# MEASURING TEMPERATURE FOR UNDER $35 USING LARVA

One of the most universal needs in the lab is to measure temperature. This could be the temperature of a tightly controlled experiment, the temperature of a portion of a structure under thermal feedback or simply the air temperature in the room.

Most data acquisition hardware costs hundreds of dollars or more and offers much more bandwidth and resolution than a typical temperature measurement requires. This application note shows how to use a resistive temperature device (costing less than $3), an Arduino Uno (costing less than $30) and the LArVa driver (free) to measure temperature over a wide variety of ranges.

LArVa is a free driver that turns an under $30 Arduino microcontroller into an ultra-low cost data acquisition system. While spending hundreds, thousands or more dollars on data acquisition system is necessary in some cases, inexpensive data acquisition can do a lot for very little cost… and with more and more amazing and inexpensive sensors and actuators available every day, there are vast possibilities.

## WHAT YOU NEED

You will need:

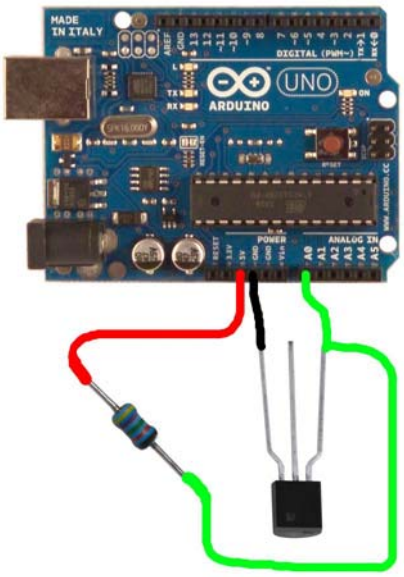
- An RTD. We are using the Honeywells TD5A. Available from digikey.com for under $3.
- An Arduino microcontroller. We are using the Arduino Uno, available from SparkFun.com for under $30.
- A computer running Windows OS.
- A USB cable to connect your computer to the Arduino
- The LArVa Simple Graph Application. LArVa is free from AngstromDesigns.com.
    - For download and installation, see User Guide UG01 –LarVa Install Guide for Windows at AngstromDesigns.com.
    - For basic voltage measurement, see Application Note AN01 – Basic Voltage Acquisition Using LArVa
- A 47.5 kohm resistor, some wire and a means to connect everything (soldering iron, solder-less breadboard, etc.)
- Single inline, male headers to plug into the Arduino are recommended.

## WIRING UP THE RTD

The TD5A data sheet recommends using a sense current of less than 100uA so the sense current will not heat the RTD and affect the temperature measurements. The TD5A resistance changes from 3,128 ohms at +150˚C to 1,584 ohms at -40˚C. For maximum flexibility, we will allow for using this full range, but for applications that don't require such a wide temperature range, we could trade less temperature range for more resolution and accuracy.

We'll use the 5V power line on the Arduino, which means to get a current of about 100uA, we should use a resistance around 50 kohms in the voltage divider with our TD5A RTD. A 47.5 kohm resistor is standard and will work well for us. Measure your resistor with a multimeter so you know it's exact resistance, ours is 47.1 kohms.

Wire up your Arduino, RTD and 47.5 kohm resistor, as shown below:

