用户手册

SPD1000X 可编程线性直流电源

UM0501X-C02A

版权和声明

版权

商标信息

SIGLENT 是深圳市鼎阳科技股份有限公司的注册商标

声明

- 本公司产品受已获准及尚在审批的中华人民共和国专利的保护。
- 本公司保留改变规格及价格的权利。
- 本手册提供的信息取代以往出版的所有资料。
- 未经本公司许可，不得以任何形式或手段复制、摘抄、翻译本手册的内容。

一般安全概要

了解以下安全性预防措施，保护人身安全，并防止损坏本产品以及与产品相连的任何产品。为避免可能存在的危险，请按照规定使用本产品。

正确使用电源

只使用所在国家认可的本产品专用电源线。

电源供应

AC 输入电压 100V/120V/220V/230V ±10%，50/60HZ

保险丝

保险丝型号：100V/120V：T6.3A/250V；220V/230V：T3.15A/250V

开机前确保使用正确的保险丝型号；

保险丝替换前不要连接电源线；

替换保险丝前确定保险丝烧断原因。

将产品接地

本产品通过电源的接地导线接地。为避免电击，接地导体必须与地相连。在连接本产品的输入与输出之前，请务必将本产品接地。

查看所有终端额定值

为避免起火或过大电流的冲击，请查看产品上所有的额定值和标记说明，在连接产品前查阅产品手册以了解额定值的详细信息。

保持适当的通风

通风不良会引起仪器温度升高，进而引起仪器损坏，使用时请保持良好的

通风，并定期检查通风口和风扇。

操作环境

位置：户内、无强光、无尘、几乎无干扰性污染；

相对湿度：＜80%

海拔：＜2000m

温度：0℃到 40℃

防静电保护

静电会造成仪器损坏，应尽可能在防静电区进行测试。在连接电缆到仪器前，应将其内外导体短暂接地以释放静电。

请勿在易燃易爆的环境下操作

为避免仪器损坏或人身伤害，请勿在易燃易爆的环境下操作仪器。

保持产品表面的清洁和干燥

为避免灰尘或空气中的水分影响仪器性能，请保持产品表面的清洁和干燥。请勿将仪器放置在长时间受到日照的地方。请根据使用情况定期对仪器进行清洁。方法如下：

1. 断开电源。
2. 用柔和的清洁剂或清水浸湿软布擦拭仪器外部，请注意不要刮伤 LCD 显示屏

怀疑产品出故障时，请勿进行操作。

如果您怀疑产品出现故障，请联络 **SIGLENT** 授权的维修人员进行检测。任何维护、调整或零件更换必须由 **SIGLENT** 相关负责人执行。

安全术语和标记

本手册中的术语。以下术语可能出现在本手册中：

警告

警告性声明指出可能会危害操作人员生命安全的条件和行为。

注意

注意性声明指出可能导致本产品损坏或数据丢失的条件和行为。

本产品上的术语。以下术语可能会出现在本产品上：

DANGER 表示标记附近有直接伤害危险存在。

WARNING 表示标记附近有潜在的伤害危险。

CAUTION 表示对本产品及其他财产有潜在的危险。

本产品上的标记。以下标记可能会出现在本产品上：

警告高压

保护性终端

小心

测量接地端

电源开关

SPD1000X 简介

SPD1000X 可编程直流电源配备了 2.8 英寸 TFT-LCD 显示屏，具有可编程和实时波形显示功能，带给用户全新的体验。其中SPD1168X具有一组16V,8A输出能力，SPD1305X具有一组 30V,5A 输出能力，同时都具有四线补偿模式功能，输出短路和过载保护，可以在不同类型的生产和研究中使用。

SPD1168X

SPD1305X

主要特点：

- 单路高精度可编程输出：

  SPD1168X:16V/8A，总功率 128W

  SPD1305X:30V/5A，总功率 150W

- 结构紧凑，使用方便，功能强大，实验台电源的理想选择
- 稳定，可靠，低噪声：≤ 350uVrms/3mVpp; 快速瞬态响应时间<50us
- 最高 5 位电压和 4 位电流显示，最小分辨率为 1mV、1mA
- 支持面板定时输出功能

- 2.8 英寸的 TFT 液晶显示屏，240 *320 高分辨率
- 两种工作模式：两线模式; 四线补偿模式，最大补偿 1V 压降。
- 兼容 100V/120V/220V/230V 的交流电源输入，以满足不同电网的需求
- 智能温控风扇，有效降低噪音
- 清晰的图形化界面，具有波形显示功能
- 支持 5 组系统参数保存/调取
- 提供 EasyPower 上位机软件，可通过 USB、LAN 实现实时控制，支持 SCPI 程控命令集和 LabView 驱动包，满足远程控制和通信需求。

目录

第一章 入门指南

本章介绍 SPD1000X 的面板和显示界面，首次使用仪器的注意事项以及新机检查。通过本章的介绍，您可快速了解 SPD1000X 的操作方法。本章内容如下：

- 一般性检查
- 前面板
- 后面板
- 连接电源
- 用户界面
- 输出检查
- 更换保险丝

1.1 一般性检查

请您按照以下步骤执行新机检查：

1. **检查运输包装**

   如运输包装已损坏，请保留被损坏的包装和防震材料，直到货物经过完全检查且仪器通过电性和机械测试。因运输造成的仪器损坏，由发货方和承运方联系赔偿事宜，**SIGLENT** 恕不进行免费维修或更换。

2. **检查整机**

   若存在机械损坏、缺失，或者仪器未通过电性和机械测试，请及时联系您的 **SIGLENT** 经销商。

3. **检查随机附件**

   请根据装箱单检查随机附件，如有损坏或缺失，请联系您的 **SIGLENT** 经销商。

1.2 前面板

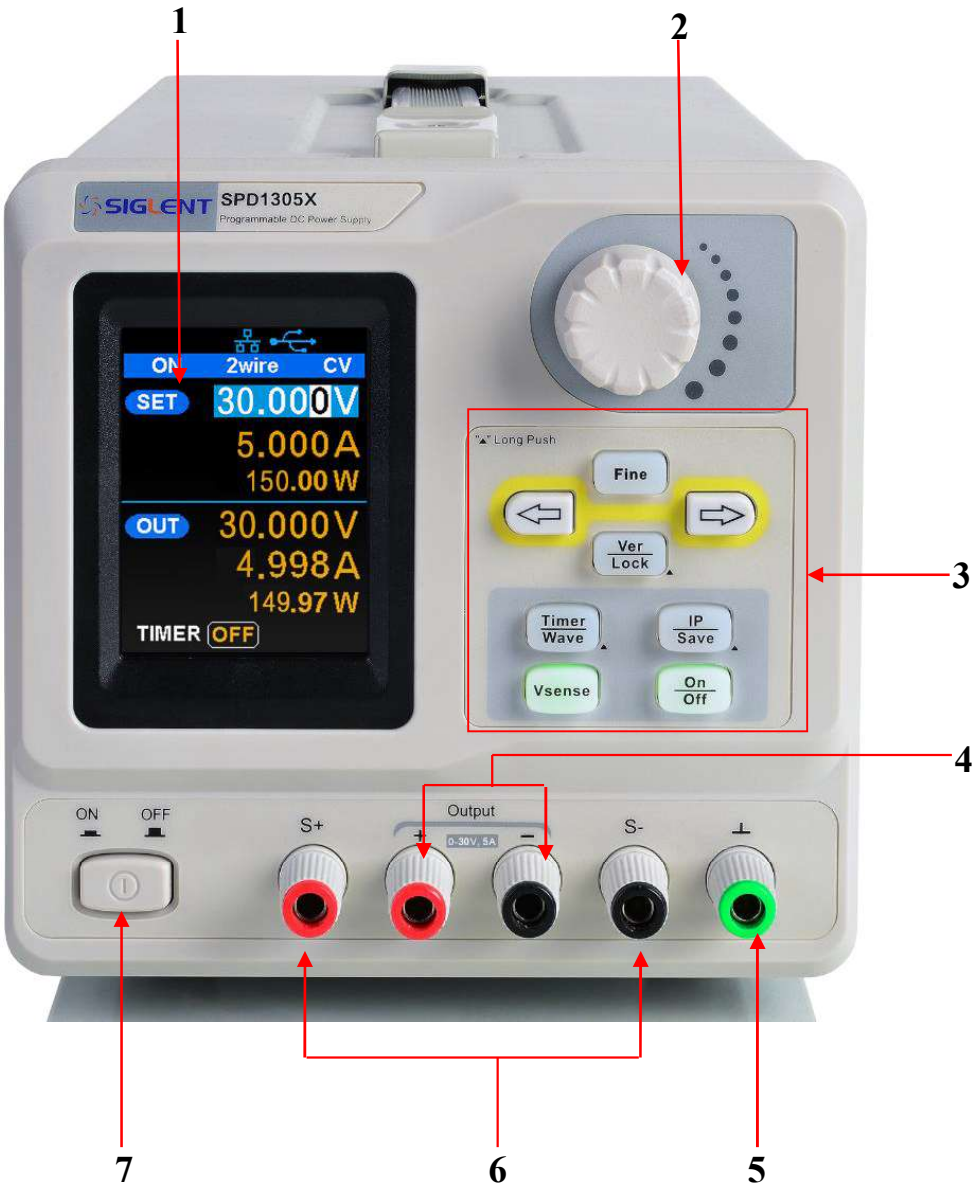


图 1-1 SPD1000X 前面板

1. LCD

2.8英寸的TFT显示屏，用于显示系统参数设置、系统输出状态、菜单选项以及提示信息等

2. 旋钮

设置参数时，旋转旋钮可以增大或减小光标处的数值。

设置对象（定时器开关、存储或读取文件等）时，旋转旋钮可以快速移动光标位置或切换选项，按下旋钮可以使当前设置生效。

3. 功能按键与输出开关

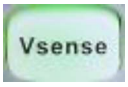
设置参数时，用于移动光标选择数值的数位。

左右方向键，移动光标选择所要设置的参数。长按左键以减，长按右键以加。

短按该键，查看系统信息；长按该键，开启锁键功能。

短按该键，进入定时器设置界面，长按左键以减，长按右键以加，在定时器界面或主界面，长按On/Off键以开关定时器；长按该键，进入波形图显示界面。

短按该键，进入IP设置界面，再长按左键以减，长按右键以加，选中HDCP，短按On/Off键以开/关；长按该键，进入存储系统，短按Fine以选择子项，长按Fine以确定选择。



