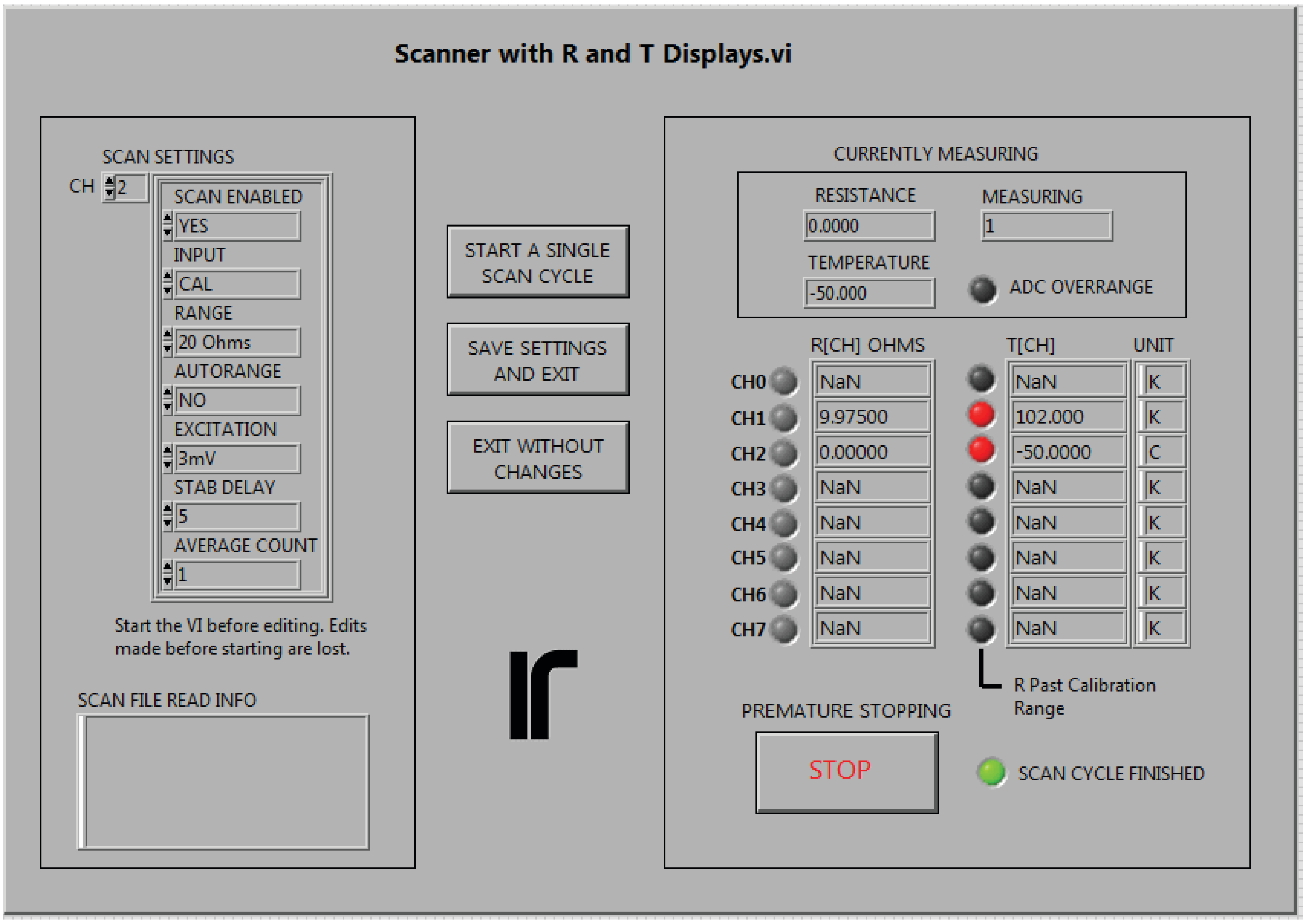
USB-Picobus
AVS-47B Software for LabView™

User Guide



Scanner with R and T Displays.vi

SCAN SETTINGS

CH 2

SCAN ENABLED: YES
INPUT: CAL
RANGE: 20 Ohms
AUTORANGE: NO
EXCITATION: 3mV
STAB DELAY: 5
AVERAGE COUNT: 1

Start the VI before editing. Edits made before starting are lost.

SCAN FILE READ INFO

CURRENTLY MEASURING

RESISTANCE: 0.0000
MEASURING: 1
TEMPERATURE: -50.000

	R[CH] OHMS	T[CH]	UNIT
CH0	NaN	NaN	K
CH1	9.97500	102.000	K
CH2	0.00000	-50.0000	C
CH3	NaN	NaN	K
CH4	NaN	NaN	K
CH5	NaN	NaN	K
CH6	NaN	NaN	K
CH7	NaN	NaN	K

R Past Calibration Range

PICOWATT
Veromiehentie 14
FI-01510 VANTAA, Finland
phone 358 50 3375192
fax 358 9 822184
Internet: www.picowatt.fi
e-mail: reijo.voutilainen@picowatt.fi

CONTENTS

INTRODUCTION

USB-Picobus software for LabView 3
Requirements 3
VI’s in this package 4
Software installation 5
Installation of the USB-232 Converter 5
Connecting Picobus 5
Jumper settings for older AVS-47x brigdes 5

VIRTUAL INSTRUMENTS

Manage Configuration File.vi 6
Settings of the AVS-47B 6
Settings of the TS-530A 8
InitPort.vi 11
GoLocal.vi 11
Simplest.vi 11
ReadAvs47b.vi 12
Free Running Avs47b.vi 13
FunctionalResistanceGlobal.vi 15
ExampleOnUsingFunctionalGlobalVariable.vi 15
Free Running Avs47b With Autoranging.vi 16
Autoranging.vi 16
Free Running AVS47b with Temperature Display.vi 17
Manage RT Files.vi 19
Checking the calibration files 20
Structure of a calibration file 20
Interpolate R to T.vi 22
Test AVS47B Remote Control. vi 23
SendConfigurationToAvs47b.vi 24
Test TS530A Remote Control.vi 25
Avs47Ave.vi 26
OhmsNumberToStringWithUnit.vi 28
TemperatureControlSystem with R display.vi 28
Control System with R and T displays.vi 29
Interpolate T to R.vi 30
Scanner with R and T Displays.vi 31
Manage Scan Info File.vi 33
Low-level VIs for Picobus 34
INDEX 36

INTRODUCTION

USB-Picobus software for LabView

The AVS-47B AC Resistance Bridge is an analog instrument without intelligent digital circuits. This guarantees the lowest possible RF emissions inside a cryostat room.

Common computer interfaces like IEEE-488, USB or RS-232 all require digital intelligence, so they are impossible choices for an analog instrument. In order to facilitate remote control, the AVS-47B has been equipped with a simple, synchronous, optically isolated primary interface, which is called **Picobus**.

Although Picobus can be implemented using a PC-computer’s RS-232 (Com:) port’s four hardware handshake signals, it **IS NOT RS-232**. RS-232 is an asynchronous interface whereas Picobus is strictly *synchronous* and almost bullet-proof. Thanks to the introduction of USB-232 converters, Picobus no longer requires, that the controlling PC be equipped with a serial port.

This user guide assumes that your Windows computer runs a recent version of National Instrument’s LabView™ , and that you have their USB-232 converter (NI part number 778472-01, only for Windows) for converting your computer’s USB port into an RS-232 port with full hardware handshaking capabilities (RTS, DTR, CTS and DSR). This external and reasonably priced converter is easy to install, but if your computer still has a physical Com: port, you do not need the converter.

The **USB-Picobus software package** consist of 13 middle-level VIs (*Virtual Instruments*) for building applications, 11 low-level routines used by middle-level VIs and 12 application examples. The low-level routines must not be altered, and there should be no need to use them directly. The application examples range from simple to complicated . They are suitable for use as independent “instruments” or systems, but after some modifications one may be able to them also as parts of a bigger system.

LabView is trade mark of National Instruments, USA

> This software is not a commercial product. It is given for free, without any kind of warranties, and the VIs can be freely modified by the customer to better suit his/her needs. These VIs and this user guide may contain errors, and we would be glad to get feedback, corrections and suggestions for improvements. You can download the sofware package from our WEB site at www.picowatt.fi.

Requirements

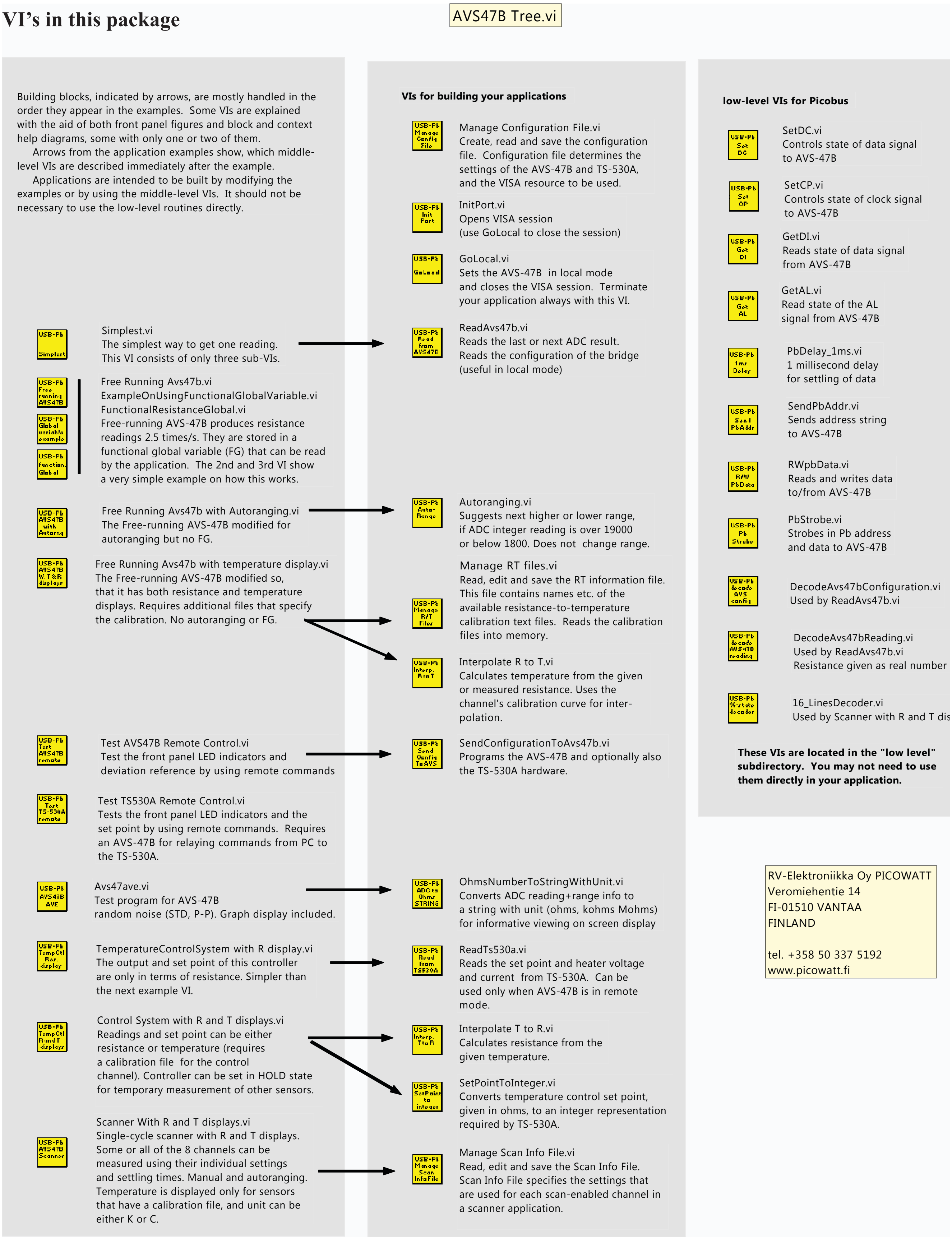
- Windows operating system and a free USB port.
- LabView2012 or later. These VIs were written and compiled with LV2012 Full Development version under Windows 7 operating system.
- AVS-47B AC Resistance Bridge. Older bridge versions (-47 and -47A) can also be used, but without optical isolation between the bridge and the computer.
- USB-232 Converter from National Instruments.
- A 25-to-9 pin adapter. This is a standard accessory of the AVS-47B. It is used between the USB-232 converter and the Picobus Cable.
- Picobus Cable. This is a standard AVS-47B accessory. Using this specially wired 5 meter long cable quarantees optical isolation. The shield is grounded permanently only at the converter end in order to prevent ground currents.

For display in terms of temperature:

- R/T calibration file for each sensor or sensor type that should have temperature display.

For temperature control:

- TS-530A Temperature Controller
- Data Cable (this short 37-way ribbon cable is a standard accessory of the TS-530A)
- Analog input cable (this short 30cm coaxial cable is a standard accessory of the TS-530A)
- Optimum heater resistance is 100 ohms.



VI’s in this package

AVS47B Tree.vi

Building blocks, indicated by arrows, are mostly handled in the order they appear in the examples. Some VIs are explained with the aid of both front panel figures and block and context help diagrams, some with only one or two of them.

Arrows from the application examples show, which middle-level VIs are described immediately after the example.

Applications are intended to be built by modifying the examples or by using the middle-level VIs. It should not be necessary to use the low-level routines directly.

Simplest.vi
The simplest way to get one reading.
This VI consists of only three sub-VIs.

Free Running Avs47b.vi
ExampleOnUsingFunctionalGlobalVariable.vi
FunctionalResistanceGlobal.vi
Free-running AVS-47B produces resistance readings 2.5 times/s. They are stored in a functional global variable (FG) that can be read by the application. The 2nd and 3rd VI show a very simple example on how this works.

Free Running Avs47b with Autoranging.vi
The Free-running AVS-47B modified for autoranging but no FG.

Free Running Avs47b with temperature display.vi
The Free-running AVS-47B modified so, that it has both resistance and temperature displays. Requires additional files that specify the calibration. No autoranging or FG.

Test AVS47B Remote Control.vi
Test the front panel LED indicators and deviation reference by using remote commands

Test TS530A Remote Control.vi
Tests the front panel LED indicators and the set point by using remote commands. Requires an AVS-47B for relaying commands from PC to the TS-530A.

Avs47ave.vi
Test program for AVS-47B random noise (STD, P-P). Graph display included.

TemperatureControlSystem with R display.vi
The output and set point of this controller are only in terms of resistance. Simpler than the next example VI.

Control System with R and T displays.vi
Readings and set point can be either resistance or temperature (requires a calibration file for the control channel). Controller can be set in HOLD state for temporary measurement of other sensors.

Scanner With R and T displays.vi
Single-cycle scanner with R and T displays. Some or all of the 8 channels can be measured using their individual settings and settling times. Manual and autoranging. Temperature is displayed only for sensors that have a calibration file, and unit can be either K or C.

VIs for building your applications

Manage Configuration File.vi
Create, read and save the configuration file. Configuration file determines the settings of the AVS-47B and TS-530A, and the VISA resource to be used.

InitPort.vi
Opens VISA session
(use GoLocal to close the session)

GoLocal.vi
Sets the AVS-47B in local mode and closes the VISA session. Terminate your application always with this VI.

ReadAvs47b.vi
Reads the last or next ADC result.
Reads the configuration of the bridge (useful in local mode)

Autoranging.vi
Suggests next higher or lower range, if ADC integer reading is over 19000 or below 1800. Does not change range.

Manage RT files.vi
Read, edit and save the RT information file. This file contains names etc. of the available resistance-to-temperature calibration text files. Reads the calibration files into memory.

Interpolate R to T.vi
Calculates temperature from the given or measured resistance. Uses the channel's calibration curve for interpolation.

SendConfigurationToAvs47b.vi
Programs the AVS-47B and optionally also the TS-530A hardware.

OhmsNumberToStringWithUnit.vi
Converts ADC reading+range info to a string with unit (ohms, kohms Mohms) for informative viewing on screen display

ReadTs530a.vi
Reads the set point and heater voltage and current from TS-530A. Can be used only when AVS-47B is in remote mode.

Interpolate T to R.vi
Calculates resistance from the given temperature.

SetPointToInteger.vi
Converts temperature control set point, given in ohms, to an integer representation required by TS-530A.

Manage Scan Info File.vi
Read, edit and save the Scan Info File. Scan Info File specifies the settings that are used for each scan-enabled channel in a scanner application.

low-level VIs for Picobus

SetDC.vi
Controls state of data signal to AVS-47B

SetCP.vi
Controls state of clock signal to AVS-47B

GetDI.vi
Reads state of data signal from AVS-47B

GetAL.vi
Read state of the AL signal from AVS-47B

PbDelay_1ms.vi
1 millisecond delay for settling of data

SendPbAddr.vi
Sends address string to AVS-47B

RWpbData.vi
Reads and writes data to/from AVS-47B

PbStrobe.vi
Strobes in Pb address and data to AVS-47B

DecodeAvs47bConfiguration.vi
Used by ReadAvs47b.vi

DecodeAvs47bReading.vi
Used by ReadAvs47b.vi
Resistance given as real number

16_LinesDecoder.vi
Used by Scanner with R and T dis

These VIs are located in the "low level" subdirectory. You may not need to use them directly in your application.

RV-Elektroniikka Oy PICOWATT
Veromiehentie 14
FI-01510 VANTAA
FINLAND

tel. +358 50 337 5192
www.picowatt.fi

Software installation

The software is downloaded as one ZIP file. The name of the ZIP file is the only place, where the revision number of the included VIs appears.

Create a new folder in which you extract all zipped files. Let us call this folder “USBpicobus”, for example. After unzipping, 11 files should have gone to subfolder “USBpicobus\Low Level Picobus VIs” and the rest of files -both the middle level files and the application examples - are in the “USBpicobus” folder. Your own USB-Picobus applications should also be in this folder (or you must change a large number of pathnames in the VIs).

Installation of the USB-232 Converter

Please follow National Instrument’s instructions for installing the converter. After installation of the driver, start NI’s “Measurement & Automation Explorer” , select “Devices and Interfaces” and verify that the **NI USB-232 Interface** is listed. Double click for seeing, which Com: port has been given for this converter (e.g. Com4:)

Connecting Picobus

Connect the 5 meter Picobus Cable to the USB-232 converter using the 25-to-9 pin adapter in between them. The Picobus Cable has 25 pin connectors for compatibility reasons (all AVS-47 family bridges since 1994 have 25 pin connectors, and so has also the GPIB-Picobus option AVS47-IB).

It is possible to omit the Picobus Cable, but then the cable shields connect the grounds of the computer and the bridge together, and optical isolation does not work. The shield of the Picobus Cable is connected **only** to the computer by default.

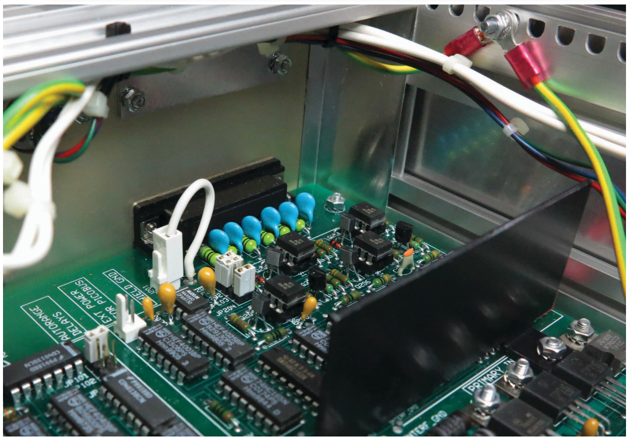
Jumper settings for older AVS-47x brigdes

It is possible to control even old AVS-47 bridges (prior to version “B”) remotely via Picobus and without the AVS47-IB protocol converter, but optical isolation cannot be achieved. This is because only the “B” version has a separate isolated power supply for the primary interface. Old bridges can get isolated power from the external AVS47-IB box, otherwise the power must be taken from the bridge itself, which in turn prohibits the isolation.

If using these VIs with an old AVS-47 or AVS-47A, check that jumpers JP201 and JP202 near to the rear panel 25-way connector are set as in the figure below.

If using the bridge with the GPIB box, be sure to remove JP201 and JP202. Failure to do so may lead to overheating and damage to the power supplies.

AVS-47B Bridges must have a short-circuit piece ONLY in JP204.



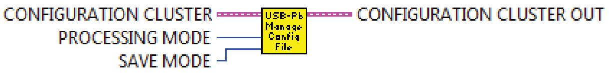
AVS-47 and AVS-47A Bridges must have three short-circuit pieces: JP201, JP202 and JP204. With these bridges, optical isolation requires an AVS47-IB box (GPIB option).

About this user guide

The main purpose of this software guide is to explain, how the middle-level VIs were intended to be used for creating applications. Therefore, the applications are described starting from the simplest and ending to the the most complex VI. The operation of middle-level VIs is explained after the application examples, where they are used, whereas the low-level routines are left to the end of this document. They are discussed only very briefly.

