RIGOL

Programming Guide

DS1000E, DS1000D
Series Digital Oscilloscope

Sept. 2010

RIGOL Technologies, Inc.

Copyright

© 2009 RIGOL Technologies, Inc. All Rights Reserved.

Trademark Information

RIGOL is 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, DS1000E, DS1000D has passed CE certification.

Content

Chapter 1   Programming Introduction

This chapter provides the introduction about the interfces and commands so as to control DS1000E,DS1000D series digital oscilloscopes via remote commands.

The chapter contains following topics:

- Communication Interfaces
- Command Introduction
  Command Syntax
  Symbol Instruction
  Command Input
  Parameter Type

Communication Interfaces

Computers communicate with the oscilloscope by sending and receiving messages over an I/O port such as an USB or RS-232 port. Commands appear as ASCII character strings embedded inside the output statements of a "host" language available on your computer.

Basic operations that you can do with a computer and an oscilloscope include:

- Set up the oscilloscope.
- Make measurements.
- Retrieve data (waveforms or measurements) from the oscilloscope.

Connection:

- USB: connect the oscilloscope with the computer through an USB data cable.
- RS232: connect the oscilloscope with the computer through an RS232 serial port Cable.

Command Introduction

Command Syntax

The command system of DS1000E, DS1000D series digital oscilloscopes presents a multiscale tree structure, each of the subsystem consists of a "Root" keyword and one or multilayered keywords. Generally, a command line starts with ":" (except for IEEE commands) and ":" is also used for separating different keywords, meanwhile, parameters are permitted to follow the keywords; in addition, "?" after a command line denotes to query its function and "space" is used to divide command and parameter.

For example:
:TRIGger:EDGE:SLOPe {POSitive|NEGative}
:TRIGger:EDGE:SLOPe?

TRIGger is the root keyword of this command, **EDGE** and **SLOPe** are second and third keyword respectively, all of these keywords are separated by ":". Connects enclosed in the "{}" denote the parameters permitted to be set by user; queries are formed by adding a question mark (?) to the end of the commands and "space" is used to divide the command :TRIGger:EDGE:SLOPe and the parameter. For the command with multiple parameters, "," is generally used for separating the parameters, for example:
:TRIGger:DURation:PATTern <value>,<mask>

Symbol Instruction

The following symbols are not "real" parts of the commands, but they are usually used to assist to explain the parameters containd in a command line.

1. **Braces { }**
    The parameters enclosed in a command line must be selected. When several elements separated by a vertical line (|) are enclosed by braces, { }, only one element may be selected, for example:
    :MEASure:TOTal {ON|OFF},
    Thereinto, {ON|OFF} indicates that only ON or OFF may be selected, not both.

2. **Square Brackets [ ]**
    Items enclosed in square brackets [ ] are optional, for example:
    :TIMebase[:DELayed]:OFFSet <offset>
    Thereinto, [:DELayed] could be omitted.

3. **Triangle Brackets < >**
    Items enclosed in < > should be replaced by an effective value, for example:
    :DISPlay:BRIGhtness <ncount>
    Thereinto, < ncount > must be a numerical value such as 25.

Command Input

All the commands of both DS1000E, DS1000D series are case-insensitive. You can use capital letter to input the whole command or you can input the abbreviation. Note that if use abbreviation, the capital letters specified in commands must be written completely. For example:

    :TRIGger:EDGE:SLOPe

can be entered as:

    :TRIG:EDGE:SLOP

Parameter Type

The commands contains 5 kinds of parameters, different parameters have different setting methods.

1. **Boolean**
   The parameter should be "OFF" or "ON", for example:
   :MEASure:TOTal {ON|OFF}
   Thereinto, "ON" denotes trun on (enable) this function and "OFF" denoets turn off (disable).

2. **Consecutive Integer**
   The parameter should be a consecutive integer, for example:
   :DISPlay:BRIGhtness <ncount>
   Thereinto, <ncount> could be an integer between 0 and 32 (including 0 and 32).

3. **Consecutive Real Number**
   The parameters could be any value within effective range and precision permitting, for example:
   :TRIGger:EDGE:SENSitivity <count>
   Thereinto, <count> could be a real number between 0.1 and 1(including 0.1 and 1).

4. **Discrete**
   The parameters can only be the cited value, for example:
   :ACQuire:AVERages <count>
   Thereinto, <count> could be 2, 4, 8, 16, 32, 64, 128, 256.

5. **ASCII Character String**
   The parameter should a combination of ASCII character string, for example:
   :TRIGger:MODE <mode>
   Thereinto, <mode> could be "EDGE", "PULSe", "VIDEO", "SLOPe", "PATTern", "DURation" or "ALTernation".

Chapter 2   Command Systems

This chapter gives detailed information on each command supported by DS1000E, DS1000D series, including command format, function description, using considerations as well as some application examples.

The subcommands systems contain:

● General Commands
● SYSTem Commands
● ACQuire Commands
● DISPlay Commands
● TIMebase Commands
● TRIGger Commands
   Trigger Control
   EDGE Trigger
   PULSe Trigger
   VIDEO Trigger
   SLOPe Trigger
   PATTern Trigger
   DURation Trigger
   ALTernation Trigger
● STORage Command
● MATH Commands
● CHANnel Commands
● MEASure Commands
● WAVeform Command
● LA Commands
● KEY Commands
● Other Commands

General Commands

IEEE standard defineds common commands for querying or executing some basic information about instrument, which usually begins with "*" and holds 3-character long command keywords.

DS1000E, DS1000D series digital oscilloscopes support following **IEEE488.2** commands:

- *IDN?
- *RST

The detailed information of each command are given as follows:

1. *IDN?

Command Format:
*IDN?

Function Explanation:
The command queries the ID character string of the instrument, including a field separated by 4 commas: manufactory, model, serial number and the version number composed of numbers and separated by ".".

Returned Format:
RIGOL TECHNOLOGIES,<model>, <serial number>, <software version number>.

Example:
RIGOL TECHNOLOGIES,DS1102E,DS1EB104702974,00.02.01.01.00

2. *RST

Command Format:
*RST

Function Explanation:
The command resets the system parameters.

SYSTem Commands

SYSTem Commands are the fundamental commands for the operation of an oscilloscope. They can either be used for operational control or screen data interception and more.

SYSTem Commands include:

- :RUN
- :STOP
- :AUTO
- :HARDcopy

The detailed information of each command are given as follows:

1.   :RUN

Command Format:
:RUN

Function Explanation:
The command initiates the oscilloscope to acquire waveform data according to its current settings. Acquisition runs continuously until the oscilloscope receives a :**STOP** command.

2.   :STOP

Command Format:
:STOP

Function Explanation:
The command controls the oscilloscope to stop acquiring data. To restart the acquisition, use the :**RUN** command.

3.   :AUTO

Command Format:
:AUTO

Function Explanation:
The command controls the oscilloscope to evaluate all input waveforms characteristics and set the optimum conditions to display the waveforms.

4.   :HARDcopy

Command Format:
:HARDcopy

Function Explanation:
The command is to extract the current information on the screen and save the bitmap into the U disc in the form of "HardCopyxxx.bmp". (Note: this command is unavailable in file system)

ACQuire Commands

ACQuire Commands set the acquire mode for the oscilloscope, including:

- :ACQuire:TYPE
- :ACQuire:MODE
- :ACQuire:AVERages
- :ACQuire:SAMPlingrate?
- :ACQuire:MEMDepth <depth>

The detailed information of each command are given as follows:

1.   :ACQuire:TYPE

Command Format:
:ACQuire:TYPE <type>
:ACQuire:TYPE?

Function Explanation:
The commands set and query the current acquire type of the oscilloscope.
<type> could be NORMal, AVERage or PEAKdetect.

Returned Format:
The query returns NORMAL, AVERAGE or PEAKDETECT.

Example:
:ACQ:TYPE AVER   Setup the acquire type as Average.
:ACQ:TYPE?   The query returns AVERAGE.

2.   :ACQuire:MODE

Command Format:
:ACQuire:MODE <mode>
:ACQuire:MODE?

Function Explanation:
The commands set and query the current acquire mode of the oscilloscope.
<mode> could be RTIMe (Real time Sampling) or ETIMe (Equivalent Sampling).

Returned Format:
The query returns REAL_TIME or EQUAL_TIME.

Example:
:ACQ:MODE ETIM   Setup the acquire mode as ETIMe.
:ACQ:MODE?   The query returns EQUAL_TIME.

3.   :ACQuire:AVERages

Command Format:

:ACQuire:AVERages <count>
:ACQuire:AVERages?

Function Explanation:

The commands set and query the average numbers in Average mode. <count> could be and integer of 2 times the power of N within 2 and 256.

Returned Format:

The query returns 2, 4, 8, 16, 32, 64, 128 or 256.

Example:

:ACQ:AVER 16   Setup the average acquisition time as 16.
:ACQ:AVER?   The query returns 16.

4.   :ACQuire:SAMPlingrate?

Command Format:

:ACQuire:SAMPlingrate? {CHANnel<n>|DIGITAL}

Function Explanation:

The command queries the current sampling rate of the analog channel or digital channel (only for DS1000D series). <n> is 1 or 2 means channel 1 or channel 2.

Returned Format:

The query returns the setting value of the sampling rate.

Example:

:ACQ:SAMP? CHANnel2   Query the sampling rate for channel 2.
100000000.000000   The query returns 100M.

5.   :ACQuire:MEMDepth <depth>

Command Format:

:ACQuire:MEMDepth <depth>
:ACQuire:MEMDepth?

Function Explanation:

The commands set and query the memory depth of the oscilloscope. <depth> could be LONG (long memory) or NORMal (normal memory)

Returned Format:

The query returns LONG or NORMAL.

Example:

:ACQ:MEMD LONG   Set the memory type as LONG.

:ACQ:MEMD?   The query returns LONG.

DISPlay Commands

DISPlay Commands setup the display type of the oscilloscope.

DISPlay Commands include:

- :DISPlay:TYPE
- :DISPlay:GRID
- :DISPlay:PERSist
- :DISPlay:MNUDisplay
- :DISPlay:MNUStatus
- :DISPlay:CLEar
- :DISPlay:BRIGhtness
- :DISPlay:INTensity

The detailed information of each command are given as follows:

1.   :DISPlay:TYPE

Command Format:
:DISPlay:TYPE <type>
:DISPlay:TYPE?

Function Explanation:
The commands set and query the display type between sampling points. <type>
could be VECTors (vectir display) or DOTS (point display).

Returned Format:
The query returns VECTORS or DOTS.

Example:
:DISP:TYPE VECT   Setup the display type as VECTors.
:DISP:TYPE?   The query returns VECTORS.

2.   :DISPlay:GRID

Command Format:
:DISPlay:GRID <grid>
:DISPlay:GRID?

Function Explanation:
The commands set and query the state of the screen grid. <grid> could be FULL
(open the background grid and coordinates), HALF (turn off the background grid)
or NONE (turn off the background grid and coordinates).

Returned Format:
The query returns FULL, HALF or NONE.

Example:
:DISP:GRID FULL   Open the background grid and coordinates.
:DISP:GRID?   The query returns FULL.

3.   :DISPlay:PERSist

Command Format:
:DISPlay:PERSist {ON|OFF}
:DISPlay:PERSist?

Function Explanation:
The commands set and query the state of the waveform persist. "ON" denotes the record points hold until disable the presist, "OFF" denotes the record point varies in high refresh rate.

Returned Format:
The query returns ON or OFF.

Example:
:DISP:PERS ON   Enable the waveform persist.
:DISP:PERS?   The query returns ON.

4.   :DISPlay:MNUDisplay

Command Format:
:DISPlay:MNUDisplay <time>
:DISPlay:MNUDisplay?

Function Explanation:
The commands set and query the time for hiding menus automatically. <time> could be 1s, 2s, 5s, 10s, 20s or Infinite.

Returned Format:
The query returns 1s, 2s, 5s, 10s, 20s or Infinite.

Example:
:DISP:MNUD 10   Setup the display time of menu as 10s.
:DISP:MNUD?   The query returns 10s.

5.   :DISPlay:MNUStatus

Command Format:
:DISPlay:MNUStatus {ON|OFF}

:DISPlay:MNUStaus?

Function Explanation:
The commands set and query the state of the operation menu.

Returned Format:
The query returns ON or OFF.

Example:
:DISP:MNUS ON   Open menu for current operation.
:DISP:MNUS?   The query returns ON

6.   **:DISPlay:CLEar**

Command Format:
:DISPlay:CLEar

Function Explanation:
The command clears out of date waveforms on the screen during waveform persist.

7.   **:DISPlay:BRIGhtness**

Command Format:
:DISPlay:BRIGhtness <ncount>
:DISPlay:BRIGhtness?

Function Explanation:
The commands set and query the brightness of the grid. The range of <ncount> is from 0 to 32 (from dark to bright).

Returned Format:
The query returns the setting value of <ncount>.

Example:
:DISP:BRIG 10   Setup the grid brightness as 10.
:DISP:BRIG?   The query returns 10.

8. :DISPlay:INTensity

Command Format:
:DISPlay:INTensity <count>
:DISPlay:INTensity?

Function Explanation:
The commands set and query the brightness of the waveform. <count> could be the integer between 0 and 32.

Returned Format:
The query returns the setting value of <count>.

Example:
:DISP:INT 12   Setup the waveform brightness as 12.
:DISP:INT?   The query returns 12.

TIMebase Commands

TIMebase Commands set the horizontal scale (time base) and the waveform horizontal position in the memory (trigger offset). The waveform will enlarge or shrink among various horizontal scale.

TIMebase Commands include:

- :TIMebase:MODE
- :TIMebase[:DELayed]:OFFSet
- :TIMebase[:DELayed]:SCALe
- :TIMebase:FORMat

The detailed information of each command are given as follows:

1.   :TIMebase:MODE

Command Format:
:TIMebase:MODE <mode>
:TIMebase:MODE?

Function Explanation:
The commands set and query the scan mode of horizontal timebase. <mode> could be MAIN (main timebase) or DELayed (delayed scan).

Returned Format:
The query returns MAIN or DELAYED.

Example:
:TIM:MODE MAIN   Setup the horizontal timebase as MAIN.
:TIM:MODE?   The query returns MAIN.

2.   :TIMebase[:DELayed]:OFFSet

Command Format:
:TIMebase[:DELayed]:OFFSet <offset>
:TIMebase[:DELayed]:OFFSet?

Function Explanation:
The commands set and query the offset of the MAIN or DELayed timebase (that is offset of the waveform position relative to the trigger midpoint.). Thereinto,
In NORMAL mode, the range of <scale_val> is 1s ~ end of the memory;
In STOP mode, the range of <scale_val> is -500s ~ +500s;
In SCAN mode, the range of <scale_val> is -6*Scale ~ +6*Scale; (Note: Scale indicates the current horizontal scale, the unit is s/div.)
In MAIN state, the item [:DELayed] should be omitted.

Returned Format:
The query returns the setting value of the <offset> in s.

Example:
:TIM:MODE MAIN   Setup the scan mode of horizontal timebase as MAIN.
:TIM:OFFS 1   Setup the offset as 1s.

:TIM:OFFS?   The query returns 1.000e+00.

3. :TIMebase[:DELayed]:SCALe

Command Format:
:TIMebase[:DELayed]:SCALe <scale_val>
:TIMebase[:DELayed]:SCALe?

Function Explanation:
The commands set and query the horizontal scale for MAIN or DELayed
timebase, the unit is s/div (seconds/grid), thereinto:
In YT mode, the range of <scale_val> is 2ns - 50s;
In ROLL mode, the range of <scale_val> is 500ms - 50s;
In MAIN state, the item [:DELayed] should be omitted.

Returned Format:
The query returns the setting value of <scale_val> in s.

Example:
:TIM:MODE MAIN   Setup the timebase as MAIN.
:TIM:SCAL 2   Setup its scale as 2s.
:TIM:SCAL?   The query returns 2.000e+00.

4. :TIMebase:FORMat

Command Format:
:TIMebase:FORMat <value>
:TIMebase:FORMat?

Function Explanation:
The commands set and query the horizontal timebase. <value> could be XY, YT
or SCANning.

Returned Format:
The query returns X-Y, Y-T or SCANNING.

Example:

:TIM:FORM YT   Setup the form of grid as YT.
:TIM:FORM?   The query returns Y-T.

TRIGger Commands

In order to make the waveform displayed more stable, it is necessary to set the trigger system. The trigger determines when the oscilloscope starts to acquire data and display a waveform. If a trigger is set up properly, it converts unstable displays into meaningful waveforms.

When the oscilloscope starts to acquire a waveform, it collects enough data so that it can display the waveform to the left of the trigger point. The oscilloscope continues to acquire data while waiting for the trigger condition to occur. After it detects a trigger, the oscilloscope continues to acquire enough data so that it can display the waveform to the right of the trigger point.

