

AVS47-Serial/USB-F CONVERTER

For Interfacing the AVS-47B with Computers
via Fibre Optic Link

User Guide



 **PICOWATT**
Veromiehentie 14
FI-01510 VANTAA, Finland
phone 358 50 3375192
Internet: www.picowatt.fi
e-mail: reijo.voutilainen@picowatt.fi



CONTENTS

WARRANTY	3
BACKGROUND	4
USB-Picobus	4
GPIB-Picobus	4
AVS47-Serial/USB-F CONVERTER	5
CONNECTING THE AVS47-Serial/USB-F	5
From AVS-47B to Box1:	5
From Box1 to Box2:	5
From Box2 to Computer:	6
Power to Box2:	6
STARTING THE AVS47-Serial/USB-F	6
Resetting the AVS47-Serial/USB-F	6
RS232 Format	6
COMMANDS AND COMMAND LINES	8
Commands	8
Responses	8
Send/Receive Serial tool	10
INITIAL COMMANDS AND QUERIES	11
HARDWARE COMMANDS	11
MEASUREMENT AND READOUT	
COMMANDS/QUERIES	12
OTHER COMMANDS AND QUERIES	16
COMMANDS FOR THE TS-530A	
TEMPERATURE CONTROLLER	18
CABLE SPECIFICATIONS	19
Picobus Cable	19
Serial Cable	19
RE-PROGRAMMING THE AVS47-Serial/USB-F	19
AVS47-Serial/USB-F TROUBLE	
SHOOTING IDEAS	20
DECLARATION OF CONFORMITY	23
INDEX	24
REVISION HISTORY	26



WARRANTY

Picowatt warrants the **AVS47-Serial/USB-F hardware** to be free from defects in materials and workmanship. Our liability under this warranty is limited to repairing or replacing any instrument or part thereof which, within three (3) years after the shipment to the original purchaser, proves defective. This warranty is void if the instrument has not been used according to the instruction manual, or if it has been used under exceptional environmental conditions.

In need of warranty repair, the instrument must be returned to **Picowatt**, prepaid, and with a detailed description of the fault or malfunction following the instrument.

The name, address and e-mail address of a person who is able to give supplementary information should be included whenever possible. If the repair was covered by warranty, **Picowatt** will return the instrument on our cost using an economical shipping method.

If no fault is found, or if there is a strong indication that the warranty is void, the purchaser is charged for the return freight and costs in addition to the repair. It is recommended that **Picowatt** be contacted prior to shipment. We can possibly give instructions for additional tests or simple component replacements so that unnecessary shipments may be avoided.

The **firmware must not be considered a commercial product**. It is given as is, for free, without any kind of warranties or liability. The program and this user guide may contain errors, and we would be glad to get feedback, corrections and suggestions for improvements.

Important: The AVS-47B uses +/- 5 Volt levels for data communications in its standard configuration, whereas the AVS47-Serial/USB-F can be damaged by applying negative voltages to its 15-pin connector.

Therefore, short circuit piece **JP204 MUST BE CHANGED** to position JP203 on circuit board "E" (the board with the power supply unit) before connecting the AVS-47B and the converter together. In case of any uncertainty, please contact factory.

If the AVS-47B is interfaced with model AVS47-IB GPIB box, or directly with a PC computer using its Com port or via a USB-232 adapter, this jumper shall be returned to JP204.

RV-Elektroniikka Oy Picowatt
Veromiehentie 14
FI-01510 VANTAA
FINLAND
telephone +358 50 337 5192
email: reijo.voutilainen@picowatt.fi



BACKGROUND

The AVS47-Serial/USB-F is an external protocol converter that creates an RS232 interface with optical fibre link for the model AVS-47B AC Resistance Bridge. Until now, **Picowatt** have offered two possibilities for interfacing the AVS-47B with computers: The direct **Picobus** interface and the model **AVS47-IB** external GPIB (IEEE-488) interface unit. Both alternatives have limitations with respect to computer type, operating system and programming language. The AVS47-Serial/USB-F removes many of these limitations, offering ultimate safety against all kinds of electromagnetic interference that could heat the resistive sensor.

A short comparison of the three available interfacing solutions follows. Fibre optic links are available also for the two first solutions.

Some commands are for the TS-530A Temperature Controller. This product has been discontinued, and those features are only for the already existing instruments.

USB-Picobus

Picobus is a proprietary synchronous, serial protocol that is based on two coming and two leaving signal lines. Suitable four lines are provided by the hardware handshake outputs and inputs of traditional Com: ports of PC-type computers. The asynchronous TXD and RXD signals of the RS232 interface are *not* used by Picobus. Today's computers seldom have physical Com: ports, but a virtual Com: port can be created by a **USB-232 converter**.

Unfortunately, low-level Picobus communication is complicated, as it requires computer program to write and read states of independent bits of some hardware registers. In order to make programs portable between different platforms, operating systems do not favour direct hardware access. For this reason, we offer driving programs (**USB-Picobus**), but only for a **Windows-PC** running **LabView2012** or newer (base version or better).

This has been a serious limitation that excludes Mac computers and programming languages other than LabView. The great advantages of Picobus are, that the protocol is almost bullet-proof, it has low EMI noise, and for customers with a suitable com-



The AVS47-IB is a protocol converter between GPIB (IEEE-488) and Picobus. It is the heart of the "GPIB-Picobus" interface. Only limited availability.

puter environment, it does not cost anything, with the exception of a possibly needed USB-232 converter. For the most noise-critical applications, an optical fibre link to the bridge is available (**AVS47-PICOLINK**). It replaces the conventional wire cable between AVS-47B and the PC computer.

GPIB-Picobus (**OBSOLETE!**)

This interfacing solution is based on an external converter, AVS47-IB, between IEEE-488 and Picobus protocols.

This very powerful converter offers automatic scanning of sensors, buffering of data and many macro commands. The box is connected to - but galvanically isolated from - the AVS-47B via Picobus and to the computer via GPIB (therefore the name of this option is **GPIB-Picobus**). It has its own mains power supply and it can be located far from the cryostat in order to minimize electromagnetic or ground current problems that the GPIB line may cause. The box can be used with computers having an GPIB controller and suitable software for GPIB communications. It is highly compatible with the IEEE-488.2 standard with its mnemonic and common commands and error reporting. We offer a versatile free **LabView Driver** that was written for LV7.1 and can still be used with today's LabView versions. For the most noise-critical applications, an optical fibre link to the bridge is available (**AVS47IB-PICOLINK**). It replaces the conventional wire cable between AVS-47B and AVS47-IB.

The GPIB-Picobus has a much wider range of applications than USB-Picobus, e.g. it can be interfaced also with Mac and Linux computers. However, for customers that do not already use GPIB, the cost of this alternative is significant.



AVS47-Serial/USB-F CONVERTER

The AVS47-Serial/USB-F is an external protocol converter for the AVS-47B AC Resistance Bridge. It changes the proprietary low-level low-noise synchronous primary interface (called “Picobus”) of the bridge to comply with the common asynchronous RS232 hardware standard. With this converter, interfacing is possible with any computer type, operating system and programming language as long as they support RS232 communication in its simplest form: 9600, 8, N, 1 without handshaking. By comparison, the primary interface without a converter is supported only for a Windows-PC running LabView.

The AVS47-Serial/USB-F differs from the AVS47-Serial/USB-W by providing an **optical fibre link** between the bridge and the remote computer. This all-plastic cable is 100% sure not to bring high frequencies into a shielded space from outside, providing the best available safety against all kinds of interference. From the software point of view these two models are exactly similar.

The converter consists of two boxes. Box 1 is located in the experimental area, near to the bridge, while box 2 should stay outside the shielded room. Even if the cryostat is not in a Faraday cage, a long physical distance - 5m or optionally 10m - can still be valuable.

Box 2 is usually connected to a **USB-232 adapter** (not included - it must be supplied by customer for ensuring compatibility).

The USB-232 adapter creates a virtual RS232 port, which the computer’s application program can access via the computer’s USB port. If the computer has a physical RS232 port, no USB-232 adapter is needed.

This device is based on the very popular Arduino Mega2560 circuit board.

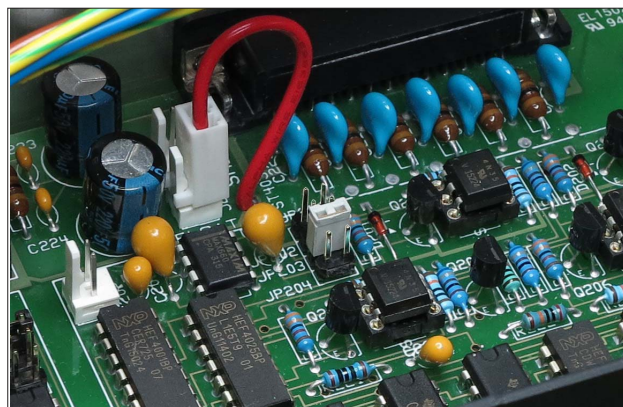


Both AVS47-Serial/USB-F units measure 130x105x60 mm. Box1 is powered by the resistance bridge and Box2 gets power from a +12V wall adapter.

