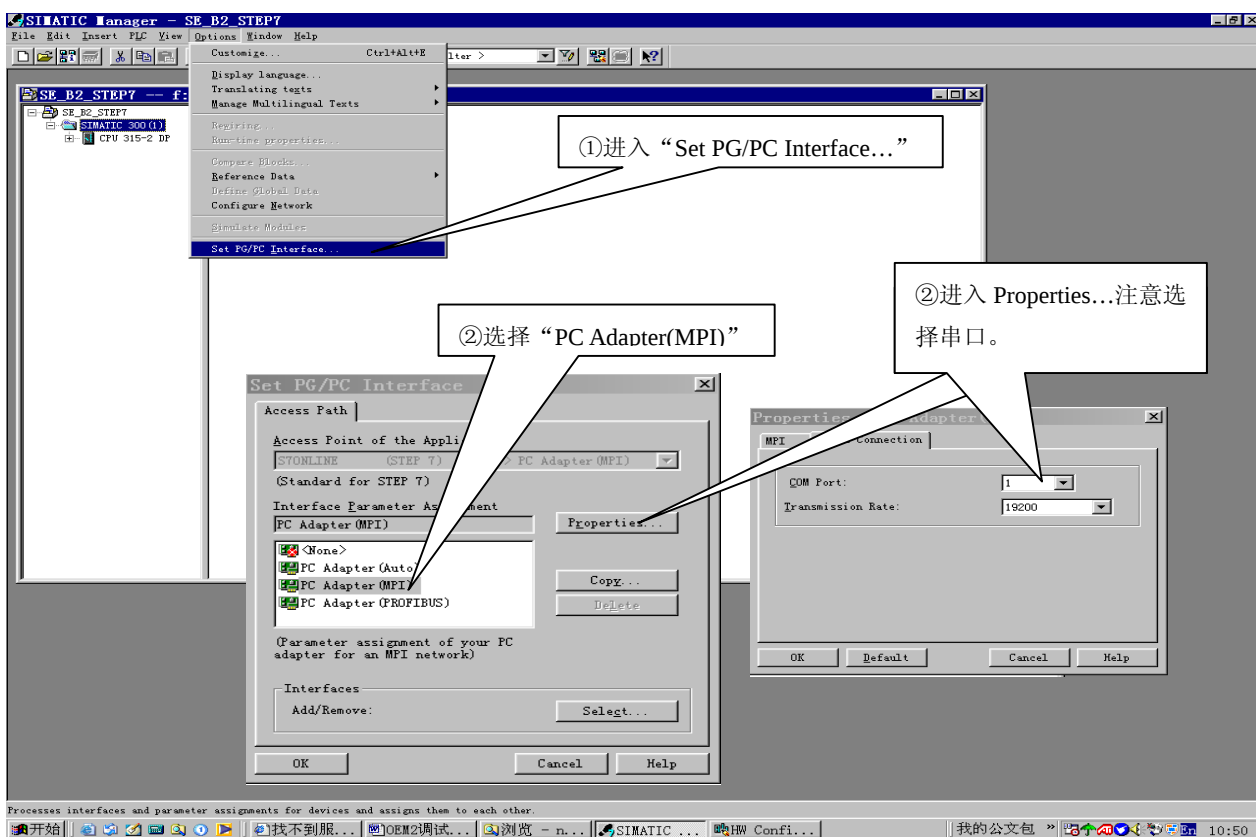


图 5-27 设置“Set PG/PC Interface...”，以便下载程序

④ 下载配置和程序，见图 5-28。配置和程序下载完成后，系统应该连通了。



图 5-28 下载配置和程序

⑤ 系统连通标志

配置和程序下载完成后，系统应该连通了。系统连通标志是：S7-300/CPU313-2DP 上的 BUSF 红灯灭并且 OEM2 接口板上的 PBF 红灯灭，见图 5-29。

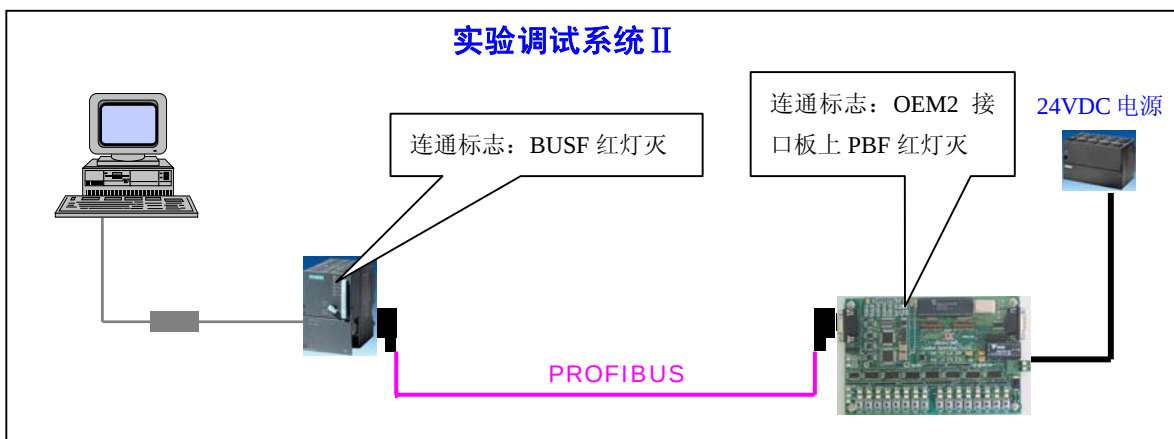


图 5-29 系统 II 结构图

⑥ 系统连通故障

i、检查实验板和 S7-300 PLC 电源。不妨对 PLC 和实验板从新上电。

注意：实验板上电后：VCC/OK、5V/OK、24V/OK、TXD、RXD 都会稳定点亮，输入/输出指示灯会有一段指示灯流水显示；

ii、检查 PROFIBUS 插头、电缆安装；

iii、检查例 2 项目文件 SE_B2_STEP7 下载。

⑦ 运行演示程序：系统连通后，将 PLC 控制按钮旋转至 “RUN” 或 “RUNP” 位置。此时，实验板应该有指示灯演示。按下任意输入键，指示灯演示停，输出指示灯显示与输入键相对应。

⑤ 在 STEP 7 中观察 PROFIBUS 输入输出变化

① 在 STEP 7 中打开项目 SE_B2_STEP7，选中 Blocks，打开变量表 VAT_1，如图 5-30、图 5-31。

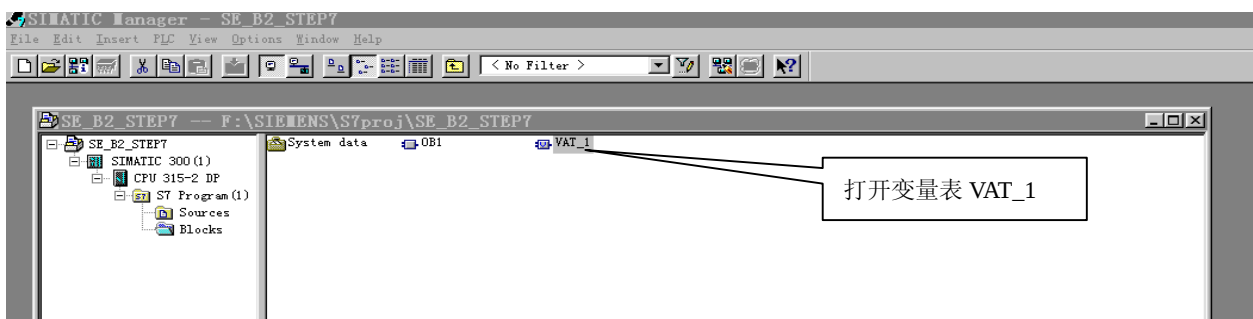


图 5-30 在 STEP 7 中打开项目 SE_B2_STEP7，选中 Blocks，打开变量表 VAT_1

② 可以看到 PROFIBUS 输出第 1、2 个字节 QB1、QB2 的变化，应该和实验板上的输出指示灯 Q0~Q15 一致。这是因为在实验板程序中将 PROFBUS 输出的第 1、2 个字节送给指示灯 Q0~Q15 显示，见例 2：实验板程序清单有如下程序段：

```
steout0_7(pb_out[0]);          /*----- PROFIBUS 输出数据第一个字节送 Q0-Q7 显示 -----*/
steout8_15(pb_out[1]);        /*----- PROFIBUS 输出数据第一个字节送 Q0-Q7 显示 -----*/
```

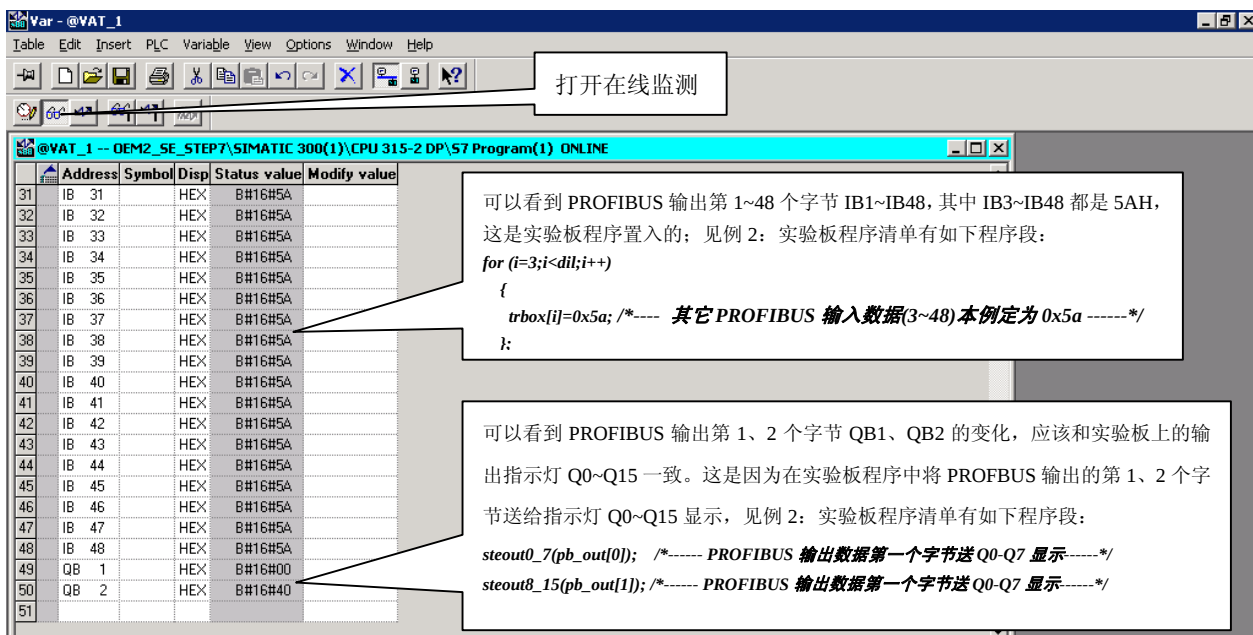


图 5-31 在 STEP 7 中观察 PROFIBUS 输入输出变化

③ 可以看到 PROFIBUS 输出第 1~48 个字节 IB1~IB48，其中 IB3~IB48 都是 5AH。这是因为在实验板的程序中将 PROFIBUS 输入第 3~48 个字节全部置入 5AH。见例 2 实验板程序清单有如下程序段：

```
for (i=3;i<di1;i++)
{
    trbox[i]=0x5a;          /*---- 其它 PROFIBUS 输入数据(3~48)本例定为 0x5a -----*/
};
```

④ 见图 5-32，可以看到：按下实验板任意输入键 I0~I15，IB1、IB2 将随之变化。由于实验板程序将输入键 I0~I15 两个字节作为 PROFIBUS 输入的第 1、2 字节，见例 2 程序中有如下程序段，因此输入键 I0~I15 的数据将对应主站中 IB1、IB2 的数据

```
trbox[1]=di0_7;          /*---- di0_7 是输入键输入 I0~I7 当前值，作为 PROFIBUS 输入的第1 字节 -----*/
trbox[2]=di8_15;         /*---- di8_15 是输入键输入 I8~I15 当前值，作为 PROFIBUS 输入的第2 字节 -----*/
```

