COMPREHENSIVE SERVICES

We offer competitive repair and calibration services, as well as easily accessible documentation and free downloadable resources.

SELL YOUR SURPLUS

We buy new, used, decommissioned, and surplus parts from every NI series. We work out the best solution to suit your individual needs.

Sell For Cash Get Credit Receive a Trade-In Deal

OBSOLETE NI HARDWARE IN STOCK & READY TO SHIP

We stock **New, New Surplus, Refurbished,** and **Reconditioned** NI Hardware.

APEX WAVES

Bridging the gap *between the manufacturer and your legacy test system.*

1-800-915-6216

www.apexwaves.com

sales@apexwaves.com

Request a Quote CLICK HERE *PCI-5911*

NI 5911

Introduction

This document contains information and step-by-step instructions for calibrating the NI 5911 digitizer. This calibration procedure is intended for metrology labs. It includes programming instructions for calibrating the NI 5911 using Measurement Studio, LabVIEW, C, or Visual Basic programming environments.

What Is Calibration?

Calibration consists of determining the measurement accuracy of a device and correcting for any measurement error. *Verification* is measuring the performance of a device and comparing the results to the factory specifications of the device. During the factory calibration process, the calibration constants are stored on the EEPROM. These values are loaded from memory and used as needed by the digitizer. The NI 5911 requires two types of calibration: external calibration and self-calibration.

External Calibration

External calibration requires using a high-precision digitizer calibrator to verify and adjust calibration constants. This procedure replaces all calibration constants in the EEPROM and is equivalent to a factory calibration. Because the external calibration procedure changes all EEPROM constants, it invalidates the original National Institute of Standards and Technology (NIST) traceability certificate. If an external calibration is done with an NIST-certified voltage source, a new NIST-traceability certificate can be issued.

Self-Calibration

Self-calibration, or internal calibration, uses a software command and requires no external connections. Self-calibration adjusts a device for use in an environment where external variables, such as temperature, may differ from those in the environment in which the device was last externally calibrated.

Why Should You Calibrate?

The accuracy of electronic components drifts with time and temperature, which can affect measurement accuracy as a device ages. External calibration restores the digitizer to its specified accuracy and ensures that it still meets National Instruments standards.

How Often Should You Externally Calibrate?

The measurement accuracy requirements of your application determine how often you should externally calibrate the NI 5911 digitizer.
NI recommends that you perform a complete calibration at least once every year. You can shorten this interval to 90 days or six months based on the demands of your application.

Test Equipment

Table 1 lists the equipment required for calibrating the NI 5911. If you do not have the recommended instruments, use these specifications to select a substitute calibration standard.

Table 1. Equipment Specifications for NI 5911 Verification and Calibration

Required Equipment	Recommended Equipment	Parameter Measured	Necessary Specifications
Digitizer Calibrator/ Ohmmeter	Fluke 9500B Oscilloscope Calibrator	AC Coupling	sine wave 10–20 Hz ±100 ppm, 1.8 Vpp ±2% into 1 MΩ
		Bandwidth	2% amplitude flatness for leveled sine wave 100 kHz–100 MHz ±50 ppm, 1.5 Vpp ±2% into 50 Ω
		Input Impedance	2-wire resistance accuracy of 0.25% for 1 MΩ measurements
		Timing/RIS	sine wave 10 kHz–10 MHz ±15 ppm, 1.8 Vpp ±2% into 1 MΩ
		Trigger Sensitivity	sine wave 100 kHz–10 MHz ±100 ppm, 90 mVpp ±2% into 1 MΩ

Table 1. Equipment Specifications for NI 5911 Verification and Calibration (Continued)

Required Equipment	Recommended Equipment	Parameter Measured	Necessary Specifications
Calibrator	Fluke 5700 Multifunction Calibrator	External Calibration/ Gain	DC ±90 mV to ±9 V, ±0.005% into 1 MΩ
		External Calibration	sine wave 1 kHz ±100 ppm, 8 Vpp ±1% into 1 MΩ
BNC Cable	—	—	50 Ω
BNC T connector	—	—	50 Ω
50 Ω Terminator	—	—	—
BNC Shorting Cap	—	Vertical Gain/Offset	0 VDC, ±0.025 mV

Test Conditions

Follow these guidelines to optimize the connections and the environment during calibration:

- Keep connections to the NI 5911 short. Long cables and wires act as antennae, picking up extra noise that can affect measurements.
- Use a 50 Ω BNC coaxial cable for all connections to the digitizer.
- Keep relative humidity between 10 and 90%, non-condensing, or consult the digitizer hardware manual for the optimum relative humidity.
- Maintain the temperature between 5 and 40 °C, or consult the digitizer hardware manual for the optimum temperature range.
- To prevent overheating, leave empty PCI slots above and below the slot where the NI 5911 is installed. Ideally, the NI 5911 digitizer that you are calibrating should be the only device installed in your computer during calibration.
- Allow a warm-up time of at least 15 minutes to ensure that the measurement circuitry of the NI 5911 is at a stable operating temperature.

Documentation

This section describes the documentation you need to calibrate the NI 5911 digitizer. In addition to this calibration document, you may need to refer to the following documents:

- *NI 5911 User Manual*
- *Where to Start with Your NI Digitizer*
- *NI-SCOPE Driver Quick Reference Guide*

You can download these documents from the NI Web site at `ni.com/manuals`.

Software

This section describes the software you need to calibrate the NI 5911 digitizer. Unless otherwise specified, calibration functions are C function calls in the NI-SCOPE driver. These function calls are also valid for any compiler capable of calling a 32-bit DLL. Many of the functions use constants defined in the `niScopeCal.h` file. To use these constants, you must include `niScopeCal.h` in your code when you write your calibration procedure.

Calibration requires the latest version of the NI-SCOPE driver on the calibration system. You can download NI-SCOPE from the NI Instrument Driver Network at `ni.com/drivers`. NI-SCOPE supports programming for all NI digitizers using a number of languages, including LabVIEW, Measurement Studio, Microsoft Visual C++, and Microsoft Visual Basic. To install and configure NI-SCOPE, refer to the instructions in *Where to Start with Your NI Digitizer*.