CONNECTING THE AVS47-Serial/USB-F

Box1 of the AVS47-Serial/USB-F uses 0/+5V voltage levels for communications with the AVS-47B Resistance Bridge, which is set for -5/+5V levels by default. **Therefore, open the top cover of the bridge and move short circuit piece JP204 to position JP203 before making any connections (“E” board).**

The jumper must be in position JP204 if AVS-47B is interfaced with an AVS47-IB or directly with a PC computer, or with a USB-232 adapter.



Change short circuit piece JP204 to position JP203 before making any connections in order to avoid negative voltages from causing damage to the AVS47-Serial/USB-F, which uses 0/+5V voltage levels for communications with the AVS-47B resistance bridge.

From AVS-47B to Box1:

Box1 of the AVS47-Serial/USB-F is connected to the AVS-47B Resistance Bridge using the supplied 1.5-meter long male DB25P / male DA15P cable. This box should be located near to the resistance bridge and far from the computer. It is powered by the bridge.

From Box1 to Box2:

The 5- or 10-meter all-plastic optical cable consists of four fibres, which have been marked with colors black, red, yellow and blue. The optical transmitters (white) and receivers (black) are marked with numbers 1..4. Connect the fibres to corresponding numbers of boxes 1 and 2. Recommended use of colors: 1=black, 2=red, 3=yellow and 4=blue.



Push the ends of the fibres into the receivers/transmitters as deep as they go and tighten the screws. Be extremely careful not to tighten the screws too much: they are quite weak and it is more convenient to repeat insertion and tightening than to replace the opto board because of a broken component. **USE ONLY FINGERS!**

The fibre ends can be renewed by cutting a millimeter or two using sharp end cutters, if they seem not to be in good condition.

Do not bend the cable - bending radius must remain below 20mm.

From Box2 to Computer:

Case A: Computer has a physical RS232 port

Connect the supplied 1.5m cable with male and female 9-pin D-connectors from Box2 to the RS232 port of your computer.

Case B: Computer has only USB ports

You need a USB-232 converter plugged into your computer's USB port and its software installed. Plug the supplied DE9P/DE9S cable between the converter and the AVS47-Serial/USB box. The USB-232 converter from National Instruments (NI part number 778472-01) is known to work well, but it is expensive. Cheaper converters are available from other manufacturers. Check the compatibility with your platform before buying one.

Power to Box2:

Check that the input voltage of the supplied 12V DC wall transformer corresponds to your local mains voltage and that the plug is of correct type. Most of our supplied wall transformers are specified for 100-240 Volts but their plugs differ and it is possible that the plug type does not match your mains socket. If you have to obtain a new wall transformer:

- it must deliver regulated +12 Volts
- the plug must be center positive 2.1mm * 5.5mm * 12mm.

STARTING THE AVS47-Serial/USB-F

The firmware in Box2 starts when it gets power from the +12V DC wall transformer regardless of whether the cables are plugged in or not. Box1 has no firmware and it starts when

- The 1.5m cable to AVS-47B is in place, and
- The AVS-47B is on

The REMOTE indicator on the AVS-47B front panel blinks weakly and quickly when Box2 boots leaving the bridge in local mode. Booting or a change from local to remote or vice versa does not alter the state of the bridge.

Resetting the AVS47-Serial/USB-F

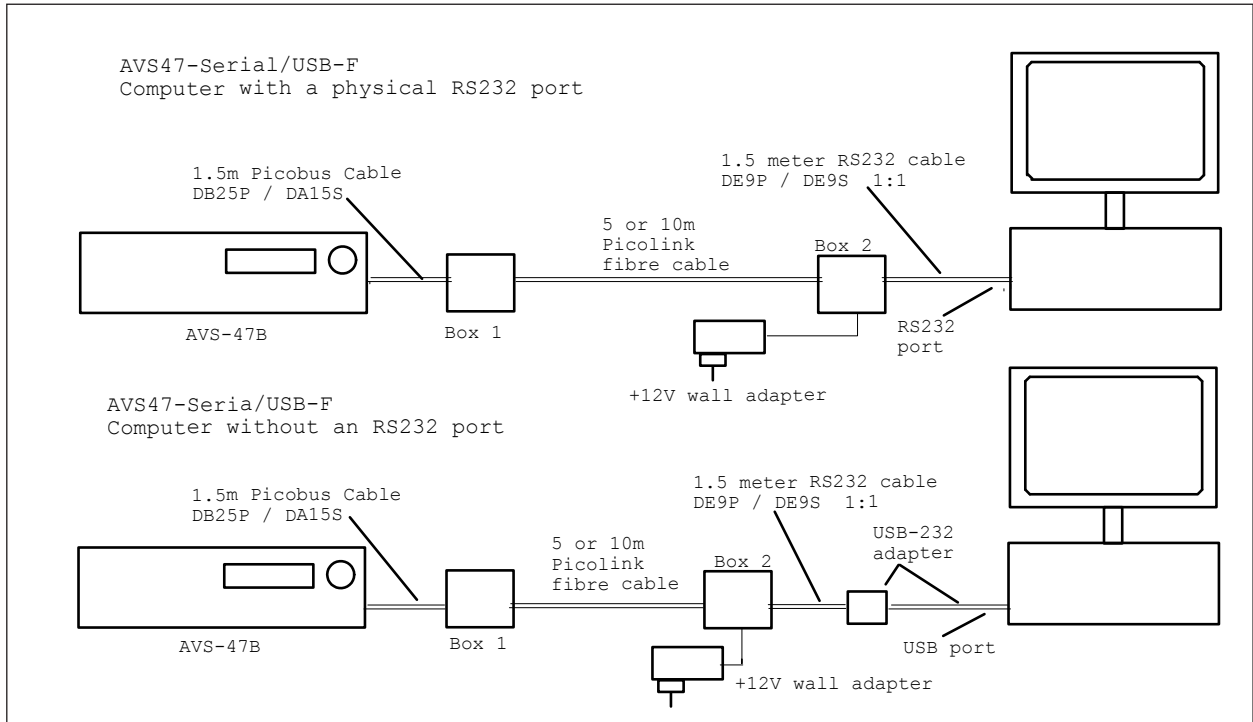
If needed, the Box2 can be re-initialized by

- unplugging and then re-connecting the +12V DC power plug. The operation does not change the states of the bridge or the TS-530A controller, except that the AVS-47B will change into manual mode. However, you must re-program the settings of the controller with the last used values, which must be in computer's memory for this purpose, as they cannot be read from the controller. Re-programming the TS-530A with its current settings does not affect its analog operation in any way.

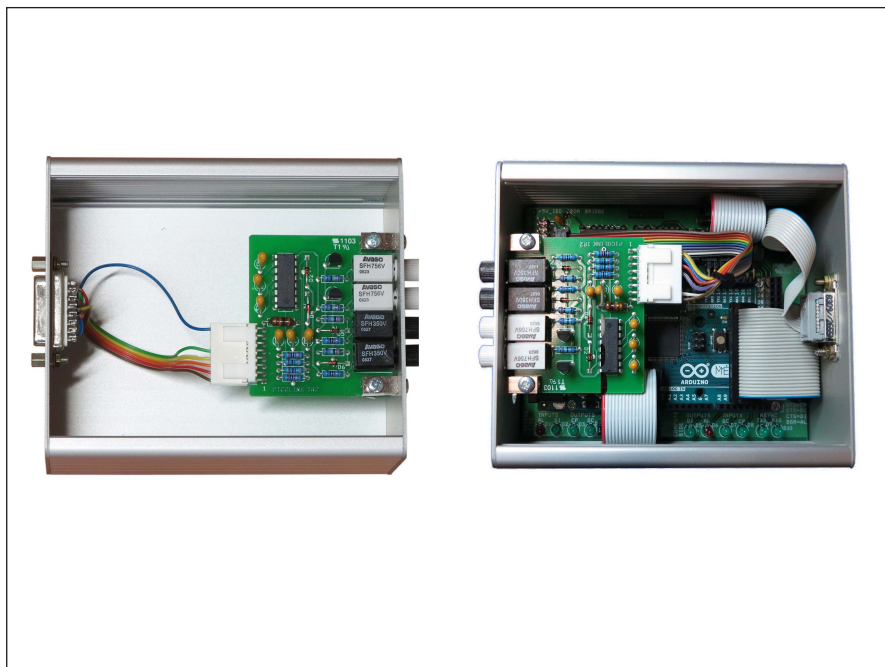
Box2 can also be reset by command "RST". It will bring the bridge to a safe setup: input=ZERO, multiplexer channel=0, range=2M Ω , excitation=3 μ V and display=R. In addition, command separator defaults to ";" (semicolon) and response line terminator to CRLF. Software reset is a less powerful way to initialize the program than the power-off-on method and it cannot be used if the CPU has hanged for some reason.

