



# Hantek



# DPO7000 series

Digital oscilloscope

## Programming Guide

202304

## **Warranties and Declarations**

### **Copyright**

The copyright of this document belongs to Qingdao Hantek Electronics Co., LTD.

### **Statement**

Qingdao Hantek Electronics Co., Ltd. reserves the right to amend this document without prior notice. Qingdao Hantek Electronics Co., Ltd. promises that the information provided is correct and reliable but does not guarantee that this document is free from errors. Before using this product, please make sure that the specifications of relevant technical documents are the latest effective version. If you use documents or products of Qingdao Hantek Electronics Co., LTD and need products, patents or works of third parties to cooperate with them, you shall be responsible for obtaining the consent and authorization of the third parties. The above consent and authorization shall not be the liability of Hantek.

### **Product certification**

Hantek certified DPO series oscilloscope to meet China's national industry standards and has passed the CE certification.

### **Contact us**

If you have any questions when using the products of Qingdao Hantek Electronics Co., LTD., you can obtain service and support through the following ways:

Email: [service@hantek.com](mailto:service@hantek.com), [support@hantek.com](mailto:support@hantek.com)

Website: <http://www.hantek.com>

---

# Contents

|                                |    |
|--------------------------------|----|
| Contents .....                 | I  |
| 1 Document overview .....      | 1  |
| 2 SCPI Command Overview .....  | 3  |
| 3 Command System .....         | 6  |
| 3.1 General commands.....      | 6  |
| 3.1.1 *CLS.....                | 6  |
| 3.1.2 *IDN? .....              | 7  |
| 3.1.3 *RST .....               | 7  |
| 3.2 :AUToscale .....           | 8  |
| 3.3 :RUNing .....              | 8  |
| 3.4 :SINGle .....              | 9  |
| 3.5 :ACQuire Commands .....    | 10 |
| 3.5.1 :ACQuire:AVERages .....  | 10 |
| 3.5.2 :ACQuire: MDEPth.....    | 10 |
| 3.5.3 :ACQuire:TYPE.....       | 11 |
| 3.5.4 :ACQuire:SRATe? .....    | 12 |
| 3.6 :Calibration commands..... | 13 |
| 3.6.1 :CALibrate:QUIT .....    | 13 |

---

|       |                             |    |
|-------|-----------------------------|----|
| 3.6.2 | :CALibrate:START .....      | 13 |
| 3.7   | :CHANnel<n> Commands .....  | 14 |
| 3.7.1 | :CHANnel<n>:BWLimit .....   | 14 |
| 3.7.2 | :CHANnel<n>:COUPling .....  | 15 |
| 3.7.3 | :CHANnel<n>: DISPlay .....  | 15 |
| 3.7.4 | :CHANnel<n>: INVert .....   | 16 |
| 3.7.5 | :CHANnel<n>: OFFSet .....   | 17 |
| 3.7.6 | :CHANnel<n>: SCALE .....    | 18 |
| 3.7.7 | :CHANnel<n>:VERNier .....   | 18 |
| 3.7.8 | :CHANnel<n>:PROBe .....     | 19 |
| 3.7.9 | :CHANnel<n>:UNITs .....     | 20 |
| 3.8   | :CURSor Commands .....      | 20 |
| 3.8.1 | :CURSor:MODE .....          | 21 |
| 3.8.2 | :CURSor:MANual:TYPE .....   | 21 |
| 3.8.3 | :CURSor:MANual:SOURce ..... | 22 |
| 3.8.4 | :CURSor:MANual:AX .....     | 23 |
| 3.8.5 | :CURSor:MANual:BX .....     | 23 |
| 3.8.6 | :CURSor:MANual:AY .....     | 24 |
| 3.8.7 | :CURSor:MANual:BY .....     | 25 |

---

|        |                                  |    |
|--------|----------------------------------|----|
| 3.8.8  | :CURSor:MANual:AXValue?.....     | 25 |
| 3.8.9  | :CURSor:MANual:AYValue?.....     | 26 |
| 3.8.10 | :CURSor:MANual:BXValue?.....     | 26 |
| 3.8.11 | :CURSor:MANual:BYValue?.....     | 27 |
| 3.8.12 | :CURSor:MANual:XDELta? .....     | 28 |
| 3.8.13 | :CURSor:MANual:IXDELta? .....    | 28 |
| 3.8.14 | :CURSor:MANual:YDELta? .....     | 29 |
| 3.8.15 | :CURSor:TRACk:SOURce1.....       | 29 |
| 3.8.16 | :CURSor:TRACk:AX .....           | 30 |
| 3.8.17 | :CURSor:TRACk:BX .....           | 31 |
| 3.9    | :MEASure Commands.....           | 31 |
| 3.9.1  | :MEASure:SOURce .....            | 31 |
| 3.9.2  | :MEASure:CLEar .....             | 32 |
| 3.9.3  | :MEASure:AMSource.....           | 33 |
| 3.9.4  | :MEASure:STATistic:DISPlay ..... | 33 |
| 3.9.5  | :MEASure:ITEM.....               | 34 |
| 3.9.6  | :MEASure:FREQuency.....          | 35 |
| 3.9.7  | :MEASure:PERiod .....            | 36 |
| 3.9.8  | :MEASure:RTIME.....              | 36 |

---

|        |                        |    |
|--------|------------------------|----|
| 3.9.9  | :MEASure:FTIME .....   | 37 |
| 3.9.10 | :MEASure:PWIDth .....  | 38 |
| 3.9.11 | :MEASure:NWIDth .....  | 38 |
| 3.9.12 | :MEASure:PDUTy .....   | 39 |
| 3.9.13 | :MEASure:NDUTy .....   | 40 |
| 3.9.14 | :MEASure:BWIDth .....  | 40 |
| 3.9.15 | :MEASure:MAXTIME ..... | 41 |
| 3.9.16 | :MEASure:MINTIME ..... | 42 |
| 3.9.17 | :MEASure:VMEAn .....   | 42 |
| 3.9.18 | :MEASure:VMAX .....    | 43 |
| 3.9.19 | :MEASure:VMIN .....    | 44 |
| 3.9.20 | :MEASure:PKPk .....    | 44 |
| 3.9.21 | :MEASure:VTOP .....    | 45 |
| 3.9.22 | :MEASure:VMID .....    | 46 |
| 3.9.23 | :MEASure:VBASe .....   | 46 |
| 3.9.24 | :MEASure:VAMp .....    | 47 |
| 3.9.25 | :MEASure:VRMS .....    | 48 |
| 3.9.26 | :MEASure:VOVr .....    | 48 |
| 3.9.27 | :MEASure:VPEr .....    | 49 |

---

|        |                               |    |
|--------|-------------------------------|----|
| 3.9.28 | :MEASure:PVRMS.....           | 50 |
| 3.9.29 | :MEASure:PVMEAS .....         | 50 |
| 3.9.30 | :MEASure:VFOv .....           | 51 |
| 3.9.31 | :MEASure:VRPr.....            | 52 |
| 3.10   | :SYSTem Commands.....         | 53 |
| 3.10.1 | :SYSTem:DATE.....             | 53 |
| 3.10.2 | :SYSTem:TIME.....             | 53 |
| 3.11   | :TIMEbase Commands.....       | 54 |
| 3.11.1 | :TIMEbase:WINDow:ENABLE ..... | 54 |
| 3.11.2 | :TIMEbase[:MAIN]:SCALE .....  | 55 |
| 3.11.3 | :TIMEbase[:MAIN]:OFFSet ..... | 55 |
| 3.12   | :TRIGger Commands .....       | 56 |
| 3.12.1 | :TRIGger:MODE .....           | 56 |
| 3.12.2 | :TRIGger:STATus? .....        | 57 |
| 3.12.3 | :TRIGger:SWEep.....           | 58 |
| 3.12.4 | :TRIGger:HOLDoff.....         | 58 |
| 3.12.5 | :TRIGger:EDGE:SOURce .....    | 59 |
| 3.12.6 | :TRIGger:EDGE:SLOPe.....      | 60 |
| 3.12.7 | :TRIGger:EDGE:LEVel.....      | 60 |

---

|         |                               |    |
|---------|-------------------------------|----|
| 3.12.8  | :TRIGger:PULSe:SOURce .....   | 61 |
| 3.12.9  | :TRIGger:PULSe:POLarity.....  | 61 |
| 3.12.10 | :TRIGger:PULSe:WHEN .....     | 62 |
| 3.12.11 | :TRIGger:PULSe:WIDTh.....     | 63 |
| 3.12.12 | :TRIGger:PULSe:LEVel .....    | 63 |
| 3.12.13 | :TRIGger:SLOPe:SOURce.....    | 64 |
| 3.12.14 | :TRIGger:SLOPe:POLarity ..... | 65 |
| 3.12.15 | :TRIGger:SLOPe:WHEN.....      | 65 |
| 3.12.16 | :TRIGger:SLOPe:TIME .....     | 66 |
| 3.12.17 | :TRIGger:SLOPe:ALEVel.....    | 67 |
| 3.12.18 | :TRIGger:SLOPe:BLEVel.....    | 67 |
| 3.12.19 | :TRIGger:VIDeo:SOURce .....   | 68 |
| 3.12.20 | :TRIGger:VIDeo:POLarity ..... | 69 |
| 3.12.21 | :TRIGger:VIDeo:MODE.....      | 69 |
| 3.12.22 | :TRIGger:VIDeo:LINE .....     | 70 |
| 3.12.23 | :TRIGger:VIDeo:LEVel.....     | 70 |
| 3.12.24 | :TRIGger:TIMEout:SOURce ..... | 71 |
| 3.12.25 | :TRIGger:TIMEout:SLOPe.....   | 72 |
| 3.12.26 | :TRIGger:TIMEout:TIME .....   | 72 |

---

|         |                                |    |
|---------|--------------------------------|----|
| 3.12.27 | :TRIGger:TIMEout:LEVel.....    | 73 |
| 3.12.28 | :TRIGger:RUNT:SOURce .....     | 74 |
| 3.12.29 | :TRIGger:RUNT:POLarity .....   | 74 |
| 3.12.30 | :TRIGger:RUNT:WHEN .....       | 75 |
| 3.12.31 | :TRIGger:RUNT:TIME.....        | 76 |
| 3.12.32 | :TRIGger:RUNT:ALEVel .....     | 76 |
| 3.12.33 | :TRIGger:RUNT:BLEVel .....     | 77 |
| 3.12.34 | :TRIGger: PATTern:SOURce.....  | 77 |
| 3.12.35 | :TRIGger:PATTern:CODE.....     | 78 |
| 3.12.36 | :TRIGger:PATTern:PATTern ..... | 79 |
| 3.12.37 | :TRIGger:DELay:SA.....         | 79 |
| 3.12.38 | :TRIGger:DELay:SLOPA.....      | 80 |
| 3.12.39 | :TRIGger:DELay:SB .....        | 81 |
| 3.12.40 | :TRIGger:DELay:SLOPB.....      | 81 |
| 3.12.41 | :TRIGger:DELay:WHEN.....       | 82 |
| 3.12.42 | :TRIGger:DELay:TIME .....      | 83 |
| 3.12.43 | :TRIGger:SHOLd:DSrc.....       | 83 |
| 3.12.44 | :TRIGger:SHOLd:CSrc.....       | 84 |
| 3.12.45 | :TRIGger:SHOLd:SLOPe .....     | 84 |

---

|         |                              |    |
|---------|------------------------------|----|
| 3.12.46 | :TRIGger:SHOLd:PATtern ..... | 85 |
| 3.12.47 | :TRIGger:SHOLd:TYPe.....     | 86 |
| 3.12.48 | :TRIGger:SHOLd:STIMe .....   | 86 |
| 3.12.49 | :TRIGger:UART:SOURce .....   | 87 |
| 3.12.50 | :TRIGger:UART:WHEN.....      | 88 |
| 3.12.51 | :TRIGger:UART:BAUD.....      | 88 |
| 3.12.52 | :TRIGger:UART:LEVel .....    | 89 |
| 3.12.53 | :TRIGger:UART:DATA .....     | 90 |
| 3.12.54 | :TRIGger:UART:WIDTh .....    | 90 |
| 3.12.55 | :TRIGger:UART:PARity.....    | 91 |
| 3.12.56 | :TRIGger:UART:POLarity.....  | 91 |
| 3.12.57 | :TRIGger:LIN:SOURce.....     | 92 |
| 3.12.58 | :TRIGger:LIN:BAUD .....      | 93 |
| 3.12.59 | :TRIGger:LIN:CONdition ..... | 93 |
| 3.12.60 | :TRIGger:LIN:ID .....        | 94 |
| 3.12.61 | :TRIGger:LIN:LEVel .....     | 95 |
| 3.12.62 | :TRIGger:CAN:SOURce.....     | 95 |
| 3.12.63 | :TRIGger:CAN:BAUD .....      | 96 |
| 3.12.64 | :TRIGger:CAN:CONdition ..... | 97 |

---

|         |                                |     |
|---------|--------------------------------|-----|
| 3.12.65 | :TRIGger:CAN:LEVel .....       | 97  |
| 3.12.66 | :TRIGger:SPI:MSIO:SOURce ..... | 98  |
| 3.12.67 | :TRIGger:SPI:SCL:SOURce .....  | 99  |
| 3.12.68 | :TRIGger:SPI:SLOPe .....       | 99  |
| 3.12.69 | :TRIGger:SPI:WIDTh .....       | 100 |
| 3.12.70 | :TRIGger:SPI:TIMEout .....     | 100 |
| 3.12.71 | :TRIGger:SPI:CLEVel .....      | 101 |
| 3.12.72 | :TRIGger:SPI:DLEVel .....      | 102 |
| 3.12.73 | :TRIGger:IIC:SDA:SOURce .....  | 102 |
| 3.12.74 | :TRIGger:IIC:SCL:SOURce .....  | 103 |
| 3.12.75 | :TRIGger:IIC:WHEN .....        | 103 |
| 3.12.76 | :TRIGger:IIC:ADDResS .....     | 104 |
| 3.12.77 | :TRIGger:IIC:AWIDth .....      | 105 |
| 3.12.78 | :TRIGger:IIC:CLEVel.....       | 105 |
| 3.12.79 | :TRIGger:IIC:DLEVel.....       | 106 |
| 3.13    | :MASK Commands.....            | 107 |
| 3.13.1  | :MASK:ENABLe .....             | 107 |
| 3.13.2  | :MASK:SOURce .....             | 107 |
| 3.13.3  | :MASK:OPERate .....            | 108 |

---

|         |                                           |     |
|---------|-------------------------------------------|-----|
| 3.13.4  | :MASK:MDISplay .....                      | 109 |
| 3.13.5  | :MASK:SOOutput.....                       | 109 |
| 3.13.6  | :MASK:OUTPut.....                         | 110 |
| 3.13.7  | :MASK:X .....                             | 111 |
| 3.13.8  | :MASK:Y .....                             | 111 |
| 3.13.9  | :MASK:CREate .....                        | 112 |
| 3.13.10 | :MASK:PASSed? .....                       | 112 |
| 3.13.11 | :MASK:FAILED? .....                       | 113 |
| 3.13.12 | :MASK:TOTal? .....                        | 113 |
| 3.13.13 | :MASK:RESet.....                          | 114 |
| 3.14    | :LA Commands .....                        | 114 |
| 3.14.1  | :LA:POD<n>:DISPlay .....                  | 115 |
| 3.14.2  | :LA:POD<n>:THReshold .....                | 115 |
| 3.15    | [ :SOURce] Commands.....                  | 116 |
| 3.15.1  | [ :SOURce<n>]:OUTPut[<n>]:[STATe] .....   | 116 |
| 3.15.2  | [ :SOURce<n>]:OUTPut[<n>]:IMPedance ..... | 117 |
| 3.15.3  | [ :SOURce<n>]:FREQuency[:FIXed] .....     | 117 |
| 3.15.4  | [ :SOURce<n>]:PHASe[:ADJust] .....        | 118 |
| 3.15.5  | [ :SOURce<n>]:FUNCTion[:SHApe].....       | 119 |

|         |                                                             |     |
|---------|-------------------------------------------------------------|-----|
| 3.15.6  | [:SOURce[<n>]]:FUNCTion:RAMP:SYMMetry .....                 | 120 |
| 3.15.7  | [:SOURce[<n>]]:VOLTage[:LEVel][:IMMEDIATE][:AMPLitude]..... | 121 |
| 3.15.8  | [:SOURce[<n>]]: VOLTage[:LEVel] [:IMMEDIATE]:OFFSet.....    | 121 |
| 3.15.9  | [:SOURce[<n>]]:PULSe:DCYCLE .....                           | 122 |
| 3.15.10 | [:SOURce[<n>]]:MOD[:STATe] .....                            | 123 |
| 3.15.11 | [:SOURce[<n>]]:MOD:TYPE .....                               | 124 |
| 3.15.12 | [:SOURce[<n>]]:MOD:AM:INTernal:FREQuency .....              | 124 |
| 3.15.13 | [:SOURce[<n>]]:MOD:AM[:DEPTH] .....                         | 126 |
| 3.15.14 | [:SOURce[<n>]]:MOD:FM[:DEVlation] .....                     | 126 |
| 3.15.15 | [:SOURce[<n>]]:MOD:PM[:DEPTH] .....                         | 127 |
| 3.15.16 | [:SOURce[<n>]]:MOD:PWM[:DEPTH] .....                        | 128 |
| 3.15.17 | [:SOURce[<n>]]:APPLY .....                                  | 129 |
| 3.16    | :MATH Commands.....                                         | 130 |
| 3.16.1  | :MATH:DISPlay .....                                         | 130 |
| 3.16.2  | :MATH:OPERator .....                                        | 131 |
| 3.16.3  | :MATH:SOURce1 .....                                         | 132 |
| 3.16.4  | :MATH:SOURce2 .....                                         | 133 |
| 3.16.5  | :MATH:SCALE .....                                           | 133 |
| 3.16.6  | :MATH:OFFSet .....                                          | 134 |

---

|         |                                 |     |
|---------|---------------------------------|-----|
| 3.16.7  | :MATH:FFT:SOURce.....           | 135 |
| 3.16.8  | :MATH:FFT:WINDow.....           | 135 |
| 3.16.9  | :MATH:FFT:UNIT.....             | 136 |
| 3.16.10 | :MATH:FFT:HSCale .....          | 137 |
| 3.16.11 | :MATH:FFT:HCENter .....         | 137 |
| 3.17    | :SEGMENTation Commands.....     | 138 |
| 3.17.1  | :SEGMENTation:WRECORD .....     | 138 |
| 3.17.2  | :SEGMENTation:ENABLE.....       | 139 |
| 3.18    | :HISTORY Commands .....         | 139 |
| 3.18.1  | :HISTORY:WRECORD.....           | 139 |
| 3.18.2  | :HISTORY:WRECORD:FINTerval..... | 140 |
| 3.19    | :DISPlay Commands.....          | 141 |
| 3.19.1  | :DISPlay:TYPE .....             | 141 |
| 3.19.2  | :DISPlay:GRADing:TIME .....     | 141 |
| 3.19.3  | :DISPlay:WBRightness .....      | 142 |
| 3.19.4  | :DISPlay:GRID .....             | 143 |
| 3.19.5  | :DISPlay:GBRightness.....       | 143 |
| 3.20    | :BUS Commands .....             | 144 |
| 3.20.1  | :BUS:CAN:SOURce.....            | 144 |

---

|         |                               |     |
|---------|-------------------------------|-----|
| 3.20.2  | :BUS:CAN:THReshold.....       | 145 |
| 3.20.3  | :BUS:CAN:STYPe .....          | 145 |
| 3.20.4  | :BUS:CAN:BAUD .....           | 146 |
| 3.20.5  | :BUS:CAN:SPOint .....         | 146 |
| 3.20.6  | :BUS:IIC:SCLK:SOURce .....    | 147 |
| 3.20.7  | :BUS:IIC:SDA:SOURce .....     | 148 |
| 3.20.8  | :BUS:IIC:SCLK:THReshold ..... | 148 |
| 3.20.9  | :BUS:IIC:SDA:THReshold .....  | 149 |
| 3.20.10 | :BUS:IIC:ADDRes .....         | 149 |
| 3.20.11 | :BUS:LIN:SOURce .....         | 150 |
| 3.20.12 | :BUS:LIN:THReshold .....      | 151 |
| 3.20.13 | :BUS:LIN:BAUD .....           | 151 |
| 3.20.14 | :BUS:LIN:PARity .....         | 152 |
| 3.20.15 | :BUS:LIN:STANdard .....       | 153 |
| 3.20.16 | :BUS:RS232:BAUD .....         | 153 |
| 3.20.17 | :BUS:RS232:TX .....           | 154 |
| 3.20.18 | :BUS:RS232:TX:THReshold ..... | 154 |
| 3.20.19 | :BUS:RS232:RX .....           | 155 |
| 3.20.20 | :BUS:RS232:RX:THReshold ..... | 156 |

---

|         |                                      |     |
|---------|--------------------------------------|-----|
| 3.20.21 | :BUS:RS232:POLarity .....            | 156 |
| 3.20.22 | :BUS:RS232:ENDian .....              | 157 |
| 3.20.23 | :BUS:RS232:DBITs .....               | 157 |
| 3.20.24 | :BUS:RS232:PARity .....              | 158 |
| 3.20.25 | :BUS:RS232:STOP .....                | 159 |
| 3.20.26 | :BUS:SPI:SCLK:SOURce .....           | 159 |
| 3.20.27 | :BUS:SPI:SCLK:SLOPe .....            | 160 |
| 3.20.28 | :BUS:SPI:MISO:SOURce .....           | 160 |
| 3.20.29 | :BUS:SPI:MOSI:SOURce .....           | 161 |
| 3.20.30 | :BUS:SPI:SCLK:SOURce:THReshold ..... | 162 |
| 3.20.31 | :BUS:SPI:MISO:SOURce:THReshold ..... | 162 |
| 3.20.32 | :BUS:SPI:MOSI:SOURce:THReshold ..... | 163 |
| 3.20.33 | :BUS:SPI:MODE .....                  | 163 |
| 3.20.34 | :BUS:SPI:CS:SOURce .....             | 164 |
| 3.20.35 | :BUS:SPI:CS:POLarity .....           | 165 |
| 3.20.36 | :BUS:SPI:CS:THReshold .....          | 165 |
| 3.20.37 | :BUS:SPI:ENDian .....                | 166 |
| 3.20.38 | :BUS:SPI:DBITs .....                 | 166 |
| 3.20.39 | :BUS:SPI:POLarity .....              | 167 |

---

|         |                                    |     |
|---------|------------------------------------|-----|
| 3.21    | :SEARCh Commands .....             | 168 |
| 3.21.1  | :SEARCh:STATUs .....               | 168 |
| 3.21.2  | :SEARCh:TYPE .....                 | 168 |
| 3.21.3  | :SEARCh:MARK:STATUs .....          | 169 |
| 3.21.4  | :SEARCh:EDGE:SOURce .....          | 169 |
| 3.21.5  | :SEARCh:EDGE:SLOPe .....           | 170 |
| 3.21.6  | :SEARCh:EDGE:THReshold .....       | 171 |
| 3.21.7  | :SEARCh:SLOPe:SOURce .....         | 171 |
| 3.21.8  | :SEARCh:SLOPe:SLOPe .....          | 172 |
| 3.21.9  | :SEARCh:SLOPe:WHEN .....           | 172 |
| 3.21.10 | :SEARCh:SLOPe:TIME .....           | 173 |
| 3.21.11 | :SEARCh:SLOPe:HIGh:THReshold ..... | 174 |
| 3.21.12 | :SEARCh:SLOPe:LOW:THReshold .....  | 174 |
| 3.21.13 | :SEARCh:RUNT:SOURce .....          | 175 |
| 3.21.14 | :SEARCh:RUNT:POLarity .....        | 175 |
| 3.21.15 | :SEARCh:RUNT:WHEN .....            | 176 |
| 3.21.16 | :SEARCh:RUNT:TIME .....            | 177 |
| 3.21.17 | :SEARCh:RUNT:HIGh:THReshold .....  | 177 |
| 3.21.18 | :SEARCh:RUNT:LOW:THReshold .....   | 178 |

---

|         |                               |     |
|---------|-------------------------------|-----|
| 3.21.19 | :SEARCh:PULSe:SOURce .....    | 178 |
| 3.21.20 | :SEARCh:PULSe:SLOPe.....      | 179 |
| 3.21.21 | :SEARCh:PULSe:WHEN .....      | 180 |
| 3.21.22 | :SEARCh:PULSe:TIME.....       | 180 |
| 3.21.23 | :SEARCh:PULSe:THReshold ..... | 181 |
| 3.21.24 | :SEARCh:RS232:SOURce .....    | 182 |
| 3.21.25 | :SEARCh:RS232:POLarity.....   | 182 |
| 3.21.26 | :SEARCh:RS232:BAUD.....       | 183 |
| 3.21.27 | :SEARCh:RS232:WHEN.....       | 183 |
| 3.21.28 | :SEARCh:RS232:DBITs.....      | 184 |
| 3.21.29 | :SEARCh:RS232:PARity.....     | 185 |
| 3.21.30 | :SEARCh:RS232:STOP .....      | 185 |
| 3.21.31 | :SEARCh:RS232:DATA .....      | 186 |
| 3.21.32 | :SEARCh:RS232:THReshold ..... | 186 |
| 3.21.33 | :SEARCh:IIC:SCLK:SOURce ..... | 187 |
| 3.21.34 | :SEARCh:IIC:SDA:SOURce .....  | 188 |
| 3.21.35 | :SEARCh:IIC:WHEN .....        | 188 |
| 3.21.36 | :SEARCh:IIC:ADDRes .....      | 189 |
| 3.21.37 | :SEARCh:IIC:DIRection .....   | 189 |

---

|         |                                  |     |
|---------|----------------------------------|-----|
| 3.21.38 | :SEARCh:IIC:SCLK:THReshold ..... | 190 |
| 3.21.39 | :SEARCh:IIC:SDA:THReshold .....  | 191 |
| 3.21.40 | :SEARCh:SPI:SCLK:SOURce .....    | 191 |
| 3.21.41 | :SEARCh:SPI:SCLK:SLOPe .....     | 192 |
| 3.21.42 | :SEARCh:SPI:MISO:SOURce .....    | 192 |
| 3.21.43 | :SEARCh:SPI:CS:SOURce .....      | 193 |
| 3.21.44 | :SEARCh:SPI:CS:POLarity .....    | 194 |
| 3.21.45 | :SEARCh:SPI:WHEN .....           | 194 |
| 3.21.46 | :SEARCh:SPI:DBITs .....          | 195 |
| 3.21.47 | :SEARCh:SPI:TIMEout:TIME .....   | 195 |
| 3.21.48 | :SEARCh:SPI:DATA .....           | 196 |
| 3.21.49 | :SEARCh:SPI:SCLK:THResold .....  | 196 |
| 3.21.50 | :SEARCh:SPI:MISO:THReshold ..... | 197 |
| 3.21.51 | :SEARCh:SPI:CS:THReshold .....   | 198 |



# 1 Document overview

This document describes how to quickly understand the front and back panels, user interfaces, and basic operation methods of the DPO7000 series digital oscilloscopes.



## Tip:

The latest version of this manual can be downloaded at (<http://www.hantek.com>).

## Document number:

202304

## Software version:

Software upgrade may change or increase product functionalities, please pay attention to Hantek website for the latest version.

## Document format conventions:

### 1 Virtual keys and main interface icons

Use **[name]** to represent virtual keys and main interface icons. For example, **[Utility]** is for .

### 2 Menu

Use "menu text (bold) + color" to represent a label or a menu option. For example, **I/O** means to click the "I/O" option on the current operation interface to enter the function configuration menu of "I/O".

### 3 Operation steps

Use "->" to represent the next step. For example, **[Utility]** -> **Language** means click **Utility** label before clicking **Language** menu.

## Document content conventions:

DPO7000 series tablet oscilloscope consists of the following models. Unless otherwise specified, this manual uses DPO7504E as an example to describe the DPO7000 series and basic operations.

| Model    | Bandwidth | Channel | Signal source | Digital Channels |
|----------|-----------|---------|---------------|------------------|
| DPO7104E | 100MHz    | 4       | 1             | 16               |
| DPO7102E | 100 MHz   | 2       | 1             | 16               |
| DPO7104C | 100 MHz   | 4       | -             | -                |

| Model    | Bandwidth | Channel | Signal source | Digital Channels |
|----------|-----------|---------|---------------|------------------|
| DPO7102C | 100 MHz   | 2       | -             | -                |
| DPO7204E | 200 MHz   | 4       | 1             | 16               |
| DPO7202E | 200 MHz   | 2       | 1             | 16               |
| DPO7204C | 200 MHz   | 4       | -             | -                |
| DPO7202C | 200 MHz   | 2       | -             | -                |
| DPO7354E | 350 MHz   | 4       | 1             | 16               |
| DPO7352E | 350 MHz   | 2       | 1             | 16               |
| DPO7354C | 350 MHz   | 4       | -             | -                |
| DPO7352C | 350 MHz   | 2       | -             | -                |
| DPO7504E | 500 MHz   | 4       | 1             | 16               |
| DPO7502E | 500 MHz   | 2       | 1             | 16               |
| DPO7504C | 500 MHz   | 4       | -             | -                |
| DPO7502C | 500 MHz   | 2       | -             | -                |

## 2 SCPI Command Overview

SCPI (Standard Commands for Programmable Instruments) is a standardized instrument programming language that is built upon the existing standard IEEE 488.1 and IEEE 488.2 and conforms to various standards, such as the floating point operation rule in IEEE 754 standard, ISO 646 7-bit coded character set for information interchange (equivalent to ASCII programming). The SCPI commands provide a hierarchical tree structure, and consist of multiple subsystems. Each command subsystem consists of one root keyword and one or more sub-keywords.

### Syntax

The command line usually starts with a colon; the keywords are separated by colons, and following the keywords are the parameter settings available. The command ending with a quotation mark indicates querying a certain function and returns the query results. The keywords of the command and the first parameter are separated by a space.

For example:

```
:ACQUIRE:TYPE <type>
```

```
:ACQUIRE:TYPE?
```

ACQUIRE is the root keyword of the command, TYPE is the second-level keyword. The command line starts with a colon ":", and different levels of keywords are also separated by colons. <type> indicates a settable parameter. The command ending with a quotation mark "?" indicates querying a certain function. The command keywords :ACQUIRE:TYPE and the parameter <type> are separated by a space.

In some commands with parameters, ",", " is often used to separate multiple parameters.

For example:

```
[ :TRACe[<n>] ]:DATA:VALue volatile, <points>, <data>
```

### Symbol Description

The following symbols are not sent with the commands.

#### 1. Braces { }

The contents in the braces can contain one or multiple parameters. These parameters can be omitted or used for several times. Parameters are usually separated by the vertical bar "|". When using the command, you must select one of the parameters.

#### 2. Vertical Bar |

The vertical bar is used to separate multiple parameters. When using the command, you must select one of the parameters.

#### 3. Square Brackets [ ]

The contents in the square brackets can be omitted.

#### 4. Angle Brackets < >

The parameter enclosed in the angle brackets must be replaced by an effective value.

### Parameter Type

#### 1. Bool

The parameter can be set to ON, OFF, 1, or 0. For example:

:MEASure:ADISplay <bool>

:MEASure:ADISplay?

Wherein, <bool> can be set to {{1|ON}}{{0|OFF}}. The query returns 1 or 0.

#### 2. Discrete

The parameter can be any of the values listed. For example:

:ACQuire:TYPE <type>

:ACQuire:TYPE?

Wherein,

- <type> can be set to NORMal|AVERages|PEAK|HRESolution.

- The query returns an abbreviated form: NORM, AVER, PEAK or HRES.

#### 3. Integer

Unless otherwise specified, the parameter can be any integer (NR1 format) within the effective value range.



#### NOTE

**Do not set the parameter to a decimal, otherwise, errors will occur.**

For example:

:DISPlay:GBRightness <brightness>

:DISPlay:GBRightness?

Wherein, <brightness> can be set to an integer ranging from 0 to 100. The query returns an integer ranging from 0 to 100.

#### 4. Real

The parameter can be any real number within the effective value range, and this command accepts parameter input in decimal (NR2 format) and scientific notation (NR3 format). For example:

:TRIGger:TIMEout:TIME <time>

:TRIGger:TIMEout:TIME?

Wherein, <time> can be set to any real number ranging from 1.6E-8 (that is, 16 ns) to 1E+1 (that is, 10 s). The query returns a real number in scientific notation.

#### 5. ASCII String

The parameter can be the combinations of ASCII characters. For example:

:SYSTem:OPTion:INSTall <license>

Wherein, < license > can be set to PDUY9N9QTS9PQSWPLAETRD3UJHYA

**Command Abbreviation**

All the commands are case-insensitive. They can all be in upper case or in lower case. If abbreviation is used, you must input all the capital letters in the command. For example:

:DISPlay:GBRightness?

can be abbreviated to

:DISP:GBR?

## 3 Command System

---

This chapter introduces the syntax, functions, parameters, and usage of each DPO7000 command.

### NOTE

1. Unless otherwise specified, the descriptions in this manual take DPO7504E as an example to illustrate the commands of the DPO7000 series.
2. For the parameter setting command (time, frequency, amplitude, etc.), the digital oscilloscope can only recognize the numbers, unable to recognize the unit sent together with them. The unit of the parameter is a default one. For the default units of various parameters, refer to the descriptions for the specified command.

### 3.1 General commands

---

General commands are used to query basic instrument information or perform commonly used basic operations. These commands usually start with "\*" and the length of the command keyword is 3 characters.

#### 3.1.1 \*CLS

---

##### Syntax

\*CLS

##### Description

Clear the values of all event registers to zero while clearing the error queue.

##### Parameter

N/A

##### Remarks

N/A

##### Return Format

N/A

##### Example

\*CLS 16        /\*Enable bit 4 (decimal 16) of the register.\*/\*

### 3.1.2 \*IDN?

---

**Syntax**

\*IDN?

**Description**

Query the ID string of the instrument.

**Parameter**

N/A

**Remarks**

N/A

**Return Format**

Query returns HANTEK, <model>, <serial number>, <software version>.

Among them,

<model>: Instrument model;

<serial number>: instrument serial number;

<software version>: Instrument software version.

**Example**

\*IDN?            /\*HANTEK, DPO7504E, CN2142000000035, (2023.03.25)\*/

### 3.1.3 \*RST

---

**Syntax**

\*RST

**Description**

Restore the instrument to its factory default state.

**Parameter**

N/A

**Remarks**

N/A

**Return Format**

N/A

**Example**

\*RST            /\*Restore the instrument to its factory default state\*/

## 3.2 :AUToscale

---

**Syntax**

:AUToscale

**Description**

Enable the automatic waveform setting function. The oscilloscope will automatically adjust the vertical gear, horizontal time base, and triggering method based on the input signal to achieve the optimal waveform display. This command function is equivalent to pressing the **[AUTO Scale]** button on the front panel.

**Parameter**

N/A

**Remarks**

- When applying the waveform automatic setting function, for sinusoidal signals, the theoretical frequency should not be less than 10Hz; For square wave signals, the theoretical value is related to the duty cycle, which should be greater than 1% and have an amplitude of at least 10mVpp (when the probe ratio is 1X).
- When the current status of the pass/fail function is Allow Test (see MASK: ENABLE command), if this command is sent, the oscilloscope first turns off the pass/fail function, and then executes the waveform automatic setting function.
- When the waveform recording function is turned on or when the recorded waveform is played back, this command is invalid.

**Return Format**

N/A

**Example**

:AUToscale            /\*The oscilloscope performs the AUTO function\*/

## 3.3 :RUNing

---

**Syntax**

:RUNing

**Description**

Command to start the oscilloscope operation.

#### Parameter

| Name  | Type     | Range             | Default |
|-------|----------|-------------------|---------|
| <type | Discrete | {{1 ON}}{{0 OFF}} | ON      |

#### Remarks

This command is invalid when the waveform recording function is turned on or when the recorded waveform is played back.

#### Return Format

N/A

#### Example

:RUNing ON      /\*The oscilloscope starts running\*/

## 3.4 :SINGle

#### Syntax

:SINGle

#### Description

Set the oscilloscope to single trigger mode. This command function is equivalent to pressing the **[Single]** button on the front panel.

#### Parameter

N/A

#### Remarks

- In a single trigger mode, the oscilloscope will trigger once and then stop when the trigger conditions are met.
- When the waveform recording function is turned on or when the recorded waveform is played back, this command is invalid.

#### Return Format

N/A

#### Example

:SINGle      /\*Single acquisition of oscilloscope\*/

## 3.5 :ACQuire Commands

The :ACQuire commands are used to set the memory depth of the oscilloscope, the acquisition mode, the average times, as well as query the current sample rate.

### 3.5.1 :ACQuire:AVERages

#### Syntax

```
:ACQuire:AVERages <Vaule>
```

```
:ACQuire:AVERages?
```

#### Description

Set or query the average number of times under the average acquisition method.

#### Parameter

| Name    | Type    | Range                                                 | Default |
|---------|---------|-------------------------------------------------------|---------|
| <Vaule> | Integer | 2^n (n is an integer, and its range is from 1 to 10). | 2       |

#### Remarks

- Can send: ACQuire: TYPE command to set the acquisition method.
- Under the average acquisition method, the higher the average number of times, the smaller the noise and higher the vertical resolution of the collected waveform, but the slower the response of the displayed waveform to waveform changes.

#### Return Format

The query returns an integer between 2 and 1024.

#### Example

```
:ACQuire:AVERages 64 /*Set the average number of times to 64*/
```

```
:ACQuire:AVERages? /*Query returned 64*/
```

### 3.5.2 :ACQuire: MDEPth

#### Syntax

```
:ACQuire:MDEPth <mdep>
```

```
:ACQuire:MDEPth?
```

#### Description

Set or query the storage depth of the oscilloscope (i.e. the number of waveform points

that can be stored in a single trigger acquisition), with the default unit being pts (points).

#### Parameter

| Name   | Type     | Range                            | Default |
|--------|----------|----------------------------------|---------|
| <medp> | Discrete | Please refer to the instructions | AUTO    |

#### Remarks

For analog channels:

- Single channel open, the range of<medp>is  
{AUTO|25,000|250,000|2,500,000|25,000,000|50,000,000|100,000,000|250,000,000|500,000,000|2,000,000,000}.
- When dual channels are open, the range of<medp>is  
{AUTO|12,500|125,000|1,250,000|12,500,000|25,000,000|50,000,000|125,000,000|250,000,000|1,000,000,000}.
- When three/four channels are open, the range of<medp>is  
{AUTO|6,250|62,500|625,000|6,250,000|12,500,000|25,000,000|62,500,000|125,000,000|500,000,000}.

The relationship between storage depth, sampling rate, and waveform length satisfies the following equation:

Storage depth=sampling rate × When selecting AUTO for waveform length, the oscilloscope automatically selects the storage depth based on the current sampling rate.

#### Return Format

The query returns the actual number of points (integer) or AUTO.

#### Example

```
:ACQuire:MDEPth 25000000      /*Set storage depth to 25M*/
:ACQuire:MDEPth?              /*Query returned 32000*/
```

### 3.5.3 :ACQuire:TYPE

#### Syntax

```
:ACQuire:TYPE <Value>
:ACQuire:TYPE?
```

#### Description

Set or query the acquisition method for oscilloscope sampling.

#### Parameter

| Name   | Type     | Range                              | Default |
|--------|----------|------------------------------------|---------|
| <Value | Discrete | {NORMAl AVERAges PEAK HRESolution} | NORMAl  |

**Remarks**

- **NORMAL:** In this mode, the oscilloscope samples the signal at equal time intervals to reconstruct the waveform. For most waveforms, using this mode can produce the best display effect.
- **AVERAges:** In this mode, the oscilloscope averages multiple sampled waveforms to reduce random noise on the input signal and improve vertical resolution. The average number of times can be set by the: ACQuire: AVERAges command. The higher the average frequency, the smaller the noise and the higher the vertical resolution, but the displayed waveform also responds slower to waveform changes.
- **PEAK (peak detection):** In this mode, the oscilloscope collects the maximum and minimum values of the sampling interval signal to obtain the envelope of the signal or narrow pulses that may be lost. Using this mode can avoid signal aliasing, but the displayed noise is relatively high.
- **HRESolution (High Resolution):** This mode uses a supersampling technique to average adjacent points of the sampled waveform, reducing random noise on the input signal and producing a smoother waveform on the screen. Usually used when the sampling rate of a digital converter is higher than the storage rate of the acquisition memory.

**Return Format**

The query returns NORM, AVERAge, PEAK, or HRESolution.

**Example**

```
:ACQuire:TYPE AVERAges          /* Set the acquisition method to average */
:ACQuire:TYPE?                  /* Query returns AVERAge */
```

### 3.5.4 :ACQuire:SRATe?

**Syntax**

```
:ACQuire:SRATe?
```

**Description**

Query the current sampling rate, with a default unit of Sa/s.