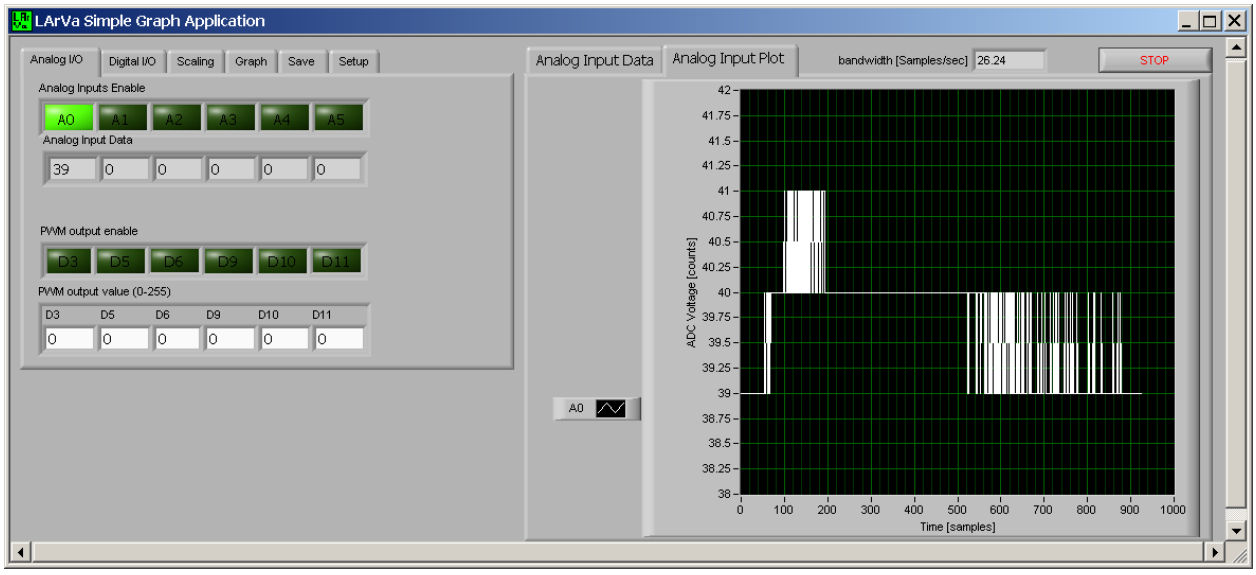



One lead of the RTD is connected to ground. The other side of the LED is connected to the resistor and to A0. The opposite side of the resistor is connected to 5V power.

## MEASURING THE RTD VOLTAGE

To measure the voltage, open the LArVa simple Graph application. For instructions on the free download and installation, see User Guide UG01 –LarVa Install Guide for Windows at http://angstromdesigns.com/larva/user-guides

Simple Graph will start up and will start gathering data on channel A0. This is the output voltage of our RTD. The number 0 corresponds to 0V and the number 1023 corresponds to 5V. Right now our resolution is very low. You can see this for yourself if you grab the RTD with your fingers, the A0 value only changes by one or two. Right-clicking on the Y-axis of the graph and selecting 'AutoScale Y' will zoom in the Y axis so you can see these small changes on the graph.

