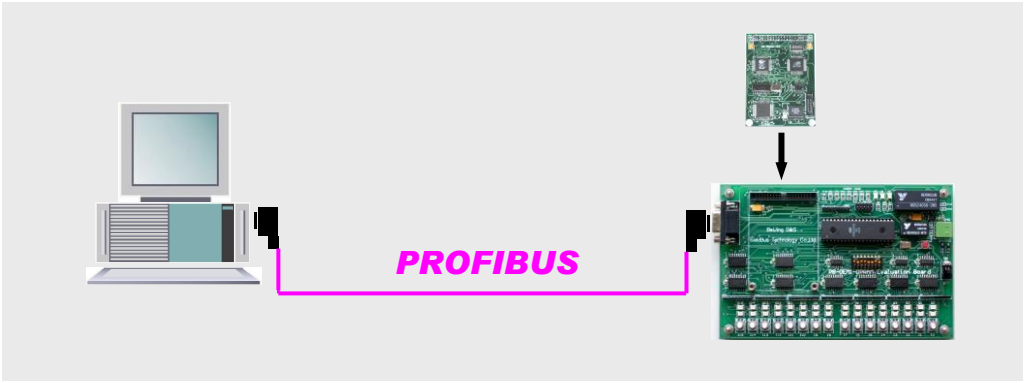
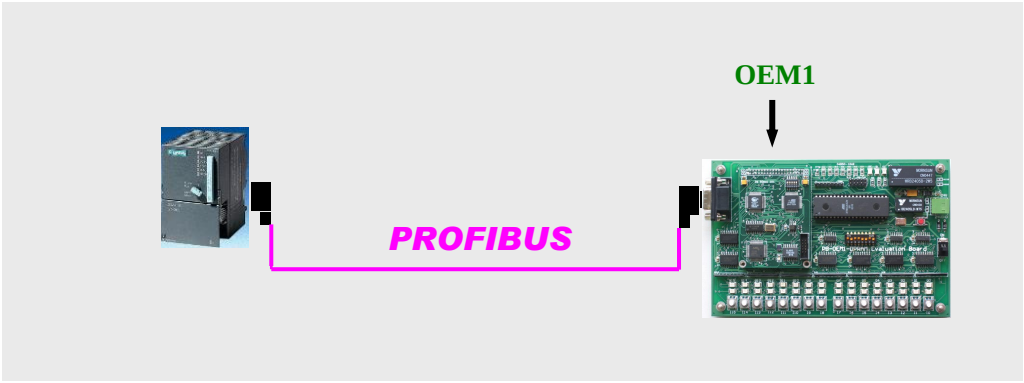
PROFIBUS-OEM1 调试实验系统

用户手册

V 1.0





北京鼎实创新科技有限公司

2010-8

目 录

第一章　PROFIBUS-OEM1 调试实验系统概述

OEM1 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 OEM1 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 OEM1

嵌入式 PROFIBUS 接口 OEM1（以下有时称 OEM1 卡）专为自主开发具有 PROFIBUS-DP 通信功能产品的用户，以 OEM 方式提供从站通信接口。产品型号是 PB-OEM1-DPRAM，《PB-OEM1-DPRAM 产品手册》中有详细使用及原理说明。

2. OEM1 调试实验系统主要用途

⑴ 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。

⑵ OEM1 调试实验系统包括：PROFIBUS 主站、PROFIBUS 从站（开发实验板+OEM1 卡）、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。调试实验系统可提供两种调试工作模式。

⑶ 系统包括一块开发实验板。这是 OEM1 卡的评估板，主 CPU 选用的是 Atmel 公司的 AVR 系列：Atmega162V。开发实验板的硬件原理图和 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

⑴ 调试模式 1：实验板调试模式（离线模式）

主要调试功能：建立主站、从站的配置、PROFIBUS 系统的连接、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，观察 OEM1 接口板初始化阶段的状态变化；然后再将自主开发的用户板与 PROFIBUS 主站连接，见图 1-1。

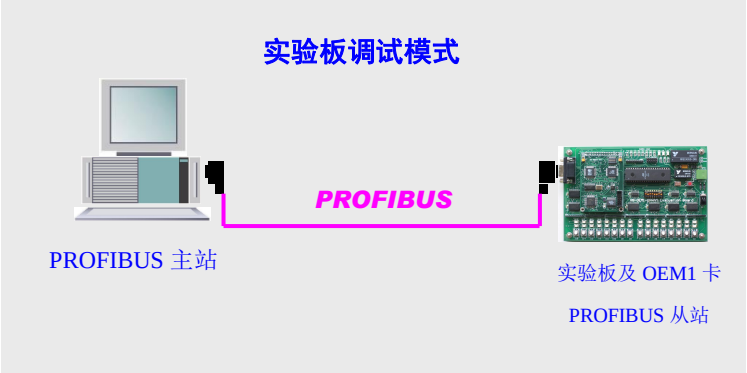


图 1-1-1 调试模式 1—实验板调试模式

实验板程序下载：在离线模式下，若用户需要修改实验板上 AVR CPU 的程序，则需要用 AVR 的 C 编译器软件 ICC AVR 对实验板 CPU 的 C 源程序进行编译、连接生成 .HEX 目标代码后，通过 SLISP 软件和

AVR 下载线下载到实验板中。此方式只用于修改和下载实验板的 CPU 程序，没有在线连接和调试功能。见图 1-1-2。使用 AVR 程序烧录器亦可完成同样的功能，但成本要高一些。

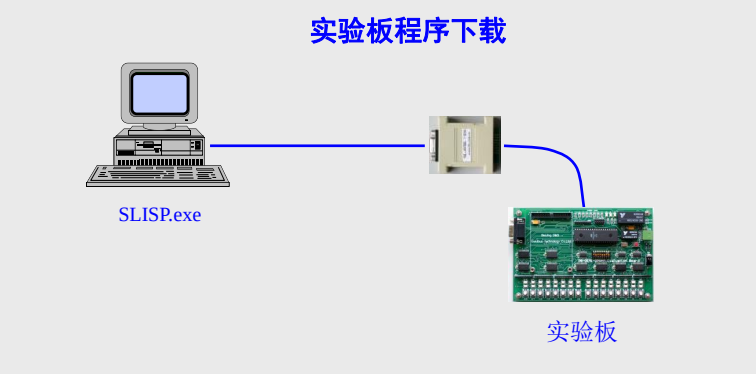


图 1-1-2 实验板程序下载

⑵ 调试模式 2：OEM1 接口板调试模式（在线模式）

主要调试功能：通过 AVR-JTAG 仿真器及配套的 ICC AVR C 编译器和 AVR Studio 4 调试软件，对实验板 CPU 的 C 源程序进行编译和下载，并可由此对实验板 CPU 的 C 源代码进行在线调试。从而可监测 OEM1 接口板的 DPRAM 数据和握手信号、可观察接口板初始化过程、接口板状态、中断过程、PROFIBUS I/O 数据、用户参数数据等。是学习、调试 OEM1 接口板的主要方法,见图 1-2。

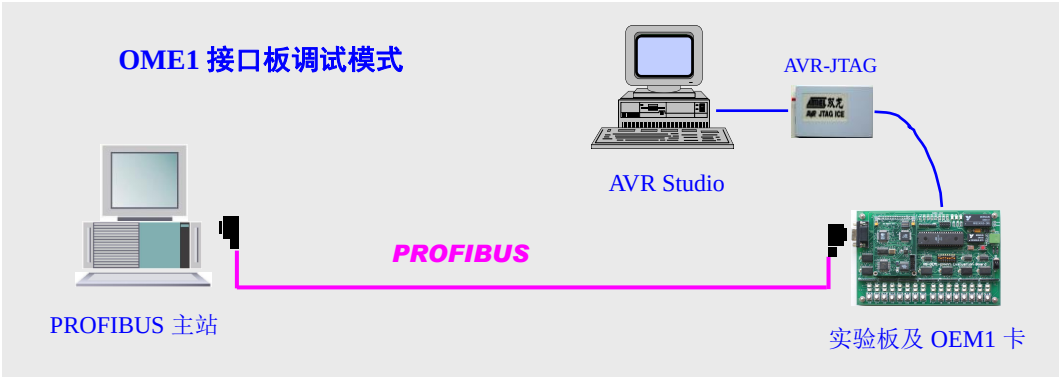


图 1-2 调试模式 2—OEM1 接口板调试模式

4. OEM1 调试实验系统主要部件及选件

⑴ 基本配置

PROFIBUS-OEM1 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-OEM1-DPRAM 嵌入式 PROFIBUS 接口	鼎实科技	1 块	
2	PB-OEM1-SAMPLE 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、例程及 GSD 文件、实验板硬件原理图、实验板 C 程序源码。	鼎实科技	1 张 CD	随系统赠送

	手册（电子版）： 《PB-OEM1-DPRAM 产品手册》 《PROFIBUS-OEM1 调试实验系统使用手册》			
PROFIBUS-OEM1 调试实验系统自选配置（用户根据需要自选）				
5	AVR-JTAG-ICE 仿真器	双龙公司	1 套	
6	SL-AVRL 下载线	双龙公司	1 套	

⑵ 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

② 选择 PLC 做主站，优点：系统可靠，使用方便，可做为批量生产产品时的产品出厂测试系统。

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	
2	组态软件（光盘）： STEP 7　V5.2(DEMO 版)；WinCC V5.1(DEMO 版)； SIMATIC NET V6.2(DEMO 版)；	西门子	3CD	随系统赠送

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 V5.2+SP1(DEMO 版)	西门子	1CD	随系统赠送

第二章 OEM1 开发实验板

OEM1 开发实验板是调试系统中 PROFIBUS 从站部件，是 OEM1 接口板的评估板。型号：PB-OEM1-SAMPLE。

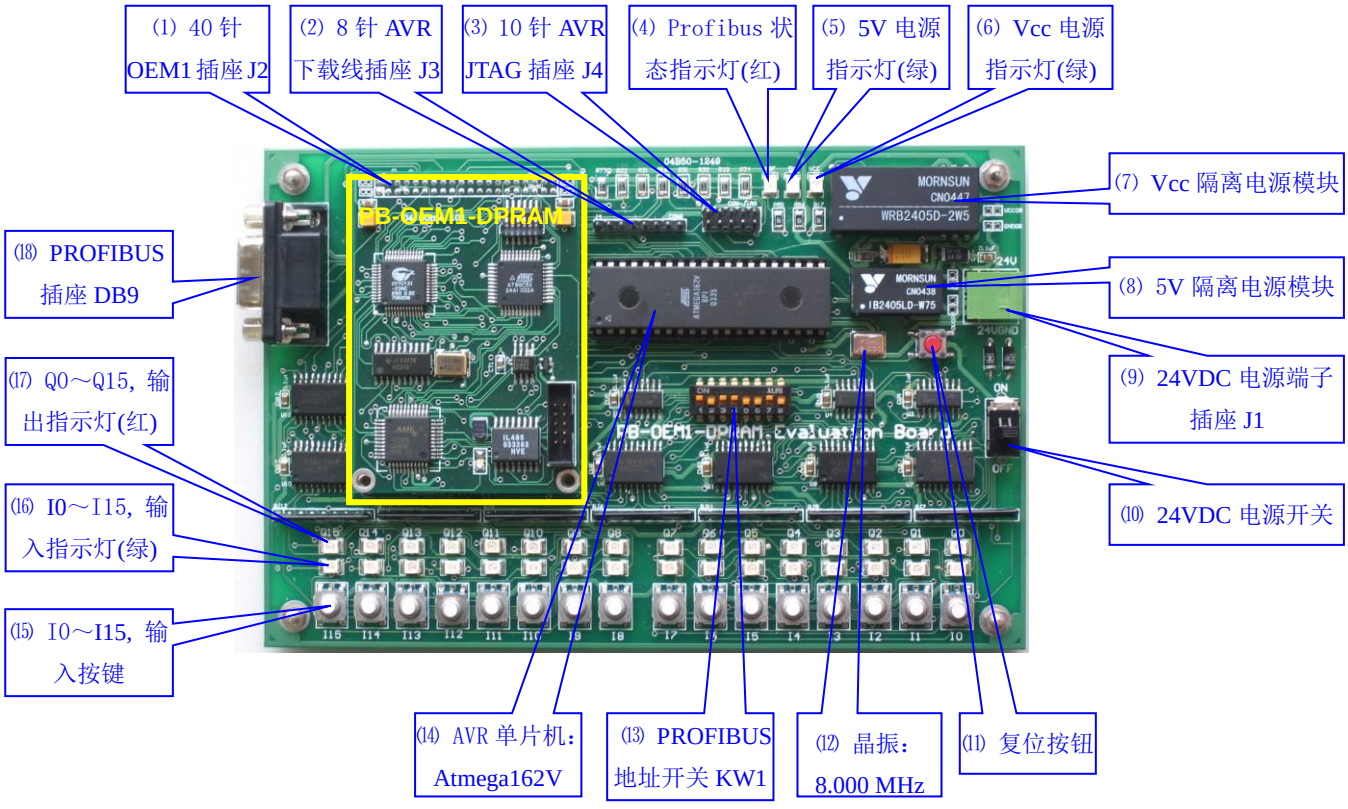
开发实验板为用户提供了一块开发样板。实验板的全部硬件原理图和C源码程序可在本手册中找到。用户使用 OEM1 开发实验板可以实现：

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-OEM1-DPRAM 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 2 种调试模式；

1. PB-OEM1-SAMPLE 开发实验板布局图



(1) 40 针插座 J2：实验板上连接 PB-OEM1-DPRAM 的 40 针（male）双排插座。针距：2mm。

(2) 8 针插座 J3：实验板上连接 AVR - SPI 下载线的 8 针（male）单徘插座。针距：2.54mm。

(3) 10 针插座 J4：实验板上连接 AVR - JTAG 仿真器的 10 针（male）双徘插座。针距：2.54mm。

(4) Profibus 通讯状态指示灯（红色）：OEM1 接口板与 Pofibus 主站连通后，该灯熄灭。

(5) 5V 电源指示灯（绿色）：5V 隔离电源模块正常工作时，该灯亮起。

(6) Vcc 电源指示灯（绿色）：Vcc 隔离电源模块正常工作时，该灯亮起。

⑺ **Vcc 隔离开关电源模块:** 实验板及 OEM1 的 CPU 工作的主电源模块，DC 24V→DC 5V 隔离 2.5W。

⑻ **5V 隔离开关电源模块:** 与 Vcc 隔离的 Profibus 通讯电源模块，DC 24V→DC 5V 隔离 0.75W。

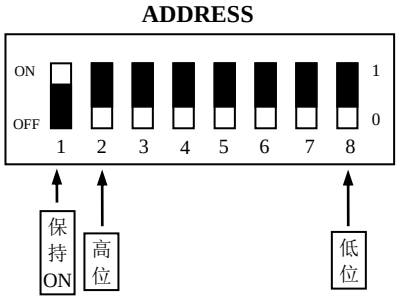
⑼ **24V 电源输入端子:** 24V 直流电源进线端子，可插拔。

⑽ **24V 电源开关:** 实验板电源总开关。

⑾ **复位按扭：**向 AVR CPU 提供低有效复位信号，并经 J2 插座向 OEM1 提供复位信号。

⑿ **晶振 8.000 MHz：**有源晶振，向 AVR CPU 提供工作时钟。

⒀ **PROFIBUS 地址开关 KW1：**PRODIBUS 从站地址设置开关（低 7 位有效），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。



KW1 设置 PRODIBUS 从站地址例：

从站地址$_{(max)}$=126$_{(Dec)}$=7E$_{(Hex)}$； KW1 设置 = 80$_{(Hex)}$ +7E$_{(Hex)}$ = FE$_{(Hex)}$ = 11111110$_{(Bin)}$

从站地址=50$_{(Dec)}$=32$_{(Hex)}$； KW1 设置 = 80$_{(Hex)}$ +32$_{(Hex)}$ = B2$_{(Hex)}$ = 10110010 $_{(Bin)}$

从站地址=19$_{(Dec)}$=13$_{(Hex)}$； KW1 设置 = 80$_{(Hex)}$ +13$_{(Hex)}$ = 93$_{(Hex)}$ = 10010011$_{(Bin)}$

⒁ **单片机：**Atmega l62v，Atmel 公司高性能 8 位单片机。

⒂ **输入按键：**共 16 个输入按键 I0 ～I15。

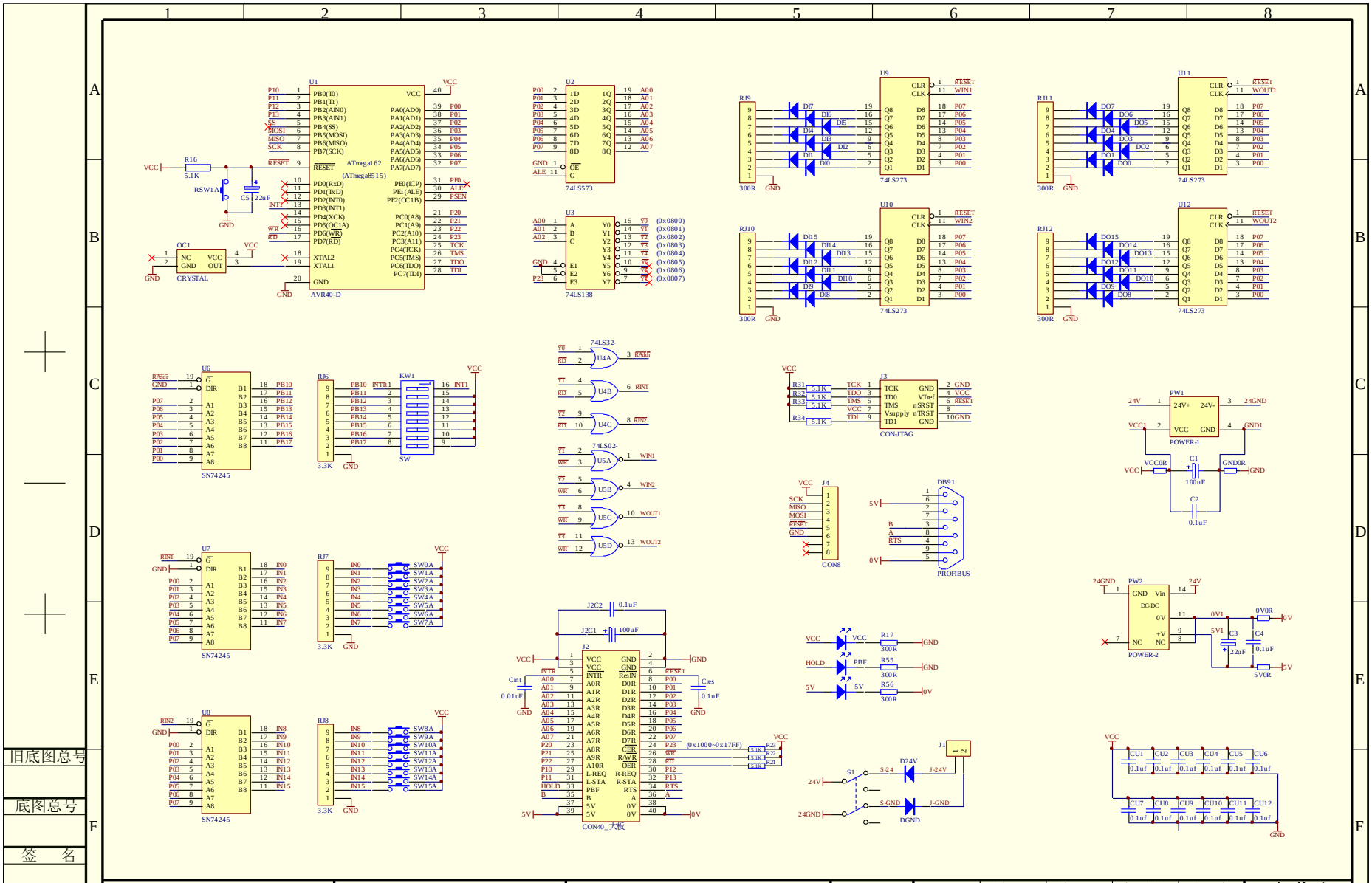
⒃ **输入指示灯：**共 16 个绿色 LED 输入指示灯。

⒄ **输出指示灯：**共 16 个红色 LED 输入指示灯 Q0 ～ Q15。

⒅ **PROFIBUS 插座 DB9:** 用于连接 PROFIBUS 电缆的插座。为 9 孔（female）D 型插座。

2．OEM1 开发实验板原理图

在实验系统中，开发实验板与 OEM1 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-OEM1-DPRAM 产品手册》、了解 OEM1 模块应用技术后，可以参考实验板原理图开发用户从站。



设计单位	项目名称	图纸名称		设计	制图	校对	标准化	审批	文件名
		D_RAM评估板	签字						
			日期						第 张
			阶段	2004.12.1.		比例			共 张

旧底图总号

底图总号

签 名

日 期

开发实验板原理图各部件：

⑴ **U1：**实验板 CPU，ATMEL / Atmeg162V。

⑵ **U2：**SN74AHCT573，8 位锁存器；用于 CPU 低 8 位地址锁存。

⑶ **U3：**SN74AHCT138，3-8 译码器；用于 CPU 地址译码。

⑷ **U4/U5：**SN74AHCT32 / SN74AHCT02，4 或门 / 4 或非门；用于 CPU 读写逻辑。

⑸ **U6-U8：**SN74AHCT245，8 总线收发器；用于读取地址开关和输入按键状态。

⑹ **U9-U12：**SN74AHCT273，8 D 触发器；用于锁存输入按键状态和 Profibus 输出状态。

⑺ **OC1：**8.000 MHz 晶振。

⑻ **KW1：**Profibus 地址拨码开关。

⑼ **SW0-SW15：**16 个输入操作按键（无自锁）。

⑽ **DI0-DI15 / DO0-DO15：**16 个绿色 LED 输入按键指示灯 / 16 个红色 LED Profibus 输出指示灯。

⑾ **RJ6-RJ12：**接地排电阻。

⑿ **PW1：**实验板主电源，DC-DC：24V/5V/2.5W。

⒀ **PW2：**Profibus 电源，DC-DC：24V/5V/0.75W。

⒁ **Vcc/5V LED：**实验板主电源 / Profibus 电源指示灯，绿色 LED。

⒂ **PBF LED：**Profibus 通讯故障指示灯，红色 LED。

⒃ **J1/S1：**24V 电源端子/ 电源开关；输入电源：24V ±5% 。

⒄ **J2：**40 针 PB-OEM1-DPRAM 模块连接插座。

⒅ **J3/J4：**8 针 AVR – SPI 下载线插座/10 针 AVR - JTAG 仿真器插座。

⒆ **DB91：**PROFIBUS 总线标准 DB9 插座。

⒇ **D24V/DGND：**24V 电源极性保护二级管。

3．OEM1 开发实验板中有关地址的定位

P27~P24	P23~P20	A7,A6,A5,A4	A3,A2,A1,A0	地址	/RD	选中	/WR	选中
0000	0000	0000	0000	0000		AVR 内部		AVR 内部
0000	0111	1111	1111	~ 03FF		RAM(1K)		RAM(1K)
0000	1xxx	xxxx	x000	0800	√	/RAddr		/RAddr
0000	1xxx	xxxx	x001	0801	√	/RIN1	√	/WIN1
0000	1xxx	xxxx	x010	0802	√	/RIN2	√	/WIN2
0000	1xxx	xxxx	x011	0803			√	/WOUT1
0000	1xxx	xxxx	x100	0804			√	/WOUT2
xxx1	0000	0000	0000	1000	√	DPRAM(1K)	√	DPRAM(1K)
xxx1	0111	1111	1111	~ 13FF				

DPRAM 首地址定位在: 1000H

第三章　实验板程序、GSD 文件

1. 实验板程序

(1) 实验板 CPU

采用 ATMEL 公司 Atmega162V，主要技术指标：

产品特性
- 高性能、低功耗的**8**位 **AVR®** 微处理器
- 先进的 **RISC** 结构
  - **131** 条指令 – 大多数为单时钟周期指令
  - **32** 个 **8** 位通用工作寄存器
  - 全静态操作
  - 工作于 **16 MHz** 时性能高达 **16 MIPS**
  - 两个时钟周期的片内乘法器
- 非易失性的程序和数据存储器
  - **16K** 字节的系统内可编程 **Flash**（擦写寿命**: 10,000** 次）
  - 具有独立锁定位的可选 **Boot** 代码区
    通过片上 **Boot** 程序实现系统内编程
    真正的同时读写操作
  - **512** 字节的 **EEPROM**（擦写寿命**: 100,000** 次）
  - **1K** 字节的片内 **SRAM**
  - **64K** 字节可选外部存储器空间
  - 用于软件加密的可编程锁定位
- **JTAG** 接口**(** 与**IEEE 1149.1** 标准兼容**)**
  - 符合**JTAG** 标准的边界扫描功能
  - 支持扩展的片内调试功能
  - 通过**JTAG** 接口可编程**Flash**、**EEPROM**、熔丝位和锁定位
- 外设特点
  - **2**个具有独立预分频和比较功能的**8** 位定时器/ 计数器
  - **2**个具有预分频、比较功能和捕捉功能的**16** 位定时器/ 计数器
  - 具有独立振荡器的实时计数器
  - 六通道 **PWM**
  - **2**个可编程的串行接口**USART**
  - 具有主机/ 从机模式的**SPI** 串行接口
  - 具有独立片内振荡器的可编程看门狗定时器
  - 片内模拟比较器
- 特殊的微控制器特点
  - 上电复位以及可编程的掉电检测
  - 经过标定的片内 **RC** 振荡器
  - 片内/ 片外中断源
  - 五种休眠模式：空闲模式、省电模式、掉电模式、**Standby** 模式以及扩展的**Standby** 模式
- **I/O** 口与封装
  - **35** 个可编程的**I/O** 口引脚
  - **40**引脚**PDIP** 封装**, 44** 引脚**TQFP** 与**44** 引脚**MLF** 封装
- 工作电压**:**
  - **ATmega162V**：1.8 **- 5.5V**
  - **ATmega162**：　2**.7 - 5.5V**
- 速度等级
  - **ATmega162V**：**0 - 8　MHz**
  - **ATmega162**：　**0 – 16 MHz**

⑵ 实验板程序

① 出厂时实验板 CPU 程序是“例 2：监测 DPRAM 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 DPRAM 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 DPRAM 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，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，包括：

•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，可直接用“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 设备进行配置。

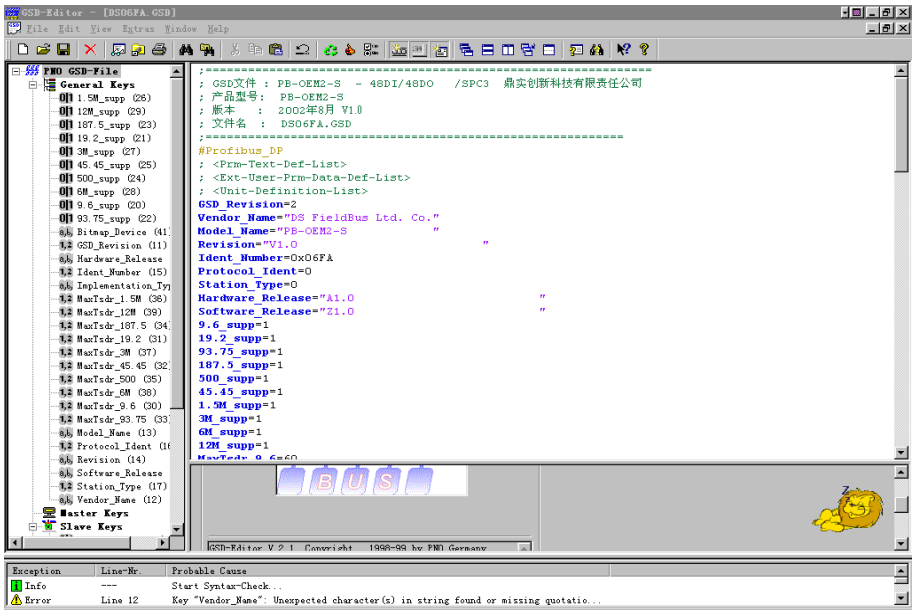


图 3-1：GSD-EDIT 打开 DS_06FA.GSD 文件

⑤ 国际 PROFIBUS 组织 PI 提供了 GSD 文件编辑软件：“GSD-Editor”，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。