**Parameter**

N/A

**Remarks**

- The sampling rate refers to the frequency at which the oscilloscope samples the signal, which is the number of waveform points sampled per second.
- The relationship between sampling rate, storage depth, and waveform length satisfies the following equation:  
Storage depth=sampling rate × Waveform length.

**Return Format**

The query returns the sampling rate in real form.

**Example**

```
:ACquire:SRate?          /* Query returned 1.250e+06*/
```

## 3.6 :CALibration commands

---

### 3.6.1 :CALibrate:QUIT

---

**Syntax**

```
:CALibrate:QUIT
```

**Description**

Abandon self calibration operation at any time.

**Parameter**

N/A

**Remarks**

N/A

**Return Format**

N/A

**Example**

```
:CALibrate:QUIT          /* Exit self calibration */
```

### 3.6.2 :CALibrate:START

---

**Syntax**

```
:CALibrate:START
```

**Description**

The oscilloscope begins to perform self calibration operations.

#### Parameter

N/A

#### Remarks

- The self calibration operation can quickly make the oscilloscope reach the optimal working state to obtain the most accurate measurement values.
- Before performing self calibration, please ensure that all channels are not connected to signals until the self calibration operation is completed.
- During the self calibration process, most of the button functions have been disabled.

#### Return Format

Returns the current calibration status.

#### Example

```
:CALibrate:START          /* Oscilloscope calibration enabled */
```

## 3.7 :CHANnel<n> Commands

The :CHANnel<n> commands are used to set or query the bandwidth limit, coupling, vertical scale, vertical offset, and other vertical system parameters of the analog channel.

### 3.7.1 :CHANnel<n>:BWLimit

#### Syntax

```
:CHANnel<n>:BWLimit <type>
```

```
:CHANnel<n>:BWLimit?
```

#### Description

Set or query the bandwidth limit parameters for the specified channel.

#### Parameter

| Name   | Type     | Range                    | Default |
|--------|----------|--------------------------|---------|
| <n>    | Discrete | {0 1 2 3 4}              | —       |
| <type> | Discrete | {OFF 20M 100M 200M 350M} | 20M     |

#### Remarks

- OFF: Turn off bandwidth limitation, and the high-frequency components contained in the measured signal can pass through.

**Return Format**

Query returns OFF, 20M, 100M, 200M, 350M.

**Example**

```
:CHANnel1:BWLimit 100M          /* Turn on 100MHz bandwidth limit*/
:CHANnel1:BWLimit?              /*Query return 100M*/
```

**3.7.2 :CHANnel<n>:COUPling****Syntax**

```
:CHANnel<n>:COUPling<coupling>
:CHANnel<n>:COUPling?
```

**Description**

Sets or queries the coupling mode of the specified channel.

**Parameter**

| Name       | Type     | Range       | Default |
|------------|----------|-------------|---------|
| <n>        | Discrete | {1 2 3 4}   | —       |
| <coupling> | Discrete | {AC DC GND} | DC      |

**Remarks**

AC: the DC components of the signal under test are blocked.

DC: both DC and AC components of the signal under test can pass through the channel.

GND: both DC and AC components of the signal under test are blocked.

**Return Format**

The query returns AC, DC, or GND.

**Example**

```
:CHANnel1:COUPling AC          /*Selects the AC coupling mode.*/
:CHANnel1:COUPling?            /*The query returns AC.*/
```

**3.7.3 :CHANnel<n>: DISPlay****Syntax**

```
:CHANnel<n>:DISPlay <bool>
:CHANnel<n>:DISPlay?
```

**Description**

Enables or disables the specified channel; or queries the on/off status of the specified channel.

**Parameter**

| Name   | Type     | Range            | Default                   |
|--------|----------|------------------|---------------------------|
| <n>    | Discrete | {1 2 3 4}        | —                         |
| <bool> | Bool     | {{1 ON}}{0 OFF}} | CH1:1 ON<br>CH2-CH4:0 OFF |

**Remarks**

N/A

**Return Format**

The query returns 1 or 0.

**Example**

```
:CHANnel1:DISPlay ON          /*Enables CH1.*/
:CHANnel1:DISPlay?           /*The query returns 1.*/
```

### 3.7.4 :CHANnel<n>: INVert

**Syntax**

```
:CHANnel<n>:INVert <bool>
:CHANnel<n>:INVert?
```

**Description**

Turns on or off the waveform invert for the specified channel; or queries the on/off status of the waveform invert for the specified channel.

**Parameter**

| Name   | Type     | Range            | Default |
|--------|----------|------------------|---------|
| <n>    | Discrete | {1 2 3 4}        | --      |
| <bool> | Bool     | {{1 ON}}{0 OFF}} | 0 OFF   |

**Remarks**

When the waveform invert is turned off, the waveform is displayed normally; when the waveform invert is turned on, the voltage values of the displayed waveform are inverted.

**Return Format**

The query returns 1 or 0.

#### Example

```
:CHANnel1:INVert ON      /*Enables the waveform invert for CH1.*/
:CHANnel1:INVert?       /*The query returns 1.*/
```

### 3.7.5 :CHANnel<n>: OFFSet

#### Syntax

```
:CHANnel<n>:OFFSet <offset>
:CHANnel<n>:OFFSet?
```

#### Description

Sets or queries the vertical offset of the specified channel. The default unit is V.

#### Parameter

| Name     | Type     | Range                                                                                                                                                                                                        | Default             |
|----------|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|
| <n>      | Discrete | {1 2 3 4}                                                                                                                                                                                                    | --                  |
| <offset> | Real     | It is related to the vertical gear and probe ratio.<br>When the probe ratio is 1X,<br>Vertical gear >1.28V/div: -50V to +50V<br>Vertical gear<1.28V/div: -10V to +10V<br>Vertical gear<128mV/div: -1V to +1V | 0V (probe ratio 1X) |

#### Remarks

The vertical displacement value set is influenced by the vertical gear and probe ratio. The range of legal values varies with the set vertical gear and probe ratio values. If you set a value that is offset beyond the legal value range, the offset value will automatically be set to the closest legal value.

#### Return Format

The query returns the vertical offset in scientific notation.

#### Example

```
:CHANnel1:OFFSet 0.01      /* Set the vertical offset of CH1 to 10mV. */
:CHANnel1:OFFSet?         /* Query returns 1.000e-02*/
```

### 3.7.6 :CHANnel<n>: SCALE

#### Syntax

:CHANnel<n>:SCALE <scale>

:CHANnel<n>:SCALE?

#### Description

Set or query the vertical gear of the specified channel, with a default unit of V.

#### Parameter

| Name    | Type     | Range                                                     | Default                    |
|---------|----------|-----------------------------------------------------------|----------------------------|
| <n>     | Discrete | {1 2 3 4}                                                 | —                          |
| <scale> | Real     | Related to probe ratio<br>Probe ratio is 1X: 500uV to 10V | 1V (probe ratio<br>of 10X) |

#### Remarks

- The adjustable range of the vertical gear is related to the current probe ratio (set by the: CHANnel<n>: PROBe command).
- You can use the: CHANnel<n>: VERNier command to turn on or off the fine-tuning settings for the specified channel's vertical gear. The fine-tuning setting is turned off by default. At this time, you can only set the vertical gear in steps of 1-2 to 5, namely 500uV, 1mV, 2mV, 5mV, 10mV.. 10V (probe ratio is 1X). When fine-tuning settings are turned on, you can further adjust the vertical gear within a smaller range to improve vertical resolution. If the amplitude of the input waveform is slightly greater than the full scale in the current gear, and the amplitude displayed in the next gear waveform is slightly lower, fine tuning can be used to improve the waveform display amplitude, which is conducive to observing signal details.

#### Return Format

Query returns the vertical gear value in scientific counting form.

#### Example

```
:CHANnel1:SCALE 1          /* Set the vertical gear of CH1 to 1V */
:CHANnel1:SCALE?           /* Query returns 1.000e+00 */
```

### 3.7.7 :CHANnel<n>: VERNier

#### Syntax

:CHANnel<n>:VERNier <bool>

:CHANnel<n>:VERNier?

### Description

Turn on or off the fine tuning function for the specified channel vertical gear, or query the status of the fine tuning function for the specified channel vertical gear.

### Parameter

| Name   | Type     | Range            | Default |
|--------|----------|------------------|---------|
| <n>    | Discrete | {1 2 3 4}        | —       |
| <bool> | Bool     | {{1 ON}}{0 OFF}} | 0 OFF   |

### Remarks

The fine-tuning setting is turned off by default. At this time, you can only set the vertical gear in steps of 1-2 to 5, namely 500u, 1mV, 2mV, 5mV, 10mV... 10V (probe ratio is 1X). When fine-tuning settings are turned on, you can further adjust the vertical gear within a smaller range to improve vertical resolution. If the amplitude of the input waveform is slightly greater than the full scale in the current gear, and the amplitude displayed in the next gear waveform is slightly lower, fine tuning can be used to improve the waveform display amplitude, which is conducive to observing signal details.

### Return Format

Query returns 1 or 0.

### Example

```
:CHANnel1:VERNier ON      /* Turn on the fine adjustment function of CH1 vertical
gear */
:CHANnel1:VERNier?        /* Query returns 1*/
```

## 3.7.8 :CHANnel<n>:PROBe

### Syntax

```
:CHANnel<n>:PROBe <atten>
:CHANnel<n>:PROBe?
```

### Description

Set or query the probe ratio for the specified channel.

### Parameter

| Name    | Type     | Range           | Default |
|---------|----------|-----------------|---------|
| <n>     | Discrete | {1 2 3 4}       | —       |
| <atten> | Discrete | {1 10 100 1000} | 1       |

### Remarks

- Set the probe ratio to display the collected signal multiplied by a specified multiple (without affecting the actual amplitude of the signal).
- Set the probe ratio to affect the adjustable range of the current vertical gear.

#### Return Format

Query returns the probe attenuation ratio in scientific counting form.

#### Example

```
:CHANnel1:PROBe 10      /* Set the attenuation ratio of CH1 probe to 10X */
:CHANnel1:PROBe?       /* Query returns 1.000000 e+01*/
```

### 3.7.9 :CHANnel<n>:UNITs

#### Syntax

```
:CHANnel<n>:UNITs <atten>
:CHANnel<n>:UNITs?
```

#### Description

Set or query the display unit for a specified channel.

#### Parameter

| Name    | Type     | Range                | Default |
|---------|----------|----------------------|---------|
| <n>     | Discrete | {1 2 3 4}            | —       |
| <atten> | Discrete | {WATT AMPere VOLT U} | VOLT    |

#### Remarks

Set the units displayed for the current channel.

#### Return Format

Query the units displayed on the current channel.

#### Example

```
:CHANnel1:UNITs VOLT    /* Set the amplitude display unit of CH1 to V */
:CHANnel1:UNITs?       /* Query returns VOLT */
```

## 3.8 :CURSor Commands

The CURSOR command is used to measure the X-axis value (such as time) and Y-axis value (such as voltage) of the screen waveform.

### 3.8.1 :CURSor:MODE

#### Syntax

:CURSor:MODE <mode>

:CURSor:MODE?

#### Description

Set or query the mode of cursor measurement.

#### Parameter

| Name   | Type     | Range              | Default |
|--------|----------|--------------------|---------|
| <mode> | Discrete | {OFF MANual TRACk} | OFF     |

#### Remarks

- OFF: Turn off the cursor measurement function.
- MANual: Turn on the manual cursor measurement mode.
- TRACk: Turn on the cursor tracking measurement mode.

#### Return Format

Query returns OFF, MAN, TRAC.

#### Example

:CURSor:MODE MANual /\* Select manual cursor measurement mode \*/

:CURSor:MODE? /\* Query returns MAN \*/

### 3.8.2 :CURSor:MANual:TYPE

#### Syntax

:CURSor:MANual:TYPE <type>

:CURSor:MANual:TYPE?

#### Description

Set or query the cursor type for manual cursor measurement.

#### Parameter

| Name   | Type     | Range | Default |
|--------|----------|-------|---------|
| <type> | Discrete | {X Y} | X       |

#### Remarks

- X: Select the X-shaped cursor. The X-shaped cursor consists of a vertical solid line (cursor A) and a vertical dashed line (cursor B), commonly used to measure time parameters.
- Y: Select the Y-shaped cursor. The Y-shaped cursor consists of a horizontal solid line (cursor A) and a horizontal dashed line (cursor B), commonly used to measure voltage parameters.

#### Return Format

Query returns X or Y.

#### Example

```
:CURSor:MANual:TYPE X      /* Select X-shaped cursor */
:CURSor:MANual:TYPE?      /* Query returns X */
```

### 3.8.3 :CURSor:MANual:SOURce

#### Syntax

```
:CURSor:MANual:SOURce <source>
:CURSor:MANual:SOURce?
```

#### Description

Set or query the channel source for manual cursor measurement.

#### Parameter

| Name     | Type     | Range                                          | Default  |
|----------|----------|------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH LA } | CHANnel1 |

#### Remarks

- Only the currently open channel can be selected as the channel source.
- When selecting LA, the cursor Type cannot be set to Y (CURSOR: MANual: TYPE).

#### Return Format

Query returns CHAN1, CHAN2, CHAN3, CHAN4, MATH, or LA.

#### Example

```
:CURSor:MANual:SOURce CHANnel2 /* Set channel source to CH2*/
:CURSor:MANual:SOURce?         /* Query returns CHAN2*/
```

### 3.8.4 :CURSor:MANual:AX

#### Syntax

:CURSor:MANual:AX <x>

:CURSor:MANual:AX?

#### Description

Set or query the horizontal position of cursor A during manual cursor measurement.

#### Parameter

| Name | Type    | Range  | Default |
|------|---------|--------|---------|
| <x>  | Integer | 0-1000 | 400     |

#### Remarks

The horizontal and vertical positions of the cursor are defined by the pixel coordinates of the screen. The pixel coordinate range of the screen is (0, 0) to (1000480). Among them, (0, 0) is the top left corner of the screen, and (1000480) is the bottom right corner of the screen. The horizontal pixel range is 0 to 1000, and the vertical pixel range is 0 to 480.

#### Return Format

Query returns an integer between 0 and 1000.

#### Example

:CURSor:MANual:AX 200            /\* Set the horizontal position of cursor A to 200\*/

:CURSor:MANual:AX?            /\* Query returns 200\*/

### 3.8.5 :CURSor:MANual:BX

#### Syntax

:CURSor:MANual:BX <x>

:CURSor:MANual:BX?

#### Description

Set or query the horizontal position of cursor B during manual cursor measurement.

#### Parameter

| Name | Type    | Range  | Default |
|------|---------|--------|---------|
| <x>  | Integer | 0-1000 | 600     |

**Remarks**

The horizontal and vertical positions of the cursor are defined by the pixel coordinates of the screen. The pixel coordinate range of the screen is (0, 0) to (1000480). Among them, (0, 0) is the top left corner of the screen, and (1000480) is the bottom right corner of the screen. The horizontal pixel range is 0 to 1000, and the vertical pixel range is 0 to 480.

**Return Format**

Query returns an integer between 0 and 1000.

**Example**

```
:CURSor:MANual:BX 200      /* Set the horizontal position of cursor B to 200*/
:CURSor:MANual:BX?         /* Query returns 200*/
```

### 3.8.6 :CURSor:MANual:AY

**Syntax**

```
:CURSor:MANual:AY <y>
:CURSor:MANual:AY?
```

**Description**

Set or query the vertical position of cursor A during manual cursor measurement.

**Parameter**

| Name | Type    | Range | Default |
|------|---------|-------|---------|
| <y>  | Integer | 0-480 | 100     |

**Remarks**

- The horizontal and vertical positions of the cursor are defined by the pixel coordinates of the screen. The pixel coordinates of the screen range from (0, 0) to (1000480). Among them, (0, 0) is the top left corner of the screen, and (1000480) is the bottom right corner of the screen. The horizontal pixel range is 0 to 1000, and the vertical pixel range is 0 to 480.
- When selecting LA as the signal source for manual cursor measurement, there is no need to use a Y-shaped cursor.

**Return Format**

The query returns an integer between 0 and 480.

**Example**

```
:CURSor:MANual:AY 200      /* Set the vertical position of cursor A to 200*/
```

```
:CURSor:MANual:AY?          /* Query returns 200*/
```

### 3.8.7 :CURSor:MANual:BY

#### Syntax

```
:CURSor:MANual:BY <y>
```

```
:CURSor:MANual:BY?
```

#### Description

When setting or querying manual cursor measurements, the vertical position of cursor B.

#### Parameter

| Name | Type    | Range | Default |
|------|---------|-------|---------|
| <y>  | Integer | 0-480 | 300     |

#### Remarks

- The horizontal and vertical positions of the cursor are defined by the pixel coordinates of the screen. The pixel coordinates of the screen range from (0, 0) to (1000480). Among them, (0, 0) is the top left corner of the screen, and (1000480) is the bottom right corner of the screen. The horizontal pixel range is 0 to 1000, and the vertical pixel range is 0 to 480.
- When selecting LA as the signal source for manual cursor measurement, there is no need to use a Y-shaped cursor.

#### Return Format

The query returns an integer between 0 and 480.

#### Example

```
:CURSor:MANual:BY 200      /* Set the vertical position of cursor B to 200*/
```

```
:CURSor:MANual:BY?        /* Query returns 200*/
```

### 3.8.8 :CURSor:MANual:AXValue?

#### Syntax

```
:CURSor:MANual:AXValue?
```

#### Description

When querying manual cursor measurements, the X value at cursor A. The unit is determined by the currently selected horizontal unit.

**Parameter**

N/A

**Remarks**

N/A

**Return Format**

Query returns the X value at the current cursor A in scientific counting form.

**Example**

```
:CURSor:MANual:AXValue?          /* Query returned -4.000000 e-06*/
```

### 3.8.9 :CURSor:MANual:AYValue?

---

**Syntax**

```
:CURSor:MANual:AYValue?
```

**Description**

When querying manual cursor measurements, the Y value at cursor A. The unit is determined by the currently selected vertical unit.

**Parameter**

N/A

**Remarks**

N/A

**Return Format**

- When selecting CHANnel1 | CHANnel2 | CHANnel3 | CHANnel4 | MATH as the source, the query returns the Y value at cursor A in scientific count form.
- When selecting LA as the source, the query returns the decimal value corresponding to the binary weighted sum of D15 to D0 bits at the current cursor A in integer form (unopened channels default to 0).

**Example**

```
:CURSor:MANual:AYValue?          /* Query returns 2.000000 e+00*/
```

### 3.8.10 :CURSor:MANual:BXValue?

---

**Syntax**

:CURSor:MANual:BXValue?

**Description**

When querying manual cursor measurements, the X value at cursor B. The unit is determined by the currently selected horizontal unit.

**Parameter**

N/A

**Remarks**

N/A

**Return Format**

Query returns the X value at the current cursor B in scientific counting form.

**Example**

:CURSor:MANual:BXValue? /\* Query returns 4.000000e-06\*/

### 3.8.11 :CURSor:MANual:BYValue?

---

**Syntax**

:CURSor:MANual:BYValue?

**Description**

When querying manual cursor measurements, the Y value at cursor B. The unit is determined by the currently selected vertical unit.

**Parameter**

N/A

**Remarks**

N/A

**Return Format**

- When selecting CHANnel1 | CHANnel2 | CHANnel3 | CHANnel4 | MATH as the source, the query returns the Y value at cursor B in scientific count form.
- When selecting LA as the source, the query returns the decimal value corresponding to the binary weighted sum of D15 to D0 bits at the current cursor B in integer form (unopened channels default to 0).

**Example**

:CURSor:MANual:BYValue? /\* Query returned -2.000000 e+00\*/

### 3.8.12 :CURSor:MANual:XDELta?

---

#### Syntax

:CURSor:MANual:XDELta?

#### Description

When querying manual cursor measurements, the difference BX-AX between the X values at cursor A and cursor B. The unit is determined by the currently selected horizontal unit.

#### Parameter

N/A

#### Remarks

N/A

#### Return Format

Query returns the current difference in scientific count form.

#### Example

:CURSor:MANual:XDELta? /\* Query returned 8.000000e-06\*/

### 3.8.13 :CURSor:MANual:IXDELta?

---

#### Syntax

:CURSor:MANual:IXDELta?

#### Description

When querying manual cursor measurements, the reciprocal of the absolute difference between the X values at cursor A and cursor B is  $1/|dX|$ . The unit is determined by the currently selected horizontal unit.

#### Parameter

N/A

#### Remarks

N/A

#### Return Format

Query returns 1/| dX | in scientific count form|.

#### Example

```
:CURSor:MANual:IXDELta?          /* Query returned 1.250000e+05*/
```

### 3.8.14 :CURSor:MANual:YDELta?

#### Syntax

```
:CURSor:MANual:YDELta?
```

#### Description

When querying manual cursor measurements, the difference between the Y values at cursor A and cursor B is BY-AY. The unit is determined by the currently selected vertical unit.

#### Parameter

N/A

#### Remarks

N/A

#### Return Format

When selecting CHANnel1 | CHANnel2 | CHANnel3 | CHANnel4 | MATH as the source, the query returns the current difference in scientific count form; When selecting LA as the source, the integer 4294967295 is returned.

#### Example

```
:CURSor:MANual:YDELta?          /* Query returned -4.000000 e+00*/
```

### 3.8.15 :CURSor:TRACK:SOURce1

#### Syntax

```
:CURSor:TRACK:SOURce1 <source>
:CURSor:TRACK:SOURce1?
```

#### Description

When setting or querying cursor tracking measurements, the channel source for cursor A measurement.

#### Parameter

| Name     | Type     | Range                                      | Default  |
|----------|----------|--------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH} | CHANnel1 |

**Remarks**

Only open channels can be selected as channel sources.

**Return Format**

Query and return CHAN1, CHAN2, CHAN3, CHAN4, or MATH.

**Example**

```
:CURSor:TRACk:SOURce1 CHANnel2    /* Set channel source to CH2*/
:CURSor:TRACk:SOURce1?              /* Query returns CHAN2*/
```

### 3.8.16 :CURSor:TRACk:AX

**Syntax**

```
:CURSor:TRACk:AX <x>
:CURSor:TRACk:AX?
```

**Description**

Set or query the horizontal position of cursor A when tracking measurements.

**Parameter**

| Name | Type    | Range  | Default |
|------|---------|--------|---------|
| <x>  | Integer | 0-1000 | 400     |

**Remarks**

The horizontal and vertical positions of the cursor are defined by the pixel coordinates of the screen. The pixel coordinate range of the screen is (0, 0) to (1000, 480). Among them, (0, 0) is the top left corner of the screen, and (1000, 480) is the bottom right corner of the screen. The horizontal pixel range is 0 to 1000, and the vertical pixel range is 0 to 480.

**Return Format**

Query returns an integer between 0 and 1000.

**Example**

```
:CURSor:TRACk:AX 200    /* Set the horizontal position of cursor A to 200*/
:CURSor:TRACk:AX?       /* Query returns 200*/
```

### 3.8.17 :CURSor:TRACk:BX

#### Syntax

:CURSor:TRACk:BX <x>

:CURSor:TRACk:BX?

#### Description

Set or query the horizontal position of cursor B when tracking cursor measurements.

#### Parameter

| Name | Type    | Range  | Default |
|------|---------|--------|---------|
| <x>  | Integer | 0-1000 | 600     |

#### Remarks

The horizontal and vertical positions of the cursor are defined by the pixel coordinates of the screen. The pixel coordinate range of the screen is (0, 0) to (1000480). Among them, (0, 0) is the top left corner of the screen, and (1000480) is the bottom right corner of the screen. The horizontal pixel range is 0 to 1000, and the vertical pixel range is 0 to 480.

#### Return Format

Query returns an integer between 0 and 1000.

#### Example

:CURSor:TRACk:BX 200           /\* Set the horizontal position of cursor B to 200\*/

:CURSor:TRACk:BX?           /\* Query returns 200\*/

## 3.9 :MEASure Commands

### 3.9.1 :MEASure:SOURce

#### Syntax

:MEASure:SOURce <sour>

:MEASure:SOURce?

#### Description

Set or query the source of information for the current measurement parameter.

#### Parameter

| Name   | Type     | Range                                                                                            | Default  |
|--------|----------|--------------------------------------------------------------------------------------------------|----------|
| <sour> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

**Remarks**

Only the currently open channels are optional.

**Return Format**

Query and return CHAN1, CHAN2, CHAN3, CHAN4, MATH, D0, D1, D2, D3, D4, D5, D6, D7, D8, D9, D10, D11, D12, D13, D14, D15.

**Example**

```
:MEASure:SOURce CHANnel2      /* Set the Parameter measurement source to CH2*/
```

```
:MEASure:SOURce?              /* Query returns CHAN2*/
```

### 3.9.2 :MEASure:CLEar

**Syntax**

```
:MEASure:CLEar <item>
```

**Description**

Clear any or all of the last 5 open measurement items.

**Parameter**

| Name   | Type     | Range                               | Default |
|--------|----------|-------------------------------------|---------|
| <item> | Discrete | {ITEM1 ITEM2 ITEM3 ITEM4 ITEM5 ALL} | ——      |

**Remarks**

The Measure: ITEM command can be used to open the parameters to be measured among the 36 parameters. The last 5 parameters are determined by the order in which you open them and will not change because you delete one or more measurement items.

**Return Format**

N/A

**Example**

```
:MEASure:CLEar ITEM1          /* Clear measurement item ITEM1*/
```

### 3.9.3 :MEASure:AMSource

#### Syntax

```
:MEASure:AMSource <src>
```

```
:MEASure:AMSource?
```

#### Description

Set or query the source of information for all measurement functions.

#### Parameter

| Name  | Type     | Range                                     | Default |
|-------|----------|-------------------------------------------|---------|
| <src> | Discrete | {OFF CHANnel1 CHANnel2 CHANnel3 CHANnel4} | OFF     |

#### Remarks

N/A

#### Return Format

Query returns OFF, CHAN1, CHAN2, CHAN3, and CHAN4.

#### Example

```
:MEASure:AMSource CHANnel1      /* Set the signal source to CH1 */
```

```
:MEASure:AMSource?              /* Query returns CHAN1*/
```

### 3.9.4 :MEASure:STATistic:DISPlay

#### Syntax

```
:MEASure:STATistic:DISPlay <bool>
```

```
:MEASure:STATistic:DISPlay?
```

#### Description

Turn on or off the statistical function, or check the status of the statistical function.

#### Parameter

| Name   | Type | Range            | Default |
|--------|------|------------------|---------|
| <bool> | Bool | {{1 ON}}{0 OFF}} | 0 OFF   |

#### Remarks

When the statistics function is turned on, the oscilloscope counts and displays the statistical results of up to 5 measurement parameters that were last opened.

**Return Format**

Query returns 1 or 0.

**Example**

```
:MEASure:STATistic:DISPlay ON      /* Turn on statistics function */
:MEASure:STATistic:DISPlay?        /* Query returns 1*/
```

**3.9.5 :MEASure:ITEM****Syntax**

```
:MEASure:ITEM <item>[, <src>[, <src>]]
:MEASure:ITEM? <item>[, <src>[, <src>]]
```

**Description**

Measure any waveform parameter of the specified signal source, or query the measurement results of any waveform parameter of the specified signal source.

**Parameter**

| Name   | Type                    | Range                                                                                                                                                                                                                                                           | Default |
|--------|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| <item> | Discrete                | { FREQuency PERiod VAVG VMAX VMIN VPP <br>VTOP VMID  VBASE VAMP VRMS OVERshoot <br>PREShoot PERIOD_RMS  PERIOD_MEAN RTI<br>Me FTIME PWIDth NWIDth PDUTy  NDUTy FRR<br> FFF BWIDTh FRF FFR LRR LRF LFR LFF <br> MAX_TIME MIN_TIME PHASE_R PHASE_F V<br>ARIANCE } | ---     |
| <src>  | Please refer to Remarks |                                                                                                                                                                                                                                                                 |         |

**Remarks**

Parameter [, <src>[, <src>]] is used to set the signal source of the tested parameter.

If the value of <item> is PERiod, FREQuency, PWIDth, NWIDth, PDUTy, NDUTy, the range of <src> is: {D0 | D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8 | D9 | D10 | D11 | D12 | D13 | D14 | D15 | CHANnel1 | CHANnel2 | CHANnel3 | CHANnel4 | MATH}.

If the value of <item> is other measurement parameters, the range of <src> is:

{CHANnel1 | CHANnel2 | CHANnel3 | CHANnel4 | MATH}

If the measurement parameters are single source (VMAX, VMIN, VPP, VTOP, VBASE, VAMP, VAVG, VRMS, Overshot, PRESshot, MARea, MPAREa, PERiod, FREQuency, RTIME, FTIME)

PWIDth, NWIDth, PDUTy, NDUTy, TVMAX, TVMIN, PSLEWrate, NSLEWrate,

VUpper, VMID, VLOWer, VARIANCE, PVRMS, PPULses, NPULses, PEDGs,

NEDGes), then only one signal source needs to be set. If this parameter is omitted, it

defaults to the signal source selected by the Measure: Source command.

If the measurement parameter is two sources (RDElay, FDElay, RPHase, FPHase), the input command must include two sources, otherwise the command is invalid.

If this parameter is omitted, it defaults to the signal sources selected by the: MEASURE: SETUp: DSA and: MEASURE: SETUp: DSB, : MEASURE: SETUp: PSA, and: MEASURE: SETUp: PSB commands.

#### Return Format

Query returns the current measurement value in scientific counting form.

#### Example

```
:MEASure:ITEM OVERshoot, CHANnel2      /* Open the overshoot measurement
for channel 2 */
:MEASure:ITEM? OVERshoot, CHANnel2      /* Query returned 8.888889e-03*/
```

### 3.9.6 :MEASure:FREQuency

#### Syntax

```
:MEASure:FREQuency <source>
:MEASure:FREQuency?<source>
```

#### Description

Set or query the current measurement item frequency.

#### Parameter

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

#### Remarks

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

#### Return Format

Query to return the current value in Hz.

#### Example

```
:MEASure:FREQuency CHANnel1      /* Add channel 1 measurement item
frequency */
:MEASure:FREQuency? CHANnel1      /* Query and return the current measurement
```

value of 1.000000 e+03 (1kHz)\* /

### 3.9.7 :MEASure:PERiod

#### Syntax

:MEASure:PERiod <source>

:MEASure:PERiod?<source>

#### Description

Set or query the current measurement item cycle.

#### Parameter

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

#### Remarks

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

#### Return Format

Query returns the current value. The unit is s.

#### Example

```
:MEASure:PERiod CHANnel1      /* Add measurement item cycle for channel 1 */
:MEASure:PERiod? CHANnel1     /* Query and return the current measurement
value of 2.000000 e-06 (2us)* /
```

### 3.9.8 :MEASure:RTIME

#### Syntax

:MEASure:RTIME <source>

:MEASure:RTIME?<source>

#### Description

Set or query the rise time of the current measurement item.

#### Parameter

| Name     | Type     | Range                            | Default  |
|----------|----------|----------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHAN | CHANnel1 |

| Name | Type | Range                                                            | Default |
|------|------|------------------------------------------------------------------|---------|
|      |      | nel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} |         |

**Remarks**

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

**Return Format**

Query returns the current value. Unit s.

**Example**

```
:MEASure:RTIME CHANnel1      /* Newly added channel 1 measurement item rise
time */
:MEASure:RTIME? CHANnel1      /* Query and return the current measurement
value of 6.000000 e-07 (600ns)*/
```

### 3.9.9 :MEASure:FTIME

**Syntax**

```
:MEASure:FTIME <source>
:MEASure:FTIME? <source>
```

**Description**

Set or query the descent time of the current measurement item.

**Parameter**

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

**Remarks**

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

**Return Format**

Query returns the current value. Unit s.

**Example**

```
:MEASure:FTIME CHANnel1      /* New channel 1 measurement item descent time
```

added \*/  
:MEASure:FTIME? CHANnel1 /\* Query and return the current measurement  
value of 6.000000 e-07s (600ns) \*/

### 3.9.10 :MEASure:PWIDth

#### Syntax

:MEASure:PWIDth <source>  
:MEASure:PWIDth? <source>

#### Description

Set or query the positive pulse width of the current measurement item.

#### Parameter

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

#### Remarks

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

#### Return Format

Query returns the current value. Unit s.

#### Example

:MEASure:PWIDth CHANnel1 /\* Add channel 1 measurement item positive  
pulse width \*/  
:MEASure:PWIDth? CHANnel1 /\* Query and return the current measurement  
value of 1.000000 e-06 (1us) \*/

### 3.9.11 :MEASure:NWIDth

#### Syntax

:MEASure:NWIDth <source>  
:MEASure:NWIDth? <source>

#### Description

Set or query the negative pulse width of the current measurement item.

**Parameter**

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

**Remarks**

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

**Return Format**

Query returns the current value in seconds.

**Example**

```
:MEASure:NWIDth CHANnel1      /* New channel 1 measurement item negative
pulse width added */
:MEASure:NWIDth? CHANnel1      /* Query and return the current measurement
value of 1.000000 e-06 (1us) */
```

### 3.9.12 :MEASure:PDUTy

**Syntax**

```
:MEASure:PDUTy <source>
:MEASure:PDUTy? <source>
```

**Description**

Set or query the positive duty cycle of the current measurement item.

**Parameter**

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

**Remarks**

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

**Return Format**

Query returns the current value percentage.

**Example**

```
:MEASure:PDUTy CHANnel1      /* New channel 1 measurement item positive
duty cycle */
:MEASure:PDUTy? CHANnel1      /* Query and return the current measurement
value of 5.0000000e-01 */
```

**3.9.13 :MEASure:NDUTy****Syntax**

```
:MEASure:NDUTy <source>
:MEASure:NDUTy? <source>
```

**Description**

Set or query the negative duty cycle of the current measurement item.

**Parameter**

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

**Remarks**

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

**Return Format**

Query returns the current value percentage.

**Example**

```
:MEASure:NDUTy CHANnel1      /* New channel 1 measurement item negative
duty cycle added */
:MEASure:NDUTy? CHANnel1      /* Query and return the current measurement
value of 5.0000000e-01*/
```

**3.9.14 :MEASure:BWIDth****Syntax**

```
:MEASure:BWIDth <source>
:MEASure:BWIDth? <source>
```

**Description**

Set or query the bandwidth of the current measurement item.

**Parameter**

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

**Remarks**

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

**Return Format**

Query returns the current value in seconds.

**Example**

```
:MEASure:BWIDth CHANnel1      /* Add channel 1 measurement item bandwidth */
:MEASure:BWIDth? CHANnel1      /* Query and return the current measurement
value of 1.000000 e-06 (1us) */
```

### 3.9.15 :MEASure:MAXTIME

**Syntax**

```
:MEASure:MAXTIME <source>
:MEASure:MAXTIME? <source>
```

**Description**

Set or query the maximum value time of the current measurement item.

**Parameter**

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

**Remarks**

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

**Return Format**

Query returns the current value in seconds.

#### Example

```
:MEASure:MAXTime CHANnel1      /* New channel 1 measurement item maximum
value time */
:MEASure:MAXTime? CHANnel1     /* Query and return the current measurement
value of 5.000000e-06 (500ns)*/
```

### 3.9.16 :MEASure:MINTIME

#### Syntax

```
:MEASure:MINTIME <source>
:MEASure:MINTIME? <source>
```

#### Description

Set or query the minimum value time of the current measurement item.

#### Parameter

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

#### Remarks

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

#### Return Format

Query returns the current value in seconds.

#### Example

```
:MEASure:MINTIME CHANnel1      /* New channel 1 measurement item minimum
time */
:MEASure:MINTIME? CHANnel1     /* Query and return the current measurement
value of 5.000000e-06 (500ns) */
```

### 3.9.17 :MEASure:VMEAn

#### Syntax

```
:MEASure:VMEAn <source>
:MEASure:VMEAn? <source>
```

**Description**

Set or query the average value of the current measurement item.

**Parameter**

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

**Remarks**

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

**Return Format**

Query to return the current value in units of v.

**Example**

:MEASure:VMEAn CHANnel1 /\* Added channel 1 measurement item average value \*/

:MEASure:VMEAn? CHANnel1 /\* Query and return the current measurement value of 8.760000 e-02 (87.6 mv)\*/

### 3.9.18 :MEASure:VMAX

**Syntax**

:MEASure:VMAX <source>

:MEASure:VMAX? <source>

**Description**

Set or query the maximum value of the current measurement item.

**Parameter**

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

**Remarks**

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

**Return Format**

Query to return the current value in units of v.

**Example**

```
:MEASure:VMAX CHANnel1      /* New channel 1 measurement item maximum value
added */
:MEASure:VMAX? CHANnel1      /* Query and return the current measurement
value of 1.000000 e+00 (1V) */
```

**3.9.19 :MEASure:VMIN****Syntax**

```
:MEASure:VMIN <source>
:MEASure:VMIN? <source>
```

**Description**

Set or query the minimum value of the current measurement item.

**Parameter**

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

**Remarks**

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

**Return Format**

Query to return the current value in units of v.

**Example**

```
:MEASure:VMIN CHANnel1      /* Add minimum value of measurement item for
channel 1 */
:MEASure:VMIN? CHANnel1/* Query and return the current measurement value of
1.000000 e+00 (1V)*/
```

**3.9.20 :MEASure:PKPk****Syntax**

```
:MEASure:PKPk <source>
```

```
:MEASure:PKPk? <source>
```

### Description

Set or query the bimodal value of the current measurement item.

### Parameter

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

### Remarks

Set or query the bimodal value of the current measurement item.

### Return Format

Query to return the current value in units of v.

### Example

```
:MEASure:PKPk CHANnel1 /* Add channel 1 measurement item bimodal value */
```

```
:MEASure:PKPk? CHANnel1/* Query and return the current measurement value of  
2.000000 e+00 (2V)*/
```

## 3.9.21 :MEASure:VTOP

### Syntax

```
:MEASure:VTOP <source>
```

```
:MEASure:VTOP? <source>
```

### Description

Set or query the top value of the current measurement item.

### Parameter

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

### Remarks

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

**Return Format**

Query to return the current value in units of v.

**Example**

```
:MEASure:VTOP CHANnel1      /* Add top value of measurement item in channel 1 */
:MEASure:VTOP? CHANnel1      /* Query and return the current measurement
value of 1.1000000 e+00 (1.1V) */
```

**3.9.22 :MEASure:VMID****Syntax**

```
:MEASure:VMID <source>
:MEASure:VMID? <source>
```

**Description**

Set or query the intermediate value of the current measurement item.

**Parameter**

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

**Remarks**

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

**Return Format**

Query to return the current value, unit: v.

**Example**

```
:MEASure:VMID CHANnel1      /* Add intermediate value for measurement item in
channel 1 */
:MEASure:VMID? CHANnel1/* Query and return the current measurement value of
1.1000000 e+00 (1.1V) */
```

**3.9.23 :MEASure:VBASe****Syntax**

```
:MEASure:VBASe <source>
```

:MEASure:VBASe? <source>

### Description

Set or query the bottom value of the current measurement item.

### Parameter

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

### Remarks

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

### Return Format

Query to return the current value in units of v.

### Example

:MEASure:VBASe CHANnel1 /\* Add bottom value of measurement item in channel 1 \*/

:MEASure:VBASe? CHANnel1 /\* Query and return the current measurement value -1.100000e+00 (-1.1V) \*/

## 3.9.24 :MEASure:VAMp

### Syntax

:MEASure:VAMp <source>

:MEASure:VAMp? <source>

### Description

Set or query the amplitude of the current measurement item.

### Parameter

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

### Remarks

Only the currently open channels are available for selection, and no more than 7

measurement items are displayed.

#### Return Format

Query to return the current value, unit: v.

#### Example

```
:MEASure:VAMp CHANnel1      /* Add channel 1 measurement item amplitude */
:MEASure:VAMp? CHANnel1      /* Query and return the current measurement
value of 2.000000 e+00 (2V) */
```

### 3.9.25 :MEASure:VRMS

#### Syntax

```
:MEASure:VRMS <source>
:MEASure:VRMS? <source>
```

#### Description

Set or query the root mean square of the current measurement item.

#### Parameter

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

#### Remarks

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

#### Return Format

Query to return the current value in units of v.

#### Example

```
:MEASure:VRMS CHANnel1      /* New channel 1 measurement item root mean
square */
:MEASure:VRMS? CHANnel1      /* Query and return the current measurement
value of 1.000000 e+00 (1V)*/
```

### 3.9.26 :MEASure:VOVr

#### Syntax

```
:MEASure:VOVr <source>
```

```
:MEASure:VOVr? <source>
```

### Description

Set or query the current measurement item for overshoot.

### Parameter

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

### Remarks

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

### Return Format

Query returns the current value.

### Example

```
:MEASure:VOVr CHANnel1          /* Add channel 1 measurement item overshoot */
:MEASure:VOVr? CHANnel1        /* Query and return the current measurement value of
3.000000 e-02 (3%)*/
```

## 3.9.27 :MEASure:VPEr

### Syntax

```
:MEASure:VPEr <source>
```

```
:MEASure:VPEr? <source>
```

### Description

Set or query the current measurement item for pre flushing.