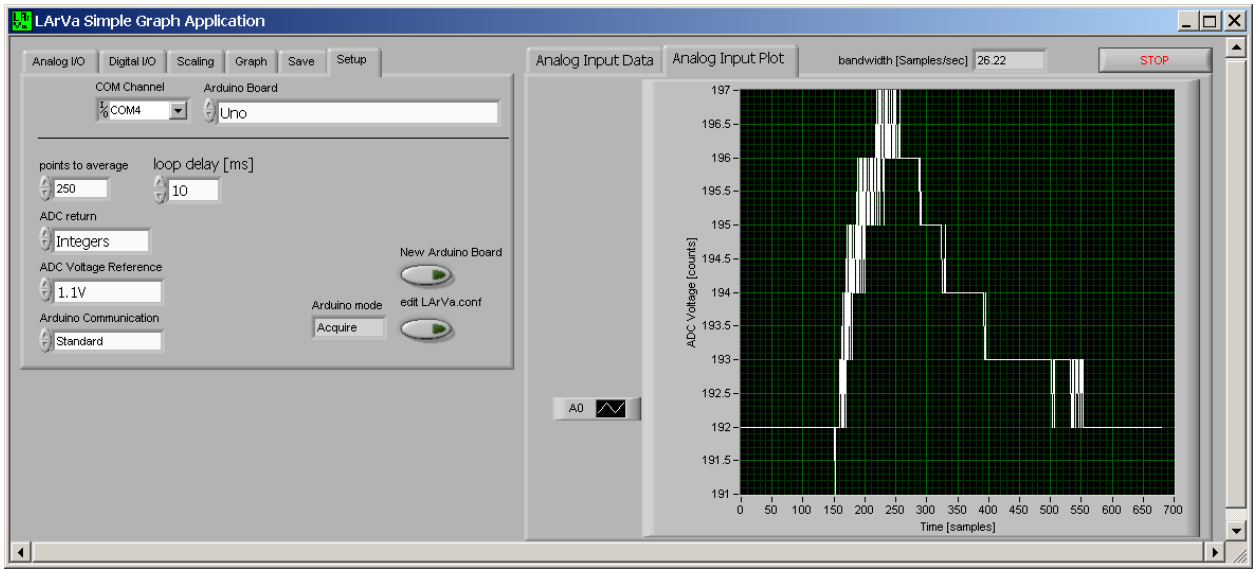


We are now gathering data from our RTD, but we need better resolution and we would like the output to be in degrees, rather than in counts.

## IMPROVING THE RESOLUTION WITH A VOLTAGE REFERENCE

Right now our analog input channel A0 is spreading it's 12 bits across 0-5V range. Because our measured voltage will only go between 0.31V (for +150˚C) and 0.16V (for -40˚C) we do lose resolution by having this large range available. We can limit the range to use all 12 bits across a 0-1.1V range by going to the 'Setup' Tab in Simple Graph and changing the value of ADC Voltage Reference to 1.1V.

We now have almost 5 times more resolution, so when you grab the RTD with your fingers you can see a larger change.




This is an improvement, but we'd like to do even better.

One option would be to build a precision voltage divider and use the AREF pin on the Arduino and the AREF pin voltage setting in Simple Graph ADC Voltage Reference to get another factor of 2 or 3 in resolution. You are invited to do this if you need the resolution, but we can also improve our measurements enough for gathering the air temperature in the lab by averaging many samples.

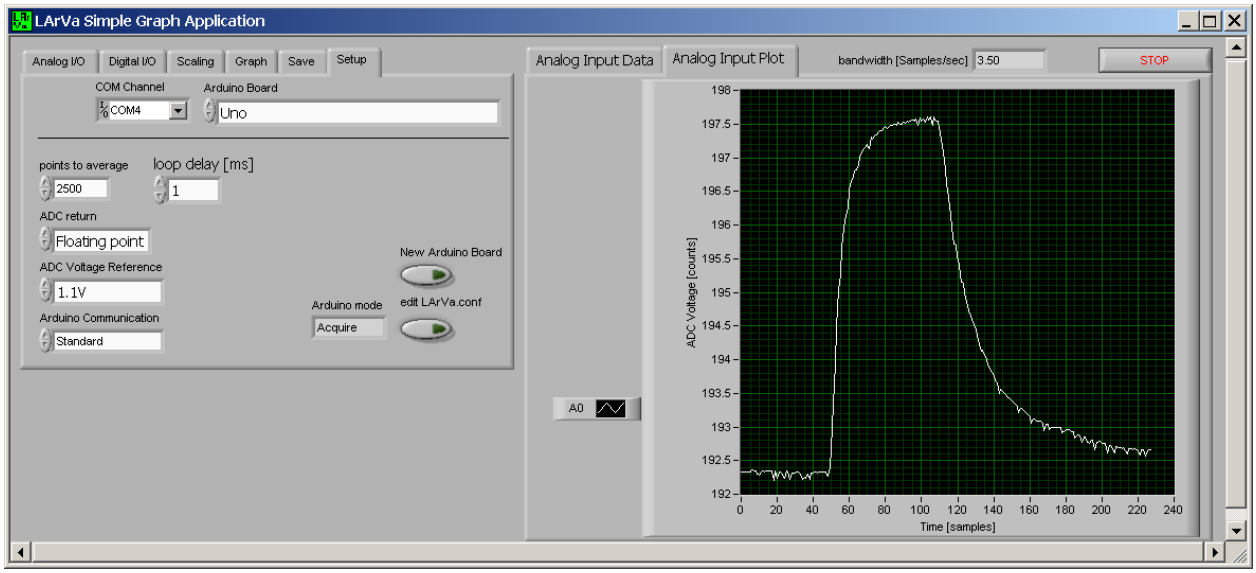
## IMPROVING THE RESOLUTION WITH AVERAGING

