U9311 Hipot tester Programming Guide

CHANGZHOU EUCOL ELELCTRONIC TECHNOLOGY CO.,LTD.
NO.1,NORTH QINGYANG ROAD,TIANNING DISTRICT,CHANGZHOU
TEL: +86-519-85505199
FAX: +86-519-85505169
WWW.EUCOL.COM.CN
SALES@EUCOL.COM.CN

Programming guide provides guidance for user to program this hipot tester with existing commands, mainly dealing with notation conventions and definitions, short-form rules of command and parameter, commands introduction and appendix.
You can further program this hipot tester with the commands mentioned in this guide.

1 Command Introduction

1.1 Notation Conventions and Definitions

: A colon is used to separate the higher level commands and the lower level commands.
? A question mark is used to generate a query for the command in front of it.
; The semicolon can be used as a separator to execute multiple commands on a single line.
* Asterisk is used to indicate that the command followed is a common command.
, Comma is used to separate the multi-parameters in the command.
- White space is used as a separator between a command and a parameter.
<> Words or characters enclosed in angle brackets symbolize a program code parameter.
[] Items that enclosed in square brackets are optional.
{} When several items are enclosed by brace, only one of these elements may be selected.
NR1 Specify integer data (For example: 12)
NR2 Specify fixed-point data (For example: 12.3)
NR3 Specify exponential data in floating point format (For example: 2.000000e-03)
NL New Line character (ACSII decimal 10) is the end of the input/output string.
Note: behind every command string must be enclosed in NL (ASCII is 10) as command terminator.

1.2 Short-form Rules of Command and Parameter

For memory and writing conveniences to long-form commands or parameters, we will use the following rules to shorten the long-form commands or parameters.
If the length of the command word is four letters or less, no short form version exists.

Example:
TYPE=TYPE
These following rules apply to command words that exceed four letters:
1. If the fourth letter of the command word is a vowel, delete it and all the letters after it.
2. If the fourth letter of the command word is a consonant, retain it but drop all the letters after it.
Examples:
LIMit abbreviates to LIM.
DISPlay abbreviates to DISP.
If the long-form mnemonic is defined as a phrase rather than a single word, then the long-form mnemonic is the first character of the first word followed by the entire last word. The above rules, when the long-form mnemonic is a single word, are then applied to the resulting long-form mnemonic to obtain the short form.
Example:
Save TYPe, whose long form would be STYPe, abbreviates to STYP.

2 Command System

U9311 support the following subsystem commands:

- Common Commands
- DISPlay Subsystem Commands
- STARt/STOP Subsystem Commands
- FETCh Subsystem Commands
- STEP Subsystem Commands
- AC Hipot Subsystem Commands
- DC Hipot Subsystem Commands
- IR Subsystem Commands
- PA Subsystem Commands
- PRESet Subsystem Commands
- Mass MEMory Subsystem Commands
- KEY Subsystem Commands
- SYSTem Subsystem Commands

2.1 Common Commands

The common commands defined by the IEEE 488.2-1987 standard are basic commands in instrument command system which can work with other commands as a command set and also can execute special functions independently.
Common commands used in instrument command system are shown as table 2-1-1.

Command	**Query**	**Return Format**
N/A	*IDN？	Eucol Electronic Technology Co., Ltd.,<model>,< serial number>,< software version>
*RST	N/A	N/A
*RCL <value>	N/A	N/A
*SAV <value>	N/A	N/A
*TRG	N/A	N/A
*CLS	N/A	N/A
*ESE <0-255>	*ESE?	Event Status Enable Register
N/A	*ESR?	Event Status Register
*OPC	*OPC?	Returns 1
*SRE <0-255>	*SRE?	Service Request Enable Register
N/A	*STB?	Service Request Register
N/A	*LRN?	Returns instrument settings

Table 2-1-1

1. ***IDN?**

The *IDN ? query returns the instrument information, including company name, instrument model, instrument serial number and software version.

Query Syntax: *IDN?

Return Format: Eucol Electronic Technology Co., Ltd., <model>, <serial number>, <software revision><NL>

Example:

*IDN? Eucol Electronic Technology Co., Ltd., U9311, 904-A13-105, VER1.0 10 070623A

2. ***RST (Reset)**

The *RST command places the instrument in a known state—factory default state.

Query Syntax: *****RST

3. ***RCL <value>**

The *RCL **<**value**>** command restores the state of the instrument from the specified setup file position, <value>= { 1 to 100 }.
Query Syntax: *RCL <value>
Example:
*RCL 1 Restore the state of the instrument from the specified Setup01.

4. ***SAV <value>**

The *SAV command stores the current state of the instrument to the specified setup file position. <value>= { 1 to 100}.
Command Syntax: *SAV <value>[,"name"], name is the file name,the length of name should be less then 20 charactors.
Example:
*SAV 1 Store the current state of the instrument to the specified Setup01.

5. ***TRG**

The *TRG command generates forcible triggering signal. When an acquisition is completed, the instrument is stopped (similar to single+force trig).
Command Syntax: *TRG

6. ***CLS**

The *CLS command clears the status register, output buffer data and the Request-for-OPC flag.
Command Syntax: *CLS

7. ***ESE <0-255>**

*ESE common command sets the bits in the Standard Event Status Enable Register. The Standard Event Status Enable Register contains a mask value for the bits to be enabled in the Standard Event Status Register. A “1” in the Standard Event Status Enable Register enables the corresponding bit in the Standard Event Status Register.

ESE (Event Status Enable Register)

PON		CME	EXE		QYE		OPC

Event Descriptions

Bit	**Name**	**Description**	**When Set to 1, Enables**
7	PON	Power on	Event when an OFF to ON transition occurs.
5	CME	Command Error	Event when a command error is detected.
4	EXE	Execution Error	Event when an execution error is detected.
2	QYE	Output data loss	Event when data and command in output buffer
0	OPC	Operation complete	Event when an operation is complete.

Command Syntax: *ESE <0-255>

Query Syntax: *ESE?

Return Format：<NR1><NL>　　Return the ESE register value.

8. ***ESR?**

The *ESR? query returns the contents of the Standard Event Status Register. When you read the Event Status Register, the value returned is the total bit weights of all of the bits that are high at the time you read the byte. Reading the register clears the Event Status Register.

ESR（Event Status Register）

Bit	Name	Description	When Set to 1, Indicates:
7	PON	Power on	An OFF to ON transition has occurred.
5	CME	Command error	A command error has been detected.
4	EXE	Execution error	An execution error has been detected.
2	QYE	Output data loss	Output data loss has been detected
0	OPC	Operation complete	Operation is complete.

Query Syntax: *ESR?

Return Format: <NR1><NL>　　Return the current status.

9. ***OPC**

The *OPC command places an ASCII “1” in the output queue when all pending device operations have completed.

Command Syntax: *OPC

Query Syntax: *OPC?

Return Format: <1><NL>

Note: The interface hangs until this query returns.

10. ***SRE <0-255>**

The *SRE command sets the bits in the Service Request Enable Register. The Service Request Enable Register contains a mask value for the bits to be enabled in the Status Byte Register. A “1” in the Service Request Enable Register enables the corresponding bit in the Status Byte Register. A “0” disables the bit.

SRE（Service Request Enable Register）

		ESB	MAV				

Event Descriptions

Bit	Name	Description	When Set to 1, Enables:
5	ESB	Event Status Bit	Interrupts when enabled conditions in the Standard Event Status Register (ESR) occur.
4	MAV	Message Available	Interrupts when messages are in the Output Queue.

Command Syntax: *SRE <0-255>

Query Syntax: *SRE?

Return Format: <NR1><NL> Return the current value of the Service Request Enable Register.

11. ***STB?**

The*STB? query returns the current value of the instrument's status byte.

Status Byte Register (STB)

	RQS	ESB	MAV				

Event Descriptions

Bit	Name	Description	When Set to 1, Indicates:
6	RQS	Request Service	When polled, indicates that the device is requesting service or not.
5	ESB	Event Status Bit	Indicates that an enabled condition in the Standard Event Status Register (ESR) has occurred.
4	MAV	Message Available	Indicates that there are messages in the Output Queue.

Query Syntax: *STB?

Return Format: <NR1><NL> Return Service Request Register value.

2.2 DISPlay Subsystem Commands

DISPlay commands are used to control the display system.

DISPlay:PAGE

The DISPlay:PAGE command set up the display page of instrument. The DISPlay:PAGE? Command returns the abbreviated page name currently displayed on the LCD screen.

Command Syntax：DISPlay:PAGE <page name>

<page name> as follows:

MEASurement	set the instrument display page to Measurement display page.
LIST	set the instrument display page to List Sweep page.
MSETup	set the instrument display page to Meas Setup page
MCONfig	set the instrument display page to Meas Config page
SYSTem	set the instrument display page to System page
INFO	set the instrument display page to System Information page
FLISt	set the instrument display page to File list page

Query Command: DISPlay:PAGE?

Return Format： {MEAS| LIST | MSET | MCON | SYST | INFO | FLIS}<NL>

Note: During the measuring, the query command is ignored.

2.3 STARt/STOP Subsystem Commands

STARt/STOP commands are used to control the measurement of the Hipot.

[SOURce:]SAFEty:STARt[:ONCE]
The [SOURce:]SAFEty:STARt[:ONCE] command is used to start the testing.
Command Syntax: [SOURce:]SAFEty:STARt[:ONCE]

[SOURce:]SAFEty:STARt:OFFSet GET
The [SOURce:]SAFEty:STARt:OFFSet GET command is used to clear the offset readings.
Command Syntax: [SOURce:]SAFEty:STARt:OFFSet GET

[SOURce:]SAFEty:STARt:OFFSet {ON | OFF}
The [SOURce:]SAFEty:STARt:OFFSet {OFF | ON} command sets the status of the offset mode. The [SOURce:]SAFEty:STARt:OFFSet? query returns the current status of offset mode.
Command Syntax: [SOURce:]SAFEty:STARt:OFFSet {OFF | ON}
Query Syntax: [SOURce:]SAFEty:STARt:OFFSet?
Return format: {0 | 1}<NL> 0: means OFF, 1: means ON

[SOURce:]SAFEty:STOP
The [SOURce:]SAFEty:STOP command is used to stop a test. It also cancels FAIL,PASS,STOP status.
Command Syntax: [SOURce:]SAFEty:STOP

[SOURce:]SAFEty:STATus?
The [SOURce:]SAFEty:STATus? query returns the current status of the instrument.
Command Syntax: [SOURce:]SAFEty:STATus?
Return format: {RUNNING | STOPPED}<NL>

2.4 FETCh subsystem commands

The FETCh subsystem commands are used to output test result and judgment results.

[SOURce:]SAFEty:FETCh?

The [SOURce:]SAFEty:FETCh? query returns all the test results.

Command Syntax: [SOURce:]SAFEty:FETCh? [<item>]{,<item>}

Return format:

Item	Description
STEP	The current step during a test, 0 means stop
MODE	The current mode during a test, NONE means stop
OMETerage	The readings of test voltage
MMETerage	The readings of current(for ACW and DCW test) or resistance(IR test)
RELApsed	The executed rise time during a test
RLEFt	The rest of the rise time during a test
TELApsed	The executed test time during a test
TLEFt	The rest of the test time during a test
FELApsed	The executed fall time during a test
FLEFt	The rest of the fall time during a test
FELApsed	The executed DC charge waiting time during a test
FLEFt	The rest of the DC charge waiting time during a test
CHANnel	The status of the scanning channels

For example: WrtCmd(“[SOURce:]SAFEty:FETCh? STEP,MODE,OMET”);

Returns “1,AC,+5.000000E+02”. Means step 1,AC mode, 0.500kV.

[SOURce:]SAFEty:RESult:ALL[:JUDGment]? query returns the judgment result.

Query Syntax: [SOURce:]SAFEty:RESult:ALL[:JUDGment]?

For example: returns “116” means Pass. See the following table for details.

[SOURce:]SAFEty:RESult:ALL:OMETerage? query returns the test voltage results for all steps.

Query Syntax: [SOURce:]SAFEty:RESult:ALL:OMETerage?

For example: returns “+5.100000E+01” means the test voltage is 0.051kV.

[SOURce:]SAFEty:RESult:ALL:MMETerage? query returns the measuring results for all steps.
Query Syntax: [SOURce:]SAFEty:RESult:ALL:MMETerage?
For example: returns “+6.1200000E-05” means the measuring current is 0.612mA.

[SOURce:]SAFEty:RESult:ALL:TIME[:TEST]? query returns the test time for all steps.
Query Syntax: [SOURce:]SAFEty:RESult:ALL:TIME[:TEST]?
For example: returns “+3.000000E+00” means the test time is 3s.