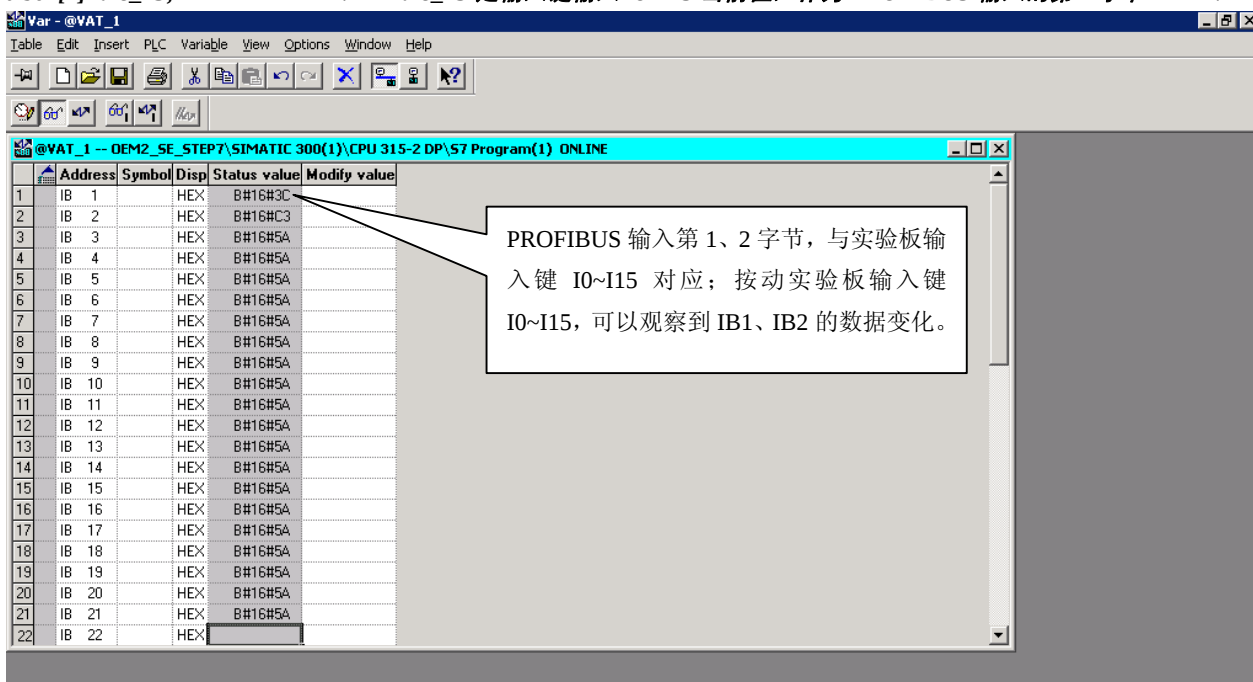


图 5-32 输入键 I0~I15 的数据将对应主站中 IB1、IB2 的数据

(6) PROFIBUS 输入/输出数据地址

在上述实验演示中，为什么 PROFIBUS 输出第 1~48 字节对应 QB1~QB48、PROFIBUS 输入第 1~48 字节对应 IB1~IB48？这是配置项目文件 SE_B2_STEP7/Hardware Config 时系统自动分配的。见图 5-33 进入 Hardware Config，图 5-34 Hardware Config。

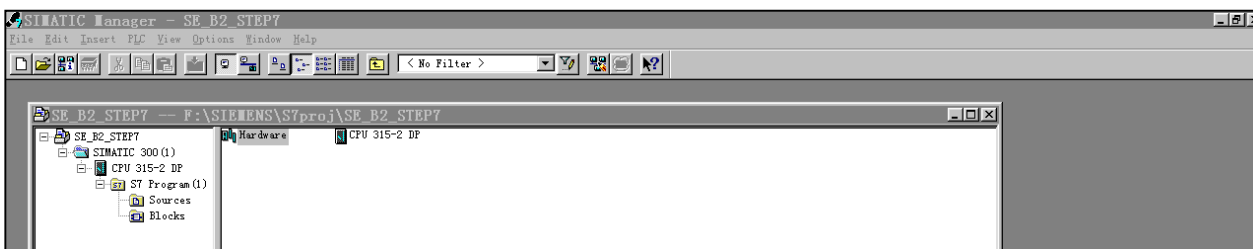


图 5-33 进入 Hardware Config

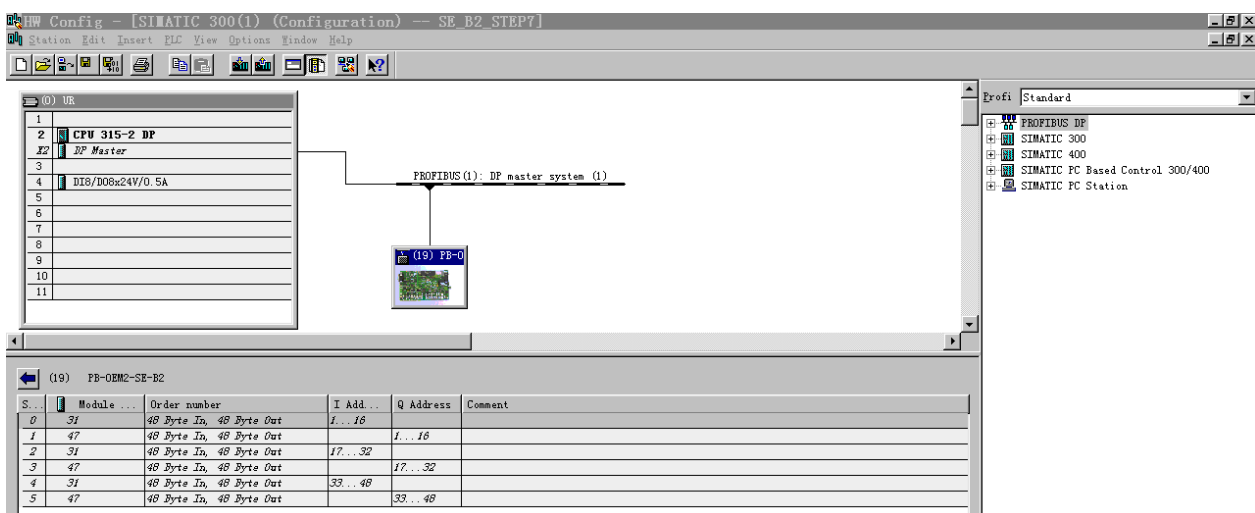


图 5-34 Hardware Config

系统自动为实验板分配了输入地址 IB1~IB48，对应 PROFIBUS 输入第 1~48 字节、分配了输出地址 QB1~QB48，对应 PROFIBUS 输出第 1~48 字节。如果人为将实验板 PROFIBUS 输入地址改为 IB11~IB58，如图 5-35 所示，那么 IB11 将对应 PROFIBUS 输入第 1 个字节、IB58 将对应 PROFIBUS 输入第 48 个字节。同样，如果人为将实验板 PROFIBUS 输出地址改为 QB21~QB68，那么 QB21 将对应 PROFIBUS 输出第 1 个字节，QB68 将对应 PROFIBUS 输出第 48 个字节。

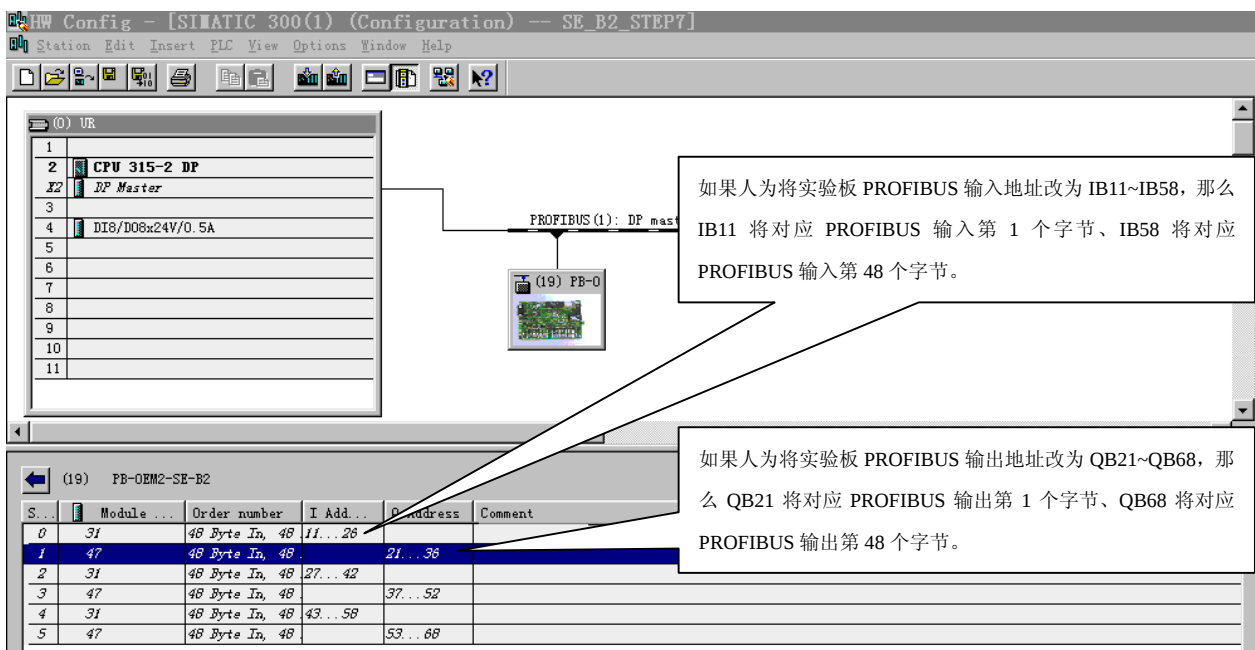


图 5-35 手动设置 PROFIBUS I/O 地址后的对应关系

(7) 自己完成例 2 配置实现系统连通实验

① 确认：GSD 文件 OEM2_B2.GSD 拷贝至 Step7\S7data\gsd\、图标文件 SE_B.bmp 拷贝至 Step7\S7data\nsbmp\。

