UNI-T

编程手册

UDP6730系列开关直流电源编程手册

2022-9-26

UNI-T TECHNOLOGIES， INC

保证和声明

版权

商标信息

UNI-T是优利德中国科技有限公司的注册商标。

文档编号

1.0.1

软件版本

00.00.01

软件升级可能更改或增加产品功能，请关注UNI-T网站获取最新版本手册或联系UNI-T升级软件。

声明

产品认证

UNI-T认证本产品符合中国国家产品标准和行业产品标准及ISO9001：2008标准和ISO14001：2004标准， 并进一步认证本产品符合其它国际标准组织成员的相关标准。

联系我们

如您在使用此产品或本手册的过程中有任何问题或需求，可与UNI-T联系：

UNI-T技术支持热线： 400-876-7822

网址：http://www.uni-trend.com

SCPI 指令简介

SCPI（Standard Commands for Programmable Instruments，即可编程仪器标准命令集）是一种建立在现有标准IEEE 488.1和IEEE 488.2基础上，并遵循了IEEE754标准中浮点运算规则、ISO646信息交换7位编码符号（相当于ASCII编程）等多种标准的标准化仪器编程语言。本节简介SCPI命令的格式、符号、参数和缩写规则。

指令格式

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。**命令行通常以冒号“：”开始；关键字之间用冒号“：”分隔，关键字后面跟随可选的参数设置。命令关键字和第一个参数之间以空格分开。命令字符串必须以一个 <换行> (<NL>) 字符结尾。命令行后面添加问号“?”通常表示对此功能进行查询。**

符号说明

下面四种符号不是SCPI命令中的内容，不随命令发送，但是通常用于辅助说明命令中的参数。

- **大括号 { }**
  大括号中通常包含多个可选参数，发送命令时必须选择其中一个参数。
  如:DISPlay:GRID:MODE ｛FULL | GRID | CROSS | NONE}命令。

- **竖线 |**
  竖线用于分隔多个参数选项，发送命令时必须选择其中一个参数。
  如:DISPlay:GRID:MODE ｛FULL | GRID | CROSS | NONE}命令。

- **方括号 [ ]**
  方括号中的内容（命令关键字）是可省略的。如果省略参数，仪器将该参数设置为默认值。
  例如：对于:MEASure:NDUTy? [<source>]命令，[<source>]表示当前通道。

- **三角括号 < >**
  三角括号中的参数必须用一个有效值来替换。例如：以DISPlay:GRID:BRIGhtness 30的形式发送DISPlay:GRID:BRIGhtness <count>命令。

参数说明

本手册介绍的命令中所含的参数可以分为以下5种类型：布尔型、整型、实型、离散型、ASCII字符串。

- **布尔型**
  参数取值为“ON”（1）或“OFF”（0）。例如：:SYSTem:LOCK {{1 | ON} | {0 | OFF}}。

- **整型**
  除非另有说明，参数在有效值范围内可以取任意整数值。注意：此时，请不要设置参数为小数格式，否则将出现异常。例如：:DISPlay:GRID:BRIGhtness <count>命令中的参数< count >可取0到100范围内的任一整数。

- **实型**
  除非另有说明，参数在有效值范围内可以取任意值。
  例如：对于CH1，CHANnel1:OFFSet <offset>命令中的参数<offset>的取值为实型。

- **离散型**
  参数只能取指定的几个数值或字符。例如：:DISPlay:GRID:MODE { FULL | GRID | CROSS | NONE}命令的参数只能为FULL、GRID、CROSS、NONE。

- **ASCII 字符串**
  字符串参数实际上可包含所有 ASCII 字符集。字符串必须以配对的引号开始和结尾；可以用单引号或双引号。引号分隔符也可以作为字符串的一部分，只需键入两次并且不在中间添加任何字符，例如设置IP：SYST:COMM:LAN:IPAD "192.168.1.10"。

简写规则

所有命令对大小写都能识别，可以全部采用大写或小写。如果要缩写，必须输完命令格式中的所有大写字母。

数据返回

数据返回分为单个数据和批量数据返回，单个数据返回相对应的参数类型，其中实型返回用科学计数法表示，e 前部分小数点后面保留三位数据，e 部分保留三位数据；批量数据返回必须符合 IEEE 488.2 #格式的字符串数据，其格式：‘#’ + 长度所占的字符位数[固定为一个字符] + 有效数据长度的 ASCII 值 + 有效数据 + 结束符[‘\n’]，例如#3123xxxxxxxxxxxxxxxxxxx\n 表示的具有 123 个字节有效批量数据返回格式，其中‘3’表示“123”占 3 个字符位

SCPI 指令详解

IEEE488.2 通用命令

*IDN?