### Parameter

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

### Remarks

Only the currently open channels are available for selection, and no more than 7

measurement items are displayed.

#### Return Format

Query returns the current value.

#### Example

```
:MEASure:VPEr CHANnel1      /* Add channel 1 measurement item pre flushing */
:MEASure:VPEr? CHANnel1     /* Query and return the current measurement value of
3.000000 e-02 (3%) */
```

### 3.9.28 :MEASure:PVRMS

#### Syntax

```
:MEASure:PVRMS <source>
:MEASure:PVRMS? <source>
```

#### Description

Set or query the root mean square of the current measurement period.

#### Parameter

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

#### Remarks

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

#### Return Format

Query to return the current value in units of v.

#### Example

```
:MEASure:PVRMS CHANnel1      /* New channel 1 measurement item cycle root
mean square */
:MEASure:PVRMS? CHANnel1     /* Query and return the current measurement
value of 1.000000 e+00 (1V) */
```

### 3.9.29 :MEASure:PVMEAS

#### Syntax

:MEASure:PVMEAS <source>

:MEASure:PVMEAS? <source>

### Description

Set or query the average value of the current measurement period.

### Parameter

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

### Remarks

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

### Return Format

Query to return the current value in units of v.

### Example

:MEASure:PVMEAS CHANnel1 /\* Added channel 1 measurement item cycle average value \*/

:MEASure:PVMEAS? CHANnel1 /\* Query and return the current measurement value of 1.000000 e+00 (1V) \*/

## 3.9.30 :MEASure:VFOv

### Syntax

:MEASure:VFOv <source>

:MEASure:VFOv? <source>

### Description

Set or query the overshoot of the falling edge of the current measurement item.

### Parameter

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

### Remarks

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

Return Format

Query returns the current value.

Example

```
:MEASure:VFOv CHANnel1      /* Added channel 1 measurement item falling
edge overshoot */
:MEASure:VFOv? CHANnel1      /* Query and return the current measurement
value of 3.000000 e-02 (3%)*/
```

3.9.31 :MEASure:VRPr

Syntax

```
:MEASure:VRPr <source>
:MEASure:VRPr? <source>
```

Description

Set or query the pre offset of the falling edge of the current measurement item.

Parameter

| Name     | Type     | Range                                                                                            | Default  |
|----------|----------|--------------------------------------------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 MATH D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15} | CHANnel1 |

Remarks

Only the currently open channels are available for selection, and no more than 7 measurement items are displayed.

Return Format

Query returns the current value.

Example

```
:MEASure:VRPr CHANnel1      /* Newly added channel 1 measurement item falling
edge pre flushing */
:MEASure:VRPr? CHANnel1      /* Query and return the current measurement value of
3.000000 e-02 (3%)*/
```

## 3.10 :SYSTem Commands

### 3.10.1 :SYSTem:DATE

#### Syntax

:SYSTem:DATE <year>, <month>, <day>

:SYSTem:DATE?

#### Description

Set the system's year, month, and day, and query the system's year, month, and day.

#### Parameter

| Name    | Type    | Range          | Default |
|---------|---------|----------------|---------|
| <year>  | Integer | 4-digit number | 2018    |
| <month> | Integer | [1-12]         | 1       |
| <day>   | Integer | [1-31]         | 1       |

#### Remarks

The returned string contains 3 segments separated by commas: year, month, day.

#### Return Format

N/A

#### Example

The following command sets the system date to October 12, 2022

```
:SYSTem:DATE 2022, 10, 12      /* Set the date of the oscilloscope to October 12,
2022 */
```

```
:SYSTem:DATE?                  /* Query return date is October 12, 2022*/
```

### 3.10.2 :SYSTem:TIME

#### Syntax

:SYSTem:TIME <hour>, <minute>, <second>

:SYSTem:TIME?

#### Description

Set the system's hours, minutes, and seconds, and query the system's hours, minutes, and seconds.

**Parameter**

| Name     | Type    | Range  | Default |
|----------|---------|--------|---------|
| <hour>   | Integer | [0-23] | 6       |
| <minute> | Integer | [0-59] | 6       |
| <second> | Integer | [0-59] | 6       |

**Remarks**

The returned string contains 3 segments separated by commas: hour, minute, second.

**Return Format**

N/A

**Example**

The following command sets the system hours, minutes, and seconds to 6:6:6

:SYSTem:DATE 6, 6, 6 /\* The following command sets the system hours, minutes, and seconds to 6:6:6\*/

:SYSTem:DATE? /\* The query return time is 6:6:6 (6:6:6 minutes and 6 seconds)\*/

## 3.11 :TIMebase Commands

The TIMebase command is used to set the horizontal system, such as turning on delayed scanning, setting the horizontal time base mode, etc.

### 3.11.1 :TIMebase:WINDow:ENABle

**Syntax**

:TIMebase:WINDow:ENABle <bool>

:TIMebase:WINDow:ENABle?

**Description**

Turn on or off the dual window function, or check the dual window status.

**Parameter**

| Name   | Type | Range             | Default |
|--------|------|-------------------|---------|
| <bool> | Bool | {{1 ON}}{{0 OFF}} | 0 OFF   |

**Remarks**

The dual window mode can be used to horizontally zoom in on a waveform for viewing waveform details.

**Return Format**

Query returns 1 or 0.

**Example**

```
:TImebase:WINDow:ENABle ON      /* Turn on delayed scanning */
:TImebase:WINDow:ENABle?        /* Query returns 1*/
```

### 3.11.2 **:TImebase[:MAIN]:SCALE**

**Syntax**

```
:TImebase[:MAIN]:SCALE <scale>
:TImebase[:MAIN]:SCALE?
```

**Description**

Set or query the main time base gear, with a default unit of s/div.

**Parameter**

| Name    | Type    | Range                                                                                              | Default |
|---------|---------|----------------------------------------------------------------------------------------------------|---------|
| <scale> | Integer | YT mode: 5ns/div to 1ks/div in steps of 1-2-5<br>Roll mode: 100ms/div to 50s/div in steps of 1-2-5 | 1μs/div |

**Remarks**

When the horizontal time base mode is YT and the horizontal time base is 200ms/div or larger (i.e. "slow scan" mode), this command is not available during the stopping process of the oscilloscope.

**Return Format**

Query returns the main time base level in scientific counting form.

**Example**

```
:TImebase:MAIN:SCALE 0.0002      /* Set the main time base gear to 200 μ s/div */
:TImebase:MAIN:SCALE?            /* Query returned 2.000000e-04*/
```

### 3.11.3 **:TImebase[:MAIN]:OFFSet**

**Syntax**

```
:TImebase[:MAIN]:OFFSet <offset>
:TImebase[:MAIN]:OFFSet?
```

**Description**

Set or query the main time base offset, with a default unit of s.

**Parameter**

| Name     | Type    | Range                   | Default |
|----------|---------|-------------------------|---------|
| <offset> | Integer | Please refer to Remarks | 0       |

**Remarks**

The range of<offset>is related to the current horizontal time base mode (please refer to TIMEbase: MODE) and operating status of the oscilloscope.

- YT mode

RUN:  $(-0.5 \times \text{MemDepth}/\text{SampleRate})$  to 1s (when the horizontal time base is less than 200ms/div)  $(-0.5 \times \text{MemDepth}/\text{SampleRate})$  to  $(10 \times \text{MainScale})$  (when the horizontal time base is greater than or equal to 200ms/div, i.e. "slow scan" mode) STOP:  $(-\text{MemDepth}/\text{SampleRate})$  to  $(1\text{s}+0.5 \times \text{MemDepth}/\text{SampleRate})$ .

- Roll mode

RUN: This command is not available. STOP:  $(-12 \times \text{MainScale})$  to 0, where, MemDepth is the current storage depth of the oscilloscope, SampleRate is the current sampling rate of the oscilloscope, MainScale is the current main time base gear of the oscilloscope. When the horizontal time base mode is YT and the horizontal time base is 200ms/div or larger (i.e. "slow scan" mode), this command is not available during the stopping process of the oscilloscope.

**Return Format**

Query returns the main time base offset in scientific counting form.

**Example**

```
:TIMEbase:MAIN:OFFSet 0.002          /* Set the main time base offset to 2ms */
:TIMEbase:MAIN:OFFSet?                /* Query returns 2.000000 e-03*/
```

## 3.12 :TRIGger Commands

The TRIGGer command is used to set the triggering system of the oscilloscope.

### 3.12.1 :TRIGger:MODE

**Syntax**

```
:TRIGger:MODE <mode>
```

```
:TRIGger:MODE?
```

**Description**

Select or query the trigger type.

#### Parameter

| Name   | Type     | Range                                                                                    | Default |
|--------|----------|------------------------------------------------------------------------------------------|---------|
| <mode> | Discrete | <EDGE PULSe VIDeo SLOPe TIMEout <br> RUNT PATTeRn DELaY SHOLd RS232 LI<br>N CAN SPI IIC> | EDGE    |

#### Remarks

N/A

#### Return Format

返回

EDGE, PULSe, VIDeo, SLOPe, TIMEout, RUNT, PATTeRn, DELaY, SHOLd, RS232, LIN, CAN, SPI, IIC.

#### Example

```
:TRIGger:MODE SLOPe      /* Set the trigger type to slope trigger */
:TRIGger:MODE?           /* Query returns SLOPE */
```

### 3.12.2 :TRIGger:STATus?

#### Syntax

```
:TRIGger:STATus?
```

#### Description

Query the current triggering status.

#### Parameter

N/A

#### Remarks

- TRIG: The waveform data of the oscilloscope is effectively triggered.
- WAIT: Oscilloscope waiting to trigger state.
- AUTO: The oscilloscope is in automatic operation mode.
- STOP: The oscilloscope is in a stopped state.

#### Return Format

Query returns TRIG WAIT, AUTO, or STOP.

#### Example

:TRIGger:STATus? /\* Query returns AUTO \*/

### 3.12.3 :TRIGger:SWEep

#### Syntax

:TRIGger:SWEep <sweep>

:TRIGger:SWEep?

#### Description

Set or query the triggering method.

#### Parameter

| Name    | Type     | Range                | Default |
|---------|----------|----------------------|---------|
| <sweep> | Discrete | {AUTO NORMAl SINGle} | AUTO    |

#### Remarks

- AUTO: Automatically triggered, with waveform display regardless of whether the triggering conditions are met.
- NORMAL: Normal trigger, displaying the waveform when the triggering conditions are met, maintaining the original waveform display when the triggering conditions are not met, and waiting for the next trigger.
- Single: Single trigger, the oscilloscope waits for triggering, displays the waveform when the triggering conditions are met, and then stops.

#### Return Format

Query returns AUTO NORM or SING.

#### Example

:TRIGger:SWEep SINGle /\* Choose a single trigger method \*/

:TRIGger:SWEep? /\* Query returns SING \*/

### 3.12.4 :TRIGger:HOLDoff

#### Syntax

:TRIGger:HOLDoff <value>

:TRIGger:HOLDoff?

#### Description

Set or query the trigger release time, with a default unit of seconds.

#### Parameter

| Name    | Type | Range   | Default |
|---------|------|---------|---------|
| <value> | Real | 8ns-10s | 2us     |

**Remarks**

Trigger release can stably trigger complex waveforms (such as pulse series). The release time refers to the time it takes for the oscilloscope to reactivate the triggering circuit, and the oscilloscope will not trigger before the release time ends.

The triggering methods include video triggering, timeout triggering, establish hold. When UART, LIN, CAN, IIC or SPI, this setting is not available.

**Return Format**

Query to return the trigger release time in scientific counting form.

**Example**

```
:TRIGger:HOLDoff 1          /* Set the trigger release time to 1ms */
:TRIGger:HOLDoff?          /* Query returns 1.000000 e-03*/
```

### 3.12.5 :TRIGger:EDGE:SOURce

**Syntax**

```
:TRIGger:EDGE:SOURce <source>
:TRIGger:EDGE:SOURce?
```

**Description**

Set or query the trigger source for edge triggering.

**Parameter**

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

**Remarks**

When the digital channel is used as the trigger source channel, the digital probe must be connected to the oscilloscope.

**Return Format**

Query returns CHAN1, CHAN2, CHAN3, CHAN4, EXT.

**Example**

```
:TRIGger:EDGE:SOURce CHANnel1    /* Set the trigger source to CH1*/
:TRIGger:EDGE:SOURce?            /* Query returns CHAN1*/
```

### 3.12.6 :TRIGger:EDGE:SLOPe

#### Syntax

```
:TRIGger:EDGE:SLOPe <slope>
```

```
:TRIGger:EDGE:SLOPe?
```

#### Description

Set or query the edge type triggered by the edge.

#### Parameter

| Name    | Type     | Range                   | Default |
|---------|----------|-------------------------|---------|
| <slope> | Discrete | {RISIng FALLing EITHer} | RISIng  |

#### Remarks

- RISING: Rising edge
- FALLING: Falling edge
- EITHer: Any edge

#### Return Format

Query returns RISI FALL or EITH.

#### Example

```
:TRIGger:EDGE:SLOPe FALLing      /* Set the edge type to the falling edge */
```

```
:TRIGger:EDGE:SLOPe?             /* Query returns FALL */
```

### 3.12.7 :TRIGger:EDGE:LEVeL

#### Syntax

```
:TRIGger:EDGE:LEVeL <level>
```

```
:TRIGger:EDGE:LEVeL?
```

#### Description

Set or query the triggering level when the edge is triggered, with the unit consistent with the current amplitude unit of the selected signal source.

#### Parameter

| Name    | Type | Range                                                   | Default |
|---------|------|---------------------------------------------------------|---------|
| <level> | Real | (-5×VerticalScale-OFFSet)-<br>(5× VerticalScale-OFFSet) | 0       |

#### Remarks

This setting command is only valid when the selected signal source is an analog channel.

#### Return Format

Query to return the trigger level value in scientific counting form.

#### Example

```
:TRIGger:EDGE:LEVel 0.16      /* Set the trigger level to 160mV */
:TRIGger:EDGE:LEVel?          /* Query returns 1.600000 e-01*/
```

### 3.12.8 :TRIGger:PULSe:SOURce

#### Syntax

```
:TRIGger:PULSe:SOURce <source>
:TRIGger:PULSe:SOURce?
```

#### Description

Set or query the trigger source for pulse width triggering.

#### Parameter

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

#### Remarks

When the digital channel is used as the trigger source channel, the digital probe must be connected to the oscilloscope.

#### Return Format

Query returns CHAN1 CHAN2, CHAN3, CHAN4, EXT.

#### Example

```
:TRIGger:PULSe:SOURce CHANnel1      /* Set the trigger source to CH1*/
:TRIGger:PULSe:SOURce?              /* Query returns CHAN1*/
```

### 3.12.9 :TRIGger:PULSe:POLarity

#### Syntax

```
:TRIGger:PULSe:POLarity <polarity>
:TRIGger:PULSe:POLarity?
```

**Description**

Set or query the triggering polarity of pulse width triggering.

**Parameter**

| Name       | Type     | Range                  | Default  |
|------------|----------|------------------------|----------|
| <polarity> | Discrete | {POSitive   NEGActive} | POSitive |

**Remarks**

- POSitive: Positive polarity
- NEGActive: Negative polarity

**Return Format**

Query returns POSitive, NEGActive.

**Example**

```
:TRIGger:PULSe:POLarity POSitive      /* Set the triggering polarity to positive
polarity */
:TRIGger:PULSe:POLarity?              /* Query returns POSitive */
```

### 3.12.10 :TRIGger:PULSe:WHEN

**Syntax**

```
:TRIGger:PULSe:WHEN <when>
:TRIGger:PULSe:WHEN?
```

**Description**

Set or query the triggering conditions for pulse width triggering.

**Parameter**

| Name   | Type     | Range                          | Default |
|--------|----------|--------------------------------|---------|
| <when> | Discrete | { EQUAL  NEQUAL  GREAt  LESS } | EQUAL   |

**Remarks**

- Equal=(equal to time value): can only be triggered when the positive or negative pulse width of the input signal is equal to the set pulse width. [Pulse width error is 5%]
- NEQUAL= (Not equal to time value): Only when the positive or negative pulse width of the input signal is not equal to the set pulse width, can it trigger 【 Pulse width error is 5% 】 .
- GREAt>(greater than time value): Only when the positive or negative pulse width of the input signal is greater than the set pulse width can it trigger [pulse width error is

5%].

- LESS<(less than time value): Only when the positive or negative pulse width of the input signal is less than the set pulse width can it trigger [pulse width error is 5%].

#### Return Format

Query returns QUAL, NEQUal, GREAt, LESS.

#### Example

```
:TRIGger:PULSe:WHEN LESS      /* Set the trigger condition to LESS */
:TRIGger:PULSe:WHEN?          /* Query returns LESS */
```

### 3.12.11 :TRIGger:PULSe:WIDTh

#### Syntax

```
:TRIGger:PULSe:WIDTh <width>
:TRIGger:PULSe:WIDTh?
```

#### Description

Set or query the pulse width value when triggered, with a default unit of s.

#### Parameter

| Name    | Type | Range   | Default |
|---------|------|---------|---------|
| <width> | Real | 8ns-10s | 20ns    |

#### Remarks

This command is applicable to trigger conditions.

#### Return Format

Query returns the pulse width value in scientific counting form.

#### Example

```
:TRIGger:PULSe:WIDTh 0.000003    /* Set the pulse width value to 3 μ s */
:TRIGger:PULSe:WIDTh?            /* Query returned 3.000000 e-06*/
```

### 3.12.12 :TRIGger:PULSe:LEVel

#### Syntax

```
:TRIGger:PULSe:LEVel <level>
:TRIGger:PULSe:LEVel?
```

**Description**

Set or query the triggering level when pulse width is triggered, with the unit consistent with the current amplitude unit.

**Parameter**

| Name    | Type | Range                                                                                                | Default |
|---------|------|------------------------------------------------------------------------------------------------------|---------|
| <level> | Real | $(-5 \times \text{VerticalScale} - \text{OFFSet}) - (5 \times \text{VerticalScale} - \text{OFFSet})$ | 0       |

**Remarks**

This setting command is only valid when the selected signal source is an analog channel.

**Return Format**

Query to return the trigger level value in scientific counting form.

**Example**

```
:TRIGger:PULSe:LEVel 0.16          /* Set the trigger level to 160mV */
:TRIGger:PULSe:LEVel?              /* Query returns 1.600000 e-01*/
```

### 3.12.13 :TRIGger:SLOPe:SOURce

**Syntax**

```
:TRIGger:SLOPe:SOURce <source>
:TRIGger:SLOPe:SOURce?
```

**Description**

Set or query the trigger source for slope triggering.

**Parameter**

| Name     | Type     | Range                                 | Default  |
|----------|----------|---------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4} | CHANnel1 |

**Remarks**

Can only be triggered for analog channels.

**Return Format**

Query returns CHAN1 CHAN2, CHAN3, or CHAN4.

**Example**

```
:TRIGger:SLOPe:SOURce CHANnel2    /* Set the trigger source to CH2*/
:TRIGger:SLOPe:SOURce?             /* Query returns CHAN2*/
```

### 3.12.14 :TRIGger:SLOPe:POLarity

#### Syntax

```
:TRIGger:SLOPe:POLarity <polarity>
:TRIGger:SLOPe:POLarity?
```

#### Description

Set or query the edge type triggered by slope.

#### Parameter

| Name       | Type     | Range                 | Default  |
|------------|----------|-----------------------|----------|
| <polarity> | Discrete | {POSItive   NEGAtive} | POSItive |

#### Remarks

- POSItive: Rising edge triggered
- NEGAtive: Falling edge triggered

#### Return Format

Query returns POSItive, NEGAtive.

#### Example

```
:TRIGger:SLOPe:POLarity POSItive    /* Set rising edge trigger */
:TRIGger:SLOPe:POLarity?             /* Query returns POSItive */
```

### 3.12.15 :TRIGger:SLOPe:WHEN

#### Syntax

```
:TRIGger:SLOPe:WHEN <when>
:TRIGger:SLOPe:WHEN?
```

#### Description

Set or query the triggering conditions for slope triggering.

#### Parameter

| Name   | Type     | Range                          | Default |
|--------|----------|--------------------------------|---------|
| <when> | Discrete | { EQUAL  NEQUAl  GREAT  LESS } | EQUAl   |

**Remarks**

- **Equal**=(equal to time value): can only be triggered when the positive or negative pulse width of the input signal is equal to the set pulse width. [Pulse width error is 5%]
- **NEQUAL**=(Not equal to time value): Only when the positive or negative pulse width of the input signal is not equal to the set pulse width, can it trigger 【 Pulse width error is 5% 】 .
- **GREAt**=(greater than time value): Only when the positive or negative pulse width of the input signal is greater than the set pulse width can it trigger [pulse width error is 5%].
- **LESS**<(less than time value): Only when the positive or negative pulse width of the input signal is less than the set pulse width can it trigger [pulse width error is 5%].

**Return Format**

Query returns QUAL, NEQUal, GREAt, LESS.

**Example**

```
:TRIGger:SLOPe:WHEN LESS      /* Set the trigger condition to LESS */
:TRIGger:SLOPe:WHEN?          /* Query returns LESS */
```

### 3.12.16 **:TRIGger:SLOPe:TIME**

**Syntax**

```
:TRIGger:SLOPe:TIME <time>
:TRIGger:SLOPe:TIME?
```

**Description**

Set or query the time value when the slope is triggered. The default unit is s.

**Parameter**

| Name   | Type | Range   | Default |
|--------|------|---------|---------|
| <time> | Real | 8ns-10s | 20ns    |

**Remarks**

This command is applicable to trigger conditions.

**Return Format**

Query returns a time value in scientific counting form.

**Example**

```
:TRIGger:SLOPe:TIME 0.000003      /* Set the time value to 3 μ s */
```

```
:TRIGger:SLOPe:TIME?
```

```
/* Query returned 3.000000 e-06*/
```

### 3.12.17 :TRIGger:SLOPe:ALEVel

#### Syntax

```
:TRIGger:SLOPe:ALEVel <level>
```

```
:TRIGger:SLOPe:ALEVel?
```

#### Description

Set or query the upper limit of the triggering level when the slope is triggered, with the unit consistent with the current amplitude unit.

#### Parameter

| Name    | Type | Range                                                                                                | Default |
|---------|------|------------------------------------------------------------------------------------------------------|---------|
| <level> | Real | $(-5 \times \text{VerticalScale} - \text{OFFSet}) - (5 \times \text{VerticalScale} - \text{OFFSet})$ | 2V      |

#### Remarks

This setting command is only valid when the selected signal source is an analog channel.

#### Return Format

Query to return the trigger level upper limit in scientific counting form.

#### Example

```
:TRIGger:SLOPe:ALEVel 0.16
```

```
/* Set the upper limit of the trigger level to 160mV
```

```
*/
```

```
:TRIGger:SLOPe:ALEVel?
```

```
/* Query returns 1.600000 e-01*/
```

### 3.12.18 :TRIGger:SLOPe:BLEVel

#### Syntax

```
:TRIGger:SLOPe:BLEVel <level>
```

```
:TRIGger:SLOPe:BLEVel?
```

#### Description

Set or query the lower limit of the triggering level when the slope is triggered, with the unit consistent with the current amplitude unit.

#### Parameter

| Name    | Type | Range                                                                                                | Default |
|---------|------|------------------------------------------------------------------------------------------------------|---------|
| <level> | Real | $(-5 \times \text{VerticalScale} - \text{OFFSet}) - (5 \times \text{VerticalScale} - \text{OFFSet})$ | 2V      |

**Remarks**

This setting command is only valid when the selected signal source is an analog channel.

**Return Format**

Query to return the lower limit of trigger level in scientific counting form.

**Example**

```
:TRIGger:SLOPe:BLEVel 0.16      /* Set the trigger level lower limit to 160mV */
:TRIGger:SLOPe:BLEVel?          /* Query returns 1.600000 e-01*/
```

### 3.12.19 :TRIGger:VIDeo:SOURce

**Syntax**

```
:TRIGger:VIDeo:SOURce <source>
:TRIGger:VIDeo:SOURce?
```

**Description**

Set or query the trigger source for video triggering.

**Parameter**

| Name     | Type     | Range                                 | Default  |
|----------|----------|---------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4} | CHANnel1 |

**Remarks**

N/A

**Return Format**

Query returns CHAN1 CHAN2, CHAN3, or CHAN4.

**Example**

```
:TRIGger:VIDeo:SOURce CHANnel2  /* Set the trigger source to CH2*/
:TRIGger:VIDeo:SOURce?          /* Query returns CHAN2*/
```

### 3.12.20 :TRIGger:VIDeo:POLarity

#### Syntax

:TRIGger:VIDeo:POLarity <polarity>

:TRIGger:VIDeo:POLarity?

#### Description

Select or query the video polarity when triggered.

#### Parameter

| Name       | Type     | Range               | Default  |
|------------|----------|---------------------|----------|
| <polarity> | Discrete | {POSitive NEGative} | POSitive |

#### Remarks

N/A

#### Return Format

Query and return POS or NEG.

#### Example

:TRIGger:VIDeo:POLarity POSitive /\* Set video polarity to positive polarity \*/

:TRIGger:VIDeo:POLarity? /\* Query returns POS \*/

### 3.12.21 :TRIGger:VIDeo:MODE

#### Syntax

:TRIGger:VIDeo:MODE <mode>

:TRIGger:VIDeo:MODE?

#### Description

Set or query the synchronization type when the video is triggered.

#### Parameter

| Name   | Type     | Range                                         | Default  |
|--------|----------|-----------------------------------------------|----------|
| <mode> | Discrete | {SCANLine LINE ODDField EVENfield ALIN<br>es} | SCANLine |

#### Remarks

- SCANLine: Scanning line

- LINE: For NTSC and PAL/SECAM video standards, triggered on specified rows in odd or even fields.
- ODDField: Triggered at the rising edge of the first sawtooth wave in an odd numbered field.
- EVENField: Triggered at the rising edge of the first sawtooth wave in an even numbered field.
- ALINs: triggered on all horizontal synchronization pulses.

#### Return Format

Query returns SCANL LINE, ODDF, EVEN, or ALIN.

#### Example

```
:TRIGger:VIDeo:MODE ODDField      /* Set synchronization type to odd field */
:TRIGger:VIDeo:MODE?              /* Query returns ODDF */
```

### 3.12.22 :TRIGger:VIDeo:LINE

#### Syntax

```
:TRIGger:VIDeo:LINE <line>
:TRIGger:VIDeo:LINE?
```

#### Description

Set or query the number of lines when the video is triggered.

#### Parameter

| Name   | Type    | Range                   | Default |
|--------|---------|-------------------------|---------|
| <line> | Integer | Please refer to Remarks | 10      |

#### Remarks

The minimum value is 10.

#### Return Format

Query returns an integer.

#### Example

```
:TRIGger:VIDeo:LINE 100          /* Set line number to 100*/
:TRIGger:VIDeo:LINE?             /* Query returns 100*/
```

### 3.12.23 :TRIGger:VIDeo:LEVel

#### Syntax

```
:TRIGger:VIDeo:LEVel <level>
```

```
:TRIGger:VIDeo:LEVel?
```

### Description

Set or query the triggering level when the video is triggered, with the unit consistent with the current amplitude unit.

### Parameter

| Name    | Type | Range                                                                                                | Default |
|---------|------|------------------------------------------------------------------------------------------------------|---------|
| <level> | Real | $(-5 \times \text{VerticalScale} - \text{OFFSet}) - (5 \times \text{VerticalScale} - \text{OFFSet})$ | 0       |

### Remarks

This setting command is only valid when the selected signal source is an analog channel.

### Return Format

Query to return trigger level in scientific counting form.

### Example

```
:TRIGger:VIDeo:LEVel 0.16      /* Set the trigger level to 160mV */
```

```
:TRIGger:VIDeo:LEVel?          /* Query returns 1.600000 e-01*/
```

## 3.12.24 :TRIGger:TIMEout:SOURce

### Syntax

```
:TRIGger:TIMEout:SOURce <source>
```

```
:TRIGger:TIMEout:SOURce?
```

### Description

Set or query the trigger source triggered by timeout.

### Parameter

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

### Remarks

When the digital channel is used as the trigger source channel, the digital probe must be connected to the oscilloscope.

### Return Format

Query returns CHAN1 CHAN2, CHAN3, CHAN4, EXT.

#### Example

```
:TRIGger:TIMEout:SOURce CHANnel2      /* Set the trigger source to CH2*/
:TRIGger:TIMEout:SOURce?               /* Query returns CHAN2*/
```

### 3.12.25 :TRIGger:TIMEout:SLOPe

#### Syntax

```
:TRIGger:TIMEout:SLOPe <slope>
:TRIGger:TIMEout:SLOPe?
```

#### Description

Set or query the edge type triggered by timeout.

#### Parameter

| Name    | Type     | Range               | Default  |
|---------|----------|---------------------|----------|
| <slope> | Discrete | {POSitive NEGative} | POSitive |

#### Remarks

- Positive: Start timing by triggering the level on the rising edge of the input signal.
- NEGative: Start timing by triggering the level on the falling edge of the input signal.

#### Return Format

Query and return POS or NEG.

#### Example

```
:TRIGger:TIMEout:SLOPe NEGative      /* Set the edge type to the falling edge */
:TRIGger:TIMEout:SLOPe?               /* Query returns NEG */
```

### 3.12.26 :TRIGger:TIMEout:TIME

#### Syntax

```
:TRIGger:TIMEout:TIME <NR3>
:TRIGger:TIMEout:TIME?
```

#### Description

Set or query the timeout triggered by timeout. The default unit is s.

#### Parameter

| Name | Type | Range | Default |
|------|------|-------|---------|
|------|------|-------|---------|

| Name  | Type | Range    | Default |
|-------|------|----------|---------|
| <NR3> | Real | 10ns-10s | 10ns    |

**Remarks**

N/A

**Return Format**

Query returns the timeout value in scientific counting form.

**Example**

```
:TRIGger:TIMEout:TIME 0.002      /* Set the timeout time to 2ms */
:TRIGger:TIMEout:TIME?           /* Query returns 2.000000 e-03*/
```

### 3.12.27 :TRIGger:TIMEout:LEVel

**Syntax**

```
:TRIGger:TIMEout:LEVel <level>
:TRIGger:TIMEout:LEVel?
```

**Description**

Set or query the triggering level when timeout is triggered.

**Parameter**

| Name    | Type | Range                                                                                                | Default |
|---------|------|------------------------------------------------------------------------------------------------------|---------|
| <level> | Real | $(-5 \times \text{VerticalScale} - \text{OFFSet}) - (5 \times \text{VerticalScale} - \text{OFFSet})$ | 0       |

**Remarks**

This setting command is only valid when the selected signal source is an analog channel.

**Return Format**

Returns the trigger level value in scientific counting form.

**Example**

```
:TRIGger:TIMEout:LEVel 0.16      /* Set the trigger level to 160mv */
:TRIGger:TIMEout:LEVel?          /* Query returns 1.600000 e-01*/
```

### 3.12.28 :TRIGger:RUNT:SOURce

#### Syntax

```
:TRIGger:RUNT:SOURce <source>
```

```
:TRIGger:RUNT:SOURce?
```

#### Description

Set or query the trigger source for underamplitude triggering.

#### Parameter

| Name     | Type     | Range                                 | Default  |
|----------|----------|---------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4} | CHANnel1 |

#### Remarks

Can only simulate channels as trigger sources.

#### Return Format

Query returns CHAN1 CHAN2, CHAN3, or CHAN4.

#### Example

```
:TRIGger:RUNT:SOURce CHANnel2      /* Set the trigger source to CH2*/
:TRIGger:RUNT:SOURce?                /* Query returns CHAN2*/
```

### 3.12.29 :TRIGger:RUNT:POLarity

#### Syntax

```
:TRIGger:RUNT:POLarity <polarity>
```

```
:TRIGger:RUNT:POLarity?
```

#### Description

Set or query the pulse polarity triggered by underamplitude.

#### Parameter

| Name       | Type     | Range                 | Default  |
|------------|----------|-----------------------|----------|
| <polarity> | Discrete | {POSitive   NEGative} | POSitive |

#### Remarks

- Positive: Triggered on the forward underamplitude pulse.
- NEGative: Triggered on negative underamplitude pulses.

**Return Format**

Query returns POS, NEG.

**Example**

```
:TRIGger:RUNT:POLarity NEGative      /* Set the pulse polarity to negative polarity
*/
:TRIGger:RUNT:POLarity?                /* Query returns NEG */
```

**3.12.30 :TRIGger:RUNT:WHEN****Syntax**

```
:TRIGger:RUNT:WHEN <when>
:TRIGger:RUNT:WHEN?
```

**Description**

Set or query qualifiers for underamplitude triggering.

**Parameter**

| Name   | Type     | Range                          | Default |
|--------|----------|--------------------------------|---------|
| <when> | Discrete | { EQUAL  NEQUAl  GREAt  LESS } | EQUAl   |

**Remarks**

- Equal=(equal to time value): can only be triggered when the positive or negative pulse width of the input signal is equal to the set pulse width. [Pulse width error is 5%]
- NEQUAl= (Not equal to time value): Only when the positive or negative pulse width of the input signal is not equal to the set pulse width, can it trigger 【 Pulse width error is 5% 】 .
- GREAt>(greater than time value): Only when the positive or negative pulse width of the input signal is greater than the set pulse width can it trigger [pulse width error is 5%].
- LESS<(less than time value): Only when the positive or negative pulse width of the input signal is less than the set pulse width can it trigger [pulse width error is 5%].

**Return Format**

Query returns QUAL, NEQUAl, GREAt, LESS.

**Example**

```
:TRIGger:RUNT:WHEN LESS      /* Set qualifiers to<*/
:TRIGger:RUNT:WHEN?          /* Query returns LESS */
```

### 3.12.31 :TRIGger:RUNT:TIME

#### Syntax

:TRIGger:RUNT:TIME <NR3>

:TRIGger:RUNT:TIME?

#### Description

Set or query the pulse width triggered by underamplitude, with a default unit of s.

#### Parameter

| Name  | Type | Range   | Default |
|-------|------|---------|---------|
| <NR3> | Real | 8ns-10s | 2us     |

#### Remarks

N/A

#### Return Format

N/A

#### Example

:TRIGger:RUNT:TIME 0.02 /\* Set the upper limit of pulse width to 20ms \*/

:TRIGger:RUNT:TIME? /\* Query returned 2.000000e-02\*/

### 3.12.32 :TRIGger:RUNT:ALEVel

#### Syntax

:TRIGger:RUNT:ALEVel <level>

:TRIGger:RUNT:ALEVel?

#### Description

Set or query the upper limit of the triggering level when triggering under amplitude, with the unit consistent with the current amplitude unit.

#### Parameter

| Name    | Type | Range                                                                                                | Default |
|---------|------|------------------------------------------------------------------------------------------------------|---------|
| <level> | Real | $(-5 \times \text{VerticalScale} - \text{OFFSet}) - (5 \times \text{VerticalScale} - \text{OFFSet})$ | 2V      |

#### Remarks

This setting command is only valid when the selected signal source is an analog channel.

**Return Format**

Query to return the trigger level upper limit in scientific counting form.

**Example**

```
:TRIGger:RUNT:ALEVel 0.16      /* Set the upper limit of the trigger level to 160mV */
:TRIGger:RUNT:ALEVel?          /* Query returns 1.600000 e-01*/
```

**3.12.33 :TRIGger:RUNT:BLEVel****Syntax**

```
:TRIGger:RUNT:BLEVel <level>
:TRIGger:RUNT:BLEVel?
```

**Description**

Set or query the lower limit of the triggering level when triggering under amplitude, with the unit consistent with the current amplitude unit.

**Parameter**

| Name    | Type | Range                                                                                                | Default |
|---------|------|------------------------------------------------------------------------------------------------------|---------|
| <level> | Real | $(-5 \times \text{VerticalScale} - \text{OFFSet}) - (5 \times \text{VerticalScale} - \text{OFFSet})$ | 0       |

**Remarks**

This setting command is only valid when the selected signal source is an analog channel.

**Return Format**

Query to return the lower limit of trigger level in scientific counting form.

**Example**

```
:TRIGger:RUNT:BLEVel 0.16      /* Set the trigger level lower limit to 160mV */
:TRIGger:RUNT:BLEVel?          /* Query returns 1.600000 e-01*/
```

**3.12.34 :TRIGger: PATTern:SOURce****Syntax**

```
:TRIGger:PATTern:SOURce <Source>
:TRIGger:PATTern:SOURce?
```

**Description**

Set or query the trigger source for logical triggering.

**Parameter**

| Name     | Type     | Range                                 | Default  |
|----------|----------|---------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4} | CHANnel1 |

**Remarks**

When the digital channel is used as the trigger source channel, the digital probe must be connected to the oscilloscope.

**Return Format**

Query returns CHAN1 CHAN2, CHAN3, CHAN4.

**Example**

```
:TRIGger:PATtern:SOURce CHANnel2    /* Set the trigger signal source to CH2*/
:TRIGger:PATtern:SOURce?             /* Query returns CHAN2*/
```

### 3.12.35 :TRIGger:PATtern:CODE

**Syntax**

```
:TRIGger:PATtern:CODE <CODE>
:TRIGger:PATtern:CODE?
```

**Description**

Set or query the code type configuration triggered by logic.

**Parameter**

| Name   | Type     | Range                         | Default |
|--------|----------|-------------------------------|---------|
| <code> | Discrete | {H L X RISIng FALLing EITHer} | H       |

**Remarks**

H: High level

L: Low level

X: When set to X, the oscilloscope will not trigger

RISIng: Rising edge

FALLing: Falling edge

EITER: Any edge

**Return Format**

Query returns H, L, X, RISIng, FALLing, EITHer.

**Example**

```
:TRIGger:PATtern:CODE L      /* Set the code type configuration to L */
:TRIGger:PATtern:CODE?      /* Query returns L */
```

**3.12.36 :TRIGger:PATtern:PATtern****Syntax**

```
:TRIGger:PATtern:PATtern <pattern>
:TRIGger:PATtern:PATtern?
```

**Description**

Set or query the logical type triggered by the logic.

**Parameter**

| Name      | Type     | Range    | Default |
|-----------|----------|----------|---------|
| <pattern> | Discrete | {OR AND} | OR      |

**Remarks**

OR: or; AND: AND.

**Return Format**

Query returns OR AND.

**Example**

```
:TRIGger: PATtern:PATtern OR    /* Set logical type to OR */
:TRIGger:PATtern:PATtern?      /* Query returns OR */
```

**3.12.37 :TRIGger:DELAy:SA****Syntax**

```
:TRIGger:DELAy:SA <Source>
:TRIGger:DELAy:SA?
```

**Description**

Set or query the triggering signal source of source A when delayed triggering occurs.

**Parameter**

| Name | Type | Range | Default |
|------|------|-------|---------|
|------|------|-------|---------|

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

**Remarks**

When the digital channel is used as the trigger source channel, the digital probe must be connected to the oscilloscope. Cannot set source A and source B as the same source at the same time.

**Return Format**

Query returns CHAN1 CHAN2, CHAN3, CHAN4, EXT.

**Example**

```
:TRIGger:DElay:SA CHANnel2      /* Set trigger source A to CH2*/
:TRIGger:DElay:SA?              /* Query returns CHAN2*/
```

### 3.12.38 :TRIGger:DElay:SLOPA

**Syntax**

```
:TRIGger:DElay:SLOPA <slope>
:TRIGger:DElay:SLOPA?
```

**Description**

Set or query the edge type of edge A when delayed triggering occurs.

**Parameter**

| Name    | Type     | Range               | Default  |
|---------|----------|---------------------|----------|
| <slope> | Discrete | {POSitive NEGative} | POSitive |

**Remarks**

- Positive: Triggered by the rising edge.
- NEGative: triggered by the falling edge.

**Return Format**

Query and return POS or NEG.

**Example**

```
:TRIGger:DElay:SLOPA NEGative    /* Set the Type of edge A to the falling edge
*/
:TRIGger:DElay:SLOPA?            /* Query returns NEG */
```

### 3.12.39 :TRIGger:DElay:SB

#### Syntax

:TRIGger:DElay:SB <Source>

:TRIGger:DElay:SB?

#### Description

Set or query the triggering signal source of source B when delayed triggering occurs.

#### Parameter

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel2 |

#### Remarks

When the digital channel is used as the trigger source channel, the digital probe must be connected to the oscilloscope. Cannot set source A and source B as the same source at the same time.

#### Return Format

Query returns CHAN1 CHAN2, CHAN3, CHAN4, EXT.

#### Example

:TRIGger:DElay:SB CHANnel4 /\* Set trigger source B to CH4\*/

:TRIGger:DElay:SB? /\* Query returns CHAN4\*/

### 3.12.40 :TRIGger:DElay:SLOPB

#### Syntax

:TRIGger:DElay:SLOPB <slope>

:TRIGger:DElay:SLOPB?

#### Description

Set or query the edge type of edge B when delayed triggering.

#### Parameter

| Name    | Type     | Range               | Default  |
|---------|----------|---------------------|----------|
| <slope> | Discrete | {POSitive NEGative} | POSitive |

#### Remarks

- Positive: Triggered by the rising edge.
- NEGative: triggered by the falling edge.