② 进入  新建项目文件 SE_B2_T1。见图 5-36。

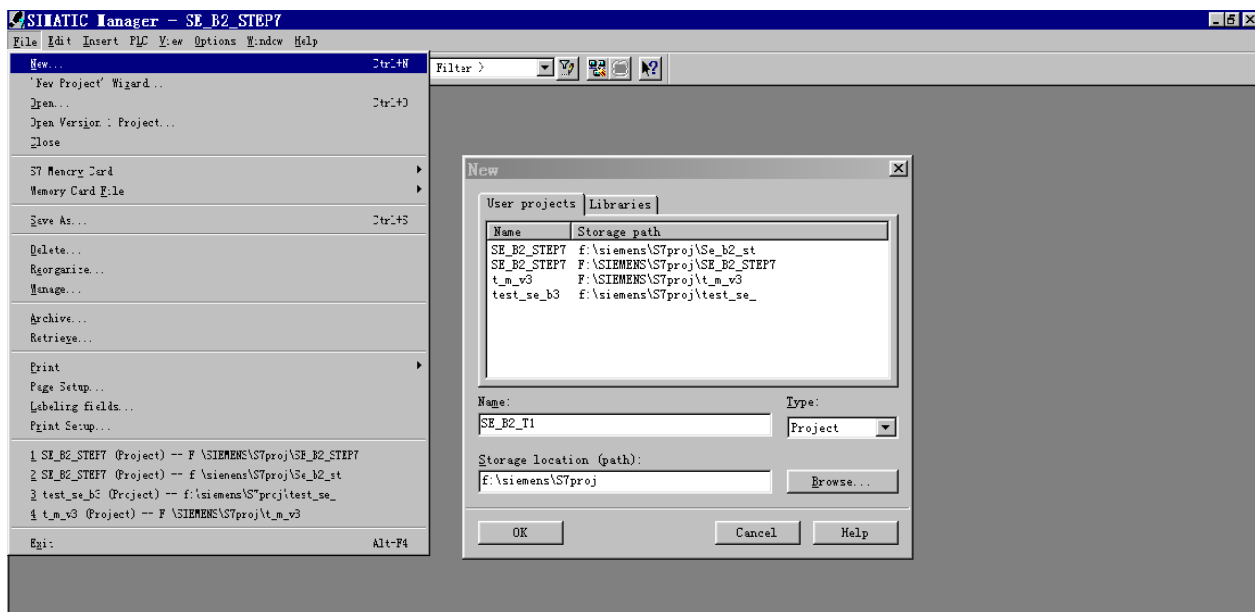


图 5-36 新建项目文件 SE_B2_T1

③ 插入主控制器(本例为 S7-300)，见图 5-37。

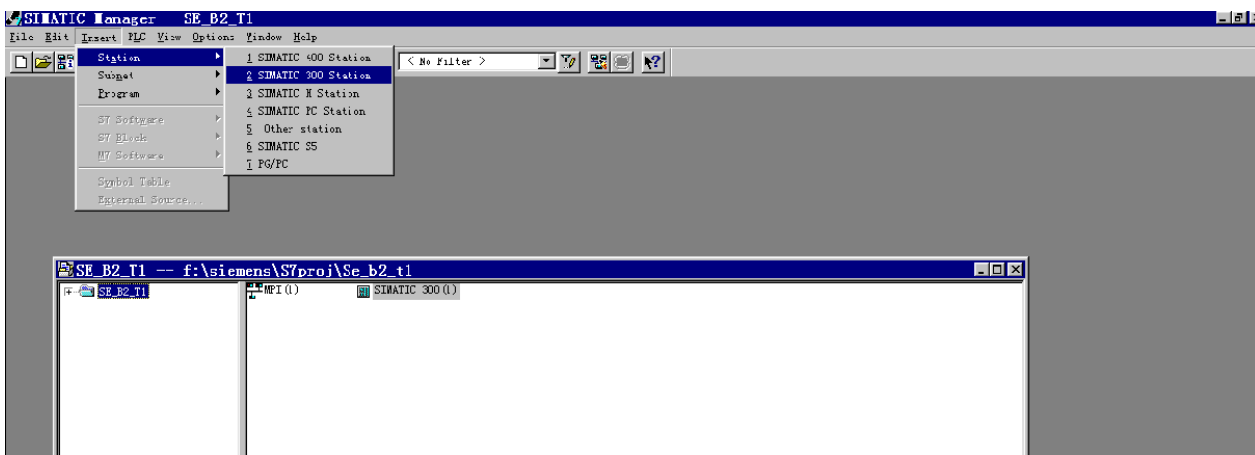


图 5-37 插入主控制器(本例为 S7-300)

④ 进入 Hardware，见图 5-38

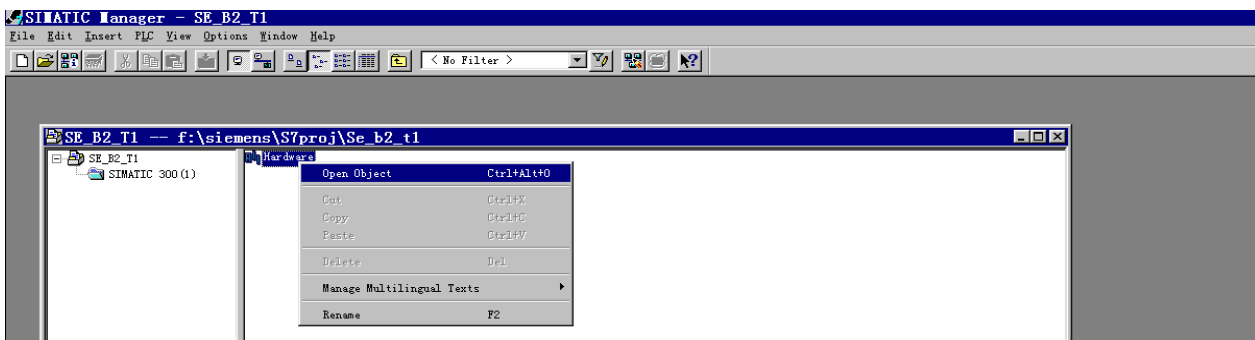


图 5-38 进入 Hardware

⑥ 配置主站，见图 5-39。

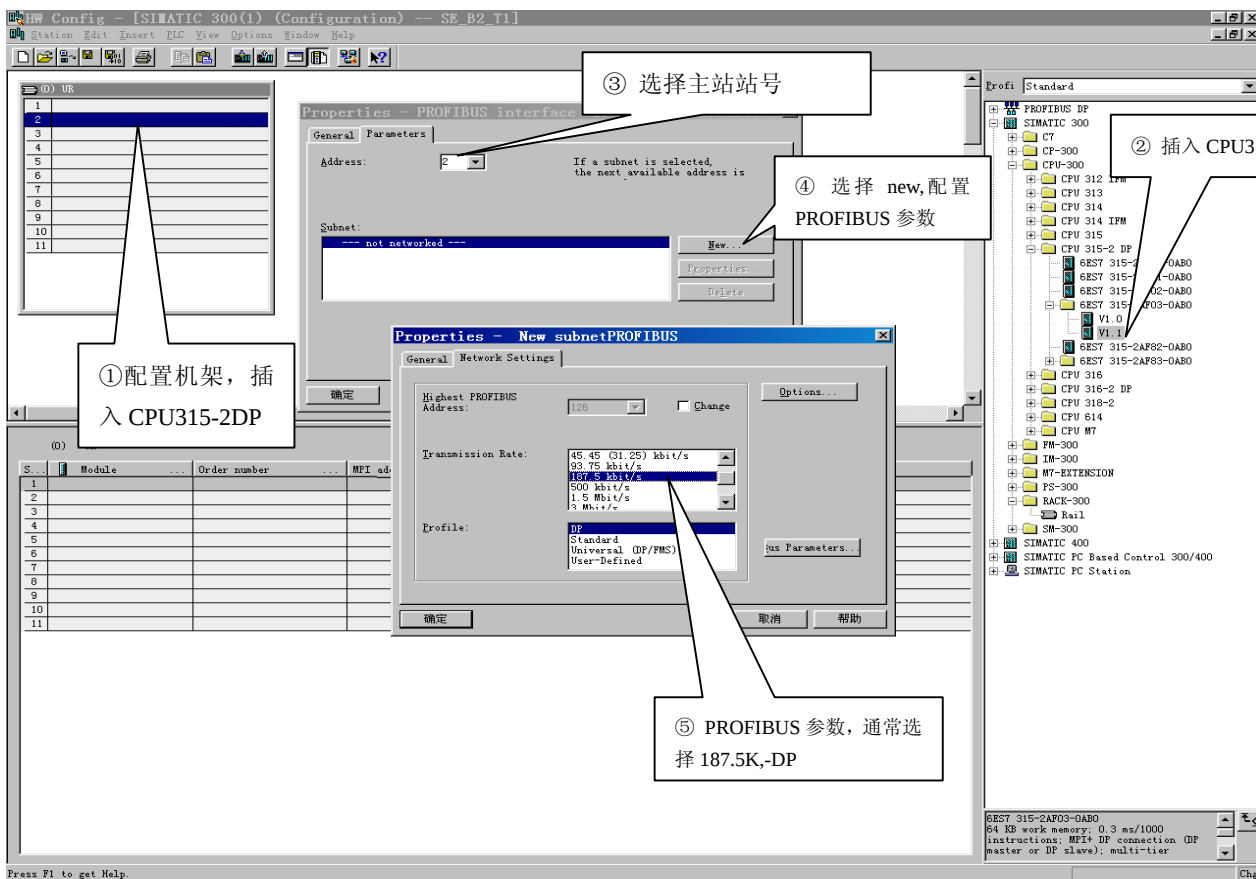


图 5-39 配置主站

⑦ 配置从站 PB-OEM2-SE-B2，见图 5-40，PB-OEM2-SE-B2 所在目录：PROFIBUS DP/Additional Field Devices/ I/O /PB-OEM2/PB-OEM2-SE-B2。