[SOURce:]SAFEty:RESult:ALL:TIME:RAMP? query returns the rise time for all steps.
Query Syntax: [SOURce:]SAFEty:RESult:ALL:TIME:RAMP?
For example: returns “+1.000000E+00” means the rise time is 1s.

[SOURce:]SAFEty:RESult:ALL:TIME:FALL? query returns the fall time for all steps.
Query Syntax: [SOURce:]SAFEty:RESult:ALL:TIME:FALL?
For example: returns “+3.000000E+00” means the fall time is 3s.

[SOURce:]SAFEty:RESult:ALL:TIME:DWELl? query returns the DC charge waiting time for all steps.
Query Syntax: [SOURce:]SAFEty:RESult:ALL:TIME:DWELl?
For example: returns “+2.500000E+00” means the DC charge waiting time is 2.5s.

[SOURce:]SAFEty:RESult:ALL:MODE? query returns the test mode for all steps.
Query Syntax: [SOURce:]SAFEty:RESult:ALL:MODE?
Return format: {AC |DC | IR | PA | OS}<NL>
For example: returns “DC” means the current test mode is DC withstanding voltage test.

[SOURce:]SAFEty:RESult:COMPlete? query whether the instrument executed all steps.
Query Syntax: [SOURce:]SAFEty:RESult:COMPlete?
Return format: {0 | 1}<NL>
0 means not completed, 1 means completed all test steps

[SOURce:]SAFEty:RESult:AREPort[:JUDGment][:MESSage] sets whether returns the judgment results automatically after the end of comprehensive testing. The

[SOURce:]SAFEty:RESult:AREPort[:JUDGment][:MESSage]? query returns the current status.

Command Syntax: [SOURce:]SAFEty:RESult:AREPort[:JUDGment][:MESSage] {{1 | ON} | {0 | OFF}}

Query Syntax: [SOURce:]SAFEty:RESult:AREPort[:JUDGment][:MESSage]?

Return format: {1 | 0}<NL>

When set to ON or 1, returns "PASS" or "FAIL" string data after test

When set to OFF or 0, does not automatically return the result.

[SOURce:]SAFEty:RESult:AREPort:OMETerage sets whether returns the test voltage automatically after the end of comprehensive testing. The [SOURce:]SAFEty:RESult:AREPort:OMETerage? query returns the current status.

Command Syntax: [SOURce:]SAFEty:RESult:AREPort:OMETerage {{1 | ON} | {0 | OFF}}

Query Syntax: [SOURce:]SAFEty:RESult:AREPort[:JUDGment][:MESSage]?

Return format: {1 | 0}<NL>

When set to ON or 1, returns the test voltage of all steps after test, if some steps are not tested, returns +9.910000E+37

When set to OFF or 0, does not automatically return the result.

[SOURce:]SAFEty:RESult:AREPort:MMETerage sets whether returns the measuring results for all steps automatically after the end of comprehensive testing. The [SOURce:]SAFEty:RESult:AREPort:MMETerage? query returns the current status.

Command Syntax: [SOURce:]SAFEty:RESult:AREPort:MMETerage {{1 | ON} | {0 | OFF}}

Query Syntax: [SOURce:]SAFEty:RESult:AREPort:MMETerage?

Return format: {1 | 0}<NL>

When set to ON or 1, returns the measuring results of all steps after test, if some steps are not tested, returns +9.910000E+37

When set to OFF or 0, does not automatically return the result.

[SOURce:]SAFEty:RESult[:LAST][:JUDGment]? query returns the judgment result of the last step.

Query Syntax: [SOURce:]SAFEty:RESult[:LAST][:JUDGment]?

For example: returns "116" means Pass. See the following table for details.

[SOURce:]SAFEty:RESult:LAST:OMETerage? query returns the test voltage results for the last step.
Query Syntax: [SOURce:]SAFEty:RESult:LAST:OMETerage?
For example: returns “+5.100000E+01” means the test voltage is 0.051kV.

[SOURce:]SAFEty:RESult:LAST:MMETerage? query returns the measuring results for the last step.
Query Syntax: [SOURce:]SAFEty:RESult:LAST:MMETerage?
For example: returns “+6.1200000E-05” means the measuring current is 0.612mA.

[SOURce:]SAFEty:RESult:STEP<n>[:JUDGment]? query returns the judgment result for the specified step.
Query Syntax: [SOURce:]SAFEty:RESult:STEP<n>[:JUDGment]?
For example: returns “116” means Pass. See the following table for details.

[SOURce:]SAFEty:RESult:STEP<n>:OMETerage? query returns the test voltage results for the specified step.
Query Syntax: [SOURce:]SAFEty:RESult:STEP<n>:OMETerage?
For example: returns “+5.100000E+01” means the test voltage is 0.051kV.

[SOURce:]SAFEty:RESult:STEP<n>:MMETerage? query returns the measuring results for the specified step.
Query Syntax: [SOURce:]SAFEty:RESult:STEP<n>:MMETerage?
For example: returns “+6.1200000E-05” means the measuring current is 0.612mA.

General Judgment result code table

Judgment result	Code(Hexadecimal)	Code(Decimal)
PASS	74	116
USER STOP	71	113
CAN NOT STOP	72	114
TESTING	73	115
STOP	70	112

Failed Judgment result code table

	AC MODE		DC MODE		IR MODE	
Judgment result	Code (Hexadecimal)	Code (Decimal)	Code (Hexadecimal)	Code (Decimal)	Code (Hexadecimal)	Code (Decimal)
HI	11	17	21	33	31	49
LO	12	18	22	34	32	50
ARC	13	19	23	35	33	----
CHECK LOW	----	----	25	37	----	----
TRIPPED	79	121	79	121	79	121
RANGE	16	22	26	38	36	54
VOLT LOW	16	22	26	38	36	54

2.5 STEP subsystem commands

The STEP subsystem command group is used to trigger a measurement or to set the trigger mode.

[SOURce:]SAFEty:SNUMber? query returns the steps numbers.
Command Syntax: [SOURce:]SAFEty:SNUMber?
Return format: <NR1><NL>
For example: returns “+2” means there are 2 steps.

[SOURce:]SAFEty:STEP<n>:DELete is used to delete the step.
Command Syntax: [SOURce:]SAFEty:STEP<n>:DELete
Where n is from 1 to 99

[SOURce:]SAFEty:STEP<n>:SET? query returns the settings of the specified step.
Command Syntax: [SOURce:]SAFEty:STEP<n>:SET?
Different test modes return different parameters, as follows:
STEP,AC,VOLT,HIGH,LOW,ARC,TIME,RAMP,FALL,REAL,CHH,CHL,DUT
STEP,DC,VOLT,HIGH,LOW,ARC,TIME,RAMP,FALL,DWELL,CHH,CHL,DUT
STEP,IR,VOLT,HIGH,LOW,TIME,RAMP,FALL,RANGE,CHH,CHL,DUT
STEP,OS,SHORT,OPEN,CHH,CHL,DUT
STEP,PA,MESSAGE,UNDER TEST SIGNAL,TIME
For example:
returns
“1,AC,+5.000000E+01,+5.000000E-04,+8.000000E-06,+2.000000E-04,+3.000000E+00,+1.000000E+00,+2.000000E+00,+3.000000E-04,(@(0)),(@(0)),1”
Means
STEP:1,MODE:AC,VOLT:0.5kV,HIGH:0.500mA,LOW:0.008mA,ARC:2.0mA;TIME:3.0s,RAMP:1.0s,FALL:2.0s,REAL:0.300mA,SCAN HIGH:0,SCAN LOW:0,DUT:1

[SOURce:]SAFEty:STEP<n>:MODE? query returns the test mode of the specified step.
Command Syntax: [SOURce:]SAFEty:STEP<n>:MODE?
Return format: {AC |DC |IR |OS | PA}<NL>

2.6 AC HIPOT subsystem commands

The AC HIPOT subsystem command group is used to set the AC hipot test parameters.

[SOURce:]SAFEty:STEP<n>:AC[:LEVel]

The [SOURce:]SAFEty:STEP<n>:AC[:LEVel] command sets the test voltage. The [SOURce:]SAFEty:STEP<n>:AC[:LEVel]? Query returns the test voltage value.

Command Syntax: [SOURce:]SAFEty:STEP<n>:AC[:LEVel] <level>

where <value> is NR3 format, AC: from 50 to 5000

Query Syntax: [SOURce:]SAFEty:STEP<n>:AC[:LEVel]?

Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:AC:LIMit[:HIGH]

The [SOURce:]SAFEty:STEP<n>:AC:LIMit[:HIGH] command sets the upper limit . The [SOURce:]SAFEty:STEP<n>:AC:LIMit[:HIGH]? Query returns the upper limit.

Command Syntax: [SOURce:]SAFEty:STEP<n>:AC:LIMit[:HIGH] <value>

where <value> is NR3 format, AC: from 0.001mA to 30.000mA.

Query Syntax: [SOURce:]SAFEty:STEP<n>:AC:LIMit[:HIGH]?

Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:AC:LIMit:LOW

The [SOURce:]SAFEty:STEP<n>:AC:LIMit:LOW command sets the lower limit . The [SOURce:]SAFEty:STEP<n>:AC:LIMit:LOW? Query returns the lower limit.

Command Syntax: [SOURce:]SAFEty:STEP<n>:AC:LIMit:LOW <value>

where <value> is NR3 format, AC:from 0.000mA to 30.000mA.

Query Syntax: [SOURce:]SAFEty:STEP<n>:AC:LIMit:LOW?

Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:AC:LIMit:ARC[:LEVel]

The [SOURce:]SAFEty:STEP<n>:AC:LIMit:ARC[:LEVel] command sets the arc detection current . The [SOURce:]SAFEty:STEP<n>:AC:LIMit:ARC[:LEVel]? Query returns the arc detection current.

Command Syntax: [SOURce:]SAFEty:STEP<n>:AC:LIMit:ARC[:LEVel] <value>

where <value> is NR3 format, AC:from 0.0mA to 15.0mA, 0 means shut off the arc detection function.

Query Syntax: [SOURce:]SAFEty:STEP<n>:AC:LIMit:ARC[:LEVel]?

Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:AC:TIME[:TEST]

The [SOURce:]SAFEty:STEP<n>:AC:TIME[:TEST] command sets the test time . The [SOURce:]SAFEty:STEP<n>:AC:TIME[:TEST]? Query returns the test time.

Command Syntax: [SOURce:]SAFEty:STEP<n>:AC:TIME[:TEST] <time> where <time> is NR3 format, from 0.3 to 999.9s.

Query Syntax: [SOURce:]SAFEty:STEP<n>:AC:TIME[:TEST]?

Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:AC:TIME:RAMP

The [SOURce:]SAFEty:STEP<n>:AC:TIME:RAMP command sets the voltage ramp time . The [SOURce:]SAFEty:STEP<n>:AC:TIME:RAMP? Query returns the voltage ramp time.

Command Syntax: [SOURce:]SAFEty:STEP<n>:AC:TIME:RAMP <time>
where <time> is NR3 format, from 0.1 to 999.9s.

Query Syntax: [SOURce:]SAFEty:STEP<n>:AC:TIME:RAMP?

Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:AC:TIME:FALL

The [SOURce:]SAFEty:STEP<n>:AC:TIME:FALL command sets the voltage fall time . The [SOURce:]SAFEty:STEP<n>:AC:TIME:FALL? Query returns the voltage fall time.

Command Syntax: [SOURce:]SAFEty:STEP<n>:AC:TIME:FALL <time>
where <time> is NR3 format, from 0.1 to 999.9s.

Query Syntax: [SOURce:]SAFEty:STEP<n>:AC:TIME:FALL?

Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:AC:CHANnel[:HIGH]

The [SOURce:]SAFEty:STEP<n>:AC:CHANnel[:HIGH] command sets the status of output channel. The [SOURce:]SAFEty:STEP<n>:AC:CHANnel[:HIGH]? Query returns the status of output channel.

Command Syntax: [SOURce:]SAFEty:STEP<n>:AC:CHANnel[:HIGH] (@(ch,ch,..)) where (@(ch,ch,…)) is from 1 to 8

Query Syntax: [SOURce:]SAFEty:STEP<n>:AC:CHANnel[:HIGH]?

Return format: (@(1,2,3))<NL> (@(0)) means all channels are OFF.

[SOURce:]SAFEty:STEP<n>:AC:CHANnel:LOW

The [SOURce:]SAFEty:STEP<n>:AC:CHANnel:LOW command sets the status of output channel. The [SOURce:]SAFEty:STEP<n>:AC:CHANnel:LOW? Query returns the status of output channel.