Simple **V**irtual **I**nstruments (VIs) are mostly described using LabView’s on-line help pictures and block diagrams. Block diagrams of the most complicated VIs are not printed on these pages. Then only front panels and the Context Help diagrams are shown.

It is recommended that you use your LabView for viewing the block diagrams.



VIRTUAL INSTRUMENTS

Manage Configuration File.vi

The configuration cluster is perhaps the most important variable in this software package. It contains all physical front panel settings of both the AVS-47B and the TS-530A plus a few system parameters. These settings and variables, most of which are (enumerated) integers, are *bundled* together to form a *cluster* (which corresponds to a “record” in some other programming languages).

The idea is, that changes in settings are made to the configuration cluster, which can then be used as input for many different VIs. The cluster contains always the last settings, and when an application ends, the configuration is saved in the configuration file. When an application is started, the configuration file is read.

OPERATING INSTRUCTIONS:

The **Manage Configuration File.vi** has three processing modes for different purposes.

1. NONE Wire the current configuration cluster to the input. The VI returns the cluster unchanged. Before that, it is saved into file “avs47bconfig.cnf” in the operating directory, if save mode is “SAVE .CNF FILE”. This mode exists mainly for saving the instrument settings before an application ends.

2. READ The VI tries to open “avs47bconfig.cnf” for reading.

   If succesful, the configuration file is read to be the output of this VI. Configuration is not saved unless the save mode is “SAVE .CNF FILE”. Saving immediately after reading is usually not meaningful, so the save mode should be “DO NOT SAVE”.

   If opening the file was not succesful, i.e. the file was not found in the operating directory, or it was not valid for some reason, the VI uses its front panel settings to bundle the configuration cluster. By selecting this processing mode, one has expected that the file exists. Therefore a file is created and the configuration is saved regardless of the selected save mode.

   This mode is used in most of our application examples. If there is no configuration file, the front panel “defaults” replace the missing file. Instead of being “factory defaults”, you can and you should edit the front panel settings according to your application. Use CREATE mode with saving for editing.

   The configuration cluster input has no relevance in this mode.

CREATE Regardless of the existence of the configuration file, the configuration cluster is bundled from the front panel settings and it is returned as output of this VI. The configuration can be saved or not.

The difference between this and the READ mode is, that CREATE lets you define the configuration unconditionally, whereas READ tries to use the configuration file at first. Use this mode with saving for editing the configuration initially.

Other

Check the VISA resource using LabView’s tools. Communication with the AVS-47B will fail, if VISA resource points to a wrong serial port (the USB-232 converter is seen as a serial port from LabView). Picobus address must be 1 and Disable AL should be “disabled”.

It is often best to start an application in the local mode (remote = no), then read the current settings from the AVS-47B and set those values for remote control before enabling remote. This way, unexpected jumps in the state of the bridge can be avoided. Many of our examples use this method.

Settings of the AVS-47B

Many of the front panel controls of this VI are enumerated. You can use their numerical values, or you can copy the required controls to your application from the front panel, which is even more convenient. Only the numerical values are shown below.

VISA Resource: This is the name of the Com: port that is used for transactions via Picobus. The port can be either a physical serial port, if the computer has one, or it can be the name of the serial port that the USB-232 converter's driver has given to the virtual serial port.

Picobus address: The hardware address of the AVS-47B. The default is 1 and it should not be altered in normal operation.

Disable AL: The slow A/D converter of the AVS-47B sets the AL signal line to 1 after a conversion has completed and it stays true until disabled. The most recent conversion result is read by keeping AL disabled until a result is needed and then enabling AL. A while loop polls the status of the AL line (RS232 name = DSR) until the ADC sets it. After a short delay, the result is read. See the examples.

Remote: 0 = local, 1 = remote. In local mode, the AVS-47B physical front panel switches are active and one can only read the status and ADC results from the bridge. In remote mode, the front panel switches are disabled. The "REF POT/REF MEM" nor "ΔR/10ΔR" switches cannot be controlled remotely.

Input: 0 = ZERO, 1 = MEAS, 2 = CAL: Position of the input selector.

Channel: 0..7 : Position of the channel selector.

Range: 0..7: 0 = no range, 1...7 = range settings from 2 Ohms to 2 Mohms. The "no range" position can be used for isolating the sensors from the bridge's current source when switching channels and excitation. However, the 2 Mohms range is recommended for this purpose, because the bridge remains then operative.

Excitation: 0..7 : 0 = no excitation, 1..7 = excitation settings from 3μV to 3mV.

Display: The display selector connects the A/D converter to one of the following 8 sources:

0: R = measured resistance (0..1.9999 Volts)

1: dR = resistance deviation from the REFERENCE

2: ADJ REF = voltage output from the front panel REF potentiometer, interpreted as resistance.

3: REFERENCE = voltage output from a 12-bit digital-to-analog converter. This converter takes its value from the ADC result that is currently displayed when the SET REF front panel switch is being operated in the local mode, **OR** when programmed remotely in the remote mode. Meaningful ways to use reference are **a)** to null the deviation by taking reference from the displayed resistance **b)** adjust the potentiometer to a convenient value with the help of the ADJ REF display and taking the reference from it. This last method can be used for extending a range beyond its normal limit (with reduced accuracy).

4: EXC VOLT. This is a very rough indicator of the excitation voltage across the sensor. Its main purpose is to enable estimation of the current lead resistance and it works well on the lowest range and high excitaton. Please refer to the AVS-47B manual for how to estimate lead resistance.

5: HEATER VOLTAGE: The voltage across the heater of a TS-530A temperature control system is measured via the 37-way data cable by the resistance bridge's A/D converter. Small voltages need a lot of amplification before measurement, so the readings must be considered approximate. Please refer to the AVS-47B manual for how to scale the reading.

6: HEATER CURRENT: The current through the heater of a temperature control system. Please refer to AVS-47B manual for how to scale the reading.

7: The voltage of the TS-530A's set point D/A converter. This converter is programmed either by thumbwheels on the physical TS-530A front panel, or remotely in the remote mode. The result can differ from the programmed value by some hundred microvolts.

Digital reference: 0...19999 unsigned integer. The D/A converter in the AVS-47B has only 12 bits, and therefore the digital reference is divided by 5 and rounded to the nearest integer before sending it to the AVS-47B. One digit of the programmed reference corresponds to 100µV, but after this division by 5, the analog reference will change in steps of 500µV.

Settings of the TS-530A

When controlling the TS-530A remotely, the AVS-47B must be in remote mode and the two instruments must be connected together with the 37-way ribbon data cable (TS-530A standard accessory).

Because of its very old design, it is not possible to read settings from the controller (but there are three voltages that can be read, the heater voltage, heater current and set point). If you control the TS-530A remotely, please program ALL its settings every time and then your application knows always, what they are.

Set Point: 10...19999. This is the set point that must be used in AVS-47B + TS-530A temperature control systems. Its resolution is 16 bits (step size 100µV).

Proportional gain: 0..11 corresponding to very approximate gains of 5dB..60dB in 5dB steps.

NOTE: This VI does not include the special PGAIN value of 15, which can be used for short circuiting the proportional amplifier so that its gain is 0. Application example about a temp. control system uses this possibility.

Integrator time constant: 0..11. These values determine relative integrator time constants of PD, 2, 5, 10, 20, 50, 100, 200, 500 and 1000, latched and PD, respectively. PD (proportional plus derivative) means that the analog integrator is short-circuited and does not contribute in the heater output. Latched means that the integrator input is left open and it maintains its state (the analog integrator will exhibit some drift, however).

The integrating effect is largest when the time constant is shortest.

Derivator time constant: 0...7. These values determine relative derivator gains of 0, 2, 5, 10, 20, 50 and 100. The response of this derivator has been modified in order to reduce its gain at unlikely high frequencies and therefore its response is not a simple derivator's response.

Derivating effect is largest when the gain is highest. Derivator can speed up settling, but it also makes the system more sensitive to noise.

Power bias: 0..5 corresponding to additional constant heating power of 0, 20, 40, 60, 80 and 100% of the full power on the currently selected power range.

Power bias may be useful in P and PD control modes where it can reduce steady-state error. In PID mode, it is hardly useful.

Power range: 0, 1..7 corresponding to maximum heating powers of 1uW, 10uW...100uW and 1W. If power range is 0, heating is completely disabled.

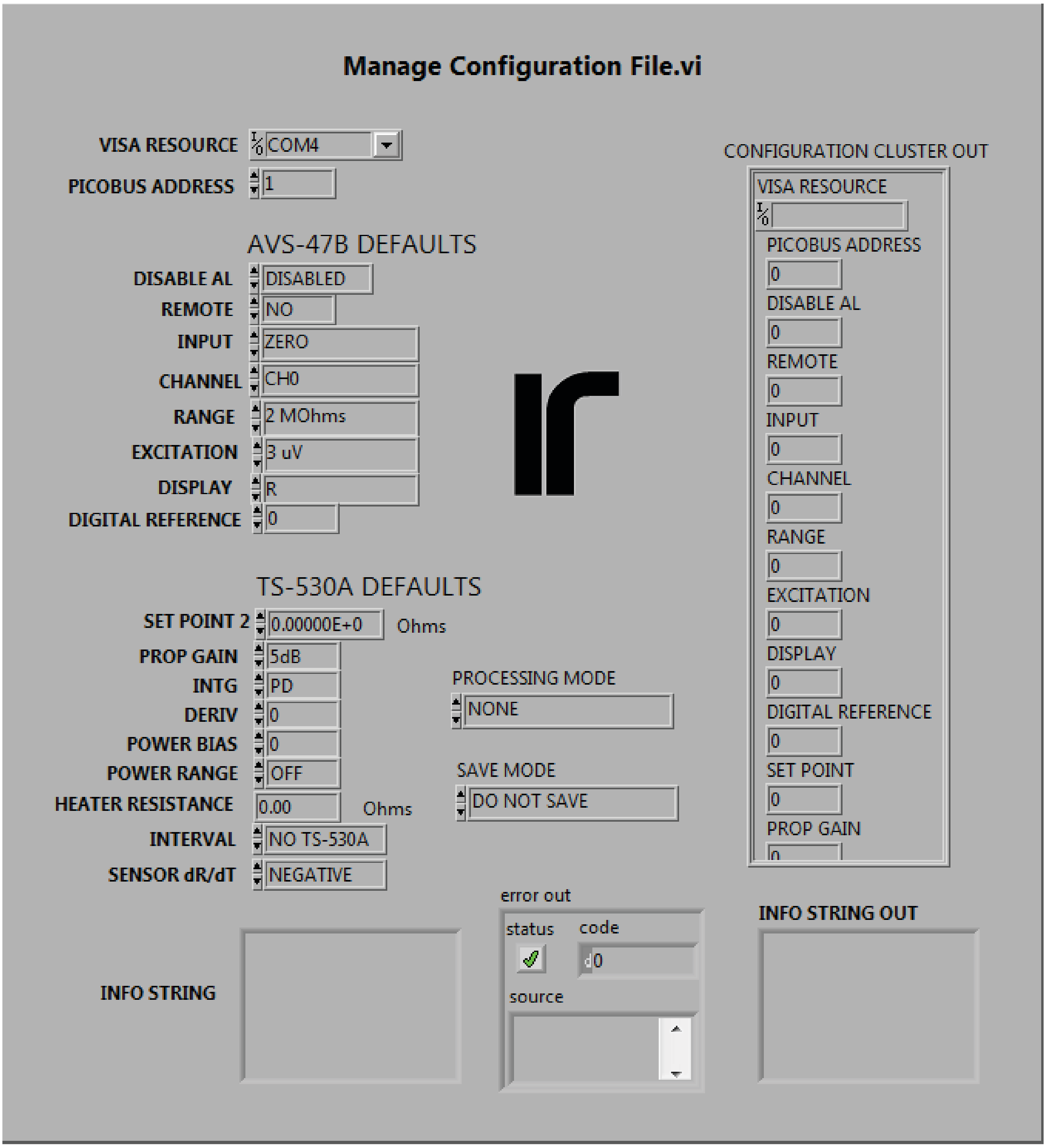
The above heater ranges are correct only for a 100-ohm heater. If resistance is higher, the 1W maximum power cannot be achieved due to the finite output compliance voltage. If resistance is lower, maximum power will be limited by the available output current. Heater voltage times heater current yields the true power in all cases.

Info string: Information string can be used for threading VIs for correct execution order and for reporting about errors or other situations.

Heater resistance: 0..200 ohms. The temperature-control application example calculates correct power ranges also for non-100-ohm heaters. Therefore the heater resistance must be given to the program. Power ranges on the TS-530A physical front panel expect a 100-ohm heater.

Interval: 0, 1...8. This parameter determines, how often the heater conditions (voltage and current) are measured in the temperature control application example.
0 = never (or no TS-530A is connected).
1..8 = 1, 15, 30, 45, 60, 120, 240, or 480 seconds.

Sensor dR/dT: 0 = negative, 1 = positive



Manage Configuration File.vi

VISA RESOURCE: COM4

PICOBUS ADDRESS: 1

AVS-47B DEFAULTS

DISABLE AL: DISABLED

REMOTE: NO

INPUT: ZERO

CHANNEL: CH0

RANGE: 2 MOhms

EXCITATION: 3 uV

DISPLAY: R

DIGITAL REFERENCE: 0

TS-530A DEFAULTS

SET POINT 2: 0.00000E+0 Ohms

PROP GAIN: 5dB

INTG: PD

DERIV: 0

POWER BIAS: 0

POWER RANGE: OFF

HEATER RESISTANCE: 0.00 Ohms

INTERVAL: NO TS-530A

SENSOR dR/dT: NEGATIVE

PROCESSING MODE: NONE

SAVE MODE: DO NOT SAVE

CONFIGURATION CLUSTER OUT

VISA RESOURCE:

PICOBUS ADDRESS: 0

DISABLE AL: 0

REMOTE: 0

INPUT: 0

CHANNEL: 0

RANGE: 0

EXCITATION: 0

DISPLAY: 0

DIGITAL REFERENCE: 0

SET POINT: 0

PROP GAIN: 0

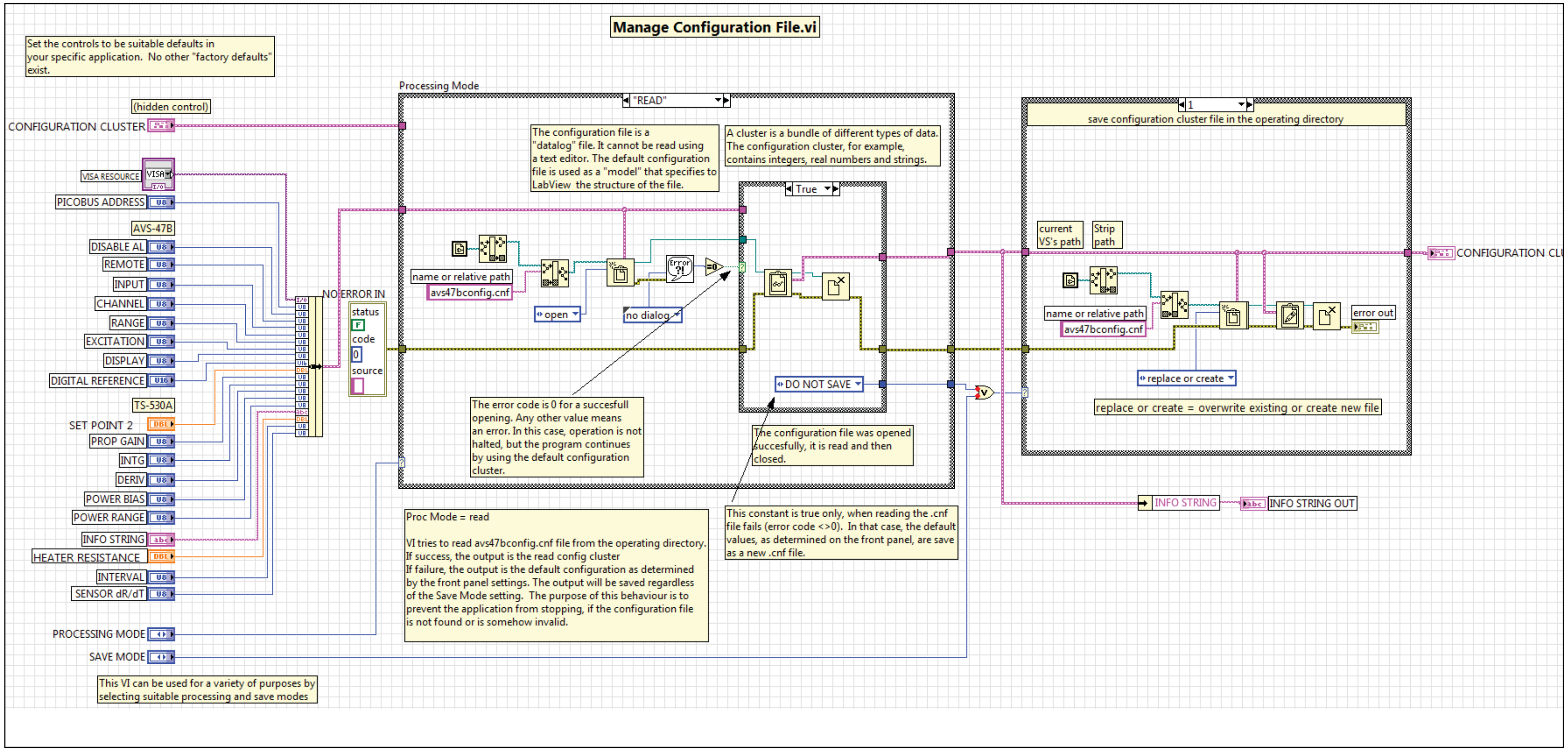
error out

status: ✓ code: 0

source:

INFO STRING

INFO STRING OUT



Manage Configuration File.vi

Set the controls to be suitable defaults in your specific application. No other "factory defaults" exist.

The configuration file is a "datalog" file. It cannot be read using a text editor. The default configuration file is used as a "model" that specifies to LabView the structure of the file.

A cluster is a bundle of different types of data. The configuration cluster, for example, contains integers, real numbers and strings.

The error code is 0 for a succesfull opening. Any other value means an error. In this case, operation is not halted, but the program continues by using the default configuration cluster.

The configuration file was opened succesfully, it is read and then closed.

Proc Mode = read

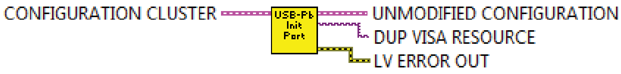
VI tries to read avs47bconfig.cnf file from the operating directory.
If success, the output is the read config cluster
If failure, the output is the default configuration as determined by the front panel settings. The output will be saved regardless of the Save Mode setting. The purpose of this behaviour is to prevent the application from stopping, if the configuration file is not found or is somehow invalid.

This constant is true only, when reading the .cnf file fails (error code <>0). In that case, the default values, as determined on the front panel, are save as a new .cnf file.

save configuration cluster file in the operating directory

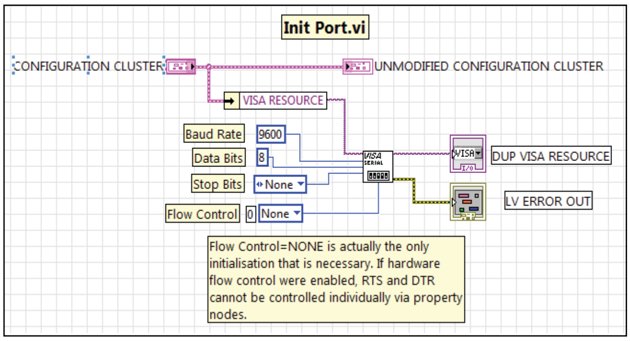
replace or create = overwrite existing or create new file

This VI can be used for a variety of purposes by selecting suitable processing and save modes



InitPort.vi

As its name suggests, this VI initialises a serial port to be used for transactions via Picobus. The port can be a physical serial port, like Com1:, if the computer still has such a port. Or it can be a virtual port, which behaves like a ComX: toward LabView, it makes use of the computer's USB port and with the help of the USB-232 converter, provides a synchronised serial output that can be used for communications with the AVS-47B.

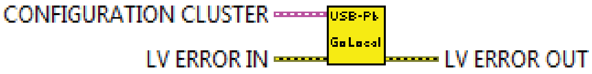


Flow Control=NONE is actually the only initialisation that is necessary. If hardware flow control were enabled, RTS and DTR cannot be controlled individually via property nodes.

The only actually needed initialisation is to disable any hardware handshaking. Then the handshake signals RTS, DTR, CTS and DSR are free to be controlled and read independently via property nodes.