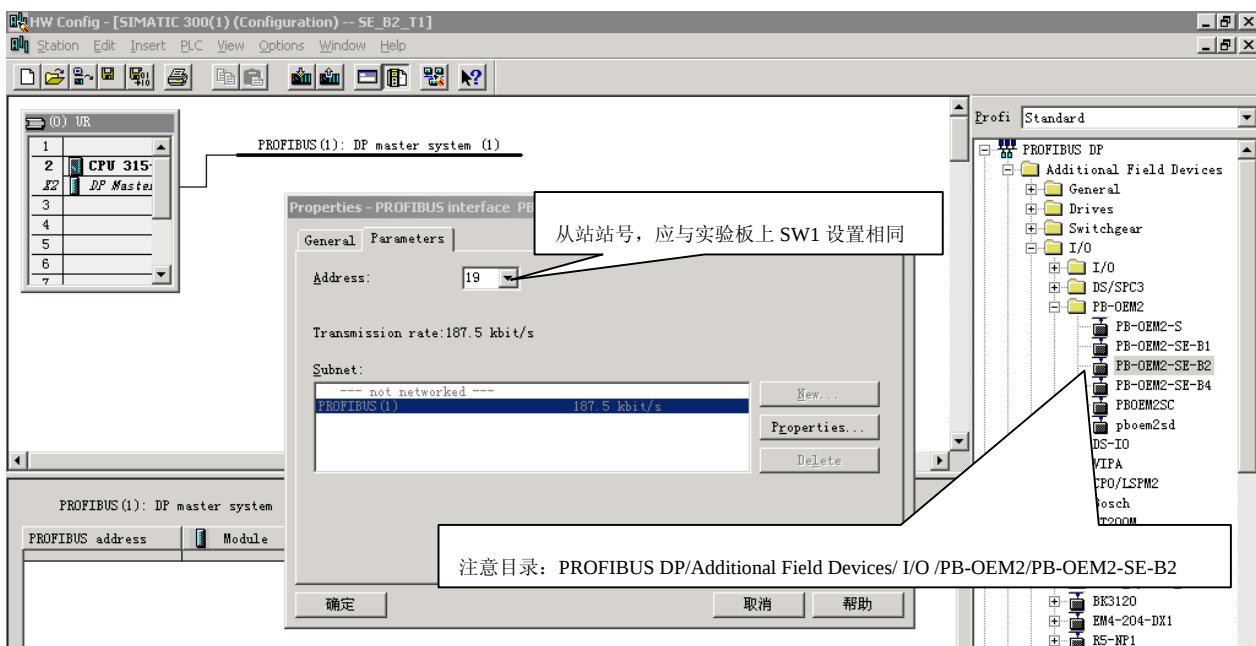


图 5-40 配置从站 PB-OEM2-SE-B2

⑧ 系统Ⅱ配置完毕，编译存盘，见图 5-41，注意系统自动分配的 PROFIBUS I/O 地址。

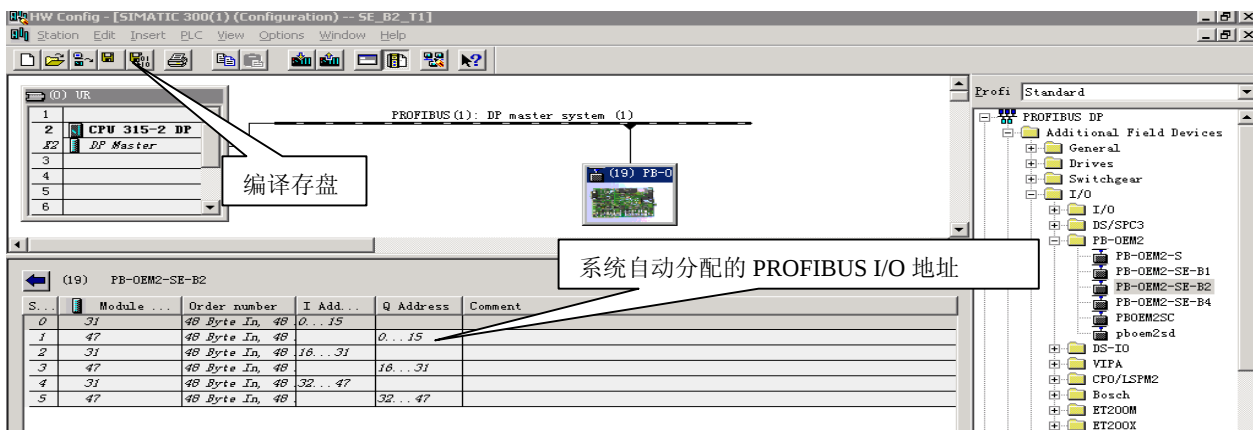


图 5-41 系统Ⅱ配置完毕，编译存盘

⑨ 建立变量表，用于观察 PROFIBUS I/O，见图 5-42、图 5-43。

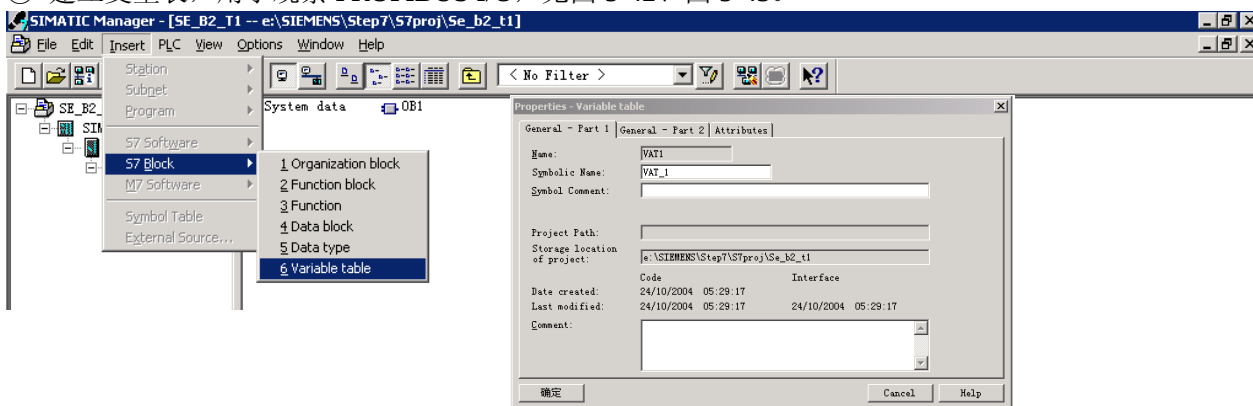


图 5-42 建立变量表

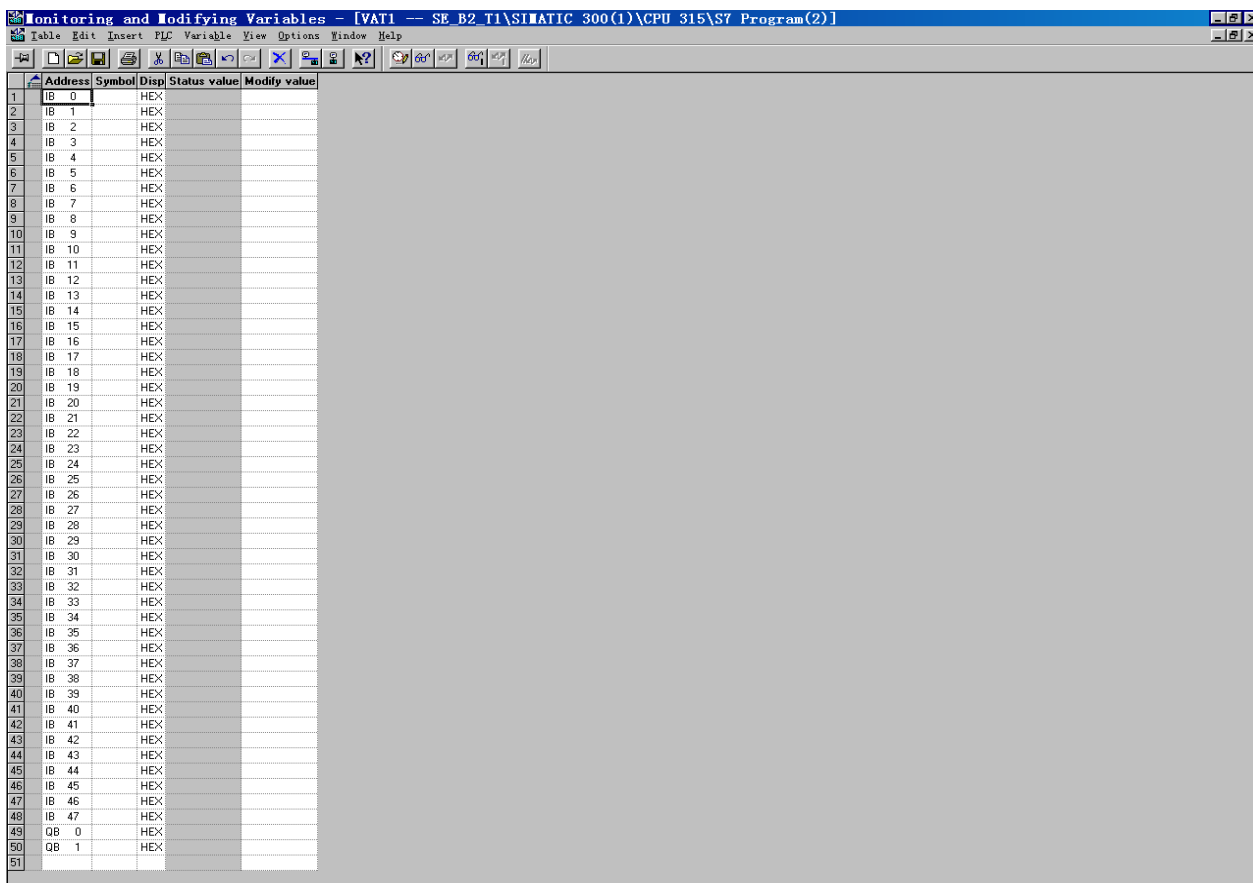


图 5-43 建立变量表 IB0~IB47、QB0~QB1

⑩ 下载项目文件，见图 5-44。



图 5-44 下载项目文件

⑩--① 系统连通后，在线观察变量表 IB0、IB1 跟随实验板按键的变化情况，IB2~IB47 均为 5AH。见图 5-45。这是因为：实验板程序将输入键 I0~I15 两个字节作为 PROFIBUS 输入的第 1、2 字节，将 PROFIBUS 输入第 3~48 个字节全部置入 5AH。见例 2 实验板程序清单有如下程序段：

```
for (i=3;i<di1;i++)
{
    trbox[i]=0x5a;          /*---- 其它 PROFIBUS 输入数据(3~48)本例定为 0x5a -----*/
};
trbox[1]=di0_7;           /*---- di0_7 是输入键输入 I0~I7 当前值，作为 PROFIBUS 输入的第1 字节 -----*/
trbox[2]=di8_15;          /*---- di8_15 是输入键输入 I8~I15 当前值，作为 PROFIBUS 输入的第2 字节 -----*/
```

	Address	Symbol	Display format	Status value	Modify value
1	IB 0		HEX	B#16#1F	
2	IB 1		HEX	B#16#F8	
3	IB 2		HEX	B#16#5A	
4	IB 3		HEX	B#16#5A	
5	IB 4		HEX	B#16#5A	
6	IB 5		HEX	B#16#5A	
7	IB 6		HEX	B#16#5A	
8	IB 7		HEX	B#16#5A	
9	IB 8		HEX	B#16#5A	
10	IB 9		HEX	B#16#5A	
11	IB 10		HEX	B#16#5A	
12	IB 11		HEX	B#16#5A	
13	IB 12		HEX	B#16#5A	
14	IB 13		HEX	B#16#5A	
15	IB 14		HEX	B#16#5A	
16	IB 15		HEX	B#16#5A	
17	IB 16		HEX	B#16#5A	
18	IB 17		HEX	B#16#5A	
19	IB 18		HEX	B#16#5A	
20	IB 19		HEX	B#16#5A	
21	IB 20		HEX	B#16#5A	
22	IB 21		HEX	B#16#5A	
23	IB 22		HEX	B#16#5A	
24	IB 23		HEX	B#16#5A	
25	IB 24		HEX	B#16#5A	
26	IB 25		HEX	B#16#5A	
27	IB 26		HEX	B#16#5A	
28	IB 27		HEX	B#16#5A	
29	IB 28		HEX	B#16#5A	
30	IB 29		HEX	B#16#5A	
31	IB 30		HEX	B#16#5A	
32	IB 31		HEX	B#16#5A	
33	IB 32		HEX	B#16#5A	
34	IB 33		HEX	B#16#5A	
35	IB 34		HEX	B#16#5A	