The LArVa firmware performs fast sampling of the analog inputs by default. We can increase this sampling to get better resolution. We can also decrease the delay between samples for faster data acquisition. Finally, we can bring the data reads back to the computer as floating point numbers, which have higher resolution than integers. All of these changes will improve our measurement.

In the 'Setup' tab in Simple Graph change the following:

- Change 'points to average' from 250 to 2500
- Change 'loop delay [ms]' from 10 to 1
- Change 'ADC return from Integers to Floating point

We now have much better resolution for our RTD:




We now have ample resolution for our measurement. But the units are not in degrees.

## CALIBRATING THE RTD OUTPUT

Calibration is a critical part of using any sensor. Our system has many possibilities for variation in it, such as variations in the resistor value, RTD performance or power supply voltages. Calibration will address these possible variations.

Calibration means measuring A0 with Simple Graph when the sensor is at a known temperature. The relationship between resistance of the RTD and temperature is given in the RTD datasheet:

$R_{RTD} = R_0 * (1 + 3.84*10^{-3} * T + 4.94*10^{-6} * T^2)$ where $R_0$ is the resistance at 0˚C, which is nominally 1,854 ohms, and T is the temperature of the RTD.

Because this relationship is quadratic, it would be best to perform a 3 point calibration. That is, to measure A0 at 3 different known temperatures. It would be best if those temperatures encompassed the full range of temperatures our sensor is likely to encounter. For example, we would like to monitor room temperature values over time, so good calibration choices would be 5˚C, 20˚C and 40˚C as the lab is very unlikely to get colder than 5˚C or hotter than 40˚C.

One common temperature calibration technique is to calibrate the sensor cold in an ice bath (container that has water and ice at equilibrium) so we know the water is 0˚C. Similarly, the hot side can be calibrated in a bath of boiling water; the water will be at 100˚C.

For simplicity, we will perform a 1 point calibration, at room temperature. Reading A0 gives me 191.2 and reading my thermometer tells me it is currently 20.1˚C in my lab. Using more than one point will improve the accuracy of your measurements.

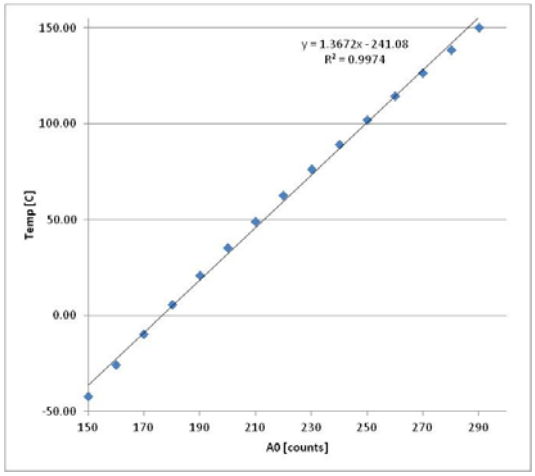
## APPLYING CALIBRATION AND SCALING BY NUMERICAL APPROXIMATION

To convert from A0 in counts to temperature we need 3 equations.

