

# Guardbanding Using Automated Calibration Software

Speaker: **Lars Andersson**/ Author: Matt Nicholas

Fluke Corporation

P.O. Box 9090

Everett, WA 98206-9090

Phone: (425) 347-6100

## Abstract

The ISO 17025 standard (“General Requirements for the Competence of Testing and Calibration Laboratories”) requires that measurement uncertainty be taken into account when statements of compliance are made. In other words, when a calibration laboratory calibrates an instrument and produces a calibration certificate indicating that the calibration verification procedure “passed” or “failed”, it is important that, for each test point, the measurement uncertainty be first calculated and then used in the determination of the test result. Guardbanding is a primary technique for assuring compliance with this 17025 requirement. Considerations of efficiency and productivity in calibration laboratories require full automation whenever possible. It is therefore desirable for automated calibration software to include guardbanding capability. This paper describes a flexible, configurable implementation of guardbanding in an automated calibration software system. A number of techniques are discussed, including both table-based and formula-based methods. Facilities for customization of the guardbanding algorithm are presented. A description of generated result data is included.

## 1. Introduction

The fundamental concept of guardbanding is to restrict the *Pass/Fail* limits applied to a calibration test based on a defined criterion.

The purpose of guardbanding is to control the risk of accepting an out-of-tolerance unit, or rejecting an in-tolerance unit.

Guardbanding affects the result of a calibration test. Without guardbanding the result of a test will be *Pass* or *Fail*. With guardbanding the result of a test will be *Pass*, *Fail*, or *Indeterminate*. Furthermore, a *Pass* or *Fail* test result without guardbanding may change to a result of *Indeterminate* with guardbanding.

### *Example*

Suppose a calibration test with the following parameters is performed:

- Cardinal Point = 10 V DC
- Test Specification =  $\pm 1\%$
- Specification Limits = 9.9 V to 10.1 V

- Measurement Uncertainty = 0.03 V
- Measured Value (UUT Reading) = 9.92 V

Notice that in this example the measured value is within the specification limits. If a *Pass/Fail* determination were made without considering the measurement uncertainty, the result of the test would be *Pass*. However, when the measurement uncertainty is considered the true value becomes  $9.92 \pm 0.03$  V. The true value may be as low as 9.89 V, which is below the lower specification limit.

When the measured value is close to a specification limit, the result is said to be *Indeterminate*. The precise definition of “close to” depends on the chosen guardbanding strategy.

## 2. Motivation

The primary motivation for guardbanding is derived from the standards document ISO 17025. In Section 5.10.4.2 the standard states:

*When statements of compliance are made, the uncertainty of the measurement shall be taken into account.*

A plausible interpretation of the 17025 statement is that when a calibration report is produced, and the report makes a specific statement as to the *Pass/Fail* result of the calibration, either for the overall calibration or for individual tests, the *Pass/Fail* determination must be made in a manner which considers the measurement uncertainty.

If a calibration report is produced which simply itemizes a series of tests, reporting for each test the reference value, the UUT value, the nominal test specification, and the measurement uncertainty, but does not report a *Pass/Fail* result, then it's reasonable to interpret 17025 as not requiring that the measurement uncertainty be *taken into account* because, in this case, no statement of compliance is being made.

## 3. Strategies

Guardbanding strategies range from *very conservative* to *no guardbanding*.

The most conservative strategy is to tighten the specification limits by the full amount of the measurement uncertainty.

The least conservative method is to do nothing. The specification limits are used *as is*. This approach is traditional, but takes no account of the measurement uncertainty. MIL STD 45662, which requires a 4:1 Test Specification Ratio (TSR), uses this approach. This approach is relatively easy to implement, and the risk of passing an out-of-tolerance UUT is usually considered to be acceptable if 4:1 TSRs can be maintained. However, in many areas of calibration it is difficult to maintain 4:1 TSRs.

Strategies which fall between the most and least conservative approaches involve dynamically adjusting the test limits based on various criteria.