3. 用户产品的 ID 号、GSD 文件及产品认证

⑴ DPRAM 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；

⑵ 由于本产品以 OEM 方式销售，用户对应用本产品开发的 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 GSD 文件中使用的 ID 号和 GSD 文件名不宜作为用户产品的 ID 号和 GSD 文件名；

⑶ 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。

⑷ 用户也可以暂时自定义一个 ID 号，在产品开发时期使用。因为在一条 PROFIBUS 总线上，不同类型、或相同类型从站的具有同一 ID 号并不影响系统连通。

⑸ 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。

第四章：实验板程序举例

例 1． 48 字节输入/32 字节输出（无用户参数）

⑴ 技术参数

PROFIBUS 站号： 50 号站= 0x32；

PROFIBUS 产品 ID 号： 0x06FA

PROFIBUS 输入/输出： 48 字节输入/32 字节输出（ 0x30 Input / 0x20 Output ）；

（对应的 I/O 配置数据为： 0x1f, 0x2f, 0x1f, 0x2f, 0x1f；

对应的 I/O 配置数据长度： CFG_LEN = 5；）

用户参数： 无；

（对应用户参数长度： User_Prm_Data_Len = 0；）

⑵ DPRAM 初始化数据

DPRAM 地址	变量	初始化数据	说明
0x1010	L_ini[0]	0x32	Profibus 站号
0x1011	L_ini[1]	0x06	ID 号高位
0x1012	L_ini[2]	0xfa	ID 号底位
0x1013	L_ini[3]	0x05	CFG 数据长度
0x1014	L_ini[4]	0x1f	CFG 数据 1
0x1015	L_ini[5]	0x2f	CFG 数据 2
0x1016	L_ini[6]	0x1f	CFG 数据 3
0x1017	L_ini[7]	0x2f	CFG 数据 4
0x1018	L_ini[8]	0x1f	CFG 数据 5
……		0x00 ……	
0x101E	L_ini[14]	0x30	48 字节输入
0x101F	L_ini[15]	0x20	32 字节输出
0x1020	L_ini[16]	0x00	无用户参数
0x1021	L_ini[17]	0x02	字校验和∑1010-1020 高位
0x1022	L_ini[18]	0x42	字校验和∑1010-1020 低位

表 4-1

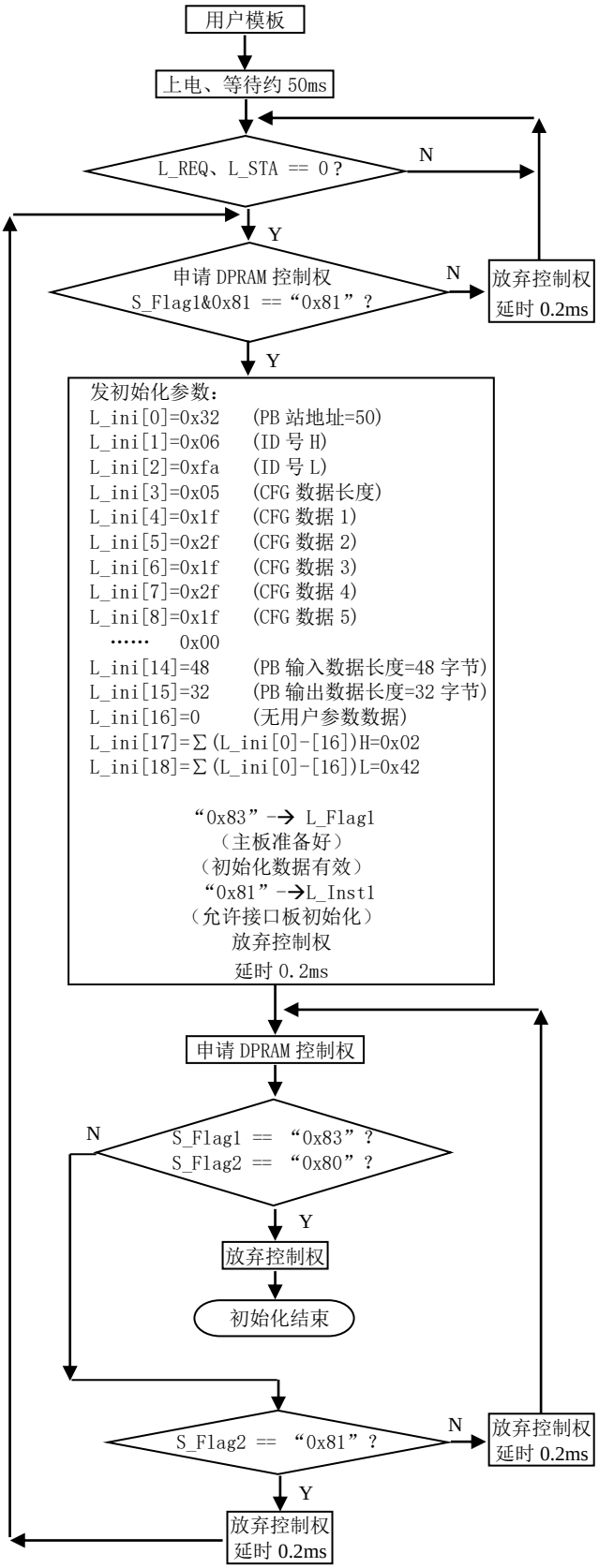
⑶ 实验板初始化开始前 DPRAM 接口板的状态

上电后 DPRAM 接口板即开始对双口 RAM 进行自检。在此期间实验板应处于等待状态，不得占用双口 RAM 控制权，亦不得对双口 RAM 进行任何读写操作。否则可能影响自检结果。

若自检不正常：则接口板将置状态字节 S_Flag2 = 0x84 后，进入死循环。此为致命性错误。

若自检正常： 则接口板将置状态字节 S_Flag1 = 0x81 后，等待实验板发初始化命令。

(4) 实验板初始化 DPRAM 接口板的过程



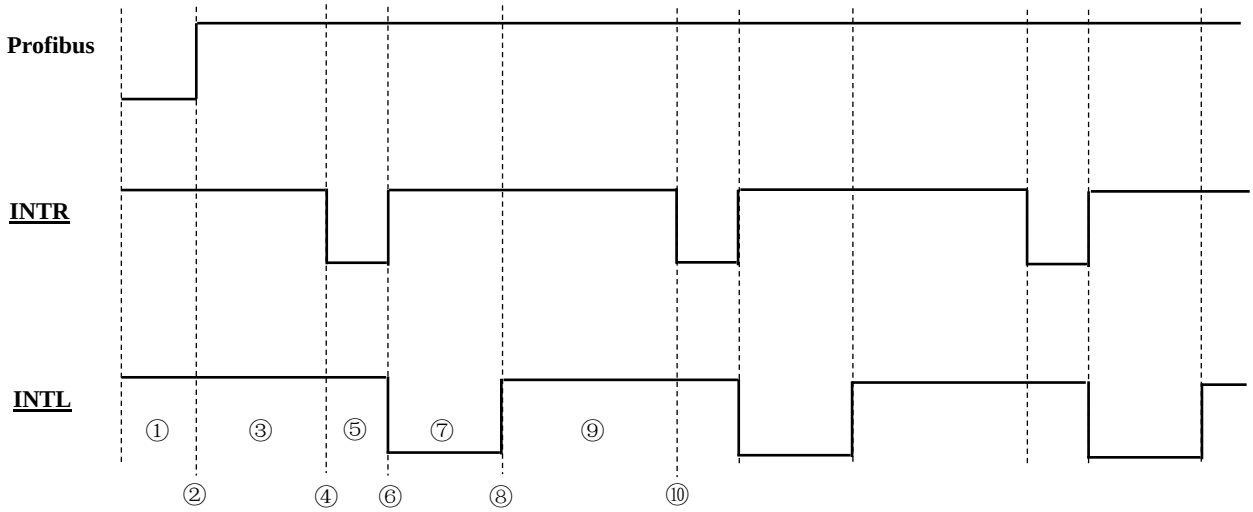
⑸ 实验板初始化完成后 DPRAM 接口板的状态

在接口板收到正确的初始化数据后，会有短暂时间状态字节 S_Flag1 = 0x91、S_Flag2 = 0x80，表明初始化正在进行。

初始化完成后 DPRAM 接口板的状态为：状态字节 S_Flag1 = 0x83、S_Flag2 = 0x80。表明 DPRAM 接口板不仅正确的接受到了初始化数据，并已正确的完成 SPC3 的初始化，处在等待 Profibus 主站连接状态。

若出现状态字节 S_Flag2 ≠ 0x80，则表明出现了某种错误，需根据错误码进行相应处理。

⑹ 与 Profibus 主站连通后，实验板与 DPRAM 接口板的数据交换过程



① 接口板已完成初始化，但尚未与 Profibus 主站连通；
② 接口板与 Profibus 主站连通，进入数据交换状态，Profibus 红色状态指示灯灭；
③ 接口板与 SPC3 交换 In/Out 数据，向 DPRAM 写入 Out 数据；
④ 接口板向实验板发首次 INTR 中断；
⑤ 实验板执行中断处理程序，向 DPRAM 写入 In 数据，并从 DPRAM 读取 Out 数据；
⑥ 实验板中断程序结束，清除 INTR 中断、并向接口板发 INTL 中断；
⑦ 接口板执行中断处理程序，从 DPRAM 读取 In 数据；
⑧ 接口板中断程序结束，清除 INTL 中断；
⑨ 接口板再次与 SPC3 交换 In/Out 数据，向 DPRAM 写入 Out 数据；
⑩ 接口板向用户模板发下一次 INTR 中断；

⑺ 实验板程序清单（例 1）

```
/*
+--------------------------------------------------------------------+
| 文件名称:   AVR 开发实验板例 1 主程序                              |
| 版      本:  V1.0                                                  |
| 制作单位:   北京鼎实创新科技公司                                   |
| 网址：        www.c-profibus.com.cn                                |
| email:        sunhm@c-profibus.com.cm                              |
| 日      期:  2005/01                                               |
+--------------------------------------------------------------------+
/*        以下头文件，请注意修改文件路径                            */
/*--------------------------------------------------------------------*/
// ICC-AVR application builder : 2003-6-8 10:39:03
// Target : M162
// Crystal: 8.000 MHz

#include <c:\icc\include\iom162v.h>
#include <c:\icc\include\macros.h>
#define UBYTE unsigned char
#define UWORD unsigned int

   UWORD      Ui,Vi,Wi;
   UBYTE      DI0_7,DI8_15;
   UBYTE      DO0_7,DO8_15;
   UBYTE      D_para[32] ;                  //实验板用户参数接收区（来自 DPRAM）

#pragma abs_address:0x1000
struct {
UWORD Hand_Flag;                          //55AA-->硬件握手,00FF-->软件握手
UBYTE Lr;                                 //实验板请求占用 DPRAM
UBYTE Ls;                                 //实验板确认占用 DPRAM
UBYTE L_Flag1;                            //实验板状态字节 1
UBYTE L_Flag2;                            //实验板状态字节 2
UBYTE L_Inst1;                            //实验板命令字节 1
UBYTE L_Inst2;                            //实验板命令字节 2
}DRL ;                                    //（来自实验板）
#pragma end_abs_address
#pragma abs_address:0x1010
UBYTE    L_ini[32] ;                      //初始化报文区（来自实验板）
#pragma end_abs_address
#pragma abs_address:0x1050
UBYTE    L_PBin[128] ;                    //Profibus 输入数据区（来自实验板）
#pragma end_abs_address

#pragma abs_address:0x1100
struct {
UWORD Spare1;                             //不用
UBYTE Sr;                                 //接口板请求占用 DPRAM
UBYTE Ss;                                 //接口板确认占用 DPRAM
UBYTE S_Flag1;                            //接口板状态字节 1
UBYTE S_Flag2;                            //接口板状态字节 2
}DRS ;                                    //（来自接口板）
#pragma end_abs_address
#pragma abs_address:0x1110
UBYTE    S_para[32] ;                     //用户参数化报文区（来自接口板）
#pragma end_abs_address
#pragma abs_address:0x1150
UBYTE    S_PBout[128] ;                   //Profibus 输出数据区（来自接口板）
#pragma end_abs_address
```

```
#pragma interrupt_handler Dpram_int:3          //双口 RAM 触发 INT1 中断

void Ctrl_On(void);                            //声明“申请 DPRAM 控制权”子程序
void Dpram_int(void);                          //声明“DPRAM 中断处理”子程序

void init_devices(void)
{
 UWORD x;

 PORTA = 0xFF;                                 //各口置为输入、上拉
 DDRA  = 0x00;
 PORTB = 0xF3;
 DDRB  = 0x0C;                                 //PB2、PB3 置为输出
 PORTC = 0xFF;
 DDRC  = 0x00;
 PORTD = 0xFF;
 DDRD  = 0x00;
 PORTE = 0x07;
 DDRE  = 0x00;

 for( x=0;x<30000;x++ ){};                     //实验板延时等待
 Ctrl_On();                                    //实验板申请 DPRAM 控制权
 MCUCR = 0x80;                                 //外部 RAM 允许
 EMCUCR = 0x00;
 PORTB = 0x00;                                 //实验板放弃 DPRAM 控制权
} //all peripherals are now initialised

/*=============================================================*/

void main(void)
{
  UBYTE      i,j,k,l,m,n,u,v,address;
  UWORD       mm,nn,x,y,z,nw;
  UBYTE      key0_7new, key8_15new;
  UBYTE      key0_7old, key8_15old;
  UBYTE      *ip,*op,*cp;
  UBYTE      *ADDRESS;
  UBYTE      *KEY_IN1,*KEY_IN2;
  UBYTE      *LED_IN1, *LED_IN2;
  UBYTE      *LED_OUT1,*LED_OUT2;

  CLI();                              //disable all interrupts
  init_devices();                     //初始化

  ip=(UBYTE*)0;
  op=(UBYTE*)0;
  LED_IN1=(UBYTE*)0;
  LED_IN2=(UBYTE*)0;
  LED_OUT1=(UBYTE*)0;
  LED_OUT2=(UBYTE*)0;
  KEY_IN1 =(UBYTE*)0;
  KEY_IN2 =(UBYTE*)0;
  ADDRESS =(UBYTE*)0;
  ip=ip+0x1000;                       // DPRAM 输入区首地址指针：      1000 H (读/写)
  op=op+0x1100;                       // DPRAM 输出区首地址指针：      1100 H (读/写)
  LED_IN1=LED_IN1+0x0801;             //输入绿 LED 指示灯 I0~I7 地址指针：  0801 H (写)
  LED_IN2=LED_IN2+0x0802;             //输入绿 LED 指示灯 I8~I15 地址指针：  0802 H (写)
  LED_OUT1=LED_OUT1+0x0803;           //输出红 LED 指示灯 Q0~Q7 地址指针：  0803 H (写)
  LED_OUT2=LED_OUT2+0x0804;           //输出红 LED 指示灯 Q8~Q15 地址指针：0804 H (写)
```

```
   ADDRESS =ADDRESS+0x0800;          //Profibus 地址开关 Address 地址指针：0800 H (读)
   KEY_IN1 =KEY_IN1+0x0801;          //输入按键开关 I0~I7 地址指针：    0801 H (读)
   KEY_IN2 =KEY_IN2+0x0802;          //输入按键开关 I0~I7 地址指针：    0802 H (读)
   cp=(UBYTE*)0;
   u=Wi=0;

x=0;y=1;
for( nn=0;nn<0x8000;nn++ )
    {
       *LED_IN1=y;                   //点绿 LED 灯
       *LED_IN2=y;                   //点绿 LED 灯
       *LED_OUT1=y;                  //点红 LED 灯
       *LED_OUT2=y;                  //点红 LED 灯
       if( nn/0x1000>x )
          {
             x=nn/0x1000;
             y=y*2+1;                //亮灯左增长
          };
      };
   *LED_IN1=0;                       //绿 LED 灯灭
   *LED_IN2=0;                       //绿 LED 灯灭
   *LED_OUT1=0;                      //红 LED 灯灭
   *LED_OUT2=0;                      //红 LED 灯灭

   address=*ADDRESS;                 //读 Profibus 地址开关

   Ctrl_On();                        //获取 DPRAM 读写控制权
   *LED_OUT1=DRS.S_Flag1;            //显示 S_Flag1 状态（0x81）
   *LED_OUT2=DRS.S_Flag2;            //显示 S_Flag2 状态（0x00）
   PORTB=PORTB&0xF3;                 //释放 DPRAM 读写控制权
   for(x=0;x<500;x++)
       { for(y=0;y<1000;y++);}       //调试，暂停 1（维持状态显示 1~2 秒）

/*======================== 向接口板发“初始化数据” ======================*/

for(m=0;m==0; )
    {
       for(mm=0;mm==0;)              //发送“初始化数据”一次
           { Ctrl_On();              //实验板申请 DPRAM 控制权
             if( (DRS.S_Flag1&0x81)==0x81 )     //判接口板准备好标志
                {
                   L_ini[0]=*ADDRESS;           //读地址开关
                   L_ini[1]=0x06;               //ID 号 H
                   L_ini[2]=0xfa;               //ID 号 L
                   L_ini[3]=0x05;               //CFG 长度：  05
                   L_ini[4]=0x1f;               //CFG 数据 1：0x1f
                   L_ini[5]=0x2f;               //CFG 数据 2：0x2f
                   L_ini[6]=0x1f;               //CFG 数据 3：0x1f
                   L_ini[7]=0x2f;               //CFG 数据 4：0x2f
                   L_ini[8]=0x1f;               //CFG 数据 5：0x1f
                   L_ini[14]=48;                //Profibus 输入数据长度：48
                   L_ini[15]=32;                //Profibus 输入数据长度：32
                   L_ini[16]=0;                 //用户参数化报文长度：0
                   x=0;
                   for( i=0;i<17;i++ )          //计算“初始化校验和”
                       { x=x+L_ini[i];}
                   L_ini[17]=(x&0xff00)/0x100;  //“初始化校验和”H
                   L_ini[18]=x&0x00ff;          //“初始化校验和”L

                   DRL.L_Flag1=0x83;            //置“实验板准备好”，“初始化数据有效”标志
```

```
                DRL.L_Inst1=0x81;                    //发初始化启动命令
                mm=1;
                }
              PORTB=PORTB&0xF3;                      //实验板放弃 DPRAM 控制权
             for( i=0;i<2000;i++ ){};                //延时
            }

      for(nn=0;nn==0;)                               //等待接口板初始化完成
          { Ctrl_On();                               //实验板申请 DPRAM 控制权
            if((DRL.L_Inst1==0x81)&&((DRS.S_Flag1&0x83)==0x83))      //若接口板初始化已完成
              {
                DRL.L_Inst1=0x00;                    //清初始化启动命令
                *LED_OUT1=0x00;                      //点灯
                *LED_OUT2=0x00;                      //点灯
                for( i=0;i<2000;i++ ){};             //延时
                nn=1;
                m=1;                                 //“初始化”过程结束
                }
      else if((DRL.L_Inst1==0x81)&&((DRS.S_Flag2&0x81)==0x81))       //若初始化校验和错
              {
                DRL.L_Inst1=0x00;                    //清初始化启动命令
                DRS.S_Flag2=0x80;                    //清初始化校验和错

                *LED_OUT1=0xff;                      //点灯
                *LED_OUT2=0xff;                      //点灯
                for( i=0;i<2000;i++ ){};             //延时
                nn=1;
                m=0;                                 //重新开始“初始化”过程
                }
              PORTB=PORTB&0xF3;                      //实验板放弃 DPRAM 控制权
            for( i=0;i<2000;i++ ){};                 //延时
          };
    };
/*======================== 发“初始化数据”END ====================*/

   Ctrl_On();
   *LED_OUT1=DRS.S_Flag1;                            //显示 S_Flag1 状态（0x83）
   *LED_OUT2=DRS.S_Flag2;                            //显示 S_Flag2 状态（0x80）
   PORTB=PORTB&0xF3;
   for(x=0;x<500;x++)                                //调试，暂停 2（维持状态显示 1~2 秒）
       { for(y=0;y<1000;y++);}

for( x=0;x<10;x++ )                                  //延时，LED 灯闪烁
 {
    for(y=0;y<7000;y++)
    {
    *LED_IN1=0xff;                                   //LED 灯亮
    *LED_IN2=0xff;
    *LED_OUT1=0xff;
    *LED_OUT2=0xff;
    };
    for(y=0;y<7000;y++)
    {
    *LED_IN1=0x00;                                   //LED 灯灭
    *LED_IN2=0x00;
    *LED_OUT1=0x00;
    *LED_OUT2=0x00;
    };
 };
```

```
   Ctrl_On();
   *LED_OUT1=DRS.S_Flag1;                    //显示 S_Flag1 状态（主站连通 0x8b，否则 0x83）
   *LED_OUT2=DRS.S_Flag2;                    //显示 S_Flag2 状态（0x80）
   PORTB=PORTB&0xF3;
   for(x=0;x<500;x++)                        //调试，暂停 3（维持状态显示 1~2 秒）
       { for(y=0;y<1500;y++);}

  SEI();                                     //开中断
  GICR= 0x80;                                //允许 INT1（低电平），屏蔽其余中断
  DI0_7 = DI8_15 = 0x00;                     //IO 变量清“0”

  for(;;)                                    //主循环开始
  {
/*=====================键盘扫描==================================*/

       if( u==0 )
          { key0_7new = *KEY_IN1;
            if(key0_7new!=key0_7old)         //键 I0-I7 有动作（按下或抬起）
                {u=1;} }
else if( (u!=0)&&(u<=10) )
          { if(key0_7new == *KEY_IN1)        //连续 10 次稳定
                {u++;}
          else{u=0;} }
else if( u>10 )
          { if(key0_7new!=0)                 //键 I0-I7 有按下
                {DI0_7^=key0_7new;
                 *LED_IN1=DI0_7; }
              key0_7old = key0_7new;
              u=0; }

       if( v==0 )
          { key8_15new = *KEY_IN2;
            if(key8_15new!=key8_15old)       //键 I8-I15 有动作（按下或抬起）
                {v=1;} }
else if( (v!=0)&&(v<=10) )
          { if(key8_15new == *KEY_IN2)       //连续 10 次稳定
                {v++;}
          else{v=0;} }
else if( v>10 )
          { if(key8_15new!=0)                //键 I8-I15 有按下
                {DI8_15^=key8_15new;
                 *LED_IN2=DI8_15; }
              key8_15old = key8_15new;
              v=0; }

    *LED_IN1=DI0_7;                          //点绿 LED 输入灯（来自按键）
    *LED_IN2=DI8_15;
    *LED_OUT1=DO0_7;                         //点红 LED 输出灯（来自 Profibus 主站）
    *LED_OUT2=DO8_15;

/*============================  键盘扫描 END ================================*/
  };
}

/*----------------------------------  取得双口 RAM 读写控制权(函数) ----------------------------------*/
void Ctrl_On(void)
{
  UBYTE     Ci;

  for(;((PORTB&0x0C)!=0x0C)||((PINB&0x03)!=0x00);)
```

```
 {
        if(((PORTB&0x04)==0x00)&&((PINB&0x03)==0x00))
          { PORTB=PORTB|0x04;}                          //实验板申请 DPRAM 控制权
   else if(((PORTB&0x04)==0x04)&&((PINB&0x03)!=0x00))
          { PORTB=PORTB&0xF3;}                          //实验板放弃申请
   else if(((PORTB&0x04)==0x04)&&((PINB&0x03)==0x00))
          { PORTB=PORTB|0x08;}                          //实验板取得 DPRAM 控制权
 }
}
/*-------------------------------------- 取得双口 RAM 控制权 END ----------------------------------------*/

//======================== 双口 RAM 中断处理程序 ========================//
void Dpram_int(void)
{
   UBYTE      i;
   UWORD       x,y;
   UBYTE      *ip;
   UBYTE      *op;

   ip=(UBYTE*)0;
   op=(UBYTE*)0;
   ip=ip+0x1000;
   op=op+0x1100;
//=================== 将“输入数据”传送到双口 RAM 的 L_PBin[] 区 ====================//

      Ctrl_On();                                        //实验板申请 DPRAM 控制权
      if( ((DRS.S_Flag1&0x83)==0x83)&&(DRS.S_Flag2==0x80) )     //若接口板“初始化已完成”，无错
        {
          L_PBin[0]=L_ini[14];                          //Profibus 输入数据长度
          L_PBin[1]=DI0_7;                              //传送 键盘“输入数据” 1
          L_PBin[2]=DI8_15;                             //传送 键盘“输入数据” 2
          x=L_PBin[0]+L_PBin[1]+L_PBin[2];
          for( i=3;i<=L_PBin[0];i++ )
              { L_PBin[i]=i;                            //填写随机数
                x=x+i;
              }
          L_PBin[i]=x/0x100;                            //“输入数据”校验和 H
          L_PBin[i+1]=x-L_PBin[i]*0x100;                //“输入数据”校验和 L
        }
      DRL.L_Flag1=DRL.L_Flag1|0x84;                     // 1-->“输入数据有效”标志

//-------------------------------------- “输入数据”传送到 L_PBin[] 结束 ----------------------------------------//

//======================== 将主站“输出数据”送 LED 显示 ========================//

      if( ((DRS.S_Flag1&0x8b)==0x8b)&&(DRS.S_Flag2==0x80) )     //若接口板“输出数据”有效
        { x=0;
          for( i=0;i<=S_PBout[0];i++ )
              { x=x+S_PBout[i]; }                       //计算“输出数据”校验和
          y=S_PBout[i]*0x100+S_PBout[i+1];              //读取“输出数据”校验和

          if( x==y )
            {
              DO0_7=S_PBout[1];                         //传送“输出数据”
              DO8_15=S_PBout[2];
            }
           DRS.S_Flag1=DRS.S_Flag1&0xf7;                //清“输出数据”有效标志
        }

//---------------------------------------------- “输出数据”显示结束 ----------------------------------------------//
```

```
    x=*(ip+0x03ff);                          //读 DPRAM 清除中断源
    *(ip+0x03fe)=0x55;                       //写 DPRAM 向接口板发中断
     PORTB=PORTB&0xF3;                       //实验板放弃 DPRAM 控制权
}//external interupt on INT1
```

