Sensors Conditioning Module

Version 1.00

Microstar Laboratories, Inc.
2265 116 Avenue N.E.
Bellevue, WA 98004
Tel: (425) 453-2345
Fax: (425) 453-3199
http://www.mstarlabs.com

Table of Contents

1. Introduction

Commands in this module

The S*ensor Conditioning Module* is a downloadable set of processing commands for the DAPL Operating System that runs on Data Acquisition Processor boards. These commands can be useful to facilitate processing of data captured from certain common types of sensor devices. The processing can be classified into two types:

1. **Aids for measurement.**

   These are primarily useful for passive sensor devices such as strain bridges and RTD thermal sensors, for which information about excitation and loading must be used in addition to the device response.

   - BRIDGE – Measure a resistive element in a bridge network configuration
   - DIVIDER – Measure a resistive element in a voltage divider network configuration

2. **Linearization.**

   This refers conventionally to the problem of taking an indirect measurement of a measured sensor response and applying an appropriate nonlinear conversion formula to calculate the value of a desired physical property in appropriate units (strain, degrees C, etc.).

   - GENPOLY – Generic polynomial linearization formula for a mildly nonlinear sensor device
   - RTD – Convert a resistance measurement to temperature using specialized polynomials
   - THERMISTOR – Convert resistance to temperature using the Steinhart – Hart model
   - THERMPOLY – Convert thermocouple potentials to temperature over a limited range

Installing the module

First install your DAP software and your DAP or xDAP unit. Run the *Data Acquisition Processor* control panel application in your Windows system, and select the *Modules* tab. Click on the Add button in the lower left. In the pop-up dialog, click the DAPL2000 or DAPL3000 button, depending on which DAP models and DAPL operating system version you have. Click the Browse button in the lower right corner to locate the copy of the `sensorm.dlm` module on your system. When you find it, select it and click *Open*. Use the default options. Back in the "Adding a module" dialog, click the *OK* button in the lower left corner. Your module should now be included in the DAPL system software, and it will be loaded automatically each time the host system is booted.

2. Command Reference

This section provides detailed descriptions for each of the commands in the Sensors module, with some examples of usage.

sensorm: BRIDGE

Convert measurements from a balanced resistive bridge network to determine resistance.

BRIDGE(VIN, VS, BALANCE, RLOAD, [RCOEFF, LTMP,] [ANGAIN,] ROUT)

Parameters

<*VIN*>
Input data pipe with bridge imbalance differential voltage measurements.

FLOAT PIPE

<*VS*>
Nominal or measured excitation voltage driving the bridge network.

FLOAT CONSTANT | *FLOAT PIPE*

<*BALANCE*>
Nominal or measured balancing network gain.

FLOAT CONSTANT

<*RLOAD*>
Nominal or calibrated load resistance, ohms at 0 degrees C.

FLOAT CONSTANT

<*RCOEFF*>
Temperature coefficient of resistance of load resistor, ohms per degree C.

FLOAT CONSTANT

<*LTMP*>
Input pipe with load temperature measurements, degrees C.

FLOAT PIPE

<*ANGAIN*>
The voltage gain used to measure the differential voltage *VIN* signal.

FLOAT CONSTANT

<*ROUT*>
Calculated output resistance measurements, in ohms.

FLOAT PIPE

Description

The BRIDGE command converts the differential voltage readings from pipe *VIN* to measure a resistance in a bridge network configuration. This is a very common configuration for strain measurements. Current from a known voltage excitation source drives a load resistance, and then passes through the unknown device to the drain voltage. On the other side of the bridge network, two known resistors form a voltage divider to establish a reference voltage. The differential measurements of voltage between the measurement and reference sides of the bridge are used to calculate the values of the unknown resistance, with results placed into the *ROUT* pipe, one output value for each input value.

