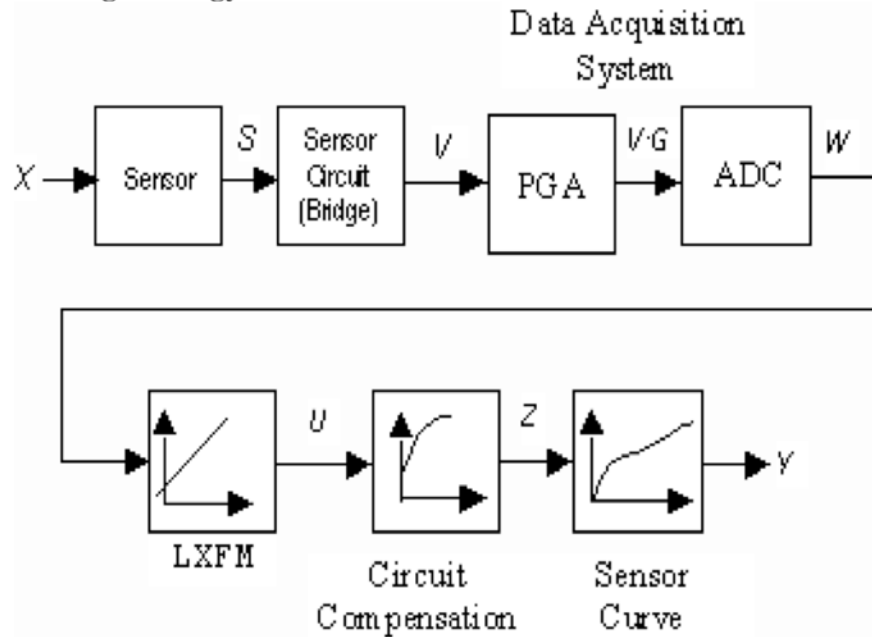


The function of a data acquisition system (DAS) is to recover the value of the sensed quantities. Each stage in the acquisition of the sensed quantities has to be “undone” to recover the original sensed quantity. This article presents the basic scheme for processing digitized sensor data.

**Acquisition and Processing Strategy**



Channel data is processed as shown in the block diagram. Each block represents a function (or "curve"), with input and output quantity. The channel input quantity to be measured,  $X$ , is transduced by the sensor into a sensor electrical quantity,  $S$ , such as resistance or voltage. The sensor circuit (such as a bridge) responds to the sensor and produces a voltage,  $V$ . Then the data-acquisition subsystem converts this voltage to a digital count,  $W$ .

The sensor, sensor circuit, and DAS together have offset and gain (slope) error. If these functions are all linear, this error can be corrected by the next block,  $U(W)$  (or LXFM). It uses the channel calibration parameters to transform the raw-data count,  $W$ , to  $U = X$  so that  $U$  represents the same value as  $X$ . In other words,  $U(W)$  inverts or "undoes" or linearly "compensates" for the combined linear error of sensor, sensor circuit and DAS. Mathematically, it is set to be:

$$U(W) = W^{-1}(X) = X(W)$$

(In mathematical function notation,  $F(G)$ ,  $F$  is a function of  $G$ ; that is,  $F$  is the output and is dependent upon  $G$  as the input.)

$U(W)$  is a linear transform and cannot compensate for nonlinear sensors or sensor circuits; it can only invert *linear* functions. In the case where both sensor and circuit are nonlinear,  $U(W)$  corrects only the DAS error (which is linear), and  $U(W)$  is set to equal  $V(W)$ .

For nonlinear sensors or circuits, the first nonlinear compensation block,  $Z(U)$ , compensates for the nonlinearity of the sensor circuit, and is set to  $S(V)$ . The last block compensates for the nonlinearity of the sensor and is based on manufacturers' sensor curves,  $Y(Z)$ .

For example, consider a single RTD temperature sensor in a bridge circuit.  $X$  is temperature, measured in °C. The RTD changes resistance with temperature, producing  $R(X)$ . The bridge circuit voltage changes nonlinearly with  $R(X)$  to produce an output voltage,  $V(R)$  according to the voltage-divider formula:

$$V = \frac{R / R_{br}}{1 + R / R_{br}} \cdot V_{br} = B(R) \cdot V_{br}$$

where,  $R_{br}$  is the divider upper resistance,  $R$  is the sensor (lower) resistance and  $V_{br}$  is the bridge voltage.