**Return Format**

Query and return POS or NEG.

**Example**

```
:TRIGger:DElay:SLOPB NEGative      /* Set the Type of edge B to the falling edge
*/
:TRIGger:DElay:SLOPB?               /* Query returns NEG */
```

### 3.12.41 :TRIGger:DElay:WHEN

**Syntax**

```
:TRIGger:DElay:WHEN <type>
:TRIGger:DElay:WHEN?
```

**Description**

Set or query the conditions for triggering delay.

**Parameter**

| Name   | Type     | Range                          | Default |
|--------|----------|--------------------------------|---------|
| <type> | Discrete | { EQUAL  NEQUAl  GREAt  LESS } | EQUAl   |

**Remarks**

- Equall=(equal to time value): can only be triggered when the positive or negative pulse width of the input signal is equal to the set pulse width. [Pulse width error is 5%]
- NEQUAl= (Not equal to time value): Only when the positive or negative pulse width of the input signal is not equal to the set pulse width, can it trigger 【 Pulse width error is 5% 】.
- GREAt>(greater than time value): Only when the positive or negative pulse width of the input signal is greater than the set pulse width can it trigger [pulse width error is 5%].
- LESS<(less than time value): Only when the positive or negative pulse width of the input signal is less than the set pulse width can it trigger [pulse width error is 5%].

**Return Format**

Query returns QUAL, NEQUAl, GREAt, LESS.

**Example**

```
:TRIGger:DElay:WHEN LESS      /* Set the delay type to<*/
```

:TRIGger:DElay:WHEN? /\* Query returns LESS \*/

### 3.12.42 :TRIGger:DElay:TIME

#### Syntax

:TRIGger:DElay:TIME <value>

:TRIGger:DElay:TIME?

#### Description

Set or query the time value when the delay is triggered.

#### Parameter

| Name    | Type     | Range   | Default |
|---------|----------|---------|---------|
| <value> | Discrete | 8ns-10s | 2us     |

#### Remarks

N/A

#### Return Format

Returns a time value in the form of scientific counting.

#### Example

:TRIGger:DElay:TIME 0.000003 /\* Set the upper limit of pulse width to 3us \*/

:TRIGger:DElay:TIME? /\* Query returned 3.000000 e-06\*/

### 3.12.43 :TRIGger:SHOLd:DSrc

#### Syntax

:TRIGger:SHOLd:DSrc <Source>

:TRIGger:SHOLd:DSrc?

#### Description

Set or query the data source that maintains triggering.

#### Parameter

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel2 |

#### Remarks

When the digital channel is used as the trigger source channel, the digital probe must be connected to the oscilloscope. The clock source and data source cannot be set to the same signal source at the same time.

#### Return Format

Query returns CHAN1 CHAN2, CHAN3, CHAN4, EXT.

#### Example

```
:TRIGger:SHOLd:DSrc CHANnel1      /* Set the data source to CH1*/
:TRIGger:SHOLd:DSrc?               /* Query returns CHAN1*/
```

### 3.12.44 :TRIGger:SHOLd:CSrc

#### Syntax

```
:TRIGger:SHOLd:CSrc <Source>
:TRIGger:SHOLd:CSrc?
```

#### Description

Set or query to establish a clock source that keeps triggering.

#### Parameter

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

#### Remarks

When the digital channel is used as the trigger source channel, the digital probe must be connected to the oscilloscope. The clock source and data source cannot be set to the same signal source at the same time.

#### Return Format

Query returns CHAN1 CHAN2, CHAN3, CHAN4, EXT.

#### Example

```
:TRIGger:SHOLd:CSrc CHANnel2      /* Set the clock source to CH2*/
:TRIGger:SHOLd:CSrc?               /* Query returns CHAN2*/
```

### 3.12.45 :TRIGger:SHOLd:SLOPe

#### Syntax

```
:TRIGger:SHOLd:SLOPe <slope>
```

:TRIGger:SHOLd:SLOPe?

### Description

Set or query the edge type that maintains triggering.

### Parameter

| Name    | Type     | Range               | Default  |
|---------|----------|---------------------|----------|
| <slope> | Discrete | {POSitive NEGative} | POSitive |

### Remarks

- Positive: triggered by the rising edge.
- NEGative: triggered by the falling edge.

### Return Format

Query and return POS or NEG.

### Example

```
:TRIGger:SHOLd:SLOPe NEGative      /* Set the edge type to the falling edge */
:TRIGger:SHOLd:SLOPe?                /* Query returns NEG */
```

## 3.12.46 :TRIGger:SHOLd:PATtern

### Syntax

```
:TRIGger:SHOLd:PATtern <pattern>
:TRIGger:SHOLd:PATtern?
```

### Description

设置 or 查询建立保持触发的数据类型 Set or query the data type that maintains triggering.

### Parameter

| Name      | Type     | Range | Default |
|-----------|----------|-------|---------|
| <pattern> | Discrete | {H L} | H       |

### Remarks

- H: High level.
- L: Low level.

### Return Format

Query returns H or L.

### Example

```
:TRIGger:SHOLd:PATtern L      /* Set the data type to L */
:TRIGger:SHOLd:PATtern?      /* Query returns L */
```

### 3.12.47 :TRIGger:SHOLd:TYPe

#### Syntax

```
:TRIGger:SHOLd:TYPe <type>
:TRIGger:SHOLd:TYPe?
```

#### Description

Set or query the retention type triggered by the establishment of retention.

#### Parameter

| Name   | Type     | Range                | Default |
|--------|----------|----------------------|---------|
| <type> | Discrete | {SETup HOLd SETHOLd} | SETup   |

#### Remarks

- SETup: When the setup time is less than the set value (TRIGger: SHOLd: STIME), the oscilloscope will trigger.
- HOLd: Hold, when the hold time is less than the set value (TRIGger: SHOLd: HTIME), the oscilloscope triggers.

#### Return Format

Query returns SET HOL, SETHOL.

#### Example

```
:TRIGger:SHOLd:TYPe SETHOLd      /* Set the retention type to establish
retention */
:TRIGger:SHOLd:TYPe?              /* Query returns SETHOL */
```

### 3.12.48 :TRIGger:SHOLd:STIME

#### Syntax

```
:TRIGger:SHOLd:STIME <NR3>
:TRIGger:SHOLd:STIME?
```

#### Description

Set or query the pulse width for maintaining triggering. The default unit is s.

#### Parameter

| Name | Type | Range | Default |
|------|------|-------|---------|
|------|------|-------|---------|

| Name  | Type | Range   | Default |
|-------|------|---------|---------|
| <NR3> | Real | 8ns-10s | 2us     |

**Remarks**

The establishment time refers to the time during which data remains stable and unchanged before the clock signal of the trigger arrives.

**Return Format**

Query returns the pulse width value when the trigger condition is established in scientific counting form.

**Example**

```
:TRIGger:SHOLd:STIMe 0.002      /* Set the pulse width value to 2ms.*/
:TRIGger:SHOLd:STIMe?           /*Query returns 2.000000 e-03*/
```

### 3.12.49 :TRIGger:UART:SOURce

**Syntax**

```
:TRIGger:UART:SOURce <source>
:TRIGger:UART:SOURce?
```

**Description**

Set or query the trigger source for UART.

**Parameter**

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

**Remarks**

N/A.

**Return Format**

Query returns CHAN1 CHAN2, CHAN3, CHAN4, EXT.

**Example**

```
:TRIGger:UART:SOURce CHANnel1    /* Set the trigger source to CH1*/
:TRIGger:UART:SOURce?            /*Query returns CHAN1*/
```

### 3.12.50 :TRIGger:UART:WHEN

#### Syntax

:TRIGger:UART:WHEN <when>

:TRIGger:UART:WHEN?

#### Description

Set or query the conditions when UART is triggered.

#### Parameter

| Name   | Type     | Range                                      | Default |
|--------|----------|--------------------------------------------|---------|
| <when> | Discrete | {START STOP READ_DATA PARITY_ERR COM_ERR } | START   |

#### Remarks

- START: When the UART start bit appears, it is triggered in the middle of the bit.
- STOP: When the UART stop bit appears, it is triggered in the middle of the bit. Regardless of whether the tested device stops at positions 1, 1.5, or 2, this machine will install one position for processing.
- READDATA: The normal data reception is completed, and the received UART data is equal to the user set data, which is triggered at the stop bit.
- PARITY-ERR: When the data is received normally and there is an error in the parity check of the data, it is triggered at the stop bit.
- COM-ERR: Triggered when an error occurs when the data is not stopped at the beginning of the verification bit.

#### Return Format

Query returns START, STOP, READ\_DATA, PARITY\_ERR, COM\_ERR.

#### Example

```
:TRIGger:UART:WHEN START      /* Set the trigger condition to START */
:TRIGger:UART:WHEN?           /* Query returns START */
```

### 3.12.51 :TRIGger:UART:BAUD

#### Syntax

:TRIGger:UART:BAUD <baud\_rate>

:TRIGger:UART:BAUD?

#### Description

Set or query the baud rate triggered by UART. The default unit is bps.

**Parameter**

| Name          | Type     | Range                                                                             | Default |
|---------------|----------|-----------------------------------------------------------------------------------|---------|
| < baud_rate > | Discrete | {600 1200 1800 2000 2400 4800 9600 19200 38400 57600 115200 230400 460800 921600} | 600     |

**Remarks**

N/A

**Return Format**

Query returns an integer.

**Example**

```
:TRIGger:UART:BAUD 4800      /* Set the baud rate to 4.8kbps */
:TRIGger:UART:BAUD?          /* Query returns 4800*/
```

### 3.12.52 :TRIGger:UART:LEVel

**Syntax**

```
:TRIGger:UART:LEVel <level>
:TRIGger:UART:LEVel?
```

**Description**

Set or query the triggering level when UART is triggered, with the unit consistent with the current amplitude unit.

**Parameter**

| Name    | Type | Range                                                                                                | Default |
|---------|------|------------------------------------------------------------------------------------------------------|---------|
| <level> | Real | $(-5 \times \text{VerticalScale} - \text{OFFSet}) - (5 \times \text{VerticalScale} - \text{OFFSet})$ | 0       |

**Remarks**

This setting command is only valid when the selected signal source is an analog channel.

**Return Format**

Query to return trigger level in scientific counting form.

**Example**

```
:TRIGger:UART:LEVel 0.16      /* Set the trigger level to 160mV*/
```

:TRIGger:UART:LEVel? /\* Query returns 1.600000 e-01\*/

### 3.12.53 :TRIGger:UART:DATA

#### Syntax

:TRIGger:UART:DATA <data>

:TRIGger:UART:DATA?

#### Description

Set or query the data value when UART trigger condition is data.

#### Parameter

| Name   | Type     | Range               | Default |
|--------|----------|---------------------|---------|
| <data> | Discrete | 0-2 <sup>n</sup> -1 | 0       |

#### Remarks

n is the current data width, with a range of 5, 6, 7, or 8.

#### Return Format

Query returns an integer.

#### Example

:TRIGger:UART:DATA 10 /\* Set the data value to 10\*/

:TRIGger:UART:DATA? /\* Query returns 10\*/

### 3.12.54 :TRIGger:UART:WIDTH

#### Syntax

:TRIGger:UART:WIDTH <width>

:TRIGger:UART:WIDTH?

#### Description

Set or query the data bit width when UART trigger condition is set to data.

#### Parameter

| Name    | Type     | Range     | Default |
|---------|----------|-----------|---------|
| <width> | Discrete | {5 6 7 8} | 5       |

#### Remarks

N/A

**Return Format**

Query returns 5, 6, 7, or 8.

**Example**

```
:TRIGger:UART:WIDTh 5      /* Set the data bit width to 5*/
:TRIGger:UART:WIDTh?      /* Query returns 5*/
```

### 3.12.55 :TRIGger:UART:PARity

**Syntax**

```
:TRIGger:UART:PARity <parity>
:TRIGger:UART:PARity?
```

**Description**

Set or query the verification method when the UART trigger condition is an error frame or verification error.

**Parameter**

| Name       | Type     | Range           | Default |
|------------|----------|-----------------|---------|
| < parity > | Discrete | {NONE ODD EVEN} | NONE    |

**Remarks**

When the triggering condition is a verification error, the verification method cannot be set to NONE. In this case, the verification method defaults to odd verification.

**Return Format**

Query returns NONE ODD or EVEN.

**Example**

```
:TRIGger:UART:PARity EVEN      /* Set the verification method to even verification
*/
:TRIGger:UART:PARity?          /* Query returns EVEN */
```

### 3.12.56 :TRIGger:UART:POLarity

**Syntax**

```
:TRIGger:UART:POLarity <polarity>
:TRIGger:UART:POLarity?
```

**Description**

Set or query the triggering polarity of UART.

**Parameter**

| Name       | Type     | Range               | Default  |
|------------|----------|---------------------|----------|
| <polarity> | Discrete | {POSITIVE NEGATIVE} | POSITIVE |

**Remarks**

- POSITIVE: Positive polarity.
- NEGATIVE: Negative polarity.

**Return Format**

Query returns POSITIVE, NEGATIVE.

**Example**

```
:TRIGGER:UART:POLARITY NEGATIVE /* Set the triggering polarity to negative polarity
*/
:TRIGGER:UART:POLARITY? /* Query returns NEGATIVE */
```

### 3.12.57 :TRIGGER:LIN:SOURCE

**Syntax**

```
:TRIGGER:LIN:SOURCE <source>
:TRIGGER:LIN:SOURCE?
```

**Description**

Set or query the trigger source for LIN triggering.

**Parameter**

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANNEL1 CHANNEL2 CHANNEL3 CHANNEL4 EXT} | CHANNEL1 |

**Remarks**

When the digital channel is used as the trigger source channel, the digital probe must be connected to the oscilloscope.

**Return Format**

Query returns CHAN1 CHAN2, CHAN3, CHAN4, EXT.

**Example**

```
:TRIGger:LIN:SOURce CHANnel1      /* Set the trigger source to CH1*/
:TRIGger:LIN:SOURce?              /* Query returns CHAN1*/
```

### 3.12.58 :TRIGger:LIN:BAUd

#### Syntax

```
:TRIGger:LIN:BAUd <baud>
:TRIGger:LIN:BAUd?
```

#### Description

Set or query the baud rate triggered by LIN. The default unit is bps.

#### Parameter

| Name   | Type     | Range                                                                              | Default |
|--------|----------|------------------------------------------------------------------------------------|---------|
| <baud> | Discrete | <600 1200 1800 2000 2400 4800 9600 19200 38400 57600 115200 230400 460800 921600 > | 600     |

#### Remarks

N/A

#### Return Format

Query returns an integer.

#### Example

```
:TRIGger:LIN:BAUd 4800      /* Set the baud rate to 4.8kbps */
:TRIGger:LIN:BAUd?          /* Query returns 4800*/
```

### 3.12.59 :TRIGger:LIN:CONdition

#### Syntax

```
:TRIGger:LIN:CONdition <condition>
:TRIGger:LIN:CONdition?
```

#### Description

Set or query the triggering conditions for LIN triggering.

#### Parameter

| Name        | Type     | Range                            | Default   |
|-------------|----------|----------------------------------|-----------|
| <condition> | Discrete | <INTERVAL_FIELD SYNC_FIELD ID_FI | INTERVAL_ |

| Name | Type | Range                                        | Default |
|------|------|----------------------------------------------|---------|
|      |      | ELD SYNC_CODE_ERROR IDENTIFIER ID_AND_DATA > | FIELD   |

**Remarks**

- The end of the interval field → triggered by the edge after the end of the LIN interval.
- The synchronization field ends → LIN synchronization field data reception is completed and triggered.
- The ID field ends and triggers the completion of data reception in the LINID field.
- Synchronization code error → triggered when LIN synchronization field data reception is completed but the synchronization field data is not equal to 0x55.
- Trigger when the frame ID → LINID field data is received and the ID data is equal to the user's set ID.
- Frame ID and data → LIN data received normally completed, The ID and data are equal to the user settings.

**Return Format**

Query returns

INTERVAL\_FIELD,SYNC\_FIELD,ID\_FIELD,SYNC\_CODE\_ERROR,IDENTIFIER,ID\_AND\_DATA.

**Example**

```
:TRIGger:LIN:CONdition INTERVAL_FIELD /* Set the trigger condition to end the interval field */
```

```
:TRIGger:LIN:CONdition? /* Query returns INTERVAL-FIELD */
```

### 3.12.60 :TRIGger:LIN:ID

**Syntax**

```
:TRIGger:LIN:ID <id>
```

```
:TRIGger:LIN:ID?
```

**Description**

Set or query the identifier triggered by LIN.

**Parameter**

| Name  | Type     | Range | Default |
|-------|----------|-------|---------|
| < id> | Discrete | 1~255 | 1       |

**Remarks**

N/A

**Return Format**

Query returns an integer.

**Example**

```
:TRIGger:LIN:ID 25      /* Set identifier to 25*/
:TRIGger:LIN:ID?        /* Query returns 25*/
```

### 3.12.61 :TRIGger:LIN:LEVel

**Syntax**

```
:TRIGger:LIN:LEVel <level>
:TRIGger:LIN:LEVel?
```

**Description**

Set or query the triggering level when LIN is triggered.

**Parameter**

| Name    | Type | Range                                                                                                | Default |
|---------|------|------------------------------------------------------------------------------------------------------|---------|
| <level> | Real | $(-5 \times \text{VerticalScale} - \text{OFFSet}) - (5 \times \text{VerticalScale} - \text{OFFSet})$ | 0       |

**Remarks**

This setting command is only valid when the channel source of the data cable is an analog channel.

**Return Format**

Query to return the trigger level value in scientific counting form.

**Example**

```
:TRIGger:LIN:LEVel 0.16      /* Set the trigger level to 160mV */
:TRIGger:LIN:LEVel?          /* Query returns 1.600000 e-01*/
```

### 3.12.62 :TRIGger:CAN:SOURce

**Syntax**

```
:TRIGger:CAN:SOURce <source>
:TRIGger:CAN:SOURce?
```

**Description**

Set or query the trigger source triggered by CAN.

#### Parameter

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

#### Remarks

N/A.

#### Return Format

Query returns CHAN1 CHAN2, CHAN3, CHAN4, EXT.

#### Example

```
:TRIGger:CAN:SOURce CHANnel1      /* Set the trigger source to CH1*/
:TRIGger:CAN:SOURce?              /* Query returns CHAN1*/
```

### 3.12.63 :TRIGger:CAN:BAUD

#### Syntax

```
:TRIGger:CAN:BAUD <baud>
:TRIGger:CAN:BAUD?
```

#### Description

Set or query the baud rate triggered by CAN. The default unit is bps.

#### Parameter

| Name     | Type     | Range                                                                             | Default |
|----------|----------|-----------------------------------------------------------------------------------|---------|
| < baud > | Discrete | <600 1200 1800 2000 2400 4800 9600 19200 38400 57600 115200 230400 460800 921600> | 600     |

#### Remarks

N/A

#### Return Format

Query returns an integer.

#### Example

```
:TRIGger:CAN:BAUD 4800      /* Set the baud rate to 4.8kbps */
:TRIGger:CAN:BAUD?          /* Query returns 4800*/
```

### 3.12.64 :TRIGger:CAN:CONdition

#### Syntax

:TRIGger:CAN:CONdition <condition>

:TRIGger:CAN:CONdition?

#### Description

Set or query the triggering conditions for CAN triggering.

#### Parameter

| Name        | Type     | Range                                                                                                                   | Default   |
|-------------|----------|-------------------------------------------------------------------------------------------------------------------------|-----------|
| <condition> | Discrete | <START_BIT REMOTE_FRAME_ID DATA_FRAME_ID FRAME_ID DATAFRAMEID_AND_DATA FRAME_ERROR ALL_ERROR ACK_ERROR OVERLOAD_FRAME > | START_BIT |

#### Remarks

N/A

#### Return Format

Query returns

START\_BIT,REMOTE\_FRAME\_ID,DATA\_FRAME\_ID,FRAME\_ID,DATAFRAMEID\_AND\_DATA,FRAME\_ERROR,ALL\_ERROR,ACK\_ERROR,OVERLOAD\_FRAME,.

#### Example

:TRIGger:CAN:CONdition START\_BIT /\* Set the trigger condition to START\_BIT \*/

:TRIGger:CAN:CONdition? /\* Query returns START\_BIT \*/

### 3.12.65 :TRIGger:CAN:LEVel

#### Syntax

:TRIGger:CAN:LEVel <level>

:TRIGger:CAN:LEVel?

#### Description

Set or query the triggering level when CAN is triggered.

#### Parameter

| Name | Type | Range | Default |
|------|------|-------|---------|
|------|------|-------|---------|

| Name    | Type | Range                                                                                                | Default |
|---------|------|------------------------------------------------------------------------------------------------------|---------|
| <level> | Real | $(-5 \times \text{VerticalScale} - \text{OFFSet}) - (5 \times \text{VerticalScale} - \text{OFFSet})$ | 0       |

**Remarks**

This setting command is only valid when the channel source of the data cable is an analog channel.

**Return Format**

Query to return the trigger level value in scientific counting form.

**Example**

```
:TRIGger:CAN:LEVel 0.16      /* Set the trigger level to 160mV */
:TRIGger:CAN:LEVel?          /* Query returns 1.600000 e-01*/
```

### 3.12.66 :TRIGger:SPI:MSIO:SOURce

**Syntax**

```
:TRIGger:SPI:MISO:SOURce <source>
:TRIGger:SPI:MISO:SOURce?
```

**Description**

Set or query the channel source for SPI triggered data lines.

**Parameter**

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel2 |

**Remarks**

The clock source and data source cannot be set to the same signal source at the same time.

**Return Format**

Query returns CHAN1 CHAN2, CHAN3, CHAN4, EXT.

**Example**

```
:TRIGger:SPI:MISO:SOURce CHANnel2      /* Set the channel source of the data
cable to CH2*/
:TRIGger:SPI:MISO:SOURce?              /* Query returns CHAN2*/
```

### 3.12.67 :TRIGger:SPI:SCL:SOURce

#### Syntax

:TRIGger:SPI:SCL:SOURce <source>

:TRIGger:SPI:SCL:SOURce?

#### Description

Set or query the channel source of the clock triggered by SPI.

#### Parameter

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

#### Remarks

The clock source and data source cannot be set to the same signal source at the same time.

#### Return Format

Query returns CHAN1 CHAN2, CHAN3, CHAN4, EXT.

#### Example

:TRIGger:SPI:SCL:SOURce CHANnel1           /\* Set the channel source of the clock line to CH1\*/

:TRIGger:SPI:SCL:SOURce?                   /\* Query returns CHAN1\*/

### 3.12.68 :TRIGger:SPI:SLOPe

#### Syntax

:TRIGger:SPI:SLOPe <slope>

:TRIGger:SPI:SLOPe?

#### Description

Set or query the Type of clock edge triggered by SPI.

#### Parameter

| Name    | Type     | Range               | Default  |
|---------|----------|---------------------|----------|
| <slope> | Discrete | {POSitive NEGative} | POSitive |

#### Remarks

- Positive: Sample SDA data at the rising edge of the clock.
- NEGative: Samples SDA data at the falling edge of the clock.

**Return Format**

Query returns POS, NEG.

**Example**

```
:TRIGger:SPI:SLOPe POSitive      /*Set the clock edge type to rising edge */
:TRIGger:SPI:SLOPe?              /* Query returns POS */
```

3.12.69

**:TRIGger:SPI:WIDTh**

---

**Syntax**

```
:TRIGger:SPI:WIDTh <width>
:TRIGger:SPI:WIDTh?
```

**Description**

Set or query the data bit width of the data channel triggered by SPI.

**Parameter**

| Name    | Type    | Range | Default |
|---------|---------|-------|---------|
| <width> | Integer | 4-32  | 4       |

**Remarks**

N/A

**Return Format**

Query returns an integer.

**Example**

```
:TRIGger:SPI:WIDTh 10      /*Set the data bit width to 10*/
:TRIGger:SPI:WIDTh?        /* Query returns 10*/
```

3.12.70

**:TRIGger:SPI:TIMeout**

---

**Syntax**

```
:TRIGger:SPI:TIMeout <time_value>
:TRIGger:SPI:TIMeout?
```

**Description**

Set or query the timeout period when the trigger condition for SPI is timeout. The default

unit is s.

#### Parameter

| Name         | Type | Range   | Default |
|--------------|------|---------|---------|
| <time_value> | Real | 8ns-10s | 16ns    |

#### Remarks

N/A

#### Return Format

Query returns the timeout time in scientific counting form.

#### Example

```
:TRIGger:SPI:TIMEout 0.001      /* Set the timeout time to 1ms */
:TRIGger:SPI:TIMEout?           /*Query returns 1.000000e-03*/
```

### 3.12.71 :TRIGger:SPI:CLEVel

#### Syntax

```
:TRIGger:SPI:CLEVel <level>
:TRIGger:SPI:CLEVel?
```

#### Description

Set or query the triggering level of the clock channel when SPI is triggered, with the unit consistent with the current amplitude unit.

#### Parameter

| Name    | Type | Range                                                                                                | Default |
|---------|------|------------------------------------------------------------------------------------------------------|---------|
| <level> | Real | $(-5 \times \text{VerticalScale} - \text{OFFSet}) - (5 \times \text{VerticalScale} - \text{OFFSet})$ | 0       |

#### Remarks

This setting command is only valid when the channel source of the clock line is an analog channel.

#### Return Format

Query to return trigger level in scientific counting form.

#### Example

```
:TRIGger:SPI:CLEVel 0.16        /* Set the trigger level to 160mV */
:TRIGger:SPI:CLEVel?           /* Query returns 1.600000 e-01*/
```

### 3.12.72 :TRIGger:SPI:DLEVel

#### Syntax

:TRIGger:SPI:DLEVel <level>

:TRIGger:SPI:DLEVel?

#### Description

Set or query the triggering level of the data channel when SPI is triggered, with the unit consistent with the current amplitude unit.

#### Parameter

| Name    | Type | Range                                                                                                | Default |
|---------|------|------------------------------------------------------------------------------------------------------|---------|
| <level> | Real | $(-5 \times \text{VerticalScale} - \text{OFFSet}) - (5 \times \text{VerticalScale} - \text{OFFSet})$ | 0       |

#### Remarks

This setting command is only valid when the channel source of the data cable is an analog channel.

#### Return Format

Query to return trigger level in scientific counting form.

#### Example

:TRIGger:SPI:DLEVel 0.16 /\* Set the trigger level to 160mV \*/

:TRIGger:SPI:DLEVel? /\* Query returns 1.600000 e-01\*/

### 3.12.73 :TRIGger:IIC:SDA:SOURce

#### Syntax

:TRIGger:IIC:SDA:SOURce <source>

:TRIGger:IIC:SDA:SOURce?

#### Description

Set or query the channel source of the I2C triggered data line.

#### Parameter

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel2 |

#### Remarks

The clock source and data source cannot be set to the same signal source at the same time.

#### Return Format

Query returns CHAN1 CHAN2, CHAN3, CHAN4, EXT.

#### Example

```
:TRIGger:IIC:SDA:SOURce CHANnel1      /* Set the data source to CH1*/
:TRIGger:IIC:SDA:SOURce?                /* Query returns CHAN1*/
```

### 3.12.74 :TRIGger:IIC:SCL:SOURce

#### Syntax

```
:TRIGger:IIC:SCL:SOURce <source>
:TRIGger:IIC:SCL:SOURce?
```

#### Description

Set or query the channel source of the clock triggered by I2C.

#### Parameter

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

#### Remarks

The clock source and data source cannot be set to the same signal source at the same time.

#### Return Format

Query returns CHAN1 CHAN2, CHAN3, CHAN4, EXT.

#### Example

```
:TRIGger:IIC:SCL:SOURce CHANnel2      /* Set the clock source to CH2*/
:TRIGger:IIC:SCL:SOURce?                /* Query returns CHAN2*/
```

### 3.12.75 :TRIGger:IIC:WHEN

#### Syntax

```
:TRIGger:IIC:WHEN <trig_type>
:TRIGger:IIC:WHEN?
```

**Description**

Set or query the triggering conditions for IIC triggering.

**Parameter**

| Name          | Type     | Range                                      | Default |
|---------------|----------|--------------------------------------------|---------|
| < trig_type > | Discrete | {START STOP RESTART MISSEDACK ADDRESS A&D} | START   |

**Remarks**

- START: Frame start
- STOP: End of frame.
- RESTART: Restart.
- MISSEDACK: Lost Confirmation
- ADDRESS: Finds the set address value and triggers it on the read/write bit.
- A&D: Simultaneously search for the set address value and data value, triggered when both "address" and "data" conditions are met.

**Return Format**

Query returns START STOP, RESTART, MISSEDACK, ADDRESS, or A&D.

**Example**

```
:TRIGger:IIC:WHEN START      /* Set the trigger condition to frame start */
:TRIGger:IIC:WHEN?           /* Query returns START */
```

### 3.12.76 :TRIGger:IIC:ADDRess

**Syntax**

```
:TRIGger:IIC:ADDRess <adr>
:TRIGger:IIC:ADDRess?
```

**Description**

Set or query the address value when the I2C trigger condition is an address or address data.

**Parameter**

| Name  | Type    | Range                                | Default |
|-------|---------|--------------------------------------|---------|
| <adr> | Integer | 0-2 <sup>n</sup> -1: 0-127 or 0-1023 | 0       |

**Remarks**

In expression 2<sup>n</sup>-1, N is the current address bit width.

**Return Format**

Query returns an integer, .

#### Example

```
:TRIGger:IIC:ADDRess 100      /* Set the address value to 100*/
:TRIGger:IIC:ADDRess?         /* Query returns 100*/
```

### 3.12.77 :TRIGger:IIC:AWIDth

#### Syntax

```
:TRIGger:IIC:AWIDth <bits>
:TRIGger:IIC:AWIDth?
```

#### Description

Set or query the address bit width when the I2C trigger condition is an address or address data.

#### Parameter

| Name   | Type     | Range  | Default |
|--------|----------|--------|---------|
| <bits> | Discrete | {7 10} | 7       |

#### Remarks

N/A

#### Return Format

Query returns 7 and 10.

#### Example

```
:TRIGger:IIC:AWIDth 10      /* Set the address bit width to 10*/
:TRIGger:IIC:AWIDth?         /* Query returns 10*/
```

### 3.12.78 :TRIGger:IIC:CLEVel

#### Syntax

```
:TRIGger:IIC:CLEVel <level>
:TRIGger:IIC:CLEVel?
```

#### Description

Set or query the triggering level of the clock line when I2C is triggered, with the unit consistent with the current amplitude unit.

**Parameter**

| Name    | Type | Range                                                                                                | Default |
|---------|------|------------------------------------------------------------------------------------------------------|---------|
| <level> | Real | $(-5 \times \text{VerticalScale} - \text{OFFSet}) - (5 \times \text{VerticalScale} - \text{OFFSet})$ | 0       |

**Remarks**

This setting command is only valid when the channel source of the clock line is an analog channel.

**Return Format**

Query to return trigger level in scientific counting form.

**Example**

```
:TRIGger:IIC:CLEVel 0.16      /* Set the trigger level to 160mV */
:TRIGger:IIC:CLEVel?          /* Query returns 1.600000 e-01*/
```

### 3.12.79 :TRIGger:IIC:DLEVel

**Syntax**

```
:TRIGger:IIC:DLEVel <level>
:TRIGger:IIC:DLEVel?
```

**Description**

Set or query the triggering level of the data line when I2C is triggered, with the unit consistent with the current amplitude unit.

**Parameter**

| Name    | Type | Range                                                                                                | Default |
|---------|------|------------------------------------------------------------------------------------------------------|---------|
| <level> | Real | $(-5 \times \text{VerticalScale} - \text{OFFSet}) - (5 \times \text{VerticalScale} - \text{OFFSet})$ | 0       |

**Remarks**

For VerticalScale, please refer to: CHANNEL<n>: SCALE command. For OFF Set, please refer to: CHANNEL<n>: OFF Set command. This setting command is only valid when the channel source of the data cable is an analog channel.

**Return Format**

Query to return trigger level in scientific counting form.

**Example**

```
:TRIGger:IIC:DLEVel 0.16      /* Set the trigger level to 160mV*/
```

:TRIGger:IIc:DLEVel? /\* Query returns 1.600000 e-01\*/

### 3.13 :MASK Commands

The MASK command is used to set and query the relevant parameters in pass/fail tests.

#### 3.13.1 :MASK:ENABLE

##### Syntax

:MASK:ENABle <bool>

:MASK:ENABle?

##### Description

Turn on or off the pass/fail testing function, or query the status of the pass/fail testing function.

##### Parameter

| Name   | Type | Range              | Default |
|--------|------|--------------------|---------|
| <bool> | Bool | {{1 ON }}{{0 OFF}} | 0 OFF   |

##### Remarks

The pass/fail test function is invalid in the following situations: horizontal time base mode in XY or Roll mode, slow scan mode (i.e. horizontal time base in YT mode, horizontal time base in 200ms/div or slower), and waveform recording.

##### Return Format

Query returns 1 or 0.

##### Example

:MASK:ENABle ON /\* Enable pass/fail testing function \*/

:MASK:ENABle? /\* Query returns 0\*/

#### 3.13.2 :MASK:SOURce

##### Syntax

:MASK:SOURce <source>

:MASK:SOURce?

##### Description

Set or query the measurement source for pass/fail tests.

#### Parameter

| Name     | Type     | Range                                 | Default  |
|----------|----------|---------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4} | CHANnel1 |

#### Remarks

This command can only set open channels and can send the: CHANnel<n>: DISPlay command to open the desired channel.

#### Return Format

Query returns CHAN1 CHAN2, CHAN3, or CHAN4.

#### Example

```
:MASK:SOURce CHANnel2      /* Set the measurement source for pass/fail tests to
CH2*/
```

```
:MASK:SOURce?              /* Query returns CHAN2*/
```

### 3.13.3 :MASK:OPERate

#### Syntax

```
:MASK:OPERate <oper>
```

```
:MASK:OPERate?
```

#### Description

Run or stop pass/fail tests, or query the running status of pass/fail tests.

#### Parameter

| Name   | Type     | Range      | Default |
|--------|----------|------------|---------|
| <oper> | Discrete | {RUN STOP} | RUN     |

#### Remarks

This command can only set open channels and can send the: CHANnel<n>: DISPlay command to open the desired channel.

#### Return Format

Before executing this command, you need to send the: MASK: ENABLE command to enable the pass/fail test function.

#### Example

```
:MASK:OPERate RUN          /* Run pass/fail test function */
```

```
:MASK:OPERate?             /* Query returns RUN */
```

### 3.13.4 :MASK:MDISplay

#### Syntax

:MASK:MDISplay <bool>

:MASK:MDISplay?

#### Description

When the pass/fail test is opened, turn on or off statistical information, or query the status of statistical information.

#### Parameter

| Name   | Type | Range             | Default |
|--------|------|-------------------|---------|
| <bool> | Bool | {{1 ON}}{{0 OFF}} | 0 OFF   |

#### Remarks

- Before executing this command, it is necessary to send the MASK: ENABLE command to enable the pass/fail test function.
- When the statistical information is opened, the test results shown in the following figure will be displayed in the upper right corner of the screen.
- Can send: MASK: PASSed?, : MASK:FAILed? And: MASK: TOTAL? Command query test results.

#### Return Format

Query returns 1 or 0.

#### Example

:MASK:MDISplay ON /\* Open Statistics \*/

:MASK:MDISplay? /\* Query returns 1\*/

### 3.13.5 :MASK:SOOutput

#### Syntax

:MASK:SOOutput <bool>

:MASK:SOOutput?

#### Description

Turn on or off stop output, or query the status of stop output.

#### Parameter

| Name | Type | Range | Default |
|------|------|-------|---------|
|------|------|-------|---------|

| Name   | Type | Range             | Default |
|--------|------|-------------------|---------|
| <bool> | Bool | {{1 ON}}{{0 OFF}} | 0 OFF   |

**Remarks**

- Open: When a failed waveform is detected, the oscilloscope will stop testing and enter the "STOP" state. At this point, the screen remains displaying the measurement results (if the display is turned on), and the rear panel [Trigger Out] (if enabled) only outputs one pulse.
- Off: Even if a failed waveform is detected, the oscilloscope will continue to test, and the test results on the screen are constantly updated. The Trigger Out on the rear panel will output pulses every time a failed waveform is detected.

**Return Format**

Query returns 1 and 0.

**Example**

```
:MASK:SOOutput ON      /* Turn on stop output */
:MASK:SOOutput?        /* Query returns 1*/
```

### 3.13.6 :MASK:OUTPut

**Syntax**

```
:MASK:OUTPut <bool>
:MASK:OUTPut?
```

**Description**

Turn on or off the sound prompt when testing fails, or check the status of the sound prompt.

**Parameter**

| Name   | Type | Range             | Default |
|--------|------|-------------------|---------|
| <bool> | Bool | {{1 ON}}{{0 OFF}} | 0 OFF   |

**Remarks**

- Off: When a failed waveform is detected, there is a display and output, but the buzzer does not sound an alarm, and the buzzer is turned off.
- Open: When a failed waveform is detected, there is a display and output, and the buzzer emits an audible alarm (independent of the sound switch status), and the buzzer opens.

**Return Format**

Query returns 1 and 0.

**Example**

```
:MASK:OUTPut ON      /* Turn on sound prompt (buzzer on)*/
:MASK:OUTPut?        /* Query returns 1*/
```

**3.13.7 :MASK:X****Syntax**

```
:MASK:X <x>
:MASK:X?
```

**Description**

Set or query the horizontal adjustment parameters in the pass/fail test rules, with a default unit of div.

**Parameter**

| Name | Type | Range                                                     | Default |
|------|------|-----------------------------------------------------------|---------|
| <x>  | Real | 0.01 to 2, with a step of 0.01 within the range of values | 0.24    |

**Remarks**

N/A

**Return Format**

Query returns the current horizontal adjustment parameter in scientific counting form.

**Example**

```
:MASK:X 0.28      /* Set the horizontal adjustment parameter to 0.28div */
:MASK:X?          /* Query returned 2.800000 e-01*/
```

**3.13.8 :MASK:Y****Syntax**

```
:MASK:Y <y>
:MASK:Y?
```

**Description**

Set or query the vertical adjustment parameters in the pass/fail test rules, with a default unit of div.

**Parameter**

| Name | Type | Range                                                     | Default |
|------|------|-----------------------------------------------------------|---------|
| <y>  | Real | 0.04 to 2, with a step of 0.01 within the range of values | 0.04    |

**Remarks**

N/A

**Return Format**

Query returns the current vertical adjustment parameter in scientific counting form.

**Example**

```
:MASK:Y 0.36          /* Set the vertical adjustment parameter to 0.36div */
:MASK:Y?              /* Query returned 3.600000 e-01*/
```

### 3.13.9 :MASK:CREate

---

**Syntax**

:MASK:CREate

**Description**

Create a pass/fail test rule by adjusting the parameters horizontally and vertically based on the current settings.

**Parameter**

N/A

**Remarks**

This command is only valid when the pass/fail test function is turned on (: MASK: ENABLE) and not running (: MASK: OPERate).

**Return Format**

N/A

**Example**

N/A

### 3.13.10 :MASK:PASSEd?

---

**Syntax**

:MASK:PASSed?

**Description**

Query the number of frames passed when passing the test.

**Parameter**

N/A

**Remarks**

N/A

**Return Format**

Query returns an integer.

**Example**

N/A

### 3.13.11 :MASK:FAILed?

---

**Syntax**

:MASK:FAILed?

**Description**

The number of frames that failed when querying failed tests.

**Parameter**

N/A

**Remarks**

N/A

**Return Format**

Query returns an integer.

**Example**

N/A

### 3.13.12 :MASK:TOTal?

---

**Syntax**

:MASK:TOTal?

**Description**

Query the total number of frames that passed/failed the test.

**Parameter**

N/A

**Remarks**

N/A

**Return Format**

Query returns an integer.

**Example**

N/A

### 3.13.13 :MASK:RESet

---

**Syntax**

:MASK:RESet

**Description**

Reset the number of frames passed, failed, and total frames in the pass/fail test.

**Parameter**

N/A

**Remarks**

N/A

**Return Format**

N/A

**Example**

N/A

## 3.14 :LA Commands

---

The LA command is used to perform related operations on digital channels.

### 3.14.1 :LA:POD<n>:DISPlay

#### Syntax

:LA:POD<n>:DISPlay <bool>

:LA:POD<n>:DISPlay?

#### Description

Turn on or off the specified default channel group, or query the status of the specified default channel group.

#### Parameter

| Name   | Type     | Range            | Default |
|--------|----------|------------------|---------|
| <n>    | Discrete | {1 2 3 4}        | --      |
| <bool> | Discrete | {{1 ON}}{0 OFF}} | OFF     |

#### Remarks

A set of digital channels consists of 12 digits, while 34 represents a set of digital channels.

#### Return Format

Query returns 1 or 0.