- **命令格式：**
  *IDN?
- **功能描述：**
  用于查询制造商名称、产品型号、产品序列号和软件版本号。
- **返回格式：**
  制造商名称，产品型号，产品序列号，由点号分隔的软件版本号。
  注意：返回的型号要与铭牌信息一致。
- **举例：**
  UNI-T Technologies, UTS3036B, 000000001, 00.00.01

MEASure 命令

- **命令格式：**
  MEASure:VOLTage?
  MEASure:CURRent?
  MEASure:POWEr?
  MEASure:ALL?
- **功能描述：**
  用于查询产品回读电压\电流\功率值。

MEASure 命令

- **命令格式：**
  MEASure:VOLTage?
  MEASure:CURRent?
  MEASure:POWEr?
  MEASure:ALL?
- **功能描述：**
  用于查询产品回读电压\电流\功率值。

SOURce 命令

[SOURce:]VOLTage

- **命令格式：**
  [SOURce:]VOLTage <value>
- **功能描述：**
  设置输出电压值，如果已打开保护功能且设置值大于保护值时，设置无效。

[SOURce:]CURRent

- **命令格式：**
  [SOURce:]CURRent <value>
- **功能描述：**
  设置输出电流值，如果已打开保护功能且设置值大于保护值时，设置无效。

[SOURce:]VOLTage:PROTection

- **命令格式：**
  [SOURce:]VOLTage:PROTection <value>
- **功能描述：**
  设置过压保护功能的保护值。

[SOURce:]CURRent:PROTection

- **命令格式：**
  [SOURce:]CURRent:PROTection <value>
- **功能描述：**
  设置过流保护功能的保护值。

[SOURce:]VOLTage:PROTection:STATe

- **命令格式：**
  [SOURce:]VOLTage:PROTection:STATe {ON|OFF}
- **功能描述：**
  打开或关闭过压保护功能。

[SOURce:]CURRent:PROTection:STATe

- **命令格式：**
  [SOURce:]CURRent:PROTection:STATe {ON|OFF}
- **功能描述：**
  打开或关闭过流保护功能。

[SOURce:]VOLTage

- **命令格式：**
  [SOURce:]VOLTage?
- **功能描述：**
  查询输出电压值。

[SOURce:]CURRent

- **命令格式：**
  [SOURce:]CURRent?
- **功能描述：**
  查询输出电流值。

[SOURce:]VOLTage:PROTection

- **命令格式：**
  [SOURce:]VOLTage:PROTection?
- **功能描述：**
  查询输出电压值。

[SOURce:]CURRent:PROTection

- **命令格式：**
  [SOURce:]CURRent:PROTection?
- **功能描述：**
  查询输出电流值。

[SOURce:]VOLTage:PROTection:STATe

- **命令格式：**
  [SOURce:]VOLTage:PROTection:STATe?
- **功能描述：**
  查询过流保护功能的开关状态。返回“ON”或“OFF”。

[SOURce:]CURRent:PROTection:STATe

- **命令格式：**
  [SOURce:]CURRent:PROTection:STATe?
- **功能描述：**
  查询过流保护功能的开关状态。返回“ON”或“OFF”。

OUTPut 命令

OUTPut

- **命令格式：**
  OUTPut {ON|OFF}
- **功能描述：**
  打开或关闭输出。

- **命令格式：**
  OUTPut?
- **功能描述：**
  查询输出开关状态。返回“ON”或“OFF”。

OUTPut:CVCC?

