RIGOL

Programming Guide

DP1116A Programmable DC Power Supply

Mar. 2010

RIGOL Technologies, Inc.

Copyright

Trademark Information

RIGOL is a registered trademark of **RIGOL** Technologies, Inc.

Notices

RIGOL products are protected by patent law in and outside of P.R.C.

RIGOL Technologies, Inc. reserves the right to modify or change parts of or all the specifications and pricing policies at company's sole decision.

Information in this publication replaces all previously corresponding material.

RIGOL shall not be liable for losses caused by either incidental or consequential in connection with the furnishing, use or performance of this manual as well as any information contained.

Any part of this document is forbidden to copy or photocopy or rearrange without prior written approval of **RIGOL**.

Product Certification

RIGOL guarantees this product conforms to the standards of national and industrial. Meanwhile, the related standards conform to other ISO will get further. At present, DP1116A has passed CE, cTUVus and LXI certification.

Structure of this Document

Chapter 1 Programming Overview

This chapter introduces you how to use SCPI commands to control the DP1116A via remote interfaces.

Chapter 2 Command System

This chapter gives detailed information on each command supported by DP1116A, including the syntax, function description and considerations when sending a command as well as some application examples.

Chapter 3 Programming Examples

This chapter provides some programming examples about common features in Visual C++ 6.0, Visual Basic 6.0 and LabVIEW 8.6 development environments.

Command Quick Reference A-Z

The Appendix lists all of the commands alphabetically in favor of quick reference.

NOTE:
For the newest edition of this manual please go to http://www.rigol.com.

Contents

Chapter 1 Programming Overview

This chapter introduces you how to use SCPI commands to control the DP1116A via remote interfaces. The chapter contains the following topics:

- Communication Interfaces
- SCPI Commands Introduction

  Syntax

  Symbol Description

  Parameter Type

  Command Abbreviation

Communication Interfaces

DP1116A provides three kinds of interfaces for connecting with a computer: USB, GPIB and LAN. For the specific application of each interface, please refer to the User's Guide of this product.

All of the command words are transmitted and identified as an ASCII string when you are programming using commands so that users can control the instrument and carry out secondary development.

By programming the DP1116A, you can:

- Specify the instrument parameters
- Enable the output of the instrument

SCPI Commands Introduction

SCPI (Standard Commands for Programmable Instrument) is based on IEEE 488.2 and usually divided into two sections: Common Commands and Control Commands defined for SCPI Instruments.

A common command is defined by IEEE 488.2 and must be executed and both the syntax and semanteme of which follow the application of IEEE 488.2. The common commands work independently of measurement and are generally used for controlling the reset, self-test and status operations. For more detalis, refer to "**Common Commands**".

A Control Command defined for SCPI Instrument is used to measure and read the data, control the status of a function or a mode and so on, involving all measurement functions and some specific functional functions.

Syntax

The SCPI commands is to be seen as a tree originating at the root keyword and dispersing into different branches depending on the function required. Each sub-system contains a root keyword and one or more sub-keywords. ":" is often used to separate keywords; parameters are permitted to follow a keyword; "?" appeared after a command line denotes to query; "space" is used to separate a command from the parameter followed.

For example:
OUTPut:TIMEr:CIRcle {<value>|INFInite}
OUTPut:TIMEr:CIRcle?

From the commands above we can see that: **OUTPut** is the root keyword; **TIMEr** and **TYPE** are the second and third keywords, respectively; "**:**" is used to separate different keywords; the contents enclosed in the "{ }" denotes the parameters and <value> denotes the assignable parameter; the keyword **OUTPut:TIMEr:CIRcle** and parameter **{<value>|INFInite}** are separated by a space.
OUTPut:TIMEr:CIRcle? denotes to query.

Besides, "," is generally used for separating different parameters that contained in the same command, such as:
OUTPut:TIMEr <secnum>,<volt>,<curr>,<time>

Symbol Description

The following symbols are not the contents of the SCPI commands but usually used in instances of the parameters from an Explanation command.

1. **Braces { }**
   The parameters enclosed in braces are optional, and you should choose at least one from them, such as:
   OUTPut:TIMEr:CIRcle {<value>|INFInite}
   In the command above, <value> and INFInite are paremeters and <value> can be defined.

2. **Vertical Bar |**
   The vartical bar separates two or more optional parameters. When you send a command, at least one of the parameters should be selected, such as
   OUTPut:OVP:STATe {ON|OFF}
   In the command above, "OFF" or "ON" should be used as its parameter.

3. **Square Brackets [ ]**
   The contents enclosed in square brackets are optional and would be executed regardless of whether they were omitted such as:
   SYSTem:BEEPer[:IMMEdiate] {ON|OFF}
   In the command above, users can omit the pararmeter [:IMMEdiate].

4. **Triangle Brackets < >**
   The parameter enclosed in triangle brackets must be an effective value such as:
   SYSTem:BRIGht <brightness>
   In the command above, the paremeter <brightness> should be an effective value such as: SYSTem:BRIGht 5.

Parameter Type

The commands presented in this manual contains 6 kinds of parameters and different parameters have different setting methods.

1. MINimum and MAXimum

A MINimum or MAXimum can be used in a command to replace some parameters, such as:

CURRent {<current>|MINimum|MAXimum}

In the command above, users can set a specific current value by <current> or directly set the current to "MINimum" or "MAXimum".

2. Boolean

The parameter should be "OFF" or "ON", such as:

DISPlay:CLASsical {ON|OFF}

In the command above, "ON" denotes to enable the classic display mode and "OFF" denotes to disable.

3. Consecutive Integer

The parameter can be any integer within the valid range, such as:

SYSTem:BRIGht <brightness>

In the command above, <brightness> can be any integer within 1 and 8.

4. Consecutive Real Number

The parameter can be any value within the valid range and under the precision requirement, such as

CURRent {<current>|MINimum|MAXimum}

The command above is used to set the current of the present channel and <current> can be any real number within the specified current range.

5. Discrete

The parameter should be an option listed in a command, such as

RECAll:LOCal {1|2|3|4}

In the command above, the parameter could only be 1, 2, 3 or 4.

6. ASCII String

The parameter should be a composition of ASCII characters, such as

STORe:LOCAl {1|2|3|4},<name>

In the command above, <name> should be composed of the ASCII characters.

Command Abbreviation

All of the commands in the DP1116A are case-insensitive, you can use any kind of them. But if you use an abbreviation, the capital letters specified in a command must be written completely. For example:

DISPlay:CLASsical?
also could be:
DISP:CLAS? or
disp:clas?

Chapter 2 Command System

This chapter gives detailed information on each command supported by DP1116A, including the syntax, function description and considerations when sending a command as well as some application examples.

General condensed summary:

1. Parameter precision
The precisions of the voltage and current that you specify are 0.001 V and 0.001 A, respectively.

2. Return precision
The precisions of the returned voltage and current are 0.001 V and 0.001 A, respectively.

3. Parameter input
The system only allows for numerical value entry and will automatically add a default unit for the entered value. For example, after you enter a voltage, or a current, or a time value, the system will automatically use "V", or "A", or "S" as its unit.

The DP1116A contains following subcommands:

- Common Commands
- APPLy Command
- MEASure Commands
- OUTPut Commands
- SOURce Commands
- DISPlay Command
- SYSTem Commands
- STORe Commands
- RECAll Commands

Common Commands

IEEE 488.2 standard defines a common command set for querying or executing some basic operations. The command in this set usually begins with a "*" and holds a keyword that is 3 characters long.