图 5-45 在线观察变量表 IB0、IB1 跟随实验板按键的变化，IB2~IB47 均为 5AH

⑩--② 强置 QB0~QB1 值，观察实验板输出指示灯的变化，见图 5-46。

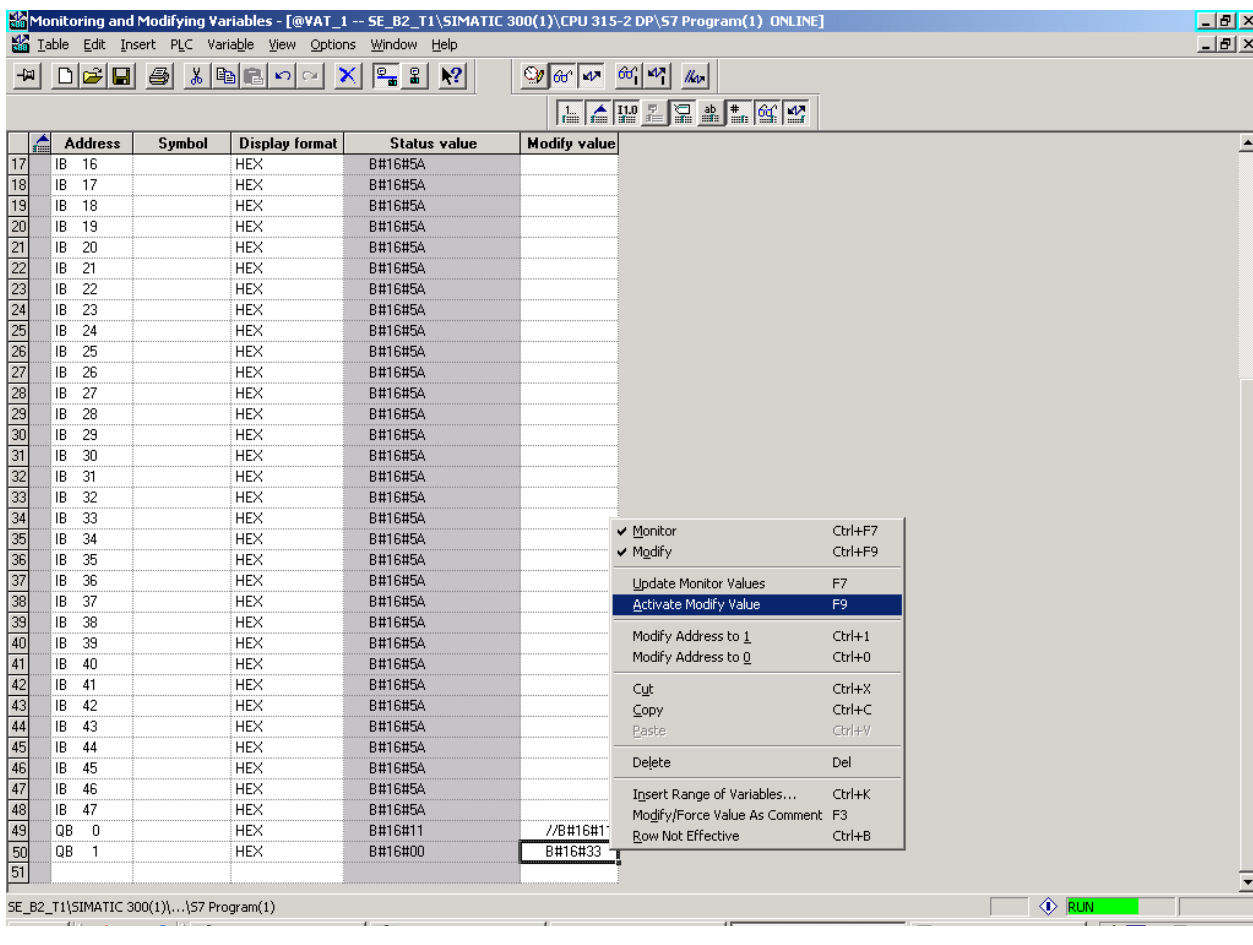


图 5-46 强置 QB0~QB1 值，观察实验板输出指示灯的变化

可以看到 PROFIBUS 输出第 1、2 个字节 QB0、QB1 的变化，应该和实验板上的输出指示灯 Q0~Q15 一致。这是因为在实验板程序中将 PROFIBUS 输出的第 1、2 个字节送指示灯 Q0~Q15 显示，见例 2：实验板程序清单有如下程序段：

```
steout0_7(pb_out[0]);          /*----- PROFIBUS 输出数据第一个字节送 Q0-Q7 显示 -----*/
steout8_15(pb_out[1]);        /*----- PROFIBUS 输出数据第一个字节送 Q0-Q7 显示 -----*/
```

4. 三种调试模式

(1) 调试模式 1：实验板调试模式

① 主要调试功能

建立主站、从站的配置，PROFIBUS 系统的连同，PROFIBUS I/O 通信。用户可以先使用实验板按照本手册完成与 PROFIBUS 主站的连接，完成例 1~3 的实验，然后再将自主开发的用户板与 PROFIBUS 主站的连接。见图 5-47、图 5-48。

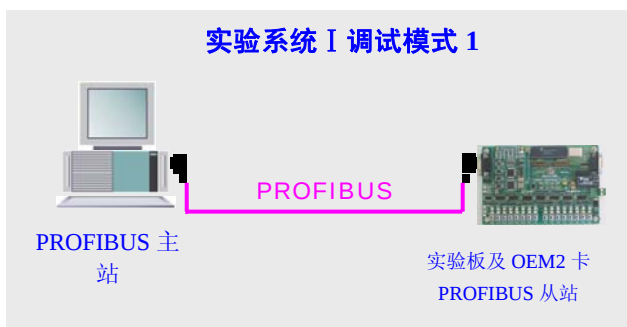


图 5-47 调试模式 1—实验板调试模式

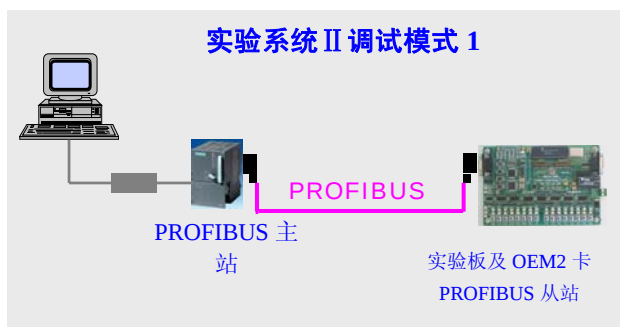


图 5-48 调试模式 1—实验板调试模式

② MODE 开关设置

$\text{MODE}=11000000_{(\text{Bin})}$;

③ 操作方法

本章讲解的全部安装、连通实验、例 2 程序演示都是调试模式 1—实验板调试模式。

(2) 调试模式 2: OEM2 接口板调试模式

① 主要调试功能

通过 PC 机可以监测实验板 CPU 与 OEM2 接口板的通信数据,可检验初始化数据,PROFIBUS I/O 数据,用户参数等数据的正确性。这是检查、调试 OEM2 接口板的主要方法。见图 5-49、图 5-50 及图 5-51,使用 PC 机监测实验板 CPU 与 OEM2 接口板的通信数据。

② MODE 开关设置

$\text{MODE}=11000010_{(\text{Bin})}$;

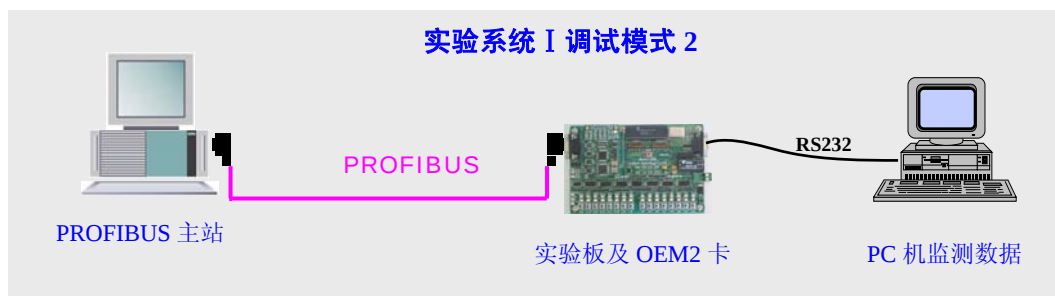


图 5-49 调试模式 2—OEM2 接口板调试模式

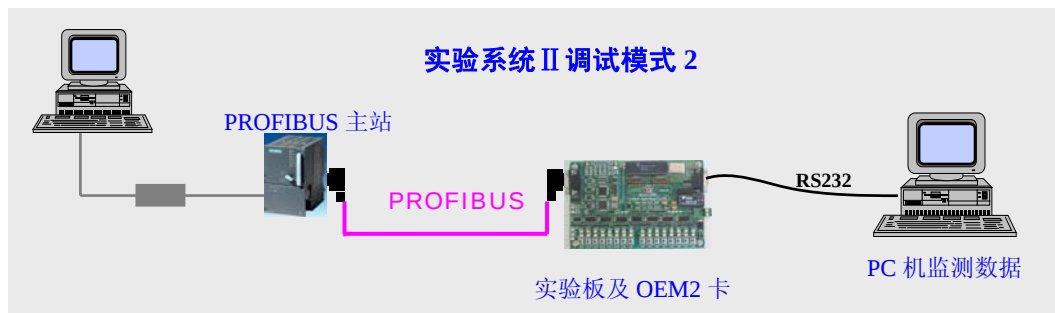
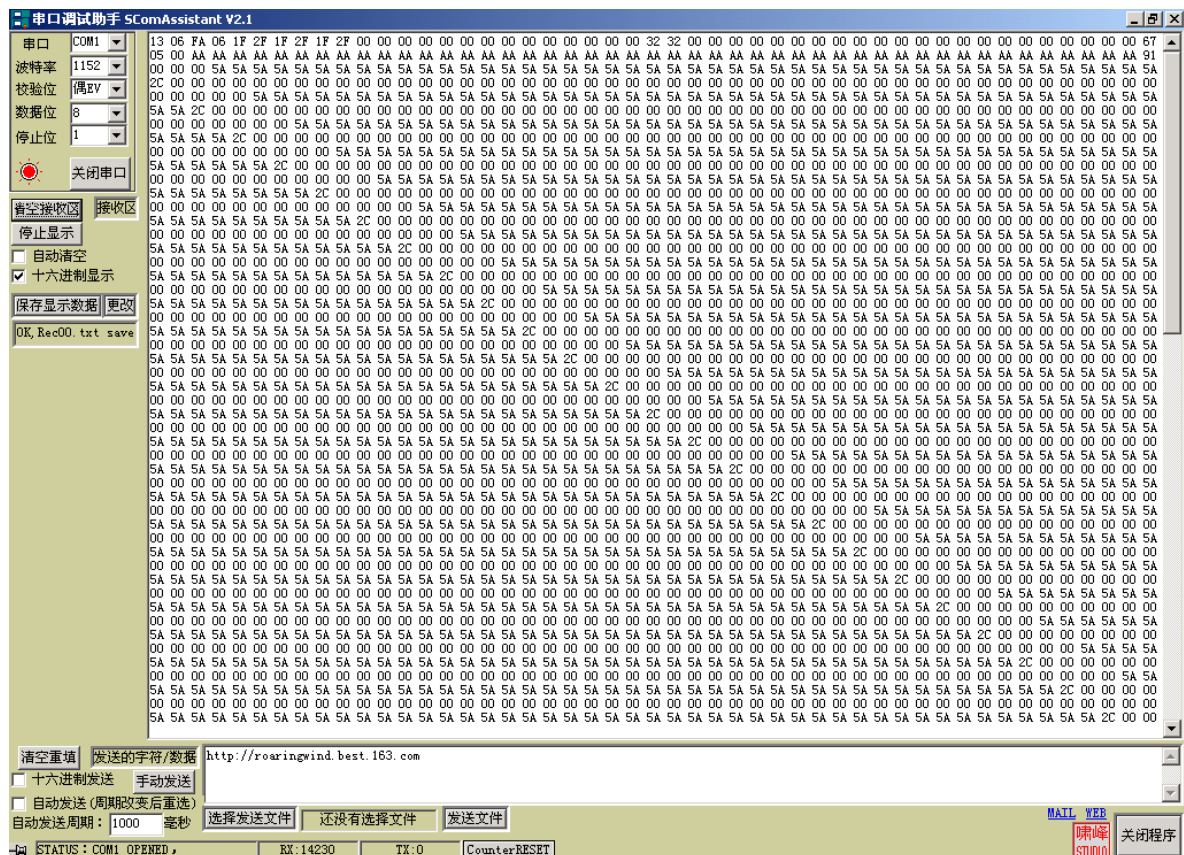


图 5-50 调试模式 2—OEM2 接口板调试模式

③ 操作方式

II、按照“图 5-51 使用 PC 机监测实验板 CPU 与 OEM2 接口板的通信数据”组成系统，本系统包括一根 RS232 电缆。

IV、观察初始化数据：可以不连通主站，打开实验板，观察“串口测试”程序接收数据，如图 5-52



“保存显示数据”可以详细分析 OEM2 与实验板初始化及数据交换过程，参考本手册“第四章 例 2、

实验板程序清单”及《PB-OEM2-SE 产品手册》。

①实验板初始化 SE 接口板数据:

13 06 FA 06 1F 2F 1F 2F 1F 2F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 32 32 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 67

注解: 站号=13H、ID 号=06FAH、I/O 配置数据长度=06、I/O 配置数据=1F, 2F, 1F, 2F, 1F, 2F、用户参数长度=0、发送数据长度=32H、接收数据长度=32H、校验和=67H;

②SE 接口板回答初始化成功报文数据:

05 00 AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA 91

注解：波特率号=05H、错误号=00H、校验和=91H

③实验板发送数据交换报文:

[illegible]

注解: req com=0, 请求接口回答PROFIBUS 输出数据:

PROFIBUS 输入 1~2=00, 00, 与实验板输入键状态有关;

PROFIBUS 输入 3~48=5AH, 与实验板程序有关;

校验和=2CH

④SE 接口板回答 PROFIBUS 输出数据:

[illegible]

注解: AB status=00, 即: A 格式、接口板接收用户板报文无错;

PROFIBUS 输出 1~48 字节=00, 由于主站没有接通, 因此得到的 PROFIBUS 输出数据为 00

⑤ 回 ③

[illegible]