RS232 Format

The AVS47-Serial/USB-F uses the most common RS232 format: baud rate 9600, 8 data bits, no parity, 1 stop bit and no flow control. This format can be changed only by changing the Arduino firmware and re-loading it. Do not change this format, if there is no compelling reason to do so.



There are two ways to connect the AVS47-Serial/USB-F: Either directly to computer's RS232 port, or via a USB-232 converter to computers USB port. The converter creates a virtual RS232 port that your high-level program will access.



The AVS47-Serial/USB-F consists of two boxes connected by the fibre optic cable. Box1 (left) is connected to the AVS-47B and contains the opto card. In addition to the opto card, Box2 contains also a mother board and an Arduino Mega2560 processor unit which contains the firmware of this protocol converter.



COMMANDS AND COMMAND LINES

It is a good idea to get acquainted with the AVS47-Serial/USB-F by using an RS232 hyperterminal program. It lets you control the bridge by writing commands/queries and reading the responses. Although such a program is no longer included in Windows, many free programs are available from the Internet. For example, we have used

<https://sourceforge.net/p/hypeterminal/wiki/Home/>

and, if you have LabView, you can try

<https://decibel.ni.com/content/docs/DOC-16284>

See also page 10.

Commands

Commands to the AVS47-Serial/USB-F are not case sensitive. You may insert blank space(s) between the command and argument part. Commands “ran5”, “RAN5”, “ran 5”, “RAN 5” or “ran 5” are all equivalent.

The first part of a command or query can contain only alphabetic letters. The second, argument part of a *command*, is made of integral numbers. The argument part of a *query* is a question mark “?” like in “RAN ?” or “ran?”.

Several commands can be placed on a single command line. The commands/queries must have a command separator, or delimiter, between them (comma or semicolon, which is the start-up default). Your own computer program can terminate the command line either by carriage return (CR or `\r`, ASCII 13), by linefeed (LF or `\n`, ASCII 10) or by CRLF. These are called line terminators. The commands/queries are performed in sequential order, the previous command must be completed before the next one can be handled.

For example: “rem1;inp1;ran3;exc7” (quotation marks are not parts of the actual string) sets the bridge in remote mode, sensor measuring input, 200Ω range and 10mV excitation. After having waited for some seconds (settling time), one can take the reading.

Maximum number of characters on one line, including command separators and possible blanks, is 60. Handling the commands starts after a line

terminator has been received. If the command line is not terminated with CR, LF or CRLF, processing will not start.

Do not issue further commands or queries before all the commands/queries on the previous command line have been executed.

Responses

The AVS47-Serial/USB-F obeys the strict principle, that only a query can produce response. So your application program needs not poll and read the serial port after commands.

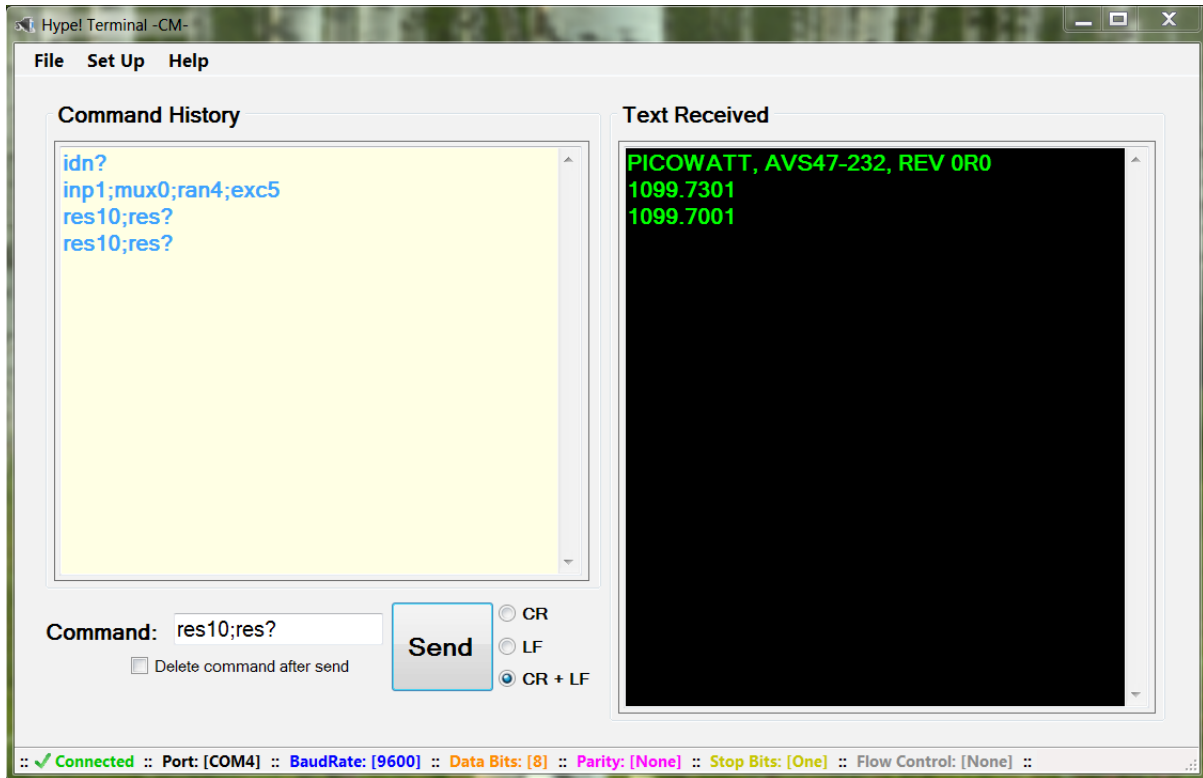
Responses can consist of printable alphanumeric characters, but most queries return only a number. Some values are output as floating point numbers, but exponential format is not supported. The responses do not have headers in order to make them easier to read into a program.

If a command line consists of more than one query, the responses are output in the corresponding order and they are separated by the specified command separator (delimiter).

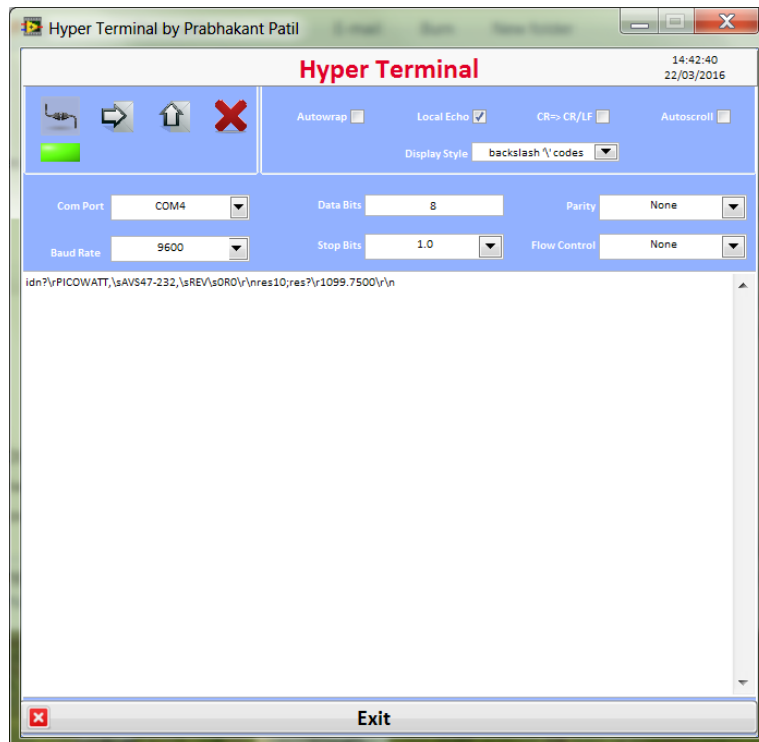
For example, command line “ARN10;RES5;RES?;RAN?” instructs the bridge to go to autorange mode, wait for 10 seconds after each automatic change of range (if needed), then take a mean of 5 A/D conversions, and place the result and the range setting (which was possibly altered by autoranging) in the output queue. The result is sent via the RS232 port to the computer, which must detect that data has arrived into the serial buffer and then read it from the buffer. The response could be like “1234.5000;4” (i.e. 1234.5Ω; 2kΩ range).

You may save programming overhead by giving *commands* for a measurement on one line. Responses to *queries* may be easier to read directly into variables, if queries are made separately for each item. Then one does not need to remove the delimiters and parse the response line. Queries are fast operations as compared with many commands.