The DAS digitizes this voltage and outputs a digital code,  $W(V)$ . Then  $U(W)$  compensates for the DAS so that  $U = V$  in digital form. Next,  $U$  is applied to the first nonlinear function block,  $R(U)$ . It converts the count,  $U$ , into the resistance of the RTD using the divider formula above. (Because of the large dynamic range of the divider formula, floating-point math is usually used, and the resistance is returned as a floating-point value.)

The next nonlinear block converts resistance into temperature, in °C, and is the function  $T(R)$ . This is the sensor curve given by manufacturers. To summarize the example, the goal is to achieve  $Y = X$  by letting:

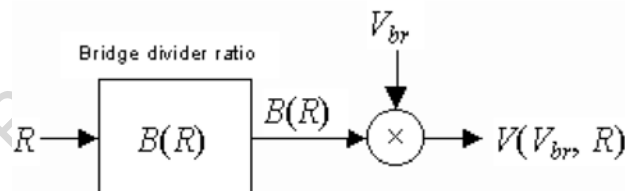
$$U(W) = W(V) \text{ and } Y(U) = X(V) = X(S(V))$$

### Bridge Voltage Compensation

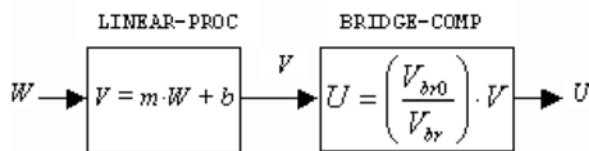
Sensor bridge-circuit sensitivity is proportional to bridge voltage. The bridge output voltage,  $V(V_{br}, R)$  depends on both the sensor output quantity (which is resistance),  $S = R$ , and the bridge voltage,  $V_{br}$ . It can be expressed as:

$$V(V_{br}, R) = V_{br} \times B(R)$$

This function can be diagrammed as follows.



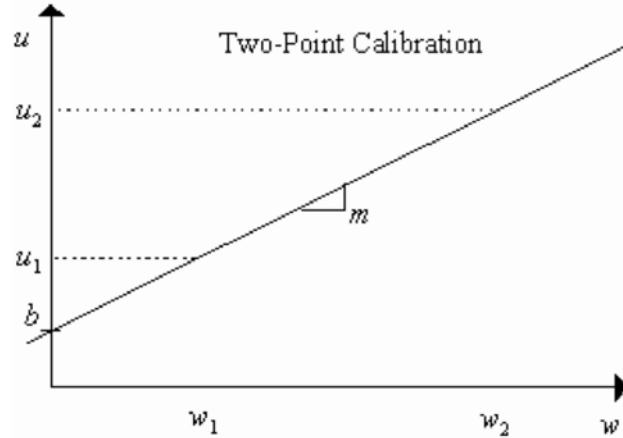
The "X" multiplier block is an additional block that is compensated by breaking  $U(W)$  into two blocks:



$U(W)$  compensates for  $W(B)$  so that  $U = B$ . In addition,  $U(V)$ , done by a bridge compensation routine, compensates for voltage drift in the bridge from its value at calibration.  $V_{br}$  is measured through the bridge-voltage channel. Its last measured value and its value at calibration time,  $V_{br0}$ , are used to form a scaling factor for compensation.

## Linear Calibration

*Two-point calibration* assumes that the sensors and DAS gains are linear. (For example, if the force sensor gain of 1 V/kN changes with the amount of applied force, it is nonlinear.) To calibrate a linear system, mC programming constructs a line through two acquired data points, as shown below. The  $w$ -axis is the raw data, the digitized count from the ADC. The  $u$ -axis is the processed data, in given units. Two measurements are made with known input values  $u_1$  and  $u_2$ . The ADC outputs raw-data counts as  $w_1$  and  $w_2$ .



We then have two equations for which slope,  $m$ , and offset,  $b$ , are solved:

$$\begin{cases} u_1 = m \cdot w_1 + b \\ u_2 = m \cdot w_2 + b \end{cases}$$

This results in the two-point calibration formulas:

$$m = \frac{u_2 - u_1}{w_2 - w_1}, \quad b = \begin{cases} u_2 - m \cdot w_2 \\ u_1 - m \cdot w_1 \end{cases}$$

The parameters,  $m$  and  $b$ , are used to transform raw-data values into processed values in the units used during calibration. For  $b$ , either right-hand-side expression can be used.

For *one-point calibration*,  $b = 0$  (no offset error is assumed), and the second data point, by default, is the origin, at  $(0, 0)$ ; only slope is calibrated.