V、先开主站，再打开实验板；观察“串口测试”程序接收数据，用户可以看到 PROFIBUS 输出数据。

VI、用例 3 实验，可以看到用户参数数据。

(3) 实验板程序下载

① 主要调试功能

可将实验板 C 源程序编译、连接生成 .HEX 目标程序，比如系统提供的例 1（B1.HEX）、例 2（B2.HEX）、例 3（B4.HEX）通过软件 WINSIP 下载到实验板。用户可以自行修改实验板的程序，实现不同 PROFIBUS 配置的连接，见图 5-53。

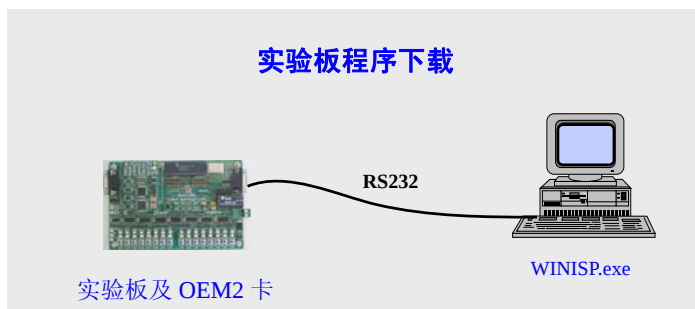


图 5-53 实验板程序下载

② MODE 开关设置

MODE=00000111_(Bin);

③ 操作方式

见附件 A：下载实验板程序操作指南及下载线制作方法

(4) 调试模式 3：PC 机调试模式

① 主要调试功能

“调试模式 3” 是使用 PC 机替代实验板 CPU，见图 5-54、图 5-55 所示。PC 机运行 OEM2 调试软件：oem2_tools.exe，实现机与 OEM2 接口板的数据交换。这种调试模式避免下载程序（至实验板 CPU），PC 机上可方便设定不同初始化数据和 PROFIBUS 输入数据，也可以监测主站的 PROFIBUS 输出数据、用户参数数据，是学习、调试 OEM2 接口板的另一种方法。

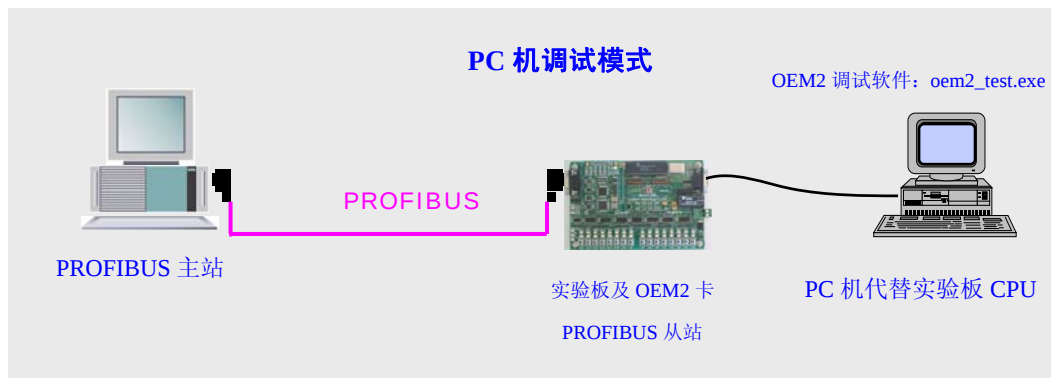


图 5-54 调试模式 3—PC 机调试模式

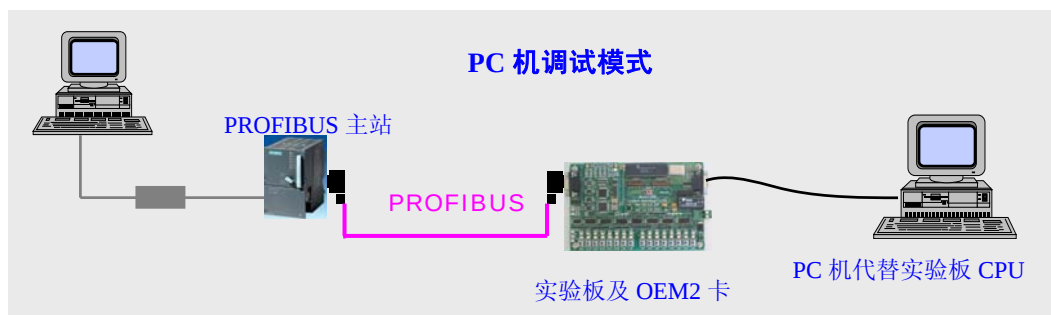


图 5-55 调试模式 3—PC 机调试模式

② MODE 开关设置

MODE=00100000_(Bin)

③ 操作方式

I、按图 5-54 或图 5-55 连接系统；

II、主站应先做好配置，注意：主站配置的 GSD 文件与下面的初始数据应该一致。主站部分可以先启动

或随后启动，取决于调试目的,实验板电源上电。注意：此时实验板 CPU 还处于运行状态，但串口已经和 OEM2-SE 接口断开。因此“PC 调试模式”与实验板 CPU 程序（甚至 CPU、晶振的有无）无关。

III、PC 机上安装 OEM2_tools 软件（OEM2_ToolsV2.exe 安装），启动 OEM2_tools.exe，见图 5-56。

IV、在“属性”中设置串口，然后“打开端口”，见图 5-56。

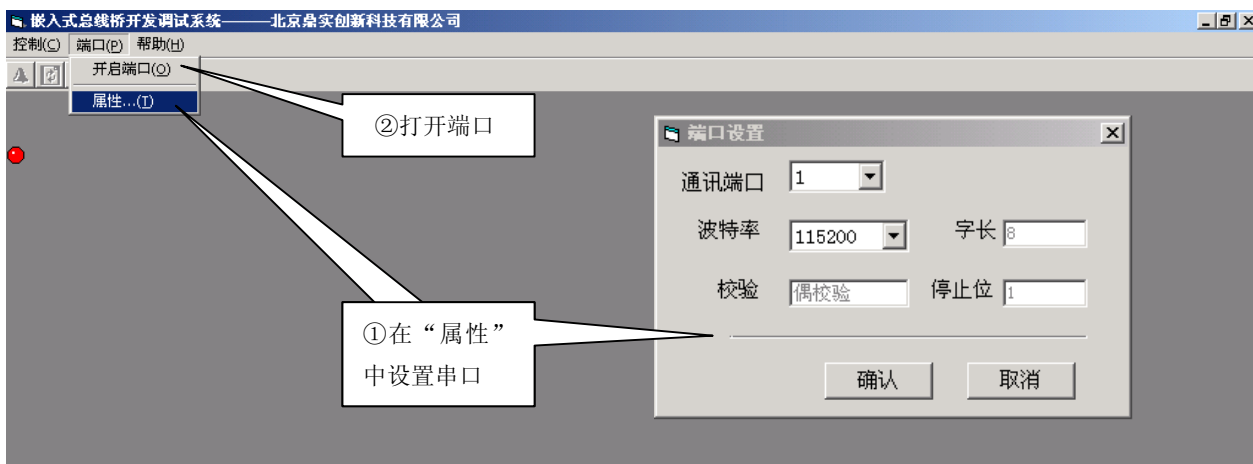


图 5-56 启动 OEM2_toolsV2.exe、在“属性”中设置串口，然后“打开端口”

V、首先进入初始化，见图 5-57；

i、左边出始化报文缺省数据是按照例 2 内容组织，用户也可以自行修改。

ii、按“数据确定”自动计算“校验和”



图 5-57 首先完成初始化

iii、按“发送数据”，观察右边“接收报文”。

▼如果根本没有回答（从实验板通信指示灯上也可以判断），检查波特率设置、OEM2-SE 接口上的波特率设置开关 SW 及初始化报文中的“站号”与实验板上“站号”设置是否一致。

▼如果“接收报文”是初始化成功报文，此时打开主站（或主站已经打开），那么，实验板上 OEM2-SE 接口板上的 PBF 红灯应该灭，表示主站已经和实验板连通。否则，检查主站配置（GSD 文件）、初始化报文及回答报文数据和错误代码。

VI、进入数据交换

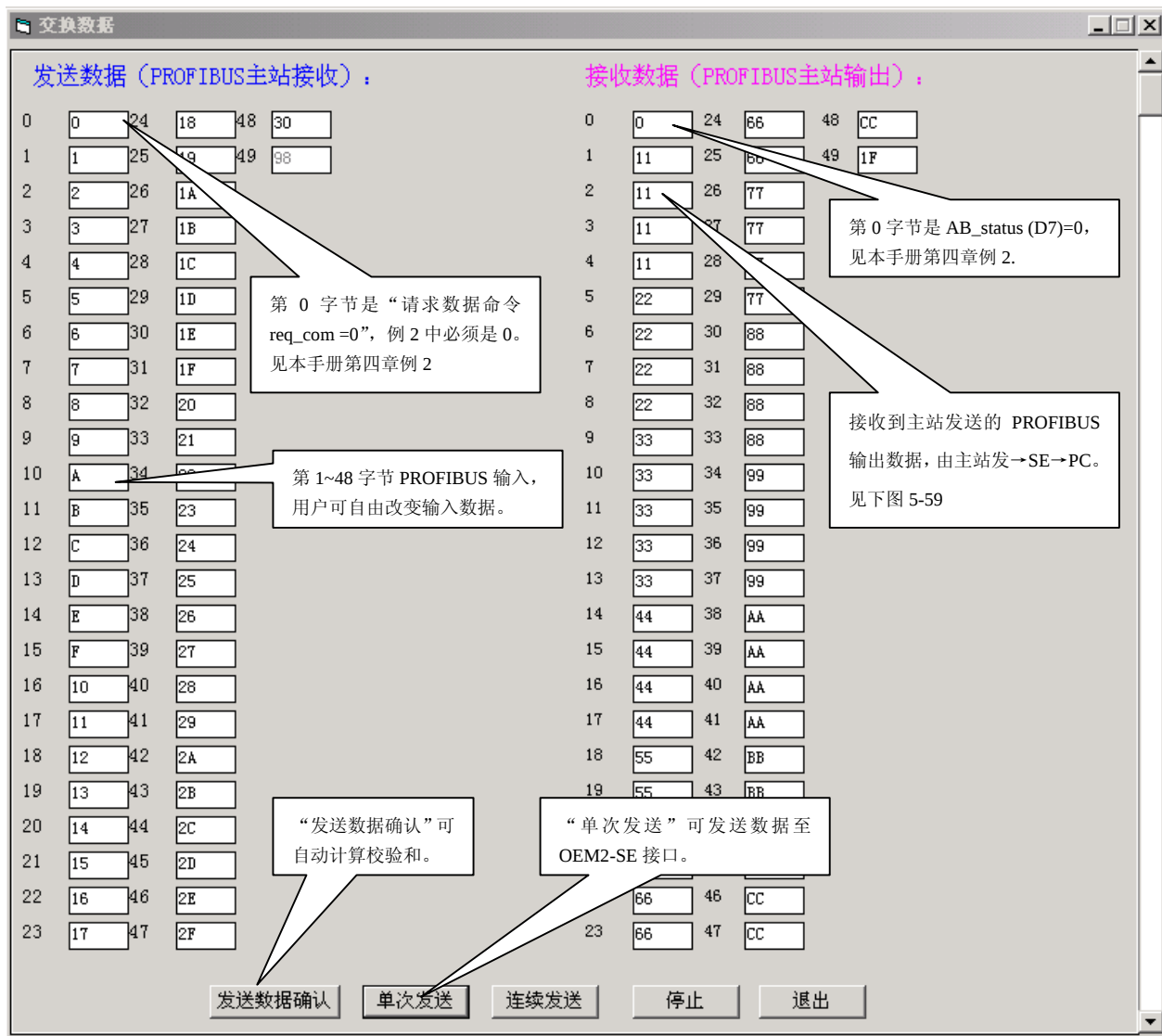


图 5-58 单次数据交换

i、单次数据交换：见图 5-58。

▼发送数据：第 0 字节是“请求数据命令 req_com=0”，例 2 中必须是 0。见本手册第四章例 2，第 1~48 字节 PROFIBUS 输入，用户可自由改变输入数据。“发送数据确认”可自动计算校验和。“单次发送”可发送数据至 OEM2-SE 接口。