DP1116A supports the following **IEEE488.2** commands:

1. *IDN?
2. *RST
3. *TST?
4. *SAV
5. *RCL

The detailed information of each **IEEE488.2** command are:

1. *IDN?	
Syntax	*IDN?
Function	Queries the instrument ID and returns a string that separated by three ",", including the manufactory, model, serial number and the edition number that consists of digits and separated by "."
Return	Rigol Technologies, DP1116A, DP1A666666666, 00.01.00.00.01.00.00.01.00.00
2. *RST	
Syntax	*RST
Function	Restores the instrument to the defaults.
3. *TST?	
Syntax	*TST?
Function	Queries the result of Self-test.
Return	The query returns "Pass" or "Error". *
*Remark: Double quotation marks would not be returned after a query unless where noted in this manual.	
4. *SAV	
Syntax	*SAV {1|2|3|4},<name>
Function	Saves the current system status into the nonvolatile memory with a specified name.

Explanation	The DP1116A provides four locations in the nonvolatile memeory for status storage (“1”, ”2”, ”3” and ”4”).
5. *RCL	
Syntax	*RCL {1|2|3|4}
Function	Recalls the saved intrument status.

APPLy Command

APPLy command is used for quickly setting the voltage and current values for the present scale, which will provide a direct approach to the remote control.

The DP1116A supports the following APPLy commands:

1. APPLy

The detailed information of the **APPLy** command are:

1. APPLy	
Syntax	APPLy {<volt>|DEFault|MINimum|MAXimum} [,{<curr>|DEFault|MINimum|MAXimum}] APPLy?
Function	Sets the voltage and current values for the present scale. The query returns the voltage and current values assigned to the present scale.
Explanation	● If you want to specify only one parameter, it could only be the voltage. ● The voltage and current values of the present scale will be changed as soon as you send this command. ● Different scales have different voltage and current setting ranges. In **16V/10A**, <volt> ranges from 0 to 16.8 V and <curr> ranges from 0 to 10.5 A. In **32V/5A**, <volt> ranges from 0 to 33.6 V and <curr> ranges from 0 to 5.25 A.
Example	● APPL 16 Sets the voltage to 16 V ● APPL 16,10 Sets the voltage and current to 16V and 10 A respectively ● APPL? Returns "16.000V,10.000A"

MEASure Commands

MEASure commands are used for querying the voltage, current or power value measured from the output terminal of the instrument.

The DP1116A supports the following MEASure commands:

1. MEASure:CURRent[:DC]?
2. MEAsure[:VOLTage][:DC]?
3. MEASure:POWEr[:DC]?

The detailed information of each **MEASure** command are:

1. MEASure:CURRent[:DC]?	
Syntax	MEASure:CURRent[:DC]?
Function	Queries the current measured from the output terminal of the present scale.
Return	Such as 6.000A
2. MEAsure[:VOLTage][:DC]?	
Syntax	MEAsure[:VOLTage][:DC]?
Function	Queries the voltage measured from the output terminal of the present scale.
Return	Such as 15.000V
3. MEASure:POWEr[:DC]?	
Syntax	MEASure:POWEr[:DC]?
Function	Queries the power measured from the output terminal of the present scale.
Return	Such as 90.000W

OUTPut Commands

OUTPut commands are used to configure and query the output settings of the instrument, including the Scale selection, O.C.P, O.V.P, Waveform display and Timer functions.

The DP1116A supports the following **OUTPut** commands:

1. OUTPut:RANGe
2. OUTPut[:STATe]
3. OUTPut:OVP:STATe
4. OUTPut:OVP
5. OUTPut:OCP:STATe
6. OUTPut:OCP
7. OUTPut:WAVE
8. OUTPut:TIMEr
9. OUTPut:TIMEr:STATe
10. OUTPut:TIMEr:SAVE
11. OUTPut:TIMEr:CIRcle

The detailed information of each **OUTPut** command are:

1. OUTPut:RANGe	
Syntax	OUTPut:RANGe {16V|32V} OUTPut:RANGe?
Function	Selects a desired scale. The query returns the present scale.
Example	OUTP:RANG 16V The query returns "16V/10A".
2. OUTPut[:STATe]	
Syntax	OUTPut[:STATe] {OFF|ON} OUTPut[:STATe]?
Function	Turns on or off the present scale. The query returns "ON" or "OFF".
Example	OUTP:STAT ON
3. OUTPut:OVP:STATe	

Syntax	OUTPut:OVP:STATe {ON|OFF} OUTPut:OVP:STATe?
Function	Enables or disables the O.V.P (Overvoltage Protection) function of the present scale. The query returns "ON" or "OFF".
Example	OUTP:OVP:STAT ON
4. OUTPut:OVP	
Syntax	OUTPut:OVP <value> OUTPut:OVP
Function	Sets the O.V.P value of the present scale. The query returns the O.V.P setting value of the present scale.
Explanation	<value> ranges from 0.1 V to 35.2 V.
Example	OUTP:OVP 25 The query returns "25.0"
5. OUTPut:OCP:STATe	
Syntax	OUTPut:OCP:STATe {ON|OFF} OUTPut:OCP:STATe?
Function	Enables or disables the O.C.P (Overcurrent Protection) function of the present scale. The query returns "ON" or "OFF".
Example	OUTP:OCP:STAT ON
6. OUTPut:OCP	
Syntax	OUTPut:OCP <value> OUTPut:OCP?
Function	Sets the O.C.P value of the present scale. The query returns the O.C.P setting value of the present scale.
Explanation	<value> ranges from 0.1 A to 11 A.
Example	OUTP:OCP 1.5 The query returns "1.5".
7. OUTPut:WAVE	
Syntax	OUTPut:WAVE {OFF|ON} OUTPut:WAVE?
Function	Turns on or off the waveform display mode.

	The query returns "ON" or "OFF".
Example	OUTP:WAVE ON
8. OUTPut:TIMEr	
Syntax	OUTPut:TIMEr <secnum>,<volt>,<curr>,<time> OUTPut:TIMEr? <secnum>
Function	Specifies the timing parameters for the present scale. The query returns a group of timing pararmeters specified by <secnum>. The returned values are Number of Section, Voltage, Current and Time.
Explanation	• <secnum> ranges from 0 to 99. • <time> ranges from 0 to 99999. • <volt> ranges from 0 to 33.6V. <curr> ranges from 0 to 10.5A. • The setting parameters of the timing are insensitive to the scale. But note that the vaule of <curr> should be less than 5.25A when the value of <volt> is specified higher than 16.8V, and the <curr> can be set to 10.5A at most when the value of <volt> is less than 16.8V.
Example	OUTP:TIME 5,9,6,2 The query returns "5,9.000V,6.000A,2s".
9. OUTPut:TIMEr:STATe	
Syntax	OUTPut:TIMEr:STATe {OFF|ON} OUTPut:TIMEr:STATe?
Function	Turns on or off the timing output. The query returns "SET", "PAUSE", "RUN" or "OFF". Therein, SET denotes the instrument is in a state of timing paramters setting; PAUSE denotes the timer is paused currently; RUN denotes the instrument is in timing state and OFF denotes the timer is closed.
Example	OUTP:TIME:STAT ON
10. OUTPut:TIMEr:SAVE	
Syntax	OUTPut:TIMEr:SAVE
Function	Saves the specified timing parameters into the nonvolatile memory.

Example	OUTP:TIME:SAVE
11. OUTPut:TIMEr:CIRcle	
Syntax	OUTPut:TIMEr:CIRcle {<value>|INFInite} OUTPut:TIMEr:CIRcle?
Function	Specifies the number of cycles for the timing output. The query returns the specified number of cycles of the timing output or "Infinite".
Explanation	<value> ranges from 1 to 1000.
Example	OUTP:TIME:CIR 2 The query returns "2".

SOURce Commands

SOURce commands are used to directly set the voltage and current values in the present scale.

