RemoSens™

Network Interface for USB Power Sensors

Manufacturer:

IFD – Ing.-Büro für Datentechnik
Jürgen D. Geltinger
Eugenbacher Str. 93a
DE-84032 Altdorf
Germany
Phone +49-871-932360
Fax +49-871-9323648
Email info@ifd.de

Table of Contents

Preface....4
Disclaimer of liability....5
Introduction....6
  General....6
Intended use....7
  Avoid distance limitation....7
  Avoid proprietary drivers....7
  Utilize standard SCPI commands....8
Placing into Operation....9
  Preliminary Note....9
  Power Supply....12
  Network connection....12
  Power sensor connection....13
  Configuring Wireless LAN....13
Advanced functions....15
  Additional SCPI Commands....15
  Local display....19
Programming hints and snippets....22
Notes....25
FAQs....26

Preface

Dear customer,

congratulations on the purchase of the intelligent interface box RemoSens. You have chosen an established and reliable product to connect your Rohde & Schwarz USB power sensor to a network.

This manual informs you about the use cases and functions of this interface converter.

See https://www.estesys.de for latest informations

Disclaimer of liability

This converter was developed with great care. After extensive tests, this product works together with the USB power sensors of the NRP-Zxx series and others in daily practice. However, errors can never be completely excluded. We can therefore not guarantee the correct processing of the measurement and control data under all combinations of the available SCPI commands. In particular, we do not make any commitments that the converter is suitable for exactly the purpose you have planned. If you as a user encounter an incompatibility, we will take care of a correction as soon as possible.

The manufacturer cannot be held liable for any direct or indirect damage - in particular damage to other software, damage to hardware, damage due to loss of use and damage caused by the converter's inoperability.

If, contrary to expectations, you encounter problems, please report them by email to support@ifd.de. The more details you provide, the better and easier is it for us to reproduce a possible problem. We will analyze such cases as soon as possible, and if there is a bug in the product, we will fix it with an update.

Introduction

General

RemoSens in an interface between a network and certain USB devices. This tiny interface box provides an easy way to connect a Rohde & Schwarz R&S NRP-Zxx series or NRPxxX USB power sensor[1] [2] to your controlling personal computer (host PC) via a network. The interface box is connected to your network either wireless (WiFi/WLAN) or by a wired Ethernet cable. To send and receive data to/from the interface box by the controlling PC, the standard VISA HiSLIP protocol is used. Thus, RemoSens turns your existing USB-type power sensor into a network power sensor... -- ...which can even be controlled wireless.

1 Product and company names listed are trademarks or trade names of their respective companies.

2 R&S, NRP and Rohde&Schwarz are registered trademarks of Rohde & Schwarz GmbH & Co. KG, Munich

The R&S NRP-Zxx series power sensors use an asynchronous, stateless, proprietary protocol to communicate with the outside world. This protocol is called NRP-Legacy protocol. For communicating with this type of sensor, it is therefore always necessary to use a special driver on the controlling host. Sometimes it can become difficult to get an appropriate driver for the operating system of the local computer. – The newer NRPxxX USB power sensors are capable of communicating with a standard VISA protocol, but these types of sensor also still provide the traditional, asynchronous NRP-Legacy protocol.

Since NRP-Legacy is the common protocol for both mentioned sensor series, RemoSens exclusively uses this protocol to communicate with any connected and supported sensor. This is here mentioned for informational purposes only; a user does not come in contact with this proprietary data content and does not have to know anything about this protocol. RemoSens completely hides these internals.

Intended use

The operation purpose of RemoSens can be charaterized by three main areas:

- Avoid distance limitation
- Avoid proprietary drivers
- Utilize standard SCPI commands

Avoid distance limitation

Connecting a USB power sensor to the network removes the distance limitation which is inherent in the USB infrastructure. RemoSens provides transparent access to a power sensor located anywhere on your network. You do not even need a wired network connection to the usage site of the sensor, because RemoSens can be configured to connect to your wireless LAN/WiFi access point. See section **Configuring Wireless LAN** in chapter **Placing into Operation** for detailed information.

Avoid proprietary drivers

Another beneficial feature of RemoSens is that it communicates with the

controlling host via a standard protocol used in today‘s measurement systems. Thereby no special proprietary driver packages are necessary on the controlling host. An industry standard VISA is sufficient. RemoSens automatically adds an adequate status systems to the (otherwise stateless) NRP-Legacy protocol used by the NRP-Zxx power sensor series. Besides this, RemoSens provides a basic error reporting system which can be retrieved by the standard and well-known VISA SYSTem:ERRor? query command.