The differential input measurement will reject "common mode" voltages, so it does not matter whether the bridge circuit is excited by balanced plus/minus supplies or a single-sided supply, as long as the common mode voltages at the reference and measurement junctions are within the measurable range. Because the quiescent voltages at the reference and measurement junctions are configured to approximately match, the differential voltage changes between these two points can be measured with gain to improve measurement resolution. If a gain other than 1.0 is used, either specify it as the value of the *ANGAIN* parameter or correct for the gain prior to sending the *VIN* data to the BRIDGE command.

If the supply voltage is very well regulated, it can be specified as a constant *VS* parameter. For best accuracy, measure the actual supply voltage accurately rather than assuming a nominal value. If the supply is subject to small but relevant variations, measure supply and bridge voltages simultaneously, and provide the supply measurements in units of volts through a *VS* pipe for each measurement from *VIN*. For balanced plus-and-minus supplies, specify the supply-to-supply voltage.

Ideally, the voltage dividers on the reference and measurement sides of the bridge produce exactly the same voltage at the nominal reference operating point within the normal operating range. Obtaining a perfect balance is not really necessary. Measure the resistances in the balancing network accurately, then set the *BALANCE* parameter equal to the ratio

```
BALANCE  =  Rg / (Rs + Rg)
```

where

Rs is the balancing resistor on the positive supply side

Rg is the balancing resistor on the negative supply or ground side

If measurement accuracy is not critical, use nominal values of the resistors to determine the value of the *BALANCE* parameter. Good balancing resistors will not cause excessive power supply loading, and will establish a zero differential reading near the center of the operating range.

Resistors in the balancing network are presumed to be located in a controlled operating environment where small temperature variations affect both balancing resistors the same, hence the ratio remains very stable. If the loading resistor is also located in a controlled environment, it too can be presumed to maintain a consistent operating temperature. Specify an accurately measured value of the *RLOAD* parameter at the stable operating temperature, in ohms. If accuracy is not critical, use the nominal loading resistor value.

When load temperature is not so well controlled and there is significant thermal variation in the loading resistor, use the optional *RCOEFF* and *LTMP* parameters. Set the *RCOEFF* parameter to the temperature coefficient of load resistance in ohms per degree Celsius. Adjust the *RLOAD* parameter if necessary so that it equals the correct loading resistance at 0 degrees Celsius. Independently measure the load temperature in units of degrees Celsius, and send these readings to the BRIDGE command through the *LTMP* pipe. The BRIDGE command will adjust the value of the loading resistor prior to each conversion.

For each input voltage measurement, the reference voltage on the balancing side of the bridge is equal to the *BALANCE* ratio times the excitation voltage. The voltage on the active side of the bridge equals this reference voltage plus the measured differential voltage. The voltage between the positive source and the measurement point appears across the known loading resistance, so the measurement-side current can be computed. Using the value of this current, and the known voltage on the measurement side of the bridge, the unknown value of resistance can be calculated.

Examples

```
BRIDGE(IPipe2, 5.0, 0.5, 1000.0, R2)
```

Voltage measurements are taken for a bridge configuration in which nominal values of components are used. The voltage at the active divider junction, relative to the balancing network junction, is obtained from the differential input sample channel pipe `IPipe2`. The excitation voltage is a nominal 5.0 volts. The nominal balancing ratio with equal-value balancing resistors is 0.5. The loading resistor is 1K, equal to the nominal operating value of the measured resistance. Default measurement gain of 1.0 is assumed. The computed resistance values are reported in pipe R2.

```
BRIDGE(IP2, 4.959, 0.5025, 1001.8, 0.087, TLOAD, 10.0, R2)
```

The same as the previous configuration, except that all components are calibrated and the loading resistor value is compensated for temperature variations to obtain maximum measurement accuracy. The supply voltage is measured at 4.959 volts. The balancing resistors are not perfectly matched and their gain ratio is 0.5025. The nominal 1K loading resistance is measured at 0 degrees Celsius where it has the value 1001.8 ohms. The loading resistor value is observed to increase by 8.7 ohms over a 100 degree temperature swing, so the temperature coefficient is 0.087 ohms per degrees C. Independent measurements of the operating temperature of the load resistance are provided by pipe `TLOAD`. The measurements use a differential input amplifier gain of 10.0. The results of the resistance calculations are returned in pipe `R2`.