Command Syntax: HIPot:STEP<n>:CHL (@(ch1,ch2,ch3)) where @(ch1,ch2,ch3) is from ch1 to ch12

Query Syntax: [SOURce:]SAFEty:STEP<n>:AC:CHANnel:LOW?

Return format: (@(1,2,3))<NL> (@(0)) means all channels are OFF.

2.7 DC HIPOT subsystem commands

The DC HIPOT subsystem command group is used to set the DC hipot test parameters.

[SOURce:]SAFEty:STEP<n>:DC[:LEVel]

The [SOURce:]SAFEty:STEP<n>:DC[:LEVel] command sets the test voltage. The [SOURce:]SAFEty:STEP<n>:DC[:LEVel]? Query returns the test voltage value.

Command Syntax: [SOURce:]SAFEty:STEP<n>:DC[:LEVel] <level>

where <value> is NR3 format, DC: from 50 to 6000

Query Syntax: [SOURce:]SAFEty:STEP<n>:DC[:LEVel]?

Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:DC:LIMit[:HIGH]

The [SOURce:]SAFEty:STEP<n>:DC:LIMit[:HIGH] command sets the upper limit . The [SOURce:]SAFEty:STEP<n>:DC:LIMit[:HIGH]? Query returns the upper limit.

Command Syntax: [SOURce:]SAFEty:STEP<n>:DC:LIMit[:HIGH] <value>

where <value> is NR3 format, AC: from 0.001mA to 10.000mA.

Query Syntax: [SOURce:]SAFEty:STEP<n>:DC:LIMit[:HIGH]?

Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:DC:LIMit:LOW

The [SOURce:]SAFEty:STEP<n>:DC:LIMit:LOW command sets the lower limit . The [SOURce:]SAFEty:STEP<n>:DC:LIMit:LOW? Query returns the lower limit.

Command Syntax: [SOURce:]SAFEty:STEP<n>:DC:LIMit:LOW <value>

where <value> is NR3 format, DC:from 0.000mA to 10.000mA.

Query Syntax: [SOURce:]SAFEty:STEP<n>:DC:LIMit:LOW?

Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:DC:LIMit:ARC[:LEVel]

The [SOURce:]SAFEty:STEP<n>:DC:LIMit:ARC[:LEVel] command sets the arc detection current . The [SOURce:]SAFEty:STEP<n>:DC:LIMit:ARC[:LEVel]? Query returns the arc detection current.

Command Syntax: [SOURce:]SAFEty:STEP<n>:DC:LIMit:ARC[:LEVel] <value>
where <value> is NR3 format, DC:from 0.0mA to 15.0mA, 0 means shut off the arc detection function.

Query Syntax: [SOURce:]SAFEty:STEP<n>:DC:LIMit:ARC[:LEVel]?
Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:DC:TIME[:TEST]

The [SOURce:]SAFEty:STEP<n>:DC:TIME[:TEST] command sets the test time . The [SOURce:]SAFEty:STEP<n>:DC:TIME[:TEST]? Query returns the test time.

Command Syntax: [SOURce:]SAFEty:STEP<n>:DC:TIME[:TEST] <time> where <time> is NR3 format, from 0.3 to 999.9s.

Query Syntax: [SOURce:]SAFEty:STEP<n>:DC:TIME[:TEST]?
Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:DC:TIME:RAMP

The [SOURce:]SAFEty:STEP<n>:DC:TIME:RAMP command sets the voltage ramp time . The [SOURce:]SAFEty:STEP<n>:DC:TIME:RAMP? Query returns the voltage ramp time.

Command Syntax: [SOURce:]SAFEty:STEP<n>:DC:TIME:RAMP <time>
where <time> is NR3 format, from 0.1 to 999.9s.

Query Syntax: [SOURce:]SAFEty:STEP<n>:DC:TIME:RAMP?
Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:DC:TIME:FALL

The [SOURce:]SAFEty:STEP<n>:DC:TIME:FALL command sets the voltage fall time . The [SOURce:]SAFEty:STEP<n>:DC:TIME:FALL? Query returns the voltage fall time.

Command Syntax: [SOURce:]SAFEty:STEP<n>:DC:TIME:FALL <time>
where <time> is NR3 format, from 0.1 to 999.9s.

Query Syntax: [SOURce:]SAFEty:STEP<n>:DC:TIME:FALL?
Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:DC:TIME:DWELl

The [SOURce:]SAFEty:STEP<n>:DC:TIME:DWELl command sets the DC charge waiting time . The [SOURce:]SAFEty:STEP<n>:DC:TIME:DWELl? Query returns the DC charge waiting time.

Command Syntax: [SOURce:]SAFEty:STEP<n>:DC:TIME:DWELl <time>

where <time> is NR3 format, from 0.1 to 999.9s.

Query Syntax: [SOURce:]SAFEty:STEP<n>:DC:TIME:DWELl?

Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:DC:CHANnel[:HIGH]

The [SOURce:]SAFEty:STEP<n>:DC:CHANnel[:HIGH] command sets the status of output channel. The [SOURce:]SAFEty:STEP<n>:DC:CHANnel[:HIGH]? Query returns the status of output channel.

Command Syntax: [SOURce:]SAFEty:STEP<n>:DC:CHANnel[:HIGH] (@(ch,ch,..))

where (@(ch,ch,…)) is from 1 to 8

Query Syntax: [SOURce:]SAFEty:STEP<n>:DC:CHANnel[:HIGH]?

Return format: (@(1,2,3))<NL> (@(0)) means all channels are OFF.

[SOURce:]SAFEty:STEP<n>:DC:CHANnel:LOW

The [SOURce:]SAFEty:STEP<n>:DC:CHANnel:LOW command sets the status of output channel. The [SOURce:]SAFEty:STEP<n>:DC:CHANnel:LOW? Query returns the status of output channel.

Command Syntax: [SOURce:]SAFEty:STEP<n>:DC:CHANnel:LOW (@(ch,ch,..))

where @(ch1,ch2,ch3) is from ch1 to ch12

Query Syntax: [SOURce:]SAFEty:STEP<n>:DC:CHANnel:LOW?

Return format: (@(1,2,3))<NL> (@(0)) means all channels are OFF.

2.8 IR subsystem commands

The IR subsystem command group is used to set the insulation resistance test parameters.

[SOURce:]SAFEty:STEP<n>:IR[:LEVel]

The [SOURce:]SAFEty:STEP<n>:DC[:LEVel] command sets the test voltage. The [SOURce:]SAFEty:STEP<n>:IR[:LEVel]? Query returns the test voltage value.

Command Syntax: [SOURce:]SAFEty:STEP<n>:IR[:LEVel] <level>

where <value> is NR3 format, DC: from 500 to 1000

Query Syntax: [SOURce:]SAFEty:STEP<n>:IR[:LEVel]?

Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:IR:LIMit:HIGH

The [SOURce:]SAFEty:STEP<n>:IR:LIMit[:HIGH] command sets the upper limit . The [SOURce:]SAFEty:STEP<n>:IR:LIMit[:HIGH]? Query returns the upper limit.

Command Syntax: [SOURce:]SAFEty:STEP<n>:IR:LIMit:HIGH <value>

where <value> is NR3 format, from 0 to 50G, 0 means shut off the upper limit.

Query Syntax: [SOURce:]SAFEty:STEP<n>:IR:LIMit[:HIGH]?

Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:IR:LIMit[:LOW]

The [SOURce:]SAFEty:STEP<n>:IR:LIMit[:LOW] command sets the lower limit . The [SOURce:]SAFEty:STEP<n>:IR:LIMit[:LOW]? Query returns the lower limit.

Command Syntax: [SOURce:]SAFEty:STEP<n>:IR:LIMit[:LOW] <value>

where <value> is NR3 format, from 1M to 50G.

Query Syntax: [SOURce:]SAFEty:STEP<n>:IR:LIMit[:LOW]?

Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:IR:TIME[:TEST]

The [SOURce:]SAFEty:STEP<n>:IR:TIME[:TEST] command sets the test time . The [SOURce:]SAFEty:STEP<n>:IR:TIME[:TEST]? Query returns the test time.

Command Syntax: [SOURce:]SAFEty:STEP<n>:IR:TIME[:TEST] <time> where <time> is NR3 format, from 0.3 to 999.9s.

Query Syntax: [SOURce:]SAFEty:STEP<n>:IR:TIME[:TEST]?

Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:IR:TIME:RAMP

The [SOURce:]SAFEty:STEP<n>:IR:TIME:RAMP command sets the voltage ramp time .
The [SOURce:]SAFEty:STEP<n>:IR:TIME:RAMP? Query returns the voltage ramp time.
Command Syntax: [SOURce:]SAFEty:STEP<n>:IR:TIME:RAMP <time>
where <time> is NR3 format, from 0.1 to 999.9s.
Query Syntax: [SOURce:]SAFEty:STEP<n>:IR:TIME:RAMP?
Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:IR:TIME:FALL

The [SOURce:]SAFEty:STEP<n>:IR:TIME:FALL command sets the voltage fall time .
The [SOURce:]SAFEty:STEP<n>:IR:TIME:FALL? Query returns the voltage fall time.
Command Syntax: [SOURce:]SAFEty:STEP<n>:IR:TIME:FALL <time>
where <time> is NR3 format, from 0.1 to 999.9s.
Query Syntax: [SOURce:]SAFEty:STEP<n>:IR:TIME:FALL?
Return format: <NR3><NL>

[SOURce:]SAFEty:STEP<n>:IR:AGC

The [SOURce:]SAFEty:STEP<n>:IR:AGC command sets status of AGC function . The [SOURce:]SAFEty:STEP<n>:IR:AGC? Query returns the status of AGC function.
Command Syntax: [SOURce:]SAFEty:STEP<n>:IR:AGC {{1 | ON} | {0 | OFF}}
Query Syntax: [SOURce:]SAFEty:STEP<n>:IR:AGC?
Return format: {1 | 0}<NL>

[SOURce:]SAFEty:STEP<n>:IR:RANGe[:UPPer]

The [SOURce:]SAFEty:STEP<n>:IR:RANGe[:UPPer] command sets the range of IR test .
The [SOURce:]SAFEty:STEP<n>:IR:RANGe[:UPPer]? Query returns the range of IR test.
Command Syntax: [SOURce:]SAFEty:STEP<n>:IR:RANGe[:UPPer] <value>
Query Syntax: [SOURce:]SAFEty:STEP<n>:IR:RANGe[:UPPer]?
Return format: <NR3><NL>
Note: This command selects a range higher than the current that can be measured based on the current entered by the user. For example, SAFEty:STEP3:IR:RANG 0.0003 means the current value measured in STEP 3 is 300uA, so the range will be set as 3mA(higher than 300uA).

[SOURce:]SAFEty:STEP<n>:IR:RANGe:LOWer

The [SOURce:]SAFEty:STEP<n>:IR:RANGe:LOWer command sets the range of IR test . The [SOURce:]SAFEty:STEP<n>:IR:RANGe:LOWer? Query returns the range of IR test.

Command Syntax: [SOURce:]SAFEty:STEP<n>:IR:RANGe:LOWer <value>

Query Syntax: [SOURce:]SAFEty:STEP<n>:IR:RANGe:LOWer?

Return format: <NR3><NL>

Note: This command selects a range lower than the current that can be measured based on the current entered by the user. For example, SAFEty:STEP3:IR:RANG:LOW 0.0003 means the current value measured in STEP 3 is 300uA, so the range will be set as 300A(lower than 300uA).

[SOURce:]SAFEty:STEP<n>:IR:RANGe:AUTO

The [SOURce:]SAFEty:STEP<n>:IR:RANGe:AUTO command sets status of Auto range. The [SOURce:]SAFEty:STEP<n>:IR:RANGe:AUTO? Query returns the status of Auto range.

Command Syntax: [SOURce:]SAFEty:STEP<n>:IR:RANGe:AUTO {{1 | ON} | {0 | OFF}}

Query Syntax: [SOURce:]SAFEty:STEP<n>:IR:RANGe:AUTO?

Return format: {1 | 0}<NL>

[SOURce:]SAFEty:STEP<n>:IR:CHANnel[:HIGH]

The [SOURce:]SAFEty:STEP<n>:IR:CHANnel[:HIGH] command sets the status of output channel. The [SOURce:]SAFEty:STEP<n>:IR:CHANnel[:HIGH]? Query returns the status of output channel.

