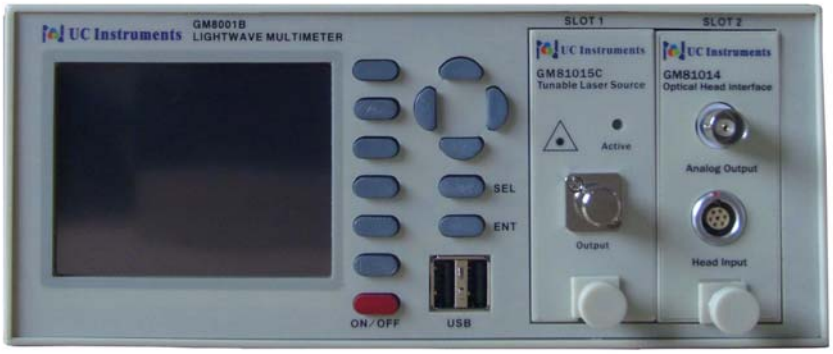
# GM8001B Lightwave Multimeter

# Programming Guide

Feb, 2011

(draft)




UC INSTRUMENTS CORP.

# Notices



# Subject Matter

The material in this document is subject to change without notice.

UC Instruments Corp. makes no warranty of any kind with regard to this printed material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

UC Instruments Corp. shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

# Warranty

This UC Instruments Corp. product is warranted against defects in material and workmanship for a period of one year from date of shipment. During the warranty period, UC INSTRUMENTS CORP. will, at its option, either repair or replace products that prove to be defective.

For warranty service or repair, this product must be returned to a service facility designated by UC Instruments Corp. Buyer shall prepay shipping charges to UC Instruments Corp. and UC Instruments Corp. shall pay shipping charges to return the product to Buyer. However, Buyer shall pay all shipping charges, duties, and taxes for products returned to UC Instruments Corp. from another country.

UC Instruments Corp. warrants that its software and firmware designated by UC Instruments Corp. for use with an instrument will execute its programming instructions when properly installed on that instrument. UC Instruments Corp. does not warrant that the operation of the instrument, software, or firmware will be uninterrupted or error free.

# Limitation of Warranty

The foregoing warranty shall not apply to defects resulting from improper or inadequate maintenance by Buyer, Buyer-supplied software or interfacing, unauthorized modification or misuse, operation outside of the environmental specifications for the product, or improper site preparation or maintenance.

No other warranty is expressed or implied. UC Instruments Corp. specifically disclaims the implied warranties of Merchantability and Fitness for a Particular Purpose.

# Exclusive Remedies

The remedies provided herein are Buyer's sole and exclusive remedies.

UC Instruments Corp. shall not be liable for any direct, indirect, special, incidental, or consequential damages whether based on contract, tort, or any other legal theory.

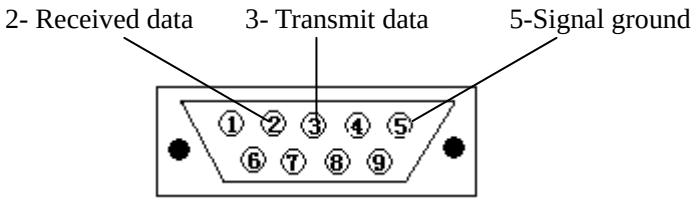
# Assistance

Product maintenance agreements and other customer assistance agreements are available for UC Instruments Corp. products. For any assistance contact the UC Instruments Corp. Maintenance Service Center.

# CONTENTS

# Communication Ports

## RS232 Serial Port




The DB9 connector and pin assignments for GM8001B

## Ethernet Port

IP address: the default is 192.168.0.200, revisable as required.
Subnet Mask: the default is 255.255.255.0, revisable as required.
Gateway: the default is 192.168.0.1, revisable as required.
Port number: 6000.