The available trigger mode provided by DS1000E, DS1000D series digital oscilloscopes are: Edge, Pulse, Video, Slope, Pattern, Duration and Alternation.

Trigger Commands include:

Trigger Control
- :TRIGger:MODE
- :TRIGger<mode>:SOURce
- :TRIGger<mode>:LEVel
- :TRIGger<mode>:SWEep
- :TRIGger<mode>:COUPling
- :TRIGger:HOLDoff
- :TRIGger:STATus?
- :Trig%50
- :FORCetrig

EDGE Trigger
- :TRIGger:EDGE:SLOPe
- :TRIGger:EDGE:SENSitivity

PULSe Trigger
- :TRIGger:PULSe:MODE
- :TRIGger:PULSe:SENSitivity

● :TRIGger:PULSe:WIDTh

VIDEO Trigger
● :TRIGger:VIDEO:MODE
● :TRIGger:VIDEO:POLarity
● :TRIGger:VIDEO:STANdard
● :TRIGger:VIDEO:LINE
● :TRIGger:VIDEO:SENSitivity

SLOPe Trigger
● :TRIGger:SLOPe:TIME
● :TRIGger:SLOPe:SENSitivity
● :TRIGger:SLOPe:MODE
● :TRIGger:SLOPe:WINDow
● :TRIGger:SLOPe:LEVelA
● :TRIGger:SLOPe:LEVelB

PATTern Trigger
● :TRIGger:PATTern:PATTern

DURation Trigger
● :TRIGger:DURation:PATTern
● :TRIGger:DURation:TIME
● :TRIGger:DURation:QUALifier

ALTernation trigger
● :TRIGger:ALTernation:SOURce
● :TRIGger:ALTernation:TYPE
● :TRIGger:ALTernation:TimeSCALe
● :TRIGger:ALTernation:TimeOFFSet
● :TRIGger:ALTernation<mode>:LEVel
● :TRIGger:ALTernation:EDGE:SLOPe
● :TRIGger:ALTernation<mode>:MODE
● :TRIGger:ALTernation<mode>:TIME
● :TRIGger:ALTernation:VIDEO:POLarity
● :TRIGger:ALTernation:VIDEO:STANdard
● :TRIGger:ALTernation:VIDEO:LINE
● :TRIGger:ALTernation:SLOPe:WINDow

- :TRIGger:ALTernation:SLOPe:LEVelA
- :TRIGger:ALTernation:SLOPe:LEVelB
- :TRIGger:ALTernation<mode>:COUPling
- :TRIGger:ALTernation<mode>:HOLDoff
- :TRIGger:ALTernation<mode>:SENSitivity

The detailed information of each command are given as follows:

Trigger Control

1. :TRIGger:MODE

Command Format:
:TRIGger:MODE <mode>
:TRIGger:MODE?

Function Explanation:
The commands set and query the mode of trigger. <mode> could be EDGE, PULSe, VIDEO, SLOPe, PATTern, DURation or ALTernation.

Returned Format:
The query returns EDGE, PULSE, VIDEO, SLOPE, PATTERN, DURATION or ALTERNATION.

Example:
:TRIG:MODE EDGE   Setup the trigger mode as EDGE.
:TRIG:MODE?   The query returns EDGE.

2.   :TRIGger<mode>:SOURce

Command Format:
:TRIGger<mode>:SOURce <src>
:TRIGger<mode>:SOURce?

Function Explanation:
The commands set and query the trigger source. <src> could be the input channel (CHANnel1, CHANnel2), external trigger channel (EXT), AC Line (Mains supply) or digital channel (Digital).
In EDGE mode, <src> could be CHANnel<n>, EXT, ACLine or DIGital<m>;
In PULSe mode, <src> could be CHANnel<n>, EXT or DIGital<m>;
In SLOPe mode, <src> could be CHANnel<n> or EXT;
In VIDEO mode, <src> could be CHANnel<n> or EXT;
Thereinto, <n> could be 1 or 2; the range of <m> is 0~15.

Returned Format:

The query returns CH1, CH2, EXT, ACLINE, D0~D15.

Example:

:TRIG:EDGE:SOUR CHAN1   Setup the soure of edge trigger as CHANnel1.

:TRIG:EDGE:SOUR?   The query returns CH1.

3. :TRIGger<mode>:LEVel

Command Format:

:TRIGger<mode>:LEVel <level>

:TRIGger<mode>:LEVel?

Function Explanation:

The commands set and query the trigger level. <level> could be :EDGE, :PULSe or :VIDEO; the range is: -6*Scale~+6*Scale, Scale indicates the current vertical scale, the unit is V/div.

Returned Format:

The query returns the setting value of <level> in V.

Example:

:TRIG:EDGE:LEV 1   Setup the level of EDGE trigger as 1.

:TRIG:EDGE:LEV?   The query returns 1.00e+00.

4. :TRIGger<mode>:SWEep

Command Format:

:TRIGger<mode>:SWEep {AUTO|NORMal|SINGle}

:TRIGger<mode>:SWEep?

Function Explanation:

The commands set and query the trigger type. <mode>could be :EDGE, :PULSe, :SLOPe, :PATTern or :DURation.

Returned Format:

The query returns AUTO, NORMAL or SINGLE.

Example:

:TRIG:EDGE:SWE AUTO   Setup the trigger type as AUTO.

:TRIG:EDGE:SWE?   The query returns AUTO.

5.   :TRIGger<mode>:COUPling

Command Format:

:TRIGger<mode>:COUPling {DC|AC|HF|LF}

:TRIGger<mode>:COUPling?

Function Explanation:

The commands set and query the coupling type. Thereinto,

DC: Allow all signals pass.

AC: Block DC signals and attenuate the signals lower than 10Hz.

HF: Reject high frequency signals (Higher than 150KHz).

LF: Reject DC signals and attenuate low frequency signals (Lower than 8KHz).

<mode> could be :EDGE, :PULSe or :SLOPe.

Returned Format:

The query returns DC, AC, HF or LF.

Example:

:TRIG:EDGE:COUP DC   Setup the coupling type as DC.

:TRIG:EDGE:COUP?   The query returns DC.

6.   :TRIGger:HOLDoff

Command Format:

:TRIGger:HOLDoff <count>

:TRIGger:HOLDoff?

Function Explanation:

The commands set and query the trigger holfoff time. The range of <count> is

<count>: 500ns~1.5s.

Returned Format:

The query returns the setting value of <count> in s.

Example:

:TRIG:HOLD 0.0005   Setup the holdoff time as 500ms.

:TRIG:HOLDoff?   The query returns 5.000e-04

7.   :TRIGger:STATus?

Command Format:

:TRIGger:STATus?

Function Explanation:

The command queries the operating status of the oscilloscope. The status could be RUN, STOP, T`D, WAIT or AUTO.

Returned Format:

The query returns RUN, STOP, T'D, WAIT or AUTO.

8.   :Trig%50

Command Format:

:Trig%50

Function Explanation:

The command sets the trigger level to the vertical midpoint of amplitude.

9.   :FORCetrig

Command Format:

:FORCetrig

Function Explanation:

The command forces the oscilloscope to trigger signal, which is usually used in "Normal" and "Single" mode.

EDGE Trigger

1. :TRIGger:EDGE:SLOPe

Command Format:
:TRIGger:EDGE:SLOPe {POSitive|NEGative}
:TRIGger:EDGE:SLOPe?

Function Explanation:
The commands set and query the type of edge trigger. The type could be
POSitive (Rising edge) or NEGative (Failing edge).

Returned Format:
The query returns POSITIVE or NEGATIVE.

Example:
:TRIG:EDGE:SLOP POS   Set up the edge of trigger as POSitive.
:TRIG:EDGE:SLOP?   The query returns POSITIVE.

2. :TRIGger:EDGE:SENSitivity

Command Format:
:TRIGger:EDGE:SENSitivity <count>
:TRIGger:EDGE:SENSitivity?

Function Explanation:
The commands set and query the sensitive of edge trigger. The range of
<count> could be 0.1div~1div.

Returned Format:
The query returns the setting value <count> in div.

Example:
:TRIG:EDGE:SENS 0.5   Setup the sensitive as 0.5div.
:TRIG:EDGE:SENS？   The query returns 5.00e-01.

PULSe Trigger

1.   :TRIGger:PULSe:MODE

Command Format:
:TRIGger:PULSe:MODE <mode>
:TRIGger:PULSe:MODE?

Function Explanation:
The commands set and query the pulse condition. <mode> could be +GREaterthan (positive pulse greater than), +LESSthan (positive pulse less than), +EQUal (positive pulse equals to), -GREaterthan (negative pulse greater than), -LESSthan (negative pulse less than) or –EQUal (negative pulse equals to).

Returned Format:
The query returns +GREATER THAN, +LESS THAN, +EQUAL, -GREATER THAN, -LESS THAN or -EQUAL.

Example:
:TRIG:PULS:MODE +GRE   Setup the trigger condition as +GREaterthan.
:TRIG:PULS:MODE?   The query returns +GREATER THAN.

2.   :TRIGger:PULSe:SENSitivity

Command Format:
:TRIGger:PULSe:SENSitivity <count>
:TRIGger:PULSe:SENSitivity?

Function Explanation:
The commands set and query the sensitive of pulse trigger. The range of <count> could be 0.1div~1div.

Returned Format:
The query returns the setting value of <count> in div.

Example:

:TRIG:PULS:SENS 0.5   Setup the sensitive as 0.5div.
:TRIG:PULS:SENS?   The query returns 5.00e-01.

3.   :TRIGger:PULSe:WIDTh

Command Format:
:TRIGger:PULSe:WIDTh <wid>
:TRIGger:PULSe:WIDTh?

Function Explanation:
The commands set and query the pulse width. The range of <wid> is 20ns ~
10s.

Returned Format:
The query returns the setting value of the <wid> in s.

Example:
:TRIG:PULS:WIDT 0.001   Setup the width of the pulse as 1ms.
:TRIG:PULS:WIDT?   The query returns 1.000e-03.

VIDEO Trigger

1. :TRIGger:VIDEO:MODE

Command Format:
:TRIGger:VIDEO:MODE <mode>
:TRIGger:VIDEO:MODE?

Function Explanation:
The commands set and query the synchronous mode of the video trigger.
<mode> could be ODDfield, EVENfield, LINE or ALLlines.

Returned Format:
The query returns ODD FIELD, EVEN FIELD, LINE or ALL LINES.

Example:
:TRIG:VIDEO:MODE EVEN   Setup the synchronous mode as EVENfield.
:TRIG:VIDEO:MODE?   The query returns EVEN FIELD

2. :TRIGger:VIDEO:POLarity

Command Format:
:TRIGger:VIDEO:POLarity {POSitive|NEGative}
:TRIGger:VIDEO:POLarity?

Function Explanation:
The commands set and query the video polarity. The polarity could be POSitive
or NEGative.

Returned Format:
The query returns POSITIVE or NEGATIVE.

Example:
:TRIG:VIDEO:POL POS   Setup the polarity of video trigger as POSitive.
:TRIG:VIDEO:POL?   The query returns POSITIVE.

3. :TRIGger:VIDEO:STANdard

Command Format:
:TRIGger:VIDEO:STANdard {NTSC|PALSecam}
:TRIGger:VIDEO:STANdard?

Function Explanation:
The commands set and query the type of video trigger standard.

Returned Format:
The query returns NTSC or PAL/SECAM.

Example:
:TRIG:VIDEO:STAN PALS   Setup the video standard as PALSecam.
:TRIG:VIDEO:STAN?   The query returns PAL/SECAM.

4. :TRIGger:VIDEO:LINE

Command Format:
:TRIGger:VIDEO:LINE <value>
:TRIGger:VIDEO:LINE?

Function Explanation:
The commands set and query the number of specified line of synchronous. In NTSC standard, the range of <value> is 1~525; in PAL/SECAM standard, the range of <value> is 1~625.

Returned Format:
The query returns the numbers of current line.

Example:
:TRIG:VIDEO:LINE 25   Setup the number of specified line as 25.
:TRIG:VIDEO:LINE?   The query returns 25.

5. :TRIGger:VIDEO:SENSitivity

Command Format:

:TRIGger:VIDEO:SENSitivity <count>
:TRIGger:VIDEO:SENSitivity?

Function Explanation:

The commands set and query the trigger sensitive, the range of <count> is: 0.1div ～1div.

Returned Format:

The query returns the setting value of <count> in div.

Example:

:TRIG:VIDEO:SENS 0.5   Setup the trigger sensitive as 0.5div.
:TRIG:VIDEO:SENS?   The query returns 5.00e-01.

SLOPe Trigger

1.   :TRIGger:SLOPe:TIME

Command Format:
:TRIGger:SLOPE:TIME <count>
:TRIGger:SLOPE:TIME?

Function Explanation:
The commands set and query the time setting about slope trigger. The range of <count> is 20ns~10s.

Returned Format:
The query returns the setting value of <count> in s.

Example:
:TRIG:SLOP:TIME 0.01   Setup the slope time is 10ms.
:TRIG:SLOP:TIME?   The query returns 1.000e-02.

2.   :TRIGger:SLOPe:SENSitivity

Command Format:
:TRIGger:SLOPe:SENSitivity <count>
:TRIGger:SLOPe:SENSitivity?

Function Explanation:
The commands set and query the trigger sensitive. The range of <count> is: 0.1div ~1div.

Returned Format:
The query returns the setting value of <count> in div.

Example:
:TRIG:SLOP:SENS 0.5   Setup the trigger sensitive as 0.5div
:TRIG:SLOP:SENS?   The query returns 5.00e-01

3.   :TRIGger:SLOPe:MODE

Command Format:
:TRIGger:SLOPe:MODE <mode>
:TRIGger:SLOPe:MODE?

Function Explanation:
The commands set and query the slope condition. <mode> could be +GREaterthan (positive slope greater than), +LESSthan (positive slope less than), + EQUal (positive slope equals to), -GREaterthan (negative slope greater than), -LESSthan (negative slope less than) or –EQUal (negative slope equals to).

Returned Format:
The query returns +GREATER THAN, +LESS THAN, +EQUAL, -GREATER THAN, -LESS THAN 或-EQUAL.

Example:
:TRIG:SLOP:MODE +GRE   Setup the slope condition as +GREaterthan.
:TRIG:SLOPe:MODE?   The query returns +GREATER THAN.

4.   :TRIGger:SLOPe:WINDow

Command Format:
:TRIGger:SLOPe:WINDow <count>
:TRIGger:SLOPe:WINDow?

Function Explanation:
The commands set and query the type of trigger level which can be adjusted by ⊚LEVEL.
When the slope condition is +GREaterthan, +LESSthan or + EQUal, <count> could be PA (rising edge Level A), PB (rising edge Level B) or PAB (rising edge Level AB);
When the slope condition is -GREaterthan, -LESSthan or –EQUal, <count> could be NA (falling edge Level A), NB (falling edge LevelB) or NAB (falling edge LevelAB).

Returned Format:

The query returns P_WIN_A, P_WIN_B, P_WIN_AB, N_WIN_A, N_WIN_B or N_WIN_AB.

Example:
:TRIG:SLOP:WIND PA   Setup the type of trigger level as rising edge Level A
:TRIG:SLOP:WIND?   The query returns P_WIN_A.

5.   :TRIGger:SLOPe:LEVelA

Command Format:
:TRIGger:SLOPe:LEVelA <value>
:TRIGger:SLOPe:LEVelA?

Function Explanation:
The commands set and query the upper boundary "Level A" of trigger level. The range of <value> is LevelB~+ 6*Scale; Scale indicates the current vertical level, the unit is V/div.

Returned Format:
The query returns the setting value of level in V.

Example:
:TRIG:SLOP:LEVA 2   Setup the upper boundary of trigger level as 2V.
:TRIG:SLOP:LEVA?   The query returns 2.000e+00.

6.   :TRIGger:SLOPe:LEVelB

Command Format:
:TRIGger:SLOPe:LEVelB <value>
:TRIGger:SLOPe:LEVelB?

Function Explanation:
The commands set and query the lower boundary "LEVel B" of trigger level. The range of <value> is -6*Scale~LevelA; Scale indicates the current vertical level, the unit is V/div.

Returned Format:

The query returns the setting value of level in V.
Note: Level A (upper boundary) can not be less than the maximum of Level B
(lower boundary).

Example:
:TRIG:SLOP:LEVB -1.5   Setup the lower boundary of trigger level as -1.5V.
:TRIG:SLOP:LEVB?   The query returns -1.500e+00.

PATTern Trigger

1. :TRIGger:PATTern:PATTern

Command Format:
:TRIGger:PATTern:PATTern <value>,<mask>, [<edge source>,<edge>]
:TRIGger:PATTern:PATTern?