Command Syntax: [SOURce:]SAFEty:STEP<n>:IR:CHANnel[:HIGH] (@(ch,ch,..)) where (@(ch,ch,…)) is from 1 to 8

Query Syntax: [SOURce:]SAFEty:STEP<n>:IR:CHANnel[:HIGH]?

Return format: (@(1,2,3))<NL> (@(0)) means all channels are OFF.

[SOURce:]SAFEty:STEP<n>:IR:CHANnel:LOW

The [SOURce:]SAFEty:STEP<n>:IR:CHANnel:LOW command sets the status of output channel. The [SOURce:]SAFEty:STEP<n>:IR:CHANnel:LOW? Query returns the status of output channel.

Command Syntax: [SOURce:]SAFEty:STEP<n>:IR:CHANnel:LOW (@(ch,ch,..)) where @(ch1,ch2,ch3) is from ch1 to ch12

Query Syntax: [SOURce:]SAFEty:STEP<n>:IR:CHANnel:LOW?

Return format: (@(1,2,3))<NL> (@(0)) means all channels are OFF.

2.9 OS subsystem commands

The OS subsystem command group is used to set the OS resistance test parameters.

[SOURce:]SAFEty:STEP<n>:OSC[:LIMit]:OPEN

The [SOURce:]SAFEty:STEP<n>:OSC[:LIMit]:OPEN command sets the percentage value as the open-circuit judgment .

The [SOURce:]SAFEty:STEP<n>:OSC[:LIMit]:OPEN? Query returns the open-circuit judgment value.

Command Syntax: [SOURce:]SAFEty:STEP<n>:OSC[:LIMit]:OPEN <value>
where <value> is NR3 format

Query Syntax: [SOURce:]SAFEty:STEP<n>:OSC[:LIMit]:OPEN?

Return format: <NR3><NL>

For example,

SAFEty:STEP4:OSC:LIM:OPEN 0.3 means set the open-circuit judgment as 30%.

SAFEty:STEP4:OSC:LIM:OPEN? returns +3.000000E-01, means the open-circuit judgement values as 30%.

[SOURce:]SAFEty:STEP<n>:OSC[:LIMit]:SHORt

The [SOURce:]SAFEty:STEP<n>:OSC[:LIMit]:SHORt command sets the percentage value as the short-circuit judgment .

The [SOURce:]SAFEty:STEP<n>:OSC[:LIMit]:SHORt? Query returns the short-circuit judgment value.

Command Syntax: [SOURce:]SAFEty:STEP<n>:OSC[:LIMit]:OPEN <value>
where <value> is NR3 format

Query Syntax: [SOURce:]SAFEty:STEP<n>:OSC[:LIMit]:SHORt?

Return format: <NR3><NL>

For example,

SAFEty:STEP4:OSC:LIM:SHORt 3.0 means set the short-circuit judgment as 300%.

SAFEty:STEP4:OSC:LIM:SHORt? returns +3.000000E+00, means the open-circuit judgement values as 300%.

[SOURce:]SAFEty:STEP<n>:OSC:GET

The **[SOURce:]SAFEty:STEP<n>:OSC:GET** command is used perform a Get Standard test and set the test value as the nominal value.

Command Syntax: [SOURce:]SAFEty:STEP<n>:OSC:GET

[SOURce:]SAFEty:STEP<n>:OSC:CHANnel[:HIGH]

The [SOURce:]SAFEty:STEP<n>:OSC:CHANnel[:HIGH] command sets the status of output channel. The [SOURce:]SAFEty:STEP<n>:OSC:CHANnel[:HIGH]? Query returns the status of output channel.

Command Syntax: [SOURce:]SAFEty:STEP<n>:OSC:CHANnel[:HIGH] (@(ch,ch,..))
where (@(ch,ch,…)) is from 1 to 8

Query Syntax: [SOURce:]SAFEty:STEP<n>:OSC:CHANnel[:HIGH]?

Return format: (@(1,2,3))<NL> (@(0)) means all channels are OFF.

[SOURce:]SAFEty:STEP<n>:OSC:CHANnel:LOW

The [SOURce:]SAFEty:STEP<n>:OSC:CHANnel:LOW command sets the status of output channel. The [SOURce:]SAFEty:STEP<n>:OSC:CHANnel:LOW? Query returns the status of output channel.

Command Syntax: [SOURce:]SAFEty:STEP<n>:OSC:CHANnel:LOW (@(ch,ch,..))
where @(ch1,ch2,ch3) is from ch1 to ch12

Query Syntax: [SOURce:]SAFEty:STEP<n>:OSC:CHANnel:LOW?

Return format: (@(1,2,3))<NL> (@(0)) means all channels are OFF.

2.10 PA subsystem commands

[SOURce:]SAFEty:STEP<n>:PAuse[:MESSage]

The [SOURce:]SAFEty:STEP<n>:PAuse[:MESSage] command sets the message strings.
The [SOURce:]SAFEty:STEP<n>:PAuse[:MESSage]? Query returns the message string.

Command Syntax: [SOURce:]SAFEty:STEP<n>:PAuse[:MESSage] “message”

Note: where message must be double quotes

Query Syntax: [SOURce:]SAFEty:STEP<n>:PAuse[:MESSage]?

Return format: <”message”><NL>

[SOURce:]SAFEty:STEP<n>:Pause:UTSIgnal

The [SOURce:]SAFEty:STEP<n>:Pause:UTSIgnal command sets the status of under test signal.
The [SOURce:]SAFEty:STEP<n>:Pause:UTSIgnal? Query returns the status of under test signal.

Command Syntax: [SOURce:]SAFEty:STEP<n>:Pause:UTSIgnal {{1 | ON} | {0 | OFF}}

Query Syntax: [SOURce:]SAFEty:STEP<n>:Pause:UTSIgnal?
Return format: {0 | 1}<NL>

[SOURce:]SAFEty:STEP<n>:Pause:TIME
The [SOURce:]SAFEty:STEP<n>:Pause:TIME command sets the time of PA test.
The [SOURce:]SAFEty:STEP<n>:Pause:TIME? Query returns the time of PA test.
Command Syntax: [SOURce:]SAFEty:STEP<n>:Pause:TIME <time>
Where time is NR3 format
Query Syntax: [SOURce:]SAFEty:STEP<n>:Pause:TIME?
Return format: <NR3><NL>
For example,
SAFEty:STEP5:PA:TIME 5 means set the time of PA test for STEP 5 as 5s.
SAFEty:STEP5:PA:TIME? returns 5.000000E+00, means the time of PA test for STEP 5 is 5s.

2.11 PRESet subsystem commands

[SOURce:]SAFEty:PRESet:TIME:PASS
The [SOURce:]SAFEty:PRESet:TIME:PASS command sets the time of buzzer sound when PASS.
The [SOURce:]SAFEty:PRESet:TIME:PASS? Query returns the time of buzzer sound.
Command Syntax: [SOURce:]SAFEty:PRESet:TIME:PASS <time>
Where <time> is NR3 format, from 0.2s to 99.9s
Query Syntax: [SOURce:]SAFEty:PRESet:TIME:PASS?
Return format: <NR3><NL>

[SOURce:]SAFEty:PRESet:TIME:STEP
The [SOURce:]SAFEty:PRESet:TIME:STEP command sets the time of two steps.
The [SOURce:]SAFEty:PRESet:TIME:STEP? Query returns the time of two setps.
Command Syntax: [SOURce:]SAFEty:PRESet:TIME:STEP {time | KEY}
Where <time> is NR3 format, from 0.0s to 99.9s
Query Syntax: [SOURce:]SAFEty:PRESet:TIME:STEP?
Return format: {NR3 | KEY}<NL>

[SOURce:]SAFEty:PRESet:TIME:SDELay

The [SOURce:]SAFEty:PRESet:TIME:SDELay command sets the delay time before starting test.

The [SOURce:]SAFEty:PRESet:TIME:SDELay? Query returns the delay time before starting test.

Command Syntax: [SOURce:]SAFEty:PRESet:TIME:SDELay <time>

Where <time> is NR3 format, from 0.0s to 99.9s

Query Syntax: [SOURce:]SAFEty:PRESet:TIME:SDELay?

Return format: <NR3><NL>

[SOURce:]SAFEty:PRESet:RJUDgment

The [SOURce:]SAFEty:PRESet:RJUDgment command sets the rising time discriminating function switch.

The [SOURce:]SAFEty:PRESet:RJUDgment? Query returns the status of the rising time discriminating function switch.

Command Syntax: [SOURce:]SAFEty:PRESet:RJUDgment {{1 | ON} | {0 | OFF}}

Query Syntax: [SOURce:]SAFEty:PRESet:RJUDgment?

Return format: {0 | 1}<NL>

[SOURce:]SAFEty:PRESet:AC:FREQuency

The [SOURce:]SAFEty:PRESet:AC:FREQuency command set the frequency of AC.

The [SOURce:]SAFEty:PRESet:AC:FREQuency? Query returns the frequency of AC.

Command Syntax: [SOURce:]SAFEty:PRESet:AC:FREQuency {50 | 60}

Query Syntax: [SOURce:]SAFEty:PRESet:AC:FREQuency?

Return format: {50 | 60}<NL>

[SOURce:]SAFEty:PRESet:WRANge[:AUTO]

The [SOURce:]SAFEty:PRESet:WRANge[:AUTO] command sets the auto range of Hipot test.

The [SOURce:]SAFEty:PRESet:WRANge[:AUTO]? Query returns the status of the range.

Command Syntax: [SOURce:]SAFEty:PRESet:WRANge[:AUTO] {{1 | ON} | {0 | OFF}}

Query Syntax: [SOURce:]SAFEty:PRESet:WRANge[:AUTO]?

Return format: {0 | 1}<NL>

[SOURce:]SAFEty:PRESet:DAGC

The [SOURce:]SAFEty:PRESet:DAGC command sets the switch of AGC function.

The [SOURce:]SAFEty:PRESet:DAGC? Query returns the status of AGC function.

Command Syntax: [SOURce:]SAFEty:PRESet:DAGC {{1 | ON} | {0 | OFF}}

Query Syntax: [SOURce:]SAFEty:PRESet:DAGC?

Return format: {0 | 1}<NL>

[SOURce:]SAFEty:PRESet:GFI[:SWITch]

The [SOURce:]SAFEty:PRESet:GFI[:SWITch] command sets the switch of GFI function.

The [SOURce:]SAFEty:PRESet:GFI[:SWITch]? Query returns the status of GFI function.

Command Syntax: [SOURce:]SAFEty:PRESet:GFI[:SWITch] {{1 | ON} | {0 | OFF}}

Query Syntax: [SOURce:]SAFEty:PRESet:GFI[:SWITch]?

Return format: {0 | 1}<NL>

[SOURce:]SAFEty:PRESet:FAIL[:OPERation]

The [SOURce:]SAFEty:PRESet:FAIL[:OPERation] command sets the action after a test fails.

The [SOURce:]SAFEty:PRESet:FAIL[:OPERation]? Query returns the action after a test fails.

Command Syntax:

[SOURce:]SAFEty:PRESet:FAIL[:OPERation] { STOP|CONTinue|RESTart|NEXT }

Weher ,

STOP means stop test

CONTinue means continue the test

RESTart means press START key start the current step again after a test fails

NEXT means press START key start the next step after a test fails

Query Syntax: [SOURce:]SAFEty:PRESet:FAIL[:OPERation]?

Return format: {STOP|CONTINUE|RESTART|NEXT}<NL>

[SOURce:]SAFEty:PRESet:ARC[:MODE]

The [SOURce:]SAFEty:PRESet:ARC[:MODE] command sets the mode of ARC function.

The [SOURce:]SAFEty:PRESet:ARC[:MODE]? Query returns the mode of ARC function.

Command Syntax: [SOURce:]SAFEty:PRESet:ARC[:MODE] { CURRent|LEVel }

Weher,

CURRent means current mode

LEVel means level mode

Query Syntax: [SOURce:]SAFEty:PRESet:ARC[:MODE]?
Return format: {CURRENT|LEVEL}<NL>

2.12 Mass MEMory subsystem commands

The Mass MEMory subsystem commands load and store files. Figure 2-14-1 shows the Mass MEMory subsystem command tree.