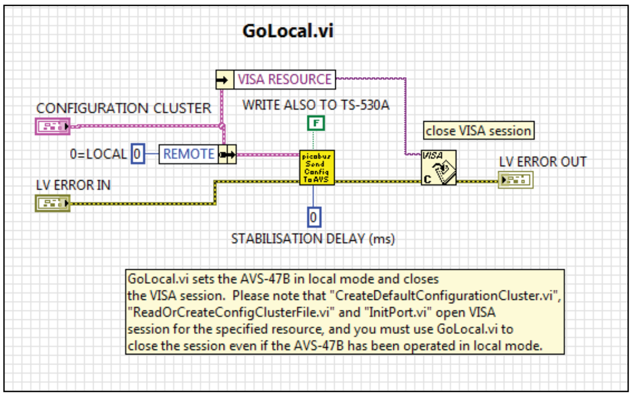
If you do not know, which port has been assigned for the USB-232 converter, launch National Instrument's "Measurement & Automation Explorer - Devices and Interfaces", and check there the port name.

The initialisation opens a VISA session for the assigned port. After you application ends, close the session by running the "**GoLocal.vi**". (During the development phase, you can also or alternatively instruct LabView to automatically close open VISA sessions).



GoLocal.vi

This VI sets the AVS-47B to local mode so, that it can be controlled from its physical front panel. After that, the VISA session is closed and the application ends.



GoLocal.vi sets the AVS-47B in local mode and closes the VISA session. Please note that "CreateDefaultConfigurationCluster.vi", "ReadOrCreateConfigClusterFile.vi" and "InitPort.vi" open VISA session for the specified resource, and you must use GoLocal.vi to close the session even if the AVS-47B has been operated in local mode.

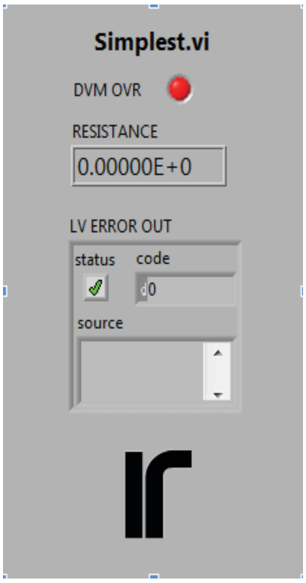
Simplest.vi

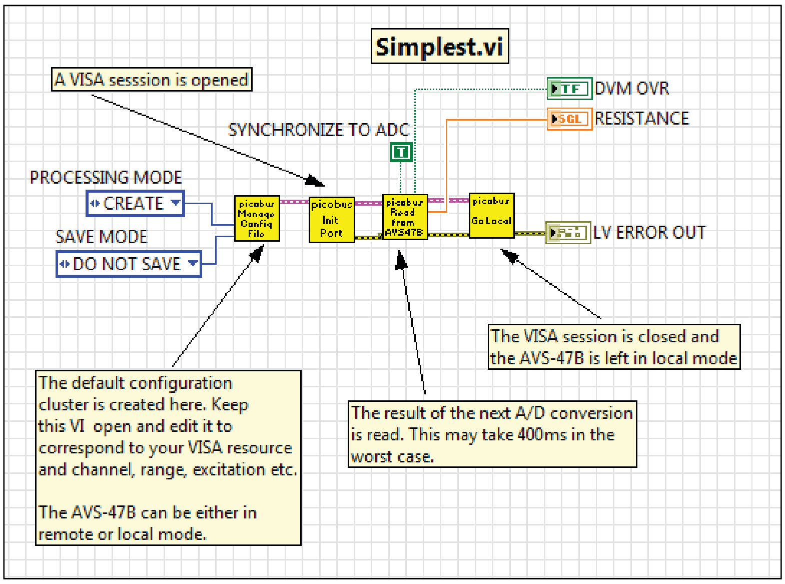
Application Example 1

This is perhaps the simplest way to control the AVS-47B remotely and to get resistance readings from it.

Settings of the AVS-47B are obtained from the **Manage Configuration File.vi.** Open the VI, edit the settings and save the VI.

ReadAvs47b.vi sends the settings to the bridge and reads the result of the next A/D conversion. The reading is scaled using the currently selected range and it is expressed as resistance in ohms. If the measured resistance is too high for the A/D converter (the converter input exceeds 1.9999 Volts), output will be exact zero and the DVM OVR indicator shows overload. Please do use the OVR indicator (or a corresponding indicator in other VIs) for differentiating between overload and true zero result.





A VISA sesssion is opened

The VISA session is closed and the AVS-47B is left in local mode

The default configuration cluster is created here. Keep this VI open and edit it to correspond to your VISA resource and channel, range, excitation etc.

The AVS-47B can be either in remote or local mode.

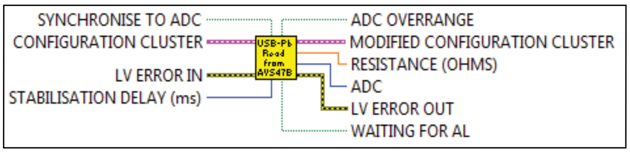
The result of the next A/D conversion is read. This may take 400ms in the worst case.

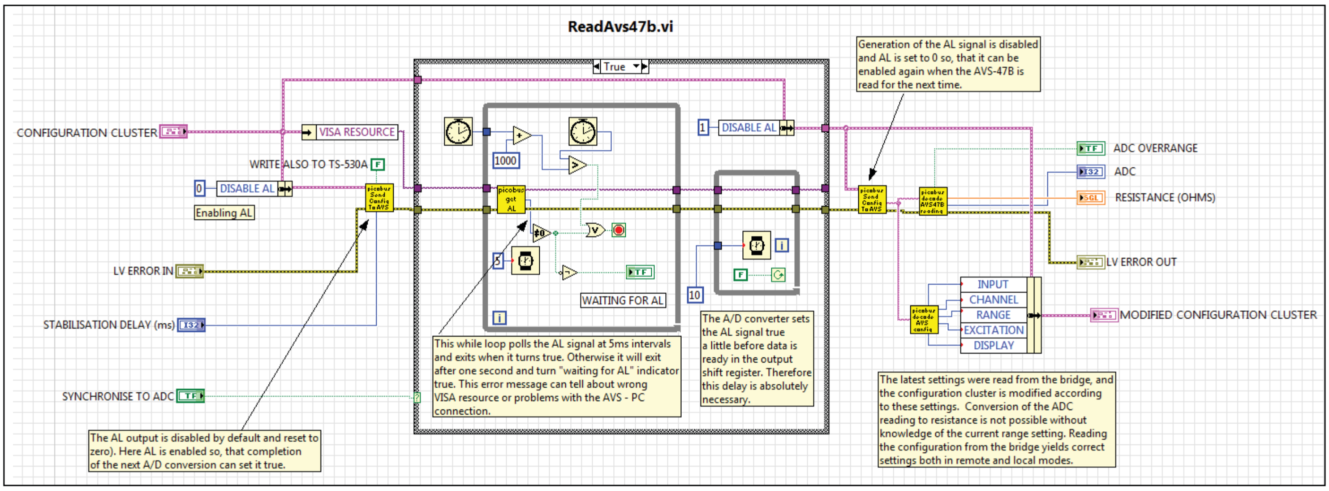
The Manage Configuration File.vi is here used in the CREATE mode. By keeping this VI opened and changing the front panel settings, you can control, how resistance is measured.

ReadAvs47b.vi

You will need this VI in most applications. Configuration cluster is the only mandatory input. By default, synchronisation to the ADC is enabled and stabilisation delay is zero.

So far, generation of the AL (alarm) signal inside the AVS-47B has been disabled. It is now enabled. This is told to the AVS-47B by sending the full configuration to it as a 48-bit long binary string. The





The AL output is disabled by default and reset to zero). Here AL is enabled so, that completion of the next A/D conversion can set it true.

This while loop polls the AL signal at 5ms intervals and exits when it turns true. Otherwise it will exit after one second and turn "waiting for AL" indicator true. This error message can tell about wrong VISA resource or problems with the AVS - PC connection.

The A/D converter sets the AL signal true a little before data is ready in the output shift register. Therefore this delay is absolutely necessary.

Generation of the AL signal is disabled and AL is set to 0 so, that it can be enabled again when the AVS-47B is read for the next time.

The latest settings were read from the bridge, and the configuration cluster is modified according to these settings. Conversion of the ADC reading to resistance is not possible without knowledge of the current range setting. Reading the configuration from the bridge yields correct settings both in remote and local modes.

AVS-47B is programmed by writing always the full configuration, it is not possible to send only parts of it (see “SendConfigurationToAvs47b.vi”). Writing to and reading from the AVS-47B form one single “transaction”. When reading, one also programs the bridge with the current or changed settings and when writing, one always reads the complete last state of the bridge, including the A/D conversion result, that was in the output buffer just before the transaction.

Writing settings to the optional TS-530A temperature controller is disabled, as this VI is for the bridge only. The stabilisation delay (in milliseconds) can be zero, so that there is no delay before proceeding to a while-loop, which polls for ending of an A/D conversion.

If synchronisation is enabled (true case), the state of the AL signal is polled in a while-loop. Polling continues at 5ms intervals until AL becomes true, or until 1 second has passed without success. In both cases, program continues, but if AL was not found, the “waiting for AL” indicator is set true. Because of the peculiar design of the ICL7135 A/D converter, a 10 ms delay is required between detection of AL and reading the data.

“SendConfigurationToAvs47b.vi” is used again, but now for resetting and disabling AL. The transaction returns the conversion result as a 17-bit BCD string together with the full configuration of the bridge, 48 bits altogether. These are decoded by two low-level VIs, “DecodeAvs47bReading.vi” and “DecodeAvs47bConfiguration.vi”.

The reading is decoded to a plain ADC result, which is an integer from -19999 to 19999 regardless of range, and to resistance in ohms, which is a real number. ADC overrange is an important indicator, because the 7135 outputs zero reading when it is overloaded.

If synchronisation is disabled (false case), there is no polling of the AL signal and the program proceeds immediately after the optional stabilisation delay to reading the ADC and the configuration. This mode is OK, if one reads only seldom and there is no danger of reading the same result more than once. It can also be used, if one is only interested in the configuration. The purpose of synchronisation is to prevent multiple readings of a single ADC result.

The stabilisation delay can be zero, if settings like range, channel or excitation remain unchanged, so that the AVS-47B needs no time to settle.

Free Running Avs47b.vi

Application Example 2

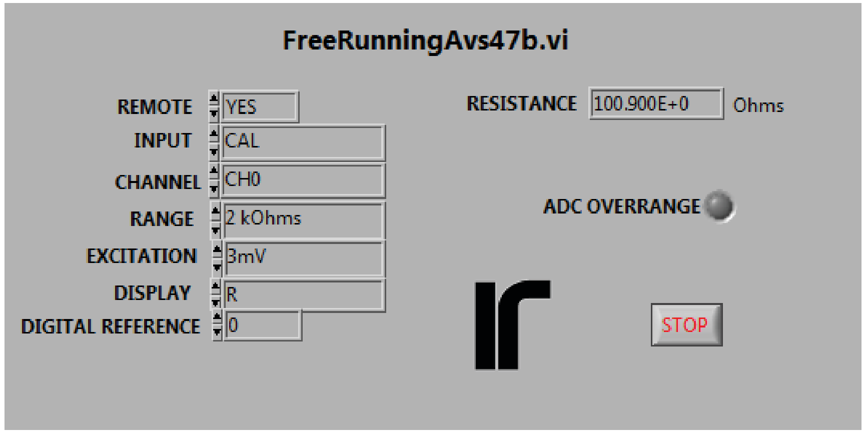
The instrument settings are read from the configuration file by “Manage Configuration File.vi” and VISA session is initialized for the resource specified in the config file. The bridge is assumed to be in local mode.

In addition to receiving, “ReadAvs47b.vi” also writes the settings, that came from the configuration file, to the bridge, but because the bridge was in local, the state of the bridge does not change. The current settings of the bridge are output from the reading VI in the configuration cluster. The AVS-47B front panel settings are then *unbundled by name* and they are written to 6 local variables. This application starts in the local mode for safety. Therefore the control mode is forced to be local.

After these initial operations, the program enters in a while loop. LabView has written the values of the local variables to the front panel controls of the VI so, that the bridge continues running in its current state without any possibly unwanted changes. You are now free to select between remote and local modes and change settings remotely, if needed. Inside the while loop, the controls’ values are *bundled by name* back into the configuration cluster. **This method of using local variables for transferring initial values into a while-loop is used in most of our application examples as well.**

The latest A/D conversion result and the current settings of the bridge are read by “ReadAvs47b.vi”. This information is used differently in local and remote modes:

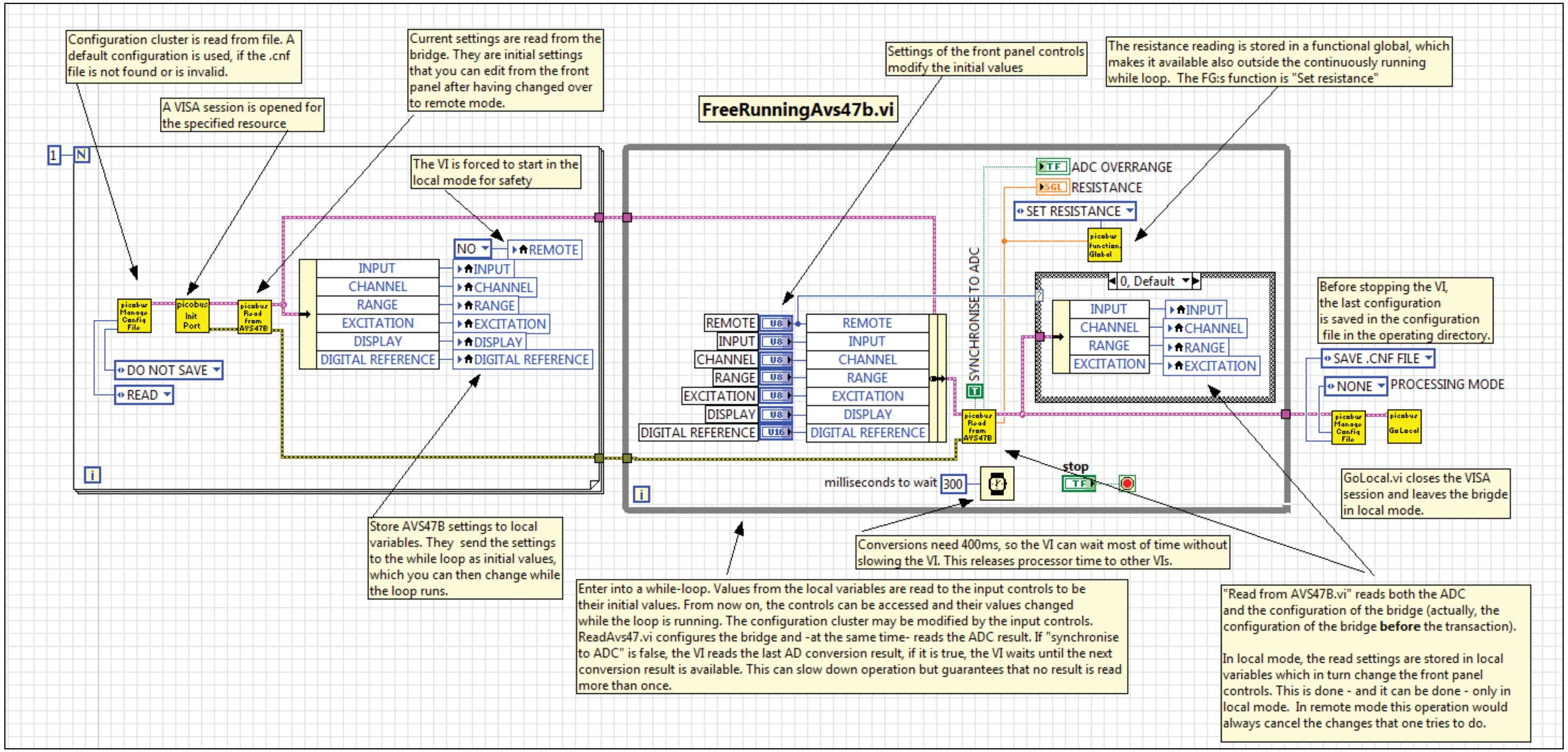
- In local mode (remote=0), input, channel, range and excitation are transferred back to the front panel controls of the VI so, that the controls remain synchronized with the locally operated bridge. This is done by using local variables.



FreeRunningAvs47b.vi

REMOTE: YES
INPUT: CAL
CHANNEL: CH0
RANGE: 2 kOhms
EXCITATION: 3mV
DISPLAY: R
DIGITAL REFERENCE: 0

RESISTANCE: 100.900E+0 Ohms



FreeRunningAvs47b.vi

Configuration cluster is read from file. A default configuration is used, if the .cnf file is not found or is invalid.

A VISA session is opened for the specified resource

Current settings are read from the bridge. They are initial settings that you can edit from the front panel after having changed over to remote mode.

The VI is forced to start in the local mode for safety

Store AVS47B settings to local variables. They send the settings to the while loop as initial values, which you can then change while the loop runs.

Settings of the front panel controls modify the initial values

The resistance reading is stored in a functional global, which makes it available also outside the continuously running while loop. The FG:s function is "Set resistance"

Before stopping the VI, the last configuration is saved in the configuration file in the operating directory.

GoLocal.vi closes the VISA session and leaves the brigde in local mode.

Conversions need 400ms, so the VI can wait most of time without slowing the VI. This releases processor time to other VIs.

Enter into a while-loop. Values from the local variables are read to the input controls to be their initial values. From now on, the controls can be accessed and their values changed while the loop is running. The configuration cluster may be modified by the input controls. ReadAvs47.vi configures the bridge and -at the same time- reads the ADC result. If "synchronise to ADC" is false, the VI reads the last AD conversion result, if it is true, the VI waits until the next conversion result is available. This can slow down operation but guarantees that no result is read more than once.

"Read from AVS47B.vi" reads both the ADC and the configuration of the bridge (actually, the configuration of the bridge **before** the transaction).

In local mode, the read settings are stored in local variables which in turn change the front panel controls. This is done - and it can be done - only in local mode. In remote mode this operation would always cancel the changes that one tries to do.

- In remote mode (remote=1) this is not needed and it cannot be done. Trying to do that would lead to a rival situation.

The integrating dual-slope A/D converter of the AVS-47B is very slow: it needs 400 milliseconds for one conversion. Therefore the loop waits for 300 milliseconds before starting to poll for a new ADC result. This time is available for other simultaneously running LV processes.

In this example, every new resistance value is written into “FunctionalResistanceGlobal.vi”. This VI is a *Functional Global Variable (FG)*, which can be used for transferring values from inside a running loop to VIs outside the loop or to other running loops.

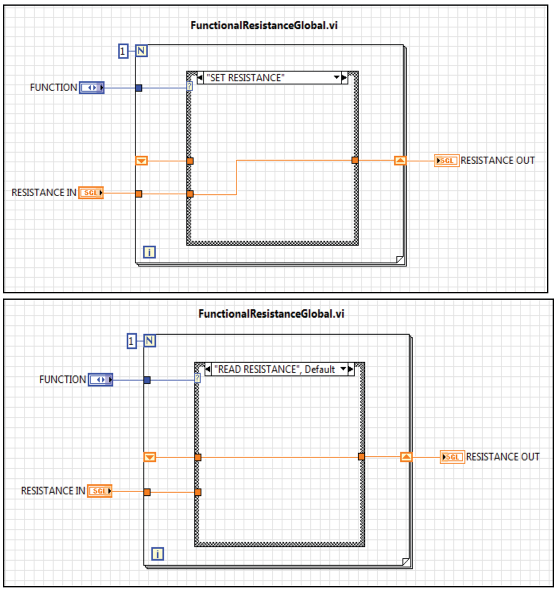
After stopping this example, the last configuration is saved in “avs47bconfig.cnf” and the bridge is set in local mode.

FunctionalResistanceGlobal.vi

Functional Global (FG) is usually a non-reentrant VI. It consists typically of a one-shot FOR or WHILE loop, which encapsulates the code. Data is stored in one or more uninitialized shift registers, which “pass” through a CASE statement. An FG can have several different cases, which determine the action of the VI (like read, write or reset). Because the FG is not re-entrant, its shift registers have one single place in the computer’s memory. This makes the data readable or changeable at all levels of a LabView program.

This FG example is built inside a FOR loop that runs only once. There is one single variable, the resistance, in one shift register and the case structure has two cases, here called “Functions”. If the Function control is “Set Resistance”, the input value is wired to the the register, and the register assumes that value. If the Function is “Read Resistance”, the stored value can be read as many times and in as many places as needed without changing the value.

This kind of an FG can be used for transferring resistance readings, or any other data, between a running loop or VI to other simultaneously running loops or VIs.