▼观察回答报文：第 0 字节是 AB_status (D7)=0；见本手册第四章例 2；主站发送的 PROFIBUS 输出数

据，由主站发→SE→PC。

▼对系统 I，观察 WinCC 主站数据，如图 5-59。

**PROFIBUS 数据接收区
(48 Bytes)**

输入字节0:	1	输入字节24:	19
输入字节1:	2	输入字节25:	1A
输入字节2:	3	输入字节26:	1B
输入字节3:	4	输入字节27:	1C
输入字节4:	5	输入字节28:	1D
输入字节5:	6	输入字节29:	1E
输入字节6:	7	输入字节30:	1F
输入字节7:	8	输入字节31:	20
输入字节8:	9	输入字节32:	21
输入字节9:	A	输入字节33:	22
输入字节10:	B	输入字节34:	23
输入字节11:	C	输入字节35:	24
输入字节12:	D	输入字节36:	25
输入字节13:	E	输入字节37:	26
输入字节14:	F	输入字节38:	27
输入字节15:	10	输入字节39:	28
输入字节16:	11	输入字节40:	29
输入字节17:	12	输入字节41:	2A
输入字节18:	13	输入字节42:	2B
输入字节19:	14	输入字节43:	2C
输入字节20:	15	输入字节44:	2D
输入字节21:	16	输入字节45:	2E
输入字节22:	17	输入字节46:	2F
输入字节23:	18	输入字节47:	30

**PROFIBUS 数据发送区
(48 Bytes)**

输出字节0:	11	输出字节24:	66
输出字节1:	11	输出字节25:	77
输出字节2:	11	发送字节26:	77
输出字节3:	11	发送字节27:	77
输出字节4:	22	发送字节28:	77
输出字节5:	22	发送字节29:	88
输出字节6:	22	发送字节30:	88
输出字节7:	22	发送字节31:	88
输出字节8:	33	发送字节32:	88
输出字节9:	33	发送字节33:	99
输出字节10:	33	发送字节34:	99
输出字节11:	33	发送字节35:	99
输出字节12:	33	发送字节36:	99
输出字节13:	44	发送字节37:	AA
输出字节14:	44	发送字节38:	AA
输出字节15:	44	发送字节39:	AA
输出字节16:	44	发送字节40:	AA
输出字节17:	55	发送字节41:	BB
输出字节18:	55	发送字节42:	BB
输出字节19:	55	发送字节43:	BB
输出字节20:	55	发送字节44:	BB
输出字节21:	66	发送字节45:	CC
输出字节22:	66	发送字节46:	CC
输出字节23:	66	发送字节47:	CC

[返回](#)

主站接收的 PROFIBUS 输入数据，由 PC→SE→WinCC, 见上图 5-58

用户可自行键入 PROFIBUS 输出数据，由 WinCC→SE→PC, 见上图 5-58

图 5-59 单次数据交换，观察 WinCC 主站数据

▼对系统 II，观察 PLC 主站数据，如图 5-60。

Address	Symbol	Disp	Status	value	Modify value
1	IB 1	HEX	B#16#01		
2	IB 2	HEX	B#16#02		
3	IB 3	HEX	B#16#03		
4	IB 4	HEX	B#16#04		
5	IB 5	HEX	B#16#05		
6	IB 6	HEX	B#16#06		
7	IB 7	HEX	B#16#07		
8	IB 8	HEX	B#16#08		
9	IB 9	HEX	B#16#09		
10	IB 10	HEX	B#16#0A		
11	IB 11	HEX	B#16#0B		
12	IB 12	HEX	B#16#0C		
13	IB 13	HEX	B#16#0D		
14	IB 14	HEX	B#16#0E		
15	IB 15	HEX	B#16#0F		
16	IB 16	HEX	B#16#10		
17	IB 17	HEX	B#16#11		
18	IB 18	HEX	B#16#12		
19	IB 19	HEX	B#16#13		
20	IB 20	HEX	B#16#14		
21	IB 21	HEX	B#16#15		
22	IB 22	HEX	B#16#16		
23	IB 23	HEX	B#16#17		
24	IB 24	HEX	B#16#18		
25	IB 25	HEX	B#16#19		
26	IB 26	HEX	B#16#1A		
27	IB 27	HEX	B#16#1B		
28	IB 28	HEX	B#16#1C		
29	IB 29	HEX	B#16#1D		
30	IB 30	HEX	B#16#1E		
31	IB 31	HEX	B#16#1F		
32	IB 32	HEX	B#16#20		
33	IB 33	HEX	B#16#21		
34	IB 34	HEX	B#16#22		
35	IB 35	HEX	B#16#23		

主站接收的 PROFIBUS 输入数据，由 PC→SE→WinCC, 见上图 5-58

图 5-60 单次数据交换、观察 PLC 主站数据

ii、连续数据交换：见图 5-61、图 5-62。

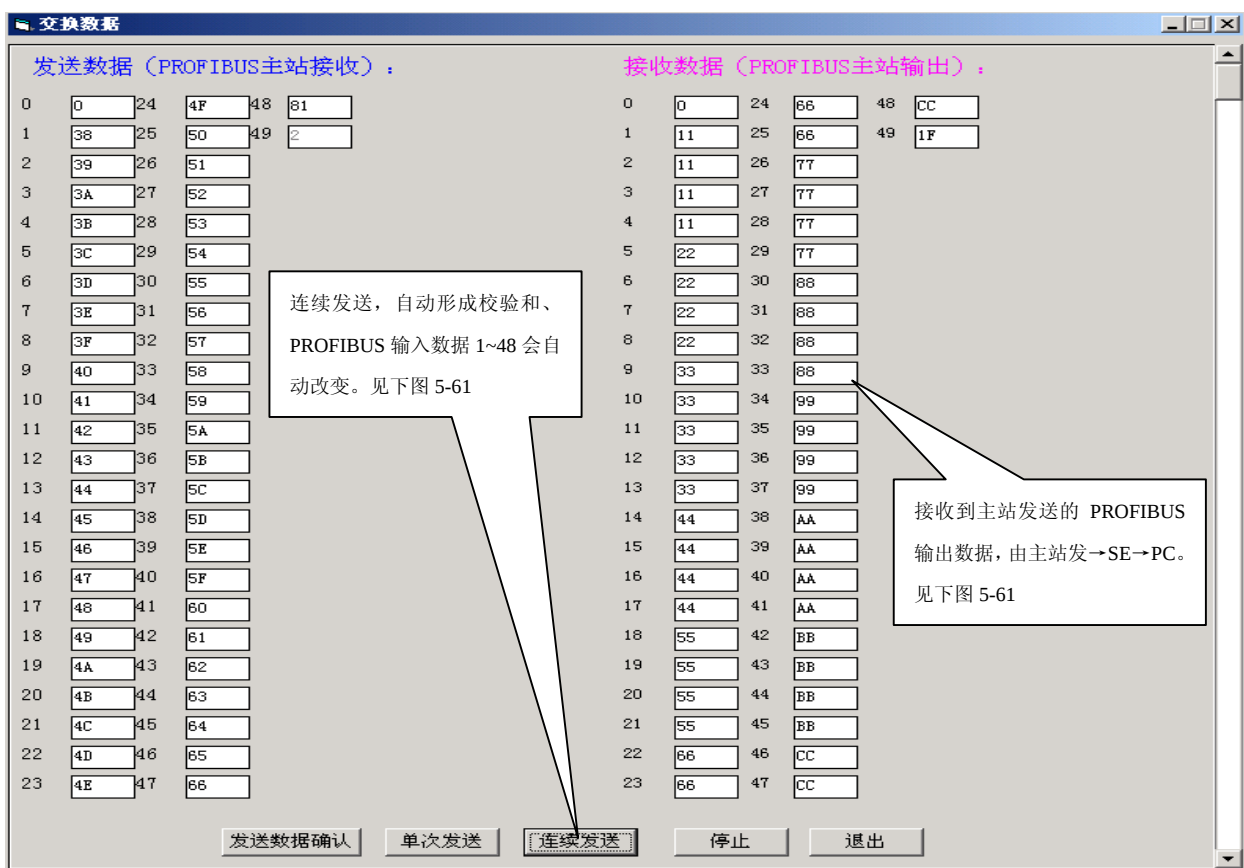


图 5-61 连续数据交换、自动形成校验和、PROFIBUS 输入数据 1~48 会自动改变

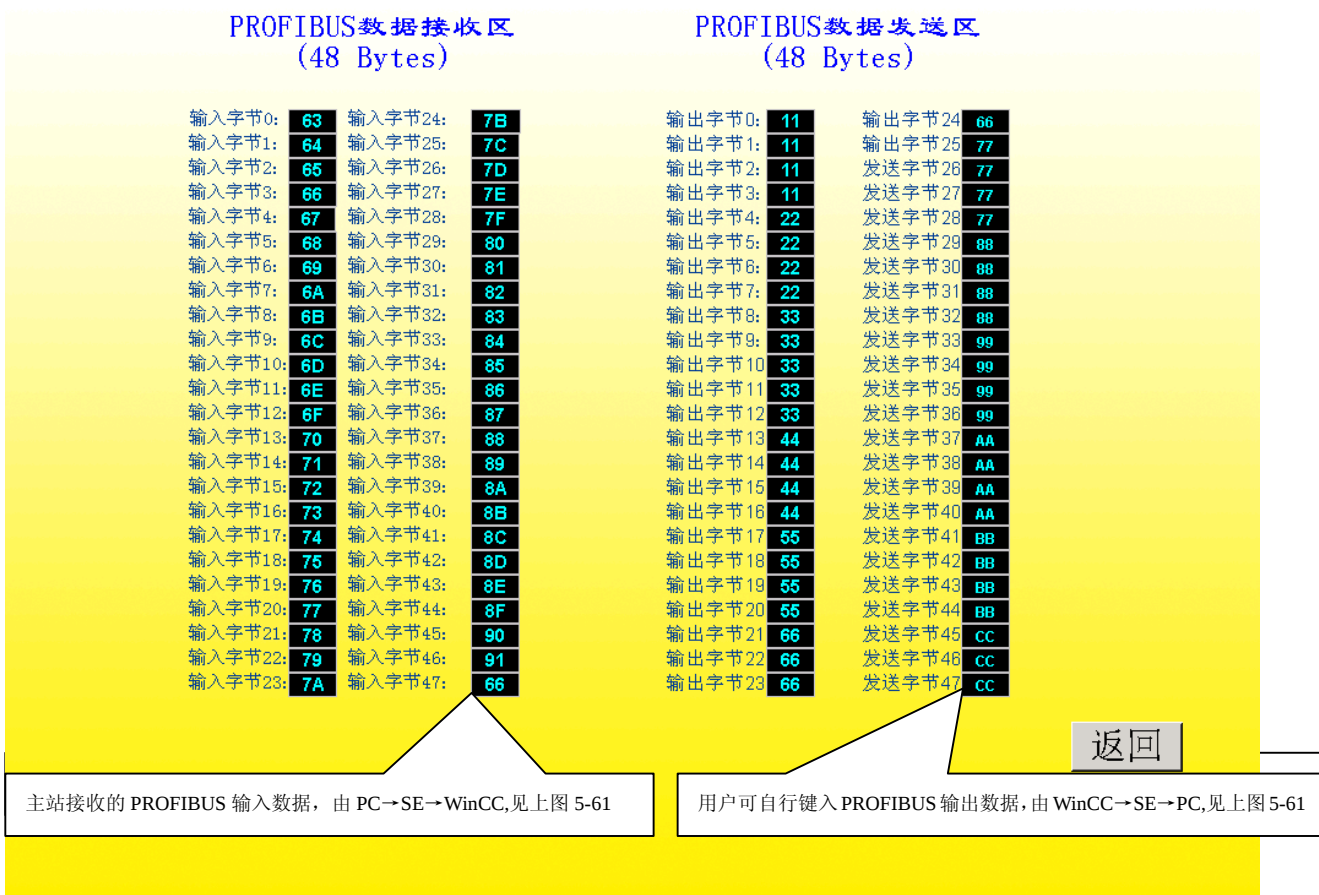


图 5-62 连续数据交换、观察 WinCC 主站数据

附件 A：下载实验板程序操作指南及下载线制作方法

本实验板支持对具有 ISP (IN-SYSTEM PROGRAMMING)功能的 CPU，进行程序在线式写入（如 PHILIPS 的 P89V51RD2）。

本指南以 P89V51RD2 为例，但不同厂家的操作方法各有异同，请参见其产品手册中的详细介绍，勿完全效仿。（不具有 ISP 功能的 CPU 芯片请勿使用本功能，以免芯片或电路板的烧毁）。若有硬件改动，请根据提供的原理图自行飞线处理。