Utilize standard SCPI commands

Via the RemoSens interface virtually all SCPI commands which are available in the NRP-Legacy protocol, can be utilized. These commands are usually listed and well documented in the user manuals of the power sensors. The commands usually can be used transparently and unmodified. RemoSens „knows“ how to translate between the synchronous nature of the VISA world and the asynchonous environment of the NRP-Legacy environment.

In addition, RemoSens adds a few status and synchronization capabilities which are missing in the NRP-Legacy protocol of the NRP-Zxx power sensors. Although the more modern NRPxxX sensor series provide an extended SCPI status system, these extensions are not used by RemoSens, because there would not be a similar or resonable emulation in the NRP-Legacy world. However the (emulated) status system of RemoSens is sufficiently extensive to support all the necessary synchronization processes in the common use of the power sensors.

Placing into Operation

Preliminary Note

RemoSens is based on an ordinary Linux operating system (at the time of this writing Ubuntu 20.04/64-bit). After power-up, RemoSens usually contacts a DHCP server on your local network to obtain a TCP/IP address.

By default, the device is configured to accept ‚ssh' connections via its LAN interface for configuration purposes. Since the device is shipped with a default username and password, it is important that you select an own secret password for the device in order to not induce any security vulnerability on your network.

The device is shipped with the following settings:

User Name: ubuntu

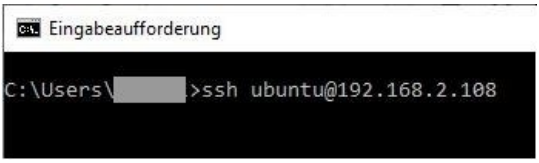
Default Password: ubuntu

Please peek into the following sections **Power Supply** and **Network Connection** to physically bring the device into operation. Then use your favorite desktop PC (which should run Windows 10 or a popular Linux OS to execute the configuration steps described subsequently) to connect to your RemoSens device and change the default password to a secure alternative of your choice. Do not forget the chosen individual password.

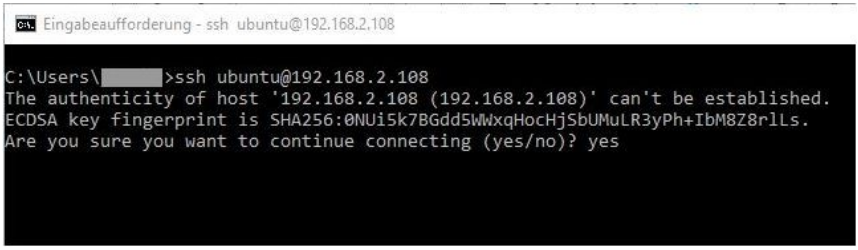
‚ssh' is a command-line utility which comes with the operating system (Windows 10, Linux). To open a command-line window under Windows 10, press the <Windows> key together with R, enter cmd and press the <Enter> key. The command window should appear.

Now login as user ‚ubuntu' to the RemoSens operating system. For this you need to know the IP address, which is shown on the RemoSens display (192.168.2.108 in this example):

```
ssh ubuntu@192.168.2.108
```