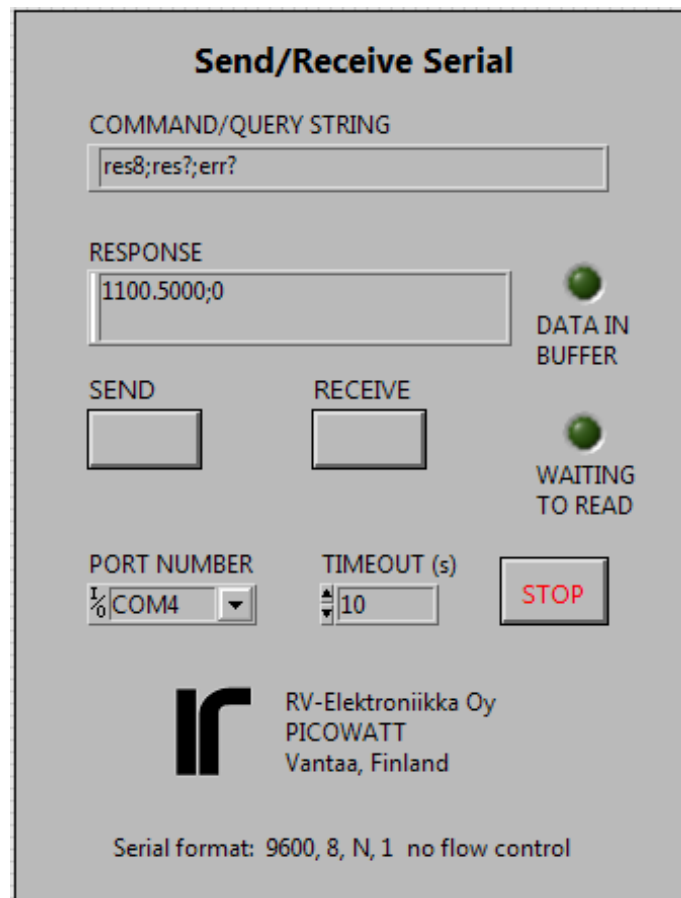
The response ends by the line terminator specified by the TER command (default is CRLF). The terminator can, but it must not, be the same for both transmitting and receiving. The AVS47-Serial/USB-F will always recognize any of the three line terminators, CR, LF or CRLF.



The “Hype! Terminal” (see text for link) is easy to use: Just specify the Com port number (SetUp) and it is ready. After identification, the bridge is set for measure input, channel 0, range 2k Ω , and excitation 300 μ V. Then 10 A/D conversions are taken (RES10) and the output is read by RES?. This RS232 terminal program for a Windows PC does not require LabView.



Users of LabView can also try this program (see text for link). The output includes all printing and non-printing characters in both transmitted and received strings, which can be useful for debugging. But otherwise it is less convenient to use than the previous example.



Send/Receive Serial tool

This handy LabView VI was written for developing programs for the AVS47-Serial/USB-F. It allows one to send commands and queries and to read the response to queries. It requires the base version or better of LabView2012 or later.

Check the jumper position JP203 inside the AVS-47B bridge and connect the cables as was instructed earlier. The CPU Box2 starts when it gets power from the +12V mains adapter. Select the correct Com: port. The serial format is fixed to the most common default, 9600,8,N,1 with no flow control. Run the VI and click “SEND” for sending the default IDN? query. DATA IN BUFFER should light. Click “RECEIVE” for reading the response. If this works, save the VI so that you do not need to re-enter the port number when you load the VI next time.

Use this tool for getting acquainted with the behaviour of the software before starting to write your own application in whatever programming language you prefer. This VI uses the ‘\n’ (newline) char-

acter for terminating the transmitted string, but your program can as well use ‘\r’ (carriage return) or ‘\r\n’.

Define the response line terminator by the command TER. You can see the non-printing characters in the response if you stop the VI, click the RESPONSE field and select ‘\’ Codes Display or Hex Display from the context menu. Start the VI again.

If you try to receive before a response is available, the program waits until the input buffer is non-empty or timeout occurs. See the block diagram, how this was done. You can put the OPC? query after a time-taking long average like RES100;OPC? This “operation complete” query places character ‘1’ in the output queue when averaging is ready. But anyway, your program must poll the input in order to see when you can send the RES? query and read the response.

This VI can be downloaded from our WEB site at www.picowatt.fi/interfacing/computer_interfaces.html (link : “read and write using LabView”).



INITIAL COMMANDS AND QUERIES

All queries work in both remote and local modes, so that you can read bridge settings although it is in local. Commands that control the operation of the AVS47-Serial/USB-F firmware are also effective in both modes, whereas hardware commands to the AVS-47B can be used only in remote. These initial commands are for making the AVS47-Serial/USB-F firmware to correspond to your application program. The commands and responses are same for the -W and -F versions of the converter.

IDN?, *IDN? Identification query. You can check successful starting and firmware version of the AVS47-Serial/USB by this query. The response has four comma-separated fields and is like "PICOWATT, AVS47-Serial/USB,0, REV 1R3". The serial number field is identically "0".

HW? Hardware version query. Returns the version of the AVS47-Serial/USB hardware. The response is like "PICOWATT, RS232PB_A2". The latter item is the version of the mother board.

AL? Alarm line query. This query transacts with the AVS-47B bridge and checks the state of the "AL" Picobus signal line (AL is used for preventing multiple reads of a single A/D conversion). Response should be 1 if the bridge is powered, if the 25/15 pin cable is in place and everything is OK. If the response is 0, check that the cable is plugged and OK. Re-start the box by re-connecting its +12V power. The bridge should end up in local mode. If AL? still returns zero, refer to the trouble-shooting procedure (pp. 20) or contact factory.

LIM [0..1] Select the command delimiter (or command separator).
0 = semicolon (dec ASCII 59). This is the start-up default, which is used also by the IEE-488.2 standard.

1 = comma (dec ASCII 44). May be useful if a comma-separated format (CSV) is preferred.

TER [0..3] Select the response line terminator. Depending on your computer software, you may need to modify the default value of CRLF. When *reading* the serial port, the AVS47-

Serial/USB firmware looks for both CR and LF, and when either of them is encountered, the line is considered as ended. The possibly remaining terminator is neglected. You can modify the response line terminator by sending the TERx command:

0 = nothing

1 = linefeed (LF, \n, dec ASCII 10)

2 = carriage return (CR, \r, dec ASCII 13)

3 = CRLF (start-up default)

REM [0..1 | ?] Remote mode command/query. The change from local to remote does not change the state of the bridge: the program first reads the setup in local mode and then sends this setup to the bridge in remote mode. Any hardware commands sent before the REM1 command are forgotten. The AVS-47B shows remote mode by a yellow light on the front panel.

0 = local

1 = remote

REM? returns 0 or 1

HARDWARE COMMANDS

INP [0..2 | ?] Input selector command/query.
0 = Grounded input (zero resistance)
1 = Measure the selected sensor channel
2 = Calibrate (bridge measures internal 100Ω)
INP? returns 0..2

MUX [0..7 | ?] Multiplexer channel command/query. The bridge will need a settling time after channel has been changed. The required time is longer if excitation is low. Use the DLY or SCK command after setup commands like MUX, RAN and EXC, before starting to take readings.
0..7 = sensor channel
MUX? returns 0..7

RAN [0..7 | ?] Range command/query
0 = no range is connected. No excitation can run into the sensor, and output is random. Avoid using this value in order to avoid accidental heating of the sensor when another range is selected later. If you do not know the proper range, start from 2MΩ.



1..7 = ranges from 2Ω to $2M\Omega$
RAN? returns 0..7

EXC [0..7 | ?] Excitation command/query. Excitation voltage, as the term is used in the context of the AVS-47B, means the RMS voltage across a sensor whose value is half of the selected range. Excitation is symmetrical square wave -shaped current at about 13.7Hz.

0 = no excitation

1..7 = $3\mu\text{V}$, $10\mu\text{V}$, $30\mu\text{V}$... 3mV

EXC? returns 0..7.

REF [0..20000] Reference command for deviation (ΔR) measurements. For example, REF10000 sets the AVS-47B's reference DAC to 1 Volt, which corresponds to the middle of any currently selected resistance range.

The resulting DAC voltage can be measured by switching the ADC input to DIS3 (the internal reference). Then issue ADCx for measuring the output voltage of the DAC.

The programmed reference DAC has only 12 bits (0..4095), whereas the 0..20000 output of the bridge corresponds to about 14 bits. The reference value is therefore divided internally by 5 for scaling it to range 0..4000. The DAC output changes in steps of 5 digits ($500\mu\text{V}$) and has a typical accuracy of a low-cost 12-bit D/A converter.