See also:

`DIVIDER`

sensorm: DIVIDER

Convert measurements from a voltage divider network to determine resistance.

DIVIDER(VIN, VS, RLOAD, [RCOEFF, LTMP,] [ANGAIN,] ROUT)

Parameters

<*VIN*>
Input measurements of the junction voltage from a voltage divider network.

FLOAT PIPE

<*VS*>
Nominal or measured excitation voltage driving the divider network.

FLOAT CONSTANT | *FLOAT PIPE*

<*RLOAD*>
Nominal or calibrated load resistance, ohms at 0 degrees C.

FLOAT CONSTANT

<*RCOEFF*>
Temperature coefficient of resistance of load resistor, ohms per degree C.

FLOAT CONSTANT

<*LTMP*>
Input pipe with load temperature measurements, degrees C.

FLOAT PIPE

<*ANGAIN*>
The voltage gain used to measure the analog *VIN* signal.

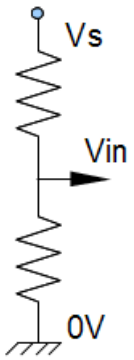
FLOAT CONSTANT

<*ROUT*>
Calculated output resistance measurements, in ohms.

FLOAT PIPE

Description

The DIVIDER command converts the voltage readings from pipe *VIN* to measure an unknown resistance using a voltage divider network configuration. If all measurements are relative to a common ground at 0 volts without offset, single-ended measurements can be used for this data. In the voltage divider network, an unknown device to be measured is connected between the reference voltage and the measurement junction. An accurately known loading resistor connects between that point and an accurately regulated excitation voltage. The calculated results, in ohms, are returned through the *ROUT* pipe, one result per divider measurement input value.



The voltage readings *VIN* are observed at the junction between the measured element and loading resistor. The input divider voltage measurements received through the *VIN* pipe have units of volts. By default, this command assumes that the voltage measurement are obtained using an amplifier gain of 1.0, but if you use a higher gain, specify it as the *ANGAIN* parameter.

If the supply voltage is very well regulated, you can measure it once and specify the value as a constant `VS` parameter. If you use the 5.0 volt regulated voltage of the Data Acquisition Processor for this excitation, you could specify 5.0 volts and omit the calibration measurement. If the excitation is subject to small but potentially relevant variations, make simultaneous voltage measurements of the supply voltage, in units of volts, and provide this second input stream to the DIVIDER command through a *VS* pipe, one supply voltage measurement per divider voltage reading.

Specify the loading resistor value *RLOAD* parameter in units of ohms. Sometimes a nominal resistor value is close enough, but for better results enter an accurately measured value.

Most of the time, variations in operating temperature are insignificant, and the load resistance can be presumed constant. For these cases, omit the *RCOEFF* and *LTMP* parameters. However, if significant temperature variations are anticipated, the DIVIDER command can adjust the effective load resistance to compensate for temperature-dependent changes. Set the *RCOEFF* parameter to the temperature coefficient of load resistance in ohms per degree Celsius. Adjust the *RLOAD* parameter if necessary so that it reports the correct loading resistance for 0 degrees Celsius. Measure the load resistance temperatures in units of degrees Celsius and send these measurements to the DIVIDER command through the *LTMP* pipe, one temperature reading per divider measurement.

For processing each input value, the difference between the known excitation voltage source and the measured divider junction appears across the known loading resistor, allowing the divider current to be computed. This known divider current passes through the unknown element to produce the measured voltage, so its resistance can be computed, producing the results placed into the *ROUT* pipe.