Function Explanation:
The commands set and query the type of pattern for digital signal. Thereinto,
When <value> is a 16-bit integer without sign, high is 1, and low is o;
When <mask> is an 16-bit integer without sign, enable is 1, and X is 0;
The range of <edge source> is 0~15 (DIG0~DIG15);
<edge> is 1 at rising edge and 0 at falling edge, separately.

Returned Format:
The query returns the setting value of <value>, <mask>, <edge source> and
<edge> in turns. Thereinto, the value returned by <value> and <mask> is a
decimal data.

Example:
:TRIG:PATT:PATT 65535,65535,2,1   Setup the type of pattern.
:TRIG:PATT:PATT?   The query returns 65535, 65535, DIG2,
                                                              Positive.

DURation Trigger

1.   :TRIGger:DURation:PATTern

Command Format:
:TRIGger:DURation:PATTern <value>,<mask>
:TRIGger:DURation:PATTern?

Function Explanation:；
The commands set and query the type of pattern for digital signal. Thereinto,
When <value> is a 16-bit integer without sign, high is 1, and low is o;
When <mask> is a 16-bit integer without sign, enable is 1, and X is 0.

Returned Format:
The query returns the setting value of <value>, <mask> in turns.

Example:
:TRIG:DUR:PATT 65535,655535   Setup the type of pattern.
:TRIG:DUR:PATT?   The query returns 65535,65535.

2.   :TRIGger:DURation:TIME

Command Format:
:TRIGger:DURation:TIME <time>
:TRIGger:DURation:TIME?

Function Explanation:
The commands set and query the limiting time of duration trigger. The range of
<time> is 2ns~10s.

Returned Format:
The query returns the setting value of <time>.

Example:
:TRIG:DUR:TIME 0.05   Setup the limiting time of duration trigger as 50ms.
:TRIG:DUR:TIME?   The query returns 5.00e-02.

3.   :TRIGger:DURation:QUALifier

Command Format:
:TRIGger:DURation:QUALifier <qual>
:TRIGger:DURation:QUALifier?

Function Explanation:；
The commands set and query the limiting condition of trigger time. <qual>
could be GREaterthan, LESSthan or EQUal.

Returned Format:
The query returns GREATER THAN, LESS THAN or EQUAL.

Example:
:TRIG:DUR:QUAL GRE   Setup the limiting condition as "GREaterthan".
:TRIG:DUR:QUAL?   The query returns GREATER THAN.

ALTernation Trigger

1. :TRIGger:ALTernation:SOURce

Command Format:
:TRIGger:ALTernation:SOURce <src>
:TRIGger:ALTernation:SOURce?

Function Explanation:
The commands set and query which channel is to be used. <src> could be CHANnel1 or CHANnel2.

Returned Format:
The query returns CH1 or CH2.

Example:
:TRIG:ALT:SOUR CHAN2   Select CHANnel2 as current channel.
:TRIG:ALT:SOUR?   The query returns CH2.

2. :TRIGger:ALTernation:TYPE

Command Format:
:TRIGger:ALTernation:TYPE <value>
:TRIGger:ALTernation:TYPE?

Function Explanation:
The commands set and query the trigger type in alternation mode. <value> could be EDGE, PULSe, SLOPe or VIDEO.

Returned Format:
The query returns EDGE, PULSE, SLOPE or VIDEO.

Example:
:TRIG:ALT:TYPE EDGE   Setup the trigger type as EDGE.
:TRIG:ALT:TYPE?   The query returns EDGE.

3.   :TRIGger:ALTernation:TimeSCALe

Command Format:
:TRIGger:ALTernation:TimeSCALe <value>
:TRIGger:ALTernation:TimeSCALe?

Function Explanation:
The commands set and query the timebase of the current channel. The range of <value> is: 2ns～20ms.

Returned Format:
The query returns the setting value of the timebase in s.

Example:
:TRIG:ALT:TSCAL 0.001   Setup the timebase as 1ms.
:TRIG:ALT:TSCAL?   The query returns 1.000e-03.

4.   :TRIGger:ALTernation:TimeOFFSet

Command Format:
:TRIGger:ALTernation:TimeOFFSet <value>
:TRIGger:ALTernation:TimeOFFSet?

Function Explanation:
The commands set and query the horizontal timebase offset of the current channel. Thereinto,
In NORMAL mode, the range of <value> is 1s ~ end point of memory;
In STOP mode, the range of <value> is -500s ~ +500s；
In ROLL mode, the range of <value> is -6*scale ~ +6*Scale.
Scale indicates the current horizontal scalre, the unit is s/div.

Returned Format:
The query returns the setting value of offset in s.

Example:
:TRIG:ALT:TOFFS 0.0002   Setup the horizontal timebase offset as 200us.
:TRIG:ALT:TOFFS?   The query returns 2.000e-04.

5.   :TRIGger:ALTernation<mode>:LEVel

Command Format:
:TRIGger:ALTernation<mode>:LEVel <value>
:TRIGger:ALTernation<mode>:LEVel?

Function Explanation:
The commands set and query the trigger level of the current channel. <mode> could be :EDGE, :PULSe or :VIDEO. The range of <value> is - 6*Scale ~ + 6*Scale，Scale indicates the current vertical scale, the unit is V/div.

Returned Format:
The query returns the setting value of <value> in V.

Example:
:TRIG:ALT:EDGE:LEV 2   Setup the trigger level of the current channel as 2V.
:TRIG:ALT:EDGE:LEV?   The query returns 2.00e+00.

6.   :TRIGger:ALTernation:EDGE:SLOPe

Command Format:
:TRIGger:ALTernation:EDGE:SLOPe <value>
:TRIGger:ALTernation:EDGE:SLOPe?

Function Explanation:
The commands set and query the current edge type. <value> could be POSitive (rising edge) or NEGative (falling edge).

Returned Format:
The query returns POSITIVE or NEGATIVE.

Example:
:TRIG:ALT:EDGE:SLOP POS   Setup the edge type.
:TRIG:ALT:EDGE:SLOP?   The query returns POSITIVE.

7.   :TRIGger:ALTernation<mode>:MODE

Command Format:
:TRIGger:ALTernation<mode>:MODE <value>
:TRIGger:ALTernation<mode>:MODE?

Function Explanation:
The commands set and query the current pulse condition and slope condition or the synchronization condition of the video trigger. <mode> could be :PULSe, :SLOPe or :VIDEO.
In PULSe or SLOPe mode, the <value> could be +GREaterthan, +LESSthan, + EQUal, -GREaterthan, -LESSthan or -EQUal;
In VIDEO mode, the <value> could be ODDfield, EVENfield, LINE or ALLlines.

Returned Format:
The query returns the setting value of <value>.

Example:
:TRIG:ALT:PULS:MODE +GRE   Setup the pulse condition.
:TRIG:ALT:PULS:MODE?   The query returns +GREATER THAN.

8.   **:TRIGger:ALTernation<mode>:TIME**

Command Format:
:TRIGger:ALTernation<mode>:TIME <value>
:TRIGger:ALTernation<mode>:TIME?

Function Explanation:
The commands set and query the current pulse width or slope time. <mode> could be :SLOPe or :PULSe, the range of <value> is 20ns~10s.

Returned Format:
The query returns the setting value of <value> in s.

Example:
:TRIG:ALT:SLOP:TIME 0.002   Setup the slope time as 2ms.
:TRIG:ALT:SLOP:TIME?   The query returns 2.000e-03

9.   :TRIGger:ALTernation:VIDEO:POLarity

Command Format:

:TRIGger:ALTernation:VIDEO:POLarity {POSitive|NEGative}

:TRIGger:ALTernation:VIDEO:POLarity?

Function Explanation:

The commands set and query the current video polarity.

Returned Format:

The query returns POSITIVE or NEGATIVE.

Example:

:TRIG:ALT:VIDEO:POL POS   Setup the video polarity as POSITIVE.

:TRIG:ALT:VIDEO:POL?   The query returns POSITIVE.

10. :TRIGger:ALTernation:VIDEO:STANdard

Command Format:

:TRIGger:ALTernation:VIDEO:STANdard {NTSC|PALSecam}

:TRIGger:ALTernation:VIDEO:STANdard?

Function Explanation:

The commands set and query the current video trigger standard.

Returned Format:

The query returns NTSC or PAL/SECAM.

Example:

:TRIG:ALT:VIDEO:STAN NTSC   Setup the video standard as NTSC.

:TRIG:ALT:VIDEO:STAN?   The query returns NTSC.

11. :TRIGger:ALTernation:VIDEO:LINE

Command Format:

:TRIGger:ALTernation:VIDEO:LINE <value>

:TRIGger:ALTernation:VIDEO:LINE?

Function Explanation:
The commands set and query the current number of specified line in synchronous mode. In NTSC standard，the range of <value> is 1~525；In AL/SECAM standard, the range of <value> is 1~625.

Returned Format:
The query returns the setting value of <value>.

Example:
:TRIG:ALT:VIDEO:LINE 100   Setup the number of specified line in
                                                       synchronous mode as 100.
:TRIG:ALT:VIDEO:LINE?   The query returns 100.

12. :TRIGger:ALTernation:SLOPe:WINDow

Command Format:
:TRIGger:ALTernation:SLOPe:WINDow <count>
:TRIGger:ALTernation:SLOPe:WINDow?

Function Explanation:
The commands set and query the type of trigger level which can be adjusted by ⊚LEVEL in current state.
When the slope is +GREaterthan, +LESSthan or + EQUal, the <count> could be PA (rising edge Level A), PB (rising edge Level B) or PAB (rising edge Level AB); When the slope condition is -GREaterthan, -LESSthan or –EQUal, the <count> could be NA (failing edge Level A), NB (failing edge LevelB) or NAB (failing edge LevelAB).

Returned Format:
The query returns P_WIN_A, P_WIN_B, P_WIN_AB, N_WIN_A, N_WIN_B or N_WIN_AB.

Example:
:TRIG:ALT:SLOP:WIND PA   Setup the type of trigger level as rising edge
                                            Level A.
:TRIG:ALT:SLOP:WIND?   The query returns P_WIN_A.

13. :TRIGger:ALTernation:SLOPe:LEVelA

Command Format:
:TRIGger:ALTernation:SLOPe:LEVelA <value>
:TRIGger:ALTernation:SLOPe:LEVelA?

Function Explanation:
The commands set and query the current upper boundary "Level A" of trigger level in slope trigger mode. The range of <value> is LevelB ~ +6*Scale; Scale indicates the current vertical scale, the unit is V/div.

Returned Format:
The query returns the setting value of <value> in V.

Example:

:TRIG:ALT:SLOP:LEVA 2	Setup the upper boundary "Level A" of trigger level as 2V.
:TRIG:ALT:SLOP:LEVA?	The query returns 2.000e+00.

14. :TRIGger:ALTernation:SLOPe:LEVelB

Command Format:
:TRIGger:ALTernation:SLOPe:LEVelB <value>
:TRIGger:ALTernation:SLOPe:LEVelB?

Function Explanation:
The commands set and query the current lower boundary "Level B" of trigger level in slope trigger mode. The range of <value> is -6*Scale ~ LevelA，Scale indicates the current vertical scale, the unit is V/div.
Note: The value of Level A (upper boundary) should not be less than the maximum of Level B (lower boundary).

Returned Format:
The query returns the setting value of <value> in V.

Example:

:TRIG:ALT:SLOP:LEVB -1.5	Setup the lower boundary "Level B" of trigger level as -1.5V.

:TRIG:ALT:SLOP:LEVB ?   The query returns -1.500e+00.

15. :TRIGger:ALTernation<mode>:COUPling

Command Format:
:TRIGger:ALTernation<mode>:COUPling {DC|AC|HF|LF}
:TRIGger:ALTernation<mode>:COUPling?

Function Explanation:
The commands set and query the coupling mode. Thereinto,
DC: Allow all signals pass;
AC: Block DC signals and attenuate the signals lower than10Hz;
HF (High frequency reject): Reject high frequency signals (Higher than 150 KHz);
LF (Low frequency reject): Reject DC signals and attenuate low frequency signals (Lower than 8KHz).
<mode>could be :EDGE, :PULSe or :SLOPe.

Returned Format:
The query returns DC , AC, HF or LF.

Example:
:TRIG:ALT:EDGE:COUP DC   Setup the coupling mode as DC.
:TRIG:ALT:EDGE:COUP?   The query returns DC.

16. :TRIGger:ALTernation<mode>:HOLDoff

Command Format:
:TRIGger:ALTernation<mode>:HOLDoff <count>
:TRIGger:ALTernation<mode>:HOLDoff ?

Function Explanation:
The commands set and query the trigger holdoff. The <mode> could be EDGE, :PULSe, :SLOPe or :VIDEO. The range of <count> is 500ns~1.5s.

Returned Format:
The query returns the setting value of <count> in s.

Example:

:TRIG:ALT:PULS:HOLD 1   Setup the holdoff time as 1s.

:TRIG:ALT:PULS:HOLD?   The query returns 1.000e+00.

17. :TRIGger:ALTernation<mode>:SENSitivity

Command Format:

:TRIGger:ALTernation<mode>:SENSitivity <count>

:TRIGger:ALTernation<mode>:SLOPe:SENSitivity?

Function Explanation:；

The commands set and query the trigger sensitive. The <mode> could be :EDGE, :PULSe, :SLOPe or :VIDEO. The range of <count> is 0.1div ~1div.

Returned Format:

The query returns the setting value of <count> in div.

Example:

:TRIG:ALT:EDGE:SENS 0.5   Setup the trigger sensitive as 0.5div.

:TRIG:ALT:EDGE:SLOP:SENS?   The query returns 5.00e-01.

STORage Command

STORage Command is used for recalling the factory settings, including:

● :STORage:FACTory:LOAD

The detailed information of each command are given as follows:

1.   :STORage:FACTory:LOAD

Command Format:

:STORage:FACTory:LOAD

Function Explanation:

The command recalls the system settings before leaving factory.

MATH Commands

MATH (math operation) Commands are used for executing the add, reduce, multiply and FFT operations of wavefrom from CH1 and CH2 and display the results. The results are measured by grid or cusor.

MATH Commands include:

- :MATH:DISPlay
- :MATH:OPERate
- :FFT:DISPlay

The detailed information of each command are given as follows:

1.   :MATH:DISPlay

Command Format:
:MATH:DISPlay {ON|OFF}
:MATH:DISPlay?

Function Explanation:
The commands set and query the On/Off state of the math operation.

Returned Format:
The query returns ON or OFF.

Example:
:MATH:DISP ON   Enable math operation.
:MATH:DISP?   The query returns ON

2.   :MATH:OPERate

Command Format:
:MATH:OPERate <operate>
:MATH:OPERate?

Function Explanation:
The commands set and query the math operation type of the current channel.
<operate> could be A+B, A-B, AB or FFT.

Returned Format:
The query returns A+B, A-B, A*B or FFT.

Example:
:MATH:OPER A-B   Setup the operation as A-B.
:MATH:OPER?   The query returns A-B.

3.   :FFT:DISPlay

Command Format:
:FFT:DISPlay {ON|OFF}

:FFT:DISPlay?

Function Explanation:
The commands set and query the On/Off state of FFT operation.

Returned Format:
The query returns ON or OFF.

Example:
:FFT:DISP ON   Enable FFT operation.
:FFT:DISP?   The query returns ON.

CHANnel Commands

CHANnel Commands are used to control the two channels of the oscilloscope, each channel has an independent vertical menu and each option could be set separately based on different channels.

CHANnel Commands include:

● :CHANnel<n>:BWLimit
● :CHANnel<n>:COUPling
● :CHANnel<n>:DISPlay
● :CHANnel<n>:INVert
● :CHANnel<n>:OFFSet
● :CHANnel<n>:PROBe
● :CHANnel<n>:SCALe
● :CHANnel<n>:FILTer
● :CHANnel<n>:MEMoryDepth?
● :CHANnel<n>:VERNier

The detailed information of each command are given as follows:

1.   :CHANnel<n>:BWLimit

Command Format:
:CHANnel<n>:BWLimit {ON|OFF}
:CHANnel<n>:BWLimit?

Function Explanation:
The commands set and query the On/Off state of bandwidth limit. <n> could be 1 or 2.

Returned Format:
The query returns ON or OFF.

Example:
:CHAN2:BWL OFF   Turn off the bandwidth limit to CH2.
:CHAN2:BWL?   The query returns OFF.

2.   :CHANnel<n>:COUPling

Command Format:
:CHANnel<n>:COUPling {DC|AC|GND}
:CHANnel<n>:COUPling?

Function Explanation:
The commands set and query the coupling mode of channel. DC indicates both the AC and DC components passed from input signal; AC indicates the blocked DC components; GND indicates to cut off the input of signal; <n> could be 1 or 2.

