

# PicoScope 9000 Programmer manual

---

Document date: 2010, October 26

### *Table of contents*

|   |    |
|---|----|
| PicoScope 9000 Programmer manual.....               | 1  |
| PicoScope 9000 API Reference .....                  | 4  |
| PicoScope9000 COM Server .....                      | 4  |
| COMRC Object .....                                  | 4  |
| ExecCommand Method .....                            | 4  |
| Commands syntax .....                               | 5  |
| Command and Query Structure .....                   | 5  |
| Overview .....                                      | 5  |
| Messages .....                                      | 5  |
| Commands .....                                      | 5  |
| Queries .....                                       | 6  |
| Headers .....                                       | 6  |
| Command Entry .....                                 | 6  |
| Rules .....   | 6  |
| Concatenating .....                                 | 7  |
| Command classification (Types of the commands)..... | 8  |
| Executing type commands .....                       | 8  |
| On/Off type commands.....                           | 8  |
| On/Off Group type commands.....                     | 9  |
| Selector type commands .....                        | 10 |
| Integer type commands .....                         | 10 |
| Float type commands .....                           | 10 |
| Data type commands.....                             | 11 |
| Full list of Command .....                          | 12 |
| Header On/Off command.....                          | 12 |
| GUI command.....                                    | 12 |
| System Commands: .....                              | 12 |
| Channels commands .....                             | 13 |
| Time base Commands.....                             | 14 |
| Trigger Commands .....                              | 16 |
| Acquisition commands.....                           | 17 |
| Display commands.....                               | 19 |
| Save/Recall commands .....                          | 21 |

### 3

|   |    |
|---|----|
| Work with Memo Zones (M1, M2, M3, M4) ..... | 21 |
| Work with Disk.....                         | 22 |
| Work with Setups.....                       | 23 |
| Markers commands .....                      | 24 |
| Measures commands .....                     | 25 |
| Measures of the Time Domain Signals .....   | 25 |
| Measures of the Spectrum Signals .....      | 30 |
| Getting Measures Results .....              | 31 |
| Limit Tests commands .....                  | 32 |
| Mathematics commands.....                   | 36 |
| FFT commands .....                          | 39 |
| Histogram commands .....                    | 39 |
| Setting Histogram Parameters.....           | 39 |
| Getting Results of Histogramm .....         | 43 |
| Mask Test commands .....                    | 44 |
| Common Mask Commands.....                   | 44 |
| Standard Mask Commands .....                | 46 |
| Getting Mask Results .....                  | 47 |
| Eye Diagramm commands.....                  | 49 |
| Setting Eye Parameters .....                | 49 |
| Getting Eye Measures Results .....          | 51 |
| Utilities commands .....                    | 52 |
| Waveforms commands.....                     | 53 |
| Programming Examples.....                   | 56 |
| Delphi.....                                 | 56 |
| LabView.....                                | 56 |
| Visual Basic .NET .....                     | 57 |
| Last revisions .....                        | 57 |

## PicoScope 9000 API Reference

PicoScope 9000 provides API for any third party application or library to control the oscilloscope and get signals. The API is *COM-based* and is provided by PicoScope 9000 GUI application.

### PicoScope9000 COM Server

COM server implementing the API is called *PicoScope9000* and is physically PicoScope 9000 GUI application (PicoScope9000.exe). It is registered in the system during the setup process, and can be explicitly unregistered and registered again by executing PicoScope9000.exe with /UnregServer or /RegServer switches.

### COMRC Object

The server exposes only one object implementing the API, which is called *COMRC*. The object supports automation, so it can be used by high-level languages like JavaScript (HTML pages) or VBA (Microsoft Word). However, low-level languages like C are also supported. The string defining system-wide name of the object and used for object creation is "*PicoScope9000.COMRC*".

### ExecCommand Method

COMRC object contains only one method *ExecCommand*. The method has one argument—a text string with a command or query. The method returns:

- *NULL* (*Nothing* in Visual Basic) if a command without query has been successfully executed;
- Text string "*ERROR*" if the command was invalid;
- Other text string with query results if the command was a query or a command with query.

The syntax of the commands and query, as well as the full list of commands is described in these documents below.

## Commands syntax

### *Command and Query Structure*

#### Overview

PicoScope 9000 commands consist of set commands and query commands (usually called commands and queries). Commands modify instrument settings or tell the instrument to perform a specific action. Queries cause the instrument to return data and information about its status.

Most commands have both a set form and a query form. The query form of the command differs from the set form by its question mark on the end. For example, the set command `ACQuire:Ch1:MODE` has a query form `ACQuire:Ch1:MODE?`.

Not all commands have both a set and a query form. Some commands have set only and some have query only.

#### Messages

A command message is a command or query name followed by any information the instrument needs to execute the command or query. Command messages may contain five element types, defined in the following table.

| <i>Command message elements</i> |   |
|---------------------------------|---|
| Symbol                          | Meaning   |
| <Header>                        | This is the basic command name. If the header ends with a question mark, the command is query. The header may begin with a colon (:) character. If the command is concatenated with other commands, the beginning colon is required.              |
| <Mnemonic>                      | This is a header of the sub-function. Some command headers have only one mnemonic. If a command header has multiple mnemonics, a colon (:) character always separates item from each other.   |
| <Argument>                      | This is a quantity, quality, restriction or limit associated with the header. Some commands have no arguments while others have multiple arguments. A <space> separates arguments from the header. A <comma> separates arguments from each other. |
| <Comma>                         | A single comma is used between arguments of multiple-argument commands. Optionally, there may be white space characters before and after the comma.   |
| <Space>                         | A white space character is used between a command header and its argument. Optionally, a white space may consist of multiple white space characters.  |

#### Commands

Commands cause the instrument to perform a specific function or change one of its settings. Commands have the structure:

[ : ] <Header> [ <Space> <Argument> [ <Comma> <Argument> ] . . . ]

A command header consists of one or more mnemonics arranged in a hierarchical or tree structure. The first mnemonic is the base or root of the tree and each subsequent mnemonic is a level or branch off the previous one. Commands at a higher level in the tree may affect those at a lower level. The leading colon (:) always returns you to the base of the command tree.

## Queries

Queries cause the instrument to return information about its status or settings. Queries have the structure:

- [ : ] <Header>?
- [ : ] <Header>? [ <Space> <Argument> [ <Comma> <Argument> ] . . . ]

You can specify a query command at any level within the command tree unless otherwise noted. These branch queries return information about all the mnemonics below the specified branch or level. For example, HISTogram:STATistics:STDdev? returns the standard deviation of the histogram, while HISTogram:STATistics? returns all the histogram statistics, and HISTogram? returns all the histogram parameters.

## Headers

You can control whether the instrument returns headers as part of the query response. Use the HEADER command to control this feature. If header is on, the query response returns command headers and formats itself as a valid set command. When the header is off, the response includes only the values. This may make it easier to parse and extract the information from the response. The table below shows the difference in responses.

| <i>Comparison of Header Off and Header On Responses</i> |            |                      |
|---|------------|----------------------|
| Query   | Header Off | Header On            |
| Ch1:Scale?  | 200 mV/div | CH1:SCALE 200 mV/div |
| Acq:Ch1:RecLen?   | 512        | ACQ:CH1:RECLen 512   |

## Command Entry

### Rules

The following rules apply when entering commands:

- You can follow a mnemonics by any letters for more easy understanding of the program's text. For example, the commands  
Ch1:ATTEN:DIMENS Volt,  
Ch1:ATTENuator:DIMENSION Volt and  
Ch1:ATTENblabla:DIMENSblabla Volt are equivalent. But arguments must be writing without any following letter.
- You can enter commands in upper or lower case.
- You can precede any command with white space characters. White space characters include any combination of the ASCII control characters 00 through 09 and 0B through 20 hexadecimal (0 through 9 and 11 through 32 decimal).
- The instrument ignores commands consisting of any combination of white space characters and line feeds.

## Concatenating

You can concatenate any combination of set commands and queries using a semicolon (;). The instrument executes concatenated commands in the order received. When concatenating commands and queries, you must follow these rules:

- Separate completely different headers by a semicolon and by the beginning colon on all commands except the first one. For example, the commands,  
TRIGger:MODE FREE and ACQuire:NUMAVg 10, can be concatenated into the following single command:  
TRIGger:MODE FREE;:ACQuire:NUMAVg 10
- If concatenated commands have headers that differ by only the last mnemonic, you can abbreviate the second command and eliminate the beginning colon. For example, you can concatenate the commands  
ACQuire:Ch1:MODE ENVMINMAX and  
ACQuire:Ch1:NAVG 10 into a single command:  
ACQuire:Ch1:MODE ENVMINMAX; NAVG 10 The longer version works equally well:  
ACQuire:CH1:MODE ENVMINMAX;:ACQuire:NAVG 10
- Set commands and queries may be concatenated in the same message. For example, ACQuire:CH1:MODE AVGSTAB;NAVG? is a valid message that sets the acquisition mode to Stable Averaging. The message then queries the number of acquisitions for averaging. Concatenated commands and queries are executed in the order received.