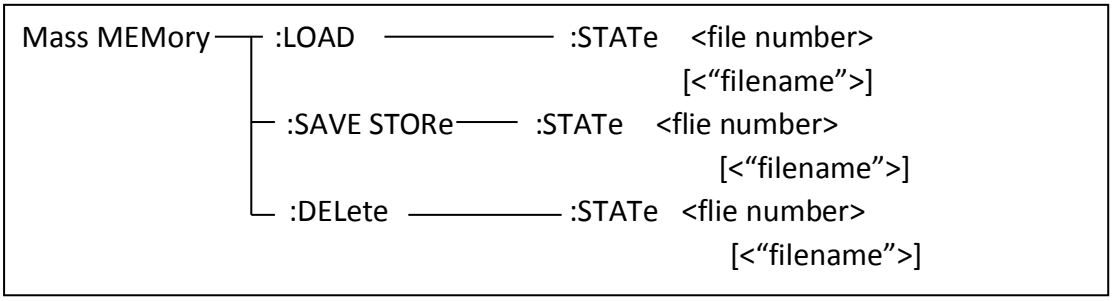


Figure 2-14-1

ⓘ NOTE: The Mass MEMory subsystem commands will be ignored in the phase of testing.

MMEMory:LOAD:STATe
The MMEMory:LOAD:STATe command is used to load the stored file.
Command Syntax: MMEMory:LOAD:STATe <flie number>
Where,
<file number> is the file serial number ranging from 1 to 600 without unit.
For example: WrtCmd("MMEM:LOAD:STAT 1"); load file 1.

ⓘNOTE: 1. If the file you want to load is not available, "File not exist" message will be displayed on the system message line.
2. If the input file number is out of 1 to 600, message "Out of file range" will be displayed on the system message line.

Command Syntax: MMEMory:LOAD:STATe <"filename">
The command is used to find and load the file using the file name directly.

MMEMory:SAVE:STATe or **MMEMory :STORe:STATe**
The MMEMory:SAVE:STATe or MMEMory :STORe:STATe command is used to save the current setting data to a file.

Command Syntax: MMEMory:STORe:STATe <flie number> [,<“filename”>]

Where,

<file number> is the file serial number ranging from 1 to 600 , NR1 format without unit.

<“filename”> The file name consists of less than 20 ASCII characters. <Unnamed> will be the default name, if you don’t input a file name.

For example: WrtCmd(“MMEM:STOR:STAT 1，“#U9311*””)；

ⓘ NOTE: U9311 will not give a warning message when the existent file is to be over written.

☞NOTE：The file name assigned by bus will be quoted without any change, thus user can enter some special characters such as special symbols and letters in lower case that cannot be input on the panel of the instrument.

MMEMory:DELete:STATe

The MMEMory:DELete:STATe command deletes a file.

Command Syntax: MMEMory:DELete:STATe <file number>

Where,

<file number> is the file serial number ranging form 1 to 600, NR1 format without unit.

For example: WrtCmd(“MMEM:DEL:STAT 1”); delete file 1.

ⓘ NOTE: U9311 will not give a warning message when a file is to be deleted.

Command Syntax: Mass MEMory:DELete:STATe “filename”

This command is used to delete a file using the filename directly.

2.13 KEY subsystem commands

KEY commands are used to control the keys and knobs on the operation panel of U9311.

KEY:LOCal	Exit the remote control status
KEY:KEYLock	enable/lock the front panel operation
KEY:MEASure	enter into the <MEAS DISP> page
KEY:SETup	enter into the <MEAS SETUP> page
KEY:SYSTem	enter into the <SYSTEM > page
KEY:UPPer	upper the cursor
KEY:DOWN	down the cursor
KEY:LEFT	left the cursor
KEY:RIGHt	right the cursor
KEY:NUM<n>	numeric key， n range from 0 to 9
KEY:DOT	decimal point key
KEY:SIGN	minus key
KEY:ENTer	enter key
KEY:BACKsapce	backspace key
KEY:ESC	escape key
KEY:HARD	save key
KEY:F<n>	soft key, n is from 1 to 6

2.14 SYSTem subsystem commands

The SYSTem subsystem commands are used for system setups.

SYSTem:SDATe

The SYSTem:SDATe command sets the date of system. The SYSTem:SDATe? query returns the current date.

Command Syntax: SYSTem:SDATe <year>,<month>,<day> year,month,day is NR1 format.

Where, month can be string format：{JANuary | FEBruary | MARch |APRil |MAY | JUNe | JULy | AUGust | SEPtember | OCTober | NOVember | DECember}

Query Syntax: SYSTem:SDATe?

Return format: <NR1>,<NR1>,<NR1><NL>

SYSTem:STIMe

The SYSTem:STIMe command sets the time of system. The SYSTem:STIMe? query returns the current time.

Command Syntax: SYSTem:STIMe <hour>,<minute>,<second> hour, minute, second is NR1 format.

Query Syntax: SYSTem:STIMe?

Return format: <NR1>,<NR1>,<NR1><NL>

SYSTem:STYPe

The SYSTem:STYPe command sets the type of files. The SYSTem:STYPe? query returns the current type of files.

Command Syntax: SYSTem:STYPe { CSV | GIF | BMP | PNG }

Query Syntax: SYSTem:STYPe?

Return format: {CSV | GIF | BMP | PNG }<NL>

SYSTem:PRESet

The SYSTem:PRESet command is used to reset system setting parameters(does not include password, date and time).

Command Syntax: SYSTem:PRESet

SYSTem:BEEPer:PASS[:MODE]

The SYSTem:BEEPer:PASS[:MODE] command sets the alarm of passed judgment. The SYSTem:BEEPer:PASS[:MODE]? query returns the current status.
Command Syntax: SYSTem:BEEPer:PASS[:MODE] {CONTinue | GAP}
Where CONTinue means alarm sound is continue
GAP means alarm sound is gap(interval).
Query Syntax: SYSTem:BEEPer:PASS[:MODE]?
Return format: {CONT | GAP}<NL>

SYSTem:BEEPer:PASS:VOLume

The SYSTem:BEEPer:PASS:VOLume command sets the alarm volume of passed judgment. The SYSTem:BEEPer:PASS:VOLume? query returns the current status.
Command Syntax: SYSTem:BEEPer:PASS:VOLume {OFF | LOW | MEDium | HIGH}
Query Syntax: SYSTem:BEEPer:PASS:VOLume?
Return format: {OFF | LOW | MED | HIGH}<NL>

SYSTem:BEEPer:FAIL[:MODE]

The SYSTem:BEEPer:FAIL[:MODE] command sets the alarm of failed judgment. The SYSTem:BEEPer:FAIL[:MODE]? query returns the current status.
Command Syntax: SYSTem:BEEPer:FAIL[:MODE] {CONTinue | GAP}
Where CONTinue means alarm sound is continue
GAP means alarm sound is gap(interval).
Query Syntax: SYSTem:BEEPer:FAIL[:MODE]?
Return format: {CONT | GAP}<NL>

SYSTem:BEEPer:FAIL:VOLume

The SYSTem:BEEPer:FAIL:VOLume command sets the alarm volume of failed judgment. The SYSTem:BEEPer:FAIL:VOLume? query returns the current status.
Command Syntax: SYSTem:BEEPer:FAIL:VOLume {OFF | LOW | MEDium | HIGH}
Query Syntax: SYSTem:BEEPer:FAIL:VOLume?
Return format: {OFF | LOW | MED | HIGH}<NL>

SYSTem:STYLe:SKIN

The SYSTem:STYLe:SKIN command sets the display color. The SYSTem:STYLe:SKIN? Query returns the current display color.
Command Syntax： SYSTem:STYLe:SKIN {GRAY | BLACk | BLUE | CYAN}
Query Command: SYSTem:STYLe:SKIN?
Return Format： {GRAY | BLACK | BLUE | CYAN}<NL>

SYSTem:STYLe:LANGuage

The SYSTem:STYLe:LANGuage command sets the display language. The SYSTem:STYLe:LANGuage? query returns the current display language.

Command Syntax： SYSTem:STYLe:LANGuage {ENglish | Chinese}

Query Command: SYSTem:STYLe:LANGuage?

Return Format： {EN | CH}<NL>

SYSTem:STYLe:KSOUnd

The SYSTem:STYLe:KSOUnd command sets the key sound. The SYSTem:STYLe:KSOUnd? query returns the current state of key sound.

Command Syntax： SYSTem:STYLe:KSOUnd {{1 | ON} | {0 | OFF}}

Query Command: SYSTem:STYLe:KSOUnd?

Return Format： {1 | 0}<NL>

3 Error and warning message

The bus commands may have some spelling errors, syntax errors or wrong parameters. U9311 executes a command after the command is analyzed. If one of above errors occurs, U9311 halts the command analysis, and the rest commands will be ignored. If a command (for example a trigger command is ignored.) is ignored, the rest commands will be executed. The error and warning messages will be displayed on the system message line.

The following table shows the common error and warning messages, which will be displayed on the message line when they occur.

Error message	Description
Undefined message	Unknown command is received. Usually there is a spelling error in the command. For example: TRG should be TRIG DISP:PAG MEAS should be DISP:PAGE MEAS
Data out of range	The data is out of range. For example: SAFE:STEP1:AC:LEV 6000, the AC output voltage is out of its range.
Invalid parameter	Unrecognizable parameter is used. For example: SAFE:PRES:AC:FREQ 100， 100 is not the correct frequency and should not used.
Invalid suffix	Units are unrecognizable, or the units are not correct. For example: SAFE:STEP1:AC:TIME:RAMP 1000us， us can not be the unit of the high voltage.
Data too long	Data is too long. For example: The number of characters for a file name can not exceed 20 characters and numeric parameter, 20 characters.
Syntax error	Error syntax, for example:DISP.PAGE TSET, where(.)should be (:) .
Command ignored	Some command may be ignored. For example: DISP:PAGE MSET When U9311 is in the testing state, this command will be ignored.

4 Programming Examples

This chapter lists three programming examples in the development environments of Visual C++ 6.0, Visual Basic 6.0 and LabVIEW 8.5. All the examples are based on VISA (Virtual Instrument Software Architecture).

VISA is an API (Application Programming Interface) used for controlling instruments. It is convenient for users to develop testing applications which are independent of the types of instrument and interface. Note that “VISA” here we mention is NI (National Instrument)-VISA. NI-VISA is an API written by NI based on VISA standard. You can use NI-VISA to achieve the communication between the oscilloscope and PC via GPIB, USB, RS232, LAN and such instrument bus. As VISA has defined a set of software commands, users can control the instrument without understanding the working state of the interface bus. See NI-VISA User Manual and NI-VISA Programmmer Reference Manual for more information about NI-VISA API.

A typical application of VISA contains the following parts:

1. Set up the conversation for the existing resource
2. Configure the resource (such as: Baud rate)
3. Close the conversation

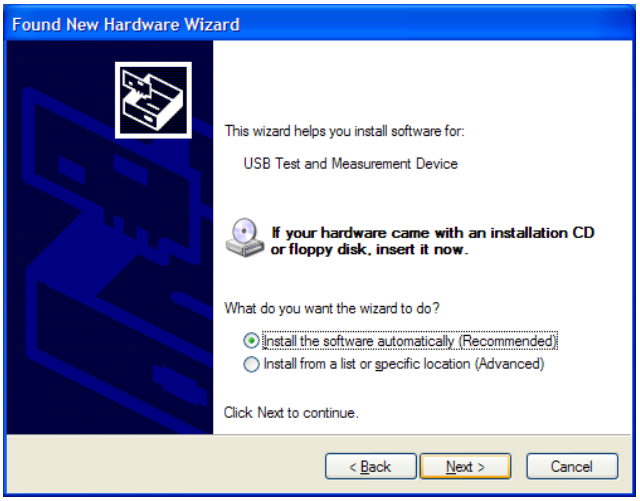
Preparation for Programming

Download NI-VISA software from http://www.ni.com to install it. The installing path is C:\Program Files\IVI Foundation\VISA.

Take U9311 as an example to show how to construct the communication between an hipot tester and a PC. Use a USB cable with one teminal connecting the DEVICE interface on the rear panel of the instrument and the other one connecting the USB interface of PC, as is shown in figure 4-1.

Figure 4-1

Switch the instrument power on. An upgrading guide dialoge will pop up and you can install USB Test and Measurement Device software by prompt information.



4.1 Visual C++ 6.0 Programming Example

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

1. Create a project based on MFC.

2. Choose Project→Settings→C/C++; select **"**Code Generation**"** in Category and **"**Debug Multithreaded DLL**"** in Use run-time library; click OK; as is shown in figure 4-1-1.