按下该键，打开/关闭 Sense（远端感应）工作模式。

按下该键，打开/关闭通道输出。

4. 输出端子

用于输出电流和电压。

5. 接地端子

该端子与机壳、地线（电源线接地端）相连，处于接地状态。

6. Sense 端子

用于检测负载端实际电压以补偿负载引线引起的压降。

7. 电源键

用于打开或关闭仪器。

1.3 后面板

图 1-2 SPD1000X 后面板

1. **警告信息**

   提示仪器接地、非专业人员勿拆装机器等注意事项。

2. 输入电源要求

交流输入电源的频率、电压与保险丝规格的对应关系。

3. 电源插孔

交流电源输入接口。

4. 保险丝

所需的保险丝规格和实际的输入电压有关（请参考仪器后面板“输入电源要求”的说明）。

5. 电压选择器

用于选择输入电压的规格（100V/120V/220V/230V）。

6. LAN 接口

通过 RJ45 接口接入局域网。

7. USB DEVICE 接口

通过该接口，仪器作为“从设备”与计算机连接。

8. 风扇通风口

1.4 连接电源

SPD1000X电源支持多种规格的交流电源输入，连接不同规格的输入电源时，后面板电压选择器的设置也不同，如下表所示。

表 1-1 交流输入电源规格

交流输入电源	电压选择器设置
100Vac±10%，50Hz~60Hz	
120Vac±10%，50Hz~60Hz	
220Vac±10%，50Hz~60Hz	
230Vac±10%，50Hz~60Hz	

请严格按照如下步骤连接电源。

1. 检查输入电源

请确保欲连接到仪器的交流电源符合表1-1中的要求。

2. 检查后面板电压选择器

请确保仪器后面板电压选择器的设置（100V、120V、220V 或 230V）与实际输入电压相匹配（匹配关系请参考表 1-1）

3. 检查保险丝

仪器出厂时，已安装指定规格的保险丝。请参考仪器后面板“输入电源要求”的说明确保保险丝与实际输入电压相匹配。

4. 连接交流电源后开机

请使用附件提供的电源线将仪器连接至交流电源。按下前面板的电源开关按键  ，仪器启动并进入开机界面，稍后启用默认设置状态。

警告

切换输入电源电压前，请先断开电源连线，再设置电压选择器至相应档位。

警告

为避免电击，请确认仪器已经正确接地。

1.5 用户界面

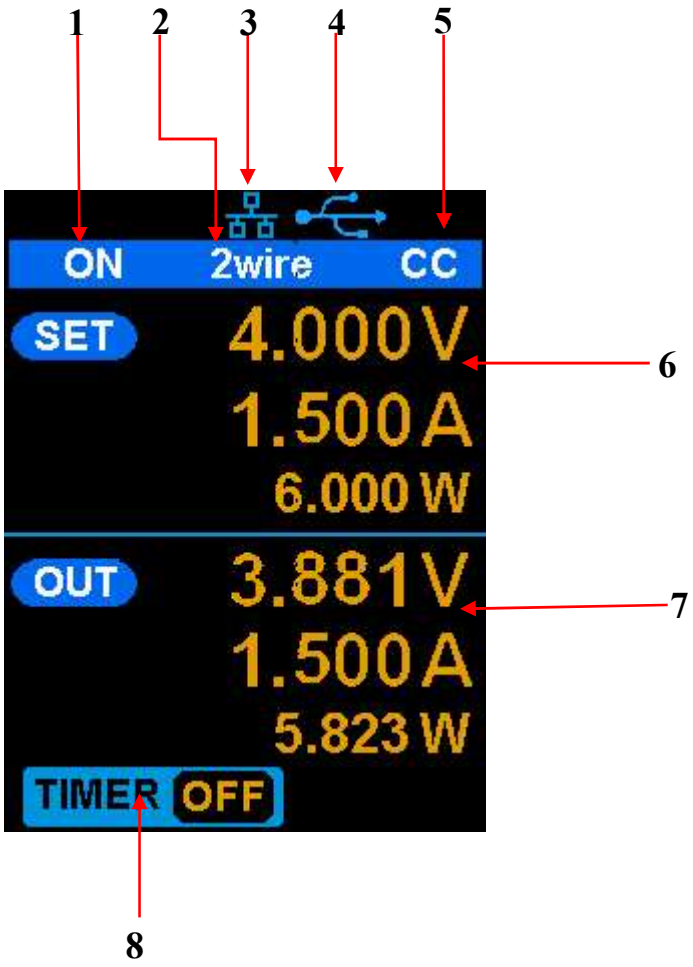


图 1-3 用户界面

1. **输出状态**

   ON：打开，OFF：关闭。

2. **工作模式**

   2 Wire：二线模式，4 Wire：四线（远端感应）模式。

3. **LAN 口连接标识**

   仪器通过 LAN 口连接到网络时，显示该标识。

4. USB 连接标识

仪器通过 USB DEVICE 接口与计算机连接时，显示该标识。

5. 输出模式

CV：恒压，CC：恒流。

6. 设置值

电压、电流、功率设置值。

7. 实际输出

电压、电流、功率实际输出。

8. 定时器状态

ON：打开；OFF：关闭。

1.6 输出检查

输出检查主要包括输出端空载时的电压检查和短路时的电流检查，从而确保仪器可以正确响应前面板操作。

1、电压输出检查

（1） 仪器空载，开启电源，并确认通道的电流设置不为零。

（2） 按下 On/Off ，通道处于恒压（CV）模式，检查“SPD1168X电压可否从 0 调节到最大值 16V”，“SPD1305X电压可否从 0 调节到最大值 30V”。

2、电流输出检查

（1） 打开电源，并确认通道的电压设置不为零。

（2） 使用外表有绝缘的导线，连接 Output 输出端子的正负极；

（3） 按下 On/Off 键，通道处于恒流（CC）模式，检查SPD1168X电流可否从 0 调节到最大值 8A”，“SPD1305X电流可否从 0 调节到最大值 5A”

1.7 更换保险丝

所需保险丝的规格与实际的输入电压有关，如下表所示。您也可以参考仪器后面板的“输入电源要求”。

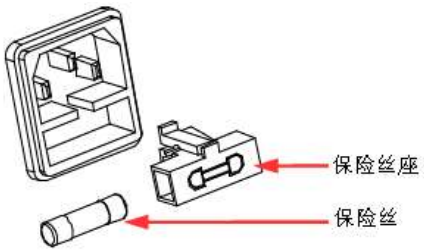
表 1-2 保险丝规格

输入电压	保险丝规格
100Vac/120Vac	T6.3A
220Vac/230Vac	T3.15A

如需更换保险丝，可按如下步骤进行操作。

1. 关闭仪器，移除电源线。

2. 使用小一字螺丝刀插入电源插口处的凹槽，轻轻撬出保险丝座。



3. 若需要，请手动调节电源电压选择器选择与实际输入电压相匹配的电压档位（请参考表 1-1）。

4. 取出保险丝并更换指定规格的保险丝（请参考仪器后面板的“输入电源要求”或表 1-2）。

5. 将保险丝座重新插入电源插口（请注意方向）。

警告

为避免人身伤害，更换保险丝前，前先切断电源；为避免电击或火灾，连接电源前，请选择与实际输入电压相匹配的电源规格，并更换该规格下适用的保险丝。

第二章 控制面板操作

本章将详细介绍 SPD1000X 的控制面板的功能及其操作方法。本章内容如下：

- 输出综述
- 二线模式
- 四线模式
- 网络接口设置
- 存储和调用
- 定时器
- 波形显示
- 版本信息
- 按键锁
- 版本升级

2.1 输出综述

SPD1000X 提供一组可调独立输出，输出额定值：

SPD1168X:0~16V，0~8A；SPD1305X:0~30V，0~5A；

两种输出模式：恒压输出（CV）和恒流输出（CC）；

两种工作模式：二线模式和四线（远端感应）模式。

恒流输出/恒压输出：

恒流模式下，输出电流为设定值，并可通过前面板控制。界面上显示输出模式为“CC”，电流维持在设定值，此时电压值低于设定值，当输出电流低于设定值时，则自动切换到恒压模式；

恒压模式下，输出电流小于设定值，输出电压通过前面板控制。界面上显示输出模式为“CV”，电压值保持在设定值，当输出电流值达到设定值，则切换到恒流模式。

二线模式/四线（远端感应）模式：

二线模式下，当输出打开时，仪器自动检测并显示 Output 端子的实际输出。界面上显示工作模式为“2 Wire”；

四线模式下，当输出打开并将 Sense 端子连接到负载两端时，仪器自动检测并显示负载端的实际输入。界面上显示工作模式为“4 Wire”。

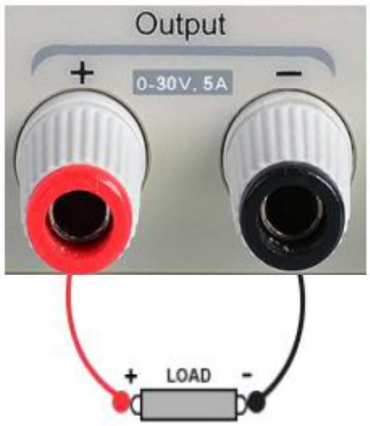
2.2 二线模式

本节介绍在二线模式下，设置电源输出的操作方法。

操作方法：

1. **连接输出端子**

   如下图所示，将负载与输出端子连接。



注意

连接时注意正负电极，以免损坏仪器或与仪器连接的设备。

2. **设置电压和电流输出**

   a) 通过方向键移动光标选择需要修改的参数（电压、电流），

   b) 按 Fine 键选择数位，再旋转多功能旋钮改变相应参数值。

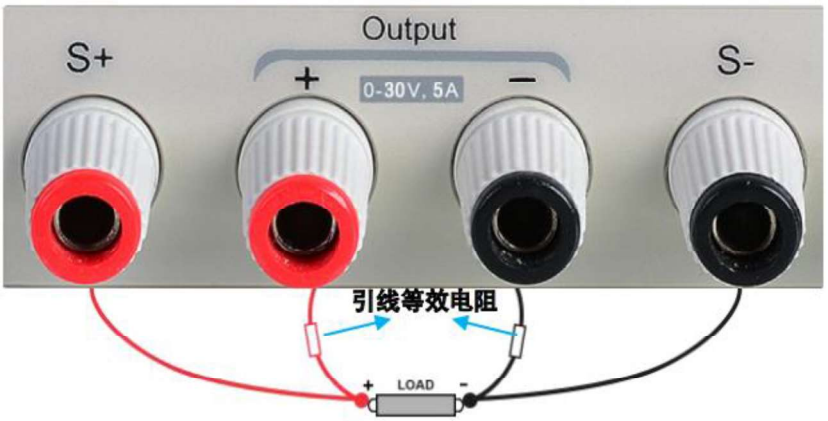
3. **打开输出**

   确保当前模式为二线模式（“Vsense”按键灯不亮，界面显示工作模式为“2 Wire”）。按下 On/Off 键，按键灯被点亮，通道输出打开，界面显示输出状态为“On”。