Here are some invalid concatenations:

- DISPlay:STYLE DOTS;ACQuire:NAVG 10  
(no colon before ACQuire)

- `DISPlay:STYLE DOTS;:FORMAT YT`  
(extra colon before `FORMAT`; use `DISPlay:STYLE DOTS;FORMAT YT` instead)
- `Acq:Ch1:Mode Sample;Ch1:RecLen 1024`  
(levels of the mnemonics are different; either remove the second use of `Ch1:` or place `:Acq:` in front of `Ch1:`)

## Command classification (Types of the commands)

Most of commands can be relate to one of few types. For example, the executing type commands tell the instrument to perform a specific action; the selector type commands modify a specific instrument setting to the one of few fixed values, and so on. All commands in any of these types have equal behavior.

### Executing type commands

The executing type commands tell the instrument to perform a specific action, for example

```
*Run
*ClrDispl
```

There are no arguments in these commands.

The all executing type commands has a set form only, and has not a query form.

### On/Off type commands

The On/Off type commands tell the instrument turn on or turn off specific function, for example

```
Header Off
Ch1:Display 0
```

There are four fixed arguments possible in these commands: `On`, `Off`, `0`, `1`. Arguments `On` and `1` is synonym and turn on the corresponding function. Arguments `Off` and `0` is synonym too and turn off the specific function.

The all On/Off type commands has a query form. The queries return one of two fixed values: `ON` or `OFF`. Also possible use query form with argument, for example

```
Ch1:Display? 0
```

These command turn off the graphic of Channel 1 and return `OFF`.



## On/Off Group type commands

Some functions of the instruments has a set of items, at that each item may be On or Off independent. Situation with all items is on or all items is off is possible also. A good example of this type of commands is the command `Meas:Ch1:XParam`. This command is set of parameters for automatic X-axis measurements for Ch1. Possible select up to 10 parameters from list of 18 items (parameters): `Period`, `Freq`, `PosWidth`, `NegWidth`, `Rise`, `Fall`, `PosDuty`, `NegDuty`, `PosCross`, `'NegCross`, `BurstWidth`, `Cycles`, `TimeOfMax`, `TimeOfMin`, `PosJitterPp`, `'PosJitterRMS`, `NegJitterPp`, `NegJitterRMS`

There are 2...64 custom items in the On/Off Group type commands. The full set of the items is specified for each commands in the list of commands (see below).

The On/Off Group type commands have a several kinds. Every command of these types can be used in every kind.

First kind of the On/Off Group commands is used for control to the one command's item and not change other items. In this case the item's mnemonic is added to the end of command via colon (:) character. Then must be follow a space character and one of the next arguments: **on**, **off**, **0**, **1**. For example, next command may be used for turning on a frequency measurement for Channel 1:

```
Meas:Ch1:XParam:Freq 1
```

First kind has a query form similar On-Off commands. So, the query

```
Meas:Ch1:XParam:Period 1
```

or

```
Meas:Ch1:XParam:Freq?
```

return the ON (**or** OFF) .

The second kind of the On/Off Group type commands use for simultaneous turning on the custom group of items. In this case the `:Include` mnemonic is added to the end of command Then must be follow a space character and a few items separated by comma (,). For example, next command turning on a rise time and fall time measurements for channel 1:

```
Meas:Ch1:XParam:Include Rise,Fall
```

The third kind of the On/Off Group type commands use for simultaneous turning off the custom group of items. In this case the `:Exclude` mnemonic is added to the end of command Then must be follow a space character and a few items separated by comma. For example, next command turning off a frequency and period measurements for channel 1:

```
Meas:Ch1:XParam:Exclude Freq,Period
```

The fourth kind of the On/Off Group type commands use for simultaneous turning off the all items. In this case the `:ClearAll` mnemonic is added to the end of command. For example, next command turning off all measurements for channel 1:

```
Meas:Ch1:XParam:ClearAll
```

The second, third and fourth kind of the On/Off Group type commands has not a query form.

The fifth kind of the On/Off Group type commands use for query about all turning on items now. This kind of commands has query form only. For example

```
Meas:Ch1:XParam?
```

The answer may be `ClearAll` – if all items is turned off; or one or few items separated by comma (,). For example, answer `Freq,Period` means two turned on items.

## Selector type commands

The selector type commands modify a specific instrument setting to the one of few fixed values, for example:

- `Trig:Source` command has a available arguments: `Direct`, `ExtHF`, `IntClock`;
- `Trig:Mode` command has a available arguments: `Free`, `Trig`.

There are 2...32 custom arguments possible in these commands. The full set of the arguments is specified for each commands in the list of commands (see below).

The selector type commands has a query form. Also possible use query form with argument, for example

```
Trig:Source? Direct
```

This command set the `Direct` input as trigger's source and return `DIRECT`.

## Integer type commands

The integer type commands modify a specific integer value functions. For example, the command `Acq:Ch1:RecLen 1024` set the length of Channel 1 signals to 1024 points. Valid diapason and step of value is different and describe in the list of commands (see below).

The integer type commands has a query form. Also possible use query form with argument, for example

```
Acq:Ch1:RecLen? 24
```

return 32, since 32 is a minimum valid length of signals.

## Float type commands

The float type commands modify a specific real value functions. For example, the command `Ch1: Scale 0.1` set the Y-scale for Channel 1 to 100 mV/div. Valid diapason and step of value is different and describe in the list of commands (see below).

The float type commands has a query form. Also possible use query form with argument, for example

```
Ch1:Scale? 0.1
```

return 100 mV/div, when V/div is dimension of the scale, and the prefix m is milli.

The commands

```
TB:ScaleA? 0.0000001,
```

```
TB:ScaleA? 100e-9,
```

```
TB:ScaleA? 0.1u,
```

```
TB:ScaleA? 100p
```

are equal and set the Scale A of timebase to value 100 ns/div. And all of these commands

return 100 ns/div.

### Data type commands

The Data type commands are used for the sending a some data to the instrument or for receiving a some data from the instrument, such as a acquired signal's array of points, a result of measurements, and so on.

Some Data type commands has a query form only, other Data type commands has a both commands and query forms. A structure of data is different for any command and specified for each command in the list of commands (see below).

## Full list of Command

### ***Header On/Off command***

Header: Header

Type – On/Off command

Action: Enable/Disable headers as part of the query response.

### ***GUI command***

Header: Gui

Type – Selector type command

Arguments: RemoteLocal, RemoteOnly, Invisible

Action: set the behavior of the GUI when it controls by COM-object.

Version: This command can be used with PicoScope SW v.2.3.2 or later.

## **System Commands:**

### ***Clear Display***

Header: \*ClrDispl

Type: Executing command

Action: Clear Display immediately.

### ***Start Cycle Acquisition***

Header: \*Run

Type – Executing command

Action: Run the instrument.

### ***Start Single Acquisition / Stop Acquisition***

Header: \*StopSingle

Type – Selector type command

Arguments: Stop, Single

Action: Single - Start the single acquisition; Stop – Immediately stop the acquisition.

Response: Stop – the instrument in stop now; Single – the instrument in the acquisition stage.

### ***Start the Autoscaling***

Header: \*Autoscale

Type – Selector type command

Arguments: Auto, SingleVal, NRZ, RZ

Action: set the type of signal and start the autoscaling of instruments.

Response: selected type of signals.

### ***Recall Default Setup***

Header: \*DefSetup

Type – Executing command

Action: Set the Default Setup of the instrument.

## **Channels commands**

### ***Display of the Channel***

Header: Ch1:Display; Ch2:Display

Type – On/Off command

Action: turn on or turn off the display of corresponding channel's signal.

### ***Acquisition of the Channel***

Header: Ch1:Acquire; Ch2:Acquire

Type – On/Off command

Action: turn on or turn off the acquisition of the channel's signal when it's display is turned off.

### ***Scale of the Channel***

Header: Ch1:Scale; Ch2:Scale

Type – Float type command

Argument: 0.002...0.5, or another when attenuator is used

Action: set the specified display scale in V/div.

### ***Offset of Channel***

Header: Ch1:Offset; Ch2:Offset

Type – Float type command

Argument: -1...+1, or another when attenuator is used

Action: set the specified compensation voltage in the channel in V.