Examples

```
DIVIDER(IPipe2, 5.000, 10000.0, ROUT)
```

Read the voltage at the junction between a load resistance of exactly 10K ohms and a resistive sensor of approximately 10K nominal ohms when a 5.000 volt excitation voltage is applied across the divider network. The voltages across the resistive sensor element are measured with the default gain 1 and received from input sample channel pipe `IPipe2`. The computed resistance values are reported in pipe `ROUT`.

```
DIVIDER(IP2, 1.968, 10008.0, 0.42, PAmb, 4.0, ROUT)
```

This example is very similar to the previous configuration, except calibrated for maximum measurement accuracy. Careful measurement determined that the excitation supply voltage level is a fixed 1.968 volts. The computations use a temperature-adjusted value of load resistance for each conversion. The measured load resistor value is 10008 ohms at zero degrees Celsius. The nominal load resistance is observed to increase by 42 ohms over a 100 degree temperature swing, so a temperature coefficient 0.42 ohms per degrees C is specified. The operating temperature is separately measured, in degrees Celsius, with these measurements provided in the `PAmb` pipe. The measurements of divider voltage are captured using an amplifier gain of 4.0.

See also:

BRIDGE

sensorm: GENPOLY

Convert measurements using a specified, generic low-order polynomial formula.

GENPOLY(VIN, [NORDER,] VCOEFFS, VOUT)

Parameters

<VIN>
Stream of input data values, typically measurements.

FLOAT PIPE

<NORDER>
The order of the polynomial used for the conversion.

WORD CONSTANT

<VCOEFFS>
Coefficients of the polynomial.

FLOAT VECTOR

<VOUT>
Calculated output values in arbitrary units.

FLOAT PIPE

Description

The GENPOLY command is a general purpose mapping from a stream of input data received from pipe *VIN*, to produce a stream of output data in pipe *VOUT*. One common application of this command is mapping measurements of an arbitrary nonlinear sensor into the equivalent measurement in the desired physical units. This can be efficient and very effective for some almost-linear sensor devices.

The polynomial mapping is determined by the *VCOEFFS* vector. The coefficients specified in this vector start with the zero-order term, followed by the first-order term, followed by the second-order term, etc. up to the order specified by the *NORDER* parameter. (The number of coefficients to specify is one larger than *NORDER*.) If you omit the *NORDER* parameter, the command will count terms given in the polynomial coefficient vector and deduce the corresponding polynomial order.

Using only a first-order polynomial, this command becomes equivalent to a SCALE command, in effect, adjusting the effective offset and gain. Or from another point of view, this command covers the functionality of the SCALE command at the same time that it performs nonlinear mappings. When using this command for "linearizing" the response of a nonlinear sensor, it can at the same time provide gain and offset corrections.

Examples

```
VECTOR   VCOEFFS FLOAT  =  ( 0.242, 3.085e-3, -5.707e-7 )
CONSTANT ORDER   WORD   =  2
...
GENPOLY( PIN, ORDER, VCOEFFS, POUT )
```

Take values `X` from the `PIN` pipe one at a time, and for each of these values apply the second-order polynomial mapping

$$Y = 0.242 + 3.085e\text{-}3*X + -5.707e\text{-}7*X^2$$

to obtain the values `Y` that are placed into the output data stream `POUT`.

See also:

SCALE, RTD, THERMPOLY

sensorm: RTD

Convert resistance measurements to the corresponding temperature for an RTD device.

RTD(RIN, VMODEL, TEMPOUT)

Parameters

<*RIN*>
Stream of input resistance values in ohms.

FLOAT PIPE

<*VMODEL*>
Coefficients for polynomial conversion model, standard or adjusted.

FLOAT VECTOR

<*TEMPOUT*>
Stream of output data, the temperatures in degrees Celsius.

FLOAT PIPE

Description

The RTD command converts a stream of resistance values received from the *RIN* pipe, using the device model specified by the *VMODEL* vector, and delivering the corresponding output temperatures in degrees C to the *TEMPOUT* pipe. Some other processing is required to convert the original raw voltage measurements into the observed resistance readings.