注意： 电源内置过压保护，当输出端实际电压大于过压值（SPD1168X为22V±2V，SPD1305X为36V±2V)时，输出端将自动短路，并限制电压输出。此时需要重新打开通道开关，才能恢复正常输出。

2.3 四线模式

电源在输出大电流时，负载引线上的压降将变得不可忽略。为确保负载获得准确的压降，SPD1000X提供四线（远端感应）工作模式。在该模式下，检测的是负载端电压而不是电源输出端的电压，这使仪器能自动补偿负载引线引起的压降，从而确保用户设定的电源输出值与负载所获得的电压一致。前面板Sense连线方式如下图所示。



操作方法：

1. **连接输出端子和 Sense 端子**

   按上图所示将前面板的输出端子、Sense端子分别连接至负载两端。连接时，注意极性。

2. **设置电压和电流输出**

   a) 通过方向键移动光标选择需要修改的参数（电压、电流），

   b) 按 Fine 键选择数位，再旋转多功能旋钮改变相应参数值。

3. **打开四线模式**

   按下 Vsense 键，按键灯被点亮，界面上显示工作模式为“4 Wire”。

4. **打开通道输出**

按下 On/Off 键，按键灯被点亮，通道输出打开，界面显示输出状态为“On”。

注意：四线模式下，电源最大补偿压降为 1V，当输出端与 Sense 端电压的压差大于 1V 时，电源将自动关闭输出。

2.4 网络接口设置

SPD1000X 支持 USB Device 和 LAN 接口。您可以通过这些接口远程控制 SPD1000X。当使用 LAN 接口时，请首先设置接口参数。

操作方法：

1. 使用网线将电源后面板上的 LAN 口与计算机或计算机所在网络进行连接；
2. 短按 IP/Save 键，进入网络设置界面
3. 设置完IP值后，按下多功能旋钮或长按Fine键使设置生效，再连续按左右方向键，使光标停留在 DHCP 行上，然后旋转旋钮设置 DHCP 为ON或者OFF，再按下多功能旋钮或短按On/Off键使IP打开或关闭。

   - **ON**：电源将根据当前接入网络，自动设置 IP 地址、子网掩码和网关自动加载。
   - **OFF**：用户可手动设置 IP 地址、子网掩码和网关。

     - 短按左/右方向键，改变光标位置
     - 旋转多功能旋钮或长按左、右方向键改变数值
     - 按下 Fine 键改变数位
     - 按下旋钮或长按Fine键保存设置（只有按下旋钮或长按Fine键，所有设置才能生效）
4. 再次短按 IP/Save 键，退出网络设置界面，返回主界面。

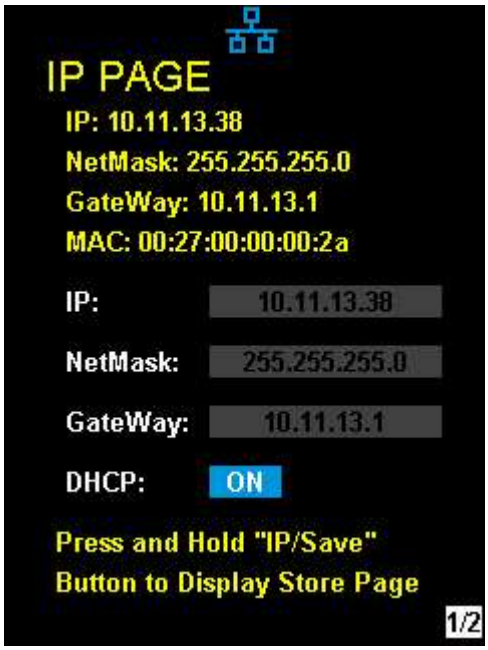


图 2-1 网络接口设置界面

2.5 存储和调用

SPD1000X允许用户将当前的仪器状态（包括工作模式、电压/电流设置和定时器参数等）保存至内部存储器中，并在需要时对已保存文件进行调用。

- **保存**

操作步骤：

1. 设定要保存的状态；
2. 长按 IP/Save 键，进入存储设置界面；
3. 按方向键，移动光标至“FILE CHOICE”；
4. 旋转旋钮或短按Fine键，选择文件保存的位置（FILE1~FILE5）；
5. 按方向键，移动光标至“OPER CHOICE”；
6. 旋转多功能旋钮，选择“STORE”，按下旋钮或长按Fine键选择“OK”，保存当前设定。保存成功后，相应的文件名会变成黄色。

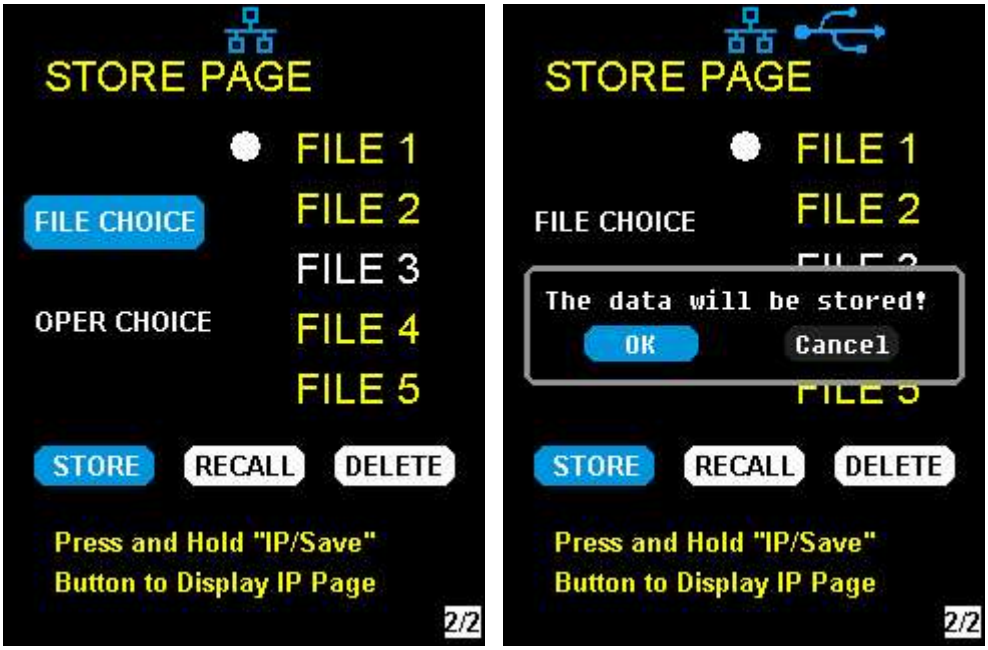


图 2-2 “保存”操作界面

- **读取**

操作步骤：

1. 长按 IP/Save 键，进入存储设置界面；
2. 按左右方向键，移动光标至“FILE CHOICE”；
3. 旋转旋钮或短按Fine键，选择准备调出的仪器状态文件（FILE1~FILE5）；
4. 按方向键，移动光标至“OPER CHOICE”；
5. 旋转多功能旋钮，选择“RECALL”，按下旋钮或长按Fine键选择“OK”，读取已保存的仪器状态文件。

图 2-3 “读取”操作界面

- **删除**

1. 长按 IP/Save 键，进入存储界面；
2. 按左右方向键，移动光标至“FILE CHOICE”；
3. 旋转旋钮或短按Fine键，选择准备删除的仪器状态文件（FILE1~FILE5）；
4. 按方向键，移动光标至“OPER CHOICE”；
5. 旋转多功能旋钮，选择“DELETE”，按下旋钮或长按Fine键选择“OK”，删除已保存的仪器状态文件


STORE PAGE
FILE 1
FILE 2
FILE 3
FILE 4
FILE 5
FILE CHOICE
OPER CHOICE
STORE
RECALL
DELETE
Press and Hold "IP/Save"
Button to Display IP Page
2/2
STORE PAGE
FILE 1
FILE 2
FILE CHOICE
The data will be deleted!
OK
Cancel
STORE
RECALL
DELETE
Press and Hold "IP/Save"
Button to Display IP Page
2/2

图 2-4 “删除”操作界面

2.6 定时器

SPD1000X 提供定时器功能。定时器可以保存五组定时设置，每组设置之间相互独立，可以根据需要，设定参数范围内的任意电压和电流值。定时器支持连续输出，且每组最长定时时间为 10000 S。

- **设置定时器参数**

操作步骤：

1. 短按 Timer/Wave 键，进入定时器设定的界面，同时按键灯被点亮；
2. 按方向键，移动光标位置，选择需要设置的参数；
3. 使用 Fine 键和多功能旋钮或长按左右方向键，设定对应的参数值；
4. 再次短按 Timer/Wave，退出定时器界面，返回主界面。