Communication setting:

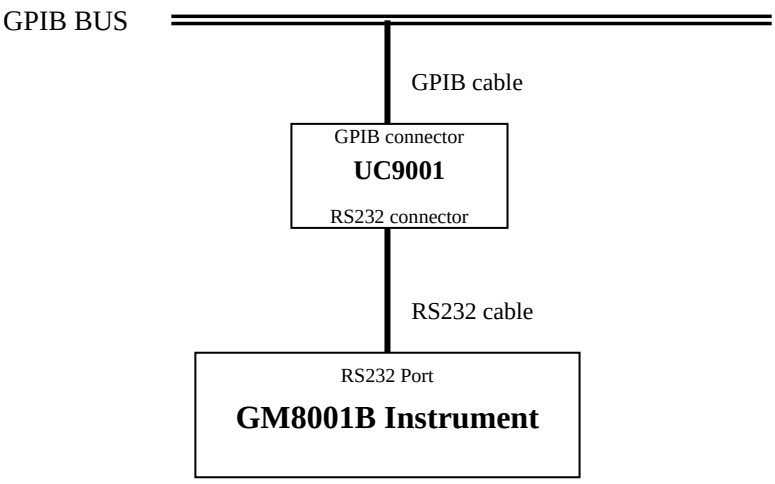
1 Start bit, 8 Data bit, 1 Stop bit, No parity checking. Baud rate: 115200 bps.

## GPIB Interface

The GM8001B Instruments uses UC Instruments's UC92001 GPIB-RS232 Convert Module to act as an extended GPIB port. The UC92001 meets the physical and electrical specifications for the IEEE 488 and can convert the signal between a RS232 port and a GPIB port.

If you want to know well the GPIB-RS232 Convert Module, refer to the UC92001 GPIB-RS232 Convert Module User Manual.

## Connection Diagram




## Setting the GPIB Address

You can set the GPIB address by using the GPIB-RS232 Convert Module.
See the UC92001 GPIB-RS232 Convert Module User Manual for more information.
The default GPIB address is 10.

# Syntax

## Commands format

The following symbols describe the syntax of commands in the following chapters.

The command is case-insensitive and can be written in upper case or in lower case or in both upper and lower case.

*Example* The command
*SOURCE1 : STATE OFF*
can also be written in lower case as
*source1 : state off*
or it can be written as
*SOURCE1 : State off*

Put a colon (**:**) before a component to indicate a move to the next level of the combination. The first colon can be left out for the first command or query in your message.

*Example* SOURCE1 : FREQUENCY : OFFSET ?

A command message is ended by a line feed character (LF) or <CR><LF>.

You can send several commands in a single message. Each command must be separated from the next one by a semicolon (;).

<…> The characters between angled brackets show the kind of data that you require, or that you get in a response. You don't type the angled brackets in the actual message.

[…] The characters between square brackets show optional information that you can include with the message.

/ The oblique line shows an either-or choice of data, for example, a/b means either a or b, but not both simultaneously.

Spaces are ignored, they can be inserted to improve readability.

All characters not between angled brackets are terminal symbols and must be sent exactly as shown. Items between angled brackets are not- terminal symbols, descriptions of these items follow the syntax description.

## Short Form and Long Form

The instrument accepts messages in short or long forms.
For example, the message
:STATUS:OPERATION:ENABLE 768
is in long form.

The short form of this message is
:STAT:OPER:ENAB 768

In general, the messages are written in a combination of upper and lower case. Upper case characters are used for the short form of the message.

For example, the above command would be written
:STATus:OPERation:ENABle

The first colon can be left out for the first command. That is, the example given above could also be sent as
STAT:OPER:ENAB 768

## Units

The following table lists the base units in use.

| Units | Default | Extended |
|---|---|---|
| Meter | nm | nm |
| Decibel | dB | dB |
| Second | s | ms, s |
| Decibel/1mW | dBm | dBm |
| Hertz | Hz | Hz |
| Watts | mW | pW, nW, uW, mW |

## Data type

| | |
|---|---|
| Boolean | this can be data (ON or OFF), or a number. In a response you get 0, for OFF, or 1, for ON. |
| Value | is an Integer or Float numeric data. |
| String | is ASCII data. |
| Unit | is one of DB, DBM, W (Watts), or S (seconds), or NM. It is the unit in a value. |

# Slot and Channel Numbers

Each module is identified by a slot number and a channel number. For commands that require you to specify a channel, the slot number is represented by [n] in a command and the channel number is represented by [m].