#### Example

:LA:POD1:DISPlay 1 /\* Open POD1 (D0 to D4)\*/

:LA:POD1:DISPlay? /\* Query returns 1\*/

### 3.14.2 :LA:POD<n>:THReshold

#### Syntax

:LA:POD<n>:THReshold <thre>

:LA:POD<n>:THReshold?

#### Description

Set or query the threshold for the specified default channel group, with a default unit of V.

#### Parameter

| Name | Type     | Range     | Default |
|------|----------|-----------|---------|
| <n>  | Discrete | {1 2 3 4} | --      |

| Name   | Type | Range       | Default |
|--------|------|-------------|---------|
| <thre> | Real | -7.0V-+7.0V | --      |

**Remarks**

4 default channel groups: D1~D4.

**Return Format**

Query returns the current threshold of a specified channel group in scientific counting form.

**Example**

```
:LA:POD1:THReshold 3.3      /* Set the threshold for POD1 (D1 to D4) to 3.3V */
:LA:POD1:THReshold?          /* Query returns 33000000 e+00*/
```

## 3.15 **[:SOURce] Commands**

The [: SOURCE [<n>] command is used to set the parameters related to the built-in signal source< n> Take 1 to indicate the corresponding built-in signal source channel. If<n>or: SOURCE [<n>] is omitted, it defaults to operating on signal source 1. Only machines with signal source function are supported for setting.

### 3.15.1 **[:SOURce[<n>]]:OUTPut[<n>][:STATe]**

**Syntax**

```
[:SOURce[<n>]]:OUTPut[<n>][:STATe] <bool> [:SOURce[<n>]]:OUTPut[<n>][:STATe]?
```

**Description**

Turn on or off the output of the specified signal source channel, or query the output status of the specified signal source channel.

**Parameter**

| Name   | Type     | Range            | Default                                     |
|--------|----------|------------------|---------------------------------------------|
| <n>    | Discrete | {1 2}            | When omitted, default to source 1 operation |
| <bool> | Bool     | {{1 ON}}{0 OFF}} | 0 OFF                                       |

**Remarks**

N/A

**Return Format**

Query returns OFF or ON.

**Example**

```
:SOURce1:OUTPut 1      /* Open the output of source 1 */
:SOURce1:OUTPut?       /* Query returns ON */
```

### 3.15.2 **[:SOURce[<n>]]:OUTPut[<n>]:IMPedance**

**Syntax**

```
[:SOURce[<n>]]:OUTPut[<n>]:IMPedance <impedance>
[:SOURce[<n>]]:OUTPut[<n>]:IMPedance?
```

**Description**

Set or query the impedance of the specified signal source channel.

**Parameter**

| Name        | Type     | Range        | Default                                     |
|-------------|----------|--------------|---------------------------------------------|
| <n>         | Discrete | {1 2}        | When omitted, default to source 1 operation |
| <impedance> | Discrete | {OMEG FIFTy} | OMEG                                        |

**Remarks**

OMEG: high resistance;  
FIFTy: 50  $\Omega$ ;

**Return Format**

Query returns OMEG or FIFT.

**Example**

```
:SOURce1:OUTPut:IMPedance FIFTy      /* Set the output impedance of source 1 to
50  $\Omega$ */
:SOURce1:OUTPut:IMPedance?           /* Query returns FIFT*/
```

### 3.15.3 **[:SOURce[<n>]]:FREQuency[:FIXed]**

**Syntax**

```
[:SOURce[<n>]]:FREQuency[:FIXed] <frequency>
[:SOURce[<n>]]:FREQuency[:FIXed]?
```

**Description**

If the modulation is not turned on for the specified signal source channel, this command

is used to set or query the output frequency of the specified signal source channel; If modulation is enabled on the specified signal source channel, this command is used to set or query the carrier frequency of the specified signal source channel, with the default unit being Hz.

#### Parameter

| Name        | Type     | Range                                                                                                                                               | Default                                     |
|-------------|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------|
| <n>         | Discrete | {1 2}                                                                                                                                               | When omitted, default to source 1 operation |
| <frequency> | Real     | Sine wave: 0.1Hz to 25MHz<br>Square wave: 0.1Hz to 15MHz<br>Pulse: 0.1Hz to 1MHz<br>Sawtooth wave: Any wave from 0.1Hz to 100kHz:<br>0.1Hz to 10MHz | 1kHz                                        |

#### Remarks

N/A

#### Return Format

Query returns frequency values in scientific counting form, such as 2.0000000 e+05.

#### Example

```
:SOURce1:FREQuency 1000          /* Set the output frequency of source 1 to
1kHz */
:SOURce1:FREQuency?              /* Query returned 1.0000000 e+03*/
```

### 3.15.4 [:SOURce[<n>]]:PHASe[:ADJust]

#### Syntax

```
[[:SOURce[<n>]]:PHASe[:ADJust] <phase>
[:SOURce[<n>]]:PHASe[:ADJust]?
```

#### Description

Set or query the starting phase of the specified signal source channel signal, with the default unit being degrees (degrees).

#### Parameter

| Name | Type     | Range | Default                                     |
|------|----------|-------|---------------------------------------------|
| <n>  | Discrete | {1 2} | When omitted, default to source 1 operation |

| Name    | Type | Range | Default |
|---------|------|-------|---------|
| <phase> | Real | 0-360 | 0       |

**Remarks**

N/A

**Return Format**

Query returns the starting phase value in scientific counting form, such as 1.000000 e+00.

**Example**

```
:SOURce1:PHASe 90          /* Set the starting phase of source 1 to 90 °*/
:SOURce1:PHASe?            /* Query returned 9.000000 e+01*/
```

### 3.15.5 **[:SOURce[<n>]]:FUNCTion[:SHAPE]**

**Syntax**

```
[:SOURce[<n>]]:FUNCTion[:SHAPE] <wave>
[:SOURce[<n>]]:FUNCTion[:SHAPE]?
```

**Description**

If the specified signal source channel is not modulated, this command is used to select or query the output signal waveform. If the specified signal source channel has modulation enabled, this command is used to select or query the modulated carrier. At this time, if PULSE is selected NOISe or DC, the modulation function will automatically turn off.

**Parameter**

| Name   | Type     | Range                                                                                        | Default                                     |
|--------|----------|----------------------------------------------------------------------------------------------|---------------------------------------------|
| <n>    | Discrete | {1 2}                                                                                        | When omitted, default to source 1 operation |
| <wave> | Discrete | {SINusoid SQUare RAMP PULSe DC NOISe SINC EXPRise EXPFall ECG GAUSSs LOrentz HAVersine ARB } | SINusoid                                    |

**Remarks**

ARB: Any wave

The signal source provides 14 types of built-in waves: sine wave, square wave, sawtooth wave, pulse, DC, noise Sinc, Exponential increase, exponential decrease, electrocardiogram, Gaussian, Lorentz, half vector, and arbitrary wave.

**Return Format**

Query returns

SIN,SQU,RAMP,PULS,DC,NOIS,SINC,EXPR,EXPF,ECG,GAUS,LOR,HAV,ARB.

**Example**

```
:SOURce1:FUNCtion SQUare      /* Set the output waveform of source 1 to square
wave */
```

```
:SOURce1:FUNCtion?           /* Query returns SQU */
```

### 3.15.6 [:SOURce[<n>]]:FUNCtion:RAMP:SYMMetry

**Syntax**

```
[:SOURce[<n>]]:FUNCtion:RAMP:SYMMetry <val>
```

```
[:SOURce[<n>]]:FUNCtion:RAMP:SYMMetry?
```

**Description**

Set or query the symmetry of the sawtooth wave output by the specified signal source channel, that is, the percentage of the period that the sawtooth wave waveform occupies during the rising period.

**Parameter**

| Name  | Type     | Range | Default                                     |
|-------|----------|-------|---------------------------------------------|
| <n>   | Discrete | {1 2} | When omitted, default to source 1 operation |
| <val> | Real     | 0-100 | 50                                          |

**Remarks**

N/A

**Return Format**

Query returns the current symmetry in scientific counting form, such as 5.0000000 e+01.

**Example**

```
:SOURce1:FUNCtion:RAMP:SYMMetry 50 /* Set the symmetry of source 1 sawtooth
wave to 50%*/
```

```
:SOURce1:FUNCtion:RAMP:SYMMetry?    /* Query returns 5.0000000 e+01*/
```

### 3.15.7 [[:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate][:AMPLitude]

#### Syntax

```
[[:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate][:AMPLitude] <Amplitude>
[:SOURce[<n>]]:VOLTage[:LEVel][:IMMediate][:AMPLitude]?
```

#### Description

Set or query the amplitude of the output signal of the specified signal source channel, with a default unit of Vpp.

#### Parameter

| Name        | Type     | Range                                                                                               | Default                                     |
|-------------|----------|-----------------------------------------------------------------------------------------------------|---------------------------------------------|
| <n>         | Discrete | {1 2}                                                                                               | When omitted, default to source 1 operation |
| <Amplitude> | Real     | Related to the current output impedance<br>High resistance: 10mVpp to 7Vpp<br>50 Ω: 5mVpp to 3.5Vpp | 500mVpp                                     |

#### Remarks

Send the [: Source [<n>]: OUTput [<n>]: IMPedance command to set the output impedance.

#### Return Format

Query returns the amplitude value in scientific counting form, such as 1.0000000 e+00.

#### Example

```
:SOURce1:VOLTage 2          /* Set the output amplitude of source 1 to 2V */
:SOURce1:VOLTage?           /* Query returns 2.0000000 e+00*/
```

### 3.15.8 [[:SOURce[<n>]]: VOLTage[:LEVel] [:IMMediate]:OFFSet

#### Syntax

```
[[:SOURce[<n>]]:VOLTage[:LEVel] [:IMMediate]:OFFSet <offset>
[:SOURce[<n>]]:VOLTage[:LEVel] [:IMMediate]:OFFSet?
```

#### Description

Set or query the DC offset of the output signal of the specified signal source channel, with the default unit being VDC.

#### Parameter

| Name     | Type     | Range                                                                                                                            | Default                                     |
|----------|----------|----------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------|
| <n>      | Discrete | {1 2}                                                                                                                            | When omitted, default to source 1 operation |
| <offset> | Real     | It is related to the current output impedance and amplitude.<br>High resistance: -3.5V to 3.5V;<br>50 $\Omega$ : -1.75V to 1.75V | 0VDC                                        |

#### Remarks

Send the [: Source [<n>]: OUTput [<n>]: IMPedance command to set the output impedance.

Send the [: Source [<n>]: VOLTage [: LEVel] [: IMMEDIATE] [: AMPNumber] command to set the current amplitude

.

#### Return Format

Query returns the DC offset value in scientific counting form, such as 1.0000000 e+00.

#### Example

```
:SOURce1:VOLTage:OFFSet 0.5      /* Set the DC offset of source 1 to 500mVDC. */
:SOURce1:VOLTage:OFFSet?         /* Query returns 5.0000000e-01*/
```

### 3.15.9 [:SOURce[<n>]]:PULSe:DCYCLE

#### Syntax

```
[:SOURce[<n>]]:PULSe:DCYCLE <percent>
[:SOURce[<n>]]:PULSe:DCYCLE?
```

#### Description

Set or query the duty cycle of the pulse output by the specified signal source channel, that is, the proportion of high level in one pulse cycle.

#### Parameter

| Name      | Type     | Range | Default                                     |
|-----------|----------|-------|---------------------------------------------|
| <n>       | Discrete | {1 2} | When omitted, default to source 1 operation |
| <percent> | Real     | 1-99  | 50                                          |

**Remarks**

N/A

**Return Format**

Query returns the current duty cycle in scientific counting form, such as 5.0000000 e+01.

**Example**

```
:SOURce1:PULSe:DCYClE 50      /* Set the duty cycle of source 1 pulse to 50%*/
:SOURce1:PULSe:DCYClE?        /* Query returns 5.0000000 e+01*/
```

### 3.15.10 [:SOURce[<n>]]:MOD[:STATe]

**Syntax**

```
[:SOURce[<n>]]:MOD[:STATe] <bool>
[:SOURce[<n>]]:MOD[:STATe]?
```

**Description**

Turn on or off the modulation of the specified signal source channel, or query the modulation status of the specified signal source channel.

**Parameter**

| Name   | Type     | Range            | Default                                     |
|--------|----------|------------------|---------------------------------------------|
| <n>    | Discrete | {1 2}            | When omitted, default to source 1 operation |
| <bool> | Bool     | {{1 ON}}{0 OFF}} | 0 OFF                                       |

**Remarks**

Support AM FM, PM, PWM.

Any wave of sine wave, square wave, sawtooth wave, built-in wave, or non DC signal can be used as a carrier.

**Return Format**

Query returns OFF or ON.

**Example**

```
:SOURce1:MOD ON      /* Turn on the modulation function of source 1 */
:SOURce1:MOD?        /* Query returns ON */
```

### 3.15.11 [:SOURce[<n>]]:MOD:TYPE

#### Syntax

```
[:SOURce[<n>]]:MOD:TYPE <type>
[:SOURce[<n>]]:MOD:TYPE?
```

#### Description

Set or query the modulation type of the specified signal source channel.

#### Parameter

| Name   | Type     | Range          | Default                                     |
|--------|----------|----------------|---------------------------------------------|
| <n>    | Discrete | {1 2}          | When omitted, default to source 1 operation |
| <type> | Discrete | {AM FM PM PWM} | AM                                          |

#### Remarks

AM: Amplitude modulation, which means that the amplitude of the carrier wave changes with the amplitude of the modulated wave.

FM: Frequency modulation, which means that the frequency of the carrier wave changes with the amplitude of the modulated wave.

PM: Amplitude modulation, which means that the phase of the carrier wave changes with the amplitude of the modulated wave.

PWM: Frequency modulation, which means that the duty cycle of the carrier varies with the amplitude of the modulated wave.

Any wave of sine wave, square wave, sawtooth wave, built-in wave, or non DC signal can be used as a carrier.

You can choose sine wave, square wave, triangular wave, or noise as the modulation waveform.

#### Return Format

Query returns AM FM, PM, or PWM.

#### Example

```
:SOURce1:MOD:TYPE AM      /* Set the modulation type of source 1 to AM */
:SOURce1:MOD:TYPE?        /* Query returns AM */
```

### 3.15.12 [:SOURce[<n>]]:MOD:AM:INTernal:FREQuency

#### Syntax

```
[:SOURce[<n>]]:MOD:AM:INTernal:FREQuency <frequency>
[:SOURce[<n>]]:MOD:AM:INTernal:FREQuency?
```

```

[:SOURce[<n>]]:MOD:FM:INTernal:FREQuency <frequency>
[:SOURce[<n>]]:MOD:FM:INTernal:FREQuency?
[:SOURce[<n>]]:MOD:PM:INTernal:FREQuency <frequency>
[:SOURce[<n>]]:MOD:PM:INTernal:FREQuency?
[:SOURce[<n>]]:MOD:PWM:INTernal:FREQuency <frequency>
[:SOURce[<n>]]:MOD:PWM:INTernal:FREQuency?

```

### Description

Set or query the specified signal source channel AM The modulation wave frequency of FM, PM, or PWM modulation, with a default unit of Hz.

### Parameter

| Name        | Type     | Range     | Default                                     |
|-------------|----------|-----------|---------------------------------------------|
| <n>         | Discrete | {1 2}     | When omitted, default to source 1 operation |
| <frequency> | Real     | 1Hz-50kHz | 1kHz                                        |

### Remarks

The MOD: TYPE command can be sent to set the modulation type.

AM: Amplitude modulation, which means that the amplitude of the carrier wave changes with the amplitude of the modulated wave.

FM: Frequency modulation, which means that the frequency of the carrier wave changes with the amplitude of the modulated wave.

PM: Amplitude modulation, which means that the phase of the carrier wave changes with the amplitude of the modulated wave.

PWM: Frequency modulation, which means that the duty cycle of the carrier varies with the amplitude of the modulated wave.

You can choose sine wave, square wave, triangular wave, or noise as modulation waves to send

[ : Source [<n>]: The MOD: AM: Internal: FUSION command or [ : Source [<n>]: MOD: FM: Internal: FUSION command can be selected.

### Return Format

Query returns an integer.

### Example

```

:SOURce1:MOD:AM:INTernal:FREQuency 100      /* Set the AM modulation wave
frequency of source 1 to 100Hz */
:SOURce1:MOD:AM:INTernal:FREQuency?          /* Query returns 100*/

```

### 3.15.13 [:SOURce[<n>]]:MOD:AM[:DEPTTh]

#### Syntax

```
[:SOURce[<n>]]:MOD:AM[:DEPTTh] <depth>
[:SOURce[<n>]]:MOD:AM[:DEPTTh]?
```

#### Description

Set or query the AM modulation depth of the specified signal source channel. The modulation depth represents the strength of amplitude modulation, expressed as a percentage.

#### Parameter

| Name    | Type     | Range | Default                                     |
|---------|----------|-------|---------------------------------------------|
| <n>     | Discrete | {1 2} | When omitted, default to source 1 operation |
| <depth> | Real     | 0-120 | 100                                         |

#### Remarks

When the modulation depth is 0%, the output amplitude is half of the carrier signal amplitude. When the modulation depth is 100%, the output amplitude is equal to the carrier signal amplitude. When the modulation depth is greater than 100%, envelope distortion will occur, which must be avoided in actual circuits. At this time, the output of the instrument will not exceed 5 Vpp (with a load of 50 Ω).

#### Return Format

Query returns an integer.

#### Example

```
:SOURce1:MOD:AM 80      /* Set the AM modulation depth of source 1 to 80%*/
:SOURce1:MOD:AM?        /* Query returns 80*/
```

### 3.15.14 [:SOURce[<n>]]:MOD:FM[:DEVlation]

#### Syntax

```
[:SOURce[<n>]]:MOD:FM[:DEVlation] <dev> [:SOURce[<n>]]:MOD:FM[:DEVlation]?
```

#### Description

Set or query the frequency offset of the specified signal source channel FM modulation, with a default unit of Hz.

#### Parameter

| Name  | Type     | Range                                      | Default                                     |
|-------|----------|--------------------------------------------|---------------------------------------------|
| <n>   | Discrete | {1 2}                                      | When omitted, default to source 1 operation |
| <dev> | Real     | 0.1Hz to the current set carrier frequency | 1kHz                                        |

#### Remarks

You can send the [: Source [<n>]: FREQuency [: FIXed] command to set the carrier frequency, and send the [: Source [<n>]: MOD: FM: Internal: FREQuency command to set the FM modulation frequency.

When the modulation amplitude reaches its maximum value, the frequency of the carrier increases by "frequency offset". When the modulation amplitude reaches its minimum value, the frequency of the carrier decreases by "frequency offset".

The frequency offset of FM modulation can only be set when the FM modulation type is currently selected for the specified signal source channel.

#### Return Format

Query returns an integer.

#### Example

```
:SOURce1:MOD:FM 100          /* Set the frequency offset of source 1FM modulation
to 100Hz */
:SOURce1:MOD:FM?             /* Query returns 100*/
```

### 3.15.15 [:SOURce[<n>]]:MOD:PM[:DEPT h]

#### Syntax

```
[:SOURce[<n>]]:MOD:PM[:DEPT h] <depth>
[:SOURce[<n>]]:MOD:PM[:DEPT h]?
```

#### Description

Set or query the specified signal source channel PM modulation depth. The modulation depth represents the strength of amplitude modulation, expressed as a percentage.

#### Parameter

| Name    | Type     | Range | Default                                     |
|---------|----------|-------|---------------------------------------------|
| <n>     | Discrete | {1 2} | When omitted, default to source 1 operation |
| <depth> | Real     | 0-120 | 100                                         |

#### Remarks

When the modulation depth is 0%, the output amplitude is half of the carrier signal amplitude. When the modulation depth is 100%, the output amplitude is equal to the carrier signal amplitude. When the modulation depth is greater than 100%, envelope distortion will occur, which must be avoided in actual circuits. At this time, the output of the instrument will not exceed 5 Vpp (with a load of 50  $\Omega$ ).

#### Return Format

Query returns an integer.

#### Example

```
:SOURce1:MOD:PM 80      /* Set the PM modulation depth of source 1 to 80%*/
:SOURce1:MOD:PM?        /* Query returns 80*/
```

### 3.15.16 [:SOURce[<n>]]:MOD:PWM[:DEPT h]

#### Syntax

```
[:SOURce[<n>]]:MOD:PWM[:DEPT h] <depth>
[:SOURce[<n>]]:MOD:PWM[:DEPT h]?
```

#### Description

Set or query the PWM modulation depth of the specified signal source channel. The modulation depth represents the strength of amplitude modulation, expressed as a percentage.

#### Parameter

| Name    | Type     | Range | Default                                     |
|---------|----------|-------|---------------------------------------------|
| <n>     | Discrete | {1 2} | When omitted, default to source 1 operation |
| <depth> | Real     | 0-120 | 100                                         |

#### Remarks

When the modulation depth is 0%, the output amplitude is half of the carrier signal amplitude. When the modulation depth is 100%, the output amplitude is equal to the carrier signal amplitude. When the modulation depth is greater than 100%, envelope distortion will occur, which must be avoided in actual circuits. At this time, the output of the instrument will not exceed 5 Vpp (with a load of 50  $\Omega$ ).

#### Return Format

Query returns an integer.

#### Example

```
:SOURce1:MOD:PWM 80      /* Set the PWM modulation depth of source 1 to
80%*/
```

:SOURce1:MOD:PWM? /\* Query returns 80\*/

### 3.15.17 [:SOURce[<n>]]:APPLY

[:SOURce[<n>]]:APPLY:NOISe

[:SOURce[<n>]]:APPLY:PLUSe

[:SOURce[<n>]]:APPLY:RAMP

[:SOURce[<n>]]:APPLY:SINusoid

[:SOURce[<n>]]:APPLY:SQUare

[:SOURce[<n>]]:APPLY:USER

#### Syntax

```
[:SOURce[<n>]]:APPLY:NOISe [<amp>[, <offset>]]
[:SOURce[<n>]]:APPLY:PULSe [<freq>[, <amp>[, <offset>[, <phase>]]]]
[:SOURce[<n>]]:APPLY:RAMP [<freq>[, <amp>[, <offset>[, <phase>]]]]
[:SOURce[<n>]]:APPLY:SINusoid [<freq>[, <amp>[, <offset>[, <phase>]]]]
[:SOURce[<n>]]:APPLY:SQUare [<freq>[, <amp>[, <offset>[, <phase>]]]]
[:SOURce[<n>]]:APPLY:USER [<freq>[, <amp>[, <offset>[, <phase>]]]]
```

#### Description

Configure the specified signal source channel to output a signal with the specified waveform and parameters.

#### Parameter

| Name   | Type     | Range                                                                                                                                          | Default                                     |
|--------|----------|------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------|
| <n>    | Discrete | {1 2}                                                                                                                                          | When omitted, default to source 1 operation |
| <freq> | Real     | Sine wave: 0.1Hz to 25MHz<br>Square wave: 0.1Hz to 15MHz<br>Pulse: 0.1Hz to 1MHz<br>Sawtooth wave: 0.1Hz to 100kHz<br>Any wave: 0.1Hz to 10MHz | 1kHz                                        |
| <amp>  | Real     | High resistance related to the current set output impedance:<br>10mVpp to 7Vpp;                                                                | 5Vpp                                        |

| Name     | Type | Range                                                                                                              | Default     |
|----------|------|--------------------------------------------------------------------------------------------------------------------|-------------|
|          |      | 50 $\Omega$ : 5mVpp to 3.5Vpp                                                                                      |             |
| <offset> | Real | High resistance related to current output impedance and amplitude: -3.5V to 3.5V;<br>50 $\Omega$ : -1.75V to 1.75V | 0 $\mu$ VDC |
| <phase>  | Real | 0°-360°                                                                                                            | 0°          |

**Remarks**

This series of commands is used to select waveform shapes. NOISe: Noise PULSE: Pulse RAMP: Sawtooth wave

SINusoid: sine wave SQUARE: square wave EXT: Any wave

<freq>: Set the frequency of the specified waveform (noise does not have this parameter), with a default unit of Hz< Amp>: Set the amplitude of the specified waveform, with a default unit of Vpp< Offset>: Set the DC offset of the specified waveform, with the default unit being VDC< Phase>: Set the starting phase of the specified waveform (noise does not have this parameter), with a default unit of degrees (degrees).

This series of commands allows users to omit one or more Parameters. When all parameters are omitted, this series of commands only configure the specified signal source channel to the specified waveform, without modifying the corresponding parameters.

The four parameters<freq>, <amp>, <offset>, and<phase>have a sequential nature, and this series of commands cannot omit the preceding parameters and directly set the following parameters, that is, you cannot omit<freq>and directly set<amp>.

**Return Format**

N/A

**Example**

N/A

## 3.16 :MATH Commands

The MATH command is used to set various calculation functions for waveform between channels.

### 3.16.1 :MATH:DISPlay

**Syntax**

:MATH:DISPlay <bool>

:MATH:DISPlay?

**Description**

Turn on or off the mathematical operation function or query the status of the mathematical operation function.

**Parameter**

| Name   | Type | Range            | Default |
|--------|------|------------------|---------|
| <bool> | Bool | {{1 ON}}{0 OFF}} | 0 OFF   |

**Remarks**

N/A

**Return Format**

Query returns 1 or 0.

**Example**

```
:MATH:DISPlay ON      /* Open mathematical operation function */
:MATH:DISPlay?         /* Query returns 1*/
```

### 3.16.2 :MATH:OPERator

**Syntax**

```
:MATH:OPERator <opt>
:MATH:OPERator?
```

**Description**

Set or query operators for mathematical operations.

**Parameter**

| Name  | Type     | Range                                                                                                       | Default |
|-------|----------|-------------------------------------------------------------------------------------------------------------|---------|
| <opt> | Discrete | {ADD SUBTract MULTIply DIVision FFT AND ORXOR NOT INTG DIFF SQRT LG LN EXP ABS LPASs HPASs BPASs BStOp AXB} | ADD     |

**Remarks**

When REF is selected as the Parameter for the: MATH: SOURce1 and/or: MATH: SOURce2 commands, this command is used to set the operator for the outer layer operation of the combination operation. The range of<opt>is {ADD | SUBtract | Multiply | Division | FFT | AND | OR | XOR | NOT | INTG | DIFF | SQRT | LG | LN | EXP | ABS | LPASs | HPASs | BPASs | BStOp | AXB}.

**Return Format**

Query returns

ADD,SUBT,MULT,DIV,FFT,AND,OR,XOR,NOT,INTG,DIFF,SQRT,LG,LN,EXP,ABS,LPA,HPA,BPA,BST,AXB.

**Example**

```
:MATH:OPERator INTG      /* Set mathematical operators as integral operations */
:MATH:OPERator?           /* Query returns INTG */
```

**3.16.3 :MATH:SOURce1****Syntax**

```
:MATH:SOURce1 <src>
:MATH:SOURce1?
```

**Description**

Set or query the source A for algebraic operations.

**Parameter**

| Name  | Type     | Range                                     | Default  |
|-------|----------|-------------------------------------------|----------|
| <src> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 REF} | CHANnel1 |

**Remarks**

For algebraic operations, this command is used to set source A.

For function operations, only use this command to set the signal source.

For combinatorial operations, when the outer operation is an algebraic operation, this command is used to set the source A of the outer operation, and the range of<src>is {CHANnel1 | CHANnel2 | CHANnel3 | CHANnel4 | REF}; When the outer operation is a function operation, only use this command to set the source of the outer operation, <src>can only be set to REF.

Note: For combination operations with algebraic operations in the outer layer, at least one of the sources A and B in the outer layer operation needs to be selected as REF.

**Return Format**

Query returns CHAN1 CHAN2, CHAN3, CHAN4, REF.

**Example**

```
:MATH:SOURce1 CHANnel3    /* Set the source A of algebraic operations to CH3*/
:MATH:SOURce1?            /* Query returns CHAN3*/
```

### 3.16.4 :MATH:SOURce2

#### Syntax

:MATH:SOURce2 <src>

:MATH:SOURce2?

#### Description

Set or query the source B for the outer layer operation of algebraic/combinatorial operations.

#### Parameter

| Name  | Type     | Range                                     | Default  |
|-------|----------|-------------------------------------------|----------|
| <src> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 REF} | CHANnel1 |

#### Remarks

This command is only applicable to algebraic operations (containing two sources) and combination operations where the outer layer operation is an algebraic operation.

For combination operations with algebraic operations, this command is used to set the source B of the outer operation.

Note: For combination operations with algebraic operations in the outer layer, at least one of the sources A and B in the outer layer operation needs to be selected as REF.

#### Return Format

Query returns CHAN1 CHAN2, CHAN3, CHAN4, REF.

#### Example

```
:MATH:SOURce2 CHANnel3    /* Set the source B of algebraic operations to CH3*/
:MATH:SOURce2?             /* Query returns CHANnel3*/
```

### 3.16.5 :MATH:SCALE

#### Syntax

:MATH:SCALE <scale>

:MATH:SCALE?

#### Description

Set or query the vertical range of the operation result, with units related to the currently selected operator and the unit selected by the source.

**Parameter**

| Name    | Type | Range                                                | Default |
|---------|------|------------------------------------------------------|---------|
| <scale> | Real | The maximum range is 1p to 5T (in steps of 1-2 to 5) | 1.00V   |

**Remarks**

The adjustable range of the vertical gear is related to the currently selected operator and the gear of the signal channel. For integral and differential operations, it is also related to the current horizontal time base.

**Return Format**

Retrieve the vertical range of the current calculation result in scientific counting form.

**Example**

```
:MATH:SCALE 2      /* Set the vertical gear to 2V*/
:MATH:SCALE?       /* Query returns 2.000000 e+00*/
```

### 3.16.6 :MATH:OFFSet

**Syntax**

```
:MATH:OFFSet <offs>
:MATH:OFFSet?
```

**Description**

Set or query the vertical offset of the operation result, with units related to the currently selected operator and the unit selected by the source.

**Parameter**

| Name   | Type | Range                                                                                                                                                       | Default |
|--------|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| <offs> | Real | It is related to the vertical gear of the calculation result (-1000 x MathVerticalScale) to (1000 x MathVerticalScale), with a step of MathVerticalScale/50 | 0.00V   |

**Remarks**

MathVerticalScale is the vertical gear of the calculation result, which can be set by the: MATH: SCALE command.

**Return Format**

Query returns the vertical offset of the current operation result in scientific counting form.

**Example**

```
:MATH:OFFSet 2      /* Set the vertical offset to 2V */
:MATH:OFFSet?       /* Query returns 2.000000 e+00*/
```

**3.16.7 :MATH:FFT:SOURce****Syntax**

```
:MATH:FFT:SOURce <src>
:MATH:FFT:SOURce?
```

**Description**

Set or query the source of FFT operations/filters.

**Parameter**

| Name  | Type     | Range                                 | Default  |
|-------|----------|---------------------------------------|----------|
| <src> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4} | CHANnel1 |

**Remarks**

N/A

**Return Format**

Query returns CHANnel1 CHANnel2, CHANnel3, or CHANnel4.

**Example**

```
:MATH:FFT:SOURce CHANnel1    /* Set the source of FFT operation to CH1*/
:MATH:FFT:SOURce?            /* Query returns Chanel1*/
```

**3.16.8 :MATH:FFT:WINDow****Syntax**

```
:MATH:FFT:WINDow <window>
:MATH:FFT:WINDow?
```

**Description**

Set or query window functions for FFT operations.

**Parameter**

| Name       | Type     | Range                             | Default   |
|------------|----------|-----------------------------------|-----------|
| < window > | Discrete | {RECTangle HANNing HAMMING BLACKm | RECTangle |

| Name | Type | Range                   | Default |
|------|------|-------------------------|---------|
|      |      | an TR angle   FLAT top} |         |

**Remarks**

The use of window functions can effectively reduce spectral leakage effects. Each window function is suitable for measuring different waveforms and needs to be selected based on the measured waveform and its characteristics.

**Return Format**

Query returns RECT,HANN,HAMM,BLAC,TRI,FLAT.

**Example**

```
:MATH:FFT:WINDow BLACKman      /* Set the window function for FFT operation to
Blackman */
:MATH:FFT:WINDow?                /* Query returns BLAC */
```

### 3.16.9 :MATH:FFT:UNIT

**Syntax**

```
:MATH:FFT:UNIT <unit>
:MATH:FFT:UNIT?
```

**Description**

Set or query the vertical unit of FFT operation results.

**Parameter**

| Name   | Type     | Range     | Default |
|--------|----------|-----------|---------|
| <unit> | Discrete | {VRMS DB} | VRMS    |

**Remarks**

N/A

**Return Format**

Query returns VRMS or DB.

**Example**

```
:MATH:FFT:UNIT VRMS      /* Set the vertical unit of FFT operation result to VRMS */
:MATH:FFT:UNIT?          /* Query returns VRMS */
```

### 3.16.10 :MATH:FFT:HSCale

#### Syntax

:MATH:FFT:HSCale <hsc>

:MATH:FFT:HSCale?

#### Description

Set or query the frequency range of FFT operation results, with a default unit of Hz.

#### Parameter

| Name  | Type | Range                   | Default |
|-------|------|-------------------------|---------|
| <hsc> | Real | Please refer to Remarks | 250MHz  |

#### Remarks

<hsc> can be set to 1/1000, 1/400, 1/200, 1/100, 1/40, or 1/20 of the FFT sampling rate.

When the FFT mode is TRACe, The FFT sampling rate is the screen sampling rate (i.e. 100/horizontal time base).

When FFT mode is MEMory (memory), The FFT sampling rate is the memory sampling rate (ACQuire: SRATe?).

You can observe the detailed information of the spectrum by reducing the frequency range.

#### Return Format

Query returns the current frequency range in scientific counting form.

#### Example

```
:MATH:FFT:HSCale 125000      /* Set the frequency range of FFT operation results to
125kHz */
```

```
:MATH:FFT:HSCale?           /*Query returned 1.250000e+05*/
```

### 3.16.11 :MATH:FFT:HCENter

#### Syntax

:MATH:FFT:HCENter <cent>

:MATH:FFT:HCENter?

#### Description

Set or query the center frequency of the FFT operation result, which corresponds to the horizontal center of the screen. The default unit is Hz.

**Parameter**

| Name   | Type | Range           | Default |
|--------|------|-----------------|---------|
| <cent> | Real | Reference Marks | 125MHz  |

**Remarks**

- When the FFT mode is TRACe (trajectory), the range of <center> is 0 to (0.4 x FFT sampling rate); Among them, The FFT sampling rate is the screen sampling rate (i.e. 100/horizontal time base). When the FFT mode is MEMory (memory), the range of <center> is 0 to (0.5 x FFT sampling rate); Among them, The FFT sampling rate is the memory sampling rate (ACQuire: SRATE?).
- The set step is the horizontal gear/50 of the current FFT calculation result.

**Return Format**

Query returns the current center frequency in scientific counting form.

**Example**

```
:MATH:FFT:HCENTER 10000000      /* Set the center frequency of the FFT operation
result to 10MHz */
:MATH:FFT:HCENTER?              /* Query returns 1.000000 e+07*/
```

## 3.17 :SEGMENTation Commands

The SEGmentation command is used to set the parameters for segmented collection.

### 3.17.1 :SEGMENTation:WRECORD

**Syntax**

```
:SEGMENTation:WRECORD <frame>
:SEGMENTation:WRECORD?
```

**Description**

Set or query the frame rate for segmented collection.

**Parameter**

| Name    | Type    | Range   | Default |
|---------|---------|---------|---------|
| <frame> | Integer | 2-53601 | 10      |

**Remarks**

N/A

**Return Format**

Query returns the number of segmented collection frames in integer form.

#### Example

```
:SEGMENTation:WRECORD 100      /* Set the frame rate for segmented collection to
100*/
:SEGMENTation:WRECORD?          /* Query returns 100*/
```

### 3.17.2 :SEGMENTation:ENABLE

#### Syntax

```
:SEGMENTation:ENABLE <bool>
:SEGMENTation:ENABLE?
```

#### Description

Turn on or off the segmented collection function, or query the status of the segmented collection function.

#### Parameter

| Name   | Type | Range    | Default |
|--------|------|----------|---------|
| <bool> | Bool | {ON OFF} | OFF     |

#### Remarks

N/A

#### Return Format

Query returns ON or OFF.

#### Example

```
:SEGMENTation:ENABLE ON /* Turn on segmented collection function*/
:SEGMENTation:ENABLE?    /* Query returns ON */
```

## 3.18 :HISTORY Commands

The HISTORY command is used to set parameters related to waveform recording and playback.

### 3.18.1 :HISTORY:WRECORD

#### Syntax

:HISTORY:WRECORD <frame>

:HISTORY:WRECORD?

### Description

Set or query the frame rate for displaying historical waveforms.

### Parameter

| Name    | Type    | Range                                                    | Default |
|---------|---------|----------------------------------------------------------|---------|
| <frame> | Integer | 1 to the current maximum frame rate that can be recorded | 100     |

### Remarks

The current maximum frame rate that can be recorded can be achieved through:  
HISTORY: WRECORD: FMAX? Command query.

### Return Format

Query returns the current display frame rate in integer form.

### Example

:HISTORY:WRECORD 200 /\* Set the termination frame rate to 200\*/

:HISTORY:WRECORD? /\* Query returns 200\*/

## 3.18.2 :HISTORY:WRECORD:FINTERVAL

### Syntax

:HISTORY:WRECORD:FINTERVAL <interval>

:HISTORY:WRECORD:FINTERVAL?

### Description

Set or query the time interval between frames during waveform playback, with a default unit of s.

### Parameter

| Name       | Type | Range    | Default |
|------------|------|----------|---------|
| <interval> | Real | 20ms-10s | 100ms   |

### Remarks

N/A

### Return Format

Query returns the current set time interval in real form.

### Example

```
:HISTORY:WRECORD:FINTerval 0.02      /* Set the time interval to 20ms*/
:HISTORY:WRECORD:FINTerval?          /* Query returns 0.02*/
```

## 3.19 :DISPlay Commands

The DISPlay command can set the waveform display type, afterglow time, waveform brightness, screen display grid type, and grid brightness.

### 3.19.1 :DISPlay:TYPE

#### Syntax

```
:DISPlay:TYPE <type>
:DISPlay:TYPE?
```

#### Description

Set or query the display method of waveforms on the screen.

#### Parameter

| Name   | Type     | Range                    | Default |
|--------|----------|--------------------------|---------|
| <type> | Discrete | { {0 VECTors {1 DOTS}} } | 0       |

#### Remarks

- VECTors: Displayed by connecting lines between sampling points. This mode provides the most realistic waveform in most cases. It is convenient to view the steep edges of waveforms (such as square waves).
- DOTS: Directly display sampling points. You can visually see each sampling point and use the cursor to measure the X and Y values of that point.

#### Return Format

Query returns 0 or 1.

#### Example

```
:DISPlay:TYPE 1      /* Select point display method */
:DISPlay:TYPE?       /* Query returns 1*/
```

### 3.19.2 :DISPlay:GRADing:TIME

#### Syntax

```
:DISPlay:GRADing:TIME <time>
```

:DISPlay:GRADing:TIME?

### Description

Set or query the afterglow time, with a default unit of s.

### Parameter

| Name   | Type     | Range                               | Default |
|--------|----------|-------------------------------------|---------|
| <time> | Discrete | {MIN 0.1 0.2 0.5 1 2 5 10 INFinite} | MIN     |

### Remarks

- MIN: Setting the afterglow time to the minimum value allows for the observation of waveforms with high refresh rates.
- Designated value: Set the afterglow time to one of the specified values above, and observe burrs that change slowly or have a lower probability of occurrence.
- INFinite: When selecting N/A limit afterglow, the oscilloscope will not clear the previously collected waveform when displaying a new waveform. Measurable noise and jitter, capturing occasional events.

### Return Format

Query returns MIN, 0.1, 0.2, 0.5, 1, 2, 5, 10, or INFinite.

### Example

```
:DISPlay:GRADing:TIME 1      /* Set the afterglow time to 1 second */
:DISPlay:GRADing:TIME?      /* Query returns 1*/
```

## 3.19.3 :DISPlay:WBRightness

### Syntax

```
:DISPlay:WBRightness <time>
:DISPlay:WBRightness?
```

### Description

Set or query the brightness of waveform display on the screen.

### Parameter

| Name   | Type    | Range | Default |
|--------|---------|-------|---------|
| <time> | Integer | 0-100 | 100     |

### Remarks

N/A

### Return Format

Query returns an integer between 0 and 100.

#### Example

```
:DISPlay:WBRightness 50      /* Set waveform brightness to 50%*/
:DISPlay:WBRightness?        /* Query returns 50*/
```

### 3.19.4 :DISPlay:GRID

#### Syntax

```
:DISPlay:GRID <grid>
:DISPlay:GRID?
```

#### Description

Set or query the grid type displayed on the screen.

#### Parameter

| Name   | Type     | Range                           | Default |
|--------|----------|---------------------------------|---------|
| <grid> | Discrete | {{0 DOT}   {1 LINE} {2 CLOSE }} | 1       |

#### Remarks

- DOT: Grid Type is displayed as points.
- LINE: The grid type is displayed in lines.
- Close: Close the background grid.