Returned Format:
The query returns AC, DC or GND.

Example:
:CHAN2:COUP DC   Enable the coupling mode of channel 2.
:CHAN2:COUP?   The query returns DC.

3. :CHANnel<n>:DISPlay

Command Format:
:CHANnel<n>:DISPlay {ON|OFF}
:CHANnel<n>:DISPlay?

Function Explanation:
The commands set and query the On/Off state of channel. <n> could be 1 or 2.

Returned Format:
The query returns ON or OFF.

Example:
:CHAN2:DISP ON   Open the display of Channel 2.
:CHAN2:DISP?   The query returns ON.

4.   :CHANnel<n>:INVert

Command Format:
:CHANnel<n>:INVert {ON|OFF}
:CHANnel<n>:INVert?

Function Explanation:
The commands set and query the On/Off state of the waveform inverted. <n> could be 1 or 2.

Returned Format:
The query returns ON or OFF.

Example:
:CHAN2:INV OFF   Turn off the inverted display of Channel 2.
:CHAN2:INV?   The query returns OFF.

5.   :CHANnel<n>:OFFSet

Command Format:
:CHANnel<n>:OFFSet <offset>
:CHANnel<n>:OFFSet?

Function Explanation:

The commands set and query the vertical offset. <n> could be 1 or 2.

When Scale≥250mV, the range of <offset>is -40V~+40V;

When Scale<250mV, the range of <offset>is -2V~+2V.

Returned Format:

The query returns the setting value of <offset>.

Example:

:CHAN2:OFFS 20   Setup the vertical offset of channel 2 as 20V.

:CHAN2:OFFS?   The query returns 2.000e+01.

6.   :CHANnel<n>:PROBe

Command Format:

:CHANnel<n>:PROBe <attn>

:CHANnel<n>:PROBe?

Function Explanation:

The commands set and query the attenuation factor of the probe. <n> could be
1 or 2; <attn> could be 1, 5, 10, 50, 100, 500 or 1000.

Returned Format:

The query returns the setting value of <attn>.

Example:

:CHAN2:PROB 10   Setup the attenuation ratio of channel 2 as 10X.

:CHAN2:PROB?   The query returns 1.000e+01.

7.   :CHANnel<n>:SCALe

Command Format:

:CHANnel<n>:SCALe <range>

:CHANnel<n>:SCALe?

Function Explanation:

The commands set and query the vertical scale of waveform magnified by the

oscilloscope. <n> could be 1 or 2.

When the Probe is set to 1X, the range of <range> is 2mV ~ 10V;

When the Probe is set to 5X, the range of <range> is 10mV ~50V;

When the Probe is set to 10X, the range of <range> is 20mV ~ 100V;

When the Probe is set to 50X, the range of <range> is 100mV ~ 500V;

When the Probe is set to 100X, the range of <range> is 200mV ~ 1000V;

When the Probe is set to 500X, the range of <range> is 1V ~5000V;

When the Probe is set to 1000X, the range of <range> is 2V~ 10000V.

Returned Format:
The query returns the setting value of <range> in V.

Example:
:CHAN2:PROB 10	Setup the attenuation ratio of the probe as 10X.
:CHAN2:SCAL 20	Setup the vertical scale as 20V.
:CHAN2:SCAL?	The query returns 2.000e+01.

8.   :CHANnel<n>:FILTer

Command Format:
:CHANnel<n>:FILTer {ON|OFF}

:CHANnel<n>:FILTer?

Function Explanation:
The commands set and query the On/Off state of the filter. <n> could be 1 or 2.

Returned Format:
The query returns ON or OFF.

Example:
:CHAN2:FILT OFF	Turn off the filter of Channel 2.
:CHAN2:FILT?	The query returns OFF.

9.   :CHANnel<n>:MEMoryDepth?

Command Format:
:CHANnel<n>:MEMoryDepth?

Function Explanation:
The command queries the memory depth of the specified channel. <n> could
be 1 or 2.
In long memory, up to 1Mpts could be stored in single channel and 512kpts in
dual channels;
In common memory, up to 16kpts could be stored in single channel and 8kpts in
dual channels.

Returned Format:
Such as 8192 (8kpts).

10. :CHANnel<n>:VERNier

Command Format:
:CHANnel<n>:VERNier {ON|OFF}
:CHANnel<n>:VERNier?

Function Explanation:
The commands set and query the adjusting mode of scale. ON denotes Fine,
OFF denotes Coarse; <n> could be 1 or 2.

Returned Format:
The query returns Coarse or Fine.

Example:
:CHAN2:VERN ON   Enable Fine.
:CHAN2:VERN?   The query returns Fine.

MEASure Commands

MEASure Commands are used for the basic measurement only available for analog channel. Generally, the query returns the results in scientific notation.

MEASure Commands include:

- :MEASure:CLEar
- :MEASure:VPP?
- :MEASure:VMAX?
- :MEASure:VMIN?
- :MEASure:VAMPlitude?
- :MEASure:VTOP?
- :MEASure:VBASe?
- :MEASure:VAVerage?
- :MEASure:VRMS?
- :MEASure:OVERshoot?
- :MEASure:PREShoot?
- :MEASure:FREQuency?
- :MEASure:RISetime?
- :MEASure:FALLtime?
- :MEASure:PERiod?
- :MEASure:PWIDth?
- :MEASure:NWIDth?
- :MEASure:PDUTycycle?
- :MEASure:NDUTycycle?
- :MEASure:PDELay?
- :MEASure:NDELay?
- :MEASure:TOTal
- :MEASure:SOURce

The detailed information of each command are given as follows:

1.   :MEASure:CLEar

Command Format:
:MEASure:CLEar

Function Explanation:
The command clears the current parameter value.

2.   :MEASure:VPP?

Command Format:
:MEASure:VPP? [<source>]

Function Explanation:
The command queries the peak-peak value of the waveform under measure.
<source> could be CHANnel1 or CHANnel2.

Returned Format:
Such as: 5.12e+03, the unit is V.

3.   :MEASure:VMAX?

Command Format:
:MEASure:VMAX? [<source>]

Function Explanation:
The command queries the maximum of the waveform under measure.
<source> could be CHANnel1 or CHANnel2.

Returned Format:
Such as: 2.60e+03, the unit is V.

4.   :MEASure:VMIN?

Command Format:
:MEASure:VMIN? [<source>]

Function Explanation:
The command queries the minimum of the waveform under measure. <source>
could be CHANnel1 or CHANnel2.

Returned Format:
Such as: -2.52e+03, the unit is V.

5.   :MEASure:VAMPlitude?

Command Format:
:MEASure:VAMPlitude? [<source>]

Function Explanation:
The command queries the amplitude of the waveform under measure.
<source> could be CHANnel1 or CHANnel2.

Returned Format:
Such as: 5.07e+03, the unit is V.

6.   :MEASure:VTOP?

Command Format:
:MEASure:VTOP? [<source>]

Function Explanation:
The command queries the top value of the waveform under measure. <source>
could be CHANnel1 or CHANnel2.

Returned Format:
Such as: 2.55e+03, the unit is V.

7.   :MEASure:VBASe?

Command Format:
:MEASure:VBASe? [<source>]

Function Explanation:

The command queries the bottom value of the waveform under measure. <source> could be CHANnel1 or CHANnel2.

Returned Format:

Such as: -2.52e+03, the unit is V.

8. **:MEASure:VAVerage?**

Command Format:

:MEASure:VAVerage? [<source>]

Function Explanation:

The command queries the average value of the waveform under measure. <source> could be CHANnel1 or CHANnel2.

Returned Format:

Such as: 4.01e+01, the unit is V.

9. **:MEASure:VRMS?**

Command Format:

:MEASure:VRMS? [<source>]

Function Explanation:

The command queries the root-mean-square value of the waveform under measure. <source> could be CHANnel1 or CHANnel2.

Returned Format:

Such as: 1.78e+03, the unit is V.

10. **:MEASure:OVERshoot?**

Command Format:

:MEASure:OVERshoot? [<source>]

Function Explanation:
The command queries the overshoot of the waveform under measure.
<source> could be CHANnel1 or CHANnel2.

Returned Format:
Such as: 1.74e-02.

11. :MEASure:PREShoot?

Command Format:
:MEASure:PREShoot? [<source>]

Function Explanation:
The command queries the preshoot of the waveform under measure. <source>
could be CHANnel1 or CHANnel2.

Returned Format:
Such as: 0.00e+00.

12. :MEASure:FREQuency?

Command Format:
:MEASure:FREQuency? [<source>]

Function Explanation:
The command queries the frequency of the waveform under measure.
<source> could be CHANnel1 or CHANnel2.

Returned Format:
Such as: 1.00e+03, the unit is Hz.

13. :MEASure:RISetime?

Command Format:
:MEASure:RISetime? [<source>]

Function Explanation:
The command queries the rising time of the waveform under measure. <source> could be CHANnel1 or CHANnel2.

Returned Format:
Such as: 3.10e-04, the unit is s.

14. :MEASure:FALLtime?

Command Format:
:MEASure:FALLtime? [<source>]

Function Explanation:
The command measures the falling time of the waveform. <source> could be CHANnel1 or CHANnel2.

Returned Format:
Such as: 3.10e-04, the unit is s.

15. :MEASure:PERiod?

Command Format:
:MEASure:PERiod? [<source>]

Function Explanation:
The command queries the period of the waveform under measure. <source> could be CHANnel1 or CHANnel2.

Returned Format:
Such as: 1.00e-03, the unit is s.

16. :MEASure:PWIDth?

Command Format:
:MEASure:PWIDth? [<source>]

Function Explanation:
The command queries the positive pulse width of the waveform under measure.
<source> could be CHANnel1 or CHANnel2.

Returned Format:
Such as: 5.00e-04, the unit is s.

17. :MEASure:NWIDth?

Command Format:
:MEASure:NWIDth? [<source>]

Function Explanation:
The command queries the negative pulse width of the waveform under measure.
<source> could be CHANnel1 or CHANnel2.

Returned Format:
Such as: 5.00e-04, the unit is s.

18. :MEASure:PDUTycycle?

Command Format:
:MEASure:PDUTycycle? [<source>]

Function Explanation:
The command queries the positive duty cycle of the waveform under measure.
<source> could be CHANnel1 or CHANnel2.

Returned Format:
Such as: 5.00e-01.

19. :MEASure:NDUTycycle?

Command Format:
:MEASure:NDUTycycle? [<source>]

Function Explanation:
The command queries the negative duty cycle of the waveform under measure.
<source> could be CHANnel1 or CHANnel2.

Returned Format:
Such as: 5.00e-01.

20. :MEASure:PDELay?

Command Format:
:MEASure:PDELay? [<source>]

Function Explanation:
The command queries the delay relative to rising edge of channel1 or channel 2.
<source> could be CHANnel1 or CHANnel2.

Returned Format:
Such as: <-1.00e-04.

21. :MEASure:NDELay?

Command Format:
:MEASure:NDELay? [<source>]

Function Explanation:
The command queries the delay relative to falling edge of channel1 or channel 2.
<source> could be CHANnel1 or CHANnel2.

Returned Format:
Such as: <1.00e-04.

22. :MEASure:TOTal

Command Format:
:MEASure:TOTal {ON|OFF}
:MEASure:TOTal?

Function Explanation:

The commands set and query the On/Off state of all measurement functions.

Returned Format:

The query returns ON or OFF.

Example:

:MEAS:TOT ON   Turn on all measurements.

:MEAS:TOT?   The query returns ON.

23. :MEASure:SOURce

Command Format:

:MEASure:SOURce <source>

:MEASure:SOURce?

Function Explanation:

The commands select and query the current measured channel. <source> could be CHANnel1 or CHANnel2.

Returned Format:

The query returns CH1 or CH2.

Example:

:MEAS:SOUR CHAN1   Select the waveform form CH1 to measure.

:MEAS:SOUR?   The query returns CH1

WAVeform Command

WAVeform Command reads the waveform data on the screen of the oscilloscope and returns up to 1024 data.

WAVeform Command includes:

● :WAVeform:DATA?
● :WAVeform:POINts:MODE

The detailed information of each command are given as follows:

1.   :WAVeform:DATA?

Command Format:

:WAVeform:DATA? [<source>]

Function Explanation:

The command reads the waveform data in the specified source. <source> could be CHANnel1, CHANnel2, DIGital, MATH or FFT.

Returned Format:

The query returns 1024 data.

2.   :WAVeform:POINts:MODE

Command Format:

:WAVeform:POINts:MODE <points_mode>
:WAVeform:POINts:MODE?

Function Explanation:

This command sets the mode of waveform points. <points_mode> can be: NORMal, MAXimum or RAW.

Data points returned by **:WAVeform:DATA?** in different modes:

	NORmal	**RAW**		**MAX**
		Normal Memory	Long Memory	In **RUN** state, MAX is the same with NORMal; in **STOP** state, MAX is the same with RAW.
MATH	600	600	600	
FFT	512	512	512	
CHx	600	8192(8k)	524288(512k)	
Half-Channel CHx*	600	16384(16k)	1048576(1M)	
DIGITAL	600	16384(16k)	1048576(1M)	

*Note: Half-channel means only one channel is open and the MATH is closed.

Returned Format:

The query returns NORMal, MAXimum or RAW.

Example:
  :WAV:POIN:MODE NORM   Set the mode as NORMal.
  :WAV:POIN:MODE?   Return NORMal.

LA Commands

LA Commands are used for controlling the acquisition and analysis to digital signals executed by logic analyzer.

LA Commands include:

- :LA:DISPlay
- :DIGital<n>:TURN
- :DIGital<n>:POSition
- :LA:THReshold
- :LA:POSition:RESet
- :LA:GROUp
- :LA:GROUp<n>:SIZe

The detailed information of each command are given as follows:

1. :LA:DISPlay

Command Format:
:LA:DISPlay {ON|OFF}
:LA:DISPlay?

Function Explanation:
The commands set and query the On/Off state of digital channel.

Returned Format:
The query returns ON or OFF.

Example:
:LA:DISP ON   Open the digital channel.
:LA:DISP?   The query returns ON.

2. :DIGital<n>:TURN

Command Format:
:DIGital<n>:TURN {ON|OFF}
:DIGital<n>:TURN?

Function Explanation:
The commands set and query the On/Off state of specified channel in digital channel. The range of <n> is 0～15.

Returned Format:
The query returns ON or OFF.

Example:
:DIG2:TURN ON   Open the digital channel 2.
:DIG2:TURN?   The query returns ON.

3. :DIGital<n>:POSition

Command Format:
:DIGital<n>:POSition <position>

:DIGital<n>:POSition?

Function Explanation:
The commands set and query the waveform display position of specified channel.
<n> could be 0~15.
When choose Big-waveform, up to 8 waveforms could be displayed on the
screen, and the range of <position> is 0~7;
when choose Small-waveform, up to 16 waveforms could be displayed on the
screen, and the range of <position> is 0~15.

Returned Format:
The query returns the setting value of <position>.

Example:

:DIG2:POS 3   Setup the waveform of digital channel2 displays at the
                     position of D3.
:DIG2:POS?   The query returns 3.

4.   :LA:THReshold

Command Format:
:LA:THReshold<value>[suffix]
:LA:THReshold?

Function Explanation:
The commands set and query the type of threshold of digital channel. <value>
could be TTL, CMOS, ECL or User (User-defined, the range is -8.00~+8.00).
suffix could be V or mV, the minimum interval is 10mV.

Example:

:LA:THR COMS   Setup the type of threshold as COMS.
:LA:THR?   The query returns COMS.

5.   :LA:POSition:RESet

Command Format:
:LA:POSition:RESet

Function Explanation:
The command resets the waveform display position, that is to get the jumbled waves on the screen return to normal.

6.   **:LA:GROUp**

Command Format:
:LA:GROUp<n> {ON|OFF}
:LA:GROUp<n>?

Function Explanation:
The commands set and query the On/Off state of GROUp1 (D7~D0) and GROUp2 (D15~D8). <n> could be 1 or 2.

Returned Format:
The query returns ON or OFF.

Example:
:LA:GROU1 ON   Turn on the group D7~D0.
:LA:GROU1?   The query returns ON.

7.   **:LA:GROUp<n>:SIZe**

Command Format:
:LA:GROUp<n>:SIZe <size>
:LA:GROUp<n>:SIZe?

Function Explanation:
The commands set and query the waveform size from group. <n> could be 1 or 2. <size> could be Small or Big, which represents to display 16 channels or 8 channels.

Returned Format:
The query returns SMALL or BIG.

Example:
:LA:GROU1:SIZ S   Setup the waveform size from GROUp1 as Small.
:LA:GROU1:SIZ?   The query returns SMALL.