The slot number represents the module's position in the mainframe, it is from one to two for the GM8001B. These numbers are displays on the front panel on the top of each module slot.

Channel numbers apply to modules that have two inputs/outputs. Modules with two channels, use the channel number to distinguish between these channels.

For example, if you want to query slot 1, channel 2 with the command, SENS[n] CHAN[m]: POW : WAVE?

You should send:

SENS1 CHAN2: POW : WAVE?

**Note**: If you do not specify a slot or channel number, the lowest possible number is used as the default value. This means:

- Slot 1 for the GM8001B mainframes.
- Channel 1 for all channels.

# Commands Lists

## Common Commands

| Command | Function |
|---|---|
| *IDN? | Identification Query |
| *RST | Reset Command |
| *WAI | Wait Command |
| *CLS | Clear Status Command |
| *STB? | READ Status Byte Query |

***IDN?**

Syntax *IDN?

Response Identification Query
UC Instruments, GM8001B Lightwave Multimeter,
**SN:GG033616004,**HW Revision 1.00, **Firmware Revision 1.00
UC Instruments: Manufacturer
GM8001B Lightwave Multimeter: Instrument model
*****SN:GG013601004: Serial number of this instrument
*****HW Revision 1.00: Hardware revision
**Firmware Revision 3.00: Software revision

Description The *IDN? queries the instrument identification over the interface.

***RST**

Syntax *RST

Description Reset Command

The *RST command sets the mainframe and all modules to the reset setting (standard setting) stored internally.

The instrument is placed in the idle state awaiting a command. The *RST command clears the error queue.

The following are not changed:

- GPIB (interface) state
- Instrument interface address
- Output queue
- Service request enable register (SRE)
- Standard Event Status Enable Mask (SESEM)

***WAI**

Syntax *WAI

Description Wait Command

The *WAI command prevents the instrument from executing any further commands until the current command has finished executing. Some module firmware includes commands that set a "StatNOPC" flag during execution to indicate that the module is busy. *WAI blocks the GPIB bus to all commands until every module hosted by the instrument is no longer busy. All pending operations are completed during the wait period.

*** CLS**

Syntax *CLS

Description Clear Status Command

The *CLS command clears the following:

- Error queue
- Standard event status register (SESR)
- Status byte register (STB)

After the *CLS command the instrument is left waiting for the next command. The instrument setting is unaltered by the command.

*** STB?**

Syntax *STB?

Response <Value>
The bit value for the register (a 16-bit signed integer value):

| Bit | Mnemonic | Decimal Value |
|---|---|---|
| 7 (MSB) | Operation Status (OSB) | 128 |
| 6 | Not used | 0 |
| 5 | Event Status Bit (ESB) | 32 |
| 4 | Message Available (MAV) | 16 |
| 3 | Questionable Status (QSB) | 8 |
| 2 | Not used | 0 |
| 1 | Not used | 0 |
| 0 | Not used | 0 |

Description READ Status Byte Query
The *STB? command returns the contents of the Status Byte register.

Example *STB?
128

# Slot Commands

| Command | Function |
|---|---|
| **:Slot [*n*]** | |
| **:** EMPTy? | Returns whether the module slot is empty. |
| **:** IDN? | Returns information about the module. |
| **:** TST? | Returns the latest selftest results for a module. |

**:Slot [*n*] : EMPTy?**

Syntax :Slot[*n*] : EMPTy?

Response <*Boolean*>
0 There is a module in the slot.
1 The module slot is empty.

Description This command returns whether the module slot is empty.