- **命令格式：**
  OUTPut:CVCC?
- **功能描述：**
  查询输出恒压恒流状态。返回“CV”或“CC”。

MEMoy 命令

MEMory:STORe

- **命令格式：**
  MEMory:STORe{M1|M2|M3}
- **功能描述：**
  把当前的输出设置值存于相应位置。

MEMory:LOAD

- **命令格式：**
  MEMory:LOAD{M1|M2|M3}
- **功能描述：**
  从相应位置加载输出设置值并应用到当前输出值。当保护功能打开并且加载值大于保护值时，输出值被设置为保护值。

MEMory:GROUp

- **命令格式：**
  MEMory:GROUp <n>
- **功能描述：**
  设置要使用的存储组。$1 \leq n \leq 200$。

MEMory:GROUp

- **命令格式：**
  MEMory:GROUp ?
- **功能描述：**
  查询正在使用的存储组。返回 1——200 的整数。

SYSTem 命令

SYSTem:REMote

- **命令格式：**
  SYSTem:REMote
- **功能描述：**
  使设备进入远程模式。

注：远程模式时设备的面板按键功能被屏蔽（除了 OUTPUT 键可以控制输出外），但是用户仍可以通过面板的“ESC”键退出远程模式。

- **命令格式：**
  SYSTem:REMote ?
- **功能描述：**
  查询设备是否处于远程模式。返回“YES”或“NO”。

编程说明

描述在编程操作过程中可能出现的一些问题及解决方法。当您遇到如下这些问题时，请按照相应的说明进行处理。

编程准备

编程准备工作仅适用于在 Windows 操作系统下使用 Visual Studio 和 LabVIEW 开发工具进行编程。

首先确认您的电脑上是否已经安装 NI 的 VISA 库（可到 https://www.ni.com/en-ca/support/downloads/drivers/download.ni-visa.html 下载），本文中默认安装路径为 C:\Program Files\IVI Foundation\VISA。

通过仪器设备的 USB 或 LAN 接口与 PC 建立通信，请使用 USB 数据线将仪器设备后面板的 USB DEVICE 接口与 PC 的 USB 接口相连，或者使用 LAN 数据线将仪器设备后面板的 LAN 口与 PC 的 LAN 接口相连。

VISA 编程示例

本节给出了一些编程示例。通过这些例子，可以了解如何使用 VISA，并结合编程手册的命令实现对仪器设备的控制。通过下面的例子，你可以开发更多应用。

VC++示例

- 环境：Window 系统, Visual Studio。
- 描述：通过 USBTMC 和 TCP/IP 访问仪器设备，并在 NI-VISA 上发送"*IDN?"命令来查询设备信息。
- 步骤：

1. 打开 Visual Studio 软件，新建一个 VC++ win32 console project。
2. 设置调用 NI-VISA 库的项目环境，分别为静态库和动态库。

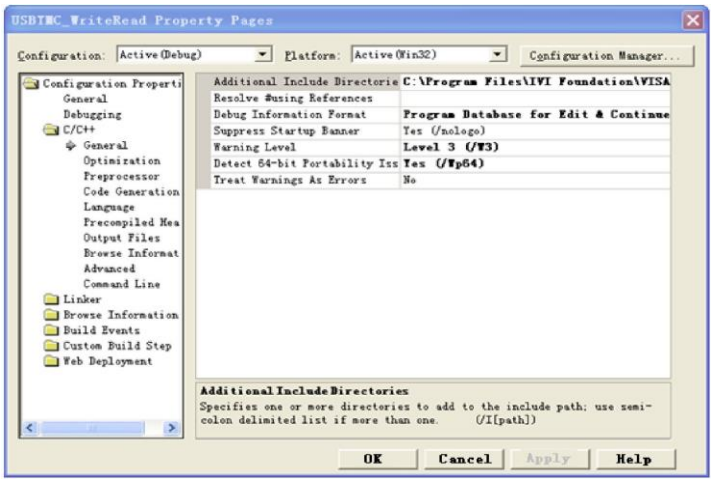
a) 静态库：

在 NI-VISA 安装路径找:visa.h、visatype.h、visa32.lib 文件，将它们复制到 VC++项目的根路径下并添加到项目中。在 projectname.cpp 文件上添加下列两行代码：