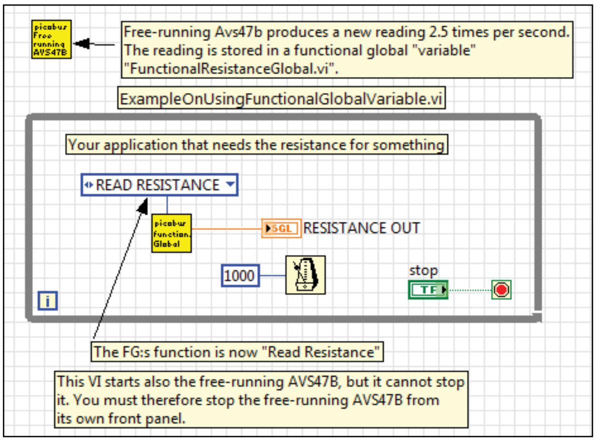


The two cases, Set Resistance and Read Resistance are placed inside a one-shot FOR-loop in this simple Functional Global Variable.

ExampleOnUsingFunctionalGlobalVariable.vi

Application Example 3

Below is a very simple example on how to get readings from the free-running AVS-47B into a running while-loop. Starting the loop starts also the AVS. This small example does not have any mechanism for stopping the free-running AVS, so it must be stopped separately.

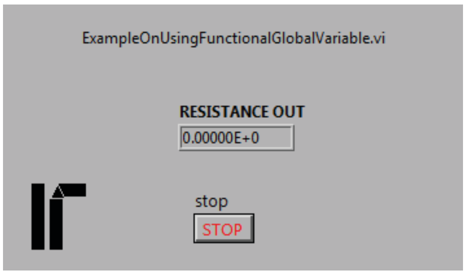

picobus Free running AVS47B
ExampleOnUsingFunctionalGlobalVariable.vi
READ RESISTANCE
picobur Function Global
RESISTANCE OUT
1000
stop

Free-running Avs47b produces a new reading 2.5 times per second. The reading is stored in a functional global "variable" "FunctionalResistanceGlobal.vi".

Your application that needs the resistance for something

The FG:s function is now "Read Resistance"

This VI starts also the free-running AVS47B, but it cannot stop it. You must therefore stop the free-running AVS47B from its own front panel.

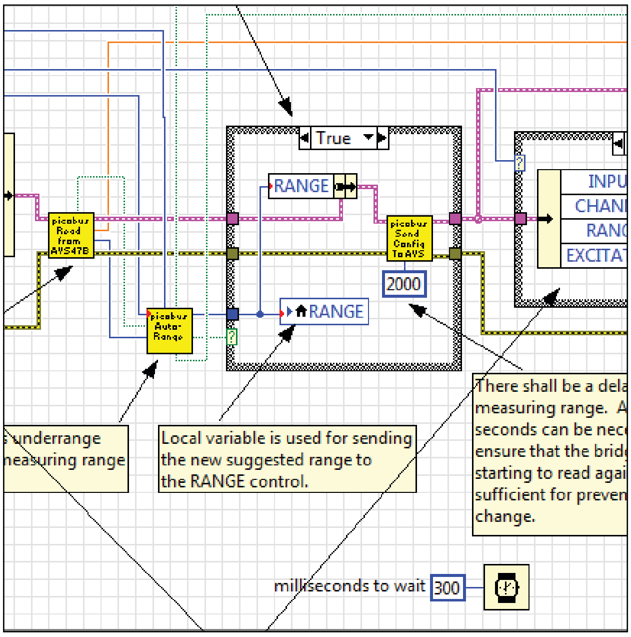


Free Running Avs47b With Autoranging.vi

Application Example 4

This is very similar to the previous FreeRunningAvs47b.vi. Now **Autoranging.vi** checks each new ADC reading and if the reading exceeds 19000 or falls below 1800, it suggests a new higher or lower measuring range and sets the “range has changed” boolean output true. This happens only in autorange mode, in manual range mode a new range is not suggested. There are also some other criteria for how to change the range. They are explained below.

The suggested new range is bundled into the configuration cluster and the new configuration is sent to the AVS-47B. The bridge changes to the new suggested range, but only in remote mode. If the bridge is in local, the old configuration is read from bridge and it is fed back to it so, that nothing happens in the local mode.



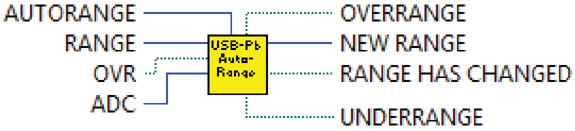
underrange
measuring range

Local variable is used for sending the new suggested range to the RANGE control.

There shall be a dela
measuring range. A
seconds can be nec
ensure that the brid
starting to read agai
sufficient for preven
change.

Free Running Avs47b With Autoranging.vi
Show is only the part that is different from “Free Running Avs47b.vi”

Autoranging.vi



“Autoranging.vi” tests the incoming ADC reading in three ways (ADC>19000, ADC<1800 and (ADC=0 and OVerLoad)). The results of these comparisons are grouped to a binary number and then converted to a decimal number. The results have the following meanings:

0: reading 1800<ADC<19000 (normal)
1: reading >19000 but <19999 (OverRange)
2: reading <1800 (UnderRange)
3: not a possible value
4: not a possible value
5: not a possible value
6: reading =0 and OVL=true (Input > 1.9999V)
7: not a possible value

The outermost CASE can have two values: Autoranging is enabled or autoranging is disabled. The two inner CASE statements determine the action and output based on the test result and on the currently selected measuring range of the AVS-47B.

<u>Autoranging enabled</u>

test=0: -suggest current range (normal operation)
test=1 or 6: -suggest higher range
-set overrange indicator true
OR
IF range is already 2 Mohms
-suggest current range
-set overrange indicator true
Test=2: -suggest lower range
-set underrange indicator true
OR
IF range is already NoRange or 2 Ohms
-suggest current range
-keep underrange indicator false

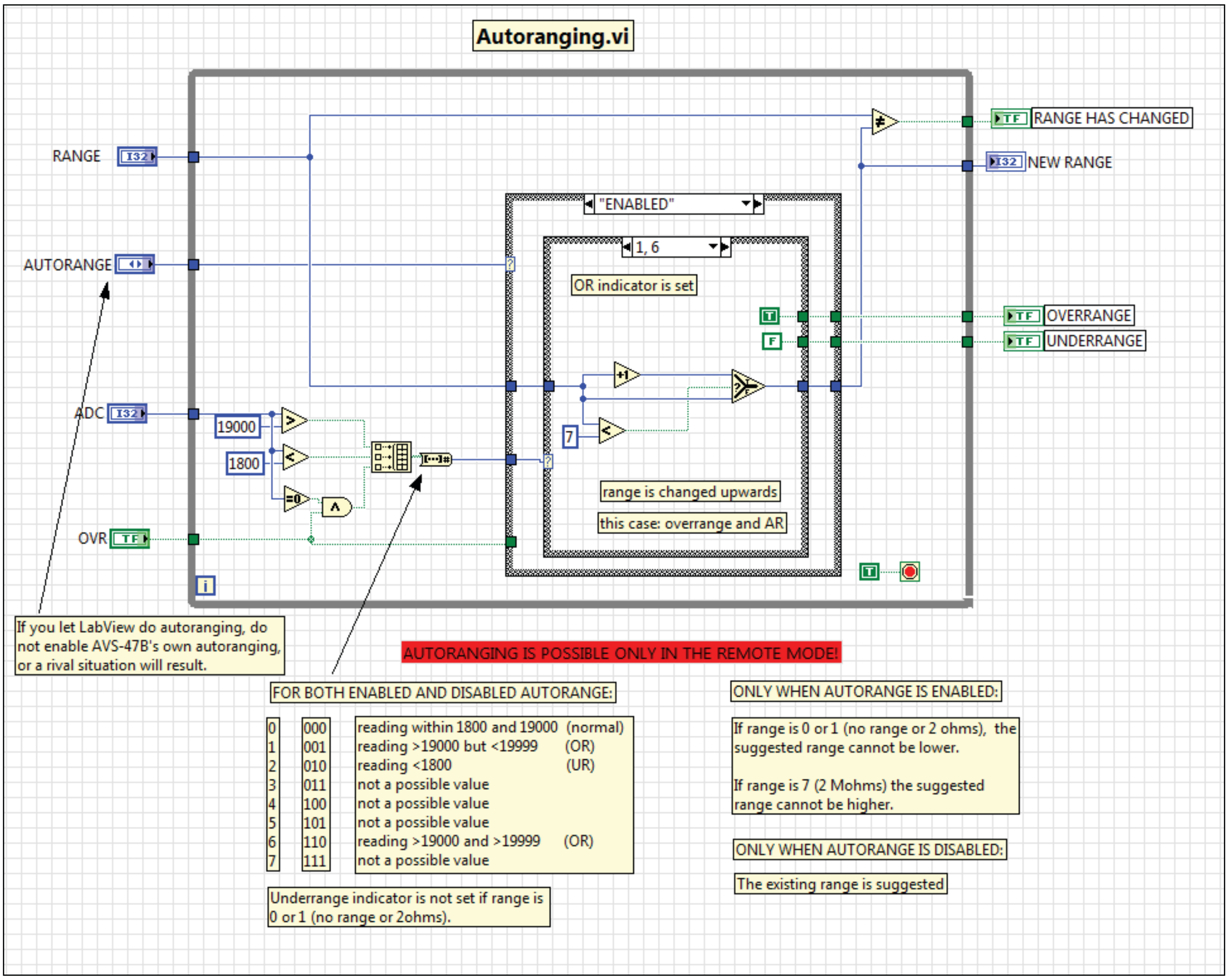
<u>Autoranging disabled</u>

Test=0: - Overrange and underrange indicators false
Test=1 or 6: - overrange indicator true
Test=2: - underrange indicator true
OR
IF range is already NoRange or 2 Ohms
- underrange indicator false

In all above cases, suggest the current range. The underrange indicator behaves differently on the 2

Ohms range, because getting values <1800 on the lowest range must be considered normal.

The “Range has changed” indicator is used for reconfiguring the AVS-47B when it is in remote mode.



OR indicator is set

range is changed upwards

this case: overrange and AR

If you let LabView do autoranging, do not enable AVS-47B's own autoranging, or a rival situation will result.

AUTORANGING IS POSSIBLE ONLY IN THE REMOTE MODE!

FOR BOTH ENABLED AND DISABLED AUTORANGE:

0	000	reading within 1800 and 19000	(normal)
1	001	reading >19000 but <19999	(OR)
2	010	reading <1800	(UR)
3	011	not a possible value	
4	100	not a possible value	
5	101	not a possible value	
6	110	reading >19000 and >19999	(OR)
7	111	not a possible value	

Underrange indicator is not set if range is 0 or 1 (no range or 2ohms).

ONLY WHEN AUTORANGE IS ENABLED:

If range is 0 or 1 (no range or 2 ohms), the suggested range cannot be lower.

If range is 7 (2 Mohms) the suggested range cannot be higher.

ONLY WHEN AUTORANGE IS DISABLED:

The existing range is suggested

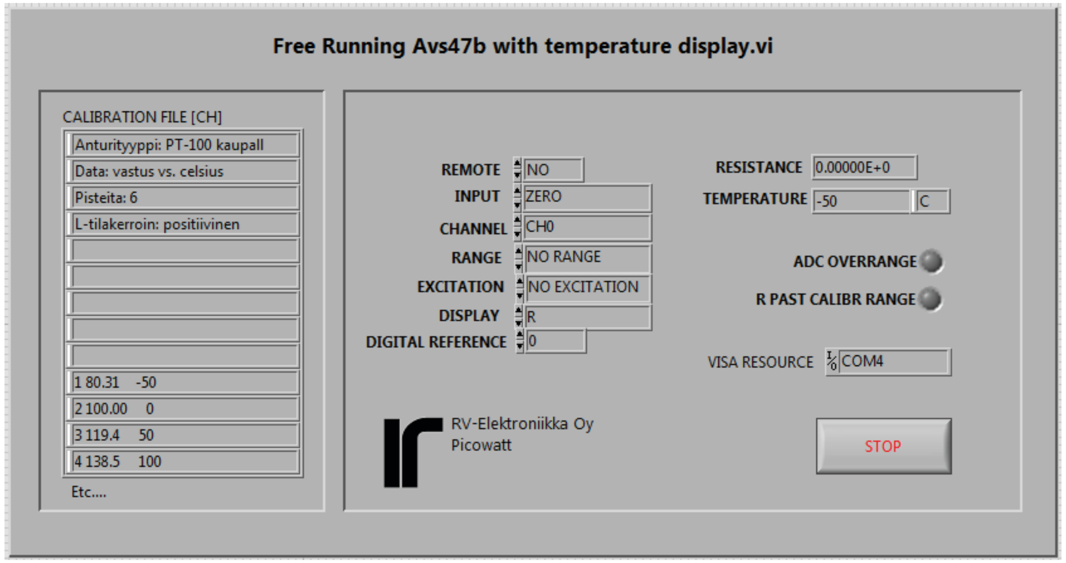
Free Running AVS47b with Temperature Display.vi

Application Example 5

This example is again very similar to the “Free Running Avs47b.vi”. Two new VIs have been added: “**Manage RT Files.vi**” and “**Interpolate R to T.vi**”. The first one starts by reading an RT information file **“avs47b_rt_file_info.cnf”** and then the actual calibration text files that were listed in the information file. The information file tells, which calibration file shall be used for which channel, and what kind of file it will be (there are only few options). The beginning of the calibration file (if it exists) for the currently selected channel is shown on the front panel.

As soon as the sensor’s resistance is known, the interpolating VI calculates the corresponding temperature. The calibration works only in the range between and including the first and last breakpoint. If the argument is outside these limits, the interpolating VI returns the value of the nearest breakpoint and sets the “R past calibration range” indicator.

Temperature unit can be K or C. The number of breakpoints is not limited but the resistances must be in ascending order. The information file is a binary datalog file, whereas the calibration files are simple text files. They can be accessed using a text editor.



Free Running Avs47b with temperature display.vi

CALIBRATION FILE [CH]

Anturityyppi: PT-100 kaupall
Data: vastus vs. celsius
Pisteita: 6
L-tilakerroin: positiivinen

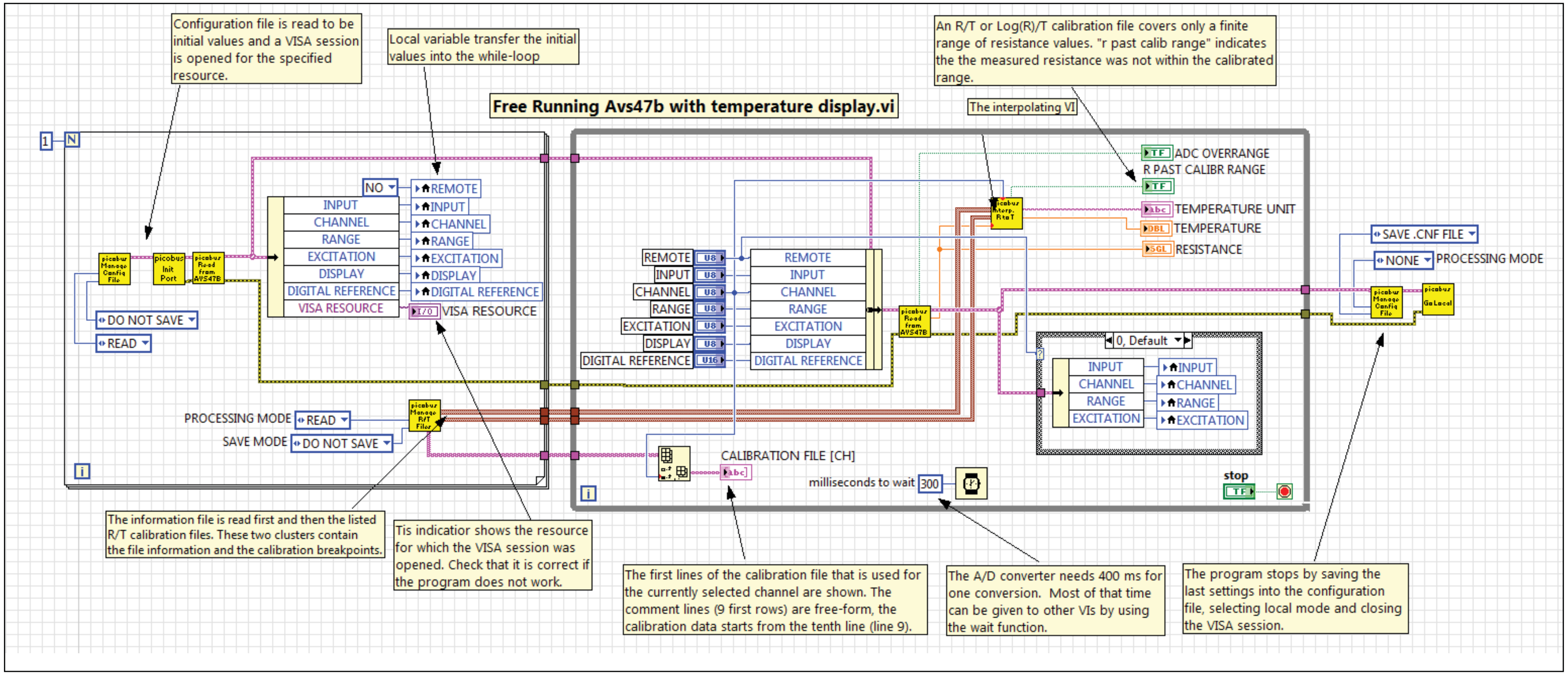
1 80.31 -50
2 100.00 0
3 119.4 50
4 138.5 100
Etc....

REMOTE: NO
INPUT: ZERO
CHANNEL: CH0
RANGE: NO RANGE
EXCITATION: NO EXCITATION
DISPLAY: R
DIGITAL REFERENCE: 0

RESISTANCE: 0.00000E+0
TEMPERATURE: -50 C

ADC OVERRANGE
R PAST CALIBR RANGE

VISA RESOURCE: COM4



Free Running Avs47b with temperature display.vi

Configuration file is read to be initial values and a VISA session is opened for the specified resource.

Local variable transfer the initial values into the while-loop

An R/T or Log(R)/T calibration file covers only a finite range of resistance values. "r past calib range" indicates the the measured resistance was not within the calibrated range.

The interpolating VI

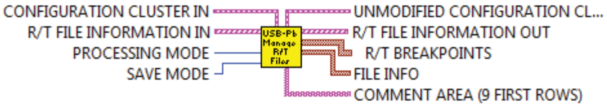
The information file is read first and then the listed R/T calibration files. These two clusters contain the file information and the calibration breakpoints.

Tis indicatior shows the resource for which the VISA session was opened. Check that it is correct if the program does not work.

The first lines of the calibration file that is used for the currently selected channel are shown. The comment lines (9 first rows) are free-form, the calibration data starts from the tenth line (line 9).

The A/D converter needs 400 ms for one conversion. Most of that time can be given to other VIs by using the wait function.

The program stops by saving the last settings into the configuration file, selecting local mode and closing the VISA session.



Manage RT Files.vi

This VI is used for handling the information file that specifies the available resistance-to-temperature calibration files. It is also used for reading the calibrations files and for preparing them for interpolation.

OPERATING INSTRUCTIONS

This VI has two save modes, “save .cnf file” and “do not save”. Depending on the selected save mode, the RT information is saved in “avs47b_rt_file_info.cnf” in the operating directory, or not saved. The calibration files themselves are never saved and they must be changed using a text editor or a spreadsheet program. This VI has four processing modes:

1. NONE The VI expects to get a valid RT information cluster as input. It reads all listed calibration files into memory and forms the calibration array of clusters, which is then output. The VI does not pause. RT information can be saved or not saved.

2. READ The VI tries to open “avs47b_rt_file_info.cnf” in the operating directory. If success, the file is read to RT file information cluster.

   If opening fails, the information cluster is bundled from default data, which you must then edit by running this VI again as such in the “read and edit” processing mode.

   Using information in the RT info file, all listed calibration files are read and they form the RT calibration array of clusters. The VI does not pause. Saving RT information without changes is possible but usually not meaningful. Set save mode to “do not save”.

3. READ AND EDIT Just like above, the VI tries to read the file and if not possible, uses default data for all channels.

   Then the VI pauses and lets you edit the information. Detailed instructions follow.

   You can either discard or accept edits, after which the program proceeds. The RT information is saved if save mode is “save .cnf file”.

   Then the available calibration text files are read and an array of clusters is made out of them.