LabVIEW virtual instruments (VIs) are not discussed in this procedure because many LabVIEW VIs have the same names as the NI-SCOPE function calls.

Writing Your Calibration Procedure

NI-SCOPE 2.0 includes all the functions necessary for calibrating NI digitizers. Because calibration support is included in `niScope_32.dll`, you can access it through any compiler capable of calling into a 32-bit DLL. If you use a C compiler, include the `niScopeCal.h` header file, which defines all calibration-specific functions and briefly explains the parameters. With Measurement Studio, the NI-SCOPE function panel `niScopeCal.fp` provides further help on these functions. LabVIEW support is installed in `niScopeCal.llb`, and all calibration functions appear in the function palette. See Table 2 for file locations.

Table 2. Calibration File Locations After Installing NI-SCOPE 2.0 or Later

File Name and Location	Description
`VXIpnp\winnt(Win95)\Bin\niscope_32.dll`	NI-SCOPE driver containing the entire NI-SCOPE API, including calibration functions
`VXIpnp\winnt(Win95)\lib\msc\niscope_32.lib`	NI-SCOPE library containing the entire NI-SCOPE API, including calibration functions
`LabVIEW\examples\instr\niScopeExamples\`	Directory of LabVIEW NI-SCOPE example VIs, including self-calibration; access the calibration examples from the LabVIEW function palette
`LabVIEW\instr.lib\Niscope\Calibrate\niscopeCal.llb`	LabVIEW VI library containing VIs for calling the NI-SCOPE calibration API; access calibration functions from the NI-SCOPE calibration section of the LabVIEW function palette
`VXIpnp\winnt(Win95)\include\niscopeCal.h`	Calibration header file, which you must include in any C program accessing calibration functions; this file automatically includes `niScope.h`, which defines the rest of the NI-SCOPE interface
`VXIpnp\winnt(Win95)\Niscope\Niscope.fp`	CVI function panel file that includes function prototypes and help on using NI-SCOPE in the CVI environment
`VXIpnp\winnt(Win95)\Niscope\niScopeCal.fp`	CVI function panel file that includes external calibration function prototypes and help on using NI-SCOPE in the CVI environment
`VXIpnp\winnt(Win95)\Niscope\Examples\`	Directory of NI-SCOPE examples for CVI, C, Visual C++, and Visual Basic
`VXIpnp\winnt(Win95)\Niscope\Documentation\ni5911cal.pdf`	This document

Self-Calibration

The NI 5911 includes an internal voltage source that is 10 times as accurate as an 8-bit digitizer resolution. Self-calibration uses this internal reference source to do the following:

- Calibrate vertical range and offset for each input range.
- Calibrate AC flatness over the entire bandwidth to within specified tolerances.
- Calibrate analog trigger levels.
- Calibrate the time-to-digital converter (TDC) used for random interleaved sampling (RIS) measurements.

You cannot adjust the internal reference source, but verifying the value of the source using a high-precision DMM provides traceability for the verification procedure. Absolute accuracy is ensured by verifying the internal reference voltage using a digital voltmeter. The verification procedure for the internal reference includes calls to `niScope_CalStart` and `niScope_CalEnd`.

Self-calibrate the digitizer before you externally calibrate. NI-SCOPE includes self-calibration example programs for LabVIEW, Measurement Studio, Visual Basic, Visual C++, and Console C. Table 2 shows the locations of these example programs.

Self-Calibrating the NI 5911

To self-calibrate the NI 5911 digitizer, complete the following steps:

1. Call `niScope_init` to obtain an instrument session handle.
2. Call `niScope_calSelfCalibrate` with option set to `VI_NULL`. The new calibration constants are immediately stored in the self-calibration section of the EEPROM, so you can include this procedure in any application that uses the digitizer.
3. Call `niScope_close` to close the session handle.

External Calibration

External calibration consists of two steps: verification and adjustment. These procedures describe the program necessary for verifying and adjusting digitizer specifications. Refer to the *Test Equipment* section of this document for accuracy requirements of input stimuli for specific test instruments.

Verifying the Performance of the NI 5911

The minimal verification procedure to determine if the NI 5911 requires external calibration involves verifying each vertical offset and vertical gain specification listed in Tables 3 and 4. If any of these tests fail, you should adjust the digitizer as described in the *Adjusting the NI 5911* section.

Note Before you begin an external calibration, self-calibrate the NI 5911 as described in the *Self-Calibrating the NI 5911* section to correct for changes in environmental conditions.

Verifying Vertical Offset

To verify vertical offset, complete the following steps:

1. Short circuit the input of the digitizer with the BNC shorting cap.
2. Call `niScope_ConfigureAcquisition` with **acquisitionType** set to `NISCOPE_VAL_FLEXRES`.
3. Call `niScope_ConfigureVertical` with the following parameters:
   - **channelList** = `0`
   - **range** = The Digitizer Parameter entry from Table 3
   - **offset** = `0.0`
   - **coupling** = `NISCOPE_VAL_DC`
   - **probeAttenuation** = `1.0`
   - **enabled** = `VI_TRUE`
4. Wait 250 ms to allow input stage to settle.
5. Call `niScope_ConfigureHorizontalTiming` with the following parameters:
   - **minSampleRate** = `1,000,000`
   - **minNumPts** = `30,000`
   - **refPosition** = `50.0`
   - **numRecords** = `1`
   - **enforceRealtime** = `VI_TRUE`
6. Call `niScope_InitiateAcquisition`.
7. Call `niScope_FetchMeasurement` with the following parameters:
   - **channelList** = `0`
   - **timeout** = `30.0`
   - **scalarMeasFunction** = `NISCOPE_VAL_VOLTAGE_AVERAGE`
8. Compare the resulting average voltage to the success condition listed in Table 3. If the result is outside the success condition range, this verification has failed.