The DP1116A supports the following **SOURce** commands:

1. [SOURce:]CURRent[:LEVel][:IMMediate][:AMPLitude]
2. [SOURce:]VOLTage[:LEVel][:IMMediate][:AMPLitude]

The detailed information of each **SOURce** command are:

1. [SOURce:]CURRent[:LEVel][:IMMediate][:AMPLitude]	
Syntax	[SOURce:]CURRent[:LEVel][:IMMediate][:AMPLitude] {<current>|MINimum|MAXimum} [SOURce:]CURRent[:LEVel][:IMMediate][:AMPLitude]? [MINimum|MAXimum]
Function	Sets the current value of the present scale. The query returns the setting value of the current of the present scale.
Explanation	● The current value of the present scale will be changed as soon as you send the command. ● The range of the current value of the present scale you can specify is different in different scale. In **16V/10A**: <current> ranges from 0 to 10.5 A. In **32V/5A**: <current> ranges from 0 to 5.25 A.
Example	SOUR:CURR:LEV:IMM:AMPL 8 The query returns "8.000A".
2. [SOURce:]VOLTage[:LEVel][:IMMediate][:AMPLitude]	
Syntax	[SOURce:]VOLTage[:LEVel][:IMMediate][:AMPLitude] {<voltage>|MINimum|MAXimum} [SOURce:]VOLTage[:LEVel][:IMMediate][:AMPLitude]? [MINimum|MAXimum]
Function	Sets the voltage value of the present scale. The query returns the setting value of the voltage of the present scale.
Explanation	● The voltage value of the present scale will be changed as soon

	as you send the command. ● The range of the voltage value of the present scale you can specify is different in different scale. In **16V/10A**: <voltage> ranges from 0 to 16.8 V. In **32V/5A**: <voltage> ranges from 0 to 33.6 V.
Example	SOUR:VOLT:LEV:IMM:AMPL 16 The query returns "16.00V".

DISPlay Command

DISPlay command is used to set the display function of the instrument.

The DP1116A supports the following **DISPlay** command:

1. DISPlay:CLASsical

The detailed information of the **DISPlay** command are:

1. DISPlay:CLASsical	
Syntax	DISPlay:CLASsical {ON|OFF} DISPlay:CLASsical?
Function	Turns on or off the classic display mode of the instrument. The query returns "ON" or "OFF".
Example	DISP:CLAS ON

SYSTem Commands

SYSTem commands provide users with some inforamtion about the system such as Language, Function setting, PowerOn setting, Instrument Control, Network setting, System Self-test and the like.

The DP1116A supports the following **SYSTem** commands:

1. SYSTem:LANGuage:TYPE
2. SYSTem:BEEPer[:IMMediate]
3. SYSTem:BRIGht
4. SYSTem:ONPOwer
5. SYSTem:REMote
6. SYSTem:COMMunicate:LAN:DHCP[:STATe]
7. SYSTem:COMMunicate:LAN:AUTOip[:STATe]
8. SYSTem:COMMunicate:LAN:MANualip[:STATe]
9. SYSTem:COMMunicate:LAN:IPADdress
10. SYSTem:COMMunicate:LAN:SMASk
11. SYSTem:COMMunicate:LAN:GATEway
12. SYSTem:COMMunicate:LAN:DNS
13. SYSTem:COMMunicate:LAN:APPLy
14. SYSTem:COMMunicate:GPIB:ADDRess
15. SYSTem:SELF:TEST:AD?
16. SYSTem:SELF:TEST:DA?
17. SYSTem:SELF:TEST:FPGA?
18. SYSTem:SELF:TEST:USB?
19. SYSTem:SELF:TEST:LAN?
20. SYSTem:SELF:TEST:FAN?
21. SYSTem:OTP

NOTE:
The command "**SYSTem:COMMunicate:LAN:APPLy**" must be executed after you send a command related to the LAN interface in order to bring the parameter you specify into effect.

The detailed information of each **SYSTem** command are:

1. SYSTem:LANGuage:TYPE	
Syntax	SYSTem:LANGuage:TYPE {EN|CH} SYSTem:LANGuage:TYPE?
Function	Specifies a language for the instrument. The query returns "English" or "Chinese".
Example	SYST:LANG:TYPE CH
2. SYSTem:BEEPer[:IMMediate]	
Syntax	SYSTem:BEEPer[:IMMediate] {ON|OFF} SYSTem:BEEPer[:IMMediate]?
Function	Turns on or off the beeper. The query returns "ON" or "OFF".
Explanation	The instrument will make a buzzing sound once any system message pops up and you will also hear a key tone once a key is pressed down during the work of the beeper.
Example	SYST:BEEP:IMM ON
3. SYSTem:BRIGht	
Syntax	SYSTem:BRIGht <brightness> SYSTem:BRIGht?
Function	Sets the LCD brightness of the instrument. The query returns the current brightness.
Explanation	<brightness> ranges from 1 to 8.
Example	SYST:BRIG 5 The query returns "5".
4. SYSTem:ONPOwer	
Syntax	SYSTem:ONPOwer {DEF|LAST} SYSTem:ONPOwer?
Function	Sets the PowerOn mode of the system. The query returns "Last" or "Default".
Explanation	• DEF denotes the system will be reset to the defaults after a power cycle. • LAST denotes the system will use the last settings after a power cycle.
Example	SYST:ONPO DEF

5. SYSTem:LOCal	
Syntax	SYSTem:LOCal
Function	Enables the Local mode of the instrument.
Explanation	In local mode, all keys on the panel of the instrument can be used as usual.
Example	SYST:LOC
6. SYSTem:REMote	
Syntax	SYSTem:REMote
Function	Enables the remote mode of the instrument.
Explanation	In remote mode, none of keys on the panel of the instrument can be used except the "Local" (7) key.
Example	SYST:REM
7. SYSTem:COMMunicate:LAN:DHCP[:STATe]	
Syntax	SYSTem:COMMunicate:LAN:DHCP[:STATe] {ON|OFF} SYSTem:COMMunicate:LAN:DHCP[:STATe]?
Function	Turns on or off the DHCP configuration mode. The query returns "ON" or "OFF".
Example	SYST:COMM:LAN:DHCP:STAT ON
8. SYSTem:COMMunicate:LAN:AUTOip[:STATe]	
Syntax	SYSTem:COMMunicate:LAN:AUTOip[:STATe] {ON|OFF} SYSTem:COMMunicate:LAN:AUTOip[:STATe]?
Function	Turns on or off the AUTOip configuration mode. The query returns "ON" or "OFF".
Example	SYST:COMM:LAN:AUTO:STAT ON
9. SYSTem:COMMunicate:LAN:MANualip[:STATe]	
Syntax	SYSTem:COMMunicate:LAN:MANualip[:STATe] {ON|OFF} SYSTem:COMMunicate:LAN:MANualip[:STATe]?
Function	Turns on or off the MANualip configuration mode. The query returns "ON" or "OFF".
Example	SYST:COMM:LAN:MAN:STAT ON
10. SYSTem:COMMunicate:LAN:IPADdress	
Syntax	SYSTem:COMMunicate:LAN:IPADdress <ip>

	SYSTem:COMMunicate:LAN:IPADdress?
Function	Sets the IP address of the instrument. The query returns the current IP address of the instrument.
Explanation	● The format of <ip>is: nnn.nnn.nnn.nnn, thereinto, the first "nnn" ranges from 0 to 223 (except 127) and the others range from 0 to 255. ● After sending this command, execute the command **SYSTem:COMMunicate:LAN:APPLy**. The new IP address is available in Manual IP mode only.
Example	SYST:COMM:LAN:IPAD 172.16.3.32 The query returns "172.16.3.32".
11. SYSTem:COMMunicate:LAN:SMASk	
Syntax	SYSTem:COMMunicate:LAN:SMASk <submask> SYSTem:COMMunicate:LAN:SMASk?
Function	Sets the subnet mask of the instrument. Queries the current subnet mask of the instrument.
Explanation	● The format of <submask> is: nnn.nnn.nnn.nnn, thereinto, "nnn" ranges from 0 to 255. ● After sending this command, execute the command **SYSTem:COMMunicate:LAN:APPLy**. The new subnet mask is available in Manual IP mode only.
Example	SYST:COMM:LAN:SMAS 255.255.255.0 The query returns "255.255.255.0".
12. SYSTem:COMMunicate:LAN:GATEway	
Syntax	SYSTem:COMMunicate:LAN:GATEway <gateway> SYSTem:COMMunicate:LAN:GATEway?
Function	Sets the gateway of the instrument. The query returns the current gateway of the instrument.
Explanation	● The format of <gateway> is: nnn.nnn.nnn.nnn, thereinto, the first "nnn" ranges from 0 to 223 (except 127) and the others range from 0 to 255. ● After sending this command, execute the command **SYSTem:COMMunicate:LAN:APPLy**. The gateway is available in Manual IP mode only.
Example	SYST:COMM:LAN:GATE 172.16.3.1