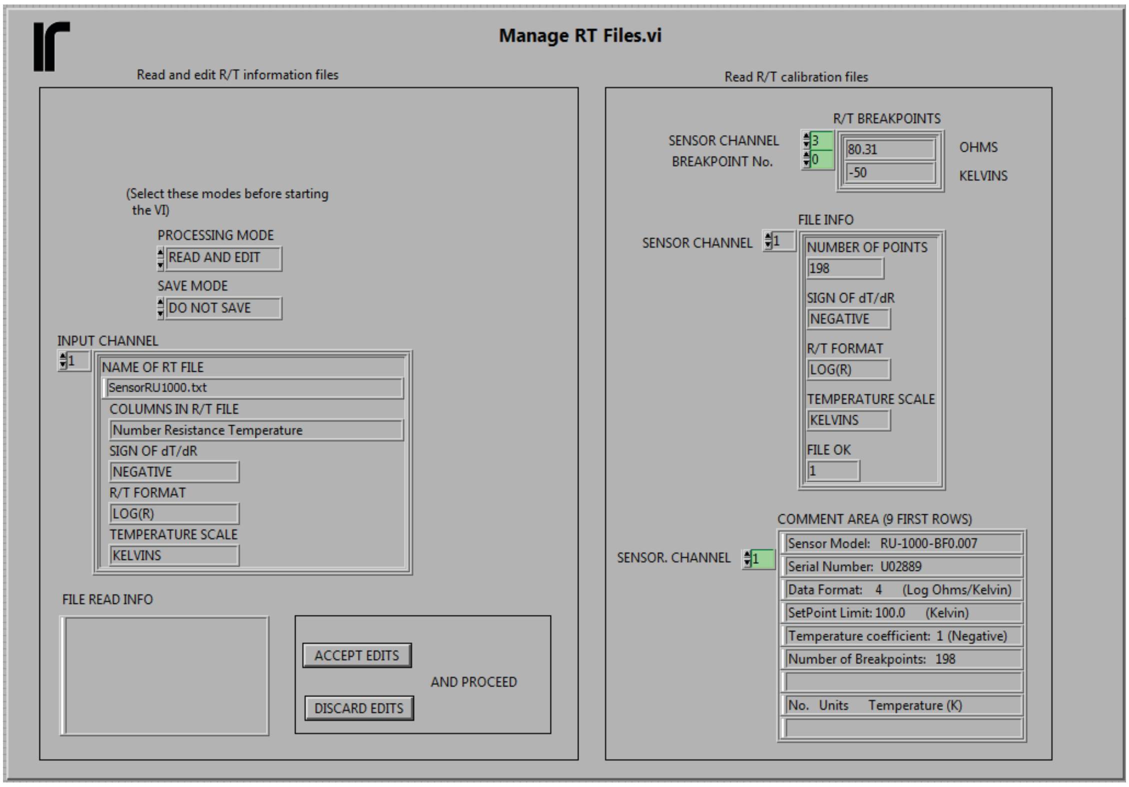
   Because the VI pauses and waits for keyboard input, this mode should be used only when the VI is run as such, not as a sub-vi within an application.

4. DEFAULTS The VI assembles the RT information cluster unconditionally from the default data, which can then be saved or not saved. The default data requires editing before it is useful.

EDITING THE RT INFORMATION

1. Select channel
2. Filename: Type the name of the calibration file for this channel, if one exists. Otherwise, leave the field blank or give a non-existing file name. Name and file extension are free, but all calibration files must reside in the operating directory.
3. Give the number of columns in the calibration file. The breakpoints in the file may have been expressed as *resistance and temperature* (2 columns), or as *breakpoint's ordinal number, resistance and temperature* (3 columns).
4. Sign of dR/dT. Positive or negative.
5. R/T format. The resistance of a reasonably linear sensor is often expressed in ohms (R). Sensors for a wide range or very nonlinear sensors may have been calibrated using log(R).
6. Temperature scale for cryogenic sensors is usually Kelvin. Room temperature sensors are often calibrated in terms of Celsius (Centigrade). Calibration in Fahrenheit is not supported.

Accept and save the edited RT information.



Manage RT Files.vi

Read and edit R/T information files

(Select these modes before starting the VI)

PROCESSING MODE: READ AND EDIT

SAVE MODE: DO NOT SAVE

INPUT CHANNEL: 1

NAME OF RT FILE: SensorRU1000.txt

COLUMNS IN R/T FILE: Number Resistance Temperature

SIGN OF dT/dR: NEGATIVE

R/T FORMAT: LOG(R)

TEMPERATURE SCALE: KELVINS

FILE READ INFO

Read R/T calibration files

SENSOR CHANNEL: 3
BREAKPOINT No.: 0

R/T BREAKPOINTS: 80.31 OHMS, -50 KELVINS

SENSOR CHANNEL: 1

FILE INFO
NUMBER OF POINTS: 198
SIGN OF dT/dR: NEGATIVE
R/T FORMAT: LOG(R)
TEMPERATURE SCALE: KELVINS
FILE OK: 1

SENSOR. CHANNEL: 1

COMMENT AREA (9 FIRST ROWS)

```
Sensor Model:  RU-1000-BF0.007
Serial Number:  U02889
Data Format:    4      (Log Ohms/Kelvin)
SetPoint Limit: 100.0     (Kelvin)
Temperature coefficient:  1 (Negative)
Number of Breakpoints:  198

No.   Units     Temperature (K)

```

Checking the calibration files

After having accepted the edits, the right half of the front panel offers some means to check the results.

In the middle field, select a channel number. The number of breakpoints, dR/dT sign, R/T format and temperature scale are shown if everything was OK.

In the bottom field, select a channel number. Nine first rows of the calibration file are shown. These rows are reserved for free-form comments that you can write to the calibration file, which is a plain text file. The tenth row (row No. 9, counting starts from 0) must be the lowest breakpoint.

In the top field, you can inspect each individual breakpoint by selecting the channel number and the breakpoint's ordinal number.

If a channel does not have a calibration file, the FILE OK =0, comment area is blank and number of breakpoints is 0.

Structure of a calibration file

Calibration files are simple text files, where

- Rows from 0 to 8 are reserved for free-form comments and information about the file
- The tenth row (row No. 9) is the lowest breakpoint
- Each breakpoint consists of either a) ordinal number, resistance and temperature or b) resistance and temperature (3 or 2 columns)
- Items on a row are separated by one or more spaces or tab characters.
- Number of breakpoints is not limited
- The *resistances* must be in ascending order and they must grow monotonically.

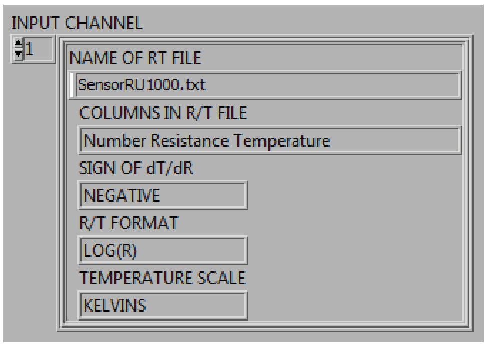
The calibration *array of clusters* is constructed as follows:

- Each breakpoint make a cluster of two elements, R and T.
- One calibration file makes up a 1-dimensional array of N elements, where each element is an (R,T) cluster and N is the number of breakpoints.
- Eight such arrays form a 2-dimensional array (channel No, breakpoint No) of (R,T) clusters.
- For nonexistent files, breakpoints are (0,0).

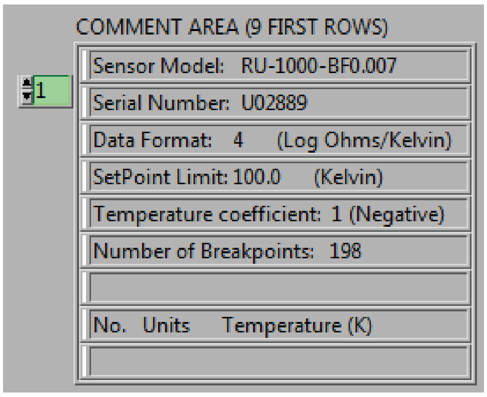
CALIBRATION FILE EXAMPLE 1

```
Sensor Model:   RU-1000-BF0.007
Serial Number:  U02889
Data Format: 4       (Log Ohms/Kelvin)
SetPoint Limit: 100.0    (Kelvin)
Temperature coefficient:  1 (Negative)
Number of Breakpoints:   198

No.    Units      Temperature (K)

1      3.02771     102
2      3.02845     99
3      3.2913      96.5
4      3.02985     94
5      3.03062     91.5
6      3.03144     89
7      3.03232     86.5
8      3.03325     84
9      3.03424     81.5
ETC...
```

AS SHOWN BY “MANAGE RT FILES.VI”



INPUT CHANNEL
1
NAME OF RT FILE
SensorRU1000.txt
COLUMNS IN R/T FILE
Number Resistance Temperature
SIGN OF dT/dR
NEGATIVE
R/T FORMAT
LOG(R)
TEMPERATURE SCALE
KELVINS

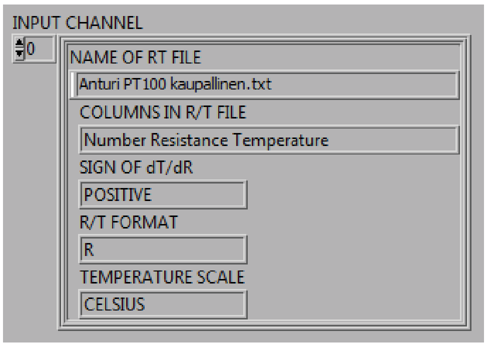


COMMENT AREA (9 FIRST ROWS)
1
Sensor Model: RU-1000-BF0.007
Serial Number: U02889
Data Format: 4   (Log Ohms/Kelvin)
SetPoint Limit: 100.0   (Kelvin)
Temperature coefficient: 1 (Negative)
Number of Breakpoints: 198

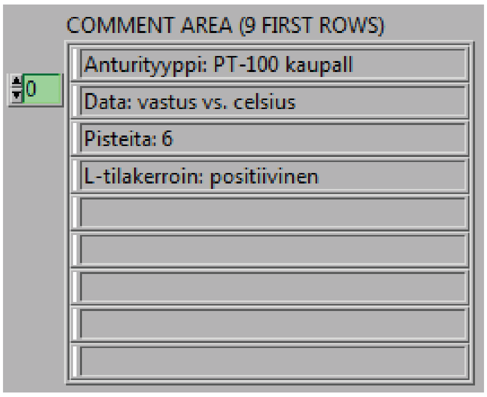
No.   Units   Temperature (K)

CALIBRATION FILE EXAMPLE 2

```
Anturityyppi: PT-100 kaupall
Data: vastus vs. celsius
Pisteita: 6
L-tilakerroin: positiivinen





1      80.31     -50
2      100.00      0
3      119.4      50
4      138.5      100
5      157.31     150
6      175.84      200
```



INPUT CHANNEL
0
NAME OF RT FILE
Anturi PT100 kaupallinen.txt
COLUMNS IN R/T FILE
Number Resistance Temperature
SIGN OF dT/dR
POSITIVE
R/T FORMAT
R
TEMPERATURE SCALE
CELSIUS



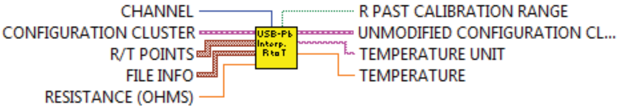
COMMENT AREA (9 FIRST ROWS)
0
Anturityyppi: PT-100 kaupall
Data: vastus vs. celsius
Pisteita: 6
L-tilakerroin: positiivinen

Interpolate R to T.vi

This VI converts resistance readings to temperature. It is based on LabView's function "Interpolate 1D array", which needs two inputs: a one-dimensional array of (R_i,T_i) points - breakpoints - which are (R_i,T_i) clusters, and a variable R, which must be within the lowest and highest values of R_i. If R is not between or including the first and last breakpoint, the function returns the T value of the nearest breakpoint. There is no error indicator available.

This VI needs the following inputs:

- Channel number
- Information about the calibration files. This information cluster is also obtained from **Manage RT Files.vi.**
- The complete 2-dimensional calibration array of clusters. For each input channel, there is a 1-dimensional array of (R,T) clusters, and the arrays of all eight input channels are grouped together to form a 2-dimensional array of breakpoints P(i,CH), where i is the ordinal number of the point. For channels that do not have temperature calibration, the breakpoints are (0,0). This calibration array is received from **Manage RT Files.vi**.
- The resistance value in ohms.



- Configuration cluster can be used for threading. It is not a necessary input.

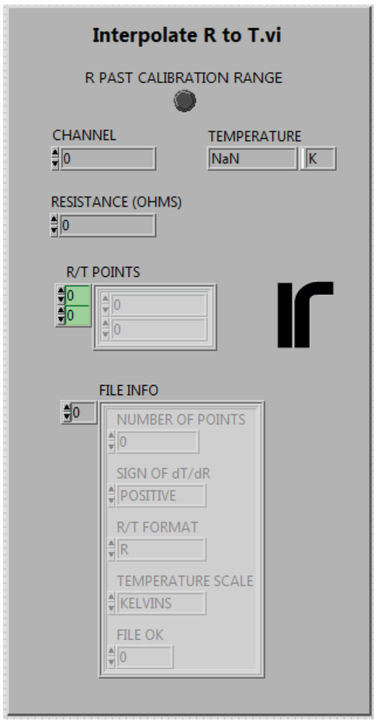
See the application examples for how to use this VI.

INTERNALS

One 1-dimensional array is extracted from the 2-dimensional calibration array by using channel number as index. The 1D array and the resistance to be converted are taken to the linear interpolator, which returns temperature. Information about this channel's calibration file is extracted from the File Info array. Depending on whether calibration was made using R or log(R), one of the two cases is executed.

1) R: The lowest and highest R-values of the breakpoints are determined (first and last values). The argument is compared against these for checking that min≤R≤max. If not, a warning flag is set.

2) log(R): A 10-base logarithm is computed of the argument in order to make it compatible with the calibration data. The logarithm is the checked against limits as above.


Interpolate R to T.vi

R PAST CALIBRATION RANGE ●

CHANNEL 0

TEMPERATURE NaN K

RESISTANCE (OHMS) 0

R/T POINTS 0 0 | 0 0

FILE INFO 0

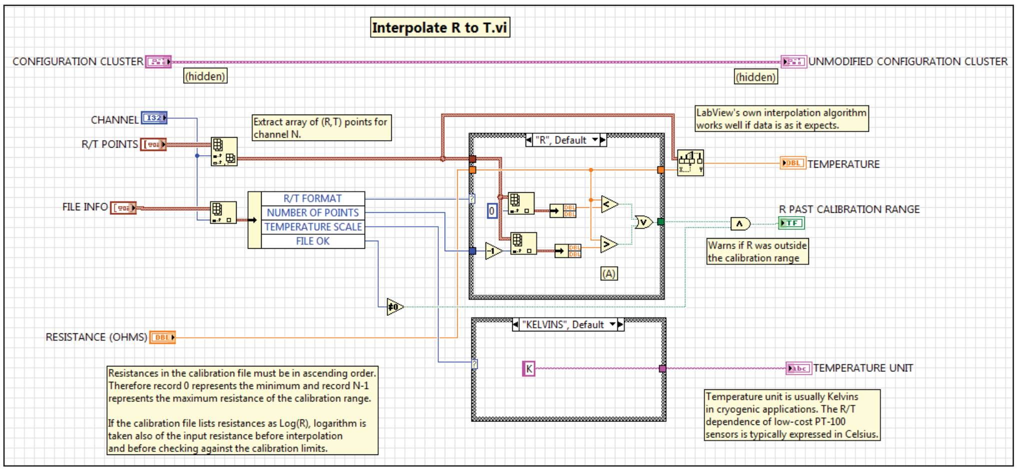
NUMBER OF POINTS 0

SIGN OF dT/dR POSITIVE

R/T FORMAT R

TEMPERATURE SCALE KELVINS

FILE OK 0



Extract array of (R,T) points for channel N.

LabView's own interpolation algorithm works well if data is as it expects.

Warns if R was outside the calibration range

Resistances in the calibration file must be in ascending order. Therefore record 0 represents the minimum and record N-1 represents the maximum resistance of the calibration range.

If the calibration file lists resistances as Log(R), logarithm is taken also of the input resistance before interpolation and before checking against the calibration limits.

Temperature unit is usually Kelvins in cryogenic applications. The R/T dependence of low-cost PT-100 sensors is typically expressed in Celsius.

Test AVS47B Remote Control. vi

Application example 6

This VI is used for production testing of the bridges. All status registers are programmed in turn. In the end, the digital reference is given a set of values. The status LEDs on the physical AVS-47B front panel show the progress of the VI.

In this example, the configuration file is not touched. Instead, a default configuration cluster is created and it is modified by using “*bundle by name*”. The actual program is a state machine consisting of a CASE structure inside a WHILE loop.

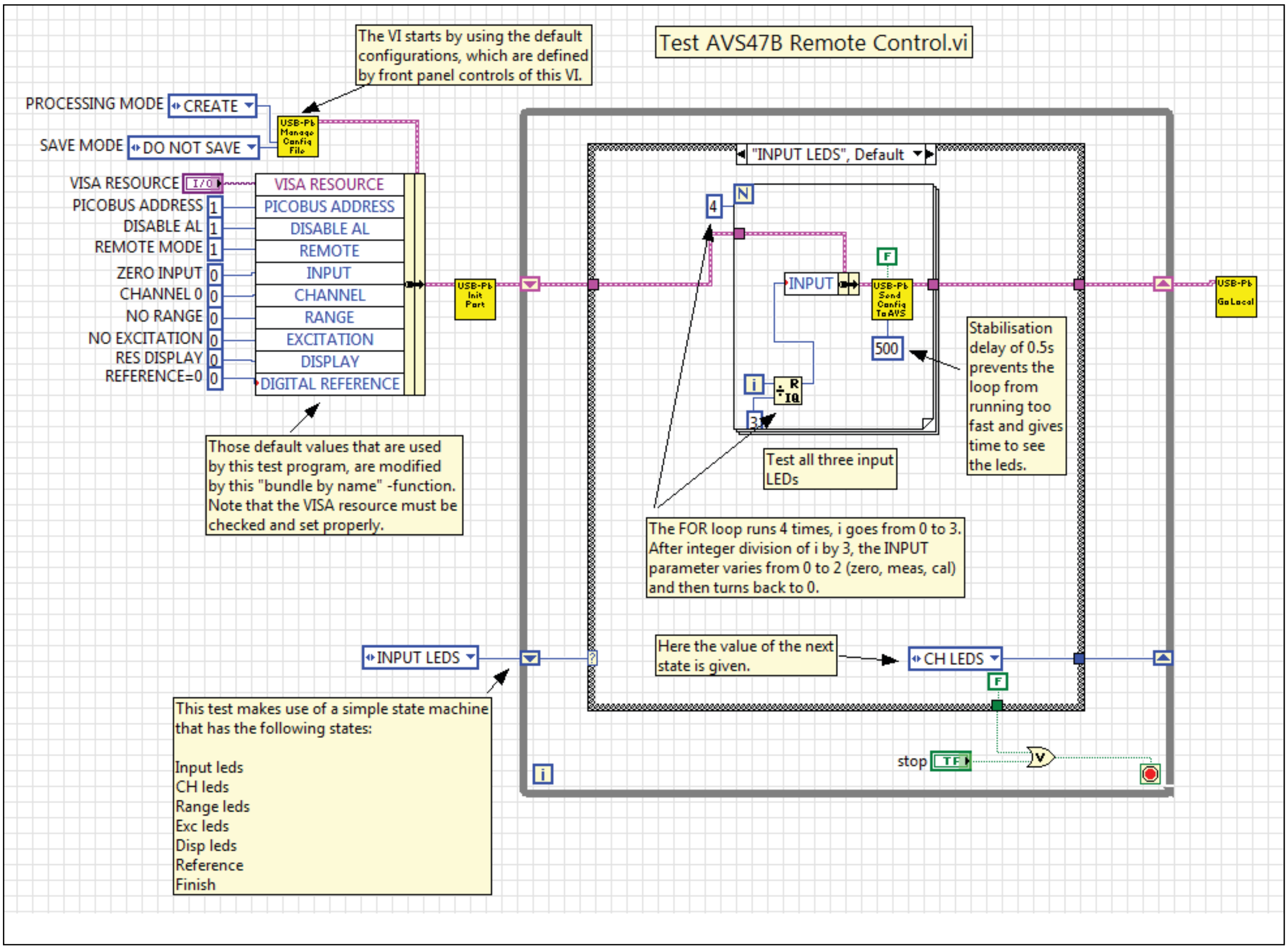
The while-loop has two shift registers: one for the configuration cluster and one for the operation to be performed. When the VI starts, the INPUT LEDs are checked first. Inside the case statement, the input is programmed for values 0, 1, 2 and 0 again, successively, having a delay of half a second between values so, that one can see the LEDs lighting.

As soon as the three LEDs have been checked, the “operation” shift register is given value “CH LEDS” (numeric value = 1). The case-structure is run again, but this time the case is CH LEDs. All eight channel indicators are tested. After all status LEDs have been tested, the digital reference is checked.