Strategies may be broadly classified as follows:

- Strategies which adjust the test limits as a function of the measurement uncertainty.
- Strategies which adjust the test limits as a function of the test uncertainty ratio (TUR).
- Strategies which adjust the test limits as a function of the test specification ratio (TSR).
- Strategies that adjust the test limits based on an algorithm which directly attempts to maintain constant risk of accepting an out-of-tolerance UUT.
- Strategies that statistically analyze historical calibration data for the UUT and/or references used in the calibration process.
- Strategies that statistically analyze population data for references and UUTs.

Strategies that analyze historical calibration data require access to a database containing the required data. Such access may be impractical in some situations. For example, a 3<sup>rd</sup>-party calibration laboratory may not have the required historical calibration data. Similarly, strategies that require access to data for a population of instruments are typically impractical except for the instrument manufacturer. Although strategies based either on historical calibration data or instrument population data may have merit, they are beyond the scope of this paper.

## 4. Automation

In addition to categorizing guardbanding according to the particular strategy or algorithm used, it is also possible to categorize guardbanding based on the level of automation.

### 4.1 Manual Guardbanding

When guardbanding is performed manually, software to automate the calibration is not used. Guardbanded test limits must be calculated prior to the calibration. Software applications like Excel may be used to assist in the calculation of the guardbanded limits. When the calibration is performed it is the responsibility of the operator to determine the result of each test as *Pass*, *Fail*, or *Indeterminate* based on the calculated guardbanded limits. Results are then recorded, either on paper, or manually entered into a calibration data collection software system.

### 4.2 Semi-Automated Guardbanding

Semi-automated guardbanding involves using software designed for automated calibration, but without any facilities specifically designed to automated guardbanding.

Test limits encoded in a calibration procedure may be adjusted according to a specified guardbanding criterion.

For example, suppose a particular calibration test requires a voltage source which generates 10 mV at 10 kHz. Suppose further that the UUT's test specification at this test point is  $\pm 0.01\%$ . A calibration procedure language must provide a way for the procedure writer to encode the test specification, either in the procedure itself, or in an external data source (e.g., a database table). The procedure writer may decide, then, based on a particular guardbanding algorithm to tighten the test specification by 20%. To accomplish this it is necessary to change the test specification from  $\pm 0.01\%$  to  $\pm 0.008\%$ .

The virtue of this approach is that once a particular procedure has been updated to use guardbanded limits, the procedure may be re-used. Full automation with guardbanding is achieved this way. The disadvantages of this approach, however, are significant:

- It requires a significant effort to update the calibration procedure.
- The guardbanding may be static, rather than dynamic. For example, it may be difficult to vary the guardbanding factor based on the calculated measurement uncertainty.
- The reporting system may not be able to easily distinguish between a calibration performed with guardbanding and one performed without guardbanding.

Some calibration software packages have sufficient flexibility that it is possible to overcome some of the disadvantages listed above. For example, the procedure writer may be able to directly take responsibility for calculating the guardbanded test limits in a dynamic way. The calibration procedure writer may also be able to generate result data containing information required to allow the reporting system to properly distinguish between guardbanded and non-guardbanded calibrations. However, even if a software package has the required, low-level flexibility to support guardbanding in this way, it is nevertheless a task which requires significant effort on the part of the procedure writer, and also complicates the calibration procedure, which may make maintenance more difficult. Furthermore, this approach entails *hardwiring* the guardbanding algorithm into the calibration procedure itself, which precludes allowing the end user to modify the guardbanding approach without changing the calibration procedure.

It is also worth noting in this context that some calibration software package implement facilities which, while not originally developed to support guardbanding, may be adaptable to a guardbanding purpose. For example, a calibration software package may include the ability to annotate a particular test as a *marginal pass*, or as *significantly out of tolerance*, or as an *accredited calibration*.