	The query returns "172.16.3.1".
13. SYSTem:COMMunicate:LAN:DNS	
Syntax	SYSTem:COMMunicate:LAN:DNS <dns> SYSTem:COMMunicate:LAN:DNS?
Function	Sets the DNS address of the network. The query returns the current DNS address.
Explanation	• The format of <dns> is: nnn.nnn.nnn.nnn, thereinto, the first "nnn" ranges from 0 to 223 (except 127) and the others range from 0 to 255. • After sending this command, execute the command **SYSTem:COMMunicate:LAN:APPLy** in order to bring the new DNS address you specify into effect.
Example	SYST:COMM:LAN:DNS 172.16.2.3 The query returns "172.16.2.3".
14. SYSTem:COMMunicate:LAN:APPLy	
Syntax	SYSTem:COMMunicate:LAN:APPLy
Function	Applys the specifed LAN parameters of the instrument.
Example	SYST:COMM:LAN:APPL
15. SYSTem:COMMunicate:GPIB:ADDRess	
Syntax	SYSTem:COMMunicate:GPIB:ADDRess <gpib address> SYSTem:COMMunicate:GPIB:ADDRess?
Function	Sets the GPIB address of the instrument. The query returns the GPIB address of the instrument.
Explanation	<gpib address> ranges from 1 to 30.
Example	SYST:COMM:GPIB:ADDR 10 The query returns "10".
16. SYSTem:SELF:TEST:AD?	
Syntax	SYSTem:SELF:TEST:AD?
Function	Queries the self-test result of the AD converter.
Return	The query returns "Pass" or "Error".
17. SYSTem:SELF:TEST:DA?	
Syntax	SYSTem:SELF:TEST:DA?

Function	Queries the self-test result of the DA converter.
Return	The query returns "Pass" or "Error".
18. SYSTem:SELF:TEST:FPGA?	
Syntax	SYSTem:SELF:TEST:FPGA?
Function	Queries the self-test result of the FPGA module.
Return	The query returns "Pass" or "Error".
19. SYSTem:SELF:TEST:USB?	
Syntax	SYSTem:SELF:TEST:USB?
Function	Queries the self-test result of the USB insterface.
Return	The query returns "Pass" or "Error".
20. SYSTem:SELF:TEST:LAN?	
Syntax	SYSTem:SELF:TEST:LAN?
Function	Queries the self-test result of the LAN insterface.
Return	The query returns "Pass" or "Error".
21. SYSTem:SELF:TEST:FAN?	
Syntax	SYSTem:SELF:TEST:FAN?
Function	Queries the self-test result of the fan unit.
Return	The query returns "Pass" or "Error".
22. SYSTem:OTP	
Syntax	SYSTem:OTP {ON|OFF} SYSTem:OTP?
Function	Turns on or off the OTP setting. The query returns "ON" or "OFF".
Example	SYST:OTP OFF

STORe Commands

STORe commands are used to save the system status information into the internal or an external memory.

The DP1116A supports the following **STORe** commands:

1. STORe:LOCal
2. STORe:EXTErnal

The detailed information of each **STORe** command are:

1. STORe:LOCal	
Syntax	STORe:LOCal {1|2|3|4},<name>
Function	Saves the current system status into the internal nonvalatile memory of the instrument with a specified name. Note this command has the same function as the command "***SAV**".
Explanation	● The file name should be within 6 Chinese characters or 12 English letters (contains numbers) long. ● The DP1116A provides 4 nonvalatile memories for status storage (No. "1", "2", "3" and "4").
Example	STOR:LOC 1,RIGOL
2. STORe:EXTErnal	
Syntax	STORe:EXTErnal <name>
Function	Saves the current system status into the root folder of the external memory with a specified name.
Explanation	The file name should be within 6 Chinese characters or 12 English letters (contains numbers) long.
Example	STOR:EXTE RIGOL

RECAll Commands

RECAll commands are used to recall the specified system status file from the internal or an external memory.

The DP1116A supports the following **RECAll** commands:

1. RECAll:LOCal
2. RECAll:EXTErnal

The detailed information of each **RECAll** command are:

1. RECAll:LOCal	
Syntax	RECAll:LOCal {1|2|3|4}
Function	Recalls the specified system status file from the internal memory. Note this command has the same function as the command "***RCL**".
Example	RECA:LOC 2
2. RECAll:EXTErnal	
Syntax	RECAll:EXTErnal <name>
Function	Recalls the specified system status file that stored under the root directory of the external memory.
Example	RECA:EXTE RIGOL

Chapter 3 Programming Examples

This chapter lists some programming examples in the development environments of Visual C++ 6.0 and Visual Basic 6.0 as well as LabVIEW 8.6. All examples are based on VISA (Virtual Instrument Software Architecture).

VISA is an API (Application Programming Interface) that is used for controlling an instrument. It is convenient for users to develop testing applications which are independent of the types of instrument and interface. Note the "VISA" mentioned here is NI (National Instrument)-VISA. NI-VISA is an API written by NI and based on VISA standards. You can use NI-VISA to make a communication between the DP1116A and the PC using USB, GPIB, LAN or such instrument bus. The VISA defines a set of software commands, users can control the instrument without understanding the working state of the interface bus. For more details, please refer to NI-VISA.

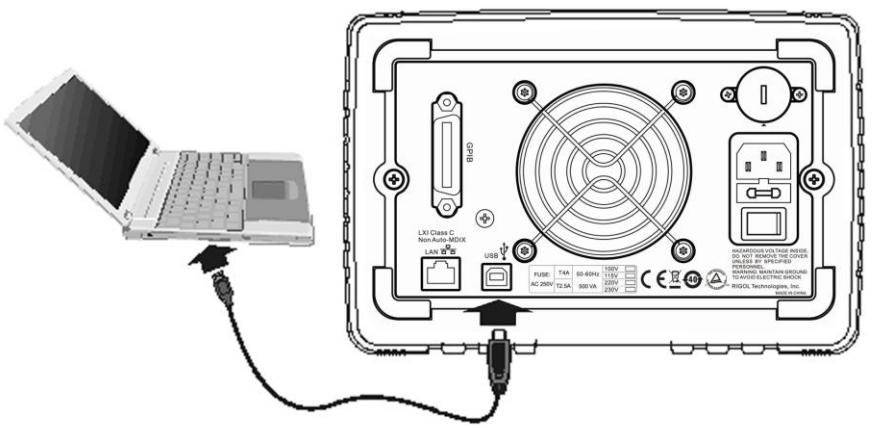
This chapter contains the following topics:

- Prepare for Programming
- Program in Visual C++ 6.0
- Program in Visual Basic 6.0
- Program in LabVIEW 8.6