#### Return Format

Query returns 0, 1, and 2.

#### Example

```
:DISPlay:GRID 2      /* Close background grid and coordinates */
:DISPlay:GRID?        /*Query returns 2*/
```

### 3.19.5 :DISPlay:GBRightness

#### Syntax

```
:DISPlay:GBRightness <brightness>
:DISPlay:GBRightness?
```

#### Description

Set or query the brightness of the screen grid.

**Parameter**

| Name         | Type    | Range | Default |
|--------------|---------|-------|---------|
| <brightness> | Integer | 0-100 | 0       |

**Remarks**

N/A

**Return Format**

Query returns an integer between 0 and 100.

**Example**

```
:DISPlay:GBrightness 60      /* Set the screen grid brightness to 60%*/
:DISPlay:GBrightness?        /* Query returns 60*/
```

## 3.20 :BUS Commands

### 3.20.1 :BUS:CAN:SOURce

**Syntax**

```
:BUS:CAN:SOURce <source>
:BUS:CAN:SOURce?
```

**Description**

Set up or query measurement sources.

**Parameter**

| Name     | Type     | Range                                                         | Default  |
|----------|----------|---------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 <br>CHANnel3 CHANnel4 <br>D1 D2 D3 D4 OFF} | CHANnel1 |

**Remarks**

N/A

**Return Format**

Query returns CHAN1,CHAN2,CHAN3,CHAN4,D1,D2,D3,D4 or OFF.

**Example**

```
:BUS:CAN:SOURce CHANnel2      /* Set the measurement source to CH2*/
:BUS:CAN:SOURce?              /* Query returns CHAN2*/
```

### 3.20.2 :BUS:CAN:THReshold

#### Syntax

:BUS:CAN:THReshold <source>

:BUS:CAN:THReshold?

#### Description

Set or query channel threshold thresholds.

#### Parameter

| Name     | Type  | Range                  | Default      |
|----------|-------|------------------------|--------------|
| <source> | float | -1.0V~1.0V,<br>unit: V | default<br>0 |

#### Remarks

N/A

#### Return Format

Query the threshold threshold for returning the current operation result in scientific counting form.

#### Example

:BUS:CAN:THReshold 1 /\* Set the channel threshold threshold to 1V\*/

:BUS:CAN:THReshold? /\* Query returns 1.000000 e+00\*/

### 3.20.3 :BUS:CAN:STYPe

#### Syntax

:BUS:CAN:STYPe <type>

:BUS:CAN:STYPe?

#### Description

Set or query the signal type.

#### Parameter

| Name   | Type     | Range                  | Default |
|--------|----------|------------------------|---------|
| <type> | Discrete | {CANH CANL RX TX DIFF} | CANH    |

#### Remarks

N/A

Return Format

Query returns CANH, CANL, RX, TX, or DIFF.

Example

```
:BUS:CAN:SType CANH      /* Set CAN signal type to CANH */
:BUS:CAN:SType?           /* Query returns CANH */
```

3.20.4 :BUS:CAN:BAUD

Syntax

```
:BUS:CAN:BAUD <type>
:BUS:CAN:BAUD?
```

Description

Set or query signal rate.

Parameter

| Name   | Type     | Range                                                                                                    | Default |
|--------|----------|----------------------------------------------------------------------------------------------------------|---------|
| <type> | Discrete | {10000 20000 33000 50000 62500 83300 100000 125000 250000 500000 800000 1000000 2000000 4000000 5000000} | 10000   |

Remarks

N/A

Return Format

Query returns 10kbps,20kbps,33.3kbps,50kbps,62.5kbps,83.3kbps,100kbps,125kbps,250kbps,500kbps,800kbps,1Mbps,2Mbps,4Mbps,5Mbps.

Example

```
:BUS:CAN:BAUD 10000      /* Set the signal rate to 10kbps */
:BUS:CAN:BAUD?           /* Query returns 10kbps */
```

3.20.5 :BUS:CAN:SPOint

Syntax

```
:BUS:CAN:SPOint <spoint>
```

:BUS:CAN:SPOint?

### Description

Set or query the sampling point location.

### Parameter

| Name     | Type    | Range | Default |
|----------|---------|-------|---------|
| <spoint> | Integer | 10-90 | 10      |

### Remarks

N/A

### Return Format

The query returns 10 to 90 (expressed as a percentage).

### Example

```
:BUS:CAN:SPOint 10      /* Set the sampling point position to 10*/
:BUS:CAN:SPOint?        /* Query returns 10*/
```

## 3.20.6 :BUS:IIC:SCLK:SOURce

### Syntax

```
:BUS:IIC:SCLK:SOURce <source>
:BUS:IIC:SCLK:SOURce?
```

### Description

Set or query the clock signal source.

### Parameter

| Name     | Type     | Range                                                    | Default  |
|----------|----------|----------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANneL2<br> CHANnel CHANnel4 D<br>0 D1 D2 D3} | CHANnel1 |

### Remarks

N/A

### Return Format

Query returns CHAN1,CHAN2,CHAN3,CHAN4,D0,D1,D2,D3.

### Example

```
:BUS:IIC:SCLK:SOURce CH2      /* Set the clock signal source to CH2*/
:BUS:IIC:SCLK:SOURce?         /* Query returns CH2*/
```

3.20.7

**:BUS:IIC:SDA:SOURce**

**Syntax**

```
:BUS:IIC:SDA:SOURce <source>
:BUS:IIC:SDA:SOURce?
```

**Description**

Set or query data sources.

**Parameter**

| Name     | Type     | Range                                             | Default  |
|----------|----------|---------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 D0 D1 D2 D3} | CHANnel1 |

**Remarks**

N/A

**Return Format**

Query returns CHAN1,CHAN2,CHAN3,CHAN4,D0,D1,D2,D3.

**Example**

```
:BUS:IIC:SDA:SOURce CHANnel2    /* Set the data source to CH2*/
:BUS:IIC:SDA:SOURce?           /* Query returns CH2*/
```

3.20.8

**:BUS:IIC:SCLK:THReshold**

**Syntax**

```
:BUS:IIC:SCLK:THReshold <source>
:BUS:IIC:SCLK:THReshold?
```

**Description**

Set or query the clock threshold threshold.

**Parameter**

| Name     | Type  | Range               | Default   |
|----------|-------|---------------------|-----------|
| <source> | float | -1.0V~1.0V, unit: V | default 0 |

**Remarks**

N/A

**Return Format**

Query the threshold threshold for returning the current operation result in scientific counting form.

**Example**

```
:BUS:IIC:SCLK:THReshold 1      /* Set the clock threshold to 1V */
:BUS:IIC:SCLK:THReshold?        /* Query returns 1.000000 e+00*/
```

### 3.20.9 :BUS:IIC:SDA:THReshold

**Syntax**

```
:BUS:IIC:SDA:THReshold <source>
:BUS:IIC:SDA:THReshold?
```

**Description**

Set or query data threshold thresholds.

**Parameter**

| Name     | Type  | Range                  | Default |
|----------|-------|------------------------|---------|
| <source> | float | -1.0V~1.0V,<br>unit: V | 0       |

**Remarks**

N/A

**Return Format**

Query the threshold threshold for returning the current operation result in scientific counting form.

**Example**

```
:BUS:IIC:SDA:THReshold 1      /* Set the data threshold threshold to 1V */
:BUS:IIC:SDA:THReshold?        /* Query returns 1.000000 e+00*/
```

### 3.20.10 :BUS:IIC:ADDRess

**Syntax**

:BUS:IIC:ADDReSS <source>

:BUS:IIC:ADDReSS?

### Description

Set or query address mode.

### Parameter

| Name     | Type     | Range     | Default |
|----------|----------|-----------|---------|
| <source> | Discrete | {RW NORW} | RW      |

### Remarks

NORW: Address bit width does not include R/W bits.

### Return Format

Query returns RW or NORW.

### Example

:BUS:IIC:ADDReSS NORW       /\* Set the address mode to NORW \*/

:BUS:IIC:ADDReSS?           /\* Query returns NORW \*/

## 3.20.11 :BUS:LIN:SOURce

### Syntax

:BUS:LIN:SOURce <source>

:BUS:LIN:SOURce?

### Description

Setting or querying LIN sources.

### Parameter

| Name     | Type     | Range                                                         | Default  |
|----------|----------|---------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 <br>CHANnel3 CHANnel4 <br>D1 D2 D3 D4 OFF} | CHANnel1 |

### Remarks

N/A

### Return Format

Query returns CHAN1,CHAN2,CHAN3,CHAN4,D1,D2,D3,D4 or OFF.

### Example

```
:BUS:LIN:SOURce CHANnel2      /* Set the measurement source to CH2*/
:BUS:LIN:SOURce?               /* Query returns CHAN2*/
```

### 3.20.12 :BUS:LIN:THReshold

#### Syntax

```
:BUS:LIN:THReshold <source>
:BUS:LIN:THReshold?
```

#### Description

Set or query channel threshold thresholds.

#### Parameter

| Name     | Type  | Range                  | Default      |
|----------|-------|------------------------|--------------|
| <source> | float | -1.0V~1.0V,<br>unit: V | default<br>0 |

#### Remarks

N/A

#### Return Format

Query the threshold threshold for returning the current operation result in scientific counting form.

#### Example

```
:BUS:LIN:THReshold 1      /* Set the channel threshold threshold to 1V */
:BUS:LIN:THReshold?        /* Query returns 1.000000 e+00*/
```

### 3.20.13 :BUS:LIN:BAUD

#### Syntax

```
:BUS:LIN:BAUD <type>
:BUS:LIN:BAUD?
```

#### Description

Set or query signal rate.

#### Parameter

| Name   | Type     | Range                           | Default |
|--------|----------|---------------------------------|---------|
| <type> | Discrete | {2400 4800 9600 19200 38400 576 | 115200  |

| Name | Type | Range                                                                     | Default |
|------|------|---------------------------------------------------------------------------|---------|
|      |      | 00 115200 230400 460800 921600 1000000 2000000 5000000 10000000 20000000} |         |

**Remarks**

N/A

**Return Format**

Query returns

2.4kbps,4.8kbps,9.6kbps,19.2kbps,38.4kbps,57.6kbps,115.2kbps,230.4kbps,460.8kbps,921.6kbps,1M,2M,5M,10M,20M.

**Example**

:BUS:LIN:BAUD 2400 /\* Set the signal rate to 2.4kbps\*/

:BUS:LIN:BAUD? /\* Query returns 2.4kbps\*/

### 3.20.14 :BUS:LIN:PARity

**Syntax**

:BUS:LIN:PARity &lt;type&gt;

:BUS:LIN:PARity?

**Description**

Set or query the checksum under LIN.

**Parameter**

| Name   | Type     | Range          | Default |
|--------|----------|----------------|---------|
| <type> | Discrete | {WITHOUT WITH} | WITHOUT |

**Remarks**

N/A

**Return Format**

Query returns either THE or THE.

**Example**

:BUS:LIN:PARity WITHOUT /\* Set the LIN checksum to WITOUT \*/

:BUS:LIN:PARity? /\* Query returns with OUT \*/

### 3.20.15 :BUS:LIN:STANdard

#### Syntax

:BUS:LIN:STANdard <type>

:BUS:LIN:STANdard?

#### Description

Set or query the LIN version.

#### Parameter

| Name   | Type     | Range           | Default |
|--------|----------|-----------------|---------|
| <type> | Discrete | { 1.X 2.X BOTH} | 1.X     |

#### Remarks

N/A

#### Return Format

Query returns 1. X, 2. X, or BOTH.

#### Example

:BUS:LIN:STANdard 1 /\* Set LIN version to 1. X \*/

:BUS:LIN:STANdard? /\* Query returns 1. X \*/

### 3.20.16 :BUS:RS232:BAUD

#### Syntax

:BUS:RS232:BAUD <baud>

:BUS:RS232:BAUD?

#### Description

Set or query the baud rate.

#### Parameter

| Name   | Type     | Range                                                                                                                       | Default |
|--------|----------|-----------------------------------------------------------------------------------------------------------------------------|---------|
| <baud> | Discrete | {600 1200 1800 2000 2400 4800 9600 19200 38400 57600 115200 230400 460800 921600 1000000 2000000 5000000 10000000 20000000} | 115200  |

#### Remarks

N/A

#### Return Format

Query returns

600bps,1.2kbps,1.8kbps,2.0kbps,2.4kbps,4.8kbps,9.6kbps,19.2kbps,38.4 kbps,57.6 kbps,115.2 kbps,230.4 kbps,460.8 kbps,921.6 kbps,1M,2M,5M,10M or 20M.

#### Example

:BUS:RS232:BAUD 2000 /\* Set the baud rate to 2000\*/

:BUS:RS232:BAUD? /\*Query returns 2.0kbps\*/

### 3.20.17 :BUS:RS232:TX

#### Syntax

:BUS:RS232:TX <source>

:BUS:RS232:TX?

#### Description

Set or query TX measurement sources.

#### Parameter

| Name     | Type     | Range                                                         | Default  |
|----------|----------|---------------------------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 <br>CHANnel3 CHANnel4 <br>D1 D2 D3 D4 OFF} | CHANnel1 |

#### Remarks

N/A

#### Return Format

Query returns CHAN1,CHAN2,CHAN3,CHAN4,D1,D2,D3,D4 or OFF.

#### Example

:BUS:RS232:TX CHANnel2 /\* Set the measurement source to CH2.\*/

:BUS:RS232:TX? /\* Query returns CHAN2\*/

### 3.20.18 :BUS:RS232:TX:THReshold

#### Syntax

:BUS:RS232:TX:THReshold <source>

:BUS:RS232:TX:THReshold?

**Description**

Set or query the channel TX threshold threshold.

**Parameter**

| Name     | Type  | Range                  | Default |
|----------|-------|------------------------|---------|
| <source> | float | -1.0V~1.0V,<br>unit: V | 0       |

**Remarks**

N/A

**Return Format**

Query the threshold threshold for returning the current operation result in scientific counting form.

**Example**

```
:BUS:RS232:TX:THReshold 1      /* Set the channel threshold threshold to 1V */
:BUS:RS232:TX:THReshold?        /* Query returns 1.000000 e+00*/
```

### 3.20.19 :BUS:RS232:RX

**Syntax**

```
:BUS:RS232:RX <source>
:BUS:RS232:RX?
```

**Description**

Set or query RX measurement sources.

**Parameter**

| Name     | Type     | Range                                                         | Default |
|----------|----------|---------------------------------------------------------------|---------|
| <source> | Discrete | {CHANnel1 CHANnel2 <br>CHANnel3 CHANnel4 <br>D1 D2 D3 D4 OFF} | OFF     |

**Remarks**

N/A

**Return Format**

Query returns CHAN1,CHAN2,CHAN3,CHAN4,D1,D2,D3,D4 or OFF.

**Example**

```
:BUS:RS232:RX CHANnel2      /* Set the measurement source to CH2*/
:BUS:RS232:RX?              /* Query returns CHAN2*/
```

### 3.20.20 :BUS:RS232:RX:THReshold

#### Syntax

```
:BUS:RS232:RX:THReshold <source>
:BUS:RS232:RX:THReshold?
```

#### Description

Set or query channel RX threshold threshold.

#### Parameter

| Name     | Type  | Range                  | Default   |
|----------|-------|------------------------|-----------|
| <source> | float | -1.0V~1.0V,<br>unit: V | default 0 |

#### Remarks

N/A

#### Return Format

Query the threshold threshold for returning the current operation result in scientific counting form.

#### Example

```
:BUS:RS232:RX:THReshold 1      /* Set the channel threshold threshold to 1V */
:BUS:RS232:RX:THReshold?       /*Query returns 1.000000e+00*/
```

### 3.20.21 :BUS:RS232:POLarity

#### Syntax

```
:BUS:RS232:POLarity <polarity>
:BUS:RS232:POLarity?
```

#### Description

Set or query RS232 polarity.

#### Parameter

| Name     | Type     | Range               | Default  |
|----------|----------|---------------------|----------|
| <source> | Discrete | {NEGative POSitive} | POSitive |

**Remarks**

N/A

**Return Format**

Query returns NEG or POS.

**Example**

```
:BUS:RS232:POLarity POSitive      /* Set RS232 polarity to positive polarity */  
:BUS:RS232:POLarity?             /*Query returns POS*/
```

### 3.20.22 **:BUS:RS232:ENDian**

**Syntax**

```
:BUS:RS232:ENDian <endian>  
:BUS:RS232:ENDian?
```

**Description**

Set or query data transmission bit order.

**Parameter**

| Name     | Type     | Range     | Default |
|----------|----------|-----------|---------|
| <endian> | Discrete | {LSB MSB} | MSB     |

**Remarks**

N/A

**Return Format**

Query returns LSB or MSB.

**Example**

```
:BUS:RS232:ENDian LSB      /* Set the data transmission bit sequence to LSB */  
:BUS:RS232:ENDian?         /*Query returns LSB*/
```

### 3.20.23 **:BUS:RS232:DBITs**

**Syntax**

```
:BUS:RS232:DBITs <value>  
:BUS:RS232:DBITs?
```

**Description**

Set or query data bit width.

**Parameter**

| Name    | Type    | Range       | Default |
|---------|---------|-------------|---------|
| <value> | Integer | {5 6 7 8 9} | 8       |

**Remarks**

N/A

**Return Format**

Query returns 5,6,7,8 or 9.

**Example**

```
:BUS:RS232:DBITs 5      /*设置数据位宽为 5 */
:BUS:RS232:DBITs?      /*Query returns 5*/
```

### 3.20.24 :BUS:RS232:PARity

**Syntax**

```
:BUS:RS232:PARity <value>
:BUS:RS232:PARity?
```

**Description**

设置 or 查询 RS232 校验方式.

**Parameter**

| Name      | Type     | Range            | Default |
|-----------|----------|------------------|---------|
| < value > | Discrete | {NONE ODD EVENT} | NONE    |

**Remarks**

N/A

**Return Format**

Query returns NONE,ODD or EVENT.

**Example**

```
:BUS:RS232:PARity ODD      /* Set RS232 verification method to odd verification */
:BUS:RS232:PARity?        /*Query returns ODD*/
```

### 3.20.25 **:BUS:RS232:STOP**

#### Syntax

:BUS:RS232:STOP <value>

:BUS:RS232:STOP?

#### Description

Set or query RS232 stop bit.

#### Parameter

| Name    | Type     | Range     | Default |
|---------|----------|-----------|---------|
| <value> | Discrete | {1 1.5 2} | 1       |

#### Remarks

N/A

#### Return Format

Query returns 1,1.5 or 2.

#### Example

:BUS:RS232:STOP 1           /\* Set the stop digit to 1\*/

:BUS:RS232:STOP?           /\*Query returns 1\*/

### 3.20.26 **:BUS:SPI:SCLK:SOURce**

#### Syntax

:BUS:SPI:SCLK:SOURce <source>

:BUS:SPI:SCLK:SOURce?

#### Description

Set or query SPI clock source.

#### Parameter

| Name     | Type     | Range                                                         | Default |
|----------|----------|---------------------------------------------------------------|---------|
| <source> | Discrete | {CHANnel1 CHANnel2 <br>CHANnel3 CHANnel4 <br>D1 D2 D3 D4 OFF} | OFF     |

#### Remarks

N/A

Return Format

Query returns CHAN1,CHAN2,CHAN3,CHAN4,D1,D2,D3 or OFF.

Example

:BUS:SPI:SCLK:SOURce CHANnel2 /\* Set SPI clock signal source to CH2\*/  
:BUS:SPI:SCLK:SOURce? /\*Query returns CHAN2\*/

3.20.27 :BUS:SPI:SCLK:SLOPe

Syntax

:BUS:SPI:SCLK:SLOPe <source>  
:BUS:SPI:SCLK:SLOPe?

Description

Set or query SPI clock edge type.

Parameter

| Name     | Type     | Range               | Default  |
|----------|----------|---------------------|----------|
| <source> | Discrete | {POSitive NEGative} | POSitive |

Remarks

N/A

Return Format

Query returns POS or NEG.

Example

:BUS:SPI:SCLK:SLOPe POSitive /\*设 Set SPI clock edge type to Positive \*/  
:BUS:SPI:SCLK:SLOPe? /\*Query returns POS\*/

3.20.28 :BUS:SPI:MISO:SOURce

Syntax

:BUS:SPI:MISO:SOURce <source>  
:BUS:SPI:MISO:SOURce?

Description

Set or query the MISO source type.

**Parameter**

| Name     | Type     | Range                                                          | Default |
|----------|----------|----------------------------------------------------------------|---------|
| <source> | Discrete | {CHANnel1 CHANnel2 <br>CHANnel3 CHANnel4 <br>D1 D2 D3 D4 OFF } | OFF     |

**Remarks**

N/A

**Return Format**

Query returns CHAN1,CHAN2,CHAN3,CHAN4,D1,D2,D3,D4 or OFF.

**Example**

```
:BUS:SPI:MISO:SOURce CHANnel2      /* Set MISO source to CH2*/
:BUS:SPI:MISO:SOURce?                /*Query returns CHAN2*/
```

**3.20.29 :BUS:SPI:MOSI:SOURce****Syntax**

```
:BUS:SPI:MOSI:SOURce <source>
:BUS:SPI:MOSI:SOURce?
```

**Description**

Set or query MOSI source type.

**Parameter**

| Name     | Type     | Range                                                          | Default |
|----------|----------|----------------------------------------------------------------|---------|
| <source> | Discrete | {CHANnel1 CHANnel2 <br>CHANnel3 CHANnel4 <br>D1 D2 D3 D4 OFF } | OFF     |

**Remarks**

N/A

**Return Format**

Query returns CHAN1,CHAN2,CHAN3,CHAN4,D1,D2,D3,D4 or OFF.

**Example**

```
:BUS:SPI:MOSI:SOURce CHANnel2      /* Set the MOSI signal source to CH2*/
:BUS:SPI:MOSI:SOURce?                /*Query returns CHAN2*/
```

### 3.20.30 :BUS:SPI:SCLK:SOURce:THReshold

#### Syntax

:BUS:SPI:SCLK:SOURce:THReshold <source>

:BUS:SPI:SCLK:SOURce:THReshold?

#### Description

Set or query clock threshold threshold.

#### Parameter

| Name     | Type  | Range                  | Default   |
|----------|-------|------------------------|-----------|
| <source> | float | -1.0V~1.0V,<br>unit: V | default 0 |

#### Remarks

N/A

#### Return Format

Query the threshold threshold for returning the current operation result in scientific counting form.

#### Example

:BUS:SPI:SCLK:SOURce:THReshold 1 /\* Set the clock threshold to 1V \*/

:BUS:SPI:SCLK:SOURce:THReshold? /\*Query returns 1.000000e+00\*/

### 3.20.31 :BUS:SPI:MISO:SOURce:THReshold

#### Syntax

:BUS:SPI:MISO:SOURce:THReshold <source>

:BUS:SPI:MISO:SOURce:THReshold?

#### Description

Set or query MISO threshold threshold.

#### Parameter

| Name    | Type  | Range                  | Default   |
|---------|-------|------------------------|-----------|
| <value> | float | -1.0V~1.0V,<br>unit: V | default 0 |

#### Remarks

N/A

**Return Format**

Query the threshold threshold for returning the current operation result in scientific counting form.

**Example**

```
:BUS:SPI:MISO:SOURce:THReshold 1      /* Set the MISO threshold to 1V */
:BUS:SPI:MISO:SOURce:THReshold?        /*Query returns 1.000000e+00*/
```

### 3.20.32 **:BUS:SPI:MOSI:SOURce:THReshold**

**Syntax**

```
:BUS:SPI:MOSI:SOURce:THReshold <source>
:BUS:SPI:MOSI:SOURce:THReshold?
```

**Description**

Set or query MOSI threshold threshold.

**Parameter**

| Name     | Type  | Range               | Default   |
|----------|-------|---------------------|-----------|
| <source> | float | -1.0V~1.0V, unit: V | default 0 |

**Remarks**

N/A

**Return Format**

Query the threshold threshold for returning the current operation result in scientific counting form.

**Example**

```
:BUS:SPI:MOSI:SOURce:THReshold 1      /* Set the MOSI threshold to 1V */
:BUS:SPI:MOSI:SOURce:THReshold?        /*Query returns 1.000000e+00*/
```

### 3.20.33 **:BUS:SPI:MODE**

**Syntax**

```
:BUS:SPI:MODE <mode>
:BUS:SPI:MODE?
```

**Description**

Set or query SPI decoding mode.

**Parameter**

| Name   | Type     | Range        | Default |
|--------|----------|--------------|---------|
| <mode> | Discrete | {TIMEOUT CS} | TIMEOUT |

**Remarks**

N/A

**Return Format**

Query returns TIMEOUT or CS.

**Example**

```
:BUS:SPI:MODE TIMEOUT      /* Set SPI decoding mode to timeout */
:BUS:SPI:MODE?              /*Query returns TIMEOUT */
```

### 3.20.34 :BUS:SPI:CS:SOURce

**Syntax**

```
:BUS:SPI:CS:SOURce <source>
:BUS:SPI:CS:SOURce?
```

**Description**

Set or query CS film selection source.

**Parameter**

| Name     | Type     | Range                                                         | Default |
|----------|----------|---------------------------------------------------------------|---------|
| <source> | Discrete | {CHANnel1 CHANnel2 <br>CHANnel3 CHANnel4 <br>D1 D2 D3 D4 OFF} | OFF     |

**Remarks**

N/A

**Return Format**

Query returns CHAN1,CHAN2,CHAN3,CHAN4,D1,D2,D3,D4 or OFF.

**Example**

```
:BUS:SPI:CS:SOURce CHANnel2      /* Set CS film selection source to CH2 */
:BUS:SPI:CS:SOURce?              /*Query returns CHAN2*/
```

### 3.20.35 :BUS:SPI:CS:POLarity

#### Syntax

:BUS:SPI:CS:POLarity <polarity>

:BUS:SPI:CS:POLarity?

#### Description

Set or query the polarity of the film selection line.

#### Parameter

| Name       | Type     | Range               | Default |
|------------|----------|---------------------|---------|
| <polarity> | Discrete | {POSitive NEGative} | POS     |

#### Remarks

N/A

#### Return Format

Query returns POS or NEG.

#### Example

:BUS:SPI:CS:POLarity NEGative /\* Set the polarity of the chip selection line to negative polarity \*/

:BUS:SPI:CS:POLarity? /\*Query returns NEG\*/

### 3.20.36 :BUS:SPI:CS:THReshold

#### Syntax

:BUS:SPI:CS:THReshold <value>

:BUS:SPI:CS:THReshold?

#### Description

Set or query the threshold threshold for line selection.

#### Parameter

| Name    | Type  | Range                  | Default |
|---------|-------|------------------------|---------|
| <value> | float | -1.0V~1.0V,<br>unit: V | 0       |

#### Remarks

N/A

Return Format

Query the threshold threshold for returning the current operation result in scientific counting form.

Example

```
:BUS:SPI:CS:THReshold 1      /* Set the threshold threshold for line selection to 1V */
:BUS:SPI:CS:THReshold?      /*Query returns 1.000000e+00*/
```

3.20.37 :BUS:SPI:ENDian

Syntax

```
:BUS:SPI:ENDian <endian>
:BUS:SPI:ENDian?
```

Description

Set or query the bit order of SPI.

Parameter

| Name     | Type     | Range     | Default |
|----------|----------|-----------|---------|
| <endian> | Discrete | {LSB MSB} | MSB     |

Remarks

N/A

Return Format

Query returns LSB or MSB.

Example

```
:BUS:SPI:ENDian LSB      /* Set the bit order of the chip selection line to LSB */
:BUS:SPI:ENDian?      /*Query returns LSB*/
```

3.20.38 :BUS:SPI:DBITs

Syntax

```
:BUS:SPI:DBITs <value>
:BUS:SPI:DBITs?
```

Description

Set or query SPI data bit width.

#### Parameter

| Name    | Type     | Range | Default |
|---------|----------|-------|---------|
| <value> | Discrete | 4-32  | 8       |

#### Remarks

N/A

#### Return Format

Query returns 4,8,16,32,Custom.

#### Example

```
:BUS:SPI:DBITs 4      /* Set the data bit width to 4 */
:BUS:SPI:DBITs?       /*Query returns 4*/
```

### 3.20.39 :BUS:SPI:POLarity

#### Syntax

```
:BUS:SPI:POLarity <polarity>
:BUS:SPI:POLarity?
```

#### Description

Set or query SPI polarity.

#### Parameter

| Name       | Type     | Range               | Default  |
|------------|----------|---------------------|----------|
| <polarity> | Discrete | {POSitive NEGative} | POSitive |

#### Remarks

N/A

#### Return Format

Query returns POS or NEG.

#### Example

```
:BUS:SPI:POLarity POSitive /* Set SPI polarity to positive polarity */
:BUS:SPI:POLarity?        /*Query returns POS*/
```

## 3.21 :SEARch Commands

### 3.21.1 :SEARch:STATus

#### Syntax

:SEARch:STATus <status>

:SEARch:STATus?

#### Description

Set or query search switch.

#### Parameter

| Name     | Type     | Range         | Default |
|----------|----------|---------------|---------|
| <status> | Discrete | { 0 OFF 1 ON} | OFF     |

#### Remarks

N/A

#### Return Format

Query returns OFF or ON.

#### Example

:SEARch:STATus OFF      /\* Set the search switch to OFF \*/

:SEARch:STATus?      /\*Query returns OFF\*/

### 3.21.2 :SEARch:TYPE

#### Syntax

:SEARch:TYPE <type>

:SEARch:TYPE?

#### Description

Set or query search type.

#### Parameter

| Name     | Type     | Range                                 | Default |
|----------|----------|---------------------------------------|---------|
| < type > | Discrete | {EDGE SLOPE RUNT PULSE RS232 IIC SPI} | EDGE    |

#### Remarks

N/A

#### Return Format

Query returns EDGE,SLOPE,RUNT,PULSE,RS232,IIC or SPI.

#### Example

```
:SEARCh:TYPE EDGE      /* Set the search type to EDGE. */
:SEARCh:TYPE?          /*Query returns EDGE */
```

### 3.21.3 :SEARCh:MARK:STATUs

#### Syntax

```
:SEARCh:MARK:STATUs <status>
:SEARCh:MARK:STATUs?
```

#### Description

Set or query the search tag table switch.

#### Parameter

| Name     | Type     | Range        | Default |
|----------|----------|--------------|---------|
| <status> | Discrete | {0 OFF 1 ON} | OFF     |

#### Remarks

N/A

#### Return Format

Query returns OFF or ON.

#### Example

```
:SEARCh:MARK:STATUs OFF      /* Set the search tag table switch to OFF */
:SEARCh:MARK:STATUs?        /*Query returns OFF */
```

### 3.21.4 :SEARCh:EDGE:SOURCe

#### Syntax

```
:SEARCh:EDGE:SOURCe <source>
:SEARCh:EDGE:SOURCe?
```

#### Description

Set or query edge search signal source.

**Parameter**

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

**Remarks**

N/A

**Return Format**

Query returns CHAN1,CHAN2,CHAN3,CHAN4,EXT.

**Example**

```
:SEARCh:EDGE:SOURce CH2      /* Set the edge search signal source to CH2 */
:SEARCh:EDGE:SOURce?          /*Query returns CHAN2*/
```

### 3.21.5 :SEARCh:EDGE:SLOPe

**Syntax**

```
:SEARCh:EDGE:SLOPe <val>
:SEARCh:EDGE:SLOPe?
```

**Description**

Set or query edge search edge judgment.

**Parameter**

| Name  | Type     | Range                   | Default |
|-------|----------|-------------------------|---------|
| <val> | Discrete | {RISIng FALLIng EITHer} | RISIng  |

**Remarks**

RISING: Rising edge

FALLING: Falling edge

EITHer: Rising or falling edge

**Return Format**

Query returns RISIng,FALLIng or EITHer.

**Example**

```
:SEARCh:EDGE:SLOPe RISIng    /* Set edge search edge judgment to RISING */
:SEARCh:EDGE:SLOPe?          /*Query returns RISIng*/
```

### 3.21.6 :SEARch:EDGE:THReshold

#### Syntax

:SEARch:EDGE:THReshold <value>

:SEARch:EDGE:THReshold?

#### Description

Set or query edge search threshold threshold.

#### Parameter

| Name    | Type  | Range                                                  | Default |
|---------|-------|--------------------------------------------------------|---------|
| <value> | float | (-4 * current voltage level~4 * current voltage level) | 0       |

#### Remarks

N/A

#### Return Format

Query the threshold threshold for returning the current operation result in scientific counting form. Query the threshold threshold for returning the current operation result in scientific counting form.

#### Example

:SEARch:EDGE:THReshold 2 /\* Set the edge search threshold to 2V \*/

:SEARch:EDGE:THReshold? /\*Query returns 2.000000e+00\*/

### 3.21.7 :SEARch:SLOPe:SOURce

#### Syntax

:SEARch:SLOPe:SOURce <source>

:SEARch:SLOPe:SOURce?

#### Description

Set or query slope to search for signal sources.

#### Parameter

| Name     | Type     | Range                            | Default  |
|----------|----------|----------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHAN | CHANnel1 |

| Name | Type | Range     | Default |
|------|------|-----------|---------|
|      |      | nel4 EXT} |         |

**Remarks**

N/A

**Return Format**

Query returns CHAN1,CHAN2,CHAN3,CHAN4 or EXT.

**Example**

```
:SEARCh:SLOPe:SOURce CHANnel2    /* Set the slope search signal source to
CH2*/
```

```
:SEARCh:SLOPe:SOURce?              /*Query returns CHAN2*/
```

### 3.21.8 :SEARCh:SLOPe:SLOPe

**Syntax**

```
:SEARCh:SLOPe:SLOPe <val>
```

```
:SEARCh:SLOPe:SLOPe?
```

**Description**

Set or query slope search edge judgment.

**Parameter**

| Name  | Type     | Range             | Default |
|-------|----------|-------------------|---------|
| <val> | Discrete | {RISIng FALLIng } | RISIng  |

**Remarks**

N/A

**Return Format**

Query returns RISIng,FALLIng.

**Example**

```
:SEARCh:SLOPe:SLOPe RISIng        /* Set the slope search edge to determine
RISING */
```

```
:SEARCh:SLOPe:SLOPe?              /*Query returns RISIng*/
```

### 3.21.9 :SEARCh:SLOPe:WHEN

**Syntax**

```
:SEARCh:SLOPe:WHEN <val>
```

```
:SEARCh:SLOPe:WHEN?
```

### Description

Set or query slope search time conditions.

### Parameter

| Name  | Type     | Range                                 | Default |
|-------|----------|---------------------------------------|---------|
| <val> | Discrete | {0 EQUAL 1 NOTEQUAL 2 GREATER 3 LESS} | EQUAL   |

### Remarks

N/A

### Return Format

Query returns EQUAL,NOTEQUAL,GREATER or LESS.

### Example

```
:SEARCh:SLOPe:WHEN EQUAL      /* Set the slope search time condition to QUAL */
```

```
:SEARCh:SLOPe:WHEN?          /*Query returns EQUAL*/
```

## 3.21.10 :SEARCh:SLOPe:TIME

### Syntax

```
:SEARCh:SLOPe:TIME <val>
```

```
:SEARCh:SLOPe:TIME?
```

### Description

Set or query slope search time threshold value.

### Parameter

| Name  | Type     | Range   | Default |
|-------|----------|---------|---------|
| <val> | Discrete | 8ns-12s | 8ns     |

### Remarks

N/A

### Return Format

Query returns the threshold value in scientific counting form.

### Example

```
:SEARCH:SLOPe:TIME 0.002          /* Set the threshold value to 2ms */
:SEARCH:SLOPe:TIME?                /*Query returns 2.000000e-03*/
```

### 3.21.11 :SEARCH:SLOPe:HIGh:THReshold

#### Syntax

```
:SEARCH:SLOPe:HIGh:THReshold <value>
:SEARCH:SLOPe:HIGh:THReshold?
```

#### Description

Set the value of the or query slope search threshold A.

#### Parameter

| Name    | Type  | Range                                                  | Default |
|---------|-------|--------------------------------------------------------|---------|
| <value> | float | (-4 * Current voltage level~4 * Current voltage level) | 0       |

#### Remarks

N/A

#### Return Format

Query the threshold for returning the current operation result in scientific counting form.

#### Example

```
:SEARCH:SLOPe:HIGh:THReshold 2          /* Set the slope search threshold A
value to 2V */
:SEARCH:SLOPe:HIGh:THReshold?            /*Query returns 2.000000e+00*/
```

### 3.21.12 :SEARCH:SLOPe:LOW:THReshold

#### Syntax

```
:SEARCH:SLOPe:LOW:THReshold <value>
:SEARCH:SLOPe:LOW:THReshold?
```

#### Description

Set or query slope search threshold B value.

#### Parameter

| Name    | Type  | Range                                                  | Default |
|---------|-------|--------------------------------------------------------|---------|
| <value> | float | (-4 * Current voltage level~4 * Current voltage level) | 0       |

**Remarks**

N/A

**Return Format**

Query the threshold for returning the current operation result in scientific counting form.

**Example**

```
:SEARCh:SLOPe:LOW:THReshold 2      /* Set the slope search threshold B to
2V */
```

```
:SEARCh:SLOPe:LOW:THReshold?      /*Query returns 2.000000e+00*/
```

### 3.21.13 **:SEARCh:RUNT:SOURce**

**Syntax**

```
:SEARCh:RUNT:SOURce <source>
```

```
:SEARCh:RUNT:SOURce?
```

**Description**

Set or query underamplitude search signal source.

**Parameter**

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

**Remarks**

N/A

**Return Format**

Query returns CHAN1,CHAN2,CHAN3,CHAN4 or EXT.

**Example**

```
:SEARCh:RUNT:SOURce CHANnel2      /* Set the underamplitude search signal
source to CH2*/
```

```
:SEARCh:RUNT:SOURce?      /*Query returns CHAN2*/
```

### 3.21.14 **:SEARCh:RUNT:POLarity**

**Syntax**

:SEARCh:RUNT:POLarity <value>

:SEARCh:RUNT:POLarity?

### Description

Set or query under amplitude search polarity.

### Parameter

| Name    | Type     | Range               | Default  |
|---------|----------|---------------------|----------|
| <value> | Discrete | {POSitive NEGative} | POSitive |

### Remarks

N/A

### Return Format

Query returns POSitive or NEGative.

### Example

:SEARCh:RUNT:POLarity POSitive      /\* Set the underamplitude search polarity to positive polarity \*/

:SEARCh:RUNT:POLarity?      /\*Query returns POSitive \*/

## 3.21.15 :SEARCh:RUNT:WHEN

### Syntax

:SEARCh:RUNT:WHEN <value>

:SEARCh:RUNT:WHEN?

### Description

Set or query under frame search time conditions.

### Parameter

| Name      | Type     | Range                         | Default |
|-----------|----------|-------------------------------|---------|
| < value > | Discrete | {EQUAL NOTEQUAL GREATER LESS} | EQUAL   |

### Remarks

N/A

### Return Format

Query returns EQUAL,NOTEQUAL,GREATER or LESS.

### Example

:SEARCh:RUNT:WHEN EQUAL      /\* Set the under amplitude search time condition

to QUAL \*/

:SEARCh:RUNT:WHEN?

/\*Query returns EQUAL\*/

### 3.21.16 :SEARCh:RUNT:TIME

#### Syntax

:SEARCh:RUNT:TIME <value>

:SEARCh:RUNT:TIME?

#### Description

Set or query under amplitude search threshold value.

#### Parameter

| Name      | Type     | Range   | Default |
|-----------|----------|---------|---------|
| < value > | Discrete | 8ns~12s | 8ns     |

#### Remarks

N/A

#### Return Format

Query returns threshold value in scientific counting form.

#### Example

:SEARCh:RUNT:TIME 0.002

/\* Set the under amplitude search threshold to

2ms. \*/

:SEARCh:RUNT:TIME?

/\*Query returns 2.000000e-03\*/

### 3.21.17 :SEARCh:RUNT:HIGh:THReshold

#### Syntax

:SEARCh:RUNT:HIGh:THReshold <value>

:SEARCh:RUNT:HIGh:THReshold?

#### Description

Set or query the value of the under amplitude search threshold A.

#### Parameter

| Name    | Type     | Range                                                  | Default |
|---------|----------|--------------------------------------------------------|---------|
| <value> | Discrete | (-4 * Current voltage level~4 * Current voltage level) | 0       |

**Remarks**

N/A

**Return Format**

Query the threshold for returning the current operation result in scientific counting form.

**Example**

```
:SEARCh:RUNT:HIGH:THReshold 2      /* Set the value of the under amplitude
search threshold A to 2V*/
```

```
:SEARCh:RUNT:HIGH:THReshold?      /*Query returns 2.000000e+00*/
```

### 3.21.18 :SEARCh:RUNT:LOW:THReshold

**Syntax**

```
:SEARCh:RUNT:LOW:THReshold <value>
```

```
:SEARCh:RUNT:LOW:THReshold?
```

**Description**

Set or query the value of the under amplitude search threshold B.