NULDEV [1..100] Null deviation command.

This is a macro command that measures whatever was previously selected by the DIS command and then sends this value to the reference DAC. The argument determines, how many A/D conversions are used for the measurement. A longer average improves accuracy if readings are noisy.

This command is intended to be used only with DIS0. The REF POT / REF MEM front panel switch must be in REF MEM position. After the NULDEV command, the rear panel DIFFERENCE BNC output is near to zero. Select DIS1 for measuring the difference.

RFS? Reference source query.

The deviation signal V_{dR} is the difference between measured resistance value (analog voltage from the self-balancing circuitry) and

the reference voltage. The reference voltage, in turn, can be either the output voltage from the reference DAC or the voltage of the front panel potentiometer. This selection is made by the front panel REF POT/REF MEM -switch. There is no remote command for changing the switch position.

The reference DAC is programmed remotely by sending the REFx command (see above). It can also be programmed manually by lifting the SET REF switch momentarily. Then the DAC takes the *displayed reading* as input. Deviation can be nulled this way manually. The NULDEV command is for making it remotely.

In remote-controlled applications, the reference-source switch should be in REF MEM position. You can verify this by quering RFS?. The response is:

0 = reference DAC (reference memory)

1 = front panel potentiometer

MAG? Magnifier query

The deviation voltage V_{dR} can be amplified by a factor of 10. Amplification is made by a simple circuit and is therefore not very accurate. The $\Delta R \times 10$ mode is suitable for recording only small changes, not for measuring absolute values. It is best when excitation is high and readings therefore less noisy. Magnification can be selected only manually. MAG? returns

0 = $1 \times \Delta R$

1 = $10 \times \Delta R$

MEASUREMENT AND READOUT COMMANDS/QUERIES

These commands are for determining what the A/D converter measures, for making single or averaged A/D conversions, and for reading the result. There is also a command for detecting possible ADC over-range.

DIS [0..7 | ?] Display selector command

This command selects one of 8 possible voltages to be measured. The current selection can be queried by DIS?.

Use the RES command and query only when



displaying DIS0 and DIS1. The RES? query can also be used for reading item 7 (Set point of the old TS-530A temperature controller). Then you get the set point in resistance, scaled correctly by the currently selected measuring range. However, because of the very old design, the TS-530A set point cannot be given in resistance but it must be given as an integer. Refer to the “SPT” command below.

Use the ADC command and query for all other display items than 0,1 and 7 because the RES values are scaled by the currently selected measuring range. The ADC? query returns integers -19999..19999 corresponding to its input voltages from -2V to +2V. Refer to the AVS-47B manual for how to use the various display items.

- 0= Voltage proportional to the sensor value R
- 1= Deviation ΔR between R and the reference
- 2= Adjust reference. This is the voltage from the front panel potentiometer
- 3= Reference. This is output voltage from the reference D/A converter
- 4= Excitation voltage. This is the approximate excitation voltage across the sensor. Useful only on the lowest resistance ranges and high excitation. Can be used for estimating current lead resistance.
- 5= 530A heater voltage, (amplified inside TS-530A, negative reading)
- 6= TS-530A heater current (actually, voltage across current sense resistor in volts)
- 7= TS-530A set point voltage (V).

ADC [1..1000] A/D conversion command

The A/D conversion is made from voltage that has been previously selected for measurement by using the DISn command. The ADC command can be used both in local and remote modes.

ADC1 makes one single measurement, ADCn makes n successive measurements and calculates their average. Conversions take 0.4 seconds each, rate is 2.5 conversions/second.

If the result is an exact zero, the conversion is automatically repeated for detecting possible overrange (because the ICL7135 ADC yields a blinking zero in case of an overload).

ADC? A/D conversion query

This query returns the mean value of n conversions (see above). The conversion result is given as an integer -19999 to 19999 for ADC input voltages -2V..+2V. Polarity is indicated by minus sign and it is also returned by the POL? query. Use ADCn and ADC? at least for measuring items 4..6, which are not directly dependent on range. Use RESn and RES? for measuring items 0, 1 or 7. Items 2 and 3 can be measured using either command. ADC overrange yields an exact zero, which can be distinguished from a real zero by checking the OVR?. See also OVR? and ARN.

RES [1..1000] A/D conversion command for resistance.

The ADC reading is scaled by the currently selected resistance range, therefore measurement of display items 4-6 can produce misleading results.

RES1 makes a single conversion, RESn makes n successive measurements at 0.4 second intervals and their mean value is calculated.

If the A/D converter outputs an exact zero, the conversion is automatically repeated for detecting a possible overload. *The OVR? check should be a routine part of your application program.*

RES? Query for resistance

Resistance is output as a floating point number with four decimal places for R and ΔR displays, and with five decimal places for $10 \times \Delta R$ display. It is scaled by the currently selected resistance range. The reading may include a preceding minus sign. Arduino Mega2560 does not support output in exponential form.

RES? query can be preceded either by and ADCn or RESn command, they are actually one and the same function. The result remains valid and can be queried until the next ADC or RES command.



POL? Polarity query

The ADC? and RES? return values are preceded by a minus-sign if the reading is negative. You will probably not need this query, it is used by the firmware. POL? returns
0= negative
1= positive

OVR?, OVL? Overrange query.

An overrange-reading from the ICL7135 ADC is an exact zero, which can be distinguished from a true zero by repeating once each measurement that yields an exact zero. A BUSY bit from the A/D converter alternates if overrange, which separates these two cases. The AVS47-Serial/USB does this check automatically.

If a *single* A/D conversion is made by using ADC or RES command and the result is overrange, the ADC? reading is coerced to 20001 and the RES? reading is coerced to 2000100.0 which both are impossible readings in normal operation. In addition, the OVR? bit is 1 and ERR? is "ADC overload".

If *n* in ADC_n or RES_n is greater than 1, the average may contain one or more overrange- readings. No coercion is made, as the average can still be useful. Overrange detections are internally OR'ed together. OVR?=1 then indicates that the final average contains at least one overrange reading (which is 0). Such an average is distorted and perhaps it should not be used.

If there is any possibility for such a situation, use autoranging. It will react to the first overranged conversion and, after autoranging and autoranging delay, averaging is re-started. This quarantees a correct mean value.

OVR? returns:

0= no overrange
1= reading was overrange, or at least one sample in the average was overrange.

OVR? gets information from the A/D conversions. Do not use OVR? alone without a preceding ADC_x conversion command. Instead, *reading* the conversion result is not necessary for asking OVR?.

MIN?, MAX? These queries return the minimum and maximum resistance values (in ohms) of conversions averaged for one RES_n or ADC_n command.

STD? Standard deviation of the averaged A/D conversions as resistance in ohms. Use a long average for getting a reliable STD. If noise is purely random and white, STD is proportional to the noise at the bridge input. If data has a trend, STD is not useful.

QRATIO? This is simply (MAX-MIN)/STD. For a purely random white noise, QRATIO is about 5. A much higher figure may tell about external interference in data, whereas a much lower figure can tell about insufficient number of samples. QRATIO is also affected by the digitising step if excitation is high. It is most useful at low excitations.

REPEAT This command repeats one command line continuously. The commands and queries behave exactly similarly as if only one line were sent. If the line includes queries, the CPU sends responses automatically and your program can read them as they come. *This is an exception to the rule that the CPU does not send anything by itself, without an explicit query.* Repeating is stopped by sending anything via the RS232 line. An example using a hyperterminal program:

```
ADC5;ADC?;REPEAT
```

response:

```
12345  
12346  
12344 etc.
```

The REPEAT command must be the last item on a command line. In the repeating state, the bridge gives new readings 2.5 times/second. Your application program can read all responses as they come, or you can read only when readings are needed. There is no buffer for saving old results in the CPU.

Stop continuous repeating by sending first any character to the CPU and give then new instructions on a new command line.



ARN [0..30] Autorange mode command.

In autorange mode, the absolute value of each A/D conversion result is compared against two limits:

- If lower than 1800 (out of 19999), range is changed downwards, provided that it is not already 2Ω .
- If the reading is higher than 19900, the range is changed upwards, provided that it is not already $2M\Omega$.
- If the argument in ADCn or RESn is greater than 1 (average of many conversions), the first encountered overloaded or underranged conversion causes an autoranging operation, and after a delay, averaging is started from the beginning. This guarantees a correct average.

The ICL7135 ADC circuit is overloaded, if reading exceeds 19999. Then the AVS-47B front panel display shows blinking zeros and an internal overload indicator blinks between true and false. During overload, a measurement like ADC;ADC? returns an exact zero, which cannot be distinguished from a real exact zero.

Because of this uncertainty, the AVS47-Serial/USB repeats once any conversion that returned zero, which enables detection of overload. The overload indicator is set to 1 (queried by OVR?).