### 4.3 Automated Guardbanding

To improve upon the manual and semi-automated guardbanding methods discussed above it is necessary to fully automate the guardbanding process. A satisfactory automated guardbanding solution should implement the following features:

- The calculation of the guardbanded test limits should be automatic.
- The procedure writer should be able to choose from a variety of built-in guardbanding algorithms.
- It should be possible for the procedure writer to enable or disable guardbanding.
- Result data sufficient to satisfy an auditor's requirements should be saved in the calibration database.
- It should be easy to generate a calibration report which distinguishes between guardbanded and non-guardbanded tests, and which shows both the guardbanded and nominal specification limits for each guardbanded test.
- A procedure writer should be able to take an existing calibration procedure which was not written with guardbanding in mind and easily change it to a procedure for which guardbanding is enabled.
- The software should be flexible and customizable. The guardbanding method and other guardbanding parameters should be configurable on a per-test, per-procedure, per-workstation, or per-site basis.
- It should be possible to define guardbanding algorithms which differ from the built-in methods provided by the software application.

The following sections describe a number of built-in guardbanding methods.

#### 4.3.1 TSR-Based Methods

*TSR*-based guardbanding methods are methods that use the *Test Specification Ratio (TSR)* as the independent parameter in the determination of the guardbanding factor.

The *TSR* is defined as:

$$TSR = (\text{Test Specification}) / (\text{Reference Specification})$$

Note that nomenclature has occasionally been inconsistent. *TSR* is sometimes referred to as *TAR* (Test Accuracy Ratio), and sometimes as *TUR* (Test Uncertainty Ratio). In this paper *TUR* refers to a ratio which involves the calculated measurement uncertainty, not simply the reference accuracy. *TSR*, as used here, is equivalent to *TAR*.

As a specific example, consider the *TSR-based* method defined by the following rules:

- If the *TSR* is greater than 4.0, use 100% of the specification limits. (The guardbanding factor is 1.0).
- If the *TSR* is between 1.5 and 4.0, use 80% of the specification limits. (The guardbanding factor is 0.8).

If the *TSR* is less than 1.5, the validity of the test may be called into question because the reference accuracy and the UUT accuracy are too close. However, a particular organization may wish to define a guardbanding factor which applies when the *TSR* is between 1.0 and 1.5.

The method described in the example above is easy to describe and relatively easy to implement. It does not require that the software have the ability to compute the measurement uncertainty, only that it have access to the reference accuracy and the nominal test specification of the UUT.

To make the method presented in the example above more flexible the automated calibration software should:

- Allow the procedure writer or site administrator to configure the *TSR* thresholds. In the example above the thresholds are 1.5 and 4.0. It should be possible to specify any reasonable *TSR* value as a threshold.
- Allow the guardbanding factors to be configured by the procedure writer or site administrator. In the example above the guardbanding factors are 0.8 and 1.0. The software should allow any reasonable values to be specified as guardbanding factors.
- As a further generalization the software should allow a user-specified table of *TSR* values with a corresponding guardbanding factor associated with each *TSR* value. Any reasonable table size should be allowed. Allowing an arbitrary table in this fashion provides the user with a method to implement, in effect, a piece-wise linearization of any guardbanding function desired, as long as the function is defined as having one independent parameter (the *TSR*) and one dependent parameter (the guardbanding factor).
- In addition, the software should also provide a configurable flag which allows the user to determine whether or not to linearly interpolate between points in the table or interpret the table as a step function, choosing the more conservative guardband factor for values which fall between points in the table.

#### 4.3.2. TUR-Based Methods

A *TUR*-based guardbanding method is similar to a *TSR*-based method, but uses as the independent variable the *Test Uncertainty Ratio* (*TUR*) rather than the *Test Specification Ratio*.

The *TUR* is defined as:

$$\text{TUR} = (\text{Test Specification}) / (\text{Measurement Uncertainty})$$

The *Test Specification* here is not normalized. The value stated by the manufacturer of the UUT is used directly. The *Measurement Uncertainty* here is the expanded measurement uncertainty, not the standard (1-sigma) uncertainty.