Though their temperature response is almost linear, RTD devices need a slightly nonlinear conversion for full accuracy. The device model is a polynomial curve, with the number of terms required dependent on the material type used to fabricate the RTD. Devices have standardized conversion curves. These give very good results, but due to small component tolerances, individual devices don't always match the standardized curves perfectly. For best accuracy, you can fit a polynomial curve to data sets of measured resistance and the corresponding temperatures, thus calibrating your own response curves.

The following conversion characteristic for an RTD device is suitable for wide temperature ranges and a variety of RTD device types.

$$R = R_0 \left(1.0 + c_1 T + c_2 T^2 + c_3 T^3 \ldots \right)$$

The RTD command supports terms up to order 6. The R_0 term corresponds to the nominal resistance of the device at 0 degrees C. For example, a type *JPT200* RTD will have a nominal resistance of 200 ohms, and a perfectly manufactured device would have a base resistance R_0 exactly equal to 200 ohms at 0 degrees C. The zero-order term of the conversion polynomial is always normalized to 1.0, so it is not stored in the *VMODEL* coefficient vector. Instead, the base resistance value R_0 is recorded in that location.

Some curve forms provided by manufacturers and standards bodies will not correspond to the general polynomial form. For these, you will need to adjust the coefficients. An important example is the platinum *PT* device series. It is very common to specify its conversion polynomial in the *Callendar - Van Dusen form*

$$R = R_0 \left(1.0 + a_1 T + a_2 T^2 + a_3 (T-100) T^3 \right)$$

There is nothing mathematically unique about that particular formulation; it remains a third-order polynomial. Formulas to convert the manufacturer's coefficients to the general polynomial form are:

```
c1  =  a1
c2  =  a2
c3  =  -100.0 a3
c4  =  a3
```

There are some additional peculiarities about RTD devices.

- The IEC751 standard specifies a different polynomial for temperatures below 0 degrees C and above 0 degrees C for the popular platinum RTD devices. To allow for this, and also to allow you to calibrate your own piecewise polynomial curves, the `RTD` command allows multiple polynomial curve sections. The nominal RTD resistance is the same for all sections. In the second, third, subsequent pieces, the *zero-order term* location is used to store the temperature level above which the evaluation scheme switches to that piece of the composite curve. The additional 6 coefficient terms for that piece follow the temperature break-point term.
- You want to know the temperature given the resistance. However, the conventional RTD models specify this backwards, the resistance given the temperature. Thus, the model is the inverse of the kind of model used in the `GENPOLY` command. Since this inverse problem is slightly nonlinear, a numerical solution method must be used to solve it.

The following table provides coefficients that you can use for typical device types.

Device type	**c1**	**c2**	**c3**	**c4**	**c5**	**c6**
Platinum (PT family, IEC751)						
above 0 degrees C	3.9083e-3	-5.7750e-7	0.0	0.0	0.0	0.0
below 0 degrees C	3.9083e-3	-5.7750e-7	4.183e-10	-4.183e-12	0.0	0.0
Platinum (JPT family, RC-4, SAMA)	3.9787e-3	-5.8686e-7	4.167e-10	-4.167e-12	0.0	0.0
Nickel (limited range)	5.485e-3	6.6650e-6	0.0	2.805e-11	0.0	0.0
Copper (limited range)	4.270e-3	0.0	0.0	0.0	0.0	0.0

To summarize, in the layout of the device characteristic vector, the terms are specified in the following sequence.

1. *Initial conversion curve section*
    1. Base resistance at 0 degrees C
    2. Coefficients `c1, c2, c3, c4, c5, c6`
       for first piece of conversion curve

2. *Additional sections of the curve, as needed*
    1. Temperature above which this curve section applies
    2. Coefficients `c1, c2, c3, c4, c5, c6`
       for this piece of conversion curve

Even for devices operating over a large temperature range, conversion errors less than ±0.1 degrees C are still possible.