### ***Bandwidth of Channel***

Header: Ch1:Band; Ch2:Band

Type – Selector type command

Arguments: Full, Narrow

Action: set the bandwidth of the channel.

***Attenuator's Unit***

Header: Ch1:Atten:Unit; Ch2:Atten:Unit

Type – Selector type command

Arguments: Off, Ratio, DB

Action: set the presence and the unit of attenuator or converter, used with the channel.

***Attenuation of Attenuator of Channel (ratio)***

Header: Ch1:Atten:Ratio; Ch2:Atten:Ratio

Type – Float type command

Argument: 0.0001...1000000

Action: set the attenuation ratio. This setting is active only when attenuator units is ratio

***Attenuation of Attenuator of Channel (dB)***

Header: Ch1:Atten:DB; Ch2:Atten:DB

Type – Float type command

Argument: -80...+120

Action: set the attenuation in dB. This setting is active only when attenuator units is decibels

***Units of Attenuator of Channel***

Header: Ch1:Atten:Dimens; Ch2:Atten:Dimens

Type – Selector type command

Arguments: Volt, Watt, Ampere, Unknown

Action: set the units of the converter, used with the channel.

***Time base Commands******Units of time base***

Header: TB:Units

Type – Selector type command

Arguments: Time, Bit

Action: set the units of the time base to s/div or bit/div.

***Mode of time base***

Header: TB:Mode

Type – Selector type command

Arguments: A, AB, B

Action: set the main, intensified, delayed time base.

***Scale of the main time base, sec/div***

Header: TB:ScaleA

Type – Float type command

Argument: 10e-12...50e-3

Action: set the scale of main time base when time units is used.

***Scale of the delayed time base, sec/div***

Header: TB:ScaleB

Type – Float type command

Argument: 10e-12...50e-3

Action: set the scale of delayed time base when time units is used.

***Scale of the main time base, bit/div***

Header: TB:BitScaleA

Type – Float type command

Argument: depended by actual bit rate

Action: set the scale of main time base when bit units is used.

***Scale of the delayed time base, bit/div***

Header: TB:BitScaleB

Type – Float type command

Argument: depended by actual bit rate

Action: set the scale of delayed time base when bit units is used.

***Delay of time base***

Header: TB:Delay

Type – Float type command

Argument: 0...10

Action: set the delay of intensified, delayed time base in divisions.

***Dual delayed time base***

Header: TB:DualDel

Type – On/Off command

Action: turn on or turn off the dual delayed time base (used in intensified or delayed time base).

***Delta delay of time base***

Header: TB:DeltaDel

Type – Float type command

Argument: 0...10

Action: set the delta delay of intensified, delayed time base in divisions (used in Dual delayed time base).

***Trigger Commands******Source of Trigger***

Header: Trig:Source

Type – Selector type command

Arguments: Direct, ExtHF, IntClock, ClockRecov

Action: set the source of the trigger.

***Trigger Level for Direct Input***

Header: Trig:ExtLevel

Type – Float type command

Argument: -1...+1

Action: set the trigger level for direct input, volt.

***Triggers Period for Internal Clock Sources***

Header: Trig:IntRate

Type – Float type command

Argument: 16e-9...0.002,

Action: set the period for internal clock trigger source, s.

***Mode of Trigger***

Header: Trig:Mode

Type – Selector type command

Arguments: Free, Trig

Action: set the Freerun or Triggered mode of the trigger.

***Trigger slope for Direct Trigger***

Header: Trig:Slope

Type – Selector type command

Arguments: Pos, Neg

Action: set the Positive or Negative slope of the trigger.



***Holdoff Time***

Header: Trig:Holdoff

Type – Float type command

Argument: 5e-6...1

Action: set the holdoff time, s.

***Hysteresis for Direct Trigger***

Header: Trig:Hister

Type – Selector type command

Arguments: Norm, HighSens

Action: set on the hysteresis for direct trigger (Norm) or set off (HighSens) .

***Attenuator's Unit for Direct Input***

Header: Trig:Atten:Unit

Type – Selector type command

Arguments: Off, Ratio, DB

Action: set the presence and the unit of attenuator or converter, used with the direct trigger input.

***Attenuation of Attenuator of Direct Input (ratio)***

Header: Trig:Atten:Ratio

Type – Float type command

Argument: 0.0001...1000000

Action: set the attenuation ratio. This setting is active only when attenuator units is ratio

***Attenuation of Attenuator of Direct Input (dB)***

Header: Trig:Atten:DB

Type – Float type command

Argument: -80...+120

Action: set the attenuation in dB. This setting is active only when attenuator units is decibels

***Acquisition commands******Type of signal***

Header: Acq:FitTo

Type – Selector type command

Arguments: `Multi`, `Single`

Action: prepare the instruments for best acquisition of single-valued or multi-valued

### *Sampling Mode*

Header: `Acq:Sample`

Type – Selector type command

Arguments: `Simult`, `Altern`

Action: `Simult` – set the simultaneous acquisitions of channels 1 and 2;

`Alternate` – set the alternate acquisitions of channels 1 and 2.

### *Acquisition Mode of Channel*

Header: `Acq:Ch1:Mode`; `Acq:Ch2:Mode`

Type – Selector type command

Arguments: `Sample`, `AvgStab`, `AvgMult`, `EnvMinMax`,  
`EnvMax`, `EnvMin`

Action: set the mode of acquisitions of specified channel.

### *Averages of Channel*

Header: `Acq:Ch1:NAvg`, `Acq:Ch2:NAvg`

Type – Integer type command

Argument: 1, 2, 4, 8, 16, ... , 4096

Action: set the averaging coefficient for specified channel.

### *Envelopes of Channel*

Header: `Acq:Ch1:NEnv`, `Acq:Ch2:NEnv`

Type – Integer type command

Argument: 1, 2, 4, 8, 16, ... , 4096, 8192

Action: set the number of signals for the envelope mode for specified channel.

Argument 8192 is used for unlimited number of signals.

### *Record Length of Channel*

Header: `Acq:Ch1:RecLen`, `Acq:Ch2:RecLen`

Type – Integer type command

Argument: 32, 64, 128, ... , 4096

Action: set the number of points for specified channel.

### *Termination of the Acquisition*

Header: `Acq:RunUntil`

Type – Selector type command

Arguments: StopBtn; NAcq

Action: set the condition of the acquisition's termination – when the Stop Button pressed or after specified number of waveforms is achieved.

### ***Number of Waveforms***

Header: Acq:NAcq

Type – Integer type command

Argument: 1...65535

Action: set the number of signals for the termination of the acquisition.

### ***Reaction when Number of Waveforms achieved***

Header: Acq:React

Type – On/Off Group type command

Items: Beep, Save

Action: if item Save is turned on the every signals is stored to the disk; if item Beep is turned on the beep signal will be sound after the specified number of waveforms achieved.

### ***Display commands***

Parameter <src> in Display Commands signifies *Source*  
( <src> is: Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3, F4, M1, M2, M3, M4, S1, S2 )

### ***Trace mode***

Header: Displ:TraceMode

Type – Selector type command

Arguments: AllLocked, PerTrace

Action: in PerTrace mode every waveforms may be displayed in its style; in AllLocked mode the display style of all waveforms is set as a style of active trace.

### ***Select the active trace***

Header: Displ:TraceSel

Type – Selector type command

Arguments: Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3, F4, M1, M2, M3, M4, S1, S2, XY

Action: select the active trace for AllLocked trace mode

### ***Set Display Style***

Header: Displ:<scr>:Style

Type – Selector type command

Arguments: Dots, Vectors, VarPersist, InfinPers, VarGrayScal, InfGrayScal, VColorGrad, IColorGrad

Action: set display's style for specified trace in PerTrace mode; set display's style for all trace in AllLocked mode if <scr> equal the active trace or nothing do if <scr> not equal the active trace.

### *Time of Persistence, s (for VarPersist Style)*

Header: Displ:<scr>:PersistTime

Type – Float type command

Argument: 0.1...20

Action: set the persistence time for specified trace in PerTrace mode; set persistence time for all trace in AllLocked mode if <scr> equal the active trace or nothing do if <scr> not equal the active trace.

### *Refresh Time, s (for VarGrayScal or VColorGrade Styles)*

Header: Displ:<scr>:RefreshTime

Type – Float type command

Argument: 1...200

Action: set the refresh time for specified trace in PerTrace mode; set refresh time for all trace in AllLocked mode if <scr> equal the active trace or nothing do if <scr> not equal the active trace.

### *Reset of the Display Style*

Header: Displ:ResetAll

Type: Executing command

Action: Reset the Display Styles for initial state (variable persistence 2 c).