The digital reference gets a set of values from 0 to 19000 while the A/D converter of the resistance bridge is connected to show the reference voltage. Look at the AVS-47B front panel display. The reference voltage has only 12-bit resolution, so deviations of at least +/- 5 digits from ideal values must be expected.

This VI is an example of a very simple state machine application. The complications are buried in the “SendConfigurationToAvs47b.vi” , which will be described next.

Before running this VI, check that the VISA resource is correct.



The VI starts by using the default configurations, which are defined by front panel controls of this VI.

Those default values that are used by this test program, are modified by this "bundle by name" -function. Note that the VISA resource must be checked and set properly.

Test all three input LEDs

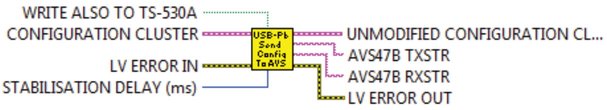
Stabilisation delay of 0.5s prevents the loop from running too fast and gives time to see the leds.

The FOR loop runs 4 times, i goes from 0 to 3. After integer division of i by 3, the INPUT parameter varies from 0 to 2 (zero, meas, cal) and then turns back to 0.

Here the value of the next state is given.

This test makes use of a simple state machine that has the following states:

Input leds
CH leds
Range leds
Exc leds
Disp leds
Reference
Finish



SendConfigurationToAvs47b.vi

This VI is the heart of this USB-Picobus LV package. It is used for all communications between the computer and AVS-47B (and the optional TS-530A). It contains low-level information about the various registers inside the resistance bridge and it formats inputs to be suitable for those registers.

The description about the internals of this VI can well be skipped - it is included for those who are interested.

OPERATING INSTRUCTIONS

Connect the **Configuration Cluster** to the input. The settings that will be sent to the AVS-47B are received from the cluster.

Stabilisation Delay (in milliseconds) prevents the program from proceeding immediately. Use this delay if the bridge needs time to settle.

Writing to the TS-530A is a slow process, so this Boolean input should be false, if there is no temperature controller, or if one does not want to change its state. If this input is true, data is written to both instruments. Writing to a missing controller does not produce any error, however.

This VI does not modify the configuration cluster. The 48-bit long string TXSTR, that is sent to the AVS-47B, is output only for trouble-shooting purposes. Instead, the received string **RXSTR** may be decoded by “DecodeAvs47bReading.vi” and “DecodeAvs47bConfiguration”. It contains all outputs from the bridge.

LVError In can be left open. **LVError Out** reacts to some errors in handling the USB or Com port.

INTERNALS

Data is transmitted to the AVS-47B as a 48-bit long binary string TXSTR. The bits are sent from MSB (most significant bit) to LSB (least significant bit) and they have the following meanings.

AVS-47B:

BIT	BITS	DATA
47..32	16	Remotely programmable reference voltage. The decimal input 0..20000 is divided by 5 and rounded to nearest integer before converting it to a 16-bit binary number and binary string. The reference DAC in AVS-47B has 12-bit resolution, so that the analog output has as large as 500µV steps although the input resolution is 100µV.
31..24	8	Address of the reference voltage register = 3.
23..22	2	00
21..20	2	Input 0..2 (zero, meas, cal)
19..17	3	Channel 0..7 (CH0..CH7)
16..14	3	Display 0..7 (R..530 set point)
13..11	3	Excitation 0..7 (0, 3µV..3mV)
10..8	3	Range 0..7 (open, 2Ohms..2Mohms)
7	1	0
6	1	Remote 0..1 (0=local, 1=remote)
5	1	0
4	1	Disable AL (0=enable, 1=disable)
3..0	4	0000

TS-530A:
All data is sent to the TS-530A via a synchronous serial line. Each register is programmed using an 8-bit address and a 16-bit data word. Each transmission includes also sending configuration to the AVS-47B. This does not affect the bridge operation, however, because the data to the bridge does not change. It is not possible to read PID settings from the TS-530A, so it is recommended to program all settings every time, and then the computer knows the state of the controller.

DATA = BITS 47..32 (16), ADDRESS = BITS 31..24 (8)

ADDRESS	DATA
9	set point 0..42000. One bit = 100µV.
10	proportional gain 0..11 (5dB..60dB), 15=zero gain
11	integrator 0=PD, 1..9=2s..1000s, 10=latched, 15=PD
12	derivator 0..7 (0..100s)
13	power bias 0..5 (0..100%)
14	power range 0..7 (OFF, 1uW..1mW)

Note that the TS-530A needs the set point as an integer, where one bit corresponds to 100µV. However, the set point is given as resistance to this VI for operating convenience. Conversion from a floating point resistance value to a decimal integer is made by “SetPointToInteger.vi” with the aid of the currently selected measuring range.

The low-level Picobus protocol and its related VIs are described briefly in the end of this document.

“ReadAvs47b.vi” is an example of how to use this VI.

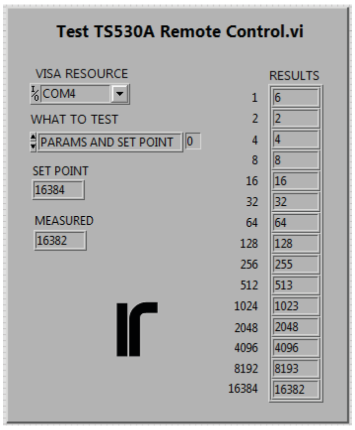
Test TS530A Remote Control.vi

Application example 7

Production of the TS-530A Temperature Controllers includes testing the remote programming of the PID parameter registers, their indicator LEDs and the set point DAC. Testing requires that the AVS-47B resistance bridge and the TS-530A are connected together using one 37-way ribbon cable for data and one short coaxial cable for the analog signal. The resistance bridge relays transparently those commands that are directed to the TS-530A.

This VI differs from the “Test Avs47b Remote Control.vi” in that the bridge is not tested but values are given to the various PID-parameter registers of the TS-530A so, that all status LEDs light in turn. The integer division takes each register back to 0 after it was tested.

The set point DAC is tested by giving it values of 2^i where i goes from 0 to 14. The DAC output is measured by the AVS-47B, therefore the display selector is set to 7 in the beginning.



Test TS530A Remote Control.vi

VISA RESOURCE: COM4

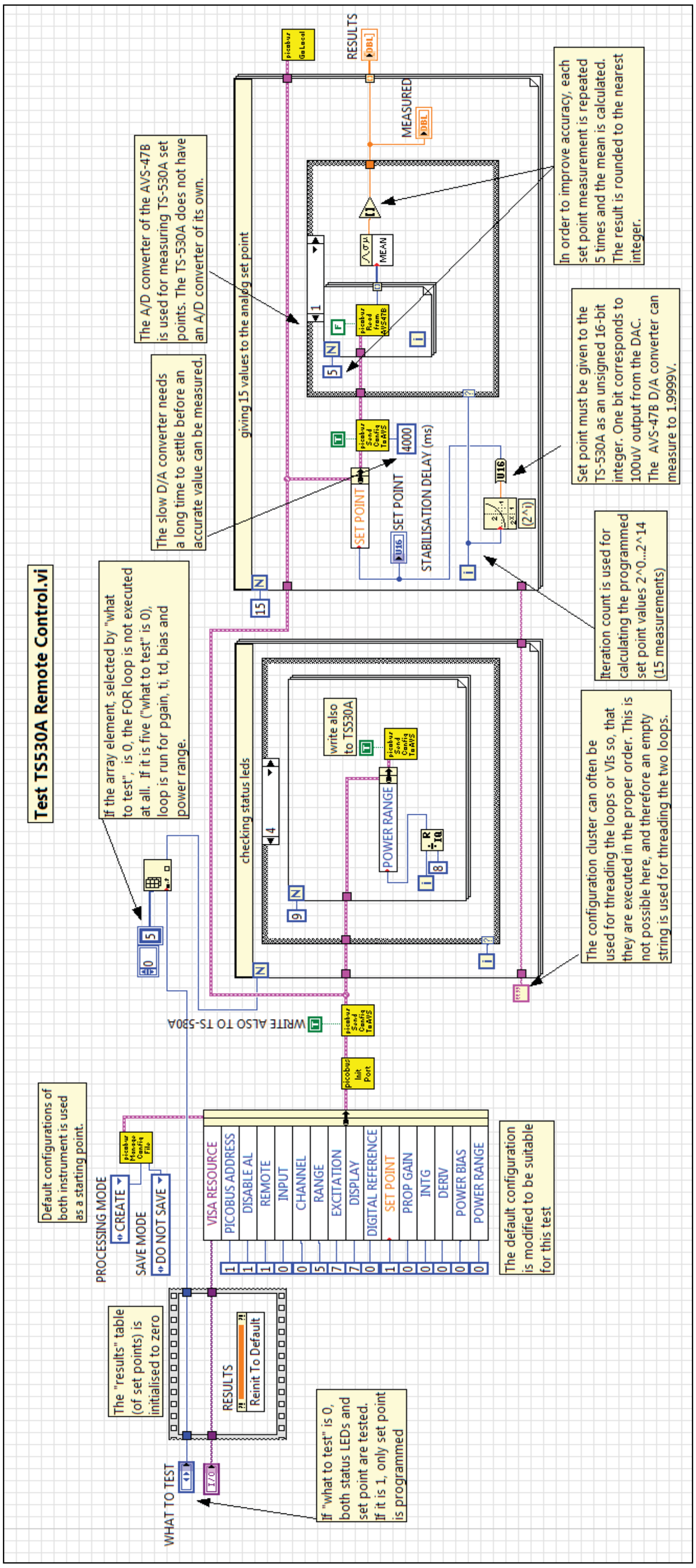
WHAT TO TEST: PARAMS AND SET POINT 0

SET POINT: 16384

MEASURED: 16382

RESULTS

1	6
2	2
4	4
8	8
16	16
32	32
64	64
128	128
256	255
512	513
1024	1023
2048	2048
4096	4096
8192	8193
16384	16382



Test TS530A Remote Control.vi

The A/D converter of the AVS-47B is used for measuring TS-530A set points. The TS-530A does not have an A/D converter of its own.

In order to improve accuracy, each set point measurement is repeated 5 times and the mean is calculated. The result is rounded to the nearest integer.

The slow D/A converter needs a long time to settle before an accurate value can be measured.

Set point must be given to the TS-530A as an unsigned 16-bit integer. One bit corresponds to 100uV output from the DAC. The AVS-47B D/A converter can measure to 1.9999V.

If the array element, selected by "what to test", is 0, the FOR loop is not executed at all. If it is five ("what to test" is 0), loop is run for pgain, ti, td, bias and power range.

Iteration count is used for calculating the programmed set point values 2^0...2^14 (15 measurements)

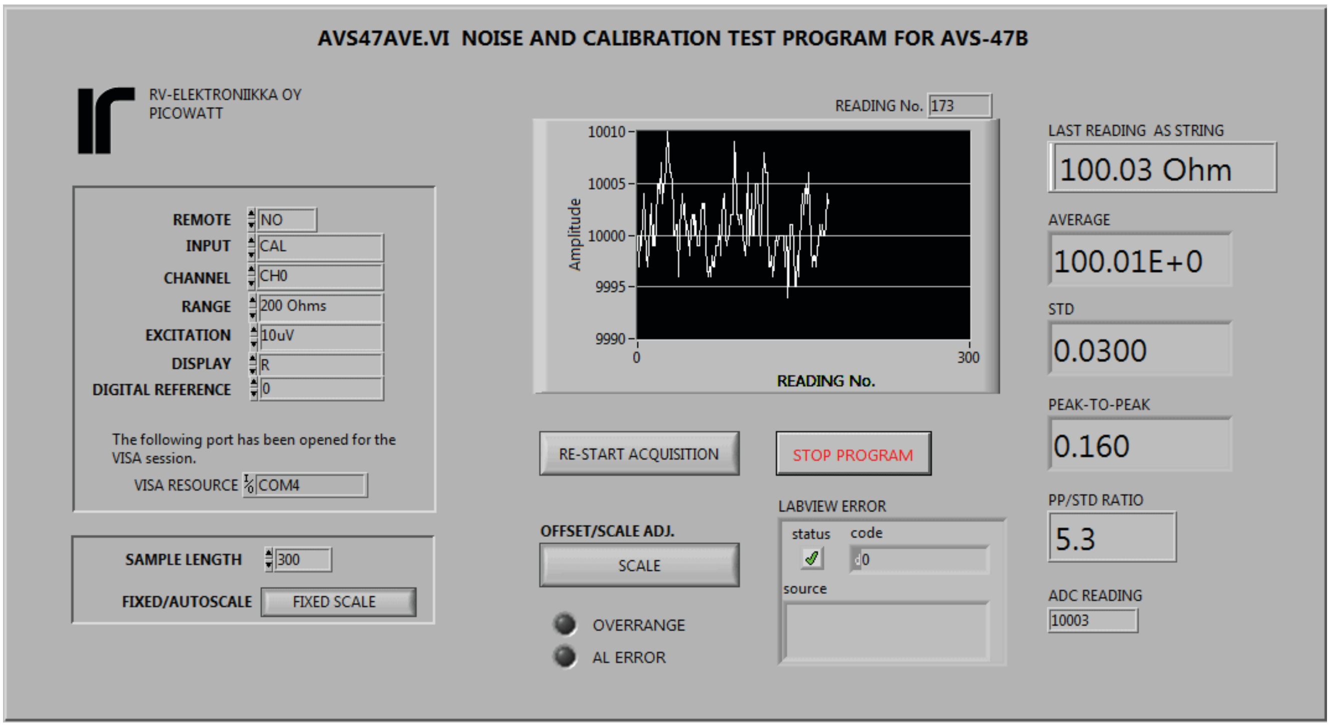
The configuration cluster can often be used for threading the loops or VIs so, that they are executed in the proper order. This is not possible here, and therefore an empty string is used for threading the two loops.

Default configurations of both instrument is used as a starting point.

The default configuration is modified to be suitable for this test

The "results" table (of set points) is initialised to zero

If "what to test" is 0, both status LEDs and set point are tested. If it is 1, only set point is programmed



AVS47AVE.VI NOISE AND CALIBRATION TEST PROGRAM FOR AVS-47B

RV-ELEKTRONIIKKA OY
PICOWATT

REMOTE: NO
INPUT: CAL
CHANNEL: CH0
RANGE: 200 Ohms
EXCITATION: 10uV
DISPLAY: R
DIGITAL REFERENCE: 0

The following port has been opened for the VISA session.

VISA RESOURCE: COM4

SAMPLE LENGTH: 300
FIXED/AUTOSCALE: FIXED SCALE

READING No. 173

LAST READING AS STRING: 100.03 Ohm
AVERAGE: 100.01E+0
STD: 0.0300
PEAK-TO-PEAK: 0.160
PP/STD RATIO: 5.3
ADC READING: 10003

LABVIEW ERROR
status: ✓
code: 0
source:

Avs47Ave.vi

Application example 8

Like the two previous examples, this is also a production test program. It is used for aligning the gains on the excitation ranges and for checking, that the preamplifier noise of each AVS-47B is within specified limits. The VI can be run either in remote or local modes.

OPERATING INSTRUCTIONS

The following settings are needed before running the VI:

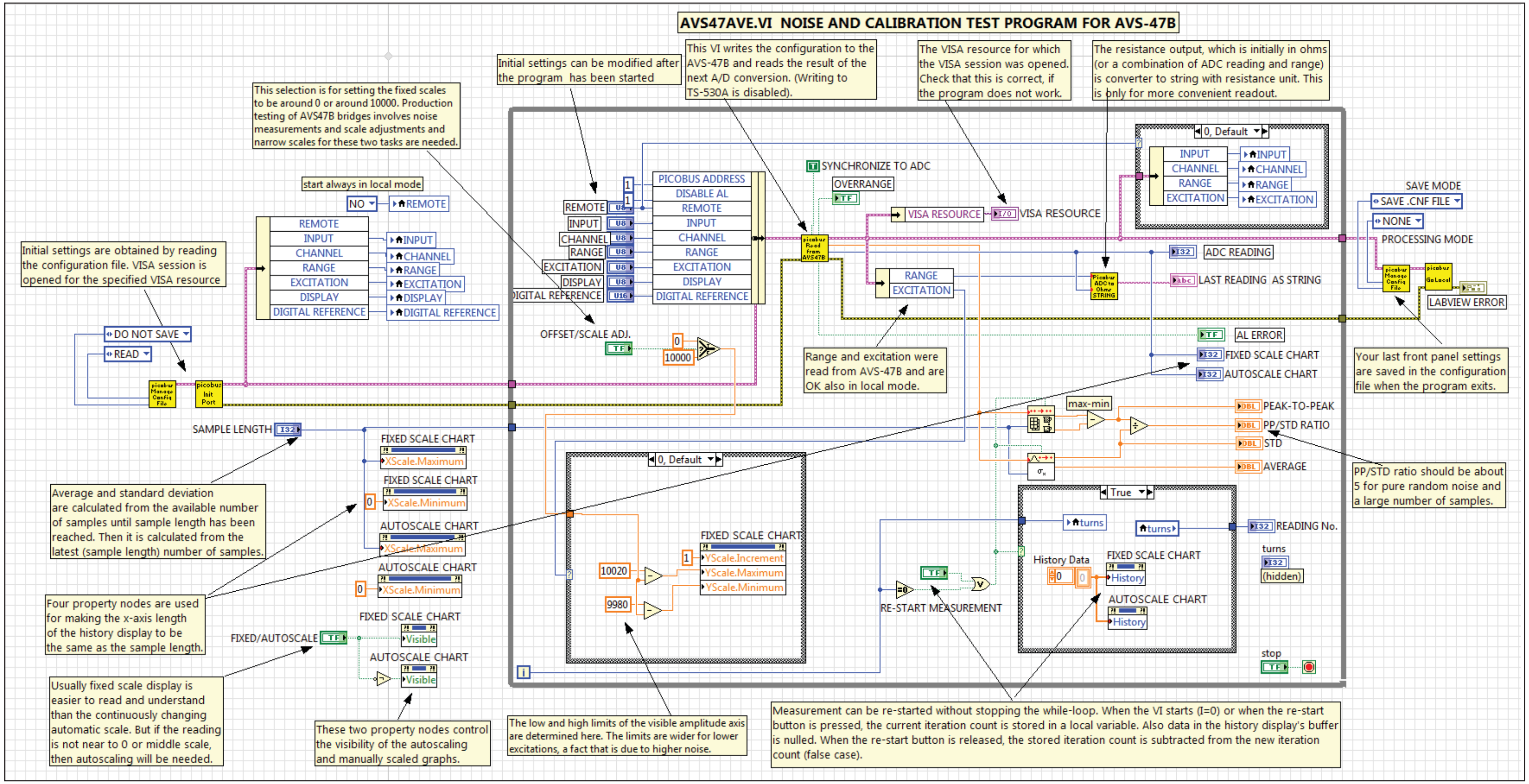
- VISA resource
- Sample length N: Low excitation ranges require long samples for a reliable average. A sample length of 500 is not too long for 3µV excitation. The display length adjusts to the sample length.
- Fixed/autoscale: Autoscale adjusts the y-axis of the display to be between the maximum and minimum value inside the last N readings. Fixed scales are different for different excitation ranges, so that noise is better visible on each range.

Often it is best to start with autoscaling. If the readings seem to be centered near to 0 or 10000, one can stop the program and re-start after having changed over to fixed scale. Center the display either around zero or 10000 by clicking “Offset/Scale Adj”. All calculations are independent of the display scaling and centering. Note that the display shows pure ADC values from 0 to 19999 regardless of resistance range, whereas calculations are made with true resistance values.

Recording can be re-started by pressing “Re-Start Acquisition” until the VI reacts. Then the display and all counters are reset. At first, the average is the mean value of the readings gathered after start or re-start, but once N readings is exceeded, it is the mean of N most recent readings. The width of the history display corresponds to the sample length.

The standard deviation STD is a measure of preamplifier RMS noise (or sensor noise, if it is more significant). For a pure random noise and a reasonably long sample, the peak-to-peak/STD ratio is about five. Much higher or lower values may be indications of high peaks (low excitation), or digitising steps of the A/D converter (high excitation).

Typical noise reading is 0.05-0.07 ohms when Input=cal, Range=200R and Excitation=3µV.



AVS47AVE.VI NOISE AND CALIBRATION TEST PROGRAM FOR AVS-47B

This selection is for setting the fixed scales to be around 0 or around 10000. Production testing of AVS47B bridges involves noise measurements and scale adjustments and narrow scales for these two tasks are needed.

Initial settings can be modified after the program has been started

This VI writes the configuration to the AVS-47B and reads the result of the next A/D conversion. (Writing to TS-530A is disabled).

The VISA resource for which the VISA session was opened. Check that this is correct, if the program does not work.

The resistance output, which is initially in ohms (or a combination of ADC reading and range) is converter to string with resistance unit. This is only for more convenient readout.