All of the configurable options described in the section on *TSR*-based methods (arbitrary table size, user-settable thresholds, user-settable guardband factors, and an interpolation flag) should be available with a *TUR*-based method as well.

Although a *TUR*-based method is essentially similar to a *TSR*-based method when viewed with respect to the guardbanding algorithm per se, *TUR*-based methods are nevertheless considerably more difficult to implement because the software must have the ability to calculate the measurement uncertainty for each calibration test.

A GUM-compliant measurement uncertainty calculation imposes significant additional requirements on the automated calibration software:

- The software must be able to automatically repeat each test a specified number of times, accumulating the measurements in order to calculate the standard deviation and include an uncertainty component based on the standard deviation.
- The software must be able to determine the resolution of the unit under test (UUT) in order to include an uncertainty component based on the UUT resolution.
- The software must allow the procedure writer or user to specify values of optional uncertainty components, and then calculate the standard uncertainty based on the reference accuracy (or accuracies if a test involves multiple references), the UUT resolution, the standard deviation of the measurements, and all specified optional uncertainty components.
- The software must implement the Welch-Satterthwaite approximation in order to determine the expanded measurement uncertainty. This in turn requires that the software allow the procedure writer or end user to specify various uncertainty-related parameters like degrees of freedom values, sensitivity coefficients, and confidence values.

If the calibration software has the capability to determine the measurement uncertainty as described above, implementing a *TUR*-based guardbanding method becomes simply a variation of implementing a *TSR*-based method.

#### 4.3.3. Normalized *TUR*-Based Methods

A guardbanding method based on the normalized *TUR* is the same as the method based on the non-normalized *TUR*, as described in the preceding section, except that the independent variable is the normalized *TUR*.

The Normalized *TUR* is defined as:

$$\text{NTUR} = (\text{Normalized Test Specification}) / (\text{Standard Uncertainty})$$

The standard uncertainty is available because the calculation of the expanded uncertainty involves first determining the standard uncertainty, then multiplying it by an appropriate coverage factor to determine the expanded uncertainty.

Determining the normalized test specification requires that the procedure writer be able to indicate the confidence associated with the UUT test specification, in order that the software has the information required to normalize the test specification to a 1-sigma value. In the absence of a direct specification of the confidence associated with the UUT specification the software application should use a default value of 2 sigma.

#### 4.3.4. Measurement Uncertainty-Based Methods

A measurement uncertainty-based guardbanding method is a method that directly uses the expanded measurement uncertainty as the independent variable in the algorithm which determines the guardbanding factor.

Let:

$GUL$  = guardbanded upper limit  
 $GLL$  = guardbanded lower limit  
 $EU$  = expanded measurement uncertainty  
 $GBF$  = guardbanding factor

Then, using a measurement uncertainty-based guardbanding method:

$GUL = (\text{Upper Specification Limit}) - (GBF * EU)$   
 $GLL = (\text{Lower Specification Limit}) + (GBF * EU)$

The calibration software should allow the  $GBF$  (guardbanding factor) to be specified on a per-site, per-workstation, per-procedure, or per-test basis in a hierarchical fashion so that a more restrictive specification overrides a more general specification.

Setting the  $GBF$  to 1.0 effectively implements the conservative approach of tightening the nominal test specifications by the full amount of the calculated expanded measurement uncertainty. This approach was articulated in the *Guide 25 Draft 5* recommendation, a precursor to the *17025* standard. The *17025* standard itself, however, does not directly state a requirement to implement the conservative approach of using a measurement uncertainty-based method with the  $GBF$  set to 1.0. Rather, the requirement is stated in a more flexible way, allowing the metrologist discretion in determining an appropriate guardbanding method. Using a measurement uncertainty-based guardbanding method with a  $GBF$  of less than 1.0 may require additional justification based on an analysis of the UUTs being calibrated and the references being used to calibrate them.