ARN 0 means manual ranging. Argument higher than 0 enables autoranging. A value between 1 and 30 determines, in seconds, how long the system waits, after having changed the range, until measurements can be started again, or until a new autorange operation can take place, or until any new command can be performed. A time like 5 seconds may suffice at high excitations, when the bridge settles quickly. A longer time, up to 15-20 seconds, should be used at the lowest excitation in order to guarantee good balance before readings are taken.

If the resistance changes several decades, which is typical when switching channel, the settling delay time of x seconds is applied after each change of range. This can make scanning unnecessarily slow or too fast.

You can avoid this by storing in your own program the last used range and excitation settings for each scanned channel. Disable autorange and set excitation to $3\mu\text{V}$ before selecting a new channel, send the saved previous good settings to the bridge, wait for settling (DLYz) and finally re-enable autoranging. This will speed up scanning, if sensors have very different values. You can still be sure that there are no over- or underranged readings.

Use the RAN? query if you want to check the possibly changed range.

Autoranging is especially useful in scanning. If you do not use autoranging, you **MUST** check with OVR? whether the single conversion or the average contained overload.

NOTE: The AVS-47B's hardware autoranging must not be enabled when the bridge is under remote control. This would cause a rival condition.



OTHER COMMANDS AND QUERIES

OPC? Operation complete query

This query can be placed after slow operations, like long averaging. When encountered, it places a “1” into the output queue. It should be the last item on a command row.

DLY [0..30] Delay command

The argument is delay time in seconds. Use DLY for giving the AVS-47B time to settle after a change in channel, range or excitation.

Note that autorange has its own settling delay after each change of range before starting to make A/D conversions. This delay is determined by n in ARNn (see ARN).

SCK[1..10] This “sign-check” function is an alternative to the fixed DLY delay. It is useful after a change in range or input channel. When the command interpreter encounters SCKn, the function starts to make A/D conversions at maximum speed and form differences between successive readings. This continues until two successive differences have had opposite signs n times, or the function has found n sets of 3 successive equal readings. Then the function concludes that data is either within the peak-to-peak noise or it has settled to within display resolution. The function exits and operation continues with the next command, which is typically ADCn.

The SCK function cannot prevent the A/D converter from being overloaded. In such a case, the ADC outputs only zeros. SCK behaves as if it had received three equal readings (zeros in this case) and exits without any additional delay.

The described method does not work if the data has a trend that is large enough to prevent differences from changing their sign. If there is such a trend, the function exits after a time-out of about 30 seconds.

The function is faster if n is small, but then the

data may not have settled properly. A large n guarantees better settling but is slower.

Some examples on how SCK could be used:

RAN3;SCK4;RES10;RES?

MUX4;RAN5;EXC2;SCK5;RES10;RES?

SCK works nicely if the measured resistance is within the selected range. In case of ADC overload, SCK exits quickly and the succeeding A/D conversion yields a zero resistance.

This can be prevented by using autorange, but autoranging does not solve the problem of good settling. **Scanning** is a situation where a good tradeoff between speed and settling accuracy is difficult to find. Following is just an idea of how automatic scanning could be programmed.

Autoranging is enabled by ARN30 or some other long delay. After having measured a scanned sensor, its range is asked by RAN? and saved as a variable for this channel. The command line could be

EXC1;MUXnew;RANnew;EXCnew;SCK1;
ADC1;SCK5;RESn;RES?

Explanation: EXC1 selects the lowest excitation for preventing unintentional heating of the next sensor during switching. MUXnew selects the new channel. RANnew is the stored range that was previously good for this sensor. EXCnew is the suitable excitation for this sensor. All these come into effect immediately.

The bridge starts to slew toward the new sensor’s value. SCK1 waits until the slewing changes direction for the first time. ADC1 makes one conversion, if there is no overload. The result is not used. SCK5 expects differences to change sign n=5 times, which indicates a reasonably good settling. Finally an average is taken and the result is asked.

If the sensor’s value exceeds the range or is below underrange limit (1800 counts), autoranging must act. ADC1 will then make so many autoranging operations that are needed for finding a correct range. Because settling



on a low excitation is very slow, a long autoranging delay (like the suggested 30 seconds) is necessary. A too short delay can cause oscillation between ranges and **this condition should be strictly avoided** - one has to reboot the CPU by disconnecting its power! Once a correct range has been found, SCK5 checks for final settling.

The long autoranging delay slows down scanning when autoranging is needed. As soon as the preset ranges are correct again, which is the normal situation, scanning is faster because fixed settling delays are replaced by SCK. It takes only the required time, not more.

RST Reset command

This command works only when the program is idling, i.e. it is not extracting commands from a previously received command line or performing those commands. *RST cannot be used for stopping long averaging or any other pending operation.* In such a case, wait or reboot the interface.

The RST command sets the AVS-47B into a known safe state: input 200Ω, channel 0, range 2M, excitation 3μV and display to R (resistance). Response line terminator is CRLF and command separator is semicolon. The AVS-47B is left in local mode, which is shown by the blanked REMOTE light.

RST works differently than initial power-on start: both leave the bridge in local mode, but start-up does not alter the previous local-mode settings of the bridge whereas RST changes settings to these “safe values”.

ERR? Error query

The AVS47-Serial/USB has limited error reporting capability. Errors are not reported automatically, they must be queried using the ERR? query. Possible responses:

0: No error

command XXXyyy not recognized:
The letter part of a command has been misspelled or is non-existent

query XXX? not recognized: The letter part of a query has been misspelled or is non-existent.

argument in XXXyyy exceeds maximum:

argument in XXXyyy less than minimum:

Every command has upper and lower limits for its argument. If the given argument is outside these limits, it is coerced to the nearest limit and an error message is available.

AL input line (No 4 black in -F version) stays at 0:

This input line from the bridge to the CPU is used for synchronising the operation so that each A/D conversion is read only once. The error message is for trouble-shooting in a case where communication fails.

ADC overload: The input voltage to the ADC has exceeded 1.9999 Volts, which is equivalent to the upper limit of each resistance measuring range.

timeout in SCK: The sign-checking delay function has not found expected criteria. Reasons can be abnormally slow settling or a trend in data. The function will exit after a timeout. Then this error message becomes available.

Error messages from one command line are chained. The messages are verbal instead of being numerical codes because they are intended to be only programming aids. The application program should minimise the possibility of error situations.

Error register is cleared by ERR?



COMMANDS FOR THE TS-530A TEMPERATURE CONTROLLER

The **TS-530A** is a very old design, and it has been discontinued. We offer now a low-cost analog temperature controller option for the **AVS-48** bridge. However, the AVS47-Serial/USB includes also commands for the TS-530A. They are for customers, who already own this controller. These commands have no corresponding queries. The analog setpoint voltage, the heater output voltage and the heater current can be *measured* by the A/D converter of the AVS-47B bridge (see the DIS command and refer also to the TS-530A and AVS-47B manuals).

The TS-530A must be connected to the AVS-47B with the supplied 37-way ribbon cable for data and with the supplied short BNC-BNC coaxial cable for the analog output from the bridge.

The TS-530A does not have a separate “remote” mode. Neither can the front panel settings be read remotely. This means that one can -but should not- change the remotely programmed PID settings by using the TS-530A front panel switches, and the firmware has no way to detect it. If this appears to be a problem, write your program so that the settings of the TS-530A are updated frequently. Re-programming the existing settings will not disturb the analog control circuitry in any way.

SPT [10..42000] Set point command

Set point is given as a long integer. One digit corresponds to 100µV and the range is from 1mV to 4.2 Volts. The slow integrating D/A converter of the TS-530A is very accurate and linear, but it does not go to exact zero, therefore 1mV is the minimum. Arguments less than 10 are coerced to 10. Maximum value for the converter is 4.2Volts, corresponding to 42000, but only 2 Volts maximum is meaningful with the AVS-47B.

If you want to give the set point in resistance, you must scale and convert it yourself to a long or unsigned integer suitable for this

converter. For example, SPT10000 produces 1 Volt setpoint. If the AVS-47B measures on range RAN, calculate the set point integer from set point resistance R_s as follows:

$$SPT = R_s / (10 ^ (RAN-1)) * 10000$$

where R_s =setpoint in ohms

If desired set point is e.g. 110 ohms and range=200R, (RAN=3)

$$SPT = 110 / (10 ^ (3-1)) * 10000 = 11000$$

PRO [0..11] Proportional gain command

Gain increases in steps of five decibels. Values are very approximate.

0..14: 5-10-15-20...60dB

15: no gain. Input of the proportional amplifier is connected to ground.

ITC [0..15] Integrator time constant command.

Values are very approximate.

0: integrator is reset to zero. P and PD mode control

1-10: 1-2-5-10-20..1000s. Higher number means weaker integration

11: analog integrator is latched by leaving its input open

12..15: integrator is reset to zero. Same as ITC=0.