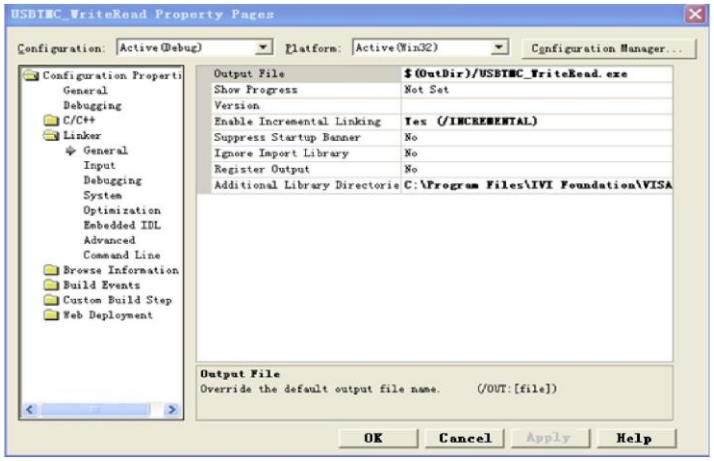
```
#include "visa.h"
#pragma comment(lib,"visa32.lib")
```

b) 动态库：

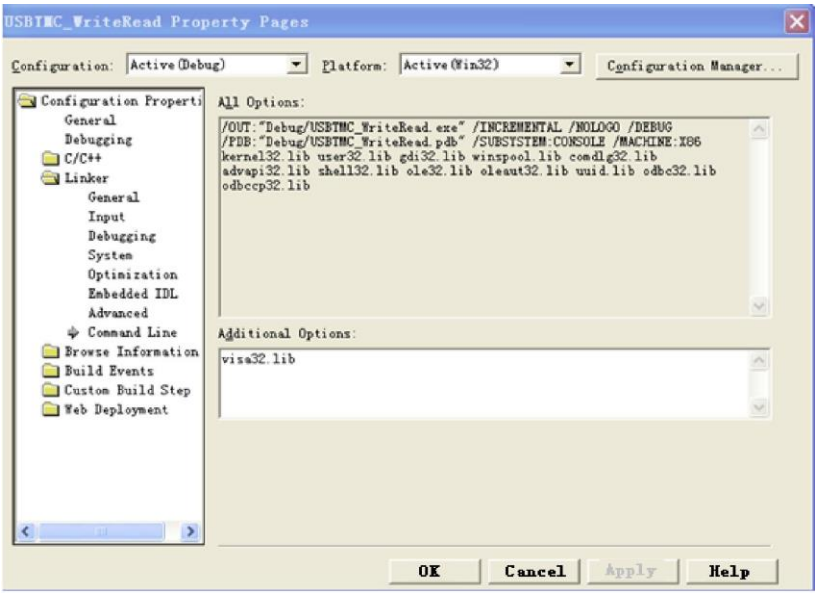
点击"project>>properties"，在属性对话框左侧选择"c/c++---General"中，将 "Additional Include Directories"项的值设置为 NI-VISA 的安装路径，(例如：C:\ProgramFiles\IVI Foundation\VISA\WinNT\include),如下图所示:



在属性对话框左侧选择"Linker-General",并将"Additional Library Directories"项的值设置为 NI-VISA 的安装路径，(例如：C:\Program Files\IVI Foundation\VISA\WinNT\include), 如下图所示：