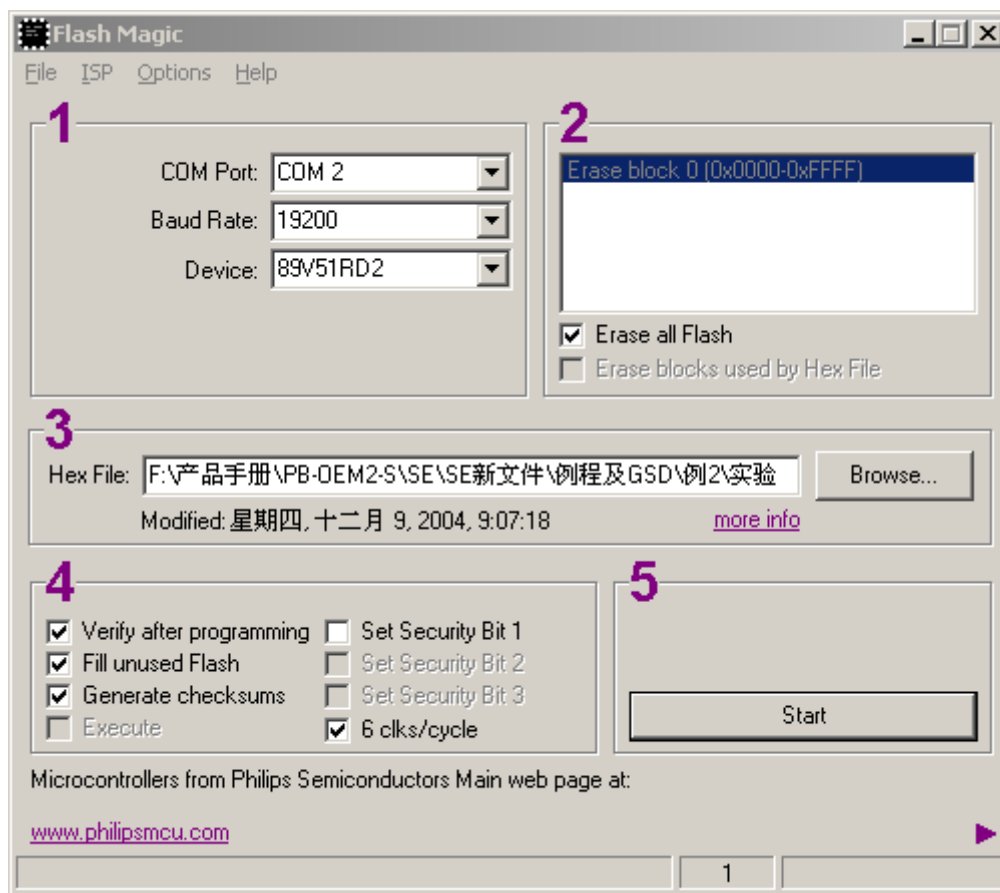
1. 实验板的下载模式选择

实验板 MODE 开关 SW2 设置：MODE=00000111_(Bin)；实验板上电。由于实验板将 CPU 置于编程模式，因此上电后 CPU 不执行程序，也就没有输出指示灯流水演示。

2. Flash Magic 软件的程序下载操作

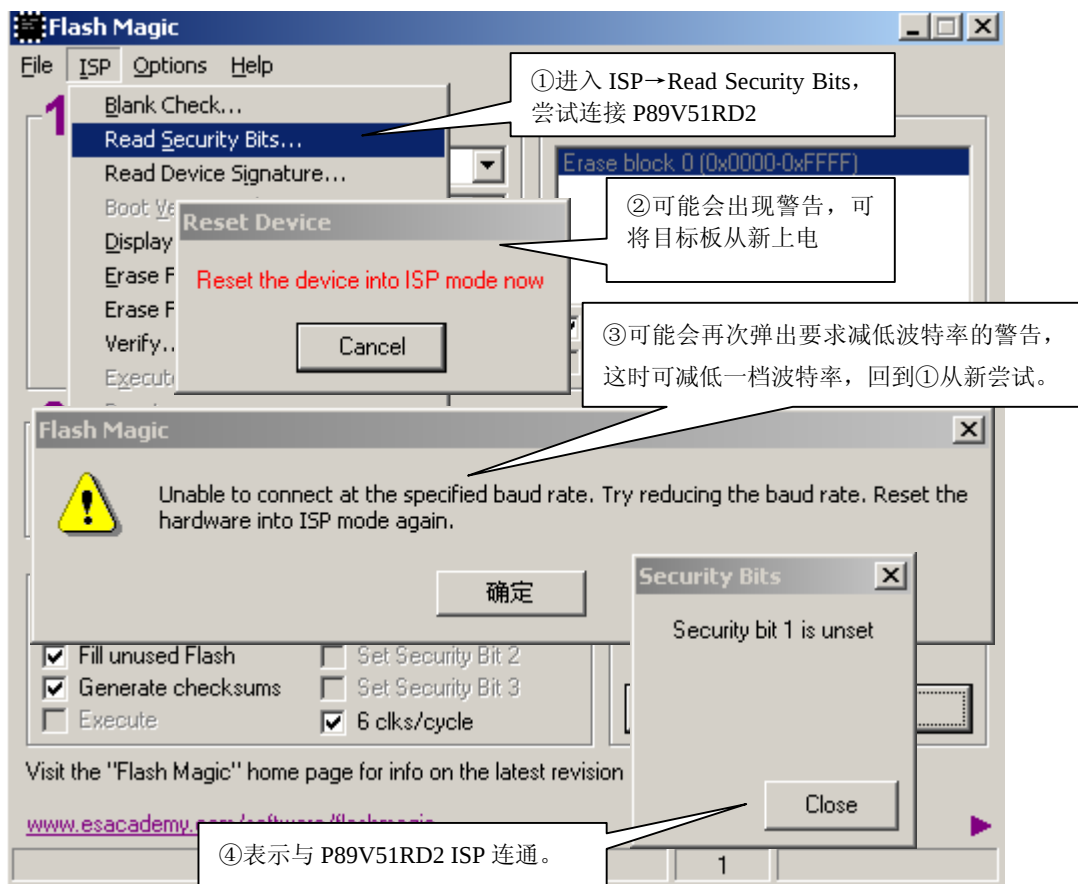
Flash Magic 是专为 PHILIPS 公司 P89X51XXX 系列产品所提供的 ISP 下载软件（见产品光盘）。将 Flash Magic 装入 PC 运行见下图附 1。

(1) 为 ISP 选定 PC 机侧所用的串口 COM1-COM4；指定所用微控制器的型号 P89V51RD2。可选择 Baud Rate=19200 作为尝试波特率。



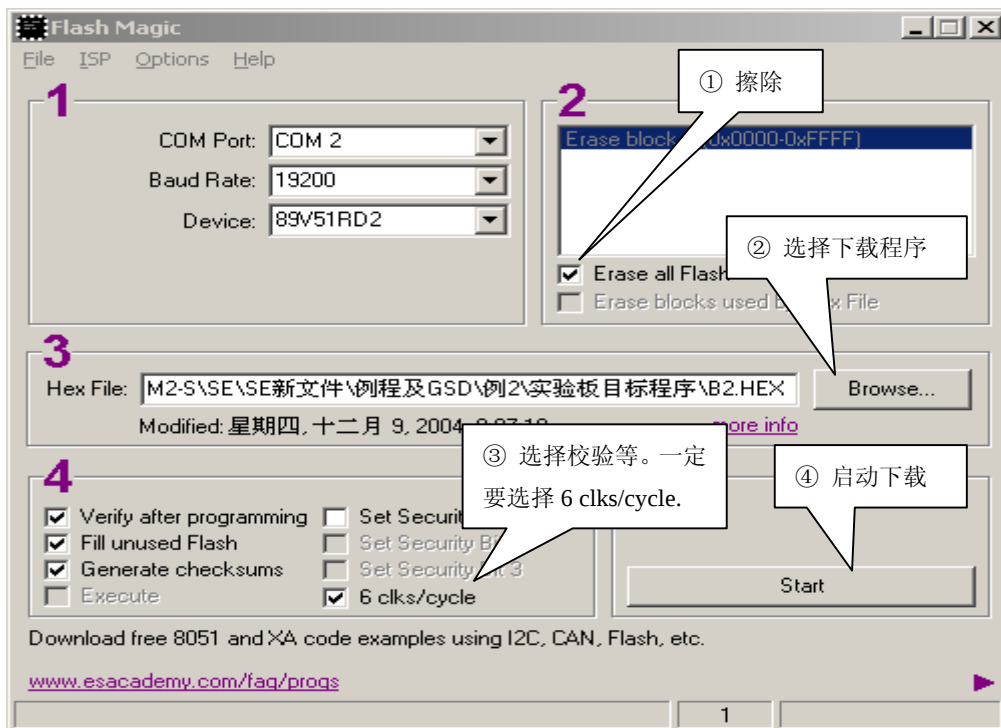
附图 1

(2) 进入 ISP→Read Security Bits, 尝试连接 P89V51RD2。可能会出现警告:“Reset the device into ISP mode now”; 可将目标板从新上电, 可能会再次弹出要求减低波特率的警告, 这时可减低一档波特率, 回到 Read Security Bits 从新尝试, 直到出现“Security bit 1 is unset”表示与 P89V51RD2 ISP 连通。



附图 2

(3) 选择功能选项并下载程序



附图 3

3. 恢复实验板运行状态

- (1) 恢复**实验板** **MODE 开关 SW2 设置**: **MODE**=11000000_(Bin) (调试模式 1) 或 **MODE**=11000010_(Bin) (调试模式 2);
- (2) 实验板上电。上电后 CPU 将执行程序, 有输出指示灯流水演示。
- (3) 按调试模式 1 或调试模式 2 调试下载的实验板程序。

4. 下载线的制作及连接方法

本系统包括一根 RS232 电缆, 可用做下载线使用。如果需要, 用户可自制 RS-232 电缆; 方法如下:

- (1) **材料**: D 型九孔插头 (female) 两个, 3 芯电缆线。
- (2) 将每个插头按 2-2, 3-3, 5-5 的规则用电缆线焊在一起, 其中任一端可接计算机的 COM 口, 另一端接连本实验板的右侧, 九针 D 型插座。

!!!注意: 在进行实验板模式切换时, 请务必下电, 以免造成不必要的损失。

!!!注意: P89V51RD2 芯片的程序写入也可由其他的编程器来完成, 但对于特殊寄存器的操作, 各个编程器有所不同, 使用时要尤为注意。具体的芯片擦除、加密及 IAP 等操作在此不作赘述, 详情参见 P89V51RD2 手册。

北京鼎实创新科技有限公司

现场总线 **PROFIBUS**（中国）技术资格中心

电话：010-82078264、010-62054940

传真：010-82078264

地址：北京德胜门外教场口 1 号，一号楼 401~404 邮编：100011

Web: www.c-profibus.com.cn

Email: tangjy@c-profibus.com.cn