**Parameter**

| Name    | Type     | Range                                                  | Default |
|---------|----------|--------------------------------------------------------|---------|
| <value> | Discrete | (-4 * Current voltage level~4 * Current voltage level) | 0       |

**Remarks**

N/A

**Return Format**

Query the threshold for returning the current operation result in scientific counting form.

**Example**

```
:SEARCh:RUNT:LOW:THReshold 2      /* Set the value of the under amplitude
search threshold B to 2V */
```

```
:SEARCh:RUNT:LOW:THReshold?      /*Query returns 2.000000e+00*/
```

### 3.21.19 :SEARCh:PULSe:SOURce

**Syntax**

```
:SEARCh:PULSe:SOURce <source>
```

:SEARCh:PULSe:SOURce?

### Description

Set or query pulse width search signal source.

### Parameter

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

### Remarks

N/A

### Return Format

Query returns CHAN1,CHAN2,CHAN3,CHAN4 or EXT.

### Example

:SEARCh:PULSe:SOURce CHANnel2      /\* Set the pulse width search signal source to CH2\*/

:SEARCh:PULSe:SOURce?      /\*Query returns CHAN2\*/

## 3.21.20 :SEARCh:PULSe:SLOPe

### Syntax

:SEARCh:PULSe:SLOPe <value>

:SEARCh:PULSe:SLOPe?

### Description

Set or query pulse width search edge.

### Parameter

| Name    | Type     | Range             | Default |
|---------|----------|-------------------|---------|
| <value> | Discrete | {RISing FALLing } | RISing  |

### Remarks

N/A

### Return Format

Query returns RISing or FALLing.

### Example

```
:SEARCh:PULSe:SLOPe RISIng      /* Set the pulse width search edge to the rising
edge */
:SEARCh:PULSe:SLOPe?              /*Query returns RISIng*/
```

### 3.21.21 :SEARCh:PULSe:WHEN

#### Syntax

```
:SEARCh:PULSe:WHEN <value>
:SEARCh:PULSe:WHEN?
```

#### Description

Set or query pulse width search trigger conditions.

#### Parameter

| Name      | Type     | Range                         | Default |
|-----------|----------|-------------------------------|---------|
| < value > | Discrete | {EQUAL NOTEQUAL GREATER LESS} | EQUAL   |

#### Remarks

N/A

#### Return Format

Query returns EQUAL,NOTEQUAL,GREATER,LESS.

#### Example

```
:SEARCh:PULSe:WHEN EQUAL      /* Set the pulse width search trigger condition to
QUAL */
:SEARCh:PULSe:WHEN?           /*Query returns EQUAL*/
```

### 3.21.22 :SEARCh:PULSe:TIME

#### Syntax

```
:SEARCh:PULSe:TIME <value>
:SEARCh:PULSe:TIME?
```

#### Description

Set or query pulse width search time threshold value.

#### Parameter

| Name    | Type     | Range   | Default |
|---------|----------|---------|---------|
| <value> | Discrete | 8ns-12s | 8ns     |

**Remarks**

N/A

**Return Format**

Query returns threshold value in scientific counting form.

**Example**

```
:SEARCh:PULSe:TIME 0.002          /* Set the pulse width search time threshold
to 2ms */
```

```
:SEARCh:PULSe:TIME?                /*Query returns 2.000000e-03*/
```

### 3.21.23 :SEARCh:PULSe:THReshold

**Syntax**

```
:SEARCh:PULSe:THReshold <value>
```

```
:SEARCh:PULSe:THReshold?
```

**Description**

Set or query pulse width search threshold.

**Parameter**

| Name    | Type  | Range                                                  | Default |
|---------|-------|--------------------------------------------------------|---------|
| <value> | float | (-4 * current voltage level~4 * current voltage level) | 0       |

**Remarks**

N/A

**Return Format**

Query the threshold threshold for returning the current operation result in scientific counting form.

**Example**

```
:SEARCh:PULSe:THReshold 2          /* Set the pulse width search threshold to 2V */
```

```
:SEARCh:PULSe:THReshold?           /*Query returns 2.000000e+00*/
```

### 3.21.24 :SEARCh:RS232:SOURce

#### Syntax

```
:SEARCh:RS232:SOURce <source>
```

```
:SEARCh:RS232:SOURce?
```

#### Description

Set or query RS232 signal source.

#### Parameter

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

#### Remarks

N/A

#### Return Format

Query returns CHAN1,CHAN2,CHAN3,CHAN4 or EXT.

#### Example

```
:SEARCh:RS232:SOURce CH2      /* Set RS232 signal source to CH2*/
```

```
:SEARCh:RS232:SOURce?        /*Query returns CHAN2*/
```

### 3.21.25 :SEARCh:RS232:POLarity

#### Syntax

```
:SEARCh:RS232:POLarity <value>
```

```
:SEARCh:RS232:POLarity?
```

#### Description

Set or query RS232 search polarity.

#### Parameter

| Name    | Type     | Range               | Default  |
|---------|----------|---------------------|----------|
| <value> | Discrete | {POSitive NEGative} | POSitive |

#### Remarks

N/A

#### Return Format

Query returns POSitive or NEGative.

#### Example

```
:SEARCh:RS232:POLarity NEGative      /* Set RS232 search polarity to negative
polarity */
:SEARCh:RS232:POLarity?               /*Query returns NEGative*/
```

### 3.21.26 :SEARCh:RS232:BAUD

#### Syntax

```
:SEARCh:RS232:BAUD <type>
:SEARCh:RS232:BAUD?
```

#### Description

Set or query RS232 baud rate.

#### Parameter

| Name   | Type     | Range                                                                                                                       | Default |
|--------|----------|-----------------------------------------------------------------------------------------------------------------------------|---------|
| <type> | Discrete | {600 1200 1800 2000 2400 4800 9600 19200 38400 57600 115200 230400 460800 921600 1000000 2000000 5000000 10000000 20000000} | 115200  |

#### Remarks

N/A

#### Return Format

Query returns  
600,1200,1800,2000,2400,4800,9600,19200,38400,57600,115200,230400,460800,921600,1000000,2000000,5000000,10000000 or 20000000.

#### Example

```
:SEARCh:RS232:BAUD 1200              /* Set RS232 baud rate to 1.2kbsSet RS232
baud rate to 1.2kbs */
:SEARCh:RS232:BAUD?                  /*Query returns 1200 */
```

### 3.21.27 :SEARCh:RS232:WHEN

#### Syntax

```
:SEARCh:RS232:WHEN <value>
```

:SEARCh:RS232:WHEN?

### Description

Set or query RS232 trigger conditions.

### Parameter

| Name    | Type     | Range                                  | Default |
|---------|----------|----------------------------------------|---------|
| <value> | Discrete | {START STOP DATA CHECK<br>ERROR ERROR} | START   |

### Remarks

N/A

### Return Format

Query returns START,STOP,DATA,CHECKERROR,ERROR.

### Example

```
:SEARCh:RS232:WHEN START      /* Set RS232 search time condition to frame start
*/
:SEARCh:RS232:WHEN?           /*Query returns START*/
```

## 3.21.28 :SEARCh:RS232:DBITs

### Syntax

```
:SEARCh:RS232:DBITs <value>
:SEARCh:RS232:DBITs?
```

### Description

Set or query RS232 bit width.

### Parameter

| Name    | Type    | Range       | Default |
|---------|---------|-------------|---------|
| <value> | Integer | {4 5 6 7 8} | 8       |

### Remarks

N/A

### Return Format

Query returns 4,5,6,7 or 8.

### Example

```
:SEARCh:RS232:DBITs 5          /*Set the data bit width to 5*/
```

:SEARCh:RS232:DBITs? /\*Query returns 5\*/

### 3.21.29 :SEARCh:RS232:PARity

#### Syntax

:SEARCh:RS232:PARity <value>

:SEARCh:RS232:PARity?

#### Description

Set or query RS232 checksum type.

#### Parameter

| Name    | Type     | Range           | Default |
|---------|----------|-----------------|---------|
| <value> | Discrete | {NONE ODD EVEN} | NONE    |

#### Remarks

N/A

#### Return Format

Query returns NONE, ODD or EVEN.

#### Example

:SEARCh:RS232:PARity NONE /\* Set the checksum type to NONE \*/

:SEARCh:RS232:PARity? /\*Query returns NONE \*/

### 3.21.30 :SEARCh:RS232:STOP

#### Syntax

:SEARCh:RS232:STOP <value>

:SEARCh:RS232:STOP?

#### Description

Set or query RS232 frame end bit.

#### Parameter

| Name    | Type     | Range | Default |
|---------|----------|-------|---------|
| <value> | Discrete | {1 2} | 1       |

#### Remarks

N/A

**Return Format**

Query returns 1 or 2.

**Example**

```
:SEARCh:RS232:STOP 1      /* Set RS232 frame end bit to 1 bit */
:SEARCh:RS232:STOP?      /*Query returns 1*/
```

**3.21.31 :SEARCh:RS232:DATA****Syntax**

```
:SEARCh:RS232:DATA <value>
:SEARCh:RS232:DATA?
```

**Description**

Set or query RS232 data values.

**Parameter**

| Name    | Type     | Range       | Default |
|---------|----------|-------------|---------|
| <value> | Discrete | 0x0 -- 0xff |         |

**Remarks**

N/A

**Return Format**

Query returns an integer.

**Example**

```
:SEARCh:RS232:DATA 4      /* Set RS232 data value to 4*/
:SEARCh:RS232:DATA?      /*Query returns 4*/
```

**3.21.32 :SEARCh:RS232:THReshold****Syntax**

```
:SEARCh:RS232:THReshold <value>
:SEARCh:RS232:THReshold?
```

**Description**

Set or query RS232 threshold threshold.

**Parameter**

| Name    | Type  | Range                                                  | Default |
|---------|-------|--------------------------------------------------------|---------|
| <value> | float | (-4 * current voltage level-4 * current voltage level) | 0       |

**Remarks**

N/A

**Return Format**

Query the threshold threshold for returning the current operation result in scientific counting form.

**Example**

```
:SEARCh:RS232:THReshold 3      /* Set the RS232 threshold to 3V */
:SEARCh:RS232:THReshold?        /*Query returns 3.000000e+00*/
```

### 3.21.33 :SEARCh:IIC:SCLK:SOURce

**Syntax**

```
:SEARCh:IIC:SCLK:SOURce <source>
:SEARCh:IIC:SCLK:SOURce?
```

**Description**

Set or query IIC clock source.

**Parameter**

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

**Remarks**

N/A

**Return Format**

Query returns CHAN1,CHAN2,CHAN3,CHAN4 or EXT.

**Example**

```
:SEARCh:IIC:SCLK:SOURce CHANnel2 /* Set the IIC clock signal source to CH2*/
:SEARCh:IIC:SCLK:SOURce?          /*Query returns CHAN2*/
```

### 3.21.34 :SEARCh:IIC:SDA:SOURce

#### Syntax

```
:SEARCh:IIC:SDA:SOURce <source>
```

```
:SEARCh:IIC:SDA:SOURce?
```

#### Description

Set or query IIC data source.

#### Parameter

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

#### Remarks

N/A

#### Return Format

Query returns CHAN1,CHAN2,CHAN3,CHAN4 or EXT.

#### Example

```
:SEARCh:IIC:SDA:SOURce CHANnel2          /*Set the IIC data source to CH2*/
```

```
:SEARCh:IIC:SDA:SOURce?                  /*Query returns CHAN2*/
```

### 3.21.35 :SEARCh:IIC:WHEN

#### Syntax

```
:SEARCh:IIC:WHEN <value>
```

```
:SEARCh:IIC:WHEN?
```

#### Description

Set or query IIC search time conditions.

#### Parameter

| Name    | Type     | Range                                          | Default |
|---------|----------|------------------------------------------------|---------|
| <value> | Discrete | {{START STOP RESTART MISSEDACK ADDRESS AANDD}} | START   |

#### Remarks

N/A

**Return Format**

Query returns START,STOP,RESTART,MISSEDACK,ADDRESS or AANDDD.

**Example**

```
:SEARCh:IIC:WHEN START      /* Set the triggering condition of IIC to frame start
*/
:SEARCh:IIC:WHEN?           /*Query returns START*/
```

**3.21.36 :SEARCh:IIC:ADDReSS****Syntax**

```
:SEARCh:IIC:ADDReSS <value>
:SEARCh:IIC:ADDReSS?
```

**Description**

Set or query IIC data address.

**Parameter**

| Name      | Type     | Range        | Default |
|-----------|----------|--------------|---------|
| < value > | Discrete | 0x0 --- 0xff | 0       |

**Remarks**

N/A

**Return Format**

Query returns an integer.

**Example**

```
:SEARCh:IIC:ADDReSS 4      /* Set IIC data address to 4*/
:SEARCh:IIC:ADDReSS?       /*Query returns 4 */
```

**3.21.37 :SEARCh:IIC:DIRection****Syntax**

```
:SEARCh:IIC:DIRection <value>
:SEARCh:IIC:DIRection?
```

**Description**

Set or query IIC read and write methods.

**Parameter**

| Name    | Type     | Range        | Default |
|---------|----------|--------------|---------|
| <value> | Discrete | {READ WRITE} | READ    |

**Remarks**

N/A

**Return Format**

Query returns READ or WRITE.

**Example**

```
:SEARCh:IIC:DIRection READ      /* Set IIC read and write mode to READ */
:SEARCh:IIC:DIRection?          /*Query returns READ*/
```

### 3.21.38 :SEARCh:IIC:SCLK:THReshold

**Syntax**

```
:SEARCh:IIC:SCLK:THReshold <value>
:SEARCh:IIC:SCLK:THReshold?
```

**Description**

Set or query IIC clock threshold threshold.

**Parameter**

| Name    | Type  | Range                                                   | Default |
|---------|-------|---------------------------------------------------------|---------|
| <value> | float | (-4 * Current voltage level -4 * Current voltage level) | 0       |

**Remarks**

N/A

**Return Format**

Query the threshold threshold for returning the current operation result in scientific counting form.

**Example**

```
:SEARCh:IIC:SCLK:THReshold 2    /* Set the IIC clock threshold to 2V */
:SEARCh:IIC:SCLK:THReshold?      /*Query returns 2.000000e+00*/
```

### 3.21.39 :SEARch:IIC:SDA:THReshold

#### Syntax

```
:SEARch:IIC:SDA:THReshold <value>
:SEARch:IIC:SDA:THReshold?
```

#### Description

Set or query IIC data threshold threshold.

#### Parameter

| Name    | Type  | Range                                                  | Default |
|---------|-------|--------------------------------------------------------|---------|
| <value> | float | (-4 * Current voltage level~4 * Current voltage level) | 0       |

#### Remarks

N/A

#### Return Format

Query the threshold threshold for returning the current operation result in scientific counting form.

#### Example

```
:SEARch:IIC:SDA:THReshold 2      /* Set the IIC data threshold to 2V */
:SEARch:IIC:SDA:THReshold?        /*Query returns 2.000000e+00*/
```

### 3.21.40 :SEARch:SPI:SCLK:SOURce

#### Syntax

```
:SEARch:SPI:SCLK:SOURce <source>
:SEARch:SPI:SCLK:SOURce?
```

#### Description

Set or query SPI clock source.

#### Parameter

| Name     | Type     | Range                                | Default  |
|----------|----------|--------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 | CHANnel1 |

| Name | Type | Range | Default |
|------|------|-------|---------|
|------|------|-------|---------|

EXT}

**Remarks**

N/A

**Return Format**

Query returns CHAN1,CHAN2,CHAN3,CHAN4 or EXT.

**Example**

```
:SEARCh:SPI:SCLK:SOURce CHANnel2      /* Set SPI clock signal source to CH2*/
:SEARCh:SPI:SCLK:SOURce?                /*Query returns CHAN2*/
```

### 3.21.41 :SEARCh:SPI:SCLK:SLOPe

**Syntax**

```
:SEARCh:SPI:SCLK:SLOPe <value>
:SEARCh:SPI:SCLK:SLOPe?
```

**Description**

Set or query SPI clock edge judgment.

**Parameter**

| Name      | Type     | Range            | Default |
|-----------|----------|------------------|---------|
| < value > | Discrete | {RISIng FALLIng} | RISIng  |

**Remarks**

N/A

**Return Format**

Query returns RISIng or FALLIng.

**Example**

```
:SEARCh:SPI:SCLK:SLOPe RISIng          /* Set SPI clock edge judgment to RISING */
:SEARCh:SPI:SCLK:SLOPe?                 /*Query returns RISIng*/
```

### 3.21.42 :SEARCh:SPI:MISO:SOURce

**Syntax**

```
:SEARCh:SPI:MISO:SOURce <source>
:SEARCh:SPI:MISO:SOURce?
```

**Description**

Set or query SPI MISO signal source.

**Parameter**

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

**Remarks**

N/A

**Return Format**

Query returns CHAN1,CHAN2,CHAN3,CHAN4 or EXT.

**Example**

```
:SEARCh:SPI:MISO:SOURce CHANnel2      /* Set SPI MISO signal source to CH2*/
:SEARCh:SPI:MISO:SOURce?                /*Query returns CHAN2*/
```

### 3.21.43 :SEARCh:SPI:CS:SOURce

**Syntax**

```
:SEARCh:SPI:CS:SOURce <source>
:SEARCh:SPI:CS:SOURce?
```

**Description**

Set or query SPI CS signal source.

**Parameter**

| Name     | Type     | Range                                     | Default  |
|----------|----------|-------------------------------------------|----------|
| <source> | Discrete | {CHANnel1 CHANnel2 CHANnel3 CHANnel4 EXT} | CHANnel1 |

**Remarks**

N/A

**Return Format**

Query returns CHAN1,CHAN2,CHAN3,CHAN4 or EXT.

**Example**

```
:SEARCh:SPI:CS:SOURce CHANnel2    /* Set SPI CS signal source to CH2*/
:SEARCh:SPI:CS:SOURce?             /*Query returns CHAN2*/
```

### 3.21.44 :SEARCh:SPI:CS:POLarity

#### Syntax

```
:SEARCh:SPI:CS:POLarity <value>
:SEARCh:SPI:CS:POLarity?
```

#### Description

Set or query SPI CS polarity.

#### Parameter

| Name    | Type     | Range               | Default  |
|---------|----------|---------------------|----------|
| <value> | Discrete | {POSitive NEGative} | POSitive |

#### Remarks

N/A

#### Return Format

Query returns POS or NEG.

#### Example

```
:SEARCh:SPI:CS:POLarity NEGative    /* Set SPI CS polarity to NEGative */
:SEARCh:SPI:CS:POLarity?             /*Query returns NEG*/
```

### 3.21.45 :SEARCh:SPI:WHEN

#### Syntax

```
:SEARCh:SPI:WHEN <value>
:SEARCh:SPI:WHEN?
```

#### Description

Set or query SPI trigger conditions.

#### Parameter

| Name    | Type     | Range        | Default |
|---------|----------|--------------|---------|
| <value> | Discrete | {CS TIMEOUT} | CS      |

#### Remarks

N/A

**Return Format**

Query returns CS or TIMEOUT.

**Example**

```
:SEARCh:SPI:WHEN CS          /* Set the SPI search time condition to CS */
:SEARCh:SPI:WHEN?           /*Query returns CS*/
```

**3.21.46 :SEARCh:SPI:DBITs****Syntax**

```
:SEARCh:SPI:DBITs <value>
:SEARCh:SPI:DBITs?
```

**Description**

Set or query SPI bit width.

**Parameter**

| Name    | Type    | Range        | Default |
|---------|---------|--------------|---------|
| <value> | Integer | { 4 8 16 32} | 4       |

**Remarks**

N/A

**Return Format**

Query returns 4,8,16 or 32.

**Example**

```
:SEARCh:SPI:DBITs 4          /* Set SPI bit width to 4 bits */
:SEARCh:SPI:DBITs?           /*Query returns 4*/
```

**3.21.47 :SEARCh:SPI:TIMEout:TIME****Syntax**

```
:SEARCh:SPI:TIMEout:TIME <value>
:SEARCh:SPI:TIMEout:TIME?
```

**Description**

Set or query SPI timeout duration.

**Parameter**

| Name    | Type    | Range   | Default |
|---------|---------|---------|---------|
| <value> | Integer | 8ns–12s | 10 μs   |

**Remarks**

N/A

**Return Format**

Query returns timeout duration.

**Example**

```
:SEARCh:SPI:TIMEout:TIME 8      /* Set SPI timeout duration to 8ns */
:SEARCh:SPI:TIMEout:TIME?        /*Query returns 8*/
```

**3.21.48 :SEARCh:SPI:DATA****Syntax**

```
:SEARCh:SPI:DATA <value>
:SEARCh:SPI:DATA?
```

**Description**

Set or query SPI data values.

**Parameter**

| Name    | Type    | Range | Default |
|---------|---------|-------|---------|
| <value> | Integer |       | 0       |

**Remarks**

N/A

**Return Format**

Query returns integer.

**Example**

```
:SEARCh:SPI:DATA 8      /* Set SPI data value to 8*/
:SEARCh:SPI:DATA?        /*Query returns 8*/
```

**3.21.49 :SEARCh:SPI:SCLK:THResold****Syntax**

```
:SEARCH:SPI:SCLK:THReshold <value>
```

```
:SEARCH:SPI:SCLK:THReshold?
```

### Description

Set or query SPI clock threshold threshold.

### Parameter

| Name    | Type  | Range                                                   | Default |
|---------|-------|---------------------------------------------------------|---------|
| <value> | float | (-4 * Current voltage level -4 * Current voltage level) | 0       |

### Remarks

N/A

### Return Format

Query the threshold threshold for returning the current operation result in scientific counting form.

### Example

```
:SEARCH:SPI:SCLK:THReshold 2 /* Set SPI clock threshold to 2V */
```

```
:SEARCH:SPI:SCLK:THReshold? /*Query returns 2.000000e+00*/
```

## 3.21.50 :SEARCH:SPI:MISO:THReshold

### Syntax

```
:SEARCH:SPI:MISO:THReshold <value>
```

```
:SEARCH:SPI:MISO:THReshold?
```

### Description

Set or query SPI MISO threshold threshold.

### Parameter

| Name    | Type  | Range                                                   | Default |
|---------|-------|---------------------------------------------------------|---------|
| <value> | float | (-4 * Current voltage level -4 * Current voltage level) | 0       |

### Remarks

N/A

### Return Format

Query the threshold threshold for returning the current operation result in scientific counting form.

Example

```
:SEARCh:SPI:MISO:THReshold 2      /* Set the SPI MISO threshold to 2V */
:SEARCh:SPI:MISO:THReshold?        /*Query returns 2.000000e+00*/
```

3.21.51 :SEARCh:SPI:CS:THReshold

Syntax

```
:SEARCh:SPI:CS:THReshold <value>
:SEARCh:SPI:CS:THReshold?
```

Description

Set or query SPI CS threshold threshold.

Parameter

| Name    | Type  | Range                                                   | Default |
|---------|-------|---------------------------------------------------------|---------|
| <value> | float | (-4 * Current voltage level -4 * Current voltage level) | 0       |

Remarks

N/A

Return Format

Query the threshold threshold for returning the current operation result in scientific counting form/.

Example

```
:SEARCh:SPI:CS:THReshold 2      /* Set the SPI CS threshold to 2V */
:SEARCh:SPI:CS:THReshold?        /*Query returns 2.000000e+00*/
```

## 3.22 **Programming Examples**

---

This chapter provides programming examples of how to use commands to implement common oscilloscope functions in Excel, Matlab, LabVIEW, Visual Basic 6.0, and Visual C++6.0 development environments. These instances are all based on VISA (Virtual) Implemented through programming with the Instrument Software Architecture library.

### **Programming preparation**

Before programming, you need to do the following preparations: First, please install Keysight IO universal software. You can log in to the Keysight official website to download the software, and then follow the instructions to install it. After installing Keysight IO, the VISA library has been automatically installed.

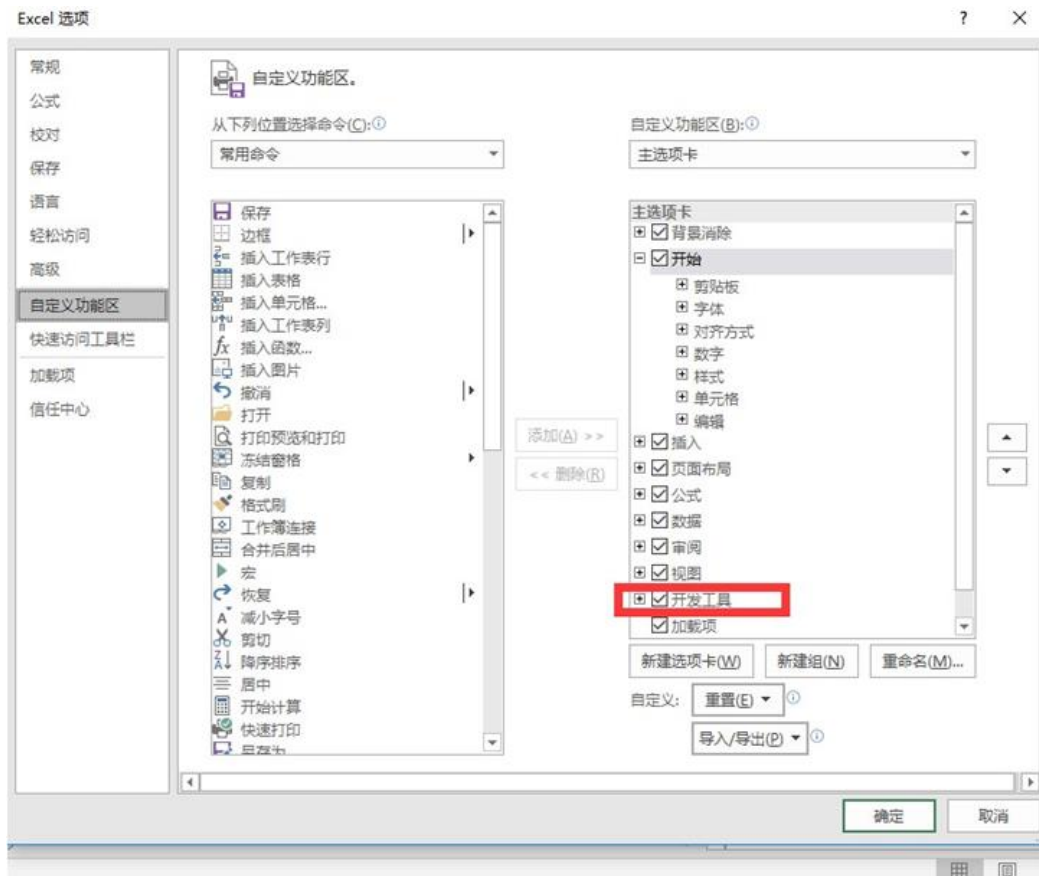
This article applies the USB interface of an oscilloscope to communicate with a PC. Please use a USB data cable to connect the USB Device interface on the back panel of the oscilloscope to the PC. After the oscilloscope is properly connected to the PC, power on the instrument and turn it on. The USB Device icon appears in the small right corner of the oscilloscope.

### **Excel Programming Example**

The program used in this example is Microsoft Excel 2016. The function implemented in this example is to send \* IDN? Command to read device information.

Create a new macro enabled Excel file, named MPO6000\_Semo.xlsx in this example. Run the MPO6000\_Semo.xlsx file, click the file button in the upper left corner of the Excel file, click "Options", open the interface shown in the figure below, select the custom ribbon, check "Development Tools" in the custom ribbon, and click "OK". At this point, the menu bar of Excel will display the "Development Tools" menu.





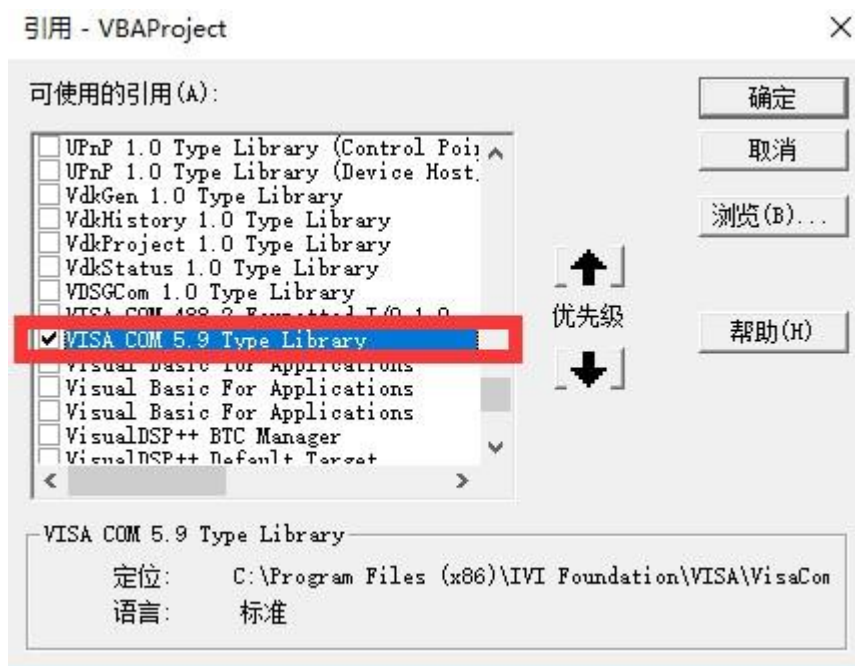
Click on the "Development Tools" menu and select the Visual Basic option to open Microsoft Visual Basic.



Select "Tools" from the menu bar on the Visual Basic page and click "Reference".



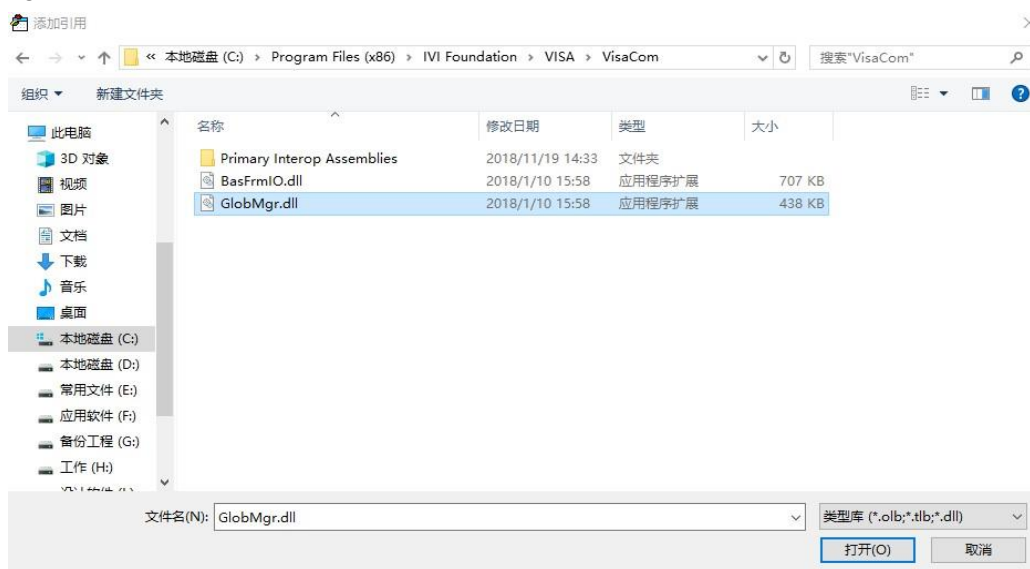
Select VISA COM 5.9 Type Library in the pop-up dialog box and click the OK button to reference VISA Library.



Explanation: If you cannot find VISA Library in the list on the left side of the above image, please follow the following method to search:

Please ensure that your computer has installed the VISA library.

Click on "Browse (B)..." on the right to search for the file named GlobMgr.dll in the range of C: \ Program Files (x86) \ VII Foundation \ VISA \ VisaCom, as shown in the following figure.



Click "View Code" under the "Development Tools" menu to enter the Microsoft Visual Basic page, add the following and save.

```
Private Sub CreateResource()
```

```
On Error GoTo errorhandler
```

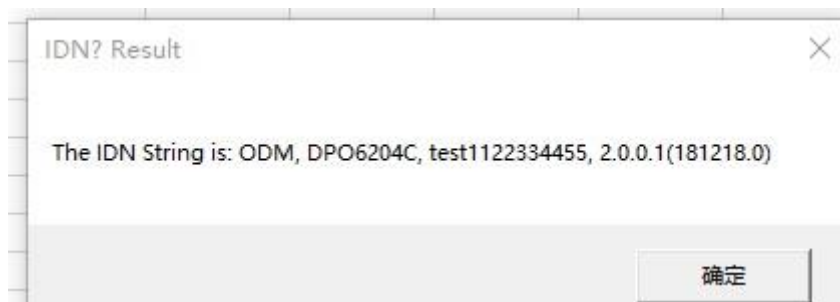
```
Dim rm As VisaComLib.ResourceManager
```

```
Dim session As VisaComLib.IMessage
```

```
Dim status As Long
```

```
Dim idn As String
```

```
' Instantiate the Global Resource Manager
Set rm = New VisaComLib.ResourceManager
' Open the session with the default values for the Lock (None), Timeout(N/A),
' and Option String (""). The return value is an IVisaSession, but the
' session variable is an IMessage interface reference, causing an implicit
' IUnknown::QueryInterface() to occur. VB handles the details. Set session =
rm.Open("USB0::0x049F::0x505E::test1122334455::0::INSTR") session.WriteString
"*IDN?" & vbLf idn = session.ReadString(1000)
MsgBox "The IDN String is: " & idn, vbOKOnly, "IDN? Result" Exit Sub errorhandler:
MsgBox Err.Description, vbExclamation, "Error Occurred", Err.HelpFile,
Err.HelpContext
End Sub
Click to run and the following dialog box will be displayed
```



### Matlab programming example

The program used in this example is MATLAB R2018a. The function implemented in this example is to read CH1 waveform data for FFT calculation and draw the waveform. Run Matlab software and modify the current path. This instance will modify the current path to G:\matlab\_work\SCPI\MPO6000\_Demo.



Click on the Matlab interface to create a new script and create a blank M file.



Add the following code to the M file:

```
%Create VISA Object
MPO6000 = visa('KEYSIGHT', 'USB0::0x049F::0x505E::test1122334455::0::INSTR');
%Set device properties, in this example, set the input cache length to 4096
MPO6000.InputBufferSize = 40000;
%Open MPO6000 device
fopen(MPO6000);
%Read oscilloscope acquisition parameters
```

```

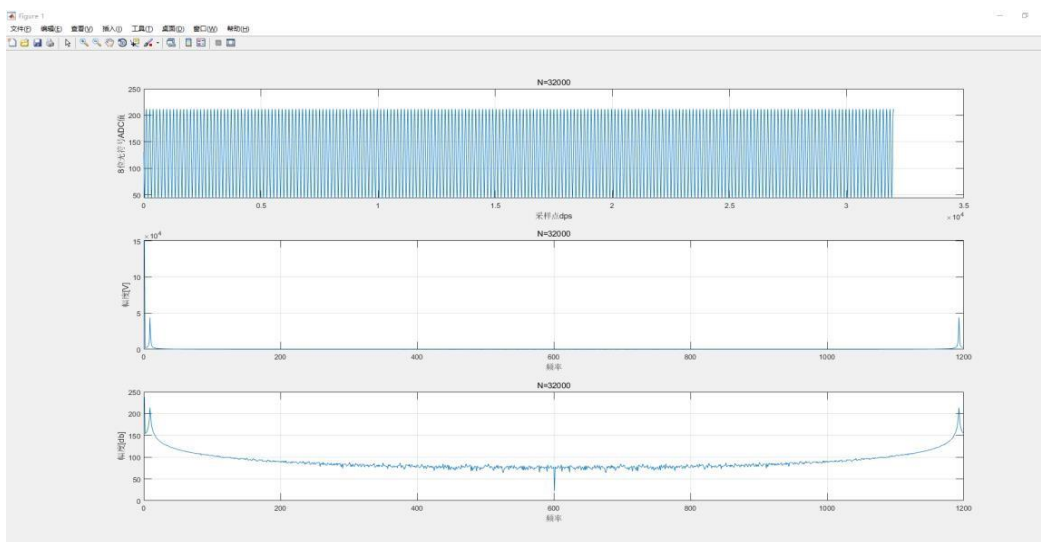
fprintf(MPO6000, ':wav:data:all?' );
%Request data
[data,len]= fread(MPO6000,4096 );
%The processing length of the waveform parameter header is 128 bytes,
tmc_head =strcat(data(1:2))';% Data [0] - data [1] (2 digits): Header # 9
cur_len=strcat( data (3:11) ');% Data [2] - data [10] (9-digit number): represents the byte
length of the current data packet
tot_len = strcat(data(12:20))';% Data [11] - data [19] (9-digit number): The total length
of bytes representing the amount of data
send_len = strcat(data(21:29))';% Data [20] - data [28] (9-digit number): represents the
byte length of the uploaded data
run_state = strcat(data(30))';% Data [29] (1 digit): indicates the current running status.
0 is paused
1 is running
trig_state = strcat(data(31))';% Data [30] (1 digit): indicates that the triggered state 0 is
no valid touch
Issue 1 has an effective trigger
ch1_offset = strcat(data(32:35))';% Data [31] - data [34] (4-digit number): represents
the offset of channel 1
ch2_offset = strcat(data(36:39))';% Data [35] - data [38] (4-digit number): represents
the offset of channel 2
ch3_offset = strcat(data(40:43))';% Data [39] - data [42] (4-digit number): represents
the offset of channel 3
ch4_offset = strcat(data(44:47))';% Data [43] - data [46] (4-digit number): represents
the offset of channel 4
CH1_voltage = strcat(data(48:54))';% Data [47] - data [53] (7-digit number):
represents the channel
The unit of voltage return in scientific and technological law is UV
CH2_voltage = strcat(data(55:61))';% Data [54] - data [60] (7-digit number):
represents the channel
The unit of voltage return in scientific and technological law is UV
CH3_voltage = strcat(data(62:68))';% Data [61] - data [67] (7-digit number):
represents the channel
The unit of voltage return in scientific and technological law is UV
CH4_voltage = strcat(data(69:78))';% Data [68] - data [74] (7-digit number):
represents the channel
The unit of voltage return in scientific and technological law is UV
ch_enabled = strcat(data(76:79))';% Data [75] - data [78] (4-digit number): represents
the channel status. Please refer to the instructions for details
sampling_rate = strcat(data(79:88))';% Data [79] - data [87] (9-digit number):
represents the sampling rate
extract_len = strcat(data(89:94))';% Data [88] - data [93] (6-digit number): represents
the sampling multiple
trig_time = strcat(data(95:103))';% Data [94] - data [102] (9-digit number): Display

```

```

trigger time of the current frame
start_time = strcat(data(104:112));% Data [103] - data [111] (9-digit number): current
frame display
Starting time of data collection at the starting point of the display
Reserve_data = strcat(data(113:128));% Data [112] - data [127] (16 digit number):
reserved bits
%The data read later is valid waveform data
send_len_data=str2num(send_len);% Convert string to number cur_1en_data=str2num
(cur_1en);% Convert string to number tot_1en_data=str2num (tot_1en);% Convert
string to number data_1en=1; While (send_1en_data+cur_1en_data<tot_1en_data)%
determines whether the read has ended
%Distribute and read waveforms for reading
fprintf(MPO6000, ':wav:data:all?' );
%Request data
[data,len]= fread(MPO6000,4096);
%Split waveform data header
tmc_head =strcat(data(1:2)) ;% Data [0] - data [1] (2 digits): Header # 9
cur_1en=strcat (data (3:11) );% Data [2] - data [10] (9-digit number): represents the byte
length of the current data packet
tot_1en = strcat(data(12:20));% Data [11] - data [19] (9-digit number): The total length
of bytes representing the amount of data
send_len = strcat(data(21:29));% Data [20] - data [28] (9-digit number): represents the
byte length of the uploaded data
send_len_data=str2num(send_len); cur_len_data=str2num(cur_len);
tot_len_data=str2num(tot_len); for i=30:1:len wave(data_len,1)=data(i);
data_len=data_len+1; end end
%Turn off the device
fclose(MPO6000); delete(MPO6000); clear MPO6000; subplot(311) plot(wave);
Xlabel ('sampling point dps'); Ylabel (8-bit unsigned ADC value); title('N=32000'); grid on;
fftSpec = fft(wave,1200); fftRms = abs(fftSpec); fftLg = 20*log(fftRms);
subplot(312); plot(fftRms); Xlabel ('frequency '); Ylabel ('amplitude [V] ');
title('N=32000'); grid on; subplot(313); plot(fftLg); Xlabel ('frequency '); Ylabel
('amplitude [db] '); title('N=32000'); grid on;
Save the M file in the current path. The M file of this instance is named
MPO6000_Semo. m.
Run the M file and display the following running result:

```



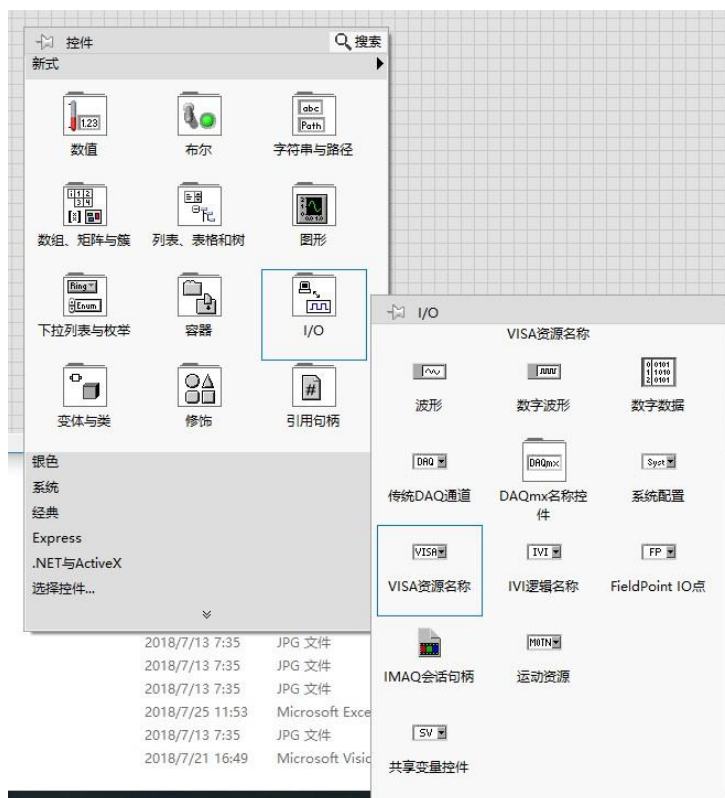
### LabVIEW Programming Example

The program used in this example is LabVIEW 2017. The function implemented in this example is to read the screen waveform data of CH1.