#### 4.3.5. Result Data

Regardless of the guardbanding method selected, the software application should save result data in the calibration database sufficient to satisfy audit requirements and facilitate report generation.

Result data should include at least:

- Specification Limits
- Guardbanding Enable/Disable Flag
- Guardbanding Method
- Guardbanded Test Limits
- Guardband Factor
- Test Result (*Pass*, *Fail*, or *Indeterminate*)
- Overall Result of Calibration (*Pass*, *Fail*, or *Indeterminate*)

An ambiguity arises in connection with the determination of the overall result of the calibration procedure when one or more individual test results are *Indeterminate*.

Without guardbanding the overall procedure result is considered to be a *Pass* only if all individual tests within the procedure have a test result of *Pass*. However, with guardbanding the situation is less clear. Three approaches may be identified:

- In the first approach an individual test result of *Indeterminate* is regarded as equivalent to *Pass* in the determination of the overall test result.
- In the second approach an individual test result of *Indeterminate* is regarded as equivalent to *Fail* in the determination of the overall test result.
- In the third approach an individual test result of *Indeterminate* is regarded as equivalent to neither *Pass* nor *Fail*. This approach leads to the situation in which the overall result of the calibration procedure may be reported as *Indeterminate*.

Since user requirements may differ with respect to determining which of the three approaches to handling *Indeterminate* results is best at a particular site, the calibration software application should be configurable in this respect.

#### *Example*

Suppose a calibration verification procedure with 10 distinct tests is executed, and suppose the test results are 6 *Passes* and 4 *Indeterminates*.

With the first approach the overall result will be *Pass*, because the 4 *Indeterminate* results will be regarded as equivalent to *Pass*.

Using the second approach the overall result will be *Fail*, because the 4 *Indeterminate* results will be regarded as equivalent to *Fail*.

Finally, using the third approach the overall result will be *Indeterminate* because the 4 *Indeterminate* results are grouped with neither the *Pass* nor the *Fail* results.

#### 4.3.6. Determination of the Test Result

As mentioned above, guardbanding affects the result of a calibration test. This section describes in some detail the typical method by which an automated calibration software application determines the test result, then shows how the method is modified to accommodate guardbanding.

The software first calculates the *Percent Error*, the error as a percentage of the test tolerance. The software then determines the test result by comparing *Percent Error* to 100%.

If *Percent Error* is less than 100% the test result is *Pass*.

If *Percent Error* is greater than 100% the test result is *Fail*.

If *Percent Error* is exactly equal to 100% the test result is, by default, *Pass*, but the system should be configurable in this respect.

In order to determine *Percent Error* the software must first determine *Error* and *Test Tolerance*.

#### Determining the *Error*

Every calibration test compares two values, the *UUT Value* and the *Reference Value*.

*Error* is a signed value that represents the difference between the two values being compared. The sign is determined by convention.

*Error* is defined as:

$$\text{Error} = \text{Measurement} - \text{Expected Value}$$

Depending on the type of the test either:

CASE 1: *Measurement* is *UUT Value*, and *Expected Value* is *Reference Value*,

or

CASE 2: *Measurement* is *Reference Value*, and *Expected Value* is *UUT Value*.

In CASE 1 the *Error* is positive if the *UUT Value* (the *Measurement*) is greater than the *Reference Value* (the *Expected Value*), zero if the two values are the same, and negative otherwise.

In CASE 2 the *Error* is positive if the *Reference Value* (the *Measurement*) is greater than the *UUT Value* (the *Expected Value*), zero if the two values are the same, and negative otherwise.