KEY Commands

KEY Commands are used for controlling the buttons and knobs on the panel of DS1000E，DS1000D series digital oscilloscopes.

KEY Commands include:

- :KEY:LOCK
- :KEY:RUN
- :KEY:AUTO
- :KEY:CHANnel1
- :KEY:CHANnel2
- :KEY:MATH
- :KEY:REF
- :KEY:F1
- :KEY:F2
- :KEY:F3
- :KEY:F4
- :KEY:F5
- :KEY:MNUoff
- :KEY:MEASure
- :KEY:CURSor
- :KEY:ACQuire
- :KEY:DISPlay
- :KEY:STORage
- :KEY:UTILity
- :KEY:MNUTIME
- :KEY:MNUTRIG
- :KEY:Trig%50

- :KEY:FORCe
- :KEY:V_POS_INC
- :KEY:V_POS_DEC
- :KEY:V_SCALE_INC
- :KEY:V_SCALE_DEC
- :KEY:H_SCALE_INC
- :KEY:H_SCALE_DEC
- :KEY:TRIG_LVL_INC
- :KEY:TRIG_LVL_DEC
- :KEY:H_POS_INC
- :KEY:H_POS_DEC
- :KEY:PROMPT_V
- :KEY:PROMPT_H
- :KEY:FUNCtion
- :KEY:+FUNCtion
- :KEY:-FUNCtion
- :KEY:LA
- :KEY:PROMPT_V_POS
- :KEY:PROMPT_H_POS
- :KEY:PROMPT_TRIG_LVL
- :KEY:OFF

The detailed information of each command are given as follows:

1.   :KEY:LOCK

Command Format:
:KEY:LOCK {ENABle|DISable}
:KEY:LOCK?

Function Explanation:
This command is used to enable/disable the buttons function on the front panel (except for "Force").

Returned Format:
The query returns ENABLE or DISABLE.

Example:
:KEY:LOCK ENAB   Enable the buttons on the front panel
:KEY:LOCK?   The query returns ENABLE.

2.   :KEY:RUN

Command Format:
:KEY:RUN

Function Explanation:
The command controls the operating mode of the oscilloscope. The running state will switch between ON and OFF if send this command continuously.

3.   :KEY:AUTO

Command Format:
:KEY:AUTO

Function Explanation:
The command sets every control values of the oscilloscope automatically in order to output appropriate waveforms for better observation.

4.   :KEY:CHANnel1

Command Format:
:KEY:CHANnel1

Function Explanation:
The command enables or disables Channel 1 and its menu display. The On/Off state of the channel along with its menu switched once you send this command continuously.

5.   :KEY:CHANnel2

Command Format:
:KEY:CHANnel2

Function Explanation:
The command enables or disables Channel 2 and its menu display. The On/Off state of the channel along with its menu switched once you send this command continuously.

6.   :KEY:MATH

Command Format:
:KEY:MATH

Function Explanation:
The command enables or disables Math function and its menu display. The On/Off state of this function along with its menu switched once you send this command continuously.

7.   :KEY:REF

Command Format:
:KEY:REF

Function Explanation:
The command enables or disables Waveform Reference and its menu display. The On/Off state of this function along with its menu switched once you send

this command continuously.

8. :KEY:F1

Command Format:
:KEY:F1

Function Explanation:
The command controls the option corresponds to F1. If some options exist in the drop-down menu relative to this key, repeat this command to select those options circularly.

9. :KEY:F2

Command Format:
:KEY:F2

Function Explanation:
The command controls the option corresponds to F2. If some options exist in the drop-down menu relative to this key, repeat this command to select those options circularly.

10. :KEY:F3

Command Format:
:KEY:F3

Function Explanation:
The command controls the option corresponds to F3. If some options exist in the drop-down menu relative to this key, repeat this command to select those options circularly.

11. :KEY:F4

Command Format:
:KEY:F4

Function Explanation:
The command controls the option corresponds to F4. If some options exist in the drop-down menu relative to this key, repeat this command to select those options circularly.

12. :KEY:F5

Command Format:
:KEY:F5

Function Explanation:
The command controls the option corresponds to F5. If some options exist in the drop-down menu relative to this key, repeat this command to select those options circularly.

13. :KEY:MNUoff

Command Format:
:KEY:MNUoff

Function Explanation:
The command enables or disables the current menu display. The On/Off state of this function switched once you send this command continuously.

14. :KEY:MEASure

Command Format:
:KEY:MEASure

Function Explanation:
The command enables or disables Measure function and its menu display. The On/Off state of this function along with its menu switched once you send this command continuously.

15. :KEY:CURSor

Command Format:

:KEY:CURSor

Function Explanation:

The command enables or disables Cursor measurement and its menu display.
If send this command continuously, the oscilloscope will switch among Turn off
cursor, Manual, Track and Auto.

16. :KEY:ACQuire

Command Format:

:KEY:ACQuire

Function Explanation:

The command enables or disables Sample function and its menu display. The
On/Off state of this function along with its menu switched once you send this
command continuously.

17. :KEY:DISPlay

Command Format:

:KEY:DISPlay

Function Explanation:

The command enables or disables Display function and its menu display. The
On/Off state of this function along with its menu switched once you send this
command continuously.

18. :KEY:STORage

Command Format:

:KEY:STORage

Function Explanation:

The command enables or disables Storage function and its menu display. The
On/Off state of this function along with its menu switched once you send this

command continuously.

19. :KEY:UTILity

Command Format:
:KEY:UTILity

Function Explanation:
The command enables or disables Utility function and its menu display. The On/Off state of this function along with its menu switched once you send this command continuously.

20. :KEY:MNUTIME

Command Format:
:KEY:MNUTIME

Function Explanation:
The command enables or disables Horizontal system and its menu display. The On/Off state of this function along with its menu switched once you send this command continuously.

21. :KEY:MNUTRIG

Command Format:
:KEY:MNUTRIG

Function Explanation:
The command enables or disables Trigger and its menu display. The On/Off state of this function along with its menu switched once you send this command continuously.

22. :KEY:Trig%50

Command Format:
:KEY:Trig%50

Function Explanation:
The command sets the trigger level to the midpoint of the signal amplitude.

23. :KEY:FORCe

Command Format:
:KEY:FORCe

Function Explanation:
The command unlocks the remote control.

24. :KEY:V_POS_INC

Command Format:
:KEY:V_POS_INC

Function Explanation:
The command increases the vertical scale of the current channel in uniformly spaced.

25. :KEY:V_POS_DEC

Command Format:
:KEY:V_POS_DEC

Function Explanation:
The command reduces the vertical scale of the current channel in uniformly spaced.

26. :KEY:V_SCALE_INC

Command Format:
:KEY:V_SCALE_INC

Function Explanation:
The command is used to increase the vertical scale of the current channel.

27. :KEY:V_SCALE_DEC

Command Format:
:KEY:V_SCALE_DEC

Function Explanation:
The command is used to reduce the vertical scale of the current channel.

28. :KEY:H_SCALE_INC

Command Format:
:KEY:H_SCALE_INC

Function Explanation:
The command reduces the horizontal scale of the current channel in 5-2-1.

29. :KEY:H_SCALE_DEC

Command Format:
:KEY:H_SCALE_DEC

Function Explanation:
The command increases the horizontal scale of the current channel in 1-2-5.

30. :KEY:TRIG_LVL_INC

Command Format:
:KEY:TRIG_LVL_INC

Function Explanation:
The command increases the trigger level in uniformly spaced.

31. :KEY:TRIG_LVL_DEC

Command Format:

:KEY:TRIG_LVL_DEC

Function Explanation:
The command reduces the trigger level in uniformly spaced.

32. :KEY:H_POS_INC

Command Format:
:KEY:H_POS_INC

Function Explanation:
The command increases the horizontal offset of the current channel in uniformly
spaced.

33. :KEY:H_POS_DEC

Command Format:
:KEY:H_POS_DEC

Function Explanation:
The command reduces the horizontal offset of the current channel in uniformly
spaced.

34. :KEY:PROMPT_V

Command Format:
:KEY:PROMPT_V

Function Explanation:
The command switches the adjust mode of the vertical scale to Coarse or Fine.
Coarse steps in 1-2-5 and Coarse steps in uniformly spaced.

35. :KEY:PROMPT_H

Command Format:
:KEY:PROMPT_H

Function Explanation:
The command enables or disables the Delayed function. The On/Off state of the analyzer switched once you send this command continuously.

36. :KEY:FUNCtion

Command Format:
:KEY:FUNCtion

Function Explanation:
The command enables the multi-function knob.

37. :KEY:+FUNCtion

Command Format:
:KEY:+FUNCtion

Function Explanation:
The command increases the offset of the multi-function knob.

38. :KEY:-FUNCtion

Command Format:
:KEY:-FUNCtion

Function Explanation:
The command reduces the offset of the multi-function knob.

39. :KEY:LA

Command Format:
:KEY:LA

Function Explanation:
The command enables or disables the logic analyzer built-in the oscilloscope.

The On/Off state of the analyzer switched once you send this command continuously.

40. :KEY:PROMPT_V_POS

Command Format:
:KEY:PROMPT_V_POS

Function Explanation:
The command resets the vertical offset to zero.

41. :KEY:PROMPT_H_POS

Command Format:
:KEY:PROMPT_H_POS

Function Explanation:
The command resets the trigger offset (or the delay offset) to zero.

42. :KEY:PROMPT_TRIG_LVL

Command Format:
:KEY:PROMPT_TRIG_LVL

Function Explanation:
The command resets the position of trigger level to the center of the screen.

43. :KEY:OFF

Command Format:
:KEY:OFF

Function Explanation:
The command turns off the CH1, CH2, MATH, REF and LA channel one by one through sending the command continuously.

Other Commands

Besides above basic commands, also there are other commands used for system language, frequency counter and beep control.

Other Commands include:

- :INFO:LANGuage
- :COUNter:ENABle
- :BEEP:ENABle
- :BEEP:ACTion

The detailed information of each command are given as follows:

1.   :INFO:LANGuage

Command Format:
:INFO:LANGuage <lang>
:INFO:LANGuage?

Function Explanation:
The commands set and query the type of system language. <lang> could be SIMPlifiedChinese, TRADitionalChinese, ENGLish, KORean, JAPanese, FRENch, GERMan, RUSSian, SPANish or PORTuguese.

Returned Format:
The query returns Simplified Chinese, Traditional Chinese, English, Korean, Japanese, French, German, Russian, Spanish or Portuguese.

Example:
:INFO:LANG SIMP   Setup the system language as SIMPlifiedChinese.
:INFO:LANG?   The query returns Simplified Chinese.

2.   :COUNter:ENABle

Command Format:
:COUNter:ENABle {ON|OFF}
:COUNter:ENABle?

Function Explanation:
The commands set and query the On/Off state of the cymometer of the oscilloscope.

Returned Format:
The query returns ON or OFF.

Example:
:COUN:ENAB ON   Turn on the frequency counter.
:COUN:ENAB?   The query returns ON.

3. :BEEP:ENABle

Command Format:

:BEEP:ENABle {ON|OFF}

:BEEP:ENABle?

Function Explanation:

The commands set and query the On/Off state of the beeper.

Returned Format:

The query returns ON or OFF.

Example:

:BEEP:ENAB ON   Turn on the beeper.

:BEEP:ENAB?   The query returns ON.

4.   :BEEP:ACTion

Command Format:

:BEEP:ACTion

Function Explanation:

The command makes the oscilloscope buzzed once (no matter whether the beeper is on or not).

Chapter 3   Programming Examples

This chapter lists some programming examples about how to realize the basic functions of the oscilloscope in the development environments of Visual C++ 6.0, Visual Basic 6.0 or LabView 8.6. Users can programme the oscilloscope based on the USB driver provided by RIGOL or VISA library. For VISA programming, your computer must have been installed the NI-VISA which can be downloaded from http://www.ni.com. In the example, we assume NI-VISA is installed under C:\Program Files\IVI Foundation\VISA.

Prepare for Programming

In this text, we use USB data cable to achieve the communication between DS1000E, DS1000D and PC. See the figure below.
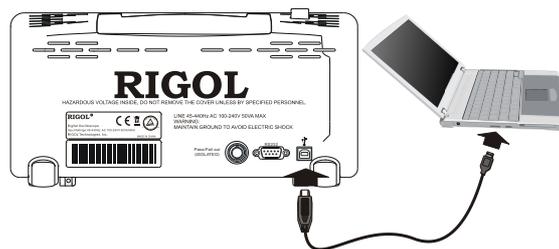
After successful connection, turn on the instrument, a dialog will guide you to install the driver of "Rigol USB Test and Measurement Device" on the PC. For VISA programming, the driver should be "USB Test and Measurement Device".

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

Programming based on RIGOL USB Driver

Program in Visual C++ 6.0

This example shows you how to create a Demo to send a command to query and return. Open Visual C++ 6.0, take the following steps:

1. Create a project **IDNDemo** based on MFC.

2. Add controls: ☐ Button、⊞Combo Box and **abl** Edit Box.

3. Modify the properties of the controls.

1) Name the Button to be "**Send and Read**". See the figure below.

2) Choose **Data** in the property of **Combo Box**, input two commands manually:
*IDN?
:ACQ:TYPE?
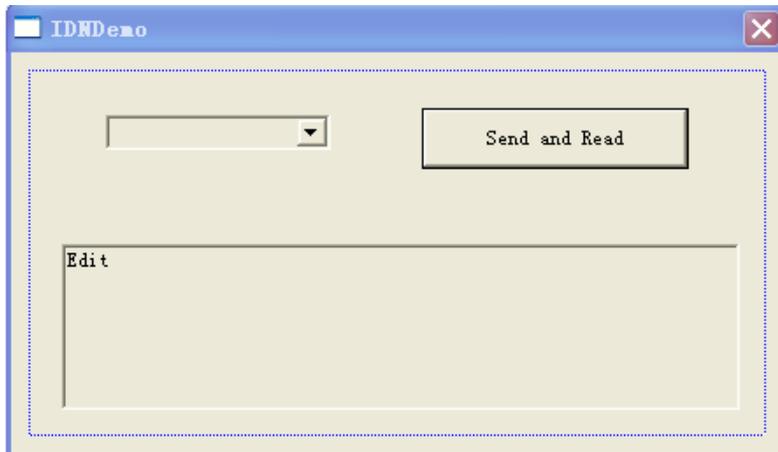
3) Respectively add two variables **m_combox** and **m_receive** for the controls of **Combo Box** and **Edit Box**. See the figure below.
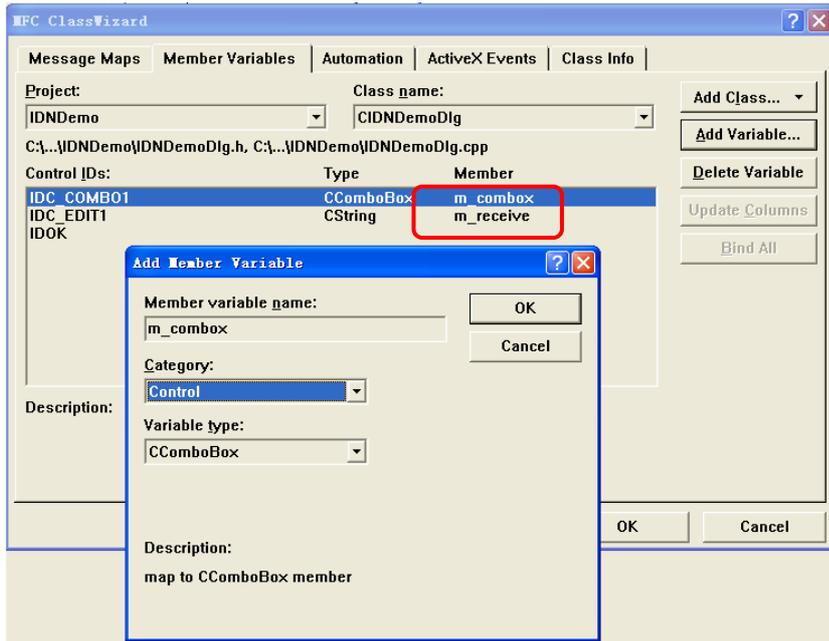
Note: Please select "**control**" of Category when adding "m_combox", and select "**value**" when adding "m_receive".