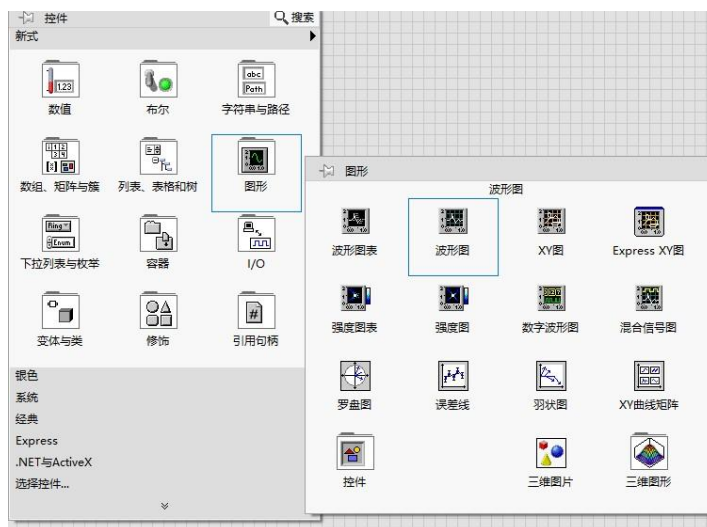
Run LabVIEW 2017 and create a new VI file named MPO6000\_Semo.



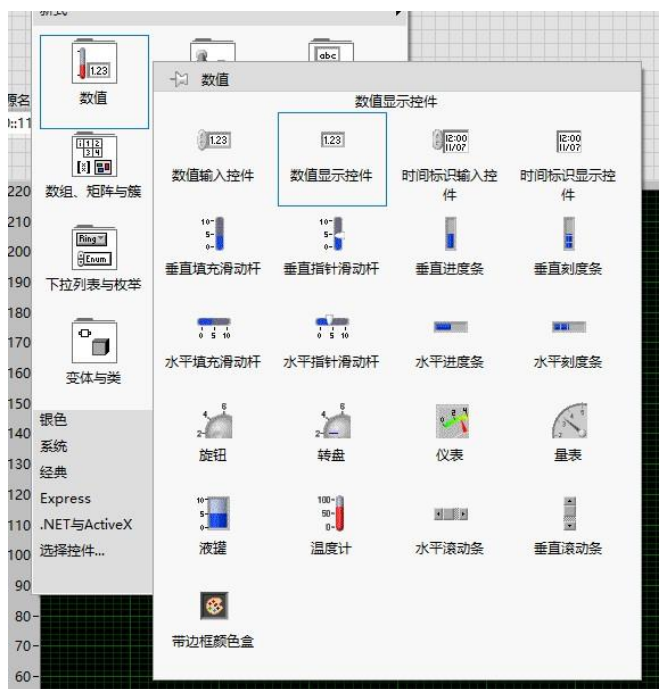
Add controls. Right click on the front panel interface and select the VISA resource name from the I/O menu, as shown in the following figure:



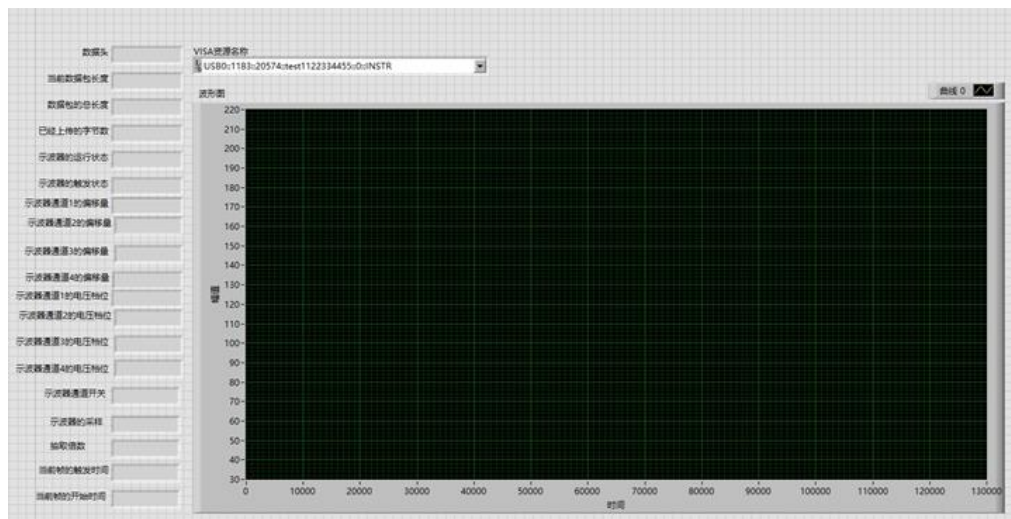
Add controls. Right click on the front panel interface and select the waveform diagram in I/O as shown in the following figure:



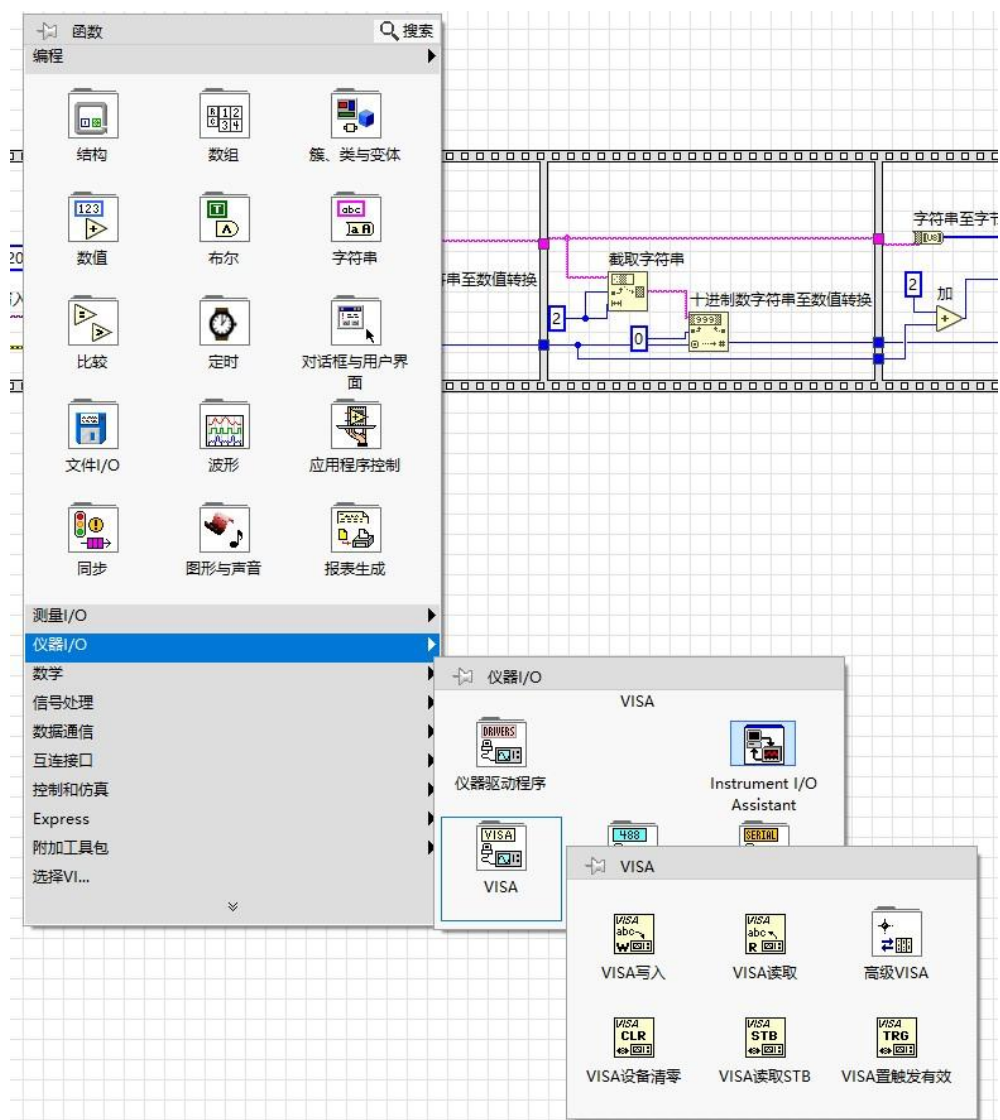
Create multiple numeric display controls for parsing data headers.



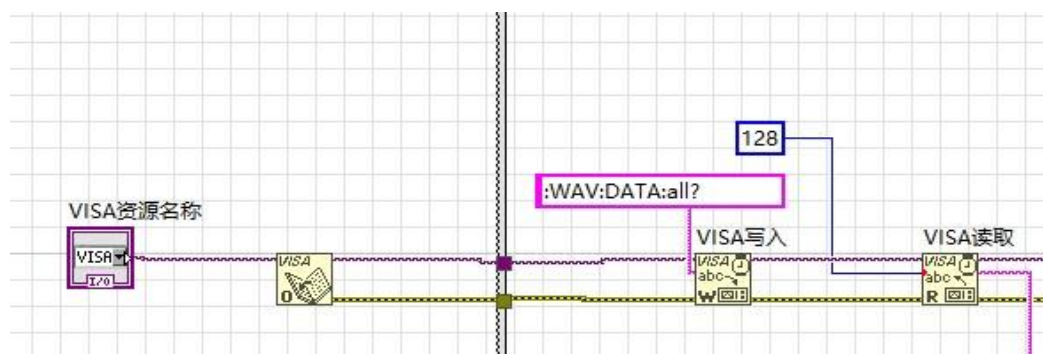
Complete the space addition as shown in the following figure:



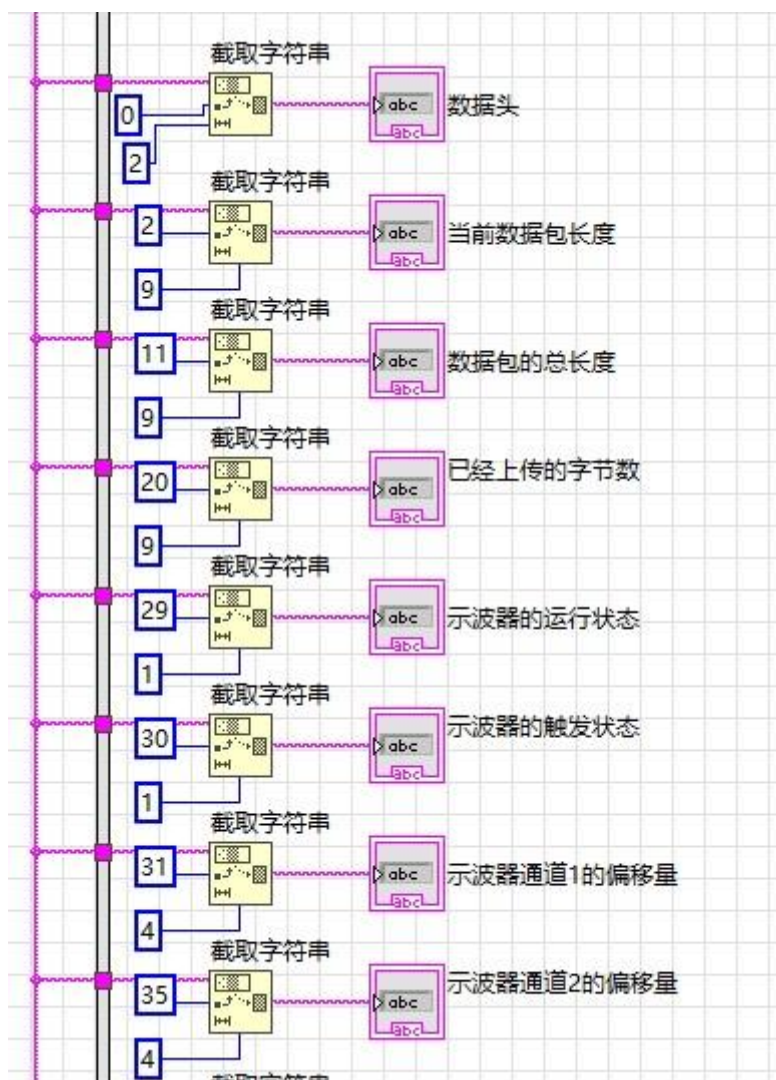
VISA open and VISA close functions.



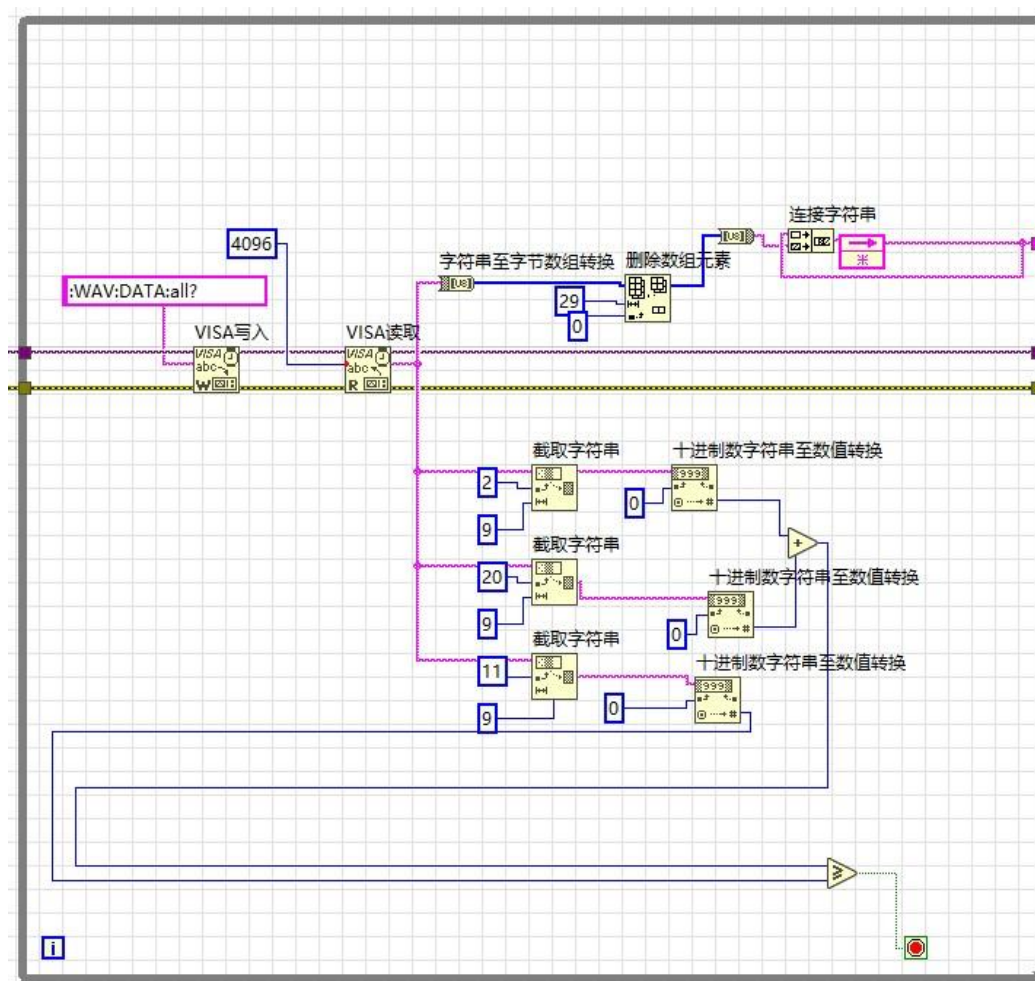
Connect the VISA resource name and VISA open, add read header data, as shown in the following figure:



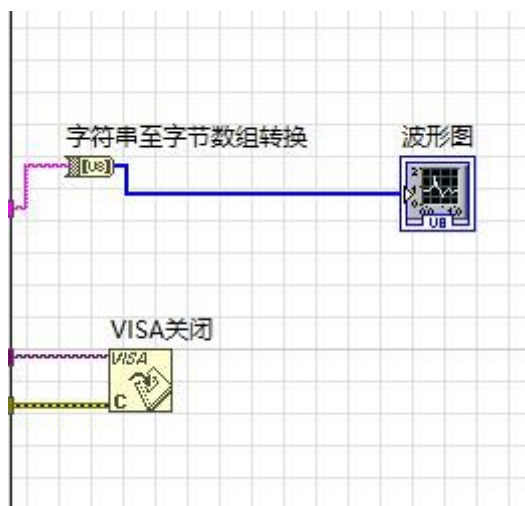
Add header data parsing processing, as shown in the following figure:



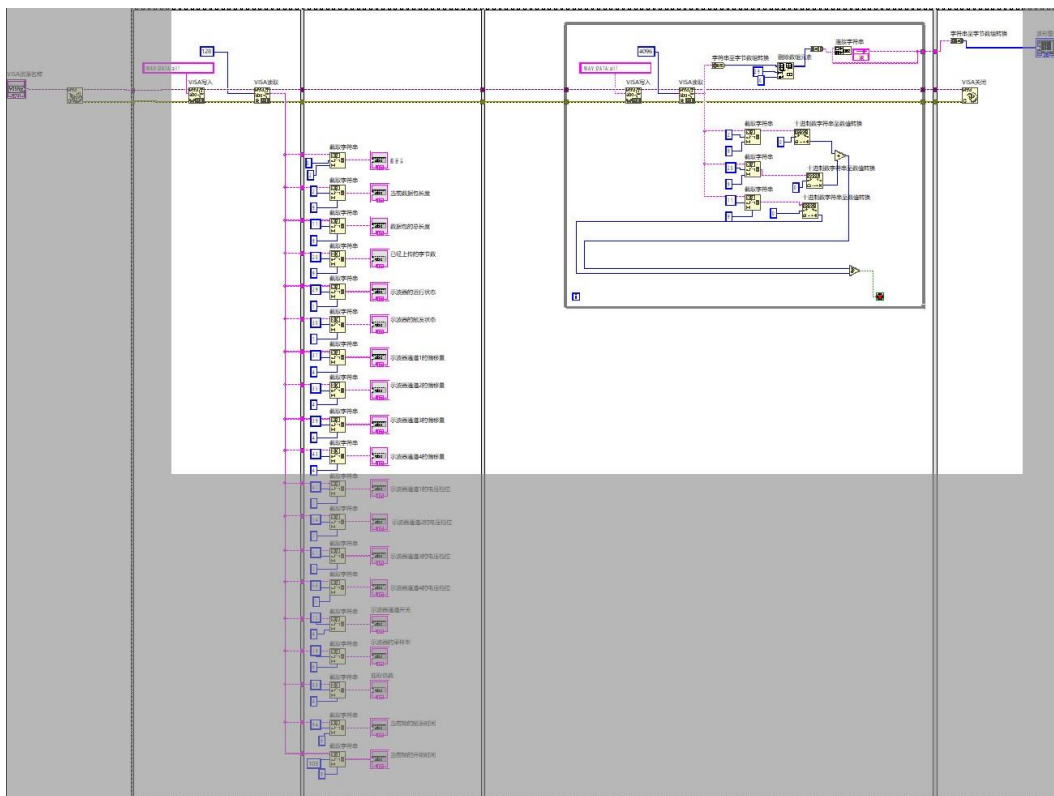
Add reading waveform data, {Note: To prevent reading errors, it is recommended to read the entire length}



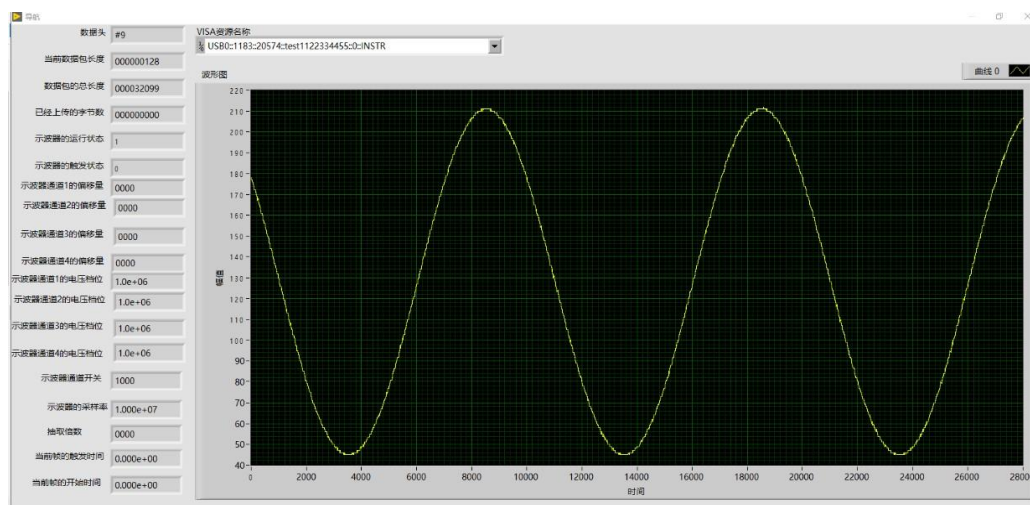
Add string conversion and waveform display module.



The complete program flowchart is shown below:



Select the device resource from the VISA resource name list box and start running.

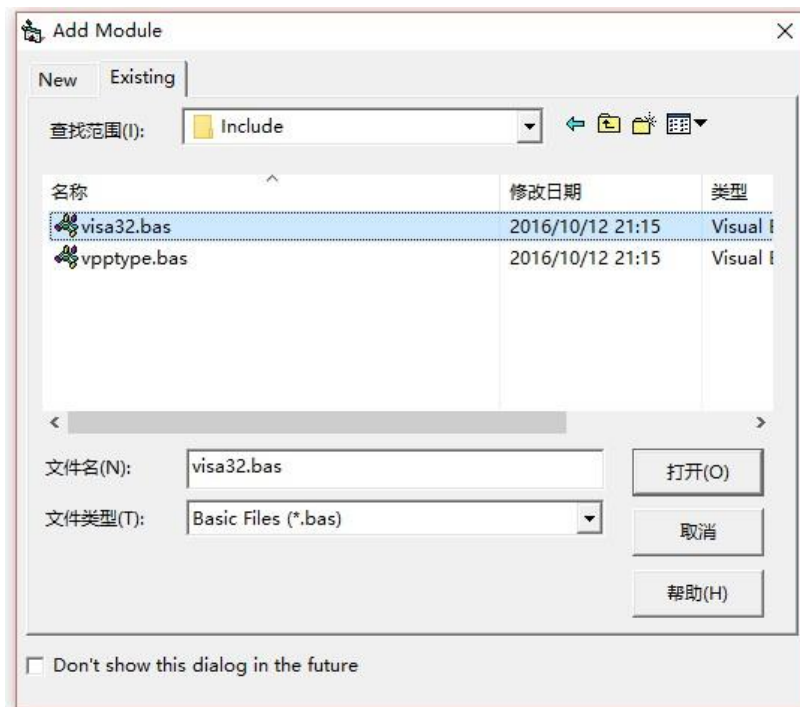


### Visual Basic Programming Example

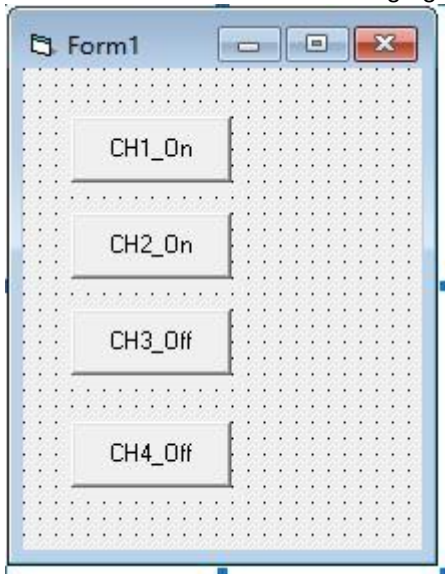
The program used in this example is Visual Basic 6.0. The function implemented in this example is to control the switch status of any channel. Enter the Visual Basic 6.0 programming environment and follow the steps below:

Establish a standard application project (Standard EXE) named MPO6000-Demo.

Open the Existing tab of ProjectAdd Module, locate the visa32.bas file in the include folder under the previous NI-VISA installation path, and add it.



Add the following four buttons in the demo, representing the channel switches for CH1 to CH4. As shown in the following figure:



Open the General tab in Project -> Project1 Properties, and select Form1 from the Startup Object drop-down menu.

Double click the CH1 button to enter the programming environment, and add the following code to achieve switch control of CH1~CH4. The following is the code for CH1, similar to other channel codes.

```
Private Sub Command1_Click()
    Dim nDevices As Long      ' Number of devices
    Dim Devices As String * 200 ' Device connection string
    Dim blsDisplay As Integer  ' Is the channel open
    Dim DefRM As Long
```

```

Dim vi As Long
Dim strRes As String * 200
Dim list As Long
' Open Visa default RM
Call viOpenDefaultRM(DefRM)
Call viFindRsrc(DefRM, "USB?*\"", list, nDevices, Devices)
' open device
Call viOpen(DefRM, Devices, 0, 0, vi)
' Send inquiry CH1 status command
Call viVPrintf(vi, ":CHANnel1:DISPlay?" + Chr$(10), 0)
' Get CH1 status
Call viVScanf(vi, "%t", strRes)      blsDisplay = CInt(strRes)
If (blsDisplay = 1) Then
' Send setting command
Call viVPrintf(vi, ":CHANnel1:DISPlay 0" + Chr$(10), 0)      Else
Call viVPrintf(vi, ":CHANnel1:DISPlay 1" + Chr$(10), 0)
End If
' close resource
Call viClose(vi)
Call viClose(DefRM)
End Sub

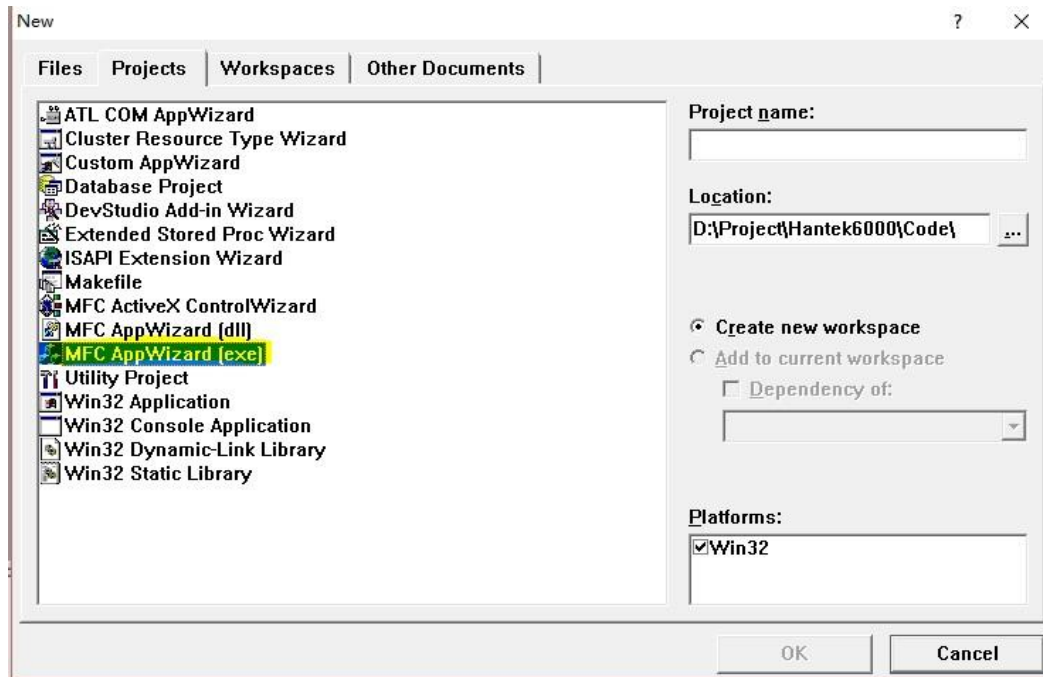
```

Save and run the entire project to obtain a single executable program for MPO6000\_Semo. When the oscilloscope is successfully connected to the PC, it can achieve on/off control of any channel.

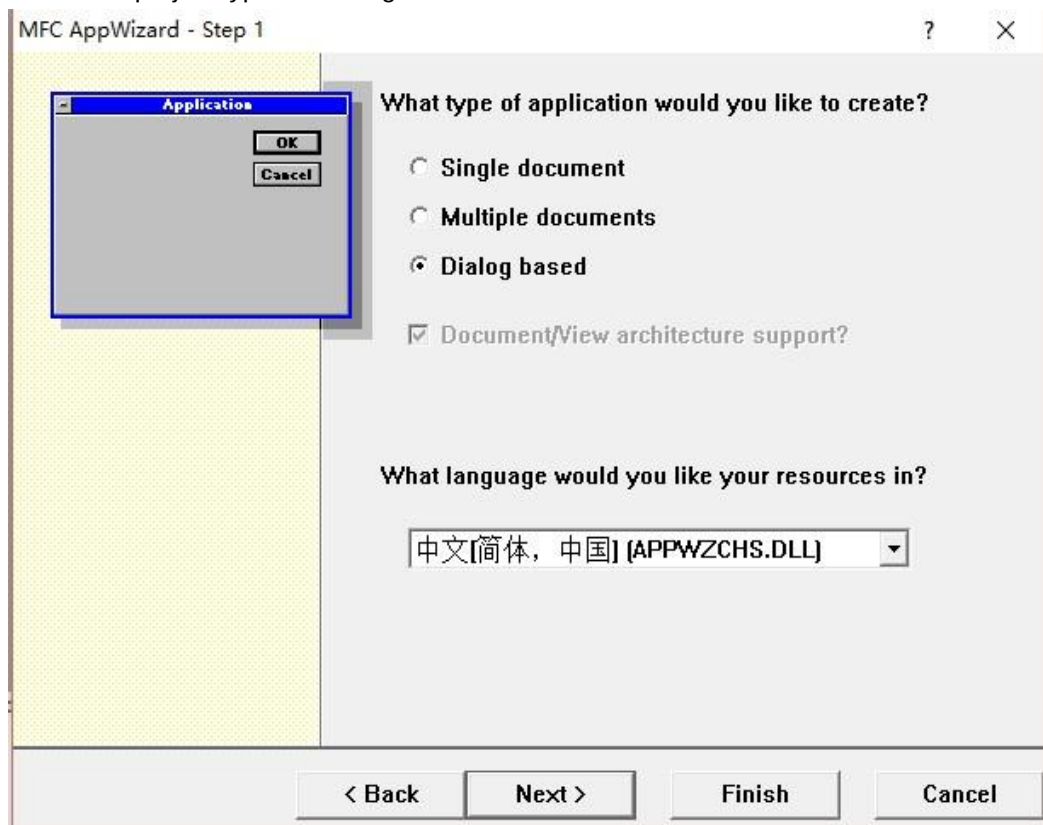
### Visual C++Programming Example

The program used in this example: Visual C++6.0

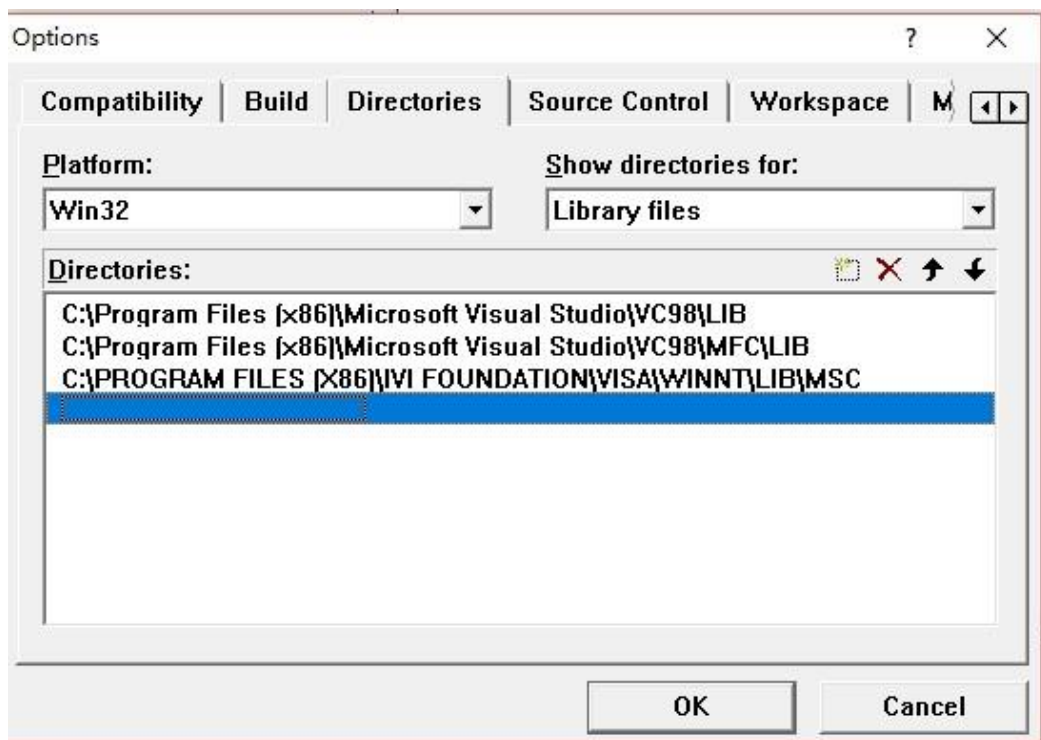
The function implemented in this example is to search for the instrument address, connect the instrument, send commands, and read the return value. Enter the Visual C++6.0 programming environment and follow the steps below: 1 Establish a dialog box based MFC project.



Select the project type as 'Dialog based '.



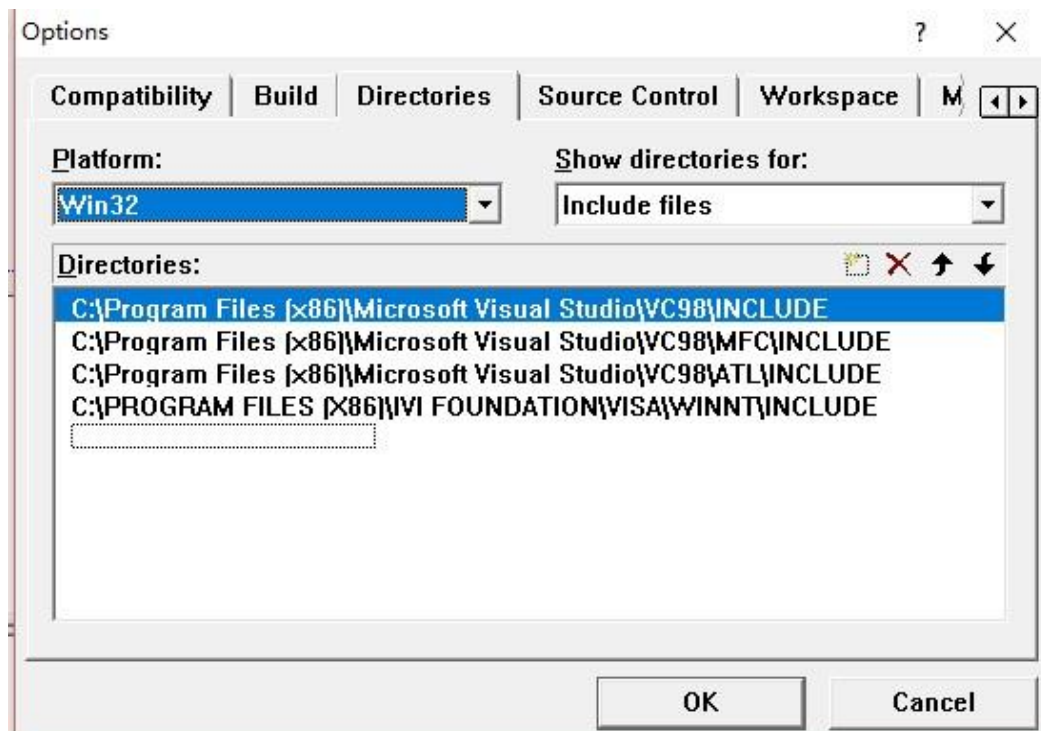
Open the Link tab in Project ->Settings and manually add visa32.lib in Object/library modules.



Open the Directors tab in Tools ->Options.

Select 'Include files' in ' Show directories for ', double-click on the blank space in the ' directories' checkbox to add Path to Include:

C:\ProgramFiles(X86)\IVIFoundation\VISA\WinNT\include. Select Library files in Show directories for, double-click the blank space in the directories checkbox to add the path to Lib: C:\Program Files(X86)\IVI Foundation\VISA\WinNT\lib\msc.



Note: At this point, the VISA library has been added.

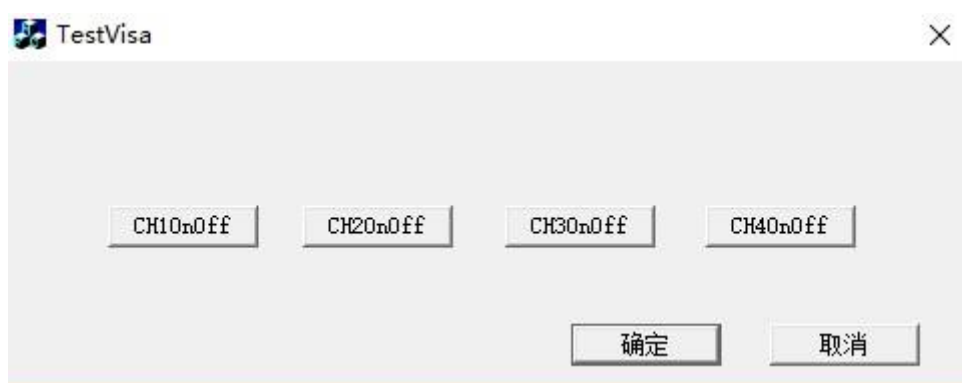
Add 4 Button controls. The layout is as follows:



Double click the "CH1OnOff" button and add the following code to its click event handling code. void CTestVisaDlg::OnButton1()

```
{
// TODO: Add your control notification handler code here      ViSession defaultRM, vi;
    char buff[256] = { 0 };      ViChar buffer[VI_FIND_BUFLLEN];
    ViRsrc Device = buffer;
    ViUInt32 nDevice;      ViFindList list;      viOpenDefaultRM(&defaultRM);
    // Obtain USB resources for Visa
    viFindRsrc(defaultRM, "USB?* ", &list, &nDevice, Device);      viOpen(defaultRM,
Device, VI_NULL, VI_NULL, &vi);
    // Send inquiry CH1 status command
    viPrintf(vi, ":CHANnel1:DISPlay?\n");      viScanf(vi, "%t\n", &buf);
    // Send setting command
    if (buff[0] == '1')
    {
        viPrintf(vi, ":CHANnel1:DISPlay 0\n");
    }
    else
    {
        viPrintf(vi, ":CHANnel1:DISPlay 1\n");
    }
    viClose(vi);      viClose(defaultRM);
}
```

Save, compile, and run the project to obtain a single executable file. When the oscilloscope is successfully connected to the PC, click "CH1OnOff" to control the on/off status of channel 1.





---

Addr: #35 Building, No. 780 Baoyuan Road, High-tech Zone, Qingdao, Shandong, China 266114

Switchboard: 400-036-7077

Email: [service@hantek.com](mailto:service@hantek.com)

Tel: (0086)532-55678770 & 55678772 & 55678773

Zip code: 266114

Website: [www.hantek.com](http://www.hantek.com)

Qingdao Hantek Electronics Co., LTD