When the operating temperature range is within interval T=0 to T=100 degrees Celsius, and maximum accuracy is not required, the simplified *"alpha coefficient"* model is often used. The *alpha coefficient* value is the slope of a straight line that matches the nonlinear RTD conversion curve at the points T=0 and T=100. If you substitute the alpha coefficient for the c1 coefficient in the conversion data, and set all of the other coefficients to zero, the conversion is linear, and therefore very fast. With platinum RTD devices, the maximum conversion error remains only a fraction of a degree, near the center of the range. This is often good enough.

RTD device resistance is often measured in:

- a voltage divider configuration – because this is easy to measure.
- a bridge configuration, because this allows a differential voltage reading, using gain, for better measurement precision.

The `DIVIDER` and `BRIDGE` commands can be helpful for measuring the RTD device resistance.

Examples

```
VECTOR  vPT100POS  FLOAT = ( 100.0,
    3.9083e-3,   -5.7750e-7,  0.0,  0.0,
    0.0,  0.0 )
...
RTD(PRESIST,vPT100POS,TEMPR)
```

A type *PT100* RTD is used to measure temperatures. The temperatures never go below 0 degrees C, so the characteristic terms for the sub-zero piece of the standard conversion curve are omitted. Otherwise, the conversion curve specified in vector `vPT100POS` matches the IEC751 standard for this device type. The nominal 100 ohms base resistance at 0 degrees C is specified in the configuration. When the RTD command executes, measurements of resistance in ohms are obtained from pipe `PRESIST`. Each resistance value is converted to the corresponding operating temperature of the RTD device by solving the inverse of the device characteristic determined from the data in the `vPT100POS` model. The resulting temperatures in degrees C are placed into pipe `TEMPR`.

```
  VECTOR  vPT100  FLOAT = (   99.86,
    3.9083e-3,   -5.7750e-7,   4.1830e-10,  -4.1830e-12,
    0.0,  0.0,
0.00,
    3.9083e-3,   -5.7750e-7,   0.0,         0.0,
    0.0,  0.0 )
...
RTD(PRESIST,vPT100,TEMPR)
```

The same example as before. except that now temperatures can range both below and above 0 degrees C, and extra effort is applied to retain as much measurement accuracy as possible. The resistance of the RTD device is calibrated by measuring accurately at 0 degrees C, and the observed value 99.86 ohms is specified as the base resistance. The first set of polynomial terms conforms to the the IEC751 standard for temperatures below 0.00 degrees. The second set of polynomial terms conforms to the the IEC751 standard for temperatures above 0.00 degrees. For temperatures above 0.00 degrees, the conversion processing switches over to the second set of standard conversion terms.

See also:

DIVIDER, BRIDGE

sensorm: THERMISTOR

Convert resistance measurements to the corresponding temperatures for a thermistor device.

THERMISTOR(RIN, VMODEL, TEMPOUT)

Parameters

<*RIN*>
Stream of input resistance values in ohms.

FLOAT PIPE

<*VMODEL*>
Coefficients for Steinhart - Hart thermistor model .

FLOAT VECTOR

<*TEMPOUT*>
Stream of output data, the temperatures in degrees Celsius.

FLOAT PIPE

Description

The THERMISTOR command converts resistance values received from the *RIN* pipe, using the device model specified by the *VMODEL* vector, and delivering the corresponding output temperatures in degrees C to the *TEMPOUT* pipe. This can be used with negative temperature coefficient thermistor devices of the sort commonly used for temperature measurements.

The device model used is the *Steinhart-Hart equation*. This equation is an empirical formula capable of conversion accuracy within a small fraction of a degree. There are no standardized conversion curves, but most thermistor manufacturers will provide Steinhart-Hart coefficient values that work over a suitable range with their devices.