⑻ 例 1 的 GSD 文件

```
;=============================================================
; GSD文件 : PB-OEM1-DPRAM -（48字节输入/32字节输出）/SPC3          鼎实创新科技有限责任公司
; 产品型号:  PB-OEM1-DPRAM
; 版本      :  2005年1月  例1
; 文件名   :  OEM1-例1.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，可按用户公司名修改
Model_Name="DPRAM"                          ;模块名称，也是在Step7等软件中组态时该产品的名称
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="DPRAM"                          ;图标文件，用户可以自制图标，亦可缺省
; Slave-Specification:
OrderNumber="PB-OEM1-DPRAM"                    ;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=3
Max_Diag_Data_Len=6
User_Prm_Data_Len=0
Modular_Station = 0                            ;一体型模块
Slave_Family=0@DPRAM                           ;组态中的分类名
; <Module-Definition-List>
Module=" 48 Byte In, 32 Byte Out "  0x1f,0x2f,0x1f,0x2f,0x1f
EndModule                                      ;I/O配置数据必须与初始化报文中的CFG一致
```

例 2. 2 字输入/2 字输出 + 16 字节输入/16 字节输出（带 10 字节用户参数）

⑴ 技术参数

PROFIBUS 站号： 50 号站= 0x32；

PROFIBUS 产品 ID 号： 0x06FA

PROFIBUS 输入/输出： 2 字输入/2 字输出 + 16 字节输入/16 字节输出（0x14 Input / 0x14 Output ）；

（对应的 I/O 配置数据为： 0x50, 0x60, 0x1f, 0x2f；

对应的 I/O 配置数据长度： CFG_LEN = 4；）

用户参数： 1 字 + 3 字节；

（对应用户参数长度： User_Prm_Data_Len = 0x05；）

用户参数的设置：

▼ 用户参数 1：输入类型 AI_M，字节型

AI_M = 0-6 代表输入类型：1~5V、0~10V、0~5V、–10~+10V、4~20mA、0~20mA、-10~+10mA，共七种选择；

▼ 用户参数 2：输入滤波系数 AI_FK，字型

FIL_K = 0~1000（无符号整型数），表示输入滤波系数；

▼ 用户参数 3：输入数据类型 AI_D，字节型

AI_D = 0~2 代表数据类型：BCD（0000~9999）、无符号整型（0~65535）、有符号整型（-32767~+32767），共三种选择；

▼ 用户参数 4：输入单/双端 AI_CH，字节型

AI_CH = 0~1 分别代表：单端输入、双端输入，共二种选择；

⑵ DPRAM 初始化数据

DPRAM 地址	变量	初始化数据	说明
0x1010	L_ini[0]	0x32	Profibus 站号
0x1011	L_ini[1]	0x06	ID 号高位
0x1012	L_ini[2]	0xfa	ID 号底位
0x1013	L_ini[3]	0x04	CFG 数据长度
0x1014	L_ini[4]	0x51	CFG 数据 1
0x1015	L_ini[5]	0x61	CFG 数据 2
0x1016	L_ini[6]	0x1f	CFG 数据 3
0x1017	L_ini[7]	0x2f	CFG 数据 4
……		0x00 ……	
0x101E	L_ini[14]	0x14	20 字节输入
0x101F	L_ini[15]	0x14	20 字节输出
0x1020	L_ini[16]	0x05	5 字节用户参数
0x1021	L_ini[17]	0x02	字校验和∑1010-1020 高位
0x1022	L_ini[18]	0x63	字校验和∑1010-1020 低位

表 4-2

⑶ 实验板初始化开始前 DPRAM 接口板的状态

同例 1。

⑷ 实验板初始化 DPRAM 接口板的过程

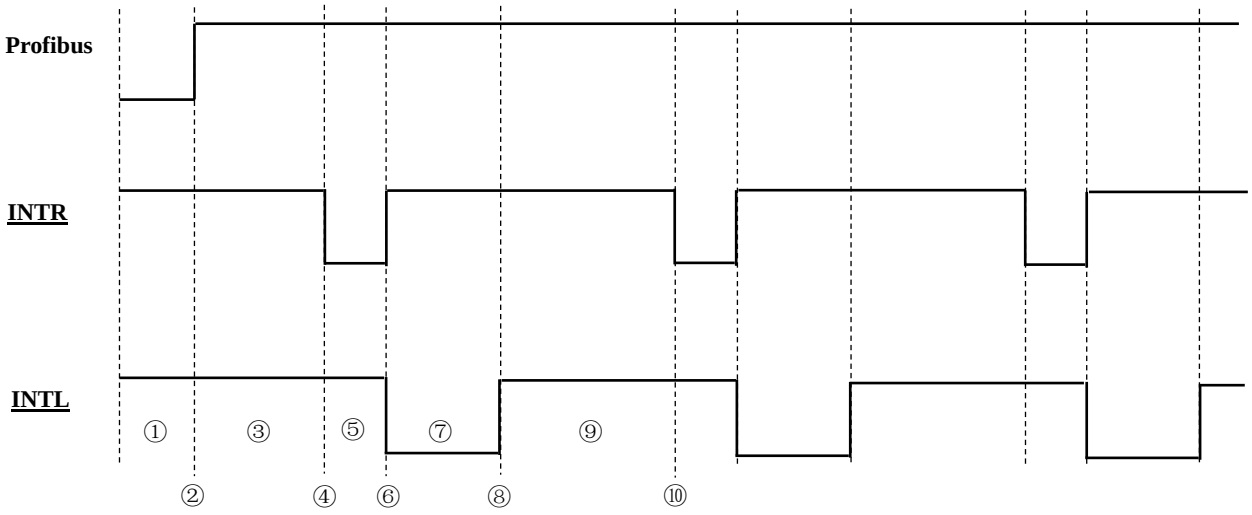
除初始化参数按表 4-2 外，整个过程同例 1。

⑸ 实验板初始化完成后 DPRAM 接口板的状态

同例 1。

⑹ 与 Profibus 主站连通后，实验板与 DPRAM 接口板的数据交换过程

基本同例 1，但在 Profibus 首次连通后，接口板可读取到用户参数数据。



① 接口板已完成初始化，但尚未与 Profibus 主站连通；
② 接口板与 Profibus 主站连通，进入数据交换状态，Profibus 红色状态指示灯灭；
③ 接口板与 SPC3 交换 In/Out 数据和 UserPrm 数据，向 DPRAM 写入 Out 数据和 UserPrm 数据；
④ 接口板向实验板发首次 INTR 中断；
⑤ 实验板执行中断处理程序，向 DPRAM 写入 In 数据，并从 DPRAM 读取 Out 数据和 UserPrm 数据；
⑥ 实验板中断程序结束，清除 INTR 中断、并向接口板发 INTL 中断；
⑦ 接口板执行中断处理程序，从 DPRAM 读取 In 数据；
⑧ 接口板中断程序结束，清除 INTL 中断；
⑨ 接口板再次与 SPC3 交换 In/Out 数据，向 DPRAM 写入 Out 数据；
⑩ 接口板向用户模板发下一次 INTR 中断；

⑺ 实验板程序清单（例 2）

```
/*
+---------------------------------------------------------------------+
| 文件名称:   AVR 开发实验板例 2 主程序                               |
| 版      本:  V1.0                                                   |
| 制作单位:   北京鼎实创新科技公司                                    |
| 网址：        www.c-profibus.com.cn                                 |
| email:        sunhm@c-profibus.com.cm                               |
| 日      期:  2005/01                                                |
+---------------------------------------------------------------------+
/*        以下头文件，请注意修改文件路径                              */
/*---------------------------------------------------------------------*/
// ICC-AVR application builder : 2003-6-8 10:39:03
// Target : M162
// Crystal: 8.000 MHz

#include <c:\icc\include\iom162v.h>
#include <c:\icc\include\macros.h>
#define UBYTE unsigned char
#define UWORD unsigned int

   UWORD      Ui,Vi,Wi;
   UBYTE      DI0_7,DI8_15;
   UBYTE      DO0_7,DO8_15;
   UBYTE      D_para[32] ;              //实验板用户参数接收区（来自 DPRAM）

#pragma abs_address:0x1000
struct {
UWORD Hand_Flag;                      //55AA-->硬件握手,00FF-->软件握手
UBYTE Lr;                             //实验板请求占用 DPRAM
UBYTE Ls;                             //实验板确认占用 DPRAM
UBYTE L_Flag1;                        //实验板状态字节 1
UBYTE L_Flag2;                        //实验板状态字节 2
UBYTE L_Inst1;                        //实验板命令字节 1
UBYTE L_Inst2;                        //实验板命令字节 2
}DRL ;                                //（来自实验板）
#pragma end_abs_address
#pragma abs_address:0x1010
UBYTE   L_ini[32] ;                   //初始化报文区（来自实验板）
#pragma end_abs_address
#pragma abs_address:0x1050
UBYTE   L_PBin[128] ;                 //Profibus 输入数据区（来自实验板）
#pragma end_abs_address

#pragma abs_address:0x1100
struct {
UWORD Spare1;                         //不用
UBYTE Sr;                             //接口板请求占用 DPRAM
UBYTE Ss;                             //接口板确认占用 DPRAM
UBYTE S_Flag1;                        //接口板状态字节 1
UBYTE S_Flag2;                        //接口板状态字节 2
}DRS ;                                //（来自接口板）
#pragma end_abs_address
#pragma abs_address:0x1110
UBYTE   S_para[32] ;                  //用户参数化报文区（来自接口板）
#pragma end_abs_address
#pragma abs_address:0x1150
UBYTE   S_PBout[128] ;                //Profibus 输出数据区（来自接口板）
#pragma end_abs_address
```

```
#pragma interrupt_handler Dpram_int:3          //双口 RAM 触发 INT1 中断

void Ctrl_On(void);                            //声明“申请 DPRAM 控制权”子程序
void Dpram_int(void);                          //声明“DPRAM 中断处理”子程序

void init_devices(void)
{
 UWORD x;

 PORTA = 0xFF;                                 //各口置为输入、上拉
 DDRA  = 0x00;
 PORTB = 0xF3;
 DDRB  = 0x0C;                                 //PB2、PB3 置为输出
 PORTC = 0xFF;
 DDRC  = 0x00;
 PORTD = 0xFF;
 DDRD  = 0x00;
 PORTE = 0x07;
 DDRE  = 0x00;

 for( x=0;x<30000;x++ ){};                     //实验板延时等待
 Ctrl_On();                                    //实验板申请 DPRAM 控制权
 MCUCR = 0x80;                                 //外部 RAM 允许
 EMCUCR = 0x00;
 PORTB = 0x00;                                 //实验板放弃 DPRAM 控制权
} //all peripherals are now initialised

/*=================================================================*/

void main(void)
{
  UBYTE      i,j,k,l,m,n,u,v,address;
  UWORD       mm,nn,x,y,z,nw;
  UBYTE      key0_7new, key8_15new;
  UBYTE      key0_7old, key8_15old;
  UBYTE      *ip,*op,*cp;
  UBYTE      *ADDRESS;
  UBYTE      *KEY_IN1,*KEY_IN2;
  UBYTE      *LED_IN1, *LED_IN2;
  UBYTE      *LED_OUT1,*LED_OUT2;

  CLI();                              //disable all interrupts
  init_devices();                     //初始化

  ip=(UBYTE*)0;
  op=(UBYTE*)0;
  LED_IN1=(UBYTE*)0;
  LED_IN2=(UBYTE*)0;
  LED_OUT1=(UBYTE*)0;
  LED_OUT2=(UBYTE*)0;
  KEY_IN1 =(UBYTE*)0;
  KEY_IN2 =(UBYTE*)0;
  ADDRESS =(UBYTE*)0;
  ip=ip+0x1000;                       // DPRAM 输入区首地址指针：      1000 H (读/写)
  op=op+0x1100;                       // DPRAM 输出区首地址指针：      1100 H (读/写)
  LED_IN1=LED_IN1+0x0801;             //输入绿 LED 指示灯 I0~I7 地址指针：  0801 H (写)
  LED_IN2=LED_IN2+0x0802;             //输入绿 LED 指示灯 I8~I15 地址指针：  0802 H (写)
  LED_OUT1=LED_OUT1+0x0803;           //输出红 LED 指示灯 Q0~Q7 地址指针：  0803 H (写)
  LED_OUT2=LED_OUT2+0x0804;           //输出红 LED 指示灯 Q8~Q15 地址指针：0804 H (写)
```

```
   ADDRESS =ADDRESS+0x0800;          //Profibus 地址开关 Address 地址指针：0800 H (读)
   KEY_IN1 =KEY_IN1+0x0801;          //输入按键开关 I0~I7 地址指针：     0801 H (读)
   KEY_IN2 =KEY_IN2+0x0802;          //输入按键开关 I0~I7 地址指针：     0802 H (读)
   cp=(UBYTE*)0;
   u=Wi=0;

x=0;y=1;
for( nn=0;nn<0x8000;nn++ )
    {
      *LED_IN1=y;                    //点绿 LED 灯
      *LED_IN2=y;                    //点绿 LED 灯
      *LED_OUT1=y;                   //点红 LED 灯
      *LED_OUT2=y;                   //点红 LED 灯
      if( nn/0x1000>x )
        {
          x=nn/0x1000;
          y=y*2+1;                   //亮灯左增长
        };
     };
   *LED_IN1=0;                       //绿 LED 灯灭
   *LED_IN2=0;                       //绿 LED 灯灭
   *LED_OUT1=0;                      //红 LED 灯灭
   *LED_OUT2=0;                      //红 LED 灯灭

   address=*ADDRESS;                 //读 Profibus 地址开关

   Ctrl_On();                        //获取 DPRAM 读写控制权
   *LED_OUT1=DRS.S_Flag1;            //显示 S_Flag1 状态（0x81）
   *LED_OUT2=DRS.S_Flag2;            //显示 S_Flag2 状态（0x00）
   PORTB=PORTB&0xF3;                 //释放 DPRAM 读写控制权
   for(x=0;x<500;x++)
      { for(y=0;y<1000;y++);}        //调试，暂停 1（维持状态显示 1~2 秒）

/*======================== 向接口板发“初始化数据” ======================*/

for(m=0;m==0; )
    {
      for(mm=0;mm==0;)               //发送“初始化数据”一次
         { Ctrl_On();                //实验板申请 DPRAM 控制权
           if( (DRS.S_Flag1&0x81)==0x81 )     //判接口板准备好标志
             {
               L_ini[0]=*ADDRESS;            //读地址开关
               L_ini[1]=0x06;                //ID 号 H
               L_ini[2]=0xfa;                //ID 号 L
               L_ini[3]=0x04;                //CFG 长度：  04
               L_ini[4]=0x51;                //CFG 数据 1：0x51
               L_ini[5]=0x61;                //CFG 数据 2：0x61
               L_ini[6]=0x1f;                //CFG 数据 3：0x1f
               L_ini[7]=0x2f;                //CFG 数据 4：0x2f
               L_ini[14]=20;                 //Profibus 输入数据长度：20
               L_ini[15]=20;                 //Profibus 输入数据长度：20
               L_ini[16]=5;                  //用户参数化报文长度：5
               x=0;
               for( i=0;i<17;i++ )           //计算“初始化校验和”
                   { x=x+L_ini[i];}
               L_ini[17]=(x&0xff00)/0x100;   //“初始化校验和”H
               L_ini[18]=x&0x00ff;           //“初始化校验和”L

               DRL.L_Flag1=0x83;             //置“实验板准备好”，“初始化数据有效”标志
               DRL.L_Inst1=0x81;                   //发初始化启动命令
```

```
                mm=1;
                }
              PORTB=PORTB&0xF3;                    //实验板放弃 DPRAM 控制权
             for( i=0;i<2000;i++ ){};              //延时
            }

      for(nn=0;nn==0;)                             //等待接口板初始化完成
         { Ctrl_On();                              //实验板申请 DPRAM 控制权
           if((DRL.L_Inst1==0x81)&&((DRS.S_Flag1&0x83)==0x83))      //若接口板初始化已完成
             {
               DRL.L_Inst1=0x00;                   //清初始化启动命令
               *LED_OUT1=0x00;                     //点灯
               *LED_OUT2=0x00;                     //点灯
               for( i=0;i<2000;i++ ){};            //延时
               nn=1;
               m=1;                                //“初始化”过程结束
               }
      else if((DRL.L_Inst1==0x81)&&((DRS.S_Flag2&0x81)==0x81))       //若初始化校验和错
               {
               DRL.L_Inst1=0x00;                   //清初始化启动命令
               DRS.S_Flag2=0x80;                   //清初始化校验和错

               *LED_OUT1=0xff;                     //点灯
               *LED_OUT2=0xff;                     //点灯
               for( i=0;i<2000;i++ ){};            //延时
               nn=1;
               m=0;                                //重新开始“初始化”过程
               }
             PORTB=PORTB&0xF3;                    //实验板放弃 DPRAM 控制权
            for( i=0;i<2000;i++ ){};              //延时
          };
    };
/*======================== 发“初始化数据”END ====================*/

   Ctrl_On();
   *LED_OUT1=DRS.S_Flag1;                          //显示 S_Flag1 状态（0x83）
   *LED_OUT2=DRS.S_Flag2;                          //显示 S_Flag2 状态（0x80）
   PORTB=PORTB&0xF3;
   for(x=0;x<500;x++)                              //调试，暂停 2（维持状态显示 1~2 秒）
       { for(y=0;y<1000;y++);}

for( x=0;x<10;x++ )                                //延时，LED 灯闪烁
 {
    for(y=0;y<7000;y++)
    {
    *LED_IN1=0xff;                                 //LED 灯亮
    *LED_IN2=0xff;
    *LED_OUT1=0xff;
    *LED_OUT2=0xff;
    };
    for(y=0;y<7000;y++)
    {
    *LED_IN1=0x00;                                 //LED 灯灭
    *LED_IN2=0x00;
    *LED_OUT1=0x00;
    *LED_OUT2=0x00;
    };
 };

   Ctrl_On();
```

```
   *LED_OUT1=DRS.S_Flag1;                    //显示 S_Flag1 状态（主站连通 0x8b，否则 0x83）
   *LED_OUT2=DRS.S_Flag2;                    //显示 S_Flag2 状态（0x80）
   PORTB=PORTB&0xF3;
   for(x=0;x<500;x++)                        //调试，暂停 3（维持状态显示 1~2 秒）
       { for(y=0;y<1500;y++);}

  SEI();                                     //开中断
  GICR= 0x80;                                //允许 INT1（低电平），屏蔽其余中断
  DI0_7 = DI8_15 = 0x00;                     //IO 变量清“0”

  for(;;)                                    //主循环开始
  {
/*====================键盘扫描==================================*/

       if( u==0 )
          { key0_7new = *KEY_IN1;
            if(key0_7new!=key0_7old)         //键 I0-I7 有动作（按下或抬起）
               {u=1;} }
else if( (u!=0)&&(u<=10) )
          { if(key0_7new == *KEY_IN1)        //连续 10 次稳定
                {u++;}
          else{u=0;} }
else if( u>10 )
          { if(key0_7new!=0)                 //键 I0-I7 有按下
               {DI0_7^=key0_7new;
                *LED_IN1=DI0_7; }
              key0_7old = key0_7new;
              u=0; }

       if( v==0 )
          { key8_15new = *KEY_IN2;
            if(key8_15new!=key8_15old)       //键 I8-I15 有动作（按下或抬起）
               {v=1;} }
else if( (v!=0)&&(v<=10) )
          { if(key8_15new == *KEY_IN2)       //连续 10 次稳定
                {v++;}
          else{v=0;} }
else if( v>10 )
          { if(key8_15new!=0)                //键 I8-I15 有按下
               {DI8_15^=key8_15new;
                *LED_IN2=DI8_15; }
              key8_15old = key8_15new;
              v=0; }

    *LED_IN1=DI0_7;                          //点绿 LED 输入灯（来自按键）
    *LED_IN2=DI8_15;
    *LED_OUT1=DO0_7;                         //点红 LED 输出灯（来自 Profibus 主站）
    *LED_OUT2=DO8_15;

/*============================  键盘扫描 END ================================*/
  };
}

/*---------------------------------- 取得双口 RAM 读写控制权(函数) ----------------------------------*/
void Ctrl_On(void)
{
  UBYTE     Ci;

  for(;((PORTB&0x0C)!=0x0C)||((PINB&0x03)!=0x00);)
  {
```

```
          if(((PORTB&0x04)==0x00)&&((PINB&0x03)==0x00))
             { PORTB=PORTB|0x04;}                          //实验板申请 DPRAM 控制权
    else if(((PORTB&0x04)==0x04)&&((PINB&0x03)!=0x00))
             { PORTB=PORTB&0xF3;}                          //实验板放弃申请
    else if(((PORTB&0x04)==0x04)&&((PINB&0x03)==0x00))
             { PORTB=PORTB|0x08;}                          //实验板取得 DPRAM 控制权
  }
}
/*------------------------------------ 取得双口 RAM 控制权 END ----------------------------------------*/

//======================== 双口 RAM 中断处理程序 ========================//
void Dpram_int(void)
{
   UBYTE      i;
   UWORD      x,y;
   UBYTE      *ip;
   UBYTE      *op;

   ip=(UBYTE*)0;
   op=(UBYTE*)0;
   ip=ip+0x1000;
   op=op+0x1100;
//=================== 将“输入数据”传送到双口 RAM 的 L_PBin[] 区 ====================//

       Ctrl_On();                                          //实验板申请 DPRAM 控制权
       if( ((DRS.S_Flag1&0x83)==0x83)&&(DRS.S_Flag2==0x80) )     //若接口板“初始化已完成”，无错
         {
            L_PBin[0]=L_ini[14];                           //Profibus 输入数据长度
            L_PBin[1]=DI0_7;                               //传送 键盘“输入数据” 1
            L_PBin[2]=DI8_15;                              //传送 键盘“输入数据” 2
            x=L_PBin[0]+L_PBin[1]+L_PBin[2];
            for( i=3;i<=L_PBin[0];i++ )
                { L_PBin[i]=i;                             //填写随机数
                  x=x+i;
                }
            L_PBin[i]=x/0x100;                             //“输入数据”校验和 H
            L_PBin[i+1]=x-L_PBin[i]*0x100;                 //“输入数据”校验和 L
         }
       DRL.L_Flag1=DRL.L_Flag1|0x84;                       // 1-->“输入数据有效”标志

//------------------------------------ “输入数据”传送到 L_PBin[] 结束 ----------------------------------------//

//======================== 将主站“输出数据”送 LED 显示 ============================//

       if( ((DRS.S_Flag1&0x8b)==0x8b)&&(DRS.S_Flag2==0x80) )     //若接口板“输出数据”有效
         { x=0;
            for( i=0;i<=S_PBout[0];i++ )
                { x=x+S_PBout[i]; }                        //计算“输出数据”校验和
            y=S_PBout[i]*0x100+S_PBout[i+1];               //读取“输出数据”校验和

            if( x==y )
              {
                DO0_7=S_PBout[1];                          //传送“输出数据”
                DO8_15=S_PBout[2];
              }
             DRS.S_Flag1=DRS.S_Flag1&0xf7;                 //清“输出数据”有效标志
         }

//------------------------------------------- “输出数据”显示结束 ------------------------------------------------//
```

```
//============================== 传送“用户参数数据”==============================//

      if( ((DRS.S_Flag1&0x87)==0x87)&&(DRS.S_Flag2==0x80) )     //若接口板“用户参数数据”有效
        { x=0;
          for( i=0;i<L_ini[16];i++ )
              { x=x+S_para[i]; }                          //计算“用户参数数据”校验和
          y=S_para[i]*0x100+S_para[i+1];                  //读取“用户参数数据”校验和
          if( x==y )
            {
              for( i=0;i<L_ini[16];i++ )
                  { D_para[i]=S_para[i]; }                //传送“用户参数数据”
            }
          DRS.S_Flag1=DRS.S_Flag1&0xfb;                   //清“用户参数数据”有效标志
         }
//---------------------------------------------- 传送“用户参数数据”结束 ----------------------------------------------//

     x=*(ip+0x03ff);                                      //读 DPRAM 清除中断源
     *(ip+0x03fe)=0x55;                                   //写 DPRAM 向接口板发中断
      PORTB=PORTB&0xF3;                                   //实验板放弃 DPRAM 控制权
}//external interupt on INT1
```