## Determining the *Test Tolerance*

The *Test Tolerance* of a test is specified in the calibration procedure. In order to calculate *Percent Error* the *Test Tolerance* must be converted to the same form as the *Error* (namely, absolute units). The procedure language implemented by the calibration software may allow the *Test Tolerance* to be represented in different ways. Examples are:

- Percentage of *Nominal*
- PPM of *Nominal*
- Percentage of *Range*
- PPM of *Range*
- Absolute Units (i.e., a floor specification)

Typically the *Test Tolerance* will be represented as an expression involving one or more terms, where each term is in one of the forms listed above. The software must convert each term to base units (for example, volts) and sum the terms to generate a single number which represents the *Test Tolerance*.

A minor complication is that the *Test Tolerance* may be asymmetrical. For example one might have a *Test Tolerance* of “+1 mV –2 mV”. When the *Test Tolerance* is asymmetrical the software must choose either the upper or the lower value when the test evaluation is performed. This choice is typically made based on a comparison of the *UUT Value* and the *Reference Value*. If the *UUT Value* is greater than or equal to the *Reference Value* the upper tolerance value is used as the *Test Tolerance*. Otherwise, the lower tolerance value is used as the *Test Tolerance*.

After *Error* and *Test Tolerance* have both been determined, *Percent Error* can be calculated as:

$$\text{Percent Error} = \frac{|Error * 100|}{\text{TestTolerance}}$$

The sign of the *Error* does not matter in determining *Percent Error*.

When guardbanding is enabled, the test result determination is as described above, but with an intervening step which adjusts the *Test Tolerance* prior to the calculation of *Percent Error*.

## 5. Conclusion

Since there is no consensus at this point on precisely how to apply guardbanding in all situations, an effective automated calibration software solution must provide a flexible and customizable implementation. The software should:

- Allow the user to enable to disable guardbanding.
- Allow the user to specify the guardbanding method.
- Allow the user to specify a table of parameter thresholds with associated guardband factors.
- Allow the user to choose whether or not to interpolate between points in the table.
- Support guardbanding when the *Test Tolerance* is asymmetrical.
- Allow the user to specify how *Indeterminate* results are handled in the determination of the overall result of a calibration verification procedure.
- Provide mechanisms to allow the user (procedure writer) to invoke external programs or retrieve data from external files, where the external programs and data files are used to specify guardbanding factors and parameters.
- Implement configurability for all parameters in a hierarchical fashion where a per-test specification overrides a per-procedure specification, a per-procedure specification overrides a per-workstation specification, and a per-workstation specification overrides a per-site specification.
- Generate result data in a standard SQL database to allow an auditor to verify the guardbanding method, and to allow the user to generate calibration reports that show all necessary data related to guardbanding.
- Implement defaults such that the procedure writer may enable guardbanding without extensive modification of the calibration procedure. In the limiting case it should be possible to enable guardbanding without any modification of a calibration procedure.

The guardbanding approach described in this paper is intended to be flexible and customizable, and to allow most users to easily implement guardbanding in a way which satisfies their organizational requirements. However, as with most software projects, there are opportunities for improvement. Two particular opportunities for enhancement are:

- The table-based method described above, where the independent parameter is determined by the user, and an associated guardband factor is specified for each row in the table, could be made more general by allowing a mathematical expression to determine the guardband factor on each row of the table.
- More work should be done on guardbanding methods that directly maintain a constant risk of accepting an out-of-tolerance UUT.

## 6. References

1. ISO 17025, *General Requirements for the Competence of Testing and Calibration Laboratories*.
2. David Abell, Agilent Corp., *17025 Calibration Definition – Executive Summary, Use of Guard Bands*, 2002
3. ISO/DIS 14253.1 E 1995, *Geometrical Product Specification – Inspection by Measurement of Workpieces and Measuring Instruments*
4. David Deaver, Fluke Corp., *How to Maintain Your Confidence*, 1993

5. David Deaver, Fluke Corp., *Guardbanding with Confidence*, 1994
6. David Deaver, Fluke Corp., *Managing Calibration Confidence in the Real World*, 1996
7. David Deaver, Fluke Corp., *Guardbanding and the World of ISO Guide 25 – Is there Only One Way?*, 1999