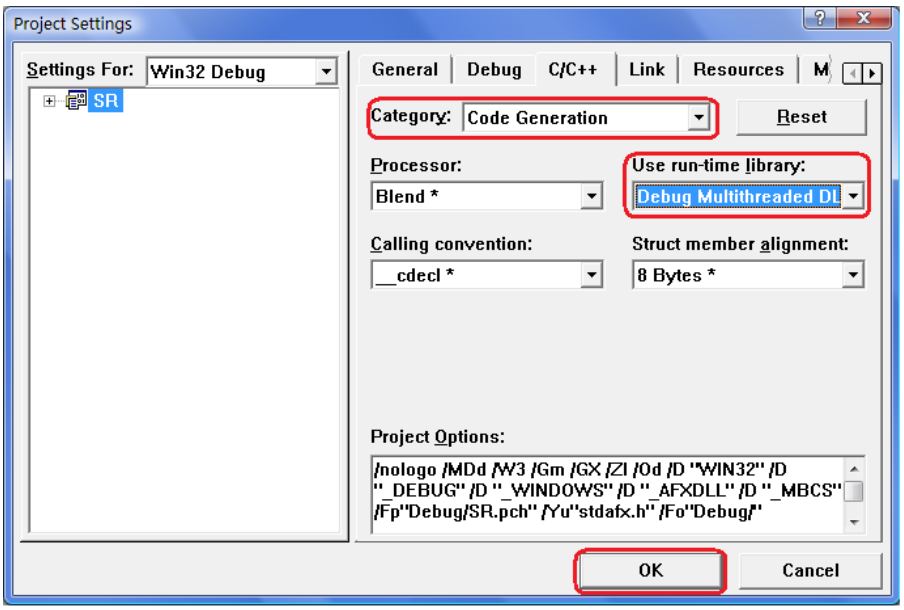


Figure 4-1-1

1. Choose Project-> Settings-> Link, add the file visa32.lib manually in Object/library modules; click OK; as is shown in figure 4-1-2.
2. Choose Tools->Options ->Directories; select Include files in Show directories for, and then double click the blank in Directories to add the path of Include: C:\Program Files\IVI Foundation\VISA\WinNT\include, as is shown in figure 4-1-3.

Select Library files in Show directories for, and then double click the blank in Directories to add the path of Lib: C\Program Files\IVI Foundation\VISA\WinNT\lib\msc.

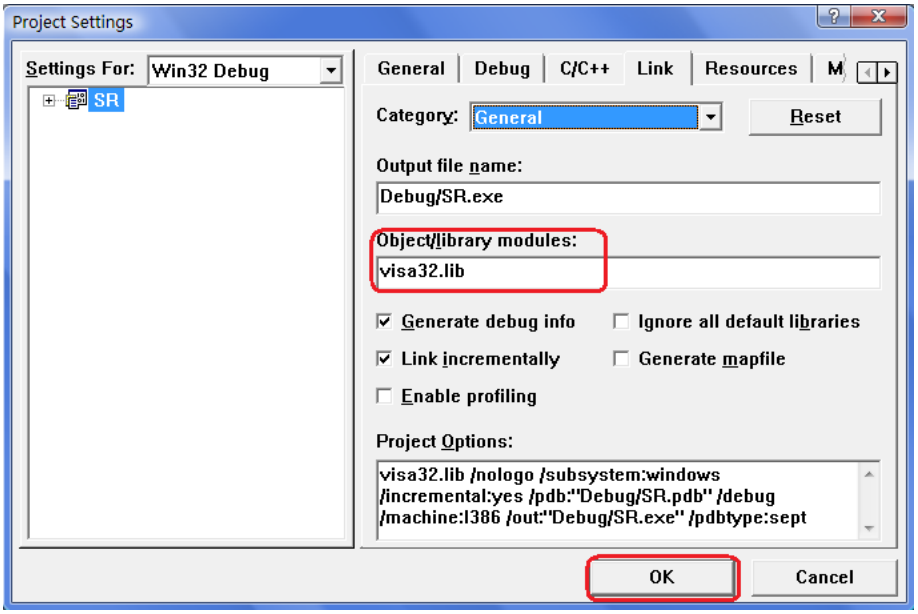


Figure 4-1-2

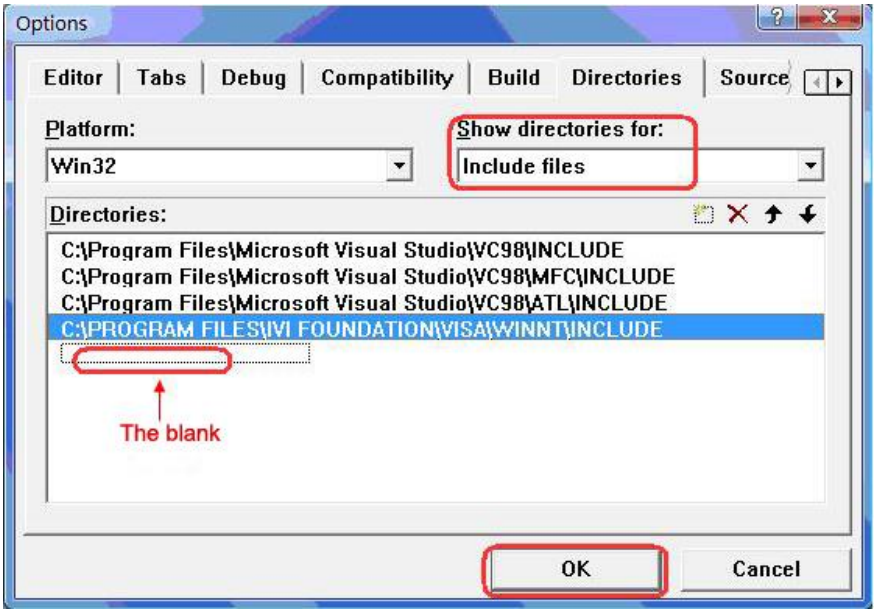


Figure 4-1-3

5. Add controls: Static Text, Edit and Button. See figure 4-1-4.

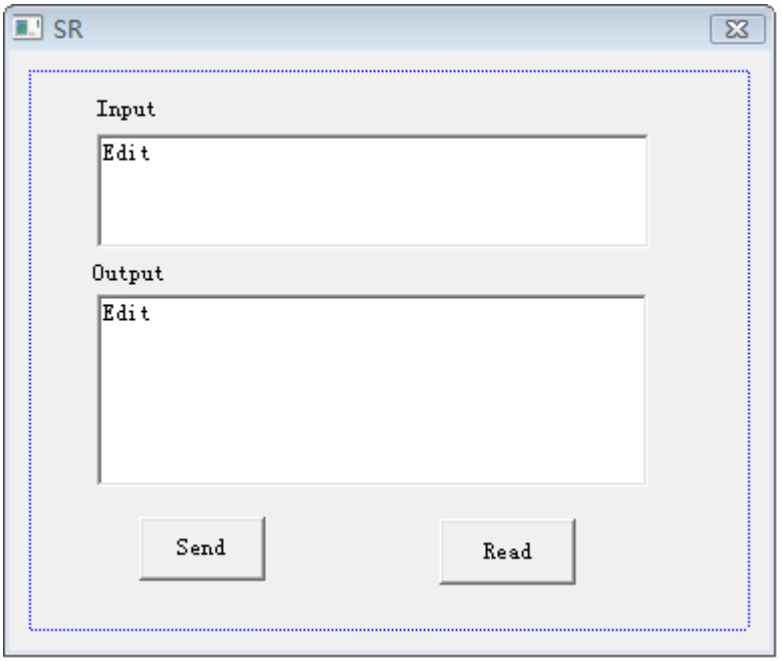


Figure 4-1-4

(1) Add two Static Text controls respectively named as Input and Output.

(2) Add two Edit controls, and then add two variables--m_send and m_read to them respectively. See figure 4-1-5 and figure 4-1-6.

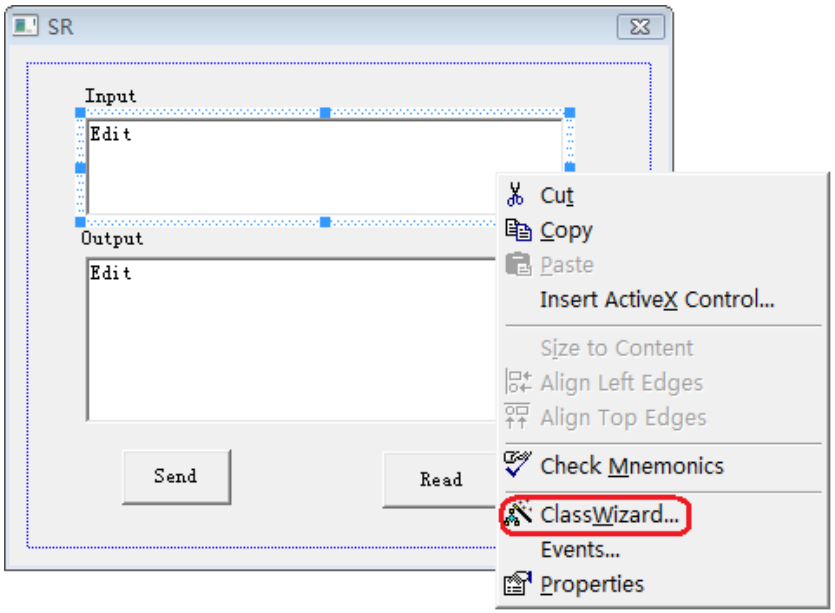


Figure 4-1-5

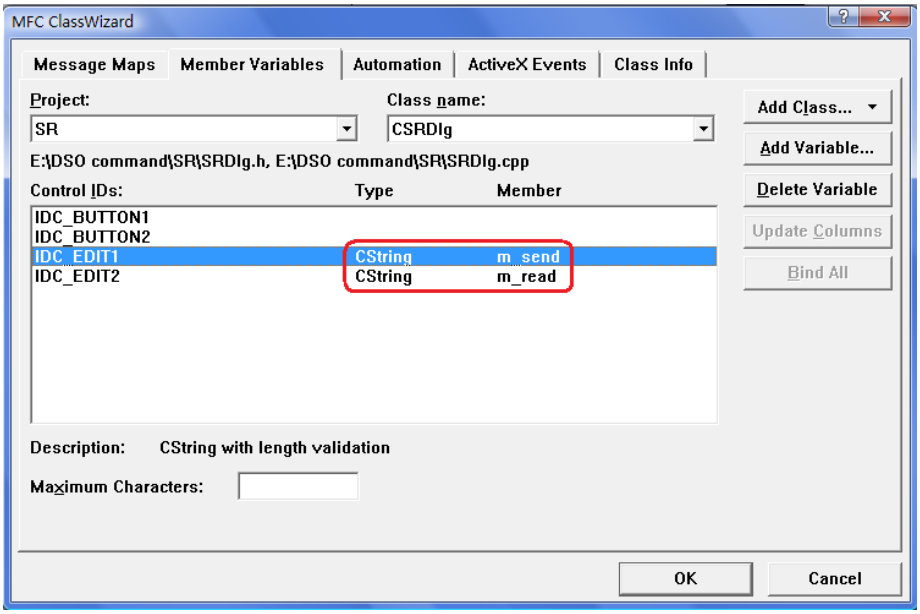


Figure 4-1-6

(3) Add two Button controls named as Send and Read respectively.

6. Double click Send, enter the programming environment.

(1) Declare “#include“visa.h”” in header file.

(2) Define relative variables and then add the following codes:

```
ViSession defaultRM, vi;
char buf [256] = {0};
CString s,strTemp;
char* stringTemp;
ViChar buffer [VI_FIND_BUFLEN];
ViRsrc matches=buffer;
ViUInt32 nmatches;
ViFindList list;
```

(3) In ::CSRDlg(CWnd* pParent /*=NULL*/)

```
: CDialog(CSRDlg::IDD, pParent), order m_send = _T("*IDN?\n");
```

(4) Add the following codes to ::OnInitDialog().

```
viOpenDefaultRM (&defaultRM);
//acquire USB resource of visa
viFindRsrc(defaultRM, "USB?*", &list,&nmatches, matches);
viOpen (defaultRM,matches,VI_NULL,VI_NULL,&vi);
```

(5) Add the following codes in Send.

```
//send the receiving commands
UpdateData (TRUE);
strTemp = m_send + "\n";
stringTemp = (char *)(LPCTSTR)strTemp;
viPrintf (vi,stringTemp);
```

(6) Add the following codes in Read.

```
//read the result
viScanf (vi, "%t\n", &buf);
//display the results
m_read = buf;
UpdateData (FALSE);
```

(7) Add the following codes in ::OnQueryDragIcon().

```
//close resource.
viClose (vi);
viClose (defaultRM);
```

7. Save, build and run the project, you will get an EXE file. When the oscilloscope has been successfully connected with PC, input a command such as *IDN? (the default input command) in Input edit box and cilck Send and Read successively, the oscilloscope will return the result which will be displayed in Output edit box. See figure 4-1-7.