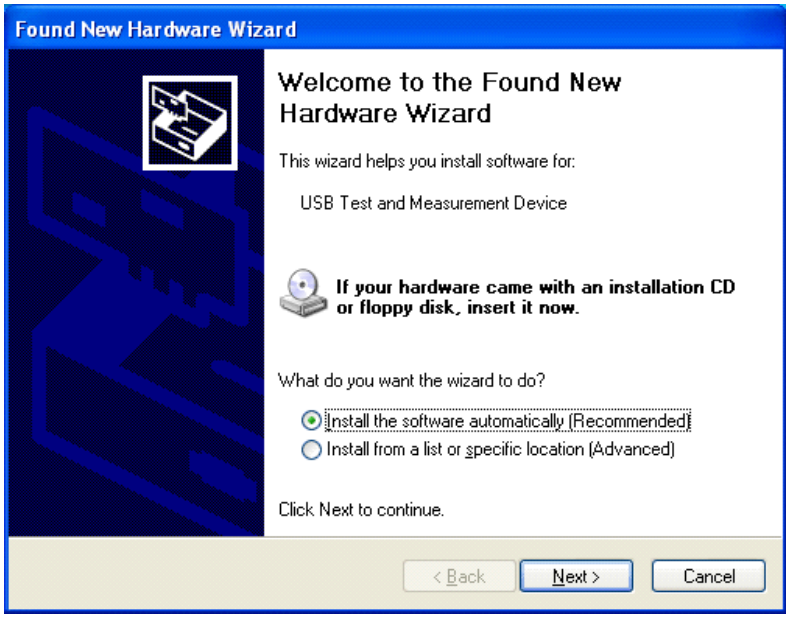
Prepare for Programming

Please make sure your computer has installed VISA library of NI (http://www.ni.com). Here we install it in the default path: C:\Program Files\IVI Foundation\VISA.

In the following text, we will use the USB interface to communicate between the DP1116A and the PC. See figure below.



After successful connection, turn on the instrument and a dialog will pop up to guide you to install the driver of "USB Test and Measurement Device" on the PC. See figure below.

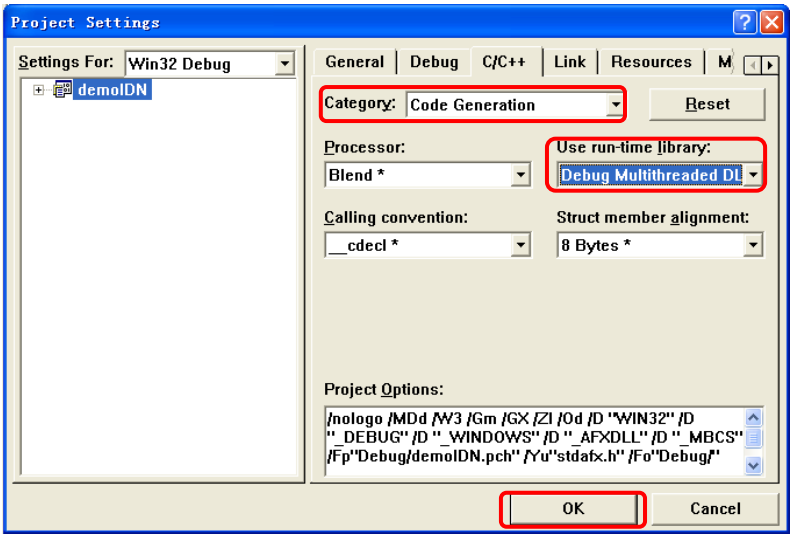


At present, you have finished all of the preparations. Next, we will give you some programming examples in Visual C++ 6.0, Visual Basic 6.0 and LabVIEW 8.6.

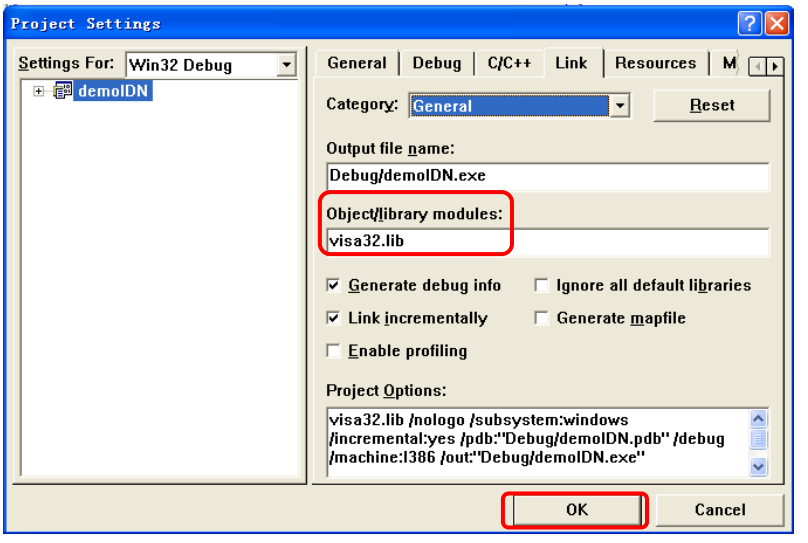
Program in Visual C++ 6.0

Open the Visual C++ 6.0 and take the following steps:

1. Create a project based on MFC.
2. Click **Project → Settings → C/C++** and respectively select files "Code Generation" and "Debug Multithreaded DLL" in **Category** and **Use run-time library** options.

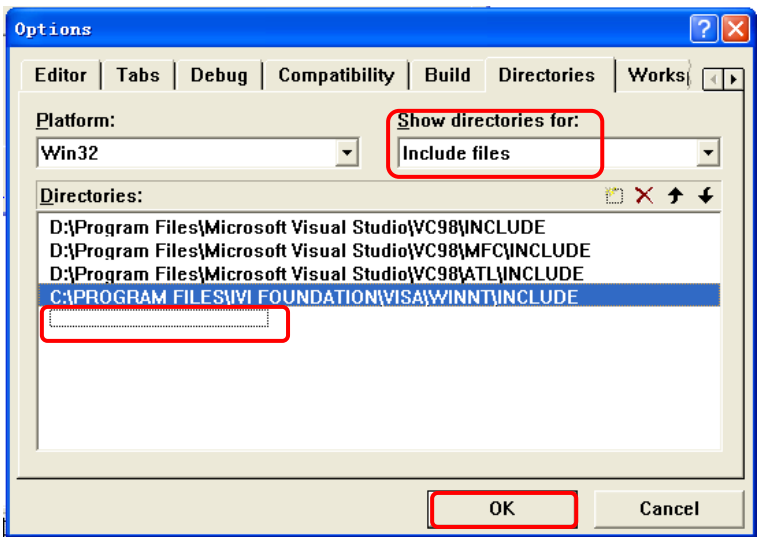


3. Click **Project → Settings → Link** and manually add the file "visa32.lib" in **Object/library modules**.



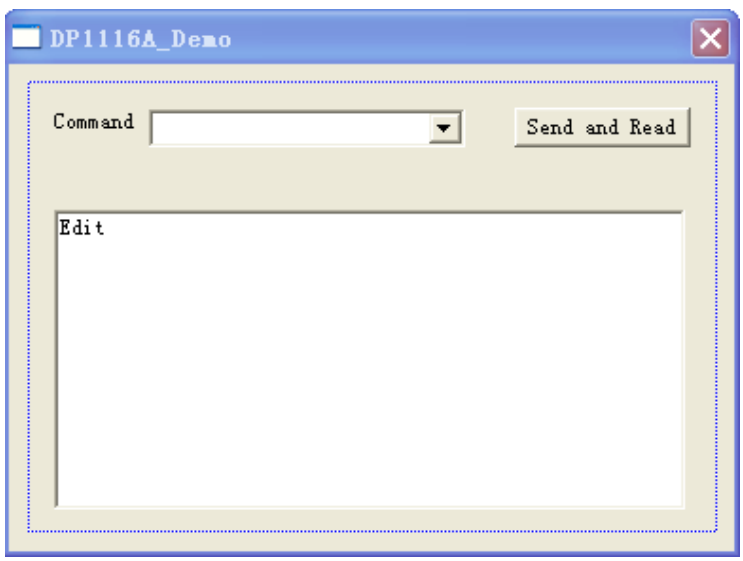
4. Click **Tools→Options→Directories** and then:
Select the "Include files" in **Show directories for** and dblclick the blank in **Directories** to add the path of "**Include"**: C:\VXIpnp\WINNT\include.