在属性对话框左侧选择"Linker-Command Line",将"Additional"项的值设置为 visa32.lib，如下图所示：



在 projectname.cpp 文件上添加 visa.h 文件:

```
#include <visa.h>
```

1. 源码：

a) USBTMC 示例

```
int usbtmc_test()
{ /** This code demonstrates sending synchronous read & write commands
   * to an USB Test & Measurement Class (USBTMC) instrument using NI-VISA
    * The example writes the "*IDN?\n" string to all the USBTMC
    * devices connected to the system and attempts to read back
    * results using the write and read functions.
    * Open Resource Manager
    * Open VISA Session to an Instrument
    * Write the Identification Query Using viPrintf
    * Try to Read a Response With viScanf
    * Close the VISA Session*/
    ViSession defaultRM;
    ViSession instr;
    ViUInt32 numInstrs;
    ViFindList findList;
    ViStatus status;
```

```
char instrResourceString[VI_FIND_BUFLEN];
unsigned char buffer[100];
int i;
status = viOpenDefaultRM(&defaultRM);
if (status < VI_SUCCESS)
{
        printf("Could not open a session to the VISA Resource Manager!\n");
        return status;
}
 /*Find all the USB TMC VISA resources in our system and store the number of resources in
 the system in numInstrs.*/
status = viFindRsrc(defaultRM, "USB?*INSTR", &findList, &numInstrs, instrResourceString);
if (status<VI_SUCCESS)
{
        printf("An error occurred while finding resources. \nPress Enter to continue.");
        fflush(stdin);
        getchar();
        viClose(defaultRM);
        return status;
}
/** Now we will open VISA sessions to all USB TMC instruments.
*       We must use the handle from viOpenDefaultRM and we must
*       also use a string that indicates which instrument to open. This
*       is called the instrument descriptor. The format for this string
*       can be found in the function panel by right clicking on the
*       descriptor parameter. After opening a session to the
*       device, we will get a handle to the instrument which we
*       will use in later VISA functions. The AccessMode and Timeout
*       parameters in this function are reserved for future
*       functionality. These two parameters are given the value 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;
        }
```

```
                /** At this point we now have a session open to the USB TMC instrument.
                *We will now use the viPrintf function to send the device the string "*IDN?\n",
                *asking for the device's identification. */
                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;
                }
                /** Now we will attempt to read back a response from the device to
                *the identification query that was sent. We will use the viScanf
                *function to acquire the data.
                *After the data has been read the response is displayed. */
                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, buffer);
                }
                status = viClose(instr);
        }
        /*Now we will close the session to the instrument using viClose. This operation frees all
          system resources.*/
        status = viClose(defaultRM);
        printf("Press Enter to exit.");
        fflush(stdin);
        getchar();
        return 0;
}

int _tmain(int argc, _TCHAR* argv[])
{
        usbtmc_test();
        return 0;
}
```

b) TCP/IP 示例

```
int tcp_ip_test(char *pIP)
{
    char outputBuffer[VI_FIND_BUFLEN];
    ViSession defaultRM, instr;
    ViStatus status;
    /* First we will need to open the default resource manager. */
    status = viOpenDefaultRM(&defaultRM);
    if (status < VI_SUCCESS)
    {
            printf("Could not open a session to the VISA Resource Manager!\n");
    }
    /* Now we will open a session via TCP/IP device */
    char head[256] = "TCPIP0::";
    char tail[] = "::inst0::INSTR";
    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("\nMesseage read from device: %*s\n", 0, outputBuffer);
    }
    status = viClose(instr);
    status = viClose(defaultRM);
    printf("Press Enter to exit.");
    fflush(stdin);
    getchar();
    return 0;
}
```

```
int _tmain(int argc, _TCHAR* argv[])
{
    printf("Please input IP address:");
    char ip[256];
    fflush(stdin);
    gets(ip);
    tcp_ip_test(ip);
    return 0;
```

C#示例

- 环境：Window 系统，Visual Studio。
- 描述：通过 USBTMC 和 TCP/IP 访问仪器设备，并在 NI-VISA 上发送"*IDN?"命令来查询设备信息。
- 步骤：

1. 打开 Visual Studio 软件，新建一个 C# console project。
2. 添加 VISA 的 C#引用 Ivi.Visa.dll 和 NationalInstruments.Visa.dll。
3. 源码：

a) USBTMC 示例

```
class Program
{
    void usbtmc_test()
    {
        using (var rmSession = new ResourceManager())
        {
            var resources = rmSession.Find("USB?*INSTR");
            foreach (string s in resources)
            {
                try
                {
                    var mbSession = (MessageBasedSession)rmSession.Open(s);
                    mbSession.RawIO.Write("*IDN?\n");
                    System.Console.WriteLine(mbSession.RawIO.ReadString());
                }
                catch (Exception ex)
                {
                    System.Console.WriteLine(ex.Message);
                }
            }
        }
    }

    void Main(string[] args)
    {
        usbtmc_test();
    }
}
```