9. Repeat steps 1 through 8 for each vertical offset entry in Table 3.

You have completed verifying the vertical offset specifications.

Table 3. NI 5911 Vertical Offset Specifications

Name	Digitizer Parameters	Stimulus Parameters	Success Condition
Vertical Offset	**range** = 20.0	Short Circuit Input	$\|x\| < 0.0021$ V
Vertical Offset	**range** = 10.0	Short Circuit Input	$\|x\| < 0.0011$ V
Vertical Offset	**range** = 4.0	Short Circuit Input	$\|x\| < 0.0005$ V
Vertical Offset	**range** = 2.0	Short Circuit Input	$\|x\| < 0.0003$ V
Vertical Offset	**range** = 1.0	Short Circuit Input	$\|x\| < 0.0002$ V
Vertical Offset	**range** = 0.4	Short Circuit Input	$\|x\| < 0.00014$ V
Vertical Offset	**range** = 0.2	Short Circuit Input	$\|x\| < 0.00012$ V

Verifying Vertical Gain

To verify the vertical gain, complete the following steps:

1. Short-circuit the input of the digitizer with the BNC shorting cap.
2. Call `niScope_ConfigureAcquisition` with **acquisitionType** set to `NISCOPE_VAL_FLEXRES`.
3. Call `niScope_ConfigureVertical` with the following parameters:
   - **channelList** = `0`
   - **range** = The Digitizer Parameter entry from Table 4
   - **offset** = `0.0`
   - **coupling** = `NISCOPE_VAL_DC`
   - **probeAttenuation** = `1.0`
   - **enabled** = `VI_TRUE`
4. Wait 250 ms to allow the input stage to settle.
5. Call `niScope_ConfigureHorizontalTiming` with the following parameters:
   - **minSampleRate** = `1,000,000`
   - **minNumPts** = `30,000`
   - **refPosition** = `50.0`
   - **numRecords** = `1`
   - **enforceRealtime** = `VI_TRUE`

6. Call `niScope_InitiateAcquisition`.
7. Call `niScope_FetchMeasurement` with the following parameters:
   - **channelList** = `0`
   - **timeout** = `30.0`
   - **scalarMeasFunction** = `NISCOPE_VAL_VOLTAGE_AVERAGE`

   Record this measurement to use in further calculations.
8. Apply the DC stimulus voltage listed in Table 4.
9. Call `niScope_InitiateAcquisition.`
10. Call `niScope_FetchMeasurement` with the following parameters:
    - **channelList** = `0`
    - **timeout** = `30.0`
    - **scalarMeasFunction** = `NISCOPE_VAL_VOLTAGE_AVERAGE`

    Record this measurement to use in further calculations.
11. Compute the error in the vertical gain using the equation:

$$error = a - b - c$$

    where

    a = the measured stimulus voltage

    b = the measured short-circuit voltage

    c = the applied stimulus voltage
12. Compare the error to the success condition in Table 4. If the error is less than the success condition value, the NI 5911 has passed the vertical gain test.
13. Repeat steps 1 through 12 for each vertical gain entry in Table 4.

You have completed verifying the vertical gain specifications.

Table 4. NI 5911 Vertical Gain Specifications

Name	Digitizer Parameters	Stimulus Parameters	Success Condition
Vertical Gain	**range** = 20.0	+9 VDC	|x| < 0.00452 V
Vertical Gain	**range** = 20.0	–9 VDC	|x| < 0.00452 V
Vertical Gain	**range** = 10.0	+4.5 VDC	|x| < 0.00226 V
Vertical Gain	**range** = 10.0	–4.5 VDC	|x| < 0.00226 V
Vertical Gain	**range** = 4.0	+1.8 VDC	|x| < 0.000904 V
Vertical Gain	**range** = 4.0	–1.8 VDC	|x| < 0.000904 V
Vertical Gain	**range** = 2.0	+0.9 VDC	|x| < 0.000452 V
Vertical Gain	**range** = 2.0	–0.9 VDC	|x| < 0.000452 V
Vertical Gain	**range** = 1.0	+0.45 VDC	|x| < 0.000226 V
Vertical Gain	**range** = 1.0	–0.45 VDC	|x| < 0.000226 V
Vertical Gain	**range** = 0.4	+0.18 VDC	|x| < 0.0000904 V
Vertical Gain	**range** = 0.4	–0.18 VDC	|x| < 0.0000904 V
Vertical Gain	**range** = 0.2	+0.09 VDC	|x| < 0.0000452 V
Vertical Gain	**range** = 0.2	–0.09 VDC	|x| < 0.0000452 V

Verifying Input Impedance

To verify input impedance, complete the following steps:

1. Connect the digitizer input to the ohmmeter with a coaxial cable.
2. Call `niScope_ConfigureVertical` with the following parameters:
   - **channelList** = `0`
   - **range** = `20.0`
   - **offset** = `0.0`
   - **coupling** = `NISCOPE_VAL_DC`
   - **probeAttenuation** = `1.0`
   - **enabled** = `VI_TRUE`
3. Wait 250 ms to allow the input stage to settle.

4. Call `niScope_Read` solely to configure and initiate the hardware. You can discard the output data. Use the following parameters:
   - **channelList** = `0`
   - **timeout** = `1.0` or greater
   - **numSamples** = `128`
5. Measure the impedance (*x*) on the ohmmeter and compare it to the success condition in Table 5. If *x* is outside the given range, the digitizer has failed this test.

You have completed verifying the input impedance.