图 2-5 定时器设置界面

- **开启/关闭定时器**

方法 1：

1. 在主界面按方向键，移动光标选中 TIMER 菜单；
2. 旋转旋钮，打开定时器状态为“ON”；

3. 按下旋钮，启动定时器工作组。

4. 旋转旋钮，设置定时器状态为“OFF”， 按下旋钮，关闭定时器。

5. 或者长按On/Off键，打开或关闭定时器。

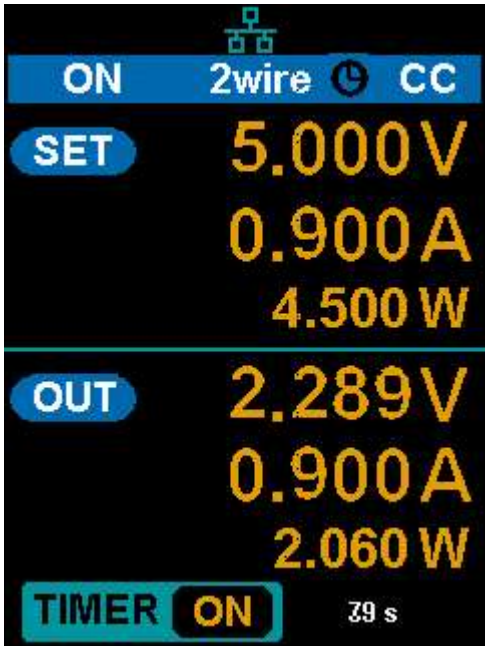


图 2--6 在主界面打开定时器

方法 2：

1、 短按 Timer/Wave 键，进入定时器界面

2、 按下旋钮，启动定时器工作组。

3、 再次按下旋钮，关闭定时器

4、或者长按On/Off键，打开或关闭定时器。

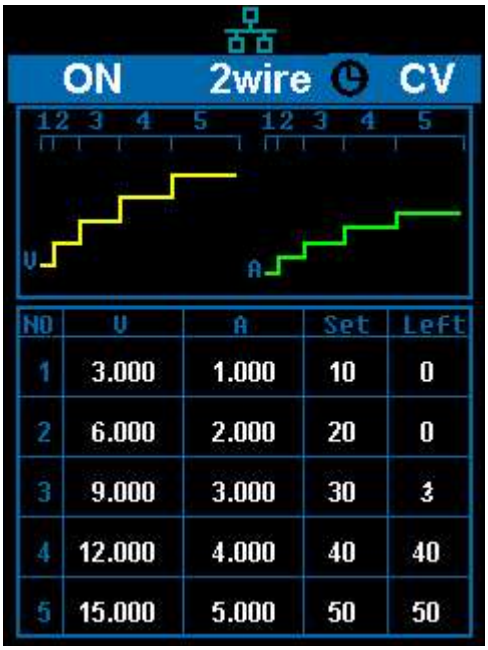


图 2-7 在定时器界面打开定时器

启动定时器之后，若按下通道开关 On/Off 键关闭通道输出，则计时停止。当再次打开通道输出时，定时器将从上次停止的时刻继续计时；倒计时结束后，定时器自动关闭。

2.7 波形显示

SPD1000X 可通过曲线绘图的形式，实时显示通道的输出电压与电流的变化情况。

操作方法：

1. 长按 Timer/Wave 键，开启通道波形显示，同时，按键灯被点亮，并进入波形显示界面。
2. 按下 On/Off 键，打开输出，此时可以观察通道输出参数（电流/电压）的实时变化。

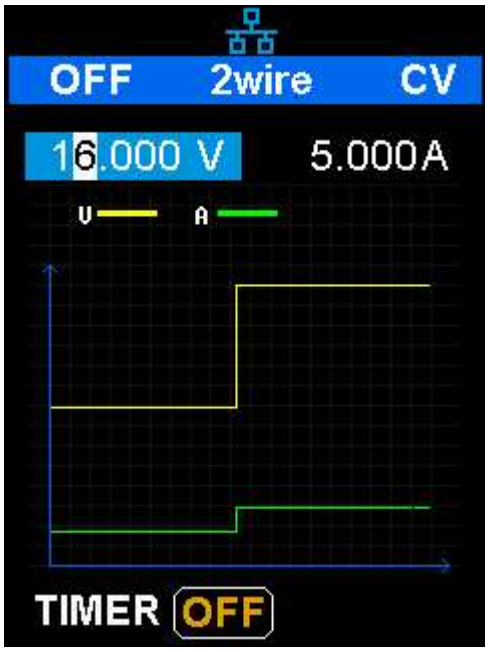


图 2-8 波形显示界面

说明：黄色线表示电压输出曲线，绿色线表示电流输出曲线，坐标纵轴表示输出值的大小。

2.8 版本信息

在任意界面下，短按 Ver/Lock 键，即进入版本信息显示界面。版本信息内容包括：开机次数、软件版本、硬件版本、产品型号、产品序列号。

图 2-9 版本信息界面

2.9 按键锁

SPD1000X 允许用户锁定前面板按键，以避免由于误操作而引起的危险。

在前面板的任意界面下，长按 Ver/Lock 键，开启按键锁功能。此时，除电源键外，前面板上的其他按键不生效。锁键功能开启后，界面上方出现“锁”的图形标志，再次长按 “Ver/Lock” 键，则关闭按键锁功能，界面上方“锁”的图形标志消失。

2.10 版本升级

软件升级运用PC端的管理软件Easypower以及升级文件，通过USB Device或LAN进行升级。升级方式如下：

一. 正常界面下升级：

1、 确认连接好 USB 线或网线，打开 EasyPower 软件，并确认与机器连接上

2、 如图 2-10 所示，点击 Version 菜单上的 Upgrade 的子菜单，进入固件升级的对话框；

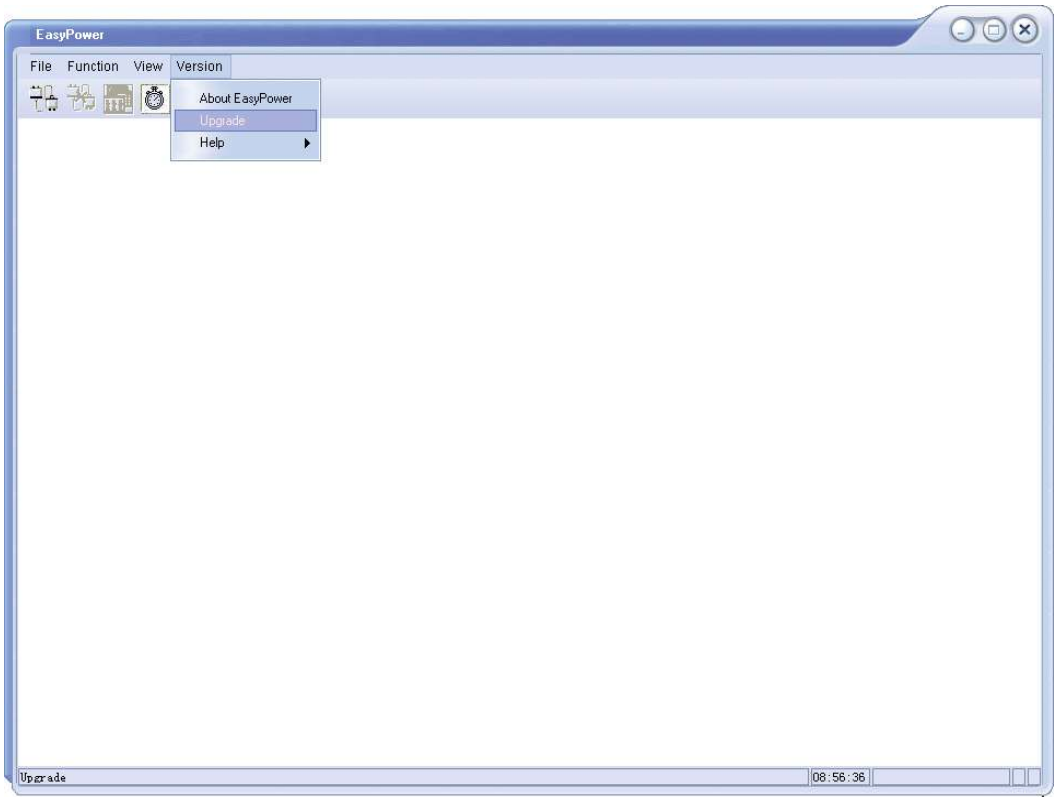


图 2-10 EasyPower 界面

3. 固件升级的对话框如图 2-11 所示，选择“Normal Mode”，点击文件选择按钮 ，会弹出如图 2-12 所示的对话框，可以选择要升级的文

件，文件名的后缀名为.ADS；

图 2-11 升级对话框

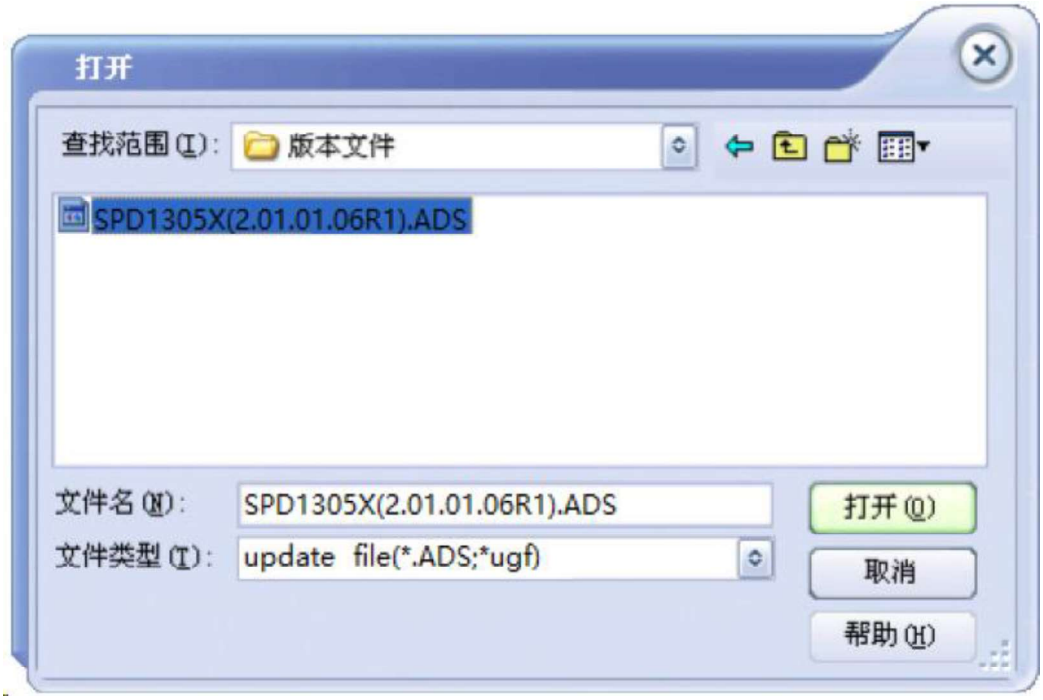


图 2-12 选择升级文件

4. 选择相应的升级文件后点击 Upgrade 按钮，弹出如图 2-13 所示的对话框，选择通过 USB（USBTMC）或 LAN（VXI11）进行升级，当进度条显示完成的时候，升级完成，升级完成后仪器会立刻运行升级后的软件版本。