b) TCP/IP 示例

```
class Program
{
```

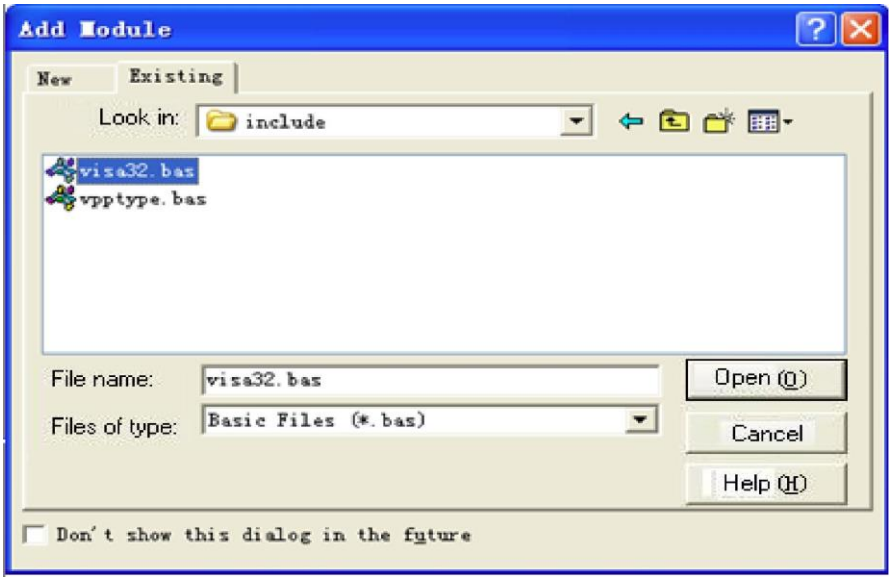
```
    void tcp_ip_test(string ip)
    {
        using (var rmSession = new ResourceManager())
        {
            try
            {
                var resource = string.Format("TCPIP0::{0}::inst0::INSTR", ip);
                var mbSession = (MessageBasedSession)rmSession.Open(resource);
                mbSession.RawIO.Write("*IDN?\n");
                System.Console.WriteLine(mbSession.RawIO.ReadString());
            }
            catch (Exception ex)
            {
                System.Console.WriteLine(ex.Message);
            }


    void Main(string[] args)
    {
        tcp_ip_test("192.168.20.11");
    }
}
```

VB 示例

- 环境：Window 系统，Microsoft Visual Basic 6.0。
- 描述：通过 USBTMC 和 TCP/IP 访问仪器设备，并在 NI-VISA 上发送"*IDN?"命令来查询设备信息。
- 步骤：

1. 打开 Visual Basic 软件，并新建一个标准的应用程序项目。
2. 设置调用 NI-VISA 库项目环境：点击 Existing tab of Project>>Add Existing Item，在 NI-VISA 安装路径下的"include"文件夹中查找 visa32.bas 文件并添加该文件。如下图所示：



3. 源码：

a) USBTMC 示例

```
PrivateFunction usbtmc_test() AsLong
' This code demonstrates sending synchronous read & write commands
' to an USB Test & Measurement Class (USBTMC) instrument using NI-VISA
' The example writes the "*IDN?\n" string to all the USBTMC
' devices connected to the system and attempts to read back
' results using the write and read functions.
' The general flow of the code is
' Open Resource Manager
' Open VISA Session to an Instrument
' Write the Identification Query Using viWrite
' Try to Read a Response With viRead
' Close the VISA Session

Const MAX_CNT = 200
Dim defaultRM AsLong
Dim instrsesn AsLong
```

```
Dim numInstrs AsLong
Dim findList AsLong
Dim retCount AsLong
Dim status AsLong
Dim instrResourceString AsString *VI_FIND_BUFLEN
Dim Buffer AsString * MAX_CNT
Dim i AsInteger

' First we must call viOpenDefaultRM to get the manager
' handle. We will store this handle in defaultRM.
status = viOpenDefaultRM(defaultRM)
If(status < VI_SUCCESS) Then
    resultTxt.Text = "Could not open a session to the VISA Resource Manager!"
    usbtmc_test = status
ExitFunction
EndIf

' Find all the USB TMC VISA resources in our system and store the
' number of resources in the system in numInstrs.
status = viFindRsrc(defaultRM, "USB?*INSTR", findList, numInstrs, instrResourceString)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "An error occurred while finding resources."
    viClose(defaultRM)
    usbtmc_test = status
ExitFunction
EndIf

' Now we will open VISA sessions to all USB TMC instruments.
' We must use the handle from viOpenDefaultRM and we must
' also use a string that indicates which instrument to open.  This
' is called the instrument descriptor.  The format for this string
' can be found in the function panel by right clicking on the
' descriptor parameter. After opening a session to the
' device, we will get a handle to the instrument which we
' will use in later VISA functions.  The AccessMode and Timeout
' parameters in this function are reserved for future
' functionality.  These two parameters are given the value VI_NULL.
For i = 0 To numInstrs
If (i > 0) Then
    status = viFindNext(findList, instrResourceString)
EndIf
    status = viOpen(defaultRM, instrResourceString, VI_NULL, VI_NULL, instrsesn)
```

```
If (status < VI_SUCCESS) Then
    resultTxt.Text = "Cannot open a session to the device " + CStr(i + 1)
GoTo NextFind
EndIf
' At this point we now have a session open to the USB TMC instrument.
' We will now use the viWrite function to send the device the string "*IDN?",
' asking for the device's identification.
status = viWrite(instrsesn, "*IDN?", 5, retCount)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "Error writing to the device."
    status = viClose(instrsesn)
GoTo NextFind
EndIf

' Now we will attempt to read back a response from the device to
' the identification query that was sent.  We will use the viRead
' function to acquire the data.
' After the data has been read the response is displayed.
status = viRead(instrsesn, Buffer, MAX_CNT, retCount)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "Error reading a response from the device." + CStr(i + 1)
Else
    resultTxt.Text = "Read from device: " + CStr(i + 1) + " " + Buffer
EndIf
    status = viClose(instrsesn)
Next i

' Now we will close the session to the instrument using
' viClose. This operation frees all system resources.
status = viClose(defaultRM)
usbtmc_test = 0
EndFunction
```