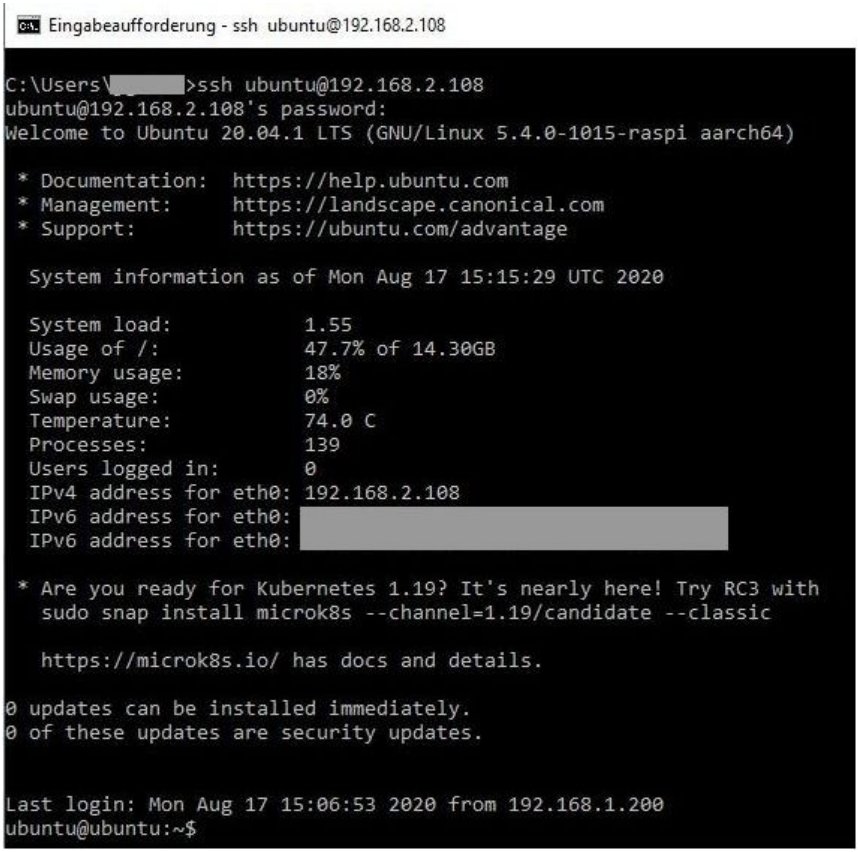
At the very first login attempt you are asked to accept the fingerprint of the new system as being valid by entering `yes` and `<Enter>`



When asked for the password, enter the ubuntu user‘s (default) password, as mentioned at the outset of this section.

You should then be logged in and reach the command prompt.

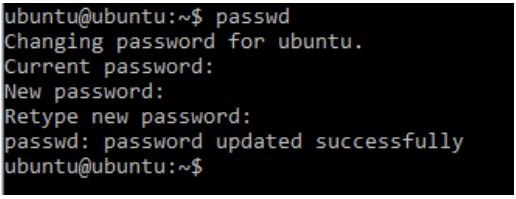
```
ubuntu@ubuntu:~$
```



From the command prompt you will change the default password to an own secure password. Just enter the command

```
passwd
```

You will be asked for the current password and then for a new password, which needs also to be repeated.



Hint: Trivial and/or too short passwords will not be accepted.

After changing the password you can leave the RemoSens operating system by entering the command

```
exit
```

which closes the `ssh` connection to the device.

Power Supply

RemoSens is powered from a single 5 V supply voltage. The power consumption of the device is up to 15 W. The necessary power normally is provided by a wall power supply. Optionally an external PoE converter can be used to gain independence from local mains. In this case the supply power is drawn directly from the Ethernet network cable, which is also carrying power (PoE, Power over Ethernet).

The supply is fed into the device via a standard USB-C type connector. Thus, either connect the device‘s USB-C plug with the wall power supply from the standard accessory, or with the output of the optional PoE converter.

Before you power-up the device, see also the following sections to connect a ethernet network cable. After switch on the power supply, the RemoSens operating system is booting. This takes 20...25 sec. During this time the display area is all in white color. Then a thin white frame appears, and if the device could establish a network connection, its obtained TCP/IP address is displayed.

Network connection

RemoSens provides an RJ45 plug for connection to a local ethernet network. The device normally configures its network parameters automatically by connecting a DHCP server on the local network. As soon as RemoSens has obtained a TCP/IP address, it will display this information on its internal display. This makes it easy to later connect to the device from any user application software.

RemoSens also has a wireless LAN (WiFi) interface. This is unconfigured by default. See the paragraph ***Configuring Wireless LAN*** to learn how to configure and put into operation the internal wireless LAN interface. As soon as the WiFi interface is configured and operational, its TCP/IP address will be shown on the display when the LAN cables is unplugged.

The wired and wireless communication channel can be used in parallel. However, keep in mind that always the last commands sent via one of the interfaces „wins“. Thus, putting the connected power sensor to a certain mode (for example ‚Continuous Average‘) by commands from one interface, and then to another mode (for example ‚Trace‘) by the other interface, you must not expect that the sensor will act in the first (‚Continous Average‘) mode afterwards.

All communication with the RemoSens interface box is done with the so called VISA HiSLIP protocol. Therefore, you always need a VISA on the PC which will be used to control RemoSens and the connected power sensor. Many vendors provide free VISA packages for various operating systems. We recommend to use Rohde & Schwarz VISA, which can be obtain by this link: https://www.rohde-schwarz.com/de/applikationen/r-s-visa-application-note_56280-148812.html

In VISA you will need a so called „resource string“, identifying the device which is to be communicated with. The appropriate resource string is easily assembled from the network address seen on the RemoSens display. In an application you would always use

`TCPIP::<ip-address-of-remosens>::HISLIP`

with a concrete TCP/IP address of ‚`192.168.2.108`‘, the VISA resource string is then

`TCPIP::192.168.2.108::HISLIP`

Never omit the ‚HISLIP‘ protocol selection part, because RemoSens does only speak the VISA HiSLIP protocol.