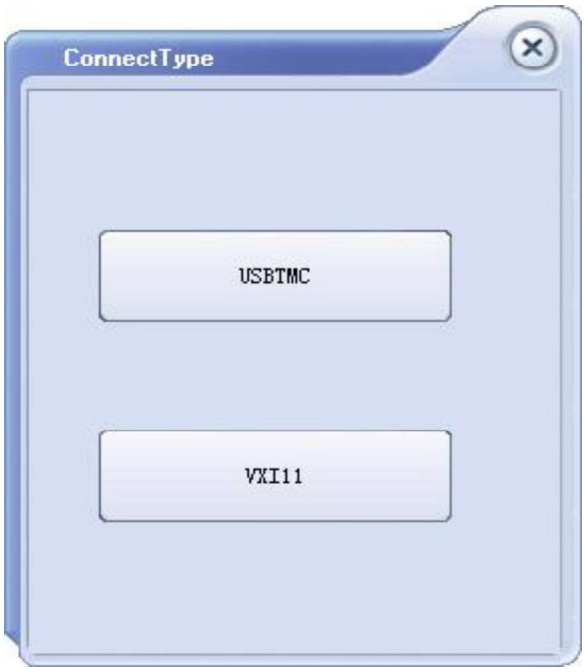


图 2-13 选择连接方式

二. 通过引导程序升级。

当通过第一种方法升级失败的时候，可以通过引导程序进行固件升级。

具体步骤如下：

1. 在仪器开机前按住旋转按钮，然后打开电源的开关，仪器会进入引导程序模式；
2. 进入引导程序模式后，升级方法与方式一基本相同，但在弹出如图 2-11 所示的固件升级对话框时，应选择“Fireware Mode”，其余操作步骤可参考方式一。

第三章 远程控制

SPD1000X 支持通过后面板上的 USB Device 和 LAN 接口与计算机进行通信。用户通过这些接口结合 SCPI（Standard Commands for Programmable Instruments）命令可以对仪器进行编程控制。

本章将介绍如何搭建编程环境，并对 SPD1000X 支持的 SCPI 命令进行说明。本章内容如下：

- 控制方式
- 语法惯例
- 命令概要
- 命令说明
- 编程实例

3.1 控制方式

基于 NI-VISA

用户可以通过使用NI （National Instruments Corporation）公司的NI-VISA，实现对仪器的远程控制。关于NI-VISA，有完整版和实时版（Run-Time Engine version）。完整的版本包括NI设备驱动器和一个名为NI MAX的工具。NI MAX是一个用户接口，用于控制该设备。实时版本比完整版小得多，它只包括NI设备驱动程序

例如，你可以从 http://www.ni.com/download/ni-visa-5.4/4230/en/下载并安装 NI-VISA 5.4 完整版。。

随后使用USB数据线将SPD1000X（通过后面板的USB Device 接口）与计算机相连，或使用网线将SPD1000X（通过后面板的LAN接口）连接至计算机所在的局域网中。

基于NI-VISA，用户通过两种方式对SPD1000X进行远程控制，一种通过上位机软件EasyPower；另一种是结合SCPI命令进行自定义编程，有关详细信息请参阅编程实例。

基于 Socket

用户也可以使用 Socket 通过网口和 SPD1000X 进行基于 TCP/IP 协议的通信。Socket 通信是计算机网络一项基本的通信技术，它允许应用程序通过网络硬件和操作系统内置的标准的网络协议机制进行通信。这种方法需要通过 IP 地址和一个固定的端口号来实现仪器和计算机网络之间双向的通信。

SPD1000X 进行 Socket 通信时的端口为 **5025**。

使用网线将SPD1000X（通过后面板的LAN接口）连接至计算机所在的局域网后，用户可结合SCPI命令进行自定义编程实现对SPD1000X的远程控制，有关详细信息请参阅编程实例。

3.2 语法惯例

SCPI 命令为树状层次结构，包括多个子系统，每个子系统由一个根关键字和一个或数个层次关键字构成。命令关键字之间用冒号“**:**”分隔，关键字后面跟随可选的参数设置，命令和参数以“空格”分开，多个参数的，参数之间用逗号“**,**”分隔。命令行后面添加问号“**?**”，表示对此功能进行查询。

大多数的 SCPI 命令是大小写字母的混合。大写字母表示命令的缩写，即短型命令。如果要获得较好的程序可读性，可以使用长型命令。例如：

[CH1:]VOLTage <voltage>

其中 VOLTage 这个关键词。您可以输入 VOLT 或 VOLTage，大小写字母随意结合。因此，VolTaGe、volt 和 Volt 都可以接受。其他格式(如 VOL 和 VOLTAG)将会产生错误。

- 大括号({ })包含了参数选择。大括号不随命令字符串发送。
- 垂直线(|)分隔参数选择。
- 尖括号(< >)表示必须给括号内的参数指定一个值。例如，上述命令尖括号中的<voltage>参数，必须为该参数指定一个值(例如"CH1:VOLT 10") ，尖括号不随命令串一起发送尖括号。
- 可选参数放在方括号内([ ])。如果您未对可选参数指定数值，则仪器将使用默认值。例如，上述命令中的[CH1:]可以省略（例如："VOLT 10"），此时命令将对当前通道进行操作。方括号不随命令串一起发送。

3.3 命令概要

1. *IDN?
2. *SAV
3. *RCL
4. *DEL
5. MEASure子系统
6. CURRent子系统
7. VOLTage子系统
8. OUTPut子系统
9. MODE
10. TIMEr子系统
11. SYSTem子系统
12. IPaddr 子系统
13. MASKaddr子系统
14. GATEaddr子系统
15. DHCP子系统
16. *LOCK子系统

3.4 命令说明

1. *IDN?

命令格式： *IDN?

功能描述： 该条命令用于查询制造商名称、产品型号、产品序列号和软件版本号和硬件版本号。

返回格式： 制造商名称，电源型号，产品序列号，软件版本号，硬件版本号。

典型响应： Siglent Technologies,SPD1305X,SPD1A134600512,2.01.01.06,V1.0

2. *SAV

命令格式： *SAV {1|2|3|4|5}

功能描述： 电源内部提供 5 个存储位置用于存储仪器状态，该命令将直接存储当前仪器状态到指定位置。

举　　例： *SAV 1

3. *RCL

命令格式： *RCL {1|2|3|4|5}

功能描述： 电源内部提供 5 个存储位置用于存储仪器状态，该命令将调用指定位置的仪器状态。

举　　例： *RCL 1

4. *DEL

命令格式： *DEL {1|2|3|4|5}

功能描述： 电源内部提供 5 个存储位置用于存储仪器状态，该命令将删除指定位置的系统状态保存文件。

举　　例： *DEL 1

5. MEASure

命令格式： MEASure:CURRent?

功能描述： 查询输出端子上的电流值。

举　　例： MEASure:CURRent?

典型响应： 3.000

命令格式： MEASure:VOLTage?

功能描述： 查询输出端子上的电压值。

举　　例： MEASure:VOLTage?

典型响应： 10.000

命令格式： MEASure:POWEr?

功能描述： 查询通道输出端子上的功率值。

举　　例： MEASure:POWEr?

典型响应： 90.000

6. CURRent

命令格式： [CH1:]CURRent <current>

功能描述： 设定输出电流值。

举　　例： CURRent 0.5

命令格式： [CH1:]CURRent?

功能描述： 查询电流值设置值。

举　　例： CURRent?

典型响应： 0.5

7. VOLTage

命令格式： [CH1:]VOLTage <voltage>

功能描述： 设定输出电压值。

举　　例： VOLTage 10

命令格式： VOLTage?

功能描述： 查询电压设置值。

举　　例： CH1:VOLTage？

典型响应： 10

8. MODE

命令格式： MODE:SET {2W|4W}

功能描述： 设置工作模式为二线或四线模式

举　　例： MODE:SET 4W

9. OUTPut

命令格式： OUTPut CH1,{ON|OFF}

功能描述： 打开或关闭通道输出。

举　　例： OUTPut CH1,ON

命令格式： OUTPut:WAVE CH1,{ON|OFF}

功能描述： 打开或关闭波形显示功能。

举　　例： OUTPut:WAVE CH1,ON

10. TIMEr

命令格式： TIMEr:SET CH1,{1|2|3|4|5},<voltage>,<current>,<time>

功能描述： 设定定时参数：包括组别{1|2|3|4|5}、电压、电流、输出时间。

举　　例： TIMEr:SET CH1,2,3,0.5,2

命令格式： TIMEr:SET?{1|2|3|4|5}

功能描述： 查询定时器指定组别的定时参数：包括电压，电流，输出时间。

举　　例： TIMEr:SET? CH1,2

典型响应： 3,0.5, 2

命令格式： TIMEr CH1,{ON|OFF}

功能描述： 打开或关闭定时输出功能。

举　　例： TIMEr CH1,ON

11. SYSTem

命令格式： SYSTem:ERRor?

功能描述： 查询并清除队列中的错误信息。

典型响应： 0　No Error

命令格式： SYSTem:VERSion?

功能描述： 用于查询软件版本信息。

典型响应： 2.01.01.06

命令格式： SYSTem:STATus?