### *Display Format*

Header: Displ:Format

Type – Selector type command

Arguments: YT, 2YT, 4YT, XY, CombYTXY, Comb2YTXY

Action: select the number and kinds of screens.

### *Define the Screen of Trace (for Format 4YT)*

Header: Displ:Screen4:<trace>,  
when <trace> is Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3,  
F4, M1, M2, M3, M4, S1, S2, Hist

Type – Selector type command

Arguments: 1, 2, 3, 4

Action: move the specified trace onto the specified screen in 4YT format.

### ***Define the Screen of Trace (for Formats 2YT, Comb2YTXY)***

Header: Displ:Screen2:<trace>,  
when <trace> is Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3,  
F4, M1, M2, M3, M4, S1, S2, Hist

Type – Selector type command

Arguments: 1, 2

Action: move the specified trace onto the specified screen in 2YT or Comb2YTXY formats.

### ***Source of the Axis X for XY-Screen***

Header: Displ:XAxis

Type – Selector type command

Arguments: Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3, F4, M1,  
M2, M3, M4, S1, S2

Action: set the specified signal as a X axis for XY screen.

### ***Source of the Axis Y for XY-Screen***

Header: Displ:YAxis

Type – Selector type command

Arguments: Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3, F4, M1,  
M2, M3, M4, S1, S2

Action: set the specified signal as a Y axis for XY screen.

### ***Type of Graticule***

Header: Displ:Gratic

Type – Selector type command

Arguments: Grid, Frame, Axis, Off

Action: define the type of graticules for the YT and XY screens.

## ***Save/Recall commands***

### ***Work with Memo Zones (M1, M2, M3, M4)***

#### ***Memory Display***

Header: Save:Memo:On

Type – On/Off Group type command

Items: M1, M2, M3, M4

Action: control of the displaying of the memory zones.

***Source for the storing into the Memory***

Header: Save:Memo:Source

Type – Selector type command

Arguments: Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3, F4, M1, M2, M3, M4, S1, S2

Action: define the signal as source for the store into the memory zone.

***Select the Memory for Saving***

Header: Save:Memo:ToMemo

Type – Selector type command

Arguments: M1, M2, M3, M4

Action: define the memory zone to the storing.

***Execute Saving into the Memory***

Header: Save:Memo:Save

Type – Executing command

Action: store the selected source into the selected memory.

**Work with Disk*****Source for storing into the file***

Header: Save:Memo:Source

Type – Selector type command

Arguments: Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3, F4, M1, M2, M3, M4, S1, S2

Action: define the signal as source for the store into the disk's file.

***Name of the File***

Header: Save:Disk:FileName

Type – Data type commands

Argument: text string

Forms: command, query, command with query.

Action: define the file name for storing the specified signal onto the Disk.

***Mode of File Name***

Header: Save:Disk:NameMode

Type – Selector type command

Arguments: Manual, Auto

Action: set the mode of file name. In Auto mode the file name consists of a base name, followed by a underline (\_) and fives-digits number. Each time you save a waveform, the number in the file name is automatically incremented, for example: `basename_00001.wfm`, `basename_00002.wfm`, `basename_00003.wfm` and so on.

### *Format of the stored files*

Header: `Save:Disk:FileFormat`

Type – Selector type command

Arguments: `Binary`, `Verbose`, `YOnly`

Action: set the format of the file.

### *Execute Saving onto the Disk*

Header: `Save:Disk:Save`

Type – Executing command

Action: store the selected source into the specified before Disk File.

### *Select Memory for loading a signal from Disk*

Header: `Save:Disk:ToMemo`

Type – Selector type command

Arguments: `M1`, `M2`, `M3`, `M4`

Action: selects which of the available memory locations the instrument loads the saved file into.

### *Execute Loading*

Header: `Save:Disk:Load`

Type – Executing command

Action: read the specified before Disk File and put the read signal into the specified before Memory Zone.

## **Work with Setups**

### *Recall Factory Setup*

Header: `Save:Setup:RecFact`

Type – Executing command

Action: returns the instrument to the manufacturer's default setting.

### *Recall Power Off Setup*

Header: `Save:Setup:RecLast`

Type – Executing command

Action: returns the instrument to the last setting before the power supply was last switched off.

### *Save Setup as Default*

Header: `Save:Setup:SvAsDefault`

Type – Executing command

Action: stores the present front-panel setup as the default setup.

### *Name of the custom Setup File*

Header: `Save:Setup:FileName`

Type – Data type commands

Argument: text string

Forms: command, query, command with query.

Action: define the file name for storing the custom Setup.

### *Save custom Setup*

Header: `Save:Setup:Save`

Type – Executing command

Action: stores the present front-panel setup as the custom setup specified before.

### *Execute Recall custom Setup*

Header: `Save:Setup:Recall`

Type – Executing command

Action: recall a setup that you have previously saved on any drive of your PC.

The name of the Setup must be defined by command

`Save:Setup:FileName` preliminarily

## **Markers commands**

### *Type of the Markers*

Header: `Mark:Type`

Type – Selector type command

Arguments: `Off`, `MX`, `MY`, `XY`

Action: set the type of the markers.

### *Sources of the Markers*

Header: `Mark:M1:Source`, `Mark:M2:Source`

Type – Selector type command

Arguments: `Ch1`, `Ch1B2`, `Ch2`, `Ch2B2`, `F1`, `F2`, `F3`, `F4`, `M1`, `M2`, `M3`, `M4`, `S1`, `S2`



Action: attach the specified marker to the specified signal.

### *X position of Marker*

Header: Mark:M1:XPos , Mark:M2:XPos

Type – Float type command

Argument: real value of the X-axis

Action: set the X position of the specified marker.

### *Y position of the Marker*

Header: Mark:M1:YPos , Mark:M2:YPos

Type – Float type command

Argument: real value of the Y-axis

Action: set the Y position of the specified marker.

### *Motion of the Markers*

Header: Mark:Motion

Type – Selector type command

Arguments: Independ, Paired

Action: when the Paired motion is selected you can move both markers with the **M1 POSITION** variable simultaneously, while the difference between markers can be moved with the **M2 POSITION** variable.

## **Measures commands**

Mnemonic <src> in some Measures Commands signifies *Source*  
( <src> is **Ch1 | Ch1B2 | Ch2 | Ch2B2 | F1 | F2 | F3 | F4 | M1 | M2 | M3 | M4 | S1 | S2** )

## **Measures of the Time Domain Signals**

### *Type of Measures*

Header: Meas:Display

Type – Selector type command

Arguments: Off, Param, Statistic

Action: set the type of measures.

### *Sources for Measures*

Header: Meas:DisplSrc

Type – Selector type command

Arguments: Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3, F4, M1, M2, M3, M4, S1, S2

Action: set the current source for the Measures.

***Viewing of the Define Parameters***

Header: Meas:View

Type – On/Off command

Action: set visibility of the markers of the define parameters for the selected Sources.

***Mode of the Measurement***

Header: Meas:Mode

Type – Selector type command

Arguments: Permanent, Single

Action: set the mode of the Measures.

***Executing of single Measure***

Header: Meas:SingleMeas

Type – Executing command

Action: execute the single measurement in Single mode

***Mode of the Statistic Measurement***

Header: Meas:StatMode

Type – Selector type command

Arguments: Permanent, Window, Weight

Action: set the mode of the Statistic Measures.

***Windows Value***

Header: Meas:Window

Type – Integer type command

Argument: 8...8192

Action: set the number of the recently acquired waveforms for the Window mode of the Statistic Measurement.

***Weight Value***

Header: Meas:Weight

Type – Integer type command

Argument: 8...8192

Action: set the weight variable for the Weight mode of the Statistic Measurement

***Top/Base definition Method***

Header: Meas:<src>:Method

Type – Selector type command

Arguments: Hist, MinMax, Marker

Action: sets the Top and Base vertical reference thresholds for amplitude measurements of the specified signals.

### ***Top Value for Marker Method***

Header: Meas:<src>:Top

Type – Integer type command

Argument: 2...1023

Action: sets the Top vertical reference threshold for the specified signals. Argument 0 correspond to the bottom of the screen, and argument 1023 correspond to the top of the screen independent of the real screen's height.

### ***Base Value (for Marker Method)***

Header: Meas:<src>:Base

Type – Integer type command

Argument: 1...1022

Action: sets the Base vertical reference threshold for the specified signals. Argument 0 correspond to the bottom of the screen, and argument 1023 correspond to the top of the screen independent of the real screen's height.

### ***Threshold definition Method***

Header: Meas:<src>:Thresh

Type – Selector type command

Arguments: 10-90, 20-80, Custom

Action: sets the lower, middle, and upper thresholds for measurements of the specified signals. May set to the fixed values 10%-50%-90%; 20%-50%-80%; or custom values.