Power sensor connection

An NRP-Zxx series power sensor can be connect to one of the four available USB plugs of the interface. Keep in mind that RemoSens only supports one power sensor at a time. Therefore, do not plug in more than one power sensor, because the protocol does not give you commands to select a certain sensor. RemoSens always uses the first power sensor which it will find on the USB port(s). This does not necessarily have to be first sensor which has been connected, though.

Configuring Wireless LAN

RemoSens internally runs an ordinary Linux operating system. Most

configuration of a Linux system is done via configuration files. This also holds true for configuring the wireless LAN interface.

To configure the wireles LAN, login as user ‚ubuntu' to the system as described above in chapter **Preliminary Note**. For example:

```
ssh ubuntu@192.168.2.108
```

At the command prompt, enter

```
sudoedit /etc/netplan/60-wireless.yaml
```

Important: Take care to keep the formatting/line indent unchanged, otherwise your network will probably entirely cease working. **Never use <Tab> for indentation.**

This opens the corresponding configuration file where you can define the credentials, which are necessary for letting RemoSens connect to your WiFi access point. Change `MY-SSID-1` and `MY-PASSWORD-1` to the locally valid information. If you alternately also connect to another WiFi access point, you can also define the credentials for a second access point by changing `MY-SSID-2` and `MY-PASSWORD-2`.

```
network:
    wifis:
        wlan0:
            optional: true
            access-points:
                "MY-SSID-1":
                    password: "MY-PASSWORD-1"
                "MY-SSID-2":
                    password: "MY-PASSWORD-2"
            dhcp4: true
    version: 2
```

Finally save your changes by pressing

```
<Ctrl-X> → Y → <Enter>
```

Advanced functions

The traditional NRP-Zxx USB power sensors use a stateless communication protocol called NRP-Legacy. Therefore, for programming these power sensors, typically proprietary drivers are necessary, which normally provide only a function based API (Application Programming Interface). Most other measurement and control devices can be programmed in a well defined and widely used command language, called SCPI (Standard Commands for Programmable Instruments).

RemoSens not only adds the missing commands to the traditional sensor communication protocol but also provides the physical interface to connect the USB power sensors to a network (wired ethernet or wireless). That way the power sensor can be integrated in a network environment which makes use of the VISA architecture. Consequently RemoSens avoids the limitations of USB cable length (< 5 m) and even provides the capability to interact with the power sensor without any physical connection to the controlling host PC by just communicating wireless „over-the-air“ (OTA)

The following chapter describes SCPI commands which will not be found in the user manuals of the (NRP-Legacy) USB power sensors. These commands are interpreted directly in the RemoSens interface and synchronized with the asynchronous communication behavior of the NRP-Legacy sensor.

Additional SCPI Commands

In general, SCPI commands provide a short format and a long format. Usually both formats are shown in user documentations. The formats are differentiated by using upper und lower case letters. The short form is used, when selecting only the uppercase letters of the corresponding desciption, the long format uses all letters. For example, a query for errors is made by the SYSTem:ERRor? query. The four valid command versions are shown here:

SYSTEM:ERROR?
SYST:ERR?
SYSTEM:ERR?
SYST:ERROR?

The upper/lower case representation is only chosen for defining the short and long formats of a command. The actual commands are not case sensitive. It is only relevant to either use the short or the long format of a command component. Hence, SYSTEM:ERROR? and system:error? are fully equivalent. Both commands, in this case, use the long format of all command component. A mixture is allowed, however each component must either be the short or the long form; syst:ERRO? is a syntax error, because the ‚ERRO' component is neither the short nor the long form.

RemoSens provides the following additional SCPI commands, which are not available from the power sensor itself. Commands can be setting commands and/or query commands. A setting command sets something or selects a certain state or setting, while query commands generally return something.

*CLS

Setting only – Clears the local (internal) status/error queue. Thereby erasing any information on possibly occured errors so far. A directly subsequent SYSTem:ERRor? query would always return `0,"No error"` in this case

FETCh?