功能描述： 用于查询机器的工作状态。

典型响应： 0x51

说　　明： 返回信息是十六进制，所以用户在确认状态的时候，需要转换成二进制格式。对应关系如下表：

位号	对应状态
0	0：恒压模式（CV）　1：恒流模式（CC）
4	0：通道输出关闭　　1：通道输出开启
5	0：二线模式　　1：四线模式
6	0：定时器关闭　　1：定时器开启
8	0：显示数值　　1：显示波形

12. IPaddr

命令格式： IPaddr <IP address>

功能描述： 用于为仪器分配一个静态 Internet 协议(IP)地址。

举　　例： IPaddr 10.11.13.214

说　　明： 当电源当前设置为自动获取网络配置（即 DHCP 处于 ON 状态）时，该命令无效

命令格式： IPaddr?

功能描述： 用于查询仪器当前的 IP 地址设置

典型响应： 10.11.13.214

13. MASKaddr

命令格式： MASKaddr <NetMasK>

功能描述： 用于为仪器分配一个子网掩码。

举　　例： MASKaddr 255.255.255.0

说　　明： 当电源当前设置为自动获取网络配置（即 DHCP 处于 ON 状态）时，该命令无效

命令格式： MASKaddr?

功能描述： 用于查询仪器当前的子网掩码

典型响应： 255.255.255.0

14. GATEaddr

命令格式： GATEaddr <GateWay>

功能描述： 用于为仪器分配一个网关。

举　　例： GATEaddr 10.11.13.1

说　　明： 当电源当前设置为自动获取网络配置（即 DHCP 处于 ON 状态）时，该命令无效

命令格式： GATEaddr?

功能描述： 用于查询仪器当前的网关

典型响应： 10.11.13.1

15. DHCP

命令格式： DHCP {ON|OFF}

功能描述： 打开或关闭仪器的自动获取网络配置功能。

举　　例： DHCP ON

命令格式： DHCP?

功能描述： 用于查询仪器当前的自动获取网络配置功能是否开启

典型响应： DHCP:ON

16. *LOCK

命令格式： *LOCK

功能描述： 开启按键锁，使本地或远程的设置失效。

举　　例： *LOCK

命令格式： *UNLOCK

功能描述： 关闭按键锁，使设置生效。

举　　例： *UNLOCK

3.5 编程实例

本节列出在 Visual C++、Visual Basic、MATLAB、Python 等环境下基于 NI-VISA 或 Socekt 使用 SCPI 命令编程的实例。

基于 NI-VISA 的编程示例

编程准备

1.首先确认您的电脑上是否已经安装NI的VISA库（可到NI网站 http://www.ni.com/china下载）。本文中默认安装路径为C:\Program Files\IVI Foundation\VISA。

2.本文主要应用电源的USB接口与PC通信，部分例子涉及使用LAN接口。请使用USB数据线将万用表后面板的USB Device接口与PC的USB 接口相连。您也可以使用LAN接口与PC通信。

3.电源首次与PC正确连接后，接通仪器电源，此时PC上将弹出“**硬件更新向导**”对话框，请按照向导的提示安装“USB Test and Measurement Device”。

至此，编程准备工作结束。下面将详细介绍在 Visual C++，Visual Basic 和 MATLAB 开发环境中的编程示例。

Visual C++编程示例

环境：Win7 32bit system, Visual Studio

示例内容：使用NI-VISA，通过USBTMC和TCP/ IP访问控制设备，发送命令，读取返回值。。

按以下步骤完成示例：

1、打开Visual Studio,创建一个新的vc++ win32项目。

设置项目环境使用ni-visa库，有两种方法可以使用ni-visa，静态方式和自动方式：

(1) 静态方式：

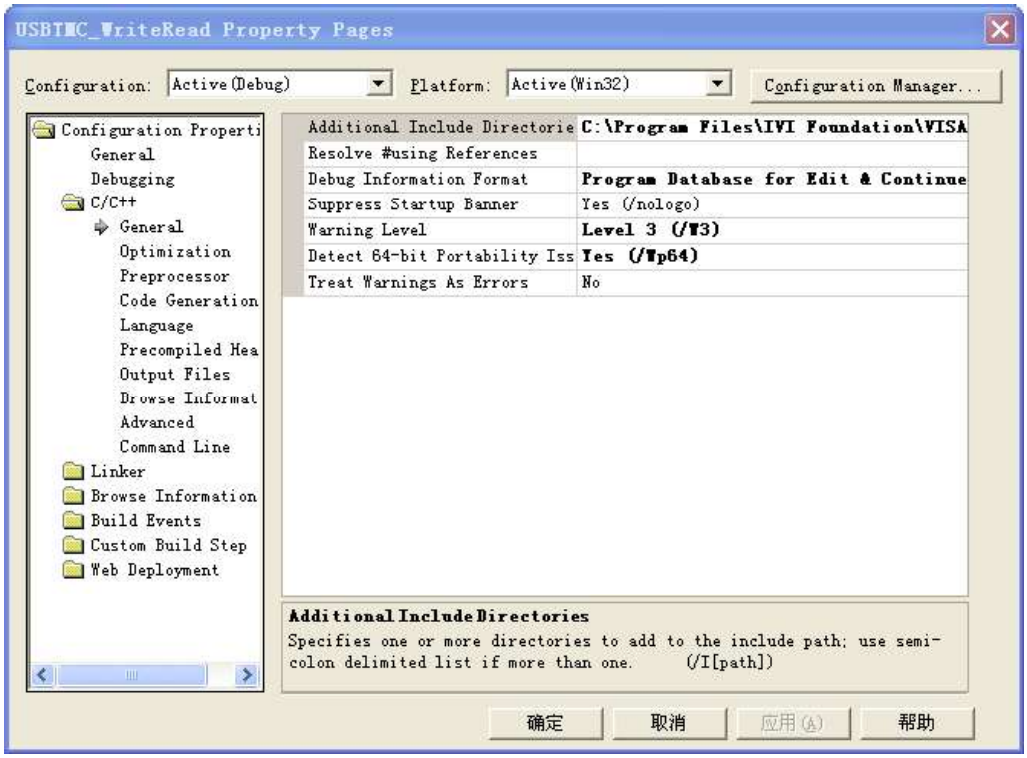
在NI-VISA安装路径查找文件：visa.h,visatype.h,visa32.lib。将它们复制到你的项目，并将它们添加到项目中。在项目.cpp文件，添加如下两行

#include "visa.h"

#pragma comment(lib,"visa32.lib")

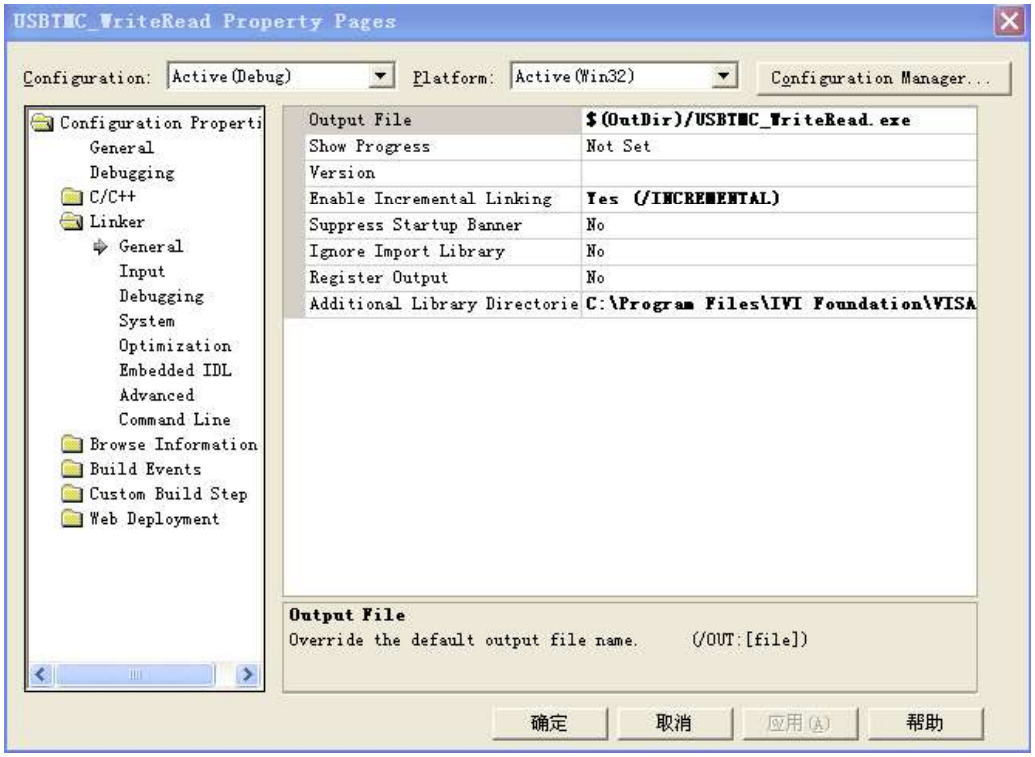
(2) 自动方式：

设置.h文件包括目录，ni-visa安装路径。在我们的电脑，我们设置的路径是：C:\Program Files\IVI Foundation\VISA\WINNT\include。设置这条路径到项目—属性—C/C++—通用—附加包含路径，如图：