⑻ 例 2 的 GSD 文件

```
;=============================================================
; GSD文件 : PB-OEM1-DPRAM -（2字输入/2字输出 + 16字节输入/16字节输出）/SPC3
; 制造厂商： 北京鼎实创新科技有限责任公司
; 产品型号: PB-OEM1-DPRAM
; 版本    : 2005年1月 例2
; 文件名  : OEM1-例2.GSD
;=============================================================
#Profibus_DP
; <Prm-Text-Def-List>
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)="-10--+10mA"
EndPrmText
;PrmText=2
;滤波系数
;EndPrmText
PrmText=3
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=4
Text(0)="单端"
Text(1)="双端"
EndPrmText

; <Ext-User-Prm-Data-Def-List>
ExtUserPrmData=10 "输入类型AI_M："
Unsigned8 1 0-6
Prm_Text_Ref=1
EndExtUserPrmData
ExtUserPrmData=20 "输入滤波系数AI_FK："
Unsigned16 50 0-1000
EndExtUserPrmData
ExtUserPrmData=30 "输入数据类型AI_D："
Unsigned8 1 0-2
Prm_Text_Ref=3
EndExtUserPrmData
ExtUserPrmData=40 "输入单端/双端AI_CH："
Unsigned8 0 0-1
Prm_Text_Ref=4
EndExtUserPrmData

; <Unit-Definition-List>
GSD_Revision=2
Vendor_Name="DS FieldBus Ltd. Co."              ;公司名，可按用户公司名修改
Model_Name="DPRAM_例2"                          ;模块名称，也是在Step7等软件中组态时该产品的名称
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="DPRAM"                          ;图标文件，用户可以自制图标，亦可缺省
; Slave-Specification:
OrderNumber="PB-OEM1-DPRAM"                    ;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=3
Max_Diag_Data_Len=6

; UserPrmData: Length and Preset:
User_Prm_Data_Len=5                            ;用户参数长度
; User_Prm_Data=0x00,0x0032,0x01,0x00          ;用户参数默认值(友好界面设计时，可删除)
Ext_User_Prm_Data_Ref(0)=10
Ext_User_Prm_Data_Ref(1)=20
Ext_User_Prm_Data_Ref(3)=30
Ext_User_Prm_Data_Ref(4)=40

Modular_Station = 1                            ; 模块型从站
Max_Module = 2                                 ; Max. 2 Module
Max_Input_Len = 20                             ; Max. 20 Byte Input-Daten
Max_Output_Len = 20                            ; Max. 20 Byte Output-Daten
Max_Data_Len = 40                              ; Max. 40 Byte Gesamt-Daten
Modul_Offset=0

Slave_Family=0@DPRAM                           ;组态中的分类名
; <Module-Definition-List>
Module=" 2 Word AI, 2 Word AO "    0x51,0x61   ;I/O配置数据必须与初始化报文中的CFG一致
EndModule
Module=" 16 Byte DI, 16 Byte DO "   0x1f,0x2f
EndModule
```

```
/*
+---------------------------------------------------------------------+
| 文件名称:   AVR 开发实验板例 2 主程序                               |
| 版      本:  V1.0                                                   |
| 制作单位:   北京鼎实创新科技公司                                    |
| 网址：      www.c-profibus.com.cn                                   |
| email:      sunhm@c-profibus.com.cm                                 |
| 日      期:  2005/01                                                |
+---------------------------------------------------------------------+
/*        以下头文件，请注意修改文件路径                              */
/*---------------------------------------------------------------------*/
// ICC-AVR application builder : 2003-6-8 10:39:03
// Target : M162
// Crystal: 8.000 MHz

#include <c:\icc\include\iom162v.h>
#include <c:\icc\include\macros.h>
#define UBYTE unsigned char
#define UWORD unsigned int

   UWORD      Ui,Vi,Wi;
   UBYTE      DI0_7,DI8_15;
   UBYTE      DO0_7,DO8_15;
   UBYTE      D_para[32] ;                  //实验板用户参数接收区（来自 DPRAM）

#pragma abs_address:0x1000
struct {
UWORD Hand_Flag;                          //55AA-->硬件握手,00FF-->软件握手
UBYTE Lr;                                 //实验板请求占用 DPRAM
UBYTE Ls;                                 //实验板确认占用 DPRAM
UBYTE L_Flag1;                            //实验板状态字节 1
UBYTE L_Flag2;                            //实验板状态字节 2
UBYTE L_Inst1;                            //实验板命令字节 1
UBYTE L_Inst2;                            //实验板命令字节 2
}DRL ;                                    //（来自实验板）
#pragma end_abs_address
#pragma abs_address:0x1010
UBYTE   L_ini[32] ;                       //初始化报文区（来自实验板）
#pragma end_abs_address
#pragma abs_address:0x1050
UBYTE   L_PBin[128] ;                     //Profibus 输入数据区（来自实验板）
#pragma end_abs_address

#pragma abs_address:0x1100
struct {
UWORD Spare1;                             //不用
UBYTE Sr;                                 //接口板请求占用 DPRAM
UBYTE Ss;                                 //接口板确认占用 DPRAM
UBYTE S_Flag1;                            //接口板状态字节 1
UBYTE S_Flag2;                            //接口板状态字节 2
}DRS ;                                    //（来自接口板）
#pragma end_abs_address
#pragma abs_address:0x1110
UBYTE   S_para[32] ;                      //用户参数化报文区（来自接口板）
#pragma end_abs_address
#pragma abs_address:0x1150
UBYTE   S_PBout[128] ;                    //Profibus 输出数据区（来自接口板）
#pragma end_abs_address

#pragma interrupt_handler Dpram_int:3          //双口 RAM 触发 INT1 中断
```

```
void Ctrl_On(void);
void Dpram_int(void);

void init_devices(void)
{
 UWORD x;

 PORTA = 0xFF;                          //各口置为输入、上拉
 DDRA  = 0x00;
 PORTB = 0xF3;
 DDRB  = 0x0C;                          //PB2、PB3 置为输出
 PORTC = 0xFF;
 DDRC  = 0x00;
 PORTD = 0xFF;
 DDRD  = 0x00;
 PORTE = 0x07;
 DDRE  = 0x00;

 for( x=0;x<30000;x++ ){};              //实验板延时等待
 Ctrl_On();                             //实验板申请 DPRAM 控制权
 MCUCR = 0x80;                          //外部 RAM 允许
 EMCUCR = 0x00;
 PORTB = 0x00;                          //实验板放弃 DPRAM 控制权
} //all peripherals are now initialised

/*============================================================*/

void main(void)
{
  UBYTE      i,j,k,l,m,n,u,v,address;
  UWORD       mm,nn,x,y,z,nw;
  UBYTE      key0_7new, key8_15new;
  UBYTE      key0_7old, key8_15old;
  UBYTE      *ip,*op,*cp;
  UBYTE      *ADDRESS;
  UBYTE      *KEY_IN1,*KEY_IN2;
  UBYTE      *LED_IN1, *LED_IN2;
  UBYTE      *LED_OUT1,*LED_OUT2;

  CLI();                            //disable all interrupts
  init_devices();                   //初始化

  ip=(UBYTE*)0;
  op=(UBYTE*)0;
  LED_IN1=(UBYTE*)0;
  LED_IN2=(UBYTE*)0;
  LED_OUT1=(UBYTE*)0;
  LED_OUT2=(UBYTE*)0;
  KEY_IN1 =(UBYTE*)0;
  KEY_IN2 =(UBYTE*)0;
  ADDRESS =(UBYTE*)0;
  ip=ip+0x1000;                     // DPRAM 输入区首地址指针：      1000 H (读/写)
  op=op+0x1100;                     // DPRAM 输出区首地址指针：      1100 H (读/写)
  LED_IN1=LED_IN1+0x0801;           //输入绿 LED 指示灯 I0~I7 地址指针：  0801 H (写)
  LED_IN2=LED_IN2+0x0802;           //输入绿 LED 指示灯 I8~I15 地址指针：  0802 H (写)
  LED_OUT1=LED_OUT1+0x0803;         //输出红 LED 指示灯 Q0~Q7 地址指针：  0803 H (写)
  LED_OUT2=LED_OUT2+0x0804;         //输出红 LED 指示灯 Q8~Q15 地址指针：0804 H (写)
  ADDRESS =ADDRESS+0x0800;          //Profibus 地址开关 Address 地址指针：0800 H (读)
  KEY_IN1 =KEY_IN1+0x0801;          //输入按键开关 I0~I7 地址指针：      0801 H (读)
```

```
   KEY_IN2 =KEY_IN2+0x0802;          //输入按键开关 I0~I7 地址指针：        0802 H (读)
   cp=(UBYTE*)0;
   u=Wi=0;

x=0;y=1;
for( nn=0;nn<0x8000;nn++ )
    {
      *LED_IN1=y;                    //点绿 LED 灯
      *LED_IN2=y;                    //点绿 LED 灯
      *LED_OUT1=y;                   //点红 LED 灯
      *LED_OUT2=y;                   //点红 LED 灯
      if( nn/0x1000>x )
        {
          x=nn/0x1000;
          y=y*2+1;                   //亮灯左增长
        };
     };
   *LED_IN1=0;                       //绿 LED 灯灭
   *LED_IN2=0;                       //绿 LED 灯灭
   *LED_OUT1=0;                      //红 LED 灯灭
   *LED_OUT2=0;                      //红 LED 灯灭

   address=*ADDRESS;                 //读 Profibus 地址开关

   Ctrl_On();                        //获取 DPRAM 读写控制权
   *LED_OUT1=DRS.S_Flag1;            //显示 S_Flag1 状态字节
   *LED_OUT2=DRS.S_Flag2;            //显示 S_Flag2 状态字节
   PORTB=PORTB&0xF3;                 //释放 DPRAM 读写控制权
   for(x=0;x<500;x++)
       { for(y=0;y<1000;y++);}       //调试，暂停 1（维持显示）

/*======================== 向接口板发“初始化数据” ====================*/

for(m=0;m==0; )
    {
      for(mm=0;mm==0;)               //发送“初始化数据”一次
          { Ctrl_On();               //实验板申请 DPRAM 控制权
            if( (DRS.S_Flag1&0x81)==0x81 )    //判接口板准备好标志
              {
                L_ini[0]=*ADDRESS;          //读地址开关
                L_ini[1]=0x06;              //ID 号 H
                L_ini[2]=0xfa;              //ID 号 L
                L_ini[3]=0x08;              //CFG 长度：  04
                L_ini[4]=0x1f;              //CFG 数据 1：40
                L_ini[5]=0x2f;              //CFG 数据 2：77
                L_ini[6]=0x5f;              //CFG 数据 3：80
                L_ini[7]=0x6f;              //CFG 数据 4：77
                L_ini[8]=0x5f;              //CFG 数据 5
                L_ini[9]=0x6f;              //CFG 数据 6
                L_ini[10]=0x5f;             //CFG 数据 7
                L_ini[11]=0x6f;             //CFG 数据 8
                L_ini[12]=0x17;             //CFG 数据 9
                L_ini[13]=0x27;             //CFG 数据 10
                L_ini[14]=112;              //Profibus 输入数据长度：112
                L_ini[15]=112;              //Profibus 输入数据长度：112
                L_ini[16]=16;                    //用户参数化报文长度
                x=0;
                for( i=0;i<17;i++ )         //计算“初始化校验和”
                    { x=x+L_ini[i];}
                L_ini[17]=(x&0xff00)/0x100;     //“初始化校验和”H
```

```
                L_ini[18]=x&0x00ff;              //“初始化校验和”L

                DRL.L_Flag1=0x83;                //置实验板准备好标志，初始化数据有效
                DRL.L_Inst1=0x81;                //发初始化启动命令
                mm=1;
                }
              PORTB=PORTB&0xF3;                          //实验板放弃 DPRAM 控制权
             for( i=0;i<2000;i++ ){};            //延时
            }

      for(nn=0;nn==0;)                           //等待接口板初始化完成
          { Ctrl_On();                           //实验板申请 DPRAM 控制权
            if((DRL.L_Inst1==0x81)&&((DRS.S_Flag1&0x83)==0x83))      //接口板初始化已完成
              {
                DRL.L_Inst1=0x00;                //清初始化启动命令
                *LED_OUT1=0x00;                  //点灯
                *LED_OUT2=0x00;                  //点灯
                for( i=0;i<2000;i++ ){};         //延时
                nn=1;
                m=1;                             //“初始化”过程结束
                }
      else if((DRL.L_Inst1==0x81)&&((DRS.S_Flag2&0x81)==0x81))  //初始化校验和错
              {
                DRL.L_Inst1=0x00;                //清初始化启动命令
                DRS.S_Flag2=0x80;                //清初始化校验和错

                *LED_OUT1=0xff;                  //点灯
                *LED_OUT2=0xff;                  //点灯
                for( i=0;i<2000;i++ ){};         //延时
                nn=1;
                m=0;                             //重新开始“初始化”过程
                }
              PORTB=PORTB&0xF3;                          //实验板放弃 DPRAM 控制权
             for( i=0;i<2000;i++ ){};            //延时
          };
    };
/*========================  发“初始化数据”END ======================*/

   Ctrl_On();                          //调试，暂停 2
   *LED_OUT1=DRS.S_Flag1;
   *LED_OUT2=DRS.S_Flag2;
   PORTB=PORTB&0xF3;
   for(x=0;x<500;x++)
       { for(y=0;y<1000;y++);}


for( x=0;x<10;x++ )                            //延时
  {
    for(y=0;y<7000;y++)
    {
    *LED_IN1=0xff;                             //LED 灯闪烁
    *LED_IN2=0xff;
    *LED_OUT1=0xff;
    *LED_OUT2=0xff;
    };
    for(y=0;y<7000;y++)
    {
    *LED_IN1=0x00;                             //LED 灯闪烁
    *LED_IN2=0x00;
    *LED_OUT1=0x00;
```

```
    *LED_OUT2=0x00;
    };
 };

  Ctrl_On();                          //调试，暂停 3
  *LED_OUT1=DRS.S_Flag1;
  *LED_OUT2=DRS.S_Flag2;
  PORTB=PORTB&0xF3;
  for(x=0;x<500;x++)
      { for(y=0;y<1500;y++);}

 SEI();                                                  //开中断
 GICR= 0x80;                                             //允许 INT1（低电平），屏蔽其余中断
 DI0_7 = DI8_15 = 0x00;                                  //IO 变量清“0”

 for(;;)                                                 //主循环
 {
/*====================键盘扫描==================================*/

      if( u==0 )
        { key0_7new = *KEY_IN1;
          if(key0_7new!=key0_7old)                       //键 0-7 有动作
             {u=1;} }
else if( (u!=0)&&(u<=10) )
        { if(key0_7new == *KEY_IN1)                      //连续 10 次稳定
             {u++;}
        else{u=0;} }
else if( u>10 )
        { if(key0_7new!=0)                               //键 0-7 有按下
             {DI0_7^=key0_7new;
              *LED_IN1=DI0_7; }
           key0_7old = key0_7new;
           u=0; }

      if( v==0 )
        { key8_15new = *KEY_IN2;
          if(key8_15new!=key8_15old)                     //键 8-15 有动作
             {v=1;} }
else if( (v!=0)&&(v<=10) )
        { if(key8_15new == *KEY_IN2)                     //连续 10 次稳定
             {v++;}
        else{v=0;} }
else if( v>10 )
        { if(key8_15new!=0)                              //键 8-15 有按下
             {DI8_15^=key8_15new;
              *LED_IN2=DI8_15; }
           key8_15old = key8_15new;
           v=0; }

for(x=0;x<1000;x++){};
/*
    z=z+1;                          //调试，变量递增
    if( z>=2500 )
      {
        z=0;
         DI0_7=DI0_7+1;
         DI8_15=DI8_15+1;
       }
*/
    *LED_IN1=DI0_7;                 //输出绿灯
```

```
    *LED_IN2=DI8_15;
    *LED_OUT1=DO0_7;          //输出红灯
    *LED_OUT2=DO8_15;

/*====================键盘扫描 END==================================*/
 };
}

/*------------取得双口 RAM 读写控制权(函数)-----------------*/
void Ctrl_On(void)
{
 UBYTE     Ci;

 for(;((PORTB&0x0C)!=0x0C)||((PINB&0x03)!=0x00);)
 {
         if(((PORTB&0x04)==0x00)&&((PINB&0x03)==0x00))
           { PORTB=PORTB|0x04;}                          //实验板申请 DPRAM 控制权
  else if(((PORTB&0x04)==0x04)&&((PINB&0x03)!=0x00))
           { PORTB=PORTB&0xF3;}                          //实验板放弃申请
  else if(((PORTB&0x04)==0x04)&&((PINB&0x03)==0x00))
           { PORTB=PORTB|0x08;}                          //实验板取得 DPRAM 控制权
 }
}
/*------------------  取得双口 RAM 控制权 END ---------------------*/

//===================  双口 RAM 中断处理程序  =====================//
void Dpram_int(void)
{
  UBYTE      i;
  UWORD      x,y;
  UBYTE      *ip;
  UBYTE      *op;

  ip=(UBYTE*)0;
  op=(UBYTE*)0;
  ip=ip+0x1000;
  op=op+0x1100;
//===================  将“输入数据”传送到双口 RAM 的  L_PBin[]  区  ====================//

      Ctrl_On();                                          //实验板申请 DPRAM 控制权
      if( ((DRS.S_Flag1&0x83)==0x83)&&(DRS.S_Flag2==0x80) )     //接口板“初始化已完成”，无错
        {
          L_PBin[0]=L_ini[14];                      //Profibus 输入数据长度
          L_PBin[1]=DI0_7;                          //读  键盘  IN  数据  1
          L_PBin[2]=DI8_15;                         //读  键盘  IN  数据  2
          x=L_PBin[0]+L_PBin[1]+L_PBin[2];
          for( i=3;i<=L_PBin[0];i++ )
              { L_PBin[i]=i;                        //填写随机数
                x=x+i;
              }
          L_PBin[i]=x/0x100;                        //“输入数据”校验和 H
          L_PBin[i+1]=x-L_PBin[i]*0x100;            //“输入数据”校验和 L
        }
      DRL.L_Flag1=DRL.L_Flag1|0x84;                 // 1-->“输入数据有效”标志

//----------------------  “输入数据”传送到  L_PBin[]  结束  ------------------------//
//======================  将主站“输出数据”送  LED  显示  ==========================//
/*
  if( ((DO8_15^DRS.S_Flag1)!=0)&&(Vi>=20) )   //调试，观察 S_Flag1
     { DO8_15=DRS.S_Flag1;   //调试时后面的“DO8_15=S_PBout[2];”须删除
```

```
        Vi=0;}
else{ Vi++;}                                          //调试，观察 S_Flag1
*/
      if( ((DRS.S_Flag1&0x8b)==0x8b)&&(DRS.S_Flag2==0x80) )     //接口板“输出数据”有效
        { x=0;
          for( i=0;i<=S_PBout[0];i++ )
              { x=x+S_PBout[i]; }                              //计算“输出数据”校验和
          y=S_PBout[i]*0x100+S_PBout[i+1];                  //读取“输出数据”校验和

          if( x==y )
            {
              DO0_7=S_PBout[1];
              DO8_15=S_PBout[2];
            }
           DRS.S_Flag1=DRS.S_Flag1&0xf7;                  //清“输出数据”标志
        }

//------------------------------  “输出数据”显示结束  ------------------------------//
//============================== 传送“用户参数数据” ==============================//

      if( ((DRS.S_Flag1&0x87)==0x87)&&(DRS.S_Flag2==0x80) )//接口板“用户参数数据”有效
        { x=0;
          for( i=0;i<L_ini[16];i++ )
              { x=x+S_para[i]; }                           //计算“用户参数数据”校验和
          y=S_para[i]*0x100+S_para[i+1];                   //读取“用户参数数据”校验和
          if( x==y )
            {
              for( i=0;i<L_ini[16];i++ )
                  { D_para[i]=S_para[i]; }                 //传送“用户参数数据”
            }
          DRS.S_Flag1=DRS.S_Flag1&0xfb;                    //清“用户参数数据”标志
         }
//--------------------------- 传送“用户参数数据”结束  ---------------------------//

     x=*(ip+0x03ff);                                       //读 DPRAM 清除中断源
     *(ip+0x03fe)=0x55;                                    //写 DPRAM 向接口板发中断
      PORTB=PORTB&0xF3;                                        //实验板放弃 DPRAM 控制权
}//external interupt on INT1
```

例 2.　2 字输入/2 字输出 + 16 字节输入/16 字节输出（带 10 字节用户参数）

例 2 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板 I/O。实验板 I/O 设计将使用“用户参数”功能。由于带用户参数时的 DPRAM 接口初始化、和数据交换过程与例 1 中基本类似。因此本例将讲解的重点放在用户参数功能的使用场合及相应 GSD 文件的编写方法上。

⑴ 什么情况下需要使用“用户参数 user_prm”

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；如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。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）。

⑵ 实验板 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，称为实验板 I/O

② 实验板 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），将实验板 I/O 配置成具有 1 路 AI + 1 路 AO 的模拟量、和 16DI + 16DO 的开关量的 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；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 实现上述功能方法是：

- 在实验板 I/O 的 GSD 文件中详细描述用于配置的用户参数；
- 主站配置中由用户选择这些配置参数；
- 当主站与从站实验板 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板 I/O；
- 实验板 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 I、和 16DI + 16DO 开关量 I/O 的功能。

⑶ 具体确定“用户参数”类型、个数、取值范围

根据实验板 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：

用户参数	代码	取值范围	物理意义
1	AI_M	0 ~ 4	代表输入类型 0~5V、0~10V、–10~+10V、0~20mA、–10~+10 mA，共五种选择；
2	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	AI_CH	0 ~ 1	分别表示单端输入、双端输入，共二种选择；
4	AO_M	0 ~ 4	代表输出类型 0~5V、0~10V、–10~+10V、0~20mA、–10~+10 mA，共五种选择；
5	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	AO_CH	0 ~ 1	分别表示单端输出、双端输出，共二种选择；
7	DI_PN	0 ~ 1	分别表示正逻辑（高有效）、负逻辑（低有效），2种输入模式；
8	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	DO_PN	0 ~ 1	分别表示正逻辑（高有效）、负逻辑（低有效），2 种输出模式；
10	DO_V	0 ~ 3	分别表示 5VDC/0.5A、24VDC/0.5A、110VAC/0.5A、220VAC/0.5A，4 种输出电压等级；

表 4-2

由表 4-2 可见，本例需要 10 个字节用户参数；用户参数长度 User_Prm_Data_Len=10。

⑷ 带有“用户参数”描述的 GSD 文件

与例 1 相比，实验板 I/O 的 GSD 文件增加了如下关于“用户参数”的描述，见下划线部分；下面是关于实验板 I/O 一个简单的、带有“用户参数”描述的 GSD 文件：

```
;============================================================
; GSD文件 : PB-OEM1-DPRAM -（2字输入/2字输出 + 16字节输入/16字节输出） 鼎实创新科技有限责任公司
; 产品型号:  PB-OEM1-DPRAM
; 版本      :  2005年1月 例2
; 文件名   :  OEM1-例2.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，可按用户公司名修改
Model_Name="DPRAM_例2"                      ;模块名称，也是在Step7等软件中组态时该产品的名称
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="DPRAM"                       ;图标文件，用户可以自制图标，亦可缺省
```

```
; Slave-Specification:
OrderNumber="PB-OEM1-DPRAM"                    ;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=3
Max_Diag_Data_Len=6
User_Prm_Data_Len=10                                          ;用户参数长度
User_Prm_Data=0x01,0x00,0x00,0x01,0x00,0x00,0x00,0x01,0x00,0x01    ;用户参数默认值

Modular_Station = 1                          ; 模块型从站
Max_Module = 2                               ; Max. 2 Module
Max_Input_Len = 20                           ; Max. 20 Byte Input-Daten
Max_Output_Len = 20                          ; Max. 20 Byte Output-Daten
Max_Data_Len = 40                            ; Max. 40 Byte Gesamt-Daten
Modul_Offset=0
Slave_Family=0@DPRAM                                ;组态中的分类名
; <Module-Definition-List>
Module=" 2 Word AI, 2 Word AO "   0x51,0x61          ;I/O配置数据必须与初始化报文中的CFG一致
EndModule
Module=" 16 Byte DI, 16 Byte DO "   0x1f,0x2f
EndModule
```

这是实现实验板I/O最简单的一种GSD文件，但用户使用起来却并不方便。用户在主站上对实验板I/O进行参数配置时，需要在表4-2中逐个查对每个用户参数的取值和物理意义，才能完成实验板I/O的配置。很不直观。

表4-1：实验板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 0~5V、0~10V、–10~+10V、0~20mA、–10~+10 mA 共五种选择；
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

实验板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；

⑸ 如何在主站配置中选择用户参数

首先以STEP 7为例：

⑤ 实验板I/O的GSD文件OEM1_B3.GSD拷至：SIEMENS\step7\S7data\gsd\

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

⑥ 进入SIMATIC Manager → HARDWARE，选择Options → Updata Catalog;