### ***Thresholds Units***

Header: Meas:<src>:Unit

Type – Selector type command

Arguments: Percent, Volt, Division

Action: sets the units of the thresholds for the specified signals. It used for custom threshold definition Method only.

### ***Position of the Upper, Middle or Lower Threshold***

Headers:

Meas:<src>:UpThresh

Meas:<src>:MidThresh

Meas:<src>:LowThresh

Type – Float type command

Arguments:

absolute voltage value ( for Volt threshold units only )

-4...+4 ( for Division threshold units only )

Action: sets the threshold position for the specified signals.

### *Percentage of the Upper, Middle or Lower Threshold*

Headers:

Meas:<src>:UpThPerc

Meas:<src>:MidThPerc

Meas:<src>:LowThPerc

Type – Integer type command

Arguments: -80...+200

Action: sets the threshold percentage for the specified signals. It used for Percent threshold units only. Argument 0 (%) correspond to the Base of the signals, and argument 100 (%) correspond to the Top of the signals.

### *Margins definition Mode*

Header: Meas:<src>:MargMode

Type – Selector type command

Arguments: Slope, Marker

Action: sets the mode of the margins definition.

### *Slope of the Left or Right Margins*

Headers:

Meas:<src>:LeftSlope

Meas:<src>:RightSlope

Type – Integer type command

Arguments: 0...127

Action: sets the margin for the specified signals onto the specified slope. It used for slope margins definition mode only. Argument 0 mean the first rise, value 1 is first fall, 2 – secont rise; 3 – second fall, and so on.

### *Thresholds of Left and Right Margins Slopes*

Headers:

Meas:<src>:LeftTresh

Meas:<src>:RightTresh

Type – Selector type command

Arguments: Upper, Middle, Lower

Action: sets the thresholds for definitions of the left or right slope. It used for slope margins definition mode only.

### ***Position of the Left or Right Margin***

Headers:

Meas:<src>:LeftMarker

Meas:<src>:RightMarker

Type – Float type command

Arguments: absolute time value.

Action: sets the position of margin for the specified signals. It used for marker margins definition mode only.

### ***List of X-Measurements***

Header: Meas:<src>:XParam

Type – On/Off Group type command

Items: Period, Freq, PosWidth, NegWidth, Rise, Fall, PosDuty, NegDuty, PosCross, NegCross, BurstWidth, Cycles, TimeOfMax, TimeOfMin, PosJitterPp, PosJitterRMS, NegJitterPp, NegJitterRMS

Action: define the set of the X-axis measures for the specified signals.

### ***List of Y-Measurements***

Header: Meas:<src>:YParam

Type – On/Off Group type command

Items: Max, Min, PP, Top, Base, Ampl, Middle, Mean, dcRMS, acRMS, Area, CycMean, CycDcRMS, CycAcRMS, CycArea, PosOver, NegOver

Action: define the set of the Y-axis measures for the specified signals.

### ***Second Source for Inter Signals Measurements***

Header: Meas:Source2

Type – Selector type command

Arguments: Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3, F4, M1, M2, M3, M4, S1, S2

Action: set the second source for the inter signals measures.

***List of X-Inter Signals Measurements***

Header: Meas:<src>:XDualPar

Type – On/Off Group type command

Items: Del1R1R, Del1R1F, Del1F1R, Del1F1F, Del1RnR, Del1RnF, Del1FnR, Del1FnF, PhaseDeg, PhaseRad, PhasePerc

Action: define the set of the X-axis inter signals measures for the specified signals.

***List of Y-Inter Signals Measurements***

Header: Meas:<src>:YDualPar

Type – On/Off Group type command

Items: Gain, DBGain

Action: define the set of the Y-axis inter signals measures for the specified signals.

***Delete all Measures for all Sources***

Header: Meas:ClearAll

Type – Executing command

Action: Clear list of all measures for all signals.

**Measures of the Spectrum Signals*****Limits Definition Method for Spectrum.***

Header: Meas:<src>:FFTMethod

Type – Selector type command

Arguments: Harmonic, Peak

Action: sets the method of the limits definition for the specified signal. It used for the spectrum signals only.

***Left and Right Spectrums Margin***

Headers:

Meas:<src>:FFTLeft

Meas:<src>:FFTRight

Type – Float type command

Arguments: absolute frequency value.

Action: sets the position of margin for the specified spectrum signals. It used for searching a peak 1 of the spectrum for Harmonic method.

***Peak Level of the Spectrum***

Header: Meas:<src>:PeakLevel

Type – Float type command

Arguments: -100...+80 (dBV)

Action: sets the level for the specified spectrum signals. It used for searching a peaks of the spectrum for Peak method.

***Left and Right Peaks of the Spectrum***

Headers:

Meas:<src>:PeakLeft

Meas:<src>:PeakRight

Type – Integer type command

Arguments: 1...41

Action: sets the first and second peaks for the specified spectrum signals.

***List of Frequency-Measurements for Spectrum***

Header: 1. Meas:<src>:XFFTPar

Type – On/Off Group type command

Items: Freq, DFreq

Action: define the set of the frequency measures for the specified signals.

***List of Magnitude-Measurements for Spectrum***

Header: 1. Meas:<src>:YFFTPar

Type – On/Off Group type command

Items: Magn, DMagn, TDH

Action: define the set of the magnitude measures for the specified signals.

**Getting Measures Results*****Get List of Measured Parameters***

Header: Meas:Res:List?

Type – Data type commands

Argument: none

Forms: query only.

Action: return text with the list of the active measures for all signals with ordinal index.

***Get Current Value of the Parameter***

Header: Meas:Res:<N>?

Parameter <N>: - index of the parameter in the list

Type – Data type commands

Argument: none

Forms: query only.

Action: return the last result of the specified measured parameter.

### ***Get Statistic Value of Parameter***

Header: Meas:Res:<N>:<Val>?

Parameter <N>: - index of the parameter in the list

Parameter <Val>: Wfm, Min, Max, Mean, StdDev

Type – Data type commands

Arguments: none

Forms: command with query only.

Action: return the specified statistic parameter of the measured parameter.

## ***Limit Tests commands***

### ***Limit Test On/Off***

Header: Limit:TestOn

Type – On/Off command

Action: Enable/Disable the Limit Test. Must be set On after full definitions of the all others Limit Test parameters.

### ***Condition of the Limit Test Terminations***

Header: Limit:RunUntil

Type – Selector type command

Arguments: StopBtn, Failur, Wfm

Action: set the condition of the Limit Test Termination.

### ***Number of Failures***

Header: Limit:Failures

Type – Integer type command

Argument: 1...10000

Action: set the number of failures for the Failur Condition of the Limit

### ***Number of Waveforms***

Header: Limit:NWfms

Type – Integer type command

Argument: 1...1000000



Action: set the number of waveforms for the Ffm Condition of the Limit

### *Action*

Header: Limit:Action

Type – On/Off Group type command

Items: Beep, Save, Stop

Action: if item Save is turned on the every signals with limit condition is stored to the disk; if item Beep is turned on the beep signal will be sound for the every limit condition; if the item Stop is turned on the acquisition immediately stop after the first limit condition.

### *Action If*

Header: Limit:If

Type – Selector type command

Arguments: AnyFail, AllPass, AllFail, AnyPass

Action: define the limit condition: AnyFail – condition spring up when one or some active measures is fail; AllPass – when all active measures is good; AllFail – when all active measures is fail; AnyPass – condition spring up when one or some active measures is good.

### *Format of the stored files*

Header: Limit:FileFormat

Type – Selector type command

Arguments: Binary, Verbose, YOnly

Action: set the format of the file.

### *Name of the File*

Header: Limit:FileName

Type – Data type commands

Argument: text string

Forms: command, query, command with query.

Action: define the file name for storing the specified signals onto the Disk.

### *Activity of Parameter*

Headers:

Limit1:Activ

Limit2:Activ

Limit3:Activ

Limit4:Activ

Type – On/Off command

Action: Enable/Disable the Limit Test for according parameter.

### *Mode of the Parameter's Limits*

Headers

Limit1:Mode

Limit2:Mode

Limit3:Mode

Limit4:Mode

Type – Selector type command

Arguments: Center, Limit

Action: set the mode of the limits for the according parameter.

### *Upper and Lower Limits of the Parameters*

Headers:

Limit1:UpLimit                      Limit1:LowLimit

Limit2:UpLimit                      Limit2:LowLimit

Limit3:UpLimit                      Limit3:LowLimit

Limit4:UpLimit                      Limit4:LowLimit

Type – Float type command

Arguments: absolute value of limit

Action: sets the limit's value. It used for Limit mode of the parameter's limit only.