Example SLOT1 : EMPT ?
0

**:Slot [*n*] : TST?**

| | |
|---|---|
| Syntax | :Slot[*n*] : TST? |
| Response | Returns an error code and a short description of the error. |
| Description | This command returns the latest selftest results for a module. |
| Example | SLOT1 : TST?<br>0 |

**:Slot [*n*] : IDN?**

| | |
|---|---|
| Syntax | :Slot[*n*] : IDN? |
| Response | GM81015C, serial number: CY13231845, VER: Software Revision 1.20E<br>GM81015C: Instrument model<br>serial number:GG0136 01004: Serial number of this instrument<br>Firmware Revision 1.20E: Firmware revision |
| Description | This command returns information about the module. |

# Source Commands

| Command | Function |
|---|---|
| **:SOUR**ce**[*n*] [:CHAN**n**el**[*m*]] : POW**er | |
| **:STAT**e? | |
| **:STAT**e | <Boolean> |
| **:UNIT?** | |
| **:UNIT** | <Units> |
| **:WAV**elength**?** | |
| **:WAV**elength | <Value> |
| **:SOUR**ce**[*n*] [:CHAN**n**el**[*m*]]** | |
| **:WAV**elength | |
| **:WAV**elength**? [Min**\| / \|**Max]** | <Value> |

**:SOURce[*n*] [:CHANnel[*m*]] : POWer : STATe?**

| | |
|---|---|
| Syntax | :SOURce[*n*] [:CHANnel[*m*]] : POWer : STATe? |
| Response | <Boolean><br>ON / 1: the laser is on. OFF / 0: the laser is off. |
| Description | This command returns the state of the chosen source output signal. |
| Example | SOUR1: POW: STAT?<br>ON |

**:SOURce[*n*] [:CHANnel[*m*]] : POWer : STATe**

| | |
|---|---|
| Syntax | :SOURce[*n*] [:CHANnel[*m*]] : POWer : STATe <Boolean> |
| Parameter | <Boolean><br>OFF / 0: disables the laser source, ON / 1: enables the laser source |
| Description | This command sets the state of the chosen source output signal. |
| Example | SOUR1: POW: STAT OFF |

**:SOURce[*n*] [:CHANnel[*m*]] : POWer : UNIT?**

| | |
|---|---|
| Syntax | :SOURce[*n*] [:CHANnel[*m*]] : POWer : UNIT? |
| Response | 0 / dBm: dBm,<br>1 / W: Watts |
| Description | This command returns the current power units. |
| Example | SOUR2 : POW : UNIT ?<br>0 |

**:SOURce[*n*] [:CHANnel[*m*]] : POWer : UNIT**

| | |
|---|---|
| Syntax | :SOURce[*n*] [:CHANnel[*m*]] : POWer : UNIT <Units> |
| Parameter | <Units><br>0 / dBm: dBm, 1 / W: Watts |
| Description | This command sets the power units. |
| Example | SOUR2 : POW : UNIT W |

**:SOURce[*n*] [:CHANnel[*m*]] : POWer :WAVelength?**

Syntax :SOURce[*n*] [:CHANnel[*m*]] : POWer :WAVelength?

Response
LOW The lower wavelength source
UPP The upper wavelength source
BOTH Both wavelength sources

Description This command returns the wavelength source for a dual-wavelength laser source.

Example
SOUR2 : POW : WAV?
UPP

**:SOURce[*n*] [:CHANnel[*m*]] : POWer :WAVelength**

Syntax :SOURce[*n*] [:CHANnel[*m*]] : POWer :WAVelength <LOW/UPP/ BOTH>

Parameter
LOWer / 1 The lower wavelength source
UPPer / 2 The upper wavelength source
BOTH / 3 Both wavelength sources

Description This command sets the wavelength source for a dual-wavelength laser source.

Example SOUR2 : POW : WAV 3

**:SOURce[*n*] [:CHANnel[*m*]] :WAVelength? [Min| / |Max]**

Syntax :SOURce[*n*] [:CHANnel[*m*]] :WAVelength? [Min| / |Max]

Response
The wavelength as a float value in meters.
Also allowed, for tunable lasermodules only, are
MIN: minimum wavelength MAX: maximum wavelength

Description This command returns the wavelength value in meters.