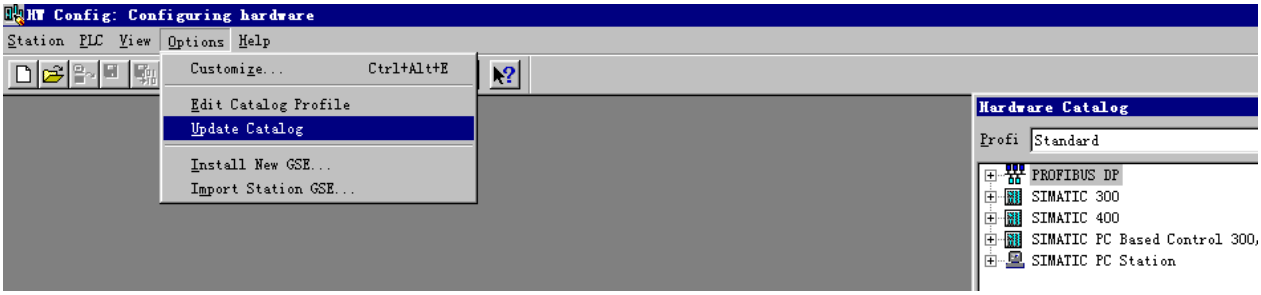


图 4-3 选择 Options → Updata Catalog

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

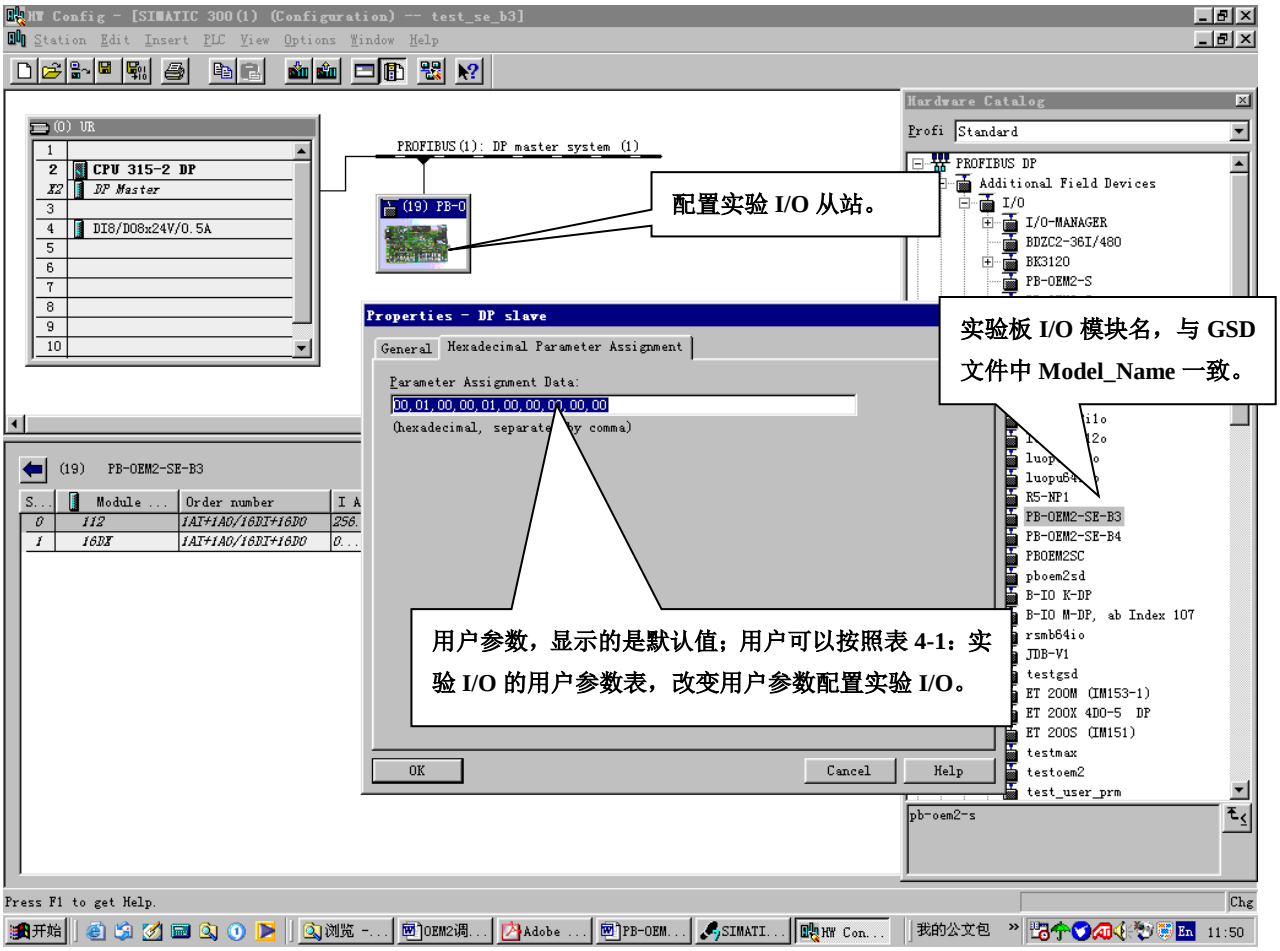


图 4-4：STEP 7—用户可以按照表 4-1 改变用户参数配置实验 I/O

⑹便于用户使用的 GSD 文件

上述实验板 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：用户可以按照表 4-1 改变用户参数配置实验板 I/O”，这对用户来说极不方便。下面是改进的实验板 I/O GSD 文件；可在配置中通过菜单配置用户参数；这对用户来说比较方便，但 GSD 文件结构相对复杂。

```
;=============================================================
; GSD 文件 ： OEM1-B4 实验板 I/O 鼎实创新科技有限责任公司
; 产品型号: PB-OEM1-实验板 I/O; 版本 : 2004 年 9 月 V1.0
; 文件名 : OEM1-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-OEM1-DPRAM-B4"               ;模块名称，也是组态时该产品的名称
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="DPRAM_B"                     ;图标文件，用户可以自制图标，缺省
; Slave-Specification:
OrderNumber="pb-OEM1-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-OEM1                  ;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：

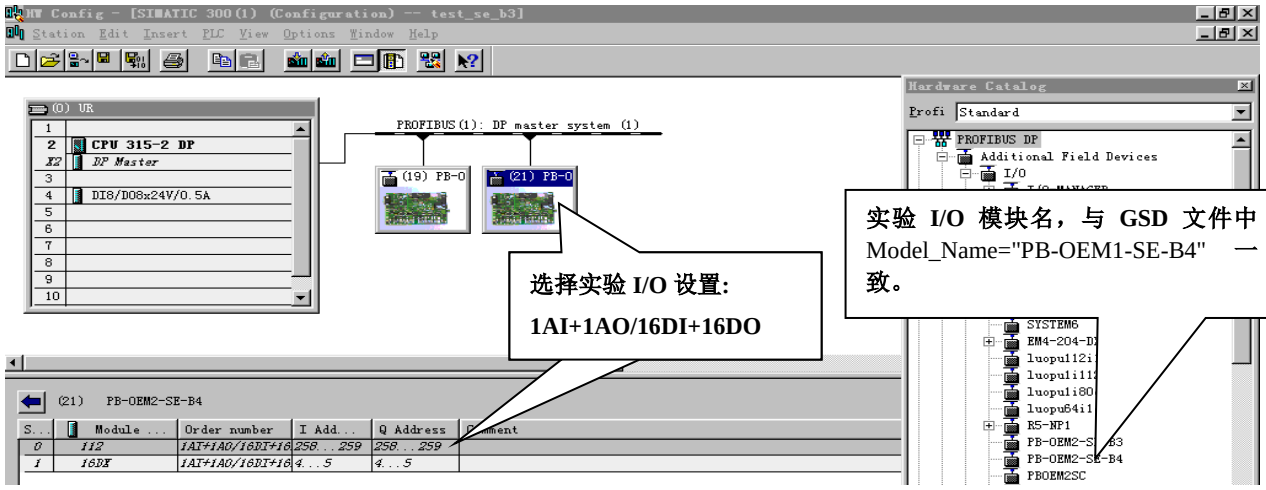


图 4-6: 选择实验 I/O 设置:1AI+1AO/16DI+16DO

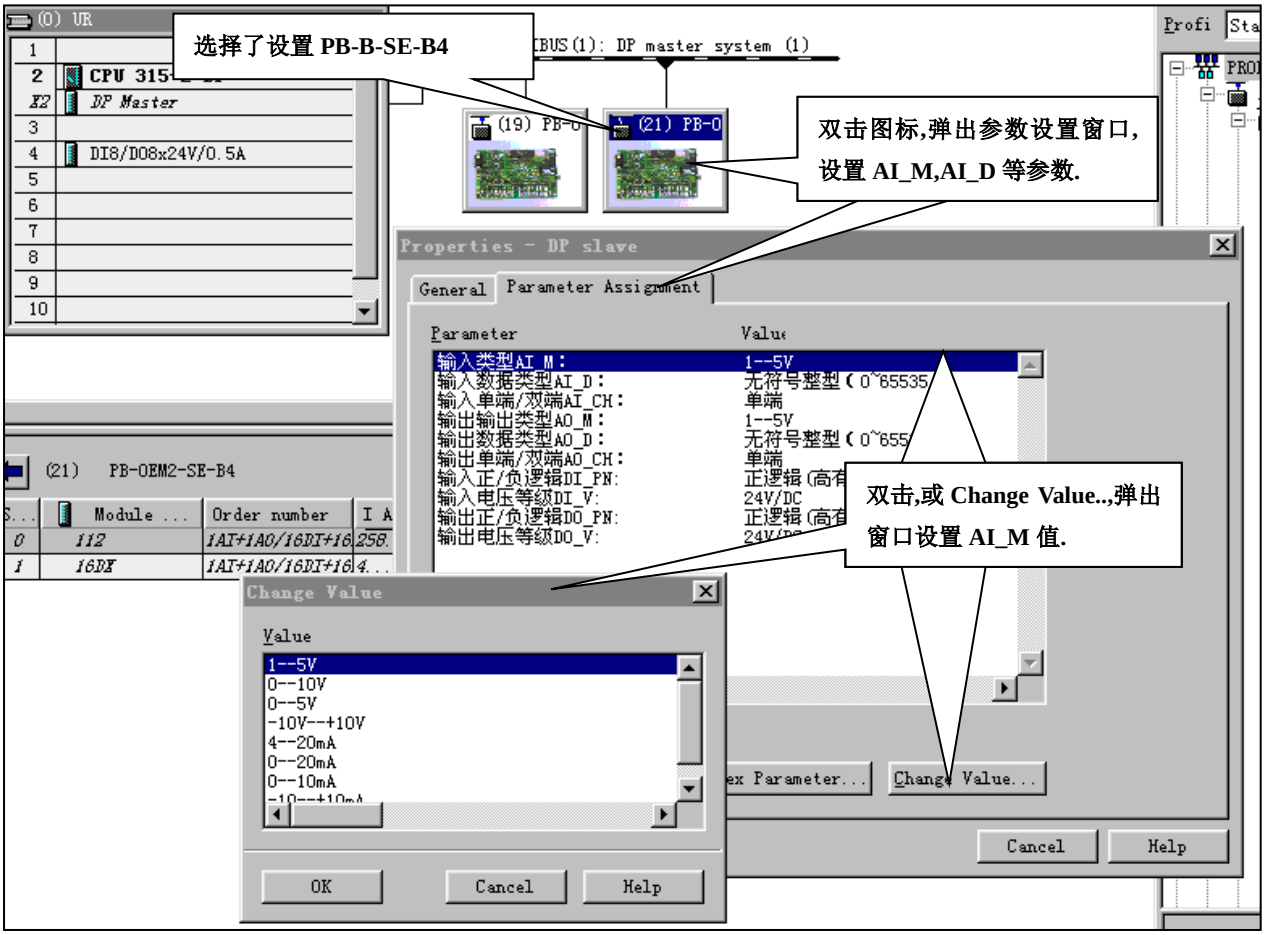


图 4-7: 进一步的参数设置

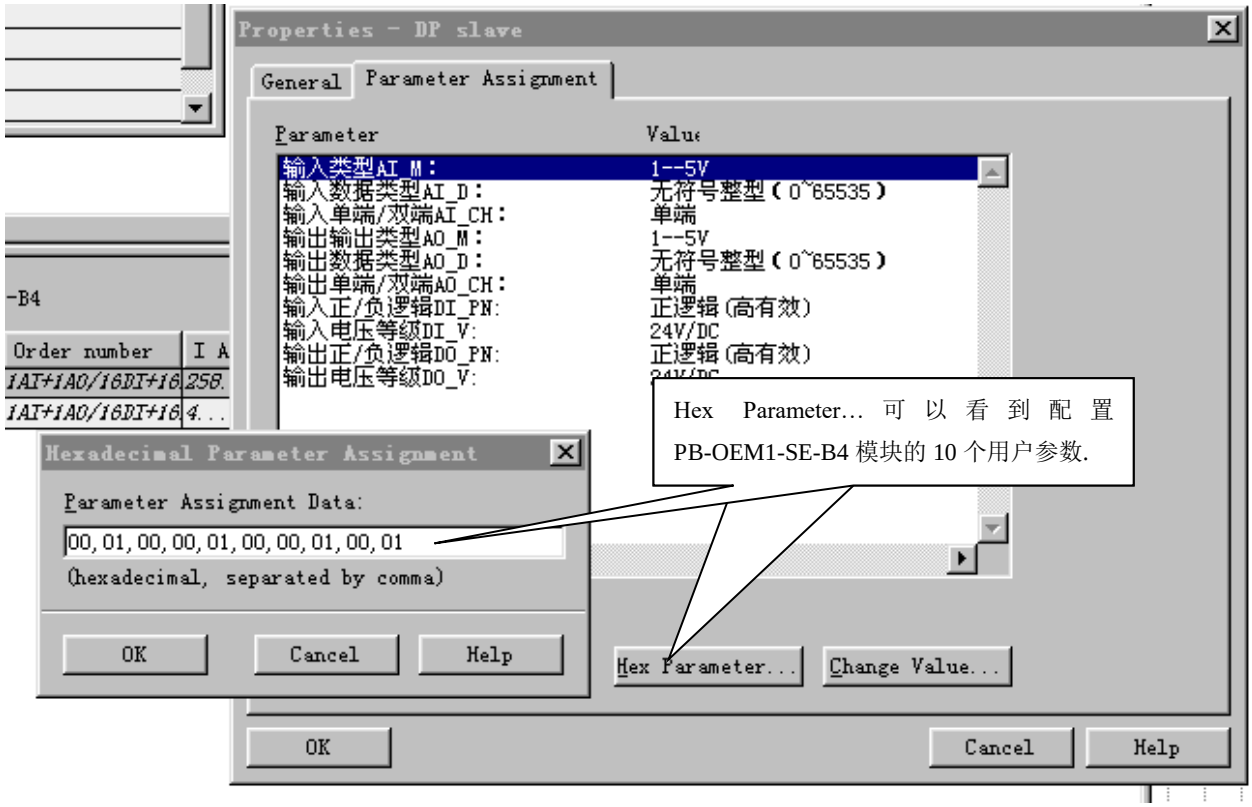


图 4-8: 可以看到配置 1AI+1AO 模块的 7 个用户参数

① 图 4-8: 可以看到配置 PB-OEM1-DPRAM-B4 模块的 10 个用户参数：00，00，01，00，00，01，00

② 第 1~6 参数：00，01，00，00，01，00，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

③ 第 7~10 参数：00，01，00，01 对应上面的配置：输入正/负逻辑 DI_PN：正逻辑（高有效）、输入电压等级 DI_V：24V/DC、输出正/负逻辑 DO_PN：正逻辑（高有效）、输入电压等级 DO_V：24V/DC。

⑦ 如果依照菜单改变配置，第 1~10 用户参数也会该变。

第五章：建立一个调试实验系统

OEM1 调试实验系统是一个最小化 PROFIBUS 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 OEM1 调试实验系统Ⅰ如何实现 PROFIBUS 系统配置及演示实验。

1．建立 OEM1 调试实验系统Ⅰ

⑴ 系统Ⅰ设备清单

从站				
序号	**名称**	**制造商**	**数量**	**备注**
1	PB-OEM1-DPRAM 嵌入式 PROFIBUS 接口	鼎实科技	1 块	
2	PB-OEM1-SAMPLE OEM1 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	PPROFIBUS 插头	西门子	2 个	
软件				