DTC [0..7] Derivator time constant command.

Values are very approximate.

0: No derivation. P and PI mode control.

1..7: 1-2-5-10-20-50-100s

Higher number means stronger derivation. High proportional gain with strong derivation leads easily to oscillation of the control system.

BIA [0..5] Power bias command

Power bias can be used to reduce control error in P and PD modes. It is not useful in PI or PID modes.

0..5: 0-20-40-60-80-100% of maximum heater power. The highest setting is sufficient for maximum output on the selected heater range when proportional input is zero and integrator is reset.



POW [0..7] Heater power range command

Power ranges are calculated for a 100Ω heater. If heater resistance is higher, output voltage compliance (about 10V) will reduce the maximum power. If heater resistance is lower, the available output current (100mA) will reduce the maximum possible output.

- 0: Heater output is disabled
- 1..7: 1μW-10μW-100μW...1W

The heater output stage has seven current sensing resistors 10kΩ, 3.16kΩ, 1kΩ, 316Ω, 100Ω, 31.6Ω and 10Ω corresponding to heater ranges 1μW..1W. One volt across a sensing resistor means full output of the range. Based on the above figures, you can calculate correct ranges for heaters other than 100Ω. Similarly, you can measure the output current using DIS6 and calculate the heating power from $R_H * I^2$.

CABLE SPECIFICATIONS

The AVS47-Serial/USB-F comes with two wire cables, Picobus Cable and Serial Cable. The 1.5 meter Picobus cable connects Box1 of the converter to the AVS-47B and the 1.5 meter Serial cable connects Box2 to the computer directly or via an USB-232 converter. This Picobus cable is distinguished from the Picobus cable, which is supplied with the AVS-47B by its male 25- and 15 pin D-connectors. The resistance bridge comes with a cable that has male and female 25-way D-connectors.

Picobus Cable (PB25P15P6W1.5M)

DB25P	DA15P	Description
1	-	braid grounded at both ends
4	4	CP clock from box to bridge
5	5	DI data from bridge to box
6	6	AL alarm line from bridge to box
7	7	Isolated ground
20	15	DC data from box to bridge
9	9	Isolated +5V (referred to pin 7)

Other pins are unused.

The male DB25P and male DA15P are connected by a braided (shielded) cable with 6 conductors (e.g. Tasker C6015). Length: 1.5 meters.

Note that the shielding braid must be grounded to the connector shell at both ends. The braid is not connected to pin 7.

Serial Cable (RS9P9S7W1.5M)

The male DE9P and female DE9S are connected by a braided 1:1 cable of 6 conductors (e.g. Tasker C6015). Length: 1.5 meters.

DE9S	DE9P	RS232	Description
1	-	-	
6	DSR		AL for Picobus applications
2	RXD		RS232 output box=>computer
7	RTS		CP for Picobus applications
3	TXD		RS232 input computer=>box
8	CTS		DI for Picobus applications
4	DTR		DC for Picobus applications
9	-		
5			Computer ground = shielding braid

Shielding braid is connected, in addition to pins 5, also to **both** connector shells.

RE-PROGRAMMING THE AVS47-Serial/USB-F

The firmware can be updated by reprogramming the Arduino Mega2560 board. In order to do this, you need

- a USB cable (type A/B)
- Arduino development software for Mega2560. This can be downloaded from Arduino WEB site.
- The new firmware version. It is available from us.

1. Follow Arduino’s instructions to download and install their programming environment software.
2. Open the four screws holding the “front” panel of the AVS47-Serial/USB, the panel with the DA15 connector. There must be no cable from this connector to the AVS-47B. Plug the “A” type connector into the USB connector of



For reprogramming the CPU you need to open one end plate. Removing the top cover is not necessary. The CPU gets its power from the USB line, do not connect the mains adapter while reprogramming.

your computer and the “B” end into the “hidden” USB connector inside the AVS47-Serial/USB. The box will now start, because it gets power from the USB.

3. Make on your hard disk a directory that has the same name as the new firmware file, but without extension, e.g. “avs47_serial_usb_1r3”. Arduino saves source codes into directories that have the same basename as the source code file. It creates such directories automatically, so it is best to create the directory yourself in a place where you want it be.
4. Place the new version of the firmware (e.g. “avs47_serial_usb_1r3.ino” in the new directory (Arduino calls the source code a “sketch”).
5. Start the Arduino environment. Under Tools, select Arduino Mega2560 board type. Select also the USB port that your computer has assigned to the CPU box.
6. Under the File menu, navigate to the new firmware version and open it into the environment.
7. Under the Sketch menu, select UPLOAD. If you do not get any error messages, updating has been done in a few seconds. You can now detach the USB cable, fix the rear panel and connect the 25/15 Picobus cable to the resistance bridge and the +12V power plug. Then

test the new firmware using an RS232 terminal program or your own software. Suggestion: start conversation always by issuing the IDN? query.

Before we email an updated version to you, please check and tell us your old firmware version so, that we can send also the old version for backup.

RV-Elektroniikka Oy Picowatt
Veromiehentie 14
FI-01510 VANTAA
Finland

e-mail: reijo.voutilainen@picowatt.fi
WEB: www.picowatt.fi

AVS47-Serial/USB-F TROUBLE SHOOTING IDEAS

If you have difficulties in getting the AVS47-Serial/USB-F protocol converter to work, you can try the following trouble-shooting procedure. The problem might not be found this way, but it at least gives us valuable knowledge of where in the system the problem probably is. For making these tests, you need a hyperterminal program, or any other program that allows you to send serial commands to the AVS47-Serial/USB-F. Typically, your computer has only USB ports and then you need a USB-232 adapter between the computer and the protocol converter box.

Communication between the bridge and the box uses four signals using visible light. Two signals go from the box to the bridge and two from the bridge to the box. They are

AL: “alarm” line telling that an A/D conversion is ready to be read from the bridge

DI: data bits from the bridge (“instrument”) to the box

CP: clock pulses from the box to the bridge

DC: data bits from the box (seen as the “computer” by the AVS) to the bridge.

The sent and received data bits are synchronized to the clock. Communication *between the box and the bridge* is based on our proprietary synchronous, serial “Picobus” protocol. Although this commu-



nication is serial, it is NOT RS232. Please do not mix synchronous and asynchronous serial formats. In synchronous Picobus communication, a 48 bits long string contains the complete state of the bridge, and it is sent every time. While sending the string, and using the same clock pulses, the CPU reads a response from the bridge. It represents the state of the bridge just before the transaction. Asynchronous serial communication *between the box and the computer*, on the other hand, uses short mnemonic commands for controlling individual bridge settings and for reading conversion results and settings currently in effect.

The expression “protocol converter” means that the complicated synchronous protocol of the bridge is programmatically turned to easily understandable commands and arguments that are sent and received in the (still) popular serial format that uses the legacy RS232 hardware standard. Because of the slow speed of the AVS-47B and tiny amounts of data, the simplest possible protocol can be used (9600 bauds, 8 data bits, no parity and one stop bit. No handshaking, briefly 9600,8,N,1). It is based on signals called TxD, RxD and ground. TxD transfers data from the external computer to the box, and RxD from the box back to the computer.

PROCEDURE

- 1) Remove the four cross-head screws that hold the rear panel of the Box2 (the panel with the four optical connectors). Then pull off the top cover lid. Place the box, so that the loose panel is to the left. You can see a row of several green and two red LEDs. They are for trouble shooting. LEDs on the left (“bridge side”) show signals to an from the primary interface of the AVS-47B. They are marked with AL, DI, CP and DC. At right are the two asynchronous signals, TxD and RxD. The DI, AL, CP and DC signals on the right have no meaning in this AVS47-Serial/USB-W application.
- 2) Connect the serial cable (RS9P9S7W1.5M) from Box2 to the USB-232 converter or to your computer, if it has a physical COM: port (RS232 port). Do not yet connect the optical cable to the resistance bridge.