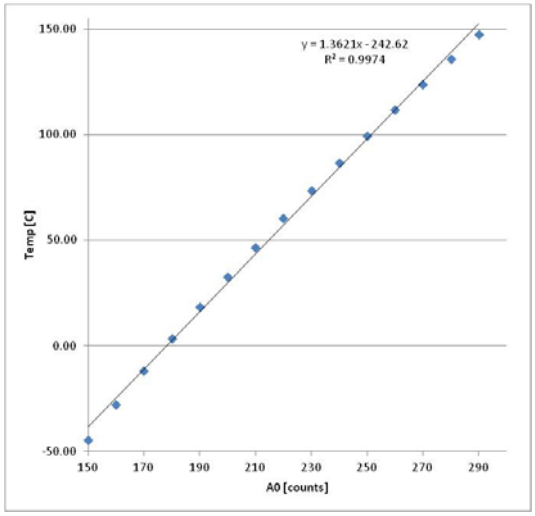
1. $V_{A0} = V_{ref} * (A0 / 1023)$ where $V_{A0}$ is the voltage at pin A0, $V_{ref}$ is the ADC reference voltage of 1.1V and A0 are the number of counts measured at the analog input pin

2. $R_{RTD} = R_{div} * V_{A0} / ( V_{cc} – V_{A0} )$ where $R_{RTD}$ is the RTD resistance, $R_{div}$ is the 47.1kohm voltage divider resistor and $V_{cc}$ is the power supply voltage of 5V.
3. $R_{RTD} = R_0 * (1 + 3.84*10^{-3} * T + 4.94*10^{-6} * T^2)$ where $R_0$ is the resistance at 0˚C, which is nominally 1,854 ohms, and T is the temperature of the RTD.

Solving these equations numerically in a spreadsheet gives a relatively linear relationship between A0 counts and RTD temperature:




Iteratively changing the $R_0$ value until the calibration of 191.2 counts corresponds to 20.1˚C gives a final value for $R_0$ of 1859.9 ohms. For a two- or three-point calibration we would fine-tune the values for the linear and quadratic terms for the RTD equation. However, our one-point calibration gives a linear approximation of temperature versus counts of T = 1.361*A0 – 243.22:

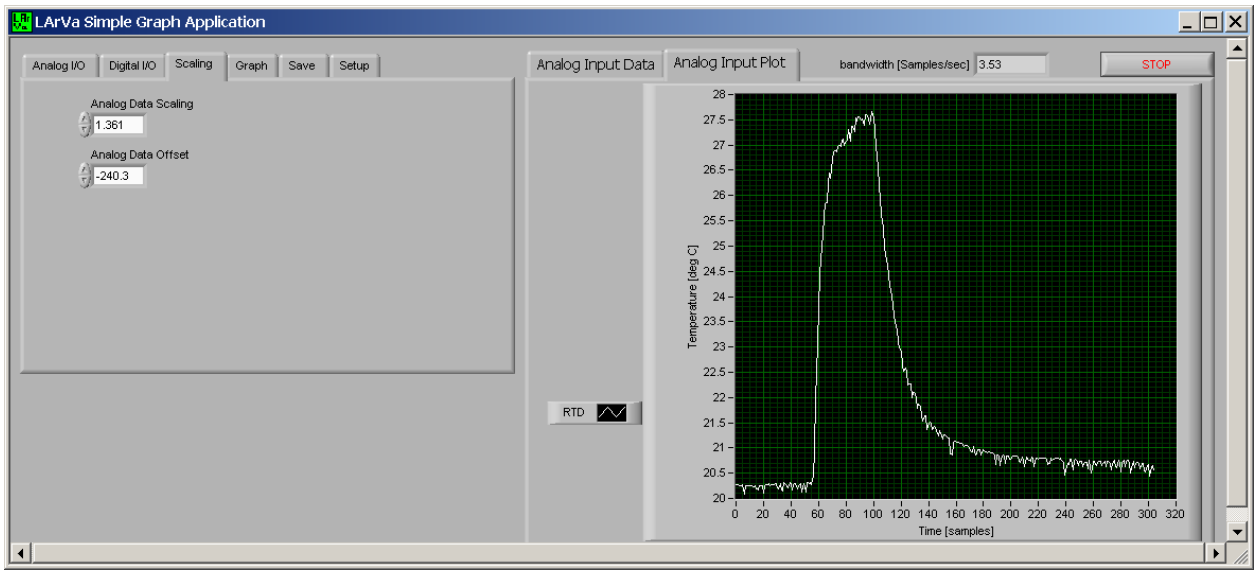



It is important to keep in mind that the linear approximation is a source of error, as can be seen in the graph above. This may not be appropriate for some measurements, see below for scaling that does not use a linear approximation.