Example

| | |
|---|---|
| SOUR1:WAV?<br>+1.5672030E-006 | Returns the current wavelength value for a tunable laser module in slot 1. |
| SOUR1:WAV? MIN<br>+1.5500000E-006 | Returns minimum wavelength for a tunable laser module in slot 1. |

**:SOURce[*n*] [:CHANnel[*m*]] :WAVelength**

Syntax :SOURce[*n*] [:CHANnel[*m*]] :WAVelength <Value>

Parameter <Value>, The value is in nm.

Description This command sets the absolute wavelength of the output.

Example SOUR2 : WAV 1550nm

## Source Sweep Commands

| Command | Function |
|---|---|

**:SOURce[*n*] [:CHANnel[*m*]] :WAVelength : SWEep**
**:STARt ?**
**:STARt** <*Value*>
**:STOP ?**
**:STOP** <*Value*>
**:STEP [:WIDTh]** <*Value*>
**:STEP [:WIDTh]?**
**[:STATe]** <*Value*>

**:SOURce[*n*] [:CHANnel[*m*]] :WAVelength : SWEep : STARt ?**

Syntax :SOURce[*n*] [:CHANnel[*m*]] :WAVelength : SWEep : STARt ?

Response The wavelength at which the sweep starts as a float value in meters.

Description This command returns the starting point of the sweep.

Example SOUR1: WAV : SWE : STAR?
+1.50000000E-006

### :SOURce[*n*] [:CHANnel[*m*]] :WAVelength : SWEep : STARt

Syntax :SOURce[n] [:CHANnel[m]] :WAVelength : SWEep : STARt <*Value*>

Parameter <*Value*>
The wavelength at which the sweep starts as a float value. If you specify no units in your command, meters are used as the default.

Description This command sets the starting point of the sweep.

Example SOUR1:WAV : SWE : STAR 1500nm

### :SOURce[*n*] [:CHANnel[*m*]] :WAVelength : SWEep : STOP ?

Syntax :SOURce[*n*] [:CHANnel[*m*]] :WAVelength : SWEep : STOP ?

Response The wavelength at which the sweep ends as a float value in meters.

Description This command returns the end point of the sweep.

Example SOUR1: WAV : SWE : STOP?
+1.55000000E-006

### :SOURce[*n*] [:CHANnel[*m*]] :WAVelength : SWEep : STOP

Syntax :SOURce[*n*] [:CHANnel[*m*]] :WAVelength : SWEep : STOP <*Value*>

Parameter The wavelength at which the sweep ends as a float value in meters. If you specify no units in your command, meters are used as the default.

Description This command sets the end point of the sweep.

Example SOUR1: WAV : SWE : STOP 1550nm

### :SOURce[n] [:CHANnel[m]] :WAVelength : SWEep : STEP [:WIDTh]

Syntax :SOURce[*n*] [:CHANnel[*m*]] :WAVelength : SWEep : STEP [:WIDTh] <*Value*>

Parameter <*Value*>
The width of the sweep step as a float value. The units are PM, NM, UM, MM or M. If you specify no units in your command, meters are used as the default.

Description This command sets the width of the sweep step.

Example SOUR1: WAV : SWE : STEP 5nm

### :SOURce[n] [:CHANnel[m]] :WAVelength : SWEep : STEP [:WIDTh] ?

Syntax :SOURce[*n*] [:CHANnel[*m*]] :WAVelength : SWEep : STEP [:WIDTh] ?

Response *< Float value>*
The width of the sweep step. The units are PM, NM, UM, MM or M.

Description This command returns the width of the sweep step.

Example SOUR1: WAV : SWE : STEP ?
5nm

### :SOURce[*n*] [:CHANnel[*m*]] :WAVelength : SWEep [:STATe]

Syntax :SOURce[*n*] [:CHANnel[*m*]] :WAVelength : SWEep [:STATe] <*Value*>

Parameter <*Value*>

| | |
|---|---|
| 0 / STOP | Stop the sweep. |
| 1 / STARt | Start a sweep, run sweep. |
| 2 / PAUSe | Pause the sweep. |
| 3 / CONTinue | Continue a sweep. |

Description This command stops, start or pauses a wavelength sweep.