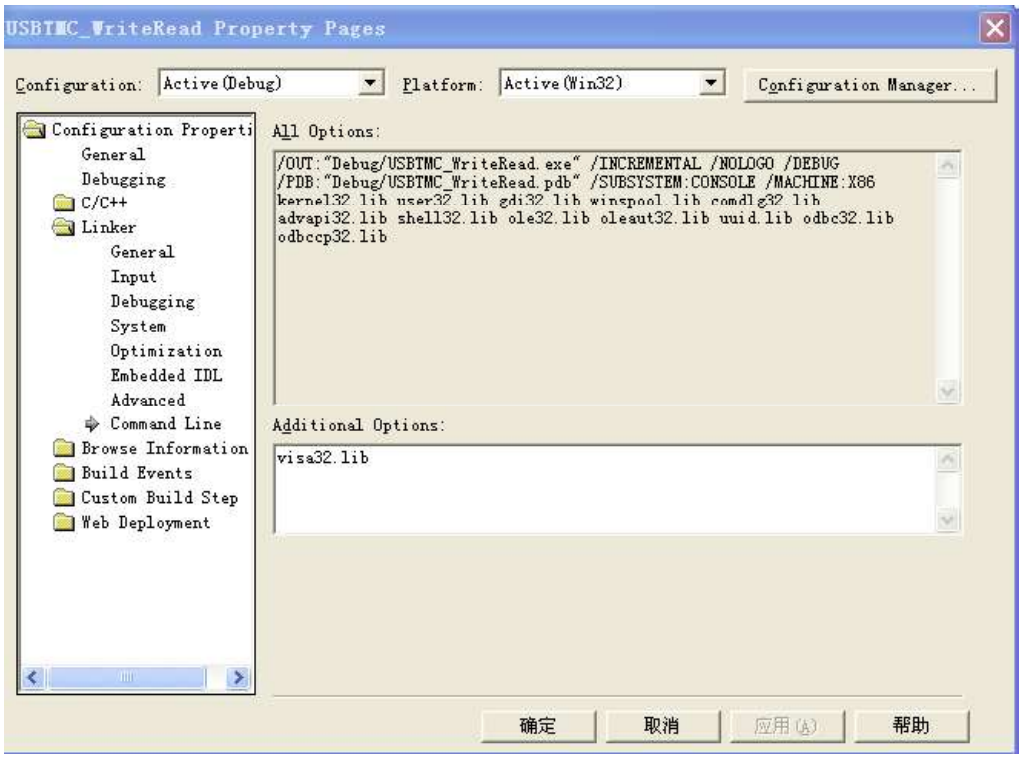


设置库路径设置库文件:

设置库路径：在ni-visa安装路径，在我们的电脑，我们设置的路径是：C：\Program Files\IVI Foundation\VISA\WINNT\LIB\MSC。设置这条路径到项目—性能—连接器—常规—附加库目录，如图：



设置库文件： project---properties---Linker---Command Line---Additional Options:visa32.lib



包括 visa.h file:在 XXX.cpp 文件里：

#include <visa.h>

3、增加代码：

(1)基于USB接口代码:：

写一个Usbtmc_test函数。

```
int Usbtmc_test()
{
    /*   这段代码演示了使用NI-VISA发送同步读取和写入命令到          */
/*   一个USB测试&测量类(USBTMC)仪器。                    */
    /*   这个例子写"* IDN ?\ n "字符串到所有连接到系统的USBTMC     */
/*   设备并试图使用读写函数读回结果。                    */
    /*   代码的一般流程是打开资源管理器                      */
/*   打开VISA会话到仪器                                  */
/*   使用viPrintf写仪器标志查询                          */
    /*   尝试随viScanf读取一个响应                           */
    /*   关闭VISA会话                                        */
    /*********************************************************/
    ViSession defaultRM;
    ViSession instr;
    ViUInt32 numInstrs;
    ViFindList findList;
```

```
    ViStatus status;
    char instrResourceString[VI_FIND_BUFLEN];
    unsigned char buffer[100];
    char stringinput[512];
    int i;
    /*首先，我们必须调用viOpenDefaultRM得到管理器的句柄。              */
    /*我们将在defaultRM存储此手柄。                                    */
    status=viOpenDefaultRM (&defaultRM);
    if (status < VI_SUCCESS)
    {
        printf ("Could not open a session to the VISA Resource Manager!\n");
        return status;
    }
    /**寻找我们的系统中所有的USB TMC VISA资源                          */
    *然后将资源的数目存储在系统中的numInstrs里。                      */
    status = viFindRsrc (defaultRM, "USB?*INSTR", &findList, &numInstrs,
instrResourceString);
    if (status < VI_SUCCESS)
    {
        printf ("An error occurred while finding resources.\nHit enter to continue.");
        fflush(stdin);
        getchar();
        viClose (defaultRM);
        return status;
    }
    /**现在，我们将对所有USB TMC仪器打开VISA会话。我们必须
    * 从viOpenDefaultRM使用句柄，也必须使用一个字符串指示要
    *打开的仪器，这就是所谓的仪器描述符。该字符串的格式可以在功
    *能面板中右键单击参数描述中找到。打开一个会话到设备后，我们
    *将得到一个仪器使用的句柄，在之后使用VISA功能时用到。在这
    *个函数的AccessMode和超时参数是为以后的功能预留。这两个参
    *数被给予值VI_NULL。                                             */
    for (i=0; i<int(numInstrs); i++)
    {
        if (i > 0)
            viFindNext (findList, instrResourceString);
        status = viOpen (defaultRM, instrResourceString, VI_NULL, VI_NULL, &instr);
        if (status < VI_SUCCESS)
        {
            printf ("Cannot open a session to the device %d.\n", i+1);
            continue;
        }
```

```
        /* *在这一点上，我们现在有一个会话打开到USB TMC仪器。现在，
        *我们将使用viPrintf函数发送字符串“*IDN?\n”到设备，要求设备识别。*/
        char * cmmand ="*IDN?\n";
        status = viPrintf    (instr, cmmand);
        if (status < VI_SUCCESS)
        {
            printf ("Error writing to the device %d.\n", i+1);
            status = viClose (instr);
            continue;
        }
        /** 现在我们将尝试从设备读回一个设备信息查询的响应。我们将
        *使用viScanf函数来获取数据。在数据被读出后，响应显示出来 */
        status =    viScanf(instr, "%t", buffer);
        if (status < VI_SUCCESS)
            printf ("Error reading a response from the device %d.\n", i+1);
        else
            printf ("\nDevice %d:%*s\n", i+1,retCount, buffer);
        status = viClose (instr);
    }
    /**现在，我们将关闭会话使用viClose仪器。此操作释放所有系统资源。*/
    status = viClose (defaultRM);
return 0;
}
```

(2)基于LAN口代码:

写一个TCP_IP_Test函数.

```
int TCP_IP_Test(char *pIP)
{
     char outputBuffer[VI_FIND_BUFLEN];
     ViSession defaultRM, instr;
     ViStatus status;
     ViUInt32 count;
     ViUInt16 portNo;
    /*  首先，我们需要打开默认的资源管理器。                    */
    status = viOpenDefaultRM (&defaultRM);
    if (status < VI_SUCCESS)
    {
        printf("Could not open a session to the VISA Resource Manager!\n");
    }
    /*现在，我们将通过TCP / IP设备打开一个会话     */
    char head[256] ="TCPIP0::";
```

```
    char tail[] ="::INSTR";
    char resource [256];
    strcat(head,pIP);
    strcat(head,tail);
    status = viOpen (defaultRM, head, VI_LOAD_CONFIG, VI_NULL, &instr);
    if (status < VI_SUCCESS)
    {
        printf ("An error occurred opening the session\n");
        viClose(defaultRM);
    }
    status = viPrintf(instr, "*idn?\n");
    status = viScanf(instr, "%t", outputBuffer);
    if (status < VI_SUCCESS)
    {
        printf("viRead failed with error code: %x \n",status);
        viClose(defaultRM);
    }else
        printf ("\ndata read from device: %*s\n", 0,outputBuffer);
    status = viClose (instr);
    status = viClose (defaultRM);
    return 0;
}
```

Visual Basic编程示例

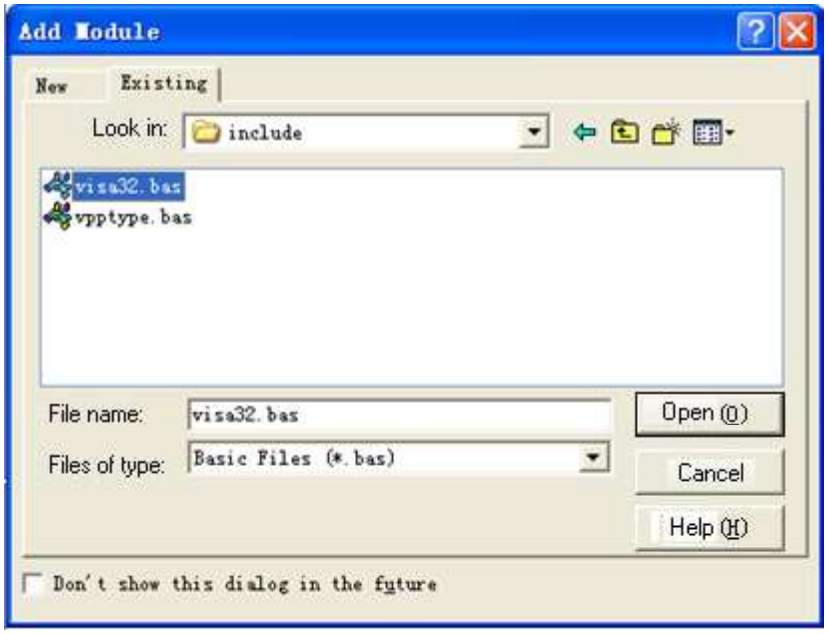
环境: Windows 7 32位系统 , Microsoft Visual Basic 6.0

示例内容：使用NI-VISA，通过USBTMC和TCP/ IP访问控制设备，发送命令，读取返回值。。

按照步骤完成的例子：

1、 打开Visual Basic，建立一个标准的应用程序项目（标准EXE）。

2、 设置项目的环境中使用NI-VISA 库，单击项目的现有标签>>添加模块。搜索visa32.bas中NI-VISA安装路径下的include文件夹文件，并添加文件。