Initial settings are obtained by reading the configuration file. VISA session is opened for the specified VISA resource

start always in local mode

Range and excitation were read from AVS-47B and are OK also in local mode.

Your last front panel settings are saved in the configuration file when the program exits.

PP/STD ratio should be about 5 for pure random noise and a large number of samples.

Average and standard deviation are calculated from the available number of samples until sample length has been reached. Then it is calculated from the latest (sample length) number of samples.

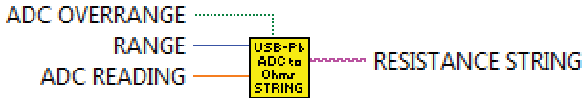
Four property nodes are used for making the x-axis length of the history display to be the same as the sample length.

Usually fixed scale display is easier to read and understand than the continuously changing automatic scale. But if the reading is not near to 0 or middle scale, then autoscaling will be needed.

These two property nodes control the visibility of the autoscaling and manually scaled graphs.

The low and high limits of the visible amplitude axis are determined here. The limits are wider for lower excitations, a fact that is due to higher noise.

Measurement can be re-started without stopping the while-loop. When the VI starts (I=0) or when the re-start button is pressed, the current iteration count is stored in a local variable. Also data in the history display's buffer is nulled. When the re-start button is released, the stored iteration count is subtracted from the new iteration count (false case).



OhmsNumberToStringWithUnit.vi

The purpose of this small VI is to convert the integer reading from the A/D converter of the AVS-47B to a real number resistance value, which is more convenient to read on computer display. The conversion is made with the help of the current measuring range.

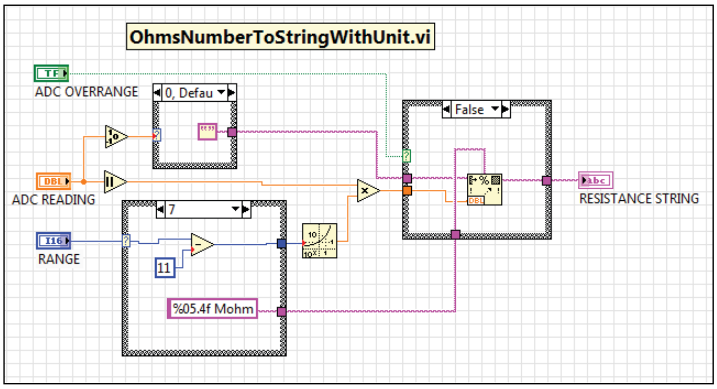
If the input voltage to the ADC exceeds 1.9999 Volts, it outputs plain zero but sets its overrange flag. In this case, the output string is “ADC OVR”.

TemperatureControlSystem with R display.vi

Application example 9

This AVS-47B + TS-530A temperature control system offers the same functionality as if the two instruments were operated from their physical front panels, with a few improvements:

- The set point is given as resistance. The physical TS-530A requires the set point as an integer whose magnitude depends on AVS-47B measuring range. Changing range is easier with this VI.
- The heater resistance must be less than 200 ohms and the output power is calculated correctly for all resistances.

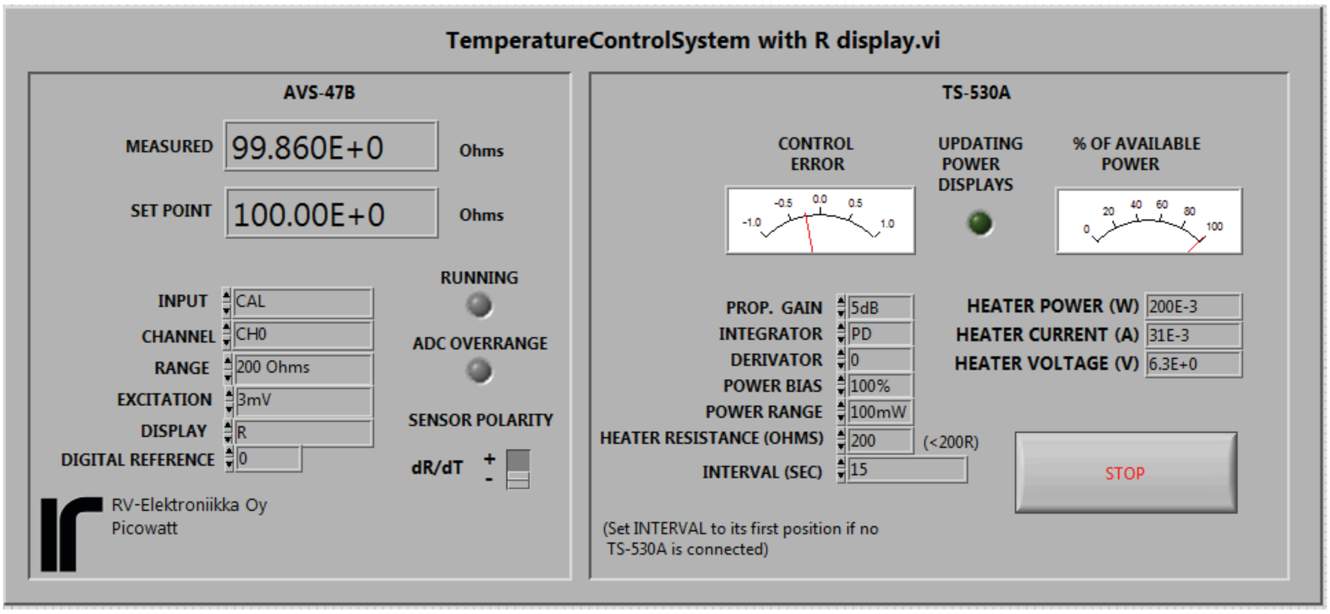


- The panel meter shows percentage of the power available with the existing heater and the selected power range. Full deflection indicates maximum power on that range using your specific heater, and the power indicator shows, what it is in Watts.

OPERATING INSTRUCTIONS

Initial values are obtained by reading the configuration cluster. The AVS-47B is set in remote mode and after that, the settings can be changed.

The dR/dT switch must be set correctly BOTH in the TS-530A and this VI. The switch in the TS-530A determines the polarity of the control loop, whereas the switch in this VI determines only the polarity of the analog error meter (positive deflection leads to more heating).


TemperatureControlSystem with R display.vi

AVS-47B

MEASURED 99.860E+0 Ohms

SET POINT 100.00E+0 Ohms

INPUT CAL

CHANNEL CH0

RANGE 200 Ohms

EXCITATION 3mV

DISPLAY R

DIGITAL REFERENCE 0

RUNNING

ADC OVERRANGE

SENSOR POLARITY

dR/dT + -

RV-Elektroniikka Oy
Picowatt

TS-530A

CONTROL ERROR

UPDATING POWER DISPLAYS

% OF AVAILABLE POWER

PROP. GAIN 5dB

INTEGRATOR PD

DERIVATOR 0

POWER BIAS 100%

POWER RANGE 100mW

HEATER RESISTANCE (OHMS) 200 (<200R)

INTERVAL (SEC) 15

HEATER POWER (W) 200E-3

HEATER CURRENT (A) 31E-3

HEATER VOLTAGE (V) 6.3E+0

(Set INTERVAL to its first position if no TS-530A is connected)

When searching for optimum PID parameters, start without using derivation or integration. Try using almost as much gain as the system tolerates without oscillating in P control mode. Reduce gain and add integration. Too much integration makes the system unstable again. Derivation increases sensitivity to noise, and makes the system easily unstable, but a small amount can speed up settling after changes in set point by reducing overshoot.

"Interval" specifies, how often the heater power is measured. Measuring power is a slow process, and during it, the resistance is not measured. Do not use a short interval, if you do not need power readings all the time.

The control error panel meter was made nonlinear so, that it has most sensitivity for small errors and low sensitivity for large errors. The meter is not calibrated in any way.

The possibly modified settings are saved in the configuration file when the VI ends.

Control System with R and T displays.vi

Application example 10

This VI adds a number of features to the previous example. First of all, measurement results of calibrated sensors are shown as both resistance and temperature. Also the control set point can be given as resistance or temperature. The TS-530A -section has a new control called "CONTROL". By selecting "HOLD", the error signal is nulled and the integrator is latched, and some short time is available for measuring other sensors while the output power remains reasonably constant.

OPERATING INSTRUCTIONS

Like before, initial values are obtained from the configuration cluster. The possibly modified settings are saved in the configuration file when the VI ends.

This (and the previous) VI can be run also without the temperature controller. Then it roughly corresponds to the "Free Running Avs47b with temperature display.vi". Now we assume that the TS-530A is connected and powered.

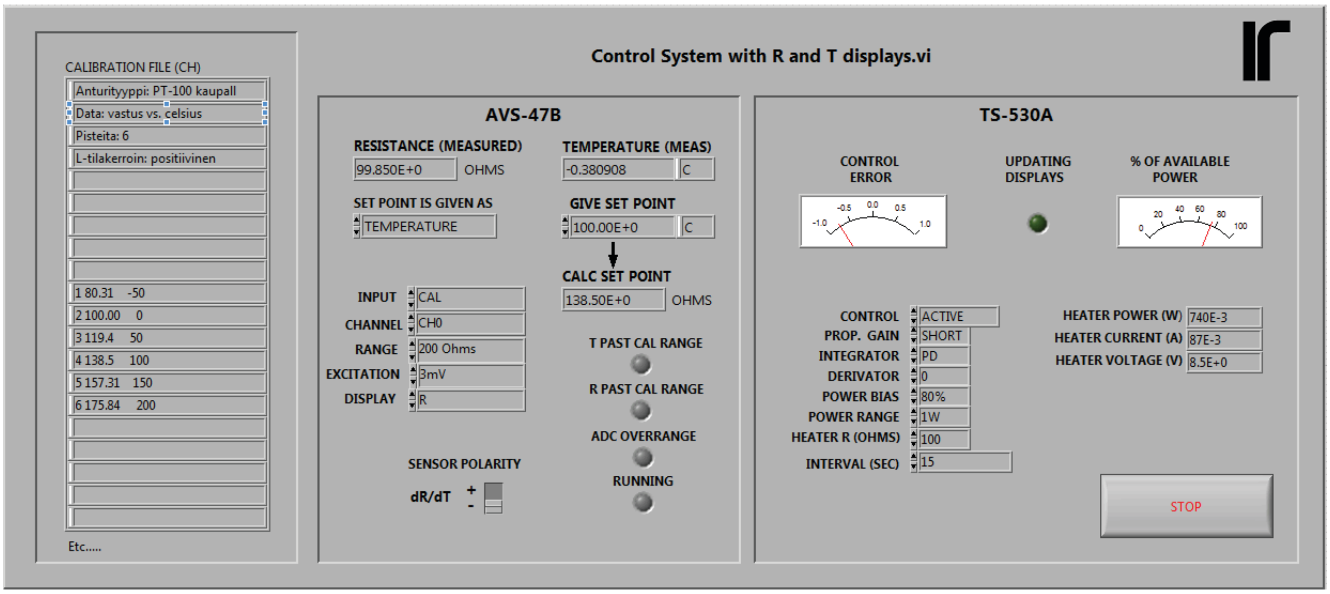
After start, select the control channel. The pane on the left shows the beginning of the calibration file. This is for your information. Set input to measure, select proper range and excitation. Display must be R. Set the dR/dT slide switch to show correct polarity (the TS-530A must have the same dR/dT setting).

Select, whether your set points will be in ohms or temperature. Then give the set point. If set point is in temperature, the CALC SET POINT indicator shows, what it is as resistance. There are some lights warning about ADC overrange and whether the set point is beyond calibration limits. The RUNNING light blinks indicating that resistance is being measured.

Select ACTIVE control mode and PID parameters (based on experience, guessing or luck). Change the heater value, if necessary. The INTERVAL should not be too long when tuning the system initially. Try 15 seconds. When the UPDATE indicator lights, the system measures the heater current and voltage. During that time, resistance is not measured, but the analog control loop works continuously.

When the system is at or near to equilibrium, you can interrupt active control momentarily by selecting control = HOLD. In equilibrium state, the error signal of a PID controller is zero and the integrator alone is responsible for the heating power. In HOLD mode, the error is forced to zero and the integrator input is disconnected so, that it more or less maintains its output voltage. An analog integrator will have some drift and the heat load may also fluctuate, and therefore one shall not stay in the HOLD mode too long before selecting the control channel again.

Return back to the control channel and restore its range, excitation etc. settings. Make control active again once the bridge has stabilised.



Control System with R and T displays.vi

CALIBRATION FILE (CH)

Anturityyppi: PT-100 kaupall
Data: vastus vs. celsius
Pisteita: 6
L-tilakerroin: positiivinen
1 80.31 -50
2 100.00 0
3 119.4 50
4 138.5 100
5 157.31 150
6 175.84 200

Etc.....

AVS-47B

RESISTANCE (MEASURED): 99.850E+0 OHMS
TEMPERATURE (MEAS): -0.380908 C
SET POINT IS GIVEN AS: TEMPERATURE
GIVE SET POINT: 100.00E+0 C
CALC SET POINT: 138.50E+0 OHMS
INPUT: CAL
CHANNEL: CH0
RANGE: 200 Ohms
EXCITATION: 3mV
DISPLAY: R
SENSOR POLARITY dR/dT + -
T PAST CAL RANGE ●
R PAST CAL RANGE ●
ADC OVERRANGE ●
RUNNING ●

TS-530A

CONTROL ERROR
UPDATING DISPLAYS ●
% OF AVAILABLE POWER

CONTROL: ACTIVE
PROP. GAIN: SHORT
INTEGRATOR: PD
DERIVATOR: 0
POWER BIAS: 80%
POWER RANGE: 1W
HEATER R (OHMS): 100
INTERVAL (SEC): 15
HEATER POWER (W): 740E-3
HEATER CURRENT (A): 87E-3
HEATER VOLTAGE (V): 8.5E+0

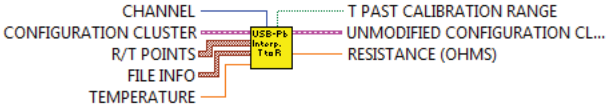
The calibration file describes a cheap PT-100 resistor with only 6 breakpoints. Because R=100 ohms at 0C°, this file can be used when input = CAL. You can also generate arbitrary inputs by selecting a suitable measuring range and setting the display to ADJ REF.

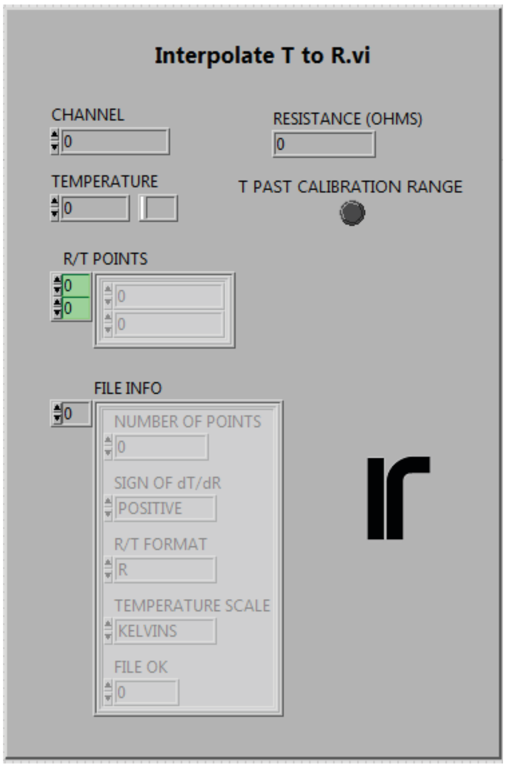
Interpolate T to R.vi

When temperature control set point is given as temperature, interpolation from T to R is required. LabView's interpolation algorithm requires that the argument be first and arranged in ascending order. For interpolation from T to R, data must be re-arranged.

Based on the selected channel, one 1D array of breakpoints is extracted from the 2-dimensional “R/T Points” -array. The size of this 1D array is determined, and a FOR loop swaps the R and T elements of each point. The resulting array of (T,R) points may have temperatures either in ascending or descending order, depending on the polarity dR/dT. If the sign is negative, the array must be reversed.

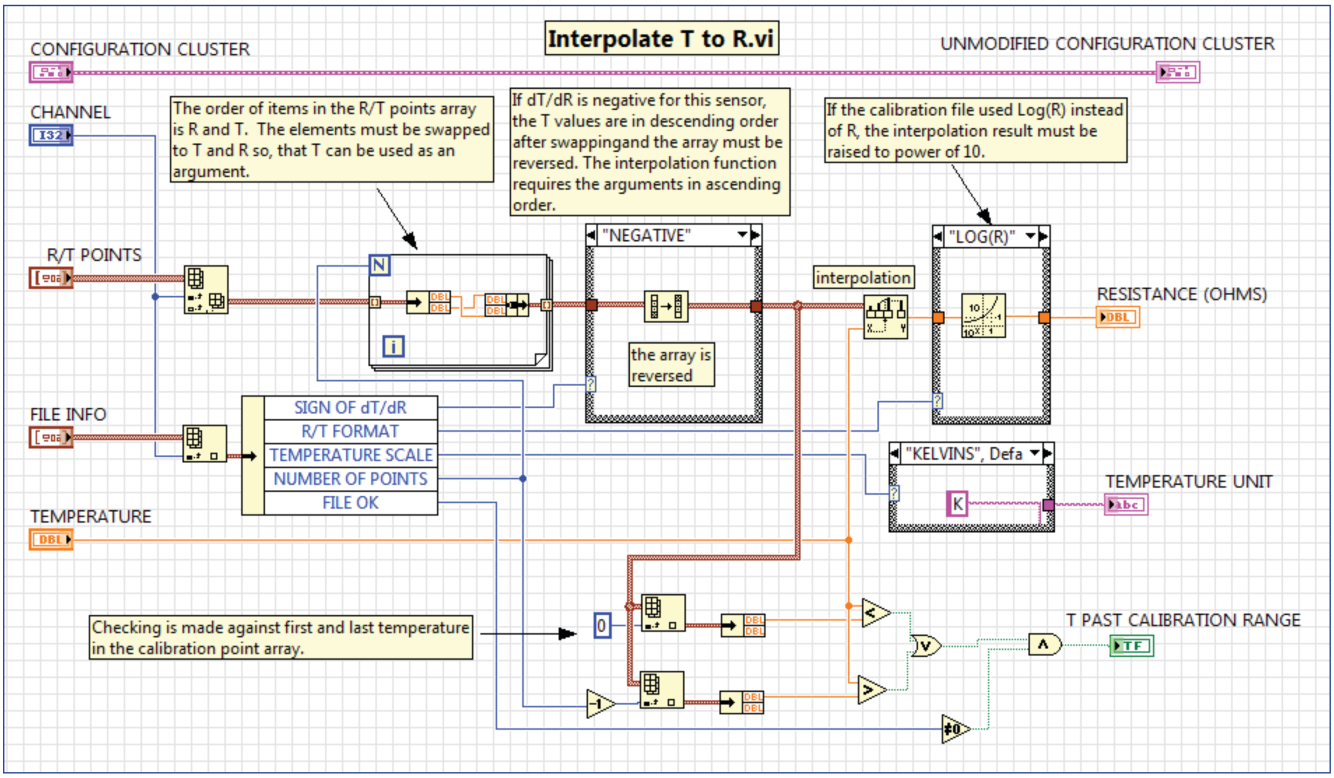
After these initial operations, linear interpolation is made by LabView's “Interpolate 1D array” -function. The input temperature is checked against the lowest and highest T value in the calibration array, and a warning is issued if the input is outside these limits. The interpolation algorithm returns the resistance of the nearest limit in that case.





Interpolate T to R.vi

CHANNEL: 0
RESISTANCE (OHMS): 0
TEMPERATURE: 0
T PAST CALIBRATION RANGE ●
R/T POINTS: 0 0 | 0 0
FILE INFO: 0
NUMBER OF POINTS: 0
SIGN OF dT/dR: POSITIVE
R/T FORMAT: R
TEMPERATURE SCALE: KELVINS
FILE OK: 0



Interpolate T to R.vi

The order of items in the R/T points array is R and T. The elements must be swapped to T and R so, that T can be used as an argument.

If dT/dR is negative for this sensor, the T values are in descending order after swappingand the array must be reversed. The interpolation function requires the arguments in ascending order.

If the calibration file used Log(R) instead of R, the interpolation result must be raised to power of 10.

the array is reversed

Checking is made against first and last temperature in the calibration point array.

Scanner with R and T Displays.vi

Application example 11

This single-cycle scanner has both resistance and temperature displays and autoranging, which can be set individually for each channel. The scan parameters, which are described below, can be defined beforehand by using “Manage Scan Info File.vi” or using this application example at the time of scanning. The latter option will be discussed here, the former alternative will be handled in the next chapter. Information about the available R/T calibration files must have been given earlier by using the “Manage RT Files.vi”. Changes in the scan info file do not affect settings in the configuration file.