Returning to Simple Graph we can plug in these values into the 'Scaling' tab. Type 1.361 into 'Analog Data Scaling' and type -242.62 into 'Analog Data Offset.' We can refine the offset value until the calibration is exact for room temperature to remove the errors in the linear approximation, but know that this just shifts our

linearization errors to different temperature values. We can also rename our Y axis to be temperature in degrees C and rename channel A0 in the 'Graph' tab.

We can now repeat our test of touching the RTD with our fingers to see a high resolution graph with proper units on the axies:



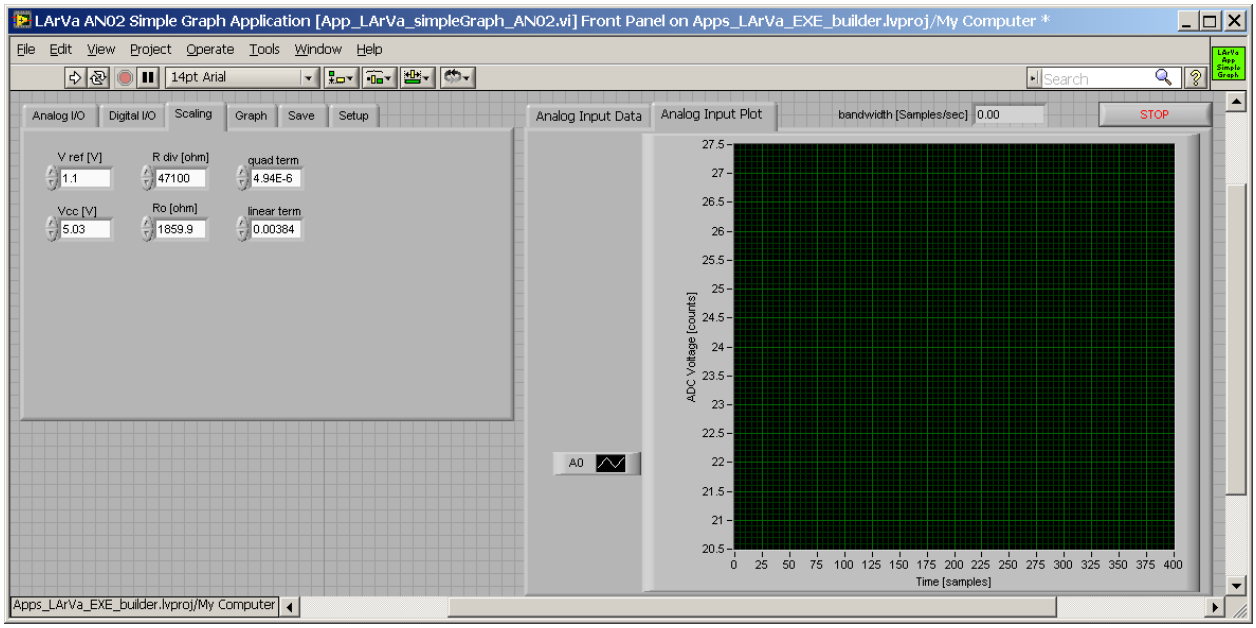

The calibration spreadsheet is included in the UG04 support files download, available at AngstromDesigns.com.