- 3) Connect the +12V DC power plug to Box2. The green “ON” light and one yellow LED on the Arduino board should turn on. All trouble-shooting LEDs (except possibly TxD, CP or DC on the right) remain off. All optos should also be off.
- 4) Send command RST (“reset”) from the computer. You should see very short activity of the TxD LED. It is not dependent on the Arduino board. If you do not see any activity although power is ON, your computer program may be configured wrong, or the cable is not in condition. *If you are not using our original cable (RS9P9S7W1.5M), please check that it is wired 1:1.* Pins 2 of both connectors must have been connected together, and pins 3 together. Pins 5 are ground. (The so-called “null modem cable” has cross-connected pins 2 and 3. It is not suitable, because the cross-connection is made inside Box2).
- 5) Send command *IDN?. This should return “PI-COWATT, AVS47-SERIAL/USB, REV 1.1” (or a later revision). You should now see short activity also at the RxD light. The firmware program has started successfully. Much of the Arduino board seems to be in order.
- 6) Send command RTS1. The green CP light on the left “bridge side” and opto transmitter LED No. 1 turn on.
- 7) Send command DTR1. The DC light and opto No. 2 should turn on. If this and 6) work, the CPU can send data to the bridge.
- 8) Send commands CTS? and DSR? in turn. Both queries should return 0 to the computer. These signals can be asserted only by the bridge.
- 9) Using one fibre of the Picolink cable, connect the light from transmitter opto No. 1 to receiver opto No. 3. The “DI “ LED should turn on and query CTS? from the computer should return 1. This means that data from the bridge can be received.
This is now a good opportunity to pull slowly the fibre outwards until the DC led blanks and push back again. You can also move the cable sideways and see, how this affects the DC light.
- 10) Move the wire from opto 3 to opto 4. The “AL” LED should turn on and query DSR? should return 1. The Alarm signal can be received from bridge.



NOTE: You can alternatively illuminate optos 3 and 4 by a bright pocket torch instead of using a fibre.

- 11) Remove the fibre. Send the RST command again and the lighted LEDs and optos should go off.
- 12) With the AVS-47B off, connect Box1 of the AVS47-Serial/USB-F to the bridge using the 1.5 meter 25P/15P cable (PB25P15P6W1.5M). Turn on the bridge. Opto transmitter No.4 (AL) turns on.

If this does not happen, the problem may be a) in optoisolator ISO204 (AL) in the bridge, b) in the cable or c) in opto No. 4 on the Picolink board.

Set the bridge manually to CAL, 200 Ω range, 3mV excitation and R display.

- 13) Connect the four fibres of the Picolink optical cable. Push the fibres into the transmitters/receivers as deep as they go. Tighten properly **but gently by fingers** - the plastic threads are not strong. Do not destroy them. The cable is wired 1-to-1, 2-to-2 etc. Recommended colors are 1=black, 2=red, 3=yellow and 4=blue. The green AL light in Box2 should now be on.
- 14) Send command REM1. This sets the bridge in remote mode, shown by the yellow REMOTE led on the front panel. The bridge state should have remained unchanged.

If the bridge remained in local mode, the problem may be a) in optoisolator ISO201 (CP) or ISO202 (DC) in the bridge, b) in the 25P/15P cable or c) in opto receiver No. 3 or 4 of Box1. The cable specification is elsewhere in this manual.

- 15) Send command RAN4. The range should advance from 200 Ω to 2k Ω . There should be no problems with this command, if the previous ones have worked. This verifies that bridge settings can be remotely controlled. You can also test MUX, EXC and DIS commands, if the problem is with channel, excitation or display.
- 16) Send command ADC10. Ten A/D conversions are made at 0.4 second intervals. All four LEDs on the bridge side in Box2 should show activity. The firmware resets the AL signal and then waits max 0.4 seconds until the next conversion turns

AL on. A reading is taken and AL is reset again. The same cycle repeats 10 times. A query ADC? returns the average of the conversions.

If the response is zero or something strange, optoisolator ISO203 (DI) may be defective.

If all these tests ended up successfully, your problem may be in the application program.

If the test stops to a failure, please let us know your results so that we can try to help.

LabView is a trade mark of National Instruments, USA.

DECLARATION OF CONFORMITY



Manufacturer: **RV-Elektroniikka Oy Picowatt**
Address: Veromiehentie 14
01510 VANTAA
Finland
Telephone: +358 50 337 5192
E-Mail: reijo.voutilainen@picowatt.fi

declares that under our sole responsibility

Product Name: AVS47-Serial/USB-F Converter
Product Description: Protocol Converter between the Picobus Primary interface of the AVS-47B Resistance Bridge and RS232 or USB port of an external computer.

is in conformity with the following Directives:

2004/108/EC: Electromagnetic Compatibility
2011/65/EU: ROHS Directive

and that the following harmonized standards have been applied:

EN 50 081-1: Generic emission standard, Part 1: Residential, commercial and light industry
EN 50 082-1: Generic immunity standard, Part 1: Residential, commercial and light industry
EN 50 581: ROHS

Additional information: This product uses +12V power from an external mains adapter

Vantaa, 14 March 2016

RV-Elektroniikka Oy Picowatt

Reijo Voutilainen

A handwritten signature in black ink, appearing to read 'Reijo Voutilainen', with a horizontal line extending to the right.

President



INDEX

A

ADC 13
A/D conversion command 13
AL 19
AL? 11
Alarm line query 11
AL Picobus signal line 11
Arduino Mega2560 5, 19
Argument part 8
ARN autorange mode command 14
Autorange mode 15
Autoranging 16
Autoranging, hardware 15
AVS47-IB 4
AVS-48 Resistance Bridge 18

B

Background 4
BIA 18
Bias power command 18
Blinking zero 13, 15

C

Cable specifications 19
Carriage return character 10
Case sensitivity of commands 8
Command delimiter 8, 11
Command line 8
Command part 8
Command separator 8
Comma-separated format 11
CP 19
CTS 19
Current sensing resistors 19

D

DC 19
D-connectors 19
Delay 15
Delay command 16
Delimiter 8
Derivator time constant 18
Deviation 12
DI 19
DIFFERENCE output 12
Directory 20
DIS 12
Disable heater output 19
Display selector cmd. 12
DLY 11, 15, 16

DSR 19
DTC 18
DTR 19

E

Environment, Arduino 19
ERR error query 17
EXC 12
Excitation command 12
Excitation voltage 13
Exponential format 8

G

GPIO-Picobus 4

H

Hardware version query 11
Heater current of TS-530A 13
Heater power range cmd. 19
Heater resistance 19
Heater voltage of TS-530A 13
HW? 11
Hyperterminal 8

I

Identification query 11
Idle 17
IDN? 11
IEEE-488.2 11
INP 11
Input selector command 11
Integrator time constant 18
Isolation, galvanic 4
ITC 18

J

Jumper JP203 5, 10

L

LabView 4, 10
LIM 11
Line terminator 8
Line terminator command 11
Linux computer 4
Local 17

M

Mac computer 4
MAG 12
Magnifier query 12
Maximum number of characters 8
MAX? maximum query 14



MIN? minimum query 14
Multiplexer channel 11
MUX 11

N

Newline character 10
NULDEV 12

O

OPC 16
Operation complete query 16
Oscillation between ranges 17
Overload 15
Overrange 13
Overrange query 14
OVL 13, 14
OVR 14, 15

P

Picobus 4
Picobus cable 19
Picobus communication 4
POL 14
Polarity query 14
POW 19
Power bias command 18
Power-on start 17
PRO 18
Proportional gain command 18
Protocol converter 4

Q

QRATIO? "quality ratio" query 14
Query 8

R

RAN 11
Range command 11
Rebooting 17
REF 12
Reference command 12
REF POT / REF MEM 12
REM 11
Remote mode 17
Remote mode command 11
REPEAT command 14
RES 12, 13
Resetting the CPU 6
Resistance measurement 13
Response headers 8
Responses 8
Revision history 26
RFS 12

RS232 format 6
RST reset command 6, 17
RTS 19
RXD 4, 19

S

Safe state 17
Scanning 4, 15, 16
SCK sign-checking delay 16
Send/Receive Serial 10
Serial cable 19
Serial format 10
Set point command 18
Set point of TS-530A 13
set point voltage of TS-530A 13
SET REF switch 12
Settling time 11, 16
Short circuit pieces 5
Sketch 20
SPT 13, 18
STD? standard deviation query 14

T

TER 8, 10, 11
Trouble shooting 20
TS-530A 13, 18
TXD 4, 19

U

Underrange 15
USB-232 converter 4, 6, 19
USB cable 19
USB connector 19
USB-Picobus 4

V

Virtual Com: port 4
Voltage levels 5

W

Warranty 3



REVISION HISTORY

1R0 => 1R1 2016-07-15

- New DTR and RTS commands
- New DSR? and CTS? queries
- Timeout if AL is not set within 1 sec

1R1 => 1R2 2018-03-25

- Correction in ADC function
- New queries MIN?, MAX?, STD?, QRATIO?
- Correction in operation of OVR query
- Correction in operation of ERR? query
- New sign checking delay function SCK
- Multiple error messages are chained

1R2 => 1R3 2018-06-03

- Added missing field for serial number (=0) in the IDN? response and removed spaces.
- Both OVR? and OVL? can be used for check-ign overrange
- Overrange of a single A/D conversion is now decoded additionally into readings ADC?=20001 and RES?=2000100.0000
- Maximum length of the input string added from 60 to 255 characters and length of a command or query from 6 to 20 characters
- Added new REPEAT command