4. Copy the Dynamic Link Library **RigolTMCUsb_UI.dll** from **Demo files** into the new project for easier use.

Explanation: In IDNDemo, only three functions from this ddl are required, which are: **GetTMCDeviceNum**, **WriteUSB** and **ReadUSB** (the function has been mentioned in the header files **Rigolusb.h**);

Now, take **GetTMCDeviceNum** as an example to introduce how to recall these three functions:

```
HMODULE module = LoadLibrary("RigolTMCUsb_UI.dll");
typedef int (CALLBACK* pGetDeviceNum)();
if(module)
{
    pGetDeviceNum GetDevice
    = (pGetDeviceNum)GetProcAddress(module,"GetTMCDeviceNum");
    int j = GetDevice(); // enter
}
```

5. Add the codes.

Enter the programming environment. First of all, declare "void SendToUSB(CString cmd)" and "unsigned char* Read_USB(void)" in header file, then add the following codes:

```
//send setting command
void SendToUSB(CString cmd)
{
    int temp,j;
    CString cmdstr;
    cmdstr = cmd;
    unsigned char * strInput;

    //recall the GetTMCDeviceNum    and  WriteUSB  function  from
RigolTMCUsb_UI.dll
    HMODULE module = LoadLibrary("RigolTMCUsb_UI.dll");

     typedef bool (CALLBACK* pWrite)(unsigned long, unsigned char,unsigned
char,unsigned long,unsigned char*);
     typedef int (CALLBACK* pGetDeviceNum)();

    temp = cmdstr.GetLength();
    strInput = (unsigned char*)cmdstr.GetBuffer(10);

    if(module)
    {
        pGetDeviceNum GetDevice =
(pGetDeviceNum)GetProcAddress(module,"GetTMCDeviceNum");
        pWrite Write = (pWrite)GetProcAddress(module,"WriteUSB");

        j = GetDevice();

        if( j > 0 )
        {
            //send a command to the oscilloscope
            //the definition of interface has been mentioned in "Rigolusb.h"
```

```
                Write(0,1,1,temp,strInput);
            }
        }
    }
    //acquire Returned Format from the oscilloscope
    unsigned char* Read_USB( void )
    {
        unsigned long infoSize;
        //open a space to save the Returned Format
        int len = 100;
        unsigned char* strInput = new unsigned char [100];
        CString str;

        //recal the function ReadUSB and WriteUSB from RigolTMCUsb_UI.dll
        HMODULE module = LoadLibrary("RigolTMCUsb_UI.dll");

        typedef bool (CALLBACK* pWrite)(unsigned long, unsigned char,unsigned
    char,unsigned long,unsigned char*);
        typedef bool (CALLBACK* pRead)(unsigned long,unsigned long*,unsigned
    char*);

        if(module)
        {
            pWrite Write = (pWrite)GetProcAddress(module,"WriteUSB");
            pRead Read = (pRead)GetProcAddress(module,"ReadUSB");

            //send a command to ask for value, after the success, a value could be
    obtained from the oscilloscope.
            Write(0,2,2,len,0);
            //read Returned Format from the oscilloscope
            Read(0,&infoSize,strInput);
        }
        ///open a space to save the effective Returned Format
        unsigned char *buffer = new unsigned char[infoSize];

        memcpy(buffer,strInput,infoSize);
        buffer[infoSize] = 0;
        delete []strInput;
```

　　　　//return the character string getting from the oscilloscope
　　　　return buffer;
}

6. Dblclick the **Button** and add the following codes:
　　unsigned char* readInfo;
　　CString strInput;
　　//char* stringTemp;
　　CString s,strTemp;

　　m_combox.GetLBText(m_combox.GetCurSel(),strTemp);
　　strInput = (char *)(LPCTSTR)strTemp;
　　SendToUSB(strInput);
　　readInfo = Read_USB();

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

7. Save, compile and run the project, you will get a single executable program about demo. When the oscilloscope has been successfully connected with PC, select a command "*IDN?" from Combo Box to send, then click the "Send and Read" button, you will get the query results as the following figure:
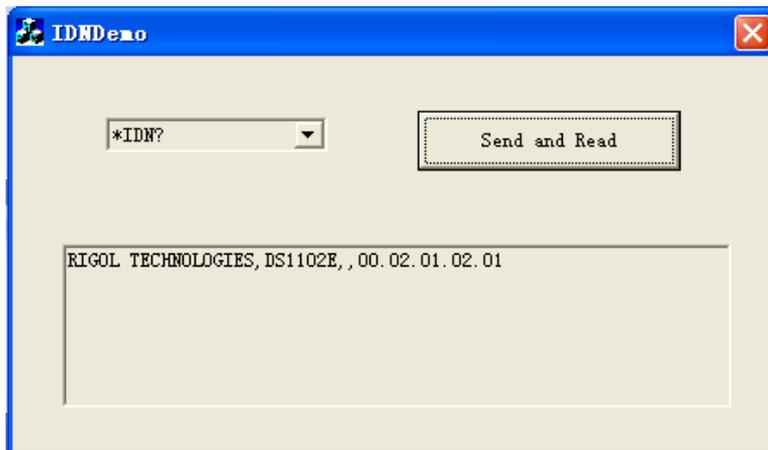
Program in Visual Basic 6.0

Example 1: Query and set the channel status

This example shows you how to create a Demo to query or set the channel status. Open Visual Basic 6.0, take the following steps:

1. Create a **Standard EXE** project and copy the Dynamic Link Library **RigolTMCUsb_UI.dll** from **Demo files** into the new project for later use.

2. Named Form1 "**Demo**". Then add three **CommandButtons** (CH1, CH2 and MATH) and three **Labels** for showing the status of these buttons respectively (the label colour varies with the channel status, that is, when the channel is on, the label shows relative colour; on the contrary, if the channel is off, the colour of label will be gray), see the figure below:
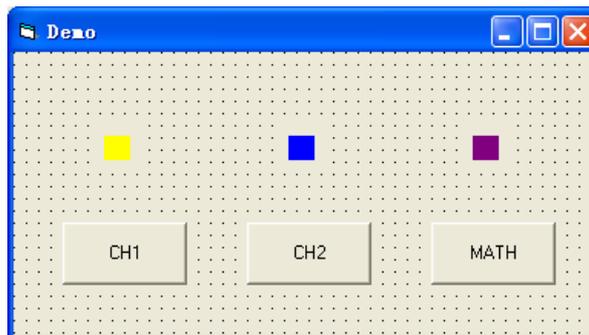
3. Enter into the programming environment, then, quote the functions (**ReadUSB,WriteUSB,GetTMCDeviceNum**) in the Dynamic Link Library **RigolTMCUsb_UI.dll** and add the following codes in the area of project code:

Private Declare Function ReadUSB Lib "RigolTMCUsb_UI.dll" (ByVal Index As Long, ByVal lpszLength As Long, ByVal lpszBuffer As Long) As Boolean

Private Declare Function WriteUSB Lib "RigolTMCUsb_UI.dll" (ByVal Index As Long, ByVal msgid As Long, ByVal tag As Long, ByVal length As Long, ByVal lpszBuffer As Long) As Boolean

Private Declare Function GetDeviceNum Lib "RigolTMCUsb_UI.dll" () As Long

4. Add a subfunction as follows to send commands and get returned values.

```
'send a setting command
Sub SendToUSB(cmd As String)
    Dim retcode As Boolean
    Dim send_buf(256) As Byte
    Dim temp As Long
    Dim cmdstr As String

    cmdstr = cmd

    temp = Len(cmdstr)
    For i = 0 To temp - 1
        tempStr = Mid(cmdstr, i + 1, 1)
        send_buf(i) = Asc(tempStr)
    Next i
    'send a command to the oscilloscope
    'the definition of interface has been mentioned in Rigolusb.h
    retcode = WriteUSB(0, 1, 1, Len(cmdstr), VarPtr(send_buf(0)))
End Sub

'/acquire Returned Format from the oscilloscope
Public Function Read_USB() As String

    Dim retcode As Boolean
    Dim rcv_buffer(256) As Byte
    Dim tmpstr As String
    Dim i, size As Long
    Dim rSize As Long

    rSize = 100
    tmpstr = ""
```

send a command to ask for value, after the success, a value could be obtained from the oscilloscope.

```
    retcode = WriteUSB(0, 2, 2, 256, 0)
    'read Returned Format from the oscilloscope
    retcode = ReadUSB(0, VarPtr(rSize), VarPtr(rcv_buffer(0)))
```

```
        For i = 0 To (rSize - 1)
            tmpstr = tmpstr + Chr(rcv_buffer(i))
        Next i

        Read_USB = tmpstr

   End Function
```

5. Dblclick CH1 and add the following codes:

```
Dim i As Long
Dim sendbuf As String
Dim readbuf As String

rSize = 100
sendbuf = ":CHAN1:DISP?"

'initialize the USB device
i = GetTMCDeviceNum

'send a query command and read the results
Call SendToUSB(sendbuf)
readbuf = Read_USB()

'confirm the light state of corresponding channel according to the Returned
```
Format
```
If readbuf = "ON" Then

' send a setting command
sendbuf = ":CHAN1:DISP 0"
Call SendToUSB(sendbuf)

Label1(0).ForeColor = &H808080    'gray

Else

sendbuf = ":CHAN1:DISP 1"
Call SendToUSB(sendbuf)
```

Label1(0).ForeColor = &HFFFF&   'yellow
End If

Note: Codes of CH2 and MATH are similar to CH1, users can modify homologous parameters to get their codes and add them to CH2 and MATH.

6. Dblclick Label and add the following codes:

Dim i As Long
Dim sendbuf As String
Dim readbuf As String

rSize = 100
sendbuf = ":CHAN1:DISP?"

i = GetTMCDeviceNum

Call SendToUSB(sendbuf)
readbuf = Read_USB()

If readbuf = "ON" Then

Label1(0).ForeColor = &HFFFF&   'yellow

Else

Label1(0).ForeColor = &H808080   'gray

End If

sendbuf = ":CHAN2:DISP?"

i = GetTMCDeviceNum

Call SendToUSB(sendbuf)
readbuf = Read_USB()
If readbuf = "ON" Then

```
Label1(1).ForeColor = &HFF0000        'blue

Else

Label1(1).ForeColor = &H808080   'gray
End If

sendbuf = ":MATH:DISP?"

i = GetTMCDeviceNum

Call SendToUSB(sendbuf)
readbuf = Read_USB()

If readbuf = "ON" Then

Label1(2).ForeColor = &H800080        'purple

Else

Label1(2).ForeColor = &H808080   'gray

End If
```

7. Save and run the project, you will get a single executable program about demo. When the oscilloscope has been successfully connected with PC, you can open/close each channel conveniently by clicking the button.
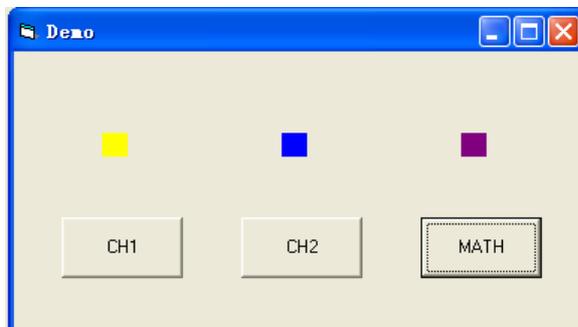
Example 2: Read the waveform data

This example shows you how to create a Demo to read the waveform data.
Open Visual Basic 6.0, take the following steps:

1. Create a **Standard EXE** project and copy the Dynamic Link Library
RigolTMCUsb_UI.dll from **Demo files** into the new project for easier use.

2. Named Form1 "**Demo**". Then add two **Commandbuttons** (CH1 and CH2) and
a **PictureBox** for showing the waveform data about CH1 and CH2, please see
the figure below:
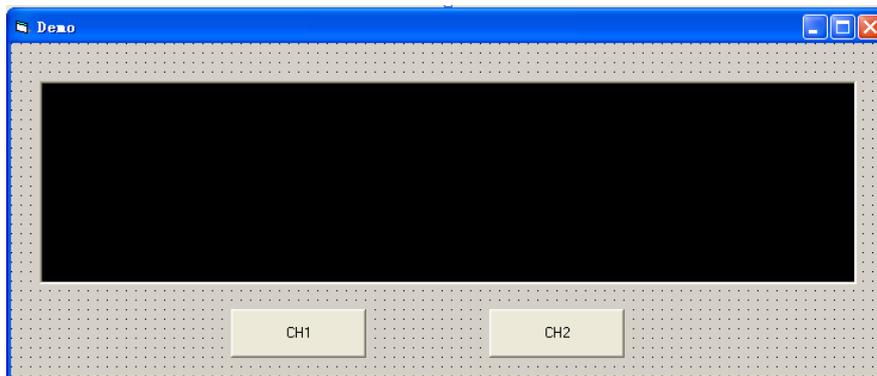
3. Project→Add Module, create a **Module** for declaring the global variable
rcv_buffer as follows:

Global rcv_buffer(10240) As Byte

Note: For the waveform data returned from the oscilloscope are too much to
save, please use this buffer.

4. Enter into the programming environment, then, quote the functions
(**ReadUSB,WriteUSB,GetDeviceNum**) in the Dynamic Link Library
RigolTMCUsb_UI.dll and add the following codes in the area of project code:

Private Declare Function ReadUSB Lib "RigolTMCUsb_UI.dll" (ByVal Index As
Long, ByVal lpszLength As Long, ByVal lpszBuffer As Long) As Boolean

Private Declare Function WriteUSB Lib "RigolTMCUsb_UI.dll" (ByVal Index As
Long, ByVal msgid As Long, ByVal tag As Long, ByVal length As Long, ByVal

lpszBuffer As Long) As Boolean
Private Declare Function GetDeviceNum Lib "RigolTMCUsb_UI.dll" () As Long

5. Add a subfunction as follows to send commands and get returned values.

```
'send a setting command
Sub SendToUSB(cmd As String)
    Dim retcode As Boolean
    Dim send_buf(256) As Byte
    Dim temp As Long
    Dim cmdstr As String

    cmdstr = cmd

    temp = Len(cmdstr)
    For i = 0 To temp - 1
        tempStr = Mid(cmdstr, i + 1, 1)
        send_buf(i) = Asc(tempStr)
    Next i
    'send a command to the oscilloscope
    ' the definition of interface has been mentioned in Rigolusb.h
    retcode = WriteUSB(0, 1, 1, Len(cmdstr), VarPtr(send_buf(0)))
End Sub

'acquire the Returned Value
Public Function Read_USB() As Long

    Dim retcode As Boolean
    Dim i, size As Long
    Dim rSize As Long

    rSize = 100

    'send the command to the oscilloscope
    retcode = WriteUSB(0, 2, 2, 10240, 0)

    'get value from the oscilloscope
    retcode = ReadUSB(0, VarPtr(rSize), VarPtr(rcv_buffer(0)))
```

```
      Read_USB = rSize

End Function
```

6. Dblclick CH1 and add the following codes:

```
Dim i As Long
Dim sendbuf As String
Dim DataLen As Long
Dim stepW As Integer
Dim stepH As Integer

sendbuf = ":WAV:DATA? CHAN1"

'initialize the USB device
i = GetTMCDeviceNum

'send a query command and read the results
Call SendToUSB(sendbuf)
DataLen = Read_USB()
stepW = Picture1.Width
stepH = Picture1.Height / 256

'clear the waveform
Picture1.Cls
For i = 0 To DataLen - 1

'link the waveform points and set the waveform color to yellow
Picture1.Line ((i * stepW / DataLen), (rcv_buffer(i)) * stepH)-(((i + 1) * stepW /
DataLen), (rcv_buffer((i + 1))) * stepH), &HFFFF&

Next i
```

Note: Codes of CH2 are similar to CH1, users can modify homologous parameters to get its codes and add them to CH2.