Select **"Library files"** in **Show directories for** and dblclick the blank in **Directories** to add the path of **"Lib"**: C:\VXIpnp\WINNT\lib\msc.



Note the VISA library at present has been added successfully.

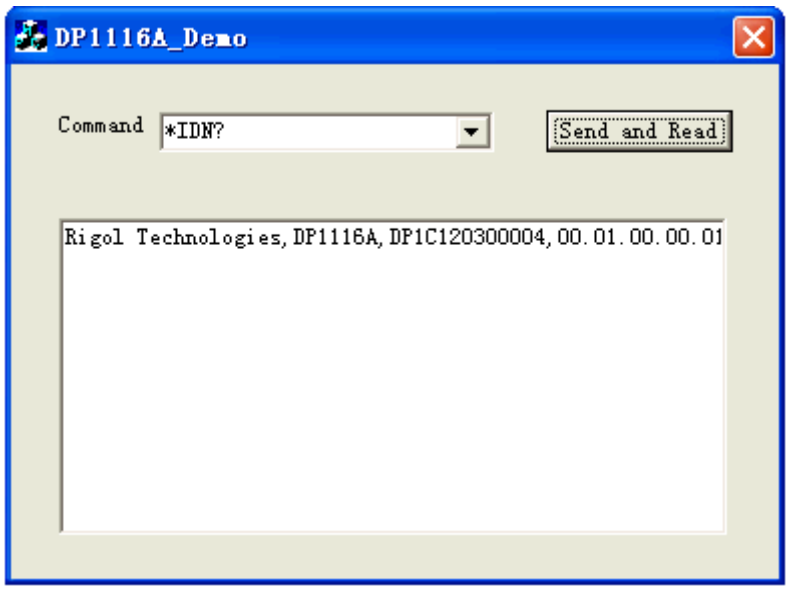
5. Add the controls: Text, Com box, Button and Edit. See figure below.



1) Name the **Text** "Command".
2) Click **Data** from the properties of **Com box** and manually enter a command "*IDN?".
3) Click **General** from the properties of **Edit** and select **Disable**.
4) Name the **Button** "Send and Read**"** and enter the following codes:

```
void CDP1116A_DemoDlg::OnSendRead()
{
    // add codes in this place
    CString strTemp;
    char buf [256] = {0};

    //send the received commands
    m_combox.GetLBText(m_combox.GetCurSel(),strTemp);
    strTemp = strTemp + "\n";
    viPrintf (m_vi,strTemp.GetBuffer(strTemp.GetLength()));

    //read the results
    viScanf (m_vi, "%t\n", &buf);

    //display the results
    UpdateData (TRUE);
    m_receive = buf;
    UpdateData (FALSE);
```

6. Check the current interface information of the instrument and update the interface information shown in the program; then save, compile and run the project, you will obtain an executable file. When a connection is alive between the instrument and the PC, selecting the command "*IDN?" and clicking "Send and Read" will display the value read from the instrument, see figure below.

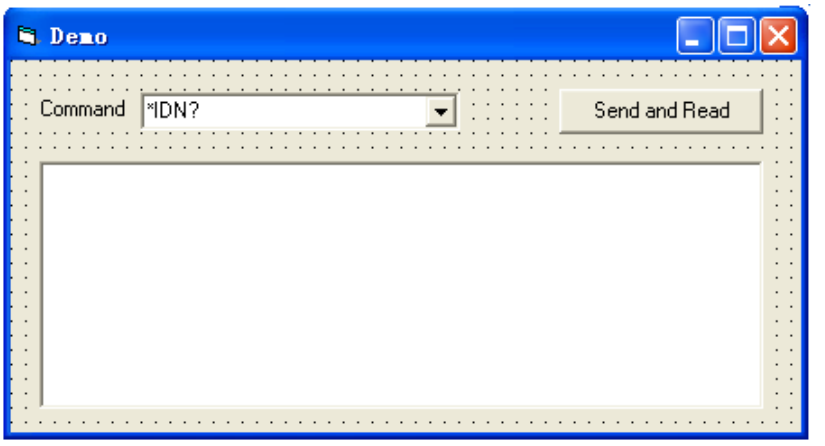

DP1116A_Demo
Command
*IDN?
Send and Read
Rigol Technologies,DP1116A,DP1C120300004,00.01.00.00.01

Program in Visual Basic 6.0

Open the Visual Basic 6.0 and take the following steps:

1. Create a **Standard EXE** project.
2. Click **Project→Add Module** and open the option of **Existing;** then search the file **"visa32.bas"** that was installed under the file path of NI-VISA from the **include** file folder and add it.

3. Add the desired **Text Box** and **CommondButton** into the **Form** to get the following interface.

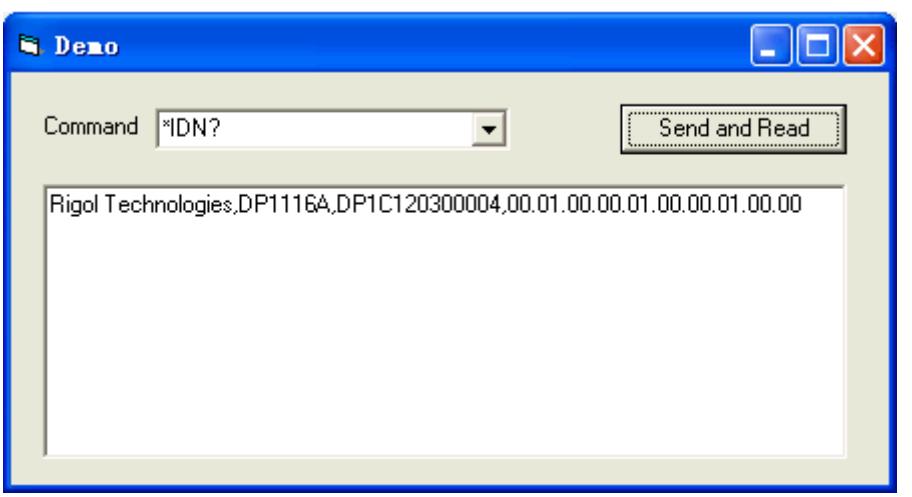


4. Enter the program and add the following codes:

```
Private Sub Form_Load()

CmdList.AddItem "*IDN?"

End Sub

Public Function SendToDevice(ByVal DeviceStr As String, RmtCmd As String)
Const MAX_CNT = 200
Dim stat As Long
Dim dfltRM As Long
Dim sesn As Long
Dim retCount As Long
Dim cmdLen As Integer
Dim fList As Long
Dim rsrcName As String * VI_FIND_BUFLEN
Dim instrDesc As String * VI_FIND_BUFLEN
Dim nList As Long
Dim Buffer As String * MAX_CNT
cmdLen = Len(RmtCmd)

Rem Begin by initializing the system
stat = viOpenDefaultRM(dfltRM)
If (stat < VI_SUCCESS) Then

Rem Error initializing VISA...exiting
Exit Function

End If

Rem Open communication with IO Device
Rem NOTE: For simplicity, we will not show error checking
stat = viOpen(dfltRM, DeviceStr, VI_NULL, VI_NULL, sesn)

Rem Set the timeout for message-based communication
stat = viSetAttribute(sesn, VI_ATTR_TMO_VALUE, 5000)

stat = viWrite(sesn, RmtCmd, cmdLen, retCount)
```

```
stat = viRead(sesn, Buffer, MAX_CNT, retCount)
'DoEvents

Ret.Text = Buffer

stat = viClose(sesn)
stat = viClose(dfltRM)

End Function


Private Sub Command1_Click()
Dim Ret  As String
Ret     =     SendToDevice("USB0::0x1AB1::0x0E10::0000000000000::INSTR",
"*IDN?")
End Sub
```