### *Mode of the Center of the Parameter*

Headers

Limit1:CenterMode

Limit2:CenterMode

Limit3:CenterMode

Limit4:CenterMode

Type – Selector type command

Arguments: CurrMean, UserDef

Action: set the mode of the center's definition for the according parameter. It used for Center mode of the parameter's limit only.

### *Value of the Center*

Headers:

Limit1:CenterVal

```
Limit2:CenterVal
```

```
Limit3:CenterVal
```

```
Limit4:CenterVal
```

Type – Float type command

Arguments: absolute value of center

Action: set the absolute center's value. It used for UserDef mode of the center of the parameter.

### ***Mode of Delta of the Parameter***

Headers

```
Limit1:Delta
```

```
Limit2:Delta
```

```
Limit3:Delta
```

```
Limit4:Delta
```

Type – Selector type command

Arguments: StdDev, UserDef, UserPerc

Action: set the mode of the delta's definition for the according parameter. It used for Center mode of the parameter's limit only.

### ***Value of Delta of the Parameter for Standard Deviation mode***

Headers:

```
Limit1:StdDev
```

```
Limit2:StdDev
```

```
Limit3:StdDev
```

```
Limit4:StdDev
```

Type – Float type command

Arguments: 0.1...100 of standard deviations of the parameter

Action: sets the delta's value. It used for StdDev mode of delta of the parameter only.

### ***Value of Delta of the Parameter for User Defined mode***

Headers:

```
Limit1>UserDef
```

```
Limit2>UserDef
```

```
Limit3>UserDef
```

```
Limit4>UserDef
```

Type – Float type command

Arguments: absolute value of delta

Action: sets the delta's value. It used for UserDef mode of delta of the parameter only.

### ***Percentage of Delta of the Parameter for User Defined mode***

Headers:

```
Limit1:UserPerc
Limit2:UserPerc
Limit3:UserPerc
Limit4:UserPerc
```

Type – Float type command

Arguments: 0.01%...90% of standard deviations of the parameter

Action: sets the delta's value. It used for UserPerc mode of delta of the parameter only.

### ***Failure When***

Headers

```
Limit1:FailWhen
Limit2:FailWhen
Limit3:FailWhen
Limit4:FailWhen
```

Type – Selector type command

Arguments: Outside, Inside, Always

Action: set the mode of the quality control for the according parameter.

### ***If Measurement undefined***

Headers

```
Limit1:NotFound
Limit2:NotFound
Limit3:NotFound
Limit4:NotFound
```

Type – Selector type command

Arguments: Ignore, Fail, Pass

Action: set the limitation's status when measurement is undefined.

## **Mathematics commands**

### ***Enable of Mathematical Function***

Headers:

```
F1:Display
```

F2:Display

F3:Display

F4:Display

Type – On/Off command

Action: enable/disable the calculation and display of the according functions.

### *Operator of Function*

Headers:

F1:Operat

F2:Operat

F3:Operat

F4:Operat

Type – Selector type command

Arguments: Add, Sub, Mult, Div, Invert, Abs, Exp\_e, Exp\_10, Log\_e, Log\_10, Dif\_al, Int\_al, IFFT, LinInt, SinInt, Smooth, Trend

Action: set the operator of the specified function.

### *Operand 1*

Headers:

F1:Source1

F2:Source1

F3:Source1

F3:Source1

Type – Selector type command

Arguments: Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3, F4, M1, M2, M3, M4, S1, S2

Action: set the first operand for of the specified function.

### *Operand 2*

Headers:

F1:Source2

F2:Source2

F3:Source2

F4:Source2

Type – Selector type command

Arguments: Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3, F4, M1, M2, M3, M4, S1, S2, Const

Action: set the second operand for of the specified function. It is used for Add, Sub, Mult, Div operator only.

**F<n>**

### *Constant value*

Headers:

F1:Const

F2:Const

F3:Const

F4:Const

Type – Float type command

Arguments: absolute value of constant

Action: set the constant for of the specified function. It used for Const as second operand.

### *Parameter of Smoothing*

Headers:

F1:SmoothLen

F2:SmoothLen

F3:SmoothLen

F4:SmoothLen

Type – Integer type command

Argument: 3, 5, 7, 9, ... , 49, 51

Action: set the length of the smoothing's interval in points for specified function. It used for Smooth operator only.

### *Measurement of Trend*

Headers

F1:TrendMeas

F2:TrendMeas

F3:TrendMeas

F4:TrendMeas

Type – Selector type command

Arguments: Period, Freq, PosWidth, NegWidth, RiseTime, FallTime, PosDuty, NegDuty

Action: set the kind of trends for specified function. It used for Trend operator only.

## **FFT commands**

### ***Enable of Spectrums***

Headers:

Spectr1:Display

Spectr2:Display

Type – On/Off command

Action: enable/disable the calculation and display of the according spectrum.

### ***Source for Spectrum***

Headers:

Spectr1:Source1

Spectr2:Source1

Type – Selector type command

Arguments: Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3, F4, M1, M2, M3, M4

Action: set the source for of the specified spectrum.

### ***Window***

Headers

Spectr1:Window

Spectr2:Window

Spectr3:Window

Spectr4:Window

Type – Selector type command

Arguments: Rectang, Hamming, Hanning, Flattop, BlackHarr, KaiserBess

Action: set the window for specified spectrum.

## **Histogram commands**

### **Setting Histogram Parameters**

#### ***Axis of the Histogram***

Header: Hist:Axis

Type – Selector type command

Arguments: Off, Vert, Horiz

Action: set the axis of the histogram.

***Source of the Histogram***

Header: Hist:Source

Type – Selector type command

Arguments: Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3, F4, M1, M2, M3, M4, S1, S2

Action: select the specified signal as source of the histogram.

***Visibility of the Histogram***

Header: Hist:Visible

Type – On/Off command

Action: set visibility of the histogram. The acquisition of the histogram is going independently of this commands.

***Condition of Histogram's Finish***

Header: Hist:RunUntil

Type – Selector type command

Arguments: StopSingle, Wfms, Samples

Action: set the condition of the finish of the histogram's acquisition.

***Number of Waveforms for Histogram***

Header: Hist:NWfm

Type – Integer type command

Argument: 1...1000000

Action: set the number of signals for the termination of the histogram's acquisition.

***Number of Samples for Histogram***

Header: Hist:NSample

Type – Integer type command

Argument: 1...10000000

Action: set the number of samples for the termination of the histogram's acquisition.

***Mode of Limits for Histogram's Window***

Header: Hist:Limits

Type – Selector type command

Arguments: Paried, Independ

Action: set the mode of the limits of histogram's window.



***Unit of Limits for Histograms Window***

Header: Hist:Units

Type – Selector type command

Arguments: Absolute, Percent

Action: set the units of the limits of histogram's window.

***Left and Right Window's Limits for vertical or horizontal Histograms***

Headers:

Hist:WVert:Left

Hist:WVert:Right

Hist:WHor:Left

Hist:WHor:Right

Type – Float type command

Argument: real value of the X-axis (for Absolute units);

0%...100% of the X-axis (for Percent units);

Action: set the X positions of the histogram's window.

***Top and Bottom Window's Limits for vertical or horizontal Histograms***

Headers:

Hist:WVert:Top

Hist:WVert:Bottom

Hist:WHor:Top

Hist:WHor:Bottom

Type – Float type command

Argument: real value of the Y-axis (for Absolute units);

0%...100% of the Y-axis (for Percent units);

Action: set the Y positions of the histogram's window.

***Visibility of the Window***

Header: Hist:Display

Type – On/Off command

Action: set visibility of the window.

***Mode of the Calculation***

Header: Hist:Mode

Type – Selector type command

Arguments: Normal, Exponent

Action: set the mode of the histogram's calculation.

***Weight for the exponential Calculation***

Header: Hist:Weight

Type – Integer type command

Argument: 8, 16, 32, ..., 8192

Action: set the number of signals for the termination of the acquisition.

### *Type of the Scale*

Header: Hist:ScaleType

Type – Selector type command

Arguments: Linear, Logarith

Action: set the type of the histogram's scales.

### *Mode of the Scale*

Header: Hist:ScaleMode

Type – Selector type command

Arguments: Auto, Manual

Action: set the mode of the histogram's scales.

### *Linear Scale of the vertical or horizontal Histograms*

Headers:

Hist:VertScale

Hist:HorScale

Type – Float type command

Argument: (10...100) %/div

Action: set the scale of the histograms. It used for Manual mode and Linear type of the scale only.

### *Linear Offset of the vertical or horizontal Histograms*

Headers:

Hist:VertOffset

Hist:HorOffset

Type – Float type command

Argument: 0%...100%

Action: set the offset of the histograms. It used for Manual mode and Linear type of the scale only.