Query only – Returns the last measurement result. This can be a scalar value (in Power Average measurement mode) or a comma separated result list (when the sensor is operated in Trace mode, for example). Please keep in mind, that most of the traditional NRP-Zxx power sensors provide a feature to activate „auxiliary" values together with the power-average result. Usually the MIN/MAX values within the measurement period are provided as additional auxiliary values. Therefore, the result, which is retrieved by a FETCh? Command, usually has the form of a comma separated data triplet

```
power-avg-value, min-value, max-value
```

for example (if the auxiliary values are not activate):

```
1.045823e-10,0.000000e+00,0.000000e+00
```

and with auxiliary values:

```
1.045823e-10,2.834974e-11,1.685922e-10
```

*OPC[?]
Setting or query – This is a command to synchonize on the execution of a (possibly long lasting) measurement. The query form of the command always returns ‚1‘.

The settings form (*OPC) can be used to hold the processing of further SCPI commands when waiting for a result of a previously started measurement. A typical sequence would be

```
INIT:IMM;*OPC;FETCH?
```

to initialize and return a new measurement result.

*STB?
Query only – Queries the Status Byte. Since the NRP-Legacy power sensors do not provide a Status Byte, RemoSens emulates this information by observing the current execution state of the power sensor. The following bits are defined:

Bit	Dec	Hex	Description
7	128	0x80	Set, if sensor is not Idle, i. e. sensor is acitively measuring or in Wait-for-Trigger state
6	64	0x40	unused
5	32	0x20	unused
4	16	0x10	Set, if data is available in the output buffer. This can be measurement result, error or status information
3	8	0x08	unused
2	4	0x04	Set, if an error has occurred
1	2	0x02	unused
0	1	0x01	unused

STATus:OPERation:CONDition?

Query only – This commands returns an (internally emulated) operation condition status register. From the 8 bits of this register only the highest bit is defined.

Bit	Dec	Hex	Description
7	128	0x80	Set, if sensor is not Idle, i. e. sensor is acitively measuring or in Wait-for-Trigger state
6	64	0x40	unused
5	32	0x20	unused
4	16	0x10	unused
3	8	0x08	unused
2	4	0x04	unused
1	2	0x02	unused
0	1	0x01	Calibration is in progress

STATus:OPERation:MEASurement?

Query only – This commands returns an (internally emulated) measurement condition status register. From the 8 bits of this register only Bit 1 is defined.

Bit	Dec	Hex	Description
7	128	0x80	unused
6	64	0x40	unused
5	32	0x20	unused
4	16	0x10	unused
3	8	0x08	unused
2	4	0x04	unused
1	2	0x02	Set, if sensor is acitively measuring or in Wait-for-Trigger state
0	1	0x01	unused

SYSTem:ERRor?
Query only – RemoSens manages an internal error queue, which stores possibly occurred error during command processing. The error queue will be read out, one entry after the other, by SYSTem:ERRor? queries.

SYSTem:SERRor?
Query only – This query has no function. It is just implemented in RemoSens to make existing application programs happy which possibly make use of static error queries. The static error query in RemoSens always returns `0,"No Error"`.

UNIT:POWer[?]
Setting or query – A traditional NRP-Zxx power sensor always returns the measured power value in ‚Watt'. Often users prefer power values in logarithmic units ‚dBm'. To select the prefered result unit, send

UNIT:POWer DBM

to select the dBm representation, or send

UNIT:POWer W

to select the Watt representation.

Local display

RemoSens provides a small TFT display which lists the commands received from the controlling host and the measurement and status results returned to the host. Normally the input/output information is in monochrome (white on black) style. An exception constitutes the command which queries for errors (SYSTem:ERRor?). If there is no error, normal white text is used; however, when an error occurred, it is shown in red color. Thus making it quickly observable that „something" went wrong.

Besides the normal information input/output listing the RemoSens display can be switched into a local measurement result display mode, where Continuous Average values are prominently shown in (dBm units) to easily follow the trend of the measured power when situated locally.

The corresponding commands are

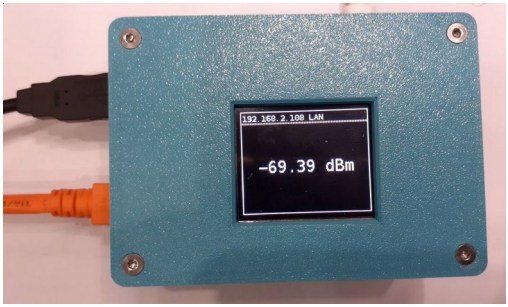
DISPlay:MODE LIST

Setting only – The RemoSens display will show input/output information in the form of sequential lines of text

This can be considered as kind of an activity indication, as well as an error indication – whenever the answer to a SYST:ERR? query is not `0,"No Error"` the output will change to red colored text.