The form of the *Steinhart - Hart equation* is

$$T = 1.0 / (a + b \ln(R) + c \ln^3(R))$$

where R is the measured device resistance in ohms, the a, b, and c terms are the model coefficients, and the ln() function is the natural logarithm. The standard result is an absolute temperature in Kelvins. The THERMISTOR command uses a slight variation of this

$$T = 1.0 / (a + b \ln(R) + c \ln^3(R)) - 273.16$$

where the final 273.16 term converts the temperature units from Kelvins to degrees Celsius.

For maximum conversion accuracy, you can calibrate the curve for the individual device you are using. Measure the actual resistance at well-selected temperature points representative of your operating range, insert these values into the Steinhart-Hart equation form, and solve for the adjusted coefficient values.

To support a very wide temperature range, the THERMISTOR command supports a multiple-piece device model. Each piece of the model is encoded into the VMODEL parameter as a break-point temperature in degrees C, followed by the three Steinhart-Hart coefficients to use at that temperature and beyond. The coefficients are specified in the order a, b, c. The model pieces are encoded in order from lowest temperature range to highest

temperature range. The first piece is the default that provides conversions at low temperatures, so its break-point term is ignored and can be set to zero. Almost all applications will use a one-piece model.

A thermistor device typically will produce very large resistance variations, with widely varying measurement resolution through the operating range. The DIVIDER command can be very helpful for producing an even measurement resolution through the measurement range. A suitable loading resistor will typically have a resistance value close to the nominal ambient- temperature resistance of the thermistor device..

Examples

```
VECTOR  THERM44007 FLOAT = ( 0.0, 1.285E-3, 2.362E-4, 9.285E-8 )
...
THERMISTOR(PRESIST, THERM44007, TEMPR4)
```

Manufacturer-provided coefficients for a model 44007 thermistor are used for thermistor temperature conversions, with the model terms specified in the vector `THERM44007`. Because this is a one-piece model, the temperature breakpoint between pieces is unused, and the first term is set to 0. Measurements of the temperature-dependent thermistor device resistance in ohms are obtained from pipe `PRESIST`. Each resistance value is converted to the corresponding operating temperature of the thermistor using the Steinhart-Hart equation, using the coefficients from the `THERM44007` vector, yielding temperature results in degrees C. The temperature results are placed into pipe `TEMPR4`.

See also:

DIVIDER

sensorm: THERMOPOLY

Apply calibrated temperature conversions over a limited range for thermocouples.

THERMOPOLY(VIN, CJT, [NORDER,] VCOEFFS, TEMPC)

Parameters

<*VIN*>
Stream of input thermocouple potential measurements.

WORD PIPE | FLOAT PIPE

<*CJT*>
Stream of cold junction temperature measurements, typically in degrees C

FLOAT PIPE

<*NORDER*>
Order of the conversion polynomial function.

WORD CONSTANT

<*VCOEFFS*>
Coefficients of the conversion polynomial.

FLOAT VECTOR

<*TEMPC*>
Stream of output temperatures, typically in degrees Celsius.

FLOAT PIPE

Description

The THERMOPOLY command provides calibrated conversions of thermocouple potential readings into temperature measurements, for improved accuracy over a restricted range. This is an alternative to the THERMO command provided by the DAPL system, which uses "standard" conversion curves only. Input voltage readings of the temperature difference from the thermocouple "hot junction" to the reference "cold junction" are received from pipe *VIN*. Any appropriate units can be used for these input values, but they must be consistent with the polynomial characteristic that is provided. Thermocouples measure a temperature difference, not an absolute temperature, so independent measurements of the "cold junction temperature" must be provided in pipe *CJT*, one cold junction reading for each thermocouple potential reading. Temperature units for the CJC temperature reading are conventionally in degrees C, but in all cases, the temperature units must be the same as the units for the final output data. Thermocouple characteristics are somewhat nonlinear and dependent on the particular temperature chosen for the "cold junction" reference, so the operating temperatures for the cold junction should not be allowed to vary much from the reference temperature used for the device calibration.