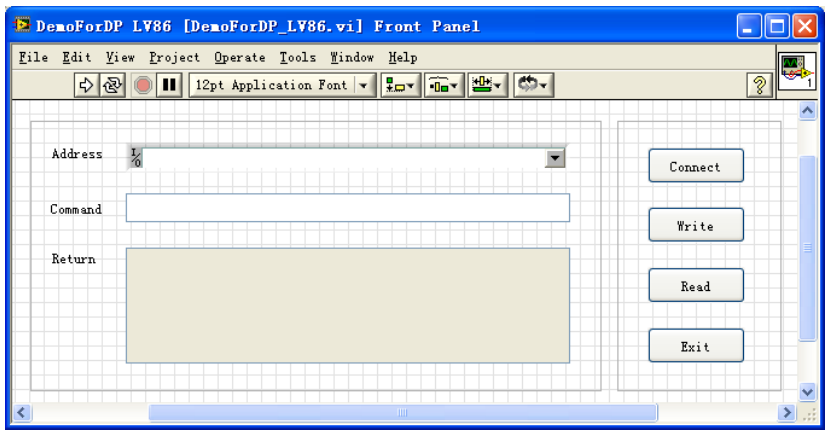
5. Check the current interface information of the instrument and update the interface information shown in the program; then save, compile and run the project to enter a command entry interface. When a connection is alive between the instrument and the PC, sending commands will control the instrument in real time, see figure below.



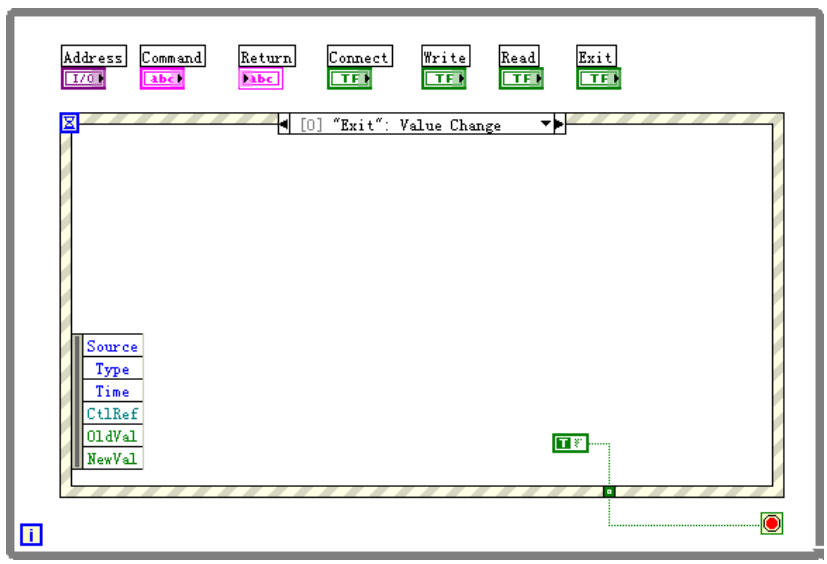
Program in LabVIEW 8.6

Open the Labview 8.6 and take the following steps:

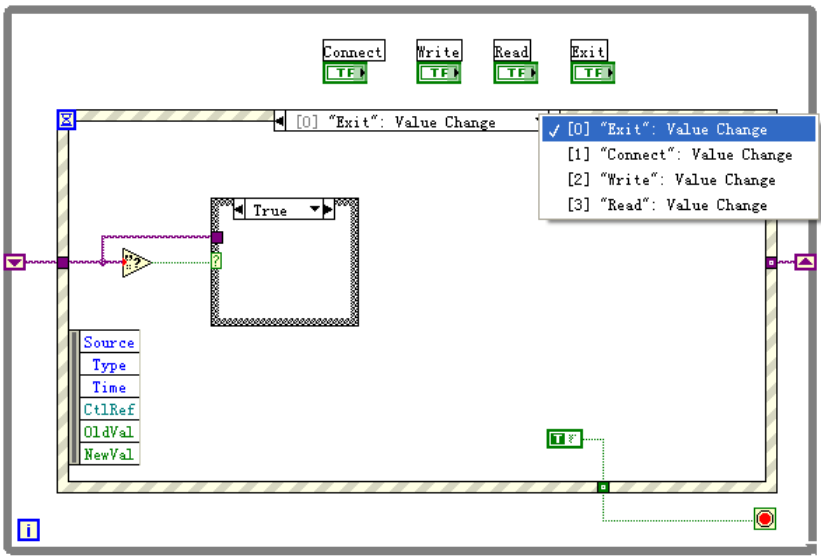
1. Create the base elements for the interface.



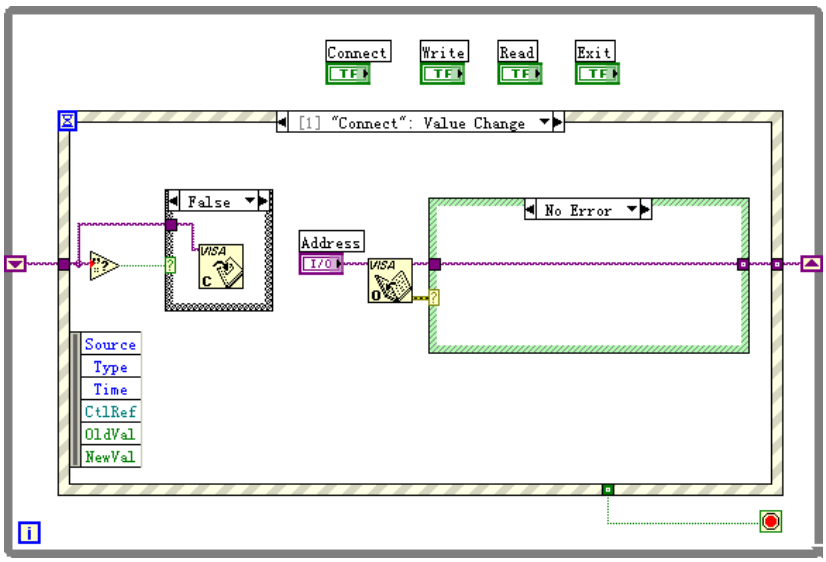
2. Create an event structure.



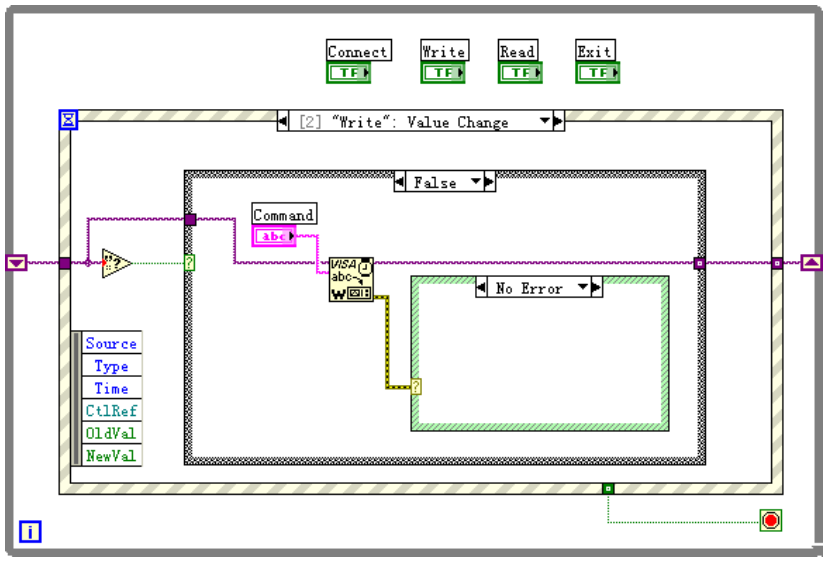
3. Add the events (contains read and write operations, instrument connection and exit).