6	组态软件（光盘）： STEP 7 V5.2(DEMO 版)；A#光盘 SIMATIC NET V6.2(DEMO 版)；B#光盘 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-OEM1-DPRAM 产品手册》 《PROFIBUS-OEM1 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-OEM1-DPRAM 产品手册》 《PROFIBUS-OEM1 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	

⑵ 系统Ⅰ结构图

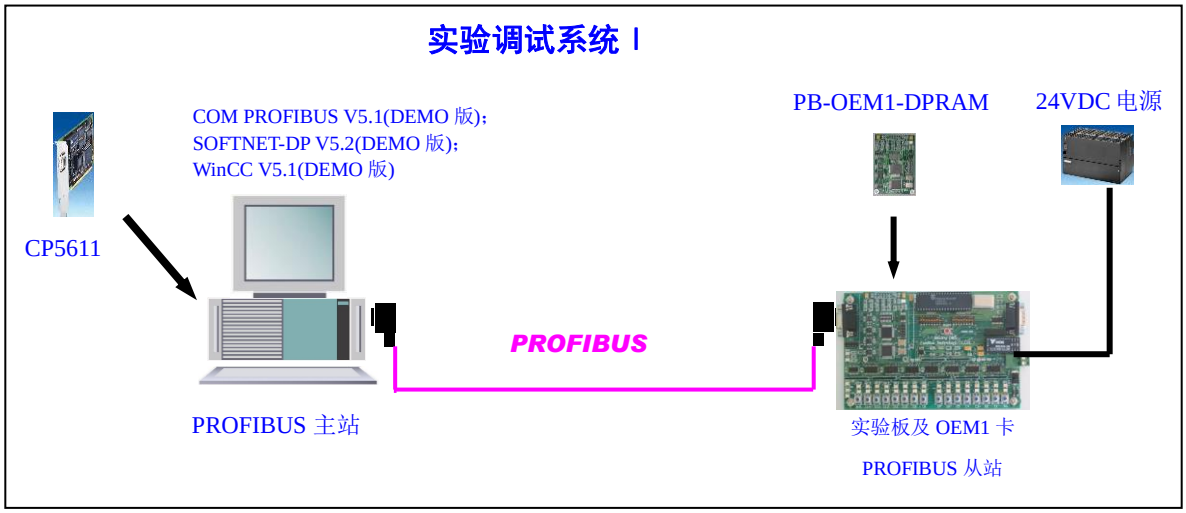


图 5-1：系统 0 结构图

⑶ 安装实验调试系统Ⅰ

㈠ 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，在本章中将开发实验板从站地址设置成：19（13H）号。

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。

㈡ 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 Win2000+SP4

⑧ 安装 SIMATIC NET V6.2(DEMO 版)

⑨ 安装 WINCC V5.1(DEMO 版)

⑩ 安装 STEP 7 V5.2(DEMO 版)

⑪ COPY PB-OEM1-DPRAM GSD 文件：

DS_DPRAM.GSD COPY 至 SIEMENS\Step7\S7DATA\GSD；

SE_B.BMP COPY 至 SIEMENS\ Step7\S7DATA\NSBMP。

⑫ 注：SE_B.BMP 是 PB-OEM1-DPRAM 的图标，不复制该图标到上述目录，不影响配置和通讯。

安装软件说明：*注意：▼当在安装 WinCC 软件时，序列号：可键入“0”；*

▼由于安装软件为 DEMO 版，因此安装中不选择“执行授权”。

2 使用 Step7 完成系统配置

⑴ 打开 Step7

⑵ 新建一个项目

见图 2.1：

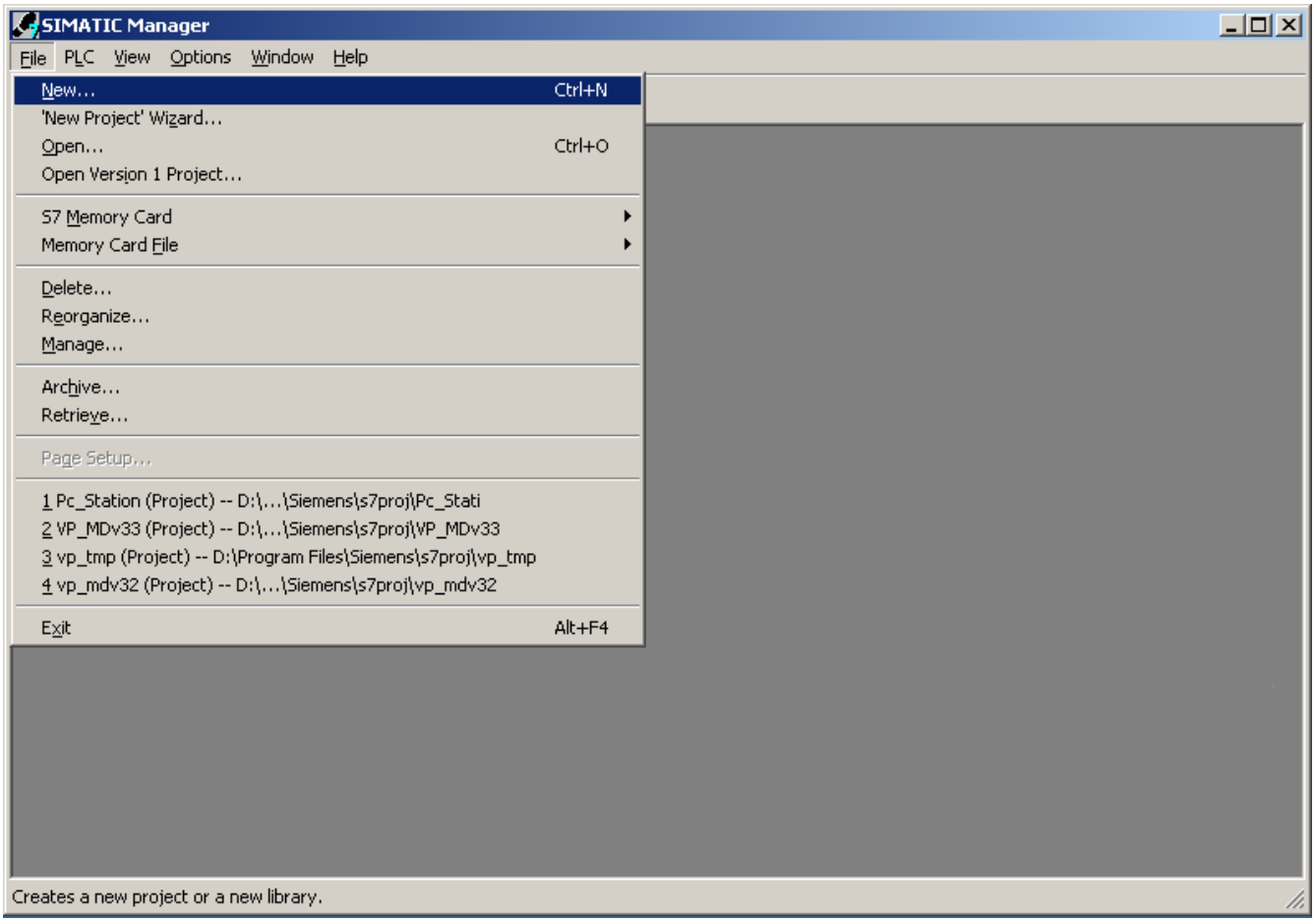


图 5.1

并将该项目命名为：DPRAM_PC_Station，如图 5.2

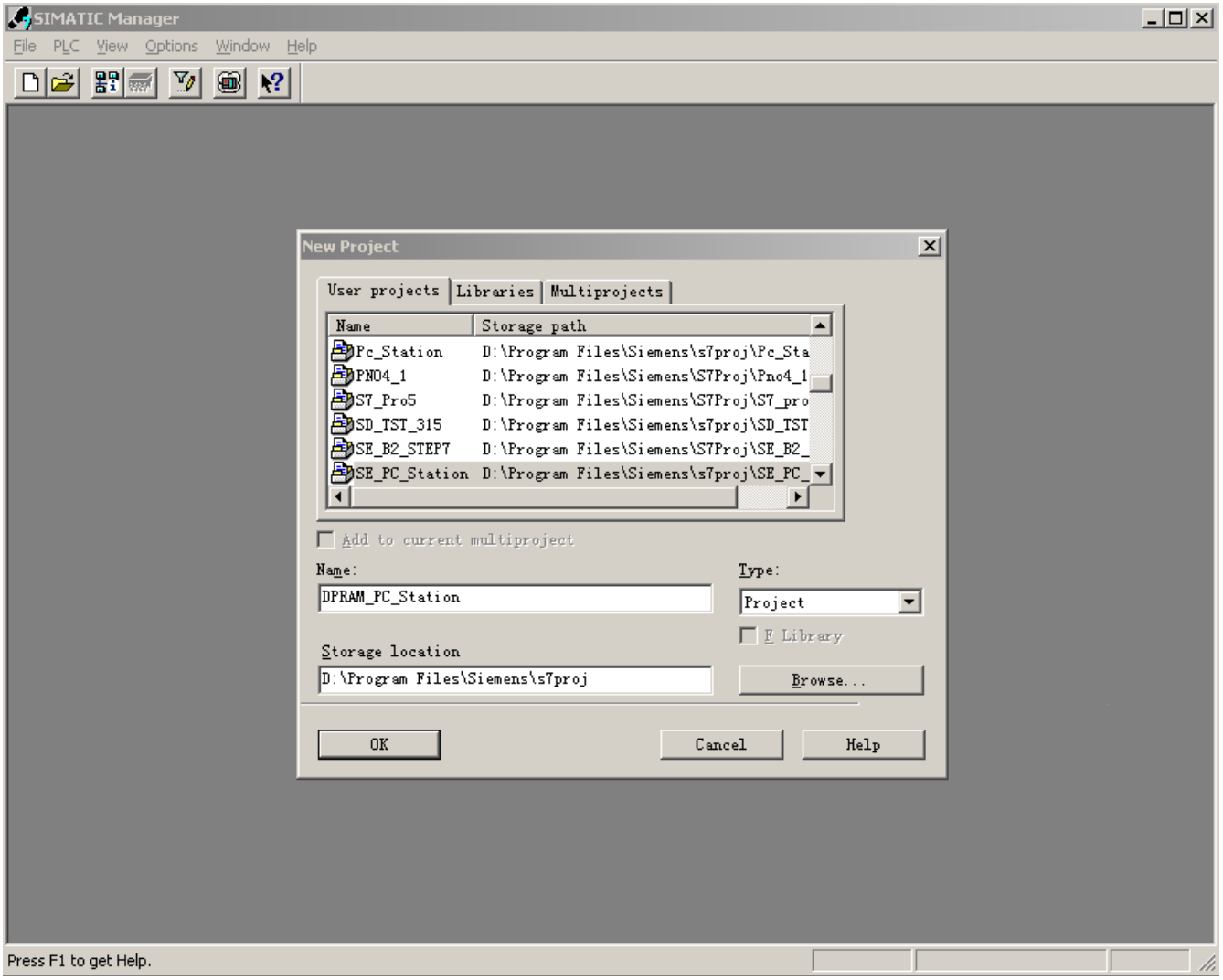


图 2.2

⑶ 添加 PC Station

在 DPRAM_PC_Station 鼠标右键弹出菜单，如图 5.3

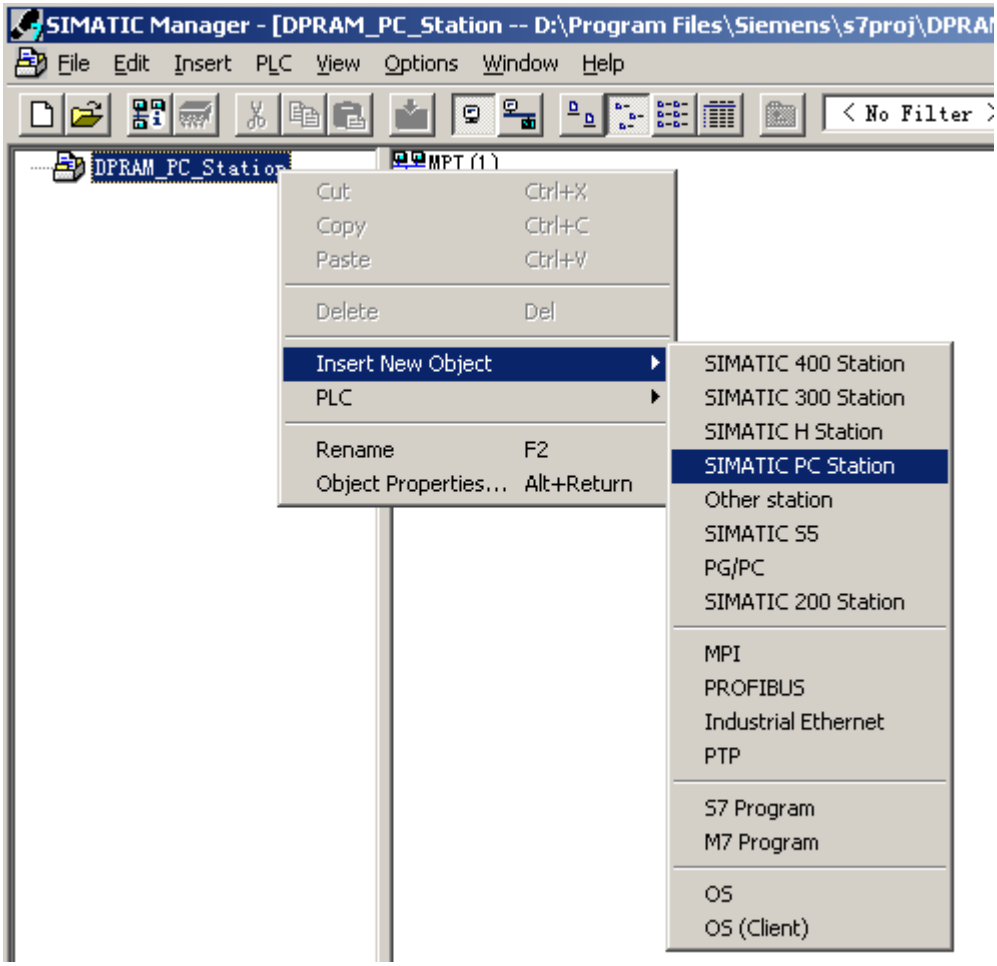


图 5.3

双击图 5.4 中的 SIMATIC PC Station(1)，SIMATIC PC Station(1)为所配置的 Station Name。

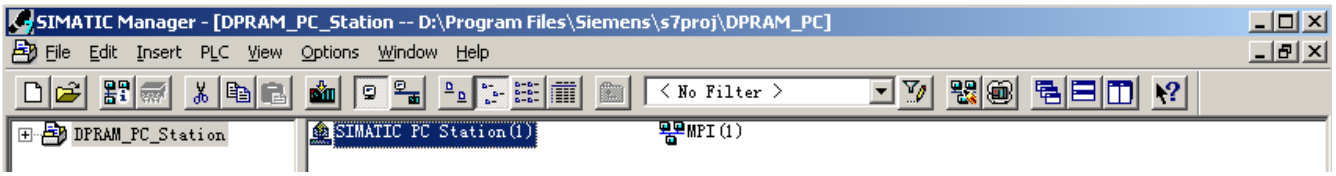


图 5.4

然后双击图 5.5 中的 Configuration，

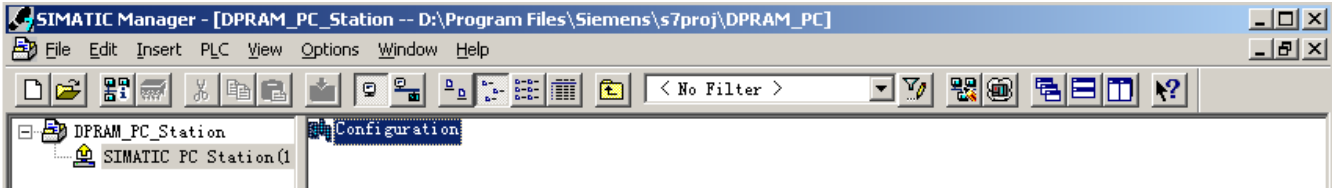


图 5.5

出现图 5.6 窗口，在改窗口上进行硬件配置。

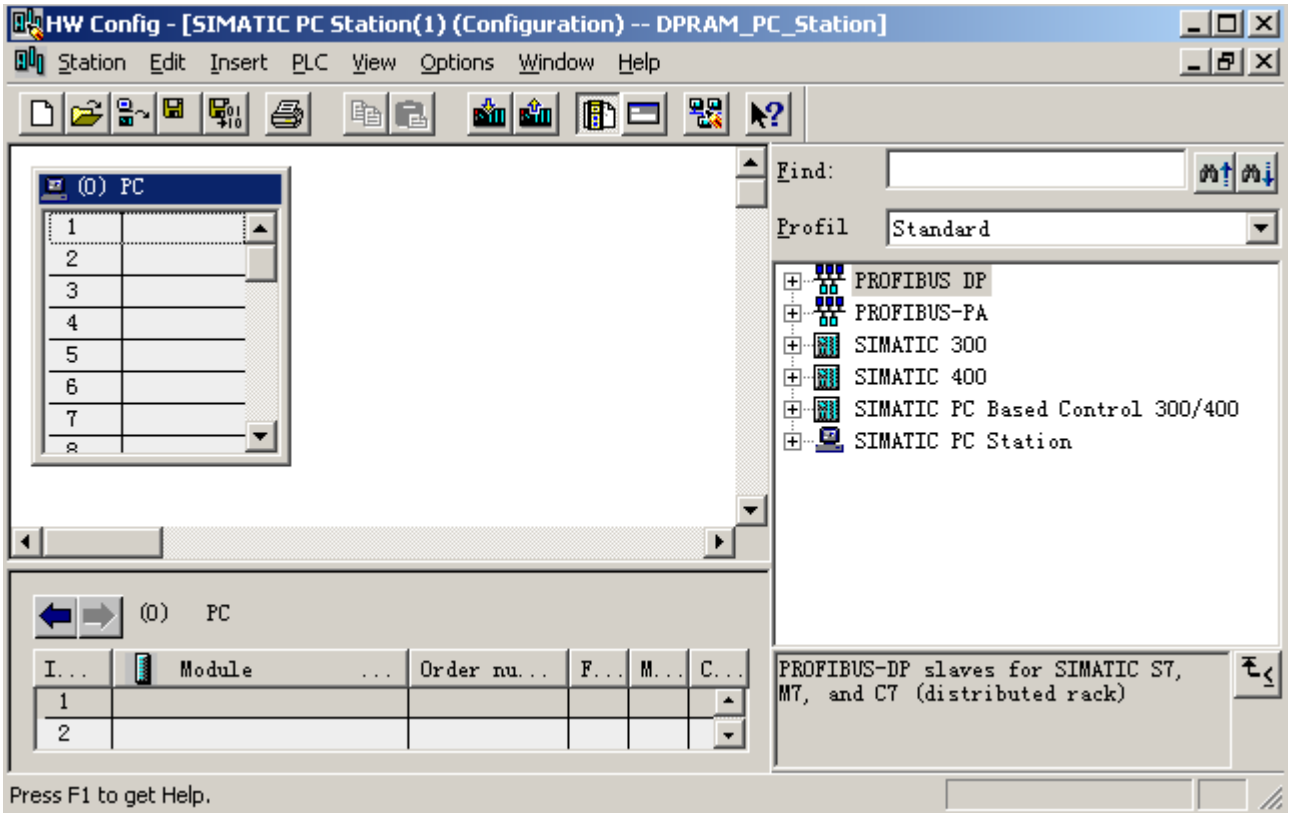


图 5.6

按照图 5.7 所示，在 rail 上点击鼠标右键，选择“Insert Object…”，选择 Application

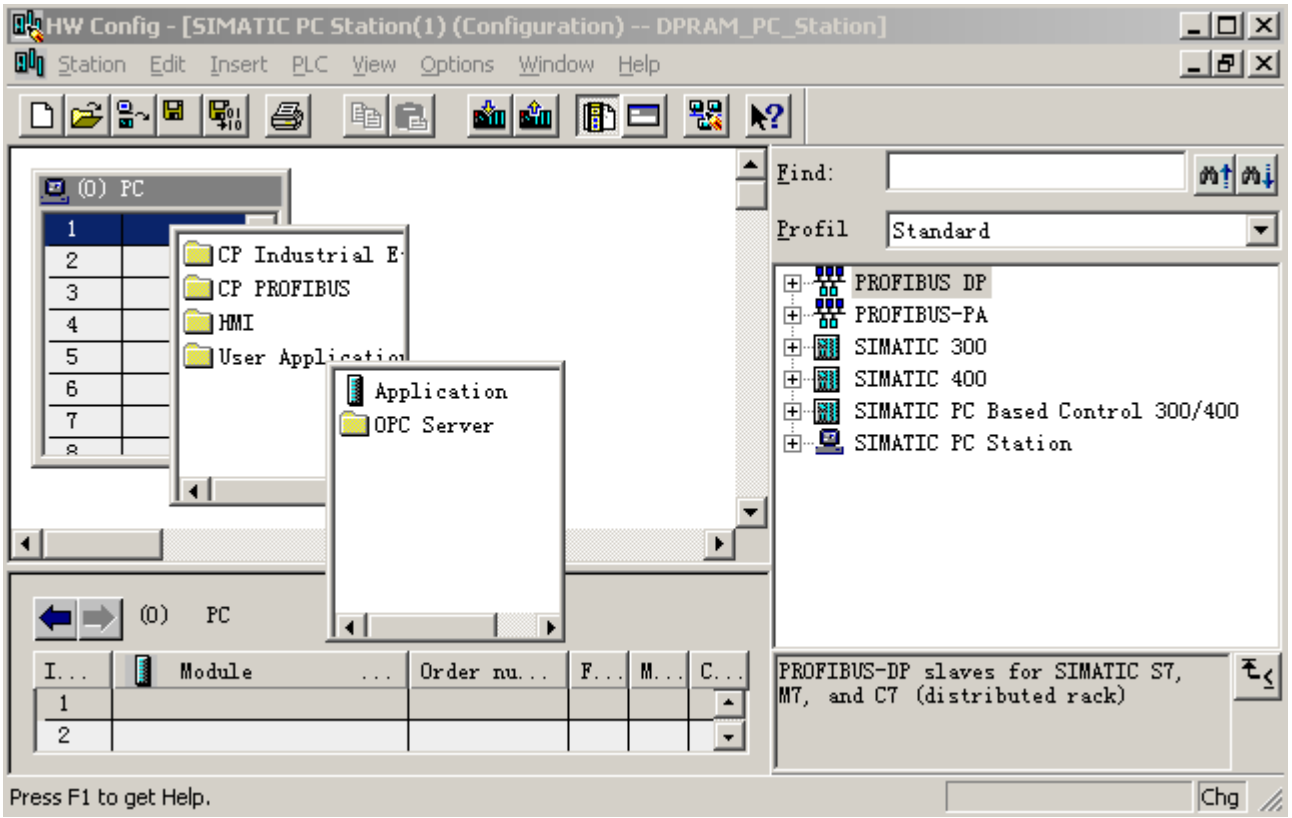


图 5.7

于是，Application 就被配置在 rail 的第一个槽中，其 index 为 1，见图 5.8：

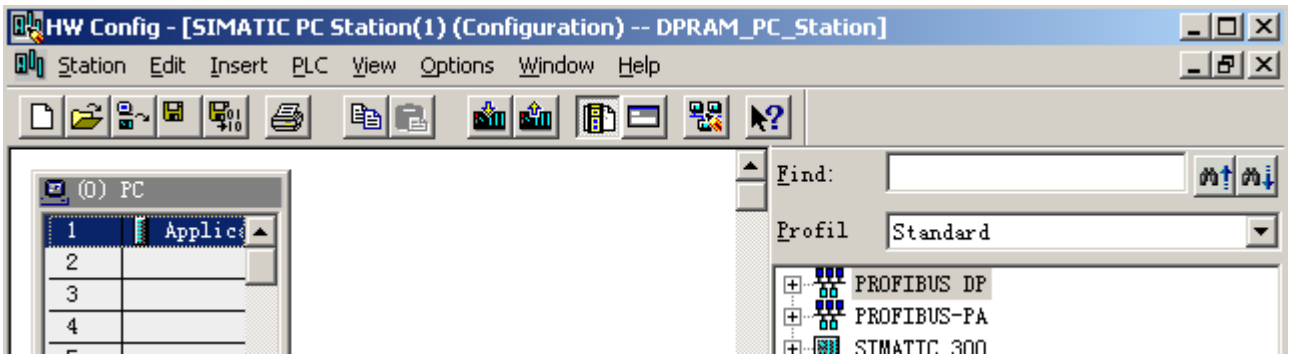


图 5.8

以同样的方法配置 rack 的第二个槽，如图 5.9，选择“CP PROFIBUS→CP 5611→SW V6.0 SP5”，

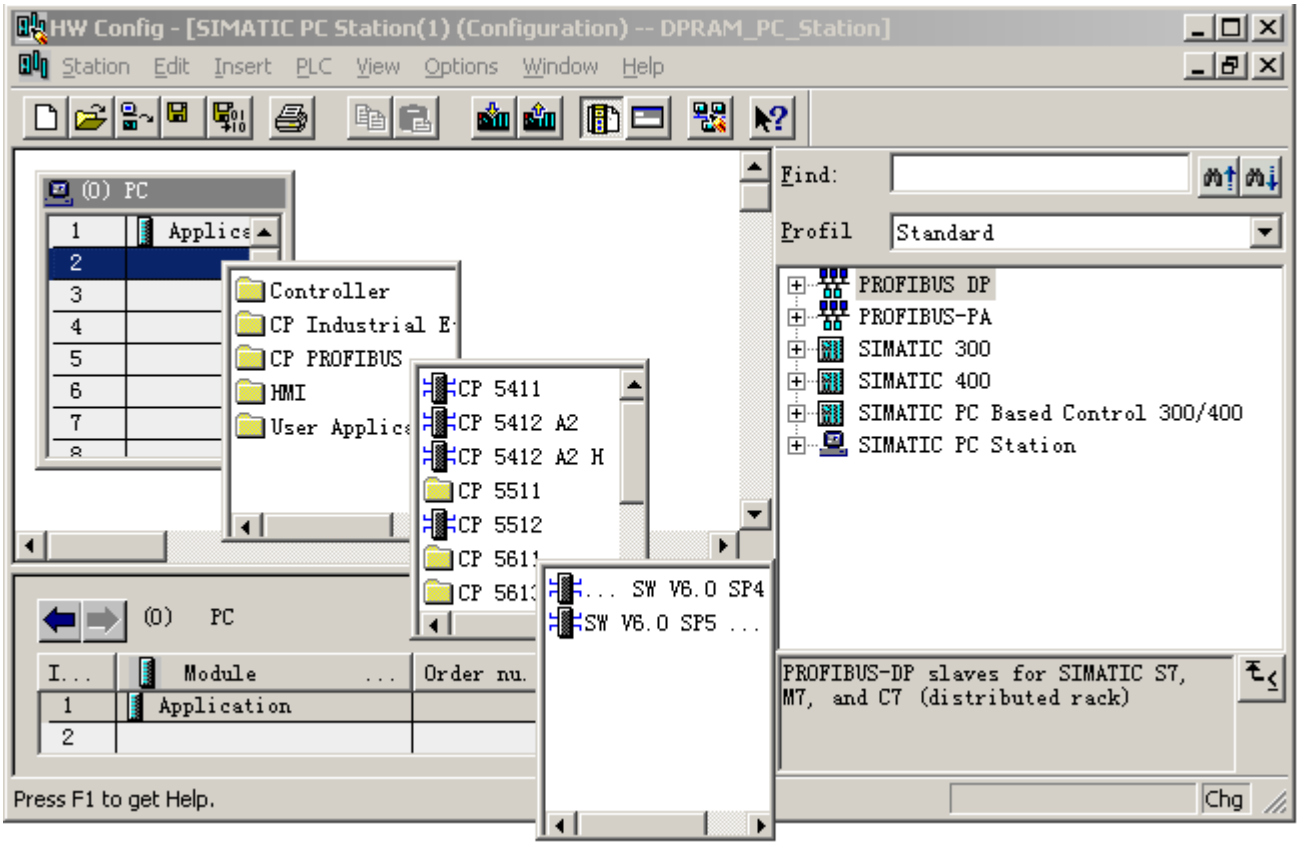


图 5.9

然后弹出图 5.10 所示的窗口，Address 为主站地址，一般默认为 2，