OPERATING INSTRUCTIONS

Start the VI. Do not edit the scan settings before starting, because the edits will be overwritten by those coming from the scan information file “avs47b_scan_info.cnf”.

Check the scan settings. Select a channel and enable or disable scanning of this channel. The channels will be scanned in numerical order, but you can disable or enable any channels at will.

Specify the input: you can also include channels with ZERO or CAL input in the cycle, if you are not short of channels. The RANGE setting defines the range to be used for this sensor in manual ranging mode; in autorange mode it is only the starting point. In autorange mode, the last used range will replace the initial setting when the VI ends.

Excitation will not be changed by autoranging.

Stabilisation delay gives the AVS-47B time to find balance after it has changed over from one channel to the next. On high excitations, some 5-10 seconds is enough, on the lowest excitations 20-30 seconds should be sufficient. It is a tradeoff between scanning speed and accuracy.

Average count determines, how many readings are acquired for the final reading. The A/D converter produces 2.5 readings/second.

After having checked and/or edited the scan settings, you have three choices:

1) start a single scan cycle. As soon as the cycle is finished, the possibly edited scan settings are saved and the AVS-47B is returned to its original state.

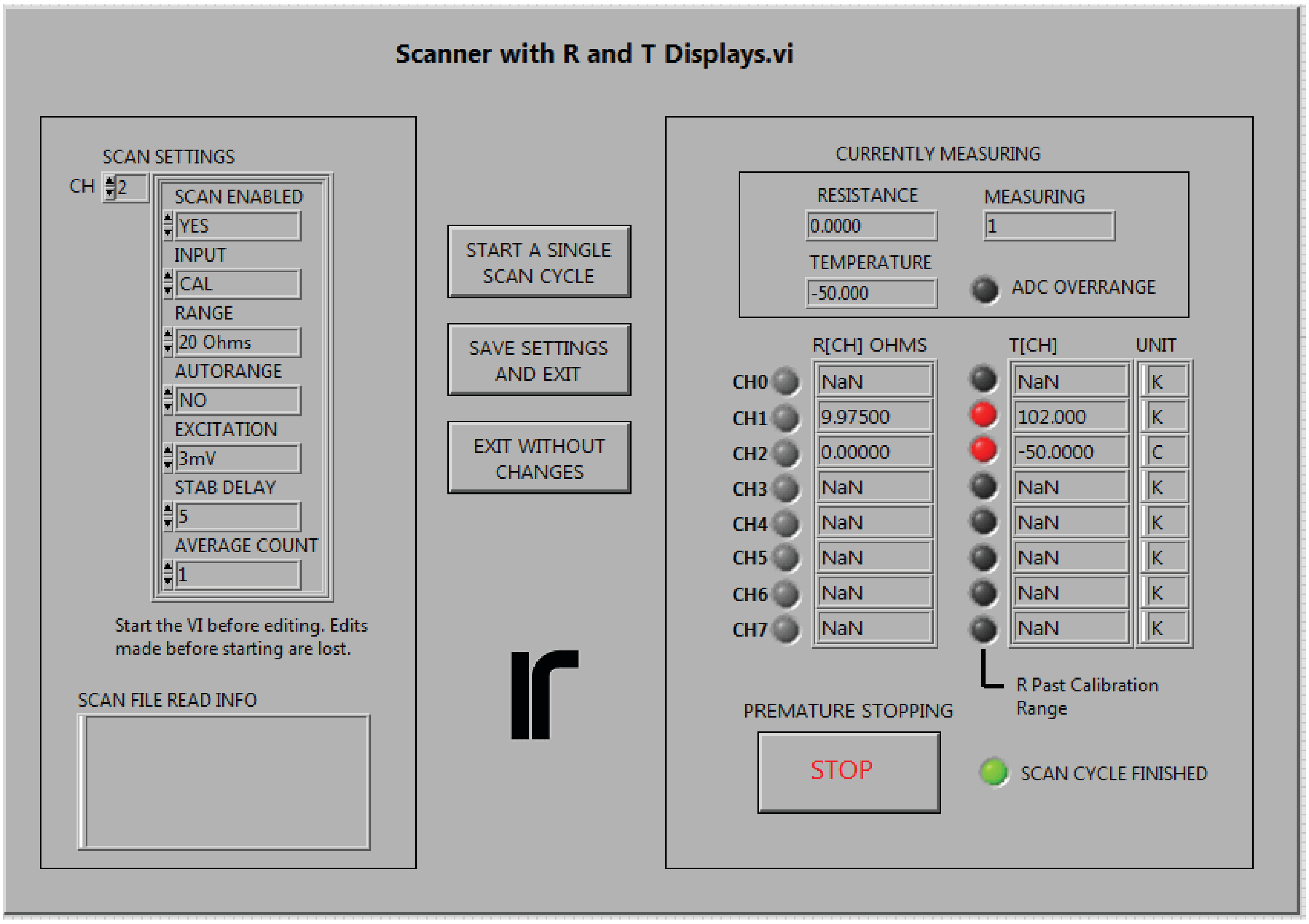
2) save the possibly edited settings and exit without scanning.

3) exit the VI. The original version of the scan settings will be saved and the changes you possibly made are discarded.

ADC OVERRANGE indication is not saved for each channel separately. Danger of overrange is highest, if autoranging is not enabled. The ADC returns an exact zero reading in case of overrange. In an average, such readings will distort the result. If all readings are overrange-zeros, the average will also be an exact zero. The possibility of overrange-distorted results is a good reason to always SCAN AUTORANGE ENABLED!!.

When autoranging is enabled, resistances below 2 Megaohms can be measured without danger of overrange.

The final results are collected in the output table. Resistors that were not scan-enabled are marked with NaN (Not a Number). Similary, channels that did not have a calibration file are marked with NaN. The temperature units come from the RT information file.



Scanner with R and T Displays.vi

SCAN SETTINGS

CH 2

SCAN ENABLED: YES

INPUT: CAL

RANGE: 20 Ohms

AUTORANGE: NO

EXCITATION: 3mV

STAB DELAY: 5

AVERAGE COUNT: 1

Start the VI before editing. Edits made before starting are lost.

SCAN FILE READ INFO

CURRENTLY MEASURING

RESISTANCE: 0.0000

MEASURING: 1

TEMPERATURE: -50.000

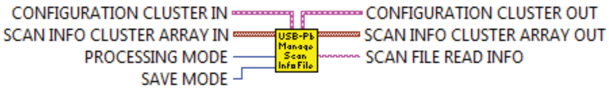
● ADC OVERRANGE

		R[CH] OHMS		T[CH]	UNIT
CH0	●	NaN	●	NaN	K
CH1	●	9.97500	🔴	102.000	K
CH2	●	0.00000	🔴	-50.0000	C
CH3	●	NaN	●	NaN	K
CH4	●	NaN	●	NaN	K
CH5	●	NaN	●	NaN	K
CH6	●	NaN	●	NaN	K
CH7	●	NaN	●	NaN	K

R Past Calibration Range

PREMATURE STOPPING

🟢 SCAN CYCLE FINISHED


CONFIGURATION CLUSTER IN
SCAN INFO CLUSTER ARRAY IN
PROCESSING MODE
SAVE MODE
USB-Pb Manage Scan Info File
CONFIGURATION CLUSTER OUT
SCAN INFO CLUSTER ARRAY OUT
SCAN FILE READ INFO

Manage Scan Info File.vi

The scan information file can be edited in two ways. When the “Scanner with R and T displays.vi”, which is a stand-alone virtual instrument, is started, it offers the user a possibility to check and edit all scanning parameters. An exception is the RT information file, which links channels with their respective calibration files. It must have been set correctly by using the “Manage RT Files.vi”.

For user’s own scanner applications, the “Manage Scan Info File.vi” offers a more versatile tool.

This VI has four processing modes and two save modes:

1) NONE: The Scan Info Cluster goes transparently through the VI. Depending on the save mode, scan settings are saved or not saved. The VI does not pause.

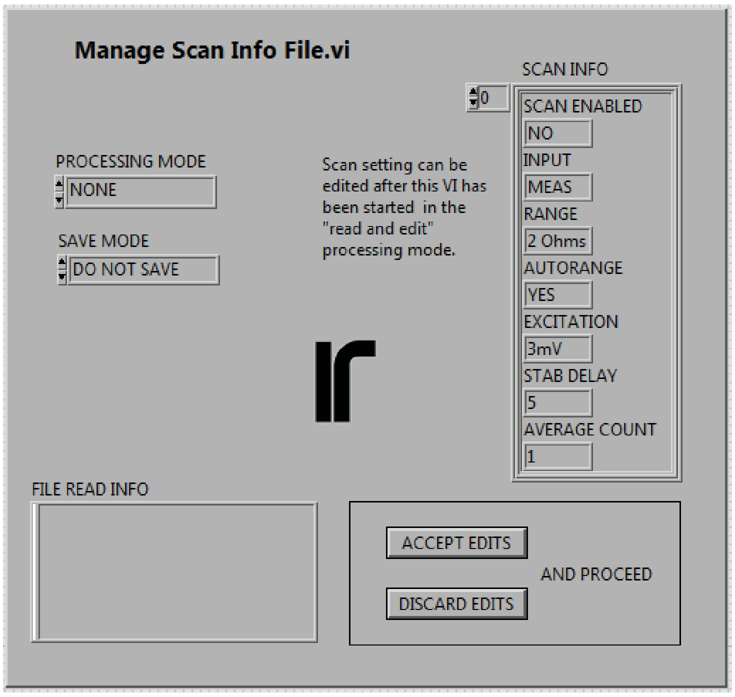
2) READ: The VI tries to open the “Avs47b_scan_info.cnf” file in the operating directory. If success, it is the output of this VI. If opening fails because the file is not there or is somehow invalid, a default cluster is used instead. The output is then the default cluster, which can be saved or not saved. The VI does not pause.

3) READ AND EDIT: The information file is read (or defaults are used, if opening fails) and they are presented as initial values. The VI pauses so, that you can edit or view the scanner settings of all channels. The editable settings were discussed in the previous paragraph.

Once ready with editing or checking you can choose either to accept edits or discard changes. The VI proceeds to end. The output is the possibly edited cluster, which can be either saved or not saved, depending on the save mode.

Note that THIS MODE PAUSES, requiring keyboard input before it proceeds. It is perhaps most suitable when run as a stand-alone program.

4) DEFAULTS: The output is the default cluster which can be saved or not saved. The difference between modes 2 and 4 is, that mode 4 uses defaults unconditionally whereas 2 reads the file if it exists. The VI does not pause.


Manage Scan Info File.vi

PROCESSING MODE: NONE

SAVE MODE: DO NOT SAVE

Scan setting can be edited after this VI has been started in the "read and edit" processing mode.

SCAN INFO: 0

SCAN ENABLED: NO

INPUT: MEAS

RANGE: 2 Ohms

AUTORANGE: YES

EXCITATION: 3mV

STAB DELAY: 5

AVERAGE COUNT: 1

FILE READ INFO

Low-level VIs for Picobus

Picobus is a proprietary protocol for communications between the AVS-47B and a computer, and it has been used in some of our products since 1994. Picobus is a synchronous serial interface, which can be implemented without digital intelligence. This fact makes it suitable for remote control of analog instruments that must emit as little RF interference as possible.

Picobus consists of four signal lines, two from the PC to the bridge and two from the bridge to the PC. They are called DC (Data from Computer), CP (Clock Pulse), DI (Data from Instrument) and AL (ALarm). Signals are transmitted as binary strings, whose length can vary depending on application and the signal's purpose. The word "Pico" comes from the typical application of the AVS-47B, which is low-temperature thermometry at sub-picowatt sensor dissipation level. The word "bus" tells that, in principle, one Picobus interface could be used for communications with many instruments. Usually Picobus controls only one instrument at a time, however. All Picobus lines are optically isolated from the computer, there is no galvanic connection between the PC and the AVS-47B.

Communications is based on transactions. One transaction consists of the following parts. All strings are sent starting from MSB (most significant bit).

1) Computer sends an 8-bit address. Four highest bits are the Picobus address (usually = 1) and four lowest bits are identically zero. Leading and trailing edges of the data and clock are separated by 1ms delays, which makes it possible to insert high-frequency filters in the Picobus lines, if they are necessary.
2) Ending of the address string is told to the instrument by interchanging the roles of the DC and CP signals: The clock is kept at zero while the data toggles two times high and low. This is something that can never happen in normal trasmission and therefore it can be recognized as the "strobe condition". If the sent address corresponds to the instrument's PBA (PicoBus Address), the strobe condition opens the instrument's input and output data ports for the next phase.
3) The data string is transmitted to the serial-in-parallel-out shift registers of the instrument. At the same time, data is taken out of the instrument's parallel-in-serial-out shift registers. The clock signal is used to synchronize the transmission in both directions. It is important to understand, that data from instrument corresponds to its state exactly at the time when the address was strobed in. Data from the AVS-47B includes all remotely controllable and readable items. It is not possible to write or read just one item, like range or A/D conversion result.
4) Ending of the data string is told to the instrument by using the strobe condition for the second time. This closes the data input and output ports, and Picobus could be used for communications with other instruments, which must have a different PBA.

Briefly, one transaction is: address-strobe-data-strobe. One transaction with the AVS-47B takes about 140ms.

Picobus needs two inputs to and two outputs from the computer. Formerly, they were conveniently available in a computer's RS-232 (or Com:) serial port. Now such ports have almost disappeared, but a USB-232 converter can replace the missing Com: port. Using the RS-232 signal names, CP=RTS and DC=DTR are outputs from the PC, DI=CTS and AL=DSR are inputs to the PC. From LabView, the states of these signals are written and read using property nodes of the appropriate VISA resource.

Following is a short description of the low-level VIs that are used for communications.

SetDC.vi and SetCP.vi Set the states of the DC and CP outputs to 1 or 0.

GetDI.vi and GetAL.vi Reads the states of the DI and AL inputs (1 or 0).

PbDelay_1ms.vi This is the shortest predictable delay that LabView (and Windows) offers. The delay time can be used by other simultaneously running VIs.

PbStrobe.vi Generates the strobe condition. Used after transmission of the address string and data string.

SendPbAddr.vi Sends the address string and strobes the address using PbStrobe.vi

RWpbData.vi Writes the configuring data and reads the configuration and the most recent A/D conversion result. The VI ends with PbStrobe.vi

DecodeAvs47bConfiguration.vi Extracts the configuration bits and interprets the configuration that existed just before the transaction: Input, channel, range, excitation and position of the display selector.

DecodeAvs47bReading.vi Interprets the most recent A/D conversion result. The conversion result is expressed in three alternative forms: 1) as an integer (-19999...+19999), 2) as a real number resistance in ohms and 3) as resistance plus unit (Ohm, kOhm, Mohm) string. Returns also a Boolean ADC Overrange indicator.

16_LinesDecoder.vi This simple VI just decodes an integer 0..15 to 16 Boolean signals that are used to light front panel LEDs in the scanner application example.

INDEX

Symbols

16_LinesDecoder.vi 35

A

About this user guide 5
ADC conversion result 13
A/D converter, of AVS-47B 15
AL (Alarm) signal 13
  Generating 12
Application examples
  1 Simplest.vi 11
  2 Free running Avs47b.vi 13
  3 ExampleOnUsingFunctionalGlobalVariable.vi 15
  4 Free Running Avs47b With Autoranging.vi 16
  5 Free Running AVS47b with Temperature Display.vi 17
  6 Test AVS47B remote control.vi 23
  7 Test TS530A remote control.vi 25
  8 Avs47Ave.vi 26
  9 TemperatureControlSystem with R display.vi 28
  10 Control System with R and T displays.vi 29
  11 Scanner with R and T Displays.vi 31
Asynchronous 3
Automatic scaling 26
Autoranging criteria 16
Autoranging.vi 16
Averaging, sample length 26
Avs47Ave.vi 26
AVS-47B
  Digital reference 8
  Settings 6
Avs47bconfig.cnf 6
AVS-47B older versions 3
Avs47b_rt_file_info.cnf 17, 19
avs47b_scan_info.cnf 31
AVS-47 jumpers 5

B

Breakpoints in calibr. file 17, 20, 22
Bundle by name 13

C

Calibration array of clusters 20, 22
Calibration files 17
  Checking 20
  Displaying beginning of 29
  File examples 1 and 2 21
  Information about 19
  Structure 20
Channel
  Selector 7
Com: port 3, 5, 34
Configuration cluster 6, 24
Configuration file 6
Connecting the Picobus 5
Control error panel meter 29
Control System with R and T displays.vi 29
Create configuration file 6
CTS 3

D

Datalog file 17
DecodeAvs47bConfiguration.vi 13, 24, 35
DecodeAvs47bReading.vi 13, 24, 35
Default configuration cluster 23
Digital reference 8
  Testing 23
Directory 6, 19
Disable AL 6, 7
Display selector 7
  Adj. ref 7
  dR 7
  Exc Volt 7
  Heater current 7
  Heater voltage 7
  Reference 7
  Set point voltage 7
dR/dT polarity 20, 30
DSR 3
DTR 3
DVM OVR indicator 11

E

Editing RT information 19
ExampleOnUsingFunctionalGlobalVariable.vi 15
Excitation
  Selector 7

F

Factory defaults 6
Fahrenheit temperature scale 19
FG global variable 15
Fixed scaling 26
Free Running Avs47b.vi 13
Free Running Avs47b With Autoranging.vi 16
Free Running AVS47b with Temperature Display.vi 17, 29
Functional Global Variable 15
FunctionalResistanceGlobal.vi 15

G

GetAL.vi 34

GetDI.vi 34
Global variable, functional 15
GoLocal.vi 11
GPIB-Picobus 5

H

Handshaking 3, 11
Heater resistance 8
Hold temperature control 29

I

IEEE-488 3
Initialisation of VISA resource 11
Initial values, transferring 13
InitPort.vi 11
Input selector 7
Installation of the USB-232 5
Interpolate R to T.vi 17, 22
  Internals 22
Interpolate T to R.vi 30
Interval, updating 8
Interval, updating power display 29
Introduction 3

J

Jumper settings of old bridges 5

L

LabView 3
Latched integrator 8
Linear interpolation 30
Local mode 6, 13
Local variables 13
Low-level VIs for Picobus 34

M

Manage Configuration File.vi 6
  Processing modes 6
Manage RT Files.vi 17, 19, 22, 31, 33
  Processing modes 19
Manage Scan Info File.vi 31, 33
  Processing modes 33
Measurement & Automation Explorer 11

O

OhmsNumberToStringWithUnit.vi 28
Optical isolation 5

P

Pathnames 5
PbDelay_1ms.vi 34
PbStrobe.vi 35
PD control mode 8
Picobus 3
  Address 6, 7, 34
  Cable 5
  Connecting 5
  Protocol 34
  Shift registers 34
  Signal lines 34
  Strobe condition 34
PID parameters
  Optimum 29
  Registers 25
Polling the AL signal 13
Property node 34

R

Range
  Selector 7
Range has changed indicator 17
ReadAvs47b.vi 11, 12, 25
Read configuration file 6
Remote mode 6, 7, 15
Requirements 3
RF emission 3, 34
RMS noise 26
R past calibration range 17
RS-232 3, 34
R/T format 19, 20
RTS 3
RWpbData.vi 35
RXSTR, string to computer 24

S

Sample length, averaging 26
Save .cnf file 6
Scan cycle 31
Scan information file 33
Scanner with R and T Displays.vi 31, 33
Scanning speed 31
Scan parameters 31
SendConfigurationToAvs47b.vi 13, 24
SendPbAddr.vi 35
Sensor
  Polarity dR/dT 8
Serial port 3
SetCP.vi 34
SetDC.vi 34
SetPointToInteger.vi 25
Shift register 15
Short circuiting the P amplifier 8
Simplest.vi 11
Software installation 5
Stabililsation delay 12, 31

Standard deviation STD 26
State machine 23
Status registers, of AVS47B 23
Structure of calibration file 20
Synchronisation to ADC 12, 13
Synchronous interface 3

T

TemperatureControlSystem with R display.vi 28
Temperature unit 17, 19
Test AVS47B Remote Control. vi 23
Test TS530A Remote Control.vi 25
Transaction 13
TS-530A 6, 24
  Analog cable 3
  Data cable 3, 8
  Derivator time constant 8
  Heater resistance 8
  Integrator time constant 8
  Interval of measuring power 8
  Power bias 8
  Power range 8
  Proportional gain 8
  Set point 8
  Set point DAC, testing 25
  Set point, R or T 29
  Set point voltage 7
  Settings 8
  Writing setting to 13
TXSTR, string from computer 24
  Description of bits 24

U

Unbundle by name 13
USB 3
USB-232 converter 3, 11, 34
  Installation 5
  Part number 3
USB-Picobus software for LabView 3

V

Virtual instruments 6
VISA resource 6
VI’s in this package 4

W

WEB site of Picowatt 3
Windows 3