这使得VISA功能和VISA的数据类型在程序中使用。

3、添加代码：

(1)基于USB接口代码：

Usbtmc_test函数.

```
Private Function Usbtmc_test() As Long
    '    这段代码演示了使用NI-VISA发送同步读取和写入命令到
    '    一个USB测试&测量类(USBTMC)仪器。
    '    这个例子写"* IDN ?\ n "字符串到所有连接到系统的USBTMC
    '    设备并试图使用读写函数读回结果。
    '    代码的一般流程是打开资源管理器
    '    打开VISA会话到仪器
    '    使用viPrintf写仪器标志查询
    '    尝试随viScanf读取一个响应
```

```
'     关闭VISA会话
Const MAX_CNT = 200

Dim defaultRM As Long
Dim instrsesn As Long
Dim numInstrs As Long
Dim findList As Long
Dim retCount As Long
Dim writeCount As Long
Dim status As Long
Dim instrResourceString As String * VI_FIND_BUFLEN
Dim buffer As String * MAX_CNT
Dim i As Integer

'首先，我们必须调用viOpenDefaultRM得到管理器的句柄。
'我们将在defaultRM存储此手柄。
status = viOpenDefaultRM(defaultRM)
If (status < VI_SUCCESS) Then
    Debug.Print "Could not open a session to the VISA Resource Manager!"
    Usbtmc_test = status
    Exit Function
End If

'寻找我们的系统中所有的USB TMC VISA资源
'和然后将资源的数目存储在系统中的numInstrs里。
status = viFindRsrc(defaultRM, "USB?*INSTR", findList, numInstrs,
instrResourceString)
If (status < VI_SUCCESS) Then
    Debug.Print "An error occurred while finding resources."
    viClose (defaultRM)
    Usbtmc_test = status
    Exit Function
End If

'现在，我们将对所有USB TMC仪器打开VISA会话。我们必须
'从viOpenDefaultRM使用句柄，也必须使用一个字符串指示要
'打开的仪器，这就是所谓的仪器描述符。该字符串的格式可以在功
'能面板中右键单击参数描述中找到。打开一个会话到设备后，我们
'将得到一个仪器使用的句柄，在之后使用VISA功能时用到。在这
'个函数的AccessMode和超时参数是为以后的功能预留。这两个参
'数被给予值VI_NULL。
For i = 0 To numInstrs
```

```
        If (i > 0) Then
            status = viFindNext(findList, instrResourceString)
        End If
        status = viOpen(defaultRM, instrResourceString, VI_NULL, VI_NULL, instrsesn)
        If (status < VI_SUCCESS) Then
            Debug.Print "Cannot open a session to the device ", i + 1
            GoTo NextFind
        End If

        '在这一点上，我们现在有一个会话打开到USB TMC仪器。现在，
        '我们将使用viPrintf函数发送字符串“*IDN?\n”到设备，要求设备识别。
        status = viWrite(instrsesn, "*IDN?", 5, retCount)
        If (status < VI_SUCCESS) Then
            Debug.Print "Error writing to the device."
            status = viClose(instrsesn)
            GoTo NextFind
        End If

        '现在我们将尝试从设备读回一个设备信息查询的响应。我们将
        '使用viScanf函数来获取数据。在数据被读出后，响应显示出来
        status = viRead(instrsesn, buffer, MAX_CNT, retCount)
        If (status < VI_SUCCESS) Then
            Debug.Print "Error reading a response from the device.", i + 1
        Else
            Debug.Print i + 1, retCount, buffer
        End If
        status = viClose(instrsesn)
NextFind:
    Next i

    '现在，我们将关闭会话使用viClose仪器。此操作释放所有系统资源。
    status = viClose(defaultRM)
    Usbtmc_test = 0
End Function
```

(2)基于LAN口代码:

```
function TCP_IP_Test函数.
Private Function TCP_IP_Test(ip As String) As Long
    Dim outputBuffer As String * VI_FIND_BUFLEN
    Dim defaultRM As Long
    Dim instrsesn As Long
```

```
    Dim status As Long
    Dim count As Long

    '首先，我们需要打开默认的资源管理器。
    status = viOpenDefaultRM (defaultRM)
    If (status < VI_SUCCESS) Then
        Debug.Print "Could not open a session to the VISA Resource Manager!"
        TCP_IP_Test = status
        Exit Function
    End If

    '现在，我们将通过TCP / IP设备打开一个会话
    status = viOpen(defaultRM, "TCPIP0::" + ip + "::INSTR", VI_LOAD_CONFIG,
VI_NULL, instrsesn)
    If (status < VI_SUCCESS) Then
        Debug.Print "An error occurred opening the session"
        viClose (defaultRM)
        TCP_IP_Test = status
        Exit Function
    End If

    status = viWrite(instrsesn, "*IDN?", 5, count)
    If (status < VI_SUCCESS) Then
        Debug.Print "Error writing to the device."
    End If
    status = viRead(instrsesn, outputBuffer, VI_FIND_BUFLEN, count)
    If (status < VI_SUCCESS) Then
        Debug.Print "Error reading a response from the device.", i + 1
    Else
        Debug.Print "read from device:", outputBuffer
    End If
    status = viClose(instrsesn)
    status = viClose(defaultRM)
    TCP_IP_Test = 0
End Function
```

MATLAB编程示例

环境：windows 7 32位系统, MATLAB R2010b

示例内容：使用NI-VISA，通过USBTMC和TCP/ IP访问控制设备，发送命令，读取返回值。

按照步骤完成的例子：

1、打开 MATLAB, 修改当前目录。在本演示中，将当前目录修改为D：\USBTMC_TCPIP_Demo。

2、点击文件>>新建>>脚本（File>>New>>Script）在Matlab界面来创建一个空的M文档

3、添加代码：

(1)基于USB接口代码：

写入 Usbtmc_test函数。

```
function USBTMC_test()
%这段代码演示了使用NI-VISA发送同步读取和写入命令到
%一个USB测试&测量类(USBTMC)仪器。

%创建一个VISA-USB对象连接到USB仪器上
vu = visa('ni','USB0::0xF4EC::0x1300::0123456789::INSTR');

%打开创建的VISA对象
fopen(vu);

%发送字符串“* IDN？”，查询设备信息。
fprintf(vu,'*IDN?');

%请求数据
outputbuffer = fscanf(vu);
disp(outputbuffer);

%关闭VISA对象
fclose(vu);
delete(vu);
clear vu;

end
```

(2)基于LAN口代码：

写入 TCP_IP_Test函数。

```
function TCP_IP_test( IPstr )
%这段代码演示了使用NI-VISA发送同步读取和写入命令到
%一个TCP/IP仪器。

%创建一个VISA-TCPIP对象连接到配置了IP地址的仪器。
vt = visa('ni',['TCPIP0::',IPstr,'::INSTR']);

%打开创建的VISA对象
fopen(vt);

%发送字符串"*IDN?"，查询设备信息
fprintf(vt,'*IDN?');

%请求数据
outputbuffer = fscanf(vt);
disp(outputbuffer);

%关闭VISA对象
fclose(vt);
delete(vt);
clear vt;

end
```

基于 Socket 的编程示例

Python 编程示例

由于操作系统自身就支持 Socket 通信，这种通信方式也是比较简明的。需要注意的是 SPD1000X 使用 Socket 通信的固定端口号为 **5025**，SCPI 命令字符串的结尾需要加上“\n”（换行符）。

环境：windows 7 32位系统, Python v2.7.5

示例内容：通过 Socket 访问控制设备，发送命令，读取返回值。

以下为脚本内容：

```
#!/usr/bin/env python
#-*- coding:utf-8 –*-
#-------------------------------------------------------------------------------
#通过 Socket 访问控制设备，发送命令，读取并打印返回值。
#-------------------------------------------------------------------------------
import socket     # for sockets
import sys   # for exit
import time # for sleep
#-------------------------------------------------------------------------------
remote_ip = "10.11.13.32"   # 设备对应的 IP 地址
port = 5025   # 设备的端口号
count = 0

def SocketConnect():
    try:
        #创建 AF_INET, STREAM socket (TCP)
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    except socket.error:
        print ('Failed to create socket.')
        sys.exit();
    try:
        #连接设备
        s.connect((remote_ip , port))
    except socket.error:
        print ('failed to connect to ip ' + remote_ip)
    return s

def SocketQuery(Sock, cmd):
    try :
```

```
            #发送命令字符串
            Sock.sendall(cmd)
            time.sleep(1)
        except socket.error:
            #发送失败
            print ('Send failed')
            sys.exit()
        reply = Sock.recv(4096)
        return reply

def SocketClose(Sock):
        #关闭 socket
        Sock.close()
        time.sleep(.300)

def main():
        global remote_ip
        global port
        global count

        # 主函数: 发送命令“*IDN?”10 次,并读取返回值
        s = SocketConnect()
        for i in range(10):
            qStr = SocketQuery(s, b'*IDN?\n')
            print (str(count) + ":: " + str(qStr))
            count = count + 1
        SocketClose(s)
        input('Press "Enter" to exit')

if __name__ == '__main__':
        proc = main()
```

第四章 常见故障处理

本仪器在使用过程中可能出现如下故障，请首先按照下述方法处理，如果故障依然存在，请与**SIGLENT**联系

1. 仪器无法开机。

(1) 检查电源线是否已正确连接。

(2) 检查前面板电源开关键是否打开。

(3) 拔掉电源线，检查电压选择器（AC SELECTOR）是否处在正确的档位，保险丝的规格是否正确及是否完好无损。如需更换保险丝，请参考“ 更换保险丝 ”。

(4) 如果故障仍然存在，请与**SIGLENT**联系。

2. 恒压输出不正常。

(1) 检查所设置的输出功率是否满足负载要求。若满足，请进行下一步。

(2) 连接负载与电源的线缆是否有短路现象，是否接触良好。

(3) 查看负载是否出现问题。

(4) 查看电流设置值是否合适，如果过低，可以适当加大电流设置值。

(5) 若问题仍无法解决，请与**SIGLENT**联系。

3. 恒流输出不正常。

(1) 检查设置的输出功率是否满足负载要求。若满足，请进行下一步。

(2) 连接负载与电源的线缆是否有断路现象，是否接触良好。

(3) 查看负载是否出现问题。

(4) 查看电压设置值是否合适，如果过低，可以适当加大电压设置值。

(5) 若问题仍无法解决，请与**SIGLENT**联系。

第五章 服务和支持

5.1 保修概要

深圳市鼎阳科技有限公司保证所生产和销售的产品，从授权经销商发货之日起三年内，不会出现材料和工艺缺陷。如产品在保修期限内确有缺陷，SIGLENT将根据保修单的详细规定，提供修理或更换服务。

若需要服务或索取保修单的完整副本，请与最近的SIGLENT销售和服务办事处联系。除此概要或适用的保修单中所提供的保修之外，SIGLENT不作其它任何明示或暗示的保修保证，包括但不限于对适销性和特殊适用性的暗含保修。SIGLENT对间接的、特殊的或由此产生的损坏不承担任何责任。

5.2 联系我们

深圳市鼎阳科技股份有限公司

地址：深圳市宝安区68区留仙三路安通达工业园4栋3楼

服务热线：400-878-0807

E-mail：market@siglent.com

http://www.siglent.com