Verifying Bandwidth

To verify bandwidth, complete the following steps:

1. Connect the BNC T to the digitizer, and connect the signal generator and the 50 Ω terminator to the two inputs of the T connector.
2. Configure the signal generator for a 50 Ω load.
3. Call `niScope_ConfigureAcquisition` with **acquisitionType** set to `NISCOPE_VAL_NORMAL`.
4. Set the signal generator to the frequency and amplitude listed in Table 5 for the reference bandwidth entry.
5. Call `niScope_ConfigureVertical` with the following parameters:
   - **channelList** = `0`
   - **range** = `2.0`
   - **offset** = `0`
   - **coupling** = `NISCOPE_VAL_DC` or `NISCOPE_VAL_AC` (both should be tested)
   - **probeAttenuation** = `1.0`
   - **enabled** = `VI_TRUE`
6. Wait 1.36 s to allow the input stage to settle.
7. Call `niScope_ConfigureHorizontalTiming` with the following parameters:
   - **minSampleRate** = The Digitizer Parameter entry in Table 5
   - **minNumPts** = `30,000`
   - **refPosition** = `50.0`
   - **numRecords** = `1`
   - **enforceRealtime** = `VI_TRUE`
8. Call `niScope_InitiateAcquisition`.
9. Call `niScope_FetchMeasurement` with the following parameters:

- **channelList** = `0`
- **timeout** = `30.0`
- **scalarMeasFunction** = `NISCOPE_VAL_AC_ESTIMATE`

Record this measurement and use it as the *reference AC estimate* in step 14.

10. Set the signal generator to the frequency and amplitude listed in Table 5 for the bandwidth entry.
11. Call `niScope_ConfigureHorizontalTiming` with the following parameters:
    - **minSampleRate** = The Digitizer Parameter entry in Table 5
    - **minNumPts** = `30,000`
    - **refPosition** = `50.0`
    - **numRecords** = `1`
    - **enforceRealTime** = `VI_TRUE`
12. Call `niScope_InitiateAcquisition`.
13. Call `niScope_FetchMeasurement` with the following parameters:
    - **channelList** = `0`
    - **timeout** = `30.0`
    - **scalarMeasFunction** = `NISCOPE_VAL_AC_ESTIMATE`

    Record this measurement and use it as the *AC estimate* in step 14.
14. Compute the response in decibels using the following equation:

$$response = 20\log_{10}\left[\frac{AC\ estimate}{reference\ AC\ estimate}\right]$$

15. Compare response to the success condition in Table 5. If *x* is outside the given range, the digitizer has failed this test.
16. Repeat steps 9 through 15 for the remaining bandwidth entries in Table 5.

You have completed verifying the bandwidth specifications.

Verifying AC Coupling Cutoff Frequency

To verify the AC coupling cutoff frequency, complete the following steps:

1. Connect the BNC cable from the signal generator to the input of the digitizer.
2. Configure the signal generator for a 1 MΩ load.
3. Set the signal generator to the frequency and amplitude listed in Table 5.

4. Call `niScope_ConfigureVertical` with the following parameters:
   - **channelList =** `0`
   - **range** = `2.0`
   - **offset** = `0.0`
   - **coupling** = `NISCOPE_VAL_DC`
   - **probeAttenuation** = `1.0`
   - **enabled** = `VI_TRUE`
5. Wait 250 ms to allow the input stage to settle.
6. Call `niScope_ConfigureHorizontalTiming` with the following parameters:
   - **minSampleRate** = `10,000`
   - **minNumPts** = `10,000`
   - **refPosition** = `50.0`
   - **numRecords** = `1`
   - **enforceRealtime** = `VI_TRUE`
7. Call `niScope_InitiateAcquisition`.
8. Call `niScope_FetchMeasurement` with the following parameters:
   - **channelList =** `0`
   - **timeout** = `30.0`
   - **scalarMeasFunction** = `NISCOPE_VAL_AC_ESTIMATE`

   Record this measurement and use it as the *AC estimate with DC coupling* in step 12.
9. Call `niScope_ConfigureVertical` with the following parameters:
   - **channelList =** `0`
   - **range** = `2.0`
   - **offset** = `0.0`
   - **coupling** = `NISCOPE_VAL_AC`
   - **probeAttenuation** = `1.0`
   - **enabled** = `VI_TRUE`
10. Wait 1.36 s to allow the input stage to settle.
11. Call `niScope_FetchMeasurement` with the following parameters:
   - **channelList** = `0`
   - **timeout** = `30.0`
   - **scalarMeasFunction** = `NISCOPE_VAL_AC_ESTIMATE`

   Record this measurement and use it as the *AC estimate with AC coupling* in step 12.

12. Compute the response in decibels using the following equation:

$$response = 20\log_{10}\left[\frac{AC\ estimate\ with\ AC\ coupling}{AC\ estimate\ with\ DC\ coupling}\right]$$

13. Compare *response* to the success condition in Table 5. If *response* is outside the listed range, the digitizer has a hardware problem. Return the digitizer to NI for repair or replacement.
14. Repeat steps 3 through 13 for each AC coupling entry in Table 5.

You have completed verifying the AC coupling cutoff frequency.

Table 5. NI 5911 Input Impedance, Bandwidth, and AC Coupling Specifications

Name	Digitizer Parameters	Stimulus Parameters	Success Condition
Input Impedance	—	—	$980{,}000 < x < 1{,}020{,}000\ \Omega$
Reference Bandwidth	**minSampleRate** = 20,000,000 S/s	100 kHz, 1.5 Vpp	—
Bandwidth	**minSampleRate** = 100,000,000 S/s	1 MHz, 1.5 Vpp	$\lvert x \rvert < 3$ dB
Bandwidth	**minSampleRate** = 50,000,000 S/s	49 MHz, 1.5 Vpp (intentionally aliased)	$\lvert x \rvert < 3$ dB
Bandwidth	**minSampleRate** = 100,000,000 S/s	99 MHz, 1.5 Vpp (intentionally aliased)	$\lvert x \rvert < 3$ dB
AC Coupling	—	2.6 Hz, 1.8 Vpp	$\lvert x \rvert < 3$ dB
AC Coupling	—	2.0 Hz, 1.8 Vpp	$\lvert x \rvert > 3$ dB

Verifying Timing

To verify timing, complete the following steps:

1. Connect the BNC cable from the signal generator to the input of the digitizer.
2. Configure the signal generator for a 1 MΩ load.
3. Set the signal generator to generate a 10 kHz, 1.8 Vpp sine wave.
4. Call `niScope_ConfigureVertical` with the following parameters:
   - **channelList** = `0`
   - **range** = `2.0`
   - **offset** = `0.0`