The polynomial characteristic used for the conversions is defined by vector VCOEFFS. The conversion is from the units are used for the input potential measurements into the units selected for the final output temperature results. Data scaling might help to avoid pathological coefficient values in the conversion function coefficients. The conversion polynomial is evaluated for each input value to determine the temperature drop across the length of the thermocouple wires. The converted measurements are combined with the cold junction reading, and the resulting absolute temperatures in degrees C are placed into the TEMPC pipe.

The most common method for determining the temperature difference from the thermocouple potential reading is to apply published conversion curves, as determined by standards organizations under careful laboratory conditions. Manufacturers do their best to produce devices match these standard curves, but the match is never perfect. The

approach of applying standard curves is used in the DAPL system's THERMO processing command. As reported in NIST studies, applying standard curves is unlikely to produce net measurement errors much better than ±2 degrees C, even under the best of conditions, due to expected device variations.

Despite the variations in their characteristics, individual thermocouple devices tend to produce results that are very repeatable when operating consistently over a limited range, and under these conditions the THERMOPOLY command can be effective. When calibrated for one individual thermocouple device, the net measurement accuracy might be better than ±1 degrees C. This approach is *unlikely to be effective* over a wide operating temperature range, however, because the polynomial that he THERMOPOLY command uses for its conversions maps from voltage to temperature, while the "standard" curves map from temperature to voltage.

The coefficients for the polynomial conversion formula are defined in the VCOEFFS vector. The coefficients start with the zero-order term, followed by the first-order term, followed by the second-order term, and so on, up to the order specified by the NORDER parameter. If you omit the NORDER parameter, the command will count terms and assume the corresponding polynomial order. A first-order polynomial provides the equivalent of offset and gain corrections using the SCALE command, and it will not be very accurate. A second-order polynomial is sufficient over a lesser range, perhaps 100 degrees C or so. A third-order to fourth-order polynomial is typically sufficient to represent an interval of two or thee hundred degrees C.

The measured thermocouple potential is very small, so it must be amplified to obtain a signal level large enough to digitize. Small offset or gain errors are amplified along with the signal and can have a significant effect. The polynomial of the THERMOPOLY command can account for gain and offset effects at the same time that it corrects for nonlinearity. It can also cover any required scaling changes related to converting the raw A/D digital readings to physical units.

So-called *cold junction compensation* is a common technique for using thermocouples with terminals at an operating temperature displaced a small known amount from the reference cold junction temperature used for the calibration (typically 0 degrees C) . Given the measurement of the actual thermocouple terminal temperature from pipe ***PCJC***, the calibrated thermocouple characteristic is used in reverse, to accurately estimate the potential difference that would occur between this point and terminals that existed at the reference temperature . The voltage as measured at the actual terminals is then adjusted by this amount. The calibrated characteristic then can be applied as if the terminals were at the reference temperature. If the cold junction temperature is not measured, provide a stream of constant nominal ambient temperature values. The temperature measurement will be no more accurate than the error present in the cold junction estimates.

Examples

```
VECTOR   VCF FLOAT  =  ( -0.01897, 25.41881, -0.42456, 0.04368 )
 ...
THERMOPOLY( PKIN, PCJC, VCF, PTOUT )
```

Use a fast third-order approximation for a type K thermocouple to measure water and high-pressure steam temperatures (within the range 0 to 200 degrees C). Take junction potential measurements captured (with a gain of 1000) from pipe PKIN. Read the corresponding ambient temperature values from the PCJC pipe and determine the appropriate adjustment to the measured thermocouple potential. The adjusted thermocouple potential is converted to a temperature reading using the third-order polynomial function defined by coefficient vector VCF, producing the output temperature measurements placed into pipe PTOUT.

See also:

SCALE, GENPOLY, THERMO

--- END OF DOCUMENT ---