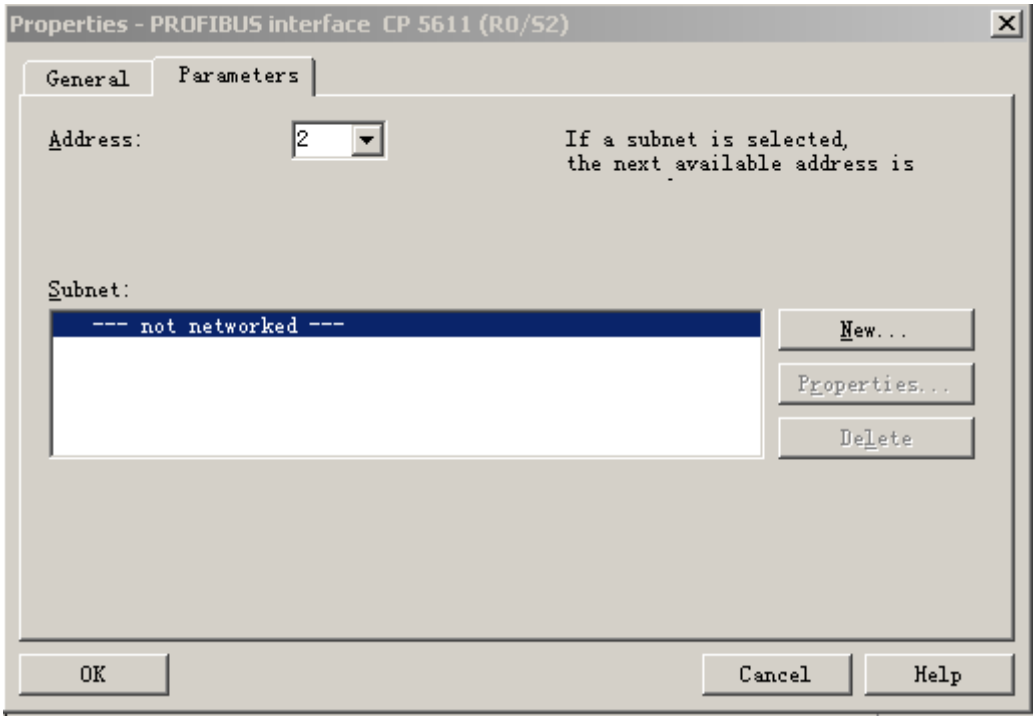


图 5.10

点击图 5.10 中窗口的“NEW”按钮新建一个 PROFIBUS 网络，出现图 5.11 所示的窗口，选择“Network Settings”，并将波特率设置为 187.5Kbps，因为这个波特率比较常用。然后，点击“OK”按钮，退出当前窗口。

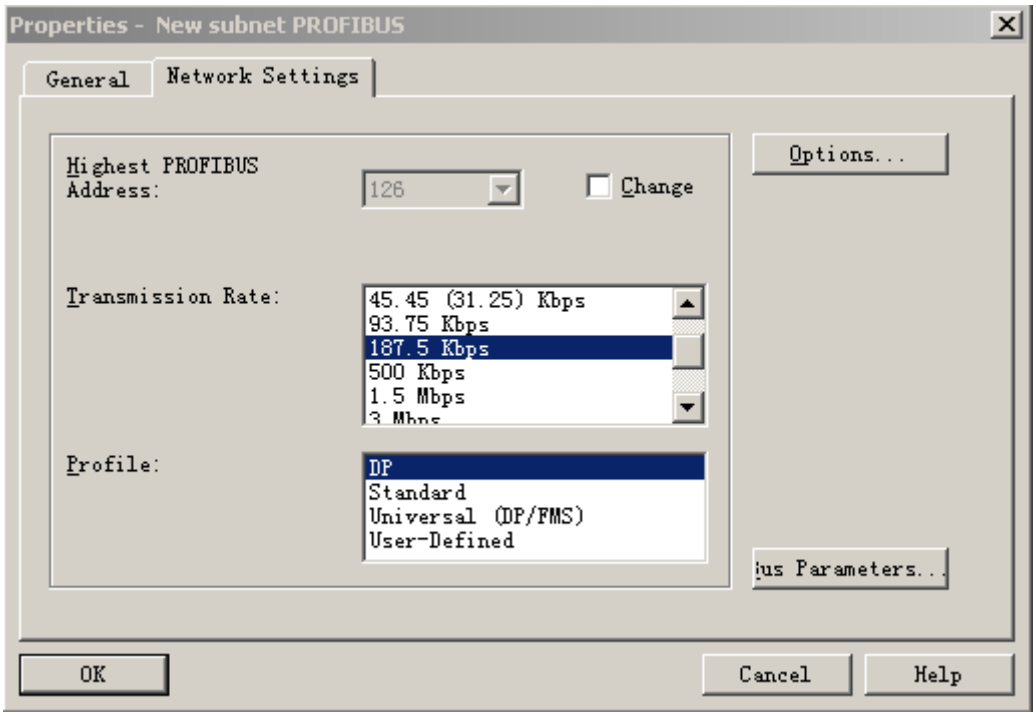


图 5.11

然后图 5.10 所示的窗口变为图 5.12 所示的窗口，

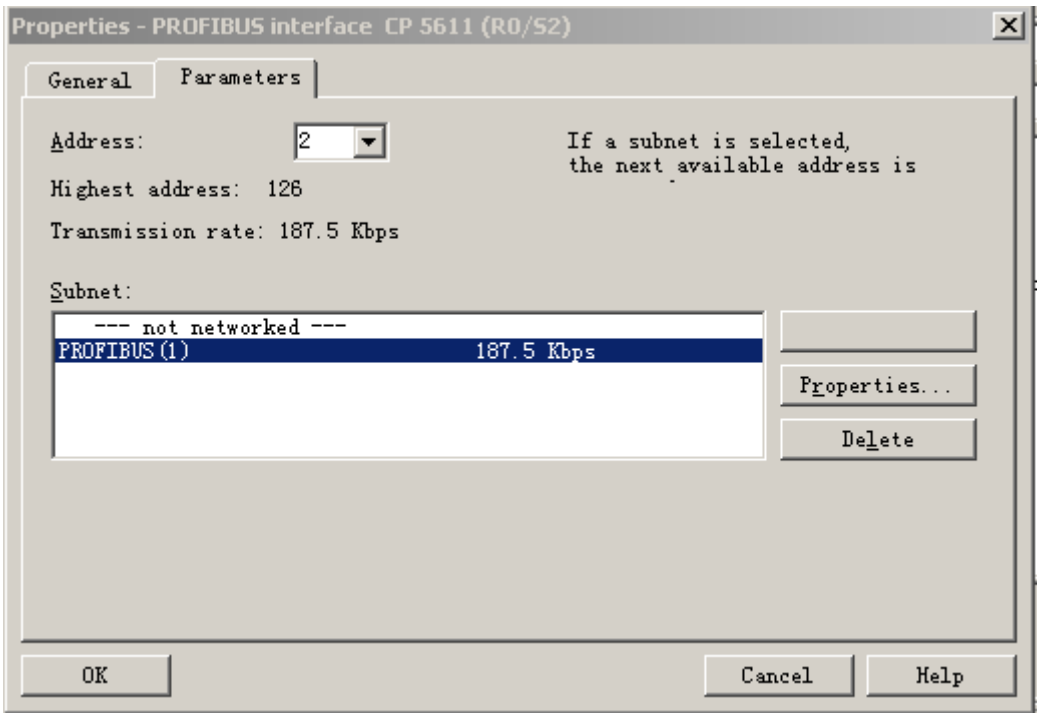


图 5.12

点击图 5.12 所示窗口中的“OK”按钮，退出当前窗口，如图 5.13 所示。

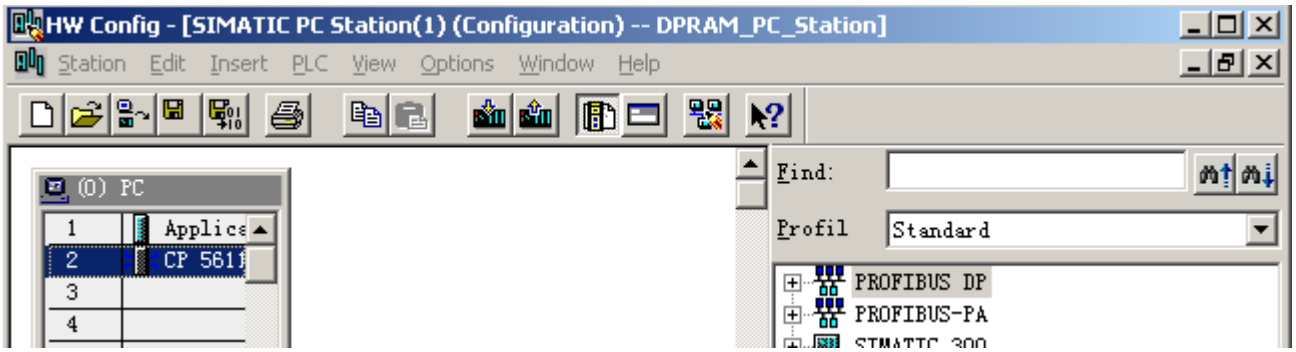


图 5.13

选择图 5.13 中 CP 5611，右击鼠标右键，选择“Add Master System”如图 5.14

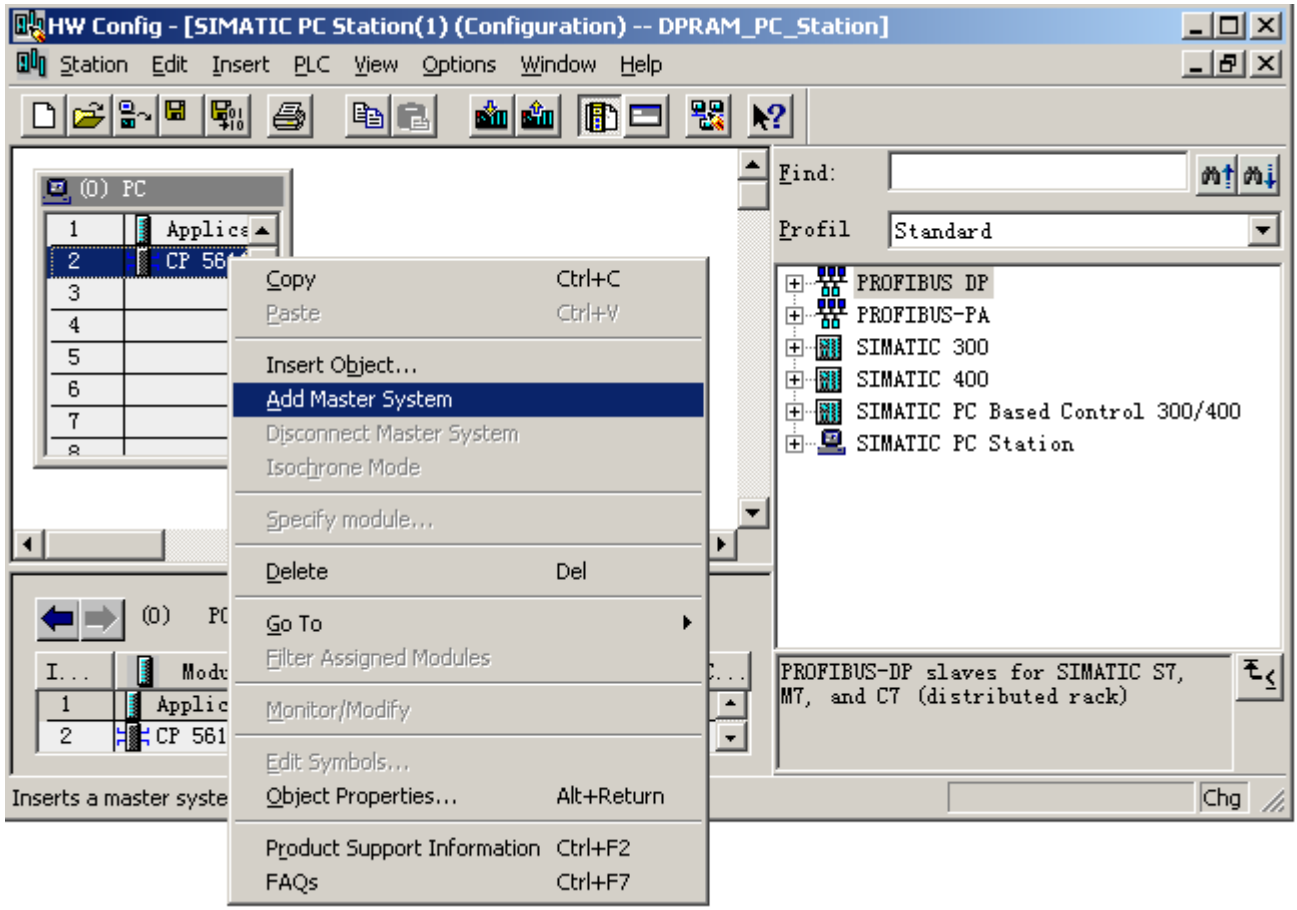


图 5.15

然后出现图 5.16 所示窗口，选择“Application”，然后点击“OK”，退出当前窗口。

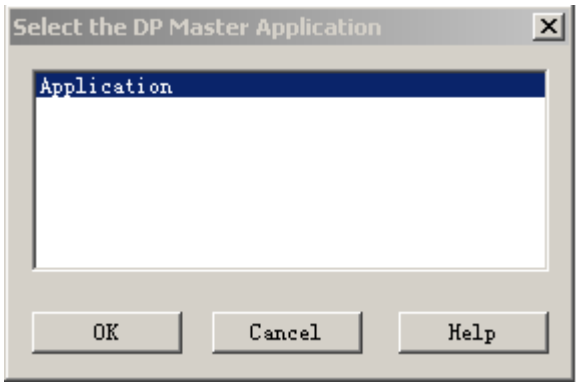


图 5.16

则出现图 5.17 所示的窗口，

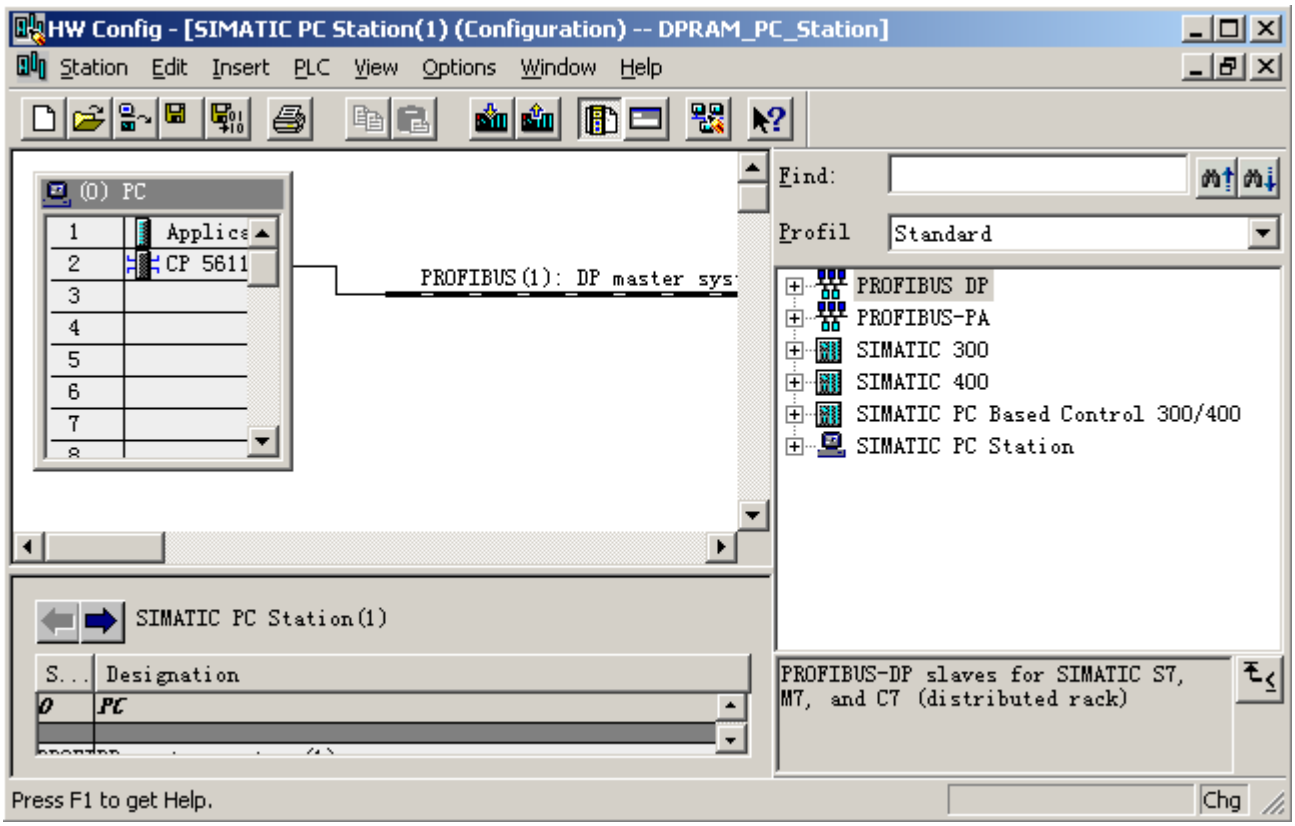


图 5.17

⑷ 添加 PB-OEM1-DPRAM

通过对比图 5.13 和图 5.17，发现从 CP 5611 扩展出了一个 PROFIBUS 网络，用鼠标点中该网络，然后从窗口右边的 Catalog 区域选择 PB-OEM1-DPRAM，双击该产品型号，如图 5.18，

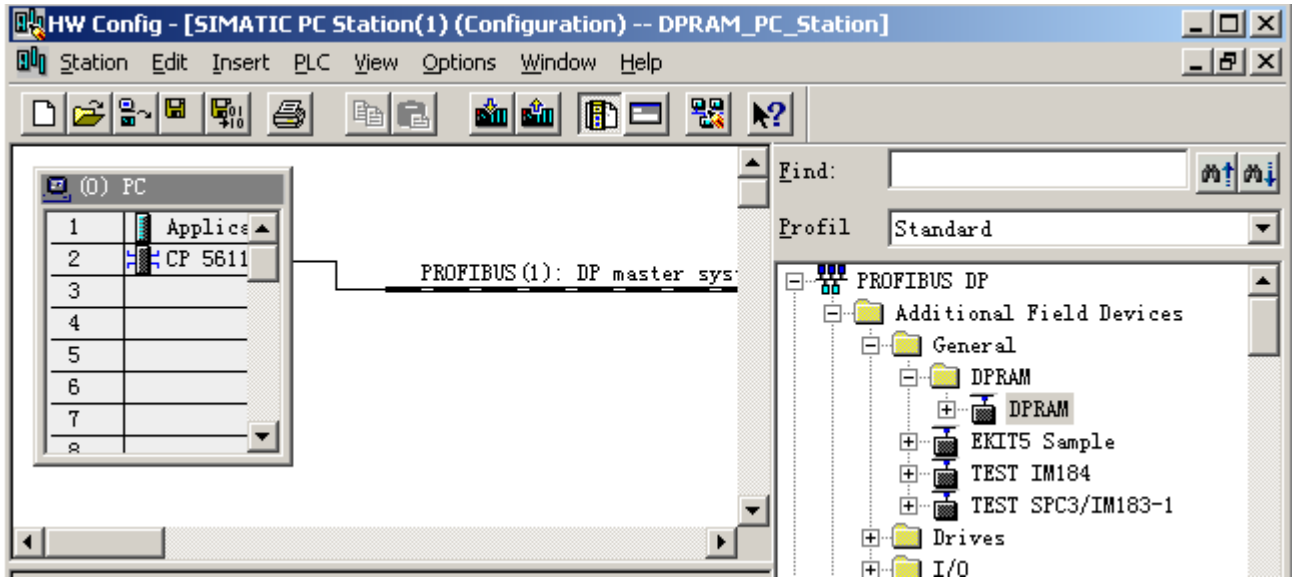


图 5.18

注：如果在 Catalog 中没有找到 DPRAM，首先确认是否将 DPRAM.GSD 复制到 SIEMENS\Step7\S7DATA\GSD 目录下，然后用户是否更新。如果没有更新 Catalog，则需要保存当前配置，然后关闭当前配置的子窗口，保留主窗口。选择主窗口“Options→Updata Catalog”，如图 5.18-1

所示。

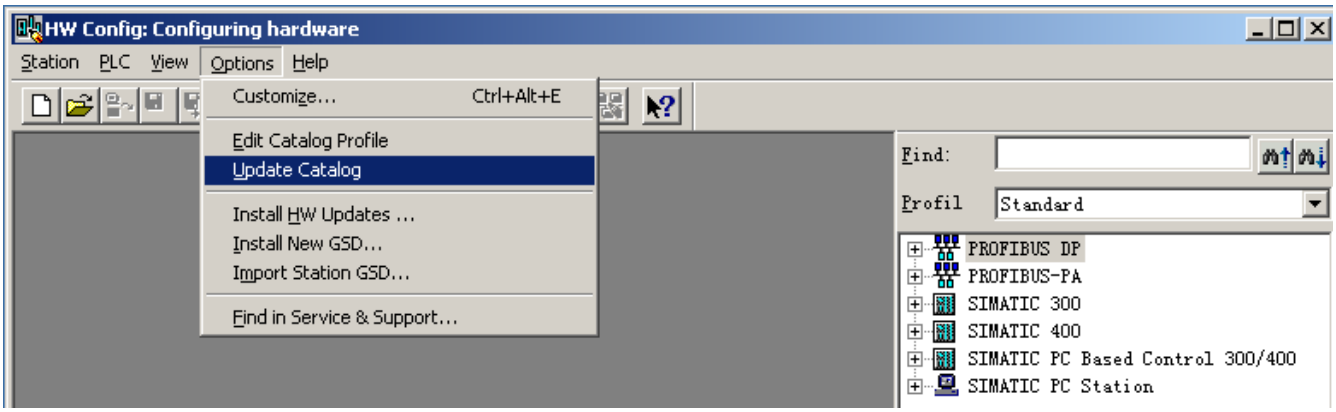


图 2.18-1

然后弹出如图 5.19 的窗口，并将从站地址设置为 19（十进制）；该地址必须与实际设备的地址一致。用户必须将 PB-OEM1-DPRAM 调试实验板的地址设置 13（十六进制），拨码开关的从左至右应为：00010011(二进制)。选择“OK”退出当前窗口。

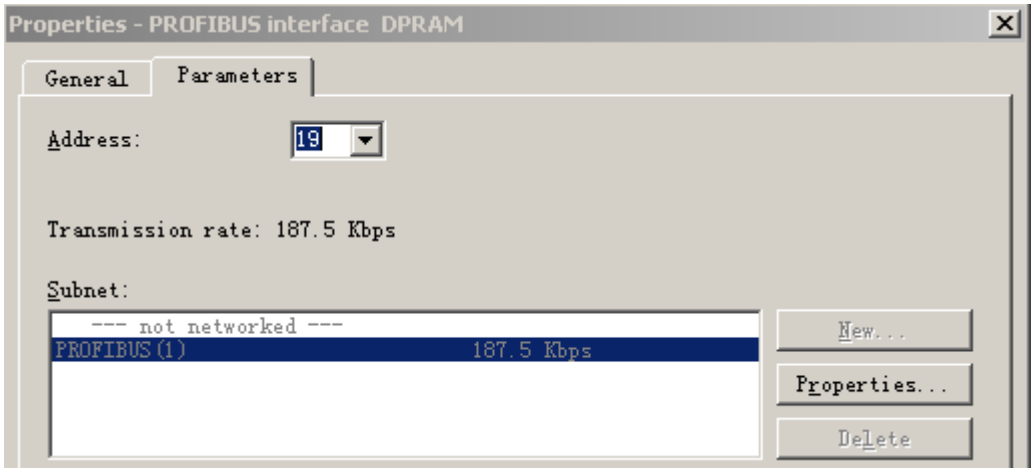


图 5.19

则从站配置完成，如图 5.20 所示，

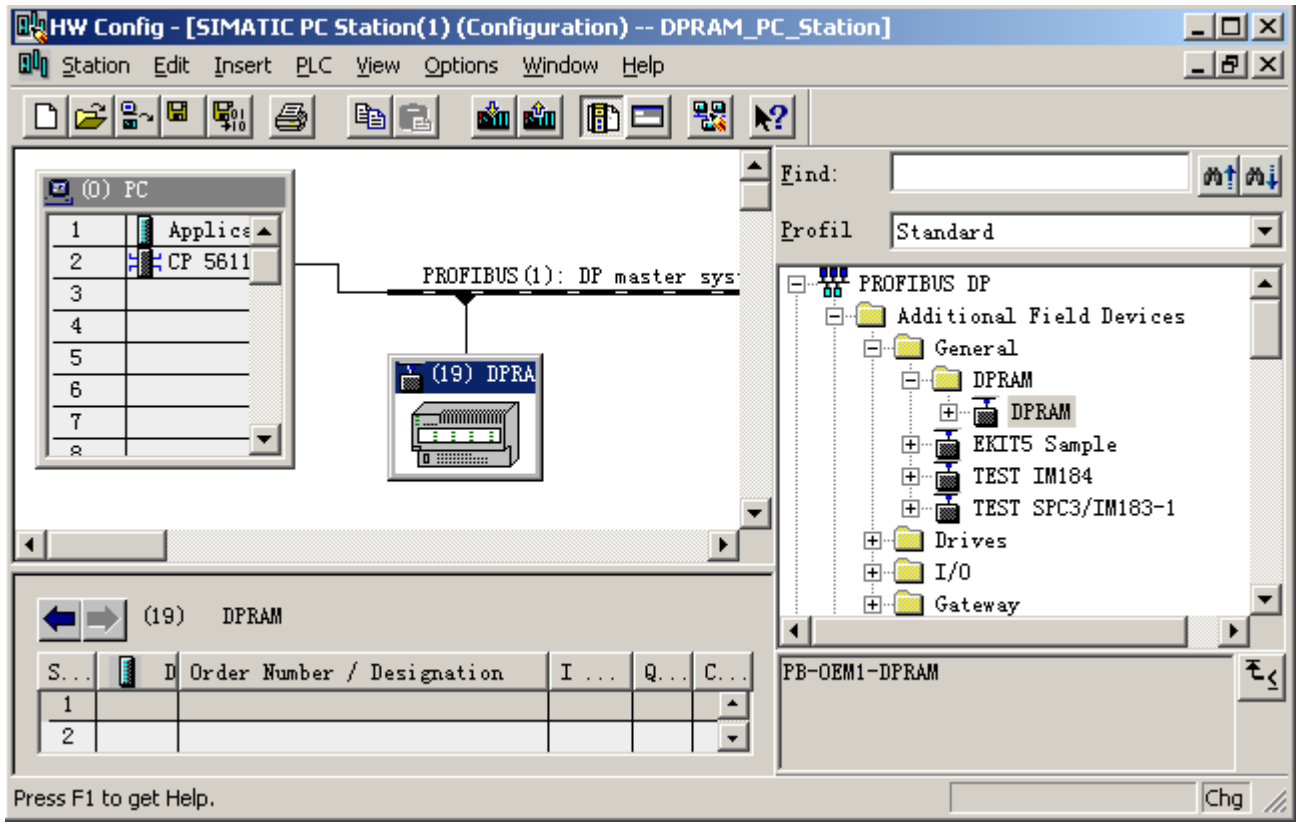


图 5.20

在如图 5.20-1 所示的插槽中，配置 112 Byte In，112 Byte Out，

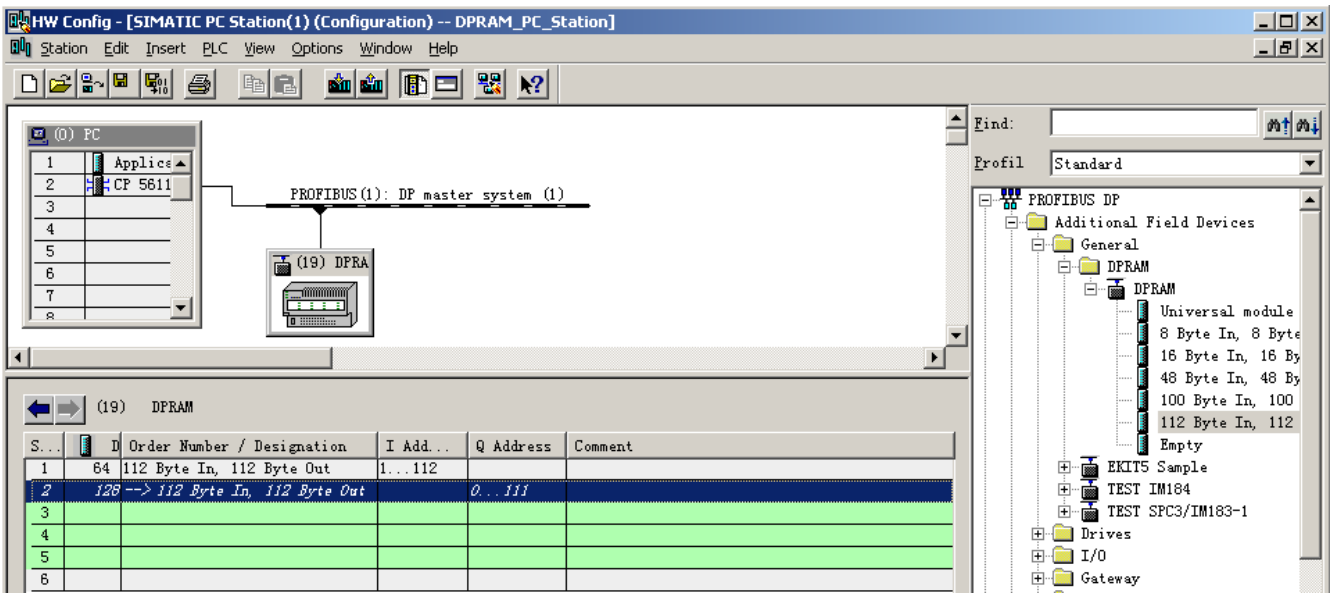


图 5.20-1

点击图 2.20 窗口中工具栏  图标，存盘编译，退出，则完成了 Step7 配置。

3 Set PG-PC Interface 的设置

进入“控制面板”，打开 Set PG-PC 的图标 ，将 S7ONLINE 指向 PC internal，CPL2_1 指向 CP5611(PROFIBUS)，如图 5.20-1，5.20-2 所示，完成后“OK”退出。

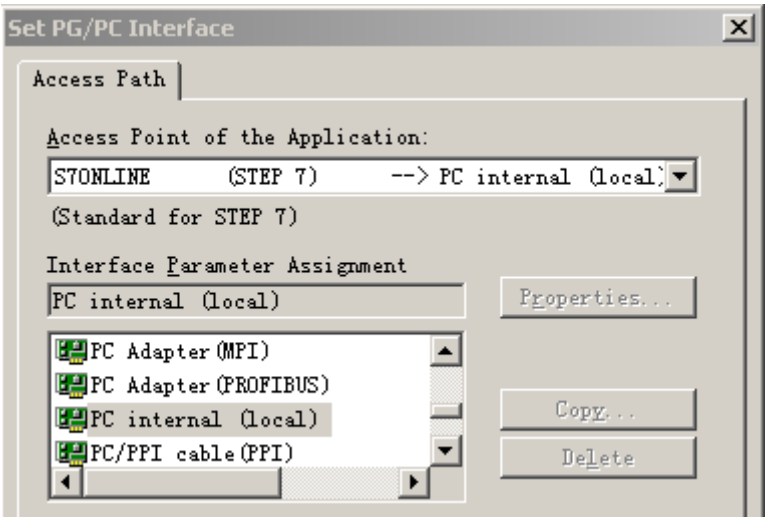


图 5.20-1

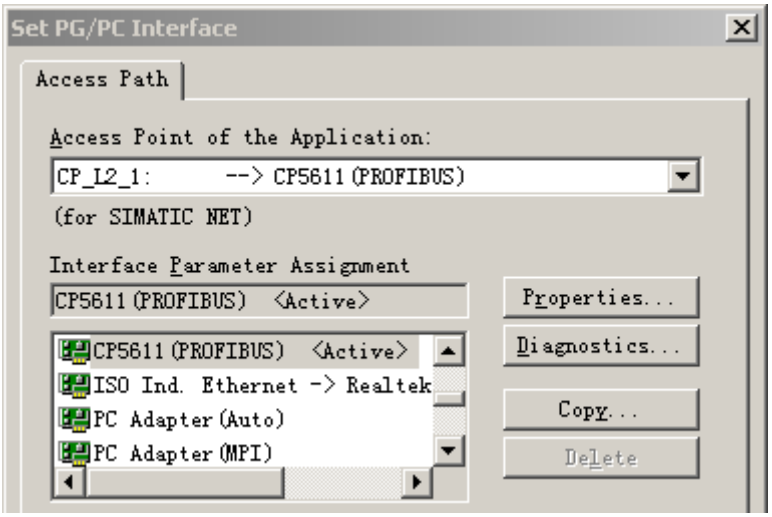


图 5.20-4

4 Simatic Net 的设置

按照图 5.20-3 所示选择 Simatic Net 配置程序，

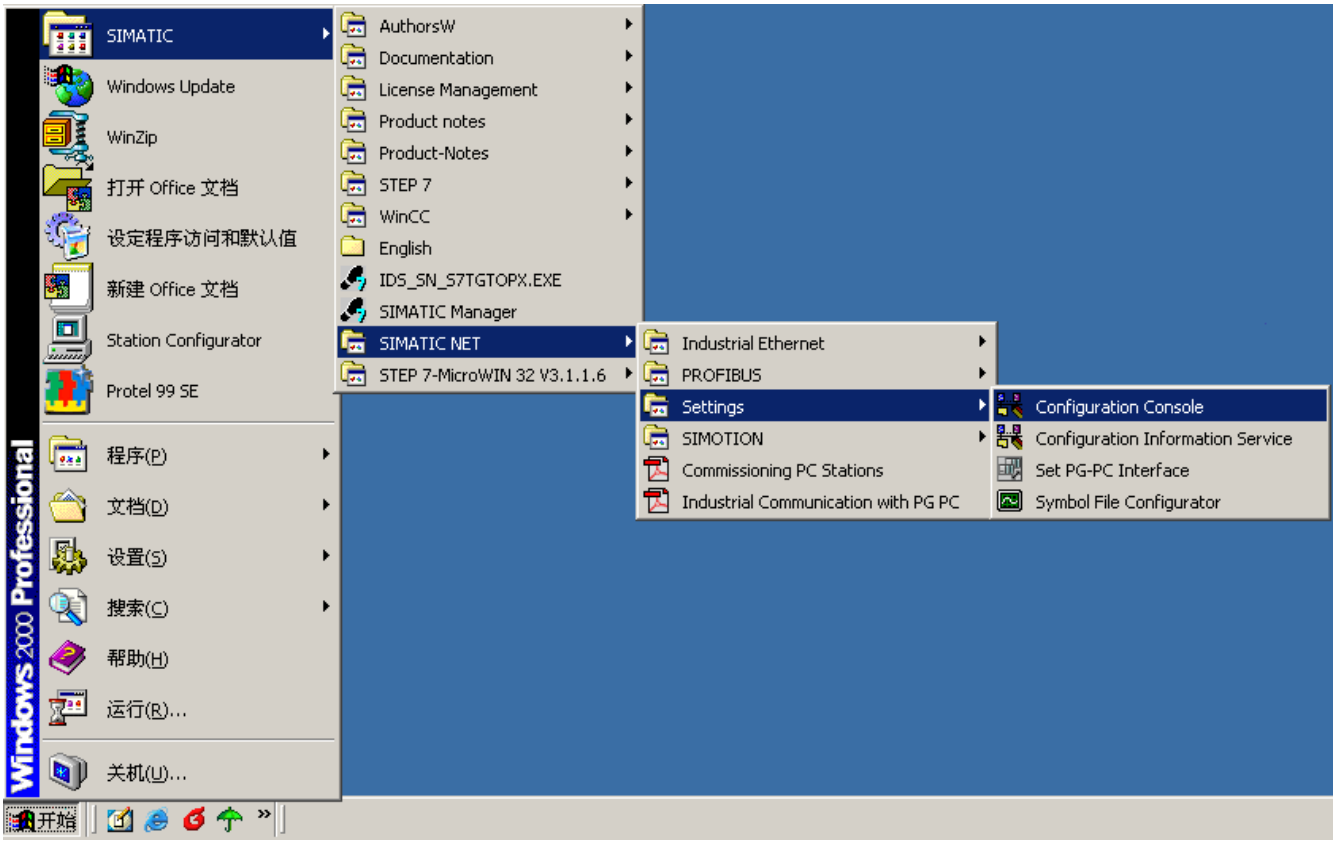