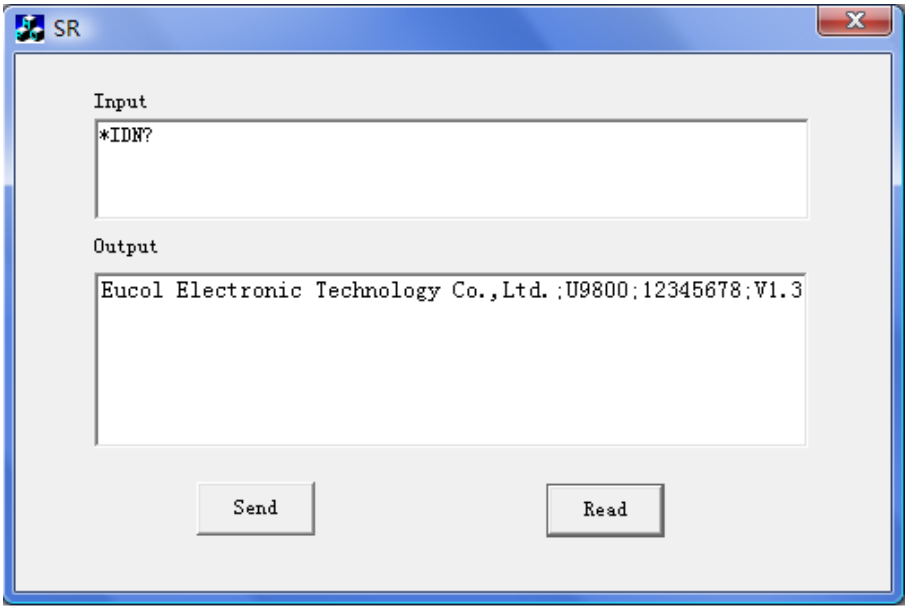

SR
Input
*IDN?
Output
Eucol Electronic Technology Co.,Ltd.:U9800:12345678:V1.3
Send
Read

Figure 4-1-7

4.2 Visual Basic 6.0 Programming Examples

Open Visual Basic6 6.0, take the following setps:

1. Create a Standard EXE project.

2. Choose Project-> Add Module->Existing; find the visa32.bas file in the Add Module under the path of NI-VISA: C:\Program Files\IVI Foundation\VISA\WinNT\include, and then add it. See figure 4-2-1.

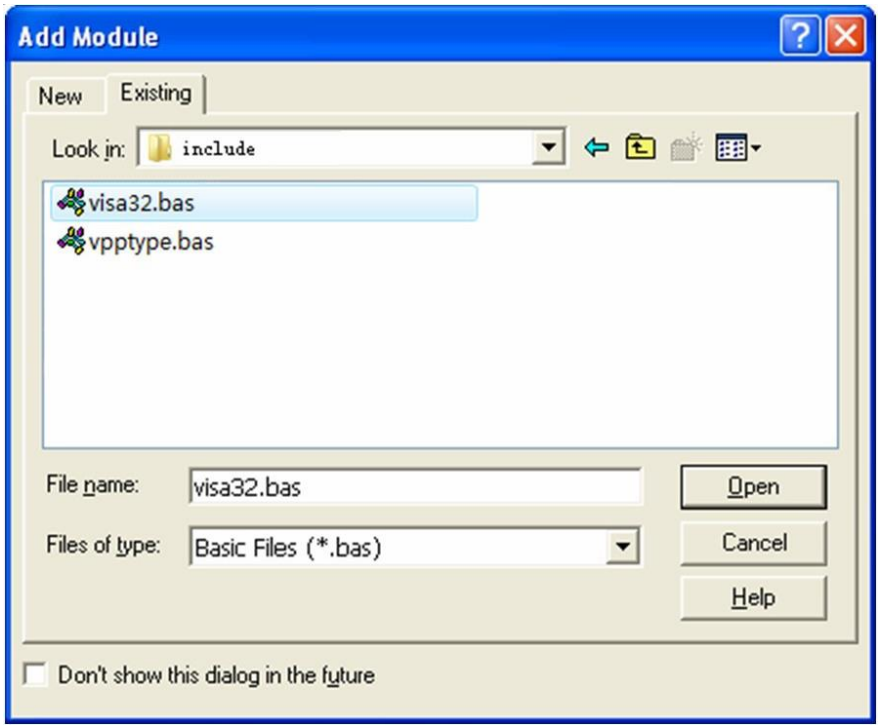


Figure 4-2-1

3. Add two Lables respectively named as Input and Output, two TexBox and two CommandButtons named as Send and Read seperately. Set Text in the attribute of TextBox under Input as *IDN?. See figure 4-2-2.

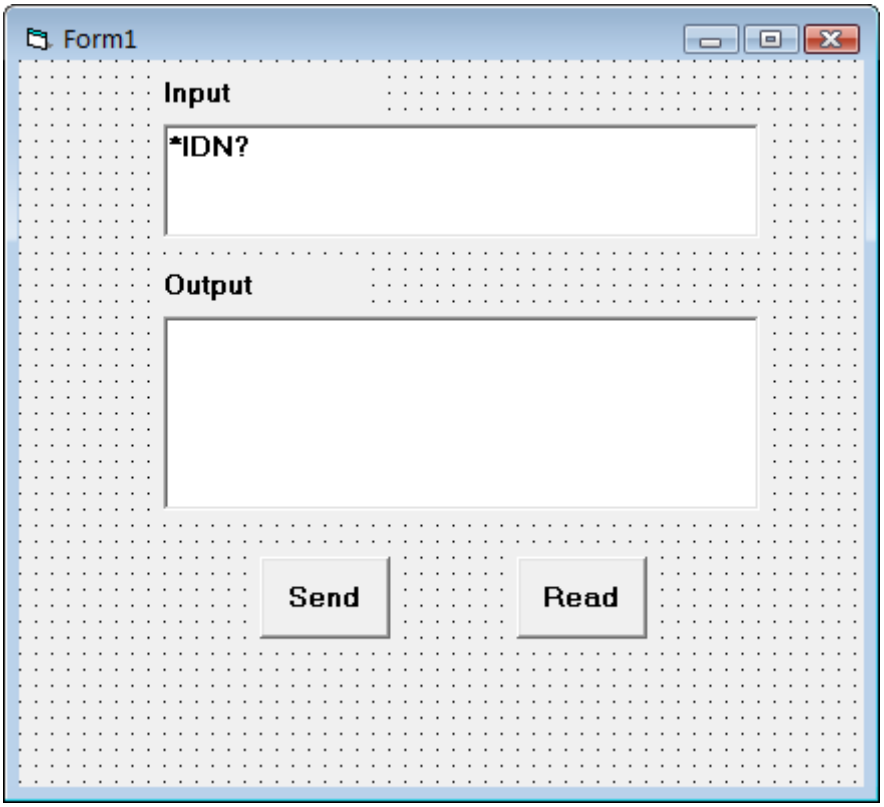


Figure 4-2-2

4. Choose Project->Project1 Properties->General, Select Form1 form the drop down box of Startup Object.

5. Double click Send, enter the programming environment and add the following codes:

```
Dim defrm As Long
Dim vi As Long
Dim list As Long
Dim nmatches As Long
Dim matches As String * 200 ' reserves to acquire the equipment ID.
Dim strRes As String * 200

Private Sub Cmd_Read_Click()
' acquire the command return state
Call viVScanf(vi, "%t", strRes)
```

```
Txt_output.Text = strRes
End Sub

Private Sub Cmd_Send_Click()
' send the command to query
Call viVPrintf(vi, Txt_input.Text + Chr$(10), 0)
End Sub

Private Sub Form_Load()
' acquire the usb source of visa
Call viOpenDefaultRM(defrm)
Call viFindRsrc(defrm, "USB?*", list, nmatches, matches)
' open the device
Call viOpen(defrm, matches, 0, 0, vi)
End Sub

Private Sub Form_Unload(Cancel As Integer)
' close the resource
Call viClose(vi)
Call viClose(defrm)
End Sub
```

6. Save and run the project, you will get a single executable program. When the oscilloscope has been successfully connected with PC, you can input a command such as *IDN? (the default input command) in Input edit box and cilck Send and Read successively, the oscilloscope will return the result which will be displayed in Output edit box. See figure 4-2-3.

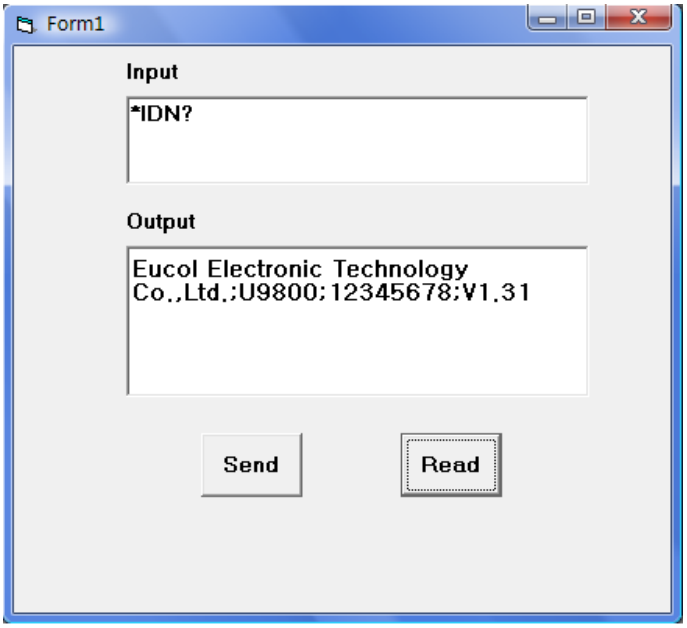

Form1
Input
*IDN?
Output
Eucol Electronic Technology Co.,Ltd.;U9800;12345678;V1.31
Send
Read

Figure 4-2-3

4.3 LabVIEW 8.5 Programming Examples

Run LabVIEW8.5, take the following steps.

1. Enter Getting Started, choose New>>Blank VI to create a new VI.

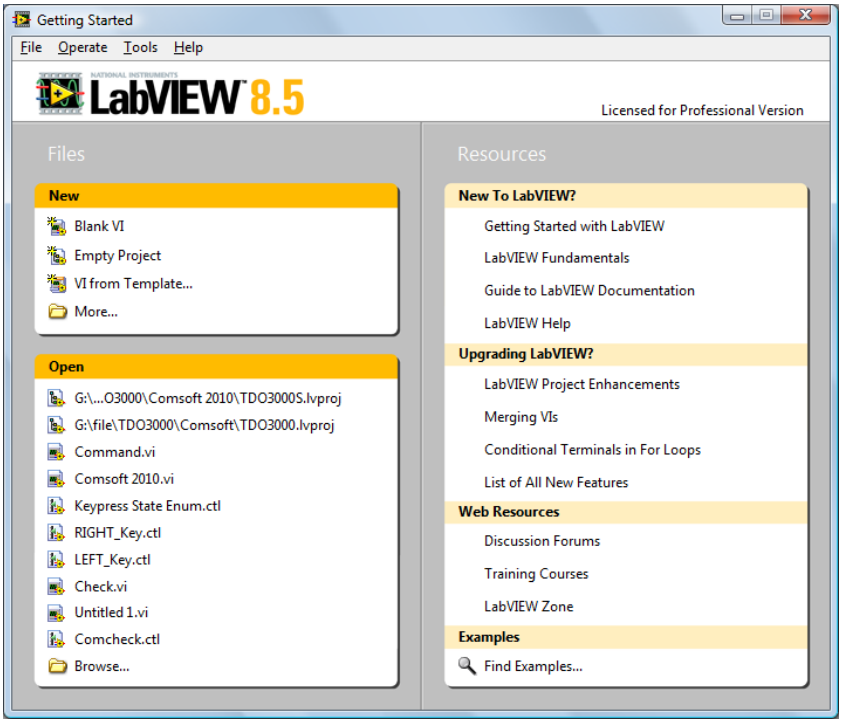


Figure 4-3-1

2. Right-click the Front Panel to choose Controls>>Modern>>Boolean>>OK Button; add three buttons and respectively define them as Write, Read and Stop. See figure 4-3-2.