- **coupling** = NISCOPE_VAL_DC
- **probeAttenuation** = 1.0
- **enabled** = VI_TRUE

5. Wait 250 ms to allow the input stage to settle.
6. Call niScope_ConfigureHorizontalTiming with the following parameters:
   - **minSampleRate** = 1,000,000
   - **minNumPts** = 100,000
   - **refPosition** = 50.0
   - **numRecords** = 1
   - **enforceRealtime** = VI_TRUE
7. Call niScope_InitiateAcquisition.
8. Call niScope_FetchMeasurement with the following parameters:
   - **channelList** = 0
   - **timeout** = 30.0
   - **scalarMeasFunction** = NISCOPE_VAL_AVERAGE_FREQUENCY

   The returned frequency value must be between 9999 and 10,001 Hz, or a hardware error exists. If the digitizer fails this test, return it to NI for repair.
9. Set the signal generator to generate a 1.8 Vpp, 10 MHz sine wave. This wave is intentionally undersampled, where the sampling rate is an even multiple of the sine wave frequency.
10. Call niScope_InitiateAcquisition.
11. Call niScope_FetchMeasurement with the following parameters:
    - **channelList** = 0
    - **timeout** = 30.0
    - **scalarMeasFunction** = NISCOPE_VAL_AVERAGE_PERIOD

    Record the *period* measurement to use in step 13.
12. If the returned status is NISCOPE_ERROR_UNABLE_TO_PERFORM_MEASUREMENT, call niScope_errorHandler with **errorCode** set to the returned error value. If the timing is perfectly aliased, the waveform is a DC level and the period measurement fails. Therefore, if the error description indicates the measurement failed because of insufficient crosspoints, the digitizer has passed the test.

13. If the returned status is anything other than `NISCOPE_ERROR_UNABLE_TO_PERFORM_MEASUREMENT`, compute the actual sample rate (x), assuming a perfect source, using the following equation:

$$x = \frac{specified\ sample\ rate \times source\ frequency \times period}{source\ frequency \times period - 1}$$

which is:

$$x = \frac{10^{13} \times period}{10^{7} \times period - 1}$$

14. Compare the actual sample rate (x) to the success condition in Table 6. If x is outside the range of the success condition, return the digitizer to NI for repair.

You have completed verifying timing.

Verifying Trigger Sensitivity

To verify trigger sensitivity, you must test the smallest signal on which the digitizer triggers (with default hysteresis) by trying all possible trigger levels. Complete the following steps:

1. Connect the BNC cable from the signal generator to the digitizer input.
2. Configure the signal generator for a 1 MΩ load.
3. Apply a 1 MHz sine wave with zero vertical offset, and 90 mVpp voltage.
4. Call `niScope_ConfigureVertical` with the following parameters:
   - **channelList** = `0`
   - **range** = `2.0`
   - **offset** = `0.0`
   - **coupling** = `NISCOPE_VAL_DC`
   - **probeAttenuation** = `1.0`
   - **enabled** = `VI_TRUE`
5. Wait 250 ms to allow the input stage to settle.

6. Call `niScope_ConfigureHorizontalTiming` with the following parameters:
   - **minSampleRate** = `20,000,000`
   - **minNumPts** = `128`
   - **refPosition** = `50.0`
   - **numRecords** = `1`
   - **enforceRealtime** = `VI_TRUE`
7. Call `niScope_ConfigureTriggerEdge` with the following parameters:
   - **triggerSource** = `0`
   - **level** = calculated trigger level as discussed in step 10
   - **triggerCoupling** = `NI_SCOPE_VAL_AC`
   - **slope** = `NI_SCOPE_VAL_POSITIVE`
   - **holdoff** = `0.0`
   - **delay** = `0.0`
8. Call `niScope_Read` solely to configure and initiate the hardware. You can discard the output data. Use the following parameters:
   - **channelList** = `0`
   - **timeout** = `1.0` or greater
   - **numSamples** = `128`

   If `niScope_Read` returns a maximum time exceeded error, the digitizer did not trigger and has failed this test. Otherwise, the digitizer has passed this test, and you can stop the test.
9. Call `niScope_Abort`.
10. Repeat steps 6 through 9, incrementing the trigger level by the trigger level delta value in Table 6, until the digitizer triggers or until the trigger level is greater than the high trigger level listed in the table. If all trigger levels have been tested and the digitizer has never triggered, a hardware problem exists with the trigger sensitivity. Return the digitizer to NI for repair or replacement.

You have completed verifying the trigger sensitivity.

Verifying Vertical Sensitivity

To verify vertical sensitivity, complete the following steps:

1. Short-circuit the input of the digitizer with the BNC shorting cap.
2. Call `niScope_ConfigureAcquisition` with **acquisitionType** set to the appropriate value in Table 6.
3. Call `niScope_ConfigureVertical` with the following parameters:
   - **channelList =** `0`
   - **range** = The first Digitizer Parameter entry from Table 6
   - **offset** = `0`
   - **coupling** = `NISCOPE_VAL_DC`
   - **probeAttenuation** = `1.0`
   - **enabled** = `VI_TRUE`
4. Wait 250 ms to allow the input stage to settle.
5. Call `niScope_ConfigureHorizontalTiming` with the following parameters:
   - **minSampleRate** = The Digitizer Parameter entry from Table 6
   - **minNumPts** = `50,000`
   - **refPosition** = `50.0`
   - **numRecords** = `1`
   - **enforceRealtime** = `VI_TRUE`
6. Call `niScope_InitiateAcquisition`.
7. Call `niScope_FetchMeasurement` with the following parameters:
   - **channelList** = `0`
   - **timeout** = `30.0`
   - **scalarMeasFunction** = `NISCOPE_VAL_AC_ESTIMATE`

   Record this measurement to use in step 8 as *measured AC estimate*.
8. Calculate the noise referred to input (x) with the following equation:

$$x = 20\log_{10}\left(\frac{measured\ AC\ estimate \times 2 \times \sqrt{2}}{range \times \sqrt{bandwidth}}\right)$$

   where *bandwidth* is the value from Table 6.
9. Compare x to the success condition listed in Table 6. If x is greater than or equal to the success condition, the digitizer has failed this test.
10. Repeat steps 1 through 9 for each vertical sensitivity entry in Table 6.