## APPLYING CALIBRATION AND SCALING BY MODIFYING SOURCE FILES

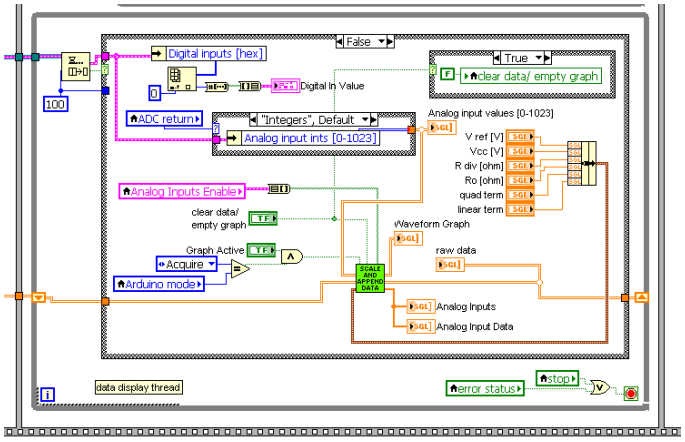
Simple Graph has values for analog data scaling and offset in the 'Scaling' tab. As we've seen above because the temperature is not linearly related to the analog input reading we will not be able to get accurate scaling over the full range using these settings. In order to get accurate scaling over a large range, custom scaling will be needed by modifying the source files that are included with the free, LArVa Simple Graph application. Note that a three-point calibration spanning the expected temperature range would also be best for sensitive measurements.

Open LArVa Simple Graph.vi using Labview 2010 or higher. For download and installation, see User Guide UG01 –LarVa Install Guide for Windows at AngstromDesigns.com.

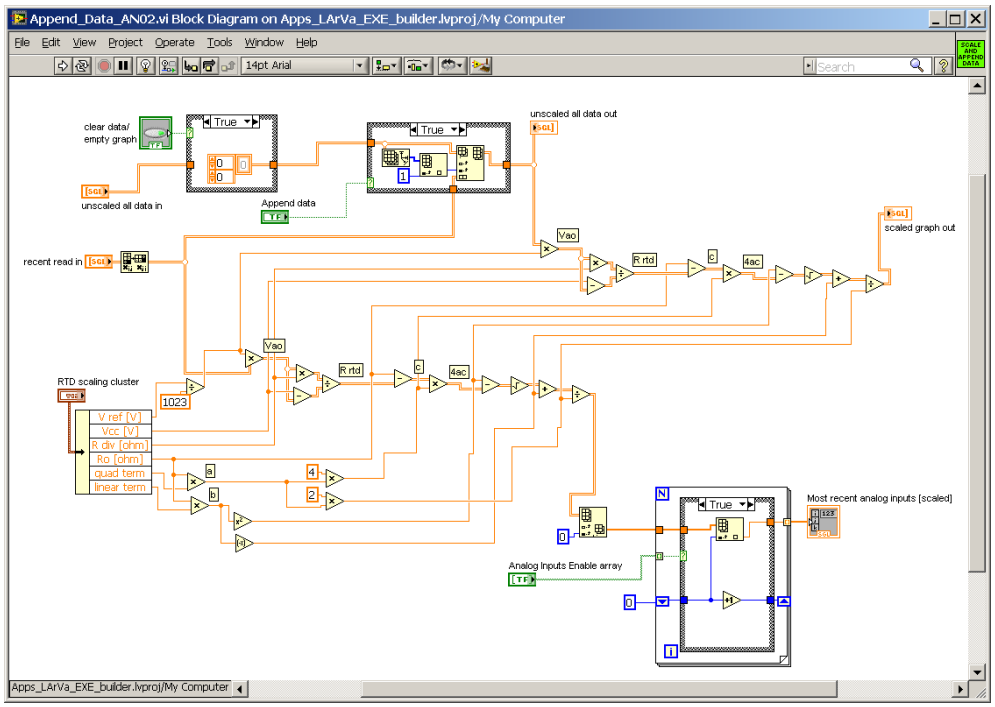
We want to include all of the calculations that were performed in the calibration spreadsheet in Labview, so we will need to replace the scaling and offset controls in the 'Scaling' tab with new controls and add the rest of our calibration controls:



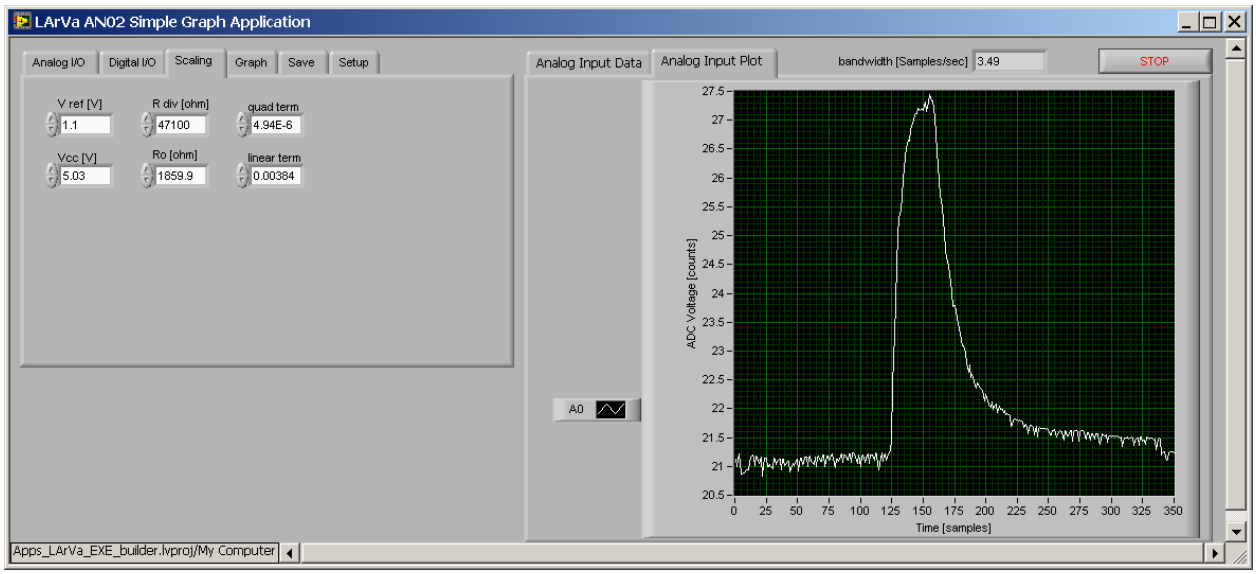

On the block diagram we see that the scaling values are applied in the data display thread (the bottom of the three threads) in the Append_Data subVI. Below, we have bundled up all of the controls we will need and passed them to this subVI:




Inside the Append_Data subVI we perform the calibration functions, just as in the section above but without the linear approximation:




We return to the modified Simple Graph application and repeat your measurement. Don't forget to set the front panel values, such as Y autoscaling, points to average, ADC return, loop delay and ADC Voltage Reference as above.




That's it. We now have calibrated measurement data in the proper units in real time. Note that final adjustment of the offset values is not necessary when we do this exact scaling.

Source files of this application note, an executable of this code and other support files are available for download. Look for the Support Files for UG04 at www.AngstromDesigns.com.

## SAVING THE DATA

To save the data, go to the 'Save' tab and click the button. The data will be in text format, so using the .txt file extension would be best.

## MOVING FORWARD TO OTHER SENSORS

This application note shows how to gather small signals using very inexpensive hardware. Many sensors have signal outputs and calibration issues that are very similar to RTDs, so the lessons used here can be applied to many different sensors. This will enable you to gather signals at a fraction of the cost of more traditional data application systems.

## LEARN MORE AND STAY IN CONTACT

For more applications, user guides, help, questions and too see what else is going on with LArVa, go to www.AngstromDesigns.com.