7. Save and run the project, you will get a single executable program about demo. When the oscilloscope has been successfully connected with PC, you can read the waveform data of each channel conveniently by clicking the button.
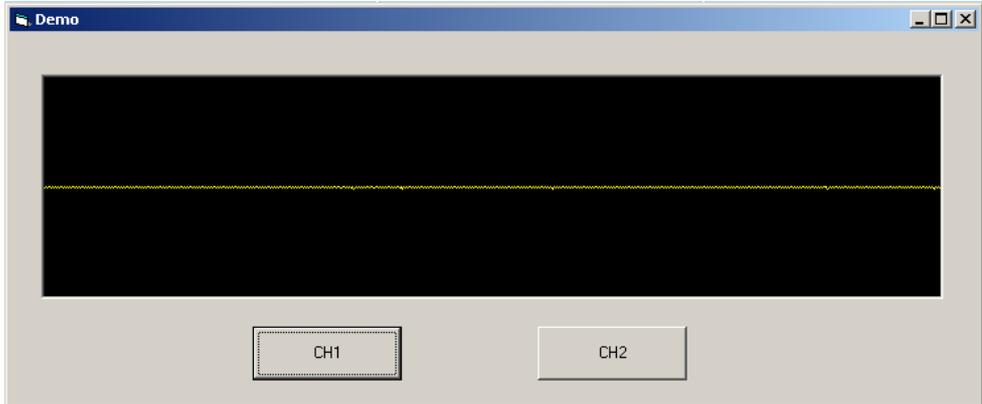
Programming based on VISA

Program in Visual C++ 6.0

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**.
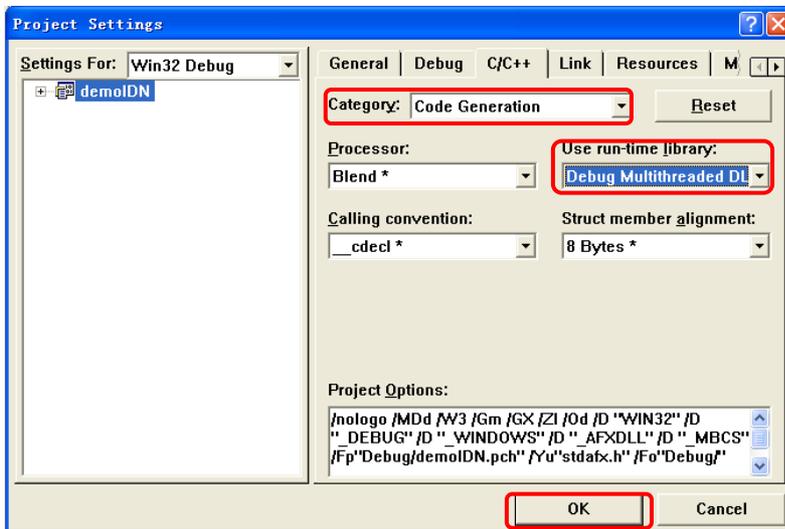
3. Choose **Project→Settings→Link**, add the file **"visa32.lib"** manually in **Object/library modules**.
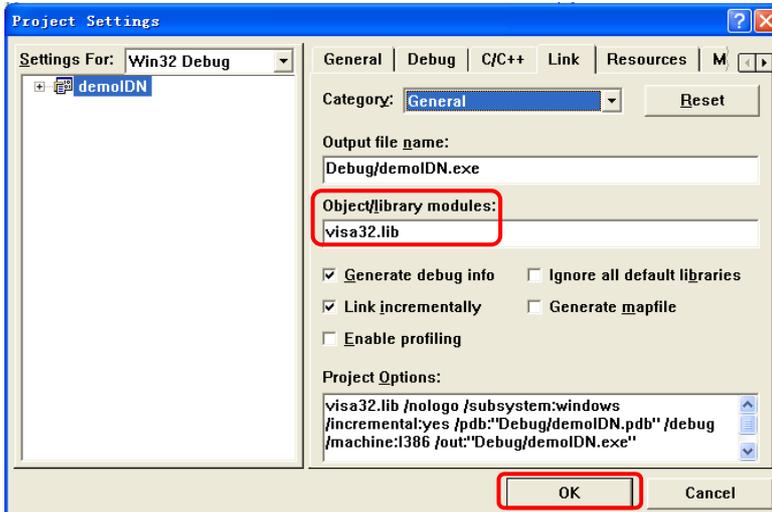
4. Choose **Tools→Options→Directories**; select **"Include files"** in **Show directories for**, and then dblclick the blank in **Directories** to add the path of "**Include"**: C:\Program Files\IVI Foundation\VISA\WinNT\include.

Select **"Library files"** in **Show directories for**, and then dblclick the blank in **Directories** to add the path of **"Lib"**:
C:\Program Files\IVI Foundation\VISA\WinNT\lib\msc.
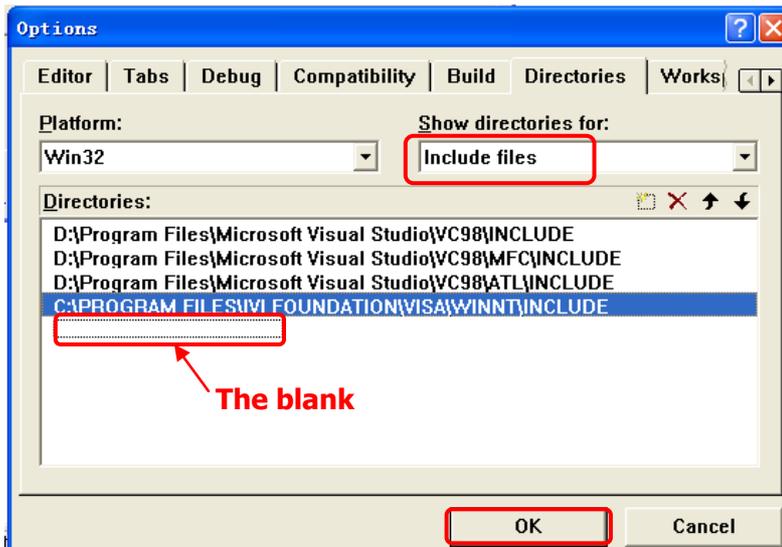
Note: At present, VISA library has been added successfully.

5. Add controls: **Text**, **Com box**, **Button** and **Edit**. See the figure below.
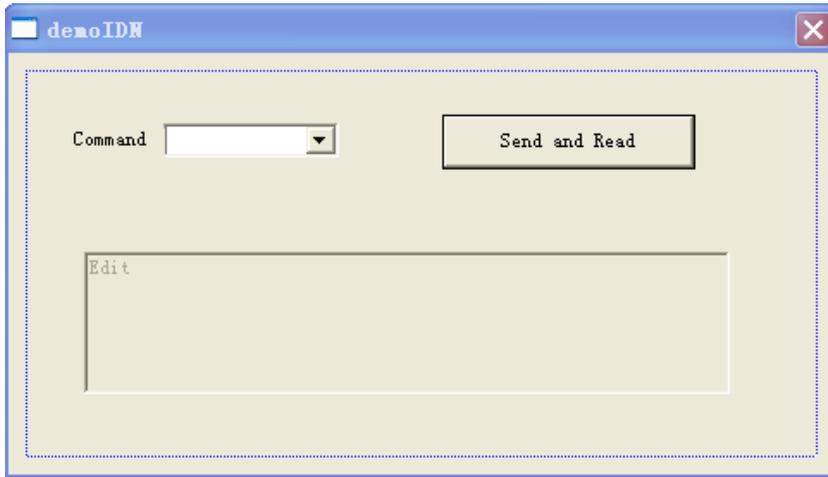
6. Modify the properties of the controls.

1) Name the Text to be "**Command**".

2) Choose **Data** in the property of **Com box**, input three commands manually:
   *IDN?
   *OPC?
   :ACQuire:TYPE?

3) Choose **General** in the property of **Edit** and select **Disable**.

4) Modify the name of **Button** such as: Send and Read.

7. Respectively add two variables **m_combox** and **m_receive** for the controls of **Com box** and **Edit**.
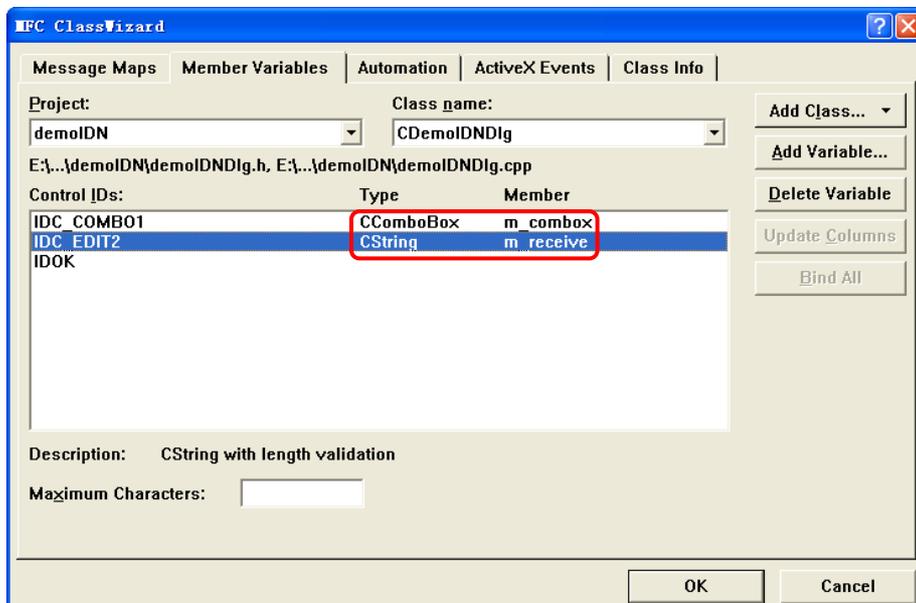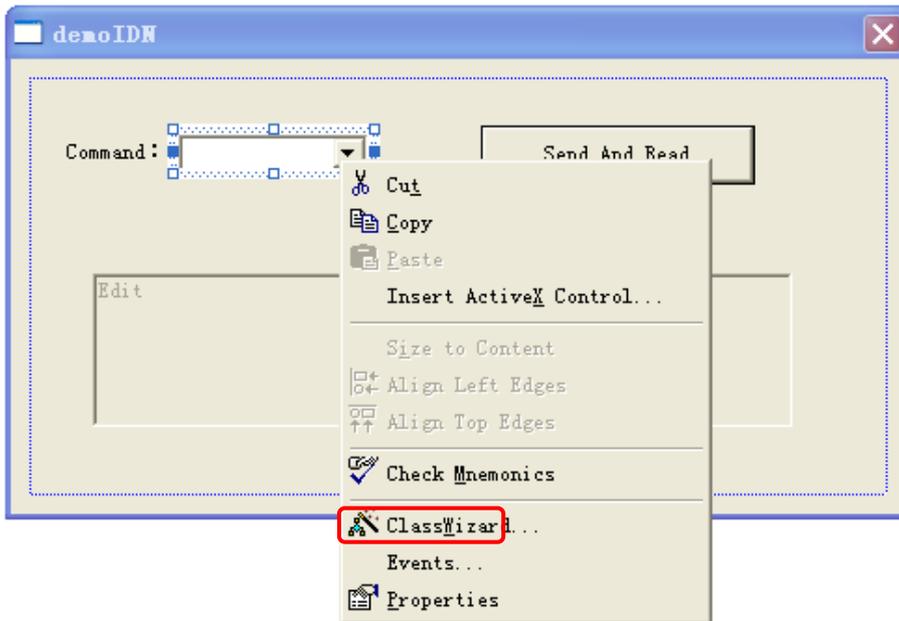
8. Add the codes.

Dblclick the **Button**, enter the programming environment. First of all, declare "#include <visa.h>" in header file, 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;

viOpenDefaultRM (&defaultRM);

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

// send the receiving commands
m_combox.GetLBText(m_combox.GetCurSel(),strTemp);
strTemp = strTemp + "\n";
stringTemp = (char *)(LPCTSTR)strTemp;
viPrintf (vi,stringTemp);

// read the result
viScanf (vi, "%t\n", &buf);

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

9. Save, build and run the project, you will get an EXE file. When the oscilloscope has been successfully connected with PC, choose a command such as ***IDN?** and click **"Send and Read"**, the oscilloscope will return the result.

Program in Visual Basic 6.0

Open Visual Basic 6.0, take the following steps:

1. Create a **Standard EXE** project.

2. Choose **Project→Add Module→Existing**; find the **"visa.bas"** file in the filefolder of **include** under the path of NI-VISA and add;
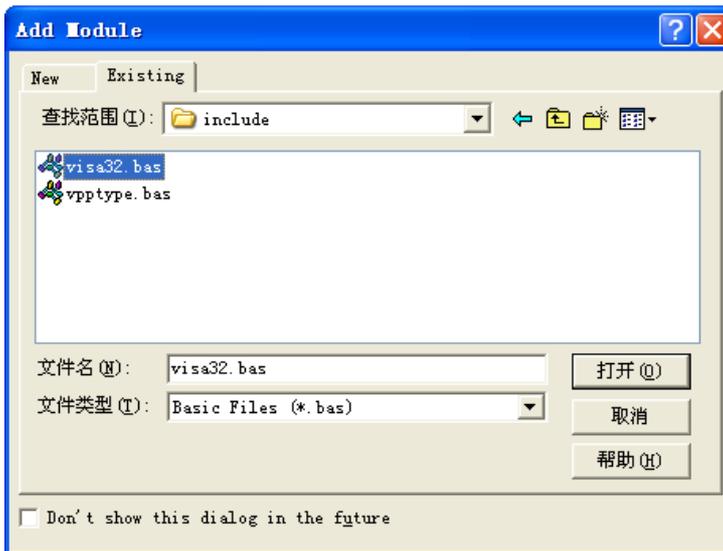
3. Add two **Command Buttons** and **Lables** to the demo, each button denotes each channel, and each Label denotes different states (yellow and light blue which is the channel's color indicates opening, while gray indicates close) of the channels. See the figure below.
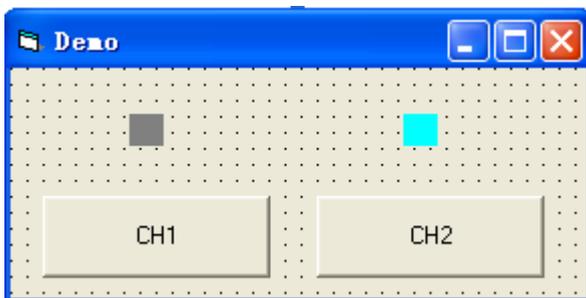
4. Choose **Project→Project1 Properties→General**, select **"Form1"** from the drop down box of **Startup Object**.

5. Dblclick **CH1** button to enter the programming environment, add the following codes to achieve the control to it. (for **CH2**, the method is similar)

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

 ' acquire USB resource of visa
 Call viOpenDefaultRM(defrm)
 Call viFindRsrc(defrm, "USB?*", list, nmatches, matches)

 ' open the equipment
 Call viOpen(defrm, matches, 0, 0, vi)

 ' send the command to query the state of CH1
 Call viVPrintf(vi, ":CHAN1:DISP?" + Chr$(10), 0)

 ' get the state of CH1
 Call viVScanf(vi, "%t", strRes)

 If Left(strRes, 2) = "ON" Then

 ' send the setting command
 Call viVPrintf(vi, ":CHAN1:DISP 0" + Chr$(10), 0)
 Label1(0).ForeColor = &H808080   ' gray

 Else

 Call viVPrintf(vi, ":CHAN1:DISP 1" + Chr$(10), 0)
 Label1(0).ForeColor = &HFFFF&   ' yellow