You have completed verifying the vertical sensitivity.

Verifying RIS Timing

The TDC provides an extremely accurate trigger time resolution between two samples. This trigger should happen with a uniform distribution between two digitizer samples to accurately reconstruct the period signal. To measure this distribution, complete the following steps:

1. Connect the signal generator to the digitizer. This test requires a signal generator that is completely independent of the digitizer. The source cannot be a signal derived from the PCI/PXI chassis or digitizer, and it cannot be the output of a function generator that is synchronized with the digitizer.
2. Configure the signal generator for a 1 MΩ load.
3. Apply the signal from the stimulus parameter column in Table 6.
4. Call `niScope_ConfigureVertical` with the following parameters:
   - **channelList =** `0`
   - **range** = `2.0`
   - **offset** = `0`
   - **coupling** = `NISCOPE_VAL_DC`
   - **probeAttenuation** = `1.0`
   - **enabled** = `VI_TRUE`
5. Wait 250 ms to allow the input stage to settle.
6. Call `niScope_ConfigureHorizontalTiming` with the following parameters:
   - **minSampleRate** = `100,000,000`
   - **minNumPts** = `128`
   - **refPosition** = `50.0`
   - **numRecords** = `1`
   - **enforceRealTime** = `VI_TRUE`
7. Call `niScope_ConfigureTriggerEdge` with the following parameters:
   - **triggerSource** = `0`
   - **level** = `0.0`
   - **triggerCoupling** = `NISCOPE_VAL_DC`
   - **slope** = `NISCOPE_VAL_POSITIVE`
   - **holdoff** = `0.0`
   - **delay** = `0.0`

8. Call `niScope_CalMeasureRISDistribution` with the following parameters:
   - **distributionSize** = `10`
   - **maxTime** = `10,000`
   - **distribution** = A pointer to an array of **distributionSize** number of elements

   If you do not want **distribution** returned, set **distribution** to `VI_NULL`.

   The function `niScope_CalMeasureRISDistribution` acquires 2,000 data points and creates a probability distribution based on the initial *x* value, which includes the TDC value.
9. Compare the returned **minimumBinPercent** (*x*) to the success condition in Table 6. If *x* is outside the range of the success condition, the digitizer has failed this test.

You have completed verifying the RIS distribution.

Table 6. NI 5911 Timing, Trigger Sensitivity, and RIS Distribution Specifications

Name	Digitizer Parameters	Stimulus Parameters	Success Condition
Timing	—	10 kHz, 1.8 Vpp	999,950 < *x* < 1,000,050 Hz
Trigger Sensitivity	low trigger level = –0.9 V high trigger level = 0.9 V trigger level delta = 0.01 V	90 mVpp	digitizer triggers with any valid trigger level
Vertical Sensitivity	**acqType** = NISCOPE_VAL_NORMAL **range** = 20.0 V **minSampleRate** = 100,000,000 S/s **bandwidth** = 100,000,000 Hz	Short Circuit Input	*x* < –120 dBfs/sqrt(Hz)
Vertical Sensitivity	**acqType** = NISCOPE_VAL_FLEXRES **range** = 20.0 V **minSampleRate** = 12,500,000 S/s **bandwidth** = 3.750,000 Hz	Short Circuit Input	*x* < –135 dBfs/sqrt(Hz)
Vertical Sensitivity	**acqType** = NISCOPE_VAL_FLEXRES **range** = 20.0 V **minSampleRate** = 5,000,000 S/s **bandwidth** = 2,000,000 Hz	Short Circuit Input	*x* < –143 dBfs/sqrt(Hz)

Table 6. NI 5911 Timing, Trigger Sensitivity, and RIS Distribution Specifications (Continued)

Name	Digitizer Parameters	Stimulus Parameters	Success Condition
Vertical Sensitivity	**acqType** = NISCOPE_VAL_FLEXRES **range** = 20.0 V **minSampleRate** = 2,500,000 S/s **bandwidth** = 1,000,000 Hz	Short Circuit Input	$x < -152$ dBfs/sqrt(Hz)
Vertical Sensitivity	**acqType** = NISCOPE_VAL_FLEXRES **range** = 0.2 V **minSampleRate** = 1,000,000 S/s **bandwidth** = 400,000 Hz	Short Circuit Input	$x < -128$ dBfs/sqrt(Hz)
RIS Distribution	—	1 MHz, ±100 kHz, 1.8 Vpp	$x > 2.0$

Adjusting the NI 5911

This section explains how to adjust the calibration constants if the digitizer has failed any of the verification tests. The adjustment procedure is password-protected so that users cannot accidentally access or modify calibration constants. The password is initially set to zero (`0`) or the empty string (`""`). You can change the password using the function `niScope_CalChangePassword`.

To adjust the calibration constants of the NI 5911, complete the following steps:

1. Connect the calibrator DC voltage source to the digitizer input using a 50 Ω BNC cable.
2. Apply a 5.001–6.0 VDC voltage to the digitizer input.
3. Call `niScope_CalStart` to obtain a calibration-specific device handle.The default external calibration password is set to zero (`0`) or the empty string (`""`).
4. Adjust the internal reference by calling `niScope_CalAdjustInternalReference` with **stimulus** set to the exact stimulus voltage applied in step 2. This function may take several minutes to complete.
5. Apply a 1 kHz, 8 Vpp sine wave to the input of the digitizer.