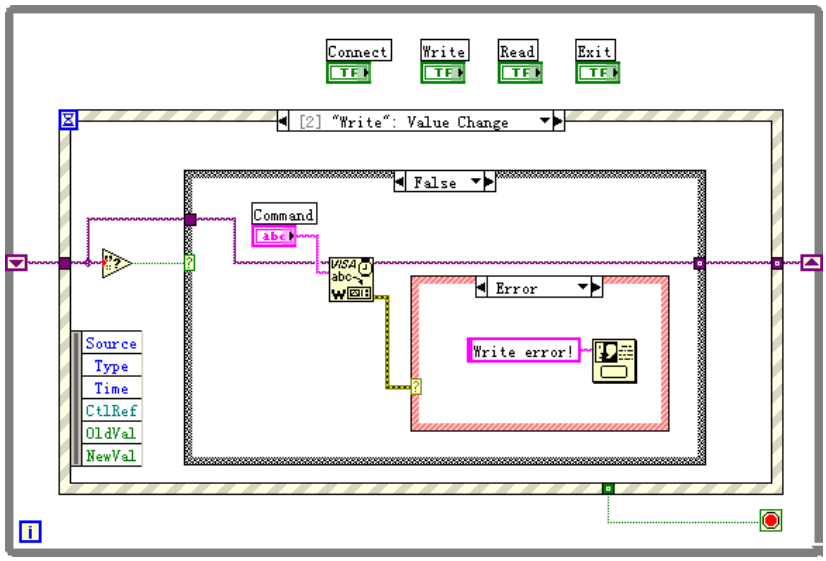


4. The codes for instrument connecting.

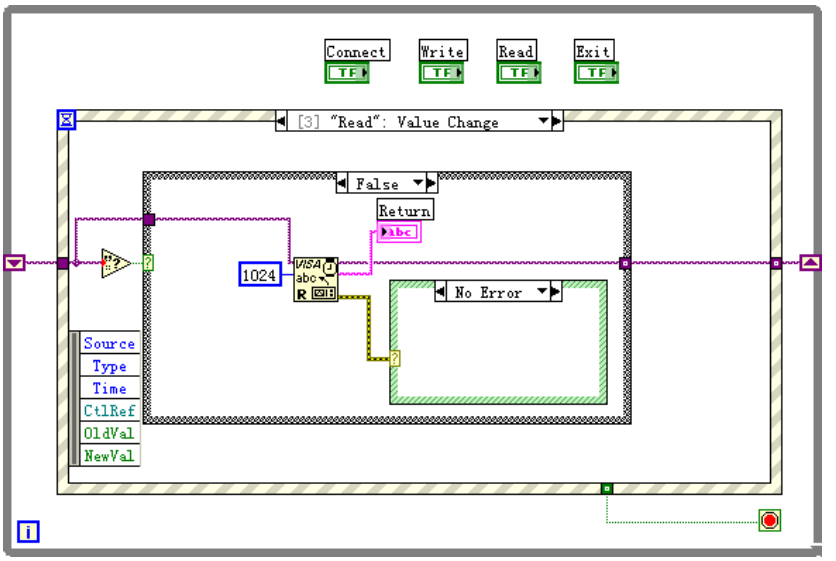


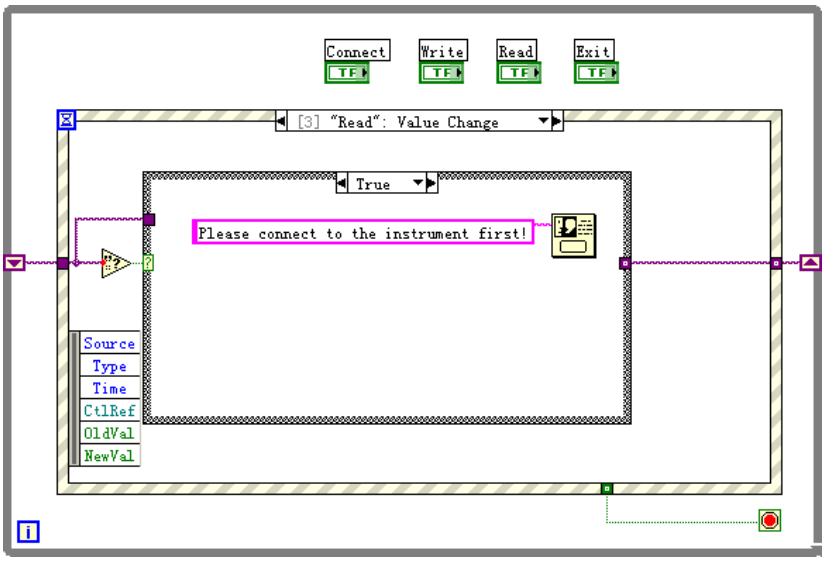
5. The codes for writing operation (contains error judging).



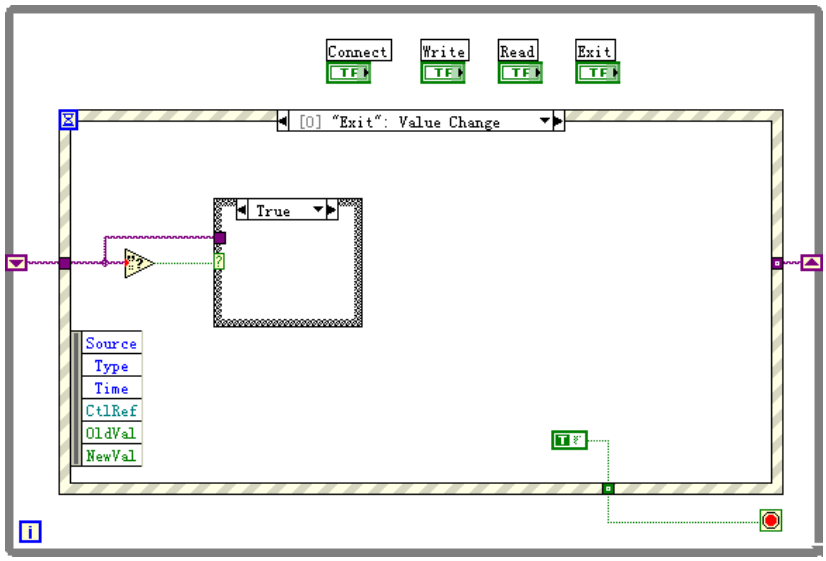


6. The codes for reading operation (contains error handling).

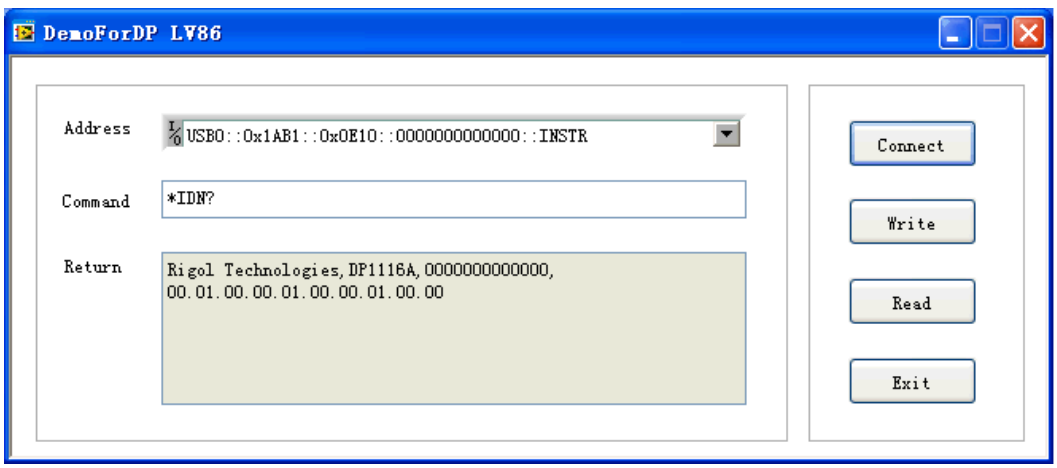




7. The codes for exiting.



8. The running results.



Command Quick Reference A-Z