 End If

 ' close the resource
 Call viClose(vi)
 Call viClose(defrm)
```

6. Save and run the project, you will get a single executable program about demo. When the oscilloscope has been successfully connected with PC, you can open/close each channel conveniently by clicking the button.

Program in LabVIEW 8.6

Open LabVIEW 8.6, take the following steps:

1. Open **Block Diagram**; choose **Instrument I/O→VISA**; then separately add four functions: **"VISA Open"**, **"VISA Read"**, **"VISA Write"** and **"VISA Close"**. See the figure below.
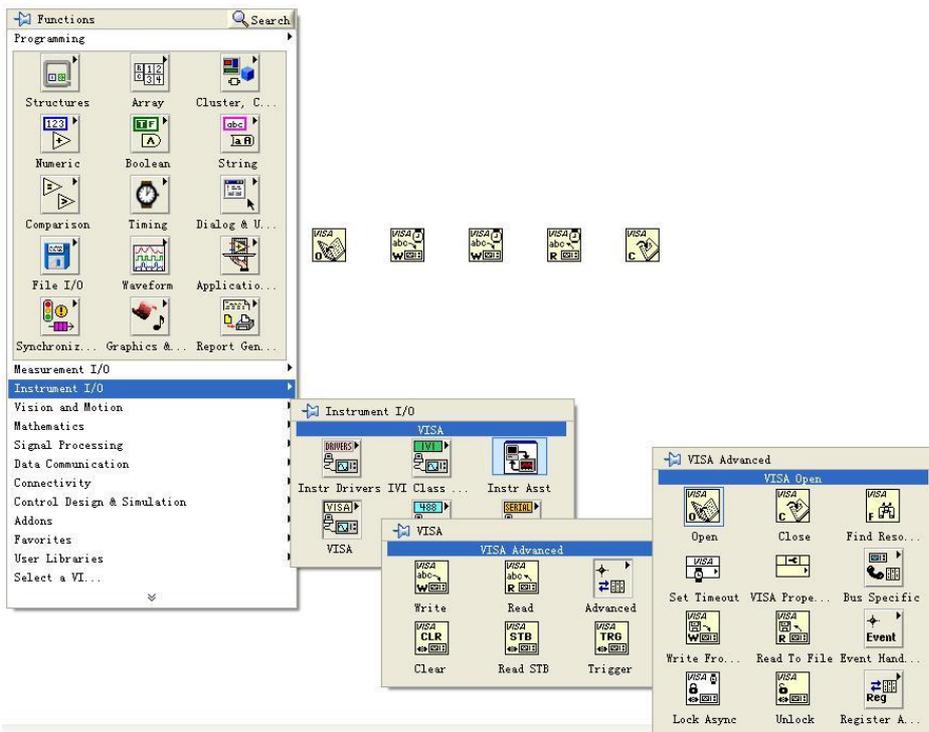
2. Move the mouse to the item of **"VISA resource name"** on the control of **"VISA Open"**; right-click the mouse to choose **Create→Control**. See the figure below.
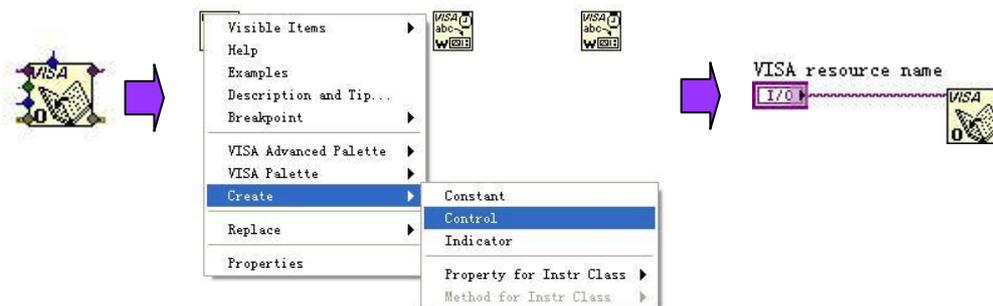
3. Separately connect **"VISA resource name"** with **"VISA resource name out"** and **"error out"** with **"error in"** of all the functions. See the figure below.

4. Add a textbox written with **":WAV:DATA?\n"** on the **"VISA Write"** control. while the latter reads the waveform data shown on the screen.
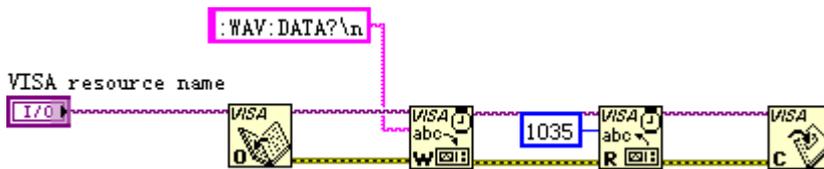
5. Open the **Front Panel**; choose **Modern→Graph→Waveform Graph** to add a **Waveform Graph** control. See the figure below.
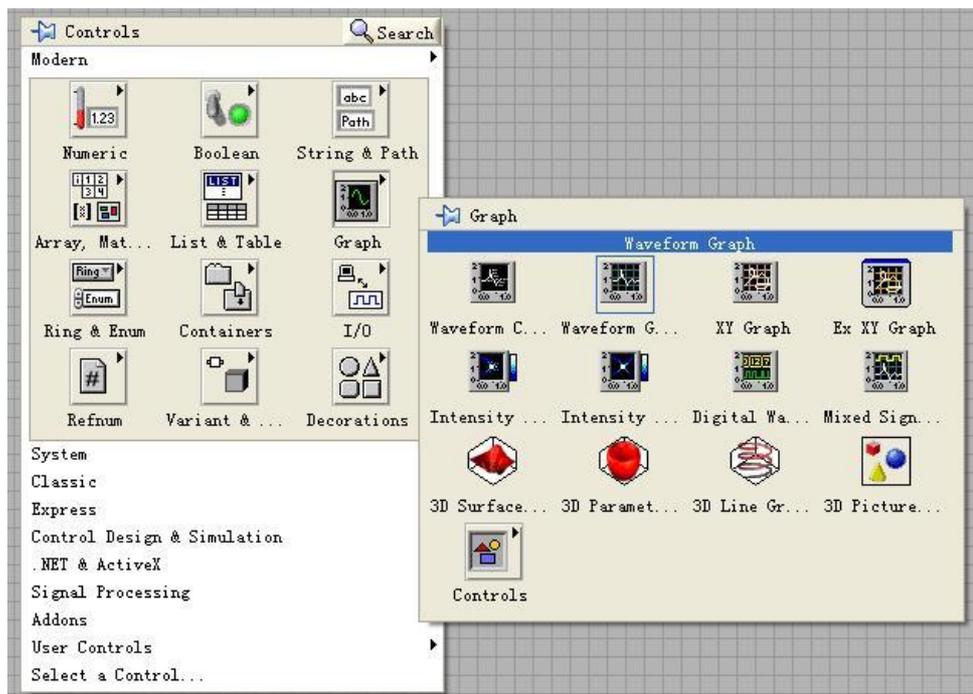
6. open **Block Diagram**; right-click and choose **Programming → String → String/Array/Path** and select "String To Byte Array"; then, use this function to connect **"read buffer"** on **"VISA Read"** function with the **Waveform Graph**.

See the figure below.
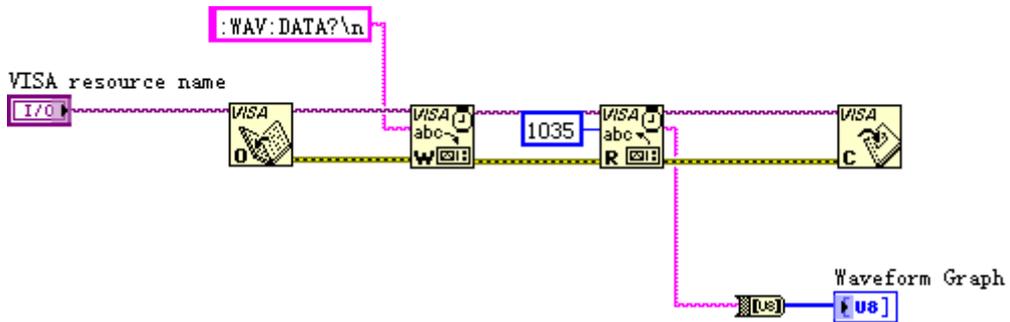
7. Add an **Event Structure** and a **While Loop** as well as two buttons, one of the buttons is used to control the start of waveform fetching, and the other one is to stop capturing. See the figure below.
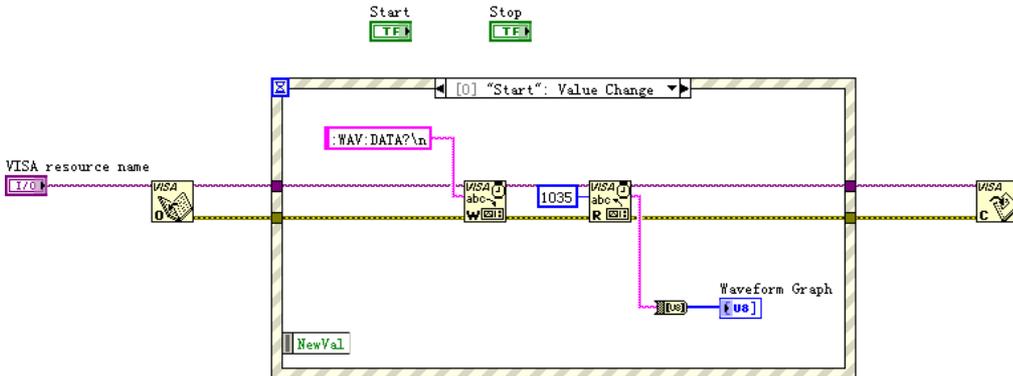
8. Right-click the **"selector label"** and choose **"Edit Events Handled by This Case"** or **"Add Event case"** to add events respectively for each button. Press **"Start"** to capture waveform and **"Stop"** to exit the program.
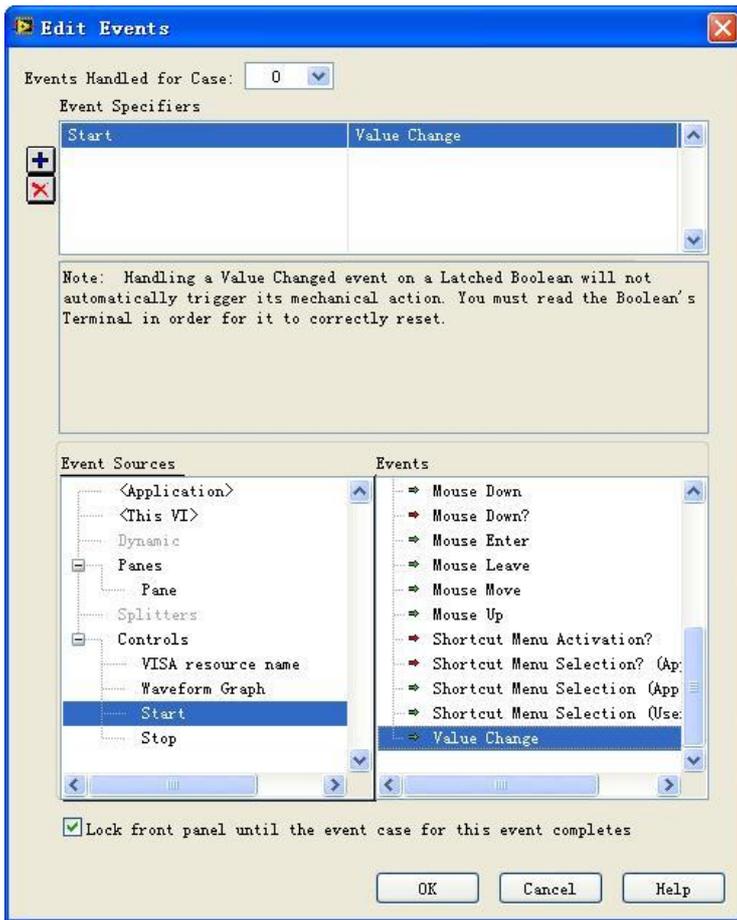
After you set the **"Start"** event, see the result below.
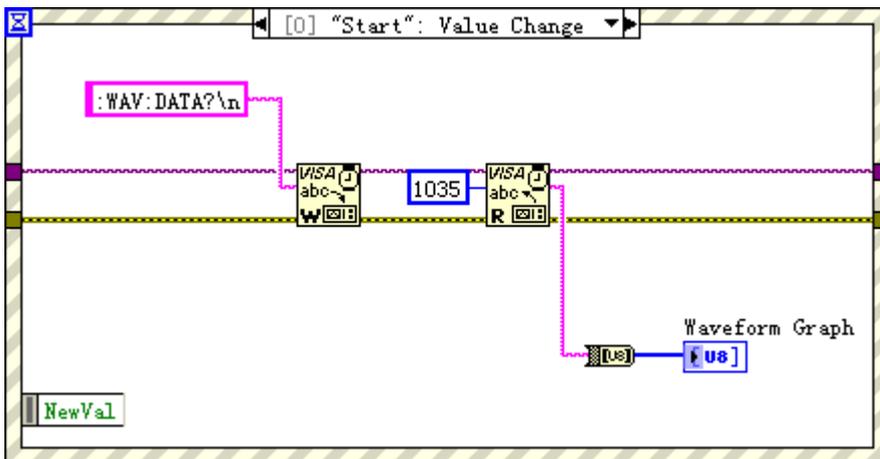
9. Add a **While Loop**; add **"Boolean"**→**"True Constant"** to point the event of the **"Stop"** button to **While** and exit.
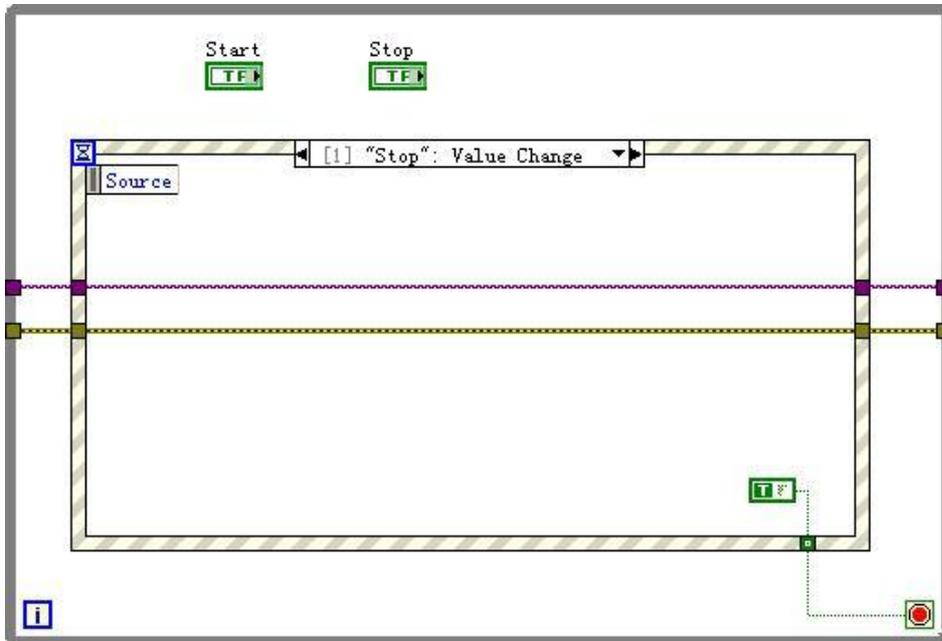
10. Change the input tunnel of VISA resource name and errors into **"Shift Register"** to finish creating program.
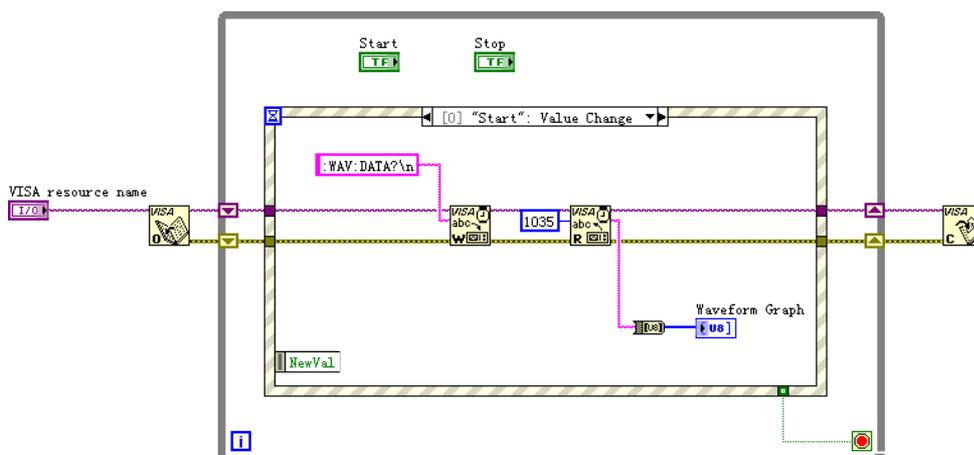
11. Adjust the style of **Front Panel** and click **"Fetch Waveform"** to get following interface. (the oscilloscope has been properly connected)
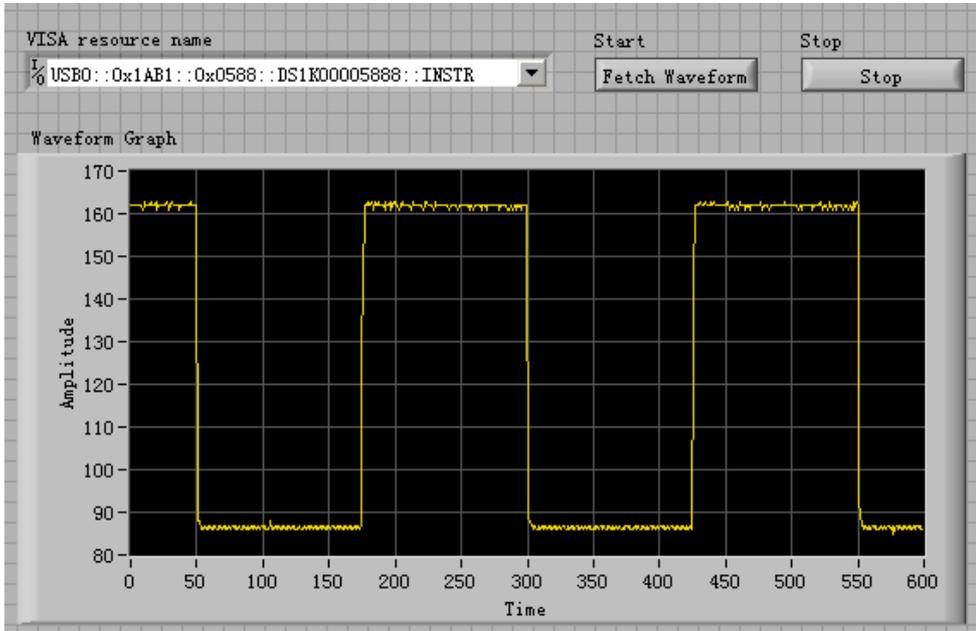
Command Quick Reference A-Z

W