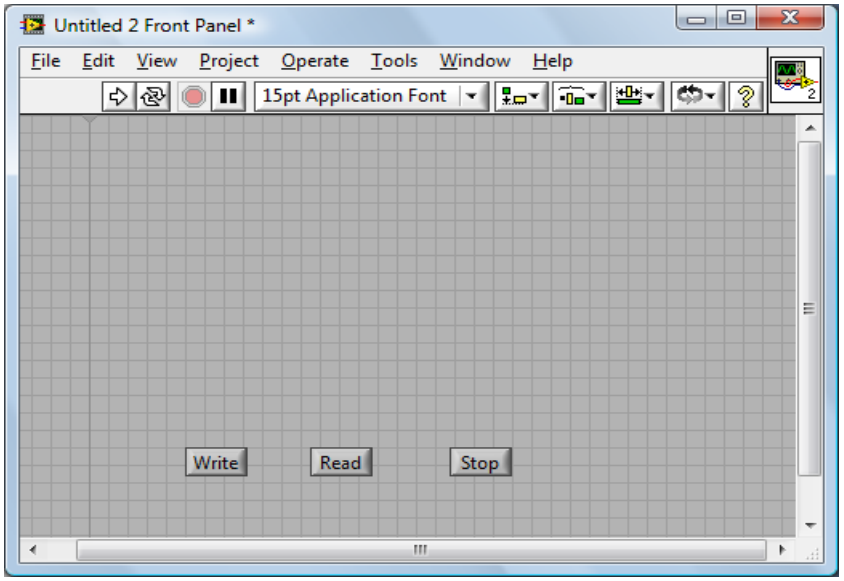


Figure 4-3-2

3. Open the Block Diagram, right-click it and choose Functions>> Programming>> Structure>> Event Structure to add an event structure.

4. Open the Block Diagram; right-click the event structure to choose Add Event Case…; Add the Value Change event for each control; drag all terminals into their own event structure.

5. Choose the Value Change event structure of the Write terminal; right-click the blank of the Block Diagram to select Functions>>Instrument I/O>>VISA>>VISA Write; add a VISA Write function for the Value Change event structure of the Write terminal.

6. Right-click the Block Diagram to choose Functions>>Instrument I/O>>VISA>>VISA Advanced>> VISA Open; add a VISA Open function on the left side of the Write structure event.

7. Right-click the VISA resource name terminal of the VISA Open function; click the shortcut menu and select Create>>Control to create a VISA resource name.

8. Wire the VISA resource out terminal of the VISA Open function to the VISA resource name terminal of the VISA Write function in the event structure; Connect the error out terminal of the VISA Open function with the error in terminal of the VISA Write function.

9. Right-click the write buffer terminal of the VISA Writer function; click the shortcut menu and choose Create>>Control to create a write buffer as shown in figure 4-3-3.

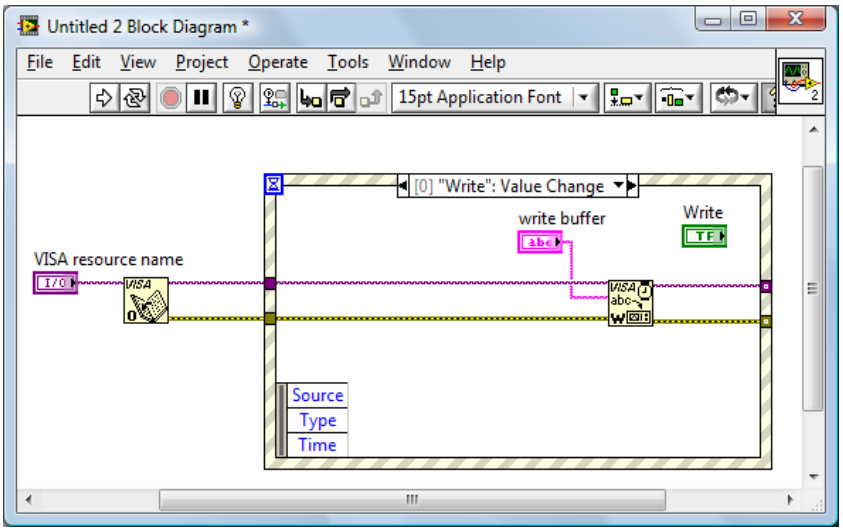


Figure 4-3-3

10. Select the Value Change event structure of the Read terminal; right-click Functions>> Instrument I/O>> VISA>> VISA Read to add a VISA Read function into the "Read": Value Change event structure.

11. Right-click the read buffer terminal of the VISA Read function; click the shortcut menu and choose Create>>Indicator to create a read butter.

12. Right-click the byte count terminal of the VISA Read function; click the shortcut menu and choose Create>>Constant to create a constant as 1024.

13. Wire the VISA resource out terminal of the VISA Open function to the VISA resource name terminal of the VISA Read function in the event structure; connect the error out terminal of the VISA Open function with the error in terminal of the VISA Read function shown as figure 4-3-4.

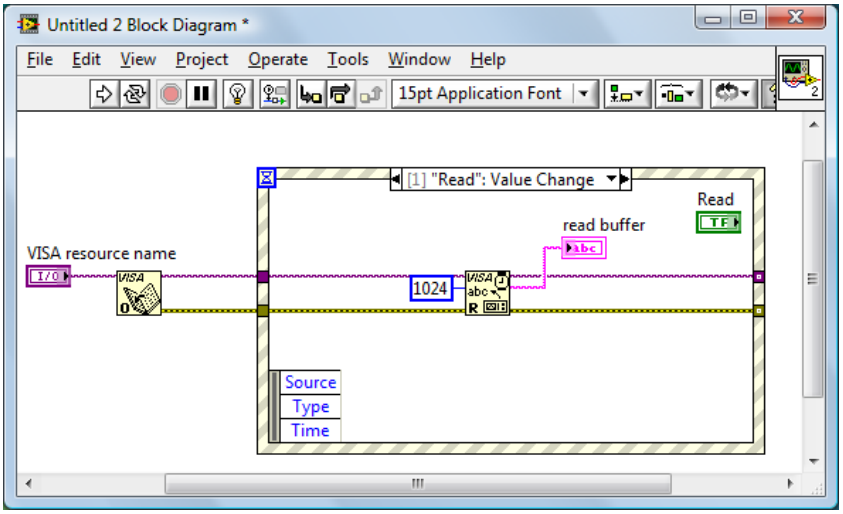


Figure 4-3-4

14. Select the Value Change event structure of the Stop terminal; right-click the blank of the Block Diagram and choose Functions>>Instrument I/O>>VISA>>VISA Advanced>>VISA Close to add a VISA Close function for the "Stop":Value Change event structure.

15. Wire the VISA resource out terminal of the VISA Open function to the VISA resource name terminal of the VISA Close function in the event structure; connect the error out terminal of the VISA Open function with the error in terminal of the VISA Close function shown as figure 4-3-5.

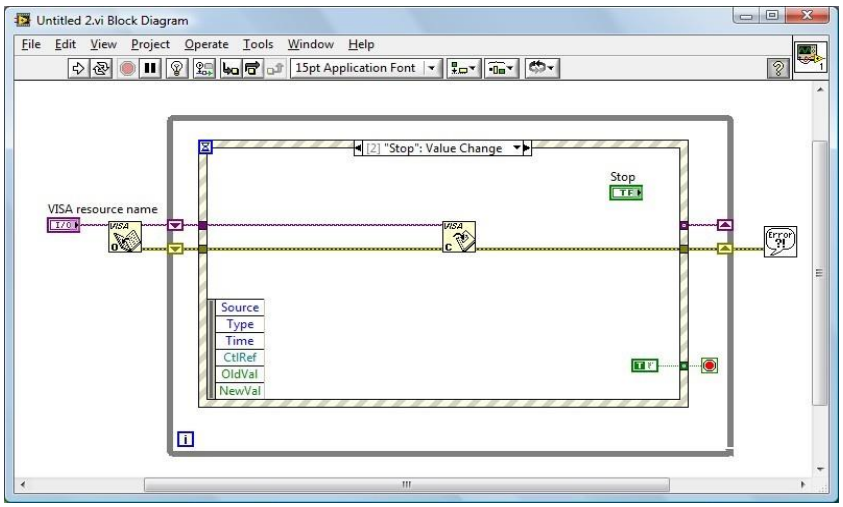


Figure 4-3-5

16. Right-click the blank of the Block Diagram and choose Functions>>Programming>> Structures >>While Loop to add a While Loop structure outside the event structure.

17. Click the Functions palette and choose Functions>>Programming>>Boolean>>True Constant to add a True Constant for the "Stop": Value Change event structure. Wire the True Constant to the stop terminal of the While Loop structure.

18. Click the Functions palette and choose Functions>> Programming>>Dialog& User Interface>> Simpel Error Handler to add a Simple Error Handler function. Wire the error out terminal of the VISA Close function to the error in terminal of the Simple Error Handler function.

19. Right-click the Loop Tunnel terminal where the While Loop structure and the error wire intersected; click the shortcut menu and choose Replace with Shift Register to create a Loop Shift Register Pair with the purpose of replacing the Loop Tunnel. Similarly with a Loop Shift Register Pair to replace the Loop Tunnel where the VISA resource out terminal of the VISA Open function and the VISA resource name terminal of the VISA Close function interested.

20. Adjust the style of the Front Panel shown as figure 4-3-6.

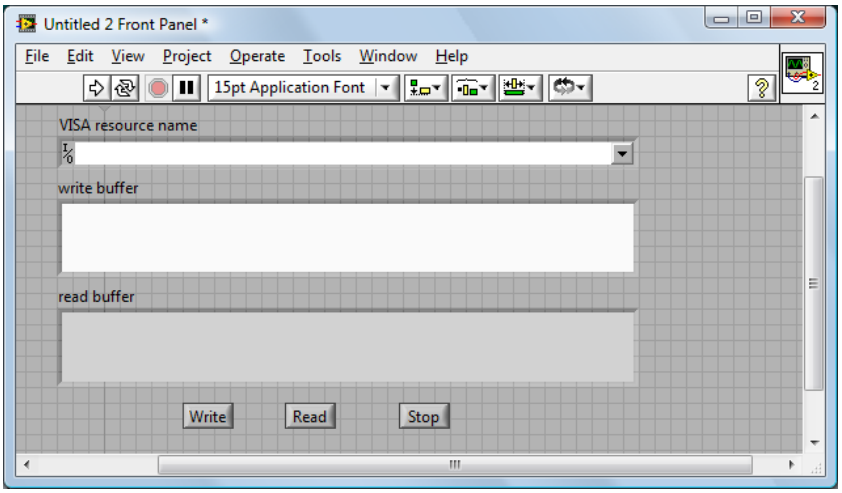


Figure 4-3-6

21. Save the current VI. Before running this VI, select the correct VISA resource name form the VISA resource name pull-down menu.

22. Run the current VI. Input your command or query in the writer buffer, for

instance*idn?; click the Write control to send the command or query; then click the Read control to read the returned information. The execution result is shown as figure 4-3-7.

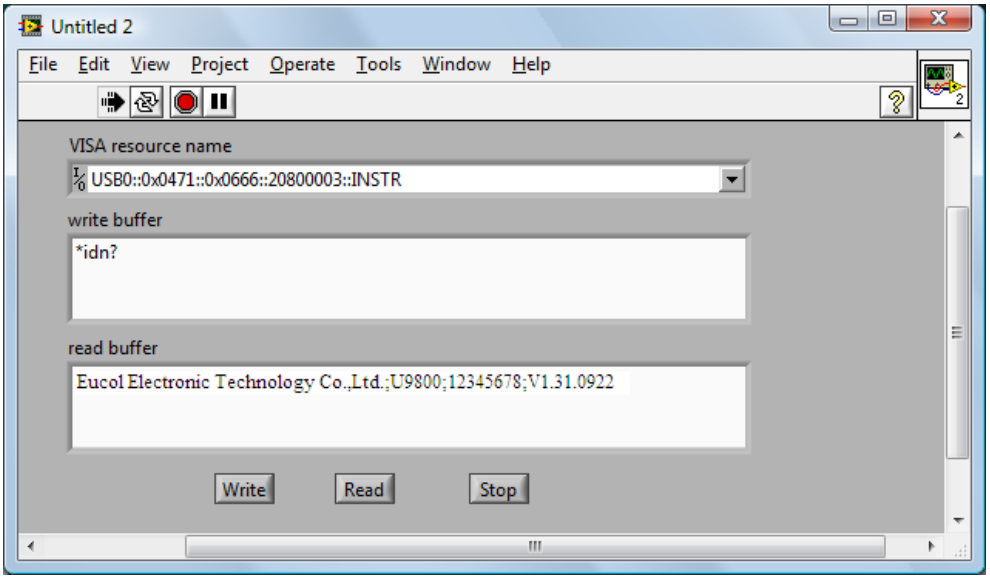


Figure 4-3-7

23. Click the Stop control to exit this program.