图 5.20-3

进入网卡配置界面，如图 2.20-4 所示，CP 5611 的模式为“Configured mode”,Index 与 Step7 配置一致，设置为 2。点击“Apply”后，退出配置界面。

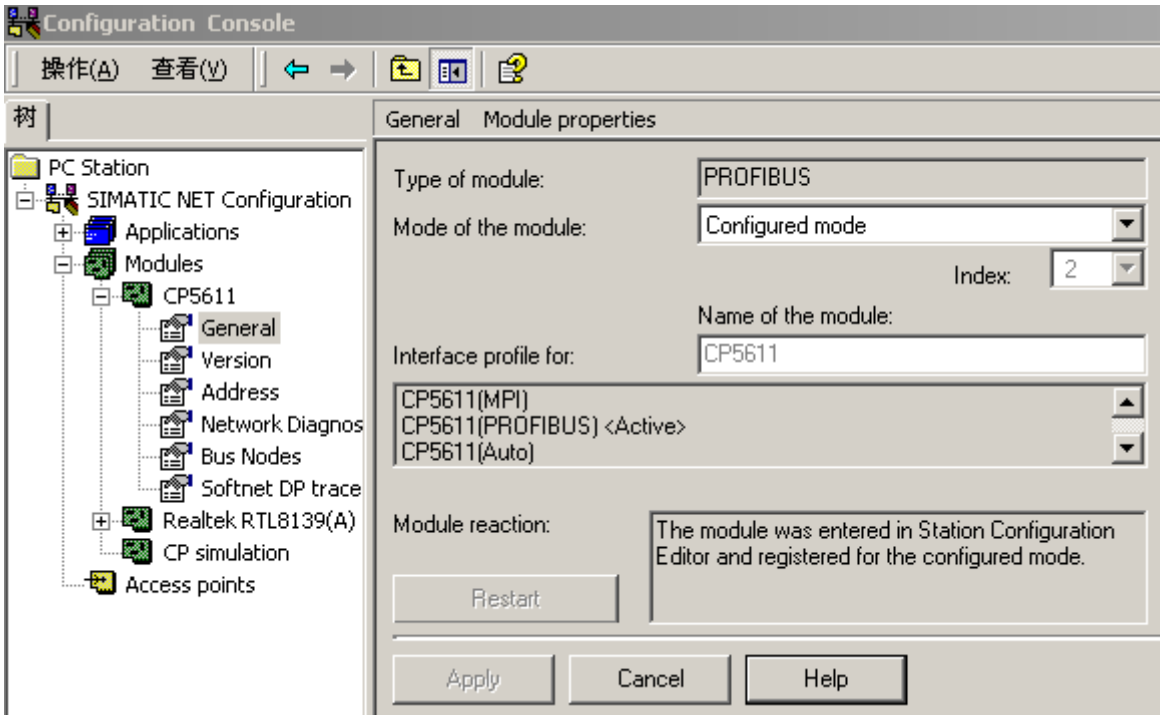


图 5.20-4

5 配置虚拟的 PC Station

⑴运行 Station Configuration

点击桌面上的 Station Configuration 快捷方式，进入虚拟的 PC Station 画面，如图 5.21 所示。

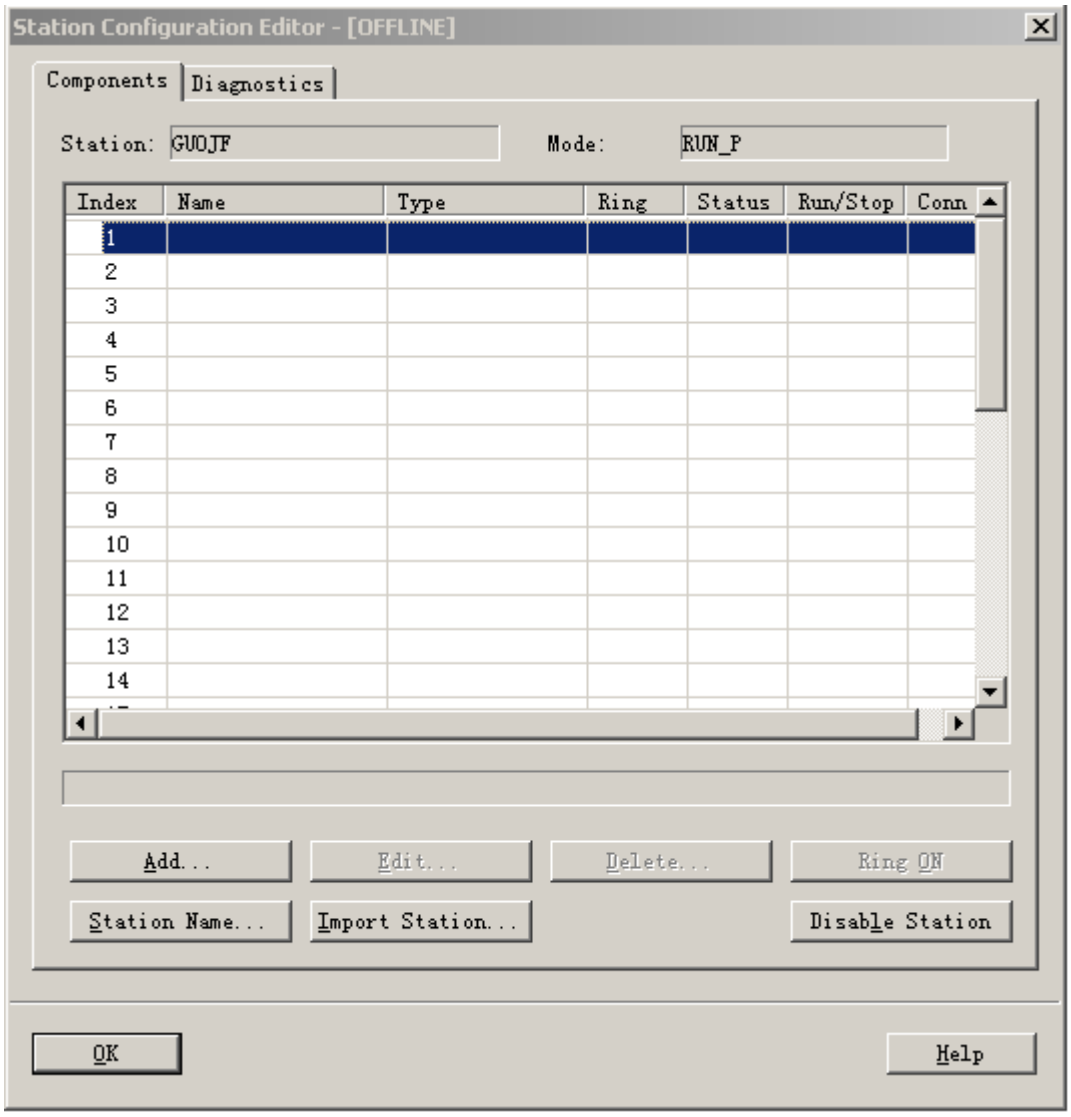


图 5.21

选择图 5.21 所示的“Import Station”按钮，将 Step7 的配置 import 到虚拟的 PC Station 中来。此时会弹出一个对话框，如图 5.22，确认后点击“Yes”按钮。

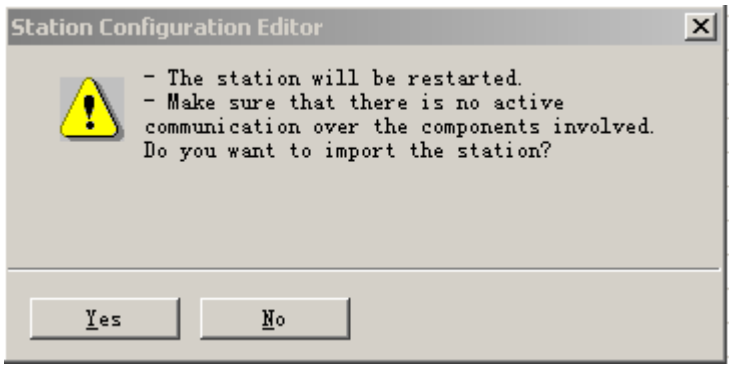


图 5.22

弹出一个对话框，选择 Step7 的配置。Step7 的配置存放在 SIEMENS\S7proj\SE_PC_ST\XDBs 下，打开该目录下扩展名为 xdb 的文件，本例中为 pcst_1.xdb，该文件即为 Step7 的配置文件，如图 5.23 所示。

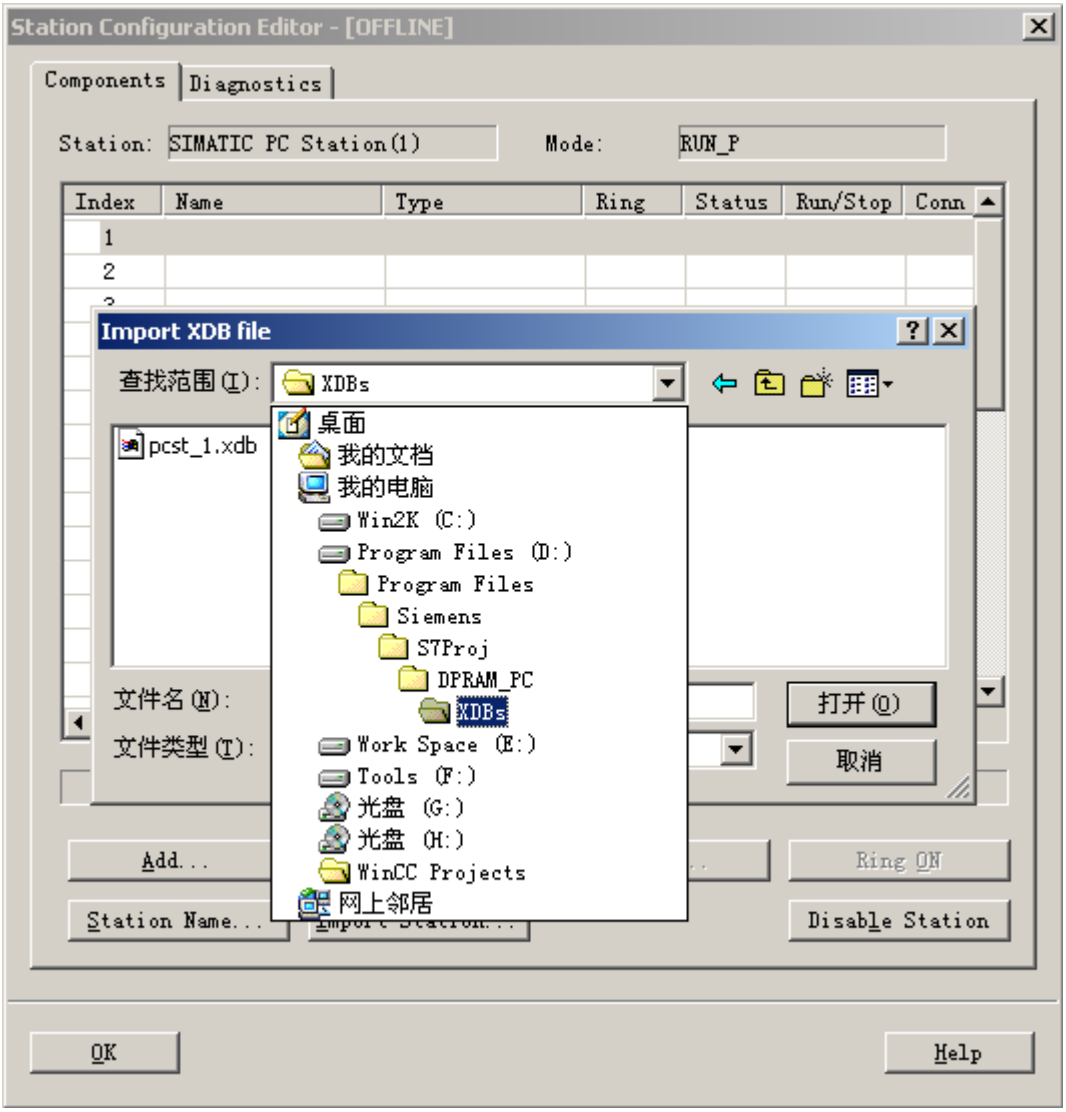


图 5.23

然后出现图 5.24 窗口，“OK“确认。

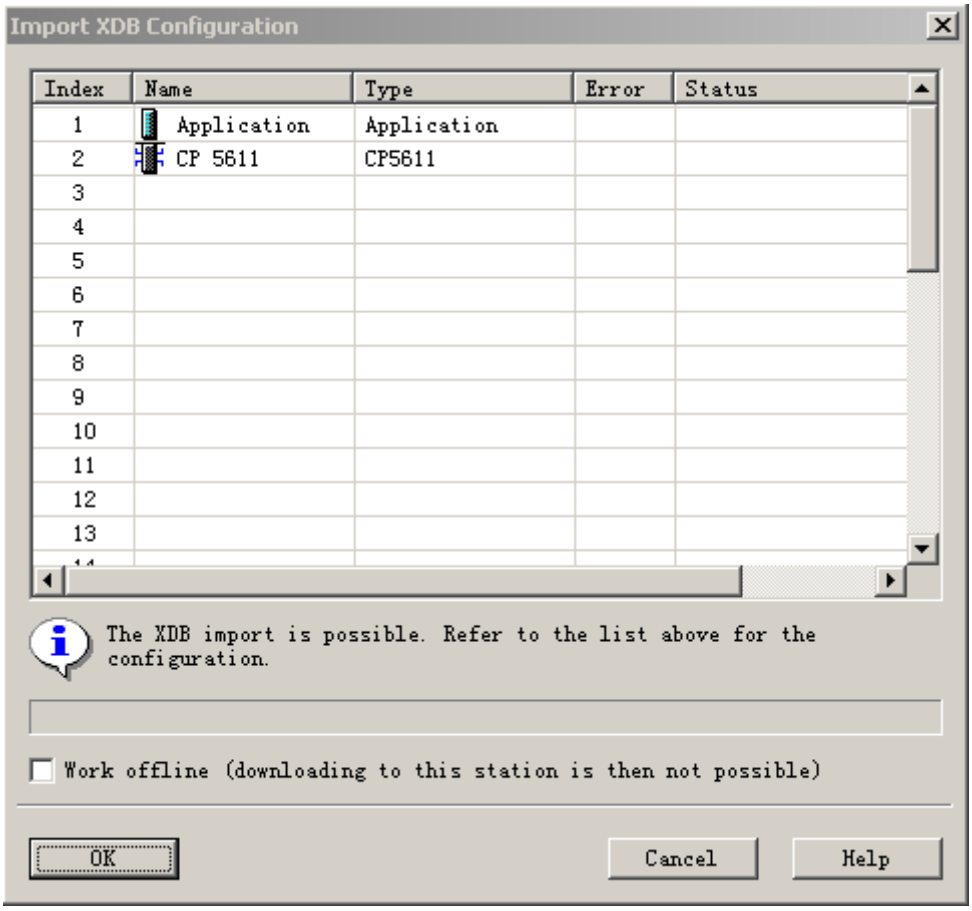


图 5.24

完成后，出现图 5.25 画面，表示配置成功。

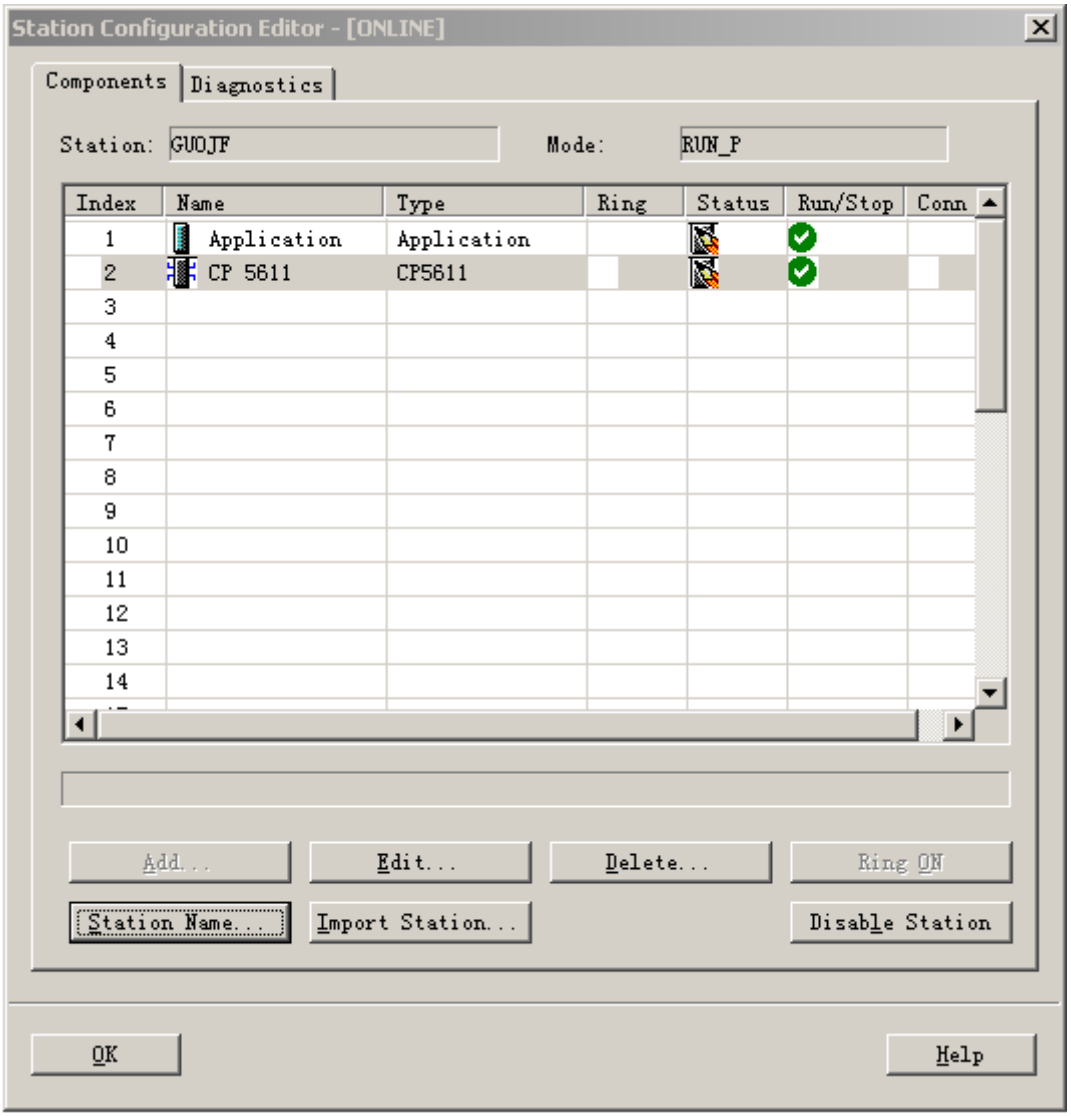


图 5.25

点击图 5.25 窗口中“Station Name”，为虚拟的 PC Station 设置站名。该名称必须与 Step7 配置的 Station Name 一致，见图 5.4 所示的 Station Name 为 SIMATIC PC Station(1)，如图 5.26 所示，“OK”退出。

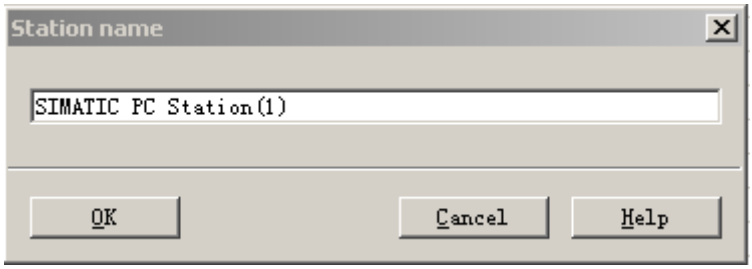


图 5.26

完成后，“OK”退出 PC Station 的配置窗口。

⑵下载硬件配置信息到虚拟 PC Station 中

打开 Step7 的配置界面，将配置信息下载到 PC Station 中，如图 5.27 所示，点击工具栏红色标注按钮

进行下载。

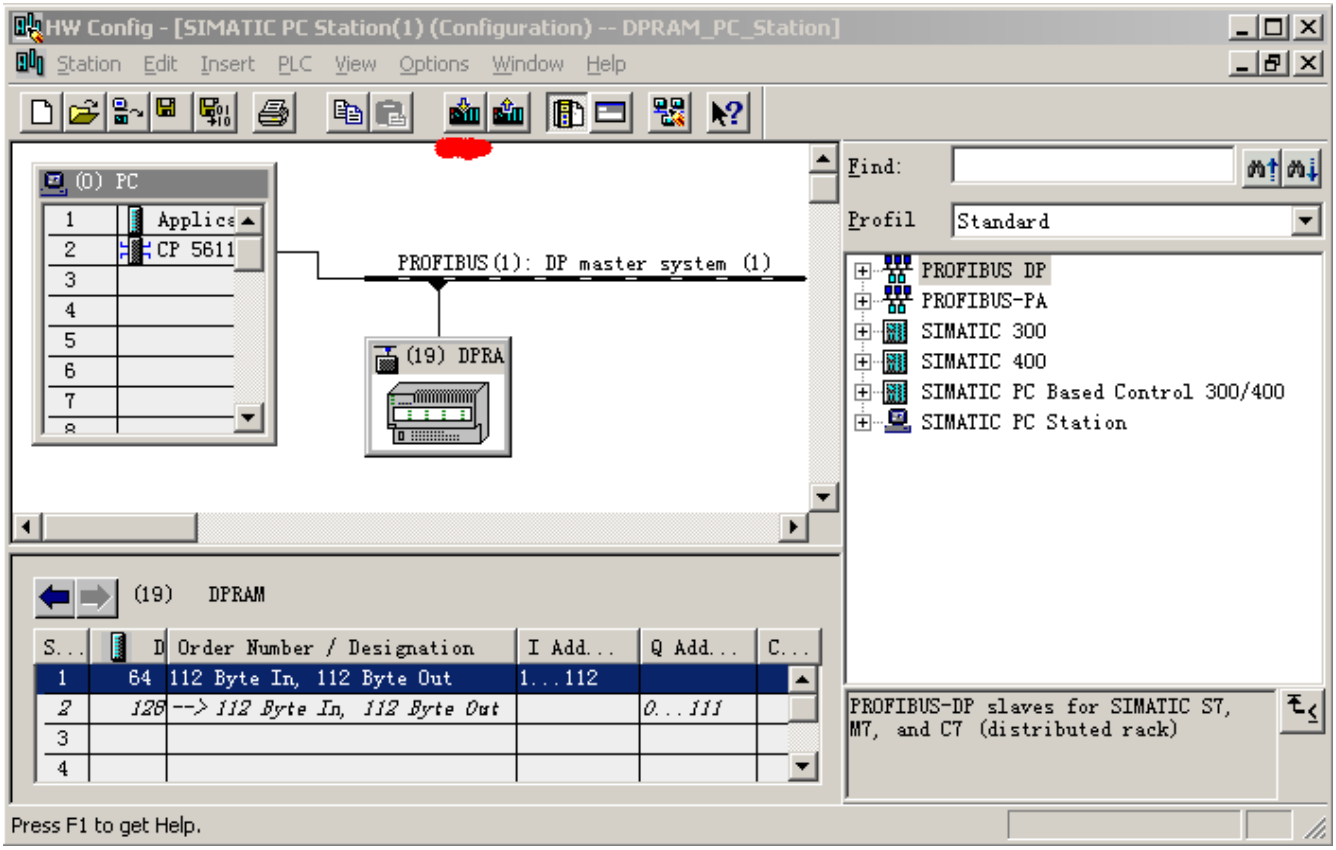


图 5.27

检查虚拟 PC Station，如图 5.25 表示状态正常。

于是整个硬件配置和 PC Station 配置完成。

6 运行 WinCC PB-OEM1-DPRAM 演示系统

⑴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”。

⑵ 进入 WinCC、打开项目文件夹 TEST_OEM2_B12，改计算机名。见图 5.28

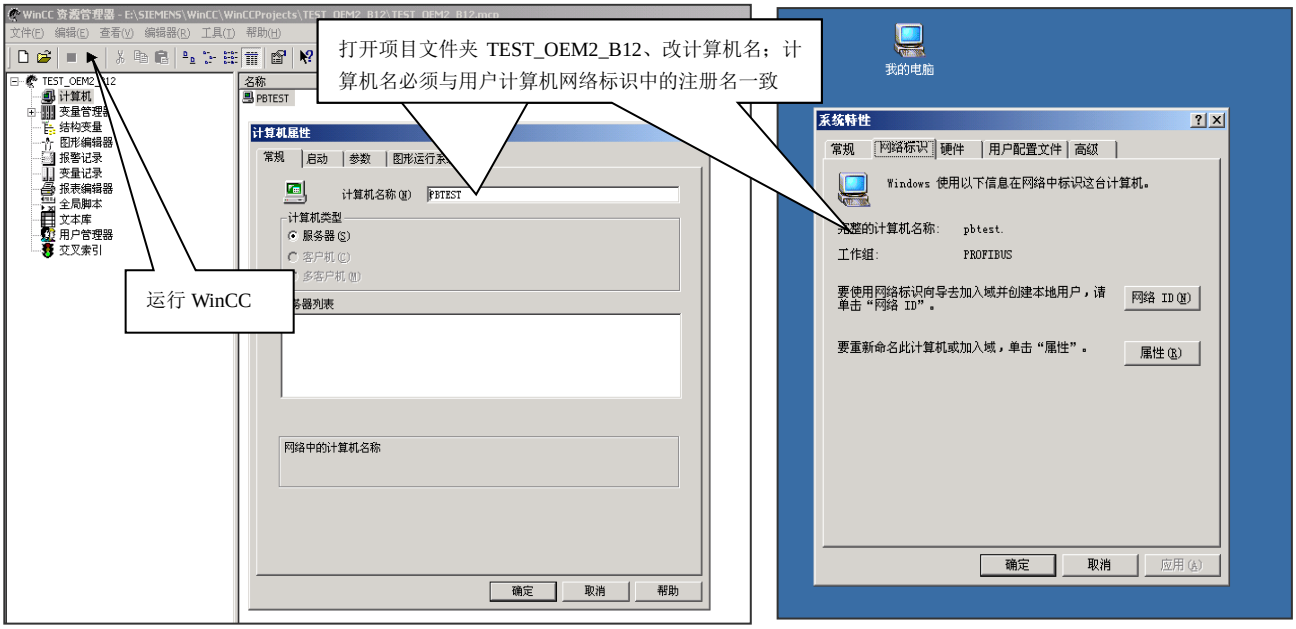


图 5.28

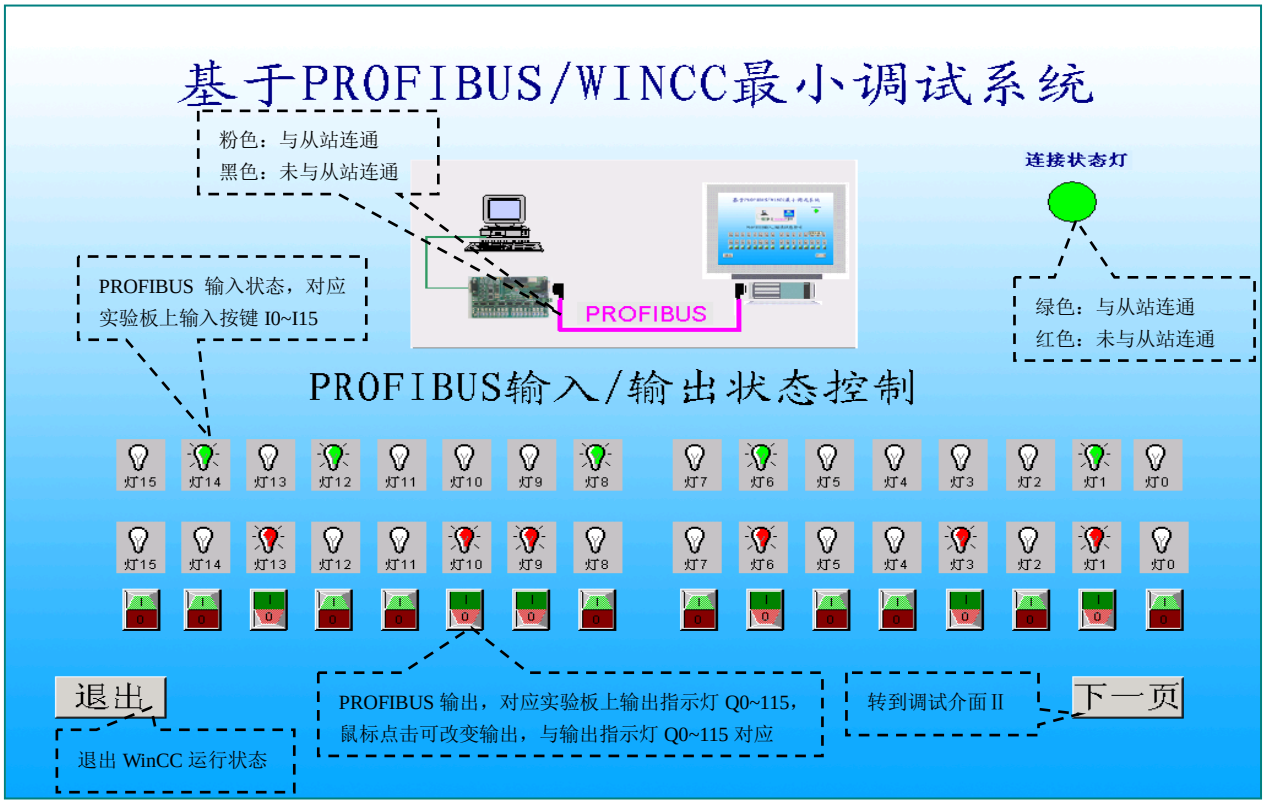


图 5.29 调试画面 I

进入“下一页”，可以看到全部 48 BYTES INPUT/48BYTES OUTPUT 数据，见图 4-3：实验项目调试界面 II 。

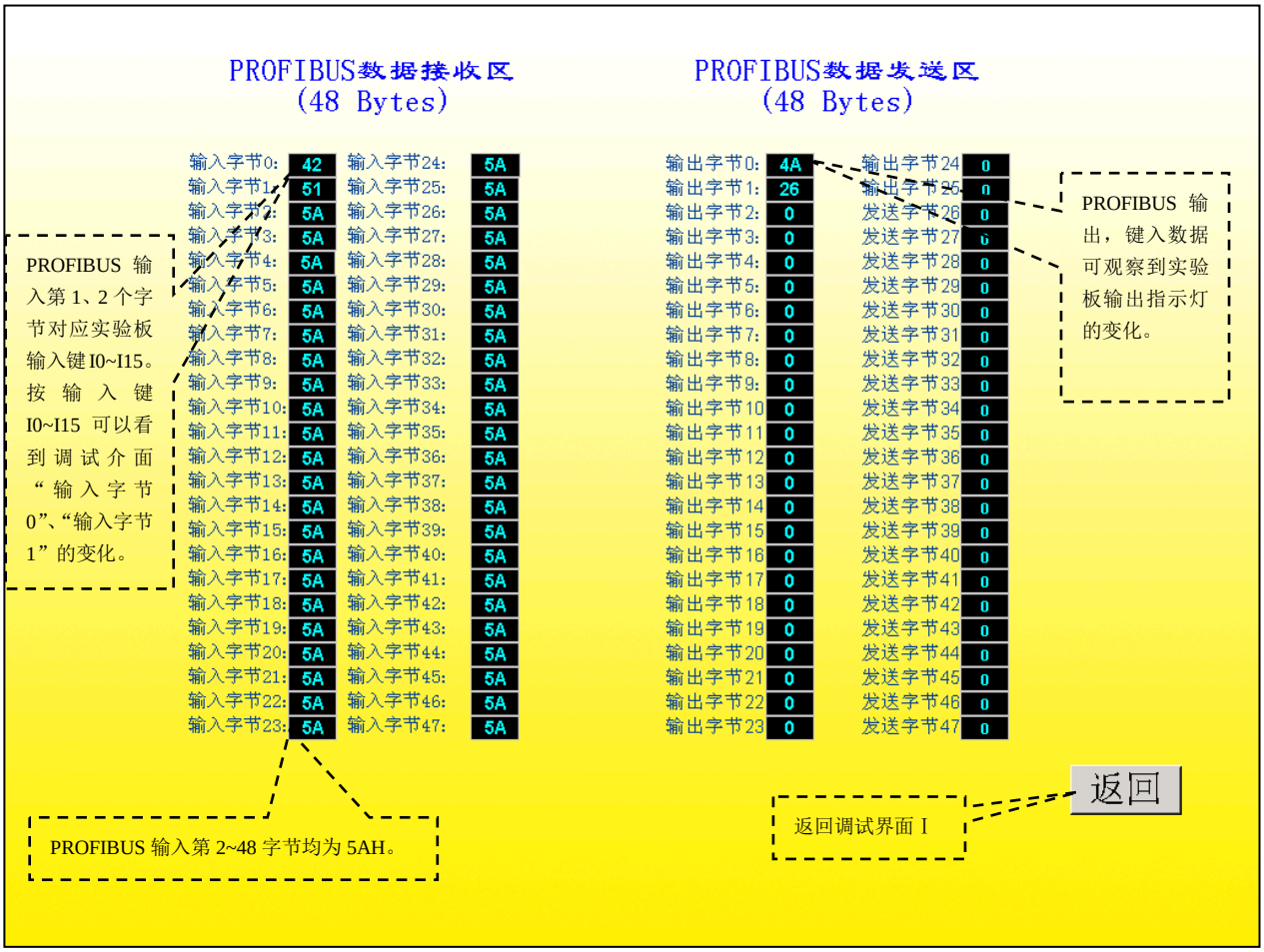


图 5.30　调试画面 II

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：010-82078031、010-62054940　　传真：010-82285084
地址：北京德胜门外教场口 1 号，5 号楼 A-1　邮编：100120
Web:www.c-profibus.com.cn　　Email: tangjy@c-profibus.com.cn