b) TCP/IP 示例

```
PrivateFunction tcp_ip_test(ByVal ip AsString) AsLong
Dim outputBuffer AsString * VI_FIND_BUFLEN
Dim defaultRM AsLong
Dim instrsesn AsLong
Dim status AsLong
Dim count AsLong

' First we will need to open the default resource manager.
```

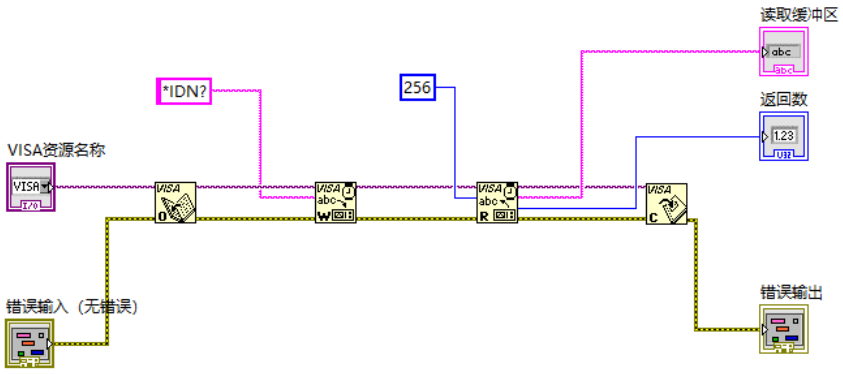
```
status = viOpenDefaultRM(defaultRM)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "Could not open a session to the VISA Resource Manager!"
    tcp_ip_test = status
ExitFunction
EndIf

' Now we will open a session via TCP/IP device
status = viOpen(defaultRM, "TCPIP0::" + ip + "::inst0::INSTR", VI_LOAD_CONFIG, VI_NULL,
instrsesn)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "An error occurred opening the session"
    viClose(defaultRM)
    tcp_ip_test = status
ExitFunction
EndIf
status = viWrite(instrsesn, "*IDN?", 5, count)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "Error writing to the device."
EndIf
    status = viRead(instrsesn, outputBuffer, VI_FIND_BUFLEN, count)
If (status < VI_SUCCESS) Then
    resultTxt.Text = "Error reading a response from the device." + CStr(i + 1)
Else
    resultTxt.Text = "read from device:" + outputBuffer
EndIf
    status = viClose(instrsesn)
    status = viClose(defaultRM)
    tcp_ip_test = 0
EndFunction
```