Example SOUR1: WAV : SWE STOP

# Sensor Commands

| Command | Function |
|---|---|
| **: FETC**h**[*n*] [:CHAN**nel**[*m*]] : POW**er?** | |
| **: INIT**iate **[*n*] [:CHAN**nel**[*m*]] [:IMM**ediate**]** | |
| **: INIT**iate **[*n*] [:CHAN**nel **[*m*]] : CONT**inuous**?** | |
| **: INIT**iate **[*n*] [:CHAN**nel **[*m*]] : CONT**inuous **<*Boolean*>** | |
| **: READ[*n*] [:CHAN**nel**[*m*]] [:SCAL**ar**] : POW**er **: ALL?** | |
| **: READ[*n*] [:CHAN**nel**[*m*]] [:SCAL**ar**] : POW**er **[:DC]?** | |
| **: SENS**e**[*n*] [:CHAN**nel**[*m*]]** | |
| **: CORR**ection**?** | |
| **: CORR**ection <*Value*> | |
| **: POW**er **: ATIM**e**?** | |
| **: POW**er **: ATIM**e <*Averaging time*> | |
| **: POW**er **: RANG**e**?** | |
| **: POW**er **: RANG**e <*Value*> | |

### :FETCh[*n*] [:CHANnel [*m*]] : POWer?

Syntax :FETCh[*n*] [:CHANnel [*m*]] : POWer?

Response The current value as a float value in dBm,W or dB.
If the reference state is absolute, units are dBm or W.
If the reference state is relative, units are dB.

Description This command reads the current power meter value (It does not provide its own triggering and so must be used with either continuous software triggering (See ":INIT[*n*]:CONT?" ) or a directly preceding immediate software trigger (See ":INIT[*n*]:IMM")

Example FETC1: POW?
+6.73370400E-004

### :INITiate[*n*] [:CHANnel [*m*]] [:IMMediate]

Syntax :INITiate[*n*] [:CHANnel [*m*]] [:IMMediate]

Description This command initiates the software trigger system and completes one full trigger cycle, that is, one measurement is made.

Example INIT

**:INIT**iate**[*n*] [:CHAN**nel **[*m*]] :CONT**inuous**?**

Syntax :INITiate[*n*] [:CHANnel [*m*]]:CONTinuous?

Response <*Boolean*>

0 / OFF measurement is not continuous.

1 / ON measurement is continuous.

Description This command queries whether the software trigger system operates continuously or not.

Example INIT2 : CONT?
1

**:INIT**iate**[*n*] [:CHAN**nel **[*m*]] :CONT**inuous

Syntax :INITiate[*n*] [:CHANnel [*m*]]:CONTinuous <*Boolean*>

Parameter <*Boolean*>

0 / OFF do not measure continuously

1 / ON measure continuously

Description This command sets the software trigger system to continuous measurement mode.

Example INIT2 : CONT 1

**:READ[*n*] [:CHAN**nel**[*m*]] [:SCAL**ar**] :POW**er **: ALL?**

Syntax :READ[*n*] [:CHANnel[*m*]] [:SCALar] : POWer : ALL?

Response 4-byte Intel float values in a binary block in Intel byte order. The values are ordered by slot and channel order.
**Note:** Data values are always in Watt.

Description This command reads all available power meter channels. It provides its own software triggering and does not need a triggering command.

**Note:** The power meters must be running for this command to be effective.

Example READ1:POW:ALL?
interpreted as
+1.33555600E-006|+1.34789100E-006|+1.37456900E-006

### :READ[*n*] [:CHANnel[*m*]] [:SCALar] :POWer[:DC]?

Syntax :READ[*n*] [:CHANnel[*m*]] [:SCALar] : POWer[:DC]?

Response The current power meter reading as a float value in dBm, W or dB.
**Note:** If the reference state is absolute, units are dBm or W.
If the reference state is relative, units are dB.

Description This command reads the current power meter value. It provides its own software triggering and does not need a triggering command.