DISPlay:MODE MEAS

Setting only – The RemoSens display will show measurement results of Continuous Average measurements in a bigger typeface in dBm units for easy observation by a local user.



In this mode, it can make sense to operate the connected power sensor in its „continuous“ measurement mode (INIT:CONT ON). This mode is normally not used when doing remote control. In remote control you would typically execute selective measurements started by INIT:IMM and synchronized to the end of measurement by polling the status register or on *OPC (operation complete). The DISPlay:MODE MEAS however, legitimates the continuous measurement mode, because you will probably want to see the trend of the RF power value.

See chapter **Programming hints and snippets**, item d), for an example on how to select the continuous measurement and display mode.

Programming hints and snippets

Here are some ‚pseudo-code' snippets showing the implementation of certain measurement tasks.

a) Power average measurement

Average count = 4; aperture time 1 ms; results are desired in ‚dBm'

```
write('*RST')
query('*IDN?')
write('SENS:FUNC "POW:AVG"')
write('SENS:AVER:COUN:AUTO OFF')
write('SENS:AVER:COUN 4')
write('SENS:APER 1.0e-3')
write('TRIG:SOUR IMM')
write('UNIT:POW DBM')
query('INIT:IMM;*OPC;FETCH?')
```

In this scenario the measurement is unsynchronized. That means, we do not know at which time the result will be ready. It will take as long as it takes. In case of actual short VISA timeout settings this could lead to a timeout condition, depending on the selection of average count and aperture time and/or trigger source.

b) Power average measurement (synchronized)

Preferably synchronized measurements should be executed. This mean, we only query for the measurement result by utilizing a `FETCH?` command, when we know that there is a result. In other words: when we know that the sensor is ready (= idle) again – For example, auto-average measurements may take a long time…

```
write('*RST')
query('*IDN?')
write('SENS:FUNC "POW:AVG"')
write('SENS:AVER:COUN:AUTO ON')
write('SENS:APER 10.0e-3')
```

```
write('TRIG:SOUR IMM')
write('UNIT:POW DBM')
write('INIT:IMM')

int iStat = 0
do
{
  iStat = (int)query('*STB?')
}
while ((iStat & 0x80) == 0x80)

query('FETCH?')
```

In this scenario the measurement is synchronized. That means, by polling the status byte we will recognize when the sensor is not active anymore (= IDLE again). Then we can fetch the measurement result without running the risk of getting a VISA timeout due to still not available measurement results.

c) Zero calibration

After a warmup phase or whenever the ambient temperature has significantly changed, it is recommended to execute a zero calibration with the power sensor. Zero calibration is a long lasting function which can take 7...10 sec. Therefore you should preferably poll for the end of calibration before you continue sending other commands. A pseudo sequence would look like this:

```
write('CAL:ZERO:AUTO ONCE')

int iCal = 0
do
{
  iCal = (int)query('STAT:OPER:COND?')
  iCal = iCal & 0x01
  if (iCal != 0)
    sleep(0.3)    # delay for 300ms
}
while (iCal != 0x00)
```

The above code start the zero calibration and then polls on the corresponding bit in the operation status register to recognize the end of calibration.

d) Continuous measurement w/ local display

Average count = 4; aperture time 1 ms. The local measurement display mode always selects the results in ‚dBm‘

```
write('*RST')
query('*IDN?')
write('SENS:FUNC "POW:AVG"')
write('SENS:AVER:COUN:AUTO OFF')
write('SENS:AVER:COUN 4')
write('SENS:APER 1.0e-3')
write('TRIG:SOUR IMM')
write('DISP:MODE MEAS')
write('INIT:CONT ON')
::
::
::
:: <at some later time...>
::
write('INIT:CONT OFF')
write('DISP:MODE LIST')
::
```

In this scenario RemoSens is used to continuously monitor the RF input power visually on the local display.

Notes

If you are interested in general information about the Linux operating system which works in the RemoSens interface box, see

https://ubuntu.com/

The RemoSens application uses parts of the Boost library to implement its network I/O functionality. The Boost library comes with the following license terms:

See https://www.boost.org/users/license.html for additional information.

FAQs