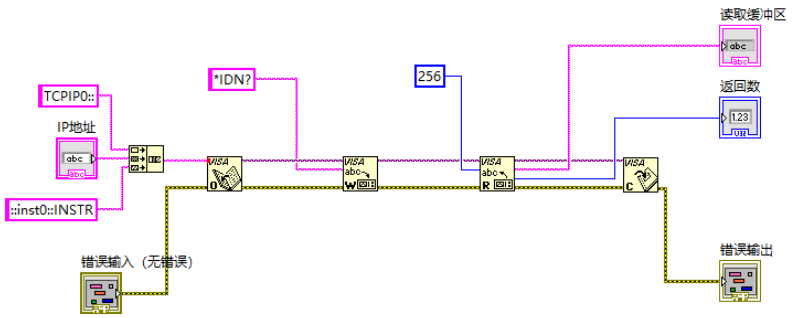
LabVIEW 示例

- 环境：Window 系统，LabVIEW。
- 描述：通过 USBTMC 和 TCP/IP 访问仪器设备，并在 NI-VISA 上发送"*IDN?"命令来查询设备信息。
- 步骤：

1. 打开 LabVIEW 软件，并创建一个 VI 文件。
2. 添加控件，右击前面板界面，从控制列中选择并添加 VISA 资源名、错误输入、错误输出以及部分的指示符。
3. 打开框图界面，右击 VISA 资源名称，并在弹出菜单的 VISA 面板中选择和添加下列功能：VISA Write、VISA Read、VISA Open 和 VISA Close。
4. VI 打开了一个 USBTMC 设备的 VISA 会话，并向设备写*IDN?命令并回读的响应值。当所有通信完成时，VI 将关闭 VISA 会话，如下图所示：



5. 通过 TCP/IP 与设备通信类似于 USBTMC,但是你需要将 VISA 写函数和 VISA 读函数设置为同步 I/O,LabVIEW 默认设置为异步 IO。右键单击节点，然后从快捷菜单中选择,"Synchronous I/O Mode>>Synchronous"以实现同步写入或读取数据，如下图所示：



MATLAB 示例

- 环境：Window 系统，MATLAB。
- 描述：通过 USBTMC 和 TCP/IP 访问仪器设备，并在 NI-VISA 上发送"*IDN?"命令来查询设备信息。
- 步骤：

1. 打开 MATLAB 软件，点击在 Matlab 界面的 File>>New>>Script 创建一个空的 M 文件。
2. 源码：

a) USBTMC 示例

```
function usbtmc_test()
% This code demonstrates sending synchronous read & write commands
% to an USB Test & Measurement Class (USBTMC) instrument using
% NI-VISA

%Create a VISA-USB object connected to a USB instrument
vu = visa('ni','USB0::0x5345::0x1234::SN20220718::INSTR');

%Open the VISA object created
fopen(vu);

%Send the string "*IDN?",asking for the device's identification.
fprintf(vu,'*IDN?');

%Request the data

outputbuffer = fscanf(vu);
disp(outputbuffer);

%Close the VISA object
fclose(vu);
delete(vu);
clear vu;

End
```

b) TCP/IP 示例

```
function tcp_ip_test()
% This code demonstrates sending synchronous read & write commands
% to an TCP/IP instrument using NI-VISA
%Create a VISA-TCPIP object connected to an instrument
%configured with IP address.
vt = visa('ni',['TCPIP0::','192.168.20.11','::inst0::INSTR']);
```

```
%Open the VISA object created

fopen(vt);

%Send the string "*IDN?",asking for the device's identification.
fprintf(vt,'*IDN?');

%Request the data
outputbuffer = fscanf(vt);
disp(outputbuffer);

%Close the VISA object
fclose(vt);
delete(vt);
clear vt;

end
```

Python 示例

- 环境：Window 系统，Python3.8，PyVISA 1.11.0。
- 描述：通过 USBTMC 和 TCP/IP 访问仪器设备，并在 NI-VISA 上发送"*IDN?"命令来查询设备信息。
- 步骤：

1. 首先安装 python，然后打开 Python 脚本编译软件，创建一个空的 test.py 文件。
2. 使用 pip install PyVISA 指令安装 PyVISA，如无法安装，请参考此链接使用说明（https://pyvisa.readthedocs.io/en/latest/）
3. 源码：

a) USBTMC 示例

```
import pyvisa
rm = pyvisa.ResourceManager()
rm.list_resources()
my_instrument = rm.open_resource('USB0::0x5345::0x1234::SN20220718::INSTR')
print(my_instrument.query('*IDN?'))
```

b) TCP/TP 示例

```
import pyvisa
rm = pyvisa.ResourceManager()
rm.list_resources()
my_instrument = rm.open_resource('TCPIP0::192.168.20.11::inst0::INSTR')
print(my_instrument.query('*IDN?'))
```