Example READ1: POW?
+1.33555600E-006

### :SENSe[*n*][:CHANnel[*m*]] : CORRection?

Syntax :SENSe[*n*][:CHANnel[*m*]] : CORRection?

Response The calibration factor as a float value. The units is either dB or MDB.
If no unit type is specified, the dB is implied.

Description The calibration factor as a float value. Units are in dB, although no units are returned in the response message.

Example SENS1: CORR?
+1.00000000E+000

### :SENSe[*n*][:CHANnel[*m*]] : CORRection

Syntax :SENSe[*n*][:CHANnel[*m*]] : CORRection <*Value*>

Parameter The calibration factor as a float value. The units is either DB or MDB.
If no unit type is specified, the dB is implied.

Description This command enters a calibration value for a module.

Example SENS1: CORR 10DB

### :SENSe[*n*][:CHANnel[*m*]] : POWer : ATIMe?

Syntax :SENSe[*n*][:CHANnel[*m*]] : POWer : ATIMe?

Response The averaging time as a float value in seconds.

Description This command returns the averaging time for the module.

Example SENS1:POW:ATIM ?
+1.00000000E+00

### :SENSe[*n*][:CHANnel[*m*]] : POWer : ATIMe

Syntax :SENSe[*n*][:CHANnel[*m*]] : POWer : ATIMe <Averaging time>

Parameter The averaging time as a float value in nS, uS, mS, or S. If you specify no units in your command, S is used as the default.

Description This command sets the averaging time for the module.

Example SENS1:POW:ATIM 100ms

### :SENSe[*n*][:CHANnel[*m*]] : POWer : RANGe?

Syntax :SENSe[*n*][:CHANnel[*m*]] : POWer : RANGe ?

Response The range setting as a float value in dBm.

Description This command returns the power range setting for the module. For a return loss module, returns the power range of the return loss diode.

Example SENS1:POW:RANG?
-2.00000000E+001

### :SENSe[*n*][:CHANnel[*m*]] : POWer : RANGe

Syntax :SENSe[*n*][:CHANnel[*m*]] : POWer : RANGe <*Value*>

Parameter The range as a float value in dBm. The number is rounded to the closest multiple of 10, because the range changes at 10dBm intervals. Units are in dBm.

Description This command sets the power range setting for the module. For a return loss module, sets the power range of the return loss diode.

Example SENS1 : POW : RANG -20dBm

# Triggering Commands

| Command | Function |
|---|---|
| **:TRIG**ger | |
| **:TRIG**er**[*n*] [:CHAN**nel **[*m*]]** | |
| **: INP**ut | |
| **: INP**ut**?** | |
| **: INP**ut **: REAR**m | |
| **: INP**ut **: REAR**m**?** | |
| **:CONF**iguration | |
| **:CONF**iguration**?** | |

**:TRIG**er

Syntax :TRIGer <*Value*>

Parameter <Value>

| | |
|---|---|
| 1 / NODEA | Is identical to a trigger at the Input Trigger Connector. |
| 2 / NODEB | Generates trigger at the Output Trigger Connector. |

Description This command generates a hardware trigger.

Example TRIG 1

**:TRIG**er**[*n*] [:CHAN**nel [*m*]] : INP**ut

Syntax :TRIGer[*n*] [:CHANnel [*m*]] : INPut <*Trigger response*>

Parameter <Trigger response>

| | |
|---|---|
| IGN | Ignore incoming trigger |
| SME | Start a single measurement. |
| CME | Start a complete measurement. |
| NEXTstep | Perform next step of a stepped sweep. |
| SWStart | Start a sweep cycle |

Description This command sets the incoming trigger response and arms the module.

Example TRIG1 : INP IGN

**:TRIGer[*n*] [:CHANnel [*m*]] : INPut?**

Syntax :TRIGer[*n*] [:CHANnel [*m*]] : INPut?

Response <Trigger response>

| | |
|---|---|
| IGN | Ignore incoming trigger |
| SME | Start a single measurement. |
| CME | Start a complete measurement. |
| NEXTstep | Perform next step of a stepped sweep. |
| SWStart | Start a sweep cycle |

Description This command returns the incoming trigger response.