### *Logarithmic Scale of the vertical or horizontal Histograms*

Headers:

Hist:VertDBScale

Hist:HorDBScale

Type – Float type command

Argument: (6...60) dB/div

Action: set the scale of the histograms. It used for Manual mode and Logarith type of the scale only.

### *Logarithmic Offset of the vertical or horizontal Histograms*

Headers:

Hist:VertDBOffs

Hist:HorDBOffs

Type – Float type command

Argument: (-60...0) dB

Action: set the offset of the histograms. It used for Manual mode and Logarith type of the scale only.

## Getting Results of Histogramm

### *Get Histogram's Data*

Headers:

Hist:Data:Vert?

Hist:Data:Hor?

Type – Data type commands

Argument: none

Forms: query only

Action: return the text string with values of the histogram (through comma).

### *Get Histogram's Measure*

Headers:

Hist:Res:<Param>?

Parameter <Param>:

- InBox                      - Number of Hints in Box
- Wfm                        - Number of Waweforms
- Peak                       - Peak Value of Histogram
- PP  
  of Signal                - Difference between highest and lowest Values
- Median  
  Signal                  - Centre between highest and lowest Values of
- Mean                      - Average of Distribution of Histogram
- StdDev                    - Standard Deviation of Histogram
- Mean1S                  - number of hints in Mean  $\pm$  StdDev Region, %

- Mean2S - number of hints in Mean  $\pm$  2StdDev Region, %
- Mean3S - number of hints in Mean  $\pm$  3StdDev Region, %
- Min - Min. Value of Signal
- Max - Max. Value of Signal
- Max-Max - Difference between two Values of Signal,  
matched two Max of Histogramm

## Type – Data type commands

Argument: none

Forms: query only

Action: return the text string with value of the specified parameters.

## Mask Test commands

## Common Mask Commands

## Mask Erasing

Header: Mask : EraseMask

Type: Executing command

Action: Clear the current mask from the display.

### Signal for Mask Testing

Header: Mask:CompareWith

### Type – Selector type command

Arguments: Ch1, Ch2

Action: select the signal for mask testing.

### Actuation of Mask Testing

Header: Mask:Test

### Type – On/Off command

**Action:** enable/disable of the mask test execution.

### *Name of the User Masks File*

Header: Mask:MaskFile

## Type – Data type commands

Argument: text string

Forms: command, query, command with query.

Action: define the file name for next loading of the user mask from the disk.

***Load User Mask***

Header: Mask:LoadUser

Type – Executing command

Action: load the specified before user mask.

***Condition of the Mask Test's Finish***

Header: Mask:RunUntil

Type – Selector type command

Arguments: StopBtn, FailedWfms, FailedSmpls, Wfms, Samples

Action: set the condition of the Mask Test Termination.

***Number of Failed Waveforms***

Header: Mask:FailWfms

Type – Integer type command

Argument: 1...1000000

Action: set the number of the failed waveforms for the FailedWfms condition of the Finish.

***Number of Failed Samples***

Header: Mask:FailSmpls

Type – Integer type command

Argument: 1...1000000

Action: set the number of the failed samples for the FailedSmpls condition of the Finish.

***Number of Waveforms***

Header: Mask:NWfms

Type – Integer type command

Argument: 1...1000000

Action: set the number of the all waveforms for the Wfms condition of the Finish.

***Number of Samples***

Header: Mask:NSamples

Type – Integer type command

Argument: 1...1000000

Action: set the number of the all samples for the Samples condition of the Finish.

***Actions of Mask Test***

Header: Mask:Action

Type – On/Off Group type command

Items: Beep, Save

Action: if item Save is turned on the every failed signals is stored to the disk; if item Beep is turned on the beep signal will be sound for the every failed signals.

***Format of the stored Files***

Header: Mask:FileFormat

Type – Selector type command

Arguments: Binary, Verbose, YOnly

Action: set the format of the file. It is used when Save action is on.

***Name of the stored File***

Header: Mask:FileName

Type – Data type commands

Argument: text string

Forms: command, query, command with query.

Action: define the file name for storing the failed signals onto the Disk. It is used when Save action is on.

**Standard Mask Commands*****Alignment of the signal with Standard Mask***

Header: StdMask:Align

Type – On/Off command

Action: enable/disable the alignment of the tested signal with the standard mask's parameters.

***Enable Margins***

Header: StdMask:MarginsOn

Type – On/Off command

Action: enable/disable the margins control of the eye-typed masks.

***Margins Value***

Header: StdMask:MarginsVal

Type – Float type command

Arguments: -100%...+100%

Action: set the margins value. It used when margins is enable.

### *Get List of Standard*

Header: StdMask:StdsList?

Type – Data type commands

Argument: none

Forms: query only

Action: return the list of the mask standards with ordinal index.

### *Select of the Standards*

Header: StdMask:StdIndex

Type – Integer type command

Argument: 0...Number of standards - 1

Action: select the current standards by its ordinal index.

### *Get List of Masks*

Header: StdMask:MasksList?

Type – Data type commands

Argument: none

Forms: query only

Action: return the list of the masks with ordinal index from the selected standard.

### *Select of the Standard Mask*

Header: StdMask:MaskIndex

Type – Integer type command

Argument: 0...Number of masks in the current standard - 1

Action: load the specified mask by its ordinal index.

## **Getting Mask Results**

### *Get Integrated Results of Mask Test*

Headers:

Mask:Res:<Param>?

Parameter <Param>:

- AllWfm                      - number of the all waveforms
- FailWfm                    - number of the failed waveforms
- AllSmpl                    - number of the all samples
- FailSmpl                   - number of the failed samples

Type – Data type commands

Argument: none

Forms: query only

Action: return the text string with value of the specified parameter.

### *Get Number of Samples in the selected Polygons*

Headers:

Mask:Res:Poly<N>?

Parameter <N>: - number of the polygon, 1..8

Type – Data type commands

Argument: none

Forms: query only

Action: return the text string with value of the failed samples on the specified polygon.

### *Get Number of Samples in the Margins of the selected Polygon*

Headers:

Mask:Res:Poly<N>Mar?

Parameter <N>: - number of the polygon, 1...4

Type – Data type commands

Argument: none

Forms: query only

Action: return the text string with value of the failed samples on the margin of the specified polygon. It is used when Margins enable.

### *Get Number of Samples in the selected Polygon with Margins together*

Headers:

Mask:Res:Poly<N>All?

Parameter <N>: - number of the polygon, 1...4

Type – Data type commands

Argument: none

Forms: query only

Action: return the text string with sum value of the failed samples on the margin and on the specified polygon. It is used when Margins enable.



## **Eye Diagramm commands**

### **Setting Eye Parameters**

#### ***Type of Eye Measures***

Header: Eye:Measure

Type – Selector type command

Arguments: Off, NRZ, RZ

Action: set the type of the eye measurements.

#### ***Sources for Eye Measures***

Header: Eye:Source

Type – Selector type command

Arguments: Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3, F4, M1, M2, M3, M4, S1, S2

Action: set the source for the eye measurements.

#### ***Number of Waveforms by one Measurement***

Header: Eye:WfmsInCycle

Type – Integer type command

Argument: 64, 128, 256, 512, 1024

Action: set the number of waveforms by one measurement.

#### ***Visibility of the Eye Frame***

Header: Eye:DisplayWind

Type – On/Off command

Action: set the visibility of the eye frame.

#### ***Statistic of the Measurements***

Header: Eye:Statistic

Type – On/Off command

Action: enable/disable of the measurement's statistic.

#### ***Mode of the Measurement's Statistic***

Header: Eye:Mode

Type – Selector type command

Arguments: Permanent, Window, Weight

Action: set the mode of statistic calculation. It is used when statistic is enable.

***Windows Value***

Header: Eye:Window

Type – Integer type command

Argument: 8, 16, 32, ..., 8192

Action: set the windows value. It is used for Window mode of the statistic.

***Weight Value***

Header: Eye:Weight

Type – Integer type command

Argument: 8, 16, 32, ..., 8192

Action: set the weight value. It is used for Weight mode of the statistic.

***Left and Right Boundary for NRZ Top/Base Finding***

Headers:

Eye:LeftBound

Eye:RightBound

Type – Float type command

Argument: 10%...90% of the NRZ period

Action: set the zone of the period of the NRZ signal for the top/base calculation.

***Mode of the Threshold's Definitions***

Header: Eye:TreshMode

Type – Selector type command

Arguments: 10-90, 20-80, Custom

Action: set the mode of the threshold's definitions.

***Upper and Lower Threshold***

Headers:

Eye:UpTresh

Eye:LowTresh

Type – Float type command

Argument: 5%...95% of the amplitude

Action: set the thresholds for the slopes calculation. It is used for Custom mode only.