6. Call `niScope_CalAdjustRange (handle, 0, 10.0, 8.0)` to adjust the AC gain correction. The vertical range parameter does not matter.
7. Disconnect the source, or apply 0 VDC.
8. Call `niScope_CalSelfCalibrate (handle, 0, VI_NULL)` to self-calibrate the digitizer.
9. Call `niScope_CalEnd` to release the session handle. Set **action** to `NISCOPE_VAL_CAL_ACTION_STORE`, as defined in `niScopeCal.h`, to store the calibration constants in the EEPROM. This action stores the external calibration date and the incremented external calibration count in the EEPROM.

You have completed adjusting the calibration constants of the NI 5911 digitizer. To ensure that the digitizer is using the new calibration constants, repeat the verification procedures described in the *Verifying the Performance of the NI 5911* section.

Calibration Function Reference

This section lists functions specific to NI-SCOPE calibration. Refer to *Where to Start with Your NI Digitizer* for instructions on installing these functions.

Function Prototype

```
ViStatus _VI_FUNC niScope_CalStart
(
    ViRsrc resourceName,
    ViConstString password,
    ViSession *sessionHandle”
);
```

Purpose

niScope_CalStart opens an external calibration session.

Using The niScope_CalStart Function

For additional security, this function compares **password** to the password stored in the EEPROM. By default, the password is set to NULL or an empty string. The password is stored in the EEPROM as an array of 4 bytes. Non-printable characters are allowed, but the array is padded with NULLs after the first NULL is found. This allows strings of less than four characters to be legal passwords.

All calibration functions require a session handle, such as **newSessionHandle,** that is returned by this function. Only the external calibration functions require a calibration session handle to allow password protection. All other functions, such as verification and fetch functions, work with both a calibration session and a session handle obtained from niScope_init. Acceptable session handles are documented for each function in this section.

You can obtain only one session handle at a time, and you must close every session by calling niScope_CalEnd. If you fail to close the session, you must unload the niScope_32.dll by closing your application or development environment before you can open another session.

If an error occurs during calibration, call niScope_errorHandler to get the error message text and niScope_CalEnd with **action** set to NISCOPE_VAL_CAL_ACTION_ABORT to close the session.

Parameters

Name	Description
resourceName	assigned by Measurement and Automation Explorer (MAX); this is a string such as “DAQ::1”
password	compared to password in EEPROM
sessionHandle	returned session handle

Function Prototype

```
ViStatus _VI_FUNC niScope_CalEnd
(
    ViSession sessionHandle
    ViInt32 action
);
```

Purpose

niScope_CalEnd closes an external calibration session.

Using the niScope_CalEnd Function

If **action** is NISCOPE_VAL_CAL_ACTION_ABORT, the session is closed, and the calibration constants are lost. The abort close is necessary when an error occurs during calibration. Some devices write to the EEPROM during calibration, in which case the **abort** parameter restores the EEPROM to its original state. It is, therefore, very important to call niScope_CalEnd each time niScope_CalStart is called, even if an error occurs during calibration.

If **action** is set to NISCOPE_VAL_CAL_ACTION_STORE, the calibration constants are stored in the EEPROM. If you call niScope_CalStoreMiscInfo during the calibration session, the miscellaneous information is stored. Otherwise, the miscellaneous information is set to 0 or the empty string ("") in the EEPROM. The current system date and an incremented external calibration count are automatically stored in the EEPROM.

Parameters

Name	Description
sessionHandle	session handle returned by niScope_CalStart
action	values defined in niScopeCal.h: NISCOPE_VAL_CAL_ACTION_STORE, NISCOPE_VAL_CAL_ACTION_ABORT

niScope_CalChangePassword

Function Prototype

```
ViStatus _VI_FUNC niScope_CalChangePassword
(
    ViSession sessionHandle,
    ViConstString oldPassword,
    ViConstString newPassword
);
```

Purpose

To use niScope_CalChangePassword, you must enter an old password and a new password. The function verifies your old password against the one stored in the EEPROM. If they match, the new password is stored in the EEPROM.

Using the niScope_CalChangePassword Function

The password can be four characters long, but shorter strings are acceptable. Non-printable values are acceptable, but zero is treated as an end-of-string character. If a zero (or end-of-string marker) is detected, zeros are added to the end to make the string four characters long.

By default, the password in the EEPROM is an array of NULLs, or the empty string ("").

Parameters

Name	Description
sessionHandle	session handle returned by niScope_CalStart or niScope_init
oldPassword	value currently stored in EEPROM (factory default is "")
newPassword	new value to store in EEPROM

niScope_CalFetchCount

Function Prototype

```
ViStatus _VI_FUNC niScope_CalFetchCount
(
   ViSession sessionHandle
   ViInt32 whichOne,
   ViInt32 *calibrationCount
);
```

Purpose

`niScope_CalFetchCount` returns the calibration count, which is the number of times the digitizer has been calibrated.

Using the niScope_CalFetchCount Function

whichOne determines whether the count is the self-calibration count or the external calibration count. Possible values are defined in `niScopeCal.h`.

Parameters

Name	Description
sessionHandle	session handle returned by `niScope_CalStart` or `niScope_init`
whichOne	values defined in `niScopeCal.h`: `NISCOPE_VAL_CAL_SELF`, `NISCOPE_VAL_CAL_EXTERNAL`
calibrationCount	returns number of times device has been calibrated

niScope_CalFetchDate

Function Prototype

```
ViStatus _VI_FUNC niScope_CalFetchDate
(
    ViSession sessionHandle
    ViInt32 whichOne,
    ViInt32 *year,
    ViInt32 *month,
    ViInt32 *day
);
```

Purpose

`niScope_CalFetchDate` returns the date of the last self-calibration or external calibration.

Using the niScope_CalFetchDate Function

If you are upgrading to NI-SCOPE 2.0 from an earlier version, the initial calibration dates are incorrect because older versions of NI-SCOPE do not support the date feature.