Example TRIG1 : INP?
IGN

**:TRIGer[*n*] [:CHANnel [*m*]] : INPut : REARm**

Syntax :TRIGer[*n*] [:CHANnel [*m*]] : INPut : REARm < *Boolean* >

Parameter <Boolean>

| | |
|---|---|
| OFF / 0 | trigger rearming disabled. |
| ON / 1 | trigger rearming enabled (default). |

Description This command sets the arming response of a channel to an incoming trigger.

Example TRIG1 : INP : REARM 0

**:TRIGer[*n*] [:CHANnel [*m*]] : INPut : REARm?**

Syntax :TRIGer[*n*] [:CHANnel [*m*]] : INPut : REARm?

Response A Boolean value:

| | |
|---|---|
| OFF / 0 | trigger rearming disabled. |
| ON / 1 | trigger rearming enabled (default). |

Description This command returns the arming response of a channel to an incoming trigger.

Example TRIG1 : INP : REARM ?
0

**:TRIG**er **: CONF**iguration

Syntax :TRIGer : CONFiguration <*Trigger mode*>

Parameter <Trigger mode>

| | |
|---|---|
| 0 / DISabled | Trigger connectors are disabled. |
| 1 / DEFault | The Input Trigger Connector is activated, the incoming trigger response for each slot ":TRIGger[n][:CHANnel[m]]:INPut" determines how each slot responds to an incoming trigger, all slot events can trigger the Output Trigger Connector. |
| 2 / PASSthrough | The same as 1/DEFault but a trigger at the Input Trigger Connector generates a trigger at the Output Trigger Connector automatically. |
| 3 / LOOPback | The same as 1/DEFault but a trigger at the Output Trigger Connector generates a trigger at the Input Trigger Connector automatically. |

Description This command sets the hardware trigger configuration with regard to Output and Input Trigger Connectors.

Example TRIG : CONF : DIS

**:TRIG**er **: CONF**iguration**?**

Syntax :TRIGer : CONFiguration?

Response <Trigger mode>

| | |
|---|---|
| DIS | Trigger connectors are disabled. |
| DEF | The Input Trigger Connector is activated, the incoming trigger response for each slot ":TRIGger[n][:CHANnel[m]]:INPut" determines how each slot responds to an incoming trigger, all slot events can trigger the Output Trigger Connector. |
| PASS | The same as DEF but a trigger at the Input Trigger Connector generates a trigger at the Output Trigger Connector automatically. |
| LOOP | The same as DEF but a trigger at the Output Trigger Connector generates a trigger at the Input Trigger Connector automatically. |

| | |
|---|---|
| CUSTOM | A custom configuration is active. |

Description This command returns hardware trigger configuration.

Example
```
TRIG : CONF?
DIS
```

## Data and File Process Commands

| Command | Function |
|---|---|
| FILE : ID < STRING > ? | |
| FILE : LIST ? | |
| FILE : DEL | < STRING > |

**FILE : ID < STRING > ?**

Syntax FILE : ID < STRING > ?

Description This command gets the details of a sweeping file specified by the file number. STRING represents the file number.

Example FILE : ID 405 ?

**FILE : LIST ?**

Syntax FILE : LIST ?

Description This command returns a list in which the file numbers of all sweeping files are.

**FILE : DEL**

Syntax FILE : DEL < STRING >

Description This command deletes a sweeping file specified by the file number. STRING represents the file number.

Example FILE : DEL 405

# UC INSTRUMENTS CORP. CONTACT INFORMATION

***UC INSTRUMENTS CORPORATION***
***Add: 37498 Glenmoor Dr., Fremont, CA 94536 USA***
***Tel: +1-510-366-7353***
***Fax: +1-510-353-1809***
***Email: sales@ucinstruments.com***
***Website: www.ucistruments.com***

***MAINTENANCE SERVICE CENTER,GUILIN,CHINA***
***Address： No.16 Yi Feng south road, GuiLin, China***
***Tel: +86 773 5850657***
***Fax: +86 773 5814532***
***Website: www.ucigl.com***