***List of X-axis NRZ Measurements***

Header: Eye:XNRZParam

Type – On/Off Group type command

Items: Area, BitRate, BitTime, CrossTime, CycleArea, DutCycDistP, DutCycDistS, EyeWidth, EyeWidthP, FallTime, Freq, JitterPP, JitterRMS, Period, RiseTime

Action: define the set of the X-axis measures for the NRZ signals.

### *List of Y-axis NRZ Measurements*

Header: Eye:YNRZParam

Type – On/Off Group type command

Items: AcRMS, AvgPower, AvgPWdBm, CrossPerc, CrossLevel, ExtRatioDB, ExtRatioP, ExtRatio, EyeAmpl, EyeHeight, EyeHeightDB, Max, Mean, Mid, Min, NegOver, PPNoiseOne, PPNoiseZero, RMSNoiseOne, RMSNoiseZero, OneLevel, PeakPeak, PosOver, RMS, SNRaio, SNRaioDB, ZeroLevel

Action: define the set of the Y-axis measures for NRZ signals.

### *List of X-axis RZ Measurements*

Header: Eye:XRZParam

Type – On/Off Group type command

Items: Area, BitRate, BitTime, CycleArea, EyeWidth, EyeWidthP, FallTime, JittPpFall, JittPpRise, JittRMSFall, JittRMSRise, NegCross, PosCross, PosDutyCyc, PulseSymm, PulseWidth, RiseTime

Action: define the set of the X-axis measures for RZ signals.

### *List of Y-axis RZ Measurements*

Header: Eye:YRZParam

Type – On/Off Group type command

Items: AcRMS, AvgPower, AvgPWdBm, Contrast, ContrastBb, ContrastP, ExtRatioDB, ExtRatioP, ExtRatio, EyeAmpl, EyeHeight, EyeHeightDB, EyeOpenFact, Max, Mean, Mid, Min, PPNoiseOne, PPNoiseZero, RmsNoiseOne, RMSNoiseZero, OneLevel, PeakPeak, RMS, SignToNoise, ZeroLevel

Action: define the set of the Y-axis measures for RZ signals.

## **Getting Eye Measures Results**

### *Get the List of Measured Parameters*

Header: Eye:Res:List?

Type – Data type commands

Argument: none

Forms: query only.

Action: return list of the active eye measures with ordinal index.

### ***Get Current Value of Parameter***

Header: Eye:Res:<N>?

Parameter <N>: - index of the parameter in the list

Type – Data type commands

Argument: none

Forms: query only.

Action: return the result of the specified measured parameter.

### ***Get Statistic Value of Parameter***

Header: Eye:Res:<N>:<Val>?

Parameter <N>: - index of the parameter in the list

Parameter <Val>: Wfm, Min, Max, Mean, StdDev

Type – Data type commands

Arguments: none

Forms: command with query only.

Action: return the specified statistic parameter of the measured parameter.

## ***Utilities commands***

### ***Start of the Autocalibration of the Channels***

Header: Flash:Calibr:AutocalCh

Type – Executing command

Action: Start the self-calibration of the channels.

### ***Start of the Autocalibration of Time Base***

Header: Flash:Calibr:AutocalTB

Type – Executing command

Action: Start the self-calibration of the time base.

### ***Get the Autocalibration status query***

Header: Flash:Calibr:AutocalResult?

Type – Integer type command

Action: command is ignored, query return the integer number:

0 - Autocalibration finished OK;

- 1 – Signal must be disconnected from Ch1 Input. Autocalibration of the Channels is aborted.
- 2 – Signal must be disconnected from Ch2 Input. Autocalibration of the Channels is aborted.
- 3 – Signal must be disconnected from Ch1 and Ch2 Inputs. Autocalibration of the Channels is aborted.
- 5. Autocalibration finished non correctly.

Version: This query can be used with PicoScope SW v.2.3.2 or later.

### ***When Device proceed the Autocalibration***

Header: Util:CalibrWhen

Type – On/Off Group type command

Items: PowerOn, Period, Temperat

Action: if item PowerOn is turned on the autocalibration is proceed during the first time after Power On; if item Period is turned on the autocalibration is proceed periodically with the specified interval; if the item Temperat is turned on the autocalibration is proceed when deviation of the temperature inside the instrument is exceed the specified value.

Note. For the PicoScope SW v.2.3.2 or later the autocalibration do not performed spontaneously independent of this command, if the Gui is in Invisible or RemoteOnly state (look at the GUI command above)

### ***Period of Autocalibration***

Header: Util:CalPeriod

Type – Float type command

Argument: (0.5...16) hours

Action: sets the autocalibration's period in hours.

### ***Deviation of the Temperature***

Header: Util:TempChange

Type – Float type command

Argument: (0.5...10) °C

Action: sets the deviation of the temperature for the autocalibration.

## **Waveforms commands**

This group of commands is destined for the receiving of the acquired waveforms from the oscilloscope.

### ***Waveform Source***

Header: Wfm:Source

Type – Selector type command

Arguments: Ch1, Ch1B2, Ch2, Ch2B2, F1, F2, F3, F4, M1, M2, M3, M4, S1, S2

Action: set the receiving signal.

### *Spectrum Format*

Header: Wfm:Complex

Type – Selector type command

Arguments: Mod, Ph, Re, Im

Action: select the receiving component of the complex signal. It used for the spectrum.

### *Get Waveform Data*

Header: Wfm:Data?

Type – Data type commands

Argument: none

Forms: query only

Action: return the text string with values of the all points of the signal (through comma).

### *Get number of Points in Waveform*

Header: Wfm:Preamb:Poin?

Type – Data type commands

Argument: none

Forms: query only

Action: return the quantity of the signal's points.

### *Get X-axis Step*

Header: Wfm:Preamb:XInc?

Type – Data type commands

Argument: none

Forms: query only

Action: return the increment of the X-axis value by every signal's points.

### *Get X-axis Origin*

Header: Wfm:Preamb:XOrg?

Type – Data type commands

Argument: none

Forms: query only

Action: return the X-axis value for the first signal's point.

***Get X-axis Unit***

Header: Wfm:Preamb:XU?

Type – Data type commands

Argument: none

Forms: query only

Action: return the X-axis physical units.

***Get Y-axis Unit***

Header: Wfm:Preamb:YU?

Type – Data type commands

Argument: none

Forms: query only

Action: return the Y-axis physical units.

## Programming Examples

Your PicoScope installation includes programming examples in the following languages and development environments:

### Delphi

The program:

PicoScopeDelphiClientExample.dproj

in the Delphi\_Client\_Example/ subdirectory of the PicoScope9000 SDK materials demonstrates how to operate PicoScope 9000 Series PC Oscilloscopes. The file:

PicoScope9000\_TLB.pas

is the description of the *PicoScope9000.COMRC* object. You must include this file in your own programs. Other required files of the example are:

MainClient.pas

MainClient.dfm

PicoScopeDelphiClientExample.dproj

PicoScopeDelphiClientExample.dsk

PicoScopeDelphiClientExample.identcache

PicoScopeDelphiClientExample.res

All these 8 files must be put into the same directory and compiled. This has been tested with Delphi 2009.

### LabView

The program:

PicoScope\_Example.vi

Test\_Get\_Data.vi

in the LabView\_Client\_Example/ subdirectory of the PicoScope9000 SDK materials demonstrates how to operate PicoScope 9000 Series PC Oscilloscopes.

This file must be put into the same directory and compiled. This has been tested with LabVIEW Base Development System 8.2.1.



## Visual Basic .NET

The project is located in the PicoScope9000VBdotNETClient/ subdirectory of the PicoScope9000 SDK. The subdirectory is a standard VB.NET project directory with the project file (PicoScope9000VBdotNETClient.vbproj), solution file (PicoScope9000VBdotNETClient.sln), MainForm.\* files and three subdirectories (bin/, "My Project/" and obj/).

Assembly obj/\*/Interop.PicoScope9000.dll, which is referenced in the source code as PicoScope9000.COMRC, is a bridge between .NET platform and Windows Component Object Model (COM). It has been created with Visual Basic IDE by executing the command Project | Add Reference... | COM | PicoScope9000.

File MainForm.vb contains the whole source code of the example. Other files were created automatically either by IDE itself or by visual form editor.

This example has been created and tested with Microsoft Visual Basic 2008 Express Edition.

## Last revisions

### *2010 October, 26*

- GUI command (p. 12) added;
- Get the Autocalibration status query (p. 52) added;
- When Device proceed the Autocalibration command (p. 53) modified;
- LabView Programming Example (p. 56) modified.