Parameters

Name	Description
sessionHandle	session handle returned by `niScope_CalStart` or `niScope_init`
whichOne	values defined in `niScopeCal.h`: `NISCOPE_VAL_CAL_SELF`, `NISCOPE_VAL_CAL_EXTERNAL`, or `NISCOPE_VAL_CAL_MANUFACTURE`
year	returned year of last calibration (for example, 2000)
month	returned month of last calibration (1–12)
day	returned day of last calibration (1–31)

niScope_CalFetchMiscInfo

Function Prototype

```
ViStatus _VI_FUNC niScope_CalFetchMiscInfo
(
    ViSession sessionHandle,
    ViChar *info
);
```

Purpose

`niScope_CalFetchMiscInfo` returns the miscellaneous information you can store during an external calibration using `niScope_StoreMiscInfo`.

Using This Function

info must be a character array of length five. The last byte is always set to `NULL` to terminate the string.

Parameters

Name	Description
sessionHandle	session handle returned by `niScope_CalStart` or `niScope_init`
info	array of 5 bytes (4 bytes of information plus 1 `NULL` byte) stored in EEPROM during last external calibration

niScope_CalStoreMiscInfo

Function Prototype

```
ViStatus _VI_FUNC niScope_CalStoreMiscInfo
(
    ViSession sessionHandle,
    ViConstString info
);
```

Purpose

niScope_CalStoreMiscInfo stores miscellaneous information in the EEPROM during external calibration. For example, you can store an operator ID for the person or company performing the calibration.

Using This Function

If you are not calling this function during an external calibration, set the miscellaneous information to NULL in the EEPROM. This setting ensures a consistent calibration date, count, and miscellaneous information values in the EEPROM. Four bytes are stored in the EEPROM, and non-printable characters are valid. However, NULL is treated as an end-of-string marker, and all bytes following the first NULL are set to NULL.

Parameters

Name	Description
sessionHandle	session handle returned by niScope_CalStart
info	array of 4 bytes of info stored in EEPROM during last external calibration

niScope_CalSelfCalibrate

Function Prototype

```
ViStatus _VI_FUNC niScope_CalSelfCalibrate
(
    ViSession sessionHandle,
    ViConstString channelName,
    ViInt32 option
);
```

Purpose

`niScope_CalSelfCalibrate` performs a self-calibration.

Using This Function

If the self-calibration is successful, the calibration constants are immediately stored in the self-calibration area of the EEPROM, along with the self-calibration date and incremented count. The only valid value for **option** is `NISCOPE_VAL_CAL_RESTORE_EXTERNAL_CALIBRATION`, which is equivalent to calling the outdated `niScope_Calibrate` with **calibrationOperation** set to `NISCOPE_VAL_RESTORE_FACTORY_CALIBRATION`. You should use this operation only if the self-calibration routine fails and you must use the digitizer rather than return it for repair. This operation restores the previous external calibration constants. Restoring the external calibration constants does not correct for environmental conditions, so the digitizer will not be in an optimal calibration state.

Parameters

Name	Description
sessionHandle	session handle returned by `niScope_CalStart` or `niScope_init`
channelName	this parameter is ignored; use `VI_NULL`
option	only `NISCOPE_VAL_CAL_RESTORE_EXTERNAL_CALIBRATION` is supported; use `VI_NULL` for a normal self-calibration operation

niScope_CalAdjustRange

Function Prototype

```
ViStatus_VI_FUNC niScope_CalAdjustRange
(
    ViSession sessionHandle,
    ViConstString channelName,
    ViReal64 range,
    ViReal64 stimulus
);
```

Purpose

`niScope_CalAdjustRange` performs an AC gain adjustment for a specified channel. `niScope_CalAdjustRange` should only be used when following the procedure in the *Adjusting the NI 5911* section of this document.

Parameters

Name	Description
sessionHandle	session handle returned by `niScope_CalStart`
channelName	string name of channel; for example, `0`
range	the voltage range to verify or adjust
stimulus	the peak-to-peak voltage of the applied signal; see the *External Calibration* section for the proper stimulus to apply

niScope_CalAdjustInternalReference

Function Prototype

```
ViStatus _VI_FUNC niScope_CalAdjustInternalReference
(
    ViSession sessionHandle,
    ViInt32 option,
    ViReal64 stimulus
);
```

Purpose

`niScope_CalAdjustInternalReference` performs an external calibration of the NI 5911 internal reference. This calibration function uses the external reference source instead of the internal reference source to calibrate the vertical range and offset. It then measures the internal reference voltage and stores this value for use during subsequent self-calibrations.

Using This Function

Use this function only when following the external calibration procedure in this document.

Parameters

Name	Description
sessionHandle	session handle returned by `niScope_CalStart`
option	no options are supported yet; use `VI_NULL`
stimulus	the applied stimulus voltage

niScope_CalMeasureRISDistribution

Function Prototype

```
ViStatus _VI_FUNC niScope_CalMeasureRISDistribution
(
    ViSession sessionHandle,
    ViConstString channelName,
    ViInt32 maxTime,
    ViReal64 *minimumBinPercent,
    ViInt32 distributionSize,
    ViInt32 *distribution
);
```

Purpose

niScope_CalMeasureRISDistribution calls niScope_Read 2000 times to acquire data from the specified channel and retrieve the initial *x* value, which includes the TDC.

Using This Function

The TDC should be a uniform distribution between two sample points because triggers should occur randomly. To test this uniformity, the distribution of initial *x* values is created. The percentage of triggers in the smallest bin of this distribution is returned for comparison to a specification to determine if RIS is operating correctly.

The distribution parameter must be declared as an array of **distributionSize** to return the distribution. Optionally, setting **distribution** to VI_NULL specifies that the distribution is not returned.

Parameters

Name	Description
sessionHandle	session handle returned by niScope_CalStart or niScope_init
channelName	string name of channel for which to store the internal reference value, for example, 0 or 1
maxTime	the **maxTime** parameter to niScope_Read, this is the maximum number of milliseconds each acquisition can take
minimumBinPercent	percent of triggers (0.0–100.0) that fall in the smallest bin
distributionSize	number of bins to use for the distribution of initial *x* values
distribution	array of **distributionSize** for the returned distribution, or VI_NULL if the distribution should not be returned